درس و کنکور سیستم عامل پیشرفته

دکتر رمضان عباس نژادورزی
مو لفیـــن :دکتـــر مرتضـــی بابـــا زاده
دکتـــر فاطمـــه عبـــدی

برخی از عناوین مهم

آشنایی با سیستم های توزیع شده
معماری های نرم افزار و فرآیندها
نام گذاری، همگام سازی، سازگاری و تکثیر
تحمل خطا، فایلهای توزیع شده و امنیت
حل تشریحی سوالات کنکور سراسری سال های ۹۱،۹۲،۹۳،۹۴
حل تشریحی سوالات کنکور دانشگاه آزاد سال های ۸۱ تا ۹۰ و ۹۳
حل تشریحی بیش از ۷۰۰ پرسش چهار گزینه ای
بیان حدود ۲۰۰ پرسش و مسئله و حل آن ها در سایت

درس و کنکور سیستم عامل پیشرفته

تألیف:

دکتر رمضان عباس نژادورزی
دکتر مرتضی بابازاده
دکتر فاطمه عبدی سقاواز

فن‌آوری نوین

سرشناسه	عباس‌نژاد ورزی، رمضان، ۱۳۴۸ -
عنوان و نام پدیدآور	درس و کنکور سیستم عامل پیشرفته/تالیف رمضان عباس‌نژادورزی ، مرتضی بابازاده ، فاطمه عبدی‌سقاوز.
مشخصات نشر	بابل: فناوری نوین ، ۱۳۹۶.
مشخصات ظاهری	۳۳۲ص.
شابک	۳۲۰۰۰۰ ریال : 978-600-7272-02-2
وضعیت فهرست نویسی	فیپا
موضوع	سیستم‌های عامل توزیع شده (کامپیوتر) -- راهنمای آموزشی (عالی)
موضوع	Distributed operating systems (Computers)-- Study and teaching (Higher)
موضوع	سیستم‌های عامل توزیع شده (کامپیوتر) -- مسائل ، تمرین‌ها و غیره(عالی)
موضوع	Distributed operating systems (Computers) -- Problems, exercises, etc.(Higher)
موضوع	سیستم‌های عامل توزیع شده (کامپیوتر)-- آزمون‌ها و تمرین‌ها (عالی)
موضوع	Distributed operating systems (Computers) Examinations, questions, etc. (Higher
شناسه افزوده	بابازاده، مرتضی، ۱۳۶۰ -
شناسه افزوده	عبدی‌سقاواز، فاطمه، ۱۳۵۶ -
رده بندی کنگره	۷۶/۹QA /پ۳۶ع۲ ۱۳۹۶
رده بندی دیویی	۳۶/۰۰۴
شماره کتابشناسی ملی	۴۸۷۲۳۷۸

www.fanavarienovin.net

بابل، کدپستی ۴۷۱۶۷-۷۳۴۴۸ تلفن: ۳۲۲۵۶۶۸۷-۰۱۱

درس و کنکور سیستم عامل پیشرفته

تألیف: رمضان عباس نژاد ورزی- مرتضی بابا زاده- فاطمه عبدی سقاواز

نوبت چاپ: چاپ اول

سال چاپ: پاییز ۱۳۹۶

شمارگان:

قیمت: ۳۲۰۰۰ تومان

نام چاپخانه و صحافی:

شابک: ۹۷۸-۶۰۰-۷۲۷۲-۰۲-۲

نشانی ناشر: بابل، چهارراه نواب، کاظم‌بیگی، جنب مسجد منصور کاظم‌بیگی، طبقه همکف

طراح جلد: کانون آگهی و تبلیغات آبان (احمد فرجی)

تهران، خ اردیبهشت، نبش وحید نظری، پلاک ۱۴۲ تلفکس: ۶۶۴۰۰۱۴۴-۶۶۴۰۰۲۲۰

فهرست مطالب

فصل اول: آشنایی با سیستم‌های توضیح شده

فصل دوم: معماری‌های نرم افزار

فصل سوم: فرآیند

فصل چهارم: ارتباطات

فصل پنجم: نام گذاری

فصل ششم: همگام سازی

مقدمه

امروزه با رشد رایانه و اینترنت، استفاده از سیستم‌های توزیع شده به خصوص رایانش ابری روز به روز در حال افزایش می‌باشد. به همین دلیل، سیستم های توزیع شده به عنوان یکی از دروس اصلی در دوره کارشناسی ارشد در وزارت علوم و دانشگاه آزاد اسلامی تدریس می‌شود. به طوری که در آزمون دکتری نرم‌افزار نیز از آن سوال می‌آید. به همین دلیل، کتاب حاضر تهیه گردید و در اختیار علاقه مندان قرار گرفته است.

این کتاب شامل مباحث زیر می‌باشد:

۱. مفاهیم سیستم‌های توزیع شده، معماری‌ها، فرآیندها.

۲. ارتباطات، نام گذاری، همگام‌سازی، تکثیر و سازگاری

۳. تحمل خطا، امنیت و مدیریت فایل در سیستم‌ها توزیع شده. البته تحمل خطا، امنیت و مدیریت فایل‌ها در سیستم های توزیع شده را می‌توانید از سایت انتشارات فناوری نوین دانلود کنید.

۴. حدود ۷۰۰ سوال چهار گزینه‌ای فراگیر و ترمی پیام نور با پاسخ تشریحی آن‌ها.

۵. سوالات دکتری سراسری سال‌های ۹۱، ۹۲، ۹۳، ۹۴ و سوالات دکتری آزاد سال ۹۴ به همراه پاسخ تشریحی.

۶. حدود ۱۸ سوال تشریحی آزمون دکتری دانشگاه آزاد اسلامی به همراه پاسخ آن‌ها

۷. حدود ۲۰۰ سوال تشریحی که پاسخ آن‌ها را می‌توانید از سایت انتشارات فناوری نوین به آدرس www.fanavarienovin.net دانلود کنید.

از تمامی اساتید و دانشجویان عزیز تقاضا داریم، هر گونه اشکال، ابهام در متن کتاب، پیشنهاد و انتقادات را به آدرس پست‌الکترونیک **fanavarienovin@gmail.com** ارسال نمایند.

در پایان امیدوارم این اثر مورد توجه جامعه انفورماتیک کشور، اساتید و دانشجویان عزیز قرار گیرد.

مولفین

fanavarienovin@gmail.com

فصل ۱ آشنایی با سیستم‌های توزیع‌شده

در این فصل به مفاهیم سیستم‌های توزیع‌شده[1] می‌پردازیم. اهداف سیستم‌های توزیع‌شده و انواع آن‌ها را موردبررسی قرار می‌دهیم. در پایان، انواع پیاده‌سازی سیستم‌های توزیع‌شده را بیان نموده، آن‌ها را با سیستم‌های متمرکز[2] مقایسه خواهیم کرد.

۱-۱. تعریف سیستم توزیع‌شده

سیستم توزیع‌شده، مجموعه‌ای از **کامپیوترهای مستقل و ناهمگن** است که از دید کاربر به‌صورت یک **سیستم واحد(منسجم)** به نظر می‌رسد. در این تعریف دو ویژگی بسیار مهم وجود دارد که عبارت‌اند از:

۱. از بعد سخت‌افزاری، ماشین‌ها مستقل و خودمختار هستند.

۲. از بعد نرم‌افزاری، کاربران فکر می‌کنند که با یک ماشین سروکار دارند. یعنی، از دید کاربر به‌عنوان یک سیستم واحد و منسجم به نظر می‌رسند.

یکی از مثال‌های سیستم توزیع‌شده، شبکه جهانی اینترنت است. در این شبکه علاوه بر این‌که ماشین‌ها به‌صورت مستقل و خودمختار می‌باشند، کامپیوترها و نرم‌افزارها می‌توانند ناهمگن باشند. از طرف دیگر، کاربر به‌سادگی با واردکردن نام **منحصربه‌فرد**[3] یا **یکتا** (نظیر www. fanavarienovin. net) می‌تواند به داده‌ها دست‌یابی داشته باشد، بدون این‌که لازم باشد، محل و آدرس واقعی داده را بداند.

در سیستم‌های توزیع‌شده، هر کامپیوتر سیستم‌عامل خاص خودش را دارد (ممکن است نوع سیستم-عامل‌ها در ماشین‌های مختلف متفاوت باشد). بنابراین، وظیفه واحد شدن (منسجم) سیستم‌ها را نمی‌توان بر عهده یک سیستم‌عامل قرار داد. پس سیستم‌های توزیع‌شده به‌منظور منسجم نمودن سیستم‌ها از ابزاری به نام **میان‌افزار**[4] استفاده می‌کنند (شکل ۱-۱). همان‌طور که در این شکل مشاهده می‌شود، میان‌افزار بین برنامه‌های کاربردی سیستم‌عامل‌ها قرار می‌گیرد تا کاربر از مسائلی که در سیستم‌های عامل و سخت‌افزارها

---

[1].Distributed Systems [2].Centralized Systems [3].URL [4].Middleware

اتفاق می‌افتد، اطلاعی نداشته باشد (بدون اطلاع باشد). درواقع، میان‌افزار جهت پشتیبانی از کامپیوترها و شبکه‌های غیر همگن است تا سیستم‌های مختلف ازنظر کاربر به‌عنوان یک سیستم واحد به نظر به رسند.

۲-۱. اهداف سیستم‌های توزیع‌شده

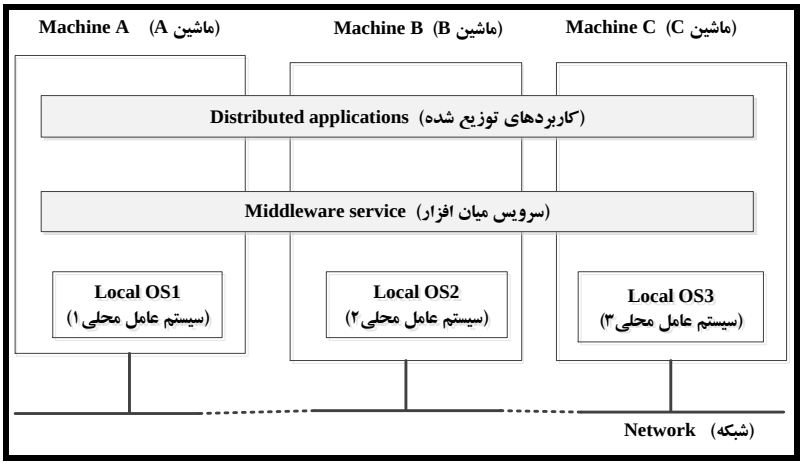
سیستم‌های توزیع‌شده اهداف زیر را دارند:

۱. برقراری ارتباط آسان بین کاربران و منابع (EasleyConnecting Users and Resources)

۲. شفافیت (Transparency)

۳. باز بودن سیستم (Openness)

۴. مقیاس‌پذیری (توسعه‌پذیری Scalability)



شکل ۱-۱ میان‌افزار و سیستم‌های توزیع‌شده.

۱-۲-۱. برقراری ارتباط آسان بین منابع و کاربران

یکی از اهداف اصلی سیستم‌های توزیع‌شده اتصال آسان بین کاربران و منابع راه دور و به اشتراک‌گذاری این منابع بین کاربران است (البته دسترسی از راه دور و به اشتراک‌گذاری باید کنترل‌شده باشد). به اشتراک‌گذاری منابع دلایل زیادی دارد که عبارت‌اند از:

- به اشتراک‌گذاری منابع موجب صرفه‌جوئی اقتصادی می‌شود. چون، کاربران به‌جای این‌که از چندین چاپگر استفاده کنند می‌توانند از یک چاپگر به‌صورت اشتراکی استفاده نمایند.
- امکان تبادل اطلاعات بین کاربران آسان‌تر خواهد شد. به‌عنوان‌مثال، تبادل اطلاعات از طریق اینترنت ساده می‌باشد.

از طرف دیگر، چون برخی از کارها ذاتاً به‌صورت توزیعی انجام می‌شوند (مانند شبکه بانک‌ها)، بااتصال منابع و کاربران این امکان فراهم می‌شود. البته باید دقت داشته باشید که هر چه استفاده اشتراکی از منابع افزایش می‌یابد، مسئله **امنیت** نیز مهم‌تر خواهد شد. یعنی، نباید اجازه داد تا افراد **غیرمجاز** به داده‌ها دست‌یابی داشته باشند. برای برقراری امنیت می‌توان علاوه بر استفاده از کلمه عبور جهت دست‌یابی از روش‌های رمزگذاری داده‌ها نیز استفاده نمود تا جلوی مشکلات امنیتی از قبیل استراق سمع و دزدیده شدن اطلاعات گرفته شود.

۲-۲-۱. شفافیت

یکی از اهداف مهم دیگر سیستم‌های توزیع‌شده، پنهان‌سازی این واقعیت است که **فرآیندها**[1] و منابع به‌صورت فیزیکی روی چندین کامپیوتر متعدد توزیع‌شده‌اند. درواقع، کاربران نباید متوجه این واقعیت شوند که فرآیندها (کارها) به‌صورت توزیعی بر روی چندین ماشین انجام می‌شوند. این مسئله **شفافیت** نام دارد. سیستم توزیع‌شده‌ای که بتواند این واقعیت را مخفی کند و خود را به کاربر و برنامه‌های کاربردی-اش، طوری نشان دهد که مانند یک کامپیوتر واحد به نظر آید، یک **سیستم شفاف**[2] نامیده می‌شود. شفافیت انواع مختلف دارد که عبارت‌اند از:

۱. شفافیت دسترسی[2]، همان‌طور که بیان گردید کامپیوترها در سیستم‌های توزیع‌شده می‌توانند ناهمگن باشند. این امر موجب می‌شود تا نمایش داده‌ها در کامپیوترهای مختلف متفاوت باشد. به‌عنوان‌مثال، کامپیوترهای نوع اینتل از فرمت (Little endian) برای تبادل داده استفاده می‌کنند. یعنی، اول بایت باارزش، سپس بایت کم‌ارزش انتقال می‌یابد. اما، کامپیوترها نوع Sun SPARC از فرمت (Big endian) برای تبادل اطلاعات استفاده می‌کنند (یعنی، ابتدا بایت کم‌ارزش، سپس بایت با ارزش انتقال می‌یابد). از طرف دیگر، چون منابع و کاربران با هم فاصله دارند، کاربران باید بتوانند از راه دور به منابع متصل شوند. **شفافیت دسترسی (دست‌یابی)**، تفاوت در نمایش داده‌ها و چگونگی دست‌یابی به منابع را از دید کاربران مخفی می‌نماید. بنابراین، سیستمی که شفافیت دسترسی را دارد، از دید کاربران تفاوتی

---

[1].Process [2].Transparence System
[2].Access Transparency [2].Location Transparency [3].Migration Transparency [4].Server
[5].Relocation Transparency [6].Laptop [7].Replication Transparency

در روش دسترسی به داده محلی و راه دور وجود ندارد و از دید کاربران چگونگی تبدیل و انتقال داده-های مختلف روی کامپیوترهای متفاوت پنهان است.

۲. شفافیت مکانی[۲]، مکان فیزیکی منابع را از دید کاربران مخفی می‌کند. یعنی، کاربران بدون این‌که محل واقعی منابع را بدانند می‌توانند به آن‌ها دسترسی داشته باشند. به‌عنوان‌مثال، آدرس www. google. comرا در نظر بگیرید. کاربران بدون این‌که آدرس واقعی آن را بدانند، به داده‌های آن دسترسی دارند. در این ساختار معمولاً آدرس‌ها و نام‌های منابع منطقی هستند. در فصل پنجم با انواع نام‌گذاری منابع در سیستم‌های توزیع‌شده آشنا خواهیم شد.

۳. شفافیت مهاجرت[۳]، انتقال منابع از یک مکان به مکان دیگر را از دید کاربر مخفی می‌نماید. به‌عنوان‌مثال، سرویس‌دهنده[۴] یاهو که در آمریکا قرار دارد، اگر به کانادا انتقال یابد، این انتقال از دید کاربر پنهان می‌ماند.

۴. شفافیت جابه‌جایی[۵]، انتقال منابع در حال استفاده از یک مکان به مکان دیگر را از دید کاربران پنهان می‌نماید. نمونه‌ای از شفافیت جابه‌جایی، استفاده کاربران سیار از لپ‌تاپ[۶] در حال انتقال از محلی به محل دیگر است. یعنی، این کاربران بدون قطع اتصال از طریق اتصال بی‌سیم می‌توانند از نقطه‌ای به نقطه دیگر منتقل شوند. فرق بین شفافیت مهاجرت و شفافیت جابه‌جایی این است که در شفافیت جابه‌جایی منبع در حال استفاده جابه‌جا می‌شود. **ایجاد شفافیت جابه‌جایی از ایجاد شفافیت مهاجرت پیچیده‌تر است.**

۵. شفافیت تکرار (تکثیر)[۷]، معمولاً چندین نسخه تکراری از یک داده یا منبع در سیستم‌های توزیع‌شده استفاده می‌شود تا **قابلیت اعتماد** و **قابلیت دسترسی** افزایش یابد. **شفافیت تکرار**، وجود چندین نسخه از داده یا منبع را از دید کاربران مخفی می‌کند و کاربران مانند زمانی عمل می‌کنند که تنها یک نسخه در سیستم وجود داشته باشد. قاعدتاً، سیستمی که شفافیت تکثیر را دارد باید شفافیت مکانی را نیز داشته باشد تا کاربران از محل واقعی داده‌ها بی‌خبر باشند. برای این منظور، باید تمام منابع تکثیرشده یک نام داشته باشند. زیرا، در غیر این صورت کاربران نمی‌توانند به کامپیوترهای مکان دیگر مراجعه کنند.

۶. شفافیت هم‌روندی[1]، یکی از اهداف سیستم‌های توزیع‌شده امکان استفاده مشترک و هم‌زمان منابع توسط کاربران است تا صرفه‌جویی اقتصادی حاصل شود. در این صورت، کاربران می‌توانند برای اخذ منابع با هم رقابت کنند. **شفافیت هم‌روندی**، این واقعیت را که منابع به‌صورت اشتراکی و هم‌زمان مورداستفاده قرار می‌گیرند و کاربران برای اخذ منابع با یکدیگر رقابت می‌کنند را از دید کاربران پنهان می‌کند. به‌عبارت‌دیگر، هر کاربر مانند این عمل می‌کند که منبع فقط در اختیار خودش است و سیستم موظف است که با قوانین خاصی داده‌ها را به‌روز (به هنگام) کرده و داده‌های صحیح را در اختیار کاربران دیگر قرار دهد. دو روش برای دست‌یابی به شفافیت هم‌روندی وجود دارد که عبارت‌اند از: ۱. روش قفل‌گذاری و دسترسی انحصاری به منابع ۲. استفاده از مفهوم تراکنش. در فصل هفتم روش‌های تکثیر داده را می‌آموزیم.

۷. شفافیت خرابی[2]، ترمیم[3]خرابی منابع را از دید کاربران مخفی می‌کند. یعنی، اگر سیستمی خراب شود، کاربر متوجه نشود که از طریق تکثیر (تکرار) قابل‌حل است. به‌عبارت‌دیگر، با خراب شدن یک منبع سیستم این قابلیت را داشته باشد تا با بقیه منابع موجود خرابی را پوشش دهد. مشکل عمده در شفافیت خرابی، عدم توانایی در تمایز بین منابع خراب (غیرفعال) و منابع فعال بسیار کند است. به‌عنوان‌مثال، زمانی که تماس به سرویس‌دهنده وب که ترافیک زیادی دارد، می‌رسد ممکن است فرصت زمانی (مهلت) مرورگر خاتمه یابد. در این صورت، گزارش می‌دهد که صفحه وب وجود ندارد. در اینجا کاربر نمی‌تواند نتیجه بگیرد که سرویس‌دهنده واقعاً خراب‌شده است یا به دلیل ترافیک، فرآیندها قادر به پاسخ‌گویی در فرصت زمانی تعیین‌شده نیست.

۸. شفافیت ماندگاری (پایداری)[4]، همان‌طور که می‌دانید داده‌ها در یک کامپیوتر بر روی حافظه قرار می‌گیرند. حافظه‌ها دو نوع‌اند:

- **حافظه اصلی**، داده‌های ناپایدار را نگهداری می‌کند که با قطع جریان برق محتوی حافظه پاک می‌شود.
- **حافظه جانبی**، داده‌های پایدار (ماندگار) را نگهداری می‌کند. این داده‌ها با قطع جریان برق پاک نمی‌شوند.

---

[1].Concurrency Transparency [2].Failure Transparency [3].Recovery [4].Persistence Transparency
[5].Object Oriented Database [6].Object [7].Openness [8].Syntax

۹. شفافیت پایداری، مکان ذخیره داده را از دید کاربر مخفی می‌نماید. به‌عبارت‌دیگر، کاربر کاری ندارد که داده کجا ذخیره می‌شود، بلکه سیستم تصمیم می‌گیرد که چه موقع داده‌ای در حافظه اصلی باشد یا در حافظه جانبی. به‌عنوان‌مثال، در بانک اطلاعاتی شی گرا[5] زمانی که متدی از یک شی[6] فراخوانی می‌شود باید این شی از حافظه جانبی به حافظه اصلی کپی گردد و این فرآیند کپی باید از دید کاربر پنهان بماند.

۳ - ۲-۱. باز بودن

سومین هدف سیستم‌های توزیع‌شده، **باز بودن**[7] است. یک سیستم توزیع‌شده باز، سیستمی است که سرویس‌ها را بر اساس قواعد و قوانین استانداردی ارائه می‌دهد که قواعد نحو[8] و معنایی آن سرویس‌ها، در سیستم‌های توزیع‌شده توصیف می‌گردد. یعنی، اگر بخواهیم نیازها یا سرویس‌های جدید ارائه کنیم، میان‌افزار باید بتواند سرویس‌های جدید را ارائه دهد. لازمه این کار این است که **یک‌زبان تعریف واسط (IDL)**[1] نظیر جاوا برای میان‌افزار استفاده شود. در IDL بیش‌تر به توصیف توابع قابل‌دسترسی، پارامترهای آن‌ها، حالت‌های استثنایی که ممکن است، اتفاق بی افتد و غیره پرداخته می‌شود. به‌عبارت‌دیگر، IDL اجازه می‌دهد که یک فرآیند دلخواه که به یک واسط نیاز دارد تا با فرآیند دیگری ارتباط برقرار کند، این واسط را تولید کند. در این صورت، برای یک برنامه کاربردی که نوشته می‌شود مهم نیست که بر روی سیستم‌عامل لینوکس اجرا می‌شود یا بر روی سیستم‌عامل ویندوز. جهت پشتیبانی از **باز بودن** باید تمام تعریف‌ها کامل و فارغ از پیاده‌سازی باشند. کامل بودن خیلی مشکل است. چون، طراحی یکسری مفروضاتی دارد و فردی که می‌خواهد سیستم را توسعه دهد ممکن است مفروضات دیگر داشته باشد. برای این‌که تعریف‌ها کامل و فارغ از پیاده‌سازی باشند باید ویژگی‌های زیر را داشته باشند:

۱. قابلیت انعطاف[2]، یعنی بتوان در سیستم از قطعات مختلف از سازندگان متفاوت استفاده کرد. همچنین بتوان به سیستم قطعه‌ای را اضافه کرد یا قطعه موجود را تعویض نمود. این عمل نباید در کار قطعات در حال کار اختلالی ایجاد نماید. به‌عبارت‌دیگر، سیستم باز باید، **بسط پذیر**[3] باشد.

---

[1].Interface Definition Language [2].Flexibility [3].Extensible [4].Portable [5].Scalability
[6].Size [7].Geographical [8].Administrative [9].Centralized Services [10].Centralized algorithms

۲. قابلیت حمل[4]، بدون این‌که تغییری در برنامه کاربردی ایجاد گردد، بر روی سیستم توزیع‌شده دیگری اجرا نمود (ممکن است سیستم‌های توزیع‌شده‌ای که برنامه کاربردی در آن نصب می‌شود و بر روی آن اجرا می‌شود، متفاوت و ناهمگن باشند).

۴-۲-۱. توسعه‌پذیری

یکی از اهداف بسیار مهم سیستم‌های توزیع‌شده، **توسعه‌پذیری (مقیاس‌پذیری)**[5] است. یعنی، به‌راحتی بتوان سیستم را توسعه و گسترش داد. توسعه‌پذیری را می‌توان در سه بعد بررسی نمود که عبارت‌اند از:

۱. توسعه‌پذیری ازلحاظ اندازه[6]، یعنی به‌راحتی بتوان تعداد منابع و کاربران سیستم را افزایش داد.

۲. توسعه‌پذیری ازلحاظ جغرافیایی[7]، یعنی بتوان فاصله فیزیکی کاربران و منابع را افزایش داد و کاربران و منابع از هم دور باشند.

۳. توسعه‌پذیری ازلحاظ سرپرستی[8]، یعنی سیستم با توجه به داشتن تعداد زیادی سازمان‌های اجرائی مستقل، بازهم به‌سادگی قابل اداره باشد.

متأسفانه با توسعه سیستم در یک یا چند بعد معمولاً کارایی آن کاهش می‌یابد. توسعه دادن سیستم مشکلات زیادی را ایجاد می‌نماید که برخی از آن‌ها در زیر آمده‌اند:

به‌عنوان‌مثال، در توسعه اندازه، اگر کاربران یا منابع زیادی را سرویس‌دهی کنیم با محدودیت‌های زیر مواجه خواهیم شد:

۱. محدودیت‌های سرویس‌های متمرکز[9]، اگر سرویس‌ها به‌صورت متمرکز روی یک سرویس‌دهنده قرار گیرند، با افزایش کاربران و منابع، این سرویس‌دهنده به یک **گلوگاه**[10] تبدیل خواهد شد که باعث **افزایش ترافیک** و **کاهش کارایی** می‌شود.

۲. محدودیت داده‌های متمرکز[1]، اگر داده‌ها به‌صورت متمرکز باشد، توسعه آن مشکل می‌باشد. به‌عنوان‌مثال، برای سازمان بزرگی مانند ثبت‌احوال قرار دادن تمام اطلاعات در یک جدول موجب ایجاد مشکل در توسعه می‌گردد. برای حل محدودیت‌هایی از قبیل سرویس‌ها و داده‌های متمرکز می‌توان از **تکرار** استفاده نمود. تکرار و تکثیر را در فصل هفتم می‌آموزیم.

---

[1].Centralized Data [2].Centralized algorithms

۳. محدودیت الگوریتم‌های متمرکز[۲]، الگوریتم‌های بهینه مسیریابی، تصمیم‌گیری را بر اساس اطلاعات تمام ماشین‌ها انجام می‌دهند. یعنی، یک ماشین اطلاعات سراسری را از تمام ماشین‌ها تحویل گرفته، عمل مسیریابی را انجام می‌دهد. در سیستم‌های توزیع‌شده بزرگ، تعداد زیادی پیام باید مسیریابی شوند. چون، این پیام‌ها برای مسیریابی باید به یک ماشین انتقال یابند، سربار زیادی را به سیستم تحمیل خواهند کرد. پس، برای بهبود توسعه‌پذیری باید از الگوریتم‌های نامتمرکز استفاده نمود. الگوریتم‌های نامتمرکز ویژگی‌های زیر را دارند:

- هیچ ماشینی اطلاعات کاملی در مورد وضعیت سیستم ندارد.
- ماشین‌ها فقط بر اساس ساعت محلی‌شان تصمیم می‌گیرند.
- خرابی یک ماشین موجب خرابی الگوریتم نمی‌گردد.
- هیچ تصوری برای وجود یک ساعت سراسری در سیستم وجود ندارد.

سه مورد اول، نیاز به توضیح ندارند. اما، مورد چهارم نیاز به توضیح دارد. چون در سیستم توزیع‌شده هیچ‌گاه نمی‌توان مطمئن شد که تمامی کامپیوترها(ماشین‌ها) ساعت یکسانی دارند. بنابراین، الگوریتم‌ها نباید بر اساس ساعت مشترک کار کنند. چون هر چه اندازه سیستم بزرگ‌تر می‌شود، احتمال اختلاف بین ساعت‌ها نیز بیش‌تر خواهد شد.

توسعه جغرافیایی نیز مشکلاتی دارد که عبارت‌اند از:

۱. مهم‌ترین مشکلی که در توسعه جغرافیایی یک سیستم توزیع‌شده اتفاق می‌افتد، تأخیر رسیدن بسته‌های اطلاعاتی زیادتر می‌شود. بنابراین، قابلیت اعتماد به صحت بسته‌های ارسالی پایین می‌آید.

۲. علاوه بر این فعلاً نمی‌توان سیستم‌های توزیع‌شده‌ای را که برای شبکه محلی طراحی‌شده‌اند، توسعه داد. زیرا، این شبکه‌ها مبتنی بر ارتباط **همگن** هستند. در این نوع ارتباط سرویس‌گیرنده سرویسی را درخواست کرده، تا زمانی که پاسخ را دریافت نکند متوقف می‌شود. از این روش نمی‌توان در سیستم‌های توزیع‌شده استفاده کرد. چون بسیار کند است.

مشکل اصلی توسعه سرپرستی تداخل سیاست‌های سازمان‌های اجرایی مختلف در سیستم است. هر چه تعداد سازمان‌های اجرایی مستقل بیش‌تری در سیستم وجود داشته باشد، احتمال به‌روز تداخل بین آن‌ها زیادتر خواهد شد.

۳-۱. تکنیک‌های توسعه‌پذیری

همان‌طور که بیان گردید، توسعه‌پذیری سیستم‌های توزیع‌شده ازلحاظ اندازه، جغرافیایی و سرپرستی مشکلات و محدودیت‌های را ایجاد می‌کند. برای بهبود مشکلات توسعه‌پذیری سیستم توزیع‌شده سه تکنیک وجود دارد که عبارت‌اند از:

۱. مخفی کردن تأخیرهای ارتباطات (Hiding Communication Latencies)

۲. توزیع (Distributed) ۳. تکثیر یا تکرار (Replication)

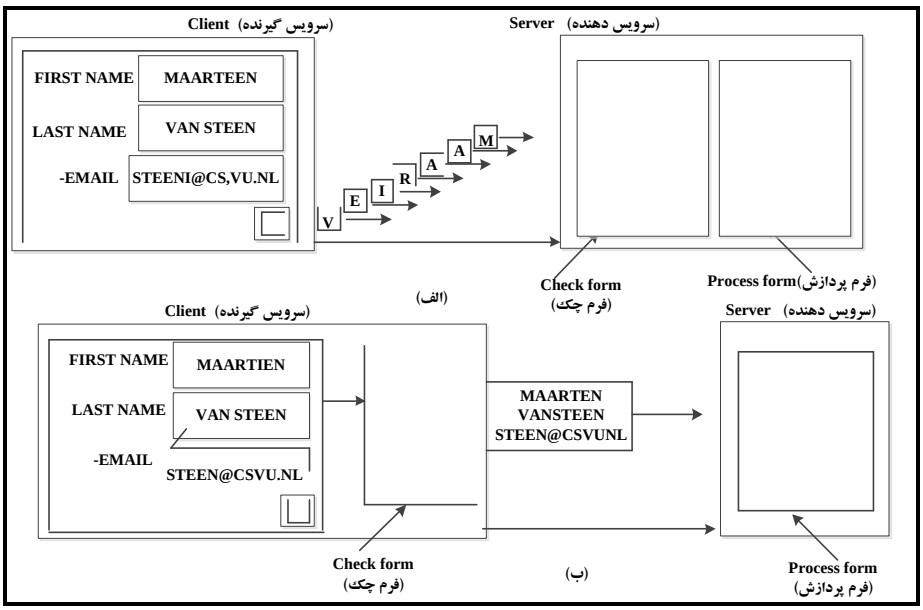
۱-۳-۱. مخفی کردن تأخیر ارتباطات

همان‌طور که بیان گردید، راه رسیدن به قابلیت توسعه جغرافیایی مناسب، پنهان کردن تأخیر ارتباطات است. یعنی، برنامه‌های کاربردی باید طوری نوشته شوند که فقط از **ارتباطات آسنکرون**[1] **(غیر هم-زمان)** استفاده کنند. در ارتباط غیر هم‌زمان، سرویس‌گیرنده درخواستی را به سرویس‌دهنده می‌فرستند، تا زمانی که درخواست پاسخ داده شود، به کارش ادامه می‌دهد و اعمال دیگر را انجام می‌دهد. وقتی یک پاسخ به سرویس‌گیرنده (درخواست‌کننده) می‌رسد، در کار آن وقفه ایجاد می‌شود و عملیات مربوط به درخواست قبلی کامل می‌شود. **ارتباط غیر هم‌زمان** در **سیستم‌های پردازش- دسته‌ای**[2] و کاربردهای **پردازش موازی**[3] مفید است. چون، در این کاربردها، تعدادی عمل که کم‌وبیش مستقل هستند، انجام می‌شوند و زمانی که در انتظار پاسخ سرویس‌دهنده است، کاربردهای دیگر می‌توانند به کارشان ادامه دهند.

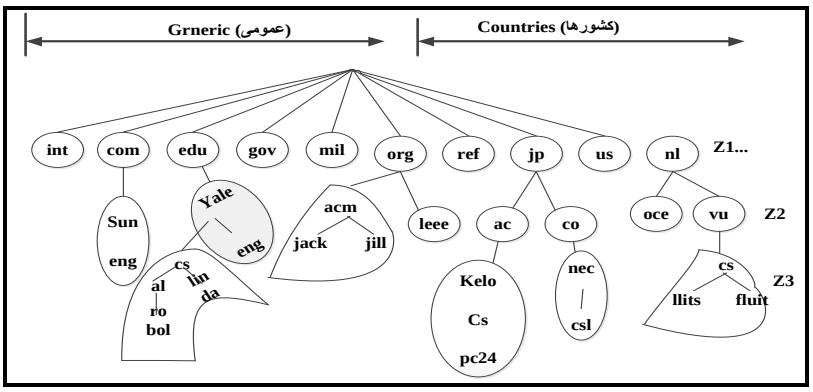
نمونه‌ای از پنهان کردن تأخیر ارتباط در شکل ۲ – ۱ آمده است. در بخش (الف) شکل، وقتی سرویس‌گیرنده درخواستی را از سرویس‌دهنده دارد، مدت‌زمان نسبتاً زیادی را باید منتظر پاسخ بماند. چون عمل کنترل فرم (Check form) سرویس‌دهنده انجام می‌شود و از طرف دیگر، تعداد سرویس‌گیرنده‌ها نیز زیاد است. درصورتی‌که همان‌طور که در بخش (ب) شکل ۲ -۱ می‌بینید، با قرار دادن عمل کنترل فرم (Check form) به عهده سرویس‌گیرنده‌ها بار سرویس‌دهنده‌ها کاهش می‌یابد و زمان پاسخ گوئی نیز کاهش خواهد یافت.

---

[1]. Asynchronous Communication [2].Batch Processing System [3].Parrallel Processing



شکل ۲ –۱. تفاوت بین تکمیل و کنترل فرم توسط سرویس‌دهنده (الف) و سرویس‌گیرنده (ب).



شکل ۳ – ۱ تقسیم فضای نام DNS به مناطق.

۳ – ۱۰ – ۱. سیستم‌های فراگیر توزیع‌شده

همان‌طور که در سیستم‌های توزیع‌شده محاسباتی و اطلاعاتی ملاحظه کردید، گره‌ها پایدار بودند و از طریق شبکه‌های مناسب به یکدیگر متصل بودند (یعنی، محل گره‌ها تقریباً ثابت بود، کیفیت اتصال و خرابی در سیستم معمولاً نرمال می‌باشد). اما در دنیای امروزی سیستم‌های متحرک و فراگیر (تعبیه‌شده)[۴] بسیار زیادی استفاده می‌شوند. این سیستم‌ها معمولاً **اتصالات** بی‌سیم دارند. متحرک هستند، کوچک

هستند، محاسبات کمی انجام می‌دهند و با باطری کار می‌کنند. پس بخش عظیم پیکربندی و ایجاد تغییرات وضیعتی آن‌ها باید به‌صورت خودکار انجام شود.

این سیستم‌ها دارای سه نیاز هستند:

۱. **پذیرفتن تغییرات موقعیتی**، یعنی باید بداند که محیط اطرافش به‌طور دائم در حال تغییرات و باید این تغییرات را تشخیص داده و عمل مناسب در برابر آن انجام دهد (مثل عدم دسترسی به شبکه در تلفن-های همراه).

۲. **تقویت ترکیب‌های ویژه**، یعنی دستگاه‌های متعدد توسط کاربران مختلف در این سیستم استفاده می‌شوند. برنامه‌های کاربردی که در این دستگاه‌ها اجرا می‌شوند، باید به‌راحتی توسط کاربران یا به‌صورت **خودکار پیکربندی**[۵] شوند (مانند پیکربندی GPRS در دستگاه‌های مختلف تلفن‌های همراه).

۳. **تشخیص اشتراک به‌صورت پیش‌فرض**، یعنی در اتصال دستگاه به سیستم که هدف اصلی آن خواندن و نوشتن داده‌ها است، فضای نوشتن و خواندن داده‌ها به‌صورت اشتراک مورداستفاده قرار می-گیرد. با توجه به این‌که اتصال متغیر است و متناوب در حال تغییر می‌باشد، این فضای اشتراکی در هر اتصال باید به‌درستی تشخیص داده شود. در سیستم‌های فراگیر توزیع‌شده، **ایجاد شفافیت سخت‌تر** از بقیه سیستم‌ها است و با توجه به سیار بودن دستگاه‌ها، گاهی بهتر است که برخی از شفافیت‌ها لحاظ نشود. چون موجب کاهش کارایی سیستم می‌شود. نمونه‌های از سیستم‌های فراگیر توزیع‌شده عبارت‌اند از:

- سیستم‌های خانگی (Home Systems)
- سیستم سلامت الکترونیکی (Electronic Health Care Systems)
- شبکه‌های حسگر (Sensor Networks)

سیستم‌های خانگی

در این نوع سیستم‌ها یک شبکه خانگی وجود دارد که سیستم‌های خانگی نظیر یک یا چند کامپیوتر شخصی، PDA، سیستم‌های صوتی و تصویری (تلفن و تلویزیون)، سیستم حرارتی، سیستم نظارت و نظایر آن در آن به هم متصل هستند و صاحب سیستم می‌تواند آن‌ها را از بیرون خانه و از طریق اینترنت کنترل و مدیریت کند. این سیستم‌ها باید دو ویژگی زیر را داشته باشند:

۱. چون محیط دائم در حال تغییر است، سیستم‌ها باید **خود پیکر بند**[1] باشند.

