Dependability Experiments with P2P Networks

Elias P. Duarte Jr.

http://www.inf.ufpr.br/elias
Federal University of Parana
Curitiba, Brazil

IFIP WG 10.4 53rd Meeting Natal Feb/2008

Outline

- GigaMan P2P: A P2P Network Management Architecture
- Streaming POTS (Peer Overlay for Transporting Streams)
- JXTA Peer Groups for Content Distribution
  - *projects that involve P2P Networking & Dependability*

P2P Dependability Experiments

GigaManP2P

Optical Networks & QoS Routing

- Optical networks are increasingly popular for implementing high speed backbones
- Applications often have QoS (Quality of Service) requirements
- Routes are selected based on those requirements
- Dynamic process, that involves:
  - Route monitoring, often across Autonomous Systems
  - Transparent rerouting when needed

P2P Networks

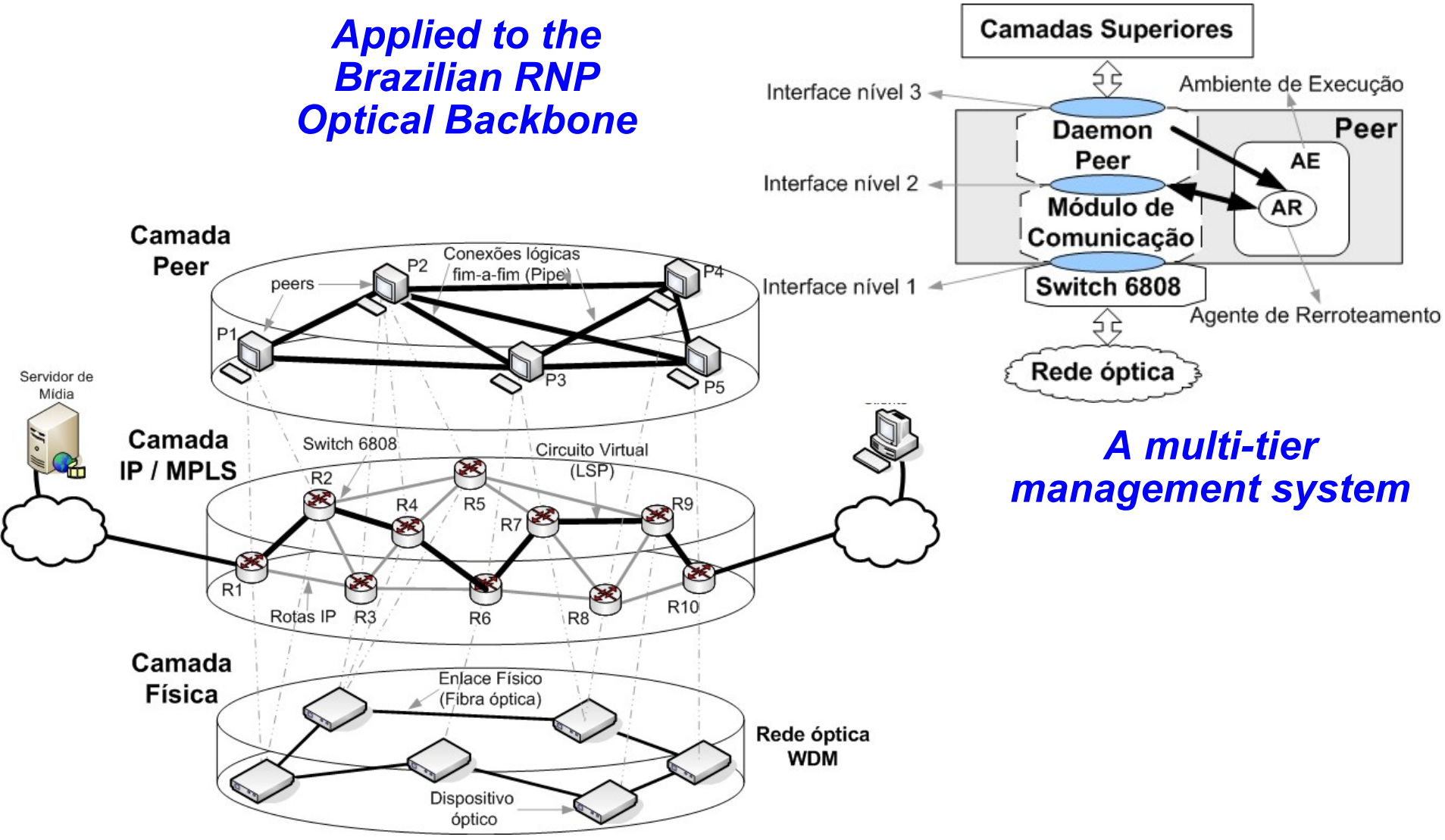
- Our point of view:
  - Nodes are processes running at hosts
    - Unstructured, semi-structured, structured
    - Search facilities are available, based on some DHT (Distributed Hash Table)
  - Peers may be structured into a hierarchy
    - Usual applications include
      - Content distribution
      - Remote task execution

A P2P Management Architecture

- GigaManP2P is a management system based on the P2P paradigm
- Peers are network management entities distributed across multiple Autonomos Domains
  - A network overlay for management, on top of the IP/MPLS Layer
- Several management applications are provided including: network monitoring, policy-based configuration management, QoS circuit monitoring and re-routing...

