

Defining Generic Architectural Requirements for the Service Delivery Platform

Rolan Christian and Hu Hanrahan
Centre for Telecommunications Access and Services¹
School of Electrical and Information Engineering
University of the Witwatersrand, Johannesburg
Telephone:(011-7177237), Fax:(011-4031929)
E-mail:{r.christian, h.hanrahan}@ee.wits.ac.za

Network Services

Abstract –The Service Delivery Platform (SDP) is a complex IT-based system that integrates into the telco network. The SDP supports convergence between telco and IT domains by simplifying access to telco resources and capabilities. As a result, the SDP hides technology, implementation and distribution specifics of underlying telco resources. However, the SDP is not standardised but has vendor-specific interpretations. In addition, SDP products use mixes of standards-based and proprietary technologies. Our research aims to contribute towards SDP standardisation by defining a framework of generic architectural concepts. The framework abstracts technologies and applies its concepts to SDP development. We define concepts for the framework, that is, SDP architectural requirements. To define the requirements we extract technology-neutral architectural concepts from various SDP interpretations. In this paper we discuss the SDP and evaluate a proposed interpretation. We describe the method used to uncover SDP requirements. Also, we define the SDP and present the requirements. We conclude that the requirements contribute to SDP standardisation, by promoting the development of SDP architectures, independent of technologies, implementation and distribution.

Index Terms – Convergence, Framework, Architecture, Requirements, Abstraction, Standardisation.

I. INTRODUCTION

The telco network contains interacting systems that are implemented using a variety of technologies. Each system performs various functions, for example, networks connect reliably and securely to deliver services to customers. Also, *independent* service platforms enable service development, provisioning and deployment. Services offered by platforms include voice and data services. Integrated into all systems is the Operations/Business Support Systems (OSS/BSS) that manages activities such as service subscriptions, billing and fault recovery.

Currently, another such system to be integrated into the telco network is the *Service Delivery Platform (SDP)* [1]. The SDP is an Information Technology (IT) based platform that integrates independent telco systems and exposes their *service-oriented* functionality to external 3rd parties. Being IT-based, the SDP also provides a point of integration between telco systems and external IT-based systems. External IT systems are located across various domains including the Internet, enterprises and other telcos.

The SDP enables telco/IT *integration* so as to support application development and service delivery. Hence, IT applications may use telco capabilities, via the SDP, to provide customers with telco and Internet-like services.

By using the SDP, the telco benefits from telco/IT *convergence*. For example, the telco increases its revenue by delivering a variety of services to customers on Internet, enterprise and fixed and mobile telco networks.

The SDP is not standardised; rather it represents a *concept* with many interpretations. These interpretations are defined by various analysts, vendors, service providers, IT-using enterprises and telcos. In addition, multiple service platform standards incorporate their view of the SDP into their architecture.

Each interpretation focuses on specific requirements that yield different business models, architectures and implementations. For example, some SDP products focus on content management, OSS/BSS integration and service creation. Also, products use a mixture of standards-based and proprietary technologies. Hence, a unified and *standardised* SDP is needed.

Our research focuses on the following question: “how does one define a complete and standardised SDP independent of technologies?” Therefore, we aim to contribute towards the standardisation of the SDP, by defining a technology neutral framework. The SDP framework contains generic concepts that are technology neutral. By applying these concepts, SDP architectures and implementations are defined.

In this paper, we define a crucial set of concepts for the SDP framework, that are independent of any technologies. These concepts are the SDP *architectural requirements*. The requirements aid in structuring SDP architectures. Similar to other standardised frameworks ([2] and [3]), defining requirements represents the starting point of the SDP framework. To define the requirements we extract generic concepts from various SDP perceptions.

In this paper we provide background on the SDP. We elaborate on a proposed SDP interpretation and discuss its limitations. We then discuss the approach taken to uncover the SDP architectural requirements. Next, we provide a generic SDP definition and uncover its requirements. Last, we discuss the implications of these requirements.