۲. سیستم‌ها باید خود را مرتب کند. یعنی ورود و خروج از سیستم اداره شود. به‌عبارت‌دیگر، بسیاری از کارها انجام می‌شوند تا دستگاه‌ها به‌طور خودکار و از طریق **استانداردهای نصب و اجرای جهانی** (upnp)[2] آدرس‌ها را به دست آورده و هم دیگر را پیدا می‌کنند.

سیستم‌های سلامت الکترونیکی

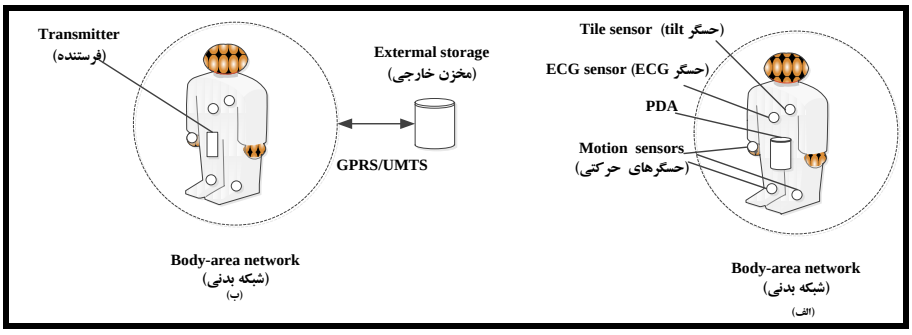
در این سیستم‌ها معمولاً یک سری حسگرهای مختلف در سطح بدن انسان توزیع می‌شوند تا سلامت بدن را کنترل و مانیتور کنند. این سیستم‌ها به‌طور خودکار سلامت افراد را نظارت می‌کنند و در صورت لزوم با پزشک تماس می‌گیرند. دو نوع این سیستم وجود دارد که عبارت‌اند از (شکل ۱۷-۱):

۱. **هاب‌دار**، در این نوع سیستم یک هاب وجود دارد که هرچند وقت به چند وقت اطلاعات را به کامپیوتر می‌دهد (هاب، می‌تواند مدیریت شبکه را نیز انجام دهد).

۲. **بدون هاب**، در سیستم فرستنده وجود دارد که گره‌ها اطلاعات را به فرستنده می‌فرستند. این شبکه، **شبکه ناحیه بدن (BAN)**[3] نام دارد.

از دیدگاه سیستم توزیع‌شده با سؤالات زیر مواجه هستیم:

- داده کجا باید ذخیره شوند.
- چگونه جلوی از دست رفتن داده‌های حیاتی را بگیریم.
- چه زیرساختی برای تولید و پخش هشدارها لازم است؟
- با توجه به این‌که شبکه ضعیف است، سیستم چگونه مقاوم و مستحکم باشد؟



[1].Self Managing [2].Universe of plug and play [3].body Area network

شکل ۱۷-۱ نظارت بر فرد در سیستم سلامت الکترونیکی فراگیر (الف) هاب دار (ب) اتصال بی‌سیم.

- امنیت چگونه برقرار می‌شود؟
- پزشک چگونه **بازخورد**[1] را به شخص بدهد؟

۱۱ - ۱. شبکه‌های حسگر

شبکه حسگر معمولاً شامل صدها و هزاران گره نسبتاً کوچک است. هر یک از گره‌ها مجهز به دستگاه حسگر هستند که درجه حرارت و عبور هر چیزی را حس می‌کنند. گره‌ها معمولاً دارای باطری هستند. محدودیت منابع و قابلیت ارتباطی (از طریق شبکه‌های بی‌سیم) و پایین بودن توان مصرفی در آن‌ها موجب شده است که در هنگام طراحی توجه ویژه‌ای به کارایی شود. در این شبکه دو وضعیت وجود دارد (در شکل ۱۸-۱ می‌بینید).

۱. گره‌ها با هم ارتباط ندارند و پردازش نیز انجام نمی‌دهند. اطلاعات را به یک کامپیوتر مرکزی می‌فرستند (شکل (الف)).

۲. گره‌ها توان پردازش و ذخیره محدود را دارند. سپس اطلاعات را به کامپیوتر مرکزی می‌فرستند (شکل ۱۸-۱(ب)).

داده حس شده توسط حسگرها به سه روش ارسال می‌شود:

- داده حس شده همراه[2] ارسال می‌شود.
- داده حس شده هرزمانی که نیاز باشد، ارسال می‌شود. یعنی مبتنی بر پرس‌وجو[3] است.
- وقتی عمل خاصی رخ دهد، داده حس شده ارسال می‌شود. یعنی مبتنی بر رویداد[4] است.

۱۲-۱. سؤالات چهارگزینه‌ای

۱. کدام مورد جزو مزیت‌های سیستم توزیع‌شده نسبت به یک سیستم مرکزی است(فراگیر - ۸۵).

الف: فاکتور $\frac{\text{قیمت}}{\text{کارایی}}$ (Price/ Performance) بالا، سرعت‌بالا، توزیع ذاتی برنامه‌ها، قابلیت اطمینان و امکان رشد سیستم.

---

[1].Feedback [2].Continues [3].Query Based [4].Event Based

ب: فاکتور $\frac{\text{قیمت}}{\text{کارایی}}$ (Price/ Performance) بالا، سرعت‌بالا، توزیع ذاتی برنامه‌ها، قابلیت اطمینان بالا و توسعه‌پذیری.

ج: فاکتور $\frac{\text{قیمت}}{\text{کارایی}}$ (Price/ Performance) بسیار کم، سرعت‌بالا، توزیع ذاتی برنامه‌ها، قابلیت اطمینان بالا و امکان رشد سیستم.

د: فاکتور $\frac{\text{قیمت}}{\text{کارایی}}$ (Price/ Performance) بسیار کم، سرعت‌بالا، توزیع ذاتی برنامه‌ها، قابلیت اطمینان بالا و توسعه‌پذیری (Scalability).

۲. کدام‌یک از مدل‌های زیر جامع‌تر و در پیاده‌سازی سیستم‌عامل توزیع‌شده به‌کاررفته است؟ (فراگیر -۸۵).

الف: SPMD ب:MIMD ج: MPMD د: MIMD

۳. به نظر شما مدل حافظه یک مالتی کامپیوتر و مدل حافظه یک مالتی پروسسور به ترتیب کدام است؟ (فراگیر -۸۵).

الف: UMA, UMA ب: UMA, NUMA

ج: NUMA, NUMA د: می‌توانند هر یک از موارد ۲ و ۳۲ باشد

۴. کدام مورد تعریف شفافیت (Transparency) صحیح است؟ (فراگیر– ۸۵).

الف: شفافیت به معنای آن است که خیلی از کارها به‌صورت ناپیدا از دید کاربر انجام شود.

ب:شفافیت مفهومی است که اگر در سیستم پیاده شود، موجب آسانی برنامه‌نویسی برای کاربر می‌شود.

ج:شفافیت مفهومی است که این امکان را به وجود می‌آورد که یک سیستم توزیع‌شده از دید کاربر به‌عنوان یک کامپیوتر واحد عمل نماید.

د: هر سه

۵. در یک مدیریت فایل توزیع‌شده شفافیت مکان(Location Transparency) به چه معنا است؟(فراگیر - ۸۵).

الف: کاربر از محل قرار گرفتن فایل موردنظرش اطلاعی ندارد.

ب: عملیات بدون اطلاع کاربر بتوانند به‌موازات هم دیگر انجام شوند.

ج: نسخه‌های متعددی از فایل موردنظر کاربر در ماشین‌های مختلف شبکه وجود داشته باشد.

د: فایل‌ها بدون آن‌که نامشان عوض شود، می‌توانند تحت صلاح‌دید مدیریت فایل از ماشینی به ماشین دیگر منتقل شوند.

۶. در کدام‌یک از انواع شفافیت (Transparency) در یک DOS، کاربرهای چندگانه می‌توانند منابع داده‌ای را به‌صورت خودکار به اشتراک بگذارند؟(فراگیر -۸۷).

الف: شفافیت مهاجرت (Migration Transparency)

ب:شفافیت موازی(Parallelism Transparency)

ج: شفافیت مکان (Location Transparency)

د: شفافیت هم‌زمانی (هم‌روندی) (Concurrency Transparency)

۷. کامل‌ترین تعریف از یک سیستم توزیع یافته (Distributed System)کدام است؟ (فراگیر -۸۸).

الف: مجموعه چند کامپیوتر که برای انجام هدف مشخص با هم در ارتباط هستند.

ب: مجموعه چند کامپیوتر که مستقل بوده، ولی از دید کاربر در حکم یک سیستم واحد هستند.

ج: مجموعه چند کامپیوتر همگن (Homogenous) است که می‌توانند چندین برنامه مختلف را اجرا نمایند.

د: مجموعه از چند کامپیوتر یا زیرسیستم است که در یک محدوده فیزیکی تعریف‌شده با هم به‌صورت مستقل از محیط بیرون در تقابل هستند.

۸. عبارت <<یک منبع می‌تواند توسط چندین کاربر مختلف در رقابت باشد>> معرف کدام از انواع شفافیت (Transparency) برای سیستم توزیع یافته است؟(فراگیر -۸۸).

الف: هم‌روندی(Concurrency) ب: تکرار (Replication)

ج: مهاجرت (Migration) د: جابه‌جایی(Relocation)

۹. این نکته که یک تراکنش(Transaction) به‌صورت کامل انجام می‌گیرد یا انجام نمی‌گیرد، تحت عنوان کدام ویژگی شناخته می‌شود (فراگیر -۸۸).

الف: یکپارچگی(Atomic) ب: ماندگاری (Durable)

ج: سازگاری (Consistency) د: انزوا(Isolation)

۱۰. کدام‌یک از موارد زیر، یک تعریف جامع در مورد سیستم‌های توزیع‌شده است؟ (فراگیر -۸۹).

الف: مجموعه‌ای از کامپیوترهای مستقل که از دید کاربر یک سیستم یکپارچه می‌باشد.

ب: مجموعه‌ای از کامپیوترهای وابسته که از دید کاربر به شکل یک سیستم متمرکز دیده می‌شوند.

ج: مجموعه‌ای از کامپیوترهای وابسته که هرکدام بخش مشخص از فرآیند را انجام می‌دهند.

د: مجموعه‌ای از کامپیوترهای مستقل که از دید کاربر می‌توانند به شکل یک یا چند سیستم یکپارچه دیده شوند.

۱۱. مخفی ماندن حرکت یک منبع به مکانی دیگر در هنگام استفاده، بیانگر کدام نوع شفافیت(Transparency) در سیستم‌های توزیع‌شده است؟ (فراگیر - ۸۹ و ۹۰).

الف: دست‌یابی(Access) ب: مکان (Location)

ج: مهاجرت (Migration) د: جابه‌جایی (Relocation)

۱۳-۱. پاسخ تشریحی سؤالات چهارگزینه‌ای

۱. گزینه (د) صحیح است. یکی از مزایای بسیار مهم سیستم‌های توزیع‌شده، توسعه‌پذیری آسان آن است. پس یکی از گزینه‌های (ب) و (د) صحیح هستند. از طرف دیگر در سیستم‌های توزیع‌شده نسبت قیمت به کارایی پایین است، پس گزینه (د) صحیح است.

۲. گزینه (د) صحیح است. پیاده‌سازی MIMD (چند دستورالعمل چند مسیر داده)، پردازنده‌های خودمختار چندگانه که هم‌زمان دستورالعمل‌های مختلفی را بر روی داده‌های چندگانه متفاوت اجرا می‌کنند. سیستم‌های رایانش توزیع‌شده عموماً بر روی این معماری است. معماری MIMD، به SPMD (تک برنامه- چند داده) و MPDM (چند برنامه- داده چندگانه) تقسیم می‌شود. پس MID و MPMD در پیاده‌سازی سیستم توزیع‌شده جامع‌تر هستند.

۳. گزینه (ب) صحیح است.

۴. گزینه (ج) صحیح است. گزینه‌های (د) و (ب) نادرست هستند. چون شفافیت، درواقع به معنی این است که کاربران نباید متوجه این واقعیت شوند که فرآیندها به‌صورت توزیعی بر روی چندین ماشین

اجرا می‌شوند. سیستم توزیع‌شده‌ای که بتواند این واقعیت را مخفی کند و خود را به کاربر و برنامه‌های کاربردی طوری نشان دهد که مانند یک کامپیوتر واحد به نظر آید، سیستم شفاف نامیده می‌شود.

۵. گزینه (الف) صحیح است. شفافیت مکان، مکان منابع را از دید کاربر مخفی می‌نماید. چون فایل یکی از منابع است، پس شفافیت مکان می‌تواند این باشد که کاربر از محل قرار گرفتن فایل (منبع) موردنظر اطلاعات ندارد.

۶. گزینه (د) صحیح است. چون شفافیت مهاجرت، انتقال منبع از یک مکان به مکان دیگر را مخفی می‌کند. شفافیت مکان، مکانی که منبع قرار دارد را از دید کاربر مخفی می‌نماید و شفافیت هم‌روندی، مشارکت منبع را بین چندین کاربر رقیب (بر سر اخذ منبع رقابت دارند) مخفی می‌کند.

۷. گزینه (ب) صحیح است. دو ویژگی بسیار مهم سیستم‌های توزیع‌شده اولاً مستقل (خودمختار) بودن کامپیوترها است. ثانیاً باید از دید کاربران به‌عنوان یک سیستم واحد (منسجم) عمل کنند.

۸. گزینه (الف) صحیح است. چون فقط شفافیت هم‌روندی، مشارکت منابع را بین کاربران مختلف و رقیب، مخفی می‌نماید. شفافیت تکرار، تکرار منابع را مخفی می‌نماید. شفافیت مهاجرت، انتقال منبع از یک محل به محل دیگر را مخفی می‌کند. اما، شفافیت جابه‌جایی انتقال منبع در حال استفاده (مانند لب تاپ) یا تلفن همراه را از یک محل به محل دیگر مخفی می‌کند.

۹. گزینه (الف) صحیح است. خاصیت یکپارچگی تراکنش به این معنی است که یا همه دستورات تراکنش باید با هم اجرا شوند یا هیچ‌کدام اجرا نشوند. یعنی، تراکنش قابل‌تقسیم نیست. خاصیت سازگاری، صحت انجام تراکنش‌ها را تضمین می‌کند. خاصیت ماندگاری، تضمین می‌کند اثر تراکنش-های نهایی شده (تثبیت‌شده) باید دائمی باشد و تحت هیچ شرایطی نباید اثرشان از بین برود. خاصیت انزوا، تضمین می‌کند که تراکنش‌ها به‌صورت هم‌روند اجرا شوند و اثر مخرب بر روی یکدیگر نداشته باشند.

۱۰. گزینه (الف) صحیح است. در سیستم‌های توزیع‌شده، کامپیوتر باید مستقل باشند(وابسته نیستند)، پس گزینه‌های (ب) و (ج) نادرست هستند. از طرف دیگر، این کامپیوتر باید به‌صورت یک سیستم منسجم(واحد) به نظر رسد که در گزینه (د) آمده است کامپیوترهای مستقل که از دید کاربر می-

توانند به شکل یک یا چند سیستم یکپارچه دیده شوند (اولاً **می‌توانند** نیست و **باید** درست است و ثانیاً یک یا چند سیستم نیست، بلکه یک سیستم منسجم است).

۱۱. گزینه (د) صحیح است. شفافیت دست‌یابی (Access)، تفاوت‌های نمایش‌ها و چگونگی دست‌یابی به منابع را مخفی می‌کند. شفافیت مکان (Location)، مکان فیزیکی وجود منبع را مخفی می‌کند. شفافیت مهاجرت (Migration)، انتقال منبع از یک محل به محل دیگر را مخفی می‌کند. شفافیت جابه‌جایی (Relocation) مانند شفافیت مهاجرت است. با این تفاوت که انتقال منبع در حال استفاده از یک محل به محل دیگر را مخفی می‌نماید.

۱۲. گزینه (الف) صحیح است. چون سیستم‌های خانگی، سلامت الکترونیک و شبکه‌های حسگر از نوع سیستم‌های فراگیر توزیع‌شده هستند. سیستم‌های محاسباتی خوشه‌ای و سیستم‌های محاسباتی شبکه‌ای از نوع سیستم‌های محاسباتی توزیع‌شده هستند. اما سیستم‌های پردازش تراکنش و جامعیت کاربرد شرکت از نوع سیستم‌های توزیع‌شده اطلاعاتی هستند.

فصل ۲ معماری‌های نرم‌افزار

همان‌طور که بیان گردید، سیستم‌های توزیع‌شده به لحاظ نرم‌افزاری از **قطعات**[۱] پیچیده‌ای از نرم‌افزار تشکیل‌شده‌اند که روی ماشین‌های مختلف توزیع‌شده‌اند. روش‌های مختلفی برای سازمان‌دهی این قطعات وجود دارد. چگونگی سازمان‌دهی قطعات را **معماری**[۲] مشخص می‌کند. **معماری نرم‌افزار**، روش سازمان‌دهی قطعات و ارتباط بین آن‌ها است. یکی از ویژگی سیستم‌های توزیع‌شده، **تطبیق‌پذیری**[۳] است. یعنی، سیستم توزیع‌شده باید به‌گونه‌ای طراحی شود که خودش را با تغییرات محیطی نظیر افزودن گره جدید، حذف گره، تغییر پروتکل و غیره وقف دهد. برای این منظور، نرم‌افزار می‌تواند خودش را پایش کرده و بر اساس آن واکنش مناسب نشان دهد.

معماری نرم‌افزار، با نمونه‌سازی و قرار گرفتن در کامپیوترهای واقعی تست می‌گردد تا به نسخه نهایی آن(یعنی معماری سیستم) برسیم. انواع معماری‌های سیستم‌های توزیع‌شده عبارت‌اند از: ۱. معماری متمرکز[۴]

۲. معماری نامتمرکز[۵] ۳. معماری ترکیبی[۶]

هدف اصلی سیستم‌های توزیع‌شده این بود که از طریق لایه **میان‌افزار** کاربردها را از پلات فرم‌ها جدا نموده تا شفافیت توزیع به دست آید. بنابراین، **هدف اصلی معماری شفافیت توزیع است**.

۱-۲. سبک معماری

سبک معماری[۷]، شامل قطعات، روش اتصال قطعات، چگونگی تبادل داده‌ها و روش پیکربندی آن‌ها به یکدیگر جهت تشکیل یک سیستم منسجم و واحد است. در این تعریف دو مفهوم زیر وجود دارد:

- **قطعه**، یک واحد **پیمانه‌ای**[۸] است که با واسط‌های ضروری و آماده شده‌ای همراه است و در محیط خود قابلیت جایگزینی دارد.

---

[1].Components [2].Architecture [3].Adaptability [4].Centralized Architecture [5].Decentralized Architecture [6].Hybrid Architecture [7].Architetur Style [8].Modular [9].Connector

اتصال‌دهنده یا کانکتور[9]، راهکاری است که ارتباط، هماهنگی و همکاری بین قطعات را فراهم می‌کند. به‌عنوان‌مثال، مبادله پیام‌ها یا جریان داده‌ها (که در فصل ۴ شرح آن‌ها را می‌بینید) می‌تواند از طریق یک کانکتور انجام شود. سبک‌های موجود معماری سیستم توزیع‌شده عبارت‌اند از:

۱. لایه‌ای(Layered)　　۲. مبتنی بر شیء (Object Based)

۳. مبتنی بر داده مرکزی (Data Centered)　　۴. مبتنی بر رویداد (Event Based)

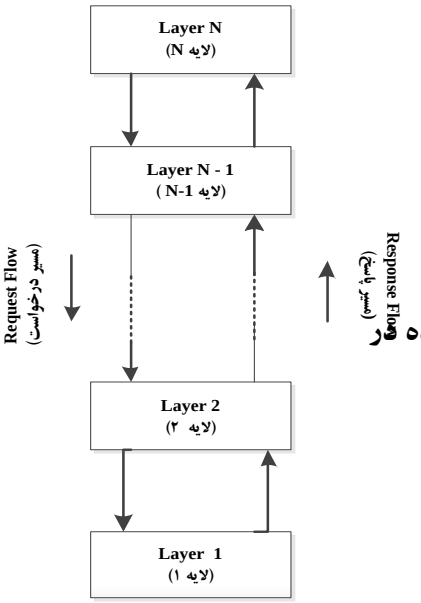
۱-۱-۲. معماری لایه‌ای

در این معماری، چندین لایه وجود دارد که قطعات در لایه‌ها قرار دارند. یک قطعه در لایه iام تنها می‌تواند قطعه لایه i-1ام را فراخوانی نماید (شکل ۱-۲). این معماری دارای ویژگی‌های زیر است:

۱. شماره گذاری لایه‌ها معمولاً از پایین به بالا است.

۲. درخواست‌ها از لایه‌های بالا به سمت لایه‌های پایین حرکت می‌کنند.

۳. نتایج درخواست‌ها از لایه پایین به سمت لایه‌های بالا حرکت می‌کنند.



معماری لایه‌ای، یک معماری پذیرفته‌شده در شبکه‌های کامپیوتری نیز است.

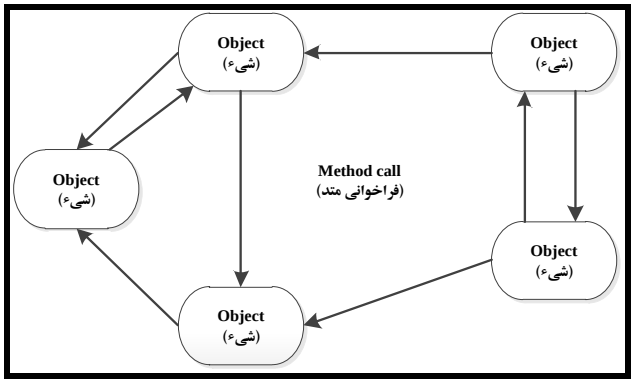
۲-۱-۲. معماری مبتنی بر شیء

این معماری سازمان‌بندی ضعیف‌تری نسبت به معماری لایه‌ای دارد. در این معماری اشیاء[1] متناظر قطعات در سیستم لایه‌ای هستند که در سیستم پراکنده‌شده‌اند. این اشیا از طریق **فراخوانی رویه**[2] **(متد)** که ممکن است از راه دور نیز باشد، با هم ارتباط برقرار می‌کنند. این معماری را در شکل ۲ – ۲ می‌بینید.

شکل ۱-۲ سبک معماری لایه‌ای.

[1].Objects　　[2].Share Distributed File Systems

معماری‌های لایه‌ای و مبتنی بر شیء مهم‌ترین سبک‌ها برای سیستم‌های نرم‌افزاری بزرگ هستند.



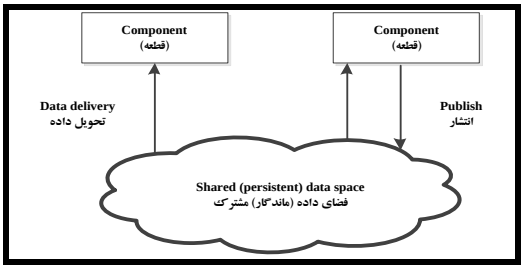
شکل ۲-۲ سبک معماری مبتنی بر شیء.

۳-۱-۲. معماری مبتنی بر داده مرکزی (داده محور)

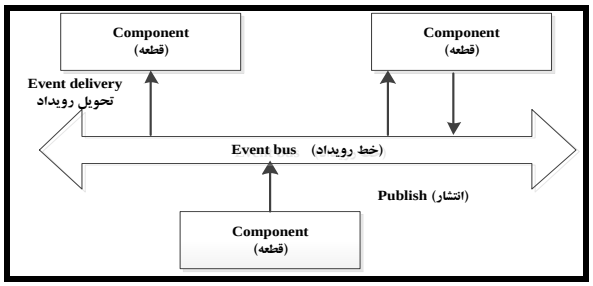
اساس معماری مبتنی بر داده مرکزی این است که فرآیندها از طریق یک فضای داده مشترک با هم ارتباط برقرار می‌کنند. نمونه‌ای از این سیستم‌ها عبارت انداز:

۱. سیستم‌های توزیع‌شده مبتنی بر وب(شکل ۳-۲)، در این سیستم‌ها فرآیندها از طریق سرویس‌دهنده‌های مبتنی بر وب مشترک با یکدیگر ارتباط برقرار می‌نمایند.

۲. سیستم‌های فایل توزیع‌شده مشترک[۲]



شکل ۳ -۲ سبک معماری فضای داده مشترک.



شکل ۴ -۲ سبک معماری مبتنی بر رویداد.

۴-۱-۲. معماری مبتنی بر رویداد

در این معماری فرآیندها از طریق پخش رویدادها با یکدیگر ارتباط برقرار می‌کنند. در این ساختار هر رویداد تعدادی عضو دارد. به‌عبارت‌دیگر، تعدادی فرآیند، عضو یک رویداد هستند. اکنون اگر یکی از فرآیندها رویدادی را منتشر کند، میان‌افزار تضمین می‌کند که فقط اعضای آن رویداد آن را دریافت خواهند کرد. به این سیستم **انتشار/ عضویت** گفته می‌شود. پس لازم نیست که فرآیندها صریحاً به هم مراجعه کنند. اصطلاحاً گفته می‌شود که فرآیندها **اتصال ضعیف** دارند. این وضعیت را **انفصال ارجاعی** نیز می‌نامند. شکل ۴-۲ نمونه‌ای از معماری مبتنی بر رویداد را نشان می‌دهد.

۳-۲-۲. معماری ترکیبی (مختلط)

بسیاری از سیستم‌های توزیع‌شده، معماری سرویس‌گیرنده/ سرویس‌دهنده (معماری متمرکز) و معماری همتا به همتا (نامتمرکز) را با هم ترکیب می‌کنند تا از مزایای آن‌ها استفاده کنند. در این بخش دو نوع از سیستم‌های ترکیبی را می‌بینید که عبارت‌اند از:

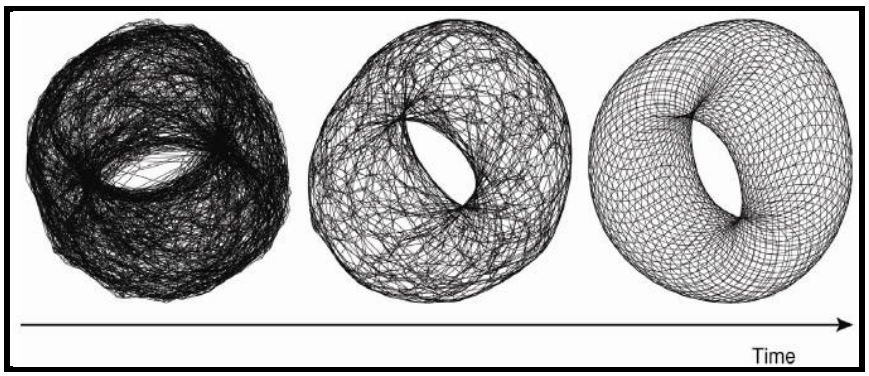
۱. سیستم‌های سرویس‌دهنده لبه (Edge-Server Systems)

۲. سیستم‌های همکاری کننده توزیع‌شده (Collaborative Distributed System)

سیستم‌های سرویس‌دهنده لبه

در سیستم‌های سرویس‌دهنده لبه معمولاً در اینترنت مستقر هستند. سرویس‌دهنده‌ها در لبه شبکه قرار می‌گیرد(شکل ۱۴-۲). به‌عنوان‌مثال، شرکت تأمین‌کننده خدمات اینترنت (ISP)، در لبه اینترنت قرار دارند. زیرا، ازیک‌طرف به اینترنت متصل می‌باشند و از طرف دیگر به کاربران نهایی در منازل، فروشگاه‌ها، سازمان‌ها و شرکت‌ها سرویس می‌دهند.

مجموعه‌ای از این سرویس‌دهنده‌ها می‌توانند برای بهینه‌سازی محتوا و توزیع برنامه‌های کاربردی استفاده شوند. در این سیستم ارتباط گره‌ها یا کاربران به‌صورت سرویس‌گیرنده / سرویس‌دهنده است و لبه‌ها (سرویس‌دهنده‌های لبه) با هم ساختار همتا به همتا را دارند. در این سیستم، سرویس‌دهنده‌های لبه (گره‌های لبه)، اطلاعات کاربران را دارند.



شکل ۱۳-۲ ایجاد یک شبکه هم‌پوشان خاص با استفاده از سیستم همتا به همتای غیر ساخت‌یافته دولایه‌ای.

سیستم‌های همکاری کننده توزیع‌شده

در سیستم‌های توزیع‌شده همکاری کننده آغاز به کار با یک ساختار سرویس‌گیرنده/ سرویس‌دهنده سنتی است. اما، در ادامه بااتصال گره‌ها به سیستم از ساختار نامتمرکز برای همکاری استفاده می‌شود.

مثالی از این نوع سیستم می‌توان سیستم‌های اشتراک داده در اینترنت به نام Bit Torrent را نام برد. در این سیستم برای جست‌وجو از یک ساختار متمرکز (مرکزی) استفاده می‌شود. ولی، برای دانلود از یک سیستم همتا به همتا استفاده می‌گردد. در این سیستم ابتدا یک LookUp(F) انجام می‌شود که روی صفحه وب Bit Torrent انجام می‌گردد. این LookUp، به یک **سرویس‌دهنده فایل**[1] ارجاع می‌دهد. در درون این سرویس‌دهنده برای فایل F فایل Torrent است که شامل اطلاعات فایل F می‌باشد. ازجمله اطلاعات، ارجاع به **ردیاب**[2] است. ردیاب، یک سرویس‌دهنده است که دارای لیستی از N گره است که فایل F را ذخیره کرده‌اند. وقتی اولین گره این اطلاعات را دریافت نموده، شروع به تماس با سایر گره‌ها و تبادل اطلاعات می‌کند. یعنی، یکی از گره‌ها، قطعاتی از فایل را که ندارند، برای دیگری ارسال می‌کنند. واضح است که در ابتدای کار گره سرویس‌گیرنده فقط دریافت‌کننده قطعات است. اما پس از دریافت چند قطعه خود نیز به دیگران سرویس می‌دهد.

اگر دو گره مثل P و Q قطعات را با هم تبادل می‌کنند، درصورتی‌که یک گره متوجه شود که گره دیگری با نرخ کم‌تری داده ارسال می‌کند، نرخ ارسال داده‌های خودش را نیز کم می‌کند.

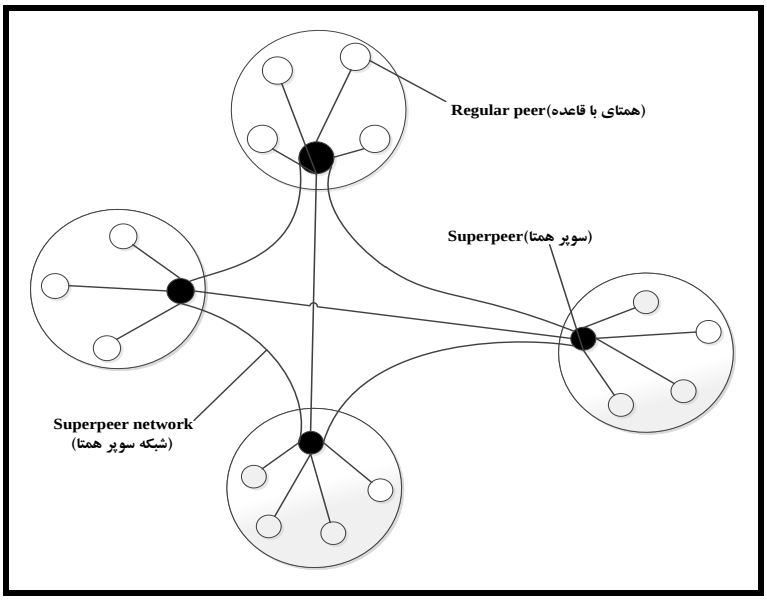
---

[1].File Server [2].Tracker

در این سیستم راه کار متمرکز و نامتمرکز ترکیب‌شده‌اند. چون ردیاب‌ها، سرویس‌دهنده‌های متمرکز هستند، درحالی‌که گره‌های دیگر به‌صورت همتا به همتا عمل می‌کنند.

معماری‌های در مقایسه با میان‌افزار

همان‌طور که بیان گردید، میان‌افزار بین برنامه‌های کاربردی و سیستم‌عامل‌های محلی توزیع‌شده است که مهم‌ترین هدف آن ایجاد شفافیت بود. در عمل میان‌افزار از یک سبک معماری نظیر شی گرا، رویداد گرا و غیره پیروی می‌کند. **به‌عنوان‌مثال، میان‌افزار CORBA از سبک معماری شی گرا پیروی می‌کند و میان‌افزار TIBCO از معماری رویدادگرا استفاده می‌نماید.**



شکل ۱۴-۲ سازمان‌دهی سلسله مراتبی گره‌ها در شبکه سوپر همتا.

مزیت مدل‌سازی میان‌افزار بر اساس معماری خاص این است که طراحی برنامه کاربردی را ساده‌تر می-کند. اما، عیبش این است که ممکن است برای طراح برنامه کاربردی ایده آل نباشد. به‌عنوان‌مثال، میان‌افزارCORBA در ابتدا تنها اشیا قابل فراخوانی را دور را پشتیبانی می‌کرد. پس از مدت‌زمانی مشخص گردید که طراحی برنامه‌های کاربردی بر اساس این نوع اشیا نمی‌تواند تمامی نیازها را پاسخ دهد. بنابراین، الگوریتم‌های تعاملی دیگری نظیر ارسال پیام نیز به آن اضافه گردید. این امر موجب بالا رفتن حجم میان‌افزار گردید. راه‌حل دیگر، ایجاد چندین نسخه از میان‌افزار برای کاربردهای متعدد بود. اما، بهترین راه‌کار، ساخت میان‌افزار واحدی که به‌راحتی قابل پیکربندی، سازگاری و تغییر باشد.

ره‌گیری

ره‌گیر[1]، یک ساختار نرم‌افزاری است که می‌تواند اجرای برنامه (مجموعه دستورات) را قطع کند و اجازه می‌دهد تا کد به خصوصی اجرا شود. برای درک بهتر ره‌گیر، سیستم‌های توزیع‌شده مبتنی بر شیء را در نظر بگیرید. همان‌طور که بیان گردید، در سیستم‌های مبتنی بر شی، شی A می‌تواند متد شیB را که در یک ماشین راه دور قرار دارد، فراخوانی نماید. برای این منظور، سه مرحله زیر انجام می‌شود:

۱. به شی A یک واسط محلی ارائه می‌گردد (این واسط مشابه همان واسطی است که به شی B ارائه می‌گردد). شی A متد موجود در آن واسط را فراخوانی می‌نماید. این مرحله به شکل زیر انجام می‌شود:

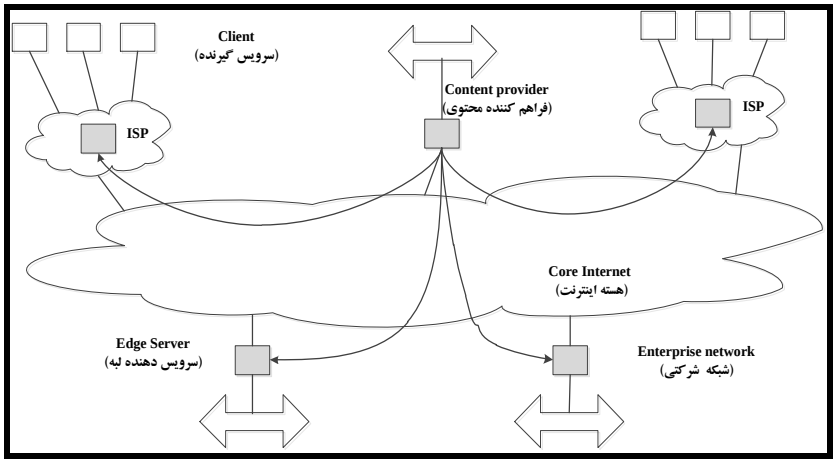
B. DO-SOMETHING(VALUE)

۲. فراخوانی توسط A توسط میان‌افزار موجود در ماشینی که A در آن قرارگرفته است، به یک فراخوانی کلی شی تبدیل می‌شود. این عمل زیر انجام می‌شود:

INVOKE (B, &DO-SOMETHING, VALVE)

۳. این فراخوانی کلی مرحله ۲ به یک پیام تبدیل‌شده و از طریق شبکه به‌سوی ماشین حاوی B ارسال می‌گردد. این عمل به شکل زیر انجام می‌شود:

SEND ([B, “DO-SOMETHNG”, VALUE])



شکل ۱۵-۲ نگاهی به اینترنت به‌عنوان مجموعه‌ای از سرویس‌دهنده‌های لبه‌ای.

حال ممکن است که B چندین نسخه تکراری داشته باشد، لازم نیست که A از این مسئله اطلاعی داشته باشد. از طرف دیگر، لازم نیست که میان‌افزار مرتبط باشی دارای قطعات خاصی برای کار با اشیا

[1].Interceptor

تکرار شده باشد. کافی است که یک ره‌گیر سطح به نام ره‌گیر سطح درخواست فراخوانی را برای هر کپی که صلاح دانست یا تمامی نسخه‌های کپی، ارسال کند. این ره‌گیر یک قطعه نرم‌افزاری است که می‌تواند به میان‌افزار اضافه شود.

۴-۲. سؤالات چهارگزینه‌ای

۱. تعریف زیر مربوط به کدام نوع معماری سیستم‌های توزیع‌شده می‌باشد؟ (فراگیر - مرداد ۸۸).

<<نوعی از معماری است که در آن هر جزء دارای وظایف مشخصی بوده و از اجزای مختلف با استفاده از مکانیزم‌های RPC با هم دیگر در ارتباط می‌باشند>>

الف: معماری لایه‌ای (Layered Architecture)

ب: معماری مبتنی بر شیء (Object–BaseArchitecture)

ج: معماری داده متمرکز (Data – Centered Architecture)

د: معماری مبتنی بر رویداد (Event – Based Architecture)

۲. معماری سه لایه‌ای سرویس‌دهنده – سرویس‌گیرنده (Three- Tiered Client - Server) چگونه می‌باشد (فراگیر–اردیبهشت۸۹). << لایه بالایی به‌عنوان.... ، لایه میانی به‌عنوان..... و لایه پایینی به‌عنوان..... می‌باشد>>

الف: واسط کاربر- سرویس‌دهنده داده (Data Server) – کاربردی واقعی (Actual Application)

ب: واسط - کاربردی واقعی (Actual Application) - سرویس‌دهنده داده (Data Server)

ج: سرویس‌دهنده داده (Data Server) - کاربردی واقعی (Actual Application) – واسط کاربر

د: واسط کاربر و کاربردی واقعی (Actual application) - سرویس‌دهنده داده (Data Server)- کاربردی واقعی (Actual Application)

۳. سیستم‌های توزیع‌شده بر مبنای وب (Web – Based Distributed Systems)، چه نوع معماری دارند؟ (فراگیر - اردیبهشت ۹۰ و تابستان ۹۰).

