مبانی مهندسی نرم افزار

مدرس: سپیده برادران هزاوه

پاییز ۹۳


بسم الله الرحمن الرحيم

مراجع برای مطالعه بیشتر

۱. تحلیل و طراحی سیستم ها در مهندسی نرم افزار		تالیف: دکتر سعید پارسا
۲. مهندسی نرم افزار	اثر: راجر.اس.پرسمن	ترجمه: محمدمهدی سالخورده حقیقی
۳. مهندسی نرم افزار	اثر: یان سامرویل	ترجمه: عین اله جعفرنژاد قمی
۴. تحلیل و طراحی سیستم ها	اثر: ایگور هوریس کیوویچ	ترجمه: عین اله جعفرنژاد قمی

ایمیل و وبلاگ

Baradaran.hezaveh@gmail.com
Baradaran-hezaveh.blog.ir

بارم بندی نمرات

پروژه	۴ نمره	(۲ نمره ارایه + ۲ نمره متن پروژه)
پایان ترم	۱۶ نمره	
مجموع	۲۰ نمره	

مفاهیم و اصطلاحات

نرم افزار چیست ؟

- نرم افزار مجموعه ای از عناصر یا اشیا است که تشکیل یک پیکربندی شامل موارد زیر می‌دهند:
  - ✓ برنامه ها یا دستورالعمل ها (وقتی اجرا می شوند وظیفه یا عملکرد مورد نظر را تامین می کنند.)
  - ✓ مستندات (نحوه عملکرد و طریقه استفاده از نرم افزار را بیان می کنند.)
  - ✓ داده (ساختمان داده هایی که برنامه را به پردازش اطلاعات قادر می سازند.)

- نرم افزار دستورالعمل هایی است که منجر به استفاده از سخت افزار می شوند.
- محصولات نرم افزار ممکن است برای یک مشتری خاص و یا برای بازار عمومی تولید شوند.

- محصولات نرم افزار ممکن است :
  - کلی باشند، توسعه آنها به منظور فروش به طیف وسیعی از مشتریان باشد. به عنوان مثال نرم افزارهایی مثل Word و یا Excel
  - خاص باشند، برای یک مشتری خاص و با توجه به نیازهای خاص مشتری توسعه داده شده باشند.

- هر نرم افزار جدید می تواند با تولید کامل یک برنامه جدید، پیکربندی یک نرم افزار عمومی و یا استفاده مجدد از یک نرم افزار موجود به وجود آید.

نقش دوگانه نرم افزار

- نرم افزار یک محصول است.
  - توانایی پردازش را فراهم می سازد.
  - وظیفه تولید، مدیریت، اکتساب، تغییر، نمایش و یا انتقال اطلاعات را بر عهده دارد.

- نرم افزار ابزاری برای تحویل یک محصول است.
  - از کارکردهای سیستم پشتیبانی می کند.
  - نرم افزارهای دیگر را کنترل می کند. (مثل سیستم عامل)
  - برای ایجاد ارتباط به کار می رود. (مثل نرم افزارهای شبکه)
  - در ساخت نرم افزارهای دیگر سودمند است. (مثل ابزارهای نرم افزاری)

- ❖ نرم افزار مهندسی می شود.
- ❖ نرم افزار فرسوده نمی شود.
- ❖ نرم افزار پیچیده است.
- ❖ نرم افزار تفکیک کننده است.

انواع نرم افزار

۱-تقسیم بندی های فنی نرم افزار

۱-۱- نرم افزارهای پایه

۱-۲- نرم افزارهای سیستمی

۱-۳- نرم افزارهای کاربردی

۱-۳-۱- بدافزارها

۱-۳-۲- ابزارهای تبلیغاتی مزاحم

۱-۳-۳- نرم افزار انبار گردانی

۱-۳-۴- نرم افزارهای جانبی

۱-۳-۵- نرم افزار پشتیبان و بازیابی اطلاعات

۱-۳-۶- نرم افزار های پردازش داده ها

۱-۳-۷- نرم افزارهای شبیه سازی و مدل سازی

۱-۳-۸- نرم افزارهای سیستم خبره

۱-۳-۹- نرم افزار های سیستم بی درنگ

۱-۳-۱۰- نرم افزارهای سسیتم نهفته

۲-تقسیم بندی های حقوقی نرم افزار

۲-۱- نرم افزارهای سفارشی

۲-۲- نرم افزارهای ردهٔ عام

۲-۳- نرم افزار رایگان

۲-۴- نرم افزارهای منبع باز و منبع بسته

۲-۵- نرم افزارهای اختصاصی

۲-۶- نرم افزارهای مشروط و نرم افزارهای جزئی

۲-۷- نرم افزار اختراعی و نرم افزار کپی رایتی

۲-۸- نرم افزار مشاع و نرم فزار غیر مشاع

۲-۹- نرم افزار مکمل و نرم افزار سازگار

۲-۱۰- نرم افزارهای واسط و غیر واسط

تقسیم بندی های فنی نرم افزار

به جهت فنی و ابعاد مرتبط با علوم رایانه ای، نرم افزار ها را با توجه به معیارهای گوناگون از جمله هدف و مأموریت نرم افزار، زمینهٔ استفاده، نوع نقش و عملکرد و یا کاربر نرم افزار، می توان مورد تقسیم بندی‌های مختلفی قرار داد. ذیلاً با توجه به معیارهای پیش گفته به بررسی انواع نرم افزارها از جهت فنی خواهیم پرداخت.

۱- نرم افزارهای پایه (Programming Software)

این دسته از نرم افزارها، یکی از انواع معمول، شناخته شده و مورد علاقهٔ کاربران در میان نرم افزار های کامپیوتری است . این نرم افزار در قالب ابزار بوده و به برنامه نویس در نوشتن برنامه های کامپیوتری کمک می کند. برنامه های کامپیوتری مجموعه ای از دستورات منطقی هستند که برای یک سیستم کامپیوتری، وظایف خاصی را انجام می دهند . ابزارهایی که به برنامه نویسان در ایجاد یک سیستم کامپیوتری کمک می کنند، شامل ویرایشگر متن ، کامپایلرها و مترجم ها ست. کامپایلرها(مفسرها)، کد منبع را که در قالب یک زبان برنامه نویسی، نوشته شده اند به زبانی که کامپیوتر آن را می فهمد، ترجمه می کنند.(اغلب در قالب دو دویی). کامپایلرها چیزهایی را که به وسیلهٔ رابط ها، تجمیع و تبدیل شده اند، تولید می کنند. دی باگرها(اِشکال زداها)، جهت بررسی و اشکال زدایی کدها استفاده می شوند. کد منبع، بعضاً یا به طور کامل، برای ابزارهای اشکال زدا Debugging tool که بر روی آنها اجرا شده و به جهت برطرف کردن هرگونه اشکال احتمالی به کار می روند، شبیه سازی می شود. مترجم ها Interpreters برنامه ها را اجرا می کنند. آنها کد منبع و یا یک کد از پیش تالیف شده را اجرا و یا کد منبع را قبل از اجرا به یک زبان میانی ترجمه می کنند.

۲- نرم افزارهای سیستمی System Software

این گونه از نرم افزارها به راه اندازی و اجرای سخت افزار رایانه ای و سیستم رایانه، کمک می کنند. نرم افزارهای سیستمی به سیستمهای عامل، درایورها، سرورها و برنامه های جانبی سیستمی utilitiesاطلاق می شود. نرم افزار سیستمی به یک برنامه نویس کاربردی در خصوص جداسازی و انتزاع زبان برنامه نویسی از سخت افزار، حافظه، و سایر اجزاء مرکب درونی یک رایانه، کمک می کند تا خودش را درگیر زبان ماشین نکند. یک سیستم عامل، برای کاربرها، با یک پلت فرم، امکان اجرای برنامه های سطح بالا را فراهم می آورد.

میان افزارها و سیستم ورودی و خروجی بایوس، ابزاری را فراهم می کنند تا سخت افزار به کار گرفته شود.

۳- نرم افزارهای کاربردی Application Software

این قسم از نرم افزارها، کاربر نهایی را قادر می سازد تا امور معینی را به انجام رساند. نرم افزارهای مربوط به کسب و کار، پایگاههای داده و نرم افزارهای آموزشی، برخی از اشکال نرم افزارهای کاربردی هستند. همچنین واژه پردازهای مختلف که باید توسط کاربر، به انجام کارهای تخصصی اختصاص داده شوند، نمونه های دیگری از نرم افزارهای کاربردی هستند.

۳-۱- بدافزارها Malware

بدافزار، اشاره به هر گونه نرم افزار مخرب داشته و یک طیف وسیع تر از نرم افزارهایی را در بر می گیرد که به هر شکل، تهدیدی برای امنیت رایانه می باشند. ابزارهای تبلیغاتی مزاحم، جاسوس افزارها، ویروس های رایانه ای، کرم های رایانه ای، اسب های تروجان و یا ترس افزارها، مصادیق نرم افزارهای مخرب می باشند. ویروس های رایانه ای، برنامه های مخربی است که به تنهایی قادر به تکثیر خود بوده و از یک رایانه به رایانهٔ دیگر در محیط شبکه و یا اینترنت گسترش می یابند. کرم رایانه ای همین کار را انجام می دهد، تنها با این تفاوت که ویروس ها نیاز به یک برنامه میزبان ضمیمه دارند و با آن گسترش می یابند، در حالی که در خصوص کرم ها لازم نیست تا خود را به برنامه ای ضمیمه کنند. تروجان نیز خود را تکثیر و اطلاعات را سرقت می کند. نرم افزارهای جاسوسی می توانند بر فعالیت های کاربر بر روی یک رایانه، نظارت داشته و اطلاعات کاربر را بدون اینکه وی بفهمد، سرقت کنند.

۳-۲- ابزارهای تبلیغاتی مزاحم Adware

ابزارهای تبلیغاتی مزاحم نرم افزارهایی هستند که با استفاده از آن، تبلیغات اینترنتی در فضای مجازی ، اجرا یا دانلود می شوند. برنامه نویسان، ابزارهای تبلیغاتی مزاحم را به عنوان وسیلهٔ تولید درآمد خود طراحی می کنند. آنها، اطلاعات کاربر، مانند وب سایت هایی را که وی اغلب بازدید می کند و صفحاتی که به عنوان صفحهٔ مورد علاقه، ثبت کرده را استخراج می کنند. تبلیغاتی که به عنوان پاپ آپ در صفحهٔ نمایش شما ظاهر می شوند، ناشی از برنامه های تبلیغاتی مزاحم هستند که شما را ردیابی می کنند. اما ابزارهای تبلیغاتی مزاحم برای امنیت رایانه و یا حریم خصوصی کاربر، مضر نیست؛ بلکه داده ها را جمع آوری کرده و تنها به وسیلهٔ پیشنهاد از طریق کلیک کاربر بر روی تبلیغات عمل می کنند.

برخی دیگر از نرم افزار ها در میان سایر نرم افزارهای رایانه ای، مانند نرم افزار مدیریت موجودی، برنامه‌ریزی منابع سازمانی ، نرم افزارهای جانبی و نرم افزار حسابداری وجود دارندکه در زمینهٔ خاص اطلاعاتی و سیستم های مدیریت داده ها، کاربرد دارند. ذیلاً نگاهی گذرا به برخی از آنها، خواهیم داشت.

۳-۳- نرم افزار انبار گردانی Inventory Management Software

این نوع از نرم افزار به یک سازمان در ردیابی کالاها و مواد خود بر اساس کیفیت و کمیت کمک می کند. توابع مدیریت موجودی انبار شامل نقل و انتقالات انبار داخلی و ذخیره سازی می باشد. این نرم افزار انبار کمک می کند تا یک شرکت در سازماندهی موجودی و بهینه سازی جریان کالا در سازمان خود بهتر عمل کرده و در نتیجه این امر به بهبود خدمات به مشتریان منجر می شود.

۳-۴- نرم افزارهای جانبی Utilities Software

همچنان که مطابق عرف معمول خدمات نرم افزاری شناخته شده اند، این گونه از نرم افزارها در خصوص مدیریت سخت افزار رایانه و نرم افزارهای رایانه ای کمک می کنند. این فرآیند نرم افزاری، طیف محدودی از وظایف و عملکرد ها را بر عهده دارد. یک پارچه سازهای دیسک سخت Disk defragmenters، نرم افزارهای سیستمی جانبی و ویروس یاب ها، برخی از نمونه های متداول نرم افزار های جانبی هستند.

۳-۵- نرم افزار پشتیبان و بازیابی اطلاعات Data Backup and Recover Software

یک نرم افزار پشتیبان و بازیابی اطلاعات ایده آل، ویژگی هایی فراتر از کپی ساده از فایل های داده را فراهم می کند. این نرم افزار اغلب، نیازهای کاربر را در خصوص تشخیص موارد و زمان پشتیبانی و حمایت برنامه ها، برطرف می کند. نرم افزارهای پشتیبانی و بازیابی، سازمان مندی اصلی فایل را حفظ کرده و هرگونه بازیابی آسان از اطلاعات پشتیبانی شده را ممکن می سازد.

۳-۶- نرم افزار های پردازش داده ها :

این مورد، رایج ترین زمینه برای تولید نرم افزار و استفاده از رایانه است. سیستم هایی نظیر حسابداری، انبارداری، حقوق و دستمزد و فروش، در این خصوص قابل ذکرند.

۳-۷- نرم افزارهای شبیه سازی و مدل سازی :

به جهات اقتصادی و ایمنی و صرفه جویی در وقت، برای آموزش و تحقیق در بسیاری از موارد، از این قابلیت در رایانه استفاده می شود. آموزش خلبانی و طراحی بدنه اتومبیل و امثال اینها، مصادیق این دسته اند.

۳-۸- نرم افزارهای سیستم خبره :

مقصود ساختن برنامه هایی است که بتواند کارهای هوشمندانه انسان نظیر گفتگو، ترجمه و تفسیر زبان و معاینه را انجام دهد. سیستم های باهوش مصنوعی، بر اساس منطق است و در آن از پایگاه دانش نیز علاوه بر پایگاه داده، استفاده می شود.

