Oracle's Technology for Bioinformatics and Future Directions

Bruce Blackwell and Siva Ravada

Oracle Corporation, New England Development Center

1 Oracle Drive

Nashua, New Hampshire 03062

USA

bruce.blackwell@oracle.com siva.ravada@oracle.com

Abstract

The Oracle relational database management system, with object-oriented extensions and numerous application-driven enhancements, plays a critical role worldwide in managing the exploding volumes of bioinformatics data. There are many features of the Oracle product which support the bioinformatics community directly already and there are several features that could be exploited more thoroughly by users, service vendors, and Oracle itself to extend that level of support. This paper will present an overview of Oracle features that support storage of bioinformatics data and will discuss extensibility features that give the product room to grow. Some attention will be given to Oracle's own efforts to use that extensibility to make commodities of many of the complex data and computation requirements of the life sciences.

Keywords: Oracle; database; bioinformatics; extensibility.

1 Introduction

The sequencing of the human genome and that of other organisms is just one element of an emerging trend in the life sciences: while the knowledge, experience, and insight of researchers remain indispensable elements, the understanding of life processes is increasingly a data-driven enterprise. Huge volumes of bioinformatics data are emerging from sequencing efforts, gene expression assays, X-ray crystallography of proteins, and many other activities. Celera Technology's genome database comprises about 250 terabytes of data. Craig Ventor, formerly of Celera, has predicted that as the focus moves from genomics to proteomics the data storage requirements will grow to multi- petabytes, and the eventual move into metabolomics and personalized pharmacology will have data management requirements in the exabyte range. Almost every important process in the development cycle for therapeutic drugs is either a producer of big data volumes or a consumer, or both.

This paper appeared at the First Asia-Pacific Bioinformatics Conference, Adelaide, Australia. Conferences in Research and Practice in Information Technology, Vol 19. Yi-Ping Chen, Ed.

Obviously database management systems have a very important role to play in the storage and serving of this explosion of data. Relational technology has emerged as the standard for databases in many fields and bioinformatics is no exception. Oracle's relational database management system (RDBMS) now holds about 85% of the bioinformatics database market. The Oracle RDBMS has evolved from a simple relational data store to one that enables complex data structures to be explicitly represented in the database, and for those data structures to be provided with diverse functional behavior within the server. It is Oracle's philosophy to continually extend the idea of "commodity" in the provision of data services in response to trends in each industry we serve. The bioinformatics industry has grown so quickly that standards for the structure of data and for computational requirements have not kept pace with the data volumes. There is increasing realization, however, about the need for such standards and activity has begun to develop and promulgate them. As these standards emerge Oracle will enhance its support to the bioinformatics community by placing implementations of the standards inside the database.

Even without Oracle's commitment to continual extension and improvement of the RDBMS product, there are many features of the current offering that are already helping bioinformatics organizations deal with their immense data problems. This paper serves to highlight features of Oracle technology that support this community as well as giving some hints about where Oracle might be headed in the future. Section 2. presents some enabling technologies which are at the core of the Oracle RDBMS as applicable to bioinformatics. Section 3. follows up with some more focused examples of applications together with some current enhancement activities.

2 Enabling Technology

2.1 The Oracle Object-Relational Database

Huge volumes of complex data characterize Bioinformatics applications. Many of these applications would benefit from databases that can store and serve complex data types like sequences and protein structures. Object-relational database management systems (ORDBMS) incorporate object-oriented capabilities in a

database environment to better support this new breed of applications. Starting with the 8i release, the Oracle database has been a fully functional object relational database. The Oracle database provides abstract data types, object identity, and the ability to create operations or procedures through the database programming interface to add behavior to these objects. The Oracle ORDBMS extends the traditional relational database systems in three main areas: (i) data types, (ii) query language, and (iii) extensible indexing and the query optimizer.

In the Oracle database, a user can define new data types that closely represent the objects used in the application. With this approach, applications can focus on object manipulation logic (the object behavior) without worrying about the object storage and retrieval logic. The query language used in all the commercial relational database systems is SQL, and it can be extended to integrate the new operations defined by applications with standard SQL queries. This has the advantage that all the operations on the database objects are handled using a standard query model resulting in simplified application logic.

