

An Open CNC Interface for Grinding Machines

*W Brian Rowe (1), **C Statham, **J W Liverton, **J L Moruzzi

*School of Engineering, John Moores University, Liverpool, L3 3AF, England, UK

**Jones & Shipman plc, Leicester, LE3 2LF, England, UK

Tel: +44-116-289-6222, Fax: +44-116-233-6974

Abstract

This paper reviews traditional and modern approaches to control systems architecture and CNC software design. A software framework is proposed, known as the open CNC interface (OCI), as a possible solution to the problem of providing portable CNC application software. This approach allows portable CNC software to be developed and re-used over a number of open CNCs. The OCI is characterized by a layered approach in which object oriented techniques are used to provide a hierarchical structure and to allow event driven programming. Implementation has been demonstrated on two models in a new range of CNC grinding machines.

Key Words: CNC, grinding, open systems

1 INTRODUCTION

In grinding, the computer numerical control (CNC) allows a set of parameter based instructions known as a part program to be entered for a series of grinding operations on a particular part or batch of parts to be produced. A CNC grinding machine is usually purchased from a machine tool builder complete with a CNC. The grinding machine builder usually purchases the CNC from a CNC manufacturer such as Siemens, Num, Fanuc or Allen Bradley. Component manufacturers sometimes wish to specify the type of CNC supplied when purchasing a new machine tool, possibly to maintain consistency with other machine tools in the plant, to reduce learning time and improve technical support. Component manufacturers may also be influenced by price and technical specification.

A request from a component manufacturer to specify the type of CNC creates a problem for the machine builder. There is a substantial amount of work involved in engineering a grinding machine with a CNC and providing customized software for the particular machine functions. If this work has to be repeated for several types of CNC each time a new machine is developed, the engineering costs are greatly increased. This results in increased lead-time for machine development. The problem is increased each time it is wished to improve the grinding software. As machine builders respond to customer demands in a more flexible manner, there is increased demand for product re-engineering. The Open CNC Interface (OCI) is designed to reduce re-engineering requirements by allowing software re-use and ease of customization with different CNCs and types of grinding machine.

