Life Sciences Data and Application Integration with B-Fabric

Can Türker, Fuat Akal, Ralph Schlapbach

Functional Genomics Center Zurich (FGCZ), UZH / ETH Zurich, Winterthurerstrasse 190,
CH–8057 Zurich, Switzerland, http://www.fgcz.ethz.ch
⟨tuerker|akal|schlapbach⟩@fgcz.ethz.ch

Summary

In this demo paper, we sketch B-Fabric, an all-in-one solution for management of life sciences data. B-Fabric has two major purposes. First, it is a system for the integrated management of experimental data and scientific annotations. Second, it is a system infrastructure supporting on-the fly coupling of user applications, and thus serving as extensible platform for fast-paced, cutting-edge, collaborative research.

1 Introduction

Core facilities like the Functional Genomics Center Zurich (FGCZ) typically face with the problem of efficiently supporting research. On the one hand, the usage of the facility, i.e., the access to its instruments and staff, must be coordinated. On the other hand, the huge amounts of data generated with the instruments and applications of the facility must be managed, made accessible, and finally archived. Without an integrated data management solution these tasks are hardly to accomplish. Consequently, we have developed B-Fabric [1], a web-based data management system for supporting life sciences projects and integrating the produced data and needed applications.

B-Fabric is composed of loosely-coupled components based on open source technologies, such as PostgreSQL as database server for structured data, Lucene as full-text search engine, and ActiveMQ for asynchronous communication between the B-Fabric components. B-Fabric is available under Reciprocal Public License (RPL). For a description of the B-Fabric architecture and its underlying technologies, please check [2]. In this demo paper, we briefly mention the functionality B-Fabric provides at its interface and sketch some of the central features.

For the interpretation and reuse of experimental data, scientific annotations are crucial. Based on our experience with many bioinformaticians and researchers and their practical experiences and difficulties with using standard schemas MIAME or Gene Ontology, we decided to apply a concise metadata schema approach for B-Fabric.

All scientific data is organized in *projects*. Within a project, a user creates and generates data resources. A *data resource* is an abstraction of a file or link to a file. Examples for data resources are raw files produced from a mass spectrometer or cel files generated from an array scanner. Each data resource is connected to an *extract* representing the biological input into the experiment or measurement that produced the data resource. We distinguish between samples and extracts describing the biological sources at different levels. The *sample* contains general information about the biological source while the extract represents an extraction of that source

which actually is used for the experiment or measurement. There might be several extracts of one sample. These extracts might be the result of different extraction procedures.

As a result of extensive discussions between the bioinformaticians and researchers at FGCZ about the question what primary (and secondary) data should be stored and especially what this data actually represents, the generic concept of a workunit was found in B-Fabric. A *workunit* is an abstraction that can be used to represent the result of an experiment, a measurement, an analysis, a search etc. In principle, a workunit is a container referencing to data resources that logically form a unit. Some of these data resources are marked as input resources meaning that they were the inputs of the processing step (*application*) that created the remaining data resources. The scientist individually decides what a workunit should represent.

2 Use Cases

From a typical researcher's point of view, B-Fabric provides the following functionality:

- *Manage projects*: A researcher can add and remove users to or from his projects to allow or restrict access to data stored within the context of a project.
- *Register samples/extracts*: The researcher describes his biological sources as detailed as possible using an extensible vocabulary that is maintained and curated by experts (in our case by FGCZ employees). This support is basis for semantic enrichment of experimental data in a dynamic research environment as the FGCZ.
- *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. This step is crucial for the correct interpretation and later reuse of the imported data especially by third parties when it become accessible after project publication.
- *Data browsing*: All core B-Fabric 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. A search may vary from certain attributes of certain objects to the content of readable attachments and data resources. This support is important as soon as the stored data sets become larger such that data browsing become apparently inefficient. Above that, it is needed for searching data of published projects.
- *Direct access*: Besides full-text search, there are direct links (shortcuts) to the core B-Fabric 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 torrent file or as a report of a given template. Since the size of a workunit may be huge, B-Fabric creates a torrent file and provides it for download. For specific needs and applications, B-Fabric allows reports that collect and prepare data for later use.
- *Run external applications*: A user can invoke external applications with B-Fabric data. These applications are provided by experts who integrate them seamlessly into B-Fabric.

