POLITECNICO FEDERALE DI LOSANNA
SWISS FEDERAL INSTITUTE OF TECHNOLOGY LAUSANNE

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Communication Systems Division (SSC)
EPFL CH-1015 Lausanne, Switzerland
http://sscwww.epfl.ch

Synchronization of Telemedicine Applications: Specification and Implementation

Raffaele Noro, EPFL-DE-TCOM

July 1997

Technical Report SSC/1997/019

Contents

Abstract

Telemedicine refers to the use of networked computers to perform health care tasks, like diagnosis, therapy and follow-up. Medical images represent the essential information exchanged by telemedicine systems and are characterized by specific requirements concerning acquisition, storage, transmission and analysis modalities. In the telemedicine systems, medical image communication is performed over Transmission-Control Protocols (TCP). Real-time protocols for ATM-based multimedia networks are under study. We propose a new system for medical image communication where the physician is enabled to browse remote medical images, at high-resolution and in real-time.

This system has two innovative features: a better organization of the image data for a fast access to the high-resolution images (hierarchical database) and the integration of synchronization mechanism at the visualization level of the application (requirements, specification and implementation).

This document addresses a method to facilitate the design of a synchronization mechanism for telemedicine applications: with this method, a Time Petri Net model of the system is obtained and the end-to-end synchronization requirements are transformed in component specific parameters. In a second phase, the Petri Net model is used to design the synchronization mechanism and to guarantee that synchronization requirements are fulfilled at run-time, monitoring and modifying the state of the system, according to the resource usage and the specified behavior.

Keywords: telemedicine, synchronization, Petri Nets.

1 Introduction

Health care practice today is moving toward telemedicine, that is connecting peripheral institutions and competences to major medical facilities through networked computers.

This currently involves health care teaching, off-line consultation and examination and, in the future also intervention and on-line consultation.

Such a trend in health care practice is facilitated by the synergy with multimedia technology, bringing to the integration of patient information, personal communication services and medical modalities in a unique platform.

Current multimedia technology permits the development of networked multimedia applications for medicine. The goal is to improve patient care quality and medicine education through these applications.

The convergence of telemedicine into multimedia is motivated by the fact that those applications typically require the treatment of data of different nature in an integrated fashion [1, 2]: medical images, video and audio conferencing, text, reports, statistics are the main types of information involved in medical environments.

Medical imaging plays the major role in the development of multimedia applications for health care: diagnosis, therapy and follow-up are founded on the analysis of medical images [3]. Unfortunately, there is not a unique class of medical images: each speciality is characterized by image size (128^2 pixels for nuclear medicine, 4096^2 for digital mammography), color depth (12-bit gray levels for magnetic resonances, 24-bit colors for pathological specimen) and pattern (motion video in cardiology, spatial sequence in computed tomography).

The systems in use were developed regardless of the specificity of medical environments: in fact, they deal with medical images like textual information, operating in store-and-forward mode and lacking of real-time performance (i.e., no image compression, no interactivity, replication of information, file transfer protocol).

The systems of the future are expected to provide fully integrated and collaborative environments [4, 5]; research is therefore focusing on the optimization of medical image processing (lossless, wavelets and JPEG), on database management capabilities for high resolution medical images (standards like DICOM[1] to be supported by PACS[2]), on the 3-D visualization tools and on the real-time performance of networked medical applications.

The real-time performance is the focus of our investigation. In the remainder of this document we describe the implementation of a prototype for remote medical diagnosis with real-time capabilities. The prototype was

[1]DICOM: Digital Imaging and Communication in Medicine

[2]PACS: Picture Archive and Coding System

developed in our laboratory [6, 7].

This document is organized as follows. In Section 2 the scenario of a telemedicine application is presented. In Section 3 a new architecture and the components that enable real-time operations are introduced, while in Section 4 innovative issues concerning the computation of low-level synchronization requirements are addressed through the described architecture. In Section 5 a mechanism for guaranteeing synchronization at run-time are proposed, that are obtained through a Petri Net model. In Section 6 we conclude with a review of the proposed approach and some directions for future investigation.

2 Identification of the scenario: a networked medical imaging framework

In telemedicine, diagnosis and consultation are carried out with a visualization station connected to a remote image server via network. The physician queries a stored image, which is transmitted and browsed on the local screen. If further information is required or a new image is necessary for the examination (this is very often the case), a new query is sent to the server and the new image is received by the client station. A scheme of these operations is presented in Fig. 1.

