ارزیابی متدولوژی های طراحی معماری نرم افزار

(مهندسی نرم‌افزار پیشرفته - قسمت اول)

دکتر ناظمی ، مهندس علی ایزدی، مهندس پیمان عمادی ،مهندس ساناز حیدری، مهندس ایمان قزلباش

کلید واژه ها :
مهندسی نرم افزار ، معماری نرم افزار

مقدمه

امروزه انتخاب یک روش مناسب برای پیش‌برد پروژه‌های نرم‌افزاری و گزینش یک معماری مطلوب جهت مدیریت موفق این‌گونه پروژه‌ها، یکی از دغدغه‌های اصلی شرکت‌های نرم‌افزاری و سازمان‌های اجراکننده پروژه‌های نرم‌افزاری می‌باشد. با توجه به رشد روزافزون سازمان‌ها و گسترده شدن فرآیندها و عملیات‌شان از یک سو و پیشرفت روزافزون علوم مرتبط با حوزه فناوری اطلاعات و ارائه روش‌های نوین در این حوزه بخصوص در زمینه مهندسی نرم‌افزار، باعث شده است که معماری‌های مختلفی جهت پیشبرد مطلوب کار در این زمینه طراحی و ارائه گردد که انتخاب صحیح یکی از این معماری‌ها در اجرای پروژه‌های بومی از اهمیت خاصی برخوردار است. در این مجموعه از مقالات ما ابتدا به معرفی انواع معماری‌های نرم‌افزار پرداخته و سپس متدهای تحلیل معماری نرم‌افزار را بررسی کرده و مورد ارزیابی قرار خواهیم داد.

معماری‌های نرم‌افزار

تاکنون، معماری‌های متفاوتی به‌منظور طراحی و پیاده سازی نرم‌افزار، متناسب با رشد فناوری در بخش نرم‌افزار ارائه شده‌است. این معماری‌ها از یک‌طرف برخاسته از امکانات و ماهیت معماری سخت افزارهای زمان خود و از طرف دیگر بسته به نوع و نگرش در فناوری‌های ساخت نرم‌افزاری و انتظارات طرح شده توسط کاربران می‌باشند. هر معماری دارای شاخص‌ها و ویژگی‌های منحصر به‌فرد بوده و نرم‌افزارهائی که با اتکاء بر هر یک از این معماری‌ها پیاده‌سازی می‌گردنند، خصایص خود را از معماری به‌کارگرفته شده به ارث خواهند برد. در زیر به شرح مختصری از هر یک از معماری‌ها می‌پردازیم مسلما معماری‌ها و چارچوب‌های جدید مبتنی بر برنامه نویسی تحت وب و اینترنت نظیر Net.، J2EE قابلیت و لزوم استفاده از معماری‌های چند لایه‌ای را توصیه می‌نمایند. علاوه بر این در این نوع معماری عدم وابستگی بین مولفه‌ها (Components) و نیز سرویس‌ها (در غالب معماری Service Oriented) مورد تاکید قرار گرفته است.

- معماری Mainframe
- بررسی معماری سرویس‌گرا
- معماری سرویس‌های وب (Web-Based)
- معماری برنامه‌های مبتنی بر داده

معماری Mainframe

ویژگی‌ها

- معماری فوق در دهه ۱۹۷۰-۱۹۶۰ مورد توجه و استفاده جدی قرار گرفت.
- کامپیوتر اصلی (Host) مسئولیت انجام تمامی پردازش‌ها را برعهده دارد.
- کاربران با استفاده از ترمینال‌ها، قادر به ایجاد ارتباط با سیستم اصلی (Host) می‌باشند.
- ترمینال‌ها هوشمند نبوده و صرفاً به یک صفحه‌کلید و نمایشگر محدود می‌باشند.
- فشردن کلیدهای صفحه‌کلید، تنها چیزی است که ارتباط بین کاربران (ترمینال‌ها) و سیستم اصلی را معنی خواهد کرد.
- داده‌ها و کد برنامه بر روی یک سیستم (Host) یکسان ذخیره می‌گردند.

مزایا

- امنیت در این نوع معماری بسیار بالا است.
- با توجه به تمرکز داده‌ها و کد برنامه، مدیریت متمرکز و اعمال آن آسان خواهد بود.

