OSIRIS, an Entirely in-House Developed Drug Discovery Informatics System

Thomas Sander,* Joel Freyss, Modest von Korff, Jacqueline Renée Reich, and Christian Rufener

Department of Research Informatics, Actelion Ltd., Gewerbestrasse 16, CH-4123 Allschwil, Switzerland

Received August 29, 2008

We present OSIRIS, an entirely in-house developed drug discovery informatics system. Its components cover all information handling aspects from compound synthesis via biological testing to preclinical development. Its design principles are platform and vendor independence, a consistent look and feel, and complete coverage of the drug discovery process by custom tailored applications. These include electronic laboratory notebook applications for biology and chemistry, tools for high-throughput and secondary screening evaluation, chemistry-aware data visualization, physicochemical property prediction, 3D-pharmacophore comparisons, interactive modeling, computing grid based ligand-protein docking, and more. Most applications are developed in Java and are built on top of a Java library layer that provides reusable cheminformatics functionality and GUI components such as chemical editors, structure canonicalization, substructure search, combinatorial enumeration, enhanced stereo perception, force field minimization, and conformation generation.

1. INTRODUCTION

In the 1970s, when mainframe computers started to be used in the industry, pharmaceutical companies developed in-house database systems to manage chemical and biological information. With the advent of commercial systems in the 1980s, in-house development changed from providing the core functionality toward the integration and customization of commercial components. Since then the number of specialized databases and computer programs in a single company grew steadily.[1,2] The number of operating systems, database engines, programming languages and interfaces, data formats, and system dependencies multiplied in parallel and added its own dimension to the increasing complexity of the systems landscape. Data warehouses were introduced as an additional database layer to simplify data access.[3,4] Recent reports in literature seem to show a trend back toward in-house software development.[5−8] Reasons may be that existing frameworks and open-source components increasingly provide reusable functionality and that modern object-oriented languages and Web technology allow for more rapid software development. Some motivation may also have been risen from dissatisfied users being forced to access a multitude of diverse systems and databases to collect relevant data. The wish for homogeneity and simplification led to the development of integrated systems such as Avalon at Novartis[9] or ABCD at Johnson&Johnson.[10] Astonishingly, even small companies sometimes decide to develop core systems themselves.[11]

In 1998, when Actelion was founded, its discovery informatics was in the fortunate position to start months before the first chemistry laboratories were operational. Since no legacy systems needed to be migrated or supported, the choice of technologies, commercial components, and approaches was an open one. Nevertheless, the challenging task remained to build up a drug discovery IM platform within half a year that would comprise basic functionality such as substance registration, biological data upload, and chemistry aware data browsing and yet would constitute a robust foundation for flexible growth in terms of users, data, and functionality.

2. CONCEPT

Requirements for a Drug Discovery IM System. From a scientific user's perspective a discovery informatics system should effortlessly capture and store all business relevant data in an organized fashion. It should provide easy access to the information to answer any foreseeable and unforeseeable questions. Visualization of multidimensional data, structure−activity relationships, and intrinsic chemical intelligence should aid in revealing concealed knowledge.[12,13] Functionality should not be restricted to certain user groups, and software components should not enforce an inefficient workflow. Core functionality should be intuitive, and a minimum of training should allow an efficient usage of the system's components. Different components of the system should have consistent user interfaces and allow for a seamless exchange of data.

From a technical point of view, a system should be scalable to adapt for significant growth in terms of users and data without sacrificing performance. It should be modular to allow changing or replacing components without affecting the rest of the system. It should be flexible enough to add and change functionality to satisfy new requirements. A high degree of independence of proprietary components, operating system or database versions, Web browsers, software vendors, etc. is a must to achieve architectural flexibility. All relevant information should be centralized within one database. A multiple database architecture would either sacrifice query performance through an additional overhead for cross database queries or it would at some time imply maintaining a data warehouse containing steadily updated summary information.

* Corresponding author phone: + 41 61 565 65 23; fax: +41 61 565 65 00; e-mail:thomas.sander@actelion.com.

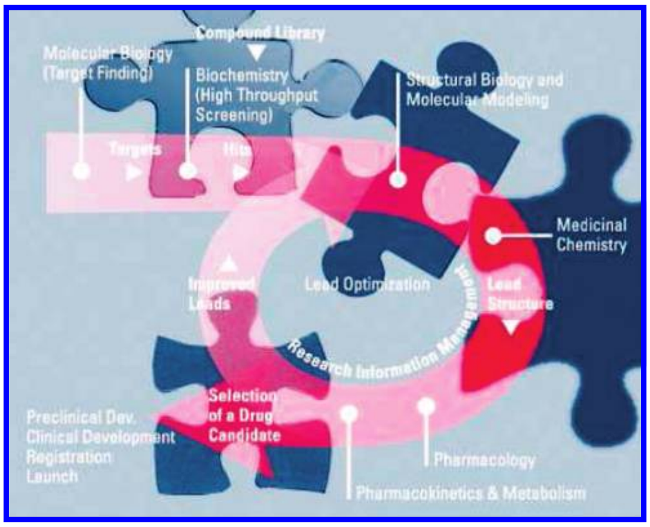

Compound Library
Molecular Biology (Target Finding)
Biochemistry (High Throughput Screening)
Structural Biology and Molecular Modeling
Targets
Hits
Medicinal Chemistry
Improved Leads
Lead Optimization
Lead Structure
Research Information Management
Selection of a Drug Candidate
Preclinical Dev. Clinical Development Registration Launch
Pharmacology
Pharmacokinetics & Metabolism

Figure 1. Standard approach: Commercial tools and databases from vendors cover different parts of the drug discovery workflow. Substantial in-house development is needed to glue commercial components together.

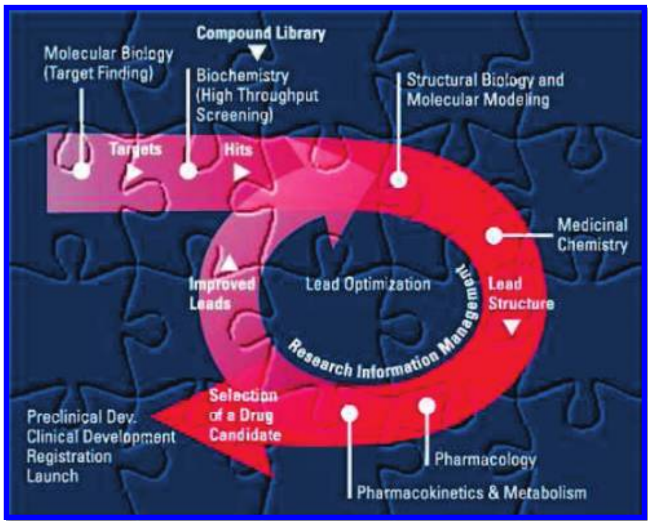

Compound Library
Molecular Biology (Target Finding)
Biochemistry (High Throughput Screening)
Structural Biology and Molecular Modeling
Targets
Hits
Medicinal Chemistry
Improved Leads
Lead Optimization
Lead Structure
Research Information Management
Selection of a Drug Candidate
Preclinical Dev. Clinical Development Registration Launch
Pharmacology
Pharmacokinetics & Metabolism

Figure 2. Actelion's approach. In-house developed tools and databases aim for complete coverage of the drug discovery workflow with modular, interoperable, consistent, robust, and independent applications and databases.

Standard Approach. The standard approach in the pharmaceutical industry for building up a drug discovery informatics platform is to license software components and database systems from a variety of vendors such as Accelrys,[14] Advanced Chemistry Development,[15] CambridgeSoft,[16] ChemAxon,[17] Chemical Computing Group,[18] Daylight,[19] ID Business Solutions,[20] Open Eye,[21] Spotfire,[22] and Symyx.[23] Typically, products from half a dozen different vendors are at work in the various areas of the drug discovery informatics landscape (Figure 1). The major areas of this landscape are chemical data management and structure search, biological data management, compound property prediction, virtual screening, data visualization, and electronic laboratory notebooks for chemistry and biology. Commercial components are often limited to one platform, have proprietary programming interfaces, use incompatible data formats, or have incompatible concepts. A substantial additional investment and development effort are often needed to integrate a new commercial component into an existing environment. A relevant example is the conversion of a paper based chemistry notebook to an electronic one. Most large pharmaceutical companies recently underwent or plan to implement such a change. Commercial laboratory notebook applications allow chemists to protocol their synthesis procedures, to calculate amounts of starting materials and solvents, enumerate combinatorial libraries, attach analytical results, and to search and browse through older experiments. For a full integration an electronic notebook needs to be customized to support the existing compound registration process, to allow direct search for chemicals in existing catalog databases, to directly access existing spectra databases, etc. It is not uncommon that the total implementation cost of an electronic notebook for chemistry exceeds U.S. $10,000,000. Despite the elevated costs, some shortcomings may be unsolvable.

Actelion's Approach. Actelion's approach is fundamentally different. With a tight workflow integration and consistent user experience in mind, we focused from the first days on a high degree of scientific software development in-house. Devoid of robust open-source cheminformatics functionality, we started in 1998 to develop cheminformatics functionality from scratch. A chemical editor, stereo perception, and structure canonicalization were among the first components to be developed. We chose Oracle as a scalable persistence layer for chemical and biological data. When the first chemistry laboratories became productive half a year later, a basic but functional informatics infrastructure was put in place: a compound registration application with novelty checking, biological data upload and querying from within MS-Excel, and a dedicated application for secondary screening evaluation. We automatically created ChemFinder[24] databases of our research projects, giving scientists access to biological data combined with substructure and similarity search options.

In 2008, 10 years later, OSIRIS covers all information handling aspects from compound synthesis via biological testing to preclinical development (Figure 2). Our electronic notebook for chemistry supports medicinal, technical, and combinatorial chemists equally. We were among the first companies to rollout a biological notebook application. In addition to our in-house compounds and results database, we maintain dedicated databases for tracking commercially available chemicals and screening compounds, in-house chemicals, biological material, laboratory animals, protein structures, and more. Our biologists use customized applications for high-throughput and secondary screening. Our scientists use a chemistry-aware data analysis and visualization tool with direct database access. A proprietary forcefield, scoring function, and 3D-pharmacophore are frequently applied to virtual screening on 300 CPUs of our computing grid. In addition, we maintain our own PDF-based document management system and a drug discovery Wiki to exchange ideas and encourage innovation. All applications and concepts are consistent in terms of user experience. The chemical editor is always the same for defining a substructure fragment, a generic reaction, or a molecule to be registered, whether the editor is used in a Web based application or in a standalone one. We consistently apply MDL's concept of the enhanced stereochemical representation. Our chemical editor refuses incompletely defined stereochemistry features. Stereocenters are always depicted with respective tags and consistent coloring. All algorithms for substructure search,

