Web-based Molecular Modeling Using Java/Swarm, J2EE and RDBMS Technologies

Yingping Huang, Gregory Madey and Xiaorong Xiang
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556

Steve Cabaniss
Department of Chemistry
University of New Mexico
Albuquerque, NM 87131

March 1, 2003

Abstract

This paper describes the application of Java/Swarm, J2EE and RDBMS technologies to the area of web-based scientific simulation. One such scientific simulation is on the behavior of natural organic matter (NOM). NOM is a heterogeneous mixture of organic molecules found in terrestrial and aquatic environments - from forest soils and streams to coastal rivers and marshes to the open sea. NOM plays a vital role in ecological and biogeochemical processes. In this paper, we present an agent-based stochastic simulation of NOM transformations, including biological and non-biological reactions, as well as adsorption and physical transport. It employs recent advances in web-based development environments such as Java 2 Enterprise Edition (J2EE), and scalable web-based relational database management systems(RDBMS), to improve the reliability and scalability of the stochastic simulations and to facilitate analysis and data mining of the resulting large datasets. The NOM simulation system may be useful in many areas including chemistry, geology, microbial ecology and environmental science.

1 Introduction

Natural Organic Matter (NOM) is a complex mixture of compounds formed as a result of the breakdown of animal and plant material in the environment. The composition of the mixture is strongly dependent on the environmental source. Generalizations regarding chemical character can be prone to misinterpretation as the character of the compounds present in the mixture is extremely diverse. However, if the complexity of the NOM is kept in mind, a broad understanding of its character is possible. NOM consists mainly of carbon, oxygen and hydrogen. Nitrogen, phosphorus and sulphur can also be present; their prevalence will depend on the source of the NOM. A range of compounds, from small hydrophilic acids, proteins and amino acids to large humic and fulvic acids, are constituents of most NOM. The organic compounds can range from largely aliphatic to highly aromatic, from molecular weights around 10,000 down to 2000, and from highly charged to uncharged compounds.

NOM is ubiquitous in terrestrial, aquatic, and marine ecosystems, playing a crucial role in such important processes as the evolution and fertility of soils; the mobility and transport of pollutants such as trace metals, radionuclides and hydrophobic organic compounds; the availability of nutrients to microorganisms and plant communities; the growth and dissolution of minerals; and the global biogeochemical cycling of the elements [1].

NOM is very important to drinking water systems. It has a significant impact on all aspects of drinking water treatment. NOM is responsible for the majority of the coagulant demand. Therefore waters with high dissolved organic carbon levels usually have a high coagulant requirement and consequent high treatment costs. Where activated carbon is used for the removal of micro-contaminants, e.g., tastes and odors, pesticides, NOM competes strongly for adsorption sites, and the amount of activated carbon required is increased or the lifetime of the carbon is drastically reduced. NOM exerts a high demand for chemicals, such as chlorine, cholramine and ozone, thus increasing the costs associated with their application. In addition, the reaction between oxidants and NOM can produce by-products, some of which are known to be harmful to health at high concentrations. NOM is responsible for fouling of membranes, reducing the flux, resulting in higher frequency of back-washing and cleaning membrane systems. NOM serves as a substrate for bacterial growth. This may lead to re-growth in the distribution system where a sufficient disinfectant residual cannot be maintained.

The importance of NOM attracts numerous researchers, including chemists, geologists or even computer scientists. Despite decades of research, we still know relatively little about the structure, chemical composition, and chemical properties such as molec-

ular weight, functional group concentrations, structure, composition, and reactivity [2] [3] [4].

During the last 50 years, some research has been done on the functional behavior of NOMs, including pH buffering, metal complexation, pollutant solubilization, adsorption onto mineral surfaces, alteration of mineral precipitation and dissolution, bioavailability to heterotrophs and promotion of photochemical reactions [5] [6] [7]. Different models are proposed to handle NOM research, including ODE (Ordinal Differential Equation), PDE (Partial Differential Equation), etc. But the diversity of the compounds present in NOM leads to the difficulty of describing the mixture adequately and makes models computationally expensive.

2 Previous Work

Over the recent years, many simulation models have been developed to simulate biochemical reactions. These models can be categorized as deterministic or stochastic.

2.1 Deterministic Models

Several simulators are available which can be used to predict the behavior of a network of chemical reactions. Among them, SCAMP [8] and METASIM [9], employ non-standard biochemical languages to specify a model in a series of command files. The simulator then uses these files to construct differential equations which are solved using numerical integration. Other deterministic models are also proposed with more specific programs to solve a single set of differential equations using similar methods [10] [11].

Under some situations, deterministic models no longer represent the physical system adequately and cannot be used to predict the concentrations of chemical species.

2.2 Stochastic Models

In 1976, Gillespie [12] developed a discrete, stochastic algorithm to simulate chemical reactions. In his model, time is quantitized into small time steps with variable lengths. At each time step, based on the rate constants and population size of each chemical species, a random number is generated to choose which reaction will occur, and another random number to determine how long the step will last. The chemical populations are altered according to the stoichiometry of the reaction and the process is repeated. To determine which chemical reaction will occur in a given time step, the Gillespie algorithm calculates the probability of each reaction relative to another by multiplying the rate

constant of each reaction with the concentration of its substrates. A random number is then used to determine which reaction will occur according to this set of weighted probabilities.

In 1995, IBM Almaden Research Center developed a Chemical Kinetics Simulator (CKS) [13]. This stochastic simulation package allows you to calculate concentrations of reactants and products in a chemical system as a function of time.

In 1997, Punch et al reported a software system called BESS which simulated the action of biodegradation pathways on compounds. It did so by encoding biodegradation pathways in a knowledge base and applying those pathways in sequence to the compound, breaking it down into metabolites [14]. BESS used the expert system approach. As its heart, BESS was a rule-based system. It was implemented using VisualWorks Smalltalk. It was then re-implemented using C++ and Java. Firth and Bray [15] [16] later developed a simulation called STOCHSIM using a stochastic model to predict cell signalling pathways. In their model, time is quantitized into discrete slices. At each time step, one or two molecules are selected from the set of molecules depending on some pre-calculated probabilities. Then first-order or second-order reactions could occur based on some generated random numbers. Note that at each time step, only one chemical reaction is allowed.

