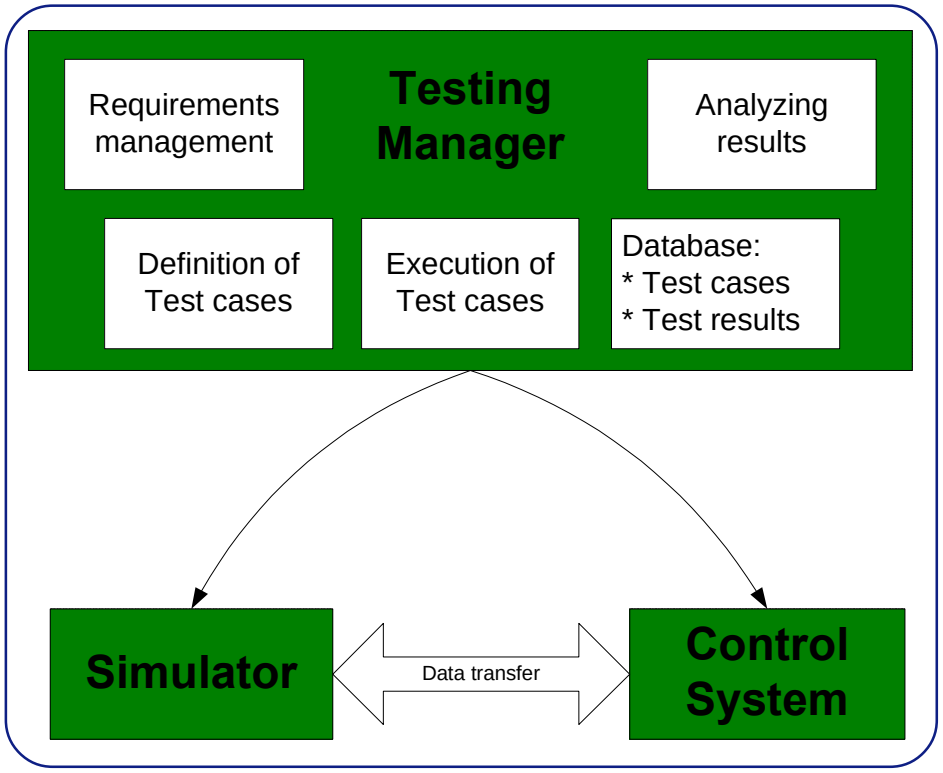

Pasi Laakso, Matti Paljakka, Petteri Kangas,
Atte Helminen, Jyrki Peltoniemi & Toni Ollikainen

Methods of simulation-assisted automation testing

VTT TIEDOTTEITA – RESEARCH NOTES 2289

Methods of simulation-assisted automation testing

Pasi Laakso, Matti Paljakka, Petteri Kangas, Atte Helminen,
Jyrki Peltoniemi & Toni Ollikainen

VTT Industrial Systems

ISBN 951–38–6542–8 (soft back ed.)
ISSN 1235–0605 (soft back ed.)

ISBN 951–38–6543–6 (URL: http://www.vtt.fi/inf/pdf/)
ISSN 1455–0865 (URL: http://www.vtt.fi/inf/pdf/)

JULKAISIJA – UTGIVARE – PUBLISHER

VTT, Vuorimiehentie 5, PL 2000, 02044 VTT
puh. vaihde 020 722 111, faksi 020 722 4374

VTT, Bergsmansvägen 5, PB 2000, 02044 VTT
tel. växel 020 722 111, fax 020 722 4374

VTT Technical Research Centre of Finland, Vuorimiehentie 5, P.O.Box 2000, FI–02044 VTT, Finland
phone internat. +358 20 722 111, fax +358 20 722 4374

VTT Tuotteet ja tuotanto, Tekniikantie 12, PL 1301, 02044 VTT
puh. vaihde 020 722 111, faksi 020 722 6752

VTT Industriella System, Teknikvägen 12, PB 1301, 02044 VTT
tel. växel 020 722 111, fax 020 722 6752

VTT Industrial Systems, Tekniikantie 12, P.O.Box 1301, FI–02044 VTT, Finland
phone internat. +358 20 722 111, fax +358 20 722 6752

Otamedia Oy, Espoo 2005

Laakso, Pasi, Paljakka, Matti, Kangas, Petteri, Helminen, Atte, Peltoniemi, Jyrki & Ollikainen, Toni. Methods of simulation-assisted automation testing. Espoo 2005. VTT Tiedotteita – Research Notes 2289. 59 p.

Keywords process simulation, automation systems, testing, apros, specifications, system design, implementation, installation, commissioning, validation

Abstract

The recent developments of the hardware, software and communication standards have enabled the usage of simulation to test the automation application. The process-wide simulation models can be executed using inexpensive standard hardware. The standardisation of the communication has decreased the costs and effort needed to connect the simulator to the automation system. To further ease the task of the test engineer, working methods and dedicated tools are needed for requirements management, test case generation and execution and analysis of the test results.

This document presents a working methods for simulation assisted automation testing. Furthermore the possibilities of the simulation during the automation system life cycle and requirements for the tools supporting the working methods are discussed.

Preface

The paper is one end product of the research project Testing Manager. The goal of the project was to specify working methods and develop software tools to be used for simulation assisted automation testing. National Technology Agency (TEKES), Fortum Nuclear Services, Metso Automation and VTT Industrial Systems funded the project.

In the course of the two project years 8 research scientists and trainees from VTT Industrial Systems, Fortum Nuclear services and Helsinki University of Technology have participated to the project.

The authors wish to thank all parties of the project and all people that have given their contributions to this research.

Espoo February 25th 2005

Authors

Contents

Definitions

Automation application is the implementation of the functionality of the automation system created using the automation system specific functions and modules.

Automation hardware is the hardware on which the automation application and software are executed, when installed to the real plant.

Automation software is the software, which executes the automation application.

Automation system consists of automation hardware, software and application.

Customer is the end user of the automation system.

Phase test plan defines why the testing should done during the phase and how it is done. E.g. which modules are tested and what are acceptance criteria.

Project test plan defines how the testing is done in the actual project. This document is connected to the requirements specification. Project test plan follows the test strategy when applicable. It contains e.g. timetables, resources and risks.

Simulation system consists of a simulator, an automation system and a communication link between them.

Simulator emulates (imitates) the behaviour of the real process.

Test policy directs how the testing is performed and documented. A process industry customer may require this from the automation system supplier.

Test strategy is a high level document, which defines the phases of the testing. The same strategy is used from project to project and it may define e.g. how the modules and systems are tested.

Virtual automation is a setup in which automation software and application are executed on an alternative platform e.g. PC.

Acronyms

DCS, Distributed Control System

EPC, Engineering, Procurement and Construction

FAT, Factory Acceptance Test

LNG, Liquefied Natural Gas

LNGC, Liquefied Natural Gas Carrier

NPP, Nuclear Power Plant

OPC, OLE for Process Control [OPC]

OPS, Operator Station

PLC, Programmable Logic Controller

SCADA, Supervisory Control and Data Acquisition

SAT, Site Acceptance Test

1. Introduction

1.1 Background

Requirements of the automation system delivery are tightening. The organisations involved in the delivery should achieve even higher productivity without compromising the quality of the automation system. The testing of the automation system has essential role when aiming at these goals. However extensive testing of the automation system is challenging. Automation systems are wide and they are tightly coupled to the process, and all information required is available only at the time of the installation. In traditional testing the functionality of the process is emulated using signal generators and automation system modules. To get more realistic process responses including all interdependencies a simulator is a very attractive alternative.

In simulation assisted automation testing the automation application is connected to a plant-wide dynamic process simulator. This environment can be used to verify the functionality of the automation system before connecting it to the actual process. This idea has several benefits when compared to traditional testing methods. Especially the whole automation application can be tested with plant like responses before installation. These tests can include also scenarios, which would be impossible to carry out at the site, because of high risks or costs. The tests made beforehand can significantly reduce the time required for the site acceptance test (SAT).

Recent development has made simulation-assisted automation testing a viable alternative. Computation power has rocketed in relation to the price: modern PC's are capable of running plant-wide models in real time. There is no longer a need to have expensive dedicated simulation server computers in the office. PC-based DCS and PLC emulation software products, also known as virtual automation systems, have been released, with basic simulation features (start/stop/save/load). There is no longer a need to purchase a second set of automation hardware for simulation use, as the entire system can run on regular office computers. Neither is there a need to build a simulation model of the automation in the simulation tool, as the original automation configuration can be used as such.

Simulation tools have developed: solvers have been validated, comprehensive model libraries have been built, and easy-to-use graphical user interfaces have been implemented. The manpower needed for building up a simulation model of a plant has significantly come down.

Industrial standardisation efforts, e.g. OPC Foundation [OPC], have produced a number of de-facto standards for plug-and-play co-use of simulation and automation software

by different providers. There is no longer a need to write product-specific driver code to bridge software together.

The technology is now being taken up in the industry. During recent years, pilot projects have been carried out using various tools and technologies with promising results. It is easy to see that the technology will have an impact in the work practices of both individual companies and virtual enterprises.

1.2 Scope of the study

This paper present a set of working methods for simulation assisted automation testing. The paper is intended mostly to automation engineers in automation system supplier companies, consulting firms and end user organisations. The emphasis is on large-scale dynamic process simulation of continuous processes in the power, pulp and paper industries. However the presented ideas are quite general and may have value also in other areas.

The simulation models range is typically plant or sub-process wide. Also process component wide models can be used but main emphasis is on larger models. It has been assumed that the accuracy of the simulation models is high enough so that the difference between the real process and the simulator can be ignored when executing test runs. The simulation model should include both the functionality of the process and other hardware components typically between the process model and automation system. These include e.g. measurements and actuators.

Automation systems are typically categorised to Programmable Logic Controller (PLC), Distributed Control System (DCS) and Supervisory Control and Data Acquisition (SCADA) systems. The presented methods are suitable for any of these system types. Especially due to its plant-wide nature, simulation is most useful when used to test high-level automation systems, which widely affect to the process e.g. upper level controllers and sequences.

Testing is only a part of the automation system delivery. Therefore the other phases of the automation system delivery are addressed in this document. Testing has an especially strong connection to requirement management, as testing is done to verify that the automation application fulfils its requirements.

This paper focuses on the testing of the automation application, which is only a part of the automation system. Typically tests are executed using the virtual version of the automation system, i.e. the automation application is executed on an emulation platform. However the ideas presented are also applicable to the stimulation setup

where the automation application is executed on actual hardware and the combination is tested using a simulator. At the moment the realisation of extensive tests using this approach is troublesome because typically stimulated automation lacks most of the basic simulation features, e.g. save and load of the automation application state, or freeze and resume simulation run.

A simulation model can be used for versatile purposes besides automation testing, which makes the building of the simulation model more worthwhile. This paper addresses also these uses although they are not in the core focus of the paper.

1.3 Basic concepts

The user of the automation system sets the primary requirements for the automation system (user requirements). Requirements define that the delivered system provides certain functionality (functional requirements) and that it fulfils quality related requirements e.g. easy maintenance, reliability, performance and usability. The main goal of simulation assisted automation testing is to verify that the automation application fulfils the functional requirements and to improve the quality of the automation system. From above categories simulation suits best to test and improve following aspects of the automation system:

Functionality – are the functional requirements of the automation system reasonable and will the design and implementation meet the functional requirements.

Reliability – is the implemented system reliable enough and how well the system will meet the reliability requirements. A reliable system works as expected during the normal operation and during the process disturbances and other unexpected cases (operator errors) the automation system ensures that the safety of the personnel or installed hardware is not compromised.

Performance – is the technical performance of the automation system (software and hardware) satisfactory and how well the implemented system meets the performance requirements. Testing the hardware performance with the aid of simulation is problematic at the moment, as typical automation system hardware does not support features that would enable large scale testing.

Usability – "Operability" – is the automation system usable for operators and how well the system meets the usability requirements. Testing of this feature using simulation is beyond the scope of this paper.

1.4 Goals of the study

To take full advantage of simulation new working methods and tools are needed. This paper suggests new methods and sketches what kind of tools would be needed and what tools are already available.

Intuitively, the testing methods and tools should include at least following features:

Test runs – The automation system on the plant must be able to react to both operator events and changes in the process state. Hence, test runs in the testing environment can be seen as sequences of process events (malfunctions, boundary changes) and operator events (setpoint changes etc.).

Reusable test cases – There may be a set of test runs related to the same test case. Test cases must be transferable from project to project.

Traceability – Each test case is connected to some user requirement. The test environment must give support for connecting the test cases to the user requirements and to the task of defining the test cases from the user requirements.

Repeatability – The execution of identical test cases should always provide identical results. This sets requirements to the infrastructure of the environment and all automation and simulation software involved.

Test coverage analysis – The testing environment must include features to analyse how widely the automation system functionality has been tested. (See chapter, Test procedures)

Reliability and Performance analyses – The environment must include tools to estimate the reliability and performance of the system according to different metrics.

2. Automation delivery project phases and simulation

Independently of the working practices used in testing, an automation delivery project can be divided in following phases [SAS 2001]:

1. In the **Specification** phase, the end-user specifies the requirements for the new automation system. The choice may be open between several possible automation suppliers.

2. In the **Design** phase, the end-user and the selected automation supplier resolve how the requirements will be met on the selected automation product.

3. The **Implementation** phase involves the configuration and/or programming of the automation applications. The phase ends in the Factory Acceptance Test (FAT).

4. In the **Installation** phase the automation hardware and software are delivered and installed to the site. The phase ends in the Site Acceptance Test (SAT).

5. The **Commissioning** phase involves the trial runs: cold commissioning and hot commissioning. After the commissioning phase, the plant is taken over by the end-user.

6. In the **Validation** phase, the system is evaluated by the end-user and possibly authorities.

7. **Operation** can on most application domains start right after the Commissioning.

