Manufacturing knowledge verification in design support systems

This item was submitted to Loughborough University’s Institutional Repository by the/an author.

Citation: COCHRANE, S. ... et al., 2009. Manufacturing knowledge verification in design support systems. International Journal of Production Research, 47 (12), pp.3179-3204.

Additional Information:

- This article was published in the journal, International Journal of Production Research [© Taylor & Francis] and is available at: http://dx.doi.org/10.1080/00207540701802452

Metadata Record: https://dspace.lboro.ac.uk/2134/4840

Version: Accepted for publication

Publisher: © Taylor & Francis

Please cite the published version.
This item was submitted to Loughborough’s Institutional Repository (https://dspace.lboro.ac.uk/) by the author and is made available under the following Creative Commons Licence conditions.

For the full text of this licence, please go to: http://creativecommons.org/licenses/by-nc-nd/2.5/
Manufacturing Knowledge Verification in Design Support Systems

Sean Cochrane\textsuperscript{a}, Robert Young\textsuperscript{a}, Keith Case\textsuperscript{a}, Jennifer Harding\textsuperscript{a}, James Gao\textsuperscript{b}, Shilpa Dani\textsuperscript{a}, David Baxter\textsuperscript{b}

\textsuperscript{a} Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University, Leicestershire, LE11 3TU, UK
\textsuperscript{b} Dept. of Enterprise Integration, School of Industrial and Manufacturing Science, Cranfield University, Bedford, MK43 0AL, UK

Abstract

This work identifies the need for a knowledge verification methodology for manufacturing facility representations in enterprise models. A methodology based on the concept of ontological commitment is then proposed. This uses a shared foundation-ontology to define a series of verification procedures. The PSL ontology is identified as being suitable for this purpose, and an evaluation of how various categories of manufacturing knowledge (typical to design support systems) map onto the PSL ontology is undertaken. Whilst the PSL ontology was found to support the verification of most categories of manufacturing knowledge, additional procedures were found to be required. These were primarily needed to verify relationships between products and manufacturing processes.

Key Words: Knowledge Representation, Validation and Verification, Enterprise Modelling, Ontology, Design for Manufacture.

1. Introduction

Making the right decision is aided by IT systems that make the right information available at the right time, in the right place, and in the right format (Bullinger et. al. 2000); and these systems often use an enterprise models to store and retrieve information. Enterprise modelling is however a complex task, and several methodologies and reference models were developed in the late 1990’s to assist this process. These included: CommonKADS (Schreiber et. al. 1999), CIMOSA (Kosanke et. al. 1999), and the Reference Model for Open Distributed Processing (ISO/IEC 10746-1). These invariably applied some form of information view, where knowledge was classified according to a suitable hierarchy. Hierarchies for manufacturing enterprises were also developed at this time (e.g. Oldham et. al. 1998, and Molina and Bell 1999).
More recent work into the modelling of manufacturing enterprises has focused on issues such as: knowledge maintenance (Guerra 2004), the representation of globally distributed supply chains (Liu and Young, 2004), and knowledge sharing between design teams using ontologies (Lin et. al. 2004). These research efforts do not however tackle knowledge verification.

The term verification is sometimes used interchangeably with validation, but refers to a subtly different task. Validation involves making the right system, whilst verification involves making the system right (O’Keefe and O’Leary, 1993). Validation must therefore ensure that a system meets the requirements of end-users, whilst verification ensures that a system meets its specified requirements (Preece, 2001). Within a design support system, verification must therefore ensure that any rules and constraints used to support engineering decisions are at least consistent. Indeed, from any contradictory knowledge, an agent would be able to deduce any conclusion, and it’s contrary (Gregoire and Mazure 2002).

