

## فصل ۲۸ مهندسی نرم افزار

### خادم / مخدم (C/S)

مفاهیم کلیدی (مرتب بر حروف الفبا)

آزمون ، اجزاء ، توزیع داده ها ، توزیع کارکرد ، طراحی پایگاه داده ها ، طراحی معماری ، گزینه های پیکربندی ،

مدل سازی تحلیلی ، معماری ، میان افزار ، واسط درخواست شی (ORB) ، CORBA

#### KEY CONCEPTS

analysis modeling , architectural design , architecture , components , configuration options , CORBA , database design , data distribution , function distribution , middleware , ORB , testing

#### نگاه اجمالی

مهندسی نرم افزار خادم / مخدم چیست ؟ معماری های خادم / مخدم C/S در سیستم های مبتنی بر کامپیوتر غالب و نافذ است . همه چیز ، از شبکه های خودکار گرفته تا اینترنت ، به آن دلیل وجود دارند که نرم افزار موجود در کامپیوتر - مخدم - از نرم افزار دیگر خادم - درخواست خدمات یا داده می کند . مهندسی نرم افزار خادم / مخدم اصول ، مفاهیم و روش های متداول را (که در بخش های قبلی این کتاب همراه با عناصر شیء گرا و مهندسی نرم افزار مبتنی بر کامپیوتر جهت خلق سیستم های C/S شرح کردیم ) با یکدیگر ترکیب می کنند .

چه کسی این امر را عهده دار می باشد؟ مهندس نرم افزار عمل تجزیه و تحلیل ، طراحی ، اجراء و آزمون های سیستم های C/S را انجام می دهد .

چرا مهندسی نرم افزار خادم / مخدم از اهمیت برخوردار است ؟ تأثیر سیستم های C/S بر کسب و کار ، تجارت ، دولت و علم فراگیر است . با تغییر نحوه ساخت سیستم های C/S به وسیله پیشرفت های مبتنی بر فناوری باید روند مهندسی نرم افزاری را در ساخت آنها اعمال کرد .

مراحل مهندسی نرم افزار خادم / مخدم کدامند ؟ مراحل دخیل در مهندسی سیستم های C/S به مراحل اعمال شده در طول OO و مهندسی نرم افزار مبتنی بر جزء تشکیل دهنده شباهت دارند . مدل فرآیند ، روندی تکاملی است و با درک نیازمندی ها آغاز می شود . کارکردپذیری به سیستم های فرعی اجزائی نسبت داده می شود که به مخدم یا خادم معماری C/S اختصاص داده می شوند . طراحی بر ادغام اجزای

موجود و خلق اجزاء جدید، تمرکز دارد. پیاده‌سازی و آزمون با کارکردپذیری خادم و مخدوم در محیط استانداردهای ادغام اجزاء و معماری C/S دخیل هستند.

محصول مهندسی نرم‌افزار خادم / مخدوم چیست؟ سیستم C/S با کیفیت بالایی محصول مهندسی نرم‌افزار C/S است. دیگر محصولات کاری نرم‌افزار نیز در این میان تولید می‌شوند.

چگونه می‌توانیم انجام درست مهندسی نرم‌افزار خادم / مخدوم را تضمین کنیم؟ با استفاده از عملیات مشابه تضمین کیفیت نرم‌افزار (SQA) اعمال شده در هر فرآیند مهندسی نرم‌افزار بررسی‌های فنی متعارف مدل‌های تحلیل و طراحی را ارزیابی می‌کند و بررسی‌های تخصصی مسائل مربوط به ادغام اجزاء و میان‌افزار (Middle ware) و آزمون برای مشخص کردن خطاها در سطح اجزاء، زیرسیستم مخدوم و سطوح خادم اعمال می‌شود.

در آغاز این قرن عرصه نسل جدید ابزارهای ماشین قادر به حفظ تئرانس‌های دقیق، مهندسین را به طراحی روند جدید کارخانه به نام تولید انبوه وادار کرد. قبل از ابداع این فناوری ابزار ماشین پیشرفته، ماشین‌ها قادر به حفظ تئرانسهای دقیق نبودند اما با استفاده از این فناوری امکان ساخت قطعات قابل تعویض - اساس تولید انبوه - فراهم شد.

وقتی سیستم مبتنی بر کامپیوتر جدید عرضه می‌شود، مهندسی نرم‌افزار با محدودیت‌های فناوری محاسبه موجود محدود می‌شود و هنگامی که فناوری‌های جدید قابلیت‌هایی را ارائه می‌کند که نسل‌های قبلی مهندسین به آنان دسترسی نداشتند، این محدودیت به قوت تبدیل می‌شود. تکامل و تحول معماری‌های کامپیوتر توزیع شده به مهندسین سیستم و نرم‌افزار امکان داده است تا رهیافت‌های جدیدی در مورد نحوه سازمان‌دهی کار و پردازش اطلاعات در سازمان ارائه کنند.

ساختارهای جدید سازمان و رهیافت‌های پردازش اطلاعات جدید پیشرفت زیادی نسبت به فناوری‌های کامپیوتر قادر و فناوری مبتنی بر میکروکامپیوتر ایجاد می‌کند. معماری‌های محاسبه جدید، فناوری را ارائه کرده که به سازمان‌ها امکان می‌دهد تا روند کاری خود را دوباره سازمان‌دهی و مهندسی کنند.

در این فصل معماری غالب در سیستم‌های پردازش اطلاعات - مبتنی بر خادم / مخدوم (C/S) در محیط‌های سیستم‌های تجاری الکترونیکی را بررسی می‌کنیم. سیستم‌های C/S همراه با پیشرفت‌های کامپیوتر رومیزی، مهندسی نرم‌افزار مبتنی بر اجزاء، فناوری‌های ذخیره‌سازی جدید، بهبود ارتباط شبکه و توسعه فناوری پایگاه داده تکامل پیدا کرده است. هدف از این فصل<sup>۱</sup> ارائه بررسی مختصری از

<sup>۱</sup>. بخشهایی از این فصل مطابق موارد درسی تهیه شده توسط جان پورتر در خصوص خادم / مخدوم و ارائه شده در دانشگاه فیرفیلد آورده شده است. (یا کسب اجازه از ایشان)

سیستم‌های C/S با تأکید بر مسائل مهندسی نرم افزار ویژه است که باید به هنگام تحلیل ، طراحی ، آزمون و پشتیبانی این سیستم‌ها مشخص شوند .

## ۲۸-۱ مقدمه

ده سال گذشته در زمینه محاسبه گام‌های بلندی برداشته شده است . اولین گام آن است که نرم افزار قدرتمندتر و ارزان تر شده است . کامپیوترهای رومیزی اکنون همان قوت کامپیوترهای بزرگ چند سال پیش را دارند . زمینه دوم ارتباطات است ، پیشرفت‌هایی مثل سیستم‌های ارتباطی ماهواره‌ای و سیستم‌های تلفن دیجیتالی به آن معنا هستند که اکنون می‌توان به شکلی کارآمدتر و ارزان تر با سیستم‌های کامپیوتری در نقاط بسیار دور از یکدیگر ارتباط برقرار کرد . این امر به مفهوم سیستم محاسبه توزیع شده منتهی شده است . چنین سیستمی از تعدادی کامپیوتر تشکیل می‌شود که به یکدیگر متصل می‌شوند و عملیات مختلفی را انجام می‌دهند . چند دلیل برای مشهور شدن این سیستم‌ها وجود دارد :

عملکرد ، عملکرد بسیاری سیستم‌های توزیع شده مختلف را می‌توان تنها با اضافه کردن کامپیوتر به آنها افزایش داد . این روش معمولاً ساده‌تر و ارزان تر از ارتقاء پردازشگر کامپیوتر بزرگ است . سیستم‌های شاخص که در آنها حصول به این افزایش عملکرد امکان پذیر است سیستم‌هایی هستند که در آنها کامپیوترهای توزیع شده پردازش‌های زیادی را انجام می‌دهند و نسبت کار ارتباطی به پردازش پایین است

اشتراک منبع ، سیستم توزیع شده به کاربران خود امکان می‌دهند تا به مقادیر زیاد داده موجود در کامپیوترها که سیستم را تشکیل می‌دهند ، دسترسی پیدا کنند . به جای آنکه نسخه های تکراری داده‌ها روی کامپیوترهای متعدد قرار گیرند ، داده ها برای کامپیوترها کوچکتر به اشتراک گذاشته می شوند . ضمن آنکه یک سیستم توزیع شده امکان دسترسی به خدمات خاص را نیز فراهم می سازد .

تلورانس شکست ، یک سیستم توزیع شده می تواند تلورانس خطا را تغییر دهد . برای مثال از آنرو که کامپیوترهای متعددی کارهای مختلفی انجام می دهند با خارج شدن از سرویس یا خراب شدن فیزیکی یکی دیگر می‌تواند جایگزین آن شود . (زیرا دیگر کامپیوتر تک منظوره نخواهد بود) . یک پایگاه داده می تواند بر روی کامپیوترهای متعددی توزیع شود بنابراین اگر یک کامپیوتر از کار بیافتد ، کاربر می تواند از پایگاه داده ها توسط دیگر ایستگاه کاری متصل بهره برد .

### نقل قول

مدل محاسباتی حاد / مخدوم (C/S) بازتابی مصادقی مشخص از فرآیند همکاری توزیع شده است. همکاری ای که بیان کننده رابطه ای سخت و نرم افزاری بین سرویس دهندگان و مشتریان و اجزاء می باشد. آنکس برسون



چه گزینه های

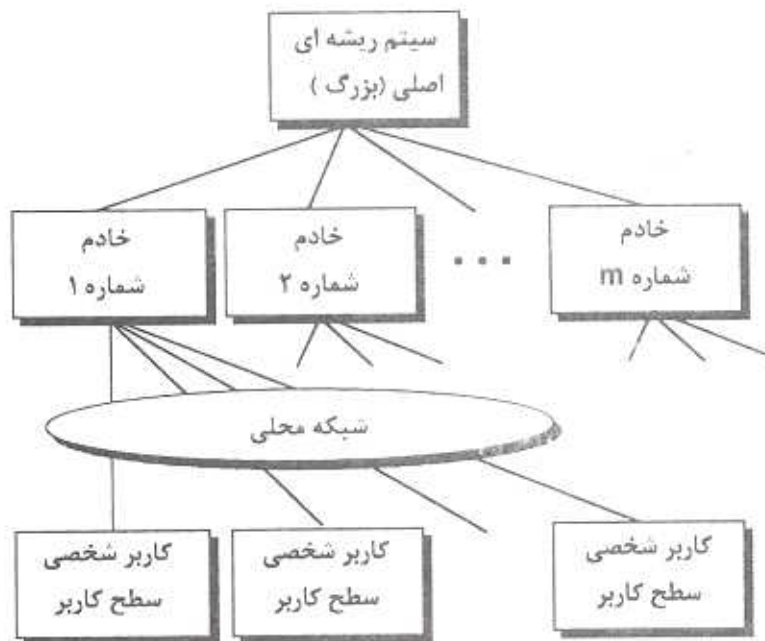
پیکربندی برای اجزاء نرم افزار C/S وجود دارد؟

## ۲۸-۲ سیستم‌های توزیعی

### ۲۸-۲-۱ مخدومها و خادما

هدف این قسمت معرفی ایده مخدوم و خادم است. اینها قطعات اصلی سازنده یک سیستم توزیعی هستند و از این رو شناخت وظایف و قابلیت‌های آنان به هنگام توصیف طراحی و توسعه چنین سیستم‌هایی ضرورت خواهد داشت.

یک خادم کامپیوتری است که خدماتی را انجام می‌دهد که به‌طور طبیعی مستلزم قدرت پردازشی بالاست. یک مخدوم کامپیوتری است که منقضی خدمات ارائه شده توسط یک یا چند خادم می‌باشد و البته بخشی از پردازش را خود انجام می‌دهد. این شکل سازمان‌دهی کامپیوتری کاملاً متفاوت با دو مدلی است که در دهه ۱۹۸۰ و اوایل دهه ۹۰ رایج بود.



شکل ۲۸-۱ معماریهای توزیع شده کامپیوترهای تنظیم شده برای یک شرکت

اولین مدل، به پردازنده میزبان معروف است. در این مدل سازمان‌دهی، تمامی پردازش لازم برای یک سازمان‌دهی، توسط یک کامپیوتر بزرگ - معمولاً یک کامپیوتر مرکزی و اصلی، انجام می‌شد. در حالی که کاربران از پایانه‌های ساده کامپیوتر یا PCهایی با قدرت بسیار کم برای ارتباط با میزبان استفاده می‌کنند. این سازمان‌دهی مشکلاتی را به همراه دارد. دو مشکل بسیار جدی، دشواری در ارتقاء و هماهنگی با رابط‌های مدرن GUI است. زمانی که برنامه‌های کاربردی بزرگتر و بزرگتر می‌شوند، کامپیوتر بزرگ به‌جایی می‌رسد که لازم می‌شود با سخت‌افزار پردازشی جدید، حافظه یا ذخیره فایل ارتقاء یابد. ارتقاء چنین کامپیوترهایی اکنون راحت‌تر از قبل است. هر چند که هنوز هم این روند ممکن است تا



کدام گزینه‌ها برای به هم پیوستن زیرسیستم‌ها در اختیار می‌باشد؟



انداره‌های دشوار و پر هزینه باشد و قطعاً نسبت به افزودن یک خادم جدید مبتنی بر PC بر مجموعه‌ای از کامپیوترهای آرایش یافته به صورت مخدومها و خادما، پر هزینه‌تر و دشوارتر است. مشکل دوم به رابط‌های جدید GUI مربوط می‌شود. ایجاد یک کامپیوتر جهت نمایش صفحه‌ای نسبتاً ابتدایی با در برداشتن چند دکمه و یک نوار بیمابشی، مستلزم ترافیک شدید در خطوط ارتباطی است. به‌طوری‌که سیستم تنها به واسطه داده‌هایی برای راه اندازی و برقراری یک‌سری از رابط‌های مبتنی بر GUI به‌کاربرده می‌شود، به‌راحتی قابل اسداد است.

شکل دوم کامپیوتری آن است که تعدادی کامپیوتر به صورت خادم عمل می‌کنند. اما پردازش اندکی انجام می‌دهند. به‌طور عادی این خادماهای غیرهوشمند به‌عنوان خادماهای پرونده یا جاب برای تعدادی از PCهای قوی یا مینی کامپیوترها عمل می‌کنند که پردازش را انجام داده و به یک شبکه محلی متصلند. کامپیوترهای مخدوم، خدماتی در مقیاس وسیع، مثل دستیابی به یک پرونده و سپس پردازش روی آن فایل را انجام می‌دهند. این مسئله مجدداً به مشکلات ترافیکی منجر می‌شود، به‌عنوان مثال انتقال فایل‌های بزرگ به تعدادی مخدوم مستلزم آن است که هم‌زمان برای ارسال این فایل‌ها زمان پاسخ‌گویی شبکه، بر یک گام کند کاهش یابد. جفت کامپیوتر مخدوم و خادم، تلاشی برای ایجاد تعادل پردازشی در یک شبکه است تا جایی که قدرت پردازشی بین

کامپیوترهایی که خدمات تخصصی نظیر دستیابی به پایگاه داده‌ها (خادما) را انجام می‌دهند و آنهایی که وظایفی چون نمایش GUI را برعهده دارند و کاملاً مناسب نقطه پایانی شبکه هستند، تقسیم می‌شود. به‌عنوان مثال، این شیوه امکان انطباق کامپیوترها با اعمال تخصصی نظیر پردازش پایگاه داده‌ها را به‌وجود می‌آورد که در آن سخت‌افزار و نرم‌افزاری تک‌منظوره برای انجام پردازش سریع پایگاه داده‌ها به‌کار می‌رود. در مقایسه با سخت‌افزار موجود در کامپیوترهای بزرگ که بایستی با دامنه وسیعی از برنامه‌های کاربردی هماهنگ باشند.

## ۲۸-۲-۲ انواع خادما

انواع زیادی از خادما توسعه یافته و ایجاد شده‌اند. لیست تکمیل شده زیر از [ORF99] گرفته شده است.

خادم پرونده. خادم پرونده، فایل‌هایی را برای مخدومها فراهم می‌کند. خادماهای پرونده هنوز در برخی برنامه‌های کاربردی به‌کار می‌روند که در آنها مخدومها به پردازش پیچیده‌ای نیاز دارند که خارج از حوزه معمول پردازش موجود در پایگاه‌های داده‌ای تجاری می‌باشد. به‌عنوان مثال، یک برنامه کاربردی که فرضاً برای یک کار تولیدی بر ذخیره و دستیابی طراحی‌های فنی نیاز دارد، جهت ذخیره و توزیع رسم‌ها به هر یک از مخدومها، از سرویس‌دهنده پرونده استفاده می‌کند.

به طور مثال، این مخدومها توسط مهندسانی که عملیات رسم کشی را انجام می دهند، مورد استفاده قرار می گیرند. عملیات حمایت آنها با به کارگیری یک کامپیوتر مرکزی قوی بسیار بر هزینه خواهد بود. به شرط آن که فایلهای درخواست شده، چندان بزرگ نبوده و بین تعدادی زیاد کاربر مشترک نباشد، یک خادم برونده هنوز هم روش عالی برای ذخیره و پردازش فایلهای به شمار می رود.

خادم پایگاه داده ها. خادماهای پایگاه داده ها، کامپیوترهایی هستند که مجموعه های وسیعی از داده های ساخت یافته را ذخیره می کنند. به عنوان مثال یک بانک، از خادم پایگاه داده ها در ثبت سوابق مشتری با اطلاعاتی از قبیل نام یک حساب، نام دارنده حساب، موجودی جاری حساب، محدودیت اضافه برداشت حساب استفاده می کند. یکی از خصوصیات این قبیل پایگاه های اطلاعاتی که کاربرد خادماهای برونده را بی ثمر می سازند این است که فایلهای ایجاد شده، بسیار بزرگ بوده و اگر به صورت عمده به عهده مخدوم گذاشته شود، ترافیک را کند می کنند. خوشبختانه در اکثریت وسیعی از این انتقال عمده ضرورتی ندارد، به عنوان مثال در کاربرد بانک داری، پرس و جوهای عادی از پایگاه داده ای بانک عبارتند از:

- حساب های مشتریانی که اضافه برداشتی بیش از ۲۰۰۰ دلار دارند را پیدا کنید.
- موجودی تمام حساب های متعلق به آقای جان اسمیت را بیابید.
- تمامی سفارش های منظم صورت گرفته توسط خانم جین اسمیت را پیدا کنید.
- کل معاملات انجام شده دیروز در شعبه مرکزی منچستر را بیابید.

حال یک پایگاه داده ای معمول بانک داری میلیون ها رکورد را دربر خواهد داشت. گرچه پرس و جوهای فوق شامل اطلاعات منتقل شده به یک مخدوم است که فقط بخش بسیار کوچکی از آن را تشکیل می دهد. در یک محیط پایگاه داده ای خادم و مخدوم، مخدومها معمولاً با به کارگیری GUI، سؤالاتی را به پایگاه داده ها می فرستند. این سؤالات به زبان SQL (زبان ساخت یافته پرس و جو) به خادم ارسال می شوند. خادم پایگاه داده ها کد SQL را می خواند، تفسیر می کند و سپس آن را روی یک شی HCI مثل کادر ورود به متن نمایش می دهد. نکته اصلی در این جا این است که خادم پایگاه داده ها تمام پردازش را انجام می دهد و مخدوم مراحل استخراج یک سؤال را از برخی ورودی ها مثل یک فیلد متنی، ارسال پرس و جو و نمایش پاسخ خادم پایگاه داده را روی خروجی ها مثل جعبه حرکتی را به انجام می رساند.

خادم گروه افزار. گروه افزار اصطلاحی برای توصیف نرم افزاری است که کار گروهی از کاربران را

سازماندهی می کند. سیستم گروه افزار معمولاً وظایف زیر را ارائه می دهد:

- عنصری در رابط گرافیکی کاربر.
- مدیریت یادداشت های روزانه افراد یک تیم.
- مدیریت جلسه برای یک گروه، به عنوان مثال تضمین این مطلب که اعضای تیمی که بایستی در

جلسه حضور یابند، در زمان برنامه ریزی شده جلسه، وقت آزاد دارند.

- مدیریت گردش کار ، وقتی اعمال وظایف به اعضا محول می شوند و سیستم گروه افزار اطلاعاتی را درباره اتمام کار فراهم نموده و تذکر و یادآوری را جهت انجام وظایف برای کارکنان ارسال می کند.
- مدیریت پست الکترونیک، به عنوان مثال ترتیب ارسال یک email حتماً به اعضا، یک نیم به هنگام تکمیل یک عمل خاص.

خادم گروه افزار اطلاعات را نگهداری و این وظایف را می کند. مثلاً لیست email ها را ذخیره کرده و برای کاربران سیستم گروه افزار امکان برقراری ارتباط با آن را فراهم می سازد. به عنوان مثال آگاه ساختن آنان از اتمام یک عمل یا اطلاع از تاریخ جلسه ای برخی اعضا، پرسنل در آن حضور دارند.

خادم وب، مدارک و اسناد وب به صورت صفحاتی در کامپیوتری معروف به ( خادم وب ) ذخیره می شوند. وقتی شما برای نمایش صفحه وب از مرورگر استفاده می کنید، به طور طبیعی یک خط ارتباطی را در سند موجود وب کلیک می کنید. این منجر به بیعاری می شود که به خادم وب دارنده صفحه منتقل می گردد. سپس این خادم با ارسال صفحه به کامپیوتر شما یعنی جایی که مرورگر می تواند آن را نمایش دهد ، پاسخ خواهد داد. بنابراین خادم های وب به عنوان نوعی خادم برونده عمل می کنند که فایل های نسبتاً بزرگ را بین کاربران توزیع کرده و آنها نیز یک مرورگر را برای بررسی این صفحه ها به کار می برند. زمانی که مرورگر را به کار می برد، به منظور برقراری ارتباط با مرورگر وب، از پروتوکل معروف به HTTP استفاده می کند.

خادم پستی، خادم پستی، ارسال و دریافت پست برای گروهی از کاربران را کنترل می کند. معمولاً یک PC میان برد برای این منظور به کار می رود. پروتوکل هایی وجود دارند که برای پست الکترونیکی مورد استفاده قرار می گیرند. خادم پستی فقط به یک کاربرد اختصاص خواهد داشت.

خادم شی ای: یکی از مهیج ترین پیشرفت در محاسبات توزیعی طی پنج سال گذشته ، تحولات انجام شده توسط محققین و نیز توسعه دهندگان در ایجاد اشیا توزیعی بوده است. اینها اشیا هستند که روی یک کامپیوتر، معمولاً یک خادم ذخیره می شوند و مخدومها می توانند با ارسال پیام هایی به شیء و مطابق با روش هایی تعیین شده توسط کلاس شیء ، عملکرد شیء را راه اندازی کنند.

