ESDS: Materials Technology Knowledge Bases Supporting Design of Boeing Jetliners.

Mark A. Dahl

The Boeing Company
P.O. Box 24346, MS 73-43
Seattle, WA 98124-0346
dahl@bcstec.ca.boeing.com

Abstract

This paper describes a large rule-based application called Engineering Standards Distribution System (ESDS). ESDS was built and deployed by Boeing and helps design engineers specify materials, manufacturing processes, and parts for commercial jetliners. Its 50 knowledge bases accumulatively cover most jetliner part types and are derived from many knowledge sources. ESDS knowledge acquisition methods are designed to maximize participation of domain experts and employ a domain-specific declarative representation language. Since the initial deployment of ESDS in 1989, ongoing maintenance requirements have led to development of new tools and methods. ESDS represents a unique and effective application of knowledge-based technology to the problem of distributing corporate knowledge assets. It also minimizes the need for subsequent transcription and re-interpretation of its results by providing an unambiguous model of material and processing requirements which feed down-stream computing systems.

Introduction

One of today's safest modes of travel involves machines with over a million parts moving with high speeds through an environment barely reachable from the world's highest mountains. Despite their intricate mixture of mechanical and electronic components, these machines have operated with high reliability for decades. This is undoubtedly an exceptional technical accomplishment. However, the many competing goals defining the successful design and manufacture of commercial jetliners leave few in the industry who are not seeking improvements. To this end, Boeing has built a suite of knowledge bases (KBs) called Engineering Standards Distribution System (ESDS). ESDS provides Boeing aerospace engineers with materials technology expertise relevant to designing commercial jetliners. This paper describes the ESDS application.

Problem Descriptions

The following two sections describe the two basic problems addressed by ESDS.

Problem #1 - Distributing Materials Technology Knowledge Assets.

Many opportunities for improvement in airplane design and manufacture hinge on what is driving the aerospace industry's focus on "concurrent engineering" or "design/build teams": improving the effectiveness of a corporation's knowledge assets. A corporation's knowledge assets comprise the knowledge and experience it could potentially draw upon. In the case of concurrent engineering, benefits are obtained by front-loading the design process with expertise from down-stream users of designs, typically manufacturing organizations.

In general, knowledge assets are spread throughout a corporation and take many forms such as standards, documented processes, past designs, and employee experience. Since the complexity inherent in products of the aerospace industry require decomposition of the problem space and specialization of employee skills, timely access to corporate knowledge assets is often a bottleneck to discovering, evaluating, and adopting process and product improvements.

Materials Technology is the application of materials science to industry and is a vital knowledge asset to any aerospace corporation. Broad design goals for jetliners include optimization of range, payload, safety, development costs, manufacturing costs, and operating costs. The degree to which such goals are satisfied is influenced greatly by the materials and related manufacturing processes adopted. By improving the distribution of underlying materials technology knowledge assets, improvements can be realized across a large range of design and manufacturing activities.

Task Performed by ESDS. Through interactive dialogs called "consultations", ESDS helps Boeing design engineers select materials, processes, and standard parts to meet design criteria such as strength, weight, corrosion, toxicity, and cost. ESDS KBs cover the great majority of part types found in the current family of Boeing jetliners,

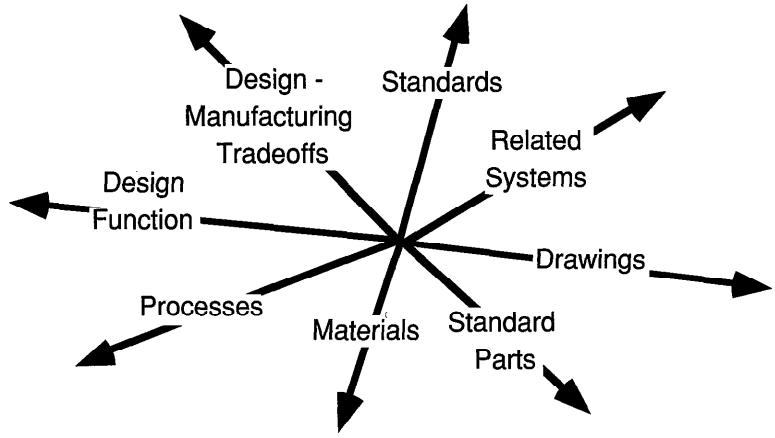

Figure 1. ESDS Knowledge Space

excluding some components not designed by Boeing, like the jet engines.

