# فصل ۵ طرح ریزی پروژه نرم افزاری

**مفاهیم کلیدی (مرتب بر حروف الفبا)**

**ابزارهای خودکار ، استفاده از منابع خارجی ، امکان سنجی ، برآورد ، برآورد مبتنی برفرآیند ، برآورد مبتنی بر مسئله ، تصمیم خرید/ساخت ، حوزه نرم افزار ، فنون تجزیه ، مدلهای تجربی ، منابع**

**KEY CONCEPTS**

**automated tools , decomposition techniques , empirical models , estimation , feasibility , make/buy decision , outsourcing , problem-based estimation , process-based stimation , resources , software Scope**

## نگاه اجمالی

برنامه‌ریزی پروژه نرم‌افزاری چیست؟ این کار درواقع دربرگیرنده تمام کارهایی است که ما در فصول ۵ تا ۹ مورد بحث قرار دادیم. در هر حال در این فصل، برنامه ریزی شامل تخمین می‌باشد یعنی تلاش شما برای تعیین میزان و پول، کار لازم، تعداد منابع و مقدار زمان لازم برای ایجاد یک سیستم یا محصول خاص مبتنی بر نرم‌افزار.

چه کسی این کار را می‌کند؟ مدیران نرم‌افزاری با استفاده از اطلاعات جمع‌آوری شده از مشتریان و مهندسان نرم‌افزاری و داده‌های متریک نرم‌افزاری حاصل از پروژه های گذشته این کار را انجام می‌دهند.

چرا این کار مهم است؟ آیا شما خانه را بدون اینکه بدانید چقدر مخارج باید صرف کنید، می‌سازید؟ مسلماً خیر، و از آنجا که اکثر سیستم های مبتنی بر کامپیوتر و اکثر محصولات تاحد بسیار زیادی گرانتر از یک خانه خرج دارند، منطقی است که قبل از شروع به ایجاد نرم‌افزار، برآوردی از هزینه انجام دهیم.

مراحل کار چیست؟ تخمین با توصیف دامنه محصول شروع می‌شود. تا وقتی که دامنه محدود است، ممکن نیست که برآورد درستی ارائه دهیم. هر مسئله متشکل از مجموعه‌ای مسائل کوچکتر بوده و هرکدام از آنها با استفاده از اطلاعات و تجربیات بعنوان یک راهنما، برآورد می‌شود. توصیه می‌شود که تخمین خود را حداقل با استفاده از دو روش متفاوت انجام دهید. پیچیدگی مسئله و خطر قبل از برآورد نهایی درنظر گرفته می‌شوند.

**محصول نهایی کار چیست؟** جدول ساده ای که جزئیات کارهایی را که باید صورت بگیرند، توابعی که باید اجرا شوند و هزینه، تلاش، زمان لازم برای هر یک را نشان می‌دهد. فهرستی از منابع لازم نیز ارائه می‌گردد.

**چگونه مطمئن شوم که کارم را به درستی انجام داده‌ام؟** این کار مشکل است زیرا تا وقتی که پروژه تکمیل نشود از آن حقیقتاً مطمئن نخواهید شد. اگر با تجربه باشید و از روشی نظام‌مند پیروی کنید، تخمین ها را با استفاده از اطلاعات تاریخی صرف ایجاد نموده ، نکات اطلاعاتی برآورد را با استفاده از حداقل دو روش مختلف و فاکتور پیچیدگی و خطر ارائه نموده و می توانید مطمئن باشید که به بهترین نحو کار خود را انجام داده‌اید.

مدیریت پروژه نرم‌افزاری با مجموعه فعالیتهایی آغاز می‌شود که جمعاً برنامه‌ریزی پروژه یا Project Planning نامیده می‌شوند. قبل از اینکه پروژه بتواند آغاز شود مدیر و تیم نرم‌افزاری باید کاری را که قرار است انجام گیرد، منابع لازم و زمان لازم از شروع تا انتها را تخمین بزنند. هرگاه تخمین‌ها برآورده شد نگاهی به آینده انداخته و میزانی از عدم قطعیت را بعنوان موضوع مورد بحث می‌پذیریم. به گفتهٔ فردریک بروکز: [BRO75][1] تکنیکهای برآورد، فقیر و ضعیفند....

با اینکه کار تخمین بیشتر یک هنر است تا یک علم اما این کار مهم لازم نیست الزاماً به شیوه‌ای تصادفی و بی‌حساب صورت گیرد. فنون مفیدی برای تخمین زدن زمان و کار لازم وجود دارند. متریکهای پروژه و فرآیند می توانند دیدگاه تاریخی و اطلاعات قدرتمندی برای تولید و ارائه برآوردهای سطح بالا مهیا نمایند. تجربه گذشته می‌تواند در ارائه و بازنگری برآوردها کمک فراوانی به ما بکند. از آنجا که تخمین بر پایه دیگر کارهای برنامه‌ریزی پروژه است و برنامه‌ریزی پروژه مسیر منتهی به مهندسی و طراحی موفقیت آمیز پروژه را مهیا می‌کند، بدون آن کارمان پیش نخواهد رفت.

### ۵-۱ مشاهده برآوردها

نقل قول

رهیافت های مناسب برآورد و داده های جمع آوری شده تاریخی، امیدوارکننده ترین چیزی هستند که درخواست های غیر ممکن را عملی می سازند.
کاپرز جونز

زمانی از یکی از مدیران اجرایی مهم پرسیده شد در هنگام انتخاب مدیر پروژه مهمترین مشخصه‌ای که مدنظرتان است چیست؟ پاسخ او این بود: «.... شخصی با توانایی شناخت آنچه احتمالاً به‌خطا می رود قبل از اینکه واقعاً این مسئله رخ دهد ...» ما اضافه می‌کنیم: «و شجاعت پیش‌بینی این امر که آینده چندان مساعد نیست.»

برآورد منابع، هزینه و زمانبندی کار طراحی نرم‌افزار نیازمند تجربه، دسترسی به اطلاعات خوب تاریخی و شجاعت بیان پیش‌بینی های سطح بالا و درست در مواقعی است که اطلاعات کیفی بطور کامل

1.Brooks, F.

وجود دارند. کار تخمین بطور بالفطره دارای خطراتی[1] هست و این خطر است که منجربه عدم قطعیت در کار می‌شود.

پیچیدگی پروژه تأثیر شدیدی روی عدم قطعیت دارد که در برنامه‌ریزی نهفته است. پیچیدگی و سخت بودن کار یک معیار نسبی است که تحت تأثیر آشنایی با کار و تجربه گذشته است. تولیدکننده یک برنامه کاربردی پیچیده تجارت الکترونیکی که اولین بار آن را انجام می‌دهد باید آن را بسیار مشکل بداند، اما تیم نرم‌افزاری که این کار را برای ده بار انجام داده ، آن را بسیار سهل می‌داند. یک سری معیارهای نسبی درمورد میزان پیچیدگی نرم‌افزار پیشنهاد شده‌اند [ZUS97][2]. چنین معیارهایی در سطح کد یا طراحی بکار گرفته شده و بنابراین استفاده از آنها در طول برنامه‌ریزی (قبل از اینکه طرح یا کدی وجود داشته بشاد) سخت است. سایر برآوردهای نظری تر درمورد پیچیدگی (مثل تابعی که به عوامل تطابق پیچیده اشاره می‌کند که در فصل ۴ آمده) را می‌توان در مراحل اولیه برنامه ریزی ایجاد نمود.

پیچیدگی پروژه، اندازه پروژه و درجه عدم اطمینان ساختیافتگی آن همه و همه بر قابلیت اطمینان برآوردها ، تاثیر گذار می باشند.

اندازه بزرگی پروژه[3] عامل مهم دیگری است که می تواند بر دقت و کارآیی تخمین‌ها تأثیر بگذارد. با افزایش اندازه بزرگی وابستگی درونی میان عناصر مختلف نرم‌افزار به سرعت رشد می‌کنند.[4] تجزیه مسئله که روش مهم دیگری برای تخمین زدن است، سخت‌تر می‌باشد زیرا عناصر تجزیه شده ممکن است سخت تر و مشکل تر باشند. به تعبیری دیگر قانون مورفی می‌شود: «آنچه که می‌تواند به خطا برود به خطا خواهد رفت» و اگر چیزهای بیشتری وجود دارند که محکوم به شکست هستند، چیزهای بیشتری نیز به شکست خواهند انجامید.

میزان عدم قطعیت ساختاری[5] نیز روی تخمین خطر تأثیر دارد. در اینجا، ساختار اشاره دارد به درجه‌ای که نسبت به آن شرایط سخت شده، میزان راحتی که با آن می‌توان توابع و عملکردها را بخش‌بندی نمود و ماهیت سلسله مراتبی اطلاعاتی که باید پردازش شوند.

نقل قول

نشانه فکر غیرساختاری ، آن است که با درجه ای از دقت که طبیعت یک موضوع می طلبد، بسازیم و در جستجوی حقیقتی که می تواند در دسترس باشد ( یا تقریبی بالاتر) برنیاییم.

آریستوتل ارسطو

در دسترس بودن اطلاعات تاریخی دارای تأثیر شدیدی بر تخمین خطر است. با نگاهی به گذشته، می توانیم بر چیزهایی که جواب داده‌اند، برتری یافته و در حوزه‌های پیشروی کنیم که در آنها مشکلات ایجاد شده‌اند. وقتی متریکهای نرم‌افزاری جامعی (فصل ۴) درمورد پروژه‌های گذشته در اختیار داشته باشیم، می‌توان کار برآورد را با دقت بیشتری انجام داد. می‌توان به منظور پرهیز از مشکلات قبلی، جداول برنامه‌ریزی درست کرده و خطر کلی را کاهش داد.

---

۱. فنون سیستماتیک برای تحلیل ریسک و مخاطره در فصل ۶ آمده است.

2.Zuse, H.

3.project size

۴. اندازه و سایز، اغلب با " گسترش حوزه " افزایش می یابد و آن هنگامی است که مشتری نیازمندیهایش را تغییر می دهد. افزایش سایز پروژه تاثیر مستقیم بر هزینه و زمان بندی خواهد داشت.

5.degree of structural uncertainty

خطر با میزان عدم قطعیت در انجام برآوردهای کیفی برای منابع، هزینه و جداول زمانبندی سنجیده می‌شود. اگر دامنه پروژه‌ای به خوبی شناخته نشده یا نیازمندیهای آن در معرض تغییر باشند، عدم اطمینان به میزان خطر به شدت در آن بالا می‌رود. طراح نرم‌افزار باید خواستار تکمیل تعاریف عملکرد، تابع و رابطی شود که در مشخصات سیستم[1] قرار دارند. همچنین یک برنامه‌ریز و مهمتر از آن مشتری، باید بدانند که تنوع در نیازمندیهای نرم‌افزار به معنی عدم ثبات در هزینه و زمانبندی است.

مدیر پروژه نباید درمورد تخمین وسواس بخرج دهد. شیوه‌های مدرن مهندسی نرم‌افزار (مثل مدلهای فرآیند تکمیلی) دیدگاه تکراری از تولید دارند. در چنین روشهایی، ممکن[2] است میزان برآورد را مجدداً بازبینی نمود. (زیرا اطلاعات بیشتری مشخص شده) و وقتی مشتری در نیازمندیها تغییراتی ایجاد می‌کند آن را اصلاح کنیم.

توصیه

هرچه بیشتر بدانید، برآورد بهتری خواهید داشت. بنابرین با بیشرفت پروژه، تخمین های خود را به هنگام کنید.

## ۵-۲ اهداف اصلی طرح ریزی پروژه

هدف برنامه‌ریزی پروژه نرم‌افزاری عبارت است از مهیا کردن چارچوبی که مدیر را قادر به ارائه تخمین‌های منطقی از منابع، هزینه و زمانبندی کند. این تخمین ها در یک چارچوب زمانی در آغاز پروژه صورت گرفته و مرتباً با پیشروی پروژه، به روز می‌شوند. علاوه بر این، تخمین‌ها تلاش دارند که بهترین و بدترین مورد طرحها را معین کنند تا نتیجه پروژه را بتوان محدود نمود.

هدف برنامه‌ریزی، از طریق فرآیند کشف اطلاعاتی حاصل می‌گردد که منجر به تخمین‌های منطقی می‌شوند. در بخشهای بعدی، هر یک از فعالیتهای مربوط به برنامه‌ریزی پروژه مورد بحث قرار می‌گیرند.

## ۵-۳ دامنه نرم افزار

اولین کار در برنامه‌ریزی پروژه عبارتست از تعیین دامنه آن. کارکرد و عملکرد مربوط به نرم‌افزار درطول مهندسی سیستم باید ارزیابی گردد تا دامنه‌ای ایجاد کند که غیر مبهم و در سطوح مدیریتی و فنی قابل درک باشد. گزارشی از این دامنه باید پیوست شود.

دامنه نرم‌افزار اطلاعات و کنترل مورد پردازش، عملکرد، کار، محدودیتها، رابطها و قابلیت اطمینان را توصیف می‌کند. عملکردهای توصیفی در گزارش دامنه ارزیابی شده و در بعضی موارد اصلاح می‌گردد تا قبل از شروع عملیات تخمین جزئیات بیشتری مهیا گردد. از آنجا که تخمین هزینه و زمانبندی هر دو ازنظر علمی با هم سازگار شده‌اند، اغلب مقداری تفکیک مفید است. الزامات و قیود محدودیتهای موجود در نرم‌افزار را بوسیله سخت افزار خارجی یا حافظه موجود یا دیگر سیستم‌های موجود شناسایی می‌کنند.

---

1.System Specification

۲. این بدان معنا نیست که همواره سیاستها و سیاسی کاری ها برآورد اولیه را خدشه دار می کند. یک سازمان بالغ نرم افزاری و مدیرانش می دانند که تغییرات رایگان و بدون هزینه نخواهند بود. ولی هنوز بسیاری از مشتری ها درخواستهایی ( غیرصحیح ) دارند که برآورد اولیه را با اشکال مواجه می سازد.

### ۵-۳-۱ دستیابی به اطلاعات مورد نیاز برای دامنه

در شروع پروژه نرم‌افزاری بعضی از چیزها همیشه مبهم و نامشخص هستند. نیازی بیان گردیده و اهداف اولیه آن تشریح شده‌اند، اما اطلاعات لازم برای تعریف دامنه (پیش‌نیاز تخمین) هنوز معلوم نشده‌اند.

رایج ترین تکنیک مورد استفاده برای برقراری ارتباط بین مشتری و تولیدکننده و شروع فرآیند عبارتست از برگزاری یک جلسه یا مصاحبه. اولین جلسه بین مهندس نرم‌افزار (تحلیلگر) و مشتری را می‌توان به مثابه ناراحتی اولین قرار ملاقات بین دو جوان دانست. هیچکدام نمی‌دانند چه بگویند و چه چیزی بپرسند. هر دو نگرانند که نچه که می‌گویند مورد برداشت نادرست قرار گیرد، هر دو درباره انجام کار فکر می‌کنند (هر دو اساساً انتظارات کاملاً متفاوتی دارند)، هر دو می‌خواهند کار انجام شود اما در یک زمان و هر دو می‌خواهند موفق باشند.

