COMPUTER-AIDED PRODUCTION MANAGEMENT:
FUTURE DEVELOPMENTS IN AN INTEGRATED ENVIRONMENT

Allan Hodgson, MBPICS, MSI Research Institute

SUMMARY

There is an increasing demand from users of manufacturing software for system-wide solutions, rather than for individual software packages to solve specific problems. However, system-wide computer integrated solutions have not generally performed well to-date. Success may come from the application of several new information technology developments which will have a significant effect on the way that manufacturing-related software can be developed, marketed and implemented.

This article discusses the problems associated with the integration of manufacturing software systems, not just within, but also across functional areas. Emerging integration solutions are described and compared with current approaches and the associated roles of developing database and product representation standards are considered. The implications for computer aided production management (CAPM) are presented and some potential problems are highlighted.

INTRODUCTION

Information systems managers are discovering that the key factor to success is the degree to which their software packages and any associated bespoke systems, work together. In an earlier (longer lead time) era of fewer, simpler software systems, manual or batch transfer of data was acceptable. This is no longer adequate for most businesses. Software houses are now finding that their potential customers are interested in achieving an effective overall system, not just acquiring another item of proprietary software, however capable. Failure to respond to this requirement will lead to a loss of software sales.

Providing integrated solutions is currently a time-consuming and costly business. An increasing proportion of time is spent on meeting the software interfacing or integration requirements. This time could be used more productively to meet the functional requirements of user companies.

Computer-aided production management (CAPM) software is a central part of a manufacturing enterprise's information systems. A typical CAPM system has links (either manual or automated) to sales, purchasing, design, process planning, costing, finite scheduling, personnel, accounts and other functions. It therefore presents a major integration problem due to the above large number of links. A CAPM system with poor provisions for integration will severely hamper any further work in this direction.

TRENDS IN MANUFACTURING INTEGRATION

The 'first wave' of integration, mainly during the 70's and 80's, was largely concerned with integrating software within local functional areas. The clearest example of this is the typical MRP/MRPII package - master production scheduling, material requirements planning, capacity planning, works order progressing, inventory control, standard and actual costing, etc., all use the same database. Changes made to data by one sub-function can result in changes to the outputs of other sub-functions next time they run. In reality, computer integration of such a system exists only via the database; people operate the system and cause it to exhibit the required 'behaviour', ie. to carry out activities at the appropriate times and in the appropriate sequences.

The 'second wave' of integration, is now taking place across traditional functional boundaries. It is necessitated by the increasing interoperability demands placed on manufacturing enterprises as they attempt to improve their flexibility and response times. Approaches such as concurrent engineering (CE), computer-supported co-operative working (CSCW) and computer-integrated manufacturing (CIM) all need to interact and share data across conventional functional boundaries. With lead time targets reduced in some cases from weeks to days, hours, or less, it has been necessary to automate routine human-based data transfer and intervention in many software systems.

Examples of activities requiring interaction or data sharing include finite scheduling, tool changing, NC programme downloading, inspection and test. This has necessitated linking or integrating diverse software and hardware systems. The resulting communications have not only been via the underlying databases, but have required real-time responses involving interacting software applications.

