Validation of an Elevator Maintenance Engineer Scheduling AI System and its Knowledge Refinement

Setsuo Tsuruta
1st Research Dept.
Systems Development Laboratory, Hitachi,Ltd.
1099 Ohzenji, Asao, Kawasaki, 215-0013 Japan
tsuruta@sdl.hitachi.co.japan

Hideaki Ishida
1st Research Dept.
Systems Development Laboratory, Hitachi,Ltd.
1099 Ohzenji, Asao, Kawasaki, 215-0013 Japan
ishida@sdl.hitachi.co.japan

Masaki Honma
Research & Development Center
Hitachi Building Systems Co., Ltd.
29-16-4, Nakagawa, Adachi, Tokyo 120, Japan
m_honma@cm.hbs.co.jp

Akio Nakano
Building System Design Dept.
Hitachi Building Systems Co., Ltd.
3-9-2, Kanda-Ogawa-machi, Chiyoda, Tokyo 101, Japan
a_nakano@cm.hbs.co.jp

Abstract

A considerably complex expert system called elevator maintenance engineer scheduling AI system was developed and has been practically used for more than seven years. However, its knowledge refinement and resultant system validation has been constantly required for its survival as a practically usable system. Their cost is quite expensive compared with the validation/refinement cost needed for ordinary software systems or simple expert systems. In this paper, the history of above refinement and validation is described and the problems of the AI system validation and its knowledge refinement are analyzed. Based on this experience, some ideas to overcome these problems are proposed for efficiently building the right or practically usable AI system.

1. Introduction

Even in an ordinary software system, [Carma, 1992] says that its validation and improvement are becoming essential to its survival as a practically usable system for a long time. How much more is it with a complex expert system?

A considerably complex expert system called an elevator maintenance engineer scheduling AI system was developed and has been practically used for more than seven years. However, its knowledge refinement and resultant system validation have been constantly (every year or every two years) required for its survival as a really usable system. Their cost for such a complex expert system is quite expensive compared with the validation/refinement cost needed for ordinary software systems or simple expert systems.

According to [Gonzalez, 1998], expert systems have traditionally been validated using a suite of test cases whose solution by human experts is previously known. The techniques suggested by [Abel, 1997] provide effective and efficient means of generating good test cases based on the validation criteria specified for the intelligent system. The Turing Test approach proposed by [Knauf, 1998] is a promising way to incorporate the expert's opinion in a methodical fashion.

Their approaches are very usable and promising in theory. Practically, however, many problems were found in validating the above mentioned complex AI systems such as for scheduling elevator maintenance engineers and in refining its knowledge base for gaining its long years' practical use of over 1000 various people in the whole country (Japan). Though [Jantke, 1998; O'Keefe, 1993; Boehm, 1984] say that validation deals with building the right system, what on earth can the right system be defined as? From our validation experience of the above complex AI system, practically though intuitively, it could be at least defined as a system usable for long years.

In this paper, an elevator maintenance engineer scheduling AI system and its complexity is explained first. Secondly, the history of its validation and knowledge refinement is described. Thirdly, the problems of the validation and knowledge refinement of the complex AI system are analyzed. Lastly, some ideas to overcome these problems are proposed based on our experience, for efficiently validating and building the right AI system or an AI system usable for long years.

2. Elevator Maintenance Engineer Scheduling AI System

The rapid increase of the number of elevators and computers causes the deficiency of their maintenance engineers. Under such circumstances, automatic scheduling of elevator maintenance engineers is becoming critical, as one of the most promising ways to improve both customer satisfaction and maintenance engineers' working conditions.

However, personnel scheduling such as above involving personal/social conditions usually causes difficult problems. Namely, it is quite hard to realize a usable system that automatically allocates elevator maintenance engineers.

This scheduling problem was analyzed (1), solved devising a new inference method (2), and its effectiveness was proved (3) as follows:

(1) The system is required to allocate on an average of 5 engineers for about 20 days per month, often as a pair or a team, to approximately 500 different maintenance spots. This means that the system is at least equal, in complexity, to a large scale travelling salesman problem rounding more than 500 cities. Besides, what makes the problem still more difficult, the system

involves more than 100 complex, exceptional and relatively emotional or sentimental conditions concerning human feelings such as customer satisfaction and working conditions.

(2) In order to overcome these difficulties, it was concluded that a technique called "goal intended strategic coordination inference", was necessary. In this technique, knowledge is represented as a multiple level hierarchical tree. This tree includes nodes comprising goals and their strategies to divide, execute, adjust, and integrate goals. A concrete method for this technique was devised and applied to the above scheduling system.

