

Testbench Development in a Distributed Collaborative Environment

Piotr Penkala¹, Dariusz Stachanczyk¹, Adam Pawlak¹, Matthias Bauer²

¹ *Silesian University of Technology, Gliwice, Poland, {penkala, darek, pawlak}@ciel.pl*

² *Infineon Technologies AG, Munich, Germany, matthias.bauer@infineon.com*

Abstract. The paper presents a new approach that enables distributed collaboration in a complex task of testbench generation. This approach is presented on a collaborative experiment between Infineon Technologies and SUT. Since it constitutes a real life outsourcing scenario with all security measures, conclusions from it are of more general value. The innovativeness of the approach lies in application of the distributed workflow technology and advanced collaborative infrastructure under restrictive industrial security conditions. Results and limitations of the proposed solution are summarized.

I. Introduction

New paradigms of engineering work are becoming feasible as a result of progress in information and communication technologies. The emerging new methods for product design and development, often called collaborative engineering [4], integrate widely distributed engineers for virtual collaboration. Collaboration is especially required in design of today's complex System-on-a-Chip (SoC). Distributed, collaborative design (often called concurrent design) of electronic systems has already found since mid 90's a number of outstanding examples, like WELD [12] and VELA [9]. New SoC design methodologies are proposed based on virtual components [10] and platforms [13] that support to some extent collaborative approach.

Collaborative design can become even more accepted paradigm, if technologies supporting collaboration based on the Internet would assure more security. It is authors' belief that current collaborative technologies are being developed orthogonal to protection ones, like firewalls. This explains in our opinion, one of major reasons, why with such a profound proliferation of networks, true collaborative engineering over wide distances is still a challenge. The second reason for a limited deployment of collaborative tools lies in the lack of generally accepted distributed infrastructures. Each collaborative environment comes with its own specific infrastructure. The situation resembles the one in the beginning of 90's with the CAD framework initiative. The positive difference lies however in the availability of universal access to the Internet through Web. This creates bottom-up pressure from engineers on EDA tools vendors that does really matter. The pressure is towards availability of tools over the Internet, and thus towards integration of tools across networks.

The demand for more collaborative style in engineering is globally sensed. This also results in a foreseen increase in outsourcing. With increasing outsourcing more SMEs need to be integrated into valued added chains of large companies. This was one of important incentives for Infineon Technologies when the company decided to join EU IST project E-Colleg in 1999 with a motivation to deploy new collaborative technologies to testing. Testing, and respectively testbench generation for complex components, are engineering tasks that due to their complexity and time-to-market pressure should also be supported by the new collaborative engineering style.

The paper presents shortly an experiment in collaborative testbench development that has been undertaken in the project E-Colleg "Advanced Infrastructure for Pan-European Collaborative Engineering" (IST-1999-11746). The overall objective of the E-Colleg project (www.ecolleg.org) is to provide a new platform for distributed collaborative engineering through the definition and

implementation of an advanced infrastructure that will offer sophisticated services for tool registration and management [3]. In contrast to current industrial practice and available frameworks, this infrastructure and technology will consist of a set of interacting, location transparent services that can be dynamically configured and adapted to arbitrary tool configurations and location-independent design teams at run-time. A prototype of such infrastructure - ASTAI® integration platform [1] has been provided to the E-Colleg consortium by Siemens Business Services (C-Lab, Paderborn).

II. Testbench Development at Infineon Technologies

The test environment called “Reflective” [2] (Fig. 1) that enables an engineer to reduce functional test complexity by a high degree of re-use is being used at Infineon Technologies. The main part of the Reflective Testbench Environment is an application-independent global synchronization unit called master controller. Its task is to control and synchronise the execution of commands of all testbench elements. Therefore, it reads a control file that contains the top-level functional test encoded in company-specific Reflective Test Script Language (RTSL). With this information, in a first step an internal data structure is created and the syntactic and semantic correctness of the input file is checked. After that, during a second step the specified commands are distributed to the Reflective Testbench Elements (RTBE).

