A Model for Resource Specification in Mobile Services

Hamid Mukhtar, Djamel Belaïd and Guy Bernard
Institut TELECOM/TELECOM & Management SudParis

3rd ACM International Workshop: SIPE 08
07 July 2008 @ Sorrento, Italy

Presentation Outline

- Introduction & Problem Description
- Motivation
- Service Component Architecture (SCA)
- Composite Capability/Preference Profile
- Proposed Extension to CC/PP
- Integration of SCA and CC/PP
- Conclusions

Introduction & Problem Description

- The SOA introduced the software-as-a-service concept
  - Applications built as consuming/providing services
- SOA specifications are currently focused only on enterprise and business services
  - Interfaces, granularity, heterogeneity, security, transaction etc.
- Mobile services pose additional requirements to the service architecture
  - Hardware capabilities/resources of devices
  - Specific constraints posed by the user application
- When creating mobile services, developers should be able to specify service/application requirements
  - Specify these requirements both abstractly and concretely
- Concrete:1 MB of memory can be specified concretely
- Abstract: to require some *input* mechanism
  - Input can be concretized by keyboard, stylus or speech recognition

Motivation

- Developers should be able to specify resources abstractly
- Abstract resources mapped to concrete resources depending on certain policies, which can be done dynamically

An Example

- A chat application developed in Java
- Tested on Symbian platform with certain minimum amount of required memory
- Useful only if the device has an input mechanism
- Connect to the Internet using WiFi (due to QoS reasons)
- Log the communication and use Kerberos for authentication

We propose an approach

- Policy-based resource description by the developers
- Build on existing SOA specifications (SCA) and resource models (CC/PP)

Service Component Architecture - Overview

- SCA provides a programming model for building applications and systems based on SOA
- An SCA application (composite) is an assembly of heterogeneous components, which implement particular business functionality
- Allows to build distributed applications, which are technology-, protocol-, and implementation-agnostic
- Every SCA component relies on a common set of abstractions:
  - services, references, properties, and bindings

Service Component Architecture

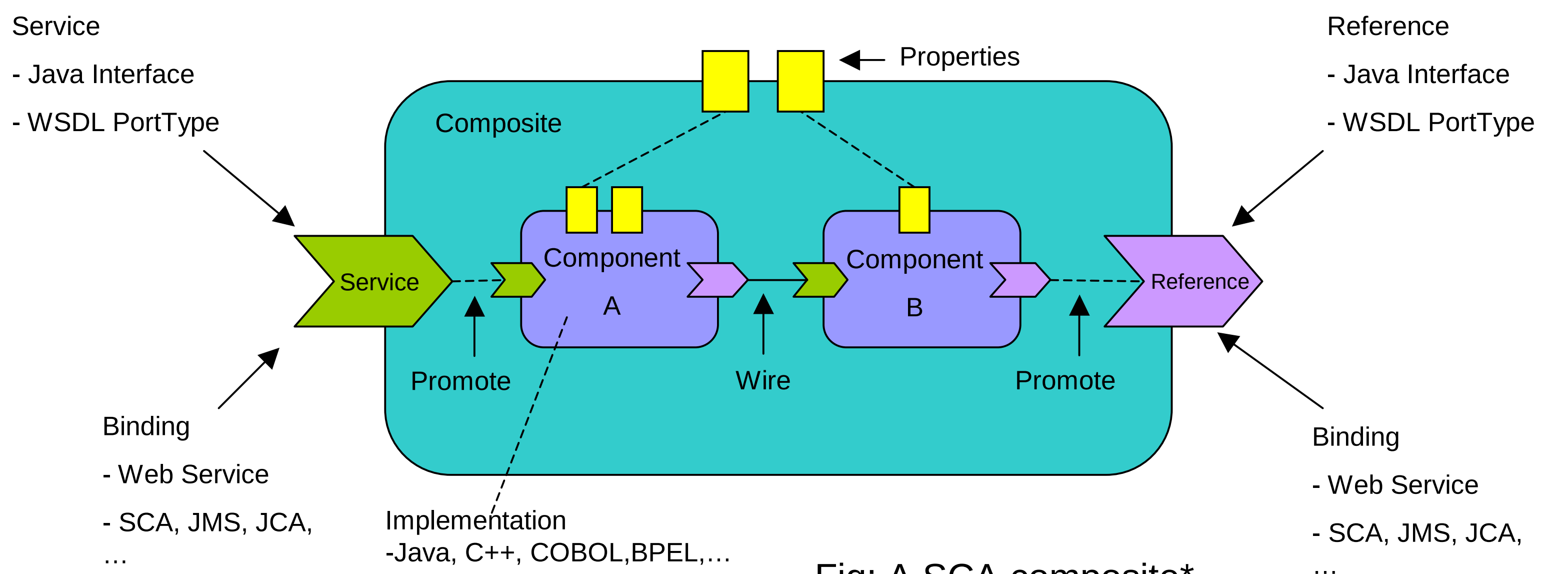


Fig: A SCA composite*

*Figure adopted from SCA v1.00 specs (c) OSOA

The Example Chat Application

```
<composite name="ChatApp">
   <service name="ChatService"/>
   <interface.java interface="ChatItf"/>
   <component name="ChatServiceComponent">
      <implementation.java class="services.ChatServiceImpl"/>
      <reference name="Connection"/>
   </component>
</composite>
```