الف: مبتنی بر شی (Object - Based)

ب: داده متمرکز (Data – Centered)

ج: مبتنی بر رویداد (Event – Based)

د: مبتنی بر لایه (Layered – Based)

۴. در کدام نوع معماری سیستم‌های توزیع یافته، اجزاء به کمک یک مکانیزم فراخوان پردازه (Procedure Call) با هم در ارتباط هستند؟ (فراگیر - دی ۹۰).

الف: لایه‌ای (Layered)　　ب: مبتنی بر شیء (Object - Based)

ج: داده متمرکز (Data Centered)　　د: مبتنی بر رویداد (Event – Based)

۵. در معماری یک سیستم توزیع یافته، در بالاترین لایه، یک واسط کاربر سرویس‌گیرنده، در لایه میانی یک کاربرد (Application) و لایه پایین، پایگاه داده در نظر گرفته‌شده است. این نوع معماری در کدام دسته از معماری‌های زیر واقع می‌شود؟ (فراگیر - دی ۹۰).

الف: متمرکز (Centered)　　ب: هیبرید (Hybrid)

ج: نظیر به نظیر (Peer – To - Peer)　　د: نامتمرکز (Decentralized)

۶. از معماری سیستم‌های توزیع یافته، کدام عبارت در مورد توزیع عمودی و توزیع افقی صحیح می‌باشد؟ (فراگیر - اردیبهشت ۹۰).

الف: در توزیع عمودی، هر لایه روی یک ماشین متفاوت تحقق می‌یابد.

ب: در توزیع افقی، یک تک لایه روی چندین ماشین پیاده ساری می‌شود.

ج: در هر دو نوع توزیع، لایه‌های مختلف قابل‌انتقال از یک ماشین به ماشین دیگر می‌باشد.

د: موارد (الف) و (ب)

۵-۲. پاسخ تشریحی سؤالات چهارگزینه‌ای

۱. گزینه (ب) صحیح است. در این نوع معماری، اشیا متناظر در سیستم پراکنده هستند و هر یک از اشیا وظیفه مشخص دارند. این اشیا با فراخوانی رویه‌ای راه دور (RPC) ممکن است با هم دیگر ارتباط داشته باشند.

۲. گزینه (ب) صحیح است. چون سه لایه به ترتیب از بالا به پایین عبارت‌اند از: ۱. لایه واسط کاربر، معمولاً توسط سرویس‌گیرنده‌ها پیاده‌سازی می‌شود و وظیفه آن ارتباط با کاربر است. ۲. سطح پردازش (کاربرد واقعی)، سطح میانی است که وظیفه آن انجام عملیات اصلی برنامه کاربردی است. ۳. سطح داده (سرویس‌دهنده داده)، در پایین‌ترین لایه قرار دارد و وظیفه آن نگهداری داده‌های واقعی پایدار است.

۳. گزینه (ب) صحیح است. معماری داده متمرکز بر این ایده استوار است که فرآیندها از طریق یک منبع مشترک ارتباط برقرار می‌کنند. از طرف دیگر، در سیستم‌های توزیع‌شده مبتنی بروب، فرآیندها از طریق سرویس داده‌های مبتنی بر وب مشترک، با یکدیگر ارتباط برقرار می‌نمایند (سیستم‌های فایل توزیع‌شده مشترک (Share Distributed File Systems)، نمونه دیگری از معماری متمرکز داده است).

۴. گزینه (ب) صحیح است. چون در معماری مبتنی بر شیء اشیا اجزای تشکیل‌دهنده سیستم هستند. این اشیا از طریق فراخوانی رویه‌ها که ممکن است راه دور نیز باشد، با هم ارتباط برقرار می‌کنند.

۵. گزینه (الف) صحیح است. این تعریف، معماری لایه‌ای را شرح داده است که معماری سه لایه‌ای سرویس‌گیرنده و سرویس‌دهنده از نوع معماری متمرکز می‌باشد.

۶. گزینه (د) صحیح است. چون در توزیع عمودی مانند دولایه‌ای و سه لایه‌ای هر لایه روی یک ماشین متفاوت تحقق می‌یابد. اما، در توزیع افقی سرویس‌گیرنده یا سرویس‌دهنده ممکن است ازلحاظ فیزیکی به بخش‌هایی که ازلحاظ منطقی برابر هستند، تقسیم شوند. این عمل موجب می‌شود تا توازن بار ایجاد شود. نمونه‌ای از این سیستم می‌توان معماری نظیر به نظیر (Peer To Peer) را نام برد. در این معماری، کارهایی که باید توسط سیستم توزیعی ارائه شوند، از طریق هر یک از فرآیندها ارائه می‌گردند. ازآنجایی‌که در این مدل فرآیندها برابر هستند، پس گزینه (ب) نیز صحیح است.

۷. گزینه (د) صحیح است. گزینه (ب) نادرست است. چون، این گزینه از ویژگی‌های توزیع افقی (مدرن) است که در آن یک لایه نظیر سرویس‌گیرنده یا سرویس‌دهنده ممکن است به‌طور افقی در عرض چندین ماشین تقسیم شود. اما، در توزیع عمودی هر لایه می‌تواند بر روی یک ماشین متفاوت قرار گیرد. پس، گزینه‌های (الف) و (ج) ویژگی‌های توزیع عمودی هستند.

فصل ۳ فرآیندها

همان طور که در سیستم عامل بیان گردید، فرآیند یک برنامه در حال اجرا است که از طریق بلوک کنترلی فرآیند (PCB[1]) در سیستم عامل شناخته می‌شود. یک فرآیند شامل سه بخش زیر است:

۱. **کد برنامه**، شامل دستوراتی است که برنامه باید اجرا کند.

۲. **پشته فرآیند**، داده‌های موقتی نظیر پارامترهای توابع، آدرس‌های بازگشت و تغییرات موقتی را نگه داری می‌کند.

۳. **بخش داده‌ای**، متغیرهای سراسری را شامل می‌شود.

هر فرآیند در سیستم عامل با یک بلوک کنترلی فرآیند خودش نشان داده می‌شود. یک بلوک کنترلی فرآیند(PCB)، یک بلوک داده‌ای می‌باشد که اطلاعاتی از قبیل **اطلاعات هویتی فرآیند** (نام، شماره شناسایی فرآیند و فرآیند پدر)، **وضعیت فرآیند** (آماده، اجرا، منتظر)، **شمارنده برنامه** (آدرس دستور بعدی که باید اجرا شود)، **ثبات پردازنده** (تمام اطلاعات وضعیتی برنامه که در هنگام رخ دادن یک وقفه نگهداری می‌شود)، **اطلاعات زمان‌بندی پردازنده‌ها** (پارامترهای زمان بندی و الویت فرآیندها)، **اطلاعات مدیریت حافظه** (شامل تمام اطلاعات مربوط به حافظه و فرآیند مربوطه) و **اطلاعات وضعیت ورودی / خروجی**(شامل لیستی از فایل‌های باز، دستگاه‌های ورودی / خروجی تخصیص داده شده به این فرآیند) را دارد.

برای اجرای برنامه، سیستم تعدادی فرآیند مجازی[2] ایجاد می‌کند که هر کدام برای اجرای برنامه‌های مختلفی به کار می‌رود. جهت ردیابی این فرآیندهای مجازی سیستم عامل جدول فرآیند[3] را دارد که شامل وارده‌هایی برای ذخیره مقادیر ثبات CPU، نقشه‌های حافظه، فایل‌های باز، اطلاعات حساب‌رسی (همان اطلاعات PCB) است. فرآیند معمولاً برنامه در حال اجرا است (یعنی، برنامه‌ای که در یکی از فرآیندهای مجازی سیستم عامل در حال اجرا است). سیستم عامل مراقبت می‌کند که فرآیندهای مستقل نتوانند به

---

[1].Process Control Block [2].Virtual Process [3].Process Table [4].Cache
[5].Translation Look Aside Buffer

صحت یک دیگر آسیب بزنند. یعنی، جهت استفاده هم‌زمان فرآیندها از منابع باید شفافیت ایجاد شود، ایجاد این شفافیت، هزینه بالایی دارد. به عنوان مثال، برای تعویض CPU بین دو فرآیند علاوه بر ذخیره اطلاعات PCB، سیستم عامل باید ثبات‌های واحد حافظه را اصلاح نماید و حافظه نهان[۴] ترجمه آدرس نظیر TLB[۵] را نامعتبر کند.

قبل از ادامه بحث در مورد فرآیند و نخ به چند مفهوم می‌پردازیم که عبارت‌اند از:

- **متن پردازنده**، حداقل مجموعه مقادیر ذخیره شده در ثبات‌های یک پردازنده که برای اجرای تعدادی از دستورات نیاز است. متن پردازنده شامل اشاره‌گر پشته، ثبات‌های آدرس دهی و شمارنده برنامه است.
- **متن نخ**، حداقل مجموعه مقادیر ذخیره شده در ثبات و حافظه که برای اجرای تعدادی از دستورات نیاز می‌باشد.متن نخ شامل زمینه پردازنده و حالت آن است.
- **متن فرآیند**، حداقل مجموعه مقادیر ذخیره شده در ثبات‌ها و حافظه که برای اجرای یک نخ استفاده می‌شود (متن نخ و حداقل مقادیر ثبات‌های MMU).

هزینه سوئیچ فرآیند برابر با هزینه‌های **تعویض CPU+به روز رسانی ثبات MMU + کش کردن + TLB**است(شکل ۱-۳).

هزینه سوئیچ نخ برابر با **هزینه‌های تعویض CPU + اطلاعات ضروری برای مدیریت نخ**است.

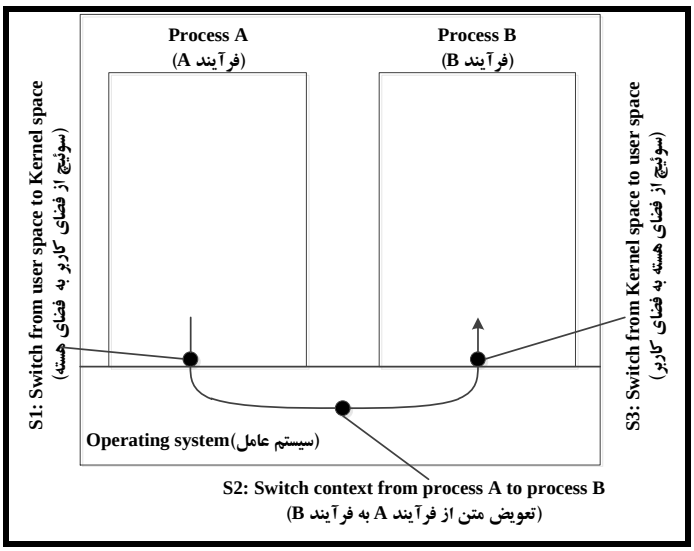
باید توجه داشت، چیزی که توزیع شده است، در واقع فرآیندها هستند و این‌ها هستند که در ماشین-های مختلف توزیع شده‌اند.

۱-۳. نخ‌ها

هر فرآیند شامل یک مسیر اجرایی است. این مسیر اجرایی نخ[۱] نام دارد. به طور سنتی هر فرآیند دارای یک نخ است که سیستم عامل بین فرآیندها سوئیچ می‌کند. به عبارت دیگر، نخ‌ها به عنوان اجزایی از فرآیندها شناخته می‌شوند که می‌توانند به طور هم‌زمان فعالیت کنند. قطعاً نخ زیر مجموعه‌ای از یک فرآیند است و قطعاً در یک فرآیند مجموعه نخ‌ها با یک دیگر همکاری می‌کنند تا وظیفه فرآیند را انجام

[1].Thread [2].Multi Threaded

دهند(رقابت نمی‌کنند). یکی از مهم‌ترین دلیل استفاده از نخ انجام چند عمل به صورت هم‌زمان (**موازی‌سازی**) است. این در حالتی اتفاق می‌افتد که در یک فرآیند چند نخ هم‌زمان اجرا شوند. این عمل را **چند نخی**[۲] می‌گویند. یعنی، استفاده از چند نخی، پردازش موازی را فراهم می‌کند و کار با برنامه‌های بزرگ ساده‌تر می‌شود.



شکل ۱- ۳ سوئیچ فرآیند.

باید توجه داشت که در هر لحظه فقط یک نخ درون یک فرآیند در حال اجرا است. اما، اگر آن نخ متوقف شود، نخ دیگری درون همان فرآیند می‌تواند اجرا شود. به عنوان مثال، نخ اولی در حال اجرا است و تقاضای پورت می‌نماید و متوقف می‌شود. در این جا، برای افزایش کارایی بیکار نبودن پردازنده (CPU)، نخ دوم که عملیات پردازش کردن را دارد، به اجرا در می‌آورد. سپس نخ دوم متوقف می‌شود و نخ اول به کارش ادامه می‌دهد. این امر موجب افزایش کارایی پردازنده و پردازش موازی می‌گردد.

سوئیچ نخ‌ها[۱] ارزان‌تر از سوئیچ فرآیندها است. چون در سوئیچ نخ‌ها تنها وضعیت فرآیند باید ذخیره و بارگذاری شود.چون نخ‌ها در داخل یک فرآیند هستند و از **حافظه اشتراکی**[۲] استفاده می‌کنند. از طرف دیگر، تنها راه ارتباط بین فرآیندها، IPC[۳] است. یعنی، باید فراخوانی سیستم[۴] داشته باشیم. در حالی که برای نخ به جز این می‌توان از حافظه اشتراکی هم استفاده کرد. حفاظت از نخ‌ها در مقابل یک دیگر در

[1].Thead Switching [2].Shared Memory [3].Inter Process Communication [4].System Call

سطح یک فرآیند در اختیار برنامه نویس است. در صورتی که حفاظت از فرآیندها در مقابل دسترسی غیر مجاز به فضای آدرس و داده‌ها آن‌ها در اختیار سیستم عامل است. چنانچه پردازنده‌های بیش‌تری داشته باشیم، می‌توان هر نخ را به یک پردازنده داد و کارایی بالاتری به دست آورد. این همان انجام **پردازش موازی‌سازی** است.

نخ‌ها حتی در سیستم‌های غیر توزیعی هم می‌توانند مفید باشند. به عنوان مثال، در یک برنامه کاربردی صفحه گسترده جایی منتظر ورود داده هستیم و جایی دیگر محاسبات انجام می‌شود. این اعمال را می‌توان به صورت چند نخی انجام داد، وقتی یکی منتظر عملیات ورودی/ خروجی(I/O) است. نخ دیگری محاسبات را انجام دهد.

به طور خلاصه دلایل استفاده از نخ عبارت‌اند از:

۱. کارایی

۲. اگر فراخوانی سیستم بلاک گردید، آن گاه کل فرآیند بلاک نمی‌شود.

۳. موازی سازی در سیستم‌های چند پردازنده‌ای

۴. کمک برای ارتباط درونی بین فرآیندها.

نخ‌ها شماره شناسایی (ID) مختص به خود را دارند و می‌توانند به طور مستقل عمل کنند.

نخ‌های مربوط به یک فرآیند به طور مستقیم می‌توانند با یک دیگر ارتباط برقرار کنند. به عنوان مثال، نخ‌های مستقل می‌توانند به متغیر سراسری دسترسی داشته و آن را به روز کنند.این مدل سر بار بالقوه IPC که هسته مجبور به تحمل آن بود را از بین می‌برد.

چون نخ‌ها در فضای آدرس یکسان هستند، یک سوئیچ نخ ارزان و سریع می‌باشد.

برجسته‌ترین دغدغه نخ‌ها همگام سازی است، به ویژه اگر یک منبع به اشتراک گذاشته شده، مشخص گردد. به عنوان مثال، یک ناحیه بحرانی وجود دارد. در این جا، وقتی یک قطعه کد به منبع به اشتراک گذاشته شده دسترسی دارد، نباید به طور هم زمان توسط بیش از یک نخ دست‌یابی شود. چون هر نخ می‌تواند به طور مستقل اجرا شود، دست یابی به منابع به اشتراک گذاری شده به طور طبیعی تبدیل نمی‌شود.

نسبت نخ به فرآیند مانند نسبت فرآیند به ماشین است، ولی استقلال کم‌تری دارد.

نخ‌ها قرار است با یکدیگر همکاری کنند، اما فرآیندها با هم رقابت می‌کنند.

مدل‌های نخ

مدل‌های مختلف نخ عبارت‌اند از:

۱. چند به یک (Many to One) ۲. یک به یک (One toOne)

۳. چند به چند (Many to Many)

مدل نخ چند به یک، این مدل دارای ویژگی‌های زیر است:

۱. چند نخ سطح کاربر به یک نخ سطح هسته نگاشت می‌شوند.

۲. مدیریت نخ در فضای کاربر انجام می‌شود.

۳. اگر یک فرآیند یک فراخوانی سیستم بلاک را ایجاد کند، تمامی آن فرآیند بلاک خواهد شد.

۴. چون فقط یک نخ می‌تواند در یک زمان به هسته دسترسی داشته باشد، نخ‌های چندگانه قادر به اجرا به صورت موازی روی چند پردازنده نیستند.

۵. روی سیستم‌هایی استفاده می‌شود که نخ‌های هسته را پشتیبانی نمی‌کند.

مدل یک به یک، این مدل دارای ویژگی‌های زیر است:

۱. هر نخ سطح کاربر به یک نخ سطح هسته نگاشت می‌شود.

۲. این مدل هم زمانی بیش‌تر را فراهم می‌کند.

۳. نخ‌های چندگانه می‌توانند به صورت موازی روی سیستم‌های چند پردازنده‌ای اجرا شوند.

۴. وقتی یک نخ بلاک می‌شود، نخ‌های دیگر می‌توانند به اجرای‌شان ادامه دهند.

مشکلات مدل یک به یک عبارت‌اند از:

۱. ایجاد یک نخ سطح کاربر به ایجاد یک نخ سطح هسته متناظر آن نیاز است.

۲. اکثر سیستم‌ها تعداد نخ‌های پشتیبانی شده توسط سیستم را با توجه به سربار در ایجاد نخ‌های سطح هسته محدود می‌کنند.

مثال‌هایی از این مدل می‌توان سیستم‌های عامل ویندوز ۹۵، ۹۸، NT، ۲۰۰۰، XP و سیستم عامل Os/2 را نام برد.

مدل چند به چند، برخی از ویژگی‌های این مدل عبارت‌اند از:

۱. چند نخی سطح کاربر را به یک تعداد کوچک‌تر یا مساوی نخ سطح هسته تسهیم (مالتی پلکس) می‌کند.

۲. تعداد نخ‌های سطح هسته ممکن است برای چه یک کاربرد خاص و یا چه یک ماشین خاص مشخص باشد.

۳. توسعه دهندگان می‌توانند در صورت نیاز چند نخ کاربر را ایجاد کنند.

۴. نخ‌های مربوط به هسته می‌توانند به صورت موازی در چند پردازنده اجرا شوند.

۵. هنگامی که یک نخ فراخوانی سیستم بلاک شونده را اجرا می‌کند، هسته می‌تواند نخ دیگری را زمان بندی کند.

نخ‌ها را به دو روش می‌توان پیاده‌سازی نمود که عبارت‌اند از:

۱. نخ سطح کاربر[1] محض، زمانی که تمام کار مدیریت نخ توسط کاربر صورت می‌گیرد و هسته از وجود آن بی اطلاع باشد (بدون اطلاع سیستم عامل)، در این صورت می‌گویند که سیستم از **نخ‌های سطح کاربر محض** پشتیبانی می‌کند. یعنی، کتاب خانه از نخ‌هایی که توسط برنامه نویس ایجاد می‌شود، پشتیبانی می‌کند. این کتاب خانه شامل **کد ایجاد نخ، کد تخریب نخ، کد تبادل پیام و داده‌ها بین نخ‌ها، کد زمان بندی اجرایی نخ‌ها و کد ذخیره و بار کردن متن نخ‌ها** است. تمام این اعمال در فضای کاربر و در داخل یک فرآیند واحد بدون اطلاع هسته انجام می‌شود. **کتاب خانه نخ‌ها** شامل مجموعه‌ای از برنامه‌های سودمند در سطح کاربر و مورد اشتراک تمام کاربردها می‌باشد.

مشکلات نخ‌های سطح کاربر عبارت‌اند از:

۱. زمانی که نخی یک فراخوانی سیستم را اجرا کند، خودش و کلیه نخ‌های داخل آن فرآیند (مربوط به آن فرآیند) مسدود می‌شوند.

۲. یک کاربرد چند نخی نمی‌تواند از امتیازات چند پردازشی بین نخ‌های خودش استفاده نماید (یعنی، هسته در هر زمان یک فرآیند را به یک پردازنده نسبت می‌دهد).

---

[1].User Level Thread [2].System Level [3].Light Weight Process

۲. نخ سطح هسته[۲] محض، تمام کار مدیریت نخ توسط هسته انجام می‌شود (هیچ گونه کد مدیریت نخ در سطح کاربر وجود ندارد). زمان بندی نخ‌ها توسط هسته (سیستم عامل) انجام می‌پذیرد. هسته می‌تواند به صورت هم زمان نخ‌های چندگانه از یک فرآیند واحد را روی پردازنده‌های مختلف زمان‌بندی نماید.

اما، **مشکل اصلی نخ سطح هسته** این است که تعویض از یک نخ به نخ دیگر در داخل یک فرآیند نیازمند به تغییر حالت پردازنده به حالت هسته است که این کارایی را به شدت کاهش می‌دهد. با توجه به مشکلات نخ‌های سطح کاربر و سطح هسته، شاید بهترین راه حل، ترکیب نخ‌های سطح کاربر و سطح هسته باشد. دو روش ترکیبی وجود دارد که عبارت‌اند از:

۱. فرآیند سبک وزن (LWP)[۳]

۲. سابقه فعالیت زمان بند (Scheduler Activation).

۲-۶-۳. مهاجرت و منابع محلی

همان‌طور که بیان گردید، یک فرآیند از سه قطعه کد، منبع و اجرا تشکیل می‌شود. تا کنون فقط مهاجرت‌های قطعه کد و اجرا بیان گردید. در مهاجرت یک فرآیند، مهاجرت قطعه منبع مسئله‌ای بسیار مهمی است. آن چه که مهاجرت را دشوار می‌کند، این است که همیشه قطعه منبع بدون تغییر منتقل گردد. به عنوان مثال، ارجاع به فایلی از طریق یک آدرس URL مطلق، صرف نظر از ماشینی که فرآیند مالک URL در آن مستقر است، معتبر باقی می‌ماند. برای درک بهتر این موضوع، به انواع انقیاد فرآیندها به منابع می‌پردازیم که عبارت‌اند از:

۱. **انقیاد با شناسه (Binding by identifier)،** زمانی است که یک فرآیند با شناسه یک منبع به آن مراجعه می‌کند. نمونه‌های از این انقیاد می‌توان مراجعه از طریق **پورت**[۲]، **وب سرویس**[۳] و **آدرس**URL را نام برد. این نوع انقیاد، **قوی‌ترین نوع انقیاد** است (شناسه یک مقدار منحصر به فرد (یکتا) می‌باشد).

۲. **انقیاد با مقدار (Binding by Value)،** زمانی است که فرآیند فقط با مقدار منبع سرو کار دارد (یعنی فرآیند تنها به مقدار منبع نیاز دارد). به عنوان مثال، فرآیندی که از پایگاه داده می‌خواند یا فرآیندی که از فایل کتاب خانه‌ای در برنامه نویسی جاوا یا C می‌خواند، آدرس فایل‌های کتاب‌خانه در ماشین‌های

مختلف ممکن است متفاوت باشد.این نوع انقیاد (انقیاد با مقدار) **ضعیف‌تر** از انقیاد با **شناسه** است (آدرس در این نوع انقیاد امکان دارد تغییر یابد).

۳. **انقیاد با نوع (Binding by type)**، زمانی است که فرآیند از نوع خاصی از منبع استفاده می‌کند. مثلاً فرآیند مشخص می‌کند که به دستگاه محلی مانند یک چاپگر یا صفحه نمایش نیاز دارد. **انقیاد با نوع، ضعیف‌ترین نوع انقیاد است**.

در هنگام مهاجرت کد معمولاً نیاز است تا مراجع به منابع تغییر داده شود، اما نوع انقیاد فرآیند به منبع را نمی‌توان تغییر داد. منابع از نظر مهاجرت سه نوع هستند که عبارت‌اند از (جدول ۲-۳):

۱. **منابع غیر متصل (Unattached resource)**، این منابع به راحتی بین ماشین‌های مختلف انتقال می‌یابند. به عنوان مثال، هر فایل داده مربوط به برنامه‌ای است که باید به همراه برنامه انتقال یابد.

۲. **منابع بسته شده (اتصال Fasten resource)**،که انتقال یا کپی این منابع امکان پذیر است (یعنی، منبع قابل جدا شدن از ماشین و انتقال است). اما، هزینه زیادی دارد، مانند پایگاه داده بزرگ و سایت‌های بزرگ.

۳. **منابع ثابت (Fixed resource)**،که با یک ماشین یا محیط کاملاً مقید شده است و غیر قابل انتقال می‌باشد (یعنی، منبع از ماشین جدا نمی‌شود). مثل چاپگر، درایوهای دیسک محلی، پورت‌های ارتباطی و غیره.

جدول ۲-۳. ترکیب انواع انقیاد و انواع منابع.

	غیر متصل (Unattached)	بسته شده (الصاقی)(Fasten)	ثابت (Fixed)
شناسه(Identifier)	MV یا (GR)	MV یا GR	GR
مقدار(Value)	CP یا (MV، GR)	CP یا GR	GR
نوع (Type)	RBیا (MV، CP)	RB یا (GR، CP)	RB یا (GR)
GR(Global Reference): ایجاد یک مرجع سراسری در سیستم			
MV(Move): انتقال منبع			
CP(Copy): کپی مقدار منبع			

RB(Rebind): انقیاد مجدد فرآیند منبع قابل دسترسی به صورت محلی

- **در مهاجرت ضعیف فقط کد انتقال می‌یابد.**
- **در مهاجرت قوی، ماشین دیگری وظایف ماشین مبدأ را انجام می‌دهد.**
- **اگر منبع ثابت (Fixed) باشد، مهاجرت منبع وجود ندارد.**
- **مهاجرت قوی به دلیل خراب شدن سیستمی که فرآیند بر روی آن اجرا می‌شود، می‌باشد.**

- **در مهاجرت ضعیف فقط کد انتقال می‌یابد.**
- **در مهاجرت قوی، ماشین دیگری وظایف ماشین مبدأ را انجام می‌دهد.**
- **اگر منبع ثابت (Fixed) باشد، مهاجرت منبع وجود ندارد.**
- **مهاجرت قوی به دلیل خراب شدن سیستمی که فرآیند بر روی آن اجرا می‌شود، می‌باشد.**

۷-۳. سوالات چهار گزینه‌ای

۱. اهداف مهاجرت پروسس کدام گزینه است؟ (پیام نور- نیم سال اول ۹۱ -۹۰ و نیم سال اول ۹۴-۹۵).

الف: تعادل بار، کارایی ارتباطات، قابلیت دسترسی و بهره‌گیری از قابلیت‌های ویژه.

ب: تعادل بار و کاهش ترافیک شبکه ج: افزایش کارایی و قابلیت دسترسی

د: تعادل بار و قابلیت دسترسی

۲. کدام یک از موارد زیر جزء مزایای مهاجرت کد در سیستم‌های توزیع شده نیست؟(فراگیر – آذر ۹۰).

الف: موازی سازی ب: افزایش کارایی (Performance)

ج: ترکیب بندی سیستم‌های توزیع شده به صورت پویا و انعطاف پذیر

د: متوقف کردن اجرای کد بر روی یک ماشین

۳. کدام عبارت در مورد نخ‌ها(Threads) صحیح است؟ (فراگیر – شهریور۸۶).

الف: هسته از حضور نخ‌های (Threads) سطح کاربر آگاه می‌باشد.

ب: هزینه استفاده از نخ‌های (Threads) مستقل از نوع آن‌ها است.

ج: نخ‌های (Threads) سطح هسته در تک پردازنده‌ها موجب سربار بیش‌تر می‌شوند.

د: تصمیم‌های مربوط به زمان‌بندی در نخ‌های (Threads) سطح کاربر نسبت به نخ‌های(Threads) سطح هسته بهتر انجام می‌گیرد.

۴. کدام یک از موارد زیر جزء ویژگی‌های نخ‌ها (Threads) نمی‌باشد؟(فراگیر – شهریور۸۶).

الف: نخ‌ها (Threads) قابلیت دسترسی کامل به فضای آدرس را دارند.

ب: نخ‌های (Threads) مختلف در یک فرآیند کاملاً از هم مستقل هستند.

ج: نخ‌ها (Threads) می‌توانند به صورت موازی روی پردازشگرها اجرا شوند.

د: نخ‌ها (Threads) می‌توانند متغیرهای عمومی یکسان در فضای آدرس را به اشتراک بگذارند.

۵. کدام مورد از مزایای نخ‌های (Threads) سطح کاربر به حساب نمی‌آید؟ (فراگیر – شهریور۸۶).

الف: نخ‌ها (Threads) با یک دیگر در رقابت هستند.

ب: قابلیت انعطاف‌پذیری در زمان بندی دارند.

ج: برای پشتیبانی این نوع نخ‌ها(Threads) نیازی به اصلاح هسته نمی‌باشد.

د: در این نوع نخ‌ها (Threads)، برای ایجاد، حذف و سوئیچ کردن نیازی به فراخوانی سیستم نمی‌باشد.

۶. داشتن تماس‌های مسدود نشده (Non-Blocking Calls)، از ویژگی‌های کدام مدل سرور (Server) است؟ (فراگیر –آذر ۹۰).

الف: ماشین حالت متناهی (Finite-State Machine)

ب: سرویس دهنده تک نخی (Multi Threaded Server)

ج: سرویس دهنده فایل چند نخی (Single Threaded File Server) د: موارد (ب) و (ج)

۸–۳. پاسخ تشریحی سوالات چهار گزینه‌ای

۱. گزینه (الف) صحیح است. مزایای مهاجرت کد عبارت‌اند از: ۱. موازی سازی ۲. استفاده از قدرت پردازنده‌ها و بالا رفتن کارایی (افزایش کارایی)، ۳. توازن و تعادل بار ۴. انعطاف پذیری (قابلیت

دسترسی) ۵. کاهش ارتباط شبکه (کارایی ارتباطات)۶. پیکربندی سیستم‌های توزیع شده به صورت پویا (Dynamic Configuration)

۲. گزینه (د) صحیح است(مانند پاسخ تست شماره ۱).

۳. گزینه (د) صحیح است.گزینه (الف) نادرست است. چون، یکی از ویژگی‌های نخ‌های سطح کاربر این است که تمام کار مدیریت نخ توسط کاربر انجام می‌شود و هسته از وجود آن بی اطلاع است. گزینه (ب) نیز نادرست است. چون هزینه استفاده از نخ‌ها وابسته به نوع آن‌ها است. همان طور که بیان گردید، نخ‌ها سطح هسته به دلیل این که سوئیچ کردن از یک نخ به نخ دیگر نیازمند به تغییر حالت پردازنده به حالت مد هسته است، کارایی به شدت کاهش می‌یابد و در صورتی که تعویض نخ‌ها سطح کاربر نیاز به تغییر حالت پردازنده به مد هسته نمی‌باشد. با بیان این توضیحات مشخص گردید که گزینه (د) صحیح است، (چون بدون اطلاع سیستم عامل انجام می‌شود). گزینه (ج) نیز نادرست است. چون در هر لحظه یک نخ در داخل هر فرآیند قابل اجرا است. اگر یک نخ متوقف شود (مثلاً تقاضای پورت نماید) برای افزایش کارایی می‌توان نخ دوم را اجرا کرد. اگر نخ دوم متوقف شود، نخ اول می‌تواند به کارش ادامه دهد و این امر موجب افزایش کارایی پردازنده و پردازش موازی خواهد شد.

۴. گزینه (ج) صحیح است.چون یکی از مزایایی بسیار مهم نخ‌ها موازی سازی است، پس گزینه (ج) صحیح می‌باشد. گزینه (د) نیز درست است، زیرا، نخ‌های برای تبادل اطلاعات به یکدیگر می‌توانند حافظه را به اشتراک بگذارند.گزینه (ج) نادرست است، چون نخ‌های یک فرآیند به هم وابسته هستند. زیرا، در نخ‌های سطح کاربر، با مسدود شدن یکی از نخ‌ها، بقیه نخ‌های همان نخ نیز مسدود می‌شوند.

۵. گزینه (الف) صحیح است.نخ‌ها با یک دیگر همکاری می‌کنند نه رقابت.

۶. گزینه (الف) صحیح است.چون طبق جدول ۱-۳، یکی از ویژگی‌های مدل‌های سرویس دهنده فایل چند نخی و سرویس دهنده تک نخی، داشتن تماس انسدادی(مسدود شده) است. اما، ویژگی ماشین حالت متناهی، داشتن تماس‌های غیر انسدادی (مسدود نشده)است.

فصل ۴ ارتباطات

ارتباطات بین فرآیندها یکی از مهم‌ترین مسائل در سیستم‌های توزیع‌شده است. چون متداول‌ترین روش‌های ارتباط در سیستم‌های توزیع‌شده، استفاده از ارسال پیام است. اما، ایرادی که بر سیستم مبتنی بر ارسال پیام وارد می‌باشد، این است که ایجاد سیستم‌های بزرگ بر پایه شبکه امن کار چندان ساده نیست. در حقیقت به دلیل امن نبودن شبکه، ممکن است پیام‌های ارسالی گم شوند و ارتباط برقرار نگردد.

در این فصل به موضوع انواع ارتباطات در سیستم‌های توزیع‌شده، مزایا و معایب هرکدام می‌پردازیم.

انواع مدل ارتباطی در یک سیستم مبتنی بر لایه عبارت‌اند از:

۱. مدل فراخوانی رویه‌های راه دور (RPC)[۱] ۲. مدل احضار متدهای راه دور (RMI)[۲]

۳. مدل میان‌افزار مبتنی بر پیام (MOM)[۳] ۴. مدل جریان[۴] داده‌ای

همان‌طور که بیان گردید، این مدل‌های ارتباطی در سیستم مبتنی بر پروتکل لایه‌ای می‌باشند. بنابراین، قبل از بررسی این مدل‌ها به پروتکل‌های لایه‌ای موجود در سیستم‌های توزیع‌شده و شبکه‌های ارتباطی می‌پردازیم.

۱ - ۴. پروتکل‌های لایه‌ای

همان‌طور که در فصل اول بیان گردید، در سیستم‌های توزیع‌شده حافظه اشتراکی وجود ندارد. بنابراین، تمام ارتباطات با تبادل پیام‌ها از طریق شبکه ارتباطی انجام می‌شود. برای ارسال پیام از یک ماشین به ماشین دیگر باید توافقات مختلفی بین این ماشین‌ها انجام گردد. توافقاتی که بین لایه‌های مختلف انجام می‌شود، همان **پروتکل** است. پروتکل‌ها برحسب مدل‌های مختلفی دسته‌بندی می‌شوند و در هر مدل پروتکل خاصی قرار می‌گیرد. یکی از معروف‌ترین مدل‌ها، مدل مرجع **ارتباط سیستم باز**(OSI)[۵] است. این مدل توسط **سازمان جهانی استاندارد** (ISO)[۶] ارائه گردید. **یک سیستم باز**، سیستمی است که

---

[1].Remote Procedure Call [2].Remote Method Invocation [3].Message Oriented Middleware
[4].Stream [5].Open System Interconnection [6].International Standard Organization
[7]. Connection oriented [8].Connectionless

بتواند تحت استانداردها و قواعدی که ساختار داده‌ها را مشخص می‌نماید با هر سیستم باز دیگری ارتباط برقرار کند.

پروتکل‌ها دو نوع هستند که عبارت‌اند از:

۱. پروتکل اتصال گرا[۷]، پروتکلی است که در آن قبل از تبادل اطلاعات بین فرستنده و گیرنده، یک مسیر ارتباطی ایجادشده، سپس اطلاعات انتقال می‌یابد و در پایان، ارتباط ایجادشده قطع می‌گردد (مانند برقراری ارتباط از طریق تلفن یا موبایل).

۲. پروتکل بدون اتصال[۸]، برای تبادل اطلاعات هیچ‌گونه ارتباطی بین فرستنده و گیرنده برقرار نمی‌شود. فرستنده هرگاه که حاضر بود، بسته اطلاعاتی خود را روی شبکه می‌فرستد (مانند ارسال نامه از طریق پست یا پست الکترونیک).

۱-۱-۴. تفاوت پروتکل اتصال گرا و بدون اتصال عبارت‌اند از:

۱. در پروتکل اتصال گرا برای ارسال بسته‌های مرتبط به هم فقط یک‌بار مسیریابی انجام می‌شود، اما در پروتکل بدون اتصال برای ارسال هر بسته مسیریابی مجزای انجام می‌شود.

۲. در پروتکل اتصال گرا، بسته‌ها به ترتیب خاصی به گیرنده می‌رسند، ولی در پروتکل بدون اتصال، ارسال بسته‌ها ترتیب خاصی ندارد.

۳. در پروتکل اتصال گرا کنترل ترافیک راحت‌تر است. چون مسیر بسته را می‌دانیم.

۴. در پروتکل اتصال گرا، ازآنجای که بسته‌ها به ترتیب به مقصد می‌رسند، فقط اولین بسته، اطلاعاتی کاملی از گیرنده دارد و بقیه بسته‌ها فقط شماره شناسه بسته را دارند. اما، در ارتباط بدون اتصال، همه بسته‌ها باید اطلاعات کاملی از گیرنده بسته را داشته باشند.

معایب ارتباط اتصال گرا عبارت‌اند از:

۱. اگر به هر دلیلی مسیر ارتباطی قطع گردد، کل بسته‌ها باید مجدداً ارسال شوند (قطعات ارسال‌شده و قطعات ارسال نشده بسته باید مجدداً ارسال گردند).

۲. در صورت وجود ترافیک شدید امکان تغییر مسیر وجود ندارد. بنابراین، در برابر تغییرات ناگهانی مسیر شبکه ضعیف می‌باشد.