Simulation can be used as a tool in all these phases except Installation. In the next chapter, the automation delivery project is represented and the current utilisation of simulation is reviewed. Finally, new ways of simulation-assisted automation testing are shown.

The simulation should be applied through the whole life cycle of automation. This way the benefits of simulation can be maximised.

2.1 Current use of simulation in automation delivery project

This chapter presents a typical course of an automation delivery project together with notes on the use of simulation in each phase so far. A more detailed description of the delivery project is in [SAS 2001].

The Figure 1 shows the process of automation delivery project.

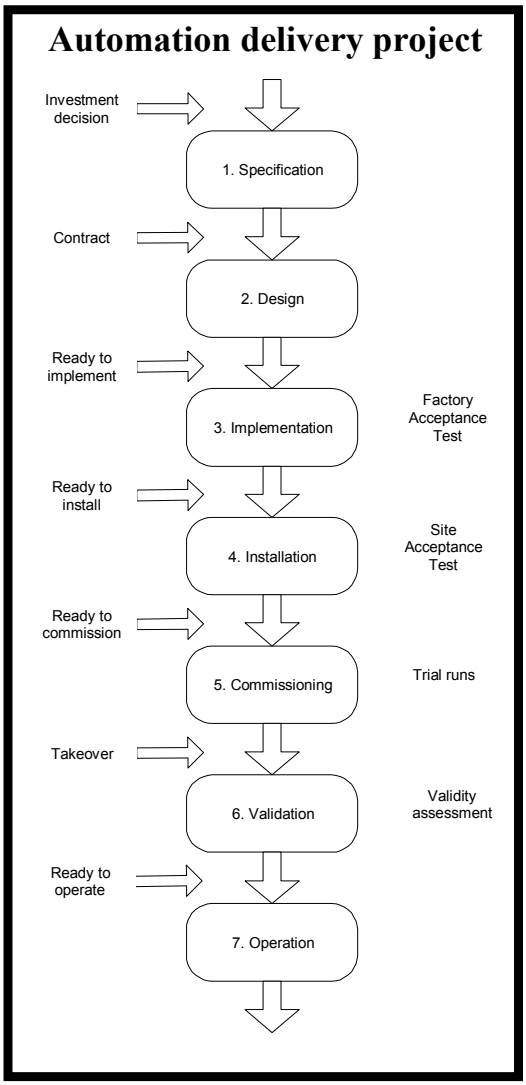

Figure 1. Automation delivery project phases [SAS 2001].

2.1.1 Specification

The requirements of automation must be defined before selecting the automation supplier by the customer or a consultant representing the customer. This phase of the project is also known as the preliminary design phase.

The user requirements comprise

- Documentation on functional requirements
- PI diagrams
- Device descriptions
- Signal lists describing the interface between the process and the automation.

Potential suppliers, applicable automation type, available resources and the automation degree has to be determined in preliminary design. The customer invites tenders based on this information.

A functional specification document is defined during basic design. The agreement between the customer and the supplier is based on this document. Basic design is done in co-operation with the customer and the supplier. Operating principles are discussed in meetings. The requirements of the preliminary design are used as input. The functional specification document may include [Ahonen 2004]:

- System description
- Functions of the automation system
- Information the system will use in operation
- Interfaces
- Non-functional features (availability, maintainability).

There are several web-based tools for manage design information. These tools allow flexible workflow of basic design. In addition this documentation can be easily converted to help documentation of the automation system [Paljakka 2003].

At the end of specification phase, the customer makes a contract with the supplier to build the automation system.

Currently, there is no standardised way to specify the requirements. Indeed, there is a need for ways to express user requirements more formally. It is also to be noted that the current set of requirements mainly address the functionality and the physical structure of the system to be delivered. There are fewer requirements related to reliability, performance and usability.

There are a few examples of cases in which simulation has been used in specification phase [Rinta-Valkama 2000, Yli-Petäys 2001 and ProcessVision 2000]. In these cases, the use of simulation has brought about clear benefits by improving the mutual understanding on the feasibility and the goals of the project and the validity of the requirements.

2.1.2 System design

In the design phase, the selected automation supplier defines how the system will be implemented. The set of standard solutions in automation product is selected. The

supplier produces a software design specification and a hardware design specification [Ahonen 2004]:

- Electrification diagrams
- Hardware descriptions
- Software descriptions.

Realising the functional specification is the objective of all design solutions.

Simulation has not been used during the design phase probably because so far simulation tools and automation-engineering tools have not been integrated to the extent that the use of simulation would be feasible at this phase. By using simulation, the automation design engineer could test alternative solutions.

2.1.3 Implementation

The supplier purchases, manufactures and assembles the automation system specified in the design phase during the implementation phase. Software programming and device configuration are parts of the implementation. Several tests are made [Ahonen 2004]:

- a modular test level, in which groups of interactive control modules are tested to verify their logic
- integrated internal non-customer witnessed test that is performed using the system hardware and software
- customer witnessed Factory Acceptance Test, FAT.

When automation system passes factory acceptance test, it is ready to be delivered to customer's site. This is an important checkpoint for both customer and supplier. The importance of factory acceptance tests is acknowledged. [Paljakka 2003]

Simulation is used in implementation phase, mainly in FAT [Paljakka 2003]:

- Modern automation systems can typically be run in simulation mode, which means that a control causes a response signal. E.g. when a pump is started, the system responds as if the pump had started. This feature is used practically by all automation vendors.
- Several companies have implemented transfer function models that produce process response for the controls. The models have been typically implemented using the tools of the automation system platform.

- Large-scale simulation has been used in a few cases [Rinta-Valkama 2000, Yli-Petäys 2001]. The benefit from large-scale simulation compared to the other types of simulation is that also the cross-dependencies between the control loops are taken into account.

2.1.4 Installation

Automation system is delivered to its final location. The system is installed and connected to the field devices. Software and hardware from several suppliers has to be engaged. Site/System Acceptance Test, SAT is done to ensure the functionality of all components.

Following tests are made:

- The signal and loop testing, SLT. Signals from field device to screen of automation system are tested.
- Functionality and calibration of field devices.
- Connections to other systems (buses etc.).

SAT is understood to prove that the system tested in the FAT is properly connected to the field devices (SAS 2001).

Simulation has not been used during the installation phase, and it is difficult to see any possibilities to use simulation in this phase.

2.1.5 Commissioning

In commissioning, the functionality of the whole system is tested in its final operation environment. The cold commissioning is done by using water or some other inert substance as the process media. The hot commissioning is done with the actual process media. The automation system is finally ready for production after commissioning.

Short commissioning times are within everybody's interests. The customer wants to operate plant as soon as possible. The supplier prefers fast release of the automation system.

The simulation has not been used during commissioning phase, however, simulation can be used for rehearsing the commissioning, which significantly accelerates commissioning phase. It may be a good idea to run at FAT the set of simulation runs that are going to be run at commissioning on the site [Rinta-Valkama 2000].

2.1.6 Validation

The final phase of the automation project is its validation of the delivered system. Validation is made by the customer and it is a proof that the system produces continuously the outcome specified in the beginning. A great deal of the validation is based on the material formed during the project. Process industry does not necessary validate the automation and process. Operation phase begins right after the commissioning. Authorities require the validation phase in some special fields of process industry, e.g. nuclear and pharmaceutical industries.

Simulation is currently not used in validation phase. In case the validation procedure involves some validation runs on the plant, it is surely a good idea to execute these on the simulator before executing them on the site.

2.1.7 Operation

Simulation has been used during operation to develop new control strategies and operational practices, e.g. [Lappalainen 2003]. The improvements made based on simulation have saved significant amounts of money.

Another traditional and popular use of simulation during the plant operation is operator training, e.g. [Yli-Petäys 2001]. Simulation is an invaluable help in transferring knowledge from experienced operators to newcomers.

2.2 Potential uses of simulation

Ten different ways to utilise simulation-assisted automation testing can easily be listed.

1. Evaluation of automation design
2. Evaluation of process design
3. Validation of system requirements
4. Comparing automation products
5. Verification of automation implementation
6. Tuning of automation parameters
7. Validation of process models
8. Estimating system reliability
9. Verification of operator instructions
10. Testing automation and simulation products at version release.

The relations of these ideas to the phases of automation delivery project are presented in the next picture.

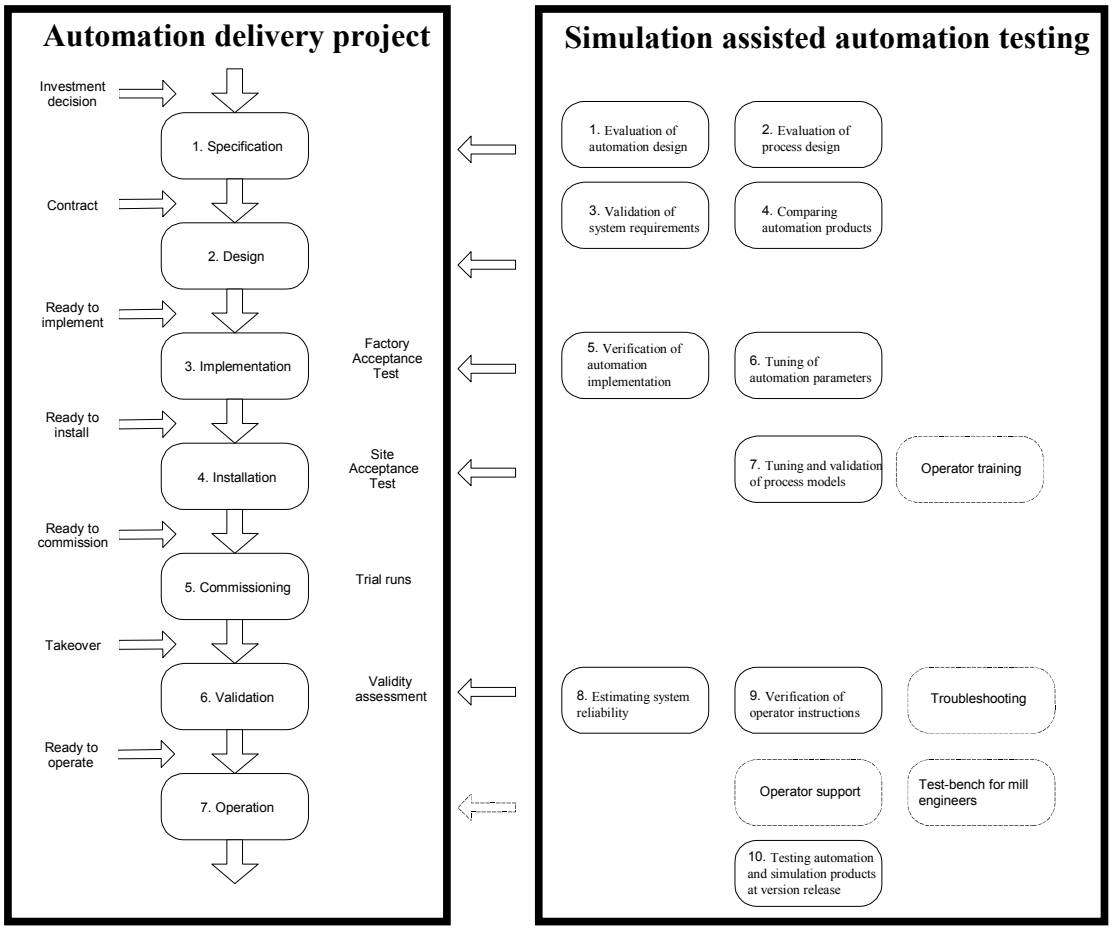

Figure 2. Utilising simulation during automation delivery project.

2.2.1 Evaluation of automation design

The response of a plant-wide process model brings about most benefit in testing of plant-wide phenomena. The problems that are found when running plant-wide simulations are more likely to be related to design than implementation, i.e. control strategies and cross-dependencies of control loops.

Simulation-assisted evaluation of automation design should be carried out as early in the project as possible. As the automation platform product may not have been selected, simulated automation is to be used.

Simulation sequences can be made for evaluating system performance with alternative automation designs. The testing of cross-dependencies and overall system behaviour must be carried out interactively by domain experts.

2.2.2 Evaluation of process design

As processes become faster, the process and automation designs become more iterative and more integrated. Especially it may be hard to tell whether a performance related requirement is a requirement from process or automation. The use of dynamic simulation to evaluate alternative process designs can be seen highly beneficial. Naturally, the most essential controls need to be included in the simulation model in order to have qualitatively correct dynamic behaviour for the system.

As to the use of simulation, the same applies as for the evaluation of automation design. Some of it can be made automatically, but a lot of it must be done interactively.

2.2.3 Validation of system requirements

Simulation can be used for improving the understanding, what are realistic expectations from the performance of the system, and what are justified limits for alarms and protection logics. These simulation runs must naturally be carried out early in the project as they have an effect on the automation system specifications. As the automation platform product may not have been selected, simulated automation is to be used.

For validating the requirements, automatic simulation sequences can well be used in case the criteria for the validity of each requirement are easy to specify. The runs can later be re-used in the comparing of automation products and in verification of system implementation.

2.2.4 Comparing automation products

Provided that the testing system uses open standards that are implemented in several automation platform products, the testing system can be used for running pilot-scale tests using different automation products.

For running these simulations, simulation sequences configured for evaluating the automation design and for validating system requirements can be re-used.

2.2.5 Verification of automation implementation

Simulation-assisted automation factory acceptance test is generally regarded to have a great potential benefit. Changes made before installation are far less expensive than changes made to a system after installation.

In factory acceptance tests, the system is evaluated based on the requirements and in most cases demonstrated to the customer. If simulation is used, the project staff can practise the trial runs. The idea is that this way, most of the flaws that would be caught on site, will be detected before installing the system, which will shorten the time spent on site and enable earlier take-over of the system.

