مقدمه

چرا یک پروژه کمبود بودجه پیدا می کند و به موقع تمام نمی شود؟ برای برآورد هزینه یک پروژه نرم افزاری چه چیزهایی لازم است؟ چگونه اندازه کار لازم برای توسعه یک نرم افزار اندازه گیری می شود؟

مدیران پروژه های نرم افزاری چطور می توانند از مدلهای برآورد هزینه استفاده کنند و نتایجی که از این مدلها بدست می آیند چقدر قابل اطمینان است؟

امروزه ساخت سیستم های نرم افزاری کاری دشوار و پر هزینه است. جواب سوالات بالا برای برنامه ریزی و مدیریت پروژه های نرم افزاری اهمیت زیادی دارد. علم مهندسی نرم افزار راههایی برای اندازه گیری کمی یک پروژه را ارائه می کند. مدیر پروژه معیارهای نرم افزار را می سازدو فرآیند های نرم افزار را براساس این معیارها اندازه می گیرد. بدست آوردن اندازه پروژه اولین قدم در انجام برآورد نرم افزار است. مدیر پروژه بر اساس اطلاعاتی که از سازمان،پرسنل و پروژه های تکمیل شده قبلی دارد و همینطور بر اساس تجربیات شخصی اش برآود هزینه نرم افزار را انجام می دهد.

بخش اول پایان نامه تعریف مهندسی نرم افزار و مدیریت نرم افزار و معیار های نرم افزاری را شرح می دهد. بخش دوم درباره برآورد هزینه نرم افزار تکنیک ها و مدلهای برآورد هزینه نرم افزار است، اینکه چرا یک برآورد باید انجام شود، چگونه و کی انجام می شودو چه عواملی در انجام یک برآورد مفید و درست نقش دارند. اندازه یک سیستم نرم افزای چگونه محاسبه می شود، مدل برآورد هزینه نرم افزار چیست و مدل برآورد الگوریتمی توضیح داده می شود و مدل کوکومو به تفصیل بررسی می شود. در بخش چهارم سیستم برآورد بر خط، محیط ویژوال استودیو دات نت قسمتهای مختلف سیستم و چگونگی پیاده سازی سیستم شرح داده می شود. چگونگی استفاده از نرم افزار برای انجام برآورد در بخش پنجم توضیح داده می شود. در بخش ششم نتایج حاصل از انجام برآورد برای یک سیستم فرضی بوسیله چند مدل مختلف مقایسه می شود و سرانجام برآورد برای یک سیستم واقعی تکمیل شده انجام می شود.

فصل اول –

تعریف مهندسی نرم افزار

۱-۱ نگاهی کلی به مهندسی نرم افزار

هدف مهندسی نرم افزار توسعه سیستمهای نرم افزاری، مطابق با نیازهای کاربران است. بصورت زمانبندی شده و با بودجه معین.

یک نرم افزار را می توان یک محصول مهندسی مثل یک هواپیما یاتومبیل یا تلوزیون در نظر گرفت یا هر محصولی که احتیاج به درجه بالایی از مهارت برای تبدیل شدن به یک محصول مفید دارد.برخی از مشخصه های یک نرم افزار به این ترتیبند: یک نرم افزار

- یک موجودیت است نه یک سند
- به طور کلی قسمتی از یک سیستم نرم افزاری یا سخت افزاری بزرگتر است
- جایگزین مولفه های مهندسی سخت افزاری گذشته است
- اغلب یک واسط کاربری از یک نرم افزار یا یک سخت افزار دیگر دارد
- قبل از استفاده باید تست شود
- خیلی بزرگتر و پیچیده تر از آن است که بدون نقشه ساخته شود
- ساخت آن هزینه زیادی دارد

عنوان مهندسی نرم افزار برای نخستین بار در یک گردهمایی در آلمان بکاررفت و ابتدا به عنوان یک تکنیک و سپس به عنوان یک شغل پذیرفته شد. هدف مهندسی نرم افزار ایجاد اصولی مهندسی برای توسعه نرم افزار است و برای این بوجود آمده که برخی مسائل را مانند تحویل دیرهنگام، هزینه اضافی پیدا کردن و نرسیدن به اهداف اولیه نرم افزار را کاهش دهد.

مهندسی نرم افزار کاربرد یک روش سیستماتیک، علمی و کمیت پذیر در بسط، راه اندازی و نگهداری نرم افزار است، یعنی استفاده از مهندسی در توسعه نرم افزار. [۱]

تعاریف متعددی برای مهندسی نرم افزار وجود دارد. وجه اشتراک همه آنها عبارت است از:مهندسی نرم افزار از قواعد مهندسی در تولید نرم افزار ها استفاده می کند و شامل تکنیکهایی است که باعث می شود نرم افزار ها بتوانند بصورت گروهی تولید شوند. مهندسین نرم افزار علاوه بر دارا بودن اطلاعات کافی در خصوص تکنیکهای محاسباتی، باید بتوانند از طریق مذاکره یا مکاتبات با دیگران ارتباط برقرار کنند. باید به اهمیت مدیریت پروژه آگاه بوده و مسائلی را که کاربران سیستم در عمل با آن مواجه می شوند درک نمایند. منظور از نرم افزار، فقط برنامه های کاربردی مربوط به یک کاربرد نیست. علاوه بر برنامه ها، نرم افزار حاوی مستنداتی است که برای نصب، به کارگیری و نگهداری آن لازم است. برای سیستمهای بزرگ اهمیت تهیه مستندات به اندازه اهمیت خود برنامه هاست. تجربه جدید در ساختن سیستمهای نرم افزاری بزرگ نشان داد که متد های فعلی تولید نرم افزار، کافی نیستند. پروژه های بزرگ گاهی چندین سال طول می کشیدند، هزینه های آنها بیش از مقدار پیش بینی شده بودند،

نگهداری آنها مشکل بوده و به سختی اجرا می شدند. وضعیت توسعه نرم افزار بسیار بحرانی بود. هزینه های سخت افزار کاهش می یافت، در حالیکه هزینه نرم افزار به سرعت در حال افزایش بود. تکنیکها و متدهای جدید، برای کنترل پیچیدگی نرم افزار ضروری بودند.

مهندسی نرم افزار مانند سایر مهندسی ها فقط به تولید نمی اندیشد بلکه به تولیدی با هزینه موثر فکر می کند. رقابت مهندسین نرم افزار در تولید نرم افزار کیفی با تعداد خطاهای اندک در یک زمانبندی پیش بینی شده است.

در کتاب مهندسی نرم افزار سامرویل کیفیت نرم افزار بر اساس چهار صفت تعیین شده که باید در هر نرم افزار مبتنی بر علم نرم افزار وجود داشته باشد برای یک نرم افزار که کارایی لازم را دارد این چهار صفت به این ترتیبند:

- **نرم افزار باید قابل نگهداری باشد**. چون باید تغییراتی در نرم افزارهایی با طول عمر زیاد ایجاد گردد، برای کم کردن هزینه این تغییرات، باید مستنداتی نوشته شود.
- **نرم افزار باید قابل اعتماد باشد**. یعنی انتظار کاربر را برآورده کرده و در اغلب موارد با شکست مواجه نشود.
- **نرم افزار باید کارامد باشد**. این ضرورتا بدین معنی نیست که از همه امکانات سخت افزار استفاده گردد؛ بیشینه کردن کارایی نرم افزار ممکن است اعمال تغییرات در آن را مشکل نماید. منظور از کارایی این است که نرم افزار نباید منابع سیستم مانند حافظه و سیکلهای پردازشگر را به هدر دهد.
- **ارتباط کاربر با نرم افزار باید ساده باشد**. قابلیتهای زیادی از نرم افزارها به دلیل مشکل بودن ارتباط کاربر بانرم افزار قابل استفاده نیستند.طراحی واسط کاربر باید متناسب با تواناییهای کاربران باشد.بهینه سازی همه صفات در یک سیستم مشکل است(به عنوان مثال، ایجاد ارتباط بهتر با کاربر ممکن است باعث کاهش کارایی سیستم شود) . ارتباط بین هزینه و اعمال بهبود ها در هر کدام از این صفتها خطی نیست و بهبود کوچکی در هر کدام از این صفات ممکن است گران تمام شود.

مهندسی نرم افزار یک فناوری لایه ای است . مهندسی نرم افزار مانند چسبی عمل می کند که لایه های فناوری را به هم می چسباند و به توسعه درست و به موقع نرم افزارهای کامپیوتری کمک می کند این لا یه ها ابزار ها و روشهای مهندسی نرم افزار هستند.

روشهای مهندسی نرم افزار، شیوه های فنی برای ساخت نرم افزار فراهم می کنند. روشها در حقیقت راهی برای انجام وظایف مهندسی نرم افزار هستند این وظایف برای مثال شامل تحلیل خواسته ها، طراحی، ساخت برنامه ها، آزمایش و پشتیبانی می شود.

ابزارهای مهندسی نرم افزار، شامل پشتیبانی خودکار یا نیمه خودکار برای روشها هستند. هنگامی که ابزارها گرد هم آیند، به طوریکه اطلاعات ایجاد شده توسط یک ابزار، توسط ابزار دیگر قابل استفاده باشند، سیستمی برای پشتیبانی بسط نرم افزار شکل می گیرد که مهندسی نرم افزار به کمک کامپیوتر (CASE) نام دارد.

کار مهندسی نرم افزار را در سه بخش کلی می توان گروهبندی کرد. که به نوع برنامه کاربردی، اندازه پروژه یا پیچیدگی آن مربوط نمی شود.[۱]

تعریف، مهندس نرم افزار تعیین می کند چه اطلاعاتی باید پردازش شود، کدام عمل و کارایی مطلوب است، چه رفتارهای سیستمی قابل انتظار است، چه محدودیتهایی در طراحی وجود دارد، و چه ملاکهای نشان دهنده تعریف یک سیستم موفق مورد نیاز است و خواسته های کلیدی سیستم و نرم افزار تشخیص داده می شود. گرچه اعمالی که در بخش تعریف انجام می شوند به الگوی مهندسی نرم افزار بستگی دارند ولی در هر حال سه کار هرچند به راههای مختلف انجام می شود: مهندسی اطلاعات ، طرح ریزی پروژه نرم افزاری و تحلیل خواسته ها**.** [۱]

توسعه، یعنی در طول توسعه، مهندس نرم افزار تعیین می کند داده ها چه ساختاری دارند، عملیات چگونه در یک معماری نرم افزاری پیاده سازی شوند، جزئیات روالها چگونه پیاده سازی شود، واسطها چه ویژگیهایی داشته باشند، طراحی چگونه به یک زبان برنامه سازی ترجمه شود و آزمایش به چه نحوی انجام شود.روشهای انجام توسعه متغیرند ولی سه وظیفه فنی همواره باید به انجام برسد: طراحی نرم افزار، تولید کدها و آزمایش نرم افزار.

پشتیبانی شامل تغییرات و تصحیحات مورد نیاز برای تکامل نرم افزا ر است و نیز تغییراتی که ناشی از تغییر خواسته های مشتریان هستند.درپشتیبانی چهار نوع تغییر روی نرم افزار انجام می شود:

- **تصحیح**. حتی در صورتی که بهترین روشها برای اطمینان از کیفیت انجام شود ، باز هم این احتمال وجود دارد که مشتری معایبی را در نرم افزار بیابد. نگهداری تصحیحی، به نرم افزار برای بر طرف کردن معایب و اشکالاتش کمک می کند.
- **تطابق**. به مرور زمان، محیط اولیه(مثل CPU ، سیستم عامل، قواعد کار، ویژگیهای محصول) که نرم افزار برای آنها بسط یافته است، احتمالا تغییر می کند. نگهداری تطابقی اصلاحاتی در نرم افزار بوجود می آورد تا با تغییرات محیط خارجی مطابقت کند.
- **بهبود**. وقتی نرم افزار مورد استفاده قرار می گیرد، کاربر متوجه عملکردهایی می شودکه افزودن آنها موجب بهتر شدن نرم افزار می شود. نگهداری بهبودی این عملکرد ها را که بیشتر از خواسته های اولیه نرم افزار هستند اضافه می کند.
- **جلوگیری**. نرم افزار کامپیوتری در اثر تغییرات زیاد کارایی خود را از دست می دهدو از این رو، نگهداری پیشگیرانه که غالبا مهندسی مجدد نرم افزار نامیده می شود باید اجرا شود تا نرم

افزار نیازهای کاربران نهایی خود را براورده سازد. در اصل، نگهداری پیشگیرانه تغییراتی در برنامه های کامپیوتر اعمال می کند که بتوان آنها را راحت تر تصحیح کرد، تطبیق داد و بهبود بخشید.

برای حل مسائل واقعی ، یک مهندس نرم افزار یا تیمی از مهندسان باید یک روش توسعه نرم افزار تعیین کنند که در برگیرنده روشها و ابزارها وبخش های کلی شرح داده شده باشد. این روش را غالبا مدل یا الگوی مهندسی نرم افزار می نامند. یک مدل برای مهندسی نرم افزار بر اساس نوع پروژه و نوع کاربرد ، روشها و ابزار مورد استفاده و کنترلها و قطعات قابل تحویل مورد نیاز انتخاب می شود.
در کتاب مهندسی نرم افزار پرسمن توسعه کل نرم افزار بصورت یک حلقه در نظر گرفته شده که چهار مرحله دارد: وضع موجود، تعیین مسئله، توسعه فنی، و جامعیت راه حل ؛ وضع موجود، وضعیت کنونی امور را نشان می دهد؛ تعیین مسئله مشخص می کند که چه مسئله خاصی باید حل شود؛ توسعه فنی، مسئله را با به کارگیری فناوری حل می کند و با جامعیت راه حل ، نتایج (مانند مستندات، برنامه ها، داده ها، وظایف کاری جدید و محصول جدید) تحویل کسانی می شود که تقاضای حل مسئله را کرده بودند. این مراحل و بخشها کاملا با بخشهای مهندسی نرم افزار یعنی تعریف توسعه وپشتیبانی سازگاری دارند.
در جهان واقعی ، مرز بندی میان مراحل توسعه نرم افزار به این صورت ایده آل دشوار است، زیرا بین مراحل تداخل رخ می دهد. ولی از همین دید ساده شده هم می توان نتایج خوبی گرفت. مدل انتخاب شده برای یک محصول نرم افزاری، هر چه که باشد، همه مراحل وضع موجود، تعیین مسئله ، توسعه فنی و جامعیت راه حل، همزمان در یک سطح از اهمیت و با هم وجود دارندو بصورت بازگشتی هر چهار مرحله به یک میزان در تحلیل یک کاربرد و تولید قطعه ای کوچک از کد اجرا می شود.
مدلهای متفاوتی برای مهندسی نرم افزار وجود دارد. این الگوها یک فعالیتهای مهندسی نرم افزار را منظم می کنند. هریک از این مدلها به شیوه ای تعریف شده است که به کنترل و هماهنگ سازی یک پروژه نرم افزاری واقعی کمک کند. برخی از مدلها:

- **مدل ترتیبی خطی**[1] . این مدل که به آن مدل آبشار یا چرخه حیات نیز می گویند، یک روش سیستماتیک و ترتیبی برای توسعه نرم افزار پیشنهاد می کند که یعنی نرم افزار از چند مرحله تشکیل شده است مانند: تحلیل، طراحی، کد نویسی، آزمایش و پشتیبانی . بعد از انجام هر مرحله ، مرحله بعدی توسعه نرم افزار انجام می شود.
- **مدل ساخت نمونه اولیه**[2]. الگوی ساخت نمونه اولیه با جمع آوری خواسته ها آغاز می شود. مشتری و سازنده با هم ملاقات می کنند و اهداف کلی نرم افزار را تعیین می کنند. سپس یک

[1] Water fall
[2] Prototyping

طراحی سریع صورت می پذیرد. در طراحی سریع هدف اصلی ارائه آن دسته از ویژگیهای نرم افزار است که به چشم مشتری یا کاربر می آیند. طراحی سریع منجر به ساخت نمونه اولیه می شود. نمونه اولیه مورد ارزیابی مشتری یا کاربر قرار گرفته از آن برای تعیین خواسته های نرم افزاری استفاده می شود که قرار است ساخته شود. در حقیقت برنامه ابتدایی برای آزمایش کاربر ساخته می شود، نیازهای دقیق سیستم تعیین می شود و مجددا نرم افزار پیاده سازی می شود.

- **مدل RAD** [1]. که همین مدل را سامرویل برنامه نویسی اکتشافی[2] می نامد .توسعه کاربردی سریع یک نرم افزار، توسعه تدریجی نرم افزار است که در یک چرخه توسعه بی اندازه کوتاه انجام می شود. این مدل شکل پر سرعت مدل ترتیب خطی است که در آن توسعه سریع با استفاده از مولفه های موجود انجام می شود. به این ترتیب درحداقل زمان یک سیستم ایجاد می شود اصلاحات لازم روی آن انجام می شود . این مدل برای سیستمهایی مفید است که کاربران نمی توانند نیازهایشان را دقیقا مشخص کنند.
- **مدل گام به گام** . یک مدل تکاملی است در مدلهای تکاملی مهندس نرم افزار هر بار نسخه ای از نرم افزار را تولید می کند که از قبل کاملتر است . در مدل گام به گام مدل آبشاری به صورت تکراری انجام می شود ودر هر بار تکرار مراحل مدل آبشاری یک قطعه قابل تحویل از نرم افزار تولید می شود.در گام نخست غالبا محصول هسته ای[3] تولید می شود یعنی به خواسته های پایه ای می پردازد ولی نرم افزار بسیاری از ویژگیها را ندارد. مشتری محصول هسته ای را ارزیابی می کند در نتیجه طرحی برای گام بعدی توسعه می یابد. این فرآیند به دنبال تحویل هر قطعه تکرار می شود تا اینکه محصول کامل تولید شود.
- **مدل مارپیچی**[4]. این مدل هم یک مدل تکاملی تولید نرم افزار است که که جنبه تکرارش مثل مدل ساخت نمونه اولیه است و مراحل ترتیبی مدل آبشاری را هم دارد مدل مارپیچی هم نرم افزار را بصورت نسخه هایی که هر بار کاملتر می شوند توسعه می دهد . مدل مارپیچی به چند فعالیت اساسی تقسیم می شود که به آنها نواحی کاری هم می گویند [۱]که می تواند شامل اینها باشد: ارتباط با مشتری، طرح ریزی، تحلیل ریسک، مهندسی، ساخت و ارائه، ارزیابی مشتری.

[1] Rapid Application Development
[2] Exploratory programming
[3] Core product
[4] Spiral

- **اسمبل کردن سیستم با اجزای قابل استفاده مجدد**[1]. در این روش سیستم از اجزای موجود که قابلیت استفاده مجدد دارند ساخته می شود و به این ترتیب نرم افزار به جای ایجاد فقط اسمبل می شود

مدلهای توسعه نرم افزار را یک تیم پروژه نرم افزاری بکار می برند و برای همین ابزارهایی برای کمک به یک سازمان نرم افزاری برای استفاده از این مدلها برای تحلیل فرآیند جاری نرم افزار و سازماندهی وظایف کاری و کنترل نظارت بر این وظایف و مدیریت کیفیت نرم افزار بوجود آمده اند.

کاری که این ابزارها انجام می دهند این است که یک مدل خودکار برای فرآیند های مشترک سازمان بسازند که بصورت شبکه ای تمامی فرآیند های جاری سازمان را با آن تحلیل کنند بطوریکه استفاده از چنین مدلهای خودکاری ممکن است باعث کاهش هزینه و زمان در سازمان شود.

حتی هنگامی که با این مدل خودکار فرآیند مناسب برای سازمان ساخته شد ، از ابزارهای دیگری هم می توان برای تخصیص، نظارت و حتی کنترل کلیه وظایف مهندسی نرم افزار درمدل استفاده نمود. هریک از اعضای تیم پروژه نرم افزاری می تواند از چنین ابزاری برای تهیه لیست کنترلی از وظایف کاری که باید انجام شود، محصولات کاری که باید تولید شود و فعالیتهای تضمین کیفیتی که باید به اجرا درآید، استفاده کند. این ابزارها را می توان برای هماهنگ ساختن ابزارهای دیگر مهندسی نرم افزار به کمک کامپیوتر که برای یک وظیفه کاری خاص مناسب باشند به کار برد. در حقیقت استفاده از این مدلهای توسعه باعث می شوند که کار توسعه نرم افزار بصورت سیستماتیک و منظم انجام شود طوریکه حتی بتوان به کمک برخی ابزار ها اعمال نظارتی و کنترلی را هم بصورت خودکار انجام داد.

[1] System assembly from reusable components

۲-۱مدیریت پروژه های نرم افزاری

در کتاب مهندسی نرم افزار پرسمن مدیریت نرم افزار به این شکل تعریف شده است: مدیریت پروژه شامل طرح ریزی، نظارت و کنترل افراد، فرآیند و رویدادهایی می شود که به موازات تکامل نرم افزار از مفهوم مقدماتی تا پیاده سازی عملی رخ می دهند. سامرویل عقیده دارد نمی توان یک توصیف استاندارد از شغل مدیریت در سازمانها داشت و فقط مسئولیتهایی را که اغلب مدیران در سازمانها بر عهده دارند می شمارد این و ظایف عبارتند از نوشتن پیشنهاد[1] ، برنامه ریزی و زمانبندی پروژه ، انتخاب وارزیابی کارکنان و نوشتن گزارشها. در مدیریت موثر پروژه های نرم افزار، بر چهار مورد تاکید می شود:افراد، محصول، فرآیند، پروژه. مدیری که فراموش کند امر مهندسی نرم افزار تلاشی شدیدا انسانی است، هرگز در مدیریت پروژه موفق نخواهد بود. مدیری که نتواند ارتباط مفهومی با مشتری را در شروع پروژه برقرار کند، ممکن است مسئله ای را که مورد نظر مشتری نیست حل کند مدیری که بدون یک طرح پروژه از پیش آماده شروع به کار کند، موفقیت محصول را به مخاطره می افکند.

مدیر،افراد را باید در قالب تیمهایی کارامد سازماندهی کند، در آنها انگیزه کار نرم افزاری با کیفیت بالا ایجاد کندو آنها را هماهنگ کند تا ارتباط موثری بینشان بوجود آیدانتظاراتی را که مشتری از محصول دارد باید بخوبی برای سازندگان محصول بفهماند، نیازمندیهای محصول را به قطعات کوچکتر تقسیم کند و در اختیار تیم نرم افزاری قرار دهد.مدل فرآیندی که برای توسعه نرم افزار انتخاب می شود باید با افراد و نوع مسئله مطابقت داشته باشد. الگوی مهندسی نرم افزار مناسب انتخاب می شود، و یک مجموعه وظایف کاری برای انجام کارمشخص می گردد. در حقیقت توسعه محصول طوری برنامه ریزی می شود که تیم نرم افزار بتواند در کار خود موفق شود.

یک عنصر محوری در همه پروژه های نرم افزاری، افراد است. مهندسان نرم افزار را می توان در قالب ساختارهای تیمی متفاوت سازماندهی کرد مثل سلسله مراتبهای کنترلی سنتی یا تیمهای مبتنی بر الگوی باز که در آنها همه افراد تیم در یک رتبه قرار دارند. چند تکنیک هماهنگ سازی و ارتباطی هم می توان برای پشتیبانی تیم به کار برد.

مدیر پروژه های نرم افزاری در ابتدا ی هر پروژه نرم افزاری جدید با معضل روبروست. لازم است تحلیل مفصلی از نیازمندیهای نرم افزارانجام شود،اطلاعات ضروری برای برآورد ها را فراهم شود، و انجام تحلیل هم غالبا هفته ها یا ماهها زمان می برد. بدتر اینکه نیازمندیهاممکن است متغیر باشند وطی توسعه نرم افزار تغییر کنند. از این رو محصول و مسئله ای که باید حل شود در همان ابتدای پروژه مورد بررسی قرار می گیرد.پرسمن عقیده دارد به عنوان حداقل کار، حوزه محصول باید مشخص گردد و نخستین فعالیت مدیریت نرم افزار، تعیین دامنه کاربرد نرم افزار است. همچنین پرسمن مشخص می

[1] Proposal

کند که دامنه کاربرد نرم افزار چیست و می گوید دامنه کاربرد با پاسخ دادن به پرسشهای زیر مشخص می شود:

محیط. نرم افزاری که قرار است ساخته شود چگونه در یک سیستم بزرگتر یا محیط قرار می گیرد ، در چه محیطهایی می تواند قرار گیرد و این محیط چه محدودیتهایی دارد.

