Specification of synchronization in multimedia conferencing services using the TINA lifecycle model

To cite this article: Martin Jacobs and Peter Leydekkers 1996 Distrib. Syst. Engng. 3 185

Related content
- Some principles for quality of service management
  Gregor v Bochmann and Abdelhakim Hafid
- CORBA and RM-ODP: parallel or divergent?
  N Dunlop, J Indulska and K A Raymond
- A QoS adaptive multimedia transport system: design, implementation and experiences
  Andrew Campbell and Geoff Coulson

Recent citations
- Approaches to end-user applications portability in the cloud: A survey
  Sami Yangui et al
- Ahmad F. B. Alam et al
Specification of synchronization in multimedia conferencing services using the TINA lifecycle model

Martin Jacobs and Peter Leydekkers†
KPN Research, PO Box 15000, 9700 CD Groningen, The Netherlands
Received 22 November 1995

Abstract. Multimedia conferencing services have specific performance requirements with respect to the exchange of continuous media. One of these requirements is the synchronization within and between related streams. This article identifies synchronization requirements and solutions relevant for conferencing services that operate in an open distributed environment as defined in TINA-C. Based on the TINA service lifecycle, synchronization requirements and policies are described from different stakeholder perspectives. Synchronization is then specified in detail using the TINA information, computational and engineering languages. Finally, a description of the implementation of synchronization for a multimedia conferencing service is provided using a DPE platform. The synchronization object is proposed as an additional component available for the DPE that needs to deal with real-time audio-visual streams.

1. Introduction

The Telecommunications Information Networking Architecture Consortium (TINA-C) is a world-wide initiative that propagates a distributed processing environment called the TINA-DPE. The TINA-DPE facilitates the interworking between different stakeholders such as public network operators, end-users and service providers offering services such as video on demand or audio-visual conferencing facilities. The TINA-DPE should meet the requirements of heterogeneity and interoperability and it distinguishes itself from other developments in the sense that it combines developments in the IT-industry such as distributed computer platforms, with specific telecommunication issues like connection management. The TINA-DPE hides the complexity of remote communication from the applications, such as the establishment and release of connections and heterogeneity in data presentation. The TINA-DPE provides several forms of distribution transparencies to applications to hide the details of complexity such as access, location, failure, concurrency, replication and migration transparencies [28]. The TINA-DPE can be regarded as the infrastructure or platform on which distributed multimedia applications will operate.

Real-time audio and video exchange is an important aspect of distributed multimedia services and the ability to control and coordinate multiple audio/video streams is identified as an important requirement for distributed computing platforms [18]. This article focuses on three synchronization relations relevant for audio and video exchange in a distributed environment, see figure 1. First, voice and video samples need to arrive in time at the receiver before display or playout time to maintain the continuity of playback. This is called intra-stream synchronization. Second, inter-stream synchronization is needed to present voice and video samples in a certain relation. Different forms of inter-stream synchronization exist of which lip-synchronization is a well known one. Third, all participants in a multimedia conference should receive audio-visual data at the same time although geographically distributed. This form of synchronization is called spatial synchronization.

This article is based on the state-of-the-art regarding audio/video synchronization. Solutions found in the literature are summarized for the three identified synchronization requirements. We use the TINA lifecycle model as a roadmap to structure the specification, and in addition the TINA/ODP information, computational and engineering languages to specify audio/video synchronization in detail. This article shows the complete specification trajectory from analysis to implementation for a specific problem domain. Besides specifications we show how audio/video synchronization can be implemented on the TINA-DPE. This part is currently missing in the TINA architecture.

This article is structured as follows: section 2 presents a short overview of the TINA lifecycle model and the refinements added to specify synchronization. Section 3 presents an overview of the literature regarding audio/video
synchronization that is used as a basis for our work. Section 4 describes the analysis phase providing the requirements and policies from the different stakeholders involved. The specification phase in section 6 consists of information, computational and engineering specifications of our problem domain. The development phase in section 7 shows the realization of the synchronization function on a TINA-DPE platform. Finally, the conclusions are presented in section 8.

2. TINA lifecycle model

We use the TINA lifecycle model [1] for the specification of our problem domain: synchronization of continuous data streams for multimedia conferencing services. The lifecycle model defines an ordered set of steps that are required to support the development, deployment, and operation of a service. The lifecycle model is a combination of traditional software engineering methodologies and the activities required to operate, use, and maintain a service. We applied only a part of the lifecycle model (i.e., the construction lifecycle) since we were interested in the development of the synchronization service from a designer’s perspective. The construction lifecycle is apt for this purpose but it is rather general and does not provide many guidelines how to structure a specification in each phase. Therefore, we added more details to it using the ODP/TINA viewpoint languages for certain phases as shown in figure 2. It should be noted that the TINA lifecycle model is not a strict waterfall model (i.e., not a strict top-down approach) of system development. It is possible in each phase of the construction cycle to return to a previous phase if refinements and requirements are added during system development. This flexibility is also reflected in the ODP viewpoint specifications where it is possible to have, for instance, a detailed computational model before the information model is completed. It is possible to develop each ODP viewpoint specification independently in detail during the system development. This is in accordance with the ODP-RM philosophy of the use of viewpoints [20].