Figure 1 shows typical CNC modules often written in different software languages

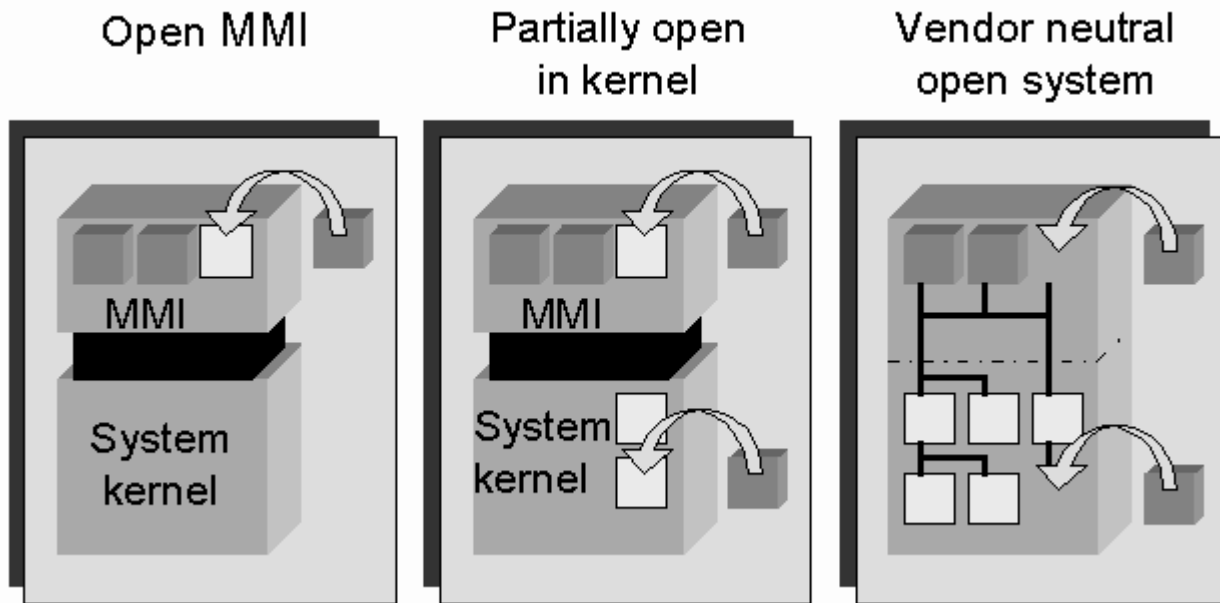


Figure 1 CNC Architecture

2. CNC ARCHITECTURE

Man-Machine Interface (MMI) the MMI regulates interactions between the machine and the human operator or a computer integrated manufacturing network. The MMI provides utilities for part programming as well as management of the machine and numerical control resources.

Axis Control Module The axis control for a typical cylindrical grinding machine provides for motion of the table traverse, the table infeed, and the workhead rotation. The axis control system issues commands to the servo drives and monitors positional feedback by the use of linear scales and rotary encoders.

Logic Control Module The logic control module controls switching of the actuators and ancillary equipment. The logic control is responsible for input of data and signals from machine sensors. Manual functions and operator push buttons may also be controlled and monitored by the logic control system. Configuration of the logic system is often by means of a ladder logic program.

Configuration Module The configuration module allows the machine builder to customize the functionality of the CNC software and hardware to match the requirements of the particular machine and process. The type of drives and servomotors used are characterized by adjustable parameters within the configuration module. Other parameters within the module allow selection of default speeds and feeds, machine capacity, axis nomenclature, and optional control functions such as tool radius compensation.

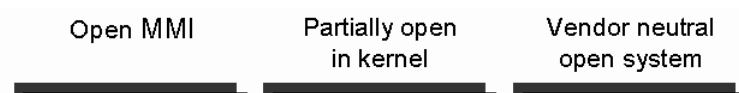


Figure 2 Classes of open CNC

Program Module The program module provides facilities to edit, store, retrieve and execute user programs. The module usually contains a program interpreter responsible for deciphering the user program and initializing specific functions such as requests for axis motion or mathematical evaluations. Two types of program may be executed; part programs for grinding specific part and machine builder sub-programs for functions or tasks. Machine builder sub-programs are often compiled outside of the CNC environment into code that can be directly executed by the program module without interpretation. The machine builder prior to delivery to the customer loads the compiled sub-programs into the CNC. This type of sub-programming is often referred to as canned cycle or macro cycle programming. Typical macro cycles for a cylindrical grinding machine include plunge grinding, shoulder grinding and traverse grinding.

System Kernel The operating system of the CNC lies within the system kernel. The kernel is responsible for ensuring each module in the CNC performs the required tasks. The kernel passes data and information between the modules, and allocates resources such as system memory and processing time. The kernel takes control when a module fails or detects an error.

3 OPEN CNC DEVELOPMENTS

CNC manufacturers have traditionally provided complete CNC solutions. The protocols and networks within a closed CNC are proprietary and remain private. Pritschow [1] comments that the lack of openness within CNC systems means that extensions and modifications are only possible by the CNC manufacturer.

The OSACA project for the introduction of open system architecture for controls and automation aims to specify a system architecture which is manufacturer independent and characterized by interoperability, portability, scalability, and interchangeability.

The trend for more versatile machine tools able to produce a variety of components means that a strongly modular and highly configurable concept for hardware and software is required in order to achieve a fast and cost-effective adaptation to new problems. Closed CNC systems mean a high dependency of the machine tool builder on the control system manufacturer and limited possibilities for the integration of machine builder know-how. Pritschow suggests that OSACA will result in shortening the development time and increased flexibility for the adaptation of controls to the specific demands of machine tools and manufacturing cells. He also suggests that the costs of development, adaptation, commissioning, training, documentation and maintenance will be reduced.