اهداف اطلاعات. نرم افزار داده هایی را برای مشتری تولید می کند؟ و چه اشیای داده هایی برای ورودی لازم است؟یعنی اینکه کلا داده های مورد استفاده نرم افزار به چه شکلی هستند.

عملکرد و کارایی. نرم افزار چه عملی برای تبدیل داده های ورودی به داده های خروجی انجام می دهد؟ آیا باید کار خاصی برای بدست آوردن کارایی بهتر انجام شود؟

دامنه پروژه نرم افزاری بایدطوری تعیین شود که در سطح فنی و مدیریتی قابل فهم باشد.یعنی، داده های کمی (مثل تعداد کاربرهای همزمان، حداکثر زمان پاسخ مجاز) به طور واضح بیان شده باشند؛ محدودیتها و یا شرایط تعیین کننده (مثلا هزینه محصول،محدودیت حافظه) ذکر شود، و عوامل مثبت(مثلا الگوریتم های مطلوب به خوبی درک شده اند و به زبان ویژوال بیسیک در دسترس هستند) شرح داده شود.

تعیین دامنه کاربرد نرم افزار کار ساده ای نیست و بنا براین تکنیکهایی برای حل مسئله تعیین دامنه بکار می رود از جمله تکنیک تجزیه مسئله .تجزیه مسئله که به آن تقسیم مسئله هم می گویند، فعالیتی است که برای تحلیل نیازمندیهای نرم افزاری را انجام می شود. طی فعالیت تعیین دامنه کاربرد،مسئله بطور کامل تجزیه نمی شود. بلکه تجزیه فقط به این صورت انجام می شود که مشخص شود قابلیتهای محصولی که باید تحویل داده شودچیست و فرآیندی که برای تحویل آن بکار می رود چیست.

انسان وقتی در مقابل یک مسئله پیچیده قرار می گیرد، معمولا از روش تقسیم و حل استفاده می کند. یعنی ، یک مسئله پیچیده به دو مسئله کوچکتر تقسیم می شودکه حل کردن آنها راحت تر است. این روشی است که در شروع طراحی پروژه انجام می شود. ابتدا وظایفی که نرم افزار باید انجام دهد تحلیل می شوند تا پیش از شروع برآورد جزئیات لازم فراهم شود. چون برآورد های هزینه ای و زمانی برای عملیات توسعه کل نرم افزار انجام می شود ، قدری تجزیه معمولا مفید واقع می شود. مدیریت پروژه نرم افزاری یکی از فعالیتهای چتری در مهندسی نرم افزار است. یعنی این فعالیت بیش از هر فعالیتی آغاز می شودو در سرتاسر مراحل تعریف، توسعه و پشتیبانی نرم افزار کامپیوتری ادامه می یابد.

در کل می توان گفت فعالیت مدیر پروژه شامل اندازه گیری معیارها، انجام برآوردها، تحلیل ریسک، رمانبندی پیشبرد پروژه و کنترل کل پروژه می شود .

مدیران نرم افزار مسئول برنامه ریزی پروژه، اجرای دقیق کار و حصول اطمینان از اجرای استاندارد های مورد نیاز، خاتمه نرم افزار در زمانی معین و با سرمایه تعیین شده هستند. مدیریت خوب می تواند

موفقیت پروژه را تضمین کند، مدیریت بد یا پشتیبانی ناقص مدیریت، احتمالا منجر به تاخیر در تحویل نرم افزار شده، هزینه ها را از مقدار تخمین زده شده بالاتر می برد . از نظر سامرویل مدیریت پروژه نرم افزار با سایر انواع مدیریت پروژه مهندسی داری تفاوتهای زیر است:

محصول نرم افزاری، پیچیده است . در سایر انواع مدیریتها مدیر کاملا می تواند روند انجام کار را ببیند مشاهده کند که چه قسمتهایی از کار انجام شده و آیا مطابق زمانبندی است یا از زمان پیش بینی شده عقب است. نرم افزار قابل مشاهده یا لمس نیست . مدیر پروژه فقط می تواند به کمک مستندات، پیشرفت پروژه را کنترل می کند.

فهم درستی از فرآیند نداریم. سایر مهندسی ها معمولا یک تاریخ طولانی و یک اصول ثابت و امتحان شده دارند بنابراین، مراحل توسعه بخوبی درک شده و قابل پیش بینی است. در مهندسی نرم افزار اینطور نیست. مدلهایی مثل مدل آبشاری دوران زندگی نرم افزار ، نمایشهای ساده شده ای هستند.و غالبا در واقعیت نرم افزار را به این ترتیب نمی توان مرحله بندی کرد.

سیستمهای نرم افزار بزرگ اغلب پروژه های جدیدی هستند. آنها با پروژه های قبلی فرق می کنند. تجربه ای در خصوص این پروژه ها وجود ندارد.اکثرپروژه های نرم افزاری فقط یکبار نوشته می شوند و تفاوتهای زیادی با پروژه های قبلی دارند.

بنا براین مدیریت کارامد پروژه نرم افزار، به برنامه ریزی دقیق پیشرفت پروژه، پیش بینی مشکلاتی که ممکن است بروز کنند و آماده نمودن راه حلهای عملی این مشکلات بستگی دارد. از ابتدای پروژه، نقشه مشخصی باید برای کنترل پروژه مورد استفاده قرار گیرد. این نقشه ثابت نیست و باید با پیشرفت پروژه اصلاح شود زیرا با پیشرفت پروژه اطلاعات بهتر و بیشتری در اختیار خواهیم داشت. برای برنامه ریزی پروژه ابتدا محدودیتهایی که پروژه را تحت تاثیر قرار می دهند ارزیابی می شوند. سپس تخمینی از پارامترهای پروژه مثل نوع پروژه، اندازه وعملکردهای آن انجام می شود. بعد از آن، مواردی که باید تحویل داده شوند تعریف می شوند. سپس این مراحل در یک حلقه تکرار می شود. زمانبندی پیشبرد پروژه تنظیم شده و فعالیتهای تعریف شده در این زمانبندی انجام می شود.

با جمع آوری اطلاعات بیشتر، مدیر پروژه فرضهای پروژه را دوباره بررسی کرده و تاثیر آنها را بر زمانبندی ، برآورد می کند. اگرباعث تاخیر در پروژه شوند، ممکن است مجبور شوند دوباره با مشتری در باره محدودیتهای پروژه و مواردی که قرار است تحویل آنها شود مذاکره کنند. اگر مذاکره مجدد موفق نباشد و اثری در پیشبرد پروژه نداشته باشد، ممکن است نیاز به مرور تکنیکی پروژه باشد. تامشخص شود آیا روشهای دیگری وجود دارند که محدودیتها و زمانبندی پروژه را رعایت کنندو تمامی نیازهای

مشتری را برآورده سازد یا خیر.

هیچوقت مدیر تصور نمی کند که همه چیز به درستی انجام می شوند.مشکلات زیادی در طول پروژه بروز می کند ولی باید فرضهای اولیه ثابت بمانند. احتمالات کافی باید در نقشه وجود داشته باشد، به طوریکه محدودیتها ی پروژه در هر حلقه تکرار برنامه ریزی نیازمند مذاکره مجدد نباشد. همچنین سامرویل عقیده دارد نقشه توسعه نرم افزار فقط یکی از نقشه هایی است که باید برای پروژه ایجا شود. سایر نقشه های مورد نیاز عبارتند از:

- نقشه ارزیابی.
- نقشه مدیریت پیکر بندی.
- نقشه توسعه و آموزش کارکنان.
- نقشه نگهداری.

نقشه ارزیابی کنترل کیفیت نرم افزار را در طول توسعه نرم افزار انجام می دهد و نقشه مدیریت پیکر بندی مدیریت تغییر سیستم است و اینکه چطور این سیستم های تغییر یافته را به مشتری عرضه کنیم.

۳-۱ معیارهای پروژه و فرآیند نرم افزار

اندازه گیری، اساس تمامی رشته های مهندسی است و مهندسی نرم افزار نیز از این قاعده مستثنی نیست. اندازه گیری اطلاعاتی را فراهم می کند که ما را قادر می سازد پدیده ها و اهدافمان را بهتر درک کنیم. لرد کلوین گفته است:" وقتی می توانید آنچه را که از آن صحبت می کنید، اندازه گیری کنید و بر حسب اعداد بیان کنید، درباره آن آگاهی دارید. وقتی نتوانید آن را بر حسب اعداد بیان کنید، دانش شما ناقص است: ممکن است دانشی ابتدایی باشد ولی به ندرت می توان آن را علم دانست."

جامعه نرم افزاری هم این این سخن راپذیرفته و شروع به اندازه گیری نرم افزار ها نموده است. اندازه گیری نرم افزارها ر اساس معیارهای نرم افزاری انجام می شود. اندازه گیریرا می توان برای بهینه کردن فرآیند توسعه نرم کمک به برآوردهزینه و زمان توسعه نرم افزار، کنترل کیفیت، و کنترل پروژه به کار برد. استفاده از اندازه گیری به مهندس نرم افزار کمک می کند تا کیفیت محصولات را ارزیابی کرده و به تصمیم گیریهای تاکتیکی در طول پیشرفت پروژه نیز کمک می کند.معیارهای محصول و فرآیند نرم افزاری، مقادیری کمی هستند که با آنها بازدهی نرم افزار هایی را اندازه گیری می کنند که از آن معیار ها به عنوان قواعد کاری استفاده می کنند معیارها یکسری قواعد هستند که بصورت کمی قابل اندازه گیری اند و به این ترتیب می توان بازدهی نرم افزارها و پروژه هایی را که از این معیارها استفاده می کنند اندازه گیری کرد. داده هایی که از اندازه گیری معیارها بدست می آیند، تحلیل می شوند، با میانگین های گذشته مقایسه می شوند و مورد ارزیابی قرار می گیرندتا معین شود که بهره وری چقدر بوده آیا بهبود بهره وری و کیفیت رخ داده است یا خیر. معیارها برای مسائل و مشکلات ریز نرم افزار نیز به کار می روند به طوریکه بتوان راه حلی برای آنها یافته و فرآیند نرم افزار را بهبود بخشد. پرسمن میزان، در مهندسی نرم افزار را به این شکل تعریف می کند: در مهندسی نرم افزار، کمیتی است که نشانگر حد، مقدار، ابعاد، ظرفیت یا اندازه صفتی از محصول یا فرآیند است.و اندازه گیری در حقیقت عمل تعیین میزان است.

هنگامی که یک سری داده ای منفرد جمع آوری شده باشد(مثل تعداد خطاهای کشف شده در یک پیمانه) یک میزان ایجاد شده است. اندازه گیری با جمع آوری میزانهایی از تعداد خطاها برای هر مورد انجام می شود .معیار نرم افزاری با تک تک میزانها به طریقی در ارتباط است(مثل میانگین تعداد خطاهای یافت شده به ازای هر بازبینی یا تعداد میانگین خطاهای یافت شده به ازای هر نفر – ساعت صرف شده روی بازبینی ها).

مهندس نرم افزار معیارهایی را جمع آوری و ایجاد می کند، تا شاخصهایی از آنها به دست آید. شاخص، معیار یا تر کیبی از معیار هاست که درکی از فرآیند نرم افزار، پروژه نرم افزاری یا خود محصول به دست می دهد.[۱]

شاخص دیدی برای مدیر پروژه یا مهندسان نرم افزارفراهم می کند تا آنها را قادر به تنظیم فرآیند شوند. اندازه گیری در جهان مهندسی بسیار متداول است.اما اندازه گیری در جهان مهندسی نرم افزار کمتر متداول است. مشکل این است که مهندسان نرم افزار سر چیزهایی که باید اندازه گیری شود و میزانهایی که باید مورد ارزیابی قرار گیرد توافق ندارند. منظور از ایجاد معیارها بدست آوردن شاخص هایی برای پروژه است بنابراین معیارها باید طوری جمع آوری شوند تا بتوان شاخص ها را به خوبی کشف کرد . شاخصهای فرآیند به سازمان مهندسی نرم افزار این امکان را می دهند که از بازدهی فرآیند موجود دیدی درست بدست آورند .مدیران و سازندگان را قادر می سازد که بتوانند ارزیابی کنند چه چیزهایی عملی هستندو چه چیزهایی نیستند. معیارهای فرآیند، از همه پروژه ها و در مدت زمانی طولانی جمع آوری می شوند. هدف از استفاده آنها فراهم آوردن شاخص هایی است که منجر به بهبود فرآیند نرم افزار در بلند مدت می شود. شاخص های پروژه مدیر پروژه را قادر می سازد تا به وضعیت پروژه در حال پیشرفت دست پیدا کند؛ خطرات بالقوه را دنبال کند ؛ نواحی مشکل آفرین را پیش از بحرانی شدن آنها کشف کند؛ جریان کار یا وظایف را تنظیم کند و توانایی تیم پروژه در کنترل کیفیت محصولات کاری نرم افزار را ارزیابی کند. [۱]

فصل دوم-
شرح برآورد پروژه

۱-۲برآورد پروژه نرم افزاری

منظور از برآورد پروژه نرم افزاری تخمین هزینه و زمان توسعه نرم افزار پیش از شروع پروژه است. اما برآورد کار و هزینه هرگز علم دقیقی نخواهد بود.متغیرهای بیشمار انسانی ، فنی ، محیطی، سیاسی می توانند هزینه نهایی نرم افزار و کار انجام شده برای توسعه آن را تحت تاثیر قرار دهند. ولی برآورد پروژه نرم افزاری را می توان به یک سری مراحل سیستماتیک تبدیل کرد که برآورد هایی با ریسک قابل قبول فراهم کند . در کتاب مهندسی نرم افزار پرسمن برای دستیابی به برآورد های قابل اطمینان هزینه و کار چند انتخاب پیشنهاد شده :

- برای انجام برآورد تا اواخر پروژه صبر کنیم . هر چه صبر کنیم اطلاعات مفیدتر و صحیح تری از پروژه بدست می آوریم و این اطلاعات تا اتمام پروژه بیشتر می شود تا اینکه پس از کامل شدن پروژه برآوردی صد در صد درست خواهیم داشت
- از بهترین برآورد های انجام شده روی پروژه هایی که قبلا کامل شده اند استفاده کنیم.
- از تکنیک های تجزیه نسبتا ساده برای تولید برآورد های هزینه ای و کاری پروژه استفاده کنیم
- برای برآورد هزینه و کار نرم افزار از یک یا چند مدل تجربی استفاده کنیم.

انتخاب اول که عملی نیست. برآورد هزینه ها باید قبل از شروع پروژه انجام شود فراهم آید هر چند می دانیم که هر چه بیشتر صبر کنیم بیشتر خواهیم دانست و هرچه بیشتر بدانیم احتمال بروز خطاهای جدی در برآورد ها کمتر می شود.

انتخاب دوم می تواند نتایج خوبی بدهد به شرط اینکه پروژه های فعلی مشابه پروژه های گذشته باشد و عوامل دیگری که بر پروژه اثر می گذارند مثل مشتری ،شرایط تجاری و غیره هم مشابه باشند.متاسفانه تجربه گذشته همیشه نمی تواند معیاری برای آینده باشد.

دو انتخاب باقیمانده روشهایی هستند که امکان استفاده از آنها در برآورد پروژه نرم افزاری بیشتر است . برای بدست آوردن بهترین نتیجه این دو روش را باید همراه یکدیگر به کار برد طوریکه هریک دیگری را تایید کنند. در روشهای تجزیه ای از یک روش تقسیم و حل برای برآورد پروژه نرم افزاری استفاده می شود .با تجزیه پروژه به یکسری اعمال اصلی می توان هزینه و کار را به صورت مرحله ای برآورد نمود. مدلهای برآورد تجربی را هم می توان برای تکمیل تکنیکهای تجزیه ای به کار برد. این مدل، یک مدل مبتنی بر تجربه(داده های تاریخی) بوده به شکل زیر است:

$$D = f(Vi)$$

که d یکی از مقدار هایی که می خواهیم برآورد کنیم (مثل کار، هزینه، مدت زمان پروژه) و Vi پارامتر های مستقل که برای انجام برآورد انتخاب کرده ایم (مثل LOC یا Function Point برآورد شده) است.

ابزارهای برآورد خودکار تقریبا زیادی ساخته شده اند. ابزارهای برآورد خودکار یک یا چند تکنیک تجزیه ای یا مدل تجربی را پیاده سازی می کنند. هنگامی که ابزارهای خودکار با یک واسط گرافیکی ساخته شوند ابزار جذابی برای برآورد هستند. در چنین سیستمهایی ویژگیهای سازمانی که نرم افزار در آن توسعه می یابد (مثل تجربه و محیط) و نرم افزاری که قرار است توسعه یابد توصیف می شوند. برآورد هزینه از این داده ها به دست می آید. هر یک از انتخابهای ممکن برای برآورد هزینه های نرم افزاری ، فقط به خوبی داده های تاریخی بستگی دارد که برای انجام برآورد به کار رفته اند. اگر هیچگونه داده های تاریخی موجود نباشد تعیین هزینه قابل اعتماد نیست. ولی پیش از انجام برآورد باید دامنه کاربرد نرم افزار تعین شده باشد و اندازه آن را برآورد شده باشد.

درستی برآورد پروژه نرم افزاری به چند چیز بستگی دارد:

- درستی برآورد اندازه محصول
- توانایی اینکه از برآورد اندازه پروژه بتوانیم کار، زمان و هزینه ها را بدست آوریم که به دسترسی به معیارهای قابل اطمینانی که از پروژه های قبلی بدست آمده بستگی دارد.
- انعکاس تواناییهای تیم نرم افزاری در برنامه ریزی
- پایداری خواسته های پروژه و محیطی که نرم افزار در آن توسعه می یابد.

اندازه پروژه های نرم افزاری بر اساس دو معیار خط کد[1] و نقطه عملکرد[2] بدست می آید.داده های LOC و Function Point در اثنای برآورد پروژه نرم افزاری به دو شیوه به کار برده می شود:

- برای تعیین اندازه هر یک از عناصر نرم افزار
- بصورت معیارهایی که از پروژه های گذشته بدست آمده اند و برای برآورد هزینه و کار بکار می روند.

برآورد های LOC و Function Point تکنیکهای برآورد متمایزی هستند ولی شباهت هایی هم دارند. برنامه ریز ابتدا سعی می کند دامنه کاربرد نرم افزار را تعیین کند، دامنه کاربرد همانطور که قبلا توضیح داده شد صفتهای محیطی که نرم افزار در آن توسعه می یابد، شکل و چگونگی داده های مورد نیاز و داده های خروجی نرم افزار و عملکرد و کارایی نرم افزار است بعد از تعیین همگی اینها نرم افزار به یکسری عملیات کوچکتر تقسیم می شود تا هر کدام جداگانه برآورد شوند و سپس خط کد و یا نقطه عملکرد برای هر یک از این عملیات برآورد می شود. برنامه ریز ممکن است از کلاسها یا اشیا هم برای تعیین اندازه استفاده کند. سپس معیارهای بهره وری (مثل LOC/pm یا FP/pm) در مورد متغیر

[1] Line of Code
[2] Function Point

برآورد مناسب به کار برده می شود و هزینه یا کار برای عمل مورد نظر به دست می آید. برآورد های مربوط به همه عملیات باهم ترکیب شده کل پروژه را برآورد می کنند.
اما معیارهای بهره وری به عوامل دیگری هم بستگی دارند. میانگینهای LOC/pm یا FP/pm را باید بر اساس نوع و دامنه پروژه محاسبه کرد. یعنی، پروژه ها را باید بر حسب اندازه تیم، حیطه کاربردی، پیچیدگی و پارامتر های دیگر گروهبندی کرد. سپس باید میانگین دامنه های محلی را محاسبه کرد. هنگامی که پروژه جدید برآورد شد ابتدا باید نوع و دامنه راتعیین کرد و سپس میانگین دامنه مناسب را باید برای بهره وری به کار برد تا برآورد صورت پذیرد.

۲-۲مدلهای برآورد تجربی

یک مدل برآورد برای نرم افزار های کامپیوتری از فرمولهای تجربی برای پیش بینی کار به عنوان تابعی از LOC یا FP استفاده می کند. داده های تجربی که اکثر مدلهای برآورد از آن استفاده می کنند، از یک نمونه محدود پروژه ها بدست می آیند، به همین دلیل هیچ مدل برآوردی برای کلیه نرم افزار ها و در همه محیطهای توسعه مناسب نیست. بنا براین نتایج حاصل از این مدلها همیشه قابل اطمینان نیستند.

۱-۲-۲ ساختار مدلهای برآورد

یک مدل برآورد معمولا با استفاده از تحلیل و پردازش روی داده هایی که از پروژه های نرم افزاری قبلی جمع آوری شده اند بدست می آید . ساختار کلی چنین مدلهایی در کتاب مهندسی نرم افزار پرسمن به این صورت نشان داده شده:

$$E = A + B * (ev) ^ C$$

که در آن A و B و C ثابتهایی هستند که بطور تجربی بدست می آیند، E کار برآورد شده بر حسب نفر – ماه و ev متغیر برآورد (LOC یا FP) است. در کتاب مهندسی نرم افزار سامرویل هم فرمول مشابهی پیشنهاد شده است

$$Effort = A (KDSI)^b$$

که KDSI[1] برآوردی از اندازه پروژه وبقیه ثابتهایی هستند که بسته به نوع پروژه تغییر می کنند. به این تر تیب اکثر مدلهای برآورد شکلی دارن که کار بر اساس خصوصیات دیگر پروژه (مثل پیچیدگی مسئله، تجربه کارمندان و محیط توسعه) تنظیم می شود.

۳-۲ابزارهای برآورد خودکار

تکنیکهای تجزیه و مدلهای برآورد تجربی که قبلا بحث شدند، به عنوان بخشی از ابزارهای متنوع برآورد نرم افزار در دسترس قرار دارند.این ابزارهای برآورد خودکار، به برنامه ریز امکان می دهند تا هزینه و کار را برآورد کند. گرچه ابزارهای برآورد خودکار فراوانی وجود دارد، همه آنها دارای ویژگیهای کلی یکسان بوده همگی شش عمل کلی زیر را انجام می دهند:[۱]

- **تعیین اندازه قسمتهای مختلف پروژه:** اندازه یک یا چند مولفه نرم افزاری برآورد می شود. مثل صفحات نمایش، گزارشها ،مستندات
- **انتخاب فعالیتهای توسعه پروژه:** مجموعه وظایف مهندسی نرم افزار مشخص می شود.
- **پیش بینی سطوح پرسنلی:** تعداد افرادی که برای انجام کار در دسترس هستند مشخص می شود. چون رابطه میان افراد در دسترس و کار بسیار غیر خطی است، این یک ورودی مهم بشمار می رود.
- **پیش بینی کار لازم برای نرم افزار:** ابزار های برآورد از یک یا چند مدل استفاده می کنندکه بر اساس اندازه قطعاتی که باید تحویل داده شود کار لازم برای تولید آنها مشخص می شود.
- **پیش بینی زمانبندی پروژه:** هنگامی که کار، سطح پرسنلی، و فعالیتهای پروژه مشخص شد، یک زمانبندی را می توان با تخصیص کارها به فعالیتهای مهندسی نرم افزار تولید کرد.

[1] KDSI در شرح مدل کوکومو توضیح داده شده است.

