

UCSF

UC San Francisco Electronic Theses and Dissertations

Title

Informatics support for human membrane transporter pharmacogenics studies

Permalink

<https://escholarship.org/uc/item/5zz4r300>

Author

Yan, Qing,

Publication Date

2001

Peer reviewed|Thesis/dissertation

INFORMATICS SUPPORT FOR HUMAN MEMBRANE TRANSPORTER PHARMACOGENOMICS STUDIES

by

Qing Yan

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

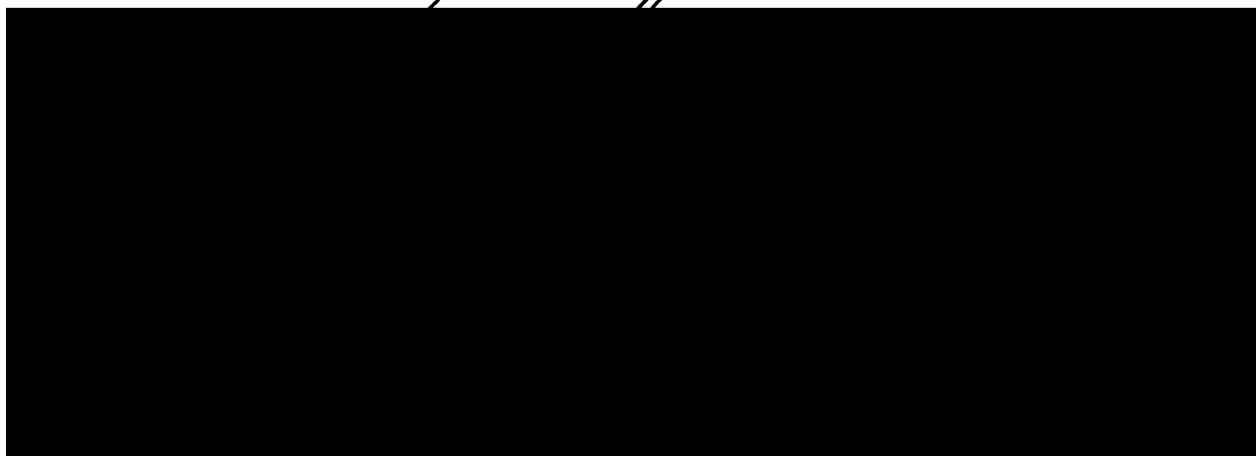
BIOLOGICAL AND MEDICAL INFORMATICS

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA SAN FRANCISCO



Date

University Librarian

Degree Conferred:

Copyright (2001)

by

Qing Yan

ACKNOWLEDGEMENT

I would like to thank my research advisor Dr. Wolfgang Sadee for giving me the opportunity to do the work, for his kind help, and for his support to make this work possible.

I would like to thank Dr. Betty-ann Honer for helping me edit the dissertation, for her kindness, and for her time sitting on my dissertation committee.

I would like to thank Dr. Eric Goldman for his time sitting on my dissertation committee.

I would like to thank Dr. Suzanne Bakken for her kind help and for her time sitting on my qualifying exam committee.

I would like to thank my brother Yong Yan for his unconditional support and encouragement, and for his influence on my interest in science and technology.

I would like to thank my parents for all the years.

**INFORMATICS SUPPORT FOR HUMAN MEMBRANE TRANSPORTER
PHARMACOGENOMICS STUDIES**

Qing Yan

ABSTRACT

My research has been trying to solve some fundamental problems in two major biomedical informatics subjects. One is the knowledge domain of pharmacogenomics, applied specifically to human membrane transporters. This is a relatively new area that lacks informatics studies and available methodologies. The multidisciplinary characteristics of this area have barriers between disciplines for informatics support. The huge amount and the heterogeneous nature of the transporter data demand serious effort of organization to support drug targeting and microarray studies.

The other domain is bioinformatics itself. In this area, lack of common literacy has been recognized as one of the most important obstacles in bioinformatics development.

A series of methodologies have been developed to solve these problems. First of all, three correlations, structure-function, gene-drug, and genotype-phenotype, were found as fundamental themes for understanding pharmacogenomics. The analysis of these correlations clarifies the complexity of some problems and defines the domain boundary.

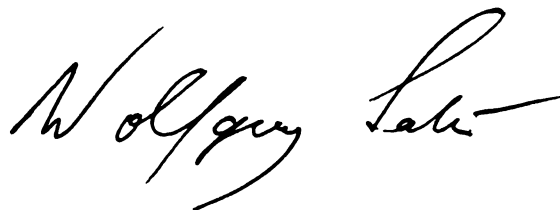
With the correlations as major targets, data modeling was attempted to lay the ground for further data analysis and system implementation. To this end, a three-step methodology has been developed to create an object-oriented (OO) model for

pharmacogenomics. This methodology was designed to overcome the barriers in different disciplines and to establish a standard methodology in pharmacogenomics informatics studies.

A database powered decision support system (DSS) was developed to implement this OO model. This DSS has a three-tiered client-server architecture and is accessible from the Web to provide broad applications. The system user interface was designed with domain expertise to make its usage easier and more efficient.

Because of the heterogeneous nature of transporter data, specific methodologies for data collection and integration were developed to support pharmacogenomics studies. The data integration process addressed some special features to provide the most complete information about transporters, including classification and terminology.

The methodologies developed here also provide generic solutions for problems in the bioinformatics domain. The successful implementation of the system demonstrated that applying unified modeling language (UML) to biomedicine is a generic bioinformatics methodology. For the first time, this work recognized bioinformatics design patterns that can be reused for building more bioinformatics systems.

A handwritten signature in black ink, appearing to read "Wolfgang Salz". The signature is written in a cursive style with a long horizontal stroke at the end.

AC

AB

TA

LIS

LIS

LIS

PR

CH

DF

11

11

11

11

12

12

12

12

12

12

12

12

TABLE OF CONTENTS

ACKNOWLEDGEMENT	iii
ABSTRACT	iv
TABLE OF CONTENTS	vi
LIST OF TABLES	x
LIST OF FIGURES	xi
LIST OF ABBREVIATIONS	xiii
PREFACE	xvi
CHAPTER 1 BUILDING A TRANSPORTER PHARMACOGENOMICS INFORMATICS SUPPORT SYSTEM: BACKGROUND AND PROCESSES	1
1.1 Transporter pharmacogenomics studies: an overview	1
1.1.1 Human membrane transporters: essential for normal physiological functions	1
1.1.2 Pharmaceutical relevance of transporters	3
1.1.3 Pharmacogenomics studies in transporters	6
1.2 Informatics support in transporter pharmacogenomics studies	9
1.2.1 Bioinformatics: advances and challenges	9
1.2.2 Informatics support in pharmacogenomics	14
1.2.3 The process of building the informatics support system	17
1.2.4 Object-oriented technology for system design	20
1.2.5 Unified Modeling Language	22
1.2.6 Design patterns and reusable software	24
1.2.7 System implementation: database management system	27

1.2.8	Data representation and user interface to facilitate decision support	30
1.3	Summary	31
CHAPTER 2 SYSTEM ANALYSIS AND DESIGN: OBJECT-ORIENTED DATA MODELING AND DESIGN PATTERNS		
2.1	Object-oriented analysis and design: the methodology	33
2.1.1	The methodology of design	33
2.1.2	Unified modeling language: what it is and how it is used	38
2.1.2.1	Use case diagrams	38
2.1.2.2	Concept abstraction and class diagrams	39
2.1.2.3	Sequence diagrams	42
2.2	Object-oriented data modeling of pharmacogenomics studies in transporters	43
2.2.1	Key problems in transporter pharmacogenomics studies: domain requirement analysis	44
2.2.1.1	The structure-function relationship	44
2.2.1.2	The genotype-phenotype correlation	46
2.2.1.3	The gene-drug interaction and the comprehensive view	48
2.2.2	Biomedical models	51
2.2.3	The use case diagram	55
2.2.4	Class diagrams	59
2.2.4.1	Concept abstraction	59
2.2.4.2	Data modeling of the structure-function correlation	60
2.2.4.3	Data modeling of the gene-drug interaction	65

2.2.4.4	Data modeling of the genotype-phenotype correlation	68
2.2.4.5	Biomedical design patterns	69
2.2.5	The sequence and a real world example	70
2.3	Summary	73
CHAPTER 3 THE DEVELOPEMNT OF TRANSPORTER PHARMACOGENOMICS		
DECISION SUPPORT SYSTEM (TPDS)		76
3.1	The system architecture for the web application of TPDS	79
3.2	The logical and physical data processing in TPDS	80
3.3	Java coding of data definition	81
3.4	Physical database development	82
3.4.1	Database design	82
3.4.2	Creating the database with SQL	89
3.5	Data sources and collection	90
3.6	Data integration	96
3.6.1	From data to information	96
3.6.2	Transporter classification	100
3.6.3	Transporter nomenclature and terminology	104
3.7	Summary	107
CHAPTER 4 TPDS APPLICATION		109
4.1	Data access and query execution	109
4.1.1	The data access procedure	109
4.1.2	Queries through SQL	111
4.2	The user interface of TPDS	112

4.2.1	The iterative design approach	112
4.2.2	The development of the user interface	114
4.3	Data presentation and data application samples	120
4.4	Summary	128
CHAPTER 5 CONCLUSION		130
5.1	The iterative life cycle of prototyping TPDS	130
5.2	An interdisciplinary approach where biomedicine meets computer science	134
5.3	Methodology provides the solvent	135
5.4	Expanding the application	137
REFERENCES		139

LIST OF TABLES

Table 1.1	A brief overview of categories of some online public biomedical databases	12
Table 1.2	The summary of the most important problems for decision support in transporter pharmacogenomics studies	32
Table 2.1	The Summary of UML Diagrams	37
Table 2.2	The Biomedical Concept Category List	60
Table 3.1	Sample data sources of TPDS	93
Table 3.2	A sample list of transporter families and members according to the TC system	103
Table 3.3	Sample transporters with at least five symbols for one gene	106
Table 4.1	Sample list of chromosome locations of transporters in the ABC superfamily	123
Table 4.2	Sample list of transporters in tissues liver, kidney, intestines, and brain	125
Table 4.3	Sample list of transporter-related diseases	126
Table 5.1	The most important methodologies used in the development of TPDS	137

LIST OF FIGURES

Fig. 1.1	The multi-disciplinary characteristic of pharmacogenomics: interlinked and overlapping knowledge domains	15
Fig. 1.2	Building the DSS: the overall process	18
Fig. 2.1	The process of system analysis and design	35
Fig. 2.2	The process of concept abstraction and class diagram creation	41
Fig. 2.3	The correlation between genotype and phenotype	47
Fig. 2.4	The comprehensive diagram of the three correlations: structure-function, gene-drug, genotype-phenotype	51
Fig. 2.5	The biomedical model of correlations of transporters at the molecular level	53
Fig. 2.6	The biomedical model of correlations of transporters at the systemic level	55
Fig. 2.7	The use case diagram of the Transporter Pharmacogenomics Decision Support System	57
Fig. 2.8	The class diagram of the structure-function correlation	62
Fig. 2.9	The class diagram of gene-drug interaction	66
Fig. 2.10	The sequence diagram of the scenario “identifying candidate genetic markers in tamoxifen resistance”	71
Fig. 3.1	The three-tiered client-server architecture of TPDS	79
Fig. 3.2	The three-layer logical process of data transformation in TPDS	80
Fig. 3.3	The data architecture of TPDS	81
Fig. 3.4	Sample Java code for data definition in TPDS	86

Fig. 3.5	The ER diagram of TPDS	88
Fig. 3.6	Sample SQL statements for the database creation and update	90
Fig. 3.7	Data is available from different sources	91
Fig. 3.8	Finding the “best” data sources	92
Fig. 3.9	The methodology of searching for transporter genes	95
Fig. 3.10	The process of transforming data into information	98
Fig. 3.11	An example of the TC number	102
Fig. 4.1	The structure of data access components in TPDS	110
Fig. 4.2	Sample SQL code for various queries in TPDS	111
Fig. 4.3	The iterative approach of the user interface (UI) design of the TPDS prototype	113
Fig. 4.4	The original page of the system prototype	118
Fig. 4.5	The revised page with improved functions	119
Fig. 4.6	The help file	120
Fig. 4.7	The sample result of a query “What is the polymorphism of TAP1?”	122
Fig. 5.1	The iterative life cycle of prototyping TPDS	132
Fig. 5.2	The time and efforts in each phase spent on the development of the TPDS prototype	133
Fig. 5.3	Disciplines in both biomedicine and computer science involved in TPDS development	135

LIST OF ABBREVIATIONS

5-HTT	serotonin transporter
ABC	ATP-binding cassette
ACE	angiotensin-converting enzyme
AD	Alzheimer's disease
API	application programming interface
AUC	area under the plasma drug concentration-time curve
BCRP	breast cancer resistance proteins
BLAST	Basic Local Alignment Search Tool
BLOSUM	block substitution matrix
CFTR	cystic fibrosis transmembrane conductance regulator
CGI	Common Gateway Interface
CLUSTAL W	A multiple alignment tool
CNS	central nervous system
CRC	Classes, Responsibilities, and Collaborators
DAT	dopamine transporters
DBMS	database management system
dbSNP	database of single nucleotide polymorphisms
DSS	decision support system
EC	Enzyme Commission
EER	extended entity-relationship
G6PD	glucose 6-phosphate dehydrogenase

GABA	gamma-aminobutyric acid
GDB	Genome Database
GUI	graphical user interface
HMTD	Human Membrane Transporter Database
HTML	Hyper Text Markup Language
HUGO	Human Genome Organization
IE	Internet Explorer
IKCa1	intermediate conductance Ca ²⁺ -activated K ⁺ channel
IP	Internet Protocol
ISO	International Organization for Standardization
IT	information technology
JDBC	Java Database Connection
MDR	multiple drug resistance
MRP	multi-drug-resistance protein
NAT2	N-acetyltransferase 2
NBD	nucleotide-binding domains
NCBI	National Center for Biotechnology Institute
NCR	Nucleic Acid Research
OMIM	Online Mendelian Inheritance in Man
OMG	Object Management Group
OMT	Object Modeling Technique
OO	object-oriented
PAM	point accepted mutation

PDB	Protein Data Bank
P-gp	P-glycoprotein
SGLT1	Na⁺-dependent glucose transporter
SNP	single nucleotide polymorphisms
SQL	structured query language
TC	Transport Commission
TMD	transmembrane domain
TPDS	transporter pharmacogenomics decision support system
UML	unified modeling language
URL	Uniform Resource Locator
WWW	World Wide Web

PREFACE

"I see Mr. Nedry has spotted the next phase of our work," Wu said. "How we identify the DNA we have extracted. For that, we use powerful computers."

■ *Jurassic Park, by Michael Crichton*

Bioinformatics has a history of less than thirty years. In such a short period, it has become indispensable for the whole biomedical science. With the development of the human genome project and molecular medicine, bioinformatics was truly flourishing only in the recent five to six years. The impact of this new discipline lies not only in its techniques and algorithms, but also in its methodologies, which is changing the way we think in biomedicine.

Methodology has always been a part of the development of science. Modern science evolved into what it is now by the work on human reasoning by philosophers such as Rene Descartes (deductive reasoning) and Francis Bacon (inductive method). In the past two centuries, great changes have taken place in medical thinking as the ascendancy of scientific method. The importance of scientific methodology itself has gained more recognition in medicine. Studies have shown that the delivery of care is better in hospitals where scientific findings are better translated into clinical practice (Chen et al., 1999). However, this translation has been difficult to accomplish in the current structure of biomedical information.

Today's scientific knowledge (such as molecular genetics) has become so advanced that its relation with phenotypic characteristics is difficult to understand in

clinical studies (David and Evans, 1993). The mapping of molecular genetics to the drugs effects requires painstaking work. Medical practitioners are swamped by the large amount of data and literature in which the knowledge for good practice is hidden and ignored. The difficulties are also caused by the translation of the information and the communication among experts in different areas. To provide high-quality care to patients demands to break the traditional boundaries and to collaborate and share resources.

Consolidation and management of the information across the continuum of biomedical research areas have become one of the biggest challenges. The bridge across the gap between science and medical practice is crucial for a better medicine. Modern medicine has evolved from treating isolated diseases to the understanding at a complex system level. New and better treatments do not come from ad hoc experimental discoveries or isolated intellectual theories, but from a greater system of knowledge. Pharmacogenomics is an emerging knowledge area based on this system level view of human body, with further recognition of individual differences in the whole population.

Different from traditional biomedical sciences that are grounded in the observation of the physical world, bioinformatics is the rational study (at an abstract level) of the way we think about biomedicine, the way we understand the biomedical facts, and the way we apply the biomedical knowledge. Methodology study is a crucial part of bioinformatics in that it is to find ways to meet the challenges and bridge the gaps. One of the reasons in the battle between empiricism and theorization in medicine

is the lack of clear reasoning on both sides with too much unknown and unexplainable. Bioinformatics methodology studies should provide the methods to help open these “black boxes”, expose the unknown, and link the evidence with their theoretical foundations.

For example, a system based on bioinformatics methodologies can embrace the knowledge from different disciplines, trace the processes of treatments, and support the decision making for an individualized optimal prescription. The usage of such a computer support system in clinics can influence the thinking process at every stage in clinical practice, from physical examination to diagnosis, from prescription to outcome analysis. It can also support the thinking process in pharmaceutical industry. In this envision, scientific findings can then be *really* translated into clinical practice, and even revolutionize the way of medical practice.

Such a decision support system may sound ambitious, but it is achievable if bioinformaticians can find good methodologies.

Chapter 1. Building An Informatics Support System for Transporter

Pharmacogenomics Studies: Backgrounds and Processes

The second was to divide up each of the difficulties which I examined into as many parts as possible, and as seemed requisite in order that it might be resolved in the best manner possible.

—*Discourse on the Method, Rene Descartes, 1637*

1.1 Transporter Pharmacogenomics Studies: An Overview

1.1.1 Human Membrane Transporters: Essential for Normal Physiological Functions

Human membrane transporters are essential in normal physiological processes. Transporters are proteins that span the lipid bilayer and form a transmembrane channel lined with hydrophilic amino acid side chains. Membrane transporters play important roles in the maintenance of electrochemical potentials, uptake of nutrients, removal of wastes, endocytotic internalization of macromolecules, and oxygen transport in respiration (Lehninger, 1993; Lodish et al, 1995).

Substrates of transporters move across the lipid bilayer through the transmembrane channels and increase the rate of transmembrane passage. Multiple α -helices constitute transmembrane domains (TMD) which form the secondary structure of transporters. During the process of solute translocation across the membrane, transporters undergo conformational changes.

About one-third of the proteins of a cell are embedded in biological membranes and about one-third of these proteins function to catalyze the transport of molecules across the membrane (Paulsen et al., 1998a, b). Based on mechanisms and energetics, membrane transporters can be categorized into two broad classes, passive transporters and active transporters. Passive transporters include ion channels, such as the Na^+ channel, and facilitated diffusion, such as glucose transporter. Primary active transporters, such as H^+ -ATPase and Na^+K^+ ATPase, make use of ATP, light, or substrate oxidation as energy resources. Secondary active transporters, such as Na^+ /amino acid symporters and H^+ /peptide transporters, use ion gradients as their energy source. Many primary active transporters contain an ATP-binding cassette (ABC) and belong to ABC super family that comprises proteins with very diverse functions (Higgins, 1992). More than fifty transporters have been identified in this super family, including the TAP1, 2 proteins that are involved in MHC-class antigenic peptide transport, and multi-drug resistance proteins (p-glycoprotein).

In addition to transport mode and energy coupling, phylogenetic grouping reveals structure, function, mechanism, and substrate specificity, providing a reliable secondary basis for classification (Saier, 2000). The tertiary basis for classification is substrate specificity and polarity of transport, which are more readily altered during evolutionary history.

The importance of human membrane transporters is demonstrated by some of the disorders in different systems of the human body, caused by transporter malfunction. For example, glucose-galactose malabsorption characterized by severe diarrhea, is associated with defects in the Na^+ -dependent glucose transporter (SGLT1)

(Martin, 1997). The loss of the transporters for Lys, Arg, and Cys from intestinal and renal brush borders can cause cystinuria and kidney stones. Mutations in transporter protein SLC3A1 (also known as rBAT) has been determined to be the cause of type I cystinuria (Palacin et al., 2000). The genetic disease cystic fibrosis is caused by dysfunction of cystic fibrosis transmembrane conductance regulator (CFTR) (Sheppard and Welsh, 1999).

Genetic polymorphisms can also cause physiological disorders. Polymorphisms are allelic variants in genes that exist stably in the population, typically with an allele frequency above 1%. Polymorphism within the promoter region of the serotonin transporter gene (5-HTT) is considered as a potential genetic risk factor for Alzheimer's disease (AD) (Hu et al, 2000). Polymorphisms of the dopamine transporter (DAT) and N-acetyltransferase 2 (NAT2) have been found to have significant association with Parkinson's disease (Le Couteur et al., 1997; Kim et al., 2000; Tan et al., 2000).

With the completion of the sequencing of the entire human genome and the annotation of the sequences, it will be possible to catalog all transporter genes. Other features of transporters, including tissue distribution and functions, as well as their sequence variants, can also be analyzed systematically. However, it will not be possible to do this without the assistance of new information technologies, because of the large number of genes, the tremendous structural and functional heterogeneity of the transporters, and complex associations between them.

1.1.2 Pharmaceutical Relevance of Transporters

Membrane transporters are also vital in drug therapeutic effects. Transporters play important roles in the absorption of oral medications across the gastrointestinal tract. For example, dipeptide transporters are H⁺-coupled, energy-dependent transporters. These transporters are crucial in the oral absorption of beta-lactam antibiotics, angiotensin-converting enzyme (ACE) inhibitors, renin inhibitors, and an anti-tumor drug, bestatin (Lee, 2000).

Membrane transporters also affect drug distribution. Nucleosides and their analogs including antivirals and antineoplastics depend on specific transporters to reach their target sites. Transporters for amino acids, monocarboxylic acids, organic cations, hexoses, nucleosides, and peptides are involved in drug transformation across the blood-brain barrier (Tamai and Tsuji, 2000). Without these transporters, hydrophilic compounds cannot cross the barriers. Recently, regulating the activity of efflux transporters has been suggested to improve the brain entry of certain substrates (Sugiyama et al, 1999). In addition to drug entrance, membrane transporters are also crucial for drug exit from the body. For example, diverse secretory and absorptive transporters in the renal tubule enable renal disposition of drugs (Inui et al., 2000).

The development of the biology of transporters is of particular pharmaceutical relevance (Sadée, 1995). The development of drugs that targets transporters may improve drug therapeutic effects such as oral absorption. For instance, the intestinal peptide transporter can be used to increase the bioavailability of several classes of peptidomimetic drugs, especially ACE inhibitors and beta-lactam antibiotics (Oh, 1999).

The bioavailability of poorly absorbed drugs can be improved by the transporters that are responsible for the intestinal absorption of various solutes and/or by inhibiting the transporters involved in the efflux system. Structural modification and specific transporter targeting are considered promising strategies for drug design with increased bioavailability and tissue distribution. Currently an approach is being explored to enhance therapeutic efficacy, by pharmacological modulation of P-glycoprotein (P-gp) function to improve drug bioavailability to the body and the drug target (Silverman, 1999).

The study of membrane transporters may even result in breakthroughs in the discovery of new drugs. The antiepileptic drug tiagabine, a gamma-aminobutyric acid (GABA) uptake inhibitor, came from the research on amino acid transporters (Iversen, 2000). Neurotransmitter transporters are suggested to be the “fruitful targets” for central nervous system (CNS) drug discovery. In addition, multiple drug resistance (MDR) genes, which are implicated in native and acquired resistance to antineoplastic agents, have drawn intense interest (Rund et al., 1999; Gerlach et al., 1986; Dalton et al., 1991).

Although the role of membrane transporters in drug effects has attracted much recent interest, for most drugs the relevant transporters are still unknown. In some cases a transporter is known to interact with a drug. However it is uncertain whether there are still some other transporters that recognize the same drug with unknown interactions. Therefore, a primary goal of current research in drug discovery and development is to fully understand the interactions between transporters and drugs. That is, which

transporters recognize a drug candidate, and which transporters can be utilized for targeting the drug to its site of action.

The newly developed microarray technology enables the analysis of gene expression patterns for thousands of genes simultaneously and may expedite the application of pharmacogenomics in clinical practice (Kennedy 2000; Sadee 1998; Shi et al., 1999; March 2000). Using the microarray technology for the analysis of membrane transporters may help establish a systematic view of the interactions between transporters and drugs. For example, microarray technology can be used to examine transporter gene expressions in compound-treated cell lines. Obviously, the whole process of doing such experiments, from preparation to data analysis, will heavily rely on information support. With powerful information technology, data obtained from microarray experiments can provide not only information for known transporter-drug interaction, but also grounds for predicting the roles of new drugs.

1.1.3 Pharmacogenomics Studies in Transporters

With the impending identification of most human genes, molecular biology is moving from the structural phase toward the functional phase (Kennedy, 2000). As an emerging scientific discipline, pharmacogenomics is translating functional genomics into clinical medicine (Evans and Relling, 1999). Pharmacogenomics studies the genetic basis of the individual variations in response to drug therapy (Shi et al, 1999). It involves the analysis of gene expression variations related to drug response.

The term “pharmacogenomics” has been used interchangeably with “pharmacogenetics”. The history of pharmacogenetics can be traced back to Pythagoras in Croton, southern Italy 510 B.C. He recognized the “danger of some, but not other, individuals who eat the fava bean.” This danger was hemolytic anemia in people lacking glucose 6-phosphate dehydrogenase (G6PD).

In the 1950s, a series of clinical observations of inherited differences in drug effects gave rise to the area of “pharmacogenetics”, a term first introduced by Friedrich Vogel (Vogel, 1959) in 1959.

These observations include hemolysis after antimalarial therapy and the inherited level of erythrocyte G6PD activity (Carson et al., 1956). At that time, this field mostly was concentrated on genetic polymorphisms in drug-metabolizing enzymes, and how the differences affect drug effects (Nebert, 1997). Today, people use “pharmacogenomics” to represent the entire spectrum of genes that determine the drug behavior and sensitivity, although the two words are used with similar meanings in most occasions.

In the therapeutic context, pharmacogenomics can establish the correlation between specific genotypes and certain phenotypes. Such analysis may be useful in diagnosis and predicting drug response at any stage in the clinic (Emilien et al., 2000). The development of pharmacogenomics can have great impact on every phase of biomedicine, from clinical laboratory tests to personalized (or individually tailored) medicine (Hess and Cooper, 1999; Roses 2000; March, 2000; Emilien et al., 2000).

Pharmacogenomics studies in transporter genes may contribute significantly to our understanding of interindividual variability to numerous therapeutic agents. For

example, polymorphisms of the dopamine transporter gene (DAT1) have been found to play a role in response to methylphenidate, which was used to treat attention-deficit hyperactivity disorder in children (Winsberg, 2000).

Sequence variations in a transporter P-glycoprotein (P-gp) may have functional importance for drug absorption and elimination, as well as clinical relevance to drug resistance response. A significant correlation has been observed between a polymorphism in exon 26 (C3435T) of MDR-1 and the expression levels and function of MDR-1 (Hoffmeyer, 2000). Individuals homozygous for this polymorphism were found to have significantly reduced duodenal MDR-1 expression and increased digoxin plasma levels. This polymorphism was suggested to influence the absorption and tissue concentrations of other substrates of MDR-1.

Another example of polymorphism with impact on drug efficacy is in the serotonin transporter (5-HTT). Serotonin (5-HT) is a neurotransmitter that plays important roles in many physiological processes, whose disorder may cause severe depression. 5-HTT is critical in the termination of serotonin neurotransmission. This transporter is the target for selective serotonin-reuptake inhibitors. A functional polymorphism in the transcriptional control region upstream of the coding sequence of 5-HTT has been reported (Lesch et al., 1996). It has been observed that this polymorphism influences the antidepressant response to antidepressants fluvoxamine and paroxetine (Smeradi et al., 1998; Kim et al., 2000; Pollock et al., 2000; Zanardi et al., 2000).

The polymorphic characteristics of transporters increase the complexity. The systematic application of microarray technology to the analysis of membrane

transporters may help determine tissue expression and allelic distributions on an individual basis (Sadée et al., 2000). Such technology may also facilitate the identification of putative defective alleles and the exclusion of affected patients from therapy with a given drug. With the amount of available data rapidly increasing as microarray technology is implemented, the need for strong information technology support becomes increasingly urgent.

1.2 Informatics Support in Transporter Pharmacogenomics Studies

1.2.1 Bioinformatics: Advances and Challenges

As we enter the transition era from structural to functional genomics, bioinformatics becomes increasingly important. Like the word itself, “bioinformatics” is an independent field created by the union of computer science and molecular biology. This merge was started in 1970s, when it was found that RNA secondary structure might be predicted with computational techniques (Pipas and McMahon, 1975; Studnicka et al., 1978). At that time, people started to build databases of nucleic acid (Erdmann, 1978) and proteins (Dayhoff, 1972). Algorithms and programs were developed to translate DNA sequences into protein sequences (Korn et al., 1977; McCallum and Smith, 1977), and to detect patterns including restriction enzyme recognition sites (Fuchs et al., 1978; Gingeras et al., 1978). With years of efforts, sequence analysis methodologies are maturing, and have become the most important part of bioinformatics.

In the mean time, experiments in laboratories and bioinformatics work itself, are generating huge amount of data, which requires and stimulates the growth of another branch of bioinformatics, data management and analysis. Nowadays, the amount of data in molecular biology alone is growing at an exponential rate. The size of GenBank is doubling every 16 months (Benson et al., 1999). In fact, problems in bioinformatics came up as early as the biological data. In 1980, an article in *Science* described a nucleic acid databank containing 200 entries and a total of 200,000 residues (Dayhoff et al., 1980). Just a year later, a notice was published in *Nature*, questioning “Too many databanks?” (1981). By now, a comprehensive catalog of biological databases contains at least 500 database entries (Discala et al., 2000). One journal, *Nucleic Acids Research* (NCR), devotes their first issue of each year entirely to biological databases. In 1998, it described 64 databases. In the 2000 issue, 115 databases were described.

Data structure, storage, representation, and retrieval are critical for handling the biological data. Friendly user interfaces and the media for data communication are becoming essential to satisfy users in different fields of biology from all over the world. In 1980, the full content of a database could be published in a paper journal (Roberts, 1981). Today almost every database has grown too large to be completely put on paper. Advanced database technologies are needed to handle the ever-increasing volume of data with ever-increasing complexity. Five years ago the most common type of a biological database system used a word-processing system. Now it is no longer an advantage to store data in the flat files of such a system. Instead, hierarchical databases and recently relational databases have been found to be more useful in dealing with biological data.

The World Wide Web (WWW) has also had a significant impact on the daily practice of biology. The web technology allows public access of databanks, fast delivering of information, and convenient communication among users. Another useful application is that it enables links between different biological databases, which make it possible for interpreting and sharing of data. Links can display associations between different biological entities, such as relationships between a transporter protein and its cDNA.

