The flexible integration of machine objects within
distributed manufacturing systems

To cite this article: J M Edwards and I A Coutts 1995 Distrib. Syst. Engng. 2 183

View the article online for updates and enhancements.

Related content
- An overview of real-time ANSAware 1.0
  Guangxing Li
- An object-oriented framework for
dynamically configuring extensible
distributed systems
  D C Schmidt and T Suda
- Types and their management in open
distributed systems
  Wayne Brookes, Stephen Crawley,
  Jadwiga Indulska et al.
The flexible integration of machine objects within distributed manufacturing systems

J M Edwards† and I A Coutts

MSI Research Institute, Department of Manufacturing Engineering, Loughborough University of Technology, Loughborough, Leicestershire, LE11 3TU, UK

Received 6 April 1995, in final form 21 November 1995

Abstract. Manufacturing processes are required to adapt to change as businesses respond to global competition. The paper describes a framework for building distributed manufacturing processes based on an integrating infrastructure. Through a decomposition based on application function, application interoperation and application interaction, proposals are made for structuring the integration software required to flexibly implement distributed systems which can evolve to support required change. The paper describes the architecture used in a proof of concept implementation of a machine vision process made up of distributed application objects. Support for change is demonstrated through comparison with a conventional distributed system which does not make use of the services of an integrating infrastructure.

In describing the system architecture, requirements for "soft" CIM building blocks which can be "plugged" into an integrating infrastructure are defined. This work together with a range of approaches to application interoperation proposed by researchers at the MSI Research Institute are being further investigated through an EPSRC funded project entitled 'Manufacturing Software Interoperability: Steps towards Interoperating Distributed Objects'.

1. Introduction

Manufacturing machines are primarily complex pieces of mechanical engineering with associated electronic programmable controllers. They are designed with their immediate functionality in mind as stand alone processes capable of being bought 'off the shelf'. As such, the main design priorities are often software execution speed, reliable processing and mechanical accuracy. Unfortunately when viewed as a building block of an integrated manufacturing system these machines are often insular, inflexible, and do not enable modification as business requirements change. Much of their inflexibility stems from a lack of consideration given to the way such systems may be integrated.

Advances in computing technology have enabled the use of computer based machines within industry to become increasingly viable. Their application in future integrated manufacturing systems based on distributed technology implies the need for viewing these machines as application objects, or CIM building blocks, which provide manufacturing services and can be combined to achieve some integrated manufacturing process. In order that these objects can form part of a useful open integrated manufacturing system, where they can offer services to open client applications, the requirements for implementation within an integrated manufacturing system must be established.

This paper describes a framework for structuring the software required to link client applications with the services provided by a machine vision application object. This framework has been used to build a proof of concept distributed system implemented across Sun workstations and PC platforms. The framework, together with proposed implementation mechanisms, provide an integration methodology which supports systemized implementation and change, within an open distributed system. The proposals represent a position on the migration path from conventional systems to future manufacturing systems based on distributed interoperating object technology [4] which will become available as we approach the 21st century (typically through the work of the Object Management Group (OMG) [13]).

Current work at the MSI Research Institute, supported by the Engineering and Physical Science Research Council (EPSRC), is investigating the migration from conventional to distributed systems through the use of Object Request Broker technology, together with the investigation of mechanisms for object interoperation.

The framework proposed in this paper is based on elements which address the following issues:

† E-mail: j.m.edwards@lut.ac.uk
the architecture of soft† CIM building blocks which can be plugged into and removed from a software infrastructure which underpins a CIM system. This involves the separation of the following three issues:
  - manufacturing application functionality;
  - application interoperation functionality;
  - application interaction functionality;

  - the provision of interaction mechanisms which involve the use of services provided by a layer of infrastructural software, and the interfacing of alien devices such that they become compliant with these integration services;

  - the structuring of interoperation mechanisms which involve the creation of a virtual vision server and corresponding support for client applications, through a set of vision service functions. The virtual vision server then requires a mapping onto the real vision application object.