The construction phase is defined in [1] as ‘all the off-line activities required in designing and developing the software and any special hardware associated with a service’. The construction lifecycle is further decomposed into several activities:

- in the analysis phase the relevant synchronization requirements, obligations, and policies are identified from the different stakeholders involved. A stakeholder could be the end-user, service provider or network provider. The ODP enterprise language is used to express the requirements related to the synchronization service;
- the definition phase provides a description of the desired effect of the synchronization service as seen by the stakeholders. The description specifies what the synchronization service will do and not how it will be implemented. The ODP information language is suitable for describing the information aspects related to the synchronization service independent from any implementation;
- the specification phase provides a formal and unambiguous description of the synchronization service and synchronization mechanisms. It should include a specification of how the service can be implemented. For the specification of synchronization for multimedia conferencing services we will apply the computational and engineering concepts as defined in [27–29];
- in the verification phase the equivalence between the service specification (as a result of the specification phase) and initial requirements (result of the analysis/definition phase) are studied. In the case of differences adaptations have to be made;
- the development phase comprises the development of the software and hardware modules needed for the synchronization service. In this phase we use ANSAware that is a possible implementation of the TINA-DPE;
- the validation phase consists of verifying the developed software and possible hardware modules in the development phase against the (paper) specifications of the specification phase. The validation phase can be subdivided into the following two subphases;
- the conformance testing phase that includes checking the implementation for conformance to architectural rules and standards used in the design. Thus, it checks the conformity between the output of the development phase versus definition phase versus specification phase;
- the system testing phase that comprises the testing of software and hardware modules in a (possible) operational environment.

We focus in this article on the analysis, specification and development phases of the construction-lifecycle but first a short overview is provided of the state-of-the-art regarding synchronization techniques applicable to distributed multimedia services. Based upon this, the construction phase of the lifecycle model is applied to specify audio/video synchronization.

Figure 1. Synchronization in multimedia conferencing services.
3. Synchronization

The exchange of real-time continuous streams in multimedia conferencing services requires synchronization of these streams. Synchronization in multimedia conferencing systems refers to mechanisms used to coordinate the ordering of events (i.e., the playout of an audio sample, the display of a video frame) in time. This section details the three synchronization forms, and summarizes different techniques that can be used for synchronization.

3.1. Intra-stream synchronization

Audio and video streams are isochronous in nature. They are sequences of finite sized samples, created at fixed time intervals. A time relation between the stream units exists that must be preserved to offer a good quality of presentation. However, several factors may influence the time relation between the stream units:

- processing and network delay jitter (i.e., the variance in delay);
- variations in rates of recording and playback;
- unreliable transmission of stream data units.

To deal with network jitter the incoming stream data units can be buffered at the receiver. Discontinuity caused by processing in the end-nodes can be solved by buffering and careful process scheduling. Jitter can be reduced if the same constant rate is ensured at both the source and sink. However, small variations between the source and sink may cause buffer overflow or starvation (e.g., due to temperature differences). A number of solutions have been proposed in the literature to solve these problems. Examples are:

- **buffer monitoring:** the buffer occupation is monitored and beginning overflow or starvation is detected and dealt with [2, 3];
- **feedback technique:** the sink periodically transmits feedback messages to the source, containing the stream unit number that was currently played [4]. Due to these feedback messages, and the known bound delay on the network, the source can estimate the actual playback times of stream units at the sink and adjust the stream transmission rate;
- **global clock:** synchronization is achieved by means of a global clock. For example, with the stream synchronization protocol [7] each stream unit will contain a reference time when it should be played out at the sink. The reference time is calculated at the source of a stream unit using the global time and adding the delay and jitter across the connection.

Losing packets can also result in buffer starvation at the sink. Additional techniques must be applied for the detection of lost or late packets. For intra-stream synchronization, knowledge is required about the maximum jitter caused by transportation and processing. Network and processing jitter can be reduced by buffering and scheduling. This requires that sufficient buffer space should be available at the sink. However, a trade-off between buffer space and acceptable delay jitter will always exist: reducing jitter will require more buffer space. Especially with live media, the timeliness of data (i.e., the validity time of data) should be considered to determine the acceptable delay jitter.

Intra-stream synchronization can be easily achieved by ensuring rate-synchronized clocks (or devices) between the source and the sink. Buffer monitoring is a simple technique to deal with small rate variations of the source and the sink. The feedback and global clock techniques can provide an alternative, although they will increase communication overhead due to respectively feedback messages and the time reference.
3.2. Inter-stream synchronization

A temporal relationship may exist at the source between multiple continuous streams. This relationship must still hold after transportation of these streams through possibly different routes. This is assured when an inter-stream synchronization mechanism is applied. Solutions proposed in the literature are:

- **multiplexing of streams:** with this preventative technique, the different streams are merged onto one stream at the source, and separated at the sink. During the transport of the stream units, the relationship between these units remains fixed [14];
- **aggregation in one data structure:** one data structure is defined that is composed of multiple stream types including synchronization relations (e.g., a multimedia document [5]);
- **global clocks:** same technique as for intra-stream synchronization;
- **synchronization marker:** at the source, markers are introduced in the streams indicating sections that should be presented simultaneously [8]. A more advanced technique using logical time stamps is proposed in [9];
- **synchronization channel:** a synchronization coordinating node is created. Information about the presentation of the streams, such as playout rates, is periodically transmitted across separate synchronization channels between the coordinating node and the source and sink nodes [10];
- **feedback technique:** a similar mechanism to that presented for intra-stream synchronization although multiple sinks send feedback messages to the source.

