Why Object Technology Makes Business Sense

by Barry Herring

In the information technology community today, there is a growing acceptance that Object Orientation will be required to produce the complex software applications of the future. All the major programming languages and development environments are evolving OO characteristics. This paper will discuss the economic implications of this software evolution.

What is Object Orientation?

Object orientation is based on the idea that computer systems should allow a more natural representation of real-world entities from the task domain. To do this, OO software systems use discrete building blocks, called objects, that contain related code and data.

These objects are each "black boxes" that hide their internal organization, but which perform operations when stimulated by messages sent by outside agents, usually other objects. Because they include both code and data, objects logically replace both the programs and the databases that comprise the procedural programming world.

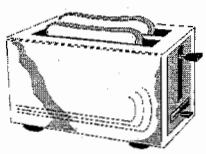

Fig. 1 Complexity Hiding

Imagine a household appliance, such as a stereo, a microwave oven, or a toaster (shown above). Although it may contain many complicated electronic components, these internal details are hidden from view—to use it, all you need to do is press the buttons on the front panel. (This is more convenient, and much safer, than having to reach inside your toaster to connect wires together.) Objects can be thought of as little information appliances that are more useful when they hide their internal complexities.

Technical Advantages of OO

The most important feature of objects is that they can more closely model real-world entities that require an automated representation. Objects keep their code and data to themselves through a principle called *encapsulation*. Objects are grouped into hierarchical categories called *classes*, and similar classes of objects can share features through a process known as *inheritance*. This results in a greater degree of *reusability*, reducing the amount of coding required.

Economic Advantages of OO

By separating computing tasks into discrete sub-tasks that accurately represent the task domain, OO programmers can successfully create systems whose complexity would have overwhelmed the capacity of procedural programmers. For highly complex computing tasks, and for lesser ones, the benefits of OO are also seen as lower life-cycle costs resulting from:

- simplified design because OO accurately represents familiar real-world entities;
- fewer defects because encapsulation reduces unintentional interactions;
- diminished total code because inheritance permits one class to reuse code defined by another;
- faster deployment resulting from the above three reasons;
- easier rework for new features because there are fewer hidden interdependencies.

Cost Savings Experience of OO Application Developers

Most software is delivered late and over budget. The OO strategy of *information hiding* makes reuse easier by simplifying interactions between objects.

Quantitative data on the efficacy of OO is hard to come by, but industry consensus leads one to believe that the three biggest benefits of OT are as follows:

1. that application-level objects can be reused,
2. that discrete components can be developed separately, and
3. that business rules can be re-used.

Sentry Research Service's *1996-97 Application Development Tools Market Report* states that a majority of respondents believe these claims are true.

In addition, David A. Taylor in *Object Oriented Technology: A Manager's Guide* states that faster development requires fewer resources. Buying off-the-shelf components is generally cheaper than making them, and reducing maintenance work also reduces cost: "The programming effort is lessened because you can assemble new programs out of existing components. Rapid prototyping reduces the design and administrative aspects of software development, which often account for as much as eighty percent of the total effort. These reduced resource demands translate directly into cost savings."

Implications of OO for M Technology

M Technology has had great success in producing complex software applications, but in the future its attractiveness to developers will certainly be eroded by competition from OO-proficient alternative systems. The advantages of M Technology are greatly enhanced when they are combined with OO, enabling it to be even more successful at doing what it does best. There are OO proposals before the MUMPS Development Committee (MDC) that could address this need. Readers interested in encouraging OO adoption into M Technology should contact the MDC chairperson.

M

Barry Herring is VP of Business Development for ESI Technology Corporation.