Rapid Prototyping for MAP/MMS based CIM-OSA Environments

Milena M. Didic
Institute for Applied Informatics
Kernforschungszentrum Karlsruhe
PO Box 3640, D-7500 Karlsruhe, Germany
milena@iai.kf.k.de

Abstract—For the validation of the CIM Open System Architecture (CIM-OSA) a demonstrator called McCIM has been developed. First steps to link the CIM-OSA Modelling Framework and its Integrating Infrastructure (IIS) are described. Models of CIM applications are executed by the IIS to run a rapid prototype of on-line model-based flexible manufacturing. The Manufacturing Automation Protocol (MAP) and the related Manufacturing Message Specification (MMS) are used for the communication between network nodes and various components inside the network nodes.

I. INTRODUCTION

The set of functions in the execution control, resource management, integration of machines, integration of human work, integration of application programs, distributed information and communication handling is generic in all particular enterprises. There are various approaches for the enterprise integration proposed around the world. Most of them have proprietary solutions as basis. The only 'open' one is the Computer Integrated Manufacturing Open System Architecture (CIM-OSA) [KOS92].

CIM-OSA is composed of a consistent set of complementary reference models [VER92] capable of modelling the reference requirements in terms of functions, information, resources and organisation. For realising the necessary functionality, CIM-OSA defines the internal and external behaviour of the Integrating Infrastructure (IIS) providing a set of generic functions, supporting the integration of specific functions inherent in a particular system [QUE92].

To validate CIM-OSA concepts and rules by application in various industrial environments is the main goal in the ESPRIT II Project 5510 VOICE (Validating OSA in Industrial CIM Environments). The work gives the opportunity to validate CIM-OSA in various real industrial environments. Significant benefits consist in the real and diversified industrial applications, first time in this scale and gives us also the opportunity to get a significant experience in a very important field.

The application will be provided by modelling of VOICE industrial users' applications and a set-up of CIM-OSA on test beds for a broad spectrum of manufacturing environments: car assembly, machine-tool fabrication and aluminium rolling and milling. From the architecture models the functionality of both, the IIS services and application programs, which are necessary to build CIM applications in test beds, will be derived.

On the basis of the CIM-OSA Formal Reference Base I [AMICE] the following Integrating Infrastructure Services have been considered in VOICE:

- Business Service (BS) to control enterprise operations
- Front-End Service (FS) for the integration of enterprise controllable resources (machines, humans, applications)
- Information Service (IS) for data access, data integration and data manipulation
- Communication Service (CS) providing the transparent synchronous and asynchronous communication required for the interactions of the Services BS, IS and FS.

In the scope of the validation we worked extensively with the ESPRIT II Project 2422 AMICE. As main objectives it has been stated in [KL91b] that for a minimum demonstration of CIM-OSA functionality the following has to be shown:

1. The principle of the Business Service and in particular of its parts Business Process Control and Activity Control by executing the Functional Model of a Manufacturing Cell
2. The principle of Front-End Service converting so-called Functional Operations into device specific commands.
To fulfil the above goals the following components of a CIM-OSA demonstrator are required:

a. An Engineering Tool for creation of sequential control structure as CIM-OSA Function Model
b. An interpreter for the on-line execution of the sequential control structure of the model
c. A presentation tool to visualise flows of actions in the sequential control structure
d. An interface of the control interpreter to the communication network of the demonstrator
e. An application focusing on the shop floor control

Considering the above objectives and requirements the remainder of this paper is structured as follows. Chapter II explains the design trajectory of the first steps to a Rapid CIM-OSA Prototype. The methodology for the execution of CIM-OSA models by the IIS, as well as the novel structuring concepts are introduced in the chapter III. The main features of the Integrating Enterprise Engineering tool, which fulfils the requirements a, b, c and d are described, too. A shop floor control application (e) is described as a CIM-OSA Function Model of a Business Process in the chapter IV. The online model execution of the sequential control structure (req. b) is presented in chapter V. This includes the principles of the Business Process Control and the Activity Control (see objective 1). The dynamic behaviour of the Activity Control and its Interaction Control is illustrated by snapshots during the model execution. Finally, the configuration of the resulting demonstrator McCIM is presented in chapter VI. An interface between the model interpreter and the MAP/MMS based network in McCIM has been developed; see requirement (d). The CIM-OSA Front-End Service is fully implemented in McCIM (objective 2); its principle is briefly described in the chapter VI; a detailed description of the related concepts can be found in [WNH92].

