Interconnecting SCA Applications

Presented by

François Lévesque,

Steve Bernier

Advanced Radio Systems

Communication Research Centre Canada

November 7, 2007

Outline

- Introduction
- Application reusability
- Aggregate Application concept
- Implementation options
- SCA support for aggregate applications
- Conclusion

Introduction

- **The context**
  - The SCA is a component-based development (CBD) framework for embedded systems
  - Reusability of components is an important aspect of CBD
  - Components can have different level of granularities
    - Fine (e.g. a filter component)
    - Medium (e.g. a demodulator component)
    - Large (e.g. an FM receiver component => can also be an application made of smaller components)
  - An SCA application is made of components interconnected through ports

Introduction

- **The problem**
  - The SCA doesn't specify how Applications can be interconnected
    - What identification mechanism can be used?
  - Radio networking limitations
    - Avoid application reusability
    - Increase storage capacity requirements
  - Proprietary solutions lead to application portability issue

Application Reusability

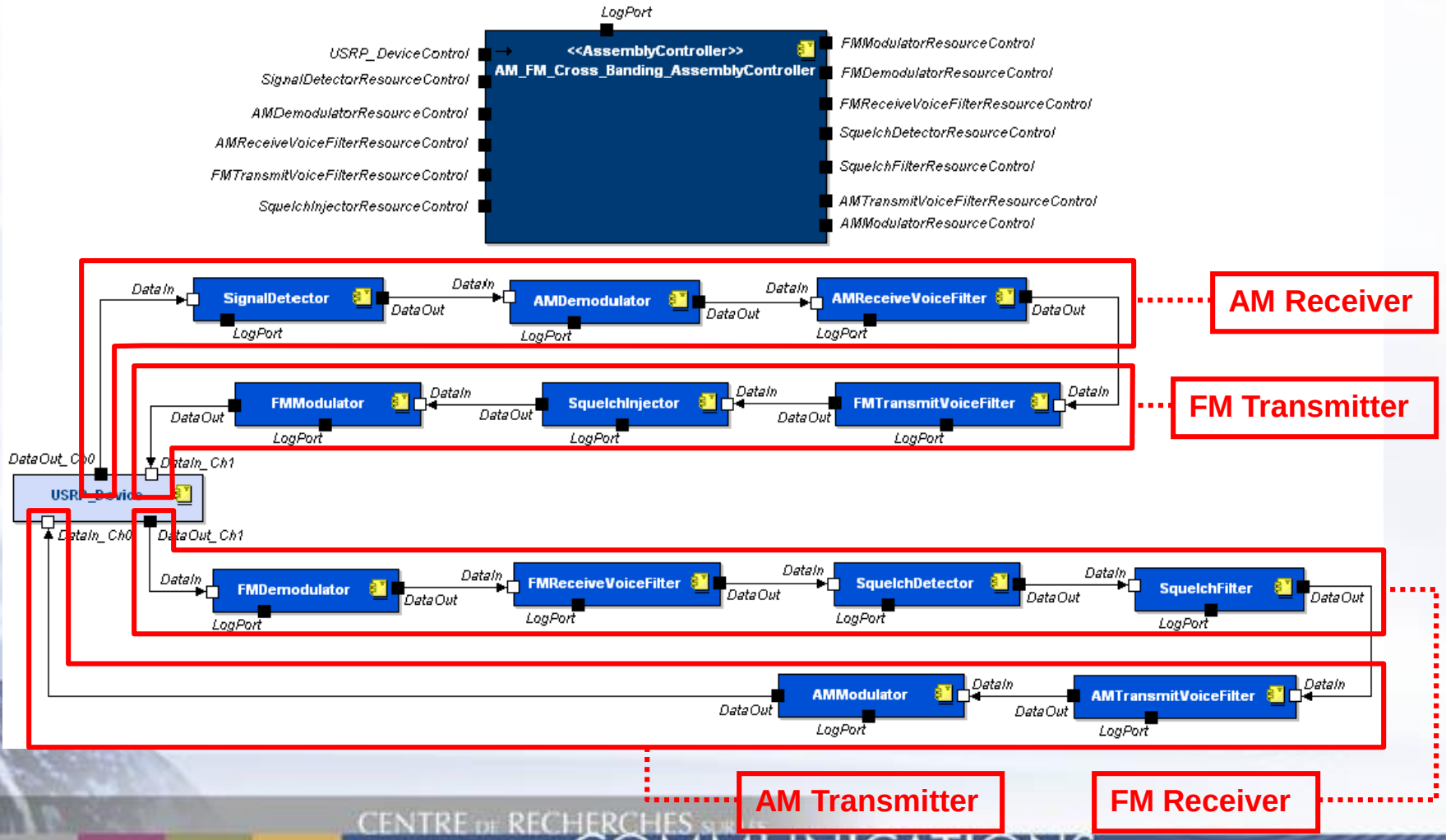
- **Reusability is the corner stone of CBD**
- **An SCA application is defined as an assembly of components (i.e. Resources)**
  - SCA application := *Resource*$^{+}$
  - Resources can be reused in multiple applications
- **Applications are the only way to group *Resources* to implement a specific functionality**
  - Incapability to define sub-assemblies lead to larger components
  - Prevent developer to reuse existing applications to create other applications