¹The Centre is supported by Telkom SA Limited, Siemens Telecommunications, Vodacom and the THRIP Programme of the Department of Trade and Industry

II. SDP CONCEPT

The SDP has no agreed definition, however, a popular interpretation specifies the SDP as an “IT solution that is deployed by fixed and mobile network service providers to deliver next generation voice and data services to consumers and enterprises” [1].

Aligned with this definition, [1] defines a minimal set of architectural requirements. The SDP:

- manages service creation, provisioning, execution and billing;
- supports the delivery of services in a network and device-independent manner;
- provides a single standardised point for developers to find and use diverse services and content; and
- provides external developers and IT using enterprises with open and secure access to telco capabilities.

Accompanying this definition and requirements, is an SDP architecture. The original architecture is defined in [1] and shown in *Figure 1*. We provide additional detail on the architecture to include converged networks, interfaces, a distribution plane and technology mappings. In the figure, the architecture defines various layers and platforms:

- Service Exposure Layer: provides developers with access to services that abstract applications contained in lower platforms.
- Service Execution Platform: hosts various voice and data applications, that provide services to customers.
- Content Delivery Platform: manages delivery of content to customers.
- Network Abstraction Layer: provides a point of integration between SDP and converged networks.
- Integrating into all layers and platforms are management platforms and data stores.

In the figure we illustrate technologies and Application Programming Interfaces (API) implementing platforms and layers. For example, Parlay and Parlay X [4] implement the service exposure layer. However, Parlay also implements the network abstraction layer, execution platform and content delivery platform. For the network abstraction layer, standards such as Parlay X, JAIN [5], OMA [6] and IMS [7] are used.

A. Limitations

The SDP architecture in [1] is derived from various products, such as [8], [9] and [10], and influenced by their specific requirements and technologies. As a result, this SDP is a simplification of common functions offered by various products.

The SDP architecture in [1] is incomplete, since layers and platforms require further *decomposition* to uncover additional functionality. Also, no appropriate customer/device platform is defined. Therefore, communication between device, SDP and converged networks is not managed.

The SDP architecture maps standards-based APIs onto the architecture in [1]. However, different APIs are used to implement a single platform or layer. Thus, using various

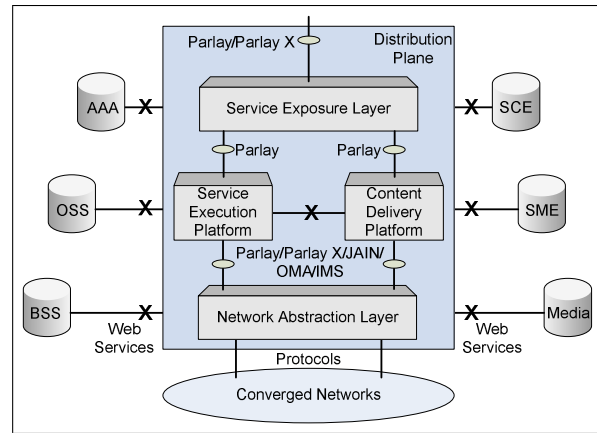


Fig. 1

SDP INTERPRETATION

technologies causes *inconsistency* among SDP implementations. In addition, proprietary web-based APIs are used to expose OSS/BSS and data store functionality to layers and platforms.

The SDP architecture also incorporates the SDP within a distributed environment, that is supported by a distribution *plane*. This plane supports communication between diverse implementations of layers, platforms, services and interfaces. However, [1] does not provide detail on the distribution of the SDP, the distribution plane and its technology choices.

III. UNCOVERING ARCHITECTURAL REQUIREMENTS

We view the SDP concept as the convergence of ideas from various telco, IT and Internet perspectives. Each perspective specifies SDP requirements and defines corresponding architectures to satisfy those requirements. However, these architectures are technology-oriented and use both standards-based and proprietary technologies.

As shown using the interpretations in [1] and *Figure 1*, technology-oriented SDP architectures exhibit various limitations. We aim to overcome these limitations by defining the SDP’s architectural requirements, independent of any product and technology biases. These requirements provide a technology neutral foundation that contributes towards the development of generic SDP architectures.

To specify requirements we *extract* common architectural concepts from current SDP perspectives. The architectural concepts are technology independent. The SDP perspectives originate from telco operators, standards bodies, product vendors, IT using enterprises, service providers, content providers and customers.