پرسش

چگونه می توانیم ارتباط و محاوره مناسب بین توسعه دهندگان و مشتریان را برقرار سازیم؟

با این‌حال ایجاد ارتباط باید آغاز شود. گاز و وینبرگ [GAU89][1] بیان می‌دارند که تحلیلگر با پرسیدن سئوالات آزاد مستقل از متن[2] آغاز می‌کند. یعنی مجموعه‌ای از سئوالاتی که منجر به درک مقدماتی مسئله، شناخت افرادی که در جستجوی راه حل هستند، ماهیت راه حل مطلوب و میزان تأثیر اولین برخورد آن می‌شود.

اولین مجموعه سئوالات آزاد روی مشتری متمرکز است و بعد اهداف کلی و مزایا، مثلاً، تحلیلگر ممکن است بپرسد:

- چه کسی در پشت تقاضای این کار است؟
- چه کسی از این راه حل استفاده می‌کند؟
- مزایای اقتصادی یک راه حل موفق چیست؟
- آیا منبع دیگری برای این راه حل وجود دارد؟

مجموعه سئوالات دیگری تحلیلگر را قادر به شناخت بهتری از مسئله و مشتری نموده تا برداشتهایش را درمورد راه حل ارائه دهد:

- شما چگونه بازده خوب را که توسط یک راه حل موفق ارائه می شود، توصیف می‌کنید؟
- این راه حل چه مشکلاتی را مورد خطاب قرار می‌دهد؟
- آیا شما می توانید محیطی را به من نشان دهید (یا توصیف کنید) که در آن راه حلی استفاده نشود؟
- آیا موضوعات عملکردی خاص یا محدودیتهایی وجود دارد که بر شیوه راه حل مورد استفاده تأثیر بگذارند؟

---

1.gause, D.C

2.context-free questions

مجموعه سئوالات آخری بر میزان تأثیر جلسه تأکید دارد. گاز وینبرگ آن را فوق پرسش می‌نامند و فهرست زیر را پیشنهاد می‌کنند:

- آیا شما شخص مناسبی برای پاسخ به این سئوالات هستید؟ آیا این سئوالت رسمی هستند؟
- آیا سئوالات من با مسئله ای که مورد حل قرار داده‌اید مرتبطند؟
- آیا من سئوالات زیادی پرسیده‌ام؟
- آیا کس دیگری وجود دارد که اطلاعات اضافه‌ای مهیا کند؟
- آیا چیز دیگری هست که بتوانم از شما سئوال کنم؟

این سئوالات کمی به تحرک اولیه مسئله کمک کرده و ارتباطی را که برای ایجاد دامنه پروژه لازم است ایجاد می‌کنند. اما قالب یک جلسه پرسش و پاسخ روشی نیست که بسیار موفق بوده باشد. در حقیقت بخش سئوال و جواب تنها باید برای اولین برخورد استفاده شده و سپس قالب جسله‌ای جایگزین آن شود که عناصر حل مسئله، مذاکره و مشخصات را مرتبط می‌سازد.

ارجاع به فصلها

تکنیکهای کسب نیازمندیها در فصل ۱۱ توضیح داده شده اند.

مشتری و مهندسان نرم‌افزار اغلب دارای تصور ناآگاهانه‌ای از «ما و آنها» هستند. بجای کار بعنوان یک تیم برای شناسایی و تفکیک نیازمندیها، هر مشتری سرحدات[1] خود را معین نموده و از طریق یک سری یادداشت، اوراق رسمی وضعیت، اسناد و بخشهای پرسش و پاسخ ارتباط برقرار می‌کند. تاریخ نشان داده که این روش چندان موفق نبوده است. برداشتهای نادرستی ایجاد شده، اطلاعات مهم حذف شده و رابطه کاری موفق هرگز ایجاد نمی‌شود.

با تصور چنین مشکلاتی است که تعدادی از محققین مستقل رهیافت تیمی درمورد جمع‌آوری نیازمندیها ارائه داده‌اند که می‌توان از آن برای کمک به ایجاد دامنه پروژه کمک گرفت. این روش با نام فنون ساده شده مشخصات برنامه های کاربردی[2] یا FAST، ایجاد تیم مشترکی از مشتریان و تولیدکنندگانی که برای حل مسئله با هم کار می‌کنند را تشویق نموده، عناصر راه حل را پیشنهاد کرده، روشهای مختلفی را مورد مذاکره قرار داده و مجموعه اولیه‌ای از نیازمندیها را مشخص می‌سازد.

نقل قول

۱۰۶ کیلومتر تا شیکاگو راه داریم، با یک باک پر از بنزین و نصف پاکت سیگار و درحالیکه هوا تاریک است، عینک آفتابی به چشم زده اید.
برادران بلوز

### ۵-۳-۲ امکان سنجی

وقتی دامنه شناسایی شد (ضمن هماهنگی با مشتری)، منطقی است بپرسیم که : «آیا می‌توانیم نرم‌افزاری بسازیم که این دامنه را برآورده کند؟» آیا این پروژه شدنی است. اغلب مهندسین نرم‌افزار از این سئوالات صرفنظر می‌کنند (یا بوسیله مشتری یا مدیران بدون صبر مجبور به نادیده گرفتن آنها می‌شوند) تا

1. Territory
2. facilitated application specification techniques(FAST)

صرفاً در پروژه‌ای درگیر شوند که از ابتدای آن لعنت شده است. پوتنام و مایرز [PUT97a] [1] این موضوع را مورد بحث قرار داده و می‌گویند:

*... هرچیز قابل تصوری در عمل شدنی نیست حتی در نرم‌افزار و ممکن است برای دیگران و ناظران خارجی زودگذر و محو بنظر برسد. در مقابل، عملی بودن نرم‌افزار دارای چهار بعد صرف است: فنآوری[2]، آیا پروژه ازنظر فنی شدنی است؟ آیا مطابق آخرین پیشرفتهای علمی است؟ آیا می‌توان معایب را به سطحی مطابق با نیازهای کار تقلیل داد، امور مالی[3] - آیا این کار از نظر مالی شدنی است؟ آیا تولید را می‌توان با هزینه‌ای که سازمان مشتری یا بازار بتواند از عهده آن برآید تکمیل نمود؟ زمان[4] - آیا زمان تولید پروژه تا رسیدن به بازار می‌تواند در رقابت پیروز شود؟ منابع[1] - آیا سازمان منابع لازم برای موفقیت را دارد؟*

در مورد بعضی از پروژه‌ها در نواحی معینی پاسخها بسیار ساده است. شما پروژه‌هایی همچون این را قبلاً نیز انجام داده‌اید. بعد از چند ساعت یا گاهی چند هفته از انجام تحقیقات، مطمئن می شوید که می‌توانید این کار را انجام دهید.

پروژه‌هایی که از نظر تجربه در حاشیه قرار دارند، چندان به راحتی پاسخ خود را نمی‌گیرند. ممکن است تیمی چند ماه وقت صرف فهمیدن این موضوع کند که شرایط اصلی و صعب‌الاجرای *این برنامه حقیقتاً کدامند؟ آیا این شرایط خطراتی را ایجاد می‌کنند که پروژه را غیرممکن می‌سازند؟ آیا می‌توان بر این خطرات فائق شد؟ تیمی که این کار را بررسی می‌کند باید طراحی و معماری اولیه این شرایط پرمخاطره را تاحدی مدنظر داشته باشد که بتواند به این سئوالات پاسخ بگوید. در بعضی از موارد، وقتی تیمی با پاسخ منفی روبرو می‌شود، ممکن است تقلیل نیازمندیها صورت بگیرد.*

*درعین حال، مدیران ارشد با عصبانیت با انگشتان خود روی میزهای بزرگ خود ضرب می‌گیرند. اغلب آنها سیگارهای چاق و چله خود را به شیوه‌ای اشراف منشانه تکان داده و از میان پرده‌ای از دود می‌گویند «کافی است! آن را انجام دهید.»*

*بسیاری از پروژه‌هایی که چند سال بعد خبر شکستشان را با سر و صدا در روزنامه می‌شنویم این گونه آغاز شده‌اند.*

پوتنام و مایرز اعتقاد دارند که تعیین دامنه کافی نیست. وقتی دامنه فهمیده شد، تیم نرم‌افزاری و دیگران باید مشخص سازند که با ابعاد روشن شده، آیا امکان کار وجود دارد یا خیر. که این خود قسمتی از فرآیند تخمین و برآورد خواهد بود.

توصیه

امکان سنجی فنی و تکنیکی مهم است، اما نیازمندیهای تجاری حتی از آن هم مهمتر است. باید توجه داشت که ساختن سیستم با فن آوری بالا، یا محصولی خوب و مناسب که با استقبال واقعی مواجه نشود، امر مطلوبی نخواهد بود.

---

1.Putnam, L.

2.Technology

3.Finance

4.Time

### ۵-۳-۳ یک مثال از تعیین دامنه

ارتباط با مشتری منجر به تعریف اطلاعات و کنترل اطلاعات پردازش شده، توابعی که باید اجراشوند، عملکرد و محدودیتهایی که سیستم را احاطه کرده‌اند و اطلاعات مربوطه می‌شوند. بعنوان مثال، نرم‌افزاری را برای نوعی سیستم مرتب کردن خط حمل و نقل یا CLSS[2] درنظر بگیرید. وضعیت دامنه آن برای CLSS به شکل زیر است:

CLSS جعبه‌هایی را که در طول خط (نقاله) حمل و نقل حرکت می‌کنند مرتب یا دسته‌بندی می‌کند. هر جعبه بوسیله یک بارکُد شناسایی می‌شود که حاوی شماره‌ای جداست و در یکی از شش محفظه انتهای خط قرار دارد. جعبه ها از یک ایستگاه مرتب سازی رد می‌شوند که حاوی یک بارکد خوان و یک کامپیوتر است. این PC به یک مکانیزم تغییر خط متصل است که جعبه‌ها را درون محفظه‌ها مرتب می‌کند. جعبه‌ها بصورت تصادفی عبور نموده و بصورت فرد قرار می‌گیرند. این خط با سرعت پنج فوت در دقیقه حرکت می‌کند. CLSS از نظر شماتیک در شکل۵-۱ آمده است.

نرم‌افزار CLSS اطلاعات ورودی را از یک بارکُدخوان در فواصل زمانی مطابق با سرعت خط حمل، دریافت می‌کند. اطلاعات بارکُد بصورت قالب شناسایی جعبه، از حالت کُد خارج می شوند. نرم‌افزار در پایگاه داده ای ارقام که حاوی حداکثر ۱۰۰۰ سری ورودی است به جستجو می‌پردازد تا محل محفظه مناسب را برای جعبه‌ای که درحال حاضر در خواننده در ایستگاه مرتب سازی است، تعیین کند. محل مناسب محفظه به یک تغییردهنده خط مرتب‌سازی وارد می‌شود که جعبه‌ها را در محفظه مناسب قرار می‌دهد. رکوردی از محفظه مقصد برای هر جعبه از نظر بازیافت و گزارش دهی برای بعدها، حفظ می‌گردد. CLSS اطلاعاتی از تاکومتر پالسی دریافت می‌کند که برای همزمان‌سازی سیگنال کنترل در مکانیزم تغییر خط استفاده می‌شود. براساس تعداد پالس‌های تولیدی بین ایستگاه مرتب سازی و محل تغییر خط، نرم‌افزار یک سیگنال کنترلی در ایستگاه تغییر خط ایجاد می‌کند تا جعبه درست در محل قرار گیرد.

برنامه‌ریز پروژه وضعیت دامنه را بررسی نموده و تمام عملیات نرم‌افزاری مهم را استخراج می‌کند. این فرآیند به نام تفکیک و تجزیه[3] است که در فصل ۳ مورد بحث قرار گرفت و منجربه توابع زیر شد:[4]

- خواندن داده‌های کد ستونی.
- خواندن پالس تاکومتر.

---

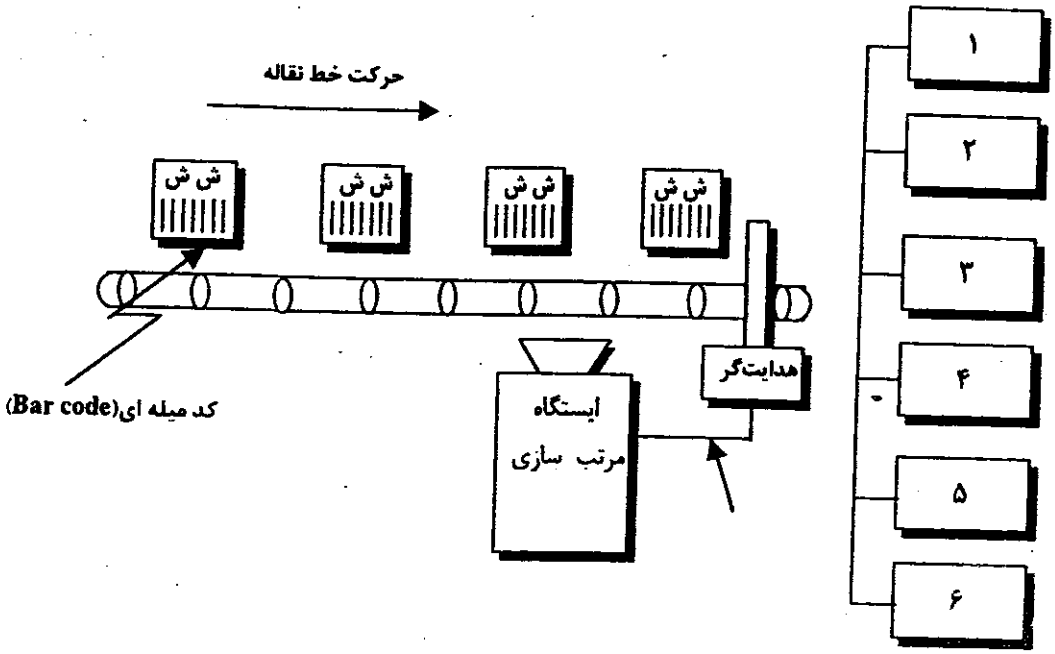
1.Resources

2.Conveyor Line Sorting System

3.decomposition

۴. در واقع، تفکیک و تجزیه کارکردی طی مهندسی سیستم انجام می گیرد(فصل ۱۰). طرح ریز اطلاعات تحصیل شده از مشخصه های سیستم را مورد استفاده قرار می دهد تا کارکردهای نرم افزار را تعریف کند.