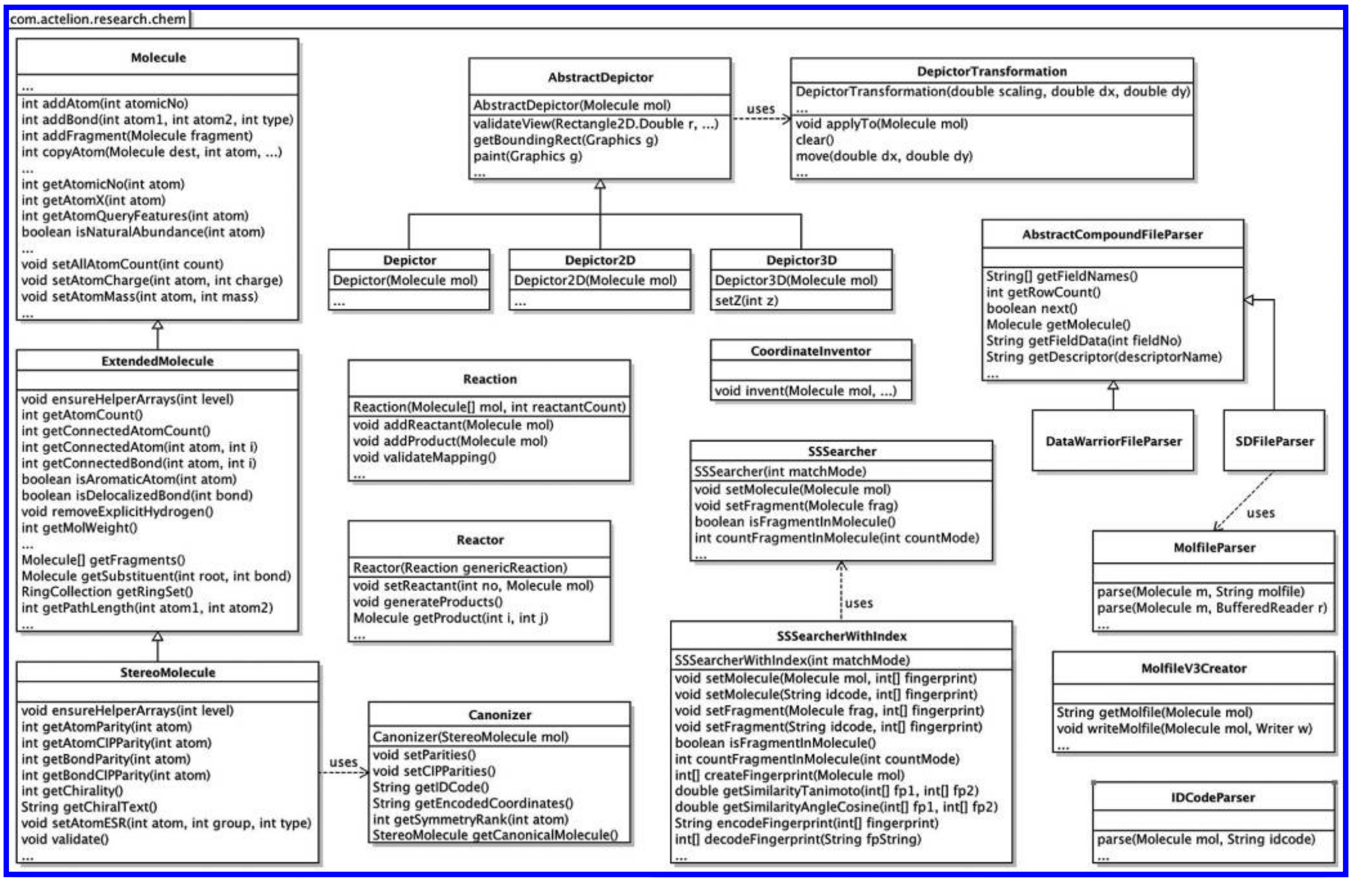

com.actelion.research.chem
Molecule
...
int addAtom(int atomicNo)
int addBond(int atom1, int atom2, int type)
int addFragment(Molecule fragment)
int copyAtom(Molecule dest, int atom, ...)
...
int getAtomicNo(int atom)
int getAtomX(int atom)
int getAtomQueryFeatures(int atom)
boolean isNaturalAbundance(int atom)
...
void setAllAtomCount(int count)
void setAtomCharge(int atom, int charge)
void setAtomMass(int atom, int mass)
...
ExtendedMolecule
void ensureHelperArrays(int level)
int getAtomCount()
int getConnectedAtomCount()
int getConnectedAtom(int atom, int i)
int getConnectedBond(int atom, int i)
boolean isAromaticAtom(int atom)
boolean isDelocalizedBond(int bond)
void removeExplicitHydrogen()
int getMolWeight()
...
Molecule[] getFragments()
Molecule getSubstituent(int root, int bond)
RingCollection getRingSet()
int getPathLength(int atom1, int atom2)
...
StereoMolecule
void ensureHelperArrays(int level)
int getAtomParity(int atom)
int getAtomCIPParity(int atom)
int getBondParity(int atom)
int getBondCIPParity(int atom)
int getChirality()
String getChiralText()
void setAtomESR(int atom, int group, int type)
void validate()
...
uses
Canonizer
Canonizer(StereoMolecule mol)
void setParities()
void setCIPParities()
String getIDCode()
String getEncodedCoordinates()
int getSymmetryRank(int atom)
StereoMolecule getCanonicalMolecule()
AbstractDepictor
AbstractDepictor(Molecule mol)
validateView(Rectangle2D.Double r, ...)
getBoundingRect(Graphics g)
paint(Graphics g)
...
uses
DepictorTransformation
DepictorTransformation(double scaling, double dx, double dy)
...
void applyTo(Molecule mol)
clear()
move(double dx, double dy)
...
Depictor
Depictor(Molecule mol)
...
Depictor2D
Depictor2D(Molecule mol)
...
Depictor3D
Depictor3D(Molecule mol)
setZ(int z)
Reaction
Reaction(Molecule[] mol, int reactantCount)
void addReactant(Molecule mol)
void addProduct(Molecule mol)
void validateMapping()
...
Reactor
Reactor(Reaction genericReaction)
void setReactant(int no, Molecule mol)
void generateProducts()
Molecule getProduct(int i, int j)
...
CoordinateInventor
void invent(Molecule mol, ...)
SSSearcher
SSSearcher(int matchMode)
void setMolecule(Molecule mol)
void setFragment(Molecule frag)
boolean isFragmentInMolecule()
int countFragmentInMolecule(int countMode)
...
uses
SSSearcherWithIndex
SSSearcherWithIndex(int matchMode)
void setMolecule(Molecule mol, int[] fingerprint)
void setMolecule(String idcode, int[] fingerprint)
void setFragment(Molecule frag, int[] fingerprint)
void setFragment(String idcode, int[] fingerprint)
boolean isFragmentInMolecule()
int countFragmentInMolecule(int countMode)
int[] createFingerprint(Molecule mol)
double getSimilarityTanimoto(int[] fp1, int[] fp2)
double getSimilarityAngleCosine(int[] fp1, int[] fp2)
String encodeFingerprint(int[] fingerprint)
int[] decodeFingerprint(String fpString)
...
AbstractCompoundFileParser
String[] getFieldNames()
int getRowCount()
boolean next()
Molecule getMolecule()
String getFieldData(int fieldNo)
String getDescriptor(descriptorName)
...
DataWarriorFileParser
SDFileParser
uses
MolfileParser
parse(Molecule m, String molfile)
parse(Molecule m, BufferedReader r)
...
MolfileV3Creator
String getMolfile(Molecule mol)
void writeMolfile(Molecule mol, Writer w)
...
IDCodeParser
parse(Molecule mol, String idcode)
...

Figure 3. Toolbox chemistry package - subset of classes with subset of methods.

canonicalization, combinatorial library enumeration, etc. handle the enhanced stereo information in a correct manner.

3. IMPLEMENTATION

Technology. Our primary design principles were platform and vendor independence, a modular, object-oriented architecture, scalability and robustness, and open interfaces wherever possible. Because of the broad vendor and community support we considered the Java SE platform the only reasonable option.[25] It combines a modern programming language with a functionality framework that allows focusing on the cheminformatics aspects rather than on the construction of user interface elements..NET was not considered because we aimed for platform independence.

Most applications are designed as client-side Java applications. Where a direct communication to the database was desired they employ a JDBC[26] connection. Where chemical data types need to be passed or where complex or calculation intense server processes are involved, a three-tier-architecture is used. In these cases we use RMI[27] or CORBA[28] connection layers, because they allow to directly pass chemical objects rather than to encode them into strings for passing. A platform specific client-side DLL provides native clipboard support of chemistry objects. Some applications were realized as true Web applications using applets, an Apache Web server, Tomcat,[29] JSP,[30] Servlets,[31] and Struts.[32]

Oracle[33] was chosen as the database engine for its robustness, excellent Java support, multiplatform availability, and defacto standard in the pharmaceutical industry. In order to extend its search capabilities we developed an Oracle cartridge in Java that provides chemical search functions. Java stored procedures handle all data upload and ensure data integrity by checking and ensuring complex constraints and record level access rights.

Reusable Software Toolbox. Information systems for drug discovery differ fundamentally from their peers in other branches of the industry where often exclusively alphanumerical data is processed. Drug discovery information is closely associated with chemical structures, connectivity graphs, chemical reactions, protein structures, etc. Generic database systems and programming frameworks miss the functionality to display, edit, and process these uncommon data types. We developed a framework of reusable Java classes providing low level chemical intelligence, user-interface elements, and utility functionality. This toolbox of components provides prefabricated functionality and hides complex algorithms behind simple interfaces of Java classes. It provides the foundation for end-user applications and server components. Similar to the benefit of widely used pipelining tools such as Knime[35] and Pipeline Pilot,[36] a moderately skilled Java developer can quickly build complex workflows from basic toolbox components. The toolkit is organized in 36 packages with a total of about 700 classes and interfaces of which about 200 are user interface elements. An UML diagram (Figure 3) illustrates the concepts behind some chemistry core classes. Important examples for low-level cheminformatics classes are as follows:

The *Molecule* class contains a molecule or query fragment as connection table and methods to manipulate atom and bond properties, query features, etc. *ExtendedMolecule*, derived from *Molecule*, includes automatic detection of neighbors, rings, aromaticity, and implicit hydrogens. *Ste-*

reoMolecule, which extends *ExtendedMolecule,* contributes full stereo perception, symmetry and meso detection, stereo parity calculation, and Cahn-Ingold-Prelog assignment of stereo features. A *Canonizer* object creates so-called *idcodes* from Molecule objects, which are compact canonicalized molecule encodings that include all stereo and query features. *Idcodes* play an important role for structure uniqueness checking, for efficient storage of large molecule sets, and for achieving response times within a few seconds for the retrieval of thousands of compounds from databases.

Depictor and *Depictor2D* draw *Molecule* objects on pixel based devices like computer screens or in publishing quality for devices being capable of vector graphics, respectively. Classes like *Reaction* and *MarkushStructure* handle more complex chemical objects. Some classes convert frequent molecule exchange formats such as MDL molfiles,[37] SD-Files, SMILES,[38] and *Idcodes* into *Molecule* objects and back. A *SSSearcher* class performs substructure-searches by applying a graph-matching algorithm while simultaneously considering query and stereo features. *SSSearcherWithIndex* is used for searching thousands to millions of molecules by applying a fragment based fingerprint to preselect candidates. A *CoordinateInventor* creates new 2D-coordinates for all or some of the atoms of a molecule. Specialized *DescriptorHandler*s create various topology or conformation based descriptors, which can be used to compute the chemical or biological similarities of two molecules. A *Clusterer* clusters thousands of compounds based on any binary descriptor. Where accurate clustering hits the computational limit, a centroid vector based *DiversitySelector* is used that ranks up to a few millions of molecules according to their potential to increase a chemical library's overall diversity. A *Reactor* enumerates combinatorial library structures, given a generic reaction and a list of reactants for every generic reactant. Dedicated classes predict physicochemical properties or toxicity risks.[39] A *CompoundTableModel* provides persistent handling of chemistry aware tabular data, which may be visualized by interactive viewer objects, like tables, 2D or 3D graphical views, or chemical structure grids.

A *ConformationSampler* employs a self-organization algorithm to generate diverse conformers. An implementation of the MM2 forcefield, Forcefield, was inspired from the Tinker[40] code originally written in Fortran. This implementation evaluates the intramolecular energy as the summation of all bond-, angle-, torsion-, dipole-, and Lennard Jones VDW-forces. The extramolecular energy is calculated from the class interaction statistics defined below. The LBFGSOptimizer uses the Broyden-Fletcher-Goldfarb-Shanno algorithm to find the local minimum of a ligand's conformation.

StudyClassInteraction is a class that analyzes protein−ligand interactions from the Protein Data Bank (PDB) and establishes statistics on the number of occurrences of each protein−ligand atom-type pair. For each atom-type pair, we established a VDW like potential function as defined by Gohlke and Cie.[41] The Docker class uses a genetic algorithm to find the global minimum of a ligand within a protein pocket.

The 2D structure editor in the user-interface branch of components may be parametrized to either handle molecules, fragments with query features, reactions, or markush structures. It automatically checks for completeness of stereo configurations and consistency of query features. Other components are used to display molecule collections in a panel, molecules in table cells, etc. A *ClipboardHandler* reads and writes MDL sketch format and Windows Metafiles and ensures native clipboard support of chemical objects. "Drag and drop" of chemical objects between our user interface components is supported. A form view with an associated form designer is used to display database records that contain chemical or image information. A docking framework for JPanels allows for the arrangement of views by moving them around or stacking them on top of each other. Other important components are a 3D structure viewer and editor. All 3D user-interface components support vertical interlacing for autostereoscopic monitors.

Architecture Overview. Figure 4 illustrates the dataflow and connection lines between major components of the system. All Web applications naturally require server components on an application server and therefore are designed in three-tier-architecture. Most stand-alone Java client applications, however, talk directly to the database. An exception is the chemical notebook *Mercury*, which handles all compound registrations through a CORBA based middle-tier service. This service provides a uniform mechanism for requesting *Actelion Numbers* as substance identifiers directly from *StereoMolecule* instances. It checks for structural novelty and handles salts and isotopes with a reproducible logic. It is also called during the registration of purchased screening compounds from a different, dedicated application.

All alphanumerical data inserts and updates into the OSIRIS database are performed through stored procedures on the database itself. They ensure proper constraint and privilege handling. All major applications are explained in the following in more detail.

