AFRL-IF-RS-TR-2005-245
Final Technical Report
June 2005

INTEGRATION OF ANALYTIC AND SYNTHETIC BIOSYSTEM MODELS AND DATA

BBN Technologies

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. M304

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2005-245 has been reviewed and is approved for publication

APPROVED: /s/

PETER J. ROCCI, JR.
Project Engineer

FOR THE DIRECTOR: /s/

JOSEPH CAMERA, Chief
Information & Intelligence Exploitation Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 074-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** JUNE 2005	**3. REPORT TYPE AND DATES COVERED** Final Sep 01 – Dec 04

4. TITLE AND SUBTITLE
INTEGRATION OF ANALYTIC AND SYNTHETIC BIOSYSTEM MODELS AND DATA

5. FUNDING NUMBERS
C - F30602-01-C-0210
PE - 61101E
PR - BIOC
TA - M3
WU - 04

6. AUTHOR(S)
Jonathan Delatizky and Jonathan Webb

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
BBN Technologies
10 Moulton Street
Cambridge Massachusetts 02138-1119

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFED
525 Brooks Road
Rome New York 13441-4505

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2005-245

11. SUPPLEMENTARY NOTES

AFRL Project Engineer: Peter J. Rocci/IFED/(315) 330-4654/ Peter.Rocci@rl.af.mil

12a. DISTRIBUTION / AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 Words)*

This effort, funded under DARPA's Biocomputing (BioComp) program, was directed at providing data services for the BioSpice infrastructure. BioSpice is an environment akin to the Electrical Engineering SPICE modeling and simulation package, in which an open environment and standards-based modularity enable an enormous range of tools for the development and understanding of electronic circuits to be applied without restrictions resulting from proprietary or closed interfaces. BBN also teamed with the University of Pennsylvania to develop an intuitive tool called BioSketchpad, for creating and parameterizing models and connecting the resulting models to a simulator. A modeling language (Systems Biology Markup Language (SBML), was also developed for representing interesting molecular biology models and to support model interchange between different tools. The SBML effort required collaboration with a separate systems biology development effort (which had created the first version of SBML) that was an outgrowth of other research efforts.

14. SUBJECT TERMS
Bio-Computation, BioSpice, Open Source, Biological Models, Systems Biology Markup Language

15. NUMBER OF PAGES
79

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

1 Introduction

BBN Technologies (BBNT) contract under the DARPA BioComp program was initially directed at providing data services for the embryonic BioSpice effort. BioSpice was envisaged by DARPA as an environment akin to the Electrical Engineering SPICE modeling and simulation package, in which an open environment and standards-based modularity enable an enormous range of tools for the development and understanding of electronic circuits to be applied without restrictions resulting from proprietary or closed interfaces. Since nothing of the sort existed in the open source biological modeling arena, DARPA's goal was to jump-start its development and subsequently transition its further evolution to the wider biological modeling and associated engineering communities.

BBNT's role was envisaged as supporting the system integrator, SRI, and the numerous biologists and modelers who would be developing and integrating simulators, analytical tools, models, and other elements, in data management and data flow within the BioSpice system as it evolved.

2 Overview

In the first year of the program, BBNT performed the following tasks in support of this role:

- Surveyed existing database systems used by biologists and modelers and published a listing categorizing databases by content, organization, and access mechanisms
- Stood up and ran a database working group, with participation from most performers under the program, to assess BioSpice needs for data management and to direct develop of a data management architecture
- Published a white paper defining a data management architecture for BioSpice based on these inputs
- Proposed and performed an initial implementation of a Java-base class library for the realization of this architecture.

In addition, recognizing that there was a dearth of tools for visual modeling of molecular biological systems, we teamed with the University of Pennsylvania to initiate development of an intuitive tool for creating and parameterizing models and connecting the resulting models to a simulator. The specific simulator chosen was Penn's CHARON, a hybrid simulation environment well suited to the discontinuities that occur in the kind of biological models of interest. The tool we developed was called BioSketchpad.

BBNT's efforts in the subsequent year were directed by DARPA PM Dr. Sri Kumar towards further development of the BioSketchpad modeling tool and into development of a modeling language - Systems Biology Markup Language (SBML) - for representing interesting molecular biology models and to support model interchange between different tools. The

SBML effort required collaboration with a separate systems biology development effort (which had created the first version of SBML) that was an outgrowth of other research efforts. The collaboration between these researchers and the BioSpice teams led to both an acceleration in SBML evolution as well as an increase in the quality of the resulting language specification.

In the third year, our efforts were focused by Dr. Kumar on continuing the model language development and on the creation of tools for validating models expressed in SBML. We developed a set of validation rules for models expressed in SBML. In further support of this effort, we subcontracted with the Institute of Software Integrated Systems, Vanderbilt University, to deliver an implementation of the validation rules using XSLT.

The remainder of this report describes each of these efforts. Additional detail will be found in the white papers and other documents prepared during contract execution, copies of which are attached to this report.

3 Data Management

3.1 Database Working Group

The data management efforts in the first year were focused around the Database Working group (DBWG), an informal organization within the BioSpice performers which included representatives of almost every research group. DBWG met by conference call every two weeks through the first year, and less frequently for the next six months. Its functions were subsequently inherited by the Systems Engineering Task Force (SEPDTF) and the Experimentalists' Working Group (EWG), which carried the needs for consistent data definitions into the BioSpice product development cycles.

The concerns of the DBWG included

- What data sources and databases were available
- What the needs of the BioSpice modelers would be for such data
- What data would be generated by the community
- How might the data move around an integrated system
- What the implications were for the system integration process

3.2 Database Summary

The first question was addressed by conducting a survey of relevant database and data sources. Most of the candidate sources were identified by participants in the DBWG discussions and were reviewed by BBNT. Drafts of the resulting survey were circulated for comment and a final version published and delivered in April 2002. A copy is attached to this document.

The results of the survey provided an input to the SRI data warehouse development task, providing some additional guidance as to which databases would be of most value to the BioSpice community.

3.3 Data Management Infrastructure

The DBWG discussions also drove development of a proposed data architecture for BioSpice, which we called the Data Management Infrastructure (DMI). The DMI proposal was based on the following perceived requirements (excerpted from the white paper):

- **Heterogeneous Data Access.** The Bio-SPICE DMI must support access to many different types of data, from sources of differing structure, in a wide range of locations. Some will be under the control of the Bio-SPICE Program or its participants, while many others will not.
- **Flexible and Extensible.** Each user may work with different organisms, models, databases, etc. The system needs to be able to incorporate and adapt to these individual needs. Users must be able to adapt the system themselves.
- **Data Integrity.** The data managed by the DMI must be protected against corruption. Internal consistency must be guaranteed. In addition, data extracted from other sources must be traceable to their origins.
- **Defined Semantics.** Semantic relationships between data elements managed by the system must be maintained. They must also be available to other Bio-SPICE code for computational purposes and in human-readable form for browsing.
- **Performance.** The DMI must provide adequate performance for both local and network operations. When the nature of a data operation is likely to introduce significant delays, the DMI must inform the user and provide a mechanism to cancel or modify the operation to improve system responsiveness.
- **Access Controls and Data Sharing.** The DMI must provide mechanisms for identification and authentication of users, and selective sharing of contents with identified or general users (i.e. discretionary access control to its content).
- **Version control.** The DMI must have the ability to manage multiple versions of the objects in its stores.
- **Update Mechanisms.** The DMI must provide mechanisms to update the core elements as the system evolves, and to determine whether the data obtained from external sources that it manages has been updated. At user discretion, automatic and manual methods for incorporating the updated information must be available.

A hierarchical architecture fulfilling these requirements was designed and described in the white paper entitled “A Data Management Infrastructure for BioSpice.” The final version of the white paper was completed and delivered in April 2002. A copy is attached. We developed an initial implementation of a Java class library to support the data interchange mechanisms presented in the white paper, but were directed to focus resources on other areas before the implementation was complete.

The concepts embodied in the paper were utilized by the SEPDTF and EWG groups, though the subsequent implementation choices they made differed in detail from what had been presented in the white paper. In part, the choice of the NetBeans environment as the substrate for BioSpice constrained the architectural options for data management and led to a somewhat different implementation, although similar in philosophy to the original proposal.

4 BioSketchpad

Informal discussions between BBNT staff and other BioSpice performers at the second PI meeting led to the realization that non-commercial user-friendly tools for creating, editing, parameterizing, and executing models were lacking. Following further discussions between BBNT and the University of Pennsylvania group led by Harvey Rubin and Vijay Kumar led to the proposal that our two groups collaborate on development of a visual model creation tool that would allow a biologist to “draw” a model on a computer screen “canvas,” supply parameters and initial conditions, and execute the model in Penn’s CHARON simulation engine. The result of this collaboration was BioSketchpad (BSP). BSP drew on the insights of modelers at Penn and elsewhere (most notably John Tyson and Cliff Schaffer at Virginia Tech) to create the biologist-friendly front end for visually defining the model geometry and associating appropriate rate laws etc with the pathways that were defined. BSP would then recast the model in the form required by CHARON and run the simulation, presenting its graphical output on the screen.

The architecture of BSP was generalized after the initial proof of concept release to decrease the dependence of the code generation engine on CHARON as a first step towards making BSP useful with a wider range of simulators.

BSP development after the initial releases concentrated on increasing the interoperability of the tool with other BioSpice components. These included

- Development and partial implementation of an API to allow programmatic control of BSP
- Development of the ability of BSP to read and write models expressed in SBML, as well as in its own internal representation
- Numerous enhancements to rate law mechanics and the range of supported rate laws
- Support for rational stoichiometric coefficients (initially only integer coefficients were supported). This allowed BioSketchpad to be used to visualize the reaction geometries and stoichiometry of the Harvard group’s Flux Balance Analysis methodology.

- Development of a mechanism for providing visual layout information to an unannotated imported SBML model, using graphviz. Without this mechanism, all nodes and pathways in an imported SBML model would be overlaid.
- Initial development of the capability to add Jdesigner annotations to exported SBML, so that a model imported into that tool would be rendered in a reasonable fashion.
- Wrapped BSP using the XML-wrapper mechanism for incorporation into the BioSpice dashboard, in which it was used and demonstrated at PI meetings.

Experience working with SBML import and export for BSP played an important role in our contributions to the SBML language development and model interchangeability activities within BioSpice (see further discussion in Section 5 below).

Several BSP deliveries were made to DARPA and to the BioSpice participants.

5 Model Language Development

BBNT became involved in model language development efforts within BioSpice when the need for model interchange capabilities became apparent. This was driven in part by the development of BSP and by the insights gained in the DMI architecture task. A Model Definition Language working group (MDL) was stood up by DARPA and BBNT's Jonathan Webb joined the group and later was asked by Dr. Kumar to become co-chair with Oleg Sokolsky of Penn.

The MDL working group reviewed potential languages for use in BioSpice and concluded that adopting – as a basis for BioSpice – the Systems Biology Markup Language (SBML), an XML dialect developed for the same purpose and with its own pre-existing standards group, would be the most effective way to proceed. The MDL determined to participate actively in the SBML development and evolution process, thereby ensuring the maximum utility to BioSpice and the maximum interoperability with the remainder of the molecular biology modeling community.

Working with the MDL, BBNT supported development of the SBML Level II standard, which was approved by the SBML forum in 2004, and contributed several proposals for incorporation into SBML Level III. Our work with model interchange between BSP/CHARON and the Virginia Tech JigCell environment also made clear that the same model can be expressed in different valid but incompatible idioms in SBML. For example, representations of mathematical relationships (such as rate laws) can be represented implicitly in the formulation of the model itself, or referenced in a manner conceptually equivalent to a function call in a procedural programming language. A simulation engine designed to understand one of these formats will not be able to utilize a model created using the other. From these experiences, we proposed the need for “Style Guides” so that sets of tools that subscribe to the same style would be able to interchange models efficiently. Tools with differing style allegiances would at least have difficulty exchanging models; in some cases interchange would not be possible without significant loss of information. However, knowledge of which style was applicable to any tool would make it possible to know in advance how to express models that would be compatible, and whether it would support existing models based on their own styles.

An additional consideration in working with SBML is that the verbose XML format of the language and the sheer size of any interesting (non-trivial) models makes it extremely difficult to determine whether the SBML of a model is valid and self-consistent, and whether a model written using a specific style guide in fact adheres to that style. It is therefore necessary to provide tools for model validation. Such tools should be general, so that they can be adapted to function with newer versions of the language as it evolves, as well as to be able to validate against specific styles. One mechanism for implementing such a validator is to use XSLT – a generalized tool for analyzing XML – together with a set of rules that specify validity constraints. BBNT developed a set of such XSL rules for SBML Level II (a listing of the rules is attached to the electronic form of this report) and the XSLT implementation was carried out under subcontract to BBNT by the Institute of Software Integrated Systems at Vanderbilt University.

6 Conclusion

BBNT's tasking under this contract provided significant value to DARPA and to the BioSpice Program. Our contributions and their significance included

- Data Management – architectural requirements, identification of relevant data sources, data commonality across modalities, recognition of interoperability challenges and strategies for mitigation,
- Visual Modeling – provision of an effective prototype that was useful in its own right and subsequently helped motivate and direct effective development of model creation and editing tools by other BioSpice performers
- Model Representation – development and refinement of SBML beyond level I, need for consistency in usage (style guides), validation methodology and sample implementations
- Systems Engineering – our knowledge and understanding of systems engineering challenges were made available through our participation in SEPDTF, EWG, and MDL working groups and by our feedback and comments throughout the program.

Appendix A: Bio Comp Database Resource Summary Version 1.2

This document was developed by the DBWG (Data Base Working Group). It is based on an initial list provided by Adam Arkin, which was augmented with inputs provided by other PIs. Its purpose is to

(1) summarize databases of potential interest to the Bio-SPICE community, and
(2) provide reference information on those databases.

This is an evolving document which should grow to capture the evolving needs and thoughts of the Bio-SPICE community as a whole.

To suggest changes, please send email to biospice-dbwg@bbn.com.

To join the working group (and get on the email list), send email to majordomo@bbn.com and type the following in the body of the email "subscribe biospice-dbwg".

This document is organized as follows:

1. The listing of public databases likely to be of interest to the BioSpice community
2. An enumeration of the databases considered to be of high importance by members of the DBWG
3. An area for write-ins of other public databases that should be added to the master list.