- بازکردن بخش کد اطلاعات.
- انجام کار جستجوی بانک اطلاعاتی.
- تعیین محل محفظه.
- تولید سیگنال کنترلی برای تغییر خط.
- ضبط رکوردی از مقصد های جعبه‌ها




ش ش : شماره شناسایی

**شکل ۵-۱ یک سیستم مرتب سازی خط نقاله**

در این مورد، عملکرد توسط سرعت خط حمل و نقل تعیین می‌گردد. فرآیند مربوط به هر جعبه باید قبل از اینکه جعبه دیگر به قسمت خواننده بارکد برسد، تکمیل گردد. نرم‌افزار CLSS بوسیله سخت افزاری که باید به آن دسترسی پیدا کند محدود می گردد، همچنین بوسیله حافظه موجود و پیکربندی کلی خط حمل و نقل.

کارآیی، عملکرد و محدودیتها باید با هم ارزیابی شوند. کارکرد یکسان می‌تواند ترتیب اختلاف میزان بزرگی را در کار تولید تسریع کند وقتی که در متن ارتباطات مختلف کاری در نظر گرفته می‌شود. کار و هزینه لازم برای تولید نرم‌افزار CLSS درصورتی که کارکرد یکسان اما عملکرد متفاوت باشد، می تواند کاملاً تفاوت داشته باشد. مثلاً اگر میانگین سرعت خط حمل و نقل بوسیله فاکتور ۱۰ (عملکرد) افزایش یابد و جعبه‌ها دیگر بصورت فرد قرار نگیرند (یک محدودیت) نرم‌افزار بسیار پیچیده تر شده و کار بیشتری نیز برای آن لازم است. در ابتدا کارکرد، عملکرد و محدودیت با هم مرتبطند.

**توصیه**

امکان سنجی فنی و تکنیکی مهم است، اما نیازمندیهای تجاری حتی از آن هم مهمتر است. باید توجه داشت که ساختن سیستم با فن آوری بالا، یا محصولی خوب و مناسب که با استقبال واقعی مواجه نشود، امر مطلوبی نخواهد بود.

نرم‌افزار با دیگر عناصر سیستم مبتنی بر کامپیوتر ارتباط متقابل دارد. طراح ماهیت و پیچیدگی هر رابط را درنظر می‌گیرد تا هرگونه تأثیری را بر منابع تولید، هزینه و جدول زمانبندی تعیین کند. مفهوم رابط بصورت هر یک از این موارد تعریف شده است:

یک ملاحظه حوزه نرم افزاری باید مشتمل بر ارزیابی و سنجش تمام رابط های خارجی باشد.

۱- سخت افزار (مثل پردازشگر، وسائل جانبی) که نرم‌افزار و وسایلی مثل ماشین‌ها و صفحات نمایش را که بطور غیر مستقیم توسط نرم‌افزار کنترل می شوند را اجرا می‌کند.

۲- نرم‌افزاری که هم اکنون وجود دارد (مثل مسیرهای دسترسی به پایگاههای اطلاعاتی) و باید با نرم‌افزار جدید مرتبط شود.

۳- افرادی که از طریق صفحه کلید یا دیگر وسایل ورودی/خروجی از نرم‌افزار استفاده می‌کنند.

۴- رویه‌هایی که بعنوان سری پیاپی عملیات بعد یا قبل از نرم‌افزار می‌آیند. در هر مورد اطلاعات انتقال یافته در طول رابط باید بدرستی درک شوند.

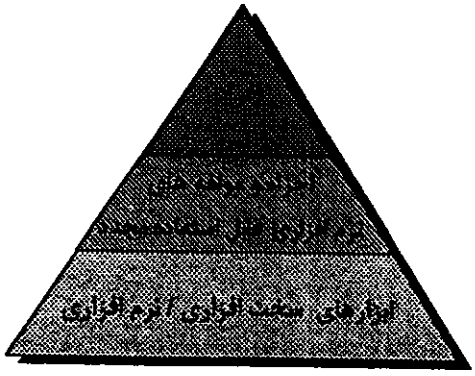
اصلی ترین منبع اطلاعاتی برای تعیین حوزه، کدام است؟

اگر مشخصات سیستم[1] به درستی ارائه شده باشند، تقریباً همه اطلاعات لازم برای توصیف دامنه نرم‌افزار در دسترس بوده و قبل از اینکه برنامه‌ریزی پروژه نرم‌افزاری آغاز شود، ثبت می‌گردند. درمواردی که در آن یک مشخصه ارائه نشده باشد، برنامه‌ریز باید نقش تحلیلگر سیستم را بعهده بگیرد تا شیوه‌ها و تعهداتی را که بر کارها تأثیر می‌گذارند، معین کند.

## ۴-۵ منابع

دومین کار در برنامه‌ریزی نرم‌افزاری تخمین منابع لازم برای نیل به کار تولید نرم‌افزار است. شکل ۵-۲ منابع تولید را بصورت یک هرم نشان می‌دهد. محیط تولید[2] ابزارهای (سخت افزار و نرم‌افزاری) در پایین‌ترین سطح هرم قرار دارند و زیربنای پشتیبانی کننده کار تولید را مهیا می‌کند. در بالاترین سطح نیز با اجزای نرم‌افزاری[3] قابل استفاده مجدد برخورد می‌کنیم، بلوکهای ساختاری نرم‌افزار که می توانند بطور قابل توجهی هزینه‌های تولید را کاهش داده و تحویل را سرعت بخشند. در نوک هرم منبع اولیه است یعنی مردم[4] هر منبع دارای چهار مشخصه است: توصیف منبع، وضعیت دسترسی به آن، زمان تقویمی که در آن موقع منبع مورد نیاز است، مدت زمانی که منبع بکار گرفته می‌شود. دو مشخصه آخری را می‌توان بعنوان یک پنجره درنظر گرفت. در دسترس بودن منبع برای یک پنجره بخصوص باید در اولین زمان ممکن صورت گیرد.

1.System Specification
2.development environment
3.software Component
4.people

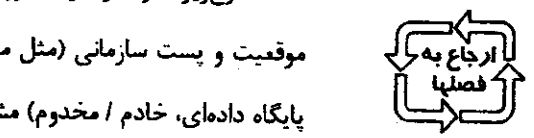



**شکل ۵-۲ منابع پروژه**

### ۵-۴-۱ منابع انسانی

طرح‌ریز کار خود را با ارزیابی دامنه و انتخاب مهارتهای لازم برای تکمیل تولید، شروع می‌کند. موقعیت و پست سازمانی (مثل مدیر، مهندس ارشد نرم‌افزار و غیره) و تخصص (مثل ارتباطات از راه دور، پایگاه داده‌ای، خادم / مخدوم) مشخص می شوند. درمورد پروژه‌های نسبتاً کوچک (یک نفر- سال یا کمتر) ممکن است هر فرد کلیه مراحل مهندسی را با مشورت با متخصصین لازم انجام دهد.

تعداد افراد لازم برای پروژه را می‌توان تنها بعد از تخمین کار تولید ارائه کرد. فنون مربوط به کار تخمین بعداً در این فصل مورد بحث قرار می‌گیرند.




نقشی که افراد نرم افزاری ایفاء می کنند و نیز انواع سازمانهای تیمی در فصل ۳ تشریح شده است.

### ۵-۴-۲ منابع نرم افزاری با قابلیت استفاده مجدد

مهندسی نرم‌افزار مبتنی بر اجزاء[1] (CBSE)[2] بر قابلیت کاربرد مجدد تأکید دارد یعنی ایجاد و استفاده مجدد بلوکهای ساختمان نرم‌افزار [H0091][3]. چنین بلوکهایی به نام جزء یا اجزاء[4] باید بخاطر سهولت در مرجع بندی بصورت فهرست در آمده، برای سهولت کاربرد استاندارد شده و از نظر سهولت ادغام ارزیابی گردند.

بناتان [BEN92][5] چهار گروه منبع نرم‌افزاری را بیان می دارد که باید در برنامه‌ریزیمد نظر قرار گیرند :

برای آنکه به گونه ای موثر بتوان استفاده مجدد از اجزاء نرم افزاری داشت، آنها را باید مستند نمود، استاندارد کرده و از اعتبار آنها اطمینان حاصل کنید.

---

۱. مهندسی نرم افزار مبتنی بر اجزاء به تفصیل در فصل ۲۷ توضیح داده شده است.

2.Component – Based Software Engineering

3.Hooper, J.

4.components

5.Bennatan, E.M.

**اجزای ساخته شده حاضر و آماده .**

نرم‌افزار موجودی که بتوان آن را از شخص ثالث گرفت یا برای یک پروژه قبلی ساخته شده است. این اجزا با نام COTS[1] بلوکهای ساختاری هستند که برای استفاده در پروژه کنونی آماده بوده و بطور کامل ارزیابی شده‌اند.

**اجزایی که دارای تجربه کامل درمورد آنها هستیم.**

مشخصات، طراحی‌ها، کد یا اطلاعات آزمونی موجود برای پروژه های گذشته که مشابه با نرم‌افزاری هستند که برای پروژه جدید ساخته می‌شود. اعضای تیم کنونی تجربه کاملی از حوزه برنامه‌ای که توسط این اجزاء ارائه می‌شوند دارند. بنابراین، اصلاحات لازم برای این اجزاء نسبتاً کم خطر است.

**اجزایی که دارای تجربه نسبی درمورد آنها هستیم.**

مشخصات، طراحی‌ها، کد یا اطلاعات آزمونی موجود که درمورد طرحهای گذشته ارائه شده‌اند و مرتبط با نرم‌افزاری هستند که در پروژه کنونی می خواهیم ارائه کنیم اما به اصلاحات زیادی نیازمندند. اعضای تیم کنونی تجربه محدودی از حوزه کاربردی دارند که توسط این اجزاء ارائه شده‌اند. بنابراین اصلاحات لازم برای اینگونه اجزاء اندکی خطر به همراه دارند.

**اجزاء جدید.**

اجزاء نرم‌افزاری که باید توسط تیم نرم‌افزار ساخته شوند بویژه برای نیازهای پروژه کنونی.

رهنمودهای زیر باید مدنظر برنامه‌ریز باشند بویژه در هنگامی که اجزای قابل استفاده مجدد را بعنوان یک منبع مشخص می‌کنند:

۱- اگر اجزای آماده مصرف نیازهای پروژه را برآورد می‌کنند، آنها را بخرند. هزینه این خرید و الحاق آنها تقریباً همیشه کمتر از هزینه تولید نرم‌افزار معادل است.[2] علاوه براین، خطر آن نسبتاً کم است.

۲- اگر اجزایی که درمورد آنها تجربه کامل داریم، در دسترس هستند، خطر مربوط به اصلاح و الحاق آنها معمولاً پذیرفتنی است. طرح پروژه باید منعکس کننده استفاده از این اجزاء باشد.

۳- اگر تجربه نسبی از این جزء داریم استفاده آن برای پروژه کنونی باید تحلیل شود. اگر اصلاحات گسترده موردنیاز است کار را با دقت انجام دهید زیرا که الحاق عناصر به اجزای دیگر دارای خطر زیادی می‌باشد. هزینه اصلاح اجزایی که تجربه چندانی از آنها نداریم، گاهی می‌تواند بسیار بیشتر از هزینه تولید اجزای جدید باشد.

پرسش

هنگامی که استفاده مجدد از اجزاء نرم افزار موجود را طرح ریزی می کنیم، چه ملاحظاتی را باید مد نظر داشته باشیم؟

---

1. Commercial Off – The – Shelf

۲. هنگامیکه اجزاء موجود نرم افزاری، دریک پروژه مورد استفاده واقع می شوند. هزینه کلی کاهش چشم گیری خواهد داشت. در واقع داده های صنعتی مشخص می کنند که هزینه، زمان بازاریابی و تعداد نواقص همه کاهش می یابند.

اغلب اجزاء نرم‌افزاری قابل استفاده مجدد در طول برنامه‌ریزی نادیده گرفته می شوند تا در طول مرحله تولید فرآیند نرم‌افزار به یک دغدغه بزرگ تبدیل شوند. بهتر است پیشتر منابع موردنیاز را مشخص کنیم. به این ترتیب، ارزیابی تکنیکی گزینه‌ها را می‌توان انجام داد و خرید به موقع صورت می‌گیرد.

### ۵-۴-۳ منابع محیطی

محیطی که پروژه نرم‌افزاری را پشتیبانی می‌کند و اغلب به نام محیط مهندسی نرم‌افزار (SEE)[1] است، بکار گیرنده سخت افزار و نرم‌افزار است. سخت افزار سکویی است که ابزارهای لازم (نرم‌افزاری) برای تولید محصولات کاری را تأمین می‌کند که نتیجه یک کار مهندسی نرم‌افزاری خوب هستند.[2] از آنجا که اکثر سازمانهای نرم‌افزاری دارای سازه‌های چند گانه‌ای هستند که مستلزم دسترسی به SEE می‌باشند، برنامه‌ریز پروژه باید پنجره زمانی لازم برای سخت افزار و نرم‌افزار را توصیه کرده و تأیید کند که این منابع موجودند.

وقتی قرار است سیستمی مبتنی بر کامپروتر (بکارگیرنده نرم‌افزار و سخت افزارهای تخصصی) طراحی شود، ممکن است تیم نرم‌افزاری نیازمند دسترسی به عناصر سخت افزاری باشند که برای دیگر تیمها طراحی شده‌اند. مثلاً نرم‌افزاری برای کنترل عددی (NC) که در گروهی از ماشین ابزارها بکار رفته ممکن است به ماشین ابزار خاصی (مثلاً ماشین تراش NC) بعنوان بخشی از مرحله آزمون اعتبار نیاز داشته باشد. شاید پروژه نرم‌افزاری درمورد صفحه بندی بصورت پیشرفته به یک سیستم تایپ ستینگ دیجیتالی درطول تولید نیاز داشته باشد. هر عنصر سخت افزاری باید توسط یک برنامه ریز پروژه نرم‌افزاری مشخص شود.

## ۵-۵ برآورد پروژه های نرم افزاری

در اوایل کار محاسبه و برآورد، هزینه‌های نرم‌افزاری درصد کوچکی از کل هزینه مبتنی بر کامپیوتر را تشکیل می‌دهند. ترتیب خطای میزان قابل سنجش در برآورهای هزینه نرم‌افزار تأثیر نسبتاً کمی دارد. امروزه، نرم‌افزار عنصر گرانقیمت مجازی تمام سیستمهای کامپیوتری است. درمورد سیستم‌های رایج پیچیده، خطای عمده در تخمین هزینه می‌تواند بین سود و ضرر تفاوت ایجاد نماید. بالا رفتن زیاد هزینه می‌تواند برای تولیدکنند نابودکننده باشد.

برآورد هزینه و کار درمورد نرم‌افزار هرگز یک کار عملی دقیق نیست. متغیرهای بسیاری از جمله انسان، متغیرهای فنی، محیطی و سیاسی می‌تواند بر هزینه نمایی کار ساخت و تولید و بکارگیری آن تأثیر

**نقل قول**

اگر مدت استفاده از منابع خارجی به طول انجامیده، رقابت افزایش یابد. توانایی برآورد دقیق و صحیح، برای بسیاری از گروههای فن آوری اطلاعات به صورت فاکتوری بحرانی و حیاتی ظهور می یابد.
راب تامست

1. software engineering

۲. دیگر سخت افزارها- محیط هدف - کامپیوتری است که نرم افزاربر روی آن اجرا می گردد و به کاربر تحویل داده می شود.

