



Evolution Management and Process for  
Real-Time Embedded Software Systems

# **Essentials and requisites for the management of evolution Part 5: Glossary**

Deliverable D1 part 5  
Edited by Peter Kaiser

31 March 2003  
Version 1.0  
Status final

Public Version



**I T E A**

INFORMATION TECHNOLOGY

FOR EUROPEAN ADVANCEMENT

This document is part of the work of the EUREKA  $\Sigma!$  2023 – ITEA 00103 project EMPRESS.  
Copyright © 2002-2003 EMPRESS consortium.

Authors/Partners:

Partner	Author	Email
Fraunhofer IESE, Germany	Peter Kaiser	Kaiser@iese.fhg.de
K.U.Leuven, Belgium	Stefan Van Baelen	Stefan.VanBaelen@cs.kuleuven.ac.be

Document History:

Date	Version	Editor	Description
31 Mar 03	1.0	Stefan Van Baelen	Public version based on internal version 1.0

Filename: D1.5\_v1.0\_Public\_Version.doc

## **Abstract**

This documents contains definitions and acronyms of importance for EMPRESS.

**Keywords:**

EMPRESS, glossary, acronyms

## Table of Contents

<b>1</b>	<b>Introduction.....</b>	<b>5</b>
<b>2</b>	<b>Glossary .....</b>	<b>5</b>
<b>3</b>	<b>Acronyms .....</b>	<b>16</b>
<b>4</b>	<b>Appendix .....</b>	<b>18</b>
4.1	Domain-specific glossary .....	18
4.2	Non-used definitions .....	18
4.3	Literature.....	19

1 Introduction

## 1 Introduction

To ease discussion between the many partners within EMPRESS it is important that all partners have the same understanding of the terms used. Just as important it is for readers outside the EMPRESS consortium to have precise definitions for all terms used within the EMPRESS deliverables. This glossary helps towards a common and unambiguous usage of terms as used within EMPRESS project.

For each term a definition and the source is given. If terms are defined in several sources (e.g. many terms are defined in both [IEEE] and [OMG-UML]), the most appropriate one for EMPRESS has been selected. Sometimes, if a definition is intentionally not used (e.g. we use the IEEE definition of “component”, not the UML definition), a rationale is given in appendix 0.

Besides software development specific terms in chapter 2 there are some domain specific terms in appendix 4.1.

## 2 Glossary

Term	Definition	Source
adaptability	See: flexibility	[IEEE]
architecture	See software architecture	
availability	The degree to which a system or component is operational and accessible when required for use. Often expressed as a probability.  See also: error tolerance; fault tolerance; robustness.	[IEEE]
certification	(1) A written guarantee that a system or component complies with its specified requirements and is acceptable for operational use. For example, a written authorization that a computer system is secure and is permitted to operate in a defined environment.  (2) A formal demonstration that a system or component complies with its specified requirements and is acceptable for operational use.  (3) The process of confirming that a system or component complies with its specified requirements and is acceptable for operational use.	[IEEE]

2 Glossary

Term	Definition	Source
class	A class is a description of a set of objects that share the same attributes, methods, relationships and semantics. A class may use a set of interfaces to specify collections of operations it provides to its environments.	[OMG-UML]
component	One of the parts that make up a system. A component may be hardware or software and may be subdivided into other components.  <i>Note:</i> The terms “module”, “component” and “unit” are often used interchangeably or defined to be subelements of one another in different ways depending upon the context. The relationship of these terms is not yet standardized.	[IEEE]
configuration management	A discipline applying technical and administrative direction and surveillance to: identify and document the functional and physical characteristics of a configuration item, control changes to those characteristics, record and report change processing and implementation status, and verify compliance with specified requirements.  <i>See also:</i> baseline; configuration identification; configuration control; configuration status accounting; configuration audit.	[IEEE]
container	An object that exists to contain other objects, and that provides operations to access or iterate over its contents.	[OMG-UML]
correctness	(1) The degree to which a system or component is free from faults in its specification, design, and implementation.  (2) The degree to which software, documentation, or other items meet specified requirements.  (3) The degree to which software, documentation, or other items meet user needs and expectations, whether specified or not.	[IEEE]
efficiency	The degree to which a system or component performs its designated functions with minimum consumption of resources.  <i>See also:</i> execution efficiency; storage efficiency.	[IEEE]

