



HAL
open science

Manufacturing Knowledge Verification Design Support Systems

Sean David Cochrane, R I M Young, K Case, J A Harding, Shilpa Dani,
James X Gao, David Ian Baxter

► **To cite this version:**

Sean David Cochrane, R I M Young, K Case, J A Harding, Shilpa Dani, et al.. Manufacturing Knowledge Verification Design Support Systems. International Journal of Production Research, 2009, 47 (12), pp.3179-3204. 10.1080/00207540701802452 . hal-00513016

HAL Id: hal-00513016

<https://hal.science/hal-00513016>

Submitted on 1 Sep 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Manufacturing Knowledge Verification Design Support Systems

Journal:	<i>International Journal of Production Research</i>
Manuscript ID:	TPRS-2006-IJPR-0758.R2
Manuscript Type:	Original Manuscript
Date Submitted by the Author:	13-Jul-2007
Complete List of Authors:	Cochrane, Sean; Loughborough University, Wolfson School Young, R I M; Loughborough University, Wolfson School of Mech and Man Engineering Case, K; Loughborough University, Wolfson School of Mech and Manufacturing Engineering Harding, J A; Loughborough University, Wolfson School of Mech and Man Engineering Dani, Shilpa; Loughborough University, Wolfson School of Mech and Man Engineering Gao, James X; Cranfield University, School of Industrial and Manufacturing Science Baxter, David; Cranfield University, Department of Manufacturing
Keywords:	PROCESS PLANNING, SIMULATION, RE-USE, PRODUCT PLANNING, PROCESS MODELLING, MANUFACTURING PROCESSES, FACILITY PLANNING, ENTERPRISE MODELLING, DISTRIBUTED RESOURCE PLANNING
Keywords (user):	

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60



For Peer Review Only

Manufacturing Knowledge Verification in Design Support Systems

Sean Cochrane^a, Robert Young^a, Keith Case^a, Jennifer Harding^a,
James Gao^b, Shilpa Dani^a, David Baxter^c

^a Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University, Leicestershire, LE11 3TU, UK

^b School of Engineering, University of Greenwich Chatham Maritime, Kent, ME4 4TB, UK

^c Centre for Decision Engineering, Manufacturing Department, Cranfield University, Bedfordshire, MK43 0AL, UK

Abstract

This paper identifies the need for a *verification methodology* for manufacturing knowledge in design support systems; and proposes a suitable methodology based on the concept of *ontological commitment* and the PSL ontology (ISO/CD18629). The use of the verification procedures within an overall system development methodology is examined, and an understanding of how various categories of manufacturing knowledge (typical to design support systems) map onto the PSL ontology is developed. This work is also supported by case study material from industrial situations, including: the casting and machining of metallic components. The PSL ontology was found to support the

1
2
3 verification of most categories of manufacturing knowledge, and was shown to be
4 particularly suited to process planning representations. Additional concepts and
5
6 particularly suited to process planning representations. Additional concepts and
7
8 verification procedures were however needed to verify relationships between products
9
10 and manufacturing processes. Suitable representational concepts and verification
11
12 procedures were therefore developed, and integrated into the proposed knowledge
13
14 verification methodology.
15

16
17
18 **Key Words:** Knowledge Representation, Validation and Verification, Enterprise
19
20 Modelling, Ontology, Design for Manufacture.
21
22
23

24 25 26 **1. Introduction**

27
28
29 Many organisations use information systems to support decision makers in design and
30
31 manufacture (Young 2003). These systems (i.e. *design support systems*) often use models
32
33 of products and manufacturing processes to predict (and therefore avoid) manufacturing
34
35 issues in design. This paper describes a knowledge verification methodology for such
36
37 systems that has the ultimate aim of reducing the time and energy needed to represent
38
39 manufacturing knowledge. The scope of the methodology is (at this stage) limited to
40
41 existing types of design support system, and the test cases described by the paper are
42
43 based on typical support functions (principally the simulation of process plans derived).
44
45
46
47 The following sections outline current research into system development methodologies,
48
49 and describe why a manufacturing knowledge verification methodology is required.
50
51

52
53 System development methodologies generally follow the stages of elicitation and
54
55 representation. *Elicitation* refers to the learning, uncovering, extracting, surfacing, and/or
56
57 discovering the needs of customers, users, and other potential stakeholders (Hickey and
58
59
60

1
2
3 Davis, 2002). During elicitation, a systems engineer must understand the end user's
4 requirements and elicit product and process knowledge from the appropriate experts.
5
6 Elicitation is often complicated by the fact that an expert's knowledge may be *implicit*
7
8 (i.e. can not be easily described). The elicitation process leads to the further task of
9
10 *representation*. This usually starts with an *informal* description of elicited knowledge and
11
12 user requirements, followed by further stages of *structuring* and *formalisation*. The end
13
14 result of the formalisation process is a computer executable representation of expert
15
16 knowledge that can be used to evaluate designs (according to the requirements elicited
17
18 from the designer).
19

20
21 Knowledge validation and verification are closely related to the tasks of elicitation and
22
23 representation. *Validation* involves making the right system, and *verification* involves
24
25 making the system right (O'Keefe and O'Leary, 1993). Validation must therefore ensure
26
27 that a system meets the requirements of end-users, and can therefore be seen as part of
28
29 the elicitation process. Verification ensures that a system meets its specified requirements
30
31 (Preece, 2001), and is therefore part of representation (or more specifically,
32
33 formalisation). Within a design support system, verification must therefore ensure that
34
35 any rules and constraints used to support decisions are at least consistent. *Indeed, from*
36
37 *any contradictory knowledge, an agent would be able to deduce any conclusion, and it's*
38
39 *contrary* (Gregoire and Mazure 2002).
40
41
42
43
44
45
46
47
48

49 Models for representing manufacturing process knowledge have been extensively
50
51 discussed in the research literature since the 1990's. Sormaz and Khoshnevis (1997) for
52
53 example, provide an object-oriented knowledge representation of process planning
54
55 knowledge, and Sormaz et. al. (2004) describes the interaction between process planning
56
57
58
59
60

1
2
3 models and product design activities. Other work has focussed on developing modelling
4 hierarchies for product and manufacturing information (e.g. Oldham et. al. 1998, Molina
5 and Bell 1999, and Zhao et. al. 2000). Molina and Bell (1999) for example, developed an
6 object-oriented model for manufacturing knowledge based on the concept of a
7
8 *manufacturing strategy*. Strategies were themselves categorised according to four types,
9
10 i.e. planning, capacity, technology, and facility. *Planning* strategies describe rules for
11
12 creating and manipulating process plans, *capacity* strategies describe how many units can
13
14 be produced by a facility, *technology* strategies interpreted the information associated
15
16 with facilities (e.g. machining tolerances), and *facility* strategies described how and when
17
18 facilities should be used to achieve manufacturing objectives.
19
20

21
22 Several reference models (or *frameworks*) have also been developed to assist system
23
24 development, including: CommonKADS (Schreiber et. al. 1999), CIMOSA (Kosanke et.
25
26 al. 1999), and the Reference Model for Open Distributed Processing (ISO/IEC 10746-1).

27
28 A framework specifically for design support systems (referred to as the CAE-RM) was
29
30 also developed by Molina and Bell (2002). All of these frameworks apply some form of
31
32 information view, where knowledge is classified according to a predefined hierarchy.
33
34

35
36 More recent research has focused on the representation of globally distributed supply
37
38 chains (Liu and Young, 2004), and improved knowledge sharing between design teams
39
40 using ontologies (Lin et. al. 2004). Various computational techniques have also been used
41
42 to improve the process planning capabilities of decision support systems, including
43
44 artificial intelligence techniques (Fernandez et. al., 2005), multi-agent techniques
45
46 (Pechoucek et. al 2005), and simulated annealing algorithms (Bramall et. al. 2003).
47
48
49
50
51
52
53
54
55

56
57 Recent work focussed on the representation of process plans has also been performed by
58
59
60

1
2
3 Bock and Gruninger (2004)b. This shows how a formal ontology, i.e. the Process
4 Specification Language (PSL - ISO/CD18629) can be used to describe process plans.
5
6 Knowledge verification has also been tackled from a general “computer science”
7
8 perspective. Plant and Gamble (2003) show how verification can be supported within a
9
10 system development framework (sometimes referred to as a meta-knowledge
11
12 framework); and several techniques for identifying inconsistencies in formal knowledge
13
14 representations (e.g. contradictory rules, and unreachable conditions) have been identified
15
16 (Preece et. al. 1992, Wu and Lee 2002, and Gregoire and Mazure 2002). An approach
17
18 described as *ontological commitment* (Waterson and Preece, 1999) is also described.
19
20
21 *Ontologies* provide a set of rules and constraints associated with a class schema for
22
23 describing an environment (Smith et. al. 2003). Ontological commitment means that a
24
25 knowledge base complies with the rules and constraints associated with the ontology. The
26
27 interaction between several knowledge bases can also be verified against a shared
28
29 ontology. Bock and Gruninger (2004)b show how process plans can be represented by a
30
31 formal ontology (i.e. PSL). They do not however show how the semantics of the PSL
32
33 ontology could be applied within a complete system development methodology (that
34
35 includes knowledge verification). Section 2 of this paper describes such a methodology.
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

2. The Knowledge Verification Methodology

Methodology Overview

This section outlines a *Knowledge Verification Methodology* (KVM) that incorporates the concepts of ontological commitment and the PSL ontology. The KVM is part of a meta-knowledge framework for developing design support systems. The framework provides a series of views (i.e. enterprise, information, and computational) which are based on the CAE-RM (Molina and Bell, 2002). The framework's novelty lies in its improved application of standards (i.e. the ISO/IEC 10746-1), and its direct support for manufacturing knowledge verification. Figure 1 shows the main stages of the framework, and its role within a wider system development methodology.

Figure 1: System Development Methodology

The enterprise view describes how to informally represent user requirements and expert knowledge, and the enterprise model is structured according to the hierarchy provided by the information view. The information model is in turn formalised using the guidelines and definitions provided by the computational view. The resulting computational model can then be used by a decision support system to simulate manufacturing strategies.

