AN ADAPTATION SCHEME FOR REAL-TIME INTERACTION OVER THE INTERNET

Christophe Salzmann, Denis Gillet and Philippe Müllhaupt

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

Abstract: Real-time Interaction over the Internet (RTI2) is an Internet service that is required typically by remote experimentation applications. From a quality of service (QoS) point of view, RTI2 has constraints that differ from usual real-time multimedia services such as video streaming or video conferencing. The RTI2 QoS can be expressed by three values that represent the level of interaction, the dynamics rendering and the semantic content. The RTI2 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 structure that encompass not only to the network state, but also to the server and client applications processing capabilities. A macroscopic view of the system is exploited to ensure applications, protocols and infrastructures independence.

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 (RTI2) 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.

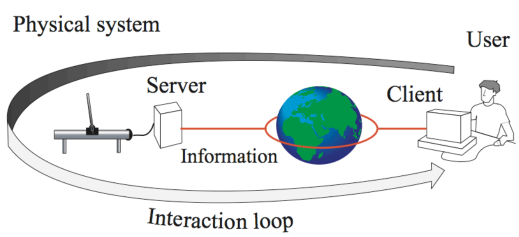

Fig. 1. Typical remote experimentation setting

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 guarantee 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 RTI2, 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 add delays to the transmission (Feng and Rexford, 1997). Video conferencing is another real-time application that seems comparable to RTI2 but it carries differences: the priorities for the video and the audio are inverted. Sound is preponderant over image for video conferencing while this is not the case for RTI2. There might even be no sound at all. In video conferencing the amount of data transferred between the two parties is generally symmetrical, this is not the case for RTI2 where only a small amount of data goes from the client to the server but a large amount of data flows from the server to the client. Another difference is the scalability of the used bandwidth. RTI2 bandwidth usage ranges from a few bytes per second to Kilobytes per second. The former bandwidth corresponds to a client application running on a PDA with a Bluetooth network access and the later correspond to a client running on a desktop computer with a LAN access. The lowest values cannot be considered for video conferencing due to sound quality constraints.

A straightforward solution to implement RTI2 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 Main results

The three key aspects of RTI2 – the level of interaction, the perceived dynamics and the information semantic content – that define an efficient user experience can be translated in three controllable quantities – the information transmission delay, the information delivery pace and the information size. The basic unit of information that transits from the server to the client application is defined as a block. The two first metrics used to evaluate the achieved QoS are the ratio between the sent block pace and the perceived block pace, and the ratio between the sent block size and the perceived block size. The third metrics being the block E2E round trip time measured at the application level. Using these metrics, an adaptation strategy has been successfully implemented to reject the perturbations induced by the bandwidth variations in the case of a Bluetooth transmission, thus guarantying an efficient user interaction.

1.4 Paper organization

Section 2. The component and block abstractions that are used to implement the end-to-end approach. Section 3. The metrics definition. Section 4. An adaptation scheme that ensure a efficient interaction using the proposed metrics. Section 5. 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 are 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).

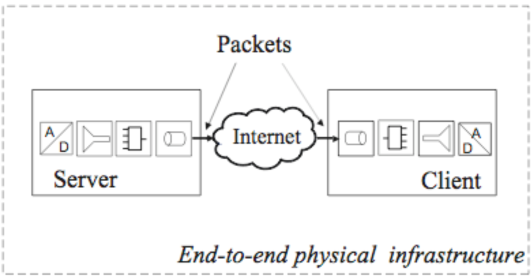

Fig. 2. Physical end-to-end infrastructure.

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).

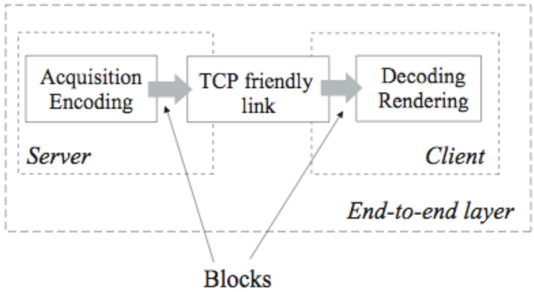

Fig. 3. End-to-end components abstraction.

The **acquisition-encoding component** trans-forms 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.

Def. A ***block*** *is defined as the aggregated information that represents the state and the operating conditions of the distant system at a given time.* It is like a snap-shot 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 (Fig. 4). Blocks of various sizes and various periods are generated by the acquisition-encoding component.