Furthermore, in case the system will go through some validation procedure, it is surely a good idea to run the validation runs on the simulation system before installing the system and running the validation runs on the plant.

The test runs need to be selected in a manner that the benefits of plant-wide models are utilised as much as possible. Compared to simpler process response, the key benefit of a plant-wide model is that the plant-wide simulation system will demonstrate, how the entire system will behave, when the new automation has been installed.

Instead of verifying each individual feature of each control loop, it is probably a good idea to use the simulation system for testing how the system-level requirements are met, in which case the simulation sequences made for the validation of system requirements can be re-used. In addition, if the system contains several similar parts, the test environment can be used for automatically performing the same test to each part. But a lot of the factory acceptance testing remains to be done interactively by the project staff that will carry out the trial runs.

2.2.6 Tuning of automation parameters

Simulation models can be used for pre-tuning of automation parameters. It may be beneficial to tune the automation in both specification phase, using a simulation model of the automation, and in the implementation phase, using virtual automation. The tuning in the specification phase results in an understanding on the potential performance of the control system and helps to validate the requirements. The tuning in the implementation phase results in a good starting point for final tuning, which remains to be made on site.

The tuning is an example of the use of automatic simulation run sequences. These sequences may become rather complex, hence the testing environment should include some visual tools for specifying the sequences and for viewing the results. The benefit of using plant-wide models is that the tuning may involve qualities and qualifiers from all over the model instead of focusing on one control loop at a time.

2.2.7 Tuning and validation of process models

It is assumed that the simulation tool used in the testing environment is capable of producing a valid and accurate enough simulation model based on the design data. However, it may be desirable to tune the model further based on measurements obtained from the plant. This can be done after the new automation has been taken into use. Same tuning algorithms and a similar approach can be used for tuning the simulation model as for tuning automation parameters.

Another use for tuning the simulation model is related to the development of the simulation platform. The tuning algorithm can be used for finding parameter values for unit operation-specific correlations.

2.2.8 Estimating system reliability

Simulation can be used for estimating the reliability of the automation application. A set of test cases that correspond to the operational profile are specified and executed. The number of test cases depends on the goal set for the reliability and on the resources in use. Instead of finding errors, the aim is to demonstrate that the system has the expected degree of reliability.

2.2.9 Verification of operator instructions

The testing tool can send both operator events to the automation system and process events to the simulation model. By running sequences of operator events that correspond to operator instructions, one can verify that the instructions are correct. The cases can be used as exercises in operator training, and the run results can be used as reference runs.

The simulation system can also be used to test the actual usability ("operability") of the automation system. Virtual automation system provides operator displays, which correspond the displays of the real plant. The simulator system can be used to initiate

different transients and experienced operators can be used to verify that the all the needed data and controls are available and sensibly presented so that operator can make correct decisions and can carry them out efficiently.

2.2.10 Testing automation and simulation products at version release

Both automation applications and simulation models used in the testing environment are assumed to consist of system software and application/model configuration. At version change of any system software component, the behaviour of the entire system should remain the same, or the change should be justified. Therefore, the testing environment can be used for testing automation and simulation products at version release.

2.2.11 Other uses during the lifecycle of automation system

When simulation-assisted automation testing is fully utilised the following additional benefits are gained.

Operator training

Operator training can begin as early as during the implementation phase if process model with simulated automation is available, or at the time of the Factory Acceptance Tests the latest. Naturally, operator training can continue during operation phase, provided that the model remains available to the end user after the project, and it is maintained whenever there will be changes in the plant.

Operator support

Tuned and validated simulator can be used as a tool for operator support. Forecasting future scenarios is possible when faster than real-time simulation is executed. This supports operator's decisions.

Troubleshooting

Simulated process and automation are used for troubleshooting: tracking pulsation, estimating time delays and reconstructing problematic situations are examples.

Test bench for mill engineers

Simulator can be used for testing new process and automation solutions. New process connections can be made and new devices can be tested virtually. Intelligent controlling methods can be tested and developed.

2.3 Simulation-assisted automation testing cases

2.3.1 Introduction

Simulation assisted automation testing has been already used in some cases. This chapter present cases where the idea has been utilised earlier.

2.3.2 Staging the Controls System with a Dynamic Simulator for a Pulp Mill

A new pulp line "C" was build at Aracruz Cellulose S.A. in Brazil. The line began operation on May 2002. Developing a simulator for control system testing and operator training was part of the project. The dynamic process simulator model included seven operating areas: Digester and Pressure Diffusers; Oxygen-Delignifigation and screening; Bleaching; Chlorine Dioxide Plant; Pulp Drying; Evaporation; and Recausticizing and Lime Re-burning. Process models were built with commercial modeling software and the DCS was emulated and connected to process simulator. Operator displays used were similar to the ones in the real plant.

Control system staging contained several steps. In the beginning the communication between the simulator and DCS was defined (I/O mapping).

Second step was to verify individual control loops. A working group including DCS technician, console operator of simulator, and configuration coder of DCS contractor was formed to carry out the tests. Three kinds of deviations were searched: deviations in the process models, deviations in the logic underlying the DCS control strategy and deviations in the DCS configuration coding. It was found out that verification should be done after the configuration of the DCS is close to final including the group start ups and sequences. Tested control loops included e.g. controllers, which handle the start up and shut down of motors. As an additional benefit the controller pre-tuning was also done and Mill operators had an opportunity to take more detailed look at the control philosophy, interlock strategies and the patterns of implementation. Testing revealed deviations in the process model and in the automation application. Typical errors in process model were configuration errors such as wrong pump parameters or wrong

address in configuration. The testing revealed also some errors in the control strategy and in the operator displays. All the deviations were added to a list that was delivered daily to the DCS coder for correction. Emergency corrections were also done immediately for items blocking the progress of the staging work.

Third step of the staging was the Joint Factory Acceptance Test. A working group including model developer, the DCS technicians, the Aracruz operators, the DCS coder and the EPC supplier's specialist was formed to carry out this step. Main emphasis was in the fidelity and truth of the model response. Several problems in the simulation model were identified and corrected in this step. Furthermore it was found out that training requires more detailed simulation model and needed adjustments were done. This phase was found useful to transfer the knowledge of the EPC supplier to the paper mill personnel.

Operators were trained before starting the new pulp line using the training simulator based on the testing simulator. The start up of the new mill was fast and effective. No problems were encountered regarding the control configuration prepared with the simulator [Bogo 2001].

2.3.3 Onboard Testing of the Control System in the LNG carrier

Simulation assisted automation was carried out for the liquefied natural gas carrier (LNGC) before connecting the control system to the ship. The motor of the LNGC consumes gas or oil to produce steam for ship thrust and electric power. The simulated process consists of cargo tank, supply of boil-off gas and oil, boiler system and steam turbine system. The control system of the process is very complicated and every error in automation system in a real ship is a safety risk [Chung 2002]. The developed simulator was also used in operator training.

The simulator of LNGC consists of process model that was connected to both real and emulated automation systems. The process model was build with block-oriented modeling technique. Each block was based on specific algorithms, which were generated using the Fortran programming language. Simulation algorithms were based on to first principles of physics i.e. to conservation equations of mass and energy. Simulation model generated and accepted the same electrical signals as the real process did. The model of LNG process was first connected to a replica of DCS. After laboratory testing the same hardware-in-the-loop (HIL) configuration was used in the LNGC ship. The connection between process model and the real DCS was made through A/D, D/A converter. Automation testing and validations were carried out by the personnel, which were in charge of the logic validation and operating. Thus, operator training and validation of the control system was made simultaneously.

The process model was also connected to emulated automation system, or software-in-the-loop (SIL) configuration through Windows based dynamic data exchange communication. This emulated system provided a stand-alone simulation package for training and severe transient analysis. Emulated control loops, logics and graphical used interface (GUI) were programmed and modeled with commercial software.

On-board testing was found to be efficient tool for control system verification. During the tests some errors in control logic were found, and these errors were interactively debugged in safe test environment before starting the real process in the LNGC ship. The ship operators also carried out training and testing of abnormal transient that hardly ever occur in a real life [Chung 2002].

2.3.4 Automation Testing of a Power Plant

Oil shale combusting power plant was modernized in Narva, Estonia. A new distributed control system (DCS) was tested in a dynamic simulation environment before commissioning during spring 2000. The power plant consists of 8 blocks each with two boilers and one steam turbine. Output of each block is 200 MW. The model of the power plant process was build with commercial simulation software and communication was based on OPC specification. The virtual version of the real automation system was used for the tests. Data for the model was obtained from the plant personnel and from the people involved in the automation design.

The automation testing was concentrated mainly on the control loops. Each control loop was checked to respond according to acceptable dynamics in the ramp response. After control loop tuning the simulator was used to define configuration of master controls that control the electric power, the turbine initial pressure and the pressure in two boilers. Different master controller variations were compared against each other to help choosing the most suitable one. The operation in steady state and load changes was checked and finally the automation system was verified to work properly in certain disturbance situations. [Rinta-Valkama 2000]

3. Testing of automation application

3.1 Introduction

Automation testing is meant to ensure that the automation implementation fulfils the user requirements. In this chapter the process of defining the requirements and testing whether they have been reached is discussed.

The development of an automation application resembles the development of software. In software development the importance of requirements management has been identified a long time ago. Plenty of literature and standards exist already [Sommerville 2001, IEEE: Std. 830-1998, ISO 9126]. In this perspective it seems a good idea to take the applicable ideas and terminology used there into use also in automation engineering. In this paper requirements management refers to all tasks concerning the requirements during the life cycle of the automation system. This includes methods to gather, define, change, verify, categorise, prioritise, maintain and document the requirements during the whole lifecycle of the process plant.

3.2 Automation delivery project and testing

Testing the automation application means a systematic way to search errors, verify that the required functionality and quality (reliability, usability, maintainability etc.) has been reached. In typical cases the functional requirements have been specified in detail. However the specification of quality-based requirements is less extensive and detailed.

Testing can be seen as a subproject of the whole project and it is done in all phases of the automation delivery project. A general test process can be described using the Figure 3. The process also includes supporting functions, not described in the figure, which are used to develop the test process and to modify test cases.

<table>
<tr><td colspan="4">Planning</td></tr>
<tr><td>Test case planning</td><td>Performing test cases</td><td>Test results</td><td>Acceptance criteria</td></tr>
<tr><td colspan="4">Control</td></tr>
</table>

Figure 3. Testing process [Stenberg 2003].

The planning is based on the test policy and strategy of the parties involved. The policy refers to the attitude towards the testing e.g. how the tests are generally executed and how the documents are inspected. Typically the strategy defines the criteria for starting and stopping of the tests, and answers to general questions what should be tested and when. The results of the planning are a test plan and a phase test plan (here jointly called test plan), that describe how, why and what test cases will be carried out, e.g. which modules are tested and what are the acceptance criteria. These documents should be tightly connected to the requirements document and they should define the needed resources and time tables for the testing.

The controlling includes time schedule updates, and how the results of the tests are reported to the management.

Test cases are based on profound understanding what must be tested and what can be tested. The inputs and desired outputs are defined based on the requirements, definitions, plans, program code and experience. The test cases are planned so that their execution is easy enough and they uncover the errors.

Application is tested using predefined inputs. The testing methods can be separated in two groups: functional (black box) and structural (white box) methods. In black box methods there is no need to know the implementation, the inputs are given and outputs are followed, whereas the white box methods are based on analysing also the program code and internal state of the application during the tests. In practice the tests done have often features both from the black box and white box testing. Typically the nature of the tests changes from white box to black box when moving up from a lower level (module testing) to a higher level. The test is successful if the output (black box testing) and the internal state (white box testing) of the application are correct after and during the tests.

Testing can be either manual or automatic. In manual case the simulation system is operated like the real plant. In automatic case simulation sequences are used to execute the test cases. In both manual and automatic case the test logs including the events and stored input and output data are essential. Automatic testing is especially useful if the same test must be done several times to the same system or there are several systems, which typically should behave similarly. At the moment the automation application is typically tested manually because of the lack of needed tools. The prototype tools developed in the Testing Manager project enable the configuring and execution of test cases.

The test results are compared to the expected results and then the decision is made whether the system fulfils the criteria stated. If there are deviations, they are written down to be corrected later. Furthermore the project management must be informed on the progress of the tests. In some cases it may be necessary to create new tests or modify existing ones during the testing.

The test case specific ending criteria are described in the test plan. Furthermore overall ending criteria can be used. These include criteria for coverage: how extensively the interfaces, modules, use cases and requirements have been gone through or has the required reliability been reached. Furthermore other overall criteria can be specified, e.g. the time used for the testing and the number of errors detected.

3.3 Requirements management

Requirements are needed to form a common understanding between the different parties about the functionality and features of the application to be delivered. It may be that in a real world automation delivery project a separate requirement document does not exist. However the corresponding information is written down during the negotiations between the customer and provider. The negotiation phase can be called requirements analysis. Some of the requirements customer has stated may be impossible to implement or unnecessary and are therefore removed at this phase. The documents created during the negotiations are typically called: preliminary design documents, offer request, functional specification and contract [AUTOHJE 2005]. At later phases the requirements are thoroughly processed and specified in more detail. Finally after the system has been delivered to the end user all requirements specified have been implemented and tested.

In this paper all the documents which define the requirements are called requirements specification. All requirements should include following information: identificator, description, reasoning, references, corresponding tests, priority, connections to other requirements and revision history. The requirements can also be validated and verified. Validation is needed to make sure that the requirements define the system that the customer requires and verification is needed to check that the actual implementation of the automation system corresponds the user requirements.

3.4 Testing of different requirements

In the previous and present sections the different phases of an automation delivery project and the different testing phases of an automation application are described. The phases of automation delivery project illustrate the overall completion of the automation system while the different phases of automation application testing try to ensure the fulfillment of different requirements set for the application. Subsection 3.3 discussed the process of defining and managing different requirements for automation applications. In the following the focus is given to following types of requirements: functional, performance and reliability requirements.