معایب

- هزینه تهیه، اجاره و پشتیبانی این نوع سیستم‌ها بسیار بالا است.
- کد برنامه (منطق) به‌همراه داده‌های مربوطه در یک محل مستقر و از محیط پردازش یکسان استفاده می‌کنند.

اغلب برنامه‌های نوشته شده بر اساس معماری فوق محیط‌های واسط کاربرگرافیکی را حمایت نمی‌نمایند.

معماری File Server

ویژگی‌ها

- چرخش ۱۸۰ درجه‌ای نسبت به معماری Mainframe
- از سرویس دهنده یا سرویس دهندگان متعدد، به‌صورت متمرکز استفاده می‌گردد.
- منابع متفاوتی نظیر چاپ‌گر و یا فضای ذخیره‌سازی (هارد) به اشتراک گذاشته می‌شوند
- سرویس دهنده، فایل‌های مورد نیاز و ذخیره‌شده توسط منابع اشتراکی را برای کاربران ارسال خواهد کرد.
- کار درخواست شده توسط کاربر (کد برنامه و داده) بر روی سیستم کاربر اجراء خواهد شد.
- کد برنامه برروی سیستم کاربر (سرویس گیرنده) اجراء خواهد گردید.
- داده‌ها بر روی سرویس گیرنده مستقر خواهند شد.

مزایا

- برای استفاده از معماری فوق نیاز به صرف هزینه‌های بالا نخواهد بود
- از لحاظ به‌کارگیری منابع دارای انعطاف پذیری مناسبی است.

معایب

- کد برنامه بر روی سرویس گیرنده اجراء خواهد شد
- وجود محدودیت‌های خاص نظیر میزان حافظه و یا نوع پردازش‌گر سرویس گیرندگان، به‌کارگیری برنامه را با مشکل مواجه می‌سازد.
- بهبود عملکرد برنامه و یا اعمال اصلاحات مورد نظر همواره یکی از چالش‌های جدی است.

معماری Client/Server

ویژگی‌ها

- در معماری فوق از سرویس دهندگان و سرویس گیرندگان با خصایص متفاوت استفاده می‌شود
- اصل تقسیم کار دنبال می‌شود و سرویس دهنده عملیات سنگین با پردازش بالا و سرویس گیرنده عملیات سبک را انجام خواهند داد
- دو بخش متفاوت یک برنامه، در جهت انجام عملیات با یکدیگر تشریک مساعی می‌نمایند.
- سرویس گیرنده با ارسال درخواست و سرویس دهنده با پاسخ به درخواست جلوه‌ای از سیستم‌های عامل سرویس دهنده و سرویس گیرنده می‌تواند متفاوت باشد.
- عملیاتی را که یک برنامه انجام می‌دهد بین سرویس دهنده و سرویس گیرنده تقسیم می‌گردد.

مزایا

- بهره‌گیری مناسب از پتانسیل‌های سخت افزاری موجود با توجه به اصل تقسیم عملیات‌ها
- بهینه سازی استفاده و به‌کارگیری منابع اشتراکی
- بهینه سازی توانائی کاربران از بعد انجام فعالیت‌های متفاوت

معایب

- عدم وجود امکانات لازم برای کپسوله نمودن سیاست‌های راهبردی نرم‌افزار
- کاهش کارائی برنامه همزمان با افزایش تعدادکاربران همزمان
- بهبود عملکرد برنامه و یا اعمال اصلاحات مورد نظر همواره یکی از چالش‌های جدی است

تاکنون مدل‌های متفاوتی از معماری فوق پیاده‌سازی شده است:

۱ـ پردازش‌های مبتنی بر میزبان: مدل فوق به‌منزله یک مدل سرویس دهنده/سرویس گیرنده تلقی نشده و مشابه مدل Mainframe است

۲ـ پردازش‌های مبتنی بر سرویس دهنده: در این مدل سرویس دهنده تمامی‌پردازش‌های مربوطه را انجام و سرویس گیرنده مسئولیت ایجاد بخش رابط کاربر را برعهده خواهد داشت.

