CHAPTER 1

INTRODUCTION

1.1 WEB SERVICES

Web services are garnering widespread acceptance in modern applications, and have revolutionized the way industry and public sectors operate. The use of web services on the World Wide Web (WWW) is growing rapidly, as the need for application-to-application communication and interoperability grows. Web services provide a standard means of communication among different software applications, running on a variety of platforms and/or frameworks (Seo et al 2009).

Web services are platform independent, self-contained, loosely coupled programmable web-enabled applications, that can be described, published, discovered, coordinated, and configured for the purpose of developing distributed interoperable applications (Tang et al 2011). Web services are versatile by design as they can be accessed by humans via a web-based client interface, or by other applications and other web services.

Web services provide many technological and business benefits (Cavanaugh 2006). They allow applications to communicate with any other application, and effectively exchange data without the need to know the underlying implementation or data formats. A service may be utilized by several clients to accomplish different business objectives. Hence, it becomes
possible to integrate disparate applications and data formats with relative ease.

Web services are designed to be accessed by other applications and can vary in complexity from simple operations, such as checking a banking account balance online, to complex applications like Customer Relationship Management (CRM) or Enterprise Resource Planning systems (ERP) (Khushali & Tirghoda 2012). Online stores such as, eBay and Amazon have benefitted by the deployment of web services, and become very popular with the customers. Web services have found application in diverse sectors, such as commerce, banking, insurance, education, government, and travel and tourism (Haidar & Abdallah 2006).

Web service architecture, therefore, enables the development of distributed applications which are dynamic and loosely coupled. The delivery of business applications as web services makes them accessible to anyone, anytime, at any location, and using any platform (Gonzalez et al 2009). The applications can be accessed from any kind of Internet ready devices, such as personal computers, workstations, laptops, smart phones, Personal Digital Assistants (PDAs), and even smart household devices fitted with sensors and having processing capability.

1.2 WEB SERVICES AND THEIR STANDARDS

A web service is defined as an interface, which describes a collection of operations that are network accessible, through standardized Extensible Markup Language (XML) messaging (Kreger 2001). The interface hides the implementation details of the service, allowing it to be used independently of the hardware or software platform on which it is implemented, and also of the programming language in which it is written.
Web services use XML at many different layers of operation, such as, data representation and data transportation layers. The use of XML introduces many characteristics like interoperability, platform independence, programming language independence, and operating system independence. XML can be used not only to represent data but also to represent complex documents. These documents can be simple like representing an address, or could be complex, such as representing an entire book. Web services support the transparent exchange of documents to facilitate business integration.

Web services are loosely coupled (Papazoglou 2008). If the logical implementation of a web service changes, it would not affect the application of the service requestor in any way. In a tightly coupled system, the client and the server logic are closely tied to one another, implying that if one changes, the other must also be updated. Unlike a tightly coupled system, the consumer of a web service is not tightly linked to the provider of the web service. The loosely coupled architecture tends to make software systems more manageable, and allows simpler integration between different systems.

Web services can be synchronous as well as asynchronous (Lea et al 2006). In synchronous invocations, the client blocks and waits for the web service to complete its operation. Asynchronous operations allow a client to invoke a service and then execute other functions. Asynchronous clients retrieve their result later, while synchronous clients receive their result once the service is completed. Asynchronous capability is a key factor in enabling loosely coupled systems.

The web service architecture is based upon the interactions among three components: the Service provider, Service registry, and Service requestor. From the business perspective, the service provider is the owner of the service and provides the platform that hosts the web service. The service provider is responsible for developing and deploying the web Services. The
provider defines the services and also publishes it in the service registry (Comert & Akinci 2003). The service requestor is a user or an application of the web service. The service requester searches the service registry for the desired web services. The service requester discovers the web service description in a registry and uses the information in the description to execute the service and obtain the results of processing. Figure 1.1 illustrates the web Services architecture, components, and their roles (Kreger 2001).

![Image of web services architecture components](image)

**Figure 1.1 Web Services Architecture Components**

The three operations in the web service architecture are publish, find and bind (Papazoglou & Dubray 2004). Publishing a web service in the service registry requires the service provider to describe the specifications of the web service. The web service descriptions that are publicly available can be used by service requestors to access the service. Web services are defined using the Web Services Description Language (WSDL). The service provider provides three types of information: business information on the web service provider; service information on the nature of the web services; and technical methods on how to invoke the web service. The technical information
includes the details of the interface such as: data types, operations, binding information and implementation of the service (Yu et al 2008).

The find operation is used to search or query the registry, to locate web services that match the business needs and requirements. The desired web service may be selected on the basis of various criteria, including accessibility and cost (Kreger 2001). The bind operation is used by the service requester to invoke the web service at runtime. The binding details in the service description are used to locate, contact and invoke the service.