Indexes are optional structures associated with relational tables. Indexes speed up the execution of SQL statements in the database by providing a faster access path to the data, primarily by reducing disk I/O. Traditional databases provide an indexing mechanism which can work with simple data but this is not suitable for use with application-defined complex data types. With Oracle Extensible Indexing, the application can define and manage the structure and semantic content of the index, and can base the index on properties of complex abstract data types that are only understood by the application. The database system then interacts with the application to build, maintain, and employ the index. The main advantage of this extensible indexing framework is that the index is always in synchronization with the data; once the index is built, all the updates on the base table will automatically result in updates in the index data. Thus the users are not required to worry about the data integrity and index currency issues. Once the index is built, it is treated like a regular B-tree index. The database server knows the existence of the index and manages all the index-related work using user defined functions. In this extensible indexing model, the query optimizer also understands the application's index structure and operators. When the user-defined operators are evaluated, the corresponding index can be used to efficiently process the SQL query. This is similar to using a B-tree index (built in to Oracle) while evaluating an equality operator. The application can store the index data either inside the Oracle database (in the form of tables) or outside the Oracle database (in the form of files) although it is highly desirable for the database to handle the physical storage of the index. By doing this, the index data will have the same security and reliability as the rest of the data in the database.

2.2 Table Functions

Bioinformatics applications typically have workflows that pass data through multiple stages. This requires some mechanism to temporarily capture the output from one stage, and feed it to the next stage. Oracle Table Functions can be used to stream the data through multiple stages of processing without the need to explicitly manage the data storage between different stages. Table functions return a collection type instance representing rows in a table. They can be queried like a table by invoking the table function in the FROM clause of a query. Pipelined table functions enable a table function to return rows on demand and thus eliminate the need to explicitly manage the results from different stages of a pipeline. A pipelined table function can return the table function's result like a stream that can be fetched on demand. When a consumer function is ready to read in more data, the producer function can produce the results on demand. This makes it possible to use pipelined table functions like a virtual table.

2.3 Oracle Text

Life sciences applications typically deal with two types of problems with documents. The first problem is finding all documents that contain a search term or key phrase. The second problem is classifying an incoming stream of documents based on their content. Text applications are classified as Text Query Applications or Document Classification Applications based on which of the above two problems are addressed. Oracle Text is a set of services that enables both types of application. Oracle Text provides indexing, word and theme searching, and viewing capabilities for text. For text query applications, Oracle Text supports most document formats for indexing and querying, including plain text, HTML and formatted documents such as Microsoft Word. Theme searching of documents is supported by storage and access to a supplied knowledge base that is a hierarchical listing of categories and concepts. For example, one can search for all documents that are about the concept politics. Documents returned might be about elections, governments, or foreign policy. The documents need not contain the word politics to score hits. Since the supplied knowledge base is a general view of the world, new industry-specific concepts can be added to it. With an augmented knowledge base, the system can process document themes more intelligently and so improve the accuracy of theme searching. For document classification applications, Oracle Text supports classifying plain text, HTML, and XML documents, relying on the theme knowledge base as well.

2.4 Oracle Real Application Clusters

Real Application Clusters harnesses the processing power of multiple interconnected computers. Real Application Clusters software and a collection of hardware known as a cluster unite the processing power of each component to create a robust computing environment. In Real Application Clusters environments, multiple active instances of the database server on different processors can concurrently execute transactions against a shared database. Real Application Clusters coordinates each instance's access to the shared data to provide data consistency and data integrity. Harnessing the power of clusters offers obvious advantages. A large task divided into subtasks and distributed among multiple nodes is completed sooner and more efficiently than if the entire task were processed on one node. Cluster processing accommodates rapidly growing user populations and data volumes by enabling expansion of the hardware configuration without changing any applications.

3 Critical Bioinformatics Application Areas

This section describes some notable current and contemplated features of Oracle technology as applicable to bioinformatics problems. They are broken down into five categories: computation; data mining and statistics; data migration and integration; XML; and web services.

