16

Integration of industrial applications: The CCE-CNMA approach

P. Pleinevaux
Computer Engineering Dept.
Swiss Federal Institute of Technology, Lausanne
EPFL-DI-LIT
CH-1015 Lausanne, Switzerland
pleinevaux@di.epfl.ch

Abstract

Integration of industrial applications has been and still is a major problem for industrial enterprises. Development of new manufacturing applications and integration of existing ones is hampered by a number of problems, such as heterogeneity of hardware and operating systems, complexity and non harmonisation of application programming interfaces, diversity of communication protocols, user interfaces and databases involved in the applications.

The ESPRIT CCE-CNMA has specified, implemented and validated an open and portable platform for integration of industrial applications. This platform, called CCE (CIME Computing Environment), hides the diversity in communication protocols, databases and access methods to the user. It provides high level interfaces that allow the user to concentrate on his application domain and not on the way to program his application.

In this paper, we review the problems faced during integration, describe the CCE architecture and discuss the relationship between CCE and the CIMOSA integrating infrastructure.

Keywords

Application integration, CIMOSA, CNMA, MAP, CCE.

1 INTRODUCTION

The ESPRIT CCE-CNMA project introduced in 1993 an open platform for the integration of industrial applications. Called CCE (CIME Computing Environment), this platform is based on a standard communication infrastructure - CNMA (Communications Network for

P. Ladet et al. (eds.), *Integrated Manufacturing Systems Engineering*

Manufacturing Applications) - and offers the appropriate services, tools and administration to develop and execute applications in the manufacturing and process control environments.

The purpose of this paper is to introduce this platform, presenting its architecture, its properties, its relationship with the CIMOSA integrating infrastructure and indicating the current status of the project.

Proprietary platforms like Digital's BASEstar or IBM's PFS/DAE exist on the market. These platforms however are not open in the sense that their interfaces are proprietary and that they are not available from different suppliers. CCE on the contrary is an open platform, available from three different vendors on a number of machines and operating systems.

The paper is organized as follows: Section 2 describes the problems found by system integrators when developing CIME applications. Section 3 describes the CCE architecture while Section 4 presents its main properties. Section 5 discusses the relationship between CCE and the CIMOSA integrating infrastructure. We conclude with a presentation of the problems that must be dealt with in future versions of CCE (Section 6).

2 PROBLEMS AND USER REQUIREMENTS

CIMOSA distinguishes three levels of integration provided by CIM (Computer Integrated Manufacturing): business integration, application integration and physical integration (AMICE, 1992). For six years, in the period 1986-1992, the ESPRIT CNMA project worked on a communication architecture that is well adapted to the physical integration of industrial applications. This architecture, compatible with the Manufacturing Automation Protocol (General Motors, 1988), provides services for communication with manufacturing devices, for the transfer of files, for access to remote databases and for the administration of the communication infrastructure. This solved one of the most costly problems of industrial companies, namely the interconnection of heterogeneous equipment provided by different manufacturers.

Yet, other problems remained. A study made by the CCE-CNMA project in 1993 showed that the following problems were encountered by all users when attempting to integrate their applications:

- Existence of many application programming interfaces (APIs): a manufacturing application commonly accesses databases, uses one or more industrial messaging protocols and displays text on man-machine interfaces. Very different APIs are found for these tasks with different mechanisms to deal with the same problems such as memory management, error handling or event management.
- Complex application programming interfaces that require the initialisation of many parameters, some of which are never used.

- Non harmonized application programming interfaces: the same problems are solved in different ways by the different APIs, leading to increased costs for programmer training.
- Lack of tools for the development and debugging of new applications.
- Lack of mechanisms to ensure the consistency of data stored in the system, especially when these data are distributed over multiple machines of different suppliers.

An independent study presented in (Deregibus *et al.*, 1991) discusses a list of user requirements that is very close to those identified by the project.

3 THE CCE ARCHITECTURE

CCE (CIME Computing Environment) is an *open* platform for the integration of industrial applications (CCE-CNMA, 1994 and 1995a). It is designed to reduce the above mentioned problems. Its main properties are described in Section 4.

