9th *LISA Symposium, Paris*
ASP Conference Series, Vol. 467
G. Auger, P. Binétruy and E. Plagnol, eds.

The LISA PathFinder DMU Software, a Global Overview

L. Gesa, V. Martin, A. Conchillo, A. Lobo, I. Lloro

ICE-CSIC/IEEC, Barcelona, Spain

Abstract. During the last 6 years, the IEEC software team has developed the Data Management Unit (DMU) software. It is a critical piece of engineering in the Lisa Pathfinder mission, in charge of the primary processing of data generated by the Phasemeter, acting also as an interface between the whole LISA Technology Package (LTP) and the On-Board Computer (OBC). It also manages and controls the diagnostics systems (Heaters, coils, magnetometers, radiation monitor, thermometers). This article briefly summarizes all the work performed, describing the complex environment built around the application generation.

1. Introduction

LISA Pathfinder will pave the way for a major ESA/NASA mission planned for the near future: eLISA/NGO (Evolved Laser Interferometer Space Antenna / New Gravitational Wave Observatory). One of its critical parts is the DMU, a data management unit running two complex softwares: the Boot Software (BSW), in charge of booting the system, and the Application Software (ASW), the full operating application.

2. Design

Application Software was designed to be highly modulable, with 5 main blocks: One is Basis library, a common routine set used in both BSW and ASW. Data information is stored in RAM storage using a System Data Pool strategy (SDP) shared between BSW and ASW. Communication Drivers were built from scratch over Sμmmit chipsets using MIL-STD-1153 bus protocols for communications. Data layer uses the ESA Pus Utilization Packet (PUS) Standard. Regarding O.S. layer, in BSW a minimalistic Operating System was completely implemented, whereas in ASW the *RTems* is used. And finally we can find the ASW itself, the core of the control & algorithms, main software, patchable by ground. Running at Real-Time, ensuring phasemeter processing and laser loops computing at 100hz, and providing information to OBC at 10hz.

3. Technology and Test

Since first commit on December 2005, system has been built basing its development on several technologies. C is used as main programing language, more precisely the C99. Used for the application itself and for the simulators, LTP and Diagnostics, for test. In other side: Python, is used as main support language for the Build System and Testing Framework. Other point to remark is that *RTems* is used, in ASW, as operating system, a well-known system in Real-Time embedded software for space missions. Finally, some pieces have been built directly in *SPARC-V7/ERC32* assembler.

Software has been deeply tested following the standard engineering software protocols: Unit Test as first stage, where each individual code method is tested individually. Validation Test, where each requirement is tested using a hardware flight equivalent model. And finally the

Stress Test, where software is executed in several full load scenarios during large time periods in order to stress the system.

4. Application in numbers

The amount of labor performed is complex to summarize in few pages. Some numbers on ASW should help grasp the extend of the task done.

406646 lines of code (LOC) (see details in table). 4.7 lines of test per line of code. 42512 lines of source comments (ASW + Basis) this means 0.75 lines of comments per LOC. 622 files in ASW source structure (387 for binary and 237 for Unit Test). 423 files in Basis Library structure (292/binary and 131/Unit Test). 137272 comments in whole Validation (Unit Test not included), that is: 0.51 lines of comments per line of validation code. 1506 files in Validation structures (Unit Test not included). 2959 commits in repository (ASW + Basis)

Module	*Lines of code*
Basis (C99)	16411
ASW (C99)	40263
Unit Test (C99)	83207
Validation and Stress	
Common (C99)	8400
Common (Python)	18697
Stress Test (Python)	5263
Interface (Python)	10219
Functional (C99)	16012
Functional (Python)	208174
Total (C99)	164293
Total (Python)	242353
Total	406646

5. Conclusion

On March 2012, the release ASW-v3.4 has been officially proposed to be accepted as flight software for LISA Pathfinder. As a critical part, the DMU software has been designed, implemented, and tested with special care to guarantee its correctness. All the targets for this complex system have been achieved: All requirements have been accomplished. It has been thoroughly tested, with almost 5 lines of test code per software code. All possible scenarios have been tested, with different load, with expected results. and it is ESA standards compliant, and accepted as a validated software. Then we can conclude that is Fully functional, Reliable, Robust, and Qualified. Hence, it is expected to be a system whose architecture could be re-used in part in future satellite mission software, in particular in GW missions.

Acknowledgments. *We acknowledge support from Project AYA2010-15709 Espacio of the Spanish Ministry of Science and Innovation (MICINN)*

References

C99 -, C-standards http://www.open-std.org/jtc1/sc22/wg14/
ERC32 -, Esa microelectronics, http://microelectronics.esa.int/erc32/erc32.html
RTems -, The real-time executive for multiprocessor systems, http://www.rtems.com