Several verification techniques are reported in the literature. These identify inconsistencies in formal knowledge representations, e.g. contradictory rules, and unreachable conditions (Preece et. al. 1992, Wu and Lee 2002, and Gregoire and Mazure 2002). An approach described as ontological commitment (Waterson and Preece, 1999) is also described. Ontologies provide a set of rules and constraints associated with a class schema for describing an environment (Smith et. al. 2003). Ontological commitment means that a knowledge base follows the rules and constraints of a supporting ontology. By committing to a shared ontology interactions between several knowledge bases are checked against a common set of rules and constraints. Whilst these verification techniques are well established, they tackle the subject from a general computer science perspective, and provide no specific guidance on the verification of manufacturing enterprise models. This research therefore aims to establish a methodology, specifically for the verification of manufacturing knowledge in design support systems.

2. Verification Methodology

The methodology developed by this research applies the concept of ontological commitment to manufacturing enterprise representations. A common set of rules and constraints (defined by a shared ontology) is enforced across multiple models of manufacturing facilities. The shared ontology provides a meta-level description of all facility models. The question arises as to which ontology should be used for this purpose?

Many forms of ontology are encountered in knowledge based systems, ranging from glossaries and taxonomies, to generic logical constraints (McGuiness, 2002). Ontological commitment however, requires a logically consistent set of constraints. This work therefore focuses on rigorous-formal ontologies (Uschold and Gruninger, 1996), which define detailed sets of rules and constraints using statements of first order logic (i.e. axioms). The Process Specification Language (PSL) (ISO/CD18629) falls into this category, and is especially relevant to representations of manufacturing enterprises.
PSL formally defines terms for describing rules and constraints relevant to processes. The ontology has been applied to the exchange of project planning information (Gruninger et al. 2003), describing process inputs and outputs (Bock and Gruninger, 2004a), and the modelling of process flows (Bock and Gruninger, 2004b). Other experimental work includes a shared ontology library (Chira et al. 2004) which plans to use PSL to formalise the representation of design processes. PSL should therefore provide most of the concepts and supporting definitions needed to verify manufacturing knowledge in design support systems.

Figure 1 describes how multiple knowledge bases can be checked against the axioms of the PSL ontology. These verification procedures will for example identify any conflicting statements regarding the sequencing of activity occurrences. A knowledge verification methodology (KVM) based on this approach is depicted in figure 2. This shows how an executable representation of the PSL-ontology supports the coding and verification of computational views of manufacturing facilities (within the RM-ODP). The KVM focuses on the computational view due to the formal nature of the PSL ontology. The enterprise and information views are essentially informal, and can not therefore be verified using the process of ontological commitment.

Figure 3 shows how the KVM can be applied to the manufacture of a new product (defined by a requirement specification). The PSL ontology is used to describe manufacturing facilities (e.g. foundries and machine tools). Facility representations can then be verified by as series of executable procedures based on the axioms of the PSL ontology. Any violations are logged by an “axiom violations report”, which highlights inconsistencies between the facility representation and the PSL ontology. Initially, verification is limited to procedures based on the PSL core and outer core. Further procedures based on the PSL extensions (and other ontologies) can be added, as required.
The use of PSL in design support systems is however relatively immature, and the extent to which the ontology supports a workable set of verification procedures needs to be established. The following sections therefore describe the categories of knowledge typical to manufacturing facility representations, and the mapping between these categories of knowledge and the PSL ontology. This established the extent to which the PSL supports the verification methodology described above.

3. Facility Representations

The Manufacturing Capability Model (Molina and Bell, 1999) was previously described as a suitable information hierarchy for the representation of manufacturing facilities. This defined a class hierarchy that included resources, processes and strategies (see figure 4). The resource class stores attributes related to a facilities use of resources. A machine shop will for example use machine tools, drill-bits, lubricating oils, and work pieces. Attributes may therefore include the minimum and maximum sizes of work-pieces; the availability of tools, bits and lubricating oils; the number of machine tools and drill-bits used for individual drilling processes; and the rate at which lubricating oils are consumed.
The process class stores attributes related to processes, e.g. milling rates, and achieved tolerances. Process constraints may also be described, e.g. the maximum input surface tolerance for a finish milling process.

