TECHNISCHE UNIVERSITÄT
ILMENAU

50. Internationales Wissenschaftliches Kolloquium

September, 19-23, 2005

Maschinenbau von Makro bis Nano / Mechanical Engineering from Macro to Nano

Proceedings

Fakultät für Maschinenbau /
Faculty of Mechanical Engineering

Impressum

Herausgeber:	Der Rektor der Technischen Universität Ilmenau Univ.-Prof. Dr. rer. nat. habil. Peter Scharff
Redaktion:	Referat Marketing und Studentische Angelegenheiten Andrea Schneider Fakultät für Maschinenbau Univ.-Prof. Dr.-Ing. habil. Peter Kurtz, Univ.-Prof. Dipl.-Ing. Dr. med. (habil.) Hartmut Witte, Univ.-Prof. Dr.-Ing. habil. Gerhard Linß, Dr.-Ing. Beate Schlütter, Dipl.-Biol. Danja Voges, Dipl.-Ing. Jörg Mämpel, Dipl.-Ing. Susanne Töpfer, Dipl.-Ing. Silke Stauche
Redaktionsschluss: (CD-Rom-Ausgabe)	31. August 2005
Technische Realisierung: (CD-Rom-Ausgabe)	Institut für Medientechnik an der TU Ilmenau Dipl.-Ing. Christian Weigel Dipl.-Ing. Helge Drumm Dipl.-Ing. Marco Albrecht
Technische Realisierung: (Online-Ausgabe)	Universitätsbibliothek Ilmenau ilmedia Postfach 10 05 65 98684 Ilmenau
Verlag:	isle Verlag ISLE, Betriebsstätte des ISLE e.V. Werner-von-Siemens-Str. 16 98693 Ilmenau

ISBN (Druckausgabe):	3-932633-98-9	(978-3-932633-98-0)
ISBN (CD-Rom-Ausgabe):	3-932633-99-7	(978-3-932633-99-7)

Startseite / Index:
http://www.db-thueringen.de/servlets/DocumentServlet?id=15745

50. Internationales Wissenschaftliches Kolloquium
Technische Universität Ilmenau
19.-23. September 2005

George Nikolov / Boyanka Nikolova / Marin Marinov / Volker Zerbe

Design of Virtual Laboratory Workbench Using Unified Modeling Language

ABSTRACT

Modeling is proven well-accepted engineering technique. Engineers study models to assess the impact of environmental forces and anticipate the behavior of actual structures. The Unified Modeling Language (UML) has become the standard for documentation and high-level design. There are lacks of any UML guidance, manual or application notes that would be useful for creating laboratory practices. In this paper the principal approach of design of virtual laboratory workbench using UML is presented. The software package LabVIEW is chosen to create virtual instruments (VIs) that support the laboratory practice. To illustrate applicability of presented approach the application layer of the magnetic hysteresis is created. The design and implementation of this virtual workbench are based on concept of virtual instrumentation and UML modeling.

INTRODUCTION

The technological innovations behind the computers industry have transformed markets, business processes, education and many other human activities over the last years. The computer has transformed measurement and industrial automation applications from loosely coupled, often incompatible, stand-alone instruments to tightly integrated, high-performance, networked test and measurement and automation solutions. Recent developments and applications, specifically the computer-based applications, have shown that many pure lecture-based engineering courses and conventional experiments (which are heavily dependant upon specialized instruments) can be updated and integrated with custom-written virtual instrumentation (VI), multifunction data acquisition systems (DAQ) and can be delivered by computers [5, 6]. In addition to this, the courses and experiments can be delivered remotely without having multiple copies of the experimental setups. Additional scientific visualizations and advanced analysis can also be added in the form of virtual instruments with minimal cost, which is limited or not possible in the conventional laboratory practice. Moreover, the virtual instrumentation approach is open to further improvements and developments, which may increase the student participation and enthusiasm while providing ideal delivery environment.

A considerable portion of virtual instrumentation takes the software. Software transforms the PC and the DAQ hardware into a complete data acquisition, analysis, and display system. The increasing sophistication of DAQ hardware, computers, and software continues to emphasize the

importance and value of good software. A most important aspect of the process of creating good software takes up modeling. Models are used to visualize the desired structure and the behavior and architecture of designed virtual system. By way of modeling the developers can:

- Visualize the system;
- Specify the structure or behavior of a system;
- Create a template of how the system should be constructed;
- Document the decisions that have made.

The Unified Modeling Language (UML) has become the standard for documentation and high-level design of modern software [1, 2, 3]. The UML is an evolutionary general-purpose, tool-supported, standardized modeling language for specifying, visualizing, constructing, and documenting the artifacts of a system intensive process. It is broadly applicable to different types of systems, domains, methods, and processes. It enables and promotes a use-case-driven, architecture-centric, iterative, and incremental process that is object oriented and component based, fundamentally supporting industry and educational engineering best practices.

It is obvious that by incorporating the mentioned design and modeling practices, as software development tools will avoid unnecessary application redesign, increase VI reuse and minimize maintenance costs. Unfortunately there are lacks of any guidance, manual or application notes that would be useful for such initiative. To make up for this deficiency in this paper the principal approach of design virtual instruments using the benefits of UML is presented. Because of its overall versatility as an engineering tool, the software package LabVIEW is chosen to create VI. LabVIEW is a graphical development tool that allows rapid automation of instrumentation systems. Many useful functions can be incorporated with the LabVIEW programs to perform very useful tasks in a laboratory virtual instrumentation system design.

