REAL-TIME INTERACTION OVER THE INTERNET:
MODEL FOR QOS ADAPTATION

Christophe Salzmann, Denis Gillet and Philippe Mullhaupt

Swiss Federal Institute of Technology (EPFL) – Switzerland
first.lastname@epfl.ch

Abstract: Real-time Interaction over the Internet (RTI^2) is an Internet service that is required typically by remote experimentation applications. From a quality of service (QoS) point of view, RTI^2 has constraints that differ from usual real-time multimedia services such as video streaming or video conferencing. The RTI^2 QoS can be expressed by three values that represent the level of interaction, the dynamics rendering and the semantic content. The RTI^2 metrics, derived from these values, are essential to successfully implement an end-to-end (E2E) control scheme that adapts the transmission parameters to the equivalent E2E infrastructure that encompass not only to the network state, but also to the server and client applications processing capabilities. A model of the end-to-end system is exploited to ensure a minimal transmission time for a given E2E system. *Copyright © 2005 IFAC*

Keywords: Metrics, Real-Time, Internet, Interaction, Adaptation, Control, Quality of Service, End-to-End Approach

1. INTRODUCTION

1.1 Context and aims

The key issue in implementing real-time interaction over the Internet with a physical system is to enable the control and the perception of the dynamics of the physical system at distance. In other words, the information acquired at the server side has to be rendered at the same pace it has been obtained with a minimal time delay between the acquisition phase and the rendering phase. Physical systems under study are typically mechatronic systems with moving parts. These systems are challenging because their intrinsic time constant are generally in the same order of magnitude as the Internet transmission time. Moreover, the challenge also comes from the wealth of information that needs to be transmitted to represent system state and its conditions of operation.

Real-time interaction over the Internet (RTI^2) has been extensively used for remote experimentation of systems such as robots, inverted pendulums, electrical drives, etc. (Fig. 1). These systems are typically used in hands-on engineering education where students can access them either locally or remotely using Web-based solutions (Gillet, *et al.*, 2001; Schilling and Perez Vernet, 2002).

An efficient interaction provides the distant user with the best possible feedback such that the drawbacks inherent of the distance are minimized and with sufficient information to reproduce the state of the distant equipment and its operational conditions.
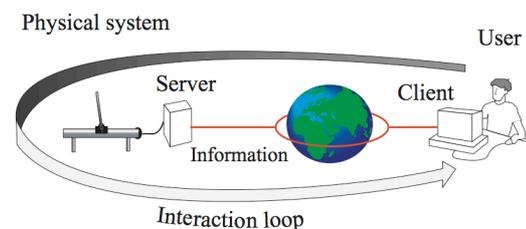

Fig. 1. Typical remote experimentation setting

The interaction loop consists of the user action transmitted to the server via the Internet and the feedback provided by the server to this action

(Fig. 1). A dedicated solution is required to satisfy the constraint that the operators gets feedback as quickly as possible on carried out actions.

There are three key aspects that need to be satisfied in order to provide a suitable quality of service, namely the level of interaction which represents how quickly a feedback is provided to the user; the perception of the dynamics which represents how accurately in time the behavior of the remote system is perceived and the amount of semantic content that represents how much of the distant system state and conditions of operation can be perceived by the client.

The applications, protocols and infrastructures independence is also required to enable an easy deployment and to avoid limiting the use of the solution to a specific group of users. This independence also guarantees that the proposed solution will adapt to future versions of its components. The various components of the solution should also behave adequately to use the available bandwidth in a fairly manner, this being compatible with the Internet best practices (Floyd and Fall, 1999).

1.2 Current implementation solutions

Various solutions can be explored to efficiently implement real-time interaction over the Internet. While the video streaming solution looks suitable for RTI^2, the use of buffers to smooth the Internet bandwidth variation and to display images at a constant rate to the user makes it inappropriate since the buffering process adds delays to the transmission (Feng and Rexford, 1997). Delays may also be the result of the temporal frame encoding, such as in MPEG, where frames differences are encoded and therefore at least two frames have to be waited. Many improvements are proposed to encode video frames, from pixels to blocks and then objects encoding such as in MPEG-4. Video streaming also implements various techniques to improve the recovery of lost information, this without resending the lost information. For example, Forward Error Corrections (FEC) encoding sent redundant information with the current information to help the information recovery (Nguyen and Zakhor, 2002). The reduced retransmission delay is made at the expense of increased bit rate and processing time.