۲-۴. لایه‌ها

در مدل OSI هفت لایه یا سطح مختلف وجود دارد (شکل ۱-۴). بالاترین لایه، لایه‌ی **کاربرد**[1] و پایین‌ترین لایه‌ی، لایه‌ی **فیزیکی**[2] است. در این مدل برای این که پیامی از یک ماشین به ماشین دیگر ارسال گردد باید از لایه‌های مختلف عبور کند. وقتی بسته اطلاعاتی به لایه‌ای می‌رسد، لایه عملیات خاصی بر روی آن انجام داده (یعنی سرآیندی[3] به آن اضافه کرده)، آن را به لایه پایین‌تر خودش می‌فرستد. اما، در سمت مقصد، اگر بسته اطلاعاتی به هر لایه برسد، این لایه سرآیند اضافه‌شده را حذف می‌نماید و آن را به لایه بالاتر می‌فرستد (شکل ۲ - ۴، نمونه بسته اطلاعاتی در مدل OSI را نشان می‌دهد).

اکنون به شرح مختصر لایه‌های مختلف OSI می‌پردازیم.

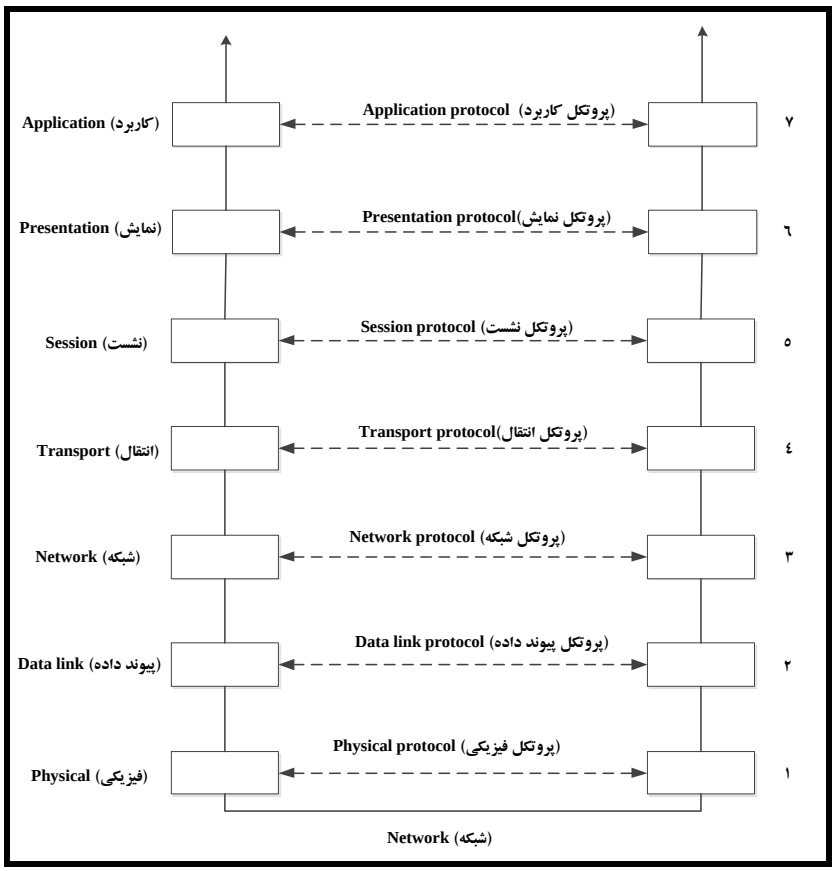
۱-۲-۴. لایه‌های پایین

سه لایه فیزیکی، پیوند داده[4] و شبکه[5] را لایه‌های پایین می‌نامند که توضیحات آن‌ها را در زیر می‌بینید:

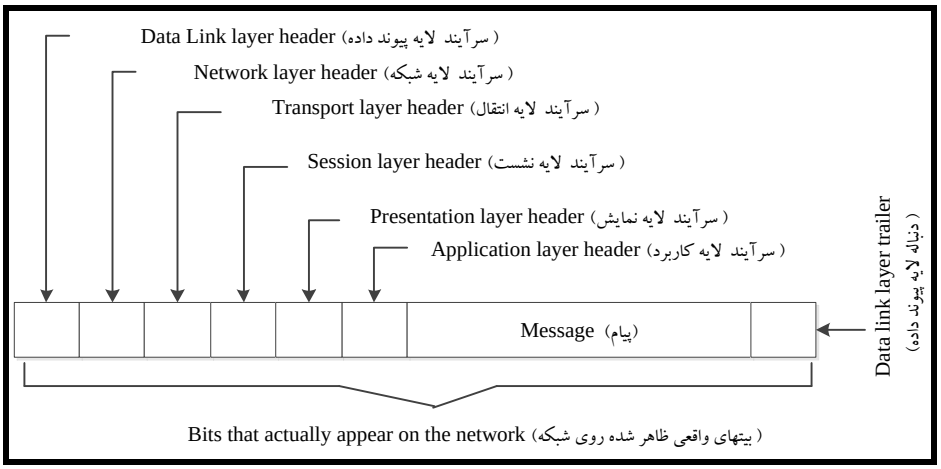
- **لایه فیزیکی**، پایین‌ترین لایه OSI است که اطلاعات دریافتی لایه بالاتر تبدیل به یک سری صفر و یک شده و برای انتقال بر روی بستر ارتباطی، تبدیل به سیگنال الکتریکی و یا موج نوری خواهند شد. در این لایه هیچ پردازشی بر روی اطلاعات ارسالی و یا دریافتی انجام نمی‌شود. نکاتی که در این لایه اهمیت دارد نوع بستر ارتباطی، پهنای باند مربوط به آن، نرخ ارسال اطلاعات و نوع مدولاسیون می‌باشد. کارت شبکه به‌عنوان واسط ارتباطی در این لایه، اطلاعات را از لایه بالاتر دریافت کرده، پس از تبدیل به صفر و یک، تحویل بستر ارتباطی می‌دهد. این لایه در سخت‌افزار شبکه‌بندی اجرا می‌شود.

---

[1].Application [2].Physical [3].Header [4].Data link [5].Network



شکل ۱ – ۴. لایه‌های مختلف مدل OSI.



شکل ۲ – ۴. نمونه بسته اطلاعاتی مدل OSI.

لایه پیوند داده، بالای لایه فیزیکی قرار دارد که وظیفه مدیریت منابع سخت‌افزاری موجود در شبکه LAN را به عهده دارد. چون در یک شبکه LAN منابع سخت‌افزاری در یک بستر ارتباطی مشترک به تبادل اطلاعات می‌پردازند، نیاز به یک سری استانداردها برای جلوگیری از تصادم بین داده وجود دارد. یعنی، وظیفه اصلی این لایه بررسی صحت **الگوی بیتی**[۱] می‌باشد. در این لایه اطلاعات دریافتی از لایه بالاتر دربسته‌هایی به نام **فریم**[۲] بسته‌بندی می‌شود که در هر فریم قابلیت تشخیص خطا وجود دارد. آدرس‌دهی هر فریم بر اساس **آدرس سخت‌افزاری**[۳] خواهد بود. یکی از سخت‌افزارهایی که وظیفه مدیریت منابع سخت‌افزاری و ارتباط هر یک از آن‌ها را بر اساس لایه دوم به عهده دارد، **سوئیچ** است. این لایه اگر خطای در الگوی بیتی، بیت‌های لایه فیزیکی ارسالی پیدا کند، از لایه بالاتر درخواست انتقال مجدد بسته را خواهد کرد.

لایه شبکه، بالای لایه پیوند داده قرار دارد که وظیفه اصلی آن مسیریابی و هدایت ترافیک است. درواقع، از بین مسیرهای مختلف، بهترین مسیر را انتخاب می‌کند. مسیریاب به‌عنوان یک دستگاه که در این لایه وظیفه **مسیریابی**[۴] و **هدایت ترافیک** را به عهده دارد. هدایت ترافیک در این لایه انجام می‌شود. پرکاربردترین پروتکل این لایه، **پروتکل اینترنت (IP)**[۵] است. پروتکل IP، یک پروتکل **بدون اتصال می‌باشد**. پروتکل دیگری که در این لایه استفاده می‌شود، پروتکل **کانال مجازی**[۶] در شبکه‌ها ATM می‌باشد که اتصال گرا است.

لایه انتقال[۷]، لایه‌های سطح پایین و سطح بالایی را از یک دیگر جدا می‌کند. وظیفه اصلی برقراری ارتباط **انتها به انتها**[۸] است. یعنی، این لایه شبکه زیرین خودش را طوری تغییر می‌دهد تا یک برنامه‌نویس بتواند از آن استفاده کند. داده کاربر پس از تحویل به این لایه، دربسته‌های استانداری (قطعات کوچک‌تر) به نام **سگمنت**[۹] شکسته می‌شوند و به هر قطعه یک **شماره ترتیبی**[۱۰] اختصاص می‌دهد تا از طریق این شماره بتواند در مقصد پیام اصلی را ایجاد کند. در این لایه نیز مجدداً وجود خطا در بسته اطلاعاتی تشخیص داده می‌شود. سه پروتکل مهم در این لایه وجود دارند که عبارت‌اند از:

---

[1].Bit Pattern [2].Frame [3].MAC Address [4].Rotting [5].Internet Protocol [6].Virtual channel [7].Transport [8].End to End [9].Segment [10].Sequence Number [11].Transmission Control Protocol [12].TCP for Transaction [13].Universal Datagram Protocol [14].Realities Transport Protocol [15].Session [16].Presantation [17].Application [18].Higher Level Protocol

۱. پروتکل TCP[11]، یک پروتکل اتصال گرا است. اکنون اکثر ارتباطات سرویس‌گیرنده / سرویس‌دهنده بر اساس این پروتکل بناشده است. عیب اصلی این پروتکل، سربار زیاد و تعداد بسته‌های ارسالی بالا می‌باشد. برای رفع این مشکل، مدل دیگری به نام T/TCP ایجادشده است که با کاهش تعداد بسته‌ها، سربار را پایین می‌آورد.

۲. پروتکل UDP[12]، یک پروتکل بدون اتصال می‌باشد.

۳. پروتکل RTP[13]، پروتکلی است که برای ارسال داده‌های **بلادرنگ** (فوری[14]) به کار می‌رود. این پروتکل معمولاً برای ارسال صوت و تصویر استفاده می‌شود. این پروتکل روی ارسال داده کنترل و نظارت می‌کند، اما راه‌کاری جهت تضمین تحویل داده فراهم نمی‌کند.

۳-۴. پروتکل‌های سطح بالا

سه لایه باقی‌مانده **جلسه (نشست)**[15]، **ارائه**[16] و **کاربرد**[17] را پروتکل‌های **سطح بالاتر**[18] می‌نامند. شرح این لایه‌ها در زیر آمده است:

- **لایه جلسه (نشست)**، در حقیقت بهبودیافته لایه انتقال است. وظیفه این لایه برقراری شرایط یک جلسه بین دو جلسه نهایی است. وظیفه **تأیید هویت**[1]، برقراری یک جلسه، مدیریت یک جلسه، اتمام جلسه و بررسی حساب[2] را به عهده دارد. اعمالی که در این لایه انجام می‌شود عبارت‌اند از:

۱. کنترل گفت‌وگو ۲. امکانات هم‌زمانی (هماهنگ‌سازی) ۳. افزودن نقطه بازرسی[3] در پیام تا در صورت بروز خرابی بتوان پیام را از آخرین نقطه بازرسی مجدداً ارسال نمود.

- **لایه کاربردی**، مسئول فعل‌وانفعال مستقیم با کاربران است که شامل مجموعه‌ای از برنامه‌های کاربردی نظیر پست الکترونیکی، پروتکل انتقال فایل (FTP)[4]، پروتکل انتقال ابرپیوند (HTTP)[5] و غیره می‌باشد. **پروتکل FTP**، برای انتقال (تبادل) فایل بین سرویس‌گیرنده و سرویس‌دهنده به کار می‌رود. **پروتکل HTTP**، جهت مدیریت و هدایت انتقال صفحات وب از راه دور به کار می‌رود.

---

[1].Authentication [2].Accountig [3].Check pointing [4].File Transfer Protocol
[5].Hypertext Transfer Protocol [6].Authentication Protocol [7].Commit protocol
[8].Looking Protocol [9].Persistent [10].Transient

۴-۴. پروتکل‌های میان‌افزار

از لحاظ منطقی میان‌افزار برنامه کاربردی است که در لایه کاربرد قرار دارد. اما، چون شامل تعداد زیادی پروتکل مختلف است به‌صورت مجزا بررسی می‌گردد. برخی از پروتکل‌های موجود در میان‌افزار عبارت‌اند از:

۱. **پروتکل احراز هویت**[۶]، فقط به کاربران و فرآیندهای مجاز اجازه دسترسی را می‌دهد.

۲. **پروتکل تثبیت**[۷]، این پروتکل تضمین می‌کند مجموعه‌ای از فرآیندهای مرتبط با یک تراکنش یا همه انجام می‌شوند یا هیچ‌کدام انجام نمی‌شوند (خاصیت اتمیک تراکنش را تضمین می‌کند).

۳. **پروتکل قفل‌گذاری**[۸]**توزیعی**، از استفاده هم‌زمان فرآیندها از یک منبع جلوگیری می‌کند.

همان‌طور که در شکل۳-۴ می‌بینید، معمولاً لایه میان‌افزار در پایین لایه کاربرد به‌جای لایه‌های ارائه و جلسه قرار می‌گیرد. **بنابراین لایه میان‌افزار بین لایه‌های انتقال و کاربرد قرار می‌گیرد**. از مزایای استفاده از میان‌افزار در لایه‌های شبکه عبارت‌اند از: ۱. شفافیت ۲. همگن‌سازی ۳. در اختیار گذاشتن منابع به‌صورت آسان برای کاربران

۱۲-۴. ارتباط ناپایدار پیام گرا

ارتباط ناپایدار پیام گرا (ارتباط گذرای پیام گرا) به دودسته تقسیم می‌شوند که عبارت انداز:

۱. سوکت برکلی (Berkeley Sockets) ۲. واسط مبادله پیام (MIP)

۱-۱۲-۴. سوکت برکلی

مفهوم سوکت[۱]، یکی از ابزارهای معروف مورداستفاده سیستم‌های پیام گرا است. درواقع سوکت، یک **نقطه نهایی ارتباطی**[۲] است که می‌توان داده‌ها را از آن خواند یا در آن نوشت. برای استفاده از سوکت مجموعه‌ای از **عملیات پایه**[۳] تعریف‌شده و توسط سیستم قابل‌استفاده می‌باشد. این اعمال را در جدول ۱ - ۴ می‌بینید.

استانداردسازی واسط لایه انتقال به دو دلیل موردتوجه قرارگرفته است که عبارت‌اند از:

---

[1].Socket [2].Communication endpoint [3].Primitive [4].Binding [5].Message Passing Interface

۱. برنامه‌نویسان بتوانند از مجموعه پروتکل‌های پیام‌دهی استفاده کنند.

۲. از طریق واسط استاندارد، انتقال برنامه‌های کاربردی موردنظر به ماشین دیگر آسان‌تر است.

الگوی ارتباطی اتصال گرا با استفاده از سوکت‌ها در شکل ۱۱ - ۴ آمده است. در این شکل دو ماشین سرویس‌دهنده و سرویس‌گیرنده وجود دارند که می‌خواهند از عملیات مربوط به سوکت جهت برقراری ارتباط استفاده کنند (این عملیات اتصال گرا هستند). همان‌طور که ملاحظه می‌شود، ابتدا سوکت در سطح سرویس‌دهنده ایجاد می‌شود. سپس، عملیات پایه **انقیاد آدرس**[۴] انجام می‌شود. در ادامه، سرویس‌دهنده اعلام آمادگی جهت برقراری ارتباط را می‌نماید (Listen) و با دستور Accept منتظر می‌ماند تا یک ارتباط برقرار گردد. در سمت مقابل سرویس‌گیرنده نیز سوکت را ایجاد می‌نماید و با دستور Connect ارتباط را با سرویس‌دهنده برقرار می‌نماید. بنابراین هنگامی‌که دستور Connect صادر می‌گردد، نوع عملکرد دستور Accept موجب ایجاد همگامی بین این دو فرآیند می‌شود.

اکنون همان‌طور که شکل ۱۱-۴ می‌بینید، اگر سرویس‌دهنده Write نماید، سرویس‌گیرنده Read می‌نماید. این چرخه آن قدر ادامه می‌یابد تا انتقال خاتمه یابد و دستور Close صادر گردد و ارتباط بین دو فرآیند حذف شود.

معایب سوکت‌ها عبارت‌اند از:

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

۲. برای پروتکل‌های اختصاصی که برای شبکه‌های متصل با هم و با سرعت بالا مناسب نیستند.

۲-۱۲-۴. واسط مبادله پیام

همان‌طور که بیان گردید، سوکت‌ها ابزار کافی ندارند و نحوه دسترسی به آن‌ها به‌سادگی انجام نخواهد شد. چون‌که سطح انتزاع آن‌ها و امکاناتی که در اختیار قرار می‌دهند، سطح انتزاع مناسبی نیست. زیرا، بعد از برقراری فقط عملیات Read و Write انجام می‌شود. از طرف دیگر، سوکت‌ها از پروتکل‌های عمومی استفاده می‌کنند (از پروتکل‌های خاص که سرعت بالاتری دارند، نمی‌توانند استفاده نمایند). برای رفع این مشکلات از ابزارهای دیگر نظیر واسط ارسال پیام (MPI)[۵] استفاده می‌شود. MPI، دارای ویژگی‌های زیر است:

۱. MPI، یک استاندارد است که وابسته به سخت‌افزار نیست.

۲. MPI، برای کاربردهای موازی طراحی‌شده و به همین دلیل از نوع ارتباط گذرا است.

۳. MPI، مستقل از پلات فرم است.

۴. MPI، استفاده مستقیم از شبکه را فراهم می‌کند.

۵. در MPI، مفهومی به نام سرویس‌دهنده ارتباطات وجود ندارد یا سطح انتزاع آن به‌طوری بالا است که این مفهوم در آن دیده نمی‌شود.

۶. در MPI، فرض می‌شود که خطاهای جدی دائمی هستند و مفهومی به نام **ترمیم خودکار**[1] وجود ندارد.

۷. MPI، فرض می‌کند ارتباط در داخل گروه شناخته‌شده‌ای از فرآیندها صورت می‌گیرد.

برخی از مهم‌ترین عملیات پایه مبادله MPI را در جدول ۲-۴ می‌بینید.

۱۳-۴. ارتباط پایدار پیام گرا

همان‌طور که دیدید، در ارتباط گذرا پیام گرا، فرستنده و گیرنده در طول ارسال پیام باید فعال باشند. اما در **ارتباط دائمی پیام گرا (ارتباط پایدار مبتنی بر پیام)**[2]، چون امکان ذخیره‌سازی پیام در فضای میانی وجود دارد، پس نیازی نیست فرستنده و گیرنده فعال باشند. یعنی، این امکان فراهم‌شده است تا از بافرها و حافظه‌های میانی برای ذخیره‌سازی پیام استفاده گردد. این عمل موجب می‌گردد که وابستگی فرستنده و گیرنده جهت ذخیره‌سازی پیام‌ها کاهش یابد. پس، ذخیره‌سازی پیام در بافر میانی موجب می‌شود تا گیرنده پیام بتواند غیرفعال. شود. MOM گارانتی می‌نماید به‌محض این‌که گیرنده پیام فعال شود، پیام را برای او ارسال و در صف آن قرار گیرد. انواع سیستم‌های دائمی پیام گرا عبارت‌اند از:

۱. سیستم صف‌بندی پیام (میان‌افزار پیام گرا) ۲. معماری کلی صف‌بندی ۳. کارگزاران پیام

۱-۱۳-۴. مدل صف‌بندی پیام

در ارتباط پیام گرا که به‌صورت پایدار هستند یک سیستم صف‌بندی پیام[3] وجود دارد که گاهی آن را میان‌افزار پیام گرا MOM[4] می‌نامند. در این سیستم، ارتباط بین برنامه‌های کاربردی از طریق اضافه کردن پیام‌ها به صف مشخص و سپس ارسال به مقصد انجام می‌شود. در این سیستم هر برنامه کاربردی صف مربوط به خود را دارد و برای ارتباط با برنامه کاربردی دیگر، پیام را در صف برنامه کاربردی موردنظر

---

[1].Auto Recovery [2].Message Oriented Persistent Connection [3].Message Queuing
[4].Message Oriented Middleware

اضافه می‌کند. در این مدل حتی اگر مقصد غیرفعال باشد، پیام در میان‌افزار (عامل ارتباطی) نگه‌داری می‌شود و درنهایت با فعال شدن مقصد به مقصد ارسال می‌شود.

در این مدل، صف می‌تواند توسط کاربرد خودش خوانده شود، اما چند برنامه کاربردی می‌توانند یک صف مشترک داشته باشند.

در این مدل، فقط به فرستنده تضمین داده می‌شود که پیام ارسال می‌گردد. اما، در مورد زمان تحویل و یا خواندن آن توسط گیرنده به فرستنده اطلاعاتی نمی‌دهد.

جدول۲-٤ برخی از مهم‌ترین عملیات پایه مبادله پیام MPI.

عملیات پایه	توصیف
MIP-bsend	پیام را به انتهای بافر ارسال محلی اضافه می‌کند. این عملیات پایه، ارتباط ناپایدار غیر همگام توسط مجموعه از عملیات/ پشتیبانی می‌شود. هنگامی‌که سرویس‌دهنده پیامی را برای ارسال تحویل می‌دهد، پیام ابتدا در یک بافر محلی در سیستم زمان اجرا کپی می‌گردد. پس از کپی شدن پیام، فرستنده به کار خود ادامه می‌دهد. سیستم زمان اجراMPI، پیام را از بافر محلی خود گرفته و پس از فراخوانی مجموعه‌ای از عملیات Receive توسط گیرنده اقدام به ارسال آن می‌کند.
MPI-send	پیام را ارسال کرده تا کپی شدن آن در بافر محلی یا بافر راه دور منتظر می‌ماند.
MPI-ssend	پیام را ارسال کرده تا شروع دریافت منتظر می‌ماند. برای برقراری ارتباط همگام استفاده می‌شود و فرستنده راه مسدود می‌کند تا درخواستش برای پردازش گرفته شود.
MPI- sendReceive	پیام را ارسال نموده منتظر پاسخ می‌ماند. یعنی، قوی‌ترین شکل ارتباط همگام را پشتیبانی می‌کند، با فراخوانی این مجموعه عملیات درخواستی به گیرنده ارسال کرده و تا برگشت پاسخ آن مسدود می‌ماند. این فراخوانی معادل به

	RPC عادی است.
MPI-isend	یک ارجاع به پیام ارسال می‌فرستد و به کارش ادامه می‌دهد. این روش برای جلوگیری از رونویسی پیام توسط کاربر پس از پایان ارتباط، MPI، مجموعه‌ای عملیات را برای بررسی تکمیل ارسال پیام ارائه می‌دهد.
MPI-iisend	یک ارجاع به پیام ارسالی می‌فرستد و تا شروع دریافت منتظر می‌ماند (مسدود می‌شود). به‌محض مطمئن شدن از دریافت پیام به کارش ادامه می‌دهد.
MPI-receive	در صورت وجود پیام آن را دریافت کرده، وگرنه مسدود می‌شود. یعنی، برای دریافت پیام فراخوانی می‌گردد و در این حالت فرا خواننده را تا رسیدن پیام، مسدود می‌کند.
MPI-ireceive	در صورت موجود بودن پیام آن را دریافت می‌نماید، وگرنه مسدود نمی‌شود. یعنی، گونه‌ای از ارتباط غیر همگام MPI-receive است که آمادگی گیرنده جهت پذیرش پیام را اعلام می‌نماید، ولی فرا خواننده مسدود نمی‌شود.

در این مدل، پیام معمولاً مستقیماً به سرویس‌دهنده مقصد منتقل می‌شود. این مدل از ارتباط از نوع **اتصال یا انسجام ضعیف**[1] استفاده می‌کند. در این نوع ارتباط، فرستنده و گیرنده می‌تواند کاملاً مستقل از هم اجرا شوند، بدون این‌که نیازی به اطلاع از وضعیت یک دیگر داشته باشند.

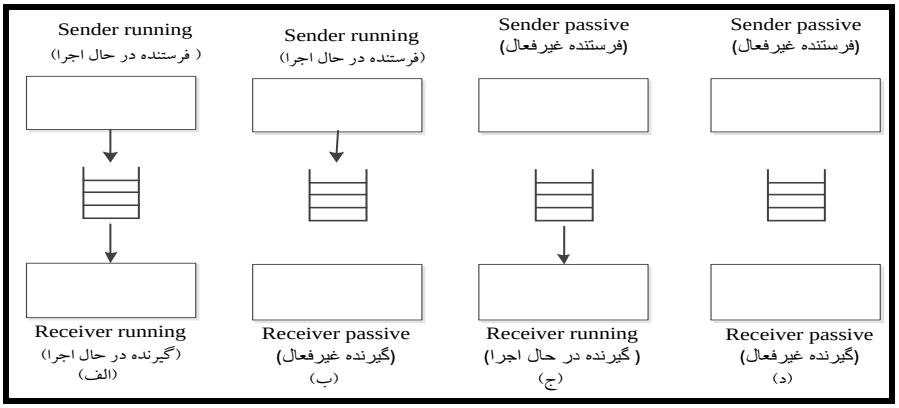
چهار ترکیب برای ارتباط با انسجام ضعیف در شکل ۱۲-۴ آمده است. همان‌طور که در بخش (الف) شکل ۱۲-۴ ملاحظه می‌شود، فرآیند فرستنده و فرآیند گیرنده در زمان انتقال پیام در حال اجرا هستند. اما بافر در یک گره میانی قرارگرفته است که پیام مربوط در آن بافر شده است. در بخش (ب) شکل۱۲-۴، در هنگام ارسال پیام فرآیند فرستنده در حال اجرا است و ممکن است فرآیند گیرنده در حال اجرا نباشد (غیرفعال باشد). در بخش (ج) شکل۱۲-۴، در هنگام دریافت لزوماً فرستنده در حال اجرا نیست. گیرنده‌ها می‌توانند پیام‌های ارسال‌شده به آن را بخوانند. اما در بخش (پ) شکل۱۲-۴، هیچ‌کدام از فرآیندهای فرستنده و گیرنده در حال اجرا نیستند. در این حالت سیستم در حال ذخیره پیام است. در این حالت وقتی

[1].Loosely Coupled

پیامی در صف قرار گرفت، در آن جا می‌ماند تا حذف شود و کاری نـدارد کـه فرسـتنده و گیرنـده‌اش در حال اجرا است یا خیر.

در این مدل پیام می‌تواند شامل هر نوع داده‌ای باشد و تنها باید به‌درستی آدرس‌دهـی شـود. یعنـی، نـام مقصد منحصربه‌فرد باشد. در مدل صف‌بندی پیام، عملیات پایه‌ایی وجود دارند که در جدول ۳-۴ آمده‌اند.

همان‌طور که بیان گردید، عملیات Notify را به نوعی می‌توان مشابه مدیریت وقفـه‌هـا در سیسـتم‌هـای عامل محلی دانست. چون در مورد وقفه‌ها نیز معمـولاً **اداره‌کننده تله**[1] فعـال مـی‌شـود. در ایـن مـدل بـا استفاده از عملیات Notify، اداره‌کننده پیام نصب می‌شود که با ورود آن پیام، اداره‌کننده مربوط فعال و آن را پردازش می‌کند.



شکل ۱۲-۴ چهار ترکیب برای ارتباط با انسجام ضعیف با استفاده از صف.

۲۴-۴. سؤالات چهارگزینه‌ای

۱. مدل ارتباطی OSI از چندلایه تشکیل‌شده است؟ (فراگیر- دی ۹۰).

الف: ۲ ب:۳ ج: ۷ د: ۱۱

۲. در کدام‌یک از مدل‌های RPC زیر قابلیت اطمینـان تضـمین نمـی‌شـود؟ (فراگیـر – شهریور ۸۶).

الف: Synchronized RPC ب: One – Way RPC

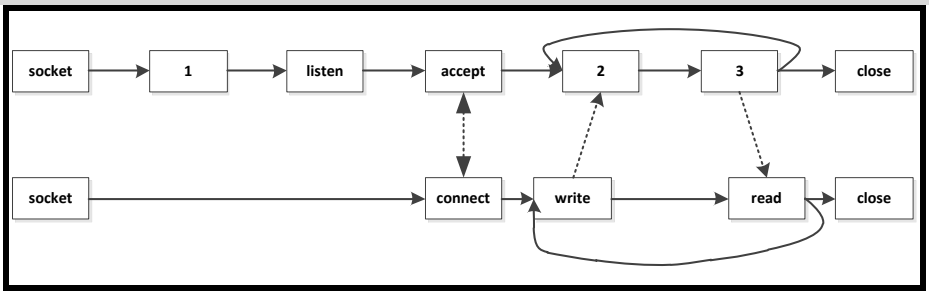
ج: ASynchronized RPC د: در هر سه مدل، قابلیت اطمینان تضمین می‌شود.

[1].Yrap Handler

۳. کدام نوع عملیات پایه (Primitive) از ارتباط بافر شده (Buffered Communication) استفاده می‌کند؟ (فراگیر – اردیبهشت ۸۹).

الف: MPI – ssend ب: MPI – bsend ج: MPI – isend د: MPI- issend

٤. شکل زیر یک الگوی ارتباطی را نشان می‌دهد. عملگرهای مناسب برای جایگاه‌های ۱، ۲ و ۳ به ترتیب (از راست به چپ) کدام است؟ (فراگیر – اردیبهشت ۸۹).



الف: Bind، Read و Write ب: Send، Read و Write

ج: Send، Write و Read د: Bind، Write و Read

۵. کدام‌یک از موارد زیر جز پروتکل‌های سطح پایین انتقال نمی‌باشد؟(فراگیر – اردیبهشت ۸۹).

الف: پیوند داده (Data Link) ب: شبکه (Network)

ج: انتقال (Transport) د: فیزیکی (Physical)

٦. روترهای ذخیره و هدایت (Store-and-forward)، در کدام نوع ارتباط (Communication) مورد استفاده قرار می‌گیرند؟ (فراگیر- اردیبهشت ۸۹).

الف: ناهمگام (Asynchronus) ب: موقت، گذرا یا ناپایدار(Transient)

ج: ناهمگام (Persistent) د: همگام (Sychronous)

۷. کدام‌یک از گزینه‌های زیر از ویژگی‌های کارگزار پیام (Message Broker) در یک سیستم صف‌بندی پیام (Message –Queuing) نمی‌باشد؟ (فراگیر – مرداد ۸۸).

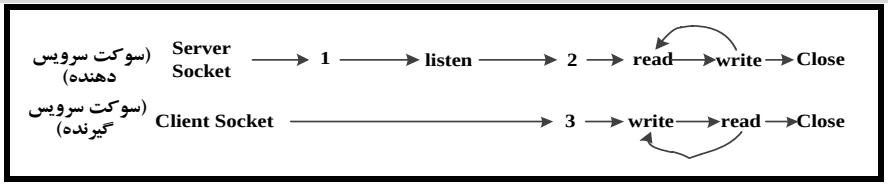
الف: به‌عنوان دروازه‌ای سطح کاربرد (Application) می‌تواند عمل کند.

ب: شامل بانکی از برنامه و قوانین انواع فرمت‌های پیغام است.

ج: تبدیل فرمت پیغام‌ها به‌صورت استاندارد ازجمله وظایف آن است.

د: معمولاً به‌عنوان جز اصلی سیستم صف‌بندی پیام (Message Queuing) می‌باشد.

۸. شکل زیر، ارتباط اتصال گرای سرویس‌گیرنده(Client) و سرویس‌دهنده (Server) را با استفاده از سوکت‌ها نشان می‌دهد. فعالیت‌های مشخص‌شده با شماره‌های ۱، ۲ و ۳ (به ترتیب از سمت راست به چپ) کدام است؟ (فراگیر –مرداد ۸۸).



الف: Accept، Connect و Send

ب: Bind، Accept و Connect

ج: Send، Connect و Accept

د: Connect، Bind و Accept

۲۵–۴. پاسخ تشریحی سؤالات چهارگزینه‌ای

۱. گزینه (ج) صحیح است. لایه‌های OSI عبارت‌اند از: ۱. کاربرد، ۲. نمایش، ۳. نشست، ۴. انتقال، ۵. شبکه، ۶. پیوند داده و ۷. فیزیکی

۲. گزینه (ب) صحیح است. در RPC همگام (گزینه الف)، سرویس‌گیرنده پس از درخواست RPC تا دریافت پاسخ بلاک می‌شود تا مطمئن شود، درخواست او پذیرفته‌شده است. در RPC ناهمگام (گزینه ج)، سرویس‌گیرنده پس درخواست RPC تا دریافت تأییدیه (ACK) منتظر مانده، پس از دریافت رسید از سرویس‌دهنده به کارش ادامه می‌دهد. در این مدل RPC نیز قابلیت اطمینان تضمین می‌شود. ولی در مدل RPC یک‌طرفه (گزینه ب)، سرویس‌گیرنده پس از ارسال درخواست، حتی منتظر دریافت تأییدیه نیز نمی‌ماند و بلافاصله پس از ارسال درخواست به کارش ادامه می‌دهد. این نوع ارتباط RPC فقط در محیط‌های که مسیر ارتباطی کاملاً قابل‌اعتماد باشد، قابل‌استفاده است.

۳. گزینه (ب) صحیح است. چون MPI- bsend، پیام را به انتهای بافر محلی اضافه می‌کند. گزینه (الف) (MPI- ssend)، پیام را ارسال کرده و تا شروع عملیات منتظر می‌ماند. گزینه (ج)(MPI- isend)، یک ارجاع به پیام ارسالی می‌فرستد و به کارش ادامه می‌دهد و اما گزینه (د) (MPI- issend)، یک ارجاع به پیام ارسالی می‌فرستد و تا شروع دریافت منتظر می‌ماند.

٤. گزینه (الف) صحیح است. (مانند شکل ۱۱-۴). همان‌طور که در این شکل می‌بینید، بعد از Socket عملیات Bind قرار می‌گیرد تا آدرس محلی را به سوکت تخصیص دهد. پس گزینه‌های (ب) و (ج) که با Send شروع‌شده‌اند، نادرست هستند. اما، چون موارد ۲ و ۳ سمت سرویس‌دهنده می‌باشد. در سمت سرویس‌دهنده ابتدا Read و سپس Write قرار می‌گیرد.

۵. گزینه (ج) صحیح است. چون لایه‌های پروتکل سطح پایین عبارت‌اند از: ۱. فیزیکی ۲. پیوند داده ۳. شبکه

٦. گزینه (ب) صحیح است. چون در این نوع روترها، اگر روتر نتواند پیامی را به روتر بعدی یا میزبان مقصد تحویل دهد، پیام حذف خواهد شد.

۷. گزینه (د) صحیح است. چون هدف اصلی کارگزار پیام، تبدیل پیام‌های ورودی است که توسط کاربرد مقصد قابل‌فهم باشد. از طرف دیگر، کارگزار پیام بخشی از سیستم صف‌بندی محسوب نمی‌شود، پس گزینه (د) نادرست است.

۸. گزینه (ب) صحیح است. چون اولین گام تخصیص یک آدرس محلی به سوکت است که با عملیات اولیه Bind انجام می‌شود. گام بعدی، گوش دادن است که با عملیات پایه‌ای Listen انجام می‌شود. عملیات بعدی، Accept است که یک فرا خواننده را تا زمانی که ارتباط مربوط برقرار شود، مسدود می‌کند. در سمت سرویس‌گیرنده با دستور Connect درواقع یک ارتباط برقرار می‌شود.

۹. گزینه (ب) صحیح است. به ترکیب دوRPC آسنکرون (ناهمگام)، RPC سنکرون معوق گویند. به‌عنوان‌مثال، فرض کنید، سرویس‌گیرنده‌ای لیستی از آدرس‌های شبکه‌ی تعدادی از گروه‌ها که در آینده با آن‌ها قرار است ارتباط داشته باشد را درخواست می‌کند (سرویس‌گیرنده مسدود نمی‌شود و به کارش ادامه می‌دهد). ابتدا سرویس‌گیرنده این درخواست را به سرویس‌دهنده می‌فرستند به‌محض دریافت ACK از سرویس‌گیرنده، به کار خودش ادامه می‌دهد، سپس، سرویس‌دهنده لیست را آماده کرده، و سرویس‌دهنده را فراخوانی می‌کند تا آدرس‌های پیداشده را به او تحویل دهد. ترکیب این دو RPC آسنکرون را RPC سنکرون معوق می‌گویند.

فصل ۵ نام‌گذاری

نام‌های نقش مهمی در تمامی سیستم‌های رایانه‌ای دارند. از نام‌ها برای به اشتراک‌گذاری منابع، شناسایی موجودیت‌ها[1] به‌طور یکتا، برای مراجعه به مکان‌ها و غیره استفاده می‌شود. نکته مهم در نام‌گذاری این است که نام می‌تواند به موجودیتی تبدیل شود که به آن اشاره می‌کند. نام‌ها برای یافتن موجودیت‌های مستقل از مکان فعلی به آن‌ها به کار می‌روند. تفاوت بین نام‌گذاری در سیستم‌های توزیع‌شده و سیستم توزیع نشده به پیاده‌سازی سیستم نام‌گذاری بستگی دارد. در یک سیستم توزیع‌شده پیاده‌سازی سیستم نام‌گذاری خود نیز در چنین ماشین توزیع می‌شود. نام‌گذاری بحث **شفافیت** را مطرح می‌کند. شفافیت با آدرس امکان‌پذیر نیست. زیرا، اولاً آدرس اطلاعات کامل را می‌دهد و ثانیاً ممکن است آدرس تغییر کند (بحث مهاجرت منبع). اما، نام فرد یکتا است و تغییر نمی‌کند. پس برای مخفی کردن مکان منبع از دید کاربران و راحت‌تر شدن انتقال منابع از نام‌گذاری استفاده می‌شود.

۱ – ۵. نام‌ها، شناسه‌ها و آدرس‌ها

برای کار کردن روی موجودیت‌ها، نیاز به **نقطه دست‌یابی**[2] به آن است. نام، نقطه دست‌یابی **آدرس** است. یعنی، برای دسترسی به موجودیت نیاز به آدرس آن است. ولی، نام نقطه دست‌یابی است.