Multiplexing is an easy and accurate solution to achieve inter-stream synchronization. Since the continuous streams are transmitted as one stream it is implied that different QoS requirements for individual streams cannot be met (e.g., different jitter bounds for audio and video). In this case, the synchronization marker provides a good alternative that is easy to implement and provides a similar accuracy.

3.3. Spatial synchronization

A number of receivers will exist in a distributed conferencing service. Especially with live audio-visual data, it is important that all participants in the conference receive the audio and video data at the same time, to maintain a fair conference. Spatial synchronization mechanisms are used for this purpose that are based on global clocks, synchronization channel or feedback techniques as proposed for inter-stream synchronization.

When global clocks are available, mechanisms based on these clocks can achieve the most accurate spatial synchronization. The stream synchronization protocol described in [7] is an example of a distributed synchronization technique whereas the synchronization channel and feedback technique require a more centralized synchronization manager.

3.4. Applied synchronization techniques

The previous overview accounted for the following choice of techniques applied in this article. Intra-stream synchronization is performed using a buffer monitoring technique. Accurate inter-stream synchronization is achieved using synchronization markers. Spatial synchronization will be performed by a centralized synchronization manager, which can apply all three techniques for spatial synchronization.

Using the feedback technique for the three types of synchronization would sound logical. However, the network delays are not known accurately on the ethernet that is used for our implementation. This means that the accuracy of the feedback technique is not sufficient and therefore not used. Similar motivations can be found for mechanisms based on global clocks since the clocks of a node in a distributed computer environment are not necessarily synchronized with each other. Additional motivations for the choice of the used synchronization techniques can be found in the engineering specification.

4. Analysis phase

The ODP enterprise language can be used to organize the synchronization requirements and policies in the analysis phase. The concepts and rules described in the ODP enterprise language are rather general and do not constrain the analysis of the synchronization service. We used the concept of ‘roles’, to identify the stakeholders involved, as well as the ‘prohibition’ and ‘policy’ concepts. We focus on the service provider, network provider and end-user roles.

The **service provider** offers the conferencing service and is responsible for receiving and broadcasting audio and video streams to the conferencing participants. The following synchronization requirements are identified:

- **intra-stream synchronization:** the service provider will manipulate incoming audio/video streams so that outgoing streams are within the 10 ms jitter boundary;
- **inter-stream synchronization:** the service provider will manipulate incoming audio/video streams so that related outgoing audio and video streams are within the $-20 \text{ ms}$ and $+40 \text{ ms}$ range [17];
- **spatial synchronization:** the service provider is responsible for ensuring that outgoing audio/video streams are played out simultaneously at the multiple users within the $0.25 \text{ s}$ boundary.

Policies to deal with spatial synchronization could be that new participants in a conference with a transmission delay not tolerable for other conferencing participants be prevented from participating in this conference, or be allowed to follow the conference in a limited way (e.g., only audio and no video).

The **network provider** provides the end-to-end transport facilities for audio and video streams. For audio/video synchronization strict QoS guarantees on throughput, maximum delay jitter, maximum delay and error rate should be provided. Based on these guarantees, the service provider and user can perform actions to
maintain synchronization. The network provider may apply different QoS policies regarding the transportation of streams [11]. Depending on the service it provides a:

- compulsory network service: a transportation service with a deterministic service is provided. This means that in the absence of total network failures, the QoS requirements of a conferencing client, such as maximum delay, throughput and jitter, are met. Fixed resource reservation must take place to be able to provide such a service;
- statistical reliable network service: a transportation service with a certain percentage of QoS violations is provided. Resource reservation is required, but now these resources can be shared with other users of this transportation service;
- best effort network service: the request from a client for a certain transport service is evaluated against the current network traffic. If load conditions change in future, the QoS provided to this client may change as a consequence.

A ‘compulsory’ or ‘statistically reliable’ service requires that a user or service provider should not exceed his specification of transport characteristics. If the agreed QoS can not be maintained the service provider or user should be informed.

Synchronization requirements from the end-user are basically determined by the limits of human perception [6]. With respect to the display of audio and video it is important that the following requirements to be met:

- lip-synchronization is a well known requirement and should be in the −20 ms to +40 ms range;
- audio or video jitter should be within the range of 10 ms;
- loss of video frames or audio samples is tolerable when less than 1% of the total sent;
- spatial synchronization should be in the range of −0.25 s to +0.25 s.