Figure 4: Design and Manufacturing Views (Zhao et. al. 2000)

Both the resource and process classes are considered as information classes rather than knowledge representations. This is because they provide no information relating to how the information stored by the class is used. Knowledge of how to interpret resource and process settings is however stored in the strategy class. Indeed, the separation of strategies from resource and process information is one of the significant contributions of the Manufacturing Capability Model. Strategies describe how a facility applies resources and processes to make products, and can therefore be considered as a knowledge representation.

The role of the strategy class is demonstrated by the following example. A constraint stating that a “grinding process can only be applied to a surface if its tolerance is already less than 500μm” can not be readily applied to different machine tools with different processing capabilities. This is because the “500μm” attribute is directly referenced by the constraint. Making indirect reference to the 500μm value however, allows the rule to be more generally applied: e.g. the “grinding process can only be applied when the surface tolerance is within the machine tool’s grinding-capability”. If the grinding capability is stored separately from the strategy (i.e. in a separate process class) then the same strategy can be reused for multiple machine tools. This principle has been described within agent systems, as the separation of declarative data from procedural knowledge (Wray et. al. 2004).

The strategy class also includes four categories of knowledge, i.e. planning, capacity, technology, and facility. A description of these categories is provided below (based on the work of Molina and Bell, 1999).
Planning Strategies

Planning strategies describe the creation and manipulation of process plans for the manufacture of a product to a given specification, and enterprise/factory configuration. This knowledge can be used to estimate how long it takes to manufacture a product, and will describe:

1. Hierarchies of processes and sub-processes, e.g. drilling and machining are all sub-processes of machining.
2. How processes should be sequenced, e.g. casting precedes machining, and setting must occur before a work piece can be milled.
3. How to calculate the duration of a process. This is often a function of a processing rate and a geometric feature of a product.

Certain levels of planning knowledge will also be relevant to different levels of facility representation. For example, a model of an individual machine tool can describe constraints on the processes under its control (e.g. setting is required before milling), but can not assume knowledge of other facilities. A constraint on “casting preceding machining” must for example be described by a factory or enterprise level model, which makes assumptions about the availability of foundries and machine tools. This allows the machine tool model (on its own) to be reused in environments using forges and other fabrication technologies.

Capacity Strategies

Capacity knowledge describes how many units can be produced by a facility given the availability of resources. This requires an understanding of how long each process demands certain resources, e.g. milling requires the use of a machine tool for the duration of an occurrence of milling; and details of how many units are produced by a process. A casting process will for example produce batches of casts, whilst a machine tool can only work on individual pieces. Capacity knowledge will also describe which processes are best for certain volumes of production.

Technology Strategies

Technology knowledge interprets the information associated with the resource and process classes described above. This may include:

1. Rules for maximum and minimum part dimensions, e.g. the foundry can not cast a part larger than 2m x 2m.
2. Relationships between facility capabilities and achieved tolerances (e.g. the surface tolerance of an object equals 0.05mm after grinding).
3. Limits on which processes can be performed on different materials, and different tolerance capabilities for different materials.

Technology strategies may also overlap with planning strategies, in the interpretation of facility attributes such as speeds and feeds. The time it takes to mill a surface will for example be a function of the area being milled (as defined in the product requirement), and the capability of the machine tool being used for the selected conditions and material.
**Facility Strategies**

Facility knowledge describes how and when facilities should be used to achieve manufacturing objectives. This may include the selection of processes, based on required tolerances, e.g. “grind a surface if the required tolerance can not be met by the milling process”.

As with the planning constraints described above, facility knowledge needs to be appropriate to the level of facility being described. The selection of different foundries and machine tools can for example only be considered at the factory/enterprise level, where the capabilities of each facility can be considered. For example, a multi purpose machine tool may not be the best choice of facility if a product only requires a simple milling operation. The choice of whether to use such a facility can however, only be made at a level that understands what machine tools are available.

The following sections examine how the PSL ontology relates to the planning, capacity, technology and facility strategies.

