IDL Profiles for Platform-Independent Modeling of SDR Applications

Document WINNF-14-S-0016

Version V2.0.1
12 June 2015

TERMS, CONDITIONS & NOTICES

This document has been prepared by the work group of WInnF project SCA-2014-001 "Revision of PIM IDL Profiles V1" to assist The Software Defined Radio Forum Inc. (or its successors or assigns, hereafter "the Forum") in evolving previous version V1.0.1. It may be amended or withdrawn at a later time and it is not binding on any member of the Forum or of the work group of WInnF project SCA-2014-001.

Contributors to this document that have submitted copyrighted materials (the Submission) to the Forum for use in this document retain copyright ownership of their original work, while at the same time granting the Forum a non-exclusive, irrevocable, worldwide, perpetual, royalty-free license under the Submitter's copyrights in the Submission to reproduce, distribute, publish, display, perform, and create derivative works of the Submission based on that original work for the purpose of developing this document under the Forum's own copyright.

Permission is granted to the Forum's participants to copy any portion of this document for legitimate purposes of the Forum. Copying for monetary gain or for other non-Forum related purposes is prohibited.

THIS DOCUMENT IS BEING OFFERED WITHOUT ANY WARRANTY WHATSOEVER, AND IN PARTICULAR, ANY WARRANTY OF NON-INFRINGEMENT IS EXPRESSLY DISCLAIMED. ANY USE OF THIS SPECIFICATION SHALL BE MADE ENTIRELY AT THE IMPLEMENTER'S OWN RISK, AND NEITHER THE FORUM, NOR ANY OF ITS MEMBERS OR SUBMITTERS, SHALL HAVE ANY LIABILITY WHATSOEVER TO ANY IMPLEMENTER OR THIRD PARTY FOR ANY DAMAGES OF ANY NATURE WHATSOEVER, DIRECTLY OR INDIRECTLY, ARISING FROM THE USE OF THIS DOCUMENT.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

This document was developed following the Forum's policy on restricted or controlled information (Policy 009) to ensure that that the document can be shared openly with other member organizations around the world. Additional Information on this policy can be found here: http://www.wirelessinnovation.org/page/Policies_and_Procedures .

Although this document contains no restricted or controlled information, the specific implementation of concepts contain herein may be controlled under the laws of the country of origin for that implementation. Readers are encouraged, therefore, to consult with a cognizant authority prior to any further development.

Wireless Innovation Forum ™ and SDR Forum ™ are trademarks of the Software Defined Radio Forum Inc.

Table of Contents

List of Figures

List of Tables

Executive Summary

This WInnF standard addresses specification of IDL Profiles setting designers' possibility for specification of application-specific interfaces of SDR Applications Components in a PIM (Platform Independent Model) way.

It has been initially elaborated by the Wireless Innovation Forum as a contribution to SCA 4.1 development, and provides a standard solution for a broad range of actors of the SDR eco-system, regardless if the SCA or CORBA are being used.

The normative content defines one "Full" and one "Ultra-Lightweight" / "ULw" PIM IDL Profiles, that set the range of possible IDL declarations attached to specification of PIM interfaces.

The Full profile maximizes the range of possible IDL declarations and types for PIM specification of application-specific interfaces, while the ULw profile further narrows the allowed possibilities to a minimal set that only fits the essential needs of processing-intensive embedded components, such as components of physical layers of waveform applications.

The support content section provides a tabular overview of the profiles contents in front of the referenced IDL Standard of the OMG, and explanations regarding possible evolutions of the standard.

The rationale section explains the applied development paradigm, and provides a detailed rationale for the choices, on a detailed feature-by-feature basis.

This WInnF standard took into account and harmonized two publicly available specifications: the ESSOR Architecture Common IDL Profile for DSP and FPGA and the SCA 4.0.1 CORBA ULw CORBA PSM.

Starting with SCA 4.1, it is expected that this WInnF standard will be broadly used within the international SDR eco-system, better supporting SDR Applications portability.

Revision history

Version	Description	Publication date
V2.0.1	**Editorial improvements** Reflecting adjudication of Issue 219: propositions of improvements in the non-normative contents, in perspective of a normative referencing of the specification by SCA 4.1. Reflecting the status of WInnF SDR Standard (denomination, Issues collection, …).	**12-Jul-2015**
V2.0.0	**Revised release** *Product of WInnF revision project SCA_2014_001 "Revision of PIM IDL Profiles V1".* Adjudication of Issue 26 and 27: - Issue 26: type "any" added to Full profile, - Issue 27: tag "RES" replaced by "IN" in summary tables. Conformance clauses improved.	**10-Feb-2015**
V1.0.1	**Editorial fixes** Adjudication of Issue 20: remaining revision marks removed.	**07-Oct-2014**
V1.0.0	**Initial release** *Product of WInnF project SCA_2013_002 "IDL Profiles Improvements for SCA 4.1".*	**25-Aug-2014**

Issues collection and processing

This document is a WInnF standard. Issues can be posted using the on-line Issues Submission Form available on the Wireless Innovation Forum website.

