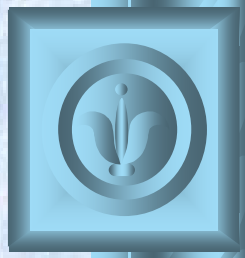


# فصل چهارم: ارتباطات

## سیستمهای توزیع شده

امیر مسعود رحمانی

# فهرست مطالب

مقدمه: نگاه کلی به سیستم عامل

سیستمهای چندپردازنده ای

فصل اول: مقدمه ای بر سیستمهای توزیع شده

فصل دوم: معماری ها

فصل سوم: فرایندها

فصل چهارم: ارتباطات

فصل پنجم: نامگذاری

# ارتباطات

**ارتباطات بین فرآیندها یکی از مهمترین مسائل در سیستم های توزیع شده است.**

**ارتباطات بین فرآیندها قلب یک سیستم های توزیع شده است.**

**سه مدل ارتباطی پر استفاده**

فراخوانی رویه راه دور(RPC)
میان افزار پیام گرا (MOM)
جریان داده ها

# انواع پروتکل ها

**۱- اتصال گرا**

قبل از مبادله داده ها، باید بین فرستنده و گیرنده اتصال برقرار شود.

تلفن، سیستم ارتباطی اتصال گرا است.

**۲- بدون اتصال (غیر اتصال گرا)**

لازم نیست از قبل اتصال برقرار شود.

قرار دادن نامه در صندوق پستی نمونه ای از ارتباط بی اتصال است

# تفاوتهای اتصال گرا و بدون اتصال

۱- در اتصال گرا برای بسته های به هم مرتبط تنها یکبار مسیریابی می شود ولی در بدون اتصال برای هر بسته مسیر یابی می شود.

۲- در اتصال گرا بسته ها به ترتیب به مقصد می رسند ولی در بدون اتصال بدون ترتیب می رسند.

۳- در اتصال گرا چون بسته ها به ترتیب می روند فقط بسته اول اطلاعات کامل درباره مقصد دارد و بقیه بسته ها فقط شناسه مورد نظـر را دارد ولی در بدون اتصال همه بسته ها اطلاعات کامل درباره مقصد دارند.

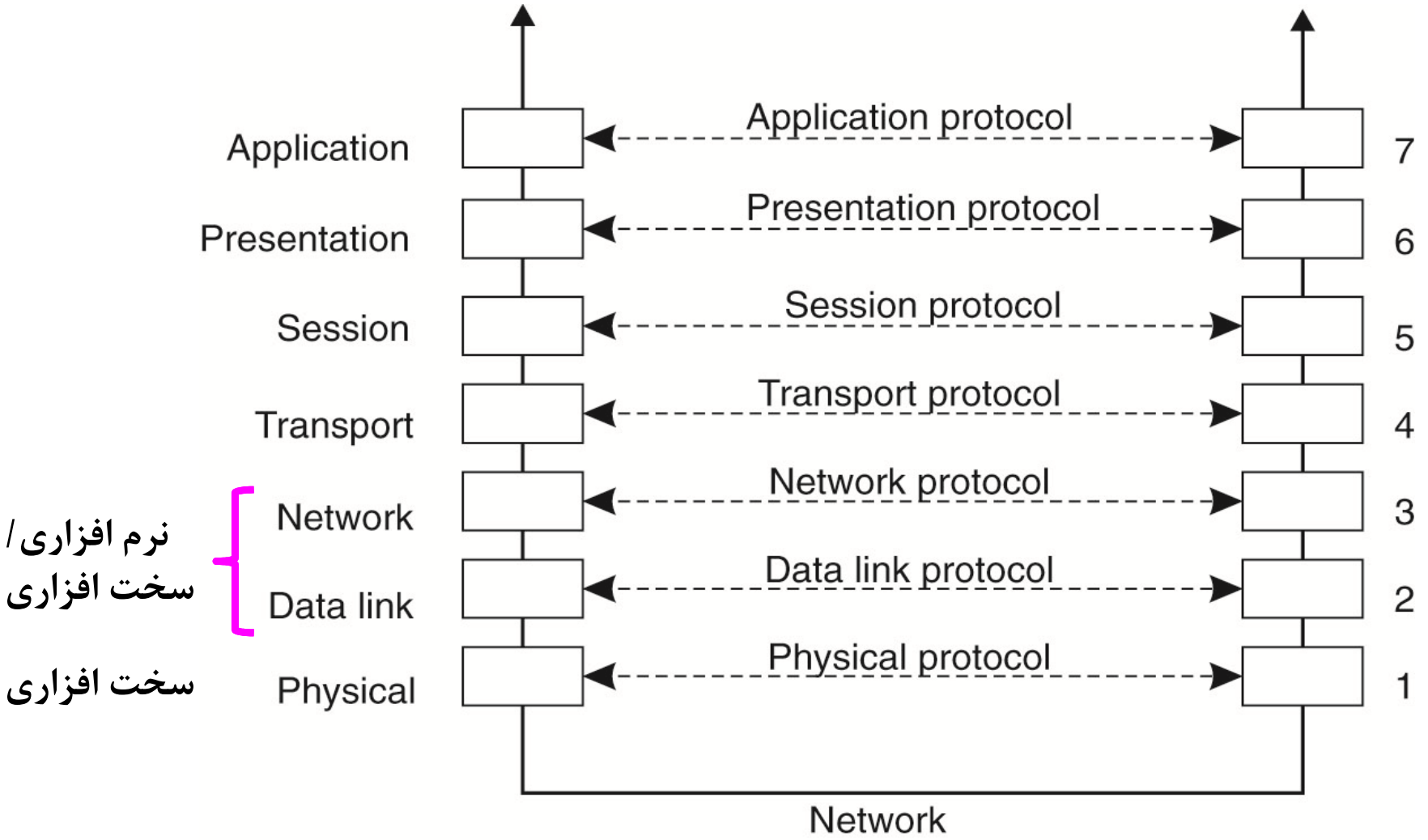
۴- در اتصال گرا چون مسیر بسته ها را می دانیم کنتـرل ترافیـک بهتـر است.

# معایب اتصال گرا

۱- در برابر تغییرات ناگهانی ترافیک شبکه ضعیف است و در صورت وجود ترافیک شدید امکان تغییر مسیر وجود ندارد.

۲- اگر بدلیلی مسیر ارتباطی قطع شود، هیچ راهی جز اینکه دوباره بسته ها را از اول بفرستد، وجود ندارد

# لایه ها، واسط ها، و پروتکل ها در مدل OSI



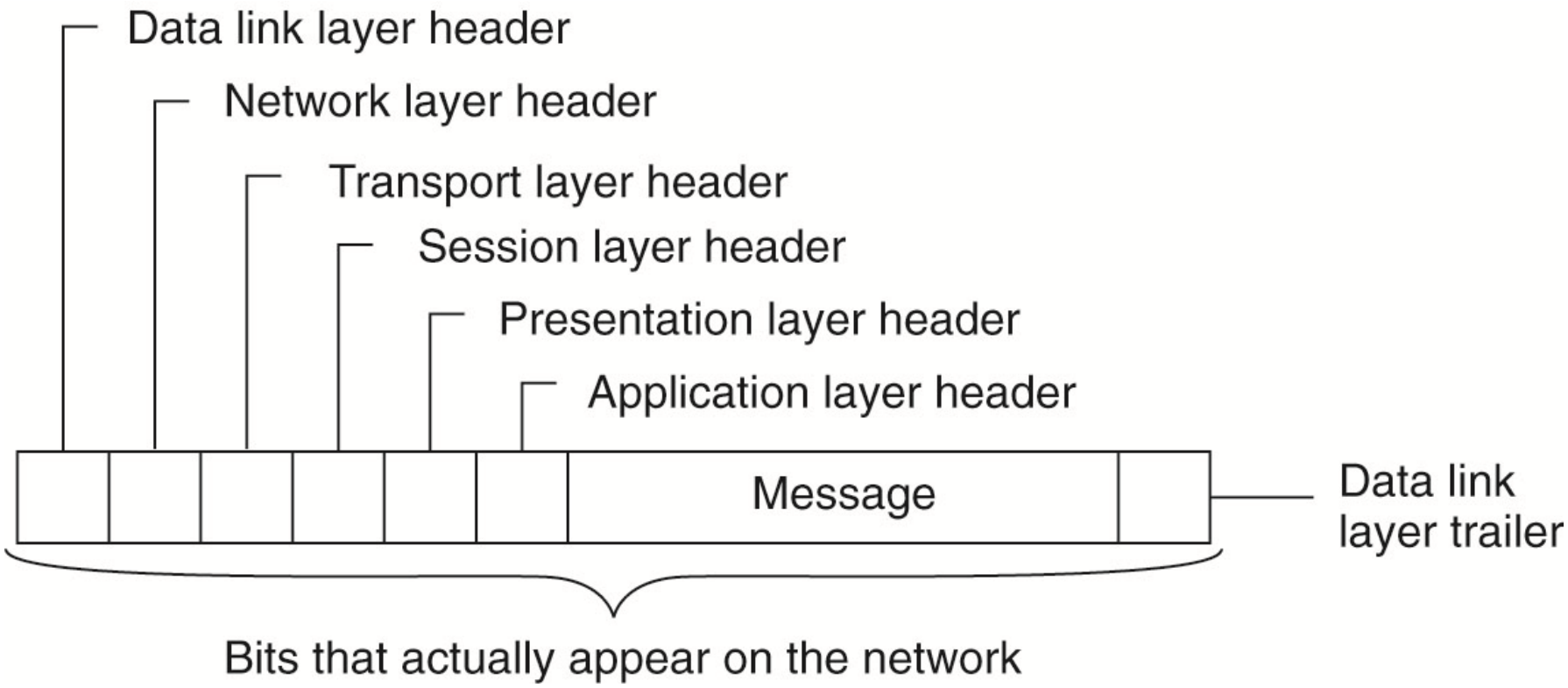

مدل **OSI** یک مدل مرجع برای اداره و کنترل سطوح مختلف موجود در ارتباطات است

# مدل OSI

در مدل OSI هر لایه، واسطی را به لایه بالاتر از خودش ارائه می کند.

