Taking an Inside Look at TDMoIP: A Tutorial

VoIP has fallen short on the promise of providing a vehicle for delivering telephony traffic over IP networks. Fortunately, TDMoIP provides an alternative by transporting encapsulated TDM data over packet networks. Here's a look at how TDMoIP works.

By Yaakov (Jonathan) Stein, RAD Data Communications; and
Brian Stroehlein, TranSwitch Corp.

CommsDesign.com
March 26, 2003 (9:19 a.m. EST)

Service providers are presently seeking to increase their profits through low cost deployment of voice and leased line services over more efficient Ethernet and IP infrastructures. At the same time enterprises are looking for ways to take advantage of the promise of convergence by integrating their voice and data networks while preserving their investment in traditional PBX and TDM equipment. The voice-over-IP (VoIP) approach is maturing, but its deployment requires a certain level of investment in new network infrastructure and/or customer premises equipment (CPE).

TDM-over-IP (TDMoIP) is a technology that enables voice and leased-line services such as video and data to be offered inexpensively over service provider IP networks while retaining the reliability and quality of the public switched telephone network (PSTN). In this article, we'll discuss the technical challenges inherent in transporting TDM circuits over IP networks, how TDMoIP technology meets those challenges, and the standards shaping TDMoIP and related technologies.

Challenges of Transporting TDM

Conventional TDM networks are highly deterministic. A source device transmits one or more octets to a destination device via a dedicated-bandwidth channel every 125 μs. The circuit delay through a TDM network is predictably low and constant throughout the life of a connection. Timing is delivered along with the data, and the permitted variability (jitter and wander) of TDM clocks is tightly defined. In addition, the infrastructure supports a rich set of user features via a vast set of signaling protocols.

Packet-switched networks (PSNs), such as IP/multi-protocol label switching (MPLS) systems, are more efficient than TDM networks due to bandwidth sharing. However, this sharing leads to PSNs being inherently non-deterministic.

Packets entering and transiting the network must compete for bandwidth and switch/router ports, leading to packet delay variation (PDV) and lost packets. A source device may inject packets into the network at regular intervals, but the network offers no guarantee that these packets will arrive at the destination edge device spaced at the same intervals, in the same order, or even that they will arrive at all.

In addition, IP networks were designed for transport of arbitrary data. Thus, TDM-related signaling is not supported.

There are two main ways that designers are trying to integrate TDM services into IP-based networks. On one hand, designers can completely replace the TDM network and end-user equipment with a new infrastructure that provides innovative mechanisms for voice transport and signaling. The other approach leaves the end-user equipment and protocols intact, tunneling TDM data through the packet network.

In the end, this second approach could provide an easier and most cost-effective migration path for carriers and equipment vendors. With that in mind, let's dive into how TDMoIP works.

Diving into TDMoIP

TDMoIP emulates T1, E1, T3, E3, and N*64K links by adapting and encapsulating the TDM traffic at the network ingress. Adaptation denotes mechanisms that modify the payload to enable its proper restoration at the PSN egress. By using proper adaptation, the TDM signaling and timing can be recovered, and a certain amount of packet loss can be accommodated.

Encapsulation signifies placing the adapted payload into packets of the format required by the underlying PSN technology. TDMoIP encapsulations are presently defined for user datagram protocol (UDP)/IP, MPLS, and Layer 2 tunneling protocol (L2TP)/IP networks, and even pure Ethernet can be utilized with minimal adjustments. Let's take a closer look at adaptation and encapsulation.

How Adaptation Works

TDMoIP can utilize several different adaptation techniques, depending on the TDM traffic characteristics. Whenever possible, TDMoIP draws on proven adaptation mechanisms originally developed for ATM. A side benefit of this choice of payload types is simplified interworking with circuit emulation services carried over ATM networks.

For statically allocated, constant bit-rate (CBR) TDM links, TDMoIP employs ATM adaptation layer 1 (AAL1). This mechanism, defined in ITU-T standard I.363.1 and ATM Forum specification atm-vtoa-0078, was developed for carrying CBR services over ATM.