Similar service commitments can be defined for the end-user to maintain the above mentioned synchronization requirements:

- compulsory end-user service: in this case the synchronization requirements must be met. Reservation of processing and storage resources and admission control for other applications at the end-user’s node is required;
- statistical reliable end-user service: a certain percentage of violations of the synchronization requirements is allowed. Reservation of processing and storage resources and admission control is also required;
- best effort end-user service: possibilities to fulfil the synchronization requirements are based on current processing and storage activities. When, for example, other applications are executed on the same node, processing resources might not suffice, and the synchronization requirements will not be met any more.

User policies are often application dependent. For our multimedia conferencing service implementation, we applied the following user policies: if lip-synchronization is lost, video frames are dropped to get in line with audio. In case of congestion in the node, video frames will be dropped in favour of audio samples.

5. Definition phase

The definition phase provides a description of the desired effect of the synchronization service as seen by the stakeholders. Based on the requirements stated in the analysis phase, the specification of synchronization for multimedia conferencing services is performed using the TINA/ODP information language [20]. For the specification of synchronization it is necessary to have insight into the QoS characteristics of the involved stakeholders, as well as the relationships between stakeholders. Synchronization of streams is only possible when the QoS characteristics of these streams and supporting system/transport are known.

Figure 3 shows an Object Modelling Technique (OMT) graphical representation of an invariant schema for our problem domain. An invariant schema specifies the states and structure of an information object that are independent of any behaviour the object might exhibit [20]. The QoS attributes [14, 26] are used to describe the characteristics (e.g., coding, frame rate) of the sources and sinks (objects) used in a conferencing service, which in turn influence the synchronization relationships. The attributes relevant for audio and video synchronization are identified independently of how they should be implemented.

For synchronization it is important to characterize the involved media types using a list of QoS attributes. When these attributes are identified a contract can be specified between the stakeholders that should be respected for the conferencing service to operate properly. A contract provides a specification of the service and ‘level of service’ agreed between the involved stakeholders. The policy agreed can be classified as ‘compulsory’, ‘statistically reliable’ or ‘best effort’ as described in section 4. A contract description exists at different levels of abstraction ranging from end-user perspective to detailed system characteristics (see figure 4).

Figure 4 shows an invariant schema for a ‘user-service provider contract’ focusing on QoS attributes relevant for synchronization. The QoS attributes identified in the different perspectives have a close relationship as illustrated in the following example: a specification by the end-user for television quality will result in a required frame rate of 25 frames/s from the system perspective. This will in turn require a throughput (depending on the coding technique, picture size and colours) of an average of 2 mbit/s at transport perspective.

The values for QoS attributes may change during a conference session. This can be expressed by a static schema description [20]. A static schema specifies relevant QoS information at a certain point in time. This implies that for a specific conferencing session the QoS attributes...
M Jacobs and P Leydekkers

Figure 3. Invariant schema of QoS relevant for audio/video synchronization.

Table 1. Example of a static schema for audio-visual streams.

<table>
<thead>
<tr>
<th>QoS attributes</th>
<th>End-user values</th>
<th>System values</th>
<th>Transport values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Video</td>
<td>Audio</td>
<td>Video</td>
</tr>
<tr>
<td>Quality</td>
<td>television</td>
<td>telephone</td>
<td>0.02 $/kb</td>
</tr>
<tr>
<td>Costs</td>
<td>$ 2,— per minute</td>
<td>good</td>
<td></td>
</tr>
<tr>
<td>Audio/video synchronization</td>
<td>(MPEG, H.261)</td>
<td>μ-law</td>
<td></td>
</tr>
<tr>
<td>Coding</td>
<td>Throughput</td>
<td>Throughput</td>
<td>Throughput</td>
</tr>
<tr>
<td></td>
<td>25 frames/s</td>
<td>8000 samples/s</td>
<td>1.5 mbit/s</td>
</tr>
<tr>
<td>Error rate</td>
<td>$ 10^{-9}</td>
<td>$ 10^{-8}</td>
<td></td>
</tr>
<tr>
<td>Sample size</td>
<td>&lt; 8 kbyte</td>
<td>1 byte</td>
<td></td>
</tr>
<tr>
<td>Synchronization accuracy</td>
<td>high quality</td>
<td>intra-stream</td>
<td>±10 ms</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inter-stream</td>
<td>±80 ms</td>
</tr>
<tr>
<td>Streams</td>
<td>low</td>
<td>spatial</td>
<td>±250 ms</td>
</tr>
<tr>
<td></td>
<td>high</td>
<td>synchronization</td>
<td></td>
</tr>
<tr>
<td>Jitter bound</td>
<td>±10 ms</td>
<td>±10 ms</td>
<td></td>
</tr>
<tr>
<td>Delay</td>
<td>&lt; 250 ms</td>
<td>&lt; 250 ms</td>
<td></td>
</tr>
</tbody>
</table>

Table 1 shows an example static schema for a specific conferencing service contract. The contract is based on a 'best effort' agreement and applies to all QoS attributes. Values described from the end-user perspective will influence the system attribute values and transport attribute values.