The specified tests are typically run to verify the fulfillment of certain requirements. Therefore, it is justified to call the testing as functional, performance or reliability testing depending on the type of requirement under verification. Characteristic to the functional testing is that it is started already in the early stage of a system development process and its purpose is to ensure that the applied control algorithms and their parameters behave as planned. In the performance testing the performance of hardware components of the automation application are introduced, or simulated, for the system, or the system model. The purpose of performance testing is to ensure the correct functionality of the system under specified performance, i.e. time and accuracy, constraints. Finally, the reliability testing is conducted usually when satisfying results from functional and performance testing have been received. Opposite to functional and performance testing the purpose of reliability testing is not only to uncover and correct faults in the system, but also to ensure that there do not exist faults in the system to certain confidence. An important part of reliability testing is the specification of the operational environment since in reliability testing the probability of the selected test cases should correspond to the actual use, i.e. operational profile, of the system as well as possible.

Figure 4 illustrates how the different phases of the automation delivery project, the phases of automation application testing and the verification of different requirements are traditionally sequenced in a process automation delivery project. From the figure we can see that the testing is mainly carried out in the design, implementation and installation phases of the automation delivery project. The focus of testing is on the testing of functional requirements while performance requirements are typically verified at the later phases of the process, usually at the site, when the automation system already has been connected to the actual process. The reliability requirements are typically verified using the operational experience of the automation application.

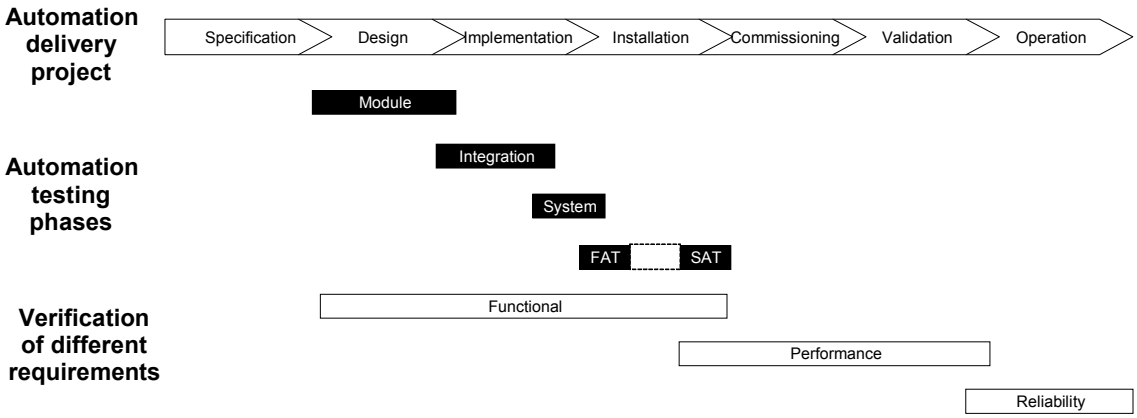

Figure 4. Chronological illustration of the phases of automation delivery project, the phases of automation application testing and the testing of different requirements.

One of the main advantages of simulated assisted testing is the ability to conduct testing in early phases of the automation delivery project. A simulation assisted testing environment does not only advance the testing of functional requirements, but also the performance and reliability requirements of the application can be tested in early phases of the project. In this research one of the key motivations was to design a testing environment with a possibility to test all kinds of requirements as early as possible in the project. In the following two subsections more detailed discussion is given on the testing of different requirements. In the report the term functional testing is used for the testing of both functional and performance requirements. The original plan in the Testing Manager project was to conduct reliability testing of the automation application as well; However, due to the piloting nature of the project and its limited resources there was not enough time for properly handling this aspect. Instead the principles of reliability testing and how to apply it in a simulation-based testing environment is discussed in subsection 3.6.

3.5 Testing Functionality

In software engineering the relation between design and testing is often described using the V-model. In this document it has been adopted to the automation application design and testing. The applied V-model is described in Figure 5. In V-model every design phase has a corresponding testing phase. During design phases the design documents grow in accuracy and detail step by step. The initial requirements transform first to the functional specification then to planning of the process or subprocess wide control concepts and finally to the functionality of single automation modules (also called circuit diagrams or control loops and interlockings). The test plan is specified alongside the design. The testing should be started from the modules and end up to the acceptance test of the whole automation system. The acceptance tests are typically done in two phases. First the concepts are tested in the automation system provider's premises (FAT) and after the installation the final tests can be executed (SAT).

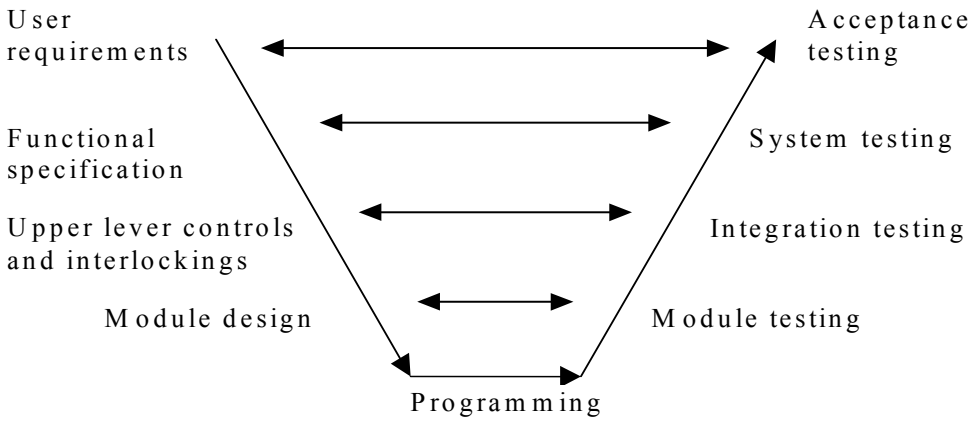

Figure 5. V-model for automation testing.

Functional testing concentrates on finding faults in the automation system. These faults could be categorized to two groups:

- Configuration faults - the application doesn't fulfill all its requirements. E.g. Alarm or interlocking is not triggered when it should or directation of the controller is faulty.
- Decision faults - mistakes that have been made in the design or specification phase. Even if the application fulfills all its requirements it might be that some of the requirements are contradictory or impossible. This causes need to change the original requirements e.g. The selected upper level control strategy isn't suitable or there are interlockings which prevent start-up of the plant. It has been found out that most of the errors in the automation system are caused by the design faults [POHA 2005].

At first the functionality of the elementary components should be verified. This is done already in product development before the automation delivery project. Simulation cannot provide much value for these tests.

If the functionality of the elementary components can be assumed correct, the next step is the testing of automation modules (or circuit diagrams). These tests verify that all connections between the components within the module are working and all inputs cause the specified outputs. Often, the tests cannot cover the whole address space of the possible inputs. However a reasonable coverage of all possible inputs is attainable. The input space can be discretised and all combinations of discretised inputs can be gone through. These tests can be done without the simulation model. However some value can be attained from a simulation model, e.g. correct directation of the controller can easily be seen. If the controller variable starts to drift to the wrong direction the directation is wrong.

After the modules of the automation system have been verified, the integration test can be made using different combinations of modules, which affect each other. These tests are very hard to perform without a realistic process response. The tests included to this category include:

- Sequences
- Upper level controllers
- Subprocesses
- Grade changes.

The next phases are the system tests, in which the entire automation system is tested as a whole. Simulation gives plenty of value for these tests. A simulator can produce realistic responses to typical cases, which need to be tested. These tests include:

- Typical operations like start up and shut-down of the plant
- management of the malfunctions and recovering from them (e.g. turbine trip)
- Possible Dead-lock situations and timings
- Messages. Do they arrive on time and does the acknowledgement work.

The system tests could be a part of the FAT during the automation delivery.

3.6 Testing reliability

3.6.1 Failure mechanisms

The reliability of computer-based systems is often divided into two subclasses: the software and the hardware reliability. The fundamental idea in the division is to emphasise the different failure mechanisms of the two reliabilities. The software reliability has generally been related to design and implementation faults, which may lead to a failure of the system under triggering conditions. In contrast, hardware reliability has been associated with ageing and external events, which usually lead to an immediate failure of the system. In modern computer-based systems with increasing hardware complexity the difference between the two reliabilities is not always so evident, since hardware may as well embody design, implementation and manufacturing faults similar to software faults.

A better approach would probably be to examine the reliability of computer-based systems over inherent, or systematic, and random failures. The systematic failures involve the design, implementation and manufacturing faults leading to a failure of the system every time the same input is given to the system with identical inner state, i.e. the parameter values in the system memory are identical. The random failures, on the other hand, cannot be repeated and are caused by sudden malfunctions of the system hardware. The different failure mechanisms of computer-based systems are illustrated in Figure 6. It is slightly modified version of a diagram presented in a report by Haapanen et al. [Haapanen 2004].

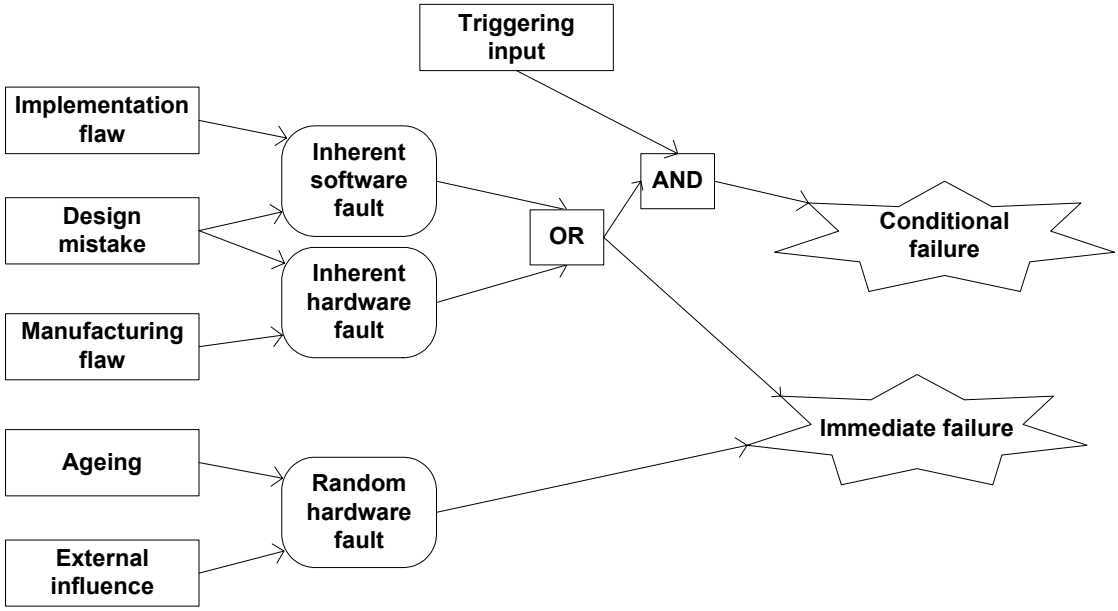

Figure 6. Failure mechanisms of computer-based systems.

In this study the key interest is on the systematic failures of the automation application software. Since the automation hardware is modelled only as a virtual automation, we can call the reliability under estimation as software reliability without any ambiguity.

3.6.2 Software reliability and reliability testing strategy

Software reliability is generally defined as the probability that software will not cause a failure of a system for a specified time under specified conditions [IEEE Std 982.1 – 1988]. According to the definition software reliability should be measured in terms of probability, and the most typical way of estimating probability of software reliability is by statistical testing. In statistical testing a sufficient amount of relevant test cases are run to ensure a required reliability level of the target application with a given confidence.

In this study a specific way of statistical testing called operational testing is applied. In operational testing the test cases are generated from the actual operational environment of the system. Parameters, their ranges and probabilities relevant to the operational environment are defined in a specific operational profile. The operational profile is then used to build a representative set of test cases for the system. A well-defined operational profile is one of the most important factors for the success of operational testing and reliability estimation in general.

The strategy of reliability testing in this study is rather straightforward and it is presented in Figure 7. For the beginning we have the functional, performance and

reliability requirements of the automation application. Next, the operational profile of the automation application is defined. Information from the operational profile is used to build the test cases for the testing environment. Operational testing is executed and the results are compared to the reliability requirements defined in the beginning. After the comparison a decision on the acceptance or further testing is made.

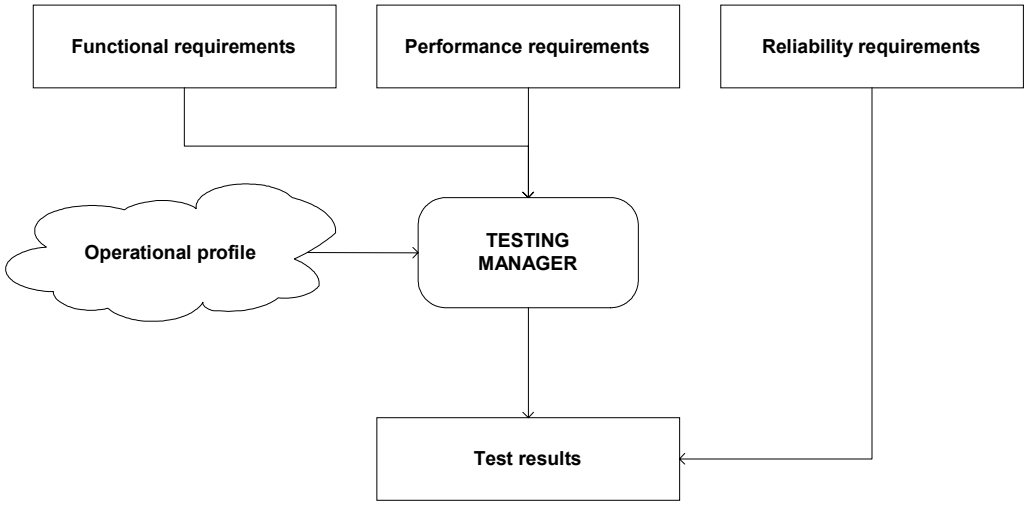

Figure 7. Reliability estimation procedure in a simulation assisted testing environment.