Any model inevitably makes assumptions (and simplifications). The question therefore arises as to how detailed a description is required, and whether a valid set of assumptions has been established. This is a particularly difficult part of knowledge elicitation, as product and manufacturing experts can often not directly state these assumptions (as they are understood implicitly, rather than laid down by any explicit set of guidelines). This research therefore follows the iterative model of software development, as discussed by

1
2
3 Hickey and Davis (2002). This results in a series of successively more sophisticated
4
5 descriptions.
6
7

8 9 **Figure 2: The Meta Knowledge Framework**

10
11 The iterations effectively stop when agreement is reached between end users and experts
12 on the validity of the model (and supporting test cases). The iterations are therefore
13 controlled by the validation stage (shown in figure 1), which reviews the enterprise model
14 and verified simulation results with designers and experts. It should be noted that for the
15 purposes of this research, the iterations stop short of defining a full system requirement.
16
17 The experimental case study described in section 5 provides a proof of concept for the
18 KVM, and the results of the first pass validation stage are discussed in section 6.
19
20
21
22
23
24
25
26

27
28 The KVM is based on the the well established use of *test cases*, i.e. derived solutions
29 from domain experts (Giarratano and Riley, 1998). Test cases are elicited from domain
30 experts, and reviewed to ensure that they accurately and adequately describe the required
31 system behaviour (i.e. they validate the system). Each test case describes the information
32 that an expert expects the system to generate for given sets of product requirements.
33
34 Verification is achieved by comparing these expected results with the actual output of the
35 executable model. The Computational View provides specific support for this validation
36 in process. A more detailed description of the enterprise, information and computational
37 views is shown in figure 2, and is discussed in the following subsections.
38
39
40
41
42
43
44
45
46
47
48

49
50 The scope of the methodology does not at this stage include advanced techniques for
51 generating and optimising process plans (such as those developed by Bramall et. al.
52 2003), but relies instead on the manual mapping of structured informal representations of
53 manufacturing knowledge onto a computational model.
54
55
56
57
58
59
60

The Enterprise View

The enterprise view applied by this research is well documented by previous researchers, notably Molina and Bell (2002). The use of diagramming formats for enterprise views is also described by Dorador and Young (2000), and Costa et. al. (2001). These papers describe how IDEF0, IDEF3, UML use case, and UML sequence diagrams can be used to provide an informal representation of user requirements, and manufacturing strategies.

These are used as part of the enterprise view, in the following ways:

- 1 Use case and sequence diagrams are first used to describe designer requirements, and their interactions with the decision support system. These representations follow well-documented procedures for the UML diagramming conventions (Quatrani, 1998).
- 2 An informal description of the product(s) being designed, including a hierarchical breakdown of features (e.g. slots, holes, and grooves), and how they combine to form products.
- 3 A description of how products are made, including a breakdown of manufacturing strategies. IDEF3 process and object schematics assist this description, and this is structured according to the manufacturing model described by the information view.

The Adapted Product/Manufacturing Model

The information view used by this research is shown in figures 3a and 3b. These are based on the product and manufacturing models proposed by Molina and Bell (1999) and Zhao et. al. (2000). Figure 3a shows how facilities describe an aggregation of strategies that use resources to perform processes. In this way, the manufacturing model represents process hierarchies, e.g. machine tools performing drilling processes, but also describes

1
2
3
4 *strategies*, for deploying processes. The adaptations made to the product and
5
6 manufacturing models used by this research are described below.
7

8
9 Firstly, figure 3b shows how the Structure class aggregates geometric characteristics and
10
11 enumerated properties. Product features such as holes, cylinders, blocks, and threads, are
12
13 represented by specialisations of the Structure class (note that these are not shown in
14
15 figures 3a or 3b). A cylinder for example, can be represented by a particular mix of
16
17 geometries, e.g. diameter, depth, and surface tolerance; and enumerated properties, e.g.
18
19 “material: iron or aluminium”. Methods for deriving geometric properties (e.g. area and
20
21 volume) are also supported by the Structure class, and these are polymorphed by each
22
23 Feature specialisation. Figure 2 shows the role of the feature library in the computational
24
25 model. This stores a range of features such as holes, cylinders, blocks, and threads. The
26
27 description of each feature is derived where possible from the STEP standard ISO/DIS
28
29 10303-224.3).
30
31
32
33
34

35 Component can be described as an aggregation of features (e.g. a bolt consisting of a
36
37 metallic cylinder, and a thread). The concepts of atomic and complex components are
38
39 also introduced. Atomic components describe the lowest (atomic) level of component
40
41 found in a product description (e.g. work pieces, plus nuts, bolts, and screws). Complex
42
43 components describe aggregations of components (e.g. assemblies of work pieces).
44
45
46

47 **Figure 3a: Adapted Product/Manufacturing Model**

48 **Figure 3b: Adapted Product/Manufacturing Model**

49
50
51
52 Secondly, previous manufacturing models have used several types of facility class to
53
54 represent different levels of facility, e.g. enterprise, factory, shop and machine classes.
55
56 These are replaced in figure 3 with a simpler hierarchy based on just complex and atomic
57
58
59
60

1
2
3 facilities. As with structures, atomic facilities represent the lowest level of granularity in
4 the representation of facilities, and complex facilities represent aggregations of facilities.
5
6 As complex facilities are also a specialised form of atomic facility, they can also deploy
7 strategies of their own. As a general guideline each facility representation should make
8 no assumptions about the availability of other facilities, unless they are an aggregated
9 sub-facility (i.e. part of the complex model). A machine shop can for example be
10 considered as an aggregation of machine tools (and is therefore a complex facility). The
11 machine shop can also be considered as a facility in its own right, with strategies that
12 cannot be described at the level of individual machine tools. For example strategies for
13 selecting the “best” machine tool for a particular task cannot be described at the level of
14 an individual machine tool (which has no knowledge of other machine tools to make an
15 effective comparison). A machine tool model can be considered as atomic (if this is the
16 lowest level of granularity selected by the modeler), but may also be broken down into
17 further atomic elements (e.g. individual cutting tools, and fixtures). Again this level of
18 granularity needs to be selected on a case by case basis, but is not limited here by the
19 manufacturing model.
20
21

22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41 Thirdly, (as with previous manufacturing models) strategies are performed by facilities.
42
43 Each strategy must however hold an objective describing what it does within the
44 manufacturing environment (e.g. improve the tolerance of a metallic surface). This
45 adaptation is derived from intelligent agent theory (Arazy and Woo, 2002), and allows
46 strategies to be translated into executable modules (or agents). Strategies are further
47 broken down into capacity, planning, technology and facility rules. These four sub-
48 classes of rules are based on the categories of manufacturing strategy described by
49
50
51
52
53
54
55
56
57
58
59
60

1
2
3 Molina and Bell (1999). Note that strategies (in the hierarchy shown above) are
4
5 effectively aggregations of rules.
6
7

8
9 Capacity rules describe how resources are consumed and demanded. Planning rules
10 describe how strategies relate to production schedules. Technology rules describe how
11 strategies change product characteristics and are constrained by the manufacturing
12 environment. Facility rules describe when a particular strategy should be applied (note
13 that these should only be applied to compound facilities, as atomic facilities can not make
14 assumptions about other facilities within the manufacturing environment).
15
16
17
18
19
20
21
22

23 Finally, figure 3b shows interfaces to CAD and MIS platforms. These provide the
24 information relevant to product representations (e.g. dimensions and required tolerances),
25 and manufacturing strategies (e.g. machining times and achieved tolerances).
26
27
28
29
30

31 ***The Computational View and the Role of the SM-API***

32
33 The computational view provides a set of methods for representing manufacturing
34 strategies that directly assist the verification of computational models. Previous
35 frameworks for decision support systems (including the CAE-RM) invariably rely on a
36 system engineer's implicit understanding of the terms used to describe manufacturing
37 strategies to generate computational models. The systems engineer is also left to develop
38 his own implicit understanding of what constitutes an inconsistency between statements
39 in a knowledge base. This implicit "system engineering knowledge" drives the
40 interpretation of the enterprise and information models during formalisation and
41 verification; and can be the source of miss alignments between shared knowledge bases.
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
The KVM tackles this issue by providing an enhanced computational view as part of the
meta-knowledge framework shown in figure 2.

1
2
3 The enhanced computational view includes a set of methods for formally representing
4 manufacturing strategies, referred to as the Shared Methods – Application Programming
5 Interface (SM-API). The SM-API is implemented as a class, and once instantiated this
6 class provides an object that can be used to represent the manufacturing environment
7 being modelled. Statements about the environment are made via a set of method
8 (implemented by the SM-API class). These provide a convenient interface for systems
9 engineers who are familiar with object-oriented programming languages and the concept
10 of an API. The SM-API also allows systems engineers to model manufacturing
11 environments using widely available coding and simulation tools for languages such as
12 Java and C++.

13
14
15
16
17
18
19
20
21
22
23
24
25
26
27 The SM-API is described as “shared” because it provides a common interpretation
28 (between systems engineers) of the terms used to formalise manufacturing strategies. The
29 shared methods also include explicitly defined procedures for detecting inconsistencies
30 between statements in knowledge bases. This allows shared knowledge to be verified
31 against a common set of procedures, rather than relying on each system engineer’s
32 implicit understanding of exception handling within computational models.

33
34
35
36
37
38
39
40
41
42 Figure 4 shows the role of the SM-API within the verification process. Manufacturing
43 strategies are expressed using the shared methods, and are organised according to the
44 information hierarchy shown in figure 3, i.e. they are split between atomic and complex
45 strategies. The overall objective of each strategy is to make the “simulated product
46 model” match the “required product model” as closely as possible. The required product
47 model represents the requirements specified by the designer, and the simulated product
48 model represents the results of manufacturing strategy (and is changed by each process
49
50
51
52
53
54
55
56
57
58
59
60

1
2
3 simulation). This allows the effects of different manufacturing strategies to be evaluated
4
5 for a range of product specifications. For the SM-API to support manufacturing strategy
6
7 formalisation and verification, its methods must be able to describe the entities associated
8
9 with products and manufacturing processes. It must also provide a means of extracting
10
11 the information needed by designers (e.g. machining durations) and be able to generate
12
13 relevant exceptions (i.e. error conditions). The SM-API is also responsible for generating
14
15 the simulation results required by designers (see figure 1). These requirements will have
16
17 been defined by the enterprise model (use case and sequence diagrams), and have been
18
19 elaborated on by the test cases developed during knowledge elicitation and validation.
20
21
22
23
24

25 **Figure 4: The Use of Test Cases to Support System Verification**

26
27 It should also be noted that errors can be generated for a variety of reasons. Error
28
29 conditions may for example highlight inconsistencies in manufacturing strategies. A
30
31 strategy for improving the tolerance of a hole in a work piece may for example attempt to
32
33 bore the hole before the hole has been created. The systems engineer would need to
34
35 examine these error conditions, and correct the enterprise, information, and/or
36
37 computational models accordingly. Other error conditions may be a valid system
38
39 response, as defined by a test case (which should be examining boundary conditions
40
41 where manufacturing strategies are not capable of meeting product specifications). The
42
43 error conditions generated by the SM-API therefore provide input to the verification
44
45 process (in terms of unexpected errors), and the validation process (in terms of verified
46
47 simulation results and valid errors).
48
49
50
51
52
53
54

55 **3. The Verification of Manufacturing Strategies**

1
2
3 This section provides a more detailed examination of the four categories of
4 manufacturing strategy identified by Molina and Bell (1999), i.e. planning, capacity,
5 technology, and facility; and shows how these relate to the constructs provided by the
6 PSL ontology. This analysis is then used to outline the requirements of the SM-API.
7
8
9
10
11

12 *Manufacturing Strategy Categorisation*

13
14
15
16 *Planning* rules describe the creation and manipulation of process plans for the
17 manufacture of a product to a given specification, and enterprise/factory configuration.
18
19

20 This knowledge can be used to estimate how long it takes to manufacture a product, and
21 describes:
22
23

- 24 • Hierarchies of processes and sub-processes, e.g. drilling and milling are all
25 sub-processes of machining.
 - 26 • How processes should be sequenced, e.g. casting precedes machining, and
27 setting must occur before a work piece can be milled.
 - 28 • How to calculate the duration of a process. This is often a function of a
29 processing rate and a geometric feature of a product.
- 30
31
32
33
34
35
36
37
38
39
40

41 Note that the process hierarchy is different from the hierarchy of atomic and complex
42 facilities (shown in figure 3b). Facilities support an aggregation of strategies that use
43 resources to perform processes. A machine tool may therefore support several strategies
44 for drilling and milling (which form part of a wider process of machining). The
45 machining process is also likely to be controlled by a complex facility model of a
46 machine shop. This will aggregate rules for selecting between individual machine tools,
47 describing how the overall machining process can be optimised.
48
49
50
51
52
53
54
55
56
57
58
59
60

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

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 rules describes how many units can be produced by a facility given the availability of resources. This includes an understanding of how long each process takes, the resources it requires, and the number of products it creates/alters. For example, a milling process requires the use of a machine tool, and operates on a single work piece at a time. A casting process on the other hand, requires a foundry, but may produce a batch of casts, in a single operation. Capacity knowledge also describes which processes are best for certain volumes of production. It may for example be better to produce a more accurate cast if high volumes are required, as this reduces the amount of machining needed to achieve the required tolerances, but can require a higher initial outlay.