۴-۲اندازه گیری نرم افزار

اندازه گیری در جهان فیزیکی به دو طریق انجام می شود، بطور مستقیم و بطور غیر مستقیم. معیارهای نرم افزاری را هم می توان به همین دو صورت اندازه گیری کرد. موازین مستقیم فرآیند مهندسی نرم افزار، شامل هزینه و انرژی بکار رفته است . موازین مستقیم محصول شامل خطوط کد(LOC) تولید شده، سرعت اجرا، اندازه حافظه و نقایص گزارش شده در یک مدت زمان تعیین شده است. و موازین غیر مستقیم هم اندازه گیری های مربوط به کیفیت نرم افزار است در اینجا مقصود از اندازه گیری، اندازه گیری موازین مستقیم فرآیند و محصول نرم افزاری است.

اگر قرار داد های مشخصی از قبل وضع شده باشند هزینه و انرژی لازم برای ساخت نرم افزار، تعداد خطوط کد تولید شده یعنی موازین مستقیم را براحتی می توان بدست آورد.

برای اندازه گیری تعداد خطوط کد و هزینه و کار معیار هایی لازم است پرسمن دو نوع معیار را برای اندازه گیری موازین مستقیم پیشنهاد کرده است. معیار های اندازه گرا و معیار ها عملکرد گرا.

۱-۴-۲معیار های اندازه گرا

برای اینکه معیاری داشته باشیم که برای تمامی پروژه ها یکسان باشد تعداد خطوط کد را انتخاب می کنیم. بنابراین برای هر پروژه می توان مجموعه ای از معیار های اندازه گیری را ایجاد کرد: تعداد خطا به ازای هر KLOC ، تعداد خطا به ازای هر نفر ماه، LOC به ازای هر نفر ماه، هزینه مصرفی به ازای هر صفحه از مستندات.

۲-۴-۲معیار های عملکرد گرا

معیار های عملکرد گرا از میزان عملکرد ارائه شده توسط برنامه کاربردی برای ایجاد یک معیار مشترک برای پروژه ها استفاده می کنند. چون عملکرد را مستقیما نمی توان اندازه گیری کرد باید آن را به طور غیر مستقیم با استفاده از موازین مستقیم دیگر بدست آوردبرای اندازه گیری عملکرد نرم افزار میزانی با عنوان نقطه عملکرد وجود دارد.نقاط عملکرد با استفاده از یک رابطه تجربی بر اساس دامنه اطلاعات نرم افزار و ارزیابی پیچیدگی نرم افزار بدست می آیند. نقاط عملکرد با کامل کردن یک جدول محاسبه می شوند.

پیچیده	متوسط	ساده	شمارش	پارامترهای اندازه گیری
۶	۴	۳		تعداد ورودیهای کاربر
۷	۵	۴		تعداد خروجیهای کاربر
۶	۴	۳		تعداد در خواستهای کاربر
۱۵	۱۰	۷		تعداد فایلها
۱۰	۷	۵		تعداد واسطهای خارجی

پنج ویژگی از دامنه اطلاعاتی تعیین می شوند و شمارشهایی در محل مناسب جدول صورت می گیرد مقادیر دامنه اطلاعات به شیوه زیر تعیین می شوند. [۱]

- **تعداد ورودیهای کاربر**. هر ورودی کاربر که داده های متما یز داشته باشد شمارش می شود. ورودیها باید از در خواستها که جداگانه شمارش می شود متمایز شود.
- **تعداد خروجیهای کاربر**. هر خروجی کاربر شمارش می شود. خروجی به گزارشها ، صفحات نمایشی، پیامهای خطا و غیره اطلاق می شود .
- **تعداد در خواستهای کاربر**. در خواست به عنوان یک ورودی On-Line تعریف می شود که منجر به تولید پاسخ بی درنگ به شکل خروجی On-Line می شود . هر یک از در خواستها شمارش می شود.
- **تعداد فایلها**. هر فایل اصلی منطقی یعنی گروهی منطقی از داده ها که ممکن است بخشی از یک بانک اطلاعاتی بزرگ یا یک فایل مجزا باشد شمارش می شود.
- **تعداد واسطهای خارجی**. همه واسطهای ماشین که برای انتقال اطلاعات به سیستم دیگری بکار می روند شمارش می شوند.

هنگامی که داده های فوق جمع آوری شد، یک مقدار پیچیدگی مشخص می شود.سازمانهایی که روشهای نقطه عملکرد را به کار می برند ملاکهایی برای تعیین اینکه یک ویژگی دامنه اطلاعاتی ساده متوسط یا پیچیده است دارند.

برای تعیین ضریب پیچیدگی برای نقاط عملکرد از رابطه استاندارد زیر استفاده می شود.

$$FP = \text{شمارش کل} * [0.65 + 0.01 * \text{مجموع (Fi) ها}]$$

که در آن شمارش کل حاصل جمع همه مدخلهای FP است و Fi ها مقادیر تنظیم پیچیدگی مبتنی بر پاسخهایی است که به پرسشهای زیر داده می شود: این پرسشها استاندارد هستند و در نرم افزار های

برآورد برای تعیین ضریب پیچیدگی استفاده می شوند در کتاب مهندسی نرم افزار پرسمن هم به اینها اشاره شده.

- آیا سیستم به پشتیبانی و بازیابی قابل اطمینان نیاز دارد؟
- آیا ارتباطات داده ای مورد نیاز است؟
- آیا عملیات پردازشی توزیع شده وجود دارد؟
- آیا کارایی اهمیت دارد؟
- آیا سیستم در یک محیط عملیاتی موجود و پرکار اجرا می شود؟
- آیا سیستم به در خواستهای On-Line نیاز دارد؟
- آیا در خواستهای On-Line نیاز به ساخت عملیات ورودی روی عملیات یا صفحات نمایش چندگانه دارد؟
- آیا فایلهای اصلی بصورت On-Line بهنگام می شوند؟
- آیا ورودی ها، خروجی ها و فایلها پیچیده اند؟
- آیا پردازش داخلی پیچیده است؟
- آیا کد طوری طراحی شده است که دوباره قابل استفاده باشد؟

۱. تبدیل و نصب در طراحی لحاظ شده است؟
۲. آیا سیستم برای نصب چند گانه در سازمانهای متفاوت طراحی شده است؟
۳. آیا برنامه کاربردی طوری طراحی شده است که تغییرات راتسهیل کند و به آسانی توسط کاربر استفاده شود؟

در پاسخ به هر یک از این پرسشها یک مقدار صفر(بی اهمیت) تا پنج (ضروری) نسبت داده می شود. مقادیر ثابت در معادله و ضرایب وزنی که در شمارش دامنه اطلاعاتی به کار برده شده اند به طور تجربی بدست آمده اند.

هنگامی که نقاط عملکرد محاسبه شدند از آنها به شیوه های مشابه LOC ، برای اندازه گیری موازین بهره وری نرم افزاری ، کیفیت و صفات دیگر استفاده می شود.

۵-۲ایجاد ارتباط میان معیارهای متفاوت

رابطه میان خطوط کد و نقاط عملکرد، به زبان برنامه نویسی بستگی دارد که برای پیاده سازی نرم افزار مورد استفاده قرار گرفته است. این دو میزان با هم ارتباط دارند و لازم است که این ارتباط وجود داشته باشد تا بتوان عملکرد تک تک قطعات نرم افزار را بدست آورد طوریکه این اندازه عملکرد به تعداد خط کد ها بستگی داشته باشد.

ارتباط میان زبان برنامه نویسی و تعداد خط کدها در هر نقطه عملکرد در کارهای پژوهشی بدست آمده ولی ثابت نیست.

جدول زیر برآورد خاصی از میانگین تعداد خطوط کد مورد نیاز برای ساخت یک نقطه عملکرد را در زبانهای برنامه نویسی گوناگون نشان می دهد:

زبان برنامه نویسی	میانگین LOC/FP
اسمبلی	۳۲۰
C	۱۲۸
C++	۶۴
کوبول	۱۰۶
فرترن	۹۰
پاسکال	۹۵
ادا	۵۳
ویژوال بیسیک	۳۲

از میزانهای خط کد و نقطه عملکرد غالبا برای بدست آوردن معیارهای بهره وری استفاده می شود ولی نکته ای در باره استفاده از داده های خط کد و نقطه عملکرد وجود دارد اینکه چگونه و چه استفاده هایی می توان از این داده ها کرد. آیا خط کد به ازای نفر-ماه در یک گروه را باید با گروه دیگر مقایسه کرد؟ آیا مدیران باید کارایی افراد را با استفاده از این معیارها ارزیابی کنند؟پاسخ این پرسشها منفی است. به این دلیل که عوامل زیادی روی بهره وری اثر می گذارند و کارایی یک گروه را نمی توان تنها با میزان کد خطهای تولیدی اش اندازه گرفت اما معیار های مبتنی بر خط کد و نقاط عملکرد برای پیش بینی هزینه و کار لازم برای توسعه نرم افزار نسبتا صحیح عمل می کنند. ولی برای استفاده از آنها در تکنیکهای برآورد باید اطلاعات و داده های تاریخی سازمان در دسترس باشد.

تکنیکهای برآورد LOC و FP از لحاظ نوع تجزیه با هم تفاوت دارند. هنگامی که کد خط به عنوان متغیر برآورد به کار می رود تجزیه با جزئیات تقریبا زیادی باید انجام شود. برای برآورد های FP تجزیه به صورت متفاوت انجام می شودهمانطور که قبل گفته شد هریک از ویژگیهای دامنه اطلاعاتی یعنی

ورودیها، خروجی ها، فایلهای داده ای، درخواستها و واسطهای خارجی و نیز چهارده مقدار تنظیم پیچیدگی برآورد می شوند، سپس برآوردهای حاصل را می توان برای بدست آوردن یک مقدار FP بکار برد که برای تولید برآورد قابل استفاده است. برنامه ریز پروژه با استفاده از داده های تاریخی یا(وقتی راه دیگری نباشد) با تجربیات شخصی خود یک مقدار اندازه خوشبینانه، محتملترین مقدار و مقدار بدبینانه برای هر عمل، یا مقدار دامنه اطلاعاتی را برآورد می کند.

سپس یک مقدار مورد انتظار را می توان محاسبه کرد . مقدار مورد انتظار برای متغیر برآورد(اندازه S،) ، را می توان به عنوان میانگین برآورد خوشبینانه(S opt) ، محتملترین برآورد(S m) و برآورد بدبینانه (S b) محاسبه کرد.برای مثال،

$$S = (S\ opt + 4\ S\ m + S\ b) / 6$$

بیشترین اعتبار را به محتملترین برآورد می دهد . ولی با این حال نمی توان مطمئن بود که برآوردها درست هستند.

۶-۲انجام برآورد موفق

برآورد در هر موضوعی مستلزم داشتن آگاهی جامع و کامل در باره آن موضوع است و این برای نرم افزار هم صدق می کند. مثلا وقتی می گوییم کار لازم برای توسعه یک نرم افزار ۱۰۰ نفر – ماه است آیا مراحل تحلیل طراحی و تست نرم افزار کاملا شناخته شده اند؟ اگر تخمین بدون دانستن یکی از اینها انجام شود دردسر بزرگی ایجاد خواهد شد.

برای انجام یک برآورد راههای مختلفی وجود دارد و اگر همواره در ذهن داشته باشیم که برای چه برآورد را انجام می دهیم می توان برآورد ساده تر و بهتری انجام داد. انجام برآورد قبل از شروع به انجام پروژه نرم افزاری علاوه بر اینکه به مدیریت درست نرم افزار کمک می کند فایده دیگری هم دارد اینکه آیا اصلا ساختن چنین نرم افزاری با این مشخصات ، هزینه و کار در این مدت زمان به صرفه است یا نه. شاید بتوان همین سیستم را با هزینه کمتری خرید. فرض کنید هدف از برآیند تهیه نرم افزار مورد نیازتان است برای خریدسیستم ۱۰۰ هزار دلار لازم است، برآورد در باره پیچیدگی سیستم، اندازه سیستم، و توانایی پرسنل انجام می گیرد و مشخص می شود که هزینه ساخت نرم افزار ۱۳۰ هزار دلار است نتیجه گیری این است که خرید نرم افزار تصمیم بهتری است .

علاوه بر بودجه و زمانبندی و استفاده از تحلیلهای خرید یا ساخت در تکنیکهای برآورد نرم افزار نکات دیگری برای کمک به تصمیم گیری وجود دارد:[۵]

- از قراردادها و تحلیلهایی برای برقراری تعادل میان هزینه، زمانبندی، کیفیت و کارایی و سرعت اجرا استفاده شود
- میزان سود و برگشت سرمایه در نظر گرفته شود.
- ریسک هزینه و زمانبندی و طراحی و تصمیمهای مدیریتی را در نظر گرفته شود.
- به کیفیت نرم افزار و بهبود بخشیدن به نرخ تولید نرم افزار توجه شود.

نکته بعدی توجه به طبیعت پویای رشته نرم افزار است. یک پروژه نرم افزاری به تدریج تکامل می یابد و به این تر تیب نمی توان یک برآورد قطعی و روشن برای نرم افزار ها داشت. در حال حاضر نرم افزار ها و مدلهای برآوردی وجود دارند که به جای یک برآورد نقطه ای برآورد حدودی انجام می دهند و به این تر تیب با متد های جدید تولید نرم افزار هم خوان می شوند.

۷-۲خطاهای مهلک در انجام برآورد

نکاتی وجود دارند که توجه نکردن به آنها یک برآوردخطا یا نامطلوب را بوجود می آورد در ادامه این نکات توضیح داده شده اند:[۴]

- **پیچیده و نامبهم کردن اهداف:**برای مثال نرم افزار های صنعتی کارها مختلفی را انجام می دهند و اهداف مختلفی را تعقیب می کنند که اغلب آنها بر پایه کارهای انجام شده هستند و بنابراین معمولا برآورد واقعی به میزان کمی انجام می شود.
- **بین چگونگی رسیدن به اهداف و برآورد تمایز قائل شوید** . زمانی که از شما خواسته می شود تا برآوردی انجام دهید مشخص کنید که خواسته آنها برآورد است یا از شما نقشه رسیدن به هدف را می خواهند.
- **موافقت، زمانی که واقعا مخالف هستید.** چرا توسعه دهندگان به سادگی موافقت می کنند؟ واقعا کار سختی است که یک برآورد فعال، درست و عاری از ریسک های شغلی انجام شود آنهم بدون استفاده از روشهای قابل اعتماد، با اطلاعات کم که به نظر مدیران کامل باشد.
- **انجام زود هنگام برآورد در زمانی که انجام برآورد غیر ممکن است**. بهتر است در چند زمان انجام شود تا به نتیجه واقعی نزدیکتر شویم.
- **برآورد کمتر از حالت واقعی، اثر زیادی روی نتیجه دارد.**باعث تاثیر غیر خطی روی میزان error ها و Defect ها دارد و در عمل موجب ریسک بالایی می شود.

- **برآورد زمان غیر ممکن**. زمانی که از زمانهای محاسبه شده توسط تمامی متد های برآورد کمتر باشد. سعی می شود تعادلی بین نیروی کار و زمان بوجود آید هیچ کس فکر نمی کند که نمی توان چنین تعادلی را بوجود آورد.
- **برآورد بیشتر از اندازه واقعی باعث می شود آمادگی برای ابزار و متد های جدید پیدا کنید**.مسائلی که ابزار و متدهای جدید دارند: هزینه آموزش، ریسک را بالا می برند، ماکزیمم کارایی در اولین استفاده مشخص نمی شود و نرخ تولید محصول کم می شود.
- **تنها استفاده از یک تکنیک برآورد**. بهتر است از روشهای مختلف و تجربه های شخصی استفاده شود.
- **استفاده نکردن از نرم افزارهای برآورد**. نرم افزار ها علمی تر هستند و برآورد هایی که انجام می دهند قابل اعتماد تر از برآورد های دستی هستند.
- **در نظر نگرفتن تاثیر ریسک روی برآورد**. پروژه های نرم افزاری حتما همراه ریسک هستند.
- **برآورد بدون فکر**.برای اینکه چنین اتفاقی نیفتد این موارد را در نظر داشته باشید:

  - با عمل برآورد مثل یک پروژه کوچک رفتار کنید
  - رویه های استانداردی برای برآورد کردن تعریف کنید
  - یک شرح روشن از برآوردی که انجام می شود داشته باشید
  - استفاده از چند مدل برآورد
  - برآورد نقاط از پیش تعریف شده در پروژه
  - استفاده از داده های تاریخی.بیشتر تکنیکهای برآورد از مقایسه پروژه های گذشته و با استفاده از حافظه افراد بوجود آمده اند
  - برآورد های بزرگ را به برآورد های کوچکتر تبدیل کنید
  - کل سیستم به پیمانه هایی تقسیم شود
  - Task های بزرگ به کوچکتر تقسیم شوند.

۲-۸ مروری بر استفاده کاربردی از برآورد و مدلهای برآورد

گزارش های پروژه های انجام شده نشان می دهد که تقریبا هیچ کنترلی روی پروژه های نرم افزاری وجود نداردو غالبا میزان کارانجام شده برای توسعه یک پروژه نرم افزاری بیشتر از مقدار برآورد شده است و بنا براین معمولا پروژه ها دیر تر از زمان برنامه ریزی شده به پایان می رسند پس شکی نیست که SCE (برآورد هزینه نرم افزار) مسئله ای بسیار جدی است. به این ترتیب که باید این سوالات ساده پاسخ داده شوند: چه مقدار زمان و هزینه برای انجام پروژه لازم است و عوامل تاثیر گذار بر هزینه چه هستند؟ متاسفانه پاسخ دادن به این سوالات کار ساده ای نیست.

مدلهای برآورد هزینه نرم افزار در حقیقت تکنیک های مدیریت پروژه هستند که برای کنترل کار و زمان انجام پروژه نرم افزاری استفاده می شوند.

۲-۹ مروری بر هزینه های توسعه نرم افزار

چیزی که اغلب اتفاق می افتد این است که نرم افزار گرانتر از آنی است که برآورد شده، دیرتر از زمان برآورد شده پایان می یابد، و سرانجام هم به تمامی اهداف اولیه اش نمی رسد.مطالعه وضعت حاضر انجام برآورد و کنترل توسعه پروژه های نرم افزاری در ۵۹۸ سازمان نشان می دهد:[۸]

- ۳۵٪ از سازمانها هیچگونه برآوردی انجام نمی دهند
- ۵۰٪ از سازمانها هیچگونه اطلاعاتی در باره پروژه های انجام شده شان ثبت نمی کنند
- ۵۷٪ محاسبه هزینه ها را انجام نمی دهند
- ۸۰٪ از پروژه های انجام شده دیرتر از زمان مقرر پایان می یابد و با کسری بودجه مواجه می شود.
- متوسط کسری بودجه و دیر کرد ۵۰٪ است.

۱۰-۲ چرا برآورد هزینه نرم افزار کار سختی است؟

دلایل کلی دشواری عمل برآورد هزینه به شرح زیر است:

۱. اطلاعات کمی در مورد پروژه های کامل شده وجود دارد. این قبیل اطلاعات به مدیر پروژه برای انجام برآورد کمک می کند.

۲. انجام برآورد معمولا با عجله انجام می شود به این علت که قبل از مشخص شدن خواسته های پروژه به برآورد آن نیاز داریم بنابراین برآورد به سرعت برای پروژه هایی که به خوبی فهمیده نشده اند انجام می شود.

۳. داشتن یک برآورد کامل و قابل اطمینان بویژه در آغاز پروژه دشوار است، انجام تغییرات، سازگاری یافتن و اضافه شدن دامنه پروژه نیازهایی هستند که زمانبندی و بودجه بندی با آنها سازگار شوند.

۴. مشخصه های نرم افزار و توسعه نرم افزار هم بر میزان دشواری انجام برآورد اثر دارند. از جمله پیچیدگی، قابل فهم بودن فرآیند و محصول

۵. فاکتور هایی وجود دارند که روی میزان کار و زمان لازم برای توسعه نرم افزار اثر می گذارند که همان cost driver ها هستند. از قبیل میزان پیچیدگی محصول، میزان توانایی توسعه دهندگان و غیره در عمل شناسایی و مقدار دهی به این فاکتور ها کار سختی است.

۶. تغییرات سریعی که در فناوری اطلاعات و متدولوژی های توسعه نرم افزار رخ می دهد باعث می شود نتوانیم یک روش برآورد هزینه ثابت داشته باشیم. اندازه گیری تاثیرات حاصل از تکنولوژی های پیشرفته و جدید دشوار است.

۷. کسی که برآورد را انجام می دهد که اغلب هم مدیر پروژه است، تجربه زیادی در انجام برآورد برای پروژه های عظیم ندارد. مگر یک نفر در طول عمرش چند پروژه ده ساله را می تواند مدیریت کند؟

۸. پیش فرضی در ذهن همه هست که یک ارتباط خطی را بین تعداد افراد توسعه دهنده نرم افزار و مدت زمان توسعه بوجود می آورد به این معنی که اگر ۲۵ نفر کاری را در دو سال انجام دهند ۵۰ نفر آن را در یک سال انجام می دهند در صورتیکه این فرض نادرستی است و پیوستن افراد جدید به پروژه ای که عقب افتاده باعث تاخیر بیشتر می شود.

۲-۱۱ پیش نیازهای برآورد هزینه نرم افزار

یک سازمان برای بهینه سازی فرآیند مدیریت پروژه انتخاب های زیادی دارد مثل چگونگی تعیین مسئولیت ها، استفاده از ابزار ها و روشهای تصمیم گیری و نظارت بر انجام وظایف. برآورد نرم افزار از دیدی اجتماعی و روان شناسی به نرم افزار می نگرد. برای مثال هماهنگی تیمی، سبک رهبری و مانند آن. همچنین جنبه های تکنیکی کار را هم مورد توجه قرار می دهد مثلا چگونگی طراحی، برنامه نویسی، تست، مستند سازی و ابزار های نرم افزاری و سخت افزاری .
پیش نیاز های انجام برآورد اطلاعاتی است که برای انجام برآورد ضروری است. اطلاعاتی در باره:

- محصولی که قرار است ساخته شود
- ابزار تولید محصول
- پرسنل توسعه دهنده محصول
- سازمان توسعه دهنده محصول
- کاربران محصول، اینکه محصول برای چه کسی ساخته می شود

در حقیقت این سوالات پنج دسته کلی را تعیین می کنند که هر کدام شامل گروه خاصی از Cost Driver هاست تعداد اینها بسیار زیاد است و آنچه که اهمیت دارد این است که یک سازمان مشخص کند کدام Cost Driver ها در سازمان نقش دارند. برای هر کدام از این پنج دسته مثال هایی از مهمترین cost Driver ها به این تر تیب است:
محصول

- اندازه نرم افزار
- کیفیت مورد نیاز
- پیچیدگی محصول
- سطح قابلیت استفادگی مجدد
- میزان مستند سازی
- نوع برنامه کاربردی

ابزارها

- محدودیت های سخت افزار
  - زمان اجرا
  - زمان پاسخ
  - اندازه حافظه
- کاربران ابزارها
- استفاده از تکنیک های مدرن برنامه نویسی
  - سطح پنهان سازی اطلاعات
  - برنامه نویسی تیمی
  - برنامه نویسی ساخت یافته
  - طراحی بالا به پایین

پرسنل

- کیفیت پرسنل
- میزان شناخت از پرسنل
- مدیریت کیفیت

پروژه

- زمان توسعه پروژه
  - زمان به اندازه کافی
  - زمان در فشار

- روش کنترل پروژه
  - نمونه سازی
  - تکاملی
  - خطی

کاربران

- تعداد کاربران
- ثبات سازمان کاربران و نحوه کار با نرم افزار
- میزان آشنایی کاربران با اتوماسیون
- سطح آموزش اتوماسیون به کاربران

مسائلی که در شناخت و مقدار دهی Cost Driver ها وجود دارد به این تر تیب است:

تعریف- برای Driverها تعریف های پذیرفته کمی وجود دارد از جمله برای اندازه، کیفیت، پیچیدگی و شناخت.