The above video streaming solutions aim at improving the audio and video quality, which mainly defines the QoS for video streaming. This objective is different than the RTI^2 QoS where audio and video qualities are important but are not the main objective of controlling the information transmission time. Improved encoding and FEC techniques can be used

by RTI^2 if they do not add improper delays to the transmission.

A straightforward solution to implement RTI^2 is to use a communication channel that can guarantee a given quality of service (Aktan, *et al.*, 1996), such as a given bandwidth and latency, via reservation or by other means. This can be done by placing additional intelligence in the network at the router level. While this solution might be a promising one, it not only requires a widely accepted agreement among manufacturers and providers regarding new communication protocols, but also asks for expensive software upgrades for most of the already deployed infrastructure.

Instead of trying to modify the routers behaviors, the proposed solution is based on an end-to-end scheme that can be implemented at the application level. Proposed Internet improvements such as differentiate services, bandwidth reservation, packets coloring, etc. are not generally available and therefore not considered *a-priori,* but the proposed approach implicitly take advantage of them when available.

1.3 Paper organization

Section 2. The component and block abstractions that are used to implement the end-to-end approach. Section 3. Metrics definition summary. Section 4. Bandwidths and throughput definitions. Section 5. A simple model for the E2E transmission is proposed. Section 6. An control scheme is proposed. Section 7. Example. Section 8. Conclusions.

2. END-TO-END APPROACH AND ABSTRACTIONS

Traditionally, the end-to-end (E2E) approach only considers the end-to-end network path from the server to the client. The proposed approach not only takes into account the transmission path over the Internet but also the client and server applications located at both ends, thus considering the information from its capture to its rendering. The adjunction of both the server and the client applications in the transmission path is required to guaranty that the metrics measurements correctly reflect the whole E2E path ability to handle the flow of transmitted information.

The client application is meant to run on various, unknown before hand, computer devices from personal digital assistants (PDA) to desktop computers that are connected to the Internet with the help of a wire or wireless connection. Since the network does not necessarily act as the bottleneck anymore, the characteristics of the client application

have to be taken into consideration to prevent the server application wasting network resource by sending more information that the client can process.

A macroscopic view of the E2E transmission path abstracts the client-server application and the network transmission into components. This abstraction frees the solution from the underlying constraints, such as the transmission protocols, the software, the hardware and the network configuration.

2.1 Components abstraction

The flow of information undergoes various transformations from the server application to the client application (Fig. 2) that can be abstracted into three components, namely the acquisition encoding component, the transmission component and the decoding–rendering component (Fig. 3).
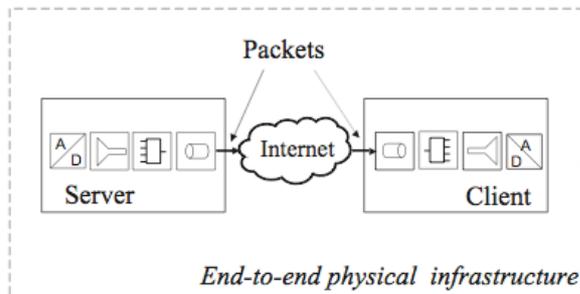

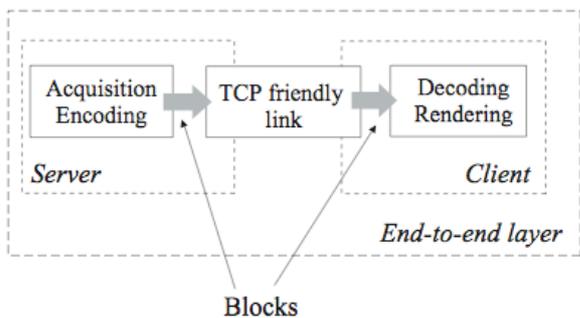Fig. 2. Physical end-to-end infrastructure.

Fig. 3. End-to-end components abstraction.

Traditional the **server** application consists of the acquisition–encoding component and the network access part of the transmission component. Similarly the **client** application includes the decoding–rendering component and the network access part of the transmission component. The proposed abstraction considers the network transmission through out the network interface seen by the application, by opposition to the network interface seen by the low level layers of the Operating System (OS).

The **acquisition-encoding component** transforms the system state and its conditions of operation to its digital representation. This is done via data acquisition interfaces. One of theses devices can be a camera that would produce an image. The image can then be compressed to reduce the size of the transmitted data. Another source of information are the measurements made via a data acquisition (DAQ) board. The various streams of information are aggregated to form the basic information unit called block, defined thereafter.