**4. Planning Strategies and Process Flows**

*Representing a Process*

Planning strategies describe the creation and manipulation of process plans (and how they change with different product requirements and facility attributes). The PSL concept of an *activity_occurrence* can be used to represent a process within a process plan. *Activity_occurrences* are instances of *activities* bound by two timepoints, i.e. the beginning and end of the occurrence (see core axiom 14). Planning strategies however, also need to be checked for inconsistencies. The following paragraphs describe how the axioms of the PSL ontology support these verification procedures.

At a basic level, verification must ensure that no process ends before it begins. This constraint is not defined as part of the manufacturing capability model, but is a reasonable statement regarding all planning strategies. The PSL core defines a series of constraints on activity occurrences that identify such conditions, i.e. Core Axiom 15: the *begin point of every activity occurrence or object is before or equal to its end point.*

\[
(\forall \ a \ x \ (\forall \ a \ x) \\
(\implies \\
(\text{or} (\text{activity_occurrence} \ x) \\
(\text{object} \ x)) \\
(\text{beforeEq} (\text{beginof} \ x) \\
(\text{endof} \ x)))))
\]

The *before* relationship describes a *total ordering between timepoints* (i.e. core axiom 2). Any process (represented as activity occurrence) ending before it begins should therefore be highlighted as a violation of core axioms 2 and 15.

Further PSL core axioms provide additional rigour to the checks performed on planning
strategies. For example, PSL core axiom 4 states that the *before relation is transitive, i.e.*

\[
\text{forall } (?t0 \ ?t1 \ ?t2) \\
\text{implies} \\
\text{ (and (before } ?t0 \ ?t1) \\
\text{ (before } ?t1 \ ?t2)) \\
\text{ (before } ?t0 \ ?t2))
\]

This means that the total ordering of timepoints (in a process plan) must be checked for transitive before relationships, as well as those directly stated by facility representations. This characteristic is central to verifying plans generated by multiple facility models, as many inconsistent-timings are likely to be inferred (consequences of other relationships), rather than directly stated.

*Process Hierarchies*

Planning strategies typically involve hierarchies of processes. A machining process may for example involve several occurrences of milling and drilling. The PSL outer core provides concepts for describing such hierarchies (i.e. the theory of activity occurrences). Bock and Gruninger (2004b) have demonstrated the application of these concepts to describing process flows, noting that *other languages do not provide a mapping to mathematical structures to verify their concepts.*

The example used by Bock and Gruninger (see below) shows a process hierarchy for an occurrence of the *ChangeColor*, using the Knowledge Interchange Format (KIF), ISO/JTC1/SC32/WG2). This consists of two sub-activity occurrences, i.e. painting and drying. The terms *root* and *leaf* are also defined by the PSL outer core, and refer to the first and final sub-occurrences in an occurrence tree.

\[
\text{(forall } (?occChangeColor) \\
\text{implies} \\
\text{ (occurrence_of } ?occChangeColor \ ChangeColor) \\
\text{ (exists } (?occPaint \ ?occDry) \\
\text{ (and (occurrence_of } ?occPaint \ Paint) \\
\text{ (occurrence_of } ?occDry \ Dry) \\
\text{ (subactivity_occurrence } ?occPaint \ ?occChangeColor) \\
\text{ (subactivity_occurrence } ?occDry \ ?occChangeColor) \\
\text{ (min_precedes } ?occPaint \ ?occDry \ ChangeColor) \\
\text{ (root_occ } ?occPaint \ ?occChangeColor) \\
\text{ (leaf_occ } ?occDry \ ?occChangeColor)))}
\]

The *min_precedes* relationship is defined by the theory of activity occurrences: Axiom 1. This indicates that *occPaint and *occDry are sub-activity occurrences of an occurrence of ChangeColor. An attempt to associate these occurrences with an activity other than ChangeColor would be therefore highlighted as an axiom violation.
Constraining Processes

Planning strategies also need to place constraints on the sequencing of processes (e.g. “painting must precede drying”). Further concepts (described in the PSL occurrence tree definitions) can be used to represent such constraints. These include the term *poss*.

