Towards More Adaptive End-to-End Applications

Jörg Ott

Helsinki University of Technology (TKK)

1 Introduction and Motivation

The Internet Protocol inherently offers a best-effort datagram delivery service, allowing for arbitrary delay, reordering, loss, and duplication of packets. While reordering and duplication occur but are not (yet) commonplace, delays and losses are inherent properties used in the operation of many Internet protocols, as they serve as a measure for congestion. Transport and application protocols (should) monitor these values and are supposed to dynamically adapt their behavior to the changing network conditions.

At the transport layer, the notion of congestion control has been introduced to TCP [Jac88] (with countless variants developed since) to share network resources fairly (at the flow level), a concept incorporated in recent protocols such as SCTP and DCCP. Transport protocols also perform timeout adaptation (e.g., TCP's RTO) to cope with varying delays across different networks, which may span more than six orders of magnitude between a local Ethernet link and a loaded GPRS network. Path MTU discovery [MD90] is used to determine an appropriate packet sizes (e.g., TCP's segment size) in order to avoid fragmentation, although today often 1460 bytes are assumed to be safe.

All these mechanisms are hidden inside the transport layer and thus invisible to the applications. The underlying assumption is that applications using such transport protocols are sufficiently *elastic*, i.e., can deal well with changing transmission rates and delays. This abstraction was criticized in the past (e.g., [MBL$^+$04]), as TCP would retransmit potentially outdated information and an application is not able to determine the detailed status of the send buffer. The channel concept of SCTP and its unreliable mode, developments such as *Structured Streams* [For07], and more direct resource control [MBL$^+$04] would allow for some more flexibility.

Inelastic applications, in contrast, have typically avoided TCP and realized the necessary protocol mechanisms on top of UDP, leaving full control to the application. This applies performing semantic fragmentation of application data units and their mapping onto packets [CT90] as is common for real-time media transmissions using RTP [HP99]. Real-time applications are also supposed to perform rate adaptation (by switching codes, adjusting packet sizes) and could dynamically adapt their behavior to the observed network conditions based upon RTCP reports.

There are two related assumptions implied: 1) The applications are capable of adapting across a wide range of transmission characteristics. In practice, however, the only truly adaptive applications appear to be file transfer, be it as a simple client-server system or a more sophisticated peer-to-peer sharing application, and others ones operating in the background without being time critical. Instead, most applications have an operational range of network characteristics acceptable to the user, leading to: 2) The best effort service achieved will be sufficient for the application needs. If it is not, users will notice and react and typically stop using the applications. This applies to (semi-)interactive TCP-based applications (such as web access, HTTP streaming, and SSH) and even more so to UDP-based real-time applications such as VoIP.

In this paper, we broaden the scope of application adaptivity in two ways: a) we capture a broader range across which to adapt, specifically extending to high delays and disconnections and b) we include more of the application semantics into the adaptation itself. Our goal is to conceptually capture those cases in which the above assumption 2) no longer holds (section 2) and devise exemplary mechanisms to fulfill assumption 1) nevertheless (section 3). We conclude with a brief discussion in section 4.

2 The (Mobile) Internet Today: When Best Effort Is Not Enough

With wireless and mobile Internet becoming increasingly dominant, the connectivity characteristics for mobile users deserve explicit consideration in protocol design [Ott06]. It is well known that wireless and mobile connectivity characteristics may differ significantly from what is observed in the fixed Internet. Specifically, wireless links are susceptible to attenuation, interference, etc. and varying load due to shared channels and may yield highly variable delays due to link layer retransmissions and queuing. Worse, these phenomena and also coverage gaps may lead to temporary loss of connectivity: from fractions of a second to minutes or hours.

The result is that loss, delay, and throughput of wireless links may vary more heavily and more abruptly than in fixed networks, making adaptation more demanding. To cope with short disconnections (specifically during handovers), network operators often perform significant queuing for their mobile data networks; however, this leads to delay being accumulated, negatively impacting TCP and interactive application protocols. On the other hand, if no or little queuing takes place, more losses occur. Longer disconnections will lead to packet losses in either case. Overall, the outcome from an application perspective is delay [Ott08].