3.1 Computational Analysis of Biological Data

3.1.1 Oracle interMedia - Manage and Process Scientific Images and Multimedia Content

Much primary life sciences data is in the form of images: cell sections; gene-expression microarray plates; or 2D electrophoresis gels. There are also movies made to record such things as cell growth or a protein folding simulation. It is often convenient to manage this multimedia data and its metadata together with other relational data, such as annotations, in the database. Oracle interMedia provides management of multimedia content (image, audio, and video) in an integrated fashion with other enterprise and scientific information. It extends Oracle's reliability, availability, and data management functionality to multimedia content. Oracle interMedia supports the storage of multimedia data both inside and outside of database in most popular image formats. The metadata of image, audio or video files is stored and managed in the database and supports queries.

Oracle interMedia also provides image processing and other functions for the data including metadata extraction, thumbnail generation, cropping, format tronscoding, image compression/decompression, and image matching using extracted features. Users may also supply custom algorithms for image analysis. Oracle is continuing to extend the reach of image-processing algorithms available within the database. The life science community is contributing greatly now in helping to define Oracle's commodity requirements for image processing. Focused improvements can be expected in the future.

With Oracle interMedia the Oracle database's scalability, security, and reliability are available for multimedia content. Both multimedia and relational data are managed in a single repository, allowing retrieval of images based on the full features of SQL.

3.1.2 Use of Oracle Extensibility by Independent Service Vendors (ISVs) and Customers for Biological Data Storage, Query, and Computation

As discussed in the section on enabling technologies Oracle is an object-relational database management system. Many ISVs and Oracle customers have taken advantage of the database features for object definition and storage to create database abstractions for sequences, structures, annotations, and other bio data. Furthermore Oracle's object-oriented extensibility permits the behaviors of the abstractions to be programmed as well. This powerful feature enables ISVs and in-house teams to offer domain products based on Oracle that have functional capabilities that are tightly integrated with the data. The externally programmed functional capabilities then become accessible to users and other applications via SQL or Java APIs.

Oracle extensibility also includes extensible indexing, a very useful feature for querying over complex scientific data types in large stores. Several life sciences vendors, including MDL and Daylight, have built chemoinformatic cartridges which use extensible indexing to do fast searching for structural subgraphs in large tables of small molecules. These proprietary indexes, whether based on hash keys (Daylight) or digital fingerprints (MDL), are fully registered and made active within the Oracle database via extensible indexing so that standard SQL queries can implicitly make use of them.

3.1.3 Oracle's Use of Extensibility in Bioinformatics

Fully consistent with Oracle's goal to serve the most reliable and efficient commodity data possible, its development cycle is mostly driven by the extension of the idea of "commodity". It is more than just storing data, or even structured data made possible by abstract data types. If there is a large data-driven application, Oracle's way of looking at it is to make the unstructured data structured within the database, and then to make the structured data smart at the service end. Many computational tasks that become standardized are best moved into the server code base. This computed data is then served as a database commodity, or enriches the versatility of queries. There are many examples of these application-driven code base extensions in Oracle, but one of the best is the Oracle Spatial offering. Oracle Spatial is an ORDBMS implementation of the international standard geometric feature type (the structuring). Oracle has built a rich operator set accessible from SQL or Java to go with this complex type (the smart). A large user base around the world can do lots of standard things with geometric-feature objects in Oracle - intersect them, find distances between them, measure their lengths and areas, transform them to new coordinates, and many more without going outside the database for the analytical tools. This rich collection of operators as well as a high speed spatial index in the server codebase make a very powerful product. Oracle has a similar philosophy for commodity data structuring and computation extension for bioinformatics, but it is at an earlier stage in a much larger problem.

Obviously to develop a major application-driven extension to the DB server the issue of what is a standard or a commodity is very important. This applies to the data dictionaries or ontologies of structured data as well as to the specifications for standard computations. The burgeoning complexity and quantity of life-sciences scientific data present such an immense challenge that uniform data standards (ontologies) for sequences, structures, gene-expression profiles, annotations, etc. are absent, with some exceptions. This is not the same as saying there is no format for the data. There are lots of formats, some with an old and awkward history. The same is true for standard computations.