For a wide acceptance of the web services, it is necessary to have a set of clearly defined and widely adopted standards (W3C Stanadard). The web services stack shown in Figure 1.2 describes the layers of components and standards, used for the development and deployment of web services (Kreger 2001).

![Web Services Protocol Stack](image)

Figure 1.2 Web Services Protocol Stack
Web services must be network accessible to be invoked by a service requester. The network protocol used in any given situation depends on the application requirements. Web services consider the use of a wide range of transport protocols, the most common one being Hypertext Transfer Protocol (HTTP). HTTP has become the de facto standard application level protocol for Internet-available Web services. Other Internet protocols, however, like Simple Mail Transfer Protocol (SMTP) and File Transfer Protocol (FTP), can also be used. For web services being provided and consumed within an Intranet, alternative application level technologies can also be used. The application level technology can be chosen based on other requirements, including security, availability, performance and reliability. Web services, therefore, provide a unified model for the development and usage of private Intranet as well as public Internet services. From the point of view of the web service developer, the communication network is hidden behind the transport protocol (Gehlen et al 2006).

The next layer provides a standard way to format and package the information that is exchanged. Simple Object Access Protocol (SOAP) is the standard messaging protocol used by web services. SOAP is based on message exchanges. Messages are seen as envelopes where the application encloses the data to be sent. SOAP provides a message format describing how a message can be expressed as XML document. SOAP message consists of a header, body and fault elements. The header element is optional and can be used for containing application specific information. The body element is mandatory, and contains the actual SOAP message. SOAP provides a model for handling faults that arise during the exchanges and the fault element, which is optional, is used for errors and status information. The SOAP standard also provides a description of how a SOAP message should be transported, and a set of rules for processing the SOAP message. SOAP specifies a generic message template for the application data. Additional
specifications are used to standardize ways in which to implement particular features in SOAP.

It is through the service description that the service provider communicates all the specifications for invoking the web service. Authentication involves validating to the service requestor. Web services become easy to use when a web service and its client rely on standard ways to specify the data, operations to understand the capabilities that the web service provides. The functionality of web services is described using Web services Description Language (WSDL). The WSDL uses XML grammar for describing web services as collections of communication end points that are capable of exchanging messages. The service requesters wishing to access a web service can read, and interpret its WSDL file to know about the location of the service and its available operations. The WSDL description divides the basic service description into two parts: the service interface, and the service implementation as shown in Figure 1.3. This enables each part to be defined separately and independently, and reused by other parts.

![Figure 1.3 Web Service Description](image-url)
A service interface definition is an abstract service definition that can be instantiated and referenced by multiple service implementation definitions (Kreger 2001). The service interface definition describes the messages and operations in a platform and language independent manner. The service interface description consists of four elements: Type, Message, Port Type and Binding. Complex data types can be specified using the Type element. The Message element is used to specify the input and output parameters of an operation. The Port type element groups and describes the operations performed by the service through the defined interface. The Binding element describes the protocol, data format, security and other attributes for a particular service interface.

The service implementation definition describes how a particular service interface is implemented by a given service provider. A web service is modeled as a service element, which contains one or more port elements. A port associates an end point (for example, a network address location or Uniform Resource Locator (URL) with the binding element from the service interface definition.

A service requester gets the web service details from the WSDL document published in the service registry by the service provider. The Universal Description Discovery and Integration (UDDI) is a standard sponsored by OASIS (Organization for the Advancement of Structured Information Standards). UDDI is a specification for creating an XML-based registry that lists information about businesses and the web services they offer. UDDI provides businesses, a uniform way of listing their services and discovering services offered by other organizations. UDDI registries can be queried by service requesters to obtain details of web services which they wish to utilize.
The information in the UDDI registry is categorized into White pages, Yellow pages and Green pages. (Chappell & Jewell 2002). White pages contain basic contact information and identifiers about a company, including the business name, address, and contact information. This information allows others to discover the web service based upon the business identification. Yellow pages consist of information that describes web service taxonomies. Green pages provide technical information, like the location of service, and describe the behaviors and supported functions hosted by the business.

UDDI also defines a data structure standard for representing service and service description information. The Basic UDDI data structure is shown in Figure 1.4.

![UDDI Data Structure Diagram]

**Figure 1.4 UDDI Data Structure**

The UDDI data structure defines four core types of information. This includes:

- Business Entity: a description of the organization that provides the service.
• Business Service: Business service entity includes a description of the service, a listing of categories that describe and classify the service, and a URL referencing further information about the service.

• Binding Template: describes the technical aspects of the service being offered.

• tModel: (“technical model”) is a generic element that can be used to store technical information on how to use the service, conditions for use, guarantees, etc.