AAL1 operates by segmenting the continuous stream of TDM data into small 48-byte cells and inserting sequencing, timing, error recovery, and synchronization information into them. For example, if the original TDM stream consisted of a DS1 with channel associated signaling (CAS), the AAL1 adaptation inserts a pointer to the beginning of the next superframe. Thus, even if cells are lost, the pointer will enable recovery from the next superframe.

TDMoIP allows concatenation of any number of AAL1 cells into a packet (note that these are AAL1 cells and not ATM cells, i.e. they do not include the five-byte "cell tax"). By allowing multiple cells per packet, TDMoIP facilitates flexible tradeoffs of buffering delay (which decreases with fewer cells per packet) for bandwidth efficiency (which increases with more cells per packet, due to the per packet overhead).

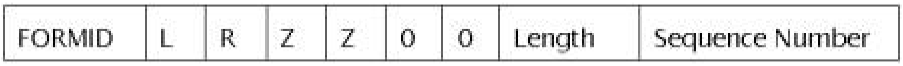
For dynamically allocated TDM links, whether the information rate varies due to activation of time slots or due to voice activity detection, TDMoIP employs ATM adaptation layer 2 (AAL2). This mechanism, defined in ITU-T standard I.366.2, was developed for carrying variable bit rate (VBR) services over ATM.

AAL2 operates by buffering each TDM time slot into short minicells, inserting the time slot identifier and length indication, sequencing, and then sending this minicell only if it carries valid information. TDMoIP concatenates the minicells from all active time slots into a single packet.

For time slots carrying high-level data link control (HDLC) data, such as data for common channel signaling (CCS), a special adaptation is provided that spots areas of non-idle data, which can then be directly encapsulated.

Encapsulating TDM Data

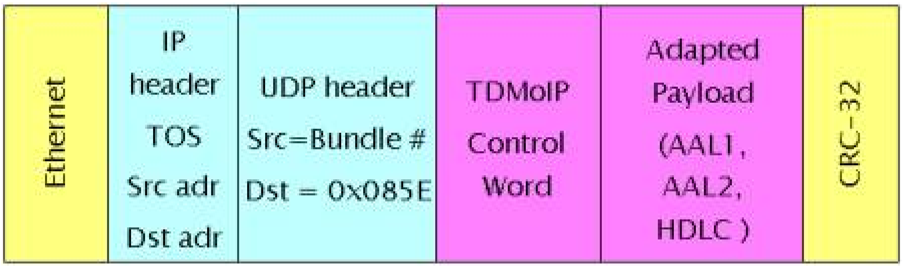
In TDMoIP packets, payload information is immediately preceded by a control word. This 32-bit control word, shown in **Figure 1**, contains the packet sequence number (needed to detect packet re-ordering and packet loss), the payload type, payload length, and alarm indications.



FORMID	L	R	Z	Z	0	0	Length	Sequence Number

Figure 1: TDMoIP's 32-bit control word.

For IP networks, UDP and IP headers precede the payload and control word. The UDP destination port takes the special value assigned to TDMoIP, while the source port is used to discriminate between different TDM bundles. The packet format is shown in **Figure 2**. Note: In this figure, we assume that Ethernet is used for layer 2.



Ethernet	IP header TOS Src adr Dst adr	UDP header Src=Bundle # Dst = 0x085E	TDMoIP Control Word	Adapted Payload (AAL1, AAL2, HDLC)	CRC-32

Figure 2: TDMoIP packet for IP networks.

For MPLS networks, an inner label precedes the payload and control word. This label acts as the TDM bundle demultiplexer, and the MPLS label stack. The packet format is shown in **Figure 3**, once again assuming Ethernet for layer 2.

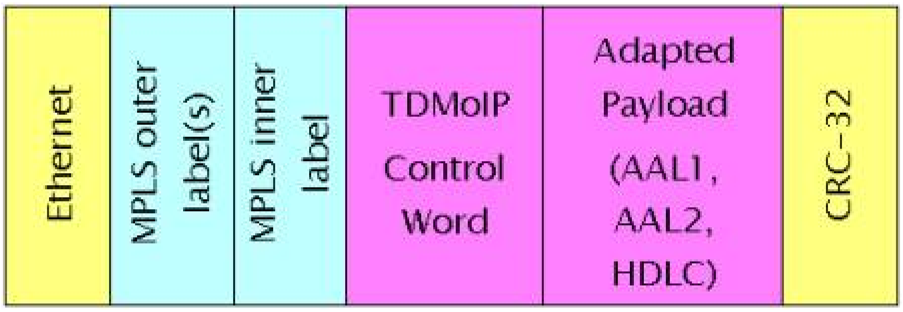


Figure 3: TDMoMPLS packet for MPLS networks.