ESDS is integrated into the Boeing Commercial Airplane Group's design process and is accessible to thousands of engineers throughout the company. It is owned and maintained by Boeing Materials Technology, the major source of Boeing's materials expertise. Therefore, ESDS performs a major role in distributing materials expertise throughout the company.

Problem #2 - Modeling Materials-Related Product Definitions.

Beyond the design itself, another factor influencing manufacturing costs is the availability of detailed and unambiguously defined "product definitions" (i.e., specifications of the jetliner that include the geometry, bill of materials, and manufacturing processes). Product definitions are more beneficial when they can be analyzed and processed by computers. Prerequisite to such benefits is the selection, development and maintenance of formal models.

Substantial progress has been made with modeling the geometric content of drawings. Boeing, for instance, is using what has been called the largest cluster of mainframe computers in the world to support its totally digital design and specification of the geometry of the new Boeing 777. Computer-based geometric modeling reduces the chance of initial designs with ill-fitting parts. It also reduces the need for physical mockups and provides computer models for use in analysis and subsequent manufacturing.

Unfortunately, many other areas of jetliner product definition are still represented using unstructured English. Some areas, like the definition of materials and processes, are semantically very rich, diverse, and challenging to model. The emerging ISO Standard for the Exchange of Product Data (STEP) reflects an interest in such modeling in many industries, but STEP also reflects the preliminary nature of such ventures. An overview of using STEP to model materials can be found in Rumble and Carpenter (1992).

Task Performed by ESDS. A completed consultation with an ESDS knowledge base provides a set of attributes that unambiguously describes the engineering requirements for materials, processes and standard parts. This description is integrated into the product definition (e.g., bill of material systems) for the aircraft being designed.

ESDS Knowledge Space

As shown in figure 1, ESDS spans a very large "knowledge space" requiring a diverse set of experts to maintain. The knowledge space has multiple dimensions spanning materials science, fabrication methods, engineering drawing conventions, documented standards, and external computing systems to which ESDS communicates (e.g. Bill of Material systems). ESDS provides comprehensive coverage of materials (e.g. metals, composites, plastics), processes (e.g. forming, heat treating, injection molding), and standard parts (e.g. fasteners, bearings, hoses, relays). Much of the knowledge comes from multiple overlapping sources such as standards defined at international, military, corporate, and project levels. From the Boeing corporate standards alone, there are over 4,000 active documents contributing to knowledge encoded in ESDS KBs.

Innovations.

The innovative aspects of ESDS include:

Application uniqueness. As far as we know, ESDS is the only large-scale application of its kind deployed in the aerospace industry to address either of the problem areas just discussed (i.e. the problem of distributing materials technology knowledge, or the problem of modeling materials-related aspects of a product definition).

Knowledge Acquisition. ESDS acquisition methods and domain specific declarative languages allow experts to directly code and inspect large portions of KBs with help from knowledge engineers. These knowledge engineers play more the role of a consultant than a controller of every detail of the acquisition process (see Application Development & Deployment section for more information of the relative roles of domain experts and knowledge engineers).

Maintenance Process. Tools and processes for long-term maintenance have been developed to support the large size, diversity, and expected maintenance needs of ESDS KBs (see Maintenance section).

Other Attempted Solutions.

Early attempts at Boeing to encode materials technology knowledge into interactive programs were short-lived but enlightening.

A few initial programs were built by a determined domain expert who clearly saw the potential of systems with functionality like that of ESDS. After much effort several small programs were built that worked on predetermined test cases. The programs were extremely hard to understand and not practical to maintain. Moreover, it was easy to find omissions, dead ends, and other problems due to the complexity of intermingling control and domain knowledge. These programs were never deployed.

These programs were later re-written by a programmer. While they gained limited use, they suffered from the same basic problems. The approach of embedding decision-making knowledge in procedural code was not proving practical. The expert was left searching for something that would scale-up.

