ABSTRACT
With the emergence of converged networks, Telecom operators are looking at composing Telecom functionality with 3rd party offerings to provide rich, innovative value-added services to their customers. In parallel, Internet Content providers like Google and Yahoo are rapidly moving towards adopting Web 2.0 technologies to enable a variety of end-user applications, and making them accessible to Telecom customers as well. This provides strong competition to similar paid-for services exposed on the Telecom portal. Telecom operators have recently started inspecting their service composition practises to see how it might be possible to exploit a Web 2.0 environment for rapid service creation. In this paper, we take a deeper look into service composition in a converged Web 2.0 Telecom ecosystem, consisting of Telecom functionality, 3rd party offerings and several categories of developers and users. We also propose a generic model for Telecom services in this ecosystem. Thereafter, we critically evaluate various existing service composition efforts for Telecom. Finally, we identify some open challenges and propose insights to address them.

Categories and Subject Descriptors
C.2 [Computer-Communication Networks]: Network Protocols

General Terms
Service Composition, Developers, Protocols, Devices

Keywords
Telecom, Web 2.0, Converged Networks

1. INTRODUCTION
The Telecom business model is evolving. As revenues from voice services get saturated, Telecom operators are aggressively looking for rich and innovative value-added services that can be rolled out rapidly to their customers. However, in the absence of any one specific killer application, differentiation on part of a Telecom operator will come from the ability to rapidly develop several services for narrow customer segments, with a quick time-to-market. Moreover, with the underlying IP and telephony networks converging, these customized services will span across both the IP network as well as the traditional Telecom network.

Telecom operators are facing strong competition from Internet Content providers. These providers are using Web 2.0 technologies to enable rapid development of digital content services. In a Web 2.0 world, almost anyone can develop an application and others in the community can contribute towards enriching it. Google Maps is a typical example of a Web 2.0 application widely used by several communities to compose innovative applications, readily available over the Web. The “walled garden” ecosystem of Telecom operators is clearly under strain.

In recent years, with the advent of Web 2.0 [16], Telecom operators are facing similar competition from Internet Content providers. These providers are using Web 2.0 technologies to enable rapid development of digital content services. In a Web 2.0 world, almost anyone can develop an application and others in the community can contribute towards enriching it. Google Maps is a typical example of a Web 2.0 application widely used by several communities to compose innovative applications. These applications can be accessed over the Web through a browser-enabled phone, and are in direct competition with similar paid-for services available on the Telecom operator portal. Examples of such services range from location-based services to content services (maps, ringtones, etc.). An increasing number of mobile users are now using browser-enabled phones to access such services, bypassing the Telecom portal. For example, it has been estimated that in Europe 90% of the mobile content traffic is off-portal [15]. Moreover, in a Web 2.0 environment, the harnessed collective intelligence of the whole community of application developers and third party service providers gives rise to a plethora of innovative applications, readily available over the Web. The “walled garden” ecosystem of Telecom operators for developing and deploying new applications is clearly under strain.

Telecom operators, however, have an edge over Internet
service providers in terms of their still unmatched core functionalities, like Location, Presence, and Call Control characterized further by carrier-grade Quality-of-Service (QoS) and high availability. The operators are realizing this and are evaluating moving towards a complete “open garden” model for application development. A handful of operators are trying to utilize the service creation potential of Web 2.0 to compete with the Internet companies and derive new sources of revenue. For example, British Telecom has released a Software Development Kit [17] that enables its network services to be utilized in Web mashups. This empowers any user, not just a Telecom operator and its select partners, to create new services using core Telecom functionality and expose them on the Web as well as mobile devices.

In this paper, we study the role of service composition tools and technologies in the light of the move from a “walled garden” ecosystem comprising of selected partners to a complete “open garden” model in a Web 2.0 environment. The rest of the paper is organized as follows. Section 2 proposes a generic Telecom service model and puts service composition for Telecom under the perspective of Web 2.0. Section 3 describes characteristics of service composition in this ecosystem. Section 4 examines current efforts and directions for creating rich Telecom services. Finally, we highlight open research issues and propose insights to address them in Section 5.

2. SERVICE COMPOSITION IN TELECOM

Converged networks integrate voice, video, and data traffic on a single network. This enables the Telecom enterprises to partner with several 3rd party service providers and introduce new value-added services that can be accessed not only over the Web but also through voice and SMS. A plethora of services have been proposed for Telecom in the recent past, for example Business Finder [21], go2 [4] and various Location Based Services (LBS). In this section, we first propose a model of Telecom services that classifies the various components of a composed service. We then put this model under the light of Web 2.0.