The “poss ?a, ?occ” relationship constrains activity ?a to being possible only after the occurrence ?occ, i.e. the activity must be a legal successor of the occurrence. Legal sequences of processes can therefore be specified and illegal sequences highlighted by appropriate verification procedures. Examples of how this might be used are discussed below.

Firstly, the drying example described above (Bock and Gruninger, 2004b) could constrain the drying activity to being possible only after an occurrence of painting, i.e. “poss ?drying ?occPaint”. This would prevent process plans from attempting to dry unpainted objects. Secondly, milling (as part of a machining process) may only be possible after an occurrence of machine setting. Any attempt to mill an object that has not been set, would therefore be identified as an illegal successor of any previous occurrences.

Processes Durations

An important part of any planning strategy is its understanding of how long it takes to perform different processes under different conditions and for different product requirements. The PSL “*durations*” extension is of particular use in representing these timing aspects of a planning strategy. The duration of an activity-occurrence is the difference between its end and beginning timepoints. In a computational implementation a “duration” can be represented by a long integer denoting the elapsed nano-seconds between two timepoints. The time it takes to perform a process can therefore be represented by the PSL concept of durations.

One issue in using durations however (as defined by the PSL ontology) is the relationship between planning and technology strategies. In many cases, durations are determined by a combination of conditions relating to the object being manufactured, and the resource/process attributes of a facility. The duration of a milling occurrence is for example, determined by the milling rate of a machine tool for a specific material, and the surface area of the object being milled. Interpreting these aspects of a technology/planning strategy requires some way of relating durations to the geometric attributes of products and facilities (e.g. speeds and feeds).

The PSL ontology has no concepts for describing and relating geometric attributes to durations. Additional concepts are therefore needed to interpret real numbers with units of: meters, meters-squared, cubic-meters, grams, Degrees-Celsius, radians, and seconds. Relationships between nano-, milli-, centi-, and kilo- measurements also need to be considered; and constraints on the manipulation of units should be enforced (e.g. meters can not be added to grams, and m3 can not be added to m2).
5. Technology Strategies and States

The influence of technology strategies on process timings were discussed above. Technology strategies however, also describe how processes are constrained by their environment, and how processes change their environment. This includes an understanding of process inputs, i.e. conditions that must be achieved before a process starts; and process outputs, i.e. any changes to the attributes of manufactured products and/or facilities that occur as a result of a process. This section examines how the PSL ontology can be used to describe these aspects of a technology strategy.

In addition to occurrence trees, PSL supports the constraining of occurrences by states. These are based on a technique referred to as situation calculus, where formal knowledge representations are derived from informal narratives. An example of how this is achieved is provided by McCarthy and Costello (1998).

Process Inputs and Outputs

Recent work has gone into modelling process inputs and outputs, using the PSL concept of states (Bock and Gruninger, 2004a). This allows situations to be described using statements such as holds and priors (similar to the example shown above). These describe states that are held following the occurrence of an activity, or set as a prior condition for an occurrence. The technique applied by Bock and Gruninger (2004a) is shown below. These describe the concepts of occurrence-inputs and occurrence-outputs, which are refinements of the PSL definition of states (again using KIF syntax).

Expressions 2, 3 & 4: Object participation

(forall (?x ?s)
 (implies
  (or (occ-input ?x ?s)
      (occ-output ?x ?s))
  (and (object ?x)
       (not (state ?x))
       (activity_occ ?s)
       (exists (?t)
        participates_in(?x,?s,?t)))))

The terms occurrence-input and occurrence-output constrain objects to being participants in an occurrence at a given timepoint t (the participates_in relationship is defined in the PSL core). This does not however say anything about how the objects participate. Expressions 7, 8 and 9 show how this can be achieved.