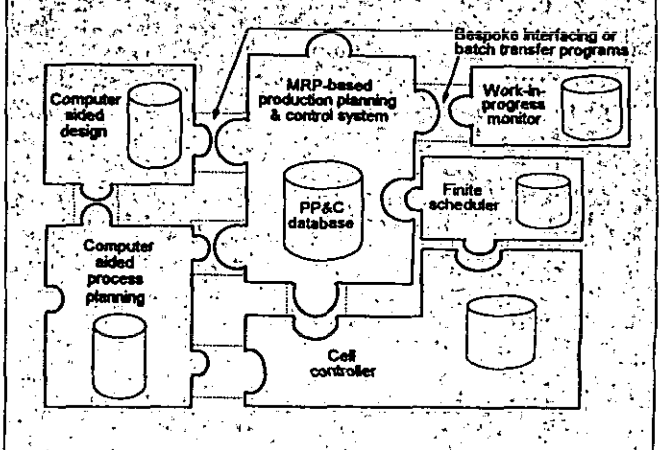
CONVENTIONAL APPROACHES TO INTEGRATION

Bespoke solutions - the ill-fitting jigsaw

Most computer-integrated systems to-date have been put together from diverse items of hardware and software. They utilise a different bespoke software interface between each pair of communicating programmes. These 'hard-integrated' (or 'pairwise integrated') systems, with many such interfaces, are typically difficult to debug and maintain. The result is expensive inflexible systems of limited scope which remain dependent on the skill and experience of the systems engineer(s) who originally implemented them. Changes to such systems often result in a rewrite of much of the bespoke code.

The reasons for the difficulties experienced with the above approach relate in part to the hard-coding of integration -related knowledge (communication paths and formats, physical names and locations of files and data fields, etc.) into each of these interface programmes (see Figure 1). Any change or addition to hardware or software can invalidate many of these programmes.

FIGURE 1
'HARD INTEGRATION'



Turnkey solutions - The pieces fit, but it's not the right jigsaw

An alternative approach, adopted by some suppliers, is to develop wide-scale software/hardware solutions which, being under the control of a single supplier, are compatible and thereby integratable. This approach may be successful in the short term, but does not provide the customer with choice.

It is unlikely that a user will find a single supplier producing the 'best' software for every application area (MRP, finite scheduling, CAD, CAPP, etc.). This approach therefore involves compromises in the user's functional performance. Also, few users would wish to find themselves reliant on a single software supplier for the foreseeable future. Eventually, when the mismatch between company requirements and actual software functionality became too great, it would be necessary to start again.

INTEGRATING INFRASTRUCTURES - A NEW APPROACH TO INTEGRATION

As described earlier, many of the problems associated with integration are caused by the ad-hoc, localised approach taken to building the system. New approaches to integration enable the elimination of the above scattered integration related software code (interfaces, etc). Instead, that integration knowledge is formally structured and located so as to be accessible to a separate block of software, the integrating infrastructure (see Figure 2).

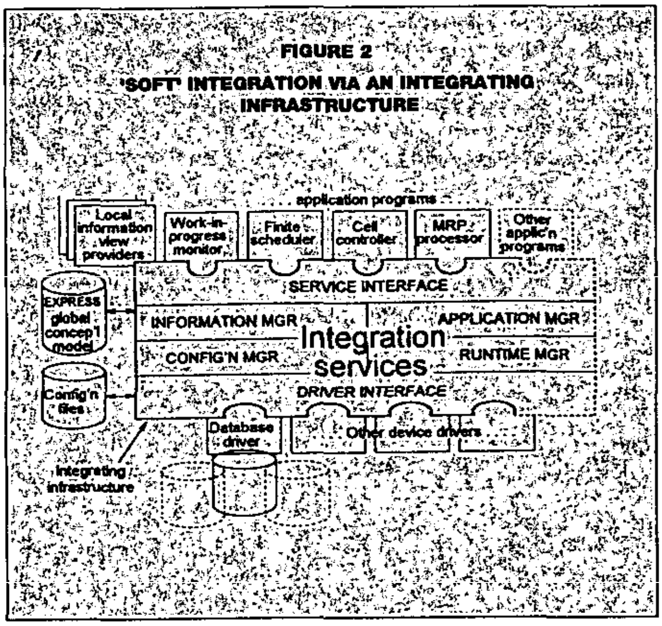


FIGURE 2
'SOFT' INTEGRATION VIA AN INTEGRATING INFRASTRUCTURE

This integrating infrastructure, potentially distributed across a number of machines, carries out integration-related activities on behalf of the various functional pieces of software in the manufacturing organisation. Such activities include passing messages to other applications, accessing and updating data stores and selecting the underlying communication media and paths.

Because integration-related information is now stored explicitly as formal parameters at a specific location, changes to a complex system require relatively small changes to this information; this contrasts with the former situation (as exemplified in Figure 1) where such information was hard-coded, fragmented and scattered.