The paper briefly describes an integrating infrastructure known as CIM-BIOSYS (CIM Building Integrated Open Systems) which provides the infrastructural software alluded to in the three points above. CIM-BIOSYS has been known as CIM-BIOSYS (CIM Building Integrated Open Manufacturing) created by researchers at MSI [1] through a government funded research project. Proposals in the paper are made cannot

  - in line with intermediate or next generation open systems. Specialist facilities to enable interaction between open applications and the vision application object are needed because specialist vision hardware is needed which cannot run CIM-BIOSYS software. The generation of a vision alien device driver for CIM-BIOSYS is thus necessitated. The significance of this implementation is not only in the mechanisms described for interaction and interoperation but also because it offers a migration path from current solutions, i.e. discrete devices, to future fully open distributed applications. These fully open systems would be based on resources that can support a common integrating infrastructure such as that proposed by OMG in its CORBA (Common Object Request Broker Architecture) specification [13].

The architecture for integration includes mechanisms which will support the requirement of machine vision systems to adapt to change. These mechanisms are based on the following principles:

  - The provision of structure through:
    - the use of an overall architectural framework [1, 7, 12];
    - the use of a layered decomposition (within the complete soft integrated vision machine) incorporating interfaces based on virtual machine abstractions [15];

  - the use of structured software where additional unforeseen requirements can be implemented within existing C code templates [5];
  - An implementation based on the use of an integrating infrastructure where the infrastructure has underlying management facilities which can support change [19].

2. A soft integrated manufacturing system based on an integrating infrastructure

Researchers at the MSI Research Institute have for some years recognized the requirement for a single consistent interface for applications requiring integration level flexibility. This flexibility could be provided through the control and management of interprocess interaction and information sharing, ensuring provision for controlled change. To address this requirement an integrating infrastructure and platform of services was derived through practical experience of solving contemporary integration problems. This infrastructure has evolved into CIM-BIOSYS.

In its current implementation CIM-BIOSYS consists of a number of functional blocks as shown in figure 1. The manufacturing functions/applications shown in figure 1, are, in this context viewed as being those processes which perform some part of a distributed, yet integrated manufacturing operation, i.e. a cell controller or a scheduling application. The device drivers hide/cater for the diversity of both functionality and implementation of system resources. Typical examples of system resources include shop floor machines, proprietary databases and human operators.

The need for an integrating infrastructure, such as CIM-BIOSYS, is now widely accepted by the manufacturing systems integration research community [6, 16]. By enabling the creation of 'open' software, the use of an integrating infrastructure can promote the more systematic generation and change of flexibly integrated systems. This provides a means of dealing with the high levels of complexity found in most manufacturing organizations [1, 19, 20]. The integrating infrastructure deals specifically with issues of application interaction, not application interworking/interoperation.

Inter-working of application objects is supported within OSI layer 7 by the specification of a coherent set of communications functions termed 'application service elements' (ASEs which typically include file transfer, virtual terminal, job transfer etc [8, 14]). In general, an application layer protocol comprises a combination of these ASEs. Within the classifications of OSI, an integrating infrastructure such as CIM-BIOSYS is positioned at a level similar to that of an ASE. The CIM-BIOSYS platform of services is directly equivalent to an ASE while its configuration and management facilities are analogous to the proposals for mechanisms to support Open Distributed Processing [8]. The CIM-BIOSYS communications drivers are loosely based on the OSI model layers 1 to 6.

CIM-BIOSYS is a tool for integration, in its role as an ASE within layer 7 it offers managed integration services to applications. It can be used in combination with other
ASEs or to embrace existing ASEs. Most importantly, the additional configuration and management facilities which make up CIM-BIOSYS and underpin its integration services enable open systems interaction to be implemented in a soft integrated manner, i.e. they can adapt to required change.

It is proposed that further mechanisms are required to carry the 'soft' integration philosophy through to support application interoperability. These mechanisms should support the separation of application issues and interoperability issues which are considered to be conceptually separate [8] but are currently implemented with no explicit boundary. There is a need for an application service element (ASE) to support the interoperability of vision processing resources/services and client manufacturing applications within an open distributed system.