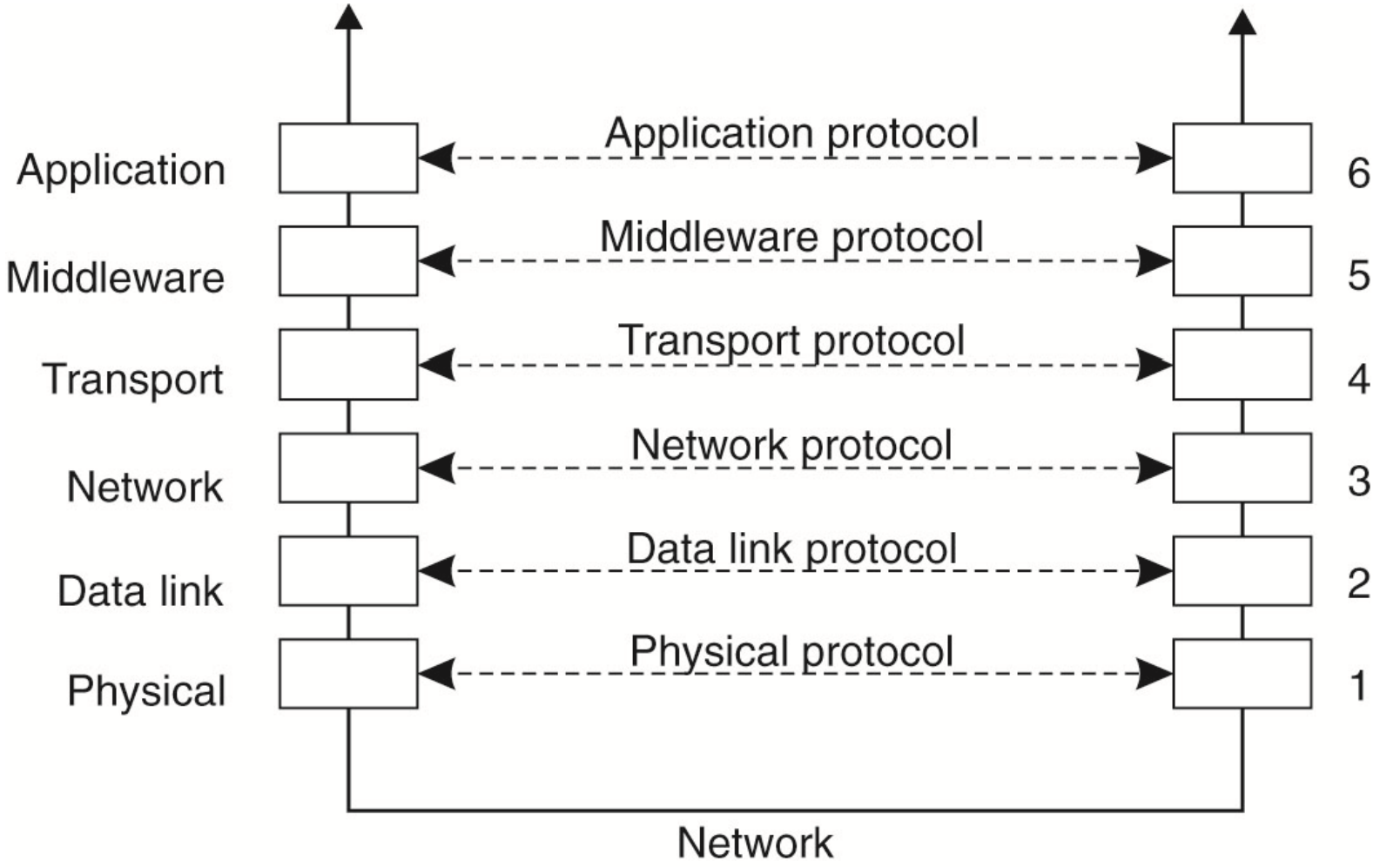
وقتی فرآیند A در ماشین ۱ می خواهد با فرآیند B در ماشین ۲ ارتباط برقرار کند، پیامی می سازد و آن را به لایه کاربرد در ماشین خودش می فرستد و نرم افزار لایه کاربرد، سرآیندی را به جلوی پیام اضافه می کندو پیام از طریق واسط به لایه نمایش ارسال می شود و به همین ترتیب ادامه می یابد.

# نمونه ای از پیام در شبکه



# مدل مرجع برای ارتباطات سیستم توزیع شده

**در سیستم توزیع شده یک Middleware اضافه می شود.**



# ارتباطات شبکه در سیستم توزیع شده

لایه کاربردی با Middleware در ارتباط است و Middleware تمامی کارها را از دید لایه کاربردی مخفی نگه می دارد.

Middleware وظیفه فراهم کردن سرویس ارتباطات را برای لایه کاربردی فراهم می کند.

# مزایای Middleware

۱- شفافیت (Transparency)

۲- همگن سازی.

۳- در اختیار گذاشتن منابع بصورت آسان برای کاربر.

# انواع ارتباطات

- پایدار (Persistent)
- ناپایدار (Transient)

- همگام (Synchronous)
- ناهمگام (Asynchronous)

# ارتباط پایدار

در ارتباط پایدار وقتی که دو شیئ برای همدیگر انتقال داده می کنند پیغام گم نمی شود.

در ارتباط پایدار نیازی نیست که گیرنده روشن باشد.

این ارتباط به سیستم صف نیز معروف است.

مثال ارتباط پایدار مثل SMS و Email

# ارتباط ناپایدار

در ارتباط ناپایدار وقتی که دو نفر برای همدیگر انتقال داده می کنند ممکن است پیغام گم شود.

در ارتباط ناپایدار باید گیرنده روشن باشد.

# ارتباط همگام و ناهمگام

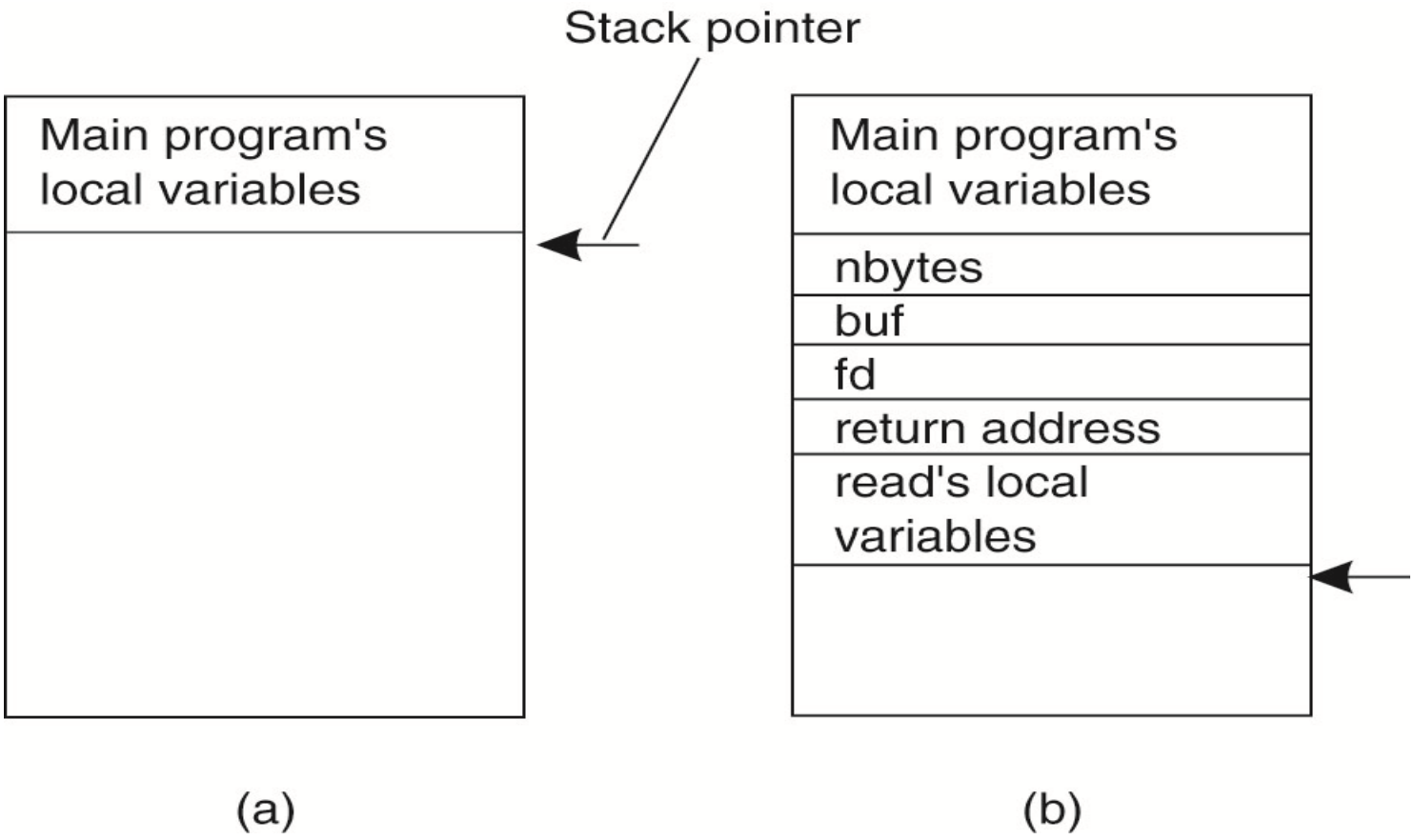
در ارتباط همگام فرستنده بعد از تحویل دادن پیام متوقف می شود تا مطمئن شود درخواست او پذیرفته شده است.

در ارتباط ناهنگام فرستنده به محض تحویل دادن پیام به کارش ادامه می دهد.

# فراخوانی رویه از راه دور (RPC)

- ایده ماورای RPC این است که فراخوانی رویه راه دور را حتی الامکان مشابه با فراخوانی محلی انجام دهد و می خواهیم RPC شفاف باشد. (رویه فراخوان نباید اطلاع داشته باشد که در کامپیوتر دیگری اجرا می شود و برعکس)
- تابع یا پروسیجری را که در کامپیوتر خودمان نیست صدا بزنیم و ارتباطات را در تابع قرار می دهیم بدون اینکه کامپیوتر مبدا متوجه شود.

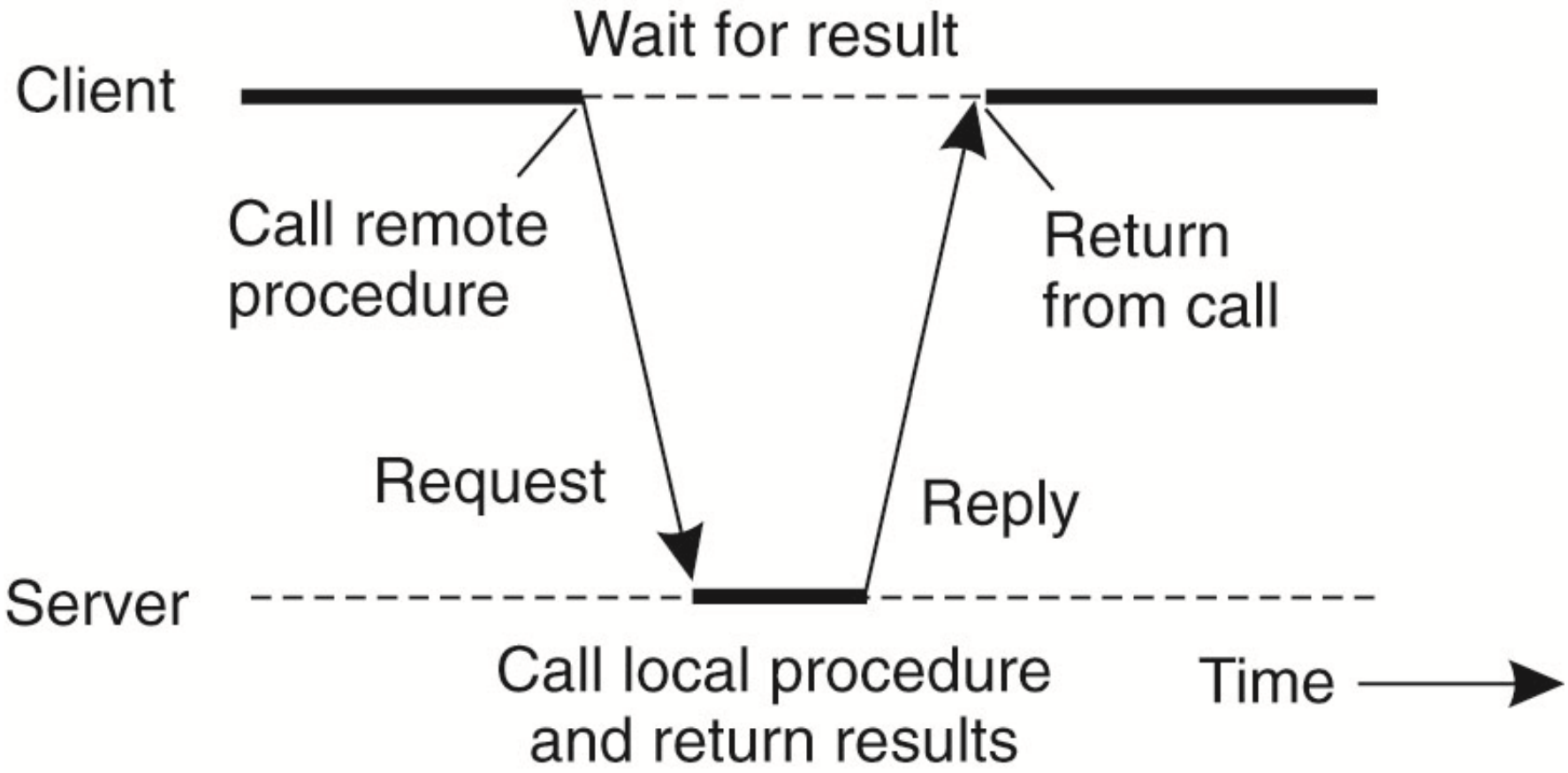
# فراخوانی رویه سنتی




a) پارامترهایی که در فراخوانی رویه محلی ارسال می شوند: پشته قبل از فراخوانی read.

b) پشته در حالی که رویه فراخوانی فعال است.

