WP12 Assessment and evaluation

Deliverable D12.1
Assessment and Evaluation plan

WP12 Assessment and Evaluation : Deliverable D12.1 Assessment and Evaluation plan
by Stanislav Benes

Published March 2003

Table of Contents

Project Coordinator iv
Participant List....iv
Document version....iv
Chapter 1. Introduction 1
1.1 Fields of requirements....1
1.2 Data of SW life cycle....1
Chapter 2. Criteria for SW life cycle documentation....2
2.1. SW development 2
2.2 SW configuration management 2
2.3 SW quality assurance 2
2.4 SW requirements 2
2.5 SW verification 3
2.6 SW quality assurance records 3
2.7 SW configuration index 4
2.8 Evaluation Report 4
2.9 Conclusion 4
Chapter 3. Appendix A - Tracing Table....5
A01 The improving of the timing performance of real-time systems....5
A02 The improving of reliability of real-time systems....6
A03 The reduction of the development time of a real-time application....7
A04 The incorporation of Quality of Service Management into real-time systems....8
A05 Two levels of scheduling in the RTLinux operating system 9
A06 Fault tolerant components for real-time applications....10
A07 Predictability, fault-tolerance and high performance in communication....11
A08 The incorporation of specific modes at the components level 11
A09 Adaptability to several HW configurations 12
A10 Support for a wide variety of applications with different level of criticality 12
A11 Scalability from a small embedded to a full-featured general purpose OS 12
A12 More competitive Europe in the field of embedded systems 12

Document Presentation

Project Coordinator

Organisation:	UPVLC
Responsible person:	Alfons Crespo
Address:	Camino Vera, 14, 46022 Valencia, Spain
Phone:	+34 963877576
Fax:	+34 963877576
Email:	alfons@disca.upv.es

Participant List

Role	Id.	Participant Name	Acronym	Country
CO	1	Universidad Politecnica de Valencia	UPVLC	E
CR	2	Scuola Superiore Santa Anna	SSSA	I
CR	3	Czech Technical University in Prague	CTU	CZ
CR	4	CEA/DRT/LIST/DTSI	CEA	FR
CR	5	Unicontrols	UC	CZ
CR	6	MNIS	MNIS	FR
CR	7	Visual Tools S.A.	VT	E

Document version

Release	Date	Reason of change
1_0	01/11/02	First release
2_0	10/03/03	First Review report recommendations

Chapter 1. Introduction

This document should provide a list of criteria for assessment and evaluation of whole OCERA project. Stated criteria are focused to SW functionality, SW measurable parameters and SW life cycle documentation.

1.1 Fields of requirements

Fields of requirements derived from Annex1 are following:

1. The improving of the timing performance of real-time systems
2. The improving of reliability of real-time systems
3. The reduction of the development time
4. Quality of service management
5. Two levels of scheduling
6. Fault tolerant components
7. Predictability, fault-tolerance and high performance in communication
8. The incorporation of specific modes at the components level
9. Adaptability to several HW configuration
10. Ability to support a wide variety of applications with different level of criticality
11. Scalability from a small embedded one to a full-featured general purpose OS
12. More competitive Europe in the field of embedded systems

Detailed list of requirements, derived from Annex1 and from D3.1 including measurable parameters, is stated in Appendix A.

1.2 Data of SW life cycle

Data of SW life cycle , which should arise during whole OCERA project are following:

1. Document OCERA Architecture and Component Definition
2. Deliverable 8.1 Integration plan
3. Deliverable D10.2 CVS server
4. Deliverable D10.4 (D10.8) Programmer's Guide
5. Deliverables D4.2_rep, D5.2_rep, D6.2_rep, D7.2_rep with a component verification description
6. Deliverables D4.4_rep, D5.4_rep, D6.4_rep, D7.4_rep with a component verification description
7. Deliverable D8.1_rep Integration and configuration tool
8. Deliverables D12.2, D12.3 Evaluation report

This documentation is in accordance with standards of system quality management usual in industry for non-critical applications. Future extension for critical applications is possible.

Chapter 2. Criteria for SW life cycle documentation

2.1. SW development

Document “OCERA Architecture and Component Definition”
This plan provides an overview of newly developed SW components including a common view of all partners to the SW architecture and HW/SW interfaces. A responsibility of consortium members for named components should be stated.

Deliverable D8.1 “Integration plan”
An intended type of HW platform, operating system, compiler and linker for development purposes should be stated.

