An autonomous agent for on-machine acceptance of machined components

Carmen M. Pancerella
Andrew J. Hazelton
Sandia National Laboratories, Livermore, CA†

H. Robert Frost
Center for Design Research, Stanford University, Stanford, CA‡

ABSTRACT

In recent years, manufacturers of high precision mechanical parts have been required to produce increasingly complex designs, in smaller lot sizes, with improved quality. These requirements demand lower process costs, shorter development cycles and more accurate manufacturing technologies. To meet these demands, manufacturers are attempting to both improve process quality and provide better CAD/CAM integration. The technique of on-machine acceptance provides one mechanism for improving the part inspection and verification process. This approach allows one machine and one process capability model to be used for both fabrication and inspection, reducing capital cost and overall cycle time. However, the on-machine acceptance technique possesses greater potential than as simply an alternative mechanism for verifying part geometry. If the inspection capability information generated by on-machine acceptance processes can be made available to designers, it can be used to create a design-for-inspectability environment and help realize the benefits of concurrent engineering. This paper proposes a novel architecture which integrates on-machine acceptance with an agent-based concurrent design environment, for reducing both the cost and production time for high quality, small lot size, mechanical parts. This work has focused on the production of stainless steel pressure vessels at the Integrated Manufacturing Technology Laboratory (IMTL) manufacturing cell, located at Sandia National Laboratories, California.

Keywords: agents, concurrent engineering, on-machine inspection, manufacturing, design-for-inspectability, STEP, EXPRESS.

1. INTRODUCTION

An increasingly competitive world economy combined with more sophisticated engineering applications has forced manufacturers to rethink the way products are designed, fabricated and inspected. Over the past decade, the most significant improvements to the manufacturing process have been obtained through application of computer-integrated manufacturing (CIM) concepts and mechanisms. CIM covers a wide range of applications, including CAD (computer-aided design), CAPP (computer-aided process planning), CAM (computer-aided machining) and the electronic exchange of product and process data. Initial CIM applications focused on specific activities (e.g., design, fabrication, etc.) within the larger product development process. This approach has resulted in so-called "islands of automation", multiple heterogeneous systems which operate on different platforms and view the development process from different perspectives and levels of abstraction. Future advances in manufacturing will require the integration of the multiple CIM systems involved in the development cycle at both the functional (coordinated activities) and knowledge levels (common models of product data). This problem is being partially addressed by the discipline of concurrent engineering, which focuses specifically on the integration of design and manufacturing.

This paper focuses on a subset of the concurrent engineering problem: integration of CAD and CAI (computer-aided inspection). Specifically, the architecture presented in this paper represents a methodology for making the process capability information generated by an on-machine inspection process available to a designer. This inspection capability information can be used by a designer to evaluate the inspectability of a part during the design process. This concurrent design environment will help a designer to only incorporate those features in the design which can be verified to the required tolerance by the anticipated machine and process. Additionally, because on-machine inspection uses the same machine to both fabricate and inspect a part, the inspection capability model can be extended without great difficulty to

† This work was supported by Sandia Corporation under Contract No. DE-AC04-94-AL85000 with the U.S. Department of Energy.

‡ This work was supported by Stanford University under Contract No. N00014-92-J1833 with the U.S. Department of Navy.

DISCLAIMER

DISCLAIMER

create a model for fabrication capability. Software agents are proposed for encapsulating the design and inspection processes and providing an interface for the exchange of formal, semantically unambiguous product information. The proposed architecture will support a product development cycle comprised of a concurrent design phase followed by fabrication and on-machine inspection.

During concurrent design, a design agent, representing the designer and CAD system, interacts, via a cell manager agent, with a machine agent representing a specific fabrication and on-machine inspection process. All information exchange between the agents will be done using a common formal language comprising portions of the PDES/STEP[1] (Product Data Exchange using STEP/Standard for the Exchange of Product model data) standard. The design agent will first receive a set of feasible fabrication features from the machine agent, and, for each feature to be used during the design process, constraints on nominal and variational feature geometry. These constraints represent, for each feature type, the range of geometric variation which can be inspected on the targeted machine. During the design process, each added feature is checked against the constraints for that feature class.

Completed designs are submitted to the cell manager, which routes the part geometry information to an NC programming agent that produces the NC program and on-machine inspection probe paths. The cell manager agent sends the corresponding NC codes to a machining agent, which monitors the fabrication and complete on-machine inspection. Upon receiving the inspection data (probe points), the on-machine acceptance agent evaluates the data and generates statistically valid certification of reliability. This certification and the accompanying data will be stored in a database and will be used to provide an empirical foundation for the process capability models from which feature constraints are generated. The customer will receive both the part and certification report.

This paper is specifically concerned with the design phase and the mechanisms necessary to exchange process capability data between a manufacturing service and the designer. How the manufacturing and inspection process plans are generated and scheduled will not be discussed in detail. The remainder of this paper presents the technical background and related research (Section 2), a discussion of the on-machine inspection process (Section 3), the proposed agent-based concurrent design architecture (Section 4), and conclusions and suggestions for future work (Section 5).

2. BACKGROUND AND RELATED RESEARCH

The material in this paper draws upon the fields of inspection process design and modeling, concurrent engineering, knowledge representation/sharing and multi-agent systems.