This same domain expert was then introduced to what would become the ESDS knowledge-based approach. A critical feature of this approach is the knowledge acquisition technique (covered later) in which experts are directly involved in much of the knowledge encoding (under the guidance of a knowledge engineer). In addition to the sense of ownership caused by giving the expert hands-on access, this approach also promised systems easier to validate and maintain. The domain expert became the ESDS project "champion", obtaining funds, and commitments to involve many other experts in building the system.

Besides ESDS, several other small systems using knowledge-based approaches were developed. Despite the talented knowledge engineers working on them, these systems, never obtained widespread use for several reasons. The KBs were not well integrated with other systems. The KBs were not "owned" by organizations that could authenticate, maintain, and integrate them into the overall business process. Finally, the KB development and maintenance processes were not shown to adequately support such activities.

Application Description

Application History & Status

In 1985, a prototype system was built to demonstrate proposed functionality to potential user organizations. The ESDS project was initiated in late 1986 and KBs have been incrementally released into full production use since 1989. Since then, the ESDS project has had dual roles of KB development and maintenance. Currently, the KB suite contains tens of thousands of rules distributed among 50 KBs. The ESDS application has a built-in need for ongoing maintenance (see Maintenance section).

AI Technology Used

ESDS is primarily a rule-based system. ESDS KBs are comprised of backward chaining rules reasoning over attribute-value pairs. Chaining is initiated by an agenda mechanism that includes conditional control constructs.

ESDS is not a typical rule-based system. As detailed in knowledge acquisition section, the KBs are written in a domain specific knowledge acquisition language and then translated to a commercially available expert system shell for execution. The language has several data types and other features not found in typical rule-based shells. The declarative semantics of the knowledge acquisition language are more akin to a logic-based system like Prolog than a production system like OPS 5.

A rule-based approach using declarative semantics was chosen for ESDS after surveying knowledge already contained in documents (e.g., design manuals) and after creating a few KB prototypes. The choice was made because the knowledge observed was:

- mostly symbolic,
- usually specified declaratively (e.g., sometimes with complex tables using footnotes to specify conditions),
- shallow, giving recommendations based on conditions without reference to the underlying causal models.

Delivery Platform

ESDS was first deployed on IBM 3090 mainframes under MVS/TSO. A migration to the MVS/IMS-DC transaction-based environment is under way (the migration will not affect KB source code or the "look and feel" of ESDS). In addition to its KBs, ESDS contains many other modules supporting consultation management, online reference information, interfaces to databases, and external applications.

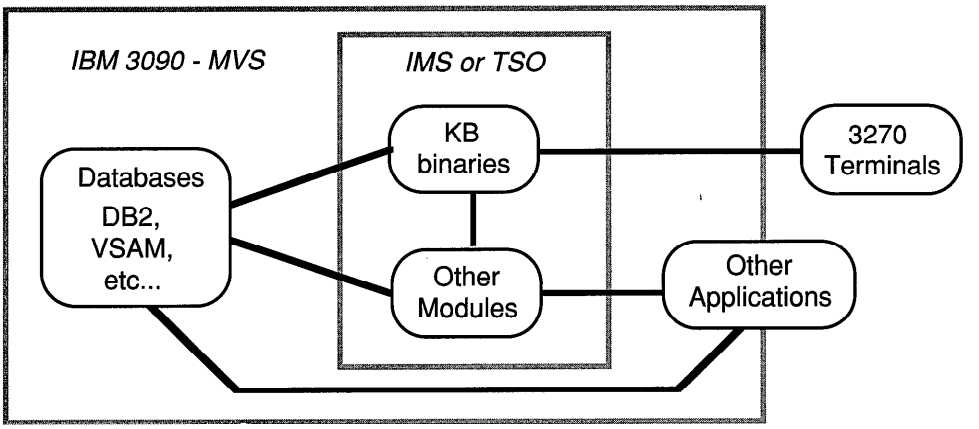

Figure 2. ESDS Production System Architecture

Figure 2 provides a high level overview of the ESDS system architecture.