Electronic Laboratory Notebook for Chemistry Mercury. Chemists traditionally have documented their experiments in paper notebooks. Early in the 1990s, some pharmaceutical companies attempted to develop or introduce computer-based notebooks, but the results were mostly mediocre. Electronic notebooks (ELNs) only gained momentum in recent years and have become a trend. Today, several software companies sell shrink-wrapped products in this area. Actelion started a Chemistry ELN project in early 2001 to support chemists in the combinatorial and parallel chemistry group. Its acceptance beyond the targeted audience (a number of medicinal chemists) led in 2003 to the development of the ELN *Mercury* for Actelion's entire chemistry community (Figure 5). We maintained a close interaction with the chemists during *Mercury*'s development to optimally meet their needs and enhance productivity.

Mercury is a Java Swing-based three-tier application that heavily uses the Actelion Toolbox, which at that time had already reached a substantial functionality. *Mercury* keeps track of performed reactions, the reactants used, and the desired products. It automatically performs a stoichiometry calculation for the reactants. Reactants and products are displayed and stored as chemical structures and can therefore be queried by exact match and substructure search.

Mercury supports chemists during experiment planning, execution, and product postprocessing. During the planning phase, the chemist selects starting materials and checks their availability by consulting an in-house chemical inventory and a database of commercially available compounds directly from within *Mercury*. When preparing chemical libraries or

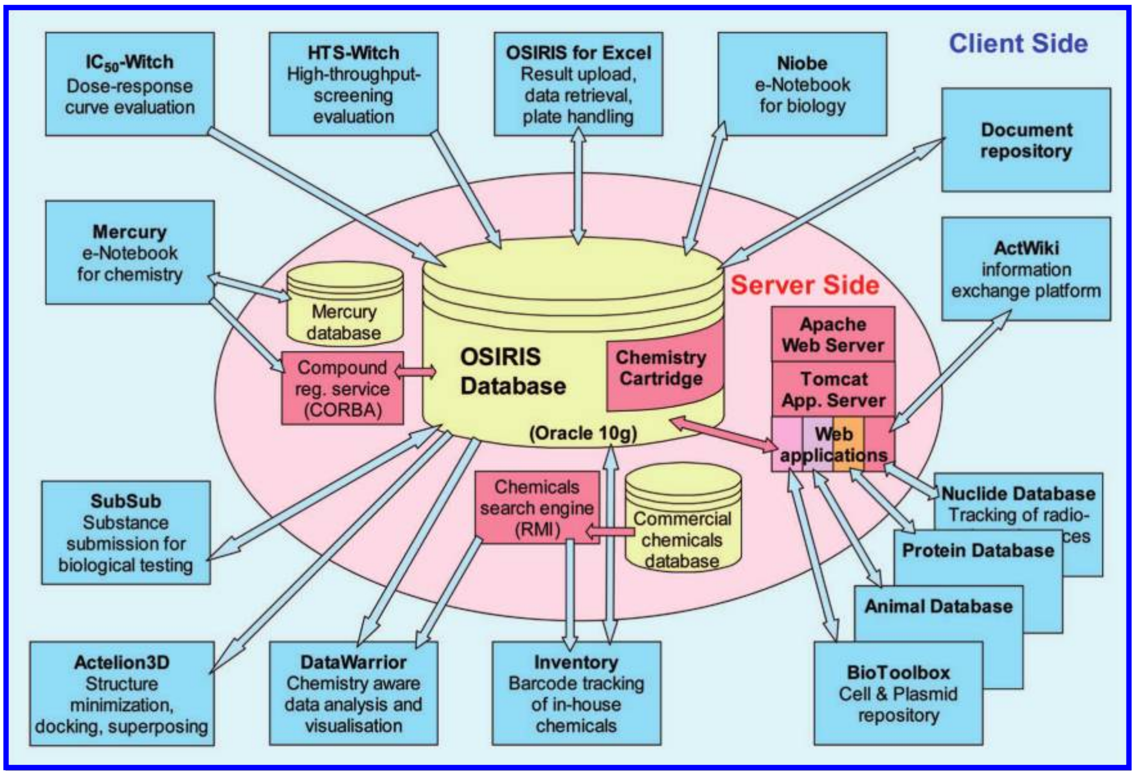
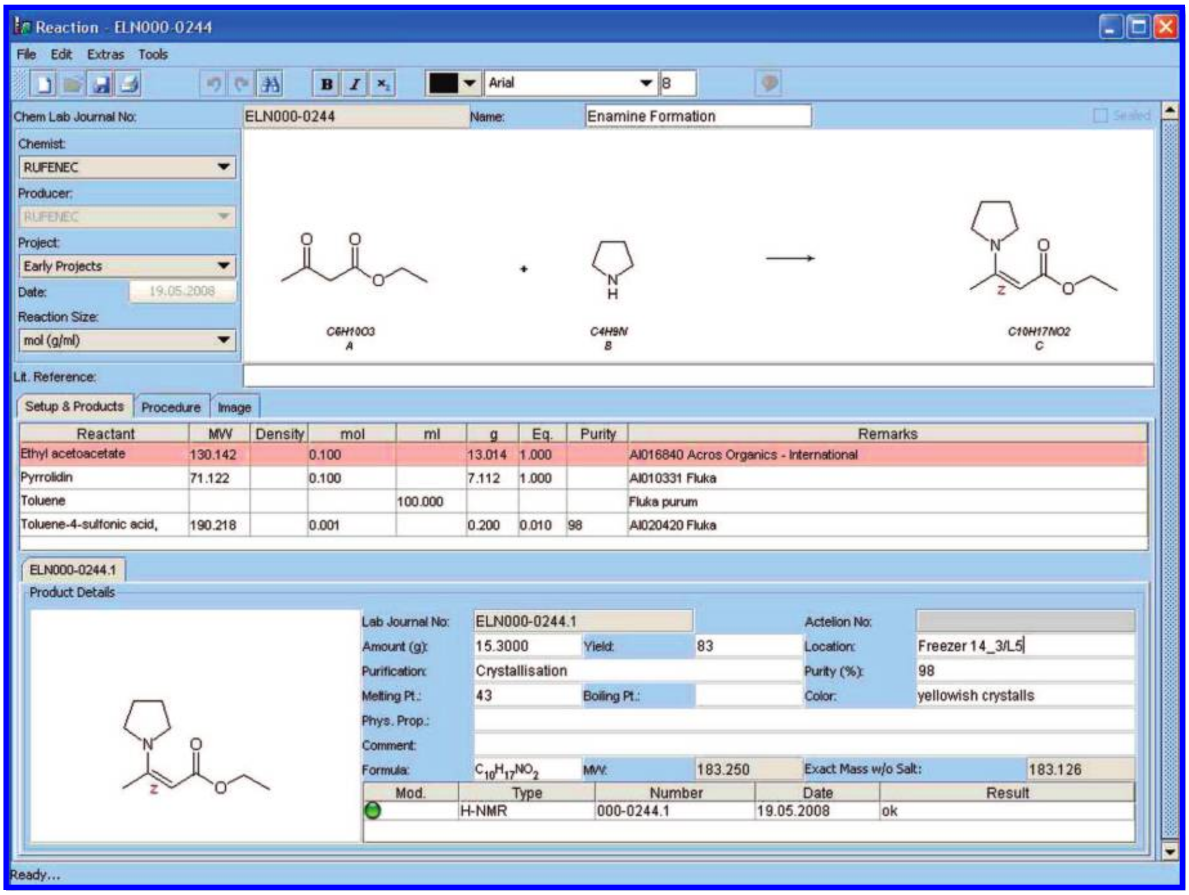

Client Side
IC50-Witch Dose-response curve evaluation
HTS-Witch High-throughput-screening evaluation
OSIRIS for Excel Result upload, data retrieval, plate handling
Niobe e-Notebook for biology
Document repository
Mercury e-Notebook for chemistry
Mercury database
Compound reg. service (CORBA)
OSIRIS Database
Chemistry Cartridge
(Oracle 10g)
Server Side
Apache Web Server
Tomcat App. Server
Web applications
ActWiki information exchange platform
Chemicals search engine (RMI)
Commercial chemicals database
Nuclide Database Tracking of radio-...ces
Protein Database
Animal Database
BioToolbox Cell & Plasmid repository
SubSub Substance submission for biological testing
Actelion3D Structure minimization, docking, superposing
DataWarrior Chemistry aware data analysis and visualisation
Inventory Barcode tracking of in-house chemicals

Figure 4. Architecture - simplified dataflow between applications and server components.


Reaction - ELN000-0244

File Edit Extras Tools

Chem Lab Journal No: ELN000-0244 Name: Enamine Formation

Chemist: RUFENEC
Producer: RUFENEC
Project: Early Projects
Date: 19.05.2008
Reaction Size: mol (g/ml)

Lit. Reference:

Setup & Products | Procedure | Image

Reactant	MW	Density	mol	ml	g	Eq.	Purity	Remarks
Ethyl acetoacetate	130.142		0.100		13.014	1.000		AI016840 Acros Organics - International
Pyrrolidin	71.122		0.100		7.112	1.000		AI010331 Fluka
Toluene				100.000				Fluka purum
Toluene-4-sulfonic acid,	190.218		0.001		0.200	0.010	98	AI020420 Fluka

ELN000-0244.1

Product Details

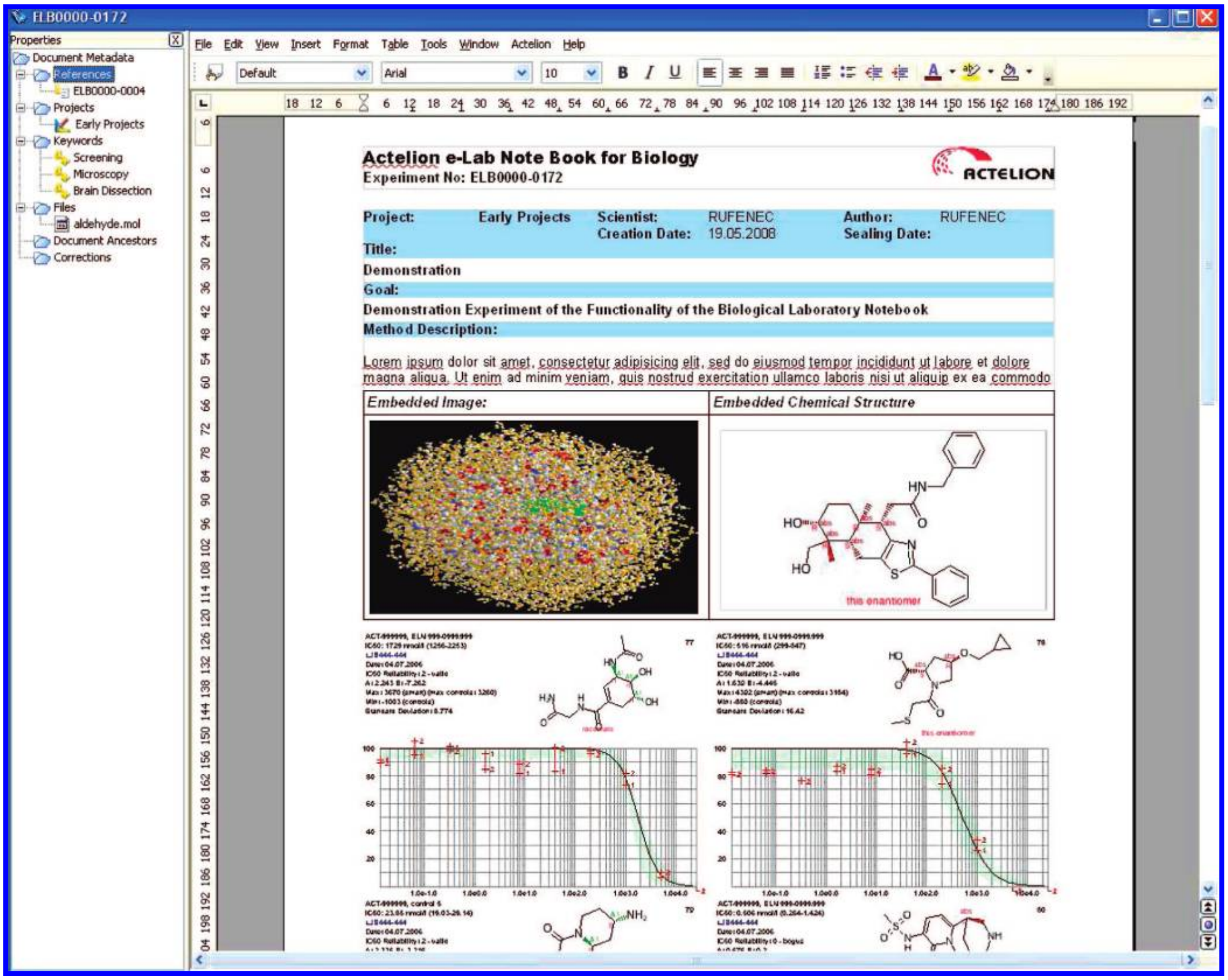
Lab Journal No:	ELN000-0244.1			Actelion No:	
Amount (g):	15.3000	Yield:	83	Location:	Freezer 14_3/L5
Purification:	Crystallisation			Purity (%):	98
Melting Pt.:	43	Boiling Pt.:		Color:	yellowish crystalls
Phys. Prop.:					
Comment:					
Formula:	$C_{10}H_{17}NO_2$	MW:	183.250	Exact Mass w/o Salt:	183.126

Mod.	Type	Number	Date	Result
●	H-NMR	000-0244.1	19.05.2008	ok

Ready...

