

SOFTWARE COMMUNICATIONS ARCHITECTURE SPECIFICATION

APPENDIX E-2 PLATFORM SPECIFIC MODEL (PSM) - C++



28 February 2012
Version: 4.0

Prepared by:

JTRS Standards
Joint Program Executive Office (JPEO) for the Joint Tactical Radio System (JTRS)
33000 Nixie Way
San Diego, CA 92147-5110

Statement A - Approved for public release; distribution is unlimited (01 April 2012)

REVISION SUMMARY

Version	Revision	Date
Next <Draft>	Initial Draft Release	30 November 2010
Candidate Release	Initial Release	27 December 2011
4.0	ICWG Approved Release	28 February 2012

TABLE OF CONTENTS

APPENDIX E-2	PSM – C++	4
E-2.1	SCOPE	4
E-2.2	CONFORMANCE	4
E-2.3	CONVENTIONS	4
E-2.4	NORMATIVE REFERENCES	4
E-2.5	INFORMATIVE REFERENCES	4
E-2.6	UML TO C++ PSM MAPPING	4
E-2.7	IDL TO C++ PSM MAPPING	5
E-2.8	ATTACHMENTS	5

APPENDIX E-2 PSM – C++**E-2.1 SCOPE**

This appendix defines the platform specific technology model using C++.

E-2.2 CONFORMANCE

See SCA Appendix E.

E-2.3 CONVENTIONS

N/A

E-2.4 NORMATIVE REFERENCES

N/A

E-2.5 INFORMATIVE REFERENCES

The following is a list of documents referenced within this appendix or used as reference or guidance material in its development.

- [1] C++ Language Mapping, Version 1.2 formal/2008-01-09, January 2008.

E-2.6 UML TO C++ PSM MAPPING

The SCA technology specific mapping to C++ consists of header files that are based upon C++ Language Mapping [1]. The UML to C++ transformation rules are not universal rules for creating any technology specific mapping, but only for the purpose of this appendix. This section defines a non-normative reference mapping, a summary of which is represented in Table 1. The rule set for transforming UML packages, interfaces, types, and exceptions into C++ constructs are still being finalized.

Table 1: UML to C++ PSM Mapping

UML Representation	C++ Representation
Interface	Class, which within the framework will inherit the Object representation
exception	Exception mechanism with exceptions mapped to classes
Object	An abstract class whose standard operations are constructors, copy constructors, destructors, and assignment operators
Primitive Types (integer, Boolean, string, unlimited natural), others are constructed	Primitive Types (void, int, float, double, char, bool)

28 February 2012

by layering formatting or constraints on top of primitives	modifiers: unsigned, long, short
Any	Approach modeled on boost library ¹ any definition
Struct	struct
Sequence	Pointer list of corresponding type
Type	typedef
Package	namespace
No return value from an operation	void
Enumeration	enum
A constrained integer data type that corresponds to an Octet	Unsigned char
Attribute	Member variable

E-2.7 IDL TO C++ PSM MAPPING

IDL is the standard representation for the SCA independent model. The C++ PSM representation is still being finalized.

The transformations expressed within the table represent the framework for the header files that are being developed. The objective of the mapping is to develop a native C++ representation that incorporates optimizations required for an efficient implementation.

A simplistic mapping would translate the UML constructs into their equivalent C++ elements however that would provide little guidance for the resultant implementation. Having a complete alignment with the “C++ Language Mapping” would provide a comprehensive, Standards driven solution but it would also introduce constructs that are not required for a native C++ implementation. The desired mapping will include not only the requisite mappings, but also the necessary set of C++ elements (e.g. constructors and support operations) that will provide the basis for a product to be implemented. The header files and mapping rules will be added to this specification once they are completed.

E-2.8 ATTACHMENTS

N/A

¹ <http://www.boost.org/>