Meeting the Real World
At first glance it shouldn't be difficult to carry voice or other TDM data over IP networks. Data is data, and data is what packet networks were designed to carry.

However, this simplistic view ignores several important issues; namely that PSNs do not have TDM signaling mechanisms, they may introduce much higher end-to-end delay than TDM networks, they don't carry the timing information needed by the far-end TDM equipment, and they occasionally lose packets. Let's look at each of these problems individually.

1. Signaling
In order to understand how TDMoIP handles TDM signaling, we must first differentiate between three types of signaling: in-band, CAS, and CCS.

In-band signaling, as its name implies, is transferred in the same audio band as speech. It can take the form of call progress tones such as dial tone and ring back, DTMF tones, frequency shift keying (FSK) for caller identification, and MFR1 in North America or MFCR2 in Europe. Since these are all audible tones, they are encoded in the TDM time slot and automatically forwarded by TDMoIP.

Speech compression algorithms, such as those used by VoIP systems, do not transmit these tones very accurately, requiring implementation of tone relay protocols to ensure that in-band signaling functions properly. Since TDMoIP delivers the original, unaltered voice samples, additional mechanisms are not required to handle in-band signaling.

CAS is carried in the same T1 or E1 frame as voice signals but not in the speech band. T1 robs bits for this purpose while E1 devotes an entire time slot to carrying four bits for each of the 30 remaining channels.

Since CAS bits are carried in the same T1 or E1 stream, they are readily handled by TDMoIP, even for fractional T1/E1 links. VoIP systems, on the other hand, would need to detect the CAS bits, interpret them according to the appropriate protocol, send them through the IP network using a messaging protocol, and finally regenerate and recombine them at the far end.

ISDN signaling and SS7 are examples of CCS and are often found occupying a TDM time slot. When these signals occupy a slot, they are forwarded by TDMoIP. If the signaling is not trunk-associated, then the network that is carrying it will continue to do so. Alternatively, a signaling gateway can be employed to encapsulate native signaling and the resulting packets forwarded as additional traffic through the PSN.

2. Delay

The PSTN places constraints on the tolerable end-to-end and round-trip delays. ITU-T G.114/G.131 states that one-way transmission times of up to 150 ms are universally acceptable, assuming adequate echo control is provided.

These constraints are not problematic for TDM networks, where the major component of the end-to-end delay is electrical propagation time. This is because a typical TDM network node (Sonet/SDH ADM, class switch, DACS, PBX, etc.) adds only 125 microseconds of latency to a trunk.

By contrast, the G.723.1 speech compression commonly used in VoIP systems adds a minimum 67.5-ms algorithmic delay, which often approaches 100 ms even before taking routing delays into account.

TDMoIP maps the TDM octets directly into the payload with no voice compression algorithms required and no resultant algorithmic delay. The buffering latency added by TDMoIP depends on the number of cells per packet but is typically in the single millisecond range. For example, a DiffServ-enabled metro router adds less than 10 ms of average latency to the TDMoIP packet. Thus, for a TDMoIP link with two hops, the total end-to-end delay is certainly no larger than 20 ms.

3. Timing

Conventional TDM networks rely on hierarchical distribution of timing. Somewhere in the network there is at least one extremely accurate primary reference clock with a long-term accuracy of one part in 10^{11}. This node, which offers Stratum 1 accuracy, provides the reference clock to secondary nodes with Stratum 2 accuracy. The secondary nodes then provide a time reference to Stratum 3 nodes. This hierarchy of time synchronization is essential for the proper functioning of the network as a whole.

