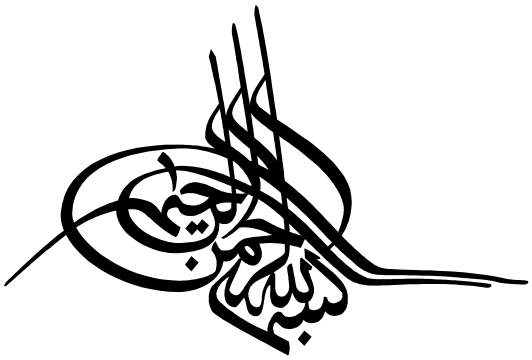


مهندسی نرم‌افزار

سری کتاب‌های کمک آموزشی کارشناسی ارشد

مجموعه مهندسی فناوری اطلاعات

مؤلف: آزاده ضیایی

سرشناسه	: ضیایی، آزاده، ۱۳۶۲
عنوان	: مهندسی نرم‌افزار
مشخصات نشر	: تهران : مشاوران صعود ماهان، ۱۴۰۲
مشخصات ظاهری	: ۱۵۳ ص
فروست	: سری کتاب‌های کمک آموزشی کارشناسی ارشد
شابک	: ۹۷۸-۶۰۰-۴۵۸-۸۱۶-۴
وضعیت فهرست نویسی	: فیپای مختصر
یادداشت	:این مدرک در آدرس http://opac.nlai.ir قابل دسترسی است.
شماره کتابشناسی ملی	: ۳۷۶۸۱۲۷

نام کتاب:..... مهندسی نرم‌افزار

مولف:..... آزاده ضیایی

مدیر تولید محتوی: .....سمیه بیگی

ناشر: .....مشاوران صعود ماهان

نوبت و تاریخ چاپ: .....اول / ۱۴۰۲

تیراژ: ..... ۱۰۰۰ نسخه

قیمت:.....: ۲/۶۳۰/۰۰ ریال

شابک: ..... ISBN: ۹۷۸-۶۰۰-۴۵۸-۸۱۶-۴

انتشـــارات مشاوران صعود ماهان: خیابان ولیعصر، بالاتر از تقـاطع مطهری،

روبروی قنادی هتل بزرگ تهـــران، جنب بانک ملی، پلاک ۲۰۵۰

تلفن: ۴- ۸۸۱۰۰۱۱۳

سخن ناشر

«ن والقلم و ما یسطرون»

کلمه نزد خدا بود و خدا آن را با قلم بر ما نازل کرد.

به پاس تشکر از چنین موهبت الهی، موسسه ماهان درصدد برآمده است تا در راستای انتقال دانش و مفاهیم با کمک اساتید مجرب و مجموعه کتب آموزشی خود برای شما داوطلبان ادامه تحصیل در مقطع کارشناسی ارشد گام موثری بردارد. امید است تلاش‌های خدمتگزاران شما در این موسسه پایه‌گذار گام‌های بلند فردای شما باشد.

مجموعه کتاب‌های کمک آموزشی ماهان به‌منظور استفاده داوطلبان کنکور کارشناسی ارشد سراسری و آزاد تالیف شده‌اند. در این کتاب‌ها سعی کرده‌ایم با بهره‌گیری از تجربه اساتید بزرگ و کتب معتبر داوطلبان را از مطالعه کتاب‌های متعدد در هر درس بی‌نیاز کنیم.

دیگر تالیفات ماهان برای سایر دانشجویان به‌صورت ذیل می‌باشد.

- **مجموعه کتاب‌های ۸ آزمون:** شامل ۵ مرحله کنکور کارشناسی ارشد ۵ سال اخیر به همراه ۳ مرحله آزمون تالیفی ماهان همراه با پاسخ تشریحی می‌باشد که برای آشنایی با نمونه سوالات کنکور طراحی شده است. این مجموعه کتاب‌ها با توجه به تحلیل ۳ ساله اخیر کنکور و بودجه‌بندی مباحث در هریک از دروس، اطلاعات مناسبی جهت برنامه‌ریزی درسی در اختیار دانشجو قرار می‌دهد.
- **مجموعه کتاب‌های کوچک:** شامل کلیه نکات کاربردی در گرایش‌های مختلف کنکور کارشناسی ارشد می‌باشد که برای دانشجویان جهت جمع‌بندی مباحث در ۲ ماهه آخر قبل از کنکور مفید می‌باشد.

