An Integrated Service Management Approach Using OSGi Technology and ACAP

M. Cochinwala, S. Moyer, H. Shim,
Telcordia Technologies
One Telcordia Way
Piscataway, NJ 08854
{munir, stanm, hyongsop}@research.telcordia.com

Abstract - IP-based, value-added services have great potential to create value for consumers and generate additional revenue streams for service providers. Telcordia’s Adaptive Communicating Applications Platform (ACAP) is a system designed to provide a uniform and integrated service development platform for developing and deploying value-added services. The OSGi Alliance is an industry wide consortium that defined a standard platform that provides lifecycle management services for value-added services. Great synergy exists between ACAP and the OSGi service platform, as ACAP provides support for the service management issues not addressed by OSGi technology, and vice versa. In this paper, we argue for integration of ACAP and the OSGi service platform in order to provide better overall service management. We present a lightweight, cost-effective integration approach.

1 INTRODUCTION

IP-based, value-added services have great potential to create value for consumers and generate additional revenue streams for service providers. In order to provide and maintain a high quality of consumer experience, service providers should be able to consistently and systematically provision, configure, and manage their services, much in the way telephony services are managed on the operating support systems in the PSTN. This is critical, especially given the current trend of declining revenue from traditional telephony services. As a result, much work is being done in defining and providing support for management of IP-based high-level services, with participation of most players, both big and small, in the industry. JAIN [9] and PARLAY [6] are primary examples of industry-wide effort in this area.

Over the past few years, Telcordia has been developing a high-level service development and management platform, called Adaptive Communicating Applications Platform (ACAP) [6][7]. Specifically, ACAP is designed to provide extensive support for service management needed for operating high-level services. For example, ACAP allows users to authenticate and access their services from different locations. Furthermore, it enables service sessions to migrate from different locations or devices. ACAP provides a platform where user-defined objects can be dynamically integrated into existing services thereby allowing for dynamic service composition and execution.

The mission of the OSGi Alliance is to specify, create, advance, and promote an open service platform for the delivery and management of multiple applications and services to all types of networked devices in home, vehicle, mobile and other environments [1]. The OSGi Alliance’s specification of APIs for the networked delivery of managed services does not define how service sessions should be managed. In fact, the OSGi specification views service session management as part of service-specific logic and thus outside of its scope – the OSGi specification merely defines APIs for a service delivery platform. Thus ACAP and OSGi technology do not overlap and can greatly complement each other by providing “missing pieces” in the other’s support for high-level service management. In particular, ACAP can make extensive use of the OSGi service platform’s life-cycle and remote management capabilities for enabling users to discover and subscribe to services they wish to use and then allowing service providers to seamlessly download, install, and configure service components in user terminals.

Synergy also exists in other areas. For example, because of the availability of the OSGi service platform on a wide variety of computing and network devices, integrating with the OSGi service platform enables ACAP to expand its base of supported terminals with minimal development effort. Furthermore, OSGi defines a device access framework [5], in which services can uniformly discover devices available on users’ networks and use their capabilities in their operation, regardless of underlying networking technologies used by these devices. Thus by integrating with OSGi, ACAP can enable service providers to efficiently integrate existing user communication, media playback, and other devices and networking technologies (e.g., Jini, UPnP, X10) in

OSGi is a registered trademark of the OSGi Alliance
© 2004 Telcordia Technologies, Inc.

0-7803-8785-6/05/$20.00 (C) 2005 IEEE
their services. In turn, this reduces development cost and thus promotes rapid service and feature development in ACAP.

In this paper, we describe how ACAP can integrate and leverage service management facilitates of the OSGi service platform to improve overall service management. The rest of this paper is organized as follows. Section 2 provides a context by presenting and discussing key challenges for next generation service delivery and management systems. Sections 3 and 4 provide an overview of OSGi technology and ACAP respectively. Section 5 describes our integration approach. Section 6 presents an architectural overview for integrated service management. Section 7 gives the current status, and Section 8 concludes this paper.

2 REQUIREMENTS
Typically, the basic role of a service delivery and management system is to facilitate end users to subscribe to offered services. Upon subscription of a service, the management system downloads, installs, and configures any functional components of the service in the computing/networking environment of the subscriber. It may also upgrade and un-install service components as, for example, a new version of the service is released, or the user cancels the subscription.