بگذارند. برآورد پروژه می‌تواند از یک سحر و جادو به یک سری مراحل نظام‌مند تبدیل شود که تخمین‌هایی با حداقل مخاطره پذیرفته شده را مهیا می‌کنند.

برای رسیدن به تخمین های مورد اطمینانی از نظر هزینه و کار، یک سری گزینه وجود دارد:

۱- کار تخمین را تا اواخر پروژه به تعویق بیاندازیم (مشخصاً می توانیم بعد از تکمیل پروژه به تخمین‌های ۱۰۰٪ درستی برسیم!).

۲- برآوردها را بر پایه پروژه‌هایی قرار دهیم که هم اکنون تکمیل شده‌اند.

۳- از فنون نسبتاً ساده تفکیک برای ارائه این برآوردها استفاده کنیم.

۴- از یک یا چند مدل تجربی برای آن استفاده نماییم.

متأسفانه، اولین گزینه با اینکه جذاب است اما عملی نیست. تخمین هزینه باید با صراحت مهیا شود. باید بدانیم که هرچه بیشتر صبر کنیم بیشتر مطلع می‌شویم و هرچه بیشتر مطلع شویم احتمال اینکه خطاهای جدی در تخمین ها رخ دهد کمتر می‌شود.

دومین گزینه می‌تواند بطور منطقی کارگر باشد اگر که پروژه کنونی کاملاً مشابه شرایط تأثیرگذارنده با پروژه قبلی باشد (مثلاً مشتری، شرایط تجاری، SEE، مهلت مقرر). متأسفانه تجربه گذشته همیشه نشانه خوبی برای نتایج آینده نبوده است.

باقی گزینه‌ها روشهای متغیری برای تخمین پروژه نرم‌افزاری هستند، مطلوب این است که فنون مورد اشاره در هر زمینه پشت سر هم بکار گرفته شوند و هرکدام بعنوان یک کنترل برای دیگری باشد. فنون جداسازی از رهیافت تقسیم و غلبه[1] برای برآورد پروژه استفاده می‌کنند. با تفکیک پروژه بصورت کارکردها و فعالیتهای مهندسی مربوطه، تخمین هزینه و کار بصورت مرحله‌ای انجام می‌گیرد. *مدلهای تجربی برآورد*[2] را می‌توان برای فنون جداسازی تکمیلی استفاده کرده و روش تخمین ارزشمندی را در جای درست خود ارائه داد.

مدلی که براساس تجربه بوده (اطلاعات تاریخی) و فرم زیر را می‌گیرد:

$$d = f(v_i)$$

که در آن $d$ یکی از مقادیر تخمینی است (مثل کار، هزینه، مدت پروژه) و $v_i$ پارامترهای مستقل انتخابی می‌باشند (مثل LOC یا FP تخمینی).

*ابزارهای خودکار تخمین و برآورد*[3]، یک یا چند تکنیک تجزیه یا مدلهای تجربی را به اجرا درمی‌آورند. وقتی این ابزارها با رابط گرافیکی کاربر ترکیب می شوند، گزینه جالبی برای انجام تخمین ارائه می‌دهند. در

ابزار برآورد

1.divide & Conquer

2.Empirical estimation models

3.Automated estimation tools

می‌دهند. در چنین سیستم‌هایی، مشخصه‌های سازمان تولیدی (مثل تجربه و محیط) و نرم‌افزاری که قرار است تولید شود، توصیف می‌گردند. برآورد هزینه و نیروی کار از این اطلاعات نشأت می‌گیرد.

هر یک از گزینه های متغیر تخمین هزینه تنها تا وقتی خوب است که اطلاعات تاریخی برای انجام تخمین استفاده می‌شود. اگر این اطلاعات وجود نداشته باشد، برآورد هزینه بر پایه‌های ارزانی استوار می‌شود. در فصل ۴ مشخصه‌های بعضی از متریکهای نرم‌افزاری را بررسی کردیم که اساس اطلاعات تخمین تاریخی را مهیا می‌کنند.

### ۵-۶فنون تجزیه

برآورد پروژه نرم‌افزاری شکلی از حل مسئله بوده و در اکثر مواقع مسئله‌ای که باید حل شود (یعنی ارائه برآورد هزینه و نیروی کار در مورد پروژه) بسیار پیچیده تر از آن است که در یک جا بررسی شود. به همین دلیل، مسئله را تجزیه کرده آن را بعنوان مجموعه ای از مسائل کوچکتر مجدداً توصیف می‌کنیم (که امیدواریم قابل ساماندهی باشند).

در فصل ۳، روش تفکیک را از دو دیدگاه مورد بحث قرار دادیم: تجزیه مسئله و تجزیه فرآیند. کار برآورد از یکی یا هر دوی این روشها استفاده می‌کند. اما قبل از انجام برآورد، برنامه‌ریز پروژه باید دامنه نرم‌افزاری را که قرار است ساخته شود شناخته و تخمینی از اندازه آن ارائه کند.

#### ۵-۶-۱ اندازه زدن نرم افزار

اندازه یک نرم افزار در دست ساخت می تواند با استفاده از معیار های مستقیم ، تعداد خطوط برنامه، یا معیارهای غیر مستقیم، امتیاز کارکردی برآورد شود.

میزان دقت تخمین پروژه براساس چند چیز پیش‌بینی می‌شود: ۱) درجه‌ای که نسبت به آن برنامه‌ریز به درستی اندازه محصول کاری را تخمین زده است. ۲) توانایی ترجمه تخمین اندازه به صورت نیروی کار انسانی، تقویم زمانی و میزان پول (کارکردی از قابلیت دسترسی متریک های قابلیت اطمینان نرم‌افزاری از پروژه‌های گذشته. ۳) درجه و میزانی که نسبت به آن طرح پروژه منعکس کننده توانائیهای تیم نرم‌افزاری است. ۴) ثبات نیازمندیهای محصول و محیطی که کار مهندسی نرم‌افزار را پشتیبانی می‌کند.

در این بخش مسئله اندازه بندی نرم‌افزار را بررسی می کنیم. از آنجا که برآورد پروژه تنها وقتی خوب است که تخمین اندازه آن نیز صورت گیرد، اندازه بندی نمایانگر اولین چالش عمده برنامه‌ریز خواهد بود. در متن کار برنامه‌ریزی پروژه، اندازه اشاره دارد به نتیجه قابل سنجش پروژه نرم افزاری، اگر از روش مستقیمی استفاده شود، اندازه را می‌توان در LOC ارزیابی کرد. اگر رهیافت غیرمستقیمی استفاده شود، اندازه بصورت FP نمایش داده می‌شود.

پوتنام و مایرز [PUT92] [1] چهار روش مختلف برای تعیین میزان مسئله بیان می‌کنند:

1.Putnam, L.

چگونه می توان اندازه نرم افزاری که ساخت آن طرح ریزی شده است، به دست آورد؟

***شیوه اندازه زدن بامنطق فازی.***

این روش از فنون استدلالی تقریبی است که اساس و شالوده منطق فازی هستند. برای بکارگیری این روش، برنامه ریز باید نوع برنامه کاربردی را شناسایی نموده، بزرگی آن را روی مقیاس کیفی مشخص کرده و سپس این شدت را در دامنه اصلی اصلاح کند. گرچه می‌توان از تجربه مشخصی استفاده نمود، اما برنامه‌ریز باید به یک پایگاه اطلاعاتی تاریخی از پروژه‌ها[1] نیز دسترسی داشته باشد بطوریکه تخمین ها را بتوان با تجربه حقیقی مقایسه نمود.

***شیوه اندازه زدن با امتیازات عملکردی.***

برنامه ریز برآوردهایی از مشخصه‌های دامنه اطلاعات که در فصل ۴ بحث شد، ارائه می‌دهد.

***شیوه اندازه زدن با جزء استاندارد.***

هر نرم‌افزار متشکل از چند جزء استاندارد مختلف است که برای یک ناحیه کابردی مشخص بطور عمومی می‌باشند. مثلاً اجزای استاندارد برای یک سیستم اطلاعاتی عبارتند از: سیستم های فرعی، پیمانه‌ها، صفحات، گزارشات، برنامه‌های محاوره‌ای، برنامه‌های دسته‌ای، فایلها، LOC و دستورالعمل ها در سطح شیء. برنامه‌ریز پروژه تعداد موارد وقوع هر یک از اجزاء استاندارد را تخمین

زده و سپس از اطلاعات تاریخی پروژه برای تعیین اندازه ارائه شده در هر جزء استاندارد استفاده می‌کند. برای تشریح مطلب، یک برنامه کاربردی سیستم‌های اطلاعاتی را در نظر بگیرید. طرح‌ریز بطور تخمینی ۱۸ گزارش را برآورد می‌کند. اطلاعات تاریخی نشانگر این هستند که ۹۶۷ خط از زبان کوبول [PUT92]در هر گزارش لازمند. این کار طراحی را قادر به برآورد این می‌کند که ۱۷۰۰۰ LOC برای جزء گزارش لازمند. تخمین‌های مشابه و محاسباتی به همین ترتیب درمورد دیگر اجزای استاندارد ارائه شده و مقدار اندازه مرکب (که بصورت آماری تنظیم شده) بدست می‌آید.

***شیوه ت اندازه زدن با تغییر اندازه.***

این روش برای وقتی مورد استفاده قرار می‌گیرد که پروژه‌ای دربرگیرنده استفادهٔ از نرم‌افزار کنونی باشد که به شکلی بعنوان بخشی از پروژه تغییر می‌یابد. برنامه‌ریز تعداد و نوع اصلاحاتی را که باید صورت بگیرند، برآورد می‌کند. با استفاده از «نسبت نیروی کار» [PUT92] درمورد هر نوع تغییر، اندازه آن تغییر برآورد می‌شود.

پوتنام و می‌یر بیان می‌دارند که نتایج هر روش اندازه گیری که در بالا مورد اشاره قرار گرفت را می‌توان بطور آماری ترکیب نمود تا تخمین سه امتیاز یا ارزش مورد انتظار[2] ایجاد شود. این کار بوسیله ارائه

---

۱. برای تشریح ابزارهای برآوردی که از پایگاه داده های تاریخی بهره می برند و دیگر فنون اندازه گیری بخش ۵-۹ را مورد مطالعه قرار دهید.

2.three-point or expected value

مقادیر خوشبینانه (کم) دارای احتمال بیشتر و بدبینانه (زیاد) درمورد اندازه و ترکیب آنها با استفاده از معادله (۵-۱) که در بخش بعدی توصیف می‌شود صورت می‌گیرد.

### ۵-۶-۲ برآورد مبتنی بر مسئله

در فصل ۴، خطوط کد (LOC) و امتیازات کارکردی (FP) به عنوان مقیاسهایی توصیف شدند که از روی آنها متریک بهره‌وری را می‌توان محاسبه نمود. LOC و FP در طول کار برآورد پروژه به دو صورت مورد استفاده قرار می‌گیرند:

۱- بعنوان متغیر تخمینی که برای تعیین اندازه هر عنصر نرم‌افزاری استفاده می‌شود.

۲- بعنوان متریک خط پایه که از روی پروژه‌های قبلی جمع آوری شده و در ارتباط با متغیرهای تخمینی برای ارائه هزینه و میزان نیروی کاری پروژه استفاده می‌شود.

برآورد LOC و FP فنون متمایزی از تخمین هستند. هر دو دارای چند مشخصه مشترک می‌باشند. برنامه‌ریز پروژه با وضعیت محدود شده دامنه نرم‌افزار شروع کرده و از روی آن تلاش می‌کند نرم‌افزار را بصورت توابعی از مسئله در بیاورد که بتوان هر کدام را به صورت مجزا برآورد کرد. LOC و FP برای هر تابع تخمین زده می‌شود. متناوباً، برنامه‌ریز جزء دیگری را برای تعیین اندازه انتخاب می‌کند مثل کلاس ها یا اشیاء، تغییرات یا فرآیندهای تجاری تحت تأثیر قرار گرفته.

سپس متریکهای بهره‌وری خط مبداء (مانند LOC/pm یا FP/pm ) با متغیر برآورد مناسب بکار گرفته شده و هزینه یا نیروی کار تابع بدست می‌اید. تخمین توابع برای ایجاد یک تخمین کمّی در مورد کل پروژه، ترکیب می‌شود.

نکته مهم مورد توجه این است که اغلب نکات پراکنده مستولی در متریک‌های بهروری یک سازمان وجود دارد که از حدس میزان بهره‌وری خط مبداء استفاده می‌کنند. بطور کلی، با میانگین های LOC/pm یا FP/pm[1] باید دامنه پروژه را محاسبه کرد. یعنی پروژه‌ها باید بوسیله اندازه تیم، حوزه کاربرد، پیچیدگی و دیگر پارامترهای مربوطه دسته‌بندی شوند. میانگین های دامنه محلی باید محاسبه گردند. وقتی پروژه جدید برآورد شد، ابتدا باید به یک دامنه تخصیص یابد و سپس میانگین مناسب دامنه از نظر بهره‌وری در ایجاد برآورد استفاده شود.

فنون تخمین LOC و FP از نظر جزئیات لازم برای تفکیک و هدف تقسیم بندی با هم متفاوتند. وقتی LOC بعنوان متغیر تخمین استفاده می شود، تجزیه[2] کاملاً ضروری است و اغلب از نظر جزئیات بررسی می‌شود. روش تفکیک زیر از فیلیپس [PHI98][3] برگرفته شده است:

پرسش

برآوردهای مبتنی بر تعداد خطوط برنامه (LOC) و امتیازات کارکردی (FP) در چه چیز مشترک می‌باشند

توصیه

هنگامی که متریک های بهره وری را برای پروژه گرد می آوریم، مطمئن باشید که از نوع طبقه بندی پروژه، آگاهید. این امر شما را قادر می سازد که از متوسط ویژگیهای حوزه آگاه بوده، برآورد دقیقتری ارائه نمایید.

تعدد روش

برای برآورد تعداد خطوط برنامه، تجزیه، متمرکز بر کارکردها خواهد بود.

---

۱. pm مخفف نفر- ماه است.

۲. بطور کلی، کارکردهای مسئله باید تفکیک شوند. با این وجود یک لیست از اجزاء استاندارد (بخش ۵-۶-۱) بجای آن می تواند به کار رود.