A dynamic schema is a specification of behaviour in which a contract can participate subject to the constraints of an invariant schema. For example, in the service provider–user contract from system perspective a set of coding techniques are indicated for video streams (i.e., MPEG and H.261). In a 'best effort' agreement this may imply that in the case of problems, the user and service provider switch from MPEG coding to H.261 coding, which is less demanding for the systems and network.
Contract from different perspectives.

Interfaces: The conferencing server has to provide a set of operations with conferencing specific functionalities and information. Operations identified in this interface are invoked by a conferencing client (e.g. start/stop/join a conference);
- the exchange of audio-visual data;
- operations dealing with the management of the conferencing service. QoS negotiations between a client and server for the exchange of audio-visual data, as well as interactions with the network provider and third-party service providers are examples of this type of operations.

A management interface of a conferencing client must provide operations related to synchronization. Table 2 shows a partial TINA-ODL specification of a client management interface dealing with synchronization. The TINA Object Definition Language (TINA-ODL) [27] is a strict extension of OMG-IDL [22].

The QoS negotiation operation enables QoS (re-)negotiations for the exchanged audio-visual streams. As a result of these QoS negotiations, parameters for jitter, sample rate(s), sample size(s), and priority are known by a conferencing client, which is then able to apply intra- and inter-stream synchronization. The Synchronize operation is required for spatial synchronization.

The server must support requests from a client, or the network provider to re-negotiate a certain agreed QoS. The conferencing server supports Synch_Error and Synch_Info operations to control synchronization (see table 3). The first one enables a client to notify a server about synchronization quality problems. Monitoring is required to detect such problems. Spatial synchronization must be coordinated regularly when a conference proceeds. As a result of a ‘best-effort’ agreement between the user and service provider, the target playout rates may not be achieved and a conferencing client may fall behind. The coordination of spatial synchronization is achieved by synchronization feedback information contained in the Synch_Info operation. This enables the conferencing server to perform spatial synchronization, using either the feedback, global clock or synchronization channel technique. PlayoutRate and BufferOccupation attributes are required for the synchronization channel technique. The CurrentRate attribute identifies the last audio unit that is played out and is used by the feedback technique. Together with the CurrentTime attribute, indicating the global playout time of that last audio unit, spatial synchronization can be performed using a global clock.

The environment contract: The environment contract of computational objects includes statements derived from the requirements and policies described in the analysis phase. The environment contract concept is not developed in ODP-RM and TINA. We used it for the specification of the QoS properties of the computational objects involved in synchronization. QoS information consists of throughput, jitter and delay information. The environment contract for the conferencing server will supply each client with the necessary synchronization information, such as target delays.

The behaviour specification: The behaviour specification must include statements on how to maintain synchronization. The following statements are included informally for a conferencing client:

### 6. Specification phase

The specification lifecycle phase presents a detailed description of audio/video synchronization using the analysis and definition descriptions as input. We refined this phase by providing a computational and engineering specification. A computational description describes a distributed application in terms of computational objects (or program components) that interact in a transparent way. We focus on the interface descriptions since the interface is the only means to access a computational object. For synchronization operations, the operational interface is important to perform control operations.

An operational interface is characterized by a signature, a behaviour, and an environment contract:

- the operational interface signature describes a set of operations. An operation is similar to a procedure and is invoked on designated interfaces. An operation is an interaction between a client object and a server object that requests the execution of some function by the server;
- the environment contract describes a contract between the computational object and its environment, including QoS constraints, usage, and management constraints;
- the behaviour that occurs at the interface can be formulated by a dynamic interaction model. Formal description techniques such as LOTOS or SDL (used in this article) are useful to express the behaviour of an object.

**Interfaces:** The conferencing server has to provide a set of services [19]. According to [16], the operations provided by a multimedia conferencing service can be grouped into three categories:

- operations with conferencing specific functionalities and information. Operations identified in this interface
related audio and video streams are kept synchronized by speeding up the playout of video samples and restricted blocking;
• speeding up or slowing down audio playout is done by respectively skipping audio samples and playing audio samples multiple times;
• introducing spatial delays is done by slowing down or speeding up the playout of audio and video samples.

Figure 5 shows an SDL diagram illustrating the behaviour of a conferencing client.

The conferencing server (MMC-server) will exhibit the following behaviour related to synchronization: on successful QoS (re)negotiation it will adapt the values of (spatial) synchronization parameters. Distribution of these new values to conferencing clients is then required. When a client disconnects, the server will adapt the spatial synchronization. In the case of synchronization errors received from the network provider or conferencing client, the server will initiate QoS renegotiation. An SDL diagram illustrating this behaviour is shown in figure 6.