Application Reusability

- **Current alternative: create larger applications composed of the amalgamation of Resources from smaller applications**
  - Reuse of existing *Resources* only
  - Assembly knowledge of the smaller applications must be properly duplicated
    - Redefine connections, property overriding, uses device relationships, etc.
    - Assembly controller (AC) of the larger application must contains the same business logic than the ACs of the smaller applications

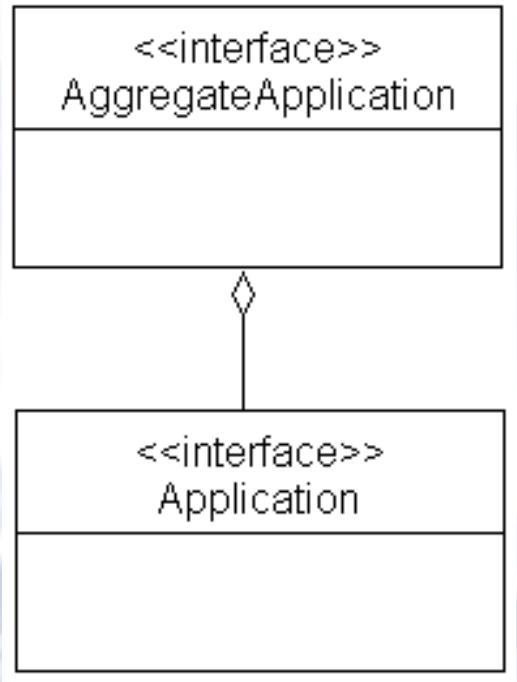
Application Reusability

- **AM-FM cross-banding application example**



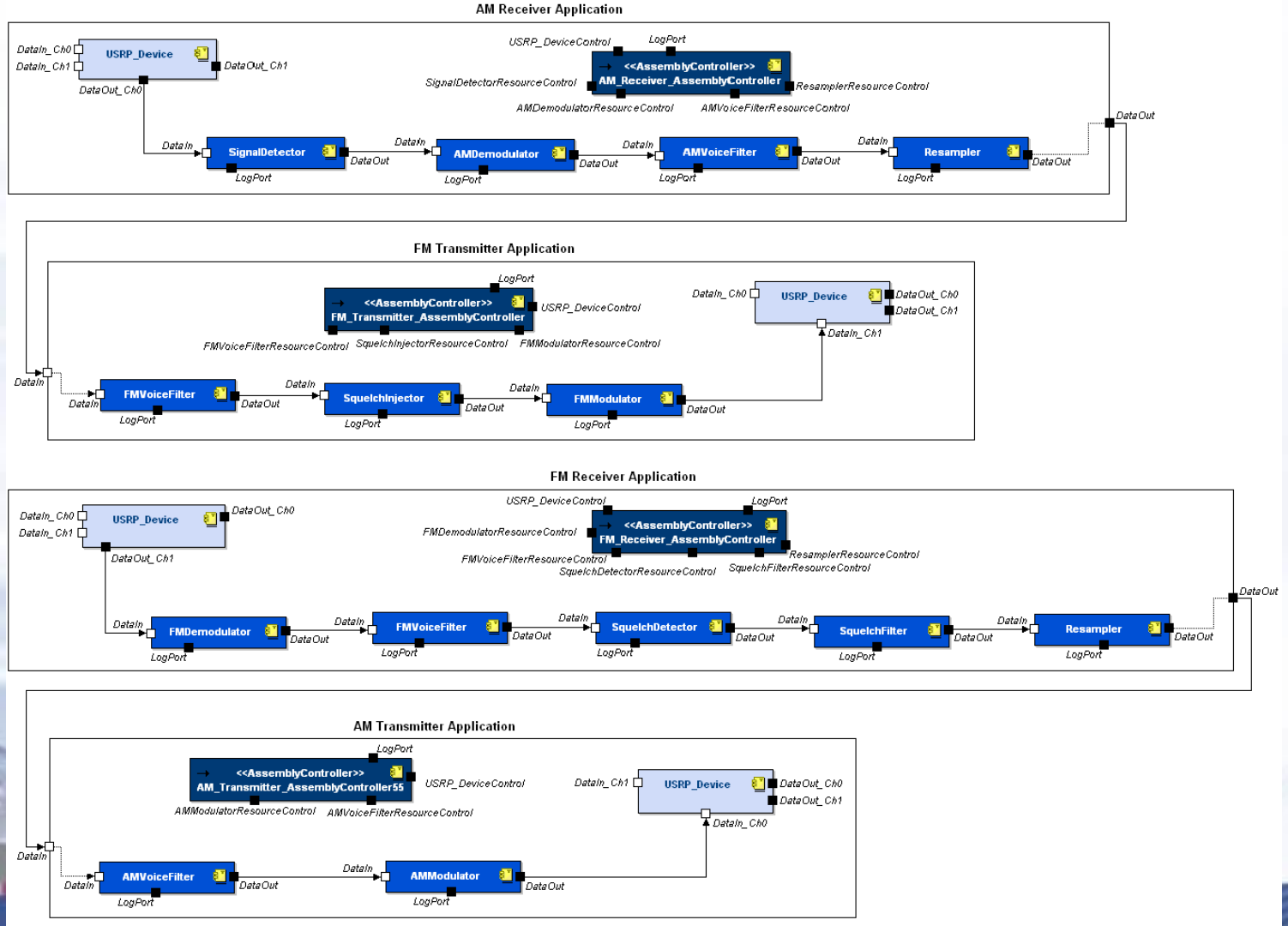
Application Reusability

- **Proposed solution: Add support of Aggregate Application to the SCA**
  - Enable reuse of existing applications



Application Reusability

- **AM-FM cross-banding aggregate application example**



Aggregate Application Concept

- **An aggregate application (AA) is made of multiple applications and/or components**
  - SCA application := (*Resource* | SCA application)$^{+}$
  - Allows the same level of reusability for applications than for components
  - By flattening the recursion, an AA ends-up being composed of *Resources* only but the difference is:
    - extra knowledge about which *Resources* are part of sub-assemblies is provided
- **Use of an application the same way than a regular component enable a CF to:**
  - Coordinate the launch of applications that need to be connected
  - Interconnect applications
    - The SCA already use the concept of external ports to define ports for an application

Implementation Options

- **Option 1: support of AA through modeling tools only**
  - Means to define an AA model is proprietary and the AA is converted into a single application before deployment
  - A CF only handle application made of *Resources* only
    - Aggregate knowledge is lost for error reporting and for the monitoring tools
  - Make it difficult to share AA across different modeling tools

- **Option 2: support of AA through CF and modeling tools**
  - A standard meta-data model is provided to describe an AA
  - Allow the concept to be supported at all levels: modeling, deploying, monitoring, and debugging
  - Requires changes to existing CFs but they are not significant and they can be made optional to implementers that do not wish to support AA

SCA Support for Aggregate Applications

- **Allow a SAD file to reference other applications (SAD files)**

```
                    ⋮
<componentfiles>
  <componentfile id="ComponentFile_1" type="SPD">
    <localfile name="Resource_A.spd.xml"/>
  </componentfile>
  <componentfile id="ComponentFile_2" type="SAD">
    <localfile name="Application_X.sad.xml"/>
    </componentfile>
  <componentfile id="ComponentFile_3" type="SAD">
    <localfile name="Application_Y.sad.xml"/>
  </componentfile>
</componentfiles>
                    ⋮