# نمونه های سرویس گیرنده و سرویس دهنده




اصل RPC بین برنامه سرویس گیرنده و سرویس گیرنده

# مراحل فراخوانی رویه راه دور (۱)

۱- رویه سرویس گیرنده، نمونه سرویس گیرنده را به روش عادی فراخوانی می کند.

۲- ریشه (Stub) سرویس گیرنده، پیامی را می سازد و سیستم عامل راه دور را فراخوانی می کند.

۳- سیستم عامل سرویس گیرنده، پیام را به سیستم عامل راه دور می فرستد.

۴- سیستم عامل راه دور، پیام را به ریشه سرویس دهنده می دهد.

۵- ریشه سرویس دهنده پارامتر را استخراج می کند و سرویس دهنده را فراخوانی می نماید.

# مراحل فراخوانی رویه راه دور (۲)

۶- سرویس دهنده کار را انجام می دهد و نتیجه را به ریشه بر می گرداند.

۷- ریشه سرویس دهنده، نتیجه را در پیام بسته بندی می کند و سیستم عامل محلی خود را فراخوانی می نماید.

۸- سیستم عامل سرویس دهنده پیام را به سیستم عامل سرویس گیرنده می فرستد.

۹- سیستم عامل سرویس گیرنده، پیام را به ریشه سرویس گیرنده می دهد.

۱۰- ریشه پیام را استخراج می کند و به سرویس گیرنده برمی گرداند.

# ارسال پارامتر در RPC

۱- Call by value

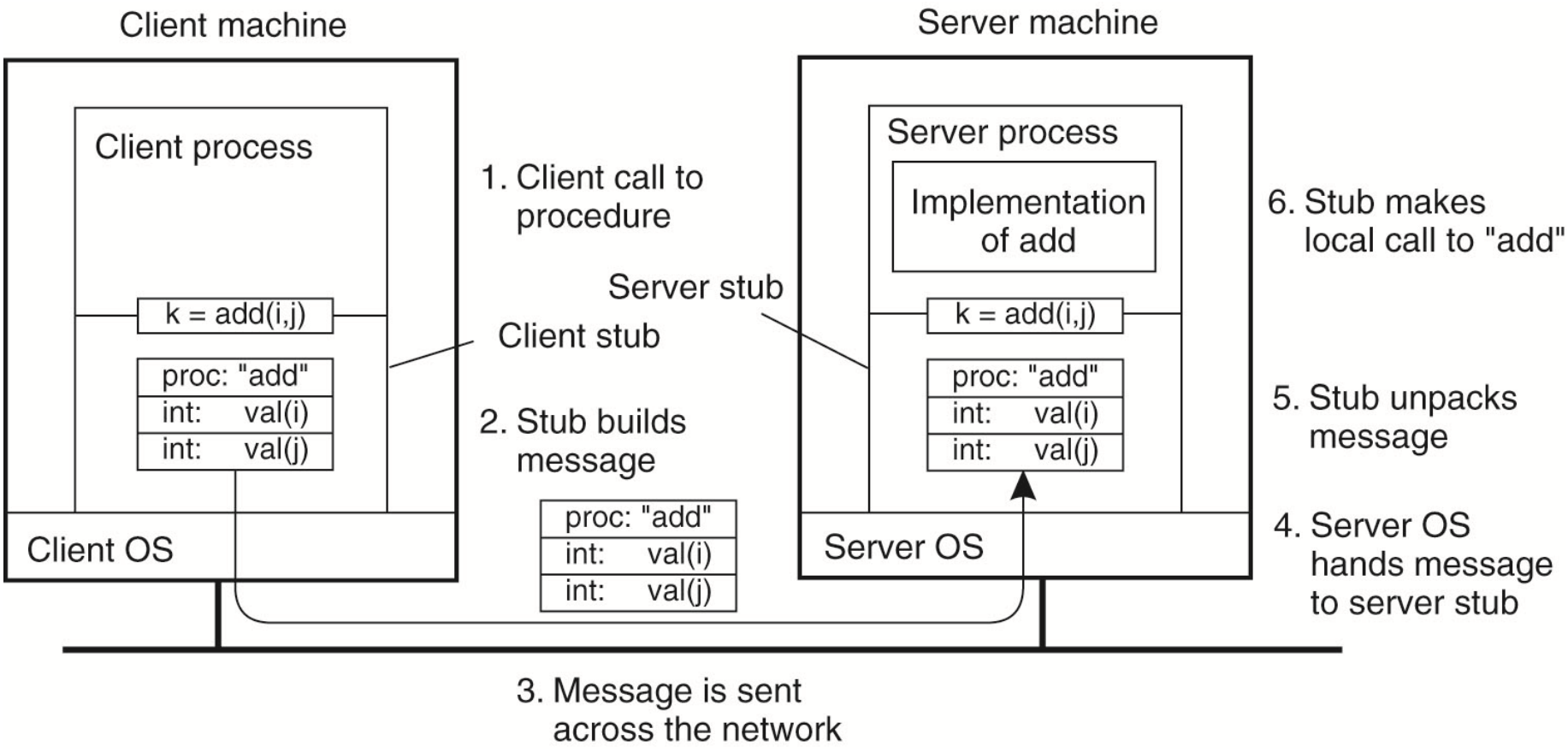
مقدار پارامتر ها را می فرستد.

۲- Call by reference

در این ارسال، آدرس مکان پارامتر فرستاده می شود.

۳- Call by copy/restore (Name)

# ارسال پارامترها با مقدار (ادامه)



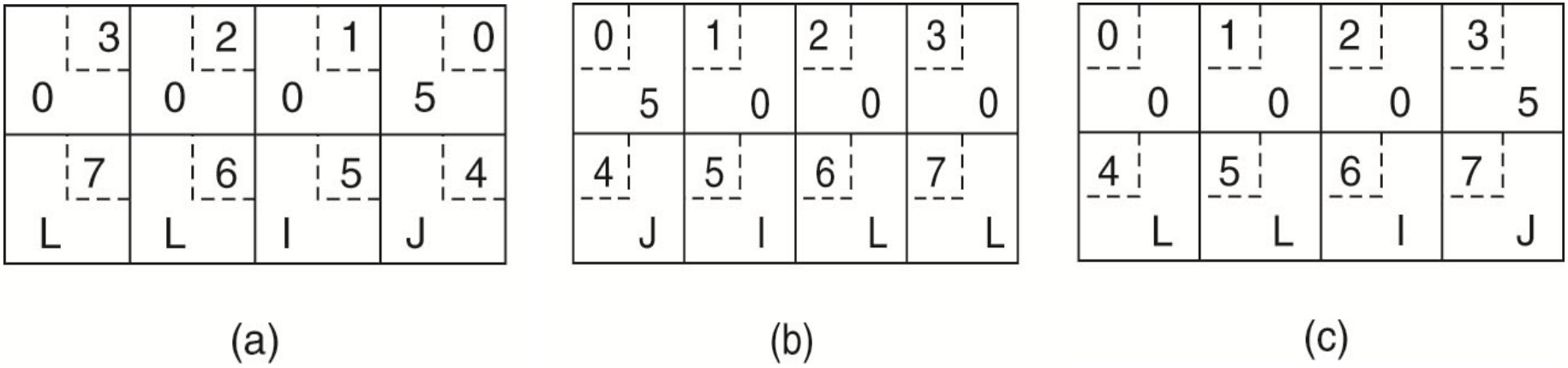

مراحل انجام محاسبات راه دور از طریق RPC.

# ارسال پارامترها با مقدار (ادامه)

**مدل فوق تا زمانی که ماشین های سرویس گیرنده و سرویس دهنده یکسان باشند و تمام پارامتر ها و نتایج از نوع اسکالر باشند به خوبی کار می کند. اما در سیستم های توزیع شده بزرگ (معمولاً ماشین های ناهمگن)، هر ماشین برای مقادیر عددی، کاراکتری و...، نمایش خاصی دارد و ارسال پارامتر با مشکل مواجه خواهد شد.**

بعنوان مثال در کامپیوترهای بزرگ IBM با کامپیوتر های شخصی IBM.

# ارسال پارامترها با مقدار (ادامه)




(a) (b) (c)

**a)پیام اصلی در پنتیوم.**

**b)پیام پس از دریافت روی SPARC.**

**c)پیام پس از معکوس شدن اعداد کوچک در کادرها،آدرس هر بایت را نشان می دهد.**

# ارسال پارامترها با مرجع

در این ارسال، آدرس مکان فرستاده می شود. و چون آدرسها در دو کامپیوتر مختلف یکسان نیست این روش امکانپذیر نیست.

راه حل ۱): اجازه ارسال شدن پارامترهای مرجع را ندهیم و تنها پارامترهای مقدار اجازه ارسال شدن داشته باشند.

راه حل ۲): ارسال پارامترها با کپی/بازیابی

# ارسال پارامترها با کپی و بازیابی

**داده ای که پارامتر مرجع به آن اشاره می کند برای سرویس گیرنده ارسال شود و آن تغییرات روی آن انجام پذیرد و سپس کل داده باز گردد.**

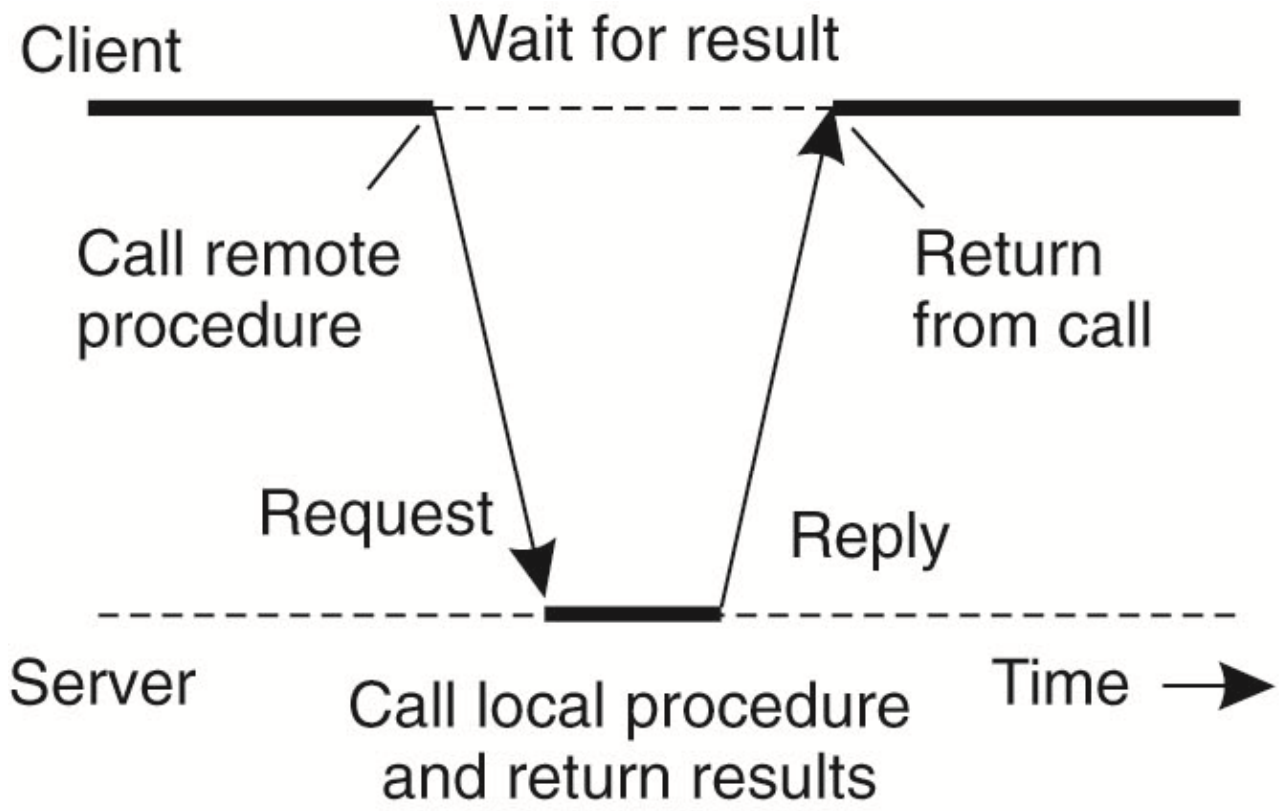
**به دلیل اینکه داده هایی که به آنها ارجاع داده می شود اغلب بزرگ هستند (مثل آرایه) این عمل** ترافیک مسیر ارتباطی **را بالا می برد.**

# RPC همگام

همانند فراخوانی های رویه معمولی، سرویس دهنده پس از درخواست RPC، مسدود می شود .