۳ـ پردازش‌های مبتنی بر سرویس گیرنده: تمامی عملیات بر روی سرویس گیرنده انجام خواهد شد. عملیات مربوط به بررسی صحت داده‌ها و سایر عملیات مربوط به منطق بانک‌های اطلاعاتی بر روی سرویس دهنده انجام خواهد شد.

۴ـ پردازش‌های مبتنی بر همیاری: در این مدل سرویس دهنده و سرویس گیرنده به‌منظور انجام یک فعالیت با یکدیگر تشریک مساعی خواهندکرد

معماری Client Server : Two Tier

ویژگی‌ها

- مشابه مدل Client Serverدر بخش قبل می‌باشد.
- در این مدل از یک سرویس دهنده و یک سرویس گیرنده در شبکه استفاده می‌گردد.
- مدل فوق از سه بخش که در دو لایه سرویس دهنده و سرویس گیرنده قرار خواهند گرفت، تشکیل می‌گردد.
- بخش‌های رابط کاربر، مدیریت پردازش‌ها و مدیریت بانک‌های اطلاعاتی بخش‌های سه گانه مدل فوق می‌باشند.
- منطق برنامه بین دو محل فیزیکی توزیع می‌گردد.

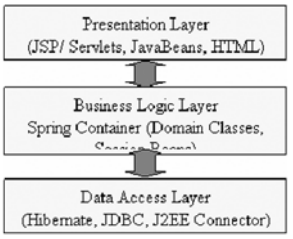
مزایا

- مناسب‌ترین روش برای پردازش‌های توزیع شده در یک شبکه با حداکثر یک‌صد کاربر
- سهولت در امر پیاده‌سازی
- نسبت‌دهی مستقیم رابط کاربر با منابع تأمین داده‌ها

معایب

- عدم وجود امکاناتی برای کپسوله نمودن سیاست‌های راهبردی نرم‌افزار
- کاهش کارآئی برنامه همزمان با افزایش تعدادکاربران همزمان
- عدم وجود انعطاف لازم از بعد انتقال یک برنامه از سرویس دهنده‌ای به سرویس دهنده دیگر بدون انجام تغییرات اساسی

معماری سه لایه‌ای، چند لایه‌ای



ویژگی‌ها

- مدل فوق در سال ۱۹۹۰ عرضه شده است
- در این مدل از یک میانی دیگر بین سرویس گیرنده (رابط کاربر) و سرویس دهنده بانک اطلاعاتی استفاده می‌شود.
- لایه میانی شامل مجموعه‌ای از ابزارها برای دستیابی به منابع سیستم، صرف‌نظر از نوع Platform است.
- لایه میانی مسئولیت مدیریت پردازش‌ها را برعهده خواهد گرفت.
- لایه میانی مسئولیت تجزیه و یا ترکیب نتایج حاصله از منابع داده‌ای نظیر بانک‌های اطلاعاتی را برعهده دارد.
- بخش‌های رابط کاربر، مدیریت پردازش‌ها و مدیریت بانک‌های اطلاعاتی بخش‌های سه گانه مدل فوق می‌باشند.
- لایه میانی خود می‌تواند به دو و یا بیش از دو بخش با عملکردهای متمایز تقسیم گردد (Multi-Tier). لایه منطق تجاری می‌تواند برروی سرویس دهنده‌های متعدد قرار گیرد.
- مدل فوق گزینه‌ای مناسب برای پیاده سازی نرم‌افزار بر روی اینترنت است.

مزایا

- افزایش کارآئی، انعطاف پذیری، قابلیت استفاده مجدد و توان پشتیبانی
- ارتقاء کارآئی همزمان با افزایش تعدادکاربران
- مخفی نمودن پیچیدگی‌های موجود با توجه به ماهیت پردازش‌های توزیع شده از دیدکاربران
- ارائه امکانات لازم به برنامه نویسان به‌منظور طراحی و پیاده سازی نرم‌افزارها با یک رویکرد مشابه
- ارائه امکانات لازم به برنامه نویسان به‌منظور تبعیت از روش‌های یکسان برای دست‌یابی به داده‌ها
- استفاده از الگوهای طراحی و برنامه نویسی در پیاده سازی و توسعه سریع‌تر