عدم قطعیت- چیزی که برای یک توسعه دهنده پیچیده است ممکن است برای دیگری اصلا پیچیده نباشد.

وابستگی- کار سختی است که یک cost Driver را به تنهایی در نظر بگیریم تغییر در مقدار یک Cost Driver باعث تغییر در مقادیر دیگران می شود.

ارتباط میان driver ها و کار لازم برای توسعه پروژه- برای انجام برآورد لازم است که ارتباط میان آنها مشخص شودبرای مثال ارتباط میان کار و اندازه نرم افزار . اما در واقع هیچ ارتباط روشنی میان آنها وجود ندارد.

مشخص کردن Cost Driver ها – نمی توان قطعا مهمترین آنها را مشخص کرد و این مسئله در شرایط مختلف متفاوت است. **کارایی و بهره وری**– اغلب میان ایندو تداخل پیش می آید. کارایی توجه زیاد به انجام درست و با دقت کار است.

فاکتور های انسانی– تقریبا همه موافق تاثیر این فاکتور ها هستند مانند میزان تجربه و توانایی پرسنل توسعه دهنده نرم افزار به این معنی که مهارت در یک توسعه خوب اهمیت دارد.

استفاده مجدد- یکی از مهمترین فاکتور های کارایی است.

۱۲-۲ تکنیک ها و ابزار های برآورد هزینه نرم افزار

تکنیک های زیادی برای برآورد هزینه های نرم افزار وجود دارد. اغلب آنها ترکیبی از اصول اولیه زیر هستند:

۱. برآورد بر اساس تجربه
۲. برآورد بر اساس مقایسه و تحلیل
۳. برآورد بر اساس توانایی ها و ظرفیت های موجود
۴. برآورد بوسیله مدلهای پارامتری

و به دو روش کلی انجام می شود:
بالا به پایین- در این روش مشخصه های کلی پروژه تعیین می شود، برآورد کلی صورت می گیرد و سپس به کامپوننت های مختلف تجزیه می شود.
روش پایین به بالا- در این روش هر کامپوننت به طور مجزا بوسیله کسی که مسئوایت توسعه آن را دارد برآورد می شود، تمامی برآورد های مجزا جمع شده و برآورد کلی را می سازند.

۱. قابل اعتماد بودن یک برآورد به قضاوت تجربی بستگی دارد و به اینکه چه مقدار این پروژه با پروژه های قبلی تطابق دارد، و توانایی به خاطر داشتن و تجزیه و تحلیل و به کار بردن داده ها و حقایق قدیمی چقدر است. مهمترین مشکلی که در این روش برآورد وجود دارد این است که برای کس دیگری امکان اتفاده از این دانش تجربی وجود ندارد. با این وجود این روش اغلب در سازمانها و برای برآورد اولیه استفاده می شود.

۲. اساس برآورد بر تحلیل های مقایسه ای است- مقایسه پایگاه داده های قدیمی پروژه های مشابه یا ماژول های مشابه پروژه های دیگر، برای اینکه موارد مشابه پروژه ها را بدست آوریم لازم است که داده های مربوط به پروژه های قدیمی به دقت نگهداری شود.

۳. برآورد بر اساس توانایی ها و ظرفیت های موجود انجام می شود یعنی ابزاری که در دسترس است مخصوصا پرسنل. برای مثال اگر برای چهار ماه آینده فقط سه نفر موجود باشند پروژه ۱۲ نفر ماه در نظر گرفته می شود. اثرات جانبی این روش این است که حتی اگر زمان کمتری لازم می بود، پروژه همین مدت طول می کشد بنا به قانون پارکینسون که می گوید کار آنقدر طول می کشد تا تمام زمان در دسترس تمام شود.[1]

[1] software cost estimation , F J Heemstra

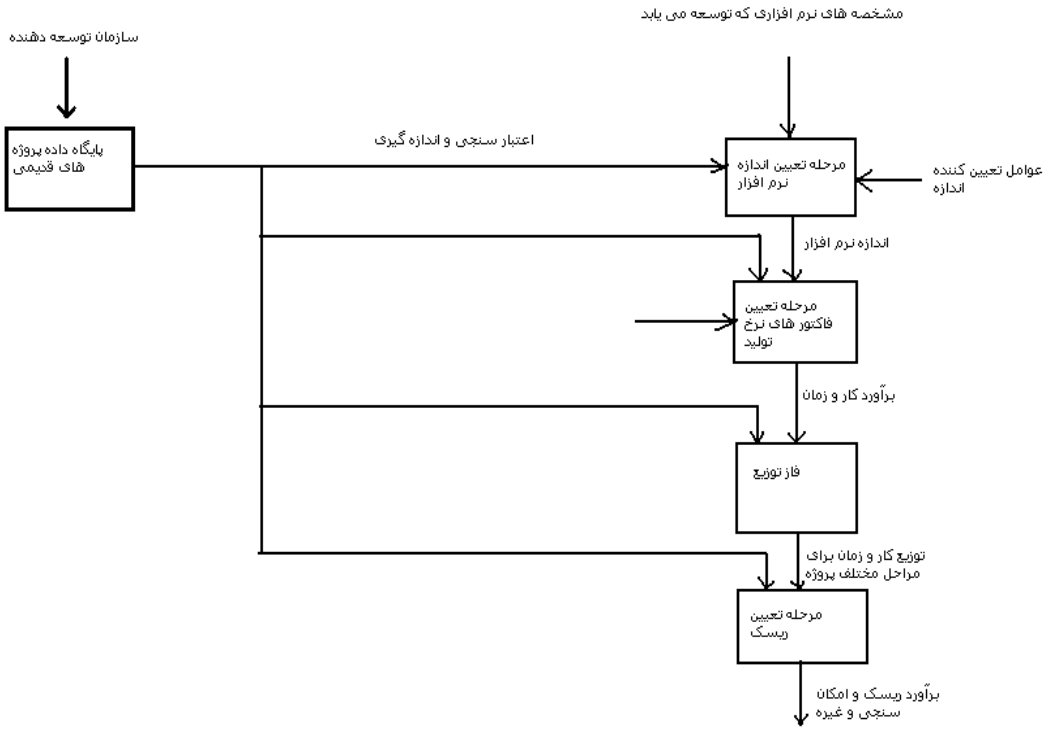
۴. در مدلهای پارامتری، زمان و کار بر اساس توابعی از تعدادی متغیر محاسبه می شوند این متغیرها مهمترین Cost Driver ها برای پروژه را نشان می دهند. هسته مدلهای برآورد تعدادی الگوریتم و پارامتر است. مقادیر پارامتر ها و الگوریتم ها از اطلاعات مربوط به پروژه های پیشین بدست آمده اند.

بررسی نشان می دهد که در اغلب (تقریبا ۶۰٪) سازمانهایی که برآورد انجام می دهند از عمل مقایسه و تحلیل بر اساس اطلاعات پروژه های تکمیل شده قدیمی استفاده می کنند و به نحوی اطلاعات مربوط به پروژه هایشان را نگهداری می کنند، ۲۵٪ از تجربه های شخصی و ۲۰٪ هم بر اساس ظرفیت های موجود برآورد انجام می دهند و تنها ۱۳٪ از روشهای پارامتری استفاده می کنند.

۱۳-۲ مدلهای برآورد هزینه نرم افزار

اغلب مدلهایی که امروزه استفاده می شوند مدلهایی دو مرحله ای هستند در مرحله اول اندازه پروژه تعیین می شود و در مرحله دوم فاکتور های نرخ تولید بدست می آیند. ابتدا اندازه پروژه ای که توسعه خواهد یافت بدست می آید. برای اینکار تکنیک های زیادی وجود دارند اما تکنیک های شناخته شده ای که امروزه استفاده می شوند نقطه عملکرد و خط کد هستند. نتیجه این مدلهای اندازه گیری نرم افزار تعداد خط کدها و یا تعداد دستور ها و یا تعداد نقطه عملکرد هاست.

در مرحله اول نیروی کار بر حسب نفر ماه بر اساس اندازه برآورد شده محاسبه می شود و در مر حله دوم فاکتور های نرخ تولید اضافه می گردند. برخی مدلهای برآورد نرم افزار بر اندازه تاکید دارندو برخی مثل COCOMO روی فاکتور های نرخ تولید بوسیله Cost Driver ها تاکید می کنند.


مشخصه های نرم افزاری که توسعه می یابد
سازمان توسعه دهنده
پایگاه داده پروژه های قدیمی
اعتبار سنجی و اندازه گیری
مرحله تعیین اندازه نرم افزار
عوامل تعیین کننده اندازه
اندازه نرم افزار
مرحله تعیین فاکتور های نرخ تولید
برآورد کار و زمان
فاز توزیع
توزیع کار و زمان برای مراحل مختلف پروژه
مرحله تعیین ریسک
برآورد ریسک و امکان سنجی و غیره

در این دیاگرام پنج مرحله اساسی برآورد هزینه نرم افزار دیده می شود. بعد از تعیین اندازه و اثر دادن Cost Driverها قاز توزیع و سپس تحلیل ریسک انجام می شود در مرحله توزیع، نیروی کار و زمان کل بدست آمده به فعالیت های جزئی پروژه تقسیم می شود. چگونگی این تقسیم بندی به اطلاعات پروژه های قبلی بستگی دارد. مرحله تحلیل ریسک بخصوص در آغاز پروژه که ابهامات زیادی وجود دارد، به مدیر پروژه کمک می کند تا فاکتور های ریسک و چگونگی Cost Driver ها را تعیین کند. داده های مربوط به پروژه های قبلی برای این مولفه هم اهمیت زیادی دارد. وقتی برای اولین بار از یک مدل برآورد استفاده می شود باید مدل اعتبار سنجی و اندازه گیری شود، از این جهت که مشخصه های پروژه و محیط با محیط و پروژه هایی که مدل از روی آنهامحاسبه شده چقدر تفاوت دارد برای اینکه بتوان عمل اعتبار سنجی و اندازه گیری را انجام داد لازم است که داده های قدیمی پروژه های سازمان در دسترس باشند. به این ترتیب برای یک برآورد می توان ۷ مرحله را در نظر گرفت البته تمامی مدلهای برآورد شامل تمامی این مراحل نیستند.

۱. تولید پایگاه داده ای از پروژه های کامل شده
۲. برآورد اندازه
۳. برآورد نرخ تولید
۴. فاز توزیع
۵. تحلیل ریسک و حساسیت پروژه
۶. اعتبار سنجی
۷. اندازه گیری

۲-۱۴ مروری بر برخی مدلهای برآورد هزینه

Function point analysis(FPA)ّ

از آنجایی که در این نرم افزار از این روش برای تعیین اندازه استفاده می کند قبلا در بخش تعیین اندازه این روش را توضیح دادیم.

PRICES

مهمترین عیبی که این سیستم در مقایسه با FPA و COCOMO دارد این است که فرض ها و دانشی که سیستم بر اساس آن کار می کند به طور عمومی تعریف نشده و کاربران با یک جعبه سیاه[1] مواجه اند. کاربران اطلاعات خود را به یک کامپیوتر در آمریکا، فرانسه یا انگلستان می فرستند و به سرعت نتیجه را در یافت می کنند. با وجود این اشکال و همچنین با اینکه استفاده از این سیستم هزینه نسبتا زیادی دارد ، این سیستم کاربران زیادی بویژه در آمریکا دارد.

Before You Leap

یک بسته تجاری است که از ادغام FPA و COCOMO بدست آمده. طریقه کار بدین صورت است که ابتدا تعداد نقاط عملکرد محاسبه می شود و سپس به SLOC ترجمه می شودمیزان SLOC به زبانی که نرم افزار با آن نوشته می شود بستگی دارد. این برآورد اندازه، ورودی قسمت کوکوموی BYL است. تاثیر Cost Driver ها در این قسمت بر کار محاسبه می شود.

Estimacs

این مدل بصورت یک بسته نرم افزاری در دسترس است.این مدل در حقیقت شامل چند ماژول است:ماژول FPA، ماژول ریسک، ماژول کار که کار لازم برای توسعه و نگهداری سیستم را محاسبه می کندو غیره. مهمترین ماژول، کار است. کاربر به بیست و پنج سوال جواب می دهد که در باره پیچیدگی های سازمان و اندازه نرم افزار است. اما روشی که این ورودی ها به اندازه کار ترجمه می شوند واضح نیست و بنابراین Estimacs یک مدل بسته است.

BIS Estimator

این مدل کاملا با مدلهای قبلی متفاوت است.آنطوری که مستندات نرم افزار نشان می دهنداین مدل قرار است بر اساس یک پایگاه دانش باشد. این موضوع کاملا پذیرفته نشده زیرا بسیاری از اصول اولیه مدل پنهان نگه داشته شده اند. در این مدل ابتدا برآوردی از نیروی کار و زمان پروژه بوسیله یکسری

[1] Black box

اطلاعات ورودی بدست می آید که به این مرحله اصلاح "نرم" [1]را داده اند، سپس برای هر مرحله یک برآورد "سخت"[2] انجام می شود. مرحله سخت در ابتدای هر مر حله و طی انجام آن مرحله پروژه انجام می شود.مهمترین مزیتی که نرم افزار دارد این است که طی انجام پروژه برآورد می تواند تغییر کند، بر اساس اطلاعات جدیدی که از پروژه بدست آورده ایم و یا بر اساس تغییراتی که در توسعه نرم افزار بوجود آمده.

۱۵-۲ مدل برآورد کوکومو

کوکومو مستند ترین و شفاف ترین مدل برآورد هزینه موجود است. در مدل کوکوموروی تاثیر پانزده Cost Driver بر کار لازم برای انجام پروژه تاکید می کند. کوکومو برای اندازه گیری کار و زمان یعنی ارتباط میان اندازه و Cost Driverها و کار همینطور میان کار و زمان از یکسری فرمول استفاده می کند که از داده های تاریخی پروژه های تکمیل شده بدست آمده اند، سپس اثر Cost Driverها روی کار بدست می آید.

تمامی مدلهای برآورد نرم افزار معادله هایی را ارائه می دهند که پایه محاسبه هزینه، کار و زمان در نرم افزارهای برآورد است. این معادله ها اغلب شکل واحدی دارند و تنها در ضرایب معادله ها متغیرند. نتایجی که از این معادلات بدست می آیند ممکن است حتی کمی متفاوت باشند ولی این موضوع عجیب نیست چون متغیرهای محلی بیشماری روی یک برآورد موثر هستند و باید بتوانیم معادله های مناسب را تشخیص و انتخاب کنیم!

این واژه مخفف عبارت Constructive Cost Model است.مدل اولیه کوکومو یکی از پر کاربرد ترین و بحث انگیز ترین مدلهای برآورد هزینه در صنعت بوده است.

مدل برآورد کوکومو که هزاران مدیر پروژه های نرم افزاری از آن استفاده می کنند برپایه مطالعه صدها پروژه نرم افزاری بوجود آمده است.بر خلاف دیگر مدلهای برآورد کوکومو یک مدل باز است به این معنی که کلیه جزئیات این مدل در اختیار همه قرار دارد. این جزئیات شامل موارد زیر هستند:

- معادلات برآورد هزینه
- تمامی فرضیاتی که در ساخت مدل در نظر گرفته شده اند.

نرم افزارهایی که ازکوکومو برای برآورد استفاده می کنند این مزایا را برای کاربرانشان فراهم می کنند:

[1] Soft
[2] Hard

- برآوردهای بر پایه کوکومو بسیار عینی تر و تکرار پذیر تر از برآورد هایی هستند که با مدلهای دیگر بدست می آید.
- کوکومو می تواند بنحوی تنظیم شود که بازتابی از محیط توسعه نرم افزار کاربر باشد تا برآوردهای دقیق تری انجام شود.

سامرویل در کتاب مهندسی نرم افزارش کوکومو را در سه شکل بنیادی ، متوسط و مفصل تعریف کرده است. دیدی کلی از مدلهای بنیادی و متوسط کوکومو برای مدلسازی الگوریتمی هزینه ارائه می شود. در مدل کوکومو فرض می شود که نیازمندیهای نرم افزار نسبتا پایدارند و پروژه به خوبی توسط مشتری و سازندگان نرم افزار اداره می شود.

مدل بنیادی کوکومو مقدار برآورد شده هزینه های نرم افزار را به دست می دهد. در این مدل از اندازه برآورد شده پروژه نرم افزاری و نوع نرم افزار در حال توسعه استفاده می شود. همچنین سامرویل پروژه های نرم افزاری را به طبقات مختلفی تقسیم کرده و برای هر طبقه فرمولها ی برآوردی پیشنهاد کرده.

- **طبقه اول**. پروژه هایی هستند که در آنها تیمهای نسبتا کوچک در یک محیط آشنا کار می کنند و کاربردهای شناخته شده ای را توسعه می دهند. بار اضافی ارتباطی کم است، اعضای تیم به آنچه که انجام می شود واقفند و می توانند به سرعت با کارها کنار بیایند.
- **طبقه دوم**. این حالت، مرحله ای بین پروژه های طبقه اول و پروژه های طبقه سوم را ارائه میدهند. دراین پروژه ها ممکن است تیم پروژه از کارمندان کارآزموده و بی تجربه تشکیل شود. ممکن است اعضای تیم تجربه محدودی درباره سیستمهای مرتبط داشته باشند و با برخی جنبه های سیستم در حال توسعه نا آشنا باشند.

فرمول محاسبه هزینه های مربوط به پروژه یا بطور دقیقتر، کار مورد نیاز برای توسعه نرم افزار بطور کلی دارای شکل زیر است:

$$\text{کار} = A\ (KDSI)\ \hat{}\ b$$

KDSIعبارت است از تعداد هزار دستورالعمل منبع تحویل شده. A و b ثابتهایی هستند که بسته به نوع پروژه تغییر می کند. از این معادله چنین بر می آید که هزینه، تابعی نمایی از اندازه است، هرچند مقدار b معمولا نزدیک به یک است.

و DSI ، هر خطی از متن منبع، بدون در نظر گرفتن تعداد واقعی دستوالعمل ها در آن خط است. بنا براین اگر دو یا چند دستوالعمل در یک خط وجود داشته باشد، به عنوان یک DSI به شمار می روند و اگر دستوری در پنج خط پراکنده شود به عنوان پنج DSI در نظر گرفته می شود.

مقادیری که در هر وضعیت خاص به این ثابتها نسبت داده می شود، باید با تحلیل داده های تاریخی پروژه تعیین گردد. به کارگیری داده های حاصل از سازمانهای دیگر، باعث می شود خطای مدل را افزایش یابد چون فرض های عملیاتی سازمانها با هم متفاوت است. برای مثال سامرویل با استفاده از داده های سازمان خود، فرمولهای زیر را برای طبقات متفاوت پروژه ها گزارش کرده است.

طبقه اول: $PM= 2.4 (KDSI) ^ 1.05$

طبقه دوم: $PM= 3 (KDSI) ^ 1.12$

نتیجه این برآورد، تعداد نفر – ماههای مورد نیاز برای تکمیل پروژه است. در اکثر نرم افزار های خودکار برآورد هزینه و دربیشتر مدلها ی کوکومو یک نفر-ماه به صورت ۱۵۲ ساعت کاری تعریف شده است.

مدل بنیادی کوکومو برای برآورد زمان توسعه پروژه نیز معادلاتی پیشنهاد می کند. معادلات زمان توسعه برای حالتهای متفاوت پروژه عبارتند از:

طبقه اول: $TDEV= 2.5 (PM) ^ 0.38$

طبقه دوم: $TDEV= 2.5 (PM) ^ 0.35$

تعداد افراد به کار گرفته شده در پروژه در طول زمان توسعه آن یکنواخت نیست و بنا براین برای بدست آوردن تعداد افراد مورد نیاز برای انجام پروژه نمی توان کار را بر زمان توسعه تقسیم کرد.تعداد بهینه کارمندان یک پروژه از یک منحنی پیروی می کند. بسته به آهنگ جذب کارمندان، چند منحنی متفاوت وجود دارد.

در آغاز پروژه تنها تعداد کمی از افراد برای طراحی پروژه مورد نیاز است. با پیشرفت پروژه و نیاز به کار بیشتر، تعداد کارمندان بیشینه می شود. پس از آنکه کار به پایان رسید و بررسی واحد تمام شد، تعداد کارمندان نزول می کند تا هنگام تحویل محصول، به یک یا دو نفر می رسد.

تجربه نشان داده که رشد بسیار سریع تعداد کارمندان باعث عدم توازن در زمانبندی پروژه می شود. بنابراین پیشنهاد می شود که مدیران پروژه نباید سعی کنندر ابتدای طول عمر پروژه ، تعداد بیش از حد کارمندان را به کار گیرند.

یکی از نتیجه های جالب مدل کوکومو این است که زمان مورد نیاز برای کامل ساختن پروژه، تابعی از کل کار مورد نیاز برای انجام پروژه است و تابعی از تعداد مهندسان در حال کار روی پروژه نیست.این نکته نشان می دهد که افزودن تعداد بیشتری از افراد به پروژه ای که از زمانبندی عقب افتاده است، احتمالا کمکی به جبران آن نمی کند.

۱-۱۵-۲ مدل میانی کوکومو

مدل بنیادی کوکومو برآورد هزینه نرم افزار را تنها بوسیله اندازه و نوع پروژه انجام می دهد ، ولی غیر از اندازه و نوع پروژه، عوامل فراوان دیگری وجود دارند که میزان کار انجام شده در طول پروژه را تحت تاثیر قرار می دهند.در مدل میانی کوکومو برخی از این عوامل به حساب آورده می شوند.

در مدل میانی کوکومو هم ابتدا اندازه و نوع پروژه تعیین می شود سپس یکسری از ضرایب بدست می آیند که به ارقام بنیادی کوکومو اعمال می گردد ، عواملی نظیر قابلیت اطمینان مورد نیاز برای محصول، اندازه بانکهای اطلاعاتی، محدودیتهای اجرایی و ذخیره سازی، ویژگیهای پرسنلی و کاربرد ابزار های نرم افزاری. در کتاب سامرویل پانزده عامل شمرده شده که می توان به حساب آورد. این عوامل به چهار دسته تقسیم می شوند: صفتهای محصول، صفتهای کامپیوتر، صفتهای پرسنلی و صفتهای پروژه.

صفتهای محصول به ویژگیهای مورد نیاز محصول نرم افزاری در حال توسعه مربوط می شود. این صفتها در مقیاسی از مقادیر بسیار کم گرفته تا مقادیر بسیار بالا درجه بندی می شود. مقادیر کم نشانگر ضریب کوچکتر از ۱، مقادیر جزئی ضریب ۱، و مقادیر بالا ضریب بزرگتر از ۱ را نشان می دهند.

- **قابلیت اطمینان** قابلیت اطمینان پایین بدان معناست که خطای نرم افزاری منجر به ناراحتی کمی می شود؛ قابلیت اطمینان جزئی بدان معناست که خطا منجر به ضایعات معتدل و قابل جبران می شود؛ قابلیت اطمینان بالا به معنای این است که خطا شامل خطر جانی برای انسان می شود.
- **اندازه بانکهای اطلاعاتی** مقدار کم این صفت نشانگر این است که اندازه بانک اطلاعاتی کمتر از ۱۰ برابر تعداد DSI هاست؛ مقدار متوسط این صفت نشان می دهد که اندازه بانکهای اطلاعاتی بین ۱۰ تا ۱۰۰ برابر اندازه سیستم است؛ مقدار بسیار زیاد آن نشانگر این است که بانکهای اطلاعاتی ۱۰۰۰ برابر بزرگتر از برنامه است.
- **میزان پیچیدگی محصول** محصولاتی که از پیچیدگی کم برخوردارند، از عملیات I/O ساده، ساختمان داده ساده و کد ساده استفاده می کنند. میزان پیچیدگی متوسط، نشانگر برخی از پردازشهای I/O ، ورودی یا خروجی چند فایلی، استفاده از روالهای کتابخانه ای است. میزان پیچیدگی بسیار بالا به معنای استفاده از کدهای بازگشتی، مدیریت فایلهای پیچیده، پردازش موازی، مدیریت داده های پیچیده و غیره است.