However, as both diversity of subscriber computing/networking environments and mobility of end users rapidly increase, the new generation of service management systems faces new, difficult challenges. In particular, they should provide support for the following requirements:

- Lifecycle management: service components should be downloaded, installed, configured, updated, and un-installed with minimum operational overhead on end users. In addition, lifecycle-managing a service should not cause undue disruption to ongoing services.
- Adaptive services: services should be able to operate in diverse computing/networking environments in a cost-effective manner. That is, it should be feasible for service providers to offer services for a wide variety of end user computing/networking platforms without having to create a custom version for each platform.
- Session management: in order to support mobile users, service sessions should be portable across different environments, e.g., from PC to PDA, without the user having to stop the current session and start a new one.
- State transfer: as a session moves to a new device, its current state should also be transferred. In addition, it should be feasible for session state to be long-lived, i.e., to be stored and retrieved at a later time. Finally, session state can be transferred on demand, as in a call transfer.

While not exhaustive, these requirements are critical for widespread deployment and acceptance of IP-based high-level services. Independently developed, OSGi mainly addresses the lifecycle management requirement, while ACAP provides support for adaptive services, session management, and state transfer by way of dynamic matching of required resources to available resources in a given environment and migratory objects.

3 OSGi TECHNOLOGY OVERVIEW
The OSGi Alliance (formerly known as the Open Services Gateway Initiative) [1, 2] is an industry consortium formed to address issues in management of value-added services in a variety of network environments. Using standard Java technologies, the OSGi specification defines a general framework for providing service lifecycle management – in fact, a simple way of describing the OSGi specification is that of APIs for “managed Java.” The framework also provides other standard services that are deemed useful for a general class of services. These include permission and user “admin” services, device discovery, configuration, and access services, and HTTP service. The framework is designed to form the basis upon which a variety of other extension services can be added on an as-needed basis. An extension may be defined either by the OSGi itself or by a third-party service provider. An entity that implements the core OSGi framework specification is called an OSGi service platform. Currently, the OSGi service platform has been deployed on diverse network/computing environments, including networked handheld devices, for a variety of applications, e.g., in-vehicle telematics and connected homes [3]. In this section, we give a brief overview of the OSGi service framework, with a particular emphasis on those design features and services that are relevant to the current discussion. For a detailed description and discussion on the framework, see [1, 2].

In the OSGi framework, a bundle is a basic unit of service deployment and management. A bundle contains a set of services, each of which performs some application-specific functionality. A service may also require functionality provided by other services contained in other bundles. Services interact with each other via known Java interfaces. The service lifecycle management of the OSGi framework mainly defines mechanisms for installing, starting, stopping, and uninstalling bundles. The OSGi framework can stop and un-install individual bundles without having to stop the operation of services or the entire platform.

The OSGi framework defines a dynamic environment, in which the exact set of bundles currently installed and activated at any given time is not known in advance.
Therefore, in order for services to interact with each other, they should first be able to discover each other. To this end, the OSGi framework defines a standard service discovery mechanism, called Service Registry, through which it manages service dependencies between collaborating bundles. Using the Service Registry, bundles can “register” their requirements or dependencies on other services. Bundles can also register services that they provide. When a bundle starts, it may register one or more services, at which point the OSGi framework sends event notifications to those bundles that depend on the new service. The dependent services may then start using the new service, for example, by importing and invoking methods on the corresponding service object. Likewise, when a bundle stops and un-registers services, the OSGi framework sends event notifications to dependent bundles.

4 ACAP OVERVIEW

The Adaptive Communicating Applications Platform (ACAP) is an object-oriented platform designed to provide support for adaptive services. ACAP defines a resource hierarchy that functions as a resource type system and is used to specify both resource requirements of services and resource availability in environments. Figure 1 shows a partial resource hierarchy. The absolute path from the root to a node in the hierarchy represents a resource type. For example, resource.comm.audio.ip.sip represents a Voice over IP (VoIP) device capable of using SIP signaling for call setup and control. Each resource type defines an interface through which services interact with resources of that type. Like a class hierarchy in an object-oriented programming language, a child resource type is a specialization of its parent. Using the hierarchy, services may specify their requirements at the level appropriate for their needs. For example, if a service requires a VoIP tool but does not care about the exact protocols used for call setup and control, it may use the interface of resource.comm.audio.ip.

