SMART

AN EXPERT SYSTEM IN PRINTED CIRCUIT BOARD MANUFACTURING

Daniel W. Rasmus
Western Digital

Jo Coffey,
P. Hariharasubramanian
Peter Hosé
Technology Applications

Presented at IPC '90/ESD

SMART
AN EXPERT SYSTEM IN PRINTED CIRCUIT BOARD MANUFACTURING

In June 1988, Western Digital created a Computer Assisted Manufacturing Department with the charter to pursue the application of advanced manufacturing technologies. Several proposed projects this department was to undertake were expert system projects. In August of 1988, Western Digital decided to proceed with the most extensive plan presented in the department creation document. This project was named SMART for *Surface Mount Assembly Reasoning Tool*.

SMART was chosen as a starting project over other projects (such as diagnosis and repair systems and the implementation of CASE tools) because it was the most tangible to the management team reviewing the new department creation document. SMART was intended to automate the process of creating and balancing pick and place programs for the Sanyo line of SMT pick and place machines.

SMART's goal was to convert an eight hour process that sometimes represented a one week elapsed time, and condense it to a one hour process. Phone calls with Sanyo representatives and other users assured Western Digital that no other tools of this kind we on the market or in development. SMART represented two types of challenges: technical and managerial. Both challenges presented their own unique learning experiences.

Early Obstacles

The SMART project was approved without people so a consulting firm would have to be used. The use of a consulting firm provided important elements to our overall expert system development effort. As a small department we could not afford to hire extensive staff, so with the consulting firm we gain software engineers, knowledge engineers, project managers and chief scientists. Writing a Request For Quotation (RFQ) for consulting when SMART was still ill-defined help us shore up our own thinking and use the expertise of several consulting firm, at no cost, to refine our thinking during the RFQ review cycle.

Some people at Western Digital believed that AI was the wrong choice. They believed that the primary idea behind line balancing would prove to be statistical in nature and AI would only complicate the tasks. But, as we built the RFQ, we found that their were several available heuristics that would not easily translate to C or Pascal. This left us with tool selection as one of the holes in the early drafts of the RFQ. After much research into the PC-based AI market, we settled on Nexpert Object because of its cross-platform compatibility, rich development environment and embedable libraries. Nexpert also allowed us to met three departmental objectives: select a tool that we would not out-grow, avoid LISP-machines and avoid a tool that would require that our consultants to learn the tool on our time.

Preliminary Design Overview

The first schema for SMART called for a phased project plan. The primary milestones were:

- Create component level manufacturing data base with information pertinent to the SMART program,
- Demonstrate a prototype on a known board type and a single pick and place machine,
- Deliver a production prototype that could program any machine in any factory
- Refine the system and install in factories.

Except for start and stops caused by the tightening or loosening of corporate purse strings, the project proceeded punctually and achieved the technical goals set out for each section of the project. As the project progressed, focus changed from the expert system technology, which was proving itself through testing trails, to the in-house difficulties of linking diverse systems.

The data base used early in the project was hand-keyed because no corporate data base existed that would link CAD, Manufacturing and Manufacturing Engineering data. Toward the end of the production prototype phase, Design Automation created a series of programs that would link this data. As we received the information for SMART trials, we realized the difficulty in getting consistent information from systems that were updated in real-time at different stations and at different times. We rarely found a trouble-free data set which worked without change We are currently undertaking the projects that will tie the update of these systems to a consistent time-frame.

The Benefits of SMART

SMART was conceived as a way to eliminate the need for manufacturing engineers to program pick and place machines "long hand." But, as we explored the benefits of such a system, we realized there were benefits that surpassed labor savings alone. Some of these benefits included:

Initial Benefits

- ***Assures optimum program instead of prototype program.***
  In some situations, manufacturing engineers do not have time to write good programs. SMART would allow them to write good programs every time.

- ***Reduces design to production cycle-time for Surface Mount Technology Boards.***
  Since the hand method needs about one week of elpased time to complete an efficient program, SMART will shorten cycle time, and time to market, by this week.• ***Writes programs optimized for a given line configuration.***

- ***Writes programs optimized for a given line configuration.***
  Novice programmers often write inefficient programs that do not run the production line at maximum throughput. These programs

may have inefficient pick and place sequences or unbalanced line situation. The use of SMART should eliminate this condition.

- ***Improves ability to react to customer demands and environmental changes.***
  Because SMART can write programs quickly, using it will allow us to create new programs for board configurations changed resulting from last minute by customer requests.

- ***Improves accuracy of programs by eliminating transcription errors in data.***
  In the long hand method, transcription errors are often encountered. With SMART using "clean data" this program will be eliminated.

