An EJB/Corba architecture for data intensive scientific computations

M. Eleftheriou[1], *M. Sidani*[1]

TJ Watson Research Center, IBM, USA

Abstract

While the efficient management of large data has made big strides in business applications, the management of data in scientific computations has lagged behind. We report on our research applying common middleware solutions in scientific computing applications. In our aproach, Enterprise JavaBeans are used for metadata, thereby combining the benefits of object persistence and relational databases. CORBA provides the link to legacy codes and the infrastructure for distributed computing.

Keywords: EJB, CORBA, Middleware

1 Introduction

Software components are reusable building blocks for constructing software systems [1]. Their emergence in business applications resulted from the need to simplify overall software development through: promoting the reuse of tested and trusted code; allowing interoperability of software developed by many different groups of developers; allowing different roles for differently skilled programmers, whereby the most experienced programmers can focus on components development and other programmers focus on program assembly using these components. Various component architectures exist today – component libraries and systems written to a certain specification. Of these, the Enterprise JavaBeans (EJB) specification [2] seemed particularly attractive for scientific computing applications. Indeed, the EJB specification provides a rich infrastructure of services, including automatic object persistence. It also provides a convenient way to interface with relational databases. To allow existing programs to call components written to the EJB specification we use CORBA [3] – a standard for object-oriented distributed computing. It supports the interoperability of different languages, and components running of different platforms.

2 The EJB/CORBA solution

A software component written to the EJB specification will heretofore be referred to as an EJB. What follows is a brief description of the architecture. This is work in progress.

2.1 EJBs

In one incarnation, EJBs can be used to front databases which, for example, contain metadata. Typically an EJB would represent a row in a table from the database. Updates to the database following modification of the EJB can be left to be done automatically by the supporting infrastructure with full transactional control. Alternatively, EJBs containing metadata can be saved for later retrieval using the persistent storage service in the EJB infrastructure. The type of support desired of the infrastructure is specified at “deployment” time – when the specific EJB is added

to a listening process, the EJB server. In our architecture, an EJB object represents metadata and have methods allowing operations to be performed on the data it references. The metadata to be represented depends on the specific application. Scientific computing applications typically involve large amounts of unstructured data stored in flat files. The choice of the metadata is then determined in such a way that common searches on data sets could be replaced with more efficient queries on the database or searches through EJB objects already in memory, followed by the retrieval of the data from the files. We mention here a particularly convenient construct available in the DB2 database program [4]: the DATALINK type. Entries of type DATALINK could be used to reference the data files. This has the advantage of providing transactional control over the access to the files, which we could also use to enforce consistency between data and metadata, by channeling all file updates through the database server.

2.2 Link to legacy programs

Legacy scientific codes are mostly written in FORTRAN and the applications in weather modeling with which we are experimenting are no exception. In our desing, the services of EJBs are made available to these codes using intermediary CORBA objects written in C. We believe that this solution provides a minimum length migration path for legacy codes into a scalable data management solution. Indeed, a client can interact with EJBs running on any of a number of servers – hence the scalabitlity. This solution also makes the data readily available over the web for rendering or downloading. Performance related questions are still open at this stage.

3 Future work

As we mentioned in the previous section, our current application is in the area of weather modeling. A general framework for implementing this solution in different application domains will result from our work and we see HEP as a potentially suitable domain.

References

1 D. Krieger, R.M. Adler, “The emergence of Distributed Component Platforms”, Computer Magazine, March 1998.

2 A. Thomas, “Enterprise JavaBeans Technology”, http://java.sun.com/products/ejb/white_paper.html, Dec. 1998.

3 Object Management Group (OMG), “CORBA/IIOP Specification”, http://www.omg.org/corba/corbaiiop.html.

4 http://www.software.ibm.com.