Table 1 is a brief review of some on-line publicly available biomedical databases. Some of these databases provide inter-database links, such as the Entrez system. The categorization of these databases is necessary to facilitate the full application of these resources. Otherwise users may not be aware of them, or what they are used for. In Table 1, these databases are categorized into several large groups, such as structure, function, phenotype, and more detailed subgroups, according to their content and usage. Many of these databases contain unstructured data which can be analyzed further. Data mining technology uses different approaches from different angles to elucidate new knowledge.

Category	SubCategory	Database Example	URL
Sequence and Structure	Nucleotide and Protein	Entrez	http://www.ncbi.nlm.nih.gov/Entrez/
	Gene-oriented Cluster	UniGene	http://www.ncbi.nlm.nih.gov/UniGene/index.html
	DNA	dbEST	http://www.ncbi.nlm.nih.gov/dbEST/index.html
	RNA	UTRdb	http://bigarea.area.ba.cnr.it:8000/EmbIT/UTRHome/
	Protein	PIR	http://www-nbrf.georgetown.edu/pirwww/aboutpir/collaborate.html
	Sequence Motif	Pfam	http://pfam.wustl.edu/hmmsearch.shtml
	Regulatory Structure	TRANSFAC	http://transfac.gbf.de/TRANSFAC/index.html
	Genetic Map	GDB	http://www.gdb.org/
	Genome	MITOMAP	http://infinity.gen.emory.edu/mitomap.html
	Gene Expression	BodyMap	http://bodymap.ims.u-tokyo.ac.jp/
	Structure (3D)	PDB	http://www.rcsb.org/pdb/
	Sequence Variation	dbSNP	http://www.ncbi.nlm.nih.gov/SNP/index.html
Function	Intermolecular Interaction	DIP	http://dip.doe-mbi.ucla.edu/
	Pathway and Cellular Regulation	KEGG	http://star.scl.genome.ad.jp/kegg/
Phenotype	Disorder	OMIM	http://www.ncbi.nlm.nih.gov/omim/
	Pathology	MTB	http://tumor.informatics.jax.org/FMPro?-db=TumorInstance&-format=mtdp.html&-view
Drug	Drug Name and review	DrugDB	http://pharminfo.com/drugdb/db_mnu.html
	Drug index	RxList	http://www.rxlist.com/
	Drug list and formulary	PCS	http://www.druglist.com/
	Drug Summary	DIZone	http://www.druginfozone.org/Pharmline/index.html
	Drug interaction	Cytochrome P450 Drug Interaction Table	http://www.georgetown.edu/departments/pharmacology/davetab.html
	Drug for Disease	P-I-E-N-O	http://www.parkinsons-information-exchange-network-online.com/drugdb/drugdb.html

Table 1.1 A brief overview of categories of some online public biomedical databases

Most of these databases have different structures and formats. The heterogeneity makes it difficult to intercommunicate between these databases. It is also difficult for users to have a complete knowledge of these databases for appropriate applications. Meanwhile, more and more databases are being built. Numerous fields of biology need exclusively devoted databases, e.g. promoters of the yeast *Saccharomyces cerevisiae* would require its own database (Zhu and Zhang, 1999). Even different labs are trying to build databases for their own areas of interest, while many are facing the difficulties of designing and implementing such systems.

Because of the need to build a large number of biological databases, intercommunication and interoperation between different databases are becoming critical issues. To solve these problems, bioinformatics is demanding a common literacy with mutual intelligibility that can be widely accepted (Frishman et al., 1998). This goal has been difficult to achieve, and is currently considered to be the first obstacle in biological knowledge modeling and encoding (Rechenmann, 2000).

Some efforts have been made to meet this objective. Much research has been done to develop ontology for knowledge sharing in molecular biology (Schulze-Kremer, 1998; Baker et al., 1999). For example, some researchers have suggested construction of a repository of terms hierarchically organized by methods of “is a subset of” and “is a member of” operators. However, the application of ontology in

biomedicine only gives a description of the terminology used in the domain, rather than providing conceptual representation of structures for storing and querying data (Paton et al., 2000).

These problems are central to providing informatics support for transporter pharmacogenomics studies because this is a complex area requiring heterogeneous data sources. These difficulties may hinder our progress in this area. On the other hand, because this can be seen as a typical area in biomedicine, the solution in this area may provide a template for solving problems in other areas. The methodologies and systems developed in this area may be reused by different researchers for different database systems.

1.2.2 Informatics Support in Pharmacogenomics

A bioinformatics decision support system (DSS) is a system that provides information to assist biomedical experts in making decisions and doing their job more effectively in both laboratory research and clinical practice. The application of microarray technologies and the analysis of gene expression and patient genetic profiles in pharmacogenomics require powerful computational support. Bioinformatics decision support systems are needed to facilitate the set up and design of such applications, and to record, store, analyze, and mine the data. Here a decision is an irreversible choice among alternative ways to allocate valuable resources.

It is difficult to construct such computer systems, considering the complexity in pharmacogenomics. As shown in Fig. 1.1, pharmacogenomics is multi-disciplinary

involving molecular biology and human genetics, genomics, bioinformatics, physiology, pharmacology, and internal medicine (Sadee 1998; Nebert 1999). Some of these domains are interlinked and overlapping. It is even difficult for experts from these different domains to communicate with each other. These multi-level characteristics and domain knowledge barriers bring great challenges to computer analysis methodologies to support decision making in clinics and labs.

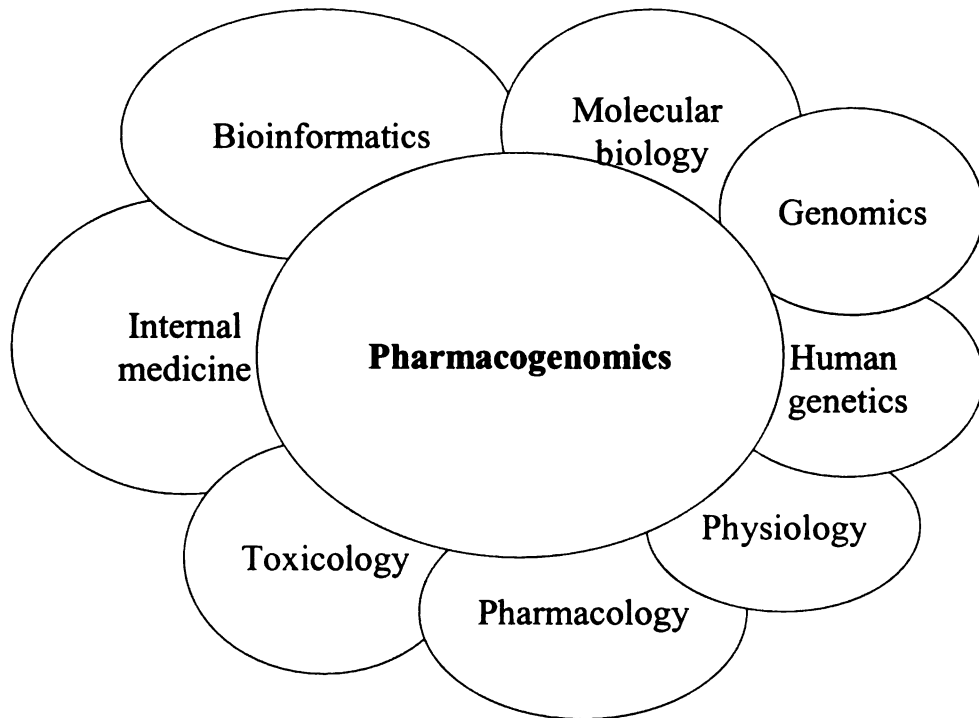


Fig. 1.1 The multi-disciplinary characteristic of pharmacogenomics: interlinked and overlapping knowledge domains

In addition, the special features of pharmacogenomics, such as genetic polymorphisms and genotype-phenotype correlations, are difficult to process using the

traditional passive and static flat files or even relational models. New and advanced data models need to be built to satisfy these new requirements. Data modeling provides a formal methodology for documenting users' data needs. The models we use for building pharmacogenomics computational systems will have a profound influence upon how a pharmacogenomics problem is attacked, how a solution is shaped, and how a result is interpreted.

To construct an accurate and usable model of a specific knowledge domain, we need to capture the important concepts and relationships of the domain knowledge, and be able to convert such understanding into physical data structures for decision support systems. In my research, I tried to attack this difficult problem and approach the complexity in pharmacogenomics by building a biomedical model from the domain requirement analysis. This model is scientific in nature and consists of accurate description and illustration of the fundamental factors and processes of our understanding of this particular area of science.

From this scientific model, important and repetitively occurring concepts and factors can be identified. These concepts and factors can be viewed as objects in the sense of object-oriented (OO) methodology. These objects, together with the interrelationships among them, should be able to represent the outputs as well as the inner workings of the biomedical model. These objects are also our building blocks for constructing sophisticated information systems.

Based on the biomedical model and abstracted concepts, object-oriented models can be constructed using the Unified Modeling Language (UML). UML is a standard modeling language that has been widely accepted in computer science. Therefore, the

model construction for the information system is through a three-step approach: domain requirement analysis and biomedical models \Rightarrow concept abstraction and object design \Rightarrow OO UML models. Such an approach can help us overcome the barriers between different knowledge domains and capture the essence of different disciplines for a coherent decision support system. The application of UML in this approach helps decompose the complexity and make the system comprehensible for pharmacogenomics data analysis.

1.2.3 The Process of Building the Informatics Support System

Fig. 1.2 shows the overall process of building the bioinformatics decision support system for pharmacogenomics studies of human membrane transporters. The first phase of the process is system design, which starts from understanding the biomedical needs and data requirements of the system users. Because of the special feature of the domain, this includes an analysis of the domain problems, understanding what attributes the user want, and which attributes are required, and which attributes are on a “wish list”. Also required are that the levels of detail or summary the users need, the type of front-end data access tool will be used, and how the users expect to see the results of their queries. For example, many users would like to access data and see the results from the Web.

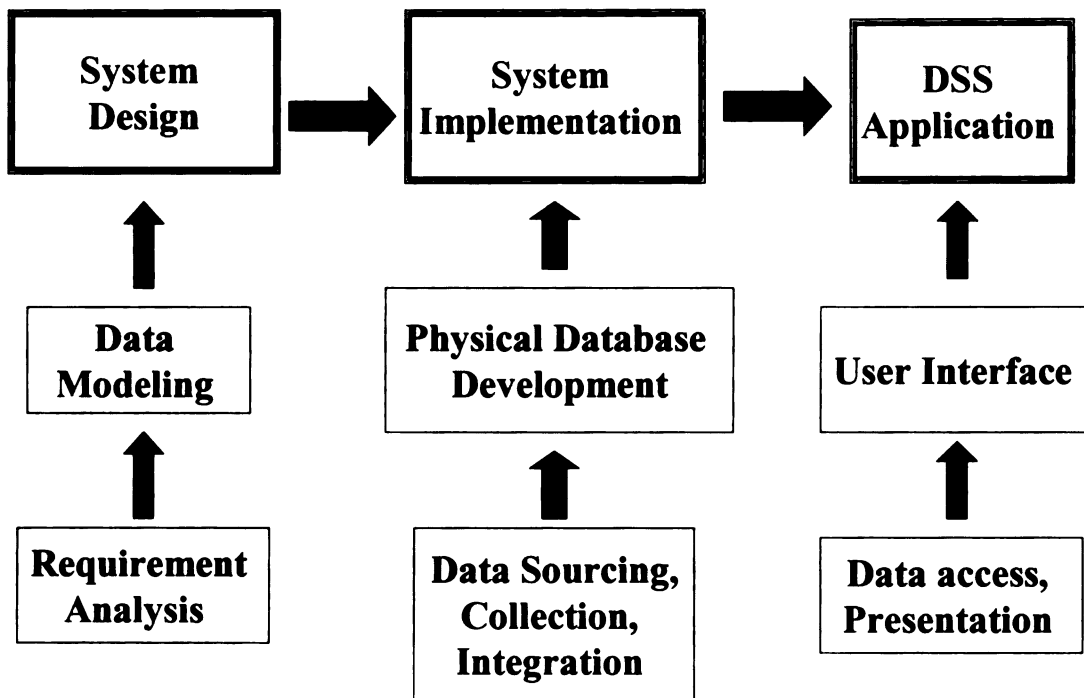


Fig. 1.2. Building The DSS: The Overall Process

The task of gathering requirements can be difficult, especially if the objective of the system has not been specifically defined. Sometimes a too broad scope of the system may also add the difficulty. To minimize these dilemmas, I define that the objective of the DSS is to focus on the most important correlations that can represent the key issues in transporter pharmacogenomics studies. These issues will be described in detail in the next chapter. Biomedical models for these correlations are built, and

logical models are developed based on these biomedical models to define the scope of the system.

Information collected during the requirement analysis and then the biomedical model construction will feed the data modeling directly. A data model covers the scope of the system development including relationships, attributes, and definitions. Issues to be concerned at this phase include how the relevant biomedical data will be reflected in the model, whether there are clear procedures with respect to the models, and how to transform a logical model to the physical mode. The methodology I used to build OO models addresses these issues (will be described in detail in Chapter 2). This approach makes it clear that those relevant biomedical data are reflected in our model. The application of UML can help us transform the logical model to the physical mode smoothly.

The next phase is system implementation. A physical database system is developed according to the data model designed in the previous phase. In the meantime, the possible source systems are defined. For example, protein sequence data may come from GenBank Entrez system, and nucleotide sequence variation information may come from Online Mendelian Inheritance in Man (OMIM), the Database of Single Nucleotide Polymorphisms (dbSNP), and Genome Database (GDB). During this phase, data analysis and integration are performed to determine the best and cleanest source of data. For example, redundancy in cDNA sequences for the same gene should be removed, and different names of the same gene should be organized. Nomenclature and classification of transporter genes are also considered during the data integration. Then data can be loaded and mapped to the target system.

The last step is system application. In this phase, data access tools are used to build reports. Multi-tiered architectures are used to support the DSS application. The reports can be developed through programming using coding languages and structured query language (SQL) statements. Structured navigation paths are developed to access the reports of data.

The types of access include parameter-based ad hoc reports, and electronic access to predefined reports. In the first type, end users can create a report for a specific purpose through modifying existing reports. In the second type, predefined reports with fixed format are produced and placed in a commonly accessible location, such as on the Web, for users to pull up for viewing if needed. Reports can be presented with a structured graphical user interface (GUI) for navigation. Usually a single way of data access is not able to provide all the capabilities of decision support. Different audiences may demand multiple means to meet a spectrum of data access needs.

1.2.4 Object-Oriented Technology for System Design

Rather than a simple combination of computer science and biology, bioinformatics should be an “organic” integration of the two. In many cases, this means to use computer technologies in solving biology problems. However, before the application of these technologies in the real biomedical world, heavy-duty research work has to be done to lay the ground for further implementation. It is the biomedical informaticians’ responsibility to bridge this gap.

In the area of “pure” computer science, the similar difficulties in system design and modeling that we encounter in bioinformatics have been studied for years. As a result of these efforts, new advances and solutions have been made. In the 1980s, “objects” began to “grow up” from computer research labs and took their first steps toward the “real world”. In real world terms, an object can be defined as a concept, abstraction, or a thing with crisp boundaries and meanings for the problem at hand (Rumbaugh et al, 1991). In software terms, an object is an intelligent piece of a program that can encapsulate code and data.

Objects provide reusability, which can be defined as “the ability of software products to be reused, in whole or in part, for new applications” (Meyer, 1988). As a relatively new concept, object-oriented system development is an extension of structured programming. Structured programming emphasizes the benefits of properly nested structures. Object-oriented development highlights the benefits of modular and reusable computer code and modeling real-world objects. However, the later approach makes the system more flexible and convenient for future use. This advantage is especially obvious in building complex systems. Compared with developing in the traditional structured programming environment, developing an object-oriented application requires even more thought about the design. This is because the focus on future reuse requires a longer-term view during analysis and design.

In the computer area, techniques and development tools for helping people do good analysis and design are just as important as object-oriented development itself. The term “object-oriented technology” covers a body of methods, processes, and tools used to construct software systems from objects (Kozaczynski and Kuntzmann-

Combelles, 1993). Object-oriented methods offer a unifying paradigm for the three traditional phases of software development: analysis, design and implementation (Korson and McGregor, 1990). This unification leads to a smooth transition from one phase to the next.

The major features of object-oriented technology include inheritance, encapsulation, polymorphism, and abstraction. The major benefits of OO technology are the support for reuse and better understanding of the software system. Inheritance allows a software developer to reuse existing objects through inheriting their capabilities, including methods and data. Encapsulation is one of the features of the object-oriented approach that is the most useful for relational databases. Encapsulation enables the data and methods to be private within an object. With this feature, developers can modify encapsulated data and methods without disturbing other portions of the application, as long as the external interfaces remain unchanged. Polymorphism in OO technology refers to the ability of different objects to receive the same message but behave in different ways. Polymorphism here means that a standard interface may be created for a related group of objects. Abstraction delineates the conceptual, but not concrete, existence of classes within a system. For example, a database may have a class hierarchy that includes classes that do not have any objects.

1.2.5 Unified Modeling Language

The review of how the problems are solved in computer science may give us a hint about how to solve problems in bioinformatics. In fact, attempts to accomplish standardization in OO methods took years of efforts by computer scientists. With the

purpose of creating standards allowing the interoperability and portability in distributed object-oriented applications, an Object Management Group (OMG) was set up in 1989. In the 1990s, an Analysis and Design Special Interest Group (SIG) of the OMG published a report of OO methods that was widely ignored.

During 1996, Grady Booch, Jim Rumbaugh, and Ivar Jacobson, now widely referred to as the three amigos, worked together to merge their different methods and came up with a new symbol system: the Unified Modeling Language (UML). By mid-1997, all the OMG submitters had congregated on UML version 1.1. Today, UML is widely accepted and used as a standard language by software designers.

UML is an object-oriented design language for specifying, visualizing, constructing, and documenting the objects of a system (OMG Unified Modeling Language Specification, 1999). This language has been extensively used in the business world and has been demonstrated to be useful for system analysis and design. However, the application of UML in the biomedical world is still rare (a search in Medline returns less than 10 relevant papers).

UML is adopted by only a few systems, mainly in the medical imaging field (Martinez et al., 1999; Cook et al., 1999). Gary et al. (2000) used the style of UML class diagram in the graphical representation of their Biomolecular Interaction Network Database (BIND) data model. However, an important part of the UML class diagram, the notations about relationships between different classes, was not used or indicated in their representation. In describing some conceptual models of genomic information,

Paton et al. (2000) adopted some notations of UML class diagrams in their schema diagrams. But none of these studies took full advantage of the modeling language, which helps define user requirements and system boundaries, and allows the mapping of both structure and functionality in the domain. These features are crucial for further query applications and data mining of the system. Systematic methodologies for full usage of this new technology in the special biomedicine field need to be developed. Successful application of such methodologies in the development of an informatics support system for transporter pharmacogenomics studies can provide an example for applying the standards in other areas of biomedicine.

1.2.6 Design Patterns and Reusable Software

Another approach for seeking the solution to the difficulties in bioinformatics is to find design patterns in bioinformatics systems, because many design problems reoccur in different systems. Patterns for software development are one of the hottest topics that emerged from the object-oriented society in 1990s. The goal of defining software design patterns is to create the literature to help solve recurring problems during the software development process. Patterns can help form a shared language to convey insight and experience about these problems and their resolutions. In addition, formally codifying these resolutions and their relationships allows us to successfully capture the knowledge that circumscribes our understanding of good system architectures. Such good system architectures can help meet the needs of system users.

A good methodology is also one that can be widely adopted in different areas. The origin of design patterns is not from software itself, but from architecture and anthropology. The use of the term “pattern” is derived from the writings of an architect Christopher Alexander who has written several books on the topics related to urban planning and building architecture. Alexander wondered whether architectural quality was objective, and what it was in a “good” architectural design. After studying many buildings, towns, and almost every other aspect of living spaces, Alexander found that good construct had commonalities, although these architectures differed from each other. He recognized that structures couldn’t be separated from the problems they were trying to solve. He distinguished similarities among high quality designs in structures solving comparable problems. He called these similarities “patterns”, and defined a pattern as “a solution to a problem in a context.” (Alexander et al., 1977)

In his books, Alexander wrote that “each pattern describes a problem which occurs over and over again in our environment and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.” (Alexander et al., 1977) Alexander thought four items were involved in a description of a pattern, including the name of a pattern, the problem the pattern solves or the purpose of the pattern, how this could be accomplished, and the constraints and forces to be considered to accomplish it.

Although Alexander’s books are presumably about architecture and urban planning, they are also appropriate for many other disciplines, including software development. Around the early 1990s, some software developers realized that problems

in software development that occur over and over again could be solved by a fairly similar approach. In 1987, Erich Gamma worked on recurring design structures or patterns in his PhD thesis. Later, he became the first author of the classic book on this topic, *Design Patterns: Elements of Reusable Object-Oriented Software* (1994). In the software community, the four authors, Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides are frequently referred to as the *Gang of Four* or just *GoF*.

As defined by Dirk Riehle and Heinz Zullighoven, a pattern is the abstraction from a concrete form that keeps recurring in specific non-arbitrary contexts (Dirk Riehle and Heinz Zullighoven, 1996). In the *GoF* book, a design pattern is interpreted to name, abstract, and identify the “key aspects of a common design structure that make it useful for creating a reusable object-oriented design”. In the book, every pattern was diagrammed using the Object Modeling Technique (OMT) notation and provided with C++ codes (UML and Java were not available at that time).

There are many benefits to studying software design patterns. Patterns provide reusable solutions for commonly recurring problems, and experience that can be shared among designers. As mentioned before, lack of common literacy is the first obstacle in bioinformatics development. Design patterns provide a common point of reference during the analysis and design phase of a project. They can help establish a common terminology in the bioinformatics community.

In addition, patterns can provide a higher level perspective on the bioinformatics problems and on the process of design, without dealing with the details too early. Too

much detail during the design phase may confuse the solution. Rather than being confused by the trees, patterns help a designer see the forest.

Design patterns interpreted with UML notations may enable great advances in biomedical informatics for system design, for constructing common literacy, and for producing generic biomedical databases. These are the most prominent difficulties mentioned above. However, the importance of design patterns in bioinformatics systems has not been recognized (no literature was found regarding the topic). From my own experience of designing several biomedical database systems, I encountered such “recurring problems” and found the study of the common patterns would be very helpful. Using the transporter system as a template, I will discuss these “patterns” in Chapter 2.

1.2.7 System Implementation: Database Management System

Based on the data model and design, the informatics support system can be implemented with a database system. A database is a coherent gathering of related data. It represents a model with the contents supporting the aspects that are relevant to a defined series of objectives, such as a set of applications. In the 1960s, the term “database” first appeared to describe complex data storage systems that embraced information about various kinds of entities in multiple tables and related transactions. Before 1970, databases were generally considered as territories through which computer programs could “navigate”, following pointers from one record to another along preset pathways.

In 1970, Dr. Edgar F. Codd (who was working at IBM at that time) proposed a completely new paradigm for thinking about data, in which all meaningful relationships among data records are represented by data values. With this paradigm, queries could be made independently of the structures and algorithms used in the database implementation. Codd's paper, "A Relational Model of Data for Large Shared Data Banks" (Codd, 1970) is one of the most influential and widely cited papers in all of computer science.

In 1973, a project that was known as "System R" was launched at the IBM Research laboratory in San Jose, California. This project was to build an industrial-strength relational prototype. The designers of System R developed a new database language, which later became the world's most widely used database language, "Structured Query Language (SQL)". In 1987, SQL was adopted by the International Organization for Standardization (ISO).

Relational database management systems (RDBMS) have become the standard for tabular databases. A database management system (DBMS) provides the means to define and construct a database, such as supporting links or relationships between related records. A DBMS is a general-purpose software system that supports efficient and reliable shared access to a database. Such a system provides mechanisms to ensure the integrity and security of the stored data. Simply put, a relational database consists of tables containing columns and rows, and rules defining the relationships between the tables. There is a language (SQL) for managing and querying the stored data. Relational databases are still the most used and most reliable tools for data management and queries.

Currently, many public biological databases are still in the free-text flat file format, and the information is stored in an unstructured way (Tsoka and Ouzounis, 2000). For example, several web sites containing information on transporters are in this format, such as the page maintained by Michael Müller in the Netherlands (Müller, 2000). In these nonrelational systems, it is difficult to find data beyond the logical structure that was originally defined. Such difficulty may hinder flexible queries and large-scale data mining. For example, to find a piece of information that is derived from raw data, the user has to go over all the forms or articles. This makes it difficult to obtain direct answers to questions such as “Which transporters are expressed in the kidney?” and “What drugs are known to interact with these transporters?”

It is also time consuming to update records in a flat-file system. A relational database overcomes these obstacles. Such a database provides a sound and reliable model to organize and maintain information in the form of tables representing records or record structures. In a relational database, data is structured and standardized to enable further application of decision support, such as ad hoc queries and data mining procedures. Structured data provides the key to convert often-used observations and entries into distinct, predictable, and reproducible selections. Structured data will also facilitate further use of the data model for decision support, such as ad hoc queries, clustering, and other data mining methods. These tasks are very hard, if not impossible, to perform with non-structured flat-text files.

In a relational database, adjustments can be easily made, and all the data elements can be queried flexibly. In addition, views of the data do not have to be predefined, and relationships can be built on a moment’s notice. Recently some

biological databases have adopted RDBMS technology, such as the Genome Sequence Database (GSDB) at the National Center for Genome Resources (Harger et al., 1998), and the Mouse Genome Database (Blake et al., 1998).

1.2.8 Data Representation and User Interface to Facilitate Decision Support

Without data representation and user interface design, decision support function would be impossible. Sometimes a valuable truth can be hidden, if it is not presented in an easily recognized format. The primary goal of a user interface is to enhance user effectiveness and satisfaction. An optimal user interface assists the user's decision-making processes and reduces the possibility of mistakes. The development of graphical user interface (GUI) introduces a limited number of options to the users, instead of asking users to memorize and enter commands by hand from a virtually unlimited set of options.

There are several principles involved in a good user interface design. It is generally accepted that the most fundamental quality should be intuitive. An intuitive interface is easy to learn. To some people, an intuitive user interface is one that users can figure out for themselves without the help from others. For example, concise, cogent, and unambiguous icons and labels can be intuitive with respect to their meanings. Another essential principle of a user interface application is consistency. For example, the use of labels and icons must always be consistent. The same label or icon should always represent the same thing. On the other hand, the same thing should always be represented by the same label or icon.

In addition, objects should also be placed in a consistent manner. Stable objects should always be provided as static orientation points around which the users can navigate. If users ever get lost or disoriented, they should be able to find the stable objects quickly and get to where they need to be from there. The third principle is simplicity. As Occam's Razor stated, the most graceful solution to any problem is the one which is the simplest. The application of this principle in user interface design may include minimizing the number of panels that must be displayed, and the number of mouse clicks or keystrokes that are necessary to complete a specific task. The fewer things users have to see and do to get their work done, the happier and more effective they will be.

In existing biomedical databases ad hoc or predefined reports are often used to support applications, although in most cases only one of them is used. The most commonly used format to support the first type, the parameter-based ad hoc report, is through a searchable form. Users fill out a form, usually through selecting the parameters of their choice and typing in some key words or genetic sequences, and then submit their queries. The system will return a report based on existing data in the database system. The other common format is that users can browse the data in predefined reports through certain navigation paths.

1.3 Summary

Table 1.2 summarizes the most important problems to be solved in the informatics support of transporter pharmacogenomics studies. The problems can occur

from the knowledge domain of pharmacogenomics, from bioinformatics, and from the process of informatics decision support itself. Finding the resolution to these problems relies on the understanding of the knowledge domain and the development of new methodologies. The following chapters will focus on the methodologies to solve these problems, supported by details of the development.

Domain	Problem
Pharmacogenomics	Multi-disciplinary characteristics Domain barriers Complex concepts Complex correlations Large amount of data Heterogeneity in data
Bioinformatics	Lack of common literacy Unstructured data Uncategorized databases Different database structure and format Data interoperability Lack of reusable models and software
Decision support	Requirement analysis Data modeling Physical database design and development System architecture Data integration Data access and representation User interface

Table 1.2 A summary of the most important problems for decision support in transporter pharmacogenomics studies

Chapter 2. System Analysis and Design: Object-Oriented Data Modeling and Design Patterns

The way we model the world influences the way we affect the world.

—*Enrico Coiera,*

Guide to Medical Informatics, the Internet and Telemedicine, 1997

2.1 Object-Oriented Analysis and Design: The Methodology

2.1.1 The Methodology of Design

As the proverb says, “owning a hammer doesn’t make one an architect”. This is also true in software development. Possessing biomedical knowledge and knowing an object-oriented language (such as Java) are necessary but insufficient in creating biomedical informatics support systems. Analyzing and designing are the critical first steps for building our transporter pharmacogenomics decision support system (TPDS). The analysis process examines the problems and requirements of a system. The analysis process identifies what the problem is and what a system needs to do. The design process provides a logical solution so that the system satisfies the requirements.

In the business world, requirement analysis usually begins with creating scenarios, use cases, or CRC (Classes, Responsibilities, and Collaborators) cards to get an overview of a problem. A scenario describes specific step-by-step examples of system utilization. The nouns applied in scenario descriptions often define classes in

later data modeling. CRC cards are usually 3-inch x 5-inch lined index cards on which classes and information about classes can be written. CRC cards are usually used by a group of people (composed of users and developers) working together to identify the classes in an application via talking through scenarios. Use cases identify generic procedures the system must be able to handle. These methods help define the overall structure of the problems. Once the requirements are specified, UML diagrams can be used to analyze, design, and develop applications.

The UML is a language only for modeling, but is not a guide in the development process or how to do OO analysis and design. The methods, models, and development processes still need to be defined by methodologists, but not UML. Different methodologies and development sequences can be used according to specific features of different projects. The methodology I used for system analysis and design, briefly a process of “domain requirement analysis and biomedical models \Rightarrow concept abstraction and object design \Rightarrow OO UML models”, is illustrated in Fig. 2.1.