صفتهای کامپیوتری محدودیتهایی اند که سخت افزار بر نرم افزار اعمال می کند. مثالهای این محدودیتها عبارتند از اندازه حافظه یا سرعت پردازش محدود. این عوامل بهره وری نرم افزار را تحت تاثیر قرار می دهند زیرا نرم افزار باید با این محدودیتها ی سخت افزاری سازگار باشد.

- **محدودیتهای زمان اجرا** درجه متوسط بدان معناست که کمتر از ۵۰٪ زمان اجرای در دسترس، مورد استفاده قرار می گیرد و درجه بسیار بالا بدان معناست که ۹۵٪از زمان در دسترس را باید مورد استفاده قرار داد.
- **محدودیت حافظه** مقدار متوسط بدان معناست که کمتر از نصف حافظه در دسترس، مورد استفادهقرار می گیرد و در جه بسیار بالا نشانگر آن است که ۹۵٪ از حافظه در دسترس مورد استفاده قرار می گیرد.
- **تغییر ناگهانی ماشین مجازی** ماشین مجازی تلفیقی از سخت افزار و نرم افزاری است که محصول نرم افزاری روی آن ساخته می شود. مقدار کوچک این ضریب بدان معناست که این ماشین فقط گاهی تغییر داده می شود(یکبار در سال) ، مقدار متوسط، نشانگر تغییرات عمده در هر شش ماه یکبارو مقدار بسیار بالای این ضریب بیانگر تغییرات مکرر ماشین مجازی است.
- **زمان برگشت کامپیوتر** این عامل از مقدار کم شروع می شود که بیانگر توسعه سیستمهای محاوره ای است و به مقدار بسیار بالا منتهی می شود که نشان می دهد زمان برگشت بیش از ۱۲ ماه است . اکنون توسعه بیشتر سیستمها با استفاده از سیستمهای اشتراک زمانی یا شبکه های ایستگاههای کاری صورت می گیرد، بطوریکه این ویژگی اکنون زاید به نظر می رسد .

پنج صفت پرسنلی به حساب آورده می شود که تجربیات و قابلیتهای کارمندان در حال کار پروژه را منعکس می سازند. این صفتها عبارتند از : قابلیت تحلیل گر، تجربه کاربردی، تجربه ماشین مجازی، قابلیت برنامه نویس و تجربه زبان برنامه نویسی. همه این صفات از مقدار بسیار کم (که نشانگر تجربه کم یا عدم تجربه است) به مقدار متوسط (که نشانگر حداقل یک سال تجربه) و مقدار بسیار بالا (که نشانگر بیش از سه سال تجربه است) سنجیده می شود.

صفتهای پروژه به استفاده از ابزارهای نرم افزاری، زمان توسعه نرم افزار و کاربرد اعمال برنامه نویسی مدرن مربوط می شوند. اعمال برنامه نویسی مدرن به عنوان اعمالی نظیر طراحی بالا به پایین ، طراحی و بازبینی کد، برنامه نویسی ساخت یافته، کتابخانه های پشتیبان برنامه و غیره تعریف می شود.

هنگامی که مدل کوکومو معرفی می شد، کاربرد برنامه نویسی ساخت یافته زیاد شناخته شده نبود و ابزارهای نرم افزاری مانند امروز در دسترس نبودند. اکنون استفاده از ابزار، گسترش زیادی یافته و استفاده از تکنیکهای ساخت یافته امری متداول است؛ بنابراین شاید صفتهایی که در آغاز تعریف شده

اند، دیگر اهمیتی نداشته باشند. به هر حال صفت زمانبندی هنوز هم اهمیت زیادی دارد. این صفت میزانی از نحوه تطابق زمان توسعه مورد نیاز با زمان متوسط برآورد شده به وسیله مدل کوکومو است اگر لازم باشد توسعه در زمان کمتری از زمان برآورد شده انجام گیرد این ضریب مقدار مثبت می گیرد.

مدلهای بنیادی و میانی کوکومو این عیب را دارند که محصول نرم افزار ی را یک کل در نظر می گیرند و کلیه ضرایب را به آن اعمال می کنند، در صورتیکه یک محصول ممکن است از قسمتهای فرعی هم تشکیل شده باشدکه برخی از طبقه اول و برخی از طبقه دوم و غیره باشند.

در مدل کامل کوکومو این نکته در نظر گرفته می شود و هزینه های کل سیستم به صورت حاصل جمع هزینه های زیر سیستمها برآورد می شود به طوری که هر یک از زیر سیستمها به طور جداگانه برآورد می شوند. این روش خطای موجود در برآورد نهایی را کاهش می دهد.

برای استفاده موثر از مدل برآورد، باید آن را برای شرایط و اوضاع محلی تنظیم کردسامرویل صفتهای جامعی را در کتابش پیشنهادکرده ولی صفتها انحصاری نیستند برخی از سازمانهای دیگر باید از صفتهای دیگری به عنوان ضرایب هزینه استفاده کنند. به عنوان مثال در دسترس بودن کتابخانه ای از قطعات نرم را می توان به حساب آورد.

بسته به حالتهای خاص در یک سازمان ، تغییر صفتهای مدل میانی کوکومو یا افزودن صفتهای جدیدی که اهمیت دارند امکان پذیر است. مثلا، صفتهای پرسنلی را می توان یک صفت کلی در نظر گرفت. صفتهای جدیدی که می توان اضافه کرد عبارتند از : اثر کار با داده های طبقه بندی شده و ایمنی همراه با آن و صفتی که نوع کاربرد در حال توسعه را مشخص کند.

مقادیر این ضرایب را می توان با برآورد هزینه های محصول، بدون استفاده از یک ضریب خاص، سنجش هزینه های واقعی و سپس یافتن بهترین ضرایب از داده های اندازه گیری شده مقدار های برآورد شده ، محاسبه کرد. سازمانهای معدودی وجود دارند که زمان کافی برای انجام آزمایش دارند تا بتوانند مقادیر را اندازه گیری کنند.

مشکل مدل کوکومو و مدلهای الگوریتمی دیگر برای برآورد هزینه پروژه این است که امکان خطا خیلی زیاد است.برآورد اندازه محصول کار بسیار دشواری است؛ مقادیر ثابت مقادیر مشخصی نیستند بلکه میانگین اند و غیره. برآورد میزان خطا نیز به همان اندازه دشوار است و بنابراین نمی توان به راحتی به مقادیر برآورد شده اطمینان کرد.

به هر حال با این فرض که خطاها نسبتا ثابت اند، مدلهای الگوریتمی هزینه یابی را می توان برای محاسبه هزینه های نسبی پروژه به کار بردو به این ترتیب این تکنیک های برآورد یک راه کمی برای کنترل انجام یک پروژه خاص را فراهم می کنند و این باعث می شود بتوان از منابع استفاده منطقی و بهینه کرد و خطر را کاهش داد.

نرم افزار های برآورد فراوانی وجود دارند واین یعنی مدیر پروژه ای که از مدل هزینه یابی الگوریتمی استفاده می کندمی تواند راههای متفاوت انجام پروژه را ببیند و سپس درباره مناسبترین راه تصمیم

بگیرد. این امر بویژه هنگامی اهمیت می یابد که هزینه های سخت افزار و نرم افزار متعادل شوندو نیاز به استخدام کارکنان جدید با مهارتهای خاص باشد.
اینطور نیست که که کم هزینه ترین انتخاب، بهترین انتخاب باشد. به عنوان مثال ممکن است کارمندان در صورت بکار گرفته نشدن در پروژه بیکار می مانندو هزینه اضافی برای سازمان دارند، بکارگیری این کارمندان می تواند بهتر از سرمایه گذاری روی سخت افزار جدید باشد.
با افزایش هزینه های نرم افزاری از اهمیت هزینه های سخت افزاری کاسته می شودو با کاهش آن بر اهمیت هزینه های سخت افزاری افزوده می گردد. مدیر کارامد پروژه همواره کم خطر ترین راه را برای توسعه پروژه نرم افزار انتخاب می کند.

۲-۱۶ مقایسه مدلهای برآورد هزینه نرم افزاری

مدلهای برآورد تفاوتهای قابل ملاحظه ای با هم دارند و تجربه نشان داده یک برآورد حتما باید با چند مدل برآورد انجام شود. ولی با همه اینهااغلب برآورد ها با کار و زمان توسعه واقعی بسیار فاصله دارند. برای اینکه بتوانیم یکسری اصول کلی درباره مدلهای برآورد داشته باشیم باید نیازمندیها و خواسته هایی را که از یک مدل برآورد داریم مشخص کنیم.این نیازمندیها به شکل زیر مرتب شده اند:

- مشخصات مدل برآورد
  - به روش کنترل نرم افزار مربوط باشد
  - در شروع پروژه کاربرد داشته باشد
  - به اطلاعاتی که در حین توسعه نرم افزار بدست می آید هم خوان باشد
  - برآورد امکان تغییر در آبجکت ها را بدهد
  - برای تعریف دامنه نرم افزار مناسب باشد
- مشخصات برنامه کاربردی
  - امکان اندازه گیری
  - دقت برآورد
- مشخصات پیاده سازی
  - استفاده از ابزار ها برای کاربران ساده باشد
  - امکان تحلیل ریسک
  - مدل باز باشد- بدانیم نتایج چگونه بدست آمده اند
  - ورودی های روشنی داشته باشد
  - خروجی کامل و با جزئیات باشد

از این موارد می توان نتیجه گیری کرد که کیفیت مدلها پایین است و احتیاج به بهبود دارند. میزان دقت مدلها قابل اندازه گیری است برای اینکار از تست های مختلفی می توان استفاده کرد. تست مدلها در دو زمینه انجام می گیرد:

۱.مشخص کردن دقت مدلها در یک محیط واقعی
۲.مشخص کردن اینکه مدیران پروژه چقدر مدل را پذیرفته اند

برای تست مدلها کلیه موارد بالا در چهارده پروژه برای مدلهای گفته شده در نظر گرفته شده و در این میان تنها دو مدل BYL و Estimacs نتایج قابل قبولی داشتند.[۸]

در حقیقت مدلهای برآورد در سازمانها پذیرفته نشده اند. یک گزارش آماری نشان داداز ۳۶۴ سازمانی که عمل برآورد را برای پروژه هایشان انجام می دهند تنها ۵۱ سازمان از مدلهای برآورد استفاده می کنندو تحلیلها نشان می دهند این ۵۱ سازمان برآورد های بهتری نسبت به سازمانهای دیگر ندارند زیرا با کیفیت پایینی مورد استفاده قرار می گیرند. [۸] برای مثال استفاده از مدلها احتیاج به اطلاعات مربوط به پروژه های گذشته سازمان دارد. بسیاری از مدلها بدون توجه به اندازه گیری فرض های مدل استفاده می شوند اگر مدلها را با شرایط سازمان هماهنگ نکنیم نتایج بدست آمده اصلا قابل اعتماد نخواهد بود.

بنا براین هم به ابزارهای برآورد بیشتر و بهتری احتیاج داریم و هم لازم است نیرو و زمان بیشتری برای توسعه نرم افزار های برآورد هزینه صرف کنیم.

وقتی در آغاز پروژه شروع به انجام برآورد می کنیم موارد غیر قطعی و مبهم زیادی وجود دارند. معلوم نیست چه Cost Driver هایی در برآورد ما مهم هستند و میزان تاثیر هر کدام روی برآورد چقدر است. افراد زیادی در پروژه شرکت دارندمثل مدیر پروژه، مشتری، توسعه دهندگان، کاربران و غیره که هر کدام عقاید و خواسته هایی در مورد پروژه دارند و اغلب بین اینها تداخل پیش می آید، مثل مینیمم کردن هزینه، ماکزیمم کردن کیفیت، مینیمم کردن زمان انجام پروژه، استفاده بهینه از کارمندان و غیره. برای یک مدیر پروژه در چنین موقعیت مبهمی پیش بینی کردن روند انجام کار پروژه کار سختی است. بدست آوردن یک برآورد نقطه ای مثلا اینکه زمان لازم برای انجام پروژه ۳۲۱ نفر ماه است که ۱۱۰ نفر ماه برای تحلیل و ۷۰ نفر ماه برای طراحی است و غیره،اهمیت کمی دارد مدیر پروژه بیشتر علاقه مند است تا سناریو هایی برای وضعیت های مختلف داشته باشد در حقیقت برای مقادیر مختلفی که Cost Driver ها می توانند داشته باشند.

برای مثال یک مدیر می خواهد بداند اگر دو تحلیل گر دیگر به پروژه اضافه شوندچه تاثیری روی نتیجه خواهد داشت. یا اینکه اگر زمانی که برای انجام پروژه داریم ناگهان به شدت کم شود چه تاثیری روی پروژه دارد؟و غیره. و همین سناریوهای متغیر مبحث کنترل پروژه را بوجود می آورند و اینکه مدیران پروژه باید روی Cost Driver هایی که تاثیر زیادی روی پروژه دارند تاکید و دقت بیشتری در طول توسعه پروزه داشته باشند.

مهمترین پیش نیازها برای یک برآورد موفق، توسعه، پذیرفتن و استفاده از یک مجموعه تعریف های استاندارد و معین است. برای مثال:

- چند بار برای یک پروژه برآورد صورت می گیرد. برای مثال ۵ بار برای پروژه هایی که هزینه ای بیشتر از ۱۲ نفر ماه دارند.
- در کدام مرحله طول انجام پروژه برآورد انجام شده. برای مثال: در طول امکان سنجی بعد از پایان طراحی و غیره
- چه افرادی در فرآیند برآورد هزینه نقش دارند. برای مثال مدیر پروژه، مشتری ها و توسعه دهندگان
- چه چیزی برآورد می شود. برای مثال تمامی فعالیت های توسعه فاز امکان سنجی، طراحی و غیره. یاتمامی فعالیت های مربوط به مستند سازی و ...
- خروجی های برآورد چیست. برای مثال هزینه به دلار، نیروی کار به نفر ماه، زمان به ماه
- فاکتور هایی که به عنوان مهمترین Cost Driver ها در نظر گرفته می شوند. برای مثال اندازه، قابلیت اطمینان، نوع برنامه کاربردی، کیفیت پرسنل و غیره.
- مجموعه ای از تعریف ها. برای مثال حجم پروژه به FP اندازه گیری می شودیا مثلا مستندات شامل چه چیزهایی هستند. تعریف پیچیدگی و غیره.

و مدلهای برآورد باید ساختاری داشته باشند که شامل چنین تعریف های استانداردی باشند.

سرانجام برای ساخت یا انتخاب یک مدل برآورد پیشنهادات زیر ارائه شده اند:

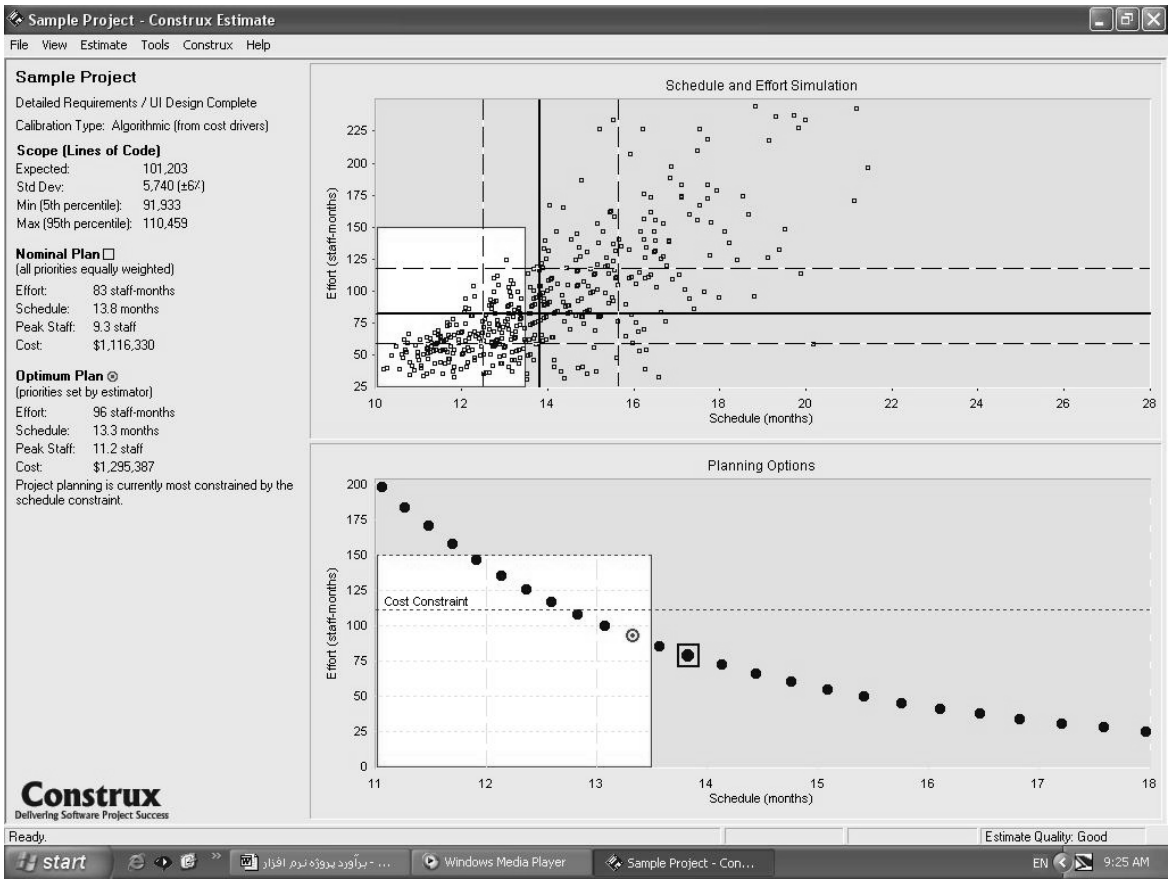
- مشخص کردن سطح عدم قطعیت.روش برآورد پروژه هایی که عدم قطعیت بالایی دارند کاملا با آنهایی که عدم قطعیت کمی دارند متفاوت است عدم قطعیت بالا شامل تحلیل ریسک و استفاده از تکنیک های برآورد پیشرفته است ولی در پروژه های با عدم قطعیت کم می توان از مدلهای برآورد هزینه، ابزارهای محاسبه تجربیات پروژه های گذشته استفاده کرد.
- جمع آوری اطلاعات. جمع آوری اطلاعات پروژه های تکمیل شده گذشته برای انجام برآورد موفق لازم است. مدلهای برآورد هزینه بر اساس مقایسه و تجزیه، به چنین اطلاعاتی احتیاج

دارند. نباید اطلاعات مربوط به سازمانهای دیگر را جمع آوری کرد زیرا این اطلاعات برای سازمانهای مختلف متفاوتند.

- استفاده از بیش از یک مدل. مدلهای برآورد کیفیت پایینی دارند. دقت پایین و غیر قابل اطمینان بودن تکنیک های برآورد هزینه باعث بوجود آمدن ریسک بالایی در پروژه های نرم افزاری می شوند، بنابراین لازم است برآورد به طور منظم در طول انجام پروژه انجام شوند و از بیش از یک تکنیک برآورد استفاده شود.دانشی که در باره پروژه در حال توسعه داریم به مرور زمان و در مراحل مختلف پروژه رشد می کند، بنابراین می توانیم از مدلهای دیگری در طول پروژه برای برآورد مجدد استفاده کنیم.

فصل سوم –
نرم افزار های برآورد پروژه

۳-۱ Construx Estimate



نرم افزار برآورد هزینه Construx

یک ابزار برآورد نرم افزار است که امکانات زیر را دارد:

- **تعیین فاز پروژه.** در کدام مرحله انجام پروژه هستیم؟ در شروع امکان سنجی؟ فرضیات پروژه تعیین شده؟ واسطهای گرافیکی طراحی شده اند؟ طراحی کامل شده؟ کد نویسی انجام شده؟ در مرحله تست هستیم؟ می توان هر کدام از اینها را در نظر گرفت و براورد را انجام داد.
- **تعیین نوع پروژه.** آیا یک سیستم تجاری است؟ سیستم کنترلی است؟ علمی مهندسی است؟ ارتباطی است؟ نرم افزار یا راه انداز سیستم است؟ نوع پروژه را می توان انتخاب کرد.
- **تنظیم Cost Driver ها.** شامل تنظیم صفات محصول : قابلیت اطمینان محصول چقدر باید باشد؟ اندازه پایگاه داده چقدر است؟ پیچیدگی محصول چقدر است؟ زمان اجرای محصول چقدر از وقت CPU را مصرف می کند؟ چقدر از منابع را مصرف می کند؟ چقدر از حجم

حافظه را مصرف می کند؟ سابقه انجام کار مشابه وجود دارد؟ تنظیم صفات پروژه: آیا کامپوننت ها ی ایجاد شده باید قابلیت استفاده مجدد داشته باشند؟ هزینه برای ایجاد سند چقدر است؟چقدر از ابزار های برنامه نویسی استفاده می شود؟ اعضای تیم چقدر و چگونه با هم ارتباط دارند؟ انعطاف پذیری چقدر لازم است؟ ریسک تا چه میزان قابل قبول است؟ صفات پرسنل: میزان توانایی طراح، میزان توانایی برنامه نویس، میزان آشنایی با محیط برنامه کاربردی، میزان آشنایی با محیط، میزان آشنایی با زبان و ابزارهای کار،هماهنگی تیمی

- **تنظیم دامنه نرم افزار**. در این قسمت اندازه نرم افزار تعیین می شود. در این نرم افزار چند روش برای تعیین اندازه وجود دارد که می توان بر طبق نیاز از هر کدام استفاده کرد.روشهای LOC، Function Point، Function/Subroutine، Class/Modules و روش Subsystem قابل استفاده اند. زبان برنامه نویسی هم می توان انتخاب کرد. قسمت مربوط به تنظیمات نقطه عملکرد هم وجود دارد.ضریب پیچیدگی تعیین می شود و ضرایب نقطه عملکرد برای ورودیهای کاربر، خروجی ها، فایلها واسطهای کاربر و غیره تعیین می گرد و نقطه عملکرد را محاسبه می شود. برای محاسبه کامپوننت های واسطهای گرافیکی کاربر هم امکاناتی وجود دارد، باید تعداد پیامهای محاوره ای، پنجره ها، گزارشها،خروجیها، فایلهای نگهداری اطلاعات و واسطهای خارجی تعیین شود و می توان هریک از این تنظیمات را برای استفاده های بعدی ثبت کرد.

با تنظیم همه اینها به سه سری نتیجه می رسیم که در هر یک از اینها میزان نیروی کار، زمان انجام کار، هزینه و حداکثر تعداد پرسنل محاسبه شده است.