In making its entry into this rich area, Oracle is focused initially in two areas : sequence searching and alignment, which together we will call sequence matching; and macromolecular structures.

In the case of sequence matching, it is the computational requirements that have become more or less standard. Although there is no single standard for the underlying data type, the linear DNA or polypeptide sequence, this is a fairly simple data type as things go with biological data. Oracle will be delivering a server-based implementation of the Basic Local Alignment and Search Tool (BLAST) [1] together with its variants BLASTP, etc. in a near release of the product. This sequence alignment capability will operate on sequences stored in the database as character large objects (CLOBs). It will be simple to move the services later to a proper ontology-based representation of sequences and annotations in the database, if suitable standards emerge for sequence objects. In the meanwhile Oracle is investigating data compression techniques for sequences so that we may one day do an efficient storage implementation of an emerging standard. Storage will remain critical and expensive because the volume of sequence data will be growing faster than Moore's law, especially since we are entering the era of proliferation of generated sequences horizontally across species and vertically over individuals. Oracle is also considering whether a full implementation of the Smith-Waterman [2] algorithm might prove ultimately more useful in the server than BLAST. BLAST has become such a commodity that there are already whole compute farms devoted to it. Further on the horizon Oracle is looking into fast ways of solving the general pattern matching problem for sequences, so that rapid searches could be done in the database for things like flexible promoter sequences, intron/exon boundaries, and statistical secondary structure predictors in the case of protein sequences. There have been no standard computational requirements to emerge for this generalized matching (although there are lots of software implementations for particular requirements), but this does not preclude R&D on powerful, general methods for matching sequence data that will be useful and efficient in the database for a lot of typical problems.

In the case of macromolecular structures, it is the data dictionary for the solved structure that has shown signs of standardization. The Protein Data Bank (PDB) [3] is the holy grail of public protein structure databases, but until recently it has only served data in a format that goes back almost fifty years to the first protein structures deposited at Brookhaven. This PDB flat-file format has a legacy in punched cards. The PDB is now supporting a specification for a structured data type for macromolecular structures based on a thorough scientific ontology called the Macromolecular Crystallographic Information File (mmCIF)[4]. This specification has received the blessing of the Object Management Group, Life Sciences committee, as a standard and has been served to users as an option by the PDB since August 2002. Oracle intends to do an efficient schema and object data type implementation of mmCIF and make it part of the DB server offering. Initially Oracle will provide a SQL and Java interface to the macromolecular structures in the database consistent with the low level accessor/depositor functions of the OMG standard. Over time Oracle will layer computations over the data structures as standard computational requirements emerge. Already Oracle has identified many common operators that are believed to be of wide and uniform applicability to molecular biology problems. In other words, Oracle will make commodities of the things researchers commonly want to ask about large molecules, whether explicitly captured in the data or derived by computation. With such a smart database implementation

of mmCIF, researchers will be able to load and utilize not just public molecular structures but also proprietary structures and those derived via homology, or ab initio protein folding, if the latter problem is ever solved.

3.2 Bioinformatics Data Mining and Statistical Modeling

There have been vast amounts of life sciences data generated to date and there continues to be an exponential increase in data volumes. This flood of data, often referred to as the "data tsunami" threatens to overwhelm life sciences professionals and the tools they employ to exploit the data. More than just the volume of data, the rich structure and connectivity of the data present an immense challenge. In such an environment data mining and statistical analysis can help to focus research interests and derive relevant information. Mining bioinformatics data can help discover relationships between gene expression, protein function, metabolic pathways, and disease states.