آدرس، نوع خاصی از نام است و به نقطه دست‌یابی موجودیت اشاره می‌کند. موجودیت ممکن است به‌آسانی نقطه دست‌یابی را تغییر دهد (مانند تعویض IP یک کامپیوتر وقتی‌که در شبکه جابه‌جا می‌شود) یا نقطه دست‌یابی ممکن است به موجودیت دیگری تخصیص یابد. از طرف دیگر، یک موجودیت ممکن است بیش از یک نقطه دست‌یابی داشته باشد. پس، نمی‌توان از آدرس برای دست‌یابی به موجودیت استفاده نمود. چون، اگر نقطه دست‌یابی (آدرس) تغییر کند، ارجاع به آن موجودیت نامعتبر خواهد بود. بنابراین، برای مشخص کردن موجودیت باید آن موجودیت دارای یک نام واحد و **مستقل از آدرس (مستقل از مکان)** باشد.

نام مربوط به موجودیت که مستقل از آدرس آن است، **نام مستقل از مکان**[3] نامیده می‌شود.

---

[1].Entities [2].Access Point [3].location Independent name [4].Identifiers

نام مستقل از مکان، جدا از آدرس، نقطه دست‌یابی است. نام‌های مستقل از مکان انعطاف‌پذیری بیش‌تری دارند.

علاوه بر نام‌ها می‌توان از شناسه‌ها برای دست‌یابی موجودیت‌ها استفاده نمود. **شناسه**[۴]، یک ارجاع بدون ابهام برای موجودیت است. یکی از شناسه‌ها، **شناسه محض** است که دارای خواص زیر می‌باشد:

۱. شناسه حداکثر به یک موجودیت اشاره می‌کند.

۲. شناسه همواره به همان موجودیت ارجاع می‌کند. یعنی، تغییر شناسه وجود ندارد.

۳. هر موجودیت فقط از طریق یک شناسه ارجاع می‌شود (یعنی، شناسه هرگز استفاده مجدد نمی‌شود).

پس شماره تلفن یا موبایل نمی‌تواند به‌عنوان یک شناسه باشد. بنابراین، اگر بخواهیم بفهمیم که دو فرآیند به یک موجودیت اشاره می‌کنند، باید شناسه‌های آن‌ها را با هم مقایسه کنیم، اگر شناسه آن‌ها یکی باشد، پس به یک موجودیت مراجعه می‌نمایند.

نوع دیگری از نام‌ها وجود دارند که به **نام‌های انسان پسند**[۱] معروف هستند(مانند نام فایل‌ها و DNS). این نام‌ها توسط انسان‌ها استفاده می‌شوند. همان‌طور که بیان گردید، سیستم نام‌گذاری DNS، تبدیل نام به آدرس را انجام می‌دهد. یعنی، اتصالات نام به آدرس را نگه‌داری می‌کند که ساده‌ترین شکل آن، جداول زوج (آدرس و نام) است. به‌عنوان‌مثال، سرویس‌گیرنده‌ای که آدرس www. fanavarienoin. net را در مرورگر خودش وارد کند، نیاز به آدرس سرویس‌دهنده وب آن دارد. سیستم نام‌گذاری DNS این نام را تبدیل به آدرس می‌کند که شرح آن را در فصل ارتباطات دیدید.

سیستم نام‌گذاری منتهی برای نام‌گذاری موجودیت‌های که مکان ثابتی دارند، استفاده می‌شود. ولی، برای **انقیاد نام به آدرس**[۲] که مرتب تغییر می‌کند، مناسب نمی‌باشد. چون با تغییر مکان موجودیت، دیگر نمی‌توان آن را دست‌یابی کرد.

۲-۵. انواع نام‌گذاری

انواع نام‌گذاری عبارت‌اند از:

۱. نام‌گذاری تخت (Flat Naming)

۲. نام‌گذاری ساخت‌یافته (Structured Naming)

---

[1].human Friendly [2].Name to Address Binding [3].Unstructured Naming

۳. نام‌گذاری بر اساس صفت (Attribute – Based Naming)

نام‌گذاری تخت

در بسیاری از موارد، شناسه‌ها رشته‌های بیتی تصادفی هستند که می‌توان آن‌ها را نام‌های تخت یا غیره ساخت‌یافته[۳] در نظر گرفت. این رشته هیچ اطلاعاتی را در خود ندارند. یعنی، فاقد اطلاعاتی درباره چگونگی یافتن مکان نقطه دست‌یابی موجودیت مربوط به آن است. به‌عنوان‌مثال، نام‌گذاری سیستم Chord به این صورت است. در این سیستم، نام‌ها، اطلاعاتی برای این‌که به چه صورت می‌توانند حل شوند، ندارند. حجم این گروه‌های نام کوچک است. روش‌های مختلفی برای یافتن موجودیت از طریق شناسه وجود دارد که عبارت‌اند از:

۱. روش‌های ساده

۲. روش‌های مبتنی در مکان خانگی (Home- Based Approaches)

۳. جدول درهم سازی توزیع‌شده (DHT = Distributed Hash Table)

۴. روش‌های سلسله مراتبی (Hierarchical Approaches)

روش‌های ساده

دو راه‌حل برای یافتن موجودیت در نام‌گذاری تخت وجود دارد (هر دو راه‌حل فقط می‌توانند به شبکه‌های محلی اعمال شوند). این دو راه‌حل عبارت‌اند از:

۱. **پخش همگانی/ چندپخشی (Broadcasting/ Multicasting).** در پخش همگانی، پیام حاوی شناسه موجودیت برای تمام ماشین‌ها پخش می‌شود و از هر ماشین درخواست می‌شود که چک کند آیا آن موجودیت را دارد یا خیر. تنها ماشین‌هایی که می‌توانند نقطه دست‌یابی را برای آن موجودیت فراهم کنند پیام پاسخ را می‌فرستند که شامل آن نقطه دست‌یابی است. معایب این روش عبارت‌اند از:

- با رشد شبکه کارایی خودش را ازدست‌داده و ناکارآمد می‌شود. یعنی، برای شبکه‌های بزرگ مناسب نیست.
- پهنای باند زیادی اشغال می‌کند. چون، پیام را به تمام اعضا می‌فرستد.

- بی‌جهت به همه وقفه می‌دهد. درصورتی‌که ارسال وقفه مربوط به آن‌ها نمی‌باشد. کامپیوترها باید کار خودشان را رها کنند و پیام‌های را پردازش نمایند. برای رفع این مشکلات می‌توان از روش چندپخشی استفاده نمود.

۲. **روش چندپخشی**، در این روش پیام برای یک گروه محدود از ماشین‌ها ارسال می‌شود. البته با چندبخشی می‌توان تا حدودی پخش همگانی را بهبود داد. برای استفاده از چندپخشی باید اعمال زیر انجام شود:

- قرار دادن میزبان‌ها در یک گروه
- ارسال پیام به گروه

در این روش لزومی ندارد که یک موجودیت فقط عضو یک گروه باشد.

آدرس چندبخشی کاربردهای زیر را دارد:

۱. آدرس چندپخشی می‌تواند به‌عنوان سرویس مکان‌یابی برای چندین موجودیت به کار رود.

۲. از چندپخشی برای یافتن نزدیک‌ترین موجودیت تکثیری[1] نیز می‌توان استفاده کرد. یک روش برای انتخاب نزدیک‌ترین کپی آن است که اول پاسخ داده است.

۳. تخمین میزان تأخیر شبکه بین گروهی از موجودیت‌ها است. در این کاربرد، پیامی به‌صورت چندپخشی به گروهی فرستاده می‌شود و میزان تأخیر هر یک، از طریق پاسخ برگشت داده‌شده از هر یک، محاسبه می‌شود.

روش‌های چندپخشی همگانی توسعه‌پذیر نیستند و در فضای کوچک قابل‌استفاده هستند.

۳. **روش اشاره گره‌های هدایت‌کننده**، زمانی که موجودیتی متحرک باشد، از این روش استفاده می‌شود. پیدا کردن آدرس موجودیت‌های متحرک مشکل است. چون، با حرکت این موجودیت‌ها آدرس آن‌ها همیشه تغییر می‌یابد. در این روش، وقتی موجودیتی از نقطه A به نقطه B می‌رود، آدرس جدید خود را در A به جا می‌گذارد. یعنی، با حرکت از A به B، یک اشاره‌گر از A به B ایجاد می‌شود. سرویس نام‌گذاری برای یافتن مکان موجودیت، مکان اول موجودیت را دریافت کرده و با دنبال کردن زنجیره‌ای از اشاره‌گرها به مکان جدید موجودیت می‌رسد. این روش دارای مزایای زیر است:

---

[1].Replicate

- عمل مهاجرت را از دید کاربران مخفی می‌کند.
- به‌سادگی قابل پیاده‌سازی است.

اما، معایب این روش عبارت‌اند از:

۵-۵. فضای نام

نام‌ها در سیستم‌های توزیع‌شده در **فضای نام**[1] سازمان‌دهی می‌شوند. فضاهای نام مربوط به نام‌های ساخت‌یافته را می‌توان به‌صورت **گراف جهت‌دار** و **برچسب‌گذاری شده** با دو نوع گره زیر نمایش داد:

۱. **گره‌های برگ**[2]، نشان‌دهنده موجودیت‌هایی هستند که هیچ یالی از آن‌ها خارج نمی‌شود، این گره‌ها معمولاً اطلاعاتی راجع به موجودیت ذخیره می‌کنند تا سرویس‌گیرنده بتواند به آن دسترسی داشته باشد. این اطلاعات می‌تواند اطلاعاتی از قبیل آدرس یا حالت[3] آن موجودیت باشد. به‌عنوان‌مثال، در سیستم فایل، گره برگ شامل اطلاعات کل فایل خواهد بود.

۲. **گره‌های دایرکتوری**[4]، شامل تعدادی یال خروجی به همراه برچسب هستند. این گره‌ها جدولی را نگه‌داری می‌کنند که در آن، یال خروجی به‌صورت جفت (شناسه گره و برچسب یال) نمایش داده می‌شوند.

نمونه‌ای از گراف نام‌گذاری را در شکل۵-۵ می‌بینید. همان‌طور که در این شکل مشاهده می‌شود، در بالای شکل گره n_0، گره ریشه است. این گره فقط خروجی دارد و ورودی ندارد، هر گره به‌جز ریشه یک لبه ورودی دارد. هر گره دقیقاً یک مسیر دارد. ممکن است گراف نام‌گذاری بیش از یک ریشه داشته باشد، ولی بسیاری از سیستم‌های نام‌گذاری دارای یک ریشه هستند.

نمایش نام مسیر[2] در گراف نام‌گذاری به‌صورت زیر است:

N: <label-1, label-2, …, label-n>

---

[1].Name Space [2].Leaf Nodes [3].State [4].Directory Nodes
[2].Path Name [2].Absolute path Name [3].Relative Path Name [4].Global Name
[5].Local Name [6].Boot [7].Supper Block [8].Index Nodes [9].Data Block

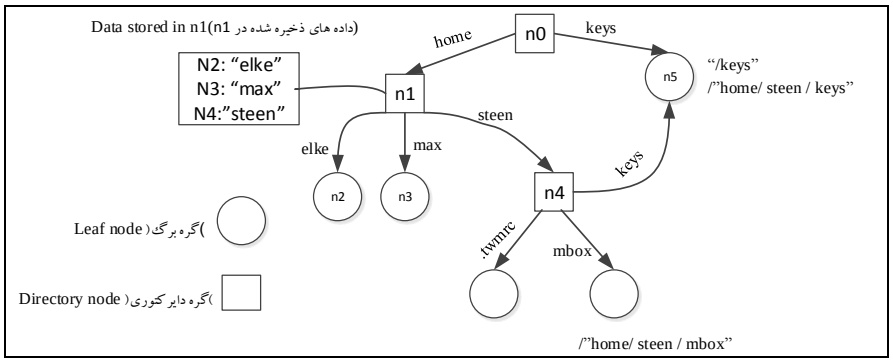
N، اولین گره در مسیر است. ممکن است یک گره با چندین مسیر نمایش داده شود (مانند گره n_5). اگر اولین گره در نام مسیر ریشه باشد، آن مسیر را **نام مسیر مطلق**[۲] می‌نامند. در غیر این صورت، **نام مسیر نسبی**[۳] نام دارد.

نام عمومی[٤]، نامی است که همان موجودیت را نشان می‌دهد و ربطی به‌جایی که آن نام در سیستم استفاده می‌شود، ندارد.

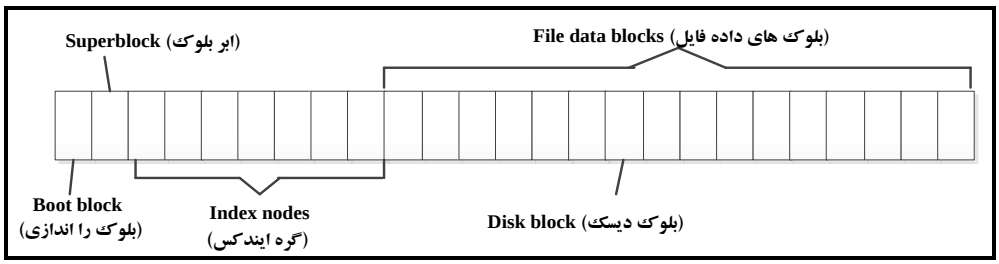
نام محلی[٥]، نامی است که تفسیر آن وابسته به‌جای است که در آن استفاده می‌شود.

در گراف نام‌گذاری در یونیکس، گره دایرکتوری، دایرکتوری فایل و گره برگ، یک فایل را نشان می‌دهد. اجرای گراف نام‌گذاری شامل تعدادی بلاک‌های پشت سر هم از یک دیسک منطقی است که شامل بلاک **راه‌انداز**[٦]، یک **ابر بلاک**[۷]، **چند گره شاخص**[۸] و **بلاک داده**[۹] است (شکل ۶-۵). وظایف هر یک از بلاک‌ها در زیر آمده‌اند:

- **بلاک راه‌انداز**، بلاک مخصوصی از داده و دستورالعمل‌هایی است که وقتی سیستم راه‌اندازی می‌شود، به‌صورت خودکار درون حافظه بار می‌گردد. این بلاک، برای بار کردن سیستم‌عامل به حافظه اصلی به کار می‌رود.



شکل ۵- ۵ گراف نام‌گذاری کلی با یک گره ریشه.



شکل۶-۵ سازمان کلی پیاده‌سازی سیستم فایل یونیکس در دیسک منطقی به‌صورت بلاک‌های پیوسته دیسک.

ابر بلاک، حاوی اطلاعاتی از قبیل اندازه، بلاک‌های روی دیسک که هنوز تخصیص نیافته‌اند و غیره است.

گره‌ها ایندکس که با Inodes نشان داده می‌شوند با یک شماره شاخص ارجاع داده می‌شوند. Inodes، از شماره صفر شروع می‌شود(که شماره صفر مربوط به ریشه است) و هر Inode اطلاعاتی دقیقی برای فایل خودش را دارد. شماره شاخص از هر Inode متناظر با ID گره در گراف نام‌گذاری است. پس هرگاه، شماره شاخص یک Inode را داشته باشیم، می‌توانیم به فایل مربوطش دسترسی داشته باشیم.

۱۰-۵. سؤالات چهارگزینه‌ای

۱. کدام‌یک جزء لایه‌های منطقی فضای نام سیستم‌های توزیع‌شده نیست؟(فراگیر- شهریور ۸۶).

الف: لایه منطقه (Zone Layer)　　ب: لایه عمومی (Global Layer)

ج: لایه مدیریتی(Managerial Layer)　　د: لایه سرپرستی (Administrational Layer)

۲. کدام مورد از ویژگی‌های تبدیل نام بازگشتی (Recursive Name Resolution) نمی‌باشد (فراگیر – مرداد ۸۸).

الف: هزینه ارتباط پایین است.

ب: در این روش سرویس‌دهنده‌های نام(Name Servers) باید دارای کارایی بالایی باشند.

ج: هزینه کش کردن در این روش نسبت به حالت تکراری (Name Servers) کم‌تر است.

د: هر سرویس‌دهنده نام (Iterative) باید آدرس سرویس‌دهنده نام لایه پایین‌تر خود را داشته باشد.

۳. کدام‌یک از موارد زیر، از مزایایی استفاده تبدیل نام بازگشتی (Recursive Name Resolution) می‌باشد؟ (فراگیر - اردیبهشت ۹۰ و جامع برون‌مرزی- تابستان ۹۰).

الف: هزینه‌های ارتباطی می‌تواند کاهش یابد.

ب: درخواست عملکرد بالایی روی هر سرور نام قرار می‌دهد.

ج: نتایج در اختیار قرار گرفتن (Caching) این روش در مقایسه با تبدیل نام تکراری(Iterative Name Resolution) بسیار مؤثرتر است.

د: موارد (الف) و (ج)

٤. کدام‌یک از سیستم‌های مبتنی بر DHT است؟(پیام نور- نیم‌سال دم ۹۲-۹۱).

الف: سیستم Client –Server ب: LAN ج: WAN د: Chord

۵. در سرویس‌های مکانی سلسله مراتبی با عمق k، زمانی که یک موجودیت مشترک تغییر مکان می‌دهد، در بدترین حالت چند رکورد باید به‌روز شود؟(فراگیر- آذر ۹۱).

الف: 2k + 1 ب: k + 1 ج: 2k د: k

٦. در یک سرویس موقعیت سلسله مراتبی (Hierarchical Location Service) با عمق K، اگر یک موجودیت متحرک(Mobility Entity)، موقعیت خودش را تغییر دهد، چه تعداد رکورد موقعیت باید به‌روز شود؟ (فراگیر – اردیبهشت ۹۰).

الف: k ب: k + 1 ج: 2k + 1 د: 2k + 2

۷. تفکیک‌پذیری نام به‌صورت بازگشتی (Recursive Name Resolution) در مورد چه نوع سرورهای نام (Name Server) دارای اهمیت است؟ (فراگیر- اردیبهشت ۸۹).

الف: Lower Level ب: High Level ج: Local د: موارد (الف) و (ب)

۸. کدام مورد در رابطه با URLهای http://www. sanjesh. org/index. html و http://www. sanjesh. com/index. html صحیح می‌باشد؟(فراگیر- اردیبهشت ۹۰).

الف: هر دو مستقل از مکان(Location Independent)می‌باشند.

ب: هر دو وابسته به مکان (Location Dependent)می‌باشند.

ج: اولین URL، وابسته به مکان (Location Dependent) و دومین URL، مستقل از مکان(Location Independent) می‌باشد.

د: اولـین URL، مسـتقل از مکـان (Location Independent) و دومـین URL، وابسـته از مکـان به(Location Dependent) می‌باشد.

۱۱-۵. پاسخ تشریحی سؤالات چهارگزینه‌ای

۱. گزینه (الف) صحیح است. چون لایه‌های منطقی فضای نام توزیع‌شده عبارت‌اند از: ۱. لایه عمومی ۲. لایه مدیریتی و ۳. لایه سرپرستی.

۲. گزینه (ب) صحیح است. چون عیب روش تبدیل نام بازگشتی این است که کارایی بالاتری را از هر سرویس‌دهنده نام می‌طلبد. زیرا، هر سرویس‌دهنده نام باید مسیر را به‌طور کامل تبدیل نماید. پس، در لایه عمومی(Global) فقط از تبدیل نام تکرار استفاده می‌شود.

۳. گزینه (د) صحیح است. چون مزایای تبدیل نام بازگشتی عبارت‌اند از: ۱. نتایج ذخیره‌شده(Caching) بسیار مؤثرتر از روش تکراری می‌باشد. ۲. هزینه ارتباطی کاهش می‌یابد. ۳. از حافظه نهان (Cache) بهتری استفاده می‌نماید. ۴. چون در روش تبدیل نام بازگشتی، آدرس پیداشده هر سرویس‌دهنده به سرویس‌دهنده بعدی نیز ارسال می‌گردد، درصد استفاده از کش در لایه‌های عمومی(Global) و سرپرستی(Administrator) افزایش خواهد یافت.

۴. گزینه (د) صحیح است. جدول درهم سازی توزیع‌شده (DHT)[1] از راه‌حل‌های نام‌گذاری تخت[2] است که انواع مختلف نظیر CAN و Chord را دارد.

۵. گزینه (الف) صحیح است. چون تغییر مکان به‌صورت ترکیبی از یک عمل درج و یک عمل حذف توصیف می‌شود. در عمل درج حداکثر موقعیت $k + 1$ رکورد تغییر می‌کند و در عمل حذف نیز $k + 1$ رکورد موقعیت آن‌ها تغییر خواهد یافت. بنابراین، $2k + 2$ رکورد موقعیت‌شان تغییر می‌یابد که رکورد موجود در ریشه بین دو عملیات مشترک است. بنابراین، حداکثر $2k + 1$ رکورد موقعیت‌شان تغییر می‌یابد.

۶. گزینه (ج) صحیح است. طبق توضیح تست شماره ۵.

۷. گزینه (الف) صحیح است.

---

[1].Distributed Hash Table [2]. Flat Naming

۸. گزینه (الف) صحیح است. هر دو می‌توانند مستقل از مکان باشند، اما، اولی کم‌تر به مکان موجودیت نام اشاره دارد. استقلال از مکان یعنی، نام موجودیت مستقل از آدرس آن باشد.

فصل ۶ همگام‌سازی

همان‌طور که بیان گردید، سیستم‌های توزیع‌شده، از مجموعه‌ای از کامپیوترهای مستقل تشکیل‌شده‌اند که به‌عنوان یک سیستم منسجم عمل می‌کنند. چون کامپیوترها مستقل هستند، می‌توانند ساعت‌های مستقلی داشته باشند و این ساعت‌ها زمان یکسانی را نشان ندهند. اما، یکی از مباحث مهم در ارتباطات، همگام‌سازی ارتباطات است. دو نوع همگام‌سازی باید انجام شود که عبارت‌اند از:

۱. همگام‌سازی فرآیند، در دست‌یابی انحصاری فرآیندها نباید به‌طور **هم‌زمان** و **هم‌روند** به منبع مشترکی دست‌یابی داشته باشد یا گاهی لازم است که فرآیندها بر روی ترتیب انجام کاری توافق کنند. به‌عنوان‌مثال، بین دو فرآیند P و Q باید بر سر این‌که فرآیند P پیام را به فرآیند Q بفرستد یا برعکس، توافق گردد.

در بسیاری از موارد، مهم است گروهی از فرآیندها بتوانند برای دست‌یابی انحصاری به منابع یک فرآیند را به‌عنوان هماهنگ‌کننده بپذیرند که می‌تواند بر اساس الگوریتم انتخاب صورت گیرد. الگوریتم‌های انتخاب هماهنگ‌کننده را در ادامه می‌آموزیم.

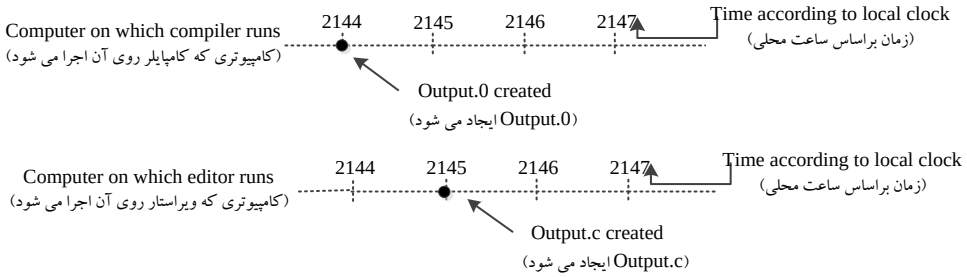
۲. همگام‌سازی ساعت، در سیستم‌های متمرکز زمان مبهم نیست، چون، اگر فرآیند A **زمان را از هسته**[1]، بپرسد و هسته پاسخ دهد. سپس، فرآیند B زمان را بپرسد، زمان پاسخ‌داده‌شده به فرآیند B، مقداری بزرگ‌تر یا مساوی زمان پاسخ‌داده‌شده به فرآیند A خواهد بود. اما، در سیستم‌های توزیع‌شده توافق بر سر زمان ساده نیست. در سیستم‌های توزیع‌شده لازم است **ساعت‌های**[2] کامپیوترها با هم تنظیم گردند. چون عدم همگامی ساعت‌ها در سیستم‌های توزیع‌شده، برخی از مشکلات را ایجاد خواهد کرد. به‌عنوان‌مثال، فرض کنید در یک سیستم توزیع‌شده، یک کامپیوتر وظیفه تایپ و ویرایش یک برنامه (مانند برنامه به زبان C) را به عهده داشته باشد و کامپیوتر دیگر وظیفه کامپایل (ترجمه) برنامه را به عهده دارد. از طرف دیگر، فرض کنید ساعت این دو کامپیوتر دو دقیقه با هم اختلاف دارد. همان‌طور که در شکل ۱-۶ مشاهده می‌گردد، کامپیوتری که وظیفه تایپ و ویرایش برنامه (output. c) را دارد، در زمان

---

[1]. Kernel [2]. Clocks

۲۱۴۲ برنامه را نوشته و در زمان ۲۱۴۴ در کامپیوتر کامپایل کننده کامپایل گردید و در حال اجرا است. اکنون در ساعت ۲۱۴۳، کامپیوتری که ویراستاری به عهده آن است، تغییری را در برنامه ایجاد می‌کند. چون ساعت این کامپیوتر از ساعت کامپیوتر کامپایل کننده ۲ دقیقه عقب‌تر است، بنابراین کامپیوتر کامپایل کننده، فایل output. c را کامپایل نمی‌کند و فایل جدید output.obj ایجاد نمی‌شود. یعنی، تغییرات اعمال نمی‌شود و در کامپیوتر اجراکننده، همان نسخه قبلی برنامه در حال اجرا است. این امر موجب ایجاد مشکلات خواهد شد. بنابراین، ساعت این دو کامپیوتر باید با هم همگام شوند. دو روش برای همگام‌سازی ساعت‌ها وجود دارد که عبارت انداز:

۱. همگام‌سازی فیزیکی، واقعاً همه ساعت‌های کامپیوترها یکسان می‌شوند.



شکل ۱- ۶ هنگامی‌که هر کامپیوتر ساعت خودش را دارد، رویدادی که پس از رویداد دیگری رخ دهد، ممکن است زمان زودتری به آن تخصیص یابد.

۲. همگام‌سازی منطقی، لزومی ندارد که ساعت همه یکسان باشد، ممکن است کامپیوترها ساعت-هایشان با هم تفاوت داشته، ولی کارشان به‌درستی انجام می‌شود.

در ادامه الگوریتم‌های همگام‌سازی فیزیکی و منطقی را می‌آموزیم.

۱ -۶. ساعت فیزیکی

تمام کامپیوترها دارای مداری الکتریکی به نام ساعت یا تایمر هستند که برای نگه‌داری زمان به کار می‌رود. تایمر شامل یک کریستال، یک ثبات شمارنده[1] و یک **ثبات نگه‌دارنده** است. هر دفعه **یک وقفه**[3] (به هر وقفه تایمر یک **تیک ساعت**[4] گویند)، رخ می‌دهد که باعث می‌شود شمارنده ساعت یک

[1]. Counter Register [2]. Holding Register [3]. Interrupt [4]. Clock tick [5]. Clock drift
[6]. Clock Skew [7]. International

واحد افزایش یابد. ثبات نگه‌دارنده ساعت وظیفه‌اش این است که ساعت را نشان دهد. با یک کامپیوتر و یک ساعت، مهم نیست که این ساعت برای مدتی خاموش شود. چون تمام فرآیندها در یک ماشین از یک ساعت استفاده می‌کنند، ازنظر داخلی نیز سازگار خواهند بود. اما، به‌محض معرفی پردازنده‌های چندگانه که هرکدام‌یک ساعت مخصوص به خود را داشتند، وضعیت کاملاً تغییر کرد. وقتی شبکه دارای n کامپیوتر باشد، تمام n کریستال تا حدودی با نرخ‌های متفاوت اجرا می‌شوند و به‌این‌ترتیب ساعت‌ها به‌تدریج از همگامی خارج می‌شوند و مقادیر متفاوتی را ارائه می‌کنند. این تفاوت در مقادیر زمان، **انحراف ساعت**[5] یا **چولگی ساعت**[6] نام دارد.

به دلایل افزایش کارایی و افزونگی، وجود چندین ساعت فیزیکی مطلوب است که منجر به دو مسئله زیر می‌گردد:

- چگونه ساعت‌ها را با ساعت‌های دنیای واقعی همگام کنیم.
- چگونه ساعت‌ها را با یکدیگر همگام کنیم.

۲-۶. ساعت اتمی

با اختراع ساعت اتمی در سال ۱۹۴۸، مستقل از حرکت و لرزش زمین، اندازه‌گیری دقیق‌تر ساعت امکان‌پذیر شد. ساعت اتمی بین‌المللی (TAI)[7] فقط تعداد میانگین تیک‌های ساعت ۱۳۳ شروع از نیمه‌شب ۱۹۵۸، ۱، Jan تقسیم‌بر ۹۱۹۲۶۳۱۷۷۰ است. زمان‌بر اساس ثانیه‌های TAI به نام **زمان هماهنگ شده‌ی جهانی** (UTC)[1] خوانده می‌شود. UTC مبنایی برای تمام ساعت‌های مدرن است. UTC، جایگزین استانداردهای قدیمی (یعنی زمان میانگین گرینویچ) شد که زمان اخترشناسی است. کار UTC ایجاد و ارسال پالس ساعت بود و کافی بود هرکسی یک گیرنده WWV داشته باشد. GMT، ساعت استاندارد ازنظر فیزیکی است. UTC خوب بود، ولی امروزه همه ترجیح می‌دهند از GMT استفاده کنند.

۳-۶. سیستم تعیین موقعیت جهانی

سیستم تعیین موقعیت جهانی (GPS)[2]، یک سیستم توزیعی ماهواره‌ای است که از سال ۱۹۷۸ کار خود را آغاز کرده است. GPS از ۲۹ ماهواره استفاده می‌کند که ماهواره‌ها به‌طور پیوسته موقعیت خود را پخش می‌کنند. هر یک از ماهواره‌ها در مداری به ارتفاع ۲۰۰۰۰ کیلومتری در حال گردش هستند. هر یک از

---

[1]. Universal Time Codrdinated [2]. Global Positionong System [3]. Maximum Driftrate

ماهواره‌ها تا چهار ساعت اتمی دارند که دائماً از ایستگاه‌های خاصی در زمین تنظیم می‌شوند. برای تعیین طول، عرض و ارتفاع یک موقعیت نیاز به حداقل سه ماهواره است. دو حقیقت در دنیای واقعی وجود دارند که باید در نظر گرفته شوند. این دو حقیقت عبارت‌اند از:

۱. قبل از این‌که داده‌های موجود در موقعیت ماهواره به گیرنده برسند، مدت‌زمانی صرف می‌شود.

۲. ساعت گیرنده با ساعت ماهواره همگام نیست.

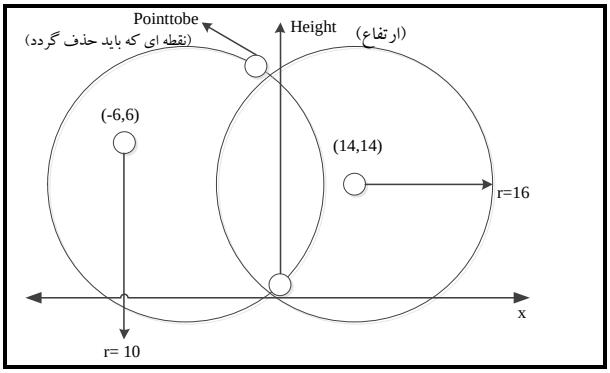
همان‌طور که بیان گردید، در این روش برای تعیین طول، عرض و ارتفاع یک موقعیت نیاز به حداقل سه ماهواره است (شکل ۲-۶). اگر Δr (انحراف ساعت گیرنده از زمان واقعی)، T_i (مهر زمانی پیام دریافتی از ماهواره iام)، Δi (تأخیر انتشار اندازه‌گیری شده از پیام ارسالی توسط ماهواره‌iام)، C (سرعت انتشار نور) باشد، d_i (فاصله واقعی) برابر است با: $di = c\Delta i - C\Delta r$

زمان واقعی (Δr) صفر فرض شده است. اگر Δr صفر فرض نشود، آنگاه $di = C\Delta i - C\Delta r$ $\sqrt{(x_i - x_r)^2 + (y_i - y_r)^2 - (z_i - z_r)^2}$ که در آن موقعیت ماهواره‌ها (zi و yi و xi) است و i می تواند از ۱ تا ۳ باشد. چون، سه ماهواره داریم.

اگر ثابتی مثل P وجود داشته باشد که: $1 - P <= \frac{dc}{dt} <= 1 + P$

این عبارت را بسامد ساعت ماشین P در لحظه t می‌گویند با $C_p^1(t)$ نشان می‌دهند. P، توسط کارخانه سازنده مشخص می‌شود و **حداکثر نرخ انحراف**[۳] نام دارد. dc (ساعت کامپیوتر) و dt (ساعت UTC) است.

در این روش، مشکل زمانی پیش می‌آید که ساعت یک سری از سیستم‌ها کند و یک سری دیگر سریع باشد. اگر هر دو به یک اندازه کند یا سریع باشند، مشکلی پیش نمی‌آید.



شکل ۲-۶ محاسبه مکان (موقعیت) در فضای دوبعدی.

اگر در یک سیستم توزیع‌شده حداقل دو ساعت در جهت مخالف ساعت جهانی (UTC) عمل کنند (یک ساعت سریع و یک ساعت کند)، بعد از زمانی Δt، فاصله زمانی آن دو ساعت $2P\Delta t$ خواهد شد.

اگر آستانه سیستم‌عامل $\Delta t = \delta$ باشد، ساعت سیستم باید در هر $\frac{\delta}{2P}$ثانیه هماهنگ شود.

در اینترنت از GMT برای همگام‌سازی استفاده می‌شود.

استفاده از **سرویس‌دهنده زمان**[1] یکی از روش‌هایی است که برای همگام‌سازی ساعت استفاده می‌شود. اما، اشکال این روش این است که **رویداد گرا**[2] است. یعنی، وقتی به اینترنت وصل می‌شویم این کار را انجام می‌دهد.

در الگوریتم همگام‌سازی ساعت، موارد زیر باید در سیستم‌عامل در نظر گرفته شود:

- اگر هیچ ماشینی دارای گیرنده WWVنباشد، هر ماشین زمان خاص خودش را نگه‌داری می‌کند.
- اگر یک ماشین دارای گیرنده WWV باشد، هدف این است که تمام ماشین‌های دیگر با آن همگام شوند.
- هر ماشینی تایمری دارد که در هر ثانیه H وقفه تولید می‌کند و رویش وقفه مربوط به یک واحد به نرم‌افزار ساعت اضافه می‌کند. مقدار این ساعت راC در نظر می‌گیریم.

فرض H وقفه در ثانیه و ساعت C در زمان t روی ماشین P زمان CP(t) است که ایده آل آن برای هر t و P آنگاه CP(t) = t یا dc/dt = 1 باشد. یعنی، ساعت دقیق است.

در عمل هر دقیقه H وقفه در تایمرها نداریم. درنتیجه دقت $۱۰^{-۵}$ است. یعنی، به‌جای ۲۱۶۰۰۰ = ۳۶۰ × ۶۰ مقادیر ۲۱۶۰۰۲ – ۲۱۵۹۹۸ را خواهیم داشت.

1- CP(t) که آن را چالگی این ساعت گویند و نشان‌دهنده این است که ساعت P چقدر از ساعت جهانی انحراف دارد.

CP(t) – t که آن را **آفست بسامد** ساعت P نسبت به زمان t گویند.

الگوریتم‌های متعددی برای همگام‌سازی ساعت وجود دارند که در این بخش به شرح سه الگوریتم همگام‌سازی می‌پردازیم. این الگوریتم‌ها عبارت انداز:

---

[1]. Time Server [2]. Event Base [3]. Network Time Protocol

۱. الگوریتم پروتکل زمان شبکه (NTP)[۳]

۲. الگوریتم برکلی (Berkeley)

۳. الگوریتم همگام‌سازی ساعت و شبکه بی‌سیم (RBS)[۱]

۱۵-۶. سؤالات چهارگزینه‌ای

۱. در الگوریتم لامپ‌ترت(Lamport) برای ساعت‌های منطقی در دریافت هر پیام، کلاک منطقی با کدام‌یک از روابط زیر تغییر می‌کند؟(پیام نور- نیم‌سال اول ۹۱-۹۰).

الف: $C_j = Max(C_j, T_i)$ ب: $C_j = 2 + Max(C_j, T_i)$

ج: $C_j = 1 + Max(C_j, T_i)$ د: $C_j = 1 - Max(C_j, T_i)$

۲. کدام‌یک از ویژگی‌های زیر بین الگوریتم‌های هم‌زمان‌سازی کریستن(Cristian) و برکلی(Berkeley) مشترک نمی‌باشد؟ (پیام نور جامع برون‌مرزی- تابستان ۹۰).

الف: مرکزی بودن هر دو الگوریتم ب: تعلق داشتن به الگوریتم‌های میانگین یابی

ج: استفاده از گیرنده WWV (UTC) د: هیچ‌کدام

۳. دو بردار برچسب زمانی $T_a = (3, 2, 4, 1)$ و $T_b = (4, 4, 4, 4)$ را در نظر بگیرید. کدام گزینه درست است؟ (فراگیر – آذر ۹۲).

الف: $T_a < T_b$ ب: $T_a > T_b$ ج: $T_a = T_b$ د: مشخص نیست.

۴. برای دو رخداد a و b با بردار با برچسب زمانی $T_a = (3, 2, 4, 1)$ و $T_b = (4, 4, 4, 4)$، کدام گزینه زیر درست است؟ (فراگیر – آذر ۹۲).

الف: $b \rightarrow a$ ب: $a \rightarrow b$ ج: هر دو رخداد هم‌روند هستند د: هیچ‌کدام

۵. کدام در مورد GPS درست است؟ (پیام نور –نیم‌سال دوم ۹۱-۹۰).

الف: یک سیستم توزیع‌شده مبتنی بر ماهواره است.

ب: ماهواره‌ها مداوم موقعیت و مهر زمان خود را به زمین منتشر می‌کنند.

ج: پیام‌های ارسالی از طریق ماهواره با تأخیر دریافت می‌شوند و مقدارشان متناسب با فاصله گیرنده زمینی تا آن ماهواره است.

د: همه موارد.