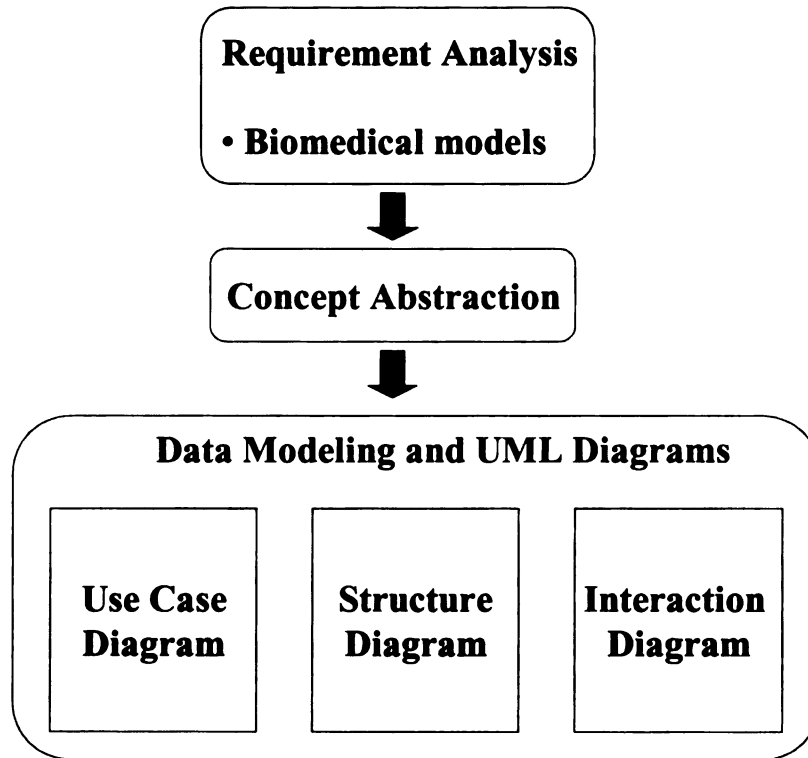


Fig. 2.1 The process of system analysis and design

The biomedical system we are dealing with is overwhelmingly complex. It is necessary to decompose it into understandable chunks to comprehend and manage the complexity. To do this, models can be constructed to describe, abstract and represent essential aspects of the system. Because biomedicine is a very specialized field that requires specific knowledge and skills to understand, it is difficult to perform the analysis and design steps the same way as in the business world. This may be why it has been difficult to apply UML in biomedical research work. To solve this difficulty, a systematic overview of the knowledge domain is necessary to clarify the target domain

first. The methodology I have designed is to analyze the knowledge domain and define the key issues, and construct biomedical models to describe the target problems, before building data models in traditional ways.

A biomedical model describes the natural scenarios, including objects, processes and relationships in a problem. It reflects the understanding and thinking process of domain users for whom the software system is built. It helps capture the most important issues that the users are concerned about. Biomedical modeling before data modeling can help clearly identify the problem domain in the real biomedical world, lay the ground for further concept identification, and facilitate the later steps.

This biomedical modeling step is necessary in order to overcome the domain barriers among different areas in biomedicine which is a significant problem in pharmacogenomics. Sometimes it is hard to decide where to start the data modeling. In many cases it is very difficult for software designers to communicate with domain experts and do the requirement analysis. The construction of biomedical models based on domain analysis helps solve these problems.

Biomedical models represent the problems in the biomedical system in an intuitive way using the language of the field. The requirement of a biomedical model includes that it should show the biological objects, associations and processes as they are understood by biomedical experts. It should be a representative picture when the expert is thinking of the problem. It is the model that a biomedical scientist would use to communicate with colleagues.

These models then need to be translated into data models that are the foundations of further software development. Based on these biomedical models, use

cases can be described and concepts can be abstracted. The step of concept abstraction can lead to the data modeling phase with the creation of series diagrams. As in usual business scenarios, nouns and things appeared in biomedical models can also be analyzed into common types and abstracted as concepts, or classes. Processes and mechanisms in biomedical models can be abstracted into relationships.

In data modeling, classes, their associations with other classes, and inheritance relationships are described using UML static structure diagrams (also called class diagrams). Then the flow of messages and events between objects are shown in UML interaction diagrams (also called sequence diagrams). Interaction diagrams provide a formal way to identify a scenario. The class and sequence diagrams are central of our analysis and design efforts. Table 2.1 summarizes the different types of the diagrams used in the TPDS data modeling. The details of the UML notations and diagrams are described in the following sections.

Model	Diagram	Explanation
Use cases	Use case diagram	Recognition of the major functional parts of the system and the interaction of actors with use cases
Object structural model	Class diagram	Defines key abstractions and their logical constitution in mechanisms, including design patterns
Object behavioral model	Sequence diagram	Paths of interest (such as typical and exceptional) through the collaborations of defined classes; Represents scenarios of applications of a system; Shows particular paths through a use case including messages sent between the use case and its related actors

Table 2.1 The Summary of UML Diagrams

2.1.2 Unified Modeling Language: What It Is and How It Is Used

2.1.2.1 Use Case Diagrams

The UML is a notation system aimed at modeling systems using object-oriented concepts. In UML terminology, user requirements are expressed in terms of *use cases*. A use case is a process that fulfills certain requirements of a system user (Harmon and Watson, 1998). A process describes a sequence of events, actions, and transactions needed to complete something of usefulness to a user, from start to finish. Use cases are stories and circumstances of using a system. A use case is a relatively large end-to-end procedure description that typically involves many steps or transactions. Normally, it is not an individual step or activity in a process, such as a simple DNA sequence retrieval step.

Use case diagrams describe the main processes in a system and the interactions between the processes (use cases) and the external systems, or actors. An actor is an entity outside the system that in some way participates in the story of the use case. An actor can be a person, a machine, or another software application. An actor usually stimulates the system with input incidents or in receiving some output from it. Actors are represented by the role they play in the use case, such as a pharmacologist, or a bioinformatician. A use case diagram illustrates the functionality of a system and its major processes, defines the system boundaries, as well as the users that will utilize the system. Use cases comprise all the system functions identified during the prior requirement analysis.

The methods to define use cases can be actor-based or event-based. Through the actor-based method, the actors associated with the system are first identified. Then for each actor, the processes they are involved in are identified. The event-based approach identifies the outside events that a system responds to, and connects the events to actors and use cases.

The intention of the use case diagram is to portray a kind of context diagram through which one can quickly understand the outside actors of a system and the key ways in which they utilize it. In a use case diagram, a rectangle with rounded corners represents the application or system (as illustrated later in Fig. 2.7). An application can have one or more use cases. Use cases are shown in ovals, with the name of the use case written inside the oval. Actors are represented in stick figures, with the name written under the figure. A line links the actors and the use cases they interact with.

2.1.2.2 Concept Abstraction and Class Diagrams

UML is a standard notation and modeling language that denotes information about both the static structure and the dynamic behavior of a system. The static structure delineates various kinds of objects essential to a system and the associations among the objects. The dynamic behavior identifies the history of objects over time and the communication among objects to carry out goals. Static structure diagrams show the things that are present in the model, their internal construction, and relationships to other things. A static structure diagram is essential because it helps define what we are dealing with. This kind of diagrams provides a static view of a

system. These diagrams are abstractions of the concepts, objects, and relationships from the biomedical diagrams.

Static structure diagrams are also called class diagrams. In UML, a class describes a set of objects that share the same attributes, methods, and relationships. A class diagram illustrates classes and the relationship between classes. In a class diagram (as shown in Fig. 2.8), a large rectangle is divided into three horizontal compartments. The name of the class is written in the top section. The second section records the attributes of the class. The lower section usually includes operations and methods.

As a rule of thumb, there are no “correct” or “wrong” designs in a class diagram, but just more or less useful ones. A class diagram is actually a means of communication. Choices about what is illustrated are made with this consideration. There is no such thing as a single accurate model. All models are approximations of the domain we are trying to understand. A good data model is one that helps people comprehend the domain by capturing the essential abstractions and information in the context of the current requirements.

The creation of a class diagram starts by abstracting concepts and translating them from the biomedical models. Fig. 2.2 illustrates this process. The concept abstraction process includes listing the candidate concepts involved in the biomedical models. To do this, I designed a Biomedical Concept Category List (as shown later in Table 2.2). This list may not be comprehensive, but contains common categories in biomedical studies that are worth considering and including in our system. The operation process is to abstract the concepts in the biomedical model to fill in the

categorical list. In addition to the categorical list, noun phrases related to the current requirements under consideration can also be abstracted from the biomedical models.

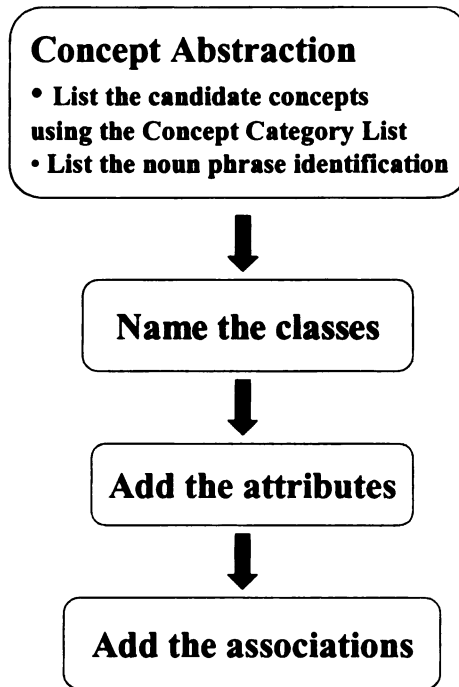


Fig.2.2 The process of concept abstraction and class diagram creation

When the classes are defined from concept abstraction, the attributes that comply with the information requirements are then added. An attribute is a logical data value of an object. Next, the associations needed to record relationships are also added. An association is a relationship between objects that designate some meaningful and interesting connection. The most generic association is denoted as a line between two classes. In some cases an association can be an object that has its own attributes. The representation of an association class is useful when there is a relationship between

many objects and many other objects, while the attributes are the characteristics of the connection itself but not any of those classes being linked. Such an association class can be connected to the regular association line with a dotted line (an example will be shown in Fig. 2.9). A dotted line stands for a dependency. A dependency implies that one of the elements will change when the other changes.

A class hierarchy can be formed when the subclasses inherit attributes and associations of a super class. The subclasses can have particular characteristics of their own. This inheritance relationship between super classes and subclasses is illustrated with an open arrowhead (examples can be found in Fig.2.8).

Another type of association is aggregation, which refers to the part-whole relationship. An aggregation relationship is represented by putting a small diamond at the end of the association line that runs between part classes and the whole classes, heading towards the whole class (as shown in Fig. 2.8).

2.1.2.3 Sequence Diagrams

The use case suggests how actors interact with the software system we are interested in constructing. Isolation and illustration of these operations can improve the understanding of the system behavior, that is, what a system does.

An interaction diagram, also called a sequence diagram, is a graphical way to describe a specific scenario of a use case. Here a scenario of a use case is a particular instance and a real example of its performance. Sequence diagrams portray a more detailed view of the interaction between the objects of the main classes in the system.

This kind of diagram is used in conjunction with structure diagrams to get hold of most of the information about a system. A sequence diagram enables dynamic views of a system. Before preparing a sequence diagram, the classes in the system should be known. So sequence diagrams are usually created after the construction of class diagrams.

A sequence diagram shows how the actors interact directly with the system, and the system events the actor generate. These diagrams allow one to study how a set of objects interacts in time. A system event is an external input incident that stimulates a responding operation generated by an actor to a system (an example is shown in Fig. 2.10).

The creation of a sequence diagram starts with defining a set of classes that will probably be involved in a scenario. The objects identified are listed along the top of the diagram, with a dotted line beneath each object (as shown in Fig. 2.10). These dotted lines are called lifelines. The objects are listed in the order they are employed in the scenario. The leftmost object is the one that makes the stimulus that starts the scenario.

Many different event sequences can be represented in the same sequence diagram. Events within a sequence, which happen later in the time order, are often represented lower on the chart (as shown in Fig. 2.10). Objects are linked by an event arrow, which suggests that a message is transferring between those two objects. The event ceases at an arrowhead.

2.2 Object-Oriented Data Modeling of Pharmacogenomics Studies in Transporters

2.2.1 Key Problems in Transporter Pharmacogenomics Studies: Domain Requirement Analysis

2.2.1.1 Structure-Function Relationship

Because of the multi-disciplinary characteristics of pharmacogenomics, it is hard to understand the area by studying concepts in separate disciplines. There are two ways to solve the “where to start” problem for informatics research in this area. One way is from concepts to associations. That is, trying to capture all the concepts and then to find the associations between these concepts. However, this approach may result in some of the concepts being ignored, and some associations being unclear. Another way to solve the problems is “finding melons from the vine” (a Chinese proverb). That is, to capture the most important correlations in the area at the beginning. From these correlations, the related and necessary concepts involved can be defined. These correlations may connect the separate concepts from different disciplines and help form a complete view of the whole area. The analysis of these correlations cannot only help us identify the domain problems and requirements for informatics support, but also clarify the informatics challenges that should be met.

One of the most important goals in transporter pharmacogenomics study is to elucidate the relationship between the structural and functional properties of transporter molecules. For example, the nucleotide-binding domains (NBD) of CFTR hydrolyze ATP to regulate channel gating, and the CFTR regulatory (R) domain phosphorylation

controls channel activity (Anderson et al., 1991; Berger et al., 1991). However, it is difficult to infer the structure-function relationships from individual data. The purpose of informatics study here is to put them into an interweaving picture.

Molecular cloning of transporter subtypes may clarify the complexity of the structure-function relationships. For example, transporter subtypes can have a similar function but different tissue distribution, regulation, and specificity towards a drug. The correlation between transporter structure and function will enable a better description of transport mechanisms.

The understanding of transport mechanisms offers insight into how the transporter proteins may be altered in diseases and regulated by therapeutic agents. In addition, the structure-function correlations will be useful in the design of more specific transporter reagents with high-quality therapeutic effects. To elucidate such correlations, a more complete understanding of transporter structure, including the three-dimensional topology and tertiary structure, is required. Therefore these elements should be included in our informatics consideration. The identification of the structural elements is necessary to explain the direction of translocation and subcellular localization.

In addition, it will be helpful to elucidate the role of a transporter in the whole genome and the relationship of a transporter gene to other genes nearby on the chromosome. The genomic analysis may provide insight into gene regulation and evolution, such as the example that vesicular choline transporter is contained entirely in the first intron of the choline acetyltransferase gene (Bejanin, 1994; Erickson, 1994).

The sequence and structural data of transporters can be found sporadically in GenBank's Entrez system and Protein Data Bank (PDB). Information on transporter functions is scattered in various databases, such as OMIM, and the literature. Currently there is no integrated database available for both structural and functional information about transporters. If researchers are interested in the structure of a particular transporter and its relationship with function, they have to look into several databases and the relevant literature. This makes it more difficult to pursue any data mining tasks, or to find any patterns in the structure-function study for future predictive exploration. A systematic collection of the data in these aspects via informatics approaches may promote the development in this area.

2.2.1.2 Genotype-Phenotype Correlation

From the perspective of translating pharmacogenomics into clinical medicine, the correlation between genotype and phenotype plays a crucial role in this translation. Fig. 2.3 illustrates the correlation between genotype and phenotype, using the role of a transporter gene in breast cancer therapy as an example. In classical genetics, the "phenotype" is usually defined as a visible trait, such as black hair or blue eyes. Clinical traits, such as drug responses, can also be defined as "phenotypes". The illustration of the correlation includes two aspects. One is the definition of a clinical phenotype such as resistance to the drug tamoxifen in breast cancer therapy, and the result of tumor progression. The other aspect is the genotype such as an altered MDR1

gene, which may perform a causal role in protein change and then the phenotypic change.

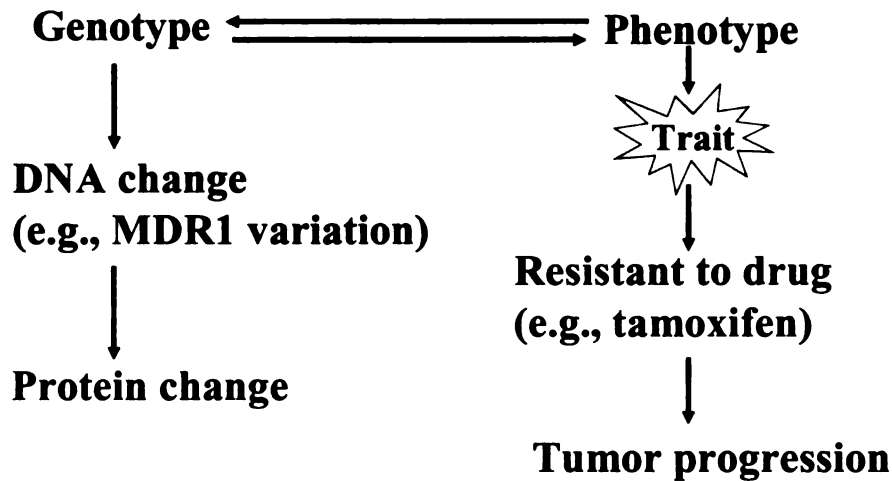


Fig. 2.3 The correlation between genotype and phenotype

Because the data of human genome sequence and genetic variability are becoming available, whatever buried in these data including the entire range of phenotypic variation. Meanwhile, microarray technology is enabling large-scale genotyping and gene expression profiling of human populations. For example, at least 5 distinct disease phenotypes, including retinitis pigmentosa, cone-rod dystrophy, and

Stargardt macular dystrophy, have been related to transporter ABCR mutations (Lewis et al., 1999). Such correlations between the sequence variation genotype and disease phenotype might also affect drug targets and the correlated drug response phenotype.

Informatics tools for managing the data, for knowledge discovery and prediction, and for visualizing results will be crucial for realizing the full potential of the microarray results. Databases comprising gene expression profiles of normal and diseased tissues will assist the linking between genotype-phenotype.

There are still many challenges in the integration of the sequence databases and their clinical phenotypic annotation. Currently existing clinical data models are separate from genomic data models, thus neither of them are sufficient to annotate this correlation. Dramatic advances will be necessary for theory and model development in this aspect of study. Systematic catalog of the correlation through databases and decision support systems should help predict disease prognosis and drug response, prevent adverse drug reactions, and find optimal prescriptions.

2.2.1.3 Gene-Drug Interaction and The Comprehensive View

The fundamental gene-drug interaction is the central part of pharmacogenomics and has been considered “extremely important” in drug development and clinical medicine (Nebert, 1999; Dirckx et al, 2000). Pharmacogenomics represents studies on the essential gene-drug interactions through genetic mechanisms and functional pharmacological context, in cooperation with clinical studies. When drugs enter the

human body, their fate is affected by uptake, binding, distribution, biotransformation, and excretion. On a molecular basis, the efficacy of a drug is influenced by the alterations in receptor affinity, transporters, or protein binding. An example of transporter gene-drug interaction is that the functional polymorphism in 5-HTT affects the antidepressant response to antidepressants fluvoxamine and paroxetine (Smeradi et al., 1998; Kim et al., 2000; Pollock et al., 2000; Zanardi et al., 2000).

Up to now most of the informatics work focused on just one domain, either gene or drug separately. The illustration of the interaction may include the responding genes to specific drugs, the expression level of these genes, the sensitivity of a cell to a drug, as well as pharmacological characteristics of drug action. For example, the polymorphism of MDR-1 could significantly increase digoxin plasma levels in patients, and affect the absorption and tissue concentrations of other substrates of MDR-1 (Hoffmeyer et al., 2000).

Through identifying these three correlations (structure-function, genotype-phenotype, and gene-drug), the major problems to be solved in transporter pharmacogenomics are clarified and our objective of informatics support can be defined. Without such problem identification, informatics analysis would be hard to begin. In fact, these three correlations are not isolated issues in pharmacogenomics. They are interwoven which help us view the problems from different angles.

Now we can look at these issues comprehensively. As shown in Fig. 2.4, the three correlations are tightly connected and interlinked. The correlation between genetic structure and observable normal functions can be represented as normal phenotypes. On the other hand, altered genetic structure may lead to malfunctions and

be expressed as disease phenotypes. In addition, varied genetic structure and the resulting altered functions can affect the drug response phenotype, as the examples above illustrated.

The drug response phenotype is also the result of the gene-drug interactions. Moreover, varied genetic structure and function can affect the gene-drug interaction. In these three correlations, genotype-phenotype is the broadest concept that covers the whole biomedical process of drug therapy. However, identification of the other two correlations is also important to provide detailed cause-effect relations in this process. With the identification of these three correlations and their interlinking relationships, then we can define the major objects that are essential elements in our informatics system modeling.

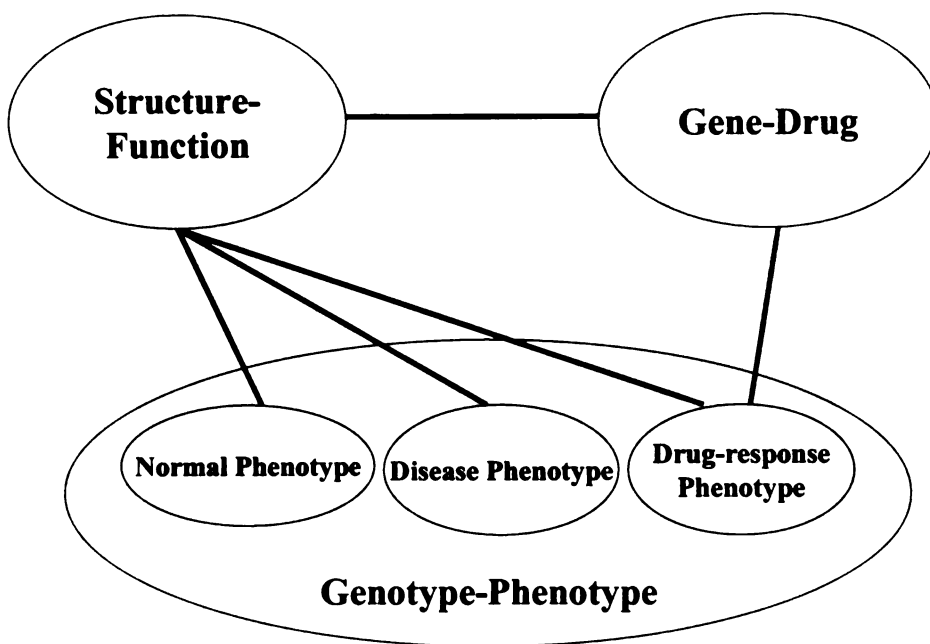


Fig. 2.4 The comprehensive diagram of the three correlations: structure-function, gene-drug, genotype-phenotype

2.2.2 Biomedical Models

Based on the domain analysis, now we can build the biomedical models. Fig. 2.5 shows a biomedical model describing the structure-function, gene-drug, and genotype-phenotype correlations of transporters at the molecular level. Two typical transporters are used as examples in the model to illustrate different aspects. G1 is used to represent common transporter families such as those in the ABC superfamily, for

example, multi-drug-resistance protein (MRP). MRPs are organic anion transporters that transport anionic drugs such as methotrexate, and neural drugs conjugated to acidic ligands such as sulfate (Muller et al., 1994; Jedlitschky et al., 1996; Hipfner et al, 1999). Compounds can be transported by MRPs in complexes with glutathione (GSH). G2 is used to represent other types such as the families of ion channels. For example, intermediate conductance Ca^{2+} -activated K^+ channel (IKCa1, also known with other names KCNN4, IK1, hKCa4, and hSK4) modulates calcium influx by regulating the membrane potential and the driving force for calcium entry. These two kinds of transporters are also used to represent the correlations at two levels, i.e., protein and nucleotide levels.

I use the modeling part of G1 to focus on the description of protein structure and functions. The topologies of these genes include transmembrane domains (TMDs). More detailed topology of transporter proteins is also described, such as the nucleotide-binding domain (NBD), and a “signature” motif that defines the NBDs of ABC transporters (Hyde et al., 1990).

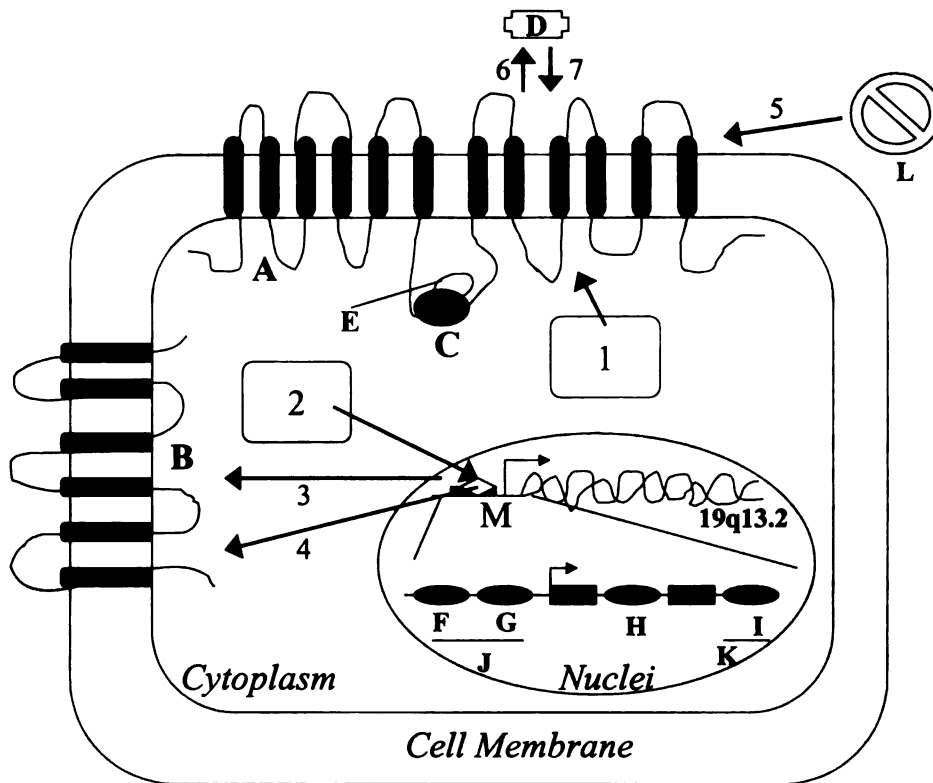


Fig. 2.5 The biomedical model of correlations of transporters at the molecular level

A: G1; B: G2; C: nucleotide binding domain (NBD); D: Drug; E: "Signal" motif; F: Enhancer; G: Promoter; H: Intron; I: Silencer; J: 5'-region; K: 3'-region; L: Inhibitors; M: Regulatory unit. 1: Biochemical pathway; 2: PKC signaling pathway; 3: increase expression; 4: mutation activates expression; 5: inhibit transporter; 6: drug efflux; 7: drug influx.

Elevated levels of G1 can confer resistance to drugs. For example, overexpression of MRP2 was found to result in resistance to cisplatin, etoposide, doxorubicin, and epirubicin (Cui et al., 1999). Because of the potential involvement of these drug pumps in the clinical phenotypes such as drug resistance, inhibitors are also

important in describing transporters' functions. For example, high-affinity substrates can be potent competitive inhibitors, such as leukotriene C4 and S-decylglutathione for MRP1 (Loe et al., 1996; Keppler et al., 1998).

In the modeling part of G2, the structure-function correlation at the nucleotide level is emphasized to represent the common mechanisms in human genes. In the genome, IKCa1 is located at chromosome 19q13.2. IKCa1 can be upregulated through the stimulation of PKC pathway (e.g., in T cells), which can trigger transcriptional activation of the IKCa1 promoter. The regulatory regions of a gene include enhancer, promoter, and silencer. These regulatory units are located in 5'- and 3'-gene flanking regions and in introns. The locations of these regulatory elements and the nucleotide sequences can describe their structure features. Their corresponding functional characteristics can be described in the effect on gene transcriptional activity, and tissue and stage specificities.

The three correlations, especially gene-drug and genotype-phenotype interactions, are described at the systemic level in Fig. 2.6. Abnormal function of transporter proteins can cause abnormal phenotypes, i.e., diseases. Transporters can also be involved in drug response phenotype resistance, toxicity, or normal response. The diagram illustrates the processes involved in drug transport, and the effect of transporter action on the bioavailability of drugs. Drug availability can be controlled by drug absorption and excretion, as shown in the diagram. Besides absorption and excretion, the interaction processes between drug and human body also include distribution and metabolism. In addition, transporters are distributed in different tissues.

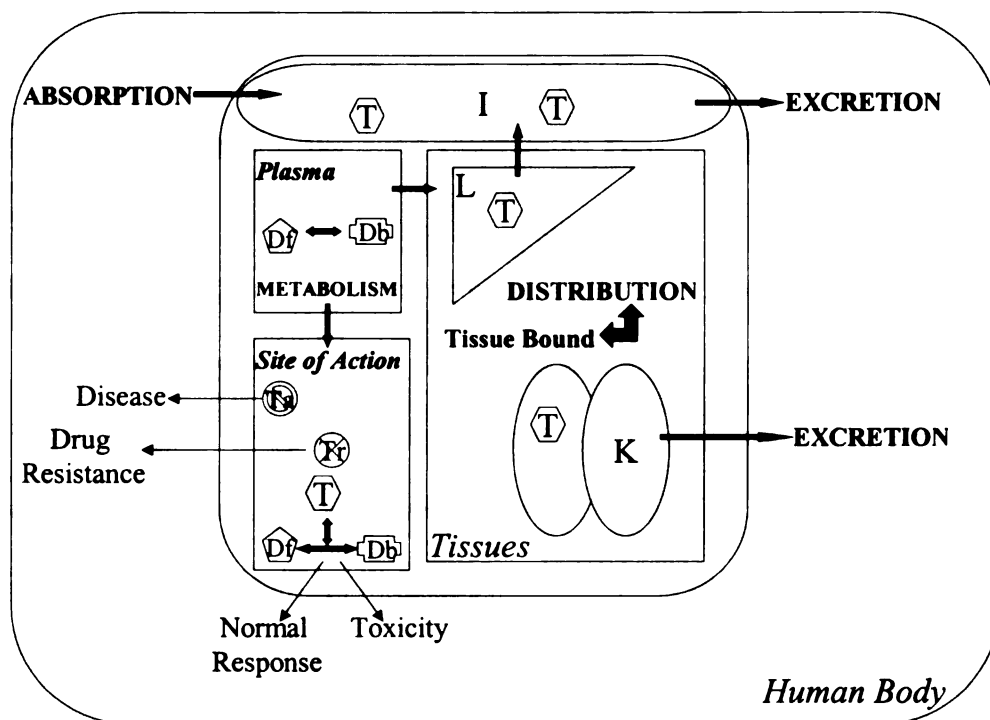


Fig. 2.6 The biomedical model of correlations of transporters at the systemic level

I: Intestines; L: Liver; K: Kidney; Df: Free drug; Db: Bound drug; T: transporter; Ta: Altered transporters (that cause diseases); Tr: Transporters that are responsible for drug resistance.

2.2.3 The Use Case Diagram

With domain requirement analysis and biomedical models built, it is time to perform data modeling with UML. Fig. 2.7 shows the use case diagram of the transporter pharmacogenomics decision support system. The actors in the diagram include molecular biologists, bioinformatician, pharmacologists, drug developers, and

clinicians. The use cases that the system supports include, but not limited to, “Correlate Structure-Function”, “Correlate Gene-Drug”, and “Correlate Genotype-phenotype”. These applications may be the major use cases in our system, although other scenarios are also possible.

The processes in these use cases can be simple data query and retrieving, as well as more complicated knowledge discovery. In the later case, the system is also functioning as a data mining tool besides the fundamental decision support role. Here I describe the use cases from the actor-based view. For each actor, one or several use cases can be involved in system applications. Several actors can also be involved in the same use case.

