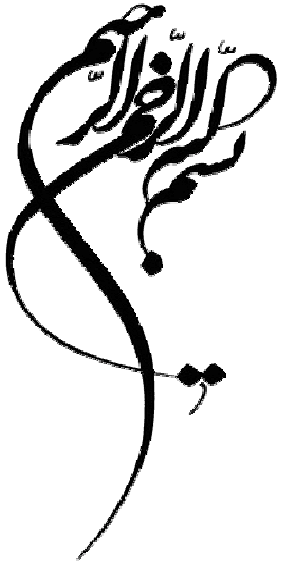


نام درس

مهندسی نرم افزار ۱

تهیه و گرد آوری : ایمان مختاری

بهار ۸۸

نام درس:مهندسی نرم افزار ۱

فصل ۱ – محصول

مقدمه

يك بناي ساختماني چيست؟

يك ساختمان محصولي از يك يا چند بنا است و شامل مصاح ساختماني چون سيمان؛گچ و ... است

نرم افزار كامپيوتري چيست؟

محصولي از يك يا چند مهندس نرم افزار است كه شامل برنامه ها و داده ها است

نرم افزار كامپيوتري را چه كساني استفاده مي كند؟

هر فردي ممكن است به صورت مستقيم يا غير مستقيم از آن استفاده كند.

علت اهميت نرم افزار؟

تاثير گسترده بر تمام جنبه هاي زندگي

مراحل ايجاد نرم افزار چگونه است؟

با بكارگيري فرآيندي خاص و طي كردن يك روند مشخص

آشنا شدن با ميزان نفوذ نرم افزار

تاثيري بسيار ژرف و عميق دارد و نمونه بارز آن ماجراي y2k در اواخر سال ۱۹۹۹ بود

نقش نرم افزار؟

نقش دو گانه اي دارد
- خودش يك محصول است و براي بكارگيري سخت افزار استفاده مي شود
- وسيله اي براي ساخت و تحويل محصول به حساب مي آيد.

نام درس:مهندسی نرم افزار ۱

- نرم افزار چه در يك تلفن سلولي قرار داشته باشد يا درون يك كامپيوتر بزرگ،يك **انتقال دهنده** اطلاعات محسوب مي گردد.
يعني اطلاعاتي را **توليد** مي كند ، **سازمان** مي دهد، **تحويل** مي گيرد، **نمايش** يا انتقال مي دهد.
اگر از آن به عنوان وسيله انتقال دهنده محصول استفاده شود، به صورت پايه و اساس كنترل كامپيوتر (سيستم عامل) ، تبادل اطلاعات (شبكه ها) و ايجاد و كنترل برنامه هاي ديگر(ابزارهاي نرم افزاري و محيط آن) استفاده خواهد شد.

نمايش يا انتقال + تحويل + سازماندهي + توليد ⇨ **انتقال دهنده**

تاريخچه نرم افزار

بحران نرم افزار اولين بار بصورت رسمی در كنفرانسی در سال ۱۹۶۸ ميلادی مطرح گرديد . عوامل اصلی اين بحران عبارتند از :

هزينه بالای ايجاد نرم افزار	تاخير در توليد و تحويل نرم افزار
نگهداری پرهزينه نرم افزار	پيشرفت سريع سخت افزار
كيفيت پايين نرم افزار	افزايش پيچيدگی محصولات

انواع روشهاي برنامه نويسي رايج
- برنامه نويسي منفرد
- برنامه نويسي تيمي

مشكلات برنامه نويسي منفرد طولاني شدن زمان پروژه-افزايش هزينه ها-مشكل خطايابي-مشكل در ارزيابي ميزان پيشرفت

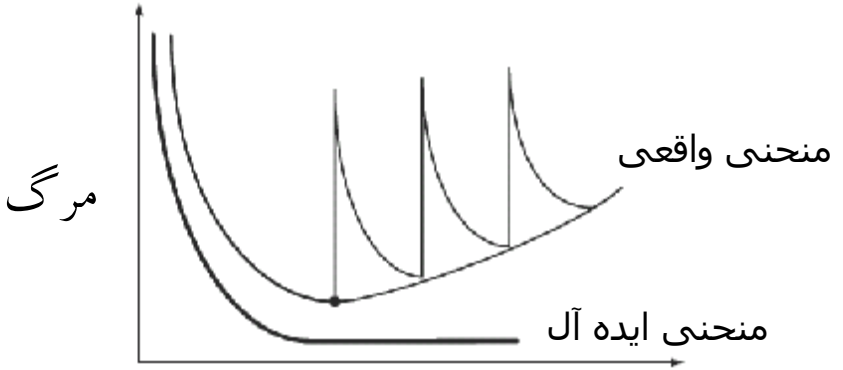
خصوصيات نرم افزار

بهترين راه شناخت مشخصات نرم افزار ،**مقايسه آن با يك جزء ديگر** ار محصولات ساخت انسان است (مثلا مقايسه با سخت افزار)

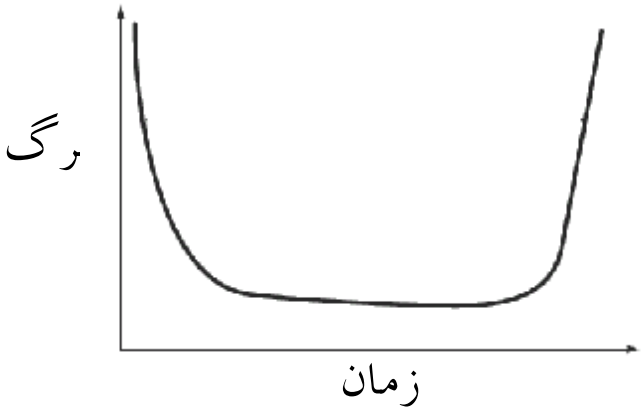
نتیجه مقایسه نرم افزار با سخت افزار

۱-نرم افزار توسعه و طراحي مي شود، اما ساخته نمي شود (فيزيكي نيست)

۲-نرم افزار كهنه نمي شود ، اما زوال يافته و منسوخ مي شود.(عدم حساسيت به عوامل محيطي)



نمودار نرخ خرابی نرم افزار



نمودار نرخ خرابی سخت افزار

۳-گرچه صنعت به سمت مونتاژ قطعات حركت مي كند ، اما نرم افزار سفارشي ساخته مي شود و قابل استفاده فراوان است.

يك قطعه نرم افزاري بايد طراحي و اجرا شود به گونه اي كه در بسياري از برنامه هاي مختلف ديگر استفاده شود.

كاربردهای نرم افزار

۱-اجراي رويه اي و از پيش تعريف شده (مگر در برنامه هاي هوش مصنوعي)

۲-محتواي اطلاعات از عوامل مهم در تعيين ماهيت برنامه نرم افزاري است

انواع نرم افزار ها:

۱- **نرم افزار سیستم:** سرويس دهي به ساير نرم افزار ها (مثل سيستم عامل)
۲- **نرم افزار زمان واقعي(بلادرنگ)(RealTime)** :نظارت ، تحليل و كنترل رويدادهاي جهان واقعي (صنعت و...)
۳-**نرم افزار تجاري:** پردازش اطلاعات تجاري و بازرگاني (امور مالي و حسابداري)
۴- **نرم افزار مهندسي و علمي** :شامل الگوريتم هاي پردازش گر عددي (نرم افزارهاي ستاره شناسي و...)
۵- **نرم افزار جاسازي شده**:جاسازي شده در دستگاههاي ROM دار (ماشين لباسشويي،ماكروفر و ...)
۶- **نرم افزار كامپيوتر شخصي** :نرم افزارهاي كاربردي (مثل :Word و ...)
۷- **نرم افزار مبتني بر وب:** طبق پروتكلهاي اينترنتي و توسط مرورگرها بازيابي مي شوند (سايت سازمان سنجش)
۸- **نرم افزار هوش مصنوعي يا AI** :حل مساله با الگوريتمهاي غيرعددي(شامل :تشخيص الگو/صدا/سيستم خبره و...)

انواع اشتباهات /تصورات و افسانه های نرم افزاری

۱. **افسانه هاي مديريتي**
- وجود كتاب استاندارد و مراحل كار / آيا استفاده مي شود؟
- خريد كامپيوترهاي جديد / استفاده از ابزارهاي مهندسي نرم مهمتر است
- اگر برنامه عقب باشد برنامه نويس اضافه مي كنيم / عقبتر مي افتيم

۲. **افسانه هاي مشتري**
- اهداف كلي براي شروع برنامه نويسي كافي است، جزييات بعدا/اطلاعات بايد دقيق باشد

۳. **افسانه هاي متخصص**
- زودتر كد را شروع كنيم كار تمام شود/زودتر كدنويسي شروع كنيد ديرتركار تمام مي شود
- وجود برنامه مشخص ساخت نرم افزار كافي است/برنامه مشخص عنصري از مراحل ساخت است

فصل ۲ – فرآیند

مقدمه

براي توليد محصول نياز به نقشه است مثلا توليد يك برد الكترونيكي

در توليد يك نقشه براي ايجاد يك محصول نرم افزاري چه كساني در ارتباط هستند؟

مديران – مهندسين - مشتريان

فرآيند نرم افزاري چيست؟ نقشه اي كه با پيمودن آن نرم افزار توليد مي شود.

دليل اهميت وجود فرايند در توليد نرم افزار؟

فراهم كننده پايداري – كنترل و سازماندهي فعاليت هاست

محصول يك فرايند نرم افزاري چيست؟ برنامه ها،اسناد و داده ها

نحوه اطمينان از صحت و درستي كار؟ توسط فرآيندها و مكانيزمای ارزيابی نرم افزار

مهندسي نرم افزار چيست؟

تعریف اول:استفاده از اصول ساده مهندسی به منظور رسیدن به یک نرم افزار مقرون به صرفه که قابل اطمینان بوده و روی دستگاههای واقعی کارآمد باشد.

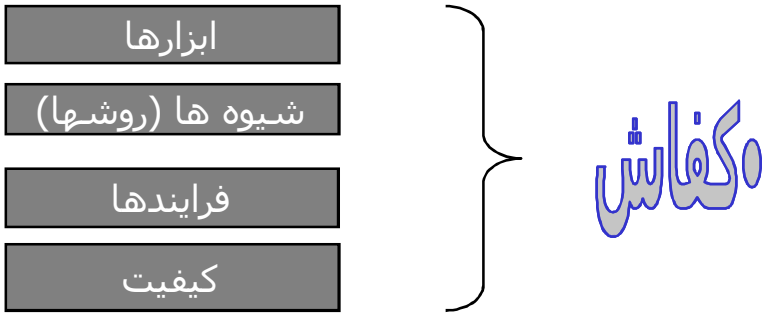
تعریف دوم(IEEE):

۱- بکارگیری یک روش سیستماتیک ،منظم کمیت پذیر برای توسع/اجرا و پشتيباني نرم افزار است یعنی بکارگیری مهندسی و شیوه های آن در نرم افزار .

۲- بررسی رهیافتهای آورده شده در قسمت(۱).

فرایندها ، شیوه ها و ابزارها

مهندسي نرم افزار يك فناوري لايه به لايه است و نقشه(فرآيند) همانند چسبي است كه لايه ها را به هم وصل مي كند و اساس مهندسي نرم افزار لايه فرآيند است



لايه هاي مهندسي نرم افزار

حوزه های اصلی پردازش (KPAs) (نقاط کلیدی)

مبنای کنترل مدیریت پروژه های نرم افزاری

بستری که در آن روشهای فنی بکار گرفته می شود،محصولات تولید می شوند

محکهایی بوجود می آیند،کیفیت سنجیده می شود و تغییرات احتمالی ترتیب داده می شوند

معرفی لایه روشها(شیوه ها): چگونگی انجام کار را از نظر فنی برای ساخت نرم افزار

معرفی لایه ابزارها: پشتيباني از فرآيندها و روش ها به صورت خودكار و نيمه خودكار توسط ابزارها

مهندسي نرم با كمك كامپيوتر(case) : استفاده ابزارها از اطلاعات ساير ابزارها

دیدگاه کلی از مهندسی در همه علوم : تحليل خواسته ها ، طراحی ، ساخت، تاييد و مديريت امور فني

دیدگاه کلی از مهندسی نرم افزار : مرحله تعريف ، مرحله توسعه ، مرحله پشتيبانی

فازهای مهندسی نرم افزار

مرحله تعریف(What)

شناسايي نوع اطلاعاتی را که باید پردازش شوند،عملکرد مطلوب چه چیز است؟،نوع وضعیت سیستم مورد انتظار ،نوع رابط هایی که باید ایجاد شوند.محدودیتهای موجود در طرح و معیارهای صحت را که برای تعریف یک سیستم موفق لازمند

مرحله توسعه (how)

چگونگی ساخته شدن داده ها،اجرای کار در ساختار نرم افزاری،چگونگی توصیف رابط ها چگونگی تبدیل طرح به زبان برنامه نویسی_یا زبان غیر رویه ای)و چگونگی انجام آزمون را توصیف کند.(طراحي نرم افزار+کد نویسي+آزمون)

مرحله پشتیباني (Change)

اصلاح خطا،ایجاد تطابقات لازم در حد لازم در رابطه با محیط نرم افزاری و تغییراتی ناشی از بهبود کار بخاطر تغییر نیاز مشتری

اصلاح(دید جدید) ، **تطابق**(تغییر شرایط محیط) ، **بهبود وضعیت** (افزایش امکانات)،**پیشگیری** (مهندسي مجدد و اصلاح آسانتر)

+فعالیتهاي چتري یا فعالیتهاي تکمیل کننده فازهاي مهندسي نرم افزار:

- کنترل پیگیری پروژه نرم افزاری
- تضمین کیفی نرم افزار
- مدیریت طرح نرم افزری
- آماده سازی اسناد و تولید
- مدیریت قابلیت استفاده مجدد
- بازنگری های فنی رسمی
- ارزیابی
- مدیریت خطر (ریسك ها)

فرآیند نرم افزاری

بعضی از کارها که هر کدام مجموعه ای از کارهای مهندسی نرم افزاری،معیارهای پروژه محصول کار و نقاط تضمین کیفیت هستند،باعث می شود که این مجموعه های معین فعالیتی در مشخصه های پروژه نرم افزری بکار گرفته شده و در مجموعه تقاضاهای تیم پروژه نیز باشند.

موسسه مهندسی نرم افزار (SEI)مدل جامعی ارائه داده که روی مجموعه ای از توانائیهای مهندسی نرم افزار که باید برای دسترسی سازمانها به سطوح مختلف رشد فرآیند وجود داشته باشد،پیش بینی شده است.

این طرح درجه بندی **میزان تطابق با توانایی مدل کامل (CMM) را مشخص** می کند که فعالیتهای اصلی لازم در سطوح مختلف فرآیند تکامل را تعریف می کند.

سطوح تکامل فرآیند (CMM)

- سطح ۱:اولیه:روش نامشخص و بستگي به تلاشهاي انفرادي
- سطح ۲:قابل تکرار: جهت تعیین هزینه و زمان کارایي
- سطح ۳:تعریف شده:استفاده از فرايند سازماني مستند و تایید شده
- سطح ۴:مدیریت شده.:اندازه گيري کامل فرايند نرم افزار و کیفیت محصول.
- سطح ۵:بهینه سازی:اصلاحات با بازخوردها و ایده هاي نو

هر سطح بالای سطح ۲ زیر مجموعه ای از سطح قبلی است.

سطح ۴ دربرگیرنده مفهوم اندازه گیری و استفاده از متریک است.

•متریک یک کمیت معنادار است که می توان آن را از روی بعضی از اسناد یا کدی در داخل پروژه نرم افزاری استخراج نمود.

سطح ۵ بالاترین سطح قابل دسترسی است .این سطح نمایانگر آنالوگ مکانیزمهای کنترل کیفی نرم افزار است که در مکانیزمهای دیگر و صنایع کامل تر وجود دارند.

سطوح تکامل فرآیند (CMM)

SEI دارای حوزه های اصلی در فرآیند است(KPA)
که هر کدام یک سطح رشد و تکامل دارند.KPAها آن گروه از فعالیت های مهندسی را توصیف می کنند که باید برای صورت گرفتن کار درست در سطح بخصوصی موجود داشته باشند.

هر KPAبا شناسایی مشخصه های زیر توصیف می شود:

اهداف – اهداف کلی که KPA باید به آن دست یابد .
تعهدات – نیازمندیهای که باید مرتفع شود (در روند انجام فرایند)
توانائیها – لوازم مورد نیاز براي رسیدن به تعهدات
فعالیتها - کارهای خاصی که لازمند تا به عملکرد KPA دست یابند
روشهای کنترل پیاده سازی – شیوه ای که با آن فعالیتها در جای خود قرار می گیرند کنترل می شوند.
روش هایی برای ارزیابی درستی پیاده سازی – شیوه ای که با آن عمل درست برای KPA مشخص می شوند

هجده KPA (هر کدام با استفاده از این خصوصیات توصیف شده اند) در طول فرآیند تکمیل کار تعریف شده و در سطح مختلف فرآیند تکاملی بصورت طرح درآمده اند . KPA زیر را در هر سطح تکمیلی فرآیند به دست می آوریم :

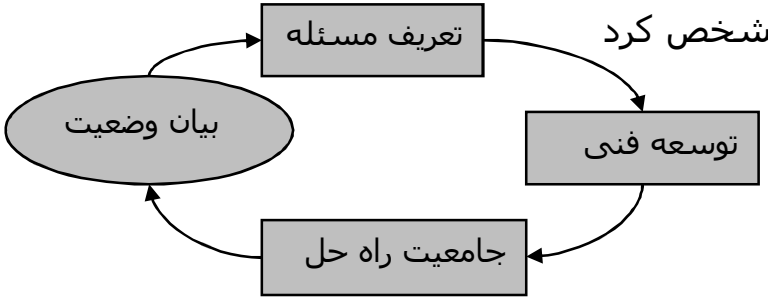
سطح ۲	مدیریت پیکربند ی نرم افزار . تضمین کیفیت نرم افزار . مدیریت پیمان کاریهای نرم افزار. پی گیری نظارت پروژه نرم افزاری . مدیریت نیازمندیها. برنامه ریزي پروژه	
سطح ۳		بازبینی های یکسان . همکاری درونی گروهی . مهندسی محصولی نرم افزاری . مدیریت یکپارچه نرم افزاری . برنامه آموزشی . تعریف فرآیند سازمانی . نقطه تمرکز فرآیند سازمانی
سطح ۴	مدیریت کیفیت نرم افزار. مدیریت فرآیند کمی.	
سطح ۵		مدیریت تغییر فرآیند مدیریت تغییر تکنولوژي . پیشگیری از نقیصه و اشکال

مدل فرآیند (Paradigm)

تيمي از مهندسين نرم افزار بايد استراتژي(راهبرد) ارائه دهد که شامل لايه های فرآيند (کفاش) و فازهاي عمومي (تعريف/توسعه/حمايت) اغلب اين راهبرد را مدل فرآيند يا پارادايم مهندسی نرم افزار می نامند

نکته:براي انتخاب مدل فرآيند براساس ماهيت پروژه و کاربردآن روشها و ابزار مورد استفاده و کنترل ها و موارد قابل ارائه که لازمند انتخاب مي شوند.

تمام کارهای توسعه وارائه یک نرم افزار را می توان بعنوان یک حلقه 4 عضوه مشخص کرد



وضعیت:وضعيت جاري پروژه

تعریف مساله:تعيين مساله اي که بايد حل شود

توسعه فني:حل و ساخت پروژه

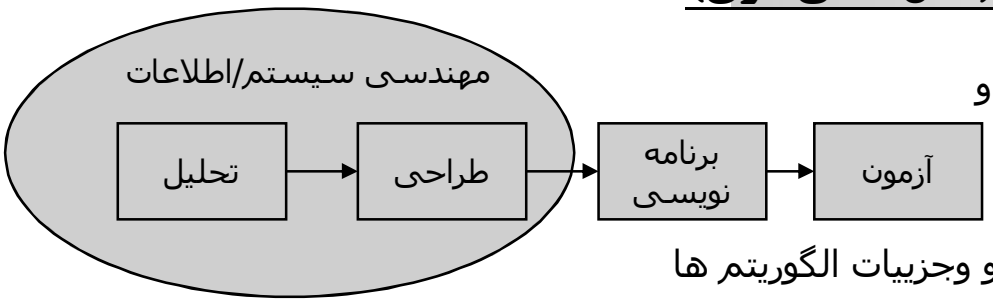
جامعیت راه حل:ارائه نتيجه به افراد رده بالا

غير از فاز بيان وضعيت ساير فازها مي توانند خود داراي مراحل چهارگانه باشند

انواع مدلهای فرآیند ارائه شده:

۱. مدل ترتیبی خطی
۲. مدل ایجاد نمونه اولیه (Prototyping Model)
۳. مدل RAD
۴. مدل افزایشی (Incremental Model)
۵. مدل حلزونی(Spiral Model)
۶. مدل حلزونی برنده - برنده (Win-Win)
۷. مدل توسعه همزمان
۸. مدل توسعه مبتنی بر مولفه (Component Base Development)
۹. مدل روشهای رسمی (Formal Method)
۱۰. تکنیکهای نسل چهارم

۱–مدل ترتیبی خطی(چرخه حیات کلاسیک)(مدل آبشاری) (مدل خطی سری)



مهندسي سيستم/اطلاعات:جمع آوري نيازها(سخت افزار/db) و امكان سنجي و تحليل و طراحي سطح بالا

تحليل:جمع آوري نيازهاي نرم افزاري

طراحي:تمركز طراحي بر ساختمان داده/معماري نرم افزار/رابطها و وجزييات الگوريتم ها

كد:برنامه نويسي و طراحي نرم افزار

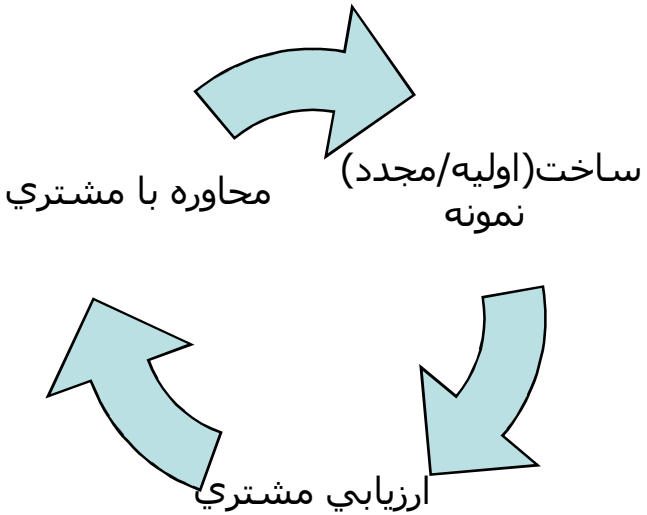
آزمون و پشتيباني:تاييد صحت برنامه / رفع خطاهاي احتمالي و تطابق با محيط

معایب روش ترتیبی خطی

- توسط پروژه ممكن است موارد گيج كننده اى براي اعضا رخ دهد
- برای مشتری مشکل است که تمام نیاز های خود را به طور مشخص و یکجا بیان کند
- رونمايي آخرين مرحله كار و انتظار مشتري و احتمال ريسك(اشتباهات)

۲–مدل نمونه سازی (نمونه اولیه)(Prototype)

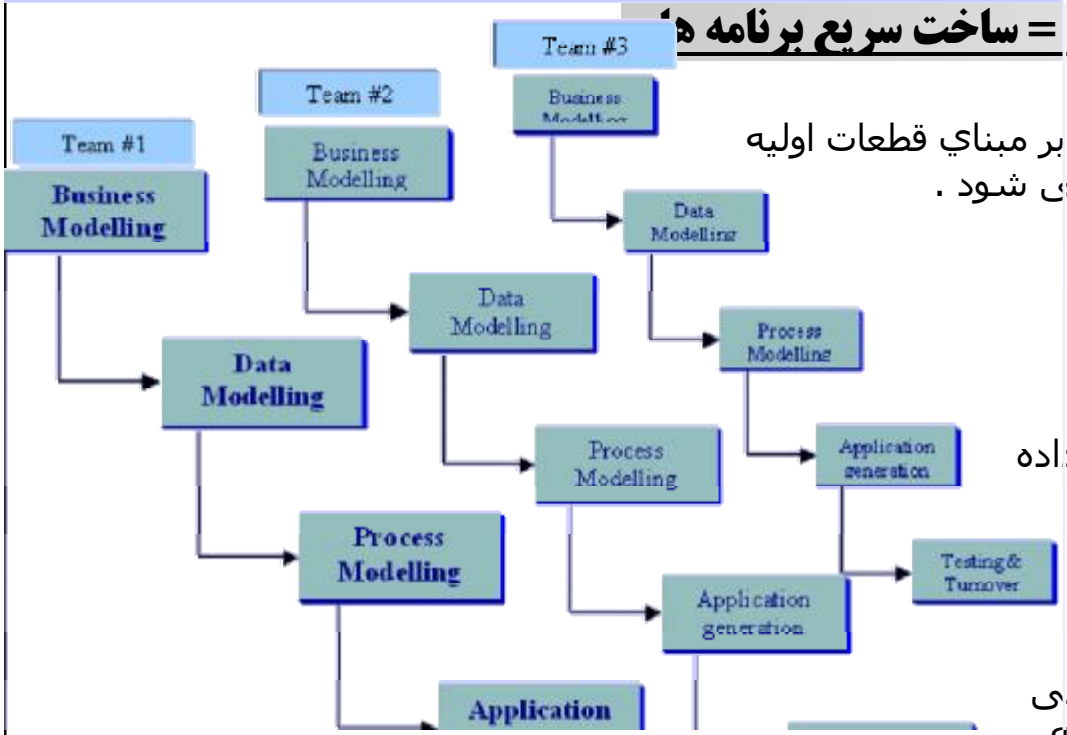
اگر اهداف كلي ذكر شود مناسب است



معایب روش نمونه اولیه

- عدم آگاهي مشتري از فوري بودن پروژه و عدم پيش بيني اهداف بلندمدت در آن
- به دليل سرعت امكان استفاده از الگوريتم ناكارآمد

۳– مدل RAD (Rapid Application Development) = ساخت سریع برنامه ه



- زماني كه تاكيد بر دوره كوتاه است اين روش مناسب است
- نوعي مدل خطي سرعت بالاست كه بر استفاده از ساخت بر مبناي قطعات اوليه استاساسا برای کارهای سیستمهای اطلاعاتی استفاده می شود .

فازهاي مدل RAD

مدلسازی تجاری

مدلسازي جريان اطلاعات بين توابع تجاري

مدلسازی داده ای

تبديل جريان اطلاعات فاز مدلسازي تجاري به مجموعه اشياي داده

مدلسازی فرآیند

تبديل اشيا داده اي، طوريكه جريان اطلاعات لازم براي پياده سازي توابع تجاري حاصل شود

ایجاد کاربرد

به جای تولید نرم افزار با استفاده از زبانهای متعارف برنامه نویسی متعارف نسل سوم فرآیندRAD تلاش می کند تا از جزء های برنامه موجود مجددا استفاده کرده یا جزء های قابل استفاده مجددی را تولید کند. (نسل چهارم)

آزمون و گردش(خاتمه)

به دليل آزموده شدن قبلي ديگر دوباره آزموده نمي شوند.

مدلهای فرآیند تکامل نرم افزار(تکاملی یعنی خروجی یک سیکل ورودی سیکل بعدی)

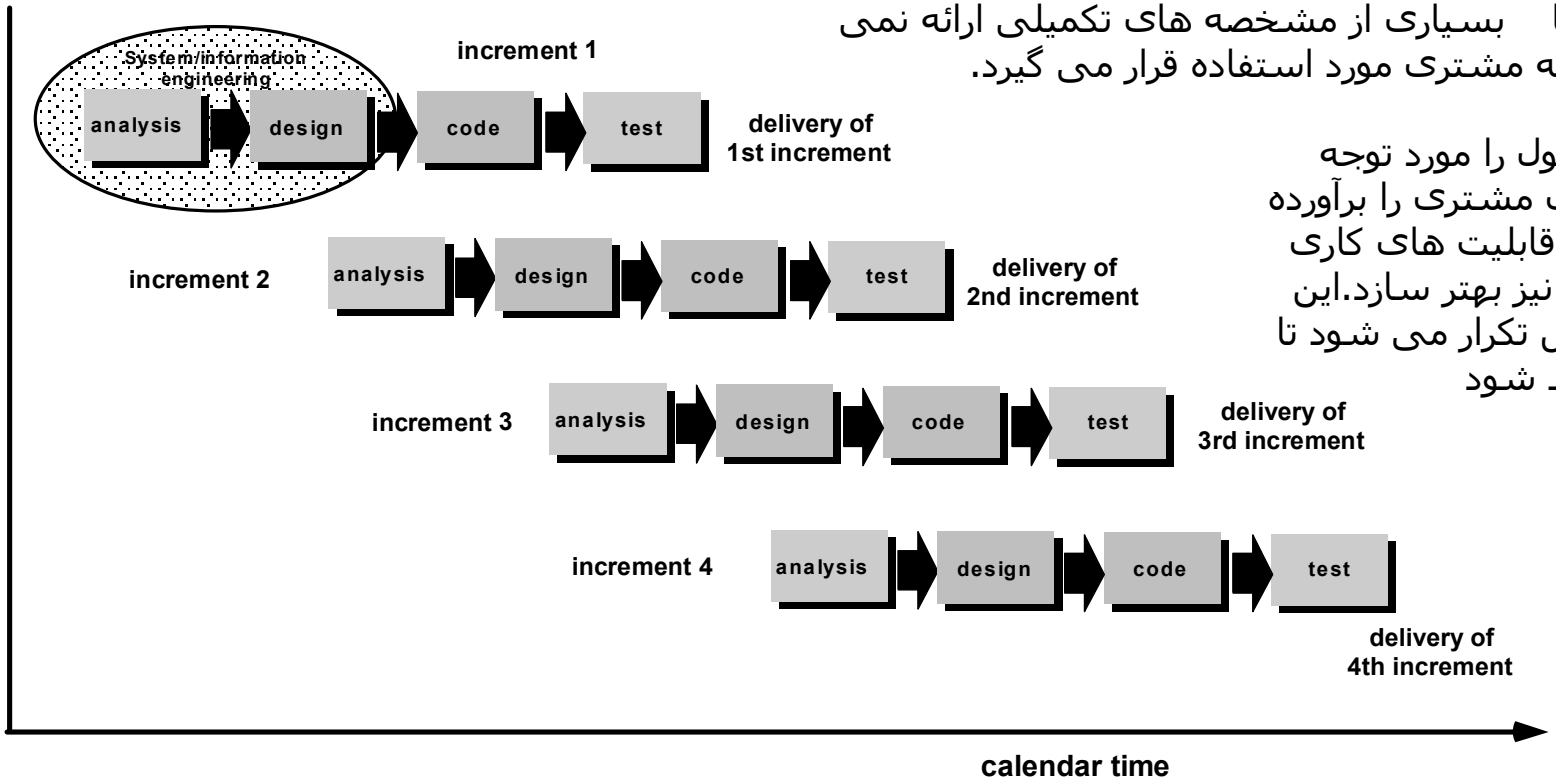
در این مدلها تکرار اصل مهم تکامل است

۴–مدل افزایشی

همان مدل خطي با ديدگاه تكرار است و برخلاف نمونه سازي، خروجي هر مرحله دور ريخته نمي شود و براي تكامل در چرخه باقي ميماند

اولین تکرار سبب ایجاد هسته اصلي محصول مي شود یعنی نیازهای اولیه مورد توجه قرار می گیرند اما بسیاری از مشخصه های تکمیلی ارائه نمی گردند.هسته محصول بوسیله مشتری مورد استفاده قرار می گیرد.

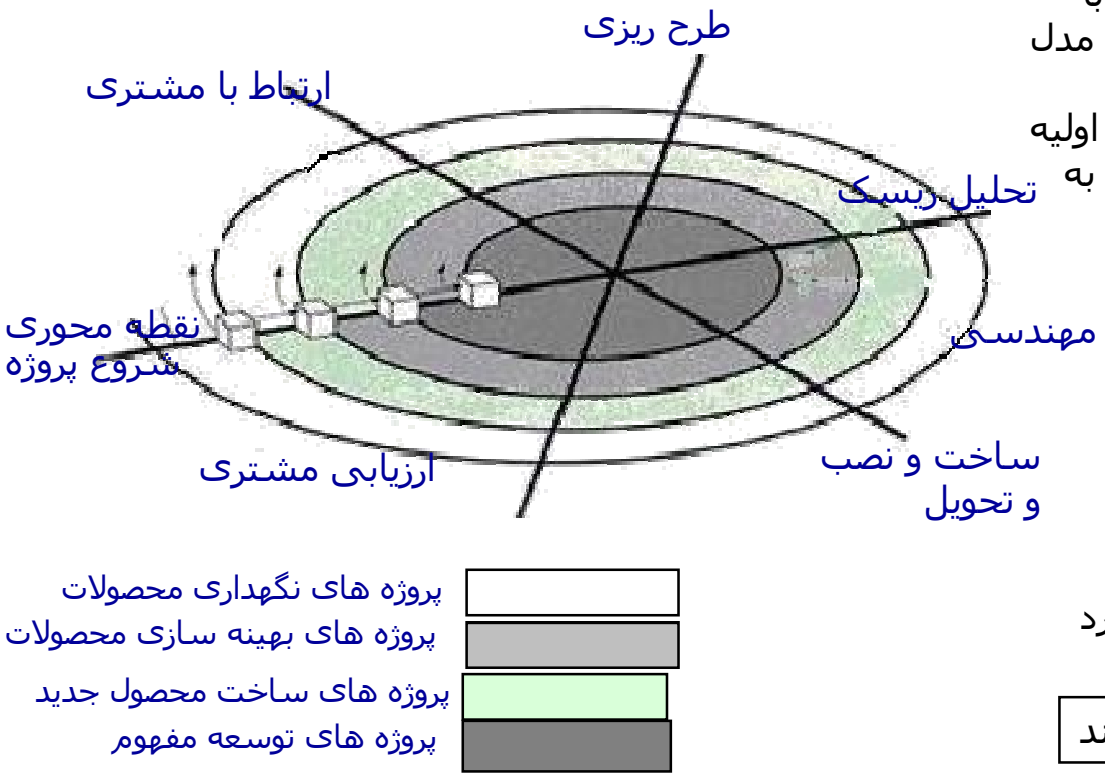
این طرح اصلاح هسته محصول را مورد توجه قرار می دهد تا بهتر نیازهای مشتری را برآورده نموده و ارائه مشخصه ها و قابلیت های کاری اضافی مورد نظر مشتری را نیز بهتر سازد.این فرآیند بدنبال تحویل هر بخش تکرار می شود تا وقتی که محصول کامل تولید شود



زماني مفيد است كه حمايت لازم براي پياده سازي كامل وجود ندارد

۵– مدل مارپیچی Spiral

- توسط بوهم پیشنهاد شد ، ماهیت تکرار نمونه سازي با محاوره مشتري را با جنبه هاي كنترلي و سيستماتيك مدل خطي تركيب نموده است
- پروژه در چندين مرحله ساخته مي شود ،شايد مراحل اوليه تنها روي كاغذ يا الگوي اوليه مهندسي باشد و سپس به ايجاد/ارتقا و نگهداري محصول برسيم
- يك دور مارپيچ شامل ۳تا۶ فاز كاري است



- ارتباط با مشتري:بررسي نيازها
- برنامه ريزي (Planing):تعريف منابع و زمانبندي
- تحليل خطر:مشخص كردن ريسكها و خطرات
- مهندسي:طراحي و ايجاد برنامه
- ساخت و ارائه:ساخت/آزمون/آموزش/نصب
- ارزيابي مشتري:بدست آوردن واكنش مشتري و بازخورد

تكرار فراوان از بروز خطا جلوگيري مي كند

معایب روش مارپیچی

- قانع كردن مشتري دردسرساز است
- ممكن است اطلاعات نادرست به ما بدهند

۶– مدل مارپیچی Win-Win

- به دليل امكان قرار نگرفتن اطلاعات كافي در ارتباطات با مشتري ، دنبال ارتباط با افراد حائز شرايط برنده مي رويم

تفاوت مارپيچي با مارپيچي Win-win :

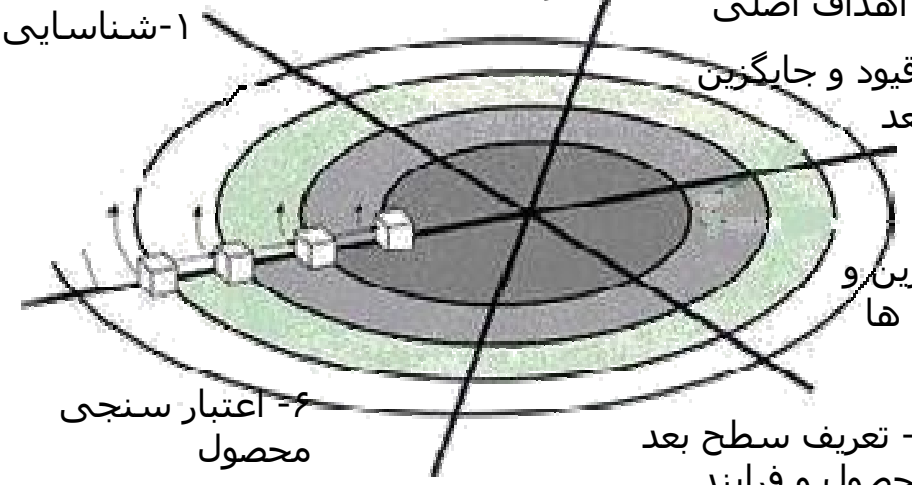
در مدل حلزونی بجای یک رابطه ارتباط مشتری ، فعالیت های زیر تعریف می شوند:
۱-شناسایی سیستم یا دارندگان اصلی سیستم های فرعی
۲-تعیین شرایط برد دارندگان سهم.
۳-مذاکره در مورد شرایط برد سهامدار برای تطابق دادن آنها درمجموعه ای از شرایط win-win برای همه افراد درگیر.

نقاط اطمينان(تكيه):

معياري براي اندازه گيري پيشرفت

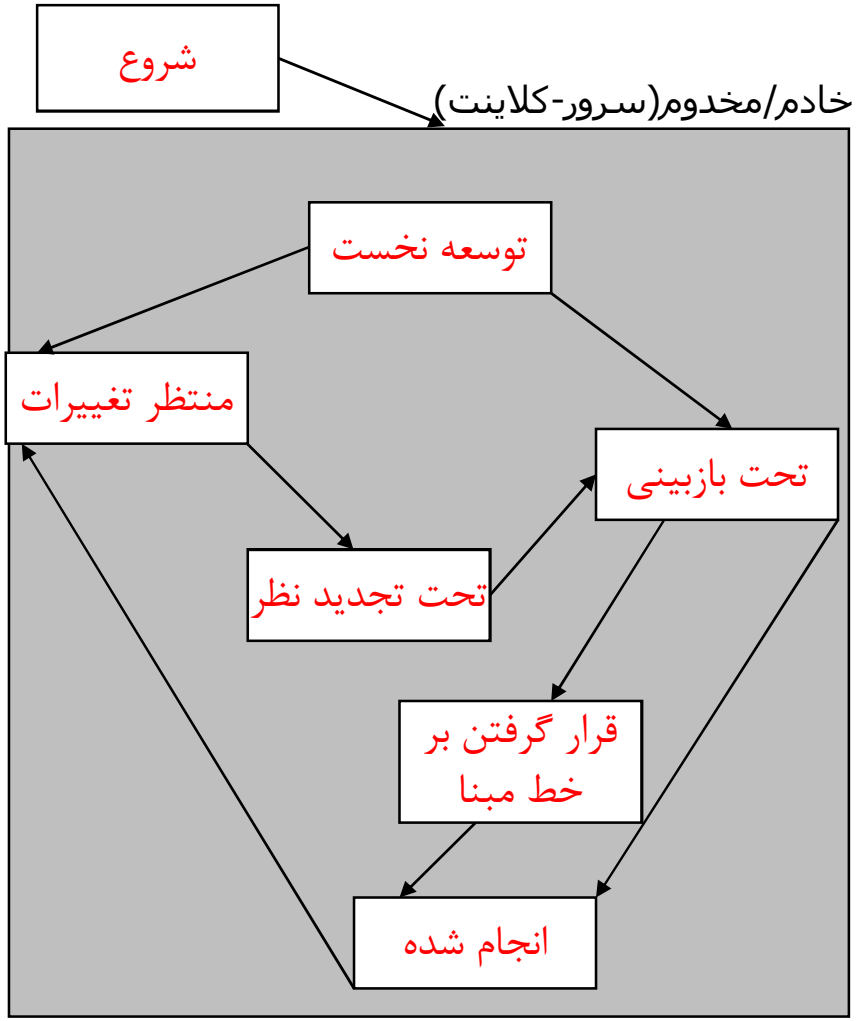
ديدگاههاي نقاط تكيه(معيار پيشرفت):

اهداف دوره زندگي (lco) :اهداف فعاليتها
معماري دوره زندگي (lca) :اهداف سيستم
توانايي عمليات اوليه(loc):آماده سازي پياده سازي



۷– مدل توسعه همروند (همزمان)(مهندسی همزمانی)

- براي دنبال كردن وضعيت يك پروژه
- تمام فعاليتها به صورت همزمان نمايش داده مي شوند و حالتهاي فراواني ايجاد مي شود كه از يك حالت به حالت ديگر با رخ دادن يك رويداد مي رويم
- تقسيم نيروها و حركت موازي
- مدل توسعه همروند ابزاري مناسب براي توسعه و ايجاد برنامه هاي خادم/مخدوم(سرور-كلاينت)
- ابعاد سيستمهاي كلاينت/سرور
  ۱. **بعد سيستم:**ساخت/مونتاژ و استفاده
  ۲. **بعد مولفه:**طراحي /بررسي



نحوه همزماني در توسعه همروند:

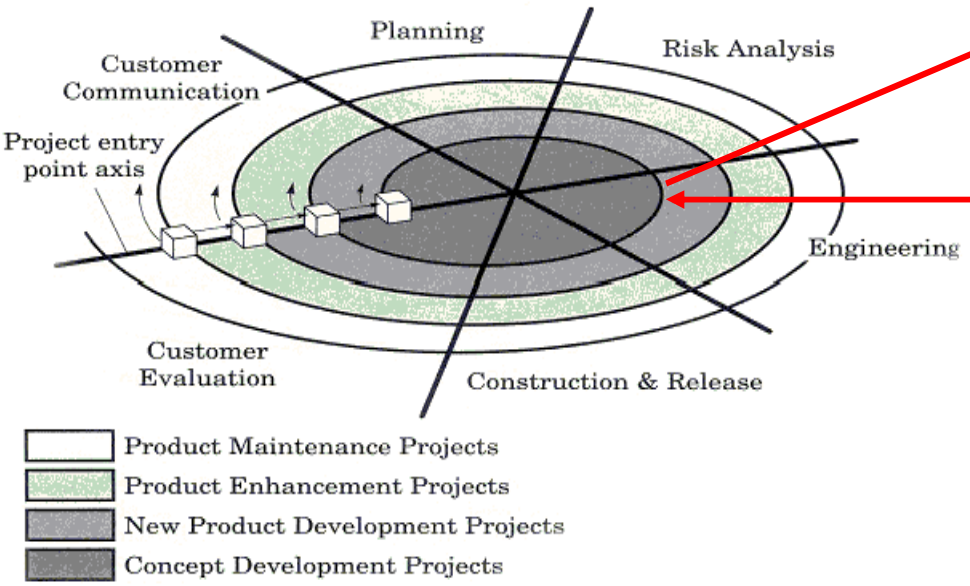
۱-همزماني فعاليتهاي سيستم و مولفه

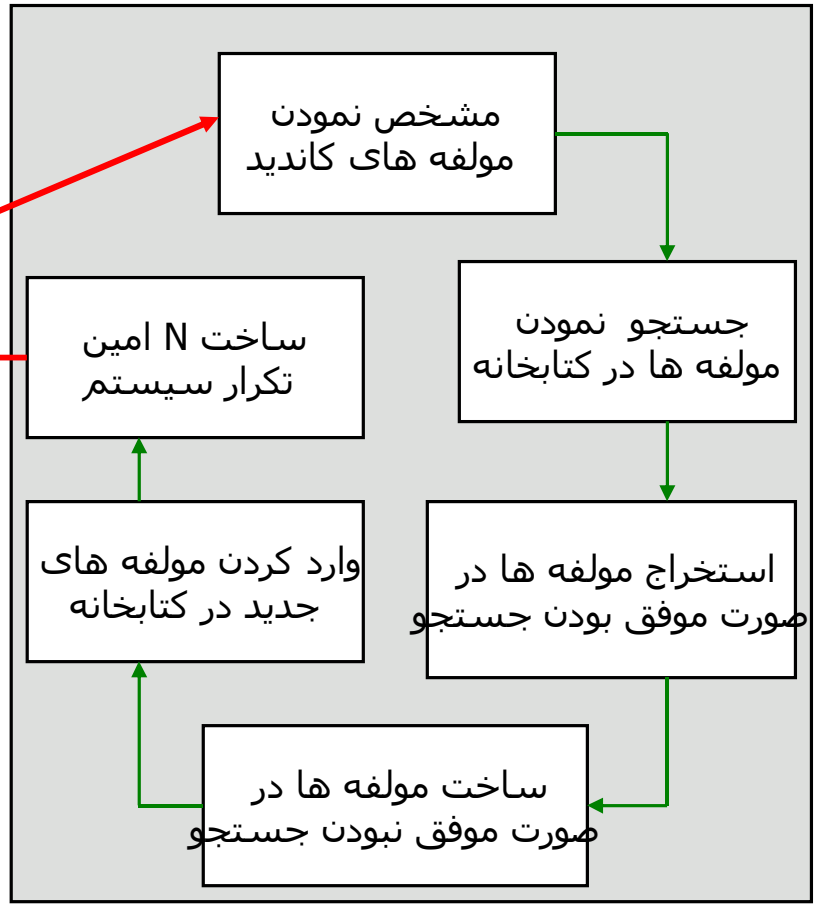
۲-طراحي همزمان اجزاي موجود در برنامه كلاينت / سرور

۸–توسعه مبتنی بر اجزا (مولفه)(CBD) Component Base Development

- داراي ماهيت تكاملي است و بسياري از خصوصيات مدل حلزوني را دارد
- در كنار ويژگيهاي مدل حلزوني از تكنولوزي شي گرايي و object نيز استفاده مي كند
- به سبب تاكيد بر استفاده مجدد و اشيا،كاهش هزينه و افزايش سرعت را به دنبال دارد

UML زبان مدلسازي است كه نماينده اين مدل در صنعت محسوب مي شود





۹– مدل روشهای رسمی

- تعیین رسمي مشخصه هاي نرم افزار با استفاده از روابط رياضي
- ابهام کمتری نسبت به روشهای غیر قراردادی دارد
- وجود نرم افزاري بدون مشكل را تضمين مي كند

مثل مهندسي نرم افزار اتاق پاك
طراحي سيستم با مدلهاي آماري قبل از
نوشتن كدهاي برنامه و استفاده از مدل
Incremental به صورت زير سيستمهاي
سيستم

معایب روشهای رسمی

- زمان و هزینه بالا
- آموزش گسترده جهت مهندسین نرم افزار
- دشواری بکارگیری مدلها در تعامل با مشتریانی که فاقد دید فنی می

۱۰– تکنیکها و فنون نسل چهارم (4GT)

- محور اصلي اين روش،استفاده از ابزارهاي مهندسي نرم افزار است
- ابتدا يك مشخصه از نرم افزار را در سطح بالا(نماد و مدل) طراحي و سپس كد توليد مي شود

خصوصيات محيطهاي توسعه نرم افزار نسل چهارم:

۱-زبان غير رويه اي(4GL) ۲-ويرايش داده ها ۳-فرمها و ارتباطات متقابل ۴-گرافيك بالا
۵-امكان توليد كد وبرنامه ۶-توانايي توليد صفحه گسترده و HTML

ادعاي موافقين وضعيت فعلي روشهاي 4GT

- استفاده از 4GT به همراه ابزارهاي مهندسي نرم،راه مناسب براي بسياري از مسائل نرم افزاري
- كاهش زمانبندي توليد نرم افزار و ميزان تحليل پروژه هاي كوچك
- نياز به تحليل/طراحي و آزمايش بيشتر براي پروژه هاي بزرگ

ادعاي مخالفين 4GT

خيلي ساده تر از زبانهاي برنامه نويسي نيست-كد توليد شده چندان كارآمد نيست-امكان توليد سيستم بزرگ نيست

رابطه محصول و فرآیند

اگر فرايند ضعيف باشد محصول نهايي بدون شك با مشكل مواجه مي شود

بخش دوم– مدیریت پروژه های نرم افزاری

فصل سوم – مفاهیم مدیریت پروژه

مقدمه

•تعریف مدیریت پروژه (Pressman) :

مدیریت پروژه شامل برنامه ریزی ، نظارت وکنترل افراد ، فرایند و رخدادهایي می باشد که در طول تکامل نرم افزار از مفهومی اولیه به یک پیاده سازی عملیاتی منجر می گردد.

•سطوح مدیریت

مهندس نرم افزار : فعالیت های روزمره طرح ریزی و کنترل کار های فنی خود را اداره می کند.
مدیران ارشد : ارتباط بین امور و متخصین نرم افزار را فراهم می کند.
مدیران پرو ژه: گروه یا تیم مهندسین نرم افزار را کنترل وطرح ریزی میکند
و.....

•دلیل اهمیت مدیریت پروژه

ایجاد نرم افزار کامپیوتر فعالیتی پیچیده است ، بخصوص اگر افراد متعددی برای زمان طولانی در آن کار نمایند ، دلیل اصلی نیاز به مدیریت پروژه این موضوع است.

•محصول مدیریت پروژه؟

طرح پروژه با آغاز و شروع فعالیت های مدیریت تهیه و تولید می شود . این طرح رونـد و کارها یی که باید انجام شود ، افرادی که کارها را انجام می دهند و مکانیزم ارزیابی خطرات ، کنترل تغییر و ارزیابی کیفیت را تشریح می کند.

•اطمینان از صحت مدیریت

هرگز به طور قطعی نمی توانید اطمینان حاصل کنید که طرح پروژه درست است مگرانکه محصول یا کیفیت را در زمان مقرر و با بودجه تعین شده ارائه کنید.

مولفه های مدیریت پروژه نرم افزاری(4p)

- افراد People
- فرایند Process
- محصول Product
- پروژه Project

مولفه های مدیریت پروژه نرم افزاری(4p)

۱– افراد

پرورش پرسنل نرم افزار کاملا ماهر و با انگیزه در واقع فاکتور انسانی { افراد یا پرسنل} آنقدر مهم است که مو سسه مهندسی نرم افزا ر یك مدل تکامل توانایي مدیریت نیروي انساني به نام PM-CMM را ارائه داده است.

مدل رشد مدیریت افراد(PM-CMM) عملکرد های کلیدي :استخدام ، مدیریت عملکرد، جبران اموزش و پیشرفت شغلی و طراحی سازمان و توسعه تیمي را تعریف مي کند

۲–محصول

قبل ازطرح ریزی پروژه باید **اهداف و حیطه محصول** مشخص شده ومحدودیت های فنی و مدیریتی تشریح شوند. بدون این اطلاعات نمی توان برآوردهای معقولی ازهزینه ، ارزیابی موثر خد مات ، تجزیه و تحلیل کارها و وظا یف پروژه یا برنا مه زمان بندی قابل اجرا پروژه همراه با نشانه های درست پیشرفت را ارا ئه کرده است. در بسیاری ازموارد **این فعا لیت بخشی از مهندسی سیستم**. یا مهندسی روند کارشروع میشود و به عنوان مرحله اول در تحلیل نیازمند ی های نرم افزار ادا مه پیدا می کند

۳–فرآیند

پروسه چهارچوبی را ارائه می کند که به کمک ان می توان **طرح جامع توسعه نرم افزار** را **تهیه** کرد . مجموعه های مختلف کاری – وظایف ،مراحل بر جسته کاری ، محصولات کاری و مقاطع تضمین کیفیت – به فعا لیت های اصلی این امکان را می دهد تا در مشخصه های پروژه نرم افزار و نیاز مندی های تیم پروژه به کار گرفته شوند.در نها یت فعا لیت های حفاظتی مثل تضمین کیفیت نرم افزار،مدیریت پیکر بندی نرم افزار و اندازه گیری و مدل فرایند حاکم هستند

۴– پروژه

ما پروژه های نرم افزاری کنترل و طرح ریزی شده را به یک **دلیل اصلی** انجام می دهیم زیرا **تنها راه شناخته شده برای اداره کردن و کنترل پیچیدگی است** و هنوزهم با آن در کشمکش و جدال هستیم . نرخ موفقیت پروژه های نرم افزاری تا حدی بهبود پیدا کرده است اما نرخ شکست و عدم موفقیت پروژه هم چنان بالا ترازحدی است که باید باشد . برای پرهیزازشکست پروژه ، مدیر پروژه نرم افزار و مهندسین نرم افزاری که محصول را می سازند باید از علائم اخطار دهنده پرهیز کنند ،عوامل موفقیت را که به مدیریت پروژه خوب می انجامد درک کنند و روش مشترک و معقولی برای طرح ریزی ، نظارت و کنترل پروژه ارائه کنند.

افراد

اهمیت افراد

در تحقیق انتشار یا فته IEEE از ۳ شرکت فنی بزرگ خواسته شد تا مهم ترین عامل دخیل در پروژه نرم افزار موفق را ذکر کنند .
پاسخ آنها به شرح زیر است :
۱- آن چیزابزار مورد استفاده ما نیست بلکه افراد یا پرسنل است .
۲- مهمترین عاملی که باعث موفقیت این پروژه شد برخورداری از پرسنل با هوش بود...دیگر عوامل اهمیت کمتری دارند.
۳- تنها قاعده ای که در مدیریت از آن استفاده می کنیم تضمین در اختیار داشتن افراد خوب و ایجاد محیطی است که در ان بتوان افراد خوبی درست کرد.
مدیران میگویند که پرسنل عامل اصلی هستند اما گاه کردار و عملکرد آنها با گفتارشان یکی نیست .

انواع گروههاي افراد در فرايند ساخت نرم افزار

۱- **مدیران ارشد** که موضوعات و مسائل کاری را مشخص میکنند که اغلب تاثیر زیادی بر پروژه دارند(تنظیم قرارداد-....).
۲- **مدیران پروژه (فني)**که باید افرادی را که کار نرم فزارانجام میدهند برنامه ریزي،تخصیص وظایف،سازماندهي و کنترل کنند.
۳- **مجریان(متخصصین):** افرادی که مهارت های فردی مورد نیاز جهت مهندسی محصول یا کاربرد ارائه میکنند.
۴- **مشتریان** کسانی که نیازمندی های نرم افزار مورد نیاز را مشخص میکنند .
۵- **کاربران نهایی** که با عرضه نرم افزار جهت استفا ده از آن بهره میگیرند.
تیم پروژه برای کا رآمد بودن با ید به نحوی سازماندهی شود که مهارت ها و قا بلیت ها ی هر فرد را به بیشترین حد ممکن برساند واین وظیفه رهبر تیم است.