معماری سرویس‌گرا

در چهار دهه اخیر، پیچیدگی نرم‌افزارها روز بروز بیشتر شده و تقاضا برای نرم‌افزارهای قدرتمندتر افزایش یافته است. در این میان، به نظر می‌رسد که روش‌های قدیمی جواب‌گوی نیازهای در حال رشد کنونی نیستند و نیاز به ایجاد و به‌کارگیری روش‌هایی است که به‌وسیله آن‌ها بتوان بر این پیچیدگی‌ها در زمان‌هایی کوتاه‌تر غلبه کرد. از طرفی امکان کنار گذاشتن سیستم‌های نرم‌افزاری موجود که تا به حال مشغول سرویس‌دهی به مشتریان بوده‌اند، وجود ندارد و می‌بایست سیستم‌های جدید را به‌صورت یکپارچه و درکنار همین سیستم‌ها به‌وجود آورد. معماری سرویس‌گرا، با تکیه بر محاسبات توزیع شده و بر پایه شبکه‌ها و لایه‌های میانی و همچنین زبان‌هایی که تولید نرم‌افزارهای توزیع شده را فراهم می‌کنند، به‌عنوان راه حلی مناسب جهت از میان برداشتن مشکلات و مسائل مذکور مطرح گردیده است. در این مقاله ضمن معرفی این معماری به محاسبات

سرویس‌گرا و نیازهای آن پرداخته و معماری سرویس‌گرای توسعه یافته به عنوان راه حلی برای رفع این نیازها ارائه می‌شود.

تعاریف گوناگونی از معماری سرویس‌گرا ارائه شده است که از جمله آن‌ها می‌توان به تعاریف زیر اشاره کرد:

مجموعه قوانین، سیاست‌ها و چارچوب‌هایی که نرم‌افزارها را قادر می‌سازد تا عملکرد خود را از طریق مجموعه سرویس‌های مجزا و در عین حال مربوط به هم در اختیار سایر درخواست‌کنندگان قرار دهند تا بتوانند بدون اطلاع از نحوه پیاده سازی و تنها از طریق رابطه‌های استاندارد و تعریف شده، این سرویس‌ها را پیدا کرده و فراخوانی نمایند.و یا معماری سرویس‌گرا روشی برای ساخت سیستم‌های توزیع شده‌ای است که در آن‌ها عملکرد سیستم بصورت سرویس در اختیار کاربران و یا سایر سرویس‌ها قرار می‌گیرد. از دیگرتعاریف ارائه شده می‌توان به "واحدهای نرم‌افزاری آماده در شبکه (Network-Available Software Unit) "یا"سرویس‌های سطح حرفه وکسب وکار (services Business-level) "اشاره‌کرد.
در حال حاضر، فناوری سرویس‌های وب (Web Services) و پیاده‌سازی نمونه‌های موفق از آن، نشان داده است که SOA یا همان معماری سرویس‌گرا می‌تواند به عنوان راه حلی عملی و دست یافتنی در طراحی سیستم‌های جدید و یکپارچه‌سازی سیستم‌های بزرگ موجود، مطرح گردد. البته ذکر تفاوت سرویس‌های وب و SOA در اینجا لازم به نظر می‌رسد.سرویس‌های وب مجموعه‌ای از فناوری‌هایی همچون WSDL،SOAP،XML و UDDI می‌باشد که امکان ارائه راه‌حل و برنامه‌نویسی جهت رفع مشکلی خاص را فراهم می‌نماید، در حالی که SOA یک معماریست و از مجموعه مشخصی از فناوری‌ها فراتر می‌باشد. اگرچه SOA بر اساس این فناوری‌ها راه حل ارائه می‌نماید، اما در عین حال مستقل از هر یک از آن‌ها است.
آن‌چه اهمیت دارد تعریف سرویس به عنوان مهم‌ترین عنصر این معماری می‌باشد. سرویس، رفتار قراردادی تعریف شده‌ای است که هر قطعه‌ای می‌تواند آن‌را جهت استفاده سایر قطعات در سیستم تهیه و پیاده سازی نماید.
در این معماری، همه توابع به عنوان سرویس تعریف می‌گردند. این توابع شامل توابع حرفه (Business functions) و فعالیت‌های حرفه (Business transactions) می‌باشند که فعالیت‌های حرفه خود شامل توابع سطح پایین (Low-level functions) و توابع سیستمی‌سرویس‌ها (System service functions) هستند.
سرویس‌ها به‌صورت مستقل طراحی و پیاده سازی شده و به عنوان جعبه سیاه عمل می‌نمایند. قطعات دیگر در خارج از این قطعه نیازی به دانستن نحوه انجام کار در این سرویس را ندارند و تنها به نتیجه آن نیازمندند.
قطعات، سرویس‌های خود را از طریق رابطه‌ها (interfaces) در اختیار قطعات دیگر قرار می‌دهند که این رابطه‌ها قابل دست‌یابی و فراخوانی هستند، بدون اینکه محل قرار گیری آن‌ها اهمیت داشته باشد (رابطه‌های محلی یا دور). در ضمن این رابطه‌ها می‌توانند به همان نرم‌افزار کاربردی یا به آدرسی در محل دیگری از شبکه مرتبط باشند.
رابطه‌ها به عنوان کلیدی در برقراری این ارتباط‌ها هستند و از طریق آن‌ها نوع پارامترهای ورودی و نتایج (خروجی) مشخص می‌گردد.
به‌علاوه، با ایجاد قابلیت توزیع شدگی به شکلی مناسب و بدون وابستگی زیاد، می‌توان سرویس‌ها را در قسمت‌های مختلف شبکه و به صورت

