Software Communications Architecture

Standardizing Software Radio Architectures
Software Communications Architecture

- History and Motivation
- Concepts
- Key Components
- Provisions for Hardware
- Operation
SCA History and Motivation

US Military had many issues integrating their vast number of “stovepipe” radios
Wanted simple way of managing their many radios
Series of programs led to Joint Tactical Radio System – JTRS
ICNIA
TAJPS
Speakeasy

A joint project of JTRS and SDR Forum – most participants are members of both
SCA Overview

An attempt to develop a “universal” SDR architecture (five identified domains)

Still a work in progress - Currently v2.2

www.jtrs.saalt.army.mil
Specific SCA Goals

- Function as a MBMMR (Multiband Multimode Radio)
- Be interoperable with all domains
- High component reuse
- Be compatible with legacy systems
- Support the insertion of new technologies
- Support advanced networking features
- Use primarily COTS components
SCA Basics

Object Oriented Architecture
- Software AND Hardware Objects
- Interfaces are important (Encapsulation)

Open Systems Architecture
- Publicly available, well-defined interfaces

Framework not a Blueprint
- Minimum functionality specified (Rules)
- Required interfaces, not implementations (Core Framework)
SCA Overview

Family

Domains

Implementations

Dashed Lines Indicate Portions Contained In the SCA

H/W
Classes and Subclasses

S/W
Core Framework (CF)

Operating Environment (OE)

Object Models &IDL

Implementation Specific Devices and Interface Descriptions

Implementation Specific Objects

MPRG
Core Framework

Core Framework defines a set of open application-layer *interfaces* and services that provide an abstraction of the underlying software and hardware layers for software application designers.

All interfaces defined in IDL

Message passing performed using CORBA

Software Implementations use POSIX

**Primary Interfaces**

Resource, Device, Application, Domain Manager, File / File Manager
Core Framework (2)
**Resource**

The fundamental component of the CF
Provides a common Interface for all software components
Serves as base class for API for hardware components.

Note: Internal operation of a Resource is not specified as part of CF
The creation / destruction of a particular resource is managed by a *ResourceFactory*.
**Resource**

- **PortSupplier**
  - `getPort()`

- **LifeCycle**
  - `initialize()`
  - `releaseObject()`

- **PropertySet**
  - `configure()`
  - `query()`

- **TestableObject**
  - `runTest()`

**Resource**

- `identifier : string`
- `start() : void`
- `stop() : void`

**Uses**

- `<<CORBAEnum>> ErrorNumberType`
Device

A Device is specialized Resource used to provide an abstraction of the interface for a hardware device

Primary Operations

State Information
Is it active, busy, locked, idle?

Meaningful way of finding device info in Domain Profile

Capacity
Provides Interface for requesting and deallocating capacity
Devices Classification

**Loadable Device**
Can load and unload software
Example: RAM, DSP

**Aggregate Device**
A collection of devices
Example: A programmable modem that can be programmed to be a CDMA modem or a TDMA modem

**Executable Device**
Can load and unload software and execute and terminate code on the device
Example: DSP
Resource Extensions

- Identifies the Relevant Functionalities that need to be supported
- Specific Resource Extensions
- ModemDevice
  - Antenna, RF, and Modem Entities
- LinkResource
  - Black Processing entity (BridgeResource, WaveformLinkResource)
Resource Extensions

- SecurityDevice
  - Security Entity (INFOSEC)
- NetworkResource
  - Internetworking Entity (RouterResource, WaveformNetworkResource)
- I/ODevice
  - External Interfaces (Serial, Ethernet, and Audio)
- UtilityResource
  - Non-Waveform Functionality (GatewayResource)
Conceptual Model of Resources
**Application**

Provides interface for implementing waveforms

Consists of one or more Resources

Connectable to other applications through ports

Interconnections and properties defined in Domain Profile (Unique File for Each Application)

Constructed by Application Factory

«CORBA Typedef»

DeviceAssignmentSequence
DeviceManager Interface

- DeviceManager interface is used to manage a set of logical Devices and services on a node.
- Typically represents a CORBA capable “board” in a system.
- Creates FileSystem object.
- UnRegisters/Registers itself and its Devices and services with the DomainManager.
- Uses its Device Configuration Descriptor (DCD) profile for determining:
  - How to obtain object reference of DomainManager.
    - Naming Service.
  - Device and Services components to be deployed.
  - File System names, etc.
DomainManager

Performs “housekeeping” on Domain

- Register/unregister services for DeviceManager(s), Device(s), Application(s), and Services
- Install/uninstall Application
- Install/uninstall Service
- “uses” FileManager
- Maintains DomainProfile
Adapters

Adapters are Resources or Devices used to provide the interfaces needed for a non-CORBA compliant entity to be available for use by other entities in the system.

Facilitates the inclusion legacy code, and devices that are not CORBA friendly, such as FPGAs, ADCs, Antennas…
Operating Environment

Primary Components

Core Framework
CORBA Middleware
Operating System

Not common to all domains due to OS
Role of POSIX in the SCA

The Operating System

Specifically used POSIX profile is a derivation of PSE-52

SCA POSIX Profile Defined in Application Environment Profile (AEP)

Approved VxWorks (single process)
Software Structure
Component Roles

- Device Manager
- Domain Manager
- Non-Core Radio Application
- Resource

- Oversees
- Registers with
- 1, ... *

- CORBA
- SCA
- POSIX
- Network
- Lan
- Board Support Packages

- Oversees

- Hardware Device
- Corba adapter

- Hardware Device
- Hardware Device
Hardware in the SCA

[Diagram showing SCA-Compliant Hardware, Chassis, HW Module(s), RF, Modem, Processor, INFOSEC, I/O, Power Supply, GPS, Reference Standard]
Rules

Software
Except for Legacy Code, should be written in higher order language

Hardware
Must provide appropriate Domain Profile files for each device
Interfaces should be open (public and well documented)