The B-Fabric Life Sciences Data Management System (Extended Abstract)

Can Türker Dieter Joho Fuat Akal Christian Panse
Simon Barkow-Oesterreicher Hubert Rehrauer Ralph Schlapbach

Functional Genomics Center Zurich, Winterthurerstr 190, 8057 Zurich, Switzerland
<tuerker|joho|akal|cp|sb|rehrauer|schlapbach>@fgcz.ethz.ch

1 A Few Basics and Facts about B-Fabric

B-Fabric [1] is a life sciences data management system for integrated management of experimental data and scientific annotations. For scientific data annotation, B-Fabric provides a concise metadata schema (see Figure 1). All scientific data is organized in projects. Within a project, a user creates and generates data resources, e.g., representing the result of an experiment or an analysis. Related data resources like those belonging to the same experiment or analysis are stored together in a workunit. Every data resource can be associated with an extract. This is for instance the case when the data resource represents an experiment in which the corresponding extract was used. Extracts and samples describe the biological sources of experiments, measurements, and whatever a workunit represents.

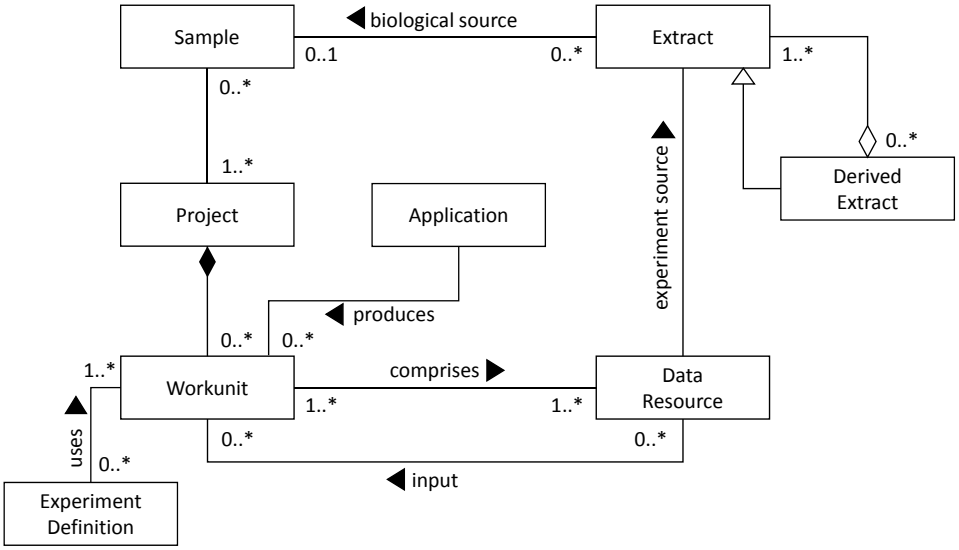

Figure 1: The Core Classes of B-Fabric's Metadata Schema (in UML)

For a user's point of view, B-Fabric provides the following functionality:

- *Register samples/extracts*: A user describes his biological sources as detailed as possible using an extensible vocabulary that is maintained and curated by experts (in our case employees of the research center).
- *Import data*: The user selects an instrument from where he wants to import data. The user then selects the files and describes the workunit. Finally, the data resources are either linked or copied to B-Fabric while the user associates the data resources to extracts.
- *Data browsing*: All main objects are bidirectionally linked together such that the user can easily browse through his/her data network.
- *Quick and advanced search*: A user may find relevant data using full-text search.
- *Direct access*: Besides full-text search, there are direct links to the central object classes, e.g. projects, samples, extracts, and workunits.
- *Data export*: A user may export the data in various ways, e.g., the entire workunit as a zip file. Since the size of a workunit may be huge, B-Fabric in the background creates a zip file and provides it for download as soon as the zip file is generated.
- *Work history*: B-Fabric saves the work history such the user can check afterwards which persistent data management operations were executed.
- *Open tasks*: B-Fabric shows the workflow tasks that the user has still to perform.
- *Run external applications*: A user can invoke external applications with B-Fabric data. Depending on the type of data resources, the B-Fabric provides the necessary invocation buttons and forms, respectively.

