20

HARPO: TESTING TOOLS DEVELOPMENT

E. Algaba, M. Monedero, E. Pérez, O. Valcárcel
Telefónica I+D
Emilio Vargas, 6
28043 - Madrid (Spain)
algaba@tid.es, mmr@tid.es, emilla@tid.es, ovr@tid.es

Abstract

This paper briefly introduces the HARPO testing tool generation toolkit. The features of the different tools included within HARPO are shown: automatic test generator, TTCN compiler, PICS editor, as well as their role in the testing tools derivation process. Finally, the main features derived from the operation of a testing tool obtained with HARPO are defined.

Keywords

ISO IS-9646, TTCN, ATS, ETS, SDL, MSC, PICS, PIXIT

1 INTRODUCTION

The ISO conformance testing methodology (ISO IS-9646, OSI Conformance Testing Methodology and Framework) is widely accepted as the main framework in telecommunication systems testing. This methodology includes general concepts on conformance testing and test methods, the test specification language TTCN (ISO IS-9646 Part 3), and the process of specifying, implementing and executing a test

Testing of Communicating Systems, M. Kim, S. Kang & K. Hong (Eds)
Published by Chapman & Hall

campaign.

The HARPO toolkit was developed according to this methodology, although its application scope can be extended to other currently more useful kind of tests: interoperability, load, traffic, end to end testing (interworking), etc. HARPO is a set of tools based on both formal system and test specifications of the protocols under test. It has been designed to automatize the process of obtaining the final executable testing tool as much as possible.

This article deals with the methodology and functionality of the HARPO development toolkit.

2 HARPO: TEST DERIVATION AND OPERATION

The HARPO toolkit allows the generation and operation of test suites, with the purpose of automatizing as much as possible the process of specification and implementation of testing tools for protocols, services and communication systems in general. HARPO is composed of a set of tools which automatize not only the process of specifying a test suite, but also that of obtaining an executable version to be run on a final execution platform. The whole process is shown in figure 1.

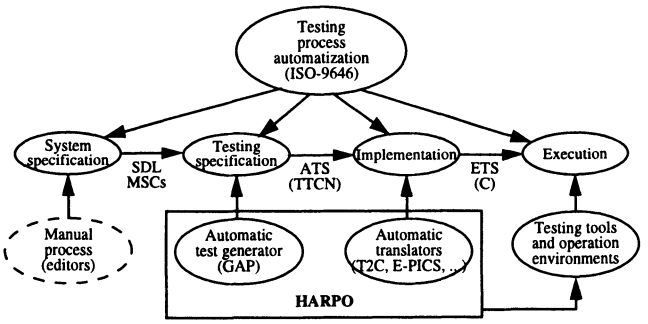

Figure 1 HARPO methodology and testing process.

From the user's point of view, HARPO offers two different set of functionalities: a set of tools to develop the testing tool, including a test generator (GAP), and TTCN translators (TTCN compiler, PICS editor, etc.), and an operation environment for the generated testing tool, using the distributed execution architecture defined in HARPO.

2.1 ATS generation: GAP

The automatic test generator, GAP (figure 2-a), enables its user to derive TTCN test suites automatically. The inputs to this tool are the formal specification of the system written in SDL (ITU-T Z.100 and Z.105) for which test cases are to be automatically derived, and the definition of the test purposes that guide the derivation process, written in MSC notation (ITU-T Z.120) extended to allow the definition of open behaviour patterns for the test purposes.