* رهبران تیم

مدیریت پروژه یا رهبر تیم فعالیتی در تعا مل با افراد است و به این دلیل مجریان و متخصصین ذیربط ، اغلب رهبران تیم خوبی نیستند .آنان از مجموعه مهارت های فردی درست و لازم بر خوردار نیستند .

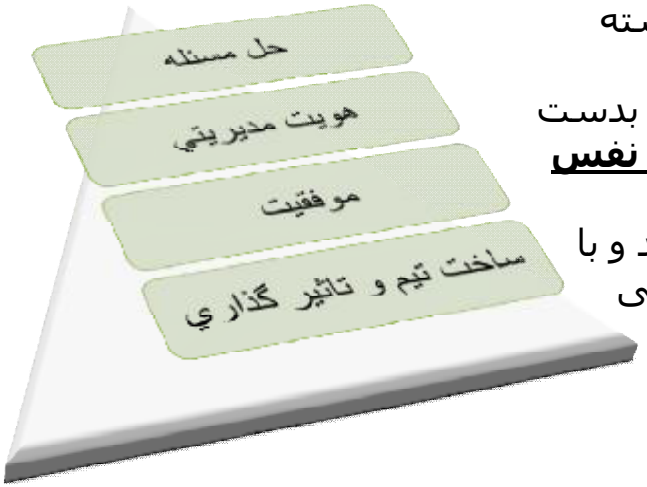
مد ل MOI (برای رهبری) :

انگیزه : قابلیت ترغیب پرسنل فنی در ایجاد بهترین قابلیت ها.
سازماندهی : توانایی ایجاد و برقراری پروسه های موجود که امکان تبدیل مفهوم اولیه به محصول نهایی را فراهم می کند.
ایده ها یا ابداع : توانایی ترغیب افراد به خلاقیت به هنگام کاردرمحدوده تعیین شده برای محصول .

نام درس:مهندسی نرم افزار ۱

چهارویژگي مديرپروژه كارآمد

حلهو موسا



۱. **حل مسئله** : مدیر پروژه نرم افزار کارامد میتواند مسائل **فنی** و سازما نی شایع را تشخیص دهد وراه حل اصولی را سازماندهی ومطالب اندوخته شده ازپروژه های قبلی را در موقعیت های جدید اعما ل کندو در صورت بی ثمر بودن راه حل های اولیه برای تغییر جهت انعطاف داشته باشد.

۲. **هویت مدیریتی** : مدیر پروژه خوب با ید مسئولیت پروژه را بر عهده بگیرد:او با ید برای بدست گرفتن کنترل در زمان مقتضی و مجاز کردن پرسنل فنی به پیگیری غرایز خود ، از **اتکا به نفس** خوبی برخوردار باشد.

۳. **موفقیت**:برای بهینه کردن بهروری تیم پروژه مدیر باید موفقیت و نواوری را **تشویق** کند و با اعمال خود نشان دهد که اگر افرادی به طور کنترل شده رسیک کنندبا **مجازات** روبرو نمی شوند.

۴. **ساخت تیم و تاثیر گذاری**:یک مدیرموفق وموثر با ید مردم شناس باشد وتحت شرایط فشاربتواند اوضاع را تحت کنترل در اورد.

تیم نرم افزار

تعداد ساختارهای سازما نی انسا نی برای توسعه نرم افزار، به اندازه تعداد سازمانهای توسعه دهنده نرم افزاری می باشد . ساختار سازمانی نمی تواند براحتی تغییر داده شود . نتایج عملی و سیاسی تغییرسازما نی در حیطه مسئولیت های مدیر پروژه نرم افزاری نمی باشد . اما تشکیلات مردمی که مستقیما در پروژه نرم افزاری دخیل هستند درحیطه کاری مدیر پروژه قراردارد.

بهترين ساختار تيمي به تعداد افراد تيم – سطح مهارتهاي آنها و ميزان پيچيدگي مساله بستگي دارد

انواع سازماندهي هاي ارائه شده

۱. **غير متمركز دموكراتيك (DD)**
۲. **غيرمتمركز كنترل شده(CD)**
۳. **متمركز كنترل شده (CC)**

نام درس:مهندسی نرم افزار ۱

انواع سازماندهي هاي ارائه شده جنبه رياست(دموكراتيك+كنترل شده) +مشورت گروهي براي حل مشكل(متمركز+غيرمتمركز)

۱.سازما ن غیر متمرکز دموکراتیک(DD) :

این تیم مهندسی نرم افزاری دارای یک رئیس دائمی نیست(موقتي). بلکه **هماهنگ کننده های** کار برای دوره کوتاهی در این سمت منصوب میشود و سپس افراد دیگری که ممکن است کارهای مختلفی را هماهنگ کنند جایگزین این هماهنگ کننده ها میشوند.و براي حل مسائل تصميمات به صورت توافق گروهي است و ارتباط اعضا به صورت افقي است

۲.سازمان غیر متمرکز کنترل شده(CD) :

این تیم مهندسی نرم افزاری دارای یک **رئیس ثابت** میباشد که کارهای خاص ومدیران ثا نوی که مسئولیت کارهای فرعی را برعهده دارند، هما هنگ و مشخص میکند اما **حل مشکلات به صورت یک کار گروهی** باقی می ما ند اما **اجرای** راه حلهای مشخص شده برای مشکل توسط رئیس تیم در میان گروه های مختلف تقسیم می شود.ارتباطات موجود میان گروههای فرعی و افراد به صورت هم سطح(افقي) می باشد.همچنین ارتباط عمودی در امتداد سلسله مراتب کنترل به وقوع می پیوندد .

۳.سازمان متمر کز کنترل شده(CC):

حل مشکلات در سطوح بالا و ایجاد هماهنگی داخلی در میان تیم تحت نظارت و کنترل رئیس و اعضا تیم به صورت عمودی باشد.

هفت فاکتور مانتي که باید در هنگام برنامه ریزی ساختار تیم های مهندسی نرم افزاری در نظر گرفته مشود:

۱- سختی مشکلاتی که باید مورد حل و فصل قرار گیرند. cc براي آسان و dd براي مشكل
۲- اندازه برنامه حاصله در تعداد خطوط و جملات برنامه کامپیوتری یا امتیازات کارکردی(درجه ارتباطات) درجه بالا:dd
۳- مدت اشتغال تیم(مدت زماني كه اعضاي تيم در كنارهم هستند) ⟵ در DDمهم است
۴- تا چه اندازه میتوان مشکلات را به صورت پیمانه ای درآورد. ⟵ در DD هرچه کمتر بهتر
۵- میزان اعتبارو کیفیت مورد نیاز که در ارتباط با سیستم باید ایجاد شود. بدون خط(پیمانه زیاد هرکس کار خودش):cc/cd
۶- ثبات تاریخ تحویل(میزان اهمیت تاریخ تحویل)
۷- میزان جامعه پذیری مورد نیاز برای پروژه(درجه ارتباط لازم با بیرون)

از آنجا یی که ساختار متمرکز کارها را سریعتر تکمیل میکند، بنابراین این ساختاردرحل مشکلات ساده مفیدترین ساختارمی باشد . تیم های غیرمتمرکز نسبت به افراد ، راه حلهای بهترو بیشتری را بوجود می آورند . بنا براین احتمال موفقیت چنین تیم ها یی هنگام بررسی مسا ئل بسیار سخت بیشتر است .

مثال روزنامه

چهار الگوي سازمانی توسط Constantine جهت همكاري تیمهای مهندسی نرم افزار پیشنهاد گردیده است :

۱. **الگوی بسته :** این نمونه ، تیم مورد نظر را با توجه به سلسله مراتب سنتی قدرت و اختیارات تشکیل می دهد.چنین تیمهایی هنگام تولید نرم افزار که کاملا مشابه با تلاش های قبلی است به خوبی عمل می کنند، اما احتمال خلاق بودن چنین تیم هایی هنگام کار کردن در یک نمونه بسته بسیار کم می باشد.(همه راه ها از بالا)(اگر پروژه تكراري باشد مناسب است)

۲. **الگوی تصادفی :** تیم را به صورت تصادفی و آزادانه تشکیل میدهد و به ابتکار فردی اعضاء تیم بستگی دارد. هنگامی که تحول ابداعات یا فناوری ها مورد نیاز باشد، تیمی که دارای نمونه تصادفی است به خوبی عمل خواهد کرد. اما چنین تیمی در صورتی که "عملكرد منظم و مرتب " مورد نیاز باشد با مشکل مواجه خواهد شد.(زمان تغییر تكنولوژي مناسب است)

۳.**الگوی باز :** این نمونه سعی دارد تیمی را به روشی تشکیل دهد که این تیم دارای بعضی از کنترل های مربوطه نمونه بسته باشد و در عین حال بیشتر ابداعات پدید آمده در صورت استفاده از نمونه تصادفی را نیز دارا باشد. کار در این نمونه به صورت دسته جمعی و همراه با ارتباط بیشتر و **توافق آرای مبتنی بر تصمیم گیری** و علائم تجاری تیم های نمونه باز انجام می شود. ساختارهای تیم نمونه باز برای یافتن راه حل برای مشکلات پیچیده بسیار مناسب هستند.اما برای حل این مشکلات به اندازه سایر تیم ها **سودمند نمی باشند**.

۴. **الگوی همگام :** این نمونه بر تشابه طبیعی مشکل(؟؟؟) و **تجزیه مساله** متکی می باشد و اعضای تیم را برای کار کردن بر روی بخش هایی از مشکل سازماندهی می کند و در طول این کار ارتباط فعال میان اعضاء تیم بسیار کم می باشد.

روش دیگر مشهور "بي نظمي ابداعي "بود که کنستانتین ارائه دادکه با اعمال تغییرات در دمکراتیك غیر متمرکز بود و کنترل هرج و مرج در آن مهم است

فاكتورهاي ليستر كه " موجب پیشرفت مسمومیت در محیط تیم می شوند. " : **مثال فوتبال**

۱. یک محیط كاري آشفته (یا خیلي شاد)که در آن اعضاء تیم انرژی را اتلاف می کنند و بر اهداف لازم الاجرا کار توجه و تاکیدی ندارند.

۲. نا امیدی شدید که به وسیله پرسنل ، کار یا فناوری ایجاد می شود و این ناامیدی میان اعضاء تیم اختلاف ایجاد می کند.

۳. شیوه های تفکیک شده یا ناقص هماهنگی یا تعریف ناقص یا انتخاب نادرست ، مانعی برای تکمیل کار خواهد بود.

۴. تعریف نامشخص نقش کارکنان موجب از بین رفتن حس مسئولیت می شود.

۵. تعریف قرار گیری در معرض شکست مکرر وپیوسته که موجب از بین رفتن اعتماد به نفس و تنزل مسائل اخلاقی می شود.

نام درس:مهندسی نرم افزار ۱

راههاي جلوگيري از انتخابهاي نادرست در روند نرم افزاري

۱.مطمئن شدن از این امر که مشخصات نرم افزار مورد ساخت مطابق با سختی و عدم انعطاف روند انتخاب شده می باشد یا نه.

۲. اجازه دادن به تیم برای انتخاب روند با شناخت کامل از این مسئله که تیم ضمن انتخاب ، مسئول آن است که یک محصول با کیفیت را ارائه کند.

• هر یک از تیم های نرم افزاری با شکست های کوچکی مواجه می شوند .**کلید اصلی برای اجتناب از جو شکست** . نا کا می آن است که فنونی بر اساس تیم برای فید بک(بازگشت) و حل مشکل ایجاد شود. با این وجود شناخت تفاوت های انسا نی اولین گام در راه ایجاد تیم همبسته می باشد.

مثال XML

هماهنگي و ارتباط

قابلیت استفاده داخلی به صورت یکی از مشخصه های اصلی بسیاری از سیستم ها در آمده است.نرم افزار جدید باید با نرم افزار موجود ارتباط برقرار کند و با محدودیت های از پیش تعیین شده ای که از سویی سیستم یا محصول تحمیل می شود ، مطابقت داشته باشد.

این خصیصه های نرم افزاری مدرن ، یعنی مقیاس ، عدم اطمینان و قابلیت استفاده داخلی حقایق زندگی هستند.برای کنترل موثر این خصیصه ها ، تیم مهندسی نرم افزاری باید شیوه های مناسب و موثری را برای هماهنگ کردن افرادی که کار را با هم انجام می دهند ، ایجاد کند. برای تکمیل هماهنگی ، مکانیسم های لازم برای روابط رسمی و غیر رسمی میان اعضاء تیم و روابط میان چند تیم مختلف باید مشخصشوند. روابط رسمی از طریق نوشتن نامه ، ملاقات رسمی و تقریبا از طریق سایر کانالهای ارتباطی غیر شخصی به وجود می آید. روابط غیر رسمی شخصی تر است.

روشهاي ايجاد هماهنگي

- رهیافت رسمی و غیررسمی
- شیوه های رسمی و میان فردی
- شیوه های غیررسمی و میان فردی
- روابط الکترونیکی
- شبکه میان فردی

روشهاي هماهنگي اعضاي تيم

رهیافت های رسمی و غیر شخصي(رسمي؟؟؟؟) :

شامل اسناد ومدارک قابل تحویل مهندسی نرم افزار، نوشته های فنی ، زمان ارزیابی پروژه ، برنامه ها و ابزار کنترل پروژه ، بوجود آمدن تغییرات مجدد و مدارک مربوطه ، گزارشهای مربوط به پیگیری خطاها و اطلاعات سری.

شیوه های رسمی ومیان فردی(اشخاص) :

بر فعالیت های تضمین کیفیت که در مورد محصول کاری مهندسی نرم افزار به کار برده شده است ، تأ کید دارد. این شیوه ها شامل جلسات تجدید نظر رسمی ، بررسی مجدد طرح ها و کدها می باشد.

شیوه های غیر رسمی و میان فردی(اشخاص) :

این شیوه ها معمولا شامل جلسات گروهی و تیم ها می باشد که به منظور پخش اطلاعات و حل مشکلات و جمع آوری اطلاعات در مورد نیازهای کارمندان و توسعه انجام شده ، برگزار می شود.

روابط الکترونیکی :

شامل پست الکترونیک ، برد های الکترونیکی پژوهشی و در حالت گسترده و پیشرفته شامل سیستم های گفتگو ویدئویی

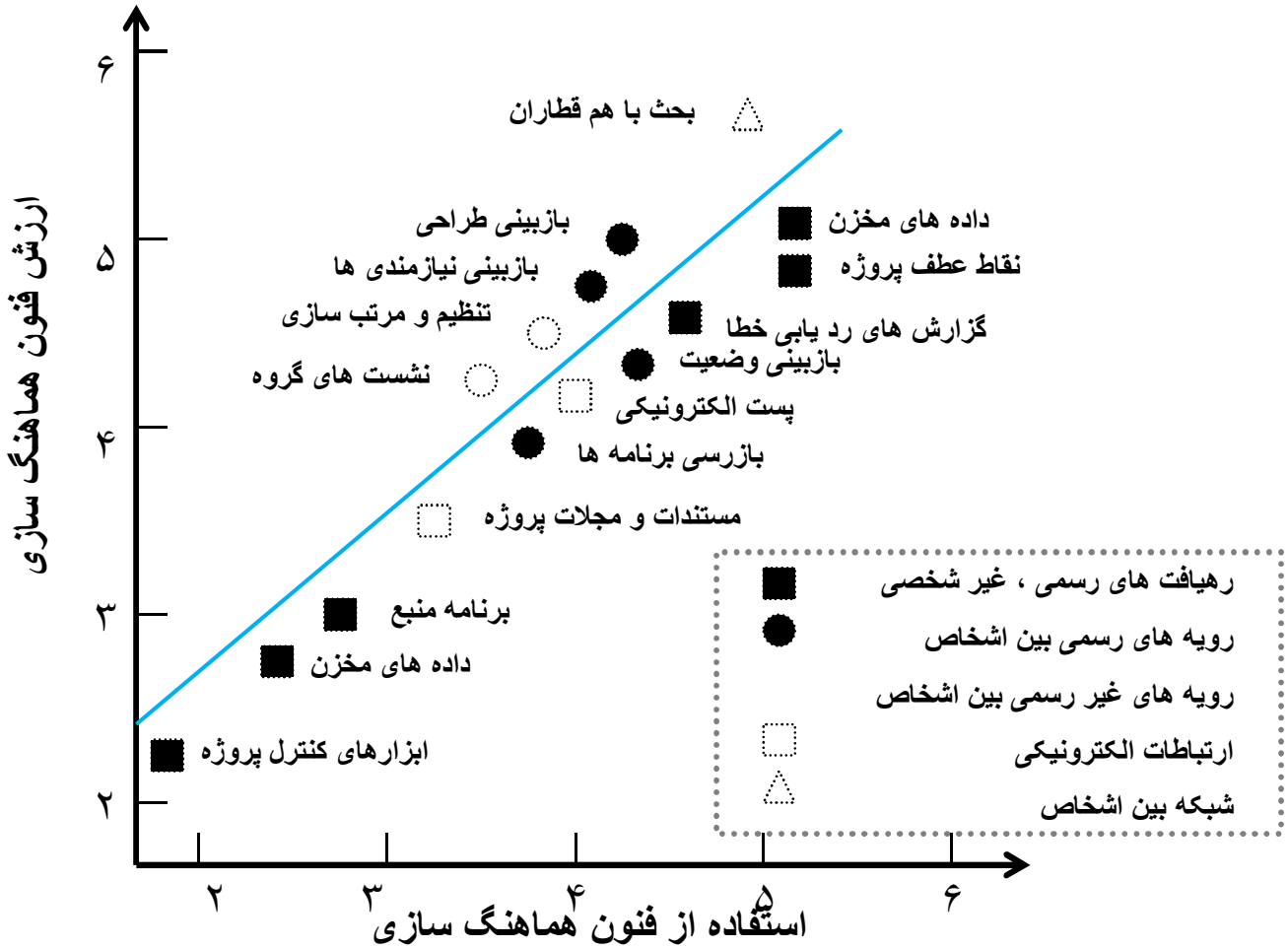
شبکه میان فردی :

در اختیار قرار دادن تجارب قبلي اعضا

نام درس:مهندسی نرم افزار ۱

ارزیابی کارایی این تکنیک ها

برای ارزیابی کارایی این تکنیک ها در هماهنگ کردن پروژه **کارول و استریتر** پروژه نرم افزاری را که شامل صدها نفر کارمند فنی مورد بررسی قرار دادد. مشخص می شود که اندازه مشخص شده برای هماهنگ کننده ها و فنون مختلف روابطی در مقابل تعداد دفعات استفاده از آنها در طول یک پروژه طرح ریزی شده است. فنونی که در قسمت بالایی **رگرسیون** قرار گرفته اند ، با در نظر گرفتن **تعداد دفعات** آنها به عنوان فنون با ارزش و مفید در نظر گرفته می شوند. اما خطوطی که در پایین خط رگرسیون واقع شده اند ، از ارزش کمتری برخوردارند.جالب است بدانید شبکه ارتباطی بین فردی به عنوان تکنیکی که دارای بالاترین میزان هماهنگی و بالاترین اندازه ارتباط است ، مشخص شد.همچنین ذکر این نکته نیز حائز اهمیت است که مکانیسم های اولیه تضمین کیفیت نرم افزاری (بازبینی نیازمندی ها و طراحی) نسبت به ارزیابی های بعدی کد برنامه منبع (بازرسی های مربوط به کدها) از ارزش بیشتری برخوردار هستند.



نام درس:مهندسی نرم افزار ۱

محصول

باید محصول و مسئله ای را که قراراست حل نماییم درهمان ابتدای شروع پروژه ، مورد بررسی قرار دهیم ، حداقل باید حوزه و حیطه ودامنه محصول تعیین گردد.

دامنه نرم افزار :

اولین فعالیت مدیریت پروژه تعیین دامنه نرم افزار است.که با پاسخ به سوالات زیر مشخص می شود:

سوالاتي براي تعيين دامنه نرم افزار:

۱. **بافت(جهت تعیین قیود لازم)** : چگونه می توان نرم افزاری ساخت که با سیستمی یا محصولی یا بافت تجاری بزرگتر هماهنگ باشد و قیود ناشی از این امر چه چیزهایی می باشد؟

۲. **اشیاء اطلاعاتی(ورودي/خروجي)** : کدام اشیاء داده ای است که مشتری به گونه ای واضح آنها را می بیند و درک می کند و به عنوان خروجی اصلی حاصل از نرم افزار بوجود می آورند ؟ برای ورودی چه اشیاء داده ای مورد نیاز است؟

۳. **کارکرد وعملکرد(عملیات)** : نرم افزار برای تبدیل داده های ورودی به داده های خروجی چه عملی را انجام می دهد؟ آیا ویژگی های عملکردی خاصی باید مورد بررسی قرار گیرد؟

برای ساخت یک طرح پروژه قابل دفاع اولویت با شکستن مسئله برای حل آن می باشد.

تجزیه و شکستن مسئله :

تجزیه مشکل که در بعضی مواقع تفکیک سازی مشکل یا تشریح مشکل نامیده می شود، فعالیتی است که **در مرکز تجزیه و تحلیل نیازمندی های نرم افزاری قرار دارد**.در طول انجام فعالیت مربوط به مشخص کردن دامنه ، هیچ تلاشی برای تجزیه کامل مشکل انجام نمی شود. بلکه تجزیه در دو محدوده اصلی ذیل به کار برده میشود:

۱زمینه هاي تجزيه :۱- عملکردهایی که باید تحویل شوند. ۲- روند استفاده شده برای ارائه این تحویلها

با مشخص شدن صورت وضعیت دامنه ، اولین مرحله تفکیک سازی به صورت طبیعی اتفاق می افتد.

نام درس:مهندسی نرم افزار ۱

فرآیند

مراحل انجام پروژه: **a-انتخاب مدل b-ایجاد طرح اولیه و cpf ها طبق مدل c-تجزیه مسئله(توابع)**

مراحل کلی مشخص کننده فرآیند نرم افزار – **تعریف ، توسعه و پشتیبانی** – در مورد تمام نرم افزارها قابل اجرا خواهند بود.
مشکل موجود در این مرحله انتخاب مدل فرآیند متناسب با نرم افزار می باشد که قرار است توسط تیم پروژه طرح ریزی شود.

مدل ترتیبی خطی	مدل ایجاد نمونه اولیه	مدل RAD	مدل افزایشی
مدل حلزونی(Spiral)	مدل برنده - برنده (Win-Win)	مدل توسعه همزمان	
مدل توسعه مبتنی بر مولفه	مدل روشهای رسمی (Formal)	تکنیکهای نسل چهارم	

a مدیر پروژه باید تصمیم گیری کند که کدام یک از نمونه های فرآیند براي نيل به اهداف زیر مناسب می باشد:

۱- مشتریانی که این محصول را سفارش داده اند و مردمی که کار مربوط به آن را انجام خواهند داد.
۲- مشخصه های خود محصول
۳- محیط پروژه که در آن تیم نرم افزاری مشغول به کار است.

b پس ازانتخاب مدل پردازشی ، تیم ، طرح اولیه پروژه را براساس مجموعه ای از فعالیت های مشترک و رایج چارچوب فرآیند تعریف خواهد کرد.پس ازمشخص شدن طرح ابتدایی ، تفکیک و تجزیه فرآیند آغاز می شود. یعنی یک طرح کامل که منعکس کننده وظایف کاری مورد نیاز برای پر کردن چارچوب فعالیت ها است باید در این مرحله انجام شود.

cتجزیه همزمان با کارهاي مديريت شروع مي شود و همانند برنامه اي ليست فعاليتها را نشان مي دهد

cpf ↘

تلفیق محصول و فر آیند :

برنامه ریزی پروژه با تعیین محصول و فرآیند آغاز می شود ، و شامل يكسري فعاليت هايی تعريف شده سازماني(زمينه كاري) است:

- **ارتباط با مشتری** : کارهایی که برای کشف موثر نیازمندیها میان توسعه دهنده و مشتری لازم می باشند.
- **طرح ریزی** : کارهایی که برای تعریف منابع ، خطوط زمانی وسایر اطلاعات مربوط به پروژه نیاز می باشد.
- **تجزیه و تحلیل خطرهای احتمالی** : کار هایی که برای ارزیابی خطرهای احتمالی فنی و مدیریتی ضروری می باشند.
- **مهندسی** : کارهایی که برای ساخت یک یا چند مورد از کاربرد های طرح لازم می باشند.
- **ساخت و واگذاری** : کارهایی که برای ساخت ، امتحان ، نصب و حمایت از کاربر لازم می باشند.
- **ارزیابی مشتری** : کارهایی برای دستیابی به بازخور مشتری بر اساس ارزشیابی انجام شده از کارهای نرم افزاری که در طول مرحله طراحی ایجاد و در طول مرحله نصب اجرا شده اند.

تلفیق محصول(مساله) و فرایند

اولین کار برای برنامه ریزی است که با انتخاب مدل فرآیند ، چارچوب روند معمول نیز با آن تطابق پیدا می کند.درهرموردی CPF فوق الذکر (در اینجا : ارتباط با مشتری ، برنامه ریزی ، تجزیه و تحلیل احتمال خطر ، طراحی ، ساخت ، انتشار ، ارزیابی مشتری) را می توان متناسب با نمونه حل کرد. این مسئله در مورد مدل های خطی ، مدل های افزایشی یا تکراری ، مدل های تکامل یافته و حتی در مورد مدل های سوارکردنی همزمان نیز صادق است. CPF ثابت وغیر معمول است و به عنوان پایه ای برای تمام کار های نرم افزاری که توسط سازمان نرم افزاری اجرا می شوند ، عمل می کند. براي هر سلول و هر تابع عمده بایست تاریخ شروع و پایان هر فعالیت ، محصولاتي که بایست تولید شوند و منابع مورد نیاز تعیین شوند

فعالیتهای چارچوب مشترک فرآیند	ارتباط با مشتری	طرح ریزی	تحلیل ریسک	مهندسی
فعالیتهای مهندسی نرم افزار				
کارکردهای محصول				
ورودی متن				
ویرایش و قالب بندی				
اصلاح خودکار				
آرایش صفحه و صفحه بندی				
شاخص بندی خودکار وToc				
مدیریت فایل				
تولید مستندات				

نام درس:مهندسی نرم افزار ۱

پروژه

- به منظور دستیابی به یک پروژه نرم افزاری موفق؟؟؟ و مدیریت یك پروژه، ابتدا باید به این مطلب پی برد که چه مشکلی وجود دارد و چگونه می توان آن مشکل را برطرف نمود.
- **علائمی که از نظر جان ریل پروژه سیستم های اطلاعاتی در معرض خطا(۱۰ ریسك) قرار گرفته اند عبارتند از:**
  - ۱. متخصصان نرم افزاری نیازهای مشتری خود را درک نمی کنند.
  - ۲.حوزه عملکرد محصول ضعیف تعریف شده است.
  - ۳.تغییرات به خوبی کنترل نمی شوند.
  - ۴.فناوری انتخاب شده تغییر می یابد.
  - ۵.نیاز های تجاری در حال تغییر هستند.
  - ۶.موعد های مقرر و مهلت ها غیر واقعی هستند.
  - ۷.کاربران مقاوم هستند.
  - ۸. ضمانت این سیستم ها هرگز به صورت مناسب و درست حاصل نشده است.
  - ۹.افراد متخصص با مهارتهای مناسب در تیم پروژه وجود ندارد.
  - ۱۰.مدیران وکارورزان از به کار بردن بهترین تمرین ها و درسهای فرا گرفته شده اجتناب می کنند.
- **ریل رهیافت پنج قسمتی قضاوت صحیح را در مورد پروژه های نرم افزاری پیشنهاد می دهد :**

•مثال بسکتبال

۱. کار را به روشی صحیح آغاز کنی.

۲.نیروی سوق دهنده پروژه را حفظ کنید.(داشتن نیروي محرك)

۳.پیشرفت حاصله را دنبال کنید.

۴.در پروژه خود تصمیم گیری صحیح و عاقلانه انجام دهید.

۵.یک تجزیه وتحلیل نیز پس از اتمام پروژه انجام دهید.

اصل WWWWWHH یا W^5HH پروژه منتهی

Boehm روشی را مطرح می نماید که در آن **اهداف پروژه ، مسئولیت ها ، زمانبندی ها ، مدیریت و رویکردهای تکنیکی ، و منابع را با ذکر سوالاتی مورد توجه قرار می دهد**. این سوالات عبارتند از :

چرا سیستم توسعه داده می شود؟

Why is the system being developed?

چه چیزی در چه موقع انجام شود؟

What will be done, by **When**?

چه شخصی مسئول چه کاری می باشد؟

Who is responsible for a function?

از لحاظ سازمانی در کجا قرار دارند؟

Where are they organizationally located?

از لحاظ تکنیکی و مدیریتی کار چگونه انجام می شود؟

How will the job be done technically and managerially?

از هر منبع چه میزان مورد نیاز است؟

How much of each resource is needed?

اصل WHH بوهم صرف نظر از پیچیدگی پروژه نرم افزاری قابل استفاده است.سوالات فوق الذکر می تواند **طرحی عالی از برنامه ریزی را برای مدیران پروژه و تیم نرم افزاری فراهم کند.**

اقدامات بحرانی(حساس و لازم الاجرا)

شورای Air Lie لیستی از کارهای حساس مهم نرم افزاری را در ارتباط با مدیریت مبتنی بر عملکرد فراهم کرده اند. این کارها به صورت ثابت و پیوسته از سوی پروژه ها و سازمان های موفق نرم افزاری که عملکرد خط نهایی آنها دائما بیش از عملکرد متوسط صنایع می باشد ، مورد استفاده قرار گرفته و به عنوان کارهای حساس در نظر گرفته می شوند.

- **مدیریت رسمی ریسک و خطر:** برنامه ها براي ۱۰ خطر و ريسك عمده
- **هزینه عملی و برآورد برنامه(زمانبندي):**براورد هزينه تخميني نرم افزار
- **مدیریت پروژه مبتنی بر متریک ها(معیارها):**وجود معيارهاي لازم براي تعيين مشكلات
- **ردگیری مقادیر بدست آمده:**ارائه گزارشات ماهانه مقادیر مربوط به معیارها و...
- **پیگیری معایب در مقابل اهداف کیفیتی:**ارائه گزارش پیرامون معایب
- **آگاهي از برنامه ريزي افراد:**بررسي عملكرد كاركنان

اگر تیم پروژه نرم افزاری نتوانند به این سوالات پاسخ دهند یا پاسخ آنها به این سوالات به صورت ناقص باشد ، بررسی کامل و جامع کارهای پروژه ضروری خواهد بود.

فصل چهارم متریک های پروژه و فرآیند نرم افزارً

اندازه گيري اساس هر روش مهندسي از جمله مهندسي نرم افزار است (مثلا گرافيك/ادبيات/معماري...)

تعریف متریک

متریک های فرآیند و محصول نرم افزاری اندازه گیری های کمیتی هستند که به افرادی که با نرم افزارکار می کنند امکان می دهند تا نسبت به کارآمد بودن فرآیند و پروژه های نرم افزاری بینش پیدا کنند.

علت اهمیت متریک

اگر اندازه گيري صورت نپذيرد، قضاوت بر اساس ارزيابي ذهني باشد

مراحل کار با متریک ها

با تعريف و جمع آوري مجموعه اي محدود از اندازه ها شروع شده و برحسب متريك اندازه گرا يا كاركرد گرا نرمال مي شوندو نتايج مورد تحليل قرار مي گيرند

انواع متریکها :

۱-متريكهاي پروژه محصولات(نرم افزار) ۲-متريكهاي فرآيندها

دلایل اندازه گیری منابع /محصولات و فرآیندها

۱.مشخص کردن براي دستيابي به فهم فرآيندها،محصولات،منابع و محيطها

۲.ارزیابی کردن تعيين وضعيت نسبت به برنامه ريزي و امكان فهم زمان انحراف پروژه ها و فرآيندها

۳.پیش بینی کردن براي انجام برنامه ريزي مورد نياز است

۴.پیشرفت کردن(اصلاح) ايجاد پيشرفت با شناسايي موانع و نواقص و فرصتهاي بهبود كيفيت

واژه اندازه ها،متریکها و معیارها

لغتهاي measure (اندازه) و metric (متريك) غالبا به جاي يكديگر به كار مي روند بايد توجه داشت كه با يكديگر تفاوت دارند.

- **Measure اندازه** = يک شاخص كمي از اندازه، مقدار، ابعاد و ظرفيت يک صفت از يک محصول يا فرآنيد را تعيين مي كند.
- **metric (متريك)** = از تركيب اندازه هاي منفرد، دركي از پروژه يا محصول حاصل مي شود
- **Indicator(شاخص)** = يك معيار شاخص يك متريك يا تركيبي از متريكهاست كه به صورت نتيجه حاصل شده است

مثال:

تيم ۱ : ۵۰ نفر ساعت كار كرده است — به ازاي هر نفرساعت ۶ خطا(متريك)

تيم ۲ : ۱۰۰ نفر ساعت كاركرده است — به ازاي هر نفرساعت ۸ خطا(متريك)

نتيجه (شاخص) :تيم ۱ بهتر بوده

متریک ها در حوزه پروژه و فرآیند

در نتيجه جمع آوري مناسب متريكها مي توان :

۱.تولید نشانگرهای فرآیندی به یک سازمان مهندسی نرم افزار امکان می دهند تا ارزیابی کنند چه چیزی موثر و چه چیزی غیر موثر است .متریک فرآیند براي توليد نشانگر در سرتاسر پروژه ها وطی زمان های طولانی جمع آوری میشود .هدف آنها ایجاد علامتهاي پروژه است كه به **بهبود طولانی مدت** در فرآیند نرم افزاری منجر می شود.

۲.تولید نشانگرها(علامتهاي) پروژه در **کوتاه مدت** به كار ما مي آيند و به مدير پروژه امكان مي دهند:

۱- خطرات بالقوه را پیدا می کند. — ۲- زمینه های مشکل آفرین را چاره میکند.

۳- جریان کار یا وظایف را تنظیم می کند. — ۴- وضعیت یک پروژه در حال اجرا را ارزیابی می کند.

۵-ارزيابي توانايي تيمهاي پروژه براي كنترل كيفيت محصول

متریکهای فرآیند و اصلاح فرآیند نرم افزار

راههای منطقی برای پیشبرد هرگونه فرآیند

۱. **اندازه گیری های** خاص آن فرآیند

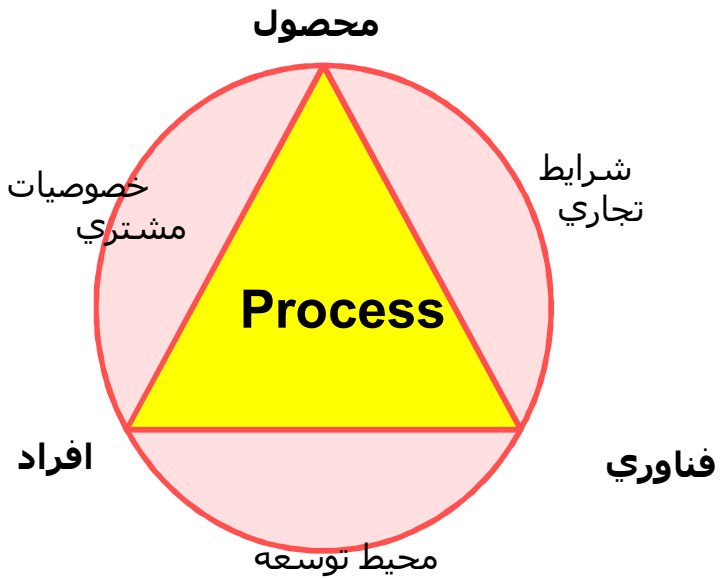
۲. توسعه ی مجموعه ای از **متریکها** ی فرآیند

۳. استفاده از این متریک ها برای **تهیه ی علامت ها و شاخص** هایی که باعث ایجاد یک راهبرد برای پیشرفت شوند.

عوامل موثر بر کیفیت نرم افزاري و كارايي سازماني

مثلث فرایند
- مهارت و انگیزه افراد بیشترین تاثیر
- پیچیدگي محصول
- فناوري يا روشهاي مهندسي نرم افزار

دایره شرایط محیطي
- محیط توسعه(ابزارها)
- شرايط تجاري(مهلت ها)
- خصوصيات مشتري(سهولت ارتباط)



چگونگي اندازه گيري كارايي فرايند

به طور غير مستقيم و توسط متريكهايي كه از نتايج قابل حصول از فرآيند استخراج مي شوند

براي داده هاي فرآيند **استفاده هاي عمومي و خصوصي** وجود دارد در حالت خصوصي يا شخصي متريكها ،**نشانگري** براي آن شخص هستند

انواع داده هاي قابل حصول از فرآيند

۱.متريكهاي شخصي فرآيند و فلسفه فرآيند نرم افزار شخصي(PSP)

يک مجموعه ساخت يافته از توصيف هاي پردازشي /اندازه گيريها و روشهايي که مي توانند به مهندسان در پيشبرد عملکرد شخصي شان کمک نمايند .اين مجموعه به آنها نشان ميدهد که چگونه فرآيند ها را تعريف کرده و کيفيت و بهره وري آن ها را اندازه گيري نمايند.يکي از اصول PSP آن است که **همه افراد با هم فرق مي کنند** و روشي که براي يک مهندس مفيد است شايد براي ديگري مناسب نباشد.داراي اهميتند چراکه راهنماي مهمي براي اصلاح فعاليتهاي هر مهندس نرم افزارند.
مثل نرخ خطاهاي فردي – خطاهاي پيمانه اي يا خطاهي يافت شده ضمن توسعه

۲.متريكهاي عمومي فرآيند

قادر به ارتقا کارايي فرايند سازماني مي باشند
و ممکن است براي افراد يا تيمي خصوصي باشند ولي براي همه اعضاي تيم عمومي هستند
مثل نرخ خطاهاي موجود در سطح پروژه

رهنمودهايي پيرامون جمع کردن متريکها نرم افزار

- استفاده از حساسيت سازماني در زمان تفسير متريکها
- امکان فيدبک و بازخور براي افراد مسئول جمع آوري اندازه ها
- براي ارزيابي افراد ، متريک و شاخص استفاده نشود
- عدم استفاده از متريکها براي ترساندن و تهديد افراد بلکه براي دستيابي به اهداف و ارتقا فرآيند
- داده هاي نمايشگر مشکل را نبايد با ديد منفي نگريست بلکه در ارتقا به کار برد

پيشرفت آماري فرآيند نرم افزار (SSPI)

استخراج معيارها و نمايشگرها از متريکها براي ارتقا آماري فرايند نرم افزار لازم است.
در اصل SSPI **استفاده از تحليل خرابي و شکست نرم افزار** براي جمع آوري تمام خطاهاي مشخص شده در ضمن توسعه برنامه کاربردي ، سيستم يا محصول

نام درس:مهندسی نرم افزار ۱

نحوه عملکرد تحلیل خرابي(شکست)

۱. همه خطاها و معایب بسته به منشاء شان **طبقه بندی** میشوند.(طبق مشخصاتشان)

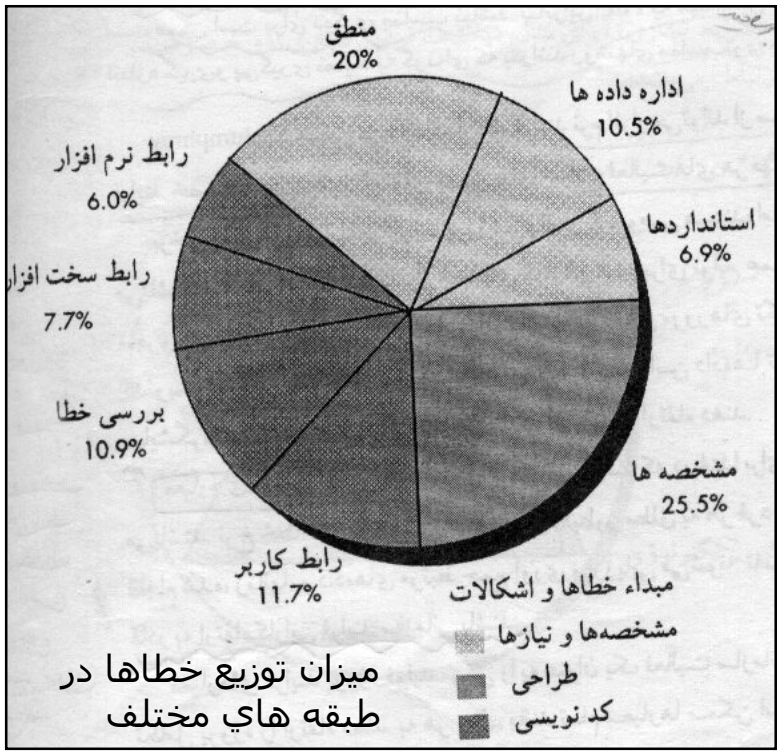
۲. هزینه تصحیح هر خطا و عیب ثبت می شود.

۳. **تعداد خطا ها و معایب در هر طبقه** شمرده میشود و آنها به ترتیب از بالا به پایین رده بندی می شوند.

۴. **هزینه کلی خطاها** و معایب در **هر طبقه** محاسبه میشود

۵.تحلیل داده ها براي **یافتن طبقه اي که بیشترین هزینه را براي سازمان** دارد

۶.**ایجاد طرحهایي** براي **حذف خطاها** از آن طبقه



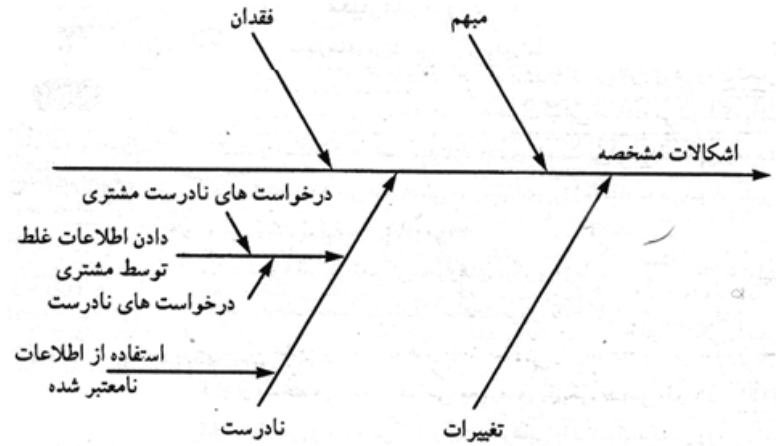
میزان توزیع خطاها در طبقه هاي مختلف

نمودار استخوان ماهي

براي کمك به تشخیص داده در هر طبقه از نمودار درصد عیوب نمودار استخوان ماهي پیشنهاد شده است

خط مرکزي : فاکتور کیفي مورد نظر

خطوط مورب:دلایل ایجاد مشکل



متریکهای پروژه

دانستيم متريكها **فرآيند** براي مقاصد **راهبردي** (و بلند مدت) استفاده مي شدند

اندازه هاي پروژه **تاكتيكي** هستند ،يعني متريكهاي حاصله براي تطبيق جريان كار پروژه و فعاليتهاي تكنيكي توسط يك مدير پروژه و تيم نرم افزار مورد استفاده قرار مي گيرند

در زمان تخمين ، براي اولين بار از متريكهاي پروژه استفاده مي شود،چرا كه متريكهاي بدست امده در پروژه هاي قبلي مورد استفاده قرار مي گيرند.

با شروع پروژه و فعاليتهاي تكنيكي ،متريكهاي ديگر پروژه مورد اهميت قرار مي گيرند،و اندازه هايي چون تعداد صفحات اسناد،تعداد خطوط كد و... محاسبه مي شوند

اهداف متريك هاي پروژه: (جنبه هاي متريكهاي پروژه)

۱. متریک ها برای به حداقل رساندن زمان برنامه پیشرفت از طریق انجام تطبیق های لازم برای پرهیز از تاخیرها و کم کردن خطرات و مشکلات بالقوه بکار می روند.

۲. متریکهای پروژه به منظور سنجش کیفیت محصول بصورت مستمر و اصلاح رهیافت فنی برای بهبود کیفیت در صورت لزوم مورد استفاده قرار می گیرد.

مواردي كه يك پروژه بايست اندازه گيري كند

۱. **ورودی ها-**اندازه منابع (یعنی افراد ومحیط) لازم برای انجام کار : ۱۰ نفر برنامه نویس ،۳ تا کامپیوتر

۲.**خروجی ها-**اندازه گیری های محصولات کاری یا قابل حمل که طی فرآیند مهندسی نرم افزار تولید میشوند.:۵ تا فرم،۲ تاجدول

۳. **نتایج-**اندازه هایی که موثر بودن کالاهای قابل حمل و تحویل را بیان میکنند. : ۸خطا در قسمت نمایش و...

اندازه گیری نرم افزار

۱. اندازه گیری مستقیم:شامل هزینه وتلاش به کار رفته می باشد. باشد.مانند:خطوط کد تولیدی –سرعت اجرا – اندازه حافظه

۲. اندازه گیری غیر مستقیم عبارتند از :عملکرد –کیفیت – پیچیدگی- کارآیی- قابلیت اطمینان و نگهداری

دلیل نیاز به نرمال سازی

عوامل زيادي بر توليد نرم افزار تاثير مي گذارند و نمي توان به صورت مستقيم متريكها را باهم مقايسه كرد

مثلا تيم A در يك پروژه ۱۸۰ خطا دارد و تيم B در يك پروژه ۲۲۰ خطا دارد كداميك بهترند؟ نمي توان جواب داد چون به ميزان پيچيدگي پروژه و ... نيز وابسته است

اگر اين داده ها نرمال شوند متريكهايي براي مقايسه حاصل مي شود

انواع متریکهای نرمال شده

۱-مبتني بر اندازه **۲-مبتنی بر کارکرد**

Project	LOC	میزان فعالیت	$(000	مستندات	Errors	عیب	افراد
alpha	12,100	24	168	365	134	29	3
beta	27,200	62	440	1224	321	86	5
gamma	20,200	43	314	1050	256	64	6
•	•	•	•	•	•		
•	•	•	•	•	•		
•	•	•	•	•	•		

۱–متریکهای مبتنی بر اندازه (اندازه گرا)

اين متريكها از طريق نرمال كردن متريكهاي كيفيت و با در نظر گرفتن اندازه و حجم نرم افزار توليد شده بدست مي آيند

براي هر سازمان نرم افزاري(پروژه) مي توان جدولي از اندازه هاي مبتني بر اندازه گيري ارائه داد

اين اطلاعات شامل تمام فازهاي ساخت پروژه هستند

عيبها بعد از تحويل محصول مشخص مي شوند و خطاها در زمان ساخت

از تعداد خطوط برنامه (LOC) مي توان براي نرمال سازي داده هاي اوليه استفاده كرد

متريكهاي نرمال شده بر اساس تعداد خطوط برنامه

۱. خطاها در klocهزار خط برنامه)
۲. تعداد عيب ها در هر هزار خط
۳. هزينه هر خط برنامهLOC
۴.تعداد صفحات مستندات به ازاي هرKLOC

متريكهاي نرمال شده بر اساس ميزان فعاليت

۵. تعداد خطا ها به ازای هر نفر-ماه
۶. تعداد خطوط برنامه به ازای هر نفر –ماه
۷.هزينه هر صفحه از مستندات

ايده موافقين با استفاده از متريكهايي چون LOCشمارش آن براي اغلب پروژه ها ساده و روشن است

ايده مخالفين با استفاده از متريكهايي چون LOC تعداد خطوط به زبان برنامه نويسي وابسته است و ممكن است طراحي خوب تعداد خطوط برنامه را بكاهد.در زبانهاي غير رويه اي قابل استفاده نيستند

۲–متریکهای مبتنی بر کارکرد (کارکرد گرا)

از یک معیار کار کردی که بوسیله برنامه نرم افزاری بعنوان یک ارزش نرمال سازی تحویل می گردد ، استفاده می نمایند .

به علت عدم امكان اندازه گيري مستقيم عملكرد **از طريق اندازه هاي مستقيم ، اندازه گيري غير مستقيمي** براي عملكرد محاسبه مي كنيم

با استفاده از یک رابطه تجربی بر اساس اندازه های قابل شمارش (مستقیم) دامنه اطلاعات نرم افزاری و ارزیابی پیچیدگی نرم افزاری بدست می آیند.

$$FP = \text{count total} \times [0.65 + 0.01 \times \sum(F_i)]$$

امتيازات عملكردي(Fp-function point) با تكميل جدول محاسبه مي شوند. مقادير جدول عبارتند از:

۱-تعداد ورودی کاربر :هر ورودی که داده های محوری و مورد نیاز برنامه را فراهم آورد.داده های ورودی باید با سوالاتی که جداگانه پرسیده می شوند فرق داشته باشند .

۲-تعداد خروجی کاربر :هر خروجی (به گزارشها ، بررسی ها ، پیام های خطا و ... اشاره دارد) که اطلاعات مبتنی بر برنامه کاربردی نرم افزار را فراهم آورد .

۳-تعداد در خواستهای کاربر :هر ورودی مرتبط که باعث بوجود آوردن چندین پاسخ نرم افزاری فوری به شکل خروجی مرتبط می شود.

۴-تعداد فایلها :هر نوع فایل اصلی منطقی (گروه بندی منطقی داده هایی که ممکن است یکی از بحثهای یک پایگاه داده های بزرگ یا یک فایل جدا باشند .)

۵-تعداد رابطه های خارجی :هر رابطه قابل خوانده شدن توسط ماشین (فایلهای داده ها بر روی نوار یا دیسک) که برای انتقال اطلاعات به یک سیستم دیگر بکار می روند .

ضرایب وزنی

پارامتر مدیریت	تعداد		ساده	متوسط	پیچیده	
تعداد ورودی های کاربر		×	3	4	6	=
تعداد خروجی های کاربر		×	4	5	7	=
تعداد درخواست های کاربر		×	3	4	6	=
تعداد فایلها		×	7	10	15	=
تعداد رابط های خارجی		×	5	7	10	=
Count total						

تعیین پیچیدگی

وقتی داده ها جمع آوری شدند ، یک مقدار پیچیدگی با هر شمارش همراه می شود ، که تعیین آن تا حدی ذهنی است .و شامل ۱۴ سوال در رابطه با پروژه ، با امتیازات مابین ۰(بي اهميت) تا ۵(خيلي مهم)

۱-آیا سیستم به پشتیبان و احیا و بازیابی فایل اطمینان نیاز دارد؟

۲-آیا ارتباطات داده ها مورد نیاز است ؟

۳-آیا عملگردهای پردازشی توزیع شده وجود دارند ؟

۴-آیا عملکرد آنها ضروری و بحرانی است ؟

۵-آیا سیستم در یک محیط عملیاتی کاربردی سنگین موجود اجرا شده است ؟

۶-نیاز به تراکنش براي ورود داده هاي آنلاين؟

۷-آیا فایلهای اصلی بطور متصل و روی خط به هنگام سازی می شوند ؟

۸-آیا فرآیندهای درونی پیچیده است ؟

۹-آیا تبدیلها و نصب چندگانه در سازمانهای مختلف طراحی شده است ؟

۱۰-امکان استفاده مجدد از کد وجود دارد؟

۱۱-آیا سیستم برای نصب چندگانه در سازمانهای مختلف طراحی شده است ؟

۱۲-امکان سهولت تغییرات ؟

۱۳-پيچيدگي ورودي/خروجي/فايلها و درخواستها؟

۱۴-آياسيستم به ورود داده ها روي خط(آنلاين) نياز دارد؟

بعد از محاسبه *امتیازات عملکردی* برای نرمال کردن اندازه ها ، جهت کیفیت و بهره وری نرم افزار و دیگر ویژگیها بر حسب FP (امتیاز کارکردي) موارد زیر نیز استفاده مي شود

۱.تعداد خطاها به ازای هر امتیاز کارکردی
۲.تعداد عیوب و نواقص به ازای هر امتیاز کارکردی
۳.تعداد صفحات مستندات به ازای هر امتیاز کارکردی
۴.تعداد امتیازات کارکردی به ازای هر نفر ماه

متریکهای توسعه یافته امتیاز کارکردی(امتیاز ویژگی/امتیازکارکردی سه بعدی)

استفاده از امتیاز کارکردي که فاقد بعد عملیاتي و رفتاري است و بر بعد داده تاکید دارد در سیستمهاي مهندسي و جاسازي شده چندان مناسب نیست

۱ يك امتياز كاركردي توسعه يافته به نام **امتياز ويژگي** به وجود امد
امتياز ويژگي ، كاربردهاي طبقه بندي شده اي را اندازه گيري مي كند كه در آنها پيچيدگي الگوريتم زياد است(مثل برنامه هاي بلادرنگ، كنترل فرايند و جاسازي شده)
براي محاسبه امتياز ويژگي ، مقدار امتياز كاركردي با ضريبي محاسبه مي شود به علاوه امتياز ويژگي ،مشخصه جديدي از نرم افزار به نام **الگوريتم** را نيز **شمارش** مي كند،يك الگوريتم به صورت مسئله اي محدود شده محاسباتي مشتمل بر يك برنامه خاص تعريف مي شود.مثل معكوس كردن ماتريس و ...

۲ توسعه ديگري براي امتياز كاركردي توسط شركت بوينگ ارائه شده ،**امتياز كاركردي سه بعدي** نام دارد
در اين روش علاوه بر بعد داده ، **بعدهاي كنترلي و عملياتي** را مشتمل مي شود

ویژگیهاي اين روش عبارتند از :شمارش شده(قابل شمارش) ، اندازه گیری شده(كمي بودن) ، تبديل شده (قابل تبديل) به اندازه ای كه معياری برای قابليت عملكرد تحويل شده توسط نرم افزار تهيه می كند .

بعد داده ها :در (مثل متریکهای مبتنی بر کارکرد)، تعداد داده های باقیمانده(فایلها) ۶و داده های بیرونی همراه با اندازه های پیچیدگی بکار می روند .

بعد عملیاتی :از طریق بررسی تعداد عملیاتهای داخلی که برای تبدیل ورودی به داده های خروجی مورد نیاز هستند.

بعد کنترل :از طریق شمردن تعداد جابجایی های بین وضعیتها اندازه گیری می شود .

نام درس:مهندسی نرم افزار ۱

وضعيت در بعد كنترلي :

نوعی حالت رفتاری بیرونی حالت مشاهده را نشان می دهد .

انتقال در بعد كنترلي :

در نتیجه حادثه ای که سبب می شود نرم افزار یا سیستم حالت رفتاری اش را تغییر دهد ، بوجود می آید .

