مفاهیم برنامه نویسی

۲-۱- پیاده سازی نرم افزار

Program = Code + Data

نرم افزار، یک برنامه کامپیوتری شامل مجموعه ای از دستورالعمل ها بوده که با هد فمندی خاصی در کنار یکدیگر قرار گرفته تا از این طریق نماینگر خواسته های مورد نظر باشند. نرم افزار مجری خواسته های مورد نظر در سرزمین سخت افزار بوده و با در اختیار گرفتن مجموعه ای از منابع سخت افزاری سعی در تحقق اهداف از قبل تعریف شده دارد.

بخش های تشکیل دهنده یک برنامه : یک برنامه کامپیوتری از دو بخش داده و منطق تشکیل می گردد. منطق تعریف شده در یک برنامه با بکارگیری داده ها و یا تولید داده ها، اهداف از قبل تعریف شده را دنبال خواهد کرد. بمنظور پیاده سازی منطق یک برنامه از مجموعه ای کدها که توسط یک زبان برنامه نویسی نوشته می گردند، استفاده خواهد شد. کدها (دستورالعمل ها) با یک نظم و انظباط خاص (با توجه به منطق تعریف شده و موجود) اجراء خواهند شد.برای نوشتن دستورالعمل ها از زبانهای برنامه نویسی استفاده می گردد. پس از نگارش دستورالعمل ها (بکمک زبان برنامه نویسی استفاده شده) و ترجمه دستورالعمل ها به زبان ماشین ، امکان اجراء و بهره مندی از منابع سخت افزاری ، فراهم خواهد شد(با توجه به امکانات فراهم شده توسط سطستم عامل). برای ترجمه دستورالعمل ها از دو رویکرد ترجمه و تفسیر استفاده می گردد. مترجم ها خود نرم افزارهائی می باشند که برنامه دیگری را بعنوان ورودی در اختیار گرفته و ضمن انجام عملیات و پردازش های لازم ، کدی را تولید خواهند کرد که قابل اجراء بر روی سخت افزار استفاده شده است . عملکرد مترجم ها و مفسرها با توجه به موضوع گفته شده و از این زاویه یکسان بوده و تنها تفاوت موجود در ماهیت و نوع انجام عملیات ترجمه است. یکی از مهمترین تفاوت های موجود بین مترجم ها و مفسرها سرعت است . مسلما" سرعت اجرای برنامه های ترجمه شده توسط مترجم ها بمراتب بالاتر از مفسرها است .

داده ها، در یک برنامه کامپیوتری دارای نقشی حیاتی و بسیار مهم می باشند. در این راستا برخی از برنامه های کامپیوتری ، داده هائی را از ورودی گرفته و ضمن انجام پردازش های لازم ، نتایج مورد نظر را بدست آمده و در صورت لزوم در دستگاههای خروجی منعکس (ذخیره) خواهند کرد.برخی دیگر از برنامه های کامپیوتری بدون اخذ اطلاعاتی خاص از ورودی ، با توجه به منطق خود و طی نمودن پردازش های لازم اقدام به تولید و ارائه خروجی مورد نظر خواهند نمود. بهرحال حیات یک برنامه کامپیوتری در ارتباط مستقیم با داده ها خواهد بود. داده ها می توانند بصورت ساده و یا پیچیده مورد توجه قرار گیرند. در بطن داده های پیچیده نوعی ارتباط و همگرائی وجود دارد . بمنظور تامین و ذخیره سازی داده ها (مرجع ورود و خروج داده ها) از منابع داده ئی استفاده می گردد. منابع داده ئی در مرحله اول ساختارهای لازم بمنظور ذخیره سازی داده ها را فراهم نموده و در ادامه با توجه به زیر ساخت ایجاد شده ، امکان مدیریت داده ها نیز فراهم خواهد شد.