PRINCIPAL APPROACH FOR VIRTUAL WORKBENCH DESIGN BY UML

Following the best practice guides [2, 3] the design and development process can be divided in four main phases:

Phase 1 - Requirements analysis

Phase 2 – Design of virtual workbench

Phase 3 - Software coding, and

Phase 4 - Verification and validation

Phase 1 Requirement analysis

The first step is the virtual laboratory workbench analysis, and the input to the analysis is the specification of the requirements. In an object oriented and UML approach the requirements are

identified with help of identifying of cases of use of the system. This is done by UML use case diagrams. The main goal of this part is to identify the most characteristic use cases, and the actors (i.e. people or other types of "users" of the system). In figure 1, a UML use case diagram shows examples of how the requirement analysis of lab workbench can be implemented. To the left in the figure can be identified an actor Student, which is a triggered actor.

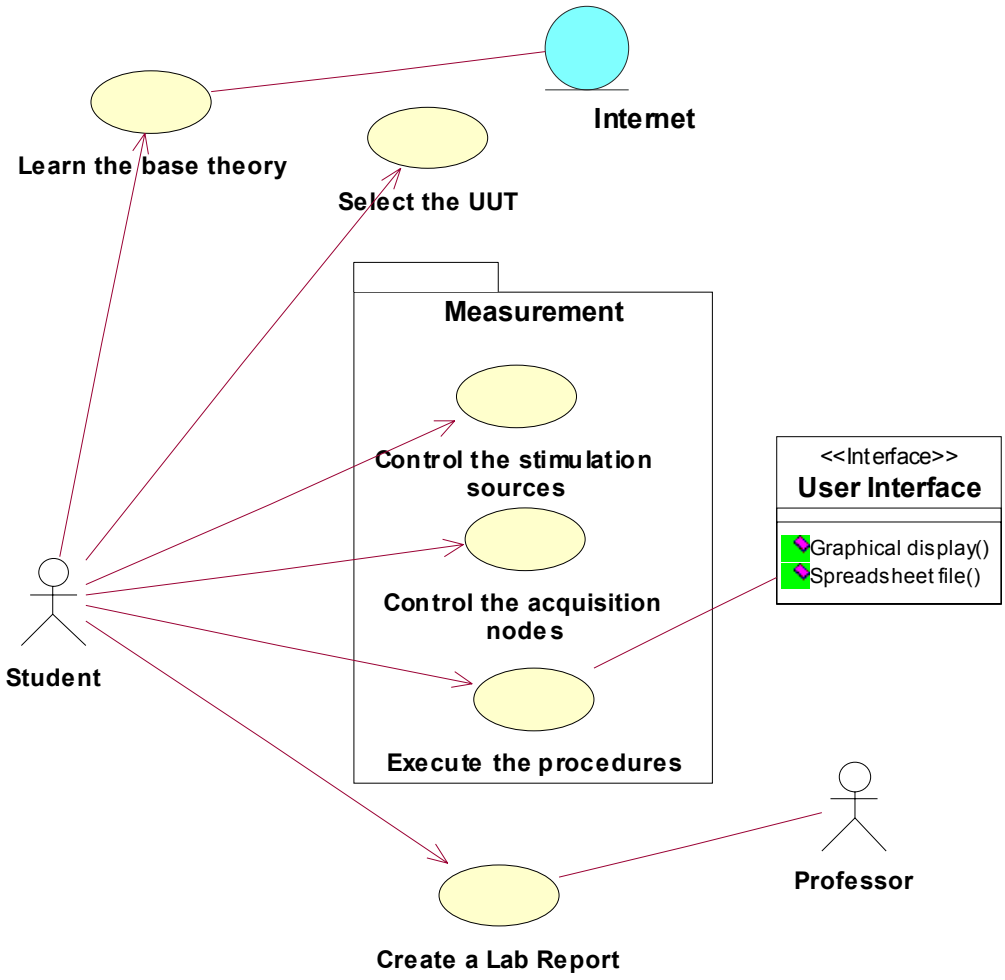

Fig. 1. Use case diagram of laboratory workbench

The process of analyzing the virtual laboratory workbench involves the following steps and considerations performed by Use Case Modeling

1. Identify and name the use case.

2. Draw a diagram indicating the use case, as well as its primary or triggering (student) and secondary (professor) actors – fig 1.

3. Describe the use case briefly. According figure 1 the table 1 is associated.

4. Describe the main flow of events in the use case (not presented in this paper). This description is used for activity diagram composition in next phase.

5. Define appropriate pre- and post-conditions for this flow of events (not presented in this paper).

Use-case name	Description
Learn the base theory	The actor Student read through the theory and lab procedure for this experiment from the textbook, datasheets or the actor (entity) Internet.
Select the UUT	The actor Student selects the appropriate unit under test (UUT) from the given opportunity.
Control the stimulation sources	The actor Student configures the DAQ system, initialized the analog output channels and buffers using appropriate VI's and set the stimulation sources.
Control the acquisition nodes	The actor Student set the resolution for analog input channels and buffers using appropriate VI's.
Execute the procedures	The actor Student executes the measurement procedures via actor User Interface (Front panel of the VI).
Create a Lab Report	The actor Student create a Lab Report, which is mainly, consists of descriptions of the experiments, measurement results and some conclusions. The actor Professor examine and rate student's Lab Report.