The Reflective high-level test command that is usually related to one or more UUT interface protocol sequences, its arguments, as well as, the return command status and their arguments is transferred between master controller and RTBE by special interface called “channel”. Fig. 2 presents a physical structure of the Reflective Testbench Element (RTBE). The centre of each RTBE is a kernel, which can be generated out of a set of parameters by using the Infineon’s in-house VHDL pre-processor “vpp”. The RTBE packages that define the commands and their parameters enable communication through existing interfaces to the surrounding environment.

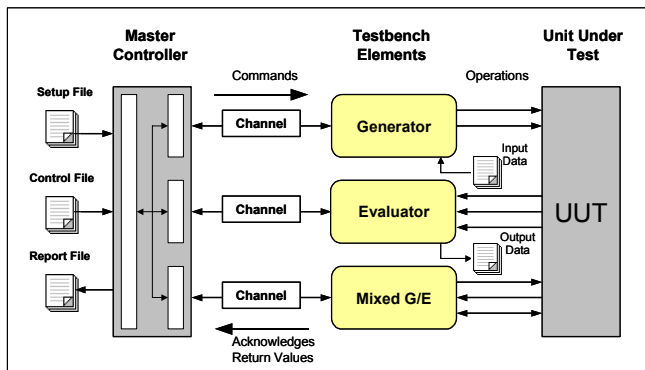


Fig. 1 Reflective overall structure

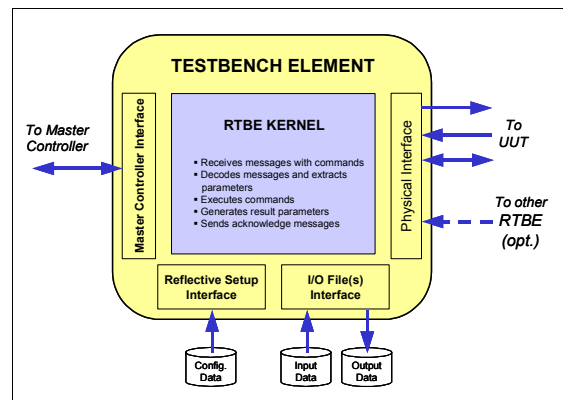


Fig. 2 Reflective Testbench Element (RTBE)

Besides an interface to the master controller (channel) the Reflective testbench element has usually an interface to the test port of the unit under test. Depending on the data transfer direction we can distinguish three kinds of RTBEs: they may only apply data to the UUT, then they are called “generator”, or only consume data from the UUT, then they are called “evaluator”. A combination of both is called “generator/evaluator”, which may be a bus functional model or a memory model. In general, the functionality of a testbench element can be described as follows. Firstly, a command received from the master controller is evaluated. Secondly, the activity associated to the received command is executed, e.g. an appropriate UUT interface is driven. Thirdly, return values including status information about the last command are sent back to the master controller.

Since the master controller is implemented as a separate VHDL design unit, it can be instantiated into a new test environment without any modification. The similar situation is with the RTBE elements, which once defined can be used in different test applications. Additionally, if it is

required, each RTBE can be instantiated in one test environment more than once. Thus, the required test environment can be quickly created by reuse of an existing RTBE elements. Simultaneously with the RTBE the already existing RTSL control file can be used. Because Reflective is widely used at Infineon Technologies, a set of different simulators must be supported. This means that the entire analysis and verification in the test environment can be performed on different VHDL simulators and in addition, different versions of these simulators.

Currently, within Infineon only centralised workflows for the Reflective environment are used. These workflows are based on the public domain revision control system CVS without a workflow manager. Each project uses several tcsh scripts together with a module concept which is also available in public domain. The first task is to check out the data from the CVS repository. With the module concept different initialisations can be made. For example, different VHDL compilers can be chosen. A “make” mechanism enables different tasks like compile or simulate. We distinguish between vpp-make files for our VHDL pre-processor and VHDL-make files, which are generated automatically. Simulation can be controlled with tcsh scripts. A regression test environment enables the verification of changes in the Reflective Master Controller, as well as, in all related Reflective VHDL packages. Also Infineon proprietary tools for checking coding rules are used.