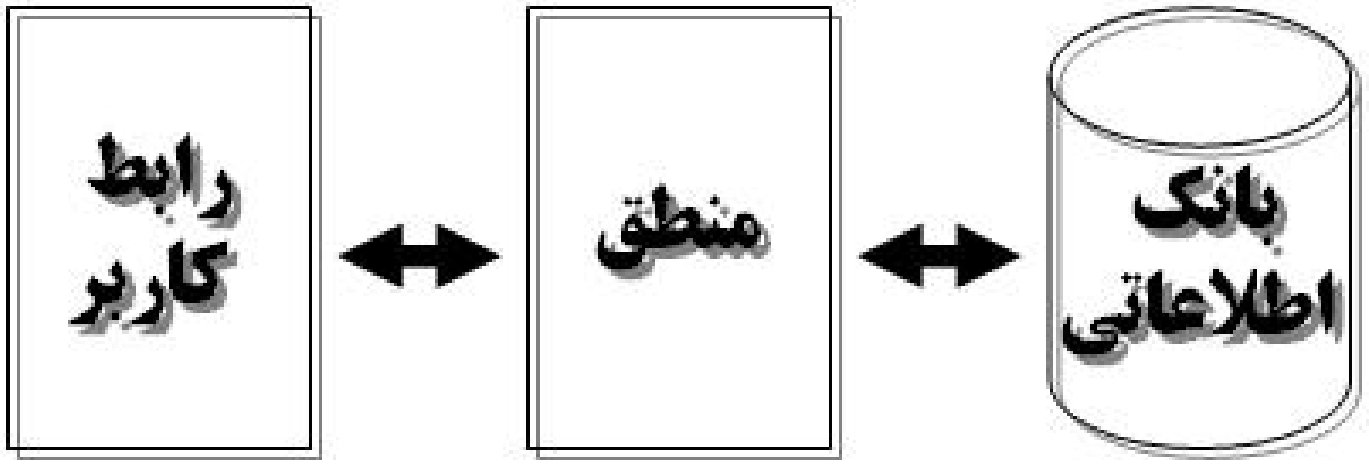
۲-۲- نیازها و انتظارات

بمنظورطراحی و پیاده سازی نرم افزار به مجموعه ای از ملزومات نیاز خواهد بود.

۱- روشی بمنظور ایجاد ارتباط با کاربر. نرم افزار محصولی است که توسط انسان طراحی و توسط گروهی دیگر مورد استفاده قرار خواهد گرفت. بدیهی است که شرط اول در بکارگیری یک نرم افزار، ایجاد ارتباط با آن است . مادامیکه کاربران قادر به ایجاد یک ارتباط مناسب با یک نرم افزار نباشند، زمینه استفاده از پتانسیل های موجود فراهم نخواهد گردید. بد ین منظور تمامی تولیدکنندگان و عرضه کنندگان نرم افزار اهتمام جدی خود را در زمینه طراحی و پیاده سازی این بخش از نرم افزارها که " رابط کاربر " نامیده می شود ، بعمل می آورند. بخش " رابط کاربر" هر نرم افزار یکی از مهمترین عوامل در موفقیت یک محصول نرم افزاری از دیدگاه کاربران است . کاربران با استفاده از بخش فوق قادر به ایجاد ارتباط و انتقال خواسته خود بوده تا از این طریق زمینه تحقق خواسته مورد نظر در کوتاهترین و سریعترین زمان ممکن ، فراهم گردد. در این زمینه اگر از کاربری خواسته شود که واژه " کیفیت " در رابطه با یک نرم افزار را تعریف نماید ، میدان و محدوده پاسخگوئی به سوال فوق عموما" به امکانات موجود در بخش " رابط کاربر" اشاره خواهد داشت . در صورتیکه کاربر مورد نظر توانسته باشد از طریق بخش " رابط کاربر"، یک ارتباط صحیح و مناسب را بمنظور تامین خواسته های خود برقرار کرده باشد ، قطعا" در مواجهه با سوال فوق ، پاسخی مثبت را ارائه خواهد داد.

۲ - روشی بمنظور تامین اهداف . در بطن واژه نرم افزار همواره یک هد فمندی خاص دنبال می گردد . بر همین اساس است که نرم افزار را می توان بعنوان یک موجودیت نمو(رشد) کننده نیز در نظر گرفت . برای دستیابی به اهداف مورد نظر می بایست پس از شناسائی و تبین اهداف، اقدام به ارائه راهکارهای لازم بمنظور نیل به آنها نمود. هر نرم افزار برای تامین اهداف خود از مجموعه ای سیاست ها و قوانین استفاده می نماید. بمنظور تحقق اهداف و بالفعل نمودن سیاست ها ی مورد نظر در یک نرم افزار ، کدهای لازم نوشته خواهند شد. کدهای فوق مجری سیاست ها و رویکردهای یک نرم افزار بمنظور نیل به اهداف خواهند بود.