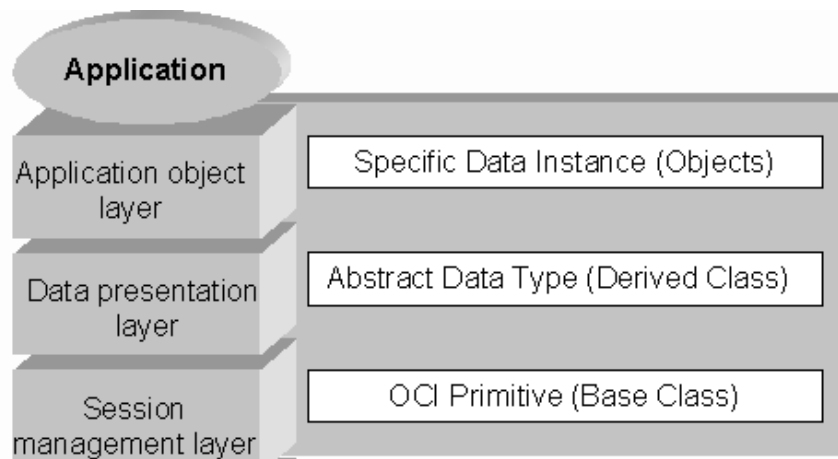


Figure 3 The Open CNC Interface (OCI) Model

CNC systems already exist that offer a level of openness. However, a major aim of this investigation was to provide a software framework that can be used with a variety of open control systems regardless of their type and level of openness. The OSACA project has identified three classifications of open control system, figure 2. These include those which have openness restricted to the man machine interface part of the control, those which offer restricted openness within the system kernel, and those which offer a completely vendor-neutral topology throughout. The latter of these is a truly open control system where interchangeability, scalability, portability and interoperability can be maintained.

Open systems are often seen as a way to incorporate additional intelligence into the system. Yellowley and Pottier [2] proposed a multi-processor arrangement for open systems where the tasks were assigned to processors suited to the particular application. Additional processors to perform other tasks could be included as and when required.

Mitsubishi [3] proposed an open architecture CNC machining system, which maintains a machining state through the use of multi-axis force information. The CNC system could prevent chatter and compensate for thermal deformation on a lathe by monitoring strain transducers connected to an external real-time control system. The external system processed the information and then sent speed and feed override values as well as axis shift values to the open architecture CNC through a standard Ethernet network connection.

Yamazaki [4], et al. proposed a new control system for high performance CNC machines. The features claimed were that it was transportable, transplantable, revisable, user-configurable, and evolving.

Altintas [5] implemented adaptive control features using a machine fitted with an open architecture CNC, and Koren [6], discussed the relationship between control system architecture and machining cycle times.

Tanenbaum [7] suggested that in the future, almost all network architectures will be standardized, and computers from one vendor will be able to communicate with computers from another vendor. To this end, the ISO open systems interconnection (OSI) reference was defined as a first step towards international standardization of the various communication protocols [8]. Most computer networks are organized as a series of layers, each one built upon its predecessor. Layering is used to allow the problem of open systems interconnection to be decomposed into independent smaller sub-systems. The purpose of each layer is to offer defined services to the higher layers, shielding those layers from the details of how the offered services are implemented. The advantages of the OSI approach are that the entire process becomes more manageable, the implementation of each layer can change from system to system although the services provided remain the same so that differing computers on the same network can communicate.

A similar approach to the OSI reference model is needed for CNCs. The software framework must be able to 'mutate' to suit the CNC hardware, and yet provide a constant set of services so that application software is undisturbed by such changes.

4 THE OCI FRAMEWORK

A framework was developed in which portable application software for any type of open CNC could be written. The framework was based on a model of an open CNC interface (OCI) which can be implemented on any open CNC system regardless of the hardware or system software in use. Key elements of the OCI are a layered approach that gives abstraction and encapsulation of application services, with a view to providing a modular and hierarchical framework for application programs. The OCI was based on an object-oriented adaptation of the OSI reference model.

Since the prime objective was a software platform for use with a variety of open control systems, it was decided that the OCI model should be concerned only with providing user oriented services. These services were made independent of the CNC. For this reason the OCI model was based on the upper four layers found in the OSI reference model. Figure 3 shows the

outline of the OCI model. Like the OSI model, the OCI model itself is not intended to be a network architecture and thus does not specify the exact services and protocols to be used in each layer. The OCI model was developed to simply identify what each layer should do. However, internal functions and standards were defined for each of the layers as part of this development. The following is a brief description of each layer of the OCI model.