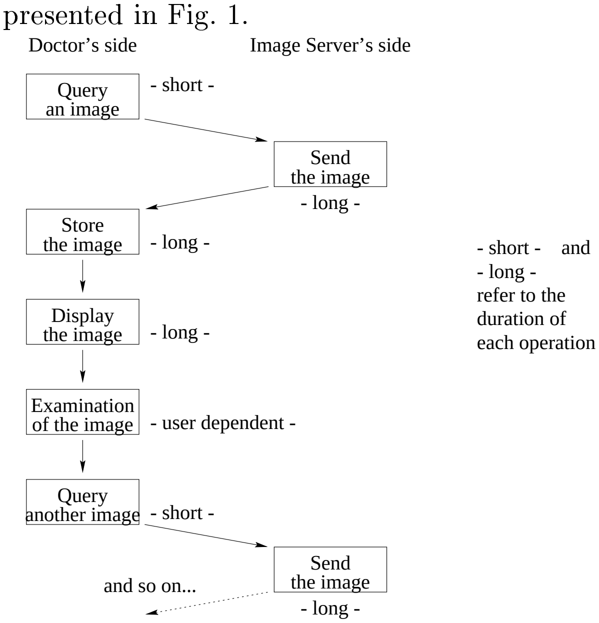


Fig. 1: Scheme of remote diagnosis operations.

So far, the nature of the information exchanged by the client and server is undefined: several options are possible, both for the queries and for the image format. For example, the queries can be filled forms including patient identifier, patient record, image folders and so forth; in this case, the server transmits, through file transfer protocol, the requested set of images. Alternatively, the query refers to a selected portion of a previous image, so the server transmits a single image per request. In Table 1 we present a set of possible combinations.

Table 1: Options for the implementation of a medical imaging application.

Option	Nature of query
a	The complete clinical folder, perfect quality
b	Selected image(s) of the clinical folder, perfect quality
c	Overview of the clinical folder, good quality
d	Magnification of an area or sub-image, perfect or good quality
e	Displacement of the area or sub-image, good quality

Option	Nature of transmitted data
a	Full resolution, all images, no losses
b	Full resolution, selected images, no or few losses
c	Down-scaled images, all images, few losses
d	Enhancement of previous data, no or few losses
e-1	Incremental information, no or few losses
e-2	Refresh image, few losses

Option	Technologies
a,b	FTP, local storage and browsing
c-1	FTP, local storage and browsing
c-2	Teleconference
d-1	FTP, local storage, processing and browsing
d-2	Teleconference or remote browsing
e-1	FTP, local processing and browsing
e-2	High-speed transfer, local processing and play-out

The choice between these option is determined by several technical, economical and human factors; in the existing systems for teleradiology, for example, the complete patient folder is transferred through FTP, stored into the local disk, browsed and examined (options *a* and *b* in Table 1). Such a solution is considered because easier to implement, robust and based on widely deployed network technologies, but has the disadvantage that user is faced to very long delays, no real-time capabilities and poor degree of interactivity. To obtain better performance, it is necessary to design the application in a different manner, primarily employing high-speed network technologies. The user, today, expects to operate in a *natural* environment where images are browsed a few *ms* after the request, images can be dynamically selected, and requests are issued through a simple user interface. The method that we propose we have captures these three features; it is schematically represented in Fig. 2.

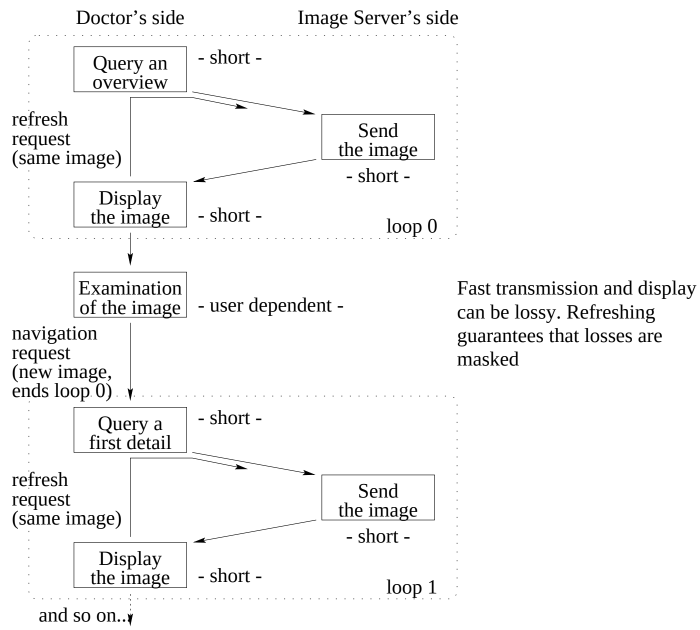


Fig. 2: Sequence of operations with the new method.

The advantage over the traditional methods is that, while operating with the same basic information, the physician can made the examination faster, analyze more images and produce the diagnosis faster.