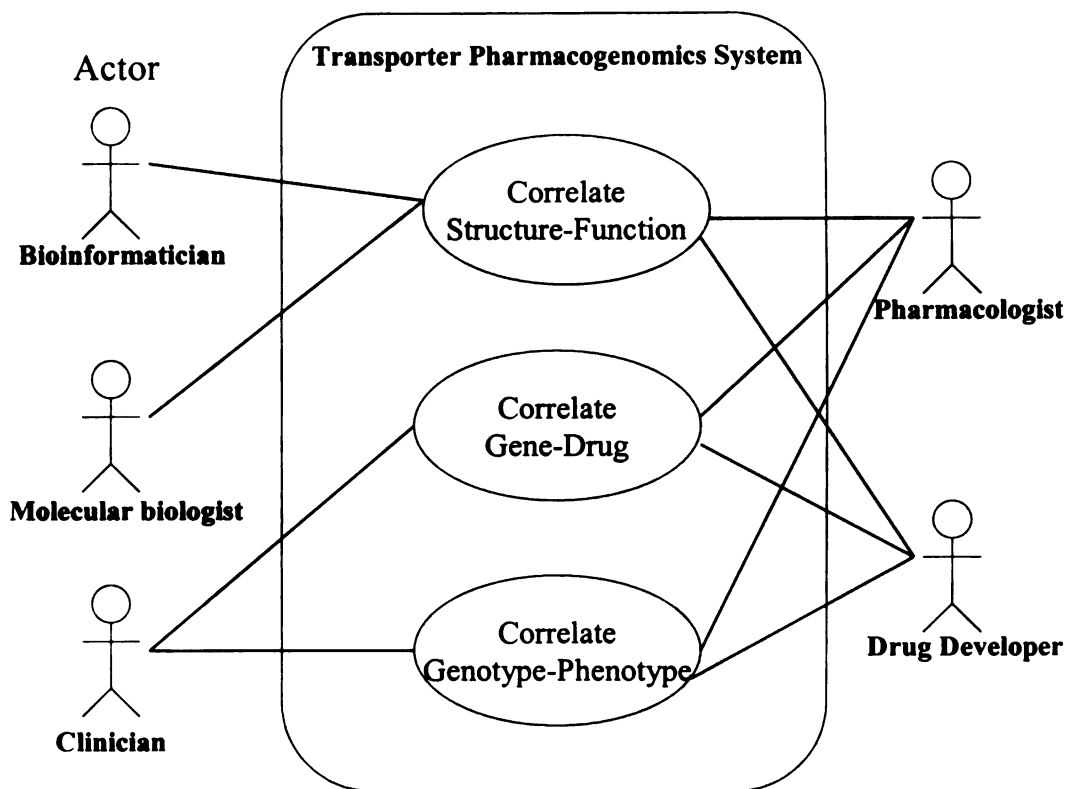


Fig. 2.7 Use case diagram of the Transporter Pharmacogenomics Decision Support System

Bioinformaticians and molecular biologists may want to emphasize their studies on the structure-function relationship. They can enter the names of the transporter genes that they are interested in to the system, and get the information of regulatory elements in these genes, as well as the effects of these regions on transcription activities. They can also perform some data mining based on the information they get from the system. They can retrieve genetic sequence information from the system to analyze homologs and common motifs of polymorphisms, and how the sequence

patterns of variation are related to functional changes such as the altered transporting behaviors and mechanisms.

Pharmacologists and drug developers may be interested in finding the gene-drug interactions. For example, genetic alterations in transporter genes BCRP and MXR are shown to be associated with resistance to mitoxantrone in breast cancer cell lines (Ross et al., 1999). It may be interesting to determine if other transporter genes are also involved in the resistance. Through input of the interested genes, the two groups of actors can find out what drugs are known to interact with these genes and what the downstream effects are. They can also categorize the genes and drugs with known interactions, which might help predict new interactions. For instance, to answer the question “for a new drug, what genes may interact with it?”, analysis of the interaction patterns in drugs with similar structures and functions might be helpful. In this data mining process, the information of the structure-function correlation may also be important.

The study of genotype-phenotype correlation may also be helpful to pharmacologists and drug developers to get some feedback about the use of drugs. This correlation information can assist more accurate drug targeting in the drug design process. For example, the identification of potential gene markers in the drug resistance phenotype may provide clues for these actors to design new drugs targeting these markers to reverse or overcome the resistance.

Clinicians may need to find out the information about both gene-drug interactions and genotype-phenotype correlations. A patient’s genetic profile can be used to predict phenotypic response to specific drugs and select the optimal medication

for the treatment. For this purpose, they may first need to know the candidate genetic markers of certain phenotypic responses (such as resistance) to specific drugs. They can use the system to retrieve the genetic information through the input of interested drugs and genotypes. This is a complex and integrated process with information of both of the two correlations involved. With the comparison of the patient's genetic profiles and candidate gene markers, they can select drugs that might have the least side effects and the strongest treating effects. The software system will be a very useful tool in these decision making processes.

2.2.4 Class Diagrams

2.2.4.1 Concept Abstraction

From the biomedical models, major concepts can be abstracted for further modeling usage. I designed a Biomedical Concept Category List to facilitate the abstraction process in a systematic way. Some example concepts in each category are shown in Table 2.2. The operation process is to abstract the concepts in the biomedical model (in Fig. 2.5 and Fig. 2.6) to fill in the categorical list. In addition to this method, noun phrases related to the current requirements under consideration can also be abstracted from the biomedical models. For example, noun concepts Topology, Motif, and Promoter can be abstracted from Fig. 2.5.

Category	Examples
Objects	Gene Drug
Classification	TransporterName Family
Physical structure	RegulatoryElement Drug2DStructure
Places	ChromosomeLocation TissueDistribution
Variations	Polymorphism Mutation
Interactions	DrugDrugInteraction GeneDrugInteraction
Processes	BiochemPathway SignalPathway
Mechanisms	TransportMechanism DrugMechanism
Functions	PhysiologicalRole Inhibitors
Phenotypic Characteristics	Disease DrugResistance
Correlations	Structure-Function Gene-Drug

Table 2.2 The Biomedical Concept Category List

2.2.4.2 Data Modeling of the Structure-Function Correlation

From concept abstraction of the biomedical models, classes and associations involved in the structure-function correlation can be identified. Fig. 2.8 shows the class diagram describing these classes and associations. The major classes are identified around nucleotide and protein sequences and structures, and their associated functions.

As shown in Fig. 2.8, gene, protein, and related genome information are needed to study how the structure affects function. To clarify concepts and facilitate further

studies, the classification of the genes should be defined. Because in many cases one transporter gene can have many different names, transporter symbols, their complete names, alternative symbols, and names in the standard nomenclature system are listed in the “Gene” class. The family and superfamily that the gene is in, and the family identification number (TC number, will be explained in detail in later chapters) are also included in this class.

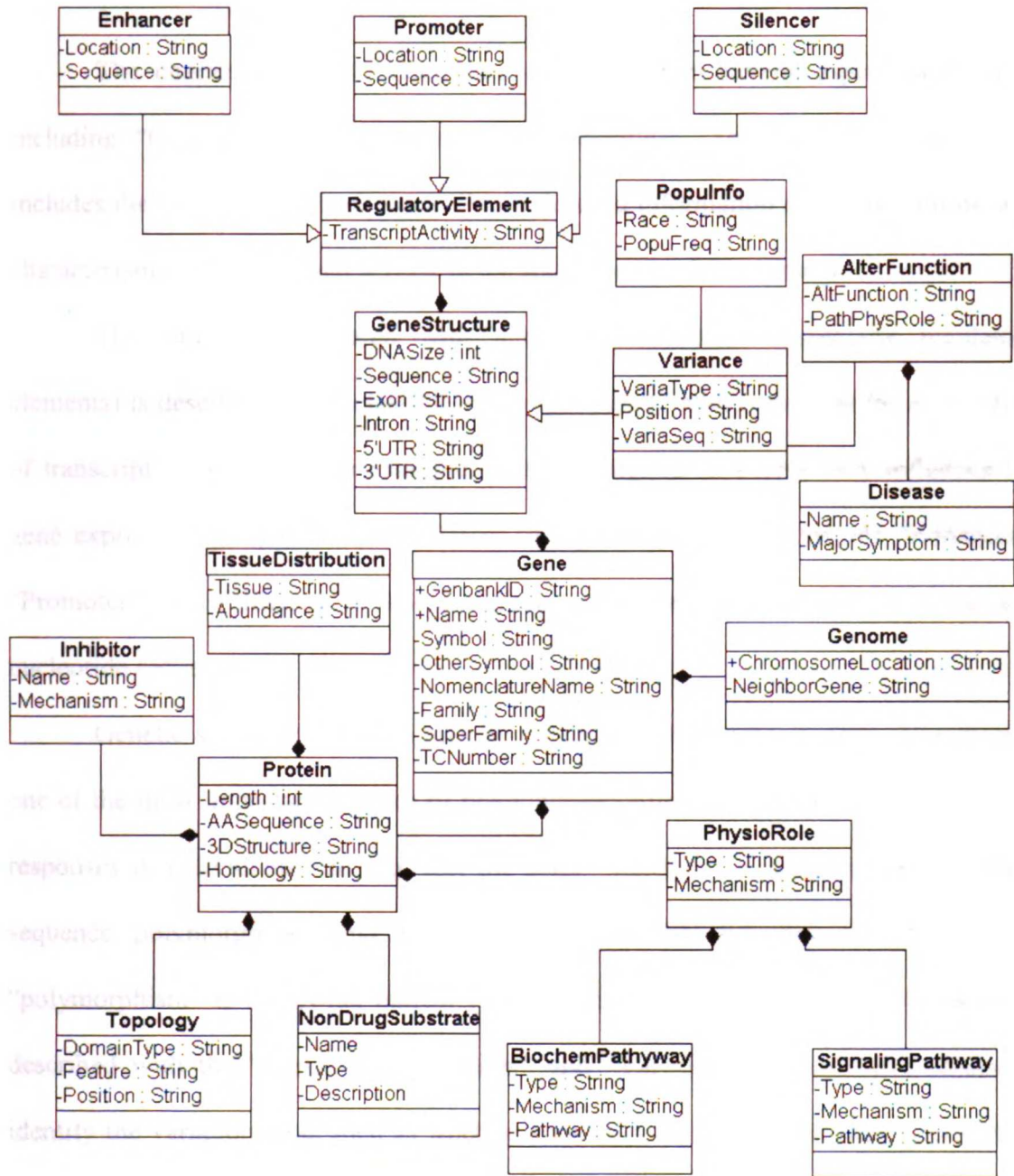


Fig. 2.8 The class diagram of the structure-function correlation

The characteristics of “Gene” are described from several aspects and levels including “GeneStructure”, “Protein”, and “Genome”. The “GeneStructure” class includes the DNA size and sequence. More detailed information of exons, introns, and characteristics of 5’ UTR and 3’UTR in the sequence may be necessary.

The function correlated with nucleotide structures (especially regulatory elements) is described in the class “RegulatoryElement” which includes the description of transcription activity. For example, promoter regions in a gene may influence the gene expression level. So the subclasses of regulatory elements include “Enhancer”, “Promoter”, and “Silencer”. These elements are described by their locations and nucleotide sequences.

Genetic sequence variation may be crucial in functional variation. Variation is one of the most important features of pharmacogenomics, which studies different drug responses in individuals. In Fig. 2.8, the class “Variance” is used to represent both sequence polymorphisms and mutations, because the definition of the concept “polymorphism” is somewhat narrow (Yan and Sadee, 2000). Sequence variation is described with the “VariaType”, “Position”, and “VariaSeq”. These three attributes identify the variation type, such as whether a variation is mutation or single nucleotide polymorphism, and the position of the sequence change, as well as the varied sequence. The population information of variance (especially polymorphisms) is used for analysis and selection of certain groups of patients. Sequence variation can alter the transporter function. The altered function, and the pathological role of the variation are represented in the class “AlterFunction”.

The chromosome information may connect the gene with neighbor genes in the genome. The position of the gene in the whole genome is identified through the description of the chromosome location and neighbor genes in the “Genome” class.

The classes in the protein structure domain include “Protein” and “Topology”. “Protein” includes amino acids sequence, protein length, 3D structure, and homology. “Topology” includes the domain type in the protein (such as transmembrane domains (TMDs)). This class also includes descriptions of the positions and features of the domain, as those functional domains are also called motifs that are necessary for the analysis of structure-function association. The abundance of transporter proteins in different tissues is described in the class “TissueDistribution”.

The functions correlated with protein structures are described in the classes “PhysioRole”, “NonDrugSubstrate”, and “Inhibitor”. “PhysioRole” describes the physiological functions of the transporter gene, such as their roles in the regulation of intracellular redox potential, in the process of ion flux, and in the elimination of endo- and xenobiotics. This class includes the transporting mechanisms such as whether it is energy dependent. The subclasses “BiochemPathway” and “SignalingPathway” put the gene in the whole picture of its functioning processes. It is described with the type of the pathway and the pathway processes that the transporter is involved in.

“NonDrugSubstrates” describes non-drug substrates including nutrients that are known to interact with the transporter. The details of drug substrates will be described in the gene-drug correlation domain. “Inhibitor” describes those molecules that can inhibit the transporter function, such as those examples in Fig. 2.5.

2.2.4.3 Data Modeling of the Gene-Drug Interaction

The major concepts involved in the gene-drug interaction include gene, drug, and their correlation. In the last section, both structure and function aspects of gene are described. This section will emphasize drug information and the correlation with genes.

Drug information such as drug structure and mechanisms are crucial for the understanding of drug resistance and toxicity mechanisms. The information can be very helpful for predicting the response of new drugs with similar structure or action. Drug information is also important for designing strategies to reverse the resistance or toxicity associated with treatment failure. The model representing drug information is illustrated in Fig. 2.9.

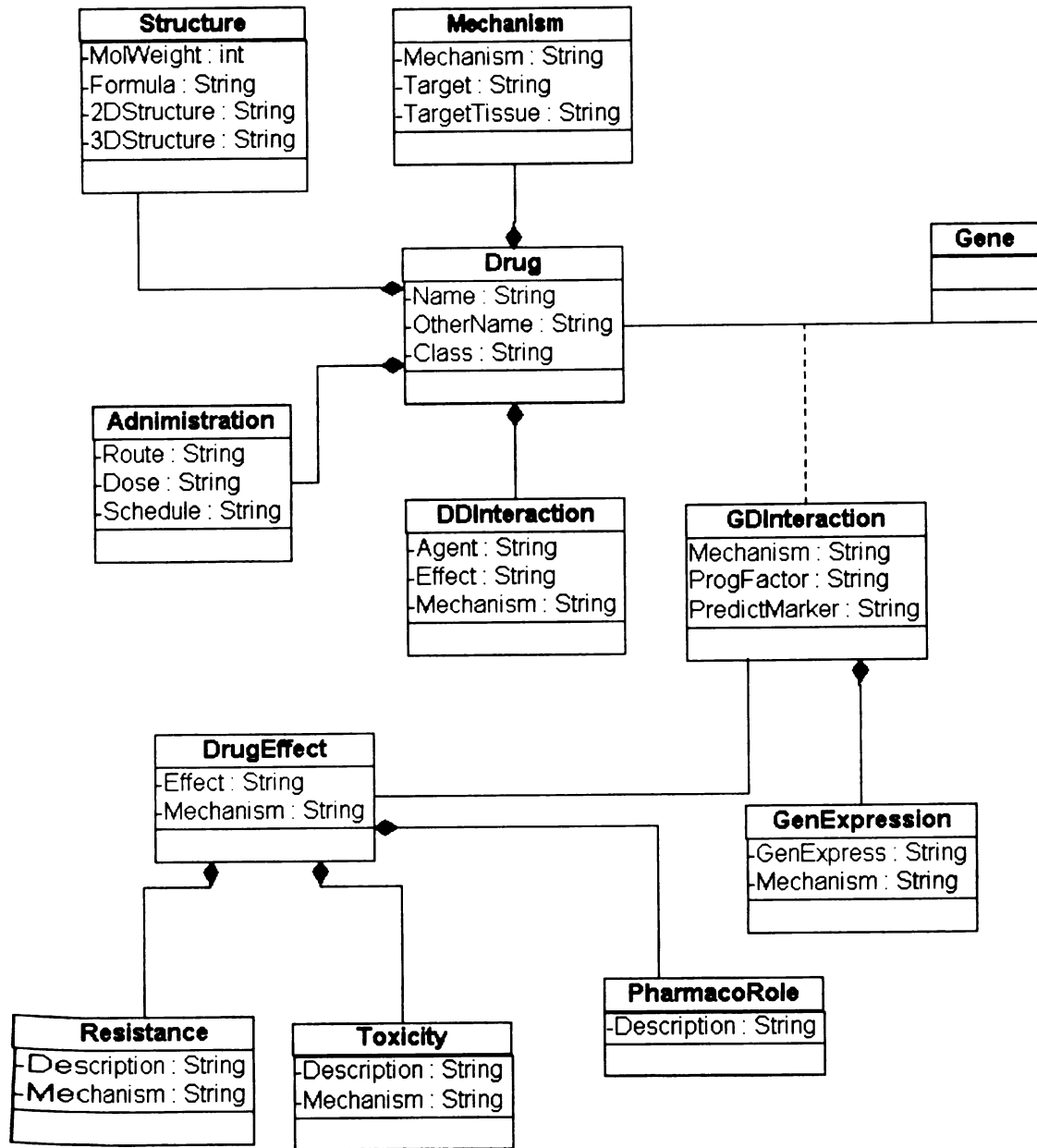


Fig. 2.9 The class diagram of gene-drug interaction

The major class “Drug” has attributes of drug name, other names, and drug class. “Structure”, “Mechanism”, “Administration”, and “DDInteraction” describe several aspects of a drug. The drug “Structure” includes molecular weight and chemical formula. Both 2D and 3D structures of a drug are also included in the description, as they may be useful for drug designers in considering its interaction with a 3D protein. “Mechanism” contains drug activities and target information. “DDInteraction” means drug-drug interaction, which describes the interactions between this specific drug and other agents, as well as the interaction mechanisms and effects. “Administration” includes the dosage, route, and schedule information, as these may be useful for both clinicians and drug developers.

The interactions between gene/human and drug in Fig. 2.5 and Fig. 2.6 incorporate two levels of response to drugs, both genotypic and phenotypic. The later level will be discussed in the next section. The class “GDInteraction” describes the overall characteristics and mechanisms of the gene-drug interaction. The significance of such response is also included, such as whether this interaction is considered to be involved as a disease prognosis factor, or a drug response prediction marker.

Because the gene-drug interaction is mutual, both gene and drug can have responsive reactions influenced by each other. Considering the drug side, the actions of a drug may be changed by genetic alterations. For example, increased drug efflux or decreased drug influx may be caused by transporter variation, as illustrated in the biomedical model Fig. 2.5. The class “DrugEffect” in Fig. 2.9 represents this kind of interaction result, with the description of the mechanism of the changes and the consequent effects.

The other side of the gene-drug interaction is “gene”, which should also be described. This side of information is represented in the class “GenExpression”, which is described with the gene expression level, and the mechanisms involved in response to a drug, such as increased or reduced expression levels. For example, over-expressed breast cancer resistance protein (BCRP) was found to mediate resistance to mitoxantrone in breast cancer therapy (Ross, 1999).

Fig. 2.8 and 2.9 can be integrated to be one complete model. This can be done with extending and connecting the class “Gene” (in Fig. 2.9) to Fig. 2.8.

2.2.4.4 Data Modeling of the Genotype-Phenotype Correlation

As illustrated in Fig. 2.4, three types of phenotypes can be involved in the genotype-phenotype correlation, including normal, disease, and the drug-response phenotype. In our pharmacogenomics studies, we emphasized the disease and drug-response aspects. Because the genotype-phenotype correlation is linked to structure-function and gene-drug correlations, the disease and drug-response aspects at this level of correlation are integrated in the two later correlations, as shown in Fig. 2.8 and 2.9 respectively. Since the genotype side in the correlation has been described in previous sections, here I will emphasize the phenotype side and the correlation itself.

The disease aspect of phenotype is represented in class “Disease” in Fig. 2.8. This phenotypic response is correlated with genotypic factors through the correlation

with the class “AlterFunction”. The “Disease” class is described with disease names and major symptoms.

The overall drug-response phenotypes are correlated with genotypic gene-drug interactions through class “DrugEffect”. These phenotypes include “Resistance” and “Toxicity”, as illustrated in Fig. 2.9. Their attributes include mechanisms and descriptions of the phenotype. If we abstract the concepts in Fig. 2.6, drug activation, inactivation (such as clearance), absorption, distribution (include transportation), can be analyzed and described in the class “PharmacRole”, which describe the pharmacological role in the phenotypic response.

To evaluate our data model and see if it captures the most important aspects in our targeting knowledge domain, we can check back with our original biological facts in Fig. 2.5 and 2.6, and some research reports in the literature. We can see that the object model is consistent with the biological facts and domain knowledge. For example, most of the information in Fig. 2.5, such as the types of the molecules involved, is represented and included in Fig. 2.8 and 2.9. In the implementation of the system, if some modifications are found necessary, the model constructed here can still be changed and improved.

2.2.4.5 Biomedical Design Patterns

How to design bioinformatics systems in different subdomains and for different usages is a recurring problem. The class diagram described here has generic significance in the whole biomedicine. The classes, their attributes, and the

relationships can be used as building blocks for other biomedical system design. For example in Fig. 2.8, the block of classes “Gene”, “GeneStructure”, “RegulatoryElements”, “Enhancer”, “Promoter”, “Silencer” and the associations among them compose a pattern of “Nucleotide Structure-Function Relationship”. Each class in this block itself can also be a pattern for specific usage, such as for a system exclusively for promoters.

The block of classes “GeneStructure”, “Variance”, “PopuInfo”, “AlterFunction”, and “Disease” forms a typical pattern that can be reused for other pharmacogenomics systems. Because the design model here reflects the correlations and components that are typical in biomedicine, recombination of these correlations and components can be included in other systems for various purposes so they would not have to be rebuilt again from the scratch.

2.2.5 Sequence Diagram and A Real World Example

The class diagrams identify the classes and associations in the transporter pharmacogenomics decision support system (TPDS). The next step is to analyze the message flow among these classes and how the system works during the usage by a user. A sequence diagram describes the dynamic procedure of a specific scenario that can show us a real application of our OO model. Fig. 2.10 shows the sequence diagram of “identifying candidate genetic markers in tamoxifen resistance”. In this scenario, a user, such as a pharmacologist or a drug developer, wants to find the mechanisms and design strategies (such as new drugs) to reverse the resistance to tomoxifen in breast

cancer therapy. To do this, they need to know the candidate genes that may be responsible for tamoxifen resistance in breast cancer therapy. These genes can be potential targets for the reversal strategies.

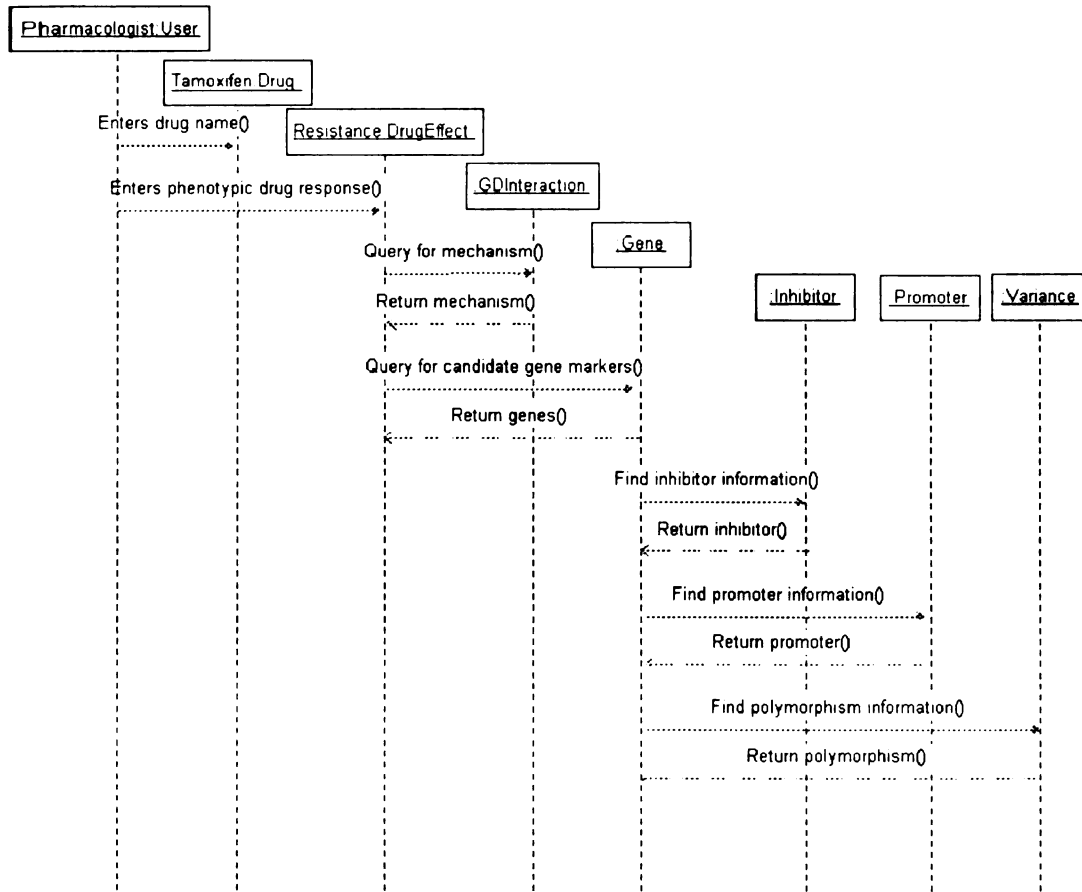


Fig. 2.10 The sequence diagram of the scenario “identifying candidate genetic markers in tamoxifen resistance”

The major classes involved in this scenario include “Gene”, “Drug”, “GDInteraction”, and “DrugEffect”. “Tamoxifen” is an instance of the class “Drug”. “Resistance” is a phenotypic response of drug effect in gene-drug interaction. A list of potential gene markers responsible for this phenotype is returned when the user selects the response phenotype and the drug name. The user can then make further queries for detailed information in these genes that are involved in the resistance phenotype.

This is a typical scenario with all the three correlations involved. The goal in this decision making process is to find *genotypes* that may be responsible for the resistance *phenotype*. With the drug name tamoxifen at the left side as input, what needs to be found out is the other side of the *drug-gene* interaction, the genes. The mechanisms of how the drug actions are influenced by the genes can also be retrieved from this interaction. Once the candidate genes and the mechanisms are known, the information about the *structure-function* correlation is needed to identify the possible targets for finding reversal mechanisms. Such information may include the known inhibitors of the genes, the functioning motifs, and regulatory elements that affect the transcription activities, as well as variations including genetic polymorphisms for individualized strategies.

Now we can see the decision support process in this scenario. For instance, a user is exploring strategies to reverse tamoxifen resistance in breast cancer therapy. One way to do this is to use the TPDS system to find out the causal mechanisms and get the candidate gene markers that can be used as targets of reversal. The user first queries the system through entering the drug name “Tamoxifen”, whose information is in the “Drug” class. The user then enters the possible phenotypic response “Resistance” to see

the possible gene markers. The system looks for the information from the classes “DrugEffect”, “GDInteraction”, and “Gene”, and extracts the genes that show altered expression as possible markers in the tamoxifen-resistance effect.

In the process of getting this information from the system, the user also wants to ask how the genetic factors affect the drug actions. This mechanism information can be retrieved from the class “GDInteraction”. Now the user is ready to look for possible reversal targets such as some regulatory elements in the genes that can be used to inhibit the resistance effect. To do this, the information of “Inhibitor”, and sequences of “Promoter” can be retrieved. To make the strategies specific for different individuals, the user wants to know the genetic variance that may occur in these genes. This kind of information can be obtained from the “Variance” class. Now armed with all the information provided by TPDS, the user is ready to design reversal strategies.

This scenario describes how the information is extracted for the three correlations, and the information flow among the objects. These applications may be also useful for clinicians to subgroup their patients for different treatments based on their genotypes, and to select the optimal regimen according to patients’ genetic profile. For example, if a patient’s genotypic profiles suggest possible resistance to tamoxifen, other choices of drugs could be considered. Although not all classes and associations are involved, this example shows how a complicated problem can be solved step by step with a decision support system.

2.3 Summary

The object-oriented UML approach provides data modeling capabilities and supports a systematic methodology for pharmacogenomics studies in transporters. For example, who will use the TPDS information and how they need to use them are among the most important considerations for the system design. Use case analysis elucidates these considerations and sets up the boundary of our problem. This methodology helps present data for multiple users including drug designers, pharmacologists, genetic researchers, clinicians, and microarray examination designers. A good methodology not only benefits software developers, but also can improve and broaden the applications of the system by the users. It can give analysts the information necessary to make sound decisions about strategic issues for research, drug development, and treatment.

The model presented here is an attempt to provide the foundation for a comprehensive system that brings pharmacogenomics into the clinic and to benefit patients more directly. The model takes into account “variation” beyond the gene sequence, since “sequence variation is only one parameter, and certainly not the dominant parameter in human variation.” (Marshall, 1997) By OO modeling using UML, we approach the problem of transporter pharmacogenomics from different angles that provide a more complete view.

The construction of the model demonstrates that UML, a modeling language that has been widely used in the business information technology (IT) industry, may be an appropriate common literacy in biomedicine if applied with appropriate methodologies. The main difficulty of applying UML in the biomedical domain is the barrier of domain knowledge. The “requirement analysis and biomedical models \Rightarrow concept abstraction

and object design \Rightarrow UML OO models” methodology developed here is a useful measure for knowledge modeling. This methodology provides a generic way in how to build a computer OO model from the crude domain knowledge.

In biomedical science, knowledge modeling could play the role of mathematical modeling in physical sciences (Rechenmann, 2000). Just as mathematicians, biomedical informaticians study models that are abstractions from the real biomedical world. As mathematical modeling encodes knowledge in a dynamic physical system, knowledge modeling (as the example shown here) in biomedical systems also embraces both structural and dynamic behavioral aspects. This characteristic allows the dynamic representation of interactions and can be especially useful for the study of structure-function, gene-drug, and genotype-phenotype associations in transporter pharmacogenomics (as shown in Fig. 2.10).

Chapter 3. The Development of Transporter Pharmacogenomics

Decision Support System (TPDS)

Systems succeed for a variety of reasons. They fail for one: lack of architecture.

—Grady Booch

3.1 The System Architecture for The Web Application of TPDS

The data modeling process described earlier provides a relatively complete view of the abstracted concepts in the knowledge domain. Based on this data model, a prototype of the transporter pharmacogenomics decision support system (TPDS) can be built. A prototype is a core application that can be expanded and enhanced into a full-blown application.

