Proposal for SCA v4.1 Push Registration - Allocation Properties

Document WINNF-14-R-0012

Version V1.0.0

16 Sep 2014

Terms and Conditions

This document has been prepared by the SCAv4.1 Backwards Compatibility Task Group to assist The Software Defined Radio Forum Inc. (or its successors or assigns, hereafter "the Forum"). It may be amended or withdrawn at a later time and it is not binding on any member of the Forum or of the SCAv4.1 Backwards Compatibility Task Group.

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.

Intellectual Property Rights

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.

Proposal

This document contains a proposal to change the SCAv4.0.1 specification to reintroduce the possibility for the DomainManager to obtain the proper allocation properties that are associated to a Device that has been instantiated in the system. This capacity was provided in SCAv222 by determining the implementation of the Device that has been launched via getComponentImplementationId() on the DeviceManager and then parsing the Device's XML. However, the getComponentImplementationId() operation on the DeviceManager has been removed in SCAv4.0.1. Without this possibility the ApplicationFactory cannot use a Device for deployment.

Proposal author:

- Hugues Latour, Communication Research Centre Canada (CRC)

Proposal reviewers:

- Gerald L Bickle, Raytheon
- Steve Bernier, NordiaSoft
- François Lévesque, NordiaSoft
- Mathieu Michaud-Rancourt, NordiaSoft
- David Hagood, Aeroflex

Recommendation

SCA v4.1 Push Registration - Allocation Properties

Topics

Description of the Issue
Summary of the Proposal
Detailed Proposal

Specifications Changes (found in a word document)

- Main Specification Changes
- IDL Specification Changes
- DTD Specification Changes

Description of the Issue

For registering SCA Devices to be considered as a potential target/host for the deployment of an application component, it must match the requirements of the application component to be deployed

- For instance, if the application component requires 10 MIPS, the target Devices considered for deployment of the component must at least offer MIPS as a capacity (allocation/external) property

SCA Components can have properties and values for the properties that can be overloaded at different levels. The basic order of precedence is described in section D-1.6.1 and it states that values defined at the implementation level have precedence over the values defined at two other levels.

Description of the Issue

Ultimately, the DomainManager accepts registration of new Devices and creates a pool of Devices that can be used for the deployment of applications.

- The DomainManager must be able to parse the metadata (XML Domain Profiles) associated with a Device in order to create a list of properties with resolved values.
- The allocation properties associated with a Device describe its capabilities (OS, processor, etc.) and capacities (MIPS, MEMORY, etc.)
- The allocation properties are used during the deployment of an application to determine which Device can host each component of the application

For the DomainManager to resolve the values of the properties associated with a registering Device, it must be able to determine which implementation of a Device has been launched.

- Since the DomainManager does not launch Devices, it does not know which implementation has been chosen.
- And since the value of a property can be overridden at the implementation level, the DomainManager needs to know which implementation of a Device has been launched.

Description of the Issue

In SCAv22x, the DomainManager finds out which implementation of a Device was launched by asking the DeviceManager that is associated with the registering Device

- The DomainManager calls CF::DeviceManager::getComponentImplementationId()

In SCAv4.0.1, the DomainManager has no possibility to find out which implementation of a Device has been launched

- The DomainManager cannot call CF::DeviceManager::getComponentImplementationId() anymore since it was eliminated

Summary of the Proposal

Have the DeviceManager provide Device allocation properties during the registration to the DomainManager.

Have the DeviceManager provide its identifier during a platform component registration to the DomainManager.

Eliminate the ManagerRegistry and have the DeviceManager use the ComponentRegistry. This change reduces many duplicate requirements and simplifies the registration process.

Detailed Proposal

In SCA v4, the platform component registration mechanism has been modified to:

- Improve the registration process using the push registration
- Simplify the registration methods to registerComponent instead of registerDevice and registerService
- Use the same registration interface for all base components: Resources, Devices, ResourceFactory, Services, ComponentFactory
- Eliminate getComponentImplementationId() and assume that a device does not have more that one implementation

Detailed Proposal

In V4, all of efforts have been spent to reverse registration to push but a device still requires a lot of "Pulls"

- For the DomainManager to extract and figure out device allocation properties it must read 2 to n xml files from an SCA file system which normally reside with the DeviceManager.
- Embedded file systems are notoriously slow. This impacts boot time.
- If there is more than one implementation the registration can lead to unknown result
- The DeviceManager not only knows what implementation was used but knows exactly what are the allocation properties.

Detailed Proposal

The new Component Type struct in v4 was an introduce to generalize an SCA component which is mainly used by the base components to register to their direct parent component. The ComponentType struct is also used by managers to store the registered components.

- To fix the platform component registration, new component-specific attributes would be required. The devicemanager id and a sequence of allocation properties.
- Adding those new attributes directly to the Component Type is not advisable since it would not be relevant to ALL component types.
- Instead a variant attribute should be added to the Component Type where a struct of component specific information could be provided.

Detailed Proposal

ComponentType Variant Attribute

In CORBA, there is a static variant , union and a dynamic variant the Any.

Using a union is not recommended because it requires that every component would need to know many SCA interfaces.

Using a CORBA::Any decouples the ComponentType struct definition with all the component specific struct definition while providing a conduit to exchange the information

Detailed Proposal

Modified ComponentType

```
string    identifier;
string    managerIdentifier;
string    profile;
ComponentEnumType   type;
Object componentObject;
CF:StringSequence supportedInterfaces;
CF::Ports providesPorts;
Any       specializedInfo; // component specific
```

Detailed Proposal

ComponentType : managerIdentifier

This field is filled by the DeviceManagerComponent for the Platform Components or by the ApplicationFactoryComponent for the Application's Components.

For base component registration to the DomainManagerComponent it would allow the registerComponent operation to add the component to the registeredComponent list of the identified manager.

Detailed Proposal

ComponentType : type (enum)

enum type should be defined at a finer granularity to better support:

- backward compatibility,
- SCA v4 component usage via the ComponentType
- the variant attribute specializedInfo, in helping determine what specific info is contained in the attribute

Detailed Proposal

ComponentType : type (enum)

Would include the legacy types

resource, resourcefactory, device, log, filesystem, namingservice, eventservice, service, devicemanager, domainmanager

Would include new type

componentfactory, applicationcomponent, manageableapplicationcomponent, platformcomponent, managableservice

Would include missing types

applicationfactory, applicationmanager, loadabledevice, executabledevice

Detailed Proposal

ComponentType : SpecializedInfo

The variant specializedInfo attribute can be used to store any component-specific information. For this proposal, only the Device and the DeviceManager is addressed but other structs for ApplicationManager and DomainManager could be defined for future proposal

Detailed Proposal

ComponentType : supportedInterfaces

This is a list of all the IDL interfaces ids that the componentObject supports.

Having this information in the ComponentType improve the SCA v4 component usage for the CF managements and external entities (eg. HCI, monitoring tools)

Detailed Proposal

SpecializedInfo Platform Component

For Device or service,

```
struct PlatformComponentInfo
{
  CF::AllocationProperties allocationProperties;
}
```

Detailed Proposal

SpecializedInfo Platform Component

Where the AllocationProperties is a sequence of AllocationPropertyType

```
struct AllocationPropertyType
{
   string id;
   CF::StringSequence values;
   CF::PropertyActionType action; //enums of the XML action types
   CF::PropertyType type;  //enums of the XML property types
}
```

Detailed Proposal

SpecializedInfo **DeviceManager**

```
struct DeviceManagerInfo
{
  CF::FileSystem fileSys;
  CF::Components registeredComponents;
}
```