3.Philips, D.

```
define product scope;
identify functions by decomposing scope;
do while functions remain
select a function j
assign all functions to sub functions list;
do while subfunctions remain
select subfunction k
If subfunction k resembles subfunction d described In a historical data base
then     note historical cost. effort. size (LOC or FP) data for subfunction d;
adjust historical cost. effort, size data based on any differences;
use adjusted cost. effort. size data to derive partial estimate. Ep;
project estimate = sum of' {Ep};
    else          if cost. effort. size (LOC or FP) for subfunction k can be estimated
then derive partial estimate. Ep;
    project estimate = sum of' {Ep};
    else subdivide subfunction k into smaller subfunctions;
    add these to subfunctions list;
    endif

    endif
enddo
    enddo
```

روش تفکیک مورد اشاره فوق این طور فرض می‌کند که می‌توان همه توابع را بصورت زیر توابع تفکیک کرد که شبیه مدخل های یک پایگاه داده‌ای تاریخی می‌شود. اگر این مورد نباشد، از روش اندازه‌گیری دیگری استفاده می‌شود. هرچه میزان تقسیم‌بندی بیشتر باشد، این احتمال بیشتر است که تخمین‌های منطقی دقیق‌تری از LOC ارائه شود.

برای برآورد امتیاز کارکردی تجزیه بر ویژگیهای حوزه اطلاعاتی متمرکز خواهد شد.

درمورد تخمین‌های FP، کار جداسازی بصورت دیگری است. علاوه بر تمرکز روی تابع، هر یک از مشخصه‌های دامنه اطلاعات یعنی ورودی‌ها، خروجی‌ها، فایلهای اطلاعاتی، درخواستها و رابطهای خارجی بعلاوه چهارده مقدار تطابق پیچیدگی که در فصل ۴ مورد بحث قرار گرفتند، برآورد می شوند. تخمین‌های بدست آمده را می‌توان برای بدست آوردن یک مقدار FP که می‌تواند با اطلاعات قبلی مرتبط شده، استفاده نمود و برای ارائه یک مقدار تخمینی استفاده کرد.

چگونه می توانیم ارزش مورد انتظار را برای اندازه نرم افزار محاسبه نماییم ؟

بدون توجه به متغیر تخمین مورد استفاده، برنامه‌ریز پروژه با برآورد یک سری مقادیر برای هر یک از تابع ها یا مقدار دامنه اطلاعاتی شروع می‌کند. با استفاده از اطلاعات تاریخی یا (وقتی بقیه موفق نباشند) حدس، برنامه‌ریز اندازه بزرگی هر تابع یا رقمی برای هر مقداری از دامنه اطلاعات را بصورت خوشبینانه یا دارای احتمال زیاد یا از روی بدبینی تخمین می‌زند. اشارت ضمنی میزان عدم قطعیت هنگامی مهیا می‌شود که دامنه‌ای از اعداد مشخص شوند.

---

۱ زبان رسمی غیر رسمی طراحی فرآیند، که در اینجا بکار رفته رهیافت کلی برای اندازه است. (که البته) تمام پیشامدهای منطقی را در نظر نمی گیرد.

سپس یک مقدار مورد انتظار یا سه امتیازی محاسبه می‌شود. مقدار منتظره برای متغیر تخمینی (اندازه) S ، را می توان بعنوان یک میانگین ارزیابی شده از تخمین‌های خوشبینانه ($S_{opt}$)، دارای احتمال بالا ($S_m$) و تخمین بدبینانه ($S_{pess}$) ارزیابی کرد. بطور مثال:

$$S = (S_{opt} + 4S_m + S_{pess}) / 6 \qquad (۱-۵)$$

بیشترین اعتبار را به تخمینی که احتمال زیادی دارد داده و توزیع احتمال بتا ($\beta$) را دنبال می‌کند. فرض می‌کنیم که احتمال بسیار کمی وجود دارد که نتیجه اندازه حقیقی خارج از محدوده مقادیر خوشبینانه یا بدبینانه قرار گیرد.

وقتی مقدار مورد انتظار درمورد متغیر تخمین معین شد، LOC یا اطلاعات بهره‌وری FP بکار گرفته می شوند. آیا این تخمین‌ها درست هستند؟ تنها پاسخ منطقی به این سئوال این است: «نمی‌توانیم مطمئن باشیم» هر گونه تکنیک دیگری بدون توجه به این که چقدر پیچیده باشد باید با روش دیگری بررسی شود. حتی بعد از آن باید با استفاده از عقل سلیم و تجربه آن را پیگیری کرد.

### ۵-۶-۳ مثالی از برآورد مبتنی بر تعداد خطوط برنامه (LOC)

بعنوان نمونه‌ای از فنون تخمین LOC و FP، بسته نرم‌افزاری را درنظر بگیرید که قرار است برای یک برنامه کاربردی طراحی با کمک کامپیوتر(CAD) برای اجزای مکانیکی تولید شود. بازنگری مشخصات سیستم[1] نشانگر این است که نرم‌افزار قرار است روی یک ایستگاه کاری مهندسی اجرا شود و باید با محیطهای گرافیکی مختلف کامپیوتری از جمله موس، دیجیتایزر، صفحه نمایش دارای تفکیک پذیری بالا و چاپگر لیزری ارتباط برقرار کند.

1 System Specifications

| فانکشن | تعداد خطوط بر اورد شده |
|---|---|
| رابط کاربر و تسهیلات کنترلی (UICF) | ۲,۳۰۰ |
| تحلیل ژئو متری دو بعدی (2DGA) | ۵,۳۰۰ |
| تحلیل ژئو متری سه بعدی (3DGA) | ۶,۸۰۰ |
| مدیریت پایگاه داده ها (DBM) | ۳,۳۵۰ |
| تسهیلات نمایش گرافیکی کامپیوتری (CGDF) | ۴,۹۵۰ |
| فانکشن کنترل تجهیزات جانبی (PCF) | ۲,۱۰۰ |
| ماژولهای تحلیل طرا حی (DAM) | ۸,۴۰۰ |
| تعداد خطوط بر اورد شده برنامه | ۳۳,۲۰۰ |

**شکل ۵-۳ جدول برآورد و تخمین برای روش تعداد خطوط بر نامه (LOC)**

با استفاده از مشخصه‌های سیستم بعنوان یک راهنما، وضعیت اولیه دامنه نرم‌افزار را می‌توان توسعه داد:

*نرم‌افزار CAD اطلاعات هندسی دو و سه بعدی را از مهندس دریافت می‌کند. مهندس با سیستم ارتباط برقرار کرده و CAD را از طریق رابط کاربری کنترل می‌کند که مشخصات یک طراحی رابط متقابل ماشین – انسان را بخوبی نشان می‌دهد. هم اطلاعات هندسی و دیگر اطلاعات پشتیبان در پایگاه داده‌ای CAD حفظ می شوند. پیمانه های تحلیل طراحی برای تولید خروجی لازم ارائه خواهند شد که روی یک سری وسایل گرافیکی نمودار می‌شوند. نرم‌افزار برای کنترل و ارتباط متقابل با وسایل محیطی ازجمله موس، دیجتایزر، چاپگر لیزری و پلاتر است.*

وضعیت دامنه فوق، اقدام اولیه است یعنی محدود نشده است. هر جمله باید بسط یابد تا جزئیات خاص و ارتباط کمی را مهیا بسازد. بطور مثال، قبل از تخمین زدن باید برنامه‌ریز تعیین کند

که «مشخصه‌هایی طراحی خوب رابط متقابل انسان – ماشین» یعنی چه یا اینکه اندازه و میزان پیچیدگی پایگاه داده‌ای CAD باید چه باشد.

به منظور دستیابی به اهداف مدنظر، ما فرض می‌کنیم که پالایش بیشتری رخ داده و عملکردهای نرم‌افزاری عمده زیر شناسایی شده‌اند:

- تسهیلات کنترلی و رابط کاربر (UICF)
- تحلیل هندسی دو بعدی (2DGA)

**توصیه**

بسیاری از برنامه های کاربردی پیشرفته مبتنی بر شبکه بوده، یا بخشی از یک معماری خادم / مخدوم می باشند. بنابرین هنگام برآورد، نیازمندیهای "زیرساختی" را نیز مد نظر داشته باشید.

- تحلیل هندسی سه بعدی (3DGA).
- مدیریت پایگاه اطلاعاتی (DBM).
- تسهیلات نمایش گرافیکی کامپیوتر (CGDF).
- کارکرد کنترل دستگاه های جانبی (PCF).
- پیمانه های تحلیل طراحی (DAM).

**توصیه**

هرگز بر نتایج حاصل بر برآورد خود، پای نفشارید و آنها را وحی منزل نپندارید. شما باید شیوه های دیگری را نیز برای برآوردهای دیگر و تخمین های بیشتر، بیازمائید.

به دنبال تکنیک برای LOC، جدول تخمین زننده که در شکل۵-۳ آمده، ارائه شده است. دامنه‌ای از تخمین‌های LOC برای هر تابع داده شده است. بطور مثال، دامنه تخمین‌های LOC برای تابع تحلیل هندسی سه بعدی عبارتست از : خوشبینانه ۴۶۰۰ LOC، با احتمال بالا، ۶۹۰۰LOC و بدبینانه ۸۶۰۰ LOC.

با بکارگیری معادله (۵-۱) مقدار منتظره برای تابع تحلیل هندسی سه بعدی می‌شود: ۶۸۰۰ LOC. سایر تخمین‌ها به شیوه‌ای مشابه مشتق می‌شوند. با جمع‌بندی عمودی در ستون LOC تخمینی، برآورد ۳۳۱۵۰ خط کد برای سیستم ACD ارائه می‌شود.

بازنگری داده‌های تاریخی نشانگر این است که میانگین بهره‌وری سازمانی برای سیستم های از این قبیل LOC/pm است[1]. براساس میزان نیروی کار هزینه بر ۸۰۰۰ دلار در ماه، هزینه هر خط کد تقریباً ۱۳۰ دلار است. براساس تخمین LOC و داده‌های بهره‌وری تاریخی کل هزینه تخمینی پروژه ۴۳۱۰۰۰ دلار و نیروی کار لازم ۵۴ نفر در ماه است.

### ۵-۶-۴ مثالی از برآورد مبتنی بر امتیاز کارکردی (FP)

**ارجاع به وب**

اطلاعات مربوط به هزینه امتیازات کارکردی و ابزارهای برآورد و تخمین‌در آدرس زیر وجود دارند: www.spr.com

تفکیک سازی درمورد تخمین مبتنی بر FP علاوه بر کارکردهای نرم‌افزاری روی مقادیر دامنه اطلاعات نیز متمرکز می‌شود. با یادآوری جدول محاسبه امتیازات عملکردی که در شکل ۵-۴ آمده، برنامه ریز پروژه، ورودی ها، خروجی ها، درخواستها، فایل‌ها و رابطه‌های خارجی را درمورد نرم‌افزار CAD تخمین می‌زند. درمورد اهداف این تخمین، عامل ارزیابی پیچیدگی بطور متوسط درنظر گرفته می‌شود. شکل ۵-۴، نمایانگر نتایج این تخمین است. هر یک از عوامل ارزیابی سختی کار تخمین زده شده و عامل تنظیم بصورت توصیف شده در فصل ۴، محاسبه می‌شود.

نهایتاً میزان تخمین‌زده شده FP به دست می آید:

$$FP_{estimated} = Count\text{-}total * [0.65+0.01 * \Sigma E(Fi)]$$

$$FP_{estimated} = 375$$

۱ـ برآوردها به هزار دلار و نفر ـ ماه، گرد شده اند. دقت ریاضی کمتر از این واقعی نخواهد بود.

میانگین بهره‌وری سازمانی درمورد سیستم هایی از این نوع می‌شود 6.5 FP/pm براساس دستمزد نیروی کار به میزان ۸۰۰۰ دلار در ماه، هزینه هر FP تقریباً ۱۲۳۰ دلار می‌شود. براساس تخمین LOC و اطلاعات بهره‌وری تاریخی، هزینه تخمین زده پروژه می‌شود ۴۶۱۰۰۰ دلار و نیروی کار تخمینی ۵۸ نفر در ماه است.

توصیه

اگر زمان اجازه دهد، می توان در شکل ۵-۵ بخش بندی های کوچکتری نیز ایجاد نمود و به وظائف کوچکتری رسید. برای مثال تحلیل می تواند به وظائف اصلی آن بشکند. و هرکدام جداگانه مورد تخمین واقع شده، برآورد شود.

| مقادیر حوزه اطلاعاتی | خوشبینانه | متوسط | بدبینانه | شمارش برآورد | وزن | امتیاز عملیاتی |
|---|---|---|---|---|---|---|
| تعداد ورودی ها | ۲۰ | ۲۴ | ۳۰ | ۲۴ | ۴ | ۹۷ |
| تعداد خروجی ها | ۱۲ | ۱۵ | ۲۲ | ۱۶ | ۵ | ۷۸ |
| تعداد پرس و جو ها | ۱۶ | ۲۲ | ۲۸ | ۲۲ | ۵ | ۸۸ |
| تعداد پرونده ها | ۴ | ۴ | ۵ | ۴ | ۱۰ | ۴۲ |
| تعداد رابطهای خارجی | ۲ | ۲ | ۳ | ۲ | ۷ | ۱۵ |
| تعدادکل | | | | | | ۳۲۰ |

**شکل ۵ - ۴ مقادیر برآورد شده در حوزه اطلاعات**

### ۵-۶-۵ برآورد مبتنی بر فرآیند

توصیه

انتظار نداشته باشید که تخمین ها در برآوردهایتان یک تا دو درصد اختلاف داشته باشد. اگر برآورد ها ۲۰ درصد انحراف داشته باشند، ارزشمند و رضایت بخش خواهند بود.

رایج‌ترین تکنیک تخمین زدن یک پروژه عبارتست از مبناء قرار دادن تخمین بر اساس شیوهٔ فرآیند مورد استفاده. یعنی فرآیند به چند مجموعه وظائف نسبتاً کوچک تقسیم می‌شود. و نیروی کاری لازم برای هر وظیفه برآورد می‌شود.

مانند فنون مبتنی بر مسئله تخمین مبتنی بر فرآیند با به تصویر کشیدن کارکردهای نرم‌افزاری بدست آمده از دامنه پروژه، آغاز می‌شود یک سری فعالیت‌های فرآیند نرم‌افزاری باید برای هر کارکرد صورت گیرد.کارکردها و فعالیتهای فرآیند مربوطه ممکن است بعنوان بخشی از جدولی مشابه با جدول ارائه شده در شکل۵-۵ به نمایش در آید.

وقتی کارکردها و فعالیتهای فرآیندی اعلام شدند، برنامه‌ریز نیروی کاری را که (مثلاً نفر- ماه) برای رسیدن به فرآیند هر کارکرد نرم‌افزاری لازم است، برآورد می‌کند. این اطلاعات شبکه اصلی جدول شکل ۵-۵ را تشکیل می‌دهند. میزان متوسط نیروی کار (یعنی هزینه/ واحد نیروی کار) در نیروی کار تخمین زده شده برای هر فرآیند بکار گرفته می‌شود. این احتمال وجود دارد که میزان نیروری کار در مورد هر مورد متفاوت باشد. کارکنان ارشد شدیداً در فعالیتهای اولیه دخیلند و معمولاً از کارکنان سطح پایین‌تر که در کارهای طراحی، ایجاد کد و آزمون اولیه بعدی دخالت دارند، گرانتر می‌باشند.