۳ - روشی بمنظور مدیریت داده ها . داده ها در یک برنامه کامپیوتری دارای جایگاه خاصی می باشند. بمنظور تولید فرآورده های اطلاعاتی می بایست در مرحله اول داده های خام به یک نرم افزار تغذیه و در مرحله بعد و پس از پردازش های لازم فرآورده های مورد نظر تولید گردند. بمنظور مدیریت داده ها در یک برنامه در ابتدا می بایست با ایجاد ساختارهای لازم اطلاعات مورد نظر ذخیره و در ادامه و با توجه به فرضیات موجود از داده های ذخیره شده ، استفاده عملیاتی بعمل خواهد آمد. چرخه استفاده ، تولید و ذخیره سازی اطلاعات مستلزم اعمال یک مدیریت قابل قبول در رابطه با داده ها خواهد بود. زمانیکه داده ها از حالت ساده خارج و به اشکال پیچیده تری جلوه می نمایند ، بانک های اطلاعاتی مطرح تا از این طریق پاسخی شایسته به این نوع از نیازها باشند. بانک های اطلاعاتی با ارائه ساختارهای لازم (جداول) و ایجاد ارتباط بین داده های مورد نظر، زمینه مدیریت داده ها را بخوبی فراهم می آورند.



شکل ۲-۲-۱

۲-۳- چالش های موجود

یک نرم افزار از مجموعه ای برنامه های جانبی و روتین ها تشکیل می گردد که بر اساس یک ساختار ماژولار با یکدیگر مرتبط خواهند شد. نرم افزار نظیر یک جدول (Puzzel) بوده که که ازعناصر متفاوتی تشکیل و با در کنار هم قرار گرفتن آنها یک شکل صحیح که همانا هدف تعریف شده است ، نمایان خواهد شد. عناصرموجود در جدول(ساختار) فوق، بدرستی تصویر مناسبی از نرم افزار را در اختیار علاقه مندان (کاربران) قرار خواهند داد. با توجه به ماهیت عملیات فوق و نقش عناصر تشکیل دهنده با مجموعه ای از رویکردها و چالش ها برخورد خواهیم نمود.

۱ - محل نصب عناصر ؟ آیا بمنظور مشاهده تصویر مناسبی از یک نرم افزار، می بایست تمامی عناصر ذیربط، بصورت فیزیکی در مجاورت هم قرار گرفته و یا امکان مجاورت منطقی عناصر نیز وجود دارد؟

۲ - محل اجرای عناصر ؟ بمنظوراجرای یک نرم افزار که از عناصر متفاوتی تشکیل شده است ،لازم است که تمامی عناصر مورد نظر در یک محل اجراء گردنند ؟ آیا بمنظور ارائه خدمات یک نرم افزار می بایست تمامی عناصر زیر یک سقف فیزیکی (سخت افزار خاص) اجراء گردنند؟ آیا می توان بخش های متفاوتی از یک نرم افزار را در محیط دیگر اجراء و یک هدفمندی و همیاری در اجراء را برای آنان تعریف نمود؟

۳ - نحوه ارتباط عناصر ؟ با توجه به اینکه یک نرم افزار می تواند از عناصر و بخش های متفاوتی تشکیل گردد ، نحوه ارتباط این بخش ها با یکدیگر به چه صورت خواهد بود ؟ زمانیکه بخش های فوق بر روی یک سیستم نصب و اجراء می گردنند ، نحوه ارتباط به چه صورت است ؟ در صورتیکه بخش های مورد نظر بر روی سیستم های متفاوت نصب و اجراء می گردنند ، نحوه ارتباط آنها با یکدیگر به چه صورت خواهد بود؟

سوالات فوق صرفا" مجموعه ای محدود از چالش های موجود در این زمینه است . بمنظور ارائه پاسخ مناسب و علمی دراین خصوص ، " معماری نرم افزار " مطرح می گردد. بر این اساس، معماری های متفاوتی با توجه به تنوع خواسته ها و مرور زمان ، بوجود آمده است . بخاطر داشته باشیم که برنامه های تحت وب نیز متکی بر یک نوع معماری خاص می باشند.