3. Building blocks of soft integrated manufacturing systems

In order to generate a soft integrated manufacturing system where application objects appear as discrete open applications (or soft CIM building blocks) within an integrated whole, two specific issues must be addressed:

- CIM-BIOSYS alien devices such as a remote vision processor must be handled in some way such that they appear to the CIM-BIOSYS software as compliant devices (i.e. provision of a CIM-BIOSYS alien device driver);
- the application objects must adhere to some predetermined messaging protocol such that they understand each other’s messaging dialogue. If they are to be implemented as applications providing open interoperability they must be supported by specialist mechanisms (as proposed in this paper) and/or adhere to some recognized messaging standard.

Figure 2A shows the logical elements or objects within a machine vision system for electronic component inspection prior to placement on a printed circuit board. A detailed treatment of the information support aspects of the system are given in [3]. Figure 2B shows a mapping of the distributed vision system onto the CIM-BIOSYS integrating infrastructure. This figure illustrates how the complete logical solution for the information model driven inspection application can be implemented on CIM-BIOSYS. The 'operator interface and information management object' is implemented as a CIM-BIOSYS compliant open manufacturing application. The 'live component information generation' object is implemented as a vision application object on a CIM-BIOSYS alien device. It is the mechanisms required within the relationship between these two objects which are the primary issues reported in this paper.

4. An architecture for soft integrated machine vision systems

The architecture proposed in this paper was developed and tested through the implementation of the physical representation shown in figure 2B. The system comprised a vision client application, an information view provision application (not discussed in this paper) and the vision alien device driver running on any Sun workstation, distribution flexibility being provided by the CIM-BIOSYS integrating infrastructure. This flexibility is provided through configuration facilities which provide a mapping between the logical elements of the distributed system and the physical whereabouts of the software objects in the system. The vision server was implemented using a Matrox [9] vision processor installed in a PC.

Figure 3 shows the layer by layer decomposition which make up the elements of the proposed architecture for a soft integrated vision machine. The decomposition between application object functionality, interaction issues and interoperaton issues is clearly indicated, as are three discrete processes, i.e. The Vision Client Application, the Vision Alien Device Driver and the Remote Vision Service Provider.

5. The open interaction of application objects

Structured and managed application interaction can be supported through compliance with the services of an integrating infrastructure. To provide CIM-BIOSYS compliance, enabling application interaction, applications should be able to invoke the CIM-BIOSYS interaction services and handle incoming CIM- BIOSYS data packets. Figure 3 shows the requirement for a 'CIM-BIOSYS

Figure 1. A functional view of CIM-BIOSYS.
Service Handling’ module in a CIM-BIOSYS compliant manufacturing application. Figure 3 also shows the additional requirement of an Alien Device Driver to provide CIM-BIOSYS compliancy for the Remote Vision Service Provider which contains the Vision Application Object.

The Alien Device Driver must provide the functionality which enables the Remote Vision Service Provider to appear to Vision Client Application as another open application, i.e. the Remote Vision Service Provider will offer vision services on the CIM-BIOSYS integrating infrastructure. To describe the functional requirements of the Vision Alien Device Driver, it is first necessary to detail the interaction aspects of the two processes which need to communicate—the Remote Vision Service Provider and the Open Vision Client Application.

The Remote Vision Service Provider and the Alien Device Driver interact using socket based client/server interprocess communications (IPC) mechanisms [17], which exist within the ‘Alien Device Server’ (PC-NFS Sockets) and the ‘Alien Device Client’ (UNIX Sockets) within the Remote Vision Server and the Alien Device Driver shown in figure 3. The alien device client software runs on any general purpose Sun workstation on the network which is running CIM-BIOSYS. When the alien device driver is invoked, its initialization routine creates an IPC socket. It then builds and sends an ‘initialisation’ data packet containing a request for the status of the remote vision service provider. On receipt of a positive response from the server, the alien device driver remains established to form its link between the vision server and open client application on the CIM-BIOSYS infrastructure.

The Open Vision Client Application and the Vision Device Driver require a general purpose processing resource, they run on Sun workstations and interact through the use of the CIM-BIOSYS service interface. Implementation of the current version of CIM-BIOSYS also achieves IPC based on connectionless sockets [17]. The IPC transfers formatted data packets between CIM-BIOSYS and open applications. A detailed treatment of the implementation mechanisms used within CIM-BIOSYS is presented in [2].