1.3 AUTHENTICATION, AUTHORIZATION AND ACCOUNTING (AAA)

Authentication, Authorization, and Accounting (AAA) refer to the security architecture for distributed systems that is used for controlling access to services, and tracking the resource usage (Goranthala et al 2012). AAA systems were first introduced for telephonic services, and later developed to control access to packet switched networks. They have become a core component in Internet Service Provider companies (Decugis 2009).

Authentication asks the question, “Who or what are you?” Authorization asks, “What are you allowed to do?” and finally, accounting wants to know, “What did you do?” Authentication involves validating the end user’s identity prior to permitting them to access the network. Many different types of identification credentials can be used, such as, username-password combination, a secret key or biometric data. The AAA server compares the user-supplied authentication data with the user-associated data stored in its database, and if the credentials match, the user is granted network access. A mismatch results in an authentication failure, and a denial of network access.
Authorization, defines what rights and services the end user is allowed, once the network access is granted (Aboba et al 2000). Accounting measures the resources that a user consumes during his access (Jimenez et al 2009). Accounting is carried out by the logging of session statistics and usage information such as: the identity of the user and nature of the service delivered (Goranthala et al 2012). This information is used for charging, billing and for collecting payment.

In general, AAA involves mechanisms, protocols and architectures. Mechanisms are methods to perform authentication, authorization, and accounting; AAA protocols specify appropriate interaction schemes for a distributed system. Finally, AAA architectures address the interworking of the various components.

The development of the Remote Authentication Dial-In User Service (RADIUS) protocol is often considered as the genesis of AAA. RADIUS, developed by Livingston Enterprises (now part of Alcatel-Lucent) in the early 1990s, became an Internet standard through the Internet Engineering Task Force (IETF) in 1997, and was one of the most widely accepted AAA protocols. RADIUS was defined in RFC 2865 and saw many revisions (Rigney et al 1997). RADIUS provides Authentication and Accounting services for the Internet, and is also widely used by Voice over IP (VOIP) service providers.

Many other protocols were developed to support the tasks of authentication, authorization and accounting. The Diameter protocol was developed in 1998 to provide a framework for AAA that could overcome the limitations of RADIUS (Sargento et al 2003). The Diameter protocol provides an AAA framework for network and IP applications. (RFC 6733). It supports stronger security through either Internet Protocol Security (IPsec) or Transport Layer Security (TLS) (RFC 6733).
Standards for AAA systems were proposed originally for telecommunication, and later on for the Internet. Standards organizations have played an important role to guarantee interoperability between vendors (Xavier et al 2013). IETF is the most referenced open standard that was the one established in 1986, to coordinate the operation, management and evolution of the Internet. The need and significance of standard protocols for the exchange of usage data and accounting attributes was standardized by the IETF. It is directly responsible for the development of the majority of Internet and TCP/IP standards.

The Internet, proliferation of mobile devices, diverse network consumers, and varied network access methods, and emerging service paradigms, have combined to place an even greater demand on AAA. Standardized and efficient solutions are required to support reliable, secure, open, and flexible remote, mobile and other service access (Convery 2007). The diversity of applications for Internet’s AAA is increasing. New applications with specific requirements on the AAA framework are also emerging, such as distributed games, geo-location, or convergence with the television industry.

AAA protocols should be generic enough to transport different kinds of information such as authentication, authorization, accounting and other related data. Protocols must be defined on the generic AAA framework to achieve authentication, authorization, and accounting, and to sustain scalability (Decugis 2009). It has been widely recognized that appropriate mechanisms are needed for collecting chargeable events in order to process the received information according to the business models adopted by the involved entities. Similar needs have been identified in the mobile telecommunications world, especially after the tight integration of the IP protocol in the mobile network infrastructure.
There is a need for extension, enhancement, and modification of AAA protocols that needs to be integrated, for the accounting of web services. AAA protocols need to support policy control, dynamic rules, and quality of service, bandwidth allocation, and new charging schemes, to adapt to emerging technologies.

1.4 MOTIVATION AND OBJECTIVES

Accounting, according to the definition of IETF is the process of collecting the resource usage information for the purpose of billing, trend analysis etc. (Aboba et al 2000). More and more enterprises are shifting their business processes to web service based distributed applications. Commercial web service models are required to support Business-to-Business (B2B) and Business-to-Consumer (B2C) transactions that measure web services for the purpose of charging and billing.

The ability to effectively and efficiently charge for the consumption of resources represents an important economical requirement for any service provider. Free service usage and flat-rate monthly billing model were initially used by service providers to attract a large number of online customers. However, there is a growing need for service providers to charge their users based on their service usage. Further, with exploding usage, the service providers needed to differentiate their services, offering varying levels of Internet access and Quality of Service (QoS). It therefore becomes critical for service providers to be able to measure exactly when, what for and how much, their services are being used. Usage data statistics would enable them to accurately price their differentiated services, improve margins, and increase profitability.