GigaManP2P: Architecture

Applied to the Brazilian RNP Optical Backbone



A multi-tier management system

GigaManP2P: Rerouting

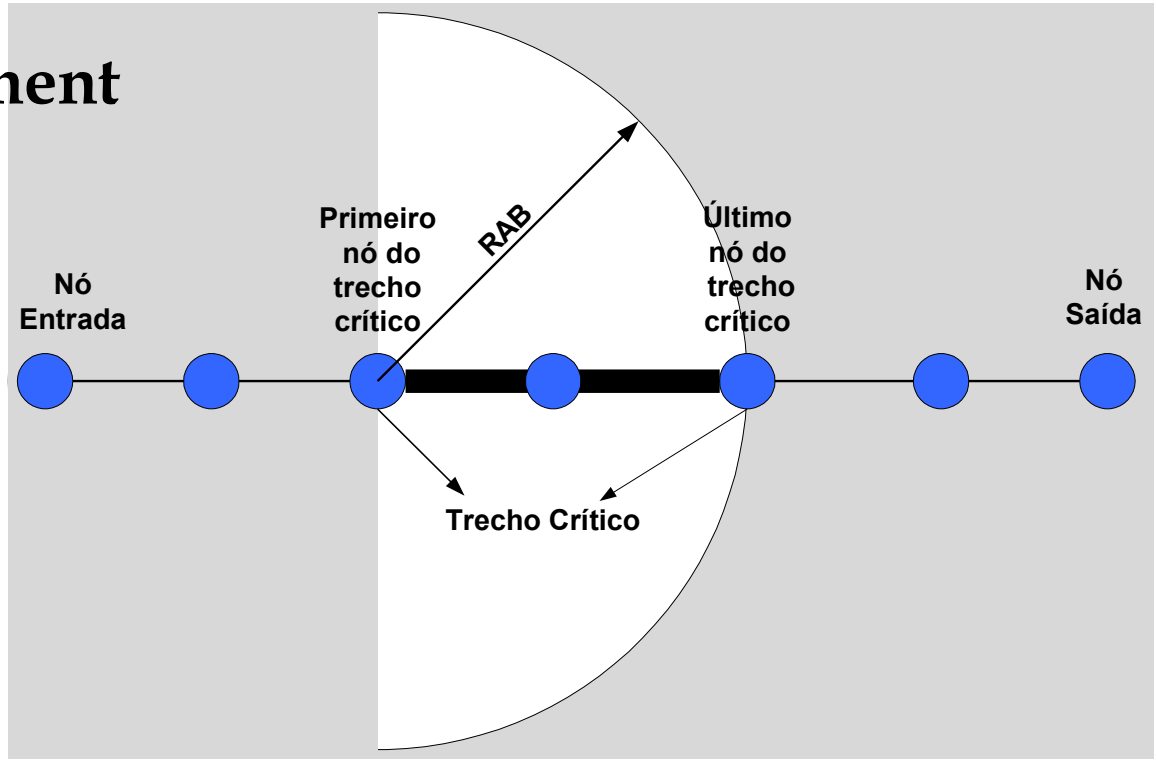
- Integrates QoS monitoring & rerouting
  - The main purpose is to avoid/reduce as much as possible QoS faults, so that circuits are assigned to applications according to QoS requirements
- Employs are pro-active rerouting strategy
  - *Traps (alarms)* are generated whenever there is a possibility of a QoS fault occurring
  - Alternative circuits are then discovered and monitored
  - Rerouting is issued as early as possible

Partial/Full Rerouting

Rerouting may be either:

- **Full: the complete circuit is replaced**
- **Partial**
  - Only a critical segment

Critical Segment Example



Rerouting: Phases

- Phase 1: Virtual Circuit is established
- Phase 2: Monitoring agents are started
- Phase 3: Alternative routes/segments are found
- Phase 4: Monitoring agents are started

Phase 5: Reroute

Agents include traditional network management entities and mobile agents that migrate in order to discover and monitor circuits

Experimental Results

- **Two main type of experiments were executed**
  - **Simulation experiments showing the feasibility of the proposed approach in terms of**
    - **Different network topologies**
    - **Varying network connectivity**
    - **Traffic type and intensity**
  - **Evaluate the overhead of the proposed P2P management structure on the real optical devices**

    Especially the latency for executing management actions

GigaManP2P: Final Remarks

- We are still working on this project
- A prototype is currently running on top of AdventNet 6808 optical switches of RNP backbone
- Papers in Portuguese were published at Brazilian national conferences/workshops
- One international conference paper in English (IEEE/IFIP ITC) preliminary results

P2P Dependability Experiments

Streaming POTS

Streaming POTS

- A P2P system to assist the distribution of multimedia streams from a given source
- A source (multimedia server) sends continuously a stream to [a pontentially large number of] clients
- This system does not scale well: the server bandwidth is limited
- Consider two clients "closer" to each other than they are to the server
  - *It might be better for one to obtain information from the other than from the server*

Streaming POTS

- Several P2P networks have been deployed for content distribution
- If content is organized in FILES
  - *The information flow completes as soon as the file is obtained*
  - *Size is fixed and known*
- In case of multimedia streaming (e.g. radio)
  - *Information is produced continuously, endlessly*
  - *There are deadlines for delivering packets*

Streaming POTS: Client & Server

- Not a "pure" P2P system: there is still a server
  - *Generates the stream content*
  - *Stores the stream in a buffer*
  - *Divides the buffer in parts*
  - *Sends the parts to SOME clients*

- The remaining clients establish agreements in order to receive other parts

Streaming POTS: Client & Server

- The Server
  - *Helps clients to find each other*
  - *Generates the stream*
  - *Sends parts of the stream to clients*
  - *Is "always there" i.e. a client can always obtain the stream for the source*
- Clients
  - *Receive and reproduce the stream*
  - *Establish agreements and send parts of the stream to other clients*

Agreements

- The stream is divided in fixed *slices*
- An *slice* is divided in *blocks*
- An agreement refers to a given block number, and the *length of an agreement* is the number of slices
- An important parameter:
  - ***Number of block copies that must be sent by the server to clients***
- The metric for selecting clients and peers from which to obtain the stream: *round-trip-time (periodically monitored)*