A. Definition

Before listing the requirements, we define the SDP as “a structured, distributed and managed IT-based system, simplifying telco infrastructure as abstract services that are offered across diverse network and business domains”. Based on this definition we note the following architectural requirements.

B. Infrastructure Integration

Many SDP products, such as [11], promote infrastructure integration to improve various telco activities. Hence, integration of heterogeneous infrastructure is an objective of the SDP. The infrastructure includes both legacy and new circuit/packet transports, service platforms, media repositories, data stores and OSS/BSS platforms. These systems are distributed and implemented using various technologies.

To integrate infrastructure, the SDP simplifies access to infrastructure parts by *separating* telco resource and capability functions from their complex physical representations. Some functions include:

- setting-up connections;
- negotiating transport QoS;
- configuring device capabilities; and
- updating billing databases.

Separated functions are complex to use since they remain technology and distribution specific. As a result, the SDP abstracts infrastructure functions into reusable, technology-neutral and distribution-neutral software-based *services*. These services provide functionality, such as:

- make multiparty calls;
- obtain customer location;
- manage service profiles;
- send and receive messages; and
- query customer accounts.

These services enable reuse of all telco infrastructure parts for various activities, such as decreasing service development effort, improving network management and streamlining business and operational processes.

These services are used by external IT-based infrastructures, such as service platforms, data stores and content management systems. Hence, SDP services support integration between telco and IT-based infrastructures.

C. Service Oriented System

SDP services contain functionality that abstracts complex infrastructure functions. Since numerous functions exist within the telco, a number of services are created.

Within the SDP, we define two categories of services: *building block services* and *composite services*. Building block services simplify access to network functions, perform distinctive tasks and do not integrate with other building block services. Composite services integrate one or more building block services, further simplifying access to infrastructure functions.

Composite services provide new functionality by integrating multiple composite services and simplifying their use. The nested nature of composite services is finite. Thus, composite services are defined until no further simplification is gained or no unique functionality is defined.

SDP services use generic mechanisms to offer access to their functionality. These mechanisms are called *interfaces*. Interfaces prescribe, in an implementation independent manner, what functionality a service offers. For example, [1] and [12] recommend a service offer the following interfaces:

- **consumption interface** - exposes a service's available functionality.
- **client interface** - enables a service to use other service interfaces.
- **management interface** - used for service administration.

Interfaces expose SDP services to IT and Internet *applications*. Applications use SDP services, via their interfaces, to provide their own services to customers. Also, applications use service interfaces independently of service implementations.

Besides providing implementation independence, interfaces exhibit distribution independence. Therefore, applications using service interfaces are not constrained to specific implementation and distribution technologies.

D. Business Case

The SDP sustains a business environment. In this SDP business environment [13] the SDP supports the convergence of various business entities. For example, the SDP supports business relationships between the telco and external IT, Internet and broadcast enterprises. These enterprises are generalised as *customers* of the SDP.

Various forms of customers exist within the SDP business environment. These customers are providers, consumers and brokers. Providers use SDP services to create new applications or provide resources to others in the business environment. Consumers subscribe, use and pay for applications that are managed by the SDP. Consumers also use SDP services for subscription, profile and account management.

Brokers support interactions between consumers and providers. Brokers are incorporated in [9], [12], [14] and [15] and use SDP services to create applications that provide services to both consumers and providers. For example, consumers may use broker applications to find services offered by application providers.

The SDP must support new customers that enter the business environment at any time. In addition, the SDP must cater for new customers' needs and requirements. Thus, the SDP requires a flexible and dynamic business model.

1) *Generic Business Model*: An illustration of a generic SDP business model is shown in *Figure 2(a)*. The business model caters for various customers and defines relationships between customers, using *business relationship points*. These points are:

- **BR_{CB}** - consumer to broker relationship.
- **BR_{BP}** - broker to provider relationship.
- **BR_{CP}** - consumer to provider relationship.
- **BR_{CC}** - consumer to consumer relationship.
- **BR_{BB}** - broker to broker relationship.
- **BR_{PP}** - provider to provider relationship.