هزینه ها و نیروی کار برای هر عملیات و فرآیند نرم‌افزاری بعنوان آخرین مرحله، محاسبه می‌شوند. اگر تخمین مبتنی بر فرآیند مستقل از LOC یا FP صورت گیرد، اکنون دو یا سه تخمین برای هزینه و

نیروی کار داریم که ممکن است مقایسه و تلفیق شوند. اگر دو مجموعه تخمین، با هم توافق نشان دهند دلیل خوبی وجود دارد که باور کنیم تخمین‌ها قابل اطمینان خواهند بود. اگر نتایج این فنون تفکیک، سازگاری اندکی با هم داشته باشند تخمین و تحلیل بیشتری باید صورت گیرد.

### ۵-۶-۶ مثالی از برآورد مبتنی بر فرآیند

به منظور تشریح استفاده از تخمین مبتنی بر فرآیند. دوباره نرم‌افزار CAD را درنظر می‌گیریم که در بخش ۵-۶-۳ معرفی شد. محاسبات سیستم و همه توابع نرم‌افزاری بدون تغییر مانده و در دامنه پروژه به آن اشاره می‌شود.

بارجوع به جدول کامل شده در شکل ۵-۵، تخمین نیروی کار (درمورد تعداد افراد در ماه) درمورد هر کار طراحی نرم‌افزاری برای هر کارکرد نرم‌افزاری CAD مهیا شده است. فعالیتهای مهندسی و ساخت/ ارائه بصورت کارهای مهندسی عمده نرم‌افزار نشان داده شده اند. تخمین های ناخالص نیروی کار آمده‌اند. جمع‌های کل افقی و عمومی نشانه نیروی کار تخمین زده لازم را برای تحلیل، طراحی، کدگذاری و آزمون ارائه می‌دهند. باید توجه داشت که ۵۳ درصد کل نیروی کار صرف کارهای مهندسی «ابتدا – انتها» می‌شود. (طراحی و تحلیل لازم) که نشانگر اهمیت نسبی این کار است.

براساس میزان میانگین نیروی کار به منظور ۵۰۰۰ دلار در ماه، هزینه تخمین زده کل پروژه ۲۳۰,۰۰۰ دلار بوده و نیروی کار تخمینی ۴۶ نفر در ماه است. اگر بخواهید میزان نیروی کار با هر کار فرآیند نرم‌افزاری یا مهندسی مرتبط شده و جداگانه محاسبه می‌گردد.

| فعالیت ← | ا.م. | طرح | تحلیل ریسک | مهندسی | | ساخت | | س.م. | جمع |
|---|---|---|---|---|---|---|---|---|---|
| وظیفه ← | | | | تحلیل | طراحی | برنامه نویسی | تست | | |
| | | | | | | | | | |
| فانکشن | | | | | | | | | |
| ↓ | | | | | | | | | |
| UICF | | | | ۰/۵۰ | ۲/۵۰ | ۰/۴۰ | ۵/۰۰ | n/a | ۸/۴۰ |
| 2DGA | | | | ۰/۷۵ | ۴/۰۰ | ۰/۶۰ | ۲/۰۰ | n/a | ۷/۳۵ |
| 3DGA | | | | ۰/۵۰ | ۴/۰۰ | ۱/۰۰ | ۳/۰۰ | n/a | ۸/۵۰ |
| CGDF | | | | ۰/۵۰ | ۳/۰۰ | ۱/۰۰ | ۱/۵۰ | n/a | ۶/۰۰ |
| DBM | | | | ۰/۵۰ | ۳/۰۰ | ۰/۷۵ | ۱/۵۰ | n/a | ۵/۷۵ |
| PCF | | | | ۰/۲۵ | ۲/۰۰ | ۰/۵۰ | ۱/۵۰ | n/a | ۴/۲۵ |
| DAM | | | | ۰/۲۵ | ۲/۰۰ | ۰/۵۰ | ۲/۰۰ | n/a | ۵/۰۰ |
| | | | | | | | | | |
| | | | | | | | | | |
| جمع | ۰/۲۵ | ۰/۲۵ | ۰/۲۵ | ۳/۵۰ | ۲۰/۵۰ | ۴/۵۰ | ۱۶/۵۰ | | ۴۶/۰۰ |
| | | | | | | | | | |
| درصد نیروی کار | ۷/۱۰ | ۷/۱۰ | ۷/۱۰ | ۷/۸۰ | ۷/۴۵ | ۷/۱۰ | ۷/۳۵ | | |

ا.م. ارتباط با مشتری　　س.م. سنجش و برآورد مشتری

شکل ۵- ۵ جدول برآورد های مبتنی بر فرآیند

نیروی کار کل تخمینی درمورد دامنه نرم‌افزاری CAD از حداقل ۴۶ نفر در ماه تا حداکثر ۵۸ نفر می‌رسد. میانگین برآورد (با استفاده از کل سه روش) ۵۳ نفر- ماه است. حداکثر میزان تغییر از متوسط تخمین زده شده تقریباً ۱۳ درصد است.

وقتی میزان سازگاری بین تخمین‌ها ناچیز است، چه اتفاقی رخ می‌دهد؟ این سئوال نیازمند ارزیابی مجدد اطلاعات استفاده شده برای انجام برآورد است. تخمین‌های بسیار متفاوت را اغلب می‌توان به یک یا دو دلیل پیگیری نمود:

۱- دامنه پروژه به اندازه کافی شناخته نشده یا برنامه ریز برداشت نادرستی از آن داشته است.

۲- اطلاعات بهره‌وری مورد استفاده برای فنون تخمین مبتنی بر مسئله، از نظر برنامه مناسب نیستند، مهجورند (دیگر بطور دقیق ساختار مهندسی نرم‌افزار را منعکس نمی‌کنند) یا درست مورد استفاده قرار نگرفته‌اند. برنامه‌ریز باید علت اختلاف را تعیین نموده و سپس تخمین‌ها را تطبیق دهد.

یک مدل تخمینی مبتنی بر نیروی انسانی پروژه ای است که آنرا هدایت می کند. بنابرین مدل حساس به حوزه خواهد بود.

## ۵-۷ مدل های برآورد تجربی

مدل تخمینی[1] درمورد نرم‌افزار کامپیوتری از فرمولهای تجربی برای پیش‌بینی نیروی کار بعنوان تابعی از LOC یا FP استفاده می‌کند. مقادیر LOC یا FP با استفاده از رشد توصیفی در بخشهای ۵-۶-

1 estimation models

۲ و ۵-۶-۳ تخمین زده شده‌اند. اما بجای استفاده از جداول آمده در این بخشها، مقادیر حاصله برای LOC یا FP در مدل تخمینی وارد شده‌اند.

اطلاعات تجربی که اکثر مدلهای تخمینی را پشتیبانی می‌کنند از یک نمونه محدود پروژه مشتق شده‌اند. به این دلیل، هیچ مدل تخمینی برای تمام رده های نرم‌افزاری مناسب نیست و در همه محیطهای تولیدی کارگر نمی‌باشد. بنابراین نتایج بدست آمده از چنین مدلهایی باید عاقلانه استفاده شوند.[1]

### ۵-۷-۱ ساختار مدل های برآورد

یک مدل تخمینی عادی با استفاده از تحلیل بازگشتی یا رگرسیون بر روی اطلاعات جمع‌آوری شده از پروژه‌های قبلی، بدست می‌آید. ساختار کلی چنین مدلهایی به شکل زیر است : [MAT94][2]

$$E = A + BX(ev)^C \qquad (۲-۵)$$

که در آن B,A و C مقادیر ثابت بدست آمده بصورت تجربی هستند، E نیروی کار بر حسب نفر ماه است و ev متغیر تخمین می‌باشد( LOC یا FP). علاوه بر ارتباط ذکر شده در معادله ۵-۲، اکثر مدلهای تخمین دارای فرمی ازجزء تطابق پروژه هستند که E را بر حسب دیگر مشخصه‌های پروژه (مثل پیچیدگی مسئله، تجربه کارکنان، محیط تولید) مقداردهی می کنند. در میان بسیاری از مدلهای تخمینی مبتنی بر LOC که در این مقاله پیشنهاد شده‌اند داریم:

$E = 5.2 \times (KLOC)^{0.91}$ Walston-Felix model

$E = 5.5 + 0.73 \times (KLOC)^{1.16}$ Bailey-Basili model

$E = 5.2 \times (KLOC)^{1.05}$ Boehm simple model

$E = 5.288 \times (KLOC)^{1.047}$ Doty model for KLOC > 9

مدلهای مبتنی بر FP نیز پیشنهاد شده‌اند. آنها مشتمل اند بر :

$E = -13.39 + 0.0545\ FP$ Albrecht and Gaffney model

$E = 60.62 \times 7.728 \times 10^{-8}\ FP^3$ Keremer model

$E = 585.7 + 15.12\ FP$ Matson, Barnett, and Mellichamp model

---

۱. به طور کلی، یک مدل برآورد باید برای شرایط محلی کالیبره و تنظیم شود. مدل باید با نتایج حاصل از پروژه های تکمیل شده، اجرا شود. داده های پیش گویی کننده مدل باید با نتایج کارآیی مدل در شرایط محلی سنجیده شود اگر پاسخ مناسب نبود، نیاز به اعمال تغییرات وجود دارد.

2.Matson, J.

بررسی سریع مدلهای فوق نشان می‌دهد که هرکدام نتیجه متفاوتی[1] درمورد مقادیر یکسانی از LOC یا FP خواهند داشت. این اشاره مشخص و روشن است. مدلهای تخمینی باید از نظر نیازهای محلی تنظیم گردند.

ارجاع به وب

اطلاعات تفصیلی در خصوص COCOMO II، شامل نرم افزار قابل پیاده سازی از شبکه جهانی، در آدرس زیر قابل دسترسی می باشد:
sunset. Uic. Edu/cocomoii/como.html

### ۵-۷-۲ مدل کوکومو (COCOMO)

باری بوهم [BOE81][2] در کتاب کلاسیک خود به نام «اقتصاد مهندسی نرم‌افزار» یک سری مدلهای تخمین نرم‌افزار معرفی می‌کند که دارای نام COCOMO هستند (مدل هزینه ساختاری[3]) مدل اصلی COCOMO از همه بیشتر مورد استفاده قرار گرفته و مدلهای تخمین هزینه در صنعت مورد بحث قرار گرفتند. مدل اصلی در یک مدل تخمین جامع‌تر به نام کوکومو ۲[4] [BOE96, BOE00][5] تکامل یافته است. مانند نسل قبلی خود COCOMOII یک سری مدلهای تخمین است که حوزه‌های زیر را مورد توجه قرار می‌دهند:

**مدل ترکیبی کاربردی.**

این مدل در طول مراحل اولیه مهندسی نرم‌افزار به کار می رود. هنگامی که نمونه های اولیه رابط کاربر مدلسازی می‌شوند، با درنظر گرفتن رابطه متقابل سیستم و نرم‌افزار، برآورد عملکرد و ارزیابی بلوغ فنآوری که البته بسیار حجیم و عظیم اند.

**مدل مرحله اولیه طراحی.**

وقتی بکار می رود که نیازمندیها مشخص و تثبیت شده و معماری مقدماتی نرم‌افزار ایجاد شده است.

**مدل مرحله آخر معماری.**

در طول ساخت نرم‌افزار بکار می رود.

---

۱. یک علت آن است که این مدل ها اغلب از پروژه هایی در حیطه های خاص کاربردی به دست آمده اند.

2.Boehm, B.

3.Putnam, L.

4.COCOMO II

5.Boehm, B.

مانند همه مدلهای تخمینی نرم افزار، COCOMOII که در بالا توصیف شد، نیازمند اطلاعات اندازه گیری است. سه گزینه مختلف از نظر اندازه بعنوان بخشی از سلسله مراتب مدل مهیا شده‌اند: امتیاز شیء[1]، امتیازکارکردی و خطوط کد منبع.

مدل ترکیب برنامه COCOMOII از امتیاز شیء استفاده کرده و در بندهای بعدی توضیح داده می‌شود. باید توجه داشت که دیگر مدلهای پیچیده تخمینی (با استفاده از FP و KLOC) نیز بعنوان بخشی از COCOMOII مهیا می‌باشند.

| نوع | وزن پیچیدگی | | |
|---|---|---|---|
| | ساده | متوسط | مشکل |
| نمایشگر | ۱ | ۲ | ۳ |
| گزارش | ۲ | ۵ | ۸ |
| مولفه | - | - | ۱۰ |

**جدول ۵-۱ وزن پیچیدگی برای انواع اشیاء [BOE96]**

پرسش
امتیاز شیئی OP چیست؟

مانند امتیازات کارکردی (فصل ۴) امتیاز شئ یک معیار نرم‌افزاری غیرمستیم است که با استفاده از شمارش تعداد ۱- صفات نمایش(در رابطه کاربر) ۲- گزارشات و ۳- اجزایی که احتمالاً برای ساخت برنامه لازمند محاسبه می‌شود. هر مورد شیء (مثلف صفحه یا گزارش) در سطوح پیچیدگی یکی از این سه، گروه‌بندی می‌شود. (یعنی ساده، متوسط یا مشکل) که با استفاده از معیارهای پیشنهادی بوهم [BOE96][2]صورت می‌گیرد. در اصل، پیچیدگی تابعی از تعداد و منبع جدول اطلاعات خادم/ مخدوم است که برای تولید صفحه یا گزارش لازم بوده و تعداد دیدگاهها یا بخش ها بعنوان بخشی از صفحه یا گزارش ارائه شده‌اند.

وقتی پیچیدگی تعیین شد، تعداد صفحات، گزارشات و اجزاء طبق جدول ۵-۱ تعیین می شوند. سپس شمارش امتیاز شیء با ضرب عدد اصلی عبارات شیء در عامل ارزیابی جدول ۵-۲ تعیین شده و جمع بندی می‌شود تا تعداد کل امتیاز شیء بدست آید. وقتی تولید مبتنی بر جزء یا کاربرد مجدد کلی نرم‌افزار بکار گرفته شد، درصد این استفاده مجدد (%reuse) تخمین زده شده و تعداد امتیاز شیء تنظیم می‌شود:

$$NOP = (\text{object Points}) \times [100 - \%\text{reuse})/100]$$

که در آن NOP بصورت «امتیاز جدید شیء» تعریف شده است.

1.object point
2.Boehm, B.