In particular, highly interactive applications such as VoIP suffer from losses and delays. Since, a range of codecs exist nowadays (some of them support variable data rates), audio communication can be adapted between some 4 kbit/s (Speex, AMR-NB) and 64 kbit/s (G.711), providing a flexible operating range to cope with congestion. In contrast, packet losses lead to unintelligible speech, and variable delays and jitter result in either losses or large delays due to playout buffering. As it was found that, for *skype* users, that delay has a less significant impact on perceived speech quality than losses [CHHL06], some flexibility for packet repair exist and short-term outages might be concealed. However, as losses, delays or the frequency of disconnections grow, interactive voice becomes essentially unusable.

This example shows that wireless and mobile communication may yield a best effort service that is insufficient for specific applications if the achievable service quality is (temporarily) outside their supported the operational range. In the above example, the application's limitations to adapt to a broader range of operational conditions stems from the interactivity requirements, i.e., the specific application semantics.[1] Rather than trying to make the network match the application needs (e.g., using QoS), we suggest to revisit the application semantics and investigate whether those can be broadened to allow expanding the applications' operational range.

3 Case Study: Adaptive Voice Communication

Users of mobile phones are well aware of varying network conditions and the resulting effects ranging from short-term outages to call disconnections. And users of VoIP over wireless networks (such as WLANs) often experience (short) periods of unintelligible speech due to losses. In either case, the voice applications are not capable of adapting to the network conditions and error handling is up to the user: from "manual synchronization" by repeating sentences to re-dialing a lost call [OX07]. We have investigated switching between plain SIP voice calls and voice messaging for IP networks, integrating automatic redial functionality [OX07] and we have developed an asynchronous voice messaging applications running on top of DTNs [Isl08]. We can generalize these ideas further and synthesize both approaches if we assume a more flexible communication substrate allowing for the exchange of arbitrarily large packets (or: messages)[2], thus creating a continuum of voice-based interactions as shown in figure 1.

[1] For other applications, this holds even if the underlying transport (and session) protocols are capable of tolerating a broader operational range (see, e.g., the countless extensions on TCP performance enhancements and disconnection tolerance as discussed in [OSC+09]).

[2] This functionality may be realized at the network layer, as in some DTNs or in a future Internet [Ott08], or at the transport or application layer on top of IP.

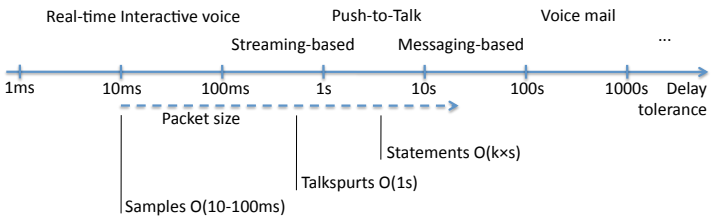

Fig. 1. Adaptivity of voice applications

As noted above, a key metric for communications is delay, since packet loss may be reduced when allowing for more delay, disconnections can be overcome by waiting sufficiently long, and the data rate of audio codecs appears sufficiently adaptive for most scenarios; if less instant capacity is available, reverting to non-real-time transmission will help, at the expense of increased delay. The mode of operation with the lowest delay are interactive real-time voice conversations, in which we assume minimal mouth-to-delay (typically $<$ 200ms) for acceptable interactivity. Longer delays are tolerable if we move towards less synchronous interactions, as known from one-way alternate communications via walkie-talkies or *Push-to-Talk*; yet some degree of interactivity is preserved. Depending on the network conditions, information exchange may be based upon real-time packet streaming (as in Push-to-talk over Cellular) or reliable exchange of messages (as in our DT-Talkie [Isl08]). Finally, voice messaging is a rather asynchronous style of interaction, more comparable to email.

If we exploit all these different modes of operation and move smoothly between them, we can make VoIP applications more elastic and expand their operational range. Assuming RTCP to monitor RTT, jitter, loss rate, and (implicitly) connectivity, two endpoints can adapt their bit rate by choosing different audio codecs and their packet rate (and header overhead) by varying packet sizes. When network conditions worsen, voice application data units (ADUs) can be increased further. With increasing ADU sizes, however, delay increases and communication loses interactivity. While using regular sampling intervals (e.g., 10–100ms) for smaller packets, an application may decide to identify talk-spurts and send (groups of) talk-spurts in ADUs to keep semantically related information together, thus ensuring the atomic (non-)delivery of related pieces and continued playback of each talkspurt. This can be expanded further to gather complete statements of a user (as with walkie-talkies), e.g., by means of local processing using heuristics, leading further towards asynchronous communication. If connectivity to a peer is lost entirely (for some time), one or more local statements may be aggregated and turned into voice mail. Continuous monitoring of networking conditions should also indicate when the situation is improving, so that the application can move again towards synchronous operation.