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

- Rules for maximum and minimum part dimensions, e.g. the foundry cannot cast a part larger than 2m x 2m.
- Relationships between processes and tolerances (e.g. the surface tolerance after grinding equals 25 μ m after grinding, and the time it takes to grind a given area).

- Limits on which processes can be performed on different materials, and different tolerance capabilities for different materials.

It should also be noted that planning strategies often refer to (or make use of) technology rules. For example, the time it takes to grind a surface is often calculated as a function of the area being ground (as defined in the product requirement), and the capability of the machine tool (as described by the technology rules associated with the machine tool).

Facility rules describe how and when processes 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 milling”. As with planning constraints, facility knowledge needs to be appropriate to the level of facility being described. 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. Facility rules are therefore best described at the complex (rather than atomic) level of facility representation.

Commitment to the PSL Ontology

The SM-API requires a basis for describing entities within a product/manufacturing environment, and a set of guidelines for detecting inconsistent strategy representations (i.e. error conditions). This can effectively be performed by a suitable ontology. The introduction to this paper highlighted the PSL ontology (ISO/CD18629) as being potentially suitable for this type of environment. An open standard such as PSL may also provide better support for knowledge sharing than a proprietary ontology (that is not subject to the public scrutiny and control of the ISO). The SM-API therefore to a large

1
2
3 extent bases its definition of methods, and exception handling, on the PSL ontology. Each
4
5 knowledge base formalised by the SM-API is effectively committed to the interpretation
6
7 of the PSL ontology built into the SM-API. This follows the principle of ontological
8
9 commitment described by Waterson and Preece (1999).
10
11

12
13 Figure 5 shows how the principle of ontological commitment is applied. Knowledge
14
15 bases A and B respectively represent two manufacturing facilities. These however, state
16
17 contradictory relationships between timepoints t1 and t2 (which may refer to the
18
19 beginning or ending of processes). The sequence of checks specified by the PSL ontology
20
21 (and implemented by the SM-API) highlights these conflicting statements as a violation
22
23 of the second PSL core axiom, *i.e. the before relationship is a total ordering*. A systems
24
25 engineer would then need to examine the circumstances that led to these conflicting
26
27 statements, and adjust the computational model accordingly. This may for example refer
28
29 to a process that has been scheduled too soon. Further details of how the PSL ontology
30
31 can be used to represent and verify manufacturing strategies are described below.
32
33
34
35
36

37 **Figure 5: Verification Using the PSL Ontology and the SM-API**

38 ***Manufacturing Strategies and the PSL Ontology***

39
40 Whilst other researchers (notably Bock and Gruninger, 2004)^b have shown how
41
42 manufacturing knowledge can be expressed by the PSL ontology, its use as a knowledge
43
44 representation and verification tool within a meta-knowledge framework (for decision
45
46 support systems) is new to this research.
47
48
49
50

51
52 The PSL ontology is centred on a core set of definitions that can be supplemented by the
53
54 PSL outer-core and extensions (see figure 6). The core is relatively straightforward, and
55
56 includes four entity types, *i.e.* activities, activity_occurrences, timepoints, and objects.
57
58
59
60

1
2
3 These can be used to represent processes as occurrences of activities over time.
4

5
6 The PSL ontology describes how timepoints can be ordered using the before relationship
7
8 (see above), and places additional constraints on the before relationship, including
9
10 transitive expansions, i.e. if “t1 is before t2” and “t2 is before t3”, then “t1 is also before
11
12 t3”. Processes can therefore be represented as “occurrences” of activities that are bound
13
14 by two timepoints (i.e. their beginning and ending). Constraints on process sequences can
15
16 therefore be described by stating that the end of one occurrence must be “before” the
17
18 beginning of another (this is referred to as the “precedes” relationship). The verification
19
20 process supported by the SM-API must therefore check process descriptions for
21
22 consistent before relationships (and their associated sequences of processes).
23
24
25
26
27

28 **Figure 6: Structure of the PSL Ontology**

29
30 Planning strategies typically describe hierarchies of processes (as well as sequences). A
31
32 machining process may for example involve several sub occurrences of milling and
33
34 drilling. The work of Bock and Gruninger (2004b) shows that the theories of Sub-
35
36 Activities, Atomic Activities, Complex Activities, and Activity Occurrences are relevant
37
38 to this type of hierarchical description. The SM-API must again check for consistencies
39
40 in the description of sub-occurrences (e.g. if A is a sub-occurrence of B, and B is a sub-
41
42 occurrence of A, then A must be equal to B).
43
44
45

46
47 Further concepts (described in the PSL occurrence tree definitions) can be used to
48
49 represent such constraints. These include the term “*poss*” which describes an activity that
50
51 becomes possible following the occurrence of another activity. Again the SM-API must
52
53 highlight any impossible occurrences by examining the “*poss*” statements made by a
54
55 manufacturing strategy, e.g. if milling is only possible after an occurrence of setting, then
56
57
58
59
60

1
2
3 any attempt to start a milling process without setting should be identified.
4
5

6 The PSL “*durations*” extension may also be used to represent timing aspects of a
7
8 planning strategy. The duration of an activity-occurrence is the difference between its end
9
10 and beginning timepoints. In a computational implementation, “duration” can be
11
12 represented by a long integer denoting the elapsed time between two timepoints.
13
14

15
16 Recent work has also gone into modelling process inputs and outputs, using the PSL
17
18 concept of states (Bock and Gruninger, 2004a). This allows processes to be described
19
20 using statements such as holds and priors (similar to the example shown above). These
21
22 describe states that are held following the occurrence of an activity, or set as a prior
23
24 condition for an occurrence; and these concepts may be particularly useful in describing
25
26 technology and facility strategies.
27
28

29
30 The PSL-concept of states is most effective however, when facility and product
31
32 descriptions are constant (as in many process planning applications). Here, it is often
33
34 enough to describe an object’s state using binary attributes, e.g. a surface has either been
35
36 milled or not milled. Technology strategies however, tend to describe the inputs and
37
38 outputs of processes in terms of geometries. For example, the tolerance held following a
39
40 milling process may be expressed in micrometers. Maximum and minimum constraints
41
42 on a milling machine may also be expressed in metres, or even Kilograms. The selection
43
44 of different processes and facilities (i.e. facility strategies) also depends in many cases on
45
46 numerical comparisons of requirements and outcomes. Here for example it is not enough
47
48 to simply state that a surface is *milled*, as the strategy must also describe the conditions
49
50 under which milling is (and is not) required, and exactly how smooth the component’s
51
52 surface is after the milling occurrence.
53
54
55
56
57
58
59
60

1
2
3 Facility rules describe how to apply resources and processes. Many of the rules
4
5 associated with the representation of these types of rules need an understanding of the
6
7 current state of manufacturing facilities and products, and the required (specified) state of
8
9 the final product. A grinding process may for example be needed if required tolerances
10
11 exceed the capability of the milling process. It is therefore useful to describe “geometric-
12
13 states”, that are held following activity occurrences, rather than limiting representations
14
15 to the existing PSL concept of binary states. A surface tolerance may for example be held
16
17 to 3.0mm after a casting occurrence, and 0.1mm following a rough milling occurrence.
18
19 This allows process models to describe the behaviour of processes more precisely by
20
21 allowing geometric attributes to hold specific values following occurrences and for prior
22
23 conditions on geometric attributes to be set for occurrences.
24
25
26
27
28

29
30 Finally, the PSL theory of resources defines a resource which can be used to describe
31
32 capacity strategies. Any object that is required by some activity – where “activity” and
33
34 “requires” are defined elsewhere in PSL (Cutting Deceelle et. al. 2003). The theory of
35
36 resources also defines the concepts of *available-quantity* and *aggregate-demand*. Broadly
37
38 speaking activities can demand a quantity of resource, and the aggregate-demand is the
39
40 sum of all demands running concurrently with an occurrence. The available quantity of
41
42 resource can also be held to a given value by an activity occurrence, and the aggregate
43
44 demand for a resource must never exceed the available quantity.
45
46
47
48

49 ***The Implementation of the SM-API***

50
51
52 The SM-API has therefore been built in the form of an object-oriented implementation of
53
54 the PSL ontology, with certain additions and interpretations (see figure 7). This allows
55
56 manufacturing strategies to be described in terms of processes using resources to meet
57
58
59
60

1
2
3 manufacturing objectives (see figures 3a and 3b). In summary (and using the lexicon of
4
5 the PSL ontology), the SM-API must be capable of representing and verifying:
6
7

- 8
9 • The occurrence of activities over time, the beginning, ending and duration of
10
11 occurrences; and the constraining of occurrence over time.
- 12
13 • Process hierarchies and descriptions of how sub-occurrences occur within an overall
14
15 process plan, and constraints on the possibility of occurrences.
- 16
17 • Product characteristics and facility conditions expressed in terms of geometries (and
18
19 their associated units), and enumerated properties.
- 20
21 • Assessments of required and manufactured product characteristics, and the
22
23 technology rules associated with atomic strategies.
- 24
25 • Details of how processes change and are constrained by product characteristics and
26
27 facility conditions.
- 28
29 • The demand for, and creation and consumption of resources, during the execution of
30
31 a manufacturing strategy.
32
33
34
35
36
37
38