Figure 5. *Mercury* - the e-notebook application for chemistry.

performing parallel chemistry experiments, building blocks can easily be added via drag and drop from any other Actelion chemistry-aware application, via copy and paste from the clipboard, or by reading them from SD-files. *Mercury* can generate all product structures from the defined generic reaction and given list of building blocks. The experiment protocol is edited within a lightweight built-in word-processor. All isolated or postulated products are


ELB0000-0172
File Edit View Insert Format Table Tools Window Actelion Help
Default
Arial
10
ACTELION

Properties
- Document Metadata
  - References
    - ELB0000-0004
  - Projects
    - Early Projects
  - Keywords
    - Screening
    - Microscopy
    - Brain Dissection
  - Files
    - aldehyde.mol
  - Document Ancestors
  - Corrections

Actelion e-Lab Note Book for Biology
Experiment No: ELB0000-0172

Project:	**Early Projects**	**Scientist:**	RUFENEC	**Author:**	RUFENEC
		Creation Date:	19.05.2008	**Sealing Date:**	

Title:
Demonstration
Goal:
Demonstration Experiment of the Functionality of the Biological Laboratory Notebook
Method Description:

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo

Embedded Image:	*Embedded Chemical Structure*

Figure 6. *Niobe* - the e-notebook for biology.

tracked, and analytical artifacts like spectra can be attached automatically or via drag and drop from the file system.

Once a product's chemical structure has been confirmed, the product is registered as a new batch in the OSIRIS database. The middle-tier service performs a novelty check of the chemical structure upon registration. Depending on the outcome, either a new Actelion Number is issued or a new batch is attributed to an existing number. A generic import function allows adding Supporting Information from any source, and a flexible export function is available as well. It is used, for instance, when writing patents to export compound lists with IUPAC names and analytical data.

Electronic Laboratory Notebook for Biology Niobe. The success of *Mercury* led to the decision to introduce an ELN for Actelion's biology department in 2006. It was apparent that we would have to pursue a different approach than with *Mercury*; biology experiments can be radically different from one laboratory to the other. This made it virtually impossible to use a form-based application approach, and a Web-based approach seemed unusable. Some people had already begun to use word processor applications for documentation purposes, and we wanted to keep this analogy. From a strategic and technical point of view, open standards and Java support were necessary. Therefore, OpenOffice was chosen as the foundation.

Like *Mercury*, *Niobe* is a Java application which uses OpenOffice for word processing and spreadsheet functionality. The OpenOffice component model, UNO, allows programmatic access to most of its internal components. For example, we intercepted the "File-Save" function and redirected the document stream to a database rather than to the file system.

Many of *Mercury*'s concepts where reused, and the lessons learned from its development were applied to designing *Niobe* (Figure 6). The word processor centric approach allows biologists to assemble documents in an almost unrestricted manner. Introducing tables, formats, and images become simple operations and allow feature-rich electronic notebook entries. New documents are created from templates or by cloning and adapting existing ones. Notebook entries are extended by meta-data, which play a key role within *Niobe*. Meta-data puts a notebook entry into context by defining the project, responsible scientist, and author of the experiment, keywords, references, supporting files, etc. References can cross-link to other *Niobe* or *Mercury* experiments, Actelion compounds, and the like. Another important meta-data element is supporting files such as raw data from instruments, large image scans, and output from data evaluation tools. By attaching these files to the notebook entry, the biologist can ensure that any person accessing the notebook entry will also have read access to these files. Automatic indexing of the document content allows users to search notebooks by either a Google-type of search or by individually querying meta-data elements. As a final step in the documentation process, the notebook entry is sealed in an unalterable format.

Lessons Learned during the ELN Projects. The key lesson learned was about integration: integration of the user in the development and integration of external systems.

The users were involved not only when formulating the requirements but also during the iterative development. We performed “development sessions” with a small number of users on a regular basis. Through preimplementing the functionality in question and interactive prototyping the users could “shape” the functionality during these sessions. This led to substantial progress in some key areas. We consider the frequent and direct communication between the drug discovery IM group and chemists and biologists a crucial success factor that allows quick adaptations from changed requirements and new user requests.

The integration of external systems became an immanent factor during the development and the life cycle of the ELNs. Nowadays almost every instrument in a laboratory is equipped with a computer. A lot of the primary or secondary data acquired by these instruments are considered important and need to be kept and referenced from within individual ELN entries. In order to respond in a fast manner to such requests we implemented a generic plug-in architecture. In addition, it allows the modeling of workflows which may be specific from laboratory to laboratory. The development of an additional plug-in does not require any changes in the ELN software itself. Examples of such plug-ins are notifications of other group members, display of chemical structures when highlighting a reference number, or the implementation of a workflow for result reporting in the DMPK group.

ELN Legal Issues. One fundamental issue during the process of introducing an ELN is the patent-related legal aspect. The United States Patent and Trademark Office (USPTO) first-to-invent system requires transparent record keeping when dealing with patent-related data, be it electronic or paper-based. In the case of a patent dispute one would have to prove at what time the invention occurred.

When we started the development of Mercury, the majority of the pharmaceutical industry was still operating on paper-based notebooks. Companies which were using ELNs were mostly running *hybrid* solutions: The data were electronically stored but backed by manually signed paper records. We had in-depth discussions with our patent department: who is responsible for the record keeping strategy and who would have to deal with the legal issues in the case of a patent dispute. Guided by them we decided for a *hybrid* solution as well. Technology has evolved since, and today we might opt for a fully electronic-based laboratory notebook solution, based on a Public Key Infrastructure (PKI), which was not available to us when Mercury was launched.

OSIRIS Database. The OSIRIS database is the heart of the system. It contains all assay results of biological and preclinical experiments and all compounds that have been tested. Currently, these are more than 500,000 compounds of which more than a quarter have been synthesized in-house. The other compounds were purchased from commercial providers. The compounds were tested in more than 800 assays leading to more than 10,000,000 individual results. Images, data tables, or formatted text may be attached to specific results. More than 10,000 details such as blood pressure or intrinsic clearance curves are available in the OSIRIS database to date.

Distinct chemical entities are *Substance*, *Batch*, and *Sample*, which contain unique chemical structures identified by Actelion numbers, in-house synthesized batches, and purchased samples for screening (Figure 7). While *Batch*es and *Samples* are independent entities with substantially different properties, they are considered equal for biological testing. A *Compound* entity serves as a decoupled reference representing either a *Batch* or a *Sample* record. Its *chem_lab_journal* column contains the primary key of either a *Batch* or a *Sample* record. On the biological side, *Results* are grouped in *Experiments* which are assigned to *Tests,* which in turn belong to *Projects*. A *Result* also references a *Compound.* It can refer to one or more *Result_Details*. Screening *Plates* are organized in plate libraries. A *Plate_Position* table links an individual screening plate cell to the *Compound* it contains.

Other tables manage project and test specific access rights, user queries, project life cycles, *SubSub* requests, etc. All table data inserts and updates are done through stored procedures, which check for record based user privileges and ensure high data quality and consistency.

Chemistry Data Cartridge. Oracle provides an API for the development of extensions to standard SQL functionality. In particular it allows new SQL operators. The basic implementation of such operators often requires the development of a so-called *Domain Index*. The aim of the *Domain Index* is to speed up the search when using a new SQL operator. Its implementation is generally not that complex but requires a thorough investigation on how to store the index data. The alternatives include BFILES, BLOBS, or an index organized table. Storing the index keys of multiple records in BFILES or BLOBS has the potential to speed up query performance but adds complexity and sacrifices performance when structures are changed or new ones are inserted. After experimenting with the options we selected the simpler solution of an index organized table.

We use Java as the implementation language for the SQL extensions that provide substructure and similarity search. This way we could reuse existing classes of our cheminformatics toolbox.

A new SSS operator, for instance, performs a substructure search on the IDCODE column of the SUBSTANCE table (Figure 7) that contains canonical molecule representations. When the SSS operator is part of the WHERE clause of an SQL query, then Oracle calls an SQL function *SSS_MATCH* for every table row. The implementing Java method decodes the *idcode* and performs a fragment match algorithm on the molecule‘s graph. The *SSS_MATCH* function is ignored, however, if the column has been indexed via the custom index type *ACTMOLSEARCHINDEX*. In this case Oracle calls a series of *ODCIIndexStart*, *ODCIIndexFetch*, and *ODCIIndexClose* functions, which also have been implemented as a Java class. The class methods then access the index data, which is stored in a secondary table SUBS_STR_IDX_I, holding ROWIDS of the SUBSTANCE table and the descriptor keys associated with the molecule. During the *ODCIIndexStart* and *ODCIIndexFetch* calls the substruc-

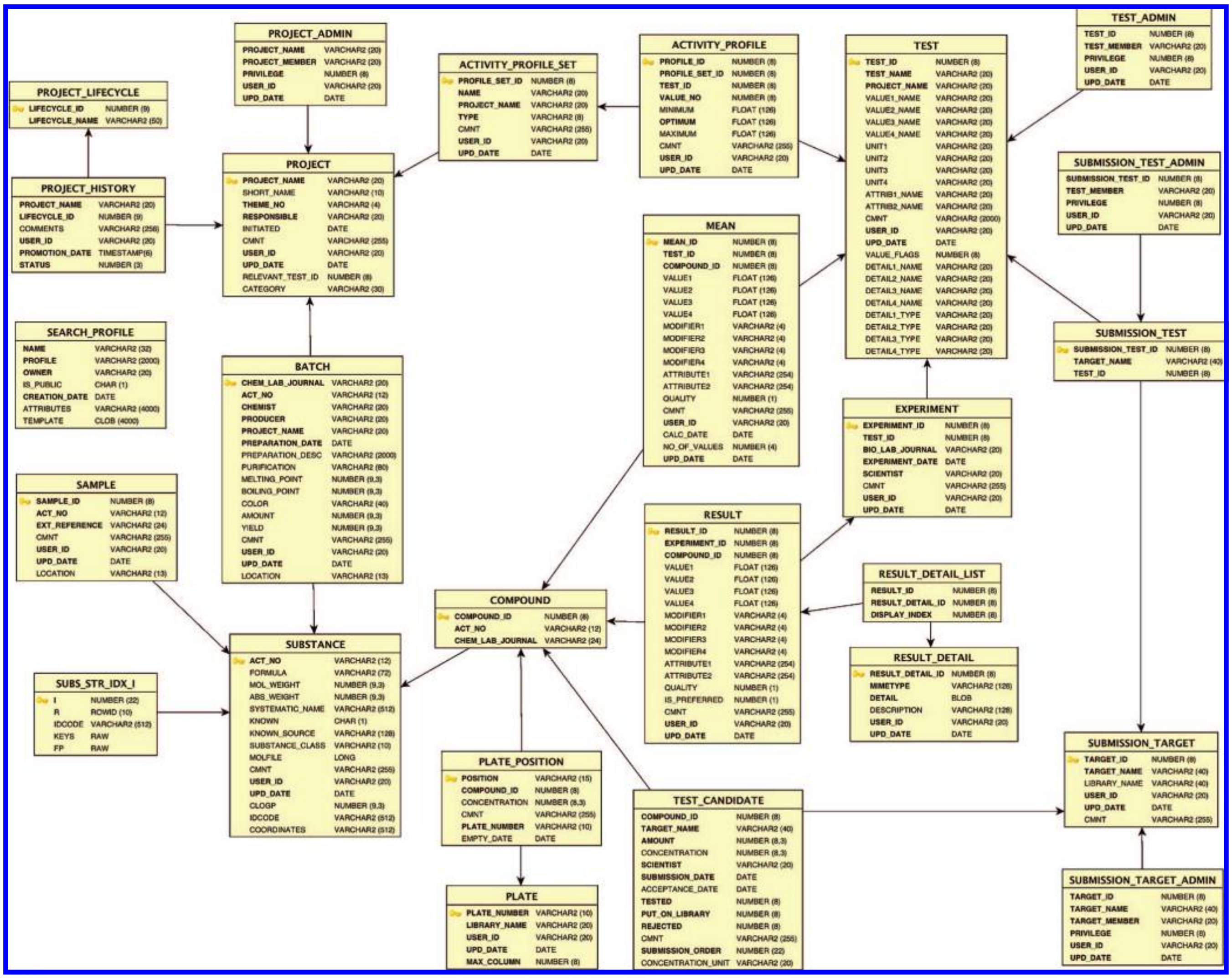


Figure 7. *OSIRIS database* - core entities.