(3) As a result of on-the-spot estimation at several representative maintenance-bases over the whole country, it was proved that the system could shorten the scheduling time previously needed approximately by 50%, including time for manual modification process. Moreover, it could obtain efficient schedules satisfying the complex conditions necessary for practical use. In this way, it was proved that the developed system was useful and the above inference technique was effective in knowledge understanding and refinement.

However, validation and knowledge refinement of such a complex expert system required a great cost and much time, for its long years' survival as a usable system. This problem is historically described in the next section.

In/after the next section, the concept of knowledge refinement involves both knowledge construction such as in the first version (step) and knowledge addition/ modification in/after the second version (step).

3. History of the Refinement and Validation

The history of knowledge refinement and validation of the developed AI system is described.

3.1 First Step of Refinement and Validation

The initial version (version 1), namely the first step of knowledge refinement and validation was as follows.

[Refined knowledge]

Knowledge is separated and enhanced as mentioned below corresponding to independent subgoals.

(1) Appointed-customers allocation knowledge

Some customers called "appointed-customers" specify various kinds of appointments under contract. These appointments include the day, the range of days, what day of the week, time zone, engineers, qualification and sexuality of engineers for maintenance. As a matter of fact, there are various kinds of allocation knowledge for such appointed-customers which are different from that for customers called general-customers having no such appointments. Thus the former was separated from the latter.

(2) General-customers allocation knowledge

Allocation knowledge for general-customers was also modified or added in order to cope with constraints such as for special areas or seasons.

[Validation]

(1) Before field-installation, several experts did validation at headquarters. The test data for validation was the input data of the maintenance schedules at several representative sites and on representative months in the past. The AI system, operated by a knowledge/system engineer, made schedules for all of these sites automatically. Then, experts checked if almost all spots (more than 98%) were allocated and if the resultant schedule satisfied the important constraints. Most of them said that the AI system was "OK" and the system was validated by so called a majority of votes. However, they could not check if the machine-made schedule could be really used, since the schedule was neither for their groups nor for the month when the schedule was about to be used. Yet, though they had their hand-made schedule corresponding to the input data, they did not compare it with the machine-made schedule. The reason was that the machine-made schedule, though it seemed usable, was considerably different from the man-made schedule and it was difficult to compare them one by one.

(2) After field-installation, each of several experts totally at 4 maintenance sites did validation, by generating just necessary schedules. The test data for validation was the input data of the maintenance schedule for their own group of their own site where the schedule was really used and for the next month when the schedule was really used. We call such schedules "just necessary schedules". Each expert made himself a schedule of his own group automatically using the AI system. Then, they checked if they could use the generated schedules. This time, some experts complained and did not use the system, though others used the generated schedules by modifying their unfavorable parts. However, even the latter users stopped using the AI system, once serious problems (as below) happened and it seemingly took time to make the generated schedules be their favorable ones.

[Results]

Previously it took 12 hours to make a monthly schedule by hand, but it took only 4~8 hours using the AI system.

[Problems]

The following problems occurred.

(1) Car-trip customers and overnight customers were allocated mixed with general customers. Here, a car-trip customer is a customer/spot where maintenance engineers visit by car. This car is called a service car that can be used on restricted days since the number of cars is still less than that of maintenance engineers or maintenance groups. An overnight customer or spot is the one where maintenance engineers stay two or three days since the spot is located very far from a maintenance office or site. Therefore, the same kind of customers should be allocated together but separately from other kinds of customers or general customers.

(2) Often, numbers of spots (e.g. more than 10%) were left unallocated. For instance, this occurred in a month with a long holiday, or in a month when engineers, especially competent ones, were busy in their vocational training or in meetings such as for making budget, the next semiannual plans, etc.

(3) Often, vicinity conditions were unsatisfied. Vicinity conditions are the ones that spots or customers maintained by an engineer within a day should be close each other or should be on the way to the farthest one among them. Concretely speaking, vicinity conditions

are represented by a vicinity code which consists of a number of three figures. The first figure represents the district code, the second figure represents the adjacent-block code and the third figure represents the block code. For example, if both the first and the second figures of two spots are equal, they are located in adjoining blocks.

(4) When both interval conditions and vicinity conditions were rigidly satisfied, the workload was unbalanced. That is to say, a schedule of some engineer and/or some day was extremely heavy (e.g. an engineer visiting more than 10 customers a day), compared with that of other days and other engineers. Now, interval conditions are such that maintenance interval should be 11-19 days for customers or spots to be maintained twice a month. This is to say, maintenance engineers must not visit and maintain such a spot or customer at least during 10 days since the elevators or escalators should not be frequently suspended from operating.