Several technological aspects support the implementation of this method. First, we identify a suitable architecture (in terms of hardware/software components) that provides the desired features. Second we quantify the end-to-end temporal relationships and component-specific requirements that are associated to each operation and represented qualitatively by - *short* - and - *long* -. Third, we guarantee that the dynamic behavior of the system (observed at run-time) matches with the static model computed through the architecture components and the temporal relationships.

The relationship dynamic and static behavior is maintained by monitoring and adapting the information flow at the component level, in order to match a set of component-specific requirements.

A straightforward quantification of the end-to-end temporal relationships is easily obtained. In Table 2 we show some suitable values. We also indicate some flexibility is allowed for some of them, which allows the system to be recovered from synchronization errors.

In the proposed medical imaging system proposed, we need to solve three problems: first, find an appropriate architecture for the overall system (i.e., the components and their interactions); second, define and transform end-to-end relationships in component specific requirements (i.e., a static

Table 2: Temporal and non-temporal relationships and degree of flexibility.

Parameter	Specified value	Characteristics
Connection set-up	2 s	Very flexible
Browsing of a first overview	2 s	Very flexible
Latency between request and display of further images (turnaround time)	0,7 s	Mandatory
Refresh rate of the inspected image	10 frames/s	Flexible (5÷20 fr./s)
Sampling of selection at user interface	10 samples/s	Flexible (5÷20 samples/s)
Image size	Natural size	Flexible (± 10 %)
Color depth	24÷8 bits/pix.	Flexible (min. 8 bits/pix.)
Image quality	High	Flexible (min. good)

model and the degree of abstraction); and third, provide the mechanism that guarantees and restores synchronization.

In the next Sections we provide solutions to each of the above problems, and we evaluate the obtained benefits through a prototype implementation of the proposed solutions.

3 Architecture components for networked medical imaging

Networked multimedia medical applications are designed as client-server applications. They enable the exchange of medical image data across the network and optionally allow the establishing of audiovisual communication among end-users. In the conventional applications, the exchange of medical information relies on file transfer protocols and no particular attention is dedicated to other network aspects.

Providing real-time capabilities is hard or even impossible in such context. Considering the huge amount of data that such applications are expected to exchange, it is clear that high-speed networking technologies must be employed. For developing our prototype, ATM is the selected technology.

Furthermore, the amount of data associated with medical images have to be properly reduced, in order to use network and host resources in an optimal manner. We employ, therefore, compression/decompression schemes and we provide the user only with the essential information. In fact, we have noticed that, before the examination of an high-resolution image, the physician is interested in the general overview of the picture(s) and, in a second phase, he focus onto the part(s) that he considers of major clinical content.

We argue, therefore, that a hierarchical organization of medical images should be appropriated. The examination starts from a low-resolution overview of the picture(s), and the user is enabled to enhance the resolution of a selected area of this image. He sends to the server the request corresponding to the selected area. Then, the server transmits the corresponding high-resolution version of the specified image, until the maximum allowed resolution is retrieved.

In summary, we consider the following basic components: a hierarchical access and browsing of medical images; a compression/decompression module; and an ATM network with the corresponding network adaptation layer. A video-conference application can also be integrated in the platform, enabling user-to-user communication (provided that the medical part is efficient, no particular performance is required). The Fig. 3 represents the set of components.

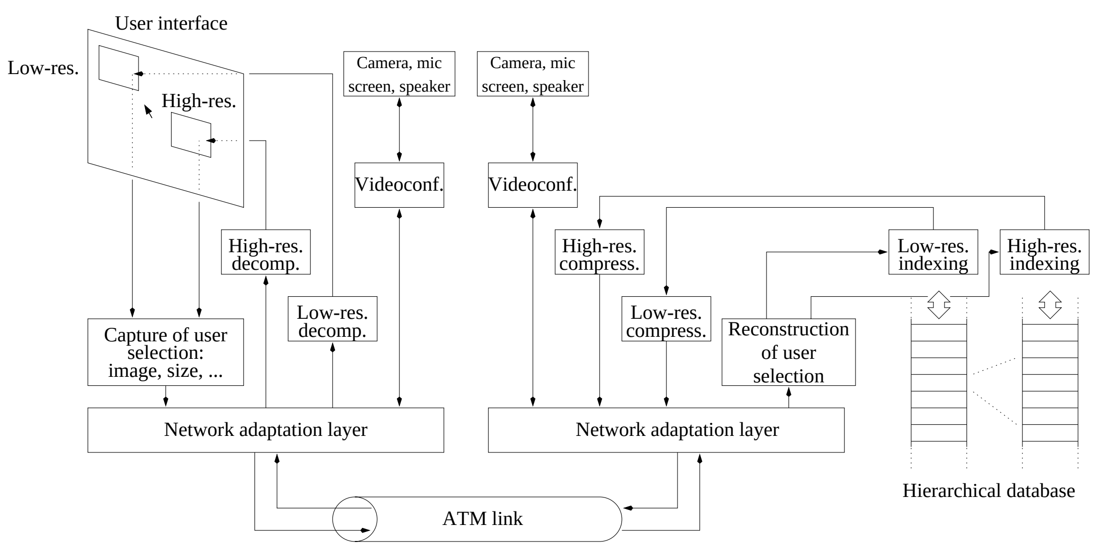


Fig. 3: Components to perform real-time medical imaging.