۹-۳- نرم افزار های سیستم بی درنگ :

در این سیستم ها، عکس العمل بلافاصله صورت می گیرند. نرم افزار های بدین منظور بیشتر در مراکز تولیدی نظیر پالایشگاه مورد استفاده دارد که با بروز هر عامل و علامتی، بی درنگ عکس العمل نشان داده و به تنظیم فرآیند تولید می پردازد.

۱۰-۳- نرم افزارهای سسیتم نهفته :

بسیاری از وسایل کوچک و بزرگ نظیر اسباب بازی، اتومبیل و تجهیزات پزشکی وجود دارند که در آنها کامپیوترهای ریزی به کار رفته است و با برنامه هایی که روی آنها نصب گردیده، کار آن دستگاه کنترل می شود.

تقسیم بندی های حقوقی نرم افزار

با توجه به اینکه نرم افزار پدیده ای فنی بوده و در عین حال، دارای ارزش اقتصادی و مطلوبیت عرفی، می باشد، نیازمند مجموعۀ قواعد و حمایت های حقوقی است تا بتوان به وسیلۀ آن، روابط پدیدآورندگان و مالکان نرم افزار را با مشتریان و مصرف کنندگان تنظیم کرده و از بروز اختلافات و نابسامانی ها و همچنین سوءاستفادۀ ناقضان احتمالی حقوق صاحبان حق در این حوزه جلوگیری به عمل آورد. در این راستا دسته بندی نرم افزار از حیث محدوده و نحوۀ بهره برداری کاربران و مشتریان نرم افزار ، امری بسیار حائز اهمیت است که ذیلاً به آن پرداخته خواهد شد.

۱- نرم افزارهای سفارشی Custom Software

نرم افزاری است که برای یک کاربر یا سازمان خاص طراحی شده و از آنجا که برای یک کاربر خاص ساخته شده است، مشخصات و ویژگی های آن مطابق با نیاز کاربر می باشد.

البته عبارت نرم افزار سفارشی نرم افزاری را به ذهن متبادر می کند که طی قرارداد سفارش اثر تولید شده. در واقع می توان نرم افزارها را به لحاظ حقوقی و از حیث دارنده حق به نرم افزارهایی که طی قرارداد سفارش تولید شده اند و نرم افزارهایی که طی رابطه کارگر و کارفرمایی تولید شده اند، از یک سو و از حیث دیگر نرم افزار را به نرم افزارهایی که یک پدیدآورنده دارند در مقابل نرم افزارهایی که اثر مشترک محسوب می شوند؛ تقسیم بندی نمود.این در حالی است که به اعتبار دیگر نرم افزارها را می توان به اختصاصی و متن باز، تقسیم کرد. در واقع نرم افزار اختصاصی یا متن باز می تواند به اشکال مختلف و در روابط مختلف(از حیث نخستین دارنده حقوق) تولید شده باشد.

۲- نرم افزارهای ردهٔ عام Off-the-Shelf Software

برخلاف نرم افزار های سفارشی، نرم افزار ردهٔ عام، بدون در نظر گرفتن رده و کاربر خاص، خریداری می شود. این نرم افزار، ممکن است جهت منظوری خاص یا در خصوص نیازهای کاربری خاص، طراحی شده باشند و یا به این منظور تولید نشده باشند، لیکن قابلیت استفادهٔ عام داشته باشند و طیف وسیعی از کاربران بتوانند با استفاده از آن نیاز خود را در زمینهٔ مطلوب خودشان برطرف نمایند؛ بدون اینکه هیچ گونه هماهنگی و ارتباطی با طراح نرم افزار داشته باشند. در حقیقت شما هنگام خرید این نرم افزار، با شرایط قرارداد لیسانس آن، موافقت می کنید.

۳- نرم افزار رایگان free Software

نرم افزاری است که یک کاربر، برای استفاده، تغییر و توزیع آن، آزاد است. نرم افزار رایگان به طور کلی بدون هزینه از آب در می آید. لیکن هزینه ها مشتمل است بر توزیع، ارائه خدمات، و نگهدارای و پشتیبانی. واژهٔ رایگان به آزاد بودن نرم افزار از نظام کپی رایت، توزیع و جرح وتعدیل اشاره دارد.البته باید در نظر داشت، نرم‌افزارهای رایگان به طور رایگان قابل دانلود و استفاده هستند اما ممکن است لزوما قابلیت استفاده مجدد و تغییر و اصلاح توسط کاربر را نداشته‌باشند. هردو نوع نرم‌افزارهای آزاد و نرم‌افزارهای متن‌باز در یک ویژگی مشترک هستند: همه‌ی کاربران آن ها باید به کدهای منبع دسترسی داشته باشند. بدین معنی که کدهای منبع این نرم‌افزارها به شخص یا شرکت خاصی اختصاص ندارند و می‌توان آن‌ها را به اشتراک گذاشت.

۴- نرم افزارهای منبع باز و منبع بسته Open Source and Closed Source Software

در نرم افزارهای مدل منبع بسته، منبع نرم افزار برای عموم منتشر نشده است؛ در حالی که در نرم افزارهای منبع باز، کد منبع آن برای اصلاح و استفاده در دسترس است. نرم افزارهای منبع باز در قالب کد منبعشان در دسترس هستند و حق تغییر، بهبود و ارتقاء و بعضاً حق انتشار کد آن از طریق لیسانس های نرم افزار اعطا می شود. در جایی که نرم افزار برای عموم مردم تولید شده باشد به آن نرم افزار منبع باز اطلاق می شود؛ چه اینکه نرم افزار یاد شده توسط یک شرکت و یا توسط یک شخص تولید شده باشد.

۵- نرم افزارهای اختصاصی Proprietary Software

در نرم افزارهای اختصاصی، حقوق قانونی منحصراً برای دارندهٔ حق تکثیر باقی مانده و اکثر نرم افزارهای اختصاصی به صورت منبع بسته در دسترس قرار می گیرند.

این نوع نرم‌افزارها متعلق به یک شرکت یا شخص حقیقی است و استفاده از آن‌ها مستلزم خرید یا کسب مجوز بهره‌برداری از دارنده است. دسترسی به کدهای منبع و تغییر آن‌ها توسط کاربر در نرم‌افزارهای اختصاصی، امکان‌پذیر نیست. مطابق شرایط و ضوابط آنچه که « موافقتنامه مجوز بهره‌برداری کاربر نهایی» نامیده می‌شود، کاربران مجاز به تکثیر، به اشتراک گذاشتن، تغییر، توزیع مجدد یا مهندسی معکوس کدهای منبع نیستند.

برخی فروشندگان نرم افزار، کد منبع نرم افزارهای اختصاصی را با دسترسی محدود، میان مشتریان توزیع می کنند.نرم افزار اختصاصی به شکل نرم افزار مشروط یا نرم افزار معرفDemo wareخواهد بودکه کاربران جهت استفاده از آنها وجهی را پرداخت نمی کنند، و فی الواقع ترکیبی از نرم افزارهای رایگان و مشروط هستند.

اینگونه نرم افزارها مشتمل بر هزینهٔ بسته بندی نیست، زیرا در قالب بسته بندی بازاری ارائه نمی شوند؛ اینگونه نرم افزارها به صورت اینترنتی یا در قالب های سخت افزارهای حامل که نیاز به بسته بندی تجاری مجزا ندارند، به مشتریان ارائه می شوند. با این وجود ممکن است برنامه نویسان از شما بخواهند تا مبلغ اندکی را جهت استحقاق دریافت نسخه های پشتیبان و کمکی، بپردازید.

۶- نرم افزارهای مشروط و نرم افزارهای جزئی Shareware and Retail Software

در حالی که نرم افزارهای مشروط به عنوان نسخهٔ آزمایشی به کاربران عرضه می شود، نرم افزارهای جزئی به کاربران نهایی فروخته می شوند. با افزایش دسترسی به نرم افزارهای مشروط و رایگان بر روی اینترنت، بازار نرم افزارهای جزئی تغییر می کند. طراحان و فروشندگان، شروع به ارائهٔ نرم افزارهایشان جهت فروش بر روی اینترنت می کنند. مکرراً نرم افزارهای مشروط به عنوان نرم افزاری غیر فعالcripple ware، که در آنها ابعاد اصلی نرم افزار عمل نمی کنند و پس از اتمام دورهٔ آزمایش به طور کلی از کار می افتند. اگرچه نرم افزار های مشروط قالب های بسیار مشهوری هستند که در این قالب، نرم افزارهای جزئی کاملاً از این شهرت و تداول برخوردار نیست. برای مثال نرم افزار Microsoft officeیک نرم افزار بسته بندی شدهٔ جزئی است که می بایستی خریده شوند. نرم افزار های جزئی ممکن است به عنوان بستهٔ تجهیزات اصلی تولید منتقل شوند OEMدر حال حاضر طراحان نرم افزار، به تولید کنندگان بزرگ بر اساس یک قرارداد لیسانس یک نسخه از کپی نرم افزار را به ایشان تحویل می دهند تا قبل از خرید امکان نصب نرم افزار مزبور را بر روی دستگاه های رایانه خود نصب کنند. بنابراین در بسته های موسوم به Box Package Formیک نسخه کپی از نرم افزار مجوز دار از طریق نمایندگی های مجاز به مشتریان تحویل داده می شود.

۷- نرم افزار اختراعی و نرم افزار کپی رایتی Patent software and copy right software

نرم افزار ازجهت این که تابع احکام کدام رژیم حمایتی حقوقی باشد قابل دسته بندی است. دو نظام حقوقی معروف و نسبتاً جا افتاده برای اموال غیر مادی وجود دارد که جنبه بین المللی نیز یافته و کنوانسیون های متعددی در خصوص آنها به تصویب رسیده است. نظام حق مولف یا کپی رایت و نظام حق اختراع. یکی از معمولترین نظام های پیشنهادی برای حمایت از پدیدآورندگان نرم افزار، نظام حق مولف است. برخی مشابهت ها بین یک اثر ادبی همانند کتاب و یک برنامه رایانه ای به خصوص در ابتدای عمر مهندسی نرم افزار که به شکل پیشرفتهٔ کنونی مطرح نبود، سبب شد قوانین کپی رایت بسیاری از کشورها، طوری اصلاح شوند که از نرم افزارهای رایانه ای نیز همانند آثار ادبی حمایت شود. این در حالی است که تفاوت های در موضوع و نیز عدم تناسب احکام حق مولف با نرم افزار موجب شده است نظرها به نظام حقوق صنعتی(حق اختراع) معطوف شود و البته حق اختراع نیز کاملاً برای نرم افزارها مناسب نبوده و نقطه ضعف هایی بر آن بار می شود.

۸- نرم افزار مشاع و نرم فزار غیر مشاع

هرگاه اشخاص متعدد در پدیدآوردن نرم افزار ، مشارکت داشته باشند، چنانچه سهم مشارکت هریک در پدیدآوردن نرم افزار مشخص باشد، حقوق مادی حاصل از آن به نسبت مشارکت به هریک تعلق می گیرد. در صورتی که کار یکایک آنان جدا و متمایز نباشد، اثر مشترک نامیده می شود و حقوق ناشی از آن، حق مشاع پدیدآورندگان است. البته باید خاطر نشان کرد که هریک از شرکا به تنهایی یا همه آنها به اتفاق می توانند در مورد نقض حقوق موضوع قوانین مربوطه به مراجع قضایی، مراجعه کنند.

۹- نرم افزار مکمل و نرم افزار سازگار

یکی از حقوقی که همیشه در مورد نرم افزار و سایر آثار دارای حق مالکیت فکری مورد بحث و اختلاف نظر بوده است «حق تولید سازگار» است که در مورد آثار ادبی و هنری با عنوان تولید آثار اشتقاقی یا تلفیقی عنوان می‌شود. به زبان ساده، آثار اقتباسی آثاری هستند که ریشه در آثار قبلی دارند. اثر جدید اگر چه از نظر محتوا و شکل بیان دارای اصالت است اما برخی از عناصر را از اثر پیشین کسب کرده است. این امر در نرم افزارها نیز بسیار رایج است. در برخی از نظام های حقوقی ، استفاده از نرم افزارهای دیگر برای ایجاد نرم افزارهای سازگار و مکمل که قابلیت ها و ظرفیت ها یا کاربری جدید ایجاد کند بلا مانع است و نقض حقوق پدیدآورنده نرم افزارهای دیگر محسوب نمی شود؛ مشروط بر اینکه پدیدآورنده نرم افزار سازگار و مکمل، رضایت کتبی پدیدآورندگان نرم افزارهایی که برای نخستین بار درآن کشور، تولید شده است را گرفته باشد.

۱۰- نرم افزارهای واسط و غیر واسط

ماده ۴ قانون حمایت از پدیدآورندگان نرم افزارهای رایانه ای چنین مقرر داشته که :

« حقوق ناشی از آن بخش از نرم افزاری که به واسطه نرم افزارهای دیگر پدید می آید متعلق به دارنده حقوق نرم افزارهای واسط نیست».

منظور از واسط، آن چیزی است که میان یک نرم افزار با نرم افزار دیگر یا سخت افزار ارتباط ایجاد می‌کند زیرا یک نرم افزار به تنهایی کاری نمی‌تواند انجام دهد مگر اینکه از طریق واسط‌ها داده‌ها را بگیرد و از طریق سخت افزار در اختیار کاربر قرار دهد به عبارت دیگر اینکه کاربر چگونه از صفحه کلید یا صفحه نمایش با سخت افزار یا نرم افزار سیستم ارتباط بر قرار کند، به ماهیت آن جزء از برنامه بستگی دارد که واسط کاربر آن است.

اغلب دعاوی، ادعای کپی شدن بخشی از برنامه است که عمدتا شامل همین واسط‌ها است. اما نکته خاصی که در برخی آراء مورد توجه قرار گرفته است، محدودیت‌های اجباری است که در تولید نرم افزار جدید اتفاق می‌افتد. عوامل ثابتی که در این صنعت وجود دارد، خواه ناخواه منجر به شباهت برخی از بخش‌ها و ساختارها می‌گردد زیرا مشخصات رایانه‌ای که به وسیله آن نرم افزار طراحی می‌شود، استانداردهای طراحی، تقاضاهای موجود در بازار و رویه‌های برنامه نویسی که به صورت عرف در آمده است، محدودیت‌هایی را در تنوع و تمایز محصول ایجاد کرده است.