برای بدست آوردن تخمین نیروی کار براساس مقدار NOP محاسبه شده، باید «میزان بهره‌وری» را به دست آورد. جدول ۵-۲ نمایانگر میزان بهره‌وری درمورد سطوح مختلفی از تجربه تولید کننده و بلوغ محیط توسعه است:

PROD = NOP / Person - month

| قابلیت/ تجربه | خیلی کم | کم | متوسط | بالا | خیلی بالا |
|---|---|---|---|---|---|
| قابلیت/ بلوغ محیط | خیلی کم | کم | متوسط | بالا | خیلی بالا |
| بهره وری (PROD) | 4 | 7 | 13 | 25 | 50 |

جدول ۵ـ۲ نرخ های بهره وری برای امتیازات اشیاء [BOE96]

وقتی میزان بهره وری تعیین شد، تخمین نیروی کار پروژه بصورت زیر حاصل می‌شود:

estimated effort= NOP / PROD

در مدلهای پیشرفته تر COCOMOII،[1] یک سری فاکتورهای سنجش، محرکهای هزینه و روشهای تطابقی لازمند. بحث کلی و کامل این موارد فراتر از دامنه این کتاب است. خوانندگان علاقمند را به [BOE00][2] یا بازدید از وب سایت COCOMOII ارجاع می‌دهیم.

### ۵-۷-۳ معادله نرم افزار (فرمول ها)

معادله نرم‌افزار [PUT92][3] یک مدل دینامیک چند متغیره است که توزیع نیروی کار معینی را درطول حیات پروژه تولید نرم‌افزار فرض می‌گیرد. مدل با استفاده از اطلاعات بهره‌وری جمع‌آوری شده از ۴۰۰۰ پروژه معاصر نرم‌افزاری ساخته شده است. براساس این اطلاعات، مدل تخمینی با شکل زیر خواهیم داشت:

$$E = [LOC \times B^{0.333} / P]^3 \times (1 / t^4) \qquad (۳-۵)$$

که در آن :

E= نیروی کاری فرد در ماه یا فرد در سال است.

T= مدت پروژه به ماه یا سال.

---

۱. همانطور که پیشتر متذکر شدیم، این مدل ها FP و LOC (امتیازکارکردی و تعداد خطوط برنامه‌نویسی) را به کار می برند.

2.Boehm, B.

3.Putnam, L.

B= عامل مهارتهای ویژه.[1]

P=پارامتر بهره وری که منعکس کننده موارد زیر می باشد :

- تکامل کلی فرآیند و شیوه های مدیریتی
- میزان و مقیاسی که نسبت به آن روشهای خوب مهندسی نرم‌افزاری استفاده می‌شوند.
- سطح زبانهای برنامه‌نویسی مورد استفاده.
- وضعیت محیط نرم‌افزاری.
- مهارتها و تجربه تیم نرم‌افزاری.
- پیچیدگی برنامه کاربردی.

مقادیر نوعی ممکن است عبارت باشد از: P= 2000 برای تولید نرم‌افزار نهفته در زمان حقیقی. P= 10000 برای نرم‌افزاری سیستمها و ارتباطات راه دور. P= 28000 برای برنامه‌های کاربردی سیستمهای تجاری.[2] می‌توان پارامتر بهره وری را از نظر شرایط محلی با استفاده از اطلاعات تاریخی جمع‌آوری شده از کارهای گذشته ارائه داد.

نکته مهم مورد توجه این است که معادله نرم‌افزار دارای دو پارامتر مستقل است:

۱- تخمینی از اندازه (LOC) .

۲- نشانگری از مدت پروژه به ماهها یا سالهای تقویمی.

به منظور ساده سازی فرآیند تخمین و استفاده از رایج ترین شکل برای مدل تخمین، پوتنام و مایرز [PUT92][3] مجموعه ای از معادلات را بیان می‌دارند که از معادله نرم‌افزاری مشتق می‌شوند. حداقل زمان تولید بصورت زیر تعریف شده:

$$t_{min} = 8.14\,(LOC / P)^{0.43} \qquad \text{برای } t_{min} > 6 \text{ month} \qquad \text{(۴-۵ الف) بر حسب ماه}$$

$$E = 180\,Bt^3 \qquad \text{برای } E \geq 20 \qquad \text{(۴-۵ ب) بر حسب نفر- ماه}$$

توجه کنید که t در معادله (۴-۵ ب) به سال نمایش داده شده است.

با استفاده از معادله (۴-۵) با P= 12000 (مقدار توصیه شده برای نرم‌افزار علمی) درمورد نرم‌افزار CAD که پیش‌تر در این فصل مورد بحث قرار گرفته داریم:

$$t_{min} = 8.14\,(33200 / 12000)^{0.43}$$

$$t_{min} = 12.6 \text{ ماه}$$

$$E = 180 \times 0.28 \times (1.05)^3$$

---

۱. B رشدی آهسته دارد، مطابق با رشد " نیاز به جامعیت، آزمون، تضمین کیفیت، مستندسازی و مهارتهای مدیریتی [PUT92] " خواهد داشت. برای برنامه های کوچک ( بین ۵ تا ۱۵ هزارخط برنامه) B = 0.12 و برای برنامه های بزرگتر از ۷۰ هزار خط B= 0.39 خواهد بود.

۲. توجه به این نکته مهم است که پارامتر بهره وری، می توند به طور تجربی از داده های پروژه های محلی تحصیل شود.

3.Putnam, L.

ارجاع به وب

اطلاعات در خصوص ابزار های تخمین هزینه نرم افزار که فرمول های نرم افزاری را شامل می شود، در آدرس زیر قرار دارد : www.qsm.com

$E = 58$ نفر - ماه

نتایج معادله نرم‌افزاری بطرز مطلوبی با تخمینهای ارائه شده در بخش ۵-۶ مرتبط است. مانند مدل COCOMO که در بخش قبلی آمد، معادله نرم‌افزار در طول دهه گذشته تکامل یافته است. بحث بیشتر درمورد نسخه توسعه یافته این تخمین را می‌توان در [PUT97b][1] یافت.

## ۵-۸ تصمیم گیری ساخت / خرید[1]

توصیه

مواقعی خواهند بود که نرم افزار خارج از طبقه ( حاضر و آماده ) یک راه حل تمام عیار و کامل خواهد بود، مگر برای ویژگیهای خاص که شما بدون آن نتوانید بسر برید. در بیشتر موارد زندگی بدون ویژگیهای خاص ارزشمند است!

در حوزه‌های متعددی از برنامه‌های کاربردی مقرون به صرفه‌تر آن است که بجای تولید نرم‌افزار کامپیوتر آن را خریداری کنیم. مدیران مهندسی نرم‌افزار با تصمیم‌گیری در خرید یا تولید مواجه هستند که می‌تواند با چند گزینه درمورد خرید پیچیده‌تر شود:

۱- ممکن است نرم‌افزار بصورت آماده خریداری شود.

۲- اجزای نرم‌افزاری بصورت «با تجربه کامل» یا «با تجربه نسبی» بدست آیند و سپس اصلاح شده و یکپارچه گردند تا نیازها را برآورده سازند.

۳- ممکن است نرم‌افزار توسط پیمانکاری خارج از سازمان طبق سفارش ساخته شده باشد تا نیازهای خریدار را مرتفع سازد.

مراحل درگیر در خرید نرم‌افزار ازنظر اهمیت نرم‌افزار مورد خریداری و هزینه نهایی تعریف شده‌اند. درمورد محصولات نرم‌افزاری گرانتر، رهنمودهای زیر را می‌توان بکار گرفت:

۱- مشخصاتی برای کارکرد و عملکرد نرم‌افزار مطلوب ارائه می‌دهد. مشخصه‌های قابل ارزیابی را هر گاه که ممکن باشد، به ما می‌دهد.

۲- هزینه اولیه تا تولید و تاریخ تحویل را می‌گوید.

۳- الف- سه یا چهار برنامه نمونه انتخاب می‌کند که به بهترین نحو با مشخصه‌های شما جور می‌شود.

۳- ب- اجزای نرم‌افزاری قابل استفاده مجددی انتخاب می‌کند که در ساخت برنامه مورد نیاز، شما را یاری می‌کنند.

۴ – شبکه مقایسه‌ای تشکیل می‌دهد که نمایانگر مقایسه سر به سر عملیات اصلی است. متناوباً، آزمونهای سنجشی برای مقایسه نرم‌افزارمعرفی شده انجام می‌دهد.

۵ – هر بسته نرم‌افزاری یا مولفه را بر اساس کیفیت محصول گذشته، پشتیبانی فروشنده، راهنمای محصول شهرت و غیره ارزیابی می‌کند.

پرسش

آیا راه سیستماتیک ای برای مرتب سازی تصمیم گیری در خصوص گزینه های خرید/ ساخت وجود دارد؟

---

1.Putnam, L.

۶ – با سایر کاربران نرم‌افزار تماس گرفته، نظراتشان را جویا می‌شود.

در تحلیل نهایی، تصمیم‌گیری خرید یا تولید بر اساس شروط زیرین تعیین می‌گردد: ۱) آیا تاریخ تحویل محصول زودتر از مدت زمان لازم برای نرم‌افزاری است که در داخل تولید می‌شود؟ ۲) آیا هزینه خرید بعلاوه هزینه سفارش کمتر از هزینه تولید آن است؟ ۳) آیا هزینه پشتیبانی خارجی ( قرار داد نگهداری) کمتر از هزینه پشتیبانی داخلی است؟ این شرایط برای هر گزینه ذکر شده فوق بکار می‌رود.

### ۵-۸-۱ ایجاد یک درخت تصمیم گیری

مراحل توصیف شده فوق را می‌توان با استفاده از فنون آماری مثل تحلیل درختی[BOE89] [۱۲] در تصمیم‌گیری مورد ارزیابی قرار داد. مثلاً، شکل۵-۶، نمودار درختی در تصمیم گیری را برای سیستم مبتنی بر نرم‌افزاری به نام X نشان می‌دهد. در این مورد، سازمان مهندسی نرم‌افزاری می‌تواند:

۱- سیستم X را از نخست و تماماً بسازد.

۲- اجزای موجودی که در مورد آنها تجربه نسبی داریم را مجدداً مورد استفاده قرار داده تا سیستم را بسازد.

۳- محصول نرم‌افزاری موجود را خریده و آن را اصلاح می‌کند تا نیازهای محلی را برطرف نماید.

۴- تولید نرم‌افزار را با یک فروشنده خارجی قرارداد می‌بندد.

اگر قرار است سیستم توسط خودمان ساخته شود، ۷۰ درصد این احتمال وجود دارد که کار سخت باشد. با استفاده از فنون تخمین مورد بحث در این فصل، برنامه‌ریز پروژه پیش‌بینی می‌کند که یک تولید مشکل، نیروی کاری با هزینه ۴۵۰،۰۰۰ دلار می‌خواهد. و اگر توسعه ساده باشد۳۸۰،۰۰۰ دلار برآورد می‌شود. ارزش مورد انتظار برای هزینه، که در طول هر شاخه از درخت محاسبه شده می‌شود :

هزینه مورد نظر = $\sum$ (مسیر احتمالی) × (هزینه مسیر تخمینی) ؛

که در آن i مسیر تصمیم گیری است. برای ایجاد مسیر تولید و ایجاد،

expected Coct build = 0.30 (£380k)+0.70(£4500k) = £ 429k

به دنبال دیگر مسیرهای درختی در مورد تصمیم‌گیری، هزینه‌های ایجاد برای کاربرد مجدد، خرید و قرارداد تحت یک سری شرایط نیز نشان داده شده‌اند. هزینه‌های منتظره در مورد این مسیرها عبارتند از:

expected Coct reuse = 0.4 (£275k)+0.60[0.20(£310k)+0.8(£490k)] = £ 382k

expected Coct buy = 0.7 (£210k)+0.30 (£400k) = £ 267k

expected Coct contract = 0.60 (£350k)+0.40(£500k) = £ 410k

ارجاع به وب

آموزشی کامل در خصوص تحلیل درخت تصمیم گیری در آدرس زیر وجود دارد : www.demon.co.uk/mindtool/dectree.html

1.make /buy decision

2.decision tree analysis

براساس هزینه‌های احتمالی و برنامه‌ریزی شده که در شکل ۵-۶ به آن اشاره شد، کمترین هزینه مورد انتظار گزینه buy (خرید) است.

نکته مهمی که باید به آن توجه داشت این است که بسیاری از معیارها، نه فقط هزینه، باید در طول فرآیند تصمیم گیری مدنظر باشند. قابلیت دسترسی، تجربه تولیدکننده/ فروشنده/ پیمانکار، تطابق با شرایط، سیاستهای محلی و احتمال تغییر وجود دارند اما تعداد معدودی از این معیارها هستند که ممکن است بر تصمیم نهایی مبتنی بر استفاده مجدد، خرید یا قرارداد تأثیر بگذارند.

### ۵-۸-۲ استفاده از منابع خارجی

دیر یا زود هر شرکتی که نرم‌افزار کامپیوتری تولید می‌کند با یک سئوال بنیادی روبرو می‌شود: «آیا راهی هست که بتوانیم از طریق آن نرم‌افزار و سیستم‌هایی را که لازم داریم با کمترین قیمت بدست بیاوریم؟» جواب این سئوال چندان ساده نیست و مباحث اساسی که در واکنش به این سئوال رخ می‌دهند همیشه منجر به یک کلمه می‌شوند: خرید کالا از خارج از سازمان (outsourcing).

منابع خارجی از نظر مفهوم بسیار ساده است. کارهای طراحی نرم‌افزار با شخص ثالثی قرارداد بسته می‌شوند که این شخص با کمترین هزینه و امیدواریم با بالاترین کیفیت، کار را انجام دهد.

تصمیم‌گیری درمورد خرید کالا از خارج یک تصمیم راهبردی یا تاکتیکی است. از نظر راهبردی مدیران تجاری این مسئله را در نظر می‌گیرند که آیا بخش مهمی از کل کار نرم‌افزار را می‌توان بصورت انعقاد قرارداد با دیگران انجام داد. از نظر تاکتیکی، مدیر پروژه معلوم می‌کند که آیا بخشی از پروژه یا کل آن می‌تواند به بهترین نحو توسط قراردادهای فرعی حاصل گردد یا خیر. [MIN95][1]

بدون توجه به گستردگی مطلب موردتوجه، تصمیم‌گیری درمورد خرید خارجی اغلب یک مسئله مالی است. بحث دقیقی از تحلیل مالی خرید خارجی فراتر از دامنه این کتاب بوده و بهتر است

بر عهده دیگران گذاشته شود.خلاصه بازنگری موارد موافق و مخالف این تصمیم‌گیری مفید و ارزشمند است.

از جنبه مثبت این قضیه، معمولاً پس اندازهای هزینه‌ای با کاهش تعداد افراد مربوط به کار نرم‌افزاری و تسهیلات و تأسیسات مربوطه، حاصل می‌گردد (مثل کامپیوترها، زیربنا) که از آنها حمایت می‌کنند. از جنبه منفی آن، شرکت کنترل نرم‌افزاری را که به آن نیاز دارد از دست می‌دهد. از آنجا که نرم‌افزار فنآوری است‌که بین سیستمها، خدمات و محصول تفاوت قائل می‌شود، شرکت(به نوعی ) ریسک نموده و سرنوشت رقابتش را در اختیار شخص ثالث می‌گذارد.