Swarm, a library implementing agent-based technology, has been used in chemical simulations. In [17], McMullin implemented an artificial chemistry called SCL using Swarm. SCL involves three distinct chemical species: Substrate, Catalyst and Link, hence SCL. SCL models the emergence and stability of autopoietic organization.

Alur at el proposed a hybrid model in which discrete events are combined with continuous differential equations to capture switching behavior [18]. They developed a programming language called Charon to model and analyze their bybrid model.

3 Agent-based Stochastic Model

In this paper, we develop a new stochastic simulation model using agent-based technology. This stochastic model of NOM represents individual molecules as discrete objects of specified elemental and functional group composition, size and reactivity. Temporal evolution of NOM from biological precursor compounds such as lignin, polysaccharides and proteins is simulated in which specific probabilities are assigned to particular transformations. The reactivity of the resulting NOM assemblage over time can be predicted based on the distributions of molecular properties.

This stochastic approach has several advantages: it is much less computationally intensive than molecular modeling or explicit kinetic simulation of hundreds of compounds,

it can be adapted to a variety of time scales and processes, and it intrincically handles NOM structural and functional heterogeneity. This approach also employs state-of-the-art information technology including web interface and scalable relational database management systems (RDBMS).

3.1 The Simulation Algorithm

The NOM simulation is implemented in a discrete 2D space with discrete time. The simulated space is a rectangular lattice. Each molecule can occupy at most one cell, and each cell can host at most one molecule. The molecules in the simulation are intended to represent a sample from a large population in the system under study. During execution of the simulation, each molecule may move to another point in the lattice or stay in a fixed position according to predefined simple rules that describe physical processes. In chemical reactions, one molecule could split and occupy two cells in the world; two or more molecules could combine and occupy just one cell. In the situation of adsorption, two molecules can be in the same cell if one of them is adsorbed.

There are 10 types of chemical reactions represented in the simulation system: ester condensation, ester hydrolysis, amine hydrolysis, microbe uptake, dehydration, strong C=C oxidation, mild C=C oxidation, Alcohol (C-O-H) oxidation, aldehyde C=O oxidation, and decarboxylation. Each molecule has a probability for each type of chemical reaction. The calculation of the probability is based on the structure of the molecule, and the environment in which the molecule resides. These environmental variables include the length of the time step, microbe density, fungal density, pH value, temperature, pKw (the equilibrium constant for the autolysis of water, which is very close to 14.0), oxygen density, light density, etc. After each chemical reaction, the probabilities of these reaction types are re-calculated. The reaction probabilities are stored in an array associated with each molecule.

In each time step, for each molecule, a random number is generated which is used to determine whether a chemical reaction will occur, and if one occurs, which reaction type. In the simulation, the sum of all the reaction probabilities is controlled to be less than 1 percent. The interval $[0, 1]$, is partitioned into 11 subintervals. The length of the first interval is equal to the probability of the first reaction type; the length of the second interval is equal to the probability of the second reaction type, etc. The length of the last interval is the probability in which no reaction will occur. The generated random number from the interval [0,1) will reside in one of these intervals, and it will decide which chemical reaction will occur, if any at all.

If the chosen chemical reaction type is a second order reaction, i.e., there will be two

molecules involved in the reaction, the second molecule will be chosen from one of its nearest neighbors who are not yet involved in a chemical reaction.

After the reaction takes place, the probability tables for involved molecules are updated at the end of the time step and the new probability tables are assigned to newly produced molecules. Note that, at each time step, one or more chemical reactions could occur.

4 Simulation Technologies

In the simulation described in this paper, multiple information technologies are integrated to investigate a new paradigm of scientific inquiry. Java is reaching performance parity with other languages (e.g., c, c++) and has begun to be used in scientific simulations[19] [20] [21]. Fox examined the paradigms of scientific study and introduced a fourth paradigm[22]. We used to speak of three approaches to science: experiment, theory and the computational approach. The fourth paradigm of scientific study is to use IT technologies such as web, databases and data mining. In this paper, we present the molecular simulation model using these IT technologies.

The NOM simulation system includes a web interface which allows users to provide input for the core simulation and to immediately view status reports of their simulation through their browsers (described in section 5.3). The web interface is written using JavaServer pages (JSP), and the core simulation is invoked by a Java servlet. To run JSP and Java servlets, we use the Java 2 Enterprise Edition (J2EE) platform. Among many J2EE platforms such as Apache Tomcat, Sun's JavaServer Web Development Kit (JSWDK), Java Web Server, Orion Server, and Oracle Application Server, we choose the Oracle9iAS container for J2EE (OC4J, also known as Orion Server) which is part of the Oracle Application Server Suite. The core simulation is written in Java with the Swarm library. The simulation data is stored in Oracle RDBMS using Java Database Connection (JDBC). Reports of the simulation results are generated upon user request, using SQL and PL/SQL language with the help of Oracle Reports. In the following section, we briefly introduce these technologies.

- Oracle RDBMS, Oracle Data Warehousing and Data Mining: According to Oracle, a database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a server reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance.

We use both Oracle8i and Oracle9i in the simulation system. For detailed introduction of Oracle database, SQL and Oracle PL/SQL languages, see Loney and Koch [23].

Oracle9i Data Mining embeds data mining functionality into the Oracle9i database, for making classifications, predictions, and associations. All model-building, scoring and metadata management operations are initiated via a Java-based API and occur entirely within the relational database.

- Oracle Reports: Oracle9i Reports is Oracle's reporting tool. It consists of Oracle9i Reports Developer and Oracle9iAS Reports Services. Oracle9i Reports Developer helps user to rapidly create web and paper reports against the Oracle databases. Oracle9iAS Reports Services publishes these reports in any format including HTML, XML, JSP to web browser and Oracle9iAS Portal. In this paper, we use Oracle Reports to publish reports in JSP format on the web so that users can view the reports through their browsers. Additional information on Oracle Reports is available in Muller [24].