Blocks need to carry additional information to permit their processing and playbacks in the decoding-rendering component, this even if blocks were partially or completely lost during the transmission. This additional information, stored in the block header, includes the block identifier, the block timestamp and the block period.

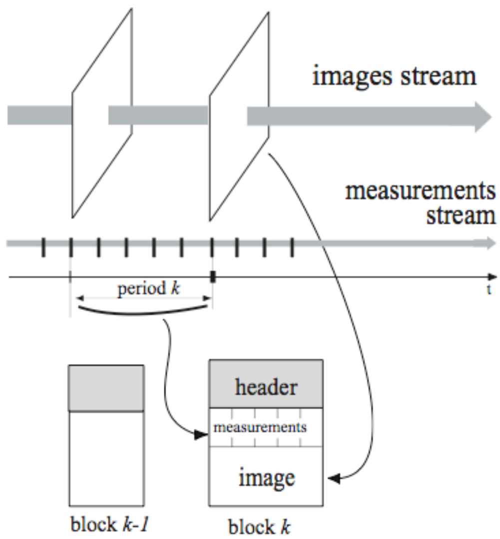

Fig. 4 Block abstraction and content.

The beginning of a block starts with a unique value. Special care is taken to ensure that this value cannot be found in any other part of the transmitted data. This mark is used by the client application to re-synchronize the receiving process when losses occur. The block **timestamp** represents the server time at which the block has been created. This timestamp defines when the block has to be processed at the client side to ensure the real-time block playback. The block **period** represents the elapsed time between the current block creation and the next block creation. The block period is the inverse of the current block pace. The block **ID** incremented by one unit every block, is used to determine if one or more blocks where lost during the transmission. The block timestamp and block period carry redundant information since the current timestamp plus the period should be equal to the timestamp of the next block. This assertion only holds if there is no block loss during the transmission. If there is one or more lost blocks, the block timestamp and block ID will be used to render adequately the newly received block.

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 RTI2

Three key aspects define the quality of service for the RTI2 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 RTI2. Since there are no direct RTI2 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.

*Def. 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 level of interaction can be directly measured by the block round trip time. This measurement takes into account not only the time for the block to transit from the server to the client but also takes into account the processing time at both the client and the server applications. These metrics should be as small as possible to ensure a valuable user experience. Many causes can increase the block round trip time, some are controllable such as the buffers at the server and the client side. Like others, the intrinsic Internet connection round trip time cannot be controlled. Block acknowledgements measured at the application level are different than network packets acknowledgements measured at the network layer level.

The next two metrics represents the E2E layer ability to handle the incoming flow of information.

*Def. The **block pace ratio τ** is the ratio between the block pace measured at the client side and the block pace chosen at the server side.* This metrics ensures that the server transmission parameters are adequately chosen to satisfy not only the network transmission characteristics but also to the client characteristics. For a given bandwidth, the server application can generate different block sizes and block paces. For example, a bandwidth of 100 Kb/sec corresponds to 10Kb blocks sent 10 times per second, but also corresponds to 20 Kb blocks sent 5 times per second. If the client computer such as a PDA were not able to decode and play more than 5 images per second, half of the image would be wasted with the first set of values.

*Def. The **block size ratio ε** is the ratio between the successfully rendered block size and the original block size.* The successfully rendered block size corresponds to the portion of the block data that was effectively processed and perceived by the user. Blocks can be lost along the E2E transmission path either during the network transmission or at the client side if the client application cannot handle the incoming flow of information. The underlying network protocol can guaranty whether or not the packets are delivered. In the former case, such as in TCP, it can be assumed that the block size ratio is always 1 provided that we wait long enough to receive the complete block and that the client application has enough resource to process the received block. In the later case, without guaranteed delivery, such as in UDP, information can be lost during the transmission. This metrics informs the source about the amount of information that has been perceived by the user. An information loss can either be due to a transmission problem, a lack of resource in the client application, or due to a client application mechanism to stop the current block rendering when a new block becomes available.

4. END-TO-END ADAPTATION SCHEME

RTI2 is successful when the acquired information is entirely rendered at the same pace it has been acquired with a minimal time delay between the acquisition phase and the playback phase. This translates to τ and ε equal to 1 with *d* as small as possible.

The proposed metrics characterize the end-to-end information transmission path as a whole. The block pace *p* and block size *S*, chosen at the server side, are the control signals used by the adaptation scheme depicted in Fig. 5.