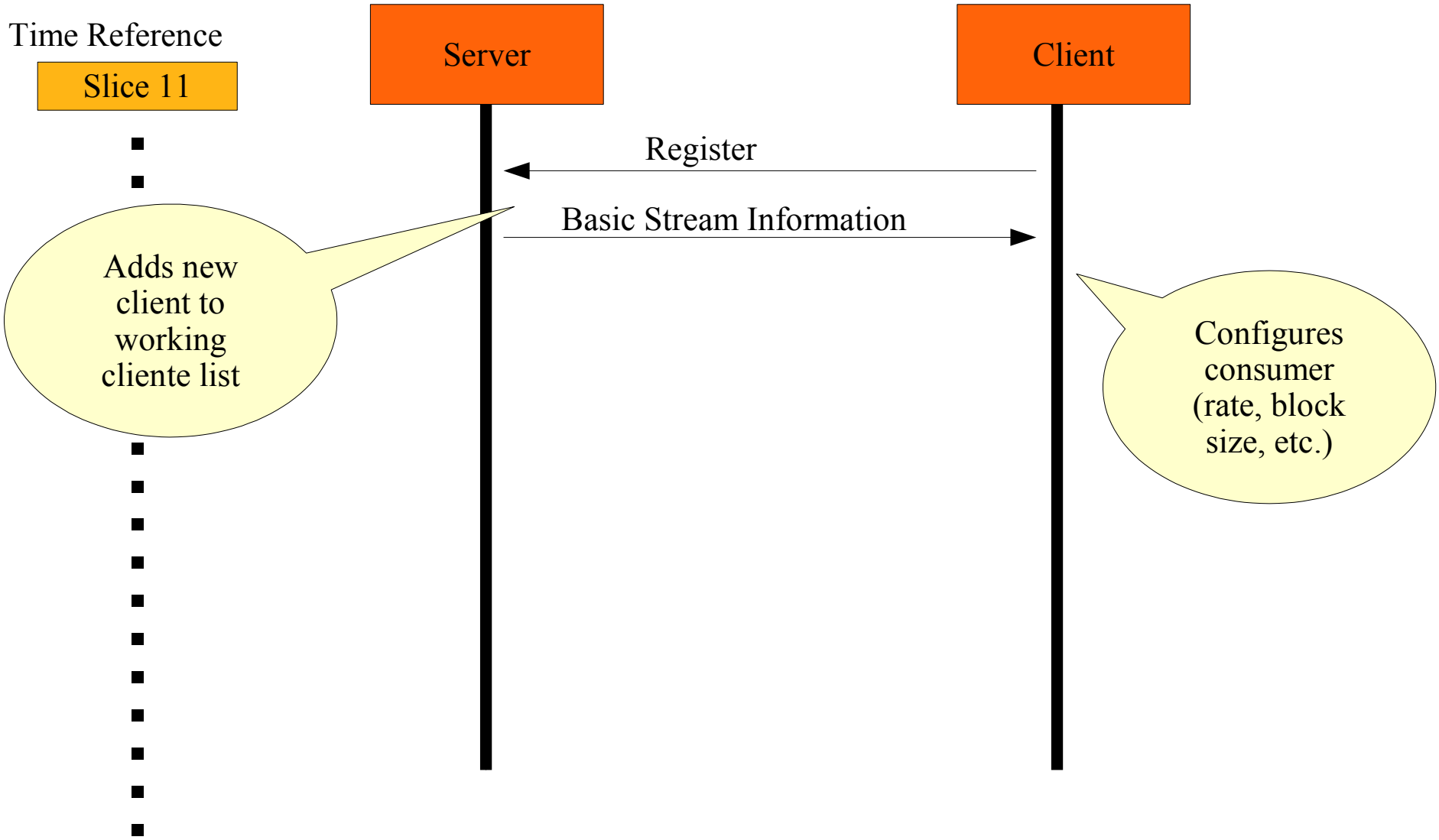
Blocks, Slices and Agreements

- **Example**

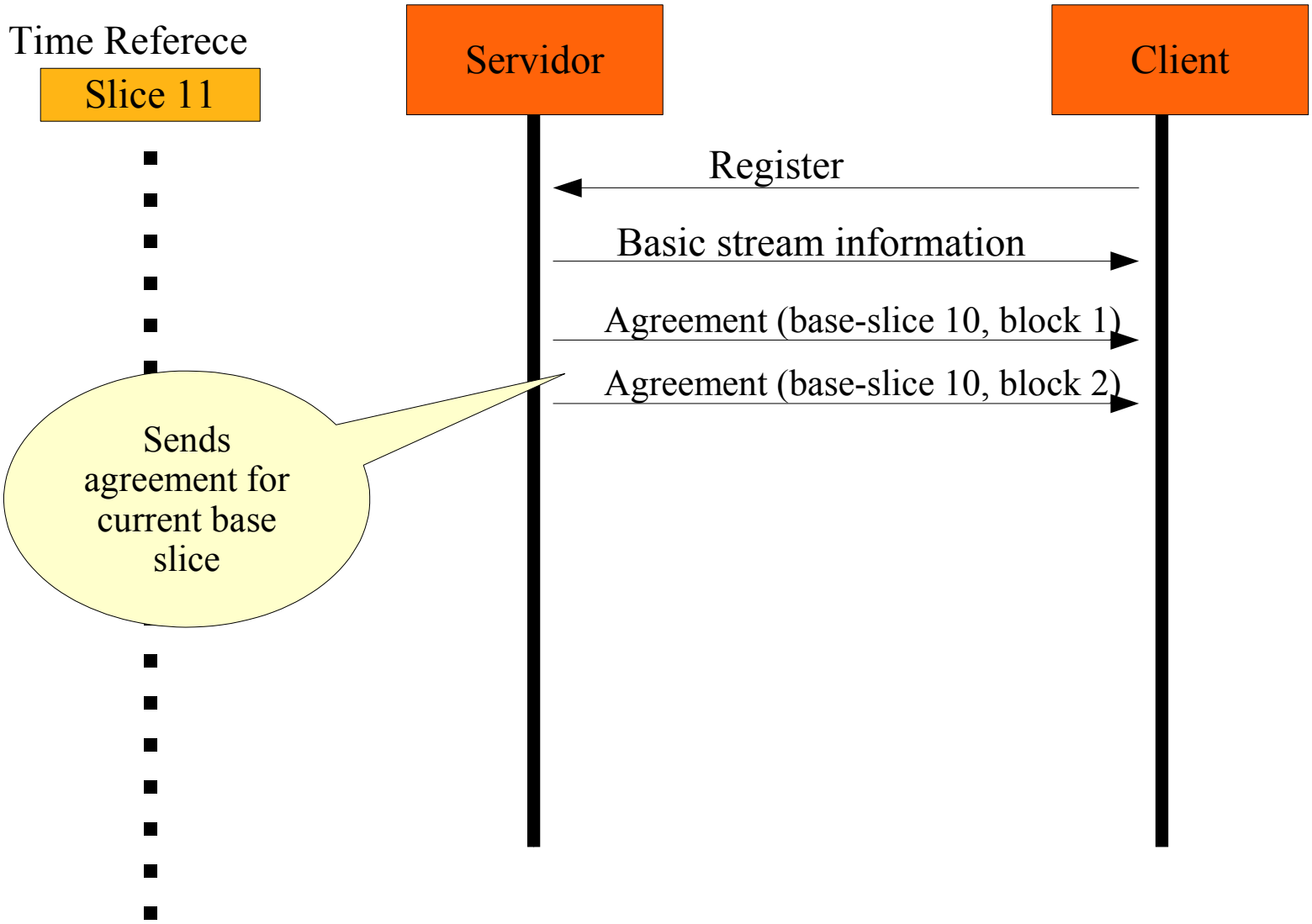
Source: X
Block #: 3
Base Slice: 20

Length of the agreement: 10 slices