The **decoding-rendering component** is similar to the acquisition and encoding component. The information simply goes the reverse path, first the block information streams are extracted, then the information is decompressed and finally rendered.

The **transmission component** encompasses the sever network interface and the client network interface. The transmitted information is handled by this component as soon as the control of the transmitted data is transferred from the server application to the underlying OS. The transmitted information leaves this component when the client application has access to this information. There is no handle to control the transmission over the Internet once the data leaves the computer and until it is received at the other end. This is due to the non-deterministic aspect of the best effort Internet network and to the nature of the protocol used to transmit the data. The routers along the transmission path simply do their utmost to deliver the data to the receiver as fast as possible despite the variation of the network load. In other words, neither the network bandwidth nor the network latency can be guaranteed.

A **block** is defined as the aggregated information that represents the short-term state history and the operating conditions of the distant system at a given time. It is like a snapshot that captures all the information regarding the distant system at a given time. A block is the basic unit of information for the semantic content. For example, a block could be made of a video image combined with the measurements acquired concurrently. Blocks of various sizes and various durations are generated by the acquisition-encoding component.

The block duration $D(k)$ represents the elapsed time between the current block creation and the next block creation. The block duration is the inverse of the current block pace.

Block is not to be confused with packets that are smaller data units transmitted over the network and handled by the communication protocol.

3.   METRICS FOR QOS IN RTI[2]

Three key aspects define the quality of service for the RTI[2] are the level of interaction, the perceived dynamics and the semantic content.

The level of **interaction** is characterized by the delay observed between a user-performed action and the perception of its effect by the user. The delay represents the information round trip time measured at the application level, not at the network level. This delay is function of many factors, especially buffers that are found along the transmission path. For a valuable user experience, the delay should be as small as possible, and should also be in accordance with the dynamics of the distant system. Systems with slower dynamics show less stringent constraints. If this delay cannot be kept to a minimal value, the interaction is deteriorated. Special care in the solution design may partially compensate for a slightly excessive delay.

The **dynamics** of the distant system need to be perceived at the client side. If the pace at which the information is acquired by the server and delivered to the client application is not adequate, the user might get a biased or wrong perception of the actual behavior of the distant system.

In multimedia applications a buffer is traditionally used at the client side to smooth the information playback. Such buffer adds delay to the transmission and therefore cannot be considered if it depreciates the user perception.

The **semantic content** has to be rich enough to enable the perception of the state and the conditions of operation at the client side. There are various options to provide this information. Video image, Virtual-Reality representation or data history can be used for that purpose. For a given type of representation, the more qualitative information is sent, the better the state can be perceived. For instance, a good quality picture, bigger in size, is more informative than a low quality image, smaller in size.

The above three key aspects define the quality of service for RTI[2]. Since there are no direct RTI[2] QoS sensors, metrics need to be defined using available information. The available information that can be measured or estimated at the client side are the achieved block size and the achieved block pace and, at the server side, the achieved E2E round trip time. The ratios between the measured block values and the initial block characteristics define the metrics: τ for the block pace ratio, ε for the block size ratio and d for the E2E round trip time.

A comprehensive definition for the presented metrics has been presented in (Salzmann *et al*. 2004). and are summarized thereafter.

*The **block round trip time d** represents the time taken by a block to be successfully acquired, transmitted and rendered at the client side and then the time for the block acknowledgement to be sent back to the server.*

*The **block duration ratio** τ is the ratio between the duration of the block perception measured at the client side and the block duration at the server side.*

*The **block size ratio** ε is the ratio between the successfully rendered block size and the original block size.*

4.   E2E BANDWITDH, THROUGHPUT AND BOTTLENECK

The figure 5 depicts the various terms used to characterize the bandwidths. The bandwidth is defined as a data over time ratio, and can be viewed as the capacity of the E2E infrastructure.
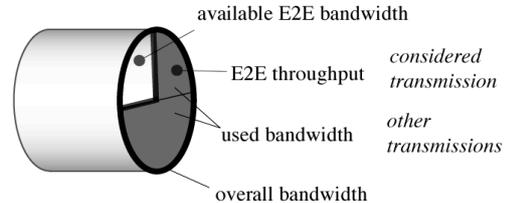

Fig. 5. E2E bandwidths and throughput.

It is important to notice that the network bandwidth that considers only the transmission link is different than the E2E bandwidth that encompasses the overall infrastructure including the transmission component, the client component and the server component. The available E2E bandwidth is the minimum of the bandwidth of the various components. The component with the smaller bandwidth is also called the bottleneck.