2.1 Telecom Service Model

Consider the development of a service that first helps a mobile phone user find a list of nearby cabs and then choose from among them. The list of cabs returned should be based on ‘real-time, dynamic’ information of the cabs being present in the vicinity of the requester, and being available (for example, the cabs should not already be ferrying other persons). Figure 1 describes how such a service can be composed using existing functionalities in the Telecom infrastructure. The service first determines the customer’s location. Location could be obtained from basic cell-site (or GPS) information stored in the phone. It then invokes Business Finder [21] service to return a list of cabs available in the vicinity, feeds the location of selected cab to Google Maps [5], and finally displays the response from Google Maps on the customer’s phone.

This example illustrates the basic components in the model of a composed Telecom service. In this model, the service is first accessed by a customer using her end-device (mobile phone, PDA, etc.). Subsequently, a series of steps are performed, involving invocation of both the device capability (for instance, location information available on the mobile phone) as well as functionalities exposed on the Web (for e.g., Google Maps) and on the Telecom network (for e.g., Presence). Underneath, the sequence of steps is bound together by a service logic. Thus, a composed service in this domain is inherently distributed across the device, hosted services and Telecom network functions. Moreover, it is important to note that a hosted service can itself be a composite service (as is the case with Business Finder in Figure 1), internally making use of various Telecom network and third party services, and having its own application logic. We divide a Telecom service into five major components:

1. **Telecom Network Functionality:** These are represented by rectangular boxes in Figure 1. They expose information like Presence and capability like Messaging, Third Party Call Control, etc. from the core Telecom network.

2. **Hosted Services:** These are third-party services hosted either on the Web (for example, Google Maps) or made available as an offering on the Telecom provider’s infrastructure (for example, ring-tones). In Figure 1, such components are represented by checkered boxes.

3. **Device Functionality:** The user device acts as both the access point (letting the user access the service) as well as the display point (for the returned response). The device is also used as a store house of various information like calendar, user profile, location (e.g. cell site) information etc. These components are indicated by octagonal boxes in Figure 1.

4. **Application Logic:** Represented by rounded rectangular boxes in Figure 1, glues the above three components together and determines the flow of the service. This logic can also include inputs from the customer during the course of service execution. For example, in our composite service, we can ask the customer to select one from the list of cabs returned after the invocation of Business Finder service.

5. **User Interface:** Apart from voice and SMS interfaces (Telecom legacy access modes), user interfaces on devices can be developed as a rich J2ME application or be browser driven.

2.2 Web 2.0 and Service Composition

Web 2.0 [16] is a phrase coined by O’Reilly Media and refers to a perceived business revolution in the Web community caused by the movement to Internet as a platform. In essence, while Web 1.0 was about connecting computers and making technology more efficient for computers, Web

---

1 Business Finder can be considered as a live, dynamic Yellow Pages third party service.
3. TELECOM SERVICE COMPOSITION ECOSYSTEM

Figure 2 describes an ecosystem for service composition in Telecom in an open-garden Web 2.0 world, comprising of multiple providers, developers and consumers. On one side, we have the Telecom operators that expose the core cellular capabilities of Messaging, Presence information, etc. On the other side, we have various third-party providers making available different kinds of content (ring tones, video, etc.) and services (e.g. Google Maps). The existing functionalities and services are used (and possibly rated) by a host of service developers to compose new services. The composition process is mediated with the help of a Services Registry that provides interfaces for look up, publish and blogging. Finally, we have the consumers who access the services published in the registry using a variety of end-user devices.

The emergence of converged networks, the associated Telecom service model and the underlying rich ecosystem provides unique characteristics, opportunities and challenges to the task of service composition. We describe some of the unique characteristics of such a service composition environment.

3.1 Several Categories of Developers

Following the Web 2.0 paradigm, there would be a plethora of developers in the ecosystem, creating a variety of long-tail applications 2. Developers range from being skilled in programming languages like Java and C to having little or no programming experience. Further, these developers will have varying degrees of Telecom expertise. Finally, these various categories of developers will follow different programming styles. For instance, the same functionality can be required by a Java programmer, an Ajax/HTML developer or a PHP user.