- JDBC: JDBC (Java Database Connectivity) is a standard Java interface for connecting from Java to relational databases such as Oracle. In addition to supporting the standard JDBC API, Oracle drivers have extensions to support Oracle-specific datatypes and to enhance performance. In this paper, JDBC is used by the core simulation to store simulation data to the Oracle databases. To learn more about Oracle JDBC, refer to Price and Wald [25].

- J2EE: The J2EE architecture is based on the Java programming language. An advantage of Java is that it enables us to write code once and deploy that code onto any platform. The process is as follows: developers write source code in Java, the Java code is compiled into bytecode, which is a cross-platform intermediary, halfway between source code and machine language. When the code is ready to run, the Java Runtime Environment (JRE) interprets this bytecode and executes it. J2EE components are transformed into bytecode and executed by a JRE at runtime. J2EE has historically been an architecture for building server-side deployments in the Java programming language. It can be used to build traditional web sites, software components, or packaged applications. For an introduction of J2EE, see Deitel et al [26].

- OC4J/Orion Server: Oracle provides a complete J2EE container written entirely in Java that executes on the Java virtual machine (JVM) of the standard Java

Development Kit (JDK). You can run the Oracle9iAS containers for J2EE (OC4J) on the standard JDK that exists on most operating systems. OC4J is based on technology licensed from Ironflare Corporation, which develops the Orion Server. OC4J supports the standard J2EE APIs including Java Servlets, JavaServer Pages (JSP), Enterprise JavaBeans (EJB), Java Database Connectivity Services (JDBC), Java Naming and Directory Interface (JNDI), Java Transaction API (JTA), Java Message Service (JMS), etc.

5 Infrastructure

The simulation model employs recent advances in web-based interfaces, and scalable web-based database management systems to improve the reliability of the stochastic simulations and to facilitate analysis of the resulting large datasets. In the previous section, we briefly introduced the technologies we used in the simulation. In this section, we describe the infrastructure of the simulation system using those technologies. Here infrastructure means the group of database servers, web servers, J2EE platform, reports servers and data mining servers, as well as the underlying software which supports our simulation.

The multi-tier infrastructure of the simulation system is shown in Figure 1. Before explaining the infrastructure in detail, let us show how users access the simulation system.

As shown in Figure 1, we have N application servers on which simulation is invoked and running. We also have M database servers which can store simulation data. It is easy to add more application servers and database servers with little change of code. To effectively use these resources, we implemented such features as load balancing and fail over. Load balancing enables us to distribute simulations evenly over application servers and database servers. Fail over enables us to continue the simulation on another resource without terminating it.

To run a simulation successfully, we need one application server and one database server. Before a simulation is running, an application server and a database server are assigned to the simulation according to the load of all application servers and database servers. Therefore, when multiple simulations are executing, the resources of application servers and database servers are evenly utilized.

To enable fail over, we store the state of the simulation system on databases. When an application server is down accidently, a new session will startup to resume the simulation

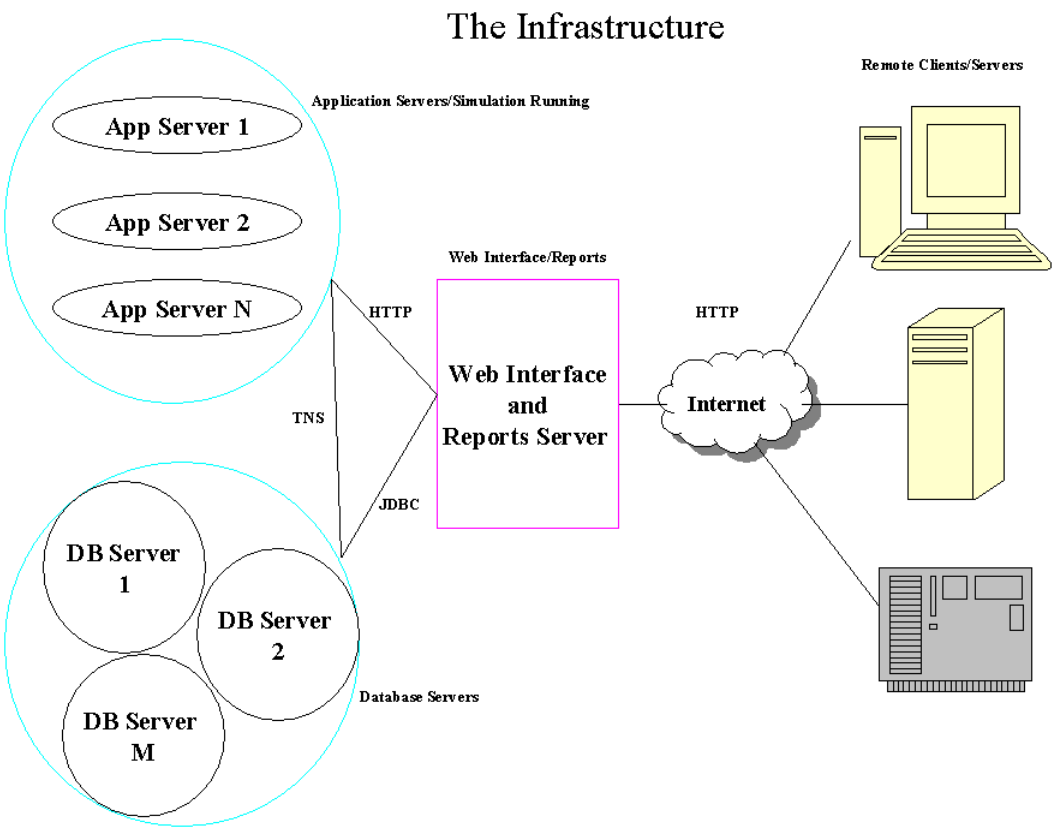

Figure 1: The infrastructure

by reading the system state to initialize the new simulation. If a database server is down accidently, a SQLException is caught and new simulation data will be stored in another database. To facilitate simulation status reports, we replicate the important aggregation tables which are discussed on the query optimization section.

6 Core Simulation Engine and Web Interface

The NOM simulation is implemented using Java, JDBC and the Swarm library. The simulation can run either in batch mode or in graphical user interface (GUI) mode. Since our main purpose is to let the users invoke their simulations through the web and obtain reports of their simulations, the GUI mode is not available for them, i.e., the user will not see how molecules move and react, but the state of the simulation for each time step is completely stored in the Oracle databases. The users can get the same information such as the trajectory of a molecule from the simulation databases.