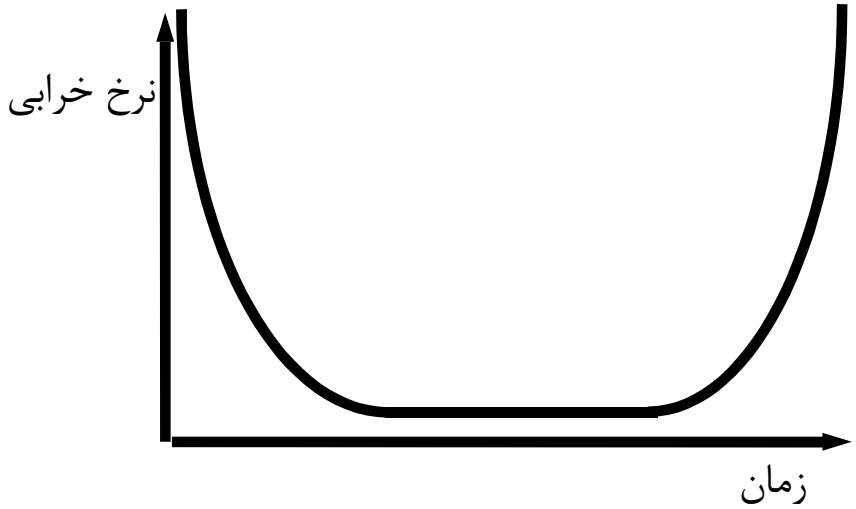
همچنین در تعریف نرم افزارهای مذکور گفته اند، منظور از نرم افزارهای واسط یا رابط برنامه‌نویسی نرم‌افزار Application Programming Interface یا API یا به صورت خلاصه رابط برنامه نویسی، رابط بین یک کتابخانه یا سیستم‌عامل و برنامه‌هایی است که از آن تقاضای سرویس می‌کنند.

رابط کارکردهایی را تعریف می‌کند که کتابخانه یا سیستم‌عامل می‌تواند ارائه دهد و مفهومی مجرد است. این کارکردها سپس در قالب یک نرم‌افزار یا کتابخانه پیاده‌سازی می‌شوند. به عبارت ساده‌تر، رابط برنامه‌نویسی مجموعه توابعی است که یک برنامه می‌تواند از یک برنامه دیگر فرا بخواند.

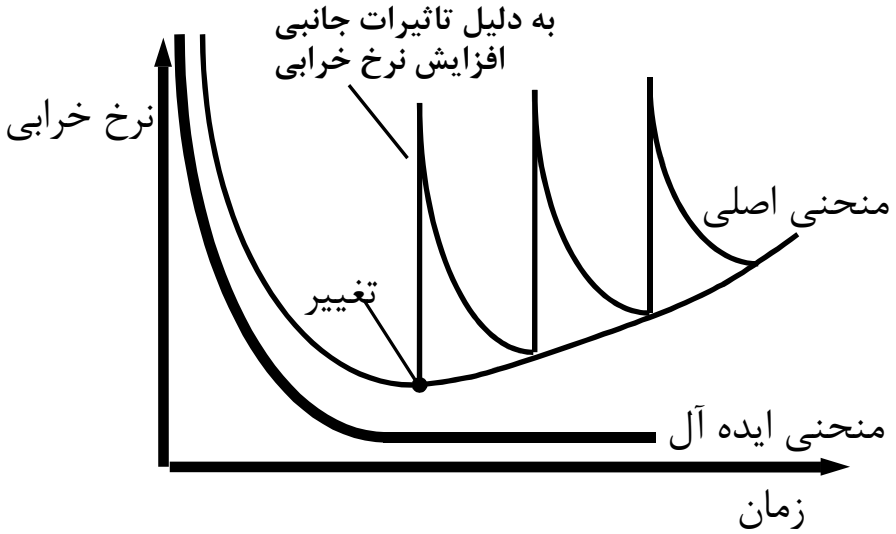
برای مثال مایکروسافت برای APIهای ویندوز مرجع‌هایی استاندارد دارد که با استفاده از آنها برنامه‌نویسان می‌توانند از قابلیت‌ها و سرویس‌های سیستم‌عامل در توسعه و نوشتن برنامه‌های کاربردی خود استفاده کنند.

همان‌طور که می بینیم از جهت حقوقی، نرم افزارهای واسط، نمی توانند مورد حمایت مضاعف واقع شده و به مناسبت تولید هر نرم افزار، مورد حمایت قرار گیرند.

مقایسه نرم افزار و سخت افزار



«نمودار نرخ خرابی سخت افزار»



«نمودار نرخ خرابی نرم افزار»

نمودار خرابی در سخت افزار شامل سه مرحله: ۱. مرگ زودرس ۲. عمرمفید ۳. فرسودگی است.
نمودار خرابی در نرم افزار شامل سه مرحله: ۱. Test/Debug ۲. عمرمفید ۳. از رده خارج شدن است.
نرم افزار، توسعه و طراحی می شود اما ساخته نمی شود (چون فیزیکی نیست).
نرم افزار کهنه و فرسوده نمی شود اما زوال یافته و منسوخ می کردد.

نرم افزار – رده بندی جدید

- نرم افزارهای فراگیر (شبکه های بیسیم)
- نرم افزارهای Net sourcing (وب به عنوان موتور محاسباتی)
- نرم افزارهای متن باز (کد نرم افزار به طور باز در اختیار استفاده کنندگان قرار دارد)

- همچنین نرم افزارهای :
  - داده کاوی
  - محاسبات توری
  - ماشین های شناختی
  - نرم افزارهایی برای نانوتکنولوژی

کار در کلاس ۱: در مورد هر یک از انواع نرم افزارهای بالا تحقیق کنید (برای هر مورد حداقل یک صفحه A4)

سیستم های موروثی Legacy Systems

- نرم افزار باید با نیازهای جدید محیط پردازشی و تکنولوژی انطباق یابد.
- نرم افزار برای پیاده سازی نیازهای جدید کسب و کار باید بهبود یابد.
- نرم افزار باید برای امکان ایجاد ارتباط با دیگر سیستم های نرم افزار مدرن یا پایگاه های داده گسترش یابد.
- نرم افزار باید برای عملکرد مناسب در محیط شبکه معماری مجدد گردد.

سیستم هایی است که در حال حاضر مورد استفاده قرار می گیرند باید با سیستم جدید سازمان هماهنگ شود. یک استراتژی معماری گرا برای برخورد با سیستم های موروثی، بیشتر روی اینترفیس ها یا واسط ها متمرکز خواهد شد و اجازه خواهد داد که سیستم های جدید و قدیم در یک روش مقرون به صرفه از نظر اقتصادی تعامل داشته باشند. اگر واسط کاربر سیستم قدیمی خیلی ساده و روان نیست می توان تصمیم گرفت که آن را تغییر داد و یا کلا" آنرا جایگزین نمود.

بحران نرم افزار

در فاصله سال های ۱۹۶۰-۱۹۷۰ تولید و فروش کامپیوتر های خانگی به دلیل تقاضای کاربران افزایش یافت. بنابراین نیاز به برنامه های مختلف کامپیوتری زیاد بود این امر باعث تولید غیرقانونی نرم افزار و در نتیجه نارضایتی مشتریان شد که در حقیقت بحران نرم افزار نامیده می شود.

بحران نرم افزار اولین بار به صورت رسمی در کنفرانسی در سال ۱۹۶۸ میلادی در کنفرانس «مهندسی نرم افزار ناتو» توسط اف.ال.بور، مدیر کنفرانس در گارمیش آلمان مطرح گردید . عوامل اصلی این بحران عبارتند از :

- هزینه بالای ایجاد نرم افزار
- تاخیر در تولید و تحویل نرم افزار
- نگهداری پرهزینه نرم افزار
- پیشرفت سریع سخت افزار
- کیفیت پایین نرم افزار
- افزایش پیچیدگی کاربردها

تکامل نرم افزار

- قانون تغییر مداوم (۱۹۷۴) : سیستم های الکترونیکی باید به طور دائم در حال تغییر باشند. در غیر این صورت رضایت از آنها روز به روز کاهش می یابد.
- قانون افزایش پیچیدگی (۱۹۷۴) : هرگاه سیستم های الکترونیکی تکامل یابند، پیچیدگی آنها افزایش می یابد مگر اینکه در جهت نگهداری یا کاهش این پیچیدگی کاری انجام شود.
- قانون خود تطبیقی (۱۹۷۴) : فرایند تکامل سیستم های الکترونیکی در جهت حرکت به سمت توزیع نرمال محصول و فرایند خود تطبیق است.
- قانون ثبات سازمانی (۱۹۸۰) : میانگین فعالیت مفید فراگیر درتکامل یک نرم افزار در طول حیات محصول ثابت است.
- قانون ثبات آشنایی (۱۹۸۰) : در حین تکامل یک سیستم الکترونیکی، تمام افرادی که با آن به نوعی در ارتباط هستند، به عنوان مثال توسعه دهندگان، پرسنل فروش، کاربران و ... باید برای دستیابی به تکامل مورد انتظار بر محتوا و رفتار آن تسلط داشته باشند.
- قانون رشد مداوم (۱۹۸۰) : محتوای کارکردی یک سیستم الکترونیکی باید به طور مداوم در حال افزایش باشد تا رضایت کاربران را فراهم نماید.
- قانون کاهش کیفیت (۱۹۹۶) : کیفیت یک سیستم الکترونیکی کاهش خواهد یافت در صورتی که به طور مداوم نگهداری و با تغییرات محیط عملیاتی انطباق داده شود.
- قانون سیستم بازخورد (۱۹۹۶) : فرایند تکامل یک سیستم الکترونیکی باید شامل یک سیستم بازخورد چند سطحی، چند عامله و چند حلقه ای بوده و باید با آن طوری برخود شود که بهبود در نرم افزار ایجاد شود.

تعریف مهندسی:

مهندسی عبارتست از تحلیل، طراحی ،ساخت، تأیید و مدیریت موجودیتهای فنی یا اجتماعی.

تعریف مهندسی نرم افزار:

تعریف ۱: 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 ۲ (IEEE):

بکارگیری یک روش سیستماتیک، منظم و کمیت پذیر برای توسعه، اجرا و پشتیبانی نرم افزار است. یعنی بکارگیری مهندسی و شیوه های آن در نرم افزار

تعریف ۳: فرایندی است که طراحی و پیاده سازی و نگهداری یک نرم افزار قابل اطمینان را در بر می گیرد و روش ها و راه حل هایی که با بکارگیری ابزارها و تکنیک های مختلف باعث تولید یک نرم افزار با کیفیت مطلوب و هزینه ی پایین می شود.

تصور نادرست = گاهی اوقات به اشتباه تصور می شود که مهندسی کامپیوتر همان برنامه نویسی است.

وظایف مهندس نرم افزار:

۱-طراحی نرم افزار یا مدیریت پروژه.
۲-پیاده سازی نرم افزار.
۳-پشتیبانی نرم افزار.

انتظارات از مهندسی نرم افزار:

۱- کاهش هزینه تولید نرم افزار
۲- افزایش کارائی
۳- قابلیت حمل
۴- کاهش هزینه نگهداری
۵- قابلیت اطمینان بالا
۶- تحویل به موقع

مهندسی نرم‌افزار: یک فناوری لایه ای

هدف از مهندسی نرم‌افزار: تضمین کیفیت + اقتصادی بودن تولید

هر گونه رهیافت مهندسی(از جمله مهندسی نرم‌افزار) باید به یک تعهد سازمانی در مورد کیفیت تکیه داشته باشد.

- فرآیند نرم‌افزاری یک چهارچوب کاری است که برای ایجاد یک نرم‌افزاردارای کیفیت مطلوب لازم است.
- روشهای مهندسی نرم‌افزار،چگونگی انجام کار را از نظر فنی برای ساخت نرم‌افزار مهیا می کنند. شامل فعالیتهای مدل سازی و دیگر فنون توصیفی است.
- ابزارهای مهندسی نرم‌افزار پشتیبانی خودکار یا نیمه خودکار را برای فرآیند و روشها مهیا می سازند. وقتی ابزارها منسجم می‌شوند بطوریکه اطلاعاتی که بوسیله یک ابزار حاصل می‌شود را می توان توسط ابزار دیگر مورد استفاده قرار دارد، آن وقت سیسیمی بوجود می آید که مهندسی نرم‌افزار به کمک کامپیوتر نامیده می‌شود.

CASE: Computer Aided Software Engineering

تعریف متدولوژی یا الگو شناسی

- یک متدولوژی مجموعه ای از روش ها و توصیه ها و قالب ها می باشد که به همراه راهبرد مشخص و طی مراحل مختلف در توسعه سیستم به کار گرفته می شود. یک متدلوژی دارای ابزار تعریف شده و مدل مفهومی می باشد و از یک گرامر مشخص استفاده می کند.
- برای مثال مدل شیء گرا و یا مدل ساخت یافته در توسعه نرم افزار دو متدلوژی توسعه نرم افزار هستند.
- متدولوژی ، مجموعه ای از روش ها ، فنون و ابزارهای تحلیل و طراحی سیستم است که در چارچوب یک انگاره مدل سازی مبتنی بر یک الگوی مفهومی برای سازماندهی روند توسعه سیستم ها به روشی نظام مند به کار بسته می شود.