Quick reference to Contents

Sequence Databases

Priority

DNA Databank of Japan (DDBJ) **(member International Nucleotide Sequence Database Collaboration) DDBJ**	
DB Structure	Flatfile (hierarchically arranged directories and files)
Primary Content	DNA and Protein sequence. Note: Although different groups administer DDBJ, EMBL, and NCBI, these databases share their contents daily and contain the same information.
Supporting Content	Identifying and functional information, features and their locations
Interfaces	Searchable by key word, accession number, homology searches, and taxonomy using various tools including FASTA, BLAST, using Sequence Retrieval System (SRS), and SSEARCH; also accessible by ftp, gopher, or email searches; shares new sequences with GenBank and EMBL
Access control	Unlimited read access; contribute via DDBJ using SAKURA or Sequin
Input format	Web-based or gui-based submissions, DDJB format
Output format	DDJB format; XML version available via ftp
URL	**http://www.ddbj.nig.ac.jp/**
License	No restrictions for publication or any other service. They request that the DDBJ be credited. No description of license or disclaimer on the web site. This description was received in email.

Priority

European Molecular Biology Laboratory (EMBL) / EBI Nucleotide Sequence Database (member International Nucleotide Sequence Database Collaboration) EMBL	
DB Structure	Flatfile
Primary Content	DNA sequence. Note: Although different groups administer DDBJ, EMBL, and NCBI, these databases share their contents daily and contain the same information.
Supporting Content	Literature references, functional information, locations of mRNAs and coding regions, positions of important mutations
Interfaces	Shares new sequences with GenBank and DDBJ; sequence format usually has to be changed for use with sequence analysis software
Access control	Unlimited read access; contribute via EMBL
Output format	EMBL format (similar to GenBank)
URL	**http://www.ebi.ac.uk/embl/index.html**

License	No restrictions for publication or redistribution. They require that the original source of the data be acknowledged by reference to the EMBL database (www.ebi.ac.uk/embl) and by citing the publication describing the EMBL database: Stoesser G. et al. 'The EMBL Nucleotide Sequence Database'. Nucleic Acids Res 30:21-26(2002).

Priority

GenBank (member International Nucleotide Sequence Database Collaboration) **NCBI**	
DB Structure	Flat File and ASN.1 versions available (< 60 GB)
Primary Content	DNA sequence. Although different groups administer DDBJ, EMBL, and NCBI, these databases share their contents daily and contain the same information.
Supporting Content	Translation products (protein sequence), literature references, functional information, locations of mRNAs and coding regions, positions of important mutations
Interfaces	Can be searched using various tools including BLAST and Entrez (Web-Entrez or Network-Entrez), shares new sequences with EMBL and DDBJ, incorporates data from Genome Sequence Data Base (GSDB); database can be downloaded via ftp
Access control	Unlimited read access; contribute via NCBI
Input format	Form-based via web interface (BankIt) or local gui (Sequin) for large submissions, email or diskette
Output format	GenBank format
URL	**http://www.ncbi.nlm.nih.gov/**
License	Copyright for the data belongs to the authors of the records. NCBI requests credit and that the disclaimer be reposted. http://www.ncbi.nlm.nih.gov/About/disclaimer.html

Priority

SwissProt (and trEMBL) **swissprot**	
Structure	Flatfile
Primary Content	Protein sequence (translated from the EMBL/EBI Nucleotide Sequence Database)
Supporting Content	Similar to EMBL but with more information about physical and biochemical properties
Interfaces	Can be searched using various tools including ExPaSy web server; database can be downloaded via ftp and is available on CD
Access control	Unlimited read access; contribute via EMBL
Input format	Similar to EMBL
Output format	SwissProt format (similar to EMBL); XML version planned
URL	**http://www.expasy.ch/sprot/sprot-top.html**

License	Providers (re-issuers) are required to pay a renewable flat rate license fee for the inclusion of all or part of the SWISS-PROT database into their service/product and making this available. They are also required to notify their users that the use of this 'parsed' copyright data from SWISS-PROT requires that they in turn acquire an end-user license from GeneBio.

Priority

Protein Information Resource (PIR) **PIR**	
Structure	Flatfile
Primary Content	Protein sequence
Supporting Content	Literature references, taxonomic and experimental information, and important features of the sequence
Interfaces	Database can be downloaded via ftp in several formats, including NBRF, XML, CODATA and FASTA
Access control	Unlimited read access; contribute via PIR
Input format	Form-based via web, via email or diskette
Output format	NBRF format (also called PIR Sequence format)
URL	**http://www-nbrf.georgetown.edu/pirwww/**
License	Redistribution is free but PIR requests being informed and that the external users keep up to date. (http://pir.georgetown.edu/pirwww/aboutpir/citepir.html#2)

Priority

The Institute for Genomic Research (TIGR) **TIGR**	
Structure	Flatfiles; multiple databases for different species
Primary Content	Gene sequence for particular species with genomic projects
Supporting Content	Project information, reference citations, taxonomic information
Interfaces	Web-based search tools available, including BLAST. Database can be downloaded via ftp.
Access control	Unlimited read access
Input format	FASTA
Output format	FASTA
URL	**http://www.tigr.org/**
License	Can not be reproduced, republished, redistributed, or transferred without the written permission of TIGR http://www.tigr.org/new/disclaimer.shtml

Structure Databases

Priority

CATH

CATH

Structure	Flatfile
Primary Content	Hierarchical domain classification of protein structures; proteins are classified by class, architecture, topology, and homologous superfamily.
Supporting Content	Various identifying annotations and references. Finer-grained classifications are also given.
Interfaces	Web-based. Search and structural analysis software available; database also accessible via ftp.
Access control	Unlimited read access
Output format	CATH format (FASTA-style)
URL	**http://www.biochem.ucl.ac.uk/bsm/cath_new/index.html**
License	No restrictions for academic use. Rights are owned by inpharmatica, who should be contacted for other uses.

Priority

FSSP

Structure	Flatfile
Primary Content	Structural alignments of pair-wise combinations of the proteins in PDB
Supporting Content	Identifying annotations, references, and alignment scores
Interfaces	Web-based, tools available for searching; automatically updated from PDB using Dali software; database also accessible via ftp.
Access control	Unlimited read access
Output format	FSSP format
URL	http://www.ebi.ac.uk/dali/fssp/fssp.html
License	

Priority

Protein Data Bank (PDB) **PDB**	
Structure	Flatfile
Primary Content	3D structure of biopolymers (atomic coordinates)
Supporting Content	Literature references, structure information, crystallographic structure factors, NMR experimental data
Interfaces	Web-based. Many tools for viewing and analyzing these data interface with this database. PDB files are accessible via ftp and are available on CD
Access control	Unlimited read access
Output format	Atomic Coordinate Entry Format (also called PDB format)
URL	**http://www.rcsb.org/pdb/**
License	The contents of PDB are in the public domain, but it is expected that the authors of an entry as well as the PDB be properly cited whenever their work is referred to. (http://www.rcsb.org/pdb/citing.html)

Priority

NCBI Molecular Modeling Database (MMDB) **MMDB**	
Structure	Flatfile
Primary content	3D structure of biopolymers, obtained from PDB
Supporting Content	Identifying and reference annotations
Interfaces	Web-based. Tools available for structural comparison, viewing, etc., including VAST and Entrez
Access control	Unlimited read access
Output format	ASN.1 format; XML also available
URL	**http://www.ncbi.nlm.nih.gov/Structure/MMDB/mmdb.shtml**
License	This database reuses data from PDB, so reuse must comply with PDB usage. There are no restrictions on the NCBI add-ons.

Priority

Structural Classification of Proteins (SCOP) **SCOP**	
Structure	Flatfile
Primary Content	Structural relationship among known protein structures (based on PDB) – hierarchically classifies protein structures from PDB
Supporting Content	Identifying and reference annotations
Interfaces	Web access; search tools available
Access control	Unlimited read access
Output format	SCOP format
URL	**http://scop.mrc-lmb.cam.ac.uk/scop/**

Priority

License	A license is required. The license is free only for academic users. (http://scop.mrc-lmb.cam.ac.uk/scoplic/licence.html)
DoubleTwist (May soon be unavailable because the creator has shut down)	
DB Structure	Proprietary - not specified in public documents
Primary Content	DNA Sequence; more then 35 integrated public and proprietary data sets and their exclusive Annotated Human and Mouse Gene Indices
Supporting Content	Extensive expert annotation, data visualization tools, data mining tools
Interfaces	Web interface
Access control	Commercial product – license fee for access
Input format	Web form interface
Output format	Web pages; custom reports (format unspecified in public pages)
URL	http://www.doubletwist.com/
License	Access to the database is by subscription. Re-publishing or re-serving is prohibited.

Priority

Biochemical and Biophysical Databases	
ENZYME	
DB Structure	Flatfile with 2-character identifiers for each logical record type
Primary Content	A repository of information relative to the nomenclature of enzymes
Supporting Content	Contains pointers to corresponding entries in SWISSPROT and identifies diseases associated with deficiencies
Interfaces	Web access through Expasy; data file can be downloaded freely
Access control	None
Output format	HTML table or raw data from database
URL	http://www.expasy.ch/enzyme
License	Copyright is owned by the Swiss Institute of Bioinformatics. Use of the ftp-capable database is unrestricted as long as content is unmodified.

Priority

BIND	
Structure	Not specified – accessed through Webgen custom software
Primary Content	Regulatory interaction networks and protein-protein interactions
Supporting Content	Connected components and alternative paths
Interfaces	Custom Java interface
Access control	None
Output format	Appears to be UI-only
URL	http://www.bind.ca/index.phtml?page=databases
License	The data are free with acknowledgement to all users. BIND source code is freely available under the GNU General Public License. (http://www.bind.ca/index.phtml?page=faq)

Biochemical Pathway Databases

Priority

PathDB	
DB Structure	RDBMS (SQL) with custom App Server on NCGR host. Java Application on user's computer.
Primary Content	A database designed to capture discrete metabolic steps. The **PathDB Database** is able to store rich information about pathways, enzymes, reactions, transport steps, and biochemical compounds. All the data are categorized by taxonomy. Focus on Arabidopsis and Yeast.
Supporting Content	
Interfaces	Java-based pathway viewer and discovery tool runs on user's machine and communicated with NCGR server. Appears to use RMI, but the web site isn't clear about this. RMI and JDBC are used internally
Access control	None
Input format	Not available. Input restricted to NCGR employees.
Output format	Primarily visualizations
URL	http://www.ncgr.org/pathdb/
License	A license is required with several restrictions. It may not be redeployed for commercial purposes. (http://www.ncgr.org/pathdb/licensing.html)

Priority

KEGG	
Structure	Flatfiles similar to PIR and GenBANK databases, with added software (DBGET components) that implement hierarchical relationships amongst elements
Primary Content	Contains several components: PATHWAY (pathways of interacting molecules or genes); GENES (sequence information linked to pathways); LIGAND (biologically active chemical compounds, also linked to pathways)
Supporting Content	Linked to DBGET database access environment, which in turn has links to most of the other major public databases
Interfaces	Web access through http://www.genome.ad.jp/kegg/. Combination of HTML and Java. Database is downloadable; free to academics, licensed to others.
Access control	None for Web access
Output format	HTML tables, Java visualizations. Flat data from downloaded copies.

URL	http://www.genome.ad.jp/kegg/
License	A license is required for any kind of reuse or redistribution. (http://www.genome.ad.jp/kegg/kegg5.html)

Priority

EMP and WIT	
Structure	Not specified. EMP appears to be RDBMS based, given a SQL example on one of the EMP web pages.
Primary Content	Enzymes and Metabolic Pathways database, EMP, covers all aspects of enzymology and metabolism and represents the whole factual content of original journal publications. The database format has about 300 subject fields. The WIT Project is based on a subset of EMP and attempts to produce metabolic reconstructions for sequenced (or partially sequenced) genomes. It currently provides a set of over 25 such reconstructions in varying states of completion.
Supporting Content	
Interfaces	Web access; EMP has a metabolic map editor that connects through a TCP port to am EMP server; details of the protocol not immediately obvious.
Access control	None
Output format	HTML pages and tables. Complex hyperlinked layout.
URL	http://www.empproject.com/ http://wit.mcs.anl.gov/WIT2/
License	http://www.anl.gov/disclaimer.html Documents authored by Argonne National Laboratory employees are the result of work under U.S. Government contract W-31-109-ENG-38 and are therefore subject to the following license: The Government is granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevokable worldwide license in these documents to reproduce, prepare derivative works, and perform publicly and display publicly by or on behalf of the Government.

Priority

AMAZE	
Structure	ODBMS (Java ObjectStore)
Primary Content	A database covering metabolic pathways from different organisms and tissues underpinned by information on enzyme function, by building on and extending existing resources , in particular BRENDA, KEGG/LIGAND, EMP.
Supporting Content	Links to key sequence databases (SWISS-PROT, EMBL-LIBRARY, GENBANK) & PDB (3D structure).

Priority

Interfaces	Java client using RMI to connect to the amaze server
Access control	None mentioned
Output format	Java-based textual and diagrammatic figures
URL	http://www.ebi.ac.uk/research/pfmp/texts/introduction.html
License	License required to use or re-publish data.
MetaCyc	
Structure	Object-oriented (a flat-file version is available for download)
Primary Content	MetaCyc is a metabolic-pathway database. The database describes pathways, reactions, and enzymes of a variety of organisms, with a microbial focus. MetaCyc contains the E. coli pathways of EcoCyc, plus additional pathways that have been gathered from a variety of literature and on-line sources.
Supporting Content	Citations to the source of each pathway.
Interfaces	Web-accessible using the graphical interface and query tools provided in Pathway Tools Software. Also available in flat-file format
Access control	None
Output format	Web output is specific to the Pathway Tools Software. It is also possible to connect directly to MetaCyc objects over the web.
URL	http://ecocyc.org/ecocyc/metacyc.html?
License	Normally re-serving or re-publishing is not allowed. However, since managed by SRI, they would like to work out a policy that allows this for Bio-SPICE.