نرم افزار چاپ برروی چک
چکاپاپ
تهیه شده جهت سازمانها ، ادارات دولتی ، شرکتها ، مؤسسات خصوصی ، بازاریان ، مغازه ها و اشخاص
- امکان تعریف انواع دسته چک های موجود و جدید بدون محدودیت
- امکان تغییر مکان مطالب و عناوین چاپی چک به صورت دلخواه
- دارای بانک اطلاعاتی همراه با امکان جستجوی سریع
- تبدیل خودکار اعداد و تاریخ به حروف
- دارای امکان پرفراژ بر روی چک
- چاپ بر روی کلیه چک های موجود
- امکان خواندن و چاپ تصویر برروی چک
- دارای امکان صدور رسید چک
- دارای کلمه عبور جهت امنیت ورود به برنامه
- بدون نیاز به CD برای اجرای برنامه
- قابل نصب برروی کلیه ویندوزها
- سازگار با کلیه پرینترها
- امکان نصب بر روی پنج کامپیوتر به صورت نامحدود
- دارای پشتیبان گیری خودکار برای حفظ اطلاعات وارد شده
- همراه با نرم‌افزار حسابدار موجود در سیناپک نسخه حرفه‌ای
موارد را که می توان بر روی چک و رسید چک چاپ نمود :
تاریخ چک به عدد
تاریخ صدور چک
تاریخ چک به حروف
مبلغ به عدد
مبلغ به حروف
پرفراژ مبلغ با اعداد فارسی یا لاتین
تبدیل مبلغ درج شده به تومان
مطالب مربوط به گیرنده چک
شماره سریال چک
نام و شعبه بانک
شماره حساب جاری
چاپ آرم و یا هر تصویر دیگر بر روی چک و یا رسید چک
چاپ دو مطلب مجزا به دلخواه (توضیحات)
چاپ تصویر در زمینه چک یا رسید آن
امکان چاپ مطالب فوق برروی رسیدهای از پیش چاپ شده
امکانات ویژه و منحصر به فرد :
بستن انتهای مبالغ عددی و حروفی با کاراکترهای دلخواه
جدا کردن سه رقم سه رقم مبلغ به عدد با کاراکتر دلخواه
جدا کردن سه رقم سه رقم مبلغ به عدد با فاصله دلخواه
همراه با امکانات سیناپک نسخه ساده به صورت رایگان
www.sinapardazeshsoft.com
نرم افزاری سیناپردازش
صندوق پستی : ۱۵۸۷۵/۹۶۶۳
واحد پشتیبانی : ۸۸۴۲۳۵۷۲(۵خط)
دفتر فروش : ۸۸۴۳۹۰۲۶(۵خط)
روابط عمومی : ۸۸۴۵۲۷۳۵
فکس : ۸۸۴۳۵۷۰۶
پذیرش نمایندگی فعال از تهران و شهرستان ها

بعضاً تکراری (مخصوصا برای سرویس‌های مهم) قرار داد تا این سرویس‌ها همیشه در دسترس بوده و در صورت زیاد شدن درخواست‌ها بتوان با استفاده از تکنیک‌های Load Balancing به تمامی آن‌ها به‌خوبی پاسخ داد.