نقل قول

به قاعده این است ، نیازهای استفاده از منابع خارجی، مهارتهای مدیریتی بالاتری را نسبت به توسعه در محیط کاری خود( با استفاده از نیروی کاری خود)، طلب می کند.

ارجاع به وب

اطلاعات مفیدی در خصوص منابع خارجی (اشاره گرها، مقالات) را می توانید در آدرس زیر بیابید :
www.outsourcing.com

1.Boehm, B.

## ۵-۹ابزارهای خودکار برآورد

ابزار برآورد

فنون تفکیک سازی و مدلهای تجربی تخمینی که در بخشهای قبلی توصیف شدند، بعنوان بخشی از طیف وسیعی از ابزارهای نرم‌افزاری مهیا می‌باشند. این ابزارهای تخمین خودکار به برنامه‌ریز امکان می‌دهند هزینه و نیروی کار را برآورد نموده و تحلیلهای«What if» (چه می‌شود اگر) را برای متغیرهای مهم پروژه مانند تاریخ تحویل یا کارکنان اجرا کند. گرچه ابزارهای تخمین خودکار بسیاری وجود دارند اما همه مشخصه‌های عمومی یکسانی نشان داده و همه شش عملکرد کلی زیر را اجرا می‌کنند: [JON96][2]

۱- **اندازه بندی موارد قابل تحویل پروژه.** اندازه یک یا چند محصول کار نرم‌افزار تخمین زده می‌شود. محصولات کار شامل نمایش نرم‌افزار از نظر خارج آن (صفحه نمایش، گزارشات)، خود نرم‌افزار (مثل KLOC)، قابلیت کاری ارائه شده (امتیازکارکردی)، اطلاعات توصیفی (مثل مستندات) می‌باشد.

۲- **انتخاب فعالیتهای پروژه.** چارچوب فرآیند مناسب (فصل ۲) انتخاب شده و مجموع کار مهندسی نرم‌افزار مشخص می‌گردد.

۳- **پیش‌بینی میزان کارکنان.** تعداد افرادی که برای انجام این کار تعیین شده‌اند. از آنجا که ارتباط بین افرادی که در دسترسی هستند و کار ( پیش بینی شده) بسیار غیر خطی است، این یکی از اطلاعات بسیار مهم است.

۴- **پیش‌بینی نیروی کاری نرم‌افزار.** ابزارهای تخمین از یک یا چند مدل استفاده می‌کنند (مثل بخش ۷-۵) که اندازه مقادیر قابل ارائه پروژه را با نیروی لازم برای تولیدشان مرتبط می‌کنند.

۵- **پیش‌بینی هزینه نرم‌افزار.** با فرض نتایج مرحله ۴، هزینه ها را می‌توان با تخصیص میزان نیروی کار با فعالیتهای پروژه‌ای فوق‌الذکر محاسبه کرد.

۶- **پیش‌بینی جداول زمانی کار.** وقتی همه چیز مشخص شد، پیش‌نویس جدول زمانی با تخصیص نیروی کار در طول همه فعالیتهای طراحی مهندسی براساس مدلهای توصیه شده برای توزیع نیرو، ارائه می‌شود. وقتی ابزارهای مختلف برآورد در اطلاعات پروژه بکار گرفته شدند، تنوع نسبتاً زیادی در نتایج تخمین بوقوع می‌پیوندد. مهمتر اینکه مقادیر پیش‌بینی شده گاهی اوقات بسیار متفاوت از مقادیر واقعی است. این کار باعث تداعی جدی این مطلب می‌شود که بازده ابزارهای تخمین باید بعنوان امتیاز اطلاعاتی از روی تخمین‌های ارائه شده استفاده شود نه بعنوان تنها منبع تخمین.

---

1.Minoly, D.

2.Jones, C.

## ۵-۱۰ خلاصه

برنامه‌ریز پروژه نرم‌افزاری باید قبل از شروع کار سه چیز را تخمین بزند: چقدر زمان می‌برد، چه مقدار نیروی کار لازم است و چند نفر در آن درگیر می‌شوند. علاوه بر این، طراح باید منابعی را (سخت‌افزار و نرم‌افزار) که برای کار لازمند و همینطور میزان مخاطرات موجود را پیش‌بینی کند.

تعریف دامنه به برنامه‌ریز کمک می‌کند تا با استفاده از یک یا چند تکنیک که در دو دسته بزرگ قرار می‌گیرند، برآورد را انجام دهد: تکنیک تفکیک و مدلهای تجربی. فنون تفکیک نیازمند به تصویر کشیدن عملیات نرم‌افزاری عمده هستند که در دنباله تخمین‌هایی از ۱- تعداد خطوط برنامه LOC ۲- مقادیر منتخب در دامنه اطلاعاتی، ۳- تعداد افراد در هر ماه که برای اجرای هر کارکرد لازمند یا ۴- تعداد افراد لازم در هر ماه برای اجرای هر فعالیت مهندسی، به کار می‌آید. تکنیک های تجربی از حس تجربه درمورد کار و زمان منتج می شوند تا کمیت این پروژه‌ها را پیش‌بینی کنند. می‌توان از ابزارهای خودکار برای اجرای مدل تجربی خاصی استفاده نمود.

معمولاً تخمین‌های دقیق در یک پروژه از حداقل دو تا سه تکنیک فوق استفاده می‌کنند. با مقایسه و بررسی تخمین‌های ارائه شده از فنون مختلف، احتمالاً برنامه‌ریز به تخمین دقیق‌تری دست می‌یابد. تخمین پروژه نرم‌افزاری هرگز علم دقیقی نیست، اما ترکیبی از اطلاعات تاریخی خوب و فنون نظام مند است که می‌توانند میزان دقت را در آن بهبود بخشند.

## مسایل و نکاتی برای تفکر و تعمق بیشتر

۵-۱ فرض کنید که شما مدیر پروژه شرکتی هستید که کار تولید نرم افزار محصولات سفارشی یا انجام می دهد. شما قراردادی برای ساخت نرم‌افزار یک سیستم امنیت خانگی عقد نموده اید. جمله ای برای حوزه کاربرد بنویسید که نرم‌افزار را توصیف کند. اطمینان حاصل کنید که جمله شما مقید شده (دارای مرز مشخص )باشد. اگر با سیستم‌های امنیتی خانگی آشنایی ندارید، پیش از نوشتن، اندکی تحقیق کنید. جایگزین : به‌جای سیستم امنیتی خانگی هر چیز دیگری را که علاقه دارید، در نظر بگیرید.

۵-۲ پیچیدگی پروژه نرم‌افزاری مختصراً" در بخش ۵-۱ بحث شد. فهرستی از خصوصیات نرم‌افزار (مثل عملیات همروند، خروجی گرافیکی و غیره) تهیه کنید که بر پیچیدگی نرم‌افزار تاثیرگذار خواهند بود. این لیست مطابق اولویت ها تنظیم کنید.

۵-۳ عملکرد از ملاحظات مهم طرح ریزی به شمار می‌رود. بحث کنید که عملکرد چگونه می‌تواند بدون وابستگی به حیطه کاربردی نرم افزار و به گونه ای متفاوت تفسیر گردد.

۵-۴ تجزیه و تفکیک عملیاتی را برای نرم‌افزار سیستم امنیت خانگی که در مسئله ۵-۱ توضیح داده اید، انجام دهید. اندازه هر عملکرد را برحسب LOC برآورد کنید. با فرض این‌که سازمان شما LOC/pm ۴۵۰ (خط به ازای هر نفر ماه) را با نرخ کار ۷۰۰۰ دلار به‌ازای هر نفر ـ ماه تولید می‌کند، نیروی کار و هزینه لازم برای ساخت نرم‌افزار را با استفاده از فنون برآورد مبتنی بر تعداد خطوط که در بخش ۵-۶-۳ شرح داده شد، برآورد کنید.

۵-۵ با استفاده از امتیازات عملکرد سه‌بعدی که در فصل ۴ بحث شد، تعداد FP را برای نرم‌افزار سیستم امنیت خانگی محاسبه کنید و نیروی کار و هزینه را با استفاده از فنون برآورد مبتنی بر FP، شرح داده شده در بخش ۵-۶-۴، برآورد کنید.

۵-۶ مدل کوکومو۲ را به کار برده و نیروی کار لازم برای ساخت نرم‌افزاری ساده جهت یک خودپرداز بانک (ATM) را برآورد کنید. فرض کنید ۱۲ صفحه نمایش و ۱۰ گزارش تولید می شود و حدود ۸۰ جزء نرم افزاری نیاز دارد. پیچیدگی و نیز بلوغ سازنده/ محیطی را متوسط فرض کنید. از مدل ترکیب کاربردی و امتیازات شیء بهره برید.

۵-۷ برای برآورد نرم‌افزار سیستم امنیت خانگی از معادله نرم‌افزار استفاده نمایید. فرض کنید معادلات (۵ـ۴) مورد استفاده واقع می شوند و P=۸۰۰۰ می باشد.

۵-۸ نیروی کار برآورد شده در مسائل ۵-۴ ، ۵ـ۵ و ۵-۷ را با هم مقایسه کنید. یک برآورد منفرد با استفاده از برآورد سه امتیازی به دست آورید. انحراف معیار استاندارد چیست و درجه قطعیت برآورد شما را چگونه متأثر می سازد؟

۵-۹ با نتایج به‌دست آمده در مسئله ۵-۸، تعیین کنید که آیا انتظار ساخت نرم افزار طی شش ماه آینده قابل قبول است و برای انجام این امر، چند نفر مورد نیاز می‌باشد؟

۵-۱۰ یک یا چند تکنیک برآورد شرح داده شده در این فصل را با استفاده از یک مدل صفحه گسترده، پیاده‌سازی کنید. به جای آن : یک یا چند مدل روی خط برای برآورد از منابع مبتنی بر وب بدست آورید.

۵-۱۱ برای یک تیم پروژه، یک ابزار نرم‌افزاری بسازید که هریک از فنون برآورد توسعه یافته در این فصل را پیاده‌سازی کند

۵ـ۱۲ غریب به نظر می رسد که برآوردهای هزینه و زمان بندی طی مرحله طرح ریزی پروژه انجام گیرد- پیش از آنکه تحلیل یا طراحی جزئیات نیازمندیها انجام گیرد. شما در این خصوص چگونه فکر می کنید؟ آیا شرایط محیطی در این امر دخیل است؟

۵ـ۱۳ با در نظر گرفتن ۵۰ در صد احتمال برای هر شاخه از درخت تصمیم گیری شکل ۵ـ۶ مقادیر مورد انتظار را مجددا" محاسبه نمایید. آیا این امر بر تصمیم نهایی شما تاثیری خواهد داشت؟

## فهرست منابع و مراجع

[BEN92] Bennatan, E.M., *Software Project Management: A Practitioner's Approach*, McGraw-Hill, 1992.
[BOE81] Boehm, B., *Software Engineering Economics*, Prentice-Hall, 1981.
[BOE89] Boehm, B., *Risk Management*, IEEE Computer Society Press, 1989.
[BOE96] Boehm, B., "Anchoring the Software Process," *IEEE Software*, vol. 13, no. 4, July 1996, pp. 73-82.
[BOEOO] Boehm, B., et al., *Software Cost Estimation in COCOMO II*, Prentice-Hall, 2000.
[BR075] Brooks, F., *The Mythical Man-Month*, Addison-Wesley, 1975.
[GAU89] Gause, D.c. and G.M. Weinberg, *Exploring Requirements: Quality Before Design*, Dorset House, 1989.
[HO091] Hooper, J. and R.O. Chester, *Software Reuse: Guidelines and Methods*, Plenum Press, 1991.
[JON96] Jones, c., "How Software Estimation Tools Work," *American Programmer*, vol. 9, no. 7, July 1996, pp. 19-27.
[MAT94] Matson, J., B. Barrett, and J. Mellichamp, "Software Development Cost Estimation Using Function Points," *IEEE Trans. Software Engineering*, vol. SE-20, no. 4, April1994,pp.275-287.
[MIN95] Minoli, D., *Analyzing Outsourcing*, McGraw-Hill, 1995.
[PHI98] Phillips, D., *The Software Project Manager's Handbook*, IEEE Computer Society Press, 1998.
[PUT92] Putnam, L. and W. Myers, *Measures for Excellence*, Yourdon Press, 1992.
[PUT97a] Putnam, L. and W. Myers, "How Solved Is the Cost Estimation Problem?" *IEEE Software*, November 1997, pp. 105-107.
[PUT97b] Putnam, L. and W. Myers, *Industrial Strength Software: Effective Management Using Measurement*, IEEE Computer Society Press, 1997.
[ZUS97] Zuse, H., *A Framework for Software Measurement*, deGruyter, 1997.

## خواندنیهای دیگر و منابع اطلاعاتی

Most software project management books contain discussions of project estimation. Jones *(Estimating Software Costs*, McGraw-Hill, 1998) has written the most comprehensive treatment of the subject published to date. His book contains models and data that are applicable to software estimating in every application domain. Roetzheim and Beasley *(Software Project Cost and Schedule Estimating: Best Practices*, Prentice-Hall, 1997) present many useful models and suggest step-by-step guidelines for generating the best possible estimates.

Phillips [PHI98], Bennatan *(On Time, Within Budget: Software Project Management Practices and Techniques*, Wiley, 1995), Whitten *(Managing Software Development Projects: Formula for success*, Wiley, 1995), Wellman *(Software Costing*, Prentice-Hall, 1992), and Londeix *(Cost Estimation for Software Development*, Addison-Wesley, 1987) contain useful information on software project planning and estimation.

Putnam and Myer's detailed treatment of software cost estimating ([PUT92] and [PUT97b]) and Boehm's books on software engineering economics ([BOE81] and COCOMO II [BOEOO]) describe empirical estimation models. These books provide detailed analysis of data derived from hundreds of software projects. An excellent

book by DeMarco *(Controlling Software Projects,* Yourdon Press, 1982) provides valuable insight into the management, measurement, and estimation of software projects. Sneed *(Software Engineering Management,* Wiley, 1989) and Macro *(Software Engineering: Concepts and Management,* Prentice-Hall, 1990) consider software project estimation in considerable detail.

Lines-of-code cost estimation is the most commonly used approach in the industry. However, the impact of the object-oriented paradigm (see Part Four) may invalidate some estimation models. Lorenz and Kidd *(Object-Oriented Software Metrics,* Prentice-Hall, 1994) and Cockburn *(Surviving Object-Oriented Projects,* Addison-Wesley, 1998) consider estimation for object-oriented systems.

A wide variety of information sources on software planning and estimation is available on the Internet. An up-to-date list of World Wide Web references that are relevant to software estimation can be found at the SEPA Web site:
**http://www.mhhe.com/engcs/compsci/pressman/resources/project-plan.mhtml**