Expression 7: Types for Input and Output States

(forall (?x ?s ?f)
 (implies
  (or (input-state ?x ?s ?f)
      (output-state ?x ?s ?f))
Input and output states are defined by their relationship to an object and an activity occurrence (i.e. they reference the state of an object that participates in an occurrence). Input states define a state that must exist prior to an occurrence, and output states indicate how the state of an object is changed by an occurrence.

**Expression 8: Preconditions for Occurrence Inputs**

\[
\text{forall } \langle ?x \ ?s \ ?f \rangle \\
\text{(implies)} \\
\text{(input-state } ?x \ ?s \ ?f) \\
\text{(and (occurrence-input } ?x \ ?s) \\
\text{(prior } ?f \ ?s) \\
\text{(exists_at } ?x \ \text{(begin_of } ?s)))
\]

**Expression 9: Postconditions for Occurrence Outputs**

\[
\text{forall } \langle ?x \ ?s \ ?f \rangle \\
\text{(implies)} \\
\text{(output-state } ?x \ ?s \ ?f) \\
\text{(and (occ-output } ?x \ ?s) \\
\text{(holds } ?f \ ?occ) \\
\text{(not (prior } ?f \ ?occ)) \\
\text{(exists_at } ?x \ \text{(end_of } ?s)))
\]

A technology strategy describes how the state of product and/or facility is changed by a process. Any states required before a process can occur, should also be specified. The state “wet” could for example be used to represent an output of the painting process described above. This would be held true following an occurrence of painting, and set as a prior condition of the drying process. This prior condition would highlight any attempts to dry an unpainted surface.

States can provide a more reusable set of constraints on process models, than occurrence trees. This is particularly true when the constraints are themselves based on the condition (or state) of the object being manufactured. For example, a drying process model may be reused in a different environment as part of a cleaning process. Drying could therefore follow an occurrence of washing (to remove grease from a surface), providing the washing occurrence sets the “wet” state to true. A poss relationship with painting (based on an occurrence tree) would however prevent the model being reused in this fashion, as it makes assumptions about previous processes rather than examining the actual state of
objects. By setting the state “wet” as a prior condition of drying, the process model can be used to describe many different scenarios, including cleaning and changing colour.

Relationship of States to Technology Strategies

The PSL-concept of states is most effective when facility and product descriptions are constant (as in process planning applications). Here, it is enough to describe an objects state using binary attributes (e.g. the component is either wet or not-wet). Technology strategies however, tend to describe the inputs and outputs of processes in terms of geometries. For example, the tolerance held following a milling process is usually expressed in millimetres. Maximum and minimum constraints on a milling machine may also be expressed in metres, or even Kilograms.

It is therefore useful to describe “geometric-states”, that are held/achieved by activity occurrences, rather than limiting representations to the existing PSL concept of binary states. A surface tolerance may therefore be held to 3mm after a casting occurrence, and 0.1mm following a milling occurrence. This allows process models to describe the behaviour of processes more precisely. Chapter 5 shows how the concept of geometric states has been adapted from the PSL ontology. This allows geometric attributes to hold specific values following an occurrence, and for prior conditions on geometric attributes to set for activity occurrences.

The benefits of using geometric states become apparent when prior constraints on processes need to be represented. A finish milling process can now for example set a maximum prior tolerance (e.g. the surface has to be within 0.5mm before finish milling can occur). Verification procedures are now capable of detecting any occurrence of finish milling that attempts to mill a surface outside the required range. Prior states also have the reuse benefit of placing no restriction on how the required prior state is achieved.

6. Facility Strategies and States

Facility knowledge describes how to apply resources and processes. Many of the rules associated with its representation require an understanding of the current state of manufacturing facilities and products, and the required state of the final product.