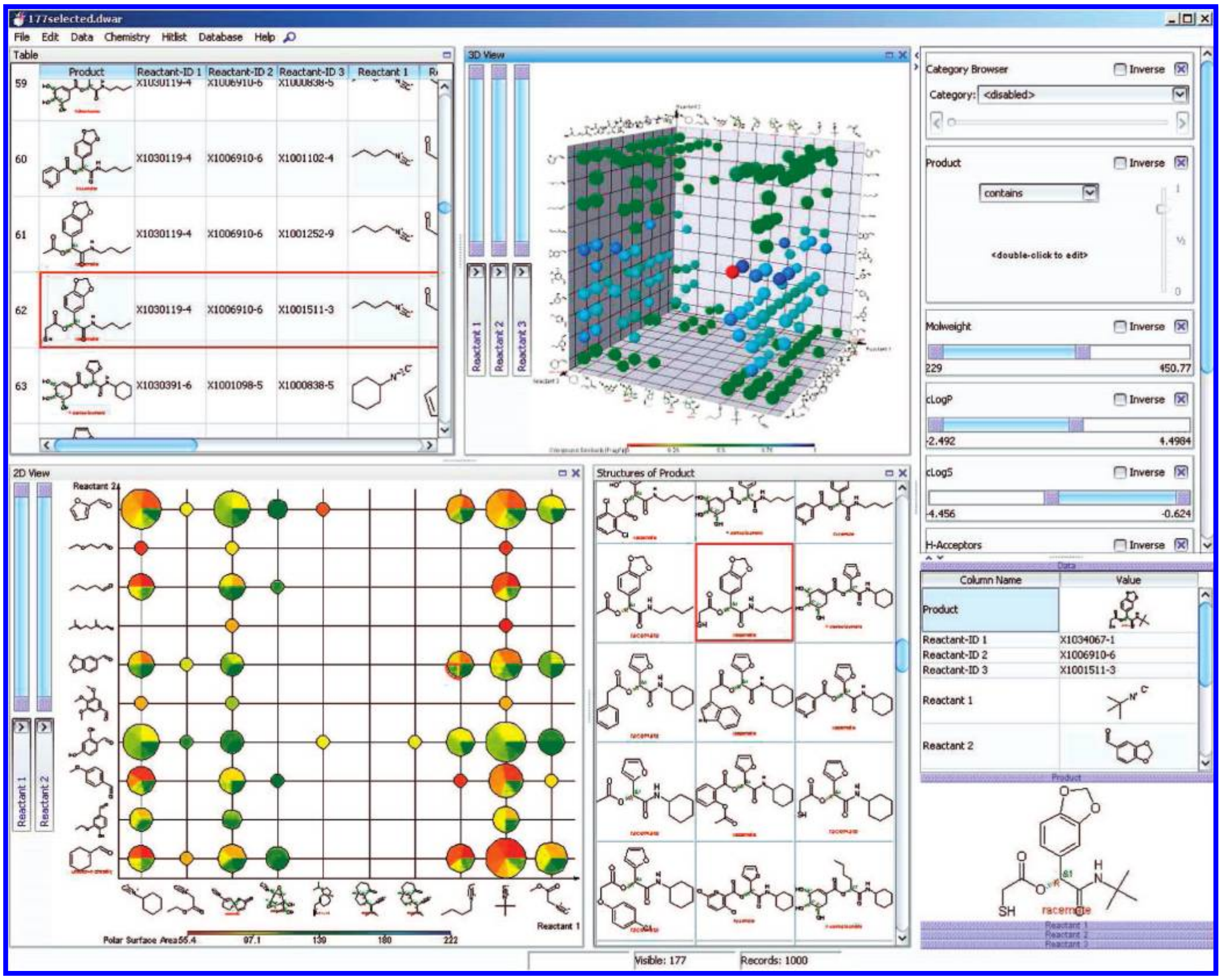
ture's descriptor keys are created and matched against the ones from the index table. A *SSS_MATCH* call only follows if all of the query fragment's keys are present in a record's descriptor key set. It is obvious that a careful selection of the descriptors is a key for the performance of the operator. We have chosen a combination of structure fragments and structure paths in order to satisfy this need.

The raw speed of such an implementation was acceptable: Most of the structure queries finished within 10−20 s on a molecule table containing 1 million structures. The speed could then be increased by a Java-to-native compilation of the Java classes. This led to a performance gain of a factor of 2−3.

After a recent migration to Oracle 10 g we experienced some challenges when the SSS operator was used in combination with other query criteria or table joins. The formerly rule based optimizer was discontinued in Oracle 10 g where now a cost based optimizer is responsible for defining the query's execution plan. Since the SSS is potentially a very costly operation, it is crucial that repeated calls during the execution of one query should be prevented. For instance, one needs to avoid that the SSS is called in an inner loop. Currently, we provide optimizer hints within the SQL statements of problematic queries to guide the cost based optimizer to influence the query execution plan accordingly. The proper way to configure the optimizer would be through Oracle's *Extensible Optimizer API.* We are currently investigating how to do this reliably.

Data Analysis Tool DataWarrior. A high throughput screening phase often starts off a new drug discovery project. Typically, a few hundred thousand compounds are tested on a biological target in order to find hits that serve as starting points for medicinal chemistry. Medicinal chemists synthesize thousands of compounds to further optimize the most promising chemical structures not only to reach high activity on the desired target but also to match a desired profile including side effects and physicochemical and pharmacokinetic properties.[42] Chemists often need to relate a dozen or more measured properties in regard to the structural differences of various compound families. The better a chemist understands trends within the data, the more efficient he/she will plan the next generation of compounds. Software tools that help chemists understand their data are a key element to accelerate the drug discovery workflow.

There was no commercially available product in 2002 that could visualize and interactively analyze alphanumerical and chemical data together. We considered Spotfire a product that might serve our needs after extensive enhancements on the chemistry side. However, we dropped the idea because of its elevated licensing costs and restriction to the Windows



	Product	Reactant-ID 1	Reactant-ID 2	Reactant-ID 3	Reactant 1
59		X1030119-4	X1006910-6	X1000838-5	
60		X1030119-4	X1006910-6	X1001102-4	
61		X1030119-4	X1006910-6	X1001252-9	
62		X1030119-4	X1006910-6	X1001511-3	
63		X1030391-6	X1001098-5	X1000838-5	

Category Browser ☐ Inverse
Category: <disabled>

Product ☐ Inverse
contains
<double-click to edit>

Molweight ☐ Inverse
229 450.77

cLogP ☐ Inverse
-2.492 4.4984

cLogS ☐ Inverse
-4.456 -0.624

H-Acceptors ☐ Inverse

Data

Column Name	Value
Product	
Reactant-ID 1	X1034067-1
Reactant-ID 2	X1006910-6
Reactant-ID 3	X1001511-3
Reactant 1	
Reactant 2	

Figure 8. *DataWarrior* - selecting from combinatorial library.

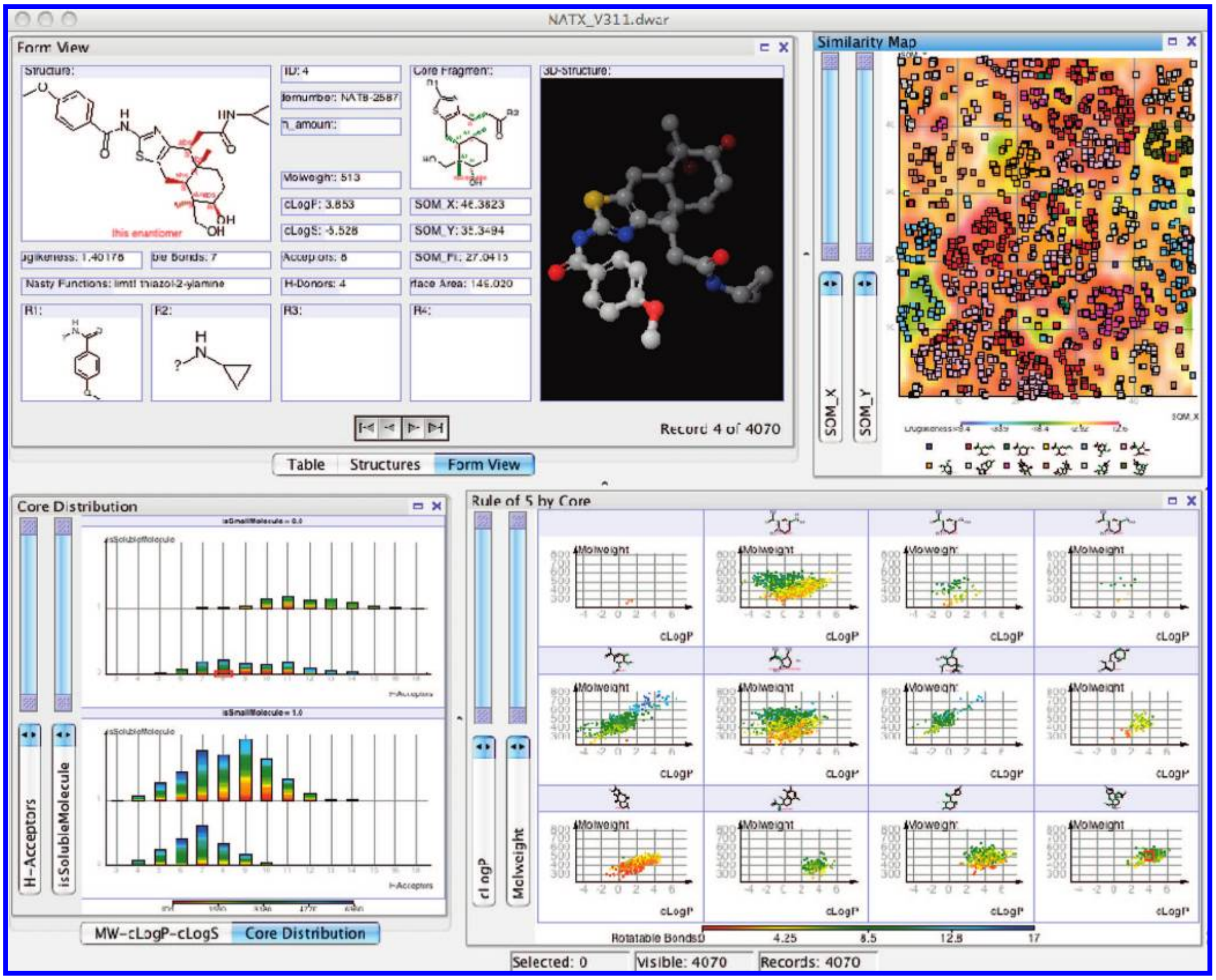
operating system. Instead we decided to develop a new tool, *DataWarrior*, to combine data visualization with chemical intelligence and direct database access.

Today *DataWarrior* is a universal drug discovery data analysis platform at Actelion. It has query interfaces to various in-house databases like the commercial chemicals, screening compounds, and microarray data. *DataWarrior* can read and write standard file formats like SMILES and SD-Files and can copy data to and from the clipboard. Data from different sources can be merged or appended to existing data. *DataWarrior*'s internal data representation consists of a table model with every row being represented by a record. Columns may contain special data types such as molecules, reactions, pictures, or HTML. Invisible columns may contain chemical descriptors, 2D- and 3D-atom coordinates that are logically associated with a parent column containing a chemical structure.

Once *DataWarrior* has read data from any source, it analyzes the content of every column. Its native data or query files contain view and filter settings, which cause *DataWarrior* to exactly show data, views, and filters as they were when the file was written (Figure 8). When opening a foreign file format, *DataWarrior* creates default views and filters depending on the nature of the data. Chemistry aware interactive views are a strength of the *DataWarrior* application. Zoomable 2D- or 3D- views allow exploration of correlations in all or in a subset of the data. All visible views show the same records, and records that are selected in one view are selected in all views. Marker colors, shapes, and sizes reflect numerical or category values or compound similarities of the respective records. 2D-views can display bar or pie charts, and entire views can be split into category-based subviews (Figure 9, bottom left and right). Other views are a chemistry aware table view, a dedicated chemical structure view, and a customizable form view which shows one record at a time (Figure 9, top left).

Filters are used to temporarily hide data from all views (Figure 8, top right). Most filters are associated with one data column and can hide records based on textual or numerical criteria, regular expressions, category types, list membership, substructure fragments, or compound similarities.