Contributors

- **Aeroflex / Cobham**, with Dave Hagood as main contributor (V1.0.0, V2.0.0),
- **Harris Corporation**, with Dave Carlson as main contributor (V1.0.0), and Ken Dingman (V2.0.1),
- **Hitachi Kokusai Electric**, with Dave Murotake as main contributor (V1.0.0, V2.0.0),
- **MITRE**, with Kevin Richardson, leader of SCA 4.1 elaboration, as main contributor (V1.0.0, V2.0.0, V2.0.1),
- **NordiaSoft**, with François Levesque as main contributor (V1.0.0),
- **Objective Interface Solutions**, with Charles Rush as main contributor (V1.0.0),
- **Prismtech**, with Ted Poole as main contributor (V1.0.0),
- **Raytheon**, with Jerry Bickle as main contributor (V1.0.0),
- **SELEX**, with Fabio Casalino as main contributor (V1.0.0),
- **THALES Communications & Security**, with Eric Nicollet as main contributor (V1.0.0, V1.0.1, V2.0.0, V2.0.1).

IDL Profiles for Platform-Independent Modeling of SDR Applications

1 Introduction

1.1 Aim of the specification

1.1.1 Overview

This specification defines two profiles of the Object Management Group (OMG) Interface Definition Language (IDL), which enable Platform Independent Modeling (PIM) of Software Defined Radio (SDR) applications. The two profiles are the Full PIM IDL Profile (Full) and the Ultra-Lightweight PIM IDL Profile (ULw).

Each PIM IDL Profile identifies a constrained set of IDL features for use in the definition of *application-specific interfaces*. Such interfaces are defined as the interfaces which are **specifically defined** between components of the SDR application.

Compliance with the profiles enhances portability of SDR applications across families of target operating environments and associated tools.

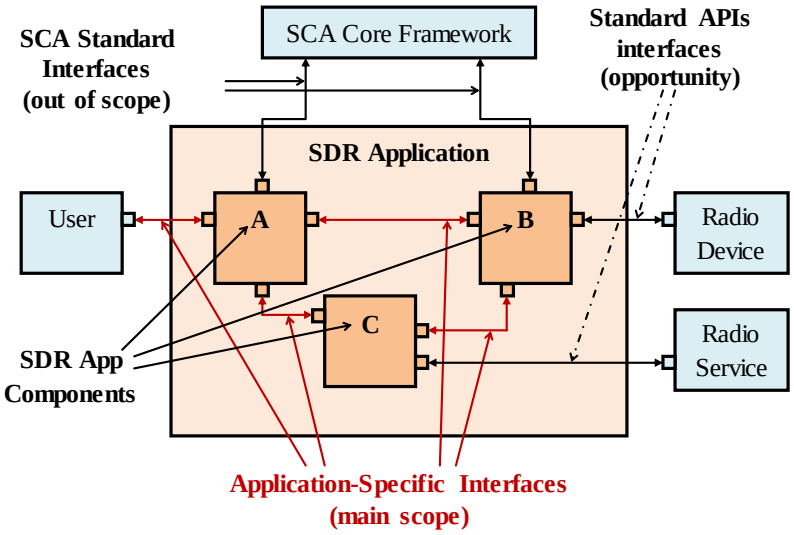


Figure 1 Representation of Application Specific Interfaces

The PIM IDL Profiles can also be used for definition of standard APIs interfaces for Radio Devices and Radio Services. Such choice **exclusively belongs** to developers of standard APIs for radio devices and radio services.

The PIM IDL Profiles do not address or impact the definition of interfaces between the components of the SDR application and the component management framework of the SDR product, such as the Software Communications Architecture (SCA) Core Framework standard interfaces.

The PIM IDL Profiles do not address or impact the definition of the software interfaces of the Operating Environment (OE), such as interfaces to the Operating Systems or Connectivity Mechanisms.

1.1.2 PIM and PSM Considerations for Improved Portability

1.1.2.1 PIM-level and PSM-level engineering stages

At the PIM stage of engineering, design considerations are made without any assumptions regarding any subsequent decisions that will be made during the implementation or integration phases. The PSM engineering stages cover the engineering activities where product implementation related decisions are made.

Through the usage of a PIM IDL Profile, application-specific interfaces can be specified during PIM engineering without making any assumption specific to the later PSM stages, in particular the programming language and the available connectivity.

The following PSM-level programming languages are typically used within distributed embedded systems:

- C and C++ for instruction-set processors, such as General Purpose Processors (GPPs) and Digital Signal Processors (DSPs),
- VHDL and Verilog for programmable logic processors, such as Field Programmable Gate Arrays (FPGAs).

The PSM-level connectivity can be *integrated*, where engineering tools provide automatic code generators to implement connections between components (e.g. utilizing CORBA or non-CORBA brokers), or *raw*, where the developer implements the connection between components utilizing a lower level connectivity mechanism. Inter-Process Communication mechanisms (see entry-level information at **[Ref7]**) can be used when components are co-localized.

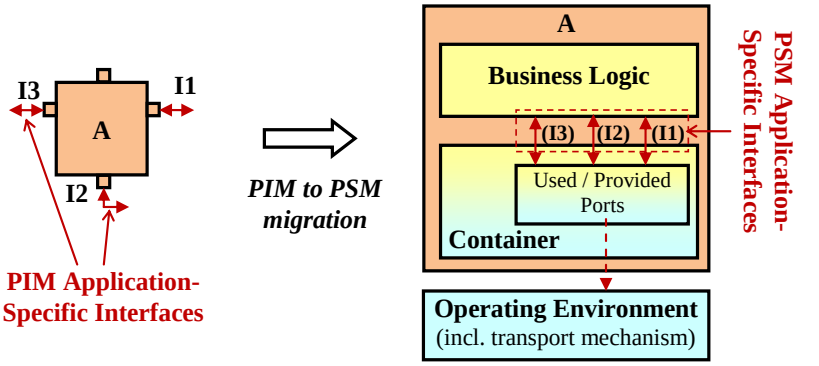
The specification does not mandate a connectivity mechanism, leaving the choice to user.