The NOM simulation system involves various NOM molecules. These molecules could be macro-molecules or micro-molecules. The macro-molecules could be polysaccharides, protein, polynucleotide, tannin, lignin structure, polyterpene and cutin. The micro-molecules may consist of phospholipids, sugars, amino acids, flavonoids and quinnones.

To represent each type of molecules, a Java class called Molecule is created. Each molecule in the simulation is represented as numbers of different atoms and functional groups, and its position in the simulated world. More precisely, each instance of Molecule has the following attributes: number of C, N, O, H, S and P atoms; number of such functional groups as phenyl groups, alcohols, phenols, ethers, esters, ketones, aldehydes, acids, aryl acid, amines, ring N, amides, thioethers, thiols, phosphoesters, H-phosphoesters, phosphates; x, y coordinates of the molecule in the simulated world which is a 2-dimensional lattice; and some other attributes such as probabilities of chemical reactions, etc.

Overall, the reactions and processes modeled are:

- Physical reactions: adsorption to mineral phases, aggregation, formation of a micelle, transport downstream, transport through porous media and volatilization. Note that there could be chemical reactions within physical reactions.

- Chemical reactions: chemical reactions could consists of abiotic bulk reactions and abiotic surface reactions, direct photochemical reactions, indirect photochemical reactions, extracellular enzyme reactions and microbial uptake of small molecules. Note that these chemical reactions will definitely change the attributes of the molecules.

The environment of the simulated world in which the molecules reside will affect the reactions and processes of the molecules. The molecules can also affect the environment, for example, increase or decrease microbial density, fungal density or pH value. In our simulation, the environmental variables include simulation time step, microbe density, fungal density, pH value, temperature, pKw, oxygen density, and light density.

6.1 Structure of the Core Simulation Engine

Figure 2 shows the Unified Model Language (UML) class diagram that defines the program structure. Only some of the core classes are listed in the class diagram.

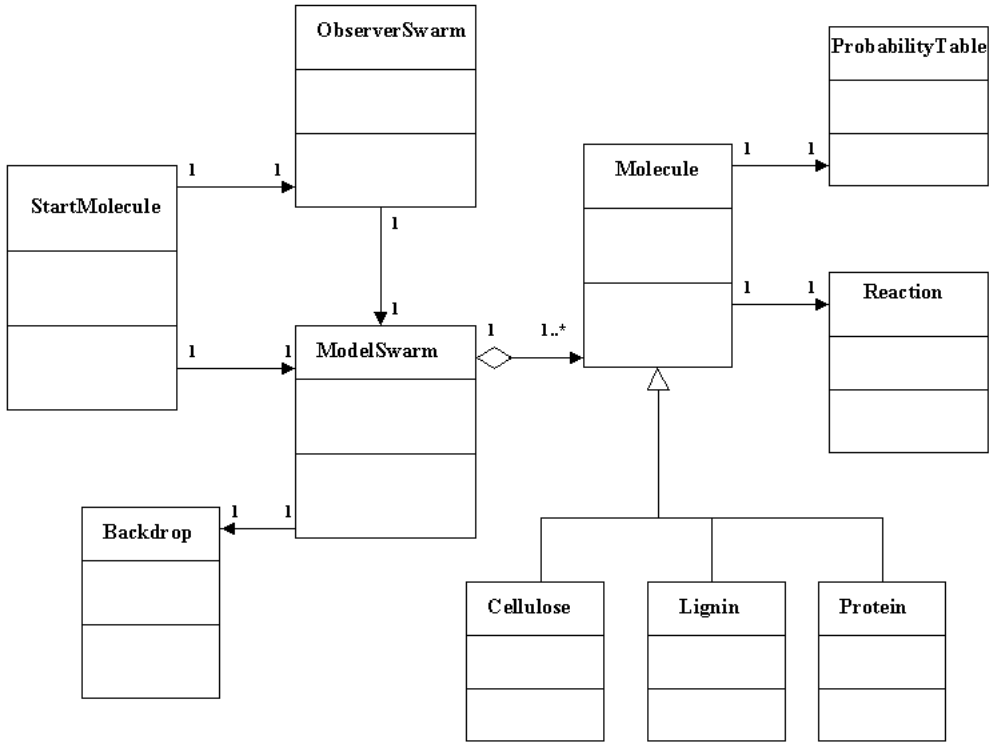

Figure 2: The class diagram

Figure 3 shows the UML use case diagram of the program. Here is a scenario that a user invokes the simulation either in GUI mode or batch mode. The observerSwarm updates the GUI and probe (which contains instance variables that describe the referent's class and type), the modelSwarm executes the scheduled simulation, updates each molecule, updates the simulated world, and writes simulation data to the database.