Table 1. The use case description

Phase 2 Virtual Workbench Design

Applying structural approach the design process can be divided in dynamic and architectural modeling. The UML's activity and state chart diagrams can describe the ***dynamic behavior*** of the developed system. In this paper the first step for virtual workbench design concerning the software portion is suggested to be ordering the flow of behavior. To explore the flow of laboratory measurement process, the activity diagram is most appropriate. In order to accomplish the modeling of flow of behavior the following steps is performed specifying workflows with Activity Diagrams

1. Break up the main success scenario into groups of interactions.
2. Each group of interactions becomes an interaction occurrence.
3. Connect the main success scenario interactions with control-flow lines to show the correct sequence.
4. When there is an alternative flow, break the control flow between interaction occurrences and insert a decision node or a fork node.
5. Use merge or join nodes to bring any alternative paths that pass through the interaction diagram back together (if necessary).

The UML activity diagrams for "Learn the base theory" scenario and package "Measurement" (complex scenario) are shown in fig. 2 and fig. 3 respectively.

The next dynamic behavior diagram is a state chart diagram. This diagram is used to express the states of the lab proceeding, or internal inside LabVIEW applications, and its transition from a state to a state triggered by a particular event. State chart diagrams are variations of finite-state machines, a standard method used in software design and programming. Figure 4 shows UML state chart diagram showing the states of the program when the laboratory measurement process proceeding. To create State Chart Diagram from scenarios the following steps are needed:

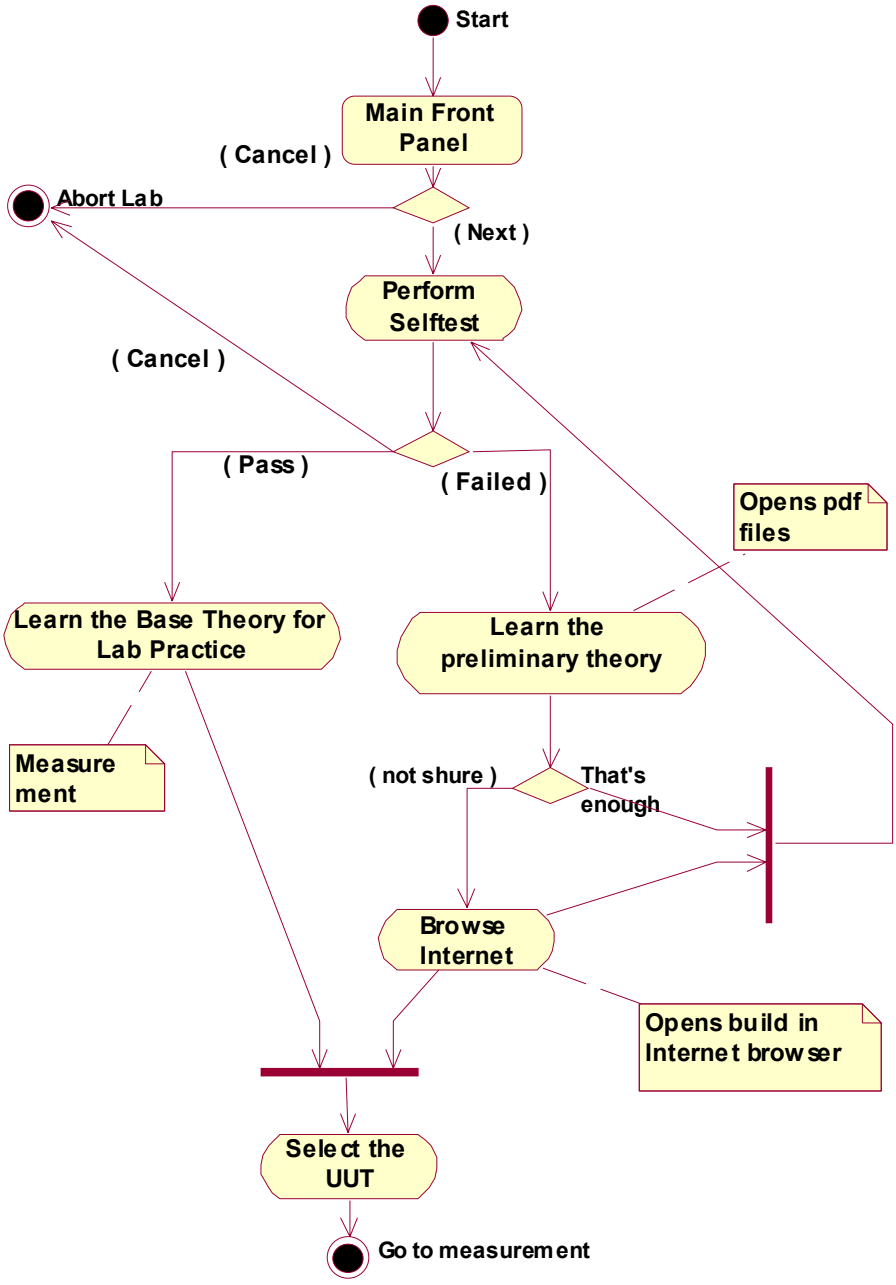

Fig. 2. Activity diagrams for "Learn the base theory" scenario

1.Get things started by adding a wait state.