3.2 Second Step of Refinement and Validation

The second step (version 2) was as follows.

[Refined knowledge]

(1) Car-trip/overnight/big customer allocation knowledge
In order to solve the problem (1) found in the initial version, Car-trip/overnight customer was allocated before general customers, and allocation knowledge for these customers was separated from that for general customers and enhanced. Besides, big customers having many (e.g. more than 5) elevators maintained together are difficult to be allocated since the number of customers or slots maintained together within a day is limited (e.g. to less than 8). Therefore, big customers are allocated first for optimization, namely for solving the problem (2) found in the initial version. This kind of knowledge is something like so-called "the most constrained value first strategy".

(2) Appointed customer allocation low-level knowledge
Knowledge for satisfying detailed and complicated conditions of appointed customers was added.

(3) Vicinity code learning function
If customers or spots having vicinity codes different from each other were maintained together in the schedule of previous months, they are close each other or on the way to the other from a maintenance base/site. Therefore, their vicinity code or at least their first or second figures could be considered as identical. Thus, the AI system was given the ability to learn vicinity code from really used schedules of previous months.

(4) Workload balancing knowledge
Coping with the problem (4) found in the initial version, this knowledge was added so that an engineer could maintain 6-8 spots or customers everyday.

(5) Pre-allocated spots adjusting knowledge
This knowledge moves already allocated spots to other day or other person to improve maintenance interval, workload balancing, etc.

[Validation]

Each expert at 15 maintenance sites did validation by generating just necessary schedules in the same way as the after-installation case (2) of the first step.

[Results]

(1) The problems with car-trip customers, overnight customers and appointed customers were all fixed.

(2) The number of spots that the AI system could allocate increased by 3~5%.

[Problems]

(1) The vicinity conditions were unsatisfied.

(2) Some spots were not allocated, while for some day a certain engineer was free. Thus workload was often extremely unbalanced.

(3) There were problems with "my-elevator-policy" spots. Though less than 1 % of the whole maintenance sites, some sites have "my-elevator-policy". In these maintenance sites, each maintenance engineer is responsible for maintaining pre-assigned elevators. Since he has pre-assigned customers or spots namely "my-elevator" spots, problems happen if one of his "my-elevator" spots was allocated to other engineers.

(4) It was still difficult for knowledge engineers to edit/understand knowledge.

(5) It was difficult to regain the confidence of disappointed users and to recover bad reputation of the AI system drastically.

3.3 Third Step of Refinement and Validation

The third step (version 3) was as follows.

[Refined knowledge]

Case-based knowledge (balance-first/interval-first strategy, each for ordinary/my-elevator policy) was newly added. A schedule of the same type of month was usually a case though explained further in section 4.1. Balance-first strategy sometimes relaxes maintenance interval conditions. On the contrary, interval-first strategy sometimes relaxes workload-balancing conditions such as the maximum number of spots or customers maintained together in a day by an engineer.

[Validation]

Each expert at 2 maintenance sites in large cities and 3 sites in local cities did validation, by generating just necessary schedules in the same way as the second step. However, what differs from the former steps is that each expert in large cities did validation 3 times each for different months when the schedule for the month was just necessary (namely at the end of the previous month).

[Results]

(1) Using balance-first strategy, most spots were allocated.

(2) Knowledge editor made refinement operation easier.

[Problems]

(1) Vicinity codes were ill maintained. In addition, spots with time-consuming inspection tasks, being allocated together with other spots, made workload ill balanced. Constraint relaxation per group/month seemed necessary for real use in special months or particular groups.

(2) Refinement of low level knowledge was expensive.

(3) For a maintenance site adopting "my-elevator-policy", it seemed very expensive to refine knowledge, since individual working conditions change drastically every month.

3.4 Fourth Step of Refinement and Validation

The fourth step (version 4) was as follows.

[Refined knowledge]

(1) Refined case-based knowledge (balance-first/interval-first strategy) was added. This knowledge greatly changes the retrieved case adaptively. This knowledge is explained further in section 4.1.

(2) Knowledge editor refined low-level rules specified in slots of strategy frames.

[Validation]

Each expert of each representative group at 2 maintenance sites in 2 very large cities did validation, before installation, by generating just necessary schedules in the same way as the large cities' case of the third step. What differs from the third step or the former steps is that the validation was done totally 12 times for 2 years each in different months (e.g. every 2 months), with severely checked and corrected inputs and parameters such as vicinity codes. We considered that the AI system could be validated at least for the groups of the above sites if, for a long time (e.g. all of the above 12 times), the experts really used the schedules generated by the AI system, permitting manual modification for less than an hour. If the system was validated for several representatives of a kind of groups or sites such as those in very large cities, we considered that the AI system was validated for that type of sites, since knowledge is not general in the current AI technology. Knowledge was also refined until the end of the fourth validation. In the 5-12th validation, experts of other groups at the same maintenance sites also did validation, by making their groups' next month schedules in the same way as above.