- ***Reduces need to hire additional ME staff for programming.***
  The use of SMART should minimize the need to hire new manufacturing engineering staff to support programming.

- ***Reduces information research.***
  SMART will use the integrated manufacturing data base for input, eliminating the need for the manufacturing engineer to research a parts manufacturing information.

- ***Reduces paperwork.***
  SMART does all report generation to support manufacturing engineering programming (load sheets, program sheets, etc.)

- ***Reduces routine efforts.***
  Manufacturing engineers who use SMART can invest more of their time in problem solving endeavours, rather than the mundane programming they have already mastered.

- ***Saves space for research materials and books.***
  Materials used for programming become obsolete with the integrated manufacturing data base. Manufacturing Engineers can use this space for other items.

- ***Demonstrate the ability of expert system technology to solve real world problems.***
  With the success of SMART, Western Digital will see the benefit of expert system technology and make more investments that will generate more savings and revenues.

Long Term Benefits

- ***Provides factory flexibility during a down machine condition.***
  If a machine goes down, SMART could write a program that would work with the machines still running.

- ***Compares alternative courses of action.***
  When a new board comes out, SMART could evalute each line in the factory to see which one would best fits the new board. Alternately, SMART could test the efficiency of new lines by using its knowledge of existing machines placed in new configurations.

- ***Writes routings for use in Western Digital's Central Manufacturing System.***
  Because SMART knows the timing on the pick and place machine, a future enhancement could allow it to determine the remaining sequence for a given board.

- ***Could be used to feed an electronic documentation system.***
  SMART output is completely electronic. If Western Digital chooses to develop a centralized electronic storage system, SMART programs could be stored in it for easy retrieval.

- ***Develop standard cost models for products.***
  If SMART knows timings and routings, cost information could be included for standard cost model development.

SMART Project Tracking

Project management for AI projects is often difficult to visualize. We have included a detailed set of the tasks in the SMART project to provide the reader with a model for his/her own systems develop planning.

Project Tracking Outline

I. Begin Project
   - Pre-project planning
     - Determine the growth strategy for the system
     - Determine administrative support
     - Size AI Consulting Vendor Base
     - Establish Component Database Task Force
     - Develop Request For Quotation
     - Get requisition approved for data entry clerk
     - Hire data entry clerk
     - Final Project Approval

II. RFQ Process
   - Send RFQ to selected vendors
   - Wait for vendor response
   - Select Vendor
   - Place Purchase order and create contracts

III. Database Tasks
   - Define scope of database
   - Define data elements and origins
   - Select database vendor for development

IV. Knowledge Acquisition
   - Interviews (1 week)
   - Write System Specification

V. Implementation (Phase 1)
   - Acquire computer and software
   - Develop prototype of component database

Gather Data base Information
Database entry
Define networking requirements
Prototype Refinement
Deliver System Specification
Software Development (C Coding)
Testing and Validation
Demonstrate prototype of Single machine system
Acceptance Testing, Training and Technology Transfer
Final Documentation

VI. Implementation (Phase 2)

Knowledge Acquisition
- Interviews (1 week)
- Write System Specification

Expand prototype to include multiple machines
Deliver System Specification
Software Development (C Coding)
Testing and Validation
Demonstrate prototype of Single machine system
Acceptance Testing, Training and Technology Transfer
Implement prototype production system for Irvine machine configuration
Implement links from component database to factories
Final Documentation
Implement international links to component database
Implement worldwide (Ireland, Puerto Rico, Korea, Singapore, etc.)
Expand prototype to include post-run editing

VII. Project Audit

TECHNICAL DISCUSSION

Technically, planning the assembly of the SMT components on a board consists of two main tasks:

- assigning to each SMT machine on the assembly line the components that it should place (line balancing)
- determining the placement sequence (the program) for the components on each machine

Each of these tasks was implemented in a separate phase of SMART. During the first phase, we built a system to determine the placement sequence for small boards built by one SMT machine. During the second, we extended this system to include line balancing for assembly lines with multiple SMT machines, and a greater variety of boards.

APPROACH

Before SMART, line balancing and program generation were done by manufacturing engineers, who had developed a set of techniques - heuristics - for accomplishing these tasks. In a series of knowledge acquisition sessions, our team of two knowledge engineers worked to elicit these techniques, with the goal of reproducing them as the expert system. We found direct representation to be possible for some heuristics, but some engineers' decisions made use of extensive information, including visual information, and were of a complexity beyond the scope of our system. For such decisions, we tried to preserve the spirit of their approach using, where appropriate, a mathematical technique to approximate it. One such example is the grouping of components of the same part type. The components of a PCB - transistors, resistors, etc. - come packaged on reels of tape; each reel carries components of a single part type. These reels are loaded on a cassette drive on the machine and the drive has a limited number of reel positions. In addition, the shop may stock a limited number of reels for a given part.