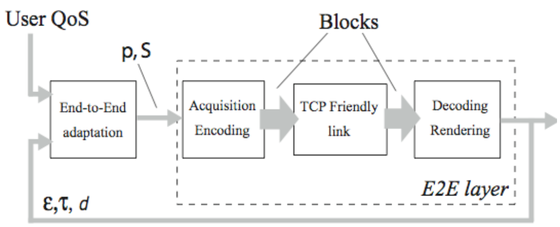

Fig. 5. End-to-End structure and adaptation scheme

To guaranty a friendly transmission, the TCP protocol is used to transmit blocks. This also insures that any block is fully transmitted, resulting in a block size ratio ε always equal to 1.

The various buffers found along the transmission path can be partially filled up resulting in an additional transmission delay. The adaptation scheme should ensure that the equivalent E2E buffer stays empty thus guarantying a minimal transmission time *d*.

When the equivalent E2E buffer fills up, the adaptation scheme must react by diminishing the amount of transmitted information. It can either lower the sending pace or reduce the block size. The later option is chosen since the block period is selected to permits the rendering of the dynamics of the physical system under study (i.e. *p* constant). The value of the τ metrics indicates whether the chosen block period *p* is reachable by the E2E layer.

The block size is adjusted by an integral controller by comparing the achieved d to the user defined d_{ref} representing the maximum time the user is willing to wait for the feedback to its actions (Fig. 6). As mentioned previously, the block size ratio ε is always equal to 1 because of the use of the TCP protocol for the transmission. The constant sending period $\overline{p}$ is chosen such that the stationary block pace ratio $\overline{\tau}$ is equal to 1 for any block size S.

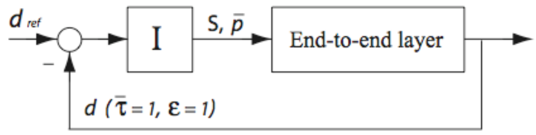

Fig. 6. E2E adaptation scheme for tracking *d*.

Fig. 7 shows the evolution of the E2E transmission delay d of a TCP connection established over a wireless Bluetooth link subject to perturbations. The reference value for the transmission delay d_{ref} is 400 ms and the constant sending pace $\overline{p}$ is 200 ms. The x-axis represents the block index k.

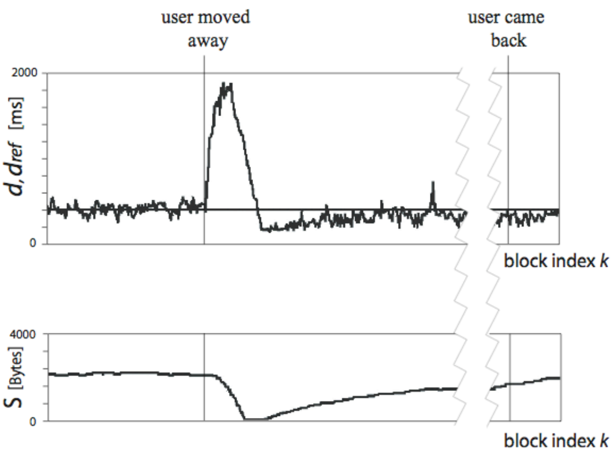

Fig. 7. E2E transmission delay (upper part) and block size (lower part) of a TCP transmission over a Bluetooth connection.

At some point, the client application running on a Palm handled device moves away from the Bluetooth transmitter; the resulting effect is a sudden increase in the transmission time due to the equivalent TCP buffering effect. The implemented integral controller reduces the sent block size, hence emptying the equivalent E2E buffer prior to adapting the block size to a new value resulting of the increase of distance between the Bluetooth transmitter and the PDA. Later the user comes back to its original position with the result in an increase of the block size S to its initial value.

5. CONCLUSIONS

In this paper, the metrics needed to measure the quality of service for real-time interaction over the Internet have been introduced. The metrics measurements are made at the application level in order to take into account the whole transmission path from the information capture at the server side to the information rendering at the client side. The proposed metrics are derived from the three key aspects of RTI2 (the level of interaction, the perception of the dynamic and the semantic content) that define an efficient user experience. These keys aspects are represented by three quantities - the information transmission delay, the information delivery pace and the information size – that can be measured. The block, the basic unit of information, and component abstractions are introduced to permit the achieved QoS estimation by the metrics. These metrics are used to develop an adaptation scheme that guaranty an efficient user experience. The proposed strategy has been successfully implemented to adapt the block size to the E2E structure characteristics and also to reject the perturbations induced by the transmission time variations when using a Bluetooth link with a mobile device.

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.