Priority

Literature Databases	
PubMed	
Structure	Not specified, probably RDBMS.
Primary Content	PubMed, a service of the National Library of Medicine, provides access to over 11 million MEDLINE citations back to the mid-1960's and additional life science journals. PubMed includes links to many sites providing full text articles and other related resources
Supporting Content	Citations, publisher and journal links,
Interfaces	Web-based, through NCBI.
Access control	None
Output format	Several popular formats, including plain text, ASN.1, XML, and numerous others.
URL	http://www.ncbi.nlm.nih.gov/entrez/query.fcgi?db=PubMed
License	If the data are to be copied, a lease is required. NCBI recommends using the hooks already in place to pass queries through to PubMed rather than download the data. PubMed links to several other resources (http://www.ncbi.nlm.nih.gov/About/disclaimer.html)

Priority

Priority

Ontology Databases	
Gene Ontology Consortium **GO**	
Structure	RDBMS (mySQL)
Primary Content	Gene and Protein vocabularies that can be applied
Supporting Content	
Interfaces	Tools are available for web-based searches. Databases can be downloaded via ftp in either XML or mySQL formats. The schema are described, and Perl-based and Java-based modules are provided for accessing the data.
Access control	None
Input format	Searches are string-based.
Output format	Search outputs are text records.
URL	www.godatabase.org/dev/database
License	The GO browser is OpenSource, distributed under the GNU Public License. There are no restrictions on the ontologies, but the GO Consortium request that they and the member organizations be cited. http://www.geneontology.org/#cite
Microarray Gene Expression Data Group **MGED**	
Structure	MGED is a collection of web sites and downloadable documents.
Primary Content	Guidelines, object models, a data exchange format, ontologies, and recommendations for data normalization for microarray data and experiments.
Supporting Content	
Interfaces	Access is through browsable web sites. Documents containing guidelines, etc., are downloadable through the web.
Access control	None.
Input format	N/A
Output format	Various documents and illustrations.
URL	http://www.mged.org/
License	

Translation Tools

(Although not Databases per se, these tools are included as resources for data services)

Priority

GCG

GCG

Content	DNA or Protein sequence
Interfaces	Web-based, will download the sequence in file format to a remote PC
Access control	Unlimited use (but at least one version is commercial – see http://www.accelrys.com/)
Input/Output format	Choice of EMBL, FASTA, GenBank, IG, PIR, STADEN, and simple ASCII
URL	http://www.nick.med.usf.edu/GCGdoc/Doc.html
License	Genetics Computer Group grants permission to the University of South Florida, Tampa, Florida, USA, to allow this document to be available on the world wide web for use by Wisconsin Package users from GCG-licensed institutions that have Version 8 of the Package. If you fall under this definition, you may retrieve and print excerpts from this hypertext version. Retrieval and printing by all others is considered a violation of Genetic Computer Group's copyright.

Priority

READSEQ

READSEQ

Content	DNA or Protein sequence
Interfaces	Web-based form
Access control	Unlimited use
Input/Output format	Choice of IG/Stanford, GenBank, NBRF, EMBL, GCG, DNAStrider, Fitch, FASTA, Zuker, Olsen, Phylip 3.2, Phylip, Plain, PIR/CODATA, MSF, ASN.1, and PAUP/NEXUS
URL	http://bimas.dcrt.nih.gov/molbio/readseq/

License	The software is freely available to the public for use. The author, Don Gilbert, does not place any restrictions on its use or reproduction. Developers are encouraged to incorporate parts in their programs. The author would appreciate acknowledgement. http://iubio.bio.indiana.edu/soft/molbio/readseq/java/Readseq2-help.html#Source_code

Priority

Regulatory Databases

TRANSFAC

TRANSFAC

Content	Eukaryotic cis-acting regulatory DNA elements and trans-acting factors. Covers the whole range from yeast to human.
Structure	Master is RDBMS-based. Flat-file exports of several tables are made available for download on a regular basis. These have the customary field identifiers to identify record types.
Interfaces	A Web-based form interface is available to registered users.
Access Control	Accounts are required, these are free for academics and non-profits. License fees are required for commercial users.
Input/Output format	From online interfaces, not specified in the site documentation. Account required to access. Flatfile formats are defined in documentation.
Supporting content	TRANSPATH (Signal transduction database) CYTOMER (Database of organs, cell types, physiological systems and developmental stages)
URL	http://www.gene-regulation.de/
License	http://www.gene-regulation.com/pub/databases/transfac/doc/misc.html The TRANSFAC® database is free for users from non-profit organizations only. Users from commercial enterprises have to license the TRANSFAC® database and accompanying programs.

Priority

REGULONDB	
Content	Transcription regulation and operon organization for different organisms. It describes regulatory signals of transcription initiation, promoters, regulatory binding sites of specific regulators, ribosome binding sites and terminators, as well as information on genes clustered in operons. These specific annotations have been gathered from a constant search in the literature, as well as based on computational sequence predictions. The genomic coordinates of all these objects in each organism are clearly indicated. Every known object has a link to at least one MEDLINE reference.
Structure	Relational
Interfaces	Web-based
Access Control	Free to non-commercial organizations. License required for others.
Input/Output format	
Supporting content	
URL	http://www.cifn.unam.mx/Computational_Genomics/regulondb/
License	http://kinich.cifn.unam.mx:8850/db/regulondb_intro.frameset RegulonDB database is free for academic users only. Users from commercial companies are allowed to use the database during a reasonable testing period. For a regular user of the web version, a license fee should be paid. For on-site installation, please contact ecoli-reg@cifn.unam.mx for additional information.

Priority

MicroArray and Gene Expression Databases	
ExpressDB	
Content	ExpressDB is a relational database containing yeast and E. coli RNA expression data.
Structure	Relational
Interfaces	Web-based forms
Access Control	none
Input/Output format	
Supporting content	
URL	http://arep.med.harvard.edu/ExpressDB/
License	Data in this database were contributed by particular authors, and those authors should be properly cited. Harvard has no informal objections to redistribution or reserving. They recommend checking with our legal department first.
Stanford Microarray Database **SMD**	
Content	SMD stores raw and normalized data from microarray experiments, as well as their corresponding image files. In addition, SMD provides interfaces for data retrieval, analysis and visualization. Data is released to the public at the researcher's discretion or upon publication
Structure	Relational (Oracle)
Interfaces	Web-based forms
Access Control	Access is free, but users can register and thereby save sessions
Input/Output format	
Supporting content	
URL	http://genome-www5.stanford.edu/MicroArray/SMD
License	Copyright for the data is owned by the contributors, who have given permission to use the information. There are no restrictions or redistribution, but the original source should be acknowledged.

Priority

Priority

Gene Expression Omnibus **GEO**	
Content	GEO is a gene expression and hybridization array data repository, as well as an online resource for the retrieval of gene expression data from any organism or artificial source.
Structure	Relational database containing tab-delimited ASCII
Interfaces	Web-based forms, different options available including Entrez
Access Control	none
Input/Output format	
Supporting content	
URL	http://www.ncbi.nlm.nih.gov/geo/
License	Similar to GenBank. Permission may be required for some of the records, which may be copyrighted by the authors. (http://www.ncbi.nlm.nih.gov/geo/info/disclaimer.cgi)

Priority

NYU Microarray Database	
Content	Microarray data
Structure	Relational (PostgreSQL)
Interfaces	
Access Control	
Input/Output format	
Supporting content	
URL	

Priority

Priority

Organism-specific Databases	
FlyBase	
Structure	Flatfiles in a particular directory structure using links for connecting data in different parts of the directory structure.
Primary Content	FlyBase is a database of various genetic and molecular data for Drosophila, including genes, alleles, sequences, gene products, references, etc.
Supporting Content	See Primary Content above.
Interfaces	FlyBase is searchable via the web-based search tools. It is also accessible via ftp. The indexing and structure of the database is described to facilitate interfacing from other databases and engines.
Access control	None
Input Format	Searches are done via strings.
Output format	Records returned from searches are text records. Sequences are in FASTA format.
URL	http://flybase.bio.indiana.edu
License	Copyright is held by The Genetics Society of America. Commercial use of the data is prohibited without written permission from the FlyBase consortium. Some parts of FlyBase are also copyrighted separately and redistribution requires permission.
Stanford Saccharomyces Genome Database **SGD**	
Structure	Relational (Oracle)
Primary Content	Sequence analysis and tools for yeast.
Supporting Content	Display map position, function, mutant phenotypes, homology with human and worms, gene expression under various experimental conditions.
Interfaces	Blast and FastA search for yeast DNA and protein. Links to GenBank, PubMed, YPD (yeast protein database), Sacch3D (protein structural information), PIR, Swiss-Prot, CYGD (MIPS comprehensive yeast genome database).
Access control	None
Input Format	
Output format	
URL	http://genome-www.stanford.edu/Saccharomyces/

License	Re-publishing or re-serving the databases requires a license from Stanford University. There is no charge for academic sharing.

Priority

Yeast Proteome Database **YPD**	
Structure	Not specified
Primary Content	Detailed information about yeast genes, their function, regulation, mutant phenotypes of deletions and overproduction, localizations, genetic interactions with and other proteins, homologous genes in other organisms
Supporting Content	
Interfaces	Web-based search tool
Access control	Commercial product. Academic institutions qualify for free access.
Input Format	
Output format	
URL	http://www.proteome.com/database/YPD/YPDsearch-long.html
License	YPD is commercial, and large-scale downloading or reuse is prohibited without prior written consent. Complimentary access may be granted to not-for-profit institutions. (http://www.proteome.com/services/policies.html)

Priority

Berkeley Drosophila Genome Project **BDGP**	
Structure	Downloadable versions are flat files with FASTA or XML content.
Primary Content	Drosophila genome sequences.
Supporting Content	This is a rich database for Drosophila genomic information. The genome is annotated with information on chromosomal position, molecular function, protein domain, etc.
Interfaces	The database can be searched using web-based tools provided by the BDGP. The data sets and annotations can be downloaded, as can several software tools useful for accessing the data.
Access control	None.
Input Format	String-based searches can be done.
Output format	Text-based outputs result from searches. Downloaded files are in FASTA or XML format and are available via http or ftp.
URL	http://www.fruitfly.org/
License	There appear to be no restrictions on the use of the data in the BDGP database, however, the project request that they be cited appropriately. http://www.fruitfly.org/about/citations.html

Priority

European Drosophila Genome Project **EDGP**	
Structure	Downloadable versions are flat files with FASTA content.
Primary Content	Drosophila X chromosome sequence.
Supporting Content	The sequence is annotated with information on chromosomal position, molecular function, protein domain, etc.
Interfaces	The database can be searched using web-based tools provided by the EDGP. The data sets and annotations can be downloaded.
Access control	None.
Input Format	String-based searches can be done.
Output format	Text-based outputs result from searches. Downloaded files are in FASTA format and are available via ftp.
URL	http://edgp.ebi.ac.uk/
License	There appear to be no restrictions on the use of the data in the EDGP database. The project requests acknowledgement: http://edgp.ebi.ac.uk/EDGPcitat.html.

Priority

Mouse Genome Database **MGD**	
DB Structure	MGD appears to be a diverse collection of files, perhaps of different formats. It also contains links to external databases that contain mouse data.
Primary Content	MGD contains information on mouse genetic markers, molecular segments, phenotypes, comparative mapping data, experimental mapping data, and graphical displays for genetic, physical, and cytogenetic maps.
Supporting Content	MGD appears to be a database rich in annotations, references, etc.
Interfaces	The database can be searched via a set of web-based tools provided on the web site.
Access control	None.
Input format	Various string-based searches can be made, e.g., using symbols, names, etc.
Output format	Searches result in textual output.
URL	http://www.informatics.jax.org/mgihome/MGD/aboutMGD.shtml
License	There appear to be no restrictions on the use of this data. However, they request acknowledgement. http://www.informatics.jax.org/mgihome/other/copyright.shtml http://www.informatics.jax.org/mgihome/other/citation.shtml

Priority

Rat Genome Database **RGD**	
DB Structure	The downloaded version is a collection of flat files in a particular directory structure.
Primary Content	Rat genetic and genomic data. This project collects these data from diverse sources and normalizes them to a single database.
Supporting Content	References, taxonomic information, map information, etc.
Interfaces	Searches are via web-based forms. Searches appear to be cgi-based, and the web site describes a method to generate these searches automatically within a URL.
Access control	None
Input format	Searches are via web-based forms.
Output format	Search results are returned as HTML.
URL	http://rgd.mcw.edu/
License	The data in RGD may be downloaded freely for non-commercial purposes. The project requests proper acknowledgement and citation: http://rgd.mcw.edu/disclaimer.shtml, http://rgd.mcw.edu/cite.shtml.

Priority

The Gene Expression Database **GDX**	
DB Structure	Unclear. The database does not appear to be downloadable wholesale in any format.
Primary Content	Gene expression information from the laboratory mouse. GXD stores and integrates different types of expression data gathered from the literature.
Supporting Content	Literature references for mouse gene expression, a mouse anatomical dictionary and cross references, e.g., to tissue and cell line.
Interfaces	Searches are done through web-based forms.
Access control	None.
Input format	Various string-based searches can be made, e.g., using symbols, names, etc.
Output format	Searches result in textual output.
URL	http://www.informatics.jax.org/mgihome/GXD/aboutGXD.shtml
License	There appear to be no restrictions on the use of this data. However, they request acknowledgement. http://www.informatics.jax.org/mgihome/other/copyright.shtml http://www.informatics.jax.org/mgihome/other/citation.shtml

Priority

DictyBase

DictyBase

DB Structure	Unclear. DictyBase appears to be a collection of diverse content.
Primary Content	A complete resource about Dictyostelium discoideum and related organisms. Content includes genetics and genomics, cell biology, literature, labs, lab techniques, conferences, etc.
Supporting Content	
Interfaces	Searches are via web-based forms. The database does not appear to be downloadable or searchable through independent tools.
Access control	None.
Input format	Searches a via string inputs to web-based forms.
Output format	Search results are returned in HTML.
URL	http://dictybase.org
License	

Priority

Curagen PathCalling Yeast Interaction Database

Structure	Not specified
Primary Content	Information on interactions between proteins in relation to the yeast genome
Supporting Content	
Interfaces	Web-based form
Access control	Appears to be unrestricted, although a commercial product
Output format	HTML pages
URL	http://portal.curagen.com/extpc/com.curagen.portal.servlet.PortalYeastList
License	Likely restricted, since it’s commercial.

Priority

SCPD Yeast Promoter Database

Content	Promoter regions of approx 6000 genes and ORFs, annotations of known and putative regulatory sites, information on transcription factors
Structure	Appears to be relational
Interfaces	Web-based
Access Control	Appears to be unrestricted
Input/Output format	

Supporting content	Numerous analytical tools
URL	http://cgsigma.cshl.org/jian/index.html
License	Permission and license from Cold Spring Harbor Laboratory are required.