To facilitate the application and access by multiple users, the TPDS prototype is accessible from the Web. To support the Web applications of TPDS, a three-tiered client-server architecture was constructed, as illustrated in Fig. 3.1. This is an overall architecture of the whole system. The data in TPDS is stored and managed in a database, Human Membrane Transporter Database (HMTD). Here Web tools and databases are two distinct technologies that are developed separately. Web tools include the Web browser and Web Server. The browser, such as Netscape Navigator or Microsoft Internet Explorer (IE), exhibits HTML pages through interpreting the HTML tags. The Web server is used to deliver HTML pages.

The three-tiered architecture has advantages in reducing network traffic, making components interchangeable, and increasing security. Client-server is a computing architecture in which client-processes ask for services and data from server-processes. Clients and servers can be existent in the same memory space. They can also be on separate computers, exchanging messages and data through a network. Two-tiered client-server architecture used to be the most general architecture that performed all the necessary functions of an application between “fat clients” and “fat servers”. The three-tiered architecture brings in a third layer (a middle layer) of processing between the client and the server. This kind of architecture allows the client and the server to become “thin clients” and “thin servers”, which enables the partitioning of application functions to be carried further to achieve better modularity. Processes such as transaction control, application logic, and report generation are now implemented in the middle layer.

As shown in Fig. 3.1, the client tier of the TPDS system is accessed by Web browsers on all kinds of machines. The user interface of TPDS is written in standard HTML that can be rendered in any browser. This HTML file is stored in the middle tier and displayed in the client tier. There can be multiple clients accessing the system at the same time through the HTML page on a Web browser. The server tier is a database management system (HMTD) that stores and manages data. The middle tier consists of a Web server and a Common Gateway Interface (CGI) program written in Java. The connection from the CGI program to the database HMTD is through the JDBC (Java Database Connection) driver. The client tier is connected with the Web server in the middle tier through the Internet.

A Web server communicates with the CGI program through environmental variables and the operating system's standard input. URL parameters and the user's IP address are passed via environmental variables. User-input from forms on a HTML page is passed via standard input. JDBC is an application programming interface (API) designed to provide a way to access data from Java programs. JDBC provides Java methods that will pass query strings to the underlying database management system (DBMS) driver. This enables a Java application to make full use of the database's native calls.

The process of a web database application begins with the Web browser sending Web page requests or data requests to the Web server. The Web Server processes the page requests and delivers the data request to the CGI program. The CGI program then accepts the requests, converts them to a form (such as SQL statements) that the database will recognize, and passes them to the database server. The database server then executes a database task, such as a query, and returns the result set to CGI. Next, the CGI program converts the database results to a HTML form that the Web server will accept, and forwards them to the Web server. Finally, the Web server passes the form to the Web browser, which is the result seen by the user.

In this chapter, I will describe how the database was constructed, where the data came from, and how the data was transformed into information. The application of the system, the user interface, and the HTML pages will be discussed in detail in the next chapter.

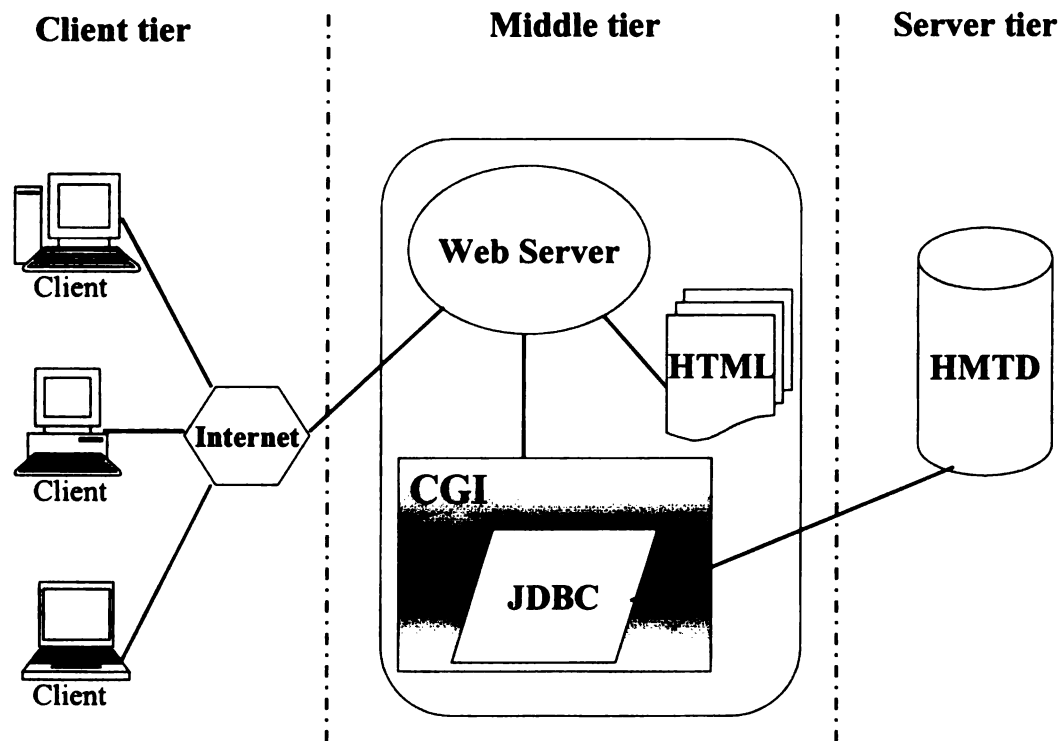


Fig. 3.1 The three-tiered client-server architecture of TPDS

3.2 The Logical and Physical Data Processing in TPDS

The transformation from data to information is through a complex route. Fig. 3.2 briefly describes the three-layer logical process of data transformation in the TPDS prototype. This is a data “evolution” process from “untreated” crude format to the ready-to-use information for decision support. The lowest layer is crude data from all kinds of data sources, including literature such as journal articles and books, as well as on-line databases such as GenBank and OMIM. The middle layer is the physical

database part (HMTD). The highest layer is the decision support layer that provides supporting functions for users in laboratories, clinics, and pharmaceutical companies.

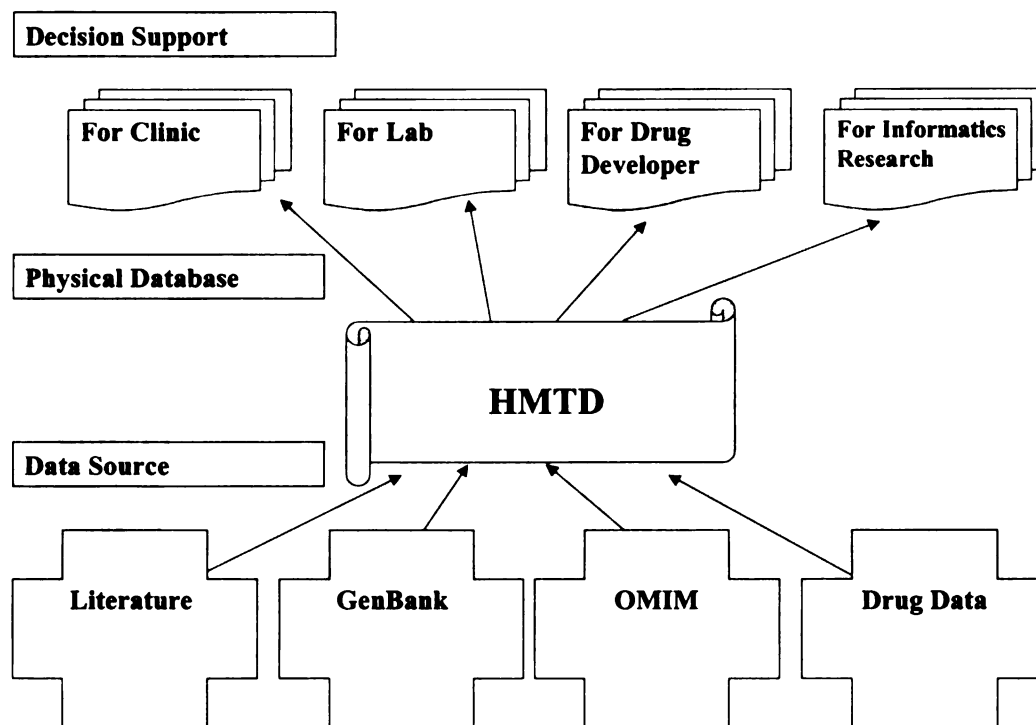


Fig. 3.2 The three-layer logical process of data transformation in TPDS

This logical data process is put into operation through the physical data architecture of TPDS, which is indicated in Fig. 3.3. This diagram illustrates at a high level about how all of the data components within the system are integrated. Such data architecture provides a framework through identifying how data will move throughout the system and be utilized. As shown in Fig.3.3, data is extracted from source systems, including different kinds of biomedical databases and flat files. The data from source

systems is cleaned before loading into the database system HMTD. The data cleansing process is to eliminate duplications and reconcile differences between various styles of data collection. Data from different databases will be processed, transformed, and then integrated into HMTD. The data organized by different subjects contains only the information necessary for decision support. The cleaned-format information can then be accessed by end-users for analysis, reporting, and mining.

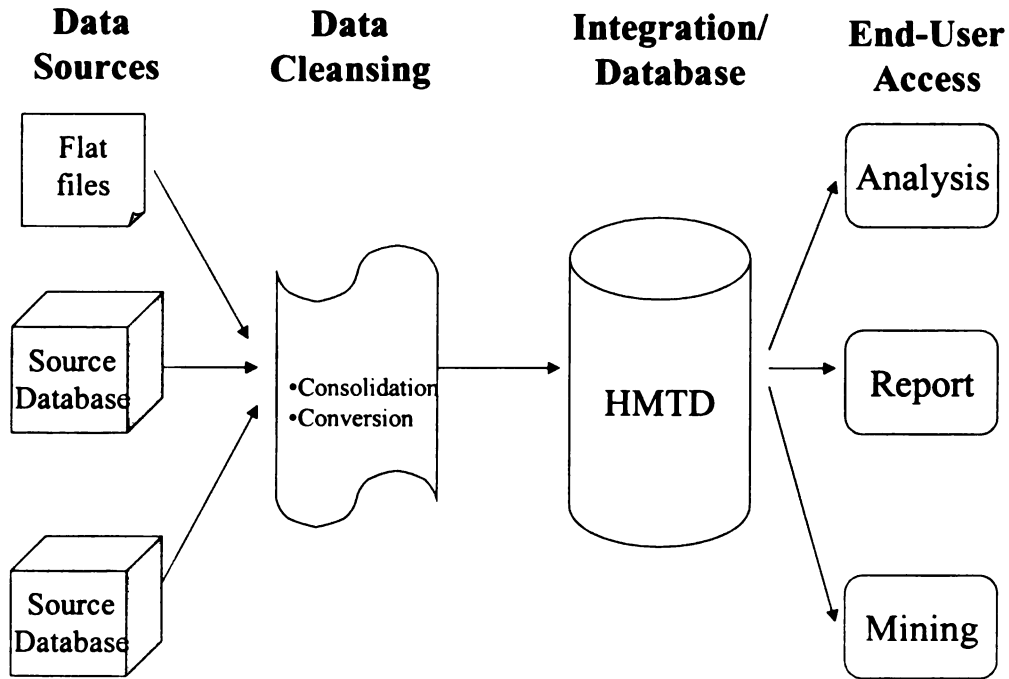


Fig. 3.3 The data architecture of TPDS

3.3 Java Coding of Data Definitions

Implementation of the system in an object-oriented programming language requires writing source code for class definitions. From the data model defined in the design phase (as described in the previous chapter), Java code can be generated to provide program solutions and data definitions. This process translates UML artifacts created during the design phase, such as class diagrams, to code for class definitions. This is an extension method from object-oriented analysis and design to object-oriented programming. Such a method has strengths in that it provides a complete end-to-end roadmap, from the very beginning of requirement analysis to the later application coding procedure. The class definition here can be directly integrated into the JDBC coding process (as illustrated in Fig. 3.1).

Fig. 3.4 shows some sample code for several classes in TPDS. The code here is reusable for bioinformatics projects in other areas.

- **Class: GenStructure**

```
public class GenStructure
{
    public GenStructure()
    {
        super();
    }
    public final int getDNASize()
    {
        return mDNASize;
    }
    public final void setDNASize(int the_mDNASize)
    {
        this.mDNASize = the_mDNASize;
    }
    public final String getSequence(String the_mSequence)
```

```

    {
        return mSequence;
    }
    public final void setSequence(String the_mSequence)
    {
        this.mSequence = the_mSequence;
    }
    public final  getExon()
    {
        return mExon;
    }
    public final void setExon(String the_mExon)
    {
        this.mExon = the_mExon;
    }
    public final String getIntron()
    {
        return mIntron;
    }
    public final void setIntron(String the_mIntron)
    {
        this.mIntron = the_mIntron;
    }
    public final String get5'UTR()
    {
        return m5'UTR;
    }
    public final void set5'UTR(String the_m5'UTR)
    {
        this.m5'UTR = the_m5'UTR;
    }
    public final String get3'UTR()
    {
        return m3'UTR;
    }
    public final void set3'UTR(String the_m3'UTR)
    {
        this.m3'UTR = the_m3'UTR;
    }

    private int mDNASize;

    private String mSequence;

    private String mExon;

```

```

private String mIntron;

private String m5'UTR;

private String m3'UTR;

}

```

- **Class: Variance**

```

public class Variation extends GenStructure
{
    public Variation()
    {
        super();
    }
    public final String getVariaType()
    {
        return mVariaType;
    }
    public final void setVariaType(String the_mVariaType)
    {
        this.mVariaType = the_mVariaType;
    }
    public final String getPosition()
    {
        return mPosition;
    }
    public final void setPosition(String the_mPosition)
    {
        this.mPosition = the_mPosition;
    }
    public final String getVariaSeq()
    {
        return mVariaSeq;
    }
    public final void setVariaSeq(String the_mVariaSeq)
    {
        this.mVariaSeq = the_mVariaSeq;
    }

    private String mVariaType;

    private String mPosition;
}

```



```
        private String mVariaSeq;
    }
}
```

- **Class: Drug**

```
public class Drug
{
    public Drug()
    {
        super();
    }
    public final String getName()
    {
        return mName;
    }
    public final void setName(String the_mName)
    {
        this.mName = the_mName;
    }
    public final String getOtherName()
    {
        return mOtherName;
    }
    public final void setOtherName(String the_mOtherName)
    {
        this.mOtherName = the_mOtherName;
    }
    public final String getClass()
    {
        return mClass;
    }
    public final void setClass(String the_mClass)
    {
        this.mClass = the_mClass;
    }

    private String mName;

    private String mOtherName;

    private String mClass;
}
}
```

- **Class: Resistance**

```
public class Resistance
{
    public Resistance()
    {
        super();
    }
    public final String getDescription()
    {
        return mDescription;
    }
    public final void setDescription(String
the_mDescription)
    {
        this.mDescription = the_mDescription;
    }
    public final String getMechanism()
    {
        return mMechanism;
    }
    public final void setMechanism(String the_mMechanism)
    {
        this.mMechanism = the_mMechanism;
    }

    private String mDescription;

    private String mMechanism;
}
}
```

Fig. 3.4 Sample Java code for data definition in TPDS

3.4 Physical Database Development

3.4.1 Database Design and The Schema of HMTD

The data model designed in the last chapter is implemented physically in a database HMTD. Fig. 3.5 shows the schema of the database in the format of an extended entity-relationship (EER) diagram. The top line of each box in Fig 3.5 represents the entity of a table. Here an entity type is “a thing which can be physically defined” (Chen, 1976). This also includes abstract concepts, such as “Function”. Attributes that describe all the properties of interest of an entity are listed under the entity. Generally attributes are identified as qualities of an entity in a text description and include all the extensional information in a database. A relationship is “an association among entities”, which is represented by a line connection between two tables (Chen, 1976).

In the diagram, the primary key is indicated through a key figure before an attribute. A primary key is an attribute that uniquely identifies tuples in a relation. For example, “GeneID” is the primary key of the table “Gene”. An attribute in one relation, that is drawn from the same domain as the primary key of the related relation, is called a foreign key. For example, “GeneID” is a foreign key in the table “Protein”.

In the design of the database schema, the integrity of a relation is managed by two rules. One is the entity constraint, which states that no part of the primary key can have a null value. The other rule is referential integrity, that is, the value of a foreign key should either be null, or equate to an existing value of the primary key in the related relation.

Generally in the development of a database, updating large numbers of values in response to a single change is inefficient and error-prone. To avoid such a situation and

minimize the necessary storage, the smallest number of values required to represent the semantics of the database are stored. Thus in the design process of the schema here, normalization is performed to produce a set of relations that contain minimum redundancy.

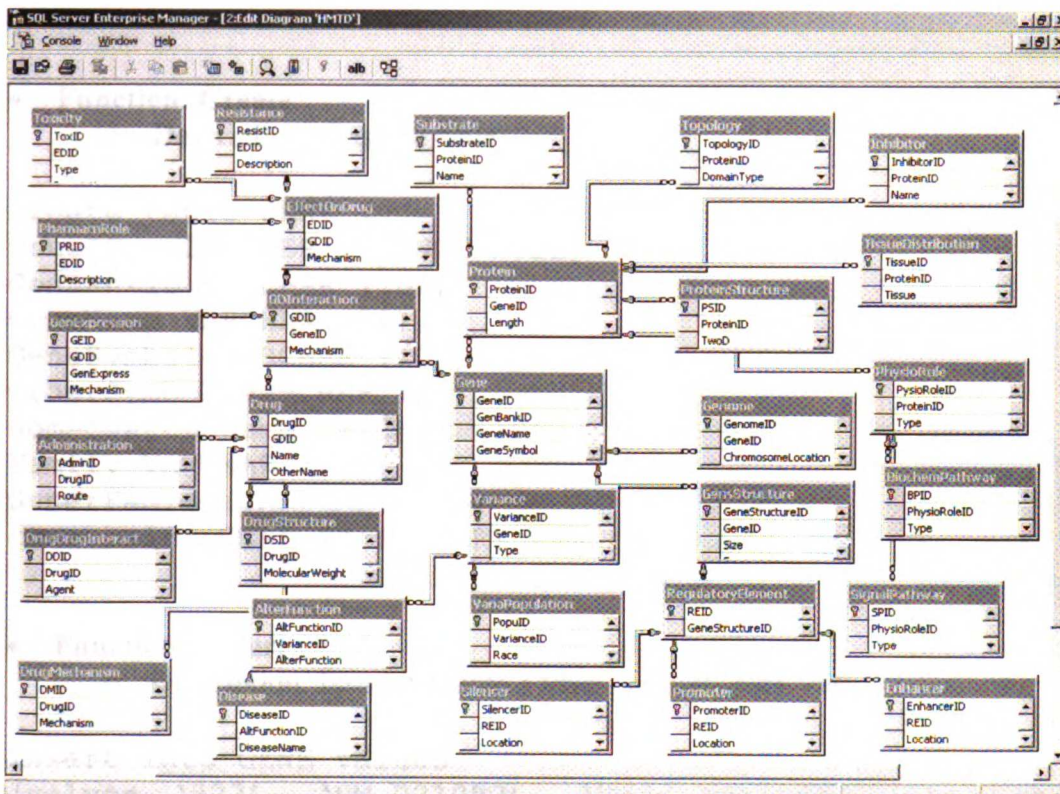


Fig. 3.5 The ER diagram of TPDS

3.4.2 *Creating the Database with SQL*

With the design of the database, HMTD is constructed using the structured query language (SQL). Fig. 3.6 lists some sample SQL statements for the creation of the database. This list also includes some sample code for alteration and update of the database, and for data integration. Thus the database is flexible and readily accommodates changes and is expandable.

- **Function: Create**

- **Description:** Create a table

```
create table GENE (  
GeneID INTIGER ( ) NOT NULL,  
GenBankID VARCHAR (10),  
GeneName VARCHAR (50),  
GeneSymbol VARCHAR (10),  
OtherSymbol VARCHAR (50),  
NomeclatureName VARCHAR (10),  
Family VARCHAR (20),  
SuperFamily VARCHAR (20),  
constraint GENE_PK PRIMARY KEY (GeneID)  
);
```

- **Function: Insert**

- **Description:** Insert data into a table, used for data entry

```
insert into GENE values  
(values '323', 'NM_021082', 'Homo sapiens solute carrier  
family 15 (H+/peptide transporter)', 'PEPT2', 'hPEPT2',  
'SLC15A2', 'Peptide Uptake Transporter', 'ABC');
```

- **Function: Alter**

- **Description:** Alter tables by adding a column (or by changing a column's definition, or dropping a column), thus a database is flexible for change

```
alter table Gene add(  
TCNumber VARCHAR (20)
```

);

- **Function: Update**

- Description: Update data in a table

```
update GENOME set ChromosomeLocation=12p13.3
where GeneID = '11';
```

- **Function: Translate**

- Description: Converts characters, used in data integration

```
select TRANSLATE('widely', 'widely', 'ubiquitous' ) AS
Tissue
From TISSUEDISTRIBUTION;
```

Fig. 3.6 Sample SQL statements for the database creation and update

3.5 Data Sources and Collection

Having constructed the database, we may now start entering data. However, data is not readily available for direct entering. The data for HMTD may come from many sources. As Fig. 3.7 shows, data comes in many forms, in many formats, and from multiple systems, even from competing sources. In addition, data format and content may change over time. Some data has missing and incomplete fields. Before the data entry process, we have to evaluate all the available sources, identify the right sources of data, and bring the right data together. This step is crucial to facilitate further data analysis.

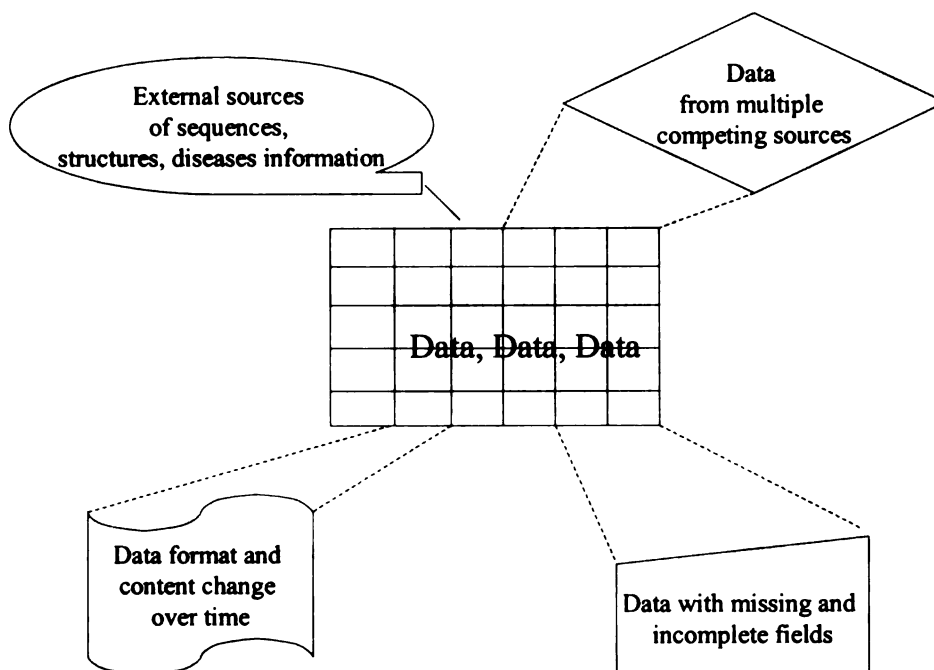


Fig. 3.7 Data is available from different sources

Fig. 3.8 illustrates the strategy to find the appropriate data sources. The data model described in the previous chapter is used as a benchmark for deciding where the “best” data is. In this process, all data sources that are available at hand are screened in order to find those that best fulfill the requirements defined in the data model. Thus appropriate data sources are chosen to include the data that can “best” represent the data model.

In addition to the data model, the criteria of these appropriate data sources also include that they should be most timely, and most complete. These features ensure that the data entered in our database is not outdated or incomplete. For example, GenBank Entrez system is one of the most complete and frequently updated sequence systems

available. So it is chosen as our major source for nucleotide and protein sequences. The chosen data sources should also be the most reliable and accurate. Some data, such as disease information and tissue distribution information, can be obtained from the literature. To guarantee the reliability of such data, the literature articles we use for our data entry are all from peer-reviewed journals, and in many cases the results of the experiments has been verified.

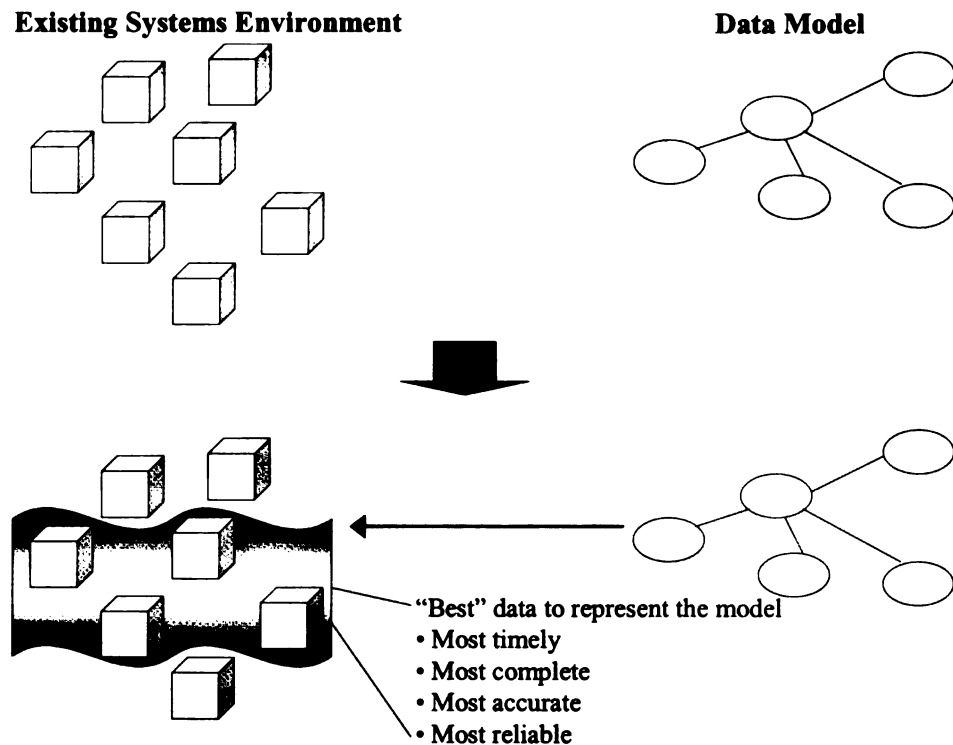


Fig. 3.8 Finding the “best” data sources

Table 3.1 lists some sample data sources for the TPDS. As mentioned in Chapter 1, many of the sources are on-line public databases containing unstructured data. Many of them contain redundant, “unclean”, and even chaotic data. Although much useful and valuable information can be extracted from these sources, the way they are stored and presented in these sources “hides” them. Structuring and mining of these “untreated” data is a recreation process that may not only clarify muddled and confused concepts, but also generate knowledge and associations that are not currently known.

To facilitate data access, some of the sources are provided as links in TPDS that can be accessed through normal queries (see examples in Chapter 4). Some newly developed databases are also included as our data sources. For example, dbSNP is a database about single nucleotide polymorphisms (SNPs) recently established by National Center for Biotechnology Institute (NCBI). Currently it contains about 25,000 SNPs.

Data Source Example	Links and Names
Entrez	http://www.ncbi.nlm.nih.gov/Entrez/
UniGene	http://www.ncbi.nlm.nih.gov/UniGene/index.html
dbEST	http://www.ncbi.nlm.nih.gov/dbEST/index.html
PDB	http://www.rcsb.org/pdb/
dbSNP	http://www.ncbi.nlm.nih.gov/SNP/index.html
OMIM	http://www.ncbi.nlm.nih.gov/omim/
Genes and Disease Map	http://www.ncbi.nlm.nih.gov/disease/Transporters.html
ABC transporter page	http://www.med.rug.nl/mdl/humanabc.htm
Transporter classification	http://www-biology.ucsd.edu/~msaier/transport/toc.html#1A
Gene nomenclature	http://www.gene.ucl.ac.uk/cgi-bin/nomenclature/searchgenes.pl
PubMed	http://www.ncbi.nlm.nih.gov/PubMed/
Book	Membrane Transporters as Drug Targets

Table 3.1 Sample data sources of TPDS

One of the most important tasks of building TPDS is to collect all the possible human transporter genes. Fig. 3.9 shows the methodology to do this, which includes several tracks. One track is to search the Entrez system for human transporters, porins, and ion channels. This can be done using a key word search in Entrez and then searching for related sequences. A systematic search track was used to search for each transporter family member (Saier's page, 2000) and the related sequences in Entrez. The gene sequences found through these tracks are then put into the BLAST program to run for similar genes. Because the GenBank systems are updated frequently and new data is available almost daily, our system also needs to be updated frequently to include the newest information as complete as possible. The system has been updated on a monthly basis.

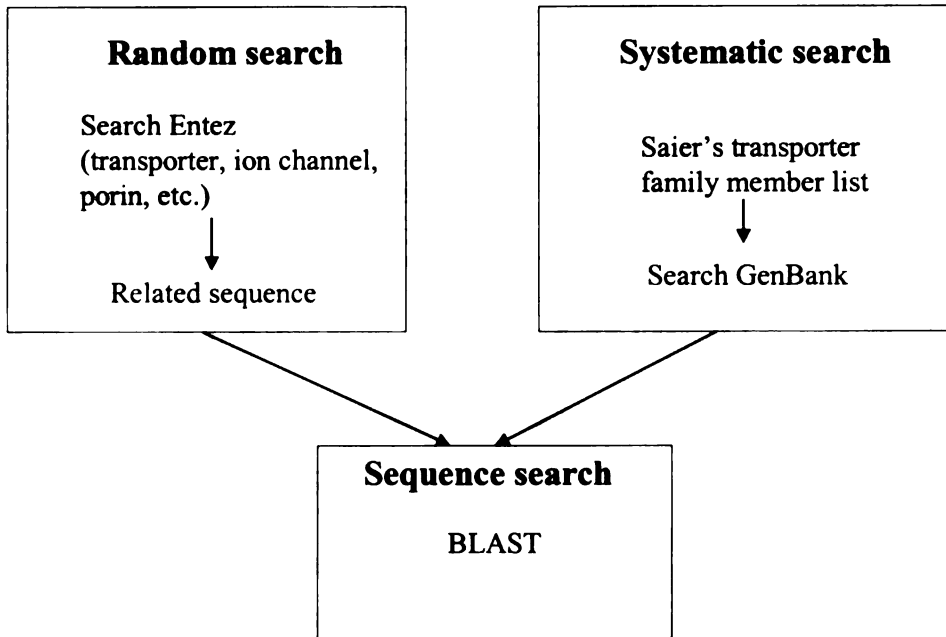


Fig. 3.9 The methodology of searching for transporter genes

3.6 Data Integration

“When I use a word,” Humpty Dumpty said, in a rather scornful tone, “it means just what I choose it to mean—neither more or less.”

“The question is,” said Alice, “whether you can make words mean so many different things.”

“The question is,” said Humpty Dumpty, “which is to be master—that’s all.”