A composition environment catering to this ecosystem should support simple, intuitive means for collaboration and reuse. More specifically, it should provide interfaces for look up and selection of available services, while offering a unified view cutting across different programming styles. Moreover, to harness community intelligence, the developers should be able to rate the services used by them and also publish utility snippets that are re-used by other developers.

3.2 Multiple, Heterogeneous Protocols

A composite Telecom service, in general, makes use of multiple functionalities, which in turn are exposed using different protocols. For example, Location and Call Control can be accessed over the Parlay-X interface, Presence related information through the SIP protocol, and Message capabilities via the SMS protocol. Similarly, response from third party services can be in different formats, like XML (for REST based services), SOAP (for Web services), HTML for services invoked over the Web, etc. The popular Google Maps service returns its response in KML, an XML based format.

To make the task of service composition easy, the development environment needs to hide protocol heterogeneity and complexity from the developers. Further, it should facilitate interoperability among different protocols. As an example, it can provide utilities to parse the output of the Presence service and send it over the SMS service.

3.3 Variety of End-user Devices

There is a plethora of end-user devices over which the services are accessed and executed in a Telecom environment. Such devices range from being very basic ones (for example, offering only the capabilities of messaging and voice) to

2The concept of long-tail refers to the existence of a large number of applications, each being used by a small segment of consumers.
being sophisticated, state-of-the-art devices like iPhone [2]. Services developed should be adapted and optimized for the particular end-device. For example, Google Maps already provides a host of implementations for hundreds of mobile phones available in the market today. For easing the incorporation of device functionality (octagonal boxes in Figure 1), the same should be presented in a form that hides the underlying low-level details.

End-user devices communicate with the Telecom network and third party services by selecting among multiple modalities of communication, made available by the underlying converged network. For example, services can be invoked over the Web interface, or through SMS and voice. The service composition process needs to abstract access modalities from application developers, so that it becomes easy for them to integrate with the application.

4. EXAMINATION OF CURRENT EFFORTS

In this section, we examine current efforts that are relevant towards enabling service composition in the Telecom domain. For this purpose, we categorize current efforts into (1) Efforts attempting to expose Telecom functions to the community, (2) Creation of common run-time platforms for execution of composed services, and (3) Tools and technologies being used by 3rd party application developers for Web 2.0 centric application composition. Apart from these, we also examine other relevant efforts in this direction.

4.1 Exposing Telecom Functions

The first wave of efforts along this direction primarily focused on creation of open standards and APIs to implement core Telecom services and functions. Existence of open standards and APIs makes it easier for Telecom services to be interoperable with other applications, thereby making the development task easier. Examples of such APIs and standards are JAIN Intelligent Network Application Protocol (INAP) and Service Logic Execution Environment (SLEE). Session Initiation Protocol (SIP) is also an open standard and widely adopted across service providers to create Voice-over-IP services. Many Telecom operators are moving towards implementing the IMS framework that would enable them to expose their core functionalities (voice service, SMS service, Call control) using SIP. Composing applications using these standards, however, require the developer to have a deep understanding of the protocols and standards.

In an effort to ease up the requirement of deep understanding of protocols, the Parlay consortium came up with the Parlay [14] and the subsequent Parlay-X standards [13]. Parlay-X is SOA-compliant and exposes a Web Services implementation for core Telecom services. Web services today are understood by a wide community of SOA programmers and developers. Parlay-X allows them to use familiar application development tools to create services composed of Telecom services as well as services belonging to 3rd party providers (available over the IP network). For example, developers can use an eclipse-based Java development environment to create a service using Telecom functions, 3rd party service like Google Maps as well as internal application logic. IBM Telecom Web Services Server (TWSS) [7] is one such implementation of Parlay-X. Alcatel is also exploring the option of SOA/REST APIs on top of its Telecom Application Server [1].

4.2 Run-time Platforms and Standards

It is clear that services in this ecosystem can be invoked by several standard-based protocols (e.g. SOAP, SIP over HTTP) and the business logic written in several languages (e.g. Java, C, Workflow orchestration language like BPEL). Industry leaders like IBM, Sun, Oracle have undertaken efforts to come up with application servers that support concurrent execution environments for heterogeneous protocols. Today, they offer application servers that include SIP servlet containers, thus allowing for Java business logic and SIP specific services to be executed on a single container. For example, the IBM Websphere Application Server product suite [8] implements a converged SIP and SOAP servlet container, thus enabling the developers to compose applications with business logic and SIP/SOAP based service invocation support within the same development environment. JSR-289 \(^3\) [11] has been proposed by Sun and Ericsson to enhance existing SIPServlet specification and support development of composed applications involving both HTTP and SIP servlets. There are several other initiatives along this front [15]. For instance, IBM supports extensions to a service orchestrated in BPEL to have inline Java code.