مسائل مطرح در متدولوژی

- متدولوژی بایست پاسخگوی موارد ذیل باشد :

  چگونه پروژه بایست به مراحل فرعی تجزیه شود؟

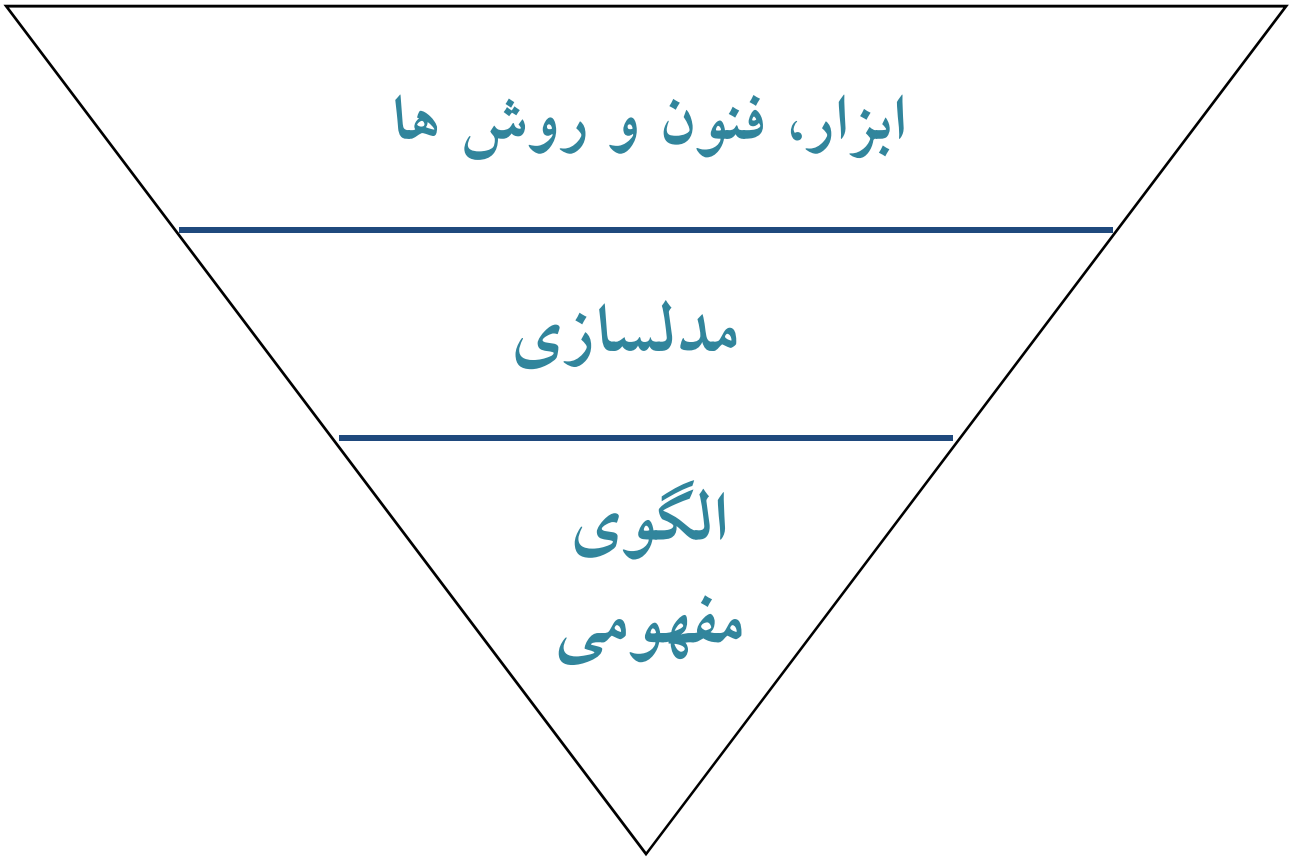
  در هر مرحله چه اقداماتی باید صورت گیرد؟

  چه خروجی هایی باید تولید شود؟

  در چه زمانی و تحت چه شرایطی باید این وظایف انجام شوند؟

  چه محدودیت هایی باید اعمال شود؟

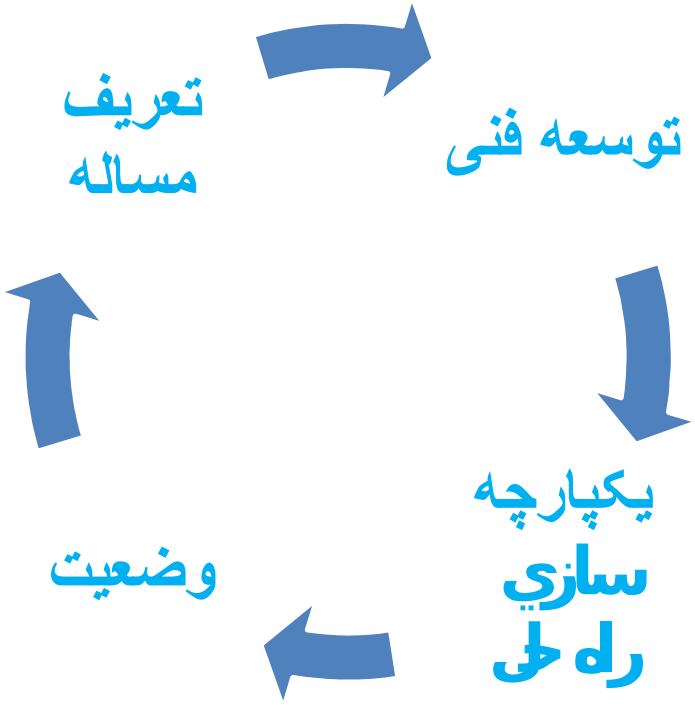
اجزای یک متدلوژی



مدل فرآیند

- فرآیند مهندسی نرم افزار مجموعه ای از قدم های قابل پیش بینی برای توسعه نرم افزار را مشخص می کند.
- مدل فرآیند نرم افزار ، قدم ها و استراتژی توسعه نرم افزار فرآیند و روش می باشد.
- برای مثال مدل آبشاری و مدل حلزونی دو مدل فرآیند توسعه نرم افزار می باشند.

فرایند حل مساله



چارچوب فرایند

فعالیت های چارچوب
فعالیت های کاری
محصولات کاری
مایل استون ها Milestone و موارد قابل تحویل
نقاط تضمین کیفیت

فعالیت های چتری

در گذشته به سنگهایی که در کنار جاده وجود داشتند و مسافت باقیمانده به شهرها را روی آن می نوشتند، Milestoneیا مسافت نما می گفتند. این کلمه در برنامه ریزی و کنترل پروژه نیز تقریبا به همین مفهوم است. بطور مثال فرض کنید تاریخ اتمام فاز طراحی در پروژه از اهمیت زیادی برخوردار است و شما می خواهید این زمان را در برنامه ریزی پروژه بطور واضح مشخص کنید و همچنین در گزارشات پروژه وضعیت آن را ارائه نمایید. بنابراین بهتر است این تاریخ را به عنوان یک مایلستون تعریف کرده و در پروژه نمایش دهید. هچنین در گزارشات مدیریتی، استفاده از مایلستونها می تواند به مدیران در مشاهده زمان تکمیل فعالیتهای مهم پروژه کمک زیادی بنماید. تعریف مایلستون در مباحث تئوریک مدیریت پروژه " فعالیتی با مدت زمان صفر است که زمان یک واقعه مهم در پروژه را نشان می دهد."

فرایند نرم افزار شامل مجموعه ای از فعالیت ها با چارچوب مشخص می‌باشد.

- **فعالیت های چارچوب (Framework Activities)**

  - ارتباطات (Communication)
  - طرح ریزی (Planning)
  - مدل سازی (Modeling)
    - آنالیز نیازها (Analysis of requirements)
    - طراحی (Design)
  - ساخت (Construction)
    - تولید کد (Code Generation)
    - تست (Testing)
  - استقرار (Deployment)

- **فعالیت های چتری (Umbrella Activities)**

  - ✓ مدیریت پروژه
  - ✓ بازنگری فنی و رسمی(Formal Technical Review , FTR)
  - ✓ تضمین کیفیت (Software Quality Assurance)
  - ✓ مدیریت پیکره بندی سیستم (Software Configuration Management)
  - ✓ آماده سازی و تولید خروجی ملموس (Work Product Preparation and Production)
  - ✓ مدیریت استفاده مجدد (Reusability Management)
  - ✓ اندازه گیری (Measurement)
  - ✓ مدیریت ریسک (Risk Management)

روند ها و جریان های حاکم بر حیطه مهندسی نرم افزار

اوائل دهه ۷۰ میلادی	برنامه نویسی ساختیافته • کد نویسی ساختیافته • برنامه نویسی Top-Down • پنهان سازی اطلاعات • انتزاع داده ها • تفکیک مرحله ای
میانه دهه ۷۰ میلادی	متدولوژی های طراحی • طراحی ساختیا • روش طراحی JSP • روش طراحی وارنیر –اور

اواخر دهه ۷۰ میلادی	متدولوژی های تحلیل • تحلیل ساختیافته • تحلیل ساختاریافته • SADT
دهه ۸۰ میلادی	تکنیک های خودکار، و طراحی و برنامه نویسی شیءگرا • PSA/PSL • تولید خودکار کد • ابزراهای مستقل مهندسی نرم افزار • Smalltalk , Ada ,Modula-2
دهه ۹۰ میلادی	مهندسی نرم افزار به کمک کامپیوتر (CASE) • ابزارهای محاوره ای برای تحلیلگران • محیط های تولید واسط کاربر • روش های شیء گرا • محیط های مجتمع مهندسی نرم افزار • ابزارهای توسعه سیستم به روش دیداری

مراحل تولید سیستم های کلاسیک

مرحله	فعالیت ها	مستندات تولید شده
۱. تحلیل نیازها	❑ارزیابی درخواست کاربر ❑هدایت مطالعات امکان سنجی ❑تعریف خواسته های کاربر ❑تهیه طرح کلی پروژه	• درخواست کاربر • پیشنهاد پروژه و برآورد هزینه • نتایج مطالعات امکان سنجی / تحلیل خواسته های کاربر • طرح کلی پروژه
۲. طراحی منطقی	❑تهیه طراحی عمومی ❑پالایش خواسته های کاربر	• توصیف کارکردی • سند خواسته های اطلاعاتی

مرحله	فعالیت ها	مستندات تولید شده
۳. طراحی فیزیکی	• تهیه طراحی تفضیلی • تعریف زیر سیستم ها • طراحی ساختار پایگاه داده	• تعیین مشخصات سیستم و زیر سیستم ها • تعیین مشخصات پایگاه داده ها • تعیین مشخصات برنامه
۴. طراحی برنامه	• کد کردن برنامه ها • آزمون واحدهای برنامه ای • مستند سازی برنامه ها	• مستندات برنامه ها

مرحله	فعالیت ها	مستندات تولید شده
۵. پیاده سازی سیستم	• اجرای آزمون زیر سیستم ها • اجرای آزمون سیستم • آموزش کاربران • استقرار کنترل های تبدیل • اجرای تبدیل داده ها	• طرح آزمون • گزارش تحلیل آزمون • راهنمای کاربران
۶. اجرای سیستم	• اجرای سیستم واقعی • نگهداری سیستم جدید • ارزیابی سیستم	• راهنمای اجرا • راهنمای نگهداری • گزارش ارزیابی

مراحل تولید سیستم های ساخت یافته

مرحله	فعالیت ها	مستندات تولید شده
۱. تحلیل نیازها	• ارزیابی درخواست کاربر • هدایت مطالعات امکان سنجی • تعریف خواسته های کاربر • تهیه طرح کلی پروژه	• درخواست کاربر • پیشنهاد پروژه و برآورد هزینه • نتایج مطالعات امکان سنجی / تحلیل خواسته های کاربر • طرح کلی پروژه
۲. طراحی منطقی (تحلیل)	• تبدیل خواسته های کاربر • پالایش ساختار منطقی سیستم	• تعیین طراحی منطقی ساختیافته

مرحله	فعالیت ها	مستندات تولید شده
۳. طراحی فیزیکی ساختیافته	• تشخیص سسله مراتب ماژول ها • تعیین واسط های بین ماژول ها • نهایی کردن اطلاعات پیکره بندی تجهیزات • تهیه آزمون طرح	• تعیین طراحی فیزیکی ساخت یافته • طرح آزمون • تعیین مشخصات برنامه
۴. پیاده سازی بالا به پایین	• کد کردن ماژول ها • یکپارچه سازی ماژول ها • مستند سازی برنامه ها	• مستندات یکپارچه نرم افزار

مرحله	فعالیت ها	مستندات تولید شده
۵. آزمون پذیرش	• تعریف سیستم قابل پذیرش • تولید موارد آزمون پذیرش	• مشخصات پذیرش • گزارش آزمون موارد
۶. تضمین کیفیت	• اجرای نهایی آزمون سیستم ها • آموزش کاربران • استقرار کنترل ها ی تبدیل • اجرای تبدیل داده ها	• گزارش آزمون پذیرش • راهنمای کاربران
۷. اجرای سیستم	• اجرای سیستم واقعی • نگهداری سیستم جدید • ارزیابی سیستم	• راهنمای اجرا • راهنمای نگهداری • گزارش ارزیابی

Lifecycle یا چرخه عمر تولید سنتی سیستم

مرحله	پرسش های اصلی	نتیجه
بررسی اولیه	مسئله چیست؟	1. تبین هدف ها 2. تعیین شاخص های عملکرد
مطالعه امکان سنجی	واقعیت ها چیست؟ خواسته های کاربر کدام است؟	1. تجزیه و تحلیل هزینه-منفعت 2. تبیین هدف های جدید
طراحی مفهومی (طراحی خام)	برای حل مسئله چه باید کرد؟	1. یافتن گزینه های جدید 2. تصویب یکی از گزینه ها
طراحی تفضیلی	مسئله چگونه باید حل شود؟	1. طراحی سیستم های نمونه 2. تنظیم برنامه آزمایش
استقرار	عملیات واقعی چیست؟ آیا دستورالعمل کاربر آماده است؟	1. مستند سازی مناسب برای ارائه به کاربر 2. تدوین برنامه کارآموزی
نگهداری	آیا سیستم باید اصلاح شود یا بهبود یابد؟	1. پاسخ به خواسته های کاربر 2. تامین رضایت کاربر

معایب روش های کلاسیک

- پروژه گرایی و نبود برنامه کلی در سطح سازمان
  - در دراز مدت ، سیستم ها ناهماهنگ و مجزا از یکدیگر خواهند بود!
- طولانی بودن زمان تحویل
- درگیری اندک کاربر
- پرخطا بودن سیستم ها
- انعطاف ناپذیر بودن سیستم ها