۲-۴- انواع معماری تولید نرم افزار

از بدو مطرح شدن نرم افزار تاکنون ، معماری های متفاوتی بمنظور طراحی و پیاده سازی ارائه شده است . معماری های فوق از یکطرف برخاسته از امکانات و ماهیت سخت افزار ها در زمان خود و از طرف دیگر نمایانگر نوع و نگرش انتظارات طرح شده توسط کاربران است . بخاطر داشته باشیم که نرم افزار دارای ماهیتی پویا بوده و در هر زمان می بایست خود را با خیل عظیم نیازها و انتظارات جدید کاربران تطبیق نماید. چراکه نرم افزار عصاره خواسته های انسانی بمنظور بالفعل شدن بر روی بستر سخت افزار در گذر زمان است . بدیهی است از گذشته تاکنون، هم طیف خواسته های انسانی تغییر کرده و خواهد کرد و هم سخت افزارها دچار تغییر و تحول گسترده ای بوده و خواهند بود. در این راستا لازم است نرم افزار نیز با رعایت کامل اصل انعطاف پذیزی ، پذیرای تمامی تحولات از گذشته تاکنون بوده و بتواند در هر زمان رسالت خود را بخوبی انجام دهد. بر همین اساس از گذشته تاکنون معماری های متفاوتی بمنظور طراحی و پیاده سازی نرم افزار ارائه شده است . هر معماری دارای شاخص ها و ویژگی های منحصر بفرد خود بوده و نرم افزارهائی که با اتکاء بر هر یک از معماری های فوق پیاده سازی می گردنند ، خصایص خود را از معماری بکارگرفته شده به ارث خواهند برد. در این بخش به رفتارشناسی هر یک از معمارهای ذیل پرداخته تا از این طریق زمینه های لازم بمنظور شناخت معماری بکارگرفته شده در برنامه های تحت وب فراهم گردد.

معماری MainFrame

معماری File Server

معماری سرویس گیرنده / سرویس دهنده

معماری Two-Tier

معماری Three-Tier

۲-۴-۱- معماری MainFrame

ویژگی :