Figure 2 provides an outline of the services associated with a typical integrating infrastructure. The service interface to the information and application managers provides a uniform 'socket' for the various software applications to plug into.

When a software application programme needs to contact another application, the application manager of the integrating infrastructure locates the other application, which may run on any compatible computer in the system. If necessary, it will activate this other software application in order for it to be able to interact. Adding new (conforming) software is relatively easy; the integrating infrastructure must be supplied with appropriate details and will then be able to activate it as necessary.

When a software application programme needs to access or update information, the information manager of the integrating infrastructure accesses an appropriate 'view provider' which represents how the application 'sees' the data structures. This 'view' is mapped to a 'global view' and on to a 'fragmented (or internal) view', which represents how the underlying data is actually stored. This complex sounding approach, the three schema architecture, isolates the majority of application programmes from changes to the database structures.

The underlying communications manager is invisible to the application software programmes and users. Depending on the hardware links (local area networks, serial links, etc.) and the performance requirements, appropriate communication routes can be defined for each application.

A typical integrating infrastructure will have a number of further software tools and functions to enable the (human) systems manager to monitor and improve the performance of complicated integrated software systems.

With the integrating infrastructural approach, each application only needs a single application interface and information interface to gain access to the 'software bus' of the integrating infrastructure (via the service interface). Contrast this with the earlier pairwise integration approach, where a 'popular' software application or suite (eg. CAPM) may need interfaces to many other software applications.

Figure 3 compares the relative systems engineering complexities and integration costs of the two approaches. In particular, the costs of small, incremental changes to complex systems are much reduced in the integration infrastructural approach.

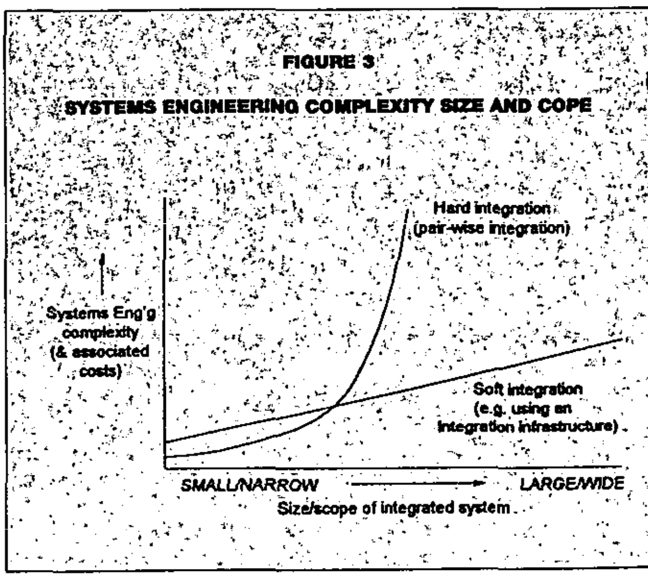


FIGURE 3
SYSTEMS ENGINEERING COMPLEXITY SIZE AND COPE

THE ROLE OF STANDARDS

Standards will play a vital role in the implementation of effective integrating infrastructures. In particular, the increasing popularity of SQL and the emergence of the STEP standard (Standard for the Exchange of product Model data) will pave the way to their widespread acceptance.

The SQL database management system (DBMS) query language is now incorporated in an increasing proportion of advanced CAPM systems. Integrating infrastructures typically provide interfaces to SQL-compliant databases, allowing general access to the information incorporated in them.

The STEP standard, ISO 10303 [1], provides a means of interchanging product information, including shape, bills-of-material, processes, orders, etc. Although the primary aim is that of an interchange standard for the manufacturing sector, STEP is likely to be used increasingly as the basis for internal data constructs and formats for software products. Associated with STEP is the information modelling language EXPRESS, in which the STEP structures are defined. STEP is an extremely complex standard which runs to many volumes. A number of 'standard' subsets are being defined, which will enable internationally recognised levels of conformance to be achieved by users.