Priority

EcoCyc	
Structure	Object-oriented (a flat-file version is available for download)
Primary Content	EcoCyc is a bioinformatics database that describes the genome and the biochemical machinery of E. coli.
Supporting Content	Citations to the source of each pathway.
Interfaces	Web-accessible using the graphical interface and query tools provided in Pathway Tools Software. Also available in flat-file format.
Access control	None
Output format	Web output is specific to the Pathway Tools Software. It is also possible to connect directly to EcoCyc objects over the web.
URL	http://ecocyc.org/ecocyc/ecocyc.html
License	Normally re-serving or re-publishing is not allowed. However, since managed by SRI, they would like to work out a policy that allows this for Bio-SPICE.

Priority

E Coli DNA Binding Site Database	
Content	Uses known binding sites for DNA-binding proteins to identify other binding sites.
Structure	Flatfile
Interfaces	Web page listings and summaries
Access Control	none
Input/Output format	
Supporting content	
URL	http://arep.med.harvard.edu/ecoli_matrices/
License	Similar to ExpressDB. Data in this database were contributed by particular authors and can be considered published. The authors should be properly cited. Harvard has no informal objections to redistribution or reserving. They recommend checking with our legal department first.

Priority

Yeast Expression Connection	
Content	Microarray data
Structure	Various - Multiple database search tool
Interfaces	Web interface that searches simultaneously the results of several microarray studies for gene expression data for a given gene or ORF
Access Control	None
Input/Output format	
Supporting content	
URL	http://genome-www4.stanford.edu/cgi-bin/SGD/expression/expressionConnection.pl
License	License from Stanford University is required for re-publishing or re-serving the database or software.

Priority

Yeast mRNA Apparent Half-Life and Transcriptional Frequency	
Content	Expression level, mRNA half-life, and transcription frequency for approximately 6000 genes
Structure	Not specified
Interfaces	Web-based forms
Access Control	
Input/Output format	
Supporting content	
URL	http://web.wi.mit.edu/young/expression/halflife.html
License	

Priority

The Schizosaccharomyces pombe Genome Sequencing Project **PomBase**	
Content	Gene sequence data.
Structure	PomBase appears to a collection of databases, each of which is a collection of flat files with particular structure and format.
Interfaces	Searches are via web-based forms. The database may also be downloaded via ftp.
Access Control	None.
Input/Output format	Searches are via web-based forms, to which the response is returned in HTML format. The DNA sequence databases may be downloaded in FASTA or EMBL format.
Supporting content	In addition to the Sequence database, the following are available: Cosmid Assembly Data, a Protein database, Gene Ontology assignments, orthologs, and contig maps.
URL	http://www.sanger.ac.uk/Projects/S_pombe
License	The data in PomBase are freely available, however, the project members request proper acknowledgement and citation: http://www.sanger.ac.uk/Projects/use-policy.shtml.

Priority

The Arabidopsis Information Resource **TAIR**	
Content	Information pertinent to Arabidopsis: genes, gene markers, clones, sequences, maps, community, and literature.
Structure	Flat files with a particular directory structure and format.
Interfaces	TAIR may be searched via web-based forms. It may also be downloaded via ftp. The TAIR web-site also has links to external resources and provides a set of tools to interface with the database.
Access Control	None.
Input/Output format	Input to searches is via string inputs to web-based forms. Output is returned to the browser as HTML.
Supporting content	
URL	www.arabidopsis.org
License	Most information may be downloaded and reproduced. However, copyright may be owned by some contributors of data, and proper citations should be made: http://www.arabidopsis.org/disclaimer.html

Priority

WormBase **WormBase**	
Content	Essentially complete genomic sequence of the nematode.
Structure	AceDB format. The database can also be downloaded as flat files with particular directory structure and format.
Interfaces	Searches are possible via web-based. The databases are accessible via ftp; tools for accessing the database are also available via ftp.
Access Control	None.
Input/Output format	Input is via web-based forms. Response is in HTML.
Supporting content	Gene mapping, phenotypic information, references, RNA and protein data, etc.
URL	http://www.wormbase.org/
License	The data in the WormBase databases are freely available. They request proper acknowledgement and citation: ftp://ftp.sanger.ac.uk/pub/AAREADME.use-policy.txt.

Priority

Wormatlas	
Content	A database of behavior and structural anatomy of the nematode, Caenorhabditis elegans.
Structure	A collection of web sites and downloadable documents.
Interfaces	These documents and web content are accessible through the web.
Access Control	None.
Input/Output format	Content is accessible only through web browsing.
Supporting content	
URL	www.wormatlas.org
License	Content of the web site are copyrighted and should be cited appropriately. http://www.wormatlas.org/copyRight.htm

High Priority Databases

Contributor	Databases
Drew Endy	NCBI EMBL SwissProt SGD YPD PDB MMDB SCOP BIND PathCalling (if being actively maintained) PubMed SMD
John Tyson	SGD YPD Gene Expression (notably SMD)
Peter Karp	MetaCyc
Wayne Rindone	FlyBase SGD

OAA Wrapper Availability

Based on the discussions in the conference call on 11 February 2002, the only databases for which an OAA wrappers already exists are KEGG and ECOCYC (done at SRI, reported by Mark Johnson) and LBL's BioDB (reported by Adam Arkin; some reservations about the general availability of that wrapper were expressed).

Other Databases

Please use the space below to identify, characterize, and prioritize other public databases that are important to you and that you think should be interfaced to BioSpice. Feel free to attach additional pages.

Appendix B: A Data Management Infrastructure for Bio-SPICE

Version 1.02

1 April 2002

Prepared for:

Defense Advanced Research Projects Agency DARPA

Prepared by:

BBN
TECHNOLOGIES
A Verizon Company

10 Moulton Street
Cambridge, MA 02138

Table of contents

List of Figures

Introduction

The purpose of this paper is to present a Database Management Infrastructure (DMI) that can meet the challenging needs and goals of the Bio-SPICE program. This infrastructure recognizes and takes into account several important factors within the Bio-Computation arena:

- While Bio-SPICE will need to provide some commonality in database organization, each user site will need to structure its data repositories to suit local needs.
- In addition to internal data repositories, Bio-SPICE users need to access data from a number of external sources; the volume and types of data to be accommodated are increasing at a rapid rate.
- The community's understanding of and abilities to model biological systems are rapidly evolving, which will translate into a set of system requirements that will evolve over time.
- The community is working towards, but has not yet established, a stable and broad-based ontology. The Bio-SPICE infrastructure must be able to adapt as this develops.

As a result, the Data Management Infrastructure proposed here for Bio-SPICE is designed to:

- Allow each Bio-SPICE site to organize its internal data structures to suit individual needs
- Provide the flexibility to adapt to new or legacy data components, including project-specific data repositories and ontologies that evolve over time; and components with heterogeneous data structures, semantics, and APIs.
- Support access to heterogeneous external data resources from within Bio-SPICE
- Support data exchange among Bio-SPICE software components
- Support sharing and exchanging of both data structures and data across the community, at the discretion of the user
- Be compatible with other aspects of the Bio-SPICE architecture (e.g., OAA, Bio-SPICE Notebook).

This approach assumes that each Bio-SPICE system installation will be the primary locus for user access and control, but it will also permit elements of each system to be remote and to be shared with other Bio-SPICE systems. Using this model, each Bio-SPICE user controls his view of the Bio-SPICE universe, but may share models, data, etc., with other users and facilities.

Requirements and Rationale

The Bio-SPICE Program will make available a software suite that will include a data management component. It is anticipated that individual research groups will install the suite on one or more computers in their laboratories.

The principal requirements for the Bio-SPICE DMI include:

- **Heterogeneous Data Access.** The Bio-SPICE DMI must support access to many different types of data, from sources of differing structure, in a wide range of locations. Some will be under the control of the Bio-SPICE Program or its participants, while many others will not.
- **Flexible and Extensible.** Each user may work with different organisms, models, databases, etc. The system needs to be able to incorporate and adapt to these individual needs. Users must be able to adapt the system themselves.
- **Data Integrity.** The data managed by the DMI must be protected against corruption. Internal consistency must be guaranteed. In addition, data extracted from other sources must be traceable to their origins.
- **Defined Semantics.** Semantic relationships between data elements managed by the system must be maintained. They must also be available to other Bio-SPICE code for computational purposes and in human-readable form for browsing.
- **Performance.** The DMI must provide adequate performance for both local and network operations. When the nature of a data operation is likely to introduce significant delays, the DMI must inform the user and provide a mechanism to cancel or modify the operation to improve system responsiveness.
- **Access Controls and Data Sharing.** The DMI must provide mechanisms for identification and authentication of users, and selective sharing of contents with identified or general users (i.e. discretionary access control to its content).
- **Version control.** The DMI must have the ability to manage multiple versions of the objects in its stores.
- **Update Mechanisms.** The DMI must provide mechanisms to update the core elements as the system evolves, and to determine whether the data obtained from external sources that it manages has been updated. At user discretion, automatic and manual methods for incorporating the updated information must be available.

The proposed Bio-SPICE DMI is architecturally a hierarchy. Queries for data are directed initially to local resources. If not found, or if no local resource can provide the data, external or remote resources are utilized. Service directories provide the necessary information as to what resources are available, both in the active installation and in other locations.

The proposed organization was selected for a number of reasons, including the following:

- This architecture provides the flexibility required by the program to accommodate local customization of the data infrastructure, diversity in data sources, and ongoing evolution of system understanding, modeling, ontology, etc.
- While the DMI is not based on a single Data Warehouse architecture, it does not preclude use of Data Warehouses combining selected external databases, where appropriate; they become equal participants in the Bio-SPICE data aggregation mechanisms provided by the DMI.
- DMI support for local caching of selected data from external resources provides for improved performance, while retaining the ability to return to the original sources for additional data.
- Separate local repositories for different types of data (1) facilitate efficient data management; (2) improve performance; (3) facilitate provision of fine-grained access control to the elements in an individual Bio-SPICE installation; and (4) support rapid incorporation of data structures provided by other laboratories or required by new software components.

Satisfaction of some requirements is dependent on implementation details, rather than on the underlying architecture. For example, data integrity and access control requirements can be addressed by the selection and configuration of the database engine with which the DMI will be built.

Implementation of this architectural model, including the ability of users to incorporate new repositories at will and customize the representation of each, does impose some additional complexity on the Bio-SPICE infrastructure software. This cost is, however, inevitable if the required flexibility is to be provided, and is manageable.

The Bio-SPICE DMI would be implemented in phased releases. Initial versions would concentrate on development of the internal management utilities to support demonstrable proofs of concept. Later releases would add the flexibility for end-users. We anticipate that elements of the DMI would be developed and implemented by several groups.

Overview of the Bio-SPICE Database Management Infrastructure

We envision that the overall Bio-SPICE system will be delivered with access engines, user interface capabilities, and organizational infrastructure that allow it to maintain and utilize an extensible collection of databases, repositories, modeling tools, and other functionalities. Key elements of the Bio-SPICE DMI (see Figure 1) to support this are:

- **DMI Infrastructure Components**. A database engine, software to implement and manage heterogeneous data access, and a coordination language used to express organization, structure, and access mechanisms for the integrated suite of Bio-SPICE databases and repositories. Since these are the infrastructure on which the DMI is built, they are not shown explicitly in Figure 1.

- **A Bio-SPICE Core Database**. The core database (which may actually be implemented as a collection of database instances) includes:
    - an extensible directory of data sources, repositories, and software components (such as simulation engines) that have been registered with and incorporated into a local Bio-SPICE installation (the ***Local Service Directory***). When a resource (internal or external) is added to those available to the installation, an entry is added to this directory.
    - an extensible internal database (the ***Relationships Database***) representing relationships among database elements (e.g. projects, sources of model parameters in external data, dependencies among particular experiments, literature citations relevant to a particular model, etc.). This is, in effect, the "glue" that ties together the separate, stand-alone databases and repositories.
    - one or more additional internal databases designed to manage sets of data elements that have been identified as ubiquitous and of general relevance to the Bio-SPICE community. Examples include an extensible internal database to document and organize Projects (the ***Project Database***), common literature references, URLs, or metabolic parameters. The structure of these databases will have been agreed to by the Bio-SPICE community, but will also allow local schema extensions. The base schemas of these databases will be instantiated when a Bio-SPICE system is installed.

.

- **Internal databases and repositories**. These are data repositories that are selected and maintained locally within a Bio-SPICE installation. They may include cached replicates of all or part of external databases, and/or project- and user-defined data stores (including model definitions, experimental output, and annotations). Internal repositories derived from external databases may utilize the same structure as the parent database, or a simplified subset; the Bio-SPICE software will permit users to define these adaptations. The local repositories serve several purposes, including:

    - Local replication of critical data to improve performance and remove dependence on networked resources
    - Ability to curate and supplement data obtained from external databases to support specific requirements or correct inaccuracies

- **External databases**. These are data resources maintained separately from the Bio-SPICE system, but for which Bio-SPICE provides access mechanisms. Examples include GenBank, SwissProt, MetaCyc, SMD, etc.

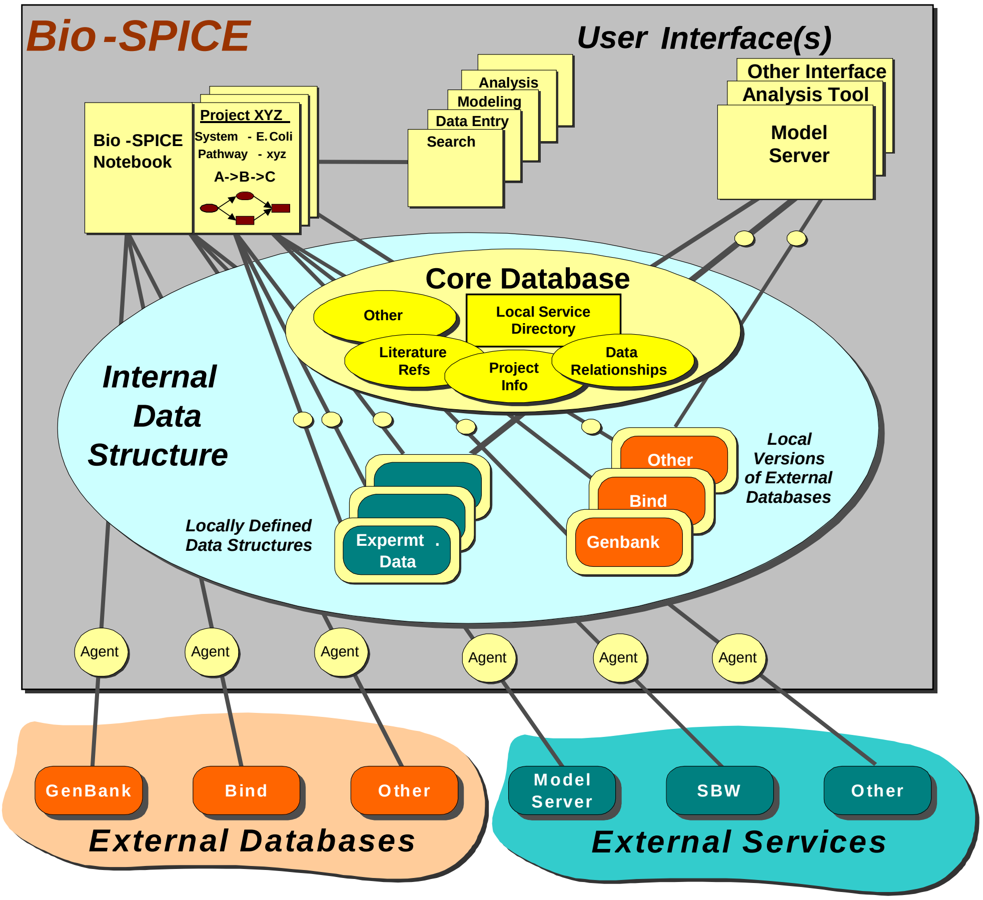

Figure 1. Overview of the Bio-SPICE Database Management Infrastructure and its context within the Bio-SPICE system. Note that the DMI provides services that can be utilized programmatically by other BioSPICE applications, as well as by notebooks and other specialized user interfaces.