مثال

یک تلفن بی سیم دارای نرم افزاری که از عملکردهای شماره گیری خودکار پشتیبانی می کند :

فشردن دکمه شماره گیری خودکار ، نمایشگر یک کد را نشان می دهد که بر مخاطب دلالت می کند . نرم افزار بلافاصله انرا به وضعیت شماره گیری تبدیل می کند .يعني تغيير حالت صورت مي پذيرد

در هنگام محاسبه امتیازات عملکردی 3D به انتقالها یک مقدار پیچیدگی نمی دهیم و از رابطه زیر برای محاسبه آن استفاده می کنیم :

$$Index = I + O + Q + F + E + T + R$$

که همگی پارامترها مقادیر وزن دار پیچیدگی را برای عناصر فوق الذکر نشان می دهند :
ورودی ها ، خروجی ها ، درخواست ها ، ساختار داده های درونی ، فایل های بیرونی ، تبدیل ، جابجایی

که هر مقدار وزن دار پیچیدگی با استفاده از رابطه زیر محاسبه می شود :

Low	Average	High

$$\text{مقدار پیچیدگي} = Nil\ Wil + Nia\ \ Wia + Nih\ \ Wih$$

که N ها **تعداد** وقوع عنصر i را برای هر سطح پیچیدگی(کم/متوسط/زیاد) نشان می دهند
و W ها **وزن** های معادل آن هستند .

ايده موافقين روشهاي مبتي بر كاركرد(fp):به زبان برنامه نويسي وابسته نيست و براي زبانهاي غير رويه اي مناسب است

ايده مخالفين روشهاي مبتني بركاركرد(fp):نيازمند تردستي هستيم و عيني نيست و گاها ذهني است.

تطبیق دو روش متریک(رابطه بین کارکردگرا و اندازه گرا)

ارتباط بين خطوط برنامه و امتيازات عملكردي وابسته به زبان برنامه نويسي استفاده شده براي پياده سازي نرم افزار و كيفيت طراحي است.
ميزان عملكردي را كه به وسيله برنامه كاربردي حاصل شود، مي توان با جزء به جزء نوشتن قطعات اصلي،داده هاي ورودي يا خروجي تخمين زد
معيارهاي loc وfp براي بدست آوردن معيارهاي قابليت توليد استفاده مي شوند.

زبان برنامه سازی	میانگین تعداد خطوط به ازای هرامتیاز کارکردی LOC / FP
Assembly Language	320
C	128
COBOL	106
FORTRAN	106
Pascal	90
C++	64
Ada 95	53
Visual Basic	32
Smaltalk	22
Powerbuilder (code generator)	16
SQL	12

•تعداد خطوط C++ تقریبا ۱.۶ برابر خطوط FORTRAN ، امتیاز و قابلیت عملکردی را ایجاد می کند .

متریکهای کیفیت نرم افزار

کیفیت برنامه کاربردی فقط با موارد زیر بدست مي آید

- مفید بودن نیازمندیهای توصیف کننده مشکل
- مفید بودن کد برنامه قابل اجرا

طراحی مربوط به مدل سازی راه حل

آزمونهایی که خطاها را آشکار می کنند

مهندس نرم افزار خوب از اندازه گیری برای موارد زیر استفاده مي کند

- ✓ ارزیابی کیفیت تحلیل
- ✓ کد منبع

مدلهای طراحی

آزمونهایی که در موقع ساخت نرم افزار بوجود آمده اند

اولین قدم در سطح پروژه براي ارزیابي کیفیت ، اندازه گیري خطاها و اشکالات است

متریکهاي زیر در **خصوص تاثیر و سودمندی فعالیتهایی** که متریکهایی برایشان به کار رفته ، بصیرت و آگاهی ایجاد مي کنند:

- ➢ تعداد خطاهای محصول کاری به ازای هر امتیاز کارکردی (نیازمندیها یا طراحی)
- ➢ خطاهای کشف شده به ازای هر ساعت بازبینی
- ➢ تعداد خطاهای کشف شده به ازای هر ساعت آزمون

نگاهی اجمالی بر فاکتورهای موثر بر کیفیت

جنبه هاي بررسي کیفیت در نرم افزار

۱. راه اندازی محصول

۲. بررسی محصول (تغییر در آن)

۳. انتقال محصول (اصلاح براي محیط دیگر)

بررسی رابطه بین عوامل کیفیتی و جنبه هاي دیگر مهندسي نرم افزار :

۱)ارائه چارچوب مکانیسمی برای اولویت بندی کیفیتها توسط مدیر پروژه

۲)ارائه ابزاری برای ارزیابی کمیتی از نحوه توسعه و پیشرفت نسبت به اهداف کلی

۳)ارائه یک چارچوب ارتباط بیشتر با پرسنل QA در تلاش برای پیشرفت

۴)استفاده از معیارهاي کیفیت پایین(شاخص) براي شناسایي استانداردهایي که بایست تقویت شوند

اندازه گیری کیفیت

اندازه های فراوانی از کیفیت نرم افزار وجود دارد ولی قابلیت زیر معیار های مفیدی ابعاد مختلف اندازه گیري هستند:

- **تصحیح (صحت)** **تعمیر و نگهداری**
- **پیوستگی (جامعیت)** **قابلیت استفاده**

- **صحت** : برنامه باید به درستی کار کند ،صحت تعیین درجه ای است که نرم افزار کار خود را انجام مي دهد رایجترین اندازه برای قابلیت اصلاح :تعداد نقصها (عدم انطباق با نیازمندیها) در هر هزار خط برنامه .

- **قابلیت نگهداری** : تلاش بیشتری می طلبد که همان سهولت تصحیح برنامه در هنگام مواجه شدن با یک خطا،انطباق درصورت تغییر یا ارتقا درصورت نیاز کاربر است
که متوسط زمان تغییر (MTTC) یک متریک ساده مبتنی بر زمان برای اندازه گیری آن می باشد.قابلیت نگهداري اگر بالا باشد مي طلبد که زمان تغییر پایین باشد

- **جامعیت (تمامیت) :** توانایی یک نرم افزار را برای مقاومت در برابر دستبردها به برنامه ها ، داده ها و اسناد ، می سنجد .
براي اندازه گیري جامعیت دو ویژگی باید بررسی شود :
  - تهدید : بروز یک حمله از نوعی خاص و در محدوده زمانی مشخص
  - امنیت : احتمال دفع نوعی خاص از حمله

تمامیت=Σ[(تهدید -1) * (امنیت -۱)]

- **قابلیت استفاده (سهولت کاربرد)** : اگر برنامه قابلیت استفاده و سازگاري با مشتري نباشد ، دچار نقص خواهد شد ، حتی در صورتیکه عملکردهای آن ارزشمند باشند .وبرحسب ۴ ویژگي اندازه گیري مي شود:
  ۱. مهارت فیزیکی یا هوشی که برای یادگیری سیستم مورد نیاز است .
  ۲. زمان مورد نیاز برای ماهر شدن در استفاده از سیستم .
  ۳. افزایش خالص بهره وری که موقعی اندازه گیری می شود که سیستم توسط فردی که بطور متوسط کارایی دارد مورد استفاده قرار گیرد .
  ۴. یک ارزیابی ذهنی و معقول از **دیدگاههای کاربران نسبت به سیستم** .

کارایی رفع نقص

DRE

یك متریك کیفیت که براي سطح پروژه و فرآیند مفید است. یک اندازه از توانایی جدا کردن فعالیتهای کنترل و تضمین کیفیت در موقعی بشمار می آید که در همه فعالیتهای چار چوب فرآیند به کار رفته باشد .

به طور ایده آل مساوی ۱ است . $DRE = E / (E + D)$

که E تعداد خطاهایی است که قبل از تحویل نرم افزار به کاربر نهایی مشاهده شده

و D خطاهایی است که بعد از تحویل یافت شده اند و در حالت عادی بزرگتر از ۰ است . – مقدار ایده آل DRE ، ۱ است

DRE را می توان راهی برای ارزیابی توانایی تیم در پیدا کردن خطاها قبل از انتقال به مهندسی نرم افزار یا فعالیت بعدی دانست .

اگر DRE بتواند ساختاري رابه وجود آورد که خطاها را طي فعالیتها پیدا کرد و رفع نمود به صورت $DRE = E_i / (E_i + E_{i+1})$ بیان مي شود که E_i تعداد خطاهاي یافت شده در مرحله i ام و E_{i+1} تعدادخطاهاي یافت شده ناشي از مرحله i در مرحله i+1 است

متریک های انسجام و جامعیت در فرایند مهندسی نرم افزار

اندازه گیري هنوز در شرکتها باب نشده است در این بخش چندین استدلال برای متریک ها را بررسی کرده و روشی را برای جایگزین کردن یک برنامه ی جمع اوری متریک ها در یک سازمان مهندسی نرم افزار ارائه می شود.

استدلالی بر متریک های نرم افزاری

علت به وجود آمدن اندازه گیري؟

اگر اندازه گیري انجام نشود راه دیگري براي ارزیابي پیشرفت وجود نداردوبدون پیشرفت شکست حتمي است

وظیفه اندازه گیري؟

فهمیدن وضیعت فعلي و یك مبناي پردازشي براي تعیین اهداف بیشتر

سؤالاتي که در سطح پروژه و تاکتیکي وجود دارد و اندازه گیري سبب پاس شدن آنها مي شود

کداميك از تجهیزات کاربران تغییر مي کند؟ براي هر پیمانه چه تعداد آزمون خطا لازم است؟

احتمال خطا در کدام پیمانه ها بالاست؟ هنگام آزمون انتظار چند خطا وجود دارد؟

Defect Removal Efficiency

استقرار خط مبنا(ایجاد یک مبنا)

از طریق تعیین یک مبنای متریک، می توان به فواید بسیاری در سطح فرایند، پروژه و محصول دست یافت
مبنای متریک های **حاوی اطلاعات جمع آوری شده از پروژه های قبلی توسعه نرم افزار** است و می تواند ساده و یامثل یک مبنای جامع که حاوی دهها اندازه گیری پروژه و متریک های مشتق از آنهاست،باشد.

•ویژگیهاي خط مبنا :

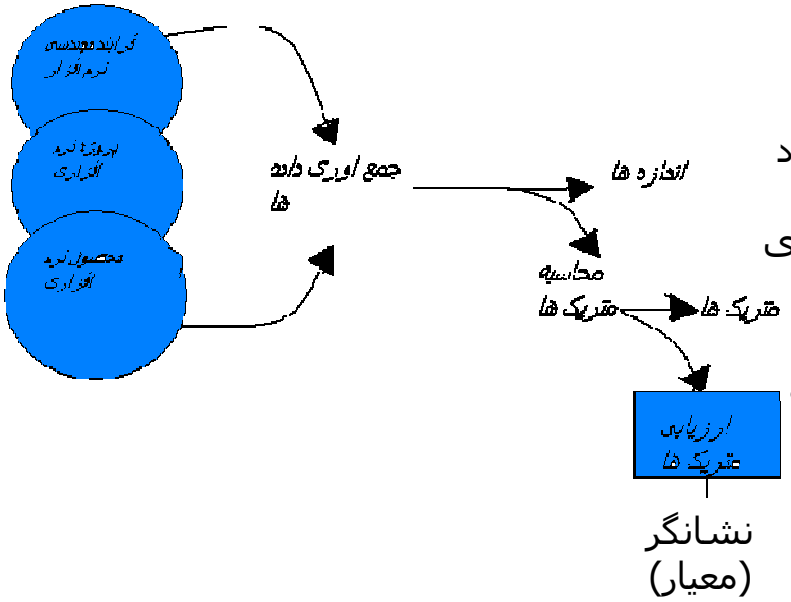
۱-داده ها صحیح باشند و حدسي نباشند
۲-داده را مي توان از هر تعدادي پروژه جمع آوري كرد
۳-اندازه ها بایست با هم تطابق داشته باشند(عدم شمارش کدHTML)
۴-برنامه ها بایست مشابه باشند

جمع آوری ، محاسبه و ارزیابی متریک ها

• در حالت ایده ال ، داده هایی که برای تعیین یک خط مبنا مورد نیاز هستند،بطورمستمر جمع اوری می شوند

• زمانی که اندازه ها جمع اوری می شوند، استفاده از متریک ها میسر مشود

•متریک ها می توانند دامنه وسیعی از متریک های fpیا loc و دیگرمتریک های کیفیت و مبتنی بر پروژه را گسترش دهند.در نهایت اینکه متریک ها باید طی مراحل براورد، کار تکنیکی، کنترل پروژه و بهبود فرایند، مورد *ارزیابی* قرار گیرند.ارزیابی متریک،**دلایل پنهان** در نتایج حاصله را آشکارمی کند و مجموعه *شاخص هایی* را برای راهنمایی پروژه یا فرایند بوجود می اورد.



•نمودار جمع اوري متريك ها براي ايجاد خط مبنا

توسعه وتکمیل متریکها و GQM (هدف– سوال – متریک)

یک سازمان برای دارا بودن یک برنامه **اندازه گیری صحیح که بیهوده نباشد** باید دارای اجزا زیر باشد:

۱. فرايندي براي توليد اهداف (از طريق بيان چند سوال)
۲. فرايندي براي تبديل اهداف به داده هاي پروژه اي.
۳. فرايندي براي تفسير داده هاي پروژ ه اي

- **GQM چیست؟**

يكي از تلاشهای اولیه برای توسعه مجموعه ای از اندازه گیریها می باشد که برای نرم افزار قابل استفاده است و به الگوی پیشرفت نرم افزاری مربوط میشود.

- **مراحل الگوي پيشرفت نرم افزار**

۱. تخمین در مورد یک پروژه و محیط آن، **تعیین اهداف کیفیتی** برای پروژه و انتخاب ابزارهای مناسب، روشهای مدیریتی و فناوری هایی برای ان اهداف تا شانس براورده شدن را داشته باشد.
۲. فرایند **اجرای** یک پروژه و نظارت بر داده های مربوط به این اهداف کیفیتی این کار در رابطه با فرایند عملکرد بر اساس داده ها در موقع براورده نشدن اهداف کیفیتی توسط مدیر پروژه انجام می شود.
۳. فرایندی برای **تحلیل** داده های جمع اوری شده در مرحله دوم بمنظور ارائه پیشنهاد برای پیشرفت بیشتر این فرایند مستلزم جستجو بدنبال مشکلات در جمع اوری داده ها و مشکلات در بکارگیری راهنمایی های کیفیتی ومشکل در تفسیر داده ها میباشد.

- **ابعاد هدف (GOAL) در GQM؟**

در سه مجموعه بازسازي مي شود كه عبارتند از : هدف ، جنبه و محيط

مدیریت تغییرات:کنترل آماری فرآیند

با توجه به اينكه پارامترهاي زيادي روي فرايند توليد نرم افزار تاثير دارند ،لذا متريكهاي بدست آمده از يك پروژه به پروژه ديگر تغير مي كنند.

- **نمودار کنترل:**

يك روش گرافيكي براي تعيين اينكه آيا تغييرات در داده هاي متريكها، معني دار هستند؟

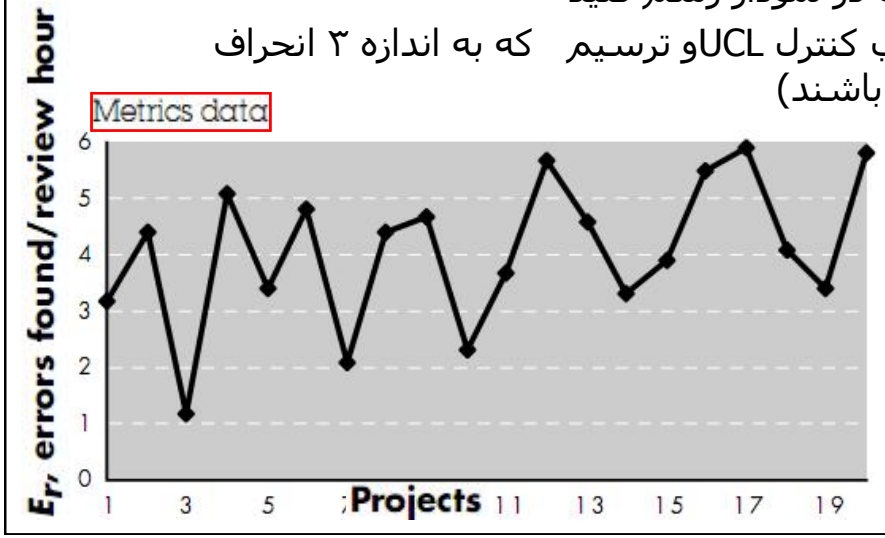
- این تکنیک افراد علاقه مند به بهبود فرایند نرم افزاری را قادر می سازد که تعیین کنند که آیا پراکندگی(تغییرات) و مکان(میانگین حرکت) متریک های فرایند پایدار(فقط شامل تغییرات طبیعي و کنترل شده) می باشند یا غیر پایدار (تغییر غیرقابل کنتر و معیارها براي پيشگويي كارايي نمي توانند استفاده شوند)

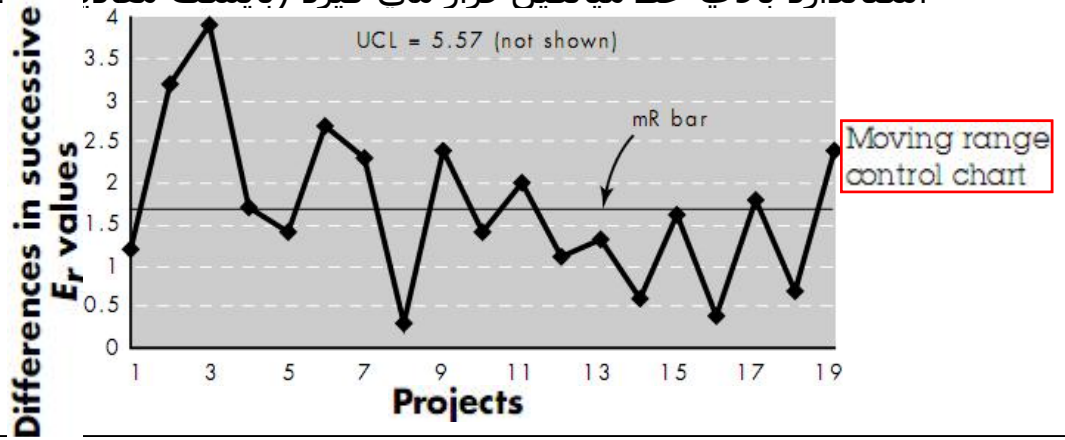
- **انواع نمودار کنترلی:**

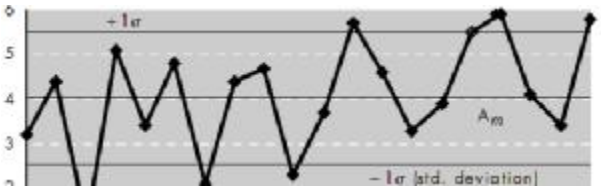
۱.نمودار کنترل طیف حرکت(MR)، ۲.نمودار کنترل فردی(مستقل)(NPL)

- **روش رسم نمودار کنترلي طيف حركت:**

a) محاسبه رنج تغيرات:قدر مطلق تفاوتهاي متوالي بين هر دو زوج نقاط داده ، و رسم اين رنج هاي متغير روي نمودار
b) محاسبه ميانگين رنج تغيرات، اين ميله MR را به عنوان خط مركزي در نمودار رسم كنيد
c) ميانگين حاصله را در ۳.۲۶۸ ضرب كرده،و آنرا به عنوان خط حد بالاي كنترل UCLو ترسيم كه به اندازه ۳ انحراف استاندارد بالاي خط ميانگين قرار مي گيرد (بايست مقادير ... UCL باشند)







❑ **روش رسم نمودار كنترلي فردي (افراد):**

a) مقادير متريک های فردی را رسم می کنیم

b) برای مقادیر متریک ها مقدار متوسط(A_m) را محاسبه می کنیم (میانگین متریکها)

c) مقدار میانگین (MR) برای مقادیر طیف حرکتی را در عدد ۲.۶۶۰ ضرب میکنیم و مقدار متوسط(A_m) بدست آمده در مرحله قبل را با ان جمع میکنیم.این نتایج حد بالای فرایند(UNPL) خواهند بود. آن را رسم مي كنيم

d) مقدار میانگین (MR)برای مقادیر طیف حرکتی را در عدد ۲.۶۶۰ ضرب می کنیم و مقدار متوسط(A_m)را از آن کم می کنیم .این نتایج حد پایین فرایند طبیعی(LNPL) خواهند بود. آن را رسم مي كنيم

e) انحراف استاندارد را از فرمول $(UNPL - A_m)/3$ محاسبه مي كنيم

❖ **قواعد مرزي(زون) :** براي ارزيابي اين مطلب استفاده شوند که آیا تغییرات نشان داده شده توسط این معیارها نشان دهنده ي فرآيندي هستند که تحت کنترل یا خارج از کنترل مي باشد، اگر هريك از اين ۴ اصل درست باشد نشان دهنده فرآيندي خارج از كنترل است:

- S= انحراف استاندارد
- يک مقدار منفرد متريک ها خارج از UNPL
- از سه مقدار متوالي متريکها دوتا ،2S از A_m فاصله داشته باشند
- از پنج مقدار متوالي متريکها سه تا، S از A_m فاصله داشته باشند
- قرار داشتن هشت مقدار متوالی متریک ها، در یک سوی مقدار متوسط

متریکهایی برای سازمانهای کوچک

روش پيشنهاد شده براي پيشنهاد متريك هاي نرم افزاري براي سازمانهاي كوچك

۱. **ساده نگهداشتن:**راهنمايي است که در حد معقولي در بسياري از فعاليتها خوب کار مي کند
۲. **برطرف سازي نيازهاي محلي**
۳. **اطمينان از افزايش ارزش**

مجموعه اندازه های ساده براي سازمان نرم افزاری کوچک

- زمانی که سپری می شود(به ساعت یا روز) تا یک درخواست داده شده ،کاملا ارزیابی شود. t_{queue}
- نیروی کار مورد نیاز(نفر_ساعت) برای ارزیابی W_{eval}
- زمانی که سپری می شود تا سفارش تغییر یک ارزیابی کامل شده به پرسنل داده شود. t_{eval}
- نیروی مورد نیاز برای اعمال تغییرات. W_{change}
- زمان مورد نیاز جهت اعمال تغییرات.t_{change}
- خطاهای یافت شده طی فعالیت اعمال تغییرات. .E_{change}
- عیوب کشف شده پس از آنکه تغییرات اعمال و محصول به مشتری تحویل گردیدهD_{change}

و استفاده از این اندازه ها براي ايجاد متريك ها

$$DRE=E_{change}/(E_{change}+D_{change})$$

ایجاد برنامه ای برای متریکهای نرم افزار

۱. اهداف تجاری خود را تعریف می کنیم
۲. مشخص نمایید که چه چیزهایی را می خواهید بدانید
۳. اهداف فرعی خود را مشخص کنید
۴. موجودیتها و صفات خاصه مرتبط با اهداف فرعی را تعریف کنید.
۵. اهداف اندازه گیری خود را رسم نمایید.
۶. پرسشهای کمی و شاخصهای مرتبط را مشخص کنید
۷. عناصر داده ای را مشخص کنید
۸. اندازه های مورد استفاده را مشخص کنید
۹. اقداماتی که برای پیاده سازی این اندازه گیری ها باید انجام دهید،مشخص کنید.
۱۰. یک طرح و برنامه برای پیاده سازی اندازه گیری ها آماده کنید

براي بدست آوردن اهداف تجاري ، ممكن است نياز به **اولويت بندي** داشته باشيم .

براي بدست آوردن بقيه سوالات ، يك **ليست سؤالات ايجاد مي كنيم** مثلا ليست سوالات موجوديت ايجاد مي كنيم . اين سوالات موجوديت به ما در شناسايي اهداف فرعي كمك مي كند.

فصل پنجم : طرح ریزی پروژه نرم افزاری

برنامه ریزی پروژه نرم افزاری چیست؟

مدیریت پروژه نرم افزاري با مجموعه فعاليتهايي آغاز مي شود كه جمعاً **برنامه ريزي پروژه** يا project planingناميده مي شود.
قبل از آغاز پروژه، مدير و تيم نرم افزاري بايد كاري را كه قرار است انجام گيرد،منابع لازم وزمان لازم از شروع تا انتها را تخمين بزنند.
شامل فعاليتهاي مختلفي چون تخمين،تحليل و مديريت ريسك،زمانبندي و پيگيري پروژه،تضمين كيفيت و ... مي باشد
در اين فصل ،برنامه ريزي شامل تخمين مي باشد يعني **تلاش شما براي تعيين ميزان پول ،كار لازم،تعداد منابع و مقدار زمان** لازم براي ايجاد يك سيستم يا محصول خاص مبتني بر نرم افزار.

❑چه كسي اين كار(تخمين) را انجام مي دهد؟

مديران نرم افزاري با استفاده از اطلاعات جمع آوري شده از مشتريان و مهندسان نرم افزاري و داده هاي متريك نرم افزاري حاصل از پروژه هاي گذشته اين كار را انجام مي دهند.

❑دليل اهميت تخمين چيست؟

شما هيچگاه خانه اي را بدون برآورد هزينه نمي سازيد منطقي است قبل از شروع به ايجاد نرم افزار،**برآوردي از هزينه** انجام دهيم.

❑چگونگي اطمينان از صحت اين مرحله؟

با داشتن تجربه و روش سيستماتيك تخمين صورت مي گيرد، با تعيين پيچيدگي و ريسك نسبت به انتخاب صحيح ديد مطمئن مي شويد

❑مراحل انجام تخمين؟

تخمين با **توصيف دامنه** محصول شروع مي شود.تا وقتي كه دامنه محدود است ،توسعه ي تخمين معني دار،امكانپذير نيست
سپس هر مسله به مسائل كوچكتر **تجزيه** مي شود و هريك توسط داده ها سابقه و تجربه **تخمين** زده مي شوند
توصيه كه تخمين خود را حداقل با دو روش متفاوت انجام دهيد. پيچيدگي مساله و خطر قبل از برآورد نهايي در نظر گرفته مي شوند.

❑محصول نهايي چيست؟

جدول ساده اي كه جزييات كارهايي را كه بايد صورت بگيرند،توابعي كه بايد ايجاد **شوند،و هزينه،تلاش،زمان** لازم براي هر يك را نشان مي دهد، فهرستي از منابع لازم نيز ارايه مي گردد.

مشاهده برآوردها(حقایقی در مورد برآوردها)

❖کار تخمين بطور بالفطره داراي خطراتي(ريسكهايي) هست و اين خطر(ريسكها) است كه منجر به عدم قطعيت مي شود.

۱. **پيچيدگي پروژه** تاثير شديدي روي عدم قطعيت در برنامه ريزي دارد. پيچيدگي و سخت بودن كار يك معيار(اندازه) نسبي است كه تحت تاثير آشنايي با كار و تجربه گذشته است.
   - ○توليد كننده يك برنامه كاربردي پيچيده تجارت الكترونيكي كه اولين بار آن را انجام مي دهد بايد آنرا بسيار مشكل بداند، اما تيم نرم افزاري كه اين كار را براي ده بار انجام داده ، آن را بسيار سهل مي داند.
   - ○برآوردهاي در مورد كد ها و طراحي داراي پيچيدگي بالا هستند

۲. **اندازه بزرگي پروژه** عامل مهمي ديگري است كه مي تواند بر دقت و كارآيي تخمين ها تاثير بگذارد. با افزايش اندازه بزرگي وابستگي دروني ميان عناصر مختلف نرم افزار به سرعت رشد مي كنند. تجزيه مسئله كه روش مهم ديگري براي تخمين زدن است ، سخت تر مي باشد زيرا عناصر تجزيه شده ممكن است سخت تر و مشكل تر باشند.

۳. ميزان **عدم قطعيت ساختاري**(عدم تعريف دقيق دامنه) نيز روي تخمين خطر تاثير دارد. در اينجا ، ساختار اشاره دارد به درجه اي كه نسبت به آن شرايط سخت شده ، ميزان راحتي كه با آن ميتوان توابع و عملكردها را بخش بندي ، نمود و ماهيت سلسله مراتبي اطلاعاتي كه بايد پردازش شوند.

۴. در دسترس بودن **اطلاعات تاريخي و سوابق قبلي** داراي تاثير مثبت شديدي بر تخمين خطر(ريسك) است.

❖خطر با ميزان عدم قطعيت در انجام برآوردهاي كيفي براي منابع،هزينه و جداول زمانبندي سنجيده ميشود. اگر دامنه پروژه اي به خوبي شناخته نشده يا نيازمنديهاي آن در معرض تغيير باشند ريسك و خطر به شدت بالا ميرود.

❖مدير پروژه نبايد در مورد تخمين وسواس بخرج دهد. شيوه هاي مدرن مهندسي نرم افزار(مثل مدلهاي فرآيند تكميلي)دبدگاه تكراري از توليد دارند. در چنين روشهايي ممكن است ميزان برآورد را مجدداً بازبيني نمود و و وقتي مشتري در نيازمنديها تغييراتي ايجاد مي كند آنرا اصلاح كنيم.

اهداف اصلی طرح ریزی پروژه

هدف برنامه ريزي پروژه نرم افزاري عبارتست از **مهيا كردن چارچوبي** كه مدير را قادر به **ارائه تخمين منطقي** از *منابع*(نيازها) ، *هزينه*(پول$$$$) و *زمانبندي*(زمان انجام فعاليتها و ...) كند. تخمين ها در يك چارچوب زماني در آغاز پروژه صورت گرفته و مرتباً با پيشروي پروژه بروز ميشوند.

تخمين ها تلاش دارند كه بهترين و بدترين طرح ها را معين كنند تا نتيجه پروژه را بتوان **محدود** نمود.

اهداف برنامه ريزي از طريق **فرآيند كشف اطلاعاتي** حاصل ميگردد كه منجر به تخمين هاي منطقي ميشود.

دامنه نرم افزار

اولين كار در برنامه ريزي پروژه عبارتست از تعيين دامنه آن.

عملكرد و كارايي مربوط به نرم افزار در مهندسي سيستم بايد ارزيابي گردد كه غير مبهم و در سطوح مديريتي و فني **قابل درك** باشد.گزارشي از اين دامنه بايد پيوست شود.

دامنه نرم افزار **توصيف كننده** اطلاعات و كنترل مورد پردازش ، عملكرد، كارايي، محدوديتها(شرايط حد☺)، رابط ها و قابليت اطمينان(اگر درست عمل نكرد بازهم به بحران نرسيم) است

- محدوديتها در نرم افزار را بوسيله سخت افزار خارجي يا حافظه موجود يا ديگر سيستمهاي موجود ايجاد مي شوند

به دست آوردن اطلاعات مورد نياز براي تعيين دامنه

❑برقراري ارتباط بين مشتري و توليدكننده

رايج ترين تكنيك مورد استفاده براي برقراري ارتباط بين مشتري و توليدكننده ،عبارتست از برگزاري **يك جلسه يا مصاحبه**.

۱-اولين مجموعه سئوالات آزاد روي **مشتري؛اهداف كلي و فوايد** تاكيد دارد

مثلاً :چه كسي در پشت تقاضاي اين كار است؟ چه كسي ا ز اين راه حل استفاده ميكند؟

حسن اقتصادي حل مساله چيست؟ آيا منبع ديگري براي حل وجود دارد؟ و ....

۲-سئوالات دوم تحليلگر را قادر به **شناخت بهتري از مسئله و مشتري** ميكند.مثلاً

اين راه حل براي چه مشكلاتي است؟ آيا محيط عملياتي با اين راه حل وجود دارد؟

۳-سئوالات آخر(سوم) بر ميزان **موثربودن جلسه** تاكيد دارد و آن را فوق پرسش مينامند و فهرست زير پيشنهاد ميشود :

آيا من سئوالات زيادي پرسيدم؟ آيا سئوالات من با مسئله اي كه مورد حل قرار داده اید مرتبط اند؟...

اين سئوالات (و سوالات ديگر) كمي به تحرك اوليه مسئله(؟؟؟)(ترجمه بهتر: **شكستن سردي بين افراد**) كمك كرده و ارتباطي را كه براي ايجاد دامنه پروژه لازم است ايجاد ميكند.

❖بخش سئوال و جواب تنها بايد براي اولين برخورد استفاده شود و سپس در قالب جلسه اي جايگزين آن شود.كه عناصر حل مسئله مذاكره و مشخصات را مرتبط ميسازد.

❖ **FAST** [1]: (ايجاد تيم مشترك به جاي يادداشت و رابطه ما و آنها) = پروژه اي است با نام **فنون ساده شده مشخصات برنامه هاي كاربردي** كه ايجاد تيم مشتركي از مشتريان و توليد كنندگاني كه براي حل مسئله با هم كار ميكنند را تشويق نموده،عناصر راه حل را پيشنهاد كرده و مجموعه اي از نيازمنديها را مشخص ميكند.

امكان سنجي(مرحله تكميلي بعد از تعيين دامنه)

•قدم بعد از تعيين دامنه (Scope) ، امكان سنجي است

•وقتي دامنه شناسايي شد منطقي است بپرسيم كه **آيا اين پروژه شدني است؟**

❖ **عملي بودن نرم افزار (امكان سنجي)داراي چهار بعد است**

۱. **فن آوري** : آيا از لحاظ تكنيكي قادر به انجام پروژه هستيم؟
۲. **هزينه:** آيا با هزينه هاي صورت گرفته امكان انجام پروژه هست؟
۳. **زمان:**آيا زمان توليد تا رسيدن پروژه به بازار ، قابل قبول است؟
۴. **منابع:**آيا سازمان تمام منابع مورد نياز را در اختيار تيم قرار داده است؟

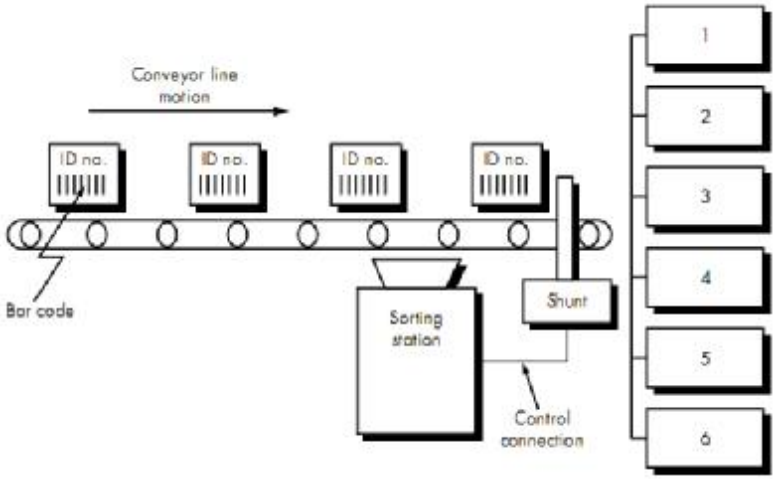
1. facilitated application specification techniques

تعیین دامنه کافي نیست.وقتي دامنه فهمیده شد ، تیم نرم افزاري و دیگران باید مشخص سازند که با ابعاد روشن شده آیا **امکان کار** وجود دارد یا خیر ؟ که این خود **قسمتي از فرآیند تخمین و برآورد** خواهد بود

مثال

مثال / نرم افزاري را براي نوعي سیستم مرتب کردن خط حمل و نقل یا CLSS در نظر بگیرید وضعیت دامنه را براي آن مشخص کنید.

[برای اطلاعات بیشتررجوع شود به صفحه ۱۶۰]



•تجزیه(طبق فصل ۳)

- خواندن ورودی بار کد.
- خواندن پالسهای دورسنج.
- دکد نمودن داده‌های کد شده.
- تعیین مکان قرارگیری جعبه.
- تولید سیگنال کنترلی برای اهرم.
- نگهداری اطلاعات مقصد جعبه‌ها.

دامنه نرم افزار **توصیف کننده** اطلاعات و کنترل مورد پردازش ، عملکرد، کارایی، محدودیتها(شرایط حد☺)، رابط ها و قابلیت اطمینان(اگر درست عمل نکرد بازهم به بحران نرسیم) است

•عملکرد را کارایي مشخص مي کند
•مثال کارایي: امکان تغییر سرعت از ۵ به ۱۰
•مثال محدودیت:جعبه ها از هم فاصله نداشته باشند
•مثال رابطه: با سخت افزار+نرم افزار مثلا:بانك اطلاعات ذرجه قطعات+افراد + رویه هاي قبل از بعد نرم افزار
•قالیت اطمینان : قطعه بي نشان را نگه ندارد چون خطا ایجاد مي شود

نام درس:مهندسی نرم افزار ۱

منابع

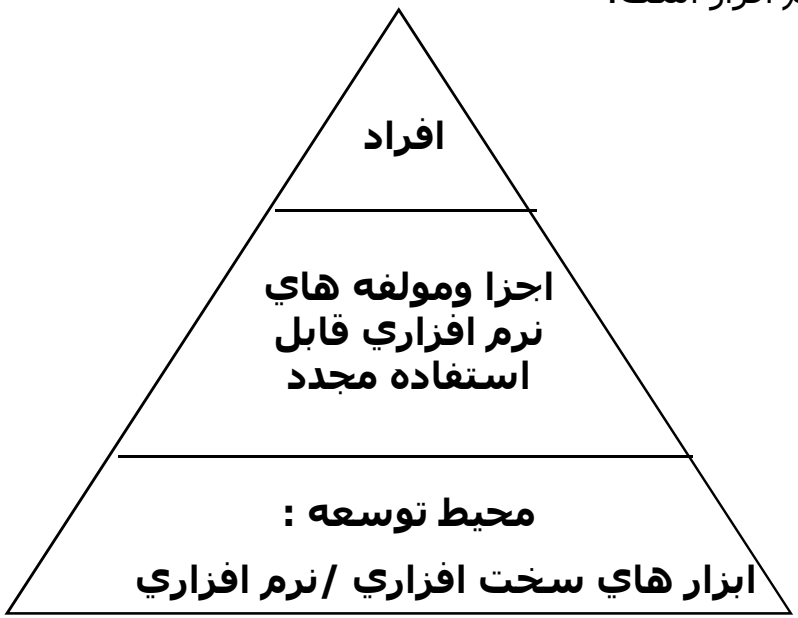
دومين كار در برنامه ريزي نرم افزاري تخمين منابع لازم براي نيل به كار توليد نرم افزار است.

❑**منابع مورد نياز براي توليد نرم افزار :**

۱-**محيط توسعه:**شامل ابزارهاي نرم افزاري و سخت افزاري و به عنوان زيربناي براي حمايت از تلاشهاي توسعه است

۲-منابع نرم افزاري با قابليت استفاده مجدد:كه سبب پايين آوردن هزينه توليد و سرعت بخشيدن به تحويل نرم افزار

۳-منابع اوليه: شامل مردم مي باشد در نوك هرم قرار گرفته اند



❑**هر منبع داراي چهار مشخصه است:**

•[۱]توصيف منبع ، [۲]وضعيت دسترسي به آن، [۳]زمان تقويمي كه در آن موقع مورد نياز است، [۴]مدت زماني كه منبع بكار گرفته ميشود.

•دو مشخصه [۳]زمان تقويمي كه در آن موقع مورد نياز است، [۴]مدت زماني كه منبع بكار گرفته ميشود را ميتوان بعنوان يك **پنجره** در نظر گرفت.

•در دسترس بودن منبع براي يك پنجره بخصوص بايد در اولين زمان ممكن صورت گيرد.

•**منابع**
- **۱-منابع انساني**
- **۲-منابع نرم افزاري قابل استفاده مجدد**
- **۳-منابع محيطي**

منابع انساني

تعداد افراد لازم براي پروژه را مي توان تنها بعد از تخمين كار توليد، ارائه كرد.

منابع نرم افزاری با قابلیت استفاده مجدد:

مهندسي نرم افزار برمبناي اجزاء (CBSE) برفعاليت كاربرد مجدد تاكيد دارد
يعني ايجاد و استفاده مجدد بلوكهاي ساختمان نرم افزار :چنين بلوكهايي به نام جزء يا اجزا ناميده مي شوند
اجزا براي ارجاع آسان بايد طبقه بندي شوند،براي به كارگيري آسان بايد استاندارد شوند و براي تركيب ساده بايد ارزيابي شوند.

❑ **چهار گروه نرم افزاري را بايد در برنامه ريزي مد نظر قرار گيرند**

۱-اجزاي ساخته شده حاضر و آماده (COTS) :آنها را آماده مي خريم و آماده و ارزيابي شده اند

۲-اجزايي كه داراي تجربه كاملي در مورد آنها هستيم:اجزاي پروژه هاي قبلي كه مشابه نرم افزار فعلي اند وافراد تيم جاري كاملا با اجزاي قبلي آشنا هستند و نيازمند پاره اي از اصلاحات هستند و كم خطر (ريسك) اند.

۳-اجزايي كه داراي تجربه نسبي در مورد آنها هستيم.:اجزاي پروژه هاي قبلي كه مشابه نرم افزار فعلي اند وافراد تيم جاري با اجزاي قبلي آشنايي محدودي دارند و نيازمند پاره اي از اصلاحات هستند لذا درجه پاييني از ريسك وجود دارد

۴-اجزاي جديد:اجزايي كه توسط تيم نرم افزار بايست ايجاد شوند.

❑ **هنگامي كه اجزاي قابل استفاده مجدد بعنوان يك منبع استفاده ميشوند رهنمودهاي زير بايد مد نظر باشند:**

۱. اگر اجزاي آماده مصرف(خارجي) نيازهاي پروژه را برآورد ميكنند آنهارا بخرند.
۲. اگر اجزايي كه در مورد آنها تجربه كامل داريم در دسترس هستند خطر مربوط به اصلاح و الحاق آنها معمولاً پذيرفتني است.و استفاده مشكلي ندارد
۳. اگر تجربه نسبي از اين جز داريم استفاده آن براي پروژه كنوني بايد تحليل شود.

منابع محيطي

- محيطي كه پروژه نرم افزاري را پشتيباني ميكند ، محيط مهندسی نرم افزار(SEE)است به **كار گيرنده سخت افزار و نرم افزار** است.
- سخت افزار سكويي است كه ابزارهاي لازم(نرم افزاري)براي توليد محصولات كاري را تامين ميكند كه نتيجه يك كار مهندسي خوب است.
- برنامه ريز پروژه بايد **پنجره زماني** لازم براي سخت افزار نرم افزار را مشخص كند

برآورد پروژه های نرم افزاری

•امروزه تخمينهاي هزينه و فعاليت نرم افزار به دليل پيچيده شدن و گران شدن نرم افزار از يك هنر ناشناخته به يك سري مراحل سيستماتيك تبديل شده است

❑براي رسيدن به تخمين هاي مورد اطميناني از نظر هزينه و كار يك سري گزينه وجود دارد:

۱. كار تخمين را تا اواخر پروژه به تعويق بيندازيم.(واضح است تخمين بعد از پروژه ۱۰۰% درست است(براي پروژه هاي بعدي)
۲. برآوردها را بر پايه پروژه هايي قرار دهيم كه هم اكنون تكميل شدند.
۳. از فنون نسبتاً ساده تفكيك و تجزيه براي ارائه اين برآوردها استفاده كنيم.(تخمين زيرمسئله ها...)
۴. از يك يا چند مدل تجربي براي آن استفاده نماييم.

روش اول چندان عملي نيست – روش دوم در بسياري موارد خوب است ولي نه هميشه

روش تجزيه از رهيافت تقسيم و غلبه استفاده مي كند، با تفكيك پروژه بصورت توابع و فعاليتهاي مهندسي مربوطه تخمين هزينه و كار بصورت مرحله اي انجام ميگيرد.

مدل تجربي برآورد را، ميتوان براي فنون جدا سازي تكميلي استفاده كرد و روش تخمين ارزشمندي را در جاي درست خود ارائه داد.

مدل تجربي به صورت $d = f(v_i)$ تعريف مي شود كه d يكي از مقادير تخمين(زمان،هزينه) را شامل مي شود و v_iپارامتر مستقل انتخابي (**مثل fp تخميني يا loc تخميني**)

نكته :**ابزارهاي خودكار تخمين برآورد** يك يا چند **تكنيك تجزيه يا مدلهاي تجربي** را به اجرا در مي آورند وقتي اين ابزارها با رابط گرافيكي كاربر تركيب ميشوند گزينه جالبي براي انجام تخمين ارائه مي دهند.در اين سيستمها مشخصه هاي سازمان توليدي (مثل تجربه و محيط) و نرم افزاري كه قرار است توليد شود توصيف ميگرددو تخمين از داده ها بدست مي آيد

بدون داده هاي سابقه ، برآورد هزينه اساس متزلزلي خواهد داشت.

فنون تجزیه

در اکثر موارد ارائه برآورد هزينه و فعاليت بزرگتر از حدي است كه به صورت يك قطعه در نظر گرفته شود،در نتيجه برنامه به قطعات كوچكتري تجزيه مي شود در فصل ۳ تجزيه را از دو ديدگاه : تجزيه مسئله و تجزيه فرآيند بررسي نموديم
در تخمين از يكي يا هردو شكل تجزيه استفاده مي كنيم .قبل از انجام برآورد(تخمين) برنامه ريز ، پروژه بايد دامنه نرم افزاري را كه قرار است ساخته شود شناخته و تخميني از اندازه آن ارائه كند.

❑**ميزان دقت تخمين پروژه بر اساس چند چيز پيش بيني ميشود(اصول صحت تخمين):**

۱-درجه درستي برآورد اندازه محصول:درجه اي كه نسبت به آن برنامه ريز به درستي اندازه محصول كاري را تخمين زده است.

۲- توانايي ترجمه ،تخمين اندازه نيروي كار انساني،تقويم زماني و ميزان پول.

۳- درجه انعكاس توانايي هاي تيم نرم افزاري :درجه كه نسبت به آن طرح پروژه منعكس كننده تواناييهاي تيم نرم افزاريست.

۴- پايداري خواسته هاي پروژه و محيط: ثبات نيازمنديهاي محصول و محيطي كه كار مهندسي نرم افزار را پشتيباني ميكند.

تعیین اندازه نرم افزار

تخمين اندازه نشان دهنده اولين كوشش عمده برنامه ريز پروژه مي باشد ، اندازه به نتيجه كمي پروژه اشاره دارد و اگر روش مستقيم به كار گرفته شود،اندازه مي تواند بر حسب LOC و اگر روش غير مستقيم به كار گرفته شود اندازه بر حسب FP نشان داده مي شود

❑**چهار روش مختلف براي تعيين ميزان مسئله :**

۱- شيوه اندازه زدن با منطق فازي :بيان اندازه برمبناي مقياس كمي و پالايش آن در محدوده اصلي

۲- شيوه اندازه زدن با امتيازات عملكردي: فصل ۴ و بحث FP

۳- شيوه اندازه زدن با جزء استاندارد:تخمين تعداد وقوع هر جز استاندارد(مثل:صفحات-گزارشها-فايلها-LOC و...)و استفاده از داده هاي سابقه براي تخمين اندازه نهايي محصول

۴- شيوه اندازه زدن با تغير اندازه:زمان اصلاح نرم افزار استفاده مي شود

۱–برآورد مبتنی بر مساله

داده هاي بدست آمده از روشهاي خطوط کد (LOC) وامتیازات کاربردی(FP)که از روی آنها می توان متریک را حساب کرد علاوه بر استفاده اي که در ارزيابي کيفيت داشتند در طول کار پروژه به دو صورت مورد استفاده قرار میگیرند:

۱- به عنوان **متغیر تخمینی** که برای تعیین اندازه هر عنصر نرم افزاری استفاده می شود.

۲- به عنوان **متریک خط مبناي** که از روی پروژه های قبلی جمع آوری شده همراه با متغیرهای تخمینی برای ارائه هزینه و میزان نیروی کار پروژه استفاده می شود.

•براورد LOCوFP فنون متمایزی از تخمین هستند.هردو دارای چندمشخصه مشترک هستند .

•مراحل تخمين کمي در مورد پروژه:

۱. برنامه ریز پروژه با وضعیت محدود شده دامنه نرم افزارشروع کرده و از روی آن تلاش می کند نرم افزار را به صورت توابعی ازمساله در بیاورد که بتوان هر کدام را به صورت مجزا براورد کرد. **LOCوFP**برای هر تابعی تخمین زده می شود. یا برنامه ریز جز دیگری را برای تعین اندازه انتخاب می کند مثل کلاسها یا اشیا،تغییرات یا فرایندهای تجاري تحت تاثیرقرار گرفته

۲. **سپس متریک های بهره وری خط مبدا مانند(LOC/pm یا FP/pm-pmمخفف نفر-ماه)با متغیر برآورد مناسب بکار گرفته شده** و هزینه یا نیروی کار تابع به دست می آید

۳. از روي تخمين توابع، **تخمين کمي** در مورد کل پروژه بدست مي آيد.

•نکته مهم مورد توجه این است که اغلب نکات پراکنده عمده اي در متریک های بهره وری یک سازمان وجود دارد که باعث مشکوك شدن استفاده از يك متريك توليد مبنا مي شوند

• فنون تخمینLOCوFP از نظر جزئیات لازم برای تفکیک و هدف تقسیم بندی با هم متفاوتند

•وقتی **LOC**به عنوان متغیر تخمین استفاده می شود **تجزیه** کاملا ضروری است و اغلب از نطر جزئیات بررسی می شود. و بر روي توابع نرم افزار متمرکز مي شويم

•در مورد تخمین های FPکار جدا سازی به صورت دیگری است علاوه بر تمرکز تابع هر یک از مشخصه های دامنه اطلاعات یعنی ورودی ها خروجی ها فایلهای اطلاعاتی و...به علاوه ۱۴ مقدار تطابق پیچیدگی (f_i ها) برآورده می شود.

❑مراحل تخمین

•برنامه ریز بدون توجه به متغیر تخمین با برآورد یک سری مقادیر برای هر یک از تابع ها یا مقدار دامنه اطلاعاتی شروع می کند، با استفاده از اطلاعات تاریخی یا حدس برنامه ریز اندازه بزرگی هر تابع را به صورت خوشبینانه یا از روی بد بینی تخمین می زند.میزان عدم قطعیت هنگامی مهیا می شود که دامنه ای از اعدادمشخص شوند.

•سپس یک مقدار مورد انتظار یا سه امتیازی محاسبه می شود.

•مقدار منتظره برای متغیر تخمینی (اندازه)S,را می توان به عنوان یک میانگین ارزیابی شده از تخمین های خوشینانه (S_{opt}),دارای احتمال بالا(S_m)و تخمین بدبینانه (S_{pess})ارزیابی کرد .به طور مثال:

•Pessimistic

•most likely

•optimistic

$$S=(S_{opt} +4S_m+S_{pess})/6$$

•بیشترین اعتبار را به تخمینی که احتمال زیادی دارد داده و توزیع احتمال بتا را دنبال می کند.

•وقتی مقدار مورد انتظار متغیر تخمین معین شد FPوLOC به کار گرفته می شوند .

•آیا این تخمین ها درست هستند؟

نمي توان مطمئن بود

مثال ۱: برآورد مبتنی بر خطوط برنامه(LOC)

۱. **وضعیت اولیه دامنه(نامحدود)**

- به عنوان نمونه ای از فنون تخمین LOCوFP,بسته نرم افزار باید در نظر بگیرید که قرار است برای یک برنامه کاربردی طراحی با کمک کامپیوتر (CAD)برای اجزای مکانیکی تولید شود.
- بازنگری مشخصات سیستم نشانگر این است که نرم افزار قرار است روی یک ایستگاه 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.
- نرم افزار CADاطلاعات هندسی دو و سه بعدی مهندس را دریافت می کند.

۲. سپس اقدامات اولیه را بسط داده تا **جزئیات اساسي و محدوديتهاي كمي** ایجاد شوند

- مهندس با سیستم ارتباط برقرار کرده و CADرا از طریق رابط کاربردی کنترل می کند که مشخصات یک طراحی رابط متقابل ماشین – انسان را به خوبی نشان میدهد . یا اینکه اندازه و میزان پیچیدگی پایگاه داده ای CADباید چه باشد.

۳.به منظور دستیابی به اهداف مدنظر ,ما فرض می کنیم که پالایش بیشتری رخ داده و **عملکردهای** نرم افزاری زیر **شناسایی** شده است :

۱- تسهیلات کنترلی و رابط کاربر(UICF)
۲- تحلیل هندسی دو بعدی (2DGA) ۳- تحلیل هندسی سه بعدی (3DGA)
۴- مدریت پایگاه اطلاعاتی (DBM)
۵- تسهیلات نمایشی گرافیک کامپیوتر(CGDF)
۶- کارکرد کنترل دستگاه های جانبی (PCF)
۷- پیمانه های تحلیل طراحی(DAM)

computer-aided design=cad

نام درس:مهندسی نرم افزار ۱

۴- ایجاد جدول برآورد و تخمین برای روش تعداد خطوط برنامه

با استفاده از معادله : $S=(S_{opt} + (4S_m+S_{pess})/6$

ستون تخمین LOC را ایجاد مي كنيم

براي تابع هندسي سه بعدي ، بهترين حالت ۴۶۰۰ ، در محتملترين حالت ۶۹۰۰ و در بدترين حالت ۸۶۰۰ مقدار منتظره برای تابع هندسی سه بعدی ۶۸۰۰ می شود .LOCسایر تخمین ها به شیوه ای مشابه مشتق می شود .با جمع بندی در ستون LOCتخمینی برآورد ۳۳۱۵۰ خط کد برای سیستم ADC ارائه می شود.

طبق سابقه سازماني توليد 620loc/pm پس =**۵۴ نفر ماه**

فانکشن	*تخمینLOC*
تسهیلات کنترلی و رابط کاربر(UICF)	۲۳۰۰
تحلیل هندسی دو بعدی (2DGA)	۵۳۰۰
تحلیل هندسی سه بعدی (3DGA)	۶۸۰۰
مدریت پایگاه اطلاعاتی (DBM)	۳۳۵۰
تسهیلات نمایشی گرافیکي(CGDF)	۴۹۵۰
کارکرد کنترل دستگاه های جانبی (PCF)	۲۱۰۰
پیمانه های تحلیل طراحی(DAM)	۸۴۰۰
تعداد خطوط برآورد شده برنامه	۳۳۲۰۰

مثال ۲: برآورد مبتنی بر امتیاز کارکردی (FP)

•تفکیک سازی در مورد تخمین مبتنی بر FPعلاوه بر کارکردهای نرم افزاری روی مقادیر دامنه اطلاعات نیز متمرکز می شود.

•بایادآوری جدول محاسبه امتیازات عملکردی ,برنامه ریز پروژه ,ورودی ها ,خروجی ها,درخواستها,فایلهاو رابطه های خارجی را در مورد نرم افزار CADتخمین می زند.

•در مورد اهداف این تخمین ,عامل ارزیابی پیچیدگی به طور متوسط در نظر گرفته می شود .