Oracle Data Mining provides five data mining algorithms that are embedded within the database server. By supporting data mining inside the database, the data stays securely in the database eliminating the extra step of moving the data outside the database for analysis. Oracle Data Mining provides two supervised learning techniques for classification and prediction: naive Bayes and decision trees (using adaptive Bayes networks) and three unsupervised learning algorithms: association rules, hierarchical k-means clustering, and clustering using the Oracle proprietary O-Cluster algorithm. In supervised learning, a target field or dependent variable is identified. The supervised-learning technique then sifts through data trying to find patterns and relationships between the independent variables and the dependent variable. One can use supervised learning to determine which single-nucleotide polymorphisms are implicated in disease states. In unsupervised learning, the objective is not indicated and clustering algorithms make no assumptions about the target field. Instead, data mining finds independent associations and clusters in the data. One can use unsupervised learning to identify new disease classes from patterns in gene expression data. Additionally, Oracle Data Mining supports “feature selection” through its Attribute Importance facility. Attribute Importance helps identify a proper subset of attributes that are most relevant to predicting the target. All Oracle Data Mining’s functionality is accessible via SQL and a Java-based API. Combined with other Oracle features such as Oracle Transparent Gateways and Oracle Workflow, it is possible to build a data mining application that automates the uploading of new data, the pre-processing, and the data mining of that information, creating in effect a virtual bio-analysis pipeline entirely within the database. Oracle Data Mining can be used to find relevant biomarkers such as differentially expressed genes, to classify biological samples, to suggest new drug targets, and to discover new knowledge in the scientific literature.

Oracle offers integrated online analytical processing (OLAP) for interactive drill-down and analysis of data, including predictive analytical functions such as modeling, forecasting, and scenario management. Oracle OLAP is fully integrated in the relational database together with all the data and metadata so that queries are processed efficiently.

Oracle provides several commonly used statistical functions so that third party tools are not needed for many common statistical analyses. Included in Oracle and available through SQL are the following: summary statistics, including mean, standard deviation, and variance; correlations; graphs, including scatter plots; and linear regression. Oracle is continuing to gather statistical processing requirements from the life sciences community and is increasing the scope of the statistical functions available in the database.

3.3 Biological Database Migration and Integration

Biological databases are traditionally stored in multiple files and in different file formats. There is a vast amount of existing data that is used by the current breed of bioinformatics applications. Migrating these huge volumes of disparate data to the database has been and will continue to be a challenging task. Oracle provides various tools to make this data migration as easy as possible.

3.3.1 Oracle Internet File System (iFS)

Oracle iFS provides a central and easy to use repository for managing all types of application data. With iFS, the data in the database can be accessed as if the data were in a file system. iFS understands HTTP, FTP, NFS and other standard file access protocols. For example, one can map a network drive or a web folder on an NT system to access data stored in Oracle. If an application is developed based on a file system, that application can be very easily ported to work with data in the database since iFS can be configured to present the same file system for the data in the database. Oracle iFS understands different file formats like simple text, XML, Word, and many other common file formats. This makes it easy to migrate the data to the database without losing the functionality of any of the existing applications.

3.3.2 External Tables

In some situations, an application built for a database may need to access data that is not yet in the database. In such situations, Oracle's External Tables functionality provides a table-like interface over the external data source. When this feature is used, the application thinks that the data is coming from the database, while in reality the data can by anywhere on the local host or in some files on a remote system over the internet.

Applications can access data in external sources as if it were in a table in the database. Applications can connect to the database and create metadata for the external table, using standard data definition language (DDL). The DDL for an external table consists of two parts: one part that describes the Oracle column types, another part (the access parameters) that describes the mapping of the external data to the Oracle data columns. An external table does not describe any data that is stored in the database, nor does it describe how data is stored in the external source. Instead, it describes how the external table layer needs to present the data to the server. It is the responsibility of the access driver and the external table layer to do the necessary transformations required on the data in the data file so that it matches the external table definition. External tables are read-only; therefore, no data manipulation operations are possible, and no index can be created on them. When an external table is defined, the table definition does not have to exactly match the file format. That is, the source file can contain more or fewer fields than there are columns in the external table. And the data types for fields in the data source can be different from the columns in the external table. The access driver ensures that data from the data source is processed so that it matches the definition of the external table.