The generic business model may be decomposed to reveal detailed entities and relationships.

A specific SDP business model derived from the generic model is shown in *Figure 2(b)*. The figure illustrates consumer decomposition into a service subscriber and end-user. Providers such as connectivity, application, service

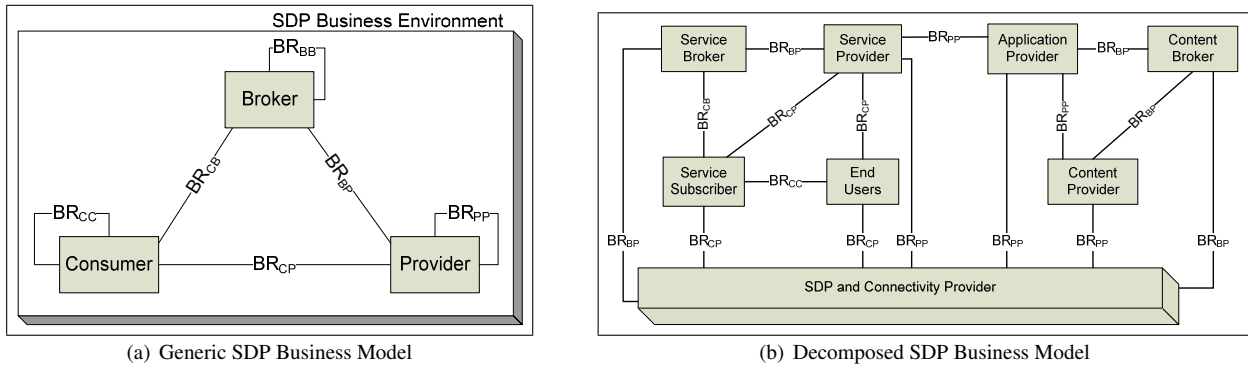


Fig. 2
SDP BUSINESS MODELS

and content providers are defined. Brokers such as service and content brokers are also shown.

In the example business model most business relationship points are used. For example, service subscribers obtain a list of services from service brokers (BR_{CB}). The service subscriber registers for selected services from a service provider (BR_{CP}). Service subscribers allow end-users to use services (BR_{CC}). End-users consume services from service providers (BR_{CP}). End-users use their connectivity provider (BR_{CP}). Service providers offer services on behalf of application providers (BR_{PP}). Application providers use content provisioned by content providers (BR_{PP}).

The SDP services provide the required functionality to implement these business relationship points.

E. Overall Management

The SDP contains management services that abstract and expose OSS/BSS functions. These management services are used via their interfaces by management applications. These applications implement telco business processes.

As an example, a complete set of telco business processes are defined in the enhanced Telecommunication Operations Map (eTOM) [16]. Also, limited services and

interfaces used to implement eTOM are defined by [17]. A management model illustrating business processes, services, interfaces and OSS/BSS are shown in *Figure 3*.

Apart from supporting business processes, the SDP itself is managed. More importantly, the SDP ensures traditional telco “quality requirements” [18] such as high availability, security, reliability, scalability and fault tolerance are maintained.

A model illustrating these management requirements are shown in *Figure 3*. The figure shows varying levels of management supported by the SDP. The business process management level administers the execution of business processes involving the telco and its customers. The service and application management level administers applications, services, interfaces, reliable communication, security, atomic transactions and life cycles.

The platform management level administers the interworking between services and infrastructure functions. This level ensures services use telco functions independent of technology, distribution and implementation. The network resource management level ensures the physical containers for telco infrastructure functions are available for use by SDP services.

F. Architectural Structure

The SDP integrates a distributed collection of telco and IT-based systems. Also, customers use the SDP and its services for various reasons. As a result, each system or customer perceives the SDP differently.

SDP perceptions are business oriented, customer oriented, service oriented and functional oriented. Also, each perception views the structure of the SDP differently. Hence, the SDP must exhibit a generic structure that is easily reconfigurable. To structure the SDP we use generic concepts, such as layers, domains and planes.

