Salecker, Dirk; Knauf, Rainer :

Validation Manager: A Tool for AI Systems Evaluation by A Turing Test-Like Methodology

Zuerst erschienen in:
Proceedings of the twelfth international Forida AI Research Society conference : Orlando, Florida, 3 - 5 May 1999 (FLAIRS 1999) / ed. by Amruth N. Kumar ... - Menlo Park, Calif. : AAAI Press, 1999, S. 531-535
Referenz-Link AAAI:
http://www.aaai.org/Library/Conferences/FLAIRS/FLAIRS-1999/Abstracts/flairs99-095.html

Validation Manager
A Tool for AI Systems' Evaluation by A Turing Test–Like Methodology

Dirk Salecker and **Rainer Knauf**
Technical University of Ilmenau
Faculty of Computer Science and Automation
PO Box 10 05 65, 98684 Ilmenau, Germany
Dirk.Salecker@In.Stud.TU-Ilmenau.de

Abstract

The task of the Validation Manager is the support of the validation of software systems by using a Turing Test–like methodology. The basic idea of this algorithm is published in (Jantke, Knauf and Stephan 1997). The author described the considerations to develop a first program version in (Salecker 98). In the following months the prototype was implemented. This article describes the Validation Manager and shows the functions which are implemented so far.

Motivation

Software systems are used in all spheres of industry and commerce. While controlling industrial plants errors in the applications could result in expensive downtimes or danger for people and environment. So, the validation of software systems can be an important aspect for the acceptance of a software system. With a validity seal the user could be sure whether or not a program satisfies his expectation.

But modern software systems are very complex and mostly designed by a team, not by a single person. Simple validation algorithms can not show all aspects of the system validity. That is a reason why the validation of these software systems is at least as complex as the systems themselves and very costly. The Validation Manager supports a test case based evaluation algorithm for AI Systems.

The Validation Methodology

The validation process is divided into four steps, as can be seen in figure 1. The *Test Case Generation Stage* and *Test Case Selection Stage* generate the test case set for the evaluation by a Test Case Generator. This set contains a selection of possible cases and is reduced by some criteria, so that only the test cases with a large influence on the system behavior are in it. The algorithms of these stages are described in (Abel, Knauf and Gonzalez 1996) and (Abel and Gonzalez 1997).

After that the evaluation starts with the Validation Manager. In the *Test Case Evaluation Stage* some experts and the software system solve all test cases of the test case set. All solutions are mixed and the experts assess the solutions not knowing whose solution they just assess. An assessment consists of an agreement and a certainty statement. In the *Evaluation Analysis Stage* these values are taken to generate a competence statement for every expert and a validity statement for the software system.

The *Test Case Evaluation Stage* and the *Evaluation Analysis Stage* are the stages of the Turing Test–like algorithm. The reader can find more information about the algorithm in (Jantke, Knauf and Stephan 1997) and (Knauf and Gonzales 1997).

The Validation Manager

The Validation Manager supports the *Test Case Evaluation Stage* and the *Evaluation Analysis Stage*. The *Test Case Generation Stage* and *Test Case Selection Stage* are realized in the Test Case Generator, which is not part of the program.

The tasks of the Validation Manager are

- to consult the experts,
- to manage the test cases, solutions and assessments of the software system and the experts, and
- to get the validity statement of the software system with the Turing Test.

But it is possible that a validation session can take a long time. And it could be complicated to assemble all experts for this time. A program without a solution for this problem is not very practicale and will not be accepted by the involved persons.

The Validation Manager is a solution for this problem. The program's advantage is that the administrator and the experts can work at different places. Therefore the program is divided in two independent tools, the *Validation Host* and the *Validation Client*, as can be seen in figure 2.

The experts use the *Validation Client* to solve the test cases and to assess the solutions. Their results are fed into the Validation Manager to get a validity statement. Thus, the experts can decide when they work at their Validation Client. It is not necessary to assemble all experts for the validation session, they can work at different places.