ویژگی‌های سرویس و محاسبات سرویس‌گرا

محاسبات سرویس‌گرا (SOC)، نمونه‌ای از محاسبات است که در آن طراحی و توسعه سیستم‌های کاربردی بر پایه سرویس به عنوان عنصر اساسی، انجام می‌گیرد. سرویس‌ها عناصری هستند که مستقل از پلتفرم بوده و در ساخت سیستم‌های توزیع شده سریع و ارزان قیمت کمک می‌نمایند. همچنین سازمان‌ها را قادر می‌سازند تا توابع خود را از طریق زبان‌ها و پروتکل‌های بر پایه XML پیاده‌سازی و بر روی اینترنت یا اینترانت ارائه نمایند.
از آن‌جا که سرویس‌ها از طریق سازمان‌ها و شرکت‌های گوناگون تهیه می‌شوند و جهت دسترسی کاربران مختلف می‌بایست همواره در دسترس باشند، رعایت ویژگی‌های زیر ضروری می‌باشد:

● مستقل از فناوری باشند؛ به این معناکه به‌کارگیری و مکانیزم فراخوانی و پیدا کردن سرویس‌ها به راحتی و از تمام محیط‌ها (سیستم‌های عامل مختلف و زبان‌های برنامه‌سازی گوناگون) میسر بوده و وابسته به پلتفرم خاصی نباشد.

● وابستگی بسیار پایینی بین درخواست‌کننده و ارائه دهنده سرویس وجود داشته باشد و یا به‌عبارتی Loosly Coupled باشد. به این معناکه درخواست کننده نباید هیچ نیازی به دانستن ساختار داخلی و نحوه پیاده سازی سرویس داشته باشد. برای این منظور، فراخوانی سرویس از طریق به‌کارگیری مکانیزم پیغام (Message) بجای فراخوانی API انجام می‌گردد.

● درخواست‌کننده نباید نیازی به دانستن محل قرارگیری سرویس داشته باشد و به‌عبارتی معماری سرویس‌گرا می‌بایست Location Transparency را پشتیبانی نماید. به این ترتیب که محل قرارگیری سرویس و مشخصات آن در مخزنی (Repository) قرار می‌گیرد و درخواست‌کننده، محل و اطلاعات لازم را از طریق بازیابی آن از این مخزن بدست می‌آورد.

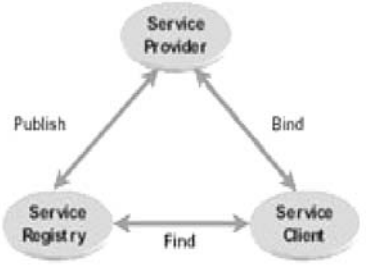
سرویس‌ها می‌توانند به دو شکل ساده و ترکیبی ارائه شوند. سرویس‌های ترکیبی، سرویس‌هایی هستند که بر اساس به‌کارگیری چند سرویس ساده (یا ترکیبی) ایجاد می‌شوند. برای مثال، ممکن است سیستم توزیع شده‌ای بر اساس چند سرویس ساده شبیه به صدور صورتحساب، ثبت سفارش، مدیریت روابط مشتری و... سرویس‌های ترکیبی گسترده‌تری در ارتباط با حرفه‌ای خاص ایجاد نماید.
سیستم‌های ساخته شده بر اساس سرویس، ترکیبی از سرویس‌های مستقل هستند که عملکردهای خود را از طریق رابطه‌های تعریف شده‌ای در اختیار کاربران (بالقوه) خود قرار می‌دهند. Web Service Description Language (WSDL) از جمله راه‌هایی است که به‌طور گسترده برای تعریف این رابطه‌ها به‌کار می‌رود تا به‌وسیله آن جزئیات لازم برای اتصال درخواست کننده به ارائه دهنده سرویس تعریف شود.

