MSCML Protocol: The Key to Unlocking a New Generation of Multimedia SIP Services

Eric Burger
Vice President
CTO of Next-Generation Communications

Greg Pisano
Market Development Director
Carrier Enhanced Services

cantata
TECHNOLOGY

table of contents

cantata TECHNOLOGY

abstract

Multimedia communications are quickly evolving as carriers transition from traditional circuit-switched networks to next- generation IP infrastructures. This shift in network topologies is allowing service providers to introduce a new wave of innovative multimedia services made possible by IP communications technology. Cantata Technology is in the forefront of delivering highly scalable and flexible media processing solutions by driving standards that support enabling network architectures such as IMS upon which this new generation of IP multimedia services can be built.

Control and signaling protocols are critical to the success of these new services. New IP protocols have been developed and deployed to provide the necessary glue for the development, integration and management of a new generation of IP services and infrastructure. Cantata through its acquisition of SnowShore Networks in 2004 pioneered the development of two key protocols to address this need. This document discusses these protocols and how they interact with other standard protocols in enabling a new generation of IP-based applications.

introduction

The Media Server Control Markup Language (MSCML) is a protocol used in conjunction with the Session Initiation Protocol (SIP) to enable the delivery of advanced multimedia conferencing services over IP networks. A complementary protocol that also functions with SIP is the Network Announcement (NETANN) protocol, which provides simple announcement and basic conferencing services. Both MSCML and the NETANN protocol were originally authored by SnowShore. In December 2005, NETANN was published by the IETF under RFC 4240. The MSCML specification has been thoroughly reviewed by the SIPPING WG and IESG, and is currently in the IETF RFC editor's queue expected to be published as an RFC in 2006.

MSCML adds value to the SIP protocol

SIP is a broad, application-level protocol for setting up, changing and terminating multimedia sessions between participants. However, as developers began building SIP applications, and, in particular, enhanced IP conferencing applications, it became clear there was a need for enhanced control for multimedia conferencing services.

Prior to MSCML, IP conferencing application developers grappled with ways to develop conferencing services to deliver a full range of advanced features. SIP provided the basic tools for simple conferencing functionality, but its capabilities fell short of providing sophisticated control functionality. MSCML provides a solution to this problem by enabling application developers to easily build and extend conferencing from simple three-way conferences to robust and scalable multi-party conferencing applications.

MSCML leverages the proven SIP protocol, which is a flexible call control and signaling protocol currently driving the delivery of converged services in next- generation networks. SIP was developed through the Internet Engineering Task Force (IETF) for establishing real-time IP session connectivity across diverse networks. These sessions consist of end-to-end connections between two or more intelligent endpoints. SIP brings three key capabilities to telecommunications:

- **Innovation:** by leveraging the proven web development and deployment model and the ready availability of programming talent and tools.
- **Scale:** since SIP is inherently distributed.
- **Ubiquity:** because SIP is based on established protocols and has access to a large body of open implementations, it ensures availability of a vibrant, competitive set of offerings.

MSCML in the applications and services architecture

SIP and MSCML are used to develop and deploy services within the IP applications and services architecture. This network topology consists of application servers and media servers which work together in a client-server relationship, with application servers *(clients)* providing the service logic for each specific application and the media server *(server)* acting as a shared media processing resource for the applications.

The media server operates as an independent entity, managing and allocating its processing resources to match the requirements of each application. Its primary role is to handle requests from the application server for performing media processing on packetized media streams. It is not a peripheral or unaware device within the network, but is a self-managing, multimedia processing platform utilized by multiple applications to power each respective service.

Application servers interface with media servers today largely using SIP for call control and VoiceXML for supporting advanced IVR applications. From an application development perspective, MSCML is increasingly being used to develop and deliver advanced conferencing features to SIP-based applications. When these services are deployed, MSCML enables application servers and media servers to communicate in the pure peer-to-peer relationship of the client-server model.

This long- established architectural paradigm allows network operators increased scalability and flexibility when developing and deploying IP services. This approach contrasts with other signaling protocols that orient the application server and media server differently in a more rigid, inflexible master-slave relationship.

The client-server model inherently enables multiple applications to share the same processing resources of the media server, providing greater economies of scale for service providers. Since the media server is a self-managing entity, it can dynamically allocate processing resources to match the needs of each application and offer network operators a greater degree of deployment flexibility.

how MSCML works

From a programming standpoint, the use of SIP and MSCML reduces application development complexity and gives developers increased flexibility to add new features. For example, in a simple SIP conference, basic SIP constructs establish basic conferencing sessions among any number of participants.

Destinations in SIP are represented with Uniform Resource Indicators (URIs), which are formatted similarly to email addresses. The SIP URI provides a natural mechanism for identifying a specific SIP conference, while INVITE and BYE methods elegantly implement conference join and leave semantics.

MSCML serves as the interface between an application server and a media server and extends the basic SIP conferencing session into an enhanced conferencing session. MSCML enables enhanced conference control functions such as muting individual callers or legs in a multi-party conference call.