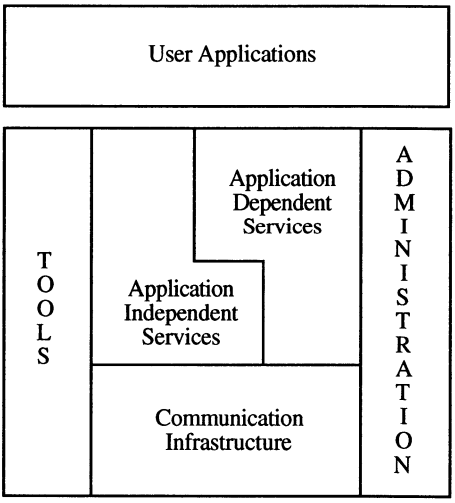

Figure1 The CCE architecture.

The CCE architecture (Figure 1) distinguishes the following components:

- a communication infrastructure
- application independent services
- application dependent services

- tools
- an administration

In the following sections, we briefly describe each of these components.

3.1. Communication infrastructure

At the lowest level of the architecture lies the communication infrastructure. The standard adopted by CCE is the CNMA architecture (CCE-CNMA, 1993), compatible with the Manufacturing Automation Protocol (General Motors, 1988). The services offered to the user by this communication profile are the following:

- MMS (Manufacturing Message Specification) for remote control and monitoring of industrial devices. With MMS, users can download and upload programs or data, start and stop programs, read or write variables (CCE-CNMA, 1995b).
- RDA (Remote Database Access) for access to relational databases using SQL. The protocol is independent of the database used, for example ORACLE, INGRES or INFORMIX.
- FTAM (File Transfer Access and Management) for the transfer and remote manipulation of files.
- X.500 Directory Service, to obtain information on users, applications or objects accessible in the network.

While CNMA is the recommended communication infrastructure, CCE is actually able to run on a variety of communication protocols, including the Internet Protocol Suite (TCP/IP) and proprietary protocols. CCE hides the protocols used by applications to communicate with one another.

3.2. Application independent services

A first layer above the communication infrastructure offers application independent services. At the upper interface of this layer, the user views a uniform environment consisting of CCE objects accessible through operations and able to send notifications when interesting events occur. The environment is composed of application independent objects such as variables, programs or domains.

One of the application independent services offered by CCE is transaction management, a service that offers the mechanisms needed to ensure consistency of objects stored in CCE. In this area, the project adopted the X/Open Transaction Demarcation Interface as the standard interface. Transaction management is available on CCE platforms that are based on an OLTP (On-Line Transaction Processing) execution environment (Gray and Reuter, 1993).

3.3. Application-dependent services

While application independent services can always be used to develop an application, there is often a need in industrial applications for higher level abstractions. Indeed, when designing its application, the user does not reason in terms of MMS variables, domains or programs but in terms of real objects such as pallets, tools, parts, etc. The application dependent interfaces offer services to manipulate objects of this kind with object-specific operations. CCE-CNMA defined such interfaces for mould management, electronic kanban and tool management. Additional interfaces have been considered for pallet and warehouse management.

As an example, let us consider tool management. In a manufacturing cell, numerical controllers of machine tools and machining centres have to be supplied with data that describe the dimensions and age of the tools used to manufacture parts. Depending on the manufacturer of the machines, these data are stored in very different ways: some data are stored in MMS structured variables, others in MMS domains. In CCE, the tool manager hides to the user the differences in data representations. The user can access the following services, among others, to manipulate tools:

- `CreateTool`: to create an object in CCE that stores data on a given tool.
- `DownloadToolData`: to transfer the tool data associated to a program and a given machine from CCE - where it is stored in a database - to the given manufacturing device. The user does not need to read the database, this work is done for him by CCE.
- `UploadToolData`: to transfer back the data once the machine is done with them. The data are updated in the database without user intervention.
- `GetToolData`: to read the current values of the tool data. CCE fetches the data where they are most up-to-date: in the CCE database or in the manufacturing device.

Application-dependent interfaces are one of the most important features of CCE that make it different from classical platforms like the OSF DCE (Distributed Computing Environment), Comandos (Cahill *et al.*, 1993) or ANSAware (Herbert, 1994). It is important to note that these platforms provide the means for the user to build distributed applications from scratch, and in this sense they can be said to be general-purpose. But a tremendous work must be done on these platforms to provide the user with the right abstractions (objects) that are used by industrial applications. In CCE, these abstractions are built-into the application independent and dependent interfaces, thus relieving the user from developing these abstractions and allowing him to concentrate on his application and not on meta problems.