Limitations of SCA

- Does not consider the resources required by a service or its implementations
- Greater flexibility, but it also affects the way various services are to be considered during binding
  - services shouldn't be tied to any implementation/reference
  - resource requirements should be satisfied
- Both the service providers and clients should express their resources related QoS requirements/specifications
- The interoperability between them will be satisfied only if the requirements are met
  - apart from matching their functional interfaces

Use a Resource Model

- Resources can be specified abstractly and concretely
- CC/PP seemed to be the best choice
  - Extensible, declarative, reusable, expressive, independent

Composite Capability/Preference Profile

- W3C standard for describing device capabilities and user preferences
  - a model providing core vocabulary
- Designed for small, wireless devices such as PDA's and smart-phones
- Defines a two-level hierarchy consisting of components, and their attributes, described in a profile
- A CC/PP profile is an XML document based on Resource Description Framework (RDF)
  - Enables an extensibility mechanism for CC/PP-based schemas

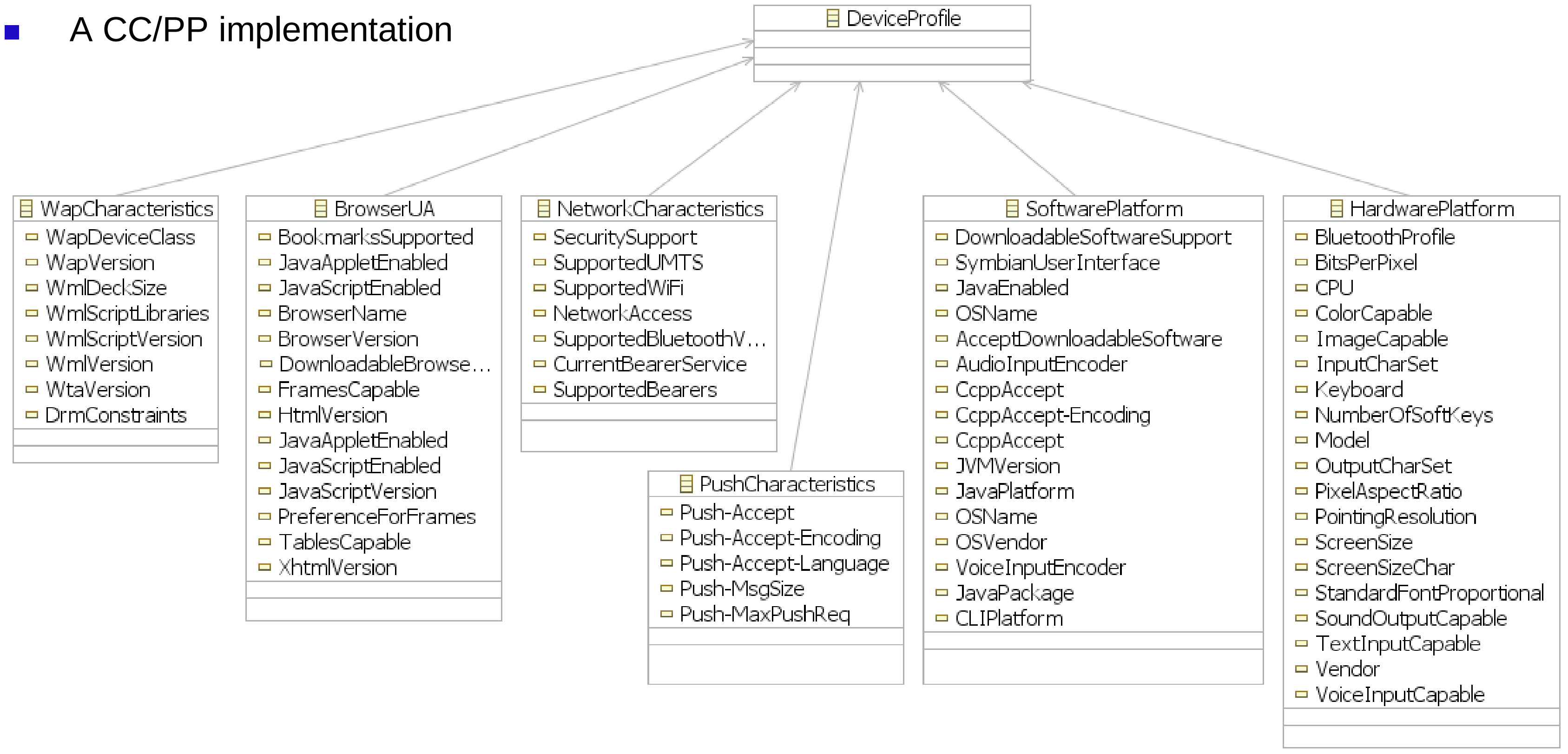
Extending CC/PP

- To meet our requirements
- Enrich the existing model with additional components/attributes

OMA CC/PP Specifications

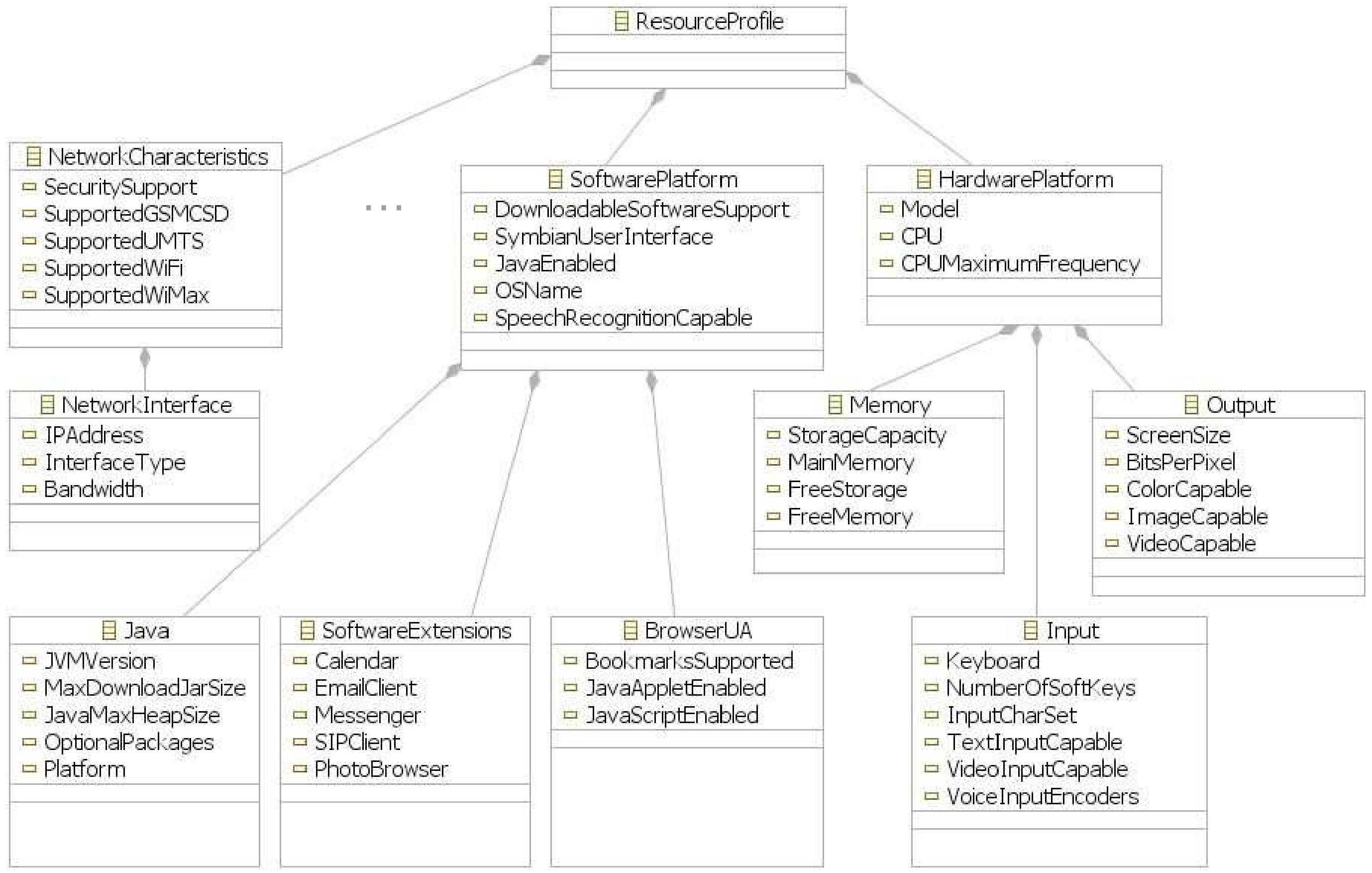
Open Mobile Alliance UAPROF 2.0 Specification