این فناوری در نهایت برنامه نویسان را از ضرورت برنامه نویسی برتولکی سطح پایین به هنگام دسترسی به سایر کامپیوترهای یک شبکه ، بی نیاز خواهد ساخت و در نتیجه به برنامه نویسان امکان می دهد تا اشیا را در مکان های دور چنان مورد عمل قرار دهد که گویی در کامپیوتر محلی آنها قرار دارند. خادمی که اشیا را نگهداری کرده و امکان دسترسی به آنها را از فاصله دور فراهم می کند، به نام خادم شی ای معروف است.

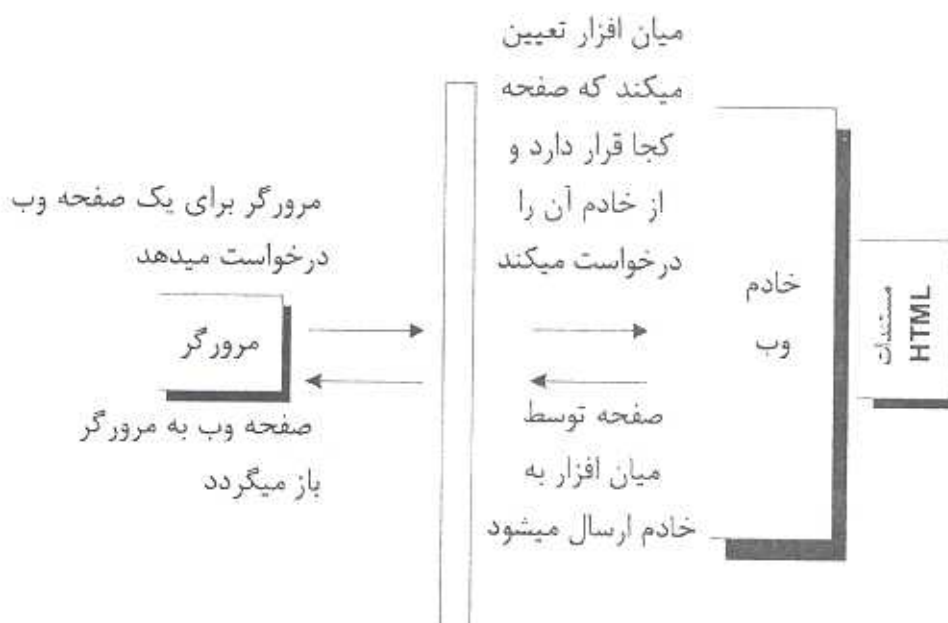


خادم وب

خادم چاپ. خادم چاپ به درخواست‌های یک مخدوم دور سرویس می‌دهد. این قبیل خادم‌ها مبتنی بر PC‌های کاملاً ارزان هستند و وظایف محدود ردیف کردن درخواست چاپ و راهبری چاپگر در روند چاپ را انجام داده و به کامپیوترهای مخدوم اطلاع می‌دهند یک درخواست خاص چاپی به‌انجام رسیده است. خادم کاربردی. خادم کاربردی به یک کاربرد منفرد اختصاص دارد. این خادم‌ها اغلب با به‌کارگیری زبانی چون جاوا یا C++ نوشته می‌شوند. نمونه خاص این‌گونه خادم که در طراحی یک تولیدکننده هواپیمایی مورد استفاده قرار گرفت و بر انواع گوناگون طرح‌های ایجاد شده توسط کارکنان فنی نظارت دارد، مناسب نوعی پردازش تولیدی است.

### ۲۸- ۲- ۳ میان افزار

شاید تا کنون برداشت شما این باشد که ارتباط بین مخدوم و خادم، مستقیم و بی‌واسطه است. اما متأسفانه این تصور نادرست است. چرا که معمولاً دست‌کم یک لایه نرم‌افزاری بین مخدوم و خادم وجود دارد. این لایه، میان‌افزار نام دارد. به‌عنوان نمونه میان‌افزار، شکل ۲۸-۲ را در نظر بگیرید. این شکل ارتباط بین یک مخدوم که از مرورگری مثل internet explore استفاده می‌کند و خادم وب را نشان می‌دهد.



شکل ۲۸-۲ میان افزار و خادم وب

در این جا میان‌افزار که مابین خادم وب و مخدوم هدایت‌کننده مرورگر وب واقع شده است، هر گونه درخواستی از جانب مرورگر را در بین راه متوقف می‌سازد. اگر درخواستی برای صفحه وب



صورت گیرد، میان افزار محل سند وب را تعیین کرده و درخواست صفحه را صادر می کند. خادم به درخواست پاسخ می دهد و صفحه را به میان افزار ارجاع داده و آن نیز صفحه را به مرورگر هدایت می کند و مرورگر هم آن را روی صفحه نمایش مخدوم، نشان می دهد. دو نوع میان افزار وجود دارد که عبارتند از: میان افزار عمومی و میان افزار خدماتی. اولی نرم افزاری مرتبط با خدمات کلی مورد نیاز تمامی مخدومها و خادماست. نرم افزار خاصی که تحت این عنوان نامیده می شود، عبارت است از:

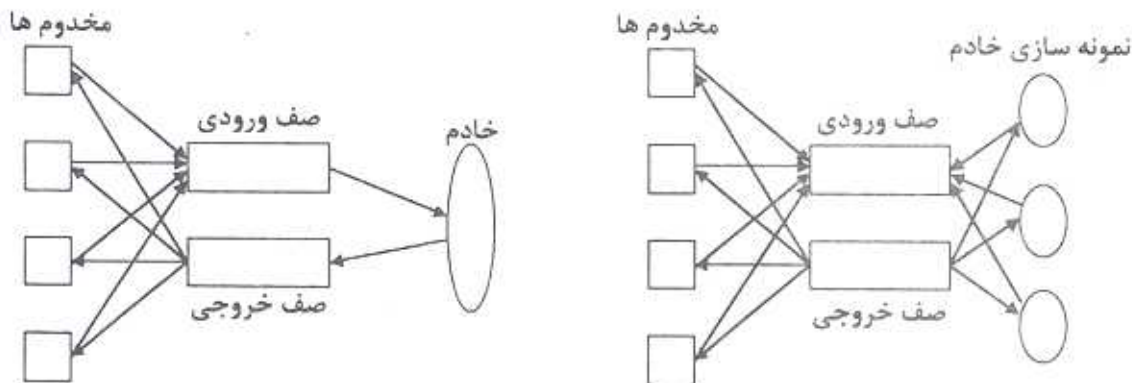
- نرم افزاری برای انجام ارتباطات با به کارگیری TCP / IP و سایر پروتکل های شبکه.
- نرم افزار سیستم عامل که به عنوان مثال ذخیره فایل توزیعی را به عهده دارد. این مجموعه فایل هایی است که در سیستم توزیعی پراکنده شده اند. هدف این بخش از سیستم عامل تضمین این نکته است که کاربران بدون نیاز به دانستن محل فایل ها می توانند به آنها دسترسی پیدا کنند.
- نرم افزار تأیید که نقش آن کنترل این مسئله است که آیا کاربری که قصد استفاده از یک سیستم توزیعی دارد، واقعاً مجاز به این کار می باشد.

- میان افزار پیغام گرا که بر ارسال پیامها از مخدومها به خادما و دریافت آنها، نظارت دارد.

میان افزار خدماتی، نرم افزاری مرتبط با یک سرویس خاص می باشد. نمونه های خاص این نوع نرم افزار شامل موارد است:

- نرم افزاری که باعث می شود پایگاه های داده ای متمایز به یک شبکه مخدوم / خادم متصل شوند. احتمالاً بهترین نمونه این نوع نرم افزار، ODBC (اتصال باز پایگاه داده ها) است که توسط مایکروسافت تولید می شود. این نرم افزار وجود اکثریت عمده سیستم های مدیریت پایگاه داده ها در شبکه را امکان پذیر می سازد.
- نرم افزار ویژه شیء توزیعی مثل نرم افزار مرتبط با CORBA. کوربا فناوری اشیاء توزیعی است که حضور توأم اشیاء نوشته شده به زبان های مختلف در یک شبکه را میسر می سازد به طوری که هر یک از اشیاء می تواند پیام هایی را به شیء دیگر ارسال کند. میان افزار CORBA وظایفی نظیر راه اندازی اشیاء توزیعی، ارتباط بین اشیاء و امنیت شیء را به عهده دارد.

- میان افزار وب مرتبط با پروتکل HTTP
- میان افزار گروه افزار که فایل های توصیف گر گروه های کاری و روابط متقابل آنها را اداره می کند.
- میان افزار مرتبط با تولیدات خاص امنیتی. نمونه خوبی از این نوع، میان افزار مرتبط با چیزی است که تحت عنوان سوکت های ایمن معروفند. اینها سوکت هایی هستند که برای ارسال داده های ایمن در سراسر یک شبکه می توانند مورد استفاده قرار گیرند به طوری که استراق سمع داده ها توسط افراد مزاحم تقریباً غیر ممکن می گردد. این فناوری در قسمت ۲۸-۸ توصیف می شود.



شکل ۲۸-۳ پیکربندی MOM

## ۲۸-۲-۴ نمونه های از میان افزار

میان افزار پیغام گرا، میان افزاری است که بر جریان پیامها از سوی مخدوم و به سوی مخدوم نظارت دارد. این قسمت جهت دست یافتن به نمونه ای از یک نوع خاص میان افزار، به طور مفصل به تشریح این نرم افزار می پردازد. میان افزار پیغام گرا اغلب توسط سه واژه MOM مورد اشاره قرار می گیرد. این میان افزار، صف هایی که را شامل پیام هایی ارسالی و دریافتی به مخدومها و خادمها هستند، به گونه ای مؤثر اداره و مدیریت می کند. شکل ۲۸-۳ (الف) برخی آرایشها و پیکربندیها را نشان می دهد. تصویر ۲۸-۳الف تعدادی مخدوم را نشان می دهد که به دو ردیف دسترسی دارند. یکی صف ورودی که به پیام هایی را به آن می افزاید و دیگری صف خروجی پیامها را از آن دریافت می کند و یک خادم با وظیفه خواندن و نوشتن پیغامها. پیغام ورودی ممکن است پیغامی باشد به خادم فرمان می دهد تا اطلاعاتی را از پایگاه داده ای پیدا کند و پیام خروجی ممکن است از پایگاه داده ای استخراج شده است. شکل ۲۸-۳ (ب) مخدوم های چندگانه و برخی معرفی های بیشتر شیء برنامه خادم را نمایش می دهد که همزمان روی صف های یکسانی کار می کنند.



ارتباطات بهره مند از  
MOM نامیگامند.

برخی نکات قابل ذکر درباره نرم افزار MOM عبارتند از:

- طی مدتی که مخدوم و خادم ارتباط مقابل دارند، حفظ اتصال اختصاصی ضرورتی ندارد.
- در واقع مدل ارتباطی بین مخدومها و خادمها بسیار ساده است و آنها از طریق صفهایی با یکدیگر در ارتباطند. برنامه نویسی برنامه مخدوم یا خادم فقط لازم است پیغامی را به صف اضافه کند.
- در حقیقت نرم افزار MOM نا همگام با نقطه ای است که یک نیمه از زوج مخدوم/ خادم ممکن است با نیمه دیگر ارتباط برقرار نشود؛ به عنوان مثال، مخدوم ممکن است اجراء نکند. و احتمال دارد برای

تعمیر متوقف شود ولی با این حال خادم می تواند پیام هایی را به صف های MOM اضافه نماید. همچنین ممکن است مخدوم با پیام هایی که منتظر پردازش آنها توسط خادم است، قطع ارتباط کند. خادم می تواند این پیام ها را داخل صف قرار دهد، به نحوی که در نوبت بعدی برقراری ارتباط مخدوم، با این پیام ها، قابل خواندن است. این سناریو برای محاسبات سیار، ایده آل و مطلوب می باشد.

### ۲۸ - ۳ ساختارهای لایه ای (ردیفی)

هدف این بخش، کشف ایده معماری لایه ای برای کاربرد مخدوم و خادم است. این قسمت به توصیف معماری معروف دو لایه ای و سه لایه ای محدود می شود. معماری دو لایه ای برنامه کاربردی مخدوم/ خادم شامل لایه ارائه خدمات، لایه منطقی و لایه پایگاه داده ای است. لایه خدمات و لایه منطقی مجموعه ای از چیزهای بصری را به کاربر ارائه می دهد و پردازش لازم روی داده های تولید شده توسط کاربر و داده های برگشتی توسط خادم را به انجام می رساند. مثلاً این لایه شامل کدی است که بر فشردن دکمه ها، ارسال داده ها به خادم و هر گونه محاسبات محلی لازم روی داده های برگردانده شده از سوی خادم، نظارت دارد. لایه پایگاه داده ای به منظور ذخیره اطلاعات لازم برای کاربرد، مورد استفاده قرار می گیرد. این داده ها ممکن است در یک پایگاه داده ای متعارف یا یک فایل ساده ذخیره شده یا آن که حتی در حافظه نگهداری شود. این لایه روی خادم واقع می شود.

به طور طبیعی معماری های دو لایه زمانی به کار می روند که پردازش کم داده ها مورد نیاز است. معماری خادم، مرورگر وب، نمونه خوبی از یک معماری دو لایه ای است. مرورگر مخدوم روی لایه ارائه خدمات و لایه منطقی قرار دارد، در حالیکه داده های خادم وب یا صفحات وب روی لایه پایگاه داده ها واقع شده است. نمونه دیگری از یک برنامه کاربردی که در آن یک معماری دو لایه به طور طبیعی مورد استفاده قرار می گیرد، یک برنامه کاربردی ثبت داده های ساده است که در این کاربرد، وظایف اصلی کاربران وارد کردن داده ها با اشکال مختلف در یک پایگاه داده ای دور دست می باشد. به عنوان مثال برنامه کاربردی ورود داده ها در نظام بانکی که در حساب های جدید کاربر وارد پایگاه داده ای مرکزی حساب ها می شود. طرف مخدوم این برنامه کاربردی، در لایه ارائه خدمات و لایه منطقی واقع شده، در حالی که پایگاه داده ای حساب ها در لایه پایگاه داده ای قرار خواهد داشت.

هر دو برنامه کاربردی فوق الذکر از یک ویژگی اصلی برخوردارند. که وجه تمایز برنامه های کاربردی دولایه با برنامه های کاربردی چندین لایه به شمار می رود. و آن این واقعیت است که میزان پردازش لازم بسیار کم است. به عنوان مثال در برنامه کاربردی معتبر سازی داده ها تنها پردازش لازم از سوی مخدوم، معتبر سازی داده ها است که بدون رجوع به پایگاه داده ای حساب ها قابل انجام است. مثال مربوطه کنترل این است که نام مشتری رقمی را دربر ندارد. بقیه معتبر سازی در لایه پایگاه داده ها انجام می گیرد و شامل کنترل پایگاه داده ای در تضمین این نکته است که سوابق تکراری مشتری به پایگاه داده افزوده نشده است.

ریس [REE 97] معیارهایی را ارائه داده که بایستی به هنگام بررسی انتخاب معماری دو لایه مخدوم/خادم مورد استفاده قرار گیرند و عبارتند از:

- آیا برنامه کاربردی از یک پایگاه داده‌ای منفرد استفاده می‌کند؟
- آیا پردازشگر پایگاه داده‌ها روی یک پردازنده مرکزی واحد قرار دارد؟
- آیا پایگاه داده‌ها نسبتاً ثابت است و شما رشد کمی را در سایر و ساختار آن پیش‌بینی می‌کنید؟
- آیا پایگاه کاربر ثابت است؟
- آیا نیازمندی‌ها ثابت است یا امکان بسیار کمی برای تغییر آنها در طول حیات سیستم وجود دارد؟
- آیا تعمیر و نگهداری سیستم حداقل خواهد بود؟

گرچه برخی از این پرسش‌ها به یکدیگر مرتبطند (تغییرات نیازمندی‌ها به افزایش وظائف نگهداری منجر می‌گردند) ولی با این وجود آنها مجموعه سؤالات خوبی هستند که بایستی قبل از انتخاب معماری دو لایه مدنظر قرار گیرند.

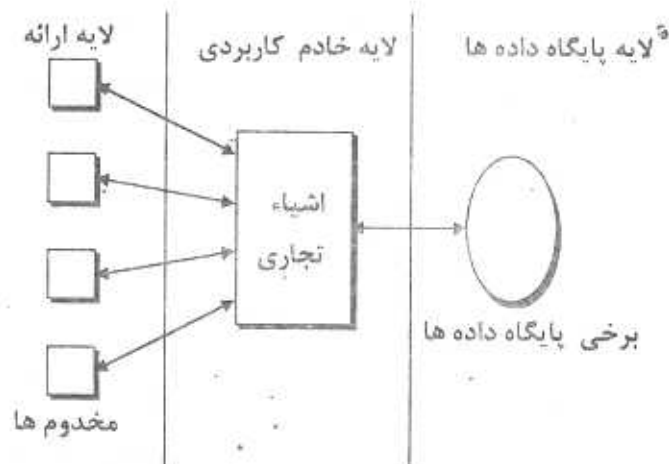
زمانی که یک برنامه کاربردی مستلزم پردازشی قابل توجه می‌باشد، آن‌گاه مشکلات مربوط به رهیافت دولایه شروع به ظاهر شدن می‌کنند. به ویژه برنامه‌های کاربردی که در طول عمر خود تغییرات کاربردی را تجربه می‌کنند. همچنین یک معماری دولایه که در آن خلاصه کد پردازشی وابسته به اعمالی چون فشار دکمه‌ها یا مننی که وارد یک فیلد متنی می‌شود معمولاً کم است، بسیار متمایل به رویدادهای خاص می‌باشد تا داده‌های زیربنایی برنامه کاربردی، و از این‌رو استفاده مجدد دشوار می‌گردد.

شکل ۲۸-۴ یک معماری سه لایه را نشان می‌دهد. این ساختار شامل یک لایه ارائه خدمات، یک لایه پردازشی (یا لایه خادم برنامه کاربردی) و لایه پایگاه داده‌ها می‌باشد. لایه ارائه خدمات مسئول نمایش دیداری برنامه کاربردی است، لایه پایگاه داده‌ها، داده‌هایی را برای برنامه کاربردی، دربر دارد و لایه پردازشی مسئول پردازش است که در برنامه کاربردی صورت می‌گیرد. به عنوان مثال، در برنامه کاربردی بانکداری، کد یا برنامه در لایه ارائه خدمات تنها نظارت رویدادها و ارسال داده‌های ورودی به لایه پردازشی را انجام می‌دهد. این لایه میانی دربردارنده اشیائی است که با عناصر برنامه کاربردی مطابقت دارند. مثلاً در کاربرد بانکداری، اشیاء نمونه عبارتند از بانک‌ها، مشتری، حساب‌ها و تغییرات.



کفتی یک برنامه کاربردی مشتمل بر پردازشهای قابل ملاحظه‌ای باشد، پیاده سازی رهیافت دولایه‌ای سخت‌تر و سخت‌تر خواهد شد.





شکل ۲۸-۴ معماری سه لایه ای

لایه پایانی، لایه پایگاه داده‌ها می‌باشد. این لایه فایل‌های نگهدارنده داده‌های برنامه را دربردارند. لایه میانی قابلیت نگهداری و کاربرد مجدد را به همراه دارد. این لایه شامل موجودیت‌ها و اشیاء است و توسط کلاسهای قابل کاربرد مجدد که می‌توانند بارها در سایر برنامه‌های کاربردی مورد استفاده قرار گیرند، تعیین و مشخص می‌گردند. این اشیاء اغلب تحت عنوان اشیاء تجاری مورد اشاره قرار می‌گیرند. آنها شامل گستره عادی سازنده‌ها، روش‌هایی برای قرار دادن و دریافت متغیرهای شی‌ای، روش‌هایی که محاسبات را انجام می‌دهند و شیوه‌های معمولاً خصوصی هستند با لایه پایگاه داده‌ای ارتباط برقرار می‌کنند. لایه ارائه خدمات پیام‌هایی را به اشیاء این لایه میانی ارسال خواهد کرد که آن نیز یا مستقیماً پاسخ می‌دهد یا گفتگویی را با لایه پایگاه داده‌ای انجام می‌دهد و این لایه سپس داده‌هایی را به لایه ارائه خدمات برگشت خواهد داد.

محل قرارگیری لایه میانی، یک تصمیم طراحی است. این لایه می‌تواند روی خادم نگهدارنده لایه پایگاه داده‌ای قرار گرفته و از سوی دیگر می‌تواند روی یک خادم جداگانه واقع شود. تصمیم‌گیری در مورد محل قرارگیری این لایه بستگی به تصمیمات طراحی دارد و تحت تأثیر عواملی چون چگونگی بارگذاری یک خادم خاص و نزدیکی آن با مخدوم‌ها می‌باشد. محل این لایه از مزایای عمده فراهم شده توسط یک معماری لایه ای سه گانه نمی‌گاهد:

- اول آن‌که، فناوری را که باعث مجزا شدن لایه پایگاه داده‌ها می‌شود، امکان پذیر می‌سازد به طوری که تغییر آن فناوری را می‌توان می‌کرد به عنوان مثال، یک برنامه کاربردی، ممکن است ابتدا فناوری رابطه‌ای را برای لایه پایگاه داده‌ها به کار برد. اما اگر یک پایگاه داده‌ای شی‌گرا فراهم شود که هم‌چون فناوری رابطه‌ای، مؤثر باشد، آن‌گاه به راحتی قابل ادغام است و آنچه نیاز به تغییر دارد روش‌های برقراری ارتباط با پایگاه داده‌ها می‌باشد.

• کد یا برنامه زیادی را از مخدوم دور می‌کند. مخدومهایی که در بردارنده کد زیاد هستند تحت عنوان مخدومهای غنی شناخته می‌شوند. در محیطی که غالباً تغییر ضرورت دارد، این مخدومهای غنی می‌توانند به کابوس نگهداری تبدیل شوند. هر بار که تغییر لازم می‌شود، سازمان بایستی تصمیم کند که برنامه صحیح به هر یک از مخدومها، منتقل شده است. یا یک لایه میانی، بخش اعظمی از کد برنامه کاربردی مخدوم/ خادم در یک محل مستقر می‌شود (یا مکان‌های بسیار کم در صورتی که خادم‌های پشتیبان به کار روند) و تغییرات نگهداری به‌طور مرکزی رخ می‌دهد.

• ایده سه لایه ای با روشهای جدید شیء‌گرا تناسب دارد. همه پردازش به‌جای مجموعه کدهای مرتبط با هر چیز، در لایه ارائه خدمات، به واسطه پیام‌های ارسالی به موجودیت‌ها یا اشیاء انجام می‌گیرد. نکته‌ای که باید در این‌جا مورد تأکید قرار گیرد این است که بین لایه میانی خادم برنامه و میان‌افزار نمای وجود دارد. در حالی که اولی نرم‌افزار واسطه برنامه بین مخدوم و خادم را توصیف می‌کند، دومی به نرم‌افزار سیستم اختصاص می‌یابد.



