Papyrus GIS Demonstration

W. Hasan, M. Heytens, C. Kolovson, M.-A. Neimat, S. Potamianos, D. Schneider

Hewlett-Packard Labs
1501 Page Mill Road
Palo Alto, CA 94304
email: *lastname*@hpl.hp.com

The goal of the Papyrus project [3] is to provide tools and services to enable the integration and parallelization of specialized data managers so that data-intensive applications can be constructed easily and efficiently. In our terminology, a *data manager* (DM) is a set of specialized methods that manage persistent data. A collection of *functions* defines the interface to a DM and provides the only means of accessing its persistent data.

The focus of the demonstration is on the integration aspect of the project. Through the tools provided by the Papyrus system, a number of specialized data managers can be integrated and users or applications may pose queries that span the integrated data managers' functions. In this demonstration, we emphasize the need for a comprehensive cost-based query optimization strategy to achieve high performance [6]. We show through examples that the lack of such a strategy can result in unacceptable performance.

A simple Geographic Information System (GIS) was chosen as a demonstration vehicle since GIS is an application that typically requires the manipulation of spatial data, as well as attribute data as traditionally handled by relational DBMSs. To build the application, we integrated:

- A commercial relational storage manager similar to System R's RSS. It is the storage component of Hewlett-Packard's Allbase relational DBMS. We used it to store attribute data and refer to it as the Attribute Data Manager (ADM).
- A commercial spatial DM that was obtained from Etak Inc.[1] We used it to store spatial objects representing streets. We refer to it as the Street Data Manager (SDM).
- A simple spatial DM that we prototyped, and which we used to store point spatial objects representing businesses. We refer to it as the Business Data Manager (BDM).

The demonstration shows how queries that span the three data managers are optimized.

The Papyrus Interface Language (PIL) [5] is the common interface to all Papyrus-registered DM functions. It supports the invocation of functions, the communication of data between functions, and the combination of data-retrieval and general computation. It is a functional language with declarative constructs that permit the optimization of data-intensive applications.

To register a DM [2] with the Papyrus system, the following attributes of the DM must be disclosed. This information is used during query optimization and query execution.

1. The signatures of the functions that define the interface to the DM.
2. For each function, the method(s) that implement it and their binding constraints.
3. Conversion methods from/to the DM data formats to the data formats used by the Papyrus system.
4. The cost of executing a single invocation of each method.
5. Statistics about persistent data managed by the DM.
6. Semantic properties of methods expressed through *rewrite rules*.

[1] Etak Inc. (Menlo Park, CA) is a company that designs vehicle navigation devices and produces digital map databases.

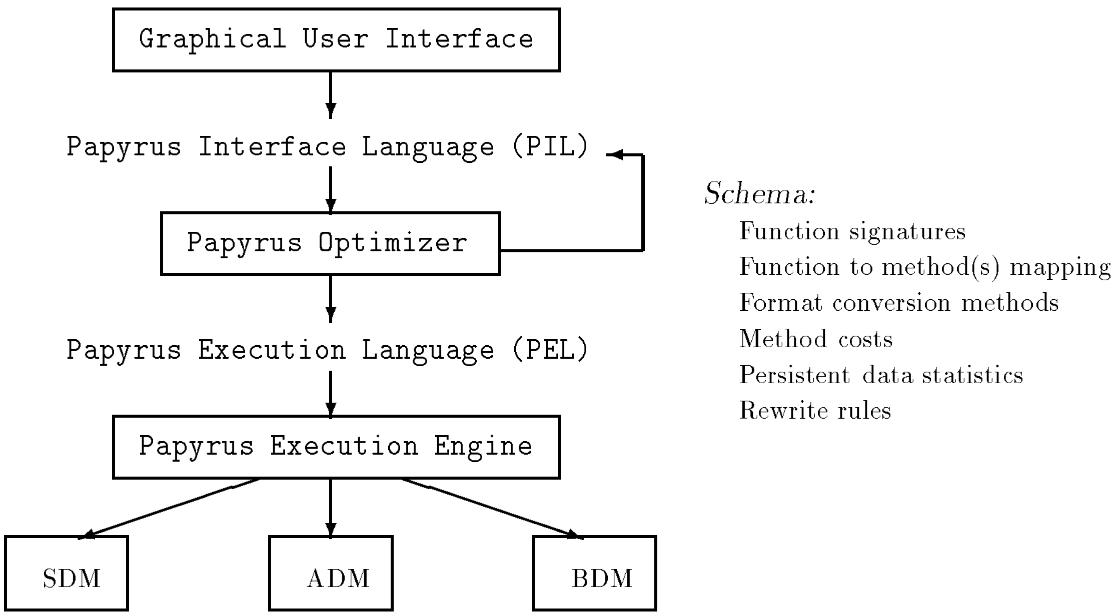

Figure 1: GIS Prototype Architecture

Figure 1 illustrates the architecture of the Papyrus prototype. Programs are submitted to Papyrus in PIL. The programs contain expressions that may span multiple DMs. A PIL program is submitted to the Papyrus optimizer [1] which transforms it into an equivalent but more efficient PIL program. The final (optimal) PIL program is translated to the Papyrus Execution Language (PEL). PEL [8] has the annotations necessary to execute a program. The Execution Engine [8] interprets the PEL program. It handles the control flow features of PIL such as conditionals and loops, as well as the data passing between methods.

The demonstration shows, through a series of examples, how a comprehensive cost-based query optimization strategy is essential to achieving high performance. Other aspects of the project that are not presented in this demonstration are parallelization [4, 8], and transaction management [7].

References

[1] S. Chaudhuri, R. Krishnamurthy, S. Potamianos, K. Shim. *Query Optimization Strategies in Papyrus.* Technical Memo HPL-DTD-93-1, Jan 1993.

[2] S. Chaudhuri, M.-A. Neimat, S. Potamianos, K. Shim. *FFRL: A registration language for Foreign Functions to enable query optimization.* In preparation.

[3] T. Connors, W. Hasan, C. Kolovson, M.-A. Neimat, D. Schneider, K. Wilkinson. The Papyrus Integrated Data Server. *1st Int'l Conf. on Parallel and Distributed Systems*, Miami Beach, Florida, Dec 1991.

[4] S. Ganguly, W. Hasan, R. Krishnamurthy. Query Optimization for Parallel Execution. *Proc. of the ACM SIGMOD Conf.*, San Diego, California, June 1992.

[5] W. Hasan and R. Krishnamurthy. PIL: An Optimizable Functional Language for Data Intensive Applications. *Fifth Int'l Workshop on Persistent Object Systems Design, Implementation, and Use*, Pisa, Italy, Oct 1992.

[6] C. Kolovson, M.-A. Neimat, and S. Potamianos. Interoperability of Spatial and Attribute Data Managers: A Case Study. *Third Int'l Symposium on Large Spatial Databases*, Singapore, June 1993.

[7] R. Rastogi and M.-A. Neimat. *On Using the Encina Toolkit to Provide Recovery Support for Data Managers.* Technical Memo HPL-DTD-92-21, Nov 1992.

[8] D. Schneider and T. Connors. Managing Query Execution for an Advanced Database Programming Language. *2nd Int'l. Conf. on Parallel and Distributed Information Systems*, San Diego, Jan 1993.