Usage of standard *connectivity* mechanisms can bring major additional portability benefits, for instance through usage of:

- CORBA (see **[Ref4]**), which provides *integrated* standard connectivity,
- JTNC MOCB (see **[Ref5]**) or MHAL Communication Service (see **[Ref6]**), which provide *raw* standard connectivity.

1.1.2.2 PIM to PSM migration

Design of PSM application specific interfaces is based on the PIM application specific interfaces and a PIM-to-PSM migration strategy.



Convention: "(I)" represents the PSM interface derived from PIM interface "I"

Figure 2: Notional PIM to PSM migration

As shown in

Figure 2, PSM application-specific interfaces separate the component's business logic from its ports within the component container. The port realization encapsulates any mapping or transformation which occurs between the PSM application specific interfaces and the OE transport mechanism. The separation allows the business logic to be developed largely independent of platform or porting concerns, thus maximizing its portability.

The overall portability of the business logic depends on a variety of factors which are outside the scope of this specification, such as the PIM to PSM interface mapping, the business logic implementation choices (e.g. whether or not it is POSIX AEPs compliant) and the full design pattern (a.k.a. component model) applied between it and the container.

The application of formal mapping rules, when available, from IDL to the target programming language (e.g. OMG IDL to language mapping standards) supports the use of tools to automate uniform definition of PSM application specific interfaces.

1.1.3 Development context and applicability

The initial design objectives for this specification were to improve and consolidate, in perspective of supporting the SCA 4.1 development, two pre-existing specifications: the ULw CORBA Profile defined in Appendix E-1 of SCA 4.0.1 (see **[Ref2]**) and the Common IDL Profile for DSP and FPGA of the ESSOR Architecture (see **[Ref3]**).

The PIM IDL Profiles do not present any SCA-specific dependency, so they can be used by any interested user within the SDR eco-system.

The IDL constructs referenced by this specification are standardized within "chapter 7" of the OMG CORBA v3.2 standard, as published by the Object Management Group (see **[Ref1]**).

Since the PIM IDL Profiles are specifically defined to ensure full independence from CORBA technology, they do not impose any CORBA-specific dependency.

1.2 Implementation Considerations

A large variety of *SDR Application artifacts* or *SDR Engineering Tools* can be conformant with this specification.

Conformant *SDR Application artifacts* can be present at any PIM or PSM stage of the development cycle, for instance PIM modeling, interface requirement specification, software interface design, component source code development or executable code production.

Conformant *SDR Engineering Tools* can support development of any conformant SDR Application Artifacts, at PIM or PSM stage, through container code generation or code verification.

1.3 Conformance

1.3.1 Applicable PIM Profile

Users of the specification have to identify the *applicable PIM Profile* for their conformant work products.

1.3.2 SDR Application Artifacts Conformance

Conformance evaluation of SDR Application Artifacts relies on the evaluation of Application-Specific Interfaces.

Conformance criteria for SDR Components and SDR Applications are derived from those elementary evaluations.

Conformance criteria for an Application-Specific Interface

An Application-Specific Interface is *conformant* with one *applicable IDL Profile* if each of its operations **exclusively uses** capabilities of the *applicable IDL Profile*.

No compliance criterion is defined for Application-Specific Interface.

Conformance criteria for an SDR Component

A SDR Component is *conformant* with one *applicable IDL Profile* if **all of its** Application-Specific Interfaces are *conformant* with the *applicable IDL Profile*.

A SDR Component is *compliant* with one *applicable IDL Profile* if **at least one** of its Application-Specific Interfaces is *conformant* with the *applicable IDL Profile*.

Conformance criteria for an SDR Application

A SDR Application is *conformant* with one *applicable IDL Profile* if **all of its** SDR Components are conformant with the *applicable IDL Profile*.

A SDR Component is *compliant* with one *applicable IDL Profile* if **at least one** of its SDR Component is *compliant* with the *applicable IDL Profile*.

Verification procedures

The specification of *verification procedures* for the evaluation of conformance of *SDR Applications Artifacts* is beyond the scope of this standard.

1.3.3 Engineering Tool Conformance

Conformance criteria

An Engineering Tool is *conformant* with one *applicable IDL Profile* if it **supports production of SDR Application artifacts** that are *conformant*, as SDR Application, with the applicable *IDL Profile*.

Verification procedures

The specification of *verification procedures* for the evaluation of conformance of *Engineering Tools* is beyond the scope of this standard.

The establishment of verification procedures for engineering tools depends on the existence of verification procedures for SDR applications artifacts.

1.3.4 Benefits of conformance

1.3.4.1 Conformance of SDR Applications

For an *SDR Application*, conformance with an *applicable IDL Profile* facilitates the design of *SDR Applications* with high degree of independence from implementation choices. Depending on the work product, this independence can result in several benefits related to portability improvement.

Conformance enables improved design of SDR Applications which are distributed across several processors, and is even more beneficial with a collection of heterogeneous processors and/or programming languages.

Conformance allows developed components to be independent of the underlying transport mechanisms available in the OE. This supports better portability of SDR components.

Conformance facilitates replacing one implementation of a SDR Application component with another implementation, while maintaining the overall consistency of the SDR Application design. This supports better overall portability of SDR applications

Conformance supports the development of SDR Engineering Tools which can enhance the speed, accuracy and quality of SDR Application software development.