بین میان‌افزار و لایه کاربردی خادم تفاوت مشخصی وجود دارد.

## ۴-۲۸ پروتکل‌ها

تا کنون اصطلاح پروتکل بدون شرح و توضیح مفصل در این بخش به کار رفته است. هدف این قسمت ارائه توضیح و جزئیات درباره این اصطلاح می‌باشد.

### ۴-۲۸-۱ مفهوم

جهت شرح و توضیح دقیق یک پروتکل از یک مثال ساده استفاده خواهیم کرد، یعنی نمونه‌ای از یک مخدوم که تسهیلات بانکداری خانگی را فراهم می‌کند، مثلاً در موردی که جزئیات حساب در یک خادم نگهداری می‌شوند، به مشتری امکان می‌دهد تا به پرس‌وجوی حساب بپردازد. فرض می‌کنیم که مخدوم یا استفاده از یک‌سری پیام که اعمال مورد نیاز مخدوم را متمایز می‌سازند، با خادم ارتباط برقرار کرده و همچنین داده‌های مورد نیاز خادم را در اختیار آن قرار می‌دهد. به‌عنوان مثال، ممکن است مشتری با تایپ شماره حساب در کادر ورود منی و سپس کلیک روی دکمه ارسال پیام به خادم، عمل پرس‌وجوی موجودی حساب را انجام دهد. این پیام می‌تواند به شکل زیر باشد:

$BQ^* < \text{Account number} >$

که در آن BQ (پرس‌وجوی تراز) این واقعیت را مشخص می‌کند که کاربر حساب را برای باقی‌مانده موجودی آن پرس‌وجو نموده و شماره حساب تعیین‌کننده حساب می‌باشد. سپس خادم این بی‌فا دریافت کرده و مقدار موجودی را دریافت کرده و مقدار موجودی را طی پیامی که ممکن است با قالب باشد، اطلاع می‌دهد:

**B\* < Balance amount >**

مخدوم این پیغام را تفسیر کرده و موجودی را روی برخی خروجی ها نمایش می دهد.  
اگر مشتری قصد داشت خادم را درباره تمامی شماره حسابهای مربوط به خود، مورد پرس و جو قرار دهد، آن گاه پیام به صورت زیر بود:

AQ\*

که در آن AQ (پرس و جوی حسابها) درخواستی برای تمامی حسابها بود و برای این عمل اطلاعات بیشتر ضرورتی نداشت. خادم ممکن بود با پیغامی که با AI (اطلاعات حساب) آغاز می شد پاسخ داده و سپس به صورت زیر، شماره حسابهایی که به ستاره ختم می شدند را در جواب می گنجاند:

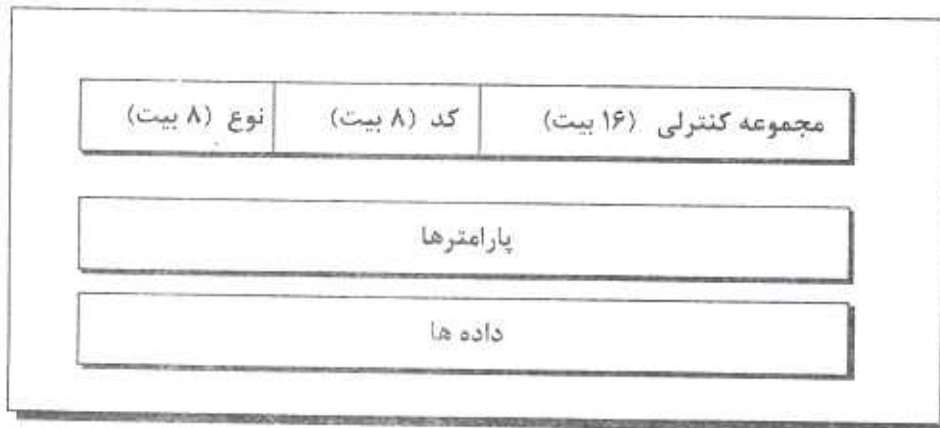
AI\* 2238997 \* 88765 \* 882234 \*

پیامهای توصیف شده، پروتوکل ساده ای را به وجود می آورند که عمکرد و دادهها بین دو عضو یک شبکه (خدمات و خادم) رد و بدل می کنند.

نکته مهم و قابل ذکر درباره پروتوکلها این است که هرگونه مکانیزمی که برای ارتباط در یک شبکه مورد استفاده قرار می گیرد، یک پروتوکل ساده در انتهای آن واقع می شود. به عنوان مثال، اشیا توزیعی، اشیایی هستند که در مکانهای دور در یک شبکه وجود دارند و می توان پیامهایی را بدانها ارسال کرد، گرچه برنامه ریزی این اشیا که در سطح بالایی زیر برنامه نویسی بوده و از برنامه نویس پنهان می گردد، کاربردی از یک پروتوکل می باشد.

پروتوکلی که توصیف شد، در حد ساخنگی و نیز یک پروتوکل کاربردی است. حال لازم است پروتوکلهای واقعی تری که مرتبط با خدمات سطح سیستم می باشند، مد نظر قرار گیرند.





شکل ۵-۲۸ قالب پروتکل ICMP

### ۲۸-۴-۲ IP و ICMP

IP یعنی پروتکلی که اینترنت را اجراء می کند، پروتکل بسیار پیچیده ای است که شرح و توصیف کامل آن از حوزه بحث این کتاب خارج است، و در واقع توضیح جامع این پروتکل و ادای حق مطلب درباره آن، به تألیف کتابی دیگر نیاز دارد. به همین دلیل تنها به بخشی کوچک اما مهم در این پروتکل به نام ICMP (پروتکل پیغام کنترل در اینترنت) خواهیم پرداخت. این بخش برای نظارت و کنترل خطاها و مشکلات در شبکه ای که از IP استفاده می کند، به کار می رود. قالب پیام هایی که بخشی از پروتکل را تشکیل می دهند، در شکل ۵-۲۸ به نمایش درآمده است.

فیلد نوع، نوع خطاهای صورت گرفته را مشخص می کند، مثلاً این فیلد بیانگر آن است که مقصد یک پیام غیرقابل دسترس بوده است. فیلد برنامه، اطلاعات کمکی را در بردارد که جهت ارائه تفسیر مشروح و مفصل فیلد نوع به کار می رود. فیلد جمع کنترلی، توسط نرم افزاری به کار می رود که کنترل انتقال صحیح پیامها را انجام می دهد. دو فیلد باقی مانده داده هایی را در بردارند که به نرم افزار کنترل کننده امکان می دهد تا مشکل ایجاد شده را مشخص کند.

ICMP توسط IP برای انجام بردارش خطا و نیز افزایش کارایی شبکه به کار می رود، به عنوان مثال اثر توسط هر یک از کامپیوترهای واقع در مسیر پیام، مسیر بهتری به کامپیوتر مقصد کشف شده باشد، برای انجام تغییر مسیر به کار می رود.

### ۲۸-۴-۳ POP3

POP3 (پروتکل اداره پست، نسخه ۳) جهت ارسال و دریافت پست الکترونیکی به کار می رود. این یک پروتکل ساده است که کاربرد وسیعی دارد. در این قسمت برخی ویژگی هایی که آن را مورد بررسی قرار خواهیم داد.



POP3 یک پروتکل

بسیار ساده و در عین

حال بسیار قدرتمند

است که بر تمام

پروتکل های موجود

دیگر ارجحیت دارد. از

این رو کاربرد آن

نوصیه می شود.



تعدادی پروتکل های پستی وجود دارد که عبارتند از SMTP و IMAP و نسخه های مختلف POP احتمالاً پیچیده ترین آنها IMAP است که دارای امکانات امنیتی و مجموعه غنی تری از پیام ها نسبت به POP می باشد.

هدف POP این است که به کاربران امکان می دهد تا از دور به پست دستیابی داشته باشند و شامل تعدادی دستور است که ارتباط متقابل برنامه های کاربر با خادم پستی POP را فراهم می کند. استاندارد POP3، تعدادی از پیام های احتمالی قابل ارسال به خادم پستی POP3 و قالب پیام های برگشت داده شده از خادم را توصیف می کند. جدول ۲۸-۱ زیر مجموعه کوچکی از پروتکل POP3 را نشان می دهد.

پیام	معنا
USER	مطلع نمودن خادم (سرویس دهنده) از آنکه یک کاربر مشخص متقاضی بازیابی Mail می باشد
PASS	تهیه رمز عبور برای کاربر بخصوص
STAT	پرسش از خادم در مورد تعداد پیامهای خوانده نشده
DELE	حذف یک پیام
RETR	بازیابی یک پیام

جدول ۲۸-۱ یک زیر مجموعه POP3

### ۲۸-۲-۲ پروتوکل HTTP

این یکی از مهم ترین پروتوکل های به کار رفته در اینترنت است، یعنی پروتوکلی که بر ارتباط بین مخدوم اجراء کننده مرورگر وب مثل Internet Explore و خادم وب نظارت می کند. وظیفه اصلی خادم وب توزیع صفحات وب، به مخدومها است. این مخدومها مرورگری را به کار خواهند برد که به درگاه ۸۰ روی خادم وب متصل می شود، و این درگاه استاندارد است که برای اینگونه خادمها به کار می رود. مرورگری که توسط مخدوم به کار می رود، پیام های تعیین شده توسط پروتوکل HTTP را صادر کرده و این پیامها توسط خادمی تفسیر خواهد شد که عملیاتی نظیر بازگرداندن صفحه وب یا پردازش شکل تعبیه شده در یک صفحه را انجام می دهد. نمونه ای از آن در زیر به نمایش درمی آید:

GET / index.html HTTP / 1.1

این پیام صادر شده توسط مرورگر در زمانی است که قصد دارد صفحه خاصی را نمایش دهد. این پیام ممکن است به طرق مختلف ایجاد شده باشد، مثلاً امکان دارد کاربر پیوند خاصی را در سند وب که به صفحه مورد نیاز اشاره می کند، کلیک کرده باشد. سطر فوق شامل واژه کلیدی GET است که مشخص

می کند فایل قرار است باز گردانده شود، و نیز نام فایل که به دنبال کاراکتر / می آید (html.index) و نسخه پروتکل HTTP که مرورگر صادرکننده پیام آن را به کار می برد.

خادم نیز برای بازگرداندن پیامها به مخدوم، پروتکل HTTP را به کار خواهد برد. به عنوان مثال پیام:

### HTTP / 1.1 200 OK

بدان معناست که خادمی که از نسخه ۱/۱ HTTP استفاده می کند، درخواست مطرح شده توسط مخدوم را با موفقیت پردازش کرده است. کد ۲۰۰ در این مورد، نمونه ای از کد وضعیت بوده و بیانگر موفقیت می باشد.

## ۲۸-۵ سیستم تجارت الکترونیک

### ۲۸-۵-۱ E-commerce چیست ؟

به منظور توضیح و تشریح چگونگی به نظر رسیدن یک سیستم توزیع شده، بهتر است نمونه اقتباس شده از یک حوزه کاربردی معروف به نام Ecommerce مورد بررسی قرار دهیم. به طور کلی اصطلاح تجارت الکترونیک (Ecommerce) کاربرد فناوری سیستم توزیعی در جهت حمایت و پشتیبانی نوعی معامله تجاری است. جزئیات برخی نمونه های این گونه سیستمها در زیر می آید:

- سیستم هایی برای فروش کالا یا خدمات از طریق اینترنت. وقتی که مشتریان با استفاده از مرورگرها با سیستم ارتباط برقرار می کنند، سیستم های نمونه تجارت الکترونیک در این مقوله عبارتند از سیستم های فروش کتب، لباس و CD ها

- سیستم هایی برای شبیه سازی نوعی فعالیت تجاری همزمان با به کارگیری فناوری شبکه. نمونه مناسبی از این نوع سیستم، حراج روی خط (آن لاین) است. یک سیستم نمونه حراج روی خط کالاها را از کاربران اینترنت درخواست کند، جزئیات آنها را روی صفحه وب قرار داده و سپس اعلام می کند که حراج آن کالا آغاز شده است. معمولاً شرکت مناقصه ای، زمانی که مناقصه کامل می گردد را تعیین می نماید.

- سیستم هایی که خدمات شبکه ای را به کاربران ارائه می دهند. احتمالاً معروف ترین های این نوع سیستم آنها می هستند که آگنت های رایگان (حساب های) پست الکترونیک را عرضه می کنند، معمولاً درآمد حاصل از چنین تجارتی ناشی از تبلیغات انجام شده روی صفحات وب به کار رفته برای سایت است. یکی دیگر از خدمات معروف، خدماتی است که توسط مراقبان وب سایت قرار می گیرد. اینها شرکت هایی هستند که با پرداخت هزینه ای بر وب سایت شما نظارت داشتند و اگر به شکلی مثل عملکرد نادرست خادم برای سایت برخورد کند، معمولاً از طریق email یا pager فراخوان برایتان پیام می فرستند.



- سیستم‌هایی که خدمات مشاوره‌ای ارائه می‌دهند، نمونه این نوع خدمات، سیستم‌هایی است که توصیف برخی کالاها مانند CD را پردازش کرده و پس از بررسی برخی سیستم‌های فروش on-line آن را با مناسب‌ترین قیمت گزارش می‌کنند.
- سیستم‌های داخلی که توسط مشتری رؤیت نمی‌شوند اما فعالیت‌های تجاری عادی‌تری را حمایت می‌کنند مثل سیستمی که از عرضه کالاها به یک خرده‌فروش حیابانی پشتیبانی می‌کند.
- سیستم‌های تبلیغاتی عمده در آمد Ecommerce ناشی از تبلیغات on-line است. بسیار از صفحات وب مرتبط با برنامه‌های کاربردی تجارت الکترونیک تبلیغات کمی را تحت‌عنوان تبلیغات banner در برخواهند داشت. این تبلیغات قابل کلیک کردن هستند و کاربر مرورگر را به سایتی هدایت می‌کند که در آن کالا یا خدمات در معرض فروش قرار دارد. سیستم‌های تبلیغاتی شکل خاصی از سیستم‌های Ecommerce هستند که اعمالی نظیر ارسال آگهی‌های banner به یک سایت، نظارت در موفقیت این آگهی‌ها و پرداخت هزینه‌های تبلیغاتی را انجام می‌دهند.

بنابراین اینها مجموعه کاربردهایی هستند که تحت‌عنوان تجارت الکترونیک مطرح می‌باشد. در ادامه این قسمت، یکی از برنامه‌های کاربردی تجارت الکترونیک که بر فروش یک کالا نظارت دارد مورد بررسی قرار خواهد گرفت و به‌طور عادی این همان چیزی است که عوام آن را تحت‌عنوان برنامه کاربردی تجارت الکترونیک می‌شناسند. گرچه امیدواریم پس از خواندن این مقدمه، شما آن را یک نوعی از سیستم تلقی کنید.



تجارت الکترونیکی برای  
خرده‌فروشی Online  
کاربرد ندارد.

## ۲۸-۵-۲ سیستم نمونه E-COMMERCE

جهت درک ماهیت سیستم‌های مخدوم/ خادم، بررسی نمونه‌ای از آنها در محیط تجارت الکترونیک لازم به نظر می‌رسد. این سیستم بر فروش یک کتابفروشی بزرگ نظارت دارد که عملکردی مشابه کتابفروشی‌های عمده نظیر Blackwells یا شعبه Amazon در بریتانیا را داراست. وظایف خاص این‌گونه سیستم‌ها به‌شرح زیر است:

### • تأمین امکانات کاتالوگ.

هر کتابی که توسط کمپانی فروخته می‌شود، وارد کاتالوگ شده و شرح کتاب در صفحه وب نگهداری خواهد شد. اطلاعات خاص مرتبط با کتاب عبارت خواهند بود از: نام کتاب، نویسنده، ناشر و قیمت آن.

### • تأمین تسهیلات جستجو.

کاربر این سیستم برای تصمیم‌گیری در خرید کتاب ملزم به مرور کاتالوگ خواهد بود. این جستجو به طرق مختلف قابل انجام است یعنی می‌توان فهرست را به‌طور ترتیبی و از اولین کتاب موجود در آن مرور

کرد یا آنکه از نوعی سؤال جستجو مثل عنوان کتاب یا ISBN (شاخص بین‌المللی کتاب) آن استفاده نمود.

#### • ایجاد تسهیلات سفارش.

زمانی که مشتری سایت تجارت الکترونیک قصد خریداری کتابی را دارد، سیستم معمولاً از طریق کارت‌های اعتباری امکان انجام این کار را فراهم خواهد کرد، به طور عادی مشتری کتاب را سفارش داده و مشخصات کارت اعتباری او پرسیده می‌شود، برخی سیستم‌های سفارشی اغلب جزئیات کارت اعتباری را به طور دائم ثبت خواهند کرد، به نحوی که در سفارش‌های بعدی تکرار آنها از سوی مشتری ضروری ندارد.

#### • ایجاد امکانات پیگیری.

این تسهیلات به مشتری امکان می‌دهد تا با استفاده از مرورگر خود، روند خرید را پیگیری می‌کند. صفحات وب اختصاصی مشتری یا پیشرفت سفارش را توصیف می‌کنند، اینکه آیا سفارش ارسال شده یا به دلیل اتمام موجودی کتاب، سفارش به تعویق افتاده است، و تاریخ احتمالی ارسال سفارش.

#### پردازش گزارش‌ها.

یکی از تسهیلاتی است که توسط سایت‌های پیشرفته‌تر فروش کتاب عرضه می‌شود و این امکان را به مشتریان می‌دهد تا گزارش کتاب‌های خریداری شده را بنویسند. این گزارش‌ها از طریق پست الکترونیک یا یک صفحه وب اختصاصی در اختیار فروشنده قرار می‌گیرد.

#### ارائه تسهیلات کنفرانس کامپیوتری.

به تعدادی از مشتریان امکان می‌دهد تا با ارسال پیام‌هایی به یک کنفرانس، با یکدیگر در ارتباط باشند. و کنفرانس به یک موضوع خاص مثل آخرین رمان (رابرت گوئدر) یا وضعیت داستان حبلی اختصاص دارد. این امکانات مستقیماً برای یک کنفرانس on-line ایجاد درآمد نمی‌کنند، گرچه اطلاعات مفیدی را درباره تمایل به خرید کتاب در اختیار کادر خرید کتاب فروش قرار می‌دهند.

#### ارائه اخبار یا بولتن به مشتریان.

یکی از امکانات رایجی که در بسیاری از سایت‌های تجارت الکترونیک مختص به فروش کالا وجود دارد. ارائه اطلاعات از طریق پست الکترونیک است. در مورد سایت فروش کتاب که در این مبحث مطرح می‌باشد، نمونه پیام‌های ارسالی به مشتریان عبارتند از ستون گزارشی، پیشنهادات یا پیام‌های خاص مربوط به یک سفارش بخصوص مثل این واقعیت که سفارش ارسال شده است.

#### کنترل و نظارت بر موجودی.

مجموعه وظایفی است که ضمن اهمیت و ضرورت در سیستم، از چشم کاربر سیستم فروش کتاب مخفی می‌ماند، این اعمال با فعالیت‌های عادی مثل سفارش کتاب، اطلاع از موجودی و سفارش مجدد و ارائه اطلاعات فروش به کادر مسئول سفارش‌دهی می‌باشد.

#### گزارش‌دهی مالی.

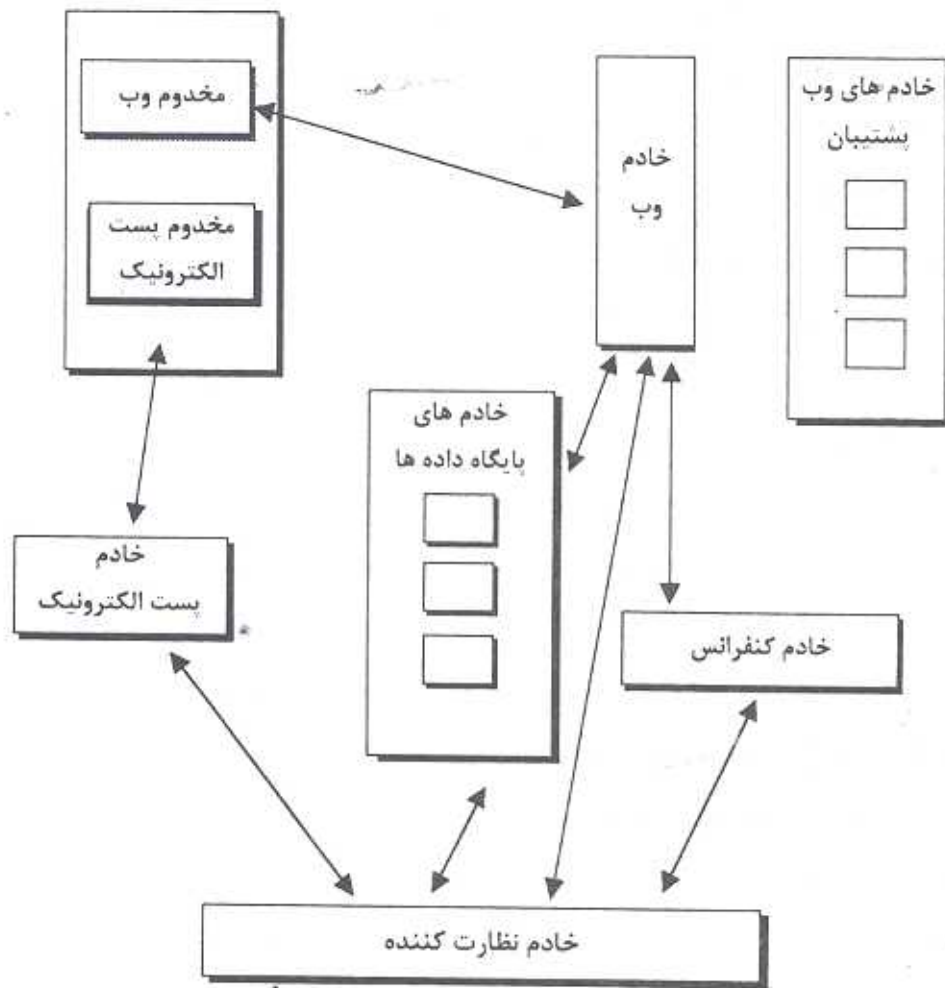


این نیز مجموعه کارهایی است که از چشم کاربر سیستم فروش کتاب مخفی مانده و در داخل سیستم از اهمیت برخوردار است. این نوع گزارش ها اطلاعاتی راجع به مدیریت مالی را در اختیار کمپانی گرداننده سیستم قرار داده و نیز درباره فروش روزانه، فروش سالانه و داده های پیچیده تر مثل تأثیر راهکارهای خاص فروش با نوحه به فروش کتاب های مرتبط با این راهکارها، اطلاعاتی را عرضه می کنند.