III. The problem of distributed testbench generation

Distributed testbench generation is understood as a development of testbenches that is performed by a group of designers collaborating in a distributed environment. The testbenches necessary for functional validation of a component or a system can be generated both manually and automatically. This distributed process obviously faces similar challenges as a traditional approach e.g., growing complexity of functional validation along with time-to-market shortage pressure. Additionally, distributed testbench generation deals with obstacles specific to distributed software in network environments. Some issues that have to be taken into account are as follows:

- preparation of a distributed verification strategy (dividing validation process, allocation to distributed resources, synchronisation, modelling of complex interactions),
- setting-up of an environment open for upcoming tools and modifications of methodology,
- resources management (a common/distributed repository, access rights, keeping data base safe and updated), and testbench reuse.

Despite that, there are many other more general advantages of the distributed approach, like:

- integration of SMEs into value added chains of large companies,
- preservation of severe industrial security precautions,
- setting-up collaboration on the Internet,
- giving access to external resources.

There are several examples of distributed environments for design and verification of electronic components and systems. Among them the most interesting from the perspective of our experiments are: MOSCITO [6][7][8], REUBEN [8], and JavaCAD [10]. These systems represent various approaches in enabling distributed, collaborative work of engineers. The most common model of collaboration enables remote access to specialised design tools only. Additional features are available in systems based on the workflow technology. Such systems enable involvement of required design tools and control of project data flow. The examples of such systems are MOSCITO and REUBEN. A different approach presents the JavaCAD framework. It supports electronic designs based on virtual components (VC) for digital system validation. The collaborative experiment that we are referring in this paper to, has been based on ASTAIR®. So, further on we shall address and assess the functionality of ASTAI® in respect to remote tools integration.

IV. Towards a new environment for collaborative testbench development - Collaborative Experiment

In order to improve functionality of the Reflective environment a new approach was adopted. This approach to testbench generation is based on the preparation of a new distributed collaborative environment. We decided to apply a distributed workflow technology [5]. This technology requires a management system to control the design flow. We utilised ASTAI® workflow management system. The process of extending the Reflective environment functionality was divided into two consecutive steps. The first step leads to enabling a distribution of the Reflective environment. The second step consists of development and deployment of the workflow (diagram) representation of the Reflective flow. The goal of the experiment was to develop a set of testbenches for functional verification of a Serial ATA (SATA) hard disk drive controller VHDL model using the distributed workflow functionality of ASTAI®. The first step in a collaborative design process defined the design flow according to the Infineon's methodology. The functionality of the Reflective was encapsulated in a set of tasks that can be invoked from the workflow environment. Then, design responsibilities were split between SUT and IFX.

A. Testbench generation workflow

The collaborative workflow comprises three main phases of a design process, e.g., development of the interface model, generation of the testbench element, and application integration. Another very important part of the workflow is the debugging process. The workflow is presented in Fig. 3. The main aim of the first phase of the workflow is implementation and verification of the SATA interface behavioural model for the Reflective Testbench Element (RTBE). The activities in the Phase I, performed at SUT, include requirement test specification, physical interface modelling, and verification activities of a physical interface. The first phase uses a number of EDA tools for code editing, debugging and verification.

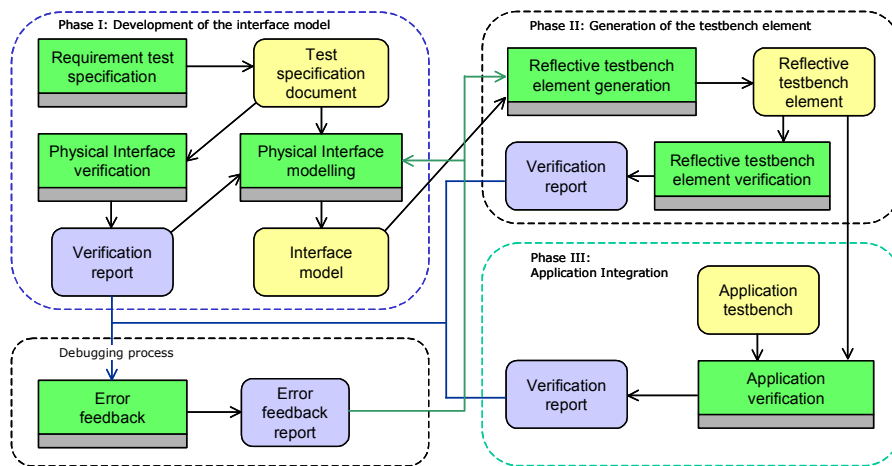


Fig. 3 Collaborative testbench development diagram.