The use of an integrating infrastructure for object interaction removes the necessity for multiple bespoke links between objects. Any client application with knowledge of
how to use the vision services can apply to CIM-BIOSYS to establish a link with the remote vision server. The CIM-BIOSYS alien vision device driver provides a single link between the vision server and any client. This approach helps to solve the complexity problems associated with distributed systems made up of many interacting objects. Objects are registered once with CIM-BIOSYS and all object interaction is managed through the use of CIM-BIOSYS.

6. The Alien Vision Device Driver

The Alien Vision Device Driver has three principal functions which respond to external events, as shown in figure 4.

These can be considered as three discrete modules as follows:

- the interface to the Remote Vision Service Provider through the Vision Service User, which responds to the reply messages sent from the remote system (this module is equivalent to the Alien Device Client Service User layer in figure 3);
- the interface to the CIM-BIOSYS integrating infrastructure through the CIM-BIOSYS Service User, which responds to request messages sent via CIM-BIOSYS from open applications requesting services from the Remote Vision Service Provider (this module is equivalent to the CIM-BIOSYS Service User layer in figure 3);
- the Alien Device Driver operator interface which provides manual control and monitoring of the driver for debug purposes, and responds to requests via a window interface on a Sun workstation.

Protocol conversion addresses the requirement to convert from CIM-BIOSYS compliant message packet format to the packet format used by the Remote Vision Server, and vice versa. This functionality is driven by, and is essentially part of, the two service user modules identified above.

The event driven operation of the three modules which implement this functionality is controlled by the ‘Notifier’. The Notifier is a Sun tool [18] which provides a mechanism for distributing events to a number of functions within a process.

The function of the Vision Service User (handler) is to field data packets from the Alien Device Server, apply error checking procedures, and to process the message. The principal process is to construct a CIM-BIOSYS data packet using the complete vision server data packet as data, and send it to the associated open client application via CIM-BIOSYS.

The CIM-BIOSYS Service User (handler) is made up of a number of functions which are registered with CIM-BIOSYS and are called in response to incoming CIM-BIOSYS messages. A typical function could terminate the association between the client application and the driver another could return the status of the Remote Vision Server.

This registering of device driver functions with CIM-BIOSYS is typical of the structured approach embodied in the use of such integration infrastructures. The approach helps solve the problems of building and modifying distributed software. A requirement for a new system component demanding the addition of a new alien device driver can be accommodated through the registration of additional functions with CIM-BIOSYS where these functions form the basis of the new driver. The knowledge of where and how to make these additions comes from the adoption of a reference architecture or framework. Once registered, CIM-BIOSYS with its new driver provides open clients with managed access to the new remote object server/system device.

7. The interoperation of the vision server and the vision client

The previous section has discussed the interaction of system objects detailing how messages are securely transferred between objects in a structured system. This section describes the provision made for interpretation of the content of the messages which enables interoperation of the system objects. In this case interoperation is supported through the use of a virtual interface.

The use of a virtual interface for interoperation enables a range of different variants of a class of manufacturing device to interoperate with client applications, such that all variants of the device appear to be uniform. The virtual interface is a mechanism which supports application interoperation, by using an abstract representation of specific functionality provided by a real manufacturing machine [14].

Figure 3 shows the vision service user (or vision ASE) within the client device and the vision server (or virtual vision machine) within the alien device.

The Virtual Vision Machine (VVM) is that portion of the Remote Vision Server that maps the services provided by the Vision Application Object on to the interaction mechanisms described in the previous section. The nature of the VVM is then governed by the services provided by the Vision Application Object, and the interaction requirements of the remote device on which it is implemented.

A vision machine can provide services which usually involve the processing of raw images, and can range from the provision of a set of application specific features of an object of interest, through classification decision information, to information relating to the understanding of the scene being viewed. The Vision Application Object in this work extracts application specific features, i.e. corner points of electronic component package legs. The full specification of the Vision Application Object requires the following interoperation with a Client Application: program invocation; information upload, and; information download.

The Virtual Vision Machine (VVM) operates on the decomposed data packet which is passed to it by the Alien Device Server (see figure 3). This information comprises the package header and data buffer. The header contains a message identification field which controls how the VVM deals with the message.