While *DataWarrior* was designed to work transparently with chemical data types, these are not a requirement. One laboratory is applying it for exploring and optimizing crystallization conditions for proteins, another one to draw conclusions from microarray data. To support multivariate data analysis, we built in the calculation of value and rank based correlation matrices (Bravais-Pearson and Spearman), principal component analysis (PCA), and the calculation of self-organizing maps[43] (SOM) (Figure 9, top right).



ID: 4
Molweight: 513
cLogP: 3.853
cLogS: -5.528
Acceptors: 8
H-Donors: 4
SOM_X: 46.3823
SOM_Y: 35.3494
Nasty Functions: limtl thiazol-2-ylamine

Figure 9. *DataWarrior* - analyzing commercial compounds.

DataWarrior has cheminformatics functionality built-in: It can enumerate product structures of a combinatorial library from a generic reaction and reactant lists. It can create a structure activity relationship table by detecting core-fragments and substituents of structure family within a data set. Various fingerprints and descriptors can be calculated and made available for similarity filtering, view coloring, clustering, diversity selection, PCAs, and SOMs. This includes the recently described Flexophore descriptor. Various physicochemical properties and toxicity risk indications can be calculated.

HTS-Witch. Lead structures, which form the chemical starting point of any drug discovery project, are often discovered by testing large libraries of diverse, lead-like compounds. Missing the most promising compounds or selecting improper lead structures could endanger a project's success. Obvious success factors are the quality of the compound library, a reliable assay procedure, and adequate error correction. The correct ranking of potential lead compounds is equally important yet often neglected. The frequent approach to value compounds based on activity or IC_{50} values rather than on ligand efficiencies creates a strong bias toward structures with high molecular weights.

Acknowledging the importance of high-throughput screening (HTS) evaluation and proper lead selection, HTS-Witch (Figure 10) was developed as an application to consistently normalize and visualize HTS results and to provide ligand efficiency values as main criteria for lead selection. There are four selectable main views which show the four subsequent stages of HTS results: raw data, corrected data, normalized data, and ligand efficiency values. The main views consist of a statistics table where individual plates may be selected for analysis, a plate view showing multiple plates with cells being color coded by the underlying numerical values, a plate-wise value distribution chart, and an overall value distribution histogram. The plate-wise and overall distribution views show control and compound values in different colors, which allow quick assessment of a screening's quality and hit-rate. HTS-Witch supports all file formats used by Actelion's screening machinery. It lets users define plate layouts, color scales, cutoff values, and switch off outliers and artifacts. Normalized results and ligand efficiency values are written into the OSIRIS database.

IC_{50}-Witch. A frequent procedure in biology is to measure the inhibition, remaining activity, agonistic, and antagonistic activity (in the following called effect) of a substance at different concentrations. The EC_{50} or IC_{50} value is the drug concentration at which the fitted curve crosses the 50% line of the maximum effect. This value represents the compound's fitness to act in a desired way on the target protein used in

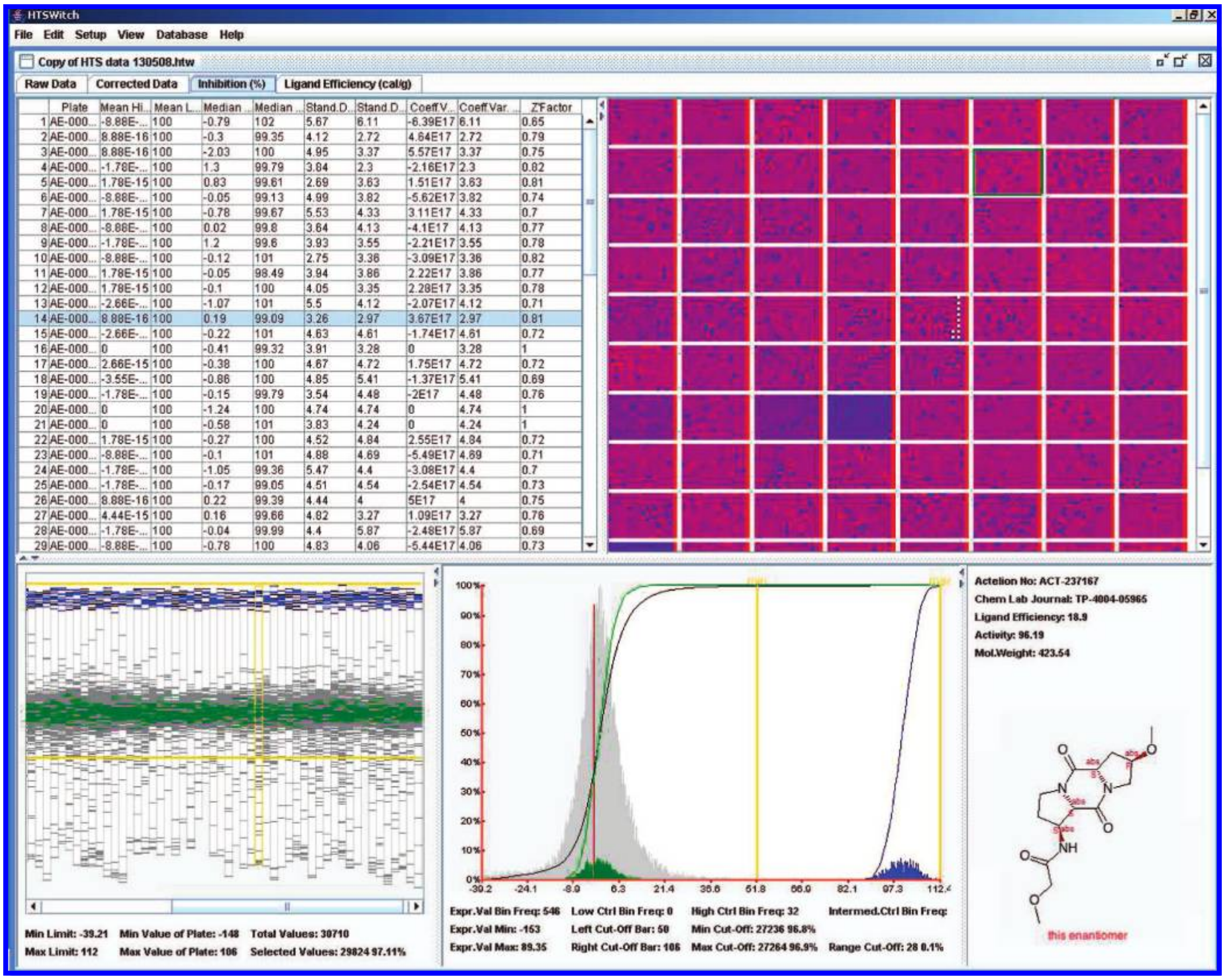


	Plate	Mean Hi...	Mean L...	Median ...	Median ...	Stand.D...	Stand.D...	Coeff.V...	Coeff.Var. ...	Z'Factor
1	AE-000...	-8.88E-...	100	-0.79	102	5.67	6.11	-6.39E17	6.11	0.65
2	AE-000...	8.88E-16	100	-0.3	99.35	4.12	2.72	4.64E17	2.72	0.79
3	AE-000...	8.88E-16	100	-2.03	100	4.95	3.37	5.57E17	3.37	0.75
4	AE-000...	-1.78E-...	100	1.3	99.79	3.84	2.3	-2.16E17	2.3	0.82
5	AE-000...	1.78E-15	100	0.83	99.61	2.69	3.63	1.51E17	3.63	0.81
6	AE-000...	-8.88E-...	100	-0.05	99.13	4.99	3.82	-5.62E17	3.82	0.74
7	AE-000...	1.78E-15	100	-0.78	99.67	5.53	4.33	3.11E17	4.33	0.7
8	AE-000...	-8.88E-...	100	0.02	99.8	3.64	4.13	-4.1E17	4.13	0.77
9	AE-000...	-1.78E-...	100	1.2	99.6	3.93	3.55	-2.21E17	3.55	0.78
10	AE-000...	-8.88E-...	100	-0.12	101	2.75	3.36	-3.09E17	3.36	0.82
11	AE-000...	1.78E-15	100	-0.05	98.49	3.94	3.86	2.22E17	3.86	0.77
12	AE-000...	1.78E-15	100	-0.1	100	4.05	3.35	2.28E17	3.35	0.78
13	AE-000...	-2.66E-...	100	-1.07	101	5.5	4.12	-2.07E17	4.12	0.71
14	AE-000...	8.88E-16	100	0.19	99.09	3.26	2.97	3.67E17	2.97	0.81
15	AE-000...	-2.66E-...	100	-0.22	101	4.63	4.61	-1.74E17	4.61	0.72
16	AE-000...	0	100	-0.41	99.32	3.91	3.28	0	3.28	1
17	AE-000...	2.66E-15	100	-0.38	100	4.67	4.72	1.75E17	4.72	0.72
18	AE-000...	-3.55E-...	100	-0.86	100	4.85	5.41	-1.37E17	5.41	0.69
19	AE-000...	-1.78E-...	100	-0.15	99.79	3.54	4.48	-2E17	4.48	0.76
20	AE-000...	0	100	-1.24	100	4.74	4.74	0	4.74	1
21	AE-000...	0	100	-0.58	101	3.83	4.24	0	4.24	1
22	AE-000...	1.78E-15	100	-0.27	100	4.52	4.84	2.55E17	4.84	0.72
23	AE-000...	-8.88E-...	100	-0.1	101	4.88	4.69	-5.49E17	4.69	0.71
24	AE-000...	-1.78E-...	100	-1.05	99.36	5.47	4.4	-3.08E17	4.4	0.7
25	AE-000...	-1.78E-...	100	-0.17	99.05	4.51	4.54	-2.54E17	4.54	0.73
26	AE-000...	8.88E-16	100	0.22	99.39	4.44	4	5E17	4	0.75
27	AE-000...	4.44E-15	100	0.16	99.66	4.82	3.27	1.09E17	3.27	0.76
28	AE-000...	-1.78E-...	100	-0.04	99.99	4.4	5.87	-2.48E17	5.87	0.69
29	AE-000...	-8.88E-...	100	-0.78	100	4.83	4.06	-5.44E17	4.06	0.73

Min Limit: -39.21 Min Value of Plate: -148 Total Values: 30710
Max Limit: 112 Max Value of Plate: 106 Selected Values: 29824 97.11%

Expr.Val Bin Freq: 546 Low Ctrl Bin Freq: 0 High Ctrl Bin Freq: 32 Intermed.Ctrl Bin Freq:
Expr.Val Min: -153 Left Cut-Off Bar: 50 Min Cut-Off: 27236 96.8%
Expr.Val Max: 89.35 Right Cut-Off Bar: 106 Max Cut-Off: 27264 96.9% Range Cut-Off: 28 0.1%

Actelion No: ACT-237167
Chem Lab Journal: TP-4004-05965
Ligand Efficiency: 18.9
Activity: 96.19
Mol.Weight: 423.54