The standards-related work of the MANDATE (MANufacturing DATa Exchange) initiative [2] is of more direct interest to CAPM vendors and users. This seeks specifically to produce a uniform manufacturing language associated with a set of manufacturing models or structures. Future CAPM systems based on this work will have a built-in advantage. It is intended that the MANDATE work will be closely compatible with the STEP standard; it is therefore represented using the EXPRESS language.

EXPRESS-to-SQL compilers are now available; these enable information structures described in the EXPRESS language (including those of STEP and MANDATE) to be translated into corresponding SQL data structures.

The US Department of Defence has placed requirements with regard to STEP conformance on its industrial contractors and sub-contractors. The standard is therefore being widely implemented in the US and will rapidly increase in importance in the coming years. There is a danger that UK software houses and users will be left behind during the development of this standard and subsequently lose business internationally as STEP is used as a selling and contract-bidding weapon. The low level of interest in STEP shown by UK industry is discussed further in [3].

It is the combination of the above standards and the earlier described integrating infrastructures that provides the greatest potential advance in software systems. The common language and set of generic structures provided by these standards enables us to design information systems based on customising the structures or models that they define, ie. to carry out top-down system design. The availability of an integrating infrastructure gives us a stable implementation environment to aim for. In total, they provide a means for model-based design and implementation of CAPM and other manufacturing information systems.

There is increasing activity in the area of object-oriented systems. In particular, there is a major initiative, CORBA (Common Object Request Broker Architecture) to develop a standard for object-oriented systems to enable the exchange and re-use of software objects. CORBA is likely to have profound effects on the way software systems are developed, but it is difficult at present to ascertain how it will relate to the STEP and MANDATE standards. A less radical object-oriented development, the object-relational database management system, offers a more evolutionary approach; most object-relational systems offer SQL compatibility plus object-oriented features.

WHERE ARE WE NOW?

There are several proprietary integrating infrastructures now available. Whereas these do enable some 'mix and match' of systems, they typically constrain the choice of hardware and/or software supplier. IBM's DAE/Plantworks, which runs over the company's OS/2 operating system, is an example of such an infrastructure.

The Loughborough University Manufacturing Systems Integration (MSI) Research Institute, of which the author is a member, has developed the CIM-BIOSYS (CIM - Building Integrated Open SYStems) integrating infrastructure [4]. CIM-BIOSYS has been implemented at a number of research sites worldwide. In addition, it has been installed to control a number of cells at a major UK electronics manufacturing site; the ease with which the system has incorporated additional manufacturing cells and other changes has demonstrated the effectiveness of the integrating infrastructure principle.

The Loughborough University MSI Research Institute has recently completed a study of means to achieve 'model-driven CIM' [5]. The aim of this research was to develop formal methods to create and test models (or representations) of software systems and to provide a software environment in which to generate the code needed to implement such systems for a computer integrated manufacturing enterprise. At the present stage of the research, it is possible to define models of required software applications and to test their performance; the required programme code can then be produced, part automatically, part manually, to run over the CIM-BIOSYS integrating infrastructure.

In addition to the above application modelling research, EXPRESS-to-SQL compilers have also been developed at the MSI Research Institute [6].

Functional demonstrator systems have been built over the CIM-BIOSYS integrating infrastructure. These systems incorporate both proprietary and in-house CAPM database structures, finite scheduling systems and dynamic costing systems. They have been used to assess the effectiveness of the integrating infrastructural approach.

ACHIEVING A FIT BETWEEN REQUIREMENTS AND AVAILABLE SOFTWARE

The availability of an integrating infrastructure removes the integration-related limitations on the number of individual (conformant) software application programmes which could, practicably, be incorporated into a system. This will, in theory, enable the selection (or development) of such software applications based on their appropriateness to the company's business, rather than based on their compatibility with each other.

A much closer fit between company requirements and available software solutions should be achievable, with less information and function duplication. A conceptual comparison is illustrated in Figure 4.

FIGURE 4
SMALLER 'BLOCKS' ENABLE BETTER FIT