The emphasis of this paper has been set on the methodology for the execution of models by the IIS, i.e. the principle of the Business Service, and in particular the fulfilment of requirements (a), (b) and (c).

II. FIRST STEPS TO A RAPID CIM-OSA PROTOTYPE

Integrating Enterprise Engineering

The needs for modelling a CIM enterprise and its associated CIM systems in a coherent way have originated a modelling approach based on a Reference Architecture from which Particular Architecture can be developed. In contrary to many other modelling tools CIM-OSA does not treat model building as a final task or just for simulation. The purpose of the models is rather to be executed and used as the management and control system of the enterprise [KL91a].

In developing models, a CIM user will start by translating his business and operational objectives into a hierarchical set of CIM system requirements. These requirements will be reconstructed during the design phase and various levels of design before finally deriving a definition of the related implementation. The results of the development process are expressed in the form of a CIM-OSA Particular Architecture. It is implemented according to the specification of the CIM-OSA Enterprise Engineering Environment residing in and controlling the Enterprise Operation Environment.

A given Particular Architecture might evolve over time following possible changes in the enterprise requirements. Still, at any time it is based on a set of reference constructs and supported by a design methodology, defined within CIM-OSA Reference Architecture. This methodology itself and the Particular Architecture are applied in and created with the aid of the CIM-OSA Enterprise Engineering Environment.

An Integrating Enterprise Engineering Tool (IEE) allows designer to identify a stable base for the evaluation and allows for changes during various phases of the design trajectory.

Rapid Prototyping Design Trajectory

From the outset the work on VOICE has put a strong emphasis on the major methodological guidelines: rapid prototyping strategy and the use of automated software engineering facilities. The use of the Formal Description Technique has been strongly recommended. Therefore, a specific VOICE task has been created: Tools, Supplements and Additions. This task is supposed to exist for the whole duration of VOICE. The related draft working plan on the necessary steps to be taken in creating a rapid prototype of IIS based test beds is presented in Figure II.1.

The first phase of the activity focuses on the identification of meta-tools suitable for the VOICE work as well as on the integration of concepts, techniques and methods into a coherent body of useful engineering guidelines for the reduction of the implementation time.

The design process is carried out in steps characterising the design trajectory (Fig. II.1). In the initial step on this trajectory the major VOICE industrial users' requirements have been analysed and partly formalised in an initial specification. This resulted in a representative set of scenarios for each user. The scenarios do not contain irrelevant details such as internal structure,
Fig. II.1: Necessary Steps to a CIM-OSA Rapid Prototype Channel / Instance Petri Net
since such details must be subject of design decisions at lower level of abstraction. Among properties of systems, one finds the different distribution transparency, e.g. access, location and configuration transparencies. As a rule, distribution transparency hides some temporarily irrelevant aspects of the system to be implemented.

Given the formal CIM-OSA framework, the VOICE results of the IIS analysis as well as a particular abstraction guided by the knowledge of VOICE users' requirements are used as inputs of the VOICE task force's work. A set of test bed scenarios has been produced.

In the next phase the abstract IIS specification has been refined into alternative, compatible models. During the stepwise refinement, the structure of the resulting system has been defined using hierarchical decomposition of models. The decomposition goes to the IIS services required and finally down to the detailed distribution of the hardware and software of the resulting system.

The formal description of IIS services and protocols is performed in parallel with the successive development of the architectural semantic of the test beds' applications. Using these models we can validate several concepts of IIS services and necessary extensions. Obviously, a good practice is to evaluate and to take a small amount of design decisions in each time, producing immediate more refined specification. During this design phase the validation of models provides the proofs of correctness of related IIS services and protocols.

In co-operation with ESPRIT II Project 2422 AMICE the products available on the market were evaluated and selected as candidates for the IIS of VOICE test beds. Since none of the candidates could be used for the Machine Front-End (MF) Service it has been developed. The relevant easy for use MAP/MMS modules (service units) [WNH92] were developed, too. For the demonstration of the integration, machines are simulated on a separate shop floor simulation node.