It is noticeable that this system allows the user to have a continuous control of the information flow and permits to perform and repeat efficiently the examination, in the shortest possible time.

Different synchronization aspects characterize the proposed system. In the following Section we develop a model that captures end-to-end requirements and maps them into component specific requirements, making use of Petri Net formalisms. Such a model describes quantitatively the expected temporal behavior at application run-time.

4 Specification of synchronization requirements

In the system for real-time medical imaging of Fig. 3, images are coded and transmitted from the server to the client station. The user is enabled to select, dynamically within the received images, an area that can be displayed with an higher resolution level.

Some end-to-end timing requirements are specified as in Table 2 for an appropriate use of the system, that are however independent from the employed architecture. Moreover, these requirements are difficult to handle directly by the application, because of the number of factors and components that contribute to the end-to-end delay. Instead, the end-to-end parameters can be mapped into component-specific parameters, corresponding to the logical components of the architecture.

Petri Net formalisms are used to do this mapping. Petri Net nodes model each architecture component and make easier the computation of the component-specific requirements that fulfil the end-to-end requirements.

4.1 Petri Nets principles

Petri Nets [8] are composed of a set of nodes partitioned into two disjoint subsets: $\mathcal{P}$ (places) and $\mathcal{Q}$ (transitions). Arcs connect places with transitions, but never two places or two transitions. Petri Nets have the advantage, with respect other formalisms, to be easy to represent in a graphical manner. Places are drawn as circles and transitions as bars. A complete Petri Net is also characterized by an initial marking that assigns tokens to each place. An example of Petri Net is given in Fig. 4.

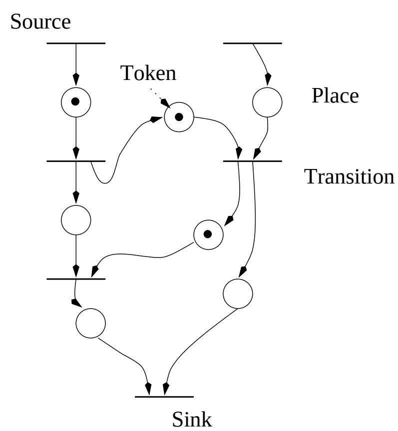


Fig. 4: A Petri Net with places, transitions and token.

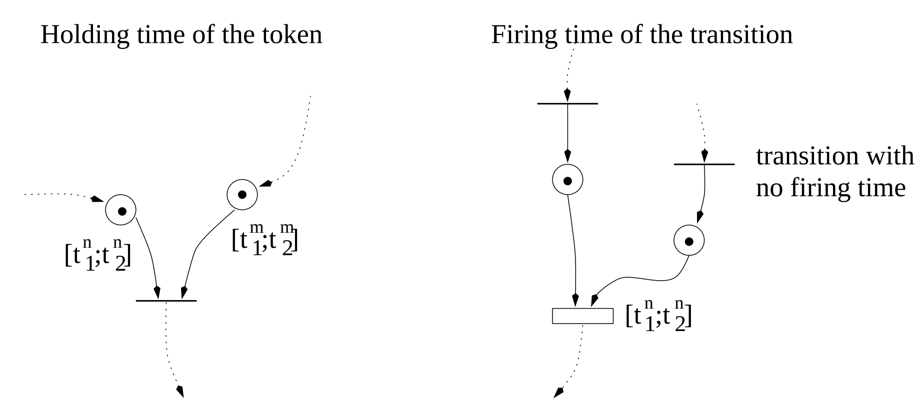
Places stand for conditions, transitions for events. A transition (event) has a number of input and output places (pre- and post-conditions). The presence of a token in a place indicates that the condition associated with that place is fulfilled. The basic rules of Petri Nets are:

- a transition is *enabled* if each upstream place contains at least one token;

- a *firing* of an enabled transition removes one token from each upstream places and add one token in each downstream place.

In the recent years, Petri Nets have been employed for multimedia synchronization specifications and have been extended to cope with temporal relationships. Indeed, questions concerning to when events take place are not addressed in the original theory of Petri Nets.

This have been obtained by associating durations to transitions or to sojourn of tokens in places: *firing* and *holding* times, respectively. The firing rules, therefore, are modified to enable logical relations between upstream places. Fig. 5 and the Table below represent these extended features of Petri Nets.