—Through the Looking Glass, by Lewis Carroll, 1871

3.6.1 From Data to Information

The two words “data” and “information” are often used interchangeably. In fact, they are quite different, and this is one of the reasons we built the database system. The term “data” implies a collection of discrete elements, such as a file. As illustrated previously in Fig. 3.7, data is rarely clean and may have different formats. When data is structured, merged, aggregated, derived, sorted, and displayed, it becomes “information.” A database system provides an area to collect, integrate, and store data to perform the actions to enrich and enhance the value of the data. In another word, a database system offers a platform to transform data into information.

Fig. 3.10 shows this transformation process. This is also a data integration process that standardizes names and values, resolves inconsistencies in representation of data, and integrates common values together. The “equal” values of data from disparate sources that represent the same biomedical facts are also resolved in this process. This

transformation process is repeated over and over again, during the original development of the target database, when adding new sources to the existing database, and when distributing data from the system to users. The data integration process itself is actually a “product” that possesses the product characteristics more than the whole database. Once it has been constructed, this process can be reused in other projects.

As shown in Fig. 3.10, the data transformation process begins with the selection of data sources. This step has been discussed in earlier sections. The selected data is then manipulated and transformed. This step includes data consolidation and conversion. Data consolidation is a procedure that analyzes and merges data from disparate sources or systems into a single, integrated data structure. This is achieved through identifying data that is common across the various source files, and investigating the rules that manage the usage of the data. For example, the polymorphism data about transporter genes can be retrieved from several sources, such as dbSNP, OMIM, and GDB. It is necessary to integrate data from these different sources into a common data structure as shown in the table “Variance” in Fig. 3.5.

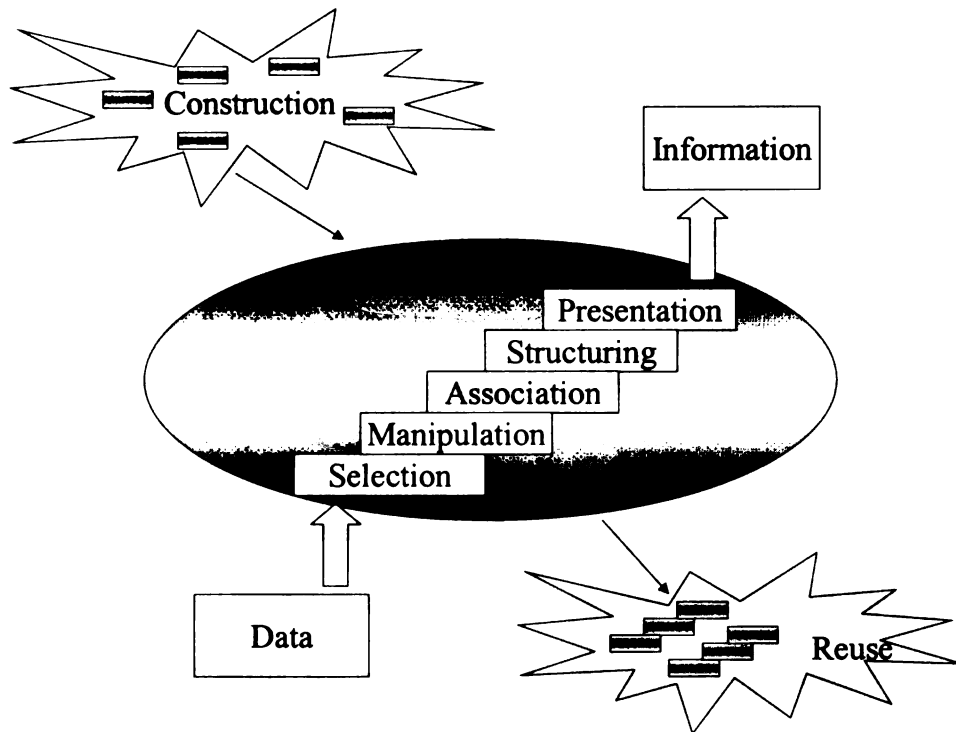


Fig. 3.10 The process of transforming data into information

During data consolidation, domain analysis is performed to analyze the content of each attribute in the source file, since each attribute has characteristic information that involves in a specific domain. This domain analysis process includes the understanding of whether the domain information actually matches the documented information in the data definition as we discussed previously. For example, it is necessary to notice that GenBank Accession numbers are used in the gene record of our database, rather than the GI numbers, although both of them are used in GenBank records.

Data consolidation also includes identifying data elements that have common biomedical meaning although the names are different (synonyms), or those that have the same name but represent different biomedical facts (homonyms). Failing to properly identify these in the source files may result in disparate data that fails to provide the true integration points. If so, a falsely cohesive view of the data that delivers the wrong information can be created. For example, failure to identify the synonyms of one transporter gene may cause these different names to be regarded as different genes, which may lead to unclean data and serious data redundancy. This is actually a complicated issue in biomedical science that involves terminology and knowledge representation studies. Specific issues involved in the transporter domain will be discussed in detail in the following sections.

The data conversion process concentrates on the data content by specifying how to adjust the source data to fit into the integrated target data structure. This includes mapping source file attributes to the physical data structure in our database, and mapping allowable values of source attributes to the target value. For example, the data structure in our database requires that family information should include Transport Commission (TC) numbers (these will be explained in detail in the next section). However, these TC numbers are not readily available for each transporter genes. In most cases the family that the gene belongs to has not been assigned. In these cases, the name, structure, and function of the gene are analyzed and compared with the definitions in the classification system (Saier's page) to map it to a TC number.

The data manipulation process also includes a data cleaning step. In this step, redundant data is removed, and outdated data is updated. Data cleaning is done together

with data consolidation and conversion. For example, sometimes the synonyms of a transporter gene are recorded as different records with redundant gene sequences and other attributes. These redundancies are removed with the identification of the synonyms. Sometimes the chromosome location of a gene from one source is not provided in detail (such as only with a number 11). When the detail (e.g., 11p15.5) can be found in checking with other sources, the more detailed data is used.

When the data is manipulated through consolidation, conversion, and cleaning, the resulting data can be associated with its meaning and presented with the structuring of the information. Such data presentation may include tables to graphs. This will be discussed in the next chapter along with system applications.

3.6.2 Transporter Classification

One of the most important parts involved in data consolidation is the classification of transporter genes. HMTD uses Saier's page (<http://www-biology.ucsd.edu/~msaier/transport/toc.html>) as the reference. This page includes a comprehensive classification system for membrane transport proteins known as the Transport Commission (TC) system. It is comparable to the Enzyme Commission (EC) system for the classification of enzymes, except that it incorporates both functional and phylogenetic information. In this TC system, transporters are classified on the basis of five criteria, and each of these criteria corresponds to one of the five numbers or letters within the TC number for a specific type of transporter. Any transport systems in the same subfamily of a transporter family that transport the same substrates are assigned

the same TC number, no matter whether they are orthologues (e.g., arose in distinct organisms by speciation) or paralogues (e.g., arose within a single organism by gene duplication).

In addition, sequenced homologues of unknown function are not normally given a TC number unless they represent a unique (sub)family. If multiple different subunits are present, they are numbered S1, S2, S3...Sn. Classification categories 8 and 9 are reserved for accessory transport proteins and incompletely characterized transporters, respectively.

According to Saier's classification, a TC number normally has five components as follows:

V.W.X.Y.Z.

- *V* (a number): the transporter class (i.e., channel, carrier (porter), primary active transporter or group translocator)
- *W* (a letter): the transporter subclass which in the case of primary active transporters refers to the energy source used to drive transport
- *X* (a number): the transporter family (sometimes actually a superfamily)
- *Y* (a number): the subfamily in which a transporter is found
- *Z*: the substrate or range of substrates transported

Fig. 3.11 shows an example of the TC number 2.A.22.2.2 and how to interpret each digit.

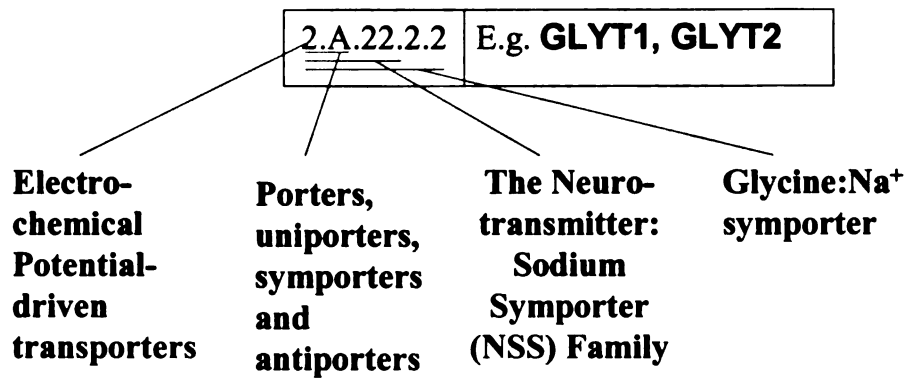


Fig. 3.11 An example of TC number

Table 3.2 shows a sample list contained in our database that represents the transporter families and their members using the TC system.

Family (TC#)	Transporter Isoform Symbol
1.A.1	KCNE1; KCNH1; EAG; KCNH2
1.A.4	ECAC1; LTRPC5; TRP4; TRPC1; TRPC2; TRPC3; TRPC4; TRPC5; TRPC6; TRPC7
1.A.5	PKD2L1; PKD2L2; PKDL
1.A.6	hBNaC1; hBNaC2; SCNN1A; SCNN1B; SCNN1G
1.A.7	P2X4
1.A.8	MIP
1.A.9	5-HT3; P2RX1; P2RX2; P2RX3
1.A.10	GRIA2
1.A.11	CLCN6; CLCN7; CLCNKA; CLCNKB
1.A.13	CaCC1; CaCC2; CaCC3
1.A.21	bcl-2; Bax; Bak1; Bak2; Bak3; bcl-w; hBAG-1
1.A.24	GJA4; GJA8; GJB2; GJB3; GJB5; CX36; CX40; CX43; CX59
1.A.27	FXYD1; FXYD2; FXYD3; FXYD5; PLM
1.A.45	AQP1; AQP2; AQP3; AQP4; AQP5; AQP6; AQP7; AQP8; AQP9
1.B.8	VDAC1; VDAC2; VDAC3
2.A.1.13	MCT1; MCT2; MCT3; MCT4; MCT5; MCT6; MCT7
2.A.1.14.6	NPT1
2.A.1.19	ORNT1; OCTN2; OCTN1; ORCTL4; ORCTL3; OCT1; OCT2; ORCTL2S
2.A.1.22	VMAT1; VMAT2; VACHT; SVAT
2.A.1.25	Ac-CoA; ACATN
2.A.1.4.5	G6PT
2.A.19	NCX1; NCX2; NCX3; NCKX1; NCKX2; NCKX3; NCKX4
2.A.20	GLVR1; GLVR2
2.A.21	SGLT1; SGLT2; SAAT1; SMIT; NIS; SMVT; HUT2
2.A.22	HTT; NAT1; DAT1; PROT; GLYT1; BGT-1; GAT1; GAT3; TAUT; CRTR
2.A.23	EAAT1; EAAT2; EAAT3; EAAT4; EAAT5; ATBo
2.A.27	HEAAC1; HEAACIII
2.A.28	NTCP; NTCP2
2.A.29	ANT1; Aralar-like; BCMP1; PMP34; HGT; ANT2; Aralar; PHC; CTP; UCP1; UCP2; UCP3; DIC; ANT3
2.A.3.3	CAT1; CAT2; CAT4; LPI-PC1; CSNU3
2.A.3.8	LAT1; LAT2; LAT3; XCT
2.A.30	NKCC1; NKCC2; KCC1; KCC3; KCC4; NCCT
2.A.31	NBC1; NBC3; NBC4; AE1; AE2; AE3
2.A.36	NHE1; NHE2; NHE3; NHE4; NHE5; NHE6; NHERF
2.A.41	CNT1; CNT2; SPNT1a
2.A.47	SDCT1; NADC3

2.A.48	FOLT; THTR1; RFC
2.A.55	NRAMP1; NRAMP2
2.A.57	ENT1(es); ENT2(ei)
2.A.58	NAPI-3B; NAPI3X; API4; NPT4; NPT3; NPT2
2.A.60	OAT1; OAT2; OAT3; OAT4; OATP1; OATP2; OATP3; OATP5; OATP8; PGT
2.A.7	UDP-GlcNAc; UGT1; UGT2; CMPST
3.A.1.15	ZNT1; ZNT2; ZNT3; ZNT4
3.A.1.201	MDR1; MDR2; ABCA2; ABCB3; ABCB4; ABCB6; ABCB8; ABCB9; ABCB10; ABCB11
3.A.1.202	MOAT-D; CFTR
3.A.1.203.1	ABCD3
3.A.1.204.1	ABC8
3.A.1.208	MRP1; MRP2; MRP3; MRP6; ABCC8; ABCC1; ABCC3; ABCC4; ABCC5; SUR2
3.A.1.209	TAP1; TAP2
3.A.1.210.4	ABCB7
3.A.1.211	ABC1; ABCA4
3.A.1.5	PEPT1; PEPT2
3.A.1.6	DTD
3.A.3	ATP1B2; ATP2A2; ATP2A3; ATP2B1; ATP2B2; ATP2B4; ATP7A; ATP7B; ATPGG; ATPase; PMCA1; FIC1; SERCA1
8.A.9.1.1	rBAT
9.A.12.1	CTR-1; CTR-2
9.B.17	FATP1; FATP2; FATP3; FATP4; FATP5; FATP6

Table 3.2 A sample list of transporter families and members according to the TC system

3.6.3 Transporter Nomenclature and Terminology

Terminology regarding genes often causes confusion. Studies of gene nomenclature are crucial in the identification of “synonyms” and in the process of data integration. The full name of a gene can be very long, such as “Homo sapiens cellular retinoic acid-binding protein 1”. Genes are usually represented by symbols, usually no

longer than six characters in length, such as “CRABP1” for this transporter gene. As defined by the Human Genome Organization (HUGO) Gene Nomenclature Committee, gene symbols are designated by upper case Latin letters or by a combination of upper case letters and Arabic numbers (White et al., 1997). Where gene products of a similar function are encoded by different genes, the corresponding loci are labeled by Arabic numerals placed immediately after the gene symbol, without any space between the letters and numbers used. For example, sodium-coupled nucleoside transporters CNT1 and CNT2.

Almost opposite to the scenario involving Alice and Humpty Dumpty (as cited at the beginning of this section), the problem in organizing transporter genes is that so many different words mean the same thing. This may even cause serious problems because if 4F2, 4F2HC, MDU1, NACAE, 4T2HC, and SLC3A2 (activator of dibasic and neutral amino acid transport) are treated as different genes, we will have six duplicated sets of records in our database. Such redundancy may even add the total number of the transporters, and lead to confusions in further analysis. If you want to look for information on neutral amino acid transporters, should you query for “SATT” or “ASCT1”? If you search these two gene symbols from the nomenclature database (<http://www.gene.ucl.ac.uk/cgi-bin/nomenclature/searchgenes.pl>), they are also termed “SLC1A4.”

HMTD is also a collection and organization of the terminology of transporter genes. In our database, we use the “Nomenclature of Mammalian Transporter Genes” as our reference and guidelines for gene symbols (<http://www.gene.ucl.ac.uk/cgi-bin/nomenclature/searchgenes.pl>). The “synonyms” of transporter genes are collected

from various sources, including GenBank, OMIM, and literatures. Table 3.3 shows a sample list of transporter genes with at least five symbols for one gene, as collected in our database. Each cell in the left column lists the “synonym” symbols of a gene. The right column of the table shows the corresponding GenBank accession number of this gene in the same row. To facilitate use, the query searching part of the database can be searched with all the related symbol(s) as transporter names. For example, to search for information about neutral amino acid transporters, you can type in “SATT,” or “ASCT1,” or “SLC1A4” as the key word.

Transporter Symbol	GenBank Accession Number
4F2, 4F2HC, MDU1, NACAE, 4T2HC, SLC3A2	<u>AH001404</u>
ABC1, ABC-1, ABCA1, TGD, CERP, HDLDT1	<u>AJ012376</u>
ABCA4, ABCR, STGD1, FFM, STGD, ARMD2	<u>NM 000350</u>
ABCB2, TAP1, D6S114E, PSF1, RING4	<u>NM 000593</u>
ABCC2, CMOAT, MRP2, cMRP, DJS	<u>NM 000392</u>
ABCD1, ALD, X-ALD, AMN, ALDP	<u>NM 000033</u>
ABCD4, PXMP1L, P70R, PMP69, EST352188	<u>NM 005050</u>
ABCP, BCRP, ABCG2, MXR, ABC15, EST157481	<u>NM 004827</u>
ANT1, T1, PEO2, PEO3, SLC25A4	<u>NM 001151</u>
ANT2, SLC25A5, 2F1, T2, SLC25A5	<u>L78810</u>
ATBo, SLC1A5, RDRC, hATBo, ASCT2, AAAT, M7V1, RDRC	<u>U53347</u>
BWR1B, SLC22A1LS, BWSCR1B, ORCTL2S, P27bwr1b	<u>NM 007105</u>
CAT1, HCAT1, ERR, REC1L, SLC7A1, ATRC1, CAT-1	<u>AF078107</u>
CAT2, SLC7A2, CAT-2a, CAT-2b, ATRC2, HCAT2	<u>D29990</u>
CNT1, SLC28A1, cit, N2, HCNT1	<u>NM 004213</u>

CNT2, SLC28A2, cif, N1, SPNT1, HCNT2	<u>NM 004212</u>
E16, LAT1, LAT-1, D16S469E, MPE16, SLC7A5	<u>AF077866</u>
E3KARP, NHERF-2, TKA-1, SIP-1, SLC9A3R2	<u>NM 004785</u>
FOLT, SLC19A1, RFC1, REFC, CHMD	<u>NM 003056</u>
GLVR2, GLCR1, MLVAR, GLVR-2, SLC20A2, PiT-2	<u>NM 006749</u>
HGT, SLC25A16, GDC, D10S105E, GDA, ML7	<u>M31659</u>
IMPT1, ORCTL2, BWR1A, BWSCR1A, ITM, TSSC5, SLC22A1L	<u>AF028738</u>
MDR2, MDR3, ABCB4, PGY3, PFIC-3	<u>M23234</u>
MOAT-C, ABCC5, MRP5, SMRP, ABC33, EST277145	<u>AF104942</u>
MRP3, ABCC3, MLP2, ABC31, MOAT-D, CMOAT2, EST90757	<u>AF085692</u>
NBC1, NBC2, pNBC, HNBC1, hhNMC, SLC4A4	<u>NM 003759</u>
rBAT, SLC3A1, D2, D2H, CSNU1, SLC3A1	<u>U60819</u>
SUR1, ABCC8, HI, PHHI, SUR, MRP8, HRINS	<u>U63421</u>
TAP2, ABCB3, D6S217E, PSF2, RING11	<u>X87344/2</u>

Table 3.3 Sample transporters with at least five symbols for one gene

3.7 Summary

This chapter has integrated many different technologies in order to develop the decision support system prototype, TPDS. The whole DSS was developed based on a three-tiered client-server architecture to allow Web accessible applications. The physical database design supports the implementation of the logical data model

developed in the last chapter. Because the data comes from different sources, data integration was an important part of the system development. Here data integration includes data selection, consolidation, cleaning, and transformation. This integration process also considered the special features of transporters, including classification, nomenclature, and terminology. Many elements developed here are reusable, including the Java coding of data definition, and the data integration process.

Chapter 4. TPDS Application

In order for a technology to be successfully adopted as part of a healthcare process, it must do one of two things: enable a user to do an existing task more easily or efficiently; or enable a user to do a previously difficult or undoable – but desirable – task.

—Dr. Jerry Cox, Biomedical Computing Sage

4.1 Data Access and Query Execution

4.1.1 Data Access Procedure

The ultimate goal of all the processes of data modeling, transformation, and management is to facilitate the access and usage of the data by end-users. To access the data in the database, several steps are needed. These steps and data access components are illustrated in Fig. 4.1. When a user makes an enquiry, the query from the end-user tool is translated to the database API and SQL queries, which are executed in the database HMTD. Then, the information found through the execution of SQL is translated through database API. The returned result is then translated into a presentation format appropriate for the end-user tool and passed back to the user.

During the process, a control-and-administration function identifies data locations and formats. On both the inquiry and response paths, data access invokes data transfer functions. On the inquiry path, the data being transferred consists of short

messages (such as the keyword for searching). However on the response path, data sizes can be significantly larger. Because the end-user is waiting for the query result, data access expects that data transfer operates at synchronous or near real-time asynchronous speeds. The whole data access process is supported by the three-tiered client-server architecture as described in the previous chapter.

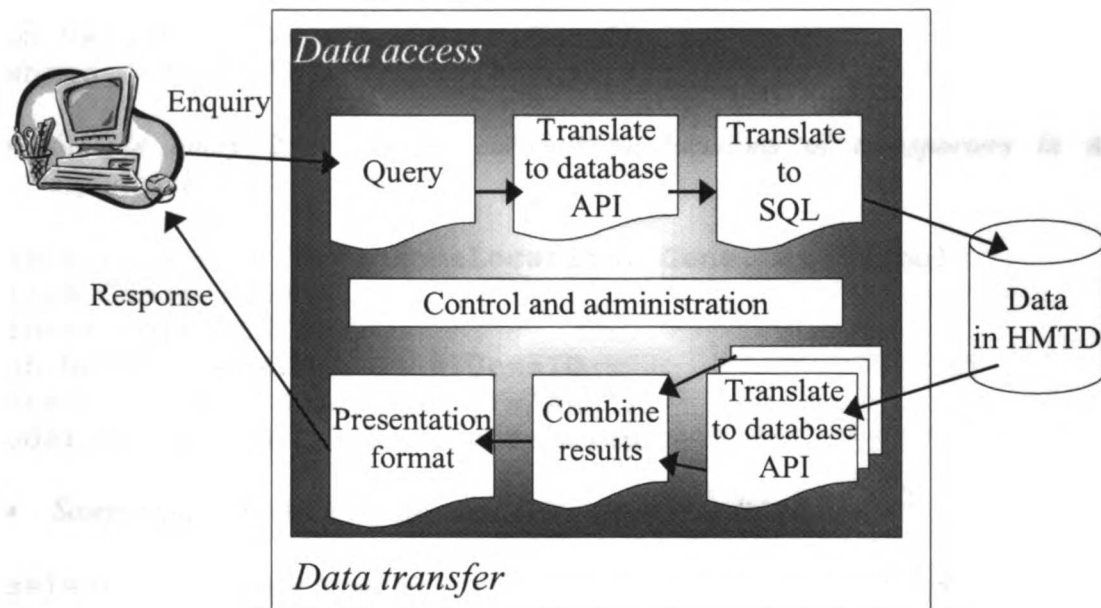


Fig. 4.1 The structure of data access components in TPDS

4.1.2 Queries Through SQL

Figure 4.2 shows some sample lists of SQL code that execute queries in our database.

- *Sample query 1: What is the polymorphism of transporter TAP1?*

```
select Variance.Polymorphism
from Variance
inner join GENE
on Variation.GeneID = Gene.GeneID
where (((Gene.GeneSymbol)='TAP1'));
```

- *Sample query 2: List the chromosome locations of transporters in ABC superfamily.*

```
select Genome.ChromosomeLocation, Gene.GeneSymbol
from Genome, Gene
inner join GENE
on Genome.GeneID = Gene.GeneID
where (((Gene.SuperFamily)='ABC'))
oder by Genome.ChromosomeLocation;
```

- *Sample query 3: What transporters are expressed in liver?*

```
select Gene.GeneSymbol
from Gene
inner join TissueDistribution
on TissueDistribution.GeneID = Gene.GeneID
where (((TissueDistribution.Tissue) Like "*liver*"))
order by Gene.GeneSymbol;
```

- *Sample query 4: What are the transporter related diseases?*

```
select Disease.Name, Gene.GeneSymbol
from Disease, Gene
where (((Disease.GeneID=Gene.GeneID))
order by Disease.Name;
```

Fig. 4.2 Sample SQL code for various queries in TPDS

4.2 User Interface of TPDS

4.2.1 The Iterative Design Approach

A user interface is not only an access to the data, but also a part of the decision support. To most people, especially the users, the user interface is just the software, or the window through which they know about the software. The users are not aware of the software design, data integration, or system architecture. All they know about the software is through the user interface. If data is not well presented and accessed, it is most likely to be ignored, although it is there. In most cases, people see things that are ready for them to see.

The primary goal of a user interface is to help the designers easily create applications that increase user effectiveness and satisfaction. A good user interface is not necessarily one with an interesting color scheme, animation applets, or complicated graphics. In the design of the user interface, several rules are considered to facilitate the usage of the system. These rules include simplicity, naturalness, consistency, intuitiveness, and ease of use.

Of all the principles, the most imperative concern of an effective design is rooted in adopting the user's point of view. This is often difficult to do, because this requires an understanding of the user's working domain and thinking habits. A well-designed application in a decision support system supports users' decision making process, makes their work easier, and offers them new capabilities. In my design, three strategies were used. The first one was to look at the user interface of other

bioinformatics systems, and analyze the effective and inefficient parts of their design. The second was to talk to the users and listen to their feedback. The third way was to use it myself to get a sense of the application.

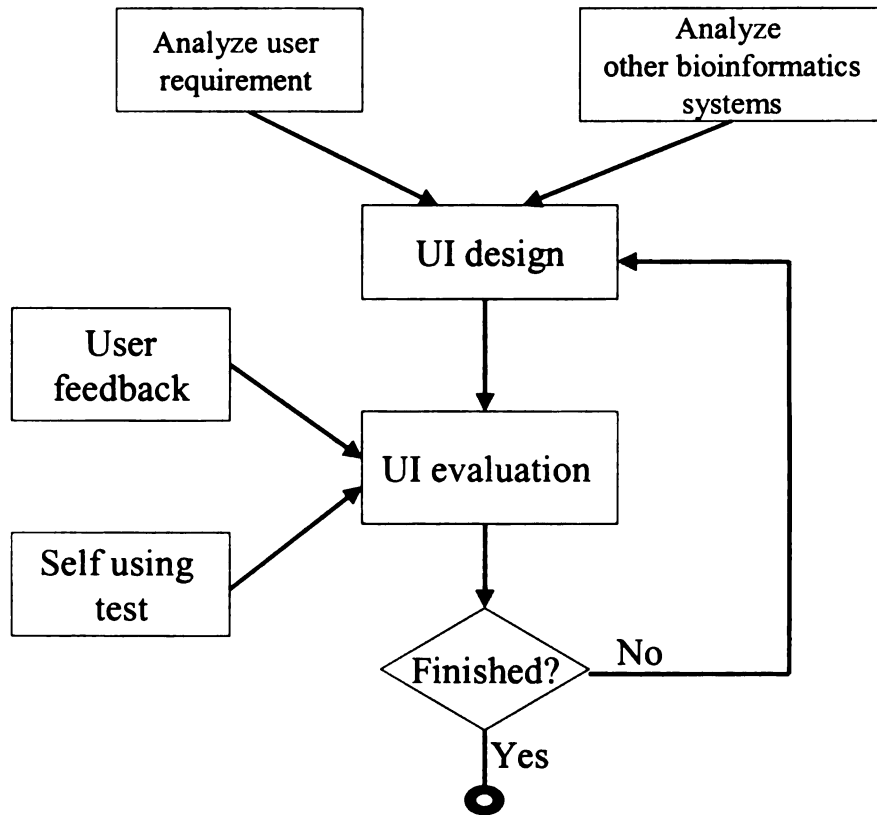


Fig. 4.3 The iterative approach of the user interface (UI) design of the TPDS prototype

Fig. 4.3 shows the iterative approach of designing the user interface for the TPDS prototype. User requirements are analyzed in the system design and data

modeling phase, using use case analysis, the class diagram, and the sequence diagram. Analyzing the user interface of other biomedical systems has the advantage in helping the users transfer familiar skills to the new situations in our system. This analysis helps keep the consistency of our new system among the existing biomedical systems. Although our user interface will be different from others, the design based on this analysis guarantees that it will not surprise the users. Instead, it will make our applications more intuitive. Users can apply the knowledge learned from other systems and reduce the amount of learning in using our system. This approach may increase users' sense of mastery, as they can transfer the experience with other systems.

The analysis of the user interface in other biomedical systems suggested that a query form for searching to be filled by the users is commonly used and accepted by biologists. Such forms usually have user-defined parameters to support dynamic and complex queries. However, some systems had forms that were too complicated, not intuitive and were very hard to figure out how to use. The presentation of the query result is also an important issue in the user interface design. Some of the existing systems presented their query results in a non-structured text file, with the data that was hard to recognize and categorize. In these cases, users might just give up after first try, leaving the system unused.

4.2.2 Development of The User Interface

With these in mind, I designed the initial user interface of our system prototype as shown in Fig. 4.4. The major applications are designed in a query form. The

application is designed to present the most necessary and common functions in a logical order. Different functions can be accessed by changing the parameters through the pull down menus in the form. A list of all the transporter genes in the database is also provided as a link. It was not easy to make decisions about the placement of functions, because all functions were important from the implementation standpoint. After communicating with users, a relatively small number of functions were selected and presented, such as nucleotide sequence and tissue distribution.

After collecting feedback from users and using it myself for a while, I analyzed the good, bad, and missing features of the initial user interface. The good features included that the interface was relatively easy to use, and the presentation of the query results were clear in a well-structured table form. The bad and missing features included that the query form only provided one defined way for data access, but lacked direct manipulation of applications. Every time users wanted to get some information, they had to select the menu and use the keyboard to fill out the form. I felt that the initial interface needed to be improved to increase the sense of control by users.

A solution for improvement with this original design was to provide multiple ways for users to access application functions and accomplish their tasks. Multiple ways of access offer the flexibility for users to select the most appropriate methods based on their experience level, personal preference, or just habit. For example, direct manipulation of applications reduces the amount of information the users need to memorize through simulating the real world, where the users can employ tools to perform tasks on physical objects. With direct manipulation in a user interface, users

can just click on the objects they are interested in and then see the results directly, rather than using a keyboard-driven command or filling out the whole form. In this way, a user action is related to an immediate visible response within a simple mouse click.

Now the problem becomes how to provide multiple ways for user access, with each way still functional, independent, and obvious to users. In the second design, I introduced a frame into the user interface. The frame separated different actions while supporting multiple ways (as shown in Fig. 4.5). In Fig. 4.5, the top part of the web page is a greeting to the database, with a DNA structure picture as background. Different colors are used in the background to differentiate the frames. Color contrast is used to distinguish top and lower frames. In the top frame, the words are written in light color on a relatively dark background, to attract web navigator's attention. In the lower frames, dark objects (such as words and buttons) are on a relatively light background. The lighter background in the lower part attracts users' attention to the system application. Two ways of data access are provided in the left and right frames separately.

