Web services 和 XML 技术在生物信息数据发布及整合中的应用

遗传学专业

硕士研究生　李校　　指导教师　张义正教授

摘　要

上个世纪 90 年代，人类基因组计划的实施标志着生物学的发展进入了基因组时代。这个时代最显著的特征是各种分子数据呈现爆炸性增长。随着 WWW 的蓬勃发展，通过因特网来传播生物信息是最快捷和最方便的手段。然而，由于生物数据的复杂性、庞大性以及生物数据格式的复杂性、多样性，使得生物数据的获取和整合变得十分困难。与传统数据库中高度结构化的数据相比，Web 上的数据最大特点就是结构化特征较弱，通常是半结构化的，甚至是无结构化的，而且每个数据源都是分散的、自治的，生物学家们要想获取相关数据异常得艰难。因此，如何有效地发布、传输及整合来自不同数据源的数据是当前生物信息学和基因组、蛋白组科学中的重要课题之一。以 XML 技术为核心的 Web Services 是下一代 Internet，已经被业界广泛应用于电子商务。作为一种完全结构化的数据格式，XML 被生物信息团体用来描述生物学数据。本论文以 Web Services 跨平台的特点，在分布式的环境中发布和整合以 XML 格式的生物数据。

利用抑制削减杂交和基因芯片技术获得一批黄孢原毛平革菌(*Phanerochaete chrysosporium*)特异表达的 EST 序列，使用 Phrap、EMBOSS、Blast、GENSCAN、MZEF 软件，基于 Linux 操作系统，构建了 EST 序列分析系统，完成了从 EST 和基因组 Blast 数据库的构建，载体序列的去除， EST 序列的分类和组装，EST 序列在基因组上的定位，外显子和内含子的识别以及基因预测。并通过使用 perl 语言结合 bioperl 模块编写的脚本程序使分析过程

自动化，从而可以快速地对大批 EST 序列进行分析，为克隆相关基因及研究黄孢原毛平革菌功能基因组学提供有用的信息。

关键词 生物数据整合 生物数据发布 Web Services eXtensible Markup Language (XML) 生物信息学 黄孢原毛平革菌 EST

Application of Web Services and XML in Bioinformatics Data Distribution and Integration

Major: Genetics

Student: Li Xiao **Advisor:** Zhang Yizheng

Abstract

In 90s of last century, the development of human genomics project indicated that biology has stepped into the age of genome. The most remarkable characteristic of the age is the volume of biology data growing at an exponential rate. More and more genomes are being sequenced and annotated, and the data of proteins and genes are accumulated. With the rapid development of WWW (World Wide Web), biological data are mostly digital and stored in a wide variety of formats in heterogeneous systems. Biological data exist all over the world as various web sites, which provide biologists with much useful information. However, the complexity of biological data and the variety of data formats make it difficult to retrieve and integrate the interesting data. Comparing with the traditional structured data, the biology ones locating at Web are semi-structured or no-structured, and have heterogeneous formats. Therefore, retrieving and integrating biology data is a very important task. Recently, it is widely recognized that exchange, distribution, and integration of biology data are the keys to improve bioinformatics and genomics in post-genomic era. The eXtensible Markup Language (XML) is rapidly spreading as an emerging standard for structuring document for exchanging and integrating data on the World Wide Web (WWW). Web service is the next generation of WWW and founded upon the open standards of W3C (World Wide Web Consortium) and IETF (Internet Engineering Task Force). This paper presents XML and Web Services

technologies and their use for an appropriate solution to the bioinformatics data exchange and integration problem.

A number of differentially-expressed cDNA fragments were obtained from *Phanerochaete chrysosporium* by using Suppression Subtractive Hybridization (SSH) and Microarray techniques and 433 of them were sequenced. To manage and analyze these EST data, based on Linux operating system, the Phrap, EMBOSS, Blast, GENSCAN, MZEF software were used to construct a platform. The platform includes constructing EST and genome databases, removing vector sequences, sorting and assembling sequences, locating on genome, identifying exons and introns, and predicting genes. Moreover, using bioperl modules, the scripts written with perl language enable analysis automatically. Results demonstrated that the robust platform could accelerate data analysis for large-scale EST sequences and offer useful information for cloning correlative genes and studying the functional genomics of *Phanerochaete chrysosporium*.

Key words: biological data integration, biological data distribution, eXtensible Markup Language (XML), web services, bioinformatics, *Phanerochaete chrysosporium*, EST .

第一部分　发布及整合核酸数据的 Web Services 平台的构建

摘　要

上个世纪 90 年代，人类基因组计划的实施标志着生物学的发展进入到了基因组时代。这个时代最显著的特征是各种分子数据呈现爆炸性增长。因此，如何有效地发布、传输及整合来自不同数据源的数据是当前生物信息学和基因组、蛋白组科学中重要的课题之一。随着 WWW 的蓬勃发展，通过因特网来传播生物信息是最快捷和方便的手段。由于生物数据的复杂性，庞大性，利用计算机和网络技术来收集，整理和交流生物数据就成为必然，也是当今生物信息学蓬勃发展的原因。然而，与传统数据库中高度结构化的数据相比，Web 上的数据最大特点就是结构化特征较弱，往往是半结构化（semi-structured）的，甚至是无结构化（no-structured）的。而且每个数据源都是分散的，自治的（autonomous），生物学家们要想获取相关数据异常得艰难。本课题旨在建立一个整合的生物信息查询系统，利用 Web Services 技术，采用 XML（eXtensible Markup Language，可扩展标识语言）为数据模型，收集整合来自各个不同数据源的生物信息数据。

关 键 词　生 物 数 据 整 合 (biological data integration)　Web Services XML(eXtensible Markup Language)　JAVA　Perl　生物信息学

1　引言

以人类基因组计划为代表的基因组时代，产生了并正加速产生着庞大的生物数据。几乎生物体各个水平的数据正在呈爆炸性的增长，从 DNA，RNA，SNP，蛋白到代谢网络等等。截止到 2004 年 5 月，GenBank 中收录的核酸数据已经达到 201 亿碱基。然而，由于收集者的兴趣和采用的数据存储方式的不同，生物数据以不同的数据格式分散存储在 INTERNET 中，我们可以称之为

生物数据的异质性：包括数据存放的地点（数据源），数据的文件格式，数据库的不同，数据库内容的关注点的不同，操作平台的不同以及检索系统的不同。生物学家如果想获得某一完整的数据，就不得不在各个数据源之间进行查找。随着后基因组时代的来临，系统生物学开始兴起，以整合的系统的观点分析生物学问题成为必然，因此，数据整合就成为系统生物学发展的起始的必然的阶段而受到广泛的重视(Stein, 2003; Stein, 2002; Kitano, 2002)。此外，因为数据整合是为了获得相关的所有数据，在生物制药中的应用也受到了广泛的重视，因为，通过整合所有相关的数据，研究者就能找出相关组分之间的联系，因而也就能更快地找到药物靶标，更快地实现药物设计。

1.1 生物数据整合的概念和困难

数据整合是和应用关联（application interoperability）、应用整合（application integration）相联系但又不同的概念。数据整合是指收集和存储来自各个不同数据源的在生物学意义上相互联系的数据。应用关联是指规范化数据接口（interface），使得从一个应用程序产生的数据能直接作为另一个应用程序输入的数据。应用整合则包括了以上的 2 个方面，是数据整合和数据关联结合在一起的使用(Siepel et al., 2001)。

生物数据整合是个复杂的工程，从技术角度来说，必须克服以下几个障碍：

1） 分散的数据源：计算机和信息技术应用于分子生物学研究以来，短时间内产生数以万计的数据源(Lacroix, 2002)。

2） 数据格式的不一致：有传统的高度结构化的关系性数据库数据格式，也有半结构化的 flat files，text 文本格式，完全无结构化的 HTML 格式，同时也有从应用程序中新产生的数据格式（如 Blast 等）。数据格式的统一严重阻碍了信息交流与整合。

3） 数据源更新的不同步性：各个数据源是自治的，分属于不同的研究机构和私营企业。出去公开和不公开的差别外，各个数据源的更新是不同步的(Searls, 2003)。

当前，生物数据普遍采用 flatfile 格式(包括 Genbank，EMBL，DDBJ，FASTA 等格式)来描述和储存。由于缺乏分布式应用，所有的数据必须下载到本地整

合。而且，flatfile 格式是无结构化的标识语言，为了抽提和整合重要的或者感兴趣的信息，解析 flatfile 格式是困难的。除了 flatfile 格式，另一个被广泛用来传播数据的是 HTML(Hypertext Markup Language)，作为一种半结构化的标识语言，解析同样是困难的。为了克服上述格式的不足，XML 作为一种新的可扩展的标识语言诞生了。最初的时候，XML 是为了解决在电子商务中传输和整合数据的。然而，由于其良好的可扩展性，开始被用于去描述生物、化学、数学等专门学科领域的数据(Archard et al., 2001; Haas et al., 2002; Benini et al., 2003; Goesmann et al., 2003) 。

1.2 Web Services

以 XML 为核心的被誉为下一代的互连网的 Web Services 如今已经开始应用于电子商务中，Web Services 具有一些重要的特性从而克服了上述的困难(De et al., 2004)：

1） Web Services 形成一个分布式环境并支持远程过程调用(RPCs, Remote Procedure Calls)。作为一种跨平台的分布式环境在生命科学中日益显得重要。通过分布式的环境，能最大效益地利用互连网。这样，一些花费大（包括硬件本身及其所处理的庞大数据）的应用程序可以直接在远程的大型服务器中调用执行。

2） Web Services 是以 XML 为基础的。这一点非常重要，因为 XML 具有完全结构化的特点，利用计算机进行解析是方便的.

3） Web Services 的组件是松耦合的。传统的数据整合必须紧紧捆绑于上游的数据源。当数据源发生改变时，数据整合就会失败。

4） 在通信方面，Web Services 依赖于开放式的 Web 标准：TCP/IP, HTTP 和 XML。包含过程调用、描述、发布、查找、绑定等内容的更高级的协议基于 XML 语法。因为多种平台都广泛支持开放式标准，并且开放式标准可越过防火墙传输信息，所以开放式标准能够确保内部可操作性以及无须紧紧捆绑于上游数据源。

SOAP (Simple Object Access Protocol), WSDL (Web Service Description

合。而且，flatfile 格式是无结构化的标识语言，为了抽提和整合重要的或者感兴趣的信息，解析 flatfile 格式是困难的。除了 flatfile 格式，另一个被广泛用来传播数据的是 HTML(Hypertext Markup Language)，作为一种半结构化的标识语言，解析同样是困难的。为了克服上述格式的不足，XML 作为一种新的可扩展的标识语言诞生了。最初的时候，XML 是为了解决在电子商务中传输和整合数据的。然而，由于其良好的可扩展性，开始被用于去描述生物、化学、数学等专门学科领域的数据(Archard et al., 2001; Haas et al., 2002; Benini et al., 2003; Goesmann et al., 2003) 。

1.2 Web Services

以 XML 为核心的被誉为下一代的互连网的 Web Services 如今已经开始应用于电子商务中，Web Services 具有一些重要的特性从而克服了上述的困难(De et al., 2004):

1） Web Services 形成一个分布式环境并支持远程过程调用(RPCs, Remote Procedure Calls)。作为一种跨平台的分布式环境在生命科学中日益显得重要。通过分布式的环境，能最大效益地利用互连网。这样，一些花费大（包括硬件本身及其所处理的庞大数据）的应用程序可以直接在远程的大型服务器中调用执行。

2） Web Services 是以 XML 为基础的。这一点非常重要，因为 XML 具有完全结构化的特点，利用计算机进行解析是方便的.

3） Web Services 的组件是松耦合的。传统的数据整合必须紧紧捆绑于上游的数据源。当数据源发生改变时，数据整合就会失败。

4） 在通信方面，Web Services 依赖于开放式的 Web 标准：TCP/IP，HTTP 和 XML。包含过程调用、描述、发布、查找、绑定等内容的更高级的协议基于 XML 语法。因为多种平台都广泛支持开放式标准，并且开放式标准可越过防火墙传输信息，所以开放式标准能够确保内部可操作性以及无须紧紧捆绑于上游数据源。

SOAP (Simple Object Access Protocol), WSDL (Web Service Description

Language) and UDDI (Universal, Description, and Discovery Integration)三个组件组成了今天 Web Services 的核心。

简单对象访问协议(SOAP)是一种用于发送可扩展格式的信息的机制。可以使用 SOAP 发送消息，也可以发送用于 XML 格式编码的远程过程调用。

Web 服务描述语言(WSDL)，WSDL 使用 XML 格式描述一个远程服务.

通用描述、发现和集成标准(UDDI)是一种注册表的定义，可以在这类注册表中发布和查找服务及其服务提供者。在这里，一个服务就是一个具体的应用程序。BLAST（Basic local alignment search tool）就是一个服务，DDBJ(DNA Data Bank of Japan)是其中的一个服务提供者。

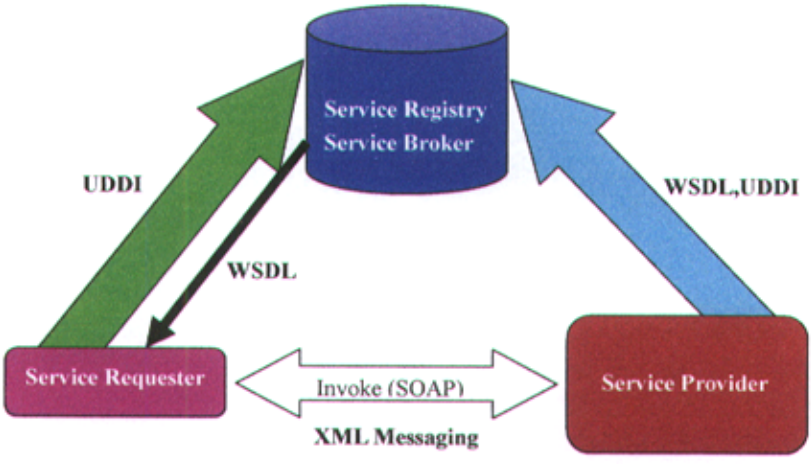

图 1　Web Services 体系结构：SOAP、WSDL 和 UDDI 的联系及工作原理

Figure.1　The architecture of web services describes the relationship and working principle of these pieces (SOAP, WSDL, and UDDI).

1.3 XML 数据模型

在因特网上被主要用来传播信息的语言是 HTML（Hypertext Markup Language, 超文本链接标示语言），用 HTML 书写的文档数据模型是半结构化

Language) and UDDI (Universal, Description, and Discovery Integration)三个组件组成了今天 Web Services 的核心。

简单对象访问协议(SOAP)是一种用于发送可扩展格式的信息的机制。可以使用 SOAP 发送消息，也可以发送用于 XML 格式编码的远程过程调用。

Web 服务描述语言(WSDL)，WSDL 使用 XML 格式描述一个远程服务.

通用描述、发现和集成标准(UDDI)是一种注册表的定义，可以在这类注册表中发布和查找服务及其服务提供者。在这里，一个服务就是一个具体的应用程序。BLAST（Basic local alignment search tool）就是一个服务，DDBJ(DNA Data Bank of Japan)是其中的一个服务提供者。

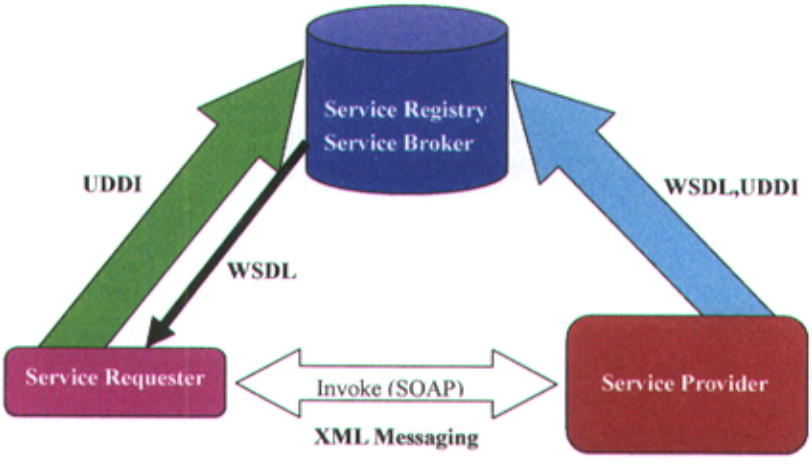

图 1　Web Services 体系结构：SOAP、WSDL 和 UDDI 的联系及工作原理

Figure.1　The architecture of web services describes the relationship and working principle of these pieces (SOAP, WSDL, and UDDI).