2.1. Inspection process design and modeling

Current manufacturing practice for small lot manufacturing of mechanical components is comprised of design and pre-production activities, machining of the component, and final inspection usually performed by a coordinate measuring machine (CMM) after the manufacturing process is complete. While statistical process control (SPC) techniques are often used to certify part quality for large lots,[2] small lot production usually requires 100% off-line inspection. In the past few years, some manufacturers have begun to inspect the parts on the machine. Although this has been applied successfully in some cases,[3] the method is criticized because errors made in machining the part due to machine geometry errors are repeated, one-for-one, when inspecting the part.

An alternate approach is to certify the process by certifying the basic accuracy of the machine. This approach is based on the deterministic manufacturing principle: "machine tools obey cause and effect relationships that are within our ability to understand and control and there is nothing random or probabilistic about their behavior."[4] Using this approach, the process engineer assesses the accuracy of individual elements of the machine tool, then combines the accuracies to determine the machine's ability to produce a good part. The traditional method for combining these error terms is the error budget. With this method, individual machine error components are identified, estimates or measurements of their magnitude are made, and a suitable combination rule is applied to determine the total machine error.[5] This method has been applied successfully to precision machine design for a number of years,[6] but has not been verified for process certification.

Recently, techniques have been developed to better certify parts while still on the manufacturing machine.[7] Termed *On-Machine Acceptance (OMA)*, these techniques provide statistical guarantees of part accuracy, eliminating the need for post-process inspection. There is a need, however, to consider this acceptance process part of the product design if the process is to be used to full advantage.

2.2. Concurrent engineering

The part development cycle can be modeled by the processes of design, fabrication and inspection. In a conventional development cycle, these three activities are implemented sequentially with little explicit consideration of fabrication or inspection during design. This approach allows designers to create features and designs which may be sub-optimal or even inconsistent with respect to available manufacturing and inspection capabilities, requiring multiple design-to-manufacturing iterations to converge on a successful part definition. These iterations can represent the bulk of the lead-time and development cost for a part. The philosophy of concurrent engineering attempts to solve this problem by explicitly considering post-design issues such as manufacturing and inspection during the design process. If a designer considers the manufacturability and inspectability of each feature, the resulting design will have a considerably higher probability of actually being made and verified.

The implementation of this philosophy has taken many different forms in both academic and industrial settings.

Industrial approaches

The most common approaches in industry for achieving concurrent engineering include team design, design reviews, rapid prototyping and product data management systems which allow manufacturing to view and critique design models. Although these practices are a large improvement over a non-concurrent environment, there are several key factors which prevent these methods from fully realizing the benefits of concurrent engineering:

1. *Manufacturing process knowledge is not represented in a formal or process-independent format.* Most industrial methods use human manufacturing experts to provide essentially ad-hoc manufacturability analysis. With this approach it is difficult to capture a record of either the reasoning technique or the applied manufacturing knowledge. The lack of explicit records makes it difficult to reapply experience gained from one product to future designs.

2. *An iterative "design a little then analyze" technique is the primary approach.* Little attempt is made to apply manufacturing process constraints before design begins. This can result in poor initial design choices — choices which will affect the remainder of the product development cycle. Furthermore, because manufacturability analysis requires human interaction, it is difficult and costly to shorten the cycle time between design critiques (the design cannot be reviewed after each feature is added).

3. *None of these methodologies are well suited for manufacturing processes which are new, unknown or geographically separated from the designer.* The need for direct human interaction and the reliance on implicit manufacturing knowledge make integration with remote and unknown manufacturing services difficult.

Academic approaches

Academic approaches to concurrent engineering have attempted to overcome the deficits of industrial systems through techniques such as formal process capability models and incremental manufacturability analysis. One technique is to represent manufacturability knowledge in an expert system which is applied at design time.[8,9] A more robust approach is to create prototypes of the unfinished design so that manufacturing and inspection costs and difficulties can be measured directly.[10] However, this approach, even with the advent of rapid prototyping processes, is comparatively expensive and time consuming. A third possibility is to conduct manufacturing simulations and process planning on the evolving design (the NextCut system[11] and other systems[12]). The simulators and planners use models of the manufacturing and inspection processes and a tradeoff consequently results among speed, accuracy and level of detail. Another approach is to design using machining features which are restricted by manufacturing process constraints.[11,13] Other concurrent engineering systems include those by Yannoulakis, *et. al.*,[14] and Lu[15] for evaluating the manufacturability of axi-symmetric parts.

An important distinction must be made between the rapid-prototyping approach and the other techniques discussed above. All of these research approaches, except for rapid-prototyping, require an abstract model of manufacturing process capability (rapid-prototyping physically simulates the process and therefore bypasses the need for a conceptual model). This paper is specifically concerned with those concurrent engineering applications which utilize formal process capability models to determine part manufacturability ("model-based" concurrent engineering).

Development of model-based concurrent engineering systems can be decomposed into three principle tasks:

1. Generation of process capability information.
2. Representation of process capability information.
3. Application of capability information during design to determine manufacturability.

Most concurrent engineering systems only focus on the tasks of representation and application of process capability models. Usually, the source for this capability information is a generic process model, and the problem of acquiring capability models for specific process instances is ignored. However, an accurate determination of design-for-manufacturability requires both generic process capability models and the dynamic capability information for a specific machine. Additionally, the lack of direct process information prevents these systems from evaluating new or unknown processes. Accordingly, there exists a need for a concurrent engineering architecture which provides an effective mechanism for obtaining formal capability models for actual processes.