39 **Figure 7: Shared Model Class Diagram**

40
41 The Timepoints class supports the representation and verification of the PSL concept of a
42
43 timepoint. This allows occurrence activities and the existence of objects to be bound by
44
45 begin and end timepoints, and for constraints on when occurrences/objects begin and end
46
47 to be stated. The intervals between the beginning and ending of occurrence/objects can
48
49 also be expressed (and manipulated) in terms of durations.
50
51

52
53 The Occurrences class allows process hierarchies to be described using sub-occurrence
54
55 relationships; and the OccurrenceTrees class supports the constraining of process
56
57
58
59
60

1
2
3 sequences in using the PSL “poss” relationships. This allows complex strategy
4
5 representations to describe how sub-occurrences form overall process plans.
6
7

8
9 The Structures class allows product characteristics, technology strategies and facility
10
11 strategies to be expressed in terms of geometries (and their associated units), and
12
13 enumerated properties. This extends the PSL concepts of states to support geometric
14
15 states, and geometric “holds” and “prior” relationships. These can be used to describe
16
17 how processes change product characteristics, place constraint on processes based on
18
19 product characteristics and facility conditions.
20
21

22
23 The demand, creation and consumption of resources, during the execution of a
24
25 manufacturing strategy, are supported by the resources class. This includes methods for
26
27 describing aggregated the demand and the availability of resources.
28
29

30 31 **5. Experimental Environment**

32 33 *Scope of the Prototype System*

34
35 This research uses a case study of a jet engine combustion chamber and its manufacturing
36
37 environment (Cochrane 2007) to provide a proof of concept for the proposed KVM. The
38
39 study was limited to a representative, but simplified prototype system. The scope of this
40
41 system was agreed with the designers and experts associated with the study as being
42
43 appropriate for a proof of concept, and shows how the SM-API (with further work) could
44
45 be applied in industrial situations.
46
47
48
49

50
51 The role of the different actors involved in building and using the prototype system
52
53 should also be considered. The designers and experts (mentioned above) help specify the
54
55 system during the elicitation stage of the development methodology shown in figure 1;
56
57
58
59
60

1
2
3 and the designers ultimately use the system to evaluate the manufacturability of new
4
5 products. A “systems engineer” is however responsible for the structuring and
6
7 formalisation stages of the methodology. This involves writing Java based descriptions of
8
9 products, processes and process relationships. This is supported by the SM-API, which
10
11 provides a set of terms (and verification procedures) for describing these relationships.
12
13 This separation of roles allows designers and experts to focus on designing and
14
15 manufacturing products (rather than building information systems).
16
17
18

19
20 A knowledge elicitation review with the designers and experts associated with the case
21
22 study was performed as part of the experiment (see figure 1). During the review, the
23
24 physical form of the combustion chamber was described, and the processes and process
25
26 plans associated with its manufacture were informally documented. This review also
27
28 provided data relating to manufacturing rates, and generated details of suitable test cases
29
30 that could be used to verify a prototype computational model. These test cases
31
32 highlighted the limits of certain processes, and established the dimensions and tolerances
33
34 of chambers that would expose those limits. The product, process and test case
35
36 descriptions generated by this review are detailed in Cochrane (2007), and are
37
38 summarised below.
39
40
41
42

43 44 *Simplified Description of a Combustion Chamber and its Manufacture*

45
46 The chamber is manufactured by an initial fabrication process (either casting or forging),
47
48 which is followed by a series of machining operations. The machining operations create
49
50 additional features such as holes, and improve the tolerances of the chamber’s surfaces.
51
52
53
54

55 **Figure 8: Simplified Combustion Chamber**

56
57 Combustion chambers can be regarded as a series of cylindrical rings with varying inner
58
59
60

1
2
3 and outer diameters (figures 8, 9, and 10). Rings form the main chamber, and additional
4 rings of increased thickness may be added to contain fractures. Each ring takes the form
5 of a conical frustum, and holes may be placed on any part of the ring. The flanges at
6 either end of the chamber are described as rings (with a short length, straight edges and
7 increased thickness). The dimensions and surface areas associated with each ring are
8 shown in figure 6. $R1$ and $R2$ represent the outer radii of each ring, and $r1$ and $r2$ its inner
9 radii. The surface areas calculated for each ring need to be machined by both rough and
10 finish turning processes to achieve the required tolerances (see tables 1 and 2).
11
12

13
14
15
16
17
18
19
20
21
22
23 The tolerance of each ring surface is initially set to the output tolerance of the casting
24 process that was used to create the initial shape, i.e. 3mm. Each surface requires multiple
25 rough turning processes, and a possible finish turning process, to achieve its required
26 tolerance. Under these conditions, each rough turning operation improves the surface
27 tolerance by either 1.0mm, or achieves the output tolerance of 0.4mm. Finish turning will
28 be required when tolerances of less than 0.4mm are specified. Table 2 uses the surface
29 area calculations, machine settings and required tolerances to calculate the total
30 machining time for the chamber. An initial 300 seconds is included for machine setting
31 (this is based on a simplified model of machine setting durations).
32
33
34
35
36
37
38
39
40
41
42
43

44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Figure 9: Ring Faces (Cross Section)

Figure 10: Ring Section (Length)

The Combustion Chamber Test Case

Tables 1, 2 and 3 below, show a series of calculations made for machining chambers to specified dimensions and tolerances. The prototype system built as part of this research needs to show (during system verification) that it can generate these results. These test

1
2
3 cases where developed as part of the knowledge elicitation process with designers and
4
5 experts, and provide a basic set of calculations (with expected results).
6
7

8
9 Table 3 shows the machining times for two holes in the surface of the cylinders. These
10
11 calculations use the volumes of each hole to estimate drilling durations, and the surface
12
13 area of each hole to estimate boring and reaming durations. Both holes have a diameter
14
15 greater than 15mm, and therefore require pilot drilling processes. This brings the total
16
17 machining time (including all rough turning, finish turning, drilling, boring and reaming
18
19 processes) to 1.4 days, or 6 days and 23hrs for five units.
20
21
22

23 **Table 1: Surface Area Calculations**

24
25 **Table 2: Rough Machining and Turning Durations**

26
27
28 **Table 3: Hole Specification and Machining Durations**

29
30 The diameter, depth and position tolerances of the required holes are also shown in table
31
32 3 (Rtol, Dtol, and Ptol respectively). Depending upon the selected manufacturing facility,
33
34 positioning errors may be observed for the specified holes if the Ptol tolerance exceeds
35
36 the machine tool's positioning capability. If this tolerance is set to 80 μ m for drilling,
37
38 boring and reaming strategies errors should be expected in the simulation results.
39
40
41

42
43 ***The Manufacturability Analysis Platform***

44
45
46 The Manufacturability Analysis Platform (figure 11) was used to evaluate the Knowledge
47
48 Verification Methodology. Manufacturing strategies (describing casting, and machining
49
50 facilities) were expressed using the terms described by the SM-API. These were
51
52 implemented as Java methods, which also perform the verification procedures described
53
54 above (based on the axioms of the PSL ontology).
55
56
57
58
59
60

1
2
3 The MAP uses a tiered architecture to separate information describing processes and
4
5 resources (stored in relational databases and/or spreadsheet tables) from rules and
6
7 constraints (coded in Java and expressed using the shared terms/methods). The platform
8
9 also implements required and manufactured product models; and separates atomic
10
11 strategies (e.g. drilling individual holes and machining surfaces) from complex strategies
12
13 for manufacturing whole components.
14
15

16
17
18 Each manufacturing strategy was generated manually (rather than by any form of
19
20 automated technique), but relied heavily on the reuse of atomic facility descriptions, and
21
22 the processes used to generate the required knowledge for each model are described in
23
24 section 3 (starting with the elicitation of user requirements).
25
26

27 28 **Figure 11: The Manufacturability Analysis Platform (MAP)**

29 30 ***Simulation Results***

31
32
33 Figure 12 shows the results of a manufacturing strategy simulation. Each simulation
34
35 examines a set of product requirements, and constructs a model of the processes and
36
37 resources needed to manufacture the product(s). Work pieces are for example instantiated
38
39 (using the Structure class), and their geometries and properties are manipulated by a
40
41 series of activity occurrences. The final work piece(s) are then compared against the
42
43 original requirement. The manipulations performed by processes are derived from the
44
45 “holds” relationships that have been described by the manufacturing strategies (using the
46
47 SM-API). Each simulation highlights any aspects of the model that contradict the axioms
48
49 of the SM-API’s underlying ontology (i.e. PSL). This may for example include violation
50
51 of any prior constraints on occurrences. The model can be updated as and when changes
52
53 to the product requirement are made.
54
55
56
57
58
59
60

1
2
3 The results shown in figure 12 contain unexpected errors in the representation of the
4 machined structure, and in the allocation of resources to the stages of manufacture. These
5 unexpected errors are consistent with the role of the SM-API shown in figure 4, and
6 indicate some form of inconsistent statements in the representation of manufacturing
7 strategies. The cause of each error needs to be traced by the systems engineer responsible
8 for formalising the strategy representations, and corrected. Cochrane (2007) provides a
9 detailed examination of several categories of errors, including violations in the
10 description of:
11
12
13
14
15
16
17
18
19
20
21
22

- 23 • Process sequences, durations and hierarchies
- 24 • Sub-occurrences and possibility trees
- 25 • Product characteristics and technology rules (e.g. geometries)
- 26 • Process inputs (required prior states) and outputs
- 27 • The demand for, and creation and consumption of resources.

28
29
30
31
32
33
34
35
36 Figure 13 shows the corrected machining strategy, and the simulated process durations
37 matching the test case expected results, including the expected simulation errors
38 (predicted by the test case). These provide feedback to the system user (rather than the
39 systems engineer building the system), and indicate that the capability of the selected
40 machine tool needs to be improved, or the required positioning tolerance of the two holes
41 needs to be relaxed. These errors were predicted by the test cases (see previous section).
42
43
44
45
46
47
48
49
50

51 **Figure 12: Unexpected Errors in the Machining Strategy**

52
53 **Figure 13: Verified Machining Calculations**
54
55
56
57
58
59
60

6. Discussion and Conclusions

The main achievement of this research has been the development of an explicit definition of how to represent and verify manufacturing strategies within decision support systems.

This has been achieved within a broader framework for the development of decision support systems based on the CAE-RM. The explicit definition has taken the form of the Shared Model – Application Programming Interface (SM-API), and is based broadly on the PSL ontology (i.e. an open standard). The SM-API was also shown to be able to present planning, capacity, technology and facility strategies.

Certain interpretations, simplifications and developments of the PSL standard were however found to be necessary. These included the extrapolation from the PSL ontology of the concepts such as: geometric states, properties and substructures. These provided a way of constraining process inputs and describing process outputs. A number of limitations were also identified by the case study and associated test cases. These included:

- The need for an “inspection process” to drive the comparison between the required and manufactured product models during strategy simulation. This was not part of the PSL ontology or the SM-API specification.
- The SM-API often replicated error messages, as the examination of transitive timing relationships and possibility trees often lead to the multiple identifications of a single error (this was more of an annoyance when interpreting the simulation outputs than a significant issue with the SM-API).