1.3.4.2 Conformance of SDR Engineering Tools

Conformant SDR Engineering Tools provide *SDR Application* designers and software developers with code generation and automated model transformation features that can bring considerable productivity improvements. A number of features that can be supported by conformant SDR Engineering Tools are identified below, for illustration purposes.

A conformant tool can **enforce** or **evaluate** compliancy of interface-specific designs with the applicable PIM IDL Profile.

A conformant tool can **automatically generate** the **header files** associated with the interfaces defined at PIM level (e.g. using standard mapping rules such as those defined by the OMG).

A conformant tool can **automatically generate** the **used / provided ports** of the container software, providing faster encapsulation of the components business logic on top of the OE.

A conformant tool can **automatically generate** the **implementation skeleton** for the business logic of the developed SDR components, based on the header files associated with the application-specific interfaces.

2 PIM IDL Profiles

2.1 Introduction

This section provides the normative content for the two standardized PIM IDL Profiles:

- The “Full” PIM IDL Profile,
- The “Ultra-Lightweight” / “ULw” PIM IDL Profile.

The Full Profile is specified by removing features of the IDL language that introduce some relationship to CORBA technology, or that serve purposes other than declaration of application-specific interfaces (e.g. keywords “attribute” or “component”).

The ULw Profile is a strict subset of the Full PIM IDL Profile that addresses the needs of computation intensive components, such as waveform physical layers components. Resource constrained components and environments can use either the Full or the ULw Profile.

Chapter 7, “IDL Syntax and Semantics”, of the OMG CORBA Standard **[Ref1]** is the normative content used as reference to identify the IDL capabilities selected in the standardized PIM IDL Profiles, taken within the following sections of **[Ref1]**:

- Section 7.6 Import Declaration,
- Section 7.7 Module Declaration,
- Section 7.8 Interface Declaration,
- Section 7.9 Value Declaration,
- Section 7.10 Constant Declaration,
- Section 7.11 Type Declaration,
- Section 7.12 Exception Declaration,
- Section 7.13 Operation Declaration,
- Section 7.14 Attribute Declaration,
- Section 7.15 Repository Identity related Declarations,
- Section 7.16 Event Declaration,
- Section 7.17 Component Declaration,
- Section 7.18 Home Declaration.

The following sections of **[Ref1]** are related to the language structure of IDL, and are not addressed in the profiles of this specification:

- Section 7.1 Overview,
- Section 7.2 Lexical Convention,
- Section 7.3 Preprocessing,
- Section 7.4 IDL Grammar,
- Section 7.5 IDL Specification,
- Section 7.20 Names and Scoping.

Content corresponding to a feature taken out of the subset defined for the considered profile is not applicable for profile conformity. Any other features are applicable.

The CORBA-related chapter Section 7.19 CORBA Module is not applicable for the defined Profiles.

The *keywords set* attached to a profile contains all the IDL keywords allowed for use in accordance with the defined profile. The following sections indicate if keywords attached to IDL features are in the *profiles keyword set* corresponding to the specified profiles.

2.2 Import Declaration

The Import Declaration, section 7.6 of **[Ref1]** is **NOT** included in the Profiles.

The keyword ***import*** is **OUT** of the two profiles keywords sets.

2.3 Module Declaration

The Module Declaration, Section 7.7 of **[Ref1]** is **fully** included within the Profiles.

The keyword ***module*** is **IN** the two profiles keywords sets.

2.4 Interface Declaration

The Interface Declaration, Section 7.8 of **[Ref1]** is **partially** included within the Profiles.

The following features are **fully** included within the Profiles:

- Section 7.8.1 Interface Header,
- Section 7.8.3 Interface Body,
- Section 7.8.4 Forward declaration.

The keyword ***interface*** is **IN** the two profiles keywords sets.

The following features are included within the Full Profile, and **NOT** included in the ULw Profile:

- Section 7.8.2 Interface Inheritance Specification,
- Section 7.8.5 Interface Inheritance.

The following features are **NOT** included in the Profiles:

- Section 7.8.6 Abstract Interface,
- Section 7.8.7 Local Interface.

The keywords ***abstract*** and ***local*** are **OUT** of the two profiles keywords sets.

2.5 Value Declaration

The Value Declaration, Section 7.9 of **[Ref1]** is **NOT** included in the Profiles.

The keywords ***custom, truncatable, supports, private*** and ***public*** are **OUT** of the two profiles keywords sets.

2.6 Constant Declaration

The Constant Declaration, Section 7.10 of **[Ref1]** is **fully** included within the Profiles.

The keyword ***const*** is **IN** the two profiles keywords sets.

2.7 Type Declaration

The Interface Declaration, Section 7.11 of **[Ref1]** is **partially** included within the Profiles.

2.7.1.1 Basic Types

The Basic IDL Types are specified in Section 7.11.1 of **[Ref1]**.

The following Basic Types are **IN** both Profiles:

- Integer Types: ***short***, ***unsigned short***, ***long*** and ***unsigned long***,
- Boolean Type: ***boolean*** (with **TRUE** and **FALSE** keywords),
- Octet Type: ***octet***.

The following Basic Types are **IN** the Full Profile, and **OUT** of the ULw Profile:

- Integer Types: ***long long*** and ***unsigned long long***,
- Floating-Point Types: ***float***, ***double*** and ***long double***,
- Any Type: ***any***.

The following Basic Type is **OUT** of both Profiles:

- Wide Char Type: ***wchar***.

2.7.2 Constructed Types

The use of Constructed Types (keyword ***typedef***), Section 7.11.2 of **[Ref1]**, is **IN** both Profiles.