---

[1]. Reference Broadcast Synchronization [2]. Time Server [3]. Dift Rate

۶. کدام گزینه در مورد الگوریتم‌های همگام‌سازی ساعت درست است؟ (پیام نور –نیم-سال اول ۹۲-۹۱ و نیم سال اول ۹۵-۹٤).

الف: روش برکلی مبتنی بر متوسط گیری ساعت ماشین‌هاست.

ب: روش پروتکل زمان شبکه از سرور زمان برای همگام‌سازی استفاده می‌کند.

ج: سیستم GPS برای همگام‌سازی ساعت و مکان‌یابی استفاده می‌شود.

الف: موارد الف و ب ب: موارد ب و ج ج: مورد ج د: موارد الف، ب و ج

۷. کدام‌یک از گزینه‌های زیر درباره ناحیه بحرانی درست است؟(فراگیر – آذر ۹۲).

الف: الگوریتم متمرکز به سه پیام برای ورود به ناحیه بحرانی و خروج از آن نیاز دارد.

ب: الگوریتم غیرمتمرکز به 2(n – 1) پیام برای ورود به ناحیه بحرانی و خروج از آن نیاز دارد.

ج: الگوریتم حلقه نشان (Token Ring) به حداکثر n -1 پیام برای ورود به ناحیه بحرانی و خروج از آن نیاز دارد.

د: الگوریتم توزیع‌شده به 2n پیام برای ورود به ناحیه بحرانی و خروج از آن نیاز دارد.

۸. کدام‌یک از گزینه‌های زیر درباره تأخیر ورود به ناحیه بحرانی درست است؟ (فراگیر – آذر ۹۲).

الف: الگوریتم متمرکز برای ورود به ناحیه بحرانی دارای ۲ واحد تأخیر است.

ب: الگوریتم غیرمتمرکز برای ورود به ناحیه بحرانی دارای 2n واحد تأخیر است.

ج: الگوریتم حلقه نشان (Token Ring) برای ورود به ناحیه بحرانی دارای 2(n -1) واحد تأخیر است.

د: الگوریتم توزیع‌شده برای ورود به ناحیه بحرانی دارای 2n واحد تأخیر است.

۹. در الگوریتم انحصاری متقابل توزیع‌شده Token-Passing اگر توکن در اختیار پروسس درخواست‌کننده برای ورود به ناحیه بحرانی نباشد، چه تعداد پیام باید در سیستم ارسال نماید (پیام نور– نیم‌سال ۹۱-۹۰).

الف: N – 1 ب: 2(N – 1) ج: 3 (N – 1) د: N

۱۶–۶.پاسخ تشریحی سؤالات چهارگزینه‌ای

۱. گزینه (ج) صحیح است. در مرحله سوم الگوریتم لامپ‌ترت آمده است که موقعی که یک پیام در سایت j دریافت شده، سایت j کلاک خود را به‌صورت $C_j \leftarrow Max[C_j, T_i] + 1$ تغییر می‌دهد (T_i، مهر زمانی دریافتی از سایت i است). یعنی، سایت دریافت‌کننده پیام ماکزیمم کلاک خود و کلاک رسیده را محاسبه می‌کند و یک واحد به آن اضافه می‌نماید تا کلاک منطقی را به دست آورد.

۲. گزینه (ب) صحیح است. چون الگوریتم‌های برکلی و کریستین از گیرنده WWV استفاده نمی‌کنند. پس، گزینه (ج) نمی‌تواند جواب باشد. گزینه (الف) نیز نمی‌تواند جواب باشد. چون الگوریتم کریستین از سرویس‌دهنده زمان (Time Server) به‌صورت متمرکز استفاده می‌کند و الگوریتم برکلی از میانگین یابی زمان توسط یک سرویس‌گیرنده استفاده می‌کند که این روش هم متمرکز است. الگوریتم برکلی فقط از الگوریتم‌های میانگین یابی استفاده می‌کند. اما الگوریتم کریستین از روش تخمین زدن استفاده می‌نماید.

۳. گزینه (د) صحیح است. زیرا چهار برچسب زمانی آن‌ها، همه دارای رابطه =، < یا > به‌صورت یکجا نمی‌باشند.

۴. گزینه (ج) صحیح است. در این برچسب‌های زمانی، یک برچسب زمانی برابر دارند (برچسب زمانی سوم که برابر ۴) است، پس تقریباً هم‌روند می‌باشند.

۵. گزینه (د) صحیح است.

۶. گزینه (د) صحیح است.

۷. گزینه (الف) صحیح است. چون طبق جدول ۱–۶، الگوریتم نامتمرکز به $2 * k * m$ پیام برای ورود و خروج از ناحیه بحرانی نیاز دارد، پس گزینه (ب) نادرست است. اما، الگوریتم حلقه نشان (Token Ring) برای ورود به ناحیه بحرانی و خروج از آن حداکثر به ∞ پیام نیاز دارد. پس، گزینه (ج) نیز نادرست است و الگوریتم توزیع‌شده (صف توزیعی نسخه دوم) به $2(n-1)$ پیام برای ورود به ناحیه بحرانی و خروج از آن نیاز دارد. لذا، گزینه (ج) نیز نادرست می‌باشد.

۸. گزینه (الف) صحیح است. چون طبق جدول۱–۶، الگوریتم نامتمرکز برای ورود به ناحیه بحرانی دارای $2mk$ تأخیر است. پس، گزینه (ب) نادرست است. اما، الگوریتم حلقه نشان (Taken Ring) برای

ورود به ناحیه بحرانی دارای 1 – n ... 0تأخیر است (لذا، گزینه (ج) نیز نادرست است) و الگوریتم توزیع‌شده برای ورود به ناحیه بحرانی دارای(1- n)2تأخیر است. پس، گزینه (د) نیز نادرست است.

۹. گزینه (الف) صحیح است. الگوریتم Token Passing برای ورود به ناحیه بحرانی به 1- N) N تعداد فرآیندها است) نیاز دارد. این الگوریتم با الگوریتم حلقه نشان (Token Ring) اشتباه نشود.

فصل ۷ سازگاری و تکثیر

یکی از مباحث مهم در سیستم‌های عامل توزیع‌شده، **تکثیر**[1] داده‌ها است. داده‌ها به‌طورکلی برای افزایش قابلیت اطمینان و بهبود کارایی تکثیر می‌شوند. یکی از مشکلات اساسی تکثیر، **سازگار**[2] نگه‌داشتن نسخه‌های تکثیرشده است.

در فصل اول آموختیم که تکثیر، تکنیکی برای رسیدن به **توسعه‌پذیری (مقیاس‌پذیری)** است.

Replica، یعنی نسخه تکثیرشده و یکسان است.

در سیستم‌هایی که فقط قابلیت خواندن وجود دارد هر چه تکثیر بیش‌تر باشد، بهتر است و سازگاری از بین نمی‌رود.

در این فصل ابتدا در مورد فواید تکثیر و ارتباط آن با توسعه‌پذیری بحث خواهیم کرد. سپس، معنی واقعی سازگاری را می‌آموزیم. در ادامه درباره دلایل اهمیت سازگاری و روش‌های تأمین آن بحث خواهیم نمود.

۱–۷. دلایل تکثیر

داده‌ها تکثیر می‌شود تا **قابلیت اطمینان** و **کارایی** بهبود یابد. پس، دو دلیل اصلی برای تکثیر داده-ها وجود دارد که عبارت‌اند از:

۱. افزایش قابلیت اطمینان، چون اگر چندین نسخه تکثیرشده از داده‌ها را داشته باشیم، وقتی‌که یک نسخه خراب شود با استفاده از نسخه‌های دیگر تکثیرشده می‌توان به کار ادامه داد.

---

[1]. Replication [2]. Consisten

۲. بهبود کارایی، اهمیت تکثیر برای بهبود کارایی زمانی مشخص می‌شود که سیستم توزیع‌شده باید ازنظر تعداد و ناحیه جغرافیایی توسعه یابد. با قرار دادن یک نسخه داده در نزدیکی فرآیندی که از آن استفاده می‌کند، زمان دسترسی کاهش می‌یابد. درنتیجه کارایی افزایش می‌یابد.

۲-۷. تکثیر به‌عنوان تکنیک توسعه‌پذیری

همان‌طور که بیان گردید، مشکل اصلی تکثیر این است که منجر به معضلات **ناسازگاری** می‌شود. باوجود چندین کپی از داده اگر یک کپی تغییر یابد، آن کپی با سایر کپی‌ها متفاوت خواهد شد. پس تغییرات باید بر روی همه کپی‌ها اعمال شود تا مطمئن شویم همه کپی‌ها با هم سازگار هستند (یعنی برای تضمین سازگاری، باید تغییرات در تمام نسخه‌های اعمال شود). زمان دقیق و چگونگی انجام تغییرات هزینه تکثیر را مشخص می‌کند. موازنه‌ای که باید صورت گیرد این است که نگه‌داشتن به هنگام کپی‌ها، به پهنای باند بیش‌تری در شبکه نیاز دارد. فرآیند P را در نظر بگیرید که N بار در هر ثانیه به نسخه تکثیرشده محلی نیاز دارد و خود نسخه تکثیرشده، M بار در هر ثانیه به هنگام می‌شود. حال فرض کنید که به هنگام رسانی، به‌طور کامل نسخه قبلی کپی، محلی را نوسازی می‌کند. اگر N << M (N خیلی کوچک‌تر از M) باشد. یعنی، نسبت دست‌یابی به هنگام سازی در فرآیند P بسیار کم باشد، درنتیجه ممکن است بهتر باشد که کپی محلی در نزدیکی P نصب نشود، یا راهبرد دیگری را برای به هنگام سازی تکثیر به کار برد.

شکل جدی‌تر سازگار نگه‌داشتن کپی‌ها، این است که خودش می‌تواند باعث ایجاد مشکلات جدی توسعه‌پذیری شود. یعنی، هنگامی می‌توان گفت مجموعه‌ای از کپی‌ها سازگار هستند که همیشه نسخه‌های تکثیرشده یکسان باشند. یعنی، اگر عمل خواندن بر روی هر یک از کپی‌ها انجام شود، همیشه نتیجه یکسانی برگردانده خواهد شد. پس، وقتی یک عمل به هنگام سازی زمانی بر روی یکی از کپی‌ها انجام می‌شود، باید قبل از این‌که عمل دیگری رخ دهد، در تمام کپی‌های دیگر اعمال شود. این نوع از سازگاری، **سازگاری قوی**[1] نامیده می‌شود.

سختی‌ها این عمل، از این حقیقت ناشی می‌شود که باید همه نسخه‌های کپی شده همگام شوند. یعنی، در ابتدا همه کپی‌ها باید به‌طور دقیق بر روی زمان به‌روزرسانی محلی توافق کنند. در بسیاری از موارد، تنها راه‌حل واقعی کم کردن قیود سازگاری است. هزینه‌ای که بابت این موضوع باید بپردازیم این است که

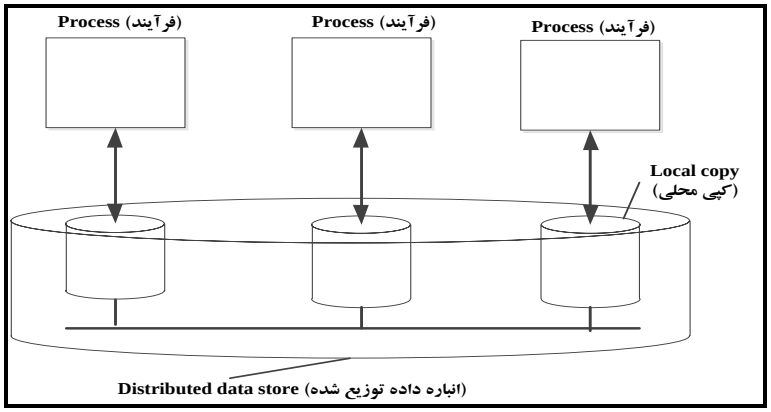
---

[1]. Tight Consistency

ممکن است کپی‌ها همیشه و همه جا یکسان نباشند. در ادامه به انواع سازگاری، معنی دقیق و پیاده‌سازی آن‌ها می‌پردازیم.

۳-۷. سازگاری مبتنی برداده

همان‌طور که بیان گردید، سازگاری زمانی مطرح می‌شود که بخواهید از روی‌داده‌های مشترک بخوانید یا بر روی آن بنویسید. داده‌های مشترک توسط حافظه اشتراکی (توزیع‌شده)، پایگاه داده اشتراکی (توزیع‌شده) یا سیستم فایل اشتراکی (توزیع‌شده) فراهم می‌گردد. در این بخش از واژه وسیع‌تری به نام **انباره داده**[۲] استفاده می‌شود. هر **انبار داده** به‌طور فیزیکی ممکن است در چندین ماشین توزیع‌شده باشد (شکل ۱-۷). هر فرآیندی که می‌تواند به داده‌های انبار دست‌یابی داشته باشد، دارای یک کپی محلی از کل انباره است. هر عملیاتی که باعث تغییر داده شود، به‌عنوان یک عملیات نوشتن محسوب می‌شود و باید به کپی‌های دیگر پخش شود. وگرنه در دسته عملیات خواندن قرار می‌گیرد.



شکل ۱-۷. سازمان کلی یک انباره داده منطقی که به‌طور فیزیکی برای چند فرآیند توزیع و تکثیرشده است.

یک مدل سازگاری، قراردادی بین فرآیندها و انباره داده می‌باشد. این قرارداد بیان می‌کند که اگر فرآیندها موافقت کنند که به‌طور دقیق از قوانین خاصی پیروی کنند، انباره متعدد می‌شود که به‌درستی عمل خواهد کرد. معمولاً، فرآیندی که عملیات خواندن را بر روی‌داده‌ای انجام می‌دهد، انتظار دارد، همان مقداری برگردانده شود که نشان‌دهنده نتیجه آخرین عمل نوشتن بر روی آن داده است. در صورت فقدان ساعت سراسری (جهانی)، تعیین دقیق آخرین عمل نوشتن مشکل خواهد بود. به‌جای آن، مجبور به ارائه

تعریف دیگری خواهیم شد که منجر به پیدایش تعدادی از مدل‌های سازگاری می‌شود. هر مدل به‌طور مؤثر، محدودیت‌هایی که بر نحوه بازگشت مقدار حاصل از یک عمل خواندن بر روی یک داده اعمال می‌کند. **استفاده از مدل‌های سازگاری با قیود(محدودیت‌های) بیش‌تر راحت‌تر است. درحالی‌که مدل‌هایی که محدودیت کم‌تری دارند، استفاده از آن‌ها سخت‌تر است.**

۴-۷. سازگاری پیوسته

همان‌طور که تاکنون بیان گردید، "**بهترین راه‌حل**" برای تکثیر داده‌ها وجود ندارد. یعنی، تکثیر داده‌ها منجر به ناسازگاری می‌شود که متأسفانه قوانین کلی برای رفع این مشکل سازگاری وجود ندارد. دقیقاً میزان تحمل ناسازگاری تا حد زیادی به برنامه کاربردی بستگی دارد. راه‌های مختلفی برای برنامه‌های کاربردی وجود دارد تا تعیین کرد چه ناسازگاری‌های را می‌توان تحمل کرد. Yu و Vahdat در سال ۲۰۰۲، از تشخیص سه محور مستقل یک روش کلی تعریف ناسازگاری ارائه کردند. این سه محور عبارت‌اند از:

۱. **انحراف در مقادیر عددی بین نسخه‌ها**[1]، اندازه‌گیری ناسازگاری برحسب انحراف‌های عددی توسط کاربردهایی استفاده می‌شود که داده‌ها برای آن‌ها معنای عددی دارند. یک کاربرد ممکن است مشخص کند که دو نسخه نباید بیش از ۰,۰۲ انحراف داشته باشند که **انحراف عددی مطلق** خواهد بود.

انحراف عددی نسبی را مشخص کرد که می‌گویند دو نسخه نباید بیش از ۰,۰۵ درصد تفاوت داشته باشند. انحراف عددی را همچنین می‌توان برحسب تعداد به هنگام سازی‌هایی بیان کرد که روی یک کپی اعمال‌شده است، اما، هنوز توسط سایر کپی‌ها مشاهده نشده است.

۱. **انحراف کهنگی**[2] **کپی‌ها**، به آخرین زمان هنگام سازی نسخه مربوط می‌شود. برای بعضی کاربردها، چنانچه داده‌ای فراهم‌شده توسط نسخه خیلی قدیمی نباشد، قابل‌قبول است، به‌عنوان‌مثال، گزارش‌های هواشناسی در مدت چند ساعت معتبر است.

۲. **میزان انحراف از ترتیب عملیات به هنگام رسانی**، برخی از برنامه‌های کاربردی اجازه می‌دهند که ترتیب به هنگام رسانی‌ها در نسخه‌های تکثیرشده متفاوت باشد، البته به شرطی که میزان انحراف

---

[1]. Numerical Deviation [2]. Staleness Deviation

محدود باشد. یعنی، انحراف ترتیب، به حداکثر تعداد نوشتن‌های آزمایشی ممکن در یک سرویس‌دهنده بستگی دارد، بدون این‌که با سرویس‌دهنده تکثیرشده دیگر همگام شود.

درک **شهودی** ترتیب نسبت به دو معیار دیگر سازگاری، سخت‌تر است.

Yuو Vahdat این انحراف‌ها را برای **ساخت سازگاری پیوسته** استفاده می‌کنند.

۵–۷. فرضیه کانیت

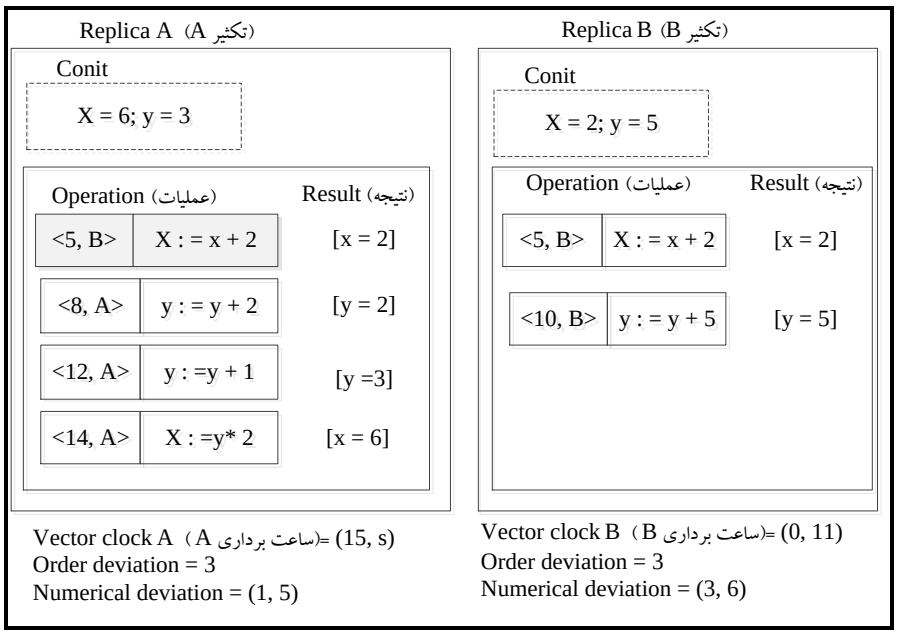
برای تعریف ناسازگاری‌ها Yu و Vahdat، **واحد سازگاری** را معرفی کردند که اختصار **کانیت** گفته می‌شود. کانیت، واحدی را مشخص می‌کند که سازگاری باید روی آن اندازه‌گیری شود. به‌عنوان‌مثال، در معامله بورس، کانیت را می‌توان رکوردی در نظر گرفت که یک سهم را نشان می‌دهد.

شکل ۲–۷ مثالی در مورد کانیت و شرح انحراف‌های عددی و ترتیبی را نشان می‌دهد. دو نسخه تکثیرشده را در شکل ۲–۷ می‌بینید. هر نسخه تکثیرشده i، یک ساعت برداری دوبعدی vci را نگهداری می‌کند. از نمادهای i و t برای بیان عملیات استفاده می‌شود که توسط نسخه iام در زمان b منطقی t اجراشده است.

دو نسخه تکثیرشده بر روی یک کانیت عمل می‌نمایند و هر کانیت شامل داده‌های x و yاست. مقدار اولیه هر دو متغیر صفر در نظر گرفته‌شده است. نسخه تکثیرشده A، عملیات زیر را از نسخه‌ی تکثیرشده B دریافت نموده و آن را دائمی می‌کند و این عملیات در A**تثبیت‌شده**[1] و قابلیت **عقب‌گرد** نمی‌باشد:

5, B: x ←x +2

[1]. commit



شکل ۲-۷. یک مثال درباره انحرافات سازگاری.

نسخه تکثیرشده A دارای سه عملیات به هنگام رسانی موقت است: (8, A)، (12, A) و (14, A) که انحراف ترتیب آن ۳ است. با انجام آخرین عملیات (یعنی (14, A))، ساعت برداری A برابر با (۵ و ۱۵) می‌شود.

عملیاتی از B که هنوز آن را ندیده است، (10, B)می‌باشد که انحراف عددی آن را با توجه به این عملیات، ۱ می‌باشد. B دارای دو عملیات به هنگام سازی موقت است که (5, B) و (10, B) می‌باشند و انحراف ترتیبی آن ۲ است. چون B هنوز یک عملیات را ندیده است، ساعت برداری آن (۱۱و ۰) می‌باشد. انحراف عددی ۳ و وزن کل آن ۶ است. مقدار ۶ ناشی از این حقیقت می‌باشد که مقدار تثبیت شده‌ی B برابر با (x, y) = (0, 0) است، عملیات موقت در A، مقدار x را به ۶ تبدیل کرد.

موازنه‌ای بین نگه‌داری کانیت‌های دانه **ریز**[1] و دانه‌درشت[2] وجود دارد. اگر کانیت حجم زیادی از داده‌ها را نشان دهد (مثل بانک اطلاعاتی کامل، آنگاه به همگام‌سازی‌های مربوط به تمام داده‌ها در کانیت جمع می‌شوند). درنتیجه، این کار ممکن است نسخه‌ها را هر چه زودتر به حالت ناسازگاری ببرد.

---

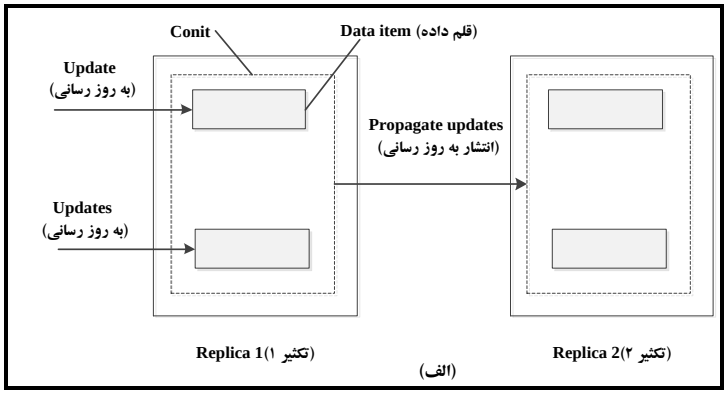
[1]. fine grained [2]. coarse-grained

فرض کنید که در شکل ۳-۷، اختلاف دو نتیجه تکثیرشده نباید بیش از یک به هنگام رسانی معوق باشد. در این صورت، وقتی هر قلم داده در شکل ۳-۷ (الف) هرکدام‌یک‌بار به هنگام شده‌اند، کپی دوم نیز باید به هنگام شود. این عمل درصورتی‌که یک کاینت کوچک‌تر انتخاب شود، انجام نمی‌شود (همان‌طور که در شکل ۳-۷ (ب) می‌بینید). در اینجا هنوز تمام نسخه‌ها به هنگام شده در نظر گرفته می‌شوند. این مسئله مخصوصاً وقتی مهم است که اقلام داده موجود در کاینت کاملاً مستقل از یک دیگر مورداستفاده قرار می‌گیرند، که در این مورد، می‌گوییم **اشتراک نادرست** در کاینت دارند.

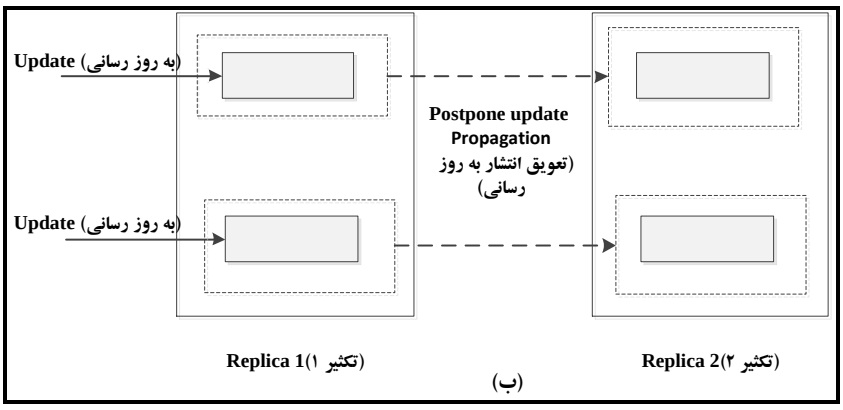
انتخاب تعداد مناسب برای کانیت‌ها خیلی مهم است. اگر کانیت‌ها خیلی کوچک باشند، مناسب نیستند، زیرا، تعداد کل کانیت‌هایی که باید مرتب شوند، افزایش می‌یابد. قبل از استفاده از کانیت‌ها باید دو نکته اساسی زیر را در نظر گرفت:

۱. برای اعمال سازگاری نیاز به پروتکل‌هایی است.

۲. توسعه‌دهندگان باید نیازمندی‌های سازگار را برای کاربردهای خودشان مشخص کنند.



شکل ۳-۷ (الف). انتخاب دانه‌بندی مناسب به هنگام رسانی.



شکل ۳-۷ (ب). هنوز به انتشار به هنگام رسانی نیاز نمی‌باشد.

۱۵-۷. سؤالات چهارگزینه‌ای

۱. اصلی‌ترین دلیل (دلایل) تکثیر (Rplicatios) کدام است؟ (فراگیر - ۸۸ و جامع برون‌مرزی- تابستان ۹۰ و نیم‌سال اول ۹۳-۹٤).

الف: سازگاری (Consistency)　　ب: کارایی (Performance)

ج: قابلیت اعتماد (Reliability)　　د: ب و ج

۲. با توجه به فرآیند زیر کدام عبارت درست است؟(فراگیر - ۸۸).

P1: w(x)a	w(x)c
P2: R(x)a	w(x)b
P3: R(x)a	R(x)c R(x)b
P4: R(x)a	R(x)b R(x)c

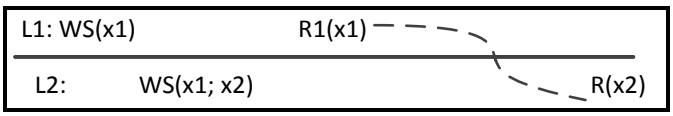
الف: سازگاری علیتی (Casually Consistent) نیست، ولی سازگاری دنباله‌ای (Sequentially Consistent) است.

ب: سازگاری علیتی (Casually Consistent) هست، ولی سازگاری دنباله‌ای (Sequentially Consistent) نیست.

ج: سازگاری علیتی (Casually Consistent) هست، و سازگاری دنباله‌ای (Sequentially Consistent) نیز هست.

د: سازگاری علیتی (Casually Consistent) نیست، و سازگاری دنباله‌ای (Sequentially Consistent) نیز نمی‌باشد.

۳. شکل زیر، کدم مدل سازگاری (Consistency) را نمایش می‌دهد؟ (فراگیر - ۸۹).



الف: خواندن نوشتن‌های‌تان(Reads-Your-Writes) ب: خواندن‌های یکنواخت (Monotonic Reads)

ج: نوشتن‌های یکنواخت(Monotonic Writes) د: نوشتن‌ها بعد از خواندن‌ها(Writes-Follow-Reads)

۴. سه فرآیند به شکل زیر در نظر بگیرید. به کمک این سه فرآیند کدام نتیجه قابل حصول نمی‌باشد؟ (فراگیر - ۸۹ و اردیبهشت ۹۰).

$p1$	p2	p3
$x \leftarrow 1;$	$y \leftarrow 1;$	$z \leftarrow 1;$

print(y, z); print (x, z); print(x,y);

الف: 001001 ب: 001011 ج: 010111 د: 101011

۵. یک فایل روی ۱۰ سرور (Server) کپی (Replicate) شده است. حدنصاب خواندن (Read Quorum) و حدنصاب نوشتن (Write Quorum) مجاز به ترتیب (از راست به چپ) بر اساس الگوریتم رأی‌گیری (Voting) کدام مورد می‌تواند باشد؟(فراگیر -۸۹).

الف: ۴ و ۶ ب: ۴ و ۷ ج: ۵ و ۷ د: ۶ و ۸

۶. کدام‌یک از گزینه‌های زیر دارای ویژگی سازگاری ترتیبی (Sequential Consistency) را دارای می‌باشد؟ (فراگیر -۹۲).

الف:

P1: W(x)a
P2 W(x)b
P3 R(x)b R(x)a
P4 R(x)b R(x)a

ب:

P1: W(x)a
P2 W(x)b
P3 R(x)a R(x)b
P4 R(x)b R(x)a

ج:

P1: W(x)a
P2 W(x)b
P3 R(x)b R(x)a
P4 R(x)a R(x)b

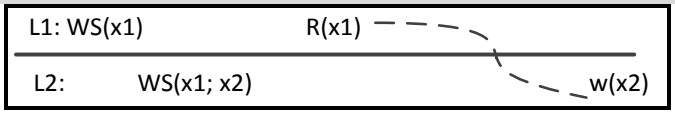
د:

P1: W(x)a
P2: W(x)b
P3: R(x)a R(x)b
P4: R(x)b R(x)a

۷. فرض کنید یک فایل روی ۱۲ سرور جایگزین (Replicate) شده است. کدام‌یک از موارد زیر نمی‌تواند یک ترتیب مجازی از حدنصاب خواندن و حدنصاب نوشتن(Read Quorum, Write Quorum) توسط الگوریتم رای گیری(Voting) می‌باشد؟ (فراگیر - اردیبهشت ۹۰ و جامع برون‌مرزی تابستان - ۹۰).

الف: ۱۲ و ۱ ب: ۶ و ۷ ج: ۸ و ۴ د: ۱۰ و ۳

۸. شکل زیر بیانگر کدام نوع سازگاری (Consistency) می‌باشد؟ (فراگیر - اردیبهشت ۹۰).



الف: خواندن‌های یکنواخت (Monotonic Reads) ب: نوشتن‌های یکنواخت(Monotonic Writes)

ج: نوشتن‌ها پس از خواندن‌ها(Writes-Follow-Reads) د: خواندن نوشتن‌های‌تان(-Reads Your-Writes)

۹. شرط زیر در کدام نوع سازگاری (Consistency) برآورده می‌شود (فراگیر - دی ۹۰).
((یک نوشته روی یک آیتم داده x(data) به‌وسیله یک فرآیند، قبل از هر عمل نوشته متوالی روی x به‌وسیله همان فرآیند، کامل می‌شود))

الف: ترتیبی (Sequential) ب: خواندن نوشتن‌های خود (Read Your Writes)

ج: نوشتن یکنواخت (Monotonic Write) د: نوشتن‌ها بعد از خواندن‌ها (Writes Follow Reads)

۱۰. کدام‌یک از فرآیندهای زیر ویژگی سازگاری دنباله‌ای (Sequential consistency) را دارد؟ (جامع برون‌مرزی- تابستان ۹۰).

P1: W(x)a
P2 W(x)b

P1: W(x)a
P2 W(x)b

P3 R(x)b R(x)a
P4 R(x)b R(x)a

(A)

P3 R(x)b
P4 R(x)a

(B)

الف: فرآیند A ب: فرایند B ج: فرآیندهای A و B د: هیچ کدام

۱۶–۷. پاسخ تشریحی سؤالات چهارگزینه‌ای

۱. گزینه (د) صحیح است. دو دلیل عمده برای تکثیر داده قابلیت **اعتماد** و **کارایی** است. تکثیر داده باعث افزایش قابلیت اعتماد می‌شود. چون اگر داده تکثیر شود، وقتی یک کپی از بین می‌رود یا خراب می‌گردد، با کپی‌های دیگر می‌توان کار را ادامه داد. با افزایش توسعه جغرافیایی کارایی سیستم پایین می‌آید. اما، با تکثیر داده‌ها و قرار دادن یک کپی از داده‌ها در کنار فرآیندی که از آن‌ها استفاده می‌نماید، زمان دست‌یابی به داده‌ها کاهش می‌یابد. پس، کارایی افزایش می‌یابد. اما، اصلی‌ترین مشکل تکثیر، سازگاری نگه‌داشتن چندین کپی (داده‌های تکثیرشده) است. لذا، گزینه (الف) نادرست می‌باشد.

۲. گزینه (ب) صحیح است. چون برای این‌که در مخزن داده‌ای سازگار علّی برقرار باشد، باید "نوشتن‌هایی که به‌طور بالقوه با هم رابطه علّی دارند، باید به ترتیب یکسانی توسط تمام فرآیندها دیده شوند. نوشتن‌های هم‌زمان ممکن است در ماشین‌های مختلف به ترتیب مختلف دیده شوند. در این شکل، نوشتن‌های $W_2(x)b$ و $W_1(x)c$هم‌زمان هستند، پس لازم نیست، همه فرآیندها آن‌ها را به ترتیب یکسانی ببینند. لذا، سازگاری علّی است. اما مخزن داده‌ای زمانی دارای سازگاری دنباله‌ای است که " نتیجه‌ی هر اجرا در صورت یکسان است که عملیات خواندن و نوشتن توسط تمام فرآیندها روی مخزن داده به ترتیب دنباله‌ای انجام شود و عملیات هر فرآیند در این دنباله، به ترتیبی ظاهر گردند که برنامه آن را مشخص می‌نماید. چون در P_3، عملیات $R_3(x)a$، $R_3(x)c$ و $R_3(x)b$ به ترتیب دیده می‌شوند. اما، در فرآیند P_4، $R_4(x)a$، $R_4(x)b$ و $R_4(x)c$ به ترتیب دیده نمی‌شوند. پس سازگاری دنباله‌ای نیست. چون در P_4، $R_4(x)c$بعد از $R_4(x)b$ آمده است (به ترتیب فرآیند P_3 نمی‌باشد).

۳. گزینه (ب) صحیح است. مخزن داده‌ای دارای سازگاری خواندن یکنواخت است که "اگر فرآیندی مقدار x را می‌خواند، هر عملیات خواندن بعدی روی x توسط آن فرآیند همیشه همان مقدار یا جدیدترین مقدار را برمی‌گرداند. در این شکل، فرآیند ابتدا عملیات خواندن را روی x در L_1 اجرا می‌نماید

که مقدار x را (در آن زمان)برمی‌گرداند. این مقدار، حاصل عملیات نوشتن در $WS(x_1)$ در L_1 می‌باشد. فرآیند $R(x_2)$ را در $WS(x_1)$ در L_1 می‌باشد. فرآیند $R(x_2)$ را در L_2 انجام می‌دهد. پس برای تضمین سازگاری خواندن یکنواخت، همه عملیات در $WS(x_1)$ باید قبل از انجام عملیات خواندن دوم، به L_2 پخش‌شده باشد. یعنی، باید $WS(x_1)$ بخشی از $WS(x_2)$ باشد که به‌صورت $WS(x_1: x_2)$ بیان گردد که آمده است.

۴. گزینه (الف) صحیح است. اگر فرآیندها به‌صورت جدول زیر اجرا شوند، گزینه‌های (ب)، (ج) و (د) تولید خواهند شد.

x←1; prints (y, z); →00 y←1; prints(x, z); →10 z←1; prints (x, y); →11 خروجی: ۰۰۱۰۱۱ (گزینه ب)	y←1; z←1; prinst(x, y); →01 prints(x, z); →01 x←1; prints (y, z); →11 خروجی: ۰۱۰۱۱۱ (گزینه ج)	x←1; y←1; prints(x, z); →10 prints(y, z); →10 z←1; prints(y, z); →11 خروجی: ۱۰۱۰۱۱ (گزینه د)

۵. گزینه (ب) صحیح است. حالت‌های ممکن عبارت‌اند از: (1, 10)، (2,9)، (3,8)، (4, 7)، (5, 6)، (6,5)، (7, 4)، (8, 3)، (9, 2) و (10, 1) هستند که گزینه (ب) یعنی (7, 4) درست است و بقیه نادرست هستند.

۶. گزینه (الف) صحیح است. گزینه (ب) نادرست است. چون در فرآیند P_3 به ترتیب $R_3(x)a$ و $R_3(x)b$ آمده است. پس در فرآیند P_4 باید $R_4(a)$ و $R_4(x)b$ به ترتیب بی‌آیند که به‌صورت $R_4(x)b$ و $R_4(x)a$ آمده است. گزینه (ج) نیز نادرست است. چون در فرآیند P_3، به ترتیب$R_3(x)b$ و $R_3(x)a$ آمده است. پس در فرآیند P_4 باید $R_4(x)b$ و $R_4(x)a$بی‌آید که $R_4(x)a$ و $R_4(x)b$ آمده است. گزینه (د) نیز نادرست است. چون در $R_4(x)a$ فرآیند P_4، به ترتیب $R_3(x)a$ و $R_3(x)b$ آمده است. لذا، در فرآیند P_4 باید $R_4(x)b$ باید بی‌آید که $R_4(x)a$ و $R_4(x)b$ آمده است.

۷. گزینه (ج) صحیح است. حالت‌های ممکن عبارت‌اند از: (1, 12)، (2,11)، (3,10)، (4, 9)، (5, 8)، (6, 7)، (7, 6)،

(8, 5)، (9, 4)، (10, 3)، (11, 2)و (12, 1) هستند که گزینه (ج) یعنی (4, 8)نمی‌تواند باشد و بقیه درست هستند.

۸. گزینه (ج) صحیح است. چون سازگاری نوشتن بعد از خوانده‌ها تضمین می‌کند عملیات نوشتن بعدی توسط فرآیندی روی آیتم داده X (نوشتن $W(X_2)$ در کپی محلی L_2)، که عملیات خواندن قبلی روی X ($R(X_1)$) در کپی محلی L_1) توسط همان فرآیند می‌آید، به روی همان مقدار یا تازه‌ترین مقدار X که خوانده‌شده، عمل می‌کند.

