IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITJ2806

The Deployment of Protocol Stack Components Using Web Services

Kevin Curran, University of Ulster, Ireland
Brendan Gallagher, University of Ulster, Ireland

ABSTRACT

Multimedia has varying optimal transport methods. The traditional methods employed by transport protocols are to ship all data through identical protocol stacks. An ideal method would transport each media through an optimized stack constructed solely for that medium, allowing improved multimedia QoS to be achieved even at runtime. Dynamically composable protocol stacks overcome the limitations imposed by generic protocol stacks. A dynamically composable protocol stack allows optimization for particular traffic. Here, protocol layers such as real-time G.711 or PCM conversion capabilities could be deployed to address the impedance matching across heterogeneous receivers. Protocol layers are created by the protocol stack (using a Protocol Profiler) according to a properties argument defined when creating an instance of Stack. The SOAP is a lightweight remote procedure calling protocol for the exchange of structured data in a decentralized environment. SOAP enables programs to run and interoperate with other SOAP applications (called Web services) in a distributed environment. The SOAP protocol is based on eXtensible Markup Language (XML) and Hypertext Transmission Protocol (HTTP), which, it is claimed, makes it a language- and platform-neutral vehicle for RPC over the Internet and through firewalls. This paper describes a SOAP Web service deployed on an apache Tomcat server that enables clients to download protocol stack components as MIME attachments. The deployment middleware framework is named Webber. Webber provides additional flexibility that can be extremely useful for environments that have not been considered by standard generic protocols.

Keywords: distributed multimedia; middleware; SOAP; Web Services

INTRODUCTION

The idea of a truly open system is the "Holy Grail" for every altruistic application and Web developer. This perhaps is why SOAP's apparent language- and platform-independent architecture has excited the world of software development. Many devices such PDAs, pocket PCs, and mobile phones can access the Internet. To access the Internet, these devices must use networking protocols similar to those used by the average desktop computer. These networking protocols, called a protocol stack, are layered such as File Transfer Protocol (FTP), Hypertext Transmission Protocol

This paper appears in the journal *International Journal of Web Services Research* edited by Liang-Jie Zhang.

(HTTP) and Transmission Control Protocol (TCP). A protocol stack consists of a linear list of protocol objects, which between them can support a range of quality of service such as reliable delivery, virtual synchrony, or encrypted communication. Our framework—Webber—provides the interlayer services necessary for supporting new communication protocols. Webber consists of a set of Java classes for representing Uniform Resource Locators, protocol stacks, the framework API, and posting objects. Dynamically composable protocol stacks overcome the limitations imposed by generic protocol stacks, allowing optimization for particular traffic. Some uses for which dynamic protocols may be used include using the best-fit protocol for a particular network and application behavior, so that performance always can be optimal; and upgrading network protocols at runtime without having to restart applications and increasing security at runtime.

The use of an object-oriented implementation language allows us to extensively use object-level design patterns. This makes any framework more generic and extensible, and creates a highly stylistic way for writing actual protocol implementations. With a suitable object-oriented design tool, the outline for the classes needed to implement a new protocol can be created quickly. The actual implementation code for the protocol actions typically takes a little longer, depending on the complexity of the protocol. Performance always will be an issue with communications protocols. Even though processing power is constantly increasing, the new applications need ever-increasing bandwidth and reasonable transfer delay. The new protocols require large transfer capacity, short and fixed delay, and lots of cryptography, among other things. There are two facets to performance. First, the processing power available should be used as efficiently as possible. The importance of this will gradually decrease as processing power increases. Second, and more important, there should not be any design limitations that set a theoretical limit to the performance of the protocols, no matter how much processing power we have. We want to allow as much parallelism as possible and build the protocol implementations such that they can be efficiently divided among a number of processors. Java, with its built-in threads and synchronization, allows parallelism to be utilized with relative ease. Central to providing an adaptable QoS is the ability to maintain multiple protocol stacks. A protocol stack consists of a linear list of protocol objects and represents a quality of service such as reliable delivery or encrypted communication. We have developed a framework called Webber, which provides the services necessary for supporting new communication protocols and qualities of service. Webber consists of a set of Java classes for representing Uniform Resource Locators, protocol stacks, the framework API, and SOAP.

SIMPLE OBJECT ACCESS PROTOCOL (SOAP)

SOAP is a lightweight XML-based protocol for exchange of structured data in a decentralized, distributed environment[1]. SOAP is a standard way of regulating data transmission between computers. The XML-based protocol is language- and platform-neutral, which means that information can be shared and messages passed among disparate parties across different platforms, languages, and programming environments. SOAP is not a competitive technology to component systems and object-request broker architectures such as the CORBA and DCOM, but instead

A Similarity Measure for Process Mining in Service Oriented Architecture

Joonsoo Bae, Ling Liu, James Caverlee, Liang-Jie Zhang and Hyerim Bae (2010). *Web Services Research for Emerging Applications: Discoveries and Trends (pp. 87-103).*

www.igi-global.com/chapter/similarity-measure-process-mining-service/41519?camid=4v1a