4.3 Tools for Web 2.0 Application Composition

One of the most commonly used terms for Web 2.0 applications is a “mashup”. As per www.wikipedia.org, a mashup is 'a website or application that combines content from more than one source into an integrated experience’. Content is picked up from multiple servers (data sources) using technologies like Ajax and REST, and \textit{composed} (or rendered) in the ‘same’ user interface (UI), typically a browser. There are several tools that aid in the creation of such mashup user interfaces. Examples are QEDWiki [6], Yahoo Pipes [18] and Aqualogic [3], with PrestoStudio [10] from JackBe probably being the first enterprise Web 2.0 tool in this space. The advantage of such tools is that it caters to a class of less sophisticated application developers who prefer “drag-and-drop” operations to create their own mashups. For example, QEDWiki provides a browser-based assembly portal where developers and users alike can choose services made available by service providers and integrate the UI in a simple mashup with other UIs. It also allows for specification of simple data flows between two UIs, all driven by a visual user interface. Moreover, these mashups can be shared with other users, who can then re-use them to create their own applications.

In a converged Telecom Web 2.0 ecosystem, services need to be exposed in similar manner, while making them easy to invoke by Web 2.0 standards such as Ajax. However, we note that in this domain, apart from allowing easy mashup in a browser-based environment, orchestrating Telecom services together with 3rd party services might require specification of complex data flows and control flows. As an example, the latitude and longitude information provided by a Location service may need to be parsed before passing it to a reverse geocoder service (a service which returns a textual description of the place given the latitude and longitude). Further, as Figure 1 indicates, a composed service can also include \textit{The focus of JSR 289 is to define application composition and convergence more clearly and also to solve many issues that were found in JSR 116, the current version of the SIP servlets API.}}
application specific logic. Much of this requires explicit coding from the application developer, thereby necessitating the need for a ‘programming’ environment as against a ‘mashup’ environment. Some enterprises have taken a step in this direction. For example, Web2IC [17] from British Telecom is a Web 2.0 based service aggregation environment. It allows developers to integrate core Telecom services and other Web 2.0 services into a single application. Web2IC hides the complexity of Parlay-X or SIP by exposing these Telecom functions as higher level APIs, suitable for invocation from a user interface or a simple Java application. Connected Services Framework Sandbox [12] from Microsoft provides high level Web services interface that hide low level Telecom protocol specific constraints and allows creation of Managed network mashups with services exposed via its service delivery platform.

4.4 Other Efforts

Triana [23] introduces a service creation environment, which aims to facilitate Web service composition by providing a higher level of abstraction and guiding developers in creating composed services. The paper presents a case study that investigates how this environment can be used in a Telecom setting. [19] discusses a service creation tool for intelligent networks so as to allow rapid introduction of new telematic services. This tool enables methods of interactive visual programming for developing new services from basic building blocks and monitoring their execution in an intelligent network. [20] demonstrates composition of a workforce management application enabled over a Telecom network, where the core Telecom functionalities like Location are exposed as semantically annotated Web services and orchestrated using AI planning techniques. However, while such efforts are interesting, they do not address the issues involved in developing services in an “open garden” Web 2.0 ecosystem.

5. OPEN CHALLENGES

Section 2 proposed a generic model for Telecom services. Section 3 highlighted a number of characteristics for service composition in this domain, while Section 4 studied some existing work. In this section, we present some of the open challenges that need to be addressed for enabling rapid roll-out of Telecom services. Some of these challenges apply to service composition in general, however, we believe these are more relevant for Telecom networks in particular.

5.1 Tools for Effective Collaboration and Reuse

As discussed in Section 2, composed Telecom services often span across components that run on the client device, and those that run on the Telecom (e.g. Presence service) and 3rd party infrastructures (e.g. Google Maps). Rapid service development requires appropriate abstractions of each of these components in a form that can be utilized by a developer community with little Telecom expertise. For instance, they can be exposed as widgets for a visual service composition environment, a Java module for a programmer, etc. Further, each of these components can have several artifacts, which are essentially additional resources associated with the component. Examples of such artifacts include sample code fragments showing how a service is invoked, meta-information about the service, utilities like code snippets to parse the response from the service, exception handlers, as well as unstructured free-text blogs and comments about the service. In a Web 2.0 world, such resources can be made available either by the component provider directly or contributed by other developers.