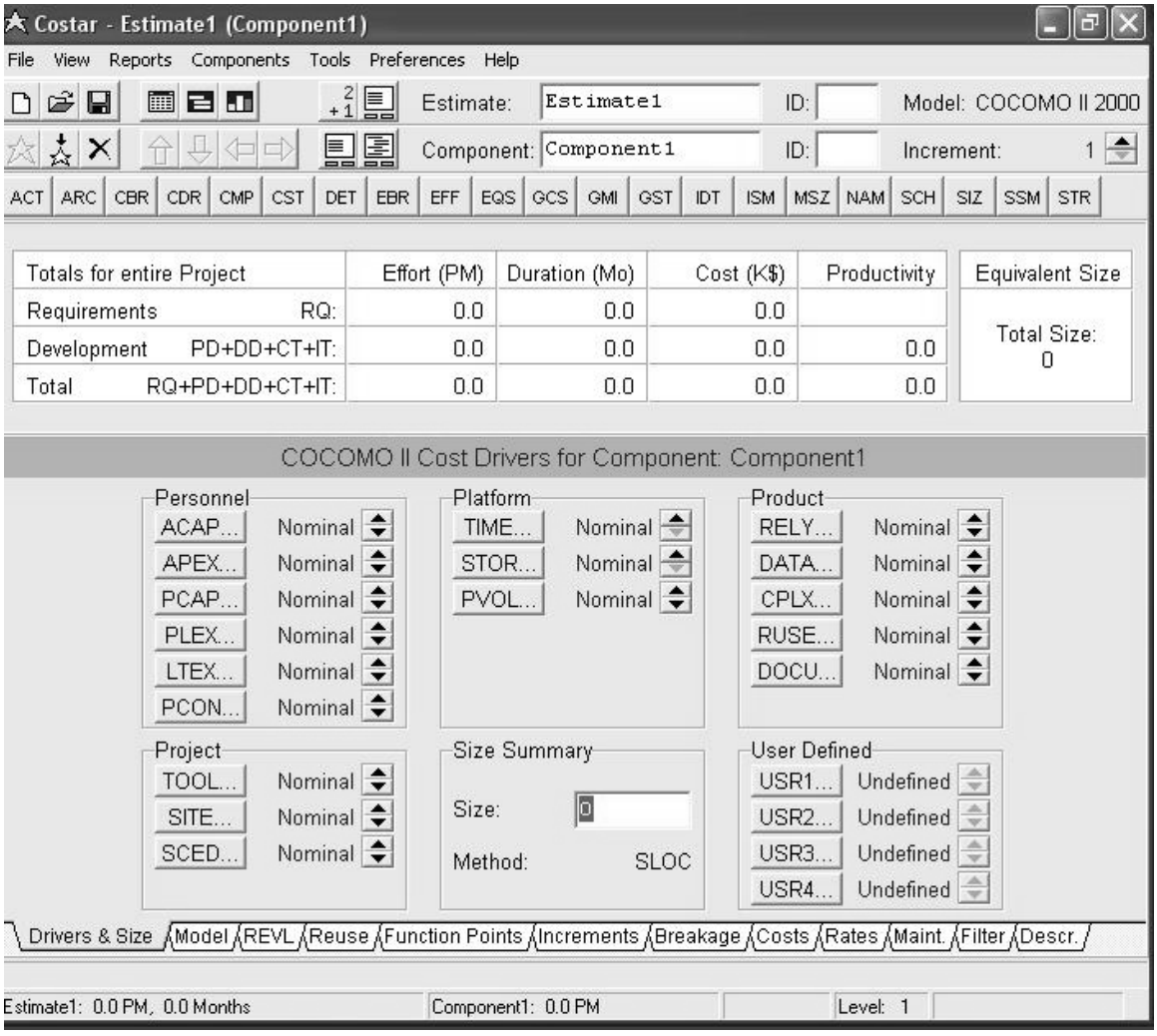
- **برآورد جزئی**: وقتی که تمامی تقدمها در وزن مساوی قرار دارند
- **برآورد بهینه**: وقتی تمامی تقدمها و انتخابها را برآورد کننده تنظیم کرده باشد.
- **نمایش گرافیکی زمان و کار**

اما فرمولهایی که با استفاده از آنها برآورد انجام می شود فرمولهای کوکومو دو هستند وهیچ امکان تغییری در نرم افزار در نظر گرفته نشده است.

برای Download کردن نرم افزار از این آدرس استفاده کنید:

http://www.construx.com/membership/login.php?dest=%2Fresources%2Festimate%2Fdownload.php

۳-۲ Calico- Costar



نرم افزار برآورد پروژه Costar

Costar ابزاری برای برآورد خودکار نیروی کار، زمان انجام کار، هزینه و نرخ تولید محصول است.

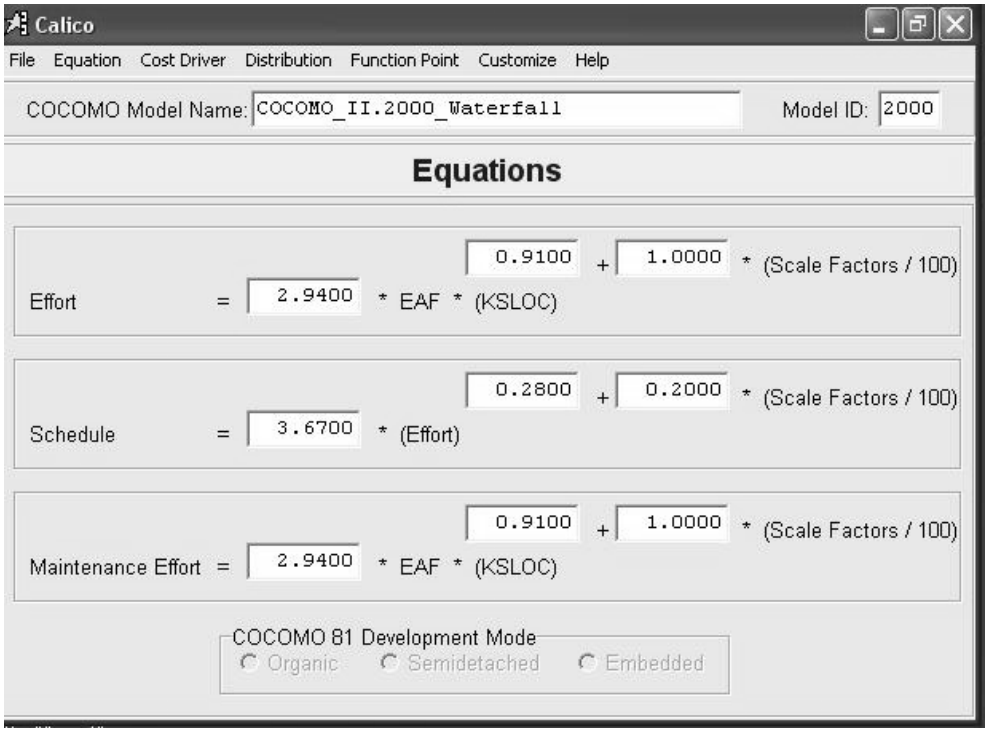
برای انجام یک برآورد باید موارد زیر تنظیم شوند

- **تنظیم Cost Driver ها**. شامل صفات پرسنل، پروژه، محصول و محیط . این عوامل باید مقدار دهی شوند:توانایی طراح پروژه، شناخت برنامه کاربردی، توانایی برنامه نویس، شناخت محیط برنامه نویسی، شناخت زبان و ابزار، نرخ جابجایی پرسنل، شناخت زبان برنامه نویسی، شناخت ماشین مجازی، توانایی پرسنل، شناخت پرسنل، برنامه های امنیتی طبقه بندی شده، استفاده از مودم در شیوه برنامه نویسی، استفاده از ابزارهای برنامه نویسی، پراکندگی سایتها، زمانبندی پیشبرد کار، سهولت کارو تغییر ناگهانی نیاز ها، مدت زمان اجرا و وقتی که از CPU

می گیرد، مصرف منابع ذخیره، زمان تغییر کار کامپیوتر، تغییر ناگهانی ماشین مجازی میزبان و مقصد، محیط، قابلیت اطمینان لازم محصول، اندازه پایگاه داده، پیچیدگی محصول، میزان اسناد تهیه شده، قابلیت اطمینان و پیچیدگی محصول و عواملی که کاربر به تشخیص خود در نظر میگیرد.

- **انتخاب روش برآورد**.در Costar می توان از مدلهای برآورد مختلفی استفاده کرد.
- **تنظیم Scale Factor ها**. این عوامل در محاسبه دخالت دارند: آیا پروژه مشابه ای قبلا انجام شده؟ میزان انعطاف پذیری در پیشبرد کار، میزان ریسک قابل قبول، هماهنگی تیمی و سطح پروژه در زمان آغاز آن.
- **محاسبه اندازه پروژه**. می تواند مستقیما بر اساس SLOC وارد شود ، بر اساس استفاده مجدد از کامپوننت های موجود محاسبه شود و یا به طریق Function Point ، ضرایب نقطه عملکرد وارد می شوند، ضریب پیچیدگی محاسبه می شود و نقطه عملکرد محاسبه می گردد.
- برای پروژه هایی که با روشهای تکاملی انجام می شوندهم امکان برآورد هر مرحله افزایشی وجود دارد.
- هزینه های هر قسمت پروژه شامل نیازمندیها، طراحی محصول، طراحی جرئیات، کدنویسی و تست واحدها، تست جامعیت و هزینه نگهداری وارد می شود.
- برآورد های مربوط به نگهداری سیستم

برای سازگار کردن مدل برآورد با مشخصات سازمان از نرم افزار مکمل Calico استفاده می شود. بر اساس داده های تاریخی سازمان می توان Cost Driver هایی را که در سازمان مورد نظر روی پروژه اثر می گذارند و میزان تاثیر آنها را اندازه گرفت و مدل برآوردی سازگار با سازمان خود ساخت.



Calico مکمل نرم افزار برآورد

تنظیماتی که در Calico انجام می شوند با عنوان یک مدل برآورد ثبت شده و سپس به Costar وارد می شوند. متغیرهایی که در Calico مقدار دهی می شوند به این ترتیب هستند:

- معادلات محاسبه کار، زمان بندی و نگهداری
- مقدار دهی به Scale Factor ها
- تعیین تعداد ساعات کار در هر ماه
- تعیین Cost Driver ها ی موثر و مقدار دهی به آنها
- مقدار دهی ضرایب توزیع، تعیین درصد قسمتهای پروژه شامل نیازمندیها، طراحی محصول، طراحی جزئیات،کد نویسی و تست واحد ها و تست جامعیت است.
- وزن ضرایب Function Point و تعیین متغیر های ضریب پیچیدگی.

برای دسترس به این نرم افزارها از این آدرس استفاده کنید:
http://www.softstarsystems.com/?referrer=ad1

فصل چهارم-
ساختار نرم افزار

۱-۴ محیط کاری Visual studio .NET

برای پیاده سازی نرم افزاری که روی وب برای پروژه های نرم افزاری برآورد هزینه انجام دهد از محیط ASP.NET استفاده شده است.

NET.محیطی است که برنامه های تحت وب و برنامه های سنتی در آن ساخته می شود. زبانهای معمول برنامه نویسی در محیط دات نت ویژوال بیسیک دات نت و سی شارپ دات نت است که این نرم افزار با ویژوال بیسیک دات نت نوشته شده. ASP.NET محیطی است که برنامه های کاربردی تحت وب به یکی از زبانهای برنامه نویسی در آن نوشته می شوند.

برای استفاده از برنامه هایی که با NET. نوشته شده اند به نسخه های جدید ویندوز نیاز داریم می توان از ویندوز ۲۰۰۰ یا xp استفاده کرد. این نرم افزار روی ویندوز xp و Windows 2000 advanced server نوشته شده است.

برای استفاده از ASP.NET کامپیوتر باید بصورت سرور عمل کند. برای اینکار باید نرم افزار IIS روی کامپیوتر نصب شود وقتی با استفاده از IIS به صفحات برنامه دسترسی پیدا می کنید دقیقا مانند این است که از راه دور با اتصال به اینتر نت این صفحات را می بینید . سرعت دسترسی به سایتهایی که با دات نت پیاده سازی شده اند بالاست زیرا این صفحات نخست کامپایل می شوند و در زمان استفاده فقط از کد های کامپایل شده استفاده می شود.

قبل از نصب Visual Studio.NET باید IIS روی کامپیوتر نصب شده باشد . IIS یکی از قطعات ویندوز ۲۰۰۰ و xp است و در قسمت نصب کامپوننت های ویندوز در کنترل پنل نصب می شود.

قبل از نصب Visual Studio.NET لازم است کامپوننت هایی روی ویندوز شما نصب شده باشد. به همین دلیل احتمالا مشکلی در نصب خواهید داشت به اینصورت که بعد از گذاشتن CD و اجرای دستور نصب پیغام خطایی ظاهر می شود و از شما می خواهد ابتدا کامپوننت های مورد نیاز را که مهمترین جز آن frame work دات نت است را روی کامپیوتر نصب کنید. معمولا آخرین دیسک حاوی فایل های این کامپوننت هاست پس ابتدا از آخرین دیسک شروع کنید و فایل نصب کامپوننت ها را پیدا کرده نصب کنید و دوباره دیسک اول را بگذارید.

۲-۴ واسط گرافیکی کاربر

همانند تمامی برنامه های کاربردی تحت اینتر نت کاربر با تعدادی فرم مواجه می شود. این فرمها اساس کار صفحات وب در ASP.NET هستند. وظیفه فرمها این است که ورودی های کاربر را در یافت کند و خروجی ها را به کاربر نشان دهد.

هر فرم وب از چندین کنترل تشکیل شده است که اینها می توانند کنترلهای HTML ای یا کنترلهای سروری باشند. در این نرم افزار تنها از کنترلهای وب سروری استفاده شده است.

کنترل ها در واقع تمامی اشیایی هستند که در یک فرم وجود دارند چه کاربر آنها را ببیند و چه از دید کاربر پنهان باشند. متنی که کاربر روی فرم می بیند یک کنترل است. دکمه ای که برای انجام عمل انتخابی اش فشار می دهد هم یک کنترل است.

- کنترلهای HTML ای-درواقع تمامی کنترلهایی هستند که در فرمهای غیر دات نت دیده می شوند. صفحات html و asp از این کنترلها استفاده می کنند.
- کنترلهای web - ظاهر این کنترلها با کنترلهای html ای فرقی ندارد ولی در واقع هر کدام اشیایی از کلاسهای از پیش تعریف شده اند که می توان به متد های آنها دسترسی داشت. مثل خاصیت Text. یا Value. در صفحات Asp.NET علاوه بر کنترلهای html ای از این کنترلها هم می توان استفاده کرد ولی برای دیدن این صفحات لازم است که frame work دات نت روی سرور نصب شده باشد.

برای ساخت فرم ها در ویژوال استودیو دات نت به این تر تیب عمل می کنیم:

ابتدا یک پروژه جدید از نوع ASP.NET web Application می سازیم و همینطورزبان برنامه نویسی راکه می خواهیم پروژه با آن پیاده سازی شود انتخاب می کنیم یک پروژه روی local host کامپیوتر ساخته می شود که در حالت پیش فرض یک فرم دارد و می توانیم فرمهای وب مورد نظرمان را به آن اضافه کنیم.

این نرم افزار از حدود سی فرم وب تشکیل شده که هر یک وظیفه ای در رابطه با کاربر بر عهده دارند.

بطور کلی فرمهای برنامه به چند دسته تقسیم می شوند:

۱. فرمهای تشخیص اعتبار کاربر
۲. انتخاب عملی که کاربر می خواهد انجام دهد شامل انتخاب مدل برآورد، ساخت مدل برآورد و حذف یک مدل برآورد.
۳. فرمهای در یافت اطلاعات مربوط به سازمان
۴. فرمهای نمایش اطلاعات حاصل از محاسبات برآورد
۵. فرمهای دریافت اطلاعات مربوط به مدل برآورد مورد نظر کار

۴-۲-۱ فرمهای انتخاب کار

کاربر در دو فرم با انتخاب روبه رو می شود.ابتدا فرم انتخاب زبان که در لینک به صفحات فارسی و انگلیسی وجود دارد و بعد فرم انجام برآورد که در آن می تواند یکی از مدلهای برآورد را که قبلاً اطلاعات مربوط به آن در پایگاه قرار گرفته از لیست انتخاب کند و برآورد را انجام دهد یا کلیه اطلاعات مربوط به یک مدل برآورد را از پایگاه داده حذف کند و یا خود یک مدل برآورد جدید را ایجاد کند. در حقیقت این قسمت محلی است که مدیر پروژ] باید مدل رابرای محیط کاری خودش اندازه گیری و سازگار کند. کاربر امکان این را دارد که نتایج حاصل از برآورد خود را در پایگاه داده وارد کند و به ترتیب ثبت کند و یا با نتایجی که قبلاً ثبت شده مقایسه کند. اختلاف این نتایج در جدولی نمایش داده می شود.

۴-۲-۲ فرم های دریافت اطلاعات مربوط به سازمان و پروژه

در این فرمها باید اطلاعات بصورت ضریب های عددی ازکاربر دریافت شوند. برای سهولت کار، کاربر این ضرایب عددی بصورت درجه بندی ازفوف العاده زیاد، خیلی زیاد، زیاد، جزئی، کم، خیلی کم، فوق العاده کم نمایش داده می شوند. مقادیر عددی این درجه بندی در پایگاه داده موجود است و با انتخاب کاربر ضرایب مدل دلخواه از آن استخراج شده و بدست می آید.
در این فرمها سوالات جامعی درباره انواع پروژه های نرم افزاری وجود دارد ولی در هر مدل برآورد تنها تعدادی از این عوامل اهمیت دارند. بقیه عوامل غیر فعالند و کاربر نمی تواند آنها را تغییری بدهد. این ضرایب عددی بزرگتر یا مساوی صفر هستند، حال آنکه حاصلضرب تمامی ضرایب مورد نظر است. بنابراین باید چک شود تا درجه ای که مقدار صفر دارد در لیست انتخاب کاربر وارد نشود.

۴-۲-۳ فرم های نمایش اطلاعات حاصل انجام محاسبات

در حقیقت فرم نمایش نتایج برآورد هزینه و کار و زمان انجام پروژه است.

۴-۳ ساختار فرم ها

در ساخت فرمهای وب از این کنترلهای سرور استفاده شده است:
Label: کنترلی است که متن ساده درآن قرار می گیرد.
Text Box: کنترلی است که کاربر می تواند داده هایش را در آن وارد کند و خروجی های برنامه هم د رآن نمایش داده می شوند.

Dropdown List: لیستی که به کاربر اجازه می دهد داده های خاصی را که در آن لیست هست انتخاب کند و معمولاً برای نمایش اطلاعات درون پایگاه داده استفاده می شود.

۴-۴ انتقال اطلاعات بین فرمها

ساختار برنامه بر اساس وب فرمهاست اطلاعات از وب فرمها وارد می شوند ولی گاهی لازم است اطلاعات بین فرمها منتقل شوند. انتقال اطلاعات بین فرمها به سادگی انجام نمی شود وقتی کاربر درخواست جدیدی به سرور می دهد اطلاعات مربوط به در خواستهای قبلی کاملا از بین می روند، یعنی سرور نمی تواند تشخیص دهد که در خواست جدید از همان کاربر قبلی است یا نه. بنابراین احتیاج به حالتی داریم که داده ها را در سرور ذخیره کنیم راههای مختلفی برای این کار وجود دارد در این در این نرم افزار از Session State استفاده شده است به این ترتیب هر کاربری که در خواستی به سرور ارسال کند یک Session State برایش ایجاد خواهد شد و اطلاعاتش در آن ذخیره خواهد شد.یعنی مقادیری را که می خواهیم نگهداری کنیم می توانیم در یک Session قرار دهیم.

حجم زیادی از اطلاعات لازم است که در این نرم افزار میان فرمها منتقل شوند از جمله شناسه مدلی که برآورد با آن انجام می شود. در هر کدام از فرمهای انجام برآورد لازم است مشخص باشد که برآورد با چه مدلی انجام می شود بنابراین در فرم انتخاب مدل Session مقدار دهی می شود به این صورت

Session (“model_id”)= model_id

و سپس در هر فرم می توان مقدار آن را بررسی کرد

Model_id= Session (“model_id”)

و به این ترتیب هر جا در نرم افزار که لازم بوده اطلاعاتی جابه جا شود از Session State استفاده شده است.

۴-۵ در یافت اطلاعات از فرمها

در یافت داده هایی که کاربر در کنترلهای وب سروری وارد می کند کار ساده ای است هر کدام از آنها اشیایی هستند که بوسیله متدهای text و Value مقادیرشان بدست می آید. دستورات دریافت اطلاعات از کنترلها به صورت زیر است :

دریافت اطلاعات از textbox:

Textbox_val=textbox.text

دریافت اطلاعات از drop down list :

Dim d7_value as double= double. Parse (d7.SelectedItem.Value)

۴-۶ نمایش اطلاعات در فرمها

نمایش اطلاعات در کنترلهای وب سرور هم به سادگی انجام می شود. باید توجه داشت که نتایج محاسبات برای نمایش باید بصورت رشته درآیند که اینکار با متد () ToString. انجام می گیرد.
دستورات نمایش اطلاعات در کنترلهای وب سرور به اینصورت است:
نمایش اطلاعات در Label :

```
Label.Text = label_val.Tostring ()
```

نمایش اطلاعات در textbox :

```
Textbox_val.text = math.round ( text_box_val,1 ).tostring()
```

همچنین برای نمایش نتیجه حاصل از انجام محاسبات که عددی با چندین رقم اعشار است باید نتیجه محاسبات گرد شود از توابع ریاضی ویژوال بیسیک دات نت برای اینکار استفاده می شود.
تابعی که از آن در این نرم افزار استفاده شده: math.round

۴-۷ فضای نام[1]

در محیط کاری net. حدود ۳۴۰۰ کلاس وجود دارد. این کلاسها به دسته های مختلفی تقسیم می شوند و کلاس های مرتبط در دسته های جداگانه قرار می گیرند. به این دسته ها name space گفته می شود.
فضای نام سیستم ریشه تمامی فضای نامهاست و همه کلاسها در آن قرار دارند. برای مثال فضای نام system.web بیشتر قابلیت ساخت صفحات وب را در ASP.NET دارد و فضای نام system.web.UI تمام عملیات مربوط به اینترفیس را در خود دارد.
وقتی در ASP.NET بخواهیم از کلاسی استفاده کنیم باید بدانیم که در کدام فضای نام قرار دارد و آن را به برنامه اضافه کنیم. مثلا برای کار با پایگاه داده باید فضای نام کلاسهای مربوط به دسترسی به پایگاه داده را در ابتدای فرم خود وارد کنیم

```
Imports System.Data
Imports System.Data.OleDb
```

[1] Name space

۸-۴ کنترلهای اعتبار سنجی:

هر کدام از صفحات پروژه حاوی فرمی است و هر فرم از چند کنترل تشکیل شده است. کنترلهایی برای دریافت اطلاعاتی مثل نام مدل و یا مقادیر ضرایب. در بسیاری موارد لازم است محتویات این کنترلها قبل از ارسال به سرور ارزیابی شوند. به عنوان مثال مشخص شود که فیلدی حاوی مقدار است یا خیر، آیا مقدار فیلدی در شرایط خاصی صدق می کند یا نه. اینکه مقادیر این کنترلها به سرور ارسال شود، سرور عملیاتی را برای تائید اعتبار آنها انجام دهد غیر ممکن نیست ولی در عمل بسیار وقت بر و پر هزینه است. کنترلهای اعتبار سنجی جاوا اسکریپت هایی هستند که مقادیر ورودی کنترلها را قبل از ارسال به سرور چک می کنند و بنا براین سرعت صفحات را بسیار بالا می برند بنابراین برای بررسی اینکه فیلدی حتما شاما مقدار است و موارد مشابه با آنها از کنترلهای اعتبار سنجی استفاده می شود. اعتبار سنجی داده ها در طرف مشتری(قبل از تحویل به سرور) از اهمیت ویژه ای بر خوردار است.
کنترلهای اعتبار سنجی از کتابخانه های Java Script استفاده می کنند که هنگام نصب ASP.Net بر روی سرور نصب می گردد. در تمامی صفحات پروژه هر جا که لازم است فیلدها حتما مقدار داشته باشد یا مقدار خاصی داشته باشد از کنترلهای اعتبار سنجی استفاده شده است تا از ورود اطلاعات غیر مجاز جلوگیری شود. برای مثال در صفحه اصلی پروژه اگر بدون وارد کردن نامی برای مدل برآورد جدید دکمه New Model را فشار دهیم به چنین اعلام خطایی بر می خوریم.

کنترلهای اعتبار سنجی

Validation control ها چند نوع دارند. در این نرم افزار از کنترل های زیر استفاده شده است :

اعتبار سنجی وجود یک فیلد[1]

این کنترل می تواند تعیین کند که یک کنترل دارای مقدار است یا نه. خواصی از این کنترل که باید مقدار دهی شوند به این شرح اند:

- Control to Validate مشخص می کند که کدام کنترل قرار است اعتبار سنجی شود.
- Error Message پیام خطایی که در صورت مقدار نداشتن کنترل باید نشان داده می شود.