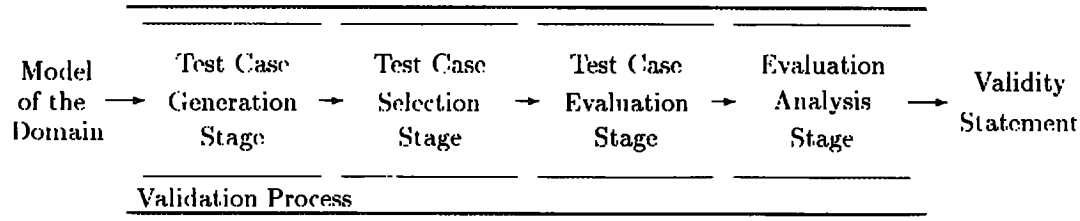

Figure 1: A Test Case Based Algorithm For Evaluation

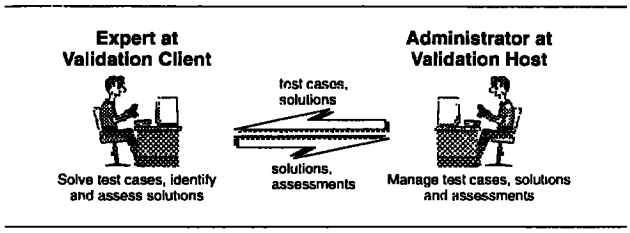

Figure 2: Validation Client and Validation Host

The validation administrator works with the *Validation Host* and manages the test cases, solutions and assessment statements. He receives the collected input data of all experts and calculates the validity statement.

The Program Schedule of the Validation Manager

A complete program session is divided into five parts (see figure 3).

In the *Test Case Import Session* the Validation Host imports the set of test data from the Test Case Generator into its database. The set contains the test cases for the evaluation and the solutions of the software system for these test cases. The set of experts' data with experts' name and address is also fed.

After that the *Test Case Solution Session* follows. Every expert gets the test data set and solves all test cases not knowing the system solutions with the Validation Client. So it is sure that every expert only works with his own knowledge. If an expert does not know a solution, he can give the special answer 'unknown'.

The *Solutions Identification Session* is the next step of the evaluation. Sometimes solutions of one test case are different in syntax, but they mean the same in semantics. Thus, it is possible that the program can not identify a solution of an expert, although the solution is identical with a solution of the software system. Since it is not possible to implement an algorithm to detect several spellings of identical solutions, every expert has to specify his solutions. He finds out which of them are identical to a system solution and unites them. So the Validation Host gets no superfluous solutions, just the necessary new solutions are added into the database.

The *Solutions Assessment Session* is the most expensive and important part of the validation. Now the experts get all solutions and assess them. An assessment consists of an agreement statement and a certainty statement. The agreement statement is an opinion of the solution's correctness, the certainty statement describes how sure the expert is that his agreement statement is correct. However, the solutions are anonymously. Nobody knows whose solutions he just assesses. If the expert feels not competent enough to assess a solution, he can give the special value 'norating'.

With the returned assessments the program calculates the validity statement during the *Validity Computation*. It builds a validity statement for every test case and adds them to the final validity statement.

The Validation Host

The Validation Host realizes the *Test Case Import Session* and the *Validity Computation*. The tasks of the Validation Host are

- to import the test case set,
- to send the test cases to the experts,
- to receive solutions and assessments from the experts and
- to calculate the validity statement.

The host is implemented with Borland Delphi 3.0 and runs on Windows 95/NT. The most important functional units are realized by dynamic link libraries. So it is possible to change single components without effects on other components and fit the program on new functions quickly. Figure 4 shows the most important parts.

The *Kernel* is the central program unit. It controls the schedule of the program and activates the processing units. The *Import Interface* imports the name and the address of the experts from an extern file and the test data and the system solutions of the test case set from the Test Case Generator. The data of the experts and the test data are entered into the Validation Database.

The *Validation Database* stores all needed records for the validation and is created by the program during the validation session. It is implemented in Paradox format. The *Import Interface* fills it with the test data, system solutions and name and address of the experts. Experts' solutions and assessments are saved in the database after the Test Case Solution Session and the Solution Assessment Session respectively.

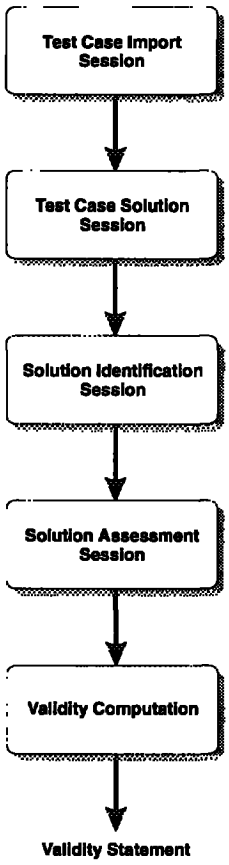

Figure 3: Schedule of a Validation Manager's session

The *Client Interface* exports the test data set for the Test Case Solution Session and the Solution Assessment Session and imports the results of the sessions, the solutions and the assessments of the experts. But the actual version only creates files and saves them on disk. The final version should connect the Validation Host and Validation Client by a network. This feature will be implemented after completing a final version of the Validation Client. Before creating the test case set for the Solution Assessment Session the solutions are mixed to change the sequence.