2 Glossary

Term	Definition	Source
embedded software	Software that is part of a larger system and performs some of the requirements of that system; for example, software used in an aircraft or rapid transit system.	[IEEE]
extendability	The ease with which a system or component can be modified to increase its storage or functional capacity.  <i>Syn:</i> expandability; extensibility.  <i>See also:</i> flexibility; maintainability.	[IEEE]
extensibility	<i>See:</i> extendability.	[IEEE]
extensible mark-up language	(XML) The Extensible Markup Language is a subset of SGML. Its goal is to enable generic SGML to be served, received, and processed on the Web in the way that is now possible with HTML. XML has been designed for ease of implementation and for interoperability with both SGML and HTML	[W3C]
fail safe	Pertaining to a system or component that automatically places itself in a safe operating mode in the event of a failure; for example, a traffic light that reverts to blinking red in all directions when normal operation fails.  <i>Contrast with:</i> fail soft.  <i>See also:</i> fault secure; fault tolerance.	[IEEE]
fault tolerance	(1) The ability of a system or component to continue normal operation despite the presence of hardware or software faults.  <i>See also:</i> error tolerance; fail safe; fail soft; fault secure; robustness.  (2) The number of faults a system or component can withstand before normal operation is impaired.  (3) Pertaining to the study of errors, faults, and failures, and of methods for enabling systems to continue normal operation in the presence of faults. <i>See also:</i> recovery; redundancy; restart.	[IEEE]

2 Glossary

Term	Definition	Source
flexibility	The ease with which a system or component can be modified for use in applications or environments other than those for which it was specifically designed. <i>Syn:</i> adaptability. <i>See also:</i> extendability; maintainability.	[IEEE]
framework	A framework is a partially complete software (sub-)system that is intended to be instantiated. It defines the architecture for a family of (sub-)systems and provides the basic building blocks to create them. It also defines the places where adaptations for specific functionality should be made. In an object-oriented environment a framework consists of abstract and concrete classes.	[Bus]
function	A task, action, or activity that must be accomplished to achieve a desired outcome.	[IEEEb]
functional requirement	A requirement that specifies a function that a system or system component must be able to perform. <i>Contrast with:</i> design requirement; implementation requirement; interface requirement; performance requirement; physical requirement.	[IEEE]
integrity	The degree to which a system or component prevents unauthorized access to, or modification of, computer programs or data.	[IEEE]
maintainability	(1) The ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment. <i>See also:</i> extendability; flexibility. (2) The ease with which a hardware system or component can be retained in, or restored to, a state in which it can perform its required functions.	[IEEE]
manufacturer	Organization receiving sub-systems (maybe COTS software) from a supplier and integrating it into a product. <i>Note:</i> typically used in the automotive domain	

2 Glossary

Term	Definition	Source
mean time between failures	(MTBF) The expected or observed time between consecutive failures in a system or component.  <i>See also:</i> up time.	[IEEE]
mean time to repair	(MTTR) The expected or observed time required to repair a system or component and return it to normal operations.  <i>See also:</i> down time.	[IEEE]
methodology	<i>Synonym for process</i>	
model driven architecture	(MDA) The MDA defines an approach to IT system specification that separates the specification of system functionality from the specification of the implementation of that functionality on a specific technology platform. [...] The MDA approach and the standards that support it allow the same model specifying system functionality to be realized on multiple platforms through auxiliary mapping standards, or through point mappings [...]	[OMG-MDA]
modularity	The degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.  <i>See also:</i> cohesion; coupling.	[IEEE]
nonfunctional requirement	In software system engineering, a software requirement that does not describe what the software will do, but how the software will do it, for example, software performance requirements, software external interface requirements, software design constraints, and software quality attributes. Nonfunctional requirements are sometimes difficult to test; therefore, they are usually evaluated subjectively.  <i>Contrast with</i> functional requirement.	[Tha]
pattern	A common solution to a common problem in a given context	[JBR]
performance requirement	A requirement that imposes behavioral conditions on functional requirements, such as the speed, throughput, response time, and memory usage.  <i>See</i> functional requirement, requirement.	[JBR]