2. Find incoming events.

3. Locate an event pair, which consists of an incoming event and the next incoming event.

4. Determine what the component is doing in response to the first incoming event.

5. Place a new state on the diagram

6. Draw a transition with the name of the first incoming event between the wait state and the new state just placed on the diagram.

7. Add transitions and states.

8. Consider the last transition.

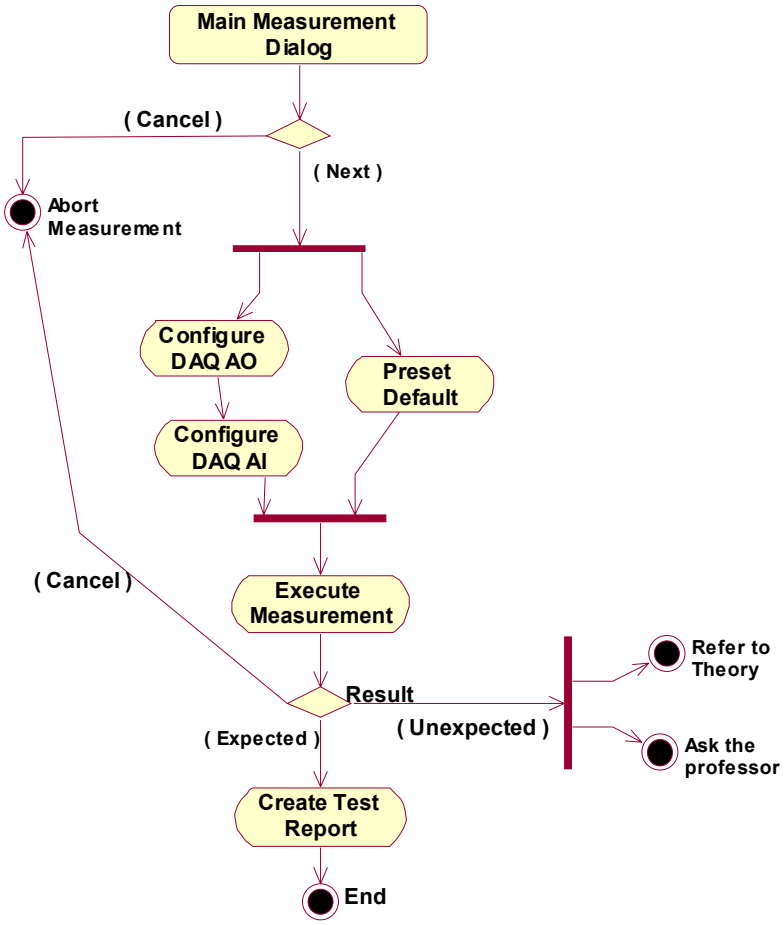

Fig. 3. Activity diagrams for "Measurement"

In the case of graphical programming presented in this paper, for ***architectural modeling*** most appropriate are component diagrams. Component diagrams describe the organization of physical software components, including source code, existing LabVIEW function, created applications and executables. Building modern virtual systems for maximum flexibility means designing with components [4]. A good design is distinguished with component that is a modular, self-sufficient, replaceable unit and works like a black box in the system.

The process for deploying the workbench's components involves the following steps and considerations:

1. Consider the design priorities
2. Review current laboratory practice.
3. Decompose the system (laboratory workbench) used to implement the lab. Take the system and break it up into smaller subsystems (so called subVI, sub-Virtual Instrument).
4. Define architecture. Once the subVIs are defined it is going to describe how those subVIs relate to each other and the hardware (in this case DAQ) that supports those subVI
5. Define the subVI's interfaces (connectors).
6. Select existing LabVIEW's components (build-in LabVIEW functions).