Internal Databases

The Internal Data Resources are maintained within a local Bio-SPICE system (although in later releases the capability will exist to distribute them across machines). They consist of a structured Core Database and separate repositories to hold collections of Bio-SPICE elements. When Bio-SPICE is installed, it instantiates a Core Database structure and, optionally, a standard set of data repositories, which can subsequently be expanded to meet the needs of an individual Bio-SPICE site.

Core Database

The core database provides (1) a common structure to store data elements that are broadly used by the Bio-SPICE community (e.g., literature references and project information); (2) a mechanism to manage the content and organization of the accessible applications and data repositories (i.e., the Local Service Directory); and (3) a mechanism to store user-defined relationships among Bio-SPICE data elements (the Relationships Database). The local service directory tracks internal components as well as external elements that can be accessed through locally provided Bio-SPICE agents (such as OAA agents). As elements are added to Bio-SPICE, such as software modules, custom databases, experimental or simulation data, information is added to the core database to allow them to be managed and to support navigation among them.

The Core Database will provide user identification, authentication, and authorization capabilities. For installations that intend to share data and to impose access controls, it will include the role information and suitable tables required to identify individual rights and privileges for users, both local and remote.

As the Bio-SPICE community develops domain ontologies, requirements for common data elements, etc., the core database elements will evolve. They will therefore require a formal versioning system. It is likely that each extensible Bio-SPICE repository should have corresponding data elements in the core database to hold relevant information such as version, status, and access permissions.

The following is a list of candidate resources that will constitute the elements of the DMI Core Database.

Relationships Database

This database characterizes the relationships among Bio-SPICE elements. Its purpose is to support bi-directional navigation between related information residing in different Bio-SPICE databases and repositories. It therefore forms the glue that relates the separate heterogeneous databases and repositories. For example, it might be used to link BIND and SMD records relevant to a particular experiment within a project with literature references, other related experiments (including some managed by Bio-SPICE installations at other labs, subject to appropriate access controls), experimental and simulation data, descriptive or executable models used, and parameters. Likewise, it might link a particular literature reference to all of the experiments that reference it, allowing access to those experiments from the literature reference record. An example of the way related information might be represented is shown in Figure 2.

The relationship data supports aggregation of related information and access from a single point. The nature of the relationships between different Bio-SPICE data elements could in the future be defined in an overall Bio-SPICE ontology. If implemented in the Relationships Database, the ontology could in turn permit the relations of a Bio-SPICE project to be computable and might be used to support automated reasoning about experiments.

Similarly, relationship information may be supplied to indicate dependencies that a project or individual user identifies between project activities and data elements. The Bio-SPICE system may be able to utilize such relationship information to support configuration management and to assist users in maintaining consistent and current project data.

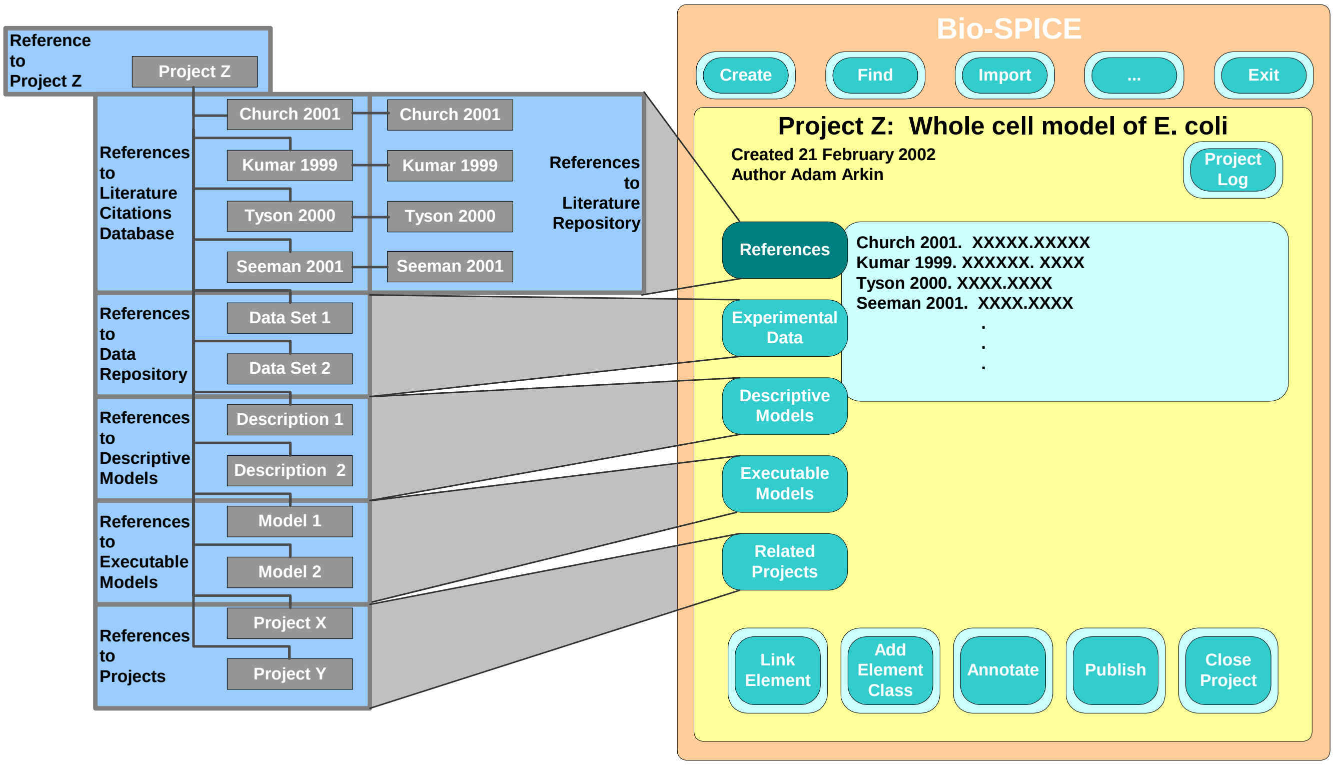

Figure 2. Example relationships than might be defined in the Relationships Database for a particular Bio-SPICE project. ***To the right is a hypothetical user interface representation of the Relationships illustrated at the left.***

Literature Reference Database

This database catalogues literature references of interest to the user. It contains the information necessary to identify the reference, such as citation data, abstract, PubMed reference data, and reference to local or public full-text versions. If a local copy is stored in the Literature Repository (see Section 4.2.3), the Relationships Database will store a link between it and the citation (stored here).

Project Information Database

This database holds user-defined information deemed necessary to document and control their projects. It will enable users to manage project-related information, such as project descriptions and notes, data and literature resources, experiment descriptions, and/or models associated with the project. The Project Information Database may be used to support desired Bio-SPICE applications, such as a Bio-SPICE lab Notebook. Note that most of the actual data associated with a project may be stored in other repositories, and will be available to the user through the Relationships Database.

Local User-specified or Extensible Databases

These are data repositories (databases or collections of data) with format and content defined by each Bio-SPICE site. They can be expanded and customized to meet individual interests. As repositories are added, they are made known to the Bio-SPICE Local Service Directory. When defining a new resource, the Bio-SPICE coordination language may be used to identify the organization, structure, and access interface for a repository. In initial releases, this process will be manual. Automated tools will be developed as resources permit to insulate users from this task.

The following subsections describe some likely extensible or user-specified data resources that are likely to become part of Bio-SPICE. These are all derived from discussions within the Bio-SPICE community. Note that these components will not all evolve at the same pace. Some may be defined and implemented on the basis of early consensus and may quickly become stable. Others will take longer to evolve, or may split into separate variants to support the needs of different subgroups within the community. These different evolution paths are easily accommodated in the DMI architecture.

Model Definitions Database

This optional database defines and tracks Bio-SPICE models that may be delivered with Bio-SPICE. It is intended to be expanded to include models added by the user. The models themselves are stored in a Bio-SPICE internal repository. The information held by this database uniquely identifies the model: version, textual description, references to the software used to either display the descriptive model or run the executable model, input definitions, references to tools and interfacing models.

Software Repository

This is a collection of software modules delivered with Bio-SPICE and is extensible to incorporate software modules added by the user. It may contain software to execute simulations, as well as tools: pathway editors, text editors, model transformation tools (e.g., that transform between descriptive and executable models), model-building tools, comparators, visualization tools, etc. The software repository may contain both legacy software equipped with Bio-SPICE wrappers, and Bio-SPICE native components.

Literature Repository

The purpose of this repository is to store full text copies of articles and other papers on the local Bio-SPICE system. It is referenced by the Literature database within the Core Database.

Experimental/Simulation Data Repository

This is an optional collection of files containing various kinds of experimental data and data generated by simulation runs. Its structure and organization will be highly dependent on the nature of the user's experiments and simulations.

Local Instantiations of External Databases

This is a collection of databases with structure and interfaces derived from those of external databases but with content maintained locally by the user. These databases may duplicate the entire richness of their source databases if desired. Alternatively, they may be simplified representations that meet the needs of the local user without requiring the full complexity of the external database structure or content. The goal is to allow users to store pertinent information from external databases locally, while not requiring undue complexity that may lead to reduced system performance and greater maintenance and support requirements.

Access to data managed by this class of repository relies on the hierarchical nature of the DMI architecture. The Local Service Directory will initially direct the DMI to direct queries for the kind of data managed by a particular database to the local instance. If the query yields no result, the DMI could then redirect the query to the parent – that is, the full external instance (query reformulation may be required when the local instance is simplified). If results are returned by the parent, the user may be provided with an option to save the resulting data locally, as well as to use it in the context in which it was requested.

Internal Agents to External Databases

This is a collection of internal Bio-SPICE agents that mediate access to publicly maintained databases such as GenBank.

External Databases

These are databases with which Bio-SPICE may interface but which are not managed within a local Bio-SPICE system. Their interfaces, structure, and semantics are externally controlled. There are no intrinsic constraints on the external databases that may be accessed from within Bio-SPICE. The Bio-SPICE system supplies access to an external database by means of an internal mediating agent. Such an agent supports access mechanisms native to the external repository and employs APIs and protocols that have been adopted as standards for exchange among Bio-SPICE components. In some contexts, these agents are referred to as "wrappers" or "ambassadors".

Data Warehouses accessed as external databases function no differently from other external databases in the Bio-SPICE DMI architecture. They will be accessed through agents of exactly the same type as those used for individual external databases. Their descriptions in the coordination language will be appropriate to the merged schema and access mechanisms of the warehouse. Note that there is no architectural or functional prohibition against building Bio-SPICE internal data repositories in the form of data warehouses, though Bio-SPICE–specific software to support this is unlikely to be generally available in early releases.

Interfaces, Data Flows and Interoperability

Summary of Interoperability Requirements and DMI Functionality

The Bio-SPICE DMI is designed with the expectation that it be able to incorporate many disparate databases from different sources. The DMI should also be able to incorporate a variety of different applications and services, and promote interoperation among different applications and data resources. The range of potential applications is wide, but as examples, we mention Differential Equation and Stochastic solvers, visualization tools, model-experiment comparators, and project administrative functions. Some such applications will be supplied along with Bio-SPICE distributions, while some will be unique to particular users and supplied by those users. The population of applications and services available to a given Bio-SPICE installation may vary over time.

To provide these desired operating capabilities, we foresee the following fundamental DMI constituents:

- A set of common exchange mechanisms (protocols / APIs) that can be used by applications (including data resources) to communicate between one another,
- The ability to construct interface agents or wrappers that adapt an application to communicate its available outputs and required inputs using the common Bio-SPICE exchange mechanisms,

- A coordination language that allows service providers and their clients to negotiate the context for service relationships, and
- A mechanism for one application to discover and establish contact with other available services and applications.

These DMI functionalities will be discussed briefly below[1].

The Bio-SPICE Wrapper and Coordination Language

As a general rule, an agent will be supplied for each database or application that resides external to Bio-SPICE or that was developed outside of Bio-SPICE. Its purpose is to mediate exchange between the application it represents and other Bio-SPICE elements. In some cases, the agent may simply be a wrapper that translates one application's output into exchange formats commonly used within Bio-SPICE. Wrappers enable all external sources to be accessed by Bio-SPICE in a consistent manner.

More generally, agents may provide procedural functionality (e.g., authentication services, web-based dialogs) of arbitrary complexity. Wrappers and other agents may make use of a coordination language supplied with the Bio-SPICE DMI. The coordination language is a mechanism to specify metadata that describes data structure and semantic content, application-specific APIs, prioritizations, and other characteristics of the relationships formed between applications and services. We envision that the coordination language will become rich enough to allow external databases or new applications to provide self-descriptive information, so that other Bio-SPICE applications may interact with them without requiring detailed prior knowledge of their interface or design. This reflective capability will support an environment to which new kinds of components can be added.