[Results]

(1) High quality schedules were automatically generated. Most spots were allocated even under hard constraint, using situational constraint relaxation knowledge.

(2) It became possible to add and modify, on the spot at each maintenance site, high level knowledge such as strategies concerning allocation.

(3) Scheduling time was reduced to 2.7 hours per month.

[Problems]

(1)People sticking to details or disliking the adjustment of vicinity codes did not use automatic scheduling.

(2)The usage became complex.

4. Analysis of Refinement and Validation

4.1 Analysis of Refinement Process

Knowledge refinement is a very expensive work, since it includes knowledge acquisition, representation, incorporation, and resultant system validation. Therefore, knowledge refinement effective only for maintenance bases minor in size or number (e.g. maintenance bases adopting "my-elevator-policy") was instantly discontinued when it seemed to require a lot of knowledge to be added or modified.

Case-based approach was helpful to refine knowledge of our AI system. This seemed because there were a lot of special conditions or knowledge differing from each other among over 1000 various maintenance groups or among their schedulers at hundreds of different maintenance bases. Yet, these conditions or knowledge were too difficult or too numerous to represent as symbolic logic (rules) or expressions.

However, our naive case-based approach had a problem. To make a schedule for an ordinary month, the previous month's schedule was retrieved as a case. For a schedule of a particular month such as with a long holiday, a similar holiday patterned month's schedule was retrieved as a case. Indeed, this was successful when various conditions were almost equal with each other among the same types of months, but it seldom happened in the real world. Mostly, they were quite different every month every year, since there were perpetual changes in off-maintenance working hours such as education (e.g. 2 weeks' education for Mr. So-and-so), meeting, or in organizations such as group members, or in contracts with customers. This seemed to require infinite number of cases.

In order to fix these problems, a mixed approach was devised. The number of cases was restricted, and, instead, the retrieved case was greatly modified for adapting itself to drastically changing conditions. Thus, this approach called "refined case-based strategy" changes the retrieved case by smaller units, namely more elaborately, than "case-based strategy" does. Experts mostly accepted the schedule made by "refined case-based strategy".

4.2 Analysis of Validation Process

The validation method had been improved as the version number increased. At the first version (step), multiple experts mainly in the system development division of headquarters did validation, using general data that are different from those for making a just necessary schedule, namely not for their own group and not for the next month. All experts accepted the system. It seems because they suffer no direct loss from easily accepting automatically generated schedules, or they could not imagine how many concrete or exceptional and serious matters such as complaint of particular customers for maintenance interval or complaint of maintenance engineers for ill-balanced workload would be involved. Therefore, many of the experts at their own maintenance bases never or infrequently used the schedule made by the first version.

On the contrary, at the fourth version (step), each responsible expert from several respective groups at 2 bases pursued validation for more than two years, over 5 times a year, by making a schedule just immediately necessary for his own group. Data for the next month's schedule were used with each validation. In this validation approach, they could have feedback such as complaints or pressure from their group members or customers when they put their generated schedule in practice. Therefore, once the system was validated, it has been really used at maintenance bases, though not everywhere but mainly in very large cities such as Tokyo.

Our validation of case-based knowledge seems different from that of [Knauf and Gonzalez, 1998], since, in our validation, a retrieved case is greatly modified by rules or procedural knowledge and therefore a case itself can not be a test case. However, this (ours) seems to happen frequently in the real world. Especially, this might be true for non-linear or scheduling problems where domino effects easily occur by a slight change in a schedule.

5. Proposals based on Practical Experiences

Based on the above analysis of our practical experience, the following ideas could be proposed to knowledge-refinement or validation of complex AI systems, especially for the use of numerous different people such as over 1000 personnel schedulers in various areas of the whole country.

5.1 Proposals for Efficient Knowledge-refinement

As for efficient knowledge-refinement of such complex AI systems, the following could be pointed out.

(1) Knowledge should be refined repetitively (e.g. every 1~2 years) for long years (e.g. for 5 years), selecting a few experts and sites. Well-balanced experts (e.g. not too minute in local matters) willing to use the system should be selected as AI system valuators as well as knowledge providers. Sites or groups having many particular conditions should be avoided.

(2) Refinement of expensive but rather not general knowledge should be discontinued as soon as possible.