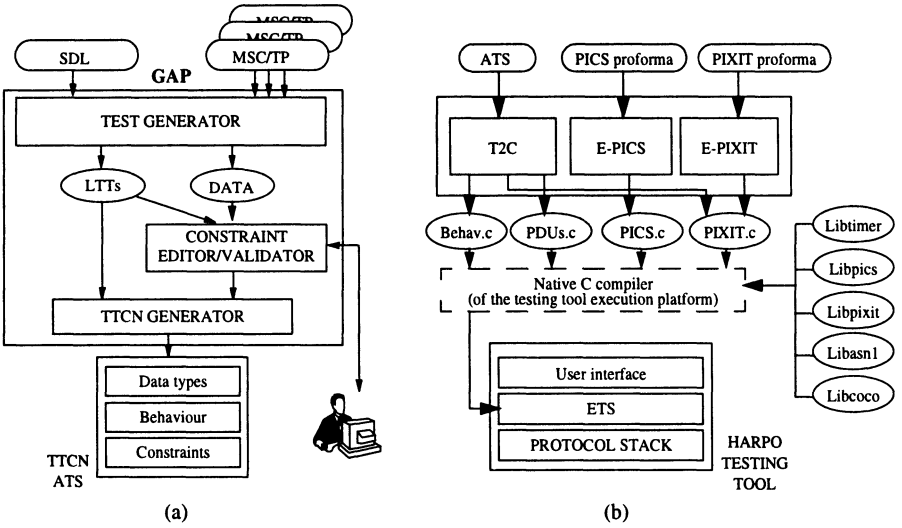

Figure 2 Test generation and ETS derivation subsystem architecture.

Each test purpose is simulated against the SDL specification, allowing GAP to generate several test cases for it, comprising all possible behaviours that fit the test purpose. Data types and constraints are also generated. Constraints with associated predicates (gathered while simulating) may need the intervention of the user to fill in the appropriate values to verify these conditions.

The output of this HARPO subsystem is a complete compilable test suite in TTCN, (behaviour, data type definitions, constraints, etc.). The system does also provide coverage measures achieved by the tests with respect to the formal specification of the system under test. State, signal and transition coverage measures are computed, as well as incremental measures, which allow for the comparison of different test suites in terms of quality.

The GAP tool highly automatizes the process of generating test specifications. Usage of this tool provides enormous advantages with respect to a manual specification of test cases: costs are significantly reduced and the quantity and quality of the generated tests are increased.

2.2 ETS derivation and operation

The ETS generation subsystem (figure 2-b) takes the TTCN ATS (concurrent or not), PICS and PIXIT definitions as its input, in order to derive an executable version of the test suite. It comprises three tools: a TTCN compiler (T2C), a PICS-proforma editor (E-PICS) and a PIXIT-proforma editor (E-PIXIT).

E-PICS provides editing and error checking capabilities, and translates the proforma into C code. E-PIXIT provides editing capabilities and translates the proforma into C code. T2C is also able to generate the C code for the PIXIT-proforma automatically, from the information included in the ATS. The proformas (in C code) are included in the executable testing tool. Thus, they can be dynamically

managed in run-time (of the testing tool), i.e., building and initializing the proforma, reading and/or writing individual values of PICS and PIXIT, performing static conformance review, etc., allowing the parameterization of the executable test suite (selection expressions, external parameters, etc.).

T2C translates a TTCN-MP ATS to the C code that implements it. T2C is ATS independent and generates C code for the dynamic part, coding, decoding, building, identifying and matching of constraints (including CMs, if concurrent) for tabular and ASN.1 definitions.

Several libraries provide auxiliary functions to the generated code: timer and pics and pixit proformas management, tabular and ASN.1 auxiliary coding/decoding functions, etc. They depend on the platform(s) on which the testing tool is to be executed.

The architecture of a HARPO generated testing tool is depicted in figure 3.

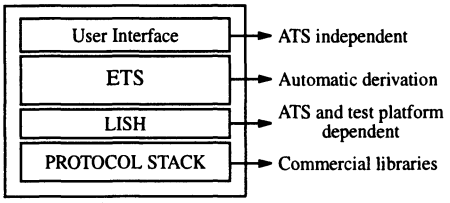

Figure 3 Architecture of a HARPO testing tool.

The LISH support library in figure 3 adapts the interfaces between the automatically generated code and the commercial protocol stack available in the execution platform.

HARPO provides the auxiliary libraries and local, remote and distributed (concurrent) user interfaces for several platforms, including WS with SunOS and Solaris, Chameleon-32 and Chameleon-Open from Tekelec, PC with Windows95 and PT-500 from HP.

Non concurrent testing tools, can be operated locally in the test execution platform or remotely from another machine. HARPO uses a client/server architecture to allow ETS remote operation. In this case, the user interface of the testing tool (figure 3) is no more a graphical user interface (local operation) but the server process of the client/server model.

The same architecture does perfectly suit the distributed operation environment imposed by concurrent TTCN, where the ETS is split in separated processes running in different test platforms. An example of concurrent testing tool is depicted in figure 4. This example shows a HARPO testing tool distributed in three different test platforms.