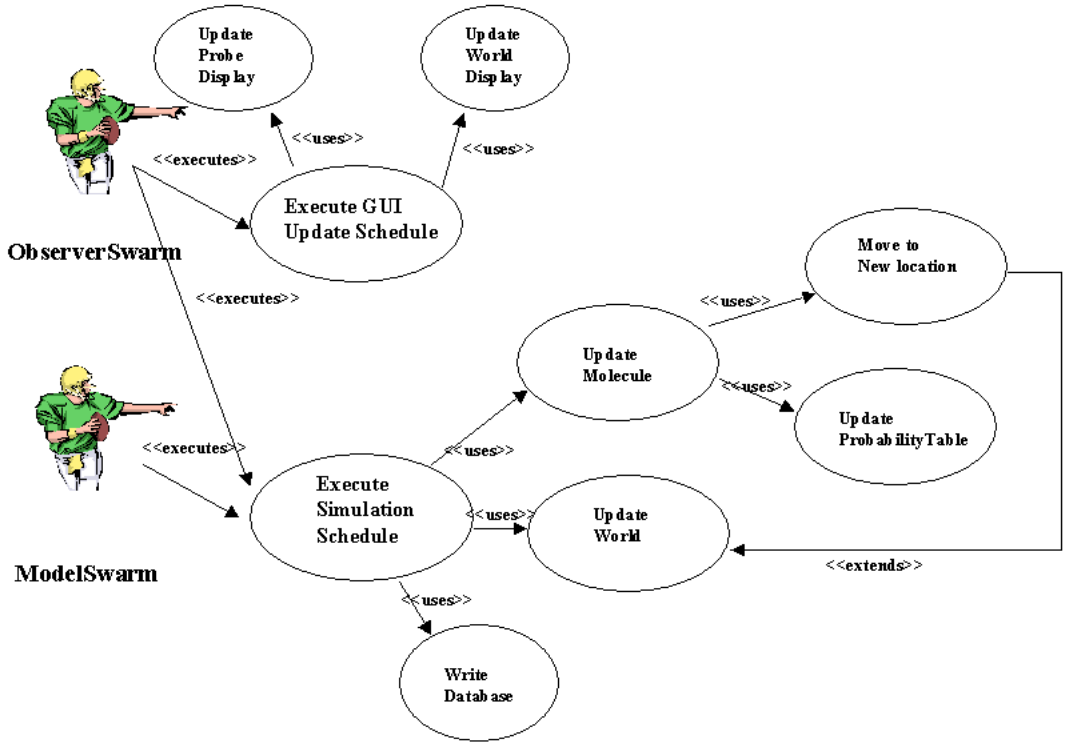

Figure 3: The use case diagram

6.2 Web Interface

The web interface allows users to access the simulation system through web browsers such as Netscape and IE. It contains a Molecule Editor which receives user input and stores

the input in the database. registering with the simulation system, the user has a unique user_id, provides a record of environment variables and a set of molecule information. Figure 4 shows the UML diagram that defines the users information database.

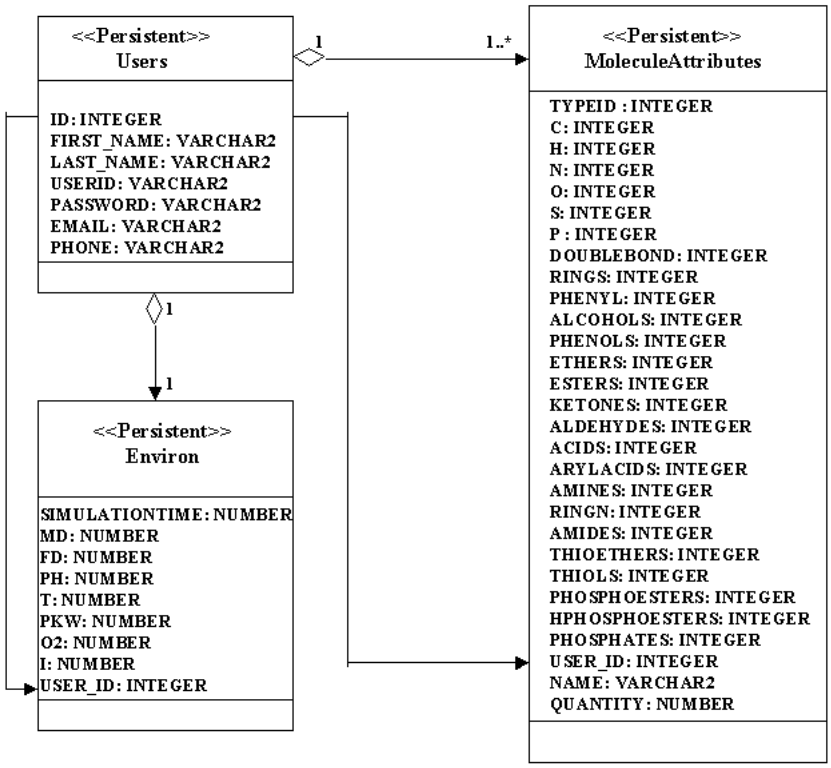

Figure 4: UML class diagram for the users information database

From the diagram, we see that there are three tables, namely, USERS, ENVIRON and MOLECULEATTRIBUTES. The table USERS is the owner of both ENVIRON and MOLECULEATTRIBUTES. The attribute ID is the primary key of USERS. The foreign key USER_ID in ENVIRON and MOLECULEATTRIBUTES references the primary key of USERS. The table ENVIRON has USER_ID as its primary key, while the table MOLECULEATTRIBUTES has TYPEID as its primary key. Oracle provides the ability to declare a *sequence*, which is an object that generates unique integers in sequence. The

user information database uses two sequences USER_ID and TYPE_ID to generate the ID in USERS and the TYPEID in MOLECULEATTRIBUTES when new records are inserted into the tables.

The web interface uses the three tables to store user information. The users provide inputs for their simulations through the interface. Then their simulations are invoked through Java Servlets. Reports of their simulations can be viewed from the web. Figure 5 shows the chemical reactions statistics of the simulation.