Service Directory and Discovery

We envision that Bio-SPICE applications will locate services through a service directory (Figure 3). Upon startup, a service provider registers itself with the service directory. When a Bio-SPICE application requires a service, it requests a service handle from the Bio-SPICE service directory. Some service providers may reside as internal elements of a Bio-SPICE installation, while others are external and remote. As illustrated in Figure 3, the DMI service discovery mechanism makes this distinction transparent to other Bio-SPICE elements. Using the service handle, the DMI locates potential providers of the service and requests information (metadata) from the service providers about structure, semantic contents, API, and other defining characteristics of the service and the provider. Using this information, the requesting application formulates its service requests, and the appropriate providers respond to the requests as specified. The service request illustrated in Figure 3 is a data query, but the types of services provided by Bio-SPICE are limited only by what can be represented in the coordination language used by the Bio-SPICE agents.

[1] Note that many of the concepts described in this section are very similar to those of the OAA. This is not meant to suggest that the DMI is an alternative to OAA. Instead it presents a vision of the DMI in the context of a larger system that could be implemented in the OAA environment. This is discussed more fully in Section 6.4.

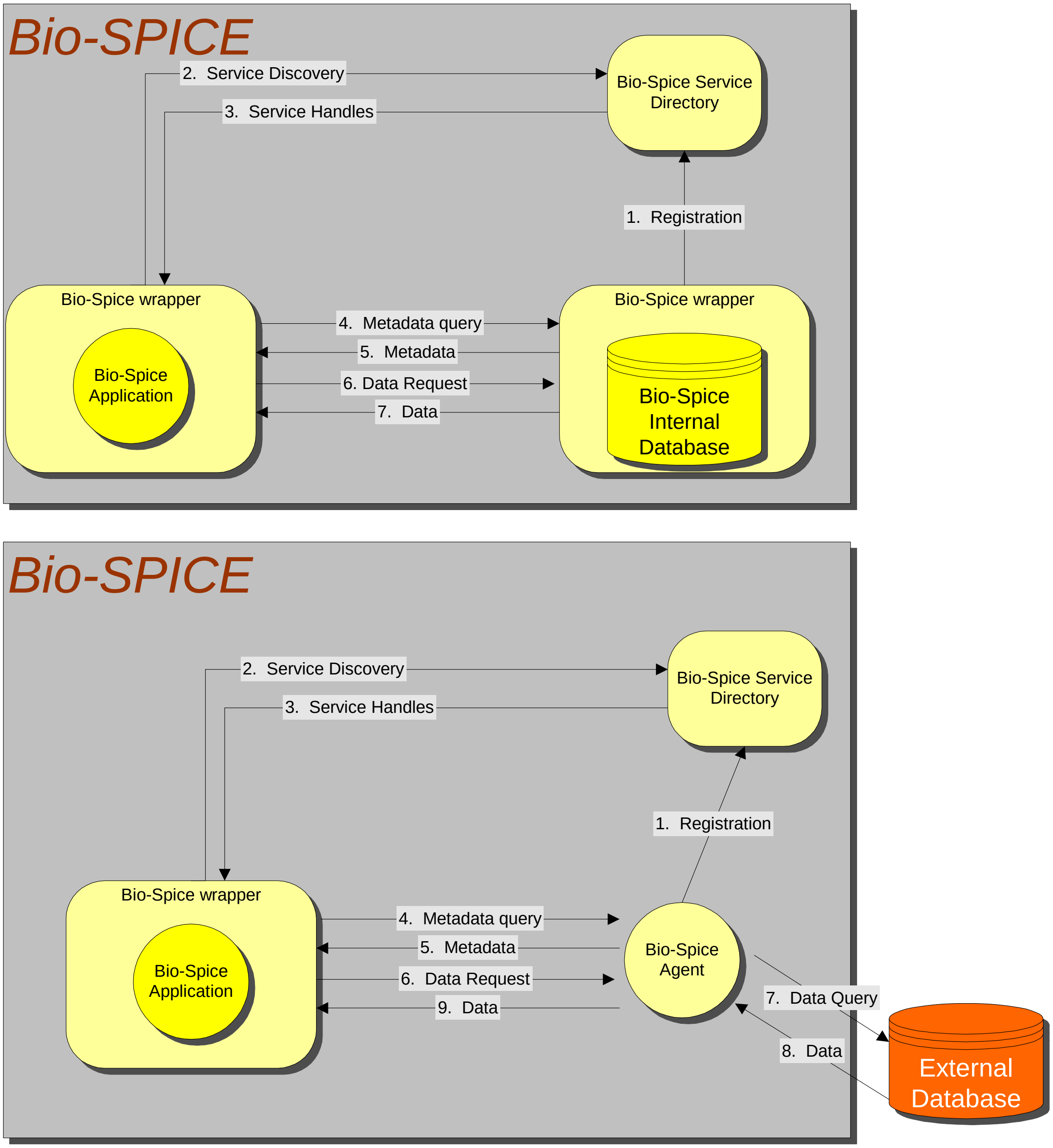

Figure 3. Data flow through the Bio-SPICE system when a Bio-SPICE application requests data from an internal or external database element. *(Although Bio-SPICE is illustrated as a single system here, individual components may be remote from each other).*

Implementation Possibilities

In this white paper we focus on architecture and functional capabilities that we propose for the Bio-Space DMI. We are not here proposing specific implementation choices. The agent methodologies that we have discussed could be implemented in a variety of ways, and indeed may come to be implemented in more than one software framework. A natural choice for database engines is to base most implementation upon SQL, but the design we are suggesting is not reliant upon such a choice. There exist a number of directory service frameworks that provide much of the capability discussed in Section 6.3, such as LDAP and JNDI. We believe that our proposal is also compatible with the design, philosophy, and capabilities of SRI’s Open Agent Architecture (OAA). The discovery and delegation properties of the OAA Facilitator seem conceptually compatible with our concept of a service directory. Likewise, the Interagent Communication Language (ICL) of OAA offers at least one possible instantiation of the kind of coordination language we envision. The DMI design presented here should be able to incorporate OAA developments, and allow the Bio-SPICE community to take advantage of attractive OAA agent capabilities, such as multimodal user interfaces and natural language facilities. The DMI should be able to support other implementation frameworks as well.

Relationship of the Bio-SPICE Notebook to the Bio-SPICE Database

The “Notebook” has been proposed by several Bio-SPICE PIs as a very desirable Bio-SPICE application. As we understand it, one of the primary functions of the notebook would be to allow the user to organize relationships among the Bio-SPICE elements, and then use these relationships to navigate among the Bio-SPICE services and data. Included in these relationships could be annotations regarding the provenance of data used in models, dependencies among experiments, and project work breakdowns. The notebook might also include personal databases that log and organize a user’s project activities. Data flows that meet these goals are shown in Figure 3.

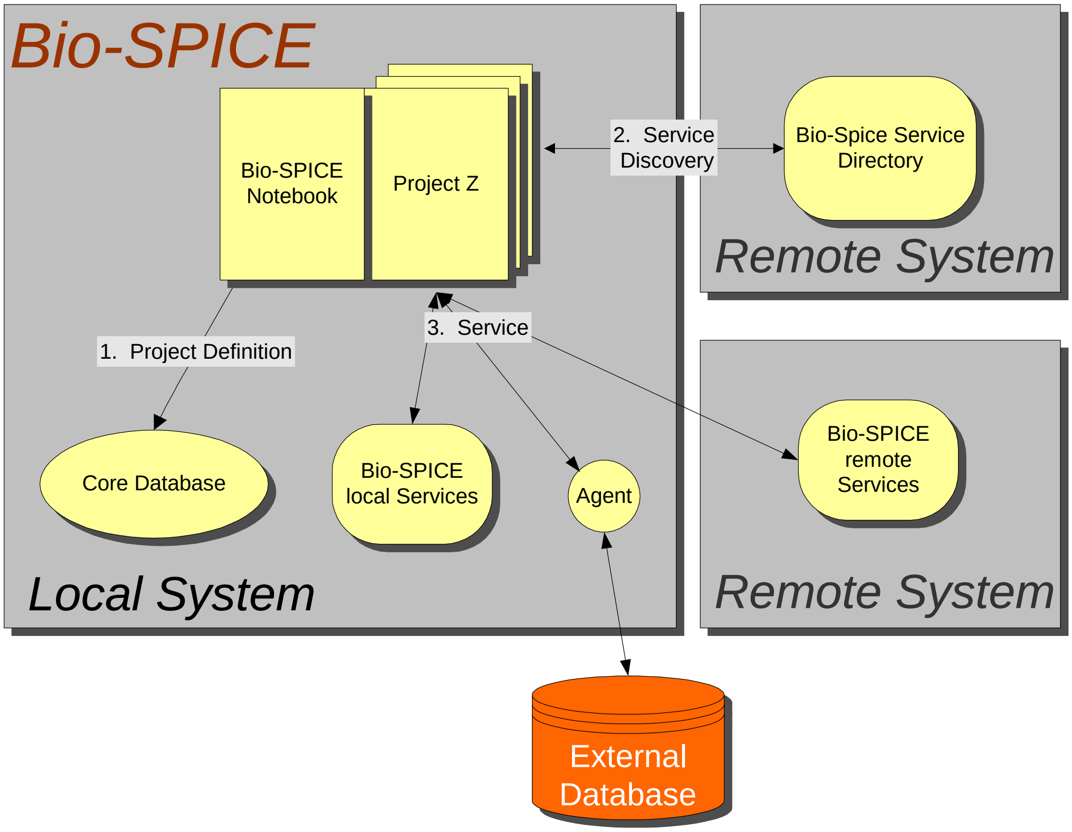

Figure 4. Data flows allowing the Bio-SPICE Notebook to aggregate the components of a Bio-SPICE Project. Some services may be remote from the Bio-SPICE Notebook and Core Database.

When the Bio-SPICE user creates a project in Bio-SPICE, a project entry is created in the core database. When the user wishes to add an element (e.g., a literature reference) to the project, the notebook adds a relationship in the project specification. In some instances, the user might want to use a Bio-SPICE service to locate elements to link to his project. For example, he might do a literature search for pertinent references. In this case, like other Bio-SPICE applications, the notebook would use the Bio-SPICE service directory to locate the appropriate service, and then invoke the service using the service handle provided by the service directory.