2.2 SW configuration management

Deliverable D10.2 “CVS server” should contain:

A guide how to use CVS under SourceForge, i.e. a description of functions, a guide how to get from CVS a file and how to put a new version in CVS.

A description of problem reporting, i.e. reporting form. Report should use identification of SW version, HW configuration and verification case.

Change control description, intentions regarding configuration identification, i.e which items are to be identified by version, when the version should be changed.

2.3 SW quality assurance

Document “OCERA Architecture and Component Definition” should contain:
A description who is responsible for what, erg. who is developer and who is verificator.

Deliverable D10.2 “CVS server” should contain:
A description who are the persons authorized to have an access to data.

Deliverable D8.1 “Integration plan” should contain:
A description who is the person responsible for SW and documentation releasing.

2.4 SW requirements

Deliverable D10.4 (D10.8) “Programmer's Guide” should contain:

For every SW component with its own API:
A description of its parameters, functions, time responses and the precision of output analog values (if exist).

A driver development guide.

2.5 SW verification

Deliverables D4.2_rep, D5.2_rep, D6.2_rep, D7.2_rep should contain:

A description of used verification methods.

A description of used verification environment, i.e. the equipment for testing, tools for testing and analysis and guidelines for applying these tools and HW test equipment.

A description of intended re-verification guidelines, i.e. rules how to find out which verification cases should be repeated after changing of some SW source code modules.

At least for every parameter stated in Appendix A of this plan:

A description of concrete tests, reviews or analysis procedures including expected results and pass/fail criteria.

A result description including a verified component identification and a final pass/fail result. Recommended form of the information is a Test Coverage Table:

Parameter name	**Verification case identifier**	**Verification result identifier**	**Passed / Failed**
POSIX Barriers	D5.2_rep, chapter POSIX Barriers, point Tests	D5.2_rep, chapter POSIX Barriers, point Verification results	Passed
POSIX Barriers - overhead	D5.2_rep, chapter POSIX Barriers, point Tests	D5.2_rep, chapter POSIX Barriers, point Verification results	Failed

Where:

D5.2_rep, chapter POSIX Barriers, point Tests is a description of a verification case in form of test or source code analysis.

D5.2_rep, chapter POSIX Barriers, point Verification results is a description of results of a test or analysis.

Deliverables D4.4_rep, D5.4_rep, D6.4_rep, D7.4_rep should contain:

At least for every parameter stated in Appendix A of this plan:

A description of concrete tests, reviews or analysis procedures including expected results and pass/fail criteria. A result description including a verified component identification and a final pass/fail result. Recommended form of the information is a Test Coverage Table again.

2.6 SW quality assurance records

SW quality assurance records should contain:
Records about each releasing of SW or documentation including signatures of

responsible persons. Alternatively it can be ensured automatically e.g. by means of CVS.

2.7 SW configuration index

Deliverable D8.1_rep “Integration and configuration tool” should describe how the following items are identified:

Executable codes.

Source codes.

Makefiles.

Compilers.

Linkers.

Archive and release media.

2.8 Evaluation Report

Deliverables D12.2, D12.3 “Evaluation report” should contain:

Software overview, i.e. a brief description of SW functions and an explaining differences from proposed intentions (e.g. in Annex1).

Software characteristic, i.e. description of main SW parameters (e.g. size, timing, resource limitations) and the means of measuring each parameter.

Software life cycle, i.e. a summary of actual SW life cycle and an explaining differences from intentions proposed in Annex1.

Software identification. i.e. version of the software which is evaluated.

Software status, i.e. summary of problem reports unresolved at the time of evaluation.

2.9 Conclusion

Appendix A contains list of main requirements derived from Annex1 and from WP3.1 “Feedback from RTOS users” and should serve as a support tool for tracing how the requirements are performed. Complete overview of SW requirements is expected in form of “Programmer's manual”, where every SW component with its own API should be described, as it is stated in this document, section SW requirements.

Filled in Appendix A (SW overview), verification results (SW characteristic, see section SW verification) and criteria stated above (SW life cycle summary) will serve as main data for the Evaluation report.

Chapter 3. Appendix A - Tracing Table

This TracingTable contains list of measurable and verifiable criteria of the project and traces them to the fields of requirements stated in Annex1 page 6.
The requirements stated in D3.1. were used as the main source of the criteria.