Firing rules for Time Petri Nets

Rule	Semantic
AND	$\theta \in \bigcap_i [t_1^i; t_2^i]$
OR	$\theta \in \bigcup_i [t_1^i; t_2^i]$
MASTER(i)	$\theta \in [t_1^i; t_2^i]$
...	and similar ones

Fig. 5: Petri Nets with timing features. Up-left, holding time and up-right, firing time.

Techniques to model multimedia synchronization were then derived, that make use of Petri Net theory: Object Composition Petri Net (OCPN, [9]),Time Stream Petri Net (TSPN, [10, 11]), Hierarchical Time Stream Petri Net (HT-SPN, [12]) and others.

We refer the reader to the existing literature for an extensive study of the original and extended Petri Net theories. Here, our interest is in the use of Time Petri Nets for the implementation of synchronization in the context of our medical imaging system.

4.2 Modeling issues of the system

A Petri Net model is a way to describe in a quantitative, detailed and simple manner the ordering and timing of events in the studied system. In addition, it can be further employed for the design of the synchronization mechanism and eventually for computing the required network and host resources.

In the case represented in Fig. 3 each architecture component is responsible of processing part of the information, causing the information flow to be stopped for the corresponding time.

The architecture is mapped on a Petri Net model by associating each component to a place and each processing time to the holding time of that place. Hence, tokens represent the units of information that are produced, consumed or transmitted by each component. The Fig. 6 reports such mapping.

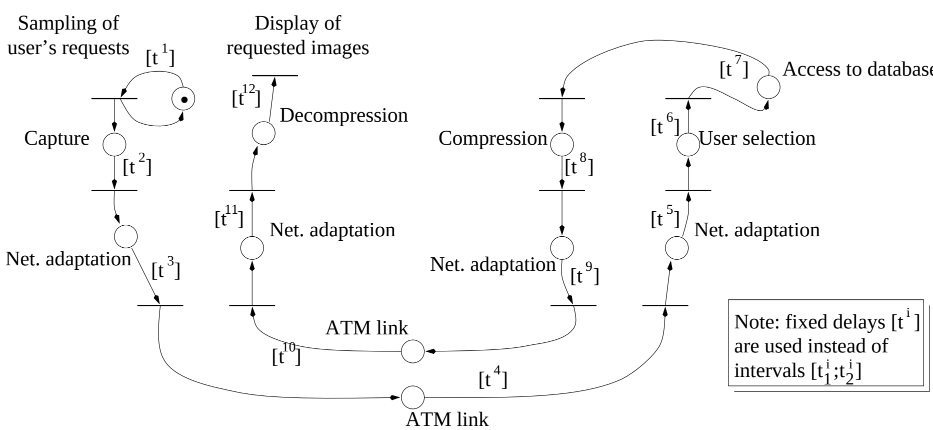


Fig. 6: Time Petri Net model of our medical imaging system.

The holding times in Fig. 6 are left undefined. They depend on two factors: the amount of information carried by each information unit (or token) and the consumption rate of the component (or place).

A practical assumption is that the information moved in the client to server direction, corresponding to the user selections, is negligible (at least with respect the amount of information moved in the opposite direction). Hence:

$$t^2 = t^3 = t^4 = t^5 = t^6 \simeq 0 \ s$$

A second assumption is that access, adaptation, decompression and compression times are proportional (for the sake of simplicity we consider them equal) independently of the image size:

$$\frac{t^8}{t^7} = \frac{t^9}{t^7} = \frac{t^{11}}{t^7} = \frac{t^{12}}{t^7} = c = 1$$

The end-to-end specifications of Table 2 are then fulfilled if:

$$\begin{aligned} t^1 &= 0.1\ s \\ t^7 + t^8 + t^9 + t^{10} + t^{11} + t^{12} &< 0.7\ s \end{aligned} \tag{1}$$

Equation (1) can be satisfied fixing some operational values as follows:

- size of information units (average),

$$\begin{aligned} \text{user requests :}\quad & negligible \rightarrow 0\ bits \\ \text{original images :}\quad & (400 \times 400\ pix) \times 8\ bit/pix = 1.28\ Mbit/image \\ \text{after compression (factor 10) :}\quad & 0.128\ Mbit/image \end{aligned}$$

  and,

- consumption rate of components/places (average),

$$\begin{aligned} & \text{access to database time, compression time,} \\ & \text{net.adaptation time, decompression time,} \\ & \text{all} \ \leq\ 0.1\ s \end{aligned}$$

  and,

- bandwidth of the ATM link (server-to-client channel) fixed by image size and rate (average),

$$\text{Mean rate (image size/image rate)} \quad = \quad \frac{0.128\ Mbit/image}{0.1\ s/image} = 1.28\ Mbit/s$$

The corresponding Time Petri Net model with the computed values is represented in the Fig. 7.