Structures

The use of Structures (keyword ***struct***), Section 7.11.2.1 of **[Ref1]**, is **IN** the Full Profile, and **IN** the ULw Profile **with restrictions**:

ULw Profile restriction: structure field types are restricted to Basic Types allowed by the ULw profile.

Discriminated unions

The use of Discriminated unions, Section 7.11.2.2 of **[Ref1]**, are **IN** the Full Profile, and **OUT** of the ULw Profile.

The keywords ***union***, ***switch***, ***case***, ***default*** are **IN** the keywords set of the Full Profile and **OUT** of the keywords set of the ULw Profile.

Constructed Recursive Types and IForward Declarations

The use of Constructed Recursive Types and IForward Declarations, Section 7.11.2.3 of **[Ref1]**, are **IN** the Full Profile, and **OUT** of the ULw Profile.

2.7.3 Template Types

The Template Types are specified in Section 7.11.3 of **[Ref1]**.

Sequences

The use of Sequences (keyword ***sequence***), Section 7.11.3.1 of **[Ref1]** are **IN** the Full Profile, and **IN** the ULw Profile **with restrictions**:

ULw Profile restriction:

- Sequences can only be typed as a ULw Profile **Basic Types** or a ULw Profile **structure**,
- The sequence must be **bounded**.

Strings

The use of Strings (keyword ***string***), Section 7.11.3.2 of **[Ref1]** are **IN** the Full Profile, and **OUT** of the ULw Profile.

Wide strings and Fixed

The use of WStrings and Fixed (keywords ***wstring*** and ***fixed***), Section 7.11.3.3 and Section 7.11.3.4 of **[Ref1]**, are **OUT** of both Profiles.

2.7.4 Complex Declarator

Support of the Complex Declarator (corresponding to use of **arrays**), Section 7.11.4 of **[Ref1]**, is **IN** the Full Profile, and **OUT** of the ULw Profile.

2.7.5 Native Types

The use of Native Types (keyword ***native***), Section 7.11.5 of **[Ref1]**, is **OUT** of both Profiles.

2.7.6 Deprecated Anonymous Types

Usage of anonymous types, Section 7.11.6 **Error! Reference source not found.**, is OUT of both P rofiles.

2.7.7 Object Type

The Object type is **IN** the Full Profile.

The Object keyword can be used as a type specification of an untyped object reference. For example, it can be the type of an operation parameter that can accept a reference to an object supporting any interface.

Type Object is **OUT** of the ULw Profile.

2.8 Exception Declaration

The Exception Declaration, Section 7.12 of **[Ref1]** is **fully part** of Full Profile, and **NOT part** of the ULw Profile.

The keyword ***exception*** is **IN** the keywords set of the Full profile and **OUT** of the keywords set of the ULw profile.

2.9 Operation Declaration

The Operation Declaration, Section 7.13 of **[Ref1]** is **partially** included within the PIM IDL Profiles, as detailed in the following sub-sections.

2.9.1 Return values

The specification of return values is **IN** both Profiles.

The keyword ***void*** is **IN** the keywords set of both profiles.

No type restrictions apply for the Full Profile, while a restriction exists for the ULw Profile.

ULw Profile restriction: return values can only be **void** or an ULw profile Basic Type.

2.9.2 Operation Attribute and Parameter Declarations

The following features are **fully part** of both Profiles:

- Section 7.13.1 Operation Attribute,
- Section 7.13.2 Parameter Declarations.

The keywords ***oneway***, ***in***, ***out*** and ***inout*** are **IN** the keywords set of both profiles.

2.9.3 Exceptions

The following feature is **part** of Full Profile, **with restrictions,** and **NOT part** of the ULw Profile:

- Section 7.13.3 Raises Expressions.

Full Profile restriction: exceptions associated with IDL attributes are NOT supported.

The keyword ***raises*** is **IN** the keywords set of the Full Profile and **OUT** of the keywords set of the ULw Profile. The keywords ***getraises*** and ***setraises*** are **OUT** of the keywords sets of both profiles.

2.9.4 Context

The following feature is **NOT part** of the Profiles:

- Section 7.13.4 of **[Ref1]** Context Expressions.

The keyword ***context*** is **OUT** of the keywords sets of the two profiles.

2.10 Attribute Declaration

The Attribute Declaration, Section 7.14 of **[Ref1]** is **NOT** included in the Profiles.

The keywords ***attribute*** and ***readonly*** are **OUT** of the keywords sets of the two profiles.

2.11 Repository Identity Related Declarations

The Repository Identity Related Declarations, Section 7.15 of **[Ref1]** are **NOT** included in the Profiles.

The keywords ***typeid*** and ***typeprefix*** are **OUT** of the keywords sets of the two profiles.

2.12 Event Declaration

The Event Declaration, Section 7.16 of **[Ref1]** is **NOT** included in the Profiles.

The keywords ***eventtype*** and ***abstract*** are **OUT** of the keywords sets of the two profiles.

2.13 Component Declaration

The Component Declaration, Section 7.17 of **[Ref1]** is **NOT** included in the Profiles.

The keywords ***component***, ***supports***, ***provides***, ***uses***, ***multiple***, ***publishes***, ***emits*** and ***consumes*** are **OUT** of the keywords sets of the two profiles.

2.14 Home Declaration

The Home Declaration, Section 7.18 of **[Ref1]** is **NOT** included in the Profiles.

The keywords ***home***, ***manages***, ***primarykey***, ***factory*** and ***finder*** are **OUT** of the keywords sets of the two profiles.