Figure 3 shows the Vision Client Application has a layer dealing with the use of vision services. It is this
area of functionality which engages in dialogue with the VVM. It must compose messages in accordance with the requirements of the VVM and be capable of interpreting and dealing with the form of reply received from the VVM.

This work recognizes that in this case the problems of interoperation have been solved through specifying a particular set of vision service requests with specific semantics. The virtual interface provides flexibility in terms of replacing the vision application object.

This philosophy demands the creation and adherence to international standards for manufacturing object interoperation. Although standards bodies have had some success in this area [10] the philosophy suffers from an inherent lag in the system, i.e. when standards are agreed, technology has moved on.

Current work at MSI is tasked with providing a more flexible solution to object interoperation through mechanisms which aim to make available the semantics of object interaction at runtime.

The work described in this paper identifies the need for a separation between object function, interaction and interoperation issues, where the separation is clearly embodied in the implemented system. This separation provides additional support for the modification of systems in line with required change.

8. A modification to provide an additional vision service

A modification to provide a new vision service has two distinct parts. Modification is required to the vision processing related issues, and also to the integration related issues to make the new service available to client applications. This paper only addresses the integration issues. The following subsection describes the modifications required to the integration elements within the complete Soft Integrated Machine Vision System. This work is measured relative to early experiments in integrating distributed applications without the use of an integrating infrastructure.

8.1. Modification within a soft integrated machine vision systems

Figure 3 shows the elements of the Soft Integrated Machine Vision System. This figure helps to identify the extent of the required modification detailed in the following points:

- new facilities within the Vision Server, or Virtual Vision Machine in the Remote Vision Service Provider, will be required;
- new facilities within the Vision Service User, or Vision ASE in the Vision Client Application will be
required, such that new open applications software can interoperate with the vision server;
- no modification is required within the Alien Device Server of the Remote Vision Service Provider;
- no modification is required within the Alien Vision Device Driver;
- no modification is required within the CIM-BIOSYS service user of the Vision Client Application.

The points above illustrate how all the modification is required within those elements of the system which implement the interoperation and application functions. No modification is required to the facilities provided to handle application interaction. This highlights the importance of the decomposition between interaction and interoperation.

The new service requires the types of generic inter-operation facilities identified as necessary for distributed machine vision. In this case modification within the VVM in the Remote Vision Provider, and the vision ASE within the Vision Client Application can be minimal.

8.2. Modification within a conventional distributed system

In a conventional client/server system it would be typical for all application functionality associated with communications issues to be contained within a single module of each application. This module would typically be structured as a set of functions implementing the interaction and interoperation requirements of the system, as suggested by MacKinnon [8]. The software to implement application interaction mechanisms within a distributed system are notoriously time consuming to develop and test.

A modification to implement a new client application which makes use of a new service provided by the server would require the implementation of the interaction mechanisms within the new client application. This requires knowledge of the interaction mechanisms used within the server software. It is this re-implementation of interaction facilities that is overcome by the use of the Alien Vision Device Driver within the framework of CIM-BIOSYS.

The interoperation functions within the client and server of a conventional system would typically be implemented through Case structures as is the case with the VVM and Vision ASE of the soft integrated system. However, the identification of the generic functions within the VVM/ASE, together with the knowledge that the VVM/ASE is where new services are added are further elements of support that are missing in a conventional implementation.

9. Results and conclusions

The changes made within the elements of the proposed integration architecture in order to provide the new service took approximately one third of the time taken to change the conventional communications software. The time consuming work on the conventional system was primarily for re-implementing socket based IPC from scratch, in the new Client application.

Change to the software which implements the integration aspects of the machine vision system clearly highlights the advantages of implementing distributed processing using an integrating infrastructure. In particular, it shows the advantage of implementing the application

† A Case structure is a common software control construct, in 'C' it is implemented using a 'Switch' statement.
interaction mechanisms once only for any number of open client applications.

The separation between interaction issues and issues of interoperation provides a modular decomposition where modification can be isolated within the mechanisms for interoperation. In the implementation described these mechanisms comprise the VVM/ASE combination. Further structure within the VVM/AE provides a modular template such that additional functionality can be easily added to provide new vision services.

