5

Rapid prototyping of integrated manufacturing systems by accomplishing model-enactment

M. W. C. Aguiar, I. A. Coutts and R. H. Weston
Manufacturing Systems Integration Research Institute
Loughborough, Leics. LE11 3TU, England, I.A.Coutts@lut.ac.uk

Abstract

This paper describes a model-driven approach to the flexible integration of software system components. This formalised approach is based on a combination and extension of state-of-the-art reference architectures centred on CIM-OSA (i.e. Open Systems Architecture for CIM). Also described within the paper are the techniques which have been devised to generate code of new software system components.

1 INTRODUCTION

Modelling provides a formalised way of coping with high levels of complexity commonly found when using computer systems to integrate manufacturing activities on an enterprise-wide scale. Many research initiatives have focused on defining models, reference models and reference architectures to capture design guidelines which help structure and support one or more aspects[1] and phases[2] of the life cycle of an Integrated Manufacturing Enterprise (IME)[3]. An analysis of the findings of such initiatives conducted by the authors in 1992 (Aguiar 1995b)

1. A modelling aspect is "an abstraction viewpoint of one total view, which emphasises some particular [view] of the model and disregards others for ease of analysis" (ESPRIT/AMICE 1993a).
2. In this paper, the life cycle of an IME is considered to encompass the following phases (or stages in the life of an enterprise): strategy definition, conceptual analysis, design, implementation, operation and maintenance. A more thorough discussion about the IME life cycle is presented in (Aguiar 1995b).
3. The terms integrated manufacturing enterprise, integrated manufacturing systems, CIM system are used interchangeably in this paper, as are CIM, system integration and enterprise integration. The difference between the use of these terms often lies in the scope or focus of their application.

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

concluded that: (1) considerable gaps exist in the formalism and completeness of any single reference architectures available at that time, hence no single architecture could lend structured support to the life cycle of an IME; (2) the most promising approaches were not yet directly usable by industry, due mainly to a lack of usable and comprehensive software tools to support the architectures; (3) potentially, a combination of architectures centred on CIM-OSA offered a way forward, if appropriate software tools could be conceived to facilitate their use. Here, the authors considered that CIM-OSA could be used as an overarching framework or skeleton which would support all key aspects and phases of enterprise-wide integration, but that this framework would need to be populated by other reference architectures and software tools created to deliver the technology to industry.

Although the CIM-OSA specification, conceived by the AMICE consortium (ESPRIT/ AMICE 1993a) can be implemented to provide solutions to major issues of Enterprise Integration, there are important issues which it does not address. This is particularly evident with respect to its 'function', 'resource' and 'organisation' modelling views at the 'design specification' and 'implementation description' modelling levels[4] (Aguiar 1993). Additionally, despite effort in on-going initiatives (Aguiar 1994a) (ESPRIT/VOICE 1995), CIM-OSA has yet to be implemented as an organised method supported by wide scope CASE tools which can progressively support modelling through to realisation of a software system running upon an integrating infrastructure.

Based on these findings, the authors proposed and have implemented SEW-OSA, which is a formal systems engineering workbench that combines CIM-OSA, generalised stochastic time Petri-nets, predicate-action Petri-nets, object-oriented design, and the services of the CIM-BIOSYS[5] integrating infrastructure to support systems integration projects.

This selection of architectures provides means of overcoming primary limitations encountered in the CIM-OSA architecture, namely:

- Petri-nets and object-oriented design were adopted to populate design specification and implementation description modelling levels of CIM-OSA, thus enabling complete support for model-building;
- Petri-nets were also adopted as a means of enabling analysis and simulation;
- CIM-BIOSYS was the only integrating infrastructure available (at the time that this research started) which could be enhanced in order to support model-enactment, thus enabling rapid-prototyping of an integrated system.

The acronym SEW-OSA (i.e. "system engineering workbench centred on CIM-OSA") was given to the workbench due to its orientation, namely: (1) it aims to address the **engineering** of an enterprise from a **systems perspective**; (2) its underlying framework is structured chiefly on the formalism of **CIM-OSA**; (3) it aims to provide the level of life-cycle support of a **workbench**, consisting of integrated tools for automating the entire design process, which

4. The CIM-OSA (Open Systems Architecture for CIM) architecture defines a modelling framework for model-building which embraces the definition of the modelling constructs required for modelling four views (i.e. function, information, resource and organisation), along three modelling levels or stages (i.e. requirements definition, design specification and implementation description) based on three levels of generality or detail (i.e. generic, partial and particular). CIM-OSA also provides the specification of an integrating infrastructure for model execution. A detailed description of the CIM-OSA constructs is presented in (Aguiar 1995a) (ESPRIT/AMICE 1993b)].
5. CIM Building Integrated Open SYStems is an integrating infrastructure produced by the Manufacturing Systems Integration (MSI) Research Institute at Loughborough University (Coutts 1992) (Weston 1990).

encapsulates: the steps to be followed during the process (i.e. a method); the constructs to be created and manipulated in order to formalise the design (i.e. a language); the considerations, analyses and decisions to be made at each step (i.e. a framework); and associated documentation of design activities.

A workbench (such as SEW-OSA) is necessary in order to enable the investigation of a plethora of integration 'problems' and possible 'solutions'[6], by providing means of: (1) describing each problem and defining possible solutions through the application of a formal language; and (2) testing the solutions. These requirements lead, respectively, to the proposition of the two main capabilities for such a workbench, namely: a **model-building capability** and a **model-enactment capability**[7] (as shown in Figure 1). Basically, each class of capability is required to support more than one modelling level of CIM-OSA, in a way which bridges gaps that exists within and between these modelling levels. Such capabilities are required to enable graceful migration from a modelling description of 'what' the system should do, to a description of 'how' the system should do it, by means of the actions executed by its components. Indeed, an important contribution envisaged for these two capabilities is that of defining a method for the organised application of the architectures, where their amalgamation would be under the framework defined by CIM-OSA.

Thus, SEW-OSA provides the first instance of an organised method for the application of CIM-OSA, by providing two classes of capability associated with its design methodology. At each stage of the design methodology, constructs[8] are manipulated in the form of diagrams and templates, thereby providing a means of organising design information as it is created.

Essentially, the method proposed is based on the application of a model-building capability to generate models (i.e. the business model shown in Figure 1) which formalise a possible relationship between problem and solution, which could then be made available to a model-enactment capability in order to test this relationship. The integrated use of these two capabilities is the fundamental support provided by SEW-OSA which this research has realised for encapsulating the formalism of selected architectures. Indeed the method entails the application of the model building capability outlined in Figure 2 which generates models which leads to the generation of models which can be input to and processed by the model enactment capability illustrated by Figure 1.