بنابراین اینها مجموعه اعمال و وظایفی است که توسط سیستم تجارت الکترونیک مختص به فروش یک کالا اجرا می شود. و در این مورد خاص، کالای مورد نظر کتاب است اما هیچ دلیلی وجود ندارد بخواهیم پوشاک، CD ها را و عتیقه جات و غیره را از این سیستم مستثنی کنیم. گرچه در این صورت وظایف سیستم تا حدی متفاوت خواهد بود. به عنوان مثال در سایت مختص به فروش لباس در انجام وظایف مرتبط با گزارش کالا نکته چندانی وجود ندارد.

قبل از بررسی ساختار و معماری این گونه سیستم ها، بهتر است به نقطه قوت توسعه این قبیل سیستم ها اشاره کنیم.

در سطح تحلیلی، تفاوت بسیار کمی بین سیستم تجارت الکترونیک و سیستم متداول تری وجود دارد. که به عنوان مثال برای گرفتن سفارش، از اپراتورهای تلفنی استفاده می کند و کانالوگ به شیوه معمول چاپ شده و برای مشتریان ارسال می شود. قطعاً وظایف و کارهایی وجود دارند که در سیستم های معمول به چشم نمی خوردند مثل اعمال مربوط به گزارش ها، هر چند که عمده اعمال و وظایف تقریباً مشابهند، و ممکن است به شیوه ای متفاوت اجراء شوند، مثلاً آگاهی از جزئیات کارت اعتباری توسط یک متصدی سفارش صورت خواهد گرفت تا یک صفحه وب.



شکل ۶-۲۸ معماری

معماری فنی سیستم نمونه کتاب در شکل ۶-۲۸ نشان داده شده است. این سیستم دارای اجزایی به

شرح زیر می باشد:

- **مخدومه های وب.** این مخدومه ها مرورگری را در اختیار دارند که با سیستم در ارتباط بوده و عمدتاً مرور کاتالوگ و دستور سفارش را به عهده دارند.
- **خادم وب.** این خادم در بردارنده تمامی صفحات وبی خواهد بود که مشتری بدانها دسترسی دارد و جهت فراهم کردن اطلاعاتی مثل در دسترس بودن کتاب، با سایر بخش های سیستم در ارتباط می باشد. قاعدتاً برای خرابی سخت افزاری بیش از یک خادم وب در دسترس خواهد بود. اگر خادم وب خراب شده و یا قطع شود، چنین رویدادی فوق العاده جدی بوده و به منزله بسته شدن در یک کتاب فروشی و از دست رفتن مشتریان آن خواهد بود. بادر نظر داشتن درآمد، سیستم های تجارت الکترونیک که در تجارت بسیار حائز اهمیتند، دارای سخت افزاری یدکی مشتمل بر خادمه های یدکی وب هستند.

• **خادم پستی** - این خادم، لیست پستی مشتریانی را نگهداری می کند که به عنوان مثال خواهند آمد تا در جریان روز ببینند که چه کتاب‌های جدید منتشر شده قرار گیرند. این خادم با خادم اصلی وب نیز در ارتباط خواهد بود. زیرا مشتریان ضمن مرور صفحات وب و برقراری ارتباط با آنها، آدرس‌های پست الکترونیک و خدمات مورد نظر خود را اطلاع می‌دهد. این مطلب بیانگر نکته مهمی درباره مخدومها و خادمها می باشد و آن این است که تعیین بی قید و شرط مخدوم و خادم در سیستم تجارت الکترونیک مطرح نیست و بلکه عملاً به رابطه بین عناصر موجود بستگی دارد. مثلاً خادم وب برای مخدومهایی که مرورگر را اجراء می‌کنند، به عنوان یک خادم عمل می‌کند. اما برای خادم پستی در حکم مخدومی انجام وظیفه می‌کند که آدرس‌های پست الکترونیک را جهت لیست‌های پستی در اختیار این خادم قرار می‌دهد.

• **خادم کنفرانس**. این خادمی است که بر کنفرانس‌ها نظارت دارد و تبادلات یک کنفرانس را به داخل برده، این نوشته‌ها را در بنجره مرتبط به کنفرانس نمایش می‌دهد و عناصر قدیمی را حذف می‌کند.

• **خادم‌های پایگاه داده‌ای** اینها خادم‌هایی هستند که بر پایگاه‌های داده‌ای مرتبط با برنامه کاربردی تجاری نظارت دارند. و عبارتند از پایگاه داده‌ای اصلی کتاب که شامل جزئیات هر کتاب است، پایگاه داده‌ای سفارشات که جزئیات سفارش‌های صورت گرفته توسط مشتریان یعنی سفارش‌های انجام شده قبلی و فعلی را در بر دارد، پایگاه داده‌ای مشتریان که حاوی جزئیات مشتریان کتابفروشان می‌باشد و پایگاه داده‌ای فروش که شامل جزئیات فروش کتاب‌های خاصی است، این پایگاه داده‌ای مخصوص برای بسیاری از کتابفروشان روی خط (آن لاین) مفید و مؤثر است، زیرا به آنها امکان می‌دهد تا لیست بر فروش‌ترین کتاب‌ها را به همراه پیشنهادات خاص در مورد آنها منتشر کنند. در بسیاری از سیستم‌های تجارت الکترونیک این پایگاه داده‌ای در تعدادی از خادم‌های اختصاصی پایگاه داده‌ای که دارای جایگزین هستند، اجراء می‌شوند. این گونه پایگاه‌های داده‌ای در فعالیت و عملکرد یک کمپانی تجارت الکترونیک تأثیر حیاتی دارند به طوری که اگر خادم پایگاه داده‌ای درست عمل نکند و خادم‌های بدکی نیز وجود نداشته باشند، این یک رخداد بسیار جدی خواهد بود که درآمدی را در پی نداشته و کتابفروش روی خط به دلیل خدمات‌دهی ضعیف از وجهه و اعتبار چندانی بر خوردار نخواهد بود. نکته مهم راجع به این بخش از سیستم آن است که هیچ دلیلی وجود ندارد که چرا یک فناوری قدیمی‌تر نظیر کامپیوتر بزرگ ناظر بر تغییرات می‌توانست در جهت اهداف پایگاه داده‌ای مورد استفاده قرار گیرد، در واقع بسیاری از سیستم‌های اولیه تجارت الکترونیک شامل یک خادم وب جلونی هستند که رابط و میانجی یک خادم بدون مخدوم می‌باشد.

غالباً این سیستم خادم بدون مخدوم همائی است که برای مدنی وجود داشته و به منظور برداشتن تجاری عادی‌تر مثل دریافت سفارش‌های تلفنی مورد استفاده واقع شده و تمامی خادم‌های پایگاه داده‌ای به واسطه نرم افزار سیستم که زمان اعمال تغییر در پایگاه داده‌ای را تعیین می‌کند، به روز خواهد شد. این

نرم افزار ابتدا تغییر را در پایگاه داده‌ای مورد استفاده اعمال می‌کند و سپس آن را در همه پایگاه‌های داده‌ای جایگزین به اجرا درمی‌آورد.

- خادم ناظر، این خادمی است که جهت نظارت بر اجرای سیستم مورد استفاده واقع می‌شود. این خادم توسط مدیر سیستم برای کنترل عملکرد صحیح سیستم و نیز تنظیم سیستم به‌منظور دستیابی به عملکرد بهینه به‌کار می‌رود.

## ۲۸-۶ فناوری‌های به‌کار رفته در تجارت الکترونیک

برخی فناوری‌های شبکه‌ای وجود دارند که در برنامه‌های کاربردی تجارت الکترونیک مورد استفاده قرار می‌گیرند. قبل از توصیف آنها لازم به‌ذکر است که بسیاری از فناوری‌های قدیمی هنوز در این نوع کاربرد به‌کار می‌روند، و بهترین نمونه آنها استفاده از فناوری پایگاه داده‌ای رابطه‌ای در ذخیره مقیاس وسیعی از داده‌هاست.



بسیاری از فناوری‌های قدیمی هنوز هم در برنامه‌های کاربردی تجارت الکترونیک استفاده می‌شود.

### ۲۸-۶-۱ سوکت‌ها

سوکت یک نوع کانال اطلاعاتی است که برای اتصال داخلی به کامپیوتر متصل به شبکه TCP/IP به‌کار می‌رود. سوکت به‌شعوی طراحی شده داده‌ها از مخدوم به سوکت قابل ارسال نبوده و نیز قابل بازگشت به مخدوم نمی‌باشد. زبان‌های برنامه‌نویسی مدرن مثل جاوا تسهیلات سطح بالایی را فراهم می‌آورند که به‌موجب آنها می‌توان سوکت را از به‌وسیله برنامه‌سازی به کامپیوتری متصل کرد که آدرس اینترنتی آن مشخص است و داده‌ها از طریق این اتصال قابل ارسالند. برنامه‌سازی لازم در انجام این کار معمولاً پیچیده‌تر از برنامه‌سازی لازم برای نوشتن یا خواندن داده‌ها بر روی یک فایل یا از روی آن نیست. سوکت‌ها اجرای سطح پایین اتصال هستند، کاربرد خاص سوکت در برنامه کاربردی کنفرانس است جایی که ورودی کنفرانس به خادم کنفرانس ارسال می‌شود که از نصب سوکت روی خادم استفاده می‌کند. سوکت‌ها یک مکانیزم سطح پایین اما شیوه‌ای بسیار مؤثر و کارآمد در تبادل داده‌ها در یک سیستم توزیع‌شده هستند که TCP/IP را اجرا می‌کند.

### ۲۸-۶-۲ اشیا توزیع شده

شیء توزیع‌شده شیء‌ای است که در یک سیستم توزیع‌شده روی کامپیوتری که معمولاً خادم است قرار می‌گیرد. سایر کامپیوترهای سیستم می‌توانند پیام‌هایی را به این شیء ارسال کنند گویی که این شیء روی خود آنها واقع شده است. نرم‌افزار سیستم مسئول قراردادن شیء، جمع‌آوری داده‌های لازم برای پیام و ارسال از طریق رسانه ارتباطی به‌کار رفته در سیستم است. اشیا توزیع‌شده نسبت به سوکت‌ها نشان‌دهنده



یک شیء توزیع‌شده چیست؟



سطح بالاتری از انتزاع هستند. صرفنظر از کد راه اندازی برنامه ساز از این واقعیت آگاه نیست که شیء در کامپیوتر دیگری قرار دارد. در حال حاضر سه فناوری رقیب برای شیء توزیع شده وجود دارد:

- **RMI**. این فناوری وابسته به زبان برنامه نویسی جاوا می باشد. این یک شیوه مختص به جاوا است. به طوری که تنها برنامه های نوشته شده از آن زبان می تواند با اشیا توزیع شده RMI در ارتباط باشند. این فناوری تنها برای سیستم های بسته جاوا مطلوب است که معمولاً این سیستم ها اگر هم اتصالی با دیگر سیستم ها داشته باشند، این اتصالات بسیار محدودند.
  - **DCOM**. این فناوری توسط شرکت مایکروسافت توسعه یافته و به برنامه های نوشته شده به برنامه هایی نظیر ویژوال بیسیک و ویژوال ++J (گونه مایکروسافت جاوا) امکان می دهد تا با اشیا واقع در کامپیوترهای دور در ارتباط باشند.
  - **CORBA**. پیچیده ترین فناوری شیء توزیع شده می باشد که توسط کنسرسیوم شرکت های کامپیوتری، مشتریان و کمپانی های نرم افزاری توسعه یافت. مهم ترین ویژگی روش CORBA آن است که این رهیافت چند زبانه است، چنان که برنامه سازان می توانند زبان های متعدد برنامه نویسی را برای ارسال پیام به اشیا CORBA به کار برند. رابط های CORBA برای زبان هایی نظیر Java، FORTRAN، LISP، Ada، Smaltalk وجود دارند. گرچه CORBA دوره های نخستین خود را می گذراند، اما نه بدیدی جدی برای تبدیل به فناوری غالب در اشیا توزیع شده به شمار می رود.
- مریت اصلی اشیا توزیع شده به سوکت ها این واقعیت است که به دلیل شمول کامل مدل شیء گرا در الگوی تحلیل و روش های طراحی مشابه به کار رفته در فناوری مداول شیء گرا، می تواند مورد استفاده قرار گیرند.

### ۲۸-۶-۳ Spaces

این فناوری نسبت به اشیا توزیع شده در سطح بالاتری از انتزاع قرار دارد که توسط دیوید خلرر استاد دانشگاه YALE توسعه یافت. فناوری Spaces از لحاظ ذخیره ریاد داده های ماندگار، یک سیستم توزیع شده را محسم می کند که کامپیوترهای موجود در این سیستم قادر به خواندن یا نوشتن آن هستند. این فناوری تداعی کننده یک سیستم توزیع شده به صورت مجموعه برنامه هایی است که با بکارگیری مکانیزمی مثل سوکت ها، پیام ها را با یکدیگر مبادله می کند و با یکسری اشیا توزیع شده که با استفاده از روش هایی با یکدیگر در ارتباطند، نمی باشد. بلکه، به کارگیری فناوری Spaces مزاحمی چون نوشتن داده ها، خواندن آنها یا کپی داده ها از یک حافظه ماندگار را در بر دارد. برنامه نویسی که از این فناوری استفاده می کند نگران جزئیاتی مثل محل ذخیره داده ها، روند جمع آوری داده ها در زمان جمع آوری آنها نمی باشد.

این فناوری هنوز در مراحل نخستین به سر می برد. گرچه اجراهای آن مدتی است که با استفاده از زبانهایی مانند C و C++ امکان پذیر بوده و هر چند که اجرای این فناوری در حوزه جاوا مثل JavaSpaces تعیین کننده آن است که فناوری مرکور بیش از پیش در کاربردهای جریان اصلی مورد استفاده خواهد بود.

### CGI ۴-۶-۲۸

اصطلاح CGI بر رابط مشترک گیت وی (Common Gateway Interface) دلالت دارد. این رابط، واسطه خادم وب است که به واسطه برنامه های اجرایی در خادم قابل دستیابی می باشد. عمده ارتباط متقابل مربوط به صفحات وب به موجب برنامه سازی دسترسی به CGI پیاده سازی می شود.

به عنوان مثال وقتی کاربر مرورگر به صفحه شامل فرم دسترسی می یابد، فرم را پر می کند و بعد آن را به خادم وب ارسال می کند. برنامه ای که به CGI دسترسی دارد، فرم را پردازش کرده و عملکرد مرتبط با فرم مثل بازیابی اطلاعات درخواستی در فرم را انجام می دهد. برنامه نویسی CGI در مجموعه وسیعی از زبان های برنامه سازی قابل انجام است. زبان انتخابی پرل (PERL) بوده که یک زبان پردازش رشته ای است هر چند که زبان های دیگری نظیر C و C++ وجود دارند در بردارنده امکاناتی برای پردازش CGI می باشند. اخیراً توسعه دهندگان جاوا این امکان را در اختیار برنامه نویسان قرار داده اند تا در این نوع برنامه سازی از زبان جاوا و فناوری معروف به servlets استفاده کنند. اینها بخش های کوچکی از برنامه جاوا هستند که در خادم وب جاسازی شده و به هنگام وقوع رویدادی مثل ارسال فرم، اجراء می شوند. سرویس ها نسبت به سایر زبان های برنامه نویسی قابلیت اجرایی بیشتری دارند.

### ۵-۶-۲۸ محتوای قابل اجراء

این اصطلاح به شمول برنامه ای در صفحه وب اطلاق می شود که به هنگام بازیابی صفحه وب توسط مرورگر اجراء می گردد. این برنامه قادر به انجام وظایف گوناگون از جمله انیمیشن و ارائه فرمی به کاربر برای وارد کردن داده ها می باشد. فناوری هایی وجود دارند که امکاناتی را برای وارد کردن محتوای قابل اجراء در صفحه وب فراهم می آورند و عبارتند از: اپلت ها، اکتیو ایکس و جاوا اسکریپت. اپلت برنامه ای نوشته شده به زبان جاواست که با صفحه وب ارتباط متقابل دارد. نکته مهم درباره این فناوری آن است که قابل انتقال می باشد. برنامه جاوا به راحتی از یک سیستم عامل به سیستم عاملی دیگر قابل انتقال بوده اما به طور بالقوه ناایمن است. دلیل ناایمنی اپلت ها این است که آنها می توانند به عنوان ابزار انتقال ویروس ها و سایر مکانیزم های دستیابی به کامپیوتر مورد استفاده قرار گیرند. وقتی صفحه مرورگر شامل اپلت توسط مرورگر منتقل می گردد، مشابه بارگذاری برنامه در کامپیوتر مخدومی است که مرورگر را اجراء می کند. خوشبختانه طراحان جاوا به نحوی این زبان را توسعه داده اند که نوشتن اپلت های ایجادکننده تخلفات ایمنی



بسیار دشوار است. متأسفانه راه حل اتخاذ شده مانع دستیابی ایلتها به منابع کامپیوتر مخدوم می باشد. به عنوان مثال ایلتها مجاز نیستند تا داده ها را از فایلی در کامپیوتر مخدوم بخوانند یا برنامه دیگری را در کامپیوتر اجرا کنند. گرچه پیشرفت هایی در ایلتها صورت گرفته که دستیابی محدود را امکان پذیر می سازد و مدل اصلی کاربرد ایلت در این شیوه محدود اجرایی می باشد.

اکتو ایکس (Active X) دیگر فناوری محتوای قابل اجرا است که توسط شرکت مایکروسافت ایجاد شد. این فناوری نیز شامل کد برنامه های حاساری شده در صفحه وب می باشد و تفاوت اصلی این فناوری با ایلتها در این واقعیت است که این اجزاء برنامه های به زبان های مختلفی نظیر ویژوال بیسیک و ++C قابل نوشتن هستند. اما این محتوای قابل اجرا نیز با مشکلات احتمالی امنیتی مواجه است. جاوا اسکریپت (Javascript) یک زبان برنامه نویسی ساده و مفسری است که مستقیماً در سند وب حاساری می شود. تفاوت آن با راه حل های اکتو ایکس و ایلتها این است که به جای برنامه مقصد در ایلت و اکتو ایکس، در برنامه جاوا اسکریپت، برنامه منبع در صفحه وب حاساری می گردد. جاوا اسکریپت یک زبان ساده است که در برنامه نویسی تقریباً سطح پایین، مورد استفاده قرار می گیرد.

#### ۲۸-۶-۶ پسته های مخدوم / خادم

این عبارت توصیف کننده مجموعه های نرم افزاری است که معمولاً نوعی از بردارش سیستم را انجام می دهند برخی نمونه های خاص این قبیل بسته ها عبارتند از:

- بسته های گپی داده ها. این نوع نرم افزار تراکشی را در یک پایگاه داده ای ایجاد کرده و آن را در تعدادی از پایگاه های داده ای تکراری اعمال می کند و تا زمانی که همگی آنها در هم زمانی و هماهنگی قرار نگیرند مانع دستیابی به این پایگاه های داده ای می شود.
- بسته های ایمنی. این بسته ها بر ترافیک داخل یک سیستم توزیع شده نظارت دارند و زمانی که یک تخلف امنیتی صورت می گیرد به مدیر سیستم هشدار می دهند، مثل وقتی که شخصی، با یک کلمه عبور ناشناخته سعی دارد وارد سیستم شود.

- ناظران تراکشن ها. اینها بسته های نرم افزاری هستند که بر تراکشن های انجام شده در یک سیستم توزیع شده نظارت دارند و متضمن آنند که بازگشت داده های صحیح حاصل تراکشن ها از ترتیب صحیح برخوردار است. عمده عملکرد این ناظران به تضمین این نکته مربوط می شود که حتی در محیطی که امکان وقوع خطاهای سخت افزاری یا انتقال وجود دارد، نتایج صحیح خواهند گشت.

#### ۲۸-۷ طراحی سیستم های توزیع شده

قبل از بررسی اصول طراحی به کار رفته در ایجاد سیستم های توزیع شده، به ویژه سیستم های تجارت الکترونیک، تصریح نکته ای که قبلاً نیز مورد اشاره گردیده، لازم به نظر می رسد و آن این است که در سطح تحلیلی تفاوت بسیار کمی بین یک سیستم توزیعی و سیستم محلی وجود دارد به طوری که مدل تحلیلی



بخطرات آورید که در سطح تحلیل تفاوت اندکی بین یک برنامه تجارت الکترونیک و یک برنامه سنتی و متعارف وجود دارد



یک سیستم جزئیات طراحی را در بر نخواهد داشت. از جمله این واقعیت که بهای یک کامپیوتر، سه کامپیوتر پردازش را انجام می دهند. این بدان معناست که ایجادکننده یک سیستم توزیع شده به طور عادی با یک مدل شی ای یا مدل کارکردی مشابه مدلی که در بخش های اولیه این کتاب معرفی شد، مواجه خواهد بود. و به همراه آن توصیف انواع کامپیوتر و سخت افزار شبکه ای به کار رفته مطرح خواهد گشت. فرآیند طراحی در بردارنده تبدیل مدل تحلیلی به یک مدل فیزیکی است که در عناصر سخت افزاری سیستم اجراء می گردد. برخی اصول طراحی وجود دارند که لازم است طراح سیستم توزیع شده از آنها آگاه باشد که طرح کلی آنها در ادامه این بخش ارائه می گردد.

### ۲۸-۷-۱ تطابق حجم انتقال با رسانه های مخابراتی

این یکی از آشکارترین اصول است، بدان معنا که برای ترافیک سنگین داده ها در یک سیستم توزیع شده باید از رسانه های مخابراتی سریع (و گران قیمت) استفاده کرد. روند تخصیص این گونه رسانه ها معمولاً پس از تصمیم گیری درباره قدرت پردازش توزیع شده در یک سیستم صورت می گیرد و گاه تکرارهای جزئی در پایان مرحله طراحی را در بردارد.

### ۲۸-۷-۲ نگهداری داده های مورد نیاز در حافظه سریع

این اصل نیز بدیهی و واضح است و مستلزم آن است که طراح، الگوهای دستیابی داده ها در یک سیستم را بررسی کرده و اطمینان حاصل کند که داده هایی که غالباً مورد دسترس هستند در رسانه ای با حافظه سریع نگهداری شوند. در بسیاری از سیستم ها، چنین داده هایی تنها ۵٪ داده های اصلی ذخیره شده در سیستم را تشکیل می دهند و از این رو اغلب راهبردهای مربوطه به ذخیره این داده ها در حافظه اصلی می توانند مورد استفاده قرار گیرند.

