WInnF SCA Next WG: Component Factory Proposal

a generalization of the Resource Factory Concept

SCA Next Roll-out
24, 25 August 2010
Washington, D.C.

Presented by: Serge Harnois, ULTRA Electronics, Montreal and
Thomas Bleichner, Rohde & Schwarz, Munich

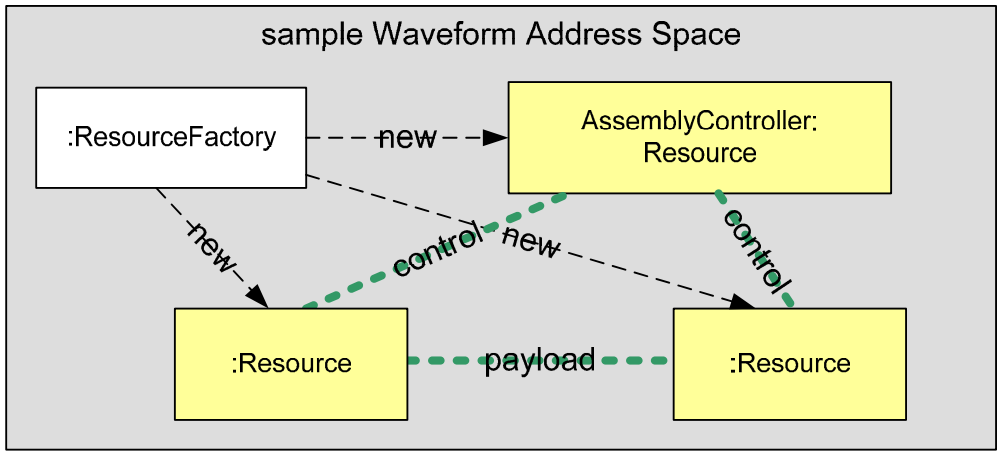
Note

The proposal as presented here was prepared by Serge Harnois, Steve Bernier and Thomas Bleichner, as members of the SCA Next Working Group within the Wireless Innovation Forum.

It contains changes against the document WINNF-10-RFI-0005 since it also replaces the Resource Factory.

The voting process regarding these changes has been started, but at this time neither the Working Group nor the Forum have completed the ballot.

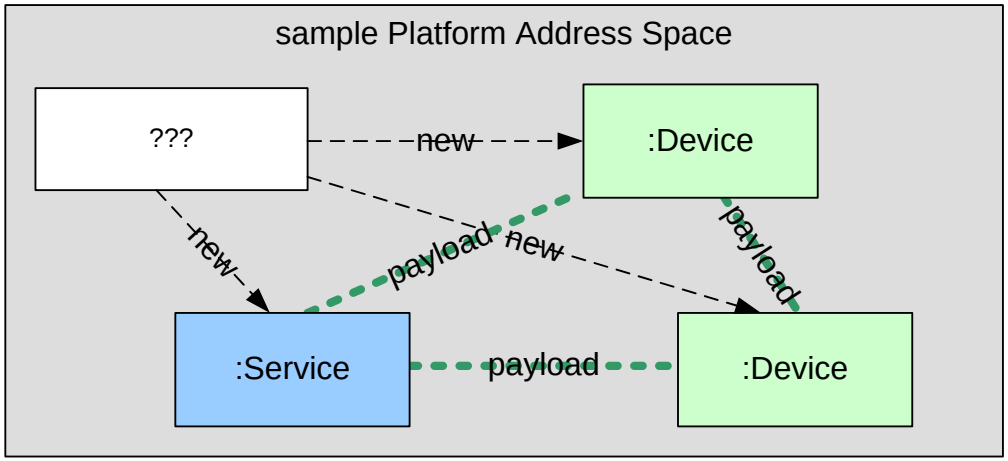
Motivation



The Resource Factory concept allows collocation of waveform Resources within one Address Space

- **Ø significant footprint savings**
- **Ø reduced start-up time**
- **Ø improved real-time performance**

Motivation



We need a technique for the collocation of platform components into one address space:

- **Ø proprietary device manager / executable device ?**
- **Ø the general concept would be a factory !**
- **Ø we see the need to achieve architectural consistency with the Resource Factory concept**

Possible Solutions

Two possible solutions have been considered:

1. **Add a Component Factory to complement the Resource Factory**
   - This Component Factory would be able to create devices and services

2. **Replace the Resource Factory by a Component Factory:**
   - This Component Factory would be able to create resources, devices and services

Solution 1

Benefits:

- Collocation of Platform Components
- Backward compatibility with 2.2.2 applications

Drawback:

- Duplication of the same factory concept in the specification which adds confusion

Solution 2

Benefits:

- Collocation of Platform Components
- Single concept of factory throughout the specification

Drawback:

- 2.2.2 applications will need a new ComponentFactory to replace the ResourceFactory
- Only affects applications that use a ResourceFactory

Selected Solution

Solution 2 has been selected for the following reasons:

- The backward compatibility issues were deemed to be acceptable
- The clarity brought by having a single factory concept throughout the specification

Component Factory Change Proposal

Table of Content:

Description of Enhancement

Allow the Device Manager to co-locate components into a single address space using the concept of a factory

The Resource Factory currently provides this capability for application Resources

Recommended Changes

The ComponentFactory shall replace the ResourceFactory

- This involves changing the SCA Specification version 2.2.2 document
- Describe the ComponentFactory and how the ApplicationFactory and the DeviceManager may use a ComponentFactory
- When a ComponentFactory is provided it can be used to create any types of component that can be launched via a DCD or a SAD. For instance the ComponentFactory could create a Resource, a Device or a service

Recommended Changes (cond't)

The ComponentFactory shall replace the ResourceFactory (cond't)

- Section "3.1.3.1.7 ResourceFactory" should be renamed to "ComponentFactory" and moved out of the section "3.1.3.1 Base Application Interfaces"
- ComponentFactory will keep all the same features as the ResourceFactory. However, the ComponentFactory interface will support (i.e. extend) the LifeCycle interface

Recommended Changes (cond't)

The members of the ComponentFactory are:

```
«readonly» string identifier;

void initialize();

void releaseObject();

Object createComponent(in string componentId, in Properties
qualifiers )  raises (CreateComponentFailure);

Object getComponent(in string componentId);

void releaseComponent(in string componentId) ;
```

Recommended Changes (cond't)

The ComponentFactory is used slightly differently from the ResourceFactory

- createComponent(…) always returns a reference to a new component
- getComponent(…) is used to get a reference to a component that's already been created
- Resources created by the ComponentFactory do not register to the NamingService, this is akin to the former ResourceFactory
- Devices created by the ComponentFactory do not register to a DeviceManager

Recommended Changes (cond't)

Change to the "Appendix D: domain profile"

- The "dtd" for the "SAD","DCD", and "SCD" will require modifications to add support for the ComponentFactory
- SPD and PRF : cosmetic changes in descriptions for how a factory is used and when factoryparams are used
- SCD : rename the component type "resourcefactory" to "componentfactory"
- SAD : cosmetic change to rename "ResourceFactory" to "ComponentFactory"
- DCD : add a componentfactoryref tag to the DCD

Note: The componentFactory will remain optional

Rationale for Change

Collocation of several components into a single address space

- Provides significant footprint savings
  - § Experimentation with 2 resources: (INTEGRITY,PPC E500,OrbExpress)
    - ð Footprint savings 43%
- Improves real-time performances
  - § According a the paper publish at the SDR09 by the CRC*. The collocation in the same address space can improve communication speed up to 90% if the ORB uses direct function invocation
- Reduces deployment time of components
  - § The time of deployment is directly proportional to the component executable file size. Each byte removed from the executable is a byte that the loader does not have to load in memory.

* S. Bernier et al., "SCA Advanced Features – Optimizing Boot Time, Memory Usage, and Middleware Communications", SDRF'09 Technical Conference, 2009.

Rationale for Change (cond't)

Similar savings can be achieved using implementation techniques, without a ComponentFactory

- Using shared libraries and its associated loader (dlopen, dlsym)

  Can be used to reduce the footprint, however this technique is specific to certain operating environments, thus it affects portability of components

- A Component Factory does not require shared libraries and special loaders. The most straightforward implementation of a Factory is to use threads.
  - It is the only platform-independent technique which provides the benefits of address space collocation

Impacts

- Existing applications will need to be updated only if they use Resource Factories
  - The required modifications will be minimal. Modifications will consists in transforming the Resource Factories into Component Factories which are almost identical in APIs
- Minor impact for Core Frameworks
  - New optional tag in the DCD
  - New component type in the SCD
  - DeviceManager must now support the use of a Factory

Questions ?

Backup Slides

Other Considerations

The ResourceFactory has always suffered from a number of issues that can lead to interpretation of the specification. Ultimately this can lead to portability issues

The WinnF SCA Next Adhoc Group and the SCA Implementers Work Group will work on a proposal which will address the Factory issues

Other Considerations

The issues can be grouped in the following topics:

- Which type of components can the ComponentFactory create?
- The ComponentFactory and the hostcolocation tag issue
- What should be the standard creation parameters?
- What should a ComponentFactory do with the requirements defined at the implementation-level of an SPD for a component?
- How can we link the ComponentFactory to the components it can create via the domain profile?