1) *Layers*: Layers are used in modeling many telco architectures, for example, TINA [19], Parlay and the IMS. Layers provide a means to *horizontally* group and structure a collection of related entities. In addition, layers model client-server communication between entities.

In telco architectures, layers are used to model switching levels and service levels. For the SDP, switching levels are

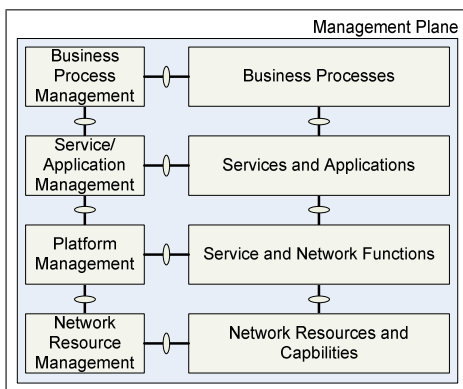


Fig. 3
SDP MANAGEMENT MODEL

called *Network Resource Layers*. These layers group physical network elements and their network-oriented functions.

For the SDP, service levels are called *Intelligent Service Layers*. These layers group SDP services. These service layers' services are realised as either building block services or composite services. A representation of SDP layers are illustrated in *Figure 4*.

Service layers use lower resource layers, by accessing their technology-specific functions. Service layers simplify access to these functions by using their technology-neutral service interfaces. Thus, service layers hide the complexities of using lower resource layers.

Besides abstracting lower resource layers, service layers simplify each other. For instance, multiple service layers are hierarchically structured with topmost service layers simplifying access to lower service layers. This represents abstraction of service layers by service layers.

Some service layers are also specific. For example, data stores located in a resource layer may have their functions abstracted by services in a separate intelligent service layer. As a result, this separate service layer may provide only data-centric services.

2) *Domains*: Domains represent areas of division across the telco network by ownership or functions. Domains are used to model various telco architectures such as IN, TINA, Parlay and TIPHON [20].

Domains *vertically* structure the distribution of layers and their components among areas of interest [18]. For instance, resource and service layers may intersect customer, network and management domains. Also, layers' components communicate synchronously and asynchronously across one or more domains. For example, communication may occur vertically across layers but within the same domain. Also, communication may occur horizontally across domains but within the same layer.

Similar components from various layers are grouped into specific domains. For example, components on the customer device are grouped within the customer domain. An illustration of domains is shown in *Figure 4*.

3) *Planes*: Specific layers and domains may be grouped together to focus on particular SDP concerns. These groupings are managed with the help of planes. Planes are reused in many telco architectures, such as the IN and IMS.

A particular plane used to structure the SDP is the *middleware* plane. The middleware plane focuses on abstract mechanisms used to support communication between services, that are distributed across layers and domains.

By providing the middleware plane, distributed services communicate independently of their location and implementation. Thus, the middleware plane hides communication complexities of the SDP, so as to simplify the definition of SDP functions, services and their relationships.

Planes group layers according to their similarities. Also, some layers may operate within distinct planes or across multiple planes. For example, layers abstracting service control functionality are grouped into a control plane. In addition, all layers (and planes) intersect at the manage-