```

  - CF will have to deal with references to applications instead of only references to *Resources*

SCA Support for Aggregate Applications

- **Extend component instantiation to application**
  - Like for a resource component, this element can be used to specify the information specific to an application instance
    - the application instance's name,
    - the value of some application properties
    - the name to register to the naming service name (optional)

```
                    :
<componentplacement>
  <componentfileref refid="ComponentFile_2"/>
  <componentinstantiation id="DCE:38140af6-5744-4d2c-95e8-55905be34ba0">
    <usagename>Sub Application X</usagename>
    <componentproperties>
      <simpleref refid="Property1" value="a value"/>
    </componentproperties>
    <findcomponent>
      <namingservice name="Application X"/>
    </findcomponent>
  </componentinstantiation>
</componentplacement>
                    :
```

SCA Support for Aggregate Applications

- CF will use an *ApplicationFactory* specific for a sub-application to create the instance of a sub-application
- CF will store the *Application* instance of a sub-application for connection and shutdown purpose

- **Connections to sub-applications can be established through *Application* objects**
  - Component instantiation reference and naming service type of connections can easily be supported the same way they are for regular components
  - Domain finder type of connection could be supported but it would require a new type “application” in the SAD’s DTD

SCA Support for Aggregate Applications

- **Extend the *Application* interface to support sub-applications**
  - A new read-only attribute containing the sequence of sub-applications of the application is required for control an monitoring purpose
  - The attribute can be added to the interface or to a new *AggregateApplication* interface that extends the *Application* interface
- **Application installation service**
  - An application installer tool must be modified to support sub-applications
  - The *DomainManager* installation service must be extended to validate the sub-application meta-data

Conclusion

- **Inter-application connections raise an issue about**
  - How an application to be involved in a connection can be identified and found
  - Application reusability
- **Aggregate Application concept enables application reusability and inter-application connections**
- **Extension to the current SCA specification**
  - No new XML required
  - Backward compatibility is kept for tools and CF
  - Changes to the SCA specification are mostly textual
  - CF Implementers wishing to support aggregate applications must perform non significant changes to their implementation
  - A new type “application” could be added to the *domainfinder* type of connection in the SAD’s DTD but it is not required