۹. گزینه (ج) صحیح است. وقتی مخزنی سازگاری ترتیبی (گزینه الف) دارد که << نتیجه اجرای هر عمل خواندن و نوشتن در صورتی یکسان است که این عملیات توسط تمام فرآیندها روی انباره داده‌های به ترتیب مشخصی انجام شوند و عملیات هر فرآیند خاص، به ترتیبی ظاهر شوند که برنامه‌ی آن مشخص می‌کند>>. اما، انباره داده‌ای سازگاری خواندن خود نوشتن‌ها (گزینه ب) را دارد که << نتیجه عملیات نوشتن روی آیتم داده‌ای x توسط یک فرآیند، همواره با عملیات خواندن بعدی x توسط همان فرآیند قابل‌مشاهده است>>. گزینه (د) در سؤال ۱۴ بیان گردیده است.

پیوست الف
تست‌های تکمیلی

۱. منظور از شفافیت دسترسی چیست؟

الف:دسترسی منابع راه دور با استفاده از نام‌های مستقل از محل.

ب: دسترسی منابع محلی و راه دور با استفاده از عملیات مشابه.

ج: دسترسی یک منبع تکثیر شده دقیقاً مشابه به دسترسی آن اگر یک شی واحد بود، می‌باشد.

د: یک منبع همه درخواست‌های را به صورت مساوی و مستقل از مکان سرویس گیرنده انجام می‌دهد.

۲. منظور از شفافیت مکان چیست؟

الف: دسترسی منابع راه دور با استفاده از نام‌های مستقل از محل.

ب: دسترسی منابع محلی و راه دور با استفاده از عملیات مشابه.

ج: دسترسی یک منبع تکثیر شده دقیقاً مشابه به دسترسی آن اگر یک شی واحد بود، می‌باشد.

د: یک منبع همه درخواست‌های را به صورت مساوی و مستقل از مکان سرویس گیرنده انجام می‌دهد.

۳. منظور از شفافیت هم‌روندی چیست؟

الف: نخ‌ها، مجاز به دسترسی به ساختمان داده‌های به اشتراک گذاشته شده می‌باشند.

ب: فرآیندها می‌توانند بدون تداخل با هم به منابع دسترسی داشته باشند.

ج: دسترسی یک منبع تکثیر شده دقیقاً مشابه به دسترسی آن اگر یک شی واحد بود، می‌باشد.

د:بدون آن که برنامه تغییر کند، گره‌ی جدید می‌تواند به سیستم اضافه شود.

٤. منظور از شفافیت خرابی چیست؟

الف: خرابی‌ها از دید کاربران یک منبع مخفی می‌شوند.

ب:خرابی‌های منبع می‌توانند استثناهای که توسط کاربر اداره می‌شوند را افزایش دهند.

ج: منابع می‌توانند فقط از طریق از کار افتادن خراب شوند.

د: یک سیستم قوی که منابع در آن خراب نمی‌شوند.

۵. منظور از شفافیت تکثیر چیست؟

الف: فرآیندها از شمای تکثیرها مطلع‌اند و می‌توانند از آن‌ها استفاده کنند.

ب: کاربران یک منبع به آن مانند حالتی که تکثیر نشده باشد، دسترسی دارند.

ج: داده‌ها تغییر ناپذیرند و می‌توانند بر روی دیسک سریال شوند.

د: پاسخ به پرس‌وجوها به منظور جلوگیری از خواندن کثیف کپی و قفل می‌شوند.

۶. نشانه‌ی معماری سرویس‌دهنده - سرویس گیرنده چیست؟

الف:سرویس گیرنده قسمت فعال است. ب: سرویس‌دهنده بیش‌ترین توان اجرایی را دارد.

ج: چند سرویس گیرنده و یک سرویس‌دهنده دارد. د: سرویس‌دهنده قسمت فعال است.

۷. یک سیستم را که یک گره همیشه به درخواست‌ها پاسخ می‌دهد و بقیه‌ی گره‌ها فقط با این گره ارتباط برقرار می‌کنند، چه نام دارد؟

الف:سیستم آسنکرون ب: سیستم سرویس‌دهنده –سرویس گیرنده

ج: سیستم همتا به همتا(Peer-To-Peer) د: سیستم سنکرون

۸. نشانه‌ی معماری همتا به همتا(Peer–To-Peer) چیست؟

الف: هیچ گره‌ی به تنهایی یک عمل را انجام نمی‌دهد.

ب: گره‌ها به صورت سلسله مراتبی سرویس‌دهنده –سرویس گیرنده معماری شده‌اند.

ج: همه‌ی گره‌ها می‌توانند مستقیماً با تمام گره‌های در شبکه ارتباط برقرار کنند.

د: همه‌ی گره‌ها فعال‌اند و می‌توانند در عملیات مختلف شرکت کنند.

۹. در سیستم سنکرون(هم گام) چه چیز را می‌دانیم؟

الف: همه‌ی عملیات زمان یکسانی می‌گیرند. ب:دقیقاً چه مقدار زمان برای رسیدن پیام طول می‌کشد.

ج: همه‌ی پیام‌ها می‌رسند. د: حد بالا برای زمان اجرای یک عملیات

۱۰. نشانه‌ی یک سیستم آسنکرون(غیرهم‌گام) چیست؟

الف: همه‌ی عملیات زمان یکسانی می‌گیرند. ب: ساعت درونی دارای رانش محدود است.

ج: حد بالای رای زمان اجرای یک عملیات وجود ندارد. د: پیام ممکن است به مقصد نرسد.

۱۱. به سیستمی که ماکزیمم زمان اجرای عملیات و رسیدن پیام مشخص است، چه می‌گویند؟

الف: سیستم تحمل خطا ب: سیستم بلادرنگ ج: سیستم آسنکرون د: سیستم سنکرون

۱۲. UDP چه چیزی را ارایه می‌دهد؟

الف: دانستن رسیدن پیام ب: جریان بایت یک طرفه بین دو پردازنده

ج: بهترین راه برای رسیدن پیام‌ها به پردازنده د: ترتیب رسیدن پیام‌ها

۱۳. TCPچه چیز را ارایه می‌دهد؟

الف:تضمین این که همیشه پیام به مقصد می‌رسد. ب: جریان دو طرفه‌ی کامل بین دو پردازنده

ج: بهترین راه برای رسیدن پیام‌ها به پردازنده د: کنترل کاربردی و ثابت

۱٤. کدام آدرس(ها) می‌تواند در سر آیندTCP وجود داشته باشد؟

الف: در سر آیند TCP هیچ قسمتی برای آدرس وجود ندارد ب: آدرس IP شبکه

ج: مشخص کننده‌ی فرآیند د: شماره(ها)ی پورت‌ها

۱۵. مشخصه‌ی یک ارتباط سنکرون چیست؟

الف: عملیات ارسال مسدود شده و برای عملیات دریافت صبر می‌کند.

ب: پیام ارسال شده همیشه دریافت می‌شود.

ج: عملیات ارسال فقط در فواصل زمانی معینی انجام می‌شود.

د: فرستنده و گیرنده باید ساعت‌های‌شان سنکرون باشد.

۱٦. مشخصه‌ی یک ارتباط آسنکرون چیست؟

الف:گیرنده بلاک می‌شود و صبر می‌کند تا پیام برسد.

ب: پیام ارسالی همیشه دریافت می‌شود. ج: بر اساس UDP اجرا می‌شود.

د: فرستنده بدون منتظر ماندن برای عملیات دریافت به کارش ادامه می‌دهد.

۱۷. آیا رابط ارتباط سنکرون می‌تواند به صورت آسنکرون استفاده شود؟

الف: بله، با اجرای عملیات ارسال به صورت نخ‌های مجزا.

ب: خیر، ارسال بلاک شده و عملیات متوقف می‌شود.

ج: خیر، کلاک‌های سنکرون همیشه زمان‌بر هستند. د: بله، با کم کردن تأخیر عملیات دریافت.

۱۸. تفاوت بین دو گره که در یک جریان سوکت ارتباط دارند، چیست؟

الف: گره‌ی که به سوکت سرویس‌دهنده متصل می‌شود گره‌ی است که باید سوکت را ببندد.

ب: گره‌ی که سوکت سرویس‌دهنده را می‌سازد باید سوکت را ببندد.

ج: هیچ چیز، هر دو حق و وظایف یکسانی دارند.

د: گره‌ی که به سوکت سرویس‌دهنده متصل است باید اولین پیام را بفرستد.

۱۹. Erlang، gen_tcp : recv (Socket , 0) را صدا می‌زند چه چیز بر می‌گردد؟

الف: یک پیام کامل همان طور که به سوکت ارسال شده بود.

ب: قسمت اول (یا ممکن است همه) پیام به سوکت ارسال می‌شود. ج: لیستی ۶۴ کاراکتری

د: همه‌ی پیام‌ها به سوکت ارسال می‌شوند قبل از اینکه توسط فرستنده بسته شوند.

۲۰. هدف از عملیات مارشال چیست؟

الف: بررسی سازگاری انواع اطلاعات ب: فشرده‌سازی ساختمان‌های داده

ج:محافظت کاربرد از درخواست‌های غیر مجاز د: رمزگذاری ساختار لایه‌ای کاربرد به صورت خارجی

۲۱. چگونه آرگومان‌ها به Java RMIپاس می‌شوند؟

الف: فقط با مرجع صدا زده می‌شوند ب: فقط با کپی صدا زده می‌شوند

ج: اشیا دور به صورت مرجع، بقیه به صورت کپی د: اشیا سریال شده به صورت مرجع و بقیه به صورت کپی

۲۲. کدام معنای احضار توسط Java RMI انجام می‌شود؟

الف: بدون تضمین ب: حداقل یک‌بار ج:دقیقاً یک‌بار د: حداکثر یک‌بار

۲۳. چه سطحی از شفافیت توسط متد احضار Java RMI فراهم می‌شود؟

الف: شفافیت دسترسی تنها زمانی که می‌خواهیم مکان دقیقاً توصیف شود.

ب: شفافیت مکانی تنها زمانی که می‌خواهیم استثنای از راه دور بگیریم.

ج: شفافیت دسترسی و مکانی د: نه شفافیت دسترسی و نه مکانی

۲٤. چگونه فرآیند Erlang بر قرار می‌شود؟

الف: تنها از طریق حافظه سراسری ب:ارسال پیام سنکرون

ج:ارسال پیام آسنکرون د: تغییر ساختار داده‌های به اشتراک گذاشته شده

پاسخ تست‌های تکمیلی

۱. **گزینه (ب) صحیح است.** گزینه (الف) شفافیت مکان و گزینه (ج) شفافیت تکثیر است.

۲. **گزینه (الف) صحیح است.** مطابق با پاسخ تست شماره ۱

۳. **گزینه (ب) صحیح است.** گزینه(الف) از ویژگی‌های نخ است و گزینه (د)، توسعه پذیری سیستم را بیان می‌کند.

۴. **گزینه (الف) صحیح است.**

۵. **گزینه (ب) صحیح است.**

۶. **گزینه (الف) صحیح است.** گزینه (ب) نادرست است، چون، بعضی اوقات ممکن است سرویس دهنده بیش‌ترین توان را نداشته باشد (مانند سرویس گیرنده کلفت). گزینه (ج) نیز نادرست است، چون این معماری ممکن است چند سرویس دهنده وجود داشته باشد (مانند معمار سه لایه‌ای) و گزینه (د) نیز نادرست است، چون در این معماری سرویس دهنده غیر فعال است.

۷. **گزینه (ب) صحیح است.**

۸. **گزینه (د) صحیح است.**

۹. **گزینه (د) صحیح است.**

۱۰. **گزینه (ج) صحیح است.**

۱۱. **گزینه (د) صحیح است.**

۱۲. **گزینه (ج) صحیح است.**

۱۳. **گزینه (ب) صحیح است.**

۱۴. **گزینه (د) صحیح است.**

۱۵. **گزینه (الف) صحیح است.**

۱۶. **گزینه (ب) صحیح است.** چون فرستنده بلاک نمی‌شود.

۱۷. **گزینه (الف) صحیح است.**

۱۸. **گزینه (ج) صحیح است.**

۱۹. **گزینه (ب) صحیح است.**

۲۰. **گزینه (د) صحیح است.**

۲۱. **گزینه (ج) صحیح است.**

۲۲. **گزینه (د) صحیح است.**

۲۳. **گزینه (ب) صحیح است.**

۲۴. **گزینه (ج) صحیح است.**

پیوست ب — سوالات دکتری سراسری سال‌های ۱۳۹۱، ۱۳۹۲، ۱۳۹۳ و ۱۳۹۴ و آزاد سال ۱۳۹۳همرا با پاسخ تشریحی

سوالات کنکور سراسری ۹۱

۱. کدام جمله در مورد یک Clone Process صحیح نیست؟

الف: در مهاجرت ضعیف کدها ممکن است ایجاد شود.

ب: فرآیندی دیگری وجود دارد که دقیقا مشابه آن است و به صورت پارالل با آن اجرا می‌شود.

ج: ایجاد این نوع فرآیند وابسته به این که مهاجرت کد از سوی چه کسی آغازدهی می‌شود (Server initiated/ Receiver initiated) نمی‌باشد.

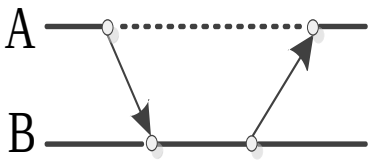
د: ب و ج صحیح نیستند.

۲. فرض کنید شبکه‌ای از کامپیوترها موجود است که در آن انواع مختلفی از سیستم-های عامل و پردازنده‌ها وجود دارد. برای فراهم آوردن امکان مهاجرت قوی کدها برروی این شبکه، استفاده از کدام مفهوم زیر را ضروری می‌دانید؟

الف: پشته مهاجرت (Migration)　　ب: Exokernel

ج: سرویس جابجایی منابع محلی (MV)　　د: سرویس امکان ترجمه مجدد کدها (Recompile)

۳. نمودار زیر روند برقراری ارتباط بین دو فرآیند A, B را به طور کلی نشان می‌دهد، کدام گزینه درست است؟



الف: این شکل نمایی از ارتباط مبتنی بر ارتباط گذرای غیر همگام (Transient Asynchronous) است.

ب: این شکل نمایی از ارتباط مبتنی بر RPC (Remote Procedure Call) است.

ج: این شکل نمایی از ارتباط ماندگار غیر همگام (Persistent Asynchronous) است.

د: این شکل می تواند نمایشی برای هر یک از سه گزینه فوق باشد.

٤. در ارتباط مبتنی بر جویبار (Stream)، وقتی که از یک جویبار پیچیده (Complex) استفاده می‌شود.

الف: جویبارهای ساده در سطح سیستم عامل همگام می‌شوند.

ب: برای همگام سازی جویبار های ساده از یک سمافور استفاده می‌شود.

ج: جویبارهای ساده ممکن است در سطح برنامه کاربردی (Application) همگام شوند.

د: همگامی جویبارهای ساده با استفاده از پروتکل RSVP در سطح شبکه انجام می‌شود.

۵. در یک DSM(Distributed Share Memory):

الف: افزایش اندازه صفحه، باعث کاهش تعداد جابجایی صفحات می‌شود، گر چه ممکن است حجم داده‌های جابجا شده افزایش یابد.

ب: افزایش اندازه صفحه، احتمال بروز **False Sharing** را افزایش می‌دهد که در این حالت تعداد جابجایی صفحات افزایش می‌یابد.

ج: کاهش اندازه صفحه باعث افزایش سربار سیستم عامل خواهد شد.

د: هر سه گزینه درست هستند.

٦. کدام عبارت توصیف مناسبی از Interoperability (عملیات متقابل) را ارائه می‌دهد؟

الف: این که دو فرآیند بتوانند با یک دیگر ارتباط برقرار و داده رد و بدل کنند.

ب: این که دو فرآنید بتوانند با یک دیگر همکاری کنند.

ج: این که پیاده سازی های مختلف و متفاوت از دو سیستم توسط شرکت‌های مختلف بتوانند با یک دیگر همکاری کرده و حضور هم زمان داشته باشند.

د: تمام موارد فوق

۷. فرض کنید در یک سیستم عامل چند رسانه‌ای، دو فرآیند متفاوب با مشخصات زیر وجود دارند. در چه صورت الگوریتم RMS اجرای این فرآیندها را گارانتی می‌کند؟

فرآیند	زمان پردازش (**C**)	دور تناوب
P1	15	30
P2	?	40

الف: C2 <20

ب: C2 <10

ج: 10< C2 <20

د: تمام موارد

۸. فرض کنید دو کاربر متفاوت در حال مشاهده یک فیلم هستند در حالی که اختلاف زمانی در حد ۵ ثانیه با یک دیگر دارند. اگر فرض کنید که نرخ استاندارد ارسال فریم‌ها برابر ۳۰ فریم در ثانیه باشد. با فرض این که حداکثر تغییر ممکن در نرخ فریم ارسالی ±20% می‌تواند باشد، حداقل زمان لازم برای هماهنگ کردن دقیق این دو کاربر اگر از تکنیک تغییر فریم استفاده شود، چقدر است؟

الف:۱۲/۵ ثانیه ب: ۶ ثانیه ج: ۱۰ ثانیه د: ۲۵ ثانیه

۹. فرض کنید، سه فایل چند رسانه‌ای برروی یک دیسک با استفاده از الگوریتم ارگان پاپپ قرار گرفته‌اند. اگر درجه مشهوریت یکی از این فیلم‌ها برابر یک و دوتای دیگر برابر ۲ باشد. مطابق قانون Zipf فاکتور نرمال سازی این مجموعه کدام است؟

الف: ۲ ب: $\frac{1}{2}$ ج: $\frac{1}{3}$ د:۳

پاسخ سوالات کنکور سراسری ۹۱

۱. گزینه (الف) صحیح است. چون در مهاجرت ضعیف فقط انتقال بخش کد و احتمالاً داده‌هایی که برای مقداردهی اولیه مورد نیازاست، امکان‌پذیر می‌باشد (یعنی، کد ایجاد نمی‌شود، بلکه انتقال می‌یابد). اما، در فرآیند همانندسازی (Clone)، به جای انتقال فرآیند در حال اجرا، یک کپی دقیق از فرآیند اصلی می‌سازد که روی ماشین متفاوتی اجرا می‌شود، پس گزینه (ب) صحیح است. از طرف دیگر، طبق شکل ۶-۳ فرآیند همانندسازی در هر دو صورت فرستنده آغاز گر و گیرنده آغازگر می‌باشد و وابسته به این که کد از سوی چه کسی آغاز می‌شود، نمی‌باشد.

۲. گزینه (الف) صحیح است. چون در این شبکه انواع مختلفی از سیستم‌های عامل و پردازنده وجود دارند، پس ناهمگن می‌باشد. بنابراینباید از مهاجرت در سیستم‌های ناهمگن استفاده کرد. در این سیستم، مهاجرت قوی از پشته مهاجرت استفاده می‌کند. در مهاجرت قوی در سیستم ناهمگن، کد سیستم کپی خودش را در پشته برنامه به طور مستقل در ماشین خودش نگه داری می‌کند، وقتی یک روال صدا زده می‌شود یا از یک روال برمی‌گردد، پشته مهاجرت به هنگام می‌گردد. این مراحل را در شکل زیر می‌بینید (این روش مهاجرت در زبانهای روالی نظیر C و جاوا کار می‌کند.

۳. گزینه (ب) صحیح است. گزینه‌های (الف) و (ج) نادرست‌اند، چون در یک ارتباط غیر همگام، فرستنده‌ای که پیام را می‌فرستند، عملیات مربوط به خودش را پس از دریافت تأییدیه (AKC) ادامه می‌دهد. اما، در این شکل فرستنده، انسداد می‌شود تا پاسخ را دریافت کند و سپس به کارش ادامه می‌دهد.

٤. گزینه (ج) صحیح است. دو روش برای همگام سازی جویبار (Stream) وجود دارد. ۱. **برنامه کاربردی (Application)**، در سمت دریافت کننده، چندین جویبار به سمت برنامه کاربردی وارد می‌شوند، و برنامه کاربردی نحوه ترکیب جویبارها را با یک دیگر مدیریت می‌کند. ۲. واسطی تعبیه می‌شود که این واسط عملیات خواندن و نوشتن و همگام سازی جویبارها را مدیریت می‌کند. در واقع، واسط می‌تواند یک میان افزار باشد که در خارج از برنامه کاربردی قرار می‌گیرد. بنابراین سطح همگام سازی می‌تواند از طریق سطح برنامه کاربردی یا سطح واسط باشد.

۵. گزینه (د) صحیح است.چون اگر اندازه صفحات کوچک‌تر انتخاب شوند، عمل انتقال زیادی خواهیم داشت که کارایی سیستم عامل را کاهش می‌دهد (سربار سیستم عامل افزایش می‌یابد). اما، اگر اندازه صفحه بزرگ انتخاب شود، در شکل False Sharing (یعنی وجود داده‌هایی متعلق به دو فرآیند در یک صفحه) رخ می‌دهد.

٦. گزینه (ج) صحیح است. Interoperability، یعنی، پیاده‌سازی متفاوت از دو سیستم توسط شرکت‌های مختلف بتوانند با هم همکاری و حضور هم زمان داشته باشند (در این تعریف پیاده‌سازی دو سیستم متفاوت است). اما، Portability (قابلیت حمل)، ویژگی است که برای توسعه است که یک برنامه کاربردی نوشته شده در سیستم توزیع شده A بدون این که تغییر کند، بر روی سیستم توزیع شده B اجرا شود (با همان واسط A).

۷. گزینه (ب) صحیح است. با توجه به الگوریتم RMS، مقدار استفاده (utilization) به صورت زیر محاسبه می‌شود:

که در آن، n، تعداد فرآیندها، C_i، زمان اجرا و T_i، پریود است. پس داریم:

$$\frac{15}{30}+\frac{x}{40}\le 2\sqrt{2}-1=\frac{1}{2}+$$
$$\le 80\times($$
$$\Longrightarrow x\le 3$$

۸. گزینه (ب) صحیح است. از آنجایی که بیان کرده نرخ استاندارد ارسال فریم‌ها ۳۰ فریم در ثانیه است و دو کاربر اختلاف زمانی ۵ ثانیه با یک دیگر دارند، پس این دو کاربر ۱۵۰ (۵ × ۳۰) فریم اختلاف دارند. از طرف دیگر، در عنوان سوال بیان شده است که نرخ ارسال فریم‌ها را می‌توان ۲۰ درصد کاهش یا افزایش داد، پس اگر نرخ ارسال فریم یک کاربر را ۲۰ درصد افزایش و نرخ ارسال فریم کاربر دیگر را ۲۰ درصد کاهش دهیم، پس از گذشت مدت زمان ۱۲ ثانیه این دو کاربر با هم هماهنگ می‌شوند. یعنی، افزایش و کاهش تعداد فریم‌های برابر با $\frac{20\times30}{100}$است که ۶ فریم می‌باشد. اگر نرخ ارسال فریم را برای یک کاربر ۶ واحد کم و برای کاربر دیگر ۶ واحد اضافه کنیم، در هر ثانیه ۱۲ فریم اختلاف ایجاد خواهد شد. چون ۱۵۰ فریم داریم، پس ۱۵۰ تقسیم بر ۱۲ (یعنی ۱۲/۵ ثانیه) حداقل زمان لازم برای هماهنگ کردن دقیق این دو کاربر خواهد بود.

۹. گزینه (ب) صحیح است. چون در قانون ZIF اگر N فایل چند رسانه‌ای بر روی یک دیسک داشته باشیم، فاکتور نرمال سازی این مجموعه برابر با $\frac{C}{k1}+\frac{c}{k2}+\cdots+\frac{c}{kn}=1$ (K، رده محدودیت است) می‌باشد. چون، درجه مشهوریت یکی از فیلم‌های برابر یک و دو تای دیگر برابر ۲ است، پس داریم:

$\underline{2}$

پیوست ج پرسش‌های حل شده در سایت

۱. توضیح دهید چگونه استفاده از توزیع داده‌ها و تکثیر به عنوان تکنیک‌های توسعه پذیری، مشکلات توسعه پذیری جدیدی را ایجاد می‌کنند.

۲. برای بهبود توسعه پذیری سرویس دهنده وب سایت غالباً کلاسترها استفاده می‌شوند. چرا این روش برای اجرای توزیع درخواست افزار – محتوی (Content-ware)، در این حالت‌ها مفید است و عیب اصلی در انجام آن چیست؟

۳. تحت چه شرایطی، استفاده از عامل‌های سیار برای رفع مشکلات توسعه پذیری مفید است؟ آیا علل دیگری وجود دارد که چرا عامل‌های سیار مفید هستند؟

۴. آداپتور شیء چه کاری را انجام می‌دهد؟

۵. تفاوت بین ره‌گیری سطح درخواست و ره‌گیری سطح پیام را همان گونه طبق مثال CORBA استفاده می‌شوند، توضیح دهید.

۶. سیستمی را در نظر بگیرید که فقط اشیاء راه دور نظیر CORBA را پشتیانی می‌کند. طرح توسعه را که از ره‌گیری های سطح درخواست برای پیاده سازی پروتکل پشتیبان- اولیه استفاده می‌کند، خلاصه کنید (از ره گیر سمت سرویس گیرنده استفاده نکنید).

۷. چگونه می‌توان سندی را به طور موفقیت آمیزی توسط کاربر غیر مجاز استفاده می‌شود، محافظت نمود؟

۸. سیستم‌های رمزگذاری کلید عمومی برای توسعه بهتر از سیستم‌های کلید مشترک درخواست می‌شوند، آیا این درخواست واقعاً نگه داری می‌شود؟ (تذکر: در مورد لغو سند فکر کنید).

۹. اصول سیستم انتشاری- اشتراکی را توضیح دهید.

۱۰. IBM MQ Series و TIB/Rendezvous دو روش متفاوت پیاده سازی سیستم انتشاری- اشتراکی دارند، شرح دهید.

۱۱. توضیح دهید سیاست‌های جداسازی مکانیزم‌ها در سیستم‌ها توزیع شده به چه معنی است و یک مثال از جداسازی ذکر کنید.

۱۲. معماری چند لایه‌ای سرویس گیرنده- سرویس دهنده چیست؟

۱۳. تفاوت اصلی بین سیستم‌های صف بندی پیام و سیستم‌های پست الکترونیکی (Email) چیست؟

۱۴. علاوه بر رابط‌های کاربر، چه چیزی در سیستم‌های توزیع شده به عنوان نرم‌افزار سمت سرویس گیرنده در نظر گرفته می‌شود؟

۱۵. چگونه سرویس دهنده بدون حالت (Stateless Server) اطلاعات روی سرویس گیرنده‌های خود را نگه‌داری می‌کند؟

۱۶. سیستم توزیع شده را در نظیر بگیرید که هر شی سیار یک یا چند نام انسان پسند و نیز یک یا چند آدرس دارد. برای مکان یابی آدرس شیء از خود نام استفاده می‌کند. معرفی شناسه شیء مستقل از مکان مفید می‌باشد، چرا؟

۱۷. مهرهای زمانی لامپورت، علّی را ذخیره نمی‌کنند، شرح دهید.

۱۸. توضیح دهید توسعه پذیری سرپرستی به چه معنی است و چرا اغلب چنین مشکلی برای حل وجود دارد؟

۱۹. تفاوت اصلی بین سیستم عامل شبکه و سیستم توزیع شده چه مواردی می‌باشند؟

۲۰. حداقل دو مثالی از پروتکل میان افزار را نام ببرید.

۲۱. استفاده گسترده از نخ‌ها در سیستم‌های توزیع شده را توضیح دهید.

۲۲. عامل سیار را در نظر بگیرید که نخ کنترلی خود را دارد. آیا وقتی که عامل به ماشین دیگر مهاجرت می‌کند، جابه‌جایی قوی داریم یا جابه‌جایی ضعیف؟جواب خود را توضیح دهید.

۲۳. توضیح دهید در سیستم‌های نام گذاری، مکانیزم بسته چیست و حداقل یک مثال بزنید.

۲۴. تبدیل نام تکراری و بازگشتی در سیستم نام گذاری توزیع شده با واسطه‌ای توسعه پذیری متفاوتی دارند، توزیع دهید مهم‌ترین تفاوت چیست؟

۲۵. توضیح دهید در هنگام سر و کار داشتن رونوشت توزیع شده (Distributed Snapshot)، حالت سراسری سازگار به چه معنی است؟

۲۶. یک فایل در ۱۰ سرویس دهنده تکثیر می‌شود. تمام ترکیبات حد نصاب خواندن و نوشتن را لیست کنید که از طریق پروتکل مبتنی بر حد نصاب در هنگام سروکار داشتن با نوشتن‌های تکثیر شده میسر می‌گردد.

۲۷. نشان داده شده که پروتکل‌های میان افزار برای نیل به سازگاری علّی باید از طریق پروتکل‌های سطح کاربردی جایگزین شوند. دلایل قوی چنین جایگزینی در چیست؟

۲۸. توضیح دهید چند پخشی مجازی هم زمان چیست و مزیت این طرح چیست؟

۲۹. طرح کارنامه گیری پیام بدبینانه چیست و چرا نسبت به روش‌های خوش بینانه ترجیح داده می‌شود.

۳۰. دو روش متفاوت برای پیاده سازی سیستم اشتراکی انتشاری مبتنی بر شیء را نام ببرید.

۳۱. تفاوت معناشناسی بین نوشتن یک نمونه تاپل با یک Java Space به طور شفاف N بار تکثیر شده و نوشتن نمونه تاپل N بار با Java Space بدون تکثیر شده چیست؟

۳۲. توضیح دهید شفافیت جا به جایی چیست و مثال واقعی از کاربرد آن بزنید.

۳۳. را حل ساده‌ای جهت پشتیبانی شفافیت مهاجرت برای اشیا را دور سیار را به طور خلاصه بیان کنید.

۳۴. تفاوت بین یک سرویس نام و یک سرویس دایرکتوری را توضیح دهید و چرا مورد دوم اغلب برای توسعه در جهان مشکل است؟

۳۵. تفاوت اصلی بین مدل‌های سازگاری مبتنی بر داده محور و مبتنی بر سرویس گیرنده محور چیست؟

۳۶. توضیح دهید که چند پخشی اتمیک چیست؟

۳۷. مدل اجرای قطعی تکه‌ای (Piece wise deterministic execution model) چیست؟

۳۸. چگونه توزیع همانند تکنیک توسعه پذیری به بهبود عملکرد سیستم توزیع شده کمک می‌کند؟

۳۹. کدام مسئله جدید کارایی ممکن است همانند تکنیک توسعه پذیری، سیستم توزیع، احتمالاً سیستم توزیع جهانی را معرفی می‌کند؟

۴۰. توضیح دهید معنا شناسی حداکثر یک بار (At-most-once) در مورد فراخوانی راه دور به چه معنی است؟

۴۱. نقش واسط پیام، در ارتباط مبتنی بر پیام چیست؟

۴۲. تفاوت بین مهاجرت قوی و مهاجرت ضعیف چیست؟

۴۳. جاوا استاندارد فقط می‌تواند مهاجرت ضعیف را پشتیبانی کند. چگونه مهاجرت قوی بدون تغییر ماشین مجازی جاوا را ایجاد می‌کنید؟

۴۴. توضیح دهید دقیقاً ارسال پیام- علّی مرتب شده به چه معنی است؟

۴۵. این عبارت که سیستم‌های توزیع شده به وسیله وقوع خرابی‌های جزئی مشخص می‌شوند، به چه معنی است؟

۴۶. معماری سرویس گیرنده/ سرویس دهنده چند لایه‌ای چیست؟

۴۷. ارتباط بین فرآیندها را می‌توان در ابعاد مختلف مشخص کرد برای هر یک از ترکیبات مثال بزنید.

	گذرا (ناپایدار)	دائمی (پایدار)
همگام	RPC/RMI	صف بندی پیام با تأییدیه
غیر همگام	پیام متنی بر سو کت	صف بندی پیام بدون تأییدیه

۴۸. توضیح دهید چند پخشی ترتیبی علّی چیست؟

۴۹. توضیح دهید شناسه صحیح (TRUE) چیست و مثالی ارائه دهید.

۵۰. آیا نام شیء شامل آدرس MAC مربوط به ماشین است جایی که شیء روی مکان مستقل ایجاد شده باشد؟ پاسخ خود را توضیح دهید.

۵۱. مزیت اصلی شناسه‌های صحیح به عنوان واسطه‌ها (میانی‌ها) در نگاشت نام و آدرس‌ها چیست؟

۵۲. سازگاری متوالی چیست؟

۵۳. در سیستمی که سازگاری مشروط (eventual) را تضمین می‌کند برخورد نوشتن – نوشتن چگونه اداره می‌گردد.

۵۴. هنگامی که یک سرویس گیرنده فایلی را فقط برای خواندن باز کند، سرویس دهنده می‌تواند یک کپی به سرویس گیرنده ارسال کند. کدا (Coda) به سرویس گیرنده اجازه می‌دهد تا خواندن فایل را ادامه دهد، حتی اگر در این میان به روز شود، چرا این رفتار را می‌توان درست تلقی کرد؟

۵۵. آدرس دهی مبتنی بر شی چیست و چگونه می‌توان آن را با استفاده از واسطه‌های پیام پیاده‌سازی کرد؟

۵۶. توضیح دهید چرا پیاده سازی یک java Space توزیع شده کار آمد مشکل است؟

۵۷. معنی سیستم توزیع شده باز چیست؟

۵۸. معنی معماری سرویس گیرنده / سرویس دهنده سه لایه‌ای چیست؟

۵۹. یک سرویس گیرنده را در نظر بگیرید که RPC را اجرا می‌کند، اما سرویس دهنده قبل از این که بتواند پاسخی بفرستد، دچار خرابی می‌شود، این سرویس گیرنده چه کاری باید انجام دهد؟

۶۰. تفاوت بین سرویس دهنده تکراری و هم روند چیست؟

۶۱. توضیح دهید که یک سرویس دهنده سوپر (supper server) چگونه کار می‌کند؟

۶۲. مثالی از مهاجرت کد گیرنده –آغاز گر ارائه دهید؟

۶۳. توضیح دهید مهره‌های زمانی لامپورت چگونه کار می‌کنند.

۶۴. شرح دهید که چگونه می‌توان چند پخشی مرتب شده کامل (totally-ordered multicasting) با مهر های زمانی لامپورت را پیاده‌سازی کرد.

۶۵. تفاوت اصلی بین کش کردن و تکثیر چیست؟

۶۶. تکثیر فعال چیست؟

۶۷. فرق بین نقطه بازرسی مستقل و هماهنگ چیست؟

۶۸. آیا NFS نسخه مدل دسترسی راه دور را دنبال می‌کند یا مدل آپلود / دانلود؟ پاسخ خودتان را توضیح دهید.

۶۹. معانی نشست در سیستم‌های فایل توزیع شده چیست؟

۷۰. توسعه پذیری جغرافیایی به چه معنی است و چگونه این مسئله قابل حل می‌باشد؟

۷۱. می‌توان بررسی نمود که سیستم‌های نظیر به نظیر، مشکل توسعه پذیری مدیریتی را حل می‌کنند، توضیح دهید، چگونه؟

۷۲. تکثیر گسترده به عنوان یک تکنیک توسعه پذیری انجام می‌شود. اما مشکل توسعه پذیری جدیدی را مطرح می‌کند، توضیح دهید، این مسئله چه چیزی است و چگونه قابل حل است؟

۷۳. حداقل یک دلیل قاطع ذکر کنید که چرا استفاده از عاملان سیار، ایده مناسبی خواهد بود.

۷۴. آیا مکان http://www.cs.vu.nl URL مستقل است. مکان Http://www.van-steen.net چه طور، در مورد پاسخ‌تان توضیح دهید.

۷۵. عموماً فضای نام پیوندهای سخت و پیوندهای نمادی را فراهم می‌کند، فرق بین این‌ها را توضیح دهید؟

۷۶. طرح متمرکز و ساده‌ای جهت اجرای چند پخشی مرتب کلی قابل اطمینان شرح دهید.

۷۷. سرویس دهنده وب را در نظر بگیرید که قرار دادهایی را برای بی اعتبار کردن کش طرف سرویس گیرنده توزیع می‌کند. چه زمانی باید فرکانس تکرار مبتنی بر قرار دادها جهت کسب رضایت مفید بودن اجرا شوند، دوره قرارداد را در جواب خود قرار دهید.

۷۸. توضیح دهید چگونه چند پخشی قابل اعتماد مبتنی بر NACK (توسعه پذیر) کار می‌کند.

۷۹. مدل دسترسی راه دور در سیستم‌های فایل توزیع شده کم‌تر رایج می‌شود. چرا؟

۸۰. مزیت مهم استفاده واسط پیام برای پیاده سازی سیستم انتشاری/ اشتراکی چیست؟

۸۱. چه چیزی سیستم‌های انتشاری اشتراکی مبتنی بر شی را برای توسعه در مسیر شبکه گسترده سیار مشکل می‌سازد؟

۸۲. مثالی از معماری سه لایه‌ای سرویس گیرنده – سرویس دهنده ارائه دهید.

۸۳. علیرغم این که معماری چند لایه‌ای واقعاً هیچ مشکل مربوط به سیستم‌های توزیع شده را حل نمی‌کنند، اما، آن‌ها یک مزیت عملی دارند. این مزیت چیست؟

۸۴. تفاوت بین انتقال همگام (هم زمان) و تک زمان چیست؟

۸۵. همگام سازی جریان چیست؟

۸۶. فرق بین سرویس دهنده‌های تکراری و هم زمان چیست؟

۸۷. توضیح دهید چگونه یک سرویس دهنده شیء را سازماندهی می‌کنید تا این که بتواند به طور هم زمان روش تکراری و هم زمان کنترل درخواست‌ها را برای اشیا مختلف پشتیبانی کند.

۱۲۳. اصول کار فراخوانی رویه راه دور (RPC) را توضیح دهید.

۱۲۴. دو دلیل قاطع ذکر کنید که چرا RPC هرگز نمی‌تواند همان معنی فراخوانی رویه محلی را مهیا کند؟

۱۲۵. تفاوت اصلی بین احضار متد راه دور (RMI) و RPC چیست؟

۱۲۶. یک سرویس دهنده فایل را تصور کنید که جدول جفت‌های (فایل و سرویس گیرنده) را نگه داری و شناسایی می‌کند که کدام سرویس گیرنده به کدام فایل دسترسی دارد. آیا این سبک سرویس دهنده با حالت (StateFul) است (پاسخ خود را توضیح دهید).

۱۲۷. توضیح دهید سرویس دهنده صف بندی پیام چیست و آیا می‌تواند برای بدون حالت (Stateless) طراحی شود؟

۱۲۸. سرویس گیرنده‌ای را در نظر بگیرید تا زمان را به سرویس دهنده در هر دقیقه یک بار هم زمان می‌کند. این سرویس گیرنده تعدادی از در خواست‌ها را همراه با نتایج نشان داده شده در بخش پایین ارسال می‌کند. چگونه سرویس گیرنده ساعت خود را پس از دریافت پاسخ تنظیم می‌کند؟ زمان پردازش در سرویس دهنده صفر است.