Appendix C: SBML Level II Validation Rules

```
   <?xml version="1.0" encoding="UTF-8" ?>
- <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:sbml="http://www.sbml.org/sbml/level2"
    xmlns:xhtml="http://www.w3.org/1999/xhtml"
    xmlns:math="http://www.w3.org/1998/Math/MathML"
    xmlns="http://www.w3.org/1999/xhtml" version="1.0">
 - <!--
   $Id: rules.xsl,v 1.7 2004/04/13 12:28:02 jwebb Exp $
   -->
   - <!--
      - validation tests for sbml level 2 version 1.  this is designed to
      - work in conjunction with schema validation.  checks performed against
      - the schema are not duplicated here.

   -->
  <xsl:output method="xml" omit-xml-declaration="no" version="1.0" encoding="UTF-8"
    doctype-public="-//W3c//DTD XHTML 1.0 Strict//EN" />
  <xsl:variable name="sbmlLevel" select="2" />
  <xsl:variable name="sbmlVersion" select="1" />
- <xsl:template match="/">
- <html>
- <head>
  <xsl:apply-templates select="//sbml:model" mode="makeTitle" />
   </head>
- <body>
  <xsl:apply-templates />
  <xsl:apply-templates select=".//sbml:unitDefinition" mode="matchUnitId" />
  <xsl:apply-templates select=".//sbml:reaction" mode="matchReactionParamId" />
  <xsl:apply-templates select=".//sbml:model" mode="matchGlobalId" />
  <xsl:apply-templates select=".//sbml:listOfSpecies" mode="speciesCompartment" />
  <xsl:apply-templates select=".//sbml:reaction" mode="reactionReferenceCheck" />
  <xsl:apply-templates select=".//sbml:compartment" mode="compartmentUnitRef" />
  <xsl:apply-templates select=".//sbml:listOfCompartments"
    mode="compartmentOutRef" />
  <xsl:apply-templates select=".//sbml:species" mode="speciesIC" />
  <xsl:apply-templates select=".//sbml:model" mode="unitDefs" />
  <xsl:apply-templates select=".//sbml:species" mode="speciesUnitRef" />
  <xsl:apply-templates select=".//sbml:parameter" mode="parameterUnitRef" />
  <xsl:apply-templates select=".//sbml:listOfRules" mode="ruleVariableRef" />
  <xsl:apply-templates select=".//sbml:listOfRules" mode="uniqueRuleRef" />
  <xsl:apply-templates select=".//sbml:event" mode="eventUnitRef" />
  <xsl:apply-templates select=".//sbml:eventAssignment" mode="eventVariableRef" />
  <xsl:apply-templates select=".//sbml:kineticLaw" mode="klawUnitRef" />
  <xsl:apply-templates select=".//sbml:functionDefinition" mode="functionLabelRef" />
  <xsl:apply-templates select=".//sbml:listOfRules" mode="ruleLabelRef" />
  <xsl:apply-templates select=".//sbml:kineticLaw" mode="klawLabelRef" />
  <xsl:apply-templates select=".//sbml:event" mode="delayLabelRef" />
  <xsl:apply-templates select=".//sbml:event" mode="triggerLabelRef" />
  <p>Scan complete</p>
   </body>
   </html>
   </xsl:template>
   - <!--
      - report on the current id or name of the model if present.  this
      - depends on the schema enforcing a single model element in the
      - stream.

   -->
- <xsl:template match="sbml:model" mode="makeTitle">
- <title>
   Validation check for
- <xsl:choose>
- <xsl:when test="boolean(@id)">
   SBML model id "
  <xsl:value-of select="@id" />
   "
   </xsl:when>
- <xsl:when test="boolean(@name)">
   SBML model name "
  <xsl:value-of select="@name" />
   "
   </xsl:when>
  <xsl:otherwise>unlabled SBML model</xsl:otherwise>
   </xsl:choose>
   </title>
   </xsl:template>
   - <!--
      - make sure the declared level and version of the model match the
      - expectations of the validation processing.

   -->
- <xsl:template match="sbml:sbml[@level!=$sbmlLevel or
    @version!=$sbmlVersion]">
- <p>
   SBML Level
  <xsl:value-of select="$sbmlLevel" />
   , version
  <xsl:value-of select="$sbmlVersion" />
   required for validation. The current model is level
  <xsl:value-of select="@level" />
   , version
```

```
<xsl:value-of select="@version" />
  .
  </p>
<xsl:apply-templates />
  </xsl:template>
  - <!--
     - unit definition identifiers need to be unique.  need a reference
     - to the various documents...

  -->
- <xsl:template match="sbml:unitDefinition[boolean(@id)]" mode="matchUnitId">
<xsl:variable name="currentId" select="@id" />
<xsl:apply-templates select="following-
  sibling::sbml:unitDefinition[@id=$currentId]" mode="duplicateUnitId" />
  </xsl:template>
- <xsl:template match="sbml:unitDefinition" mode="duplicateUnitId">
- <p>
  Duplicate
<em>unitDefinition</em>
  with id
<xsl:value-of select="@id" />
  </p>
  </xsl:template>
  - <!--
     - for any reaction, the parameters definied in the kinetic law for the
     - reaction need to have unique identifiers.

  -->
- <xsl:template
   match="sbml:kineticLaw/sbml:listOfParameters/sbml:parameter[boolean(@id)]
   " mode="matchReactionParamId">
<xsl:variable name="currentId" select="@id" />
<xsl:apply-templates select="following-sibling::sbml:parameter[@id=$currentId]"
   mode="matchReactionParamId" />
  </xsl:template>
- <xsl:template match="sbml:parameter" mode="matchReactionParamId">
- <p>
  Duplicate reaction
<em>parameter</em>
  with id
<xsl:value-of select="@id" />
  in
- <xsl:choose>
- <xsl:when test="boolean(../../../@id)">
  reaction id
<xsl:value-of select="../../../@id" />
  .
```

```
    </xsl:when>
 - <xsl:when test="boolean(../../../@name)">
    reaction name
   <xsl:value-of select="../../../@name" />
    .
    </xsl:when>
   <xsl:otherwise>unlabeled reaction.</xsl:otherwise>
    </xsl:choose>
    </p>
    </xsl:template>
      - <!--
         - for function definitions, compartments, species, reactions, rules,
         - global parameters, and events, their id's represent a single
      namespace
         - and must be unique.

    -->
      - <!--
         - this pattern doesn't seem to work...
         - sbml:model/*/*[self::functionDefinition or self::compartment or
         -   self::species or self::parameter or self::reaction or
         -   self::event][boolean(@id)]

    -->
 - <xsl:template match="sbml:functionDefinition[boolean(@id)]
     |sbml:compartment[boolean(@id)] |sbml:species[boolean(@id)]
     |sbml:model/sbml:listOfParameters/sbml:parameter[boolean(@id)]
     |sbml:reaction[boolean(@id)] |sbml:event[boolean(@id)]"
     mode="matchGlobalId">
   <xsl:variable name="currentId" select="@id" />
 - <xsl:if test="count(//sbml:functionDefinition[@id=$currentId]) +
     count(//sbml:compartment[@id=$currentId]) +
     count(//sbml:species[@id=$currentId]) +
     count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$currentId])
     + count(//sbml:reaction[@id=$currentId]) +
     count(//sbml:event[@id=$currentId]) > 1">
 - <p>
    Duplicate identifier for element
   <xsl:apply-templates select="." mode="elementLabelSelector" />
    with id
   <xsl:value-of select="@id" />
    .
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
         - the compartment reference for a species definition must refer to
         - a defined compartment.  checking for the existence of the id
         - attribute in the compartment and the compartment attribute in
```

```
        - the species is left to the schema definition.

      -->
- <xsl:template match="sbml:listOfSpecies/sbml:species"
     mode="speciesCompartment">
  <xsl:variable name="currentCompartment" select="@compartment" />
  <xsl:variable name="currentSpecies" select="@id" />
- <xsl:if test="not(boolean(//sbml:compartment[@id=$currentCompartment]))">
- <p>
    Missing compartment label
  <xsl:value-of select="$currentCompartment" />
    for species id
  <xsl:value-of select="$currentSpecies" />
    .
    </p>
    </xsl:if>
    </xsl:template>
    - <!--
        - a species reference in a reaction needs to be to a defined species.
        - the reference can be from reactants, products, or modifiers.

      -->
- <xsl:template match="sbml:listOfProducts/sbml:speciesReference"
     mode="reactionReferenceCheck">
- <xsl:apply-templates select="//sbml:listOfSpecies" mode="speciesExistenceCheck">
  <xsl:with-param name="speciesId" select="@species" />
  <xsl:with-param name="referenceId" select="../../@id" />
  <xsl:with-param name="referenceType">product</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
- <xsl:template match="sbml:listOfReactants/sbml:speciesReference"
     mode="reactionReferenceCheck">
- <xsl:apply-templates select="//sbml:listOfSpecies" mode="speciesExistenceCheck">
  <xsl:with-param name="speciesId" select="@species" />
  <xsl:with-param name="referenceId" select="../../@id" />
  <xsl:with-param name="referenceType">reactant</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
- <xsl:template match="sbml:modifierSpeciesReference"
     mode="reactionReferenceCheck">
- <xsl:apply-templates select="//sbml:listOfSpecies" mode="speciesExistenceCheck">
  <xsl:with-param name="speciesId" select="@species" />
  <xsl:with-param name="referenceId" select="../../@id" />
  <xsl:with-param name="referenceType">modifier</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
- <xsl:template match="sbml:listOfSpecies" mode="speciesExistenceCheck">
```

```
  <xsl:param name="speciesId" />
  <xsl:param name="referenceId" />
  <xsl:param name="referenceType" />
- <xsl:if test="count(sbml:species[@id=$speciesId]) = 0">
- <p>
    Missing species definition for species id
  <xsl:value-of select="$speciesId" />
    from
- <xsl:choose>
- <xsl:when test="boolean($referenceType)">
- <xsl:choose>
- <xsl:when test="boolean($referenceId)">
- <em>
  <xsl:value-of select="$referenceType" />
    </em>
    reference with reaction id
  <xsl:value-of select="$referenceId" />
    .
    </xsl:when>
- <xsl:otherwise>
    unlabeled
- <em>
  <xsl:value-of select="$referenceType" />
    </em>
    reference.
    </xsl:otherwise>
    </xsl:choose>
    </xsl:when>
- <xsl:otherwise>
- <xsl:choose>
- <xsl:when test="boolean($referenceId)">
    untyped reference with reaction id
- <em>
  <xsl:value-of select="$referenceId" />
    </em>
    .
    </xsl:when>
  <xsl:otherwise>untyped and unlabeled reference.</xsl:otherwise>
    </xsl:choose>
    </xsl:otherwise>
    </xsl:choose>
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
          - unit definitions in a compartment must reference a defined identifier
          - for a unit definition.
    -->
- <xsl:template match="sbml:compartment[boolean(@units)]"
     mode="compartmentUnitRef">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
     mode="unitDefExistence">
  <xsl:with-param name="unitId" select="@units" />
  <xsl:with-param name="refLabel" select="@id" />
  <xsl:with-param name="refType">compartment</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
      - <!--
        - this gets used in general for testing for the existence of a unit
        - definition.  based on the species existence check in the reaction
        - reference test.

    -->
- <xsl:template match="sbml:listOfUnitDefinitions" mode="unitDefExistence">
  <xsl:param name="unitId" />
- <xsl:param name="refLabel">
  <em><unspecified></em>
    </xsl:param>
- <xsl:param name="refType">
  <em><unspecified></em>
    </xsl:param>
- <xsl:if test="$unitId != "volume" and $unitId != "area" and $unitId != "length"
     and $unitId != "substance" and $unitId != "time" and $unitId != "ampere" and
     $unitId != "becquerel" and $unitId != "candela" and $unitId != "Celsius" and
     $unitId != "coulomb" and $unitId != "dimensionless" and $unitId != "farad"
     and $unitId != "gram" and $unitId != "gray" and $unitId != "henry" and
     $unitId != "hertz" and $unitId != "item" and $unitId != "joule" and $unitId !=
     "katal" and $unitId != "kelvin" and $unitId != "kilogram" and $unitId != "litre"
     and $unitId != "lumen" and $unitId != "lux" and $unitId != "metre" and $unitId
     != "mole" and $unitId != "newton" and $unitId != "ohm" and $unitId !=
     "pascal" and $unitId != "radian" and $unitId != "second" and $unitId !=
     "siemens" and $unitId != "sievert" and $unitId != "steradian" and $unitId !=
     "tesla" and $unitId != "volt" and $unitId != "watt" and $unitId != "weber" and
     count(sbml:unitDefinition[@id=$unitId]) = 0">
- <p>
    Missing unit definition for identifier
- <em>
  <xsl:value-of select="$unitId" />
    </em>
    from element type
- <em>
  <xsl:value-of select="$refType" />
    </em>
    with identifier
```

```
- <em>
  <xsl:value-of select="$refLabel" />
    </em>
    .
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
          - outside definitions on compartments must exist and must not be
          - self referential.

    -->
- <xsl:template
     match="sbml:listOfCompartments/sbml:compartment[boolean(@outside)]"
     mode="compartmentOutRef">
  <xsl:variable name="currentId" select="@id" />
  <xsl:variable name="currentOut" select="@outside" />
- <xsl:choose>
- <xsl:when test="$currentId=$currentOut">
- <p>
    Compartment
- <em>
  <xsl:value-of select="$currentId" />
    </em>
    references itself for the
  <em>outside</em>
    attribute.
    </p>
    </xsl:when>
- <xsl:when test="count(//sbml:compartment[@id=$currentOut]) = 0">
- <p>
    Compartment
- <em>
  <xsl:value-of select="$currentId" />
    </em>
    references non-existent compartment id
- <em>
  <xsl:value-of select="$currentOut" />
    </em>
    for the
  <em>outside</em>
    attribute.
    </p>
    </xsl:when>
    </xsl:choose>
    </xsl:template>
      - <!--
      - species elements can't define both an initialAmount and
      - initialConcentration attribute.

    -->
- <xsl:template match="sbml:species[boolean(@initialAmount) and
     boolean(@initialConcentration)]" mode="speciesIC">
- <p>
    Species id
- <em>
  <xsl:value-of select="@id" />
    </em>
    defines both
  <em>initialAmount</em>
    and
  <em>initialConcentration</em>
    attributes.
    </p>
    </xsl:template>
    - <!--
      - unit references in a species element must exist.

    -->
- <xsl:template match="sbml:species" mode="speciesUnitRef">
  <xsl:apply-templates select="." mode="speciesSubstanceUnitRef" />
  <xsl:apply-templates select="." mode="speciesSpatialUnitRef" />
    </xsl:template>
- <xsl:template match="sbml:species[boolean(@substanceUnits)]"
     mode="speciesSubstanceUnitRef">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
     mode="unitDefExistence">
  <xsl:with-param name="unitId" select="@substanceUnits" />
  <xsl:with-param name="refLabel" select="@id" />
  <xsl:with-param name="refType">species/substanceUnits</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
- <xsl:template match="sbml:species[boolean(@spatialSizeUnits)]"
     mode="speciesSpatialUnitRef">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
     mode="unitDefExistence">
  <xsl:with-param name="unitId" select="@spatialSizeUnits" />
  <xsl:with-param name="refLabel" select="@id" />
  <xsl:with-param name="refType">species/spatialSizeUnits</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
    - <!--
      - unit references in a parameter element must exist.

    -->
```

```
- <xsl:template match="sbml:parameter[boolean(@units)]"
    mode="parameterUnitRef">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
    mode="unitDefExistence">
  <xsl:with-param name="unitId" select="@units" />
  <xsl:with-param name="refLabel" select="@id" />
  <xsl:with-param name="refType">parameter</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
    - <!--
        - variable references in a rule must exist.  other rules need to
        - check for duplicate definitions of identifiers in relevant
        - elements.  i'm assuming rules are prohibited from updating
        - parameters defined in kineticLaw elements.

    -->
- <xsl:template match="sbml:assignmentRule" mode="ruleVariableRef">
- <xsl:call-template name="testRuleVariableExist">
  <xsl:with-param name="currentVariable" select="@variable" />
  <xsl:with-param name="ruleType">assignmentRule</xsl:with-param>
    </xsl:call-template>
    </xsl:template>
- <xsl:template match="sbml:rateRule" mode="ruleVariableRef">
- <xsl:call-template name="testRuleVariableExist">
  <xsl:with-param name="currentVariable" select="@variable" />
  <xsl:with-param name="ruleType">rateRule</xsl:with-param>
    </xsl:call-template>
    </xsl:template>
- <xsl:template name="testRuleVariableExist">
  <xsl:param name="currentVariable" />
  <xsl:param name="ruleType" />