### ۲۸-۷-۳ نزدیک کردن داده ها به محل کاربرد آنها

این اصل طراحی تلاش می کند تا زمان طی شده برای ارسال داده در رسانه های کند مخابراتی، به حداقل برساند. بسیاری از سیستم ها به گونه ای هستند که کاربران اغلب به زیر مجموعه ای از داده ها دسترسی دارند. به عنوان مثال، سیستم توزیع شده به کار رفته در کاربرد بانکداری دارای پایگاه های داده ای مشتمل بر جزئیات حساب های مشتریان هستند. اغلب سؤالات پایگاه داده ای مربوط به یک شعبه توسط مشتریان آن شعبه پرسیده خواهد شد. بنابراین اگر داده های یک سیستم بانکداری بین خادم های موجود در شعبات توزیع شود، آن گاه داده های مربوط به مشتریان یک شعبه باید در آن شعبه نگهداری شود و تمامی داده های دیگر ممکن است در خادم های سایر مکان ها موجود بوده و هر گونه پرس و جوی ناشی از آنها باید از طریق خطوط کند مخابراتی ارسال گردد.



## ۲۸-۷-۴ به کارگیری کپی داده‌ها تا حد امکان



تکرار کنترل شده  
داده‌ها تأثیر مهمی بر  
عملکرد یک سیستم  
توزیع شده خواهد  
داشت.

تکرار داده‌ها عبارت است از نگهداری کپی‌های چند گانه داده‌ها در یک سیستم به‌طور هم‌زمان که دلایلی برای انجام این کار وجود دارد. دو دلیل اصلی یکی آن است که باعث ایجاد افزونگی می‌شود و این به سیستم توزیع شده امکان می‌دهد تا حتی در صورت خرابی کامپیوتر دارای اطلاعات سهم که معمولاً علت آن نقص سخت افزاری می‌باشد، سیستم توزیع شده به فعالیت خود ادامه دهد. دلیل دیگر فراهم آوردن شیوه اجرای اصل مطرح شده در قسمت قبل می‌باشد و آن عبارت است از تضمین این که داده‌ها در نزدیکی محل کاربردها نگهداری می‌شوند. به‌عنوان مثال، یک شرکت هتل‌داری ممکن است دارای یک پایگاه داده‌ای مرکزی رزرو باشد که اطلاعات رزرو اتاق در هتل‌های تحت نظارت شرکت را در بردارد. شرکت هتل‌ها ممکن است، دارای دو نقطه تماس برای مشتریانی باشد که قصد رزرو دارند، یکی خود هتل‌ها و دیگری دفتر مرکزی رزرو. یک راه تضمین عملکرد بالا، کپی داده‌های مربوط به یک هتل خاص و نگهداری آنها در خادم قرار گرفته در هتل می‌باشد. این بدان معناست که هر گونه رزرو انجام شده توسط هتل تنها مستلزم دسترسی به یک پایگاه داده‌ای محلی بوده و ترافیک یک خط کند مخابراتی ضرورتی ندارد. این اصل بسیار ساده‌ای به‌نظر می‌رسد انجام آن نیز راحت است، اما متأسفانه کارها هرگز به این سادگی نیست. در مورد رزرو هتل لازم است تا پایگاه‌های داده‌ای محلی مرتبط با هتل‌ها، با پایگاه داده‌ای مرکزی در ارتباط باشند. دلیل این کار آن است که علاوه بر مشتریانی که از امکانات ارائه شده رزرو توسط خود هتل‌ها استفاده می‌کنند، مشتریان دیگری نیز هستند که برای رزرو اتاق به دفتر مرکزی اتاق مراجعه می‌کنند. در صورت فقدان هماهنگی بین پایگاه داده‌ای موجود در دفتر مرکزی رزرو و پایگاه‌های داده‌ای تکراری در هر هتل، مشکلاتی به‌وجود می‌آید. مثلاً ممکن است به مراجع دفتر مرکزی رزرو گفته شود که در زمان موردنظر اتاق اضافی در یک هتل موجود است، در حالی که آن اتاق قبلاً توسط مشتری دیگری مستقیماً با هتل تماس گرفته، رزرو شده باشد.

مشکل فوق‌الذکر بدان معنا است که در سیستمی که ارتباطی بویا بین پایگاه‌های داده‌ای منفرد وجود دارد، لازم است تا هر پایگاه داده‌ای سایر پایگاه‌ها از تراکشن‌ها مطلع سازد و تضمین کند که تغییرات در تمام داده‌های کپی شده شده منعکس هستند. همچنین تأخیر نیز اجتناب‌ناپذیر است زیرا برای هم‌زمان شدن یک پایگاه داده‌ای با سایر پایگاه‌ها تغییرات در صف قرار گرفته و منتظر می‌مانند، ولی این بدان معنا نیست که کپی داده نباید مورد استفاده قرار گیرد؛ بلکه بدین معنا است که به‌منظور به حداقل رساندن میزان انتقال مرتبط با آن، طراحی دقیق ضرورت دارد.

لازم است که خاطرنشان ساخت در سیستم‌های توزیعی که ارتباط بویای بسیار کمتری بین داده‌ها وجود دارد، می‌توان از راهکارهای ساده‌تری استفاده کرد که عمده حجم مخابره را برطرف می‌سازند. به‌عنوان مثال، بانک‌ها معمولاً روزی یکبار و پس از اتمام کار تغییراتی را در پایگاه‌های داده‌ای مشتری انجام می‌دهند. این بدان معنا است که سیستم توزیعی بانکداری می‌تواند داده‌ها را در شعبات خود تکرار

کند و تنها کپی داده‌های تغییر یافته را روزی یکبار به پایگاه داده‌ای مرکزی ارجاع دهد. بنابراین هماهنگ کردن مکرر پایگاه‌های داده‌ای در طول روز ضرورتی ندارد.

همچنین لازم به ذکر است که تکرار یا کپی داده‌ها در این زیر بخش بر اساس نزدیک نگاهداشتن داده‌ها، کاربران و کاهش زمان انتقال در رسانه‌های کند مخابراتی مورد بحث و بررسی قرار گرفته است. دلایل منطقی دیگری نیز برای کپی داده‌ها وجود دارند، به عنوان مثال یک پایگاه داده‌ای بر کاربرد دارای صفه‌هایی از تراکنش‌ها است که منتظر اجراء هستند. اما به واسطه برخی پایگاه‌های داده‌ای دقیقاً مشابه که در خادم‌های هم‌زمان پایگاه داده‌ای نگهداری می‌شوند، می‌توان این صف‌ها را کاهش داد.

### ۲۸-۷-۵: رفع تنگناها و گلوگاه‌ها

خادم در سیستم توزیع شده اغلب تبدیل به یک تنگنا می‌گردد چرا که مجبور است تا ترافیک زیادی را که از انتظار پردازش صف‌های طولانی تراکنش‌ها به وجود می‌آید، کنترل کند و پیامد حاصل آن است که خادم‌هایی که منتظر نتایج پردازشی هستند، در بهترین حالت به طور مختصر بارگذاری شده و در بدترین حالت از کار باز می‌مانند. راهکار معمول جهت کنترل تنگناها، تقسیم بار پردازشی بین خادم‌ها می‌باشد و معمولاً آنها را لحاظ فیزیکی به خادمی که بار اضافی دارد، نزدیک می‌شوند.

### ۲۸-۷-۶: به حداقل رساندن ضرورت دانش جامع سیستم

سیستم‌های توزیع شده اغلب نیازمند اطلاعاتی درباره وضعیت کل سیستم هستند، مثلاً ممکن است به اطلاعاتی راجع به تعداد رکوردها در پایگاه داده‌ای مرکزی نیاز داشته باشند. نیاز به این قبیل اطلاعات، ترافیک بیشتری را ایجاد کرده و کارایی سیستم را کاهش می‌دهد، زیرا ترافیک شدیدی را در خطوط کند مخابراتی به وجود خواهد آورد. طراح یک سیستم توزیعی لازم است ابتدا وابستگی سیستم به اطلاعات سراسری را به حداقل رسانده و سپس تضمین کند که دانش لازم سریعاً به اجزایی در سیستم بدان نیاز دارد، منتقل می‌گردد.

### ۲۸-۷-۷: گروه‌بندی داده‌های مرتبط در محل یکسان

داده‌هایی که با یکدیگر مرتبط هستند باید در خادم یکسانی نگهداری شوند. به عنوان مثال، در برنامه کاربردی رزرو سیستم تعطیلات، بسته‌های جداگانه تعطیلات نزدیک به داده‌هایی نگهداری شوند که رزروهای فعلی آن بسته را توصیف می‌کند. قرار دادن این بسته‌ها به صورت مجزا در خادم‌های مختلف متضمن آن است که رسانه‌های کند مخابراتی با بارگذاری زیاد، حتی بیش از بیش بارگذاری خواهند شد. طراح یک سیستم توزیع شده باید تضمین کند اقلام داده‌ای مرتبط که غالباً به همراه هم بازیابی می‌شوند، بایستی تا حد ممکن در نزدیکی یکدیگر و روی خادمی یکسان قرار گرفته یا آن که حداقل در خادم‌هایی

واقع شوند که توسط رسانه‌های سریع مخابراتی نظیر رسانه‌هایی که در یک شبکه محلی به کار می‌روند، با یکدیگر در ارتباطند.

#### ۲۸-۷-۸ توجه به کارگیری خادم‌های اختصاصی

گاهی با به کارگیری خادمی تک منظوره برای یک عمل خاص، مزایای عمده عملکردی قابل دستیابی است تا اینکه استفاده از خادم بایگانه داده‌ای در دستور کار قرار گیرد.

#### ۲۸-۷-۹ تطبیق فناوری با ضروریات عملکرد

بسیاری از فناوری‌های مورد بحث در این فصل، موافقان و مخالفان مختلفی دارند. عامل دوم در این جا ضروریات عملکردی یک فناوری خاص است. مثلاً سوگن‌ها به عنوان یک ابزار ارتباطی، معمولاً بسیار سریع‌تر از اشیا توزیع شده هستند. در انتخاب فناوری، طراح بایستی از انتقال و باربرداری موجود مطلع بوده و فناوری را انتخاب کند که این بارها را به حداقل می‌رساند.

#### ۲۸-۷-۱۰ استفاده از توازی تا حد ممکن

یکی از مزایای عمده فناوری مخدوم / خادم آن است که شما می‌توانید خادم‌ها را اضافه کرده و تا حدی عملکرد سیستم را افزایش دهید. بسیاری از کارکردهای مربوط به کاربردهای تجارت الکترونیک می‌توانند از مزایای احراء توسط برخی خادم‌های موازی بهره‌مند شوند. این یک تصمیم‌گیری ساده نیست. طراح با به کارگیری تعدادی خادم، ضرورت ایجاد ارتباط بین آنها را مطرح می‌سازد. به عنوان مثال، یک خادم ممکن است قبل از اتمام وظیفه خود، برای تکمیل کاری خاص به خادم دیگری نیاز داشته باشد. این ارتباط می‌تواند تأخیراتی را به وجود آورد که در صورت بی توجهی طراح، باعث خنثی‌سازی مزایای اجرایی حاصل از توازی می‌شوند.

#### ۲۸-۷-۱۱ استفاده از تراکم داده‌ها تا حد امکان

الگوریتم‌هایی وجود دارند که در دسترس بوده و داده‌ها را متراکم ساخته و زمان انتقال داده بین یک جزء سیستم توزیع شده و جزئی دیگر را کاهش می‌دهند. تنها نکته اضافی در کاربرد این تکنیک، زمان پردازنده و حافظه مورد نیاز جهت انجام فشرده‌سازی در کامپیوتر فرستنده و بازکردن آن در کامپیوتر گیرنده می‌باشد.

#### ۲۸-۷-۱۲ طراحی نواقص

خرابی سخت افزار در اکثر سیستم‌های تجارت الکترونیک آسفار است و در سیستم‌های فروش روی خط (ONLINE)، معادل بستن درها به روی مشتریان می‌باشد. بخش مهمی از فرآیند طراحی، تحلیل



چرا خرابی سخت‌افزار در سیستم‌های تجارت الکترونیک می‌تواند



نقائصی است که امکان وقوع آنها در یک سیستم توزیع شده وجود دارد و طراحی سیستم با افزونگی کافی جهت جلوگیری از تأثیر جدی این گونه خرابی ها است و در بهترین حالت این تأثیر ممکن است کاهش زمان پاسخ گویی به تراکنش های اصلی باشد. یکی از تصمیماتی که توسط طراح اتخاذ می شود، کپی و تکرار خادم هایی است که در عملکرد یک سیستم توزیعی اهمیتی حیاتی دارند. یکی از راهکارهای سیستم با یکپارچگی زیاد تکرار سه باره یک خادم است که هر یک عمل مشابهی را به طور موازی انجام می دهند. هر خادم نتیجه ای را به وجود می آورد که بعداً مورد مقایسه قرار می گیرد. اگر هر سه خادم با نتیجه موافق باشند، نتیجه به کاربر یا خادمی که بدان نیاز دارد، منتقل می شود. اگر [نتیجه حاصل شده] یکی از خادم ها مخالف باشد، مشکلی وجود دارد و نتیجه مربوط به اکثریت ارسال شده و مدیر سیستم از مشکل احتمالی مطلع می گردد. تکرار خادم ها به عنوان یک راهکار تعدیل نقص می تواند به همراه طراحی سیستم با ایجاد توافقی در انجام کارها، مورد استفاده قرار گیرد.

#### ۲۸-۷-۱۳ به حداقل رساندن رکورد

وقتی داده در سیستم توزیع شده از کامپیوتری به کامپیوتر دیگر در جریان است، اغلب مجبور می شود از تعدادی کامپیوتر دیگر نیز عبور کند. بعضی از این کامپیوترها ممکن است تنها وظیفه ارسال برخی داده را داشته و سایر آنها ممکن است داده ها را به نحوی پردازش کنند. زمان سپری شده در هر یکی از این کامپیوترها به زمان رکورد معروف است. طراحی خوب برای یک سیستم توزیع شده آن است که تعداد کامپیوترهای واسطه در آن به حداقل برسد.

#### ۲۸-۷-۱۴ نتیجه

این بحث براساس ضرورت نیاز مختصر بوده و برخی راهکارهای به کار رفته در سیستم های توزیع شده را که اعمال تجارت الکترونیک را پیاده سازی می کنند، مورد بررسی قرار داده است. قبل از اتمام این قسمت نکته مهمی قابل ذکر است و آن این که یک راهکار ممکن است مخالف راهکاری دیگر باشد، یا به حد اقل رساندن رکورد و تکرار در پایگاه های داده ای می توانند در تضاد با یکدیگر باشند. افزایش تعداد داده های تکراری، مدت رکورد در یک سیستم را افزایش می دهد، در نتیجه طراحی سیستم توزیع شده همانند هرکار دیگری یک هنر محسوب می شود.

#### ۲۸-۸ مهندسی امنیت

افزایش وسیع در کاربرد عمومی سیستم های توزیع شده به برخی مشکلات عمده ایمنی منجر شده است. سیستم های توزیع شده قبلی با به کارگیری فناوری هایی نظیر فناوری های شبکه محلی، اغلب به یک مکان فیزیکی محدود شدند. این قبیل سیستم ها از لحاظ فیزیکی از کاربران خارجی جدا شدند و در نتیجه



امنیت اگر چه یک مشکل به شمار می‌رود اما مثل بسیاری از سیستم‌های تجارت الکترونیک، که توسط عموم و با به کارگیری یک مرورگر قابل دسترسی هستند، مشکل حادی نبود. برخی مزاحمت‌های خاص امنیتی که امکان رخداد آنها وجود دارد، در زیر شرح داده می‌شوند:

- یک مزاحم، ترافیک جاری در یک خط مخابراتی را کنترل کرده و اطلاعات حساس تولید شده توسط کاربر را جمع آوری کند. به عنوان مثال مزاحم ممکن است بتواند شماره، تاریخ اعتبار و نام دارنده یک کارت اعتباری را بخواند. سپس او می‌تواند این اطلاعات را برای سفارش کالا در اینترنت بکاربرد.
- مزاحم می‌تواند وارد سیستم توزیعی شده، به پایگاه داده‌ای دسترسی داشته و اطلاعات موجود در آن تغییر دهد، به عنوان مثال او ممکن است بتواند موجودی حساب آنان را که در سیستم بانکداری روی خط نگهداری می‌شود، تغییر دهد.

- مزاحم می‌تواند تغییرات جاری در یک خط مخابراتی را بخواند و داده‌های موجود در آن را به نفع خود تغییر دهد. به عنوان مثال او می‌تواند دستور مشتری نظام بانکداری روی خط (آن لاین) برای انتقال داده از یک حساب به حسابی دیگر را به گونه‌ای عوض کند که انتقال به حساب فرد مزاحم صورت گیرد.

- کارمند اخراجی و ناراضی یک شرکت برنامه‌ای را به سیستم توزیعی شرکت ارسال می‌کند و زمان بردارنده در سیستم را به خود اختصاص می‌دهد. این برنامه به تدریج از یک خادم به خادم دیگری می‌رسد تا اینکه سیستم متوقف می‌شود. این نوع حمله به حمله عدم پذیرش سرویس معروف است.

- کارمند ناراضی یک کمپانی برنامه‌ای را به سیستم توزیع شده ارسال می‌کند که موجب حذف فایل‌های مهم در سیستم می‌گردد.

اینها بخشی از مزاحمت‌های احتمالی در یک سیستم توزیع شده هستند. این مزاحمت‌ها بیشتر و بیشتر شده‌اند زیرا بخش اعظم انتقال در سیستم‌های تجارت الکترونیک با یکارگیری پروتکل‌هایی که جزئیات آنها در دسترس عموم قرار دارد، در اینترنت صورت می‌گیرد.

یکی از مهم‌ترین وظائف طراح سیستم توزیع شده، مهندسی این سیستم به گونه‌ای است که امکان موفقیت مزاحمتی با خطر زیاد در آن به حداقل برسد. به منظور انجام این کار، بایستی فناوری‌های خاصی را به کاربرد که در قسمت بعد تشریح می‌شوند.

## ۲۸-۸-۱ رمزگذاری

رمزگذاری به فرآیند تبدیل داده یا متن (متن عادی) اطلاق می‌شود به نحوی که متن غیر قابل خواندن شده و به علاوه از لحاظ عملی رمزگشایی آن توسط یک مزاحم امکان پذیر نمی‌باشد. فرآیند به کارگیری این فناوری در زیر توضیح داده می‌شود:



اکنون که اینترنت در بسیاری برنامه‌های کاربردی استفاده می‌شود امنیت مهمترین مسئله می‌باشد.



رمزگذاری چیست و چرا مفید است؟

۱- کامپیوتر فرستنده طی فرآیندی به نام رمزگذاری، متن را به فرمی غیرقابل خواندن تبدیل می‌کند.

۲- سپس داده‌های رمزگذاری شده از طریق یک خط مخابراتی غیر امن ارسال می‌شود.

۳- کامپیوتر گیرنده، متن رمزگذاری شده را پردازش کرده و آن‌گاه آن را به شکل اولیه‌اش تبدیل می‌کند. این فرآیند رمزگشایی نام دارد.

دو نوع رمزگذاری به‌کار می‌روند که اولی رمزگذاری متقارن است. در این نوع، پیام با استفاده از رشته‌ای به نام کلید به فرم رمزگذاری شده تبدیل می‌شود. با استفاده از کلید، تغییراتی در پیام صورت می‌گیرد. سپس کلید از طریق یک رسانه ایمن به گیرنده مخابره شده و برای انجام رمزگشایی توسط او مورد واقع می‌شود.

رمزگذاری متقارن گرچه مفید است اما دارای اشکال عمده‌ای است. اینکه اگر کلید توسط مزاحم شناسایی و کشف شود، پیام رمزگذاری شده در معرض خطر قرار می‌گیرد. یکی از رهیافتهای این مشکل، رمزگذاری کلید عمومی است. در این نوع رمزگذاری دو کلید به‌کار می‌رود که یکی کلید عمومی و دیگری کلید خصوصی نام دارد. کاربری که مایل است پیام‌های رمزگذاری شده را دریافت کند، کلید عمومی آنها اعلام خواهد کرد. سایر کاربرانی که می‌خواهند با این کاربر در ارتباط باشند، این کلید را برای رمزگذاری پیام به‌کار می‌برند. سپس زمانی که پیام‌ها توسط کاربر اصلی دریافت می‌شوند، از طریق کلید خصوصی رمزگشایی می‌گردند. مزیت اصلی این نوع رمزگذاری، مدیریت ساده کلید است. کلید خصوصی هرگز در اختیار کسی قرار نمی‌گیرد، و مزاحمی که داده‌های رمزگذاری شده در یک رسانه مخابراتی را کنترل می‌کند، قادر به رمزگشایی هیچ یک از پیام‌ها نیست. زیرا به کلید خصوصی دسترسی ندارد. عیب اصلی این نوع رمزگذاری این است که در انجام روند رمزگشایی، منابع بسیار زیادی وجود دارد، به همین دلیل سیستم‌های کلید عمومی معمولاً به رشته‌های کوتاه متنی یا متنی که باید کاملاً محرمانه و ایمن باشد، محدود می‌شوند. همچنین این سیستم با استفاده از تکنیکی به نام امضاهای رقمی که بعداً در این قسمت توصیف می‌گردد، برای اثبات و تأیید نیز به‌کار می‌رود.

فناوری عمده به کار رفته در اینترنت برای رمزگذاری متقارن، لایه سوکت‌های ایمن یا استاندارد (SSL) می‌باشد. این فناوری برای رمزگذاری داده‌های حساس نظیر شماره‌های کارت اعتباری به هنگام حرکت آنها از یک مرورگر به یک خادم وب یا از یک برنامه کاربردی به برنامه کاربردی دیگر به‌کار می‌رود.

## ۲۸-۸-۲ تابع خلاصه پیام

تابع خلاصه پیام الگوریتمی است که رقم بزرگی را - معمولاً به طول ۱۲۸۱ یا ۲۵۶ بیت - تولید می‌کند. که این تعداد نمائنده چکیده یا خلاصه کاراکترهای موجود در یک پیغام است و این ویژگی را دارد که اگر پیغام تغییر کرده و الگوریتم مجدداً به کار رود عدد عوض می‌شود. توابع خلاصه پیام، دارای کاربردهایی است که یکی از رایج‌ترین آنها شناسایی تغییرات یک پیام است، به عنوان مثال این واقعیت که یک معامله مالی به نفع تغییردهنده، در محاسبه تغییر یافته است. قبل از ارسال پیام، تابع خلاصه پیام در مورد آن اعمال شده و عدد بزرگی به وجود می‌آید. سپس پیام با شماره ۱ افزوده به انتهای آن، ارسال می‌گردد. در پایانه گیرنده، تابع خلاصه پیام اعمال شده و شماره به دست آمده با شماره ارسالی مقایسه می‌شود. اگر این شماره‌ها یکی باشند، پس پیغام دستکاری نشده و اگر یکی نباشد، معلوم می‌شود پیغام در محاسبه تغییر یافته است.