- The verification processes only identified inconsistent strategies. Consistent strategies can still be based on incorrect data (e.g. machining rates), and so the overall accuracy of the decision support system is still highly dependant on the identification of suitable test cases. Inconsistencies in the modelling of capacity strategies were for example, only identified when processes actually ran out of resource. Additional test cases that predict the availability of resource at instances in time are therefore needed.
- Further improvements in the strategies modelled by the test cases were suggested during the first pass validation stage (see section 3). These included the use of volumes rather than surface areas to calculate rough machining times, and a more accurate representation of machine setting times. The approximations used were however sufficient as a proof of concept.
- The inclusion of cost information in the selection of processes (i.e. facility strategies) is especially important, and should be considered as one of the next stages of development for the SM-API. This will require consideration of how the PSL based ontology can handle concepts related to cost.
- The need to integrate the SM-API with improved techniques for providing feedback to designers and experts during system validation was also identified, as was the need to integrate the SM-API with more automated planning process techniques such as simulated annealing algorithms (Bramall et. al. 2003).
- The need to integrate formal frameworks (such as the SM-API) with decision support systems for the conceptual design stage. These could be based on informal ontologies for the improved search and categorisation of tacit design knowledge.

- Finally, the methodology described by this research does not eliminate the need for test cases derived from domain experts, Giaratanno and Rilley (1998). These are central to the verification approach, and are supported by a set of verification procedures derived from the PSL ontology.

Abbreviations

CIMOSA	Computer Integrated Manufacture Open System Architecture
KBS	Knowledge Based Systems
MOSES	Model Oriented Simultaneous Engineering System
NIST	National Institute of Standards and Technology
OWL	Web Ontology Language
PSL	Process Specification Language
RM-ODP	Reference Model of Open Distributed Processing
STEP	STandard for the Exchange of Product information
UML	Unified Modelling Language

Glossary

Data: symbols (e.g. text and numbers) with no specified meaning or context.

Capability: description of what a manufacturing strategy can (and can not) achieve, expressed in terms of manufacturing processes inputs, outputs, and constraints.

Elicitation: the learning, uncovering, extracting, surfacing, and/or discovering the needs of customers, users, and other potential stakeholders.

Exception handling: the detection and resolution of error conditions highlighting inconsistencies in the computational model of a manufacturing strategy.

Explicit knowledge: includes both informal and formal knowledge representations, e.g.

1
2
3 text, diagrams, procedures, rules and constraints.
4

5
6 **Formalisation:** describes the process of representing informal knowledge in terms of
7
8 executable rules and constraints.
9

10
11 **Formal knowledge:** rules and constraints describing how to apply information, expressed
12
13 in a computer language (e.g. Java), or a formal language (e.g. KIF).
14

15
16 **Framework:** provides a set of guidelines and procedures for developing knowledge based
17
18 decision support systems.
19

20
21 **Implicit knowledge:** can be inferred or implied from observable behaviour, and
22
23 potentially developed into explicit knowledge.
24

25
26 **Informal knowledge:** a natural language (e.g. English) or diagrammatic description of
27
28 how to apply information.
29

30
31 **Manufacturing strategy:** a description of a manufacturing process in terms of: its
32
33 objectives and constraints, and the resources that it demands and consumes.
34

35
36 **Representation:** the process of representing elicited knowledge, initially in terms of a
37
38 structured informal representation, followed by a formal computational model.
39

40
41 **Structuring:** describes the process of representing elicited knowledge in terms of a
42
43 structured but still informal representation.
44

45
46 **Tacit knowledge:** describes informal representations such as videos and transcripts of
47
48 conversations that can not always be formalised.
49

50
51 **Validation:** ensuring that a system meets the requirements of its end users, i.e. making
52
53 the right system.
54
55
56
57
58
59
60

1
2
3 *Verification*: ensuring that a system meets its specified requirements, i.e. making the
4 system right.
5
6
7

8 9 **Acknowledgements**

10
11
12 This work is part of a research project entitled “Knowledge Representation and Reuse for
13 Predictive Design and Manufacturing Evaluation”. This has been funded under EPSRC
14 GR/R64483/01, and actively supported by Rolls Royce plc and BOC Edwards Ltd.
15
16
17
18

19 20 **References**

21
22
23 Bock C, Gruninger M, 2004a. Inputs and Outputs in the Process Specification
24 Language, NISTIR 7152, NIST, Gaithersburg, MD, 2004. Web Accessed 17th February
25
26 2005: <http://www.nist.gov/msidlibrary/doc/nistir7152.pdf>
27
28

29
30
31 Bock C, Gruninger M, 2004b. PSL: A Semantic Domain for Flow Models. Software
32 and Systems Modeling Journal, 2004. Web Accessed 17 Feb 05: www.mel.nist.gov
33
34

35
36 Bullinger H, Lentjes H, Scholtz O, 2000. Challenges and chances for innovative
37 companies in a global information society. International Journal of Production
38 Research, 38(7), 1469–1500.
39
40

41
42
43 Bramall D, McKay K, Rogers B, Chapman P, Cheung W, and Maropoulos P, 2003.
44 Manufacturability analysis of early product designs. Int. J. Computer Integrated
45 Manufacturing, 2003, Vol. 16, No. 7–8, 501–508.
46
47

48
49
50 Chira O, Chira C, Tormey D, Brennan A, and Roche T, 2004. A Multi-agent
51 Architecture for Distributed Design. Lecture Notes in Computer Science. Volume 2744
52
53
54
55
56
57
58
59
60

1
2
3 / 2004 Title: Holonic and Multi-Agent Systems for Manufacturing. ISBN: 3-540-
4 40751-0. Chapter: pp. 213 – 224. Online Date: January 2004.
5
6
7
8 Cochrane 2007. Manufacturing Knowledge Verification in Design Support Systems.
9
10 PhD Thesis, Loughborough University, 2007.
11
12
13 Cutting-Decelle, A, Young R, Anumba C, Baldwin A, Bouchlaghem N, 2003. The
14 Application of PSL in Product Design across Construction and Manufacturing. CERA
15 Journal, Vol 11, No 1, March 2003, pp65-75, ISSN 1063-293X
16
17
18
19
20 Fernandez S, Aler R, and Borrajo D, 2005. Machine Learning In Hybrid Hierarchical
21
22 And Partial-Order Planners For Manufacturing Domains, Applied Artificial
23
24 Intelligence, 19:783–809, 2005
25
26
27
28 Giarratano J, Riley G, 1998. Expert Systems: Principles and Programming, (3rd edn).
29
30 Boston: PWS-Publishing Company.
31
32
33 Gregoire E, Mazure, B, 2002. About the incremental validation of first-order stratified
34 knowledge-based decision-support systems. Information Sciences 142 (2002) 117–129.
35
36
37
38 Gruninger M, Sriram R, Cheng J, Law K, Process Specification Language for Project
39 Information Exchange. International Journal of IT in Architecture, Engineering &
40
41 Construction, 2003.
42
43
44
45 Hickey A, Davis A, 2002. Requirements Elicitation and Elicitation Technique
46
47 Selection: A Model for Two Knowledge-Intensive Software Development Processes.
48
49 Proceedings of the 36th Hawaii International Conference on System Sciences
50
51 (HICSS'03) 0-7695-1874-5/03 2002 IEEE.
52
53
54
55 ISO/CD18629. Industrial Automation System and Integration — Process Specification
56
57 Language: Part 1: Overview and Basic Principles. ISO TC184/SC4/JWG8.
58
59
60

1
2
3 ISO/IEC 10746-1. Reference Model for Open Distributed Processing.
4

5
6 ISO/JTC1/SC32/WG2. Knowledge Interchange Format, Part 1: KIF-Core. WD,
7
8 International Organization for Standardization (ISO).
9

10
11 Kosanke K, Vernadant F, and Zelm M., 1999, CIMOSA: enterprise engineering and
12
13 integration. Computers in Industry, 40, 83–97.
14

15
16 Lin H, Harding J, Shahbaz M, 2004. Manufacturing system engineering ontology for
17
18 semantic interoperability across extended project teams. Int. J. Production. Research, 15
19
20 December 2004, vol. 42, no. 24, 5099–5118.
21

22
23 Liu S, Young R, 2004. Utilizing information and knowledge models to support global
24
25 manufacturing co-ordination decisions. Integrated Journal of Computer Integrated
26
27 Manufacturing, September 2004, Vol. 17, No. 6, 479–492.
28

29
30 Molina A, Bell R, 1999. A Manufacturing Model representation of a flexible
31
32 manufacturing facility. Journal of Engineering Manufacture: Proceedings of the
33
34 Institution of Mechanical Engineers, Vol 213, Part B, pp.225-246.
35

36
37 Molina A, Bell R. 2002. Reference models for the computer aided support of
38
39 simultaneous engineering. Int. J. Computer Integrated Manufacturing, 2002, Vol 15, No
40
41 3, 193-213.
42

43
44 O’Keefe R, O’Leary, D, 1993. Expert system verification and validation: a survey and
45
46 tutorial. Artificial Intelligence Review, 7, 3–42.
47

48
49 Oldham, K, Kneebone, S, Callot, M, Murton, A and Brimble, R, in N. Mårtensson, R.
50
51 Mackay and S. Björgvinsson (eds.), 1998. Changing the Ways We Work. Advances in
52
53 Design and Manufacturing, Vol 8, Proceedings of the Conference on Integration in
54
55
56
57
58
59
60

1
2
3 Manufacturing, Göteborg, Sweden, IOS Press, Amsterdam, October 1998, 198-207.

4
5 Accessed 01/Jun/04: <http://www.kbe.cov.ac.uk/moka>

6
7
8 Pechoucek, M. Vokrinek, J. Becvar, P. , ExPlanTech: multiagent support for
9
10 manufacturing decision making, Intelligent Systems, IEEE Jan.-Feb. 2005 Volume: 20,
11
12 Issue: 1, p. 67- 74

13
14
15 Plant R, Gamble R, 2003. Methodologies for the development of knowledge-based
16
17 systems, 1982–2002* *The Knowledge Engineering Review*, Vol. 18:1, 47–81, 2003.

18
19
20 Preece A. Evaluating Verification and Validation Methods in Knowledge Engineering.
21
22 In R Roy (ed), *Micro-Level Knowledge Management*, Morgan-Kaufman, pages 123-
23
24 145, 2001

25
26
27 Preece A, Shinghal R, Batarekh A, 1992. Principles and practice in verifying rule based
28
29 systems. *Knowledge Engineering Review*, 7(2):114-141, 1992.

