Database Requirements for Workflow Management in a High-Throughput Genome Laboratory[1]

Anthony J. Bonner[2]
bonner@db.toronto.edu

Adel Shrufi[2]
shrufi@db.toronto.edu

Steve Rozen[3]
steve@genome.wi.mit.edu

[2]University of Toronto
Department of Computer Science
10 King's College Rd
Toronto, ON, Canada
M5S 1A4

[3]Whitehead/MIT
Center for Genome Research
One Kendall Square
Building 300, Floor 5
Cambridge, MA 02139, USA

Presented at the *NSF Workshop on Workflow and Process Automation in Information Systems: State-of-the-Art and Future Directions*, May 8–10, 1996, Athens, Georgia.

This and related papers are available at the following web page:
`http://www.db.toronto.edu:8020/people/bonner/bonner.html`

1 Introduction

A critical requirement for a large genome laboratory is software to control laboratory workflow while managing the data produced in the laboratory. This software knits together a complex web of manual and automated laboratory activities, including experiment scheduling and setup, robot control, raw-data capture, multiple stages of preliminary analysis and quality control, and release of finished results. Appropriate software is necessary to make the coordination of these activities both intellectually manageable and operationally efficient. A key component of this software is a database management system (DBMS) for controlling and tracking workflow activity. This DBMS maintains a record of what happened in the laboratory. This record constitutes an audit trail, or *event history*, and has much the same function as a laboratory notebook: it records what was done, when it was done, who did it, and what the results were. It can also be used to analyze workflows, to find rate limiting steps or to investigate anomalous results. This paper discusses the requirements of this DBMS.

The discussion centers on *LabFlow-1* [3, 2], a recently developed database benchmark for high-throughput workflow management systems (WFMSs), *i.e.*, systems for managing high-volume, mission-critical workflows. LabFlow-1 is based on the data and workflow management needs of a large genome laboratory, and reflects their real-world experience. An overview of the benchmark can be found in [3], and a detailed description in [2]. Benchmark software is available at the following web site: `ftp://db.toronto.edu/pub/bonner/papers/workflow/software/`

Although it is based on genome-laboratory workflow, we believe that LabFlow-1 captures the database requirements of a common class workflow management applications: those that require a

[1]This work was supported by funds from the U.S. National Institutes of Health, National Center for Human Genome Research, grant number P50 HG00098, and from the U.S. Department of Energy under contract DE-FG02-95ER62101.

production workflow system [16]. In production workflow, activities are organized into a kind of production line, involving a mix of human and computer activities. Examples in business include insurance-claim or loan-application processing. Production workflow systems are typically complex, high-volume, and central to the organizations that rely on them; certainly these characteristics apply to the laboratory WFMSs used in high-throughput genome laboratories. Many production workflows are organized around central materials of some kind, which the workflow activities operate on. Examples of central materials include insurance claims, loan applications, and laboratory samples. As a central material is processed, workflow activities gather information about it.

Production workflow systems include the class of *Laboratory Information Management Systems*, or LIMS [21, 1, 19]. LIMS are found in analytical laboratories in a wide range of industries, including pharmaceuticals, health care, environmental monitoring, food and drug testing, and water and soil management. In all cases, the laboratory receives a continual stream of samples, each of which is subjected to a battery of tests and analyses. Workflow management is needed to maintain throughput and control quality [18].

2 Performance and Database Requirements

Much of the research on workflow management in computer science has focussed on developing extended transaction models, especially in a heterogeneous environment [10, 20, 6, 14, 13, 8, 22]. However, the *performance* of workflow management systems has so far received little attention. The need to study performance arises because commercial products cannot support applications with high-throughput workflows [10].

High-throughput workflows are characteristic of large genome laboratories, such as those operated at the Whitehead/MIT Center for Genome Research (hereafter called "the Genome Center"). Workflow management is needed to support the Genome Center's large-scale genome-mapping projects [4, 15, 5]. Because of automation in sample handling and testing, instrumentation, data capture and workflow management, transaction rates at the Genome Center have increased dramatically in the last three years, from processing under 1,000 queries and updates per day in 1992 [11], to over 15,000 on many days in 1995. Of course, peak rates can be much higher, with a rate of 22.5 updates and queries per second recently observed over a 5-minute period. These rates are expected to increase by another order of magnitude in the near future if the Genome Center begins large scale sequencing of the Human genome [4]. Moreover, unlike the simple banking debit/credit transactions of some TPC benchmarks [23], these transactions involve complex queries, plus updates to complex objects, such as arrays, sequences, and nested sets.