Another important unit of the Validation Host is the *Validity Meter*. It calculates the validity statement of the software system. First it calculates the competence statement for every expert and every test case. The competence of expert e_i for test case t_j is built with

- $slf\_est(e_i, t_j)$, his own competence's estimation by solving and assessing the test cases,
- $crt\_est(e_i, t_j)$, his confidence while assessing the solutions,
- $cns\_est(e_i, t_j)$, his consistency while assessing his own solution,
- $stb\_est(e_i, t_j)$, his certainty's stability and
- $frgn\_est(e_i, t_j)$, the assessments of his solution by the other experts.

These values are added up to the competence estimation.

The system validity for test case t_j is built with the average of all system solution's assessments weighted with the competences $cpt(e_i, t_j)$ of the n experts.

$$v(t_j) = \frac{1}{\sum_{i=1}^{n}(cpt(e_i,t_j) * s_{j\,sys}(e_i)} * \sum_{i=1}^{n}(cpt(e_i,t_j) * s_{j\,sys}(e_i) * a_{j\,sys}(e_i))$$

with

$s_{j\,sys}(e_i)$, certainty statement of expert e_i and
$a_{j\,sys}(e_i)$, agreement statement of expert e_i for system solution of t_j.

The test case validities of all m test cases are added up to the system validity v:

$$v = \frac{1}{m}\sum_{j=1}^{m} v(t_j).$$

The Validation Client

The Validation Client is the tool for the experts to solve the test cases and to assess the solutions. It realizes the *Test Case Solution Session*, the *Solution Identification Session* and the *Solution Assessment Session*. To simplify the development one version of the Client should be able to run on several operation systems. Thus, it will be implemented through HTML-pages or in Java.

For a quick prototyping the author decided in favour of a Delphi Version running only on Windows 95/NT. The advantage is, that the most class declarations and data structures of the Validation Host can be recycled for the implementation of the Client. That's why, the prototype was finished in a short time.

The connection between Host and Client is not finished yet as well. At this time the test cases for the *Test Case Solution Session* and the *Solution Assessment Session* must be imported manually.

In the *Test Case Solution Session* the experts solve all test cases. If an expert feels not sure enough to solve a test case, he can give the answer 'unknown' by clicking a checkbox.

After that the *Solution Identification Session* follows. The program tries to identify all expert solutions by comparing them with the system solutions. For the unidentified solutions experts must select an identical system solution out of a table. If no solution matches, they can select the value 'none' and the program signed this solution as new. The result is imported by the Validation Host.

The *Solution Assessment Session* starts after Validation Host receives the solved test cases from all experts. The experts open the test case file containing all