A01 The improving of the timing performance of real-time systems

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
POSIX Barriers	It is a synchronization mechanism that allows to synchronise several threads at a specific point until the last one has reach it	First implementation of the Standard	UPVLC
POSIX Barriers-latency	Time between an event occurrence and an active task activating	The same latency as mutexes latency	UPVLC
POSIX Barriers-overhead	Overhead introduced into SW by Barriers	Overhead less than using mutexes	UPVLC
POSIX Barriers - maximum number	Maximum number of Barriers in application	User defined number of Barriers	UPVLC
POSIX Messages	POSIX Messages with prioritized message queue in RTLinux	New implementation	UPVLC
POSIX Messages – messages number	Maximum number of messages in queue	User defined maxi-mum number of Messages	UPVLC
POSIX Signals	Fully UNIX compatible Signals in RTLinux	Enhanced implementation of the RTLinux version	UPVLC
POSIX Signals – interrupt latency	Time between interrupt occurrence and start of Signal handling in an active task	< 5 microseconds Pentium III/500 MHz	UPVLC
POSIX Timers	POSIX timers in RTLinux	New implementation	UPVLC
POSIX Timers - resolution	Minimum time slice of RTLinux	1 microsecond Pentium III/500 MHz	UPVLC
POSIX Timers - timer addition	A facility for new HW timers addition as a system service	User defined number of timers	UPVLC
POSIX Timers - overhead	Linear overhead dependency on the number of armed timers	Timer Overhead O(n)	UPVLC

A02 The improving of reliability of real-time systems

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
PowerPC porting	Porting RTLinux and OCERA components to a more reliable processor	New implementation	UPVLC
SQM rules keeping	System Quality Management for SW development during OCERA project, usability in the industry	ALL criteria in chapter "Criteria for SW life cycle documentation"	UC

A03 The reduction of the development time of a real-time application

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
POSIX Trace	Debugging tool, allows to register specific application events	First implementation of the POSIX Standard	UPVLC
POSIX Trace - register latency	Time between begin and end of event registration	< 1 microsecond Pentium III/500 MHz	UPVLC
POSIX Trace - overhead	Overhead introduced into SW by tracing	Added constant overhead in each syscall	UPVLC
RTLGnat	RTLGnat - ADA compiler for RTLinux	New implementation	UPVLC
RTLGnat - task execution performance	ADA compiler for RTLinux efficiency	CPU utilization >95% of RTLinux tasks	UPVLC
User RM API	Set of libraries accessing Resource Management services, facilitates to change task environment (RTLinux or Linux)	First implementation	SSSA
User RM API – rewriting effort	Efficiency of the set of libraries accessing Resource Management services for task environment changing	Amount of rewriting needed to port a process from RTLinux to Linux and viceversa	SSSA

A04 The incorporation of Quality of Service Management into real-time systems

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
Resource reservation scheduling	Resource reservation scheduling real-time tasks	First implementation	SSSA
Resource reservation scheduling - overhead	Overhead of the scheduling module under Linux	Maximum duration of standard kernel primitives compared to unmodified Linux	SSSA
Resource reservation scheduling - isolation	Temporal isolation property for Linux processes	Ability to protect one RT process from the interference of the other non real-time processes	SSSA
Resource reservation scheduling – multi-processor	Support for multi-processor platforms	First implementation	SSSA
Resource reservation scheduling - reclaiming	Distribute spare bandwidth to executing processes	Amount of reclaimed spare time (average value) and fairness of the reclamation	SSSA
QoS Manager	Identifies temporal characteristics of a task and adjust its scheduling parameters	First implementation	SSSA
QoS Manager - stability	Properties of the feedback control algorithm	Stability of the system. Formal proof and experimental evaluation	SSSA
QoS Manager – average frame rate	Measured values on a multimedia application	Measured average frame rate	SSSA
QoS Manager – frame rate variance	Measured values on a multimedia application	Measured variance of frame rate	SSSA
QoS Manager – average jitter	Measured values on a multimedia application	Measured average jitter	SSSA
DMA	Dynamic Memory Allocation in RTLinux (malloc and free functions)	First implementation	UPVLC
DMA – response time	Dynamic Memory Allocation - response time	Independent response time on amount of allocated memory	UPVLC

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
DMA - overhead	Bounded overhead introduced by the Dynamic Memory Allocation component	< 200 nanoseconds Pentium III / 500 MHz	UPVLC