III. CONCEPTS FOR MODEL EXECUTION AND IEE TOOL

THE Integrated Enterprise Engineering (IEE) Tool links CIM-OSA Modelling Framework and its Integrating Infrastructure. CIM-OSA models are formally specified and executed on-line by the invocation of distributed IIS Services. A model based control for flexible manufacturing can be achieved in this way.

The hierarchical decomposition of CIM-OSA concepts on the Function View is followed strictly (Fig. III.1). At the Design Specification Level a Function Model consists of:

- a Domain Process as a structured set of Business Processes
- for each Business Process a set of Enterprise Activities (EAs) executed by the Business Process Control (BC)
- for each Enterprise Activity a set of technology independent Functional Operations (FOs) executed by the Activity Control (AC)
- for each Functional Operation a set of Functional Actions (FA) executed by the Interaction Control (IC)
- for each Functional Action a set of technology dependent Service Units executed by the Dialogue Control (DC).

On every hierarchy level of decomposition the consistent CIM-OSA standard graphical language for the model description is applied (as suggested in [MK91a]).

The Interaction Control (IC) is introduced first time in this paper. IC interacts with relevant Front-End Service within the Dialogue Control during a FO execution. The dialogue is executed according to the Standard Control Structure (SCS) model of an FO (Fig. III.2), described by Functional Actions (FA) like Start, Stop, Terminate, Restart and Initialise.
The functionality required on each particular layer is of the same nature. The Business Process Control, Activity Control, Interaction Control and Dialogue Control use on their respective layers (Fig. III.1):

- Interpretation Engine (IE) to interpret models of a layer; it understands CIM-OSA modelling constructs and interprets them.

- Event Handler (EH) to provide connections between different constructs, which are not represented by the model layout and sequence.

- Configuration Manager (CM) to manage configuration issues of the IIS environment. Here, e.g. occurrences of particular instances of Enterprise Activities, Functional Operations are created, deleted, etc.

- Resource Engine to schedule, dispatch and control resources.

To execute a Business Process, e.g. the Business Process Control schedules resources, the Activity Control dispatches a group of resources required for an Enterprise Activity and the Interaction Control assigns a single resource for the execution of a Functional Operation.

The Event Handler, Interpretation Engine and Configuration Manager are introduced here, they are neither covered by existing CIM-OSA concepts nor by its taxonomy. The hierarchical decomposition and the pipelined structure in Fig. III.2 show connections between the related items. In addition, the Resource Engine pipe-line provides for the link missed so far between the Function View and the Resource View in the CIM-OSA Formal Reference Base.

The described concepts permits a clear separation of concerns between basic elements and can fill in the huge gap between CIM-OSA Modelling Framework and IIS.

### Integrating Enterprise Engineering Tool

A high level Petri Nets Tool using Small Talk as programming language is applied here as a meta-tool [PACE]. Petri Nets are used to construct CIM-OSA models; they are generated by the use of the graphical editor. The Interpretation Engine is built on top of the Small Talk Interpreter. Following the CIM-OSA Framework its basic modelling constructs are implemented in the Engine and the graphical presentation of Petri Net models are modified respectively.

As an example the Fig. III.3 illustrates how CIM-OSA Procedural Rules (PRs) are mapped onto Petri Net components. PRs generally separate model constructs (see below). They are used to describe possible conditions or actions desired following an event or production of the related Ending Status.

The Event Handler, Configuration Manager and Resource Engine are built using Petri Nets. The current implementation includes the basic concepts; the next implementation will be significantly extended, e.g. for the Event Handler a full Information Service will be implemented on top of a SQL Server.

For example, the Activity Control itself is also implemented using Petri Net models. It uses the Interpretation Engine to interpret models of Functional Operations, the Resource Engine to assign and release resources, the Configuration Manager to create and delete FOs, the Event Handler to convert events into actions and the Interaction Control to invoke functions of the Front-End Service providing the Dialogue Control functionality. The Front-End converts the technology independent description of Functional Actions of a particular Functional Action into a set of technology dependent Service Units used, e.g. to invoke MMS service primitives for communication with Controllable Resources (Humans, Machines).

The generation of the related sequential and concurrent control structure corresponds to CIM-OSA Model Creation Process in the Function View. Flexible changes can be easily done using menu driven window oriented graphical editor. The validation of models can be carried out by the use of the simulation mode together with Petri Net based formal verification capabilities. After the validation, the models are released for the on-line execution.