For these reasons, the engineers attempt to minimize the number of reels by sequentially loading components of the same part type. They balance the requirement to minimize the distance between sequential components against the requirement to minimize the number of reels. We did not feel we could directly capture all the information and reasoning involved in decisions of this type, and instead, implemented what seemed to us to be the key process, namely, grouping components of the same part type by location. We used statistical algorithms to identify geometric groupings of components of the same type. The combination of direct representation of expert heuristics, where possible, and approximate representation using, where appropriate, mathematical algorithms, was our general approach in building SMART.

PLACEMENT

In determining the sequence of components, the engineer takes several considerations into account. The main goal is to minimize time needed to place all components, while also satisfying other constraints, specifically, the previously mentioned need to minimize the number of reels. Contributing to the amount of time are:

- the distance between sequentially placed components,
- the speed at which parts of the machine can operate while placing specific components,
- the use of certain machine facilities.

Distance has an obvious relation to placement time; placement time will increase as the distance from the previous component increases. In addition, some systems on the machine operate more slowly for heavier components. Components placed immediately before it will also use the slower speed. Finally, certain machine facilities are mechanically paired, and a certain period of time must elapse after the first facility has been used before its partner is available for use.

These three factors, and the requirement to minimize reels, are often at odds with one another, and the engineer uses different techniques to balance them. We abstracted these techniques into several heuristics:

- Statistically identify components of the same part type to be loaded from one reel
- Group components by placement speed, and use of critical machine facilities
- Identify a path to be followed based on board geometry
- Place each group of components. The criteria used in choosing the next component within a group include the direction of movement, and distance from the previous placed component.

LINE BALANCING

Normally, an SMT assembly line consists of three or four machines, each of which is capable of placing different part types. Sometimes, two machines of the same type may exist on one line, and in any event, normally, many components can be placed by more than one machine on the line. In such a case, the goal of line balancing is to distribute the components so a minimum cycle time can be achieved.

There are essentially two parts to this task:

- Identify the number of components to be placed by each machine
- Identify which specific components will be placed by each machine.

For those components that could be placed by more than one machine on the line, the engineers use mathematical equations to determine the number to place on each machine. These equations incorporate the manufacturer's estimate of time to place components of a specific part type. SMART implements this process with standard linear programming algorithms.

In determining where specific components should be placed, SMART follows the engineers' preference for assigning components to a contiguous area on the board. SMART divides the board such that the computed number of components can be assigned to each machine.

IMPLEMENTATION

SMART is written in NEXPERT/Object and C. It runs under Windows on an AT class machine running DOS. A functional chart of the system appears in Figure 1.

NEXPERT/Object was a good choice for development because of its system integration facilities and its expert system features. NEXPERT's open architecture has enabled us to:

- Build a user interface using the full range of WINDOWS facilities
- Use a commercial Database Management System to produce system reports
- Write computationally intensive processes in C, and access these routines from the knowledge base.

NEXPERT's object oriented facilities provided an intuitive way to represent the problem domain - the board, the machines, the components, etc., and this was valuable during early system development, and even more valuable during system expansion.

We used NEXPERT rules to represent all reasoning that was not computationally intense. We also implemented the flow of control in NEXPERT.

Current Status

The initial project was delivered in April of 1989 and was modified to include line balancing in September of 1989. The most recent delivery includes refinements of the line balancing knowledge and an improved user interface. A future version will include post-editing capabilities for the program data so manufacturing engineers can fix the information when SMART does a less than perfect (or preferred) job. By including the post-editing capability SMART will evolve from an independent black-box to a manufacturing engineering partner.

Along the development path we learned several things about software development, user relations and corporate databases. In the software arena we learned that preliminary designs can be close to reality if they are thoroughly discussed and specified, but even close designs need modification in practice. A flow chart of SMART done early in the process is almost identical to one that would be drawn today, but beneath the major modules, infinitesimal design changes and major interface modifications have taken the program beyond its original description. We found that different board types required different strategies in the system. In the early days and board was a board. We discovered that manufacturing engineers want to interact with the system, not just get output. In the early days we were designing a black box.

SMART has forced Western Digital to examine and redefine it information strategy for manufacturing information. Groups that once worked independently are now working in unison to solve data compatibility problems. SMART has also caused manufacturing engineers to reexamine their requirements and their thought processes. Early speculation leaned toward the tool making programmers less efficient, but we are finding that the system is making them better programmers as they attack faults in SMART.

In the next few months SMART will be implemented at several sites world-wide. We have, in SMART, retained and transferred a high level of knowledge that was once a rare commodity. We have turned a corporate rarity into a corporate commodity.