- *Open tasks*: B-Fabric shows the workflow tasks that the user has still to perform. This helps a lot to remember and softly force the user to terminate started workflows.

From a research center's point of view, B-Fabric additionally supports typical tasks that are crucial in efficiently running daily business in a such center:

- *Manage project*: To get access to the resource of the research center, e.g. using an instrument, a user needs to participate in a running project. B-Fabric supports the entire project management process, from the application submission over the reviewing to project member management. This feature is central for efficiently managing a core facility like FGCZ.
- *Register user*: Users themselves register and provide all personal contact information that is needed to grant access to the resources (computers, instruments etc.) of the research center. With the registration corresponding user accounts are generated such that the users can use all services with the same login and password. This frees the system admin from a lot of work.
- *Register external application*: A user with special access rights can couple external applications with B-Fabric such that these application can be invoked and fed via B-Fabric (for details see [3]). Based on some generic workflows and pageflows, a large bunch of applications can be added to the system on-the-fly. This feature is the basis for integrating already established data processing applications and pipelines into a common framework and extending the system's functionality in a loosely-coupled way. Experts can easily make new applications (written in any language and running anywhere) available to the whole user community. Depending on the type of data resources, B-Fabric provides the necessary invocation buttons and forms, respectively.
- *Merge Duplicates*: Some users tend to recreate a new user account due to various reasons. Eliminating such duplicates is the basis for an integrated view and analysis of a user's data and more importantly avoids unnecessary communication problems between a user and the staff of the research center. The duplicate merging is not only supporting for user information but also for other objects types like scientific annotations and institutional contact information.
- *Order (Door) Key*: A user that needs physical access to the userlab of the research center, requires a door key. Since B-Fabric maintains personal data of the users, ordering a door key at the university administration is a just one click away. This save a lot of time for the center's secretary as well as for the user.

3 Demonstration

In this demo paper, we sketch some central features of B-Fabric. As example scenario, we use a scientist who is working on a plant named Arabidopsis Thaliana with the goal to figure out the effect of certain gene and the effect on light on it. For this purpose, he registers his samples and extracts with B-Fabric, loads his data into B-Fabric and defines his experiment. Afterwards, he runs his experiment and stores the results in B-Fabric, as illustrated in Figure 1.

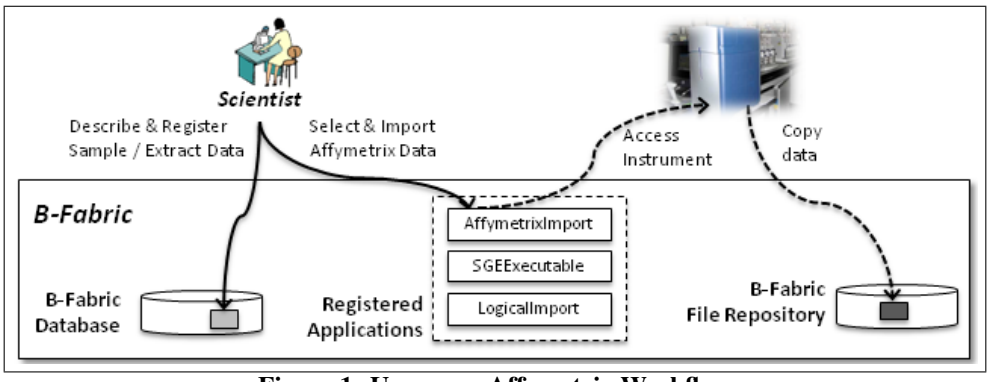

Figure 1: User runs Affymetrix Workflow

In the following, screen-shots are used to illustrate the different steps of the mentioned workflow. To avoid cluttering, the figures are clipped accordingly.