This model represents an *average* behavior in a *stable operative state.* For the actual system, any kind of exception can occur, originating instability and synchronization errors (i.e., systematic violations of end-to-end temporal relationships). It is, therefore, the task of an appropriate mechanism, running at application run-time, to check the consistency of the relationships and guarantee an appropriate evolution of the system.

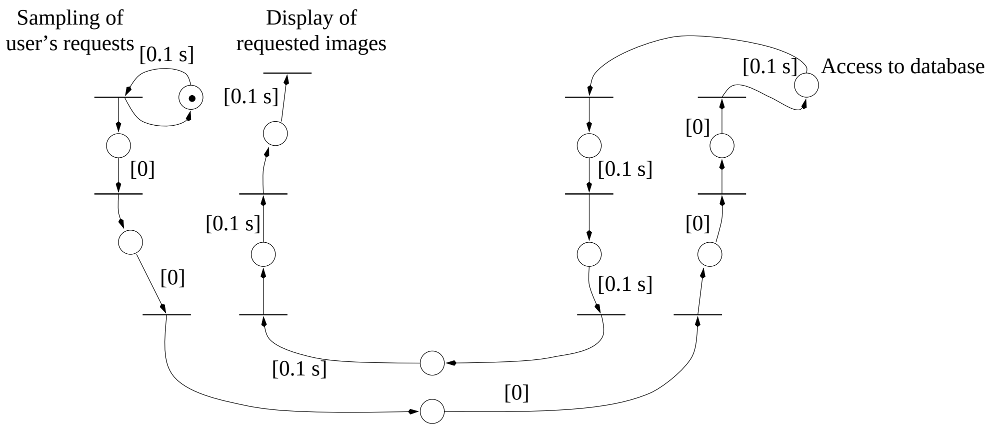


Fig. 7: Time Petri Net model with operative values.

5 Synchronization mechanism

So far we have obtained a Time Petri Net model of the medical imaging application that takes into account an average behavior of the system. The point is that, in telemedicine and in multimedia applications in general, occasional violations of the expected conditions occur, that induce the system to run out of synchronization.

The task of a synchronization mechanism is to monitor the state and modify, when and where possible, the information rate generated by the source in order to keep the global state as close as possible to the stable point. The application needs to be adaptive and the mechanism performs the adaptation tasks.

We will show that the implementation of the optimal mechanism is facilitated by the Time Petri Net model discussed above. Considering situations in which one place holds tokens/units of information more than the expected holding time (fixed by the model), the obtained behavior is slightly different from what indicated by Table 2. In Fig. 8 is represented the case of a longer decompression holding time and a shorter network adaptation holding time at client side.

Even if a shorter network adaptation time seems to compensate the longer decompression, a bottleneck appears at this node, due to the fact that delay tends to accumulate.

If no adaptation is performed, the whole set of end-to-end relationships is violated. The end-to-end performance of the application is reduced by a single bottleneck.

Intuitively, it is possible to handle this situation by reducing, somewhere, the information rate in order to not saturate the bottleneck; this will result in the reduction of one or more end-to-end relaxable parameters, while maintaining the remaining to the fixed values. In our case, we maintain a fixed

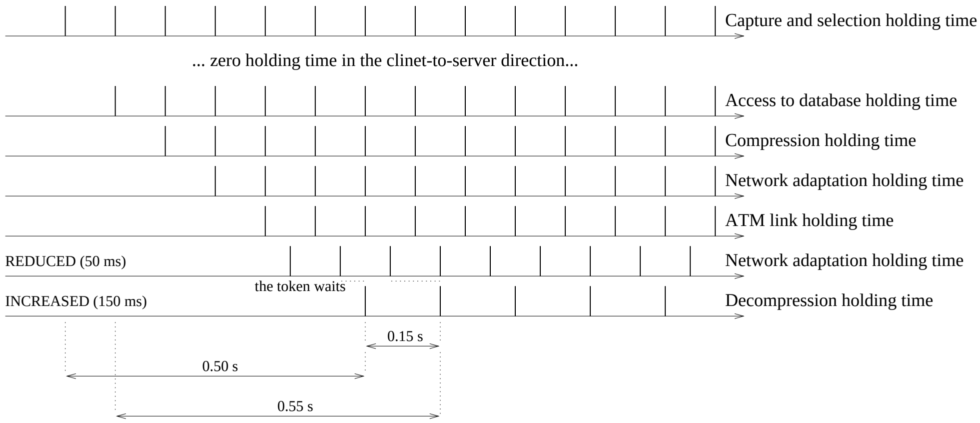


Fig. 8: Syncronization is lost due to decoder delay.

turnaround time.