The left frame allows direct manipulation through electronic access to predefined reports. Some major topics that users are most interested in is provided here represented by an object name. These names are listed with bullet points to make their appearance distinct so they are easily recognized and quickly understood. The order of the names is arranged according to their frequency of usage, and grouped by their features. Through clicking on the object name (such as Chromosome in Fig.4.5), users can see the resultant data in a predefined format listed in a table in the right frame, without having to do any typing. To differentiate the results of this database and links

to other web pages, clicking on the links to other pages will pop up another window containing that page, rather than showing them in the right frame.

One of the features of direct manipulation is that sometimes the output of one part of an application is also available as input. In this way, the user control can be extended, and the application can be arranged to make the tasks flow naturally. When using this feature, the resulting table list also integrates links to further information such as links to nucleotide sequences and SNP information. Through clicking on these links (some provided with the GenBank accession number), the users can then get the next information they want. Some explanations are provided on the top of these result tables, helping users anticipate the natural progression of each task and finish the tasks quickly.

The right frame occupies two-thirds of the whole screen. As default, it retains the parameter-based ad hoc query function through filling out a form. A help file containing examples and tips is also provided to facilitate the usage of the form (see Fig. 4.6). This frame also contains the update time of the system. To keep consistency among the applications, a link back to this default query form is provided on the top of the left frame. In case users get lost in navigating the left frame objects, clicking on this link will bring them back to the default page.

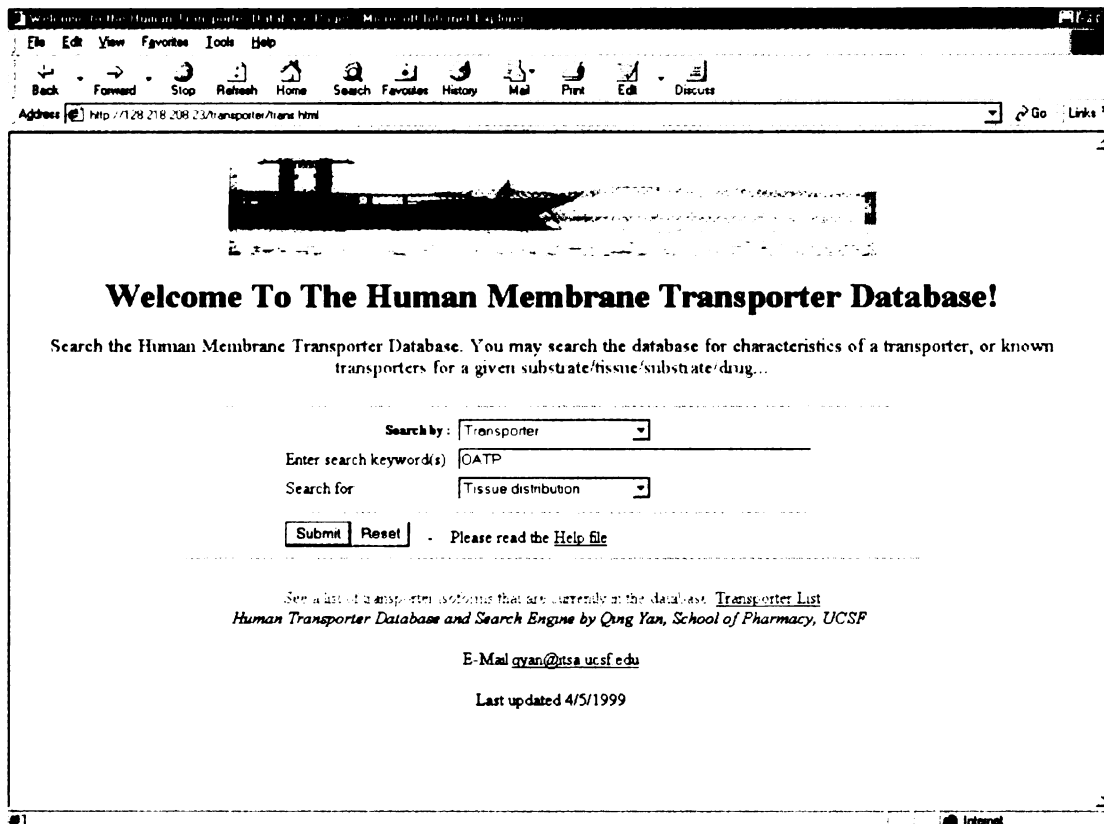


Fig. 4.4 The original page of the system prototype

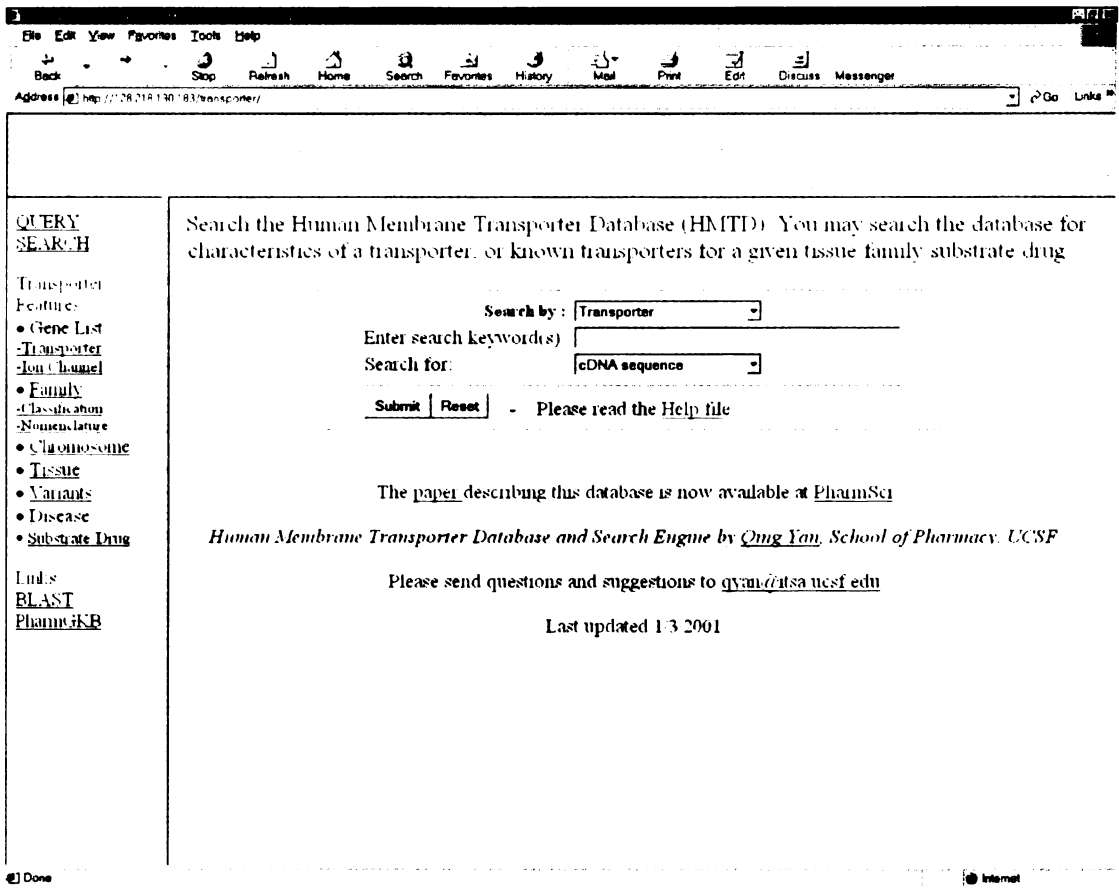


Fig. 4.5 The revised page with improved functions

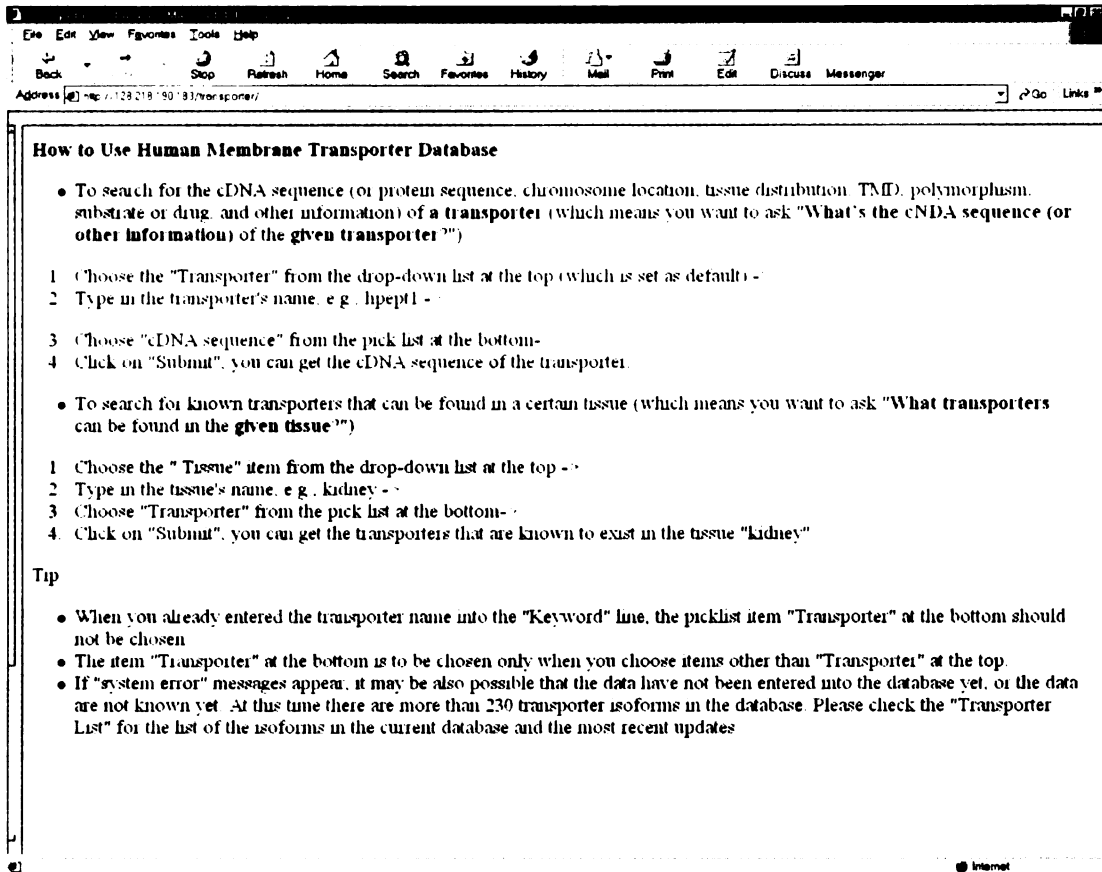


Fig. 4.6 The help file

4.3 Data Presentation and Data Application Samples

As mentioned before, some bioinformatics systems present their report in an unstructured format that is very difficult to understand or analyze. Data presentation is one of the most important components in decision support information (as described in Fig. 3.10 previously). Usually a report table containing relevant and structured information is appropriate for the presentation of query results.

Now let's see some real samples of data applications in the TPDS prototype (the sample SQL statements used to execute these queries have been described earlier in Fig. 4.2):

Sample 1: What is the polymorphism of transporter TAP1?

For this query, a more direct approach is through filling out the form on the right frame of the Web page. This can be done through selecting the type of data one would like to query (such as "polymorphism" of a "transporter"), typing in the keyword (such as "TAP1"), and clicking the button "submit". Thus the Web browser sends the request to the Web server (as the process described in Fig. 3.1). The users then see the query results (as shown in Fig. 4.7).

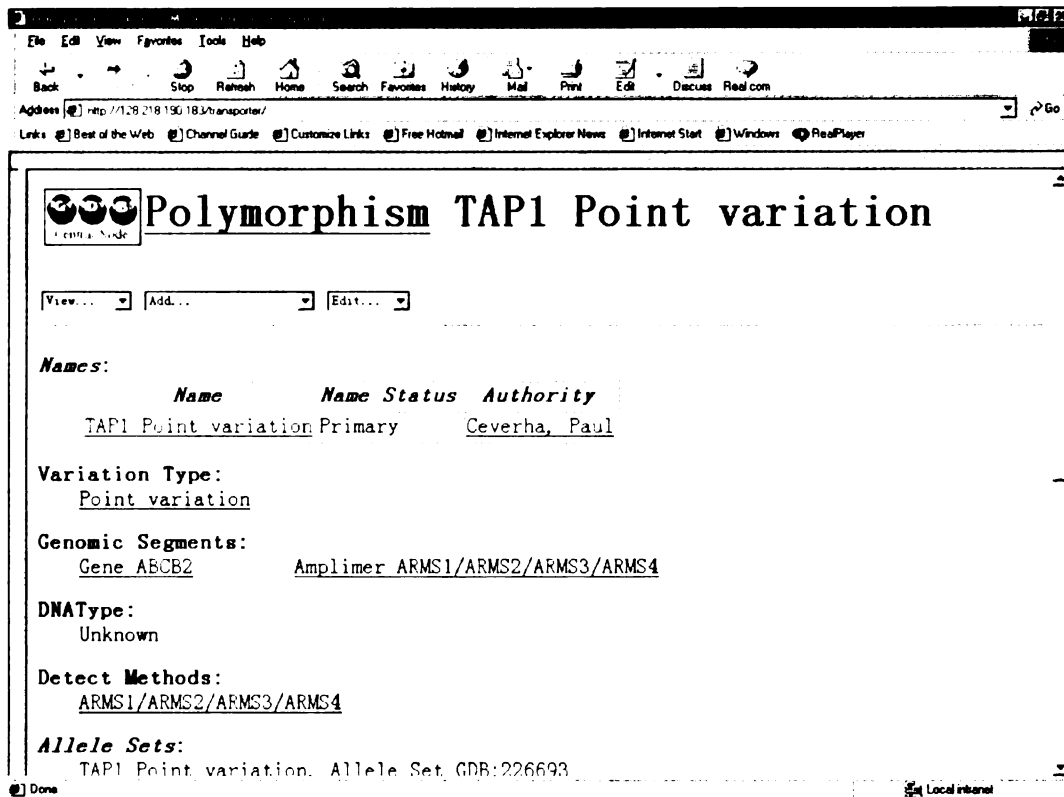


Fig. 4.7 The sample result of a query “What is the polymorphism of TAP1?”

Sample 2: List chromosome locations of transporters in the ABC superfamily.

Chromosome location is a frequently queried element of the database. Such information may provide users a sense of where the transporters (and maybe also the whole family) are in the whole genome, how they are related spatially in the genome, and what the nearby transporters are. Table 4.1 shows a sample list of chromosome locations of transporters in the ABC superfamily.

Chromosome	Transporter Isoform Symbol
1p22	ABCA4
1q21-q23	ATP1A2
1q25-q32	PMCA4; ATP2B4
2q24	BSEP
3	PEPT2
3p26-p25	ATP2B2; PMCA2
3q22-q23	ATP1B3
3q27	MRP5
4q22	ABCP
5q31-q34	DTD
6p21.3	TAP1; TAP2; NPT3; NPT4
7	ZNT3
7q21.1	MDR1; MDR2; ABCB4
7q31	CFTR
9q22-q31	ABC1
10q23-q24	MRP2
12q11-q12	ABCD2; ALDR
12q12	ALD1; hBNaC2
12q21-q23	ATP2B1; PMCA1
13q12.1-q12.3	ATP1A1
13q14.3	ATP7B
13q32	MRP4
13q33-34	PEPT1
14q24.3	ABCD4; PMP69
16p12	SERCA1
16p13.1	MRP6
16p13.12-13	MRP1
16p13.3	ABCA3
17p	ATP1B2
17q21-q23	MRP3
18q21	FIC1
19q12-q13.2	ATP1A3
19q13.1	CSNU3; SLC7A10; ATPGG
20q11.2	ZNT4
Xq12-q13	ATP7A
Xq13.1-q13.3	ABCB7
Xq28	CRTR; ABCD1; PMCA3g

Table 4.1 Sample list of chromosome locations of transporters in the ABC superfamily

Sample 3: What transporters are expressed in liver? What about kidney, intestines, and brain?

Tissue distribution is one of the most crucial information about transporters. Gaining information from such queries allows us to make judgments on the basis of which known transporters are prevalent in available tissues. Such information may have important relevance in pharmacogenomics studies. For example, the resultant information can be used in designing new drugs that need to be targeted to certain tissues. To determine which drug transporters are candidates for genotyping, this information may also be helpful. In addition, such information may assist in guiding drug therapy in individual patients on the basis of the drug transporter genotype and phenotype. Table 4.2 shows a partial sample list of transporters expressed in liver, kidney, intestines, and brain.

Liver	Kidney	Intestines	Brain
AE1	AE2	4F2HC	ALDR
ANT2	ASNA1	ACATN	ASCT1
BGT-1	CNT1	ATP2A3	ATP1A1
BSEP	ENT1 (es)	CAT1	CAT4
CAT1	FATP4	CNT1	CNT1
CAT2	GLCR2	CNT2	CNT2
CNT1	GLUT2	CTR-1	DAT1
CNT2	GLUT5	CTR-2	DIC
CTR-1	GLUT6	EAAT3	EAAT1
CTR-2	GLVR1	ENT1 (es)	EAAT2
DIC	HCAT2	FIC1	EAAT3
EAAT2	KCC1	FOLT	ENT1 (es)
EAAT5	KCC3	GLUT2	ENT2 (ei)
ENT1 (es)	KCC4	GLUT5	GAT-1
FATP4	LAT-2	GLUT6	GAT-3
G6PT	LAT-3	HEAACIII	GLCR2
GLCR2	MCT4	KCC1	GLUT1
GLUT2	MCT5	LAT-2	GLUT3
GLUT6	MDR1	MCT7	GLUT6
GLVR1	MRP1	MDR1	GLYT1
LAT-2	MRP3	MRP1	GLYT2
LST-1	NAPI-3B	MRP3	HTT
MCT7	NAPI3X	NAPI-3B	KCC1
MDR1	NCX1	NBC	KCC3
MDR2	NHE1	NCCT	KCC4
MNK	NHE2	NHE1	LAT-1
MRP1	NHE3	NHE2	LAT-2
MRP2	NHE6	NHE3	MCT2
MRP3	NKCC1	NHE6	MCT6
NHE1	NKCC2	NRAMP2	MCT7
NHE6	NPT1	NTCP2	MRP1
NPT1	NPT2	OCT1	NAT1
NRAMP2	NTCP1	OCTN2	NBAT
NTCP1	NTCP2	ORCTL2	NCKX1
NTTA	OAT1	PEPT1	NCX1
OAT2	OAT2	PGT	NHE1
OATP1	OAT3	PMCA1	NHE5
OCT1	OATP1	rBAT	NHE6

OCTN1	OCT1	RFC	NTTA
OCTN2	OCT2	SATT	OAT1
ORNT1	OCTN1	SBC2	OATP1
PEPT1	OCTN2	SDCT1	PROT
PGT	ORCTL2	SGLT1	SERT
PMCA4	ORCTL3	SGLT2	UCP4
PMP34	ORCTL4	SMVT	VACHT
PMP70	PAHT	SPNT1a	WHITE1
SFT	PEPT1	SVCT2	ZNT-1
TAUT	SMIT	TSC	ZNT-3
UGT1	TAUT	ZNT-1	ZNT-4

Table 4.2 Sample list of transporters in tissues liver, kidney, intestines, and brain

Sample 4: What are the transporter related diseases?

Disease is a crucial part of phenotype. A list of answer to such query is a direct application of genotype-phenotype correlation. Table 4.3 shows a sample list of transporter-related diseases.

Disease	TransName
acrodermatitis enteropathica	ZNT-4
Adrenoleukodystrophy	ALDR
anemia, genetic hemochromatosis	HET
ankylosis (calcification); arthritis accompanied by mineral deposition, formation of bony outgrowths, and joint destruction	ANK
anxiety-related traits, depression	HTT; SMVT
attention-deficit disorder	NRAMP2
Bartter's syndrome	NKCC2
Brody myopathy	SERCA1
chondrodysplasias/ diastrophic dysplasia	DTD
congenital chloridorrhea	AE2; AE3

Disease	TransName
congenital hypothyroidism	NBC3; NIS
Cystic fibrosis	CFTR
cystinuria diseas	NBAT; rBAT
deficiency causes glycogen storage disease 1	G6PT
DiGeorge syndrome, velocardiofacial syndrome	NORTR
Down's syndrome	NBC
Dubin-Johnson syndrome (DJS)	MRP2
elliptocytosis, ovalocytosis, hemolytic anemia, spherocytosis, renal tubular acidosis	AE1
Familia intrahepatic cholestasis	MDR2, MDR3
Folate malabsorption/megaloblastic anemia	FOLT
Gitelman syndrome	NCCT; NKCC1
Graves' disease	PMP34
HHH	ORNT1
hypocitraturia	SBC2
Isolated fructose malabsorption	GLUT5
juvenile onset psriasis	TAP1
low CNS glucose causing seizures, Fanconi-Bickel syndrome, Glycogen storage disease type Id, Non-insulin-dependent diabetes mellitus, defect in glucose transport across the blood-brain barrier	GLUT1; GLUT2; GLUT3; GLUT4
Lysinuria	4F2HC
Lysinuric protein intolerance	CAT2; LAT-2; LAT-3
Menkes syndrom	ATP7A(MNK)
Menkes/Wilsons disease	CTR-1; CTR-2
Microvillus inclusion disease	NHE2; NHE3
Neurodegeneration	ATBo; EAAT3; SATT; EAAT4; EAAT5
Neurodegeneration, Amyotrophic lateral sclerosis	EAAT1
Neurodegeneration, Dicarboxylic aminoaciduria	EAAT2
neuropsychiatric disorders	SVMT
orthostatic intolerance	NAT1
pendred syndrome, (PDS)	PDS
persistent hyperinsulinemic hypoglycemia of infancy	SUR1
primary bile acid malabsorption (PBAM)	NTCP2

Disease	TransName
Progressive familial intrahepatic cholestasis (PFIC)	BSEP; FIC1
renal glucosuria / glucose-galactose malabsorption	SGLT1; SGLT2
renal tubular acidosis	NBC1
Stargardt disease (macular degeneration in childhood)	ABCR
systemic carnitine deficiency (progressive cardiomyopathy, skeletal myopathy, hypoglycaemia, hyperammonaemia, sudden infant death syndrome)	OCTN2
Tangiers disease	ABC1
various mental disorders	SERT
Wilson disease	ATP7B (WND)
X-linked adrenoleukodystrophy (X-ALD)	ALD1
X-linked sideroblastic anemia	ABC7

Table 4.3 Sample list of transporter-related diseases

4.4 Summary

This chapter emphasized the applications of the system. Data access of the system is mainly through SQL and system API. User interface design sometimes can be a very complicated issue especially in an area with strong professional features such as biomedicine. This kind of user interface requires the understanding of the workflow and thinking habits of the biomedical users. The icons, pictures, word panels, and backgrounds are all designed to match the tone of the professional users. Direct manipulation of the user interface may strongly support the direct interaction between users and the system. Together with query tools and forms, such multiple ways of

Chapter 5. Conclusion

The Tao produced One; One produced Two; Two produced Three; Three produced All things. All things leave behind them the Obscurity (out of which they have come), and go forward to embrace the Brightness (into which they have emerged), while they are harmonized by the Breath of Vacancy.

— *Tao Te Ching, Lao Tzu*

5.1 The Iterative Life Cycle of Prototyping TPDS

Using the analysis, design, coding, and testing phases, a prototype has been built for the TPDS. Here the prototype emphasized the most important and frequently examined issues in transporter pharmacogenomics studies. The most important functionalities that the end users are interested in were developed first to assure that the system satisfies the end users. The TPDS prototype demonstrated that it could interface with all the infrastructure elements it would need to deal with. This assured that the overall infrastructure design was appropriate.

Similar to the user interface construction process, the development of the TPDS prototype was an iterative life cycle. In each cycle, problems were identified and the overall design was revised again and again until a satisfactory product to meet the requirements was achieved. Fig. 5.1 illustrates the iterative approach of prototyping TPDS. Each of the phases in the cycle was described in detail in the previous chapters. During the initial requirement effort, an overview of the entire system was developed.

The correlations in the biomedical knowledge domain were analyzed to define the system boundary. Biomedical models were constructed to clarify the concepts and facilitate concept abstraction for further data modeling. During the analysis and design phase an object data model was built and a logical system that satisfied the requirements was designed. Unified modeling language (UML) was used for the object-oriented data modeling and for defining the design patterns. The implementation details were not a concern at this stage.

Then the concrete, physical system architecture was constructed and code was written. After the initial prototype was tested, the development entered into the second cycle. In each cycle modifications were made to make improvements in the design and implementation. This iterative life cycle was integrated in the development of TPDS and several cycles have been gone through since the first prototype was built at the beginning of 1999.

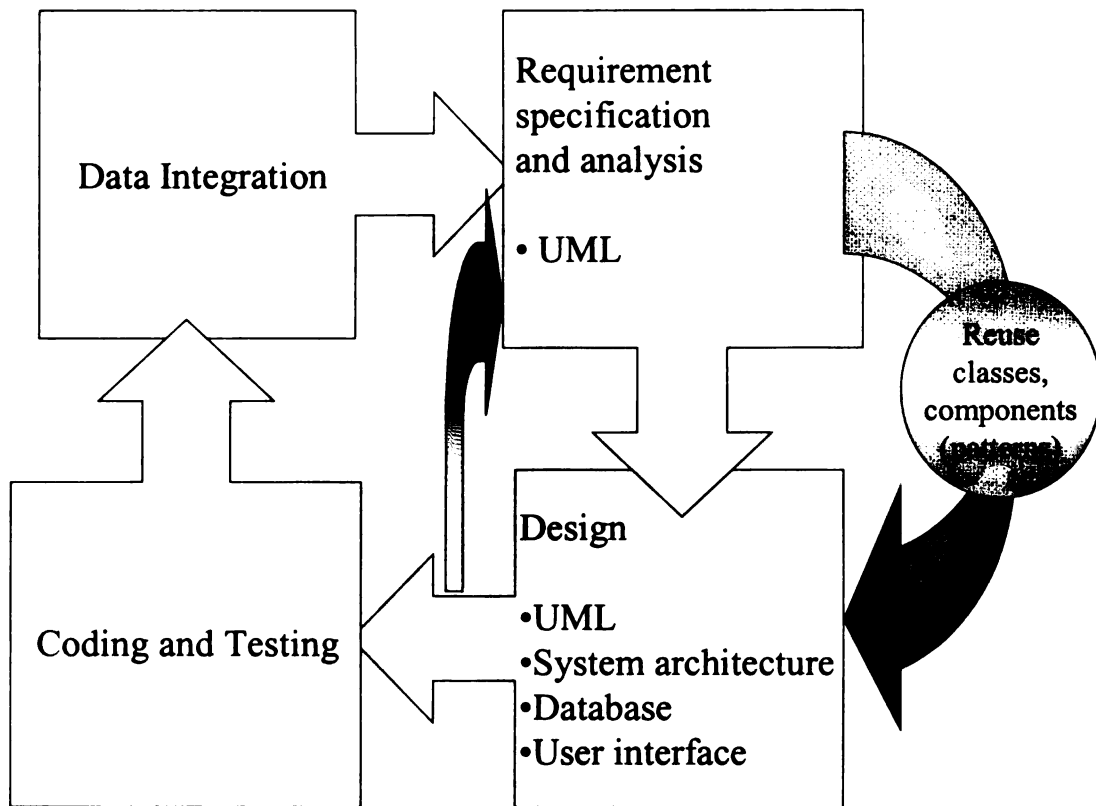


Fig. 5.1 The iterative life cycle of prototyping TPDS

Fig. 5.2 shows the time and efforts in each phase spent on the development of TPDS. Most of the efforts were spent in the design phase. This design phase had a broader meaning than just object-oriented design and data modeling for logical relationships between the objects. It also included the design of physical system architecture, which determined the infrastructure objects and different operations. During this phase, the classes and components that can be reused were also identified. In addition, the design of the database, and the design of the user interface were

included in this design phase. These design activities laid the ground for further implementation.

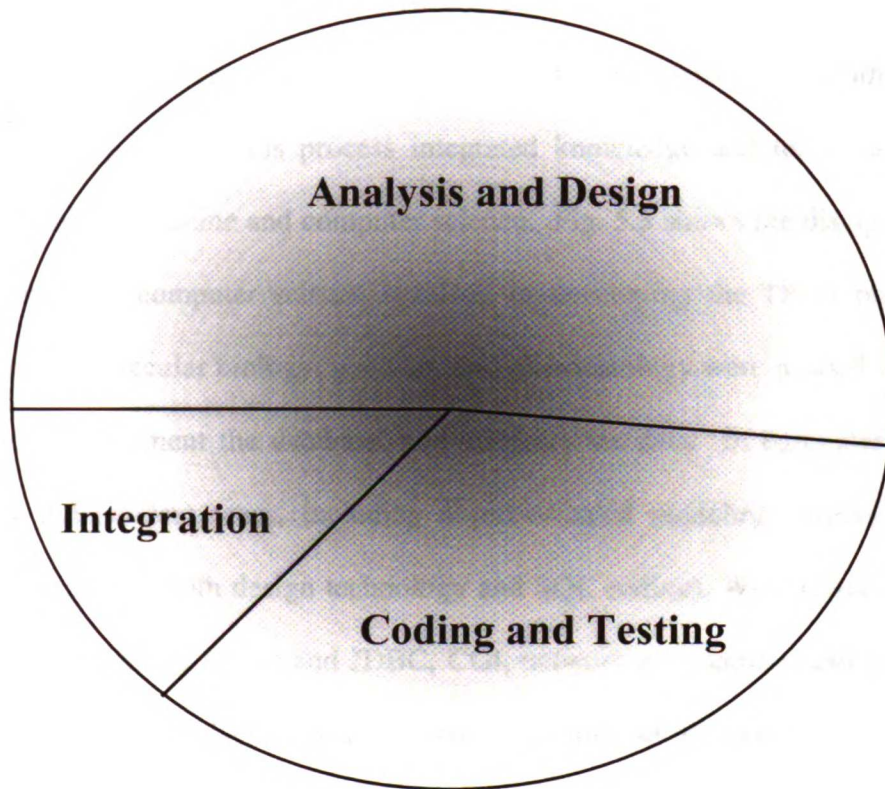


Fig. 5.2 The time and efforts in each phase spent on the development of the TPDS prototype

5.2 An Interdisciplinary Approach Where Biomedicine Meets Computer Science

As mentioned in the first chapter, pharmacogenomics is an interdisciplinary area with very complex characteristics. To solve the complexity, the development of a decision support system for transporter pharmacogenomics studies was a multidisciplinary task. This process integrated knowledge and technology in many areas in both biomedicine and computer science. Fig. 5.3 shows the disciplines in both biomedicine and computer science involved in developing the TPDS prototype. In biomedicine, molecular biology, genetics, and pharmacology were needed to design the data model, implement the database, and integrate the data. In computer science the relatively independent areas, including object-oriented modeling (especially, UML), database (including both design technology and SQL coding), Web technology, object-oriented programming (Java) and JDBC, CGI, network architecture, and user interface development were involved. Only with the integration of all these different areas was the decision support system made possible.