•نهایتا میزان تخمین زده شده FPبه دست می آید

•طبق سابقه سازماني توليد 6.5fp/pmپس =**۵۸ نفر ماه**

مقادیر حوزه اطلاعاتی	خوش بینانه	متوسط	بد بینانه	شمارش برآوذد	وزن	امتیاز عملیاتي
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	4	۸۸
تعدادپرونده ها	۴	۴	۵	۴	۱۰	۴۲
تعداد رابط های خارجی	۲	۲	۳	۲	۷	۱۵
تعداد کل						۳۲۰

$FP_{estimated}$ = count-total x [0.65 + 0.01 x Σ (F_i)] =375

۲–برآورد مبتنی بر فرایند

❖**رايج ترين تكنيک تخمين زدن** يک پروژه ،**بر مبناي فرآيندي** است که استفاده مي شود. يعني **فرآيند به چند مجموعه وظايف نسبتا کوچك تقسيم** مي شود و نيروي کار لازم براي هر فرآيند تخمين زده مي شود.

❖مانند روشهاي مبتني بر مبناي مسئله ، اين تخمين هم با نمايش وظايف بدست مي آيد.

❖وقتی کارکردها و فعالتهای فرایندی اعلام شدند **برنامه ریز نیروی کاری را که (مثلا نفر-ماه)برای رسیدن به فرایند هر کارکرد نرم افزاری لازم است براوردمی کند** میزان متوسط نیروی کار(هزینه/واحد نیروی کار)در نیروی کار تخمین زده شده برای هر فرایند به کار گرفته می شود. این احتمال وجود دارد که میزان نیروی کاردر هر مورد متفاوت باشد.

❖کارکنان ارشد شدیدا در فعالیتهای اولیه دخیلند و معمولا از کارکنان سطح پایین تر که در کارهای طراحی ,ایجاد کد و آزمون اولیه دخالت دارند گرانتر می باشند.

❖هزینه و نیروی کار برای هر عملیات و فرایند نرم افزاری به عنوان آخرین مرحله استفاده می شود.

فعالیتهای چارچوب مشترک فرآیند / فعالیتهای	ارتباط با مشتری	طرح ریزی	تحلیل ریسک	مهندسی
فعالیتهای مهندسی نرم افزار				
کارکردهای محصول				
ورودی متن				
ویرایش و قالب بندی				
اصلاح خودکار				
آرایش صفحه و صفحه بندی				
شاخص بندی خودکار وToc				
مدیریت فایل				
تولید مستندات				

مثال : برآورد مبتنی بر فرایند

•به منظور تشریح استفاده از تخمین مبتنی بر فرآیند نرم افزار CADرا در نظر می گیریم (مانند مثالهای قبل)

•بر اساس میانگین نیروی کار به منظور ۵۰۰۰دلار در ماه,هزینه تخمین زده کل پروژه ۲۳۰۰۰۰دلار بوده و نیروی کار تخمینی نفر در ماه میباشد.اگر بخواهید میزان نیروی کار با هر فرایند نرم افزاری یا مهندسی مرتبط شده و جداگانه محاسبه می شود.

•نیروی کار کل تخمینی در مورد دامنه های نرم افزاری CADاز حداقل ۴۶ نفر در ماه(روش فرآیند) تا حداکثر ۵۸نفر(روش fp) میرسد. میانگین براورد از کل سه روش ۵۳نفر –ماه است.حداکثر **میزان تغییر از متوسط تخمین زده شده** تقریبا ۱۳درصد است.

به خاطر اینکه میزان نیاز به متخصص هر فعالیت فرآیندجداگانه حساب مي شود،هزینه بر اساس کار دقیقتر محاسبه مي شود

•وقتی میزان سازگاری بین تخمین ها ناچیز است چه اتفاقی رخ داده؟ارزیابي مجدد اطلاعات

Activity →	CC	Planning	Risk analysis	Engineering		Construction release		CE	Totals
Task →				Analysis	Design	Code	Test		
Function									
↓									
UICF				0.50	2.50	0.40	5.00	n/a	8.40
2DGA				0.75	4.00	0.60	2.00	n/a	7.35
3DGA				0.50	4.00	1.00	3.00	n/a	8.50
CGDF				0.50	3.00	1.00	1.50	n/a	6.00
DBM				0.50	3.00	0.75	1.50	n/a	5.75
PCF				0.25	2.00	0.50	1.50	n/a	4.25
DAM				0.50	2.00	0.50	2.00	n/a	5.00
Totals	0.25	0.25	0.25	3.50	20.50	4.50	16.50		46.00
% effort	1%	1%	1%	8%	45%	10%	36%		

CC = customer communication CE = customer evaluation

•تخمین های بسیار متفاوت را اغلب به یکي از دلایل زیر بدست مي ایند

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

۲- اطلاعات بهره وری مورد استفاده برای فنون تخمین مبتنی بر مساله ,از نظر برنامه مناسب نیستند یا درست مورد استفاده قرار نگرفته اند.

ستونها:فعالیتهاي فرآیند
سطرها:تجزیه مساله(توابع)
خانه ها:هزینه هرتابع به ازاي نوع فعالیت فرآیند

PM(نفر)

مدل های برآورد تجربی

يك مدل تخمين براي نرم افزار كامپيوتر، از **تجارب حاصل از فرمولها** براي پيش بيني فعاليت به **صورت تابعي از LOC يا FP** استفاده مي شود، مقادير LOC يا FP طبق روشهاي برآورد گفته شده بدست مي آيند

اطلاعات تجربی که اکثر مدلهای تخمینی را پشتیبانی می کنند از تعداد محدودي پروژه بدست مي آيند.به ایندلیل هیچ مدل تخمینی برای تمام رده های نرم افزاری مناسب نیست ودر همه محیطهای تولیدی مناسب نمی باشد .لذا نتایج این مدلها با تعدیل استفاده شوند

ساختار مدل های برآورد

یک مدل تخمینی عادی بااستفاده از تحلیل بازگشتی یا رگرسیون بر روی اطلاعات جمع آوری شده از پروژه های قبلی ,بدست می آید ساختار کلی چنین مدلهایی به شکل زیر است

$$E = A + BX(ev)^c$$

که درآن A,B,C ثابت بدست آمده به صورت تجربی هستند E نیروی کار برحسب نفر-ماه و evمتغیر تخمین (fp یا LOC) می باشد

$$\text{effort} = \text{tuning coefficient} * \text{size}^{\text{exponent}}$$

$E = 5.2 \times (KLOC)^{0.91}$	Walston-Felix model
$E = 5.5 + 0.73 \times (KLOC)^{1.16}$	Bailey-Basili model
$E = 3.2 \times (KLOC)^{1.05}$	Boehm simple model
$E = 5.288 \times (KLOC)^{1.047}$	Doty model for KLOC > 9

FP-oriented models have also been proposed. These include

$E = -13.39 + 0.0545\ FP$	Albrecht and Gaffney model
$E = 60.62 \times 7.728 \times 10^{-8}\ FP^3$	Kemerer model
$E = 585.7 + 15.12\ FP$	Matson, Barnett, and Mellichamp model

مدل COCOMO

COnstructive COst MOdel.

•باری بوهوم در کتاب خود "اقتصاد مهندسی نرم افزار" یک سری مدلهای تخمین نرم افزار معرفی میکند که دارای نام کوکومو هستند(مدل هزینه ساختاری)

•مدل اصلی کوکومو از همه بیشتر مورد استفاده قرار گرفته و مدلهای تخمین هزینه در صنعت مورد بحث قرار گرفتند.

•COCOMO II مدل تخمین تکامل یافته کوکومو بود که زمینه هاي زیر را مورد توجه قرار داد :

۱. **مدل ترکیبی کاربردی:**
این مدل در طول مراحل اولیه مهندسی نرم افزار به کار میرود.هنگامی که **نمونه های اولیه** رابط کاربر مدلسازی میشوند با در نظر گرفتن رابطه متقابل سیستم و نرم افزار **بر اورد عملکرد و ارزیابی تکامل فناوری** که داراي اهمیت هستند

۲. **مدل مرحله اولیه طراحی**
وقتی بکار میرود که نیاز مندیها مشخص و معماری مقدماتی نرم افزار ایجاد شده است.

۳. **مدل مرحله آخر معماری**
در طول ساخت نرم افزار بکار میرود

•مدلهاي کوکومو ۲ ،نیازمند اطلاعات اندازه گیری است.سه گزینه از نظر اندازه به عنوان بخشی از سلسله مراتب مدل مهیا شده اند **امتیاز شی(OP)** و**امتیاز کارکردی(FP)** و **خطوط کد منبع(LOC)**

• امتیاز شی یک معیار نرم افزاری غیر مستقیم است که با استفاده از شمارش **۱- تعداد صفات نمایش**(پنجره ها) ۲-تعداد **گزارشات** و ۳ - **تعداد اجزایی که احتمالا برای ساخت برنامه لازمند (3GL)**محاسبه میشود.

•هر مورد شی در یکی از سطوح پیچیدگی(ساده-متوسط-مشکل) گروهبندی میشود.در اصل پیچیدگی تابعی از۱ - تعداد و منبع جدول اطلاعات خادم/مخدوم است که برای تولید صفحه یا گزارش لازم بوده و ۲-تعداد دیدگاهها یا بخش ها بعنوان بخشی از صفحه یا گزارش ارایه شده اند.

•فرمولهاي پيداكردن تعداد

•"امتياز جديد شی:"

درصد استفاده مجدد

NOP=(امتياز شي) x [(100 - %reuse)/100]

•برای بدست آوردن تخمین نیروی کار بر اساس مقداربالا باید "میزان بهره وری" را بدست آورد:

PROD=NOP/PM(نفرماه)

•"تخمين نيروی كار پروژه:"

estimated effort=NOP/PROD

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

وزن پیچیدگی برای انواع اشیا

قابلیت /تجربه	خیلی کم	کم	متوسط	بالا	خیلی بالا
قابلیت/بلوغ محیط	خیلی کم	کم	متوسط	بالا	خیلی بالا
بهره وری(PROD)	۴	۷	۱۳	۲۵	۵۰

نرخ های بهره وری برای امتیازات اشیا(PROD)

معادله نرم افزار(فرمولها)

معادله نرم افزار یک **مدل دینامیک چند متغیره** است که **توزیع نیروی کار معینی** را در **طول حیات پروژه** تولید نرم افزار فرض میگیرد.

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

مدل تخمینی:

E = نیروی کاری فرد در ماه یا سال

T = مدت پروژه به سال یا ماه

B = عامل مهارتهای ویژه

P = پارامتر بهره وری که منعکس کننده موارد زیر است

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

•معادله نرم افزاری دارای دو پارامتر مستقل است:

۱-تخمینی از اندازه(LOC)

۲-نشانگری از مدت پروژه به ماهها یا سالهای تقویمی

•حد اقل زمان تولید

$$t_{min} = 8.14\ (LOC/P)^{0.43} \text{ in months for } t_{min} > 6 \text{ months}$$

$$E = 180\ Bt^3 \text{ in person-months for } E \geq 20 \text{ person-months}$$

تصمیم گیری ساخت /خرید

در حوزه های متعددی از برنامه های کاربردی مقرون به صرفه تر است که بجای تولید نرم افزار کامپیوتر آن را خریداری کنیم .
مدیران مهندسی نرم افزار با تصمیم گیری در **خرید یا تولید** مواجه هستند
گزینه هاي در دسترس:

۱- ممکن است نرم افزار *بصورت آماده* خریداری شود.
۲- اجزای نرم افزاری بصورت "با *تجربه کامل"یا "با تجربه نسبی"دست* آیند و سپس اصلاح شده و یکپارچه گردند تا نیازها را بر آورده سازند.
۳- ممکن است نرم افزار توسط *پیمانکاری* خارج از سازمان طبق سفارش ساخته شده باشد تا نیازهای خریدار را مرتفع سازد.

در مورد محصولات نرم افزاری گرانتر رهنمودهای زیر را میتوان بکار گرفت:

۱- مشخصاتی برای کارکرد و عملکرد نرم افزار مطلوب ارایه 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 مشخصه های شما جور میشود.
ب : اجزای نرم افزاری قابل استفاده مجددی انتخاب میکند که در ساخت برنامه مورد نیاز شما را یاری میکنند.
۴- شبکه مقایسه ای تشکیل میدهد که نمایانگر مقایسه سر به سر عملیات اصلی است.
۵- هر بسته نرم افزاری یا مولفه را بر اساس کیفیت محصول گذشته پشتیبانی فروشنده راهنمای محصول و... ارزیابی میکند.
۶- با سایر کاربران نرم افزار تماس گرفته نظراتشان را جویا میشود

در مورد تصمیم نهايي خريد يا توليد بر مبناي شرايط زير صورت مي گيرد :

۱.آيا تاريخ تحويل محصول زودتر از تاريخ توليد داخلي است؟
۲.آيا هزينه سفارش و خريد ، كمتر از هزينه توليد است؟
۳.آيا هزينه پشتيباني خارجي ، كمتر از هزينه پشتيباني داخلي(تيم ما) است؟

ایجاد یک درخت تصمیم گیری

مراحل توصیف شده فوق را میتوان با استفاده از فنون آماری در تصمیم گیری(مثل درخت تصمیم گیري) ارزیابی کرد

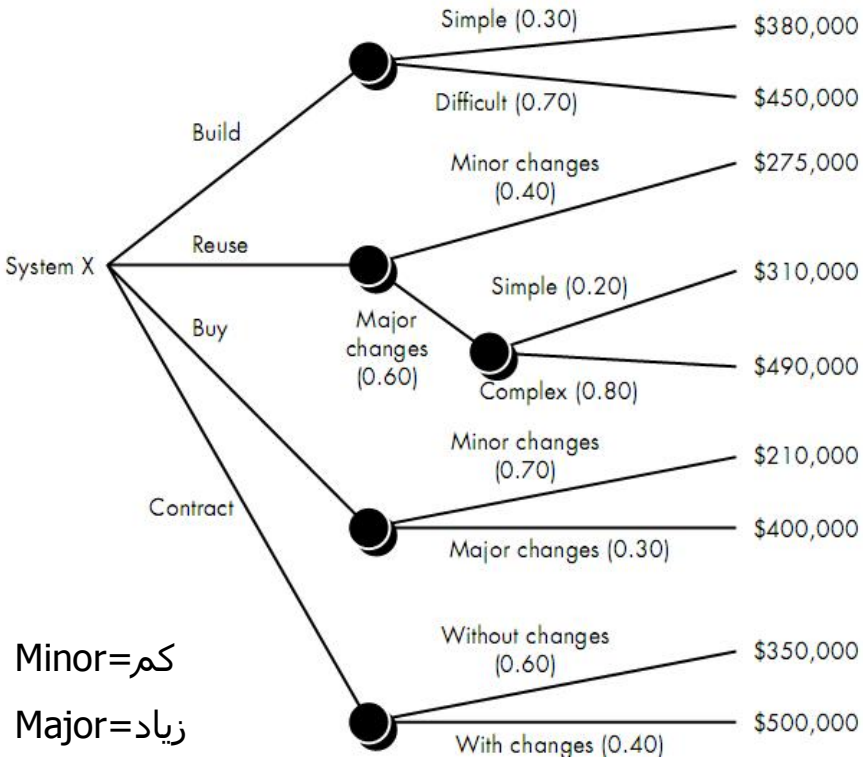
حالتهاي درخت تصمیم گیري فرضي براي ساخت سیستم x

۱-ایجاد

۲-استفاده از مولفه هاي قبلي

۳-خرید و اصلاح نرم افزار

۴-قرارداد با سایر تیم ها



Minor=کم

Major=زیاد

هزینه مورد نظر= (مسیر احتمالی)*(هزینه مسیر تخمینی)

هزینه هاي منتظره :

$$\text{expected cost}_{\text{reuse}} = 0.40\ (\$275K) + 0.60\ [0.20(\$310K) + 0.80(\$490K)] = \$382K$$

$$\text{expected cost}_{\text{buy}} = 0.70(\$210K) + 0.30(\$400K)] = \$267K$$

$$\text{expected cost}_{\text{contract}} = 0.60(\$350K) + 0.40(\$500K)] = \$410K$$

کمترین هزینه ، هزینه خرید است

استفاده از منابع خارجی

❖**منابع خارجی** از نظر مفهوم بسیار ساده هستند.کارهای طراحی نرم افزار با **شخص ثالثی قرارداد بسته میشوند** که این شخص کمترین هزینه و بالاترین کیفیت کار را انجام دهد.

❖**سطوح استفاده از منابع خارجي:**

۱-سطح استراتژيك(راهبردي):مديرتجاري تصميم مي گيرد:آيا مي توان بخش عمده اي از روند توليد را با عقد قرار داد انجام داد

۲-سطح تاكتيكي:مدير پروژه تصميم مي گيرد : آيا امكان انجام بخشي يا تمام كار توسط ديگران ميسر است؟

❖**سود:**معمولا پس اندازهای هزینه ای با کاهش تعداد افراد مربوطه به کار نرم افزاری حاصل میگردد که از آنها حمایت

❖**ضرر:** شرکت کنترل نرم افزاری را که به آن نیاز دارد از دست میدهد.

ابزارهای خود کار بر آورد

گرچه ابزارهای تخمین خودکار بسیاری وجود دارند اما همه مشخصه های عمومی یکسانی نشان داده و همه شش عملکرد کلی زیر را اجرا میکنند:

۱- اندازه بندی موارد قابل تحویل پروژه.
۲- انتخاب فعالیتهای پروژه.
۳- پیش بینی میزان کارکنان.
۴- پیش بینی نیروی کاری نرم افزار.
۵- پیش بینی هزینه نرم افزار.
۶- پیش بینی جداول زمانی کار.

- به علت اختلافات تخمينهاي ابزارهاي خودكار لذا خروجي ابزارهاي خودكار برآورد به عنوان امتياز اطلاعاتي كه از روي آنها تخمينها هدايت مي شوند استفاده شوند و نه به عنوان منبع تخمين

فصل ۶: مدیریت و تحلیل ریسک

مقدمه

- **تحلیل و مدیریت مخاطرات چیست؟**

تحلیل ریسک (مخاطره) و مدیریت عبارت است از یک سری از اقدامات که یک تیم نرم افزاری کمک می کند تا عدم قطعیت (عدم اطمینان)را دریافته و آنرا مدیریت نمایند. مشکلات زیادی میتوانند برای یک پروژه نرم افزاری ایجاد اشکال نمایند.

ریسک یک مشکل بالقوه است ممکن است اتفاق بیفتد و ممکن است اتفال نیفتد اما صرفنظر از پیامد آن، بهتر است که آن را شناسایی نماییم، احتمال وقوع آن را ارزیابی کنیم.

- **چه کسی این کار را انجام می دهد؟**

هر کسی که درگیر فرآیند نرم افزاری است در تحلیل و مدیریت مخاطرات مشارکت دارد، يعني شامل :مدیران،مهندسین و مشتریان نرم افزار است

- **چه اقداماتی باید انجام داد؟**

اولین گام : **شناسایی ریسک**= آگاهی از اینکه چه اشکالاتی ممکن است پیش آید.

دومین گام : باید هر ریسکی **تحلیل** گردد تا احتمال وقوع آن و میزان خسارتی را که در صورت وقوع به بار می آورد تعیین شود. به محض اینکه این اطلاعات مشخص شدند، خطرات بر اساس احتمال وقوع و تاثیر **طبقه بندی** می شوند.

سومین گام : طرحی پیاده می شود تا خطراتی که احتمال وقوع آنها زیاد است وتاثیر آنها بیشتر است **مدیریت** نماید.

- **محصول کاري چیست؟**

طرحي براي متعادل سازي،کنترل و مدیریت ریسك. (RMMM) یا مجموعه صفحات حاوي اطلاعات ریسك

- **چگونه می توانیم اطمینان حاصل کنیم که این کار را درست انجام داده ایم؟**

ریسک هایی که تجزیه و تحلیل و کنترل می شوند باید حاصل مطالعه کامل افراد محصول فرآیند و پروژه باشند.**RMMM** با پیشرفت پروژه بایست بازبیني شود تا اطمینان حاصل شود ریسکها **به روز** نگه داشته مي شوند.طرح های احتمالی برای کنترل و مدیریت ریسک باید واقعی باشند.

- عدم قطعیت باعث ریسك مي شود

❑**یک تعریف عقلانی و مفهومي از ریسک چیست؟**

۱- **ریسک به وقایع آینده ارتباط دارد.**

امروز و دیروز در خارج از محدوده توجه قرار دارند، زیرا ما در زمان حال آنچه را که در گذشته با اعمال خود کاشته ایم درو می کنیم. سئوالی که مطرح می شود این است که پس آیا ما می توانیم با تغییر اعمال امروزمان فرصتی برای موقعیت متفاوت و شاید بهتر در آینده ایجاد نماییم.

۲- **ریسک مستلزم تغییر است.** همچون تغییر در ذهن، ایده ، اعمال و یا مکان ها ....

(۳- درکتاب پرسمن) ریسک **مستلزم انتخاب** و عدم قطعیتی است که انتخاب ایجاب می نماید. بنابراین ریسک همچون مرگ و مالیات یکی از معدود امرهای مسلم زندگی است.

❑**سه تعریف زیر بناتی ضمنی چارته از ریسك در رابطه با مهندسي نرم افزار:**

۱.برای ما آینده اهمیت دارد :جلوگیري از خروج پروژه از مسیرش

۲.تغییر برای ما اهمیت دارد:چگونگي تاثیر تغییرات بر روي پروژه (تغییرات در نیازها-فناوري توسعه و ...)

۳.باید با انتخاب ها دست و پنجه نرم کنیم:انتخاب روشها و ابزارها،افراد ،میزان تاکید بر افراد و....

❑**نظریه پیتر دراکر :**

همانگونه که تلاش برای حذف خطر **بی ثمر** است، و تلاش برای کاستن آن سوال برانگیز می باشد، ضروری است خطراتی را که برای آنها وقت صرف می کنیم خطرات بجایی باشند. قبل از اینکه خطرات بجایی را که در طی یک پروژه نرم افزاری به آنها خواهیم پرداخت **شناسایی** نماییم، حائز اهمیت است که کلیه خطراتی را که هم برای مدیران و هم برای کارورزان مشهود است شناشایی نماییم.

راهبرد واکنش بر ریسک

راهبردهاي ريسك واكنشي به نام "مدرسه مديريت ريسك Indiana Jones " ناميده مي شوند. در فيلمهاي سينمايي كه ايندياناجونز وجود دارد او با خطرات عمده اي روبروست و هميشه مي گويد : "نگران نباشيد راهي پيدا خواهم كرد!"

يك مدير پروژه متوسط در برابر مشكلات ، ايندياناجونز نيست و اعضاي تيم مورد اعتمادش نمي باشند

اکثریت تیم های نرم افزاری منحصراً بر روی راهبردهای ریسک واکنش پذیر تکیه دارند.
مهمتر از همه اینکه یک راهبرد واکنش پذیر پروژه را برای خطرات احتمالی تحت **کنترل(نظارت)** دارد. منابعی برای مقابله با آنها در نظر گرفته می شود. تا زمانی که تبدیل به **مشکل واقعی** شدند از آنها استفاده شود. معمولاً تیم نرم افزاری در مورد ریسک کاری انجام نمی دهند تا اینکه مشکلی پیش نیاید. در این حال تیم وارد عمل شده و سریعاً برای **حل مشکل** اقدام می نماید. این وضعیت غالباً «**وضعیت پرواز آتشی(درست:جنگ با آتش)**» نامیده می شود. وقتی که این کار با شکست مواجه می شود «**مدیریت بحران**» کنترل را بدست می گیرد که در این حال پروژه با خطر واقعی مواجه می گردد.

راهبرد واکنشهای پیش کنش (غیر واکنشی)در ریسک

fire fighting **Flight**

راهبردهوشمندانه پیش کنش خیلی پیش از آنکه کار فنی آغاز گردد شروع می شود. خطرات بالقوه(داراي شانس وقوع) **شناسایی** شده، احتمال و تاثیر آنها مورد **ارزیابی** قرار گرفته و سپس بر اساس اهمیت **طبقه بندی** می شوند. سپس تیم نرم افزاری **طرحی** را برای کنترل ریسک ارائه می دهد. هدف اولیه **جلوگیری از بروز ریسک** (خطر) است اما از آنجایی که تمام خطرات اجتناب ناپذیز هستند، تیم یک طرح احتمالی ارائه می دهد تا بتواند به یک روش کنترل شده و موثر واکنش نشان دهد.

ریسک های نرم افزار

در تعريفهاي ريسك همواره داراي دو مشخصه مشابه وجود دارد:

- **عدم قطعيت**: رويدادی که مشخص می کند آیا ریسکی به وقوع خواهد پیوست یا خیر: یعنی هیچ ریسکی که احتمال وقوع آن ۱۰۰% باشد وجود ندارد.
- **خسارت** : اگر ريسک يک واقعيت باشد، پيامد ها و يا خسارات ناخواسته ای بوقوع خواهد پيوست

زمان تحليل ريسك ، تعيين عدم اطمينان(قطعيت) به صورت كمي و درجه فقدان ناشي از هر ريسك اهميت زيادي دارد.

نام درس:مهندسی نرم افزار ۱

❑دسته هاي مختلف ريسك ها

❑ريسک پروژه:

بر برنامه ريزي پروژه تاثير دارند. اگر ريسکهاي پروژه به واقعيت بپيوندند اين احتمال وجود دارد که برنامه پروژه با اشتباه مواجه شود و قيمتها افزايش يابند. ريسک های پروژه، بودجه، برنامه زمان بندی، پرسنل، منابع، مشتری بالقوه و مشکلات مربوط به نيازمندی ها و تاثير آنها را بروی پروژه نرم افزاری شناسايی می کنند

❑خطرات فنی(ريسکهاي تکنيکي) :

وقتی که خطر فنی به واقعيت می پيوندد، پياده سازی دشوار و يا غير ممکن می گردد. خطرات فنی مشکلات بالقوه مربوط به طراحی، پياده سازی، رابط ، تعين صحت و نگهداری را شناسايی می کنند. بعلاوه ابهام مربوط به ويژگی، عدم قطعيت فنی، زوال فنی و فناوری «پيشرو» از عوامل و فاکتور مربوط به ريسک هستند. خطرات فنی بدين جهت بوقوع می پيوندند **که حل مشکل از آنچه که ما تصور می کنيم دشوارتر می باشد.**

❑خطرات تجاری

عوامل ريسکهاي تجاري:

•مثال :ساختن فيلم

۱.ساخت يک محصول و يا سيستم بسيار عالی که هيچکس واقعاً طالب آن نيست. (ريسک بازار)
۲.ساخت محصول که ديگر مناسب راهبرد تجاری کلی برای شرکت مزبور نمی باشد. (ريسک راهبردی)
۳.ساخت محصولی که نيروهای فروش نمی دانند چگونه آن را بفروشند (ريسک فروش)
۴.از دست دادن حمايت مديريت ارشد بدليل تغيير نقطه عطف (ديدگاه) و يا تغيير افراد (ريسک مديريتي)
۵.از دست دادن بودجه و يا تعهد پرسنل (ريسک بودجه)

❑ريسکهاي قابل پيش بيني(شناخته شده)

آنهايي هستند که مي توانند بعد از ارزيابی دقيق پروژه ،خطرات فني و خطرات تجاري و آشکار شدن منابع قابل اعتماد (مثل ثبات تاريخ تحويل يا عدم ناقص بودن مستندات) مشخص شوند.ريسکهاي قابل پيش بيني هستند که از تجربه های پروژه های قبلي استخراج مي شوند (مثلا ارتباط ضعيف با مشتري)

❑ريسکهاي غير قابل شناسايي

ريسکهايي که به وقوع مي پيوندند و شناسايي آنها از قبل بسيار دشوار است (مثل جوکر در بازي ورق)

شناسایی ریسک

شناسایی ریسک عبارت است از یک تلاش نظام مند برای تعین تهدیدهایی که متوجه **طرح پروژه** هستند (برآوردها، برنامه ها، تخصیص منابع و غیره) و با شناسايي ريسكهاي قابل پيش بيني مدير پروژه اولين گام را در راه اجتناب از آنها و درصورت امکان کنترل آنها به کار مي برد

❑انواع كلي ريسک در مرحله شناسايي

۱. **ریسک های محصول ویژه(خاص محصول)** : را تنها می توان توسط کسانی تشخیص داده شوند که **درک روشنی** از فناوری ،مردم و محیطی که مختص پروژه دارند

۲. **ریسک عمومی** : ریسک های عمومی تهدید بالقوه ای برای هر نوع پروژه نرم افزاری می باشند.

برای **شناسایی ریسک های محصول ویژه(خاص محصول)**، طرح پروژه و گزارش نرم افزاری از حوزه مورد(گزارش دامنه نرم افزار) بررسی قرار گرفته و پاسخی برای سوال مقابل داده می شود . "چه ویژگی های خاصی از این محصول می تواند طرح پروژه ما را به خطر اندازد؟"

یک روش کاربردی برای شناسایی ریسک ها عبارت است از **تهیه یک چک لیست** حاوی موارد ریسک از چک لیست می توان برای شناسایی ریسک و تمرکز (نقطه عطف) به مسائل فرعی شناخته شده و قابل پیش بینی ریسک استفاده کرد.

❖طبقه هاي ريسك

- **اندازه محصول(PS)**:ریسک هایی که مرتبط به اندازه کلی نرم افزار شناخته شده و یا تغییر یافته هستند.
- **تاثیر تجارت(BU)** :ریسک هایی که مربوط به محدودیت های ناشی از مدیریت و یا بازار تجاری است.
- **ویژگی های مشتری(CM)** : ریسک های مربوط به پیچیدگی مشتری و توانایی سازندگان برای ایجاد ارتباط زمانبندي شده با او
- **تدوین فرآیند(PD)**: ریسک های مربوط به میزان تعریف فرآیند نرم افزار که توسط سازمان توسعه دهنده دنبال می شود.
- **محیط توسعه(EV)**:ریسک های مربوط به در دسترس بودن و کیفیت ابزار هایی که برای ساخت محصول مورد استفاده است.
- **فنآوری ساخت(TE)**: ریسک های مربوط به پیچیدگی سیستم ساخته شده و «تازگی» فنآوری که سیستم با آن ساخته مي شود.
- **اندازه و تجربه کارکنان(ST)** : ریسک های مربوط به تجربه پروژه فنی کلی مهندسین نرم افزاری که کار را انجام می دهند.

ارزیابی(؟)(شناسایی) ریسک کلی پروژه:

سوالات زیر از گروهي داده هاي ريسك نتيجه گيري شده اند كه توسط مديران با تجربه در نقاط مختلف دنيا بدست آمده اند و برحسب اهميت مرتب شده اند:

۱.آیا مدیران نرم افزار و مشتری رسماً اقدام به حمایت از پروژه نموده اند؟

۲.آیا مصرف کنندگان نهایی در خصوص پروژه و سیستم / محصول ساخته شده اظهار نظر کرده اند؟

۳.آیا تیم مهندسی نرم افزار و مشتریان آنها نیازمندیها را کاملاً درک می نمایند؟

۴.ایا مشتریان کاملاً در تعریف نیازمندیها نقش دارند؟

۵.آیا مصرف کنندگان نهایی انتظارات واقع بینانه دارند؟

۶.آیا حوزه پروژه ثابت(دامنه پایدار) است ؟

۷. آیا تیم مهندسی نرم افزار ترکیب درستی از مهارت ها دارند؟

۸. آیا نیازمندیها پروژه ثابت است؟

۹. آیا تیم پروژه تجربه ای در خصوص فنآوری در حال انجام دارند ؟

۰۱. آیا تعداد افراد تیم پروژه برای انجام کار کافی هستند؟

۱۱. آیا کلیه مشتریان/مصرف کنندگان در خصوص اهمیت پروژه و در خصوص شرایط سیستم/ محصول ساخته شده با هم توافق دارند؟

•مثال ساخت فيلم براي بچه ها

در صورت منفي بودن جواب همه سؤالها پروژه به صورت كلي همه مراحل نظارت**بدون خطا** انجام مي شود

درجه ريسك پروژه به طور مستقيم **متناسب** است با **تعداد پاسخهاي منفي** به اين سؤالات

اجزاء و محرک های(هدایت کننده) ریسک:

بر طبق رساله نيروي هوايي آمريكا در جهت شناسايي و كاهش ريسكهاي نرم افزاري است. در اين روش مدير پروژه مي بايست **محركهاي ريسك** كه روي **اجزا ريسك** نرم افزاري را **تاثير** مي گذارند را شناسايي كند
تاثير محركهاي ريسك مي تواند به چهار دسته **جزئي،مرزي ، بحراني يا فاجعه آميز** باشد.

در اين رساله **اجزاي ريسك** عبارتند از :(كارايي-هزينه-حمايت و زمانبندي)

۱.ریسک عملکرد : مدیران عدم قطعیتی که به کمک آن نیازهای پروژه به هدف مورد نظر منطبق مي شوند.

۲.ریسک هزینه : میزان عدم قطعیتی که به کمک آن بودجه پروژه تامین می گردد.

۳.ریسک پشتیبان: میزان عدم قطعیتی که اصلاح، تطابق و تقویت نرم افزار ،آسان می شود.

۴.ریسک برنامه : میزان عدم قطعیتی که زمانبندی پروژه رعایت شود و محصول بموقع ارائه می گردد.

اگر ميزان عدم قطعيت ۰ باشد(قطعي) يعني ريسك و خطري وجود ندارد

- **تاثير محركهاي ريسك** كه روي **اجزا ريسك** :

۱. قابل صرف نظر (اغماض) – جزئي
۲. مرزي
۳. بحران
۴. فاجعه آميز

جدول ارزیابی تاثیرات

مؤلفه‌ها / طبقه		عملکرد	پشتیبانی	هزینه	زمانبندی
	1	Failure to meet the requirement would result in mission failure		Failure results in increasedcosts and scheduledel delays with expected values in excess of $500k	
فاجعه	2	Significant degradation To nonachievement of technical performance	Nonresponsive or unsupportable software	Significant financial shortages,budget Overrun likely	Unachievable LOC
	1	Failure to meet the requirement would degrade systemperformance to a point where mission success is questonable		Failure results in operational delays and/or increased costs with expected value of $1 to $100 k	
بحر	2	Some reduction in technical performance	Minor delay in softwere modifications	Some shortag of financial resources,possible overruns	Possible slippage in LOC
	1	Failure to meet therequiement would result I degradation of secondary mission		Costs, impacts,and/or recoverable schedule slips with expected value of $1 to $100 k	
مرز	2	Minimal to esmall reduction in technical performance	Responsive software support	Sufficient financial resources	Realitic,achievable schedule
	1	Failure to meet the requirement would creat inconveniene or nonoperational impact		Error results in minor cost and/or schedule impact with expected value of less than $1 k	
قابل اغما	2	No reduction in technical performance	Easily supportable software	Possible budget underru	Early achievable LOC

تصویر (تخمین) ریسک

برآورد ریسک ، که تخمین ریسک نیز نامیده می شود. سعي در تعیین مقدار ریسك به دو روش دارد :

۱-احتمال و یا امکان اینکه ریسک و مخاطره واقعیت یابد و روي دهد. ۲-پیامدها و مشکلاتی که در صورت روی دادن آن ریسك

برنامه ریز پروژه همراه با مدیران دیگر براي **تصویر (برآورد) ریسك چهار فعالیت** انجام مي دهند

۱.ایجاد مقیاسی که منعکس کننده احتمال مشاهده شده یک ریسک است.

۲.مشخص کردن پیامد های ریسک .

۳.برآورد تاثیر ریسک بر روی پروژه و محصول

۴.توجه به صحت و دقت کلی برآورد ریسک بطوری که هیچ درک نادرستی وجود نداشته باشد.

ساخت و توسعه یک جدول ریسک:

مخاطرات (ریسک‌ها)	طبقه	احتمال	تاثیر	م.ن.ت. ریسک
Size estimate may be significantly	PS	60%	2	
low	PS	30%	3	
Larger number of users than planned	PS	70%	2	
Less reuse than planned	BU	40%	3	
End users resist system	BU	50%	2	
Delivery deadline will be tightened	CU	40%	1	
Funding will be lost	PS	80%	2	
Customer wil change requirements	TE	30%	1	
Technology will not meet	DE	80%	3	
expectations	ST	30%	2	
Lack of training on tools	ST	60%	2	
Staff inexperienced				
Staff inexperienced				
Staff turnover will be high				

•یک تیم پروژه با فهرست بندی تمام ریسک ها (صرفنظر از فاصله زمانی وقوع آن) در اولین ستون جدول آغاز می گردد.

•هر ریسک در ستون دوم طبقه بندی شده است.

•احتمال وقوع هر ریسک در ستون بعدی است.میزان احتمال هر ریسک را اعضای تیم تخمین بزنند.(بررسي محركهاي ريسك غيرممكن/نامحتمل/محتمل/احتمال زياد)

•میانگین مقوله های هر چهار جزء ریسک (اجزا ریسك:عملکرد، پشتیبانی ، هزینه و برنامه زمانی) برای تعیین ارزش کلی تاثیر برآورده می گردد.

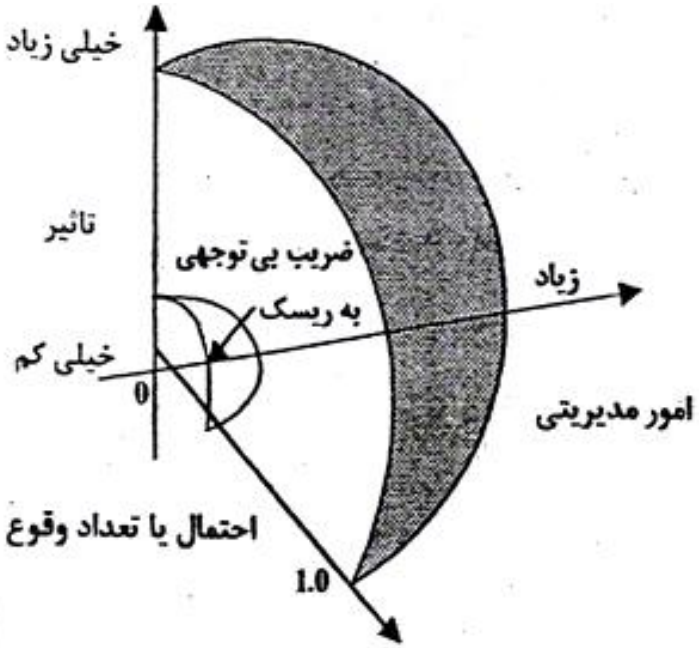
•با تکمیل ۴ ستون اول ، جدول بر اساس ستون احتمال و تاثیر مرتب مي شود

مقادیر تاثیر : ۱- فاجعه ۲- مرزی ۳- بحرانی ۴- قابل اغماض

م.ن.ت مدیریت – نظارت – تخفیف (RMMM)

CUمشتری و PSسایز پروژه TEکارکنان BUتجاری DEمحیط توسعه .

نام درس:مهندسی نرم افزار ۱



•مدیر پروژه جدول مرتب شده حاصل را مورد بررسی قرار داده و یک خط فرضی (برشی) تعریف می کند «خط فرضی» بیانگر این است که فقط به ریسک هایی که **در بالای خط قرار گرفته اند توجه بیشتری** می شود. ریسک هایی که در پایین خط قرار می گیرند مجدداً مورد بررسی قرار می گیرند تا اولویت بندی مرحله دوم انجام پذیرد

• تمام ریسک هایی که در بالای **خط فرضی** قرار می گیرند باید کنترل گردند. **ستونی که تحت عنوان م.ن.ت ریسک**(RMMM) (مدیریت – نظارت – تخفیف)نام گرفته دارای نشانگری است به طرف [۱]تخفیف و تعدیل ، [۲]کنترل و نظارت و [۳]مدیریت ریسک و یا بعبارت دیگر مجموعه ای از چند صفحه اطلاعات در خصوص ریسک که برای کلیه ریسک های که در بالای خط فرضی قرار گرفته اند در نظر گرفته شده اند .

ارزیابی میزان اثر ریسک

•بررسي نتايج وقوع ريسك تاثير مي گذارند: محز •عوامل موثر بر نتايج ريسك؟

۱.ماهيت ريسك : بيانگر **مشكلاتی** است كه در صورت **وقوع** ريسک بروز خواهند نمود.

۲.حوزه و محدوده ريسك : سختی آن را (**ميزان جدي بودن**)با پراكندگی و **توزيع** كلی(چه مقدار از پژوژه و چند مشتري دچار مشكل مي شوند) تركيب مي نمايد

۳.زمان ريسك: بيانگر اين است كه **چه وقت و برای چه مدتی** تاثير آن محسوس خواهد بود.

در بیشتر موارد یک مدیر پروژه خواهان این است که «خبر های ناخوشایند » هرچه زودتر برسد . اما در برخی مواقع ترجیح می دهند که این خبرها دیرتر برسند.

RMMM= Risk Mitigation, Monitoring and Management

❖**مراحل تعيين نتايج كلي ريسك (از ديد نيروي هوايي آمريكا)**

۱. براي هريك از اجزا ريسك مقدار احتمال متوسط وقوع را تعيين كنيد
۲. ميزان تاثير هر جز را با استفاده از معيار (**جزئي،مرزي ، بحراني يا فاجعه آميز**) تعيين كنيد
۳. تكميل جدول ريسك و تحليل نتايج

❖**ميزان در معرض ريسک قرار گرفتن** کلی را **RE** نامیده می شود و می توان با استفاده از رابطه زیر تعیین کرد.

$$RE = P^* C$$

P: احتمال وقوع یک ریسک
C: هزینه وارده به پروژه در صورت وقوع ریسک

❖**مثال**

•**تشخيص ريسك** :۷۰% از اجزا براي استفاده مجدد برنامه ريزي شده اند

•احتمال ريسك:۸۰%

•تعداد اجرا:۶۰ جزء اندازه هر جزء :100 loc هزينه هر LOC : 14$

حل:تعداد اجزايي كه در اثر ريسك دچار مشكل مي شوند : 60*30%=18
هزينه اجزاي دچار مشكل شده : 18*100*14=25200$
ميزان در معرض ريسك بودن RE=0.8 * 25200=20200$

تشخیص– (ارزیابی) ریسک

در این نقطه از فرآیند **مدیریت ریسک** ، ما یک مجموعه سه تایی به شکل زیر ایجاد نموده ایم .

$$[r_i, l_i, x_i]$$

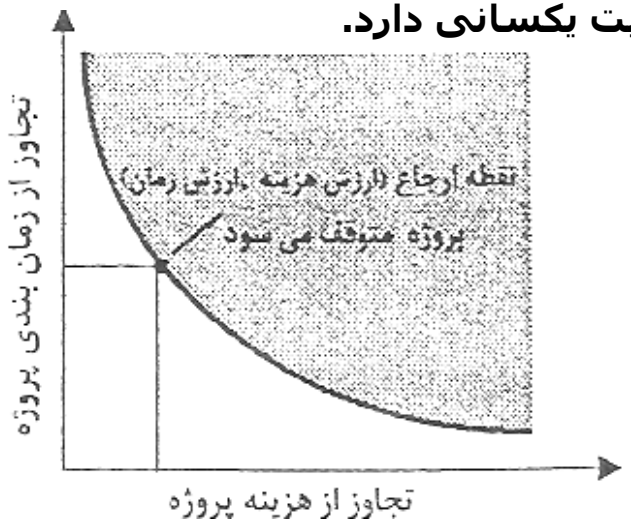
r_i: بیانگر ریسک
l_i: احتمال ریسک
x_i: بیانگر تاثیر ریسک

•سطوح ارجاعی(سطح مبنايي براي تشخيص ريسك):

برای اینکه ارزیابی سودمند واقع شود باید یک سطح ارجاعی برای ریسک تعریف نمود . در مورد بیشتر پروژه های نرم افزاری، اجزاء ریسک،(یعنی عملکرد، هزینه، پشتیبانی و زمان بندی)نیز بیانگر سطوح ارجاعی ریسک می باشند.یعنی سطحی برای کاهش کارايي و عملكرد، بالا رفتن هزينه، اشكال در پشتيباني و يا كم و كاستی در برنامه و زمانبندي كه باعث اتمام پروژه مي شوند وجود دارد.

اگر ترکیبی از ریسکها سبب ایجاد مشکلاتی شوند که منجر به ار حد **گذشتن** این **سطوح ارجاعی** گردد، **کار متوقف** خواهد شد.

سطوح ارجاعی ریسک دارای **نقطه منفردی** است که **نقطه ارجاعی و یا نقطه شکست** نامیده می شود. که در این نقطه تصمیم گیری **برای ادامه پروژه و یا ختم آن (مشکلاتی که خیلي بزرگ هستند) اهمیت یکسانی دارد.**



❖اقدامات زمان ارزیابی ریسک :

۱.سطوح ارجاعی ریسک را برای پروژه تعیین نمایید.

۲.سعی کنید بین هر **(r_i,l_i,x_i)** و **هر یک از سطوح ارجاعی** یک رابطه ایجاد کنید.

۳.مجموعه از **نقاط ارجاعی** را پیش بینی کنید و تعیین کننده یک ناحیه پایان است و بوسیله یک منحنی و یا نواحی عدم قطعیت محدود می شود.

۴.سعی کنید پیش بینی نمائید (**تاثیر**) **ترکیب های اجزاء ریسک ها** چگونه سطح ارجاعی را تحت تاثیر قرار خواهند داد.

پالايش ريسک

با گذشت زمان از آغاز پروژه نكات بيشتري در مورد پروژه و ريسك بدست مي آيد و ممكن است يك ريسك به مجموعه از ريسكها يا جزييات بيشتر پالايش شود كه هريك براي تخفيف(كاهش)،نظارت و مديريت ساده تر مي باشند. براي اين منظور مي توان ريسك را در قالب CTC (وضعيت-انتقال-نتيجه) نمايش داد

با شرايط اينكه <وضعيت> آنگاه بايد توجه داشت كه (احتمالاً)<نتيجه>

شرط

condition-transition-consequence=CTC

•**نحوه تدوين وظيفه عمومي در قالب CTC**

- ❑وضعیت فرعی ۱: اجزاء ویژه قابل استفاده مجدد توسط شخص ثالثی که آگاهی از استاندارد های طراحی ندارد ساخته شده اند.
- ❑وضعیت فرعی ۲:طرح استاندارد برای سطوح مشترک اجزاء هنوز به حالت یکپارچه در نیامده است و ممکن است با اجزاء خاص قابل استفاده مجدد موجود سازگاری پیدا نکند.
- ❑وضعیت فرعی ۳: اجزاء خاص قابل استفاده در **زبانی** بکار گرفته شده است که در محیط هدف مورد پشتیبانی قرار نگرفته است.

پالايش به جدا سازي ريسكهاي مرتبط كمك مي كند و ممكن است باعث ساده تر شدن تحليل شود

تخفیف(کاهش)، نظارت و مدیریت ریسک:

هدف تجزيه و تحليل ريسك **ارائه راهبردي** براي **پرداختن(مقابله) به ريسك** است

يک راهبرد موثر و کارا بايد سه مسئله را در نظر بگيرد:

- ❖**اجتناب از ريسک**
- ❖**کنترل ريسک**
- ❖**مديريت ريسک و برنامه ريزی احتياطی**

چنانچه تيم نرم افزاری يک روش **پيش کنش** برای ريسک انتخاب **کند،اجتناب** بهترين راهبرد خواهد بود. اين کار از طريق ارائه طرح برای تعديل و تخفيف ريسک امکان پذير است.و بايست براي کاهش عواملي که ريسك بالا محسوب مي شوند اقدامات پيشگيرانه صورت گيرد

مدير پروژه فاکتورهايي را **کنترل** و نظارت مي کند که مي توانند مشخص نمايند **هر ريسك با چه احتمالي** اتفاق مي افتد و نيز بر موثر بودن مراحل کاهش ريسك نظارت مي کند.

مديريت ريسک و برنامه ريزی احتياطی **فرض** مي کند که فعاليتهاي کاهش ريسك شکست خورده اند و **ريسك اتفاق افتاده** لذا مدير پروژه بايست برنامه ريزي لازم از جمله وجود نيروي پشتيبان را در نظر گرفته باشد.که گاها هزينه ها را بالا مي برد و شايد با افزايش ميزان ريسکها،مديريت ريسکها وقت گير هم باشد.لذا طبق **قانون ۲۰-۸۰**، با بررسي ۲۰ درصد ريسکها ۸۰ درصد ريسکها حل مي شوند. و وقت کمتري براي مديريت مورد نياز است

مخاطرات و ریسک های ایمنی:

ریسک تنها به پروژه نرم افزاری محدود نمی شود. ریسک ها ممکن است **پس از اینکه پروژه با موفقیت انجام شده** و تحویل مشتری گردید، بروز کنند. این ریسک ها غالباً مربوط به پیامدهای عدم موفقیت نرم افزار در زمان اجرا است.

خطاها معمولا حاصل از خطاهاي انساني اند

خطاهایی که در کنترل قراردادی سخت افزار می توان نادیده گرفت و بر آنها سرپوش گذاشت، بهنگام استفاده از نرم افزار نادیده گرفتن آنها بسیار دشوارتر خواهد بود.

ایمنی نرم افزار و تحلیل خطر : عبارتند از **فعالیت های مربوط به اطمینان از کیفیت نرم افزار** که به شناسایی و ارزیابی خطرات بالقوه ای می پردازد که می تواند نرم افزار را بطور منفی تحت تاثیر قرار داده و سبب بروز نقص در یک سیستم کامل گرد

طرح RMMM

•یک راهبرد مدیریت ریسک را می توان در طرح پروژه نرم افزاری گنجاند و یا مراحل مدیریت ریسک را می توان بصورت یک طرح تعدیل ، کنترل و یا مدیریت سازماندهی نمود. طرح RMMM **از تمام کارهایی که به عنوان بخشی از تحلیل ریسک انجام می پذیرد و توسط مدیر پروژه بعنوان بخشی از طرح پروژه کلی بکار می رود،گزارش مستند تهیه می نماید.**

•برخي تيمهاي نرم افزاري به جاي قالب RMMM ، **هر ريسك را به صورت مجزا با استفاده از صفحه اطلاعات ريسك(RIS)** مستند مي كنند، كه اين كار معمولا توسط سيستم بانك اطلاعاتي انجام مي شود

•با شروع پروژه و مستند شدن RMMM ،كاهش ريسك (اجتناب از ريسك) و مراحل نظارت(دنبال نمودن پروژه) شروع مي گردد

RMMM= Risk Mitigation, Monitoring and Management

risk information sheet (RIS)

قالب RIS در شکل زیر نمایش داده شده است.

Risk information sheet			
Risk ID: P02-4-32	Date: 5/9/02	Prob: 80%	Impact: high
Description: Only 70 percent of the software components scheduled for reuse will, in fact, be integrated into the application. The remaining functionality will have to be custom developed.			
Refinement/context: Subcondition 1: Certain reusable components were developed by a third party with no knowledge of internal design standards. Subcondition 2: The design standard for component interfaces has not been solidified and may not conform to certain existing reusable components. Subcondition 3: Certain reusable components have been implemented in a language that is not supported on the target environment.			
Mitigation/monitoring: 1. Contact third party to determine conformance with design standards. 2. Press for interface standards completion; consider component structure when deciding on interface protocol. 3. Check to determine number of components in subcondition 3 category; check to determine if language support can be acquired.			
Management/contingency plan/trigger: *RE* computed to be $20,200. Allocate this amount within project contingency cost. Develop revised schedule assuming that 18 additional components will have to be custom built; allocate staff accordingly. Trigger: Mitigation steps unproductive as of 7/1/02			
Current status: 5/12/02: Mitigation steps initiated.			
Originator: D. Gagne		Assigned: B. Laster	

فصل ۷: زمانبندی و ردگیری پروژه

❑داستان مهندس جوان

در اواخر دهۀ 1960 ، یک مهندس جوان برای نوشتن یک برنامه کامپیوتری برای یک کاربرد ساخت اتوماتیک انتخاب شد. دلیل انتخاب او ساده بود. او تنها فردی در گروه تکنیکی خود بود که در یک سمینار برنامه‌نویسی کامپیوتری شرکت کرده بود. او از ورودی‌ها و خروجی‌های زبان اسمبلی و FORTRAN آگاه بود ولی چیزی در مورد مهندسی نرم‌افزار و حتی کمی در مورد زمانبندی و پیگیری پروژه نمی‌دانست.

رئیس او راهنماهای مناسبی را در اختیارش گذاشت و برای او کاری را که باید انجام شود توضیح داد. به او گفته شد که پروژه باید در two ماه تکمیل شود.

او راهنماها را مطالعه نمود، روش را در نظر گرفت، و کدنویسی را بعد از دو هفته شروع کرد، رئیس او را به دفتر خود فراخواند در مورد نحوۀ پیشرفت کار از او سؤال نمود.

"بسیار عالی"، پاسخی بود که مهندس جوان با غرور و اطمینان به رئیس ارائه نمود. "این پروژه ساده‌تر از آن چیزی بود که فکر می‌کردم. احتمالاً نزدیک 75 درصد کار تمام شده است".

رئیس لبخندی زد و گفت "بسیار عالی است"، و مهندس جوان را تشویق به انجام کار نمود. آنها برای هفتۀ آینده برنامۀ ملاقات را تنظیم نمودند.

هفته بعد رئیس، مهندس را به دفترش فراخواند و پرسید، "در کجا هستیم".

مهندس جوان پاسخ داد "همه چیز خوب پیش می‌رود اما چند مشکل کوچک وجود دارد. آنها را برطرف خواهم نمود و به مسیر اصلی باز خواهم گشت".

رئیس پرسید: "در مورد مهلت چه نظری دارید؟"

مهندس جوان گفت: "مشکلی نیست، نزدیک به 90 درصد کار تکمیل شده است".

اگر شما در دنیای نرم‌افزار برای بیش از چند سال کار کرده باشید، پایان داستان را می‌دانید. تعجبی نخواهد داشت که مهندس جوان در کل زمان پروژه فقط 90 درصد آن را تکمیل نمود (با کمک دیگران) و فقط یک ماه دیرتر کار خود را به پایان رساند.

این داستان دهها هزار بار در بین توسعه دهندگان نرم‌افزار در سه دهۀ گذشته اتفاق افتاده است. یک سؤال مهم این است که چرا؟

نام درس:مهندسی نرم افزار ۱

❑زمان بندي و ردگيري پروژه چيست؟

شما يك مدل فرآيند مناسب انتخاب كرده ايد،و كارهايي كه مهندس نرم افزار بايد انجام دهد را مشخص كرده ايد، و برآورد مقدار كار و تعداد افراد را انجام داده ايدو ريسكها را ارزيابي كرده ايد.