The LabFlow-1 benchmark is a first step towards measuring the performance of WFMSs. It does not account for all the components that affect performance, such as networks, hardware platforms, and operating systems. Instead, it focuses on one dimension of the problem: the DBMS that controls and tracks workflow. Like other components, the DBMS can become a workflow bottleneck, especially in high-throughput applications. Certainly, this is often the case at the Genome Center.

Workflow management at the Genome Center imposes numerous requirements on the DBMS. First, it requires standard database features, such as concurrency control, crash recovery, consistency maintenance, a high-level query language, and query optimization. It also requires complex data types. The DBMS must provide this functionality on a mixed workload of queries and updates. In addition, it must provide the following two features, which are typical of many WFMSs:

Event Histories. The DBMS must maintain an audit trail, or event history, of all workflow activity. From this history, the DBMS must be able to quickly retrieve information about any material or activity, for day-to-day operations. The history is also used to explore the cause of unexpected workflow results, to generate reports on workflow activity, and to discover workflow bottlenecks during process re-engineering. The DBMS must therefore support queries and views on an historical database. We note that many commercial laboratories are legally bound to record event histories, since "Accountability is critical in tracking who is responsible for data and its approval for release" [17]. Salient examples include clinical drug trials and environmental testing.

Dynamic Schema Evolution. A hallmark of modern workflow management is that workflows change frequently, in response to rapidly changing business needs and circumstances [7, 10]. Typically, a workflow will acquire new activities and existing activities will evolve. In both cases, the changed workflow generates new kinds of information, which must be recorded in the database. This requires changes (usually additions) to the database schema, preferably while the workflow is in operation (so-called *dynamic workflow modification*).

It is worth observing that because the database is historical and the schema is evolving, data at different points in the history will be stored under different schemas. Thus, an historical query or view may access data with many different schemas. This presents a challenge both to database design and to the formulation of queries and views. For instance, an application program may have to query an object's schema, as well as its value.

In sum, the database requirements of the Genome Center are typical of applications with the following characteristics: (*i*) high-volume, mission-critical workflows, (*ii*) frequent workflow change and process re-engineering, (*iii*) an audit trail of workflow activity, and (*iv*) complex-structured data. The LabFlow-1 benchmark is intended for such applications.

3 Workflow in LabFlow-1

This section provides an overview of data and workflow management in LabFlow-1 from the perspective of the DBMS. To keep the discussion concrete, we frame it in terms of laboratory workflow. Additional details can be found in [2].

The database contains two main kinds of object: *materials* and *steps*. In object-oriented terms, the database has a material class and a step class, with subclasses representing different kinds of materials and steps. Step and material instances (objects) are created in two distinct situations. (*i*) As the laboratory receives or creates new materials, new material instances are created in the database to represent them. (*ii*) Each time a material instance is processed by a workflow step, a new step instance is created in the database to represent the activity. The step instance records the results of the workflow step (*e.g.*, measurements of length and sequence) and the conditions under which the step was carried out (*e.g.*, temperature and salinity).

As a material is processed by a workflow, more-and-more step instances are created in the database, all associated with the same material. These steps constitute the material's event history. Workflow steps are not always successful, however, and earlier steps sometimes have to be repeated. Thus, the event history of a material may contain several different (though similar) versions of a step. Commonly, the most recent version is of greatest interest to scientists, since it represents the most up-to-date results.

As described above, the database schema depends on the workflow. For each kind of workflow step, there is a class in the database, and for each measurement made by the step, the class has an attribute. Consequently, workflow changes are reflected in the database as schema changes. In particular, as laboratory steps change, the corresponding classes also change. For instance, if scientists decide that a particular step should measure more properties, then attributes must be added to the corresponding class in the database. Likewise, if scientists add a new step to the workflow, then a new class is added to the database schema. If scientists split a complex step into a combination of simpler steps, then new classes are introduced into the schema to represent the simpler steps.