The KBs are deployed using the ADS expert system shell by Trinzic (originally sold by AION Corp.). The KBs are delivered in compiled form, as standard MVS load modules. The other modules are written in C, FORTRAN, and ADS. ADS was used for some non-KB modules since it supports procedural and object-oriented programming, and is well integrated with MVS.

IBM's relational database management system, DB2, is accessed from the KBs directly, or by calling C or FORTRAN routines containing embedded SQL. Most interfaces to other applications are through databases or files. One interface uses TCP/IP sockets to communicate with a server application running on Unix platforms.

Expert System Shell Selection Criteria

The ADS expert system shell was originally selected for ESDS in 1986. The criteria used at that time included the standard list of shell features for backward-chaining rule bases. Some features not required were certainty factors and explanation facilities for end-users.

Criteria not common to many shells of that time included support of: MVS/TSO, DB2, standard MVS file access methods (VSAM, QSAM, and PDS), interface modules written in other languages, and automation of user interface "screens" for 3270 terminals. Since ESDS needed to run on production mainframes along with other applications, the shell also had to be robust and resource conscious.

Another important criterion was the ability to quickly deploy prototypes. Most of the original development team had no prior MVS experience, so the ability to build prototypes using the Microsoft DOS version of ADS helped meet this criterion.

The ability to automatically create user interface "screens" for 3270 terminals turned out to be more valuable than originally assumed. TSO and IMS-DC developers often estimate application development costs by counting the screens which must be built. In contrast, all the thousands of ESDS KB screens were automatically generated by the shell.

The shell's rule interpreter was used for the first few years of development. As KBs grew in size, it became clear that speed and resource demands of the production environment required the KB compiler. Now, all ESDS KBs are deployed in compiled form.

Application Use and Payoff

ESDS has been in production since 1989. Frequency of use averages several thousand consultations per month.

Several studies have been performed to estimate ESDS cost savings. Figures are based on:

- time saved by domain experts directly supporting users,
- time saved by users working on their own,
- reduction of needless variety across designs (e.g., minimizing the variety of materials, standard parts, tools, and process plans) ,
- reduction of re-design due to initial design errors,
- reduction in wasted materials and manufacturing operations due to design errors.

Estimates consistently show substantial costs savings which clearly reflect an outstanding contribution to Boeing's design/build process.