2 Glossary

Term	Definition	Source
planned evolution	<i>Synonym for anticipated change</i>	
platform independent models	(PIM) The PIM provides formal specifications of the structure and function of the system that abstracts away technical details.	[OMG-MDA]
platform specific models	(PSM) A PSM is expressed in terms of the specification model of the target platform. PSM have to use the platform concepts of exception mechanisms, parameter types (including platform-specific rules about objects references, value types, semantics of call by value, etc.), and component model.	[OMG-MDA]
portability	The ease with which a system or component can be transferred from one hardware or software environment to another.  <i>Syn:</i> transportability.  <i>See also:</i> machine independent.	[IEEE]
process	A sequence of steps performed for a given purpose; for example, the software development process.	[IEEE] <sup>1</sup>
quality assurance	(QA)  (1) A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements.  (2) A set of activities designed to evaluate the process by which products are developed or manufactured.  <i>Contrast with:</i> quality control (1).	[IEEE]
quality management	The planned actions taken to ensure the effective implementation of an organization's quality related organizational structures, procedures, processes and resources.	
quality attribute	<i>Synonym for non-functional requirement</i>	

---

<sup>1</sup> This is the first of three definitions, which has been selected for the purposes of EMPRESS

2 Glossary

Term	Definition	Source
real time	<p>Pertaining to a system or mode of operation in which computation is performed during the actual time that an external process occurs, in order that the computation results can be used to control, monitor, or respond in a timely manner to the external process.</p> <p><i>Contrast with:</i> batch.</p> <p><i>See also:</i> conversational; interactive; interrupt; on-line.</p>	[IEEE]
reliability	<p>The ability of a system or component to perform its required functions under stated conditions for a specified period of time.</p> <p><i>See also:</i> availability; MTBF.</p> <p><i>Note:</i> Can, for example, be measured in terms of system availability, accuracy, mean time between failures, defects per 1,000 lines of code (KLOC), and defects per class ([JBR])</p>	[IEEE]
requirement	<p>(1) A condition or capability needed by a user to solve a problem or achieve an objective.</p> <p>(2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.</p> <p>(3) A documented representation of a condition or capability as in (1) or (2).</p> <p><i>See also:</i> design requirement; functional requirement; implementation requirement; interface requirement; performance requirement; physical requirement.</p>	[IEEE]
requirements engineering	<p>The purpose of Requirements Engineering is to produce and analyse customer, product, and product component requirements.</p>	[SEI]
requirements management	<p>The purpose of Requirements Management is to manage the requirements of the project's products and product components and to identify inconsistencies between those requirements and the project's plans and work products</p>	[SEI]

2 Glossary

Term	Definition	Source
response time	The elapsed time between the end of an inquiry or command to an interactive computer system and the beginning of the system's response.  <i>See also:</i> port-to-port time; think time; turnaround time.	[IEEE]
reusability	The degree to which a software module or other work product can be used in more than one computer program or software system.  <i>See also:</i> generality.	[IEEE]
robustness	The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions.  <i>See also:</i> error tolerance; fault tolerance.	[IEEE]
sizing	The process of estimating the amount of computer storage or the number of source lines required for a software system or component.  <i>Contrast with:</i> timing.	[IEEE]
software	Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.  <i>See also:</i> application software; support software; system software.  <i>Contrast with:</i> hardware.	[IEEE]
software architecture	The software architecture of a program or a computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.	[Bas]
software engineering	(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.  (2) The study of approaches as in (1).	[IEEE]
software process engineering metamodel	(SPEM) This metamodel is used to describe a concrete software development process or a family of related software development processes.	[OMG-SPEM]