Between '92 and '95, the realisation of SEW-OSA provided a major focus of research effort within the "Model-Driven CIM programme"[9]. Indeed the aim of such an effort was to offer in SEW-OSA a framework of methods encapsulated into toolset within a scope which embraces the complete IME life cycle. SEW-OSA does not attempt to provide automated support for all design decisions, but it does provide a structured framework for decision making. As our detailed understanding of 'IME design best practice' grows, SEW-OSA can be advanced and

---

6. The term 'problem' is used here to describe a set of business requirements to be met or objectives to be achieved which motivated a system integration project. The term 'solution' is used to describe the final organisation of a system and specification of its components, constructed to address related requirements and objectives.
7. Model-enactment is viewed in this paper as the ability to let a model evolve over time (i.e. the evolution of its dynamic behaviour).
8. A construct (or a modelling construct) is a generic building block which characterises an element of the formalism of a modelling method or language.
9. Model-Driven CIM is a major programme of UK funded research (SERC/ACME research grant) at the MSI Research Institute.

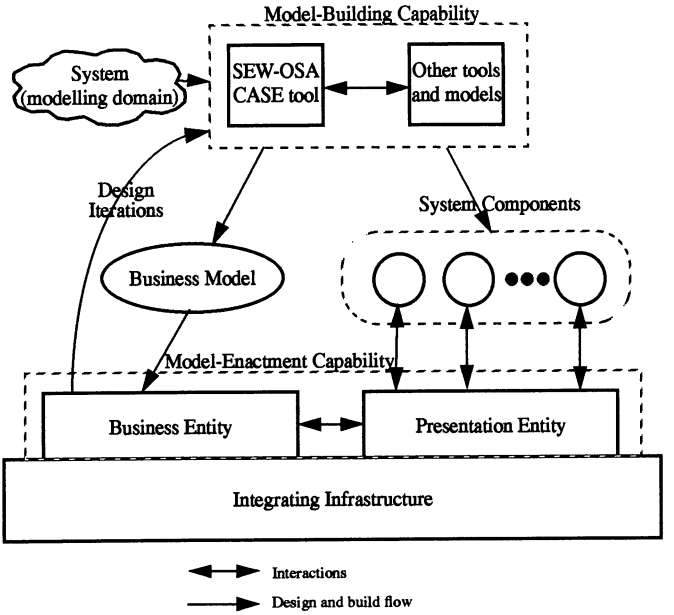

Figure 1 - SEW-OSA - a conceptual view

design knowledge[10] gradually fleshed into the framework. Here, the inclusion of design knowledge can be realised through use of models of integration 'problems' and 'solutions' to address them or by enhancing the modelling constructs of the SEW-OSA workbench.

This paper addresses two main issues. Sections 2 to 4 describe the way in which system prototyping is facilitated by the model-building and model-enactment capabilities of SEW-OSA. Here an overall description of the structure of the workbench is included. Section 5 describes the methodology adopted and implemented to generate the code of software components. Sections 6 to 7 outline major aspects of an evaluation of SEW-OSA through a case study.

2 Model Building Capability

The main contribution of the SEW-OSA model-building capability is the realisation of a modelling method which associates business process and object-oriented descriptions (as represented by the two clouds in Figure 2) in order to produce a complete business model. Such a method captures the essential definitions of the CIM-OSA modelling constructs,

10. Design knowledge is related to knowledge about 'problems', 'solutions' and possible relationships between them.

organises them into a usable form and enhances them with additional constructs from object-oriented formalisms and Petri-nets.

Following is an overview of the main diagrams which comprise the model-building capability of SEW-OSA. Correspondence between the acronyms used in the text below and the constructs manipulated in each diagram of the SEW-OSA CASE[11] tool is indicated in the model example represented in Figure 2.

Requirements definition modelling level

This modelling level embraces a business process description of the enterprise domains under consideration. This description formalises 'what' the system to integrate the domains is expected to achieve.

- **Context diagram**. This diagram defines the domains (i.e. main areas of an enterprise) under consideration, and the relationships between the domains. One context diagram is created for each individual enterprise model.
- **Domain diagram**. This diagram defines the major domain processes of a domain. One domain diagram is created for each CIM-OSA-compliant domain[12].
- **Structure diagram**. This diagram defines the functional decomposition of a domain process[13] in terms of enterprise activities and business processes[14] in a structured manner. One structure diagram is created for each domain process.
- **Behaviour diagrams**. These diagrams define the flow of control used to execute the functionality of a domain process and its business processes. Behaviour diagrams can also be referred to as process diagrams. One behaviour diagram is created for each domain process and subsequently for business processes within the domain process structure.
- **Functional diagram**. This diagram defines flows of material, information and control through the atomic building blocks of the domain process (i.e. enterprise activities[15]). One functional diagram is created for each domain process.

---

11. Computer-Aided Software Engineering.
12. According to the ESPRIT/AMICE consortium (ESPRIT/AMICE 1993b), domains "identify well-defined, totally integrated, functional areas of the enterprise". Domains are modelled either as CIM-OSA-compliant domains or non-CIM-OSA-compliant domains. A CIM-OSA-compliant domain identifies the area to be engineered [or integrated] within the enterprise. A non-CIM-OSA-compliant domain identifies other areas with which the area to be engineered interacts.
13. According to the ESPRIT/AMICE consortium (ESPRIT/AMICE 1993b), domain processes (or DP's) "are high-level processes [...] triggered by some events and producing a defined end result (function output). Domain processes are at the highest level of functional decomposition of domains. They must be triggered by nothing else than events" [12]. Domain processes can also be viewed as objects which communicate via exchange of information, material and events.
14. According to the ESPRIT/AMICE consortium (ESPRIT/AMICE 1993b), a business process (or BP) "is a sub-process of a domain process. It cannot be directly triggered by events and is always called by a parent process." A business process works as an intermediate construct between domain processes and enterprise activities. It should be noted that the term 'business process' is used in this paper in a particular sense (as adopted by CIM-OSA), and should be distinguished from 'business process' as used in a more general sense by Davenport, Shourt and Hickman (Childe 1993).
15. According to the ESPRIT/AMICE consortium (ESPRIT/AMICE 1993b), enterprise activities (or EA's) "describe [the] basic enterprise functionality (i.e. things to be done). They are defined by their function input, function output, control input, control output, resource input, resource output and ending status and have no behaviour defined at the requirements definition modelling level. They are always called by a parent process."