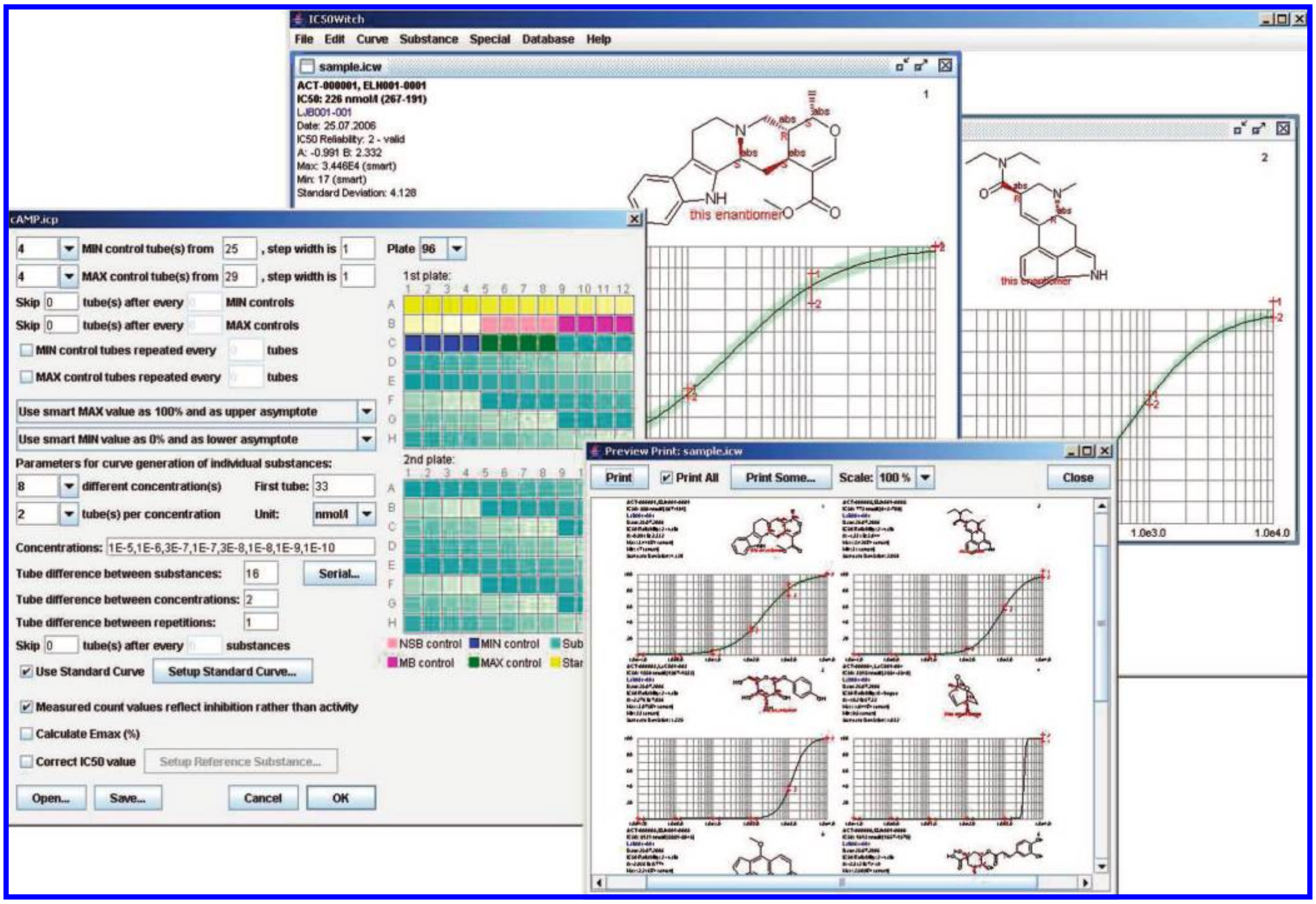
Figure 10. *HTS-Witch* - analyzing high-throughput screening results.

the assay. Typically, the experiments are performed on many compounds in parallel using 96 or 384 well microtiter plates. A plate reader measures a physical value that is proportional to the effect of the substance. In some cases, the effect is indirectly determined from a competitive binding experiment, where a reference curve is measured that shows the relationship between the measured physical property and the desired effect.

IC_{50}-Witch (Figure 11) was designed to handle the entire data flow from measured raw values to the upload of IC_{50} values into the OSIRIS database. A built-in plate layout designer allows flexible definition of how compounds and controls are distributed over the plates, which concentrations are used, whether a standard curve is used, and whether controls are repeated over multiple plates. *IC_{50}-Witch* supports two- or four-parameter curve fitting and various modes to determine and apply compound specific minimum and maximum values. It determines the maximum effect of agonists and can normalize IC_{50} values taking into consideration known inhibition of reference compounds. All raw data files created by Actelion's plate readers and top counters can directly be opened. *IC_{50}-Witch* displays fitted curves, confidence intervals, standard deviations, and substance specific information from the database. Users can check individual curves, switch off single astray data points, adapt curve specific calculation settings, and assign reliability values or comment individual results before uploading the IC_{50} values with comments and quality assignments into the database.

SubSub. As part of the drug discovery process chemists steadily synthesize new compounds, which biologists then test. Chemists and biologists work on scientific projects for a particular target or indication. When a physical compound is sent, a solution with a defined concentration is prepared, and a fraction of this solution may be added to a project specific plate library. In the early days of our drug discovery department, the physical process of submitting compounds was accompanied by project specific Excel sheets, which were shared among chemists and biologists. Chemists added new compounds and their amounts; the biologists removed these compounds from the list once they were tested.

Using Excel sheets over the network is inefficient, error prone, and inconsistently handled between projects. We developed a dedicated application to simplify and unify the process of substance submission (*SubSub*). In *SubSub* users define lists of biological assays as submission targets, and chemists assign their freshly synthesized compounds to one or more submission targets. *SubSub* suggests the amount of substance to weigh in, calculates needed DMSO amounts, controls the compound positioning on plate libraries, and



ACT-000001, ELN001-0001
IC50: 226 nmol/l (267-191)
LJB001-001
Date: 25.07.2006
IC50 Reliability: 2 - valid
A: -0.991 B: 2.332
Max: 3.446E4 (smart)
Min: 17 (smart)
Standard Deviation: 4.128

cAMP.icp

4 MIN control tube(s) from 25, step width is 1 Plate 96
4 MAX control tube(s) from 29, step width is 1
Skip 0 tube(s) after every MIN controls
Skip 0 tube(s) after every MAX controls
☐ MIN control tubes repeated every tubes
☐ MAX control tubes repeated every tubes
Use smart MAX value as 100% and as upper asymptote
Use smart MIN value as 0% and as lower asymptote
Parameters for curve generation of individual substances:
8 different concentration(s) First tube: 33
2 tube(s) per concentration Unit: nmol/l
Concentrations: 1E-5,1E-6,3E-7,1E-7,3E-8,1E-8,1E-9,1E-10
Tube difference between substances: 16 Serial...
Tube difference between concentrations: 2
Tube difference between repetitions: 1
Skip 0 tube(s) after every substances
☑ Use Standard Curve Setup Standard Curve...
☑ Measured count values reflect inhibition rather than activity
☐ Calculate Emax (%)
☐ Correct IC50 value Setup Reference Substance...
Open... Save... Cancel OK

1st plate:
2nd plate:
NSB control MIN control Sub
MB control MAX control Star

Figure 11. *IC$_{50}$-Witch* with print-preview and plate-definition-dialog.

allows biologists to track in assay specific views which compounds are to be tested.

Actelion3D. Actelion's molecular modeling tool (Figure 12) is used by chemists for 3D viewing, manipulating, superposing proteins and ligands, and for docking ligands into proteins. A user can open any protein structure in the docking module from the PDB database or from our in-house protein database. One can also drag and drop a ligand from an external application, draw a new ligand structure, or specify an Actelion number as compound reference. The ligand is automatically positioned within the protein's cavity replacing the original ligand, if there had been one. At this point, the user can start either manual or automated docking without any parametrization or protein or ligand preparation, which is often needed in commercial programs. The application returns the best pose and score.

In the second module the user can do a flexible superposition of two ligands. The program will automatically suggest a superposition pattern based on atom types, and a genetic algorithm will then optimize bond torsions to get the best match.

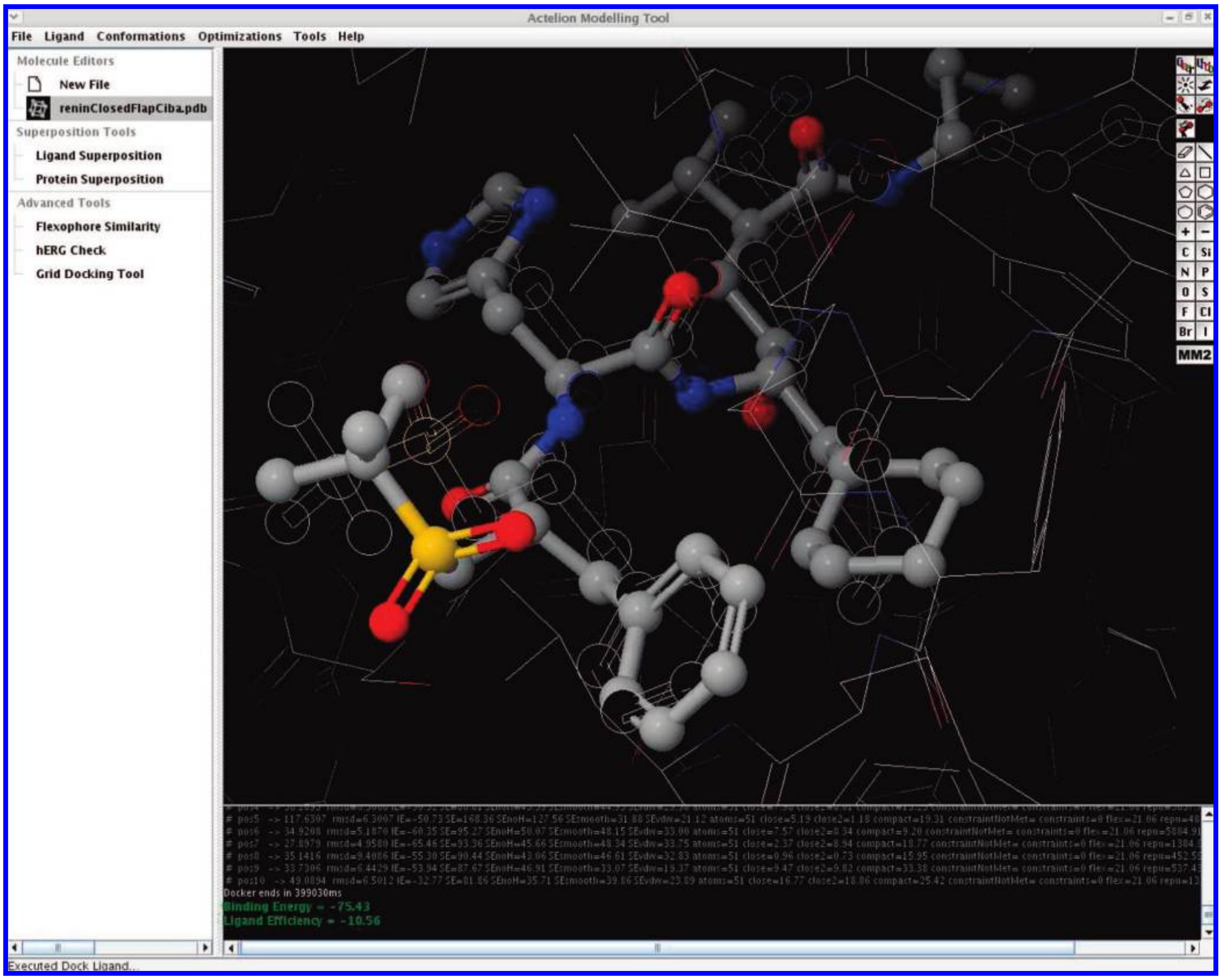
A third module lets the user launch a batch docking job on Actelion's computing grid. One can select a protein structure and define a list of potential ligands. After the job is finished, he can sort the ligands by score and mode of action and can examine docking poses of each ligand.

Document Repository. In 2002, when the OSIRIS database was already well established, biologists expressed the wish to handle research documents in a more organized fashion rather than simply storing them on network drives. We developed a multicomponent solution to tackle this request. A special printer driver was written, that instead of sending the information to a physical printer, converts the printing stream on-the-fly to a PDF document, which is stored at a central location. In addition, all text information is extracted and copied to an instance of the Oracle Context Engine, which is a full text search engine from Oracle Corporation. We also developed a Web-based query tool providing meta data search and full text search functionality to the documents. A search result consists of a list of matching documents with direct links to the underlying PDF documents.

ActWiki. An important aspect of research is the exchange of information between people, laboratories, and projects. Information regarding laboratory equipment, projects statuses, workflows, protocols, group meetings, etc. does not fit into structured databases nor does it belong into laboratory notebooks or research reports. Nevertheless, this information must be shared.

We installed the open source JSP-Wiki and extended it to fit our needs.[44] We defined team spaces for each group and research project. Access privileges to the team spaces ensure limited access, where sensitive information is exchanged. Every team space contains a list of members with links to their contact information.

We wrote various plugins to fully integrate the *ActWiki* into the OSIRIS environment. All Actelion's compound identifiers on any *ActWiki* page are automatically associated with links that provide immediate access to the chemical



```
# pos5  -> 117.6307  rmsd=6.3007 IE=-50.73 SE=168.36 SEnoH=127.56 SEsmooth=31.88 SEvdw=21.12 atoms=51 close=5.19 close2=1.18 compact=19.31 constraintNotMet= constraints=0 flex=21.06 repu=48
# pos6  -> 34.9208  rmsd=5.1870 IE=-60.35 SE=95.27 SEnoH=50.07 SEsmooth=48.15 SEvdw=33.00 atoms=51 close=7.57 close2=0.34 compact=9.20 constraintNotMet= constraints=0 flex=21.06 repu=5884.91
# pos7  -> 27.8979  rmsd=4.0580 IE=-65.46 SE=93.36 SEnoH=45.66 SEsmooth=48.34 SEvdw=33.75 atoms=51 close=2.37 close2=8.94 compact=18.77 constraintNotMet= constraints=0 flex=21.06 repu=1384.
# pos8  -> 35.1416  rmsd=9.4086 IE=-55.30 SE=90.44 SEnoH=43.06 SEsmooth=46.61 SEvdw=32.83 atoms=51 close=0.96 close2=0.73 compact=15.95 constraintNotMet= constraints=0 flex=21.06 repu=452.59
# pos9  -> 33.7306  rmsd=6.4429 IE=-53.94 SE=87.67 SEnoH=46.91 SEsmooth=33.07 SEvdw=19.37 atoms=51 close=9.47 close2=9.82 compact=33.38 constraintNotMet= constraints=0 flex=21.06 repu=537.43
# pos10  -> 49.0894  rmsd=6.5012 IE=-32.77 SE=81.86 SEnoH=35.71 SEsmooth=39.86 SEvdw=23.89 atoms=51 close=16.77 close2=18.86 compact=25.42 constraintNotMet= constraints=0 flex=21.06 repu=13
Docker ends in 399030ms
Binding Energy = -75.43
Ligand Efficiency = -10.56
```

Figure 12. *Actelion3D* - showing docking results.

structure and the compound's biological results. References to laboratory notebook entries are linked to the corresponding documents.