## ۲۸-۸-۳ امضاهای رقمی

امضا رقمی همان‌طور که از نام آن پیدا است، روش تعیین هویت فرستنده پیام برای گیرنده است. به نحوی که گیرنده اطمینان می‌یابد که پیغام واقعاً توسط فرستنده ارسال شده است و معمولاً رمزگذاری کلید عمومی برای این منظور به کار می‌رود. یک راه انجام این روند را در نظر بگیرید. دو کاربر کامپیوتری A, B قصد دارند با یکدیگر در ارتباط باشند و A می‌خواهد مطمئن شود که با B در ارتباط است. برای انجام این کار لازم است B یک کلید عمومی و اختصاصی داشته و بداند که کلید عمومی کدام است. A پیامی را برای B می‌فرستد و این پیام شامل متنی است که A از B می‌خواهد آن را با به کارگیری کلید خصوصی خود رمزگذاری کند. سپس این متن توسط B رمزگذاری شده و این پیام مجدداً برای A ارسال می‌شود و او با استفاده از کلید عمومی که B در اختیارش قرار داده پیام را رمزگشایی می‌کند. اگر پیغام همانند پیام ارسال شده باشد، B همان شخصی است که ادعا می‌کند ولی اگر پیام یکسان نباشد B هویت خود را ثابت نکرده است. این طرح از لحاظ ایمنی کلید خصوصی، از تمامی مزایای روش کلید عمومی برخوردار است هر چند که می‌تواند توسط شخصی قصد ایجاد سوءظن بین فرستنده و گیرنده را دارد مورد تهاجم قرار گیرد. این کار با تغییر پیام رمزگذاری شده و مجدداً برای فرستنده ارسال می‌شود، انجام پذیر است.

## ۲۸-۸-۴ گواهی‌های رقمی

گواهی رقمی یک سند الکترونیکی است که کاربر را درباره سازمان یا شخصی که با او در ارتباط است، کاملاً مطمئن می‌سازد. چهار نوع گواهی وجود دارد که می‌توانند مورد استفاده قرار گیرند:



• گواهی‌های مقام صادرکننده گواهی رسمی. مقام صادرکننده گواهی سازمانی است که گواهی‌های رسمی را صادر کرده است و دو نمونه از این قبیل سازمان‌ها، شرکت پست کانادا و دیگری خدمات پستی آمریکا است.

• گواهی‌های خادم. این گواهی‌ها در بردارنده جزئیاتی نظیر کلید عمومی خادم، نام سازمان دارنده خادم و آدرس اینترنتی خادم می باشند.

• گواهی‌های شخصی. اینها گواهی‌های مربوط به یک شخص هستند و شامل اطلاعات فیزیکی مانند آدرس فرد به همراه اطلاعات کامپیوتری نظیر کلید عمومی فرد و آدرس پست الکترونیک او هستند.

• گواهی‌های ناشر نرم افزار. این گواهی‌ها اطمینان می دهند که نرم افزار توسط یک شرکت خاص نرم افزاری تولید شده است.

به عنوان نمونه عملکرد این گواهی‌ها، گواهی‌های خادم را مدنظر قرار دهید. خادمی که از SSL استفاده می کند باید دارای گواهی SSL باشد. این گواهی دارای یک کلید عمومی است. وقتی مرورگر به خادم وصل می شود، آنگاه کلید عمومی برای به رمز کردن محاوره بین خادم و مرورگر به کار می رود.

## ۲۸-۹ اجزاء نرم افزاری در سیستم‌های C/S

### ۲۸-۹-۱ مقدمه

به جای در نظر داشتن نرم افزار به عنوان برنامه کاربردی یکپارچه قابل اجرا روی یک دستگاه، نرم افزاری که برای ساختار C/S مناسب است دارای چندین سیستم فرعی محزا است که می توانند به مخدوم و خادم اختصاص یافته یا آن که بین هر دو توزیع شوند:

سیستم فرعی محاوره / ارائه خدمات کاربر. این زیر سیستم تمامی توابع مرتبط با رابط گرافیکی کاربر مثل نمونه ارائه شده در مطالعه موردی کتاب فروش (GUI) را اجرا می کند.

سیستم فرعی برنامه کاربردی. این زیر سیستم نیازمندیهای تعیین شده برنامه کاربردی را در متن قلمروی عملیاتی آن اجرا می کند. به عنوان مثال، یک برنامه کاربردی تجاری ممکن است انواع گزارش‌های جایی را براساس عددی، محاسبات، اطلاعات پایگاه داده‌ای و سایر ملاحظات تولید کند. کاربر گروه افزار ممکن است امکاناتی را برای کنفرانس از راه دور یا پست الکترونیک فراهم سازد. در مطالعه موردی ما، این امر مستلزم آماده سازی گزارش‌هایی نظیر گزارش‌های توصیفی فروش کتاب می باشد. در هر دو مورد، نرم افزار کاربردی ممکن است طوری تقسیم بندی شود که برخی اجزاء روی مخدوم قرار گرفته و سایر اجزاء روی خادم واقع شوند.



ارجاع به وب

پرسشهای متداول در

خصوص اجزاء خادم و

مخدوم (C/S) در

آدرس زیر قرار دارد

[www.faqs.org/](http://www.faqs.org/)

[faqs/](http://faqs/)

[cclien\\_serfer\\_faqs/](http://cclien_serfer_faqs/)



میان افزار، زیرساختی را

فراهم می کند که اجزاء

نرم افزار C/S را قادر

به انجام عملیات داخلی

می سازد.



سیستم فرعی مدیریت پایگاه داده‌ها. این زیر سیستم، مدیریت و دستکاری لازم داده‌ها را در یک برنامه کاربردی انجام می‌دهد. مدیریت و دستکاری داده ممکن است به راحتی انتقال یک رکورد بوده یا به اندازه پردازش تراکنش‌های پیچیده SQL دشوار باشد. علاوه بر این سیستم‌های فرعی، دیگر بلوک سازنده نرم افزار که غالباً میان افزار نام دارد، در تمامی سیستم‌های C/S وجود دارد. میان افزار در قسمت ۲۸-۳-۲ توصیف گردید. عرفلی [ORF99] و همکارانش میان افزار را به عنوان «دستگاه عصبی» سیستم مخدوم / خادم معرفی کرده‌اند.

### ۲۸-۹-۲ توزیع اجزاء نرم افزاری

پس از تعیین نیازمندیهای اصلی کاربر مخدوم / خادم، مهندس نرم افزار باید تصمیم بگیرد که چگونه اجزاء نرم افزاری تشکیل دهنده سیستم‌های فرعی را که در قسمت ۲۸-۱-۱ توصیف شدند،

بین مخدوم و خادم توزیع نماید. وقتی عمده عملکرد مرتبط با هر یک از سه سیستم فرعی به خادم اختصاص می‌یابد طراحی خادم غنی (قربه) صورت گرفته است. و برعکس وقتی مخدوم عمده ارتباط متقابل و ارائه خدمات به کاربر، برنامه کاربردی و اجزاء پایگاه داده‌ای را اجرا می‌کند. طراحی مخدوم غنی (قربه) ایجاد شده است. مواجهه با مخدوم غنی معمولاً زمانی صورت می‌گیرد که معماری‌های خادم فایل و خادم پایگاه داده‌ای اجرا شوند. در این حالت، خادم حمایت از مدیریت داده را بر عهده دارد. اما تمامی برنامه کاربردی و نرم افزار GUI در مخدوم مستقر است.

خادم‌های غنی اغلب زمانی طراحی می‌شوند که تغییرات و سیستم‌های گروه افزار اجرا کردند. خادم حمایت لازم از برنامه کاربردی را جهت پاسخ‌گویی به تغییرات و ارتباط از جانب مخدومها فراهم می‌آورد. نرم افزار مخدوم منمرکز بر GUI و مدیریت ارتباط است. برای توضیح رهیافت یکی در تخصیص سیستم‌های نرم افزاری مخدوم / خادم، می‌توان مخدومهای غنی و خادم‌های غنی را مورد استفاده قرار داد. هر چند که یک رهیافت قطعی‌تر در تخصیص اجزاء نرم افزار، پنج بیکربندی مختلف را معرفی می‌کند:

• ارائه خدمات توزیع شده. در این رهیافت مقدماتی مخدوم / خادم، منطق پایگاه داده‌ای و منطق برنامه کاربردی روی خادم که معمولاً یک کامپیوتر بزرگ است باقی می‌مانند. هم‌چنین خادم با به‌کارگیری نرم افزاری نظیر CICS دارای منطقی برای آماده‌سازی اطلاعات صفحه نمایش می‌باشد. برای تبدیل اطلاعات صفحه ای کاراکتری ارسال شده از خادم، به رابط GUI در یک PC، نرم افزار خاص PC به کار می‌رود.

• ارائه خدمات از راه دور. توسعه روش نمایش توزیعی است که در آن پایگاه داده‌ای اصلی و منطق برنامه کاربردی روی خادم باقی می‌مانند و داده‌های ارسالی توسط خادم برای آماده‌سازی نمایش کاربر توسط مخدوم به کار می‌رود.



یک مخدوم " قربه " (fat)  
بسیاری از  
عملیات مشخصه‌های  
کاربردی را بر روی  
سرویس گیرنده پیاده  
سازی می‌نماید. یک  
مخدوم " نکیده " (thin)  
اکثر پردازشها  
را به سرویس گیرنده  
واگذار می‌نماید.



چه گزینه‌های  
بیکربندی برای اجزاء  
نرم افزار C/S وجود  
دارد؟

• منطق توزیع شده. تمامی وظائف ارائه خدمات به کاربر و مراحل مربوط به ورود داده مثل تأیید سطح میدانی، تنظیم پرس و جوی خادم، درخواست ها و اطلاعات به روز رسانی، به خادم محول می گردد. اعمال مدیریت پایگاه داده ای و مراحل پرس و جوی های مخدوم، تغییرات فایل خادم، کنترل مدل مخدوم و کاربردهای وسیع کاری، بر عهده خادم است.

• مدیریت از راه دور داده. برنامه های کاربردی خادم با قالب داده هایی که از جایی دیگر استخراج شده اند (مثلاً از یک منبع سطح اشتراکی) یک منبع جدید داده ای را به وجود می آورند. کاربردهای اختصاص یافته به مخدوم برای بهره برداری از داده های جدیدی به کار می رود که توسط خادم قالب بندی شده است. سیستم های پشتیبانی از تصمیم (تصمیم یار - DSS) در این مقوله جای دارند.

• پایگاه های داده ای توزیع شده. داده های تشکیل دهنده پایگاه داده ای در چندین خادم و مخدوم توزیع می شود.

از این رو مخدوم بایستی اجزاء نرم افزاری مدیریت داده را علاوه بر برنامه کاربردی و اجزاء GUI پشتیبانی کند.

در سال های اخیر، تأکید قابل توجهی بر فناوری مخدوم نکیده نیز صورت گرفته است. مخدوم نکیده یک کامپیوتر شبکه ای معروف است که تمامی پردازش برنامه کاربردی را به خادم غنی منتقل می کند. مخدومهای نکیده (کامپیوترهای شبکه ای) در مقایسه با ماشین های رومیزی، هزینه واحدی بسیار کمتری دارند و فقدان کارایی در آنها ناچیز بوده و قابل توجه نیست.

### ۲۸-۹-۳ رهنمودهایی در توزیع سیستم های فرعی برنامه کاربردی

با وجودی که قوانین مطلق دربارۀ توزیع زیر سیستم های برنامه کاربردی بین مخدوم و خادم وجود ندارد، اما عموماً رهنمودهای زیر رعایت می شود:

به طور کلی زیر سیستم ارتباط متقابل / ارائه خدمات، در مخدوم واقع می شود. دسترس پذیری محیط های مبتنی بر PC و پنجره ها و قدرت محاسباتی لازم برای میانجی گرافیکی کاربر، این روش را مقرون به صرفه می سازد.

اگر پایگاه داده ای بین چندین کاربر متصل از طریق LAN، مشترک باشد آنگاه پایگاه داده ای معمولاً روی خادم قرار می گیرد. سیستم مدیریت پایگاه داده ای و توانایی دستیابی به پایگاه داده ای نیز همراه با پایگاه داده ای واقعی روی خادم واقع می شود.

داده های ثابتی که برای ارجاع به کار می روند، بایستی به مخدوم اختصاص یابند. این کار داده ها را به کاربرانی که بدانها نیاز دارند، نزدیک می کند و ترافیک غیر ضروری شبکه ای و بارگذاری روی خادم را به حداقل می رساند.

### نقل قول

هر چند راهنمایی توزیع شده، ارزشمند می نماید. هر سیستمی باید بر شایستگی های خود تکیه کند. برای هر امتیازی که از یک سرویس گیرنده (یگوئید قریه) بدست می آید، طراح باید مجموعه معادلی از امتیازات منفی را لحاظ نماید.

زیر سیستم برنامه کاربردی براساس نوزیعی که آرایش های خادم و مخدوم را بهینه ساخته و شبکه ای که آنها را بهم متصل می سازد، بین مخدوم و خادم واقع می شود. به عنوان مثال، اجرای یک رابطه انحصاری متقابل معمولاً مستلزم جستجوی پایگاه داده ای است تا معلوم شود آیا رکوردی برای تطبیق پارامترهای یک الگوی جستجو وجود دارد. اگر تطبیقی یافت نشود، یک الگوی جایگزین جستجو به کار می رود. اگر برنامه کاربردی کنترل کننده این الگوی جستجو، کاملاً در خادم قرار گیرد، ترافیک شبکه ای به حداقل می رسد. اولین ارسال شبکه ای از مخدوم به خادم، پارامترهایی را برای الگوی جستجوی اولیه و ثانویه در بردارد. منطق برنامه کاربردی در خادم ضرورت جستجوی ثانویه را تعیین می کند. پیام پاسخ گویی به مخدوم شامل رکوردی است که حاصل جستجوی اولیه یا ثانویه می باشد. روش دیگر در قرار دادن منطق مخدوم در تعیین ضرورت جستجوی ثانویه، در برارنده پیامی برای اولین بازبینی رکورد، پاسخ گویی در شبکه در صورت یافت شدن رکورد، و پیغامی ثانوی شامل پارامترهای جستجوی ثانویه و پاسخ نهایی با رکورد بازبینی شده است. در جستجوی ثانوی ۵۰ درصد از زمان مورد نیاز است، یا قرار گرفتن منطق در روی خادم، جستجوی اولیه ارزیابی شده و در صورت لزوم جستجوی ثانوی آغاز می گردد و ترافیک شبکه تا ۳۳٪ کاهش می یابد.

تصمیم گیری نهایی درباره توزیع زیر سیستم ها به آنها باید براساس برنامه کاربردی خاصی باشد بلکه باید با توجه به ترکیب برنامه های کاربردی فعال در سیستم صورت گیرد. به عنوان مثال، یک پیاده سازی و نصب ممکن است شامل برنامه های کاربردی باشد که مستلزم پردازش وسیع GUI و پردازش کم پایگاه داده ای مرکزی هستند. این مسئله به کاربرد ایستگاه های کاری قوی در مخدوم و یک خادم منجر می گردد. با این بیکر بندی، سایر برنامه های کاربردی موافق با شیره مخدوم غنی هستند به طوری که قابلیت ها و توانایی های خادم نیازی به بهبود و ارتقاء ندارند.

با تکمیل کاربرد معماری مخدوم/ خادم، گرایش در جهت قرار دادن منطق قرار برنامه کاربردی روی خادم می باشد. این امر، آرایش تغییرات نرم افزار را به هنگام انجام تغییرات در منطق برنامه کاربردی، تسهیل می نماید. [PAU 95]

## ۲۸-۹-۴ اتصال سیستم های فرعی نرم افزار C/S

جهت ارتباط زیر سیستم های گوناگون ساختار مخدوم/ خادم، مکانیزم های مختلفی به کار می روند. این مکانیزم ها داخل شبکه و ساختمان سیستم عامل گنجانده شده و توسط کاربر نهایی در سایت مخدوم رؤیت نمی شوند. رایج ترین انواع مکانیزم های اتصال عبارتند از:

- کانال های اطلاعاتی (لوله) - به طور گسترده در سیستم های UNIX به کار رفته و پیام رسانی بین پردازنده های مختلف را که در سیستم عامل های گوناگون اجرا می شوند، امکان پذیر می سازند.



کدام گزینه ها برای  
به هم پیوستن  
زیر سیستم ها در  
اختیار می باشد؟



• فراخوانی رویه از دور - باعث می شوند که یک فرآیند، اجرای فرآیند یا پیمانه دیگری را که در سیستمی مختلف مستقر است، فراخوانی کند.

• ارتباط متقابل مخدوم/ خادم در SQL - برای انتقال درخواست های SQL و داده های مرتبط، از جزئی (که معمولاً روی مخدوم است) به جزء دیگر (معمولاً DBMS در روی خادم) مورد استفاده قرار می گیرد. این مکانیزم تنها محدود به برنامه های کاربردی مدیریت پایگاه داده های رابطه ای (RDBMS) می باشد.

• سوکت ها: در قسمت ۲۸-۶ مورد بحث قرار گرفتند.

به علاوه، پیاده سازی شیء گرای زیر سیستم های نرم افزاری C/S با استفاده از میانجی ORB نیز به اتصال منجر می شود. این رهیافت در قسمت بعدی مورد بحث قرار می گیرد.

### ۲۸-۹-۵ میان افزار و ساختارهای میانجی

زیر سیستم های نرم افزاری C/S که در قسمت های قبل مورد بحث قرار گرفتند، توسط اجزائی (اشیائی) پیاده سازی می شوند که باید در محدوده یک سیستم واحد (مخدوم یا خادم) یا در کل شبکه، قادر به ارتباط با یکدیگر باشند. «میانجی درخواست شیء» (ORB) میان افزاری است که به شیء مستقر در مخدوم امکان می دهد تا بر شیوه بسته بندی شده توسط شیء، واقع در خادم پیامی را ارسال کند. اساساً، ORB واسطه پیام است و تمامی ارتباطات و فعالیت های هماهنگی لازم را به منظور یافتن شیء، که پیام به آن ارسال شده در کنترل خود دارد، شیوه آن را فرا می خواند، داده های مناسب را به شیء ای منتقل می کند و داده های حاصل را مجدداً به شیء منتقل می سازد که در ابتدا پیام را تولید کرده بود.

سه استاندارد بر کاربرد که نظریه میانجی های درخواست شیء را اجرا می کنند - یعنی CORBA، COM، JavaBeans و این بخش به طور مختصر مورد بحث و بررسی قرار گرفتند. CORBA در توضیح کاربرد میان افزار ORB به کار خواهد رفت.

ساختار اصلی معماری CORBA در شکل ۲۸-۳ به نمایش درآمده است. به هنگام پیاده سازی CORBA در یک سیستم مخدوم/ خادم، اشیا و کلاس های شیء (فصل ۲۰) با استفاده از زبان توصیف رابط (IDL) در مخدوم و خادم تعریف می شوند. IDL یک زبان اعلانی است که به مهندس نرم افزار امکان می دهد تا اشیا، خصوصیات، شیوه ها و پیام های لازم در فراخوانی آنها را تعریف نماید. به منظور تطبیق درخواست شیء، مستقر در مخدوم با شیوه واقع در خادم، عناصر جانشین IDL در مخدوم و خادم ایجاد می شوند. اجزاء جایگزین در داده ای را به وجود می آورند که از طریق آن درخواست های شیء در سیستم C/S سازگار می گردند.



یک ORB (تجزیه  
گر درخواست شیء) یک  
شیء مقیم بر خادم را  
قادر می سازد که پیامی  
به شیوه بسته بندی  
شده به یک شیء مقیم  
در مخدوم، ارسال نماید.



### ارجاع به وب

آخرین اطلاعات در  
خصوص استانداردهای  
اجزاء در آدرس های زیر  
قرار دارد.

www.omg.com  
www.microsoft.  
com/COM  
www.java.sun.  
com/beans

### نقل قول

پذیرش CORBA  
یک گام مثبت است،  
اما برای حل چالش  
اکثر نرم افزارهای  
بحرانی کافی نخواهد  
بود. توماس موباری  
و رقائل مالو



از آنجا که درخواست های شیء در شبکه طی زمان اجراء رخ می دهند، باید مکاتیرمی برای ذخیره توصیف شیء تعیین شود تا اطلاعات مناسب درباره شیء و محل آن به هنگام ضرورت در دسترس باشند و انجام این کار به عهده مخزن میانی است.

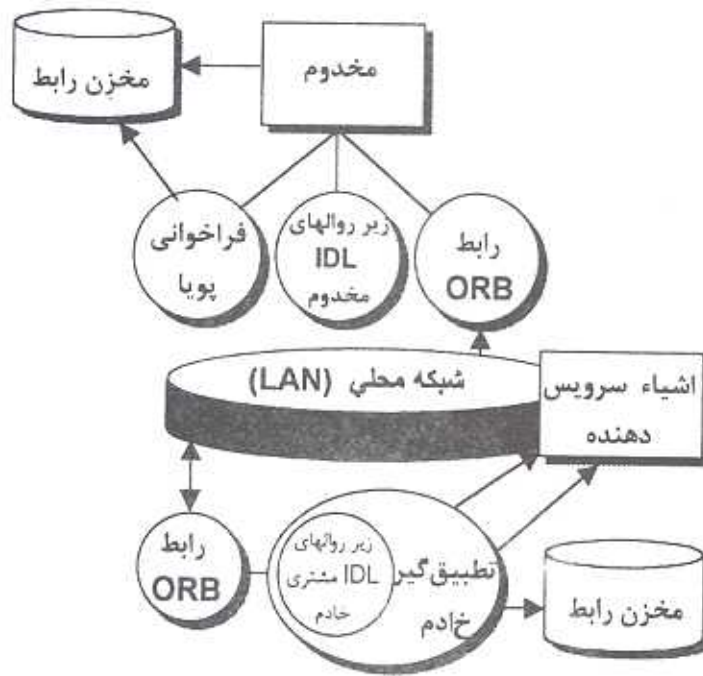
زمانی که کاربرد مخدوم بایسنی شیوه موجود در شیء را فراخوانی کند که در محل دیگری از سیستم قرار دارد، CORBA احضار پوبا را در جهت اهداف زیر به کار می برد: (۱) کسب اطلاعات مناسب و مربوط درباره روش مطلوب از مخزن رابط. (۲) ایجاد یک ساختار داده ای با پارامترهایی که با شیء منتقل می شوند. (۳) ایجاد یک درخواست برای شیء. (۴) فراخوانی درخواست. آن گاه درخواست به هسته ORB منتقل می شود. یک بخش اجرایی خاص در سیستم عامل شبکه که مدیریت درخواست ها را برعهده دارد. و درخواست اجراء می گردد.