### Table 2. TINA-ODL specification of client management interface.

```c
interface template client_management;
typedef enum Streamtype { audio, video, voice, ... }
typedef enum Format { MPEG, JPEG, PCM, u-law, ... }
typedef struct SynchroQoS {
    StreamType Stream;
    format StreamFormat;
    integer StreamId;
    integer SampleSize, SampleRate, Priority;
    ...
};
typedef QoSParameters = sequence of SynchroQoS;
operations
    QoSnegotiate(in QoSoffer: QoSParameters; out Qosresult: QoSParameters);
    Synchronize (in SpatialDelay, FeedbackInterval: Integer);
    ...
behaviour
    'This interface enables QoS negotiations with a conferencing client. Spatial synchronization of streams will be performed according to the parameters supplied by the Synchronize operation'

```

### Table 3. TINA-ODL specification of server management interface.

```c
interface template server_management
typedef client (...); /* structure containing information about a conferencing client */
typedef struct SynchronizationInfo {
    integer PlayoutRate, BufferOccupation, CurrentStamp, CurrentTime
};
typedef QoS renegotiate(...);
Synch_Error (in client: ConfClient);
Synch_Info (in client: ConfClient; in SynchInfo: SynchronizationInfo);
...
behaviour
    'An instance of this interface will enable the conferencing server to coordinate synchronization of multiple clients'
```

**Figure 5.** SDL description of synchronization behaviour of a conferencing client.

**6.1. Engineering specification**

The ODP engineering language describes how to structure
the infrastructure. The engineering specification describes the functionality of Basic Engineering Objects (BEOs) and additional engineering objects needed to support distribution transparencies used by the application. A BEO is the executable representation of a computational object.

Figure 7 shows an engineering configuration of objects that play a role in audio/video synchronization. Many additional engineering objects are available in a computer node and required in a TINA-DPE kernel, but for multimedia applications additional functions should also be present (e.g., audio and video device objects). The TINA-DPE kernel provides a set of basic functions such as communication, storage and processing capabilities. It provides the mechanisms to solve distribution transparency and hides the infrastructure from the application designer.

All engineering objects in a node use the nucleus (not shown), which coordinates processing, storage and communication functions for use by other engineering objects within the computer node to which they belong. An operating system is an example of a nucleus. The thread manager offers the use of threads within an object. The thread manager spaws and forks threads. It joins, delays and synchronizes threads. The clock access and timer manager deals with a common distributed notion of time. Time service facilities may provide a common synchronized clock across a number of nodes and delivery of timer interrupts across the distributed system. The time service offers functionality to use clocks and timers.

The communication between BEOs is modelled by channels. The interactions occur via operational and stream interfaces. An operational or stream interface is mapped onto a configuration of stub, binder and protocol objects. For stream interfaces the stub performs compression and decompression functionalities, and in the future it is expected to perform also translation functions (e.g., translate MPEG to JPEG). The binder object in a stream channel is responsible for maintaining the integrity of the channel. A protocol object communicates with other protocol objects to achieve remote interaction. Cooperating protocol objects should at least provide an end-to-end transport service. The screen/camera device function is responsible for collecting and presenting video frames. It is an abstraction over a video camera and screen. The audio device function collects and delivers audio samples. It is a representation of a microphone and speaker.

**6.1.1. Synchronization object.** The computational conferencing client objects are mapped onto engineering conferencing client BEOs (e.g., MMC client in figure 7). The environment contract for the conferencing client contains a requirement for synchronization of incoming streams. Consequently, a synchronization object is added in the engineering specification between the original stream interfaces and the channel (see figure 7). The synchronization object is thus not part of a channel. The use of synchronization depends on the application. Thus, the application must perform synchronization according to its needs. Similar ideas can be found in [14], where it is stated that the transport system must guarantee bounds on delay and jitter and it is up to the upper layers to realize the synchronization according to their needs. In [15] it is also stated that the types of operations and corresponding behaviour of the synchronization object are application-dependent.

Synchronization outside the channel reduces the jitter in the streams introduced by the node. For example, decompression that occurs in the stub object will increment delay jitter. When performing synchronization after decompression, the synchronization accuracy will improve. A synchronization object can be regarded as a resource available on a node like audio and screen devices. A conferencing client can request the node manager for synchronization of streams.

No explicit control mechanisms are added between the source and sink for intra- and inter-stream synchronization since the mechanisms are applied to live media (e.g., pausing a stream for inter-stream synchronization is not a very useful operation). Second, extensions for multicast transmission are more easily added with a loose coupling between the source and sink [21]. Other advantages of this design choice are the reduction of the complexity of mechanisms and communication overhead.

**Intra-stream synchronization:** A node must provide functions to access clocks and timers. These timers can ensure a rate-synchronization between source and sink. A buffer-monitoring technique is applied to detect starvation or overflow of the buffer. The granularity of the clocks provided by the nucleus must be in the milliseconds range to perform accurate intra-stream synchronization.

**Inter-stream synchronization:** The use of threads and the synchronization of these threads for each audio and video stream results in a reliable implementation. Synchronization of threads can then easily be based on synchronization markers. The isochronity of digitized audio and video enhances the use of the synchronization marker technique. Different QoS requirements for each stream can be met by transporting these streams across different channels. Additionally, this is a relatively simple and accurate technique: every sample contains a marker holding an identification of the source and a logical time stamp. This type of synchronization has the advantage that data streams can maintain synchronization in a network with some delayed or lost packets. The time stamps can be used to detect and deal with these packets.