3.1 Register Samples/Extracts

Users register their samples and extracts through intuitively designed forms (see Figure 2 and 3). Data entering is facilitated by providing as much drop-down menus as possible to select annotations from the system vocabularies and by dynamically drawing forms according to selected annotation values.

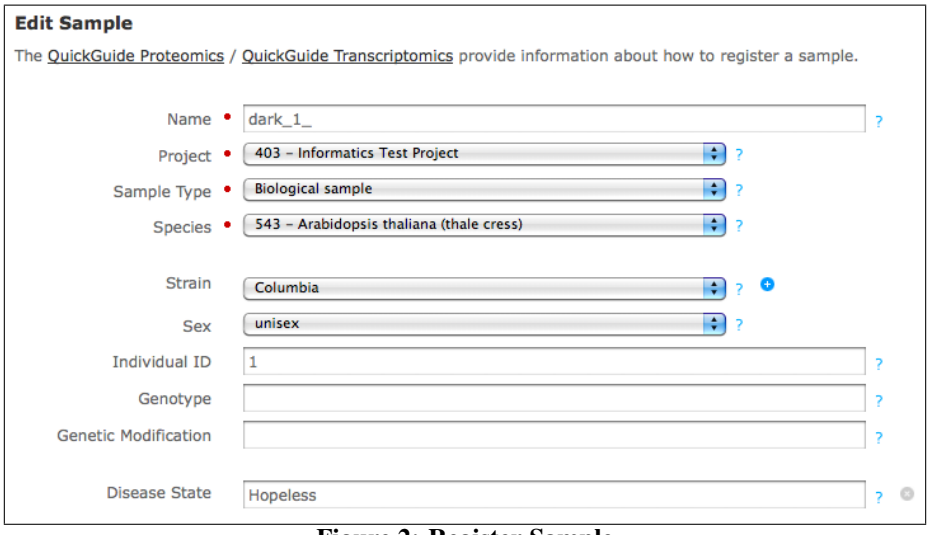

Figure 2: Register Sample

In addition, users typically register several samples and extracts where only a few attributes differ. In order to further ease the registration, cloning as well as batch registration of samples and extracts are supported.

3.2 Data Import

B-Fabric supports two ways of data import: 1) physically copying and 2) linking data files. To import data from a data source, a proper data provider must be configured. In principle, data can be imported from any configured source that supports ssh for secure data transfer. The

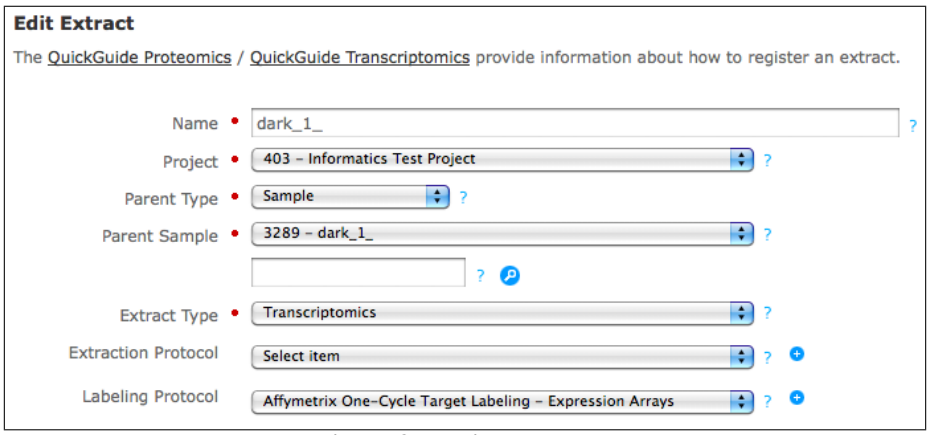

Figure 3: Register Extract

B-Fabric deployment at FGCZ allows importing data files from local file systems as well as several instruments available at FGCZ. New data providers can be added to the system easily. With the configuration of a data provider the selection of the data files in corresponding data stores can be restricted to the ones that are potentially relevant for the user. This is a crucial feature since the number of the data files can be huge. An import results in a workunit. Figure 4 shows the screen where a workunit is created by fetching files from the Affymetrix GeneChip instrument, which is an instrument already known to B-Fabric.