مثال) انتقال پول از یک حساب به حساب دیگر، افزودن داده هایی به بانک اطلاعاتی، پردازش دسته ای و...

# RPC همگام



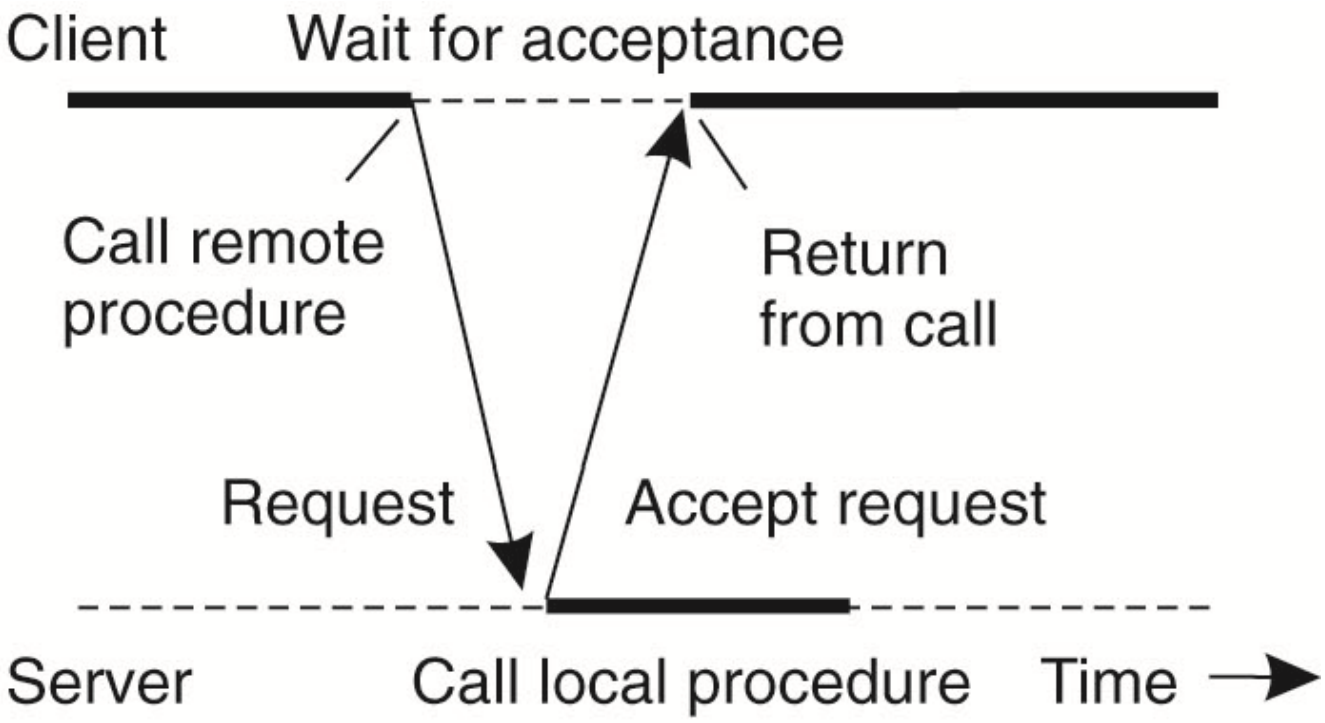

تعامل بین سرویس دهنده و سرویس گیرنده در RPC سنتی.

# RPC ناهمگام

سرویس دهنده پس از درخواست RPC، فوراً به کارش ادامه می دهد.

با RPCهای ناهمگام، سرویس دهنده بلافاصله پس از دریافت فراخوانی RPC پاسخی را ارسال می کند.

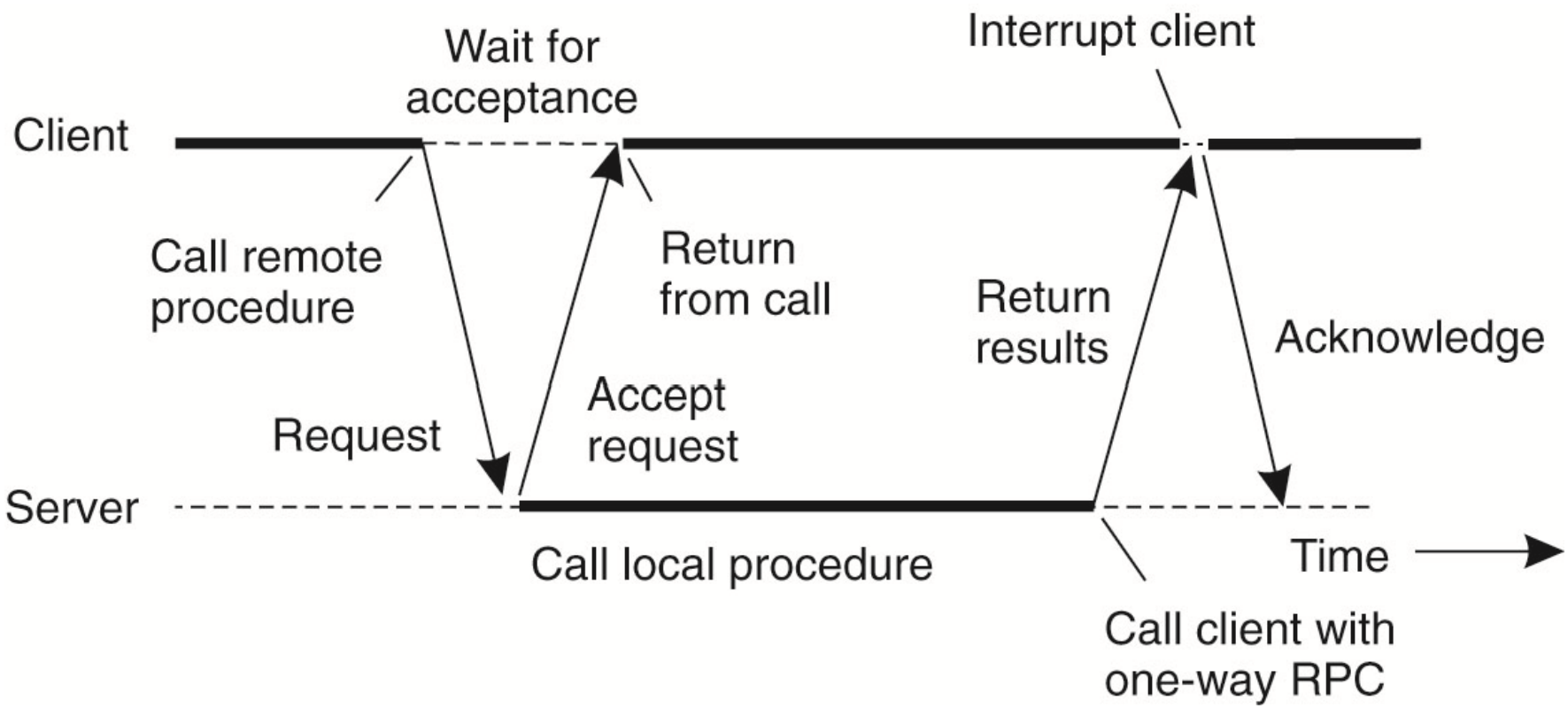
# RPC ناهمگام




(b)

**تعامل با استفاده از RPC ناهمگام.**

# RPC ناهمگام



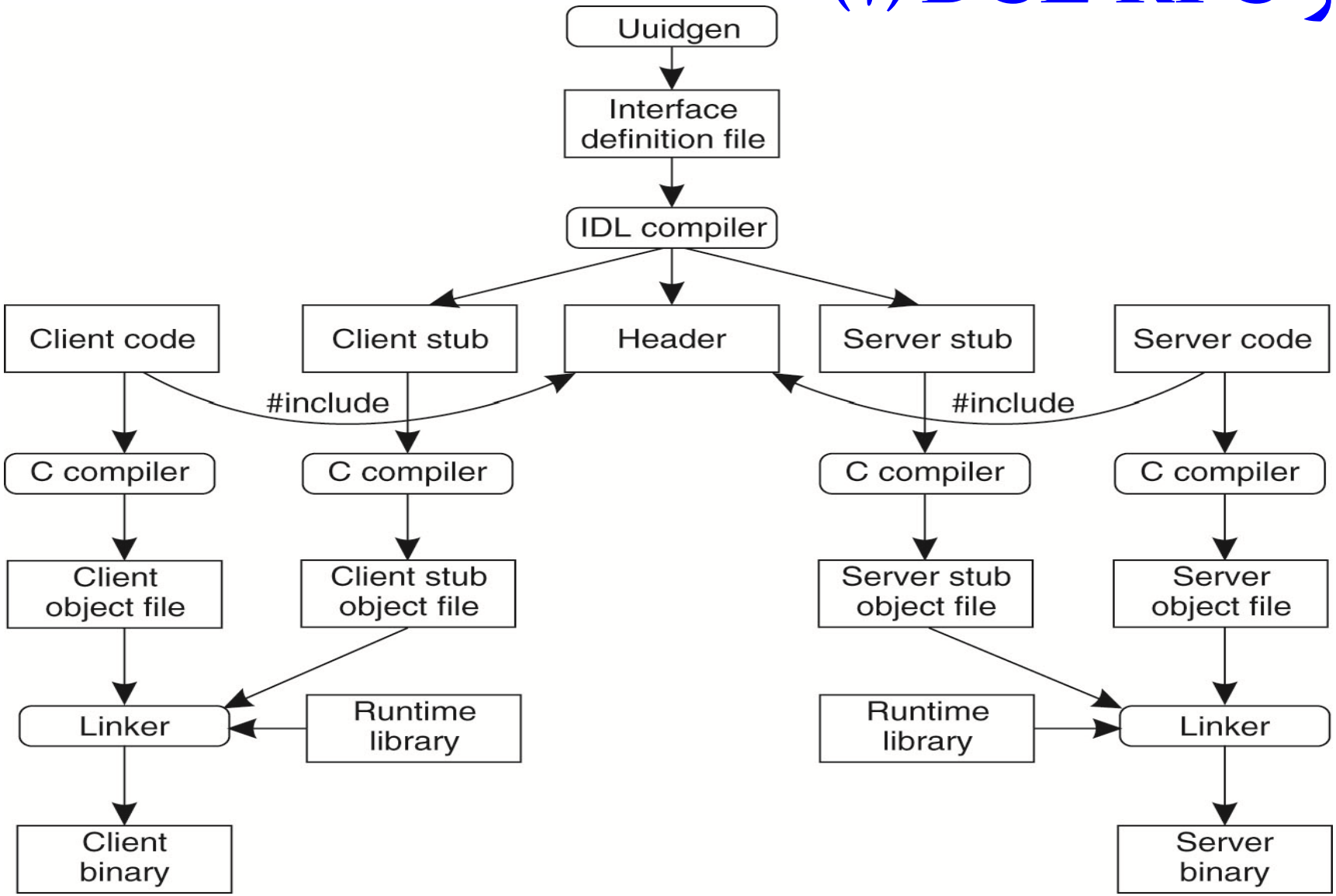

تعامل سرویس دهنده و سرویس گیرنده از طریق دو RPC ناهمگام.