- <xsl:if test="count(//sbml:compartment[@id=$currentVariable]) +
    count(//sbml:species[@id=$currentVariable]) +
    count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$currentVari
    able]) = 0">
- <p>
    Rule type
- <em>
  <xsl:value-of select="$ruleType" />
    </em>
    references non-existent element identifier
- <em>
  <xsl:value-of select="$currentVariable" />
    </em>
    as its
  <em>variable</em>
    attribute.
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
        - there are a collection of tests for unit related definitions.
        - some are simple enough to get wrapped into a single mode.

    -->
      - <!--
        - spatialDimension 0 compartments can't have a unit definition.

    -->
- <xsl:template match="sbml:compartment[@spatialDimensions=0 and
     boolean(@units) and @units!="dimensionless"]" mode="unitDefs">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
     mode="nonDimUnitCheck">
  <xsl:with-param name="unitId" select="@units" />
  <xsl:with-param name="compartmentId" select="@id" />
    </xsl:apply-templates>
    </xsl:template>
      - <!--
        - check to see if a particular unit definition represents a
        - simple, non-dimensional quantity.  note: this assumes some other
        - rule checks the name reference for the unit definition to make sure
        - it exists.  the generated report fragments are set to work with the
        - compartment unit checks.

    -->
- <xsl:template match="sbml:listOfUnitDefinitions" mode="nonDimUnitCheck">
  <xsl:param name="compartmentId" />
  <xsl:param name="unitId" />
- <xsl:choose>
- <xsl:when test="count(sbml:unitDefinition[@id=$unitId]) = 0">
- <p>
    Compartment id
- <em>
  <xsl:value-of select="$compartmentId" />
    </em>
    with spatial dimensions 0 references primitive unit specifier
- <em>
  <xsl:value-of select="$unitId" />
    </em>
    .
    </p>
    </xsl:when>
- <xsl:otherwise>
- <xsl:apply-templates select="sbml:unitDefinition[@id=$unitId]"
     mode="nonDimUnitCheck">
```

```
   <xsl:with-param name="compartmentId" select="$compartmentId" />
   <xsl:with-param name="unitId" select="$unitId" />
     </xsl:apply-templates>
     </xsl:otherwise>
     </xsl:choose>
     </xsl:template>
 - <xsl:template match="sbml:unitDefinition" mode="nonDimUnitCheck">
   <xsl:param name="compartmentId" />
   <xsl:param name="unitId" />
 - <xsl:choose>
 - <xsl:when test="count(.//sbml:unit) = 0 or count(.//sbml:unit) > 1">
 - <p>
     Compartment id
 - <em>
   <xsl:value-of select="$compartmentId" />
     </em>
     with spatial dimensions 0 references complex unit definition
 - <em>
   <xsl:value-of select="$unitId" />
     </em>
     .
     </p>
     </xsl:when>
 - <xsl:when test="count(.//sbml:unit[@kind="dimensionless"]) != 1">
 - <p>
     Compartment id
 - <em>
   <xsl:value-of select="$compartmentId" />
     </em>
     defines units with spatial dimensions set to 0.
     </p>
     </xsl:when>
     </xsl:choose>
     </xsl:template>
       - <!--
          - multiple assignment rules must not reference the same identifier as
       their
          - variable.  same for rate rules.  an assignment rule and a rate rule
          - may reference the same identifier.

     -->
 - <xsl:template match="sbml:assignmentRule" mode="uniqueRuleRef">
   <xsl:variable name="currentVariable" select="@variable" />
 - <xsl:if test="count(following-
      sibling::sbml:assignmentRule[@variable=$currentVariable]) > 0">
 - <p>
     Duplicate
```

```
  <em>variable</em>
    attribute reference in
  <em>assignmentRule</em>
    for identifier
- <em>
  <xsl:value-of select="@variable" />
    </em>
    .
    </p>
    </xsl:if>
    </xsl:template>
- <xsl:template match="sbml:rateRule" mode="uniqueRuleRef">
  <xsl:variable name="currentVariable" select="@variable" />
- <xsl:if test="count(following-sibling::sbml:rateRule[@variable=$currentVariable])
     > 0">
- <p>
    Duplicate
  <em>variable</em>
    attribute reference in
  <em>rateRule</em>
    for identifier
- <em>
  <xsl:value-of select="@variable" />
    </em>
    .
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
          - event unit declarations are restricted to the special symbols
          - time, second, or a unit definition identifier.

    -->
- <xsl:template match="sbml:event[boolean(@timeUnits)]" mode="eventUnitRef">
  <xsl:variable name="unit" select="@timeUnits" />
  <xsl:variable name="eventId" select="@id" />
- <xsl:if test="@timeUnits != "time" and @timeUnits != "second" and
     count(//sbml:unitDefinition[@id=$unit]) = 0">
- <p>
- <xsl:choose>
- <xsl:when test="boolean($eventId)">
    Event id
- <em>
  <xsl:value-of select="$eventId" />
    </em>
    </xsl:when>
  <xsl:otherwise>Unlabeled event</xsl:otherwise>
```

```
    </xsl:choose>
    references invalid time unit type
- <em>
  <xsl:value-of select="$unit" />
    </em>
    .
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
          - eventAssignment elements must reference defined variables.
        parameters
          - can't be defined in kineticLaw elements.

    -->
- <xsl:template match="sbml:eventAssignment" mode="eventVariableRef">
  <xsl:variable name="currentVar" select="@variable" />
  <xsl:variable name="currentEvent" select="../../@id" />
- <xsl:if test="count(//sbml:compartment[@id=$currentVar]) +
     count(//sbml:species[@id=$currentVar]) +
     count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$currentVar]
     ) = 0">
- <p>
- <xsl:choose>
- <xsl:when test="boolean($currentEvent)">
    Event id
- <em>
  <xsl:value-of select="$currentEvent" />
    </em>
    </xsl:when>
  <xsl:otherwise>Unlabeled event</xsl:otherwise>
    </xsl:choose>
    references undefined variable
- <em>
  <xsl:value-of select="$currentVar" />
    </em>
    .
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
          - unit references in a kineticLaw element must be to existing unit
          - definitions.

    -->
- <xsl:template match="sbml:kineticLaw" mode="klawUnitRef">
  <xsl:apply-templates select="." mode="klawTimeUnitsRef" />
  <xsl:apply-templates select="." mode="klawSubstanceUnitsRef" />
```

```
    </xsl:template>
- <xsl:template match="sbml:kineticLaw[boolean(@timeUnits)]"
     mode="klawTimeUnitsRef">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
     mode="unitDefExistence">
  <xsl:with-param name="unitId" select="@timeUnits" />
  <xsl:with-param name="refLabel" select="../@id" />
  <xsl:with-param name="refType">reaction/kineticLaw/timeUnits</xsl:with-param>
    </xsl:apply-templates>
    </xsl:template>
- <xsl:template match="sbml:kineticLaw[boolean(@substanceUnits)]"
     mode="klawSubstanceUnitsRef">
- <xsl:apply-templates select="//sbml:listOfUnitDefinitions"
     mode="unitDefExistence">
  <xsl:with-param name="unitId" select="@substanceUnits" />
  <xsl:with-param name="refLabel" select="../@id" />
  <xsl:with-param name="refType">reaction/kineticLaw/substanceUnits</xsl:with-
     param>
    </xsl:apply-templates>
    </xsl:template>
      - <!--
          - ci elements inside a function definition must refer to lexically
          - preceding function definitions or one of the bound variables.

    -->
- <xsl:template
     match="sbml:functionDefinition/math:math/math:lambda/*[position()=last()]
     //math:ci" mode="functionLabelRef">
  <xsl:variable name="var" select="text()" />
- <xsl:if test="count(ancestor::math:lambda/math:bvar/math:ci[text()=$var]) +
     count(ancestor::sbml:functionDefinition/preceding-
     sibling::sbml:functionDefinition[@id=$var]) = 0">
- <p>
    Missing or inappropriate symbol definition for reference
- <em>
  <xsl:value-of select="$var" />
    </em>
    in
  <em>functionDefinition</em>
    with id
- <em>
  <xsl:value-of select="ancestor::sbml:functionDefinition/@id" />
    </em>
    .
    </p>
    </xsl:if>
    </xsl:template>
    - <!--
      - identifiers in rules are restricted to refering to identifiers for
      - particular types of elements.  this depends on some other template
      - making checks for duplicate identifiers.  this is not checking for
      - reference cycles among rules just definition of the identifiers.

  -->
- <xsl:template match="sbml:listOfRules/sbml:algebraicRule/math:math//math:ci
    |sbml:listOfRules/sbml:assignmentRule/math:math//math:ci
    |sbml:listOfRules/sbml:rateRule/math:math//math:ci" mode="ruleLabelRef">
  <xsl:variable name="var" select="text()" />
- <xsl:if test="count(//sbml:functionDefinition[@id=$var]) +
    count(//sbml:compartment[@id=$var]) + count(//sbml:species[@id=$var]) +
    count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$var]) = 0">
- <p>
   Missing or inappropriate symbol definition for reference
- <em>
  <xsl:value-of select="$var" />
    </em>
   in
- <em>
  <xsl:apply-templates
    select="ancestor::sbml:algebraicRule|ancestor::sbml:assignmentRule
    |ancestor::sbml:rateRule" mode="elementLabelSelector" />
    </em>
   .
    </p>
    </xsl:if>
    </xsl:template>
    - <!--
      - identifiers in kineticLaw elements are restricted to local and
      - global parameters, function definitions, compartments, and
      - species included in the reaction definition.  like the other
      - reference testing rules, this depends on making uniqueness checks
      - made elsewhere.

  -->
- <xsl:template match="sbml:kineticLaw/math:math//math:ci"
    mode="klawLabelRef">
  <xsl:variable name="var" select="text()" />
- <xsl:if
    test="count(ancestor::sbml:reaction//sbml:speciesReference[@species=$var])
    +
    count(ancestor::sbml:reaction//sbml:modifierSpeciesReference[@species=$var
    ]) +
    count(ancestor::sbml:kineticLaw/sbml:listOfParameters/sbml:parameter[@id=
    $var]) + count(//sbml:functionDefinition[@id=$var]) +
    count(//sbml:compartment[@id=$var]) +
    count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$var]) = 0">
```

```
- <p>
    Missing or inappropriate symbol definition for reference
- <em>
  <xsl:value-of select="$var" />
    </em>
    in
  <em>kineticLaw</em>
    element for
- <xsl:choose>
- <xsl:when test="boolean(ancestor::sbml:reaction/@id)">
    reaction id
- <em>
  <xsl:value-of select="ancestor::sbml:reaction/@id" />
    </em>
    </xsl:when>
- <xsl:when test="boolean(ancestor::sbml:reaction/@name)">
    reaction name
- <em>
  <xsl:value-of select="ancestor::sbml:reaction/@name" />
    </em>
    </xsl:when>
  <xsl:otherwise>unlabeled reaction</xsl:otherwise>
    </xsl:choose>
    .
    </p>
    </xsl:if>
    </xsl:template>
      - <!--
          - identifiers in the delay element of an event have reference
        requirements
          - like rules; they can only reference global parameters allowed to
        vary.

    -->
- <xsl:template match="sbml:delay/math:math//math:ci" mode="delayLabelRef">
  <xsl:variable name="var" select="text()" />
- <xsl:if test="count(//sbml:functionDefinition[@id=$var]) +
     count(//sbml:compartment[@id=$var]) + count(//sbml:species[@id=$var]) +
     count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$var]) = 0">
- <p>
    Missing or inappropriate symbol definition for reference
- <em>
  <xsl:value-of select="$var" />
    </em>
    in
  <em>delay</em>
    element for
```

```
- <xsl:choose>
- <xsl:when test="boolean(ancestor::sbml:event/@id)">
    event id
- <em>
  <xsl:value-of select="ancestor::sbml:event/@id" />
    </em>
    </xsl:when>
- <xsl:when test="boolean(ancestor::sbml:event/@name)">
    event name
- <em>
  <xsl:value-of select="ancestor::sbml:event/@name" />
    </em>
    </xsl:when>
  <xsl:otherwise>unlabeled event</xsl:otherwise>
    </xsl:choose>
    .
    </p>
    </xsl:if>
    </xsl:template>
    - <!--
        - identifiers in the trigger element of an event have reference
      requirements
        - like rules; they can only reference global parameters allowed to
      vary.

    -->
- <xsl:template match="sbml:trigger/math:math//math:ci" mode="triggerLabelRef">
  <xsl:variable name="var" select="text()" />
- <xsl:if test="count(//sbml:functionDefinition[@id=$var]) +
     count(//sbml:compartment[@id=$var]) + count(//sbml:species[@id=$var]) +
     count(//sbml:model/sbml:listOfParameters/sbml:parameter[@id=$var]) = 0">
- <p>
    Missing or inappropriate symbol definition for reference
- <em>
  <xsl:value-of select="$var" />
    </em>
    in
  <em>trigger</em>
    element for
- <xsl:choose>
- <xsl:when test="boolean(ancestor::sbml:event/@id)">
    event id
- <em>
  <xsl:value-of select="ancestor::sbml:event/@id" />
    </em>
    </xsl:when>
- <xsl:when test="boolean(ancestor::sbml:event/@name)">
```

```
    event name
 - <em>
   <xsl:value-of select="ancestor::sbml:event/@name" />
     </em>
     </xsl:when>
   <xsl:otherwise>unlabeled event</xsl:otherwise>
     </xsl:choose>
     .
     </p>
     </xsl:if>
     </xsl:template>
       - <!--
         - these templates convert typical element labels into corresponding
         - text nodes.  there doesn't seem to be a way to do it using
         - combinations of the current context and the value-of operation.
         - value-of wants to return the _content_ of the node and not the
         - label of the node.

     -->
 - <xsl:template match="sbml:functionDefinition" mode="elementLabelSelector">
   <xsl:text>functionDefinition</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:compartment" mode="elementLabelSelector">
   <xsl:text>compartment</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:species" mode="elementLabelSelector">
   <xsl:text>species</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:parameter" mode="elementLabelSelector">
   <xsl:text>parameter</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:reaction" mode="elementLabelSelector">
   <xsl:text>reaction</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:event" mode="elementLabelSelector">
   <xsl:text>event</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:algebraicRule" mode="elementLabelSelector">
   <xsl:text>algebraicRule</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:assignmentRule" mode="elementLabelSelector">
   <xsl:text>assignmentRule</xsl:text>
     </xsl:template>
 - <xsl:template match="sbml:rateRule" mode="elementLabelSelector">
   <xsl:text>rateRule</xsl:text>
     </xsl:template>
       - <!--
```

```
      - a default rules, one for each rule mode, so unprocessed text and
      - attributes are not output to the report.  this needs to always be
      - at the bottom of the file.

  -->
<xsl:template match="@*|text()" />
<xsl:template match="@*|text()" mode="matchUnitId" />
<xsl:template match="@*|text()" mode="matchReactionParamId" />
<xsl:template match="@*|text()" mode="matchGlobalId" />
<xsl:template match="@*|text()" mode="speciesCompartment" />
<xsl:template match="@*|text()" mode="reactionReferenceCheck" />
<xsl:template match="@*|text()" mode="speciesExistenceCheck" />
<xsl:template match="@*|text()" mode="compartmentUnitRef" />
<xsl:template match="@*|text()" mode="unitDefExistence" />
<xsl:template match="@*|text()" mode="compartmentOutRef" />
<xsl:template match="@*|text()" mode="speciesIC" />
<xsl:template match="@*|text()" mode="unitDefs" />
<xsl:template match="@*|text()" mode="speciesUnitRef" />
<xsl:template match="@*|text()" mode="speciesSubstanceUnitRef" />
<xsl:template match="@*|text()" mode="speciesSpatialUnitRef" />
<xsl:template match="@*|text()" mode="parameterUnitRef" />
<xsl:template match="@*|text()" mode="ruleVariableRef" />
<xsl:template match="@*|text()" mode="nonDimUnitCheck" />
<xsl:template match="@*|text()" mode="uniqueRuleRef" />
<xsl:template match="@*|text()" mode="eventUnitRef" />
<xsl:template match="@*|text()" mode="eventVariableRef" />
<xsl:template match="@*|text()" mode="klawUnitRef" />
<xsl:template match="@*|text()" mode="klawTimeUnitsRef" />
<xsl:template match="@*|text()" mode="klawSubstanceUnitsRef" />
<xsl:template match="@*|text()" mode="functionLabelRef" />
<xsl:template match="@*|text()" mode="ruleLabelRef" />
<xsl:template match="@*|text()" mode="klawLabelRef" />
<xsl:template match="@*|text()" mode="delayLabelRef" />
<xsl:template match="@*|text()" mode="triggerLabelRef" />
  </xsl:stylesheet>
```