At the Functional Genomics Center Zurich (FGCZ), B-Fabric is running in daily business for over two years. Here are some figures about the FGCZ deployment (as of May 2009):

Users	1274	Samples	2195
Projects	664	Extracts	2375
Institutes	193	Data Resources	25756
Organizations	41	Workunits	14959

The typical users are researchers (usually PhD students interested in running experiments), lab heads (usually professors interested in the experiment results), and FGCZ employees (instrument experts supporting and carrying out the experiments). The latter have special rights, e.g., they are allowed to register applications. As the figures show, the users are coming from many different institutions and organizations.

Technically, B-Fabric is composed of distributed, loosely-coupled components based on open source technologies. Figure 2 sketches the architecture of B-Fabric. For a more detailed description, please visit `http://www.bfabric.org/`.

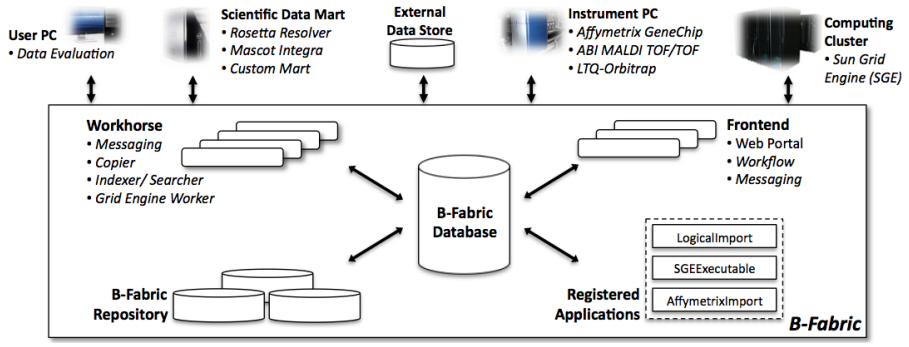

Figure 2: B-Fabric: System Architecture

The *B-Fabric repository* stores experimental data. The *B-Fabric database* manages all scientific annotations (e.g. about samples and extracts) and administrative data (e.g. about users and projects). The *frontend* acts as Web portal providing users with controlled access to the data. *Workhorses* execute specific tasks (e.g. data copying, indexing, searching) in a distributed way. Frontends and workhorses communicate via asynchronous messaging. The concept of *registered applications* allows an ad-hoc coupling of arbitrary applications with B-Fabric. These applications may be external ones, autonomously running beyond the control of B-Fabric. Applications can also be invoked within workflows to model scientific pipelines. B-Fabric interacts with several external components. *User PCs* are standard computers running a Web browser to enable access to B-Fabric through frontends. Typically, a user searches and downloads B-Fabric data for analysis reasons. Various data analysis/visualization tools are deployed on these PCs to accomplish those purposes. *Scientific data marts* correspond to external systems that provide scientific functionality. *External data stores* represent scientific data available on external systems. At any time, any external data store can be attached to and made accessible via B-Fabric. *Instrument PCs* refer to computers that are attached to an instrument that generates and holds scientific data to be imported into B-Fabric.

The following table lists the key technologies glued together in B-Fabric:

Web Development Framework	Apache Cocoon	Database Server	PostgreSQL
Object-Relational Mapping	Apache OJB	Workflow Engine	OSWorkflow
Fulltext Indexing/Search	Apache Lucene	File Transfer	OpenSSH
Asynchronous Communication	Apache ActiveMQ	Logging	Apache log4j

2 Conclusions based on Experiences with B-Fabric

Provide small metadata schema with small but extensible vocabulary. In a very dynamic research environment as the FGCZ, it is practically impossible to agree on a schema that can capture all potentially created data in the application domain at a detailed level of

granularity. Sticking one-to-one with standards such as MIAME or GO is not a solution, too. First, there are researchers doing research on issues that have not been defined yet. Second, researchers often use different notions and granularities to describe their experimental data. Third, these standards are so huge in size that for most researchers they are impractical for daily use. As an example, imagine a drop-down menu with ten thousands of terms. Consequently, B-Fabric supports a data schema with a small set of commonly agreeable attributes and restricts the underlying vocabulary to that part of corresponding standards which is potentially needed for the research technologies supported at FGCZ. To be open to new research directions, B-Fabric allows to dynamically extend the vocabularies at run-time. As soon as a new entry is provided by a user, a reviewing process is started. Depending on the project membership of that user, the coach of that project, in our case an FGCZ employee, is triggered to review and release the new entry.