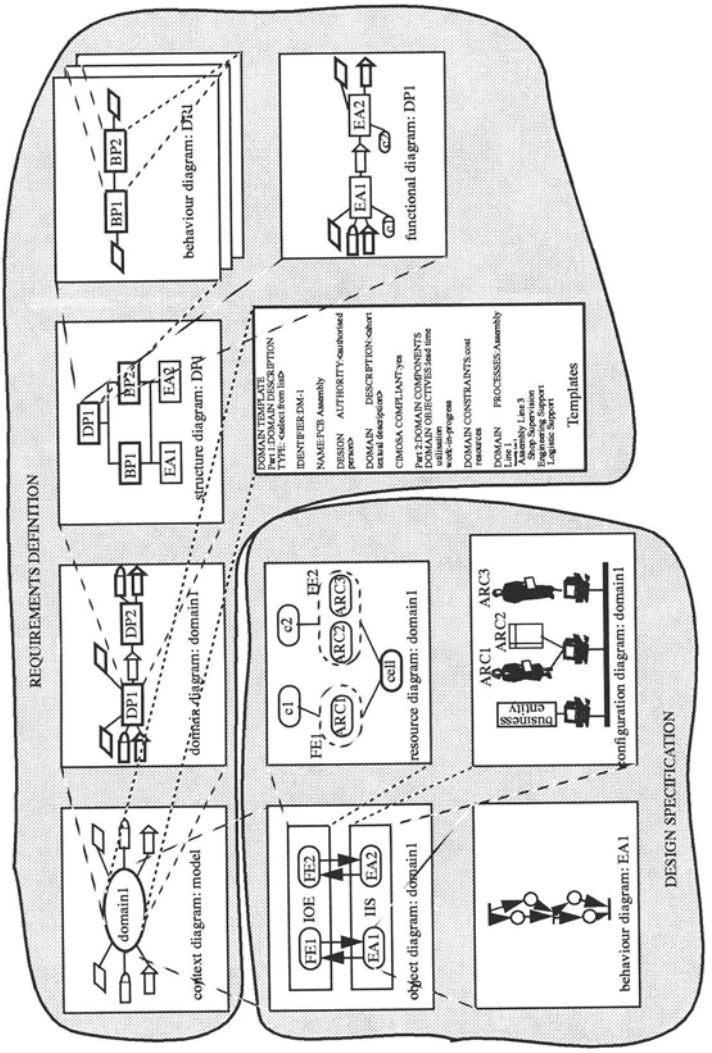

Figure 2 - SEW-OSA model-building capability

Design specification modelling level

This modelling level embraces a object-oriented description of the enterprise domains under consideration. This description formalises 'how' the integrated system achieves its purpose through interactions among its components and actions internal to each component (i.e. functions that each component performs on its own).

- **Object diagram**. This diagram defines the flow of messages between functional entities[16], the business entity[17] (i.e. enterprise activities) and the information entity of CIM-OSA. One object diagram is created for each CIM-OSA-compliant domain.
- **Activity behaviour diagram**. This diagram defines how the enterprise activity uses (via message exchanges) the integrated operation environment (i.e. a grouping of functional entities, or FE's), in order to perform its basic functionality (i.e. functional operations[18]). One activity behaviour diagram is created for each enterprise activity.
- **Entity behaviour diagram**. This diagram defines the expected external behaviour of a functional entity as perceived by the enterprise activities. Such a description is used to emulate the behaviour of an active resource component during the rapid-prototyping stage of a system. One entity behaviour diagram is created for each functional entity.
- **Resource diagram**. This diagram specifies instances of active and passive resource components (i.e. ARC and PRC)[19] associated with the classes specified by their functional entities which are able to execute the functional operations required by an enterprise activity. These functional operations are related to the capability required by the enterprise activity; the resource capability being defined in the functional diagram (see Figure 2). One resource diagram is created for each integrated operation environment associated with a particular domain (stemming from the IOE construct, as illustrated in Figure 2).
- **Configuration diagram**. This diagram defines the computer configuration of the system (i.e. where each active resource component will be executed or interfaced with). One configuration diagram is created for each segment of the IIS which serves a particular enterprise domain.

A text template is associated with each symbol represented in the diagrams of Figure 2, this prompts the user to define all necessary attributes. The text templates implemented in SEW-OSA comply with the format and attributes of the templates specified by CIM-OSA. The use of diagrams is not explicitly defined in the CIM-OSA specifications but was found by the authors to be very important to improve the usability of the method.

---

16. According to the ESPRIT/AMICE consortium (ESPRIT/AMICE 1993b), a functional entity "is a resource able to perform, completely on its own, a (class of) functional operation(s)". functional entities are viewed in this thesis as functional representations of active resources components required to fulfil the capability associated with enterprise activities identified in the functional diagram.
17. The business entity includes functions required to control the enterprise operation as described in its business model (i.e. function, resource and organisation models). This entity plays a key role in enabling business integration (i.e. coordination of interaction among system components).
18. According to the ESPRIT/AMICE consortium (ESPRIT/AMICE 1993b), a functional operation "is a basic unit of work defined at the design specification modelling level (i.e. lowest level of granularity in the function view). At run-time, [a functional operation is] fully executed or not at all."
19. An active resource component identifies a component of a system which is able to execute functional operation(s) on its own. It can also be a modelling description which characterises either a human being, an application program or a machine that possess a computerised controller (i.e. human functional entity, application functional entity and machine functional entity (ESPRIT/AMICE 1993b)). A passive resource component is an object used by the active resource component when performing functional operations.

3 Model Enactment Capability

Design information formalised in the form of models produced by the SEW-OSA model-building capability are passed in the form of a number of pieces of interpreted code (i.e. the business model in Figure 1) to the SEW-OSA model enactment capability. The model-enactment capability manipulates the design model to enable simulation and rapid prototyping of solutions. Model execution is achieved through populating the "design specification" and "implementation description" modelling levels of CIM-OSA with the elements defined in the following.

The design specification modelling level:
A capability has been implemented to facilitate analysis of Generalised-Stochastic Time Petri-nets. These nets are generated automatically by a software tool created to process CIM-OSA functional models. Also, tools have been produced to simulate the dynamic behaviour and to carry out performance evaluation of the system based on the execution of the Petri-Net model in simulated time (here various metrics can be used, such as: lead-time and cycle-time values, level of utilisation of resources, profile of work-flow, work-in-progress and cost). The methodology devised and supported to facilitate such evaluation exercises is discussed more thoroughly in (Aguiar 1993).

A capability was also conceived and implemented to facilitate rapid prototyping of system solutions and their (emulated) components. This capability was included to facilitate testing of the system structure, this by providing means of executing the business model generated by the CASE tool. In conformance with CIM-OSA ideas, the rapid-prototyping capability achieved model execution via a business entity.

The implementation description modelling level
This capability facilitates configuration of the physical system by gradually replacing its (emulated) components by physical ones which will be used in the final system (i.e. machines, application programs, data storing devices and human beings). As illustrated in Figure 1, physical components gain access to the integrating infrastructure via appropriate interfaces (i.e. the presentation entity[20]).

Before a physical system is finally commissioned, typically there will be a series of iterations between the use of the model-building and the model-enactment capabilities. Indeed, one of the most important contributions of SEW-OSA is that it facilitates these iterative processes in a structured and consistent manner, thereby traversing different phases of the IME life cycle. This can lead to much improved solutions and much shorter design-to-build lead-times via what essentially becomes an integrated systems engineering process. Thus, SEW-OSA provides the support of an organised method based on model-enactment, which brings together conventionally separate life cycle phases, from 'requirements definition' to the

20. The presentation entity provides means of integrating enterprise components (including legacy components). This entity maps the CIM-OSA internal protocol into protocols that are understood by enterprise components (e.g. proprietary machine commands). This provides the remaining entities of the CIM-OSA IIS (i.e. the business entity, the information entity, the common services entity and the system management entity (ESPRIT/AMICE 1993a)) with a uniform means of interacting with heterogeneous system components. Such a mapping is defined based on the models manipulated by the function and resource views.