The overall architecture provides a reference framework, and the interaction facilities are domain generic and can be re-used. The VVM presents a consistent interoperation interface to new open client applications, while the ASE provides a set of ready built functions for inclusion within open applications to ease the use of the VVM services.

Heterogeneity is supported through the concept of virtual machines. Service specific software maps the VVM onto a particular Vision Application Object. Modification of this software enables a completely different Application Object, implemented using different vision hardware.

The following points summarize the usefulness of the CIM-BIOSYS integrating infrastructure in underpinning the design, implementation and maintenance of the integration issues pertaining to distributed machine systems.

- **Soft Integrated Building Blocks of Manufacturing Systems**: CIM-BIOSYS provides an underlying framework which structures the creation of manufacturing applications (or application objects) so that they become building blocks of flexibly integrated manufacturing systems. These building blocks can be plugged into the CIM-BIOSYS infrastructure through registration with the CIM-BIOSYS configuration information. (An example of a CIM-BIOSYS Open Manufacturing Application could comprise an executable C program, but any executable program capable of handling UNIX IPC is supported. It is this C program that is given a logical name and registered with CIM-BIOSYS.)

- **Reconfiguration**: The reconfiguration management mechanisms within CIM-BIOSYS provide distributed applications, and other manufacturing building blocks which interoperate with them, with a means of supporting long term flexibility and change. Reconfiguration takes place through modification of the CIM-BIOSYS configuration tables. This allows the association between logical applications and physical resources which make up the system to be modified without repercussion on the functionality of the distributed system.

- **Open Services**: An Alien Vision Device Driver and its associated Remote Vision Service Provider provides a set of open vision services. These are application specific services offered at a level above that of the CIM-BIOSYS interaction services. Any open application registered with the infrastructure can access any of the vision services offered.

- **Heterogeneity**: The formal structure of CIM-BIOSYS enables the inclusion of the Alien Vision Device Driver. Alien device drivers can enable flexible integration of legacy building blocks and map device specific communication technology onto the CIM-BIOSYS interaction services. This provides a consistent interaction interface through which open applications can interoperate.

- **Overheads**: The use of CIM-BIOSYS implies an overhead in execution speed, size of the implemented system and time required to learn how to use the tool. Evaluation work which addresses the two first points (in relation to distributed manufacturing control systems) has been completed by researchers at MSI [11]. The runtime performance testing shows the overhead associated with the integrating infrastructure represents a small fraction of the overhead due to execution of the application systems themselves.

### 10. Further work

As mentioned in the introduction to this paper work in the area of interoperating distributed object based systems is continuing at MSI with support from EPSRC. Object based technological innovation is available in many forms, typically many of the principles to support structured object distribution embodied in the CIM-BIOSYS integration infrastructure are now supported by commercially available first generation object request broker systems. All the major players in the computing arena have systems for object distribution (typically IBM DSOM, Sun DOB). Some suppliers make proposals for support of interoperation, usually within specific domains, i.e. Open Doc.

The current three year project seeks to utilize this emerging technology, initially in the area of manufacturing business process systems as opposed to the machine based processes described in this paper. The work is supported by both manufacturing companies (users) and manufacturing software vendors. The principal aim of the project is to develop a strategy for migration from current large granularity software products which do not map ideally onto the requirements of re-engineered manufacturing systems designed for optimum operation, to distributed systems comprising fully interoperating multi-vendor objects.

### Acknowledgments

The authors would like to acknowledge Jack Gascoigne and Paul Clements for their work within MSI on CIM-BIOSYS and systems interoperation. Also Professor Richard Weston who heads the MSI Research Institute.

### References


Flexible integration of machine objects


[11] Application and information support systems for planning and control in CIM Final ACME Report, grant no GR/F 69192 (available from the MSI Research Institute at Loughborough University, UK)


[16] Schuster H, Jablonski S, Kirsche T and Busler C 1994 A Client/Server Architecture for Distributed Workflow Management Systems Proc. Third Int. Conf. on Parallel and Distributed Information Systems (Austin, TX, USA)