Support physical as well as logical data import. To annotate experimental data, the system must first be aware of the corresponding files. Originally, B-Fabric was designed to move all experimental data into its internal repository. Such a *physical* import is indispensable in scenarios where the data cannot be maintained persistently at the instrument PC or external data store. However, a physical import into a fully encapsulated repository complicates the use and postprocessing of the data with typical analysis and visualization tools. Since such tools require direct access to the data files, the files must be downloaded to the corresponding places in the file system. Due to the size of the files and length of the scientific pipelines, this puts the researcher into an undesired long pending state. Hence, B-Fabric also supports a *logical* import (linking) of data such that the data files can reside at an external data store. Pre-configured data providers take care of the original location of the experimental data on the instrument PC and support a unified data handling based on different protocols like *ssh*, *smb*, *jdbc* etc. Since B-Fabric provides a transparent access to the data, researchers are not affected by the actually location of the data.

Loosely couple external applications. Since most expert facilities like the FGCZ have already established data processing applications and pipelines, integrating them into a common infrastructure is often a challenge. First, the original functionality is usually provided by proprietary tools whose internal processing is not known. Second, the developer that created the application often is no longer available. Third, replacing existing code eats up resources without providing new benefits for the users. We experienced that implementing connectors to such external applications is quite laborious and that requests for connecting new applications popped up faster than the implementation could progress. As a result, we introduced the *application registration* concept. First, a connector is written for a certain type of application, e.g. for running *R* scripts on an *Rserve* system. Then, a small interface is defined to describe how the application gets its input. Finally, the researcher writes the application in any language. The advantage of this approach is that an upgrade or a replacement of components of an external application is possible without touching the B-Fabric system. Nevertheless, defining the interface towards the external application is still quite tricky. Allowing too much flexibility may result in errors since the user usually is not aware of the restrictions of the corresponding external application.

Outsource data processing. Since instruments often produce data in proprietary formats, B-Fabric is not able to interpret and process the experimental data — although it knows the scientific annotations. A solution to that problem is to move the responsibility of the data processing to external applications. B-Fabric provides the application input via a small interface. An external application is then able to fetch the information from B-Fabric to process the experimental data correctly. B-Fabric then just waits for the result. This approach requires that B-Fabric trusts the external application to provide the result in a correct way. The external applications need to take additional effort to process the data. We experienced that examples and documentation on the development of connectors are vital to support stable and generally accepted applications. However, if a certain level of commitment is achieved, additional functionality can be provided quickly.

Export data. Regardless of the systems functionality, there are always users that need to go further than any system can support. The best solution to address this issue is to provide an easy access to the data. Researchers should be able to export data and do their data processing on their own. B-Fabric provides several methods for data export. For instance, search results can be exported. Another example is the download of data resources. Due to the large size of the interconnected data, B-Fabric compresses the data and provides a download link for the user. The download of predefined reports is another example. For specific needs and applications, B-Fabric allows reports that collect and prepare data for later use. If it turns out that researchers use a certain type of export very often to perform a specific task, this task becomes a candidate for B-Fabric internal functionality.

Control data access and publishing. Life sciences researchers usually demand for strict data access control while complaining about restrictions in the usage of the data for collaboration purposes. In B-Fabric, data access is controlled at project level. While the project has not reached the publish state, the data is visible to the corresponding project members only. This approach works well as soon as the projects are small and have exact goals. Disadvantages appear in case of huge projects that run for many years, especially when the researchers involved in the same project should not be able to access each others results. A proven simple but effective solution is to divide such projects into several smaller projects according to their goals and members. In this way, the scientific data can be published as soon as the corresponding research result have been accepted for publication somewhere.

References

[1] Türker, C., Stolte, E., Joho, D., Schlapbach, R.: B-Fabric: A Data and Application Integration Framework for Life Sciences Research. In: Data Integration in the Life Sciences 2007, DILS 07, pp. 37-47, Springer-Verlag, 2007.

[2] Türker, C., Joho, D., Panse, C., Barkow-Oesterreicher, S., Akal, F., Schlapbach, R.: Application Coupling and Data Feeding in an Integrative Life Sciences Data Management System To appear in: Proc. of the Int. Conf. on Bioinformatics, Computational Biology, Genomics and Chemoinformatics, Orlando, FL, USA, July 13-16, 2009.