زمانبندي و پيگيري طرح عبارت است از وصل كردن **نقطه ها به هم ديگر(نقطه بازي؟؟؟؟)**، شبكه از وظائف مهندسي نرم افزار را به وجود مي آورد كه باعث انجام به موقع كارها مي شود.بعد از ايجاد شبكه هم بايست مسئوليت براي هر كار اختصاص يابد ،اطمينان از انجام كار بدست آيد و در صورت عملي شدن ريكسها بايست با اين شبكه منطبق شود

❑چه كسي آنرا انجام مي دهد؟

در سطح پروژه مدير پروژه ، از طريق اطلاعات حاصل از مهندسين نرم افزار ،زمانبندي و ردگيري را انجام مي دهد

و در سطح انفرادي مهندسين نرم افزار خود انجام مي دهند

❑دليل اهميت آن چيست؟

در سيستمهاي پيچيده به دليل تاثير عمقي نتيجه كاري يك مرحله روي ساير مراحل و امكان انجام فعاليتهاي موازي ،**درك ارتباطات** متقابل بدون داشتن يك برنامه زمانبندي بسيار مشكل است

همچنين **ارزيابي پيشرفت** براي پروژه هاي بزرگ يا متوسط بدون يك برنامه زمانبندي جزئي غير ممكن است

❑مراحل انجام آن كدامند

پالايش *وظايف* مهندسي نرم افزار ، و انتساب نيروي كار و زمان لازم به هر *وظيفه* و ايجاد شبكه وظايف براي تحويل به موقع نرم افزار

❑چگونگي اطمينان از صحت انجام كار زمانبندي . ردگيري؟

زمانبندي مناسب به موارد زير نياز دارد:

۱-ظهور تمام وظايف در شبكه ۲-اختصاص زمان و تلاش آگاهانه به هر وظيفه

۳-نشان دادن مناسب ارتباط متقابل بين وظايف ۴-اختصاص منابع به وظايف و كارها

۵-در نظر گرفتن شمارنده هاي نزديك به هم به طوريكه پيشرفت كار قابل پيگيري باشد

نام درس:مهندسی نرم افزار ۱

مفاهیم اولیه

عوامل اصلي كه باعث تاخير در تحويل نرم افزار مي شوند :

- یک مهلت غیرقابل تحقق توسط فردی خارج از گروه توسعهٔ نرم‌افزار ایجاد شده و فشاری بر مدیران و مجریان وارد نموده است.
- تغییر نیازهای مشتری که در تغییرات زمانبندی پیش‌بینی نشده‌اند.
- کوچک شمردن تخمین میزان فعالیت و تعداد منابعی که برای انجام کار مورد نیاز است.
- ریسک‌های قابل پیش‌بینی و غیرقابل پیش‌بینی که در زمان شروع پروژه در نظر گرفته نشده‌اند.
- مشکلات تکنیکی که از قبل قابل پیش‌بینی نبودند.
- مشکلات انسانی که از قبل قابل پیش‌بینی نبودند.
- ارتباط نادرست بین سرپرست‌های پروژه که باعث تأخیر می‌گردد.
- شکست مدیریت پروژه برای تشخیص این که پروژه فراتر از زمانبندی پیش خواهد رفت، به همراه فقدان انجام عملی برای اصلاح این مشکل.

موعدهاي غيرقابل تحقق(بی حساب کتاب) ، حقايقي از زندگي در تجارت نرم افزار مي باشند . بعضي وقتها از نظر فردي كه موعد رامقرر كرده است مشروع است ولي عقل سليم مي گويد كه مشروعيت را بايد كساني كه كار را انجام مي دهند تعيين كنند

توضیحاتی بر دیر کرد

ناپلئون گفته است: "هر فرمانده که مسئول اجرای طرح می‌باشد، و فکر می‌کند با شکست روبرو می‌شود در اشتباه است. باید بر دلایل خود تأکید گذارد، بر تغییر طرح اصرار داشته باشد، و در نهایت استعفا دهد، به جای این که ابزاری باشد برای شکست ارتش خود".

اگر بهترين ارزيابي ها نشان مي دهند كه موعد مقرر ، غير قابل تحقق است يك مدير پروژه خوب بايد ، گروه خود را در برابر فشارها(زمانبندي) حمايت كند و اين فشارها را به منشاءشان باز گرداند

❑يك مثال

پروژه اي براي تحويل ۹ ماهه درخواست شده كه با انجام براوردها مشخص مي شود حداقل ۱۴ ماه نياز است

تغيير زمان تحويل ميسر نيست از سويي رد كردن پروژه هم بيسار مضر است در اين زمان انجام اقدامات زير توصيه مي شود:

۱.مهلت مورد نياز واقعي را با استفاده از داده هاي تاريخي و تجربه بصورت مشروح مشخص كنيد

۲.با استفاده از مدل فرايند افزايشي ، يك راهبرد مهندسي توليد كنيد كه عملكردهاي مهم(اصلي) را در مهلت خواسته شده انجام دهد و وظايف ديگر را به آينده موكول مي كنيم

۳.با مشتري ملاقات كنيد و با استفاده از برآوردها پروژه هاي قبلي توضيح دهيد كه مهلت خواسته شده ؛غير واقع بينانه است و با اطمينان وقت لازم براي تحقق كامل پروژه را طلب كنيد

۴.به عنوان راه چاره مدل افزايشي را به عنوان جايگزيني ارائه دهيد (ارائه راه حلهاي ديگر:)

مي توان افزايش بودجه را با افزايش تعداد نيروي كار و احتمالا پايين آمدن كيفيت پيشنهاد كرد

انجام ندادن برخي از قابليتهاي و كارايي هاي نرم افزار براي تحويل با كيفيت در ۹ماه يا همه كارايي ها براي تحويل ۱۴ ماهه

بي خيال شدن و اميدواري براي اتمام ۹ ماهه كه ممكن است به از دست دادن مشتري منجر شود.

اصول پايه

لزوم ردگيری فعاليتهای بحرانی

از Fred Brooks که مؤلف شناخته شده‌ای برای کتاب Mythial Man – Month می‌باشد، سؤال شده است که چگونه پروژه‌های نرم‌افزاری خارج از زمانبندی خود قرار می‌گیرند. پاسخ او عمیق و ساده بود:

«يك وقتي در يك روزي»

يك پروژه نرم افزاري شامل صدها كار كوچك است كه بايد براي دستيابي به هدف بزرگتر انجام شوند. برخي كارها بدون نگراني از تاثير روي كل پروژه تكميل مي شوند ولي بعضي كارهاي ديگر روي "**مسير بحراني**" قرار مي گيرند كه اگر اين كارهاي بحراني به تعويق بيفتند ،تاريخ اتمام كل پروژه به اتمام مي افتد (پيش نيازند مثل دروس پيش نياز دانشگاه)

نام درس:مهندسی نرم افزار ۱

❑هدف مدير پروژه :

۱-توصيف همه کارهاي پروژه ۲-ساختن شبکه از روابط متقابل ۳-شناسايي کارهاي بحراني

۴-پيگيري پيشرفت آنها براي اطمينان از جلوگيري "يك وقتي در يك روزي"(نيازمند ارائه برنامه ريزي)

❑تعريف زمان بندي پروژه نرم افزاري :

فعاليتي است که نيروي انساني برآورد شده را در مدت برنامه ريزي شده پروژه ، با تخصيص فعاليت به کارهاي مهندسي نرم افزار توزيع مي نمايد

برنامه زمانبندي با گذشت زمان کامل مي شود.

❑فازهاي زمانبندي براي پروژه نرم افزاري :

۱-زمانبندي ماکروسکوپي : تمام فعاليتهاي عمده مهندسي نرم افزار شناسايي مي شوند

۲-زمانبندي جزئي :هر گزينه در زمانبندي ماکروسکوپي به صورت جزئي تبديل مي گردد

❑ديدگاههاي زمانبندي (زمان پايان) پروژه نرم افزاري :

۱-محدوديت زمان پايان : سازمان نرم افزاري مجبور است در تاريخ خاصي نرم افزار را تحويل دهد

۲-عدم محدوديت زمان پايان ولي محدوديتهادر پروژه :تاريخ پايان بعد از برآورد انجام به بهترين صورت توسط تيم پروژه مشخص مي شود

❑اصول موجود در زمانبندي پروژه نرم افزاري:(۷ اصل)

۱.سوا سازي(تقسيم):پروژه مي بايست به تعدادي فعاليت و وظايف قابل مديريت تقسيم گردد(تجزيه محصول و فرآيند(فصل۳)

۲.وابستگي متقابل(داخلي):وابستگي ها بين وظايف سواشده بايست تعيين شوند. انواع وابستگي ها مي توانند الزاما ترتيبي يا موازي باشند

۳.تخصيص زمان:زمانبندي نياز به تخصيص واحد به هرکار دارد.هروظيفه بايست تاريخ شروع و اتمام داشته باشد و تعيين شود که اساس انجام کار پاره وقت است يا تمام وقت.

❑ادامه اصول زمانبندي نرم افزار از صفحه قبل

۴.اعتبار سنجي نيروي كاري:هر پروژه داراي تعداد معيني پرسنل است.در هنگام تخصيص زمان . مدير پروژه بايست اطمينان حاصل كند كه در هيچ زماني تعدادبيشتري از افراد درنظر گرفته شده در برنامه كاري نباشند

مثلا اگر منابع ما ۳ نفر-روز نيرو باشد و قصد انجام ۷ كار كه هر كدام 0.5 نفر روز فعاليت دارند پس نيرو كم مي آوريم

۵.مسئوليتهاي معين:هركاري بايست به عضوي مشخص از تيم اختصاص يابد.

۶.نتايج معين:هر كاري بايد يك نتيجه معين داشته باشد

۷.مراحل برجسته معين:يك يا چندكار بايست دركنار مرحله برجسته باشد(نقطه عطف)،مرحله برجسته از لحاظ كيفي تاييد شده

رابطه بين افراد و فعاليت(نيروی كار)

گاهي ميتوان در پروژه هاي كوچك يك فرد مي تواند تمام مراحل تجزيه و تحليل ، برنامه نويسي و آزمون را به تنهايي انجام دهد

با افزايش پيچيدگي كار و پروژه تعداد افراد لازم افزايش مي يابد .افسانه ي معروفي وجود دارد كه با افزايش افراد مي توان عقب افتادگي ها را جبران كرد(فصل ۱) حال آنكه افزايش تاثير مخربي بر پروژه دارد و حتي باعث به هم خوردن برنامه زمانبندي مي شود افزايش افراد باعث افزايش ارتباطات شده كه ارتباطات بيشتر زمان بيشتري را در روند ساخت پروژه نياز دارند

- **مثال** چهار مهندس نرم‌افزار را در نظر بگیرید که هر یک قادر به تولید 5000 LOC/year می‌باشند، زمانی که بر روی پروژه‌های مستقل کار می‌کنند. هنگامی که این چهار مهندس در یک تیم پروژه قرار می‌گیرند، شش مسیر ارتباطی امکان‌پذیر می‌شود. هر مسیر ارتباطی نیازمند زمانی است که می‌توانست برای توسعه نرم‌افزار صرف شود. می‌توان فرض نمود که قابلیت تولید تیم (برحسب LOC) به اندازهٔ 250 LOC/year برای هر مسیر کاهش می‌یابد، که سربار ناشی از این ارتباط می‌باشد. بنابراین قابلیت تولید تیمی برابر است با 20000 − (250×6) = 18500 LOC/year که 7.5 درصد کمتر از میزان مورد انتظار است پروژه‌ای یک ساله که این تیم بر روی آن کار می‌کنند از زمانبندی عقب می‌افتد، و با دو ماه باقیمانده، دو نفر دیگر به تیم اضافه می‌شوند. تعداد مسیرهای ارتباطی به 14 می‌رسد. قابلیت تولید افراد جدید برابر است با 840 × 2 = 1680LOC برای دو ماه باقیمانده به زمان تحویل قابلیت تولید تیمی اینک برابر است با 20000 + 1680 − (250 × 14) = 18180LOC/year

❖**نکته مهم :رابطه بین افراد و کار، غیرخطی است یعني یا اضافه کردن افراد نمی توان زمان انجام کار را کاهش داد**
مثلا اشتباه است اگر بگوییم کاري که با ۳ نفر، در ۲ ماه انجام مي شود با ۶ نفر در ۱ ماه انجام مي شود.

رابطه تجربی

اثبات رابطه غيرخطي بين زمان تكميل پروژه و فعاليت انساني با بكارگيري معادله نرم افزار(براي تعيين تعداد افراد و...)

معادله رابطه افراد و كارايي را بدست مي آوريم

$$L = P \times E^{1/3} t^{4/3}$$

L = تعداد خطوط برنامه

E = نيروي مورد نياز براي توسعه برحسب نفر/ماه

P =پارامتر بهره وري است(پارامترهاي براي افزايش كيفيت / مثلا :۳۰۰۰ خط در سال)

t = مدت پروژه بر حسب ماه

رابطه ميزان كار و زمان(افزايش مهلت زماني = كاهش افراد = افزايش كارايي و سود)

$$E = L^3/(P^3 t^4)$$

E = نيروي مورد نياز براي توسعه برحسب نفر/سال

P =پارامتر بهره وري است t = زمان توسعه بر حسب سال

•مهلت در مخرج قرار دارد

توزیع نیروی کار

توزيع توصيه شده براي فعاليت در طول فازهاي تعريف و توسعه ، **قانون ۴۰-۲۰-۴۰** است، ۴۰ درصد تمام تلاشها به طرح و تجزيه و تحليل ابتدايي و پاياني اختصاص دارد . ۴۰ درصد به آزمون نهايي مربوط است و برنامه نويسي با تاكيد كمتري فقط ۲۰ درصد كل تلاش را شامل مي شود .

برنامه ريزي ۲ تا ۳ درصد. تحليل امكانات بين ۱۰ تا ۲۵ درصد و محدوده (دامنه) بين ۲۰ تا ۲۵ درصد به طرح نرم افزار مبذول مي گردد آزمايش ۳۰ تا ۴۰ درصد است ولي اگر نرم افزار حساس باشد يا انسان ارزيابي كند(ترجمه بهتر:تاثير روي حيات انسان) زمان بيشتري به آزمايش نسبت داده مي شود

تعریف یک مجموعه وظایف برای پروژه نرم افزاری

طبق مطالب بحث شده در فصل دو(فرایند) ، بر طبق فرآیند مجموعه کاري تعریف مي شود

يك مجموعه واحد وظايف براي همه پروژه ها مناسب نيست No single set of tasks is appropriate for all projects.

مجموعه كاري براي يك سيستم بزرگ و پيچيده براي يك سيستم ساده و كوچك افراطي است.

يك مجموعه كاري ، **مجموعه اي از كارهاي مهندسي نرم افزار ، هدف هاي اساسي(عملكردهاى اصلى) و قابليتهاي عرضه(تحويل) است** كه براي تكميل يك پروژه خاص بايد گرد هم آيند.و نبايد با كار بيهوده تيم پروژه را خسته كنند

❖**طبقه بندي از انواع پروژه نرم افزاري**

۱.**پروژه هاي توسعه مفهوم :**براي كشف بعضي مفاهيم تجاري يا كاربردي از يك تكنولوژي جديد(تجزيه و تحليل) (مثال :ajax)

۲.**پروژه هاي توسعه برنامه كاربردي جديد:**با يك سفارش از طرف مشتري شروع مي شود

۳.**پروژه هايي افزايش برنامه كاربردي(ارتقا و اصلاح بزرگ):**اصلاحات عمده در رابطه با عملكرد كه كاربرنهايي كاملا حس مي كند(مثل تغيير DBوسرعت)

۴.**پروژه هاي نگهداري برنامه كاربردي:**اصلاحات ،تصحيح تطبيق كه سريعا براي كاربر نهايي محسوس نيست(رفع يك باگ)

۵.**پروژه هاي مهندسي مجدد:**براي تجديد ساخت (ساخت مجدد)كلي يا جزئي و به علت تحليل قبلي شناخت ساده تر است

ميزان سختي و دقت

ميزان دقت كه با آن فرايند ساخت نرم افزار به كار گرفته مي شود در مورد پروژه هاي مختلف متفاوت است ولي همه براي ساخت نرم افزار با كيفيت كار مي كنند

❖**چهار درجه دقت متفاوت در ساخت نرم افزار قابل تعريف است:**

۱.**معمولي :** تمام فعاليتهاي چارچوب فرايند(cpf) اجرا مي شوند، فعالیتهای چتری به حداقل می رسند و مستند سازي كمتر است

❖**ادامه چهار درجه دقت متفاوت در ساخت نرم افزار قابل تعریف است:**

۲.ساختیافته :cpf ها (فعاليتهاي چارچوب فرآيند) اجرا مي شوند و فعاليتهاي چتري متناسب با نوع پروژه انجام مي شوند و تمام فعاليتهاي چتري مورد نياز براي تضمين كيفيت بالا به اجرا در مي آيند.

۳.ثابت(دقیق) :درجه بالا در نظم و ترتيب اجراي cpf ها براي تضمين كيفيت بالا و تمام فعاليتهاي چتري نيز انجام مي شوند و مستندسازي كامل صورت مي پذيرد

۴.واکنش سریع:به خاطر سرعت فقط كارهايي به كار گرفته مي شوند كه به كيفيت مناسب منتهي مي شوند

تعریف معیار تطابق

معيارهاي تطبيق و سازگاري براي تعيين ميزان دقتي كه با آن فرايند نرم افزاري بايست اجرا شود به كار مي روند

يازده معيار سازگاري و تطابق براي پروژه هاي نرم افزاري تعريف شده است:

1. **اندازه پروژه**
2. **حداكثر تعداد كاربران**
3. **حساسيت كار**
4. **ديرپايي كاربرد(طول عمر برنامه)**
5. **پايداري نيازها**
6. **سهولت ارتباط مشتري/برنامه نويس**
7. **تكامل فناوري قابل كاربرد**
8. **قيود و محدوديتهاي عملكرد**
9. **خصوصيتهاي سيستمهاي جاسازي / غير جاسازي**
10. **اعضاي تيم و كاركنان**
11. **فاكتورهاي مهندسي مجدد**

به هريك از معيارهاي تطبيق شماره اي بين ۱ تا ۵ اختصاص داده مي شود كه ۱ نشان دهنده وجود مجموعه از كارهاي فرايند و مستند سازي كم و ۵ نشان از وجود تمام كارهاي فرآيند و مستند سازي زياد است

محاسبه یک معیار انتخاب مجموعه وظایف

براي انتخاب مجموعه وظايف مناسب براي يك پروژه مراحل زير را بايد انجام داد:

۱.تعیین درجات ۱۱ معیار تطبیق و ثبت آنها در جدول

۲.تعیین ضریب یا میزان ارزش (عامل وزن)هر معیار بین 0.8 تا 1.2)

۳.ضرب عامل وزن(ارزش) در ضریب نقطه ورودي (صفر يا يك) و قرار دادن در ستون حاصلضرب

شماره * عامل وزن * ضريب نقطه ورودي

۴. ميانگين داده هاي ستون حاصلضرب را حساب كرده و نتيجه را به عنوان انتخابگر مجموعه وظايف(TSS) مي نويسيم

task set selector

ضریب امتیاز ورودی

معیار انطباق	درجه	ارزش	نقطه ورودی					محصول
			Conc.	Maint.	Enhan.	NDev.	Reeng.	
			پروژه‌های توسعه	پروژه‌های کاربرد	پروژه‌های پیشرفت	پروژه‌های نگهداری	پروژه‌های مهندسی	
اندازه پروژه	____	1.20	0	1	1	1	1	____
تعدادکاربران	____	1.10	0	1	1	1	1	____
بحرانی بودن تجارت	____	1.10	0	1	1	1	1	____
طولانی مدت	____	0.90	0	0	1	1	0	____
پایداری نیازها	____	1.20	0	1	1	1	1	____
سهولت ارتباط	____	0.90	1	1	1	1	1	____
کامل بودن تکنولوژی	____	0.90	1	0	0	1	1	____
محدودیت های اجرا	____	0.80	0	0	1	1	1	____
جاسازی شده / جاسازی نشده	____	1.20	1	0	1	1	1	____
مدیریت پروژه	____	1.00	1	1	1	1	1	____
عملکرد داخلی	____	1.10	0	1	1	1	1	____
فاکتورهای مهندسی مجدد	____	1.20	0	0	0	0	1	____

تفسیر مقدار انتخاب مجموعه وظایف (TSS)

- **اتفاقي(معمولی)** TSS < 1.2
- **ساختاري(ساختيافته)** 1.0 < TSS < 3.0
- **ثابت** TSS > 2.4

نام درس:مهندسی نرم افزار ۱

انتخاب وظایف مهندسی نرم افزار(بعد از تعیین دقت)

برای داشتن یک برنامه زمانبندی پروژه، مجموعه وظائف باید در مدت زمانی پروژه توزیع گردد. مجموعه وظائف نسبت به **نوع پروژه** و **میزان دقت** تغییر خواهد کرد.هریک از پروژه های توصیف شده در بخش قبل، ممکن است با به کاربردن یک **مدل فرایندی** که خطی، متوالی و تکراری (مثلا مدلهای افزایشی یا نمونه سازی) و یا تکاملی است (مانند مدل حلزونی) قابل دسترسی باشند، در بعضی موارد یک پروژه به آرامی به پروژه بعدی ملحق میشود.مثلا ممکن است پروژه توسعه مفهوم به پروژه توسعه کاربرد جدید تکامل یابد.یا پس از ایجاد پروژه توسعه کاربرد ممکن است پروژه افزایش کاربرد شروع شود.

این پیشرفت طبیعی و قابل پیشگویی است و بدون توجه به مدل فرآیندی که توسط یک سازمان در پیش گرفته شده است اتفاق می افتد.بنابراین وظائف مهندسی نرم افزار گفته شده در بخش هایی بعدي، در مورد تمام مدل های فرآیندی قابل اجرا هستند

❖**مثال) وظائف مهندسی نرم افزار برای یک پروژه توسعه مفهوم:**(اطمینان از قابل دستیابي بودن)

- **تعیین دامنه مفهوم** – توانایی سازمان برای انجام کاری که در دامنه پروژه نهفته است را به وجود می آورد
- **طرح ریزی اولیه مفهوم** – برقرار ساختن توانایی سازمان در به عهده گرفتن کارهای مربوط به حوزه پروژه
- **ارزیابی ریسک فنی** – خطری را که همراه فناوری مورد کاربرد حوزه پروژه است را ارزیابی می کند
- **آزمون(و اثبات) مفهوم** – کارایی و عملي بودن یک فناوری نو را حوزه نرم افزاری نشان می دهد
- **پیاده سازی مفهوم** – نمود مفهوم را به گونه ای که توسط یک مشتری قابل بررسی باشد پیاده می کند و برای اهداف بازیابی به کار می رود ، زمانی که در نظر باشد مفهومی به مشتریان با مدیریت دیگری فروخته شود
- **واکنش مشتری به مفهوم** – واکنش نسبت به مفهوم یک فناوری نو را جستجو می کند و کاربردهای مشتری خاصی را مورد هدف قرار می دهد.

حرکت مهندسی نرم افزار به سوی پروژه های توسعه مفهوم (و نیز تمام انواع دیگر پروژه ها) تا اندازه کمی از عقل سلیم بیشتر است.
تیم نرم افزار باید بداند که چه کارهایی باید انجام گیرد (تعیین حوزه)
تیم (یا مدیر) باید تعیین کند که آیا کسی برای انجام آن وجود دارد (برنامه ریزی)
خطرات ممکن را مورد ملاحظه قرار دهد (تخمین خطر) فناوری را تا اندازه ای مورد تامین قرار دهد (آزمون مفهوم)
و آن را به گونه ای شاخص پیاده نماید به طوری که مشتری بتواند آن را ارزیابی کند (پیاده سازی مفهوم و ارزشیابی مشتری)
و در پایان ، اگر مفهوم کارامد است، یک گونه تولیدی (تبدیلی) باید تولید گردد.

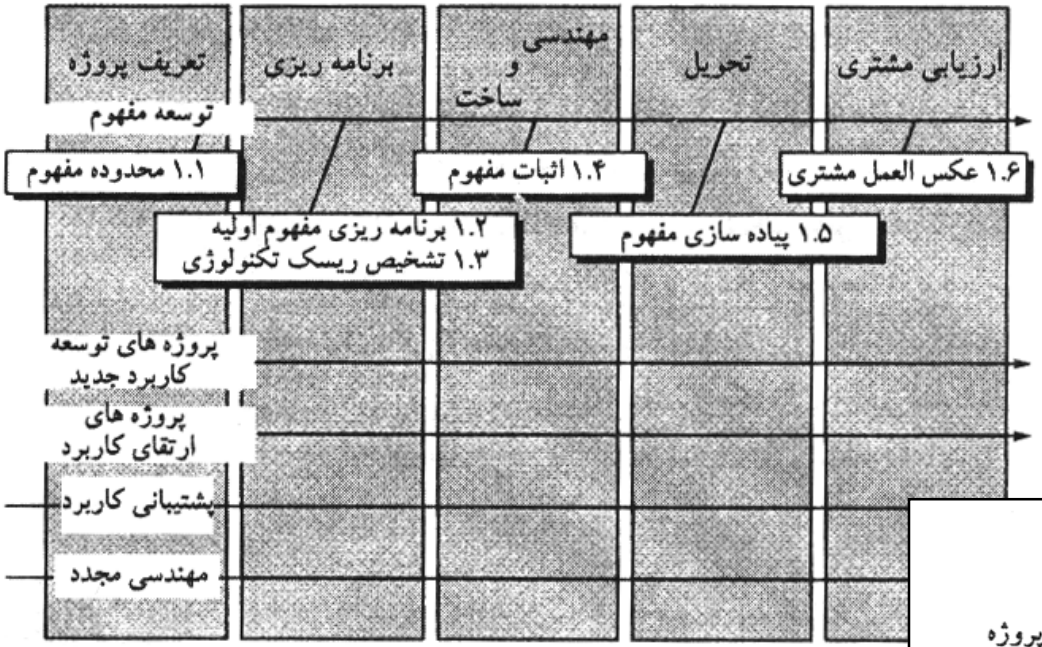
نام درس:مهندسی نرم افزار ۱

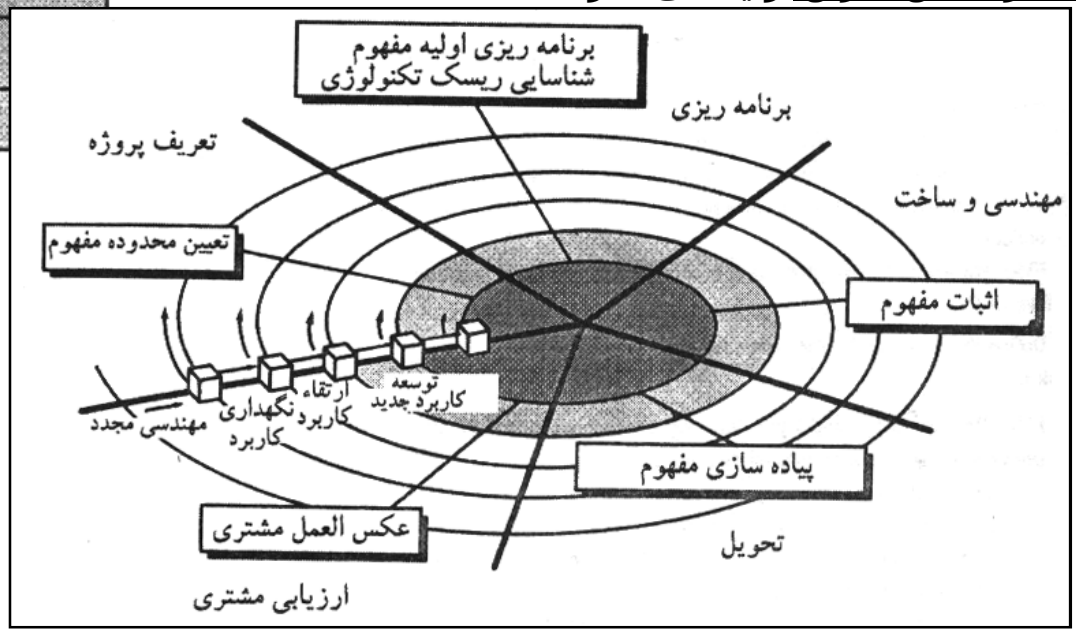
•به کارگیري مجموعه وظایف مهندسي در دو مدل فرآیند متفاوت

فعالیت های شبکه وظائف توسعه مفهوم ماهیتا تکراری است. یعنی : یک پروژه توسعه مفهوم واقعی از راه تعدادی افزایش برنامه ریزی شده به این فعالیت ها نزدیک می شود که هریک مختص تولید یک محصول قابل تحویل، که توسط مشتری قابل ارزشیابی است، هستند.

اگر یک مدل فرایند خطی انتخاب شده باشد، هر یک از این افزایشها در یک توالی تکراری تعریف شده اند.

در خلال هر مرحله، فعالیتهای پوششی (چتري) اجرا می شوند. کیفیت کنترل می شود و در پایان هر مرحله یک محصول قابل تحویل تولید می شود.





پالایش وظایف اصلی

•کارهای عمده که در بخش قبل توصیف گردیدند ممکن است برای تعریف یک برنامه زمان بندی **ماکروسکوپی** برای یک پروژه به کار روند. با این وجود، برنامه زمان بندی ماکروسکوپی برای ایجاد یک **برنامه جزیی** برای پروژه باید **پالایش** گردد. پالایش با در نظر گرفتن کارهای بزرگ و تجزیه آنها به کارهای فرعی (با محصولات کاری و هدفهای اصلی مربوط) شروع می شود.

❖**مثال:** به عنوان نمونه ای از تجزیه کردن وظیفه، تعیین حوزه برای یک پروژه توسعه مفهوم [*] را، که در بخش قبل بحث شد، مورد ملاحظه قرار دهید. پالایش وظیفه با بکار بردن یک طرح خلاصه شده قابل انجام است، اما در این کتاب، یک رهیافت زبان طراحی فرایندی برای نشان دادن جریان فعالیت تعیین حوزه مفهوم به کار برده می شود:

وظیفه I– ۱ تعیین حوزه مفهوم

I-۱-۱ نیازها، منافع ومشتریان بالقوه را شناسایی کنید.

I-۱-۲ رویدادهای برونی کنترل و درونی را که کاربردها را هدایت می کنند، تعریف کنید.

وظیفه I-۱-۲ را شروع کنید.

I-۱-۲-۱ : FTR(بازبيني رسمي فني) توصیف نوشتاری نیازها را بررسی کنید.

I -۱-۲-۲ لیستی از خروجی ها / ورودیهای قابل مشاهده مشتری بدست دهد.

بر حسب مورد : دستگاهها

اگر دستگاه ها= کیفیت و آمادگی کارکردی

با مشتری ملاقات نمایید تا نیازهای مفهومی اصلی مشخص گردند. با مصرف کنندگان نهایی مصاحبه کنید

رهیافت کنونی به مسئله . روند جاری را مورد مشاهده قرار دهید تقاضاها و شکایات گذشته را بررسی کنید

اگر دستگاه ها= تجزیه و تحلیل ساخت یافته.

لیستی از اشیا داده های اصلی تهیه کنید رابطه بین اشیا را تعریف کنید ویژگی اشیا را تعریف کنید

اگر دستگاه ها : دید عینی

لیستی از کلاسهای مسائل تهیه کنید سلسله مراتب کلاس و ارتباط کلاس را توسعه دهید ویژگی های کلاسها را توصیف کنید

پایان مورد

I-۱-۲-۳ ورودي/ خروجي ها را با مشتری مرور کنید و در موارد لازم آنها را اصلاح کنید

تعریف یک شبکه وظایف

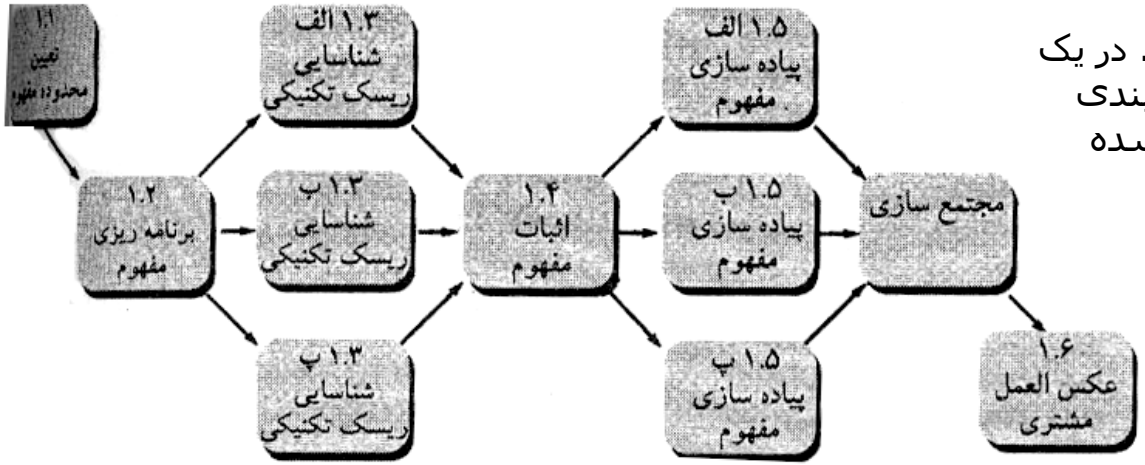
وظائف و وظائف فرعی بر پایه **ترتیبشان**، به هم وابستگی های متقابل دارند.اگر بیش از یک نفر در یک پروژه نرم افزاری باشند، احتمالا فعالیتها و وظائف توسعه ای به صورت **موازی** اجرا می شوند پس کارهای همروند باید با هم هماهنگ شوند، طوری که وقتی کارهای بعدی تولید کاری آنها را نیاز دارد آنها کامل شده باشند.

یک شبکه وظائف(= شبکه فعالیت)، **نمایشی نموداری و گرافیکی از جریان وظیفه** برای پروژه می باشد. گاهی از آن به عنوان مکانیسمی برای انتقال توالی وظیفه و وابستگی های آن به یک ابزار برنامه ریزی خودکار پروژه، استفاده می شود.

- در ساده ترین شکل آن برای ایجاد یک برنامه ماکروسکوپی به کار می رود، شبکه وظیفه در توصیف کارهای اصلی مهندسی نرم افزار به کار می رود.

ماهیت همزمانی فعالیتهای مهندسی نرم افزاری منجر به نیازمندیهای برنامه ریزی مهمی می شود. از آنجا که کارهای موازی همزمان اتفاق می افتد، برنامه ریز باید **وابستگی های بین کارها** را مشخص نماید تا پیشرفت مداوم در جهت تکمیل کار تضمین گردد به علاوه، مدیرپروژه باید باید آگاهی از وظائفی که در **موقعیت بحرانی** قرار می گیرند داشته باشد. یعنی، آن کارهایی که در برنامه زمانبندی باید کامل گردند

- **مثال :**شبکه وظایف در تصویر مقابل ماکروسکوپی است. در یک شبکه وظایف جزیی (که مقدمه ای برای یک برنامه زمان بندی جزیی است) هر فعالیتی که در تصویر مقابل نشان داده شده است باید بسط داده شود



زمانبندی

زمانبندي پروژه **توزيع زماني مجموعه فعاليتها در طول مدت پروژه** كه بسته به پروژه نرم افزاري تغيير مي كنند.**زمانبندی پروژه** نرم افزاری تفاوت چنداني با زمانبندی فعاليتهاي كاري ديگر مهندسي نرم افزار ندارد. پس با کمی اصلاح ابزارهای زمانبندی عمومی پروژه ها و نیز فنون آن در مورد پروژه های نرم افزاری نیز قابل اجرا هستند.

۱-ارزشیابی برنامه و فنون بررسی (PERT) و **۲**-روش مسیر بحرانی (CPM) دو روشی هستند که برای زمانبندی پروژه به کار می روند که برای توسعه نرم افزاری نیز قابل کاربردند.

❖ این دو تکنیک به وسیله **اطلاعات حاصل از فعالیتهای برنامه ریزی پروژه قبلی** بدست آمدند،يعني قبل از زمانبندی بايست:

- برآوردهای نیروی انسانی
- تجزیه کارکرد محصول
- انتخاب مدل فرآیند مناسب و مجموعه وظائف
- تجزیه وظائف

با به کار بردن یک شبکه وظائف امکان تعریف وابستگی های موجود بین وظائف وجود دارد.وظائف که بعضی وقتها به آنها ساختار تجزیه کار پروژه (WBS) گفته می شود، برای کل پروژه یا کارکردهای فردی تعریف می شوند

❖PERTو CPM به برنامه ریز نرم افزاری امکان می دهد:

۱.مسیر بحرانی را مشخص نماید – زنجیره وظائفی که مدت زمانی پروژه را تعیین می کند.

۲."محتمل ترین " **برآورد زمانی"** برای کارهای فردی را با به کار بردن مدلهای آماری انجام دهد.

۳.زمانهای مرزی را محاسبه کند که به تعریف یک "**پنجره**" زمانی برای یک وظیفه خاص می انجامد.(۵ نوع زمان مرزي)

محاسبه زمانهای مرزی پروژه نرم افزاری می تواند بسیار مفید باشد، مثلا: سهل انگاري در طرح یک تابع می تواند توسعه بیشتر توابع دیگر را به تاخیر اندازد.}

Critical path method (CPM)
Program evaluation and review technique (PERT)
Work Breakdown Structure (WBS)

□**زمانهای مرزی مهمی که ممکن است از CPM یا PERT بدست آیند:**

۱)زودترین زماني كه يك وظيفه مي تواند شروع شود(درصورت تكميل كارهاي قبلي)

۲)دیرترین زمان برای شروع وظیفه قبل را به تاخیر افتادن زمان حداقل تکمیل پروژه.

۳)زودترین زمان خاتمه وظیفه =زودترین زمان شروع + مدت زمانی وظیفه

۴)دیرترین زمان خاتمه وظیفه =دیرترین زمان شروع + مدت زماني وظيفه

۵)تعلیق کلی (زمان شناوري):مقدار زمان اضافي يا ميزان مجاز تغييرات زماني،طوريكه مسير بحراني پروژه بر طبق برنامه باقي بماند.

محاسبه مرزی منجر به تعیین مسیر بحرانی می شود و روشی کمی در اختیار مدیر می گذارد که با آن می تواند به هنگام تکمیل شدن کارها پیشرفت آنها را ارزشیابی کند دو روش CPM و PERT را با نرم افزارهاي زيادي مي توان پياده سازي كرد

نمودارهای خطی زمانی

هنگام ایجاد یک برنامه زمان بندی برای پروژه نرم افزاری برنامه ریز آن، کارش را با مجموعه ای از وظائف (ساختار تقسیم کار) شروع می کند. اگر ابزارهای خودکار به کار روند، تقسیم کار به عنوان شبکه وظیفه یا خلاصه وظیفه در نظر گرفته می شود.**نیروی کاری ،مدت زمانی، و زمان شروع وظیفه و ورودی هر وظیفه** در نظر گرفته می شود.همچنین ممکن است وظائف با افراد خاصی اختصاص داده شوند.

در نتیجه این کار، یک نمودار زمانی (نمودار گانت) به وجود می آید. یک نمودار زمانی را می توان به کل پروژه توسعه داد. از طرف دیگر، می توان نمودارهای جداگانه نیز برای هر یک از توابع پروژه یا برای کارهای فردی که بر روی پروژه انجام می شوند به وجود آورد.

Timeline chart(Gantt chart)

نام درس:مهندسی نرم افزار ۱

•تمام کارهای پروژه (برای تعیین حوزه مفهومی) در ستون سمت چپ لیست گردیده اند.

• باریکه های افقی مدت زمانی هر وظیفه را نشان می دهند. وقتی چندین باریکه در یک زمان اتفاق می افتند، نشان دهنده همزمانی کارهاست.

• لوزی ها نشان دهنده نقاط عطف اصلی اند.

یک بار اطلاعات لازم برای به وجود آوردن یک نمودار زمانی وارد گردیده است و اکثر ابزارهای زمانبندی پروژه های نرم افزاری **جدولهای پروژه** ای تولید کرده اند –که لیستی جدول وار از تمام وظائف پروژه، شروع برنامه ریزی شده و واقعی و زمانهای پایان کار، و انواعی از اطلاعات مربوطه میباشد.

جدولهای پروژه که به همراه جدولهای زمانی به کار می روندبه **مدیر پروژه** امکان **پیگیری پیشرفت کار**را مي دهد

کارها	شروع برنامه ریزی شده	شروع واقعی	تکمیل طبق برنامه	تکمیل واقعی	افراد اختصاص یافته	فعالیت اختصاص یافته	نیازها
۱.۱.۱ مشخص نمودن نیازها و فواید	wk1, d1	wk1, d1	wk1, d2	wk1, d2	BLS	2 p-d	Scoping will require more effort/time
ملاقات با مشتری	wk1, d2	wk1, d2	wk1, d2	wk1, d2	JPP	1 p-d	
مشخص نمودن نیازها و محدودیت های پروژه	wk1, d3	wk1, d3	wk1, d3	wk1, d3	BLS/JPP	1 p-d	
نشانگر : تعریف شدن حکم محصول	wk1, d3	wk1, d3	wk1, d3	wk1, d3			
۱.۱.۲ تعریف خروجی – کنترل – ورودی مطلوب (OCI)							
محدوده توابع صفحه کلید	wk1, d4	wk1, d4	wk2, d2		BLS	1.5 p-d	
محدوده توابع ورود صوت	wk1, d3	wk1, d3	wk2, d2		JPP	2 p-d	
محدوده مود ارتباط	wk2, d1		wk2, d3		MLL	1 p-d	
محدوده شناسایی سند	wk2, d1		wk2, d2		BLS	1.5 p-d	
محدوده توابع دیگر WP	wk1, d4	wk1, d4	wk2, d3		JPP	2 p-d	
مستند سازی	wk2, d1		wk2, d3		MLL	3 p-d	
FTR: مرور OCI با مشتری	wk2, d3		wk2, d3		all	3 p-d	
معیار : تعریف OCI	wk2, d4		wk2, d4		all	3 p-d	
۱.۱.۳ تعریف عملکر – رفتار	wk2, d5		wk2, d5				

ردگیری برنامه زمانبندی

برنامه ی زمانبندی پروژه : **وظایف** و **نقاط عطف اصلی** ای را که باید در جریان پیشرفت پروژه پیگیری و کنترل شوند، تعریف می کند.

•پیگیری به طور خلاصه به روشهای زیر صورت می پذیرد:

۱-برگزاری گردهمایی
۲-ارزشیابی نتایج تمام بررسی ها
۳-مقایسه زمان تحقق یافتن اهداف رسمی(نقاط عطف) و زمان برنامه ریزي شده
۴-مقایسه ی شروع واقعی براي هر وظیفه وشروع برنامه ریزی شده(در گانت)
۵-ملاقات غیر رسمی با کارورزان به منظور دریافت میزان پیشرفت یا خطرات
۶-محاسبه ی ارزیابی کمی پیشرفتها با بکار گیری ارز ش سنجی اکتسابی

•وظایف مدیر پروژه هنگام بروز مشکل:

به اعمال کنترل می پردازد .در این هنگام منابع بیشتری به آنها اختصاص داده می شود وترتیب واستخدام کارکنان تغییر می کند و یا برنامه ی پروژه ی زمانبندی اصلاح می شود.

•[Time-boxing]: جعبه ای کردن زمان (تلاش جهت رسیدن به هر وظیفه فقط درمحدوده خودش)

در مواردی که فشار زمانی حاد به وجود می آید،مدیران با تجربه اقدام به بکارگیری برنامه ی زمانبندی وتکنیک کنترل خاص در برنامه می کنند که جعبه ای کردن زمان نام دارد

راهبرد جعبه ای کردن زمان تشخیص می دهد که کل محصول کامل در موعد مقرر قابل عرضه نخواهد بود لذا یا مدل فرایند افزایشي انتخاب مي شود و براي هر مرحله افزایش برنامه اي تدوین مي شود.

وظیفه ای که به هر یک از مراحل افزایشی اختصاص داده شده است از نظر زمانی به اصطلاح جعبه ای شده است.

برنامه ی زمانی،برای هر وظیفه تنظیم شده است و این کار با برگشت از زمان تحویل محصول به مراحل افزایشی انجام گرفته است.یک جعبه در اطراف هر وظیفه گذاشته شده است. **هرگاه وظیفه ای به مرز زمان جعبه اش می رسد،آن وظیفه متوقف می شود و وظیفه ی بعدی آغاز می شود.**(با تلرانس ۱۰درصد انجام شده پس مابقي براي بعد)

تحلیل مقدار بدست آمده (ارزش سنجی اکتسابی):

❖**ارزش سنجی اکتسابی**

تکنیکی که برای **تحلیل کمی پیشرفت** وجود دارد،ارزش سنجی اکتسابی (EVA)نامیده می شود

❖**نظر فلیمینگ و کاپلمن در این مورد(مقادیر بدست آمده و اکتسابی):**

این ارزشیابی برداشت دقیق و قابل اطمینانی از اجرای پروژه با گذشت ۱۵%اولیه ی آن به بعد فراهم می کند.

❖**تعیین ارزش اکتسابی طبق مراحل زیر است:**

۱-ارزش بودجه ای کار زمانبندی شده (BCWS) برای هر وظیفه ای که در برنامه است ،تعیین می شود.

۲-مقدارهای ارزش بودجه ای کار زمانبندی شده(یعنی:مقادیر(BCWS)برای تمام کارها جمع شده اند، تا بودجه را در زمان تکمیل پروژه (BAC)نشان دهند.)

$$BAC = \Sigma\ (BCWS_k) \text{ for all tasks } k$$

۳-مرحله ی بعد،ارزش بودجه ای کار اجرا شده(BCWP) محاسبه می شود.این مقدار برابر است با حاصل جمع مقدارهای ارزش بودجه ای کار اجرا شده برای همه ی کارهایی که واقعا در یک نقطه ی زمانی در زمانبندی پروژه ی تکمیل شده است.

•تفاوت بین ارزش بودجه ای کار زمان بندی شده و ارزش بودجه ای کار اجرا شده در این است که اولی نماینده ی بودجه ی فعالیتهایی است که برای تکمیل آنها برنامه ریزی شده است،در حالی که دومی نماینده ی فعالیت هایی است که واقعا کامل شده اند

Earned value analysis(EVA)

budgeted cost of work scheduled (BCWS) budget at completion(BAC) budgeted cost of work performed(BCWP)

با داشتن مقدار های ارزش بودجه ای کار زمان بندی شده، بودجه ی "تکمیل کار وارزش بودجه ای کار اجرا شده،نشانگر های عمده ی پیشرفت را می توان محاسبه کرد.

Schedule Performance Index , SPI=BCWP/BCWS

ارزش بودجه ای کار زمان بندی شده/ارزش بودجه ای کار اجرا شده=**شاخص عملکرد زمانبندی**

Schedule varance,SV=BCWP-BCWS

ارزش بودجه ای کار زمان بندی شده-ارزش بودجه ای کار اجرا شده=**واریانس زمانبندی**

Percent Schedule for completion=BCWS/BAC

بودجه به هنگام تکمیل کار/ارزش بودجه ای کار اجرا شده =**در صد برنامه ریزی شده برای تکمیل کار** که نشانگر کمی از کامل شدن پروژه است

Cost Performance Index , CPI=BCWP/ACWP

ارزش واقعی کار اجرا شده/ارزش بودجه ای کار اجرا شده=**شاخص اجرایی هزینه**

Cost Variance,Cv=BCWP-ACWP

ارزش واقعی کار اجرا شده-ارزش بودجه ای کار اجرا شده=**واریانس هزینه**

- ACWP= هزینه واقعی کار انجام شده

رد گیری خطا

جستجوی خطاها به شما اجازه می دهد،فعالیت کنونی خود را با نیروی پیشین مقایسه نموده و یک معیار کمی برای کیفیت کاری که اداره می شود ،تهیه کنید.

•تیم نرم افزاری به اجرای بررسی های فنی رسمی (و سپس آزمودن)برای یافتن واصلاح خطاهایE ،که در محصولات کاری در خلال وظایفشان تولید شده اند می پردازند.
هرخطایی که کشف نشود به عنوان نقص در نظر گرفته می شود.
Dکارایی و بازده رفع نقص به صورت زیر تعریف می شود.

D=E/(E+D)

•Dre يك متريك فرآيند مي باشد كه نمايش قوي از موثربودن فعاليتهاي **تضمين كيفيت** را فراهم ميكند و همچنين مي تواند توسط مدير پروژه استفاده شود تا او را در تعيين پيشرفت پروژه در كارهاي زمانبندي حمايت كند.مدير پروژه بر حسب DRE می تواند تصمیم بگیرد که چه قسمتهایی نیاز به مرور دارند که اینکار باعث صرفه جویی در زمان می شود

طرح(برنامه) پروژه

هر مرحله در مهندسي نرم افزار بايست عنصر قابل تحويلي را توليد كند،با **اعمال برنامه ريزي طرح پروژه نرم افزاري** به وجود آمده است . كه شامل اطلاعات مبنايي براي هزينه و زمانبندي است.

خصوصيات برنامه پروژه نرم افزاري:

۱-بيان حوزه و منابع براي مدبز،سرپرست و شمتري

۲-تعريف ريسك و راههاي كاهش

۳-تعريف هزينه و زمانبندي

۴-تعيين روش كلي ساخت نرم افزار

۵-تعيين نحوه تضمين كيفيت و مديريت تغييرات

طرح پروژه ايستا نيست و در مراحل مختلف توسط تيم پروژه به روز مي شود

فصل ۸: تضمین کیفیت نرم افزار (SQA)

Software quality assurance (SQA)

فن مدبران

آز مساس

تضمین کیفیت نرم افزار چیست؟

فعاليتي چتري پوششي است كه در سراسر فرآيند به كار گرفته مي شود و شامل : ۱)روش مديريت كيفيت ۲)فناوري مهندسي نرم افزار(روشها/ابزارها) ۳)بررسيهاي فني رسمي در طول اجراي فرايند ۴)روش آزمايش چند لايه اي ۵)كنترل مستندات و تغييرات انجام شده روي آنها ۶)روالي براي تاييد انطباق با استانداردهاي توسعه نرم افزار ۷)روشهاي اندازه گيري و گزارش

تضمین کیفیت نرم افزار در مورد چه چیزي بحث مي كند؟

تنها گفتن اینکه کیفیت نرم افزاری مهم است کافی نمی باشد--- **۱)**باید به روشنی بیان کنید که وقتی میگویید "کیفیت نرم افزار"منظورتان چیست---**۲)**ایجاد فعالیت هایی را که به تضمین کیفیت بالای هر محصول مهندسی نرم افزاری کمک می کنند ---**۳**)اجراي فعاليت هاي تضمين كيفيت را بر روی هر پروژه نرم افزاری --**۴**)استفاده از متریک هایی که استراتژي هايي رابراي ارتقا فرآيند و درنتيجه ارتقا كيفيت محصول نهايي توسعه دهند

چه كسي آنرا انجام مي دهد؟

همه افراد در فرآيند مهندسي نرم افزار است ، مسئول كيفيت اند.

دلیل اهمیت تضمین کیفیت؟

اگر تيم نرم افزار در تمام فعاليتهاي مهندسي نرم افزار بر كيفيت تاكيد داشته باشد،ميزان كار مجددي كه بايد انجام شود كاهش مي يابد. يعني نياز به **دوباره كاري** نيست و كاهش هزينه ها و صرفه جويي در زمان را به دنبال دارد

مراحل آن چیست؟

[۱]تعريف كيفيت نرم افزار در سطح انتزاعي ، پس از درك مفهوم كيفيت در يك پروژه [۲]تعيين فعاليتهاي SQA براي حذف به موقع خطاها

حاصل کار چیست؟

يك **طرح تضمين كيفيت** براي تعريف راهبرد SQA به وجود مي آيد.

در ضمن تجزيه تحليل و كدنويسي ؛محصول كار اوليه SQA عبارت است از : گزارش خلاصه مرور تكنيكي

درخلال آزمون ، طرحها و رويه هاي آزمون توليد مي شوند

چگونه از صحت انجام آن مي توان مطمئن شد؟

[۱]يافتن خطاها ، قبل از آنكه به اشكلات عمده تبديل شوند و [۲]كاهش ميزان دوباره كاري

مفاهیم کیفیت

پديده گوناگوني بين نمونه ها ، همانند تفاوت دو دانه ي برف ، براي تمام محصولات ساخت انسان نيز صادق است حتي دو برد الكترونيكي اگر با دقت(ذره بين) بررسي شوند متوجه تفاوت در اندازه سيمهاي مسي و ... مي شويم

كنترل تنوع وگوناگوني(تفاوت) قسمت اصلی کنترل کیفیت است.یک تولیدکننده حتی وقتی به کاری ساده چون کپی کردن دیسک میپردازد تلاش دارد گوناگونی بین تولیدات به حداقل برساند اما ممکن است به دلیل تنوع بین نمونه ها بامشکل مواجه شود.

در هر پروژه اي نسبت به پروژه ديگر، ما تمايل داريم **تفاوت بين منابع پيش بيني شده براي كامل كردن يك پروژه و منابع واقعي** اي كه به كار برده شده اند شامل (افراد تيم ، تجهيزات و تقويم زماني) را به حد اقل برسانيم

ما نه تنها می خواهیم تعداد نقایصی را که وارد محیط ما شده اند به حداقل برسانیم بلکه تمایل داریم مطمئن شویم که **واریانس تعداد اشکالات نیز درهر نشر نسبت به دیگری به حداقل رسیده است.** هدف به حداقل رساندن تفاوت ها در سرعت ، دقت در پاسخگويي به مشكلات مشتري است.

کیفیت

کیفیت به عنوان خصصه چیزی به ویژگیهای قابل اندازه گیري چیزی اشاره دارد .

ويژگيهايي که ما می توانیم آن ها را با استاندارد های شناخته شده ای چون طول رنگ وغیره مقایسه کنیم.علی رغم فکري بودن نرم افزار مقیاس هایی برای ویژگیهای یک برنامه هم وجود دارد. این ویژگیها عبارتنداز:پیچیدگی ذاتي ، یکپارچگی تعدادامتیازات کارکردی خطوط برنامه نویسی و...

وقتی ما چیزی را بر مبنای ویژگیهای قابل سنجش(اندازه گيري) آن می آزماییم ممکن است با دو نوع کیفیت رو به رو شویم:

۱)کیفیت طراحی و **۲)کیفیت تطابق.**

کیفیت طراحی مربوط به آن ویژگیهایی است که طراح به چیزی اختصاص داده است .(درجه مواد،تلرانس ها ،مشخصات كارايي و...)

کیفیت تطابق میزان رعایت مشخصات طراحی به هنگام ساخت محصول است،بالاتر بودن درجه تطابق سبب کیفیت بالاتر مي شود

نام درس:مهندسی نرم افزار ۱

در توسعه نرم افزار ،

كيفيت طراحي : نيازها،مشخصه ها و طراحي سيستم***كيفيت تطابق** ابتدا بر پياده سازي تمركز دارد،اگر در پياده سازي ،كيفيت طراحي رعايت شودو بر اساس نيازها منطبق شودكيفيت تطابق بالا مي رود.[۳]عنصر مهم ديگري كه براي مهندس نرم افزار بايست در كنار كيفيت طراحي و كيفيت تطابق مهم باشد، **رضايت مشتري** است يعني:

رضايت مصرف كننده = محصول مورد قبول + كيفيت خوب + تحويل طبق بودجه و زمانبندي

يعني " كيفيت يك محصول تابعي است از ميزان تغييري كه آن محصول در بهتر شدن جهان انجام مي دهد."