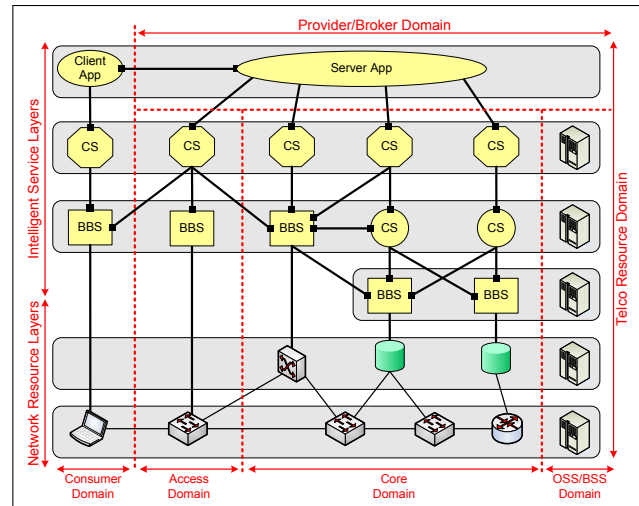


Fig. 4

STRUCTURING THE SDP USING LAYERS AND DOMAINS

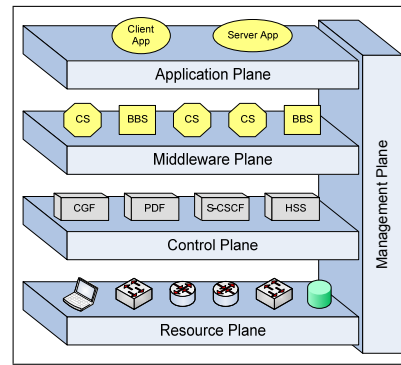


Fig. 5

STRUCTURING THE SDP USING PLANES

ment plane. An illustration of planes used in modeling a SDP are shown in *Figure 5*.

G. Standards-based Implementation

SDP architectures must be technology neutral. As a result, a SDP architecture is implementable using a variety of technologies. The technologies must implement SDP services, interfaces, applications, middleware and functions.

We motivate the use of open standards-based technologies for implementing the SDP. By using open standards, implementations remain consistent and *interoperable*. However, vendors may choose to implement the SDP using some proprietary technologies. If so, vendors must implement at least two parts of the SDP using open standards: the *north-facing* and *south-facing* SDP interfaces.

The north-facing interface represents the collection of services exposed to customers. Basic functionality exposed by these interfaces support application development. Hence, standardised north-facing interfaces promote portability of applications across SDP implementations.

The south-facing interface represents service access to telco resources and capabilities. Hence, standardised

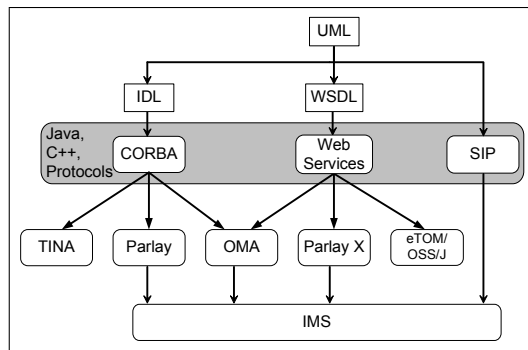


Fig. 6
EXAMPLE TECHNOLOGY MAP

south-facing interfaces enable portability of SDP implementations across different telco networks.

A variety of technologies are available to implement the SDP. We illustrate an example technology map used to implement the SDP. The map is shown in *Figure 6*. In the figure, the Unified Modelling Language (UML) [21] is used to describe and model the SDP independent of technologies. The UML is converted into various types of implementation neutral representations, such as Interface Definition Language (IDL) [22] or Web Services Description Language (WSDL) [23]. These representations are implemented using various technologies, such as C++, Java. Some representations are implemented using complex technologies, such as the Session Initiation Protocol (SIP) [24].

Many of the above technologies are incorporated into *standardised* telco and IT-based solutions, such as Parlay, the Open Mobile Alliance (OMA) [6] Service Environment [25] and the IMS service layer. These solutions include standardised architectures and technologies that may be easily incorporated into both SDP architecture and implementation. Thus, the SDP benefits from reusing and integrating these open standards.

IV. RESULTS AND CONCLUSION

We have defined the SDP concept and evaluated a proposed architectural interpretation. The proposed architecture is limited since it is derived from proprietary technologies and defines specific functionality. By extending the architecture, we have also shown common limitations with SDPs, that is, technology dependence and lack of standardisation. To overcome these limitations we extracted generic concepts from various SDP interpretations. These concepts define the architectural requirements of the SDP. The requirements focus on integration, services, interfaces, business and management modeling, layers, domains, planes and standardisation. Also, the requirements are independent of technology, implementation and distribution details. As a result, the requirements enable the development of technology neutral SDP architectures. Therefore, the requirements significantly contribute towards the SDP framework and ultimately standardisation of the SDP.