(3) Refinement incorporating case-based knowledge seems very effective for AI systems used by various users in various areas with a lot of special conditions/knowledge that are hard to represent as symbolic logic or expressions.

5.2 Proposals for Efficient Validation

As mentioned in the introduction, validation deals with building the right system, which could be defined as a system usable for long years. From this viewpoint, based on our experience, the followings are proposed, as for efficient validation of complex AI systems.

(1) Validation should ensure that the validated system could be really used for long years.

(2) Accordingly, each selected expert (a prospective real user) from several respective groups, in stead of "anonymous voting of multiple experts" in [Knauf, 1998], should independently continue long-term (yearly as well as monthly) validation. Data reflecting regional and time differences in goals, circumstances, etc. should be used for validation. Furthermore, they should do the validation just at their working spot (e.g. his maintenance site) and when the AI system or its solution (e.g. a schedule) is just necessary. Only they can rightly validate the system, or in other words, can rightly judge if they can safely use the system in their own group for a long time. Other experts can not usually do it since they are still less responsible for the use of the system than the prospective real user.

(3) Since retrieved cases seem to be greatly modified in real problems, each case itself might not be used as a test case. Instead, some AI (especially, scheduling) systems including case based systems could be validated, by checking how long the AI (scheduling) system had been used within permissible manual modifications of the AI solution. Moreover, even when multiple schedulers cooperate to generate one schedule, due to a real world experience in Japan such as [Tsuruta, 1997], validation might be done more effectively through checking if their leader accepts a machine-made schedule than through the "anonymous voting" Turing Test approach proposed by [Knauf, 1998].

6. Summary and Conclusion

An elevator maintenance engineer scheduling AI system classified as a complex expert system was developed and has been practically used for over seven years, with a great cost needed for its perpetual knowledge refinement and validation of the resultant system.

The historical steps of its knowledge acquisition/ refinement and its system validation were described and the problems in these steps were analyzed highlighting the cost and the effect of the AI system validation and knowledge refinement. Based on this experience, some ideas to overcome these problems were proposed for efficiently building the right or practically usable AI system.

References

[Abel, 1997] Abel, T. and Gonzalez, A. J., "Enlarging a Second Bottleneck: A Criteria-based Approach to Manage Expert System Validation Based on Test Cases", Proceedings of the 1997 Florida Artificial Intelligence Research Symposium. Daytona Beach, FL, May 1997.

[Boehm, 1984] Boehm B. W., "Verifying and validating software requirements and design specifications." IEEE Trans. Software, vol. 1, 1984, pp. 75-88.

[Carma, 1992] Carma M., "The Three Rs of Software Automation: Re-engineering, Repository, Reusability", Prentice-Hall, 1992.

[Gonzalez, 1998] Gonzalez, A. J., "Validation of Human Behavioral Models", in "Future Trends in Intelligent Systems Validation" Hans-Rainer Beick and Klaus P. Janke (eds.) Meme Media Laboratory Technical Report MEME-MMM-98-5, Hokkaido University, Sapporo, Japan, Sept. 1998, pp. 77-85.

[Jantke, 1998] Jantke, K. P., "Relating Validation Expertise to Domain Expertise", in "Future Trends in Intelligent Systems Validation" Hans-Rainer Beick and Klaus P. Janke (eds.) Meme Media Laboratory Technical Report MEME-MMM-98-5, Hokkaido University, Sapporo, Japan, Sept. 1998, pp. 64-68.

[Knauf, 1998] Knauf, R., Jantke, K. P., Gonzalez, A. J., and Philippow, I., "Fundamental Considerations for Competence Assessment for Validation", Proceedings of the 11th International Florida Artificial Intelligence Research Society Conference, Sanibel Island, FL, May 1998, pp. 457-461.

[Knauf and Gonzalez, 1998] Knauf, R. and Gonzalez, A. J., "Validation of Rule-based and Case-based Systems: What's common and What should be Different", in "Future Trends in Intelligent Systems Validation" Hans-Rainer Beick and Klaus P. Janke (eds.) Meme Media Laboratory Technical Report MEME-MMM-98-5, Hokkaido University, Sapporo, Japan, Sept. 1998, pp. 25-40.

[O'Keefe, 1993] O'Keefe R. M., and O'Leary D.E., "Expert system Verification and validation: A survey and tutorial." Artificial Intelligence Review, vol. 7, 1993, pp. 3-42.

[Tsuruta, 1997] Tsuruta, S., Eguchi, T., Yanai, S., Ooshima, T., "Dynamic Goal centered Coordination AI system: Application and the validation problem." Proceedings of the 42nd International Scientific Colloquium, vol. 2, pp.53 ~58, 1997. 9.