- A CC/PP implementation



Proposed Extension to CC/PP

Categorization and Refinement



Integration of SCA and CC/PP

- How to integrate CC/PP in SCA without violating the SCA specifications?
  - how to specify resources at two levels with existing SCDL?
- Use SCA Policy Framework specifications
  - Policy: describes some non-functional capability/constraint that can be applied to service components or their interactions
    - *Implementation* and *interaction* policies
  - version 1.0 discusses only the security and reliability policies
- Key concepts:
  - Intent allows to specify abstract QoS capabilities or requirements independent of their concrete realization
  - Profile allows the SCA developer to express collections of abstract QoS intents
  - Policy Set provides realization of concrete policies

CC/PP As Policy Language

- No policy language is mandated by the SCA Policy Framework
- How a policy is interpreted depends on how the policy is defined within the domain
- We can also use CC/PP as a policy language
- Define the notions of Intents, Profiles and Policy Sets
  - Intents and Profiles for specifying abstract resource requirements
  - Policy Sets for concrete resource specification
- They are matched using the same algorithm as defined in the Policy Framework specifications
  - In brief: their intersection determines the set of policies used

Abstract Resource Specification - Example

Consider the Chat Application

```
<composite name="ChatApp">
   <service name="ChatService"/requires="Hardware.Input"/>
   <interface.java interface="ChatItf"/>
   <component name="ChatServiceComponent">
      <profile intents="logging sec.authentication/kerberos">
      <implementation.java class="services.ChatServiceImpl"/>
         policySet="SymbianJava"/>
      <reference name="Connection"/requires="Network.SupportedWiFi"/>
   </component>
</composite>
```

- `Hardware.Input` specifies that in order for the client to use it, the `Input` resource from the Hardware category must be available
  - abstract specification: does not specify the type of the input character set or the type of keyboard
- The `Connection` reference specifies `Network.WiFiSupported`, requiring that the component offering `Connection` service must support WiFi

Concrete Resource Specification

- Use the PolicySet element for concrete resource specification
- A PolicySet corresponds to an intent(s)
  - It is a (sub-)profile of CC/PP

The SymbianJava Policy Set

```
<policySet name="SymbianJavaWithHighMemory" provides="SymbianJava"
        appliesTo="implementation.java">
   <ccpp:ResourceProfile xmlns:ccpp="http://example.com">
      <SoftwarePlatform JavaEnabled="true" OSName="Symbian">
         <Java Platform="CDC" OptionalPackages="VirtualKB"/>
         <Memory freeMemory="256"/>
      </SoftwarePlatform>
   </ResourceProfile>
</policySet>
```

- `@provides` specifies the corresponding abstract policy
- `@appliesTo` specifies the affected SCA element

Conclusions

- SOA: specifications aimed at enterprises
- Currently not adequate for mobile services
  - Additional hardware/software requirements
- SCA: inherits the same problems
- We proposed resource model for SCA
- Our contribution was twofold:
  - extension of CC/PP: categorization and refinement (abstract/concrete resource)
  - integrate CC/PP into SCA as a policy language
    - Preserving the existing notions of the SCA Policy Framework

Thank you

Questions?

SCA Composite Example

```
<?xml version="1.0" encoding="ASCII"?>
<composite xmlns=“http://www.osoa.org/xmlns/sca/1.0” targetNamespace="http://foo.com“
    name="MyValueComposite">
    <service name="MyValueService" promote="MyValueServiceComponent">
          <interface.java interface="services.myvalue.MyValueService"/>
          <binding.ws port="http://www.myvalue.org/MyValueService#
                    wsdl.endpoint(MyValueService/MyValueServiceSOAP)"/>
    </service>
    <component name="MyValueServiceComponent">
        <implementation.java class="services.myvalue.MyValueServiceImpl"/>
        <property name="currency">EURO</property>
        <reference name="customerService"/>
        <reference name="StockQuoteService"/>
     </component>

    <reference name="CustomerService“
          promote="MyValueServiceComponent/customerService">
          <interface.java interface="services.customer.CustomerService"/>
          <binding.sca/>
    </reference>

    <reference name="StockQuoteService“
          promote="MyValueServiceComponent/StockQuoteService">
          <interface.java interface="services.stockquote.StockQuoteService"/>
          <binding.ws port="http://www.quote.org/StockService#wsdl.endpoint(…)"/>
    </reference>
</composite>
```

Example taken from SCA v1.00 specs (c) OSOA