REFERENCES

- [1] The Moriana Group. *Service Delivery Platforms and Telecom Web Services*. Thought Leader Report, Last accessed on 01/04/2007, <http://www.morianagroup.com>, June 2004.
- [2] ISO/IEC 10746-1. *Open Distributed Processing Reference Model Part 1: Overview*. ITU-T Recommendation, May 1995.
- [3] J. Miller *et al.* *MDA Guide Version 1.0.1*. Specification omg/2003-06-01, The Object Management Group, June 2003.
- [4] Parlay Group Home Page. Last accessed on 01/04/2007, <http://www.parlay.org>.
- [5] Sun Microsystems Home Page. JSLEE and the JAIN Initiative. Last accessed on 01/04/2007, <http://java.sun.com/products/jain/>.
- [6] OMA Home Page. Last accessed on 25/06/2006, <http://www.openmobilealliance.org>.
- [7] 3GPP. *Technical Specification Group Services and Systems Aspects; Network Architecture (Release 7)*. Technical Specification TS 23.002 V7.0.0, December 2005.
- [8] Appium Home Page. Last accessed on 01/04/2007, <http://www.appium.com>.
- [9] Ericsson. *Service Delivery Platforms*. Last accessed on 01/04/2007, <http://www.ericsson.com/>, 2006.
- [10] Hewlett-Packard. *Service Delivery Platform*. Last accessed on 01/04/2007, <http://www.hp.com>, 2007.
- [11] SDP Alliance. *Service Delivery Platform*. Last accessed on 01/04/2007, <http://sdpalliance.mobilitydatasystems.com/>.
- [12] Microsoft. *Enabling Service Delivery using the Microsoft Connected Framework*. White paper, <http://www.microsoft.com>, January 2005.
- [13] G. Deckers. Cost Down, Revenues Up: SDP Business Case. In *Business Aspects of Service Convergence*, pages 178–183. 10th International Conference on Intelligence in Service Delivery Networks (ICIN), Bordeaux, France, May-June 2006.
- [14] IBM. *IBM Service Provider Delivery Environment: A Technical Overview*. IBM Telecommunication Industry White Paper, <http://ibm.com/industries/telecom/spde>, May 2005.
- [15] V. Radhakrishnan *et al.* *PIAF: An Application Framework for Unlocking IMS Engendered Network Capabilities*. In *Programming Models for the IMS*, pages 165–170. 10th International Conference on Intelligence in Service Delivery Networks (ICIN), Bordeaux, France, May 2006.
- [16] TMF. eTOM The Business Process Framework. Technical Report GB921, October 2001.
- [17] TMF OSS/J Home Page. OSS through Java Initiative. Last accessed on 01/04/2007, <http://www.tmforum.org>, 2006.
- [18] J. Bosch. *Design and Use of Software Architectures*. Software Engineering / Software Architecture. Addison-Wesley, 2000.
- [19] TINA-C Home Page. Last accessed on 01/04/2007, <http://www.tinac.com>.
- [20] ETSI. *Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON) Release 3; Abstract Architecture and Reference Points Definition; Network Architecture and Reference Points*. Technical Specification 101 314 v2.1.1, February 2002.
- [21] OMG. Unified Modelling Language (UML) Resource Page. Last accessed 01/04/2007, <http://www.uml.org>.
- [22] OMG Home Page. Last accessed 01/04/2007, <http://www.omg.org>.
- [23] E. Christensen *et al.* *Web Services Description Language (WSDL) 1.1*. Technical Report 1.1, W3C, March 2001.
- [24] J. Rosenberg *et al.* *SIP: Session Initiation Protocol*. RFC 3261, IETF Network Working Group, June 2001.
- [25] OMA. *OMA Service Environment*. Specification V1.0.2, August 2005.

BIOGRAPHIES

Rolan Christian holds a BSc and BSc Honors degree in Information Technology from the University of Natal. He also holds a MSc degree in Electrical Engineering from the University of the Witwatersrand. He is currently pursuing his PhD degree in Electrical Engineering at the University of the Witwatersrand.

Hu Hanrahan Hu Hanrahan is an Emeritus Professor at the University of the Witwatersrand. Formerly, he led the Centre for Telecommunications Access and Services (CeTAS), a research and advanced teaching centre devoted to improving knowledge and practice in telecoms access and services.