A05 Two levels of scheduling in the RTLinux operating system

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
CBS algorithm	Constant Bandwidth Server implementation at RTLinux level	First implementation	UPVLC
CBS algorithm - overhead	Overhead introduced by CBS server in the task scheduling	Same overhead than any other task in RTLinux	UPVLC
EDF algorithm	Implementation of the EDF (Earliest Deadline First) scheduling policy	First implementation as ADS example	UPVLC
ADS algorithm	A set of tasks can be scheduled using a Application-defined policy	First implementation	UPVLC
ADS algorithm - ADS overhead	Scheduled task overhead added by the ADS mechanism	Two context switches for each scheduler invocation	UPVLC
ADS algorithm - ADS overhead for kernel scheduled tasks	Overhead of the application-defined scheduling for non ADS threads	One “IF” instruction	UPVLC

A06 Fault tolerant components for real-time applications

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
Application FT monitor	Handles degraded mode management at application level on detection of abnormal event (timing error or thread abortion) by FT controller. Provides support for application mode change on faulty behaviour of an application thread.	First implementation	CEA
FT controller	Handles emergency actions on timing error or thread abortion (behaviour change + propagation of event to FT monitor). Activate an alternate threads with degraded behaviour.	First implementation	CEA
FT controller – reaction time reduction	Avoids creating a new thread to handle the situation.	Less context switch time than task creation time	CEA
Replica manager	Task replica manager - controlling of replicas (spare tasks)	First implementation	CEA
Redundancy manager	Task redundancy manager - activating / deactivating replicas	First implementation	CEA
FT design tool	Off-line specification of required fault-tolerant behaviour	First implementation	CEA
FT building tool	Configuration of required fault-tolerant behaviour	First implementation	CEA

A07 Predictability, fault-tolerance and high performance in communication

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
ORTE - RTPS 1.17	RT Ethernet - compliance to the norm RTPS v. 1.17	First implementation	CTU
ORTE - communication capacity	RT Ethernet - data throughput measures with NDDS 3.0	5000 messages per second (each message has 256 bytes) on 100Mbit/s Ethernet	CTU
ORTE - communication latency	RT Ethernet - application layer input / output reaction time	reaction time 5 ms on 100Mbit/s Ethernet with 1000 messages per second (each message has 128 bytes) transmitted from at least 3 nodes in one collision domain	CTU
ORTE - analyzer	RT Ethernet - real time Ethernet analyzer	New implementation	CTU
CanOpen - DS301	CanOpen - compliance to the norm DS301	New implementation	CTU
CanOpen - communication capacity	CAN VCA - ping-pong test of CAN API with a different bus load	2 ms for 50% loaded bus	CTU
CanOpen - communication latency	CanOpen - Process Data Object (PDO) reaction time	1 ms for the highest priority message (8 bytes), 1Mbit/s transmission rate	CTU
CanOpen - analyzer	Logging window with received messages, manual raw messages preparing and sending	New implementation	CTU
CanOpen - EDS parser	Comfortable reading and editing of device parameters	New implementation	CTU

A08 The incorporation of specific modes at the components level

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
User RM API	Set of libraries accessing Resource Management services, facilitates to change task environment (RTLinux or Linux)	First implementation	SSSA

A09 Adaptability to several HW configurations

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
PowerPC porting	RTLinux porting to a PowerPC processor	New implementation	UPVLC
StrongARM	RTLinux porting to a StrongARM processor	New implementation	UPVLC
CTU PowerPC	Porting HW dependent components to PowerPC	New implementation	CTU

A10 Support for a wide variety of applications with different level of criticality

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
SQM rules keeping	System Quality Management at SW development for non-critical applications. A possibility to extend rules of SQM for higher levels of criticality	ALL criteria in chapter "Criteria for SW life cycle documentation"	UC

A11 Scalability from a small embedded to a full-featured general purpose OS

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
Standalone RTLinux	RTLinux implementation without the Linux environment	First implementation	UPVLC
Standalone RTLinux - size	RTLinux implementation without the Linux environment minimal size	Less than 100 Kb	UPVLC

A12 More competitive Europe in the field of embedded systems

Parameter name	Parameter description	Qualitative or quantitative criterion	Responsible
Price	Just RTOS, which price is able to compete, is usable for repeated or serial production	Free	OCERA

Notice:

The criteria containing word “implementation” are qualitative (verifiable), the other criteria are quantitative (measurable).