3.6.3 Applying simulation assisted testing for reliability estimation of automation application

In the thesis of Ahonen [Ahonen 2004] a case study on simulation assisted automation testing is carried out and the results are described. The case study, mainly concentrates on the functional and performance testing of the automation application. In the following some of the points given by Ahonen in her thesis are discussed from the reliability testing and reliability estimation points of view.

As mentioned in the thesis several technical difficulties were encountered in the project. The technical difficulties were mainly related to the incompatibility and communication delay between the simulator and the automation system. In spite of the common communication protocol it seems that there is still some work to do to make the different elements of simulation assisted testing to function properly together. Because of the incompatibility the behaviour of the controller was somewhat unrealistic, and the test runs designed for the automation application were not executed extensively. The replacement of the actual hardware-based automation application with a virtual one automatically creates some uncertainty to the validity of the testing results. Therefore, it is important to solve the technical difficulties as thoroughly as possible so that the other possible sources of uncertainty can be evaluated in the results of the reliability testing.

The creation of a correct operational profile for the automation application is an important task in reliability testing. Information for the operational profiling is usually found in the user requirements. In the thesis Ahonen pointed out that user requirements do not necessary provide an automatic way of selecting test cases and test runs. User requirements only describe the different operational situations and possible transients having an effect on the system. User requirements do not explicitly state all different operation sequences and their probabilities. Therefore, a lot of work is needed in the later phases of an automation delivery project to generate an accurate operational profile for the application. The time spent in the analysis of the actual use of the automation system improves the quality of the operational profile, which in turn improves the validity of the final reliability estimation.

Despite the technical difficulties and the extra work needed for the generation of an operational profile simulation assisted testing is a promising way of carrying out reliability testing. As mentioned in the thesis the most benefits of the simulation assisted testing over the traditional testing methods is gained on the automation system level. Reliability testing is typically executed at the system level, and therefore tools specified in this study provide important support for the reliability testing. Also, as illustrated in Figure 4 the verification of reliability requirements is often left for the final phases of an automation delivery project. With simulation assisted testing the requirements can be verified earlier. This is particularly important in safety critical cases e.g. if the automation system is licensed for example for a nuclear power plant.

4. Test procedures

4.1 Introduction

This chapter contains description about the testing procedure and how the tests can be performed in practice. Some fundamental requirements are presented that must be fulfilled by the testing environment and the software involved before the testing can be done. The needed concepts are also presented. An example architecture of the testing environment is presented in Figure 8.

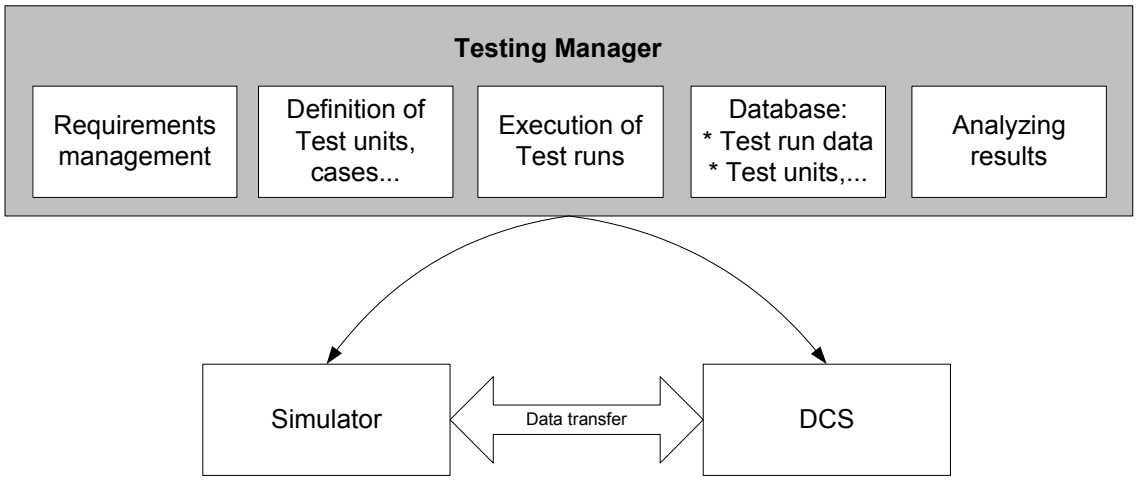

Figure 8. Architecture of the testing environment.

The testing environment can be seen as a set of tools, which together provide all required features. The tools must enable the requirements management, planning and definition of test units and cases. The execution of test runs must also be possible. The tool loads the correct initial state to the simulator and the automation system, turns them on and starts the communication between them. The tool enables the saving of old test cases and units to be used later and also the test run results so that they can be analysed later.

4.2 Fundamental requirements

A fundamental requirement from the testing environment is that there must be a way to establish a communication link between the simulator and the automation system. This link enables the transfer of data items between the simulator and the automation system and preferably includes also the functionality to synchronise the data transfer. The synchronised data transfer means that the process model and the automation system are executed using exactly the same clock. Typically synchronisation is implemented using synchronisation points. Both the simulator and automation system calculate a

predefined time forward, and the data is transferred between the communicating parties to each direction after the synchronisation point has been reached. After the data transfer both continue the calculation until the next synchronisation point is reached and data again transferred. For some test cases an unsynchronised link is sufficient but to reach repeatability of the test runs synchronised communication is required. It can imply that the simulation will not advance at real time, however the results will be correct in simulation time.

To be able to execute the test cases both simulator and automation system should support at least basic simulation commands: start, stop, save and load. Preferably also the speed of the automation system and simulator should be adjustable and not restricted to real time. This enables fast execution of test cases.

Sometimes it is enough if the simulation model is done using the components available in the automation system. The main advantage of this approach is that the communication and synchronisation problems do not exist. The main disadvantage is that the automation system may not suite well for building a plant-wide accurate simulation model. If reference data from the same or similar plant is already available they should be used to validate the simulation model. Typically data is not available and the simulation must be based on the well-validated simulation tools, which can generate accurate enough simulation results in wide operational range based on process design data only, i.e. process connections, heat balance calculations, equipment dimensions and boundary input values.

As to the automation system, normally the options are to use either a virtual automation environment or the automation components of a process simulation tool. A virtual automation environment inputs the same automation application and provides the same functionality as the actual automation hardware and software on the site. For simulated automation, the data contained in the automation application is required: automation component connections and parameters.

4.3 Tools

4.3.1 Requirements management

The management of the requirements is complicated task, which has to be done during the whole life cycle of the automation system. During the life cycle the requirements are added, changed, revised, removed and tested. In small projects these tasks can be done using simple office applications but on larger projects tools specifically made for this purpose are helpful.

A tool which is used for this purpose should have features which enable the categorisation of the requirements (e.g. using priority), attaching of key words to the requirements, linking to other requirements and generation of tailor-made reports. The tool should take into account the needs of different user groups. Project leaders, developers, testers and end users have different needs. The tool should enable simultaneous editing of requirements and take care of the security. Version control and traceability are also important features, which should exist in these tools. Integration to other parts of the testing environment is also essential. The planned and executed test cases and the test case results should be linked to the corresponding requirements.

4.3.2 Definition of test cases

This section describes how the test cases are configured and what concepts are used. The structure of the concepts is presented in Figure 9. All test cases defined are collected under the Project concept.

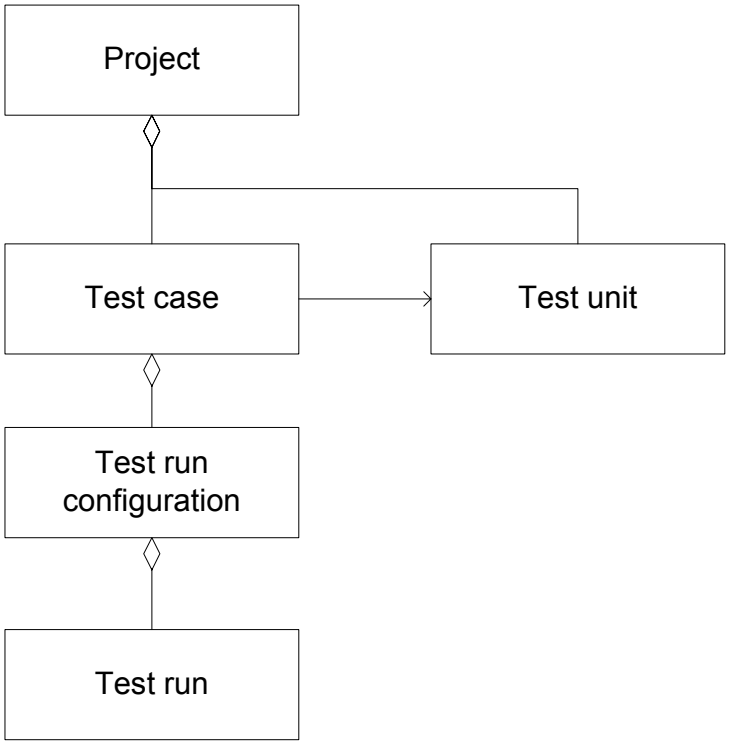

Figure 9. Main concepts of testing.

The planning of a test case is based on the requirement to be tested. The parts of the automation system needed for the test and the corresponding part of the simulation model and the communication are selected and a Test unit is formed. The effects of other parts of the automation system and the simulation model are assumed to be

irrelevant. The Test unit specifies an initial state that is used both in the simulator and in the automation system. If the initial state is changed a new Test unit must be created. The Test unit has a version number. After changes have been made either in the automation application or in the simulator, a new version number is given to the Test unit. The same Test unit may be used to verify also other requirements and several test cases may use the same Test unit.

The next step is to define what actually happens during the Test case and in which order. The Test case may be fixed or random. If the case is fixed, all events occur always in the same order and at the same time instance. In random cases some of the events occurring may have random attributes. E.g. the order of the events or the time instances, when the event occurs may be randomised. Furthermore events may have parameters. The value of a parameter can be either fixed or have some feasible values among which the value to be used is selected, e.g. the size of a leakage during an accident simulation. If random events are used, also some parameters are needed to define how many Test run configurations are generated using the Test case definitions.

Typical events include the start-up and shut down of the test case, operator actions and changes in process parameters. Possible Test cases are presented in Chapter 5. Typically a test case must define following things concerning the events:

- Order of events
- Occurring times of the events
- Parameters of events, e.g. stop of simulation can be done either based on the elapsed time or on the behaviour of a process variable.

A test case also specifies everything needed to interpret the results. All relevant variables from the simulator and automation system need to be logged. These values are saved when the test case is executed and used to analyse either automatically or manually, whether the test case has been successfully completed. If the results are automatically analysed, acceptable output values are specified. These can include upper and lower limit of the variable during the transient and the maximum gradient value. The criteria used in the Training Manager project [Lilja 2003] to define whether the student has been successfully completed the training case can be used as a basis for these criteria.

The test case definitions are used to specify Test run configurations. If the Test case is completely fixed, i.e. all events occurring have been unambiguously defined there will be only one Test run configuration. The Test run configuration always contains the same events at the same time instances. The execution of the Test run configuration is called a Test run. The same configuration can be executed several times, e.g. if the

version of the Test unit changes. Therefore the Test run configuration may contain more than one Test run.

4.3.3 Execution of Test runs

The test run configurations generated using the definition tools are executed using the Simulation sequence engine, a prototype implementation of which was done during the Testing Manager project. The engine is presented more detailed in [Mätäsniemi 2005].

Sequences consist of steps and conditions, which have to be met before moving forward. Sequences can have following types of steps:

1. Operator actions (set point changes)
2. Process events (malfunctions, changes in boundary conditions)
3. Simulation control operations (start up, shut down, load, save, run, stop)
4. Data handling (opening and closing logs, organising and post processing of data)
5. Sequence control (wait, verify, loop).

An example of a simulation sequence used in simulation-assisted automation testing is presented in the Figure 10. This kind of a sequence could be used for e.g. checking that protection logics are launched correctly. Some other use, e.g. controller tuning results in very different sequence configurations with nested iteration loops and versatile data handling.

The simulation sequences have also other usage. In the Testing Manager project the simulation sequences were also used to implement tuning tool. This use of simulation sequences is presented in Chapter 4.5.