Classes are never removed from the database schema, even if the corresponding steps are no longer carried out in the laboratory. This is because the database is historical: old classes are needed to store and interpret data gathered under earlier workflows. Thus, the database schema expands (as classes are created and attributes are added to them), but never shrinks. In effect, as a step evolves, new versions of the step are created. Each step object is associated forever with the same version of a step class; so schema changes do not require data re-organization. A similar approach to schema evolution can be found in [24].

The data representation described above is *event oriented*. That is, information about a step is kept in one place, but information about a material is scattered among different steps. This provides a straightforward record of laboratory activity, but an awkward representation of materials. In particular, retrieving information about a material requires a detailed knowledge of laboratory workflow. For each property of a material, one must know what step(s) measured its value. Moreover, because workflows change constantly, a detailed knowledge of workflow changes and workflow history are also needed. This is a common occurrence in laboratory notebooks. When the notebook is stored in a database, the problem is compounded, since application programs may have to be changed each time the workflow changes.

To alleviate these problems, the database provides a view that is *material oriented*, *i.e.*, in which information is associated with materials, not steps. In the database, observed values (*e.g.*, sequence, length, and mass) are attributes of *steps*; while in the view, they are attributes of *materials*. Using the view, an application programmer can retrieve the length of a DNA segment without knowing what step measured the length. In this way, the view isolates the application programmer from the details of workflow and workflow change.

Defining this view is not a straightforward matter. For one thing, the view definition depends on the workflow and its history. For another, different instances of the *same* kind of material can have *different* attributes in the view. For instance, segments of DNA may or may not have a `length` attribute, depending on whether or not they were processed by a step that measured their length. Thus, the attributes (or type) of a material depend on the *history* of the material, as well as its class. This rather dynamic feature reflects the flexibility demanded by workflow management.

In addition to views, the database must support historical queries. Most of these queries can be divided into four categories:

- Queries that look-up a particular experimental step. For instance, "Retrieve the most recent hybridization step involving clones `C123` and `TC456`". Like material-oriented views, these queries are used in the day-to-day operation of the lab and are very common.

- Queries that examine the workflow history of a particular material. For instance, "Retrieve all steps involving clone `C123`". These queries are used to explore anomalous experimental results, often via an interactive browser.

- Data-dredging queries. These queries are aimed at identifying bottlenecks in a workflow, and they typically involve aggregation. For instance, "How often was step 1 repeated after step 3".

- Report-generation queries. These queries scour the event history to produce summaries of laboratory activity. The summaries are hierarchically structured, so the query answer is typically a nested set or a nested list (*e.g.*, a list of lists of lists).

Detailed examples of queries and updates are given in [2].

The functions described above come largely under the heading of workflow *tracking*. In addition, a workflow management system must provide a means of workflow *modeling* [10]. A workflow model specifies the dependencies among workflow steps. The Genome Center represents the most important dependencies as a *workflow graph* [25, 2]. Workflow graphs are based on the idea that each material has a workflow *state*, and as the material is processed, it moves from one state to another. The workflow graph largely determines the workload for the DBMS. A sample graph is given in [2], one that forms the basis of the workload for the LabFlow-1 benchmark.

It is worth noting that the idea of assigning states to materials contrasts with transactional workflow, in which states are assigned to long-running activities (*e.g.*, [22, 26, 9]). This difference might be resolved if each material were associated with a single long-running activity. This activity would exist as long as the material is being processed, and would correspond to the sequence of workflow steps that process the material. In this case, material state and activity state might be identified.

References

[1] *Standard Guide for Laboratory Information Management Systems (LIMS).* American Society for Testing and Materials, 1916 Race St., Philadelphia PA 19103, U.S.A, 1993.

[2] A. Bonner, A. Shrufi, and S. Rozen. LabFlow-1: a database benchmark for high-throughput workflow management. Technical report, Department of Computer Science, University of Toronto, 1995. 53 pages. Available at `http://www.db.toronto.edu:8020/people/bonner/bonner.html`.