2.3. Knowledge representation

An important element in any distributed engineering system is the mechanism used to represent domain information. Both the academic and industrial communities have recognized the need for a semantically unambiguous and machine-readable language for encoding relevant knowledge.[16-18] Successful integration of multiple engineers and their tools requires the entire enterprise to share a common model of relevant product and domain data. Inconsistencies or discrepancies between the different information models severely degrade overall system performance and hinder scalability. In the manufacturing domain, ambiguities in the product data exchanged between the designer and manufacturing will usually result in a non-functioning product and necessitate another development cycle.

The PDES/STEP standardization effort[1,19] represents a major international drive to develop a mechanism for the exchange of product information using a formal, machine-readable syntax (EXPRESS language)[20,21] with unambiguous semantic content (provided by integrated resources).[22-27] In a concurrent engineering system, information concerning part geometry, topology, features, material properties and tolerances (STEP Parts 41, 42, 43, 45, 47 & 48)[22-27] must be represented and exchanged. Several concurrent engineering systems have been developed which utilize portions of the STEP standard for information representation and exchange.[13,28-30]

2.4. Agent architecture and knowledge sharing

Realization of a concurrent design environment which uses machine-specific capability models requires the exchange of detailed process and product models between the designer and the manufacturing service. Because manufacturing services may be remotely located from the designer, the mechanism used to exchange process capability models must provide reliable, distributed exchange of machine-readable information. Previous collaborative engineering systems have satisfied this need with autonomous software agents which encapsulate the individual system components and communicate using a formal agent-communication language.[11,16-18,28] These systems used agent-based architectures to facilitate modularization and integration of distributed heterogeneous engineering systems. By using the Internet for reliable byte-level data connectivity, a truly world-wide distributed system is created.[16]

3. ON-MACHINE ACCEPTANCE

HTM model of the machine

The components in this work are stainless steel gas bottles turned on a slant-bed lathe. A schematic of the machine with a coordinate frame attached to both the workpiece and the tool is shown in Figure 1. The lathe consists of seven major independent elements. Using the method of homogeneous transformation matrices (HTMs), a model can be developed to describe the position of the cutting tool within the workpiece coordinate system as a function of the position of each machine element.[7]

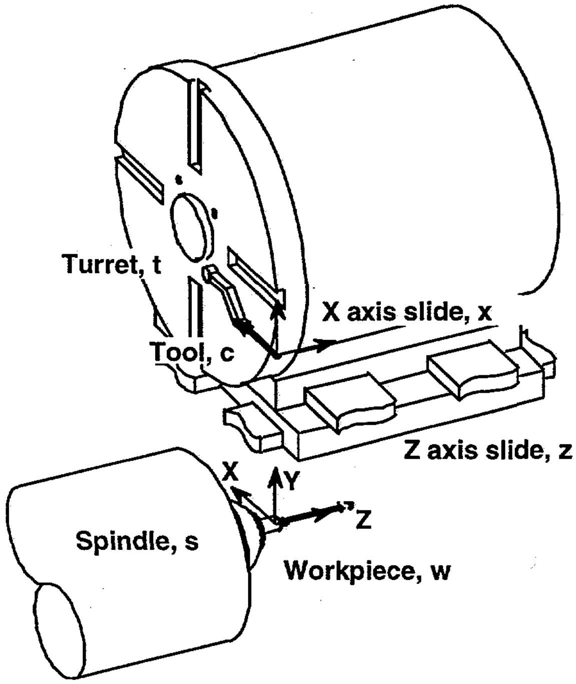

Figure 1. Schematic of the LeBlond lathe.

In addition to their intended movement, each element has six error motions, three translational and three rotational. These terms are also combined in the model to give a description of the ideal position of the machine as well as the error in that position. From the model, the error term in X is:

$$\delta x = \delta x(x) + \delta x(z) + \delta x(t) + \delta x(c) - \delta x(s) - \delta x(w) - (\varepsilon y(x) + \varepsilon y(t) + \varepsilon y(z) - \varepsilon y(s))Z_c - \left(\varepsilon y(s) - \alpha_{par}\right)z \quad (1)$$

Where $\delta a(b)$ represents a translational error in the a direction due to the b axis, and $\varepsilon c(d)$ represents a rotational error in the c direction due to the d axis. Z_c is a constant offset, α_{par} is a constant angular error, and z is the position of the z axis. A similar expression can be developed for the error in Z. These terms describe the error of the position of the tool with respect to the workpiece in the workpiece coordinate system. If the error terms are known as a function of the position of each machine element, this equation could be used to correct the position error of the machine tool. In this case, the errors are assumed to be known only as the maximum and minimum values in the work volume.

Error budget

The error equations can be used to develop an error budget for the inspection process that is based on the kinematics of the machine. The expected value of the error in X is merely (1) evaluated at the expected value of each term.[31] The variance of the error in X is:

$$Var(\delta x) = Var(\delta x(x)) + Var(\delta x(z)) + Var(\delta x(t)) + Var(\delta x(c)) + Var(\delta x(s)) + Var(\delta x(w)) + (Z_c)^2\left(Var(\varepsilon y(x)) + Var(\varepsilon y(t)) + Var(\varepsilon y(z))\right) + (Z_c - z)^2 Var(\varepsilon y(s)) + z^2 Var(\alpha_{par}) \quad (2)$$

Similar expressions can be developed for $E(\delta z)$ and $Var(\delta z)$.

Over the work volume of the machine, each error term can be considered to be uniformly distributed.[5] Machine data can be used to calculate the expected value and variance of each error term. The variance of the total error can be determined with the above equation. The resulting distribution is assumed normal by application of the Central Limit Theorem.