درخواست از داخل هسته عبور کرده و توسط خادم بردارش می گردد. در سایت خادم، وفق دهنده شیء، اطلاعات و شیء را در مخزن رابط واقع در خادم ذخیره کرده، درخواست های وارده از مخدوم را پذیرفته و مدیریت می کند و بسیاری از وظایف دیگر مدیریت شیء را انجام می دهد. [ORF99] در خادم، جزءهای جایگزین IDL که مشابه عناصر تعریف شده در سیستم مخدوم هستند، به عنوان رابط در پیاده سازی عملی شیء مستقر در سایت خادم به کار می روند.

توسعه نرم افزار سیستم مدرن C/S به صورت شیء گرا است. توسعه دهندگان نرم افزاری با استفاده از معماری CORBA که به طور خلاصه در این قسمت توصیف شد، می توانند محیطی را ایجاد کنند که در آن اشیا را می توان در یک محیط شبکه ای بزرگ مجدداً مورد استفاده قرار داد. خوانندگان علاقه مند به اطلاعات بیشتر درباره CORBA و تأثیر کلی آن در مهندسی نرم افزار برای سیستم های C/S، می توانند به منابع [HOQ99] و [SIE99] مراجعه کنند.

## ۲۸-۱۰ مهندسی نرم افزار برای سیستم های C/S

تعدادی از مدل های مختلف فرآیند نرم افزاری در فصل ۲ معرفی شدند. گرچه طی توسعه نرم افزار برای سیستم های C/S می توان هر یک از آنها را مطابقت داده و به کاربرد. اما دو روش بیش از سایرین به طور معمول مورد استفاده قرار می گیرند: (۱) یک مدل تکاملی که از مهندسی نرم افزار رویدادی و یا شیء گرا استفاده می کند. و (۲) مهندسی نرم افزار مبتنی بر اجزاء (فصل ۲۷) که کتابخانه COTS و اجزاء داخلی نرم افزاری را به کار می برد.



شکل ۲۸-۷ یک معماری ساده CORBA

سیستم‌های مخدوم/خادم، با پیشرفت سیستم و تبدیل مجموعه‌ای از نیازمندیهای کلی نحاری به مجموعه‌ای از اجزاء معتبر نرم‌افزاری که در سیستم‌های مخدوم/خادم پیاده‌سازی شده‌اند، و به‌کارگیری فعالیت‌های مهندسی نرم‌افزار نظیر تحلیل، طراحی، ساخت و آزمون - گسترش می‌یابند.

## ۲۸ - ۱۱ ملاحظات مدل‌سازی تحلیل

فعالیت مدل‌سازی نیازمندیها برای سیستم‌های C/S تفاوت چندانی با روش‌های مدل‌سازی تحلیلی به‌کار رفته در معماری‌های معمول‌تر کامپیوتری ندارد. از اینرو اصول اولیه تحلیل مورد بحث در فصل ۱۱ و روش‌های معرفی شده الگوسازی تحلیل در فصل‌های ۱۲ و ۲۱، در نرم‌افزار C/S نیز کاربرد دارند. هر چند لازم به‌ذکر است که به‌دلیل استفاده بسیاری از سیستم‌های مدرن C/S از اجزاء قابل کاربرد مجدد، فعالیت‌ای واجد شرایط‌ساز مربوط به CBSE (فصل ۲۷) نیز به‌کار می‌روند.

از آن‌جا که مدل‌سازی تحلیل از تعیین جزئیات پیاده‌سازی خودداری می‌کند، تنها به هنگام ایجاد تحول در طراحی است که مسائل مربوط به تخصیص اجزاء نرم‌افزاری به مخدوم و خادم، مورد توجه قرار می‌گیرند. گرچه به دلیل کاربرد رهیافت تکاملی مهندسی نرم‌افزار در سیستم‌های C/S، تصمیمات اجرایی درباره شیوه کلی C/S (مثل مخدوم غنی در مقابل خادم غنی) ممکن است طی تکرارهای نخستین تحلیل و طراحی صورت گیرد.

## ۲۸-۱۲ طراحی سیستم های C/S

هنگامی که نرم افزار با به کارگیری یک معماری خاص کامپیوتری، جهت پیاده سازی توسعه می یابد، شیوه طراحی باید محیط خاص ساختار را مدنظر قرار دهد و ضرورتاً طراحی باید برای سازگاری با معماری سخت افزار، سفارشی شده باشد. وقتی نرم افزار با استفاده از معماری مخدوم / خادم برای اجراء طراحی می شود، شیوه طراحی جهت مطابقت با موارد زیر، باید به طور سفارشی اعمال گردد:

- داده ها و طراحی معماری (فصل ۱۴) فرایند طراحی را تحت الشعاع قرار می دهند. به منظور استفاده مفید مؤثر از قابلیت های سیستم مدیریت پایگاه داده ای رابطه ای (RDBMS) یا سیستم مدیریت پایگاه داده ای شیء گرا (OODBMS)، طراحی داده ها در مقایسه با برنامه های کاربردی عادی اهمیت بیشتری می یابد.
- به هنگام انتخاب مدل رویدادی، مدل سازی رفتاری (یک فعالیت تحلیلی، فصل های ۱۲، ۲۱) بایستی انجام شود و جنبه های کنترل گرا در مدل عملکردی بایستی به مدل طراحی تبدیل شوند
- جزء ارتباط متقابل / ارائه خدمات به کاربر در یک سیستم C/S تمامی وظایف مربوط به رابط گرافیکی کاربر GUI انجام می دهد. بنابراین اهمیت طراحی رابط (فصل ۱۵) بیشتر می شود.
- غالباً نگرش طراحی شیء گرا (فصل ۲۲) انتخاب می گردد. به جای ساختار ترتیبی ایجاد شده توسط یک زبان رويه ای، ساختار شیء با پیوند رویداد آغاز شده در GUI و رویداد کنترل کننده عمل در نرم افزار مبتنی بر مخدوم به وجود می آید.
- گرچه بحث درباره بهترین روش تحلیل و طراحی برای سیستم های C/S همچنان ادامه دارد، اما به نظر می رسد روش های شیء گرا (فصل های ۲۱، ۲۲) بهترین ترکیب خواص را در بردارند. هر چند که می توان شیوه های معمول (فصل های ۱۲ تا ۱۶) را نیز انتخاب کرد.

## ۲۸-۱۲-۱ طراحی معماری در سیستم های مخدوم / خادم

طراحی معماری یک سیستم مخدوم / خادم اغلب با عنوان «سک فرآیندهای برقراری ارتباط» توصیف می شود. یاس و همکاری [BAS98] این معماری را به شیوه زیر توصیف می کنند:

«هدف دستیابی به کیفیت قابلیت ارتقاء است. کار خادم قرار دادن داده ها در اختیار یک یا چند مخدوم است که معمولاً در داخل یک شبکه واقعند. مخدوم، موجب فراخوانی یا احضار خادم شده و آن هم به طور همزمان یا غیرهمزمان به درخواست مخدوم پاسخ می دهد. اگر خادم به طور همزمان عمل کند، همزمان با بازگشت داده ها کنترل را نیز به مخدوم باز می گرداند. ولی اگر همزمان عمل نکند، فقط داده ها را به مخدوم برمی گرداند (مخدوم رشته کنترلی خود را دارد).»

از آن جا که سیستم های مدرن C/S مبتنی بر اجراء هستند، معماری میانجی درخواست شیء ORB (شکل ۲۸-۳) برای اجرای این ارتباط همزمان یا غیرهمزمان به کار می رود. در سطح معماری، برای



هر چند نرم افزار C/S متفاوت است، شما می توانید شیوه های طراحی متعارف یا OO را با اندک تغییراتی مورد استفاده قرار دهید.



یک توضیح مفصل در خصوص معماری در فصل ۱۴ ارائه گردیده است.

تعیین جزئیات رابط، (IDL) زبان توصیف رابط CORBA به کار می رود. استفاده از IDL به اجزاء نرم افزار کاربردی امکان می دهد تا بدون اطلاع از عملکردهای داخلی به خدمات یا (اجزاء) ORB دسترسی یابند.

همچنین ORB مسئولیت هماهنگی ارتباط بین اجزاء مخدوم و خادم را نیز بر عهده دارد. برای انجام این کار، طراح یک وفق دهنده شی ای (که wrapper نیز نام دارد) را مشخص می کند خدمات زیر را ارائه می دهد: [BAS98]

- پیاپی سازی های جزء یا شیء ثبت می گردند.
- تمامی ارجاعات جزء یا شیء تفسیر شده و تطبیق می یابند.
- ارجاعات جزء (شیء) با اجزاء جزء متناظر مطابقت می یابند.
- اشیاء فعال شده و غیرفعال می گردند.
- به هنگام ارسال پیام ها، روش ها یا (عملیات) احضار می شوند.
- مشخصات امنیتی اعمال می گردند.

برای سازگاری اجزاء COTS که توسط فروشندگان مختلف عرضه می شوند و نیز اجزاء داخلی که ممکن است با به کارگیری فناوری هایی گوناگون اجزاء شده باشند، معماری ORB باید جهت دستیابی به عمل پذیری درونی میان اجزاء طراحی شود و در انجام این کار از مفهوم پل سازی استفاده می کند. فرض کنید که مخدومی با استفاده از پروتکل ORB X و خادمی با به کارگیری پروتکل ORB Y اجزاء شده است. هر دو پروتکل تحت فرمان CORBA هستند. اما به دلیل تفاوت های داخلی اجرایی، باید با پلی در ارتباط باشند که مکانیزمی را برای تبدیل و ترجمه بین پروتکل های داخلی فراهم می سازد. پل پیام ها را طوری ترجمه می کند که ارتباط بین مخدوم و خادم به راحتی برقرار گردد.

### ۲۸-۱۲-۲ رهیافت های معمول طراحی برای نرم افزار کاربردی

در سیستم های مخدوم/ خادم، نمودار گردش داده، (فصل های ۱۲ و ۱۴) می تواند، جهت تعیین محدوده یک سیستم، شناسائی وظایف سطح بالا و نواحی داده ای موضوعی (ذخیره داده ها) مورد استفاده قرار گرفته و تجزیه وظایف سطح بالا را امکان پذیر نماید. گرچه، با فاصله گرفتن از روش قدیمی DFD، تجزیه در سطح یک فرآیند مقدماتی تجاری متوقف شده و به سطح فرآیند اتمی نمی رسد.

در بافت C/S، فرآیند مقدماتی تجاری (EBP) را می توان به عنوان مجموعه وظایفی تعریف کرد که بدون قطع توسط کاربر در سایت مخدوم انجام می شوند. وظایف یا به طور کامل انجام شده و یا اصلاً اجزاء نمی گردند.

نمودار رابطه / موجودیت نیز نقش وسیعی را برعهده دارد. این نمودار برای تجزیه نواحی داده ای موضوعی (ذخایر داده) DFD مورد استفاده قرار می گیرد تا نمایش سطح بالایی از یک پایگاه داده ای که با



استفاده از RDBMS پیاده می گردد، تعیین شود. نقش جدید آن ایجاد ساختاری برای تعیین اشیا سطح بالای کاری می باشد. (قسمت ۲۸-۴-۳).

نمودار ساختاری، به جای استفاده به عنوان ابزاری برای تجزیه کاربردی، اکنون به عنوان یک نمودار هم گذاری (اسمبلی) مورد استفاده قرار می گیرد تا اجزاء موجود در راه حل روند مقدماتی تجاری را نشان دهد. این اجزاء که شامل اشیا رابط، اشیا کاربردی و اشیا پایگاه داده ای هستند، چگونگی پردازش داده ها را تعیین می کنند.

### ۲۸-۱۲-۳ طراحی پایگاه داده ای

طراحی پایگاه داده ای برای تعریف و سپس تعیین ساختار اشیا تجاری مورد استفاده در سیستم مخدوم/ خادم به کار می روند تحلیل لازم جهت شناسایی اشیا تجاری با به کارگیری روش های مهندسی فرایند تجاری که در فصل ۱۰ مورد استفاده قرار گرفتند، انجام می شود. برای تعریف اشیا تجاری، نشان گذاری مدل سازی تحلیل قراردادی (فصل ۱۲) نظر ERD را می توان مورد استفاده قرار داد، اما برای دستیابی به اطلاعات بیشتری که با به کارگیری علائم گرافیکی مانند ERD، کاملاً قابل استناد نیستند، مخزن پایگاه داده ای نیز بایستی ایجاد گردد.

در این مخزن، شیء تجاری به عنوان اطلاعاتی تعریف می شود که برای خریداران و کاربران سیستم قابل رؤیت هستند نه پیاده کنندگان آن، مثل کتاب در مطالعه موردی فروش کتاب. این اطلاعات که با استفاده از یک پایگاه داده ای رابطه ای اجرا می شوند و در مخزن طراحی قابل نگهداری هستند. اطلاعات طراحی زیر برای پایگاه داده ای مخدوم/ خادم جمع آوری می گردند: [POR 94]

- موجودیت ها. روی ERD برای سیستم جدید شناسایی می شوند.
- پرونده ها. که عناصر شناسایی شده روی ERD را اجرا می کنند.
- رابطه پرونده با فیلد. با تعیین این که چه فیلدهایی در کدام پرونده ها قرار می گیرند، طرح بندی پرونده ها را تعیین می کند.

- فیلدها (میدان ها). فیلدهای طراحی را تعریف می کنند (فرهنگ داده ها).
- رابطه پرونده با پرونده. قابل های مرتبطی را شناسایی می کنند که برای ایجاد نماها یا پرس و جوی های منطقی قابل اتصال به یکدیگر هستند.
- اعتبارسنجی رابطه. نوع روابط به کار رفته پرونده با پرونده با فیلد را برای اعتبارسازی تعیین می کند.

- نوع فیلد جهت وراثت خصوصیات فیلد از فوق طبقات میدانی به کار می رود مثل: (تاریخ، متن، شماره، ارزش، قیمت).
- نوع داده ها. مشخصات داده های موجود در یک فیلد.

### نقل قول

سازمان یک داده در یک پایگاه داده. بازنمای معنی یا معانی صحت داده و کارایی آن می باشد. حی ویدرهولد



گزینه هایی برای رزق داده در یک سیستم C/S وجود دارد؟

- نوع پرونده. برای شناسائی محل فایل به کار می‌رود.
- توابع فایل. کلید، کلید خارجی، ویژگی، فایل مجازی، فایل اشتقاقی و غیره.
- مقادیر مجاز. ارزش‌های مجاز برای فیلدهای نوع وضعیت را مشخص می‌کند.
- قوانین تجاری. قوانینی جهت ویرایش، محاسبه فیلدهای اشتقاقی و غیره.

با رواج بیشتر معماری‌های C/S، گرایش در جهت مدیریت داده‌های توزیعی تشریح شده است. در سیستم‌های C/S، که این شیوه را اجراء می‌کنند، جزء مدیریت داده‌ها در مخدوم و نیز خادم واقع است. مسئله اصلی در یافت طراحی پایگاه داده‌ای، توزیع داده‌ها است. یعنی آنکه چگونه داده‌ها بین مخدوم و خادم توزیع شده و در گروه‌های یک شبکه پراکنده می‌گردند.

یک سیستم پایگاه داده‌ای رابطه‌ای (RDBMS) با استفاده از زبان ساخت یافته پرس‌وجو (SQL)، دستیابی آسان به داده‌های توزیعی را امکان‌پذیر می‌سازد. مزیت SQL در ساختار C/S این است که «non – navigational» (غیر راهبردی) می‌باشد. در یک RDBMS، نوع داده‌ها، با استفاده از SQL تعیین می‌گردد. اما به هیچ‌گونه اطلاعات navigational (راهبردی) نیازی نیست. البته این بدان معنا است که RDBMS بایستی به اندازه کافی پیشرفته باشد تا محل تمامی داده‌ها را نگهداری کرده و بتواند بهترین مسیر آن را تعیین کند. در سیستم‌های نه‌چندان پیشرفته پایگاه داده‌ای، درخواست داده‌ها بایستی دلالت بر این داشته باشد چه چیزی قرار است مورد دستیابی قرار گیرد و محل آن کجا است. اگر نرم‌افزار کاربردی ملزم به نگهداری اطلاعات راهبردی باشد، مدیریت داده‌ها، برای سیستم‌های C/S بسیار پیچیده‌تر می‌گردد. هم‌چنین لازم به ذکر است که

سایر فنون توزیع و مدیریت داده‌ای نیز در اختیار طراح قرار دارند: [BER92]

استخراج دستی. کاربر مجاز است تا به‌طور غیرخودکار داده‌های مناسب را از خادم به مخدوم کپی کند. این روش زمانی مفید است که داده‌های ثابت مورد نیاز کاربر بوده و کنترل انتخاب را می‌توان به خود او واگذار کرد.

تصویر لحظه‌ای. این تکنیک با تعیین عکس فوری داده‌هایی که باید در فاصله‌های زمانی قبلاً تعیین شده از خادم به مخدوم انتقال یابند، انتخاب دستی را خودکار می‌نماید. این روش برای توزیع داده نسبتاً ثابتی مفید است که تنها مستلزم تغییراتی نادر می‌باشد.

تکرار یا کپی. این تکنیک زمانی مورد استفاده قرار می‌گیرد که کپی چندتایی داده‌ها در سایت‌های مختلف نگهداری می‌شوند. (مثلاً خادم‌های مختلف یا مخدوم‌ها و خادم‌ها). در این حالت سطح پیچیدگی بیشتر می‌شود زیرا سازگاری داده‌ها، تغییرات، ایمنی و پردازش همگی باید در سایت‌های چندگانه هماهنگ شوند.

پراکندگی (تکه تکه شدن). در این شیوه، پایگاه داده‌ای سیستم در چند پردازنده، تقسیم می‌شود. گرچه از لحاظ نظری عجیب است اما اجرای پراکندگی به‌طور خارق‌العاده‌ای دشوار است و غالباً به این مسئله پرداخته نمی‌شود.

طراحی پایگاه داده‌ای و به‌طور اخص، طراحی پایگاه داده‌ای برای سیستم‌های C/S موضوعاتی است که از حوزه بحث این کتاب خارج است. خواننده علاقه‌مند برای توضیحات بیشتر باید به منابع [BER92], [BRO91], [VAS93], [ORF99] مراجعه کند.

### ۲۸-۱۲-۴ بررسی کلی یک رهیافت طراحی

پروتر [POR 95] مجموعه اقداماتی را جهت طراحی و پردازش مقدماتی تجاری پیشنهاد می‌کند که ترکیبی از عناصر طراحی معمول و عناصر طراحی شیء‌گرا می‌باشد. فرض بر این است که قبل از شروع طراحی پردازش‌های بنیادی تجاری، یک مدل نیازمندیها که اشیاء تجاری را تعریف می‌کند توسعه یافته و اصلاح شده است. آن‌گاه مراحل زیر جهت ایجاد طراحی به‌کار می‌روند:

۱- در هر یک از پردازش‌های بنیادی تجاری، فایل‌های ایجاد شده، به روز درآمده، مرجع‌دار یا حذف شده را شناسایی کنید.

۲- از فایل‌های شناسایی شده در مرحله ۱ برای تعریف اجزاء یا اشیاء استفاده کنید.

۳- برای هر جزء قوانین تجاری و سایر اطلاعات شیء تجاری را که برای فایل مرتبط ایجاد شده است، بازیابی کنید.

۴- قوانین مربوط به فرآیند را تعیین کرده و تجزیه آنها را تا سطح یک شیوه انجام دهید.

۵- در صورت ضرورت، اجزاء اضافی لازم برای روش‌ها را مشخص کنید.

پروتر [POR95] برای نمایش ساختار اجزاء یک فرآیند بنیادی تجاری، نشان‌گذاری ساختاری را پیشنهاد می‌کند. (تصویر ۲۸-۴). هرچند که یک نماد شناسی متفاوت به‌نحوی به‌کار می‌رود که نمودار با ماهیت شیء‌گرای نرم‌افزار C/S مطابقت خواهد کرد. با توجه به شکل با پنج نماد مختلف مواجه هستیم:

شیء رابط. این جزء که جزء ارتباط متقابل یا ارائه خدمات به کاربر نیز نام دارد، معمولاً بر روی یک فایل واحد یا فایل‌های مرتبطی که از طریق پرس‌وجو به یکدیگر متصل شده‌اند، ساخته می‌شود. این جزء شامل روش‌هایی برای قالب رابط یا میانجی GUI و منطق کاربردی مرتبط و واقع در مخدوم می‌باشد. همچنین دارای SQL جاسازی شده است که انجام پردازش پایگاه داده‌ای را در فایل اصلی که رابط روی آن ساخته شده، تعیین می‌کند. اگر منطق کاربردی که معمولاً با شیء رابط در ارتباط است، به‌طور نمونه با به‌کارگیری ابزارهای میان‌افزار روی خادم اجرا شود، منطق کاربردی عمل‌کننده در خادم بایستی به‌عنوان یک شیء کاربردی مجزاء شناسایی گردد.

شیء پایگاه داده‌ای. این جزء برای تعیین پردازش پایگاه داده‌ای به کار می‌رود. نظیر ایجاد یا انتخاب رکورد که بر اساس فایلی غیر از فایل اصلی است که شیء رابط روی آن ساخته می‌شود، لازم به ذکر است اگر فایل اصلی که روی آن شیء رابط ساخته می‌شود به شیوه دیگری پردازش شود، مثلاً برای بازیابی فایل با ترتیب دیگر، دستور دوم SQL به کار می‌رود، تکنیک دوم پردازش فایل باید در روی نمودار ساختاری به صورت شیء جداگانه پایگاه داده‌ای شناسایی شود.

شیء کاربردی. این جزء که توسط یک شیء رابط یا شیء پایگاه داده‌ای مورد استفاده قرار می‌گیرد، به واسطه رهاساز پایگاه داده‌ای یا فراخوانی روبه از دور احضار می‌شود. همچنین می‌تواند برای شناسایی منطق کاربردی به کار رود که معمولاً مرتبط با پردازش رابطی است که برای اجرای عملیات به خادم منتقل شده است.

پیوستگی داده‌ای. وقتی یک شیء، شیء مستقل دیگری را احضار می‌کند، بیگامی بین دو شیء رد و بدل می‌شود. نماد پیوستگی داده‌ای بر این رویداد دلالت دارد.

پیوستگی کنترل. وقتی شیء، شیء مستقل دیگری را فرا می‌خواند و هیچ داده‌ای بین آنها رد و بدل نمی‌شود، نماد پیوستگی کنترل به کار می‌رود.