With increasing ADU size, the impact of a single lost ADU grows: error concealment will work less well for 100ms of missing speech than for only 20ms and if entire statements get lost, the peer may wonder why nobody is responding. Hence, larger ADUs suggest using more reliable transport mechanisms—which are “affordable” since the acceptable delay also increases, thus e.g. allowing for retransmissions.

Finally, A suitable user interface is required to offer cues to the user about the present mode of operation; it should also allow controlling which modes are acceptable for a given conversation.

4 Discussion

The above example has provided some intuition on how more comprehensive considerations could help adaptivity of one specific application. Similar considerations can be applied to other real-time and non-real-time applications: For example, media streaming applications perform

pre-buffering to deal with varying networking conditions, a concept that has been explored further to deal with temporary disconnections. And web applications could toggle more smoothly between online and offline operation [OK06], provided that the application protocols are adapted accordingly [Ott06].

One interesting follow-up question is if commonalities can be identified across different applications for common support in future transport protocols. Another one warranting further discussion is what the implications on the present (or a future) networking infrastructure are. We have argued that a future Internet should become inherently more delay-tolerant [Ott08] but, as we discussed above, there is a feature interaction between queuing/buffering ADUs inside the network and the end-to-end control loops of the applications. This may call for limited additional interaction between endpoints and network elements—e.g., providing hints for ADU processing—while maintaining the endpoints independent of the network elements.

References

[CHHL06] Kuan-Ta Chen, Chun-Ying Huang, Polly Huang, and Chin-Laung Lei. Quantifying skype user satisfaction. In *Proceedings of the ACM SIGCOMM*, 2006.

[CT90] David D. Clark and David L. Tennenhouse. Architectural Considerations for a new Generation of Protocols. In *Proceedings of ACM SIGCOMM Symposium on Communication Architectures and Protocols*, September 1990.

[For07] Bryan Ford. Structured Streams: a New Transport Abstraction. Proceedings of ACM SIGCOMM, August 2007.

[HP99] Mark Handley and Colin Perkins. *Guidelines for Writers of RTP Payload Format Specifications*, December 1999. RFC 2736.

[Isl08] Md. Tarikul Islam. DT-Talkie: Push-to-talk in Challenged Networks. In *Demo at ACM MobiCom*, 2008.

[Jac88] Van Jacobson. Congestion Avoidance and Control. In *Proceedings of ACM SIGCOMM*, August 1988.

[MBL+04] Jeffrey Mogul, Lawrence Brakmo, David E. Lowell, Dinesh Subhraveti, and Justin Moore. Unveiling the Transport. *ACM SIGCOMM Computer Communications Review*, 34(1):99–105, January 2004.

[MD90] Jeffrey Mogul and Steve Deering. *Path MTU discovery*, November 1990. RFC 1191.

[OK06] Jörg Ott and Dirk Kutscher. Bundling the Web: HTTP over DTN. In *Proceedings of WNEPT 2006*, August 2006.

[OSC+09] Jörg Ott, Nils Seifert, Caleb Carroll, Nigel Wallbridge, Olaf Bergmann, and Dirk Kutscher. The CHIANTI Architecture for Robust Mobile Internet Access. In *Proceedings of the 10th IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM), Industry Track*, June 2009.

[Ott06] Jörg Ott. Application Protocol Design Considerations for a Mobile Internet. In *Proceedings of the 1st ACM MobiArch Workshop*, December 2006.

[Ott08] Jörg Ott. Delay Tolerance and the Future Internet. In *11th International Symposium on Wireless Personal Multimedia Communications*, 2008.

[OX07] Jörg Ott and Lu Xiaojun. Disconnection Tolerance for SIP-based Real-time Media Sessions. Proceedings of the International Conference on Mobile Ubiquitous Multimedia (MUM), 12 2007.