Machine Front-End is fully implemented as a prototype. For validation purposes the Machine Front-End Service can also be modelled in Petri Nets, hence machines can be controlled from the IEE tool directly.

An important feature of the tool is its capability to visualise flows of actions in the sequential control structure of CIM-OSA models during the execution. Even run-time changes of the order sequence can be included to achieve operational flexibility. The tool also offers step-wise traceability and walk through capabilities for the control of shop floor devices.
<table>
<thead>
<tr>
<th>start event</th>
<th>Original CIM-OSA Graphical Representation</th>
<th>High Level Petri Nets</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
</tr>
<tr>
<td>end event</td>
<td>EF1</td>
<td>EF1</td>
</tr>
<tr>
<td></td>
<td><img src="image3" alt="Diagram" /></td>
<td><img src="image4" alt="Diagram" /></td>
</tr>
<tr>
<td>unconditional</td>
<td>EF1 → PR1 → EF2</td>
<td>EF1</td>
</tr>
<tr>
<td>conditional</td>
<td>EF1 → PR1 → EF2</td>
<td>EF1</td>
</tr>
<tr>
<td>spawning</td>
<td>EF1 → PR1 → EF2</td>
<td>EF1</td>
</tr>
<tr>
<td>rendezvous</td>
<td>EF1 → PR1 → EF2</td>
<td>EF1</td>
</tr>
<tr>
<td>alternative</td>
<td>EF1 → PR1 → EF2</td>
<td>EF1</td>
</tr>
<tr>
<td>convergence</td>
<td>EF1 → PR1 → EF2</td>
<td>EF1</td>
</tr>
</tbody>
</table>

EF2 starts when EF1 is finished and PR1 is true.
Transition T is fired when PR1 is true.
EF3 starts when EF1 is finished and PR1 is true and EF3 is active.
EF3 starts when EF1 is finished and PR2 is true and PR1 is true and EF2 is not active.

**Fig. III.3: CIM-OSA Procedure Rules and Petri Nets**
IV. CIM-OSA FUNCTION MODEL OF A SHOP FLOOR CONTROL STRUCTURE

On the basis of VOICE users' requirements several scenarios have been investigated. Coordination of machines, process tracking and monitoring are mainly considered. As an extract of the above, a test case scenario has been synthesised. Intentionally, a small but clear and representative CIM-OSA Function Model for the scenario is chosen. Primarily we focus on the methodology, and to be able readily to demonstrate results and port them to test beds.

The scenario implemented currently is the order sequence of a teach in application of two robots. The movements of the master robot1 are followed by the slave robot2; the co-ordination and synchronisation of robots are monitored, too.

The order sequence is described here as a functional behaviour model of a CIM-OSA Business Process TeachRobot. It is described by a set of Enterprise Activities and the Procedural Rules separating them (Fig. IV.1). All figures in this and in the next chapter are screen areas printed from the tool as snapshot shots either during the edit or the execution (simulation) mode.

Four Enterprise Activities (EAs) compose the Business Process. They are: OpenShopFloor, Controlling, Monitoring and CloseShopFloor. The structure of the model indicates that the EAs OpenShopFloor and CloseShopFloor will be executed sequentially, whereas the Controlling and Monitoring EAs are executed concurrently.

Every EA contains one or several Functional Operation(s) on the next lower hierarchy level. Functional Operations (FOs) are defined to describe functions of a single controllable resource.

Each of the Enterprise Activities OpenShopFloor and CloseShopFloor is subdivided into two FOs: the SetUpRobot1 & SetUpRobot2 and ShutDownRobot1 & ShutDownRobot2, respectively (Fig. IV.2).

The Enterprise Activities Controlling and Monitoring contain only one Functional Operation each: Synchronize and Demonstrate, respectively (Fig. IV.3).

After building the above Particular Implementation Model it can be released for use as a part of the operational model. It contains operational (i.e. occurrence of) Business Process TeachRobot described by the Enterprise Activities and related Procedural Rules. This occurrence is operated under Business Service to execute models.

V. EXECUTION OF A CIM-OSA FUNCTION MODEL

PER definition a CIM-OSA model is executed by the Integrating Infrastructure. The structure of the IIS is modelled using Petri Nets and expressed by the CIM-OSA graphical representation (Fig. V.1). In addition to the Services BS, IS, FS and CS, the System Management (SM) Service has also been included. SM is defined by the CIM-OSA Formal Reference Base II [AMICE].