'configuration and execution' of computer integrated manufacturing systems upon an industrially tested integrating infrastructure (i.e. CIM-BIOSYS).

4 Realisation of SEW-OSA

Realising, applying and validating the model-building and the model-enactment capabilities involved the development activities identified by a tag number in Figure 3, namely: (1) developing a CASE tool that encapsulates the CIM-OSA model-building method; (2) establishing a link to a simulator to enable dynamic analysis based on use of a business model; (3) creating a business entity for CIM-BIOSYS which enables rapid prototyping of systems; (4) establishing links to other tools and services (created by other researchers working within the 'Model-Driven CIM' research programme) this to address design aspects not covered by SEW-OSA (e.g. detailed information and resource modelling) (Clements 1993), (5) developing a presentation entity for CIM-BIOSYS; (6) evaluating use of SEW-OSA through a case study application to solve shop-floor integration and coordination problems at an industrial site; and (7) validating the results obtained. Implementing the elements tagged (1) and (3) (which are of focus of interest in this paper) involved the following research activities:

CASE tool

In order to achieve such a method, two major tasks have been accomplished, namely:

- (1) to **review CIM-OSA** (by analysing, correcting and complementing those parts that were not well defined) particularly in regard to design specification and implementation description modelling levels and
- (2) to **enhance CIM-OSA** with modelling constructs from the other architectures, this leading to the implementation of part of the CIM-OSA modelling methodology in combination with predicate-action Petri-nets and an object-oriented representation in a CASE tool.

As a result of these tasks, a number of incremental contributions (in their own right) have been made (refer to (Aguiar 1995a) for details).

Business Entity

Realising the business entity involved to:

- develop a model interpreter and debugger which incorporates a capability to drive interactions between system components. This is realised by interpreting business models generated by the CASE tool. Here, the interpreter and debugger comprises the following four components, namely: an Event Handler, a Process Controller, an Activity Controller and a Resource Manager (see Figure 3). Each of these components addresses a particular aspect of the business model. These components interact with one another via the CIM-BIOSYS infrastructure. Further information about the Business Entity of SEW-OSA is presented in (Aguiar 1994b). The business entity consists of a layer of services (written in "C" which execute in an X-Windows/Unix environment) which draw upon the general integration services of the CIM-BIOSYS infrastructure (which functions as a distributed operating system, flexibly mapping application software onto manufacturing and computing resources). This layer of services achieves a link to the CASE tool and, in so doing, enacts models which it helps create, thereby achieving structured and flexible integration of system components;

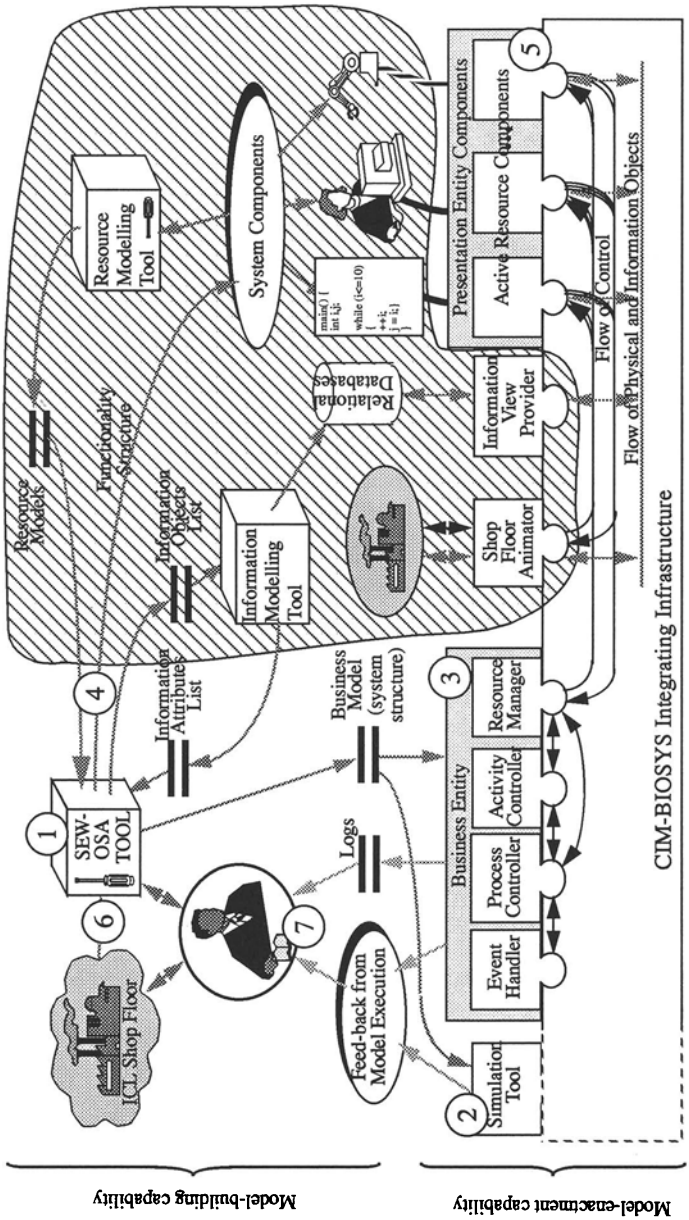

Figure 3 - System Engineering Workbench centred on CIM-OSA - a physical view

- enable rapid generation of prototypes of the system structure and its components, where the latter are generated in an emulated form. In the early stages of rapid prototyping, Active Resource Components (represented in Figure 3) consist of interpreters of predicate-action Petri-net models, which emulate their internal behaviour. The structure of the Active Resource Components is used as a basis for generating software application components, as discussed in the following sections.

5 Application Software[21] Generation

This section describes the infrastructural basis upon which SEW-OSA can be applied to the production of flexibly integrated distributed systems i.e. systems which comprises distributed interacting software components which are hardware and operating system independent.

Previous research at the MSI Research Institute (Coutts 1992) highlighted the need to facilitate interactions between distinct functional elements via the use of an integrating infrastructure. Here the infrastructure should provide means of achieving interaction between applications but should not define the manner in which they occur i.e. contain and prescribe the behaviour of a system. Recent advances in system modelling have provided methodologies and tools which can describe the behaviour of separate interacting elements, but to date have not provided a clear path to the automatic creation of code which realises such behaviour.