انواع سیستم های اطلاعاتی

- سیستم های مبتنی بر تبدیل
  - با ساختار درونی معطوف به تبدیل ورودی به خروجی مانند سیستم های اطلاعاتی مدیریتی

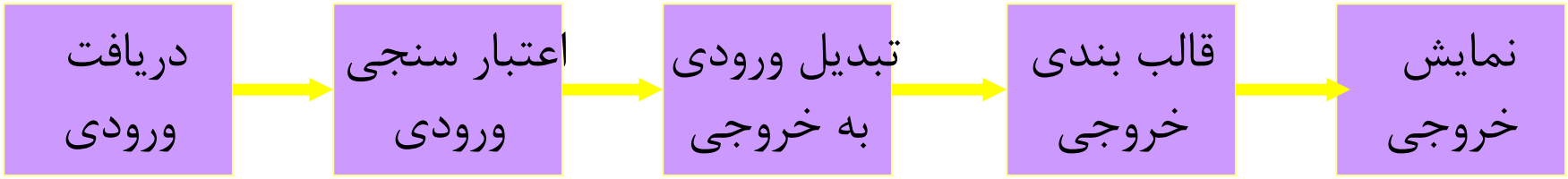
- سیستم های مبتنی بر تراکنش
  - با کارکرد مبتنی بر تشخیص و تفکیک تراکنش های مختلف مانند سیستم های بانکداری

کار در کلاس ۲:

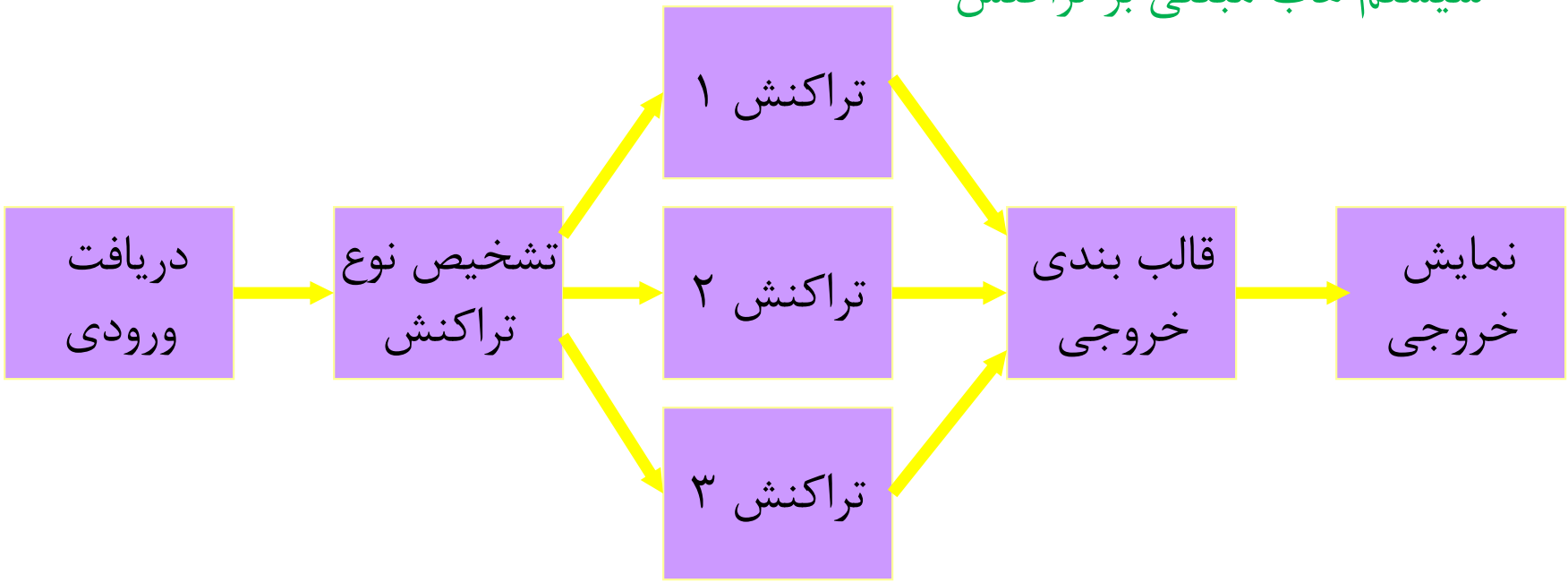
در مورد هر یک از انواع سیستم های اطلاعاتی تحقیق کنید (برای هر مورد حداقل یک پاراگراف توضیح بنویسید)

مقایسه انواع سیستم های اطلاعاتی

- سیستم های مبتنی بر تبدیل



- سیستم هاب مبتنی بر تراکنش



مدل های تولید نرم افزار

مدل های فرایند نرم افزار

مدل فرایند نرم‌افزار (Software Process Model)

تعیین یک راهکار توسعه که شامل لایه‌های فرایند، روش‌ها، ابزار و فازها(سه فاز اصلی: تعریف ، توسعه و نگهداری)است، الگوی مهندسی نرم‌افزار یا مدل‌فرایند نرم‌افزار نامیده می‌شود.

مدل‌های فرایند نرم‌افزار عبارتند از :

مدل ترتیبی خطی، مدل نمونه‌سازی (Prototyping Model)، مدل RAD، مدل افزایشی (Incremental Model)، مدل حلزونی(Spiral Model)،

مدل حلزونی برنده – برنده (Win-Win)، مدل توسعه همزمان(Concurrent Development Model)

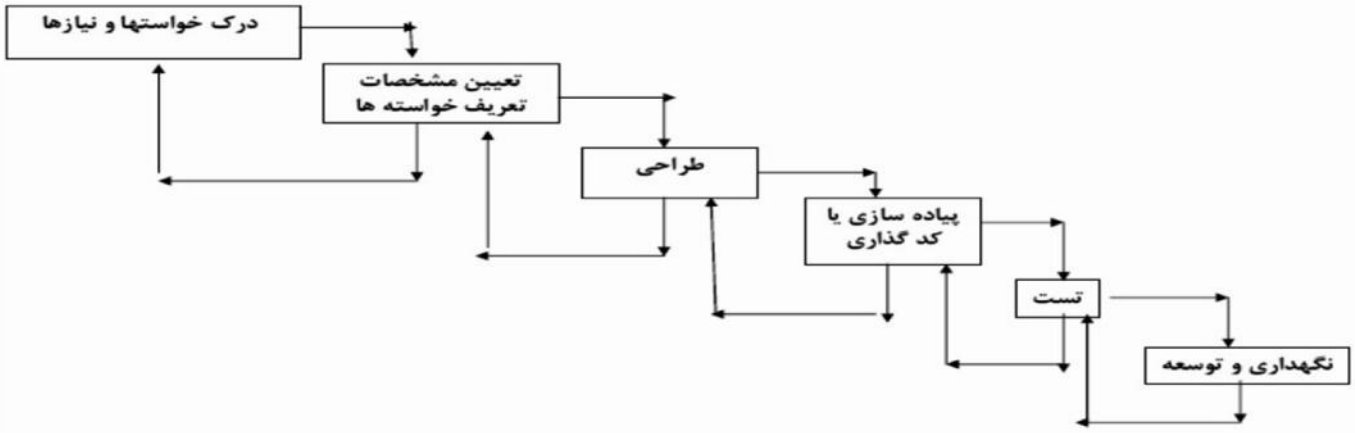
مدل توسعه مبتنی بر مولفه (Component Base Development)، مدل روش‌های رسمی(Formal Method)، تکنیکهای نسل چهارم و .....

مدل اولیه: اولین مدل غیر معروف، مدل کدنویسی واصلاح(Code & Fix Model) است که ابتداء کدنویسی وسپس اگر عملکرد درست نداشت، اصلاح انجام‌می‌شود. آیا شما تا کنون از این مدل، استفاده کرده‌اید؟!!

مدل ترتیبی خطی(The Linear Sequential Model)

تحلیل نیازها ⟵ طراحی نرم افزار ⟵ کد و برنامه نویسی ⟵ تست ⟵ تحویل به مشتری

مدل آبشاری



مدل آبشاری

تحلیل نیازمندیها

فرد تحلیلگر باید

۱- به زمینه کار نرم‌افزار اشراف داشته باشد
۲- اعمالی که بصورت دستی و قبل از کامپیوتری شدن انجام می‌شود را بداند
۳- نکاتی که باعث افزایش کارایی نرم‌افزارمی‌شود را بداند
۴- اطلاعاتی که باید در واسط کاربر(interface) نمایش داده شود را مشخص کند
۵- اهداف و نیازمندیهای کاربر را مشخص کند

طراحی

۱- مشخص کردن Data Structure
۲- دسته بندی اعمال مورد نیاز
۳- طراحی واسط های کاربر
۴- مشخص کردن جزئیات الگوریتم
۵- مشخص کردن محیط برنامه نویسی ،انتخاب سیستم عامل

ایجاد کد

طراحی باید به شکلی درآید که برای دستگاه قابل خواندن باشد. مرحله ایجاد کد این کار را انجام می دهد. اگر طراحی به صورت دقیق صورت گیرد،می‌توان بصورت مکانیکی تولید کد را انجام داد.

آزمون

وقتی کد ایجاد شد آزمون برنامه شروع می‌شود.کار آزمون روی فواصل زمانی منطقی در نرم‌افزار متمرکز می‌شود و تضمین می کند که تمام جملات آزمون شده،یعنی اجرای آزمون برای نشان دادن خطاها و تضمین اینکه داده‌های تعریف شده نتایج واقعی بوجود می‌آورند.

معایب مدل آبشاری

پروژه‌های واقعی به ندرت از این مدل استفاده می کنند.

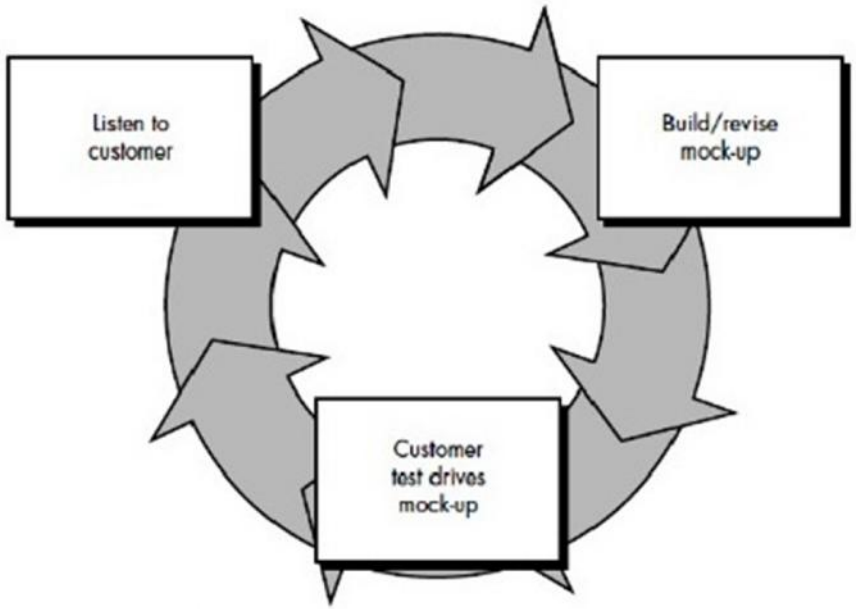
۱- تغییرات احتمالی در هریک از مراحل باعث آشفتگی دیگر مراحل می‌شود.

۲- معمولاً کاربران تمامی خواسته‌های خود را در ابتدای فرآیند تولید نرم‌افزار بیان نمی‌کنند، بنابراین ممکن است مرحله اول بطور ناقص انجام شود و در نهایت فرآیند تولید با مشکل مواجه شود.

۳- کاربر باید بسیار صبور باشد. کاربران تا رسیدن به مرحله آخر محصول را مشاهده نمی‌کنند. اگر در زمان تحویل کاربر به یکی از خواسته‌های قبلی خود اشاره کند که در نرم‌افزار تولید شده وجود ندارد، در اینصورت کل مدت زمان تولید نرم‌افزار ممکن است به هدر رود.

۴- بعضی از اعضاء تیم تولید ممکن است بیکار بمانند. چون هر مرحله وابسته به اتمام مرحله قبل است.

مدل نمونه‌سازی



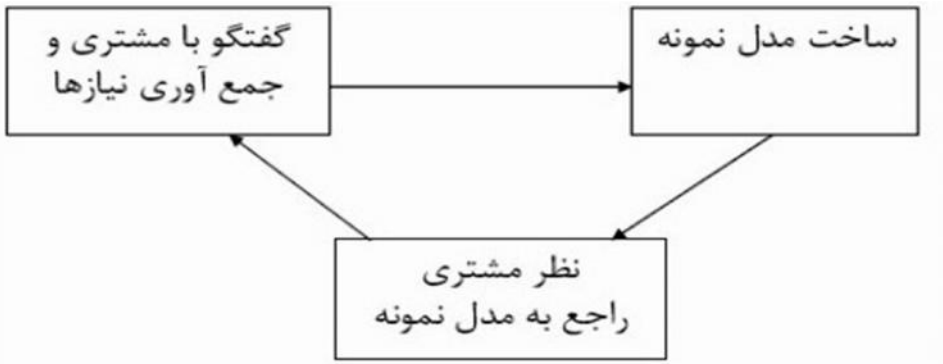
- اغلب کاربر خواسته‌ها و اهداف کلی خود را بیان می‌کند و جزئیات چگونگی دریافت اطلاعات، پردازش اطلاعات و گزارش‌گیری را مشخص نمی‌کند.
- گاهی اوقات تولید کننده از کارایی الگوریتم، انتخاب سیستم عامل، پایگاه داده و زبان پیاده‌سازی مطمئن نیست.

در اینگونه موارد مدل نمونه سازی بهترین روش است.