The developed and functionally validated SATA package is used in the Phase II of the workflow for the generation of the testbench element. The first activity invokes a set of Infineon's proprietary tools to generate the SATA RTBE. Then, the RTBE verification activity is performed to check if the created testbench element is properly combined into the Reflective test environment. In the Phase III of the workflow the generated RTBE along with an application testbench are integrated into a current industrial application. The Phases II and III are performed at Infineon. All necessary files written or generated in the workflow are stored in a CVS repository common to both partners. An access to the repository can be achieved in a secure manner using SSH protocol. At the end of

each designer's work, the verified design is stored into the repository. Simultaneously, the repository enables project file exchange between Infineon and SUT.

B. Deployment in ASTAI®

The workflow was implemented using the ASTAI® environment installed on Unix workstations in the distributed domains of Infineon (IFX) and SUT. A workflow management system controls the overall design flow, in particular, launching the project activities and data transfer. The first phase of the IFX-SUT workflow was successfully deployed and performed at SUT. As a result the bus functional VHDL model of the SATA interface was developed. Then, the model was functionally verified and delivered to IFX. The collaborative infrastructure was validated by SUT as a local area network infrastructure only. This collaborative infrastructure based on ASTAI® supported neither collaboration in a real heterogeneous environment, nor multi-user cooperation.

The SATA interface model Phase II of the IFX-SUT workflow comprises the generation of the Reflective Testbench Element. The developed RTBEs are utilised in a real application in Phase III of the workflow. Both Phase II and Phase III are performed by IFX engineers in Munich. All the necessary files written or generated in the Phase I are transferred to the sub-workflow of the Phase II. These files can be checked out from the CVS repository. This can be achieved in a secure manner using SSH protocol. Once all the required files are available at IFX side, a generation of a testbench element is carry out in the Reflective environment. The Phase II of the IFX-SUT workflow was partially implemented in the ASTAI® environment by SUT. This phase can be currently executed at IFX using shell scripts already prepared for the Testbench Element generation in the Reflective environment. No workflow management system is utilised for this phase of the workflow.

C. Comparison of centralised and distributed Reflective environments

The main drawbacks of the current centralised Reflective environment are as follows: not intuitive operation with complex control of the environment, textual and rather cumbersome user interface, availability limited to the internal Infineon's network only. Access for external contractors is very limited under severe security restrictions in the company.

Tab. I Comparison of features of a centralised and a distributed environment

	enabling collaboration	data access	workflow representation	versioning system	internet access	LAN access	knowledge of environ.
Centralised environment	no	manual	textual/script	CVS	no	yes	necessary
Collaborative environment	yes	automatic (through ASTAIR)	visual	CVS	yes	yes	not necessary

The new distributed environment overcomes some of these disadvantages and gives new functionality to the testbench generation process. First of all, deployment of the common environment unifies the design flow of involved partners. Then, it grants at least partial access to the project database (possibility of a distributed repository). Finally, a partner can easily get feedback from each other during a design process. Short comparison of both approaches is given in Tab. I. The application of the collaborative technology based on ASTAI® with dynamic, executable workflows opens new possibilities for Infineon's testbench generation tasks, because the existing environment can be used on each partner site and the distributed environments will be coupled by ASTAI®.