Application Object Layer

The application object layer of the OCI model provides the high level functions and services required by the application programs. These high level functions include passing data to and from the CNC such as part program transfer, program block transfer, axis parameter data exchange, macro parameter data exchange, logic parameter data exchange, and system parameter data exchange. These functions and services are encapsulated by a collection of classes that allow specific data objects to be created by the application program. It was decided that it would be these specific data objects, stored in the application object layer, which would be passed across the OCI.

Data Presentation Layer

The data presentation layer holds the definition of the classes that are used by the application object layer. This layer of the model defines the type of data that can be passed across the OCI. Any data conversions required are carried out in this layer. Unlike the application object layer which maps objects of data to actual parameters in the CNC hardware, the data presentation layer maps the type of data to the correct application program interface (API) function in the information transport layer. The data presentation layer does not hold any data; moreover, its purpose is to describe the data that can be passed.

Session Management Layer

The session management layer contains the OCI base class from which all other data types in the model are inherited. The OCI base class controls the way in which information is passed across the OCI. The data may exist concurrently in two locations; a local value within the application program and an external value within the CNC. The passing of data is either direct or indirect depending upon the type and availability of the data. It is the task of the Session Management layer to determine how, where and when the passing of data occurs.

Information Transport Layer

The information transport layer is the lowest layer presented in the OCI. It is the task of the information transport layer to provide the low-level connectivity to the application program interface functions that are provided by the CNC manufacturer. The main concern with the implementation of the information transport layer is that it can be easily be modified to accommodate new or different application program interface functions when porting to other CNC systems. The layer provides a dynamic binding of API functions so that they can be changed without modification to the upper layers of the model. This is achieved by using a dynamic-link library.

Dynamic-Link Library

A dynamic-link library (DLL) is an executable module that contains code or resources that are used and shared by other DLLs or applications. Unlike functions and services provided by standard library modules that are linked into executable application code at compile time (statically linked), DLL functions and services reside in a separate module and are made available to the application at run time (dynamically linked). DLLs also provide the ability for multiple applications to share a single copy of a routine they have in common. The DLL module must be available to the application at run time. When an application program is loaded into the system memory, the application dynamically links the services and functions used in the program to their entry points in the DLL. Application programs written in one language can also often use DLLs that

were written in another. The advantage of using a dynamically linked library to implement the information transport layer is that new functionality can be provided, without the need to adjust or recompile any of the upper layers.

At the time when the OSI model was being developed, high-level computer languages did not easily support the notion of interrupt or event driven programming. The semantic model of an interrupt driven system is totally at variance with traditional structured programming techniques used to implement the first versions of the OSI model. This problem was avoided in the development of the OCI by the use of object oriented design (OOD) which places a high importance on message passing and time-space semantics. In OOD, messages are passed between objects by invocation of a function or procedure (often called methods) of one object by another object. Object oriented languages allow for systems that have multiple threads of control, i.e., systems in which two or more concurrent processes may be executing at any one time, to synchronize message passing between objects. This allows event driven systems to be more easily developed where multiple threads of control are available (concurrent processing is generally supported by the operating system upon which modern application programs are implemented). The OCI uses message passing between objects to implement the functionality required within the session management layer of the model.

An important feature of the OCI is that it is able to access the right CNC data at the right time. The session management layer is responsible for this function. However, it is important to note that this layer is not concerned with the type of data that is passed across the OCI. For this reason, activation of application program interface (API) service calls in the transport layer must actually be handled by the data presentation layer (the next layer up). The session management layer provides a mechanism to indicate to the data presentation layer, when calling of the services must take place, but does not actually call the services. Calling is achieved by use of object oriented virtual methods.