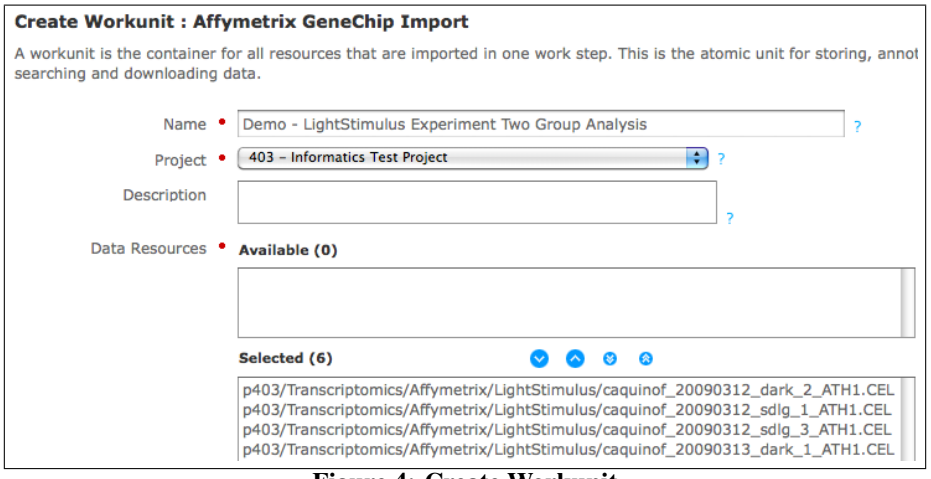

Figure 4: Create Workunit

B-Fabric implements the data import via workflows. With the initiation of a data import, the corresponding workflow becomes visible to the user. The next step to be taken by the user is highlighted in the graphical representation of the workflow. In data import workflow, for instance, the user must assign extracts to the imported files. The workflow-driven approach of B-Fabric is very useful in practice to reduce human mistakes and avoid skipping steps.

Assigning extracts to data resources also comes with some intelligence in B-Fabric. When the scientist goes to the assign extracts screen, he gets already the best matches between data resources and extract names. Typically he just needs to press the save button and continue.

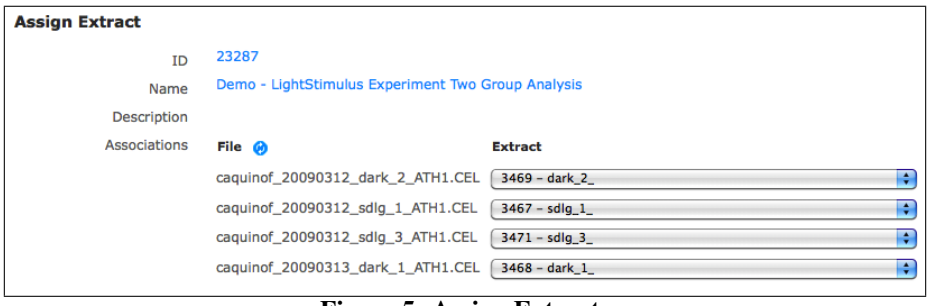

Figure 5: Assign Extracts

3.3 Application Integration

External functionality can be coupled with B-Fabric via *applications*. 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 (see Figure 6). Finally, the scientist writes the application in any language. This on-the-fly coupling of external applications is a crucial feature of B-Fabric, which allows fast extension and evolution of the system.