Other control functionality enabled by MSCML includes the ability to increase or decrease the volume on a leg or on the call and the capability to create sub-conferences. MSCML also addresses other feature requirements for large-scale conferencing applications, such as sizing and resizing of a conference,

incorporating in-conference Interactive Voice Response (IVR) operations *(e.g. recording and playing participant names to the full conference)* and conference event reporting. MSCML was developed to uniquely address these requirements while fitting into SIP's net-centric application model.

MSCML and NETANN

MSCML works with Network Announcement (NETANN) to provide enhanced conferencing services. NETANN is a general SIP convention (protocol) used for initiating specific services on a multifunction media server. Conversely, MSCML is a control protocol specific to enhanced conferencing.

Application servers and softswitches use NETANN to access basic application functions, such as announcements, media mixing and the prompting and collecting of user information from the media server. NETANN follows established protocol design principles. By design, it is not an enhanced conferencing protocol but simply a protocol that supports the basic media services required by media-rich applications.

MSCML provides developers with a framework to add enhanced conferencing features. For example, MSCML uses the SIP conventions described by NETANN to determine when to start an enhanced conference, add a leg to a conference, drop a leg from a conference and terminate a conference.

NETANN is required regardless of what other extension is used—whether the developer uses MSCML, VoiceXML or some less-functional or less established alternative. This approach is the embodiment of a very important concept in the development of Internet protocols—the use of separate protocols for discrete functions. This is why there is no file transfer capabilities specified in SIP and no media control available in NETANN. NETANN addresses a very specific challenge: how to identify different resources on a media server that is being used for multiple functions.

VoiceXML and CCXML

VoiceXML is an audio interaction language based on XML. Developers use VoiceXML to create interactive voice response (IVR) applications. By design, VoiceXML has no call control. In particular, VoiceXML has no provision for conference control. For example, VoiceXML allows the developer to define when to play a prompt or when to select or provide information. However, VoiceXML is incapable of providing higher-layer, call-oriented services efficiently—such as requesting the conference to mute a leg. MSCML addresses this need and provides the conferencing control functions necessary to support these enhanced features.

CCXML (Call Control eXtensible Markup Language) is an application language that provides the advanced call control absent in VoiceXML for the setup, monitoring, and tear down of phone calls. Although, VoiceXML does support limited call control features, its features are too basic for many applications. Cantata helped drive the formation of the CCXML working group and wrote the requirements document. CCXML is an application-level markup language. That is, it runs on the application server. MSCML, by contrast, runs on media servers. CCXML draws on MSCML constructs making the mapping between the CCXML markup and the MSCML protocol easy for advance call control and conferencing capabilities.

MSCML, VoiceXML, and CCXML are complementary technologies. MSCML provides enhanced conferencing, VoiceXML enables enhanced IVR, while CCXML provides advanced call control. MSCML adds value to SIP applications by providing developers with the tools to offer more advanced features for SIP conferencing applications. Moreover, it enables new converged applications, mixing the rich media IVR experience of VoiceXML with enhanced conferencing facilities. MSCML has already benefited from widespread market adoption within the vendor and service provider community, and it is increasingly being recognized as the proven protocol choice for IP enhanced conferencing.

why MGCP, H.248, MOML/MSML are insufficient

Occasionally, Cantata is asked about other media processing protocols that are available. This section will discuss these other protocols and why they are insufficient for conferencing.

SIP is a peer-to-peer call control protocol that uses text encoding and the web development model. Text encoding means it is very easy to develop and debug the protocol. The web development model means that developers familiar with client-server protocols such as HTTP or SIP and markup languages such as HTML will be comfortable with SIP and MSCML.

This contrasts with call control protocols such as MGCP and H.248 (Megaco). MGCP and H.248 are nearly identical in purpose and very similar in syntax to each other. These protocols, as well as two proposals, Media Objects Markup Language (MOML) and Media Sessions Markup Language (MSML), are used by call-aware control entities *(application servers)* within the network to control unaware devices *(media servers)* in a master-slave relationship. This is in contrast to SIP and MSCML, which let developers and service providers use and leverage the client-server relationship.

SIP and MSCML are flexible signaling and control protocols for applications and operate at the application layer. However, MGCP and H.248 and their two sibling proposals MSML and MOML are transport oriented control protocols. MOML and MSML have been proposed as "SIP services." However, they are, in effect, MGCP-like protocols using XML encoding and using SIP for transport. These protocols may follow the same form as SIP, but they mirror MGCP and H.248 protocols in terms of function. The end result is that these proposals suffer the same inherent weaknesses of MGCP and H.248 that make them an insufficient choice for developing and deploying IP conferencing services. Moreover, they abuse the SIP client-server semantics. In the end, the protocols have all the drawbacks of MGCP and H.248 with the added burden of XML encoding of low-level primitives—without any of the benefits of SIP or XML.

If one takes a closer look at the MOSL and MSML proposals in detail, their technical weaknesses become apparent because they:

Violate Layer Integrity: During a conference session, MOML and MSML reference calls by IP address and port number. This limits their applicability to a fixed transport structure, in this case IPv4. Conversely, MSCML provides an example of proper protocol design. It adheres to established layered protocol design principles and SIP conventions. MSCML uses the Session Description Protocol (SDP) and SIP dialog identification to describe a call leg. What this means from a practical point of view is that MSCML is ready today to support non-IPv4 environments—such as IPv6, as specified by 3GPP for the IMS—without modifications to the existing application.