Current initiatives within MSI (Clements 1993) have lead to the definition of a structure for software components which separates internal functionality and behaviour whilst enabling interaction between applications. Such a separation is of vital importance in establishing formal and generalised methods of using a combination of high level modelling techniques and "bottom-up" implementation expertise within computer aided environments for rapid prototyping of systems, this including the generation of code for final system implementation.

Figure 4 depicts the application decomposition proposed by the authors. It should be stressed that the focus here is on the generation of code which achieves both the behavioural and interaction aspects of the application, not on its internal functionality.

The previous section discussed the generation of "Enterprise Activities" and "Active Resource Components" using SEW-OSA. The behaviour of these activities and components whilst interacting with the external environment is described (within the CASE tool) by predicate action Petri-net models. Using MSI's application structure the requirement for rapid system prototyping can be met by a means of enacting these models. This approach provides a means of mapping modelled systems (represented in some abstract form) onto implemented systems.

5.1 Current Implementation of the Model Enactment Facility

A Prolog execution environment (Clocksin 1984) was chosen to execute the internal behaviour of enterprise activities and active resource components described by predicate-action Petri-nets for the following reasons:

- it provided a natural environment for predicate logic;

21. The terms 'application', 'application software' and 'software component' are used interchangeably in this paper.

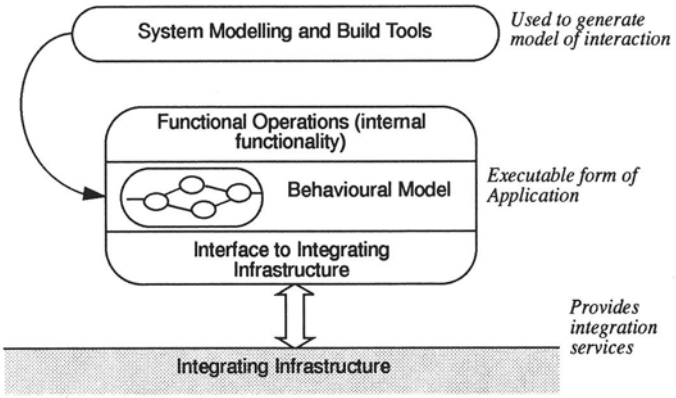

Figure 4 - Proposed Structure of an Application

- it offered the capability to extend its parser to allow user defined syntax (i.e. the transition syntax of Petri-net models);
- it facilitated incorporation of additional functionality (i.e. support for Petri-net token counts);
- it possessed an inherent ability to deal with the asynchronous nature of interacting elements (such as incoming message arrival instantiated in a Prolog database).

A link between the execution environment and the CIM-BIOSYS integrating infrastructure was established, so that the interactions described by the Petri-net models could be realised. A functional decomposition of the environment created is illustrated in Figure 5. A brief description of each function block follows.

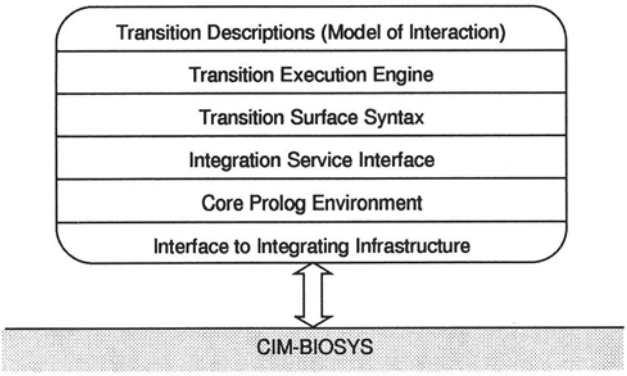

Figure 5 - Predicate-Action Petri-net Execution Environment

Interface to the Integrating Infrastructure

This comprises an inter-process communications mechanism, packet encode and decode and low level 'handshaking' functions required to interact with the CIM-BIOSYS integrating infrastructure. These have been coded in 'C' and incorporated within the runtime Prolog system. Incoming packets from other CIM-BIOSYS applications are instantiated as facts in the

Prolog database.

Core Prolog Environment
This is the "C Prolog" system as defined by Fernando Pereira, July 1982, EdCAAD, Dept. of Architecture, University of Edinburgh.

Integration Service Interface
Within this layer, the CIM-BIOSYS integration services are represented as Prolog facts. By instantiating a given fact, integration service requests are sent to CIM-BIOSYS. Responses from CIM-BIOSYS are instantiated in the Prolog database. A couple of examples follow.

Send message "start function 3" (sequence number 21) to an application called "tom".

```
send_app(tom,21, "start function 3").
```

Retrieve data regarding object "pad_positions" where attribute "no_of_legs" is equal to 2.

```
sel (pad_positions, "where no_of_legs = 2").
```

Transition Surface Syntax
This layer includes a parser which reads and evaluates both conditions and actions. The conditions determine if a particular Petri-net transition will fire. If the condition is true the accompanying action is executed. Global variables are also supported by the parser. An example of the condition syntax follows.

If variable i is equal to 2 and variable j is less than ten, send message "hello world" to an application called "fred" then increment variable i by 1.

```
on i = 2 & j < 10 do send_app (fred, "hello world") @ i is i + 1.
```

Transition Execution Engine
Once an application is invoked this engine tests and fires the transitions represented by the Petri-net model.

Transition Descriptions (Model of Interaction)
This comprises of the list of transitions and global variables which form the Petri-net model. An example of the syntax used in this model is presented as follows.

Transition name "link_to_fred", triggered when i is equal to 2 and j is less than 10. This transition establishes a link with an application called "fred" and then increments i by 1.

```
transition(link_to_fred,((i=2)&(j<10)),(est_link(fred) @ (i is i+1))).
```

6 Model Enactment Example

This section briefly outlines an industrial case-study application using the SEW-OSA workbench. The modelling domain chosen for this evaluation work comprised the Surface Mount Technology (SMT) assembly lines of a major UK supplier of electronics products. In this paper focus will be on a 'TO-BE' description of just one of these assembly lines. The model is a 'TO-BE' description in the sense that it anticipates use of an integrating infrastructure, whilst accurately modelling the current mode of operation of the assembly line. Figure 6 shows the basic arrangement and associated material and control flows for the SMT assembly line. Thus, batches of PCB's flow along the line and are inspected at two inspection points, i.e. after the printing process and whilst located on the inspection conveyor. Inspection

operations are triggered either by problems detected by the operators or as part of a regular inspection process which is programmed to occur (e. g. every "n" PCB's manufactured at the point in question). Buffers are used to smooth the flow of material and overcome imbalances in the line, thus avoiding inappropriate PCB queue sizes.

This process was modelled using the SEW-OSA model building capability in order to:

- formalise the requirements associated with the process (i.e. populate the requirements definition cloud illustrated by in Figure 2).
- define a system configuration which can meet these requirements (i.e. populate the design specification cloud illustrated by Figure 2).