Machine tool metrology

Using standard techniques for machine tool metrology,[32] the geometry of the lathe can be measured to determine the values and distributions of the error terms. For a typical gas bottle, a working volume of approximately 7 inches in Z and 3 inches in diameter (X) is chosen. The difference between the maximum and minimum values in this volume of each error term is recorded (Figure 2). With the exception of the squareness and parallelism errors, each term is considered to be centered around zero. The uniform distribution parameters μ_1 and μ_0 are calculated as plus and minus half the total spread. Finally the expected value and the variance are determined from these parameters. It should be noted that the spread determined above is the same parameter that is measured when a machine is tested in accordance with the ANSI/ASME B5 standards[33].

Several of the parameters, including spindle error motions and thermal components of rigid body errors, are estimated based on data from other machines or information provided by manufacturers.

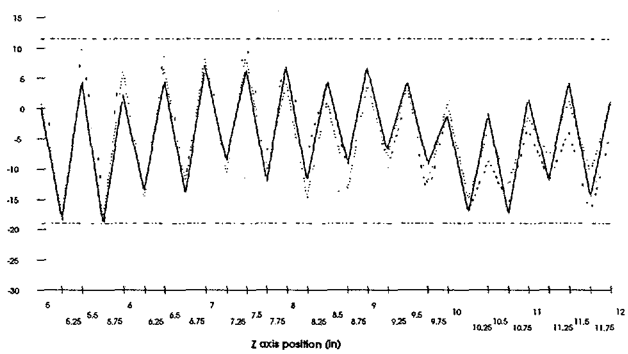

Figure 2. Z axis straightness with μ_1 and μ_0 shown.

Error budget calculation

Using (2), the kinematic error budget can be calculated based on the individual variances. Based on the metrology data of a LeBlond Baron 25 slant-bed lathe, $Var(\delta x)$ is estimated as 1.88×10^{-6} in.

Application to features

The above expressions describe the error and variance of single probed points. But features are measured using combinations of these probed points, thus alternate expressions must be developed. The mean diameter of a cylinder can be expressed as the average of several measurements in the X direction, or:

$$\varnothing = \frac{1}{N}\sum_{i=1}^{N} X_i \tag{3}$$

where N is the number of probed points and X_i is the x value of the probed point. The variance of the diameter measurement can be expressed as:

$$Var(\varnothing) = \frac{1}{N} Var(\delta x) \tag{4}$$

Cylindricity can be expressed as:

$$Cyl = \max(X_i) - \min(X_i) \tag{5}$$

A conservative estimate of the variance is:

$$Var(Cyl) = 2 \cdot Var(\delta x) \tag{6}$$

Using these expressions it is possible to express the uncertainty of measurements made with an on-machine inspection process. To do this the user first determines the level of confidence required of the measurement. For large N, a table of the standard normal distribution can then be used to translate the confidence level into a number, n_p, of standard deviations. Because the standard deviation is simply the square root of the variance, the uncertainty of the measurement at the given confidence level can be expressed as:

$$U_p(\varnothing) = \pm n_p \sqrt{Var(\varnothing)} \tag{7}$$

Using (4), (6), and (7), the uncertainty of diameter to the 99% confidence level is $\pm 772 \times 10^{-6}$ *in.* based on a 21 point measurement. The estimated uncertainty of cylindricity is 5.00×10^{-3} *in.* The on-machine inspection is considered acceptable if these values are smaller than the size and cylindricity tolerances of the cylinder. If this is not the case, the designer has a number of options. He can adjust the confidence level of the acceptance, change the required number of probe points, change the part tolerance, or request a more accurate calibration of the machine to reduce its position variance. The on-machine acceptance provides the designer with a mechanism to make these choices.

4. AGENT ARCHITECTURE AND INFORMATION INFRASTRUCTURE

Our concurrent engineering architecture provides a mechanism for utilizing the dynamic capability information provided by an on-machine inspection process to realize a concurrent design environment. An implementation of such a concurrent engineering environment must successfully address four primary challenges:

1. Process model generation.
2. Acquisition of the model by the designer.
3. Mapping the model into the design space.
4. Applying model information during design.

The goal of this architecture is not to develop "*the solution*" to concurrent engineering, but rather to implement a concurrent engineering system which will explore solutions to the challenges listed above, focusing specifically on the need for machine-specific capability information. Architectural functionality can be decomposed into process capability representation (challenge 1), agent-based knowledge exchange (challenge 2) and the concurrent design environment (challenges 3 & 4).

4.1. Representation of process capability models

The proposed architecture will use a formal object-oriented conceptual model (written in EXPRESS) to represent the capability information for an on-machine inspection process. This model will represent both the objects present in the development cycle and their relationship to one another during execution of the development cycle. In this model, knowledge relating to geometry, topology, materials, tolerances and features will be represented using STEP Parts 41,42, 43, 45, 47 and 48. This model will be hierarchical and will consist of the following primary objects (this represents a fairly standard model of the manufacturing process):

Manufacturing Service:

A manufacturing service comprises a set of machines, each of which implements one or more manufacturing processes. A manufacturing service also has a set of service level constraints, including available stock parts and those machines which are currently operational. The manufacturing service will be represented by a cell manager agent which will interact with an agent representing the designer.

Manufacturing Process:

A manufacturing process represents an activity which takes some input (subject to input constraints), performs some operation on the input (subject to operational constraints) and outputs the result. This class includes both fabrication and inspection processes.

Fabrication Process:

Here the input is a physical part, the operation is one of material removal or addition and the output is a new part. Fabrication operations are modeled as CSG (constructive solid geometry) operations (either union or difference) between the input part and a fabrication feature. Input constraints limit part materials and size; operational constraints restrict the geometry, and accuracy of applied fabrication features.