2 Glossary

Term	Definition	Source
sub-system	A sub-system is a grouping of model elements that represent a behavioral unit in a physical system. A sub-system offers interfaces and has operations. In addition, the model elements can be partitioned into realization and specification elements, where the latter, with the system operations, is realized (i.e. implemented) by the former.	[OMG-UML]
supplier	Organization delivering sub-systems (maybe COTS software) to a manufacturer  <i>Note:</i> typically used in the automotive domain	
traceability	The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor or master-subordinate relationship to one another; e.g., the degree to which the requirements and design of a given system element match.	[IEEE]
unified modelling language	(UML): The Unified Modelling Language is a language for specifying, visualizing, constructing, and documenting the artefacts of software systems, as well as for business modelling and other non-software systems. The UML represents a collection of the best engineering practices that have proven successful in the modelling of large and complex systems.	[OMG-UML]
usability	The ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component.	[IEEE]

2 Glossary

Term	Definition	Source
validation	<p>The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.</p> <p><i>Contrast with:</i> verification.</p> <p><i>Note:</i></p> <p>“Do we build the right product?” The developers (and the customer) examine if the development results in the product the customer desires. The primary focus of validation is customer satisfaction.</p> <p>Validation checks a solution against the requirements as stated by the customer. The aim is to prove that the solution is adequate regarding the customer's requirements. Validation requires considering real application conditions of the object under analysis. Thus, validation includes that the contractors, the developers, check whether they are developing exactly the product ordered by the customer. Validation is mostly carried out by testing. The customer should participate in validation, at least in acceptance testing. [DESS]</p>	[IEEE]

2 Glossary

Term	Definition	Source
verification	<p>The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.</p> <p><i>Contrast with:</i> validation.</p> <p><i>Note:</i></p> <p>“Do we build the product right?” The developers check whether they are working properly during the development.</p> <p>Verification is intended to check one development result against a previous one. For instance, an implemented, embedded subsystem against an architecture description, which was developed during design. Another example is checking a model that has been developed during design against requirements of an earlier development phase. In case a development step is being automated and in case this transformation has been proven to be accurate, verification may be simplified or even be omitted. Customer participation is mandatory for validation. Concerning verification, his participation will be certainly a useful support. Basically: the more the customer and his requirements are involved in the analysis, the sooner the quality aimed at can be achieved (catchword: customer satisfaction based V&amp;V). [DESS]</p>	[IEEE] <sup>2</sup>
verification and validation	<p>(V&amp;V) The process of determining whether the requirements for a system or component are complete and correct, the products of each development phase fulfill the requirements or conditions imposed by the previous phase, and the final system or component complies with specified requirements.</p> <p><i>See also:</i> independent verification and validation.</p>	[IEEE]

---

<sup>2</sup> The IEEE standard lists two definitions; for EMPRESS we look at the first one only; see 0 for the second definition

3 Acronyms

### 3 Acronyms

ADT	Abstract Data Types
ATAM	Architecture Trade-off Analysis Method
CASE	Computer Aided Software Engineering
CBD	Component-Based Development
CBSE	Component-Based Software Engineering
CCA	Component Collaboration Architecture
CCB	Change Control Board
CCM	CORBA Component Model
CCOTS	Complex COTS
ChgM	Change Management
CMM	Capability Maturity Model
CnfM	Configuration Management
COBRA	Common Object Request Broker Architecture
COM	Component Object Model
COTS	Commercial Of The Shelf software
DBC	Design By Contract
DCOM	Distributed COM
DD	Defect Density
ECU	Electronic Control Unit
EJB	Enterprise Java Beans
ER	Entity Relationship
FODA	Feature Oriented Domain Analysis
FP	Function Points
FR	Functional Requirement
GQM	Goal-Question-Metric
HTML	Hyper Text Markup Language
IDL	Interface Definition Language
JAR	Java ARchive
KLOC	Kilo Lines Of Code (1,000 lines of code)
KobrA	Komponenten-basierte Anwendungsentwicklung
LOC	Lines Of Code