1.3 XML 数据模型

在因特网上被主要用来传播信息的语言是 HTML（Hypertext Markup Language, 超文本链接标示语言），用 HTML 书写的文档数据模型是半结构化

的。为了有效利用 WWW 数据，需要处理半结构化数据源，解决半结构化数据的查询与整合问题。寻找一个半结构化的数据模型是解决问题的关键。

XML（eXtensible Markup Language，可扩展标识语言）在最近几年已经成为信息技术中最引人注目的技术之一(Serge, 1999)。不管使用 XML 的是哪种平台或者应用程序，它都可以用来描述信息。现在，XML 已用于许多行业中，比如，医疗，汽车制造以及多数应用（如信息交换，数据库管理和内容管理）中。

XML 是由 W3C（http://www.w3.org/XML）设计的一种元标识语言（meta-markup language）。它来源于 SGML（the Standard Generalized Markup Language），SGML 是定义描述不同类型电子文档结构和内容的国际标准。SGML 允许你利自定义标签来定义你自己的标识语言，因此它是一种元语言（meta-language）。XML 继承了这个特点（HTML 虽然也来源与 SGML，但不具这个特点），就象 SGML，XML 也是一种元语言，允许你自己定义一套标签应用于一个或多个文档。XML 是一群 SGML 的审核人员兼 Web 专家所共同制定的，而且不是为了要取代 HTML。就应用层面来看，XML 介于 SGML 与 HTML 之间，它克服了 SGML 过于复杂和 HTML 缺乏灵活的缺点。制定 XML 一个非常重要的目标是支持多重应用领域以及易于开发相关应用软件。XML 特别适合用来做数据交换的中间格式，因为它只是用来规范化存储数据，对数据的应用则交给不同的应用程序。因为 XML 文档在文档结构上几乎没有限制，所以需要一些方法将约束条件置于文档上。为 XML 文档提供约束可以使得应用程序读取它们时更具完整性。DTD（Document Type Definition，文档类型定义）是一种将约束置于 XML 文档的结构上的方法。因此，只要遵循同一个 DTD 约束的所有 XML 文档都能方便地进行数据交流。

XML 用来作为数据格式有如下几个优点：

1） 可规范化（ standard）

2） 容易操作和查询（ Easily Manipulated and Queried）

3） 可扩展的 （Can be extended）

XML 在科学研究中已引起广泛的重视。许多商业和政府研究机构都采用 XML 来作为他们管理数据的标准。化学和数学研究团体很早就开始利用 XML

来书写和交换科研数据，分别命名为 CML（the Chemical Markup Language，http://www.xml-cml.org/ ）和 MathML（the Mathematical Markup Language，http://www.w3.org/Math/ ）。CML 和 MathML 实际上是定义了 DTD，遵循该 DTD 约束的 XML 文档之间能方便地进行数据交流。

近年来，国外的生物信息研究机构也开始采用 XML 作为数据格式，用来注释序列。几个公共组织定义了 DTD：

1) The Bioinformatic Sequence Markup Language(Lichu et al., 2002)（BSML http://www.visualgenomics.com/products/index.html）

该 DTD 用来规范对 DNA，RNA 和蛋白序列以及它们的图象特征的注解。我们发现这个 DTD 定义的文档结构和 EMBL/Genebank/DDBJ 数据库的信息结构非常相似（http://www.emi.ac.uk，http://www.ncbi.nlm.nih.gov，http://www.ddbj.nig.ac.jp ）。

2）The BIOpolymer Markup Language（BioML,htpp://www.protemetrics.com/BIOML/ ）

与 BSML 有点不同的是 BioML 的目标是“allow the expression of complex annotation for protein and nucleotide sequence information, BioML was designed to mimic the hierarchical structure of a living organism” (Fenyo,1999)。该 DTD 定义了一些不同来源信息的数据整合的标准。

这 2 个团体（BSML and BioML）从商业角度上来说开发 XML 语言的目的是相同的：它们都是在指定一个编码生物信息的公共标准。另一方面，按照这个标准出售相关的软件，例如：专门的浏览器，数据整合和数据管理工具。

3) The taxonomic markup language consists of a DTD for the description of taxonomic relationships between organisms(Gilmour et al., 2000)。

4）XML 同样被基因存在论联盟（the gene ontology consortium, http://www.geneontology.org）所采用(Gene Ontology Consortium, 2000)，用来提供对于描述分子功能，生物过程以及基因产品细胞内定位信息的可控词汇（指可被接纳的词汇，包括科学命名和通用名称以及可接受的名称缩写等等）。

5）BioXML project（http://bioxml.org）更作为一个国际性的组织，正在加

强 XML 在生命科学中的应用，并使其规范化。而且 BioXML 采取开放源代码的政策，任何人都能参与开发和错误修订。

6）SBML(Hucka, et al., 2003)：the Systems Biology Markup Language(http://www.sbml.org/)

表 1 XML 在生命科学中的应用

Table 1 The XML application in biology science

Title	URL	Description
GAME	http://www.bioxml.org/Projects/game/	Genome Annotation Markup Language
BIOML	http://www.bioml.com/BIOML/	BIOpolymer Makeup Language
BSML	http://www.labbook.com/products/xmlbsml.asp	The Bioinformatic Sequence Markup Langauage
AGAVE	http://www.agavexml.org/	Architecture for Genomic Annotation, Visualization and Exchange
DAS	http://biodas.org/	Distributed Sequence Annotation System
ProML	http://cartan.gmd.de/promlweb/	Protein Markup Language
PROXIML	http://www.cse.ucsc.edu/%7Edouglas/proximl/	PROtein eXtensIble Markup Language
MAGE-ML	http://www.mged.org/index.html	MicroArray Markup Language
GO	http://www.geneontology.org/	Gene Ontology
CellML	http://www.cellml.org/	Cell Markup Language
CML	http://www.xml-cml.org/	Chemical Markup Language
SMBL	http://www.cds.caltech.edu/erato/sbml/docs/	The System Biology Markup Language
XEMBL	http://www.ebi.ac.uk/xembl/	XML project of EMBL
GIB	http://gib.genes.nig.ac.jp/	DDBJ Genome Information Broker
BIND	http://www.binddb.org/	The molecular Interaction Network

2 Web Services 服务的调用

2.1 构建平台的硬件和操作系统以及软件

本案例使用 Pentium 1.8G CPU，512M 内存，18G SCSI 硬盘。所用操作

强 XML 在生命科学中的应用，并使其规范化。而且 BioXML 采取开放源代码的政策，任何人都能参与开发和错误修订。

6）SBML(Hucka, et al., 2003)：the Systems Biology Markup Language(http://www.sbml.org/)

表 1　XML 在生命科学中的应用

Table 1　The XML application in biology science

Title	URL	Description
GAME	http://www.bioxml.org/Projects/game/	Genome Annotation Markup Language
BIOML	http://www.bioml.com/BIOML/	BIOpolymer Makeup Language
BSML	http://www.labbook.com/products/xmlbsml.asp	The Bioinformatic Sequence Markup Langauage
AGAVE	http://www.agavexml.org/	Architecture for Genomic Annotation, Visualization and Exchange
DAS	http://biodas.org/	Distributed Sequence Annotation System
ProML	http://cartan.gmd.de/proml/web/	Protein Markup Language
PROXIML	http://www.cse.ucsc.edu/%7Edouglas/proximl/	PROtein eXtensIble Markup Language
MAGE-ML	http://www.mged.org/index.html	MicroArray Markup Language
GO	http://www.geneontology.org/	Gene Ontology
CellML	http://www.cellml.org/	Cell Markup Language
CML	http://www.xml-cml.org/	Chemical Markup Language
SMBL	http://www.cds.caltech.edu/erato/sbml/docs/	The System Biology Markup Language
XEMBL	http://www.ebi.ac.uk/xembl/	XML project of EMBL
GIB	http://gib.genes.nig.ac.jp/	DDBJ Genome Information Broker
BIND	http://www.binddb.org/	The molecular Interaction Network

2 Web Services 服务的调用

2.1　构建平台的硬件和操作系统以及软件

本案例使用 Pentium 1.8G CPU，512M 内存，18G SCSI 硬盘。所用操作

系统为 RedHat Linux 8.0。我们分别使用 JAVA，Perl 语言来构建 Web Services 客户端。所用软件是 Jakarta Tomcat 4.1.24， Java (jasdk1.4.1.02), Apache Axis 1.1 以及 Apache 1.3.27。Apache Axis 是一个 SOAP 引擎，用来构建 SOAP 消息，并获取文档。Perl v5.8.0。

2.2 Web Services 提供商

本案例所有调用的服务均来自 DDBJ XML 中心(Sugawara et al., 2003)，该中心提供如下服务：

表 2 DDBJ Web 服务列表

Table 2 The list of DDBJ Web Services

Name	URL	Registrant
BLAST Demo	http://xml.nig.ac.jp/wsdl/BlastDemo.wsdl	XML Central of DDBJ
Blast	http://xml.nig.ac.jp/wsdl/Blast.wsdl	XML Central of DDBJ
ClustalW	http://xml.nig.ac.jp/wsdl/ClustalW.wsdl	XML Central of DDBJ
DDBJ	http://xml.nig.ac.jp/wsdl/DDBJ.wsdl	XML Central of DDBJ
ExClustalW	http://xml.nig.ac.jp/wsdl/ExClustalW.wsdl	XML Central of DDBJ
Fasta	http://xml.nig.ac.jp/wsdl/Fasta.wsdl	XML Central of DDBJ
GetEntry	http://xml.nig.ac.jp/wsdl/GetEntry.wsdl	XML Central of DDBJ
Gib	http://xml.nig.ac.jp/wsdl/Gib.wsdl	XML Central of DDBJ
Gtop	http://xml.nig.ac.jp/wsdl/Gtop.wsdl	XML Central of DDBJ
PML	http://xml.nig.ac.jp/wsdl/PML.wsdl	XML Central of DDBJ
SRS	http://xml.nig.ac.jp/wsdl/SRS.wsdl	XML Central of DDBJ
TxSearch	http://xml.nig.ac.jp/wsdl/TxSearch.wsdl	XML Central of DDBJ

2.3 调用 Web Services 服务获取 XML 格式的核酸数据

2.3.1 获取 XML 格式核酸数据的技术流程

XML Central of DDBJ 提供的 GetEntry 服务其中有一项功能让客户端获取 XML 格式的核酸数据，其方法为 getXML_DDBJEntry，该方法在 GetEntry.wsdl

系统为 RedHat Linux 8.0。我们分别使用 JAVA，Perl 语言来构建 Web Services 客户端。所用软件是 Jakarta Tomcat 4.1.24， Java (jasdk1.4.1.02), Apache Axis 1.1 以及 Apache 1.3.27。Apache Axis 是一个 SOAP 引擎，用来构建 SOAP 消息，并获取文档。Perl v5.8.0。

2.2 Web Services 提供商

本案例所有调用的服务均来自 DDBJ XML 中心(Sugawara et al., 2003)，该中心提供如下服务：

表 2 DDBJ Web 服务列表

Table 2 The list of DDBJ Web Services

Name	URL	Registrant
BLAST Demo	http://xml.nig.ac.jp/wsdl/BlastDemo.wsdl	XML Central of DDBJ
Blast	http://xml.nig.ac.jp/wsdl/Blast.wsdl	XML Central of DDBJ
ClustalW	http://xml.nig.ac.jp/wsdl/ClustalW.wsdl	XML Central of DDBJ
DDBJ	http://xml.nig.ac.jp/wsdl/DDBJ.wsdl	XML Central of DDBJ
ExClustalW	http://xml.nig.ac.jp/wsdl/ExClustalW.wsdl	XML Central of DDBJ
Fasta	http://xml.nig.ac.jp/wsdl/Fasta.wsdl	XML Central of DDBJ
GetEntry	http://xml.nig.ac.jp/wsdl/GetEntry.wsdl	XML Central of DDBJ
Gib	http://xml.nig.ac.jp/wsdl/Gib.wsdl	XML Central of DDBJ
Gtop	http://xml.nig.ac.jp/wsdl/Gtop.wsdl	XML Central of DDBJ
PML	http://xml.nig.ac.jp/wsdl/PML.wsdl	XML Central of DDBJ
SRS	http://xml.nig.ac.jp/wsdl/SRS.wsdl	XML Central of DDBJ
TxSearch	http://xml.nig.ac.jp/wsdl/TxSearch.wsdl	XML Central of DDBJ

2.3 调用 Web Services 服务获取 XML 格式的核酸数据

2.3.1 获取 XML 格式核酸数据的技术流程

XML Central of DDBJ 提供的 GetEntry 服务其中有一项功能让客户端获取 XML 格式的核酸数据，其方法为 getXML_DDBJEntry，该方法在 GetEntry.wsdl

服务文档中描述。客户端通过构建 SOAP 消息向 GetEntry 服务调用 getXML_DDBJEntry 方法。在这里，我们采用标准的开放式的 HTTP 协议。GetEntry 服务接收到消息后启动 getXML_DDBJEntry 方法（根据客户端提供的参数进行数据库查询），查询的结果同样以 HTTP 协议返回给客户端。
完整的技术和数据流程详见下图。

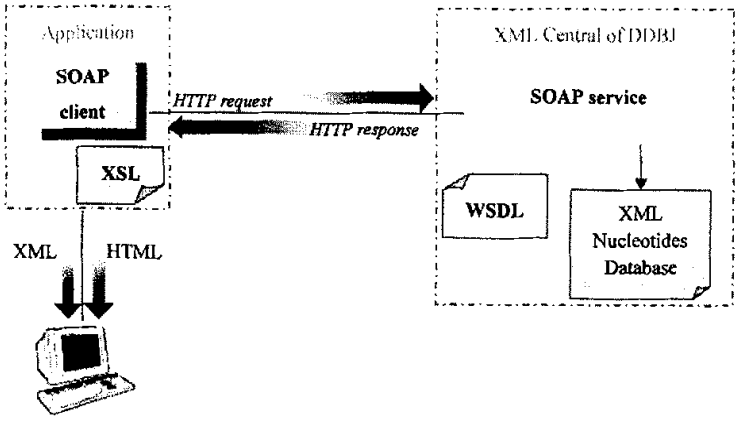

图 2 调用 Web Service 获取 XML 文档的工作流程

Figure 2 Discrete components and interactions in the web services architecture

2.3.2 了解 GetEntry 服务对象（GetEntry.wsdl）

要正确地应用服务，了解其服务对象是前提条件，在分布式计算中是要了解服务的接口与接口的实现。所谓分布式计算就是当调用一个远程对象的方法或向某个远程终端发送消息时，不论是否希望返回一个结果，在另一端发生的动作或情况是用户所不能了解与控制的。然而，在大多数情况下，之所以还会使用远程对象，是因为用户或称为客户更关心方法或消息的效果，而不关心它们是如何操作的。

WSDL 提供了一种语法，用于将服务描述成一组交换消息的端点。WSDL 文档充当一个或多个服务的（XML）描述，这种描述是与语言和平台无关的。

WSDL 文档描述了这些服务、如何访问它们以及期望的响应类型（如果有的话）。WSDL 描述了服务的类型、消息、操作、portType、位置和协议绑定。您使用 WSDL 将 Web 服务描述成一组对消息进行操作的端点。WSDL 可以描述面向文档的信息，也可以描述面向过程的信息。

portType：portType 描述 Web 服务所提供的操作。它类似于一个 Java 接口，因为它描述了一组操作。它将消息元素合并成操作。

message 和 types：message 是一个数据元素。操作用它来传送该操作的数据。消息通过列出所交换的数据类型来描述客户机和服务之间的通信。types 元素中描述了类型，通常用 XML Schema 完成描述。types 类似于 Java 类和基本类型。

operation、message 和 fault：operation 就像一个 Java 方法。它包含了传入消息、传出消息和出错消息。您可以将操作的传入消息当作是 Java 编程语言中方法的参数。可以将操作的传出消息当作 Java 编程语言中方法的返回类型。可以将出错消息当作 Java 异常。

binding：binding 将 portType 绑定到特定的协议（例如 SOAP 1.1、HTTP GET/POST 或 MIME）。

service：service 定义了某个特定绑定（binding）的连接信息。服务可以有一个或多个端口，每个端口都定义一个不同的连接方法（例如 HTTP / SMTP 等等）。

总而言之，一个 WSDL 协议实例描述五项内容：

types，Web 服务接口的数据类型（其参数和返回类型）。

message，将数据类型变量分组以进行网络传输。

portType，将消息分组成逻辑操作。

binding，描述如何将 portType 映射成传输 / 消息传递协议。

service，列出某个特定绑定的连接信息。

如下的 WSDL 文档是从 GetEntry.wsdl 中关于 getXML_DDBJEntry 方法的具体描述：

GetEntry.wsdl (http://xml.nig.ac.jp/wsdl/GetEntry.wsdl)

```
<?xml version='1.0' encoding='UTF-8'?>
```

```
<definitions name='GetEntry'
  targetNamespace='http://www.themindelectric.com/wsdl/GetEntry/'
  xmlns:tns='http://www.themindelectric.com/wsdl/GetEntry/'
  xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap/'
  xmlns:http='http://schemas.xmlsoap.org/wsdl/http/'
  xmlns:mime='http://schemas.xmlsoap.org/wsdl/mime/'
  xmlns:xsd='http://www.w3.org/2001/XMLSchema'
  xmlns:soapenc='http://schemas.xmlsoap.org/soap/encoding/'
  xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
  xmlns='http://schemas.xmlsoap.org/wsdl/'
  xmlns:tme='http://www.themindelectric.com/'>
 ......
<message name='getXML_DDBJEntryAsync51In'>
    <part name='accession' type='xsd:string'/>
</message>
<message name='getXML_DDBJEntryAsync51Out'>
    <part name='Result' type='xsd:string'/>
</message>
.......
<portType name='GetEntry'>
.......
<operation  name='getXML_DDBJEntryAsync' parameterOrder='accession'>
 <documentation>Get DDBJ entry of XML format by Accession
Number</documentation>
 <input  name='getXML_DDBJEntryAsync51In'
        message='tns:getXML_DDBJEntryAsync51In'/>
 <output  name='getXML_DDBJEntryAsync51Out'
        message='tns:getXML_DDBJEntryAsync51Out'/>
  </operation>
```

```
……..
</portType>
<binding name='GetEntry' type='tns:GetEntry'>
…….
<soap:binding style='rpc' transport='http://schemas.xmlsoap.org/soap/http'/>
…….
<operation name='getXML_DDBJEntry'>
 <soap:operation soapAction='getXML_DDBJEntry' style='rpc'/>
 <input name='getXML_DDBJEntry24In'>
    <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
               encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
 </input>
 <output name='getXML_DDBJEntry24Out'>
    <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
               encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
 </output>
</operation>
…….
<operation name='getXML_DDBJEntryAsync'>
   <soap:operation soapAction='getXML_DDBJEntryAsync' style='rpc'/>
   <input name='getXML_DDBJEntryAsync51In'>
    <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
               encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
   </input>
   <output name='getXML_DDBJEntryAsync51Out'>
    <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
               encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
   </output>
</operation>
```

```
……
</binding>
<service name='GetEntry'>
  <port name='GetEntry' binding='tns:GetEntry'>
    <soap:address location='http://xml.nig.ac.jp/xddbj/GetEntry'/>
  </port>
</service>
</definitions>
```

在上述代码中，首先将这个文档定义为兼容 XML1.0 规范。然后编写根元素，该元素包含所有必需的服务定义。所有的 WSDL 文档的根元素定义为 definitions。根元素的第一个属性是服务的通用名称，为了与文件名保持一致，将这个属性的值设为 GetEntry。第二个属性是 targetNamespace，该属性是可选的，用于指出一个标识该服务的绝对 URL 地址。接着定义多个名字空间。默认的名字空间是 WSDL 名字空间（否则，文档中所有元素，比如 definitions，都必须有前缀 wsdl 或任何与 WSDL 名字空间 URL 相关联的前缀）。

接下来定义操作所需的 SOAP 消息，包括输入消息和输出消息以及发送和输出消息所使用的数据类型（在这里，使用的是字符串类型,string），如下所示：

```
<message name='getXML_DDBJEntryAsync51In'>
    <part name='accession' type='xsd:string'/>
  </message>
  <message name='getXML_DDBJEntryAsync51Out'>
    <part name='Result' type='xsd:string'/>
  </message>
```

根据操作的命名规则，可以看出这些消息的方向是入（in）还是出（out），但是不一定非得这样命名。这样命名的好处是可以清楚地表达出所有不同的消息。然后将这些消息组合成操作，这些操作类似接口中定义的方法。这些操作组合成一个 portType，如下所示：

```
<portType name='GetEntry'>
 <operation   name='getXML_DDBJEntryAsync' parameterOrder='accession'>
```

```
  <documentation>Get DDBJ entry of XML format by Accession Number</documentation>
    <input  name='getXML_DDBJEntryAsync51In'
        message='tns:getXML_DDBJEntryAsync51In'/>
    <output  name='getXML_DDBJEntryAsync51Out'
        message='tns:getXML_DDBJEntryAsync51Out'/>
 </operation>
</portType>
```