Material Removal Process:

The operations here are restricted to material removal. This class includes processes such as machining and grinding.

Machining Process:

Fabrication operations are restricted to chip removal.

Turning Process:

Fabrication features are limited to solids of revolution. The current architecture is limited to turning processes.

Inspection Process:

The input is a physical part, operations are dimensional measurements of inspection features and the output is the input part and the measured size of an inspection feature.

Turning Inspection Process:

Inspection process implemented on a lathe and used to inspect surfaces of revolution.

Machine:

Represents a physical mechanism which implements one or more manufacturing processes. In our architecture, a single lathe is used to implement both a turning process and an inspection process.

A machine has a set of general machine constraints which apply to all implemented processes. For each implemented process the machine possesses constraints on the geometry of the input part and a set of feasible fabrication features with constraints on geometry, topology and tolerance of the corresponding inspection features. For a given machine, tolerance constraints are provided by the error budget.

Machine w/ OMA:

Represents a machine which can fabricate and inspect a set of fabrication features. This is the type of machine which is considered in the present architecture. Constraints limit applied fabrication features to those features whose nominal geometry can be fabricated and whose specified tolerance can be verified by the given machine. Whether the specified tolerance can actually be fabricated is not considered.

Manufacturing Feature:

Represents a geometry object associated with an operation implemented by a manufacturing process.

Fabrication Feature:

The volume of material which is added or removed from the part is represented by a fabrication feature (similar to the concept of a machining form feature or material removal volume). Some subset of the surfaces on each fabrication feature result in inspection features. Each fabrication feature will have geometric constraints which limit the nominal size of the feature. Constraints on the tolerance achievable by the machine for a given feature will not be modeled. The present architecture is limited to fabrication features which can be described by hollow right circular cylinders.

Inspection Feature:

A surface on the input part which has specified dimensions and tolerances. Each inspection feature corresponds to a surface on an applied fabrication feature. Constraints associated with each inspection feature limit, for a given nominal size, the tolerances which can be accurately verified for that feature on a machine which implements an appropriate inspection process (i.e., a process capable of measuring the given feature type). The present architecture is limited to inspection features described by right circular cylindrical surfaces.

Constraint:

Constraints can be represented declaratively, as either qualitative rules (e.g., material must be one of either aluminum, brass or plastic, etc.) or algebraic constraint equations (e.g., ratio of length to diameter must be less than 3, etc.) or by functions which take relevant input parameters and procedurally determine constraint satisfaction. The proposed architecture will utilize both declarative and procedural constraint representation.

Process capability, as represented in the above model, is defined by constraints at four different levels: generic process class (turning), machine class (lathe), specific machine instance (actual lathe with OMA) and manufacturing service (IMTL). The constraints associated with process and machine classes are static and apply to all instances of those classes. The constraints associated with a specific machine and the service in which that machine is contained, however, are dynamic and must be continuously updated as the state of the machine and service change. The primary focus of the present architecture is the addition of the dynamic constraints for a machine instance to the generic process and machine class constraints. Specifically, the present architecture focuses on representing the constraints for a set of inspection features which can be measured by a specific on-machine acceptance process.

4.2. Agent architecture and knowledge exchange

In our agent architecture shown in Figure 3, knowledge and functionality are encapsulated inside agents. The primary benefit of an agent-based architecture is that it facilitates the modularization and integration of large systems comprised of distributed, heterogeneous components. The integration of multiple designers with multiple manufacturing services represents such a system. An agent information infrastructure facilitates the integration of heterogeneous software tools, databases, legacy software, CAD/CAM commercial packages, and newly developed software. Agents logically unify heterogeneous distributed information and knowledge. This particular architecture is geographically distributed, with the designer residing at Stanford University, and the manufacturing facility and on-machine acceptance capability residing at Sandia/California.

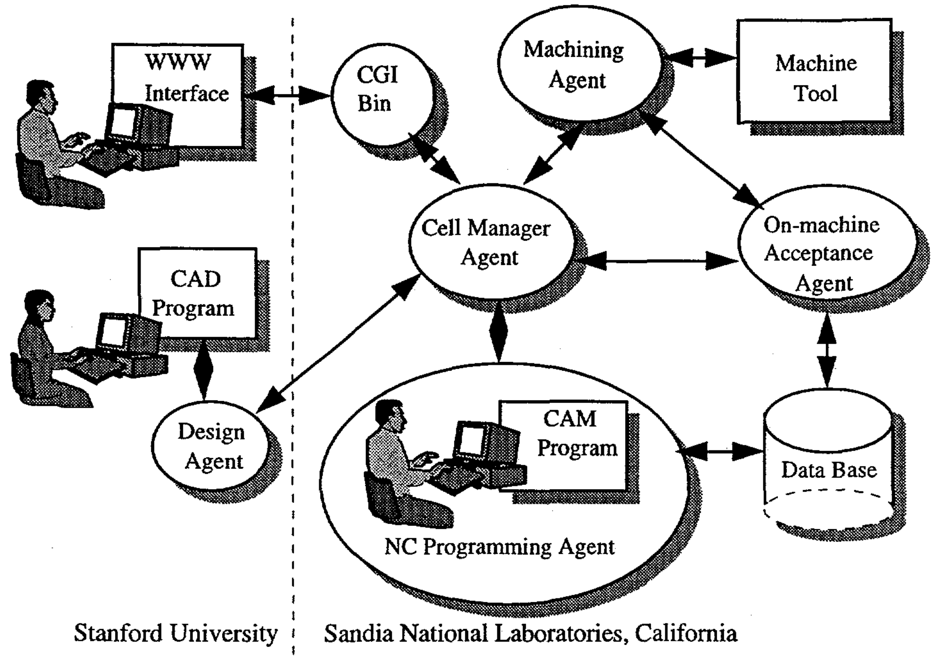

Figure 3. Agent Architecture.