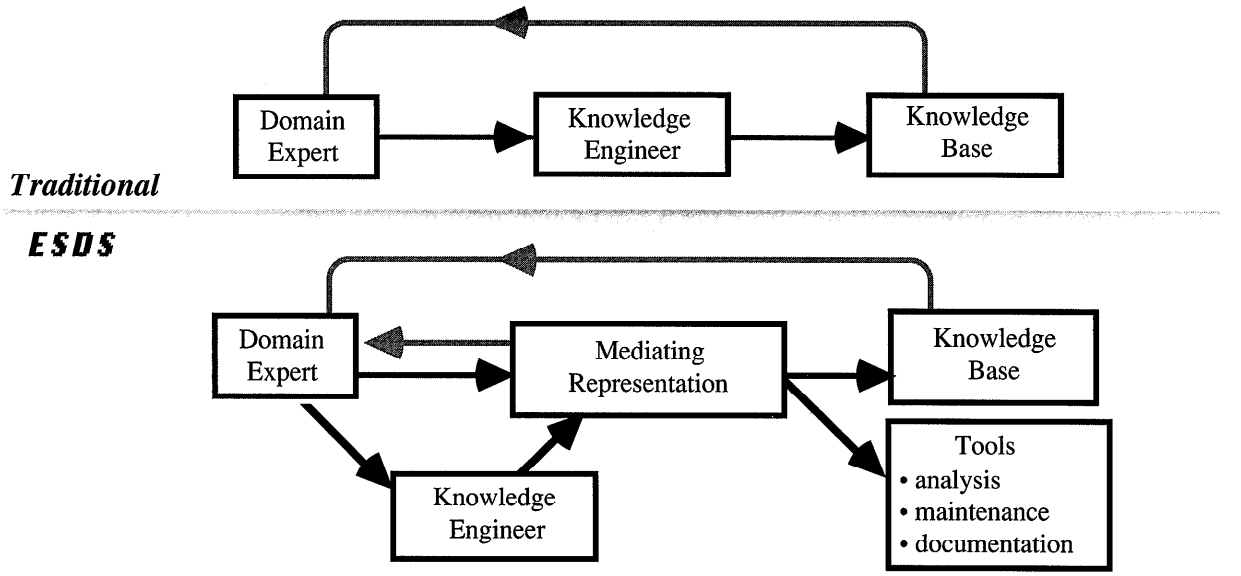

Figure 3. Traditional vs. ESDS Knowledge Acquisition

Application Development & Deployment

Development Context

The major source of domain experts for ESDS come from the Boeing Materials Technology organization (BMT) which is the major source of materials expertise for the Boeing Commercial Airplane Group. To involve BMT's many experts, it was imperative that BMT be the home of the ESDS project. ESDS development is a team effort between BMT and a computing technology group called Scientific Data Systems, who supplied knowledge engineers and developers of the non-KB portions of ESDS.

The number of ESDS knowledge engineers has ranged from 3 to 8. The ratio of domain experts to knowledge engineers has averaged 4 to 1 and has often been higher. While a few domain experts contributed heavily to the project for a year or more, most were called in as needed, spending several months at a time. This meant there has been a constant stream of new domain experts to work with.

Because domain experts have been more abundant than knowledge engineers, a way had to be found to keep the knowledge engineers from becoming the bottleneck to the knowledge acquisition process.

Knowledge Acquisition

The knowledge acquisition method employed by ESDS allows domain experts to encode large amounts of knowledge with minimal supervision by knowledge engineers. This is accomplished by using a "mediating representation" (see Boose 1990 and Marcus 1988) in the form of a declarative knowledge acquisition language called ELL (Engineering Logic Language).

Figure 3 compares the traditional mode of knowledge acquisition with the ESDS approach. Traditionally, expertise is teased out of the expert and mapped into some knowledge representation framework. The ESDS approach provides a mediating representation that is straightforward enough for the domain expert to read, and usually, to write. The ability of experts to directly read ELL provides another important avenue for verification besides running test cases (as shown by gray arrows in figure 3). The ability of experts to write ELL means the knowledge engineer doesn't have to elicit and encode every scrap of knowledge. It also increases the expert's sense of KB ownership (in fact, the title used for ESDS domain experts is "logic author").

Many knowledge acquisition techniques (e.g., repertory grids, Boose 1990) focus on eliciting expertise that is not easily expressed because it has been "compiled" into automatically triggered problem-solving behaviors. In contrast, while ELL is a convenient language in which to express most ESDS knowledge, it doesn't have any elicitation mechanisms.

Using ELL still provides a productivity gain in ESDS since a large bulk of the knowledge to be encoded is not "compiled". Rather, a lot of the knowledge comes from written documents such as design manuals and domain experts who author such standards. In fact, nothing approaching the scope of ESDS would have been feasible without the large investment Boeing initially made in codifying design knowledge into documents.

Flexibility regarding experts' availability is supported by the ESDS approach. The amount of ELL written by experts vs. knowledge engineers varies. Some experts have only enough time to be periodically interviewed.

```
$ Product_Form FOUNDBY ASKING USING LIST Valid_Product_Forms
               WITH PROMPT "Which form is the part made from?"
               WITH HELP "The forms are defined as follows..."

$ TABLE Material_Table
   (Basic_Material,  Product_Form,  Temperature_Range) =
   ("alloy 1",       "sheet",       R:(0,   350]     )
   ("alloy 2",       "sheet",       R:(350, ~)       )
   ("alloy 3",       "forging",     R:(0,   230]     )
   etc., etc., etc.,...

$ Basic_Material (rule_1) =
  "alloy 11"  IF Corrosive_Area AND Product_Form = "sheet",
  "alloy 12"  WITH WARNING "This alloy is costly..."
              IF Structural_Part AND Product_Form = "forging",
  LOOKUP IN Material_Table USING Product_Form
  WHERE Temperature INRANGE COLUMN_NAME(Temperature_Range)
              OTHERWISE
```

Figure 4. Example ELL Statements

Even in these cases, the expert often can read the ELL, still providing an additional avenue for verification.