وقتی قرار است داده ای در پایگاه داده وارد شود وجود این کنترل ها لازم است زیرا نداشتن مقدار در این کنترلها خطاهای مهلکی در نرم افزار ایجاد می کند.

اعتبار سنجی مقایسه ای[2]

می تواند محتوی یک کنترل را با یک مقدار ثابت مقایسه کند. در این برنامه که لازم است تمامی ضرایب بزرگتر یا مساوی صفر باشند چنین کنترلی لازم است.

خواصی از این کنترل که باید مقدار دهی شوند :

[1] Required field validator
[2] Compare validator

Control to validate: کنترل یکه باید اعتبار سنجی شود.
Value Compare: مقدار ثابتی که محتویات کنترل باید با آن مقایسه شوند.
Operator: عملگر مقایسه
Type: نوع داده مجاز برای کنترل

اعتبار سنجی مختصر شده[1]

در بسیاری از فرمهای نرم افزار بیش از پنجاه فیلد ورودی وجود دارد اعلانهای خطا در نقاط مختلف فرم شکل خوبی ندارد برای چنین فرمهایی از validation summery استفاده میشود که تمامی اعلانهای خطا را در صفحه پیغامی خارج از فرم نشان می دهد.

۹-۴ فرم تشخیص اعتبار کاربر

از دو متد برای تشخیص اعتبار کاربر استفاده شده. از کنترلهای اعتبار سنجی و دیگری با جستجو در پایگاه داده. ابتدا با استفاده از کنترلهای اعتبار سنجی مشخص می شود که آیا چنین کاربری وجود دارد یا نه. و سپس جستجو در پایگاه داده انجام می گیرد تا مشخص شود پسورد وارد شده درست است یا نه.

۱۰-۴قراردادن برنامه روی اینتر نت

برای اینکه برنامه از اینترنت قابل دسترسی باشد لازم است سیستم عامل سرور ویندوز باشد و frame work دات نت روی آن نصب شده باشد. یک دایرکتوری مجازی روی سرور ایجاد کرده و فرمهای وب و دایرکتوری bin را در آن کپی می کنیم این دایرکتوری زمان ایجاد پروژه های دات نت بطور اتوماتیک ساخته می شود و حاوی اطلاعات لازم برای اجرای برنامه های دات نت است.

[1] Validation summery

۴-۱۱ فارسی سازی صفحات وب

تمامی فرمهای برنامه به زبان فارسی موجود است. فارسی نویسی در وب مشکلات خاصی دارد. تمامی حروف فارسی برای اینکه در وب شکل درستی داشته باشند باید به صورت استاندارد یونیکد درآیند که یک استاندارد جهانی نمایش حروف است. هر حرف فارسی شامل چهار کاراکتر یونیکد است. برای فارسی کردن فرم های این نرم افزار مراحل زیر دنبال شد:
ابتدا متن مورد نظر در یک editor مثل notepad تایپ شده سپس copy و در محیط front page ، Paste می شود اگر متن HTML را ببینیم متن بصورت یونیکد دیده می شود. ان متن را برداشته و از آن در فرمها استفاده می کنیم. به این ترتیب صفحات ما بدون وابستگی به ماشین و سیستم عامل فونت های درستی خواهند داشت.
همچنین برای فارسی کردن کلیه کنترلها کافیست که بعد از تایپ فارسی و اضافه کردن charset=utf-8 به ابتدای فرم آن را بصورت یونیکد ذخیره کنیم.

۴-۱۲ ذخیره اطلاعات در نرم افزار:

برای نگهداری اطلاعات مورد نیاز نرم افزار از جداول Access استفاده شده. برای دسترسی به متد های آن از ADO.NET استفاده می شود.
ADO.NET امکان ارتباط با بانک اطلاعاتی را را فراهم می کند به عبارت دیگر ADO.NET فناوری است که برنامه های کاربردی ASP.NET از آن برای ارتباط با بانک اطلاعاتی استفاده می کنند.
این متد، connection less است به این معنی که وقتی بوسیله آن با بانک اطلاعاتی رابطه برقرار می کنیم ابتدا اتصال برقرار می شود اطلاعاتی که در خواست شده در یک شیی به عنوان Data set قرار می گیرد و ارتباط قطع می شود اگر اطلاعات درون Data Setرا تغییر دهیم اطلاعات اصلی data base تغییر نمی کند برای اعمال تغییرات دوباره اتصال برقرار شده و اطلاعات پایگاه داده تغییر می کند.
ADO.NET اشیا و متدهایی دارد که با استفاده از آنها با پایگاه داده ارتباط برقرار می کند مهمترین اشیا به این ترتیبند:
OledbConnection اتصال به پایگاه داده را برقرار می کند.
OledbCommand دستورات SQL را ذخیره و اجرا می کند.
OledbDataReader درخواستی را اجرا کرده و اجازه خواندن داده ها را می دهد.
OleAdaptor داده های حاصل از درخواست را برمی گرداند یا در پایگاه داده می نویسد

Data Set چند Data table دارد.

Data Table داده ها را بصورت سطر و ستون ذخیره می کند .

برای دسترسی به داده ها از طریق ADO.NET باید برنامه باید بتواند از آن استفاده کند دستور زیر اجازه استفاده از ADO.NET را به برنامه نویس می دهد

Imports System.data

برای اینکه مشخص کنیم از پایگاه داده Access استفاده می کنیم دستور زیر را بکار می بریم:

Imports System.data.Oledb

مراحل اتصال به پایگاه داده:

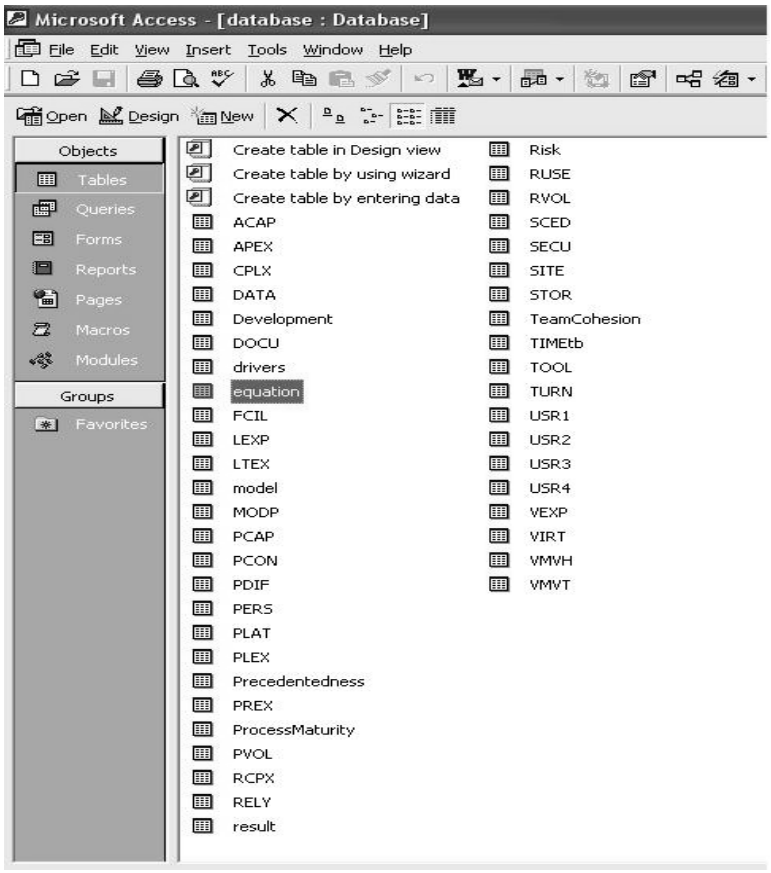
- تعریف شیی اتصال
- ایجاد رشته اتصال
- باز کردن شیی اتصال

و دستورات زیر برای کار با پایگاه داده بکار می روند:

- باز کردن پایگاه داده
- خواندن از پایگاه داده
- حذف و درج در پایگاه داده
- نمایش اطلاعات پایگاه داده در dropdownlist

پایگاه داده پروژه از چهل و پنج جدول تشکیل شده که شامل جداول نگهداری ضرایب Cost Driver ها، Scale factor ها و جداول نگهداری مشخصات یک مدل برآورد و جدول نگهداری نتیجه است.

در پایگاه داده نرم افزار جدولهایی برای هر یک از صفتهایی که برآورد در مدلهای مختلف برآورد مهمندوجود داردو در هر جدول ضرایب درجه بندی شده برای مدل خاص وجود دارند. علاوه بر این صفات لازم است عنوان مدل برآوردضرایب معادلات و نتاج حاصل از برآورد هم نگهداری شوند نحوه ارتباط کلیه جداول از طریق شناسه مدل است که در هر جدول برای شناسایی ضرایب مربوط به یک مدل خاص وارد می شود. اما لازم است بدانیم که کدامیک از عوامل در کدام مدل اهمیت دارند. هر عاملی که کاربر به عنوان عامل موثر در برآورد انتخاب کند در این جدول علامت می خورد. درج در هر جدول فقط یکبار صورت می گیرد بنابرین داده هایی که از یک جدول که در فرمهای مختلف بدست آمدخه اند جمع آوری شده و در انتها وارد جدول می شوند.



پایگاه داده پروژه

فصل پنجم –
شرح نرم افزار

۵-۱ طرز کار نرم افزار برآورد

کار انجام شده تهیه یک نرم افزار تحت وب برای برآورد On-Line پروژه های نرم افزاری است. نرم افزار در محیط Visual Studio.NET و به زبان VB.NET نوشته شده . نرم افزار تقریبا مشابه Costar است. نرم افزار در این آدرس اینترنتی قابل دسترس است:
www.aut.ac.ir/departments/comp/estimate

امکانات نرم افزار:

- برآورد هزینه، زمان،کارو نرخ تولید محصول برای یک پروژه نرم افزاری
- ایجاد مدل برآورد دلخواه
- ثبت و نگهداری نتایج حاصل از برآورد
- مقایسه نتایج برآورد فعلی با برآوردی که قبلا انجام شده
- امکان کار با صفحات فارسی و انگلیسی
- امکان دسترسی On-Line به یک نرم افزار برآورد

۵-۱-۱ انجام برآورد

صفحه اول صفحه انتخاب زبان است می خواهید از صفحات فارسی استفاده کنید یا صفحات انگلیسی دکمه هایی برای هر انتخاب در صفحه دیده می شوند.

Index.aspx

صفحه اصلی نرم افزار سه انتخاب را نشان می دهد

- انتخاب یک مدل و انجام برآورد
- ایجاد یک مدل برآورد
- حذف یک مدل برآورد

تمامی ضرایب مدل کوکومو دو قبلا وارد شده اند و در ادامه برای نشان دادن کار نرم افزار یک برآورد برای یک پروژه فرضی انجام می شود.

Default_fa.aspx

برای شروع برآورد مدل COCOMO|| 2000 را از لیست مدلها انتخاب کرده دکمه Start Estimation را فشار می دهیم . صفحه ای که واردش می شویم محل تنظیم مقادیر Cost Driver هایی است که در این مدل در مقدار ضریب EAF در معادله Effort نقش دارند.

driversize_fa.aspx

Cost Driver ها در تمامی مدلها یکسان نیستند و در هر مدل برآورد، تعدادی از آنها موثرند، و بقیه همانطور که در تصویر دیده می شود غیر فعالند، اینکه کدامها فعال باشند زمان ایجاد مدل مشخص می شود، Cost Driver های مدل کوکومو دو ۲۰۰۰ فعالند و می توان مقدار آنها را با توجه به لیستی که در مقابل هر کدام است تنظیم نمود. Cost Driverها شامل موارد زیر هستند:

- توانایی طراح پروژه چقدر است؟
- اعضای تیم پروژه تا چه حد با چنین برنامه های کاربردی آشنایی دارند؟
- توانایی برنامه نویس چقدر است؟
- اعضای تیم چقدر نسبت به محیط مقصد آشنایی دارند؟
- اعضای تیم چقدر به زبان و ابزار برنامه نویسی آشنایی دارند؟
- نرخ جابجایی کارمندان چقدر است؟
- اعضای تیم چقدر با زبان برنامه نویسی آشنایی دارند؟
- میزان آشنایی با ماشین مجازی چقدر است؟
- میزان توانایی پرسنل چقدر است؟
- میزان شناخت از پرسنل چقدر است؟
- میزان برنامه های امنیتی دسته بندی شده چقدر است؟
- چقدر از مودم برای ایجاد ارتباط در شیوه های برنامه نویسی استفاده می شود؟
- از چه ابزار نرم افزاری در پروژه استفاده می شود؟ آیا تنها از ابزارهای کد نویسی و دیباگ استفاده می شود یا از ابزار پیشرفته تر تولید کد هم استفاده می شود؟
- آیا اعضای تیم در مکانهای مختلفی پراکنده اند؟ نحوه ارتباط آنها چطور است؟
- زمانبندی انجام پروژه چگونه است؟ چه مقدار عجله و فشردگی در کار است؟
- میزان سهولت کار چقدر است؟
- تغییر ناگهانی نیازها.آیا نیازهایی که برای آنها برنامه ریزی شده ثابتند یا در طول زمان تغییر می کنند؟
- قید زمانی برنامه چقدر است؟ چقدر از وقت پروژه را می گیرد؟
- تغییرات ناگهانی ماشین مجازی چقدر است؟
- تغییرات ناگهانی ماشین مجازی میزبان چقدر است؟
- تغییرات ناگهانی ماشین مجازی مقصد چقدر است؟
- تغییرات ناگهانی محیط. محیط کار ثابت است یا تغییر می کند؟محیط کار شامل سیستم عامل و سیستم مدیریت پایگاه داده و مانند آن است.
- دشواریهای محیط

- محیط
- منابع ذخیره کننده. مثلا برنامه چه حجمی از حافظه را اشغال می کند؟
- قابلیت اطمینان محصول. شکست و نقص در نرم افزار چه ریسکی دارد؟آیا به زندگی انسانها بستگی دارد یا اهمیت کمی دارد؟
- اندازه پایگاه داده ای که برای تست نرم افزار لازم است چقدر است؟
- نرم افزار تا چه میزان می تواند پیچیده باشد؟ آیا از کدها و واسطهای ساده استفاده می شود یا از کدها و واسطهای گرافیکی پیچیده؟
- آیا نرم افزار باید قابلیت استفاده مجدد داشته باشد؟
- چه مقدار اسناد و توضیحات برای نرم افزار ایجاد شده؟
- قابلیت اطمینان و پیچیدگی محصول
- عواملی که کاربر می خواهد در تنظیم مقدار EAF موثر باشد.

هر کدام از این عوامل می توانند یکی از این مقادیر را داشته باشند: Extra High ,Very High ,High, Nominal, Low, Very Low, Extra Low

حاصلضرب اعدادی که از اینها بدست می آیند مقدار EAF را می سازد.

دکمه next صفحه تنظیم Scale Factor ها رانشان می دهد که در تمامی مدلهای برآورد موثرند و بای مقدار دهی شوند .

Scalefactor_fa.aspx

- سابقه انجام کار مشابه.آیا قبلا هم پروژه ای شبیه این انجام شده است یا نه؟ که می تواند یکی از این مقادیر را داشته باشد:آشنایی زیادی با این پروژه وجود دارد، آشنایی کلی وجود دارد، تقریبا بدون سابقه است، مقدار زیادی بی سابقه است، کاملا بی سابقه است.
- آیا نیازمندیهای پروژه قابل انعطاف هستند یا باید همه موارد را در نظر گرفت؟ آیا نرم افزار یا سخت افزار واسط قیدهای خاصی برای پروژه بوجود می آورند؟ که این مقادیر را دارند: هدفهای کلی وجود دارد، حدودا منطبق هستند، انطباق کلی دارن، فشار خفیفی وجود دارد، گهگاه بطور خفیف، فشار شدید است.
- طراحی معماری قبلا تا چه حد تعریف شده و مدیریت ریسک تا چه حد انجام گرفته؟ می توانند مقادیر ۱۰۰٪ ، ۹۰٪ ، ۷۵٪ ، ۶۰٪ ، ۴۰٪ ، ۲۰٪ داشته باشند.
- روابط مشتری، توسعه دهندگان، کاربران و کسانی که نگهداری سیستم را به عهده دارن تا چه میزان است؟می تواند این مقادیر را داشته باشد:
- تراکنش بدون محدودیت، هماهنگی بالا، هماهنگی زیاد، هماهنگی پایه ای، تراکنش های حدودا متفاوت، تراکنش های خیلی متفاوت.
- در آغاز پروژه در چه سطحی قرار دارید؟ بهینه سازی مداوم فرآیندها، فرآیند های نرم افزاری درک شده اند؟ فرآیند های نرم افزاری استانداردی وجود دارد، فرآیندهای پایه ای ساخته شده، قسمت آغازین.

Scale Factorها برای محاسبه ضریبی موسوم به E بکار می روند به این ترتیب که

$$E = \text{Sum of scale factors} * A/100 + B$$

که A و B ضرایب ثابتی هستند در معادله های کار و زمانبندی.

صفحه بعدی صفحه تعیین اندازه پروژه است. برای تعیین اندازه دو انتخاب در صفحه دیده می شود.

Fp_fa.aspx

- **عملکرد گرا**. با انتخاب این گزینه تمامی جدول محاسبه نقطه عملکرد فعال می شود .

تعداد ورودیهای کاربر، خروجی های کاربر، فایلها، واسطهای خارجی و درخواستهای کاربر با توجه به ساده یا پیچیده یا متوسط بودنشان در جدول وارد می شوند. هر سازمانی با توجه به معیارهای خودش ساده یا متوسط یا پیچیده بودن عملکرد ها را در نظر می گیرد. دو معیار دیگر هم با عنوان تعریف کاربر در نظر گرفته شده تا اگر عملکرد دیگری هم مورد نظر کاربر بود محاسبه شود. تعداد در ضریبهای استاندارد نقطه عملکرد ضرب شده و یک مجموع راتشکیل می دهند . فرمول محاسبه نقطه عملکرد به این صورت است:

$$FP = \text{شمارش کل} * [0.65 + 0.01 * \text{مجموع } Fi]$$

شمارش کل عددی است که از جمع تعداد در ضرایب FP بدست می آید و Fi ها عواملی هستند که ضریبی موسم به ضریب پیچیدگی را تعیین می کنند.

زبان برنامه نویسی انتخاب می شود و می توان ضریب پیچیدگی را تعیین کرد دکمه change صفحه تنظیم ضریب پیچیدگی را باز می کند.

- **معیار اندازه گرا**. با انتخاب این گزینه می توان اندازه پروژه را مستقیما بر حسب SLOC وارد کرد.

صفحه مربوط به تنظیم ضریب پیچیدگی:

Complxityfactor_fa.aspx

در FP استاندارد چهارده عامل همانطوری که در شکل می بینید ضریب پیچیدگی را تعیین می کنند:

- ارتباطات داده ای
- عملیات پردازشی توزیع شده
- کارایی
- محیط عملیاتی موجود پر کاربرد
- ساخت ترا کنش ها با سرعت بالا
- مدخل داده های بر خط
- تسهیل تغییرات واستفاده آسان برای کاربر
- بهنگامی برخط
- پیچیدگی فرآیند ها
- قابلیت استفاده مجدد
- طراحی نصب و تبدیل

- پیچیدگی پردازش داخلی
- نصب چندگانه
- پشتیبانی و بازیابی قابل اطمینان

که این مقادیر را دارند:

- قوی (Strong -> 5.0)
- قابل توجه (Significant -> 4.0)
- متوسط (Average -> 3.0)
- میانی (Intermediate -> 2.5)
- ملایم ومحدود (Moderate -> 1.0)
- هیچکدام (None -> 0.0)

با تغییر هر کدام از عوامل، ضریب پیچیدگی محاسبه می شود و در بالای صفحه همانطوری که در شکل مشاهده می شود نمایش داده می شود. با دکمه OK به صفحه اندازه گیری برمی گردیم و محاسبه با ضریب جدید انجام می شود.

صفحه ورود هزینه ها.

هزینه ای که برای هر نفر-ماه در نظر گرفته شده برای هر قسمت از پروژه در قسمت مربوطه وارد می شود.قسمتهای پروژه نیازمندیها، طراحی محصول، طراحی جزئیات محصول، کد نویسی و تست واحدو تست جامعیت است. شکل زیر صفحه هزینه ها را نشان می دهد:

Cost_fa.aspx

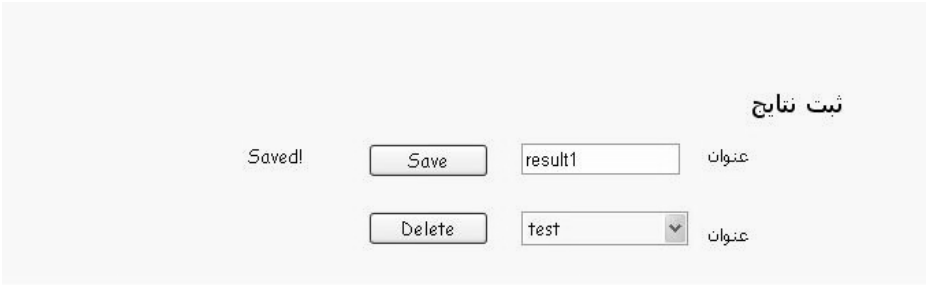
دکمه Result در هر مرحله ما را وارد صفحه نتایج می کند
صفحه نتایج چند قسمت دارد

- **قسمت نمایش نتایج**. نرخ تولید محصول، هزینه، زمان و کار محاسبه شده برای انجام پروژه را نشان می دهد. نتایج جداگانه برای نیازمندیها و توسعه سیستم محاسبه می شود و جمع کل هم نمایش داده می شود.تنها یک فرمول برای محاسبه کار ، زمان توسعه سیستم وجود دارد ولی ضرایب توزیع امکان محاسبه جداگانه را بوجود می آورند.نتایج برآورد فرضی که انجام شده را در شکل می بینید.

Result_fa.aspx

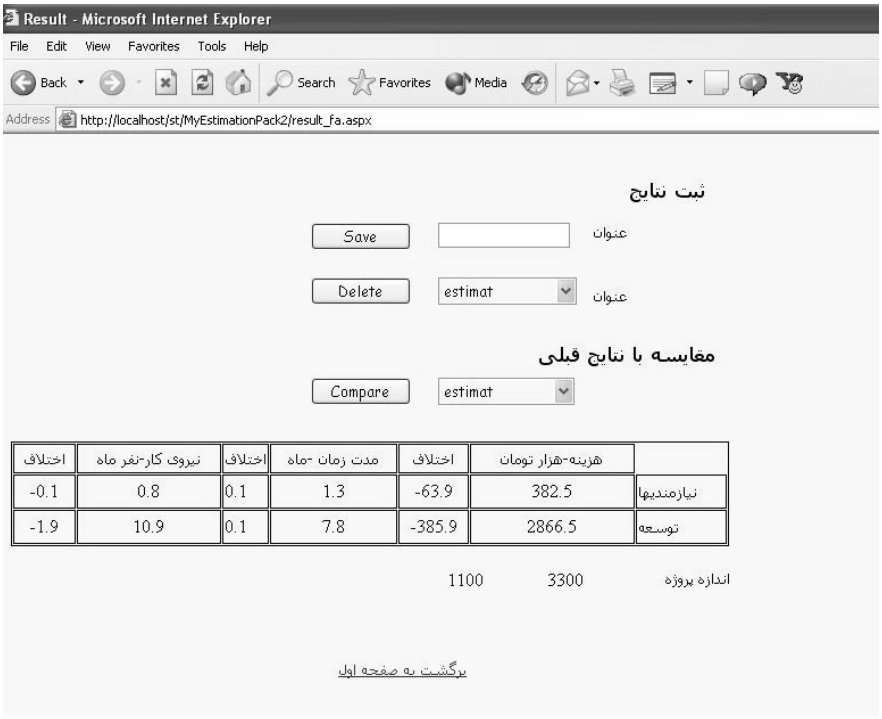
۵-۱-۲ ثبت و مقایسه نتایج

قسمت بعدی ثبت نتایج است. نتایج بدست آمده در یک پایگاه داده ثبت می شوند تا برای استفاده های بعدی در دسترس باشند. اندازه، هزینه، زمان و کار ثبت می شوند. همچنین می توان یک نتیجه ثبت شده را حذف کرد.