上述所有代码只是抽象定义了整个服务。我们为已经定义的端口类型定义了可能有的操作，还定义了这些操作使用的消息，另外还定义了组成这些消息的各个部分和数据类型。

接下来还需要指定服务的真正的物理网络实现。我们这里希望通过 SOAP 使用服务，所以还要定义一个 SOAP 绑定。我们可以通过 HTTP 传输协议调用远程方法。SOAP 还定义了编码方式，所以还要为每个操作中的消息指定一种特殊的编码方式。默认情况下，使用标准的 SOAP 编码方式。

```
<binding name='GetEntry' type='tns:GetEntry'>
 <soap:binding style='rpc' transport='http://schemas.xmlsoap.org/soap/http'/>
  <operation name='getXML_DDBJEntry'>
   <soap:operation soapAction='getXML_DDBJEntry' style='rpc'/>
  <input name='getXML_DDBJEntry24In'>
   <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
          encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
 </input>
  <output name='getXML_DDBJEntry24Out'>
   <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
          encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
 </output>
</operation>
<operation name='getXML_DDBJEntryAsync'>
```

```
  <soap:operation soapAction='getXML_DDBJEntryAsync' style='rpc'/>
  <input name='getXML_DDBJEntryAsync51In'>
   <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
              encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
  </input>
  <output name='getXML_DDBJEntryAsync51Out'>
   <soap:body use='encoded' namespace='http://tempuri.org/GetEntry'
              encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
  </output>
</operation>
</binding>
```

首先，绑定的类型要与先前定义的端口类型名字匹配。当然，同一个端口类型可以有多种绑定：一种通过 HTTP 协议绑定，另一种通过 SMTP 协议绑定等等。在这里，soap:binding 元素表示这是一个 SOAP 和 HTTP 绑定。Style 属性的值可以为 document 或 rpc（远程过程调用）；默认值是 document。我们这里使用的是简单数据类型（而不是 XML 文档），而且这个服务更面向于过程而不是文档处理，故 style 属性的值是 rpc。然后与抽象端口类型中定义的每一个操作进行匹配（通过 name 属性）并且为每个操作指定 soap:operation。可以使用 soapAction 属性指定 HTTP 消息头 soapAction，在 SOAP 协议的某些实现中使用这个属性。每个消息都链接到一个 soap:body 元素。这个标记体元素能够指定消息的编码方式。可以使用 encode 并指定一种编码方式，也可以使用 literal 不指定任何编码方式。这里使用 encode 指定了编码方式。每个 soap:operation 的 soapAction 属性和每个 soap:body 的 namespace 属性将用做调用到某个服务器的服务（与实现一起调用）的名称。

至此，我们基本上解析了这个 WSDL 文档。该文档定义了 getXML_DDBJEntry 抽象端口类型，该端口类型包含多种接收和发送消息的操作，还定义了一个或多个到传输协议的绑定。这种类型的文档可用做一个通用的服务定义，在不同的服务器程序或服务代理程序上可重复利用这类文档，只是实现方法可能会有所不同，还可以在 WSDL 文档中通过指定具体的服务和

端口定义一个真实的网络地址以便客户端用户连接，代码如下：

```
<service name='GetEntry'>
  <port name='GetEntry' binding='tns:GetEntry'>
    <soap:address location='http://xml.nig.ac.jp/xddbj/GetEntry'/>
  </port>
</service>
```

从上述代码中可以看出端口通过 binding 属性连接到一个特殊的绑定上，http://xml.nig.ac.jp 是服务器的绝对域名地址。该服务的任何用户都可以将某个 SOAP 客户定向到真正的 URL 地址上。

2.3.3 编写 Web Services 客户端获取核酸数据

为实现对核酸数据的获得，我们分别使用 JAVA 和 Perl 语言来编写 Web Services 客户端。所有程序都只是在 RedHat Linux 8.0 实现。Jakarta Tomcat 4.1.24， Java (jasdk1.4.1.02), Apache Axis 1.1 以及 Apache 1.3.27 被用来构建 JAVA Web Services 客户端。Perl v5.8.0 和 SOAP::Lite Perl 应用程序接口（API）来构建 PERL 客户端。本案例所有程序完全使用开放源代码资源。

Java (· jasdk1.4.1.02, http://java.sun.com) ， Apache · Axis 1.1 (http://www.apache.org)，Apache 1.3.27 (http://www.apache.org)，Perl v5.8.0 为 RedHat Linux 8.0 自带 Perl 版本，SOAP::Lite (http://www.soaplite.com/)。

2.3.3.1 使用 JAVA 语言实现

在这里，我们使用 Apache Axis 1.1。Apache Axis 1.1 是一个 SOAP 引擎，Apache Axis 1.1 提供了一个简单的透明的 JAVA 库，是一个关于利用 JAVA 语言发送和接收 SOAP 消息的 JAVA API（JAVA 应用程序接口），既可以用来构建 SOAP 服务器，同时也提供了构建 SOAP 客户端所需的 JAVA API。

Apache Axis 1.1 的获得与安装详见 http://ws.apache.org/axis/。

在我们的案例中，使用黄孢原毛平革菌（*Phanerochaete chrysosporium*）的锰过氧化物酶基因 3(mnp3)的全长 CDS(coding sequence)，accession number:

U70998。作为调用实例。

客户端程序如下（DdbjXml.java）

```
1  import java.util.*;
2  import java.net.*;
3  import org.apache.axis.client.Call;
4  import org.apache.axis.client.Service;
5  import org.apache.axis.client.Call;
6  import org.apache.axis.client.Service;
7  import org.apache.axis.encoding.XMLType;
8  import org.apache.axis.utils.Options;
9  import javax.xml.rpc.ParameterMode;
10 import javax.xml.namespace.QName;
11 import java.net.URL;

12 public class DdbjXml {
13 public String getResult() throws Exception{

14      System.setProperty( "http.proxySet", "true" );
15      System.setProperty( "http.proxyHost", "202.84.17.41" );
16      System.setProperty( "http.proxyPort", "80" );

17      String wsdlURL = "http://xml.nig.ac.jp/wsdl/GetEntry.wsdl";
18     String nmspace = "http://www.themindelectric.com/wsdl/GetEntry/";
19      String srvname = "GetEntry";
20     String fncname = "getXML_DDBJEntry";
21     String query    =   "U70998";
22     QName serviceQN = new QName(nmspace, srvname);
23     QName portQN       = new QName(nmspace, srvname);
24     Service service = new Service(new URL(wsdlURL), serviceQN);
```

```
25      Call call = (Call)service.createCall(portQN, fncname);
26      String result = (String)call.invoke(new Object[]{query});
27      System.out.println(result);
    }
}
```

1~12 行是导入所需要的包，其中 3~8 是 Apache Axis 1.1 中的 JAVA 类。

24，25 行我们创建了一个新的服务（Service）和一个新的调用（Call），从面向对象的观点来看，我们创建的是两个新的对象。

14-16 行是防火墙设置，SOAP 消息可以穿透防火墙而直接传送。

26 行执行远程方法调用。

27 打印结果

发送的 SOAP 消息：

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
 <ns1: getXML_DDBJEntry xmlns:ns1="
        http://www.themindelectric.com/wsdl/GetEntry/">
   <arg0 xsi:type="xsd:string">U70998</arg0>
 </ns1: getXML_DDBJEntry >
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP（Simple Object Access Protocol）简单对象访问协议，是一种用于发送可扩展格式的信息的机制。可以使用 SOAP 发送消息，也可以发送用 XML 格式编码的远程过程调用。SOAP 消息分成三个基本组件：

1：SOAP Envelop：用于定义消息中的内容、处理它的处理方式以及它的性质（是可选的或是强制的）。

2：SOAP 编码规则：这些规则向用户提供一种机制，用于交换应用程序定

义的数据类型。

3：SOAP RPC 表达形式：用于执行远程过程调用和响应。

SOAP 消息编码为 XML 文档，该文档由一个强制的 SOAP 封皮、一个可选的 SOAP 消息头以及一个强制的 SOAP 消息体组成。

1：SOAP 封皮

<SOAP-ENV:Envelope>

</SOAP-ENV:Envelope>

SOAP 封皮是任何 SOAP 消息都必需的组件。SOAP 封皮元素必须作为一个 SOAP 消息中的第一个元素出现。SOAP 封皮提供有关消息的内容，以及在某些情况下由“谁”来处理消息。关于其它元素（比如<SOAP-ENV:Body>和<SOAP-ENV:Header>）如何必须出现在 SOAP 消息中，封皮还规定了一些规则。在封皮上还可以有附加的属性，只要这些元素符合名字空间标准就可以。同样，如果有附加子元素，就可以添加到封皮上，只要子元素符合名字空间标准和在<SOAP-ENV:Body>元素之后即可。

2：SOAP 消息头

一个 SOAP 消息头是可选的，本例中未使用消息头。如果要使用，则必须保证消息头紧跟 SOAP 封皮声明之后，并在 SOAP 消息体的前面。

3：SOAP 消息体（Body 元素）

```
<SOAP-ENV:Body>
 <ns1: getXML_DDBJEntry xmlns:ns1="
       http://www.themindelectric.com/wsdl/GetEntry/">
   <arg0 xsi:type="xsd:string">U70998</arg0>
 </ns1: getXML_DDBJEntry >
</SOAP-ENV:Body>
```

2.3.3.2 使用 Perl 语言实现

本案例使用 Perl v5.8.0，SOAP::Lite 模块。SOAP::Lite 是一个使用 Perl 语言发送和接收 SOAP 消息的 Perl 模块集，一个关于使用 Perl 语言处理 SOAP 消息的 API（应用程序接口）。下载和安装方式见：http://www.soaplite.com。

以下代码用于实现获取 XML 核酸数据：

```
/usr/bin/perl –w
use SOAP::Lite;
my $service = SOAP::Lite -> service('http://xml.nig.ac.jp/' .'wsdl/GetEntry.wsdl');
$result = $service->getXML_DDBJEntry("U70998");
print $result;
```

3 结果与分析

分别应用上述的 Java 和 Perl 语言书写的代码，一份 XML 格式的核酸数据被获得，accession number（GenBank 序列号）：U70998。同时，我们构建了一个中间平台，使用户能在我们的网站上获得 XML 格式的核酸数据。我们使用 Jakarta Tomcat 4.1.24，Apache 1.3.27 来构建。查询页面如图：

图 3 查询页面 (http://scbi.scu.edu.cn/webservices/)

Figure 3 The web page for retrieving nucleotides data (http://scbi.scu.edu.cn/webservices/)

以下代码用于实现获取 XML 核酸数据：

```
/usr/bin/perl –w
use SOAP::Lite;
my $service = SOAP::Lite -> service('http://xml.nig.ac.jp/' .'wsdl/GetEntry.wsdl');
$result = $service->getXML_DDBJEntry("U70998");
print $result;
```

3 结果与分析

分别应用上述的 Java 和 Perl 语言书写的代码，一份 XML 格式的核酸数据被获得，accession number（GenBank 序列号）：U70998。同时，我们构建了一个中间平台，使用户能在我们的网站上获得 XML 格式的核酸数据。我们使用 Jakarta Tomcat 4.1.24，Apache 1.3.27 来构建。查询页面如图：

图 3 查询页面 (http://scbi.scu.edu.cn/webservices/)

Figure 3 The web page for retrieving nucleotides data (http://scbi.scu.edu.cn/webservices/)

结果页面显示如下：

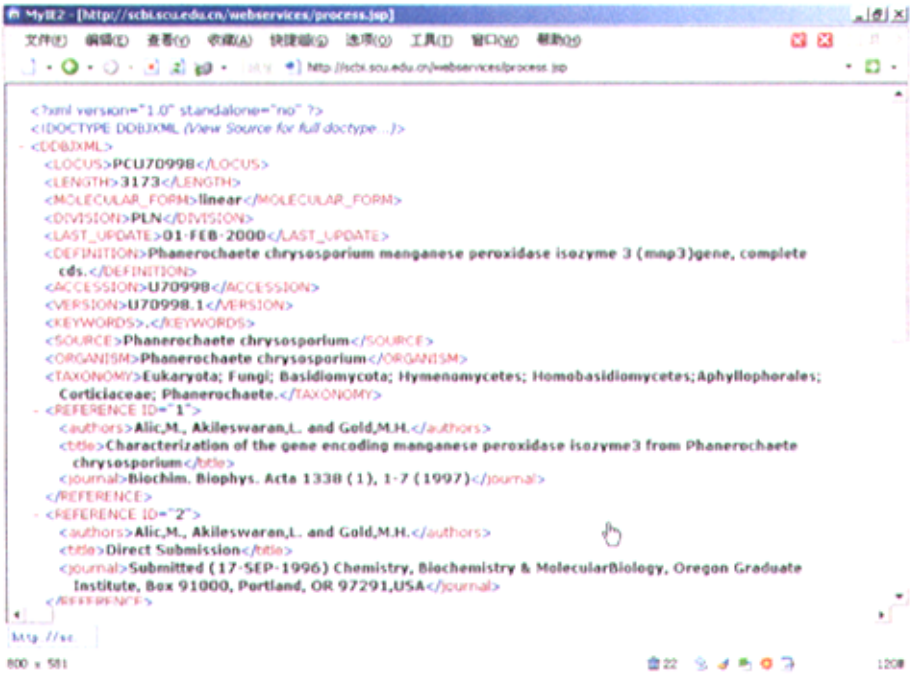

图 4　结果显示页面

Figure 4　The result web page.

完整的文档如下（U70998.xml）：

```
<?xml version="1.0" standalone="no"?>
<DDBJXML>
<LOCUS>PCU70998</LOCUS>
<LENGTH>3173</LENGTH>
<MOLECULAR_FORM>linear</MOLECULAR_FORM>
<DIVISION>PLN</DIVISION>
<LAST_UPDATE>01-FEB-2000</LAST_UPDATE>
```

<DEFINITION>Phanerochaete chrysosporium manganese peroxidase isozyme 3 (mnp3)gene, complete cds.</DEFINITION>
<ACCESSION>U70998</ACCESSION>
<VERSION>U70998.1</VERSION>
<KEYWORDS>.</KEYWORDS>

<SOURCE>Phanerochaete chrysosporium</SOURCE>
<ORGANISM>Phanerochaete chrysosporium</ORGANISM>
<TAXONOMY>Eukaryota; Fungi; Basidiomycota; Hymenomycetes; Homobasidiomycetes;Aphyllophorales; Corticiaceae; Phanerochaete.</TAXONOMY>

<REFERENCE ID="1">
<authors>Alic,M., Akileswaran,L. and Gold,M.H.</authors>
<title>Characterization of the gene encoding manganese peroxidase isozyme3 from Phanerochaete chrysosporium</title>
<journal>Biochim. Biophys. Acta 1338 (1), 1-7 (1997)</journal>
</REFERENCE>

<REFERENCE ID="2">
<authors>Alic,M., Akileswaran,L. and Gold,M.H.</authors>
<title>Direct Submission</title>
<journal>Submitted (17-SEP-1996) Chemistry, Biochemistry & MolecularBiology, Oregon Graduate Institute, Box 91000, Portland, OR 97291,USA</journal>
</REFERENCE>

<FEATURES>

<source>
<location>1..3173</location>
<qualifiers name="organism">Phanerochaete chrysosporium</qualifiers>
<qualifiers name="mol_type">genomic DNA</qualifiers>
<qualifiers name="strain">OGC101</qualifiers>
<qualifiers name="db_xref">taxon:5306</qualifiers>
</source>
<gene>
<location>1193..2671</location>
<qualifiers name="gene">mnp3</qualifiers>
</gene>
<cds>
<location>join(1193..1324,1376..1454,1506..1545,1607..1784,1843..2064,
2115..2529,2589..2671)</location>
<qualifiers name="gene">mnp3</qualifiers>
<qualifiers name="codon_start">1</qualifiers>
<qualifiers name="product">manganese peroxidase isozyme 3</qualifiers>
<qualifiers name="protein_id">AAB39652.1</qualifiers>
<qualifiers name="db_xref">GI:1763109</qualifiers>
<qualifiers name="translation">MAFASSLLALVALAAVTSAAPATTQATC
PDGTKVNNAACCAFIPLAQDLQETIFQNDCGEDAHEVIRLTFHDAIAIS
QSKGPSAGGGADGSMLLFPTIEPNFSANNGIDDSVNNLIPFMQKHDTS
AGDIVQFAGAVALTNCPGAPQLEFLAGRPNKTIPAIDGLIPEPQDSVTSL
ERFKDAGNFSPFEVVSLLASHSVARADKVDETIDAAPFDTTPFVFDTQI
FLEVLLKGVGFPGTANNTGEVASPLPLTSGSDTGELRLQSDFALARDE
TACIWQGFVNEQALMAASFKAAMAKLAVLGHDRNTLVDCSDVVPAK
PAVNKPASFPATTGPQDLELSCNTKPFPSLSVDAGAQQTLIPHCSDGDT
CQSVQFNGPA</qualifiers>
</cds>