3.3.3 SQL Loader

When there is a large amount of data in external files to be moved into an Oracle database, SQL Loader provides all the required functionality to map the data format in the files to the tables in the database and load the data. It has a powerful data parsing engine that puts little limitation on the format of the data in the external file.

SQL Loader can load data from a single file to multiple tables or data from multiple files into a single table. It can load data from standard operating system files, tape, or a named pipe. This makes it easy to load data from disparate sources as long as the data source can be presented as a named pipe within the operating system. For example, in bioinformatics applications a large amount of data is produced by automated systems. Instead of first storing this data in an operating system file, this data can be directly loaded into the database using this named pipe concept. SQL Loader is flexible in how the file to be read should be formatted. It can read both fixed record formatted files and variable record formatted files. In addition, the files can also be formatted like streams. As long as each record in the stream has a prespecified delimiter, SQL Loader can read the data from the stream and load it into the database.

3.3.4 Transportable Tablespaces

Transportable tablespaces provide the fastest way of moving data between two Oracle databases, if both the databases are on the same hardware platform. Moving data by transporting tablespaces can be orders of magnitude faster than either export/import or unload/load of the same data, because transporting a tablespace involves only copying datafiles and integrating the tablespace metadata. When table data is transported using this method, the corresponding index data can also be moved with the table data. This way, the index does not have to be rebuilt after importing or loading the table data.

A transportable tablespace lets applications move a subset of an Oracle database from one Oracle database to another on the same platform. Applications can clone a tablespace and plug it into another database, copy the tablespace between databases, or unplug a tablespace from one Oracle database and plug it into another Oracle database. To move or copy a set of tablespaces, one must set the tablespaces to read-only, copy the datafiles of these tablespaces, and use export/import to move the database information (metadata) stored in the data dictionary. Both the datafiles and the metadata export file must be copied to the target database. The transport of these files can be done using any facility for copying flat files, such as the operating system copying facility, ftp, or through offline storage such as CDs.

3.3.5 Heterogeneous Distributed Database Systems

In a heterogeneous distributed database system, at least one of the databases is a non-Oracle system. To an Oracle application, the heterogeneous distributed database system appears as a single, local, Oracle database. The local Oracle database server hides the distribution and heterogeneity of the data. For each non-Oracle system that an application accesses, Heterogeneous Services can use a transparent gateway agent to interface with the specified non-Oracle system. The agent is specific to the non-Oracle system, so each type of system requires a different agent. The transparent gateway agent facilitates communication between Oracle and non-Oracle databases and uses the Heterogeneous Services component in the Oracle database server. The agent executes SQL and transactional requests at the non-Oracle system on behalf of the Oracle database server.

Generic Connectivity enables applications to connect to non-Oracle data stores by using either a Heterogeneous Services ODBC agent or a Heterogeneous Services OLE DB agent. Any data source compatible with the ODBC or OLE DB standards can be accessed using a generic connectivity agent. The advantage to generic connectivity is that it may not be required for applications to acquire and configure a separate system-specific agent. Applications use an ODBC or OLE DB driver that can interface with the agent.

3.4 Data Intensive XML and Oracle XDB

Oracle XDB is built upon the draft standard for XML Schema 1.0, a language used to define the structure of XML data. XDB allows the database to leverage its existing data management capabilities on any data that can be represented by an XML schema. This way, one benefits from not only the flexibility that XML affords over SQL and Java objects, but also the power and functionality of an object-relational database as the storage mechanism. Unstructured data and non-schema-based XML are also supported; however, for these documents, XDB loses some of the query benefits provided by the underlying database.

Using a new SQL data type called XMLTYPE, XDB provides an abstraction layer for XML data above the existing physical storage techniques. XDB parses XML documents into their individual components and stores their elements in SQL object attributes and relational columns. It can also use LOBs to maintain data in its original text format, or it can mix LOB and object-relational storage for different elements within the same document. The particulars of the storage mapping, or how relational data is (de)normalized across tables, are hidden under the XML abstraction.