Synchronization of the threads requires real-time scheduling supported by a thread manager and pre-emptive...
scheduling to ensure delay and jitter bounds [13]. The use of synchronization markers has an important consequence: the functionality of the stub object needs to be extended since marshalling and unmarshalling of the sequence numbers is required (see stub extension object in figure 8).

Spatial synchronization: The design of a multimedia conferencing service accounts for a centralized manager coordinating spatial synchronization. The conferencing service sets up a central object that manages the synchronization process. This object coordinates the audio and video streams to ensure that they are synchronized. The conferencing service uses a synchronization object to manage the sequence numbers of the audio and video frames. The synchronization object ensures that the frames are delivered in the correct order, even if there is network delay or jitter.

The conferencing service uses a protocol to establish a connection between the sender and receiver. The protocol is responsible for sending and receiving data, and for handling errors that may occur during transmission. The protocol uses a stream interface to manage the flow of data between the sender and receiver. The stream interface ensures that the data is sent and received in a consistent manner, and that it arrives at the receiver in the correct order.

The conferencing service uses a spatial synchronization model to manage the spatial relationships between the audio and video streams. The model uses a spatial synchronization marker to indicate the position of each frame in the stream. The spatial synchronization marker is used to ensure that the frames are displayed in the correct order, and that they are synchronized with each other.

The conferencing service uses a clock and timer management object to manage the timing of the audio and video streams. The clock and timer management object ensures that the streams are synchronized with each other, and that they are displayed in the correct order.

The conferencing service uses a storage object to manage the storage of audio and video data. The storage object is used to store audio and video frames that have been received, and that have not yet been displayed. The storage object ensures that the frames are stored in a consistent manner, and that they can be accessed by the spatial synchronization object.

The conferencing service uses a protocol object to manage the communication between the sender and receiver. The protocol object is responsible for sending and receiving data, and for handling errors that may occur during transmission. The protocol object uses a stream interface to manage the flow of data between the sender and receiver. The stream interface ensures that the data is sent and received in a consistent manner, and that it arrives at the receiver in the correct order.

The conferencing service uses a spatial synchronization marker to indicate the position of each frame in the stream. The spatial synchronization marker is used to ensure that the frames are displayed in the correct order, and that they are synchronized with each other.

The conferencing service uses a clock and timer management object to manage the timing of the audio and video streams. The clock and timer management object ensures that the streams are synchronized with each other, and that they are displayed in the correct order.

The conferencing service uses a storage object to manage the storage of audio and video data. The storage object is used to store audio and video frames that have been received, and that have not yet been displayed. The storage object ensures that the frames are stored in a consistent manner, and that they can be accessed by the spatial synchronization object.

The conferencing service uses a protocol object to manage the communication between the sender and receiver. The protocol object is responsible for sending and receiving data, and for handling errors that may occur during transmission. The protocol object uses a stream interface to manage the flow of data between the sender and receiver. The stream interface ensures that the data is sent and received in a consistent manner, and that it arrives at the receiver in the correct order.

The conferencing service uses a spatial synchronization marker to indicate the position of each frame in the stream. The spatial synchronization marker is used to ensure that the frames are displayed in the correct order, and that they are synchronized with each other.

The conferencing service uses a clock and timer management object to manage the timing of the audio and video streams. The clock and timer management object ensures that the streams are synchronized with each other, and that they are displayed in the correct order.

The conferencing service uses a storage object to manage the storage of audio and video data. The storage object is used to store audio and video frames that have been received, and that have not yet been displayed. The storage object ensures that the frames are stored in a consistent manner, and that they can be accessed by the spatial synchronization object.

The conferencing service uses a protocol object to manage the communication between the sender and receiver. The protocol object is responsible for sending and receiving data, and for handling errors that may occur during transmission. The protocol object uses a stream interface to manage the flow of data between the sender and receiver. The stream interface ensures that the data is sent and received in a consistent manner, and that it arrives at the receiver in the correct order.
server should support the following spatial synchronization techniques:

- the feedback technique [4] where the sink periodically transmits feedback messages to the source, containing the stream unit number that was currently played. Due to these feedback messages and the known bound delay on the network, the source can estimate the actual playback times of stream units at the sink and adjust the stream transmission rate;
- the global clock technique [7] where each stream unit contains a reference time when it should be played out at the sink. This reference time is calculated at the source of a stream unit using the global time and adding the delay and jitter across the connection;
- the synchronization channel for which a synchronization coordinating node is created. Information about the presentation of the streams, such as playout rates, is periodically transmitted across separate synchronization channels between the coordinating node and the source and sink nodes.

If a global clock exists between conferencing clients, spatial synchronization will be coordinated using that clock. Otherwise, the feedback technique will be applied, unless delays for remote operations are not known precisely. In this case, the synchronization channel technique will be used with the evaluation of playout rates of the conferencing clients.

7. Development phase

The development stage in the TINA lifecycle model consists of the development of hardware and software for this synchronization service. We used ANSAware for the realization of the multimedia conferencing service. ANSAware is an implementation of the Advanced Networked Systems Architecture (ANSA) [25], which was one of the major inputs for ODP-RM and the TINA architecture.