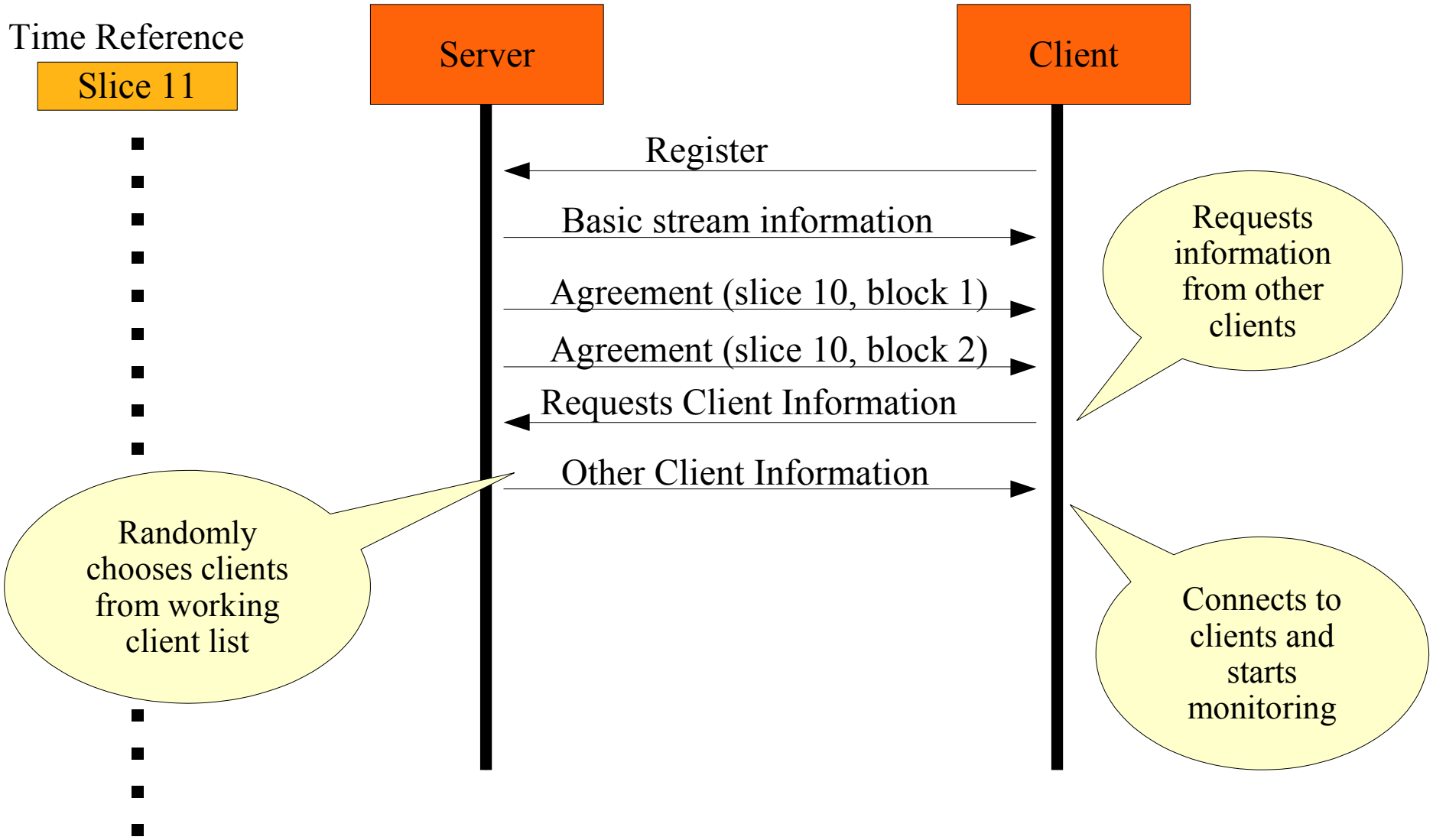
Client Initialization



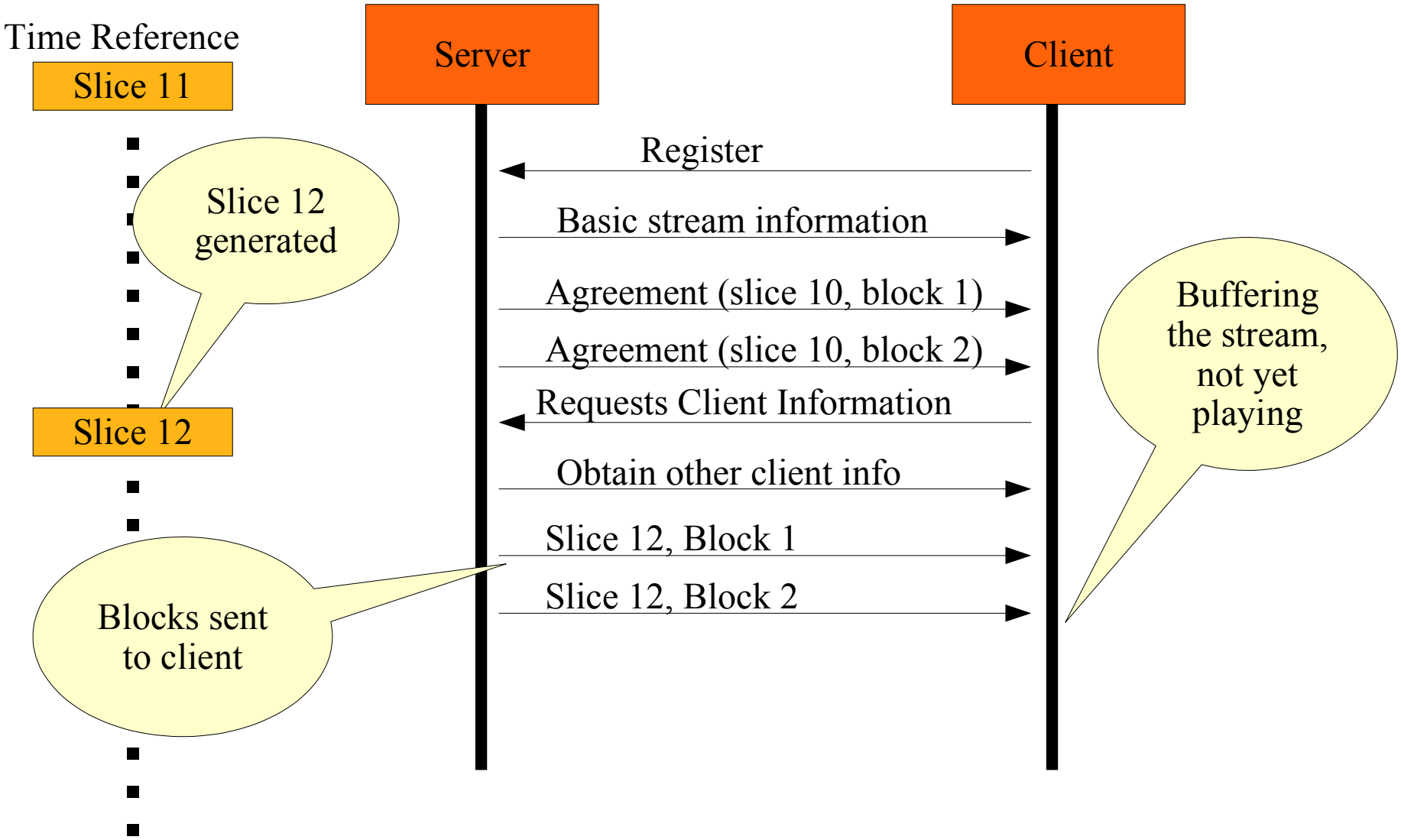
Client Initialization



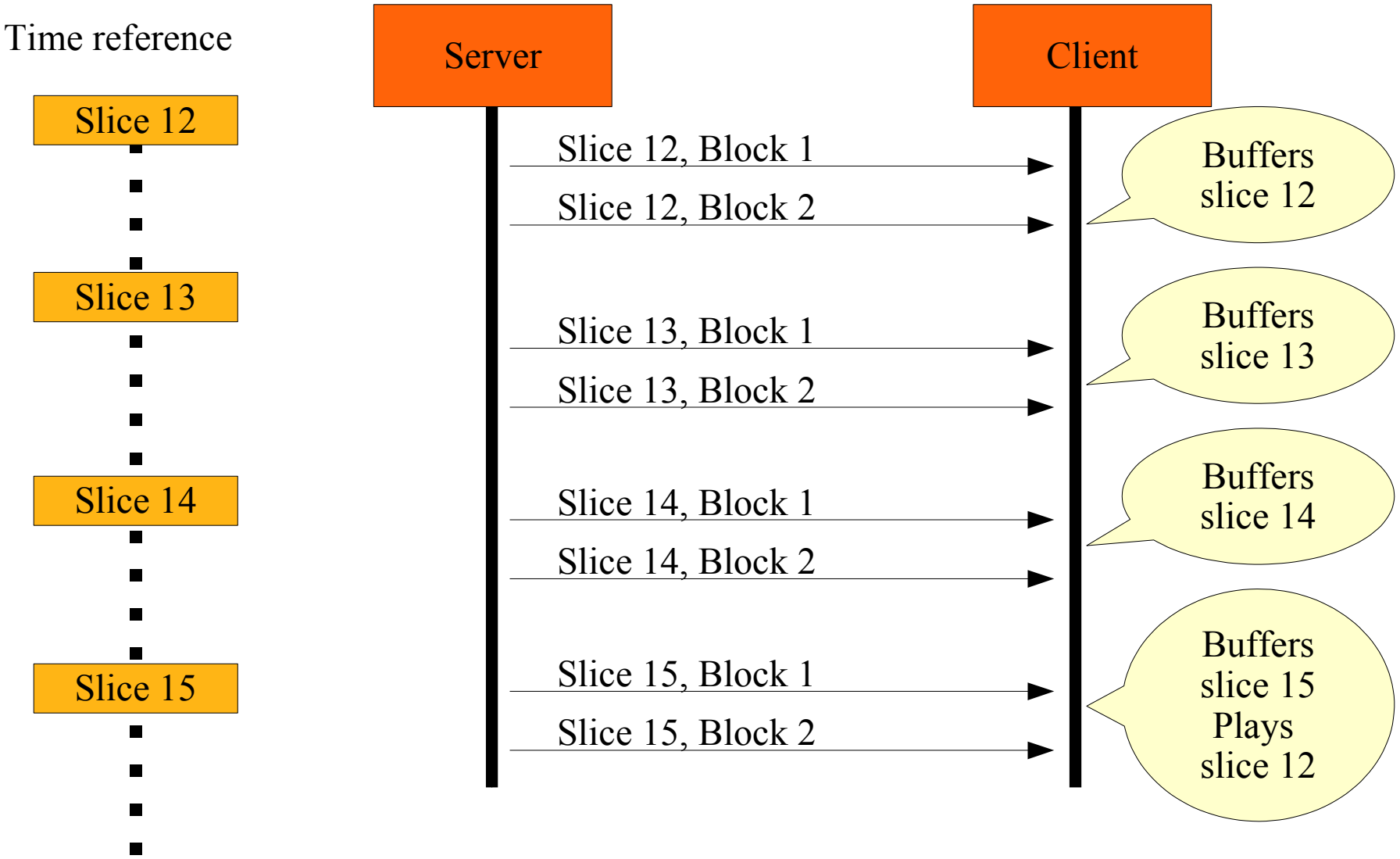
Client Initialization