In the context of this research, we define an agent as an autonomous, encapsulated software component that communicates with other agents using an agent communication language. We expand on each of these properties:

- *Autonomous*: Each agent operates independently and asynchronously and interacts with other agents on a peer-to-peer level, and not a strictly client-server communication structure.

- *Encapsulated*: Agents serve as containers for a collection of procedural and declarative knowledge representing some engineering functionality. This knowledge is only accessible via communication in the appropriate agent communication language.

- *Agent Communication Language*: An agent communication language possesses formally defined syntax and semantics and can be unambiguously represented in machine readable format. Examples of such agent communication languages are Knowledge Query and Manipulation Language (KQML)[34], Knowledge Interchange Format (KIF)[35] and STEP.

In our implementation, manufacturing process knowledge (see Section 4.1) will be represented using portions of the STEP standard and this knowledge will be exchanged as agent messages written in EXPRESS. KQML will be used for the outer structure of all agent messages. Agent messages will be sent over the Internet using the TCP/IP transport protocol.

Information, in the form of STEP schemas, that is exchanged from the inspection process to the design processes will be displayed graphically to designers as Java[36] applets. Java is a platform-independent, interpreted language which allows for the rapid prototyping of GUI's. The designer, therefore, does not need to learn or understand STEP and/or EXPRESS, and the design environment requires no additional software since several WWW browsers will support the transport and display of Java applets. Thus, the features and part information will be mapped to STEP as a method of knowledge exchange, and the designer will view this information in a format that is useful and readable to him/her.

When the designer requests information and process constraints from manufacturing services, the work is performed by the design agent, interfacing to the CAD system. The design agent serves two functions: first, as incoming agent messages are received, the contents are interpreted and based on the message contents, the CAD tool (and hence designer) is notified accordingly; and second, as design phase events occur, agent messages are constructed and sent to coordinating agents.

Similarly, the cell manager agent receives all requests from outside of the manufacturing cell, forwards requests to internal manufacturing agents as appropriate, combines any responses, and returns one or more messages to the sending agent.

4.3. Application of capability information during design

The concurrent engineering architecture will use a feature-based design system[10-12] which enables designers to evaluate the inspectability of each feature as it is added to a design. The system will consist of a CAD system (most likely PROEngineer by Parametric Technology Corporation), a design agent (see above description), and a constraint manager (see Figure 4). (Note: our architecture can support any commercial or publicly available CAD tool.) The design agent and constraint manager will communicate with the CAD system via the prescribed API. The appropriate process capability models and all relevant inspection constraints for the selected manufacturing service and machine will be acquired by the design agent (according to the above protocol) mapped into the local representation format (dependent on the CAD system and constraint manager) and loaded into the both the CAD system and constraint manager. The set of feasible fabrication features and stock parts will be loaded into the CAD system. The related process, machine and manufacturing service constraints on the fabrication features will be loaded into the constraint manager.

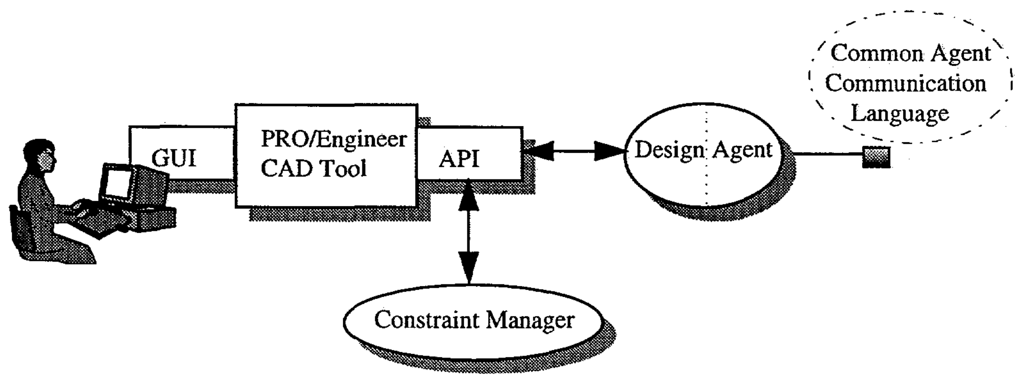

Figure 4. Design Agent: Internal Architecture.

Once the capability information has been acquired and loaded, the designer will use the CAD system to select a feasible stock and use CSG operations with feasible fabrication features to design a part. As each fabrication feature is applied to the part, the designer must identify, for the set of resulting inspection features, the desired tolerance. The specified tolerance, along with the nominal feature geometry and its position relative to the current part will be submitted to the constraint manager to determine constraint satisfaction. The constraint manager will apply all relevant declarative and procedural constraint information. If any constraint violations are found they will be reported to the designer, who may alter the applied feature or renegotiate the OMA constraints to satisfy the constraints or continue with the violating design.