We extended ANSAware to support stream interfaces as reported in [23, 24]. Based on this extension the synchronization object is added. For the end-to-end communication the Unreliable Datagram Protocol (UDP) was chosen as the protocol for stream communication. Very reliable communication is not required for audio and video data, although a mechanism was added to detect and replace lost packets (this is necessary for intra-stream synchronization).

Figure 8 shows the engineering configuration for conferencing clients realized on ANSAware. The use of shared buffers is adopted for synchronization as described in [10] and [12]. This eliminates the need for synchronous channel/application interaction. At the source, the parallax video board grabs images from the video camera and stores the samples into the storage object. The audio grabber operates in a similar way: the ‘grabbing rate’ is determined by the synchronization object, which uses the clock and scheduling functions, and adds a sequence number to the sample. The synchronization object determines the appropriate moment when the samples are offered to the channel object for marshalling and transmission.

At the sink, incoming samples are unmarshalled and stored in the storage object (in our case a shared buffer for audio and video samples). The screen device and audio device retrieve samples from the shared buffers. The retrieval rate is again determined by the synchronization object. The sequence numbers are used to sort the audio and video samples.

Intra-stream synchronization is achieved by transmitting samples at the source with a constant rate. At the sink the received data are placed in a buffer. The synchronization object maintains synchronization by sending the content of these buffers to the devices at specific rates. The rates are maintained using the internal clocks of the SUN Sparc station since the granularity of ANSAware 4.0 clocks was not accurate enough.

Inter-stream synchronization is coordinated choosing a master sink synchronizer and slave sink synchronizer (audio samples are prior to video samples). The slave follows the master and synchronization between master and slave is achieved based on the sequence numbers. Since ANSAware 4.0 does not support pre-emptive scheduling it is performed by the application. In the case of problems, the master and slave sequence numbers are kept related by dropping slave stream units first.

Spatial synchronization is maintained by introducing the spatial delay prescribed by the synchronization manager of the conferencing server. Introducing this delay dynamically can be done by adapting playout rates for a short period. The technique applied by the synchronization manager of the conferencing server must be selected in advance.

8. Conclusion

The state-of-the-art regarding synchronization showed that mechanisms based on global clocks are a very accurate way to perform synchronization. Currently, an accurate global time might not be available in the distributed computer environment, but we believe that in the future, the computing nodes constituting this environment will have the ability to adopt a certain global time. If this is the case, the synchronization mechanisms should be based on this global time.

We provided an engineering specification of a synchronization object that can be used as a basic service to provide audio/video synchronization. We support the view that a synchronization object should become a basic component of the TINA-DPE kernel that supports real-time multimedia applications. This will enable designing applications from a higher abstraction level. A (multimedia) application can achieve synchronized streams by requesting this engineering object to perform this task. The application design will abstract from the issue of how synchronization is achieved.

The TINA framework requires a number of additional engineering objects to be available in the TINA-DPE kernel, such as a thread manager and a clock access and timer manager. Requirements to be put on these objects.
should be investigated. If the TINA-DPE wants to provide possibilities for the exchange of real-time data, thread managers residing in these TINA-DPE kernels must support real-time scheduling. Similar requirements can be identified for a clock manager to use high resolution clocks to support accurate global time.

We used part of the TINA lifecycle model as a development method for the specification and implementation of a synchronization service for multimedia conferencing services. Using this model we identified the need to detail it using the ODP viewpoint languages. The more expressive ODP-RM viewpoint languages in the various phases of the construction lifecycle gave us sufficient support to design the synchronization service. Use of this methodology provides a more complete specification which can then be implemented on a TINA-DPE such as ANSAware.

However, both the lifecycle model and the ODP-RM viewpoint languages were designed to describe any arbitrary (telecommunication) service and are consequently very generic. Also detailed guidelines and rules are not described to assist the designer using the TINA lifecycle model. For example, which rules has a designer to follow in order to map an information model (definition phase) onto a computational model (specification phase)? Or, how does the designer know that two (viewpoint) specifications are consistent with each other? These, and other issues need still to be resolved to make the TINA lifecycle model a more powerful methodology to specify distributed telecommunication services.

Experience has to be obtained in the TINA consortium to apply this methodology to large scale pilots, however, the results we obtained by applying it to a synchronization service have given us some confidence that this methodology is a useful tool to describe complex distributed services.

References

[1] TINA core team 1993 Definition of Service Architecture Tina-C deliverable, TB_BMDC.012_I_0.93
[24] van Halteren A T 1994 Realisation of a multimedia stream object in ANSAware Masters thesis University of Twente
[27] TINA core team 1994 Information modeling concepts TINA-C, Document No. TB_EAC.001_3_0.94
[28] TINA core team 1995 Computational modelling concepts TINA-C, Document No. TB_A2.HC.012_3_0.95
[29] TINA core team 1995 Engineering modelling concepts TINA-C deliverable, TB_A3.NS.005_I_0.93