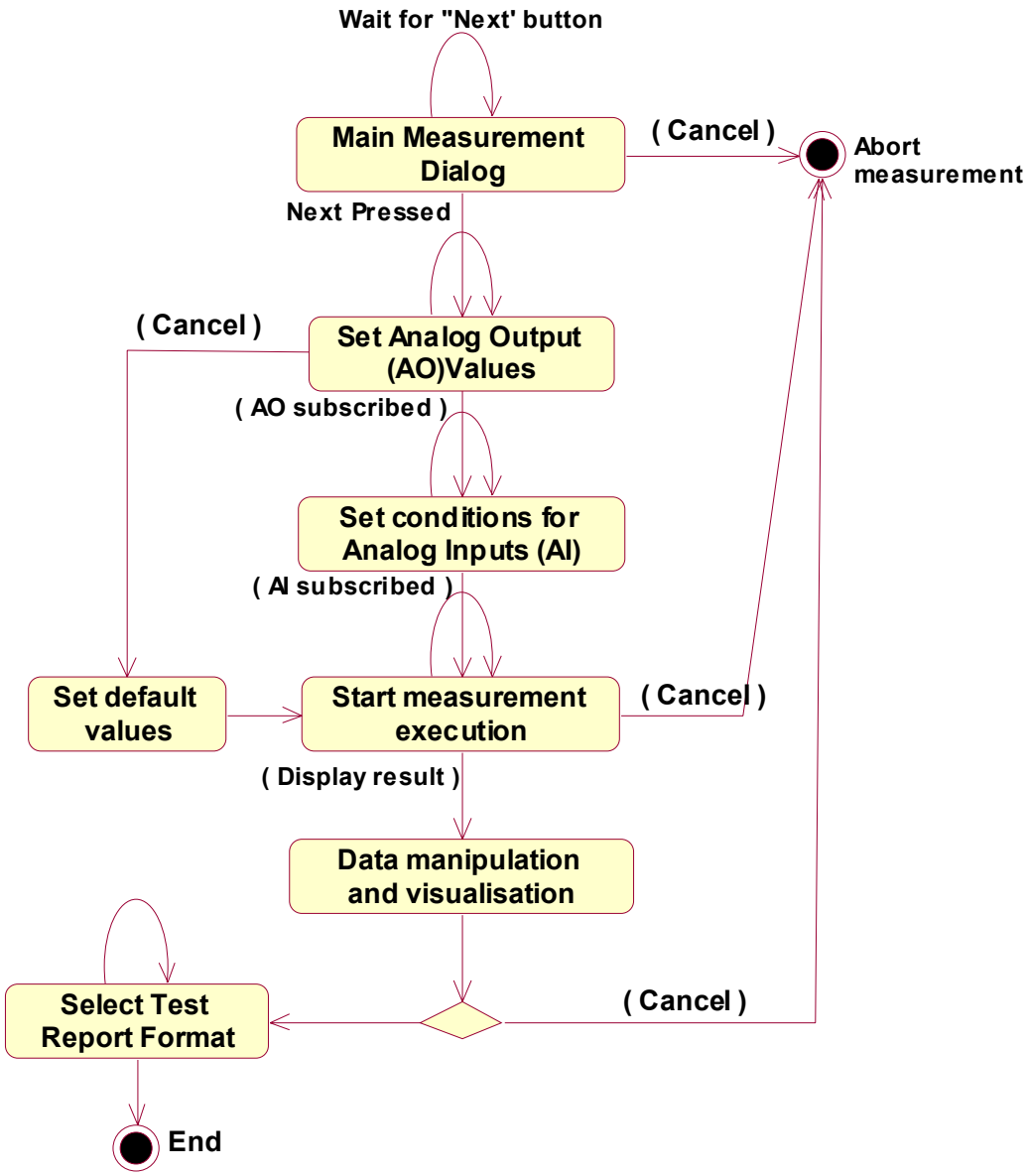

Fig. 4. State chart diagram for laboratory measurement process

7. Draw a UML component diagram describing the existing components and the component that should be created – fig. 5.

Phase 3 Software Coding

In order to create efficient programming code for laboratory practice a good practice is to use the design patterns. It is well known that design patterns represent techniques that have proved themselves useful time and time again. The state machine pattern is one of the most widely recognized and highly useful design patterns for LabVIEW. This pattern neatly implements any algorithm explicitly described by a state chart diagram. A state machine usually illustrates a moderately complex decision making algorithm, such as a investigation of UUD (Unit Under Test) or a process monitor. The standard LabVIEW state machine consists of a large “while loop”, a shift register to remember the current state, and a case structure that holds separate code to run for each