3.4.1 Internet Application Programming with XDB

The main advantage of programming with XDB is that all of its functionality is accessible by standard Java APIs. Today, the most popular methods for building web applications are Java Server Pages(JSPs) and XSL/XSPs (XML Stylesheets / XML Server Pages). JSPs are designed to access data via Java Beans, and XSL/XSPs are designed to access data via a DOM (Document Object Model) API implementation. XDB supports both styles of APIs, each of which has its own advantages and disadvantages. Java Beans are compiled objects, and require the application programmer to know the structure of the data in advance. XDB Beans are created when the schema is registered, and compile with knowledge of the data structure. Beans provide for the fastest, easiest access to XMLTYPE data.

3.4.2 XDB Data Access

The XPath standard provides the access abstraction on top of the storage model. XPath traverses nested XML elements by specifying the elements to navigate through with a slash separated list of element and attribute names. XDB supports a pathname format for identifying Internet data. Finding XML data via a pathname allows access via the widest variety of protocols (such as those supported by Oracle's Internet File System). XDB uses pathnames to locate XML schema files internally, and many XML standards (XInclude, XLink, etc.) assume support for pathname-style URLs to locate data in the database.

3.5 Web Services and Portals

Members of the bioinformatics community often need to integrate data and services from different sources to build a complete solution. For example, in a drug discovery application, data comes from different processes where each process is executed at a remote site. To enable these applications, the system needs to support a standards-based infrastructure that enables companies and their enterprise applications to communicate with other companies and their applications more efficiently. These standards should allow discrete business processes to expose and describe themselves on the Internet, allow other services to locate them, to invoke them once they have been located, and to provide a predictable response. Oracle Web Services provide this infrastructure for building and deploying such applications that share data and services over the internet.

Web Services consist of a set of messaging protocols, programming standards, and network registration and discovery facilities that expose business functions to authorized parties over the Internet from any web-connected device. A Web Service is a discrete business process that does the following: (i) A Web Service defines its functionality and attributes so that other applications can understand it. A Web Service makes this functionality available to other applications. (ii) A Web Service can be registered in an electronic Yellow Pages, so that applications can easily locate it. (iii) Once a Web Service has been located and examined, the remote application can invoke the service using an Internet standard protocol (iv) When a Web Service is invoked, the results are passed back to the requesting application over the same Internet standard protocol that is used to invoke the service.

Oracle iAS Portal is a browser-based application for building and deploying e-business portals. It provides a complete set of open, productive, self-service tools for publishing information, building applications, and deploying and administering enterprise portal environment. For example, in life sciences business, document management is a major application. Most companies in this business have a vast library of useful documents and applications that are essential for everyday tasks. It can be a challenge to locate just the document one needs. And to return to it once it is found and to keep on top of critical changes to it is ever more difficult. Oracle iAS Portal helps to meet all of these challenges. For locating essential information, Portal provides a search feature that enables users to look for information according to the ways it has been classified (via categories and perspectives), or against terms available in its content, its author, or a number of other criteria. In addition, these portals are easily customizable so that a users can add their own search preferences or subject areas. Oracle iAS Portal provides an assortment of building blocks for creating content-rich portals. These tools make it easy to build and deploy portals in an enterprise without the need to do programming.

4 References

[1] Altschul, S.F., W. Gish, W. Miller, E.W. Myers, D.J. Lipman (1990): Basic Local Alignment Search Tool. *J. Mol. Biol.*, **215**, 403-410.

[2] Smith, T.F., M.S. Waterman (1981): Identification of Common Molecular Subsequences. *J. Mol. Biol*., **147**, 195-197.

[3] Bernstein, F.C., T.F. Koetzle, G.J. Williams, E.E. Meyer, M.D. Brice, J.R. Rodgers, O. Kennard, T. Shimanouchi (1977): The Protein Data Bank: A Computer-based Archival File for Macromolecular Structures. *J. Mol. Biol*., **112**, 535-542.

[4] Bourne, P.E., H.M. Berman, B. McMahon, K. Westbrook, P.M.D. Fitzgerald (1997): The Macromolecular CIF Dictionary (mmCIF). *Methods in Enzymology*, **277**, 571-590.