### 3 Acronyms

MDA	Model Driven Architecture
MTBF	Mean Time Between Failures
MTS	Microsoft Transaction Service
MTTR	Mean Time To Repair
NFR	Non-Functional Requirement
OCL	Object Constraint Language
OMA	Object Management Architecture
OMG	Object Management Group
OOP	Object-Oriented Programming
PIM	Platform Independent Models
PLC	Programmable Logic Controller
POA	Portable Object Adapter
PSM	Platform Specific Models
QA	Quality Assurance
QoS	Quality Of Service
RE	Requirements Engineering
RM	Requirements Management
RMI	Remote Method Invocation
RUP	Rational Unified Process
SAAM	Software Architecture Analysis Method
SCADA	Supervisory Control And Data Acquisition
SCM	Software Configuration Management
SGML	Standard Generalized Markup Language
SOAP	Simple Object Access Protocol
SPEM	Software Process Engineering Metamodel
SPICE	Software Process Improvement and Capability dEtermination
UML	Unified Modeling Language
V&V	Verification & Validation
XML	Extensible Mark-up Language

4 Appendix

## 4 Appendix

### 4.1 Domain-specific glossary

Term	Description	Source
programmable logic controller	(PLC) Industrial electronic device for controlling the automatisms of a Metal Processing Line.	Industry automation
Supervisory Control and Data Acquisition	(SCADA): SW for graphical applications for HMI (Human to Machine Interface) communication, process control and supervision.	Industry automation
metal processing lines	Set of machines and control devices (PLC-s and Drives) that compose a system for metal transforming: the input to the line is a steel coil, whereas the output may be a set of sub-coils in the case of a slitting line, or a set of steel sheets in the case of a cut-to-length line.	Industry automation

Sources:

Industry automation

MSI

### 4.2 Non-used definitions

Term	Description <i>reason for not using the definition</i>	Source
requirement	A condition or capability to which a system must conform.  <i>Not used because missing the distinction between user and system requirements</i>	[JBR]
component	A physical and replaceable part of a system that conforms to and provides the realization of a set of interfaces.  <i>Not used as it only addresses physical components</i>	[OMG-UML]
verification	(2) Formal proof of program correctness. See: proof of correctness.  <i>Not used as in EMPRESS we are only dealing with the aspect of verifying the output of a development phase</i>	[IEEE]

4 Appendix

### 4.3 Literature

- [IEEE] IEEE Standard Glossary of Software Engineering Terminology, IEEE Std 610.12-1990
- [IEEEb] IEEE Guide for Developing System Requirements Specifications, IEEE Std 1233 - 1998
- [IEEEc] IEEE Recommended Practice for Software Requirements Specifications, IEEE Std 830 -1998
- [OMG-UML]  
OMG, "OMG Unified Modelling Language Specification", Version 1.3, June 1999, <http://www.omg.org/>
- [OMG-MDA]  
OMG, "Model driven architecture", ormsc/2001-07-01, July 2001, <http://www.omg.org/>
- [W3C] Tim Bray, Jean Paoli, "Extensible Markup Language (XML) 1.0", W3C, February 1998, <http://www.w3.org/>
- [OMG-SPEM]  
OMG, "Software Process Engineering Management, The Software Process Engineering Metamodel (SPEM)", ad/2001-06-05, July 2001, <http://www.omg.org/>
- [DESS] ITEA project DESS, The DESS methodology, Deliverable D1 V01-publid, December 2001
- [JBR] I. Jacobson, G. Booch, J. Rumbaugh, "The Unified Software Development Process". Addison Wesley, Reading, Massachusetts, 1999.
- [SEI] CMMI-SE/SW, V1.02. SEI, 2000
- [Tha] R.H. and M.C. Thayer: Software Requirements Engineering Glossary. In: R.H. Thayer and M. Dorfman (ed.) : Standards, Guidelines, and Examples on System and Software Requirements Engineering, IEEE Computer Society Press, 1994
- [Bas] L. Bass, P. Clements, R. Kazman, "Software Architecture in Practice". Addison-Wesley, 1998
- [Bus] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, "Pattern - Oriented Software Architecture: A System of Patterns", John Wiley & Sons Ltd., 1996.