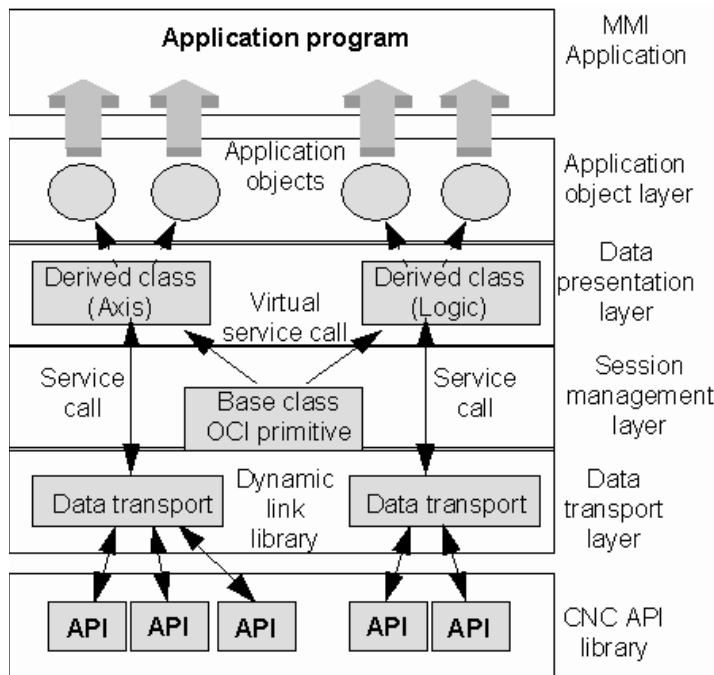


Figure 4 Virtual Service Calls in the OCI

Figure 4 shows how virtual methods are used in the OCI to activate service calls and allow flow of data to and from the CNC and application program.

Virtual methods in object oriented design and programming make it possible for procedures or functions with the same name to be implemented in different ways within a hierarchy of object types. A virtual method is a procedure or function of a class that can be redefined by one or more of its subclasses to give a polymorphic behaviour. When a class method is declared as

virtual, the implementation of the method may be overridden by a method with the same name in a descendant class. An object type can override (redefine) any of the methods it inherits from its ancestors. The scope of an override method extends over all of the descendants of the defining object, or until the method identifier is redefined. The compiler at run-time resolves virtual methods; this process is known in object oriented programming as late binding.

5 SYSTEM IMPLEMENTATION

Implementation of the OCI model was achieved using a combination of object-oriented Pascal and C++ programming languages. Using the Borland Delphi programming system it was possible to encapsulate the OCI in a number of customized components that could be added to the standard integrated development environment (IDE). By using simple point-and-click operations it was possible to quickly develop CNC application programs that could be re-used over a number of open CNC systems. The model as implemented contains components to access and manipulate data related to axis, logic, macro, offset, program, alarm, parameter and analogue-to-digital information. The implementation of the model is extendable to include other types of CNC data as and when required. An enhanced graphical user interface (GUI) was developed using the OCI and is used on the latest models of cylindrical and surface grinding machines. The transport layer of the OCI model was implemented to accommodate the GE Fanuc 180B and 210i open CNC systems.

The OCI including a range of machine builder objects for set-up and control of grinding processes was first implemented on the Jones & Shipman Supromat & Dominator grinding machines. Figure 5 shows a screen display of the machine status page from the graphical interface as used in cylindrical grinding.