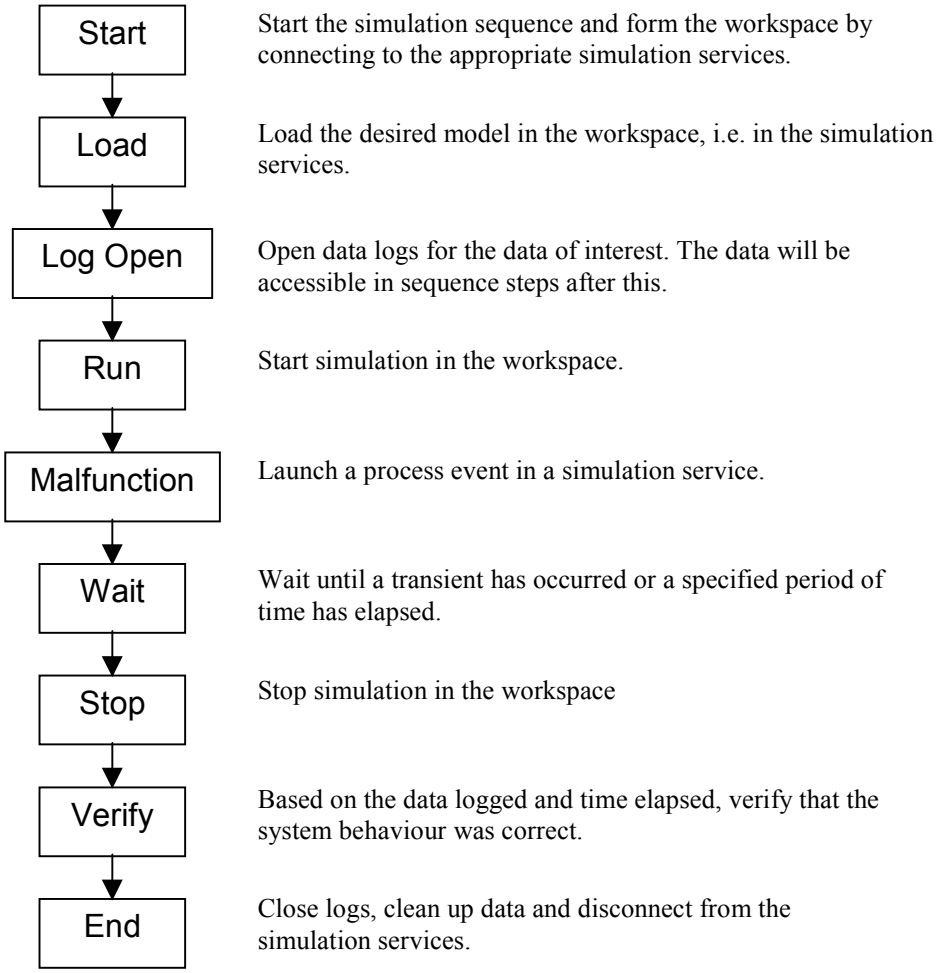

Figure 10. Simulation sequence example.

4.3.4 Storing and analysing of the results

All test run configurations and test run results are stored in a database. The configurations can be later re-used or copied to some other project. The test run results consist of the values of the defined variables during the transient with corresponding time stamps. This information is used to analyse whether the test has been successfully completed.

4.4 Testing phases

4.4.1 Introduction

It is assumed that the requirements stated in Chapter 4.2 have been fulfilled. A simulator model has been made, an automation system is ready for the testing and the communication between the automation system and simulator has been established.

Furthermore we have the requirements defined and a test plan, which includes a description of the test cases to be carried out. When starting the tests it is also assumed that the elementary components used from project to project have been tested before the project e.g. components like PI-controller is assumed to work as specified.

The phases of testing follow the V-model presented in Chapter 3.2. It is advisable that the testing will start from the bottom level modules and proceed towards the system level tests. All phases of the testing follow the pattern described in Chapter 4.3. At first the test unit is defined. If the tests only affect a small part of the process it is a good idea to disable other parts of the simulator and the automation system. When the test unit has been defined the next step is to define a test case. In the test case the possible events occurring during the test runs are specified, the data needed for the analysis and other information needed to generate test run configurations are given. The generated test run configurations are then executed. After the test run has been executed the results are analysed. If acceptable output values for the test have been specified, the automatic analysis of the results can be used. However user interaction may be needed when actually deciding whether the test has been successfully completed.

4.4.2 Module testing

Module testing or unit testing is typically done using the white-box approach. The testing verifies that the module works as specified when disconnected from the other parts of the system. The module testing can be done without an accurate simulation model. Realistic process responses are not a major concern. The main point is to test whether the different combinations of inputs generate desired outputs and whether the internal state of the module is correct during and after the test. Tests could be carried out even without a simulator by changing the of input values and following the signal paths and the outputs.

However a simulator can be helpful. It can produce a realistic feedback to the automation module and unrealistic input combinations can be ignored. A test unit is defined so that that the functionality of the other automation modules and process components can be ignored. This can be accomplished by disabling some parts of the simulator and by providing sufficient control to the other parts of the simulator so that they do not produce disturbances for the module testing. This is an important assumption since the controllers are connected together through the process. Even if the total DCS system may not be available at this point the simplified controllers are sufficient for the most of the module testing cases.

The main disadvantage of the simulator is that it may be a time-consuming task to run the simulator to all the states needed to go through the relevant inputs. The simulator must indeed be operated like a real plant in order to achieve the desired states and to execute the desired transients.

In the binary automation case testing is straightforward: the inputs (either analog or binary) are changed so that the testing covers either all or at least the most important combinations and signal paths. In the case of analog automation following other cases could be tested: the directation of the controller could be checked and the process response times could be used to tune the controller and check its parameters.

To ease the task of going through the inputs and outputs, the testing environment should provide a way to change the value of analog and binary inputs so that the signal paths can be checked.

The test coverage in module testing can be defined using the following list:

- which IO signals have been checked
- which automation modules (circuit diagrams) have been tested
- which signal paths and values have been tested (especially in binary automation).

4.4.3 Integration testing

During the integration testing several co-operating modules of the system are tested together. The main focus is to verify that the co-operation and communication between the modules work as planned.

This is the first level of testing where the simulation can be seen as a very valuable tool. The typical cases are upper level controllers connected to lower level ones, sequences and modules interconnected through the process - broadly speaking all modules are connected this way. A special case of connected controllers worth a mention are higher level controllers and optimisation toolkits, which are very hard to test and tune without a realistic process response. The test unit could include the whole process model or a subprocess, which is controlled by the part of automation system under testing. In the automation system the modules are enabled whose integral functionality is put under test. A simulator provides realistic measurements from the subprocess, and conflicts between the functions of the automation system can be found. Sequences are typically used for starting up and shutting down a part of the process. A sequence consists of steps, at which actions are executed. The sequence proceeds after the defined conditions have been met. Typically the actions and conditions are connected to many parts of the

process and to successfully test the sequence the process should respond realistically enough to the commands given by the sequence.

4.4.4 System and acceptance testing

The last two test phases of the V-model are here handled as one and called commonly system testing. In the system and acceptance tests the whole automation system is under evaluation.

In system testing the test unit includes the whole automation application and the process model of the plant. The total system is run checking the functionality defined in the functional specification and the requirements defined in the user requirements. Simulator assisted testing suits very well for this purpose. During these tests the realistic process response is very important. The complex cross-connections between the different process parts exist and the design errors causing problems in simulated transients can be pinpointed.

4.5 Tuning tool

4.5.1 Introduction

One interesting usage of simulation sequences is iterative simulation based multivariate control parameter tuning. It heavily utilises a predefined simulation sequence that is run several times using slightly varying parameters. The data collected during the execution of simulation sequences is then used to tune up the control parameters of the process.

Tuning the overall performance of process and related automation system can be a complicated task. The size of the challenge greatly depends on the level of complexity of the process. A modern paper making process is an example of a demanding and complex tuning task. A paper plant may consist of hundreds of controllers. The net effect of controllers is hard to understand even for the most experienced process engineers. This makes the finding of optimal tuning hard or impossible using conventional tuning methods. [Halmevaara1, 2004]

There exist methods for tuning a multivariable control structure [Halmevaara2, 2004]. Many of those have roots in the area of single variable control tuning. This starting point is not the best possible as the main motivation is to enhance the overall performance of process, which is greatly affected by the interaction of control loops.

Iterative simulation based multivariate control parameter tuning method is an off-process tuning method that extensively exploits continuously increasing computation power. The following chapters briefly describe the method and working process enlightening the possibilities of the proposed technique.

4.5.2 Theory and methods

Multivariable dynamic system respect of the input (u) and output (y) parameters is represented in Figure 11. If the system is viewed from higher level of abstraction, its performance can be characterised using quality measures (q) that depend on the parameters (θ). [Halmevaara1, 2004]

Iterative regression tuning (IRT) consists of global iteration steps. Global iteration steps consist of multiple local iteration steps. Local iterations are executed using randomly varying parameters (θ). Between global iteration steps a local model is constructed between the parameters (θ) and targets (q). After the modelling the parameters are updated based on the gradient direction of the cost criterion, and the next global iteration is executed in the surroundings of this new "operating point". This process is then continued until the performance is good enough or no further enhancements can be achieved. [Halmevaara1, 2004]

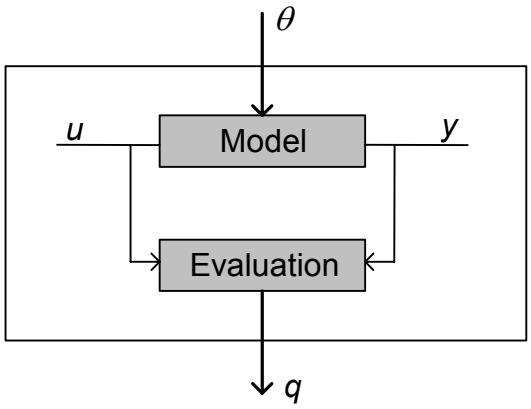

Figure 11. Relationship between parameters and target variables [Halmevaara1, 2004].

4.5.3 Applying the method

Tuning control parameters using the iterative simulation based multivariate control parameter tuning method is quite a complicated process. In [Halmevaara1, 2004] the method is tested using an environment that consists of a dynamic process simulator Apros [Apros 1988] and Matlab. The process was modelled using Apros and the data

connection between Apros and Matlab was established using quite primitive text file based system. Simulation sequences in each global iteration step were defined using text-based command queue files and between each global iteration step these files were modified by hand. Due to the complex nature of the working process and the lack of integrated working environment that would directly support the working process, it was quite time consuming to apply the method in practice. A custom made user interface that directly supports the working process and allows configuring and running tuning projects would lower the time and efforts needed. This would certainly raise the value of this method from the end user's perspective.

A more detailed description about the software architecture and design of tuning tool can be found in the technical report of the Testing Manager project [Mätäsniemi, 2005].

Figure 12 summarises the required steps needed in the tuning process. The procedure is divided into two major phases: initialising and running. This is partly over-simplified view because during the tuning phase it often emerges situations that are best handled by modifying project definitions (initialisations) e.g. by changing tuning targets or varied parameters or even by enhancing the modelling accuracy in the simulation service.

The initialisation phase in fact begins by making the simulation model from the target process, if a suitable one is not available. After that, a simulation sequence is defined. The simulation sequence consists of events that are essential to capture the process phenomena under tuning. Defining a suitable sequence requires good process knowledge as well as understanding about the characteristics of simulation model.

Parameters that are varied during the simulation sequence are defined in the next phase. Each parameter is varied randomly within a specified range during the tuning. During the tuning phase, it is the responsibility of the tuning tool to appropriately update selected variables.

Target functions are defined using predefined set of simple functions like minimum, maximum, variance and different integral functions. Each target function also consists of starting and stopping conditions that define when the evaluation of each function begins and stops.

After defining simulation sequences, varied parameters and target functions the initialisation phase is completed by specifying the lengths of local and global iterations. Iteration control can be done automatically if the tuning tool supports statistical methods for the decision making. In practice, engineering experience is usually used to specify an initial number of local iterations, which can later on be reduced. The decision to end global iterations is based on the graphical information rather than the statistical analysis.

Exploiting statistical analysis could, however, provide means for automating the whole tuning procedure.

The actual tuning process basically consists of a set of global iterations. Each global iteration consists of a set of (usually several dozens up to a hundred) local iterations. Updating selected parameters begins the execution of local iterations. After that the predefined simulation sequence is run. Finally the quality measures are calculated. This procedure is then repeated until the stopping condition of the local iterations is met.

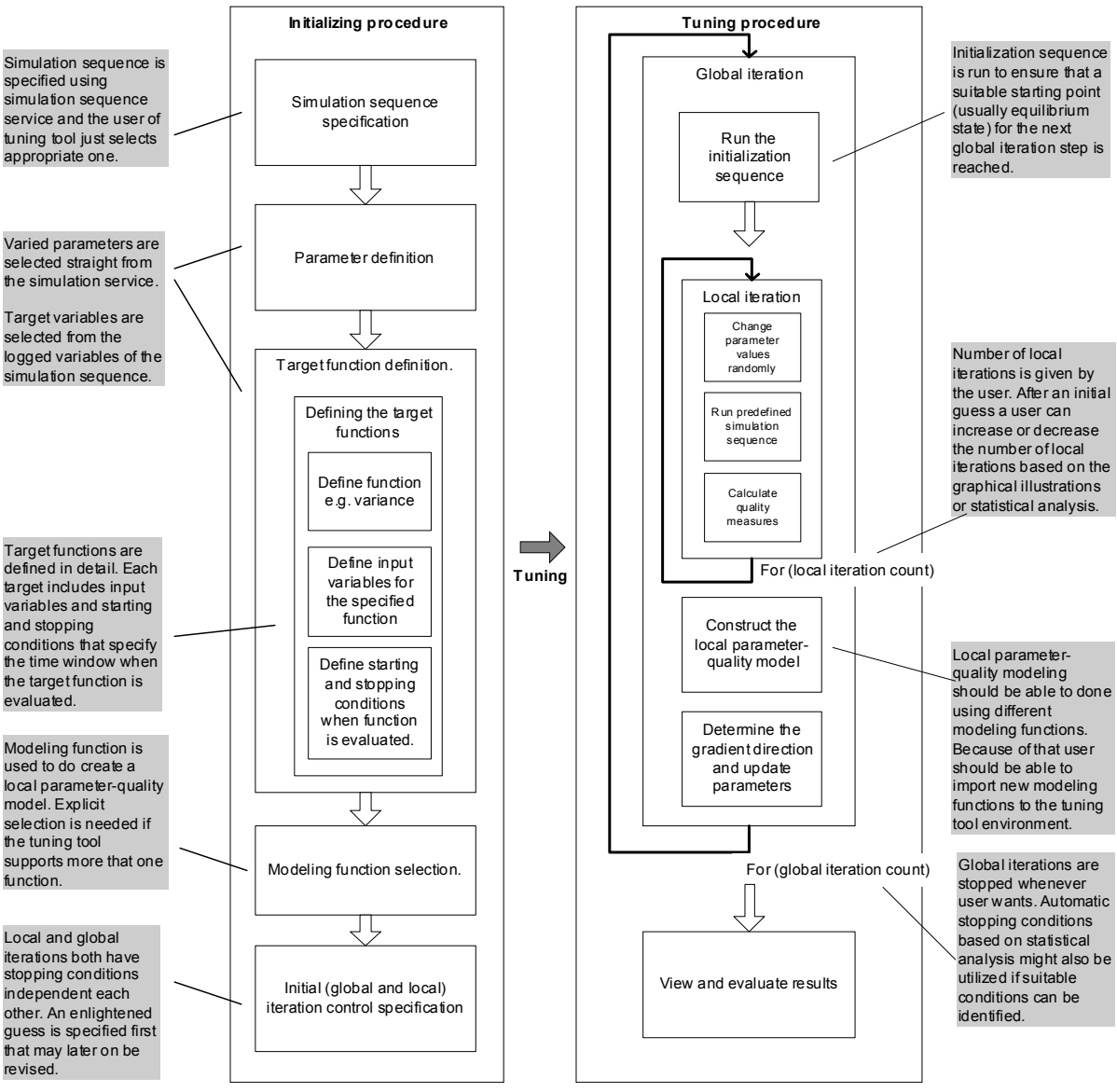

Figure 12. Tuning process consists of initialising the project by doing appropriate definitions. After the initialisation the actual tuning process begins. Modified from the [Halmevaara2, 2004].