For example, a milling process may be applied if the surface tolerance of the current manufactured part is outside the required tolerance of the final product, and within the operating range of a machine tool. The operating range may be specified in terms of an input and output tolerance. This ensures that a minimum tolerance has already been achieved prior to a milling occurrence (i.e. the input tolerance), and that milling actually improves the surface tolerance (i.e. the surface tolerance is not already less than the output tolerance of the milling process). Other criteria may also describe how to select the optimum sequence of processes, based on the end requirement, and the capabilities of the available facilities. This could for example, identify that a product only requires milling, and will therefore choose to use a simple milling machine, rather than an all-purpose facility (that can also perform drilling and turning).
The representation of facility knowledge requires a similar understanding of the manufacturing environment, to the technology constraints described above. Unlike technology representations however, they describe the way in which processes are sequenced to achieve manufacturing objectives (e.g. make a product). The technology constraints (described using hold and prior statements) ensure that facility strategies stay within the capabilities of each facility.

The boolean examination of geometric states plays an important role in describing the rules associated with facility strategies, e.g. perform process A if attribute-x is less than attribute-y at timepoint t. The state of attributes x and y at timepoint t can be determined by examining the geometric states held as a result of earlier occurrences.

7. Capacity Strategies and Resources

Capacity strategies describe the volumetric variations in manufacturing processes (e.g. what if we make 1000 units instead of 100?). Validating the representation of these effects requires some notion of resource, and how quantities of resource are demanded and changed by processes. The Process Specification Language contains a concept of resource that can be shared (and potentially committed to) by multiple facility representations.

A resource is defined as: a resource is any object that is required by some activity – where “activity” and “requires” are defined elsewhere in PSL. This definition is expressed with KIF syntax (Cutting Decelle et. al. 2003):

\[
(defrelation \text{resource} \ (?r) := (\text{object} ?r) \\
(exists \ (?a) \\
(requires \ ?a \ ?r)))
\]

The Theory of Resources (extension to the PSL ontology) also defines the concepts of available-quantity and aggregate-demand. Broadly speaking activities can demand a quantity of resource, and the aggregate-demand is the sum of all demands running concurrently with an occurrence. The available quantity of resource can also be held to a given value by an activity occurrence. For all occurrences the aggregate demand for a resource must not exceed the available quantity.

Process capabilities often vary according to production volumes. Throughput calculations must therefore take account of volumetric effects; and monitor the usage of: foundries, machine tools and work pieces. Any overloaded resources should be highlighted, and a distinction needs to be made between resources that are consumed (e.g. melt, casts, and work pieces), and resources that are not consumed (e.g. foundries and machine tools).

PSL provides a series of concepts to describe and constrain the use of resources. These include the concept of a resource with an “available quantity” and an “aggregate demand” described in the resources extension.
8. Experimental Environment and Results

The Manufacturability Analysis Platform (see figure 5) has been used to evaluate the Knowledge Verification Methodology developed by this research. Manufacturing strategies (describing how products are manufactured) are expressed using a series of shared terms (based on the PSL ontology). These terms are implemented as Java methods, which also perform the verification procedures described above (based on the axioms of the PSL ontology). These methods support the description of product attributes, processes, and the relationships between feature requirements and processes. Models of standard features (e.g. cylinders and holes) can also be referenced (in the feature library), along with customised models of bespoke features.

The MAP uses a tiered architecture to separate information describing processes and resources (stored in relational databases and/or spreadsheet tables) from rules and constraints (coded in Java and expressed using the shared terms/methods). The platform also implements required and manufactured product models; and separates low level feature strategies (e.g. drilling individual holes and machining surfaces) from high level strategies for manufacturing whole components.

Figure 6 (overleaf) shows a screen shot of the MAP user interface, and the results of a simulated component level strategy. The top left box on the screen allows users to configure both product requirements and the manufacturing enterprise. The top right box shows a progress log of the analysis (as the effects of the strategy are simulated by the tool). The bottom box shows details of any errors generated by the simulation, the
timings calculated for each process occurrence, and the resources demanded by the strategy. The errors report is central to the verification methodology as each error highlights discrepancies between the manufacturing strategy and the axioms of the underlying ontology.