4.4. An example scenario

In this section we provide a scenario to illustrate how information flows seamlessly from design to fabrication to inspection and back again, creating a continuous product development cycle. (Refer to Figure 3.) We assume that the designer (1) wants to design an axi-symmetric part which (2) will be turned on a lathe, (3) has selected the IMTL manufacturing service, and (4) knows the Internet address (URL) of this service. Accordingly, the only IMTL machine modeled will be a lathe which implements an on-machine acceptance process. In future work, we can relax these assumptions and allow design agents to negotiate[37] with multiple manufacturing services and to consider multiple manufacturing processes.

The first interaction between agents occurs when a design agent requests general process capability information from the cell manager agent for a specific process implemented on a specific machine within the service. The cell manager agent will send the following information to the design agent: service level constraints (including available raw stock), general process and machine constraints (independent from any specific features) and the set of form features which the machine can both fabricate and inspect. Much of this information will be displayed graphically to the designer, via solid models representing each fabrication feature.

If the manufacturing and inspection processes are consistent with a designer's intentions, the design agent requests on-machine inspection constraints for the subset of fabrication features that the designer intends to use during the design process. Currently, design features are limited to a simple set comprised of solids of revolution. The cell manager agent will return inspection constraints for that set of features. These constraints are bundled with the feature class to which they apply. The order of feature addition (essentially the process plan) and interactions between features will not be considered during constraint creation.

During the design process, each added feature is checked against the constraints to provide an estimation of inspectability using the constraint manager (constraint solver). A range of valid tolerances for each feature is generated. The constraint manager will record all design decisions and can backtrack within this design space. The addition of each feature will be analyzed as an isolated event. Thus, interaction between features (implicit features) will not be considered by the inspection constraints.

Once the design is completed with respect to all constraints, the designer uses an interface to the manufacturing cell on the World Wide Web (WWW) to submit part geometry, tolerances, and additional on-machine inspection requirements. As shown in Figure 3, CGI-bin scripts (Common Gateway Interface) provide an agent interface between a human on the WWW and the cell manager agent. The following on-machine inspection variables are available to the designer and can be submitted with the design: nominal geometry, level of certainty desired, number of probe points measured on machine, the date of the most recent calibration, and the tolerance, expressed in both cylindricity and size (diameter). If the part is deemed not inspectable by the agent, the designer receives a warning and these variables can be modified to allow inspection and acceptance.

The cell manager agent will return a cost quotation and equipment availability to the designer, either in an HTML document (on the WWW) or by electronic mail. The cell manager agent then coordinates the process planning, inspection planning, manufacturing, on-machine inspection and certification. The cell manager agent routes the part geometry information to an NC programming agent. At this time, the NC programming agent is a human and CAM software; however, in the future we intend to automate this process and wrap it as an agent. The human will generate the NC programs to machine the part (on the lathe) and to inspect the part, based on the on-machine inspection variables provided by the designer.

Once the NC program and on-machine inspection probe paths are produced, this information is sent to a machining agent. The machining agent submits programs to the NC machine tool (lathe) and monitors the manufacturing and on-machine inspection processes. The machine cuts the part and then performs the complete on-machine inspection. The inspection data is sent to an on-machine acceptance agent, which evaluates the data and generates statistically valid certification of reliability. The certification data is an error budget, computed as a function of tolerances. This certification and the accompanying data can be stored in a database to be used for assessing machine performance or documentation to troubleshoot failures in the field. In the future, this data can be used to generate dynamic inspection constraints, which will be sent to designers. The part and certification are returned to the requesting customer.

5. CONCLUSIONS AND FUTURE WORK

We have presented an agent-based architecture and related knowledge representation for using an on-machine inspection process in a concurrent engineering environment. This allows designers to design parts for manufacturability and inspectability, and hopefully reduces the computational cost of process planning. We are implementing this agent architecture as a distributed system where the design agent resides at Stanford University and the manufacturing agents reside at Sandia National Laboratories in Livermore, California. Agent messages are constructed in an unambiguous agent communication language, and they are sent between a designer and manufacturing services across the Internet. We believe that our concurrent engineering approach can be adapted easily to other design environments and manufacturing, and inspection processes.

Our future research with respect to the on-machine inspection process includes refining the uncertainty model to include thermal effects and relaxing the simplifying assumptions. The OMA model can be enhanced by the addition of historic on-machine acceptance data, and similarly, the OMA technology can be ported to other material removal processes (e.g., milling). With respect to constraints, we intend to support more complex geometries and features, costs associated with constraints, and fabrication constraints. Although fabrication feature constraints are not presently represented, because the same machine is used for both fabrication and inspection, the process capability model used to generate the inspection constraints should be largely reusable for creating fabrication constraints and thereby incorporating design-for-manufacturability. Finally, this concurrent design architecture will become more powerful as functionality is added to additional agents, and additional manufacturing processes (e.g., process planning and assembly) are encapsulated as agents.

6. ACKNOWLEDGMENTS

The authors would like to thank Professor Mark Cutkosky at the Center for Design Research, Stanford University for his valuable input.

7. REFERENCES

1. Mason, H. (ed.), *Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 1: Overview and Fundamental Principles*, Version 9, ISO TC184/SC4/WG PMAG Document N50, December 1991.

2. M. N. Sinha and W. O. Wilborn, *The Management of Quality Assurance*, John Wiley & Sons, Inc., 1985.

3. B. R. Taylor, "Dimensional control through the tolerancing of the solid model," *Proceedings of the 1993 International Forum on Dimensional Tolerancing and Metrology*, Dearborn, MI, 1993, 161-166.

4. J. B. Bryan, "The deterministic approach in metrology and manufacturing," *Proceedings of the 1993 International Forum on Dimensional Tolerancing and Metrology*, Dearborn, MI, 1993, 85-96.