۱۲۹. اصول کار الگوریتم برکلی را برای تنظیم ساعت‌ها در سیستم‌های توزیع شده بیان کنید.

۱۳۰. لایه ارتباطی در سیستم‌های توزیع شده می‌تواند ردیابی (مسیر) پیام‌های علّی مرتبط را نگه داری کند؟ دو ایراد اصلی در برابر کار آمدی چیست؟

۱۳۱. آیا ترتیب رویدادهای زیر با ذخیره (انباره) سازگاری متوالی امکان پذیر است. در مورد انباره داده سازگاری علّی چه طور؟ پاسخ خود را توضیح دهید.

P1:	W(x)a		W(x)c	
P2:	R(x)a	W(x)b		
P3:	R(x)a		R(x)c	R(x)b
P4:	R(x)a		R(x)b	R(x)C

۱۳۲. توضیح دهید چرا سازگاری نوشتن پس از خواندن تضمین می‌کند که روابط علّی زمانی که برای پیاده‌سازی سیستم خبری توزیع شده مورد استفاده قرار می‌گیرد، نگه داری می‌شوند.

۱۳۳. شرایط جلوگیری از برخوردهای خواندن – نوشتن و نوشتن – نوشتن در سیستم مبتنی بر حد نصاب چیست؟

۱۳۴. NFS مسلماً یک سیستم فایلی نیست. توضیح دهید چرا؟

۱۳۵. کودا خواندن سرویس گیرنده را جهت ادامه انجام کپی محلی میسر می‌سازد. حتی اگر نوشتن هم روند در سرویس دهنده اتفاق افتد، چرا این رفتار درست تلقی می‌گردد؟

۱۳۶. پنج مثال از سیاست‌های که مکانیزم‌ها در سیستم‌های توزیع شده متمایز شده است، ارائه دهید.

۱۳۷. توضیح دهید مشکلات مقیاس پذیری جغرافیایی به چه معنی است و مثالی ارائه دهید که چگونه چنینی مشکلات قابل حل می‌باشند؟

۱۳۸. وب سایت‌ها اغلب به عنوان سیستم سرویس گیرنده – سرویس دهنده چند لایه‌ای سازمان دهی می‌شوند کاربرد و سرویس دهنده پایگاه داده می‌باشند. مزایای چنین معماری چیست؟

۱۳۹. عیب اصلی استفاده از RPC در مقایسه با پیام دهی همانند سیستم‌های صف بندی پیام چیست؟

۱۴۰. توضیح دهید مسیر یابی مبتنی بر موضوع (Subject base routing) چیست و چگونه واسط‌های پیام (message brokers) می‌توانند برای پیاده سازی آن استفاده شوند؟

۱۴۱. شبکه هم پوشانی جهانی بسیاری از واسطه‌های پیام متصل به مجموعه IBM MQ را در نظر می‌گیرد. راه حل ساده‌ای برای پشتیبانی از چند پخشی در چنین محیطی را به طور خلاصه بنویسید.

۱۴۲. آیا وجود برخورد نوشتن – نوشتن (write-write) در مدل‌های سازگاری سرویس گیرنده- محور ممکن است؟ پاسخ خودتان را توضیح دهید.

۱۴۳. توضیح دهید چند پخشی همگام مجازی چیست؟

۱۴۴. در همگام‌سازی مجازی، می‌توانیم ترتیب ارسال پیام را در یک گیرنده از این حقیقت که پیام در معرض ارسال ترتیب کلی هستند جدا کنیم. فرق بین این دو چیست؟

۱۴۵. تفاوت بین قفل گذاری دو مرحله‌ای (2pl) و تثبیت دو مرحله‌ای (2pc) چیست؟

۱۴۶. گفته می‌شود که تثبیت دو مرحله‌ای (2pc) یک پروتکل مسدود کننده می‌باشد، به همین دلیل تثبیت سه مرحله‌ای (3pc) ایجاد گردید. چگونه این بلاک اتفاق می‌افتد؟

۱۴۷. مثالی ارائه دهید که یک سرویس گیرنده چند نخی بتواند شفافیت توزیع را در مقایسه با سرویس گیرنده تک نخی بهبود بخشد.

۱۴۸. تکرار فعال عموماً نیازمند ارسال پیام ترتیب کلی می‌باشد. آیا این شرایط در مورد سرویس دهنده-های چند نخی تکثیر شده فعال کافی است؟ پاسخ تان را توضیح دهید.

۱۴۹. رابطه بین یک شناسه گره و کلید را درChord توضیح دهید.

۱۵۰. در Chord، جدول انگشتی برای گره P با FTp[i] = Succ(p+2i-1) تعریف می‌شود. یک شناسه ۳۲ بیتی را فرض کنید و جدول انگشتی زیر را برای گره ۱۸ در نظر بگیرید. توضیح دهید گره ۱۸ درخواست جست وجو ی برای کلیدهای زیر K= 26, 20, 18, 17, 29 را به کجا ارسال می کند:

FT18=[20, 20, 28, 28, 4]

درخواست به ترتیب 20, 20, 18, 4, 28 ارسال می‌شوند.

۱۵۱. توضیح دهید یک گره (با شناسه یکتا p) به سادگی می‌تواند به حلقه Chord بپیوندد.

۱۵۲. سازگاری متوالی به چه معنی است؟

۱۵۳. در هنگام استفاده از همگام سازی، مهم‌ترین تأثیر در کارایی چیست؟ پاسخ خود را توضیح دهید.

۱۵۴. سیستمی را در نظر بگیرید که اشیا راه دور تکثیر شده‌اند و سازگاری ورودی را تضمین می کند. در پایان، تمام احضارهای متد در سطح سیستم در مدل با ترتیب کامل می‌شوند. آیا این پیاده سازی ضمانت-های سازگاری قوی‌تری را فراهم می کند؟ پاسخ خودتان را توضیح دهید.

۱۵۵. برای پشتیبانی از کاربردهای وب، یک CDN می‌تواند کش کردن مطلع – محتوی (Context-aware) را توسعه دهد، توضیح دهید، چگونه؟

۱۵۶. حداقل یک تکنیک دیگر از تکثیر یا کش کردن ارائه دهید که بتواند برای حل مشکلات توسعه پذیری جغرافیایی استفاده شود.

۱۵۷. کش کردن و تکثیر از تکنیک توسعه پذیری مهم هستند، اما مشکل توسعه پذیری دیگری ایجاد می کنند. آن مشکل چیست و چگونه می‌توان آن را حل نمود؟

۱۵۸. سیستم‌های توزیع شده غالباً بر اساس فرضی نادرست که توپولوژی شبکه تغییر نمی کند، طراحی می‌شوند.

۱۵۹. یک گروه ویژه از معماری‌های سرویس گیرنده- سرویس دهنده دو لایه‌ای از سرویس گیرنده‌های چاق به سرویس گیرنده‌های نازک (لاغر) در حرکت هستند و به طور موثری مسئولیت کم‌تری طرف سرویس گیرنده قرار می‌دهد. علت این تغییر چیست؟

۱۶۰. مثالی از سازمان وب سایت در معماری سه لایه‌ای ارائه دهید.

۱۶۱. تفاوت اصلی بین وب سرویس و وب سایت قدیمی چیست؟

۱۶۲. فرض کنید گره‌ها در سیستم‌های همتا به همتا نظیر Chord ارجاعات متمایزی برای predecessor در حلقه نگه داری نمی‌کنند. چطور predecessor یک گره را جست‌وجو می‌کند؟

۱۶۳. در Chord تفاوتی بین جست وجو ی تکراری و بازگشتی وجود دارد، کدام یک بهتر است؟

۱۶۴. توضیح دهید معنی سازگاری علّی داده- محور چیست؟

۱۶۵. توضیح دهید زمانی که ارسال پیام‌ها لزوماً ایده مناسبی نیست، چرا میان افزار از علیت محافظت می‌کند؟

۱۶۶. سیستمی را در نظر بگیرید که ساعت‌های برداری را برای تایید ارتباط علیتی نگه داری می‌کند، ts(m) نشان دهنده مهر زمانی (بردار) ارسالی با پیام m و VC_i، ساعت برداری برای فرآیند i می‌باشد. یک پیام از فرآیند Iبه فرآیند jارسال می‌شود که (1) $ts(m)[i]=vc_j[i]+1$ و (2)$ts(m)[k]<= c_j[k]$ برای تمام k#i باشد. تعریف صحیح این دو شرط را ارائه دهید.

۱۶۷. سیاست فعال سازی برای آداپتور شی چیست؟

۱۶۸. مثالی ارائه دهید که چگونه درجه بالایی از شفافیت توزیع می‌تواند تأثیر ناسازگاری روی کارایی داشته باشد.

۱۶۹. مشکلات اصلی توسعه پذیری در کاربرد کنفرانس ویدئویی چیست؟

۱۷۰. یک RPC، شکل موقت (گذار)، همگام ارتباط است. این به چه معنی است و عیب اصلی‌اش چیست؟

۱۷۱. یک RPC با یک سرویس دهنده تکثیر شده را می‌توان با توجه به شفافیت دسترسی، تکثیر و خرابی فراخوانی کرده و فراخوانی شونده را با درجه بالایی شفاف نمود. پاسخ خود را توضیح دهید.

۱۷۲. نشان دهید که ساعت‌های منطقی لزوماً روابط علی Potentially را ذخیره نمی‌کنند.

۱۷۳. در هنگام استفاده از ساعت برداری برای تایید علی چند پخشی مرتب شده، $VC_i[i]$ فقط زمانی که P_i پیامی را ارسال می‌کند و VC_i را به مهر زمانی TS(m) با پیام m می‌فرستد، ساعت برداری افزایش می‌یابد.

چگونه باید دو شرط را برای ارسال پیام M در هنگام دریافت با فرایند P_j تفسیر کنیم.

(1). $ts(m)[i] = VC_j[i] + 1$ (2). $ts(m)[k] <= VC_j[k]$ برای k #i

۱۷۴. همیشه با دقت در مورد ردیابی روابط علّی "potentially" با میان افزار صحبت می‌کنیم، چرا "potential"؟

۱۷۵. مدل Upload/download از نشست‌های FTp شناخته می‌شود، اما روش خود را برای سیستم‌های فایل توزیع شده پیدا کرده است. کجا و چگونه می‌توان از آن استفاده کرد؟

۱۷۶. در Coda به سرویس گیرنده اجازه داده می‌شود تا به خواندن فایل ادامه دهد، علی رغم این که سرویس دهنده اطلاع دارد به روز رسانی اتفاق افتاده است. بحث و بررسی کنید چرا این حالت سازگاری را نقض نمی‌کند؟

۱۷۷. تفاوت بین کش کردن محتوی آگاه (content aware) و محتوی ناآگاه (content blind) برای کاربردهای وب چیست؟

۱۷۸. تکثیر کردن (پایگاه داده از) یک کاربرد وب در عرض یک سرویس دهنده لبه ممکن است حقیقتاً موجب کندی اجرای کاربرد شود، چرا این گونه است؟

۱۷۹. برخی سیستم‌های توزیع شده به سادگی قابل توسعه پذیری نیستند، مهم نیست کدام تکنیک‌ها استفاده می‌شوند. در مورد چنین سیستم مثالی ذکر کنید و توضیح دهید که چرا توسعه پذیری غیرممکن است؟

۱۸۰. توضیح دهید چگونه کد ارسالی که با اپلت‌های جاوا در مورد سرویس‌های وب انجام می‌شود، می‌تواند به بهبود توسعه پذیری کمک کند. کدام مشکل توسعه پذیری حقیقتاً قابل حل است؟

۱۸۱. تفاوت بین ماشین مجازی فرآیند و ناظر ماشین مجازی را توضیح دهید.

۱۸۲. توضیح دهید چگونه تکنولوژی ماشین مجازی در سستم Planet lab استفاده می‌شود و چگونه می‌تواند از چندین کاربرد پشتیبانی کند؟

۱۸۳. ماشین‌های مجازی به مهاجرت کردن کاربردها بین سرویس دهنده‌ها مختلف کمک می‌کند. چرا این حالت آسان‌تر از راه حل‌های عمومی برای مهاجرت قوی است؟

۱۸۴. اجرایRPC نیازمند به یک سرویس گیرنده است که بتواند با سرویس دهنده‌ی تماس بگیرد. چگونهRPC این نقطه تماس را برای سرویس دهنده پیدا می‌کند و این نقطه تماس شامل چیست؟

۱۸۵. سیستم‌هایRPC نمی‌توانند از ارجاعات محلی (مانند اشاره‌گرها) پشتیبانی کنند. زیرا، این‌ها به اشیایی که صرفاً به طور محلی دسترسی دارند، اشاره می‌کنند. در عوض، ارجاع‌های شی عمومی در صورت ممکن باید مورد استفاده قرار گیرند. پیاده‌سازی چنین ارجاعات را توضیح دهید.

۱۸۶. اصول پروتکل اپیدمیک را توضیح دهید.

۱۸۷. سیستمی را در نظر بگیرید که سازگاری خواندن نوشتن‌های‌تان با سازگاری نوشتن‌های پس از خواندن‌ها را ترکیب می‌کند. آیا این سیستم نیز به طور متوالی سازگار است؟ پاسخ خود را توضیح دهید؟

۱۸۸. توضیح دهید ازدحام فلش چیست؟

۱۸۹. بهترین عمل علیه ازدحام فلش چیست؟ به طور خلاصه چگونگی عملکرد آن را توضیح دهید.

۱۹۰. مفهوم ره‌گیری‌های به کار رفته در میان افزار را توضیح دهید و بیان کنید چرا می‌توانند مفید باشند.

۱۹۱. برخلاف اشاره‌گرهای محلی، داشتن ارجاعات به شیء در سطح سیستم وسیع (system-wide) به بهبود شفافیت دسترسی در RPC کمک می‌کند. چگونه می‌توان چنین ارجاعات شیء را پیاده‌سازی کرد. نکته: در نظر بگیرید چگونه جاوا فراخوانی‌های متد راه دور را تشخیص می‌دهد.

۱۹۲. در RPC، یک سرویس گیرنده نیاز به انقیاد (Bind) به یک سرویس گیرنده را دارد. این جمله به چه معنی است و چگونه قابل شناسایی است؟

۱۹۳. دو روش ذکر کنید که در Chord برای پیاده‌سازی سیستم فایل توزیع شده استفاده کند.

۱۹۴. طرح پروتکل سازگاری متمرکز ساده را برای تکثیر فعال که سازگاری متوالی را تشخیص می‌دهد، به طور خلاصه بیان کنید.

۱۹۵. می‌توان دلیل آورد که یک NFS واقعاً یک سیستم فایل نیست، توضیح دهید چرا؟

۱۹۶. دو روش متفاوت ذکر کنید که NFS نسخه ۴ طراحی شده تا اجرای بهتر آن را در شبکه گسترده میسر سازد.

۱۹۷. برخی از نام‌های میزبان DNS برای چندین آدرس IP قابل حل هستند. هدف چنین روشی چیست؟

پیوست د
مباحث تکمیلی

تاکنون مباحث بسیار مهمی از سیستم‌های توزیع شده بیان گردید. در این فصل به مباحث تکمیلی از قبیل تحمل خطا، سیستم‌های فایل توزیع شده و امنیت در سیستم‌های توزیع شده می‌پردازیم.

تحمل خطا

هدف مهم در طراحی سیستم‌های توزیع شده، ساخت سیستمی است که به طور خودکار از خرابی جزئی ترمیم شود، بدون این که اثر جدی بر روی کارایی کل سیستم داشته باشد. وقتی خرابی به وجود می‌آید، سیستم توزیع‌شده باید تا انجام ترمیم، در حد قابل قبولی به کارش ادامه دهد، یعنی، باید عیب‌ها را تحمل کند و حتی با وجود عیب‌ها، تا حدی به کارش ادامه دهد. تحمل عیب، با **سیستم‌های مطمئن یا قابل اتکاء**(Dependable System) ارتباط نزدیکی دارد. تعدادی از نیازمندی‌های مربوط به سیستم توزیع شده در جدول زیر آمده است.

تعدادی از نیازمندی‌های مربوط به سیستم توزیع شده.

نیازمندی‌ها	هدف
قابلیت دسترسی (Availability)	یعنی، سیستم فوراً قابل استفاده است. به طور کلی، احتمال عملکرد درستِ سیستم در هر لحظه و آمادگی برای اجرای وظایف از طرف کاربرانش را، قابلیت دسترسی گویند. به عبارت دیگر، سیستمی با قابلیت دسترسی بالا، سیستمی است که در هر لحظه از زمان، با احتمال بالایی کار می‌کند.
قابلیتاطمینان(Reliability)	یعنی، سیستم به طور پیوسته و بدون خرابی اجرا می‌شود. برخلاف **قابلیت دسترسی**، قابلیت اعتماد بر حسب فاصله زمانی (به جای لحظه زمانی) تعریف می‌شود. سیستمی با قابلیت اطمینان بالا، سیستمی است که بدون این که در دوره نسبتاً طولانی از زمان دچار وقفه شود، به کارش ادامه می‌دهد.
ایمنی (Safety)	وضعیتی است که وقتی سیستم به طور موقت درست کار نمی‌کند، هیچ فاجعه‌ای (محیطی، انسانی و. . .)رخ نمی‌دهد.
قابلیت نگه‌داری (Maintainability)	بیان می‌کند که سیستم خراب شده را با چه مقدار تلاش می‌توان **ترمیم**(Recovery)کرد. سیستمی با قابلیت نگه‌داری بالا، ممکن است درجه بالایی از قابلیت دسترسی را عرضه کند، به ویژه، اگر خرابی‌ها را بتوان به طور خودکار تشخیص داد و ترمیم کرد. اغلب، سیستم‌های مطمئن باید درجه بالایی از امنیت(Security) را فراهم کنند. سیستم در صورتی **خراب**(Fail)است که نتواند تعهداتش را انجام دهد. اشکال (Error) بخشی از حالت سیستم است که ممکن است منجر به خرابی(Failure)شود. علت اشکال را **خطا**(Fault) می‌گویند. یافتن علت اشکال مهم است.

سیستم‌های فایل توزیع شده

کاربران از یک سیستم فایل توزیع شده (DFS=Distributed File System) انتظار فراهم کردن راحتی، قابلیت اطمینان و کارایی مطمئن نسبت به سیستم‌های فایل معمولی را دارند. راحتی استفاده از یک سیستم فایل توزیع شده بستگی به دو موضوع کلیدی دارد. ۱. **شفافیت**، یک سیستم فایل توزیع شده کاربران را از محل فایل‌های خود در گره‌ها و دیسک در سیستم آگاه نمی‌کند. ۲. **معنی فایل به اشتراک‌گذاری شده**، قوانین به اشتراک‌گذاری را مشخص می‌کند، که آیا و چگونه اثر تغییرات فایل توسط یک فرآیند از طریق فرآیندهای دیگر با استفاده از فایل به صورت هم‌زمان قابل مشاهده است؟ این مباحث را در جدول زیر می‌بینید:

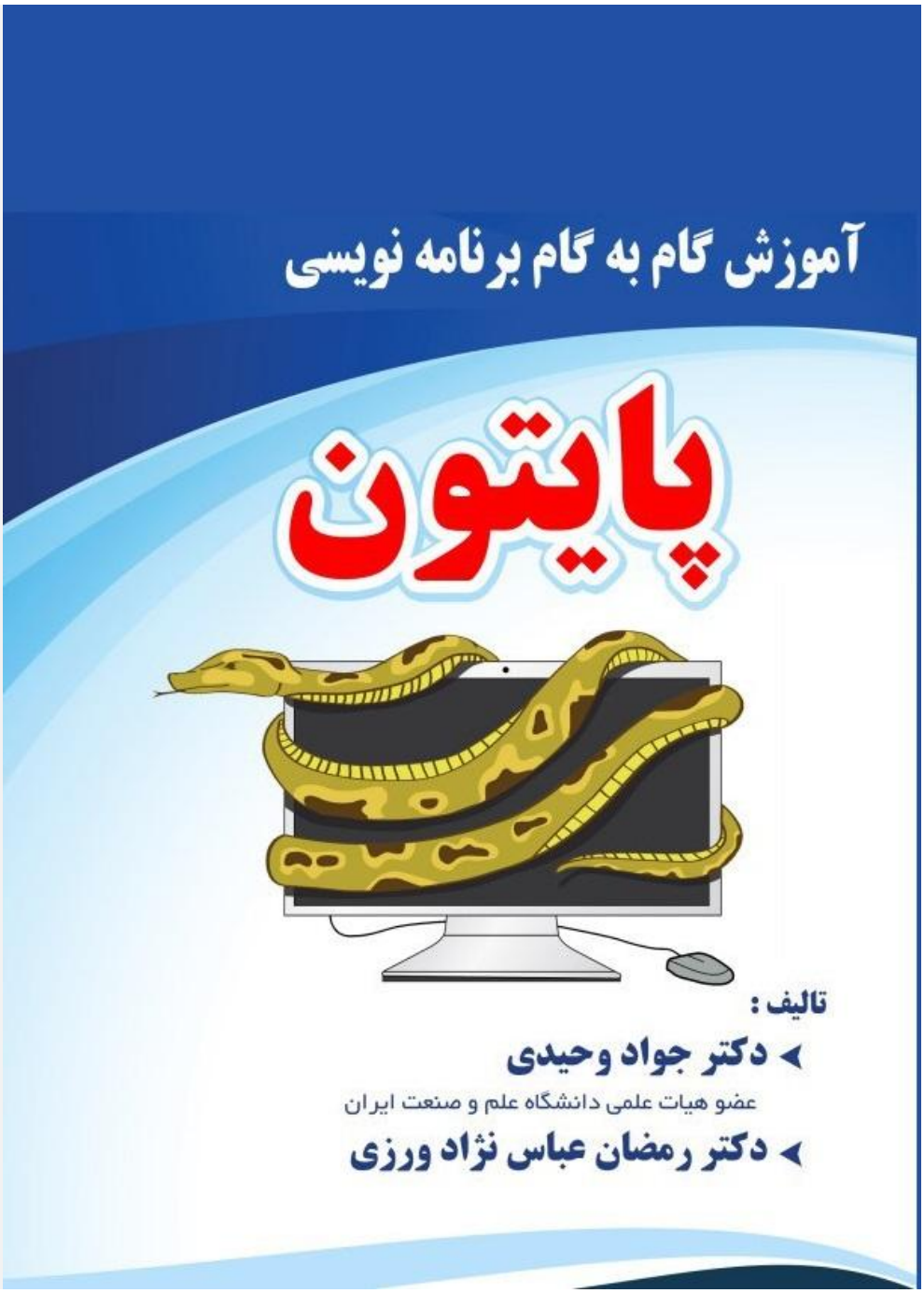
موضوعات مربوط به طراحی در سیستم‌های فایل توزیع شده.	
موضوع	**توصیف**
شفافیت	دو موضوع مرتبط با شفافیت در یک سیستم فایل توزیع شده عبارت‌اند از: چه مقدار اطلاعات در باره محل یک فایل باید در نام مسیر آن منعکس شود؟، و آیا DFS می‌تواند مکان یک فایل را جهت بهینه‌سازی کارایی دسترسی به فایل تغییر دهد؟شفافیت بالای یک سیستم فایل نشان می‌دهد که یک کاربر دانش زیادی در مورد محل فایل‌ها در یک سیستم نیاز ندارد. شفافیت دارای دو جنبه است. **شفافیت محل** دلالت براین دارد که نام فایل نباید مکان خودش را در سیستم فایل آشکار کند. **استقلال محل** دلالت بر این دارد که باید تغییر مکان یک فایل بدون نیاز به تغییر نام آن ممکن باشد.
تحمل خطا	خطا مستمر در سیستم‌های کامپیوتری و یا لینک ارتباطی ممکن است فعالیت‌های پردازش فایل را مختل کند. خطا، در دسترس بودن سیستم فایل را تأثیر می‌گذارد و همچنین باعث آسیب رساندن استحکام داده‌های فایل و ابر داده به عنوان مثال، داده‌های کنترل در سیستم فایل می‌شود. DFS ممکن است از یک تکنیک رونوشت در یک پروتکل سیستم فایل برای حفاظت از پایداری ابر داده استفاده کند یا از طراحی سرویس دهنده فایل بدون حالت استفاده کند، که آن برای محافظت از پایداری ابر داده زمانی که یک خطا رخ می‌دهد، غیرضروری می‌کند. برای محافظت فایل داده، ممکن است تراکنش معنایی را فراهم کند که در اجرای تراکنش‌های اتمیک مفید هستند.
کارایی	تأخیر شبکه یک عامل موثر در زمان دسترسی به فایل در DFS است. تأخیر شبکه بر **کارایی** و **توسعه-پذیری** یک DFS تأثیر می‌گذارد. از این رو، یک DFS باید از تکنیک‌هایی جهت ترافیک شبکه تولید شده توسط دسترسی فایل استفاده کند. عملکرد یک DFS دارای دو جنبه **کارایی** و **توسعه-پذیری** است. در یک سیستم توزیع شده، تأخیر شبکه عامل موثری بر کارایی یک فعالیت پردازش فایل است. DFS از تکنیک حافظه پنهان فایل برای نگه‌داری یک کپی از یک فایل از راه دور در گره یک فرآیند که به فایل دسترسی دارد، استفاده می‌کند. بنابراین، دسترسی به فایل باعث ترافیک شبکه نمی‌شود، هر چند قدیمی شدن داده‌ها در حافظه پنهان فایل از طریق تکنیک‌های ارتباط حافظه پنهان جلوگیری می‌شود. **توسعه‌پذیری** کارایی DFS مستلزم آن است که هنگامی که اندازه سیستم به دلیل افزایش گره‌ها یا کاربران بزرگ می‌شود، زمان پاسخ نباید افزایش یابد. یک سیستم توزیع شده ترکیبی از خوشه‌ها (کلاسترها)که شامل گروه‌های سیستم‌های کامپیوتری با شبکه‌های محلی با سرعت بالا هستند، به طوری که ذخیره یک کپی از یک فایل در یک خوشه تضمین می‌کند که کارایی دسترسی فایل برای دسترسی از یک سیستم کامپیوتری در داخل یک خوشه مستقل از اندازه سیستم خواهد بود. این عمل، همچنین ترافیک شبکه را کاهش می‌دهد. هر دوی این نتایج به افزایش توسعه‌پذیری کارایی DFS کمک می‌کند.

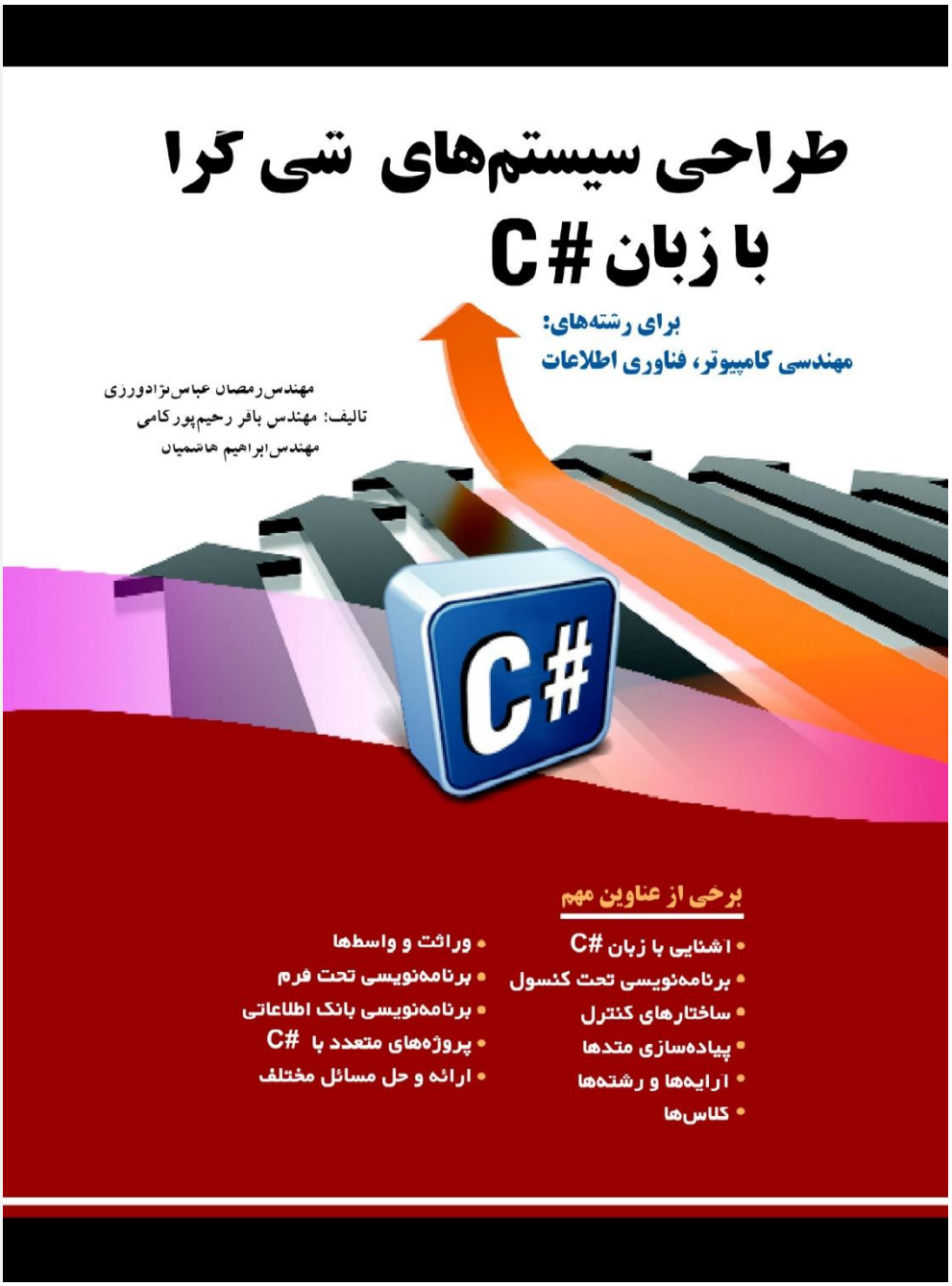
امنیت سیستم‌های توزیع شده

فرآیندها در یک سیستم عامل توزیع شده، از شبکه برای دسترسی به منابع راه دورو برقراری ارتباط با سایر فرآیندها استفاده می‌کنند. شبکه ممکن است شامل کانال‌های ارتباطی عمومی یا سیستم‌های کامپیوتری به نام پردازنده‌های ارتباطی باشد که تحت کنترل سیستم عامل توزیع شده نیستند. بنابراین، مزاحمی که در یک پردازنده ارتباطی قرار گرفته باشد ممکن است پیام‌های میان فرآیندی را تخریب کند تا عملیات فرآیندها را مختل کند یا پیام‌هایی را ایجاد کند که خود را در نقاب یک کاربر نشان دهد و به منابع او دسترسی پیدا کند.

یک سیستم عامل توزیع شده روش‌های امنیت پیام‌ها را به کار می‌گیرد تا مانع از تداخل مزاحم‌ها با پیام‌های میان فرآیندی شود. رمزگذاری ستون فقرات این روش‌ها را تشکیل می‌دهد. هر چند استفاده از رمزگذاری بر این امر دلالت می‌کند که باید از حملات رمزی جلوگیری شود و فرآیندها باید بدانند که در حین برقراری ارتباط با دیگر فرآیندها از کدام کلیدهای رمزگذاری استفاده کنند. به دو طریق با این موضوعات مواجه می‌شویم یا از طریق استفاده از رمزگذاری عمومی کلیدها یا از طریق استفاده از کلیدهای نشست که توسط مراکز توزیع کلید به صورت ایمن میان فرآیندهای ارتباطی توزیع شده‌اند. به منظور جلوگیری از تغییر چهره سیستم عامل توزیع شده ابزارهای تایید موثق شخص ثالث را برای استفاده در زمان ارسال پیام یا در حین استفاده از منابع فراهم می‌کند.

در این بخش مباحث تحمل خطا، فایل‌ها در سیستم توزیع شده و امنیت به طور خلاصه اشاره گردید. مباحث تکمیلی این بخش به صورت فایل PDF در سایت انتشارات فناوری نوین به آدرس www.fanavarienoin.net موجود است، که می‌توانید این فایل را از سایت دانلود کنید.


آموزش گام به گام برنامه نویسی
پایتون
تالیف :
دکتر جواد وحیدی
عضو هیات علمی دانشگاه علم و صنعت ایران
دکتر رمضان عباس نژاد ورزی


طراحی سیستم‌های شی گرا
با زبان C#
برای رشته‌های:
مهندسی کامپیوتر، فناوری اطلاعات
مهندس رمضان عباس‌نژادورزی
تالیف: مهندس بافر رحیم‌پورکامی
مهندس ابراهیم هاشمیان
C#
برخی از عناوین مهم
• آشنایی با زبان C#
• برنامه‌نویسی تحت کنسول
• ساختارهای کنترل
• پیاده‌سازی متدها
• آرایه‌ها و رشته‌ها
• کلاس‌ها
• وراثت و واسط‌ها
• برنامه‌نویسی تحت فرم
• برنامه‌نویسی بانک اطلاعاتی
• پروژه‌های متعدد با C#
• ارائه و حل مسائل مختلف

حل مسائل C++

(آزمایشگاه کامپیوتر مرجع کامل)

دکتر رمضان عباس نژادورزی
مولفین: مهندس عمران یونسی
مهندس یوسف عباس نژادورزی

برخی از عناوین مهم

مقدمه‌ای بر C++
ساختار تصمیم و حلقه تکرار
توابع در C++
آرایه‌ها و رشته‌ها
حل بیش از ۳۲۵ مسئله برنامه نویسی
حل بیش از ۱۴ پروژه برنامه نویسی

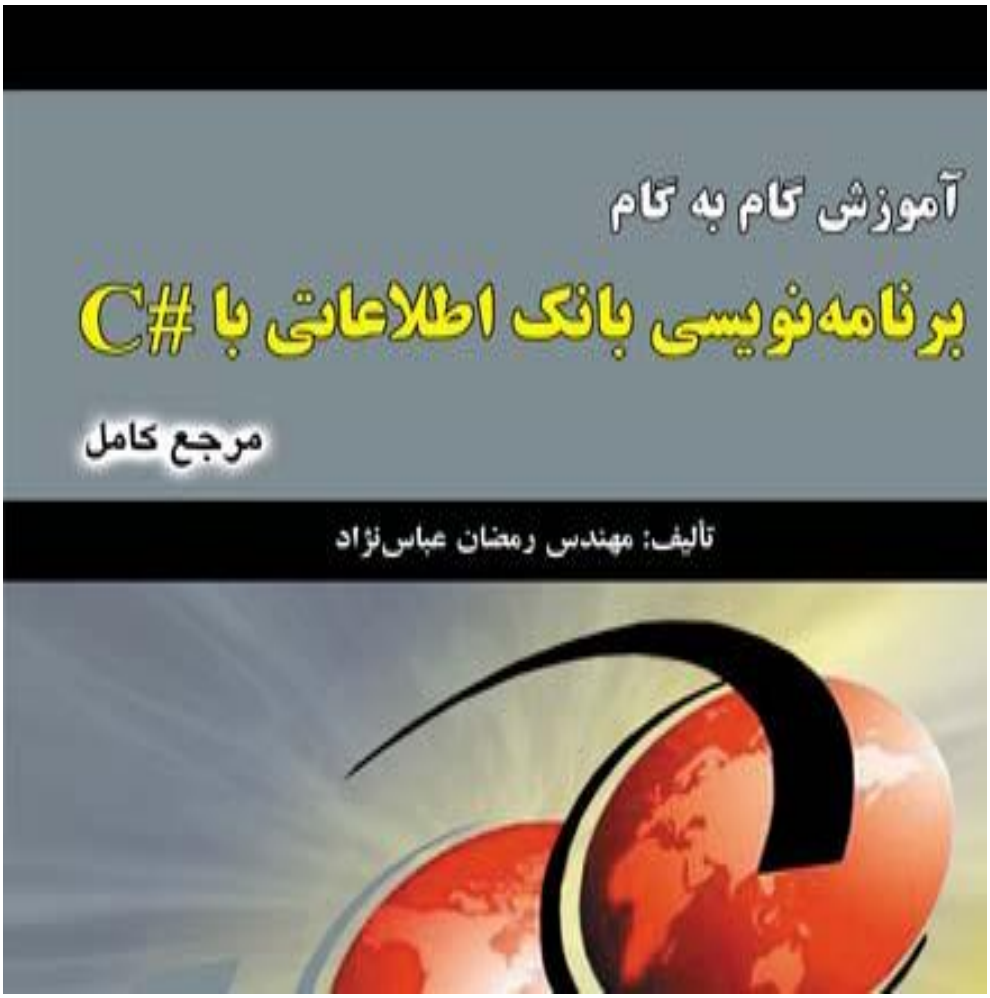
درس وکنکور پایگاه داده پیشرفته

دکتر رمضان عباس نژادورزی
مو لفین : دکتر مرتضی بابا زاده
دکتر میر سعید حسینی

برخی از عناوین مهم

تراکنش ها
تئوری پی در پی پذیری
قفل گذاری ها
زمان بندی بدون قفل
ترمیم پایگاه داده
امنیت در پایگاه داده
حل تشریحی بیش از ۲۵۰ پرسش های چهار گزینه ای
حل تشریحی بیش از ۱۰۰ مسئله پایگاه داده پیشرفته


اصول طراحی پایگاه داده‌ها
برای رشته‌های: مهندسی کامپیوتر
فناوری اطلاعات – علوم کامپیوتر
چاپ دوم
ویرایش دوم
تالیف :
مهندس رمضان عباس نژاد ورزی
مهندس علیرضا عظیمی
مهندس باقررحیم‌پورکامی
برخی از عناوین مهم
بیان مفاهیم پایگاه داده طبق سرفصل وزارت علوم
بیان مسائل متنوع و حل تشریحی آن‌ها
تست‌های چهار گزینه‌ای کارشناسی ارشد سراسری و آزاد
با پاسخ تشریحی از سال ۱۳۷۲ الی ۱۳۹۱
حل تست‌های چند دوره پیام‌نور


آموزش گام به گام
برنامه‌نویسی بانک اطلاعاتی با C#
مرجع کامل
تألیف: مهندس رمضان عباس‌نژاد