</FEATURES>

<BASE_COUNT A="652" C="966" G="841" T="714"/>

<SEQUENCE>gatcccgggtttctcaggctaaaaatggagccccaaccacaggcgtcccaatttgatgagaagc
ggggacttcctaccttctggcattgacaaaaataagcatgtgagggaatggatggcaagctgcctcgttcattcacaccg
tggagtcggttctccggaggctgttgccaaatggaccaacaaatcctactttctttcccttgctagcagcgcagcacatcc
agaagccgcgtttagtcacttttttgctgatgagacggggtaaaatgggaacgtcgaggtgcgactcctgtgaagtaccc
gctgcgcttcaggaatgacacctcacagttggactcagaacgactccggaggtgtgagggttgaaggtatgttcccatg
cactcttcttagcgtgtgtcatgtcgttgactggcacggacccagggtgaacgtctaggacgatcctcaagacagggac
atccatagagcggcagtaattgtgatgctaaaggctgcccagcgcgacggcgcgtcatctattacatgtacagtgggca
cagttaaatccaggtaatcacatcaaaccgaaagtcctagtgattttctgaacacatgccgctagatcgcgttgccctaca
acagccgcggccttgggtatggtctgcctgccaagctcccactgccggccgcgatttgcaactgcaactcatccgacgc
cgctctgccgtcttggcgaccgagtatttttctctcatacccaccacgagtccgcgtgcggcggcgcaaaacgaggcac
tccgccgacgagcccgttgcgcctcctcacgctcctgtgagcgctgcaaaccggcaaaccgcgattggcacccgctc
gccgctgaagttgtgatacgcctcggttctgtgggcgcacaacaataccgcgccgatgggccatggctactgcaggca
gtcagattagcaggatggcgttgacggcatcgagctcgccacccacagtcctgtgagctgggccgcgcgatgtaccca
taacttcggtactatgcatcacctcatccagctgcatttctgatatcctggggtataaaagctcgacaatggcagcggtaga
gtgctcaggacaacgagctctcgcctcgcacctcttctcctgacaactcgtcaagccgcctagttacttgcgaccgaccg
cactcaagccagcgcaatggcctttgcatcctcactcctcgctctggtggctcttgccgcagtcacatccgctgcgccgg
ccacgacccaggccacttgccccgacggtaccaaggtcaacaacgctgcctgctgcgctttcatacctgtaagcggcct
ttcactcgtcgaaaatgatctttacttatcatttctacagcttgcacaggatcttcaggagactatcttccagaacgactgcg
gtgaagacgcacatgaagttattcgccttactttccgtaagccgaacgcaaattgcgcaaatgccatgctaatcttttccctt
gtagacgacgctattgccatttcgcaaagcaagggaccgagcgcgtaagcttatgcccgatcttgcgaggcatttcgag
ttctgactggggtttcccttgaccagtggcggcggagctgacggctccatgctgctgttccccaccattgagcccaacttc
tctgccaacaacggtatcgatgactcggtcaacaatctcatcccgttcatgcaaaagcacgacaccatcagcgctggcg
acatcgtccagttcgccggcgccgtcgcgctcaccaactgtcctgtacgtccctctcactccgacttcatcggcgtaagtc
tgacgaatgatatgtatgtagggtgcgccgcagctcgagttccttgcgggacgcccgaataagacaatccctgccattg
atggcctcattcccgagccgcaggacagcgtgacgtcgatcctggagcgcttcaaggacgcaggcaacttcagcccgt
ttgaggtggtctcgctgctcgcgtcccactccgttgcgcgcgcagacaaggtagacgagaccatcgatgccgcgccgt

tcgatacggtgagcgcatcccgcatacagccgctatgcgtgcgctgactctcgcgcagacgccgttcgtgttcgacacg
cagatcttcctcgaggtactcctcaagggcgtcggcttcccggggacggcgaacaacacgggtgaggtcgcgtcgcc
gctgccgctgacgtcgggcagcgacacgggcgagctgcggctgcagtccgactttgcgctcgcgcgcgacgagcg
gacggcgtgcatctggcagggcttcgtcaacgagcaggcgctcatggccgcgagcttcaaggccgccatggcgaag
ctcgccgtgctcgggcacgaccgcaacacgctcgtcgactgcagcgacgtcgtgcccgcgccgaagcccgccgtga
acaagcccgcgagcttccccgccaccacgggcccccaggacctcgagctctcgtgcaacacgaagccgttcccgtcg
ctctctgttgatggtgtgtctgacctttctctggtggtgttgtcggccggctgatgggttttgcttttccagcgggcgcgcag
cagacgcttatcccgcactgctccgacggcgacatgacgtgccagagcgtccagttcaacggccctgcataagtctga
cgcgccggatatttatggtcaggtagagctgggaataccaggcatggtactatcgctttccgcaaattcgttgatgactat
ctgtttgtccttaccgtatcttgactgctccacctatcgatatctcgctgcaggctgtgtgccttaggggatatgatcgcgcat
tgcattgcatattctctccatagctgctgcagcttatatcggagatgactgatacaatcacatcttgtgtgagagacccgcc
ccgcgagaatgatgatagtaaagtctgcagcagccgttcaagtcctttcagggaattgttcgctgagctccttcgcttgca
agctattcttgatagcatttatccacttagccaaatgatcatgttttatttccgtgttagctatgtgcaattagagtgcgacaag
ggttcaacgttcagcgtccaacgttcaacgttgaacgttcagcgccctgcgcacagacaacaagtctgcagttctgcaca
agt</SEQUENCE>
</DDBJXML>

该文档的第 1 行（<?xml version="1.0" standalone="no"?>）是 XML 声明。声明的作用是表明该文档是 XML 文档。而且，引用这一行作为处理指令。该处理指令用于向 XML 解析器说明该 XML 文档符合 XML 规范的 1.0 版本。通过使用 version 及其值 1.0 进行说明。

该文档的第 2 行是一个名为<DDBJXML>的元素。该元素是"根"元素。所有 XML 文档都必须从"根"元素开始。它的用途是包含文档中所有其他的元素。例如，在 HTML 文档中，元素<HTML>是根元素。"根"元素定义 XML 文档的开始。即使 XML 声明是第一行，也不能表示文档的开始。

文档其他行是子元素。这些元素定义文档的数据块。每个元素代表一种惟一的数据块。最后一行是"根"元素的封闭标记。它表示文档的结束。所有的元素都必须封闭，特别是"根"元素 </DDBJXML>。

从上例可以看出，XML 是灵活的，允许自定义标记。要表示一段序列，就可以自定义标记，如< sequence>，就可以表示该元素是代表序列的。这样，

当解析器或其他应用程序处理该 XML 文档时，当碰到<sequence>标记，它就会明白接下来的是序列，然后就可以开始读取序列并进一步处理数据。而 HTML 语言因为不允许自定义标记，便失去了灵活性。XML 是一种定义严谨的标记语言，它不允许定义不明确的语法结构。例如：标记必须成对，而且不允许重叠出现。因为 XML 要求语法结构必须定义明确，所以不会产生模糊不清的状况。正因为 HTML 缺乏可扩展性的缺点，它已逐渐无法满足 WWW 的应用。现在各大计算机厂商都在期待一个新一代标准能够赶快出现，虽然 XML 满足了这种需求，但如果无法在短时间内指定并通过认证，远水将救不了近火。就目前而言，虽然 XML1.0 版已经问世，但就实用的层面而言，仍然还需要做很多工作。

上述的 XML 文档由于遵循 XML 规范中规定的一些简单的原则，被称之为格式良好的 XML 文档。实际上，该 XML 同时又是有效的 XML 文档。所谓有效的 XML 文档必须遵循 DTD（文档类型定义）中指定的约束条件。为定义某个特殊的 XML 文档中允许的内容，DTD 提供了一种机制。DDBJXML.dtd 文件被使用来约束上述文档。

完整的 DDBJXML.dtd 可以从我们网站上下载：（http://scbi.scu.edu.cn/XML/DDBJXML.dtd）。

XML 文档的解析（parse）

解析 XML 是应用 XML 中关键的一环。目前已经有多种语言的 XML 解析器（parser）和编程语言接口（API）。包括 java、C++和 Perl 等。它们都遵循 SAX 和 DOM 标准。应用人员可以快速高效地创建、遍历和修改 XML 文档的内容。XML 被使用去描述生物学数据的一个优势就是 XML 能被不同的平台、语言解析。

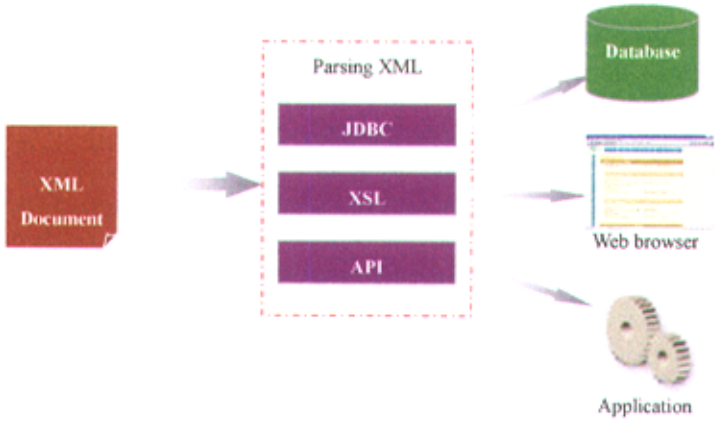

图 5　XML 文档的解析及其应用

Figure 5　Java DataBase Connectivity (JDBC) is the middleware that allows Java programs to access data from a relational database. JDBC provides a standard SQL database access interface.

在这里，我们使用 JDOM（Java-based Document Object model）作为 JAVA API 来处理、解析 XML 文档。JDOM 提供了一种简单地、高效地操纵 XML 的手段，它所包含的 API 更加直接、高效和优化。它完全兼容 SAX、DOM，又优于他们。JDOM 的源代码公开于 2000 年 4 月，最新版是 JDOM beta 10 (http://www.jdom.org)。

JDOM 使用标准的 Java 编码模式。只要有可能，它使用 Java new 操作符而不用复杂的工厂化模式，使对象操作即便对于初学用户也很方便。

用 JDOM 解析 XML 文档需要 3 个步骤：（1）建立 XML 数据的 JDOM 文档对象。（2）遍历、修改内存中的 JDOM 对象。（3）输出处理结果。

完整的程序代码详见我们的网站。

4　讨论

利用最新的计算机和信息技术来处理生物学数据，是生物学，特别是基因组学发展的一个必然结果，也是生物信息学专家们的任务之一。到现在，在生

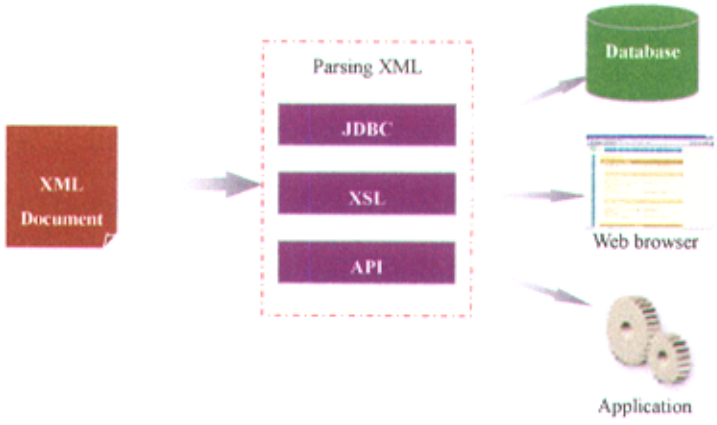

图 5　XML 文档的解析及其应用

Figure 5　Java DataBase Connectivity (JDBC) is the middleware that allows Java programs to access data from a relational database. JDBC provides a standard SQL database access interface.

在这里，我们使用 JDOM（Java-based Document Object model）作为 JAVA API 来处理、解析 XML 文档。JDOM 提供了一种简单地、高效地操纵 XML 的手段，它所包含的 API 更加直接、高效和优化。它完全兼容 SAX、DOM，又优于他们。JDOM 的源代码公开于 2000 年 4 月，最新版是 JDOM beta 10 (http://www.jdom.org)。

JDOM 使用标准的 Java 编码模式。只要有可能，它使用 Java new 操作符而不用复杂的工厂化模式，使对象操作即便对于初学用户也很方便。

用 JDOM 解析 XML 文档需要 3 个步骤：（1）建立 XML 数据的 JDOM 文档对象。（2）遍历、修改内存中的 JDOM 对象。（3）输出处理结果。

完整的程序代码详见我们的网站。

4　讨论

利用最新的计算机和信息技术来处理生物学数据，是生物学，特别是基因组学发展的一个必然结果，也是生物信息学专家们的任务之一。到现在，在生

物学中使用计算机和信息技术，还只能是一个被动的过程。然而，即使是在生物学的领域被动地接受计算机的技术也是十分困难的。主要原因在于两个领域（计算机和生物学）都处在飞速发展的阶段，也可以说是当今科学中最活跃的两个领域。因此，要同时掌握两个领域的最前沿，几乎是一个不可能完成的任务。

XML 和 Web Services 是信息技术中最为前沿的领域之一，被誉为下一代的 Internet 可见所重视的程度。XML 和 Web Services 是在电子商务发展过程中，更准确的说是商业中而发展起来的，它们的主要目的是为了解决在电子商务交换和整合数据。

然而，XML 和 Web Services 一出现，就被一些眼光敏锐的生物信息学家们重视。因为这两个技术的一些独特的优点正可以被用来解决生物学中，特别是基因组学中数据交换和整合的问题。生物学中用来记录数据（如序列等）的格式很多，不同的团体在发布自己的数据时在没有一致的标准下纷纷采用自己的格式，格式的不一致使得用计算机来处理这些数据，特别是整合来自各个不同团体发布的数据显得十分困难。XML 的出现给了我们这样一个契机：统一生物学中的数据格式。事实上，国际上一些生物信息学家们正致力于这样的工作。

准确地说，Web Services 更是一个概念，而不是一门专门的计算机或信息技术。国际上的一些大的公司是这个领域的领跑者，如 IBM，Mircosoft，SUN 等等正致力于 Web Services 的发展。虽然关于在分布式的环境中调用远程服务在前些年已经开始了研究，并且获得了一些成果，但是 Web Services 的出现最为让人激动，它使得构建在服务器端构建远程调用服务变得简单，更使得在客户端调用远程服务变得简单。同时，也使得用户有方便的途径去寻找自己需要的服务。这样的一些优点使得在可预见的未来组成一个全球化的结构严谨同时又服务丰富的生物信息服务网络成为可能。

在这一章里，由于能力和时间的关系我们只是很粗浅地描述并论证了构建 Web Services 的客户端，用来获取并整合 XML 格式的核酸数据。要完整地构建一个 Web Services 体系（服务器端）远比捕获数据困难得多。但是，构建客户端在整合不同来源的生物数据中又十分重要，因为整合数据的前提是能快速

地捕获数据，并能方便地解析数据。从这个意义上说，仅仅是构建 Web Services 的客户端在整合生物学数据中也是十分重要的。

参考文献

Archard F, Vaysseix G, Barillot E. XML, bioinformatics and data integration. Bioinformatics Review 17: 115-25, 2001.

Benini AA, Conley CE, Shdeed R, Spurway K, Yarmoshuk M. Integration of different data bodies for humanitarian decision support: an example from mine action. Disasters 27: 288-304, 2003.

De K, Guo Y, Li J, Kwan A, Yip K, Cheung D, Cheung K. A web services choreography scenario for interoperating bioinformatics applications. BMC Bioinformatics 10: 25, 2004.

Fenyo D. The Biopolymer Markup Language. Bioinformatics 15: 339-40, 1999.

Gene Ontology Consortium. Gene ontology: tool for the unification of biology. Nature Genet 25: 25-9, 2000.

Gilmour R. Taxonomic markup language: applying XML to systematic data. Bioinformatics 10: 406-7, 2000.

Goesmann A, Linke B, Rupp O, Krause L, Bartels D, Dondrup M, McHardy A, Wilke A, Puhler A, Meyer F. Building a BRIDGE for the integration of heterogeneous data from functional genomics into a platform for systems biology. J Biotechnol 19: 157-67, 2003.

Haas LM. Data integration through database federation. IBM Systems Journal 41: 578-96, 2002.

Kitano H. Systems Biology: A Brief Overview. Science 295: 1662-4, 2002.

Lacroix Z. Biological Data Integration: Wrapping Data and Tools, IEEE Trans Inf Technol Biomed. 2002, 6(2): 123-8.

Lichun W, Jean-Jack R, Alan R. XEMBL: distributing EMBL data in XML format. Bioinformatics 2002: 1147-8, 2002.

M H, A F, H MS, H B. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. Bioinformatics 19: 524-31, 2003.

Searls, DB. Data integration--connecting the dots. Nat Biotechnol. 2003 Aug;21(8):844-5.

Serge A. On views and XML. ACM SIGMOD Record 28: 30-8, 1999.

Siepel AC, A.N.Tolopko. An integration platform for heterogeneous bioinformatics components. IBM Systems Journal 40: 570-91, 2001.

Stein L. Integrating biological databases. Nat Rev Genet 4: 337-45, 2003.

Stein L. Creating a bioinformatics nation. Nature 417: 119-20, 2003.

Sugawara H, Miyazaki S. Biological SOAP servers and web services provided by the public sequence data bank. Nucleic Acids Res 31: 3836-9, 2003.

第二部分　Linux 平台下 EST 序列分析系统的构建

摘　要

利用抑制削减杂交和基因芯片技术获得一批黄孢原毛平革菌特异表达的 EST 序列，使用 Phrap、EMBOSS、Blast、GENSCAN、MZEF 软件，基于 Linux 操作系统，构建 EST 序列分析系统，完成了从 EST 和基因组 Blast 数据库的构建，载体序列的去除， EST 序列的分类和组装，EST 序列在基因组上的定位，外显子和内含子的识别以及基因预测。并通过使用 perl 语言结合 bioperl 模块写的脚本程序使分析过程自动化，从而可以快速地对大批 EST 序列进行分析，为克隆相关基因及研究黄孢原毛平革菌功能基因组学提供有用的信息。

关键词　黄孢原毛平革菌，EST 序列分析，生物信息学，基因预测，Linux 操作系统，bioperl

1 引言

黄孢原毛平革菌（*Phanerochaete chrysosporium*）是研究木质素降解的一种十分重要的模式真菌(Cullen, 1997)。目前，该菌用于降解木质素的一些基因已经被克隆，但是研究该菌在降解木质素过程中差异表达的基因还没有相关的报道。本实验室利用高灵敏性的抑制削减杂交和高通量性的基因芯片技术对此进行了研究，共得到了一批该菌具有 2d 和 3d 差异表达的序列标签（Expressed Sequence Tags, EST）。抑制消减杂交（suppression subtractive hybridization, SSH）是一种强有力的研究工具，帮助研究者对两个 mRNA 群进行比较并获取在两个群中差异表达的基因，特别适合微量表达的 mRNA(Diatchenko et al., 1996)。而基因芯片可用于分析基因表达的特征，上千个基因可通过一次杂交得到检测(Schena et al., 1995; Schena et al., 1996; DeRisi et al., 1997)。将二者结合，对基因表达调控的研究有着潜在的巨大优势(Yang et al., 1999)。首先，SSH 芯片能通过一次杂交检测上千个未知基因；其次，差异表达的基因可以通过 SSH 得到选择性富集；第 3，对芯片进行的计算机扫描为两个 RNA 群的比较提供了量化的相对丰度值；第 4，通过对芯片数据分析后发现的差异表达 cDNA 进行

测序可以一次获得大量特异性 EST 序列，为进一步研究提供了丰富的材料。

EST 对鉴定基因结构很有用，特别当它们与目标序列来自同一物种时。将 EST 序列相互之间及对基因组序列进行比较分析能为发现基因提供重要信息。对来自不同 cDNA 文库的 EST 进行比较还能提供基因表达和 mRNA 选择性剪切的线索。而且 EST 可用作鉴定基因的标签在基因组中探测匹配区域，用于构建基因组图谱。本实验室自 1985 年张义正教授在美国密歇根州立大学攻读博士学位时克隆到数个 *lip* 基因以来(Zhang et al., 1986; De et al., 1986; Zhang et al., 1988)，在木质素降解菌分子生物学的研究方面做了不少工作。最近几年，本实验室研究人员在这方面取得了新的进展。冯红博士在 1999 年通过 DNA footprinting 在 *P. chrysosporium* 木质素过氧化物酶启动子上游鉴定到两个调控序列：PBE1、PBE2，研究发现 PBE1 与人类和果蝇的 3 条基因组序列中的一段 17bp 的核苷酸序列完全同源，而 PBE2 序列则与拟南芥的一条基因组序列中的一段 20bp 的序列完全同源，虽然其功能意义尚不清楚，但仍为木质素过氧化物酶基因转录调控机理的研究提供了重要线索。

近年来，通过采用计算机和信息技术，大批量的 EST 序列分析有了快速的发展。基于 EST 和 UniGene 数据库的电子延伸技术(张成岗等，2001；张成岗等，2003)是一种获取全长 cDNA 的方法。但该方法的应用有相当的局限性，原因在于除了人和小鼠的 EST 数据较多之外，其他物种 EST 数量较少，大部分的物种没有相应的 EST 数据库。例如本实验所研究的黄孢原毛平革菌的 EST 序列在公共核酸数据库 GeneBank 中只有几十条。因此，采用已有的 EST 序列延伸获取全长 cDNA 的方法是不可行的。但采用此方法，能够在大批的 EST 中尽可能地获取长的 EST 序列，因此，也是本系统构建的步骤之一。

