White Paper
BioXM Best Practices for Agile Solution Building

The Biomax BioXM™ Knowledge Management Environment is a truly unique generic platform for dynamic modeling, visualization and analysis of biological and biomedical networks. The BioXM system masters two critical problems: How do I adequately approach a complex scientific problem adequately in silico, and how do I deploy the results of my solution to a broad end-user community? Scientific end-users are typically not interested in the technical details of a software solution, but they want to benefit from its results. Accordingly, the “solution building” process has two tightly interconnected phases, that from a design perspective, are completely distinct. Phase one focuses on solving the problem, while phase two focuses solely on the deployment to end-users.

Phase I: Solution building and refining

The agile solution building process enabled by the BioXM system is depicted in figure 1. This three step iterative process allows building the solution from the perspective of the anticipated use and benefit. The process starts with a high level specification of the solution, answering the questions “What are types of data I am dealing with?”, “How do I want them to be described?” and “How are they related to one another?”.

Figure 1: Phase one agile solution building process

The BioXM system allows drag-and-drop, point and click building of the model of the scientific domain, shielding the solution architect from any underlying technical infrastructure. When developing the data model it is important to distinguish concepts that will become objects within the BioXM data model versus data that will be used to enrich the object with additional information.
Step one is mapping relationships between objects and should be based on valid scientific relationships with evidence for each relationship incorporated into the data model.

Step two is instantiating the domain model with real world content, e.g., from an Excel spreadsheet, mapping the rows and columns to the parts of the data model. When going from the data model to importing the data from a source it is necessary to assess the overall system by answering the following questions about the data source:

2. Are vocabularies/ontologies/structured terminology used in the database? What type?
3. Who are the users? What department? What functional area? Most common queries/reports?
4. What APIs are available to access the system (e.g. web services)
5. Size of the database?
6. How fast is the database growing? Does it change daily?
7. How is the data accessed by users now? Interface? GUI? SharePoint?
8. Type of database – Oracle, MySQL or flat file.
9. Where is the database located? US? Europe?
10. What is the network connection speed?

After the system has been identified and general characteristics determined, the location of the specific data needed for the data model needs to be determined and the following questions addressed:

1. Is the data in a database table or view, in a document, etc.?
2. If it is a table or view, is there a primary key?
3. Is there other data that should be extracted from the specific source? How does this data fit into the data model?
4. Does the data provide evidence for a relationship between two concepts?

Step three is the validation of the solution by usage through the scientific domain experts. Asking “Do I get the answers to the questions that are relevant to the problem I would like to solve?” after running through these three steps usually reveals that some concept might be missing in the model of my scientific domain (step1) or that not enough data is available to answer my questions (step2) — just like a “traditional” software programming project. However, key to the agile solution building process is that I can iterate through the three steps in a matter of hours or days, not months or years. The reason for this is, that all functional aspects of the system — such as reporting, the importing system, the query language and data export — is dynamically inferred from the data model. The data model of my scientific domain can be modified in real time, even while the system is used productively, which allows for a rapid response to changing requirements for the solution.

Phase II: Solution deployment

The second phase, deploying the solution into the hands of the scientific end-user community, is built on Phase I iterations after the building cycle has been iterated at least once, and any time thereafter. Practically, both phases are run in circular, iterative process.
Deployment to an end-user community is a completely different ball game compared to the building of an initial solution of a scientific problem. End-users are primarily consumers — they are not interested in the conceptualization of the solution, i.e., data models accessing information. The expectation is that this is all there, done and organized in a way that makes the life of the end-user easy — allowing to conduct routine operations fast and without need of training.

**Figure 2:** Phase two agile solution building process

The typical types of software interactions for end-users are find, report and comment. While all this sounds very simple, the devil is in the detail. The two phases of agile solution building with the BioXM system puts tremendous power into the hands of the solution-building administrator. In Phase I, all the detail and complexity is addressed. In phase two, a completely different approach is taken focusing on configuration and deployment of simple web-apps, which access the content managed by the solution to guide the end-user intelligently and render the right information in the right context, creating an intuitive workflow of interactions.

The overall benefit of the BioXM approach is the unique combination of “ease of build and maintain” with “ease of use” in one consistent and sustainable environment.

**Technical Background**

Technically the BioXM platform follows a Java-based client–server architecture, supporting MySQL or Oracle as the server-side relational database management system (RDBMS), with a HTML-based portal front-end for deployment to end-users. The BioXM Web-Portal Builder is a set of utilities, which allow for the presentation of BioXM content within any browser and for rapidly creating special purpose applications utilizing the advanced features of the BioXM backend. The core of the Portal Builder consists of a number of plugins for an enterprise Wiki system. Data can be imported
directly through the main Java-based BioXM client application, the web portal, from the command line or any other application accessing the BioXM application programming interfaces (APIs). In addition, externally maintained data sources can be integrated transparently, either using the BioRS™ Integration and Retrieval System middleware or by using a dedicated API for integration of any data source as “virtual objects.”

Conceptual background

Conceptually, the BioXM system is based on user-configurable “semantic networks.” Semantic networks are directed graphs of nodes representing scientific concepts (e.g., metabolites, organisms, cell types, etc.) and edges representing semantic relations between these concepts (e.g., metabolic consumption, phosphorylation, inhibition, etc.). The framework-wide definition of such a semantic network represents the actual data model of a domain-specific implementation of the BioXM framework. Unlike the approach other software packages have taken (e.g., specialized pathway tools), the BioXM system allows the data model to be changed and extended dynamically at any time to reflect changing requirements of ongoing projects or emerging scientific disciplines. Network representation based on a scientific data model can be “standard” pathways (e.g., biochemical or signal transduction pathways) or automatically generated connectivity networks. Special “information layers” help with scientific interpretation by changing visualization dynamically based on freely configurable, science-driven views and queries. Powerful reporting and querying capabilities, dynamically adapting to the user-configured data model, allow efficient monitoring of ongoing projects or searching scientific patterns across projects or multiple data sources.

Specialized object type “templates” (called “static object types”) are the building blocks an administrator uses to create the dynamic data model. Examples of such static object types are elements, relations, contexts, experiments, and ontologies. Using these building blocks, an administrator can introduce new elements of type “Gene” or “Metabolite” as well as experiments of type “Affymetrix Experiment” or “Metabolomics Experiment”. Using the “Ontologies” object type, proprietary or public ontologies such as “Gene Ontology” or “Chemical Entities of Biological Interest” can be imported, which can be used to extend the project-specific data model. The “Context” object type is used to group specific sub-networks within the “global” semantic network, and is typically applied to model metabolic or signaling networks under different environmental conditions. Any additional metadata is modeled using configurable object-type specific annotation forms or "Experiment"-specific experiment formats, both supporting a variety of different attribute types from numeric or text types to images, thesauri or text files.

With respect to data modeling, this framework enables a top-down approach. Starting from the overall picture, one can gradually extend and refine the data model of the framework implementation according to the evolving biological models. Using this framework brings the modeling aspects of the project into focus, instead of having to actually develop core infrastructure or database systems. Algorithms or specialized applications developed for an ongoing project can be "plugged in" or can communicate with the BioXM framework using a variety of APIs, including Web services. BioXM APIs allow, for example, extension of the query language, transparent integration of external data as “virtual objects” or on-the-fly retrieval of data provided by data resources on the Internet.