نرم‌افزار به‌عنوان سرویس

اصل ارائه شده "نرم‌افزار ـ به‌عنوان ـ سرویس" از محاسبات سرویس‌گرا، بر اساس مدل ASP مطرح گشته است. ASP هویت سوم شخصی است که به‌کارگیری سرویس‌های نرم‌افزاری و دسترسی مشتری را به بسته نرم‌افزاری از طریق شبکه، مدیریت و میزبانی می‌نماید. به‌عبارتی ASPها راهی برای رفع نیازهای IT شرکت‌ها از طریق واگذاری بخشی از این نیازها یا تمامی آن‌ها به بیرون از سازمان می‌باشند.
برای این منظور ASP با استفاده از زیرساخت‌های خود، ارتباط بین مشتری و نرم‌افزار ارائه شده را برقرار کرده و دسترسی وی به داده‌ها و توابع موجود را بصورت در دسترس، مدیریت می‌نماید.
اگرچه نظریه "نرم‌افزار به‌عنوان سرویس" اولین بار توسط ASP ارائه شد، اما مشکلات این روش باعث ایجاد کدهایی می‌شد که معمولا قابل استفاده مجدد نبوده و محدود به مشتری خاص می‌بود، به‌عبارتی وابستگی زیادی بین سرویس ارائه شده و سیستم استفاده‌کننده به‌وجود می‌آمد.
معماری سرویس‌گرا اجازه می‌دهد تا نظریه "نرم‌افزار بعنوان سرویس"، گسترش یافته تا از طریق آن بتوان پردازش‌ها و تراکنش‌های پیچیده را به‌عنوان سرویس‌هایی با قابلیت استفاده مجدد ارائه کرد و به این ترتیب سیستم‌ها را مستقل از سرویس‌ها طراحی و تولید نمود.

معماری سرویس‌گرای مقدماتی

شیوه‌ای منطقی برای طراحی سیستم‌های نرم‌افزاری است که از طریق آن سرویس‌هایی جهت ارائه به‌کاربران یا سایر سرویس‌های توزیع شده بر روی شبکه ایجاد می‌شود و این سرویس‌ها از طریق رابطه‌های تعریف شده در دسترس می‌باشند.
معماری سرویس‌گرای مقدماتی، راهی برای تبادل اطلاعات بین عامل‌های نرم‌افزاری به‌وسیله پیغام تعریف می‌نماید. این عامل‌ها، درخواست‌کننده سرویس (مشتری) و یا تهیه‌کننده سرویس می‌باشند. علاوه بر این دو، عامل دیگری به‌عنوان عامل کشف سرویس نیز وجود دارد. در معماری سرویس‌گرا، معرفی سرویس‌ها و همچنین نحوه ارتباط این سه شرکت کننده نیز اهمیت دارد. این ارتباطات عبارتند از: منتشر کردن سرویس، پیدا کردن سرویس و متصل شدن به سرویس.



شکل ۱

در یک سناریو بر پایه سرویس، تهیه‌کننده، سرویس را پیاده‌سازی کرده و از طریق شبکه به ارائه توضیحات آن سرویس برای درخواست‌کننده یا عامل کشف سرویس می‌پردازد. درخواست‌کننده معمولا درخواست پیدا کردن سرویس را به عامل کشف سرویس می‌دهد تا از طریق آن به توضیحات ارائه

شده سرویس و محل آن دسترسی پیدا کند. سپس با به‌کارگیری این اطلاعات به تهیه‌کننده سرویس متصل شده و از سرویس ارائه شده استفاده می‌نماید. بدین ترتیب، سرویس به‌عنوان قطعه نرم‌افزاری پیاده‌سازی شده‌ای که توسط یک رابط تعریف شده مستند گردیده است و نه تنها به‌وسیله خود تهیه‌کننده بلکه توسط سایر عواملی که از نحوه پیاده سازی آن خبر ندارند نیز قابل استفاده و فراخوانی است.

این ویژگی جعبه سیاه بودن سرویس از اصل ماژولاریتی در مهندسی نرم‌افزار به ارث رسیده است. البته سرویس با تعاریفی مانند ماژول، قطعه نرم‌افزاری یا شی تفاوت دارد، زیرا نه تنها در سطح برنامه‌ها و نرم‌افزارهای کاربردی، بلکه در سطح حرفه و حتی مابین سازمان‌ها نیز قابل به‌کارگیری و استفاده مجدد می‌باشد.