Between global iterations, the local parameter-quality model is constructed and the tuning tool will calculate the gradient direction of the cost criterion. The next global iteration cycle is then executed around this new “operating point”.

In addition to be able to initialise and run tuning projects, a tuning tool should provide a rich set of ways to graphically examine results and sub-results. The simulation aided tuning process requires human interaction and cannot be, at least yet, seen as highly automated batch-style process.

5. Example cases and tools

5.1 Introduction

This chapter is mainly based on the investigations presented in chapters 5 and 6 in [Ahonen 2004]. The main points presented are collected in this chapter and some observations have been added. In the case the simulation model of the nuclear power plant (NPP) is used and the automation system controlling and protecting the steam generators is tested. The main purpose of the steam generator (SG) is to transfer heat from the primary side of the NPP to the secondary side. The water heated in the primary side of the NPP is fed into the heat exchanger pipes of the steam generator. The heated water makes the water boil in the secondary side of the SG. The generated steam goes forward towards the turbine. To compensate the steam flow, new water is fed to the SG through feed water lines. Schema about the functionality of the SG is given in the Figure 13. The case is quite small and does not reveal all benefits of simulation assisted automation testing [Ahonen 2004].

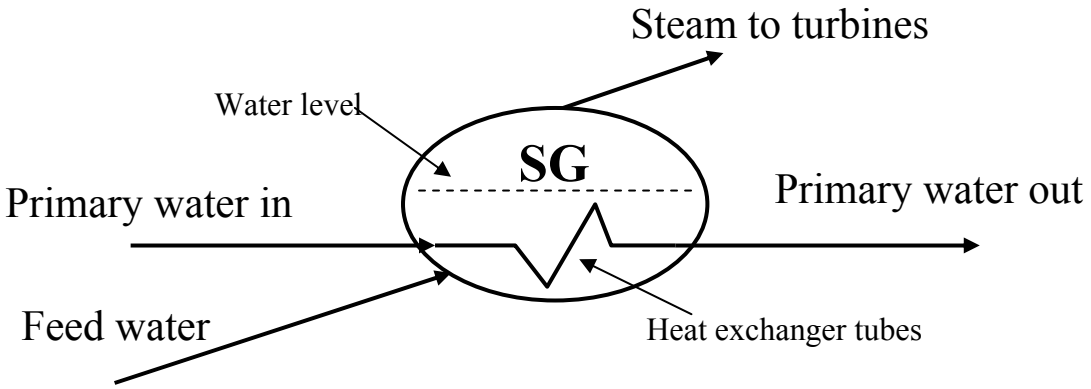

Figure 13. Steam generator.

Simulation model has been made using the Apros simulation tool. The automation system is the virtual version of metsoDNA and communication based on OPC standard is used without synchronisation. A similar configuration has been used also in the Narva case presented in Chapter 2.3.4.

5.2 Requirements

At first the user requirements were defined for the overall system. Profound understanding of the process and installed devices were necessary before the requirements for the automation system could be stated. The user requirements for the

SG were defined using the process related facts. SG has a task, which it should do in the plant and the safety of the personnel, SG or other process components should not be compromised.

The main task is that the *SG should produce dry saturated steam from the heat energy produced by the reactor*. This goal should be reached in all operational states and transients and the automation system should assist reaching this goal. In practice this means that in all cases there should be water in the steam generator that can be transformed to the steam. To protect the process equipment some further requirements must be specified. To prevent wet steam entering the turbine the level should not exceed the specified limit, and to prevent heat stress of the heat exchanger pipes of the SG they should not be exposed to steam. Furthermore there are other restrictions, which have to be taken into account when designing the automation system, e.g. the liquid level measurement of the SG is relatively inaccurate and noisy.

The test plan in this case should include a test, which tests that the user requirements are met. In this case it was tested that the liquid level stays between predefined limits in several transients like feed water pump trip, PCP trip and turbine trip. The test cases could also be automated by defining the test cases presented in the Chapter 4.3. In this case the sequences were carried out manually.

5.3 Design

After the user requirements have been stated the actual design of the automation system can be planned. Traditionally the automation system of the SG has two main parts:

- Liquid level controller to keep the liquid level as close as possible to the set point - normally the level is controlled using the steam and feed water flows and the liquid level measurement
- Protection and alarm logic, which automatically alarms the operator and protects the hardware if the liquid level is too high or low.

This design has been found already acceptable so in this case there was no need to test it. The presented example did not include any upper level controllers, interlockings or sequences and that part of the test plan could be ignored.

5.4 Modules

After the needed functionality has been designed the next step is to plan the actual automation functions and modules. In this case the following modules were planned and implemented:

- The normal liquid level controller module
- The protection close module, which closes the feed water shut off valves at high liquid level
- The plant protection module, which starts the emergency feed water pumps and opens the valves on the delivery side of the pump at low liquid level
- The isolation logic module of the steam generators at high liquid level.

The test plan includes the testing of these modules. The planned test cases are described later in chapter Testing.

5.5 Testing

5.5.1 Test unit

In all presented cases the test unit used is practically the same. The simulator model includes the nuclear power plant process and the automation system includes the controls and protections related to the steam generator. The communication has been defined so that all measurements required by the automation system are transferred from the simulator and correspondingly all the controls are transferred from the automation system to process model.

5.5.2 Module testing

Several module tests were carried out both for the analog automation part (controller) and the binary automation part (protection and isolation logic). The results were checked using manual inspection of the simulation results. All executed test cases were fixed, which means that there was only a one test run configuration for each test case.

Following tests were carried out for the liquid level controller module:

- Manual control of the valve. The valve is switched into a manual mode and opened and closed. The effect on the output of the controller is verified.

- Correct directation. The controller is in automatic mode. The liquid level, steam flow and feed water flow measurements are manipulated and the output of the controller is monitored, e.g. when the steam flow decreases the output should decrease as control valve is closing.
- Open loop step response. The controller parameters were tuned using the step response test before the implementation. This test is used to verify that the controller has correct controller parameters.
- Closed loop step response. The set point of the liquid level is changed downwards and upwards and the corresponding behaviour of the liquid level is checked.
- Anti-windup mechanism of the controller. A large error signal is added to the controller input. The output of the controller will reach the upper or lower limit. After this the sign of the error signal is changed and the output of the controller should follow properly.

The testing of protection close of the feed water valve was implemented using two tests where the liquid level measurement was manipulated. At first the measurement value was increased above the limit and after some time decreased below the limit. The test was interpreted to be successful if the shut off valve closed when the limit was reached and if the shut off valve was again controllable, when the liquid level decreased below the limit. The test run configuration corresponding this case is given in the Figure 14. Similar cases were executed also in other binary automation test cases.

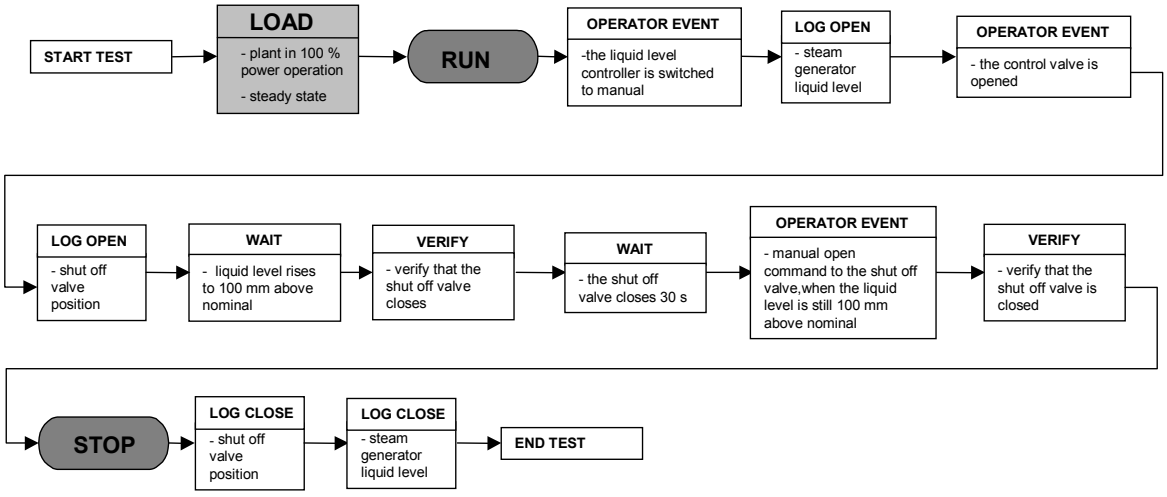

Figure 14. Test run to verify protection close command from high liquid level of SG.

5.5.3 System testing

In this case the whole automation system was rather small. Therefore the last three testing phases (integration, system and acceptance) could be coupled together. The phase is called system testing. During the system tests several transients were executed and the overall functionality of the whole automation system was under inspection. The transients included e.g. a turbine trip and break down of the feed water and recirculation pumps. The test run, which was used to inspect the functionality during the stoppage of the feed water pump is presented in Figure 15.

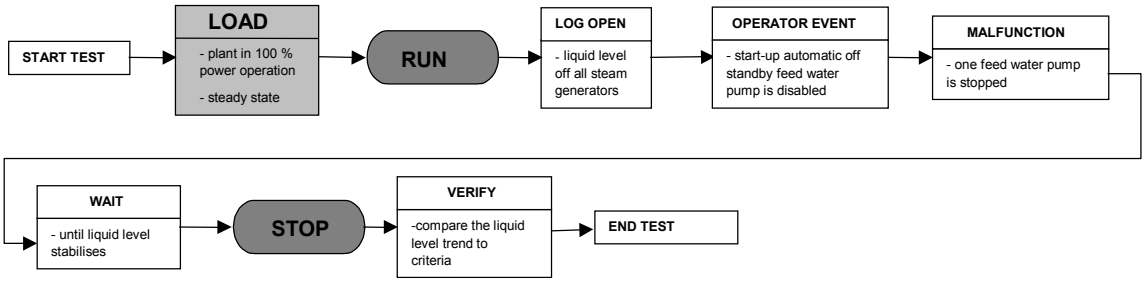

Figure 15. Test run to verify the automation system functionality, when start-up of the standby feedwater pump is prevented and the actual feed water pump is stopped.

The main acceptance criteria for all cases are related to the liquid level of the SG. Primarily the level should remain between the protection limits. In some test cases this cannot be reached and more accurate criteria were stated. E.g. a test case was executed where the liquid level in the SG was increased to the protection close limit of the feed water line. In this case the main purpose is to verify that the liquid level does not reach the turbine trip limit.

5.5.4 Observations

Here is a recap of the main points presented in [Ahonen 2004]. The specifying of the test cases directly based on the user requirements was found to be quite difficult. On the other hand specifying tests to test the functionality was found to easier task. The simulator and the automation system were driven to the state, which should initiate the functionality under inspection. The test was passed if the correct action occurred.

In practice the execution of the Test cases were somewhat problematic because of technical problems. Communication was unsynchronised and considerable communication delays were observed. This caused problems when testing and tuning the functionality of the controller. Furthermore this prevented the exact repeatability of the test cases.

Also some components within the simulator, meant to emulate the hardware between the automation system and process component, did not work correctly in all cases.

Module testing was found to be a straightforward and systematic task. Test run, which initiated the tested automation function aws executed, and the correct operation was verified. The test run was either successful or it failed.

System level tests were considered to have biggest benefits available. However in these cases the evaluation of the test results was found to be the biggest problem. The test engineer must have profound knowledge about acceptable or good enough behaviour to make the decision whether the test run was successful or not.

6. Conclusions

The earlier the errors in the automation application can be found the less expensive their correction will be. Extensive testing during the design and implementation phases decreases the number of errors in the installed system and shortens the commissioning time. Furthermore the detected and corrected errors increase the safety and financial benefits of the plant.

Automation applications are large and tightly coupled to the operational environment (e.g. process) where they are installed. This makes the extensive testing of the application very challenging. The actual operational environment is available only after the installation and even after the installation some of the test scenarios may be too risky or expensive to perform using the actual environment. Especially the testing of the automation system behaviour during accident situations may not be possible using the actual process.