A fragment of the requirements definition model is shown in Figure 7. This figure presents a simplified version of the domain diagram and is effectively the source of the behaviour diagrams of the business model. The behaviour diagrams so produced encode necessary coordination activities related to system components (such as operators and printers involved in the preparation stage of the assembly line).

A fragment of the design specification model produced during the evaluation study is shown in Figure 8. Here, enterprise activities identified in Figure 7 implement a functional transformation (i.e. act on their information and material inputs, in order to produce desired outputs). They achieve this by requesting the execution of functional operations (i.e. FO's) supplied by the functional entities of the system. Functional operations can also relate to data access operations (as shown in Figure 8).

The internal behaviour of the enterprise activities so defined is described by predicate-action Petri-nets. An example of such a description for the enterprise activity "print" is shown in Figure 9. Here the top and bottom transition represent the initialisation and termination of a run-time occurrence of the enterprise activity. On initialisation, a token is included in pEA-3_1 which enables tEA-3_2. When tEA-3_2 is fired the functional operation FO-5 is sent to FE-2 (see Figure 8). Then, EA-3 waits for the receipt of FO-6 in order to carry out its processing. FO-21 and FO-22 represent a data reading operation from a data-base. This piece of data is essentially a count of the PCBs produced between the two inspection operations (i.e. the number of executions of the enterprise activity "check" illustrated in Figure 7). Conditions tEA-3_3 and tEA-3_4 in Figure 9 indicate alternative ending statuses for this enterprise activity upon which a decision is made as to whether to check the PCB or to finish executing the procedural rules of the business process "prepare" (as shown in Figure 7).

As part of the evaluation study, similar behavioural descriptions were defined for all remaining processes (i.e. Figure 7) and activities (i.e. Figure 9) in the business model. Having completed the business model, interpreted code can be generated for the system (i.e. model-enactment for rapid-prototyping of the system, as illustrated in Figure 1) in order to test the design solution.

Figure 10 depicts example code produced by SEW-OSA, in this case to enact (and hence execute) EA-3 using the predicate-action Petri-net execution engine described in this paper. This code along with other code fragments for the remaining constructs of the business model are executed by the business entity (see Figure 3). The business entity functions as an engine which enacts the business model in order to coordinate all interactions amongst system components.

A number of iterations of model-building and model-enactment were found to be necessary in this evaluation study before a satisfactory design solution was obtained for the system represented in Figure 6. It is expected that such an iteration process would normally be