در واقع سرویس‌ها، نمایش عملکرد معنی داری از حرفه هستند که می‌توانند بنا به نیاز مشتری در سرویس‌ها و توابع بزرگ‌تر یا جدید به‌کار گرفته شوند. رابطه‌ها به سادگی، مکانیزمی جهت برقراری ارتباط بین سرویس و نرم‌افزارها یا سایر سرویس‌ها ایجاد می‌نمایند. از لحاظ تکنیکی، رابط سرویس‌ها، توضیحاتی در مورد نام و امضاء متدهای یک سرویس هستند که توسط درخواست‌کننده، قابل فراخوانی می‌باشند.

در شماره بعد به موضوع "معماری سرویس‌گرای توسعه یافته" خواهیم پرداخت.

فرم ساز
نرم افزاری سیناپردازش
نرم افزار ایجاد فرم های اداری و تجاری
دارای امکان پرکردن فرم های از پیش چاپ شده
☑ محیط کاملا فارسی
☑ دارای امکان پرکردن فرم های از پیش چاپ شده عمومی و خصوصی
☑ امکان ذخیره سازی فایل های پرشده به صورت مجزا و یا در صفحات متعدد در یک فایل
☑ عدم محدودیت در ایجاد پرونده‌ها
☑ قابل اجرا در ویندوزهای ۲۰۰۰ ـ ایکس پی و ۲۰۰۳
☑ ورود اطلاعات داخل فرم ها به سادگی و در کمترین زمان ممکن در سه حالت متفاوت و قابل انتخاب توسط کاربر
☑ سازگاری با کلیه چاپگرهای قابل تعریف در محیط ویندوز
☑ دارای فرم های از پیش طراحی شده مانند فرم ثبت سفارش بازرگانی ـ صورتحساب فروش کالا ـ سند حسابداری ـ انواع چک و رسید آن و ...
☑ طراحی و پیاده سازی فرم ها به راحتی و به توسط درگ کردن کادرها به وسیله ماوس
☑ امکان تغییر طول ، عرض ، فاصله از راست ، فاصله از بالا ، ضخامت ، رنگ زمینه ، رنگ ضخامت کادر دور و ... هر کادر ایجاد شده
☑ امکان قفل کردن تک تک یا تمام اجزاء فرم ها
☑ دارای فرمت های مختلف برای ورود متن ، تاریخ ، ساعت ، علامت تایید (چک مارک) اعداد و تصویر
☑ امکان تبدیل اعداد مندرج در متون به حروف همراه با امکان اضافه نمودن هر کلمه یا متن به انتها یا ابتدای اعداد تبدیل شده
☑ امکان تغییر ترتیب ورود اطلاعات به دلخواه
☑ دارای رمز ورود به برنامه با در نظر گرفتن محدودیت هایی برای اپراتور در هنگام اجرا
☑ امکان تعریف کادر تابع (کادر تابع عبارت است از کادرهایی که متن داخل آن ها تابع متن یک کادر دیگر بوده و درهنگام ورود اطلاعات متن خود را از کادر مادر دریافت می‌نماید. کاربرد این ویژگی در فرم‌های دو یا چند تکه یا چند صفحه‌ای می‌باشد.)
☑ امکان استفاده از تصویر زمینه برای راحتی طراحی فرم‌ها
☑ امکان درج نوشته‌های عمودی (از بالا به پایین و از پایین به بالا)
☑ امکان درج صفحات متعدد در یک فایل ایجاد شده
☑ امکان ورود کادر محاسبه جهت محاسبه کادرهای عددی
☑ دارای شمارنده عددی برای کادرهای عدد با امکان تغییر اعداد اضافه شده
تهیه شده جهت سازمانها ،
ادارات دولتی ،
شرکتها ،
و مؤسسات خصوصی
نرم افزاری سیناپردازش
صندوق پستی : ۱۵۸۷۵/۹۶۶۳
واحد پشتیبانی : ۸۸۴۲۳۵۷۲(۵خط)
دفتر فروش : ۸۸۴۳۹۰۲۶(۵خط)
روابط عمومی : ۸۸۴۵۲۷۳۵
فکس : ۸۸۴۳۵۷۰۶
www.sinapardazeshsoft.com