Even when a domain expert writes most of the ELL, the knowledge engineer defines the overall structure of the KB, oversees the domain expert's progress, and is heavily involved in verification and deployment. ESDS knowledge engineers also work as a team to hold KB inspections (similar to Fagan 1986) and to refine the KB development process. A major focus of the team is to insure the KBs are not only correct but maintainable.

Key ELL features supporting knowledge acquisition are:

- The language does not require or support procedural constructs besides the simple agenda mechanism. There are no branching or looping constructs. Writing ELL does not involve specifying a computational process (the major hurdle for non-programmers, see Marques, 1992 for further discussion).
- Attribute values can not change over time. This is even true for set- and list-typed attributes since, while they grow monotonically over time, any access to them is deferred until they can no longer grow. ELL is a declarative language which is referentially transparent, meaning that ELL authors can reference the value of an attribute without regard to how, or when it was computed.
- Rules perform single functions (usually assigning a value to an attribute). Multiple rule functions can be harder to understand since it's not always clear which rule conditions are strictly required for a particular function (see Soloway, 1987 for more on this topic)
- The ELL language contains several domain- and task-specific abstractions. For instance, there are extensive facilities to dynamically populate and query tables with structured cell types similar to tables found in aircraft design manuals.
- There is support for KB modularization.
- The language is extensible and has evolved over the entire development of ESDS.

The success of ELL has since spurred the creation of two other declarative domain specific languages based on the same principles but developed to handle auxiliary KB tasks not covered in this paper.

ELL Language Example

There are many features in the ELL language. This section describes a few using a short example. Figure 4 shows three ELL statements, each separated by a "$". Language keywords are in bold.

The first statement is an attribute definition. It states the value of the attribute named Product_Form is found by asking the user to choose from a list-valued attribute named Valid_Product_Forms. This syntax for querying users can also be used in rule actions.

The second statement defines an ELL static table with three columns: Basic_Material, Product_Form, and Temperature_Range. The third column has a special data type representing mathematical ranges (the square and open braces signify closed and open boundaries, respectively, the "~" stands for infinity). Other column types provided include lists of values. A single table may be used in multiple queries.

Use of tables over rules is encouraged whenever possible, since when tables are coupled with a lookup statement, they serve as a concise and easy to verify set of rules, which all have the same form.

The third statement is an ELL rule illustrating how default behavior can be specified. The rule has 3 condition-action pairs called rule clauses. Note each clause is written with the action first before the condition (to promote goal-directed thinking). The first two clauses represent special cases that cannot be handled by the table lookup. The last clause, using the keyword OTHERWISE in place of a rule condition, is the default action to be taken if all exception cases fail.

Rules can be lumped into Rule Groups which define a common set of conditions that must hold before any rule in the Rule Group can fire. Rule Groups provide a way of defining contexts, each which may house independent sets of defaults.

Inferencing is goal-directed and is initiated with "goal statements". For instance, the ELL statement "$ DETERMINE Basic_Material" would initiate backward chaining on the rule in figure 4.

Rather than letting ELL evolve into a general-purpose knowledge representation language, the introduction of language features was guided by the goal of making it easy for domain experts to understand the language. Because ELL has been kept simple enough for domain experts to understand, reading it serves as a form of KB verification.

Verification & Validation

We use "verification" to describe the process of testing whether a KB meets the intent of the development team (knowledge engineers and domain experts). In contrast, "validation" describes the process of testing whether the resulting KB system adequately meets the needs of its users.

Verification Strategies. Verification strategies include run-time testing by peers of domain experts, and comparisons against manually-generated cases (e.g. previous aircraft design drawings). Some testing is undirected ("try to break the system") and some is directed by test plans constructed for each KB.

Another strategy employs a "random test generator" which replaces the user interface and randomly provides input to the KB (randomness can be governed by developer specified weights). It is used to: achieve comprehensive test coverage, find holes in the rules (goals that can't be satisfied), detect circular reasoning, and find dead/unreachable code (or conversely under-qualified rules) by tracking the frequency in which rule clauses are fired over many KB consultations.