To enable effective reuse of various components, it is important for the service composition environment to place a component and its various associated artifacts in a structured format. For example, a Telecom operator can structure its core network services (Location, Presence, etc.) under a single format, including resources like sample client code, parsing utilities, link to blogs, etc. Similarly, all device specific functionalities can be templatized and populated for various kinds of devices. From a developer’s perspective, this would help in better comprehension and easier incorporation of these components. It is a challenge to design a generic structure that cuts across not only different functionalities, but also different programming styles (Java, AJAX, PHP, etc.) Furthermore, we need a networked repository that allows easy means of publishing for such structured components. Ideally, publishing should be as easy as filling up a form on the Web. This repository should also provide easy look up (search), and feedback mechanisms.

With several providers and community developers contributing towards development of services and its associated artifacts, it is important to help a developer with effective tools for inferencing and recommendation. For example, a developer wishing to invoke a certain service as part of her workflow should be made aware of the artifacts written by the community in parsing the output of the service. While there has been a lot of work in the domain of inferencing and collaborative filtering [22], this ecosystem opens up a new environment for applying current work as well as identifying interesting problems to enable efficient selection and recommendation techniques.

5.2 Encapsulating Telecom Services by providing Higher Level APIs

Today, standards like Parlay-X allow for Telecom services to be accessed through standardized Web service interfaces, with the intent of making it easy for developers to use these services. However, these APIs still require the application developers to understand the underlying protocol that the Parlay-X APIs invoke. At the very least, it requires them to understand Web services. To ease the burden on the developer, implementation of such APIs should be available at the level of the application development environment. For example, generating a client code for the underlying Web service and making it available to a Java developer would tremendously improve her productivity. Similarly, in the current set up, exceptions from a Web service invocation are captured in the form of messages in the SOAP response envelope. It would be helpful to a developer if the service composition environment can provide constructs to parse these messages and throw higher-level exceptions that can be used by the developer directly in her application.

5.3 Testing and Debugging of Services

In an application development environment where Telecom services are being composed with third party services by a Web 2.0 developer community, providing effective testing and debugging mechanisms becomes important. The developers need to test their services not only for correctness and robustness, but also estimate various QoS parameters like scalability, response time and availability. A major chal-
lenge is in performing these tasks without going ‘live’ on the network. For correctness and functionality testing, developers can be provided with a Telecom simulator that mocks the basic Telecom operations of SMS, Call Control, etc. Similarly, we can have a client-device emulator. However, this alone is not sufficient for load testing on the real Telecom infrastructure. For the QoS attributes, the developers would need some guarantees from the components they utilize. For example, a Telecom operator can annotate its offerings with common QoS parameters like response time and availability. Often, such guarantees can be tough to provide on part of the operator. For instance, delay in the SMS service would depend on the network congestion at that time.

5.4 Billing and Provisioning

Telecom operators as well as 3rd party service providers spend a significant amount of time making the necessary arrangements for provisioning a new service and connecting it to their billing system. A Telecom operator needs to worry about how much overhead a new service is going to put on the network. For example, it needs to know the average bandwidth that the new service will consume, the peak load that it will place on core Telecom components such as the Home Subscriber Service, the Location Server, etc. To roll out services rapidly, these evaluations need to be done in a timely and efficient manner. Further, a challenge for the mobile operator is to develop appropriate charging models for letting 3rd party developers access its services. These models can be of various types; for instance, charging could be based on a contract basis or a per usage basis. Finally, while provisioning for a service, Telecom operators and 3rd party service providers also need to manage security, access control and privacy of data and information. For example, access to Location information by unauthorized entities should be restricted.

6. CONCLUSION

With voice revenues getting saturated, Telecom operators are looking towards offering rich value-added services to attract end-users and create additional revenue channels. With the advent of Web 2.0, Telecom operators need to move from their “walled garden” select partner ecosystem for creating new services, to a completely “open garden” model, to keep up with the rapidity with which Internet content providers like Google and Yahoo are enabling roll out of new services. This paper closely examined service composition for Telecom in a Web 2.0 ecosystem. We proposed a generic model for Telecom services and identified various attributes associated to a service composition ecosystem in this domain. We also categorized and evaluated recent work in this direction and highlighted some open issues while proposing insights to address them.

7. REFERENCES