Note that the resource hierarchy in Figure 1 can be extended to include network-level resources. Examples of such resources include bandwidth and other QoS properties and transport-level security requirements. This capability facilitates formal specification of network resource requirements of high-level services to be generated in a consistent and automated manner. In turn, such specifications can be used by network configuration and management systems for resource planning, provisioning, and management.

ACAP is designed with the notion that each user environment would manage its own resources. To this end, every environment has an ACAP Resource Manager (RM). A RM mainly keeps track of the current state of available resources in its environment. It also provides the mechanisms by which services discover and bind to resources in a given environment. Specifically, when a service starts, it sends the local RM its resource requirements, which is a list of resource types. For each type, the local RM returns a resource object that represents a software application or hardware device available in the environment. This object implements the same interface as that implemented by the corresponding resource type. It also interacts with the actual resource in a manner specific to the resource. This way, the service can seamlessly interact with the local resource.

![ACAP Resource Hierarchy](image)

Figure 1. ACAP Resource Hierarchy

One or more resources may be available that can meet a certain requirement. For example, a service may require a resource of type resource.comm.audio.ip, and an environment may have both an H.323 phone and a SIP phone. In such a case, the exact resource to be used is determined by user preference or current state, such as availability.

5 BASIC APPROACH FOR ACAP/OSGI INTEGRATION

The main goal of ACAP/OSGi integration is to enable ACAP services to be life-cycled and remotely-managed in much the same way as general OSGi services are. To this end, we propose the integration approach illustrated in Figure 2.

The idea is to provide ACAP components and services as OSGi bundles and use the basic services of OSGi for required management and operations functions.
In particular, the OSGi Service Registry can be used by ACAP services to discover the ACAP Resource Manager and each other, and the OSGi HTTP Service can provide the data transport mechanism. This approach has the following advantages:

- Integration overhead is minimal. Required changes on ACAP components and services are limited to enabling them to interface with the OSGi Service Platform and use its services. Once they discover each other, the interaction among ACAP components and services remains the same. This means that making an ACAP component OSGi-enabled mostly involves implementing Bundle and Service management interfaces defined in the OSGi Framework Specification, with no changes to the service logic of the component.

- When OSGi-enabled, ACAP components and services are no different from other OSGi services. Thus, they can take full advantage of life-cycle and remote management services in the OSGi service platform.

- ACAP services are decoupled and adaptive by design, and can thus be independently developed, downloaded, and managed, with the OSGi Service Platform providing the “glue” for their interoperation in end user terminals.

6 ARCHITECTURAL OVERVIEW FOR INTEGRATED SERVICE MANAGEMENT

In this section, we identify and describe key architectural components for service management in an integrated environment of ACAP and OSGi. Figure 3 provides an architectural overview for integrated service management. It is based on the architecture of mPRM [4], a Prosyst product for remote management of OSGi-enabled services.

In Figure 3, the Gateway Server refers to a host terminal that runs the OSGi Service Platform – i.e., any OSGi-compliant device. The description and discussion of architectural components are general and apply to other cases, in which a device of a different type plays the role of Gateway Server.

In Figure 3, ACAP Service Providers deploy their services via a Service Directory. A Service Directory is a repository of OSGi service bundles to be downloaded to a Gateway Server and activated by its OSGi Service Platform. When ACAP Service Providers have new or updated versions of services, each creates appropriate OSGi bundles and stores them in the Service Directory.

Gateway Manager is mainly responsible for keeping track of which Gateway Servers have what bundles. When new or updated versions of bundles are available in the Service Directory, the Gateway Manager proactively retrieves them from the Service Directory and downloads them to appropriate Gateway Servers. Conversely, the Gateway Manager can query the Service Directory for new or updated bundles upon receiving requests from Gateway Servers.