3.4. Tools

A series of tools are available for the development of applications as well as for the configuration of the platform. In the following, we describe three of them.

The CCE Configuration Tool is an application designed by Alcatel-TITN-Answare that allows an administrator to create CCE objects, to monitor the operation of a running system and to modify the parameters of the platform or objects. Based on a graphical interface, it allows an easy access to the information needed by an administrator.

The SQL preprocessor allows the user to read or write data using the well-known SQL database manipulation language (Pleinevaux, 1992). The user views the system as a relational database and no longer as an object-based system. This enables the user to access CCE from offices and to reuse existing SQL applications. Object-oriented extensions based on the SQL 3 and ODMG-93 standards (Cattell, 1993) are being considered for CCE (Näf and Pleinevaux, 1995).

The Variable Generator allows a user to define simple C data structures and to automatically generate the data structures used by the application interfaces, for example MMSI, the standard interface to the MMS protocol. When the communication partner is a Programmable Logic Controller (PLC), the tool generates the corresponding PLC data structures as well.

4 CCE PROPERTIES

When using application dependent or independent interfaces, the user sees ASPI objects but does not know where these objects are stored, in which form and which protocol is used to access them. The platform has the following general properties:

- **CCE hides the nature of accessed data**: When accessing a CCE object, the user does not know whether this is an MMS variable stored in an industrial device, a database entry or an application variable.
- **CCE hides data location**: The place where the object is stored is unknown to the user. This may be in a manufacturing device, a database or a workstation.
- **CCE hides the communication protocols**: The protocol used to access the CCE object is hidden by the interface: it may be a standard protocol like MMS, a proprietary protocol like UNITE, an RPC protocol, the user does not know. This fundamental feature allows for example the migration from proprietary protocols to standard protocols without the user having to rewrite his application programs.
- **The platform is portable**: CCE has been designed and implemented in such a way that it can be easily and quickly ported on different operating systems. This is one of the major user requirements.
- **Object-orientation**: The functionality offered at the two application interfaces is presented using the object approach. The user accesses CCE objects through operations

and may receive notifications when significant events take place in the object. Objects belong to types that inherit attributes, operations and notifications from parent types. This style of interface allows a simple mapping after the analysis and design phases of the application.

- **The user can access all interfaces**: Three levels of interfaces are accessible to the user: the communication interfaces (MMS, RDA, FTAM), the application independent interfaces and the application dependent interfaces.
- **Integration of PCs under Windows 3.x**: The ASPI interface is available on Windows PC but in a DDE flavour. This allows a PC to access objects stored on other machines (UNIX, DCE, OLTP) with the same style of interface. The DDE style of interface enables the user to access CCE from any Microsoft application package supporting DDE, for example Access, Excel, etc.

5 CCE AS CIMOSA INTEGRATING INFRASTRUCTURE

The CIMOSA integrating infrastructure is a platform for the interpretation and execution of the different models that describe an enterprise. The requirements identified by CIMOSA (AMICE, 1992; Querenet, 1991) that must be met by a platform used as integration infrastructure are:

- interpret the behavior and execute the models produced by the CIMOSA methodology
- provide services common to all CIM systems
- manage resource availability
- manage data related to monitoring and control of resources
- ensure proper communication
- manage location, failure, access and performance transparencies
- handle heterogeneity of manufacturing devices
- use standards for the communication subsystem.

Conceptually, the integrating infrastructure offers four groups of services:

- *Business services* provide the functions necessary to control the execution of the models associated with the function and resource views.
- *Front-end services* provide the means to communicate with resources, either machines, humans or applications.
- *Information services* provide access in a unified way to all information of an enterprise. These services ensure consistency, integrity, and protection of an enterprise information.
- *Communication services* allow cooperation of the above services.

The service groups are themselves composed of integrating infrastructure services. For example, the front-end services are divided into machine front-end for integration of

manufacturing devices, application front-end services for example to communicate with CAD/CAM and human front-end services for communication with operators.

5.1. Relationship between CIMOSA and CCE

CCE satisfies a large number of the requirements that are specified by CIMOSA for a platform used as integrating infrastructure:

- CCE offers application independent services for access to variables, transfer of bulk data, remote control of programs and event management.
- CCE offers adequate communication services for CIM systems, namely access to manufacturing devices and databases, file transfer, access to a name service.
- CCE is based on standard communication protocols defined by the International Standards Organisation (ISO).
- CCE offers access to the Manufacturing Message Specification (MMS) for communication with heterogeneous manufacturing devices. CCE integrates proprietary communication protocols as well, for example Unitelway, Modbus or SINEC.
- CCE provides location, access and performance transparencies.