5. R. R. Donaldson, "Error Budgets," *Technology of Machine Tools*, V.5, 1980, Sec. 9.14, 1-14.

6. D. C. Thompson, "The design of an ultra-precision CNC measuring machine," CIRP Annals, 1989.

7. A. J. Hazelton, "On-Machine Acceptance of Machined Components", *Proceedings of the 10th Annual Meeting of the American Society of Precision Engineering*, Austin, Texas, October 1995.

8. A. Lazaro, D. Engquist, and D. Edwards, "An Intelligent Design for Manufacturability System for Sheet-Metal Parts", *Concurrent Engineering: Research and Applications*, 1(2):117-123, 1993.

9. D. Rosen, J. Dixon, and C. Poli, "Features and Algorithms for Tooling Cost Evaluation in Injection Molding and Die Casting", *Computers in Engineering*, ASME, 1-8, 1992.

10. P. K. Wright, "Principles of Open-Architecture Manufacturing", Engineering Systems Research Center, ESRC 94-26, University of California at Berkeley, October 1994.

11. M. R. Cutkosky and J. M. Tenenbaum, "Toward a Framework for Concurrent Design", *International Journal of Systems Automation: Research and Applications*, 1(3):239-261, 1992.

12. S. K. Gupta and D. S. Nau, "A Systematic Approach for Analyzing the Manufacturability of Machined Parts, *Computer Aided Design*, 27(5):323-324, 1995.

13. C. Feng and A. Kusiak, "Constraint-based Design of Parts", *Computer-Aided Design*, 27(5):343-352, 1995.

14. N. Yannoulakis, S. Joshi, and R. Wysk, "A Manufacturability Evaluation and Improvement System", *Design Theory and Methodology*, ASME, 217-226, 1991.

15. S. Subramanyan and S. Lu, "The Impact of an AI-based Design Environment for Simultaneous Engineering on Process Planning", *International Journal of Computer Integrated Manufacturing*, 4(2):71-82, 1991.

16. M. Cutkosky, R. Engelmore, R. Fikes, T. Gruber, M. Genesereth, W. Mark, J. Tenenbaum, and J. Weber, "PACT: An experiment in integrating concurrent engineering systems", *IEEE Computer*, January 1993.

17. J. G. McGuire, D. R. Kuokka, J. C. Weber, J. M. Tenenbaum, T. R. Gruber, and G. R. Olsen, "SHADE: Technology for Knowledge-Based Collaborative Engineering", *Journal of Concurrent Engineering: Applications and Research (CERA)*, 1(2), September 1993.

18. G. R. Olsen, M. Cutkosky, J. M. Tenenbaum, and T. R. Gruber, "Collaborative Engineering based on Knowledge Sharing Agreements", *Proceedings of the 1994 ASME Database Symposium*, September 11-14, 1994, Minneapolis, MN.

19. J. Owen, *STEP: An Introduction*, Information Geometers Ltd, Winchester, UK, 1993.

20. P. Spiby (ed.), *ISO 10303 Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 11: Description Methods: The EXPRESS Language Reference Manual*, ISO DIS 10303-11:1992(E), July 1992.

21. Schenck and Wilson, "Information Modeling: The EXPRESS Way", Oxford University Press, 1994.

22. ISO 10303-41, *Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 41: Integrated Generic Resources: Fundamentals of Product Description and Support* ISO DIS 10303-41.

23. ISO 10303-42, *Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 42: Integrated Generic Resources: Geometric and Topological Representation*, ISO DIS 10303-42.

24. ISO 10303-43, *Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 43: Integrated Generic Resources: Representation Structure*, ISO DIS 10303-43.

25. ISO 10303-45, *Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 45: Integrated Generic Resources: Materials*, ISO DIS 10303-45.

26. ISO 10303-47, *Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 47: Integrated Generic Resources: Tolerances*, ISO DIS 10303-47.

27. ISO 10303-48, Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 48: Integrated Generic Resources: Form Features , ISO DIS 10303-48.

28. D. Goldstein, "An Agent-based Architecture for Concurrent Engineering", *Concurrent Engineering: Research and Applications*, 2: 117-123, 1994.

29. P. Gu and K. Chan, "Product Modelling using STEP", *Computer-Aided Design*, 27(3): 163-179, 1995.

30. T.-H. Liu and G. Fischer, "Developing Feature-based Manufacturing Applications Using PDES/STEP", Concurrent Engineering: Research and Applications, 1: 39-50, 1993.

31. G. J. Hahn and S. S. Shapiro, *Statistical Models in Engineering*, John Wiley & Sons, Inc., 1994.

32. A. H. Slocum, *Precision Machine Design*, Prentice-Hall, Inc., 1992.

33. *ANSI/ASME B5.54, Methods for Performance Evaluation of Computer Numerically Controlled Machining Centers*, American Society of Mechanical Engineers, 1993.

34. T. Finin, J. Weber, *et. al.*, "Specification of the KQML Agent-Communication Language", The DARPA Knowledge Sharing Initiative External Interfaces Working Group, February 9, 1994.

35. M. R. Genesereth, R. E. Fikes, *et. al.*, "Knowledge Interchange Format Version 3.0 Reference Manual", Computer Science Department, Stanford University, Stanford California, June 1992.

36. "The Java Language: A White Paper", Sun Microsystems, 1995.

37. M. R. Genesereth and S. Ketchpel, "Software Agents", *Communications of the ACM*, Vol. 37, No. 7, July 1994, 48-53.