随着人的基因组计划的不断深入，许多物种的基因组计划也已经完成或者接近完成，还有许多物种基因组测序也正在准备中。目前，基因组已经测完并公布的物种共 140 种，正在进行中的有 577 种（详细信息请见：http://wit.integratedgenomics.com/GOLD/）。全基因组序列的解读，并不能使人类对编码基因这一层次有更明确的认识。因此，cDNA 测序计划就成为人们了解编码基因结构与功能的关键所在。通常，人们极其关注不同模型处理下同种组织中基因表达的差异，进而关注这些差异表达的基因与相应生物表型之间的

关联。能否发现真正的差异表达基因，一个共同的问题就是所采用的数据分析技术及其分析能力能否有效、快速地揭示大量序列数据中所蕴含的生物信息。

通过比对 EST 与基因组序列，可以找出该 EST 在基因组中的位置，确定其外显子和内含子的区域和边界，并通过各种基因预测软件对相应的基因组片段进行基因预测，对照 EST 在基因组中外显子和内含子的结构，从而能够尽可能找出正确的新基因，为克隆该基因提供有用的信息。本文拟就实验室所获得的 EST 序列的分析过程为例，介绍在 Linux 操作系统下，借助各种软件构建大规模的 EST 自动分析系统，并结合具体实例，证明本系统在分析大批量 EST 序列中的应用价值。

通常，所测定的差异基因片段往往只对应于 cDNA 序列的一段。GenBank 的 nr 数据库中所收录的 cDNA 序列也有相当一部分不是全长 cDNA 序列。因此，随着序列分析的深入进行，用户将发现全长 cDNA 序列对于进行后续研究是十分重要的常规方法是或通过实验研究获得全长 cDNA 序列，或直接基于 EST / cDNA 数据库进行电子序列延伸。如果尚未构建本地化的电子序列延伸系统，就必须依赖于网络资源进行分析。但这些资源一般只对注册用户提供服务。另一个办法是可考虑采用基于 UniGene 数据库进行电子序列延伸。新版本的 Blast 软件已增加了查询 UniDene 数据库的功能，并可下载该 UniGene 记录所对应的全部 mRNA / cDNA 序列和 EST 序列随后，用户可将所下载的全部序列用 Sequenche™ 软件进行拼接分析，得到较长的 contig 序列。这样就实现了对所分析序列片段的一轮电子序列延伸分析。该过程可不断重复，直至无法获得更长的序列，从而可获得较长的对应于用户原始序列的 cDNA 序列。用户可据此设计引物物并采用 RT－PCR 技术对其进行鉴定。不过，由于此过程要求用户对计算机操作。网络文件传输等比较熟悉，而且该过程较为耗时，一般不建议用户进行此种操作。倘若有规模较大的核酸序列电子拼接的需求，建议采用自行构建相关数据分析平台的策略进行。

2 EST 序列分析系统的构建

2.1 硬件配置与 Linux 操作系统的安装

我们使用的操作系统为 RedHat Linux 8.0，该操作系统可以通过网址：

http://www.redhat.com 或其他的镜像站点获取。所使用的硬件配置为：Pentium Ⅳ Intel CPU 1.8GHz，内存 256M，18GB SCSI 硬盘。这样的小型服务器花费少，大部分实验室都有能力配置。

2.2 数据的获取与预处理

2.2.1 EST 数据

本文在江明峰博士的指导下，结合 SSH 和 cDNA 芯片技术对限氮条件下黄孢原毛平革菌 2d 及 3d 培养物的基因表达谱进行了系统研究。分别构建了含 1556 个 3d 克隆和含 768 个 2d 克隆的 2 个 cDNA 文库，并用文库中所有 cDNA 片段制备了 cDNA 芯片。芯片用两 2 探针杂交：来自 SSH 的消减探针（SSH 探针）和由 mRNA 准备的反转录探针（RT 探针）。几乎所有片段均可与 SSH 探针杂交，但由 39.58%的 cDNA 片段不能与 RT 探针杂交。本文对结果进行了详细分析。基于 3/2d 杂交比筛选的 515 个克隆进行了测序。所有的序列按照 FASTA 格式(张成岗等，2002)存于平面文档（flatfile）中并保存该文档名为 my_db。

2.2.2 基因组数据

黄孢原毛平革菌基因组由 JGI（Joint Genome Institute，隶属美国能源部和加州大学）测序已基本完成，并于 2002 年 5 月第一次公布。可以通过下列网址 ftp://ftp.jgi-psf.org/pub/JGI_data/WhiteRot/ 获取。所获取的数据格式为 FASTA。

2.3 相关程序的获取与安装

本系统所有软件程序均为免费，可以通过网络下载或直接写邮件至作者获取。所采用的核心程序和软件包有：Blast 程序，由 NCBI 提供，用于本地化序列相似性分析(Altschul et al., 1997)；est2genome 程序，位于 EMBOSS（The European Molecular Biology Open Software Suite）软件包(Rice et al., 2000)，功能是使 EST 序列比对基因组序列，确定该 EST 在基因组上的位置及相应的外显子和内含子区域和边界；Phrap 软件包，为载体序列去除、EST 序列组装

(Ewing et al., 1998)，由 UWGC（University of Washington Genome Center）提供；GENSACN，基因预测软件(Burge et al., 1998)；MZEF，基因预测软件[17]；Clustalw 和 T-COFFEE(Notredame et al., 2000)软件为多重序列比对软件。各种软件的安装说明及使用方式可详见软件包中的说明文档（即 README 或 INSTALL 文档）以及相关网站。

表 1 本章中使用的软件及其获取方式

Table1 Source and functions of primary programs

程序名称	所属软件包	获取方式	功能描述
Blast	NCBI-toolkit	ftp:// ftp.ncbi.nih.gov/toolbox/ncbi_tools/ncbi.tar.gz	本地化序列相似性分析
est2genome	EMBOSS	ftp://ftp.uk.embnet.org/pub/EMBOSS/EMBOSS-2.6.0.tar.gz	EST 比对基因组序列
Phrap	Phrap	E-Mail: phg@u.washington.edu	EST 序列的组装
cross_match	Phrap	E-Mail: phg@u.washington.edu	载体序列的去除
GENSCAN	GENSCAN	http://genes.mit.edu/license.html	基因预测
MZEF	MZEF	ftp://cshl.org/pub/science/mzhanglab/mzef/	基因预测
Clustalw	Clustalw	ftp://ftp.ebi.ac.uk/pub/software/unix/clustalw/	多重序列比对
T-COFFEE	T-COFFEE	http://igs-server.cnrs-mrs.fr/~cnotred/Packages/	更精确的多重序列比对
Bioperl*	bioperl	http://bioperl.org/DIST/bioperl-1.2.1.tar.gz	Perl 模块

注：所有的软件包里都附有安装和使用说明，安装前请仔细阅读。*Bioperl 并非程序，为 perl 模块。

为了方便在实验室通过局域网上运行 Blast 和 EMBOSS 程序，我们同样安装了网络版 blast(张成岗等，2001)和 EMBOSS WEB 接口 W2H(Senger et al., 1998)，使用 Apache Web 服务器。为了客户端和服务器之间传输文件的方便，安装 FTP 服务器是个很好的选择，我们使用 Proftpd 软件。

2.4 EST 载体序列的去除及 Blast 数据库的建立

在服务器上建立相应的目录/blastdb，把存有 EST 序列的 my_db 文件上传至该目录，同样把基因组序列文件 WhiteRot 上传至该目录。利用 Phrap 软件包中的 cross_match 程序去除 EST 的载体序列。在该软件包中，作者收集了一部分载体序列，存在于 vector.seq 文件中。在使用程序前，先检查是否已存在所需的载体序列，否则要把载体序列加在该文件上，或者存为单独的文件，运行程序时指明载体文件名：$ cross_match my_db vector.seq -minmatch 12 -minscore 20 -screen > my_db（注：本文所有的$都表示 linux 下命令行提示符，其中 vector.seq 可以用自己的载体序列文件代替）。然后使用 Blast 软件包中附带的 formatdb 程序格式化数据库：

$ formatdb -i /blastdb/my_db -p F -o T

$ formatdb -i /blastdb/WhiteRot -p F -o T

formatdb 程序的各种参数含义可参见说明文档或者通过运行不带任何参数的 formatdb 程序获得。参数"-o T"不可少，后面使用自动化处理时采用 fastacmd 程序取出相关序列是必须的。

2.5 数据处理过程及自动化实现

为了实现分析过程的自动化，我们采用 perl 语言并结合 bioperl 来书写脚本程序。Bioperl 是由国际团体共同开发和维护的用于处理生物数据的 perl 模块(Stajich et al., 2002)，使用 bioperl 模块书写 perl 脚本可以非常方便地控制 est2genome，blast，GENSCAN，Phrap 等程序的输出结果，使之能自动成为下一个分析程序的输入，因此能实现整个过程的自动化。

整个程序设计思路由以下几个步骤构成：

从 my_db 数据库中取出指定的序列。使用 NCBI-toolkit 软件包中程序 fastacmd：$ fastacmd -d /blastdb/my_db -s <your_sequence_name>

冗余 EST 序列的去除。在所测定的 433 条 EST 中，我们发现存在大量冗余现象，有一些 EST 是完全一致或高度相似的。为了避免大量的重复性工作，去除冗余序列是必要的。这个过程可以通过运行 my_db 内部 Blast 比对完成。

序列的组装。通过运行 Phrap 程序，对高度相似的一批 EST 进行组装，尽

可能地获取较长 EST 序列。

EST 的基因组定位。通过运行 Blast（EST 比对基因组序列）和 est2genome 程序，确定该 EST 位于基因组中的 contig，并定位出该 EST 序列的外显子、内含子的区域和边界。

完整基因的预测。使用 GENSCAN 和 MZEF 程序对相应的基因组 contig 进行基因预测。对比由 EST 所确定的外显子、内含子区域和边界，确定可能的完整基因。

对预测的完整基因进行更精确的分析。通过 Blastx 分析，尝试找出同源蛋白。选择同源性较高的蛋白序列并保存至本地。使用 Clustalw 和 T-COFFEE 程序做多重序列比较，分析其可能的功能及做进化树分析。为了获得更精确的预测基因，联网至 http://genes.mit.edu/genomescan.html 做基于同源蛋白的 GenomeScan 分析(Yeh et al., 2001)。

3 结果

按照上述程序设计的思路，结合 bioperl 模块，我们用 perl 语言写了 2 个脚本程序：Autoaccemble.pl 和 Est2genoscan.pl，所有的 Linux Shell 命令集成于上述 2 个脚本，并放置在/usr/local/AutoEST 目录中。程序代码限于篇幅从略。完整的源代码、使用方法及系统构建的说明文档可以联网至 http://scbi.scu.edu.cn/AutoEST 获得。其中，Autoaccemble.pl 实现的功能是 EST 序列的分类、冗余 EST 的去除及序列组装，Estgenoscan.pl 实现的是 EST 在基因组的定位、外显子和内含子的区域与边界，基于基因组序列的基因预测。由于 blastx 比对和 GenomeScan 基因预测所需的硬件配置较高，我们建议通过网络进行。

使用该系统，我们已成功地处理了所有的 EST 序列，得到如下统计结果：

1）433 条 EST 序列经过聚类分析，去除冗余序列，分为 195 类。

2）其中 119 类由 1 个外显子转录，76 类由 2 个或更多的外显子转录，共决定 121 个内含子。

3）121 个内含子中有 111 个长度在 50~60bp 之间，占 91.34%。

4) 有 23 类 EST 能匹配基因组上多个位置，推测可能为基因家族成员。

限于篇幅，我们以一条 2d 特异表达的编号为 ssh2345 的 EST 序列为例介绍分析结果。该序列长度为 234bp。

冗余性分析：ssh2345 为单一序列，无高度相似性序列。

序列组装：由于该序列缺乏足够相似序列，组装后的序列和原来相同。步骤 1~2 使用命令：$ /usr/local/AutoEST/ Autoaccemble.pl ssh2345 完成。

EST 比对基因组序列：该序列位于基因组序列 contig： Scaffold_4 上，该 Contig 长度为 509546bp。其外显子与内含子区域为：

类型	长度	EST 边界	基因组边界
Exon-1	47	1~47	105268~ 105314
Intron-1	56		105315~ 105370
Exon-2	109	48 ~156	105371~ 105479
Intron-2	51		105480~ 105531
Exon-3	77	157 ~234	105532~ 105609

基因预测：使用 GENSCAN 程序对该基因组序列进行预测，得出如下结果：

类型	长度	起始	结束
Promotor	40	104071	104110
Init-1	402	104247	104648
Intr-2	150	104766	104915
Intr-3	198	104946	105143
Intr-4	117	105198	105314
Intr-5	109	105371	105479
Term-6	104	105532	105635
PlyA	6	105925	105930

注：限于篇幅，内含子区域从略。Init 为起始外显子，Intr 中间外显子，Term 为末尾外显子。

步骤 3~4 使用命令：$ /usr/local/AutoEST/Est2genoscan.pl ssh2345 完成。比较 Exon-1~ Intr-4，Exon-2~ Intr-5，Exon-3~ Term-6，其实际的外显子边界和预测的完全吻合（由于限制酶的作用，Exon-1 的起始端和 Exon-3 的末尾端部分被切掉）。该预测基因 CDS 长 1080bp，编码 359 个氨基酸。根据我们对黄孢原毛平革菌已知基因的统计分析，该菌基因的 CDS 长度一般为 800~1500bp，内含子长度一般为 50bp 左右，我们认为该预测基因有一定的准确度。做 Blastx 分析，并未发现高同源的蛋白序列，认为该基因可能为新基因。于是，就可以根据预测的外显子设计引物，在两天特异条件下用 PCR 扩增出该基因。

以上的分析结果使用本系统可以在几分钟内自动完成。

4 讨论

本文介绍了使用各种软件在 Linux 下构建大规模的 EST 序列分析系统，并通过结合具体实例探讨了该系统在实际应用中的价值。

使用该系统能否获得好的结果，关键在于 3 个因素：待分析物种的基因组测序是否完成、EST 序列是否足够丰富以及基因预测的准确性。当 EST 序列足够丰富时，就有可能通过序列组装获得全长 cDNA。当前，虽然基因预测软件较为成熟，但通过我们对黄孢原毛平革菌基因组片段进行的基因预测，发现效果并不理想。由 EST 确定的 121 个内含子中，GENSCAN 能准确预测到的只有 48 个，占 39.67%。原因在于 GENSCAN 和 MZEF 及其他一些基因预测软件都只基于人、玉米和拟南芥为模板对其他物种进行预测，从进化角度来说，真菌与上述 3 种模式生物的进化关系较远，从而造成基因预测的不准，这也是要期待解决的问题之一。

参考文献

张成岗, 欧阳曙光, 贺福初, 等. 基于 PC/Linux 的核酸序列分析系统的构建及其应用. 生物化学与生物物理进展, 2001, 28(2): 263~266.

Zhang C G, Ouyang S G, He F C, et al. Prog Biochem Biophys, 2001, 28(2): 263~266.

张成岗, 贺福初. 生物信息学在新基因全长 cDNA 序列分析及功能预测中的应用. 生物化学与生物物理进展, 2003, 30(1): 159~162.

Zhang C G, He F C. Application of Bioinformatics in Full - length cDNA Sequence Analysis and Function Prediction of Novel Genes. Prog Biochem Biophys, 2003, 30(1): 159~162.

张成岗, 贺福初. 生物信息学, 北京: 科学出版社, 2002, 41~43.

Zhang C G, He F C. Bioinformatics: Methods and Protocols, Beijing: Science Press, 2002, 41~43.

张成岗, 张利达, 贺福初, 等. 序列同源性分析软件Blast的WEB界面构建及其应用. 生物化学与生物物理进展, 2001, 28(6): 916~918.

Zhang C G, Zhang L D, He F C, et al. Prog Biochem Biophys, 2001, 28(6): 916~918.

Altschul S F, Madden T L, Schaffer A A, et al. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res. 1997, 25(17):3389-3402.

Burge C B, Karlin, S. Finding the genes in genomic DNA. Curr. Opin. Struct. Biol. 1998, 8(3): 346~354.

Cullen D. Recent advances on the molecular genetics of ligninolytic fungi. J.Biotechnol. 1997, 53(2-3): 273-289.

De Boer, H.A., Y.Z.Zhang, C.Collins & C.A.Reddy. 1987. Analysis of nucleotide sequence of two ligninase cDNAs from a white-rot filamentous fungus, Phanerochaete chrysosporium. Gene. 60: 93-102.

DeRisi,J.L., Iyer,V.R. and Brown,P.O. Exploring the metabolic and genetic control of gene expression on a genomic scale. (1997) Science, 278, 680-686.

Diatchenko,L., Lau,Y.-F.C., Campbell,A.P., Chenchik,A., Moqadam,F., Huang,B., Lukyanov,S., Lukyanov,K., Gurskaya,N., Sverdlov,E. and Siebert,P.D. Suppression subtractive

hybridization: a method for generating differentially regulated or tissue-specific cDNA probes and libraries. (1996) Proc. Natl Acad. Sci. USA, 93, 6025-6030.

Ewing B, Hillier L, Wendl M C, et al. Base-calling of automated sequencer traces using phred. I. Accuracy assessment. Genome Res. 1998 8(3): 175~185.

GP Yang, DT Ross, WW Kuang, PO Brown, and RJ Weigel. Combining SSH and cDNA microarrays for rapid identification of differentially expressed genes. Nucleic Acids Res., Mar 1999; 27: 1517-1523.

Notredame C, Higgins D G, Heringa J. T-Coffee: A novel method for fast and accurate multiple sequence alignment. J Mol Biol. 2000, 302(1): 205~217.

Rice P, Longden I, Bleasby A. EMBOSS: The European Molecular Biology Open Software Suite. Trends Genet, 2000, 16(6):276~277.

Schena,M., Dhalon,D., Davis,R.W. and Brown,P.O. Quantitative monitoring of gene expression patterns with a complementary DNA microarray. (1995) Science, 270, 467-470.

Schena,M., Shalon,D., Heller,R., Chai,A., Brown,P.O. and Davis,R.W. Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. (1996) Proc. Natl Acad. Sci. USA, 93, 10614-10619.

Senger M, Flores T, Glatting K, et al. W2H: WWW interface to the GCG sequence analysis package. Bioinformatics. 1998, 14(5): 452~457.

Stajich J E, Block D, Boulez K, et al. The Bioperl Toolkit: Perl Modules for the Life Sciences. Genome Res. 2002, 12(10): 1611~1618.

Yeh R F, Lim L P, Burge C B. Computational inference of homologous gene structures in the human genome. Genome Res, 2001, 11(5):803-8.

Zhang M Q. Identification of protein coding regions in the human genome by quadratic discriminant analysis. Proc Natl Acad Sci U S A. 1997, 94(2):565-568.

Zhang, Y.Z. & C.A.Reddy. 1988. Use of synthetic oligonucleotide probes for identification ligninase cDNA clones. Meth. Enzymol. 161: 228-237.