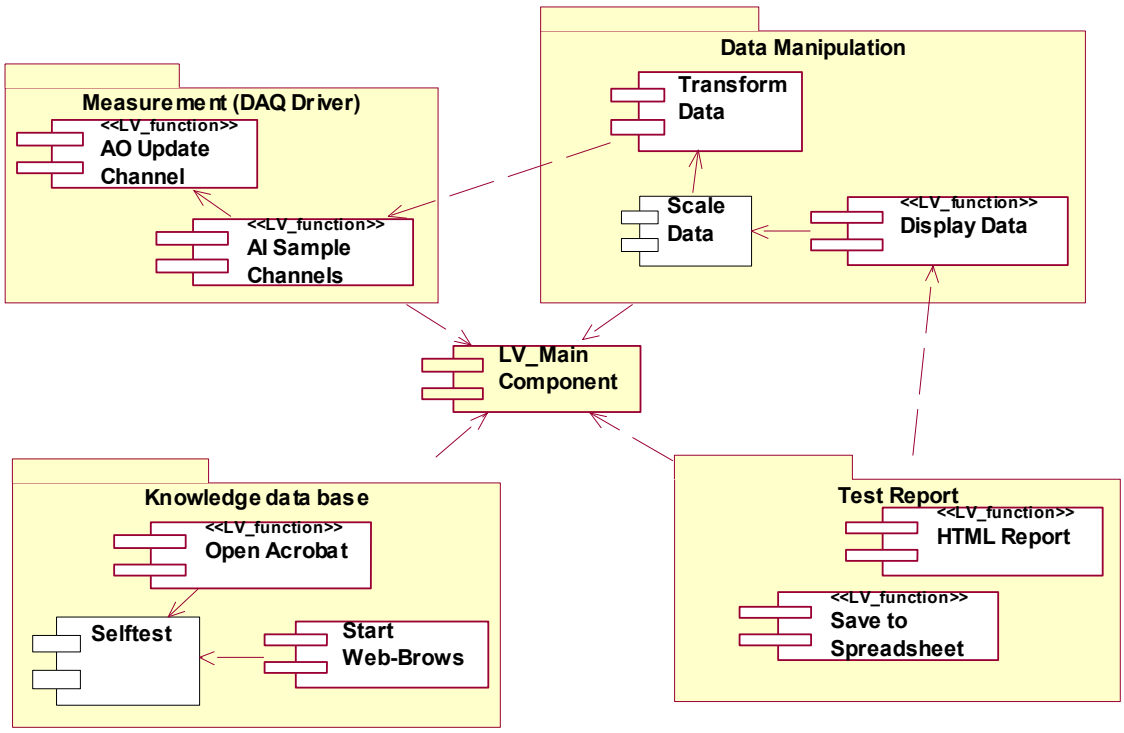

Fig. 5. Component diagram of physical software components

state. Of course, to complete the full LabVIEW program many other build-in functions and subVI are used, which are less or more described in appropriate documentation [4].

To create the programming code for virtual workbench the following steps must be fulfilled:

1. From application requirements (phase 1) choose the correct design patterns and data structures
2. Using state chart diagram (phase 2) recognize state machines and use them in application
3. Implement good programming style to create efficient VIs [4]
4. Stick to develop modular applications, which are easier to debug, maintain, and re-use
5. Document in time created VIs and subVIs
6. Use build in LabVIEW tools to evaluate inefficient VIs

Phase 4 Verification and validation

As example of verification phase and to illustrate benefits offered by suggested approach in the next topic the development of virtual magnetic hysteresis measurement system is observed.

DEVELOPMENT OF VIRTUAL MAGNETIC HYSTERESIS MEASUREMENT WORKBENCH USING UML

Passing over the main steps of suggested approach virtual laboratory workbench for magnetic hysteresis measurement is created. The one of the state machines that is created following the UML consideration (fig. 4) is shown in fig. 6. This programming code is responsible to ensure the correct sequence of measurement process.

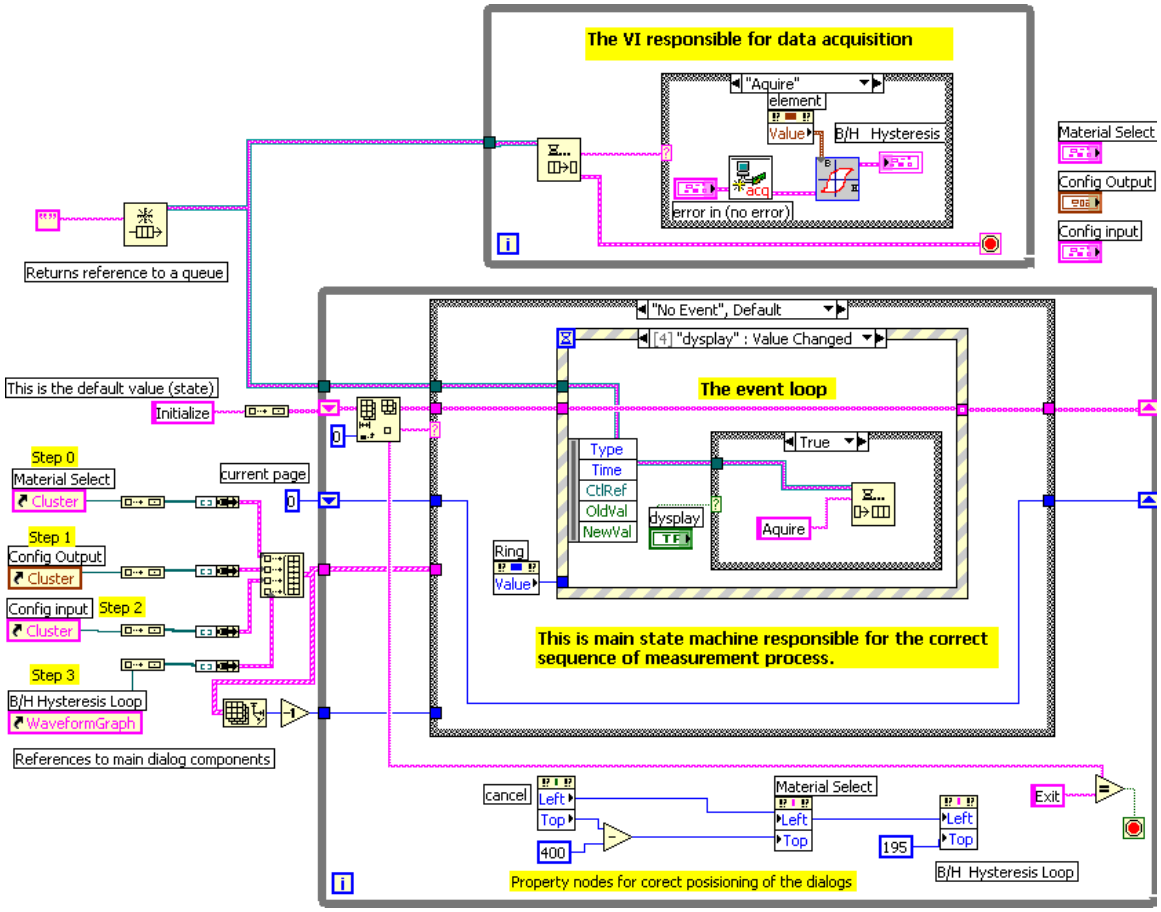

Fig. 6. The programming code of measurement process

Another design pattern that can be recognized due to UML is event loop. This powerful and efficient programming method is applied in software code and can be seen also in fig. 6. The event loop is used for handling user interaction with a LabVIEW program.