Use the Media Server as a Slave Device: The media server in the MOML and MSML model is a slave device to the application server. Each application server manages its own processing resources on the media server, constraining the media server's ability to independently optimize its resource usage because the media server's resources are locked to each individual application. This contrasts to the client-server relationship of MSCML, where the media server manages its resources independently, enabling higher scalability and flexibility. The client-server model enables application servers—which are clients to the media servers—to scale more effectively because there are fewer messages to manage, fewer states and a more efficient allocation of processing and networking resources.

Rely on Low-Level Primitives: Hand-in-hand with the layer violations and slave peripheral orientation of MSML/MOML is the use of low-level primitives. Application developers usually work in the domain of the application. In the case of conferencing, this means working with conferences, conference legs, digit collection, prompting and recording. However, MSML/MOML offers primitives like DTMF detectors, media players, media recorders, star connectors, fork connectors, mixers and so on. The developer must plumb these devices together to create services like conferences, causing undue complexity and longer and more costly development cycles.

Increase Complexity: Due to this low-level approach, MOML/MSML makes application development more complex. This results in more lines of code and testing difficulties. The use of MSML/MOML makes new applications more error prone and lengthens the development time with little tangible benefit. MSCML builds on the development environment of SIP, HTTP and XML in which the mixing resource is a server that operates on application-level constructs such as call participants. It therefore allows developers and operators to write applications at a higher level so they are not concerned with plumbing but are in fact concerned with the critical capabilities of the applications—such as how they manage multimedia services or how they add or drop call legs. Due to this simplicity, application and media server vendors have extended MSCML to build other advanced features and services including fax, video and IVR functions.

The transport-oriented approach of MGCP, H.248, MOML and MSML makes application development needlessly complex and puts the media server into a diminished role as an "unaware," peripheral slave device. This often leads to longer and more expensive application development cycles for developers and reduced flexibility, performance and scalability of the network infrastructure and applications for service providers. Unlike MOML and MSML, MSCML enables developers to develop and deliver robust, feature-rich advanced conferencing applications with greater simplicity, scalability and performance.

the MSCML protocol has been interoperability tested, deployed and field proven

The MSCML specification was written in 2000 by technologists from SnowShore Networks who saw the need for designing MSCML after consulting with numerous application developers and service providers. Their encouragement and support led to the development of MSCML The protocol was deployed in field trials in 2001 and submitted to the IETF in 2002. In 2005, the IESG approved the publication of MSCML.

intellectual property and licensing

Cantata will grant a royalty-free license to our intellectual property for the use in MSCML.

There is only one circumstance in which we will not offer a royalty-free license. This situation occurs if all of the following conditions are true:

- *You have intellectual property rights that cover MSCML, and*
- *You assert those rights on anything other than a royalty-free basis, either:*
- *Against Cantata, or*
- *Against any user or manufacturer of a product using MSCML, whether they created it, acquired it from Cantata, or acquired it from any other third party.*

Cantata strongly believes in the standards process, and that standards should be free from barriers to deployment, such as encumbering the standard with royalty payments. One might ask why we filed for a patent on a technology we clearly were going to submit for standardization. The reason is simple: We need to protect the standards community from unscrupulous companies that would try to extract royalty payments from MSCML.

Just because one party does not have intellectual property in a protocol does not protect people from other parties with intellectual property in the protocol. If another party has intellectual property rights in MSCML, then our license offers some protection to MSCML users.

Note that we extend this protection whether or not the user acquires or builds their products that use MSCML from Cantata.

As network operators continue to transition from circuit-switched to IP networks, protocols such as MSCML are crucial to helping them realize the inherent benefits of an IP network architecture—a flexible and open services infrastructure that enables the deployment of innovative, feature-rich and profitable services.

Network operators can deploy MSCML today and rest assured that this specification is widely adopted and already proven in real-world environments. It is a simple, robust protocol that enables conferencing vendors and service providers alike to move forward and innovate within a proven development framework.

an open standard

As part of its ongoing drive to fuel the growth of innovative services, Cantata has stated to the IETF that it will offer royalty-free licenses of its intellectual property necessary for implementing the MSCML standard. Specific information regarding this offer may be obtained from the IETF. This enables IP application developers, infrastructure vendors and service providers to bring to market new IP enhanced conferencing and innovative services within the universal framework of MSCML.

It is important to note that MSCML is being adopted by other media server vendors as well as vendors of application servers and other critical equipment running on next-generation network infrastructures. Industry analysts and vendors alike view MSCML as the essential protocol for enhanced conferencing call control between the media server and application server in an IP services architecture.

Corporate Headquarters
410 First Avenue
Needham, MA 02494
USA
Tel: +1 (781) 449-4100
Fax: +1 (781) 449-9009
Email: info@cantata.com

Cantata Technology maintains multiple locations worldwide in North America, Asia and Europe.

www.cantata.com