3 Support information

3.1 Tabular Representation of IDL Profiles

This section provides a tabular view of the IDL Profiles. Within the section the following conventions are used:

- IN: the feature is IN the profile,
- OUT: the function is OUT of the profile.

Table 1 Supported Declarations (except Types)

IDL Feature	Associated keyword	Full Profile	ULw Profile	Diff Full/ULw
Section 7.6 Import Declaration	Import	**OUT**	**OUT**	
Section 7.7 Module Declaration	Module	**IN**	**IN**	
Section 7.8 Interface Declaration				
Section 7.8.1 Interface Header	Interface	**IN**	**IN**	
Section 7.8.2 Interface Inheritance Specification		**IN**	**OUT**	**YES**
Section 7.8.3 Interface Body		**IN**	**IN**	
Section 7.8.4 Forward declaration		**IN**	**IN**	
Section 7.8.5 Interface Inheritance		**IN**	**OUT**	**YES**
Section 7.8.6 Abstract Interface	Abstract	**OUT**	**OUT**	
Section 7.8.7 Local Interface	Local	**OUT**	**OUT**	
Section 7.9 Value Declaration	valuetype, custom, truncatable, supports, private, public	**OUT**	**OUT**	
Section 7.10 Constant Declaration	Const	**IN**	**IN**	
Section 7.12 Exception Declaration	Exception	**IN**	**OUT**	**YES**
Section 7.13 Operation Declaration				
Section 7.13 Operation Declaration	Void	**IN**	**IN**	
Section 7.13 Operation Declaration	*return values*	**IN**	**IN (1)**	**YES**
Section 7.13.1 Operation Attribute	Oneway	**IN**	**IN**	
Section 7.13.2 Parameter Declarations	in, out, inout	**IN**	**IN**	
Section 7.13.3 Raises Expressions	Raises	**IN**	**OUT**	**YES**
Section 7.13.3 Raises Expressions	getraises, setraises	**OUT**	**OUT**	
Section 7.13.4 Context Expressions	Context	**OUT**	**OUT**	
Section 7.14 Attribute Declaration	readonly, attribute	**OUT**	**OUT**	
Section 7.15 Repository Identify Related Declaration	typeid	**OUT**	**OUT**	
Section 7.16 Event Declaration	custom, eventtype	**OUT**	**OUT**	
Section 7.17 Component Declaration	component, supports, provides, uses, emits, publishes, consumes	**OUT**	**OUT**	
Section 7.18 Home Declaration	home	**OUT**	**OUT**	

(1) Return values are restricted to ULw basic types.

Table 2 Supported Type Declarations

IDL Feature	Associated keyword	Full Profile	ULw Profile	Diff Full/ULw
Section 7.11 Type Declaration				
Section 7.11.1 Basic Types				
Section 7.11.1.1 Integer Types	short, unsigned short, long, unsigned long	**IN**	**IN**	
Section 7.11.1.1 Integer Types	long long, unsigned long long	**IN**	**OUT**	**YES**
Section 7.11.1.2 Floating-Point Types	float, double, long double	**IN**	**OUT**	**YES**
Section 7.11.1.3 Char Type	Char	**IN**	**OUT**	**YES**
Section 7.11.1.4 Wide Char Type	wchar	**OUT**	**OUT**	
Section 7.11.1.5 Boolean Type	boolean,TRUE, FALSE	**IN**	**IN**	
Section 7.11.1.6 Octet Type	octet	**IN**	**IN**	
Section 7.11.1.7 Any Type	any	**IN**	**OUT**	**YES**
N/A	Object	**IN**	**OUT**	**YES**
Section 7.11.2 Constructed Types				
Section 7.11.2 Constructed Types	typdedef	**IN**	**IN**	
Section 7.11.2.1 Structures	struct	**IN**	**IN (1)**	**YES**
Section 7.11.2.2 Discriminated Unions	Union switch, case, default	**IN**	**OUT**	**YES**
Section 7.11.2.3 Constructed Recursive Types and IForward Declarations		**IN**	**OUT**	**YES**
Section 7.11.2.4 Enumerations	enum	**IN**	**IN**	
Section 7.11.3 Template Types				
Section 7.11.3.1 Sequences	sequence	**IN**	**IN (2)**	**YES**
Section 7.11.3.2 Strings	string	**IN**	**OUT**	**YES**
Section 7.11.3.3 WStrings	wstring	**OUT**	**OUT**	
Section 7.11.3.4 Fixed Type	fixed	**OUT**	**OUT**	
Section 7.11.4 Complex Declarator	*arrays*	**IN**	**OUT**	**YES**
Section 7.11.5 Native Types	native	**OUT**	**OUT**	
Section 7.11.6 (Deprecated) Anonymous Types		**OUT (3)**	**OUT (3)**	

(1) ULw structures are restricted to ULw basic types.

(2) ULw sequences are restricted to basic types or ULw structures, and have to be bounded.

(3) Not allowed in conformance with the deprecation stated in the referenced IDL specification.

3.2 Evolution perspectives of the standard

The defined PIM IDL Profiles are constrained to the specification of application-specific interfaces. Expansion of the PIM IDL Profiles to define entire SDR Applications could be considered as a possible extension of the standard (namely including keywords such as component or attribute).

UML-based equivalents of the specified PIM IDL Profiles could be another enhancement to this standard.

4 Rationale

4.1 Development paradigm

This section provides general explanations of the decisions made during the development of the profiles.

4.1.1 Definition of the Full PIM IDL

The objective of the Full profile was to identify a set of IDL features that could be adapted to a "Platform-Independent" specification of application-specific interfaces, eliminating any IDL features which did not contribute to that goal.