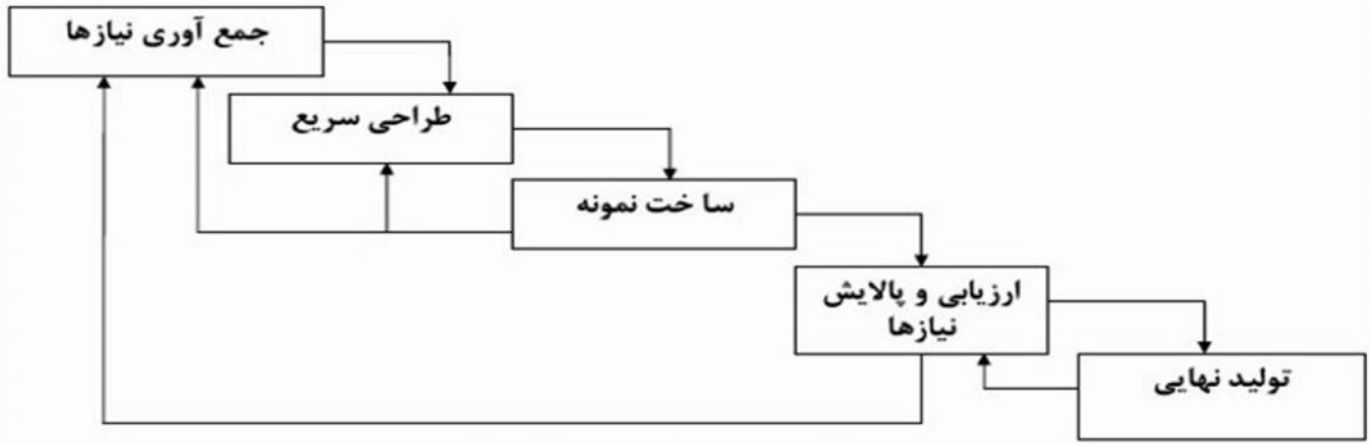
این مدل با جمع آوری نیازمندیها شروع می‌شود.مشتری و تولید کننده همدیگر را ملاقات نموده و اهداف کلی خود را در مورد نرم‌افزار بیان می کنند.هر چه مورد نیاز است را شناسایی می کنند. سپس طراحی سریع رخ می دهد. این طراحی روی نمایش آن جنبه‌هایی از نرم‌افزار متمرکز می‌شود که برای مشتری/کاربر مشهودند (مثل ساختار ورودی/خروجی).

نمونه نخست توسط مشتری/کاربر ارزیابی شده و از آن برای رفع نیازهای نرم‌افزاری که قرار است تولید شود استفاده می‌شود.

این چرخه زمانی خاتمه می یابد که رضایت مشتری جلب شود و خواسته دیگری نداشته باشد.



مدل نمونه‌سازی



مدل نمونه‌سازی در کنار سایر فازها

اشکالات مدل نمونه سازی

۱- کاربر فقط آخرین خواسته‌های خود را در نمونه مشاهده می‌کند، در صورتیکه تولید کننده برای تضمین کیفیت نرم‌افزار باید تمام قسمتهای قدیم و جدید را تست کند. گاهی اوقات تولید کننده مجبور می‌شود نمونه‌های قدیمی را کلاً کنار بگذارد و یک نمونه جدید تولید کند.

۲- گاهی اوقات تولید کننده جهت تسریع در ارائه نمونه اولیه در انتخاب زبان برنامه نویسی، سیستم عامل و یا پایگاه داده اشتباه می‌کند و تا زمان تولید نمونه نهایی متوجه اشتباه خود نمی‌شود.

مدل ساخت سریع برنامه‌ها (RAD : Rapid Application Development)

مدت زمان تولید نرم‌افزار در این مدل کوتاه‌تر از روش های قبلی است.

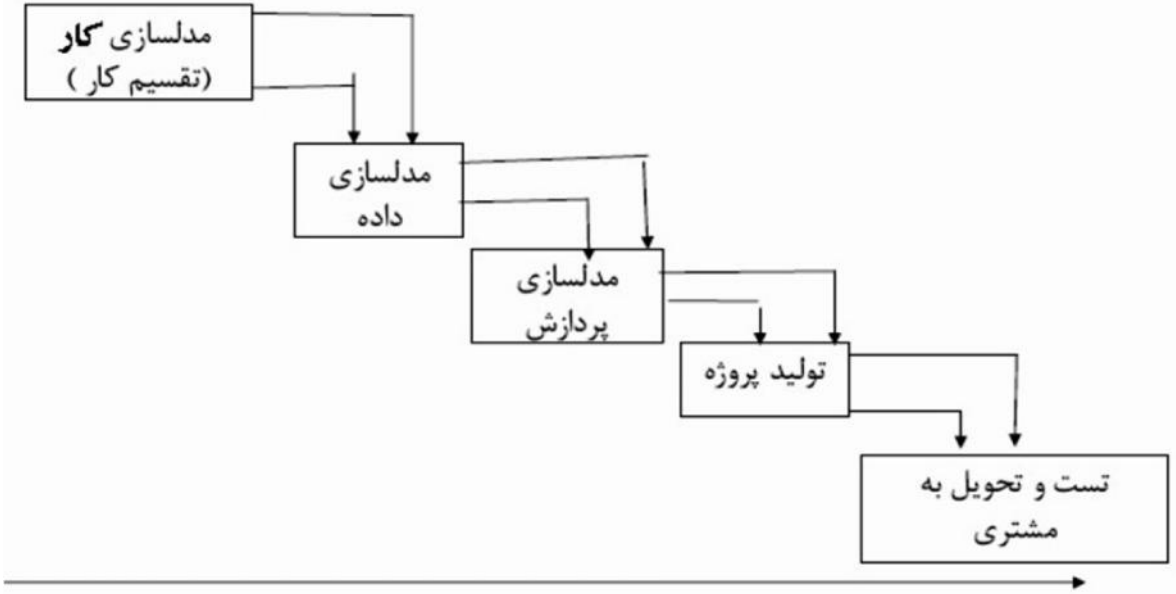
این مدل زمانی مفید خواهد بود که خواسته‌های کاربر:

۱- کاملاً مشخص و محدود باشد
۲- از پیچیدگی زیادی برخوردار نباشد
۳- قابل دسته بندی باشد

دراینصورت هر دسته از خواسته‌های کاربر را یک تیم کاری انجام می‌دهد.

اگر تیم‌های کاری بصورت موازی اقدام به تولید نرم‌افزار کنند در نهایت در مدت زمان کمتری نرم‌افزار تولید می‌شود.

مدت زمان تولید نرم‌افزار به روش RAD حدود چند ماه است.(۲ یا ۳ ماه)



مدل تولید سریع نرم‌افزار

مدل سازی تجاری

جریان اطلاعات در میان کارهای تجاری به شکلی مدل سازی می‌شود که به سوالات زیر پاسخ دهد:

۱- چه اطلاعاتی برای شروع به کار سیستم مورد نیاز است؟
۲- چه اطلاعاتی توسط سیستم تولید می‌شود؟
۳- چه کسی(بخشی) اطلاعات را تولید می‌کند؟
۴- اطلاعات تولید شده به کجا می‌رود؟
۵- چه کسی(بخشی) اطلاعات را پردازش می‌کند؟

مدل سازی داده ای

از روی جریان اطلاعات می توان مجموعه‌ای از اشیاء را تعریف نمودکه هر یک دارای یکسری مشخصه‌اند و بوسیله مشخصه‌ها شناسایی می‌شوند. بعد از تعریف اشیاء ارتباط بین آنها را مشخص می‌کنیم.

مدل سازی فرآیند

به هریک از اشیاء مرحله قبلی اعمالی که در سیستم واقعی وجود دارد اختصاص می‌یابد. این اعمال نهایتاً باعث ایجاد/ویرایش/حذف/بازیابی اشیاء می‌شوند.

تولید برنامه کاربردی

استفاده از تکنیکهای نسل چهارم جهت تسریع زمان تولید نرم‌افزار.
در این مرحله عناصری که از قبل تولید شده‌اند مورد استفاده واقع می شوند. یا اینکه به گونه‌ای تولید می شوند که قابل استفاده مجدد باشند.

آزمون و گردش

چون این مدل از عناصر از پیش آماده استفاده می کند که قبلاً تست شده اند بنابراین مدت زمان کمتری برای تست مورد نیاز است.

اشکالات مدل RAD

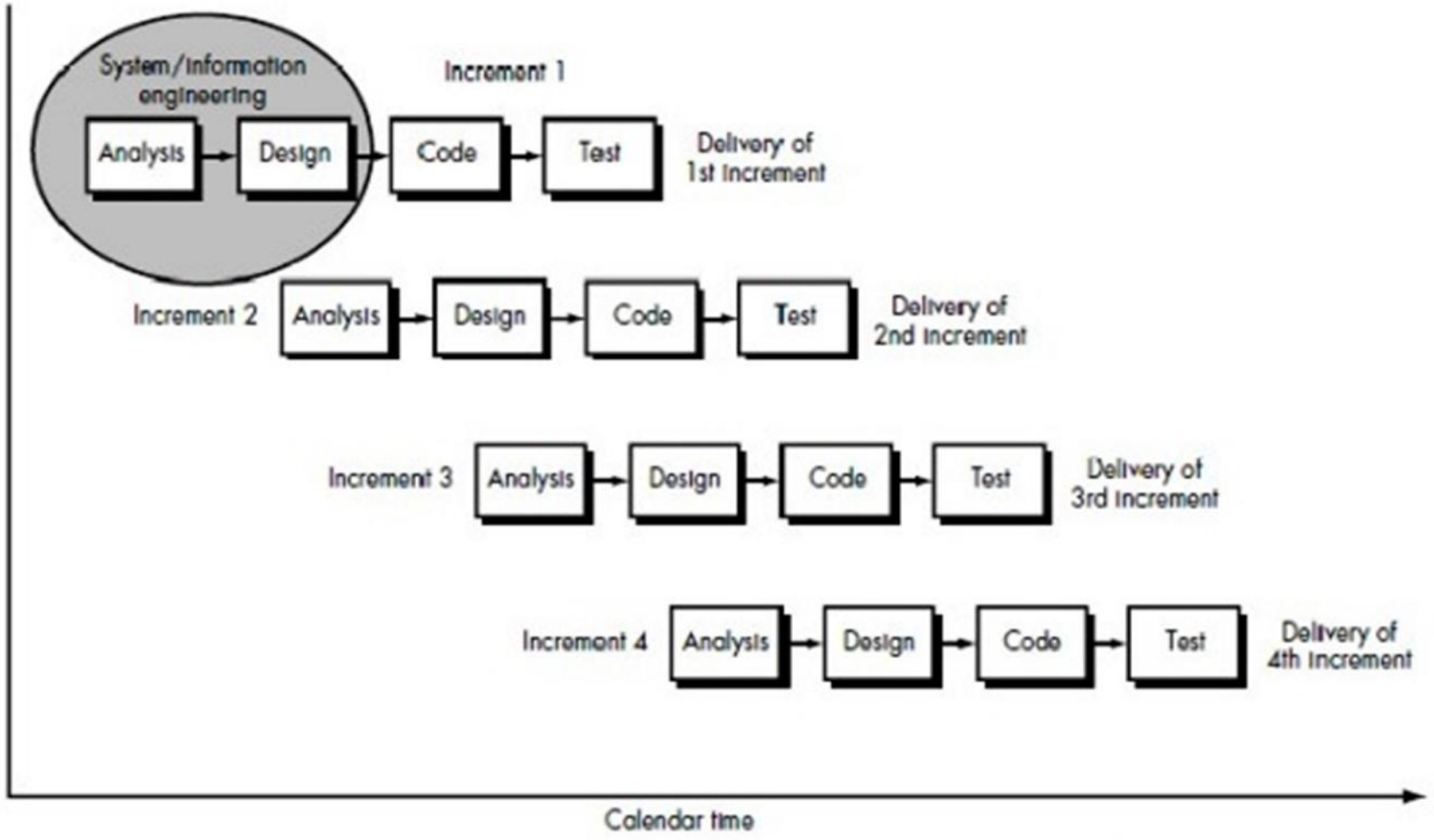
۱- برای پروژه‌های بزرگ به تیمهای کاری بیشتری نیاز است. بنابراین به منابع انسانی بیشتری نیاز خواهد بود.

۲- مدل RAD برای تولید همه نرم‌افزارها مناسب نیست. RAD برای سیستم‌های ماژولار مناسب است. اگر بتوان یک برنامه کاربردی را به شکلی پیمانه‌بندی کرد که هر تابع اصلی را در کمتر از ۳ ماه کامل کند، آن وقت برای RAD کاندید می‌شود. هر تابع اصلی را می توان با یک تیم جداگانه از RAD مورد بررسی قرار داد و سپس آن را یکپارچه نمود.

۳- RAD نیازمند تولید کننده و مشتریهایی است که بتوانند فعالیتهای سریع لازم را برای رسیدن به یک سیستم کامل در یک چهارچوب زمانی کوتاه انجام دهند. اگر این تعهد دارای هیچگونه قانونی نباشد، پروژه شکست می خورد.

۴- RAD وقتی که مخاطرات تنیکی زیاد است مناسب نیست. وقتی که برنامه نیاز به استفاده زیاد از تکنولوژی جدید دارد یا وقتی که نرم‌افزار جدید به درجه زیادی از همکاری با دیگر برنامه‌های کاربردی موجود، نیاز دارد.

مدل افزایشی (Incremental Model)



این مدل ترکیبی از مدلهای آبشاری و نمونه‌سازی است.