30
31
32 Schreiber G, Akkermans H, Anjewierden A, de Hoog R, Shadbolt N, Van de Velde W,
33
34
35 Wielinga B, December 1999. The CommonKADS Methodology. ISBN 0-262-19300-0.
36
37 <http://www.commonkads.uva.nl>

38
39
40 Smith M, Welty C, McGuinness D, 2003. Web Ontology Language (OWL) Guide
41
42 Version 1.0. W3C Working Draft 10 February 2003. Accessed 14Jul/03:
43
44 <http://www.w3.org/TR/2003/WD-owl-guide-20030210>

45
46
47 Sormaz D, Arumugam J, Rajaraman S, 2004. Integrative Process Plan Model and
48
49 Representation for Intelligent Distributed Manufacturing Planning, *International*
50
51 *Journal of Production Research*, Vol. 42, No. 17, p. 3397 - 3417, 2004

1
2
3 Sormaz D, B. Khoshnevis B, 1997. "Process Planning Knowledge Representation using
4 an Object-oriented Data Model", International Journal of Computer Integrated
5
6
7
8 *Manufacturing*, Vol. 10, No. 1-4, p. 92-104, 1997.

9
10
11 Uschold M, Gruninger M, 1996. Ontologies: Principles, Methods, and Applications.
12
13 Knowledge Engineering Review, 1996, Vol. 11, pp. 96-137.

14
15
16 Waterson A & Preece A, 1999. Verifying Ontological Commitment in Knowledge-
17
18 Based Systems. *Knowledge-Based Systems*, 12, 45-54, 1999.

19
20
21 Wu C, Lee S, 2002. KJ3: a tool assisting formal validation of knowledge based
22
23 systems. *Int. J. Human-Computer Studies* (2002) 56, 495–524

24
25
26 Young R, 2003. Informing decision makers in product design and manufacture.

27
28 International Journal of Computer Integrated Manufacturing, Vol. 16, No, 6, pp. 428-438.

29
30 Zhao J, Cheung W, Young R, 2000. The influence of manufacturing information models
31
32 on product development systems, EDC Conference, 2000.