As can be seen in the figure, many other build-in LabVIEW functions and subVI are used, in order to complete the software application. This software components less or more can be recognized in the presented UML component diagram (fig. 5) .

In order to illustrate some of benefits offered by virtual instrumentation the user interface (front panels) of virtual measurement system is also appended. This user interface corresponds of UML use case modeling in phase 1 shown in fig.1 and source code (block diagram) of fig. 6.

The first use case “Learn the base theory” is represented by example shown in fig 7. In the right of figure is shown the opportunity to investigate the experiment’s details via Internet without leave the working environment. The next use case (Select the UUT) is depicted by selection of ferrite material for investigation. It is shown in the fig. 7. As can be seen the great representational possibility of LabVIEW focus attention of the user in the base objective of experiment.

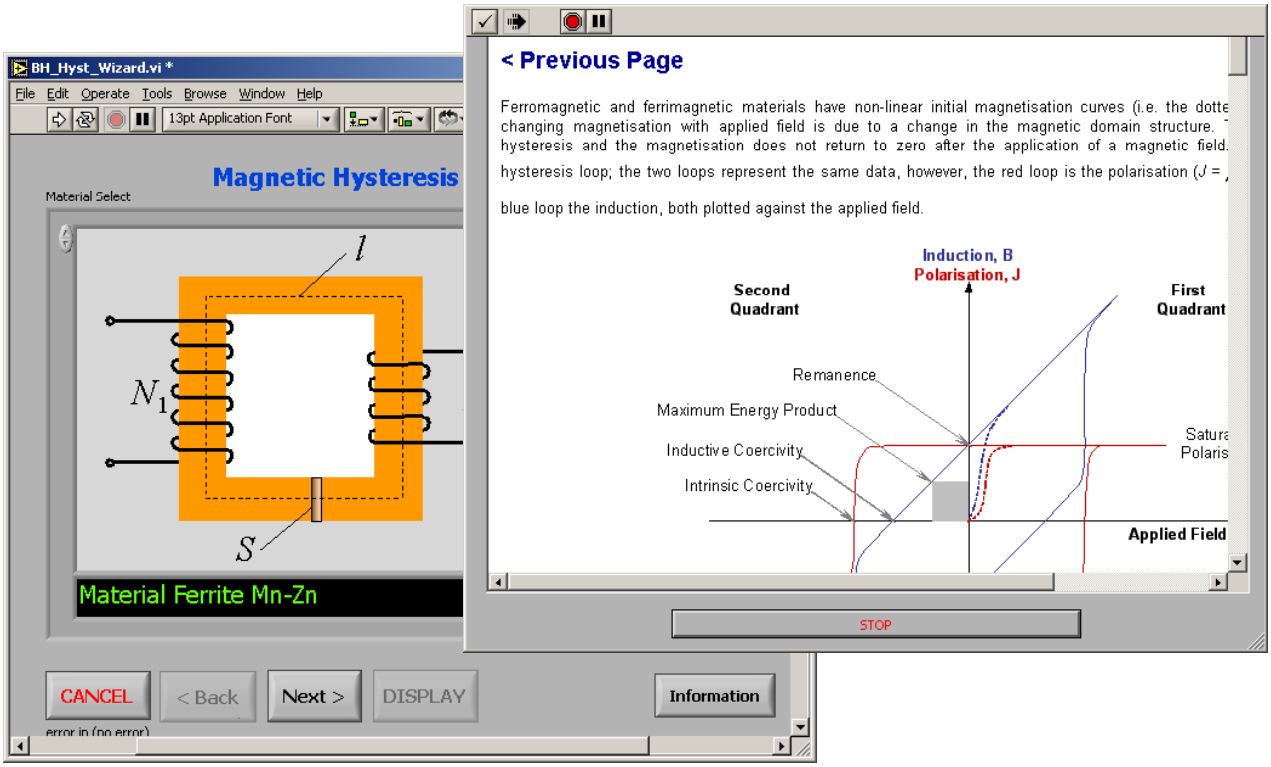

Fig. 7. Front panels of "Learn the base theory" and "Select the UUT" use cases

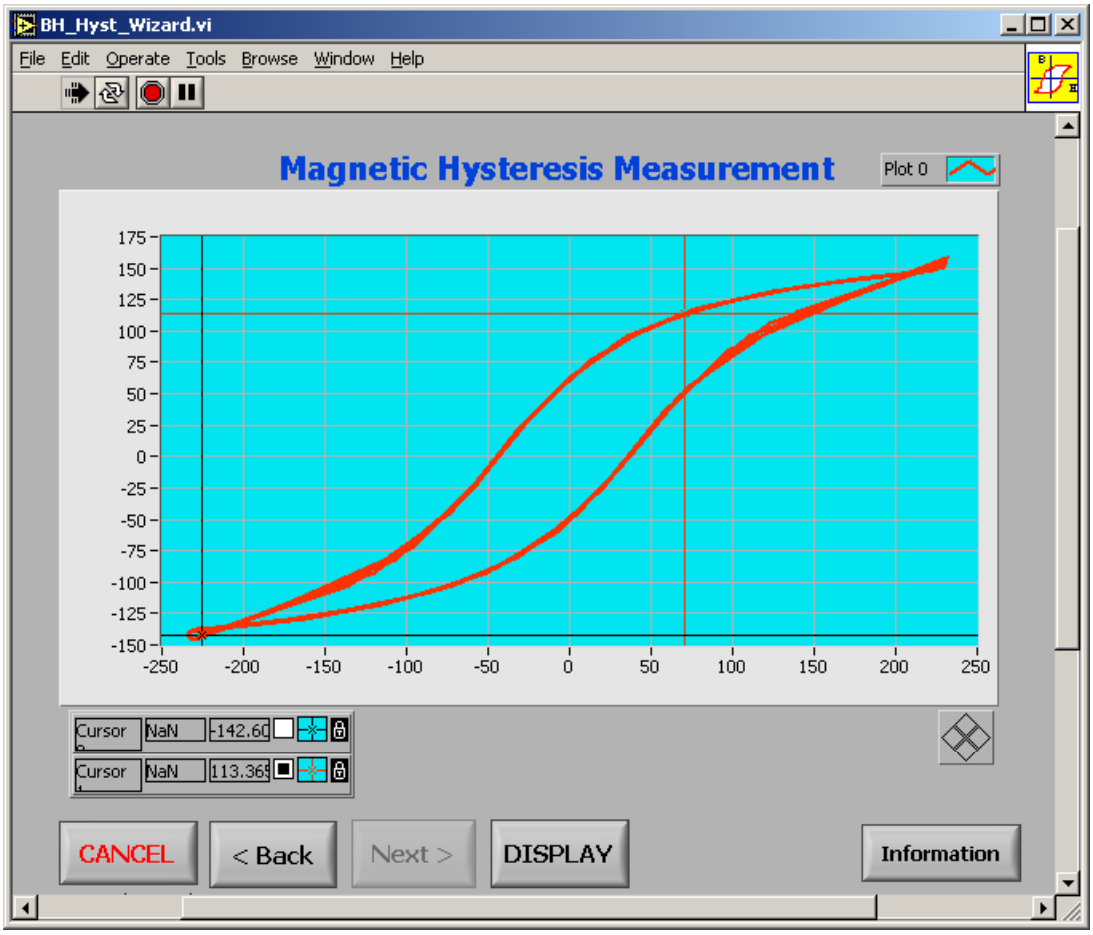

Fig. 8. Measurement results

In the next figure 8, the successful completion of the measurement procedures is shown. The user can observe the results and turn of profit the build-in markers for desired magnetic parameter extraction.

CONCLUSION

The complexity of modern virtual measurement systems is increasing. Laboratory practice that involve such systems are becoming more and more popular. To manage these new challenges, the whole software development process has to be improved. One very important aspect supporting this is virtual system modeling.