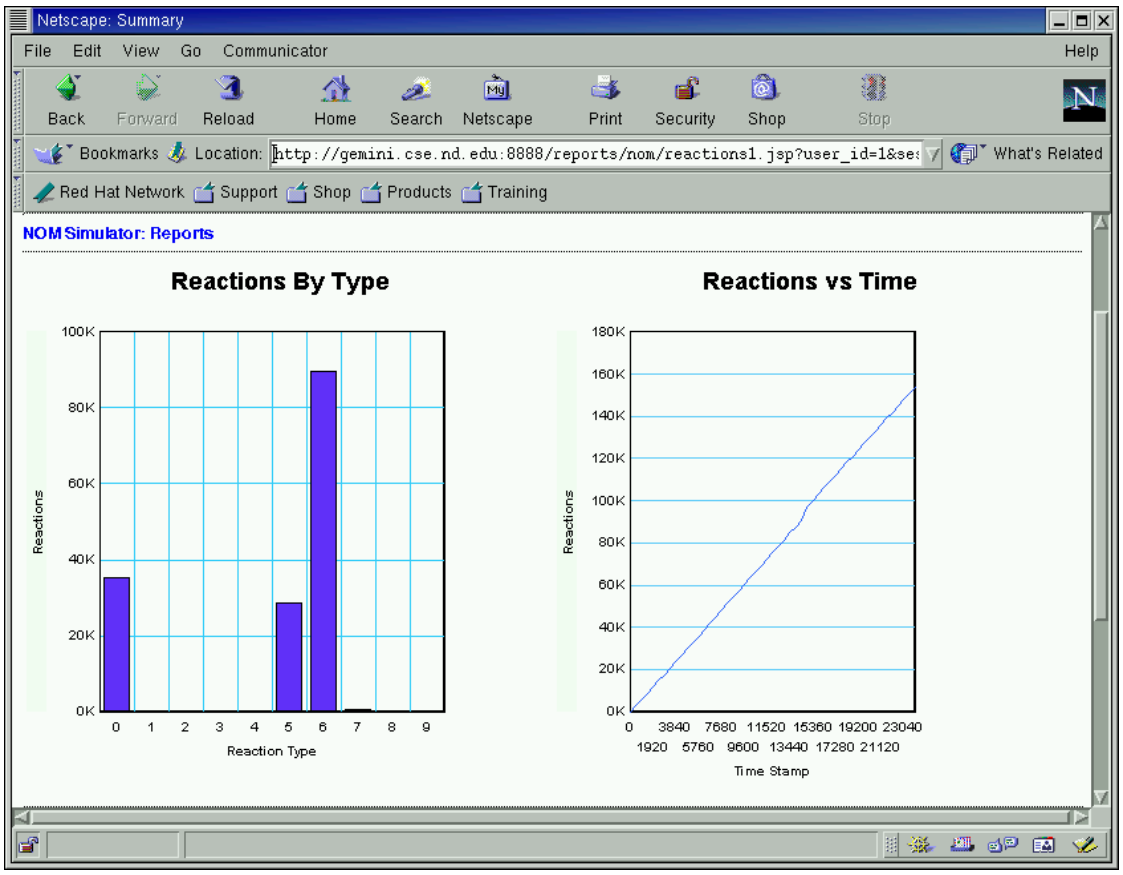

Figure 5: Chemical reactions statistics

7 Query Optimization

All the activities of the simultions are recorded in the databases. It is a critical task to both insert and retrieve information from the huge database efficiently. According to

our measurements, 100 records are inserted into the database by each simulation every second. It is not a trivial task to query the database efficiently while preserving the insertion performance. In this section, we define query optimization to include both insertion optimization and query optimization. Here we briefly discuss techniques to optimize insertion performance and query performance.

7.1 Database Design to Optimize Insertion Performance

We can store all information about the simulation system at any time step. (Recall, we use discrete time to simulation the continuous time.) At any time step, there are many molecules in the system. We may need to record every movement and reaction of these molecules. For example, the location of the molecule in the system, whether the molecule is involved in chemical reactions, and what new molecules are produced during a chemical reaction.

We discuss some parameters in the SQL statement "create table NOM." where NOM is the table to record state information of the simulations. These parameters describe aspects of managing space in data blocks. Our goal is to adjust these parameters to efficiently utilize the disk space and meanwhile, to reduce the probability of "chaining" (one row of a table spans two data blocks) which is a huge impact on query performance since it increases the amount of I/O. The size of a data block is specified at database creation, it is a multiple of OS block size which is normally 512 bytes. The PCTFREE, PCTUSED, INITRANS and MAXTRANS parameters are physical attributes that can be specified when a table is created or altered. These parameters allow you to control the use of the free space within a data block. This free space is available for inserts and updates of rows of data. For best insertion performance, we intentionally disabled constraints and indexes.

7.2 Query Optimization Case Studies

The NOM table records every action of all molecules at each time step. It's critical to retrieve information from the table and build reports in a efficient way. One of the major interests of our users is the statistics of the simulation, such as statistics for chemical reactions. We call these type of queries the **aggregation queries**. For illustration purposes, we provide two simple examples here and show how to generate fast reports for the two examples.

1. Show the number of chemical reactions for each of the ten reaction types occurred so far in the simulation in bar chart.

2. Create a line graph which shows the trend of the total number of chemical reactions vs time steps.

Figure 5 shows the graphs for the two reports, generated using Oracle Reports. Since it is a web based application, it is important to generate reports efficiently (in a few seconds), otherwise, users will not wait for the reports. The query statements for the two examples against the table NOM is quite complex, and execution of the query statement is very time-consuming. To get around this, we use a technique called "temporary aggregation tables".

The idea of aggregation is pretty simple. We create temporary tables which store the statistics of the system at the end of each time step. Therefore, we will have the information available whenever a query needs it, and bypass computations in the query. This approach will slow down insertion a little, but the gain of query performance is huge.

For the two queries, we create two tables, namely, REACTIONS_BY_TYPE and REACTIONS_BY_TIME. Table 1 and Table 2 show the structures of the two tables. In the REACTIONS_BY_TYPE table, REACTIONS denotes the number of chemical reactions occurred so far in the system for the particular RTYPE (reaction type). In the REACTIONS_BY_TIME table, TOTAL denotes the total number of chemical reactions occurred so far at this TIMESTAMP (time step). As can be seen, our reports will just query the two tables, instead of the huge NOM table.

Table 1: The REACTIONS_BY_TYPE Table

Name	Null?	Type
SESSION_ID	NOT NULL	NUMBER(38)
RTYPE		NUMBER(38)
REACTIONS		NUMBER(38)

Table 2: The REACTIONS_BY_TIME Table

Name	Null?	Type
SESSION_ID	NOT NULL	NUMBER(38)
TIMESTAMP		NUMBER(38)
TOTAL		NUMBER(38)

Next we show how to populate the two tables. In the core simulation program, we need to update or insert records into the tables at the end of each time step. In

the REACTIONS_BY_TYPE table, for a particular SESSION_ID, which identifies this session, at the end of the first time step, a new record is inserted into the table. At the end of other time steps, this record is updated. This record stores the number of chemical reactions occurred so far for each reaction type. In the REACTIONS_BY_TIME table, for a particular SESSION_ID, at the end of each time step, a new record is inserted into the table, which stores the total number of chemical reactions occurred so far in the system. As can be seen, the query statements for the two reports are simplified to the following two statements:

- select rtype "Reaction Type", reactions "Reactions"
  from reactions_by_type where session_id=:session_id;

- select timestamp "Time Step", total "Total Reactions"
  from reactions_by_time where session_id=:session_id;

The time to execute the two queries is reduced to 0.01 second and 5.26 seconds respectively. The two queries form the "Data Model" in Oracle Reports. To transform the data model to bar chart and line graph as shown in Figure 5, a tool called "Oracle Graphics" is used to generate the graphs. Since there are too many records (over 24 thousands) in the second query, it is hard to plot so many points (one point for one record) in the line graph (where stamp is the x-axis and total is the y-axis). To get around this, we only need to plot a small portion of sample points from the query. For example, we may just want to plot around 50 points uniformly selected from all the records. Since the table REACTIONS_BY_TIME is growing all the time, we need to know how many records are in the table when we uniformly choose sample points. The following simple method shows how to generate the sample points.

