CoreLinux++ Requirements Index

The Corelinux Consortium

Revision: 1.7
Created on May 8, 2000
Revised on September 5, 2000

Abstract

This document is the `Corelinux++` development process details.

Contents

Copyright notice

1 Overview

This document contains links to CoreLinux++ Requirement Documents. As requirements are received and approved, as described in the CoreLinux++ Process document, they will be added to this index. From the requirements document you can link to the current Analysis and or Design documentation for that requirement once they are available.

1.1 Functional Requirements (libcorelinux++)

1. Foundation classes
2. Inter-process communication
3. Process and threads
4. Design Patterns

(a) Creational Design Patterns
(b) Structural Design Patterns
(c) Behavioral Design Patterns

1.2 Foundation Classes

1. Requirement 4568 String - *In Analysis*
2. Requirement 20873 Memory - *In Implementation*
3. Requirement 22377 Environment - *In Implementation*

1.3 Inter-process communication

1. Requirement 1589 Semaphores - *In Implementation*

1.4 Process and threads

1. Requirement 1588 Thread - *In Implementation*

1.5 Design Patterns

1. **Creational**
   (a) Requirement 5095 Abstract Factory Pattern - *Completed*
   (b) Requirement 5096 Builder Pattern - *Completed*
   (c) Requirement 5097 Factory Method Pattern - *Completed - implemented as AbstractAllocator in AbstractFactory*
   (d) Requirement 5098Prototype Pattern - *Completed*
   (e) Requirement 5099 Singleton Pattern - *Completed*
2. **Structural**
   (a) Requirement 2872 Adapter Pattern - *Completed*
   (b) Requirement 3142 Bridge Pattern - *Completed*
   (c) Requirement 3143 Composite Pattern - *Completed*
   (d) Requirement 3144 Decorator Pattern - *Completed*
   (e) Requirement 3145 Facade Pattern - *Completed*
   (f) Requirement 3146 Flyweight Pattern - *Completed*
   (g) Requirement 3147 Proxy Pattern - *Completed*
3. **Behavioral**
   (a) Requirement 5972 Iterator Pattern - *Completed*
   (b) Requirement 6587 Chain of Responsibility Pattern - *Completed*
   (c) Requirement 6588 Command Pattern - *Completed*

(d) Requirement 6590 Mediator Pattern - *Completed*

(e) Requirement 6591 Memento Pattern - *Completed*

(f) Requirement 6592 Subject/Observer Pattern - *Completed*

(g) Requirement 6594 State Pattern - *Completed*

(h) Requirement 6595 Strategy Pattern - *Completed, shares design with AbstractFactory*

(i) Requirement 6598 Template Method Pattern - *Completed, used throughout library*

(j) Requirement 6599 Visitor Pattern - *Completed, shares Design with Composite*

2 Functional Requirements (libclfw++)

1. Core Framework
2. Framework Abstractions

2.1 Core Framework

1. Requirement 10658 MetaClass - *In Analysis*

2.2 Framework Abstractions

1. Requirement 4865 Library Load - *Completed*

References

Grady Booch. *Object Oriented Analysis and Design With Applications*. Benjamin/Cummings, Redwood City, CA, 2nd edition, 1994.

The Corelinux Consortium. *The Corelinux C++ Coding Standards*. The Corelinux Consortium, 1.3 edition, May 2000a. http://corelinux.sourceforge.net/cppstnd.php.

The Corelinux Consortium. *The Corelinux Object Oriented Design Standards*. The Corelinux Consortium, 1.3 edition, May 2000b.

Phillip B. Crosby. *Quality Is Free*. McGraw-Hill, New York, NY., 10020, 1976.

FSF. *GNU Autoconf Manual*. FSF, 2.13 edition, 1999.

FSF. *GNU Automake Manual*. FSF, 1.4 edition, 2000a.

FSF. *GNU Libtool Manual*. FSF, 1.3.4 edition, 2000b.

D.E. Knuth. Structured programming with goto's. *ACM Computing Surveys*, Vol 6(No. 4), December 1974.

Bertrand Meyer. *Object Oriented Software Construction*. Prentice Hall, Englewood Cliffs, NJ, 1988.

Inc. Taligent. *The Taligent Guide to Well-Mannered Object-Oriented Programs*. Taligent Inc., Cupertino, CA., 1994.