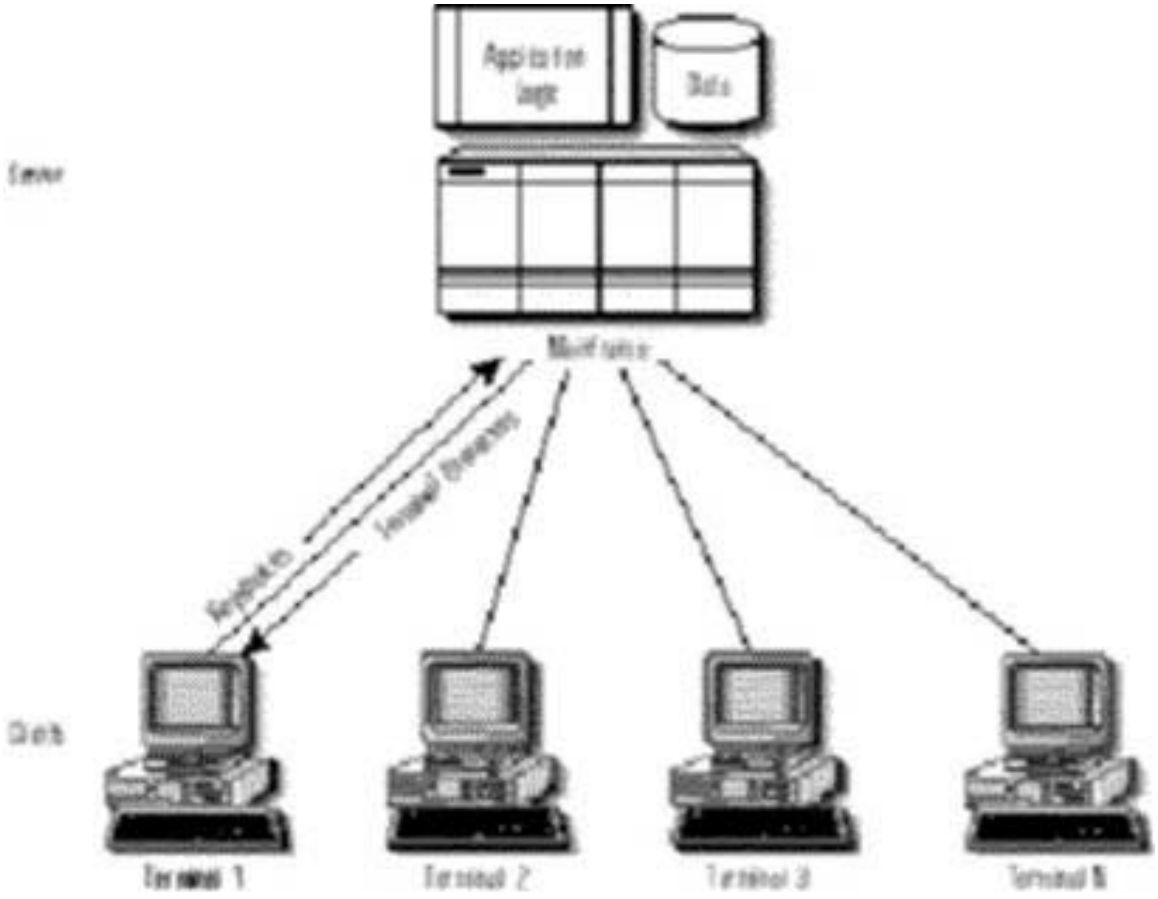
- معماری فوق در دهه های ۱۹۶۰ الی ۱۹۷۰ مورد توجه و استفاده جدی قرار داشت .
- کامپیوتر اصلی (Host) مسئولیت انجام تمامی پردازش ها را برعهده دارد.
- کاربران با استفاده از ترمینال ها ، قادر به ایجاد ارتباط با سیستم اصلی (host) می باشند.
- ترمینال ها هوشمند نبوده و صرفا" به یک صفحه کلید و نمایشگر محدود می باشند.
- فشردن کلیدهای صفحه کلید ، تنها چیزی است که ارتباط بین کاربران(ترمینال ها) و سیستم اصلی را معنی خواهد کرد.
- داده ها و منطق برنامه بر روی یک سیستم (Host) یکسان ذخیره می گردنند. .

مزایا :

- امنیت در این نوع معماری بسیار بالا است .
- با توجه به تمرکز داده ها و منطق ، مدیریت متمرکز و اعمال آن آسان خواهد بود.

معایب :

- هزینه تهیه ، اجاره و پشتیبانی این نوع سیستمها بسیار بالا است .
- برنامه (منطق) بهمراه داده های مربوطه در یک محل مستقر و از یک محیط پردازش یکسان استفاده می کنند.
- اغلب برنامه های نوشته شده بر اساس معماری فوق محیط های رابط کاربر گرافیکی را حمایت نمی نمایند



شکل ۲-۴-۱

۲-۴-۲-معماری File server

- ویژگی :