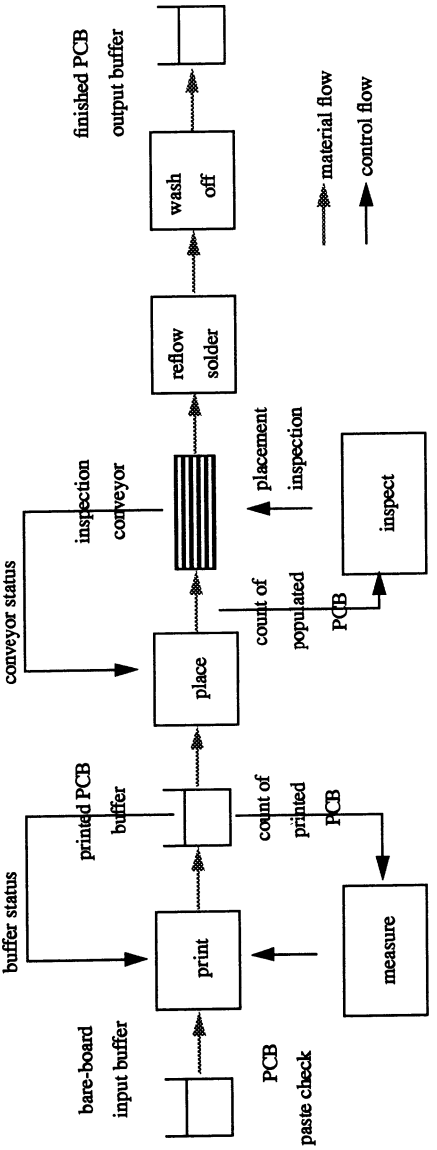

Figure 6 - Surface Mount Technology Assembly Line

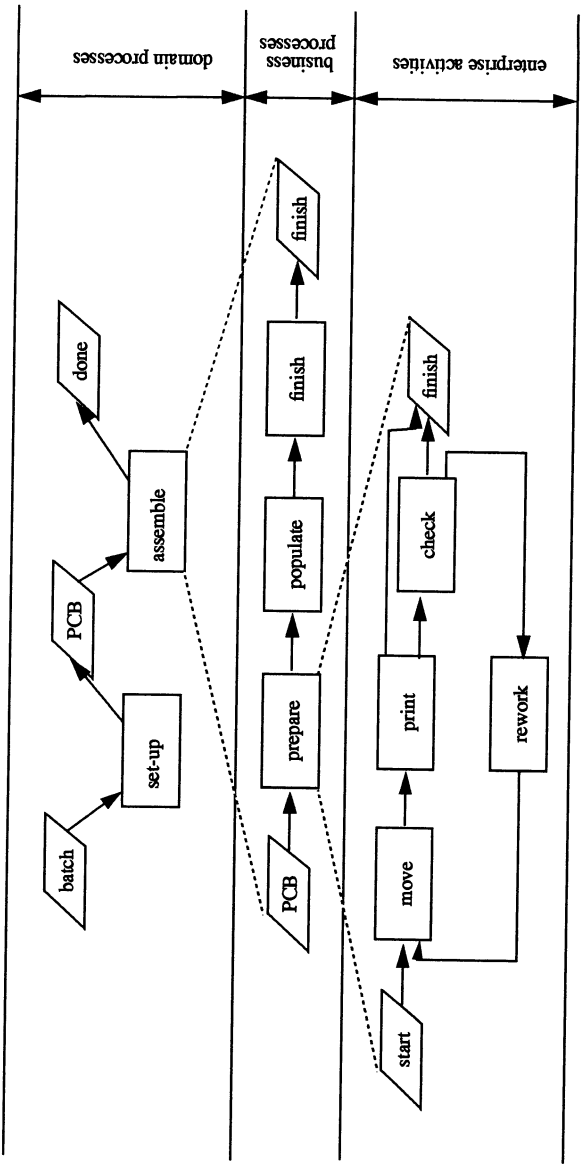

Figure 7 - Fragment of the behavioural model of the SMT assembly line

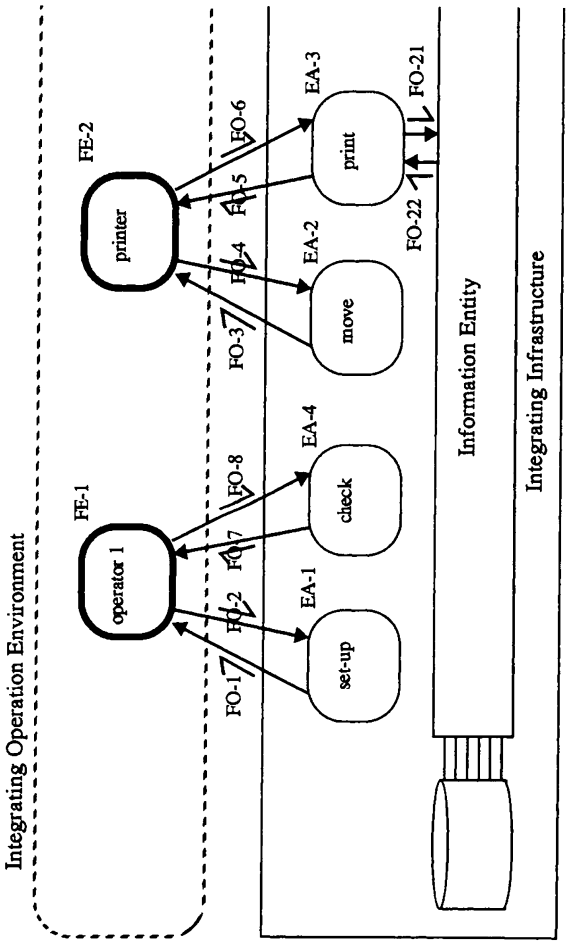

Figure 8 - Object Diagram: smt line

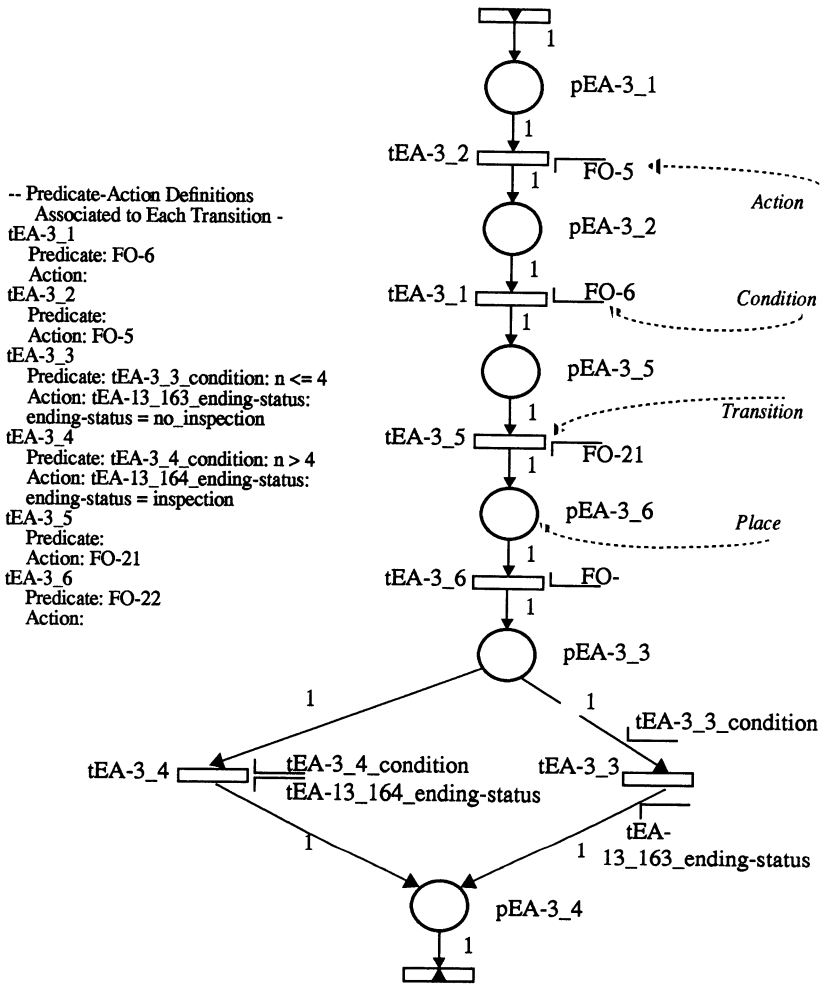

Figure 9 - EA-3: Behaviour Diagram: print

required, particularly where a largely untested system model is being created and used.

Predicate-action Petri-nets have provided a satisfactory way of describing internal functional flow and external infrastructure interaction, but the Prolog environment used for their execution does not possess the runtime performance required to implement complex systems. Hence ongoing work at MSI is focused on re-implementing the model enactment facility in the form of a "compiler/post processor" which will convert the transition descriptions into 'C' code. This code will then be "linked" with user supplied functionality to compose a complete software component.

7 Analysis of Results

In various evaluation exercises similar to that summarised in Section 6, use of the SEW-OSA workbench has led to findings which can be classified under four headings: implementation results, case study results, architectural results and research results.

The **Implementation Results** obtained relate to what has actually been implemented and how well it has worked. Analysis of these results has been based on the definition and use of performance metrics. This has led to an analysis of ability to handle complexity and associated performance issues when addressing typical industrial situations.

Case Study Results include qualitative and quantitative results obtained so far based on the gathering of industrial data. Activities which have led to these results consisted of simulation runs as well as analysis of performance associated with the proposition of a re-engineered system to coordinate the SMT shop-floor. The main result obtained consists of a specification of how shop-floor re-engineering can be achieved in order to operate on a model-driven manner.

The **Architectural Results** obtained relate to architectural definitions which were produced as part of the development of SEW-OSA in the context of the "Model-Driven CIM" project. This includes a technical evaluation of the specifications of SEW-OSA, as proposed in this research. A key issue here is how much synergy can be obtained from the proposed combination of architectures.

Research Results include an examination of the outcome of the research in the light of the issues that were set out to be investigated. This includes an evaluation of the basic axioms that motivated the underlying proposition of SEW-OSA (i.e. the desirable features of an architecture). At this stage, benchmarking against systems and tools stemming from current practice is also being developed. Finally, the methodology proposed in this paper is itself being analysed in regard to its efficacy and efficiency.

Findings under these categories have demonstrated that real industrial problems can be tackled with SEW-OSA and that its inherent methods enable the very rapid-prototyping of solutions. This should significantly reduce engineering effort and timescales involved when realising and changing integrated manufacturing systems. However, current limitations on the widespread industrial use of SEW-OSA exist in respect to:

- performance limitations related to the way in which certain components of the workbench have been implemented, as well as inherent limitations of the CIM-BIOSYS integrating infrastructure. This currently places limits on the complexity of the models that can be enacted, although means of achieving re-implementation have been identified which will extend the complexity of systems that can be handled;

```
variable(ptrans_in, 1).
variable(pEA3_1, 0).
variable(pEA3_2, 0).
variable(pEA3_3, 0).
variable(pEA3_4, 0).
variable(pEA3_5, 0).
variable(pEA3_6, 0).
variable(ending_status).

transition(trans_in,(ptrans_in = 1),(pEA3_1 is pEA3_1 + 1 @ ptrans_in is ptrans_in - 1)).
transition(tEA3_2,(pEA3_1 >= 1),(pEA3_2 is pEA3_2 + 1 @ pEA3_1 is pEA3_1 - 1 @ send_app(ac, "make()
    FO-5"))).
transition(tEA3_1,(pEA3_2 >= 1 & recv_app(ac, "made()")),(pEA3_5 is pEA3_5 + 1 @ pEA3_2 is pEA3_2 - 1)).
transition(tEA3_5,(pEA3_5 >= 1),(pEA3_6 is pEA3_6 + 1 @ pEA3_5 is pEA3_5 - 1 @ send_app(ac, "obtain(n)
    FO-21"))).
transition(tEA3_6,(pEA3_6 >= 1 & recv_app(ac, "value(n,<value of n>)")),(pEA3_3 is pEA3_3 + 1 @ pEA3_6 is
    pEA3_6 - 1)).
transition(tEA3_3,(pEA3_3 >= 1 & n <= 4),(pEA3_4 is pEA3_4 + 1 @ pEA3_3 is pEA3_3 - 1 @ ending_status
    is "not_inspect")).
transition(tEA3_3,(pEA3_3 >= 1 & n > 4),(pEA3_4 is pEA3_4 + 1 @ pEA3_3 is pEA3_3 - 1 @ ending_status is
    "inspect")).
transition(trans_out,(pEA3_4 >= 1),(pEA3_4 is pEA3_4 - 1 @ send_app(ac,"finish(ending_status)") @ halt)).
```

Figure 10 - Code fragment for the enterprise activity "print"

- the unavailability of reference models to provide guides related to good practice for system designers and builders. Without these models, users of the workbench must build models and generate solutions from scratch, this requiring greater levels of experience, longer learning and development times and most likely will result in significantly less than optimal outcomes.

8 CONCLUSIONS

This paper has reported progress on a research initiative which is combining and extending reference architectures to support the life cycle of Integrated Manufacturing Enterprises. The research has led to the definition and realisation of SEW-OSA (a Systems Engineering Workbench centred on CIM-OSA), which represents a major advance on currently available formal methods and tools, in terms of the support it provides for the life cycle of integrated manufacturing systems. Primary features of SEW-OSA's model building and model enactment capabilities are described along with an outline of initial results from an industrial case study application.

It was found that SEW-OSA can support life cycle phases from 'conceptual analysis' through to 'system operation (at run-time)', this by bringing together 'process-oriented' and 'object-oriented' techniques in a consistent manner. By so doing, SEW-OSA consolidates various models of an enterprise at different levels of abstraction. Its ability to enact models, by enabling simulation, emulation and execution stages in an iterative and consistent way leads to model-driven solutions which can be advanced and extended to meet the challenges of modern customer-driven markets. Indeed, for the SMT assembly line discussed here, the approach can potentially facilitate important enhancements in system co-ordination and control, whilst positively supporting system 're-configuration and change'.

Through being coupled with other "Model-Driven CIM" methods and tools produced at the MSI Research Institute (Edwards 1995) which enact other modelling perspectives of a manufacturing enterprise, the use of SEW-OSA promises to provide a highly effective way to re-engineering systems, so that they more closely meet high-level business requirements. Indeed, in the not too distant future such requirements may themselves be expressed formally as models which can be enacted via SEW-OSA and other "Model-Driven CIM" tools.

9 REFERENCES

Aguiar, M. W. C. and Weston, R. H. (1993) "CIM-OSA and stochastic time petri nets for behavioural modelling and model handling in CIM systems design and building". Proceedings of the Institution of Mechanical Engineers Part b - Journal of Engineering Manufacture, 1993, vol. 207, no, 3, pp. 147-158, England.

Aguiar, M. W. C. (1994a) "Benchmark of SEW-OSA against other tools and methods" Working Paper. UK.

Aguiar, M. W. C.; Weston, R. H. (1994b) "The business entity of SEW-OSA - Systems Engineering Workbench centred on CIM-OSA". Proceedings of the 10th National Conference on Manufacturing Research. England, pp 245-249.

Aguiar, M. W. C. (1995a) "An approach to enacting business process models in support of the

life cycle of integrated manufacturing systems". Doctor of Philosophy Thesis, Loughborough University, England.

Aguiar, M. W. C. and Weston, R. H. (1995b) "A model-driven approach to enterprise integration". International Journal of Computer Integrated Manufacturing, England.

Childe, S.; Bennett, J.; Maull, R. (1993) "Manufacturing re-engineering around business processes". The International Conference on Managing Integrated Manufacturing - Organization, Strategy & Technology. England: KAMG - Keele University.

Clements, P., Coutts, I., Weston, R. H. (1993) "A life-cycle support environment comprising open systems manufacturing modelling methods and the CIM-BIOSYS infrastructure tools". MAPLE'93 - Symposium on Manufacturing Automation Programming Language Environments. Ottawa, Canada.

Clocksin W.F. and Mellish C.S. (1984) "Programming in Prolog", 2ed., Springer-Verlag.

Coutts, I. A. et al. (1992.) "Open Applications within Soft Integrated Manufacturing Systems", Proc. of Int. Conf. on Manufacturing Automation, Hong Kong, ICMA 92.

Edwards, J. M.; Murgatroyd, I. S.; Gilders, P. Aguiar, M. W. C.; Weston, R. H. (1995) "Methods and tools for manufacturing enterprise modelling and model enactment". Submitted to the Proceedings of the Institution of Electrical Engineers - Special issue on Science, Measurement and Technology. England.

ESPRIT/AMICE. (1993a) "CIM-OSA Architecture Description", AD 1.0. 2. ed 1993.

ESPRIT/AMICE. (1993b) "CIM-OSA formal reference base".

ESPRIT/VOICE. (1995) "Validation of CIM-OSA (Open Systems Architecture) - A joint ESPRIT projects report".

Weston, R. H., Hodgson, A., Coutts, I. A., Murgatroyd, I. S. and Gascoigne, J. D. (1990) "Highly extendable CIM systems based on an integration platform". Proceedings of CIMCON'90, NIST, USA.

10 BIOGRAPHY

M W Aguiar: Seven years as managing director in charge of the Integrated Automation Division of a Research and Development Institute of the Federal University of Santa Caterina/ Brazil. Three years as a member of the MSI Research Institute, involved in the Model-Driven CIM project, with particular responsibility for the conception, realisation, application and evaluation of SEW-OSA.

I A Coutts: Two years at Marconi Research as a research scientist, working on industrial assembly automation and robotics projects. Seven years as a member of the Loughborough SI group. Particular responsibilities have included work on the group's software integration platform.

Prof. RH Weston: Over fifteen years experience of research in areas of machine control and systems integration. Supervisor for over fifty RAs and PhD students during that time. Author (or joint author) of over 200 refereed journal and conference publications in the field.

Chairman of member of various national (PSRC, DTI, LINK and BS) committees (ISO, IFAC, IFIP) serving the area and member of the board of five journals.