كنترل كيفيت

كنترل تغير ممكن است با **كنترل كيفيت** برابر گردد.كنترل كيفيت عبارت است از **مجموعه بازرسيها بررسي و آزمون هايي** كه در روند فرايند نرم افزاری انجام ميگيرد وهدف آن تضمين تطبيق توليدهر كالا با نيازهايی است كه به خاطرآن توليد می شود.كنترل كيفيت شامل يك **حلقه بازگشتي** است كه محصول نهايي را توليد مي كند.

تركيب سنجش(اندازه گيری)+فيدبك باعث مي شود كه اگر خصوصيتهاي محصول برآورد نشد، به صورت بازگشتي فرايند ايجاد آنها تنظيم شوند در اينصورت كنترل كيفيت بخشي از فرآيند توليد است اين فعاليتها را مي توان دستي،خودكار يا تركيبي از هر دو انجام داد

تضمين كيفيت

تضمين كيفيت شامل حسابرسی وگزارش كاركردهای مديريت است.هدف تضمين كيفيت فراهم كردن **اطلاعات لازم برای مديريت به منظور اطلاع از كيفيت محصول** و در نتيجه اطمينان كامل كه كيفيت توليد در جهت براوردن اهداف است.

اگر داده هاي حاصل از تضمين كيفيت مشكلات را آشكار كنند ،وظيفه مدير به كارگيري منابع براي حل مسائل است

هزينه كيفيت

هزينه كيفيت شامل تمام هزينه هايی است كه در جهت نيل به كيفيت يا انجام فعاليت های مربوط به كيفيت است.مطالعات هزينه كيفيت جهت فراهم كردن بستری برای هزينه كيفيت فعلی شناسايی كنند و راههايی برای كاهش هزينه كيفيت و فراهم كردن يک اساس طبيعی برای مقايسه می باشد(اكترا دلار).

هزینه های کیفیت ممکن است به هزینه هایی که مربوط به [۱]**پیشگیری** ، [۲]**ارزیابی و** [۳]**شکست** میباشد تقسیم کرد.

هزینه های **پیشگیری** عبارتند از:

- برنامه ریزی کیفیت
- بررسیهای منظم فنی
- تجهیزات آزمون
- آموزش

هزینه های **ارزیابی** شامل فعالیت هایی برای آگاهی از وضعیت تولید در آغاز هر فرایند میباشد.مثالهایی از هزینه های ارزیابی:

- بازرسی درون-مرحله ای و بین-مرحله ای
- تعمیر و نگهداری تجهیزات
- انجام آزمون

- هزینه های **شکست** در صورت معلوم نشدن نقصی قبل ازتحویل محصول به مشتریان از بین مي روند.هزینه های شکست ممکن است به هزینه های **شکست درونی** وهزینه های **شکست بیرونی** تقسیم گردند .

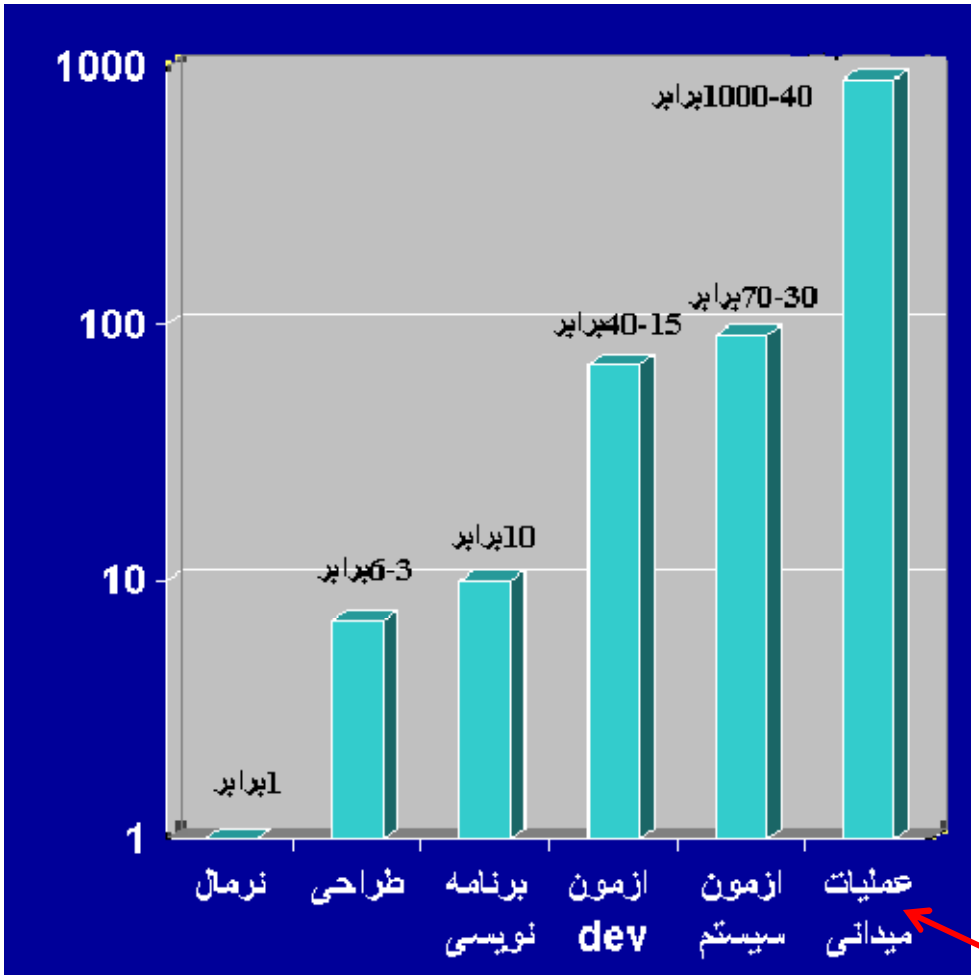
- هزینه های شکست داخلی عبارتند از:

۱) دوباره کاري ۲)تعمیر ۳)تجزیه وتحلیل وضعیت شکست

- هزینه های شکست بیرونی انهایی هست که مربوط یه نقایصی هستند که بعد از ارسال محصول به مشتری پیدا شده اند.نمونه هایی از هزینه های شکست بیرونی عبارتند از:

- هزینه رفع و رجوع شکایات و دعاوی حقوقی
- بازگشت و تعویض
- پشتیبانی و کمک های حمایتی
- انجام تعهدات تضمین داده شده

•همان طور که انتظار می رفت هزینه نسبی پیدا کردن و رفع یک نقص باحرکت مااز پیشگیری به تشخیص و سپس به هزینه های شکست داخلی و شکست خارجی به میزان قابل ملاحظه ای افزایش می یابد.



نمودار هزینه نسبی اصلاح خطا

•مثال :

•شرکت IBM بازبيني 7053 ساعته ۲۰۰ هزار خط کد و يافت ۳۱۱۲ اشکال و در کل هزينه ۲۸۲ هزار دلار براي پيشگيري

•اگر پيشگيري وجود نداشت و در شرايطي خوب فقط ۲۰۰ اشکال و ۲۵۰۰۰ دلار براي هر خطا سبب هزينه ۵ميليون دلاري براي اصلاح خطا

نام درس:مهندسی نرم افزار ۱

حرکت کیفی

امروزه مدیران مسئول کمپانی های سراسر جهان صنعتی فهمیده اند که تولید با کیفیت بالا منجر به صرفه جویی در هزینه ها ومحصول پایانی بهترمی شود. ژاپنی ها با مبناقراردادن عقاید آقاي **دمینگ** روشهاي **سیستماتیکي** را براي **حذف عوامل ریشه اي** اشکالات در محصولات توسعه دادند . کار آنها به جهان غرب منتقل شد و نامهایي چون مدیریت کیفیت یکپارچه و کامل (**TQM**) به آن داده شد اگر چه کمپانی ها و نویسندگان مختلف اصطلاحات متفاوتی در این زمینه به کار می برند اما هر برنامه TQMمعمولا با یک توالی چهار مرحله ای اصلی به عنوان زیر بنا دارد: (**مراحل TQM**)

مرحله اول: کایزن گفته میشود. سیستمی با روند پیشرفت دایمی دارد. هدف ان **به وجود اوردن فرایندی قابل مشاهده قابل تکرار و قابل سنجش واندازه گیری است.**

مرحله دوم: که پس از به نتیجه رسیدن مرحله کایزن اغاز می گردد. اتاریما هینشیتسو گفته میشود.این مرحله به بررسی عوامل پنهانی موثر بر فرآیند مي پردازد و تلاش مي کند تاثیر انها بر روند کار راخوش بینانه نماید.

- **مرحله سوم** که کانسئی(معني:حواس پنج گانه) گفته میشود به جاي فرآیند بر مصرف کننده محصول تمرکز میکند.در اساس با بررسی چگونگی کاربرد محصول از سوی مصرف کننده کانسئی به بهبود بخشیدن خود محصول می پردازد وبطور بالقوه نیزفرایندی راکه به وجود اورده است تقویت میکند.

مرحله چهارم:میریوکوتکی هینشیتسو.این مرحله ،مرحله ای تجارت گراست که به دنبال یافتن فرصت در زمینه های مشابه است.وبا مشاهده کاربرد کالا دربازار قابل شناسایی اند.در جهان نرم افزار میریوکوتکی هینشیتسو میتواند تلاشی برای آشکار کردن تولیدات و کاربردهای نو و سود آور در نظر گرفته شودکه پیامد سیستم های مبتنی بر کامپیوتر موجود میباشد.(شناسایی محصولات پرقایده) در بیشتر شرکت ها توجه به کایزن باید در اولویت باشد.(پیش نیاز سایر مراحل)

total quality management" (TQM)

تضمین کیفیت نرم افزار

تعریف کیفیت نرم افزاری عبارت است:[۱] **تطابق با نیازهای عملکردی** وکارکردی که به وضوح بیان شده اند.[۲] **استانداردهای** توسعه ای که بی هیچ ابهامی ثبت گردیده اند .و [۳] **ویژگی های تردیدناپذیری** که وجود انها درهر نرم افزار حرفه ای توسعه یافته ای انتظار میرود.از تعریف فوق برای تاکید بر سه نکته مهم استفاده میشود:

۱)نیازمندیهای نرم افزاری مبناي اندازه گيري كيفيت است .فقدان هم سویی با نیازمندیها فقدان کیفیت است.

۲)استاندارد های خاصی یک سری معیارهای توسعه را تعریف می کنند که مسیر مهندسی نرم افزار را مشخص میکند.اگر این معیار ها دنبال نگردند تقریبا به طور حتم فقدان کیفیت رخ خواهد نمود

۳)یک سری نیازمندیهای ضمنی وجود دارد که اغلب به صراحت بیان نشده اند.اگر نرم افزار با نیازمندیهای اصلي اش سازگار باشد اما نیازمندیهای ضمنی خود را براورده نسازد کیفیت نرم افزار مورد شک قرار می گیرد.

موضوعات زمینه

پیشینه تضمین کیفیت درتوسعه نرم افزار موازی با پیشینه کیفیت در ساخت سخت افزار است.در خلال **اولین** روزهای به وجود امدن کامپیوتر **کیفیت** تنها **وظیفه برنامه نویس** بود.استاندارد های تضمین کیفیت برای نرم افزار در دهه ی ۷۰ برای نخستین بار درقرارداد های نظامی توسعه نرم افزار به وجود امد و سپس به سرعت به توسعه نرم افزار در جهان تجارت گسترش یافت.با بسط تعریف ارائه شده قبلی کیفیت نرم افزار "**الگوی نظام مند و برنامه ریزی شده فعالیت هاست.که برای تضمین کیفیت نرم افزار الزامی اند.**شاید به بهترین نحو بتوان عرصه مسئولیت تضمین کیفیت را با شعار سازندگان اتومبیلی که زمانی مورد توجه همگان بود مشخص کرد:"حرف نخست را کیفیت می زند"پس گروههاي مختلف چون مهندسین نرم افزار ، مدیران پروژه،مشتري،فروشنده و افراد SQA در تضمین کیفیت نقش دارند.

گروه SQA به عنوان نماینده دایم مشتری عمل مي نمایند یعني اجرا کنندگان SQA بایست از دید مشتری به نرم افزار نگاه کنند

فعالیت های تضمین کیفیت نرم افزار(SQA)

تضمین کیفیت نرم افزار از انواع گوناگونی از وظائف تشکیل یافته است که به دو گروه متفاوت مربوط می باشد.

۱. **مهندسان نرم افزار** که کار فنی انجام می دهند
۲. **یک گروه SQA** که مسئولیت برنامه ریزی تضمین کیفیت نظارت ثبت و ضبط تجزیه وتحلیل و گزارش را بر عهده دارند.

مهندسین نرم افزار با بکارگیري روشهاي فني مشخص و معيارها؛اجراي آزمونهاي نرم افزاري خوب کیفیت را مورد توجه قرار دارند
فعالیت گروه SQA،هدایت تیم نرم افزار براي دستيابي به محصول نهايي با کیفیت بالا مي باشد
فعالیتهای SQA توسط گروه مستقلي انجام مي شود که

۱)**طرح SQA** برای پروژه اماده می کند.طرح در مدت برنامه ریزی پروژه به وجود می اید و توسط تمام گروه های علاقه مند **بررسی** می شود.طرح فعالیت های تضمین کیفیت توسط تیم مهندسی نرم افزار اجرا می شود و گروه SQA را مشخص مي کند.طرح به تشخیص موارد ذیل می پردازد:

- ارزشیابی که باید انجام گیرد.
- استانداردهایی که درمورد پروژه قابل اعمال اند.
- مستنداتی که باید توسط گروه SQA به وجود اینند.
- حسابرسی ها و بازبینی هایی که باید انجام گیرد.
- روال های گزارش و رد گیری خطاها.
- مقدار باز خوردی که به تیم پروژه نرم افزاری بازگردانده شده است.

۲)**در توصیف فرآیند پروژه نرم افزاري شرکت مي کند:**توصیف فرآیند براي سازگاري با استانداردهاي داخلي نرم افزار و استانداردهاي شديدا الزامي خارجي (تحميلي)

۳)**بررسي تطابق فعاليتهاي مهندسي نرم افزار و فرآيند:**انحرافها مستند سازي مي شوند و پي گيري و بازبيني براي اصلاح

۴)**بررسي تطابق محصول کاري نرم افزار با محصول کاري تعيين شده:**انحرافها مستند سازي مي شوند و پي گيري و بازبيني براي اصلاح

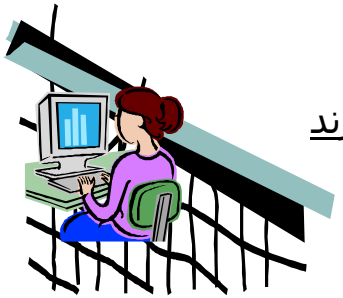
۵)حصول اطمينان از مستندسازي انحرافات و طبق يك روال مستند اداره مي گردند

۶)هرنوع ناهماهنگي ثبت شده و به مدير ارشد (عالي) گزارش مي دهد.

باز بینی های نرم افزار

۱.بررسي هاي نرم افزار "**فيلتري**"براي فرايند مهندسي نرم افزار مي باشند.يعني در خلال توسعه نرم افزار در نقاط مختلف بررسيها اعمال ميشوند واز انها براي **شناسايي خطاها و نقص ها** و سپس **برطرف كردن انها** استفاده مي شود.بررسي هاي نرم افزاري به منظور"**پاك كردن**" فعاليت هاي مهندسي نرم افزاري كه ما انها را تجزيه و تحليل طراحي و برنامه نويسي مي گوييم اعمال ميشوند.و براي كارفني مثل **پاك كن براي مداد** هستند.

۲.معمولا افراد در پيدا كردن خطاهاي خودخوب عمل مي كنند ولي دسته اي از خطاها خيلي آسانتر از ديد كننده كار پنهان مي مانند



يك بررسي راهي است براي استفاده از مسيرگروهي از مردم به منظور:

۱)نشان دادن اصلاحات لازم در توليد توسط يك فرد تنها يا يك گروه

۲)تعيين آن بخش هاي از توليد كه يا علاقه اي به اصلاح انها وجود ندارد يا اصولا نيازي به اصلاح شدن ندارند

۳)دست يابي به كار فني يك دست تر يا دست كم قابل پيش بيني تر به منظور بالا بردن قابليت

_ مديريت كار كه در نتيجه ان مي توان بدون انجام بررسي به كيفيت دست يافت.

موثرترين روش فيلتر ، روش بررسي فني رسمي (وارسي يا بازرسي) (FTR) و باعث ايجاد كيفيت مي شود.

تاثير عيوب بر هزينه نرم افزاری

در واژه نامه ها IEEE: **نقص** عبارت است از يك ناهنجاري در محصول و **عيب** هم تعريف شده است از اشكالي در دستگاه يا يك مرحله فرايندیا تعريف داده اشتباه

در نرم افزار اين دو واژه مترادف و هم معني هستند و به وجود اشكال كيفي آشكار شده بعد از تحويل محصول اشاره دارند.لازم به يادآوري است .واژه **خطا** ، قبلا براي اشاره به مشكلي در كيفيت كه قبل از تحويل آشكار شده استفاده شد.

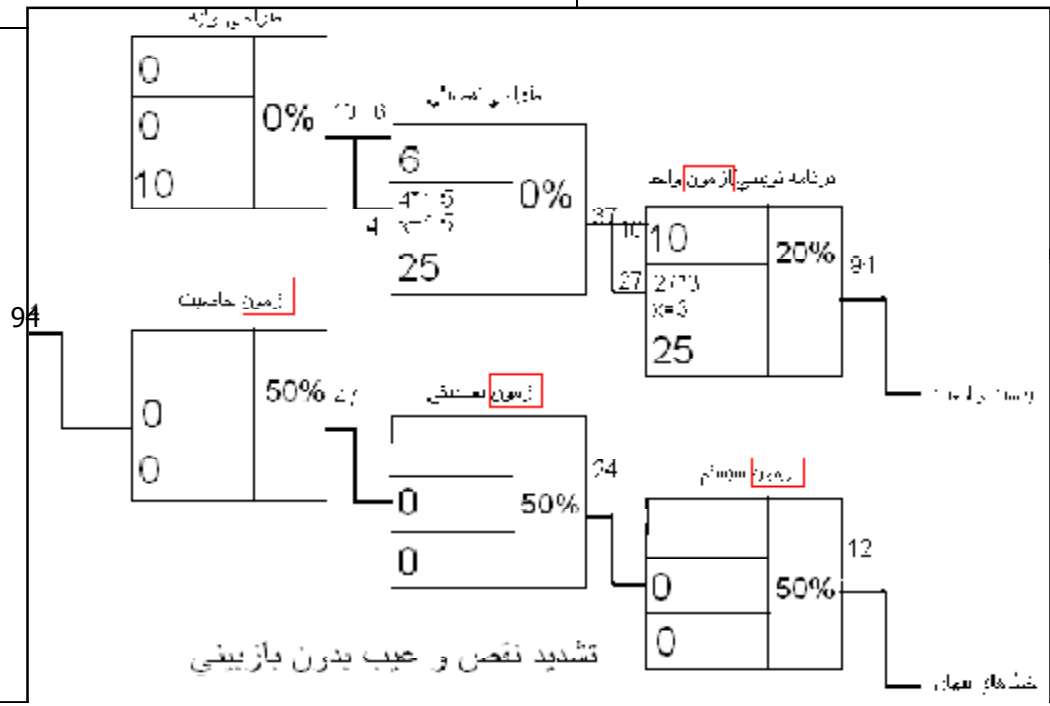
اولين هدف بررسي هاي فني رسمي (FTR)يافتن خطاها در جريان فرايند است به طوري كه انها بعد از عرضه نرم افزار تبديل به نقص نگردد.مزيت اشكار بررسي هاي فني رسمي كشف زود هنگام خطاست به طوري كه انها نتوانند به مرحله بعدي در فرايند نرم افزار نشر يابند.تعدادي از مطالعات صنعتي نشان مي دهد كه فعاليت هاي طراحي بين ۵۰ تا ۶۵ درصد خطاها را در خلال فرايند نرم افزاري ايجاد مي كنند.با اين حال نشان داده شده است كه فنون بررسي رسمي تا ۷۵ درصد در اشكار كردن نقايص طراحي موثر مي باشد.با شناسايي و رفع درصد بالايي از اين خطاها فرايند بررسي ها به طور عمده اي هزينه مراحل بعدي توسعه و تضمين را كاهش داده است.

formal technical review

تشديد نقص و برطرف كردن آن

يك **مدل توسعه نقص** براي **نمايش توليد و رد يابي خطاها** در جريان طراحي اوليه ،طراحي تفصيلي و جزييات و مراحل برنامه نويسي فرايند مهندسي نرم افزار قابل كاربرد است.اين مدل به صورت نمودار در تصوير زير نشان داده شده است.

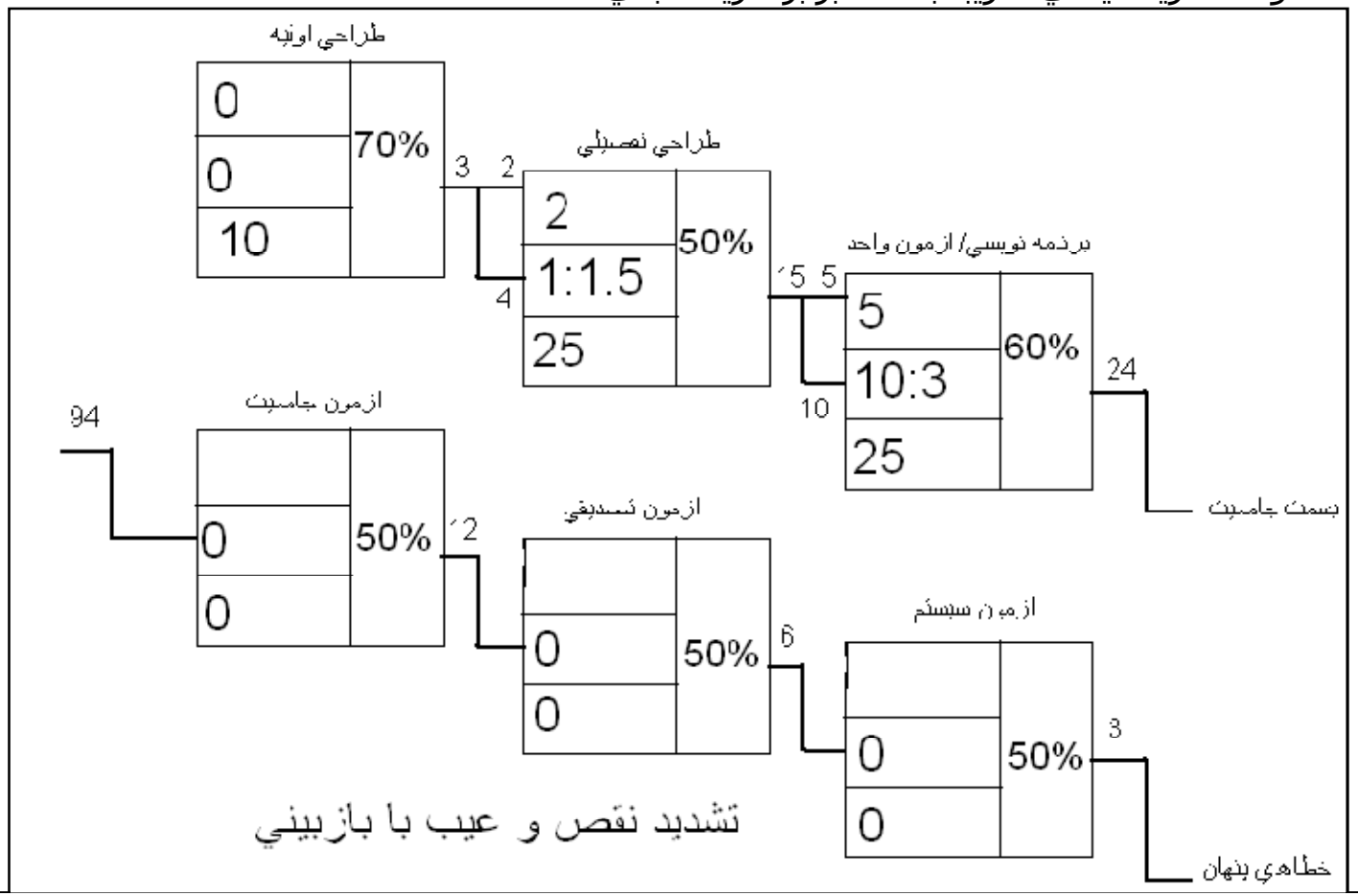
هر جعبه يكي از مراحل توسعه نرم افزار را نشان مي دهد.در جريان هر مرحله خطاها ممكن است سهوا توليد شوند.بررسي ها ممكن است نتوانندخطاهاي جديد و يا خطاهاي قبلي را اشكار كنندو در نتيجه تعدادي خطا را دوباره فعال كند.در بعضي موارد خطاهاي باقي مانده از مراحل قبلي به وسيله كار كنوني **گسترش** مي يابند.تقسيمات فرعي جعبه ها نمايانگرهر يك از اين ويژگي ها ودرصد كارايي خطاهاي رديابي شده مي باشد كه كاركردي از اجراي كامل بررسي ها مي باشد.



•تصوير زير يك مثال فرضي از توسعه و تشديد نقص در يك فرايند توسعه نرم افزاري را نشان مي دهد كه دران هيچ گونه بررسي انجام نشده است.با توجه به تصوير فرض مي شود كه هر يك از مراحل ازمون براي اشكار سازي و تصحيح ۵۰ درصد از همه خطاهاي كنوني بدون توليد خطاهاي جديد به كار ميرود.قبل از اغاز مراحل ازمون ده نقص طراحي اوليه به ۹٤ خطا گسترش يافته و تشديد شده اند.دوازده خطاي پنهان نيز وارد محيط كار شده اند.

نام درس:مهندسی نرم افزار ۱

❑تصوير زير نيز شرايطي مشابه را مورد ملاحظه قرار مي دهد با اين استثنا كه بررسي طراحي و برنامه به عنوان بخشي از هر مرحله انجام گرفته است .در اين مورد ده خطاي طراحي اوليه قبل از اغاز ازمون ها به ٢٤ خطا افزايش يافته اند-تنها سه خطاي پنهان وجود دارد.با به ياداوردن هزينه هاي نسبي مربوط به كشف و تصحيح خطاها ارزش كلي قابل بيان خواهد بود.تعداد خطاهاي اشكار شده درهر يك از مراحل مورد اشاره تصاويربا هزينه برطرف كردن هر خطا افزايش يافته است.با به كار بردن اين داده ها هزينه كلي توسعه و نگهداري در جريان انجام بررس ها برابر است با ٧٨٣ واحد هزينه.اگر هيچ گونه بررسی اي انجام نگرفته باشد هزينه كلي برابر خواهد بود با ٢١٧٧ واحد هزينه-يعني تقريبا با سه برابر هزينه قبلي.



نام درس:مهندسی نرم افزار ۱

بازبینی های فنی–رسمی

یک بررسی فنی _ رسمی (FTR) یک فعالیت تضمین کیفیت است که توسط مهندسان نرم افزار(ودیگران) اجرا می شود. FTR درواقع گروهی ازبررسی هاست که **شامل جستجوها، بازرسی ها،بررسی های مخفی ودیگرارزیابی های فنی گروههای کوچک نرم افزاری است.** اهداف آن عبارتند از:

۱. آشکارکردن خطاهای موجود درکارکرد ، یا اجرای هرنوع ارائه نرم افزار.
۲. بررسی این نکته که نرم افزارتحت بررسی با نیازمندی هایش هم خوانی دارد.
۳. اطمینان ازاینکه نرم افزار بر اساس استانداردهای از پیش تعریف شده ارائه شده است.
۴. دستیابی به نرم افزاری که به شیوه ای همگن(یکنواخت) تولید شده باشد.
۵. افزایش قابلیت اداره پروژه ها.

علاوه براین FTRبه عنوان یک زمینه آموزشی به کار می رود، هم چنین موجب تقویت پیشرفت و انسجام کار می شود.

نشست باز بینی

هر FTR به صورت يك جلسه هدايت مي شود، صرف نظرازقالبی که برای FTR انتخاب شده است ، هرگردهم آیی بررسی باید به وسیله قیود ذیل به پیش رود.

- معمولا سه تا پنج نفر در این بررسی دخالت داشته باشند
- آمادگی قبلی باید وجود داشته باشد.
- مدت زمان گردهم آیی بررسی بایدکمتر از دو ساعت باشد.

FTR بر روي كل نرم افزار تاكيد ندارد بلكه **براي هر محصول مياني به طور مجزا برپا مي گردد**

تاكيد FTR بر **محصول كاري است** و با اتمام ساخت يك محصول كاري از طريق مسئول پروژه از مسئول تيم مرور درخواست جلسه مي شود . اين جلسه با حضور [۱]سازنده محصول كاري، [۲]مرورگرها و [۳]مسئول بررسي شروع مي شود كه يكي از مرورگرها نقش منشي و ثبت كننده را به عهده دارد

در انتهاي جلسه يك گزارش نتيجه گيري نوشته مي شود .

قبول محصول بدون اصلاح ، رد کردن محصول ، قبول محصول با در نظرگرفتن یکسری نکات

گزارش بازبینی ها و ثبت موضوعات

منشي و ثبت كننده تمام موضوعات مطرح شده در جلسه مرور را ثبت مي كند و يك لیست از **موضوعات** مطرح شده و يك **گزارش** ايجاد مي شود كه اين گزارش مختصری از بررسی فنی رسمی امور تهیه می گردد.این گزارش به سه سوال پاسخ می دهد:

۱. چه چیزی بررسی شده است؟
۲. چه کسی آن را بررسی کرده است؟
۳. نتایج یافته ها چه چیزی بودند؟

ليست موضوعات بررسي شده نيز به دو منظور به كار مي رود:

۱.شناسايي زمينه هاي مشكل دار محصول
۲.به عنوان چك ليست عملكردها براي راهنمايي توليد كننده

رهنمودهای بازبینی

يك بررسي FTR بايست درست و كنترل شده باشد . استفاده از رهنمودهاي زير براي انجام FTR توصيه مي شود:

۱. **محصول را بررسی کنید نه تولیدکننده را:**اين كار در فضاي آرام و سازنده باشد نه فضاي خشك و بي روح
۲. **یک دستور کار تنظیم و آن را حفظ کنید:** ايجاد دستور كار براي زمانبندي و جلو گيري از انحراف
۳. **منازعه و جدل متقابل را محدود کنید.**
۴. **زمینه های دارای مشکل را اعلام دارید،اما سعی نکنید تمام مشکلات اعلام شده را حل کنید.**
۵. **یادداشت های کتبی تهیه کنید:**ثبتي ها در معرض ديد همه باشند
۶. **تعدادشرکت کنندگان را محدود کنید وبرآمادگی قبلی تاکید کنید.:**جلسه ۱۴ نفري نشانه برتري بر جلسه ۴ نفري نيست
۷. **لیستی برای هر محصول که احتمال بررسی آن می رود تهیه کنید.**
۸. **منابع و زمان لازم به FTRتخصیص دهید.**
۹. **آموزش های هدفمندی برای همه بررسی کنندگان(مرورگرها) فراهم کنید.**
۱۰.**بررسی های اولیه ی خود را بررسی کنید.**(خلاصه سازی بررسی ها)

رهیافتهای(روش) رسمی برای تضمین کیفیت نرم افزار(SQA)

کیفیت نرم افزار برعهده همه افراد است که درمراحل تحلیل و برنامه سازی و آزمون همان قدرباید مورد توجه باشد که در باز بینی های رسمی فنی ، در راهبرد های تلاش چندگانه آزمون و کنترل بهتر محصولات کاری و تغییر آنها ودریافت استانداردهای مهندسی نرم افزار کیفیت می تواند به طیف گسترده ای از عوامل کیفی و اندازه های غیرمستقیم تعبیر شود که شاخص ها ومتریک های متعدد آن را نشان می دهند . یک بخش کوچک اما ویژه ازجامعه نرم افزار معتقدند روشي رسمي براي تضمين كيفيت نرم افزار مورد نیاز است آنها معتقدند که یک برنامه کامپیوتری یک شی ریاضی محسوب می شود و یک **نحو و معنای ثابتی** میتواند در هر زبان برنامه سازی تعریف شود که در این صورت رهیافت ثابتی نیز برای مشخصه های نیازمندی های نرم افزاری در پیش رو خواهد بود.

تضمین کیفیت نرم افزار به صورت آماری(SSPI)

تضمین کیفیت آماری منعکس کننده يك روند رو به رشد در صنعت برای **کمی نمودن کیفیت** است در مورد نرم افزار تضمین کیفیت آماری متضمن مراحل زیر است:

۱. **اطلاعات درباره نقص های** نرم افزار جمع آوری و دسته بندی می شود.
۲. تلاشی برای ردیابی **علت زیربنایی(اصلی) هر نقص** انجام می گیرد.
۳. اصول **پارتو** به کار بسته می شود.(۸۰ درصد مشکلات در ۲۰ درصد علل قرار دارند) تا ۲۰ درصد **شناسایی** شود
۴. در حین شناسایی علل حیاتی (مرحله قبل) حرکت به سمت **اصلاح** مسائلی که ریشه نواقص بوده اند انجام می گیرد

به منظورکاربرد SQAآماری جدول زیر طراحی شده است.این جدول نشان می دهد که MCCو IESو EDRبه عنوان دلایل اصلی انتخاب خواهندشد درحالیکه ۵۳ درصد تمام خطاها هستند. بعداز اینکه دلایل اساسی تعیین گردیدند سازمان مهندسی نرم افزاری ممکن است برای بهبود کیفیت مشتری و مشخصات تعیین شده، به جای اجرای فنون مشخصات کاربرد تسهیل شده بپردازد.

- incomplete or erroneous specifications (IES)
- misinterpretation of customer communication (MCC)
- intentional deviation from specifications (IDS)
- violation of programming standards (VPS)
- error in data representation (EDR)
- inconsistent component interface (ICI)
- error in design logic (EDL)
- incomplete or erroneous testing (IET)
- inaccurate or incomplete documentation (IID)
- error in programming language translation(PLT)
- inconsistent human/computer interface (HCI)
- miscellaneous (MIS)

	کل		مهم		متوسط		جزئی	
نوع خطا	تعداد	درصد	تعداد	درصد	تعداد	درصد	تعداد	درصد
IES	205	22%	34	27%	68	18%	103	24%
MCC	156	17%	12	9%	68	18%	76	17%
IDS	48	5%	1	9%	24	6%	23	5%

نکته : فعالیت اصلاحی دروهله نخست **برعلل اساسی** متمرکزاست، با تصحیح علل اساسی،مسایل جدیدی برای تصحیح شدن پیدا می شود. فنون تضمین کیفیت آماری برای نرم افزار،نشان دهنده فراهم آوردن بهبود کیفیت آماری است.درکنارجمع آوری اطلاعات درباره نقایص کار سازندگان نرم افزار می تواند یک شاخص خطا (EI) برای هر یک ازمراحل اصلی فرآیندمحاسبه کنند.

بعد از تجزیه و تحلیل ، طراحی ، برنامه نویسی ، آزمودن و عرضه، اطلاعات زیر جمع آوری شده اند.

Ei : تعدادکل خطاهایی که درخلال مرحله i ام روند مهندسی نرم افزار

Si : تعداد خطاهای اساسی Mi : تعداد خطاهای میانه

Ti : تعدادخطاهای کوچک PS : اندازه محصول(از لحاظ تعداد خطوط،ریال،مستندات و...) در مرحله i ام

Wt,Wm,Ws : ضریب ارزش خطاهای اساسی – متوسط و جزیي

در هر مرحله ازفرآیند نرم افزاری یک شاخص مرحله PIi محاسبه می شود:

$$PIi = Ws (Si / Ei) + Wm (Mi / Ei) + Wt (Ti / Ei)$$

شاخص خطا Ei بامحاسبه اثرانباشتی(تجمعی) PIi محاسبه شده است:

$$EI=\sum (i * PIi) / PS=(PI1+2PI2+3PI3+...+iPIi)/ PS$$

برای توسعه یک نشان کلی ازبهبود به وجود آمده درکیفیت نرم افزاری می توان شاخص خطا رابه همراه اطلاعات گردآوری شده در جدول قبل به کار برد. کاربرد SQA آماری وقاعده پارتو را در یک جمله واحدمی توان خلاصه کرد: زمان خود راصرف تمرکز بر چیزهایی کنید که حائز اهمیت هستند،امانخست مطمئن شویدکه شماواقعا درک می کنید چه چیزی مهم است.

قابلیت اطمینان (اعتبار) نرم افزار

قابلیت اطمینان برای یک برنامه عنصر مهمی در کنترل کیفیت نرم افزار می باشد.اعتبارنرم افزاربرخلاف بسیاری دیگرازفاکتور های کیفیت با به کاربردن اطلاعات تاریخی و توسعه ای به طور مستقیم قابل اندازه گیری است ومی توان آن رابرآورد کرد.

در اصطلاحات آمار اعتبار نرم افزاری به صورت " **احتمال عملکرد بدون خرابی یک برنامه کامپیوتری دریک محیط خاص و در یک زمان مشخص**".

به عنوان نمونه : برنامه X برآورد شده است و دارای اعتبار ۰.۹۶ است درطول ۸ ساعت فرایند سپری شده.

مفهوم :اگربرنامه Xبرای ۱۰۰ بار به اجرا در آمدن طراحی شده باشد ونیاز به ۸ ساعت زمان فرایند سپری شده داشته باشد (زمان اجرا) احتمال دارد که ۹۶ بار از ۱۰۰ بار مورد انتظار را به طور صحیح عمل نماید.

خرابی چیست؟ دریافت هربحثی درباره کیفیت و اعتبارنرم افزاری، خرابی یا ناسازگاری با نیازمندی های نرم افزار .

خرابی دارای درجه بندی می باشد . دو درجه بندی خرابی عبارتند از :

۱. **اذیت کننده:** در چند ثانیه تصحیح می شود.
۲. **مصیبت بار:** نیاز به هفته ها یا حتی ماهها برای تصحیح شدن دارد.

اندازه گیری قابلیت اطمینان(اعتبار) و دسترسی

Measures of Reliability and Availability

اندازه گیری قابلیت اطمینان(اعتبار)

کارهای اولیه برروی اعتبارنرم افزاری تلاش داشتند تا ریاضیات تئوری اعتبار سخت افزاری را برای پیش بینی اعتبار نرم افزاری به کار برند.

خرابی ها ناشی از	سخت افزار	نرم افزار
کهنگی و فرسودگی	قسمت اعظم	نقش کمرنگ
مشکلات طراحی یا پیاده سازی	نقش کم رنگ	نقش اساسی

اگر یک سیستم کامپیوتری را مورد مطالعه قرار دهیم اندازه گیری ساده اعتبار و قابلیت اطمینان عبارت خواهد بود از:

MTBF = MTTF + MTTR

MTBF(***meantime-between-failure***) : متوسط زمان بین دو شکست

(**mean-time-to-failure**) MTTF : فاصله زمانی بروز یک اشکال

(**mean-time-to-repair**) MTTR : فاصله زمانی برطرف کردن آن اشکال

بسیاری محققان معتقدند که MTBF یک اندازه گیری بسیار مفیدتر از خطاها درKLOC یا خطاها در FPاست. آن ها این بیان ساده رامطرح می کنند که یک مصرف کننده نهایی باخرابی ها سروکاردارد نه محاسبه خطای کلی.

از آنجایی که **هر خطایی** که دریک برنامه موجود است دارای **نرخ خرابی** یکسانی نمی باشد، محاسبه خطای کلی نشان اندکی از اعتبار سیستم به دست می دهد.

اندازه گیری میزان دسترسی

قابلیت دسترسی : قابلیت دسترسی نرم افزار احتمال عملکرد یک برنامه بر طبق نیازمندیهای خاص در یک نقطه زمانی است وبه صورت **Availability = [MTTF/(MTTF+MTTR)]*100%** تعریف می شود:

در میزان اعتبار MTBF زمانهاي MTTF و MTTR به يك ميزان موثرند.میزان قابلیت دسترسی تا اندازه بیشتری به MTTR حساس است،که میزان غیرمستقیمی از قابلیت نگهداری نرم افزار است.

ایمنی نرم افزار

ایمنی نرم افزار یک فعالیت تضمین کیفیت نرم افزار است که بر**شناسایی و ارزیابی خطرات بالقوه ای** ممکن است بر روی نرم افزار منفی داشته باشند وموجب خرابی یک سیستم کلی گردند تمرکز دارد یک فرایند مدل سازی وتجزیه وتحلیل به عنوان بخشی از ایمنی نرم افزار اجرا شده است.

قدم اول : شناسایی خطرات .(مثلا ورود حرف به جاي عدد در فيلد تاريخ)

قدم دوم : به کارگرفتن فنون تجزیه و تحلیل برای تعیین شدت یا احتمال اتفاق افتادن خطرات .

قدم سوم : تعیین نیازمندی های مربوط به ایمنی نرم افزار .

تفاوت اعتبارنرم افزار(شکستها) و ایمنی نرم افزار(خطرات)

اگرچه اعتبارنرم افزاروایمنی نرم افزارارتباط نزدیکی باهم دارند، درک تفاوت ظریف بین آنها دارای اهمیت است. اعتبار و قابلیت اطمینان نرم افزار برای تعیین احتمال این که یک خرابی نرم افزاری اتفاق بیفتد ازتجزیه وتحلیل آماری استفاده می کند، بااین وجودرخ دادن یک خرابی الزاما منجر به یک خطرا اتفاق سوء نمی شود.

ایمنی نرم افزار به آزمون راههایی می پردازدکه ازطریق آنها خرابی ها منجر به شرایطی می شوندکه ممکن است به یک اتفاق سوء بیانجامد،این بدان معنی است که خرابی هادرخلاءدرنظر گرفته نشده اند،بلکه دربافت سیستمهای کامپیوتری ارزشیابی شده اند.

مصونیت نرم افزاردربرابراشتباه(نرم افزارضد خطا)

روش پوکا یوک(ضد اشتباه) یک تکنیک تضمین کیفیت که نتیجه آن **پیشگیری یا پیش گیري و تصحیح زود هنگام خطاها** درروند ساخت است ازابزارهای پوکا-یوک (مصونیت از اشتباه) استفاده می کند.مکانیزمهایی که منجربه (۱) **جلوگیری** از مسائل کیفی بالقوه قبل از رخ دادن آنها شد (مثلا ماشین در صورت نبستن تمام درها حرکت نکند)یا (۲) **تشخیص** سریع مسائل کیفی درصورتی که واردکارشده باشند، می شد(مثلا درصورت کمبود بنزین آلارم داده شود....).یک ابزارپوکا-یوک کارآرادارای یک سری ویژگیهایی است:

۱. ساده و ارزان باشد.

۲. بخشی از فرآیند مهندسي است است.

۳. درنزدیکی وظیفه اي ازفرآیند که اشتباهات رخ می دهند.که باعث فیدبك سریعي مي شود و صرفه جويي در زمان کشف مشکل

استانداردهای کیفیت ایزو ۹۰۰۰

ايزو ۹۰۰۰ ، **عناصر تضمين كيفيت** را به صورت **عمومي** توصيف مي كند كه مي توانند براي هر شغلي و با هر نوع محصولات و سرويسهايي كه ارائه مي دهند به كار گرفته شود

این استاندارد که توسط بیش از ۱۳۰ کشور پذیرفته شده است به طور مستمر بر اهمیت آن به عنوان ابزاری که به وسیله آن مشتریان می توانند توانایی سازندگان نرم افزاررامورد قضاوت قراردهند افزوده شده است.یکی ازمشکلات(؟؟؟) مربوط به سری استانداردهای Iso9001مربوط نبودن آنها به صنعت خاصی است.این استاندارد با اصطلاح های عمومی بیان گردیده و به وسیله تولید کنند گان محصولات مختلفی قابل تعبیر است.

رهیافت ایزو جهت سیستم های تضمین کیفیت

برای صنعت نرم افزار استانداردهای مربوط عبارتند از :

- **Iso9001** مدلی برای تضمین کیفیت در طراحی ، توسعه، تولید، نصب و خدمات دهی است.
- **Iso9000** توصیه هایی است برای به کارگیری Iso9001 در کار توسعه و در عرضه و نگهداری نرم افزار وسند خاصی است که Iso9001 رابرای تولیدکننده نرم افزار معنی می کند.
- **Iso9004** مدیریت کیفیت وعناصرسیستم کیفیت.و حمایت از مصرف کننده

ايزو ۹۰۰۱ در نرم افزار شامل ۲۰ نيازی است كه بايد برای يک سيستم تضمين كيفيت موثر وجود داشته باشد اين نيازمنديها در زير ۲۰ عنوان گروه بندی شده است: مسئوليت مديريت، تجهيزات آزمون و اندازه گيری بازرسی، سيستم كيفيت، بازرسی وضعيت آزمون،بررسی قرارداد، اقدام اصلاحی، كنترل طرح ، كنترل برمحصول ناسازگار، كنترل سند، جابجايی، انبار، بسته بندی وعرضه،خريد،ثبت كيفيت،محصول عرضه شده خريدار، بازرسی درونی كيفيت،قابليت پيگيری و تشخيص توليد، آموزش، كنترل مرحله ای،سرويس كردن،بازرسی و آزمون ، فنون آماری.

استاندارد ایزو ۹۰۰۱

به منظورحفظ مطابقت محصول بانيازمنديهای ساخت آن،توليدكننده بايد به كنترل، تعيين وحفظ بازرسی، اندازه گيری وفراهم كردن وسايل آزمون محصول بپردازد.

اولين چيزی كه بايد بدان توجه كرد**عموميت** آن است،آن رامی توان در مورد توليد كننده هرمحصولی به كاربرد.دومين چيزی كه بايدبه آن توجه كرد **دشواری تعبير پاراگراف** است. (اين پاراگراف به روشنی بر مراحل مهندسی استاندارد و همزمان به كيفيت ابزارها و تجهيزات تاكيد دارد)

طرح تضمین کیفیت نرم افزار SQA

این طرح که به وسیله گروه SQA به وجود آمده **راهي را براي ايجاد تضمين كيفيت نرم افزار** فراهم مي كند و به عنوان **الگویی** برای فعالیتهای SQA ارائه می شوند که برای هر پروژه نرم افزاری برقرار گردیده اند.

مثلا ISO9000 طرحي است براي ISO9001

استانداردی به وسیله IEEE برای **طرح های SQA** پیشنهاد شده است.

۱-بخش مدیریت : طرح جایگاه SQA را درساختارسازمانی ، کارهای SQA وفعالیتهاوجایگاهشان درفرآیندنرمافزاری، و نقشهای سازمانی ومسئولیتهای مربوط به کیفیت محصول را توصیف می کند.

۲-بخش مستندسازی (رجوع) : به هریک از محصولات کاری که به عنوان بخشی از فرآیند نرم افزاری تولید شده اند به توصیف آن ها می پردازد. این موارد عبارتند از :

- اسناد پروژه (مثلا، طرح پروژه)
- مدلها (مثلا، ERDها ، سلسله مراتب طبقه بندی)
- اسناد فنی (مثلا، مشخصات ، طرح های آزمون)
- اسناد کاربرد (مثلا،فایلهای کمک و راهنمایی)

علاوه براین مجموعه حداقل محصولات کاری را معرفی می کند.

۳-بخش بررسی ها و وارسی های طرح : بررسی ها و وارسی هایی را که توسط تیم مهندسی نرم افزاری، گروه SQAو خریدارباید به عمل آیند مشخص می کند .

۴-بخش آزمون : این بخش به روال طرح آزمون نرم افزارنظردارد . همچنین نیازمندیهای نگهداری نتایج ثبت شده آزمونها را مشخص می کند، گزارش مشکل و اقدام اصلاحی، مراحل گزارش، پیگیری وبرطرف کردن اشکالات ونقایص رامشخص می کند،ومسئولیتهای سازمانی دربرابراین فعالیتها را مشخص می کند.

قسمتهای دیگر طرح SQAابزارهاوروشهای تقویت فعالیتهاوکارهای SQAرا مشخص می کند.

خلاصه فصل هشتم(تضمین کیفیت)

تضمین کیفیت نرم افزاری یک " فعالیت چتری" است که در هریک از مراحل فرآیند نرم افزاری به کار بسته شده است. SQA در بردارنده مراحل به کار گیری موثر روشها، بررسی های فنی منظم، تکنیکها وراهبردهای آزمون، ابزارهای پوکا – یوک، روالهایی برای کنترل تضمین مطابقت بااستانداردها،وسنجش مکانیزمهای گزارش کردن می باشد.

طبیعت پیچیده کیفیت نرم افزاری موجب پیچیدگی SQA شده است. یکی از خواص برنامه های کامپیوتری که به عنوان " مطابقت با نیازمندی های تعریف شده صریح وضمنی " تعریف شده است. اما وقتی کیفیت نرم افزاری رابصورت عمودی تعریف کنیم بسیاری از محصولات مختلف، فاکتورهای پردازشی ومتریک های مربوطه را دربرمی گیرد. بررسی های نرم افزاری یکی ازمهم ترین فعالیتهای SQA می باشند. بررسی ها به عنوان فیلترهایی در تمام فعالیت های مهندسی نرم افزاری عمل می کنند ، که باعث برطرف شدن خطاها ، زمانی که آنها برای یافتن وبرطرف کردن گسترش زیادی نیافته اند، خواهند شد . بررسی فنی منظم یک گردهم آیی روش مند است که معلوم شده، برای مشخص کردن خطاها بسیار موثر است.

برای تضمین مناسب کیفیت نرم افزار،اطلاعاتی درباره فرآیند مهندسی نرم افزار باید گردآوری، ارزشیابی و پخش گردد . SQAآماری بر بهترکردن کیفیت محصول و خود فرآیند نرم افزار کمک می کند. مدلهای اعتبار نرم افزاراندازه گیریهایی را بسط می دهد که برآورد و ارزیابی داده های گردآوری شده درمورد نقایص را در قالب نرخ های ناکامی موجوددرطرح وپیش بینیهای اعتبار، ممکن می سازد.

در پایان سخنان دان و آلمن را بازگو می کنیم : " تضمین کیفیت نرم افزاری ترسیم مقررات مدیریتی و اصول طراحی تضمین کیفیت در فضای قابل کاربرد فنی و مدیریتی مهندسی نرم افزار است ".

فصل ۹: مدیریت پیکربندی نرم افزار *Software configuration management (SCM)*

❑مدیریت پیکربندي نرم افزار چیست؟

تغییرات در فرايند ساخت نرم افزار اجتناب ناپذير است.تغییرات باعث **سردرگمي مهندسين و متخصصين** مي شوند.
مدیریت پیکربندی نرم افزار فعاليتي جامع است که در طول فرایند نرم افزاری اجرا می گردد(چون تغییرات در طول فرايند اتفاق مي افتند) وداراي اهداف و مراحل زير است:

۱-شناسایی تغییرات
۲-کنترل تغییرات
۳-کسب اطمینان از حفظ کیفیت،با انجام تغییرات
۴-گزارش تغییرات به افراد دستندرکار و علاقمند

❑چرا اهمیت دارد؟

اگر شما تغییرات را کنترل نکنید، تغییر شما را کنترل مي کند وجود تغییرات باعث **سردرگمي** مي شود و در روند انجام عمليات در يك پروژه موفق اختلال ايجاد مي كنند

❑مراحل انجام آن کدامند؟

۱-شناسايي تغییرات ممکن در محصولات کاري
۲-مکانیزمهاي **کنترل** تغییر
۳-حصول اطمینان از **حفظ کیفیت** حتي با انجام شدن تغییر
۴-گزارش تغییرات به کساني که بايست بدانند

❑محصول کار چیست؟

طرح مدیریت پیکربندی نرم افزار (CSR)

❑تفاوت بین پشتیبانی ازنرم افزار ومدیریت پیکربندی نرم افزار

پشتیبانی یعنی مجموعه ای از فعالیت های مهندسی نرم افزار که پس از تحویل نرم افزاربه مشتری واستفاده از آن به وقوع می پیوندد.مدیریت پیکربندی نرم افزار یعنی مجموعه ای از فعالیت های پیگیری وکنترل که زمانی شروع می شوند که یک پروژه مهندسی نرم افزاری آغاز می گردد وتنها زمانی پایان می یابند که نرم افزار از کارمی افتد.

مدیریت پیکربندی نرم افزار (SCM):

خروجی فرایند نرم افزاری اطلاعاتیست که می توان آن را به سه گروه تقسیم کرد:

- ❖ برنامه های کامپیوتری
- ❖ مستندات که برنامه های کامپیوتری را شرح می دهد
- ❖ داده ها(در داخل یا خارج برنامه)

❑ **عوامل ایجاد سردرگمي**

۱-در حال انجام فرايند تعداد آيتم هاي پيكربندي نرم افزار(SCI) افزايش مي يابد

مستندات مختلف ← طرح پروژه و نیازها ← مشخصه سیستم

توليد SCI جديد در اثر هر SCI باعث سر در گمي مي شود

۲- متغیردیگری درفرایند نرم افزاری وجود دارد بنام "**تغییر**". تغییر می تواند در هر زمانی وبه هردلیلی روی دهد.درواقع اولین قانون مهندسی سیستم بیان می دارد که:مهم نیست که شما درکجای چرخه زندگی سیستم قرار دارید، سیستم تغییر خواهدکردوتمایل به تغییر آن در تمام چرخه زندگی ادامه پیدا خواهد کرد.