As described in chapter, 2, at first all Integrating Infrastructure Services are described formally using Petri Nets. Stepwise, some of them are substituted by real models. Currently, the Machine Front-End and the Communication Service are implemented as real modules; in Fig. V.1 it is represented by the black inside of the related icons.
The model of the Business Process under consideration (TeachRobot) is supposed to be stored in a Database, therefore it is accessed by the Information Service.

The Machine Front-End Service is actually fully implemented on a separate network node. However, a simplified Petri Net model may be used during the simulation and validation of models.

**Business Process Control and Activity Control**

We start here on the second level of the hierarchy where the Business Process Control uses the Interpretation Engine to interpret the Business Process Model. With the start event (symbol $\mathcal{S}$) of the Business Process, the Business Process Control (BC) requests the Resource Engine to schedule the Resource Capacity, in our case two robots. After the successful allocation and when the first Enterprise Activity (OpenShopFloor) reaches its start time, BC inquires the Activity Control to execute the Activity.

When the last Enterprise Activity is executed (with the final event (symbol $\mathcal{E}$) of the Business Process) the Event Handler informs the Business Process Control and the later will ask the Resource Engine to release used resources (two robots) and the Event Handler will create an end event for the Business Process occurrence.

**Functions & Actions & Interaction Control**

*Functions* are the basic units of processing offered by the IIS Services. Various functional actions are defined. A *request* is a primitive issued by the client to invoke a function of other Service. A *response* is a primitive issued by the service to complete the function previously invoked by the client.

The Activity Control controls the execution of the Activity OpenShopFloor by dispatching concurrently the Functional Operations SetUpRobot1 and SetUpRobot2, which constitute the Activity. It uses on the next lower layer of decomposition the Interaction Control.

The dynamic description of the Interaction Control (Fig. V.2) illustrates how sequences of interactions are related in time. It takes a form of sequential list of events and functional actions (FAs), linking events to actions.
TeachRobot

Fig. V.1: Model of the Integrating Infrastructure

Three Functional Actions (FAs) are implemented in the Interaction Control so far:
- Create FO and its counterpart Delete FO use Configuration Manager to invoke System Management Service (SM)
- Start FO uses Dialogue Control to invoke the relevant Machine Front-End Service (FS)

Only the Functional Action Start FO is considered here; a full implementation of a Standard Control Structure is currently in the test phase.

In the following, the Function Operation SetUpRobot1 will be described in more details examining the Fig. V.3 with its snapshot during the execution of FOs.

Using the action Create FO the Configuration Manager will create an object for this new FO occurrence and return a valid net-wide unique identifier (e.g. identifier 101 for the first FO SetUpRobot1) as response. This identifier is expected in the following requests of the Interaction Control to the Front-End Service.

The description of Functional Actions related to the Front-End Service begins with the FA name, the FO identifier followed by FO name, the device and possibly other request and response parameters. In our example the parameters are:

- Function: START
- FO identifier: 101
- FO name: SetUp
- device: robot1

The interaction Control uses above parameters to execute the Functional Action. Actually, the Interaction Control invokes Machine Front-End Service to execute it. In our case either the simulated Machine Front-End is invoked, or the one fully implemented on the separate node.
The implemented MF [WNH92] will use the internal structure of the particular FO included in its knowledge base to convert the FO into a sequence of particular (possibly MMS based) machine specific processing steps to be executed. For instance, the sequence for the SetUpRobot1 is composed of two e.g. MMS based Service Units: Down Load and Start Program Invocation. The related service units include several MMS service calls already available in the library. No additional programming work with MMS primitives is necessary.

After the execution of the Functional Operation, MF reports the result back to the Interaction Control. In Fig. V.2 the response is presented by its protocol data unit (101, Setup Robot1 EXECUTED, REUSABLE), where EXECUTED means the event and REUSABLE the state after the execution of the Machine Functional Operation.

During the next Functional Action concerning the same FO, the Configuration Manager would ask the System Management to delete the related FO.