1. To simplify the query, another table with same structure of the REACTIONS_BY_TIME called REACTIONS_BY_TIME_SHORT is created which is used to store the sample records chosen from the big table.

2. The records in REACTIONS_BY_TIME_SHORT with SESSION_ID equal to the current session identifier is deleted first if there are any.

3. The total number of records rows of REACTIONS_BY_TIME is obtained by querying COUNT(timestamp).

4. An integer called steplength is calculated by the expression ceil(rows/50) where ceil is the ceiling function.

5. The sample records are those records from REACTIONS_BY_TIME where timestamp is a multiple of steplength and stored in the REACTIONS_BY_TIME_SHORT table.

6. The query statement is changed as follows:
   select timestamp "Time Step", total "Total Reactions"
   from reactions_by_time where session_id=:session_id;

The total time of operations in the above steps is less than 1 second. The generated line graph only needs to plot around 50 points, which greatly reduces the time to generated the whole reports page as shown in Figure 5.

8 Basic Data Mining Features

A data warehouse is designed using both the detail-and-summary schema and star schema. We use the Oracle Data Mining tool to discover interesting patterns from the data warehouse. Currently, we only applied clustering to the data warehouse. Clustering is useful for exploring data. It is particularly useful when there are many attributes and no natural groupings. Clustering analysis identifies clusters embedded in the data. A cluster is a collection of data objects that are similar in some sense to one another. A good clustering method produces high-quality clusters to ensure that the inter-cluster similarity is low and the intra-cluster similarity is high; in other words, members of a cluster are more like each other than they are like members of a different cluster.

Oracle implemented two clustering algorithms called Enhanced K-means and O-Cluster. We used both algorithms to the data warehouse and some interesting clusters are discovered. One way to explain these clusters is that dense areas exist in the simulated world; the movements and reactions of these molecules enable the molecules to form clusters. This discovery could be useful to determine the physical distributions of molecules in the evolution of soil. As shown in Figure 6, one more interpretation of the clusters is that molecules form micelles.

Another interesting problem is to predict whether certain molecules will be adsorbed in a certain period of time. This is a typical application of classification. We plan to apply some classification methods such as Naive Bayes algorithm or decision tree algorithm to build a classification model, which then can be used to predict whether certain types of molecules will be adsorbed or not in a certain time period.

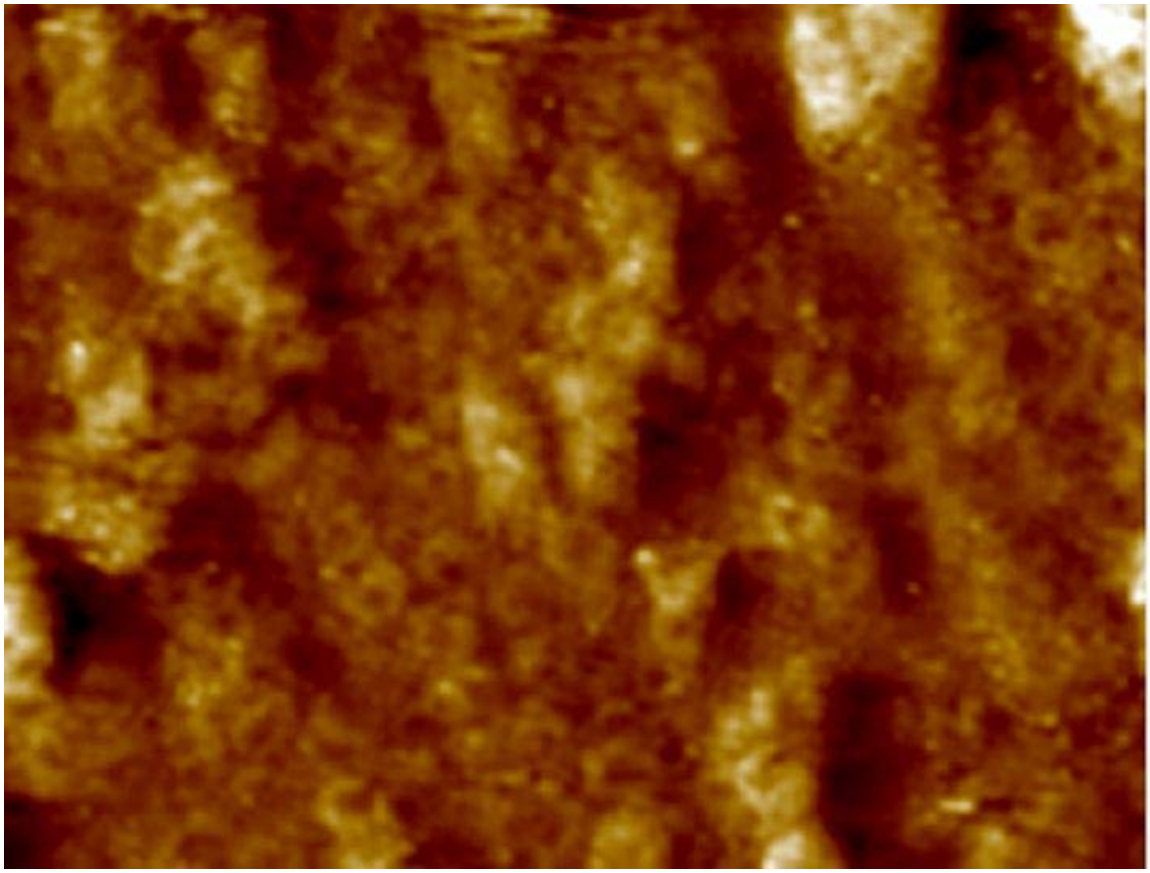

Figure 6: Aggregation/Micelle Formation (Maurice, 1999)

9 Conclusion

In this paper, we have presented an agent-based stochastic model to simulate natural organic matter (NOM). It acts as a starting point of a powerful tool which can be applied to aquatic ecosystem studies, soil and crop science, environmental protection, remediation in the surface and sub-surface, and global climate change prediction. Unlike current models, this model explicitly treats NOM as a heterogeneous mixture, so that distributions of physical, chemical and biological properties can be predicted.