Result_fa.aspx

قسمت بعدی امکان مقایسه نتایج بدست آمده را با با نتایجی که قبلا ثبت شده ایجاد می کند.فرض کنید می خواهیم نتایج واقعی را بعد از انجام پروژه با نتایج برآورد های قبلی مقایسه کنیم در صفحه ای که نتایج واقعی وارد شده برآورد مورد نظر را از لیست مقایسه انتخاب کرده و دکمه Compare را فشار می دهیم نتیجه در شکل مشاهده می شود

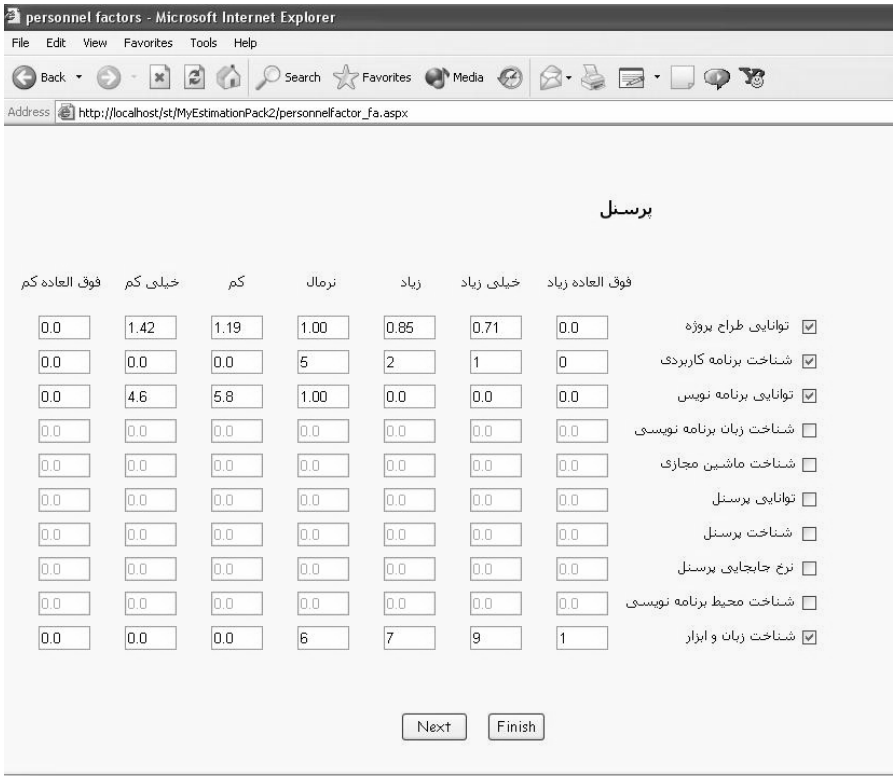


Result_fa.aspx

۳-۱-۵ ایجاد مدل برآورد

با استفاده از لینک برگشت به صفحه اول به صفحه اصلی بر می گردیم و مراحل ایجاد یک مدل برآورد را می بینیم.

اولین کار تنظیم Cost Driver هاست.اینکه کدام ها در مدل موثرند و میزان تاثیرشان چقدر است تک تک ضرایب باید مقدار دهی شوند.در کنار هر کدام یک check box وجود دارد علامت زدن هر چک باکس نشان می دهد که آن عامل را برای برای مدل برآوردمان انتخاب کرده ایم و باید ضرایبش را وارد کنیم. Cost Driver ها در چند صفحه مشخص می شوند که برای نمونه یکی از آنها را در شکل می بینید.



Personnelfactor_fa.aspx

به این ترتیب طی چند صفحه تمامی Cost Driver ها شناسایی شده مقدار می گیرند.در صفحه انجام برآورد تنها ضرایبی که مقدار غیر صفر دارند دیده می شوند.

در صفحه بعد Scale Factor ها تنظیم می شوند، ضرایب معادله های کار و زمان و همچنین ضرایب توزیع وارد می شوند. معادلات کار و زمان همانطور که قبلا گفته شد یک ضریب و دو توان دارند که بطور تجربی محاسبه شده اند. فرمولها به این صورت هستند

$$\text{Effort} = Ae * EAF * (KSLOC)^{Be} + Ce * \text{Sum of scale factors} / 100$$
$$\text{Schedule} = As * (\text{Effort})^{Bs} + Cs * \text{sum of scale factors} / 100$$

Ae, Be, Ce, As, Bs, Cs ضرایبی هستند که کاربر در این صفحه وارد می کند. EAF از حاصلضرب مقادیر Cost Driver ها بدست می آید . KSLOC اندازه محاسبه شده برای پروژه است.دکمه Finish مدل برآورد ساخته شده را ثبت کرده و صفحه اول را باز می کند.

Editscalefactor_fa.aspx

۲-۵ مقایسه با **Costar** :

این نرم افزار بسیار مشابه Costar است و در حقیقت با توجه به آن ساخته شده و مانند آن برای انجام برآورد با متد های مختلف کوکومو مناسب است که تمامی فرمولها و جرئیاتش در اختیار همه قرار دارد. نتایجی که از این نرم افزار حاصل می شودمشابه نتایج Costar است و امکانات هر دو تقریبا برابر است اما تفاتهایی هم وجود دارد:

- اولین و مهمترین تفاوت.این یک نرم افزار تحت وب است و بدون وابستگی به دستگاه هر جا که امکان دسترس به اینترنت وجود داشته باشد می توان برآورد را انجام داد.
- Costar برای ایجاد یک مدل جدید برآورد احتیاج به نرم افزار مکملی چون Calico داردتا روشهای محاسبه جدید ساخته و به Costar وارد شوند در حالیکه این امکانات یکجا در این نرم افزار وجود دارد.

- نتایج حاصل از برآورد در costar بر روی hard disk ثبت می شود ولی در این نرم افزار در یک پایگاه داده نگهداری شده همواره قابل دسترس است
- امکان مقایسه برآورد ها در costar وجود ندارد

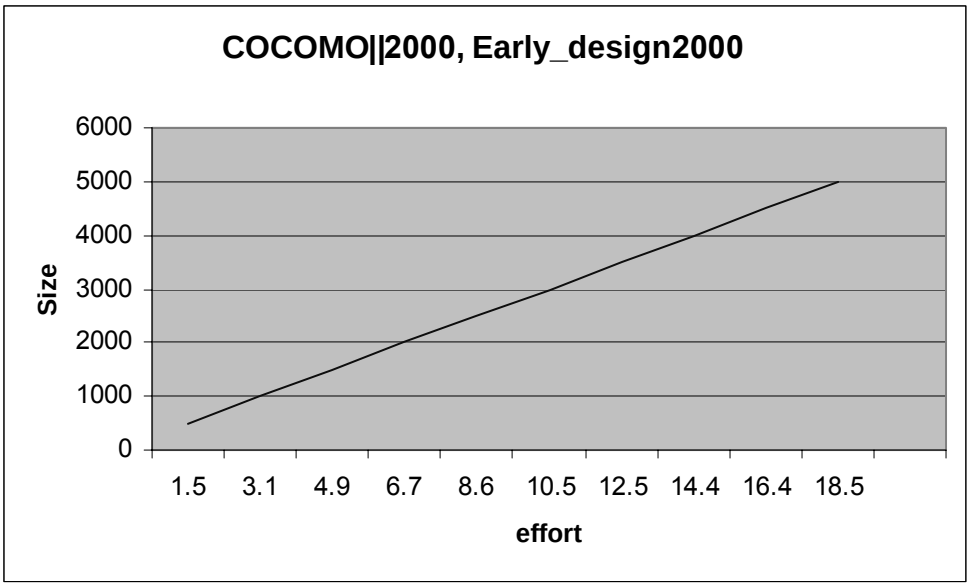
فصل ششم –
مقایسه مدلهای برآورد

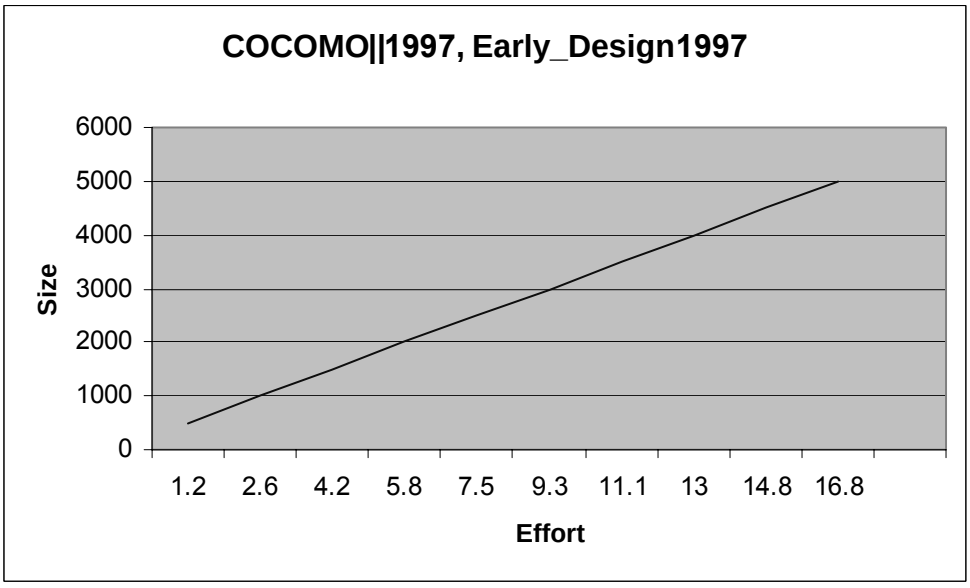
۶-۱ مقایسه نتایج حاصل از مدلهای برآورد مختلف

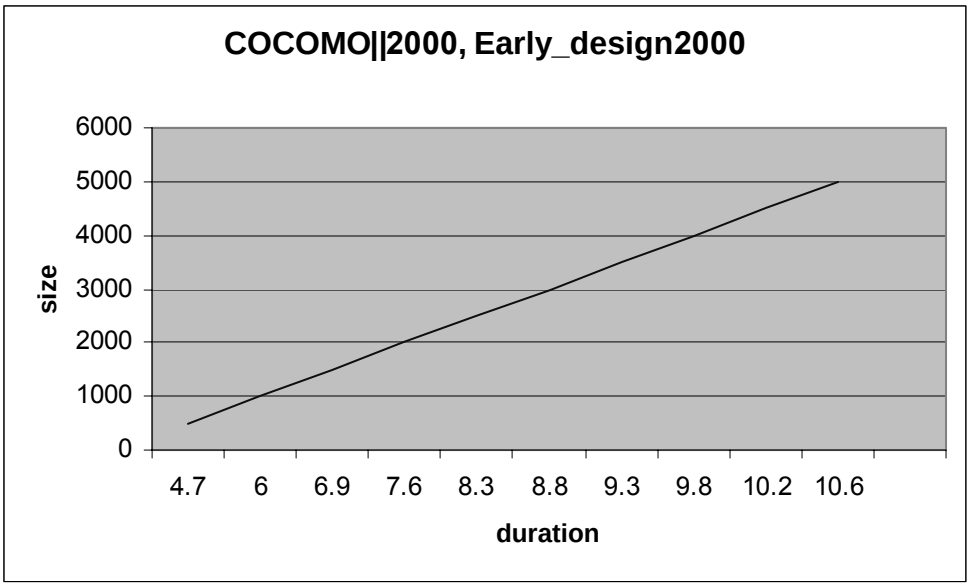
چهار مدل برآورد باهم مقایسه شده اند: Early_Design2000, Early_Design1997 ,COCOMO||2000 , COCOMO||1997,
به این ترتیب که یک پروژه با شرایط یکسان با هر چهار مدل برآورد شده و نتایج روی نمودار نمایش داده شده است. شرایط یکسان به این ترتیبند که تمامی Cost Driver ها جزئی در نظر گرفته شده که شامل اینها هستند: توانایی طراح پروژه، شناخت برنامه کاربردی، توانایی برنامه نویس، شناخت محیط برنامه نویسی، شناخت زبان و ابزار، نرخ جابه جایی پرسنل، استفاده از ابزار های نرم افزاری ، پراکندگی سایتها، زمانبندی پیشبرد پروژه، مدت زمان اجرا، منابع ذخیره کننده، تغییر ناگهانی محیط، قابلیت اطمینان لازم، اندازه پایگاه داده، پیچیدگی محصول، قابلیت استفاده مجدد و اسناد تهیه شده.
و Scale factor ها به این ترتیب تنظیم می شوند:

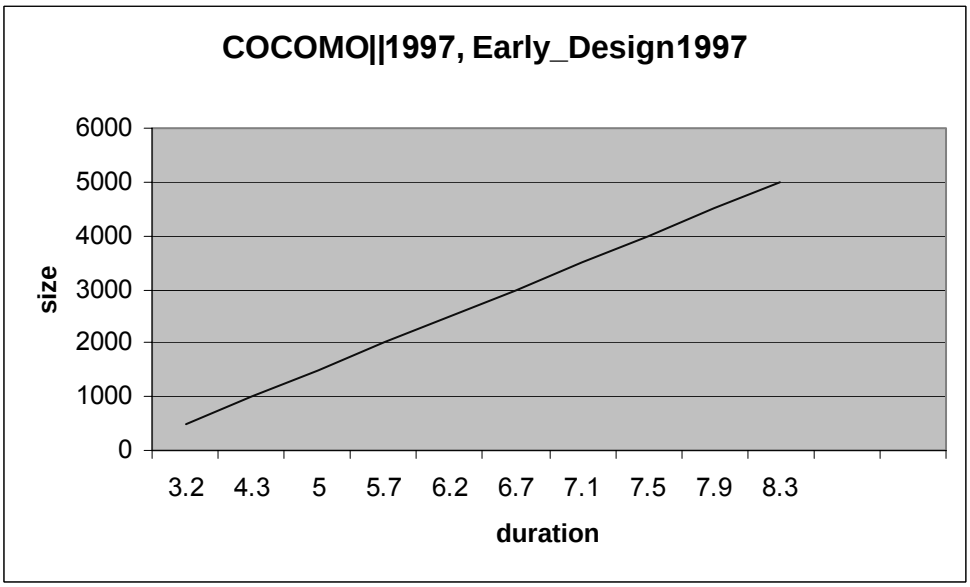
- سابقه انجام کار مشابه.آیا قبلا هم پروژه ای شبیه این انجام شده است یا نه؟ تقریبا بدون سابقه است
- آیا نیازمندیهای پروژه قابل انعطاف هستند یا باید همه موارد را در نظر گرفت؟ آیا نرم افزار یا سخت افزار واسط قیدهای خاصی برای پروژه بوجود می آورند؟ فشار خفیفی وجود دارد
- طراحی معماری قبلا تا چه حد تعریف شده و مدیریت ریسک تا چه حد انجام گرفته؟ ٪۶۰
- روابط مشتری، توسعه دهندگان، کاربران و کسانی که نگهداری سیستم را به عهده دارن تا چه میزان است؟ هماهنگی پایه ای
- در آغاز پروژه در چه سطحی قرار دارید؟ فرآیند های نرم افزاری استانداردی وجود دارد

با این شرایط اندازه پروژه تغییر داده شده و نتایج برآورد روی نمو دارها مشاهده می شود:


COCOMO||2000, Early_design2000
6000
5000
4000
3000
2000
1000
0
Size
1.5
3.1
4.9
6.7
8.6
10.5
12.5
14.4
16.4
18.5
effort


COCOMO||1997, Early_Design1997
6000
5000
4000
3000
2000
1000
0
Size
1.2
2.6
4.2
5.8
7.5
9.3
11.1
13
14.8
16.8
Effort



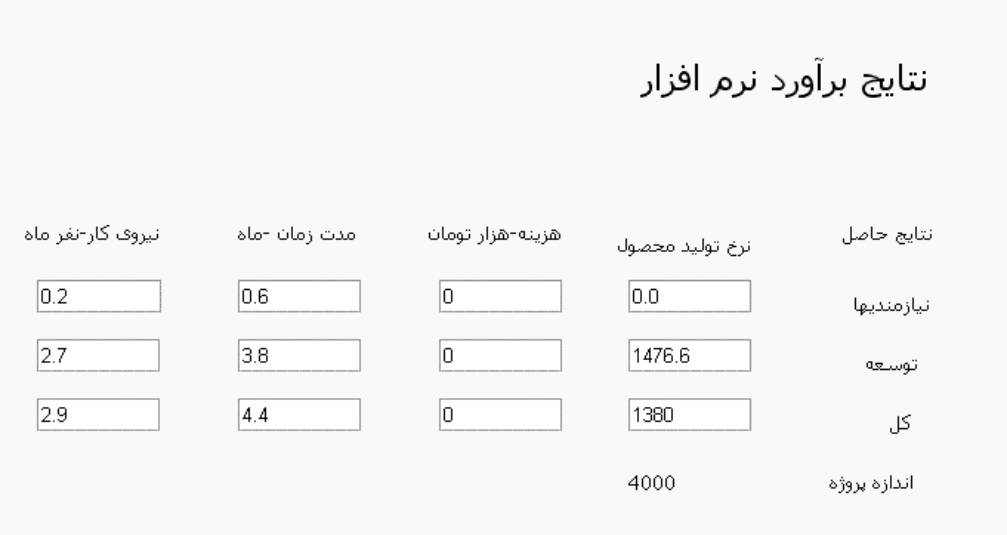


همانطور که مشاهده می شود نتایج حاصل از مدلهای سال ۲۰۰۰ کاملا مشابه است و همینطور مدلهای سال ۱۹۹۷ هم نتایج مشابهی دارند تفاوت این مدلها در Cost Driver هاست در هر سازمان با توجه به مواردی که در برآورد مهم اند می توان یکی از این دو مدل را انتخاب کرد.

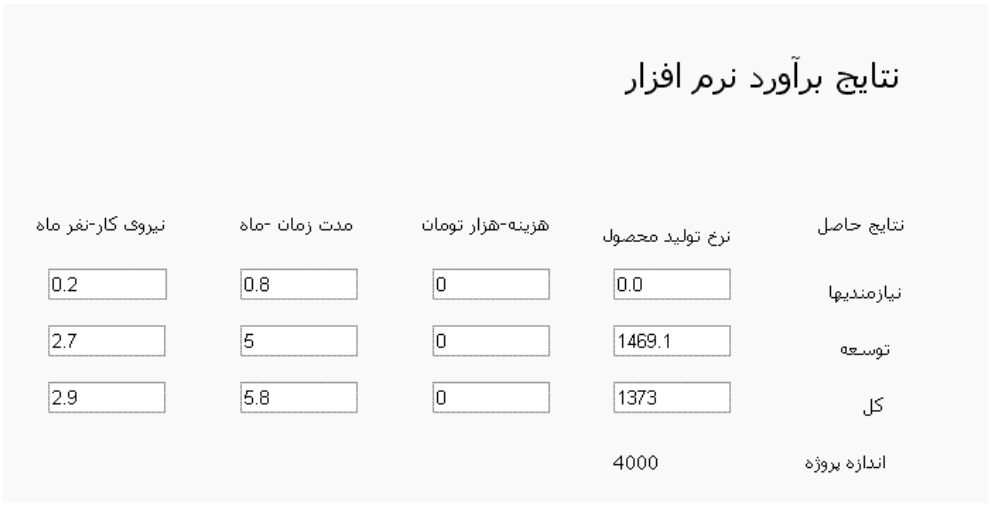
نکته قابل توجه دیگر این است که اعداد بدست آمده از مدلهای جدیدتر چه در میزان نیروی کار و چه در میزان طول انجام پروژه بزرگتر است . یعنی در شرایط یکسان و برای پروژه های با اندازه یکسان مدلهای جدید تر اعداد بزرگتری را برآورد می کنند. هر چند این اعداد تفاوت چشمگیری با هم ندارند و احتمالا از پیشرفت تکنولوژی ها و ابزار برنامه سازی بوجود آمده اند.

۲-۶ انجام برآورد برای یک پروژه واقعی

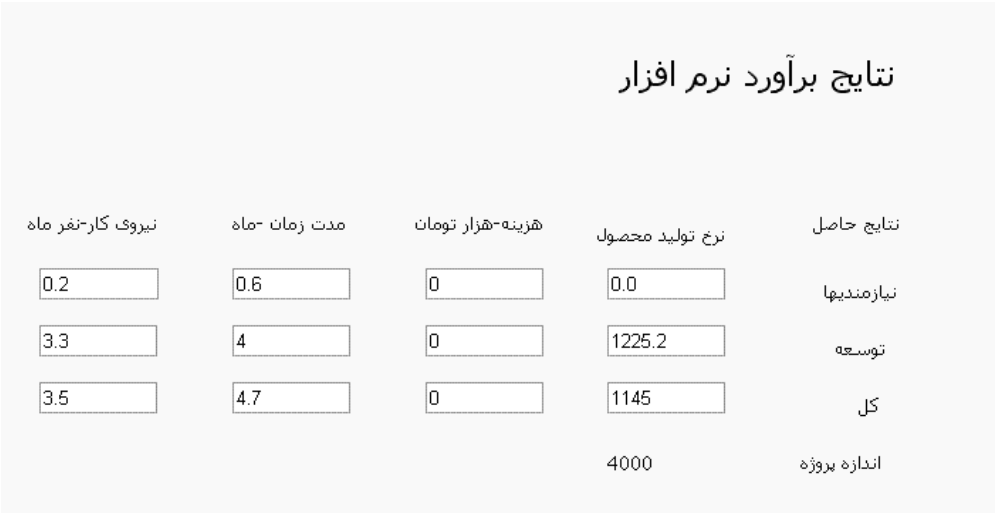
پروژه واقعی که مورد برآورد قرار گرفته یک سیستم تعیین کیفیت نرم افزار است که در زمان انجام تخمین در مراحل پایانی انجام بوداطلاعات مورد نیاز بوسیله مدیر پروژه تکمیل شد و برآورد به کمک سیستم و با چهار مدل برآورد انجام شد نتایج حاصل از انجام برآورد تقریبا با نتایج واقعی مطابقت دارد :



COCOMO||1997



COCOMO||2000



Early_Design1997

Early_Design2000

پروژه موردنظر با ۳ نفر ماه نیروی کار در حدود ۵ ماه مورد نظر مدل برآورد یعنی ۱۵۲ ساعت کار انجام شده.

نتیجه گیری

مهمترین عامل در انجام یک برآورد موفق داده های تاریخی یک سازمان و تجربیات شخصی و توانایی مدیر پروژه در استفاده از آنهاست. بدون اینها انجام برآورد تقریبا غیر ممکن است. نتایجی که مدلهای برآورد محاسبه می کنند اگر اطلاعات درستی در اختیار باشند مشابه اند و لی برای اطمینان بهتر است از چند مدل برای برآورد استفاده کنیم.

مراجع

1. Pressman,R. **Software Engineering**. MC Graw-Hill,2001

۲. جعفر نژاد قمی، عین اله. **مرجع کامل ASP.NET** . نشر علوم ، ۱۳۸۳

3. Sommervil,I. **Software Engineering**. Addition-Wesley Publishing Company, 1998

4. Lewis,J. **Larg limits to Software Estimation**. Disney TSl, 2001

5. Boehm,B. and Fairly,R. **Software Estimation perspective**. IEEE Software,2000

6. Duthie,G. **ASP.NET step by step**. Microsoft Press,2001

7. Dorfam,M. and Bohem,B. **Software Estimation**.I EEE Computer Society Press, 1997

8. Hemmstra,F. **Software Cost Estimation**. Software Technology and Information,1992

9. Smith,J. and Kenneth, B. **Engineering Quality Software**, Elsevier Science Publishers LTD,1992