# DCE RPC

Distributed Computing Environment DCE یک سیستم میان افزار محض است که طراحی شد تا به عنوان لایه ای از انتزاع بین سیستم های عامل (شبکه) موجود و کاربردهای توزیع شده اجرا شود.

ایده DCE : مشتری بتواند مجموعه ای از ماشین های موجود را بگیرد، نرم افزار DCE را اضافه کند، وسپس بتواند کاربردهای توزیع شده را اجرا نماید.

# مراحل نوشتن سرویس گیرنده و سرویس دهنده در DCE RPC (۱)



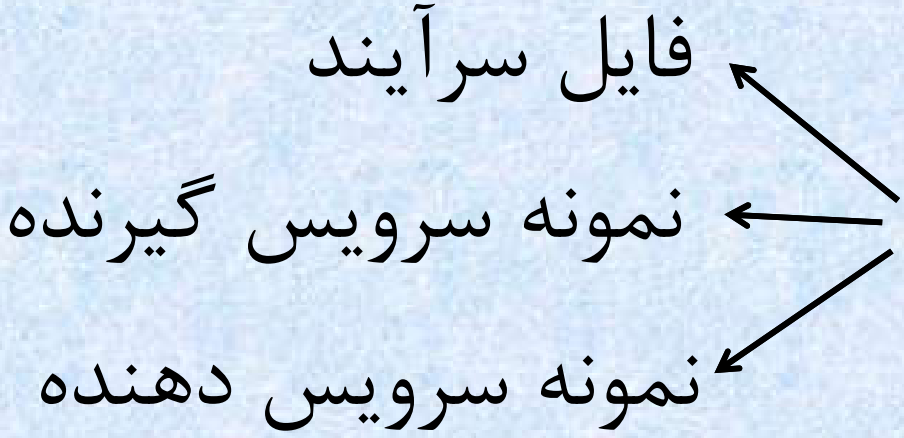
# مراحل نوشتن سرویس گیرنده و سرویس دهنده در DCE RPC (۲)

اولین مرحله فراخوانی برنامه Uuidgen است، و الگوی فایل IDL را ایجاد می نماید.

مرحله بعدی، ویرایش فایل IDL است، که اسامی و رویه های راه دور و پارامتر های آن مشخص می شود.

وقتی فایل IDL کامل شد، کامپایلر IDL ، برای پردازش آن فراخوانی می شود.

# مراحل نوشتن سرویس گیرنده و سرویس دهنده در DCE RPC (۳)




مرحله بعدی، نوشتن کد سرویس دهنده و سرویس گیرنده است.

سپس هر دو ترجمه می شوند و...

# ارتباط ناپایدار پیام گرا
# Transient Message-oriented

سوکت برکلی

واسط مبادله پیام (MPI)

# سوکت برکلی

توجه ویژه ای به استاندارد سازی واسط لایه انتقال انجام شد تا به برنامه نویسان اجازه دهد از طریق مجموعه ای از عملیات های پایه، از کل پشته پروتکل ها استفاده کنند.

واسط های استاندارد، انتقال کاربرد را به ماشین مختلف آسان می سازد.

# عملیات های پایه سوکت برای TCP/IP

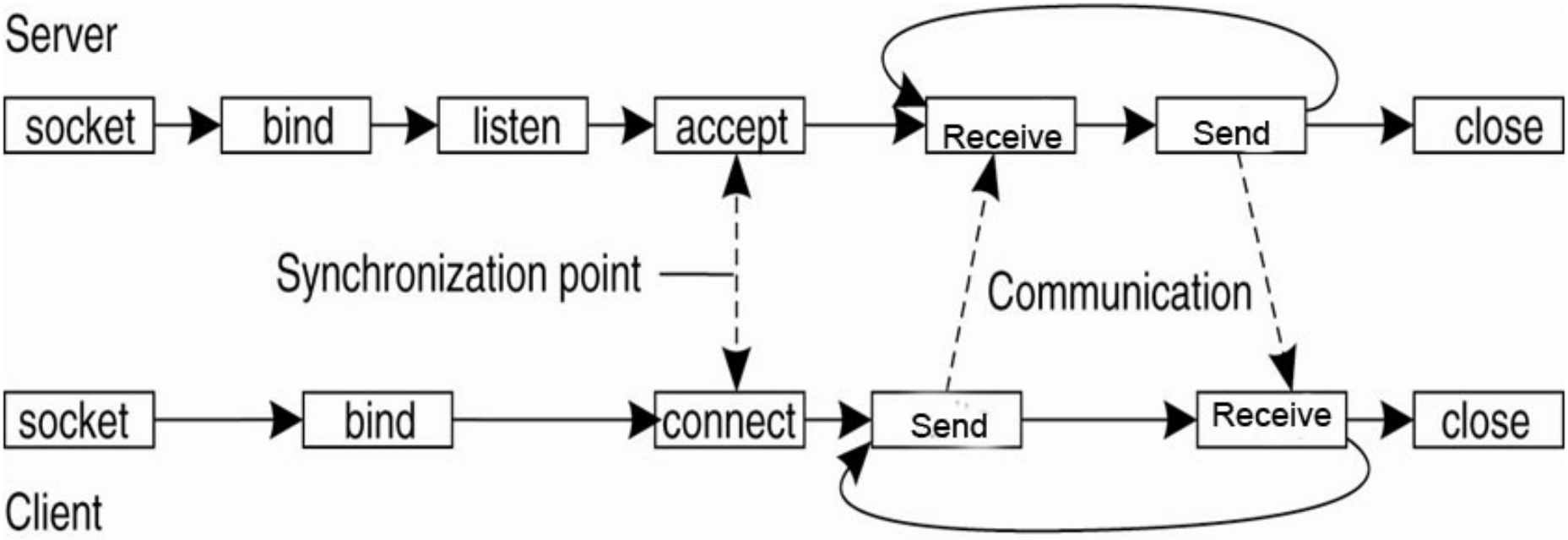
| Primitive | Meaning |
|---|---|
| Socket | Create a new communication end point |
| Bind | Attach a local address to a socket |
| Listen | Announce willingness to accept connections |
| Accept | Block caller until a connection request arrives |
| Connect | Actively attempt to establish a connection |
| Send | Send some data over the connection |
| Receive | Receive some data over the connection |
| Close | Release the connection |

فقط server

فقط client

سرویس دهنده ها معمولاً چهارعملیات پایه اول را به همین ترتیب اجرا می کنند.

# الگوی ارتباطی اتصال گرا با استفاده از سوکت ها



# معایب سوکت ها

۱- با پشتیبانی از عملیات پایه **Send** و **Receive** ساده، در سطح نادرستی از انتزاع قرار دارند.

۲- برای پروتکل های اختصاصی که برای شبکه های متصل به هم و با سرعت بالا مناسب نیستند.

# واسط مبادله پیام (Massage Passing Interface)

**MPI برای کاربردهای موازی طراحی شد و به همین دلیل از نوع ارتباط گذرا است.**

**MPI استفاده مستقیم از شبکه را فراهم می سازد.**

**MPI مستقل از پلات فرم است.**

**MPI فرض می کند ارتباط در داخل گروه شناخته شده ای از فرآیند ها صورت می گیرد.**

# واسط مبادله پیام (MPI)

**بعضی از مهمترین عملیات های پایه (Primitive) مبادله پیام MPI**

| Primitive | Meaning |
|---|---|
| MPI_bsend | Append outgoing message to a local send buffer |
| MPI_send | Send a message and wait until copied to local or remote buffer |
| MPI_ssend | Send a message and wait until receipt starts |
| MPI_sendrecv | Send a message and wait for reply |
| MPI_isend | Pass reference to outgoing message, and continue |
| MPI_issend | Pass reference to outgoing message, and wait until receipt starts |
| MPI_recv | Receive a message; block if there is none |
| MPI_irecv | Check if there is an incoming message, but do not block |

# ارتباط پایدار پیام گرا

سیستم صف بندی پیام (میان افزار پیام گرا(MOM))

Message-Oriented Middleware

معماری کلی صف بندی پیام

کارگزاران پیام

# سیستم صف بندی پیام (۱)

برنامه های کاربردی با قرار دادن پیام ها در صف خاص، با یکدیگر ارتباط برقرار می کنند.

پیام معمولاً مستقیماً به سرویس دهنده مقصد منتقل می شود.

هر کاربرد دارای صف اختصاصی خودش است که کاربردهای دیگر می توانند پیام هایی را به آن بفرستند.

صف فقط می تواند توسط کاربرد خودش خوانده شود، اما چندین کاربرد می توانند در یک صف مشترک باشند.

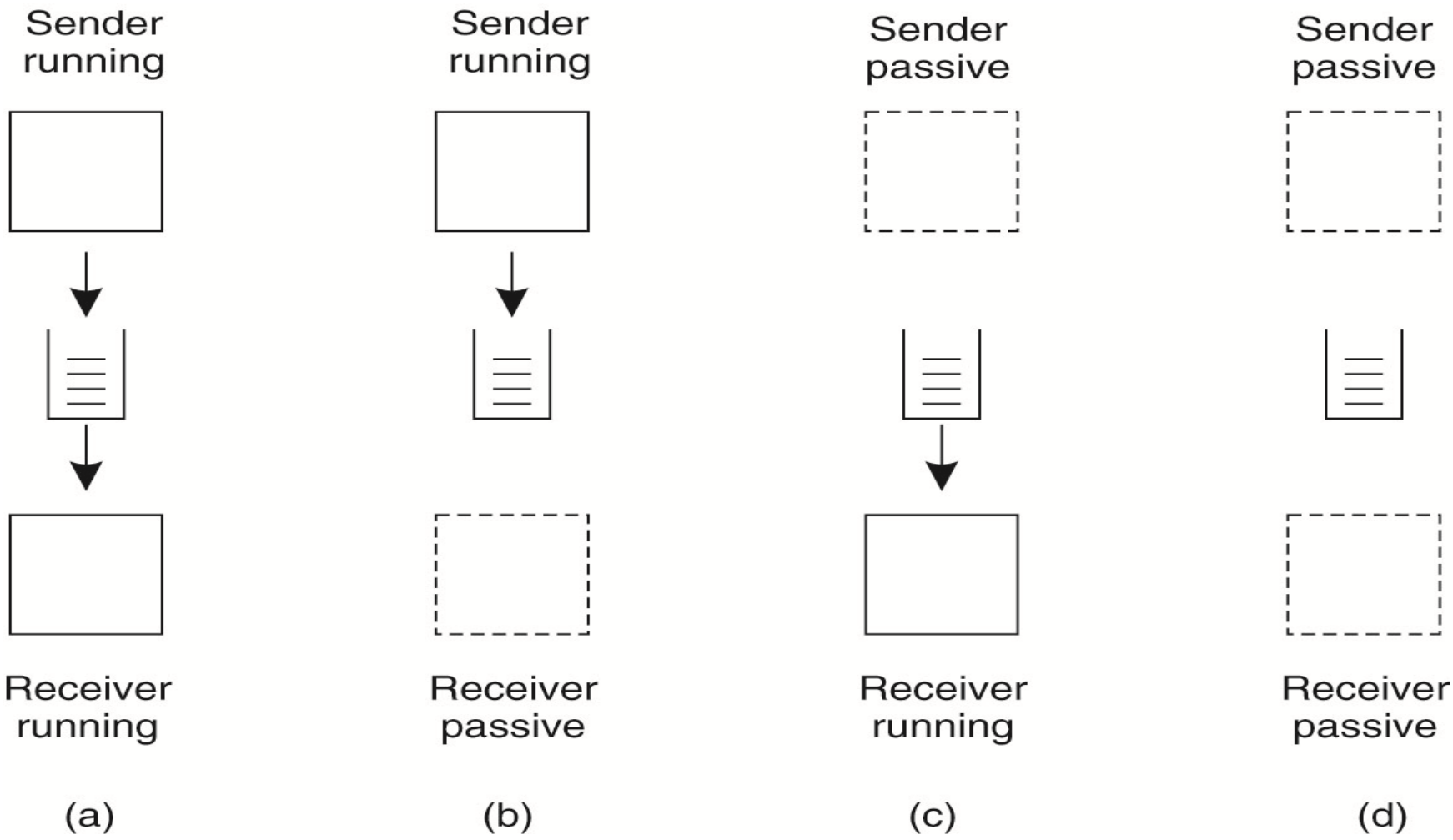
# سیستم صف بندی پیام (۲)

جنبه مهم این مدل این است که به فرستنده اطمینان داده می شود که پیامش سرانجام در صف گیرنده قرار خواهد گرفت.