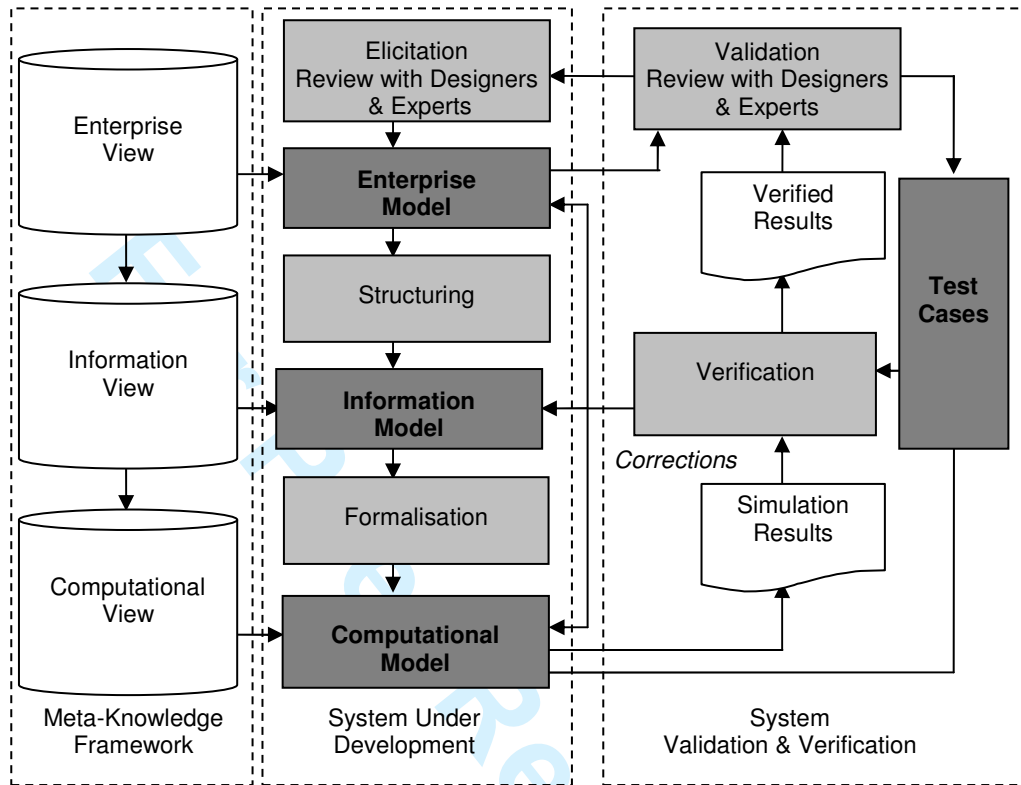


Figure 1: System Development Methodology

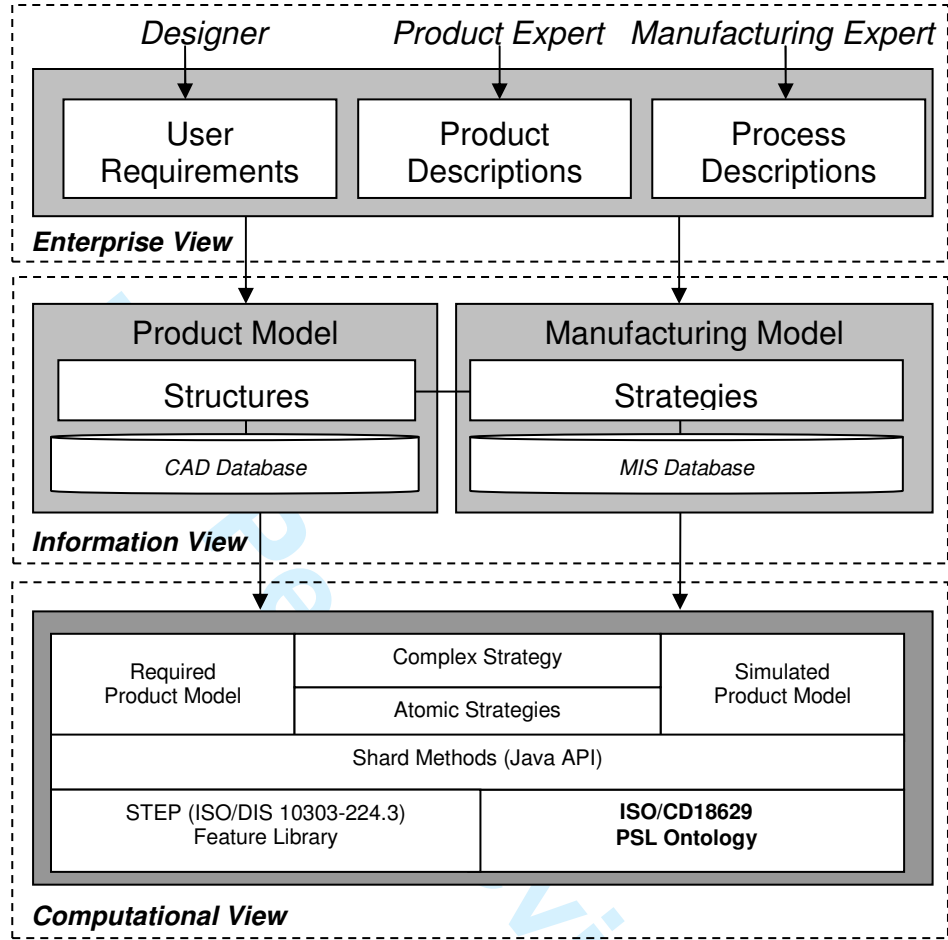


Figure 2: The Meta Knowledge Framework

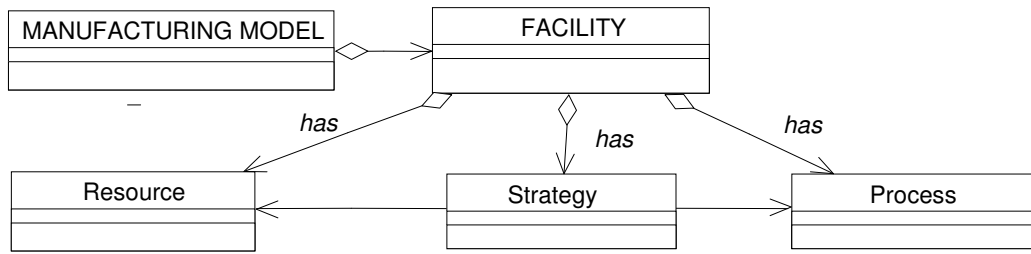


Figure 3a: Adapted Product/Manufacturing Model

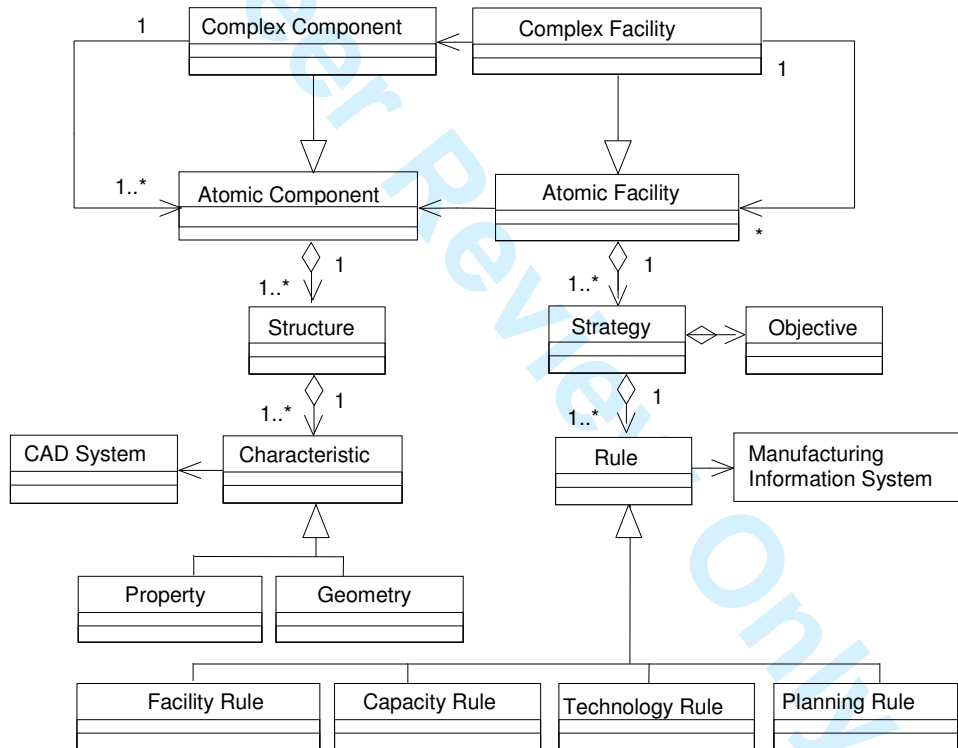


Figure 3b: Adapted Product/Manufacturing Model

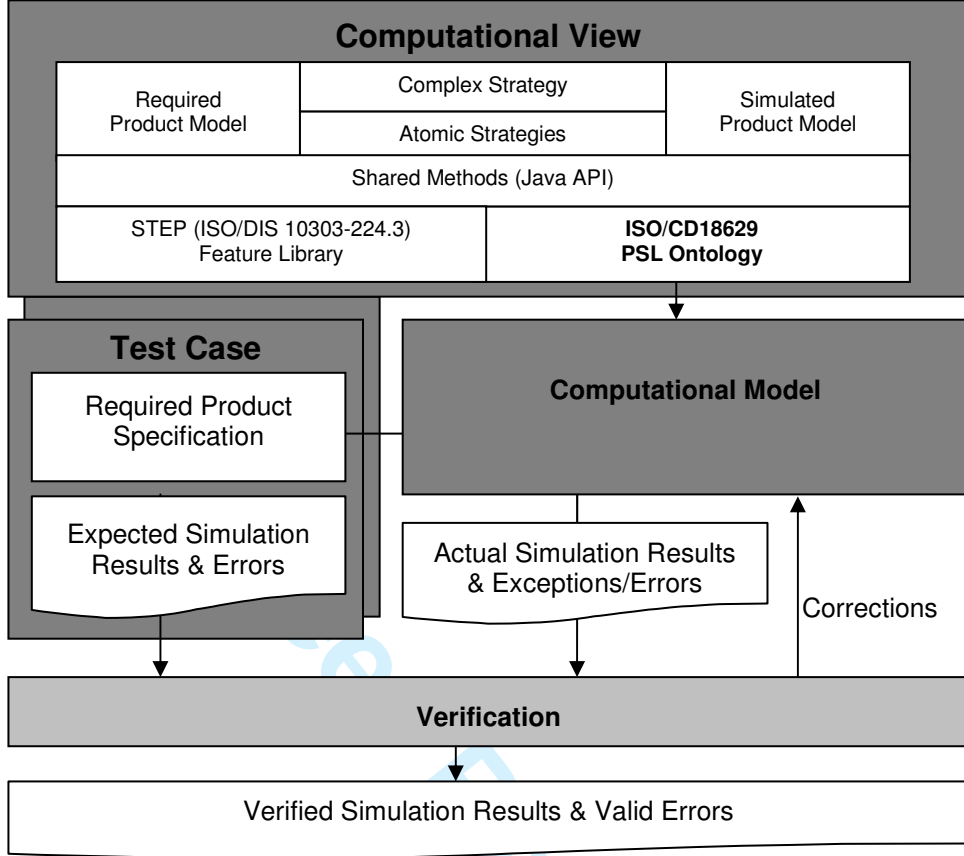


Figure 4: The Use of Test Cases to Support System Verification

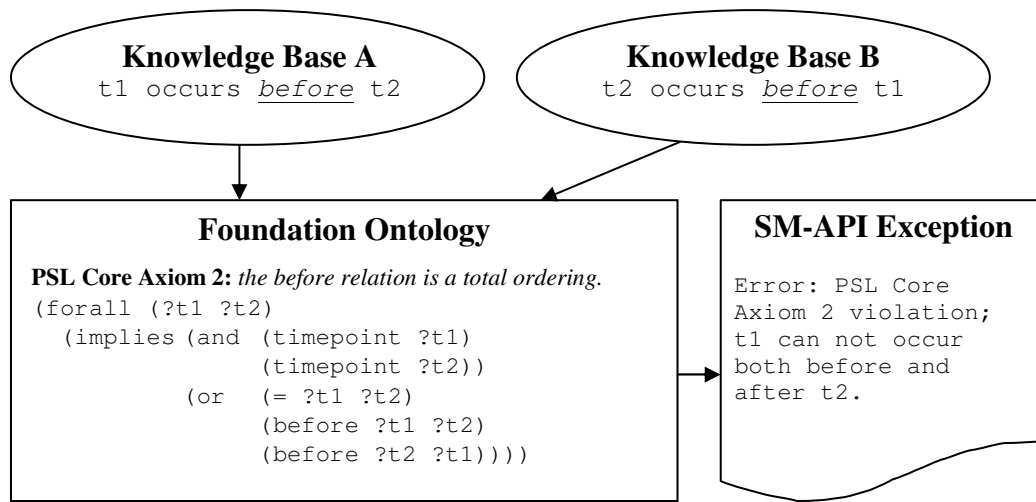


Figure 5: Verification Using the PSL Ontology and the SM-API

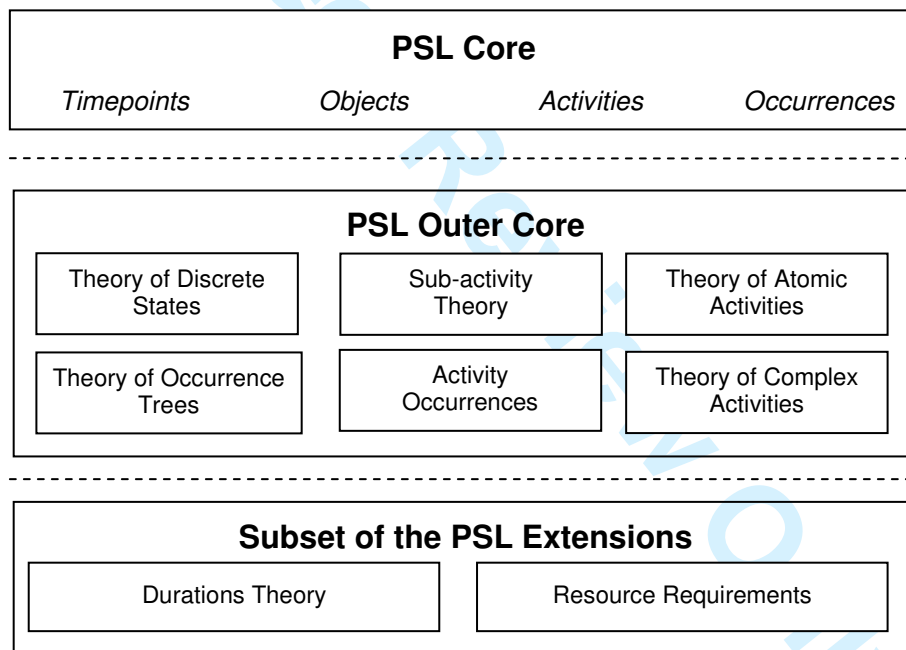


Figure 6: Structure of the PSL Ontology

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

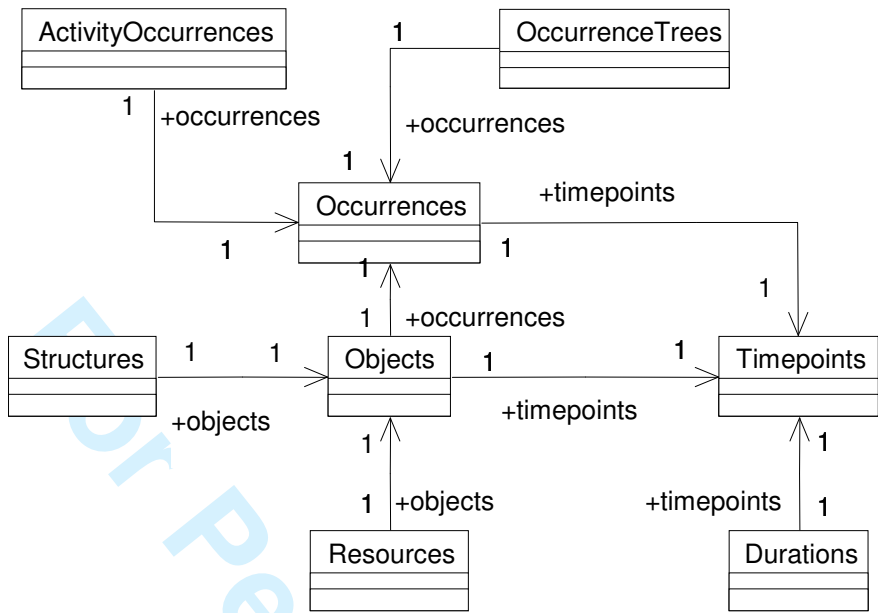


Figure 7: Shared Model Class Diagram

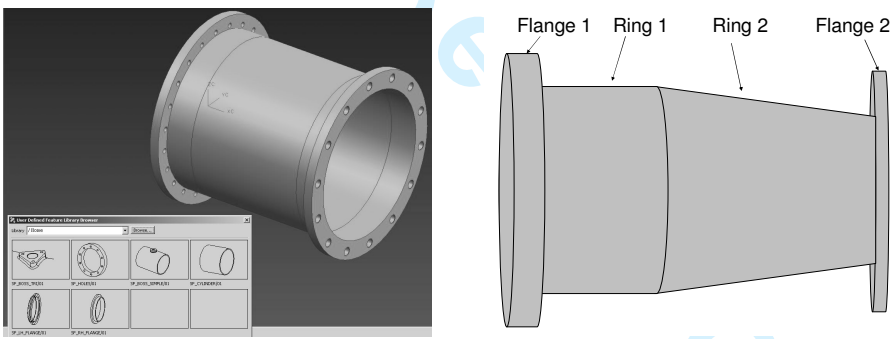


Figure 8: Simplified Combustion Chamber

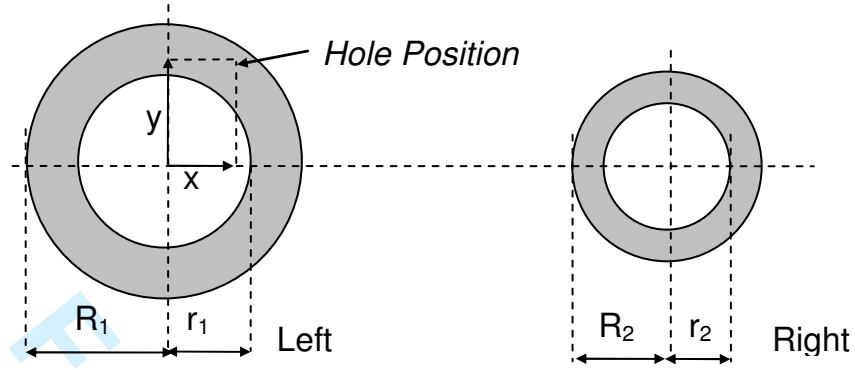


Figure 9: Ring Faces (Cross Section)