The removed features belong primarily to two groups: those which imply usage of particular types (such as ValueType), and those associated with specifying entities other than software interfaces (such as Attribute and Component declarations).

4.1.2 Definition of the ULw PIM IDL

The main driver behind the ULw profile was to identify a more restricted set of features, tailored to suit the *essential needs* of computation intensive components, such as waveform physical layers components.

Other considerations, such as limited support of some digital representations (e.g. 64 bits, floating point) in embedded resource constrained processors and limited availability of IDL to VHDL code generation solutions also played a role in determining which features to include within the ULw IDL Profile.

4.2 Design detailed rationale

This section provides detailed rationale for the development choices made in accordance with the paradigm explained in previous section.

The rationales are provided on a section-by-section basis, first addressing the definition of the Full profile, and then explaining any additional restrictions introduced in the ULw profile.

4.2.1 Import Declarations

The Import Declaration is not directly related to the specification of interfaces so it is OUT of the specified profiles.

That being said, this capability could be used within some environments, but those implementations cannot be re-used directly within environments that support the defined profile.

4.2.2 Module Declarations

The Module Declaration is IN both profiles, since having the ability to define a namespace discrimination capability is useful, even in a resource constrained environment.

4.2.3 Interface Declarations

Interfaces enable grouping of operations. This capability is therefore IN both profiles and seen as essential.

Abstract interfaces are OUT of both profiles since they are specifically tied to object-oriented programming, and viewed as software programming concepts that could be reintroduced at the PSM level where model refinements take place.

Local interfaces are OUT of both profiles for similar reasons.

While *interface inheritance* is supported by the Full profile, the capability has been excluded from the ULw profile, in order to not over-constrain development environments attached to FPGAs.

4.2.4 Value Declarations

Value Declarations are OUT of both profiles since the feature is not perceived to be valuable by the majority of existing SDR Applications.

4.2.5 Constant Declarations

Constants are IN both profiles, since they are a valuable addition and not difficult to implement.

4.2.6 Type Declarations

4.2.6.1 Basic types

All of the possible **Basic Types** are included in the Full profile, except **wchar**.

The **wchar** type has been excluded since it is not a useful feature for embedded SDR Applications.

Type any

The **any** type was excluded from V1.0.0 since it was specific to CORBA technology, and an important driver of the specification effort was the definition of a CORBA-agnostic solution for the specification of interfaces.

It was understood that the CORBA PSM profiles would likely integrate type **any**, since it is used in SCA standard interfaces where type PropertySet appears.

Issue 26 received on V1 suggested that the **any** type should be included in the Full profile, and remain out of the ULw, to maximize backwards compatibility with existing SCA 2.2.2 Applications and existing APIs.

This suggestion was followed in V2.0.0 of the specification.

ULw restrictions

The types corresponding to 64-bit architectures are OUT the ULw profile, since 64-bit architectures are not commonly available in resource constrained processors. This applies to the keywords **long long** and **unsigned long long**.

For similar reasons, support for floating point types is OUT of the ULw profile. This applies to the keywords **float**, **double** and **long double**.

Type **char** is OUT of the ULw profile since handling characters is not relevant for signal processing resource constrained SDR components.

Type **any** is OUT of the ULw profile since the potential overhead associated with marshaling and availability costs is perceived to not justify the benefits of using this type within a resource constrained environment where generic types are generally not used.

4.2.6.2 Constructed Types

Structures

The ULw profile structures are limited to having members with Basic Types as allowed by the applicable profile.

Discriminated Unions

Discriminated unions and all associated keywords, **union**, **switch**, **case** and **default,** are IN the Full profile, since their usage within existing waveforms has been reported in several discussions.

They are OUT of the ULw profile since support of this feature can represent significant development effort on FPGAs while no essential use case was identified for this capability within constrained SDR components.

Constructed Recursive Types and IForward Declarations

The support of recursive type construction is included IN the Full profile, since it is a commonly supported feature.

They are OUT of the ULw profile since resource constrained environments do not require such elaborate typing capabilities.

Enumerations

The **enumerations** are IN both profiles.

4.2.6.3 Template Types

Sequences

No restriction of any kind applies to sequences as required by the Full IDL profile.

The ULw profile only allows bounded sequences of Basic Types and Structure. This was introduced in consideration of technology limits present within IDL to VHDL code generation solutions (such as FPGA ORBs), and seen as consistent limitations in front of common resource constrained components limitations.

Strings

Type **string** is IN the Full profile.

Type **string** is OUT the ULw profile since handling characters strings is not relevant for signal processing resource constrained SDR components.

WStrings and Fixed Types

Due to the absence of reported usage in SDR Applications, the corresponding types are OUT of both profiles.

4.2.6.4 Complex Declarator

This concept corresponds to support of the ***arrays*** declaration.

It is IN the Full profile, but OUT of the ULw profile due to technology maturity issues that exist within FPGA code generation suites such as FPGA ORBs.

4.2.6.5 Native Types

Native types are **OUT** of both profiles, since native types imply usage of programming language-specific constructs, which are abstracted by the PIM specification.

4.2.6.6 Deprecated Anonymous Types

In concurrence with the deprecation of the anonymous types indicated in **[Ref1]**, usage of anonymous types is **OUT** of both profiles.

4.2.6.7 Object Type

In order to allow the PIM to support generic programming (see http://www.generic-programming.org/), the keyword **Object** is IN the full IDL profile.