### ۲۸-۱۲-۵ تکرار طراحی فرآیند

مخزن طراحی (قسمت ۲۸-۴-۳) که برای نشان دادن اشیا تجاری به کار می‌رود، در ارائه رابط، برنامه کاربردی، و اشیا پایگاه داده‌ای نیز مورد استفاده قرار می‌گیرد. عناصر زیر شناسایی می‌شوند:

- شیوه‌ها. توصیف‌گر چگونگی اجرای یک قانون تجاری هستند.
- فرآیندهای مقدماتی پردازش‌های مقدماتی تجاری شناسایی شده در مدل تحلیلی را تعریف می‌کنند.
- اتصال فرآیند/ اجزاء. اجزائی را شناسایی می‌کند که راه حل یک فرآیند اولیه تجاری را به وجود می‌آورند.
- اجزاء. اجزاء نشان داده شده روی نمودار ساختاری را توصیف می‌کند.
- اتصال قانون تجاری و اجزاء. اجزائی را که در اجرای یک قانون فرضی تجاری اهمیت دارند، شناسائی می‌کند.
- اگر مخزن با به کار گیری یک RDBMS پیاده‌سازی شود، طراح به یک ابزار مفید طراحی دسترسی خواهد داشت. طراح را برای کمک به ساختمان و نگهداری آتی یک سیستم C/S ارائه می‌دهد.



## ۲۸-۱۳ مسایل آزمون

ماهیت توریعی سیستم های مخدوم / خادم، مشکلات خاص و منحصر به فردی را برای آزمون کنندگان نرم افزاری به وجود می آورد. بایندر [BEN92] تمرکز بر نکات عمده زیر را توصیه می کند:

- ملاحظات رابط گرافیکی مخدوم.
- محیط هدف و ملاحظات گوناگونی بایگامی.
- ملاحظات پایگاه داده ای توزیع شده (شامل داده های تکراری).
- ملاحظات پردازش توزیع شده (از جمله پردازش های تکراری).
- محیط غیرمقاوم هدف.
- روابط عملکرد غیرخطی.

راهبرد و تاکتیک های مربوط به آزمون C/S بایستی به شیوه ای طراحی شود که هر یک از مسایل فوق الذکر در آن مدنظر قرار گیرند.

## ۲۸-۱۳-۱ راهبرد کلی آزمون C/S

به طور کلی، آزمون نرم افزار مخدوم / خادم در سه سطح مختلف انجام می گیرد: (۱) کاربردهای مجزای مخدوم به شیوه غیرمتصل آزمون می شوند یعنی عملکرد خادم و شبکه زیربنایی مورد توجه قرار نمی گیرند. (۲) نرم افزار مخدوم و کاربردها. - - - - - به طور هماهنگ آزمون می شوند؛ اما عملیات شبکه به طور آشکار آزمون نمی شوند. (۳) معماری کامل C/S از جمله عملیات و عملکرد شبکه مورد آزمون قرار می گیرد. گرچه در هر یک از سطوح جزئی فوق، انواع مختلف آزمون ها انجام می گیرد، اما در کاربردهای C/S معمولاً با روش های آزمونی زیر مواجه هستیم:

آزمون های کارکرد برنامه کاربردی. کارکرد برنامه های کاربردی مخدوم با استفاده از روش های مورد بحث در فصل ۱۷، آزمون می شود. اساساً برنامه کاربردی به شیوه ای مستقل و ضمن تلاش و جهت کشف خطاها در عملکرد آن، مورد آزمون قرار می گیرد.

آزمون های خادم. اعمال هماهنگی و مدیریت داده ها توسط خادم، آزمون می شوند. عملکرد خادم

(زمان کلی پاسخ گویی و تحصیل داده ها) نیز مدنظر قرار می گیرند.

آزمون های پایگاه داده ای. صحت و یکپارچگی داده های ذخیره شده توسط خادم، آزمون می شود، تراکنش های اعمال شده توسط برنامه های کاربردی مخدوم مورد بررسی قرار می گیرند تا تضمین شود که داده ها به درستی ذخیره شده، به روز درآمده و بازیابی می شوند. بایگانی نیز مورد آزمون قرار می گیرد.



ارجاع به وب

اطلاعاتی مفید در

خصوص آزمون C/S

و منابع مرتبط در

آدرس زیر ارائه شده

است:

www.icon\_stl.net

-djmosley/



کدام گونه آزمون ها

برای سیستم های

C/S برقرار گردیده

اند؟

آزمون تغییرات. مجموعه آزمون‌هایی ایجاد می‌شوند تا تأیید شود هر یک از طبقات مطابق با ضروریات پردازش می‌شوند. این آزمون‌ها بر درستی پردازش و نیز مسایل عملکردی تأکید دارند مثل (زمان‌های پردازش تغییرات و آزمون حجم آنها).

آزمون ارتباطات شبکه. این آزمون‌ها تأکید می‌کنند که ارتباط میان گره‌های شبکه به درستی صورت گرفته و نیز این که انتقال پیام، تغییرات و ترافیک شبکه‌ای، بدون خطا انجام می‌شود.

آزمون‌های ایمنی شبکه نیز ممکن است به عنوان بخشی از این فعالیت آزمونی، انجام گیرند. برای انجام این روش‌های آزمونی، موسی [MUS93] توسعه پروفایل‌های مشتق از سناریوهای کاربردی مخدوم/ خادم را توصیه می‌کند. «پروفایل عملیاتی» بیانگر آن است که چگونه انواع مختلف کاربران با سیستم C/S عمل‌پذیری درونی دارند. یعنی پروفایل‌ها یک الگوی کاربردی را ایجاد می‌کنند که به هنگام طراحی و اجرای آزمون‌ها می‌توانند به کار روند. به عنوان مثال، برای نوع خاصی از کاربر، پرس‌وجوها، تغییرات و دستورهای چه درصدی از تراکنش‌ها را تشکیل خواهند داد؟

جهت گسترش پروفایل عملیاتی، اشتقاق مجموعه‌ای از سناریوهای کاربر که مشابه موارد کاربرد قبلاً طرح شده در این کتاب هستند، ضرورت دارد. هر سناریو به نکات زیر می‌پردازد: WHO, WHERE, WHAT, WHY یعنی آن که کاربر کیست، ارتباط متقابل سیستم در کجا روی می‌دهد (در معماری فیزیکی C/S)، تراکنش چیست و چرا رخ داده است. سناریوها از طریق قنون کسب نیازمندیها یا بحث‌های معمولی با کاربران نهایی قابل اشتقاقند. هر چند که نتیجه باید یکسان باشد. هر سناریو باید نمادی از وظایف و اعمال سیستم باشد که جهت سرویس‌دهی به یک کاربر خاص ضرورت دارند، ترتیب ضرورت آن وظایف را نشان دهد، بر زمان‌بندی و پاسخ‌گویی موردنظر دلالت داشته و بسامد کاربرد هر عمل را مشخص سازد. سپس این داده‌ها برای ایجاد (پرو فایل عملیاتی) یا یکدیگر ترکیب می‌شوند (برای همه کاربران).

راهبرد آزمون معماری C/S قبلاً تمایس با راهبرد آزمون سیستم‌های نرم‌افزاری است که در فصل ۱۸ توصیف شدند. آزمون در ابعاد کوچک آغاز می‌شود. یعنی آن که کاربرد یک مخدوم واحد مورد آزمون قرار می‌گیرد. یکپارچگی مخدومها، خادم و شبکه به تدریج مورد آزمون واقع شده، در نهایت کل سیستم به عنوان یک موجودیت عملیاتی مورد آزمون قرار می‌گیرد.

آزمون‌های متعارف پیمانه / زیر سیستم / جامعیت سیستم را (فصل ۱۸) به صورت بالا به پایین، پایین به بالا یا ترکیبی از هر دو را دنبال می‌کردند. یکپارچگی پیمانه واحد در توسعه C/S گرایش بیشتری به سمت توسعه موازی و یکپارچگی پیمانه‌ها در تمامی سطوح طراحی دارند. بنابراین، آزمون یکپارچگی در پروژه‌های C/S گاهی با استفاده از شیوه غیر افزایشی یا با یک ره یافت «big bang» (انفجاری) به بهترین وجه انجام می‌گیرد.



قنون تعیین نیازمندیها  
و Use-Case ها در  
فصل ۱۱ توضیح داده  
شده‌اند.

### نقل قول

برخی تحلیل‌گران  
محاسبات خادم /  
مخدوم را به چشم  
موج چهارم تغییرات  
در تاریخچه محاسبات  
می‌بینند  
و نادانان.



مباحث اصلی آزمون،  
ناحیه ای است که  
ارتباطی خوب از  
مشترکات موجود بین  
سیستم های متعارف و  
سیستم های مبتنی بر  
C/S برقرار می سازد.  
کلی بورن



مشخصه های آزمون  
خادم / مخدوم

این واقعیت که سیستم با استفاده از سخت افزار/ نرم افزار از قبل تعیین شده ساخته نمی شود، بر آزمون آن تأثیر می گذارد. ماهیت شبکه ای و پایگاهی متقابل در سیستم های C/S مستلزم آن است که به آزمون ترکیب بندی و سازگاری توجه و اهمیت زیادی نشان دهیم.

نظریه آزمون بیکربندی باعث آزمون سیستم در تمامی محیط های سخت افزاری و نرم افزاری شناخته شده ای می شود که سیستم در آنها عمل خواهد کرد. آزمون تطابق تضمین کننده رابطی است که از نظر کارکردی در پایگاه های سخت افزاری و نرم افزاری سازگار و هماهنگ باشد. به عنوان مثال، یک رابط از نوع پنجره ای ممکن است بسته به محیط اجرایی، از لحاظ بصری متفاوت باشد، اما بدون در نظر داشتن استاندارد رابط مخدوم، اعمال مشابه کاربر باید به نتایج یکسان منجر گردند.

### ۲۸-۱۳-۲ تاکتیک های آزمون C/S

حتی اگر سیستم C/S با استفاده از فناوری شی ای اجراء نگردد اما فنون آزمون شیء گرا (فصل ۲۳) مناسب به نظر می رسند زیرا می توان داده ها و مراحل تکراری را در طبقات شی ای سازمان دهی کرد که از مجموعه خصوصیات واحد برخوردارند. وقتی موارد آزمون برای کلاسی از

اشیا ایجاد می شوند (یا معادل آنها در یک سیستم توسعه یافته قرار دادی)، آن موارد بایستی در تمامی نمونه های آن کلاس، به طور وسیع قابل کاربرد باشند. دیدگاه یا نقطه نظر OO به ویژه زمانی ارزشمند است که رابط گرافیکی کاربر در سیستم های مدرن C/S مدنظر قرار گیرد. GUI اصولاً شیء گرا است و از رابط های قدیمی متمایز می گردد. زیرا بایستی در پایگاه های زیادی عمل کنند. به علاوه آزمون بایستی تعداد زیادی از مسیرهای منطقی را شناسایی کند. زیرا GUI، گستره وسیعی از اشیا گرافیکی را ایجاد کرده، دستکاری نموده و تغییر می دهد. پیچیدگی آزمون بیشتر می شود زیرا اشیا ممکن است حضور داشته یا غایب باشند. آنها ممکن است برای مدت زمانی وجود داشته و می توانند در هر نقطه ای از رومیزی ظاهر شوند.

این بدان معنا است که رهیافت قدیمی ضبط و بخش برای آزمون رابط های قراردادی کاراکتری باید جهت سازگاری با پیچیدگی های محیط GUI تغییر یافته و اصلاح گردند. نوع کاربردی الگوی ضبط و بخش به نام ضبط و بخش ساخت یافته، برای آزمون GUI توسعه یافته است. ضبط و بخش قدیمی ورودی را به صورت ضربه های کلیدی و خروجی را به صورت تصاویر صفحه نمایش ثبت می کند که اینها ذخیره شده و با ورودی ها و تصاویر خروجی آزمون های بعدی مقایسه می گردند. ضبط و بخش ساخت یافته براساس نمایش داخلی (منطقی) فعالیت های خارجی می باشد. روابط متقابل برنامه کاربردی با GUI به عنوان رویدادهای داخلی ضبط می شوند که به صورت Script یا زیربرنامه نوشته شده و به زبان ویژوال بیسیک مایکروسافت یا یکی از گونه های C به زبان خصوصی فروشنده، قابل ذخیره و نگهداری هستند.



ابزارهایی که GUIها را آزمون می کنند، به ضرورت هایی چون تأیید داده ها یا آزمون مسیر نمی پردازند. روش های آزمونی جعبه سیاه و جعبه سفید که در فصل ۱۷ مورد بحث قرار گرفتند در بسیاری از موارد کاربرد دارند و شیوه های خاص شیء گرای مطرح شده در فصل ۲۳، برای هر دو نرم افزار مخدوم و خادم مناسبند.

## ۲۸-۱۴ خلاصه

گرچه سیستم های مخدوم/ خادم می تواند یک یا چند مدل فرآیند نرم افزار را برای آزمون فرآیندهای تحلیل، طراحی و آزمونی قبلاً طرح شده در این کتاب را اتخاذ کنند، اما مشخصات خاص معماری C/S، مستلزم اختصاصی سازی روش مهندسی نرم افزار است. به طور کلی، مدل فرآیند نرم افزاری به کار رفته در سیستم های C/S از لحاظ ماهیتی تکاملی است و روش های فنی اغلب به شیوه های شیء گرا و یا مبتنی بر اجزاء گرایش دارند. توسعه دهنده بایستی اشیایی که به اجرای ارتباط متقابل کاربر و ارائه خدمات به او منجر می گردند، پایگاه داده ای و زیر سیستم های برنامه کاربردی را توصیف کند. اجزاء (اشیا) تعیین شده برای این سیستم های فرعی بایستی به سیستم مخدوم یا خادم اختصاص یابند و از طریق میانجی درخواست شیء قابل اتصال باشند. معماری های میانجی درخواست شیء، از طراحی های C/S که در آنها اشیا مخدوم پیام هایی را به اشیا خادم ارسال می کنند، پشتیبانی می نمایند. استاندارد CORBA از زبان توصیف رابط استفاده کرده و مخازن رابط و درخواست های اشیا را بدون توجه به محر آنها در شبکه مدیریت می کند.

تحلیل و طراحی سیستم های مخدوم/ خادم، نمودارهای گردش داده ها، نمودارهای رابطه موجودیت نمودارهای ساختاری تغییر یافته و سایر علائمی را به کار می برد که در توسعه برنامه های کاربردی قراردادی با آنها مواجه هستیم. راهبردهای آزمونی باید برای سازگاری با آزمون هایی که ارتباطات شبکه ای و تأثیر متقابل بین نرم افزار واقع در مخدوم و خادم را بررسی می کنند، نا تغییر و تحول یابند.



## مسایل و نکاتی برای تفکر و تعمق بیشتر

- ۱-۲۸ استفاده از نشریات تجاری یا منابع اینترنتی برای اطلاعات زمینه، گروهی از معیارهایی را برای ارزیابی مناسب مهندسی نرم افزار خادم / مخدوم معرفی می نماید.
- ۲-۲۸ پیج درخواست ( دستورالعمل ) ارائه دهید که یک خادم کامل برای داشتن طراحی و راهبرد مناسب باید دارا باشد.
- ۳-۲۸ پیج درخواست ارائه دهید که یک مخدوم با طراحی و راهبرد مناسب باید داشته باشد.
- ۴-۲۸ تحقیقات بیشتری در خصوص استانداردهای CORBA انجام داده و معین کنید که چگونه آخرین انتشارات آدرس های استاندارد میان انواع مختلف ORB های تهیه شده توسط دستفروشان، عمل می کنند.
- ۵-۲۸ در زمینه زبانهای دارای ساخت صف تحقیق کرده و مثالهای مختصری را تهیه کنید که چگونه یک معامله ممکن است استفاده از یک زبان را مشخص کند.
- ۶-۲۸ در زمینه نرم افزارهای گروهی تحقیقات گسترده ای انجام دهید و مطالب مختصری را برای کلاس خود تهیه نمایید. استاد شما ممکن است که وظایف خاص را به فرد ارائه کننده اختصاص دهد.
- ۷-۲۸ شرکتی در حال ارائه تقسیمات جدید بازرگانی برای فروش غیر جدی اسباب و لوازم و کالاهای بیرونی خود است. کانالوگ ها در شبکه گسترده جهانی (وب) نشر شده و سفارشات نیز از طریق سایت های وب یا پست الکترونیکی یا حتی از طریق فکس یا تلفن گرفته خواهند شد. سیستم خادم / مخدوم برای پشتیبانی روند این سبک برای سایت آن شرکت ساخته خواهد شد. گروهی از موضوعات ( مرحله بالا ) را که برای سیستم پردازش سفارش مناسب است، معرفی کرده و این موضوعات را به سه بخش فهرست نمایید: فعل و انفعالات کامپیوتر، پایگاه داده ها و درخواست نامه.
- ۸-۲۸ برای سیستمی که در شماره ۷-۲۸ توضیح داده شد، برای زمانی که یک محموله آماده می شود قوانین تجارت را معرفی کنید، و این در صورتی است که پرداخت آن به وسیله کارتهای اعتباری خواهد بود. سپس توانایی را در مورد پرداخت با چک نیز به آن بیافزایید.
- ۹-۲۸ برای جزئیات و توضیحات قابل درک در خصوص ورود یک کارمند به بخش خرید و فروش بازرگانی PC، نمودار تحول را شرح دهید.
- ۱۰-۲۸ مثالهایی از سه یا چهار پیام تهیه کنید که ممکن است به واسطه هر درخواستی از طرف مخدوم برای روشهای حمایت شده (موجود) در خادم، ناشی شده باشد.

## فهرست منابع و مراجع

- [BAS98] Bass, L., P. Clements, and R. Kazman, *Software Architecture in Practice*, Addison-Wesley, 1998.
- [BER92] Berson, Alex, *Client/Server Architecture*, McGraw-Hill, 1992.
- [BIN92] Binder, R., "A CASE-Based Systems Engineering Approach to client-Server Development," *CASE Trends*, 1992.
- [BIN95] Binder, R., "Scenario-Based Testing for Client Server Systems," *Software Development*, vol. 3, no. 8, August 1995, pp. 43-49.
- [BRO91] Brown, A.W., *Object-Oriented Databases*, McGraw-Hill, 1991.
- [FAR93] Farley, K.J., "Software Testing for Windows Developers," *Data Based Advisor*, November 1993, pp. 45-46, 50-52.
- [HOQ99] Hoque, R., *CORBA for Real Programmers*, Academic Press/Morgan Kaufmann, 1999.
- [ORF99] Orfali, R., D. Harkey, and J. Edwards, *Essential Client/Server Survival Guide*, 3rd ed., Wiley, 1999.
- [MOS99] Mosley, D., *Client Server Software Testing on the Desk Top and the Web*, Prentice-Hall, 1999.
- [MUS93] Musa, J., "Operational Profiles in Software Reliability Engineering," *IEEE Software*, March 1993, pp. 14-32.
- [PAU95] L.G. Paul, "Client/Server Deployment," *Computerworld*, December 18, 1995.
- [POR94] Porter, J., *O-DES Design Manual*, Fairfield University, 1994.
- [POR95] Porter, J., *Synon Developer's Guide*, McGraw-Hill, 1995.
- [SIE99] Siegel, J., *CORBA 3 Fundamentals and Programming*, Wiley, 1999.
- [VAS93] Vaskevitch, D., *Client/Server Strategies*, IDG Books, 1993.

## خواندنیهای دیگر و منابع اطلاعاتی

Although software engineering methods for client/ server systems are quite similar to conventional and OO systems, specialized knowledge and techniques are required. Worthwhile introductions to basic concepts have been written by Lowe and Heldt (*Client/Server Computing for Dummies*, 3rd ed., IOG Books Worldwide, 1999) and Zantinge and Adriaans (*Managing Client/Server*, Addison-Wesley, 1997). At an intermediate level, McClanahan (*Developing Client-Server Applications*, IOG Books Worldwide, 1999) covers a broad range of cis topics. On a more sophisticated level, Orfali and his colleagues [ORF99] and Linthicum (*Guide to Client/Server and Intranet Development*, Wiley, 1997) provide detailed guidelines for engineering cis applications. Berson (*Client/Server Architecture*, 2nd ed., McGraw-Hill, 1996) discusses component and architecture issues.

Network computers have become a hot technology topic (and a risky business strategy) in recent years. Sinclair and Merkow (*Thin Clients Clearly Explained*, Morgan Kaufmann, 1999), Friedrichs and Jubin (*Java Thin-Client Programming for a Network Computing Environment*, Prentice-Hall, 1999), Dewire (*Thin Clients*, McGraw-Hill, 1998),

and Kanter (*Understanding Thin-Client/Server Computing*, Microsoft Press, 1998) provide worthwhile guidance on how to design, build, deploy, and support thin-client systems.

Beginning with the modeling of business events, Ruble (*Practical Analysis and Design for Client/Server and GUI Systems*, Yourdon Press, 1997) provides an in-depth discussion of techniques for the analysis and design of cis systems. Books by Goldman, Rawles, and Mariga (*Client/Server Information Systems: A Business-Oriented Approach*, Wiley, 1999); Shan, Earle, and Lenzi (*Enterprise Computing with Objects: From Client/Server Environments to the Internet*, Addison-Wesley, 1997); and Gold-Bernstein

and Marca (*Designing Enterprise Client/Server Systems*, Prentice-Hall, 1997) consider c/s in a broader enterprise context.

Loosley and Douglas (*High-Performance Client/Server*, Wiley, 1997) explain the principles of software performance engineering and apply them to distributed systems architecture and design. Heinckiens and Loomis (*Building Scalable Database Applications: Object-Oriented Design, Architectures, and Implementations*, Addison-Wesley, 1998) emphasize database design in their guide for building client/server applications. Ligon (*Client/Server Communications Services: A Guide for the Applications Developer*, McGraw-Hill, 1997) considers a wide variety of communication-related topics including TCP/IP, ATM, EDI, CORBA, messaging, and encryption. Schneberger (*Client/Server Software Maintenance*, McGraw-Hill, 1997) presents a framework for controlling cis software maintenance costs and optimizing user support.

Hundreds of books address vendor-specific cis systems development. The following represents a small sampling:

Anderson, G.W., *Client/Server Database Design with Sybase: A High-Performance and Fine Tuning Guide*, McGraw-Hill, 1997.

Barlotta, M.J., *Distributed Application Development with powerbuilder 6*, Manning Publications, 1998.

Bates, R.J., *Hands-on Client/Server Internetworking*, McGraw-Hill, 1997.

Mahmoud, Q.H., *Distributed Programming with java*, Manning, 1998.

Orfali, R. and D. Harkey, *Client/Server Programming with javaBeans*, Wiley, 1999.

Sankar, K., *Building Internet Client/Server Systems*, McGraw-Hill, 1999.

Detailed guidebooks for cis testing have been written by Mosley [MOS99] and Bourne (*Testing Client/Server Systems*, McGraw-Hill, 1997). Both authors provide in-depth discussion of testing strategies, tactics, and tools.

A wide variety of information sources on client/server software engineering is available on the Internet. An up-to-date list of World Wide Web references that are relevant to cis systems can be found at the SEPA Web site:

<http://www.mhhe.com/engcs/compsci/pressman/resources/client-server.mhtml>