Client Initialization



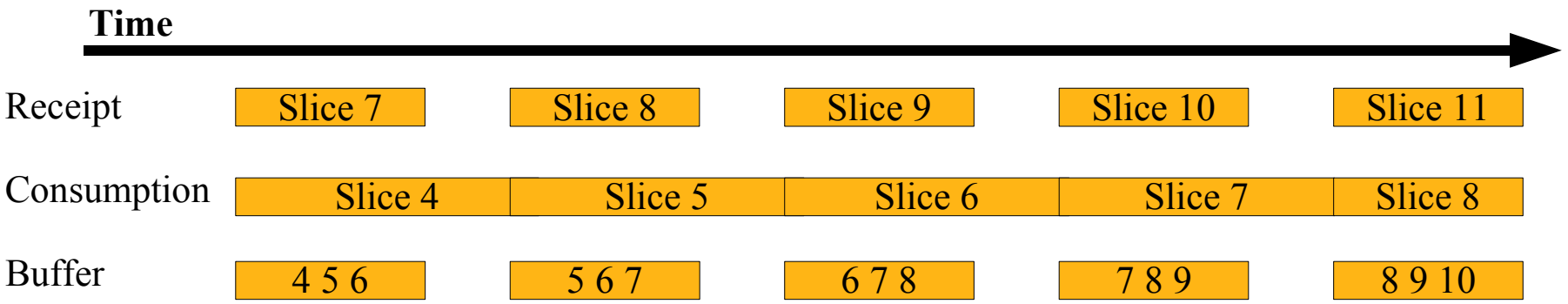
Client Initialization



Streaming POTS

- Not for real time systems: clients must wait before the stream is reproduced
- Blocks are collected from the server and other clients
- The wait interval must be long enough for the client to
  - *Receive blocks from established agreements*
  - *Obtain missing blocks*