We extended the search engine such that one query will search the *ActWiki*, our document management system, and the employees database. Searching for a compound identifier or notebook reference will also yield the same results, provided the access rights are sufficient. The benefit of such a tool was immediately obvious to the end-user, but the benefit of such a tool is only proportional to the amount of information it contains.

Another pharmaceutical company successfully running a Wiki is Pfizer with its PfizerPedia.[45]

Specialized Tools and Databases. In addition to the software solutions described above, which are used by a fair fraction of the drug discovery personnel, quite a few smaller tools and databases were developed that serve a local need and often adhere to legal requirements. Many are inventory databases of special resources like laboratory animals, cell and plasmids, or radio-active compounds. Others databases manage results which are not related to in-house compounds or that require special handling. Some examples are the conditions and observations of protein crystallization experiments, protein X-ray structures, and gene expression data from PCR or microarray experiments.

We developed a computing grid framework to run tasks written in Java on 300 computers in parallel. The Windows only grid client software consists of a simple *Windows Service* written in C++ that launches a local Java based grid controller program which opens an RMI connection to a central grid server. The server maintains a list of tasks awaiting completion. A task is defined by specific input data and is assigned to a specific process. Processes are essentially programs written in Java and stored in a.jar file on the grid server database. If the grid client computer matches pre-defined, process specific hardware requirements, then the server sends the.jar along with the input data to the client. Then the client grid controller spawns the.jar file's program within a Java sandbox as a low priority process, which is almost unnoticeable to the user. Upon completion of the process, a result is returned to the grid server and a new task is requested. The grid is frequently used for protein docking and Flexophore[34] comparisons of commercial compounds or virtual libraries.

Drug Discovery IM Staff. In 1998, an organic chemist with 10 years of software development experience, who had spent 5 years in a drug discovery informatics department of a large pharmaceutical company, laid the foundation of Actelion's drug discovery IM system. In 2000 he was joined by a professional software engineer with chemistry and cheminformatics background. In 2003 two more people joined the group, a professional software engineer and a software developer with a scientific background. In 2004 Actelion acquired Axovan, a small biotech company, and hired a complete group of scientists from another biotech company. Three people being formerly involved in discovery informatics of these companies joined Actelion's team. In 2007 one person left and two more were hired, a chemist with software engineering experience and a physicist with software engineering and teaching experience. Currently, the drug discovery IM team consists of eight members. Besides a strong informatics background, they cover a broad scope of natural science with master degrees in information science, mathematics, chemistry, pharmacy, and archeological science. The team not only develops and maintains the scientific informatics systems used by the 300 plus members of the drug discovery department but also promotes tool usage and educates the scientists on their tools and databases.

4. RESULTS

Within 10 years, after investing about 25 man-years of development efforts, a drug discovery IM system was implemented that supports all major activities of Actelion's drug discovery process. The system consists of a broad span of diverse components, from a chemistry aware Oracle Cartridge to multidimensional visualization software and from electronic laboratory notebooks to an interactive modeling application. The more diverse the different tools and databases are, the more homogenious are the technologies used to implement them. Apart from some platform specific code in C++, the large majority of source code was written in Java. It amounts to approximately 800,000 lines and about 10,000 Java classes. To put this into perspective: The source code for WindowsXP includes about 40 million lines. Core cheminformatics or user interface classes were designed to hide complexity behind simple interfaces. A growing toolbox of such classes permits the quick development of new applications through reusing existing building blocks. The Canonizer class, for instance, is used at 206 different locations to generate *idcodes*.

Many of the tools were developed for a specific purpose and often for a small number of users. These tools include databases for tracking animals, radioactive compounds, or observations of protein crystallization experiments. Typically, they occupy a developer for a few weeks or months until they are set productive and used for years without many adaptations. Other applications evolve over time and steadily adapt to changing requirements or growing expectations of a large user community. Introducing a new wanted feature is often a matter of a few days or even hours. Short response times of user requests have changed the scientist's attitude toward research software. They request new tools or new features more frequently. In the opinion of the authors the real value of Actelion's approach is the ability to quickly provide custom tailored software on demand.

Some example applications based on OSIRIS technology are publicly available at http://www.cheminformatics.ch.

5. CONCLUSION

We consider our experience valid proof that a high degree of in-house software development may be a justifiable alternative to the supposedly less risky approach of licensing best-of-breed systems and then to substantially invest into their integration. A consistent user-interface, a closer match to the user's requirements, a lower learning curve, and eventually a higher user satisfaction may lead to wider and more efficient software usage and therefore to an overall acceleration of the drug discovery process.

ACKNOWLEDGMENT

We thank our drug discovery management Walter Fischli, Thomas Weller, and Martine Clozel and our CEO Jean-Paul Clozel for their trust and support. Without their confidence it would have never been possible to pursue a potentially risky strategy that is radically different from the course the rest of the industry is taking. Furthermore, we thank Bernard Przybylski for user education, user support, data curation, and for running the Oracle databases for over 5 years. We also thank Isabelle Giraud for successfully educating and tutoring our users, promoting our applications within the user community, and for delivering many useful enhancement requests. We thank Axel Klenk and Tobias Fink for their manifold contributions to the OSIRIS system. We thank Christopher Snyder for proofreading and correcting the manuscript.

REFERENCES AND NOTES

(1) Searls, D. B. Data integration: challenges for drug discovery. *Nat. Rev. Drug Discovery* **2005**, *4*, 45–58.
(2) Brown, F. Saving big pharma from drowning in the data pool. *Drug Discovery Today* **2006**, *11*, 1043–1045.
(3) Stahura, F. L.; Bajorath, J. Bio- and chemo-informatics beyond data management: crucial challenges and future opportunities. *Drug Discovery Today* **2002**, *1*, 41–47.
(4) Claus, B. L.; Underwood, D. J. Discovery informatics: its evolving role in drug discovery. *Drug Discovery Today* **2002**, *7*, 957–966.
(5) Shen, J.; Polyakov, V.; Rachapudi, R.; You, T. Decision Support Systems in Drug Discovery. *Mini-Rev. Med. Chem.* **2004**, *4*, 1019–1028.
(6) Stahl, M.; Guba, W.; Kansy, M. Integrating molecular design resources within modern drug discovery research: the Roche experience. *Drug Discovery Today* **2006**, *11*, 326–333.
(7) Fay, N. The role of the informatics framework in early lead discovery. *Drug Discovery Today* **2006**, *11*, 1075–84.
(8) Searls, D. B. A view from the dark side. *PLoS Comput Biol.* **2007**, *3*, e105.
(9) Freitag, B.-J. ChemInfo - A Bridge Connecting Data Islands. Presented at the 2006 ICIC Meeting, Nimes, France, October 22−25, 2006. ICIC Web site. http://www.infonortics.com/chemical/ch06/06chempro.html (accessed Nov 26, 2008).
(10) Agrafiotis, D. K.; Alex, S.; Dai, H.; Derkinderen, A.; Farnum, M.; Gates, P.; Izrailev, S.; Jaeger, E. P.; Konstant, P.; Leung, A.; Lobanov, V. S.; Marichal, P.; Martin, D.; Rassokhin, D. N.; Shemanarev, M.; Skalkin, A.; Stong, J.; Tabruyn, T.; Vermeiren, M.; Wan, J.; Xu, X. Y.; Yao, X. Advanced Biological and Chemical Discovery (ABCD): Centralizing DiscoveryKnowledge in an Inherently Decentralized World. *J. Chem. Inf. Model.* **2007**, *47*, 1999–2014.
(11) Gobbi, A.; Funeriu, S.; Ioannou, J.; Wang, J.; Lee, M.-L.; Palmer, C.; Bamford, B.; Hewitt, R. Process-Driven Information Management System at a Biotech Company: Concept and Implementation. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 964–975.
(12) Defanti, T. A.; Brown, M. D. Visualization in scientific computing. In *Advances In Computers*, 1st ed.; Academic Press Professional, Inc.: San Diego, CA, U.S.A., 1991; pp 247−305.
(13) Huber, W.; Li, X.; Gentleman, R.; Visualizing Data. In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*; Gentleman, R., Carey, V. P., Huber, W., Irizarry, R. A., Dudoit, S., Eds.; Springer: London, 2005.
(14) Accelrys Ltd. http://accelrys.com/ (accessed Aug 25, 2008).
(15) Advanced Chemistry Development. http://www.acdlabs.com/ (accessed Aug 25, 2008).
(16) CambridgeSoft. http://www.cambridgesoft.com/ (accessed Aug 25, 2008).
(17) ChemAxon. http://www.chemaxon.com/ (accessed Aug 26, 2008).

(18) Chemical Computing Group. http://www.chemcomp.com/ (accessed Aug 25, 2008).
(19) Daylight. http://www.daylight.com/ (accessed Aug 25, 2008).
(20) ID Business Solutions. http://www.idbs.com/ (accessed Aug 25, 2008).
(21) OpenEye. http://www.eyesopen.com/ (accessed Aug 26, 2008).
(22) Spotfire. http://spotfire.tibco.com/index.cfm (accessed Aug 25, 2008).
(23) Symyx. http://www.symyx.com/ (accessed Aug 25, 2008).
(24) *ChemOffice, Version 4.5*; CambridgeSoft: Cambridge, MA 1998.
(25) Java SE. http://java.sun.com/javase/ (accessed Aug 29, 2008).
(26) Java Database Connectivity. http://java.sun.com/products/jdbc/ (accessed Nov 30, 2008).
(27) Remote Method Invocation. http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp (accessed Aug 25, 2008).
(28) CORBA. http://www.omg.org/gettingstarted/corbafaq.htm (accessed Aug 25, 2008).
(29) Apache Tomcat. http://tomcat.apache.org/ (accessed Nov 30, 2008).
(30) JavaServer Pages Technology. http://java.sun.com/products/jsp/ (accessed Aug 25, 2008).
(31) Java Servlet Technology. http://java.sun.com/products/servlet/ (accessed Aug 25, 2008).
(32) Struts. http://struts.apache.org/ (accessed Aug 25, 2008).
(33) Oracle. http://www.oracle.com/index.html (accessed Aug 29, 2008).
(34) von Korff, M.; Freyss, J.; Sander, T. Flexophore, a new versatile 3D pharmacophore descriptor that considers molecular flexibility. *J. Chem. Inf. Model.* **2008**, *48*, 797–810.
(35) Knime. http://www.knime.org/ (accessed Aug 25, 2008).
(36) Pipeline Pilot. http://accelrys.com/products/scitegic/ (accessed Aug 25, 2008).
(37) MDL file formats. http://www.mdli.com/solutions/white_papers/ctfile_formats.jsp (accessed Aug 25, 2008).
(38) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comput. Sci.* **1988**, *28*, 31–36.
(39) von Korff, M.; Sander, T. Toxicity-Indicating Structural Patterns. *J. Chem. Inf. Model.* **2006**, *46*, 536–544.
(40) Ponder, J. W.; Richards, F. M. An Efficient Newton-like Method for Molecular Mechanics Energy Minimization of Large Molecules. *J. Comput. Chem.* **1987**, *8*, 1016–1024.
(41) Gohlke, H.; Hendlich, M.; Klebe, G. Knowledge-based scoring function to predict protein-ligand interactions. *J. Mol. Biol.* **2000**, *295*, 337–56.
(42) Oprea, T. Lead Structure Searching: Are We Looking at the Appropriate Property. *J. Comput.-Aided Mol. Des.* **2002**, *16*, 325–334.
(43) Kohonen, T. *Self-Organizing Maps*, 3rd ed.; Springer: New York, 2001.
(44) JSPWiki. http://www.jspwiki.org/ (accessed Aug 25, 2008).
(45) Mullin, R. Seeing The Forest At Pfizer. A radical knowledge-sharing initiative takes hold at the world's largest drugmaker. *Chem. Eng. News* **2007**, *85* (36), 29.

CI800305F