Zhang, Y.Z., G.J.Zylstra, R.H.Olsen & C.A.Reddy. 1986. Identification of cDNA clones for ligninase from Phanerochaete chrysosporium using synthetic oligonucleotide probes. Biochem. Biophys. Res. Commun. 137(2): 649-656.

文 献 综 述

生物信息数据整合与 Web Services、XML 技术研究进展

生物信息数据整合与 Web Services、XML 技术研究进展

引言

随着 WWW 的蓬勃发展，通过因特网来传播生物信息是最快捷和方便的手段。由于生物数据的复杂性，庞大性，利用计算机和网络技术来收集，整理和交流生物数据就成为必然，也是当今生物信息学蓬勃发展的原因。然而，与传统数据库中高度结构化的数据相比，Web 上的数据最大特点就是结构化特征较弱，往往是半结构化（semi-structured）的，甚至是无结构化（no-structured）的。而且每个数据源都是分散的，自治的（autonomous），生物学家们要想获取相关数据异常得艰难。

Web Services 将是下一代分布式系统的核心。Web Service 提出了一种新的面向服务的体系结构,它结合了面向组件方法和 Web 技术的优势。它建立在 W3C 和 IETF 开放的标准之上，已逐渐成为电子商务、企业界公认的技术准则。XML 是一种新型的完全结构化的可扩展标识语言。Web Services 和 XML 技术给分布和整合生物信息数据提供了一个新的方向。

1 生物信息数据整合

1.1 数据整合（data integration）的概念

以人类基因组计划为代表的基因组时代，产生了并正加速产生着庞大的生物数据。几乎生物体各个水平的数据正在呈爆炸性的增长，从 DNA，RNA，SNP，蛋白到代谢网络等等。然而，由于收集者的兴趣和采用的数据存储方式的不同，生物数据以不同的数据格式分散存储在 INTERNET 中，我们可以称之为生物数据的异质性：包括数据存放的地点（数据源），数据的文件格式，数据库管理系统（DBMS）的不同，数据库内容的关注点的不同，操作平台的不同以及检索系统的不同。生物学家如果想获得某一完整的数据，就不得不在各个数据源之间进行查找，花费大量的时间和精力，而且还不定能找到自己需

要的所有信息(Searls, 2003)。且随着后基因组时代的来临，系统生物学(Kitano, 2002)开始兴起，以整合的系统的观点分析生物学问题成为必然，因此，数据整合就成为系统生物学发展的起始的必然的阶段而受到广泛的重视。此外，因为数据整合是为了获得相关的所有数据，在生物制药中的应用也受到了广泛的重视，因为，通过整合所有相关的数据，研究者就能找出相关组分之间的联系，因而也就能更快地找到药物靶标，更快地实现药物设计(Sensmeier, 2003; Durand et al., 2003; Chapman et al., 2003; Yan, 2003; Pincus et al., 2003)。

数据整合是和数据关联（application interoperability），应用整合（application integration）相联系但又不同的概念。数据整合是指收集和存储来自各个不同数据源的在生物学意义上相互联系的数据(Anari et al., 2004; Cantor et al., 2003; Benini et al., 2003)。数据关联是指规范化数据接口（interface），使得从一个应用程序产生的数据能直接作为另一个应用程序输入的数据。应用整合则包括了以上的两个方面，是数据整合和数据关联结合在一起的使用。

1.2 生物数据整合的困难

生物数据整合是个复杂的工程，从技术角度来说，必须克服以下几个障碍(Lincoln, 2002; Ideker et al., 2001)：

1） 分散的数据源：计算机和信息技术应用于分子生物学研究以来，短时间内产生数以万计的数据源。

2） 数据格式的不一致：有传统的高度结构化的关系性数据库数据格式，也有半结构化的 flat files，text 文本格式，完全无结构化的 HTML 格式，同时也有从应用程序中新产生的数据格式（如 Blast 等）。数据格式的统一严重阻碍了信息交流与整合。

3） 数据源更新的不同步性：各个数据源是自治的，分属于不同的研究机构和私营企业。出去公开和不公开的差别外，各个数据源的更新是不同步的。

由于以上几个因素，要作到对生物信息的整合和对信息的精确查询是非常困难的，以下的图显示了由于数据源的分散和数据源更新的不同步性所造成查询记录的错误。

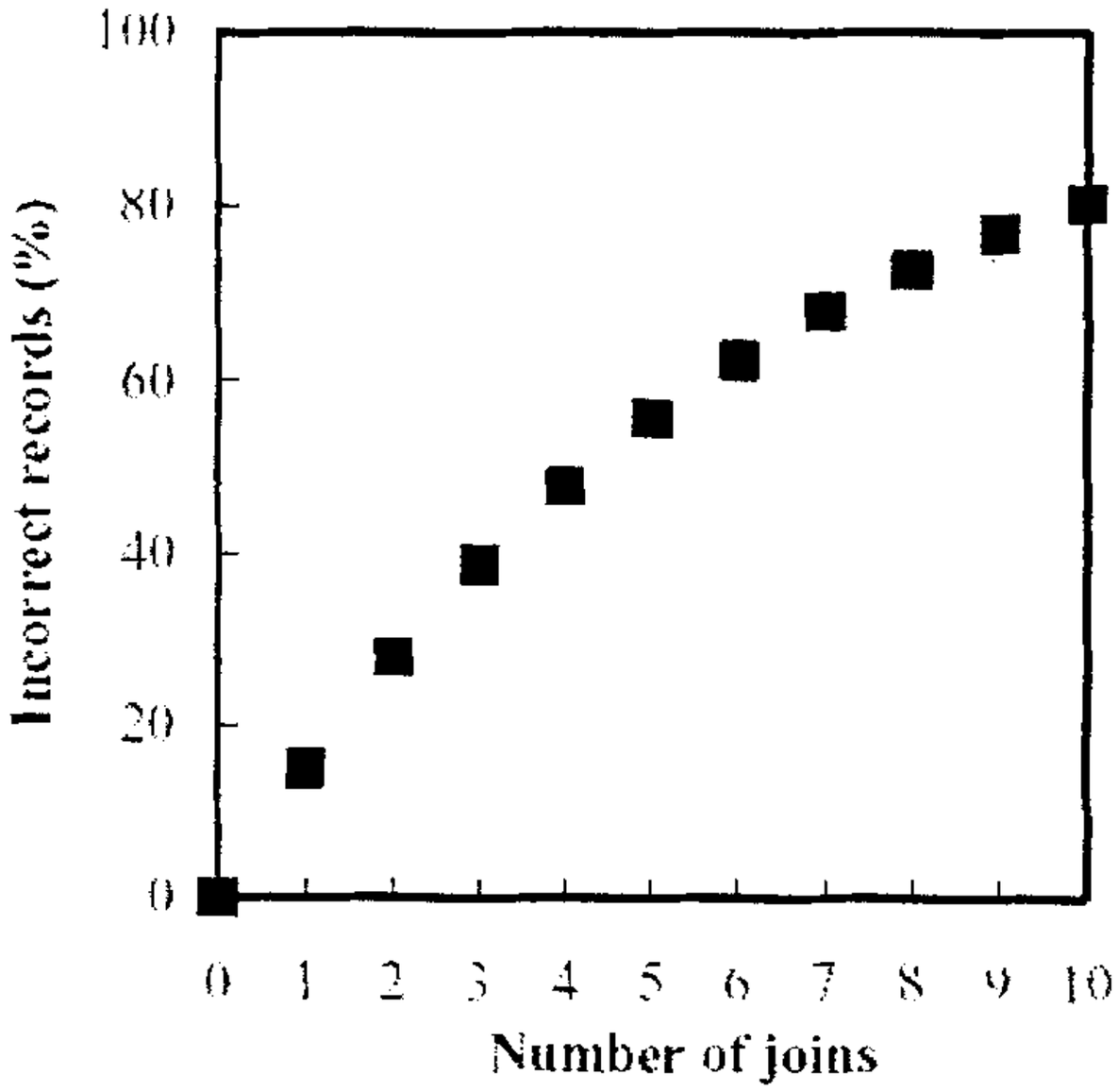

图 1 节点的数量与不正确的整合记录关系

Figure 1 Number of incorrect entries in a result set as a function of the number of joins between data sources. The calculation assumes that for each join between any two data source, 15% of the data retrieved is incorrect. The percentage of incorrect records increases geometrically with the number of joins and reaches nearly 50% by the 4th join. The calculation does not take into account records that should logically be included in the results but are missing due to the absence of cross-reference between data sources.

在这里一个接入点（join）表示不同数据源之间的联系，一般是代表同一信息的标识符。例如为了查找来自 SWISS-PORT 的一条蛋白序列存在于 GENEBANK 的所有相关核酸序列，那么这个接入点就是存在 SWISS-PORT 记录里的关于 GENEBANK 的交叉索引--Accession number。上图说明了查询的错误记录与接入点的关系，如果考虑每一个接入点将收到 15%的错误记录，那么增加到第四个接入点时，错误记录将呈几何数增长，占到整个查询记录的 50%。

因此，用于整合的数据源越多，错误的结果将越多。而数据格式的不一致更是个突出的问题，从数据管理的观点来看，WWW 上的数据比传统的数据库中的数据更难管理。传统的数据库都有一定的数据模型，可以根据数据模型来具体描述和操纵所要整合管理的数据。而 WWW 上的数据类型和数据间的关系非常复杂，没有特定的模型描述，或者虽有模型但却隐含与数据中，这些都是靠传统的数据库技术难以解决的。生物数据结构的不规范成为阻碍数据整合的关键因素，Lincoln Stein(Lincoln, 2002)把当今的生物信息状态比喻为中世纪，文艺复兴前的意大利，当时意大利分为几个行政区，各个行政区之间语言，重量的测量等等都不一致，缺乏规范而最终被外敌征服。他呼吁规范化所有的生物信息，以便能在更大更深的范围内实现信息的共享。

当然，从技术的角度来讲要完全解决上述问题还是不可能的。所有的数据整合技术都不得不有意地回避这些问题，采取折中的方式。

1.3 生物数据整合的理论研究现状（方法和策略）

生物数据整合就目前来说主要有四种方法：数据库联盟（database federation），数据仓库和数据市场（data warehouses and datamarts），专一化数据库（specialized databases），点方案（points solutions）。

1.3.1 数据库联盟（database federation）

该方式又可分为两种(Haas et al., 2002; Siepel et al., 2001)。一种是现实数据库联盟（Concrete database federation）是把外部的数据直接下载到本地，组成一个复杂的多结构的数据库然后通过设计复杂的查询系统进行本地化查询。另一种方式是虚拟数据库联盟（Virtual database federation）是设计查询系统的中间平台，把用户提交的查询要求解析成一系列的子查询(sub-queries)，然后把这些子查询分别发送给各个异地数据源，当从各个数据源返回到服务器中后，服务器对返回的文件进行解析，整合到一个文件中返回给用户。这两种方式各有优缺点：第一种方式由于把外源数据完全下载到本地，所以可以不依赖数据源，具有完全自主化，可以根据自己的爱好设计检索系统，缺点是自动化底，不能及时得到最新的数据，且由于数据库太多，数据库结构过于复杂，对服务器要

求很高。第二种方式由于不改变外源数据，只是一个中间平台，访问的数据也是数据源本身，所以能及时得到最新的数据。但由于各个数据源的结构和输出格式不一样，造成解析文件的困难，因为对于每个数据源都要设计专门的解析器，但对服务器要求底。

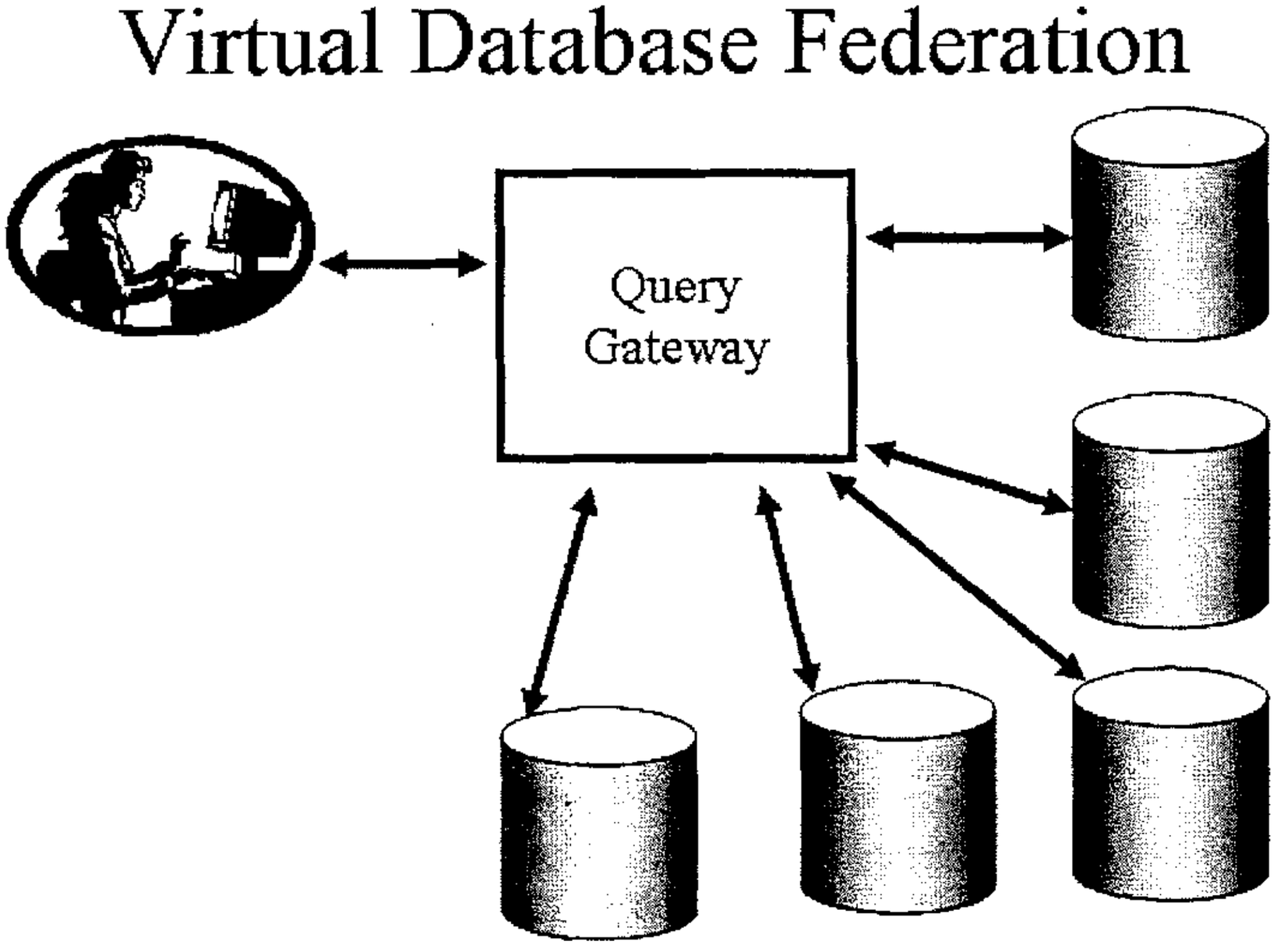

图 2　虚拟数据库联盟的工作流程

Figure 2　The work principle of virtual database federation

但从总的来说，数据库联盟只是一个对数据的汇总，并没有从中抽出信息，也不能改变原始数据，不能插入新的数据。对无用的数据也不能检测和清除。所以该方式还必须和其他的方式一起使用才行。

设计虚拟数据库联盟的目的是为了提供给用户对所获取的数据最好的观察方法，为了作到这一点，必须能获取数据（retrieval）并对数据进行抽提（extraction）。获取数据（retrieval）通过从那些已经格式化的文档中（XML，

flat files，output of a databases query or a tool, or an HTML output）选择文档。从理论上讲，一个完整的功能强大的数据库整合系统应该能对上述的所有文档格式进行获取，然而关系将会是异常复杂的。因此，实际上不是每个系统都能做到这一点。输出的文件格式是由数据源本身决定的。整合系统根据自己关注点选择合适的数据源，同时有些数据源提供几种格式的输出文件，整合系统根据自己的技术能力和爱好选择适合自己的文件格式。其中有些文件格式要解析极其困难，如 HTML，作为半结构化的数据格式，HTML 语言中没有固定的有实际意义的标签(tags)，它的标签仅仅是为了显示的(我们通过浏览器所看到的)，这些标签代表的意义是不容改变的。当然，如果数据源输出的 HTML 文件格式能一直保持不变，解析 HTML 文件，从中提取信息也是容易作到的。问题是 HTML 及易发生变化。而对于解析器来说，即使 HTML 文件格式中发生了微小的变化，解析工作就回失败！所以，要整合来自数据源的 HTML 文档信息，维护服务器将花费大量的时间和金钱。

例子：LION's Sequence Retrieval System (SRS) (Etzold et al., 1993)，IBM's DiscoveryLink(Haas et al., 2001)，geneticXchange's K1 integration middleware(Ideker et al., 2001)。

1.3.2 数据仓库和数据市场（data warehouses and datamarts）

该方法是在上述方法的基础上构建一个自定义的数据库用来储存通过上述方法整合而来的数据(Kimball, 1996)。同时还得设计应用程序来组织数据库并提供接口以便适应使用者应用软件来进一步分析这些数据。构建这样的数据库一般使用数据仓库和数据市场（data warehouses and datamarts）。

1.3.3 专一化数据库（specialized databases）

用来进行数据整合的专一化数据库不同于数据仓库。这种数据整合系统合并了数据仓库和处理系统的特性。例如，有时候我们要对实验室得到的数据作出及时的反应，而这些数据从实验室出来时是一些无结构性的 flat files 和 text 文本文件，记录间也缺乏内部联系，因此，必须对这些数据进行预处理才能整合进数据库或者数据仓库，因此这种整合系统应该包含对数据的处理系统。

例子：GeneX (http://www.genex.com)， aMAZE (http://www.amaze.com/)

1.3.4 点方案（points solutions）

该方案关注的焦点是找到查询到结果的最短距离。如果给定一组相关的数据源和所研究的问题，创建一个专门的软件包来回答该问题是可能的。例如，如果要根据 SWISS-PORT 的记录找到所有存在 EMBL 中相关的核酸序列，只要写一个脚本（script）就行，而无须建立专门的数据库和数据仓库。为了快速地创建这样的脚本，Perl 语言往往是一个好的选择。

很显然，四种方法各有特点，采取何种方法取决于所要解决的生物学问题。有时候用一种方法就能达到目的，更多的则必须多种方法组合使用。近年来，许多数据库团体开始致力于大规模的数据整合。数据整合一般由两个部件组成(Lacroix, 2002)：1）捕捉（wrapping）数据源，2）把捕获的数据存入数据仓库或者直接返回给用户。捕捉数据源意味着从数据源查询到数据并把它转换为用于整合的数据格式。