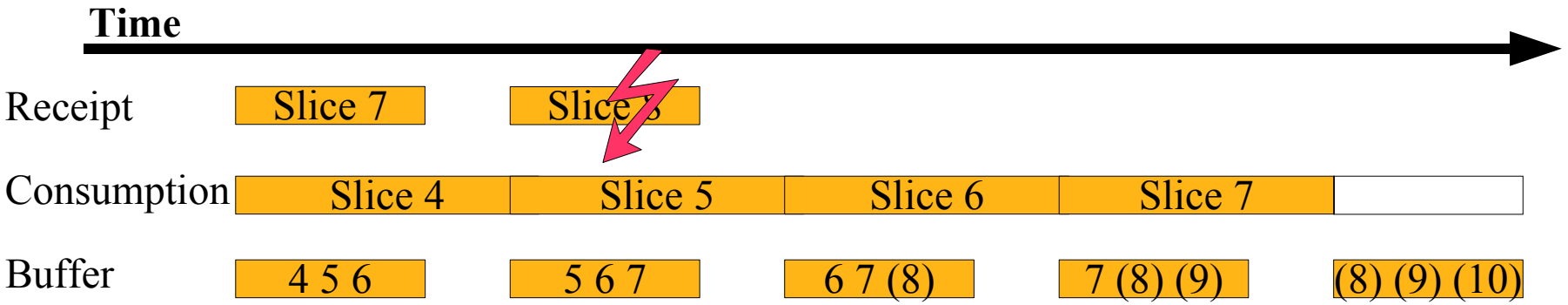
Tolerating Missing Blocks



- **After slices are received an interval elapses before they are consumed**

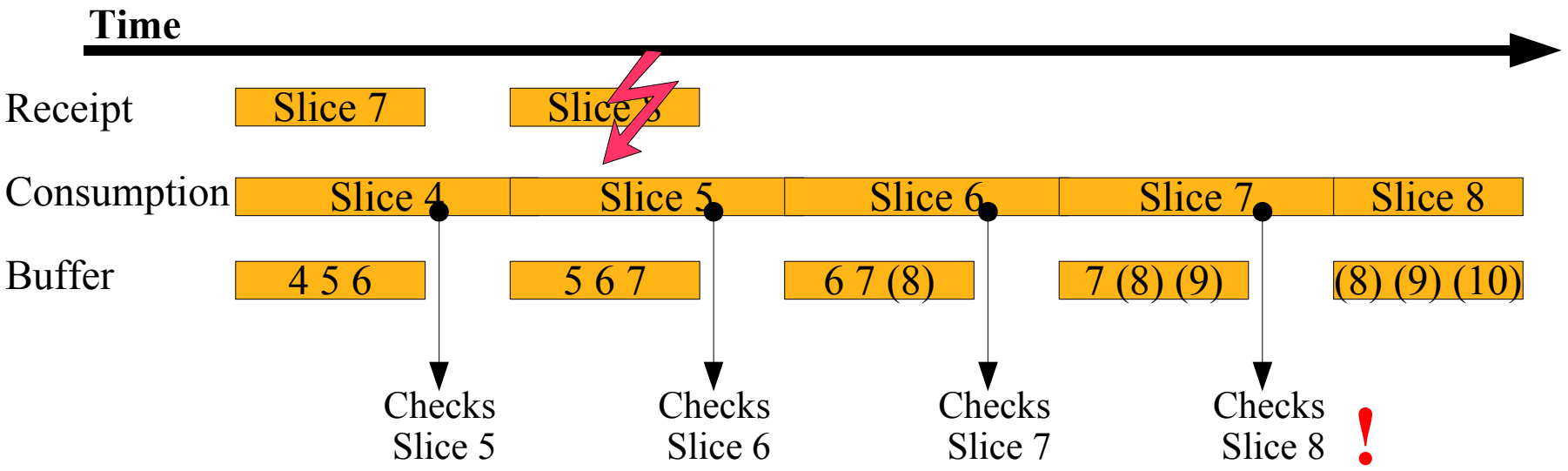
- **Slices are buffered until they are consumed**

Tolerating Missing Blocks



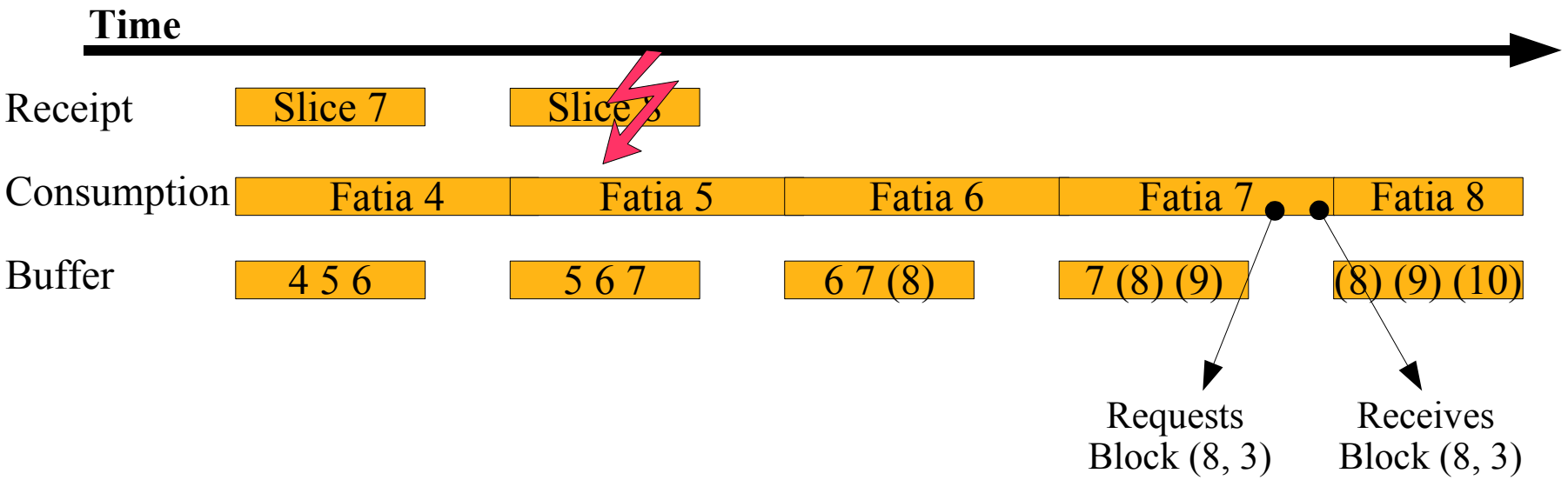
- **Slices may be missing due to *several* reasons**

Tolerating Missing Blocks



- **Before a given slice is consumed, a completeness check is performed**
- **If any block is missing, a request is issued to the source server**

Tolerating Missing Blocks



- **A request is issued for the absent block needed, not for the whole slice**

Streaming POTS Final Remarks

- If a peer does not send an agreed block: the agreement is broken
- We have a working Java implementation: transport protocol TCP (UDP was no good!)
- A paper was submitted to a Brazilian national conference, not submitted to international venues yet

JXTA Peer Content Groups

JXTA Peer Content Groups

- In usual P2P content distribution networks: the client gets a list of peers from which content can be obtained
- The client (either the process or the user herself) selects a peer from which to download the file
- Peers are highly dynamic, at any time this source peer can leave the system
- In order to avoid the selection process, we propose the construction of *Content Groups*

Peer Group Service

- This project has the main purpose of making the source selection transparent for the user
- Transparency: the way a user accesses a single peer is identical to accessing a *content group*
- We employ group membership concepts to this highly dynamic environment but strong consistency is not guaranteed!
- Implemented as a middleware:
  - *Application <-> Middleware <-> P2P Network*