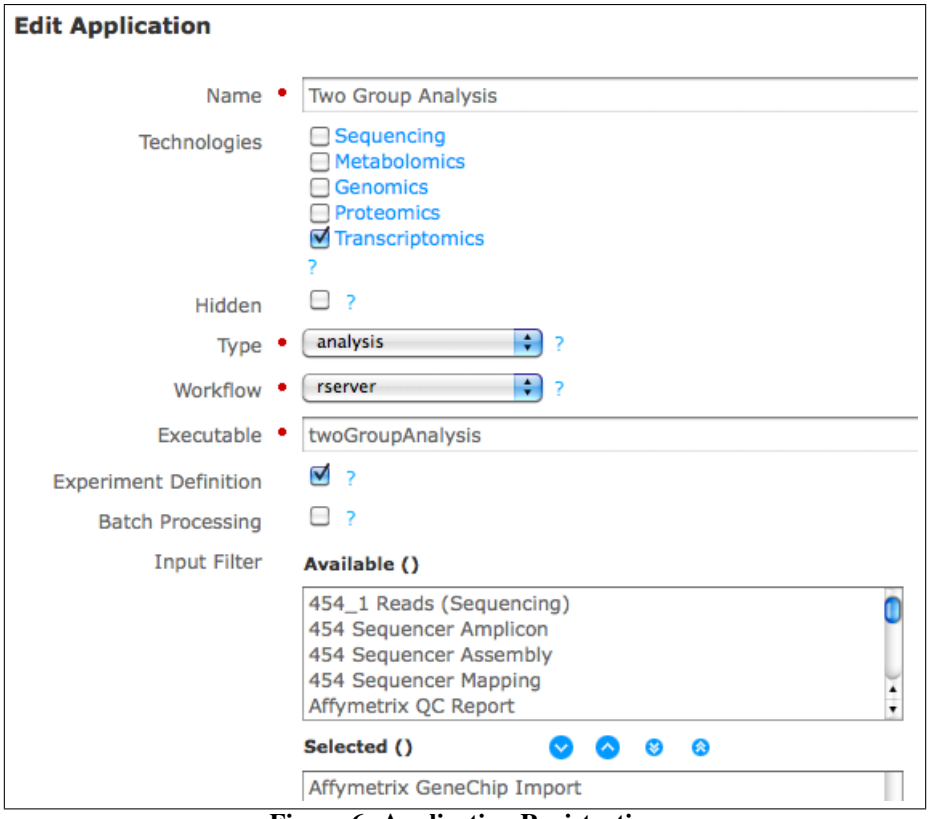

Figure 6: Application Registration

Once an application is registered, an experiment can be created to run this application. As an example, Figure 7 shows the definition of the experiment that will be conducted on the Arabidopsis Thaliana plant as mentioned earlier in this section. Defining an experiment consists of a selection of data resources, samples, extracts, and arbitrary number of attributes (e.g. species and treatment in the example.) that will be used as input for the application.

Figure 8 shows how easily a previously registered application (two group analysis) can be

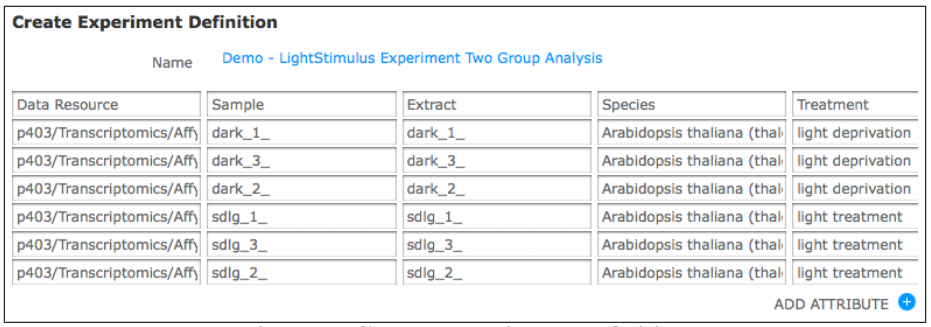

Figure 7: Create Experiment Definition

invoked to conduct the desired experiment. This step requires a name for the resulting workunit which contains the result files of the application along with specific parameters regarding the experiment, e.g. reference group.