The ratio of the block size $S(k)$ over the block duration $D(k)$ defines the E2E throughput when the block k is transmitted. The throughput represents the part of the available E2E bandwidth used for RTI[2] transmission. The available E2E bandwidth can be estimated based on block size $S(k)$ and the block round-tip-time $d(k)$.

There is no global measurement for the overall transmission bandwidth that includes all the users sharing a transmission links. Some segments of the Internet provide guarantied bandwidth or bandwidth

measurements that informs about the available bandwidth. Since this information cannot be made ubiquitously available, it is assumed that only the E2E throughput and the E2E bandwidth $B_e(k)$ can be observed.

5. A SIMPLE MODEL FOR RTI[2]

The end-to-end layer (fig. 3) can be represented simply as the combination of an equivalent propagation delay, an equivalent bandwidth and an equivalent E2E buffer. While the propagation delay cannot be avoided, the E2E buffer should be kept empty to avoid undesirable delays.

The equivalent buffer $R(k)$ represents the combination of the various buffers found along the transmission path. It includes buffers found in the various network devices, buffers due to transmission protocol and buffers at both the server and the client side.

The equivalent propagation delay $C(k)$ represents the time for the block k to actually transit from one end to the other. It is the sum of all the propagation delays found in the E2E transmission. Every block, whatever its size, has to wait for a duration of $C(k)$.

The end-to-end block round-trip time $d(k)$ is measured at the application level. Measurements show that the $d(k)$ can be expressed with good approximation as the following:

$$d(k) = C(k) + \frac{S(k) + R(k)}{B_e(k)} \qquad (1)$$

Where $C(k)$ and $B_e(k)$ are parameters varying slowly. The dynamic of the E2E buffer is given by

$$R(k+1) = R(k) + S(k) - B_e(k) D(k) \qquad (2)$$

Where
- C: propagation delay, in [s]
- S: block size, in [Bytes]
- D: block duration, in [s]
- R: size of the equivalent buffer, in [Bytes]
- B_e: E2E bandwidth, in [Bytes/s]

The aim of RTI[2] is to provide the user with a feedback delay as small as possible while sending a maximum of information. From (1) we can see that the minimal value for the block round-trip-time d is obtained when the equivalent buffer is empty (ie. R=0). This gives for typical operating conditions:

$$d(k) = C(k) + \frac{S(k)}{B_e(k)} \qquad (3)$$

We consider the case when p is chosen a-priori according to the remotely controlled system

dynamics. $S(k)$ is the control signal adapted according to the measured $d(k)$.

The parameters C and B_e can be estimated using least square techniques. The estimated E2E bandwidth B_e gives the maximum block size S_{max} that can be chosen for a given D.

$$S_{max}(k) = B_e(k)D \qquad (4)$$

In other words, the maximum block size S_{max} for a given D ensure that the equation (3) holds and that the equivalent E2E buffer stays empty.

6. END-TO-END ADAPTATION

The objective of the E2E adaptation (Fig. 6) is to compensate for the fast E2E layer characteristics variations to ensure that either the equivalent E2E buffer does not fill-up or to ensure that the available bandwidth is fully used.
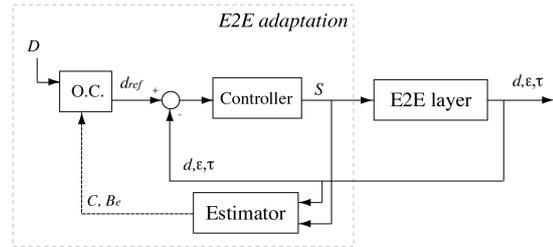

Fig. 6. E2E adaptation scheme.

The measured block round trip time d is compared to the block round-trip time reference d_r computed by the Operation Condition block (O.C.). The block round-trip time d_r is computed using (3) with the block size $S(k) = S_{max}$ derived from the estimated B_e and the defined block duration D.

$$d_r(k) = \alpha \left[\overline{C} + \frac{S_{max}(k)}{\overline{B}_e} \right] \qquad (5)$$

Where α is chosen slightly smaller than 1 to maintain the adhoc operating conditions (O.C.)

Since there is no dynamics in the E2E structure when the equivalent buffer is empty, any controller could be chosen to track d. An educated guess would be to choose an integral controller. The integral controller has the advantage of approaching smoothly the chosen reference value according to the integral gain ki. Another benefit is the low-pass filtering effect that smoothes the potential noise added the block round-trip time measurement. Anti-reset windup ARW mechanism should be added to limit the internal dynamics of the controller.