As will be explained below, CCE offers services that comply with the above classification of CIMOSA integrating infrastructure services. CIMOSA considers two classes of information services: (1) system-wide data provides a unified access to data without concern for the location and structure of the data; (2) data management which provides services to store and retrieve in a unified way the data stored in the system.

In CCE, access to data is possible in three different ways: the CNMA application protocols, the ASPI interface and the SQL interface. When an application requires the use of a single application protocol to communicate with a remote application, the user can in this case directly call the application interface of this communication protocol.

When an application makes use of multiple application protocols to simultaneously access data in different stores, the user can still use the above approach or call the CCE Application Service Programming Interface (ASPI). This interface gives access to CCE objects that can be mapped on objects of all CNMA application protocols, in particular MMS objects, database entries and application specific objects. The main advantage of this approach is that the user does not know where the data are stored and which access method is used to read or write them. The ASPI offers in particular a filter mechanism which allows to access a group of objects having similar names.

For access to large collections of data, CCE offers the SQL interface (Pleinevaux, 1992). With SQL, the user is able to read or write data about objects stored either as MMS objects, Network Management objects, database entries or CCE objects. All these objects are viewed as tuples of relations on which the classical relational operations can be

performed. An SQL pre-processor translates programs with embedded SQL statements into programs containing CCE interface calls.

CIMOSA consider two groups of communication services: system-wide exchange handles all intra-node data exchanges and forwards inter-node communications to communications management; the latter provides access to protocols such as the OSI protocols. CCE satisfies the requirements identified for the CIMOSA communications services. For intra-node communications, CCE offers the mechanisms of the underlying execution environment, namely DDE (Dynamic Data Exchange) for Windows or the DCE Remote Procedure Calls (RPCs) for the OSF DCE environment. For inter-node communications, CCE gives access to the CNMA communication architecture, the Internet Protocol Suite (with TCP/IP) or any proprietary protocol used by the application.

As can be seen from the above discussion, CCE covers the requirements of communication, front-end and information services. The only area in which significant work remains to be done is the business services, which are a distinctive aspect of CIMOSA. The idea is to allow execution of the models derived from the CIMOSA views. One of the benefits of this idea is that an evolution of the enterprise results in an update of the models which are themselves updated for use by the integrating infrastructure. In CCE, application knowledge is still mainly provided to the system in the form of programs, not in executable models. The object approach however, allows to store data and knowledge in entities - the objects - that can be dynamically created, deleted, updated or moved in the system. Addition of new object types and instances can also be made at any time while the system is up and running.

6 STATUS OF THE PROJECT

The project started in January 1993 and ended in May 1995. CCE has been specified, implemented and tested in 1993-1994. A validation and evaluation phase started in the summer of 1994 with four pilot installations: Aerospatiale, EFACEC, Magneti Marelli and Mercedes-Benz. The purpose of these pilots is to assess the validity of the design in real industrial environments, from car manufacturing to production of electrical transformers.

CCE is currently available on Lynx, AIX, SCO UNIX and Windows 3.1. On Windows, CCE gives access to MMS and the application independent interfaces through DDE (Dynamic Data Exchange), a protocol defined by Microsoft for communication between Windows applications. This approach allows the reuse of a large base of existing software packages available on Windows, for example In Touch or Excel.

CCE however is not completed. A good basis is available but complementary work needs to be done to address, among other issues, the following ones:

Evolution towards CORBA and OLE:
The heart of CCE is an entity that enables users to invoke operations on CCE objects. Two standards are emerging in this area. CORBA (OMG, 1991) from the Object Management Group, is a specification for an Object Request Broker (ORB), precisely this type of entity. The CORBA standard has been widely accepted by major vendors in the computer industry, the second version addressing the problem of ORB interoperability. OLE (Object Linking and Embedding) is the Microsoft standard for communication among Windows applications. OLE 2.0 replaces the DDE mechanism mentioned above by an RPC (Remote Procedure Call) mechanism. CCE will have to follow these evolutions, in both areas to allow for interoperability of CCE with office applications.