Rapidly growing web service usage creates measurement challenges. Usage measurement and billing are critical operational support
activities for a web service provider. Charging models based on usage, content or transaction should ideally be introduced not only to cover the cost, but also to earn revenue from the services consumed. Web service meters have to be deployed to record the session time, data downloaded or transferred, as well as the usage of other resources. Enabling such metering and accounting components for web services, is an important and essential component of web service accounting and management.

The challenges related to web accounting can be attributed to the distributed nature of the web services. AAA is complex as the web services are distributed in several locations. AAA protocols are also required between the entity that provides the service and the server(s) where the AAA data is located. There is an increasing need for AAA services to also include Charging and Auditing (AAAAC). Extensibility to functions beyond AAA, like charging and auditing is complicated, since the components are not defined in a generic way. Current AAA architectures, protocols, and implementations do not cope with heterogeneous application scenarios and requirements related to different provisioning of services.

Existing accounting systems like that of Amazon Web Services (AWS) are propriety and developed for the specification application. Other applications like the Kaplan online tutoring have offer specific pricing packages like pre-paid. Pricing model, Priceline model (Priceline.com) describes a Name -Your –Own –Price (NYOP) system where a buyer specifies a price for a product. However if no dealer accepted the user price, the user gets nothing.

There is a need for a comprehensive accounting management system that covers the process of metering, charging, accounting, billing, payment and auditing has become essential for web services. Currently, there are no accounting protocols defined specifically for web service accounting,
and there is no standard format defined to record the service usage data. The main objectives of this research work are:

- To propose an integrated accounting architecture to automate the accounting functions of web services
- To support different pricing models of service providers
- To design a metering framework to measure and record the service usage information of web services
- To design, develop and integrate the modules for billing, payment, accounting and auditing

1.5 CONTRIBUTION SUMMARY

The research work proposes an integrated accounting architecture that covers all aspects of accounting: metering, charging, accounting, billing and payment. The contributions of the research work have been summarized below:

- An integrated accounting architecture has been proposed to automate the accounting functions of web services. Prior work has discussed metering and accounting functions but implementation issues have not been dealt with. The research work has aimed to design and develop an integrated accounting system with modules for billing, payment, accounting and auditing.

- The proposed accounting architecture supports different pricing schemes of various service providers. Prior work has discussed and analyzed the need for different payment possibilities. Existing applications like Kaplan online tutoring offer fixed
packages like pre-paid. AWS supports different pricing schemes which are specific to their application products.

- Charging and Accounting policies have been proposed in the research work to support the different pricing schemes. WS-Policy standard has been proposed to define the policies. A tool has been developed to automate the creation of charging and accounting policies.

- WS-RADIUS protocol has been developed to perform the metering function for web services. This protocol modifies and extends the standard AAA Radius protocol, which was developed for telecommunication and Internet. COPS protocol primarily supports policy control that supports the exchange of policy information between a policy server and its clients. It was primarily defined for use with quality of service protocols such as RSVP.

- The format and content of web service usage data, bills and payment have been designed. Prior work has discussed the measurement issues for service oriented computing, and has suggested the use of logs to store the metered data. The research work has proposed the use of Internet Protocol Detail Record (IPDR) standard to standardize the log formats.

1.6 THESIS STRUCTURE

A brief outline of the various chapters of the thesis is as follows:

Chapter 2 discusses the related work with respect to accounting in general and with respect to web services. AAA standards and protocols have
been explained. Accounting systems related to telecommunication, mobile networks, cloud and grid have been described. Literature review of work relating to web service accounting is presented.

In **Chapter 3**, the proposed integrated accounting architecture for web services is explained in greater detail. This chapter explains with examples the charging and accounting policies that can be used to define the different pricing schemes offered by service providers. Policy generation tool which has been developed to create policies in WS-Policy format is explained. Query facility for the policies has been explained with the use of XQuery.

In **Chapter 4**, the design of the integrated accounting architecture is presented. The components of the integrated accounting architecture have been explained. Metering of web services usage using WS-RADIUS protocol has been explained. IPDR standard has been proposed for recording the service usage and accounting data. The format and content of the data to be maintained is discussed.

**Chapter 5** discusses case studies that illustrate the varying charging patterns required by commercial applications. The charging patterns required by different applications, such as, online newspaper, online educational system and online gaming have been discussed.

**Chapter 6** concludes the thesis. This chapter presents a detailed summary of the contributions of the research work. A critical analysis of the proposed research work has been provided. Suggestions for future work have also been presented.