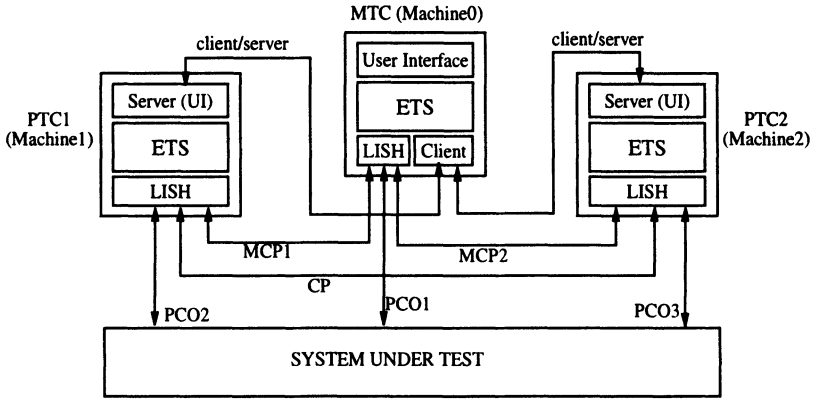

Figure 4 Example of concurrent operation environment.

Since code generated by HARPO allows distributed operation for concurrent test cases, the elements included in such HARPO testing tool must correspond to those defined in the TTCN specification. The ETS is splitted in different components (MTC and PTCs), in such a way that it is possible to build small HARPO testing tools to implement each ETS part, which are coordinated through the defined coordination points (MCPs and CPs) using the corresponding coordination messages (CMs). Concurrent operation and coordination is carried out within LISH, which is responsible of two different matters: PCOs and CPs management. The user interfaces of the PTCs become server processes in communication with the client running in the MTC.

To sum up, developing an executable testing tool is greatly automatized using the HARPO ETS derivation subsystem, thus reducing the development time and the complexity of the process.

3 CONCLUSIONS

The HARPO toolkit provides a high level of automatization in the testing tools development and operation process due to the automatic test generation subsystem (complete TTCN test suite, few user inputs required), the ETS derivation subsystem (data types and constraints handling for tabular and ASN.1, ATS and test platform independent, PICS and PIXIT management embedded in the ETS) and its final testing tool operation environment (local, remote and distributed --concurrent-- testing architecture based on client/server technologies, generic dynamic user interface).

The tools provided by HARPO, following ISO-9646 standard, impact directly on the productivity of the testing tools development and operation:

- Reducing the testing specification and implementation time.
- Generating many more test cases than in a manual process. Quality measures of the generated test suites are available (coverage).

- Easy maintenance and updating of generated testing tools.
- High automatization degree in the complete testing tool development and operation process.
- The C-code produced by HARPO can be easily run on a wide variety of hardware platforms (general purpose computers, protocol analysers, etc.).

HARPO has been successfully used to develop a high number of testing tools: Multilink X.25, ISDN-FAX G4 (transport, session and application protocols), X.32, SS7 (TUP, ISUP, ISDN Access), ISDN interworking (EURESCOM P-412), Core-INAP/SSP (Intelligent Network), ATM UNI (ATM layer), ISDN Supplementary Services, TBR3&4.

These tools are running in many different hardware platforms: WS with SunOS and Solaris, Chameleon-32 and Chameleon-Open from Tekelec, PC with Windows95, PT-500 from HP, etc.

4 BIOGRAPHY

Enrique Algaba is the project manager of testing engineering group in Telefónica I+D (R&D Labs.). Since joining Telefónica I+D in 1988, he has been engaged in the research and development of protocol testing tools and automatizing the testing process.

Miguel Monedero joined Telefónica I+D (R&D Labs.) testing engineering group in 1995. He has been working since then in the fields of test generation and testing automatization.

Esteban Pérez joined Telefónica I+D (R&D Labs.) testing engineering group in 1993 and has been engaged with test generation techniques and in automatizing the development and operation of protocol testing tools.

Oscar Valcárcel began working in Telefónica I+D (R&D Labs.) in 1988, joining the public telephone system development group. In 1992 he was assigned to the testing engineering group, where he has been devoted to automatizing the development and operation of protocol and service testing tools.