D. Security aspects

Two elements of the collaborative infrastructure were missing and thus prevented us from a full deployment of distributed workflows between IFX and SUT. The ASTAI® environment does not allow for a coupling of distributed workflows when a very strict network security policy is applied. For the same reason ASTAI® did not allow for a file exchange between IFX and SUT. These both elements are relevant for the Internet connection between Infineon and SUT. The only possible direct communication through the Internet succeeded when Infineon initiated a file transmission. However, a workflow management system, i.e. ASTAI®, can be deployed once the security problems are solved.

V. Conclusions

Distributed testbench generation spanning over a number of organisations still constitutes a real challenge. Nevertheless, with the experiments undertaken in E-Colleg, the important elements of the approach, namely: application of the distributed workflow technology to testbench generation and validation, and the advanced collaborative infrastructure with innovative services supporting remote invocation of tools in a secure manner, have been achieved. The main difficulty experienced with the current technology is to cross firewalls and proxy servers. E-Colleg project develops a solution based on the new, flexible approach, called “Tool Registration and Management Services”. TRMS [11] addresses security aspects, as well as, registration and management of remote design tools. It is supposed that the new TRMS technology will comply with a strict network security policy of such companies like Infineon. Therefore, TRMS should enable coupling of distributed workflows and file transfer among designers across firewalls.

Acknowledgements

The authors would like to thank all partners of the E-Colleg consortium for fruitful cooperation that has allowed to advance the development of the collaborative infrastructure that was used in collaborative testbench generation. Many thanks to C-Lab group, namely Dr. Wolfgang Mueller, Tim Schattkowsky, Dr. Heinz-Josef Eikerling, and Siegfried Bublitz for support to ASTAI® experiments, and collaboration on TRMS. The work was partially supported by European Commission (project E-Colleg, IST-1999-11746) and State Committee for Scientific Research (KBN).

References

- [1] ASTAI®: <http://www.c-lab.de/astair>
- [2] Bauer, M.; Ecker, W.: REFLECTIVE, a reusable extendable flexible testbench, DAC, Anaheim, USA, June 1997.
- [3] Bauer, M., Eikerling, H.J., Mueller, W., Pawlak, A., Siekierska, K., Soderberg, D., Warzee, X.: Advanced Infrastructure for Pan-European Collaborative Engineering, e-Business and e-Work Conf., Venice, Italy, IOS Press, 2001.
- [4] Cutkosky M.: Madefast: Collaborative Engineering over the Internet, Comm.of the ACM, Sept. 1996, vol. 39, no. 9.
- [5] Kokoszka A., Siekierska K., Trung N., Fras P., Pawlak A.: Are Workflow Management Systems useful for Collaborative Engineering? ECPPM, eWork and Business in AEC, Portorož, Slovenia, Sept. 9-11, 2002.
- [6] Schneider A., Ivask E., Miklos P., Raik J., Diener K.-H., Ubar R., Cibáková T., Gramatová E.: Internet-based Collaborative Test Generation with MOSCITO. DATE 2002, Paris, March 4-8, 2002.
- [7] Schneider P., Schneider A., Bastian J., Reitz S., Schwarz P.: MOSCITO - A Program System for MEMS Optimization. DTIP, Cannes, May 2002.
- [8] Ivask, E.; Ubar, R.; Raik, J.; Schneider, A.: Internet-based test generation and fault simulation, DDECS 2001, Gyor (Hungary). CBL Demos <http://www.cbl.ncsu.edu/demos/>
- [9] Vela Project on Collaborative Distributed Design: New Client/Server Implementation, http://www.cbl.ncsu.edu/publications_misc/1999-EETimes-vela.pdf
- [10] Delpasso M., Bogliolo A.: Specification and validation of distributed IP-based designs with JavaCAD, DATE 1999.
- [11] Eikerling, H.J., Mueller, W., Wegner, J.: Werkzeugintegration und -verwaltung in heterogenen Computernetzwerken, it+ti 3/2002, Oldenbourg, 2002.
- [12] WELD: Web-based Electronic Design project at Univ. of California, Berkely, <http://www-cad.eecs.berkeley.edu/Respep/Research/weld/>
- [13] Sangiovanni-Vincentelli, A.: Defining Platform-based Design. EEDesign of EETimes, February 2002.