1.4 结语

综上所述，生物信息数据整合是当今生物信息发展的重点同时也难点之一。各种信息技术如今都被生物信息学家们用来进行分布和整合生物数据，但到今天并没有一个十分好的解决方法。生物数据的复杂性决定了要整合生物数据将是一个漫长而艰辛的路程。但是，随着信息技术飞跃的发展，随着许多大公司、特别是商业性的公司参与，如 IBM、SUN 等，数据整合的方法也在不停地发展和完善。

2 XML 概述与研究进展

2.1 XML 技术概况

2.1.1 标记语言

标记语言早于 Web 许多年出现，起起源可以追溯到商业出版初创阶段。在编辑们准备出版手稿时，常常用符号标记出文本的各个章节。例如，需要用斜体开始的节就会用字母 I 来标记，要指明斜体应该在何处结束时，该节的结

尾也要标记。用于指定文档结构的第一个正式的标记语言是 IBM 公司在 20 世纪 60 年代创建的。它就是众所周知的通用标记语言（Generalized Markup Language，GML），最初它被用于标准化 IBM 的内部文档。不久，它被扩展为标准通用标记语言（Standard Generalized Markup Language，SGML）。SGML 成为了许多不同工业采纳的信息表示标准。SGML 是 ISO 在 1986 年所制定的描述文档资料的结构与内容、实现文档交换和共享的国际标准。它是数据描述、数据模型化和数据交换的标准，同时又是一种元语言，元语言是一套可以用来定义其它更专门性的标记语言的通用规则，HTML 就是由 SGML 所定义出来，专门使用在 WWW 上的标记语言。

SGML 能描述任何的信息结构与任何复杂的文件，其应用可以简单如 HTML，也可以复杂得像 TEI、EAD、CIMI；SGML 是完全可扩展的，可以针对各种类型的文件结构制定出合适的标签集(tag set)；SGML 是理想的资料储存格式，提供了相当多的选项功能，可以适用于最复杂的信息处理。SGML 并不专属于特定的平台与特定的应用系统，因此 SGML 文件可以在彼此不兼容的系统间交换，不会造成信息遗失，这个特性使得 SGML 文件可以长久保存。通过 SGML 文件内容模块的再利用，使得文件的产生更有效率，SGML 文件的内容可以重复利用，或者被其它的 SGML 文件使用，不须重新产生内容。同一份文件内容也可以透过样式表(style sheet)以多种显示方式表达出来。

然而，尽管 SGML 有上述的特点，或者说优点。但是 SGML 同样存在着缺点，不适合现代信息技术的发展。SGML 庞大并且复杂的选项功能虽然使得 SGML 具有较高的灵活性，但也增加了应用程序开发上的难度，即使 SGML 工具的主要供应厂商 ArborText 所发表的产品，也没有百分之百支持 SGML 标准。事实上，SGML 有许多选项很少被应用，如果把这些不常用的选项去掉，将使得应用程序的开发变得更容易。要能够浏览 SGML 文件，必须要有文件类型定义(DTD)及样式表(Style Sheet)。DTD 定义了文件结构间的关系，样式表定义了这些结构的显示格式，如果少了 DTD 与样式表就只能看 SGML 文件的原始码。由于目前 Web 上常用的浏览器只支持 HTML，HTML 文件并不需要 DTD 与分离的样式表，因此 SGML 文件在 Web 上只能通过特定的浏览器(如 Panorama)才能阅读，不过这类的浏览器并不普及。如果希望 SGML 信息能在

Web 上被大多数人浏览，只好通过转换程序将 SGML 转成 HTML，但这样的转换往往会造成信息遗失(Information Loss)，原本 SGML 文件中所标记的结构在转换成 HTML 文件后并无法继续存在。Web 上的主流浏览器厂商 Microsoft 与 Netscape 支持 HTML 的发展，但并没有支持 SGML 的意愿，由于 SGML 过于复杂，也只有少数厂商愿意投资开发 SGML 的相关应用程序，这使得 SGML 在普及上存在很大的障碍。

随着万维网（WWW）的开始兴起，需要一门标记语言来表示在线传输的电子文档，HTML（Hypertext Markup Language）就是在这样的背景下诞生了。HTML 是一种专为 WWW 网页显示及浏览而设计的简易标记语言，目前是 WWW 上制作网页的标准语言格式。Tim Berners-Lee 对 HTML 所下的定义是："HTML 是一种用以创造超文件的简易资料格式，其所创造出来的文件可在不同的操作平台间移动"。由此可知，可移植性与简易性是 HTML 的两大特征。HTML 文件除了包含文字信息外，还可包括声音、影像等多媒体信息，而 HTML 的超链接除了网页内的链接，也包括网页之间的链接。毫无疑问，HTML 的成功超出了任何人的期望。然而，尽管如此，HTML 的一些局限性越来越不适应下一代网络技术的发展，特别是对于网络数据交换、分布式应用等方面。HTML 最大的局限就是它的标签集是固定的，而这些标签主要用来指定网页的显示格式，这个特性使得 HTML 只能支持固定、简单的文件结构，而且在信息再利用、资料交换与机读方面都存在很大的局限。许多相同的信息都有以不同的形式出版的情况，例如印刷版本、CD-ROM 版本、Web 版本等，尤其随着电子出版时代来临，数字化资料不论在复制、编辑、传播上都较传统出版来得方便，将同样的信息以各种不同的形式出版也变得更可行。如果以 HTML 作为电子出版的资料格式，设定不同的显示格式，如标题字体的大小、列表与表格的使用等，就能产生不同的 Web 版本，如果打印出来就是相对应的印刷版本，但由于 HTML 文件的资料内容与显示外观是结合在一起，如果原始文件的内容有所改变的话， 所有不同形式的版本全部都要跟着转换，这道转换的程序必须耗费不少的人力与时间。如果采用 SGML 作为电子出版的资料格式，由于资料内容与显示外观是分开处理，因此可以避免因原始文件内容改变而造成所有的版本都必须转换的问题。由于 Web 的普及，上网人口不断增加，

使得 Web 成为资料获取与交换最理想的场所，但由于 HTML 的标签集是固定的，利用 HTML 作为资料交换的格式，很难对每一项所要交换的资料作清楚的描述。例如：有一家网络书店想要通过 Web 从出版商那里取得一些新出版书籍的书目资料，并希望把这些资料自动转入自己的数据库中，再动态地把新书信息显示在网站上，书目资料包括了作者、书名、出版社、ISBN.等字段，以 HTML 标签来标记这些书目资料，并没有逐一标记每个字段，通常是把它制成表格的形式，以利于浏览，但如此一来却没有办法利用程序将 HTML 中的书目资料转入数据库中，因为程序没法分辨 HTML 文件中哪一段信息是作者、哪一段信息是书名，就算出版商以 SGML 来储存书目资料，清楚地描述每一个书目资料的字段，但一旦要通过 Web 传送， 将 SGML 转成 HTML 后，这些书目资料的字段结构就无法存在了。自动化处理可节省人力操作的成本，降低人工输入的错误，改善整体作业流程的质量，并提高文件传递的速度。目前在 Web 上一些应用程序就是自动化处理的简单应用， 如有些网上问卷调查或网上投票系统，使用者将填完的问卷调查资料直接传入主机的数据库后，可以直接实时读取数据库的统计结果。由于 HTML 的标签集是固定的，因此 HTML 文件所能做的自动化处理事实上有很大的限制。所有文件处理高度自动化的流程，都必须通过统一的资料格式，而且这个资料格式必须能携带丰富的内容，从这个角度来说 HTML 并不是一种适合作自动化处理的资料格式。

目前在 Web 上使用者可以透过搜索引擎所提供的关键词查询来寻找相关的信息，但由于目前 Web 上的信息不断增加，使得搜索引擎的查询结果往往会找到太多的信息，而这些信息又不一定能符合自己的信息需求，使用者往往会花大量时间去过滤信息。搜索引擎的准确率不高是因为所用的查询模式是对网页进行全文检索，虽然也可以将搜寻的目标限制在 HTML 文件的 Title 部分来提高准确率，但这样又会降低查全率。此外，HTML 的不断修订增加了许多网站额外的维护工作。由于 HTML 是一个发展中的标准，每当 HTML 的标签集不能满足需求时，W3C 就会为 HTML 加入新的标签，推出新的 HTML 版本。每当新的 HTML 版本推出，维护部门就得重新回头检查这些旧版的 HTML 文件，看看有没有需要重新标记文件。除了 W3C 会以官方立场身分修订 HTML 外，Microsof t 以及 Netscape 也会伴随着新版的浏览器推出自己的 HTML 扩展

标准，而两家厂商推出的扩展标准又不完全兼容，对于许多网站维护人员来说，每当有新版的浏览器问世，就代表着可能又要对部分的网页重新标记。有些组织为了避免重新标记文件，干脆决定采用 SGML 来标记文件，再把 SGML 转换成 HTML，因为将 SGML 转成 HTML 只需通过转换程序批量进行并不需花费太多资源。

虽然在因特网上被主要用来传播信息的语言是 HTML（Hypertext Markup Language，超文本链接标示语言），但是用 HTML 书写的文档数据模型是半结构化的。为了有效利用 WWW 上的数据，需要处理半结构化数据源，解决半结构化数据的查询与整合问题。寻找一个半结构化的数据模型是解决问题的关键所在。

2.1.2 XML

XML（eXtensible Markup Language，可扩展标识语言）是 W3C (http://www.w3c.org/xml) 在 1996 年底提出的标准，它是从 SGML 衍生出来的简化格式，也是一种元语言，可以用来定义任何一种新的标记语言(Needleman et al., 1999; XML, 1999; Serge, 1999; John, 1999)。XML 的制定是为了补足 HTML 的不完美，使得在 Web 上能够传输、处理各类复杂的文件，它去除了 SGML 复杂不常用及不利于在 Web 传送的选项功能，让使用者可以很容易地定义属于自己的文件类型，程序设计员也能在更短的时间开发 XML 相关应用程序。XML 的发展获得了各界的支持，其中包括了 Sun，Mcrosystems，Microsoft，Netscape，Adobe，ArborTex t 等软件大厂。

XML 在最近几年已经成为信息技术中最引人注目的技术之一。不管使用 XML 的是哪种平台或者应用程序，它都可以用来描述信息。现在，XML 几乎用于每一个可能的行业中，比如，医疗，汽车制造以及多数应用（如信息交换，数据库管理和内容管理）中(Bart et al., 1998; Russo et al., 2001; Franky et al., 2002)。XML 来源于 SGML（the Standard Generalized Markup Language），SGML 是定义描述不同类型电子文档结构和内容的国际标准。SGML 允许你利自定义标签来定义你自己的标识语言，因此它是一种元语言（meta-language）。XML 继承了这个特点（HTML 虽然也来源与 SGML，但不具这个特点），就象 SGML，

XML 也是一种元语言，允许你自己定义一套标签应用于一个或多个文档。XML 是一群 SGML 的审核人员兼 Web 专家所共同制定的，而且不是为了要取代 HTML。就应用层面来看，XML 介于 SGML 与 HTML 之间，它克服了 SGML 过于复杂和 HTML 缺乏灵活的缺点。制定 XML 一个非常重要的目标是支持多重应用领域以及易于开发相关应用软件。XML 特别适合用来做数据交换的中间格式，因为它只是用来规范化存储数据，对数据的应用则交给不同的应用程序。因为 XML 文档在文档结构上几乎没有限制，所以需要一些方法将约束条件置于文档上。为 XML 文档提供约束可以使得应用程序读取它们时更具完整性。DTD（Document Type Definition，文档类型定义）是一种将约束置于 XML 文档的结构上的方法(Chung et al., 2002)。因此，只要遵循同一个 DTD 约束的所有 XML 文档都能方便地进行数据交流。

XML 的主要特点(Achard et al., 2001; Bernadette et al., 2003; Kristensen, 1998; Aloisio et al., 1999; Kristensen, 1999)。

(1) 简单性。XML 为程序员和文档作者提供了一个友好的环境。XML 的严格定义和规则集使人类和机器都能更容易地阅读文档。XML 文档语法包含一个非常小的规则集，使开发者能立刻开始工作。根据文档的结构，DTD 既可以通过一个标准过程创建，也可以由专家创建。XML 文档建立在基本嵌套结构的一个核心集的基础之上。当一层又一层的细节被增加，使结构变得越来越复杂时，作者或开发者只需要为内部结构的复杂化付出非常少的努力。这些基本结构可以被用来代表复杂的信息集合，而不需要改变结构自身。XML 的语法分析器也非常容易创建。

(2) 可扩展性。XML 在两个意义上是可扩展的。首先，它允许开发者创建他们自己的 DTD，有效地创建可被用于多种应用的“可扩展的”标签集。其次， 使用几个附加的标准，您可以对 XML 进行扩展，这些附加标准可以向核心的 XML 功能集增加样式、链接和参照能力。作为一个核心标准，XML 为可能产生的别的标准提供了一个坚实的基础。

(3) 互操作性。XML 可以在多种平台上使用，而且可以用多种工具进行解释。因为文档的结构是相容的，所以解释它们的语法分析器就可以以较低的费用建立。XML 支持用于字符编码的许多主要标准，允许它在全世界许多不同

的计算环境中使用。XML 对 Java 进行了很好的补充，许多早期的 XML 开发是用 Java 进行的。一个用于语法分析器的普通的应用程序接口——XML 的简单 API (SAX)，可以免费获得。也可获得用 C++、C、Javascript、TCL 和 Python 等编写的语法分析器。目前，XML 语法分析器的开发集中在免费的插件上，这些插件为 XML 应用提供了语法分析能力，极大地降低了使用 XML 建立实际应用的费用。

(4) 开放性。XML 所采用的标准技术在 Web 上是完全开放的，可以免费获得。W3C 组织的成员已经较早地得到了这些标准，不过一旦此标准完成了，结果就是大家都可获得的。XML 文档自身也较为开放，任何人都可以对一个结构良好的 XML 文档进行语法分析，如果提供了 DTD，还可以校验这个文档。

XML 和 HTML 虽然都是 SGML 的扩展，但是它们之间也着极大的不同。XML 信息提供者能任意定义新的标签与属性名称。XML 文件结构可以是任意层次或复杂的嵌套结构。XML 文件可以包含语法的选择描述，让必须执行结构确认应用程序使用。XML 不像 HTML 只有内建的样式，XML 提供了样式表标准，称为可扩展样式语言(Extensible Style Language；简称 XSL)。XML 除了支持像 HTML 的简单链接，也提供了几种功能更强大的超链接机制。XML 的超链接机制被制定为 XML 链接语言(XML Linking Language；简称 XLL)与 XML 指针语言(XML Pointer Language；简称 XPointer)。

2.2 XML 在生命科学应用中的研究进展

XML 在科学研究中已引起广泛的重视。许多商业和政府研究机构都采用 XML 来作为他们管理数据的标准。化学和数学研究团体很早就开始利用 XML 来书写和交换科研数据，分别命名为 CML（the Chemical Markup Language，http://www.xml-cml.org/ ）和 MathML（the Mathematical Markup Language，http://www.w3.org/Math/ ）。CML 和 MathML 实际上是定义了 DTD，遵循该 DTD 约束的 XML 文档之间能方便地进行数据交流。

近年来，国外的生物信息研究机构也开始采用 XML 作为数据格式，用来注释序列。几个公共组织定义了 DTD：

1) The Bioinformatic Sequence Markup Language (BSML ,

http://www.visualgenomics.com/products/index.html）。该 DTD 用来规范对 DNA，RNA 和蛋白序列以及它们的图象特征的注解(Lichun, 2002)。我们发现这个 DTD 定义的文档结构和 EMBL/Genebank/DDBJ 数据库的信息结构非常相似（http://www.emi.ac.uk， http://www.ncbi.nlm.nih.gov， http://www.ddbj.nig.ac.jp ）。

2）The BIOpolymer Markup Language （BioML；htpp://www.protemetrics.com/BIOML/ ）。过程与 BSML 有点不同，引用作者的话(Fenyo, 1999)，BioML 的目标是："allow the expression of complex annotation for protein and nucleotide sequence information, BioML was designed to mimic the hierarchical structure of a living organism"。该 DTD 定义了一些不同来源信息的数据整合的标准。

这两个团体（BSML and BioML）从商业角度上来说开发 XML 语言的目的是相同的：他们都旨在指定一个编码生物信息的公共标准。另一方面，按照这个标准出售相关的软件，例如：专门的浏览器，数据整合和数据管理工具。

3）The taxonomic markup language consists of a DTD for the description of taxonomic relationships between organisms(Gilmour, 2000).

4）XML 同样被基因存在论联盟（the gene ontology consortium）（http://www.geneontology.org）所采用(Gene Ontology Consortium, 2000)，用来提供对于描述分子功能，生物过程以及基因产品细胞内定位信息的可控词汇（指可被接纳的词汇，包括科学命名和通用名称以及可接受的名称缩写等等）。

5）BioXML project（http://bioxml.org）更作为一个国际性的组织，正在加强 XML 在生命科学中的应用，并使其规范化。而且 BioXML 采取开放源代码的政策，任何人都能参与开发和错误修订。

6） SBML（the Systems Biology Markup Language, http://www.sbml.org/ ）: is a free, open, XML-based format for representing biochemical reaction networks. SBML is a software-independent language for describing models common to research in many areas of computational biology, including cell signaling pathways, metabolic pathways, gene regulation, and others(Hucka et al., 2003).