Several possibilities for using simulation during automation delivery project were discussed in this paper. Typical uses are evaluation of the process design, verification of the automation implementation, tuning of the control loops and training of operating personnel. Simulation has been considered a useful tool to improve the quality of the automation delivery as well as minimize the costs.

With the aid of process simulation the behaviour of the process can be emulated and the functionality, performance, reliability and usability of the automation system can be but under evaluation before the automation system is connected to the actual process. The recent developments of the hardware, software and communication standards have enabled the usage of simulation to test the automation application. The process-wide simulation models can be executed using inexpensive standard hardware. The standardisation of the communication has decreased the costs and effort needed to connect the simulator to the automation system. To further ease the task of the test engineer, dedicated testing tools are needed for requirements management, test case generation and execution and analysis of the test results.

Even if the recent development has enabled the testing there are still much room for improvement. Typical automation systems do not support all features required to be fully simulator friendly. The virtual automation system may require some tuning after installing it to the virtual environment, i.e. virtual and actual versions are not fully compatible. All automation system products do not support faster that real time execution, which would enable more tests in shorter time. Also the communication would require improvement: synchronous execution is required for repeatability of simulation runs, which is a significant benefit, or in many cases a necessity. Also the

initialisation of the whole simulation system to a new state is problematic at the moment. At the moment if new initial state is required the simulated process alongside with the automation system must be run like the real plant to the desired state, which is then saved. Obtaining a comprehensive set of initial states using this method is time-consuming task, and if changes in either the simulation model or the automation application are done the changes must be implemented to all initial states.

One big obstacle for the take-up of simulation assisted automation testing methods is the manpower required for developing an accurate simulation model. The simulation model has proven to be too expensive to be made for testing purposes alone. Other usage of the same simulation model is therefore required, e.g. operator training.

After the Testing Manager project, further research is required in the following topics:

- The relation between testing and requirement management was not in the original focus of the project. However, the research showed that this relation is a key issue for the automatic generation of simulation sequences as well as traceability. Besides it turned out that the user requirement management is an open question in automation industry.
- This research produced a good start for assessing the reliability of the automation system by using process simulation. This is a new idea, which needs to be integrated with other research e.g. the definition of operational profiles.
- The iterative multivariate tuning method can be generalised as a simulation assisted parameter optimisation method for different phases of the plant life cycle, which is a new research topic.

The research has opened following topics of product development:

- The tuning algorithm developed in the project gave very promising results indeed given that the computational power is still raising. The tuning tool whose prototype was built in the project has a good potential as a product.
- As the research on the requirement management and automation testing advances, tool integration and possibly new software tool development is required to aid in generating simulation sequences and tracing back to user requirements.
- The new generation web service based process simulation and automation infrastructure and the simulation sequence mechanisms must be further developed and existing products must be integrated to the infrastructure.
- Synchronisation features and high enough performance in communication must be implemented in existing products.

References

[Ahonen 2004] Ahonen, A. The process of selecting test runs in simulation aided automation testing. M.Sc. Thesis, Helsinki University of Technology, Espoo 2004. 97 p.

[Apros 1988] Juslin, K., Sivennoinen, E., Kurki, J. & Porkholm, K. APROS: An Advanced Process Simulator for Computer Aided Design and Analysis. 12th IMACS World congress on Scientific Computation, Paris, France, July 18–22, 1988.

[AUTOHJE 2005] End report of the VTT project called "Automaatiojärjestelmien ohjelmistotuotannon kehittäminen" to be published 2005 in Finnish.

[Bogo 2001] Bogo, A., Nunes, P., Hidalgo, G. & Herschmiller, D. Aracruz Uses a Dynamic Simulator for Control System Staging and Operator Training. Technical Paper, Ideas Simulation Inc.

[Chung 2002] Chung, J., Heejong, Y., Ingu, L., Junwon, P. & Yongseung, Y. Onboard testing of the Control System in the LNG Carrier Using Dynamic Simulator. Simulation, Vol. 78, Issue 2, February 2002, p. 114–121.

[Haapanen 2004] Haapanen, P., Helminen, A. & Pulkkinen, U. Quantitative reliability assessment in the safety case of computer-based automation systems. STUK-YTO-TR 202, STUK, Helsinki 2004. 36 p.

[Halmevaara1, 2004] Halmevaara, K. &. Hyötyniemi, H. (2004). Iterative Simulation Based Multivariate Control Parameter Tuning Method. Proceedings of the 5th EUROSIM Congress on Modeling and Simulation. Paris, France.

[Halmevaara2, 2004] Halmevaara, K. & Hyötyniemi, H. Process performance optimization using Iterative Regression Tuning. HUT, Control Engineering Laboratory, Report 139.

[IEEE Std 982.1 – 1988] IEEE. Std 982.1 – 1988 IEEE Standard dictionary of measures to produce reliable software. The Institute of Electrical and Electronics Engineers. New York, 1988. 36 p.

[IEEE: Std. 830-1998] IEEE: Std. 830-1998 Recommended Practice for Software Requirements Specifications, June 1998.

[ISO 9126] ISO 9126 Software engineering product quality.

[Lappalainen 2003] Lappalainen, J., Myller, T., Vehviläinen, O., Tuuri, S. & Juslin, K. Enhancing grade changes using dynamic simulation. TAPPI Journal, Online Exclusive, Vol. 2 (2003) 12. http://www.tappi.org/

[Lilja 2003] Lilja, R., Ollikainen, T. & Laakso, P. Self-access Studying Environment for Control Engineering Education. The 6th IFAC Symposium on Advances in Control Education (ACE 2003), 16–18 June 2003, Oulu, Finland.

[Mätäsniemi 2005] Mätäsniemi, T., Peltoniemi, J. & Paljakka, M. Technical specification of testing manager environment. VTT Research report to be published 2005.

[OPC] Web page of OPC foundation. http://www.opcfoundation.org

[Paljakka 2003] Paljakka, M. & Kangas, P. Current Work Practices in Automation Testing. Espoo 2003, VTT, Research Report (Confidential). 9 p.

[POHA 2005] End report of the VTT project called "Uudet tekniikat teollisuuslaitosten poikkeustilanteiden hallinnassa" to be published 2005 in Finnish.

[Process Vision 2000] Dynamic Simulation in Control System Planning and Operator Training – Case Västerås. http://www.vtt.fi/tuo/63/apros/pdf/success/success - vasteras.pdf

[Rinta-Valkama 2000] Rinta-Valkama, J., Välisuo, M., Karhela, T., Laakso, P. & Paljakka, M. Simulation Aided Automation Testing. Proceedings of the TAPPI 1999 Engineering/Process & Product Quality Conference, Pergamon 2000. P. 277–280.

[SAS 2001] Tommila, T. et al. Laatu automaatiossa – parhaat käytännöt. Suomen automaatioseura ry, Helsinki 2001. 245 p. (In Finnish)

[Sommerville 2001] Sommerville, I. Software Engineering. Addison-Wesley, 6th Edition, 2001. 693 p.

[Stenberg 2003] Stenberg, A. Ohjelmiston testaus. SataSPIN koulutusmateriaali 2003.

[Yli-Petäys 2001] Yli-Petäys, J. Apros ohjelmiston hyödyntäminen voimalaitosten simuloinnissa. M.Sc.Thesis, University of Oulu, Espoo 2001. 68 p.

Published by

Series title, number and report code of publication

VTT Research Notes 2289
VTT–TIED–2289

Author(s)
Laakso, Pasi, Paljakka, Matti, Kangas, Petteri, Helminen, Atte, Peltoniemi, Jyrki & Ollikainen, Toni

Title

Methods of simulation-assisted automation testing

Abstract

The recent developments of the hardware, software and communication standards have enabled the usage of simulation to test the automation application. The process-wide simulation models can be executed using inexpensive standard hardware. The standardisation of the communication has decreased the costs and effort needed to connect the simulator to the automation system. To further ease the task of the test engineer, working methods and dedicated tools are needed for requirements management, test case generation and execution and analysis of the test results.

This document presents a working methods for simulation assisted automation testing. Furthermore the possibilities of the simulation during the automation system life cycle and requirements for the tools supporting the working methods are discussed.

Keywords
process simulation, automation systems, testing, apros, specifications, system design, implementation, installation, commissioning, validation

Activity unit
VTT Industrial Systems, Tekniikantie 12, P.O.Box 1301, FI–02044 VTT, Finland

ISBN	Project number
951–38–6542–8 (soft back ed.) 951–38–6543–6 (URL: http://www.vtt.fi/inf/pdf/)	G5SU01004

Date	Language	Pages	Price
March 2005	English	59 p.	B

Name of project	Commissioned by
Testing manager	National Technology Agency (Tekes), Fortum Nuclear Services, Metso Automation

Series title and ISSN	Sold by
VTT Tiedotteita – Research Notes 1235–0605 (soft back edition) 1455–0865 (URL: http://www.vtt.fi/inf/pdf/)	VTT Information Service P.O.Box 2000, FI–02044 VTT, Finland Phone internat. +358 20 722 4404 Fax +358 20 722 4374

VTT TIEDOTTEITA – RESEARCH NOTES

VTT TUOTTEET JA TUOTANTO – VTT INDUSTRIELLA SYSTEM –
VTT INDUSTRIAL SYSTEMS

2235 Lehto, Taru & Murtonen, Mervi. Toiminnan kehittämisen vaikutukset ja päätöksenteko. PRIMA-työkalupakki kehittämistoimenpiteiden valintaan ja suunnitteluun. 2004. 52 s. + liitt. 35 s.

2240 Jarimo, Toni. Innovation Incentives in Enterprise Networks. A Game Theoretic Approach. 2004. 63 p. + app. 3 p.

2243 Ventä, Olli. Älykkäät palvelut -teknologiatiekartta. 2004. 71 s. + liitt. 11 s.

2250 Sippola, Merja, Brander, Timo, Calonius, Kim, Kantola, Lauri, Karjalainen, Jukka-Pekka, Kortelainen, Juha, Lehtonen, Mikko, Söderström, Patrik, Timperi, Antti & Vessonen, Ismo. Funktionaalisten materiaalien mahdollisuudet lujitemuovisessa toimirakenteessa. 2004. 216 s.

2251 Riikonen, Heli, Valkokari, Katri & Kulmala, Harri I. Palkitseminen kilpailukyvyn parantajana. Tuotantopalkkauksen kehittämismenetelmät vaatetusalalla. 2004. 67 s.

2254 Nuutinen, Maaria. Etäasiantuntijapalvelun haasteet. Työn toiminta- ja osaamisvaatimusten mallintaminen. 2004. 31 s.

2257 Koivisto, Tapio, Lehto, Taru, Poikkimäki, Jyrki, Valkokari, Katri & Hyötyläinen, Raimo. Metallin ja koneenrakennuksen liiketoimintayhteisöt Pirkanmaalla. 2004. 33 s.

2263 Pöyhönen, Ilkka & Hukki, Kristiina. Riskitietoisen ohjelmiston vaatimusmäärittelyprosessin kehittäminen. 2004. 36 s. + liitt. 9 s.

2264 Malm, Timo & Kivipuro, Maarit. Turvallisuuteen liittyvät ohjausjärjestelmät konesovelluksissa. Esimerkkejä. 2004. 90 s. + liitt. 4 s.

2265 Alanen, Jarmo, Hietikko, Marita & Malm, Timo. Safety of Digital Communications in Machines. 2004. 93 p. + app. 1 p.

2269 Mikkola, Markku, Ilomäki, Sanna-Kaisa & Salkari, Iiro. Uutta liiketoimintaa osaamista yhdistämällä. 2004. 65 s.

2271 Häkkinen, Kai. Alihankintayhteistyö konepajateollisuudessa ja sen laadun arviointia. 2004. 64 s. + liitt. 17 s.

2277 Kondelin, Kalle, Karhela, Tommi & Laakso, Pasi. Service framework specification for process plant lifecycle. 2004. 123 p.

2283 Lemström, Bettina, Holttinen, Hannele & Jussila, Matti. Hajautettujen tuotantolaitosten tiedonsiirtotarpeet ja -valmiudet. 2005. 62 s. + liitt. 10 s.

2284 Valkonen, Janne, Tommila, Teemu, Jaakkola, Lauri, Wahlström, Björn, Koponen, Pekka, Kärkkäinen, Seppo, Kumpulainen, Lauri, Saari, Pekka, Keskinen, Simo, Saaristo, Hannu & Lehtonen, Matti. Paikallisten energiaresurssien hallinta hajautetussa energiajärjestelmässä. 2005. 87 s. + liitt. 58 s.

2287 Wahlström, Björn, Kettunen, Jari, Reiman, Teemu, Wilpert, Bernhard, Maimer, Hans, Jung, Juliane, Cox, Sue, Jones, Bethan, Sola, Rosario, Prieto, José M., Martinez Arias, Rosario & Rollenhagen, Carl. LearnSafe. Learning organisations for nuclear safety. 2005. 58 p. + app. 7 p.

2289 Laakso, Pasi, Paljakka, Matti, Kangas, Petteri, Helminen, Atte, Peltoniemi, Jyrki & Ollikainen, Toni. Methods of simulation-assisted automation testing. 2005. 59 p.

Tätä julkaisua myy	Denna publikation säljs av	This publication is available from
VTT TIETOPALVELU PL 2000 02044 VTT Puh. 020 722 4404 Faksi 020 722 4374	VTT INFORMATIONSTJÄNST PB 2000 02044 VTT Tel. 020 722 4404 Fax 020 722 4374	VTT INFORMATION SERVICE P.O.Box 2000 FI–02044 VTT, Finland Phone internat. + 358 20 722 4404 Fax + 358 20 7226 4374

ISBN 951–38–6542–8 (soft back ed.) ISBN 951–38–6543–6 (URL: http://www.vtt.fi/inf/pdf/)
ISSN 1235–0605 (soft back ed.) ISSN 1455–0865 (URL: http://www.vtt.fi/inf/pdf/)