❑ **چهارمنبع تغییر اساسی وجود دارد:**

۱) تغییر در ***نیازهاي محصول یا قوانین کار*** در اثر شرایط جدید کاري یا تغیرات بازار (نه تقصیر ما ، نه تقصیر مشتری)

۲) ***در اثر نیازهای جدید مشتریان*** اصلاحات داده هایی که توسط سیستم های اطلاعاتی ایجاد می شوند، عملکردی که توسط محصولات عرضه می شوند،ویا خدماتی که توسط یک سیستم کامپیوتری ارائه می شوند(مقصر مشتری)

۳) ***تغییرات در اولویت پروژه ویا ساختار تیم مهندسی نرم افزار*** در اثر سازماندهی دوباره ویا گسترش/کاهش تجارت (مقصر ما)

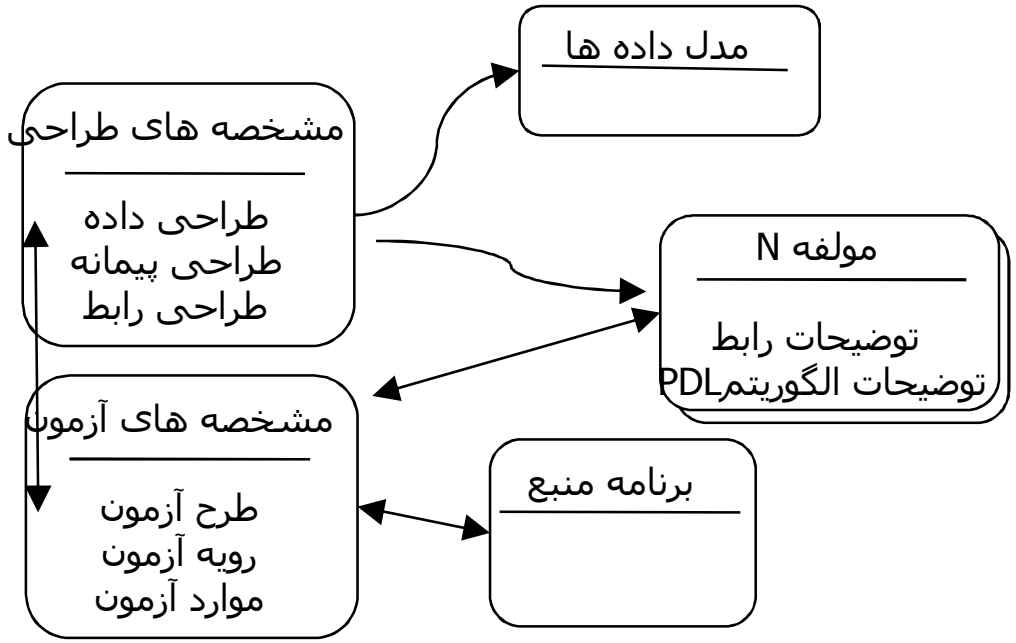
۴) ***مشکلات مربوط به بودجه ویا برنامه ریزی*** که سبب تعریف دوباره سیستم ویا محصول می گردد.(مقصر مشتری)

مدیریت پیکربندي (وضعیت؟؟؟) نرم افزار یعنی *مجموعه ای از فعالیت هایی که برای کنترل تغیردر طول چرخ زندگی نرم افزار کامپیوتری به وقوع می پیوندد.*

اقلام پیکربندی نرم افزار (SCI)

اطلاعاتی که به عنوان بخشی از فرایند مهندسی نرم افزار به وجود می آیند. یک SCI عبارت است از یک **سند**، مجموعهٔ کاملی از موارد **آزمون**، ویا **جزء** معروفی از یک **برنامه**.(مثلا يك تابع در c++)

در واقع SCI ها **برای تشکیل اشیاء پیکربندی** سازماندهی می شوند که می توانند در پایگاه داده های پروژه با یک نام جداگانه به فهرست در آیند، شی پیکربندی دارای **نام و صفت** می باشد و از طریق یک سری **روابط** با سایر شی ها ارتباط پیدا می کند.



- SCI حاصل از محصولات كاري – ابزارها(مثل IDE و محيط برنامه نويسي و كامپايلر) جزئي از پيكربندي نرم افزارند.

- فلش خميده رابطه تركيبي را نمايش مي دهد يعني مدل داده و مولفه N از مشخصه هاي طراحي اند

- *Software Configuration Items*
- *configuration objects*

خطوط مبنا

خط مبنا عبارت است از یک مفهوم مدیریت وضعیت(پیکربندي) نرم افزار که به ما کمک می کند تا بدون این که توجه جدی به تغییر داشته باشیم، **تغییر را کنترل کنیم**.

IEEE خط مبنا را به شرح زیر تعریف می کند:

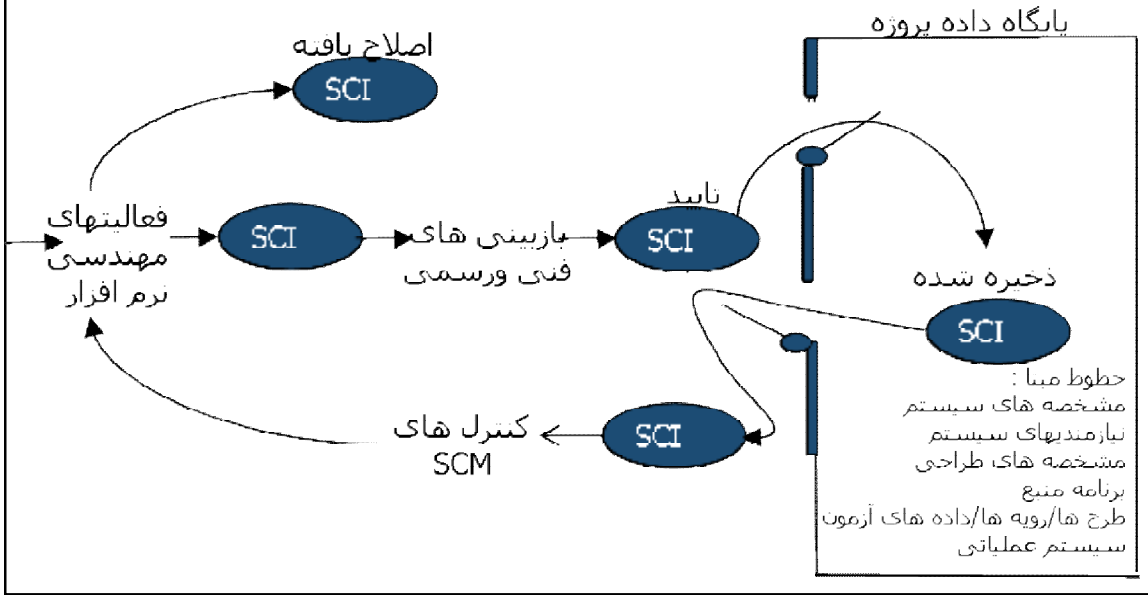
یک ویژگی ویا محصولی که به طور اساسی مورد **بررسی قرار گرفته وبه تأیید رسیده**، وپس از آن به عنوان پایه واساسی برای مراحل تکامل بعدی مورد استفاده قرار خواهد گرفت وفقط می تواند به واسطه شیوه های اساسی کنترل تغییر، تغییر یابد.

توصیف مبنا با مثال:

در يك رستوران از آشپزخانه به محيط رستوران ۲ در ، يكي براي ورود و يكي براي خروج وجود دارد اگر گارسون يك غذا را بردارد و قبل از اينكه از در خروج، خارج شود متوجه اشتباه بودن غذا شود بدون تشريفات و خيلي سريع و راحت آنرا تغيير مي دهد.

ولي اگر وي از در خارج شود، غذا را تحويل مشتري دهد و متوجه اشتباه خود شود بايست ۱-براي اطمينان از اشتباه به فيش نگاه كند ۲-از مشتري عذرخواهي كند ۳-از طريق در ورودي وارد آشپزخانه شود ۴- اشتباه را توضيح دهد و....

خط مبنا مثل درهاي آشپزخانه هستند ،قبل از اينكه يك SCI (آيتمهاي پيكربندي نرم افزار) به مبنا تبديل شود تغييرات خاصي وجود نداردو تغييرات غير رسمي هستند ولي با عبور از خط مبنا ،تغييرات اساسي ايجاد مي گردد



پیشرفت وقایع سبب ایجاد خط مبنا می شود، کارهای مربوط به مهندسی نرم افزار سبب ایجاد یک یا چند SCI می شوند. با بررسي و تایید SCI ها، دریک پایگاه داده پروژه قرار می گیرند.وقتی که یکی از اعضای تیم مهندسی نرم افزار می خواهد بر روی SCI دارای مبنا تغییرایجاد کند این SCI از روی پایگاه داده های پروژه بر روی فضای شخصی مهندسی کپی می شود. اما این SCI استخراج شده را تنها زمانی می توان تغییر داد که کنترل SCI ادامه پیدا کند.

فرایند مدیریت پیکربندی نرم افزار (SCM)

مدیریت پیکربندی نرم افزار رکن مهمی از تضمین کیفیت نرم افزار است. مسئولیت اولیه آن، **کنترل تغییر** می باشد. اما SCM همچنین مسئول **شناسایی SCIهای** جداگانه **ونسخه های گوناگون نرم افزار،رسیدگی به پیکربندی نرم افزار** برای حصول اطمینان از این که به طور مناسبی تکمیل شده و گزارش تمام تغییرات اعمال شده به پیکربندی.

SCM به ۵ سوال زیر پاسخ مي دهد:

۱-چگونگي سازگار شدن تغييرات در برنامه ها

۲-چگونگي كنترل نسخه نرم افزاري قبل از تحويل به مشتري

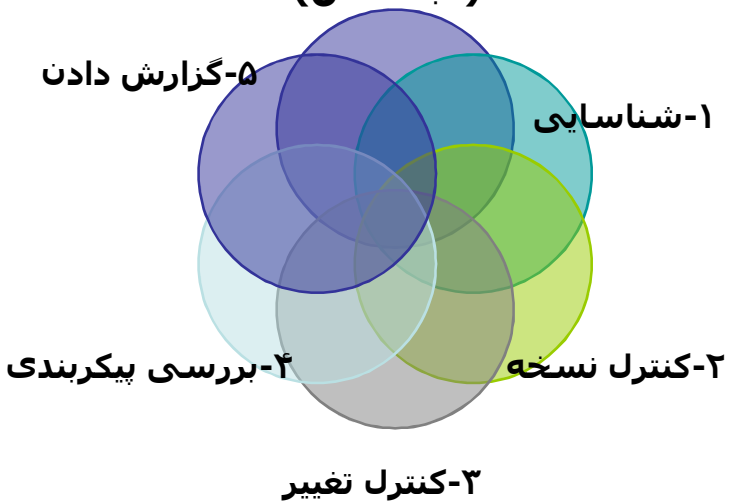
۳-چه كسي مسئول درجه بندي تغييرات است

۴-چگونگي اطمينان از انجام تغييرات

۵-مكانيزم ارزيابي تغييرات؟

پنج کار SCM :

(کبك گش)



پنج کار SCM

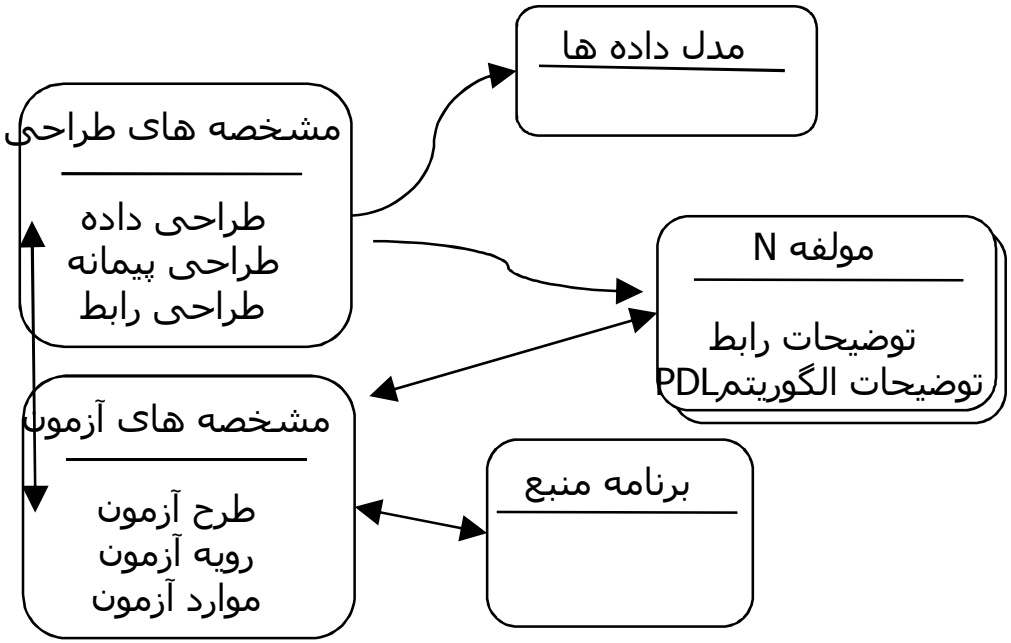
۱) شناسایی اقلام پیکربندی و ایجاد خط مبنا
۲) کنترل نسخه
۳) کنترل تغییر
۳-۱) کنترل دسترسی
۳-۲) کنترل همزمانی
٤) بررسی پیکربندی
٥) گزارش وضعیت

شناسایی اشیاء در پیکربندی نرم افزار

برای **کنترل** واداره قلم های پیکربندی نرم افزار، هر یک از آنها باید به طور جداگانه نام گذاری شده وسپس با استفاده از یک رهیافت "شی گرا" سازماندهی گردند. دو نوع شی قابل شناسایی هستند:

۱-شی پایه: عبارت است از **واحدمتن** که توسط یک مهندس نرم افزار وبه هنگام تحلیل، طراحی، کددهی ویا آزمون ایجاد می شود.

۲-شی مجتمع: مجموعه ای از شی های پایه و سایر شی های مجتمع.("ویژگی طراحی" یک شی مجتمع است)



هر شی دارای مجموعه ای از ویژگی های مشخص است که آن را به طور منحصر به فردی شناسایی می نماید:یک نام،یک توصیف، فهرستی از منابع،ویک تحقیق.

۱-نام شی عبارت است از یک رشته کاراکتر که شی را به طور مبهمی شناسایی می نماید.

۲-توصیف شی عبارت است از فهرستی از قلم های داده ها که موارد زیر را شناسایی می نماید:

- نوع SCI که بوسیله شی مشخص می شود
- شناسه پروژه
- اطلاعات مربوط به نسخه ویا تغییر

۳-منابع موجودیت هایی هستند که ارائه شده،پردازش شده، ارجاع داده شده ویا توسط شی درخواست می گردند.

۴-تحقیق نشانگری است در مورد "واحد متن"یک شی پایه وبی اعتبار در مورد یک شی مجتمع.

•**شناسايي پيكربندی شي**

شناسايي روابط بين اشياست، دو نو رابطه بين اشيا وجود دارد:

۱-ارتباط بين اشيا مجتمع

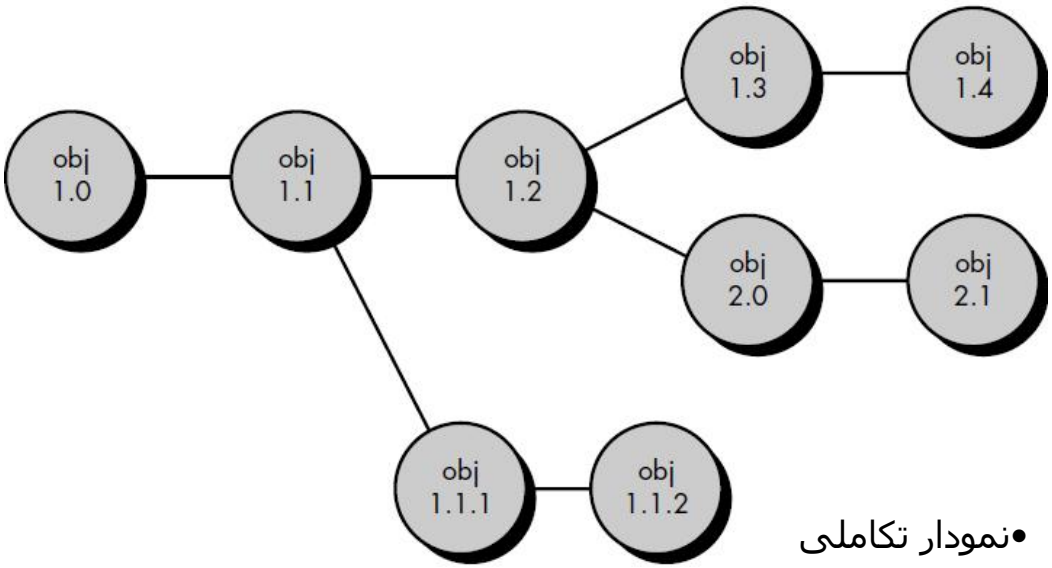
۲-ارتباط بين يك شي تجمعي (DataModel) و شي پايه (Class m)

•**روابط بين اشيا**

روابط بين شی های پیکربندی را می توان با استفاده از یک زبان برهم بندی پیمانه ای (MIL)نشان داد. یک MIL وابستگی بین شی های پیکربندی را توصیف می کند وکمک می کند تا هر نوع نسخه ای از یک سیستم به طور خودکار ساخته شود.

شی ها در طول فرایند نرم افزاری به وجود می آیند.پیش از آنکه یک شی به صورت خط مبنا درآید وحتی پس از این که یک خط مبنا ایجاد گردد ممکن است چندین بار تغییر کند.می توان برای هر شکل یک نمودار تکاملی ایجاد کرد.**نمودار تکاملی تاریخچه تغییر شی را توصیف می کند.**

•بازبيني ها باعث تبديل شي 1.0 به شي 1.1 مي شوند و انجام اصلاحات جزيي و توليد نسخه ۱.۱.۱ و ۱.۱.۲ و با اصلاحات عمده ، نسخه ۱.۲ ايجاد مي شود و ....



•نمودار تکاملی

module interconnection language

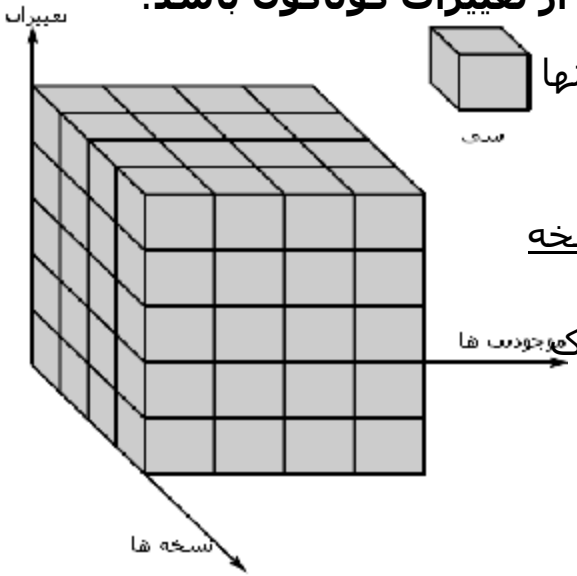
کنترل نسخه

کنترل نسخه ؛روش ها وابزارها را ترکیب می نماید تا نسخه های گوناگونی از شی های پیکربندی را که طی فرایند نرم افزار ایجاد شده اند مهار نمایند.

"آقاي كلم" کنترل نسخه در متن SCM را به شرح زیر توصیف می کند:

مدیریت پیکربندی به یک کاراین امکان را می دهد تا گزینه های پیکربندی گوناگونی از سیستم نرم افزاری را، از طریق انتخاب نسخه های مناسب مشخص نماید.این امر از طریق ربط دادن صفات با هر یک از نسخه های نرم افزاری وسپس مشخص نمودن یک پیکربندی از طریق توصیف مجموعه ای از صفات دلخواه، پشتیبانی می گردد.

- یکی از راه های *نمایش نسخه های گوناگون* یک سیستم *نمودار تکاملی* است.
- هر یک از گره های موجود در نمودار، یک **شی مجتمع**،یعنی **نسخه کاملی از نرم افزار**،می باشند.
- هر نسخه نرم افزار عبارت است از **مجموعه ای از SCI ها و هر نسخه می تواند ترکیبی از تغییرات گوناگون باشد**.



- روش دیگر مفهومی نمودن رابطه بین موجودیت ها،متغیرها ونسخه ها عبارت است از ارائه آنها به عنوان یک **مخزن اشیاء.**
- در **مخزن اشیا** رابطه بین موجودیت ها،متغیرها ونسخه ها مي تواند به صورت فضايي سه بعدي نمايش داده شود ویک موجودیت تشکیل شده است از مجموعه ای از شی ها که نسخه تجدیدنظر شده آنها در یک سطح قرار دارند.(عناصر(موجودیت ورودی))

یک متغیر(حالت تغییر) مجموعۀ متفاوتی از شی هاست که نسخۀ تجدیدنظر شدۀ آنها در یکسطح است و بنابراین در موازات سایر متغیرها قرار دارد.

چندین روش خودکارگوناگون برای کنترل نسخه پیشنهاد وارائه شده؛تفاوت اصلی این شیوه ها در پیچیدگی صفاتی است که برای ساخت نسخه ها و متغیرهای خاصی از یک سیستم وروش کارفرایند ساخت به کار می رود

شی های پایه و شی های مجتمع، تشکیل یک مخزن اشیا را می دهند که نسخه ها و گونه های متفاوت دیگر از روی آن ساخته می شوند.کنترل نسخه عبارت است از مجموعه ای از شیوه ها و ابزارها که برای کنترل استفاده از این شی ها به کار می روند

نام درس:مهندسی نرم افزار ۱

engineering change order
Change control authority

کنترل تغییرات

به عقیده جیمز باچ کنترل تغییر **خیلي زیاد** باعث ایجاد مشکلات مي شود و کنترل تغییر **خیلي کم** هم مشکلات دیگري را ایجاد مي کند.

یک درخواست تغیر، برای ارزیابی قابلیت فنی،تأثیرات جانبی بالقوه، تأثیرکلی بر روی سایر شی های پیکربندی وعملکرد سیستم، و هزینهٔ پیش بینی شدهٔ تغییر، ارائه وارزیابی می گردد. نتایج ارزیابی به صورت یک گزارش تغییر ارائه می گردد که توسط یک **محوز کنترل تغییر**(CCA) مورد استفاده قرار می گیرد.CCA فرد یا گروهي است که تصمیم نهایي را در مورد وضعیت و اولویت تغییرات مي گیرد برای هر تغییر به تأیید رسیده، یک ،**دستور(ترتیب؟؟؟) تغییر مهندسی** (ECO) ایجاد می شود. ECO تغییری را که قرار است به وجود آید توصیف می نماید؛محدودیت هایی که باید اعمال شوند، ومعیارهایی برای مطالعه وبررسی، شی که باید تغییر کند از پایگاه داده های پروژه بیرون کشیده شده ،تغییر اعمال شده و فعالیت های SQA مناسبی اعمال می گردند. سپس شی وارد پایگاه داد ها می شود.

- فرایند "بیرون کشیدن"و"وارد نمودن" (از بانك اطلاعاتي) دو رکن اصلی کنترل تغییر را ایجاد مي کند:

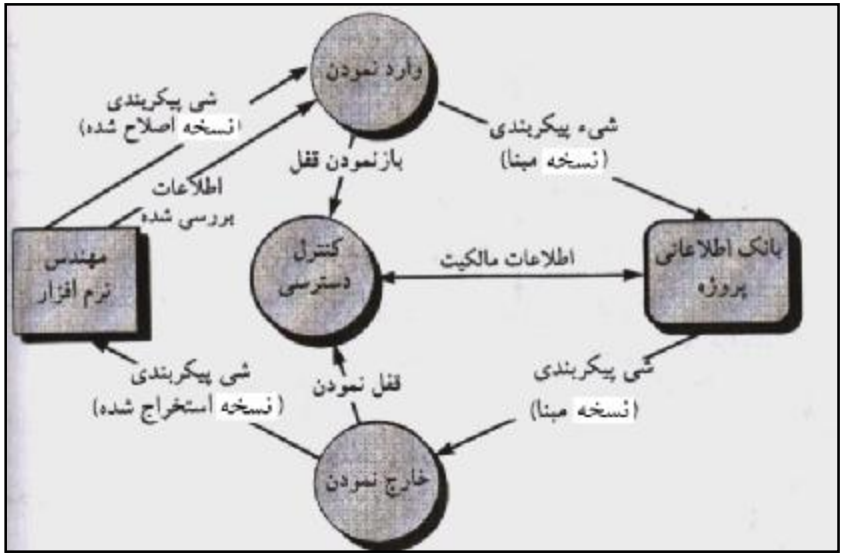
۱. کنترل دستیابی :

تعیین می کند کدام مهندس نرم افزار اختیار ارزیابی و اصلاح یک شی پیکربندی خاص را دارند.(زمانی که یک قلم پیکربندی به عنوان خط مبنا در مخزن ذخیره می شود و فردی درخواست تغییر می دهد مدیر باید بررسی کند که آیا او اجازه دسترسی به این قلم را دارد یا خیر)

۲. کنترل هم زمانی:

کمک می کند اطمینان حاصل شودکه تغییرات مشابهی که توسط دو فرد مختلف انجام می شوند؛بر روی هم کپی نشوند.(دو نفر به طور همزمان نمی توانند یک قلم پیکربندی را تغییر دهند. برای این منظور با هر دسترسی به یک قلم، آن قلم باید قفل شود.)

نام درس:مهندسی نرم افزار ۱



•کنترل دسترسي و همزماني

- یک مهندس نرم افزاربر اساس درخواست تغییر تأیید شده و ECO یک شی پیکربندی را بیرون می کشد.
- عمل ارزیابی کنترل به ما اطمینان می دهد که مهندس نرم افزار اختیار بیرون کشیدن شی را داردو هم زمانی کنترل شی را در پایگاه داده های پروژه نگه می دارد،به طوری که این شی به هیچ عنوان به روز نمی شود تا زمانی که نسخه ای که به تازگی بیرون کشیده شده جایگزین آن گردد.
- یک نسخه از شی دارای مبنا، که نسخه استخراج شده نامیده می شود، توسط مهندس نرم افزار اصلاح می گردد.
- پیش از آنکه یک SCIتبدیل به یک خط مبنا گردد، تنها نیاز به استفاده از کنترل تغییر غیر رسمی وجود دارد.
- برای ایجاد تغییر، چنانچه تغییر سایر SCIها را تحت تأثیر قرار دهد، سازنده باید از مدیر پروژه ویا از CCAکسب اجازه نماید.

•فرایند کنترل تغییر:

- ۱- نیاز به تغییر تشخیص داده می شود ۲-درخواست تغییر از کاربر می رسد ۳-سازنده ارزیابی می کند ۴- گزارش تغییر تولید می شود ۵-فرد مجاز به کنترل تغییر تصمیم می گیرد : A.درخواست تغییر پذیرفته نمی شود و در مرحله بعد کاربر مطلع می شود.
  Bدرخواست تغییر در نوبت اقدام قرار می گیردو ECO تولید می شود:
- افراد به اشیاء پیکربندی نسبت داده می شوند←اشیاء(اقلام) پیکربندی خارج می شوند←نیاز به تغییر تشخیص داده می شود←تغییر اعمال می شود←تغییر مورد وارسی قرار می گیرد←اقلام تغییر یافته پیکربندی ،وارد می شوند←خط مبنایی برای آزمون برقرار می شود←فعالیت های تضمین کیفیت وآزمون انجام می شود← تغییرات برای ارتقاء نسخه بعدی مستقر می شود← تغییر در نسخه جدید قرار می گیرد← نسخه جدید توزیع می شود

مسئول کنترل تغییر (CCA) نقش فعالی در لایه های دوم وسوم کنترل ، ایفا می کند.نقش CCA عبارت است از داشتن یک دیدگاه جهانی؛ سوالاتی مانند اینکه تغییر چگونه می تواند درک مشتریان را نسبت به محصول اصلاح نماید؟ توسط CCA مورد بررسی قرار می گیرند.

وارسی پیکربندی

- شناسایی، کنترل نسخه و کنترل تغییر به سازنده نرم افزار کمک می کند تا ترتیب را برقرار نماید.و از آشفتگي جلوگيري كند

مکانیزم های کنترل فقط تا زمانی تغییر را دنبال می نمایند که یک ECO ایجاد گردد.

برای کسب **اطمينان از پياده سازي مناسب تغيير** ؛از طریق ۱- بررسی های فنی رسمی ۲-وارسی پیکربندی نرم افزار عمل مي كنيم.

- **بررسي فني رسمي** :**به صحت فنی شی پیکربندی** که **اصلاح** گردیده می پردازد. SCI ها و نارسایی ها ویا اثرات جانبی بالقوه را ارزیابی می نماید. بررسی فنی رسمی باید در مورد تمام وحتی جزیی ترین تغییرات صورت پذیرد.
- **وارسی پیکربندی نرم افزار:**بررسی فنی رسمی را از طریق *ارزیابی ویژگی های* یک شی پیکربندی که *معمولاً طی بررسی در نظر گرفته نمی شود*.تکمیل مي نمايد
- در وارسي يك سري سوالات براي كسب اطمينان پرسيده مي شود
- **نتیجه:**وارسی پیکربندی عبارت است از یک فعالیت SQA که به حصول اطمینان از کیفیت به هنگام اعمال تغییرات کمک می نماید

گزارش وضعیت

- گزارش وضعیت پیکربندی که به آن **صورت وضعيت(حسابرسي وضعيت)** هم می گویند؛ یکی از وظایف SCM است.

جریان اطلاعات در مورد گزارش وضعیت پیکربندی را به اختصار CSR می نامند. هر بار که یک SCI هویت جدید و یا به روزی را انتخاب می نماید، یک مورد CSR ایجاد می شود. هر بار که وارسی پیکربندی صورت می پذیرد ، **نتایج(وارسی پیکربندی)** به عنوان بخشی از کار CSR گزارش میگردند. خروجی CSR در یک پایگاه داده ممکن است روی خط قرار بگیرد، که در این صورت سازندگان و یا اداره کنندگان نرم افزار می توانند اطلاعات مربوط به تغییر را به وسیلهٔ کلید واژه ارزیابی نمایند.

یک گزارش CSRبر طبق یک مبنای ثابت ارائه می شود و هدف آن این است که مدیر و سازندگان به ارزشیابی تغییرات ادامه دهند.

گزارش وضعیت پیکربندی نقش مهمی در موفقیت پروژه های بزرگ نرم افزاری ایفا می کند.

نتیجه :گزارش وضعیت ، اطلاعاتی را دربارهٔ هر تغییر به کسانی که نیازمند آگاهی از آن هستند ارائه می نماید

Configuration status reporting

استاندارد های مدیریت پیکربندی نرم افزار

- بسیاری استانداردهای مدیریت پیکربندی نرم افزار بیش از دو دهه است که ارائه می شود.

بسیاری از **استانداردهای اولیه بر توسعه وساخت نرم افزار های نظامی متمرکز شده وتاکید داشته** اند.

استانداردهای ANSI/IEEE برای نرم افزارهای **غیرنظامی** به کار می رود و برای سازمانهای مهندسی نرم افزار کوچک و بزرگ توصیه می شود.

- استانداردهای اولیه مانند:

MIL-STD-483 ، DOD_STD_480A، و MIL-STD-1521A

- استانداردهای ANSI-IEEE مانند:

ANSI/IEEE Stds. NO.828-1983 ،NO.1042-1987،

فصل دهم : مهندسی سیستم

- **مهندسی نرم افزار در نتیجه فرایندی بنام مهندسی سیستم ایجاد می گردد.**
- **مهندسی سیستم(دیدگاه پرسمن) :**

۱. **مهندسی فرایند تجاری**(Business Process Engineering) :

چنانچه محدوده کار مهندسی تمرکز بر روی کار و تجارت داشته باشد ، مهندسی سیستم را مهندسی فرایند کاری می نامند.

۲. **مهندسی محصول** (Product Engineering) **:**

زمانی که محصولی ایجاد می گردد، فرایند مهندسی سیستم را مهندسی محصول می نامیم .

❑مهندسی سیستم در مورد چه چیزی بحث می کند؟

پیش از آنکه نرم افزاری طرح ریزی و تهیه گردد باید سیستمی که نرم افزار در آن قرار داده می شود مورد مطالعه قرار گیرد. باید هدف کلی سیستم مشخص گردد و نقش سخت افزار، نرم افزار، پایگاه داده ها، پردازه ها و دیگر اجزاء سیستم شناخته شده و نیازهای عملکردی آن بررسی، تحلیل و مشخص گردد.

❑کار مهندس سیستم چیست؟

سعی مهندسان سیستم بر آن است تا از راه تماس با خریداران، کاربران آینده و دیگر افراد ذیربط سیستمی ایجاد شود که پاسخگوی نیاز آنها باشد.

❑اهمیت مهندسی سیستم چیست؟

يك ضرب المثل مي گويد «جنگل را نمي توان قبل از ديدن درختان ديد»

- جنگل=خود سیستم
- درختان=ارکان فنآوری

ولي چنانچه پيش از فهم سيستم به ساخت اجزاء فنی پرداخته شود بی شک دچار اشتباهاتی خواهيم شد.پس پيش از انديشيدن درباره درختان ، بايستی به درک جنگل پرداخت.

❑مراحل شناخت سیستم (درك جنگل) کدام اند؟

اهداف و جزئيات بيشتر درباره نيازمنديهای عملياتی را می توان با ۱-آگاهی از خواسته ها و نيازهای خريداران کسب نمود.سپس

۲-نيازمنديها تحليل شوند تا از ميزان وضوح، جامع بودن و قابليت انطباق و ثبات آنها تعيين گردد.

۳-سرانجام تمامی کارهای لازم جهت برآورده شدن نيازمنديهای مذکور صورت می گيرد تا اطمينان حاصل شود که تغييرات به بهترين نحو انجام شده است.

❑حاصل کار چیست؟

تهیه الگویی کارآ از سیستم بایستی منجر به پدید آمدن مهندسی سیستم گردد.کار نمونه سازی می تواند تعیین خصوصیات سیستم مورد نظر و یا حتی تهیه نمونه ای نمادین از آن باشد

❑چگونه از درستی انجام کارها مطمئن گردیم؟

کار تولید شده توسط سیستم باید از نظر میزان وضوح، جامع بودن، قابلیت انطباق و ثبات مورد بررسی مجدد قرار گیرد. چنانچه انجام تغییراتی دیگر ضروری باشد آن تغییرات بایستی با استفاده از SCM سه بعدی صورت گیرد.

سیستمهای مبتنی بر کامپیوتر

•**تعریف سیستم** در واژه نامه Websters:

۱.مجموعه و یا اجزاء آرایشی که چنان پیوندی با هم دارند که شکلی واحد و یا کل سازمانی را تشکیل دهند.

۲.چند حقیقت، اصول، قاعده و غیره که چنان طبقه بندی شده و نظم یافته اند تا شکلی سازمان یافته با پیوندهای منطقی موجود میان اجزاء را نشان دهند.

۳.روش یا نقشه ای از نظم و ترتیب و یا رده بندی موجود.

۴.راهي براي انجام چيزي ، روشي ، رويه و....

❑تعریف سیستم کامپیوتري

مجموعه يا آرايشي از اجزاء كه به گونه اي سازمان يافته اند تا برخي اهداف از پيش تعريف شده را از راه پردازش اطلاعات انجام دهد.اين هدف ممكن است شامل برخي امور تجاري يا ارائه محصولي براي فروش باشد

❑ارکان مختلف سیستم:

۱.نرم افزار: شامل برنامه های کامپیوتر، ساختار داده ها و ارائه مستندات مربوطه می باشد که در انجام روش، روند و یا کنترل منطقی اعمال مورد نظر موثر است.

۲.سخت افزار: ابزاری الکترونیکی است که قابلیت انجام محاسبات را ممکن ساخته است.(سویچ/سنسور/موتور و ...)

۳.افراد: عبارتند از کاربران و استفاده کنندگان سخت افزار و نرم افزار.

۴.پایگاه داده: مجموعه وسیع و سازمان یافته اطلاعات که از راه نرم افزار قابل دسترسی است.

۵.مستندات: اطلاعاتی توصیفی است که راه استفاده و یا نحوه کارکرد سیستم را می نمایاند.

۶.رویه ها: مراحلی است که کاربرد ویژه هر یک از ارکان سیستم و یا متن رویه ای موجود در سیستم مورد نظر را توضیح می دهد.

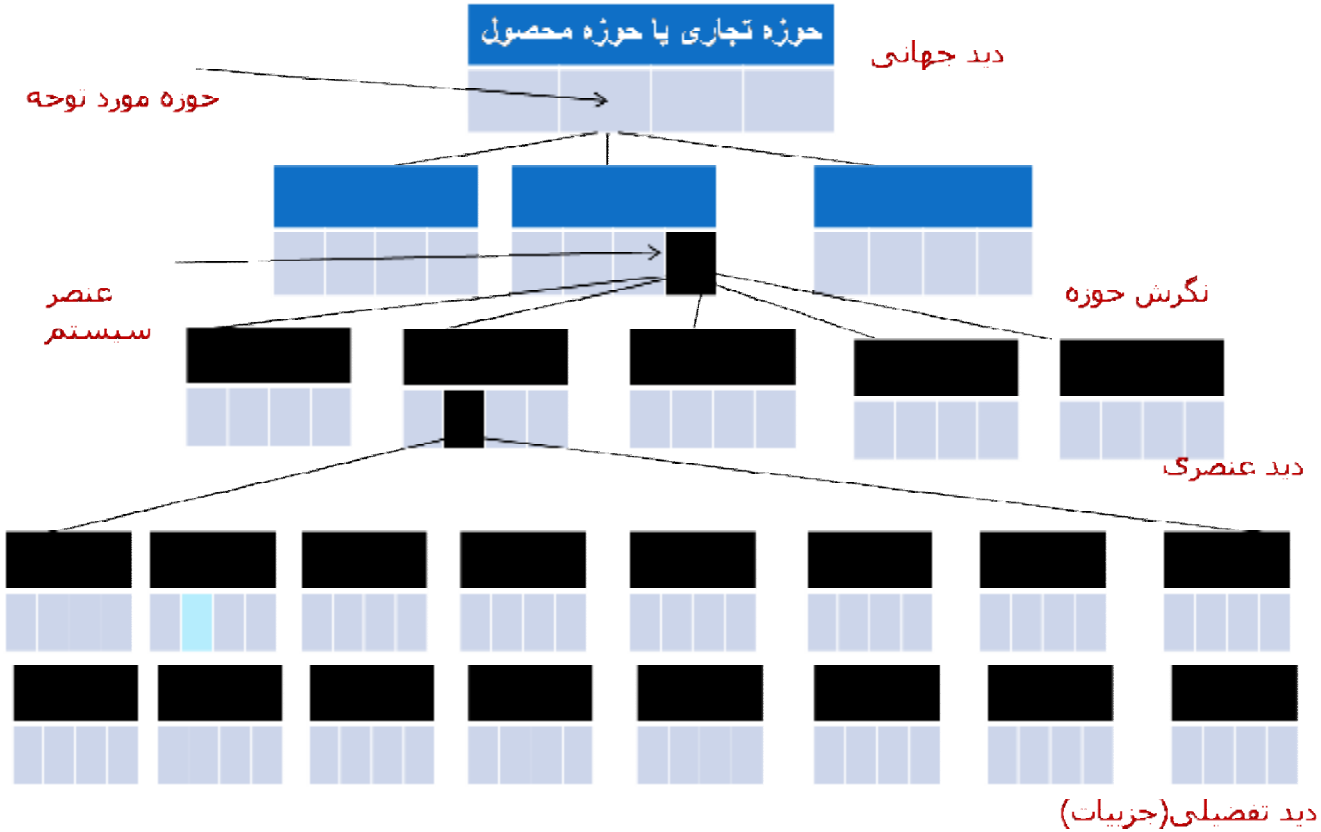
نام درس:مهندسی نرم افزار ۱

سلسله مراتب مهندسی سیستم

•سلسله مراتب در مهندسي سيستم از بالا به پايين و نيز از پايين به بالاست.

•روند مهندسی سیستمها با چشم انداز جهانی آغاز می گردد. سپس دایره چشم انداز جهانی را آنقدر تنگ می کنند تا هر چه بیشتر بر یک جنبه ویژه از خصوصیت مورد نظر، کانون توجه قرار گیرد. سرانجام مرحله های تحلیل، طراحی و ساخت سیستم مورد نظر انجام می شود.

•در بالاترین رده سلسله مراتب، موضوعهای بسیار کلی و در پایین ترین رده، جزییات کارهای تکنیکی قرار دارد.



WV={D1,D2,...,Dn}

world view (WV) =Set of domains (D_n)

Di={E1,E2,...,Em}

Domain (Di) =Set of elements (E_m)

Ej={C1,C2,...,Ck}

elements (Ej) =Set of technical components(C_k)

مدلسازی سیستم

مهندسی سیستم عبارت است از فرآیند مدل سازی.

خصوصیات مدلهای تهیه شده بدون تمرکز بر روي ديدگاه كلي يا جزئي:

- فرآیند مناسب جهت برآورده ساختن نیازهای دیدگاه مورد نظر تعیین گردد.
- چگونگي رفتاري فرآيندها و فرض هايي را كه رفتارها بر آن اساس پايه گذاري شده اند بيان كند
- به روشنی ورودی درون رو و برون رو مدل (ارتباط با هم سطحی ها یا با سطوح دیگر) مربوطه تعریف شود.
- تمامی پیوندها (ازجمله خروجي) که درک بهتر دیدگاه مورد نظر را برای شخص مهندس میسر می سازد تشریح و نمایانده شود.

- **عوامل مهار کننده در ساختن مدل سیستم**

در ساختن مدل سیستم مهندس باید چندین عامل مهار کننده (بازدارنده) را در نظر گیرد:

۱. **فرضیات(مفروضات)**: شمار تبدیلات و متغیرهای ممکن را کاهش می دهد و در نتیجه مدل تهیه شده می تواند مشکلات موجود را به شیوه ای منطقی بازتاب دهد.(مثلا:عدم پیچیده شدن پا دور بدن انسان باعث کاهش حالات در بازيهاي رايانه اي مي شود)

۲. **ساده سازی**: امکان می دهد مدل طبق برنامه ریزی زمانی و به موقع آماده گردد.(مثلا : به حداقل رساندن تعداد منابع ورودي)

۳. **محدود سازی**: محدوده سیستم مورد نظر را تعیین می کند.(مثلا: با تعیین **هدف** ،حوزه هاي ديگر نيز محدود مي شوند...)

۴. **تنگناها**: سبب می گردد تا هنگام ساخت مدلها رهیافت و تصمیمات مناسبی اتخاذ گردد.(مرز امكانات)

۵. **اولویتها**: ساختار مطلوب فنآوری و عملیات داده ها را تعیین می نماید.

مدل سيستم بدست آمده با هر ديدگاه مي تواند به يك راه حل خودكار كامل،راه حل نيمه خودكار و يا روش غيرخودكار منتهي شود.

در نتيجه مهندسي سيستم ، فقط ارتباط نسبي بين عناصر مختلف سيستم را اصلاح مي نمايد تا مدلهايي از هر نوع را بدست آورد

نام درس:مهندسی نرم افزار ۱

شبیه سازی سیستم

•سالها پیش روش ساخت سیستمها مانند هواپیما ساختن برادران رایت بود. وسیله ای را بطور کامل می ساختیم و بعد آن را بکار می بردیم، خراب می شد و دوباره از ابتدا ساخته می شد.

•امروزه ابزارهای نرم افزاری جهت شبیه سازی و مدل سازی سیستم در راه رفع موارد غافلگیرکننده در **ساخت سیستمهای واکنشی مبتنی بر کامپیوتر**به خدمت گرفته شده اند.

•مهندس سیستم توسط ابزارهای مدل سازی و شبیه سازی می تواند خصوصیات یک سیستم را مورد آزمون قرار دهد.

مهندسی فرآیند تجاری (BPE) business process engineering

•هدف مهندسی فرآیند تجاری توضیح معماریهایی است که یک حرفه را قادر می سازد تا از اطلاعات استفاده بهینه کند.

•هنگامی که نیازهای فنآوری اطلاعات شرکتی را از دیدگاه جهانی مورد بررسی قرار می دهیم دیگر در وجود نیاز به مهندسی سیستم هیچ شکی باقی نمی ماند. در این راه هم آگاهی از خصوصیات و جزئیات ساختار مناسب محاسباتی مورد نیاز است و هم آنکه ساختار نرم افزار که اشکال گوناگون و منحصر به فرد منابع نامتقارن محاسباتی را در خود جای می دهد، باید ارتقاء یابد.

•*مهندسی فرآیند تجاری روشی برای ایجاد طرحی فراگیر جهت پیاده سازی معماری محاسباتی می باشد.*

❑**جهت حصول دیدگاه کلی از نیازهای تکنولوژی یک شرکت، از مهندسی سیستم استفاده می نماییم. برای این منظور سه معماری مختلف می بایست در حیطه اهداف تجاري تحلیل و طراحی گردد:**

1. **معماری داده ها**
2. **معماری کاربردها**
3. **فناوری فراساختاری**

❑ **معماري داده ها**

چهارچوبی را جهت اطلاعات مورد نیاز یک رشته تجاری و یا جهت انجام عملیات تجاری فراهم می سازد. بلوکهای ساختاری جدا جدای این معماری، داده هایی هستند که در زمینه تجاری کاربرد دارند. **یک شیء داده ای دربردارنده تعریف یک رشته صفات خاصه آن، از برخی جنبه ها، کیفیت، خصوصیت، یا دیگر ویژگی هایی است که وصفی از داده مورد بحث ارائه می دهد.**

هر بار که مجموعه اشیاء داده تعریف می شود، **رابطه ها پیوندهای میان آنان نیز بررسی و بیان می گردد.** «پیوند موجود» چگونگی ارتباط یک شیء را با دیگراشیاء شرح می دهد. یعنی یا خریدار کالای A را می خرد و یا کالای A توسط خریدار، خریداری می شود.

نام درس:مهندسی نرم افزار ۱

❑ **معماری کاربرد**

شامل **عناصر یک سیستم** است که **اشیاء** را به منظور رسیدن به برخی **اهداف** تجاری به روند معماری **داده** ها انتقال می دهد. معماری کاربرد نقش افراد و روندهای تجاری را که هنوز خودکار نشده اند شامل می شود.

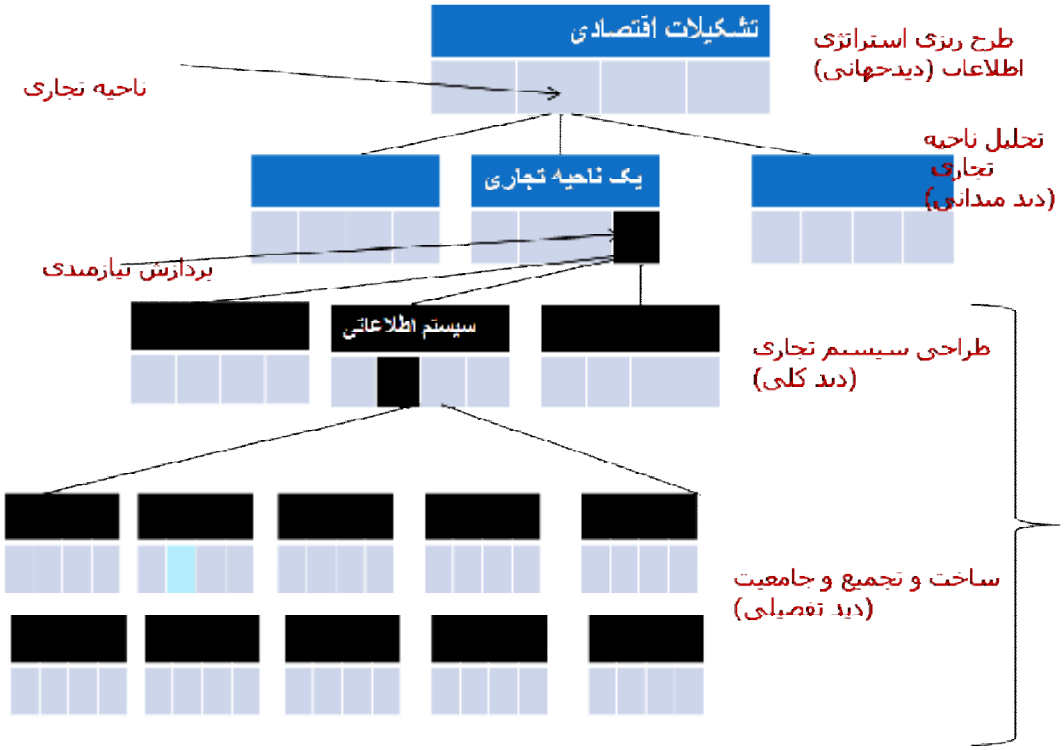
❑ **فناوری فراساختاری**

«زیرساخت فنآوری» پایه ریزی داده ها و معماری کاربرد را میسر می سازد. **زیرساختها دربرگیرنده سخت افزار و نرم افزارهایی است که جهت پشتیبانی کاربردها و داده ها به کار گرفته می شود**. و شامل کامپیوترها ؛ سیستم عاملها ؛ اتصالات و .... است

❑**مدلسازي معماري هاي سيستم**

براي مدلسازي معماري هاي سيستم ، سلسله مراتبي از فعاليتهاي مهندسي فرآيند تجاري تعريف مي شود.با توجه به شكل مقابل ميتوان دريافت كه ديدگاه جهانی برگرفته از «برنامه ریزی راهبرد اطلاعات» **ISP**است. این برنامه ریزی تمامی امور تجاری را به عنوان ورودی در نظر گرفته و حوزه های مختلف آن را از یکدیگر مجزا می کند.حوزه این دیدگاه موسوم به فعالیت تحلیل ناحیه ای تجاری»»BAA است.

- **BAA** جزئیات انواع داده ها و نیز اثرهای متقابل را تعیین می نماید. BAA تنها به امر ارائه تعریف هایی که در زمینه تجاری مورد نیاز است سروکار دارد. در این س سیستم



information strategy planning (isp)

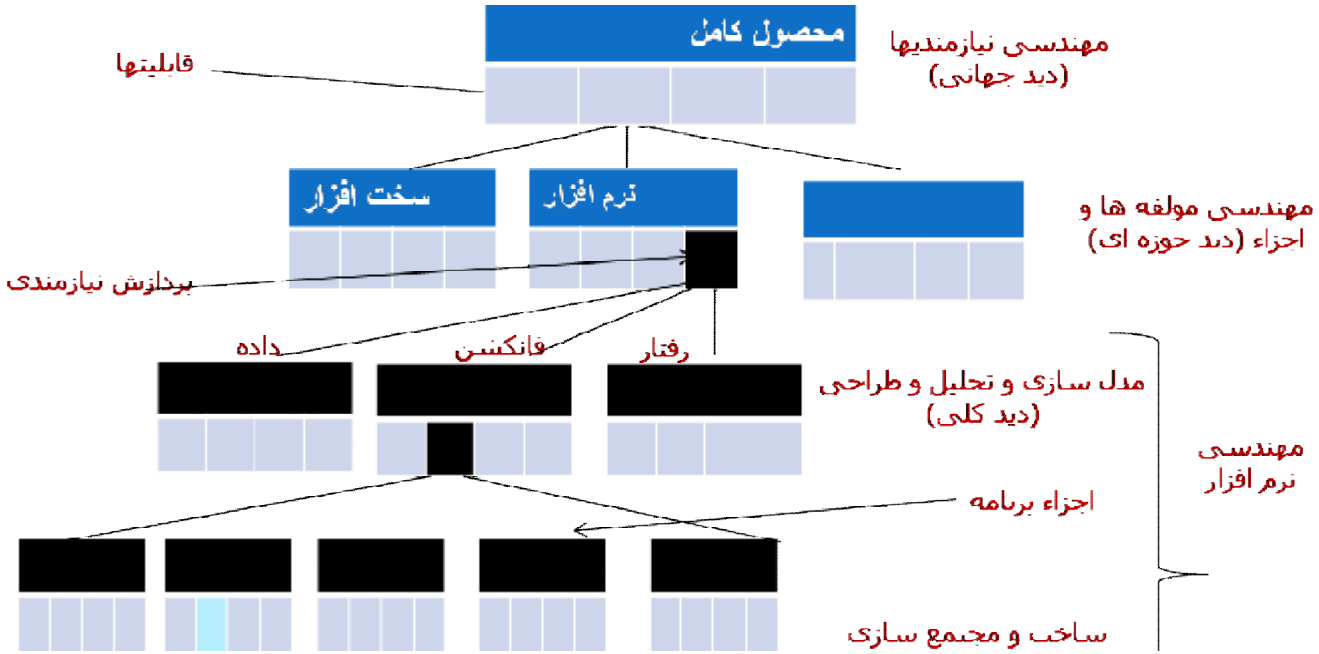
نام درس:مهندسی نرم افزار ۱

•وقتی یک سیستم اطلاعات جهت پیشرفت بیشتر ایزوله می شود، مسیر BPE وارد **مرحله مهندسی نرم افزار می گردد.** با فراخواندن مرحله « طراحی سیستم تجاری» (**BSD**)، **نیازهای اساسی سیستم اطلاعات مورد نظر مشخص و طراحی می** شوند، سپس نیازمندیهای مذکور به موضوعات معماری داده ها، معماری کاربرد و زیرساختهای فنآوری تبدیل می شوند.
•در مرحله نهایی BPE یعنی مرحله «**ساختمان و مجتمع سازی**» **(C&I)** به تکمیل و انجام جزئیات پرداخته می شود.

مهندسی محصول

هدف مهندسی محصول این است که خواستهای خریداران در مورد یک رشته از قابلیتهای تعریف شد را به محصول بالفعل تبدیل نماید.معماری چهار جزء مشخص و مجزای سیستم را در بر می گیرد که عبارتند از:

1-نرم افزار ۲-سخت افزار
۳-داده ها ۴-افراد



مهندسی نیازمندیها

بهترین راه حل براي كسب اطمينان از قدرت سيستم در برآورده شدن نيازهاي مشتري ، مهندسي نيازمنديها است

مهندسی نیازمندیها مکانیزم مناسبی را جهت درک خواسته های خریداران فراهم می سازد و نیز امکان تحلیل آن نیازها، برآورد امکانات، دستیابی به راه کارهای منطقی، بیان راه حل مذکور به طور روشن و واضح، ارزیابی خصوصیات و مدیریت نیازمندیها هنگامی که به سیستم عملیاتی تبدیل می شوند را فراهم می آورد.

فرآیند مهندسی نیازمندیها شامل شش قسمت زیر است:

۱-مرحله **یافتن** نیازمندیها

۲-مرحله **تحلیل** نیازمندیها و انجام بحث و گفتگو درباره آنها

۳-مرحله **تعیین مشخصات** نیازمندیها

۴-مرحل **مدل سازی** سیستم

۵-مرحله **ارزیابی نیازمندیها**

۶-مرحله **مدیریت نیازمندیها**

تعیین نیازمندیها

ظاهرا تعیین نیازمندیهای خریداران ساده به نظر می رسد اما حقیقت آن است که پاسخ به سوالات مورد نظر دشوار است. چند مشکل موجود عبارتند از:

مشکلات موجود در دامنه دید مسائل(محدوده): تعریف حدود سیستم ناقص صورت می گیرد و یا اینکه خریدار به ذکر جزئیات تکنیکی غیرضروری می پردازد.كه مي تواند باعث سردرگمي كل اهداف سيستم شود.

مشکلات موجود در زمینه درک مسائل(مفهومي): كاربرها يا مشتري ها از موارد مورد نياز خود آگاهی چندانی ندارند.مطمئن نيستند.اطلاعاتي را حذف مي كنند كه به عقيده آنها واضح است و نيازهايي را مشخص مي كنند كه مبهم يا غيرقابل آزمايش هستند

دشواری ارزیابی: نیازها در طول زمان تغییر می کنند.