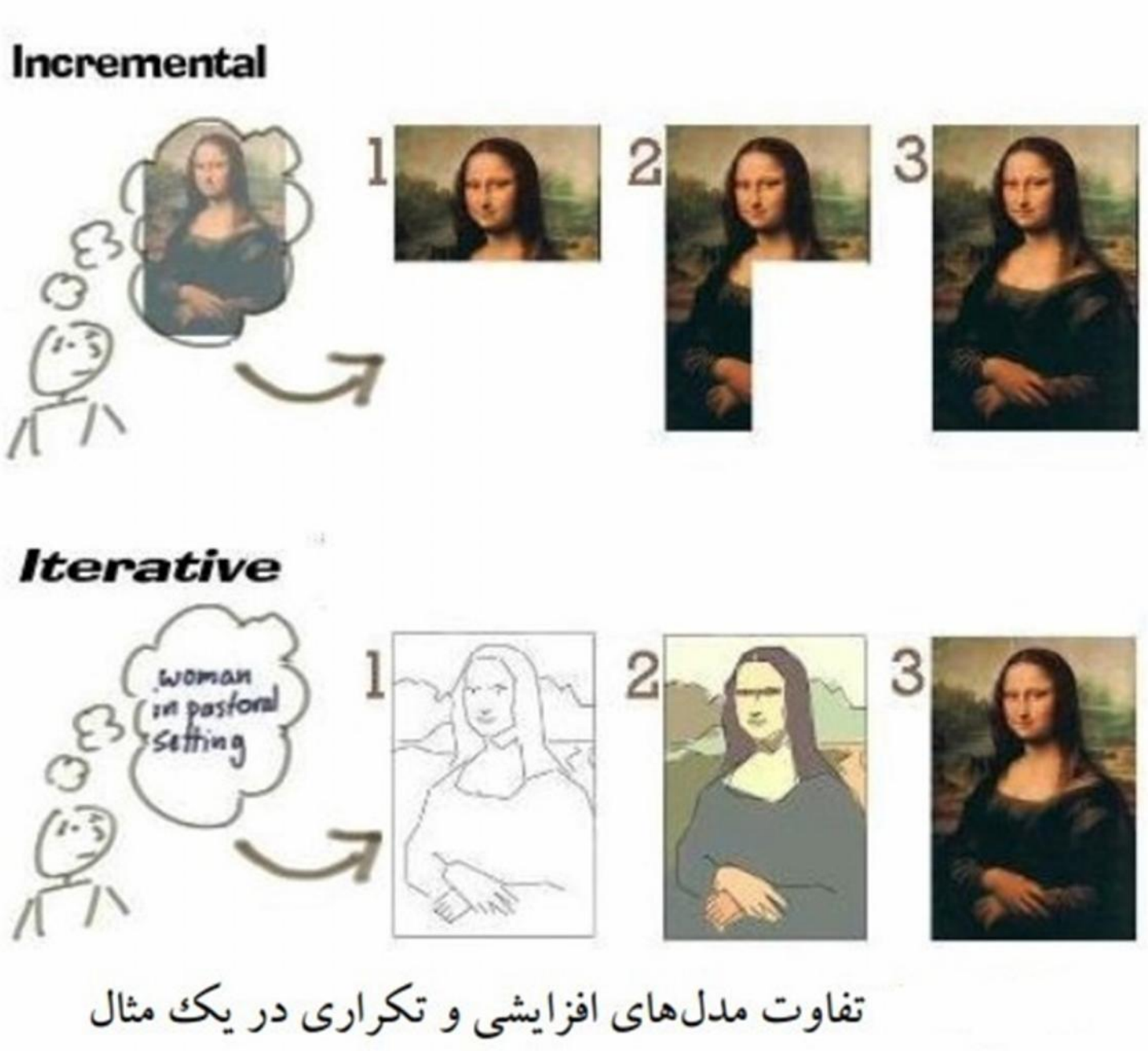
در مدل آبشاری پس از اتمام چرخه تولید، نرم افزار در اختیار کاربر قرار می گیرد. در مدل نمونه سازی مشتری نمونه‌هایی از نرم افزار را بررسی می کند و در نهایت نمونه آخر را استفاده می کند.

در مدل افزایشی نرم افزار در چند مرحله تولید و به مشتری تحویل داده می‌شود. (نرم افزار تولیدی در هر مرحله مورد استفاده کاربر قرار می گیرد.) به عنوان مثال در نرم افزار پردازش کلمه که با مدل افزایشی تولید شده است ممکن است در تحویل اول مدیریت فایل اولیه، ویرایش، و عملیات تولید سند و در دومین تحویل ویرایش و تولید اسناد پیچیده‌تر، تحویل سوم بررسی املا و گرامر و در تحویل چهارم صفحه قابلیت صفحه‌آرایی پیش رفته ارائه شود.

در این روش یا مدل بهتر است نیازهای اصلی مشتری در نمونه اول پوشش داده شود و سایر قسمتهای پروژه در نمونه‌های بعدی تکمیل شود.به نمونه اول اصطلاحاً محصول پایه یا core product گفته می‌شود. محصول پایه توسط مشتری استفاده یا ارزیابی می‌شود. در نتیجه آن یک برنامه برای تکرار بعدی نوشته می‌شود. در این برنامه تغییرات لازم در محصول پایه و ویژگیهای و قابلیتهای دیگری که باید به آن اضافه شود، ذکر می‌شود. این فرآیند تا زمانتولید محصول نهایی تکرار می‌شود.

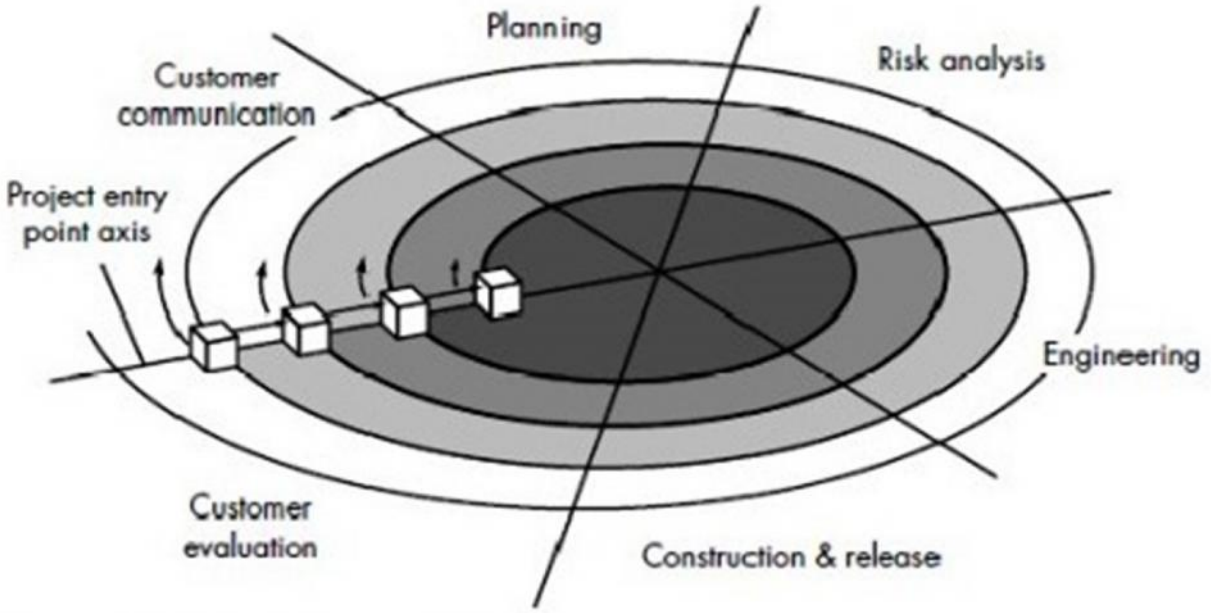
شباهت این مدل با مدل نمونه سازی در طبیعت تکراری آن است. اما برخلاف مدل نمونه‌سازی در این مدل در هر تکرار یک محصول عملیاتی به مشتری تحویل می‌شود.

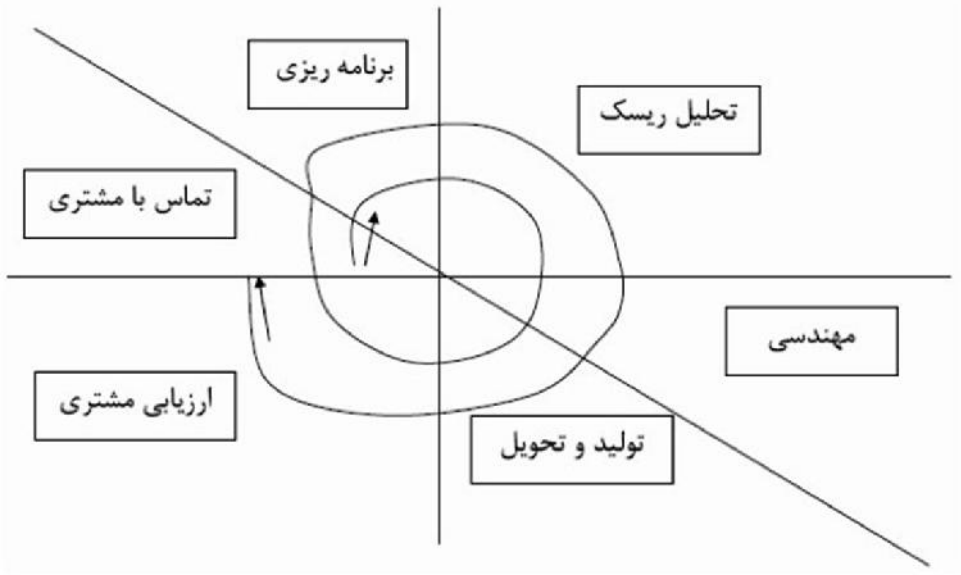
مدل افزایشی زمانی مناسب است که نیروی انسانی برای پیاده سازی کامل پروژه در دسترس نیست. نمونه‌های اولیه را می‌توان با نیروی انسانی کمتری پیاده سازی کرد. اگر از محصول پایه مورد قبول مشتری قرار گرفت می‌توان برای تکرارهای بعدی در صورت لزون نیروی انسانی اضافه کرد. همچنین تکرارها را می‌توان طوری برنامه ریزی کرد که ریسکهای فنی را مدیریت کنند. بعنوان مثال یک سیستم مهم ممکن است به در دسترس بودن یک قطعه سخت افزاری که در حال ساخت است نیاز داشته باشد، که تاریخ تحویل آن قطعی نیست. می‌توان تکرارهای اولیه را طوری برنامه ریزی کرد که به استفاده از این قطعه نیازی نداشته باشند.



تفاوت مدل‌های افزایشی و تکراری در یک مثال

مدل حلزونی یا چرخشی(Spiral Model)





مدل حلزونی

مدرس: سپیده برادران هزاوه

این مدل طبیعت تکراری مدل نمونه سازی و جنبه‌های کنترل شده و منظم مدل آبشاری را با هم ترکیب کرده است. این مدل بطور بالقوه قابلیت تولید نسخه‌های افزایشی از نرم افزار را دارد. با استفاده از مدل حلزونی ارائه نرم افزار در یک سری از نسخه‌های افزایشی امکان پذیر است. در تکرارهای اولیه محصول ممکن است یک مدل کاغذی یا نمونه باشد. در حین تکرارهای بعدی نسخه‌های کاملتری از سیستم مهندسی شده تولید می‌شود.

مدل حلزونی به چند دسته فعالیت که به آنها ناحیه وظایف نیز گفته می‌شود،تقسیم شده است.این نواحی ممکن است بین سه تا شش ناحیه باشد.

۱. **ارتباط با مشتری:**کارهای لازم جهت برقراری ارتباط کارآمد مابین تولید کننده و مشتری
۲. **برنامه ریزی:** کارهای لازم برای تعریف منابع، زمانبندی، و دیگر اطلاعات مربوط به پروژه
۳. **تحلیل مخاطرات:** کارهای لازم برای ارزیابی مخاطرات فنی و مدیریتی
۴. **مهندسی:** کارهای لازم برای ساخت یک یا چند نمایش از کاربرد
۵. **ساخت و تحویل:** کارهای لازم برای ساخت، آزمون، نصب و پشتیبانی (مثل مستندسازی و آموزش)
۶. **ارزیابی مشتری:** کارهای لازم برای دریافت بازخورد مشتری بر اساس ارزیابی نمایشهای نرم افزار که در حین مراحل مهندسی ساخته شده در مرحله نصب پیاده سازی شده اند.

به محض شروع فرایند، تیم مهندسی نرم افزار در جهت عقربه‌های ساعت حول حلزون می گردد. نتیجه اولین گردش ممکن است مشخصات محصول باشد و گردشهای بعدی ممکن است برای تولید نمونه و سپس بطور پیشرونده نسخه‌های پیچیده‌تری از نرم‌افزار باشد. نتیجه هر گذر از ناحیه برنامه ریزی تنظیم برنامه پروژه است. هزینه و زمانبندی بر اساس بازخوردهای دریافت شده از مرحله ارزیابی مشتری تنظیم می‌شوند. بعلاوه مدیر پروژه تعداد تکرارهای لازم برای تکمیل نرم افزار را تنظیم می کند.

برخلاف فرآیندهای کلاسیک که با تحویل نرم افزار پایان می یابند مدل حلزونی را می توان طوری ساخت که در طول زندگی نرم افزار کامپیوتری قابل اجرا باشد.