- چرخش ۱۸۰ درجه ای نسبت به معماری

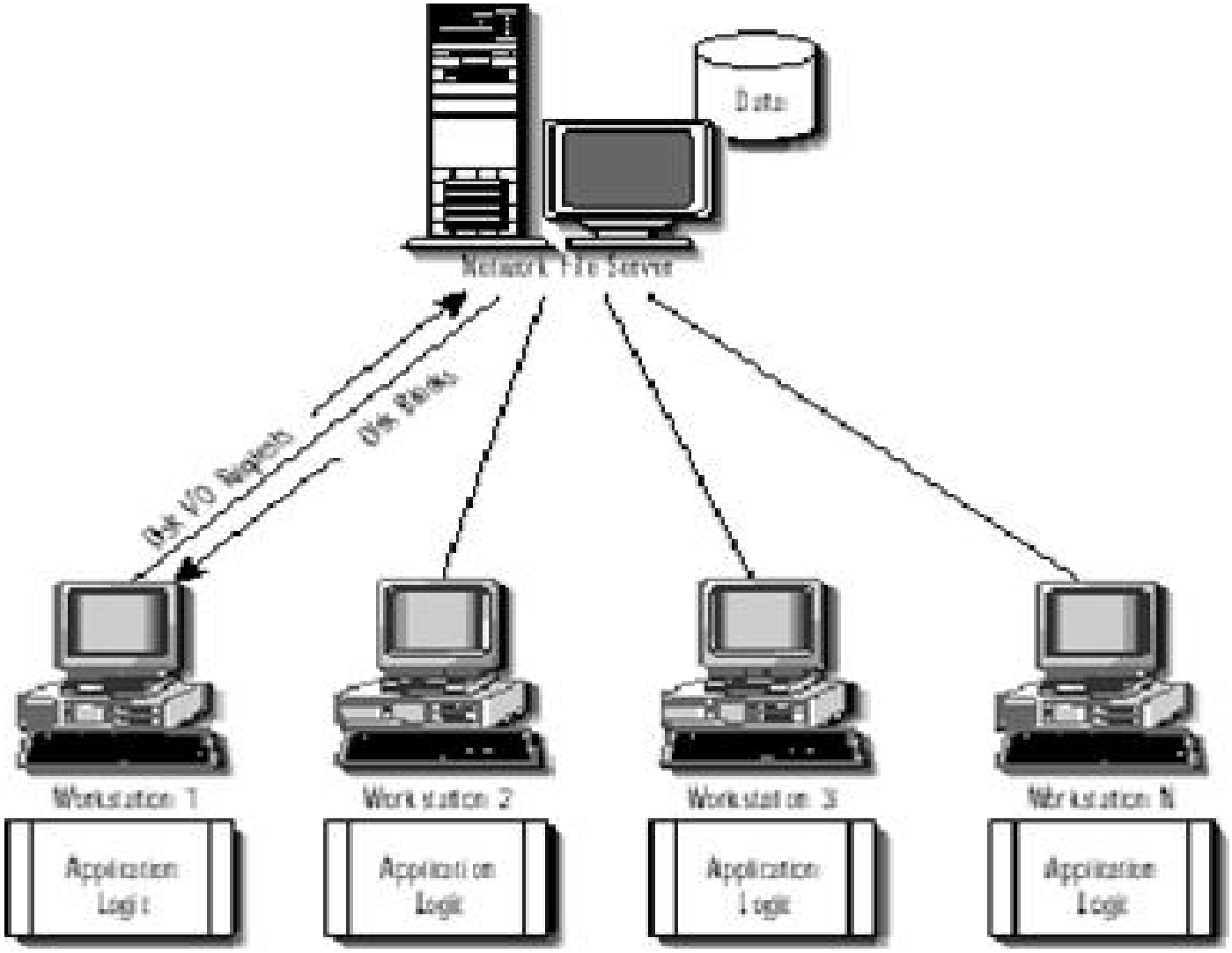
- از سرویس دهنده یا سرویس دهند گان متعدد، بصورت متمرکز استفاده می گردد.
- منابع متفاوتی نظیر چاپگر و یا فضای ذخیره سازی (هارد) به اشتراک گذاشته می شوند.
- سرویس دهنده فایل های مورد نیازو ذخیره شده توسط منابع اشتراکی را برای کاربران ارسال خواهد کرد.
- کار درخواست شده توسط کاربر (منطق و داده) بر روی سیستم کاربر اجراء خواهد شد.
- منطق برنامه برروی سیستم کاربر (سرویس گیرنده) اجراء خواهد کردید.
- داده ها بر روی سرویس گیرنده مستقر خواهند شد.

- مزایا :

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

- معایب :

- تمامی منطق برنامه بر روی سرویس گیرنده اجراء خواهد شد .
- وجود محدودیت های خاص نظیر میزان حافظه و یا نوع پردازشگر سرویس گیرندگان، بکارگیری برنامه را با مشکل مواجه می سازد.
- بهبود عملکرد برنامه و یا اعمال اصلاحات مورد نظر همواره یکی از چالش های جدی است .