The simulation system employs recent advances in web-based interfaces such as J2EE, and scalable web-based database management systems such as Oracle to improve the reliability and scalability of the stochastic simulations and to facilitate analysis of the resulting large datasets. We applied other information technologies such as data warehousing and data mining to investigate the fourth paradigm of research proposed by Fox.

There are several possible extensions of our current project. First of all, more features can be added into the core simulation. Currently, we only allowed ten most popular types of chemical reactions in the model. Obviously, there are far more reactions existing in the real world. We could add a feature to allow the users to build new reaction types.

References

[1] G.R. Aiken, D.M. McKnight, R.L. Wershaw, and P. MacCarthy. *Humic substrances in soil, sediment, and water - geochemistry, isolation and characterization.* John Wiley Sons, Berkeley, 1985.

[2] D.M. McKnight and G.R. Aiken. *Sources and age of humus.* Springer-Verlag, Berlin, 1985.

[3] S.E. Cabaniss, Q. Zhou, P.A. Maurice, Y.P. Chi, and G.R. Aiken. A log-normal distribution model for the molecular weight of aquatic fulvic acids. In *Environ. Sci. Technol. 34*, pages 1103–1109, 2000.

[4] K. Kalbitz, S. Solinger, J.H. Park, B. Mchalzik, and E. Matzner. Controls on the dynamics of dissolved organic matter in soils:a review. In *Soil Sci.165*, pages 277–304, 2000.

[5] E.M. Perdue, J.H. Reuter, and R.S. Parrish. A statistical model of proton binding by humus. In *Geochim, Cosmochim. Acta 48*, pages 1257–1263, 1984.

[6] H.F. Hemond. Acid neutralizing capacity, alkalinity and acid-base status of natural waters containing organic acids. In *Environ. Sci. Technol. 24*, pages 1486–1489, 1990.

[7] J.A. Leenheer, G.K. Grown, P. MacCarthy, and S.E. Cabaniss. Models of metal-binding structure in fulvic acid from the suwannee river, geogia. In *Environ. Sci. Technol. 32*, pages 2410–2416, 1998.

[8] H.M. Sauro. Scamp: a general-purpose simulator and metabolic control analysis program. In *Comput. Appl. BioSci 9*, pages 441–450, 1993.

[9] D.J.M. Park and B.E. Wright. Metasim: a general purpose metabolic simulator for studying cellular transformations. In *Comput. Prog. Biomed. 3*, pages 10–26, 1973.

[10] D. Bray and R.B. Bourret. Computer analysis of the binding reactions leading to a transmembrane receptor-linked multiprotein complex involved in bacterial chemotaxis. In *Mol. Biol. Cell. 6*, pages 1367–1380, 1995.

[11] K.W. Kohn. Functional capabilities of molecular network components controlling the mamalian g1/s cell cycle phase transition. In *Oncogene 16*, pages 1065–1075, 1998.

[12] D.T. Gillespie. A general methods for numerically simulating the stochastic time evolution of coupled chemical reactions. In *J. Comp. Phys., 22*, pages 403–434, 1976.

[13] IBM Almaden Research Center. Chemical kinetics simulator 1.0 user's manual. In *IBM Corporation*, 1995.

[14] K. Wight R.J. Larson P.H. Masschelen W.F. Punch, A. Patton and L. Forney. A biodegradability evalution and simulation system (bess) based on knowledge of biodegradation pathways. In *Biodegradability Prediction, Peijnenburg and Darborsky Editors, NATO ASI Series, Number 2, Vol 23*, pages 65–84, 1997.

[15] C.A.J.M. Firth and D. Bray. Stochastic simulation of cell signalling pathways. In *Computational Modeling of Genetic and Biochemical Networks, MIT Press*, 2001.

[16] C.A.J.M. Firth. Stochastic simulation of cell signalling pathways. In *University of Cambridge*, 1998.

[17] *http://www.santafe.edu/sfi/publications/Working-Papers/97-01-002.pdf*

[18] R. Alur, C. Belta, V. Kumar, M. Mintz, G. J. Pappas, H. Rubin and J. Schug *Modeling and Analyzing Biomolecular Networks. IEEE: Computing in Science and Engineering*, pages 20-31, 2002.

[19] V. Getov, G. von Laszewski, M. Philippsen and I. Foster *Multiparadigm communications in java for grid computing.* Communications of ACM, Volume 14, Issue 10, 2001.

[20] G.K. Thiruvathukal. *Java at Middle Age: Enabling Java for Computational Science. IEEE: Computing in Science and Engineering*, pages 74-84, 2002.

[21] O. Vormoor. *Quick and Easy Interactive Molecular Dynamics Using Java3D. IEEE: Computing in Science and Engineering*, pages 98-104, 2001.

[22] G. Fox. *E-Science Meets Computational Science and Information Technology. IEEE: Computing in Science and Engineering*, pages 84-85, 2002.

[23] K. Loney and G. Koch. *Oracle8i: The complete reference.* Osborne McGraw-Hill, Berkeley, 2000.

[24] R.J. Muller. *Oracle Developer Starter Kit.* Osborne, Berkeley, 1999.

[25] J. Price and L. Wald. *Oracle9i JDBC Programming.* Osborne McGraw-Hill, Berkeley, 2002.

[26] *http://java.sun.com/j2ee.*

[27] M. Hall. *Core Servlets and JavaServer Pages.* Prentice Hall, New Jersey, 2001.