In the presented paper the UML based approach for developing virtual laboratory workbenches is considered. The proposed method is explained step by step in its practical aspect. The presented approach can be used for various laboratory experiments of different engineering syllabuses. It is applicable also for other software languages especially graphical. Finally the suggestion of how by applying conception of virtual instrumentation and UML it is possible to create cost-effective solution for magnetic hysteresis measurement is appended.

References

[1] OMG Unified Modeling Language Specification, Version 1.5, March 2003.
[2] Charvat J., "Project Management Methodologies—Selecting, Implementing, and Supporting Methodologies and Processes for Projects", ISBN:0471221783, John Wiley & Sons, 2003
[3] Rumbaugh, J., Jacobson, I., Booch, G., "The Unified Modeling Language-Reference Manual", Addison-Wesley, ISBN: 0-201-300998-X, 1999.
[4] National Instruments "LabVIEW Development Guidelines", 2000.
[5] National Instruments, "Measurement and Automation Catalog", 2005
[6] Kis P., A. Iványi, "Computer Aided Magnetic Hysteresis Measurement in LabVIEW Environment", Journal of Electrical Engineering, Vol 53. No 10/S, 2002, 10-11.

Authors:

G. Nikolov, Dr. M. Marinov,
Technical University Sofia, Faculty of Electronics,
P.O. Box 43, BG-1756 Sofia, Bulgarien, Tel.: +3592 965 3677, e-mail: gnikolov@tu-sofia.bg, mbm@tu-sofia.bg

Dr. B. Nikolova
Technical University Sofia, Faculty of Communication,
BG-1756 Sofia, Bulgarien, Tel.: +3592 965 3203, e-mail: bgnikol@tu-sofia.bg

Volker Zerbe
Technical University of Ilmenau, Institute of Technical and Theoretical Computer Science, P.O. Box 100565, D-98684 Ilmenau, Germany, zerbe@theoinf.tu-ilmenau.de