Such advanced programming features are not consistent with the definition approach of the ULw profile, therefore the keyword **Object** is OUT of the ULw profile.

4.2.7 Exception Declarations

The support of exceptions is IN the Full profile since the possibility of having components react to abnormal signaling was considered to be valid, and not specific to PSM considerations.

The capability is OUT the ULw profile since it is not natively supported by several popular programming languages for resource constrained environments, notably C and VHDL.

4.2.8 Operation Declarations

The **in**, **out** and **inout** keywords for signature declarations are IN both profiles.

Similarly **return values** are IN both profiles, with a limitation identified in the ULw profile to not overload designs of resource constrained components.

The **oneway** keyword is IN both profiles, since it is appropriate to explicitly specify execution concurrency at the PIM specification level.

For consistency with the choices made regarding exceptions, **raises** is IN the Full profile and OUT of the ULw profile.

For consistency with the choices made regarding attributes, **getraises** are **setraises** are OUT of both profiles.

The support of the **context** capability is OUT of both profiles since it is not commonly used, not supported by most ORBs and insufficiently specified in the IDL standard.

4.2.9 Attribute Declarations

Attribute Declarations are NOT in the specified IDL profiles, since the purpose of the specification is limited to specifying software interfaces of SDR components.

Attribute Declarations are identified as a possible part of a **future extension** of this specification that would enable expression, beyond the SDR component interfaces, of the entire SDR Application structure (identification of SDR components, their attributes and their used and provided interfaces).

4.2.10 Repository Identify Related Declarations

Repository Identify Related Declarations are NOT in the specified IDL profiles, since they are related to run time access to software parts that are not part of PIM-level specification of Application-specific interfaces.

4.2.11 Event Declarations

Event Declarations are NOT in the specified IDL profiles, since the purpose of the specification is limited to the specification of software interfaces of SDR components.

Event Declaration is identified as a possible **future extension** of this specification that would enable expression, beyond the SDR component interfaces, of the entire SDR Application structure (identification of SDR components, of their attributes and their used and provided interfaces).

4.2.12 Component Declarations

Component Declarations are NOT in the specified IDL profiles, since the purpose of the specification is limited to specification of software interfaces of SDR components.

Attribute Declaration is identified as a possible **future extension** of this specification that would enable expression, beyond the SDR component interfaces, of the entire SDR Application structure (identification of SDR components, of their attributes and their used and provided interfaces).

4.2.13 Home Declarations

Home Declarations are NOT in the specified IDL profiles, since they correspond to deployment-related features that are not in the scope of this specification.

5 Acronyms

API	Application Programming Interface
CORBA	Common Object Request Broker Architecture
COTS	Commercially-Off-The-Shelf
DSP	Digital Signal Processor
ESSOR	European Secure Software Radio
FPGA	Field Programmable Gate Array
GPP	General Purpose Processor
IDL	Interface Definition Language
IPC	Inter-Process Communication
JTNC	Joint Tactical Networking Center
MHAL	Modem Hardware Abstraction Layer
MOCB	MHAL-On-Chip Bus
OE	Operating Environment
OMG	Object Management Group
ORB	Object Request Broker
SCA	Software Communications Architecture
SDR	Software Defined Radio
ULw	Ultra-lightweight
WInnF/WINNF	Wireless Innovation Forum

6 References

URLs below are provided as a convenience, with valid hyperlinks as of the publication date of the specification.

6.1 IDL standard in CORBA

[Ref1] *Chapter 7 "IDL Syntax and Semantics",* in *Common Object Request Broker Architecture (CORBA) Specification, Part 1: CORBA Interfaces*, © 2011 Object Management Group, Version 3.2, November 2011.
URL: http://www.omg.org/spec/CORBA/3.2/Interfaces/PDF (.pdf file).

6.2 SCA 4.0.1 Appendix E-1

[Ref2] *Software Communications Architecture Specification, Appendix E-1: Platform Specific Model (PSM) – Common Object Request Broker Architecture (CORBA)*, Joint Tactical Networking Center, Version 4.0.1, 01 October 2012.
URL: http://jtnc.mil/sca/Pages/sca1.aspx (.pdf file).

6.3 ESSOR Common profile for DSP and FPGA

[Ref3] *Extract from ESSOR SDR Architecture relative to CORBA Profiles for SCA Next*, input document to WInnF submitted by SELEX ELSAG on behalf of ESSOR Industries, WINNF-11-I-0008.
URL: http://groups.winnforum.org/d/do/4691 (.pdf file).

6.4 CORBA

[Ref4] *Common Object Request Broker Architecture (CORBA) Specification, Part 1: CORBA Interfaces*, © 2011 Object Management Group, Version 3.2, November 2011.
URL: http://www.omg.org/spec/CORBA/3.2/Interfaces/PDF (.pdf file).

6.5 MOCB

[Ref5] *MHAL on Chip Bus API*, Joint Tactical Networking Center, Version 1.1.5, 26 June 2013.
URL: http://jtnc.mil/sca/Pages/api1.aspx (.pdf file).

6.6 MHAL Communication Service

[Ref6] *Chapters A, B, C and D* of *Modem Hardware Abstraction Layer (MHAL) API*, Joint Tactical Networking Center, Version 3.0, 02 Oct 2013.
URL: http://jtnc.mil/sca/Pages/api1.aspx (.pdf file).

6.7 Inter-Process Communication (IPC)

[Ref7] Inter-process communication, © Wikipedia
URL: http://en.wikipedia.org/wiki/Inter-process_communication (on-line article).

END OF THE DOCUMENT