فرستنده و گیرنده می توانند کاملاً مستقل از یکدیگر اجرا شوند. (ارتباط از نوع اتصال ضعیف)

وقتی پیامی در صف قرار گرفت، در آنجا می ماند تا حذف شود و کاری ندارد که فرستنده و گیرنده اش در حال اجرا است یا خیر.

# سیستم صف بندی پیام (۳)



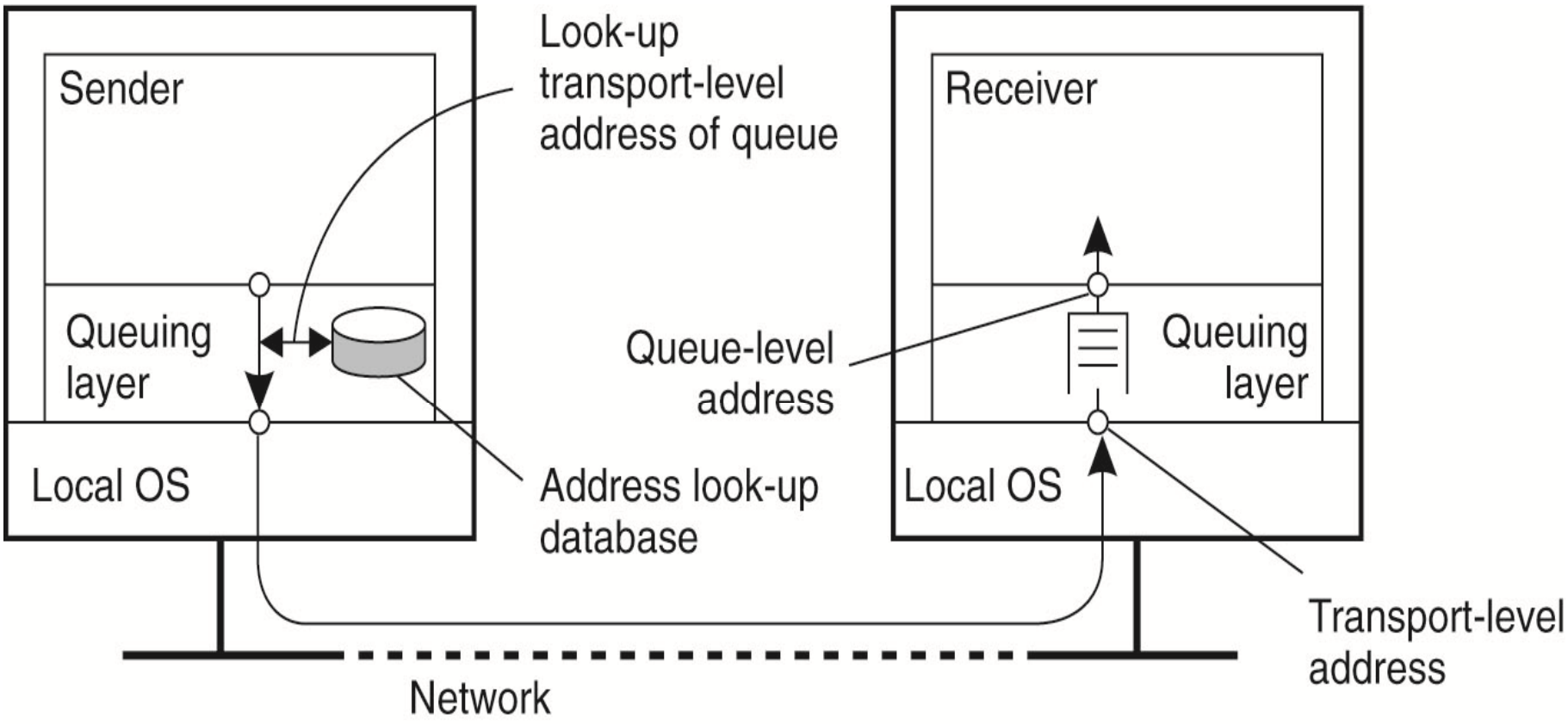

**چهار ترکیب برای ارتباط با انسجام ضعیف با استفاده از صف**

# سیستم صف بندی پیام (۴)

**پیام ها می توانند شامل هر داده ای باشند و تنها باید به درستی آدرس دهی شوند.**

| Primitive | Meaning |
|---|---|
| Put | Append a message to a specified queue |
| Get | Block until the specified queue is nonempty, and remove the first message |
| Poll | Check a specified queue for messages, and remove the first. Never block |
| Notify | Install a handler to be called when a message is put into the specified queue |

# سیستم صف بندی پیام (۵)




رابطه بین آدرس دهی سطح صف و آدرس دهی سطح شبکه.

# معماری کلی سیستم صف بندی پیام (۱)

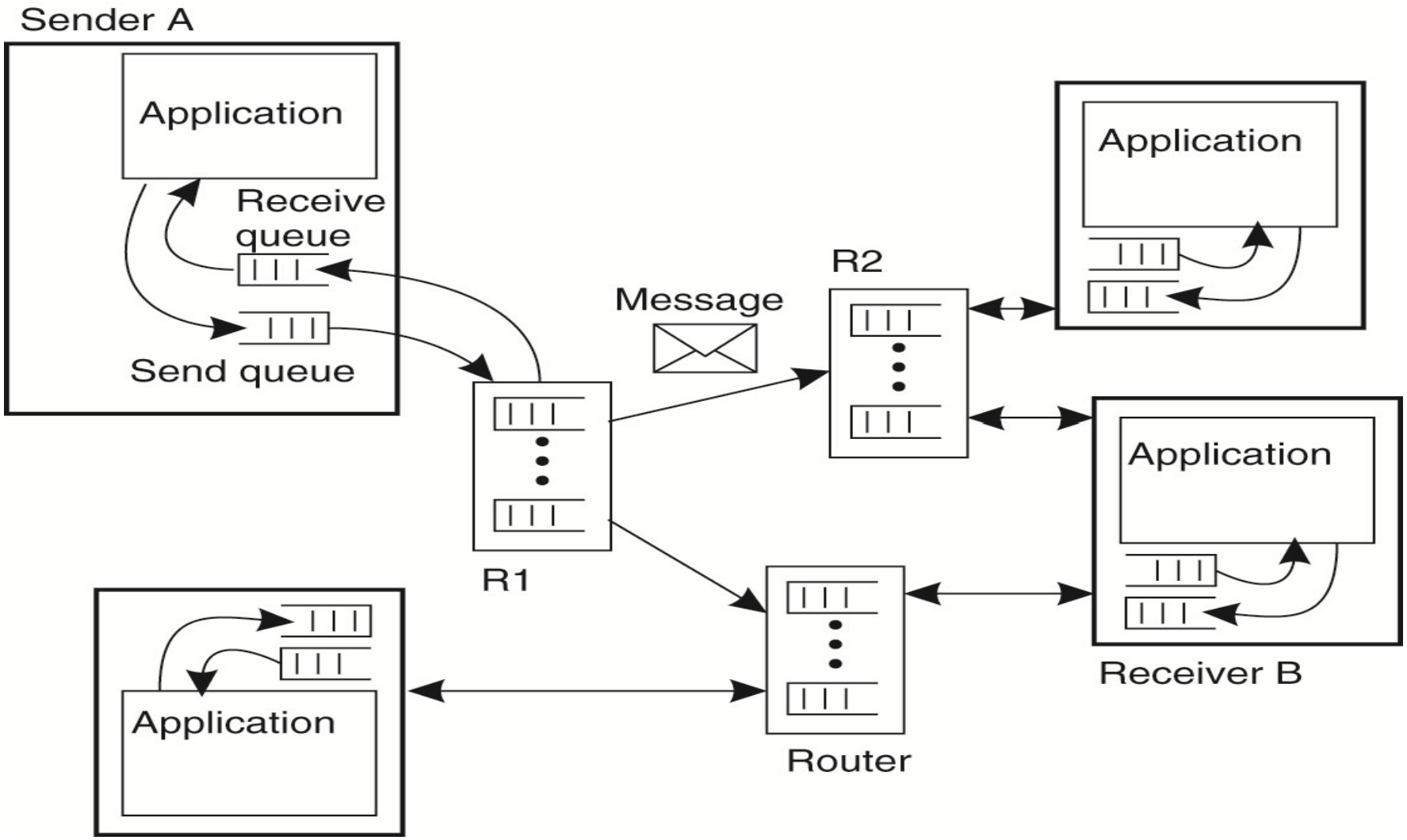
مجموعه ای از صف ها در چندین ماشین توزیع شده اند و برای اینکه سیستم صف بندی پیام بتواند پیام را منتقل کند، باید نگاشتی از صف ها به مکان های شبکه را نگهداری نماید.

این نگاشت شبیه استفاده از DNS در اینترنت است.

صف ها به وسیله مدیران صف اداره می شوند.

مدیر صف مستقیماً با کاربردی که پیام را می فرستد یا می گیرد، تعامل دارد. اما مدیران صف خاصی وجود دارند که مثل مسیریاب یا تقویت کننده عمل می کند.