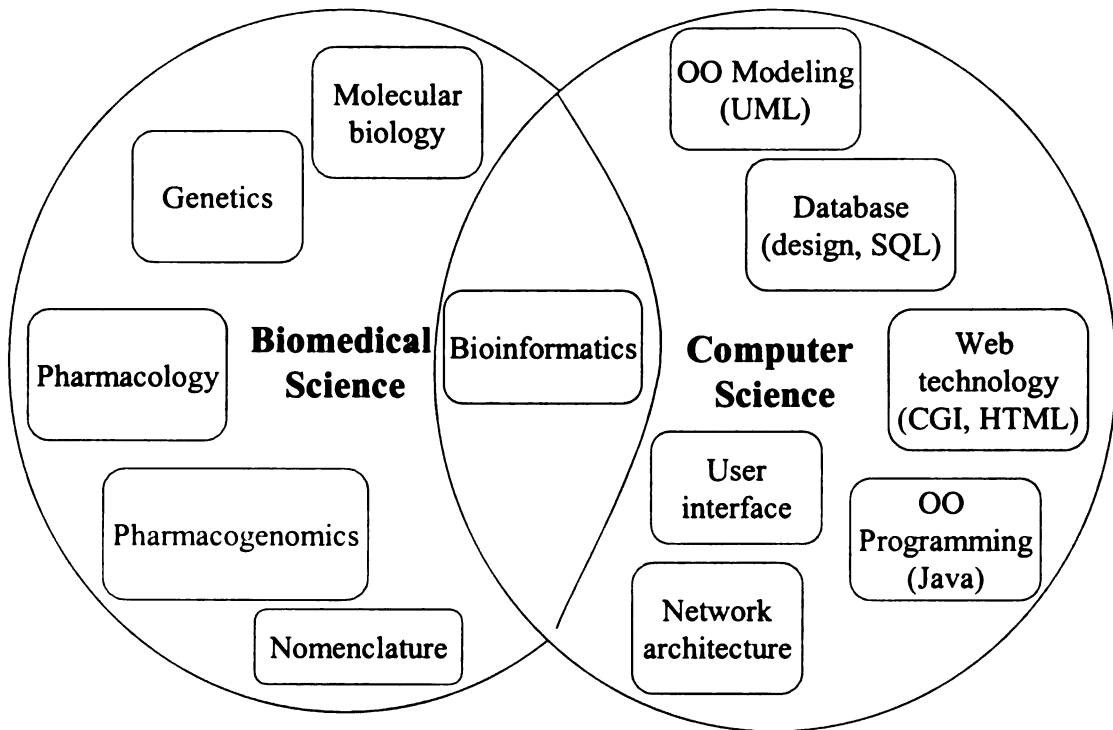


Fig. 5.3 Disciplines in both biomedicine and computer science involved in TPDS development

5.3 Methodology Provides the Solvent

There have been many discussions about the definition of bioinformatics since its birth. At the beginning of the development of bioinformatics, nucleotide and protein sequence analysis was one of the most important problems that molecular biology was facing. Many computer tools were developed to solve the difficulties. These included the sequence analysis tools such as BLAST, CLUSTAL W, algorithms such as BLOSUM, PAM, and sequence databases such as GenBank, SwissProt, etc. A popular definition of bioinformatics is: “The use of extensive computerized databases to solve

information problems in the biological sciences. These databases generally contain protein and nucleic acid sequences, genomes, etc. Bioinformatics also encompasses computer techniques such as 3-D molecular modeling.” (Biospace glossary, 2000).

As molecular biology itself is growing in the functional genomics era, this definition is somewhat narrow. The emergence of pharmacogenomics also requires overcoming the barriers among different disciplines, and demands a more complete view of the whole biomedical/biopharmaceutical system. Older methodologies in bioinformatics are too limited to satisfy these requirements. More comprehensive methodologies are essential if we want to solve these more complex problems. Although many problems are listed in Table 1.2, all of them are caused by one fact: lack of sound methodologies to solve them.

Table 5.1 summarizes the methodologies used in the development of the TPDS prototype. Many of the new methodologies used here are integrative and comprehensive products of some old methodologies. With these new methodologies, bioinformatics can evolve from a mere technical level tool for data analysis and management, to an integrative part of the science itself to support a more “complete” view. As an interdisciplinary science, methodology studies in bioinformatics can benefit the development of both biology and computer science. Improved methodology can help elucidate the existing concepts and discover new knowledge in bioscience. Meanwhile, the applications of the methodologies in biomedical field can stimulate new technology development in computer science.

Problem	Methodology
Transporter pharmacogenomics studies	A computer decision support system
Domain barriers and complexity	System requirement and domain boundary analysis, UML
Large amount and heterogeneity of data	A relational database with structured and integrated data
System prototyping	Iterative life cycle
Data modeling and system design	*Biomedical model - concept abstraction – UML modeling
Domain understanding and requirement analysis	*Identifying most important correlations, and the correlation among correlations
Concept abstraction	*Biomedical concept category listing
UML modeling	Use case - class diagram - sequence diagram
Lack of common literacy and reusable software	*Biomedical design patterns, UML
System architecture	Three-tiered client-server architecture
Sequence data collection	*Entrez + Classification +BLAST
Different database structure and format	*Data source analysis + terminology analysis, data integration processes
Data access	Database API-SQL
User interface design	Requirement analysis +other system review ↔ UI design

* Novel methodologies developed in this project

Table 5.1 The most important methodologies used in the development of TPDS

5.4 Expanding the Application

Transporter pharmacogenomics is only a part of biomedical studies, but it can be used as a prototype. The goal of my research was not only to establish decision support methodologies for transporter pharmacogenomics studies, but also the methodologies to solve the problems listed in Table 1.2 in Chapter 1. These problems come from different domains, including pharmacogenomics, bioinformatics, and decision support

itself. The data model developed suggests that UML can be used as a common modeling language, and the methodology for constructing the model can be a generic methodology using this language. The data structure designed here can also be used for database interoperation.

Many of the components developed in this decision support project can be expanded and reapplied to other DSS development in biomedicine. Because many of the components and processes are repeatable in other projects, this project can also be used as a template for broader applications. Not only can the methodologies be reused, but the components, such as the design patterns, the classes, the architectures, and the data integration processes, can also be reused.

We might not find one solution for all problems, but we may develop some reusable solutions that can be expanded to solve more problems.

REFERENCES

- Alexander,C., Ishikawa,S., Silverstein,M. (1977) A Pattern Language: Towns, Buildings, Construction. Oxford Univ Pr., pp. 1-1171.
- Anderson,M.P., Berger,H.A., Rich,D.P., Gregory,R.J., Smith,A.E., Welsh,M.J. (1991) Nucleoside triphosphates are required to open the CFTR chloride channel. *Cell*, 67, 775-784.
- Bader,G.D. and Hogue,C.W. (2000) BIND—a data specification for storing and describing biomolecular interactions, molecular complexes and pathways. *Bioinformatics*, 16, 465-477.
- Baker,P.G., Goble,C.A., Bechhofer,S., Paton,N.W., Stevens,R., Brass,A. (1999) An ontology for bioinformatics applications. *Bioinformatics*, 15, 510-520.
- Bejanin,S., Cervini,R., Mallet,J., Berrard,S. (1994) A unique gene organization for two cholinergic markers, choline acetyltransferase and a putative vesicular transporter of acetylcholine. *J Biol Chem.*, 269, 21944-21947.
- Berger,H.A., Anderson,M.P., Gregory,R.J., Thompson,S., Howard,P.W., Maurer,R.A., Mulligan,R., Smith,A.E., Welsh,M.J. (1991) Identification and regulation of the cystic fibrosis transmembrane conductance regulator-generated chloride channel. *J Clin Invest.*, 88, 1422-1431.
- Benson,D.A., Boguski,M.S., Lipman,D.J., Ostell,J., Ouellette,B.F., Rapp,B.A., Wheeler,D.L. (1999) GenBank. *Nucleic Acids Res.*, 27, 12-17.
- BioSpace glossary. Available at http://www.biospace.com/gls_detail.cfm?t_id=49804
- Blake,J.A., Eppig,J.T., Richardson,J.E., Davisson,M.T. (1998) The Mouse Genome Database (MGD): a community resource. Status and enhancements. The Mouse Genome Informatics Group. *Nucleic Acids Res.*, 26, 130-137.
- Carson,P.E., Flanagan,C.L., Ickes,C.E., Alving,A.S. (1956) *Science*, 124, 484.
- Chen,J., Radford,M.J., Wang,Y., Marciniak,T.A., Krumholz,H.M. (1999) Do “America’s Best Hospitals” perform better for acute myocardial infarction? *N. Engl. J. Med.*, 340, 286-292.
- Chen,P. P. (1976) The entity-relationship model: Towards a unified view of data. *ACM Trans. Database Systems*, 1, 9-36.
- Codd,E.F. (1970) A Relational Model of Data for Large Shared Data Banks. *CACM.*, 13, 377-387.

Cook,J.F., Rozenblit,J.W., Chacko,A.K., Martinez,R., Timboe,H.L. (1999) Meta-manager: a requirements analysis. *J Digit Imaging*, 12(2 Suppl 1):186-188.

Croop,J.M. (1998) Evolutionary relationship among ABC transporters. In Ambudkar, S.V. and Gottesman,M.M. (eds), *ABC Transporters:biochemical, cellular, and molecular aspects*. Academic Press, California, pp. 1-853.

Cui,Y., Konig, J., Buchholz,J.K., Spring,H., Leier,I., Keppler,D. (1999) Drug resistance and ATP-dependent conjugate transport mediated by the apical multidrug resistance protein, MRP2, permanently expressed in human and canine cells. *Mol Pharmacol.*, 55, 929-937.

Dalton,W.S., Miller,T.P. (1991) Multidrug resistance. In DeVita,V.T.Jr., Hellman,S., Rosenbery,S.A. (eds), *Cancer: Principles and Practice of Oncology*. Lippincott Williams & Wilkins Publishers, Philadelphia, pp. 1-13.

David, A., and Evans,P. (1993) *Genetic factors in drug therapy: clinical and molecular factors in drug therapy*. Cambridge University Press, New York, pp. 1-657.

Dayhoff,M.O. (1972) *Atlas of protein sequence and structure*. National Biomedical Research Foundation, Washington, DC, pp.1-418.

Dayhoff,M.O., Schwartz,R.M., Chen,H.R., Hunt,L.T., Barker,W.C., Orcutt,B.C. (1980) Nucleic acid sequence bank. *Science*, 209, 1182.

Dirckx,C., Donati,M.B., Iacoviello,L. (2000) Pharmacogenetics: a molecular sophistication or a new clinical tool for cardiologists? *Ital. Heart J.*, 1, 662-666.

Discala,C., Benigni,X., Barillot,E., Vaysseix,G. (2000) DBCat: a catalog of 500 biological databases. *Nucleic Acids Res.*, 28, 8-9.

Emilien,G., Ponchon,M., Caldas,C., Isacson,O., Maloteaux,J.M. (2000) Impact of genomics on drug discovery and clinical medicine. *QJM.*, 93, 391-423.

Erdmann,V.A. (1978) Collection of published 5S and 5.8S ribosomal RNA sequences. *Nucl. Acids Res.*, 5, r1-r13.

Erickson,J.D., Varoqui,H., Schafer,M.K., Modi,W., Diebler,M.F., Weihe,E., Rand,J., Eiden,L.E., Bonner,T.I., Usdin,T.B. (1994) Functional identification of a vesicular acetylcholine transporter and its expression from a "cholinergic" gene locus. *J Biol Chem.*, 269, 21929-21932.

Evans,W.E., Relling,M.V. (1999) Pharmacogenomics: translating functional genomics into rational therapeutics. *Science*, 286, 487-491.

Frishman,D., Heumann,K., Lesk,A., Mewes,H.W. (1998) Comprehensive, comprehensible, distributed and intelligent databases: current status. *Bioinformatics*, 14, 551-561.

Fuchs,C., Rosenvold,E.C., Honigman,A., Szybalski, W. (1978) A simple method for identifying the palindromic sequences recognized by restriction endonucleases: the nucleotide sequence of the *Ava*II site. *Gene*, 4, 1-23.

Gamma,E., Helm,R., Johnson,R., Vlissides,J. (1994) *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison Wesley, pp.1-395.

Gerlach,J.H., Kartner,N., Bell,D.R., Ling,V. (1986) Multidrug resistance. *Cancer Surv.*, 5, 25-46.

Gingeras,T.R., Milazzo,J.P., Roberts,R.J. (1978) A computer assisted method for the determination of restriction enzyme recognition sites. *Nucleic Acids Res.*, 5, 4105-4127.

Harger,C., Skupski,M., Bingham,J., Farmer,A., Hoisie,S., Hraber,P., Kiphart,D., Krakowski,L., McLeod,M., Schwertfeger,J., Seluja,G., Siepel,A., Singh,G., Stamper,D., Steadman,P., Thayer,N., Thompson,R., Wargo,P., Waugh,M., Zhuang,J.J., Schad,P.A. (1998) The Genome Sequence DataBase (GSDB): improving data quality and data access. *Nucleic Acids Res.*, 26, 21-26.

Harmon,P. and Watson, M. (1997) *Understanding Uml: The Developer's Guide: With a Web-Based Application in Java*. Morgan Kaufmann Publishers, pp.1-340.

Hess,P. and Cooper, D. (1999) Impact of pharmacogenomics on the clinical laboratory. *Mol Diagn.*, 4, 289-298.

Higgins,C.F. (1992) ABC transporters:from microorganisms to man. *Curr. Opin. Cell Biol.*, 8:67-113.

Hipfner,D.R., Deeley,R.G., Cole,S.P. (1999) Structural, mechanistic and clinical aspects of MRP1. *Biochim Biophys Acta.*, 1461, 359-376.

Hoffmeyer,S., Burk,O., von Richter,O., Arnold,H.P., Brockmoller,J., Johne,A., Cascorbi,I., Gerloff,T., Roots,I., Eichelbaum,M., Brinkmann,U. (2000) Functional polymorphisms of the human multidrug-resistance gene: multiple sequence variations and correlation of one allele with P-glycoprotein expression and activity in vivo. *Proc. Natl. Acad. Sci. U. S. A.*, 97, 3473-3478.

Hu,M., Retz,W., Baader,M., Pesold,B., Adler,G., Henn,F.A., Rosler,M., Thome,J. (2000) Promoter polymorphism of the 5-HT transporter and Alzheimer's disease. *Neurosci. Lett.*, 294, 63-65.

Hyde,S.C., Emsley,P., Hartshorn,M.J., Mimmack,M.M., Gileadi,U., Pearce,S.R., Gallagher,M.P., Gill,D.R., Hubbard,R.E., Higgins,C.F. (1990) Structural model of ATP-binding proteins associated with cystic fibrosis, multidrug resistance and bacterial transport. *Nature*, 346, 362-365.

Inui,K.I., Masuda,S., Saito,H. (2000) Cellular and molecular aspects of drug transport in the kidney. *Kidney Int.*, 58, 944-958.

Iversen,L. (2000) Neurotransmitter transporters: fruitful targets for CNS drug discovery. *Mol. Psychiatry*, 5, 357-362.

Jedlitschky,G., Leier,I., Buchholz,U., Barnouin,K., Kurz,G., Keppler, D. (1996) Transport of glutathione, glucuronate, and sulfate conjugates by the MRP gene-encoded conjugate export pump. *Cancer Res.*, 56, 988-994.

Kennedy,G.C. (2000) The impact of genomics on therapeutic drug development. *EXS.*, 89, 1-10.

Keppler,D., Leier,I., Jedlitschky,G., Konig,J. (1998) ATP-dependent transport of glutathione S-conjugates by the multidrug resistance protein MRP1 and its apical isoform MRP2. *Chem. Biol. Interact.*, 111-112:153-161.

Kim,D.K., Lim,S.W., Lee,S., Sohn,S.E., Kim,S., Hahn,C.G., Carroll,B.J. (2000) Serotonin transporter gene polymorphism and antidepressant response. *Neuroreport.*, 11, 215-219.

Kim,J.W., Kim,D.H., Kim,S.H., Cha,J.K. (2000) Association of the dopamine transporter gene with Parkinson's disease in Korean patients. *J. Korean Med. Sci.*, 15, 449-51.

Korn,L.J., Queen,C.L., Wegman,M.N. (1977) Computer analysis of nucleic acid regulatory sequences. *Proc. Natl. Acad. Sci. U. S. A.*, 74, 4401-4405.

Korson,T. and McGregor,J. (1990) Understanding Object-Oriented: A Unifying Paradigm. *CACM*, 9, 40-60.

Kozaczynski,W., and Kuntzmann-Combelle,A. (1993) What it Takes to Make OO Work. *IEEE Software*, 1, 20-23.

Loe,D.W., Deeley,R.G., Cole,S.P. (1996) Biology of the multidrug resistance-associated protein, MRP. *Eur. J. Cancer*, 32A, 945-957.

Le Couteur,D.G., Leighton,P.W., McCann,S.J., Pond,S. (1997) Association of a polymorphism in the dopamine-transporter gene with Parkinson's disease. *Mov. Disord.*, 12, 760-763.

Lee,V.H. (2000) Membrane transporters. *Eur. J. Pharm. Sci.*, 11, S41-S50.

Lehninger,A.L., (1993) *Principles of Biochemistry*, Worth Publishing, New York, pp. 1-1013.

Lesch,K.P., Bengel,D., Heils,A., Sabol,S.Z., Greenberg,B.D., Petri,S., Benjamin,J., Muller,C.R., Hamer,D.H., Murphy,D.L. (1996) Association of anxiety-related traits with a polymorphism in the serotonin transporter gene regulatory region. *Science*, 274, 1527-1531.

Lewis,R.A., Shroyer,N.F., Singh,N., Allikmets,R., Hutchinson,A., Li,Y., Lupski,J.R., Leppert,M., Dean,M. (1999) Genotype/Phenotype analysis of a photoreceptor-specific ATP-binding cassette transporter gene, ABCR, in Stargardt disease. *Am. J. Hum. Genet.*, 64, 422-434.

Lodish,H., Baltimore,D., Berk,A., Zipursky,S.L., Matsudaira,P., Darnell,J. (1995) *Molecular Cell Biology*. Scientific American Books, New York, pp. 1-1417.

Mancinelli,L., Cronin,M., Sadee,W. (2000) Pharmacogenomics: The Promise of Personalized Medicine. *AAPS Pharmsci*, 2, 4.

March,R. (2000) Pharmacogenomics: the genomics of drug response. *Yeast*, 17, 16-21.

Marshall,A. (1997) Laying the foundations for personalized medicines. *Nat. Biotechnol.*, 15, 954-957.

Martin,M.G., Lostao,M.P., Turk,E., Lam,J., Kreman,M., Wright,E.M. (1997) Compound missense mutations in the sodium/D-glucose cotransporter result in trafficking defects. *Gastroenterology*, 112, 1206-1212.

Martinez,R., Rozenblit,J., Cook,J.F., Chacko,A.K., Timboe,H.L. (1999) Virtual management of radiology examinations in the virtual radiology environment using common object request broker architecture services. *J. Digit. Imaging*, 12:181-185.

McCallum,D. and Smith,M. (1977) Computer processing of DNA sequence data. *J. Mol. Biol.*, 116, 29-30.

Meyer,B. (1988) *Object-Oriented Software Construction*, Prentice Hall, pp.1-534.

Muller,M. (2000) Available at <http://www.med.rug.nl/mdl/tab3.htm>

Muller,M., Meijer,C., Zaman,G.J., Borst,P., Scheper,R.J., Mulder,N.H., de Vries,E.G., Jansen,P.L. (1994) Overexpression of the gene encoding the multidrug resistance-associated protein results in increased ATP-dependent glutathione S-conjugate transport. *Proc. Natl. Acad. Sci. U. S. A.*, 91, 13033-13037.

Nebert,D.W. (1997) Polymorphisms in drug-metabolizing enzymes: what is their clinical relevance and why do they exist? *Am. J. Hum. Genet.*, 60, 265-271.

Nebert,D.W. (1999) Pharmacogenetics and pharmacogenomics: why is this relevant to the clinical geneticist? *Clin Genet.*, 56, 247-258.

Oh,D.M., Han,H.K., Amidon,G.L. (1999) Drug transport and targeting. Intestinal transport. *Pharm. Biotechnol.*, 12, 59-88.

OMG Unified Modeling Language Specification. (1999) Object Management Group, Inc., pp.1-808.

Palacin,M., Bertran,J., Zorzano,A. (2000) Heteromeric amino acid transporters explain inherited aminoacidurias. *Curr. Opin. Nephrol. Hypertens.* 9, 547-553.

Paton,N.W., Khan,S.A., Hayes,A., Moussouni,F., Brass,A., Eilbeck,K., Goble,C.A., Hubbard,S.J., Oliver,S.G. (2000) Conceptual modelling of genomic information. *Bioinformatics*, 16, 548-557.

Paulsen,I.T., Sliwinski,M.K., Saier,M.H. Jr. (1998a) Microbial genome analyses: global comparisons of transport capabilities based on phylogenies, bioenergetics and substrate specificities. *J. Mol. Biol.*, 277, 573-592.

Paulsen,I.T., Sliwinski,M.K., Nelissen,B., Goffeau,A., Saier,M.H. Jr. (1998b) Unified inventory of established and putative transporters encoded within the complete genome of *Saccharomyces cerevisiae*. *FEBS Lett.*, 430, 116-125.

Pipas,J.M., McMahon,J.E. (1975) Method for predicting RNA secondary structure. *Proc. Natl. Acad. Sci. U. S. A.*, 72, 2017-2021.

Pollock,B.G., Ferrell,R.E., Mulsant,B.H., Mazumdar,S., Miller,M., Sweet,R.A., Davis,S., Kirshner,M.A., Houck,P.R., Stack,J.A., Reynolds,C.F., Kupfer,D.J. (2000) Allelic variation in the serotonin transporter promoter affects onset of paroxetine treatment response in late-life depression. *Neuropsychopharmacology*, 23, 587-90.

Rechenmann,F. (2000) From data to knowledge. *Bioinformatics*, 16, 411.

Riehle,D. and Zullighoven, H. (1996) Understanding and Using Patterns in Software Development. Theory and Practice of Object Systems 2, 1, 3-13.

Roberts,R.J. (1981) Restriction and modification enzymes and their recognition sequences. Nucleic Acids Res., 9, r75-96.

Roses,A.D. (2000) Pharmacogenetics and pharmacogenomics in the discovery and development of medicines. Novartis Found Symp., 229, 63-66.

Ross,D.D., Yang,W., Abruzzo,L.V., Dalton,W.S., Schneider,E., Lage,H., Dietel,M., Greenberger,L., Cole,S.P., Doyle,L.A. (1999) Atypical multidrug resistance: breast cancer resistance protein messenger RNA expression in mitoxantrone-selected cell lines. J. Natl. Cancer Inst., 91, 429-433.

Rumbaugh,J., Blaha,M., Premerlani,W., Eddy,F., Rumbaugh,J., Lorenson,W. (1991) Object-Oriented Modeling and Design. Prentice Hall, pp.1-500.

Rund,D., Azar,I., Shperling, O. (1999) A mutation in the promoter of the multidrug resistance gene (MDR1) in human hematological malignancies may contribute to the pathogenesis of resistant disease. Adv. Exp. Med. Biol., 457, 71-75.

Sadee,W. (1998) Genomics and drugs: finding the optimal drug for the right patient. Pharm. Res., 15, 959-63.

Sadee,W., Drubbisch,V., Amidon,G.L. (1995) Biology of membrane transport proteins. Pharm Res., 12, 1823-1837.

Saier,M.H. Jr. (2000) A functional-phylogenetic classification system for transmembrane solute transporters. Microbiol. Mol. Biol. Rev., 64, 354-411.

Saier's Transport Classification Page. (2000) Available at <http://www-biology.ucsd.edu/~msaier/transport/toc.html#1A>

Schulze-Kremer,S. (1998) Ontologies for molecular biology. Pac. Symp. Biocomput., 695-706.

Sheppard,D.N., Welsh,M.J. (1999) Structure and function of the CFTR chloride channel. Physiol. Rev., 79, S23-45.

Shi,M.M., Bleavins,M.R., de la Iglesia, F.A. (1999) Technologies for detecting genetic polymorphisms in pharmacogenomics. Mol. Diagn., 4, 343-351.

Silverman,J.A. (1999) Multidrug-resistance transporters. Pharm. Biotechnol., 12, 353-386.

Smeraldi,E., Zanardi,R., Benedetti,F., Di Bella,D., Perez,J., Catalano,M. (1998) Polymorphism within the promoter of the serotonin transporter gene and antidepressant efficacy of fluvoxamine. *Mol. Psychiatry*, 3, 508-511.

Studnicka,G.M., Rahn,G.M., Cummings,I.W., Salser,W.A. (1978) Computer method for predicting the secondary structure of single-stranded RNA. *Nucleic Acids Res.*, 5, 3365-3387.

Sugiyama,Y., Kusuhara,H., Suzuki,H. (1999) Kinetic and biochemical analysis of carrier-mediated efflux of drugs through the blood-brain and blood-cerebrospinal fluid barriers: importance in the drug delivery to the brain. *J. Control Release*, 62, 179-186.

Tamai,I. and Tsuji,A. (2000) Transporter-mediated permeation of drugs across the blood-brain barrier. *J. Pharm. Sci.*, 89, 1371-1388.

Tan,E.K., Khajavi,M., Thornby,J.I., Nagamitsu,S., Jankovic,J., Ashizawa,T. (2000) Variability and validity of polymorphism association studies in Parkinson's disease. *Neurology*, 55, 533-538.

Tsoka.S. and Ouzounis,C.A. (2000) Recent developments and future directions in computational genomics. *FEBS Lett.*, 480, 42-48.

Vogel,F. (1959) Moderne probleme der Humangenetik. *Ergeb Inn Med Kinderheilkd*, 12, 52-125.

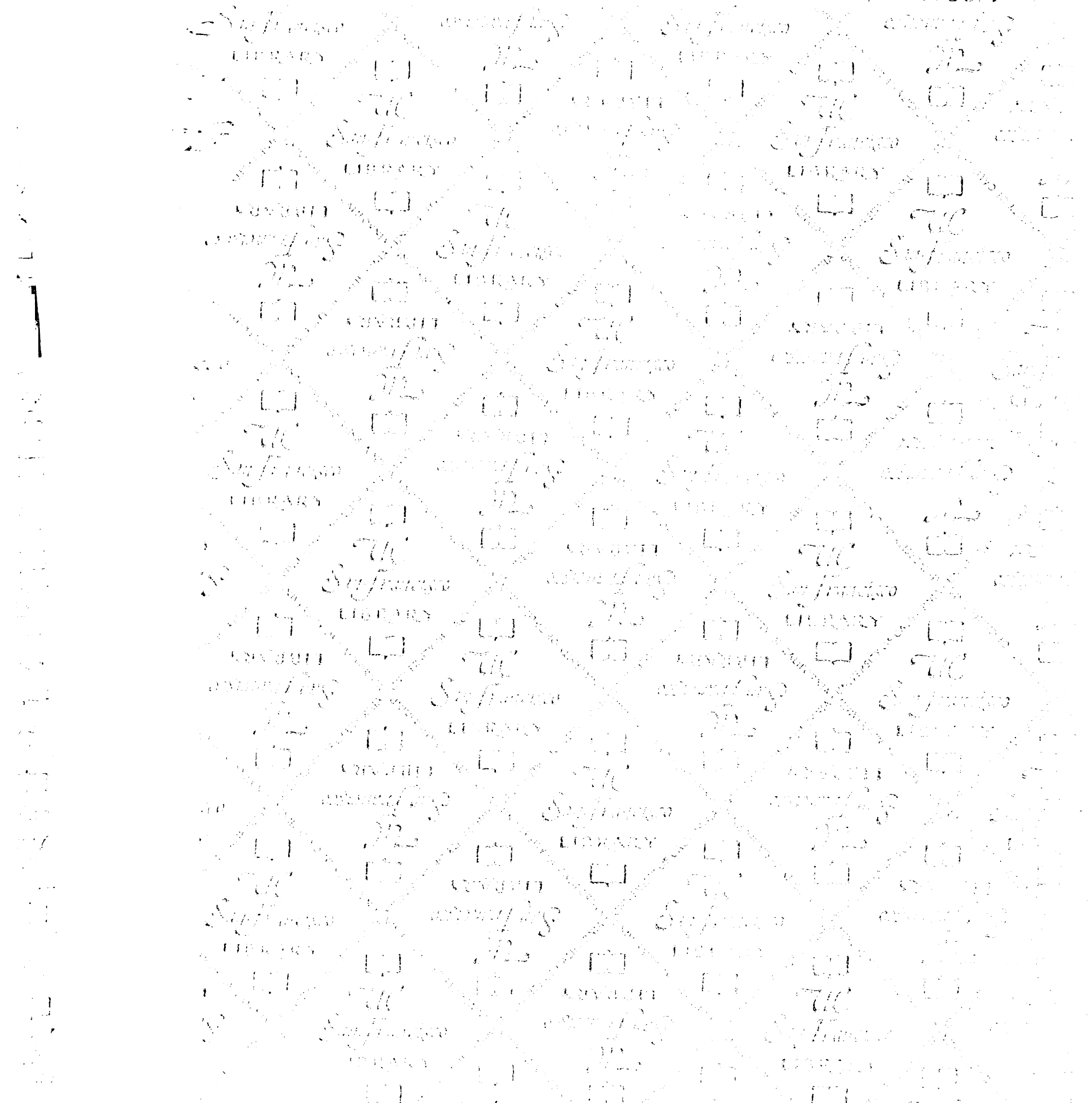
White,J.A., McAlpine,P.J., Antonarakis,S., Cann,H., Eppig,J., Frazer,K., Frezal,J., Lancet,D., Nahmias,J., Pearson,P., Peters,J., Scott,A., Scott,H., Spurr,N., Talbot,C.Jr, Povey,S. (1997) Guidelines for human gene nomenclature. HUGO Nomenclature Committee. *Genomics*, 45, 468-471.

Winsberg,B.G. and Comings,D.E. (1999) Association of the dopamine transporter gene (DAT1) with poor methylphenidate response. *J. Am. Acad. Child Adolesc. Psychiatry*, 38, 1474-1477.

Yan,Q., Sadée,W. (2000) Human Membrane Transporter Database: A Web-Accessible Relational Database for Drug Transport Studies and Pharmacogenomics . *AAPS Pharmsci*, 2, 20.

Zanardi,R., Benedetti,F., Di Bella,D., Catalano,M., Smeraldi,E. (2000) Efficacy of paroxetine in depression is influenced by a functional polymorphism within the promoter of the serotonin transporter gene. *J. Clin. Psychopharmacol.* 20, 105-107.

Zhu,J. and Zhang,M.Q. (1999) SCPD: a promoter database of the yeast *Saccharomyces cerevisiae*. *Bioinformatics*, 15, 607-611.



For Not to be taken
from the room.
reference

628388



3 1378 00628 3884