مهندسان سیستم جهت فایق آمدن بر مشکلات فوق باید فعالیتهای مربوط به جمع آوری اطلاعات در زمینه نیازهای خریداران را به شیوه ای سازمان یافته انجام دهند

نام درس:مهندسی نرم افزار ۱

❑رهنمودهای سامرویل و ساویر برای یافتن نیازمندیها

- ❖ تخمین کار و امکان سنجی تکنیکی سیستم مورد نظر
- ❖ شناخت افرادی که به مشخص کردن نیازمندیها و نیز به درک خواسته های خریداران و تمایل به سازمان یافتگی،کمک مي كنند.
- ❖ تعریف کردن محیط تکنیکی برای عرصه ای که سیستم یا محصول در آن جای می گیرد.
- ❖ تعریف کردن «حوزه محدودیت ها» که دایره عملیات و یا کارکرد سیستم یا محصولی که قصد ساختن آن را داریم، محدود می کند.
- ❖ تعریف و تعیین یک یا چند روش جهت یافتن نیازمندیها
- ❖ از افراد زیادی جهت شرکت در نشست ها دعوت به عمل آید تا نیازمندیها از دیدگاههای مختلف تعریف و بررسی گردد.
- ❖ نیازمندیهای دارای ابهام برای ساخت نمونه اولیه در نظر گرفته می شود.
- ❖ سناریو کاربرد پدید آورده می شود تا خریداران/کاربران بتوانند نیازمندیهای کلیدی خود را بشناسند.

❑محصول کاری(نقاط عطف) در سیستم بدست آمده در نتیجه فعالیت دریافت نیازها:

- ❖نیازمندیها و قابلیت برآورده ساختن آنها تصریح شود
- ❖حیطه سیستم و یا محصول تعیین گردد
- ❖فهرستي از مشتري ها،کاربران و... که در دریافت تیازها موثرند ایجاد شود
- ❖محیط تکنیني سیستم تعریف و توصیف شود
- ❖فهرستی از نیازمندیها ارائه شده و حوزه محدودیت هایی که در مورد هر یک از آنها اعمال می شود بیان گردید.
- ❖تهیه یک رشته سناریوی استفاده که امکان مشاهده سیستم مورد استفاده یا محصول را زیر شرایط مختلف عملکرد فرآهم می آورد
- ❖هر یک از طرح های اولیه را بایستی ارتقاء داد تا نیازمندیهای مطرح شده تعریف بهتر و روشن تری داشته باشند.

تحلیل نیازمندیها و مذاکرات مربوطه

پس از جمع آوری اطلاعات، در مرحله «تحلیل نیازها» نیازمندیها **دسته بندی** شده و به صورت **مجموعه ها و زیرمجموعه های** مربوطه در می آیند، **رابطه نیازها** بایکدیگر بررسي مي شود، آنها از نظر ثبات و سازگاری، قابلیت حذف شدن، و عدم وجود ابهام سنجیده و مورد آزمون قرار می گیرد و سرانجام اولویت نیازمندیها بر اساس خواسته ها و انتظارات خریداران/کاربران تعیین می گردد.

•پرسشهایی که در حین فعالیت تحلیل نیازمندیها باید پاسخ داده شود:

- آیا هر نیازمندی با هدف کلی سیستم و یا محصول مورد نظر سازگاری دارد؟
- آیا تمامی نیازمندیها بطور مختصر و مفید مطلب مورد نظرتان را بیان می دارند؟
- آیا نیازمندیهای مطرح شده براستی ضرورت رفع آن را ایجاب می کند و یا با تحقق آن صرفا خصوصیتی به سیستم اضافه می گردد که جزء اهداف ساخت آن سیستم نیست؟
- آیا محدوده هر نیازمندی مشخص و بدون ابهام است؟
- آیا هر نیازمندی موارد استناد نیز دارد، بدین معنی که آیا منبعی برای هر نیاز در نظر گرفته شده است؟
- آیا نیازمندی ای هست که با دیگر نیازمندیها مغایرت داشته باشد؟
- آیا می توان به هر یک از نیازمندیها در محیط تکنیکی که سیستم تولید شده در آن جای می گیرد دست یافت؟
- چنانچه نیازمندی ای برآورده شد آیا حاصل کار را می توان آزمود؟

•مهندس سیستم بایستی از راه فرایند بحث و گفتگو درباره پیشنهادهای مغایر با یکدیگر(مشتري ها) ، راه صحیح و میانه را بیابد.ریسکها همراه با هر نیاز تحلیل مي شوند. میزان تاثیر هر نیاز بر هزینه و زمان محاسبه مي شود و نیازها به گونه اي درست حذف ، ترکیب یا اصلاح مي شوند.

تعیین مشخصات نیازمندیها

يك مشخصه مي تواند يك سند كتبي،الگوي گرافيكي،مدل فرمولي رياضي،مجموعه اي از سناريوهاي مورد استفاده،اشكال اوليه و يا تركيبي از اين اجزا باشد

«تعیین مشخصات سیستم» مرحله نهایی کار ساخت و نیز کار انجام شده توسط سیستم و مهندس نیازها می باشد. تعیین مشخصات شالوده و زيربناي مهندسی سخت افزار، نرم افزار، مهندسی پایگاه داده و مهندسی انسانی را تشکیل می دهد.

عمل تعین مشخصه ها، حوزه عمل هر یک از ارکان سیستم کامپیوتر را به حدود معینی محدود می نماید.

«تعیین مشخصات سیستم» همچنین اطلاعات را که عبارت از ورودی و خروجی های سیستم است تعریف می کند.

مدل سازی سیستم

به منظور تعیین خصوصیات و مشخصات آشپزخانه ای که قرار است ساخته شود، لازم است مدلی معنادار داشته باشیم، یعنی برنامه کار و یا نمایش سه بعدی موجود باشد که موقعیت کابینت ها و لوازم آشپزخانه و هم ارتباط آنها با یکدیگر را نشان دهد.

مدل سازی سیستم نیز به دلیل استدلالهای انجام شده در مورد تهیه برنامه کار یا نمایش سه بعدی آشپزخانه تهیه می شود.

اعتبارسنجی (ارزیابی)نیازمندیها

نتایج مهندسي نيازها در مرحله ارزيابی مورد **سنجش** قرار می گیرد. به هنگام «اعتبارسنجی» مشخصات سیستم را بررسی می کنند تا مطمئن شوند **تمامی نیازمندیها** بطور روشن بیان شده، **ناهماهنگیها، خطاها و ... کشف و برطرف گردیده** و محصول با استانداردهای موجود در روند کار پروژه مطابقت دارد.

اولين گام در **ارزيابي نيازمنديها روش بررسي فني رسمي است** كه با تشكيل گروهي به بررسي مشخصات سيستم ، جهت يافتن خطاهاي مربوط به محتواي مشخصات ،اطلاعات حذف شده و عدم تطابق ها (تناقض در نيازها و دست يافتني نبودن آنها)

بهترين حالت براي بررسي نيازمنديها استفاده از يك چك ليست(سوال /جواب) است

پرسشهایی در مورد اعتبارسنجی محصول:

- آیا نیازمندیها به وضوح بیان شده است؟ آیا امکان سوء تعبیر آنها وجود دارد؟
- آیا منشا بروز نیازمندی شناخته شده است؟
- آیا آنچه اظهار نهایی نیازمندی مورد بحث استنباط می گردد با خود نیاز مطابقت دارد؟
- آیا در بیان نیازها مقادیر کمی نیز بکار رفته است؟
- چه نیازمندیهای دیگری به نیازمندیهای مطرح شده وابسته است؟
- آیا نیازهای مطرح شده قابل آزمون هستند؟
- آیا پاسخ به نیازمندیها مذکور را می توان در یکی از الگوهای ساخته شده سیستم جست و جو کرد؟
- آیا وجود آن نیازمندیها را می توان میان اهداف کلی سیستم/ محصول جست و جو کرد؟
- آیا نیازمندیها در پیوند با اجزاء سیستم، رفتار و ویژگی عملکردی آن به روشنی شرح داده شده اند؟

مدیریت نیازمندیها

«مدیریت نیازمندیها» عبارت است **رشته فعالیتهایی که به گروه امکان می دهد تا ضمن پیشبرد پروژه، در هر مقطع زمانی بتواند نیازها و تغییر نیازها را شناسایی، کنترل و ردیابی نماید.**

مدیریت نیازمندیها مانند SCM با عملیات شناسایی آغاز می گردد.و به هر نیاز يك **شناسه منحصر به فرد** داده مي شود كه به اين صورت است

<شماره نیازمندی><نوع نیازمندی>

نیاز عملکردی(تابعي): F نیاز داده ها: D نیازمندی رفتاری: B نیازمندی رابط و واسط: I نیازمندی خروجی:P

يعني نياز F9 ، نهمين نياز و از نوع تابعي است

جدولهای ردیابی

•این جدولها ، هر يك ،نیازهای مشخص شده را با يك يا چند جنبه از سیستم یا محیط مرتبط مي كنند

•نیازمندی	01	02	03	04	05	•جنبه ای مشخص از سیستم و محیط			
R_{01}									
R_{02}									
R_{03}									
R_{04}									
R_{05}									
...									
R_{nn}									

انواع جدولهای ردیابی

•«جدول ردیابی **ویژگی ها**» نشان می دهد که نیازمندیها چگونه با موارد استفاده ویژگیهای قابل توجه سیستم/محصول مرتبطند

•«جدول ردیابی **منابع**» که منبع هر نیازمندی را شناسایی می کند.

•«جدول ردیابی **وابستگی**» نشان می دهد که چگونه نیازمندیها به یکدیگر وابستگی دارند.

•«جدول ردیابی **زیرمجموعه های سیستم**» نیازمندیهای زیرسیستمها را طبقه بندی می کند.

•«جدول ردیابی **رابط**» نشان می دهد نیازهای چگونه با رابطهای داخلی و خارجی سیستم مرتبط است.

مدل سازی سیستم

•**هر سیستم مبتنی بر کامپیوتر** را می توان بعنوان **سیستم انتقال دهنده اطلاعات** که قالب «**اطلاعات ورودی-انجام پردازش- اطلاعات خروجی**» را بکار می گیرد مدل سازی کرد.

•برای تهیه و تکمیل الگوی سیستم «قالب مدل سیستم» (الگويي از مدل سيستم) مورد استفاده قرار می گیرد. مهندس سیستم ارکان آن را به یکی از پنج ناحیه پردازشی زیراختصاص می دهد:

۱-رابط کاربر **۲-اطلاعات ورودی** **۳-عملکرد و کنترل سیستم**

۴-اطلاعات خروجی **۵-پشتیبانی و انجام خودآزمون**

- الگوي و قالب مدل سیستم در شکل زیر ترسیم شده است

<table>
<tr><td colspan="3">پردازش رابط کاربر</td></tr>
<tr><td>پردازش خروجی</td><td>کارکردهای کنترل و فرآیند
نگهداری و آزمون خودکار</td><td>پردازش ورودی</td></tr>
</table>

•**نمودار بافت(زمینه) سیستم (SCD)** *System context diagram*

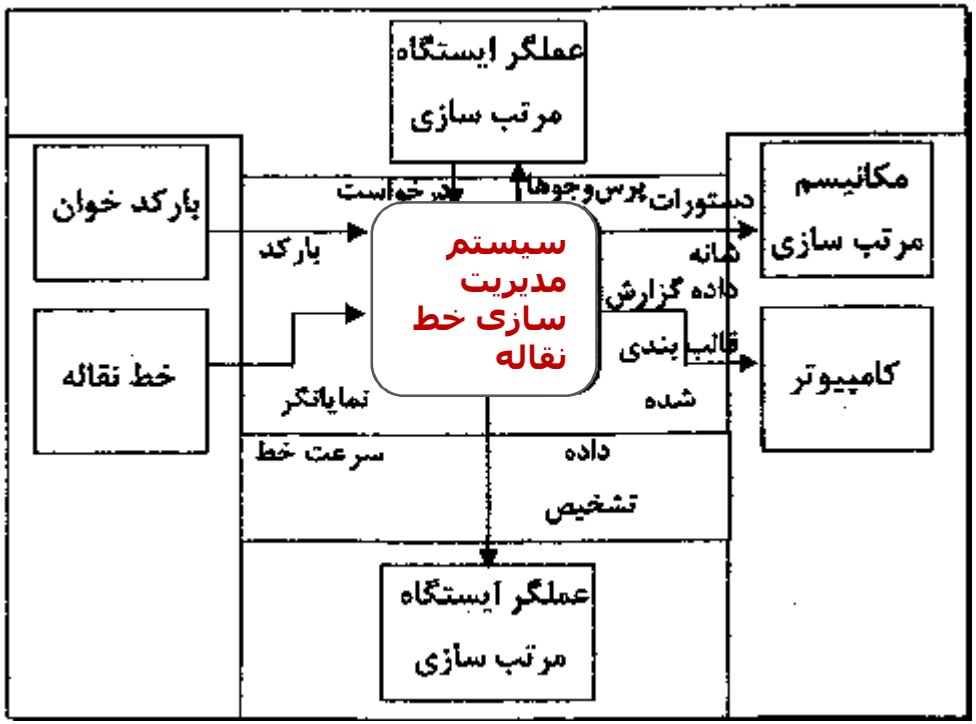
•در راس این سلسله مراتب قرار دارد . این نمودار ، محدوده و **مرز** اطلاعات **بین سیستم** در حال ساخت و **محیطي** که سیستم مذکور در آن عمل مي كند را مشخص مي كند. يعني SCD تعريف كننده تمامي رويه هاي خارجي اطلاعات مورد استفاده سيستم و ارتباط همه مراجعي كه از راه رابط يا انجام پشتيباني يا خود آزمايي را برقرار مي سازد

•**مثال نمودار بافت (SCD) در سیستم مرتب سازي (Sorting) (CLSS)**

•**اهداف CLSS**

CLSS باید چنان ارتقاء یابد تا جعبه هایی را که درطول خط انتقال در حرکتند شناسایی کرده و در انتهای خط آنها را بصورت شش «بین» (Bin)مرتب و دسته بندی نماید. جعبه ها توسط ایستگاهی شناسایی، عبور داده می شوند و شماره ای بر روی یک طرف جعبه نقش می گردد. سپس بسوی بین های مربوطه هدایت می شوند. جعبه ها بصورت تصادفی حرکت کرد و بطور برابر جای داده می شوند.

در این مورد CLSS گسترش یافته و از کامپیوتر شخصی در سایت ایستگاه مرتب سازی بهره می گیرد. کامپیوتر شخصی مذکور تمامی اعمال CLSS نرم افزار مربوطه را انجام می دهد.



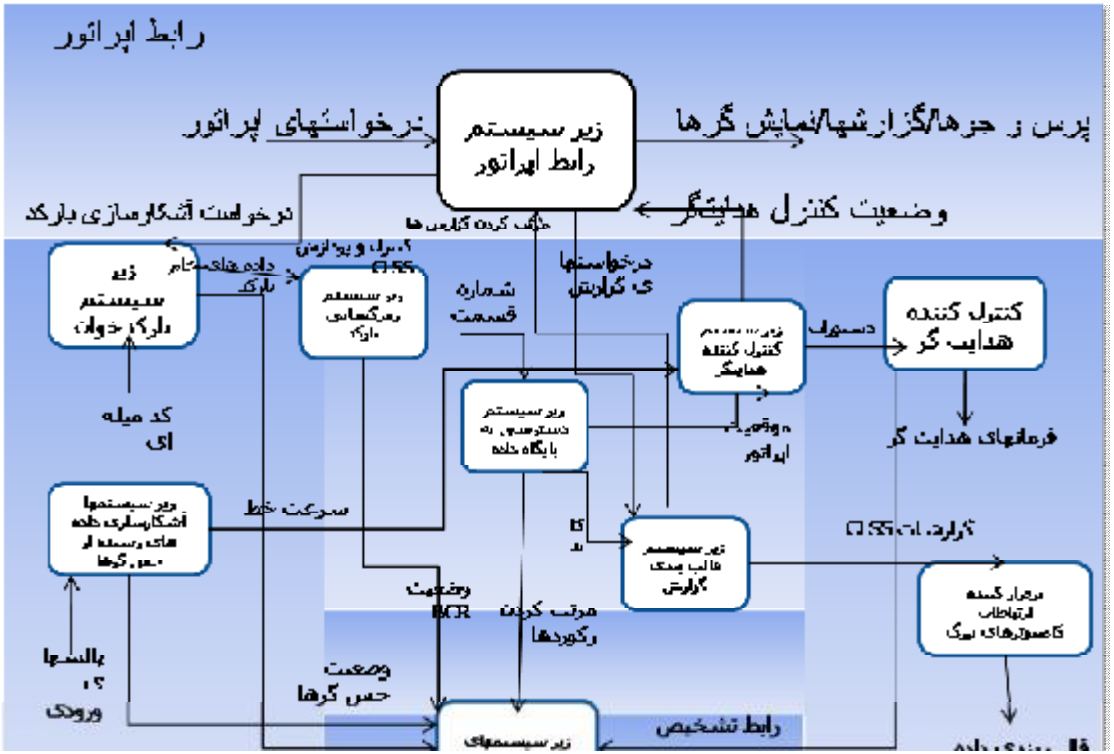
•چهارگوش: نمایانگر یک موجودیت یعنی تولیدکننده یا مصرف کننده

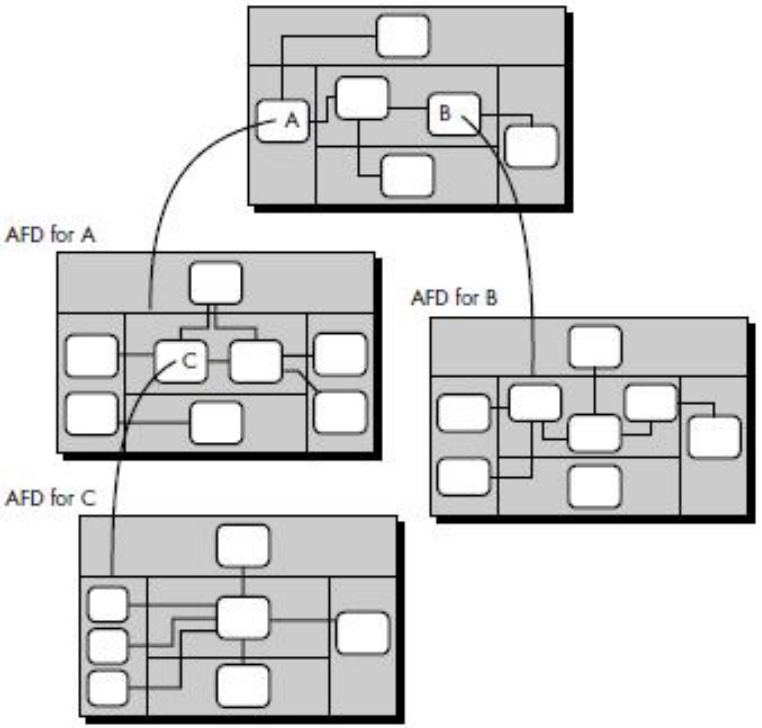
•مستطیل گوشه گرد شده: نماد کل سیستم. از این رو CLSS در ناحیه فرآیند و کنترل در SCD نشان داده شده است.

•پیکان ها: نشانگر اطلاعات هستند از محیط خارجی به درون سیستم CLSS حرکت می کنند.

•**نمودار جریان سیستم (SFD)** *System* **Flow** *diagram*

زیر سیستمها در این نمودار تعریف مي شوند ، جریان اطلاعات(داده و کنترل) با جزییات بیشتر ایجاد شده اند.





فصل یازدهم: اصول و مفاهیم تحلیل

- **چه کسی کار را انجام میدهد:**

برای بکارگیری تحلیل نیازمندیها در زمینه کارهای تجاری پیچیده یک " **تحلیل گر سیستم**" اموزش دیده در جنبه های تجاری حوزه بکارگیری مینواند این کار را انجام دهد.

- **چرا این کار اهمیت دارد؟**

نتیجه تحلیل نکردن این است که پول و زمان هدر رفته و دلسردی شخصی پیش امده و مشتریان ناراضی خواهند بود.

- **مراحل کار چه میباشد؟**

نیازمندیهای داده ها .کارکردی و رفتاری از طریق **استخراج** اطلاعات از مشتری مشخص میشوند. نیازها **پالایش شده و تحلیل میشوند** تا وضوح و بی عیبی و **سازگاری درونی انها ارزیابی شود**. یک روش تعیین مشخصات که مدلی از نرم افزار را در خود لحاظ کرده است ایجاد شده و سپس توسط مهندسین نرم افزار و مشتریان/کاربران هر دو اعتبارات میابد.

- **حاصل کار چیست؟**

یک طریقه **ارائه موثر نرم افزار** باید بعنوان پیامد تحلیل نیازمندیها بوجود اید. نیازمندیهای نرم افزاری مانند نیازمندیهای سیستم میتوانند با استفاده از نمونه اولیه یک مشخص سازی یا حتی مدل نمادین ارائه شوند.

مهندسی نیازهای نرم افزاری یک فرایند کشف.پالایش.مدل سازی و مشخص سازی است.

- ***دیدگاه " دونالد ریفر " از مهندسی نیازمندیها:***

مهندسی نیازمندیها بکارگیری نظام مناصول اصلاح شده و تکنیک ها و زبانها و ابزارهای تحلیل هزینه موثر و تنظیم سند و تکامل جاری نیازهای مصرف کننده و مشخص ساختن رفتار برونی یک سیستم جهت جلب رضایت ان گروه از نیازهای کاربر میباشد.

نام درس:مهندسی نرم افزار ۱

تحلیل نیازها

تحلیل نیازمندیها وظیفه مهندسی نرم افزار بوده که شکاف بین مهندسی نیازمندیهای سطح سیستم و طراحی نرم افزار را پر میکند.

فعالیت مهندسی نیازمندیها منجر به **مشخص سازی خصوصیات عملیاتی** نرم افزار شده و بیانگر **تعامل با دیگر عناصر سیستم** بوده و محدودیتهای نرم افزار را تعیین میکند.

اقسام تحلیل نیازمندیهای نرم افزاری:

۱- فهم مشکل. ۲- ارزیابی و تلفیق. ۳- مدل سازی. ۴- مشخص سازی. ۵- مرور

ابتدا تحلیل گر مشخصات سیستم را مطالعه میکند. و بعد طرح پروژه نرم افزار را بررسی میکند. سپس برای تحلیل باید ارتباطات برقرار شود تا از فهم مشکل اطمینان حاصل شود.

هدف : شناخت پیدا کردن از عناصر مشکل اصلی است انگونه که توسط مشتری/ کاربر دریافت و استنباط میشود.

در طی ارزیابی و یافتن راه حل تمرکز اصلی تحلیل گر بر **روی " چه" تا " چگونه" است**.(جزییات توابع نرم افزاری و ...)

سیستم چه داده هایی را تولید و مصرف میکند؟ چه کارکرد هایی تعریف شده اند؟ و چه محدودیت هایی اعمال میشوند؟

در طی دوره ارزیابی و فعالیت یافتن راه حل تحلیل گر مدل هایی از سیستم در تلاش برای درک بهتر داده هاو جریان کنترل.پردازش کارکردها .رفتار عملیاتی و مضمون اطلاعات را خلق میکند. این مدل بعنوان زیر بنای طراحی نرم افزار و پایه خلق یک مشکل سازی برای نرم افزار خدمت خواهد کرد.

تعیین نیازمندیهای نرم افزار:

قبل از انکه بتوان نیازمندیها را تحلیل و مدل سازی و یا مشخص نمود باید انها را طی یک پروسه استخراج گرداوری نمود.

- **راه اندازی فرایند:**

رایج ترین تکنیک استخراج نیازمندیها برگزاری **نشست یا مصاحبه** میباشد. بنابراین ارتباط باید شروع شود.

" گاز و وینبرگ " : پیشنهاد میدهند که تحلیل گر ملاقات را با طرح سوالات خارج از مضمون اصلی شروع کند. یعنی یک سری سوالات که به شناخت اساسی مشکل افرادی که برای ان راه حل دارند و ماهیت راه حل مطلوب و کارایی اولین دیدار منتهی خواهد شد.

اولین سری سوالات رها از مضمون اصلی بر روی مشتری و اهداف و منابع کلی متمرکز میشوند. مثلا:

۱- چه کسی تقاضای این کار را داشته؟

۲- چه کسی از این راه حل استفاده خواهد کرد؟

۳- سود اقتصادی یک راه حل موفق چه خواهد بود؟

۴- ایا منبع دیگری برای راه حل مورد نظر شما وجود دارد؟

سری بعدی سوالات تحلیل گر را قادر میکند تا درک بهتری از مشکل و مشتری پیدا کرده و نظرات خود را درباره راه حل به او بگوید:

۱- چگونه خروجی خوب را که یک راه حل موفق بدنبال می اورد تشریح میکنید؟

۲- این راه حل چه مشکلاتی را مورد توجه قرار میدهد؟

۳- ایا امکان دارد که محیط بکار گیری راهحل را برایم تشریح کنید؟

۴- ایا مباحث ویژه عملکردی یا محدودیت هایی در مسیر یافتن راه حل وجود دارد؟

اخرین سری سوالات بر روی کارایی جلسه متمرکزند.(**فرا پرسش**)

" گایز و وینبرگ " به انها **فوق سوالات** گفته و لیست مختصر شده زیر را پیشنهاد میدهند:

۱- ایا شما فرد مناسبی برای پاسخ گویی به این سوالات هستید؟

۲- ایا پاسخ های شما رسمی هستند؟

۳- ایا سوالات من به مشکل شما مربوط میشوند؟

۴- ایا من بیش از اندازه در حال طرح سوالات هستم؟

۵- ایا فرد دیگری وجود دارد که بتواند اطلاعات اضافی در اختیار بگذارد؟

۶- ایا مورد دیگری وجود دارد که از شما سوال کنم؟

این سوالات به " شکستن یخ" کمک کرده و ارتباط را برقرار میکند که برای موفقیت تحلیل گر ضروری میباشد.

جلسه پرسش وپاسخ فقط برای اولین برخورد است و سپس باید جلساتی با چارچوب تلفیق عناصر حل مشکل مذاکره و مشخص سازی همراه ان برگزار شود.

فنون تسهیل مشخص سازی کاربردی: FAST

غالبا مشتریان و مهندسین نرم افزار دارای یک ذهنیت ناخود اگاه **" ما " و " انها "** هستند.(بجای انکه بصورت تیم فعالیت کنند) که این رهیافت نتیجه خوبی بهمراه ندارد. سوء تفاهم در حد بالا و اطلاعات مهم حذف شده و نیز هیچگاه رابطه کاری موفق برقرار نمیشود.

روش فنون تسهیل مشخص سازی (**FAST**) **ایجاد یک تیم مشترک از مشتریان و تحلیل گران را تشویق میکند** تا با یکدیگر در زمینه **تعیین مشکل** .پیشنهاد عناصر **راه حل.**مذاکره در مورد راهکارهای مختلف و مشخص ساختن یک مجموعه ابتدایی نیازمندیهای راه حل همکاری نمایند.

این تکنیک پتانسیل بهبود روابط را در کلیه زمینه های کاربردی ارائه میکند.

انواع رهیافت های FAST تغییراتی را اعمال میکنند که رهنمودهای زیر انها را ارائه میکند:

۱- ملاقاتی در مکانی بی طرف برگزار و مهندس نرم افزار و مشتریان حضور دارند.

۲- قوانین برای اماده سازی و تدارک تثبیت شده اند.

۳- دستور کاری پیشنهاد میشود که انقدر رسمی است تا نکات مهم را پوشش دهد ولی انقدر غیر رسمی نیز هست تا جریان ازاد عقاید را امکانپذیر می سازد.

۴- یک "تسهیل گر" جلسه را کنترل میکند.(یه نفر جلسه را کنترل کند)

۵- یک"مکانیسم تعریف"بکار میرود.(تابلوی اعلانات؛نمودار خاص..)

۶- هدف تعیین مشکل .پیشنهاد عناصر **راه حل.**مذاکره در مورد رهیافتهای گوناگون و تعیین مجموعه مقدماتی نیازمندیهای راه حل در یک فضای مفید برای تحقق هدف میباشد.

ملاقاتهای اولیه بین مشتری و مهندس نرم افزار برگزار و پرسش و پاسخهای اساسی به روشن شدن دامنه مشکل کمک کرده و ادراک کلی از راه حل بدست می اید. ماحصل این جلسات اولیه ان است که **مشتری و مهندس نرم افزار هر دو یک گزارش یک یا دو صفحه ای درباره درخواست محصول مینویسند.(نتیجه : جلسات با سه دسته سوال)**

مکان.زمان و تاریخ جلسه **FAST** انتخاب شده و یک تسهیل گر (ناظر) نیز انتخاب می شود.

نام درس:مهندسی نرم افزار ۱

شرکت کنندگان از هر دو طرف مشتری و مهندس نرم افزار برای شرکت در جلسه دعوت شده و درخواست محصول قبل از برگزاری جلسات توزیع می شوند.

از هر شرکت کننده خواسته می شود که (۱)لیستی از اشیاء.موضوعات اصلی.خدمات.قیود و معیارهای عملکردی تهیه نمایند.لیست نباید خسته کننده باشد اما باید چشم انداز خود را درباره سیستم منعکس کند.حتی با ارائه اطلاعات بیشتر هنوز ابهاماتی وجود دارد. احتمال حذف مواردی و امکان وقوع خطا نیز وجود دارد.

تیم "فنون تسهیل مشخص سازی" متشکل از نمایندگانی از بخش های بازاریابی و مهندسی نرم افزار و سخت افزار و تولید میباشد. یک تسهیل گر(ناظر) خارجی نیز باید حضور داشته باشد.

هر شرکت کننده لیستهای عوامل محدود کننده خود را تهیه میکند. (۲)پس از شروع جلسه اولین موضوع بحث نیاز و توجیه محصول جدید است. لیستها باید به نحوی ارائه شوند که بتوان انها را با یکدیگر ترکیب کرده یا حذف کرده و یا اضفاتی در انها انجام داد . در این مرحله انتقاد و بحث اکیدا ممنوع است.

پس از ارائه لیستها جداگانه (۳)یک لیست ترکیب توسط گروه در مورد یک موضوع تهیه میشود که این لیست لیستهای تکراری را حذف کرده و نظرات جدید را به انها اضافی میکند اما چیزی را پاک نمیکند.

پس از گرداوری و تنظیم لیستهای ترکیب(۴)مباحثه که توسط تسهیل گر هماهنگ می شود اغاز می گردد. لیست ترکیب کوتاه یا بلند شده یا مجددا نوشته می شود تا بتواند بخوبی منعکس کننده سیستم/محصول پیشنهادی برای تولید باشد.

هدف : بدست اوردن یک لیست **"اتفاق نظر"** در هر زمیته و موضوع میباشد.

زمانیکه لیستهای اتفاق نظر تکمیل شدند(۵) تیم مربوطه به تیم های کوچک تری تقسیم میشود که هر یک در جهت تهیه یک مشخص سازی کوچک برای یک یا چند مورد اضافه شده به هر یک از لیست ها فعالیت خواهد کرد.

(۶)" مینی – مشخص سازی "(جزییات) پرداختن معضل به یک کلمه یا عبارت بکار رفته در یک لیست می باشد.

هر تیم فرعی و سپس هر یک از مینی – مشخص سازیها خود را به کلیه شرکت کنندگان در **FAST**برای بحث روی ان ارائه می کند. اضافات و موارد حذف و توضیحات بیشتر انجام میشود. در برخی موارد توسعه مینی – مشخص سازیها به اشیاء خدمات یا محدودیتها یا نیازمندیهای عملکردی جدید منتهی میشود که به لیستهای اصلی اضافه خواهند شد.

پس از انکه مینی – مشخص سازیها تکمیل شد هر شرکت کننده در جلسه یک (۷)لیست معیار اعتبار برای محصول/سیستم تهیه کرده و لیست خود را به تیم ارائه میدهد. سرانجام یک یا چند شرکت کننده (۸)پیش نویس کامل مشخصات را بر عهده می گیرد.

تنظیم کارکرد کیفیت:

Quality Function Deployment

گسترش (تنظیم) کارکرد کیفیت **QFD تکنیک مدیریت کیفیتی است که نیازهای مشتری را به نیازهای فنی نرم افزار ترجمه می کند.**

QFDسه نوع نیازمندی را شناسایی کرده است:

۱- نیازمندیهای عادی : مقاصد و اهدافی که برای محصول یا سیستم در طول ملاقات با مشتری مطرح شده اند . چنانچه این نیازمندیها براورده شوند انگاه مشتری راضی خواهد بود.

۲- نیازمندیهای مورد انتظار : این نیازمندیها میتوانند به صورت تلویحی(ضمنی) درباره محصول یا سیستم مطرح شده و به میزانی بنیادی باشند که مشتری انها را بطور صریح مطرح نکند. فقدان انها میتواند باعث نارضایتی شدید شود.

۳- نیازمندیهای هیجان انگیز : این ویژگی ها فراتر از انتظار مشتری رفته و زمانی که ارائه می شوند بسیار رضایت بخش خواهند بود.

QFDاز مصاحبه ها.مشاهدات. پژوهش ها و ازمون های تاریخی داده ها بعنوان داده های خام برای جمع اوری نیازها استفاده می کند. سپس این داده ها به جدول نیازها – که جدول صدای مشتری نامیده میشود – ترجمه می شوند که از سئی مشتری مورد بررسی قرار می گیرند.انواع دیاگرام ها و پانویس ها و روش های ارزشیابی برای استخراج نیازهای مورد انتظار بکار می روند و تلاش می شود تا نیازهای هیجان انگیز استخراج شوند.

USE-CASE:

مهندس نرم افزار (تحلیل گر) میتوانند مجموعه ای از سناریوهایی را که یک رسمان کاربرد (مسیر استفاده)سیستمی که باید ساخته شود را تعیین نمایند. این سناریو ها که غالب **USE-CASE** نامیده می شوند شرحی از **چگونگی استفاده از سیستم** را فراهم می کنند.برای ایجاد یک **USE-CASE** باید در **اغاز انواع مختلف افراد/ابزارها را که از سیستم یا محصول استفاده خواهند کرد تعیین نماید.** این **بازیگرها** نقشهایی را بعنوان **اپراتورهای سیستم** ایفا میکنند را نشان میدهند. یک بازیگر زمانی که بطور رسمی تر تعیین میشود همان چیزی است که با سیستم با محصول ارتباط برقرار میکند و نسبت به سیستم یک **عامل خارجی** محسوب می شود.هر بازیگری و یا کاربری با یکدیگر یکسان نیستند . یک کاربر معمول می تواند تعداد مختلفی نقش را بهنگام بکارگیری سیستم ایفا نماید . در حالیکه **بازیگر** نماینده گروهی از عناصر خارجی می باشد که فقط ایفاگر **یک نقش** هستند.

از انجا که استخراج نیازمندیهای یک فعالیت رو به تکامل است تمام بازیگرها در ظرف اولین تکرار مشخص نمی شوند. امکان دارد که بازیگران اولیه را در اولین تکرار تعیین کرده و بازیگران ثانویه را پس از شناخت بیشتر سیستم معین کرد.

بازیگران اولیه برای دستیابی به کارکرد مطلوب سیستم وارد تعامل میشود و سود مورد نظر از سیستم را حاصل میکند . انها مستقیما و اغلب با نرم افزار کار میکنند.

بازیگران ثانویه برای حمایت سیستم حضور دارند با بازیگرا ن اولیه بتوانند در کار خود موفق شوند.

" یاکوبسن " پیشنهاد میدهد که **سوالات** زیر برای **توسعه USE-CASE**پاسخ داده شوند:

۱- وظائف و کارکرد های اصلی که بازیگر باید انجام دهد چیست؟

۲- چگونه سیستم اطلاعات توسط بازیگر تولید تغییر یافته یا بدست می اید؟

۳- ایا بازیگر باید سیستم را در مورد تغییرات در محیط خارجی ان مطلع کند؟

۴- چه اطلاعاتی را بازیگر از سیستم میطلبد؟

۵- ایا بازیگر علاقمند به مطلع شدن درباره تغییرات غیر منتظره هست؟

بطور کلی یک **USE-CASE** یک شرح روایتی ساده است که نقش هر بازیگر را ضمن تعامل با سیستم روشن میکند.

USE-CASEها سناریوهایی را شرح میدهند که توسط بازیگرهای مختلف بصورت متفاوت فهمیده میشوند.

" وایدر" پیشنهاد میکند:

پیاده سازی کارکرد کیفیت میتواند برای توسعه یک مقدار اولویت ارزیابی شده در رابطه با هر UseCase به کار گرفته شود.برای تحقق این امر **USE-CASE**از نقطه نظر تمام بازیگران تعریف برای سیستم ارزشیابی می شوند. یک مقذار اولویت توسط هر بازیگر برای هر **USE-CASE**تعیین میشود.

سپس یک اولویت میانگین محاسبه شده و اهمیت ادراک شده هر یک از **USE-CASE**مشخص میشود.

زمانی که یک مدل فرایند تکرار برای مهندسی نرم افزار معطوف به اشیاء بکار میرود اولویتها می توانند بر روی تعیین اولین قابلیت کارکردی سیستم که می تواند ارائه شود تاثیر بگذارند.

نام درس:مهندسی نرم افزار ۱

اصول تحلیل:

پژوهشگران مشکلات تحلیل را تعیین کرده و با تشخیص دلایل بوجود امده انها انواع راه های مدل سازی و مجموعه های راه گشای موثر در برطرف کردن انها را ارائه داده اند.

هر مدل تحلیل دارای یک نکته منحصر بفرد است .

تمام مدلهای تحلیل توسط مجموعه ای از اصول عملیاتی با یکدیگر مرتبطند (موارد مشابه در مدلهای تحلیل)

۱- حوزه اطلاعات یک مرحله باید بازنمایی و ادراک شود.

۲- کارکردهایی که نرم افزار باید انجام دهد مشخص شوند.

۳- رفتار نرم افزار باید بازنمایی شود.

۴- مدلهایی که نمایشگر اطلاعات و کارکرد و رفتار که باید به روشی تقسیم بندی شوند که جزئیات را به طریقی لایه بندی شده(سلسله مراتبی) اشکار نماید.

۵- فرایند تحلیل باید از اطلاعات اساسی به سوی جزئیات پیاده سازی حرکت کند.

" دیویس " یک مجموعه از اصول راهنما را برای مهندسی نیازها پیشنهاد میکند:

۱- مشکل را قبل از شروع به ایجاد مدل تحلیل درک کنید.

۲- نمونه های اولیه را ایجاد کنید که کاربر را قادر می سازد تا چگونگی تعامل کاربر- ماشین را ادراک نماید.

۳- ریشه و دلیل هر نیازمندی را ثبت کنید.

۴- از نظرات مختلف درباره نیازمندیها استفاده کنید.

۵- طبقه بندی نیازمندی ها .موعدهای زمانی می توانند پیاده سازی هر نیازمندی نرم افزاری را متوقف کنند.

۶- بر روی رفع ابهامات کار کنید.

نام درس:مهندسی نرم افزار ۱

میدان اطلاعات:

تمام کاربردها نرم افزار را میتوان بصورت گروهی " **پردازش داده ها**" نامید.

نرم افزار برای پردازش داده ها تغییر انها از شکلی به شکل دیگر بوجود می اید. بعبارت دیگر پذیرش **ورودی** به معنی کنترل و **پردازش** ان و سپس **بیرون دادن ان** است.

باید دقت کنیم نرم افزار **رخدادها را نیز پردازش** میکند. یک واقعه نماینده جنبه ای از کنترل سیستم است که در واقع چیزی بیش از داده های صفر و یک نیست.

داده ها(اعداد؛متن،تصاویر و...) و کنترل(وقایع) هر دو در میدان اطلاعات یک مسئله قرار می گیرند.اولین اصل تحلیل عملیاتی نیازمند یک میدان اطلاعات و ایجاد یک مدل داده ها میباشد.

میدان اطلاعات در بر دارنده سه نقطه نظر در رابطه با داده ها و کنترل است:

۱- مضمون (محتوای) اطلاعات و روابط (مدل داده ها).

۲- جریان اطلاعات.

۳- ساختار اطلاعات برای درک کامل میدان اطلاعات.

مضمون اطلاعات نماینده داده های جداگانه و اشیاء کنترل است که مجموعه ای بزرگتر از اطلاعات تغییر یافته توسط نرم افزار را تشکیل میدهند.

اشیاء داده ای و کنترل را میتوان به اشیاء داده ای و کنترل دیگر ربط داد.

جریان اطلاعات: نماینده طریقه تغییر داده ها و کنترل بگونه ای است که هر یک در سیستم حرکت می کنند. مطابق شکل (ص۳۶۸) اشیاء ورودی به اطلاعات میانی تغییر شکل میدهند که در نهایت به خروجی تبدیل می شوند .

در راستای این تغییر شکل مسیر اطلاعات اضافی می توانند از داده های موجود ذخیره شده معرفی شوند.

ساختار اطلاعات : نماینده سازمان درونی انواع داده ها و قلم های کنترل است.(ارایه – کلاس یا ....)

مدل سازی

مدلهای کارگردی برای درک بهتر انچه باید در عمل ساخته شوند خلق می شوند.

زمانی که این مسئله یک مورد **فیزیکی** است : مدلی ایجاد می کنیم که از نظر فرم و شکل یکسانند ولی از **نظر مقیاس کوچکترند**.

زمانی که مسئله مورد نظر **نرم افزار** باشد مدل باید **نماینده اطلاعاتی باشد که نرم افزار تغییر شکل میدهد**.

اصول دوم و سوم تحلیل عملیاتی نیازمند ساخت مدلهای کارکرد و رفتار است.

مدلهای کارکردی :

نرم افزار اطلاعات را با حداقل سه کارکرد عمومی تغییر شکل میدهد :

۱- ورودی. ۲- پردازش. ۳- خروجی.

سپس مهندس نرم افزار بر روی کاراکتر های ویژه مسئله متمرکز می شود.

مدل کارکردی با یک مدل در سطح بافت منفرد شروع می شود . طی یک سری تکرارها جزئیات کاراکتری بیشتری فراهم می شود تا اینکه یک تصویر از تمام قابلیت های کارکردی سیستم ارائه شود.

مدل رفتاری :

اکثر نرم افزار ها به **رخداد از جهان بیرونی پاسخ میدهد**. این خصوصیت محرک – پاسخ اساس مدل رفتاری را تشکیل میدهد.

مدلهای فراهم شده در طی تحلیل نیازمندیها چند **وظیفه و نقش را بر عهده دارند**:

۱- مدل به تحلیل گر کمک میکند تا اطلاعات و کارکرد و رفتار سیستم را درک کرده و از این طریق امر تحلیل نیازها را ساده تر کرده و بیشتر نظام مند نماید.

۲- مدل نقطه کانون توجه می شود و بنابراین کلیدی برای تعین کامل بودن سازگاری و دقت مشخص سازی خواهد بود.

۳- مدل پایه طراحی واقع شده و برای طراح شناختی اساسی از نرم افزار فراهم میکند تا بتواند برای پیاده سازی ان در بافت مربوطه " نقشه " تهیه نماید.

افراز (تجزیه و تقسیم):

مسائل اغلب بیش از اندازه بزرگ و پیچیده هستند بنابراین ما تمایل داریم تا انها را **تقسیم بندی** نماییم.

تحلیل چهارم عملیاتی چنین نظر میدهد که اطلاعات و کارکرد و حوزه رفتاری نرم افزار تقسیم بندی شوند.

تجزیه و تفکیک فرایندی است که نتیجه ان در تهیه کامل داده ها و کارکردها یا رفتار مشهود است . انرا میتوان بطور افقی یا عمودی انجام داد.

طریقه تقسیم بندی:

۱- عرضه جزئیات رو به افزایش با حرکت **عمودی** در سلسله مراتب.

۲- متلاشی نمودن مسئله بطور **کار کردی** از طریق حرکت **افقی** در سلسله مراتب.

دیدگاه اساسی و پیاده سازی:

یک **دید ضروری** از نیازمندیهای نرم افزار کارکردهایی را که باید اجرا شوند و اطلاعاتی را که باید پردازش شوند را بدون در نظر گرفتن جزئیات پیاده سازی ارائه میکند.

دید پیاده سازی نیازمندیهای نرم افزار ظاهر شدن کارکردهای پردازش در دنیای واقعی و ساختارهای اطلاعات را میسر میکند.

تحلیل گر باید محدودیت های تحمیل شده را که از سوی عناصر سیستم از پیش تعریف شده متوجه شده و دید پیاده سازی کارکرد و اطلاعات را زمانی که چنین دیدی مناسب است در نظر بگیرد.

دید پیاده سازی نباید ضرورتا بعنوان یک بازنمایی چگونگی تفسیر شود بلکه مدل پیاده سازی نمایندگی حالت عملیاتی جاری را بر عهده دارد.

مدل اساسی را بر این مفهوم که عملی شدن کارکرد بطور صریح اظهار نشده است متداول است.

نمونه سازی نرم افزار:

اشکال تحلیل با یکدیگر متفاوتند .

در برخی موارد امکان اعمال اصول تحلیل عملیاتی و بر گرفتن یک **مدل نرم افزاری** برای کار طراحی امکانپذیر است.

در دیگر موقعیت ها استخراج نیازمندیها انجام میشود. اصول تحلیل اعمال میشوند و **مدلی برای نرم افزاری** ایجاد میشوند که **نمونه اولیه نامیده می شود.**

نمونه اولیه برای ارزیابی مشتری و توسعه دهنده ساخته شده است.

در اخر شرایطی وجود دارند که ساخت یک نمونه اولیه را برای اغاز تحلیل ضروری میسازند.

سپس این مدل بصورت نرم افزار نهایی به تکامل میرسد.

انتخاب رهیافت نمونه سازی:

پارادایم ساخت نمونه اولیه می تواند " **انتها – بسته** " یا " **انتها – گشوده** " باشد.

راهکار انتها – بسته فالبا " نمونه دور انداختنی اولیه " نامیده میشود. با بکار گیری این راهکار یک نموته اولیه فقط بعنوان یک نماد کلی از نیازمندیها بکار می اید.

راهکار انتها – گشوده " نمونه اولیه سازی تکاملی " نامیده میشود. این راهکار از نمونه اولیه بعنوان اولین قسمت کار تحلیلی که به طراحی و تولید منتهی خواهد شد استفاده میکند.

نمونه اولیه نرم افزار اولین مرحله تکاملی سیستم به اتمام رسیده است.

پیش از انتخاب یکی از این دو راهکار باید در نظر گرفت که ایا برای سیستم مورد نظر میتوان نمونه اولیه ساخت یا خیر؟

عوامل کاندیدا در نمونه سازی اولیه:

۱- محدوده بکار گیری. ۲- پیچیدگی بکار گیری. ۳- خصوصیات مشتری. ۴- خصوصیات پروژه

کاندیدا نمونه اولیه سازی:

هر گونه کاربردی که صفحه نمایش های تصویری پویا خلق می کند شدیدا با کاربر تعامل میکند و یا به الگوریتم ها و فرایند های ترکیبی نیاز دارد که باید بطریق تکاملی بوجود اید را کاندیدایی برای نمونه اولیه سازی تلقی می کنیم.

اگر بکارگیری یک کاندیدا نیاز به ایجاد **ده ها هزار** خط برنامه داشته باشد **نمونه اولیه سازی بیش از حد پیچیده** است. اگر بتوان پیچیدگی را تقسیم بندی کرد امکان ساخت نمونه اولیه برای قسمتهای اولیه نرم افزار وجود دارد.

اگر مشتری بخواهد با نمونه اولیه در مراحل بعدی تعامل کند باید:

۱- منابع مشتری در خدمت ارزیابی و پالایش نمونه اولیه باشد.

۲- مشتری قادر به اتخاذ تصمیمات مربوط به نیازمندی بصورت بهنگام باشد.

شیوه ها و ابزار های نمونه سازی:

برای کارامد بودن نمونه اولیه نرم افزار یک نمونه اولیه باید با سرعت ایجاد شود تا مشتری نتایج را ارزیابی کرده و تغییرات را توصیه کند.

روشها و ابزارهایی که برای ارائه سریع نمونه اولیه سازی فراهم هستند:

۱- فنون نسل چهارم :

طبقه وسیعی از پرسش ها بر پایه داده ها و گزارش زبانها و برنامه و مولدهای بکارگریر و سایر زبانهای غیر – رویه ای سطح بالا را در بر میگیرد. این فن از انجا که مهندس نرم افزار را قادر به اجرا سریع برنامه میسازد برای نمونه اولیه سازی ایده ال است.

۲- اجزاء قابل استفاده مجدد نرم افزار :

راهکار دیگر جفت و جور کردن و سوار کردن اجزاء بجای ساختن نمونه اولیه است که با کمک اجزاء نرم افزار های موجود بکار گرفته میشود.

۳- مشخصات رسمی و محیط های نمونه اولیه سازی:

طی دو دهه گذشته تعدادی ابزارهای مشخص سازی رسمی بوجود امده اند که برای فنون مشخص سازی زبان جانشین می شوند.

طراحان این زبانهای رسمی در تعامل فرایند ایجاد محیط های تعاملی هستند که :

۱- تحلیل گر را قادر سازد تا بطود تعاملی یک مشخص سازی بر پایه زبان از سیستم یا نرم افزار را **خلق** نماید.

۲- از ابزار های خود کاری مدد بگیرد که مشخص سازی بر پایه زبان را برای برنامه های قابل اجرا **ترجمه** کند.

۳- مشتری را قادر سازد تا برنامه قابل اجرا نمونه اولیه را برای **پالایش نیاز مندیهای رسمی** بکار گیرد.

تعیین مشخصات:

حالت **مشخص سازی** با **کیفیت** راه حل ارتباط زیادی دارد.

اصول تعیین مشخصات:

مشخص سازی می تواند بعنوان یک فرایند نماد و نمود تلقی شود. نیازها به طریقه ای ارائه شده اند که در نهایت به پیاده سازی نرم افزار موفق منتهی می شود.

اصول مشخص سازی بر گرفته از تحقیق " بالزر و گودمن ":

۱- جدا کردن قابلیت کارکردی از پیاده سازی.

۲- توسعه مدلی از رفتار مطلوب سیستمی که داده ها و پاسخ های کارکردی سیستم را به انواع محرک های محیط در بر میگیرد.

۳- ایجاد متنی که نرم افزار از طریق مشخص سازی طریقه تعامل اجزاء سیستم با نرم افزار در ان عمل نماید.

۴- تعریف محیطی که سیستم در ان عمل میکند و روشن کردن چگونگی واکنش مجموعه ای شدیدا در هم تنیده از عوامل با محرک درون محیطی که توسط ان عوامل ایجاد شده است.

۵- ایجاد یک مدل عقلانی تا یک مدل طراحی یا پیاده سازی یک سیستم را انگونه که از سوی جامعه کاربران ادراک می شود تشریح میکند.

۶- فهم این مسئله که : مشخص سازی باید نواقص را تحمل کرده و قابل بهبود باشد" .

۷- ایجاد مضمون و ساختار مشخص سازی به طریقه ای که انرا قابل تطبیق با تغیرات نماید.

نام درس:مهندسی نرم افزار ۱

باز نمایی:

نیازمندیهای نرم افزار می تواند به طرق مختلف مشخص شود.

در این مورد رهنمودهای زیر ارزش بکارگیری دارد:

۱- نمایش و ارائه چارچوب و مضمون باید با مسئله مرتبط باشد.

۲- اطلاعات لحاظ شده در مشخص سازی باید تو در تو باشند.

۳- دیاگرامها و دیگر علائم باید از نظر تعداد محدود شده و از نظر بکارگیری سازگاری داشته باشد.

۴- بازنمایی باید قابل تجدید نظر باشد.

تعیین مشخصات نیازمندیهای نرم افزار :

مشخص سازی نرم افزار حاصل نهایی کار تحلیلی است.

معرفی مشخص سازی نیازهای نرم افزار بیانگر اهداف و مقاصد نرم افزار و شرح ان در بافت سیستم استوار بر کامپیوتر است.

شرح اطلاعات یک توصیف تفصیلی از مسئله ای را که نرم افزار باید طی کند فراهم می کند . مضمون اطلاعات و جریان و ساختار ان مستند می شوند. سخت افزار و نرم افزار و تعاملات انسانی برای عناصر سیستم خارجی و کارکردهای درونی نرم افزار شرح داده شده اند.

شرح هر کارکرد در شرح کارکردی ارائه میشود . یک شرح روایتی پردارش گر برای هر کارکرد ارائه میشود. محدودیتهای طراحی بیان و توجیه میشوند . خصوصیات عملکردی ان اظهار شده و یک یا چند دیاگرام برای نمایش گرافیکی ساختار کلی نرم افزار لحاظ میشوند.

شرح رفتاری مشخص سازی عملیات نرم افزار را بعنوان خصوصیات سلسله ای از حوادث بیرونی و درونی مورد ازمون قرار میدهد.

معیار اعتبار سنجی مهمترین و در عین حال مورد بی اعتنایی ترین بخش مشخص سازی نیازهای نرم افزار است.

سرانجام مشخص سازی شامل یک " کتاب شناسی " و " پیوست" میشود.

کتاب شناسی ارجاع به تمام مستندات مربوط به نرم افزار دارد.

نام درس:مهندسی نرم افزار ۱

باز بینی مشخصات:

از انجا که مشخص سازی زیر بنای مرحله توسعه را تشکیل داده باید در مورد بررسی و مرور دقت زیادی صورت گیرد. مرور در اغاز در سطح ماکروسکوپیک (کلان) انجام میشود. برای بررسی کامل هر یک از حوزه ها باید بررسی دارای جزئیات بیشتری شود و نیز طریقه ارائه شدن نیازمندیها بررسی و کنترل می شود.

پس از کامل شدن بررسی مشخص سازی نیازهای نرم افزار از سوی تحلیل گر و مشتری به کناری نهاده می شود .مشخص سازی برای توسعه نرم افزار مانند یک "قرارداد" شده است. تقاضا در مورد اعمال تغییرات در نیازها پس از نهایی شدن مشخص سازی حذف نخثاهند شد. اما مشتری باید بداند که اینگونه تغییرات پس از نهایی شدن کار بمعنای صرف هزینه بیشتر قرارداد خواهد بود.

ازمودن مشخص سازی بهر روش معنی داری مشکل و دشواراست . بنابراین عدم سازگاری یا موارد حذف شده می توانند بدون انکه تشخیص داده شوند وارد کار شوند. در طول بررسی تغییرات اعمال شده در مشخص سازی توصیه می شود

این جزوه خلاصه بوده و سعی شده مطالب اصلی و مهم درج شود و موارد اشتباه و نادرست حذف شده باشد
امیدوارم توانسته باشم کمکی به همه دوستان و دانشجویان پیام نوری و اساتید و همکارانی که در این ترم از اسلایدها استقبال نمودند و به عنوان کمک آموزشی استفاده کردند و سایر افرادی که نیاز به مطالعه کتاب دارند انجام داده باشم.....اشتباهات موجود را به بزرگواری خود ببخشین.
شاد و پیروز باشین – یا حق – ایمان مختاری