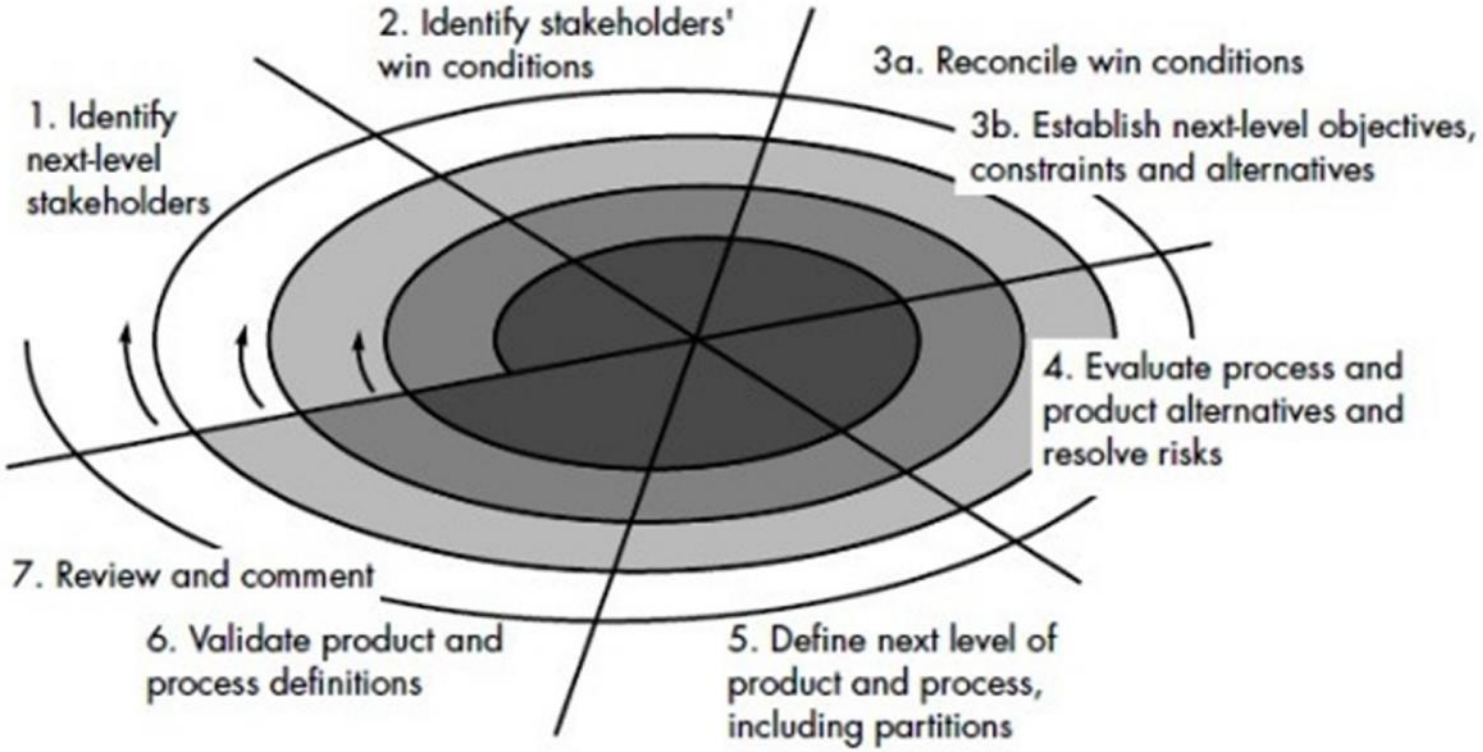
مدل حلزونی یک راهکار واقع‌بینانه برای تولید سیستمها و نرم افزار بزرگ است. چون نرم افزار در یک فرایند پیش رونده رشد می‌کند، تولید کننده و مشتری مخاطرات را بهتر شناخته و در برابر آنها واکنش نشان می دهند. این مدل از نمونه سازی بعنوان مکانیزم کاهش مخاطرات در هر مرحله استفاده می کند. یک راهکار مرحله ای منظم را که توسط چرخه حیات کلاسیک توصیه شده است را نگهداری می کند اما آنرا با یک چهارچوب تکرار شونده ترکیب کرده است. مدل حلزونی نیازمند در نظر داشتن مستقیم مخاطرات در همه مراحل پروژه است و اگر به خوبی مورد استفاده قرار گیرد، بایستی مخاطرات را قبل از اینکه منجر به مشکل شوند کاهش دهد.

اما همچون دیگر مدلها، مدل حلزونی نیز علاج نیست.

معایب

۱. قانع کردن مشتری در مورد اینکه این مدل قابل کنترل است ممکن است کار ساده ای نباشد
۲. نیازمند ارزیابی قابل توجه مخاطرات توسط افراد ماهر است و وابسته به مهارت آنها برای موفقیت است.
۳. اگر یک مخاطره مهم پوشش داده نشود و مدیریت نگردد، بدون شک مشکل به وقوع خواهد پیوست.
۴. این مدل به اندازه مدل آبشاری و نمونه سازی مورد استفاده قرار نگرفته است.

مدل حلزونی برد برد (WINWIN Spiral Model)



در مدل حلزونی فعالیتی به نام ارتباط با مشتری داشتیم.هدف از این فعالیت روشن کردن نیازمندیهای پروژه از طرف مشتری است. در حالت ایده آل انتظار می رود که تولید کننده نیازمندیهای مشتری را بپرسد و مشتری نیز جزئیات کافی را ارائه کند. اما در واقع به ندرت این اتفاق می افتد. در حقیقت تولید کننده و مشتری وارد مذاکره می شوند، که در آن مشتری تقاضای توازن میان عملیاتی بودن، کارایی و دیگر ویژگیهای محصول یا سیستم با هزینه و زمان تحویل آنرا دارد.

بهترین حالت مذاکره برد برد است. در آن مشتری برنده می شود با دریافت سیستم یا محصولی که نیازهای مهم وی را برآورده می کند و تولید کننده برنده می شود با کار واقعی و بودجه و زمان تحویل قابل دسترس.

مدل حلزونی برد برد مجموعه ای از مذاکرات که در ابتدای شروع هر گردش به دور حلزون باید انجام شود را تعریف می کند. علاوه بر ارتباط با مشتری، فعالیتهای زیر نیز تعریف شده اند:

۱. شناخت ذینفعان کلیدی سیستم یا زیر سیستم
۲. تعیین شرایط برد ذینفعان
۳. مذاکره بر سر شرایط برد ذینفعان برای قانع کردن آنها با مجموعه ای ازشرایط برد-برد برای همه (از جمله تیم پروژه نرم افزار)

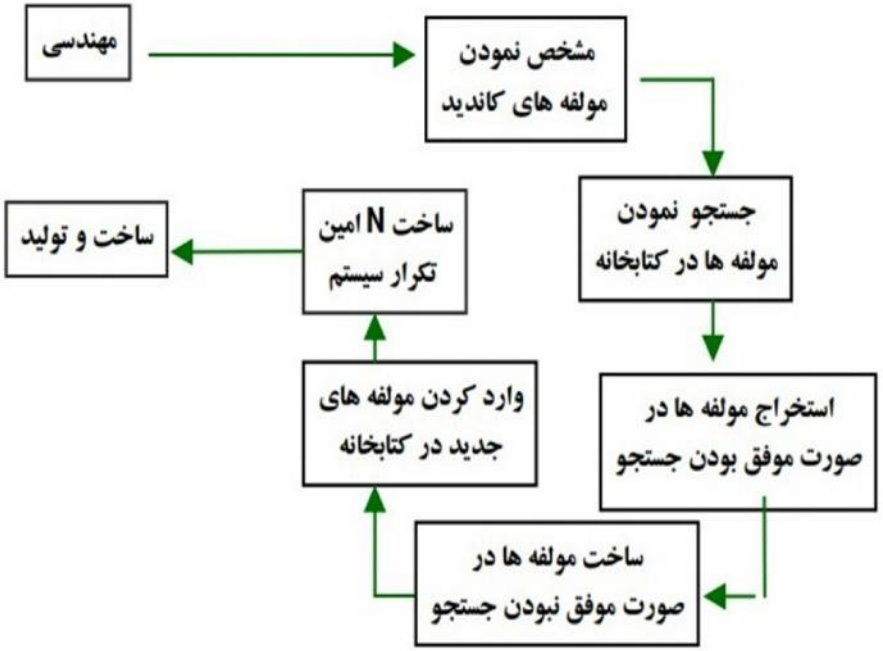
تکمیل موفقیت آمیز این مراحل اولیه منجر به نتایج برد-برد می شود.

مراحل مدل حلزونی برد–برد

۱. تعیین ذینفعان سطح بغد
۲. تعیین شرایط برد ذینفعان
۳. مرحله
   a. تطبیق شرایط برد
   b. تعیین اهداف، محدودیتها و راهکاری دیگر سطح بعد
۴. ارزیابی فرآیند و راهکارهای دیگر تولید و حل مخاطرات
۵. تعریف سطح بعدی تولید و فرآیند، از جمله قسمت بندیها
۶. درستی یابی محصول و تعاریف فرآیند
۷. بازبینی و نظرات

مدل توسعه مبتنی بر مولفه‌ها

(Component Base Development = CBD)



مدل CBD

سایر مدل‌ها: مدل توسعه همروند(Concurrent Development Model) مدل روش‌های رسمی یا فرمال (Formal Method Model)

تکنیک‌های نسل چهارم (4GT= Forth Generation Tool)، روش چابک (Agile Method)، مدل XP، مدل V، مدل فواره‌ای(Fountain) و مدل UP

کار در کلاس ۳:

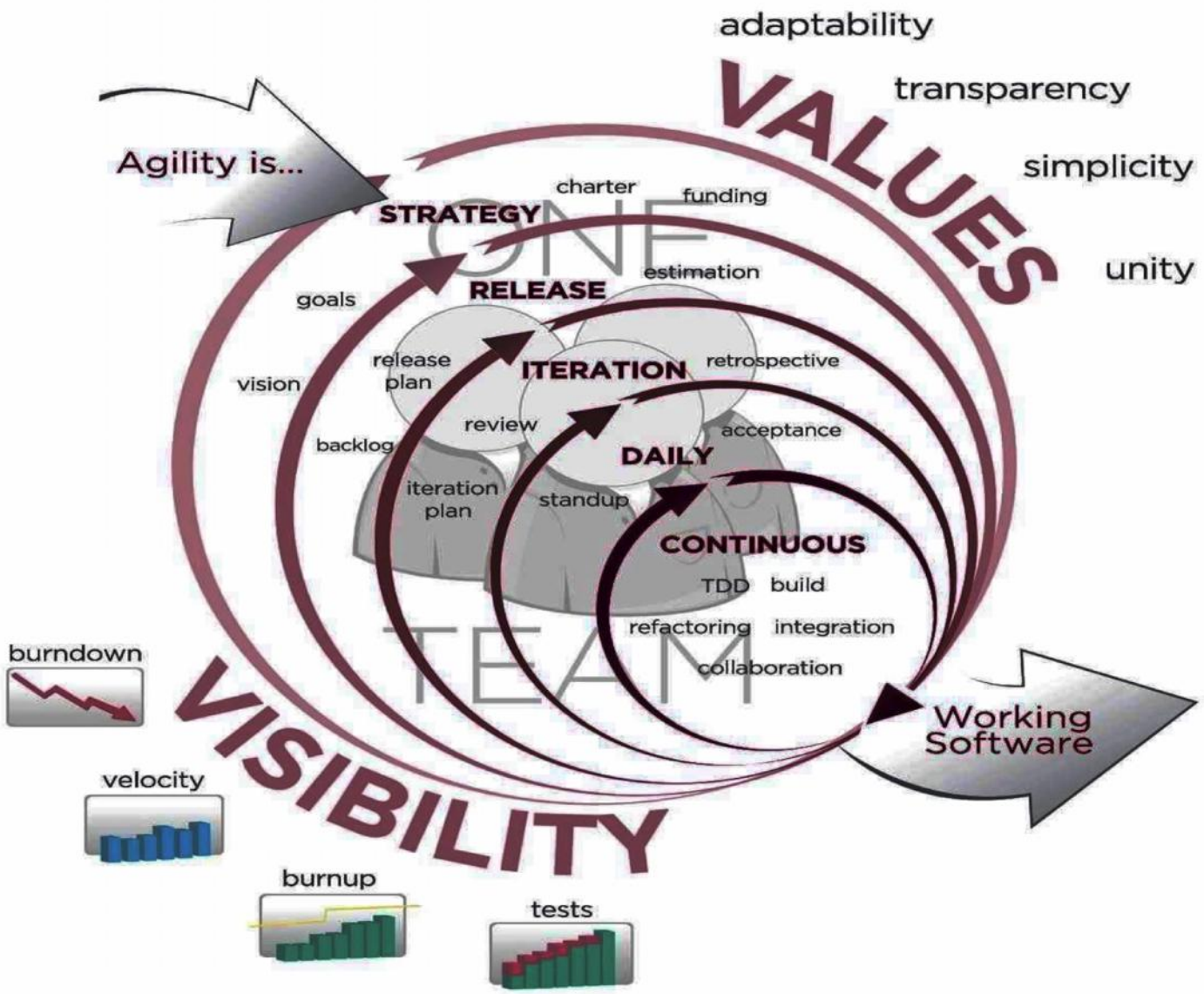
در مورد هر یک از ۸ مدل جدید که در بالا نام برده شده است، تحقیق کنید (برای هر مورد حداقل یک صفحه A4 توضیح بنویسید)

Agile Methodology

توسعه چابک نرم‌افزار یا توسعه نرم‌افزاری چابک گروهی از متدهای توسعه نرم‌افزار مبتنی بر تکرار و به شکل تدریجی است که در آنها، راه‌حل‌ها از طریق خودسازمان‌دهی و همکاری بین تیم‌های مختلف کاری، انجام می‌شوند. این روش برنامه‌ریزی تطبیقی، توسعه و تحویل تکاملی و رویکرد زمان بسته‌بندی تکرارشونده را ارتقا می‌بخشد و پاسخ‌های سریع و انعطاف‌پذیر برای انجام تغییرات را تقویت می‌کند. در واَقع چابک‌سازی یک چارچوب مفهومی است که پیش‌بینی تعاملات در سراسر چرخه توسعه را بهبود می‌بخشد.

در متدهای توسعه‌ی چابک مشخصات زیادی وجود دارند، که بیشترشان توسعه، کار تیمی، همکاری و سازگاری فرایند در چرخه‌ی حیات پروژه را ترفیع می‌دهند. متدهای چابک وظایف را به گام‌های کوچک با کمترین میزان برنامه‌ریزی می‌شکنند که به طور مستقیم با برنامه‌ریزی‌های طولانی‌مدت درگیر نیستند. تکرارها بسته‌های زمانی کوتاه‌مدتی هستند که معمولاً بین یک تا چهار هفته طول می‌کشند. هر تکرار دارای یک تیم متقابل عملکردی در تمام مأموریت‌ها است: تحلیل نیازمندی‌ها، طراحی، کدنویسی، واحد تست، و قبولی در تست. در پایان هر تکرار یک محصول کاری به ذی‌نفعان نشان داده می‌شود. این، ریسک کلی را به حداقل رسانده و اجازه می‌دهد پروژه خیلی سریع با تغییرات منطبق شود. ممکن است یک تکرار قابلیت کافی برای تضمین پخش در بازار را نیفزاید، اما هدف، انتشار در دسترس با حداقل شکاف در پایان هر تکرار است. شاید تکراهای چندگانه نیاز به انتشار یک محصول یا ویژگی‌های جدید داشته باشند

AGILE DEVELOPMENT



ACCELERATE DELIVERY

اصول تحلیل و طراحی سیستم ها در مهندسی نرم افزار