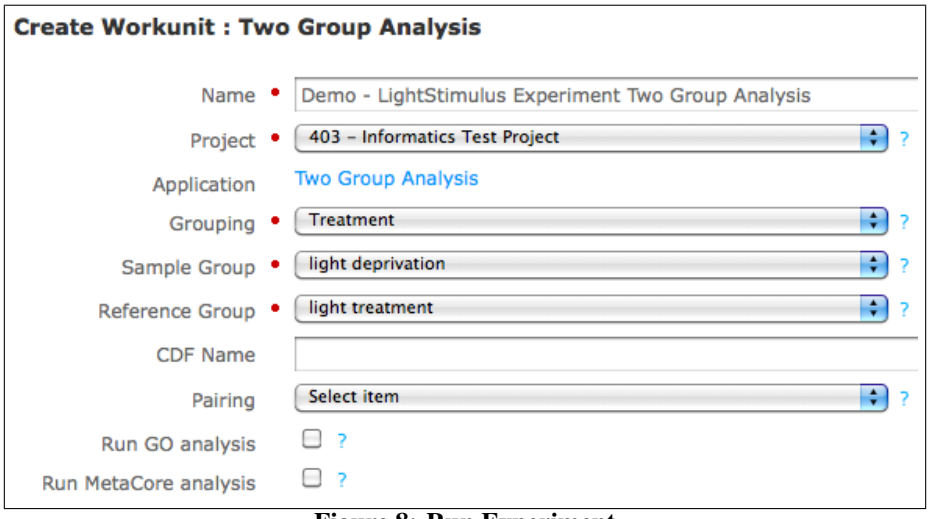

Figure 8: Run Experiment

Once the experiment is started, a corresponding workflow is initiated. The graphic presentation of the workflow is also used to show what is happening underneath in the system. The example workflow (generate an R report) is quite simple and consists of a single step (see Figure 9). Note that B-Fabric can support arbitrary complex workflows based on its underlying workflow engine.

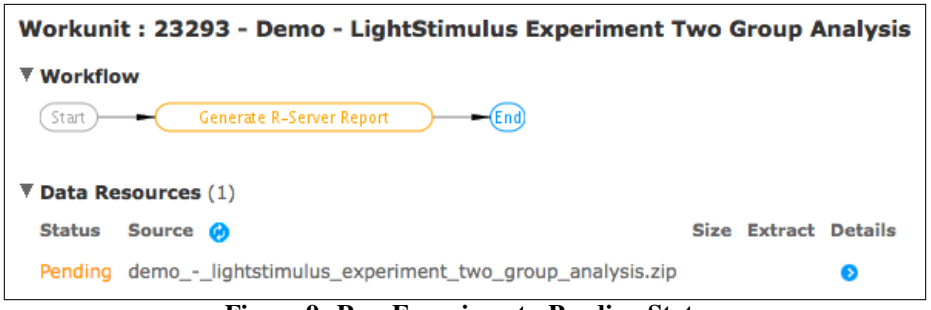

Figure 9: Run Experiment - Pending State

When the experiment is done, the scientist can view the experiment results by clicking the proper link on the screen (see Figure 10). The results of the experiment are also presented to the user as a zip file so that they can easily be transferred to another medium.