The experimental results shown in figure 6 indicated that the selected strategy and enterprise configuration can not manufacture the component to its required tolerances (as described in the required component model). These tolerances are represented by geometric state conditions (see section 6) that must be achieved prior to an inspection processes. This checks whether all design requirements have been met by the selected component strategy. The tolerance errors generated by this experiment can be cleared by either relaxing the required tolerances, or by selecting a more accurate machine tool. A designer would then need to decide whether the relaxed tolerances and/or additional costs associated with the machine tool were acceptable.

The other errors in figure 6 show violations of timing constraints (i.e. the strategy can not make the required component within the required time). These timing constraints are represented within the strategy models by the beforeEq relationship (as defined by the PSL ontology). These errors indicate that the delivery process begins before the inspection process ends. Further simulations have shown that these can be eliminated by either relaxing the build time specification, relaxing certain tolerances (so that less time is spent machining the component), and/or using a faster machine tool. Again designers must decide whether these compromises are acceptable.

Other error conditions (not shown in figure 6) that have been simulated using the MAP...
include resource errors, and process sequencing errors. Resource errors are created when resources (e.g. machine tools, foundries and welding stations) are over utilised by a manufacturing strategy. These are set using the resource demands and resource holds constraints (as defined by the PSL ontology). The purpose here is to analyse a process plan generated by a strategy during the design process (this is different to generating an actual production schedule). Again these errors can be cleared by allocating more resource and/or lengthening the time available (to avoid concurrent resource demands). Sequencing errors occur when activity trees have been defined which constrain the order of events. These have been simulated using the MAP for machine setting sequences, and have been used to ensure that no machining processes occur without an earlier occurrence of machine setting.

9. Conclusions and Further Work

The aim of this research was to establish a methodology for the verification of manufacturing knowledge in design support systems. The information hierarchies proposed by Molina and Bell (1999) support the classification of manufacturing knowledge. It was found however that recent work on knowledge reuse, maintenance and sharing had applied and improved these hierarchies, but had not addressed verification. An examination of verification research identified the technique of ontological commitment as a potential foundation for a verification methodology, but also identified the need to relate this technique to manufacturing knowledge in design support systems. A further review of ontologies identified PSL as a suitable ontology. A Knowledge Verification Methodology (KVM) based on ontological commitment was therefore developed into a set of verification procedures derived from PSL axioms. The need for additional (non PSL based procedures) was also identified. These are needed to represent and verify relationships between product descriptions and manufacturing processes.

Knowledge verification is however only one stage in the overall process of knowledge validation. Verification ensures that knowledge bases are consistent, but this does not mean that representations are accurate, or in line with end-user requirements. Validation stages are therefore required. Giarratano and Riley (1998) describe the validation of general forms of knowledge. Their methodology included the comparison of system performance with derived solutions from domain experts. It would therefore be useful to develop guidelines on what to include in a set of “derived solutions” suitable for manufacturing and design environments. It would also be useful to examine some of the existing (general) knowledge based systems development methodologies, and their approach to validation. Figure 7 shows how the KVM could be integrated with the Meta-Knowledge approach. This is based on the Meta-Knowledge Model (Plant and Gamble, 2003) which provide guidelines on how to specify and validate each level of representation. This may include recommended information structures, and a formal foundation-ontology.

The PSL ontology has the potential form a significant part of the meta-knowledge model required at the formal level of representation, and the shared methods could assist code
creation at the detailed implementation level. A more detailed assessment of the meta-
knowledge required for manufacturing facility representations in design systems would
however be required for each level of representation. This should include the procedures
needed to perform validation and verification. It is likely that the conceptual level meta-
knowledge model is likely to focus on the requirements of the end-user, and how these
should be expressed in an initial specification. Additional research would be needed to
develop these concepts into a validation methodology specifically for design and
manufacturing support systems.

![Diagram of the MK Model](image)

**Figure 7: Adaptation of the MK Model (Plant and Gamble, 2003).**

**Acknowledgements**

This work is part of a research project entitled “Knowledge Representation and Reuse for
Predictive Design and Manufacturing Evaluation”. This has been funded under EPSRC
GR/R64483/01, and actively supported by Rolls Royce plc and BOC Edwards Ltd.

**References**