# معماری کلی سیستم صف بندی پیام (۲)




**سازمان کلی سیستم صف بندی پیام با مسیر یاب ها.**

# کارگزاران (Broker) پیام (۱)

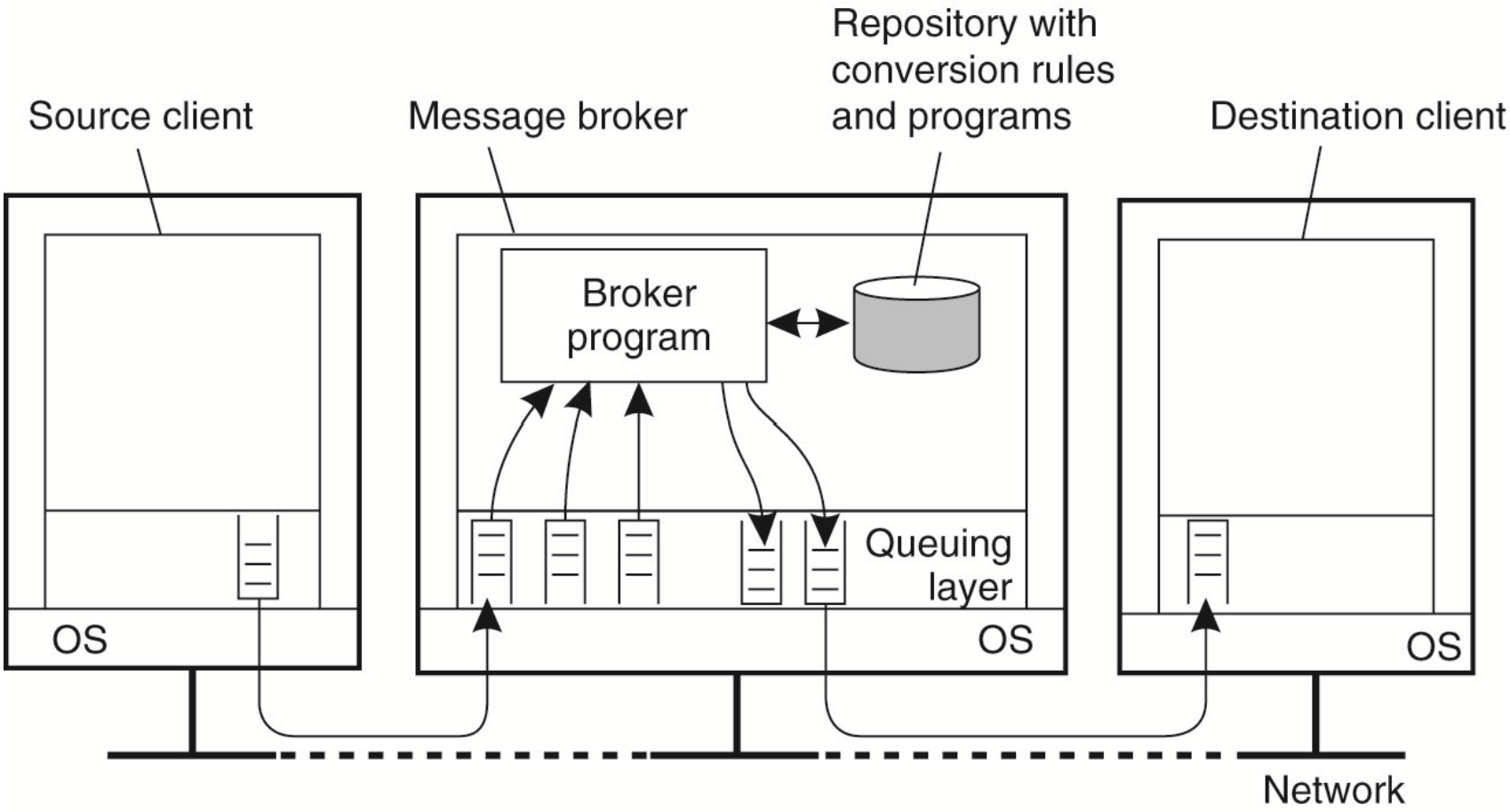
هدف اصلی تبدیل پیام های ورودی است، به طوری که توسط کاربرد مقصد قابل فهم باشند.

کارگزاران پیام بخشی از سیستم صف بندی محسوب نمی شود.

استفاده از کارگزاران پیام برای جامعیت کاربرد است.

در قلب کارگزاران پیام، مخزنی از قوانین و برنامه ها وجود دارد که می تواند پیامی از نوع T1 را به T2تبدیل کند.

# کارگزاران پیام (۲)




**سازمان کلی کارگزاران پیام در سیستم صف بندی پیام.**

# جریان داده (Data Stream) (۱)

جریان داده، دنباله ای از واحد های داده است.

جریان های داده می توانند به رسانه پیوسته و گسسته تقسیم شوند.

زمان برای جریان های داده پیوسته مهم است. مثال : نمایش فیلم برای پیوسته. مثال تصاویر ثابت برای گسسته

برای دستیابی به جنبه های زمانی معمولاً بین حالت های مختلف انتقال، تمایز قائل می شویم.

# جریان داده (۲)

**در** حالت انتقال ناهمگام asynchronous، **اقلام داده ها در یک جریان داده بدون محدودیت زمانی یکی پس از دیگری منتقل می شوند، این کار معمولاً در جریان های داده گسسته صورت می گیرد.**

**در** حالت انتقال همگام synchronous، **حداکثر تأخیر انتها به انتها وجود دارد که برای هر واحد داده تعریف شده است. مهم نیست که واحد داده سریع تر از حداکثر تأخیر منتقل شود یا خیر. مثال: شبکه حسگر برای فهم رخداد آتش**

**در** حالت انتقال همزمان isochronous، **باید الزاما یک تاخیر زمانی ثابت رعایت شود. مثال: کاربرد مولتی مدیا**

# جریان داده (۳)

**جریان ها می توانند ساده یا پیچیده باشند.**

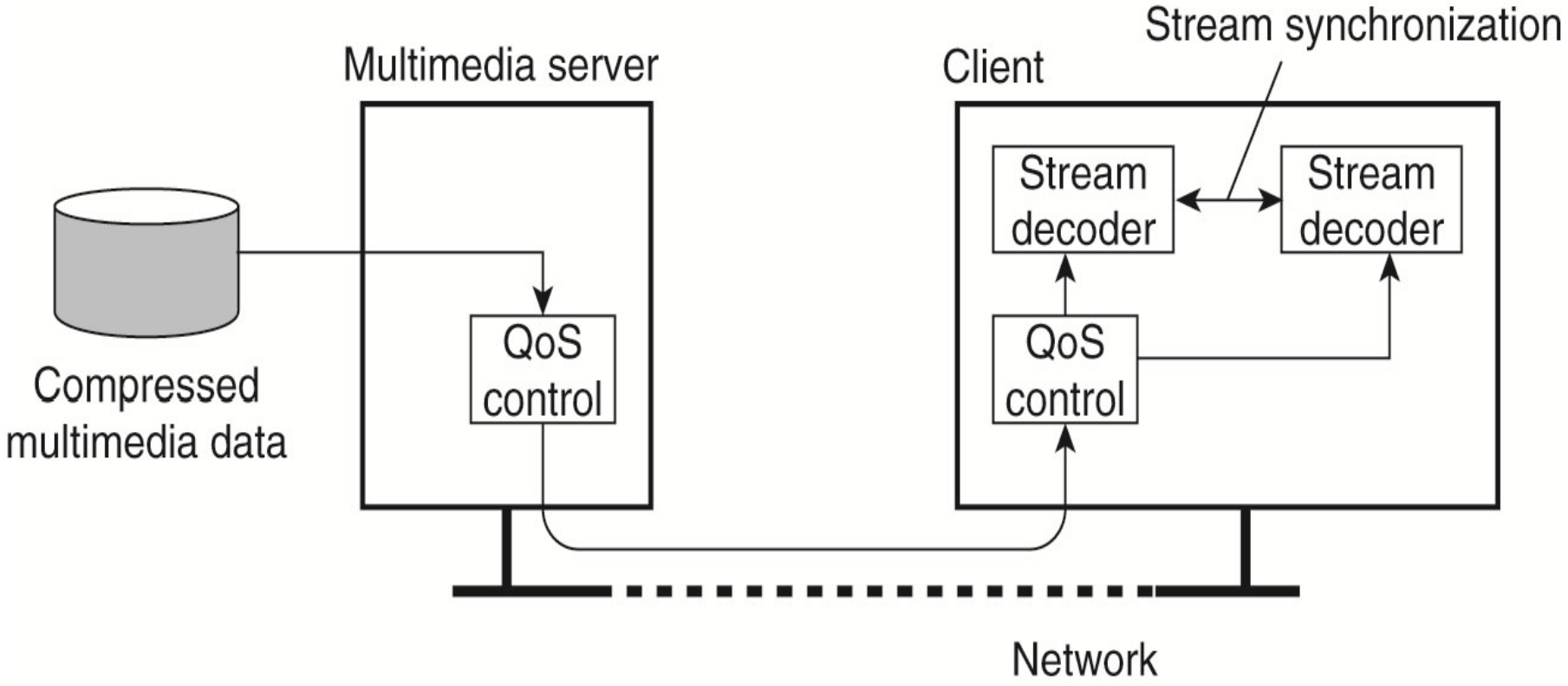
جریان ساده simplex **فقط شامل یک دنباله از داده ها است،در حالی که**

جریان پیچیده complex **شامل چند جریان ساده مرتبط، به نام زیر جریان ها substream است.**

**ارتباط بین زیر جریان ها در جریان پیچیده، به زمان وابسته است.**

**مثال) صدای استریو یا انتقال فیلم.**

# جریان داده (۴)




**معماری کلی برای جریان داده های چند رسانه ای ذخیره شده روی شبکه.**

# کیفیت سرویس(۱)

**نیازمندیهای زمانی معمولاً به عنوان نیازمندی های کیفیت سرویس (QoS) نامیده می شوند.**

**کیفیت خدمات معمولا از سه جنبه سنجیده می شود:**

۱-زمان

۲-حجم و اندازه

۳- قابلیت اطمینان(تحمل پذیری خطا)

# کیفیت سرویس(۲)

**خصوصیاتی برای QoS:**

۱- نرخ بیتی مورد نیاز که داده ها باید منتقل شوند.

۲- حداکثر تأخیر برای برقراری نشست (یعنی زمانی که کاربرد می تواند شروع به ارسال داده ها کند)

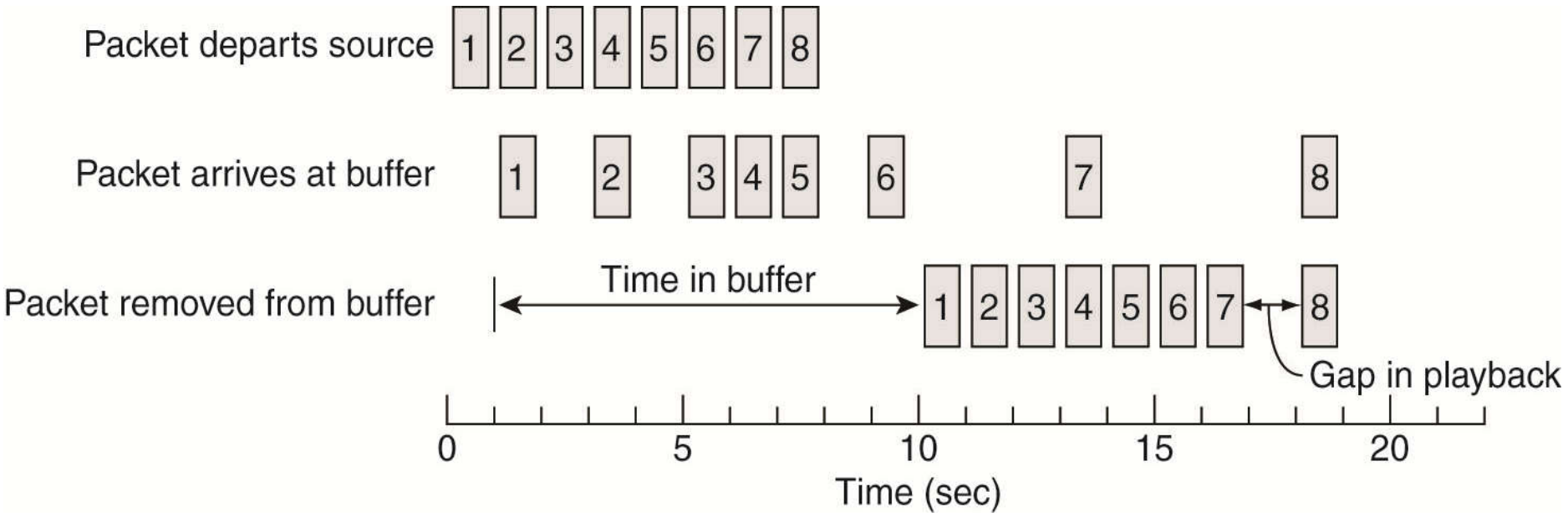
۳- حداکثر تأخیر انتها به انتها (چقدر طول می کشد تا داده ها به گیرنده برسد)