7. EXAMPLE

The figure 7 shows an example of the evolution of the E2E transmission delay d for a TCP connection between the client, a hand-held computer (PDA), and a desktop computer (server). The information transits through the Internet using 3 routers. The final link, a wireless Bluetooth connection, is chosen such that it can easily be perturbed. The TCP protocol is used to guaranty a friendly transmission and to insure that any block is fully transmitted. The constant sending pace is D=250 ms, the identified C=130ms, this gives a d_r of 380ms.

Figure 7a shows the block round-trip time $d(k)$ variations resulting from putting the PDA at a distance from the Bluetooth access point. Before time t_1, the PDA is close to the access point. Then, at time t_1, the PDA is rapidly (~2 seconds) placed at a distance of about 5 meters from the access point and it stays there until time t_2. At time t_2, the PDA is brought back to its original position.
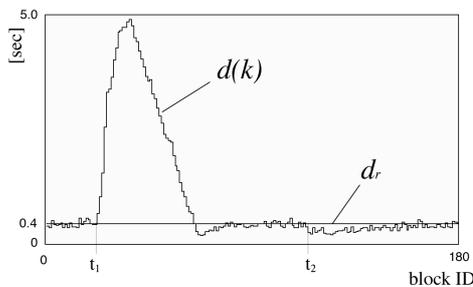

Fig. 7a. The measured d(k) subject to perturbations. At time t_1, the PDA is rapidly placed at a distance; at time t_2 it is brought back to its original position.
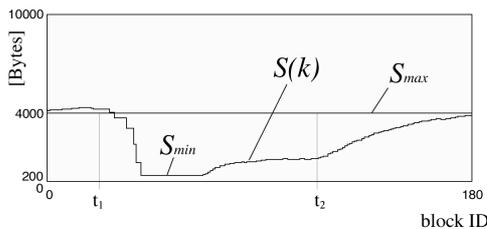

Fig. 7b. The block size S(k) evolution. After t_1, the controller rapidly reduces S(k) to its minimal value, then it increases S(k) to an intermediate value until t_2 when the PDA is brought back to its original position.

The perturbation applied to the E2E infrastructure is considerable. As a result, the block round-trip time increases to about 5 seconds. The controller quickly reduces the block size $S(k)$ in order not to worsen the situation (Fig 7b). Since the controller cannot send blocks of a negative size so as to empty the E2E buffer, it sends blocks of a minimal size until $d(k)$ crosses dr. At that time, the controller can increase

the block sizes to a larger value, but smaller than $Smax$, resulting from the new PDA position.

The controller successfully tracks the reference value dr even if the E2E characteristics are rapidly and significantly modified. After the transient phase the controller is able to reject the remaining constant perturbation. In other words, the controller successfully guarantees the QoS, i.e. the interaction, when a perturbation is applied to the system.

8. CONCLUSIONS

In this paper, the quality of service for real-time interaction over the Internet and the associated metrics are defined. Then a model for the E2E infrastructure is presented. This model is the combination of an equivalent propagation delay, an equivalent bandwidth and an equivalent buffer. In order to provide a minimal transmission time the equivalent buffer should remain empty. Using the proposed model, the E2E bandwidth is estimated and the maximum block size can be controlled. The proposed control strategy has been successfully implemented to adapt to the varying E2E transmission characteristics. It also rejects the perturbations induced by an E2E characteristics variation and adapt to the new operating conditions.

REFERENCES

Gillet D., C. Salzmann and P. Huguenin (2001). Distributed Architecture for Teleoperation over the Internet. *Lecture Notes in Control and Information Sciences 258: Nonlinear Control in the year 2000,* Springer-Verlag, 399-407, London.

Schilling K., M. Perez Vernet (2002). Remotely Controlled Experiments with Mobile Robots, *Proceedings IEEE Southeastern Symposium on System Theory*, Huntsville.

Floyd S. and K. R. Fall (1999). Promoting the use of end-to-end congestion control in the Internet. *EEE/ACM Transactions on Networking*, 7, 4, pp. 458-472.

Feng W. and J. Rexford (1997). A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video, *Proceedings of the IEEE INFOCOM conf.*, pp. 58-66.

Aktan B., C.A. Bohus, L.A. Crowl and M.H. Shor (1996). Distant learning Applied to Control Engineering, *IEEE Transaction on Education,* 39, pp. 320-32.

Salzmann C., D. Gillet, and P. Müllhaupt (2004). Metrics for QoS in Real-Time Interaction over the Internet. *Porceedings of the CCCT'04 Conference,* Austin, TX, USA.