网址：

AGAVE, http://www.agavexml.org/

aMAZE: http://www.ebi.ac.uk/research/pfmp/

BioJava: http://www.biojava.org

BSML, http://www.bsml.org/

BioML, http://www.bioml.com/BIOML/

CellML, http://www.cellml.org/public/about/what_is_cellml.html

CML, http://www.xml-cml.org/

DAS, http://biodas.org/

GAME, http://www.bioxml.org/Projects/game/

GeneX: http://www.ncgr.org/genex/

MAGE, http://www.mged.org/Workgroups/MAGE/mage.html

BioPerl: http://www.bioperl.org

SBML, http://www.sbml.org

SP-ML, http://www.ebi.ac.uk/swissprot/SP-ML/

SRS6, http://srs.ebi.ac.uk/

2.3 结语

XML 因为具有高度结构化以及良好扩展性已经被广泛用于商业、工业乃至整个科学领域。采用 XML 格式来表达生物数据能够解决困扰生物数据传输、发布以及整合的大部分问题，是生物信息领域研究的热点以及重点。近年来，国外许多生物信息团体，包括商业团体都注重 XML 在生命科学领域中的应用，并取得了较大的进展。尽管如此，XML 作为一种生物数据格式并没有得到广泛的应用，特别是在我国。本身因为生物信息在我国起步较晚，几乎所有的领域都大大落后与国外研究团体。一个原因是我国的生物信息人才匮乏，特别是跨学科领域严重落后。虽然近两年来，有些大学，如北京大学、浙江大学等开始了生物信息方面的教育，但还是始终停留在学习他们的成果上。XML 作为一种标准提供了我国生物信息发展的一个契机，有助于提供我国生物信息的发展。

3 Web Services 研究进展

3.1 Web Services 概述

Web Services 应是一种基于组件的软件平台，是面向服务的 Internet 应用。通过对 Web Services 的构建，人们可以期望得到一个可编程的 Internet。这个观点包括了两层含义：首先，Web Services 应是应用于 Internet 的，而不是限于局域网或试验环境。这要求提出的 Web Services 框架必须适用于现有的 Internet 软件和硬件环境，即服务的提供者所提供的服务必须具有跨平台、跨语言的特性。其次，Web Services 所提供的服务不仅是向人，更需服务于其它应用系统。现有的 Web 网站也可以认为是面向服务的，但这种服务仅仅可以提供给人使用(只有人类才可以读懂浏览器下载的页面)。而新一代的 Web Services 所提供的服务应能被机器所读懂，例如其它应用程序及移动设备中的软件系统。这样，我们可以看出，Web Services 的发展方向实际上是构造一个在现有 Internet 技术上的分布计算系统(Ingram, 1996; Huang et al., 2003; Ardissono et al., 2003; Mark, 2003)。

Web services 是封装成一个单一实体并通过网络发布给其它程序使用一系列功能集。它是自包含、自描述、模块化的应用，可以发布、定位、通过 Web 调用。Web Services 可以执行从简单的请求到复杂商务处理的任何功能，一旦部署以后，其他 Web Services 应用程序可以发现并调用它部署的服务。因此，Web Services 是构造开发的分布式系统的基础模块，它们允许所有的企业和个人快速、廉价建立和部署全球性的应用(Francisco et al., 2003)。

新一代 Web Services 框架已浮出水面，IBM 和 Microsoft 公司都推出了面向开发者的支持工具，其核心技术包括 SOAP，WSDL 和 UDDI，它们都是以标准的 XML 文档的形式表达的(Gengler, 2002; Narayanan et al., 2003; Sycara et al., 2003; Hongbing et al., 2004)。Web Services 服务提供方通过 WSDL(Web Services Description Language)描述所提供的服务，并将这一描述告知 Web Services 注册服务器。注册服务器依据 WSDL 的描述，依照 UDDI (Universal Description Discovery and Integration)的协定更新服务目录并在 Internet 上发布。用户在使用 Web Services 前先向注册服务器发出请求，获得 Web Services 提供者的地址和服务接口信息，之后使用 SOAP 协议(Simple Object Access Protocol)

与 Web Services 提供者建立连接，进行通信。

Web Service 主要利用 HTTP 和 SOAP 协议使商业数据在 Web 传输，SOAP 通过 HTTP 调用商业对象执行远程功能，Web 用户能够使用 SOAP 和 HTTP 通过 Web 调用的方法来调用远程对象(Crow et al., 2001)。

3.2 Web Services 工作原理

Web Service 的服务原理和体系结构见图一。其主要优点有(Zhuge et al., 2004; Cardoso et al., 2004; Brahim et al., 2003):

（1）采用 W3C 标准，真正的与平台无关。允许在不同平台上、以不同语言编写的各种程序以基于标准的方式相互通信。这虽然与 CORBA 和 DCE 有着相同的目标，但是 SOAP 比以前的方法要简单得多。

(2) 使用标准的 Web 协议——XML，HTTP 和 TCP/ IP。许多公司都已经建立了 Web 基础结构，同时它们的员工在维护方面也都具备相应的知识和经验。因此，引入 XML Web Service 与引入以前的技术相比，其成本要低得多。

(3) 现有的远程访问协议——DCOM，CORBA，RMI 不能很好用于互联网环境。

(4) 不受现有的代理和防火墙的限制。

(5) 可以利用 HTTP 验证模式,支持安全套接层(SSL) 。 .

Web Services 工作原理：

(1) XML Web Service 通过标准的 Web 协议向 Web 用户提供有用的功能。多种情况下使用 SOAP 协议。

(2) XML Web Service 可以非常详细地说明其接口，这使用户能够创建客户端应用程序与它们进行通信。这种说明通常包含在称为 Web 服务说明语言(WSDL) 文档的 XML 文档中。

(3) XML Web Service 已经注册过，以便潜在用户能够轻易地找到这些服务，这是通过通用发现、说明和集成(UDDI)来完成的。

UDDI(Universal Description Discovery Intergration)——通用发现、说明和集成(UDDI)是 Web 服务的黄页。UDDI 允许你查找提供的 Web 服务的公司。

WSDL (XML Web services Description Language)——描述 Web service 的

语言规范，相当于访问 Web service 的接口。WSDL (Web Services Description Language)表示 Web 服务说明语言。WSDL 文件是一个 XML 文档，用于说明一组 SOAP 消息以及如何交换这些消息。换句话说，WSDL 对于 SOAP 的作用就像 IDL 对于 CORBA 或 COM 的作用。由于 WSDL 是 XML 文档，因此很容易进行阅读和编辑；但大多数情况下，它由软件生成和使用。

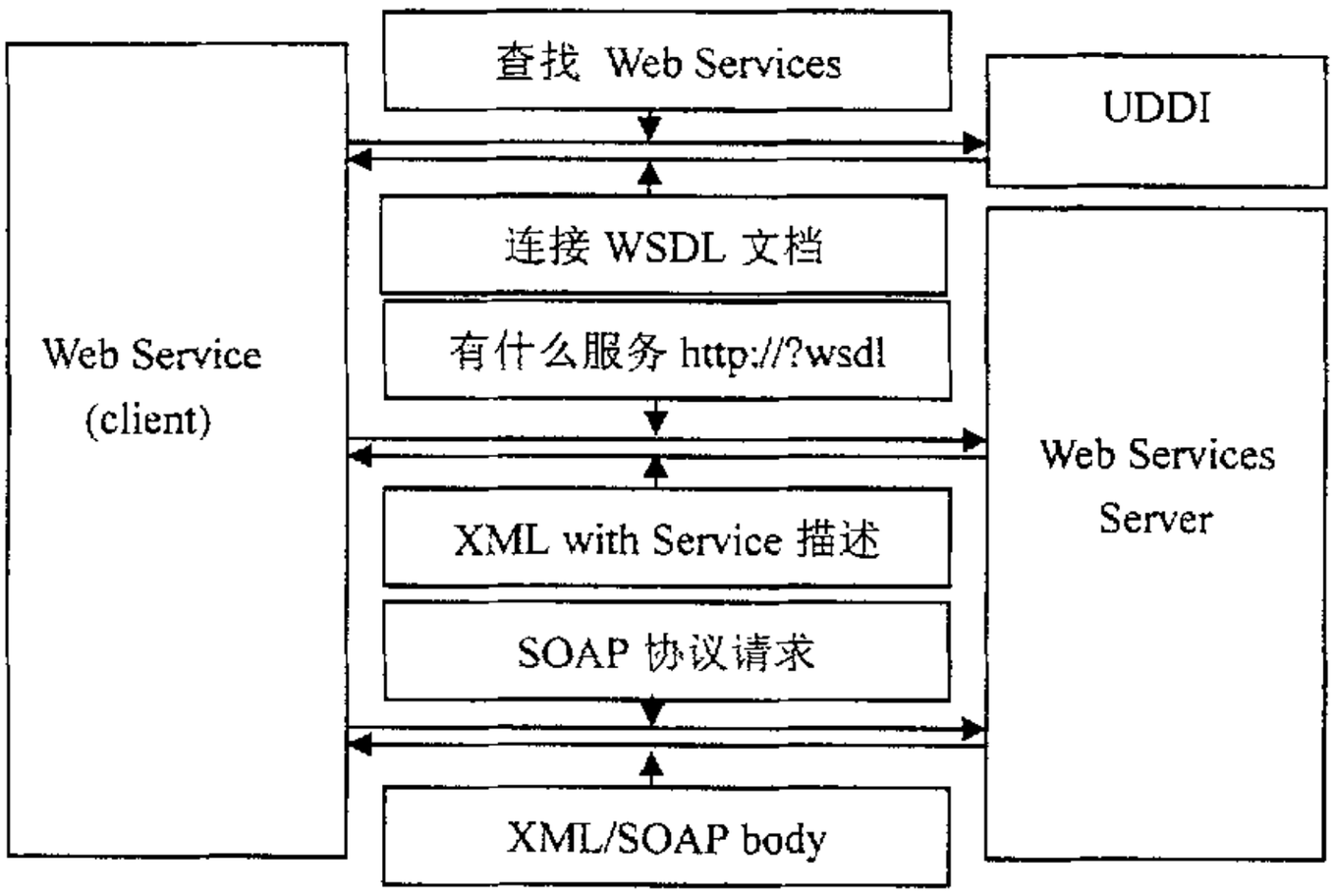

图 3　调用 Web Services 服务的工作流程

Figure 3　The work principle of calling Web services

Web Services 的体系结构是基于 Web 服务提供者、Web 服务请求者和 Web 服务中介者三个角色和发布、发现、绑定三个动作构建的。Web 服务提供者设计实现 Web 服务，并将调试正确后的 Web 服务通过 Web 服务中介者发布，并在 UDDI 注册中心注册；Web 服务请求者向 Web 服务中介者请求特定的服务，中介者根据请求要求查询 UDDI 注册中心，为请求者查找到满足请求的服务；Web 服务中介者向 Web 服务请求者返回满足条件的 Web 服务描述信息，该描述信息用 WSDL 写成，各种支持 Web 服务的机器都能阅读。Web Services 是构件技术和 Web 技术的融合，实现 Web 上可调用的构件、实现构件功能不是

问题，问题在于怎样对服务进行描述和注册、怎样发现服务、定义调用服务的消息格式，等等。XML 技术及 XML 技术与其它协议之间的绑定，构成 Web Services 发展的技术核心。

Web Services 的这种网络技术提供了一种不同机群之间在跨平台的应用程序下的数据共享能力，使所有的计算机群、相关设备（例如手机、PDA 等）和服务协同工作，提供更广泛和多样的服务，解决了目前信息孤岛的问题。

参考文献

Achard, F., et al. 2001. XML, Bioinformatics and Data integration. Bioinformatics 17: 115-125.

Aloisio, G.; Milillo, G.; Williams, R.D. An XML architecture for high-performance web-based analysis of remote-sensing archives. Future Generation Computer Systems. 1999, 16(1):91-100

Anari MR, Sanchez RI, Bakhtiar R, Franklin RB, Baillie TA. Integration of knowledge-based metabolic predictions with liquid chromatography data-dependent tandem mass spectrometry for drug metabolism studies: application to studies on the biotransformation of indinavir. Anal Chem. 2004 Feb 1;76(3):823-32.

Ardissono. L, A. Goy, G. Petrone. Web technologies: Enabling conversations with web services. Proceedings of the second international joint conference on Autonomous agents and multiagent systems. 2003, July(14-18):819-826

Bart Meltzer, Robert Glushko. XML and electronic commerce: enabling the network economy. ACM SIGMOD Record. 1998, 27(4):21-24

Benini AA, Conley CE, Shdeed R, Spurway K, Yarmoshuk M. Integration of different data bodies for humanitarian decision support: an example from mine action. Disasters. 2003 Dec;27(4):288-304.

Bernadette Farias Lósio, Ana Carolina Salgado, Luciano do Rêgo Galvo. XML and information integration: Conceptual modeling of XML schemas. Proceedings of the fifth ACM international workshop on Web information and data management. 2003, November 7-8:102-105

Brahim M, Athman B, Ahmed K. E. Composing Web services on the Semantic Web. The VLDB Journal - The International Journal on Very Large Data Bases. 2003, 12(4):333-351

Cantor MN, Lussier YA. Putting data integration into practice: using biomedical terminologies to add structure to existing data sources. Proc AMIA Symp. 2003;:125-9.

Chapman A, Yu C, Jagadish HV. Effective integration of protein data through better data modeling. OMICS. 2003 Spring;7(1):101-2.

Cardoso, J; Sheth, A; Miller, J; Arnold, J; Kochut, K. Quality of service for workflows and web

service processes. Web Semantics: Science, Services and Agents on the World Wide Web. 2004, 1(3): 281-308

Chung, Tae-Sun; Kim, Hyoung-Joo. Extracting indexing information from XML DTDs. Information Processing Letters. 2002, 81(2):97-103

Crow, L; Shadbolt, N. Extracting focused knowledge from the semantic web. International Journal of Human-Computer Studies. 2001, 54(1): 155-184

Durand P, Medigue C, Morgat A, Vandenbrouck Y, Viari A, Rechenmann F. Integration of data and methods for genome analysis. Curr Opin Drug Discov Devel. 2003 May;6(3):346-52.

Etzold T and P. Argos. SRS--an indexing and retrieval tool for flat file data libraries. Computer Applications in the Biosciences. 1993, 9: 49-57.

Fenyo D, The Biopolymer Markup Language, Bioinformatics, Apr 1999; 15, 339 - 340.

Francisco C, Rania K, Nirmal M, Stefan T, Sanjiva W. Service-oriented computing: The next step in Web services. Communications of the ACM. 2003, 46(10):29-34

Franky Lam, Nicole Lam, Raymond Wong. XML transactions: Efficient synchronization for mobile XML data. Proceedings of the eleventh international conference on Information and knowledge management. 2002, November 4-9:153-160

Gene Ontology Consortium, Gene ontology: tool for the unification of biology, Nature Genet, . 2000; 25:25-29.

Gengler, B. Web services push. Computer Fraud & Security. 2002, Apr 1, 4-4.

Gilmour, R. Taxonomic markup language: applying XML to systematic data, Bioinformatics, 2000; 10:406-407.

Haas, L.M. 2002. Data integration through database federation. IBM Systems Journal 41: 578-596.

Haas, L. M., P. M. Schwarz, et al. DiscoveryLink: A system for integrated access to life sciences data sources. IBM Systems Journal. 2001, 40(2): 489-511.

Hongbing, Wang; Huang, Joshua Zhexue; Qu, Yuzhong; Xie, Junyuan. Web services: problems and future directions. Web Semantics: Science, Services and Agents on the World Wide Web. 2004, 1(3): 309-320

Huang, Ying; Chung, Jen-Yao. A Web services-based framework for business integration

solutions. Electronic Commerce Research and Applications. 2003, 2(1): 15-26

Hucka M, A. Finney, H. M. Sauro, H. Bolouri, et al. The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models, Bioinformatics, Mar 2003, 19:524-531

Ideker, T., V. Thorsson, et al. Integrated Genomic and Proteomic Analyses of a Systematically Perturbed Metabolic Network. Science. 2001, 292:929-934.

Ingram, P. Web developments and the Internet. Computers & Geosciences. 1996, 22(5): 579-584.

John B. Bedunah. XML: the future of the Web. Crossroads. 1999,6(2)

Kimball, R. The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses, John Wiley & Son, Inc. (1996)

Kitano, H. Systems Biology: A Brief Overview. Science. 2002, 295(5560), 1589-1780.

Kristensen, Anders. Formsheets and the XML forms language. Computer Networks and ISDN Systems. 1999, 31(11-16): 1189-1201.

Kristensen, Anders. Template resolution in XML/HTML. Computer Networks and ISDN Systems. 1998, 30(1-7):239-249

Lacroix Z. Biological Data Integration: Wrapping Data and Tools, IEEE Trans Inf Technol Biomed. 2002, 6(2): 123-8.

Lincoln Stein. Creating a bioinformatics nation. Nature. 2002, 417:119-120.

Lichun Wang, Jean-Jack Riethoven, and Alan Robinson, XEMBL: distributing EMBL data in XML format, Bioinformatics, Aug 2002; 18: 1147-1148

Mark Little. Service-oriented computing: Transactions and Web services. Communications of the ACM. 2003, 46(10):49-54

Narayanan, S; McIlraith, S. Analysis and simulation of Web services. Computer Networks. 2003, 5(5): 675-693.

Needleman, Mark H. XML. Serials Review. 1999, 25(1): 117-121

Pincus Z, Musen MA. Contextualizing heterogeneous data for integration and inference. Proc AMIA Symp. 2003;:514-8.

Russo, Mark F., Rubin, A. Erik. An Introduction to Using XML for the Management of

Laboratory Data. Journal of the Association for Laboratory Automation. 2001, 6(6):89-94

Searls DB. Data integration--connecting the dots. Nat Biotechnol. 2003 Aug;21(8):844-5.

Sensmeier J. Advancing the state of data integration in healthcare. J Healthc Inf Manag. 2003 Fall;17(4):58-61.

Serge A. On views and XML. ACM SIGMOD Record. 1999, 28(4):30-38

Siepel, A. C., A. N. Tolopko, et al. 2001. An integration platform for heterogeneous bioinformatics components. IBM Systems Journal 40(2): 570-591.

Sycara, K; Paolucci, M; Ankolekar, A; Srinivasan, N. Automated discovery, interaction and composition of Semantic Web services. Web Semantics: Science, Services and Agents on the World Wide Web. 2003, 1(1): 27-46

XML - extremely muddled language, or the future of the internet? Current Biology. 1999, 9(22): R833-R834

Yan Q. Bioinformatics and data integration in membrane transporter studies. Methods Mol Biol. 2003;227:37-60.

Zhuge, Hai; Liu, Jie. Flexible retrieval of Web Services. Journal of Systems and Software. 2004, 70(1-2): 107-116

在读期间发表或被接收的论文

1 Xiao Li and Yizheng Zhang. Bioinformatics Data Distribution and Integration via Web Services and XML. Geno., Prot. & Bioinfo. 2003, 1(4):299-303

2 李校，张义正. Linux 平台下 EST 序列分析系统的构建及其在抑制削减杂交数据处理中的应用实例. 四川大学学报自然科学版. 2004（已接收).

3 Sun Xun, Jiang Ming-Feng, Li Xiao, Zhang Yi-Zheng. Rapid Screening of Expressed Genes of *Trametes gallica* by cDNA Microarray. Prog. Biochem. Biophys. 2004; 31(4):356-360

致　谢

首先，我要感谢我的导师张义正教授，先生渊博的知识、忘我的工作以及对科学执着的追求和无私的奉献精神给我留下了深刻的印象。在先生严格的要求下，我的专业水平得到了较大的提高。更要感谢先生对我的宽容，能让我从事自己感兴趣的研究方向，并从始至终给予我支持和鼓励，让我能一直保持对科学研究的信心和勇气。

在本论文的研究过程中，成都电子科技大学的杨世铭和张国庆同学给予了技术上的支持。对他们的帮助向他们表示衷心的感谢。

同时也非常感谢本实验室的所有老师和同学，感谢他们在我的整个研究生阶段给予的鼓励和支持。同他们的学习、交流使我收益非浅。

最后，我要感谢我的父母以及所有关心我帮助我的亲戚和朋友，正是由于他们的支持、关心、鼓舞和帮助，才能使我顺利完成我的学业。

声　明

本人声明所呈交的学位论文是本人在导师张义正教授的指导下进行的研究工作及取得的研究成果。据我所知，除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得四川大学或其他教育机构的学位或证书而使用过的材料。同时，文中使用的所有软件和语言均为开放源代码资源，其中有一些是得到作者同意后使用。与我一同工作的同志对本研究所做的任何贡献均已在论文中做了明确的说明并表示感谢。

本学位论文成果是本人在四川大学读书期间在导师指导下取得的，论文成果归四川大学所有，特此声明。