۴- حداکثر واریانس تأخیر یا لرزش Jitter

۵- حداکثر تأخیر رفت و برگشت

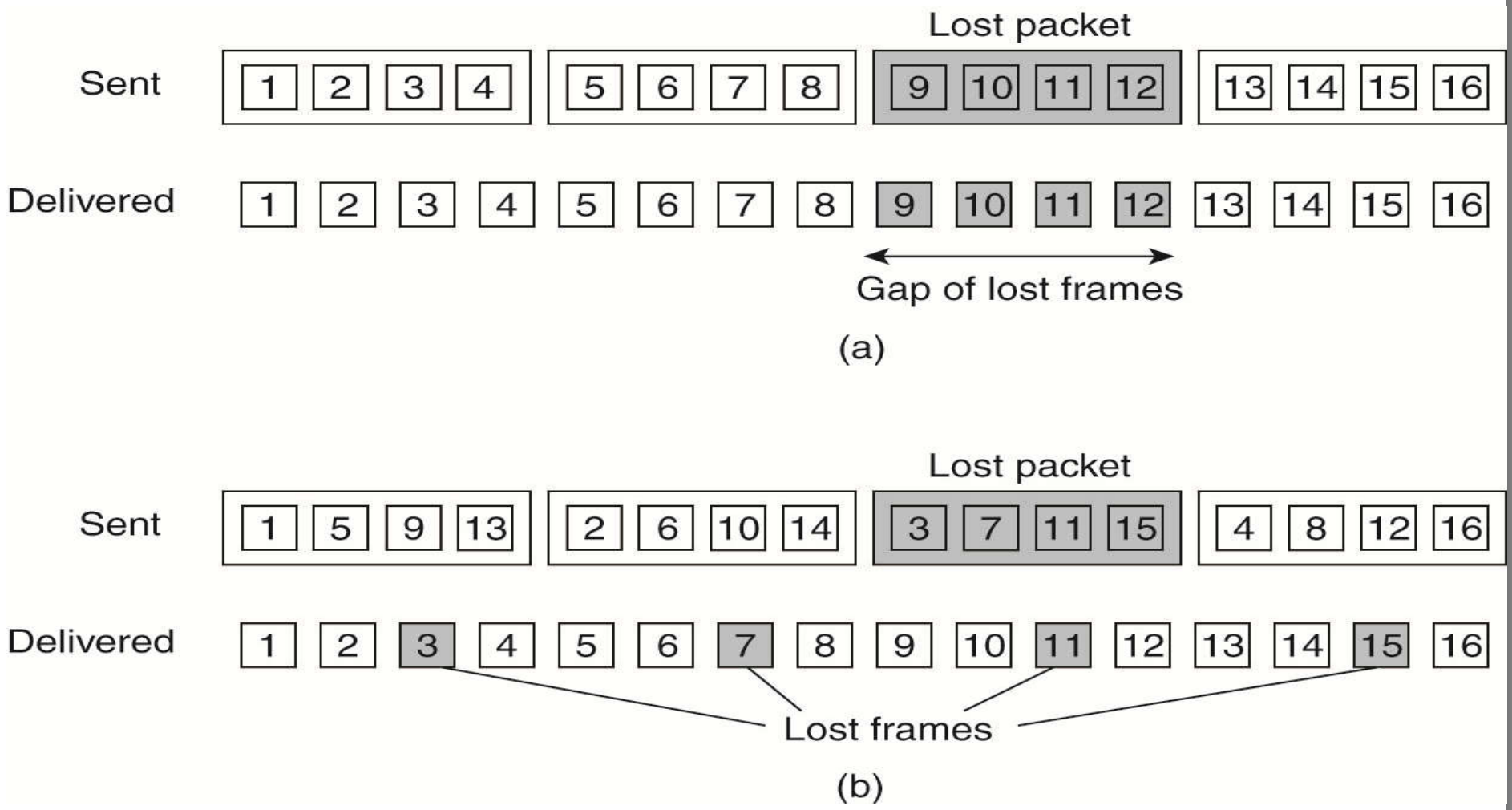
۶- نرخ خطا

# فراهم کردن QoS



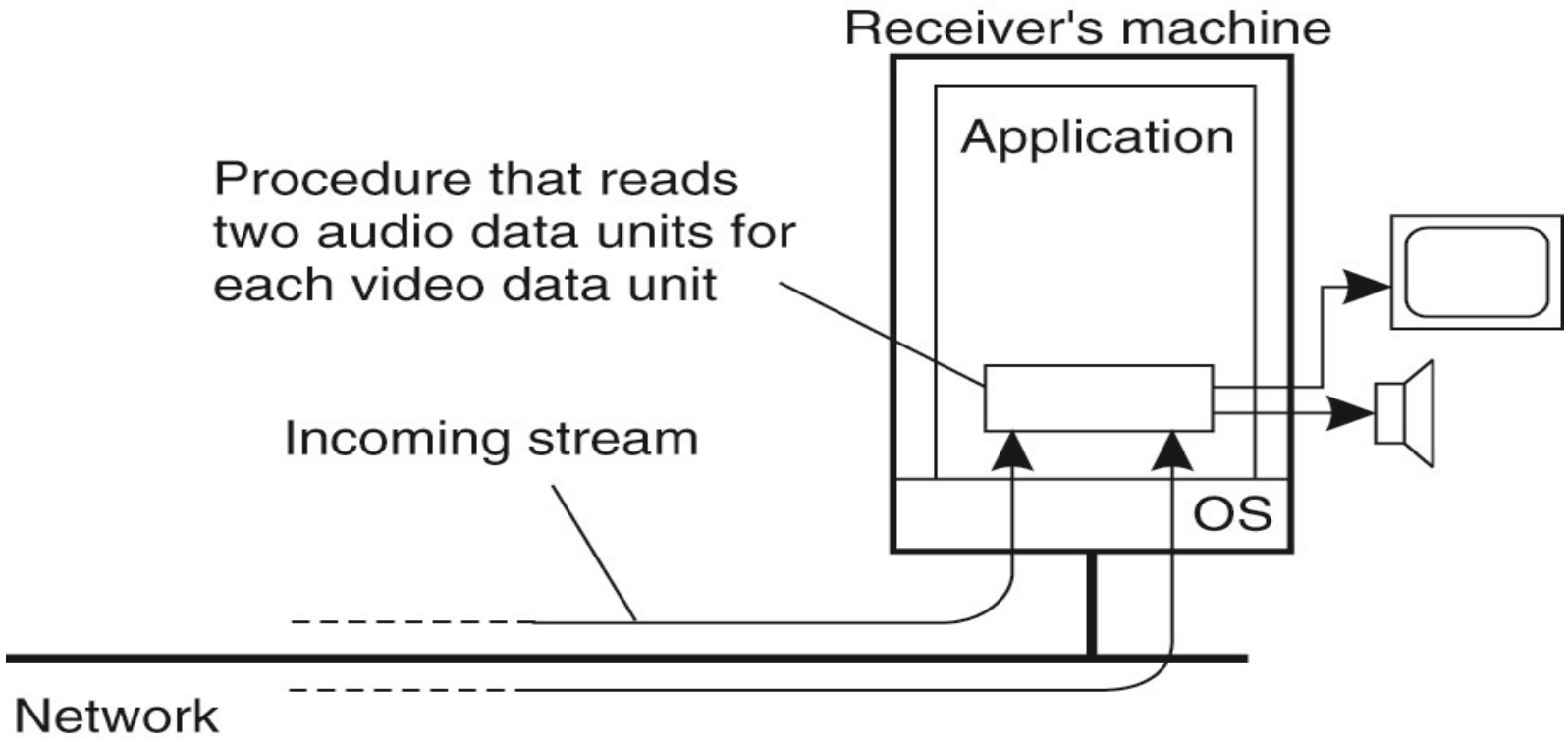

**Using a buffer to reduce jitter**

# اثر مفقود شدن بسته




a) انتقال بدون فاصله گذاری  b) انتقال با فاصله گذاری

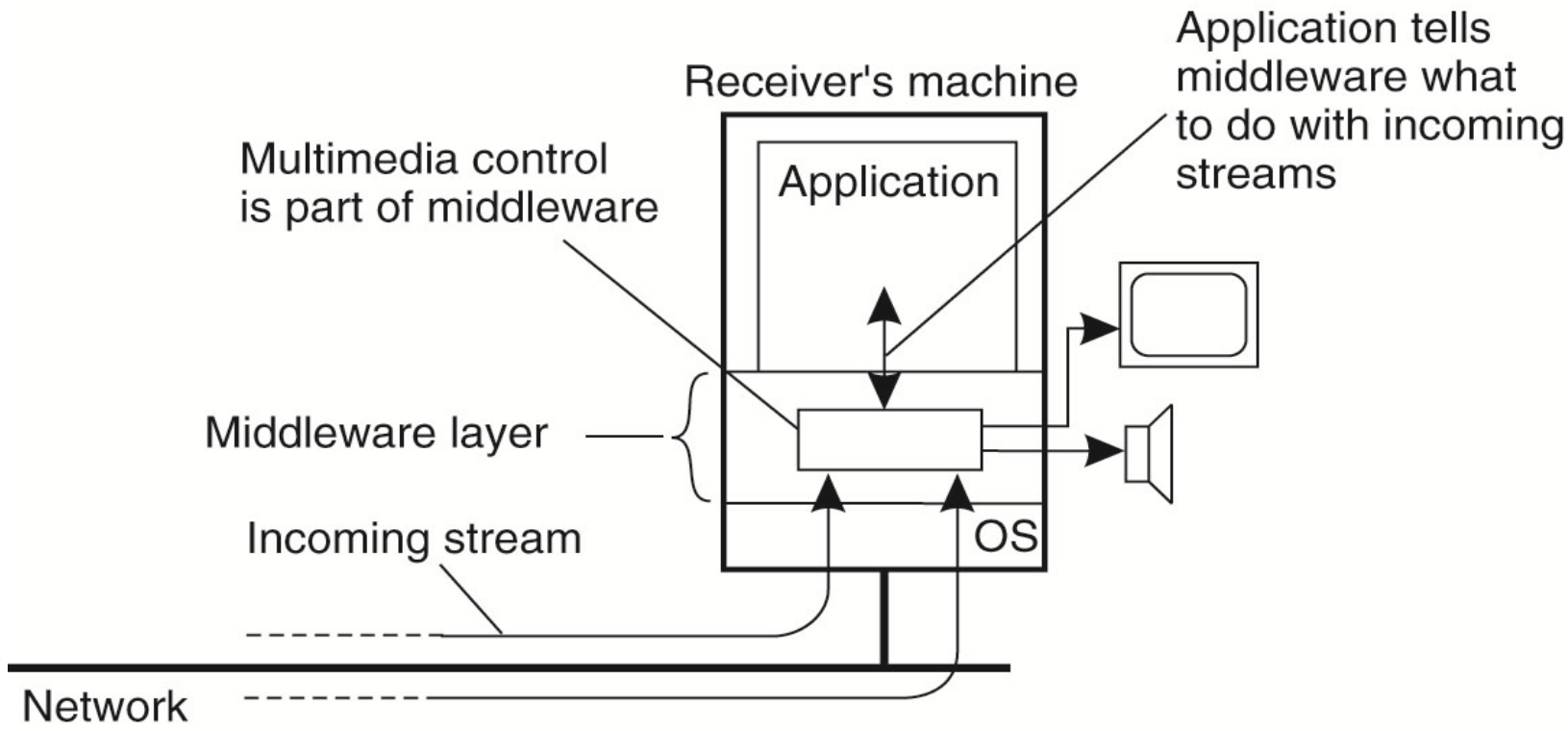
# راهکار های همگام سازی(۱)




**اصل همگام سازی صریح در سطح واحد های داده**

**دو زیر داده تصویر و صدا که بایستی همگام شوند مانند دوبله. بر اساس کیفیت تصویر مثلا ۳۲۰*۲۴۰ پیکسل (هر پیکسل یک بایت در کل ۷۶۸۰۰ بایت) در نرخ بیتی مشخص مثلا 3Mb/s**

# راهکار های همگام سازی(۲)




**اصل همگام سازی که توسط واسط های بالا پشتیبانی می شود.**

**دو زیر داده تصویر و صدا که بایستی همگام شوند مانند دوبله. بر اساس کیفیت تصویر مثلا ۳۲۰*۲۴۰ پیکسل (هر پیکسل یک بایت در کل ۷۶۸۰۰ بایت) در نرخ بیتی مشخص مثلا 3Mb/s**

# Multicast communication

- **Multicast communication: sending data to multiple receivers simultaneously**
- **Node organize into an overlay network, which is then used to disseminate information to its members**
- **Two ways for organizing the network:**

  **Tree-based network: a unique (overlay) path between every pair of nodes**

  **Mesh network: every node will have multiple neighbors, hence multiple paths between pair of nodes (more robust)**