The mechanism performs automatically this adaptation procedure. Its tasks are, therefore, the following:

1. recognize the relaxable parameters,
2. recognize the bottleneck,
3. request upstream to reduce the rate until the consumption rate of the bottleneck supports the incoming rate.

Tasks 2 and 3 must be performed jointly and can be iterated.

Making use of the Time Petri Net model, token counters are associated to each place, reporting the status of the node, as represented in Fig. 9.

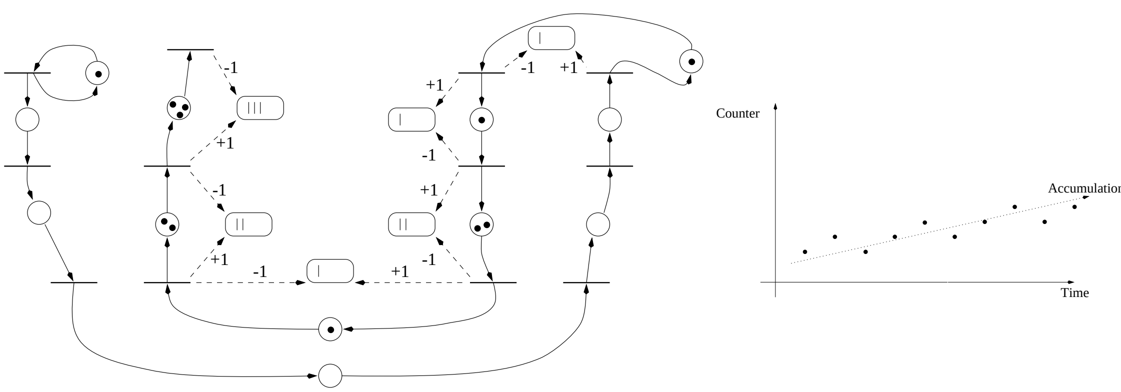


Fig. 9: Counters associated to the nodes and accumulation of tokens.

The bottleneck appears in the unstable nodes; if tokens keep accumulating in a specific place, the upstream place is generating much more information. The mechanism notify the upstream place that the rate should be reduced. If this place is able to perform the required adaptation, the rate is adapted locally; if not, the notification is forwarded upstream until a suitable place is reached. It is, for example, the image compression place, where both the rate (images/s at fixed quality) or the quality (bits/frame at fixed rate) can be modified. Alternatively, the adaptation can be performed at the user request capture place, by reducing the sampling rate.

The sequence monitoring-notification-adaptation is iterated until the instability is removed.

The reverse operation is also performed, when system is recognized as stable; the information rate can be increased of a predetermined amount (user should feel more comfortable) until the boundary of instability is crossed again.

It is noticeable that initial values, like bandwidth, size of images, sampling rate and so forth, should be fixed once but the mechanism is allowed to modify them at run-time, according to the priority defined for each single parameter.

The mechanism that we have implemented, based on a Time Petri Net model of the system, is able to guarantee the temporal relationship expected at the user interface and to cope with occasional or permanent degradations of performance [13]. It can monitor and control all host and network components and, in case of synchronization errors, it can gracefully recover the expected performance.

5.1 Some implementation issues

In the actual application, the token counter is implemented computing the difference between incoming and out-coming units, and the instability is detected when such value has an increasing trend in a predefined temporal window. Upstream blocks are notified and adaptation is performed consequently.

In pseudo-code, counter and adaptation are implemented as follows:

```
/* counter */

      T= 0;                 /* starting time */
      tokens(0)= 0;         /* tokens held */
      window_width= ...     /* temporal window */

loop: wait(transition_event(T))

        if transition_event is ''+1'' then
                tokens(T)= tokens(T)+1;
                if [tokens(T)- tokens(T- window_width)]/window_width is
                 greater than ''threshold'' then
```

```
                notify(reduction);

        if transition_event is ''-1'' then
                tokens(T)= tokens(T)-1;

        goto loop;

/* adaptation */

      period= 0.1          /* initial rate */

loop: wait(period)

        generate(token);
        if notification_pending then reduce(period);

        goto loop;
```

According to the described mechanism, we have implemented a prototype of the medical imaging system.

This prototype, implemented in C, operates between in two Unix hosts (client and server workstations) connected through a direct ATM link.

The first prototype, implemented in June 1996 enabled the exchange of medical images but did not guarantee real-time. The successive integration of a synchronization mechanism (November 1996), as described in this Section, permitted to fulfil the end-to-end requirements of Table 2.

Key points in the implementation are the values of `window_width` and `threshold`, as well as the actual operations performed by `reduce(.)`.

6 Conclusion

Digital imaging for medicine is going to play a major role in the future practice of health care. So far, the applications developed for medicine provide the ability to manage and communicate medical data, in particular images. They operate with plain transmission of image files for a subsequent interpretation, without providing the ability to exchange, in real-time, patient data and related information.