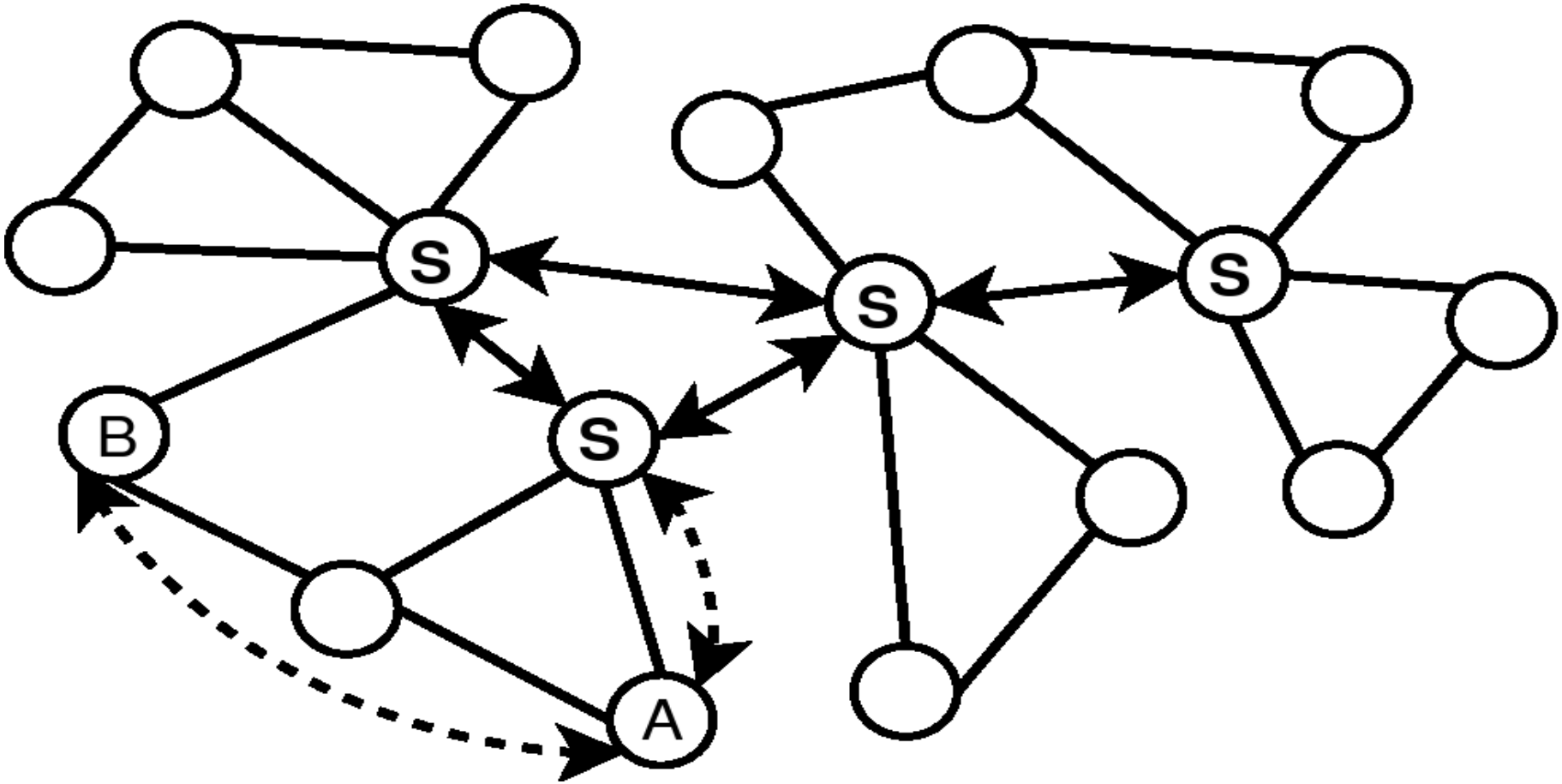
JXTA Peer Content Group: System Model

- Logical P2P network
- Fully connected (complete graph)
- Asynchonous system
- Fault model: crash with repair
- We assume reliable communication channels

JXTA

- The system was implemented within the JXTA Platform (www.jxta.org)
- JXTA provides several facilities for implementing P2P systems & applications
- Employs both HTTP and regular transport protocols (TCP/UDP)
- Offers services such as relays (to bypass firewalls & NAT) and rendezvous (network directories)
- Open source

Semi-Structured P2P Network



Peer Group Service

- Maintains peer content groups
- Dictates how peers interact to form groups
- Allows peers to join/leave the system
- Keeps a view of fault-free members
- Interface: functions for transmitting messages & obtaining a group view

JXTA Peer Content Group: Views

- A view consists of a list of working peers sorted by peer identifier
- Each peer stores TWO views
- **Local view**: used by the peer to update information on peer state
- **External view**: the last group view for which a partial agreement was established

Peer Monitoring & View Construction

- At predefined time intervals
  - *Each peer checks whether other peers left/joined the system*
  - *Sends new event information to other peers*
- Messages carry the local view of the sender

Peer Monitoring and View Construction

- As a peer receives messages from another peer:
  - *The source peer is inserted in the local view*
  - *If this is new information: message is sent to the group*
- As a peer receives a message carrying a vision without itself:
  - *"Leaves" the system and reinitializes*
- Initialization: peer sends join message to the group
  - *Waits for responses: updates local view with received view info*

A File Sharing Application

- Allows files to be downloaded from groups of peers
- Based on JXTA
- Transparent: downloading from an individual peer is identical to dowloading from a content group (from the client perspective)
- Client does need a "front-end"
- Each group keeps 1 file
- Peers can participate in as many groups as the number of files they have

Client Algorithm

- Searches for a file (e.g. non-copyrighted music name)
- The client does not connect to the group...
- ... just sends a download request
- Receives information about the communicantion channel to open
- Receives the file
- Frontend sends a “completed” message to the group, otherwise requests missing parts

Server Algorithm

- Searches for a file group (using JXTA facilities)
- If the group is found: connect to the group
- Otherwise: create the group
- Listen and waits for requests
- Request received: run leader election
- Leader sends the file

JXTA Peer Content Groups: Final Remarks

- We have a working prototype
- One paper was published at the Brazilian Workshop on P2P Computing (in Portuguese)

Conclusions

- A *very* brief overview of 3 research projects involving P2P Networks & Dependability
  - QoS Rerouting on optical networks with a GigaManP2P
  - JXTA Peer Content Groups
  - Streaming POTS (Peer Overlay for Transporting Streams)