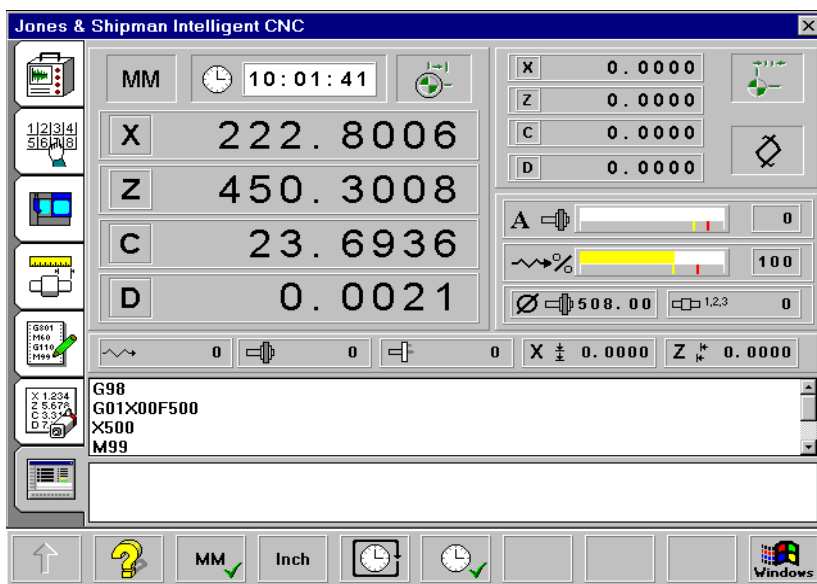


Figure 5. Machine status page

6 ADAPTIVE CONTROL FEATURES

The OCI makes it easier to develop and incorporate advanced process technology into the control system. Intelligent algorithms have been shown to increase productivity for a specified accuracy by optimization of feedrate, control of dwell time (spark-out) and by control of target position [Rowe, 9].

Using an object oriented generic intelligent control system, it has been demonstrated that adaptive control can be applied to a range of CNCs [Rowe, 10]. However, the earlier system developed on a personal computer was limited in operation by being constrained to interact within the constraints of previously developed machine builder software

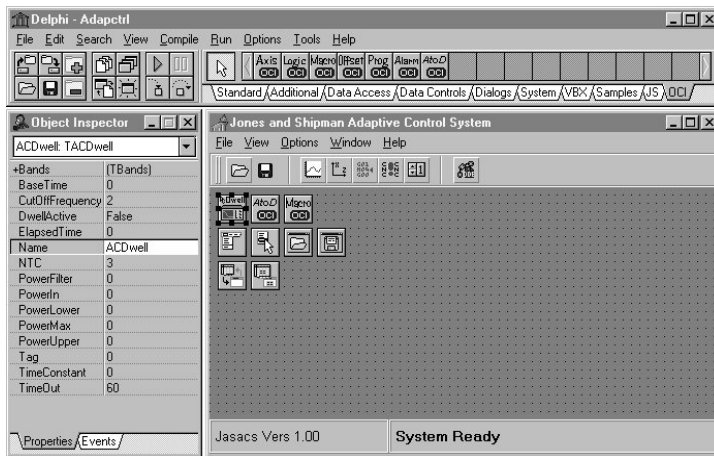


Figure 6 Adaptive Control objects in a Delphi Application.

Figure 6 shows a screen display for an adaptive control application developed in the OCI using the Borland Delphi system. This application uses a macro component and an analogue-to-digital component from the OCI palette of components.

The macro component is used by the adaptive control application to access process parameters during the grinding cycle. The analogue-to-digital component is used to monitor the grinding power level in order for optimization to be achieved.

In this development, the generic intelligent control system is integrated as part of the machine builder software. . This has a great advantage for speed of operation as decisions can be taken based on real-time process information. For example, it is possible to measure process time constant within the grinding cycle and immediately adjust the dwell period according to the current cutting condition of the grinding wheel.

7 SUMMARY AND CONCLUSIONS

The Open CNC Interface represents a way to build on recent developments in open CNCs. The OCI allows the latest process technology to be

incorporated into the machine builder's proprietary software. The OCI facilitates re-use of the proprietary software over a range of CNCs for a particular grinding machine or the extension of software application to a range of machines and processes. Advantages include improved software quality and reduced lead-time for new machine development.

The layered structure of the OCI is based on the principles of the ISO OSI reference model. Implementation of a layered structure is achieved using object-oriented design, which allows event driven programming.

The OCI is used for two models of open CNC for a range of cylindrical and surface grinding machines. The ability to incorporate adaptive control techniques as a selectable feature has been demonstrated for cylindrical grinding.

REFERENCES

- [1] Pritschow, G., Daniel, C., Junghans, G., Sperling, W., 1993, Open System Controllers, CIRP Annals, 42,1, 449-452
- [2] Yellowley, I., Poitier, P., 1994, The Integration of Process and Geometry within an Open Architecture Controller, J. Mach. Tools and Manufacture, 34,2, 277-293