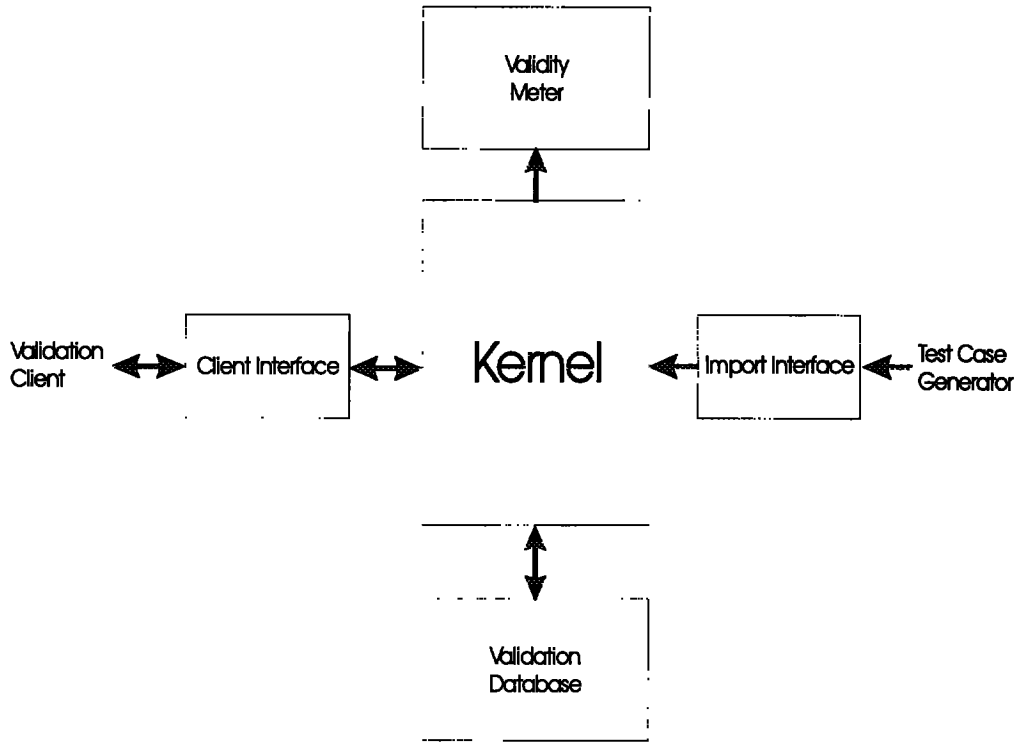

Figure 4: Structure of the Validation Host

test cases, system solutions and expert solutions. The experts give an agreement and a certainty statement for all solutions. The solutions are mixed and nobody knows whose solution he just assesses. This values are the foundation for the calculation of the validity statement.

At this time the Validation Client is just finished in a first prototype. Some details will be changed when the first tests and validation sessions will start.

Conclusions

The Validation Manager realized the test case based evaluation algorithm described in (Jantke, Knauf and Stephan 1997) and (Knauf and Gonzales 1997). The program will be an useful tool for the validation of AI Systems. In next weeks the program will be completed and the functional tests will start. After that the first validation sessions testing the algorithm will follows. Another problem the author must solve is the connection between Validation Client and Validation Host. And maybe the Client will be implemented in Java to run on several operation systems.

References

ABEL, TH.; GONZALEZ; A. 1997.
Utilizing Validity Criteria for Expert System Validation – An Approach to Reduce a Set of Test Cases.
In: (Dankel 1997), pp. 402 – 406.

ABEL, TH.; KNAUF, R.; GONZALEZ, A. 1996.
Generation of a Minimal Set of Test Cases that is Functionally Equivalent to an Exhaustive Set, for Use in Knowledge–Based System Validation.
In: (Stewman 1996), pp. 280 – 284.

DANKEL, D.D. (ED.) 1997.
Proceedings of the Tenth Florida Artificial Intelligence Research Symposium (FLAIRS'97).
Daytona Beach, FL.1997.

GENS, W. (ED.) 1997.
Proceedings of the 42nd International Scientific Colloquium, Ilmenau University of Technology, Vol. II.
TU Ilmenau, 1997.

GRIESER, G.; BEICK, H.-R.; JANTKE, K.P. (EDS.) 1998.
Aspects of Intelligent Systems Validation.
Meme Media Laboratory Hokkaido University, Report MEME-MMM-98-2, 1998.

JANTKE, K.P.; KNAUF, R.; STEPHAN, A. 1997.
Validation von Anwendungssoftware: Von der For-

schung zum Marktfaktor.
In: (Wittig and Grieser 1997), pp. 41 – 61.

KNAUF, R.; GONZALEZ, A.J. 1997.
Estimating an AI System's Validity by TURING Test.
In: (Gens 1997), pp. 65 – 70.

SALECKER, D. 1998.
Validation Manager - An Assistant for a Test Case Based Evaluation of AI Systems.
In: (Grieser, Beick and Jantke 1998), pp. 71 – 77.

STEWMAN, J.H. (ED.) 1996.
Proceedings of the Ninth Florida Artificial Intelligence Research Symposium (FLAIRS'96).
Key West, FL., 1996.

WITTIG, W.S.; GRIESER, G. (ED.) 1997.
LIT'97 · 5. Leipziger Informatik–Tage an der HTWK Leipzig 25.–26. September 1997 – Tagungsbericht.
FIT Leipzig, Forschungsinstitut für Informations-Technologien Leipzig e.V., 1997.