In the next future, new applications are expected to integrate such features in a more general networked multimedia framework for medicine.

The optimal use of data sources and infrastructure resources is still an investigation area. We have proposed a method for the development of networked medical imaging applications allowing real-time exchange of images.

This method is based on the access to medical image data in a hierarchical fashion, enabling the doctor to choose and obtain, on-line, the most important clinical information.

Real-time constraints are imposed to our system. We have also studied, within this method, how to guarantee real-time access and display of medical images stored in a remote server.

We have defined the logical blocks that build-up the medical application and analyzed their timing characteristics. Through these blocks, we have obtained a Time Petri Net model of the system that captures and implement end-to-end temporal constraints. The same model is then used to monitor and control the run-time behavior, maintaining the most relevant end-to-end parameters and relaxing some others in case of instability. Fig. 10 summarizes the proposed approach.

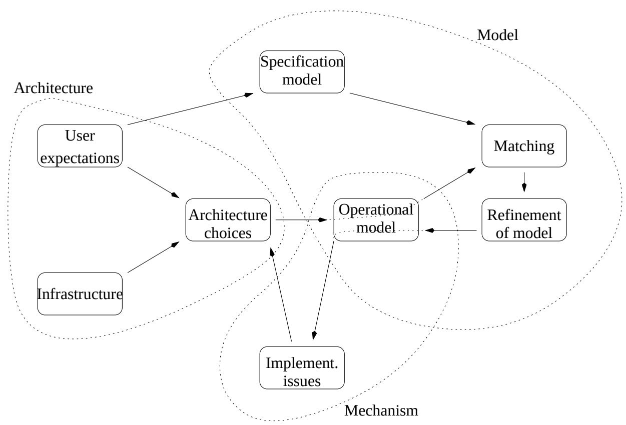


Fig. 10: An overview of the proposed approach for medical imaging

Finally, we also implemented a prototype in UNIX workstations connected through ATM links, according to the proposed method.

The applicability of this method to the design of multi-point/multi-party applications, based on a similar approach, should be investigated. We should also take into account the impact of heterogeneity in the infrastructure and network resources that can occur in wide scale networked applications. It is also necessary to validate and improve our method through a set of clinical trials.

References

[1] R. Steinmetz and K. Nahrstedt, *Fundamentals in Multimedia Computing and Communications*, Prentice - Hall, Englewood Cliffs, NJ, USA, 1995.

[2] G. Blakowski and R. Steinmetz, "A Media Synchronization Survey: Reference Model, Specification, and Case Studies", *IEEE JSAC*, vol. 14, pp. 5–35, January 1996.

[3] S.T.C. Wong and H.K. Huang, "Networked Multimedia for Medical Imaging", *IEEE Multimedia*, vol. 2, pp. 24–35, April-June 1997.

[4] ACTS Programme, *BONAPARTE AC022 Project*, EU, 1995, Technical Annex.

[5] M. Eichelberg, E. Cordonnier, J. Piqueras, A.J. Hewett, and P. Jensch, "RETAIN: Multimedia Teleradiology on the Pan-European Information Superhighway", *in CAR '96*, pp. 531–536, Paris, June 1996.

[6] A. Basso, O. Verscheure, and R. Noro et al., "A Multimedia Architecture for Medical Tele-Imaging over ATM", *in CAR '96*, Paris, France, June 1996.

[7] R. Noro, J.P. Hubaux, R. Meuli, R. Laurini, and R. Patthey, "Real-Time Telediagnosis of Radiological Images through an Asynchronous Transfer Mode network: The ARTeMeD Project", *Journal of Digital Imaging*, vol. 10, pp. 116–127, August 1997, Suppl. 1.

[8] F. Bacelli, G. Cohen, G. Jan Olsder, and J.P. Quadrat, *Synchronization and Linearity*, John Wiley & Sons, 1992.

[9] T.D.C. Little and A. Ghafoor, "Spatio-Temporal Composition of Distributed Multimedia Objects for Value-Added Networks", *IEEE Computer*, vol. , pp. 42–50, October 1991.

[10] M. Diaz and P. Sénac, "Time Stream Petri Nets, a model for multimedia streams synchronization", *in Multimedia modeling '93*, Singapore, November 1993.

[11] P. Sénac, M. Diaz, and A. Léger et al., "Modeling Logical and Temporal Synchronization in Hypermedia Systems", *IEEE JSAC*, vol. 14, pp. 84–103, January 1996.

[12] P. Owezarski, *Conception et formalisation d'une application de visioconférence coopérative. Application et extension pour la téléinformation*, PhD thesis, LAAS-CNRS, December 1996.

[13] R. Noro, "Un mécanisme distribué de synchronisation appliqué à un cas de téléconsultation", *in JDIR '96*, Paris, France, September 1996.