ORB then forwards my request to your software and returns any results, handling
the language mapping at both ends.

Services commonly required by many objects-lifecycle services, persistence
services, query services, and others are the subjects of standardization
specifications as well.

Is this environment appropriate for high-performance physics applications? If
the physics community ignores these approaches, does it do so at its own peril?

Among the questions that must be addressed are these:

- Is the Interface Definition Language rich enough to capture the interfaces
required by data-intensive physics applications?
- Is the performance penalty of brokered interactions inherently too great?
- Can we use an ORB simply to connect our applications, and then get it out
of the way?
- If the ORB does get out of the way, do we lose language-independence, and
are we back to home-grown low-level interfaces?
- What is the appropriate level of granularity for brokered interactions?
- The potential location transparency provided by an ORB is appealing, but
will performance considerations require that I provide a "smart proxy" to run
on your machine when you invoke software on my machine, in order to sustain
brokered interactions at a reasonable cost?
- If so, is proxy support a nightmare for providers of general-use software, or
can proxy generation be standardized or automated?
- What are the implications of proposed persistence services specifications in
this environment?

We explore these and other issues in a case study, in which we use
commercially available request brokers in an examination of a variety of potential
implementations of a statistical computation on physics data extracted from a
persistent data store.

---

RD45 - Object persistency for HEP - Status report
JAMIE SHIERS (CERN)

RD45 is a CERN project to investigate the issues concerning the storage and
manipulation of a persistent objects for LHC era HEP experiments. Objects
are typically created by a given process and cease to exist when that process
terminates. Such objects are called Transient objects. Persistent objects, on the
other hand, are those which continue to exist upon termination of their creating
process.

There are a number of existing efforts committed to the exploration of Object
Oriented (OO) techniques in a wide variety of key areas for HEP computing in
the LHC era. All of these projects will need to handle persistent objects but none
of them are addressing this question directly.

The goal of the RD45 project is to address the requirements of these projects
and of the LHC experiments themselves in terms of object persistency.

An important theme of the project is the use of standards. We start by
identifying the standards involved, we explain their interaction and examine
their suitability for HEP computing environments. We then describe the various
prototyping activities that we have undertaken and present the results that have
been obtained so far. Finally, we examine future directions and discuss the impact
of the proposed technologies on HEP computing in general.

---

Experience using a distributed Object Oriented Database (OODB) for a DAQ system
B. JONES (CERN)

To configure the RD13 data acquisition system, we need many parameters
which describe the various hardware and software components. Such information
has been defined using an entity-relation model and stored in a commercial
memory-resident database. During the last year, Itasca, an OODB, was chosen
as a replacement database system. We have ported the existing databases (hw
and sw configurations, run parameters etc) to Itasca and integrated it with the
run control system. We believe that it is possible to use an OODB in real-time
environments such as DAQ systems. In this paper, we present our experience
and impression: why we wanted to change from an entity-relational approach,
some useful features of Itasca, the issues we meet during this project including
integration of the database into an existing distributed environment and factors
which influence performance.