As mentioned earlier, packets in the PSN reach their destination with delay that has a random component, known as PDV. When emulating TDM transport on such a network, this randomness may be overcome by placing the TDM packets into a "jitter buffer" from which data can be read out at a constant rate for delivery to TDM end-user equipment. The problem is that the TDM source time reference is no longer available, and the precise rate at which the data are to be "clocked out" of the jitter buffer is unknown.

In certain cases, such as "toll-bypass" links, the endpoints of the TDMoIP tunnel are full TDM networks, and timing may (indeed must) be derived from the respective network clocks. Since each of these clocks is highly accurate, they necessarily agree to high order.

For cases where at most one side of the TDMoIP tunnel has a highly accurate time standard, there are several ways to address this problem. Designers could provide independent time standards, such as atomic clocks or GPS receivers, to all TDMoIP devices, thus relieving the packet network of the need to send synchronization information. This approach, however, could be prohibitively expensive.

Another possibility is to supplement the PSN with a synchronous clock distribution network. But this approach requires deployment and maintenance of two separate networks.

For ATM networks, which define a physical layer that carries timing, the synchronous residual time stamp (SRTS) method is applicable. IP/MPLS networks, however, do not define the physical layer and thus cannot specify the accuracy of its clock.

Often the only alternative is to attempt to recover the clock based exclusively on the TDMoIP traffic. This is possible since the source TDM device is producing bits at a constant rate determined by its clock. Unfortunately, these bits are received in packets that suffer packet delay variation, a random process. The task of clock recovery is thus an "averaging" process that negates the effect of the random PDV and captures the average rate of transmission of the original bit stream. A phase-locked loop (PLL) is well suited for this task because it can lock onto the average bit rate, regenerating a clean clock signal that approximates the original bit rate.

One conventional means of clock recovery employs adapting a local clock that is based on the level of the receiver's jitter buffer. To understand the operation of this mechanism let's assume for a moment that there is no PDV but that the local clock is initially lower in frequency than the source clock. The jitter buffer fills with bits faster than it is emptied, and the fill-level starts to rise. This rise is detected and compensated by increasing the frequency of the local clock.

When PDV occurs along with a clock discrepancy, the jitter buffer level no longer rises or falls smoothly but fluctuates wildly about its average level. By using a PLL that locks onto the average bit rate, any frequency discrepancy between the source and destination clocks is eventually compensated. The receiver's jitter buffer will settle on the level corresponding to precise frequency alignment between the two clocks.

The PLL method has two main faults. First, the PLL must observe the sequence of level positions for a long period before it can lock onto the source clock, resulting in a lengthy convergence time. Second, the jitter buffer level may settle down far from its desired position at buffer center, thus making it vulnerable to overflow and underflow conditions. Alternatively, the jitter buffer size may be increased to lower the probability of underflow/overflow, but such a size increase inevitably adds to latency.

By using more sophisticated clock recovery algorithms, recovered TDM clocks can be made to comply with ITU-T G.823 and G.824 specifications for T1/E1 jitter and wander control while simultaneously delivering optimal latency.

4. Lost Packets

While proper application of traffic engineering and quality-of-service (QoS) is expected to minimize packet loss, packets will at times arrive at the egress out of order. They may also have been dropped altogether within the PSN.

The TDMoIP control word described above includes a 16-bit sequence number for detecting and handling lost and mis-ordered packets. In the case of lost packets, TDMoIP requires insertion of interpolation packets to maintain TDM timing. Misordered packets may be either reordered or dropped and interpolated.

While the insertion of arbitrary packets may be sufficient to maintain the TDM timing, in voice applications packet loss can cause gaps or errors that result in choppy, annoying, or even unintelligible speech.

The precise effect of packet loss on voice quality and the development of packet loss concealment algorithms have been the subject of detailed study in the VoIP community, but their results are not directly applicable to the TDMoIP case. This is because VoIP packets typically contain between 80 samples (10 ms) and 240 samples (30 ms) of the speech signal, while TDMoIP packets may contain only a small number of samples.

Since TDMoIP packets are so small, it is acceptable to simply insert a constant value in place of any lost speech samples. Assuming that the input signal is zero-mean (i.e. contains no DC component), minimal distortion is attained when this constant is set to zero.