Note that the Gateway Manager is outside the scope of the OSGi service platform specification, which defines how service bundles get integrated into the OSGi Service Platform once they “arrive.” Rather, it makes use of and extends the baseline facilities in the specification to automate the process of remote service and life-cycle management. mRPM [4] defines a similar architectural component.

7 SERVICE AGGREGATION: A NEW CAPABILITY

Future communications services will be delivered to consumers from a multitude of service providers on a variety of consumer electronic devices and platforms. Portable middleware will be required to tie these services and platforms/devices together. In addition, the
role of a service aggregator is emerging for aggregating third-party (and their own) services and presenting a unified front-end to end-users (consumers).

By integrating ACAP with the OSGi service platform, a service aggregator can deploy an integrated service management platform. A service aggregator enables multiple service providers to offer services to a consumer (end-user) on a variety of devices and platforms. In this case, as long as the end device or platform is running an OSGi service platform, the service providers’ services can be portable. The service aggregator de-couples the service providers from the end-users, so that the service providers do not have to concerned with where (and on what device) the end-users wants to access the service. The end-user may want to use the service at home, in their car, or on a handheld mobile device. The gateway manager, ensures that the service is available to the end-user wherever it is required. In addition, the end-user need not worry about which service provider is providing a specific function (or service) — the service aggregator encapsulates the complexity of the multiple service providers so that the end-user only sees the service/application that they want (and not how it is delivered, managed, billed, etc.). The combination of ACAP with the OSGi service platform enables a network provider or operator to deploy this “services hub” for service aggregation as shown in Figure 4.

8 CURRENT STATUS

In ACAP, adaptive services are built using migratory objects [6]. Services, which consist of interconnected objects, can dynamically move and execute in different environments on user demand or environment changes. For example, service objects related to call setup and control in a source environment can dynamically move to a destination environment, discover and bind to available resources via the RM in that environment, and establish the call by communicating with their counterparts in the source environment. See [6] for a detailed description. Migratory objects and their management have been built using a commercial platform, called Voyager [10].

As a proof-of-concept implementation, a limited set of supported resources and devices are provided, with which ACAP services can be built. Supported resources include a collaborative whiteboard tool, FTP support, and modules for interfacing with VoIP and PSTN phones. Systematic support for life-cycle management of these resources is also limited.

In addition to its comprehensive support for service life-cycle management, the OSGi provides ready support for integrating with diverse communications and computing devices and home appliances. By leveraging this feature, the ACAP can dramatically expand its support for diverse end user environments and devices. Currently, we are exploring ways to employ ACAP and OSGi services in providing a flexible design, development, and management environment for telemedicine applications.

9 CONCLUSION

The public Internet and growth of Internet-based services require support for system management of services that can execute in and adapt to diverse environments. Both ACAP and the OSGi service platform are designed to provide core components for high-level, value-added consumer services. Because of their mutually exclusive objectives, great synergy exists between ACAP and OSGi technology. ACAP provides a platform where user-defined objects can be dynamically integrated into existing services thereby allowing for dynamic service composition and execution. Meanwhile, the OSGi specification defines a general service development and deployment framework that provides service lifecycle management. From OSGi service platform’s point of view, service creation and deployment is part of application logic. Thus integration of ACAP and OSGi can provide a complete service management environment, in which services developed by multiple, independent third-party providers can seamlessly be lifecycle- and session-managed in a variety of end user computing/network platforms.

In this paper, we have presented our approach to integrating ACAP and OSGi capabilities. Specifically, the individual ACAP components are re-designed as OSGi bundles, which then use the services of the OSGi framework to interact with each other. This approach is cost-effective in that the re-implementation effort is limited to the client code. More importantly, the integrated approach matches the design goals of ACAP where service provider components can be developed.

Figure 4. Services Aggregation Example
and lifecycle-managed independently of each other. ACAP focuses on service creation and execution where services have to adapt to different resources. ACAP assumes that resource management including introduction of new resources will be managed by external resource managers and the OSGi service platform contains functionality that matches a resource manager perfectly. Furthermore, the OSGi service platform is already deployed on a wide variety of end user computing/network devices. Thus ACAP can greatly leverage this fact in expanding its base of supported terminals.

10 REFERENCES