[3] A. Bonner, A. Shrufi, and S. Rozen. LabFlow-1: a database benchmark for high-throughput workflow management. In *Proceedings of the International Conference on Extending Database Technology (EDBT)*, number 1057 in Lecture Notes in Computer Science, pages 463–478, Avignon, France, March 25–29 1996. Springer-Verlag.

[4] *Communications of the ACM*, 34(11), November 1991. Special issue on the Human Genome Project.

[5] N. G. Copeland et al. A genetic linkage map of the mouse: Current applications and future prospects. *Science*, 262:57–66, Oct. 1993.

[6] U. Dayal, H. Garcia-Molina, M. Hsu, B. Kao, and M.-C. Shan. Third generation TP monitors: A database challenge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 393–397, Washington, DD, May 1993.

[7] E. Dyson. Workflow. In *Forbes*, page 192. November 23 1992.

[8] A. K. Elmagarmid, editor. *Database Transaction Models for Advanced Applications.* Morgan Kaufmann, San Mateo, CA, 1992.

[9] H. Garcia-Molina, D. Gawlick, J. Klein, K. Kleissner, and K. Salem. Modeling long-running activities as nested sagas. *Bulletin of the Technical Committee on Data Engineering (IEEE Computer Society)*, 14(1), March 1991.

[10] D. Georgakopoulos, M. Hornick, and A. Sheth. An overview of workflow management: From process modeling to infrastructure for automation. *Journal on Distributed and Parallel Database Systems*, 3(2):119–153, April 1995.

[11] N. Goodman. An object oriented DBMS war story: Developing a genome mapping database in C++. In W. Kim, editor, *Modern Database Management: Object-Oriented and Multidatabase Technologies*. ACM Press, 1994.

[12] J. Gray, editor. *The Benchmark Handbook for Database and Transaction Processing Systems*. Morgan Kaufmann, San Mateo, CA, 1991.

[13] M. Hsu, Ed. Special issue on workflow and extended transaction systems. *Bulletin of the Technical Committee on Data Engineering (IEEE Computer Society)*, 16(2), June 1993.

[14] M. Hsu, Ed. Special issue on workflow systems. *Bulletin of the Technical Committee on Data Engineering (IEEE Computer Society)*, 18(1), March 1995.

[15] T. J. Hudson et al. An STS-based map of the human genome. *Science*, 270:1945–1954, Dec. 1995.

[16] S. Khoshafian and M. Buckiewicz. *Introduction to Groupware, Workflow, and Workgroup Computing*. John Wiley & Sons, Inc., 1995.

[17] D. Mattes. LIMS and good laboratory practice. In [19], pages 332–345. 1985.

[18] R. McDowall. Introduction to laboratory information management systems. In [19], pages 1–16. 1985.

[19] R. McDowell, editor. *Laboratory Information Management Systems: Concepts, Integration, Implementation*. Sigma Press, Wilmslow, U.K., 1985.

[20] C. Mohan. Tutorial: A survey and critique of advanced transaction models. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, page 521, Minneapolis, MN, May 1994. Tutorial.

[21] A. S. Nakagawa. *LIMS: Implementation and Management*. Royal Society of Chemistry, Thomas Granham House, The Science Park, Cambridge CB4 4WF, England, 1994.

[22] M. Rusinkiewicz and A. Sheth. Specification and execution of transactional workflows. In W. Kim, editor, *Modern Database Systems: The Object Model, Interoperability, and Beyond*. Addison-Wesley, 1994.

[23] O. Serlin. The history of debit credit and the TPC. In [12], chapter 2, pages 19–117.

[24] A. Skarra and S. Zdonick. The management of changing types in an object-oriented database. In *Proceedings of the Conference on Object-oriented Programming Systems, Languages, and Applications*, pages 483–495, 1986.

[25] L. Stein, S. Rozen, and N. Goodman. Managing laboratory workflow with LabBase. In *Proceedings of the 1994 Conference on Computers in Medicine (CompMed94)*. World Scientific Publishing Company, 1995. In press. Available at `ftp://genome.wi.mit.edu/pub/papers/Y1995/workflow.ps.Z`.

[26] H. Wächter and A. Reuter. The ConTract model. In A. K. Elmagarmid, editor, *Database Transaction Models for Advanced Applications*, pages 219–264. Morgan Kaufmann, San Mateo, CA, 1992.