Designers can employ a more sophisticated approach that calls for them to replace the missing sample with the previous one. This method is somewhat more justifiable in the VoIP case where the quasi-stationarity of the speech signal means that the missing buffer is expected to be similar to the previous. Even in the single sample case it is better than zero insertion due to the typical low-pass quality of speech signals and to the fact that during intervals with significant high frequency content (e.g. fricatives) the error is less noticeable.

A TDMoIP packet is usually declared lost following receipt of the next packet, hence both the samples prior to the missing ones and the samples following it are available. This enables even more sophisticated implementations to conceal the packet loss event by estimating the missing sample values. The simplest estimation is linear interpolation. More advanced methods of packet loss concealment (PLC) entail model-based prediction. Recent research has shown that five percent packet loss can be effectively concealed with perceptual voice quality still exceeding that of cellular phones.

Robustness and Performance Monitoring

If desired, connection-level redundancy can be achieved with TDMoIP by configuring duplicate transmit bundles with different IP destination addresses.

TDMoIP provides an OAM signaling mechanism for reporting on bundle status and collecting performance statistics. TDMoIP OAM messages include a connectivity check for setup/configuration and to prevent PSN flooding; one-way and round-trip delay measurements; and various other performance measurement messages for remote diagnostics.

Standardization Efforts

TDMoIP standardization is proceeding in several standards bodies and technical forums. In the Internet Engineering Task Force (IETF), work is being carried out under the auspice of the Pseudo-Wire Emulation Edge-to-Edge (PWE3) working group.

A pseudo-wire is defined as an emulation of a native service (such as ATM, frame relay, Sonet/SDH, or TDM) formed by tunneling that service's data through a packet network. A pseudo-wire is intended to emulate the behavior and characteristics of the original service as faithfully as possible, but that emulation need not be perfect. TDMoIP is known in the PWE3 working group as draft-anavi-tdmoip, a work in progress that has undergone several revisions.

In the MPLS Forum, TDMoIP has been divided into two proposed implementation agreements, "TDM Transport over MPLS using AAL1" and "I.366.2 Voice Trunking over MPLS". A similar division exists in the ITU-T, where Study Group 13 has developed a draft recommendation for voice services over MPLS, which includes the AAL2 mode. Work on an AAL1 mode in ITU-T is being carried out under the name “TDM-MPLS Interworking”.

References

1. ATM Forum: atm-vtoa-0078 "Circuit Emulation Service Interoperability Specification, Ver 2.0"
2. IETF: draft-anavi-tdmoip.txt "TDM over IP"
3. IETF: RFC 768 "User Datagram Protocol (UDP)"
4. IETF: RFC 791 "Internet Protocol (IP)"
5. IETF: RFC 2475 "An Architecture for Differentiated Services"
6. ITU-T: I.363.1 B-ISDN ATM Adaptation Layer (AAL) specification: Type 1
7. ITU-T: I.366.2 AAL Type 2 Service Specific Convergence Sublayer for Trunking
8. ITU-T: G.823 "The Control of Jitter and Wander Which Are Based on the 2048 kbit/s Hierarchy"
9. ITU-T: G.824 "The Control of Jitter and Wander Which Are Based on the 1544 kbit/s Hierarchy"
10. IEEE: 802.1p "Traffic Class Expediting and Dynamic Multicast Filtering"
11. IEEE: 802.1Q "Virtual LANs"
12. ATM Forum: atm-vtoa-0078 (CES 2.0) "Circuit Emulation Service Interoperability Specification Ver. 2.0 "
13. ATM Forum: atm-vmoa-0145 (LES) "Loop Emulation Service Using AAL2"

About the Authors

Yaakov (Jonathan) Stein *is a chief scientist at RAD Data Communications. He holds a BSc in mathematics and physics and a MSc and PhD in theoretical physics from the Hebrew University of Jerusalem. Yaakov is the author of Digital Signal Processing: A Computer Science Perspective and can be reached at yaakov_s@rad.com.*

Brian Stroehlein *is the product marketing manager for access/edge products at TranSwitch Corp.. He received a BS Electrical Engineering from Cornell University, Ithaca, NY and an MBA from the University of Connecticut. Brian can be reached at stroehlb@txc.com.*