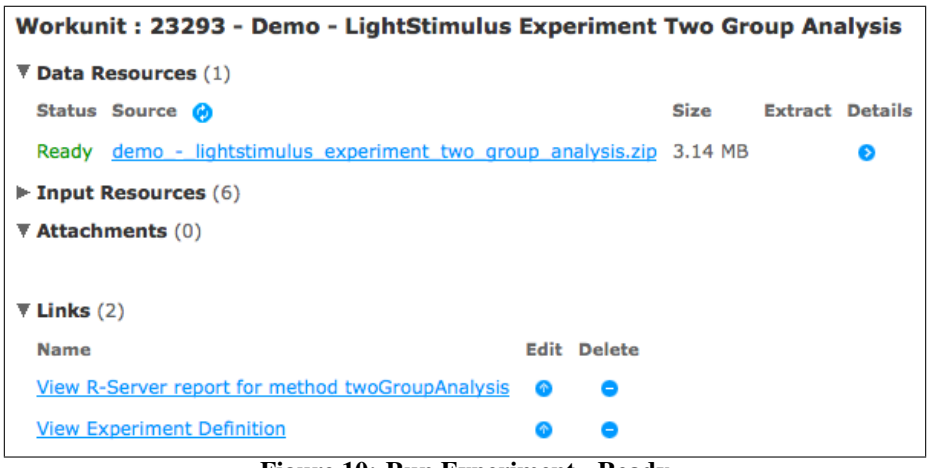

Figure 10: Run Experiment - Ready

4 Conclusions

B-Fabric is running in daily business at FGCZ since beginning of 2007. Here are some figures about the FGCZ deployment as of July 2011:

Users	2129	Projects	985	Samples	5917	Extracts	7708
Institutes	394	Organizations	98	Data Resources	84494	Workunits	54775

We now conclude this paper by sharing some experiences we made with running B-Fabric at FGCZ:

- *Metadata Schema with 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. Life science research data require two types of annotations in order to be able to interpret the contained information: (i) attributes describing the measurement parameters and (ii) attributes characterizing the biological samples measured. Both types cannot be well covered with a fixed annotation vocabulary. A life science data management system has to provide a flexible annotation scheme in order to be used at a center like the FGCZ that serves many different research fields.

- *Physical and Logical Data Import*. To annotate experimental data, the system must first be aware of the corresponding files. Originally, B-Fabric was designed to move or copy 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 for different reasons. 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,i.e. copied, 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.

- *Loosely Coupling of 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. Furthermore, there exists another problem we encountered at the FGCZ: the computer science field and the bioinformatics field have a different software development culture. Bioinformaticians, e. g., are usually no trained software developers using sophisticated methods that ensure software quality. Programming for the always changing requirements in the life sciences research groups has to be adaptive, with fast cycles and close user interaction. Expert knowledge in the application field is often more relevant than a thorough software development training. This results often in rapidly written, script based, glue-type programs that are used only by a small number of users. Examples for such types of applications are visualization of mass spectra data, conversion tasks, or Gene Ontology enrichment analysis of microarray expression data. To speed up the integration into the B-Fabric system it was beneficial to completely separate the bioinformatics development from the infrastructure development and therefore minimize the need for communication.

- *Outsource Data Processing*.

  Not all data processing needed in a life sciences data management can be done within a single monolithic system. Since instruments often produce data in proprietary formats and the processing sometimes has to be done using vendor libraries that are installed on specific workstations, limiting amongst other things, the choice of operating system, B-Fabric is not able to interpret and process the experimental data itself — although it knows all the scientific annotations. A solution to this problem is to move the responsibility of data processing to external applications. B-Fabric provides the application input via a generic interface. An external application is then able to fetch the information to process the experimental data correctly while B-Fabric awaits the results before importing them back into it's database. This approach means that B-Fabric has to trust the external application to provide the results in a correct way which means that additional effort has to be done on the application's side.

- *Data Export*.

  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 the user with diverse export and reporting features, including an easy access to the raw data for further external processing. B-Fabric provides several methods for data export. For instance, results of a B-Fabric search query can be exported in csv or xls format. The user is able to search the B-Fabric database, i. e., the visible part according to his access rights, based on a Apache Lucene index, filter the results on screen according to his interest, and download a report. Another example of data access is the download of entire data resources. Every