According to the description of the Activity OpenShopFloor it is obvious that its two FOs SetUp Robot1 and SetUp Robot 2 should be processed concurrently. This can be seen in Fig. V.2 where besides the response protocol data unit of the FO 101, Start 102 SetUp Robot2 also appears. The later is the requesting protocol data unit on the interface between the Interaction Control and Front-End Service (FS) for the robot2.

After the both FOs are processed the robot 1 and robot2 have been prepared for operation. the Activity Control (AC) reports this to the Business Process Control (BC). The Business Process Control then uses the Interpretation Engine to interpret the Business Process further. The Activity Control will process the next two activities in parallel (Controlling and Monitoring). The procedure is repeated until the last FO of the Activity CloseShopFloor and its last FA is executed.

VI. MCCIM AS CIM-OSA DEMONSTRATOR

The work performed followed the necessary steps to a CIM-OSA rapid prototype resulted in the development of so called McCIM demonstrator (see Fig. VI.1). Its components are distributed over three network nodes:
the Integrated Engineering Environment (IEE) node implementing the tools for model creation, simulation, verification, Business Service and System Management Services
- the IIS-node implementing the Machine Front-End prototype and protocol visualisation (Human Front-End is currently in the test phase.
- the shop floor node where controllable resources (shop floor devices) are simulated.

The communication between distributed components is supported by the CIM-OSA Communication Service implemented by the use of MAP/MMS on all three nodes.

**Machine Front End Service**

McCIM Machine Front-End is the first prototype [WNH91] of the Front-End Service conform with the CIM-OSA Reference Base I. It consists of a model supported knowledge base and a control engine. The separation of these basic elements has the crucial importance for the development of CIM applications. The knowledge base [WNH92] contains the application specific control knowledge of the CIM-OSA models, while the generic control mechanism is the kernel of the control engine. The approach provides an easy way for the migration of existing proprietary applications into CIM-OSA systems.

McCIM Machine Front-End can control and manage several devices concurrently. The MMS communication module LiveData [CYCLE] has been used for the implementation of service units. LiveData allows the access to MMS services from various McCIM applications. Nearly all MMS services can be used, MAP boards from Concord (1210 and 1215) and Computrol (LP 25) are supported.

**IEE/IIS Node**

**IIS Node**

**Shop Floor Node**

According to the model of the Business Process TeachRobot the shop floor environment includes two robots; they are simulated in McCIM so far. The Process Tracking visualises actions on the shop floor and communicates with McCIM Machine Front-End using MMS based service units.

To achieve verification and test of IIS protocols a comprehensive visualisation of the protocol flow between different CIM-OSA components a monitoring system is implemented in VisualBasic. This module will be also used for presentation and educational purposes in the scope of the information dissemination inside and outside the project.
VII. CONCLUSION

The ultimate goal of the system design process is to derive a system realisation as a particular instance of a system that fulfils user requirements. According to the state-of-the-art a novel approach has been implemented here, starting with a graphical presentation of models and achieving a executable rapid prototype of a MAP/MMS based CIM environment. Thus, all the goals of the demonstrator of the CIM-OSA functionality, described in chapter I have been fulfilled.

The benefit achieved is the provision of an model based approach for the integration of CIM applications into MAP/MMS based CIM target environments. This can contribute to the flexibility and efficiency of integration in manufacturing.

ACKNOWLEDGEMENTS

This work has been sponsored by the European Strategic Programme for Research and Development in Information Technology (ESPRIT).

The author is deeply indebted to Manfred Klittich from the project AMICE for his help to apply CIM-OSA, and to her colleagues Frank, Franz, Wu-Nan and Leszek for their common work on the McCIM demonstrator. Wojciech Molisz’s assistance in improving this paper as well as Dominique Diqutlou’s support in keeping track with VOICE project administration are appreciated, too.

REFERENCES

[AMICE] ESPRIT II Project 2422 AMICE Consortium, CIM-OSA Reference Base I and II


[PACE] PACE Reference Manual, Version 1.0.0, Grossenbacher Electronik AG, Spinnerrstr. 8, CH-9008 St. Gallen, Switzerland

[QUE92] B. Querenet, CIM-OSA - A European Development for Enterprise Integration, Part III: Enterprise Integrating Infrastructure, ICEIMT Conference, Hilton Head USA, June '92

[VER92] F. Vernadat, CIM-OSA - A European Development for Enterprise Integration, Part II: Enterprise Modelling, ICEIMT Conference, Hilton Head USA, June '92