Definition by the user of new object types:
In the current version of CCE, the platform is delivered to the user with a limited number of object types. To introduce new types in its application, the user must implement special software components that provide the semantics of these types. In the present state of CCE, this operation requires a good knowledge of system programming. In the future, it is planned to have a tool allowing the user to define his object type, compile it and integrate it in the platform.

Definition of standard object types:
This area is of fundamental importance to the user. If an agreement can be found among users on a common definition of widely used manufacturing objects, then these definitions can be implemented by vendors in the application dependent interfaces, thus relieving the user of this task. The MMS Companion Standards (ISO 9506-3 and 4) are a first attempt in this direction but the results are not satisfactory.

Fault-tolerant infrastructure:
While the current version of CCE supports some degree of fault tolerance with persistent storage of CCE objects, there is a clear need for mechanisms that allow for quick recovery from failures. In this area, CCE could draw upon the work performed by the ESPRIT Delta-4 project (Powell, 1991) which specified, implemented and validated a communication architecture, based as far as possible on international standards, that is close to the CNMA architecture.

7 CONCLUSION

The ESPRIT CCE-CNMA project defined, implemented and validated an open platform for the integration of CIME applications. This platform is based on the CNMA communication architecture but is able to deal with proprietary communication protocols. The platform offers application dependent and independent services that allow the rapid creation of new industrial applications. Data nature, data location and communication protocols are hidden to the applications which can thus be reused in different contexts. Tools for the platform allow for automatic code generation, compiling programs containing embedded SQL statements and for configuration of the system.

CCE differs from other general purpose platforms like the DCE, Comandos or CORBA by the fact that it integrates a number of communication protocols like MMS or RDA and is available with predefined application dependent and independent interfaces, specifically designed for industrial applications.

CCE can be regarded as a realization of the CIMOSA integrating infrastructure. The provision of simple high level programming interfaces allows the reduction of application development time. Work remains to be done in the areas of CIMOSA business services, fault tolerance, standardisation of classes of manufacturing objects and tools for the integration of new object types in the platform.

REFERENCES

AMICE (1992), ESPRIT Consortium AMICE, *CIMOSA: Open System Architecture for CIM*. Springer-Verlag.

Cahill, V., Balter, R. and Harris, N.R. (1993) *The Comandos Distributed Application Platform*. Springer-Verlag.

Cattell, R.G. (1993), *The Object Database Standard: ODMG-93*. Morgan Kaufmann, San Matteo, California.

CCE-CNMA (1993), ESPRIT Project 7096, CNMA Implementation Guide, Revision 6.0.

CCE-CNMA (1994), ESPRIT Project 7096, Introduction to the CIME Computing Environment - A Platform for the Creation and Execution of Industrial Applications. Document 7096.94.08/D2.PD available on the CNMA information server at http://litwww.epfl.ch/~ppvx/cc_CCE-CNMA.html.

CCE-CNMA (1995a), ESPRIT CCE-CNMA Consortium, *CCE: An Integration Platform for Distributed Manufacturing Applications*. Springer-Verlag.

CCE-CNMA (1995b), ESPRIT CCE-CNMA Consortium, *MMS: A Communication Language for Manufacturing*. Springer-Verlag.

Deregibus, F., Bobbio, M., and Rusina, F. (1991) Open Systems and Manufacturing Software Integration Platforms, in *Proc. of the 7th CIM-Europe Annual Conference*, Turin, Italy, 33-43, Springer-Verlag.

General Motors (1988) *Manufacturing Automation Protocol*, Version 3.0.

Gray, J.N. and Reuter, A. (1993) *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann, San Matteo, California.

Herbert, A. (1994) An ANSA Overview. *IEEE Network*. **8**(1), Jan./Feb. 1994, 18-23.

Näf, M. and Pleinevaux, P. (1995) On the use of SQL3 and ODMG to access CCE objects. Technical report 7096.95.03/F1.PD, available on the CNMA information server.

Object Management Group (1991) *The Common Object Request Broker: Architecture and Specification*.

Pleinevaux, P. (1992) An SQL Interface to CCE. EPFL-LIT Internal Report, December 1992.

Powell, D. (Editor) (1991) *Delta-4: A Generic Architecture for Dependable Distributed Computing*. Springer-Verlag.

Querenet, B. (1991)The CIMOSA integrating infrastructure. *Computing and Control Engineering Journal*, May 1991, 118-125.