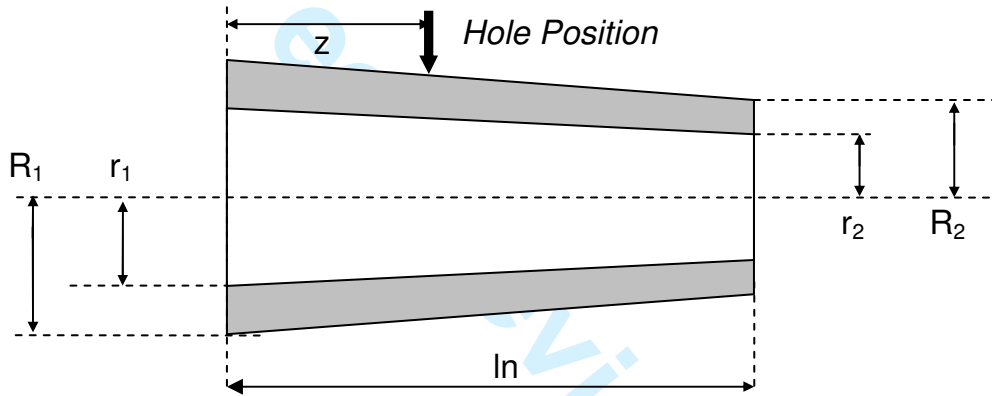


Figure 10: Ring Section (Length)

$$\text{Surface Area} = \pi * (r_1 + r_2) * ((r_1 - r_2)^2 + l_n^2)^{1/2}$$

	R1	R2	ir1	ir2	ln	Face 1	Face 2	Outer	Total	Total
	m	m	m	m	m	Area	Area	Area	Outer	Inner
Flange 1	0.3	0.3	0.22	0.22	0.04	0.131	0.086	0.075	0.292	0.055
Ring 1	0.25	0.35	0.22	0.32	0.1	0.000	0.000	0.267	0.267	0.240
Ring 2	0.35	0.35	0.32	0.32	0.2	0.000	0.000	0.440	0.440	0.402
Flange2	0.4	0.4	0.32	0.32	0.04	0.118	0.181	0.101	0.399	0.080

Total Surface Areas m2	1.398	0.778
-------------------------------	--------------	--------------

Table 1: Surface Area Calculations

Process		Setting	Rough 1	Rough 2	Rough 3	Turning	Requ.
Tolerance mm		3.00	2.00	1.00	0.40	0.25	Tol
Minutes per m2			167	167	167	667	mm
Feature	Area m2	Time in Minutes					
Inner							
Flange 1	0.292	5	49	49	0	0	1.00
Ring 1	0.267		44	44	44	178	0.30
Ring 2	0.440		73	73	73	293	0.30
Flange2	0.399		67	67	0	0	1.00
Outer							
Flange 1	0.055		9	9	9	37	0.30
Ring 1	0.240		40	40	40	160	0.30
Ring 2	0.402		67	67	67	268	0.30
Flange2	0.080		13	13	13	54	0.30

Roughing and Turning time per unit

1967 minutes

1.37 days

Table 2: Rough Machining and Turning Durations

Parent	Part	Att	Diam mm	Depth mm	Rtol um	Dtol um	Ptol um
Chamber	Hole01	Flange1	25.00	30.00	100.00	90.00	51.00
Chamber	Hole02	Flange1	20.00	15.00	80.00	60.00	30.00

Process		Pilot	Drill	Bore	Ream
Minutes per mm2 or mm3		0.6	0.6	9.0	36.0
Hole 01	Surface area mm2			21206	84823
	Volume mm3	14726	8836	8836	
Hole 02	Surface area mm2	942			33929
	Volume mm3	4712	2827	2827	

Drilling, boring and reaming time per unit		45 minutes
		0.03 days
Total machining time	1 unit	1.40 days
	5 units	6.99 days
Verification of simulated machining time		6.99 days

Table 3: Hole Specification and Machining Durations

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

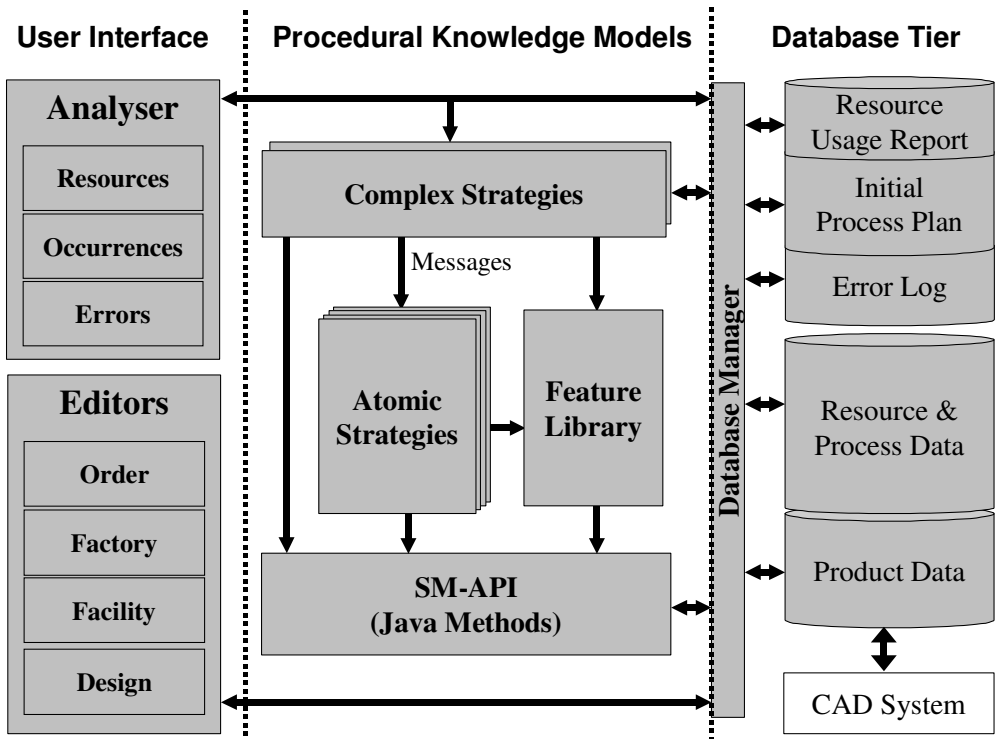


Figure 11: The Manufacturability Analysis Platform (MAP)

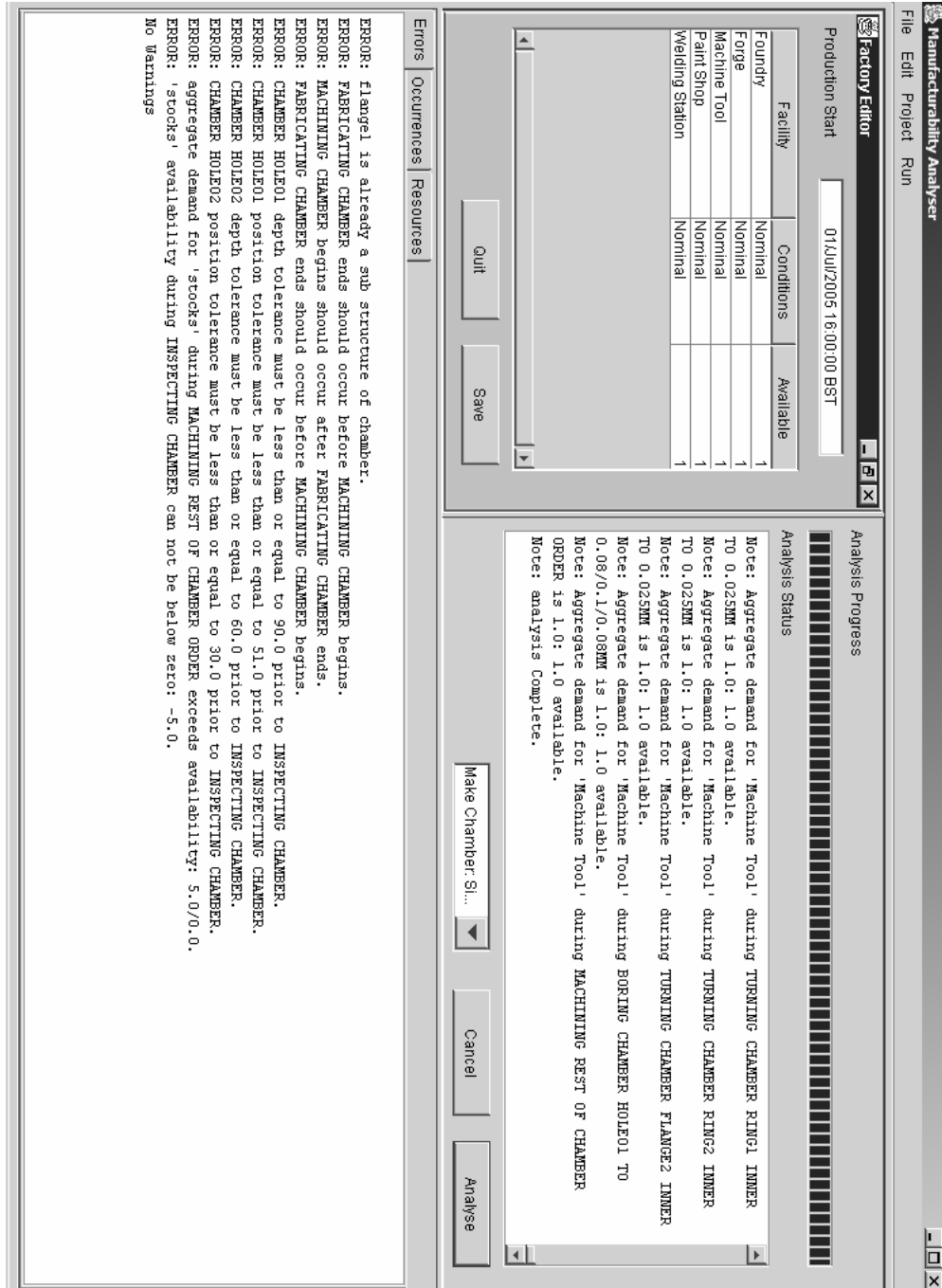


Figure 12: Unexpected Errors in the Machining Strategy

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

Factory Editor

Production Start: 01/Jul/2005 16:00:00 BST

Facility	Conditions	Available
Foundry	Normal	1
Forge	Normal	1
Machine Tool	Normal	1
Paint Shop	Normal	1
Welding Station	Normal	1

Analysis Progress

Analysis Status

Note: Aggregate demand for 'Machine Tool' during BORING CHAMBER HOLE01 TO 0.08/0.1/0.08MM is 1.0; 1.0 available.
 Note: Aggregate demand for 'Machine Tool' during REAMTING CHAMBER HOLE01 TO 0.02/0.02/0.08MM is 1.0; 1.0 available.
 Note: Aggregate demand for 'Machine Tool' during REAMTING CHAMBER HOLE02 TO 0.02/0.02/0.08MM is 1.0; 1.0 available.
 Note: Aggregate demand for 'stocks' during MACHINING REST OF CHAMBER ORDER is 5.0; 64.0 available.
 Note: Aggregate demand for 'Machine Tool' during MACHINING REST OF CHAMBER ORDER is 1.0; 1.0 available.
 Note: analysis complete.

Errors | Occurrences | Resources

ERROR: CHAMBER HOLE01 position tolerance must be less than or equal to 51.0 prior to INSPECTING CHAMBER.
 ERROR: CHAMBER HOLE02 position tolerance must be less than or equal to 30.0 prior to INSPECTING CHAMBER.
 No Warnings

Buttons: Quit, Save, Make Chamber St..., Cancel, Analyse

Figure 13: Verified Machining Calculations