Overview 2

2.1 Introduction

This chapter uses typical problems and their OMG solutions to describe what the OMG's Object Management Architecture tries to achieve. The examples in this chapter should put the rest of the guide in perspective and help the user to appreciate the proposed technology, as outlined in the following chapters.

This chapter makes several assumptions: first, that the OMG's Object Management Architecture is mature; second, that conforming platforms and applications are abundant; and third, that the examples, taken from the mechanical CAD world, apply to other application areas as well.

2.2 The Advantages of OMA for System Vendors

Situation

A supplier of CAD workstations has, on the basis of hardware and a CAD-M application, a good position in the CAD-M market.

Problem 1: A large customer requires that the manuals related to his design be created on another supplier's publishing workstations, directly accessing the design information.

Solution: With the OMG's Object Management Architecture in place, the CAD-M design objects are stored on a database server. These objects are accessed by CAD application functions (methods) for drawing, parts explosion, and others. The new documentation requirements can be implemented by connecting the publishing equipment to the CAD network. A multimedia editor runs on the publishing workstations and addresses the CAD-M design objects to generate the drawings to be included in the documentation. Some editing is of course required: an illustration in a

manual usually takes a different format than a design drawing. The design objects may require a method to generate a representation in a format compatible with the publishing software, since there are many standards in the CAD world.

Equipment as well as software from different sources can interoperate. Substantial functional extensions can be made by applying a minimum of glue between nonrelated but existing applications. Software is reusable and extensible.

Problem 2: A major competitor expands its product with support for circuit board design. To be competitive, the system vendor must port software supplied by a leading OEM.

Solution: The solution of the second problem is even simpler: if the system vendor's hardware and systems software is indeed a platform supported by the Object Management Architecture and the circuit board design application conforms to the Object Management Architecture, the porting of the circuit board design application is straightforward and the system vendor as well as the OEM software supplier can enjoy a productive OEM contract. Applications written in conformance with the Object Management Architecture are portable. Translating these technical opportunities in business terms, it means that the system vendor can profit from the OMG's Object Management Architecture due to:

- Reduced initial cost to make new functionality available (alternatively, more functionality offered to the customers at the same cost) since existing software (from the System Vendor or from OEM sources) can easily be integrated.
- Early availability of a larger number of functions, leading to an extension of the market size.

2.3 The Advantages of OMA for Independent Software Vendors

Situation

An independent software vendor (ISV) has a position in stress analysis of mechanical parts.

Problem: The ISV wants to extend the functionality with software for the simulation of dynamic behavior of mechanical designs. The target is a set of existing CAD-M platforms.

Solution: Object Management Architecture is not magic: complex applications (like simulation of dynamic behavior of mechanical designs) require hard work. However, OMG's technology allows the ISV in this example to concentrate on the essentials: the complexity of simulation.

The Object Management Architecture comes with standardized Object Services (for application controlled management of objects) and a set of Common Facilities, in this case, helping to visualize the results of the simulation.

When used in a CAD-M environment that provides methods for parts explosion and retrieval of design data, OMG technology results in extended end-user functionality while conserving the existing design environment.

Since the simulation of dynamic behavior may involve heavy number crunching, calculations can be delegated to a dedicated server, providing the necessary megaflops, and leaving the CAD workstation resources free for interactive design work.

The Object Management Architecture improves productivity by reusing existing parts (and thus focusing on essentials). Object-oriented software design using the standard components of the Object Management Architecture imposes a design philosophy that leads to less iteration during the design phase. Object Management Architecture provides transparency over heterogeneous networks, allowing specific tasks to be delegated to specific machines.

2.4 The Advantages of OMA for End Users

Situation

A central CAD department of a pump factory supplying a large variety of pumping equipment.

Problem: The management of the central CAD department of the pump factory gets the following tasks:

- Fine-tune the design of a series of high-precision pumps. A CAD-M system is available but fine-tuning requires stress analysis and simulation of dynamic behavior of the moving parts in the pump's design.

- Refine reporting of design costs to reflect the cost per product line. The accounting department is equipped with PC workstations connected to the factory's LAN. Reporting from accounting is based on regular word processing and spreadsheet applications.

- Propose, in cooperation with the documentation department, an improved procedure for the production of manufacturing and service documentation. The documentation department uses desktop publishing equipment, connected to the factory's LAN. The supplier of the CAD equipment has a stress analysis package in its catalogue and is able to install that package on the current equipment. (Refer to 1.) The same supplier has recently announced the availability of an extension allowing simulation of dynamic behavior of mechanical designs. A contact with the ISV supplying that software indicates that the necessary vibration analysis can indeed be done but will require more number-crunching capacity than is currently available.

Solution: The pump factory's computer consultant indicates how the classes of the CAD-M package can be extended with an automatic time-stamp facility. Each week, the collection of time stamps is put into a spreadsheet format and is sent to Accounting. The implementation work needed is assigned to a software house specializing in Object Management Architecture–conforming software.

Another extension to the CAD-M classes is an option to extract drawings from the CAD-M object database in a format required by the desktop publishing software. The documentation department gets (read-only) access to released design objects in the database of the CAD department.

For the end user, standardized interfaces for object-oriented application software provide the option of extensibility of existing software. In addition, through encapsulation of non-conforming applications (like the spreadsheet package of Accounting), the transition to the new technology can be a gradual one. The productivity of an organization can be improved through exploitation of the interoperability of conforming software over heterogeneous networks. In addition, the object-oriented approach to application software guarantees an intuitive user interface: computer-simulated objects correspond with real world objects, whether they are the blades of a pump, a pump's service manuals, or weekly reports on the time spent on a specific design. Standardization of object-oriented technology creates uniformity and consistency over different and independently developed applications.

2.5 Object Management Architecture

The Object Request Broker component of the Object Management Architecture is the communications heart of the standard. This is referred to commercially as CORBA (Common Object Request Broker Architecture). It provides an infrastructure allowing objects to communicate, independent of the specific platforms and techniques used to implement the addressed objects. The Object Request Broker component will guarantee portability and interoperability of objects over a network of heterogeneous system. Specifications for the Common Object Request Broker are contained in *CORBA: Common Object Request Broker Architecture and Specification.*

- The Object Services component standardizes the life cycle management of objects. Functions are provided to create objects (the Object Factory), to control access to objects, to keep track of relocated objects and to consistently maintain the relationship between groups of objects. The Object Service components provide the generic environment in which single objects can perform their tasks. Standardization of Object Services leads to consistency over different applications and improved productivity for the developer. Specifications for the Object Services that have been adopted as standards by the OMG are contained in *CORBAservices: Common Object Services Specifications.*

- The Common Facilities component provides a set of generic application functions that can be configured to the requirements of a specific configuration. Examples are printing facilities, database facilities, and electronic mail facilities.

Standardization leads to uniformity in generic operations and to options for end users to configure their configurations (as opposed to configuring individual applications).

- The Application Objects part of the architecture represents those application objects performing specific tasks for users. One application is typically built from a large number of basic object classes, partly specific for the application, partly from the set of Common Facilities. New classes of application objects can be built by modification of existing classes through generalization or specialization of existing classes (inheritance) as provided by Object Services. The multi-object class approach to application development leads to improved productivity for the developer and to options for end users to combine and configure their applications.

- Domain Interfaces are domain-specific interfaces for application domains such as Finance, Healthcare, Manufacturing, Telecom, Electronic Commerce, and Transportation.