An extensive set of static analysis tools was recently developed to support KB maintenance (see Maintenance section). These tools detect logical inconsistencies and redundancies. They also find several types of completeness problems. The analysis tools are based the KB Reducer algorithm of Ginsberg (1991), and are further described in Dahl and Williamson (1992).

Validation Strategies. The ongoing process of validation relies on beta tests, interviewing and surveying end-users, and searching for trends in change requests. Confidence in the usefulness of ESDS also comes from the fact that the domain experts' organization, BMT, has integrated the KBs into the overall support they provide to end-users.

Deployment

ESDS is deployed on 4 IBM 3090 mainframe clusters and is accessible from most Boeing sites. Application updates are distributed in maintenance releases by another organization, which eliminates many deployment tasks for the development team. However, fitting updates for 50 KBs into a pre-determined set of releases creates its own project management challenges.

Maintenance

The KBs require ongoing maintenance to reflect new Boeing products, advances in materials and processes, and evolving standardization efforts. KBs are maintained by the same materials and computing technology organizations responsible for development.

KB maintenance presents different challenges than KB development, which led us to define a separate maintenance process. One major challenge is that KB maintainers are inevitably not as familiar with KBs as are developers. This is even true when original developers perform maintenance after not being involved with the KB for several months. A related challenge is the re-discovery of knowledge needed for maintenance that was left implicit in the KB. One example of implicit knowledge is an undocumented dependence on the order in which rules fire.

In responding to these challenges we have built tools to perform extensive static analysis, regression testing, to aid KB comprehension, and to locate areas of a KB likely to be affected by a proposed change. For example, a tool that detects logical inconsistencies includes an option that also detects rules with implicit order dependencies. These tools are described in Dahl and Williamson (1992).

Recent experience with static analysis tools have led to insights about detecting logical inconsistencies and redundancies. We've cataloged many common causes that underlie these logical errors and which sometimes complicate error reporting. We've also found that using these tools can uncover potential maintenance traps that are usually not found by running the KB. An account of our experiences can be found in Dahl and Williamson, 1993.

Conclusions

The ESDS project has successfully used AI technology to improve the distribution of materials expertise to Boeing design engineers. The project also has contributed to enriching the representation of materials-related information in aircraft product definitions.

The size of the project led to the development of innovative knowledge acquisition methods. Since it appears ESDS will be used for many years to come, we have focused on optimizing the cost and safety of the maintenance process.

References

Boose, J. 1990. Knowledge Acquisition Tools, Methods, and Mediating Representations, In Proceedings of the First Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop (IJAW '90), 25-64. Ohmsha Ltd.: Tokyo.

Dahl, M., and Williamson, K. 1992. A Verification Strategy for Long-Term Maintenance of Large Rule-Based Systems. In Proceedings of the AAAI-92 Workshop on Verification and Validation of Expert Systems.

Dahl, M., and Williamson, K. 1993. Experiences of using Verification Tools for Maintenance of Rule-Based Systems. In Proceedings of the AAAI-93 Workshop on Verification and Validation of Expert Systems.

Fagan, M. E. 1986. Advances in Software Inspections. *IEEE Transactions on Software Engineering*, SE-12(7): 744.

Ginsberg, A. 1991. Knowledge-Base Reduction: A New Approach to Checking Knowledge Bases For Inconsistency & Redundancy. In *Validating and Verifying Knowledge-Based Systems*. IEEE Computer Science Press.

Marcus, S. 1988. A knowledge acquisition tool for Propose-and-Revise Systems. In *Automating Knowledge Acquisition for Expert Systems.* , edited by Sandra Marcus. Boston, Mass: Kluwer Academic.

Marques, D. et al., 1992. Easy Programming. *IEEE Expert* 7(3):16-29.

Rumble, J. and Carpenter, J 1992. Materials 'STEP' into the Future. *Aviation Week & Space Technology*, October, 1992, 23-27.

Soloway, E., et al., 1987. Assessing the maintainability of XCON-in-RIME: Coping with the problems of a very large rule-base. In *Proceedings of AAAI-87*, July, 1987.