Data
Network File Server
Disk I/O Requests
Disk Blocks
Workstation 1
Workstation 2
Workstation 3
Workstation N
Application Logic
Application Logic
Application Logic
Application Logic

شکل ۲-۴-۲

ویژگی :

- در معماری فوق از سرویس دهند گان و سرویس گیرند گان با خصایص متفاوت استفاده می شود.
- اصل تقسیم کار دنبال 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 می نمایند.
- سرویس گیرنده با ارسال درخواست و سرویس دهنده با پاسخ به درخواست جلوه ای از همیاری در پردازش عملیات را بنمایش می گذارند.
- پلات فورم و سیستم های عامل سرویس دهنده و سرویس گیرنده می تواند متفاوت باشد.
- عملیاتی را که یک برنامه انجام می دهد بین سرویس دهنده و سرویس گیرنده تقسیم می گردد.

مزایا :

- بهره گیری مناسب از پتانسیل های سخت افزاری موجود با توجه به اصل تقسیم عملیات ها
- بهینه سازی استفاده و بکارگیری منابع اشتراکی .
- بهینه سازی توانائی کاربران از بعد انجام فعالیت های متفاوت

معایب :

- عدم وجود امکانات لازم برای کپسوله نمودن سیاست های راهبردی نرم افزار
- کاهش کارائی برنامه همزمان با افزایش تعداد کاربران همزمان
- بهبود عملکرد برنامه و یا اعمال اصلاحات مورد نظر همواره یکی از چالش های جدی است .

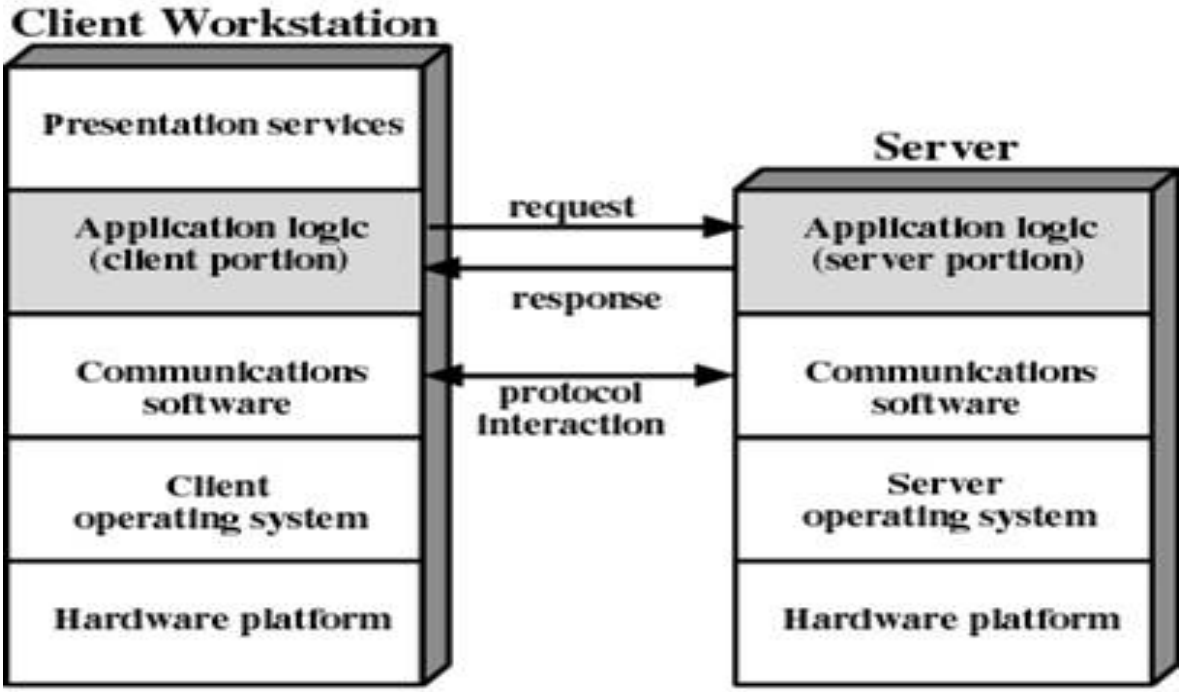


Figure 13.2 Generic Client/Server Architecture

شکل ۲-۴-۳

تاکنون مدل های متفاوتی از معماری فوق پیاده سازی شده است .

۲-۴-۳-۱- پردازش های مبتنی بر میزبان

مدل فوق بمنزله یک مدل سرویس دهنده / سرویس گیرنده تلقی نشده و مشابه مدل MainFarme است .