بدین‌وسیله از مجموعه اساتید، مولفان و همکاران محترم خانواده بزرگ ماهان که در تولید و به‌روزرسانی تالیفات ماهان نقش موثری داشته‌اند، صمیمانه تقدیر و تشکر می‌نماییم.

دانشجویان عزیز و اساتید محترم می‌توانند هرگونه انتقاد و پیشنهاد درخصوص تالیفات ماهان را از طریق سایت ماهان به آدرس mahan.ac.ir با ما در میان بگذارند.

موسسه آموزش عالی آزاد ماهان

سخن مؤلف

مهندسی نرم‌افزار یکی از دروس مهم در رشته مهندسی کامپیوتر گرایش نرم‌افزار است. هدف اصلی مهندسی نرم‌افزار تولید یک نرم‌افزار با کیفیت در بازه زمانی مشخص است و اساس مهندسی نرم‌افزار بهترین تجربه است. پس می‌توان گفت مهندسی نرم‌افزار تنها علمی است که ابتدا ساخته شده سپس بر اساس آن کتاب نوشته‌اند.

هدف از تدریس این درس چگونگی ساخت یک نرم‌افزار با کیفیت است. در این کتاب تلاش شده است تا تمام مباحث پر اهمیتی که در درس مهندسی نرم‌افزار بر اساس سرفصل وزارت علوم در دوره کارشناسی مطرح بوده جمع‌آوری شوند و منبع اصلی این جمع‌آوری کتاب پرسمن است. همچنین در هر فصل تست‌های مرتبط به این درس در کنکورهای سراسری یا آزاد و سوالات دانشگاه پیام نور آورده شده است. امید است که این کتاب برای دانشجویان و داوطلبان گرامی مفید واقع شود.

آزاده ضیایی

فهرست

عنوان | صفحه

فصل اول

فرآیند مهندسی نرم‌افزار و محصول آن

- تعریف نرم‌افزار
- طبقه‌بندی نرم‌افزار
- ویژگی‌های نرم‌افزار
- دوره‌های تکامل نرم‌افزار
- معیارهای ارزیابی نرم‌افزار
- فرآیند نرم‌افزار تا مهندسی نرم‌افزار
- مراحل مهندسی نرم‌افزار
- مدل‌های فرآیند نرم‌افزار
- تیم نرم‌افزاری

فرآیند مهندسی نرم‌افزار و محصول آن

تعریف نرم‌افزار

برای نرم‌افزار تعاریف مختلفی وجود دارد که بعضی از این تعاریف عبارتند از:
۱. محصول فرآیند مهندسی نرم‌افزار است که توسط یک مهندس نرم‌افزار طراحی و ایجاد می‌شود.
۲. به برنامه‌هایی اطلاق می‌شود که کنترل و هماهنگی بین کارهای سخت‌افزاری و پردازش داده‌ها را بر عهده دارند.
۳. واژه نرم‌افزار به هر گونه برنامه کامپیوتری اطلاق می‌شود که سرویسی را برای استفاده‌کننده فراهم می‌نماید.
۴. ساختمان داده‌هایی که برنامه‌ها را قادر می‌سازند تا بر روی اطلاعات عملیات انجام دهند.
۵. مستنداتی هستند که عمل برنامه و چگونگی استفاده از آن را شرح می‌دهند.

طبقه‌بندی نرم‌افزارها

از جمله عوامل مهم تعیین‌کننده ماهیت کاربرد نرم‌افزار، محتوای اطلاعاتی و قطعیت اطلاعاتی[1] (Information Determinacy) است. محتوا به معنا و شکل اطلاعات ورودی خروجی دلالت دارد و قطعیت اطلاعات بر قابلیت پیش‌بینی ترتیب و زمان استفاده از آن اطلاعات مربوط است.
به‌طورکلی می‌توان نرم‌افزارها را به انواع زیر تقسیم کرد:

نرم‌افزارهای سیستمی

مجموعه‌ای از برنامه‌ها است که برای سرویس‌دهی به برنامه‌های دیگر طراحی و ایجاد می‌گردد. نمونه‌ای از نرم‌افزارهای سیستمی که دارای قطعیت اطلاعات هستند، عبارتند از: کامپایلر و ویرایشگرها و برنامه‌های مدیریت فایل.
نرم‌افزارهای سیستمی که دارای قطعیت اطلاعات نیستند، سیستم عامل و پردازنده‌های ارتباط راه دور هستند که دارای مشخصات زیرند:
۱- کاربران بسیار زیاد
۲- اشتراک منابع و مدیریت فرآیند
۳- ساختمان داده‌های پیچیده
۴- تعامل زیاد با سخت‌افزارها
۵- زمان‌بندی کارها
۶- واسط‌های خارجی چندگانه

۱- تشخیص‌پذیری اطلاعات

نرم‌افزارهای بلادرنگ (Real- time software)

نرم‌افزارهایی هستند که برای نظارت و تحلیل و کنترل رویدادهای محیط به محض وقوع این رویدادها طراحی و پیاده‌سازی شده‌اند که این نرم‌افزارها باید در کم‌ترین زمان ممکن به رویدادها پاسخ دهند.

۴ عنصر اصلی این نرم‌افزارها عبارتند از:

۱- عنصر جمع‌آوری داده‌ها برای قالب‌بندی اطلاعات از محیط خارجی.

۲- عنصر تحلیل‌کننده که در مواقعی که نرم‌افزار نیاز دارد اطلاعات را انتقال دهد.

۳- عنصر کنترل‌کننده خروجی که به محیط خارجی طی یک مدت زمان مشخص پاسخ موردنیاز را می‌دهد.

۴- عنصر نظارت که وظیفه هماهنگی بین سایر اعضای دیگر را دارد که بتوانند با همکاری در بازه زمانی مشخص پاسخ مورد نظر را تولید کند[۱].

همان‌طورکه در وظایف این سیستم‌ها گفته شد، این سیستم‌ها باید طی یک مدت زمان مشخص نتیجه مورد نظر را تولید کنند (از ۱ میلی ثانیه تا ۱ دقیقه) در غیراین‌صورت نتایج فاجعه‌باری اتفاق خواهد افتاد در صورتی‌که در سیستم‌های دیگر می‌تواند زمان پاسخ تغییر پیدا کند بدون آن‌که نتایج فاجعه‌باری به‌وجود آید.

از نمونه این سیستم‌ها می‌توان به سیستم کنترل اطلاعات هواپیماهای درحال پرواز نام برد.

✓تست: نرم‌افزارهایی که نظارت، تحلیل و کنترل رویدادهای جهان واقعی را بر عهده دارد کدام است؟

(پیام نور،درس مهندسی نرم‌افزار ۸۸-۸۷)

۱) نرم‌افزارهای هوش مصنوعی
۲) نرم‌افزار سیستمی
۳) نرم‌افزاز جاسازی شده
۴) نرم‌افزار Real- time

☑پاسخ: گزینه «۳»

نرم‌افزارهای تجاری (Business Software)

این نرم‌افزارها روی اطلاعات کار خاصی در جهت اهداف معینی کار می‌کنند.

نوع تکامل‌یافته این نرم‌افزار را سیستم‌های اطلاعات مدیریت یا MIS (Management Information System) گویند که مجموعه نرم‌افزارهای مرتبط به هم و یکپارچه شده (Integrated) مربوط به یک سازمان است که یکپارچگی خصوصیت اصلی MIS است. نوع دیگری از نرم‌افزارهای تجاری، طرح‌ریزی منابع سازمانی یا ERP (Enterprise Resource Planning) است که مجموعه‌ای از نرم‌افزارهای یکپارچه است و از این جهت با MIS شباهت دارد و تفاوت آن با MIS در این است که هر نرم‌افزاری که برای سازمان طراحی می‌شود مطابق با جریان کاری سازمان است؛ درحالی‌که این نرم‌افزارها سفارشی هستند و برای تهیه آن‌ها از الگوهای سازمانی بهینه و توسعه‌یافته استفاده می‌شود و می‌توان گفت ERPها قابلیت تغییر بسیار زیادی ندارند و بسیار محدودکننده هستند یعنی یک سازمان نمی‌تواند بعد از خرید این نوع نرم‌افزار تغییرات زیادی در آن اعمال نماید زیرا که این تغییرات برای آن‌ها گران‌قیمت است و مقرون به صرفه نیست.

ERPها برای سازمان‌هایی مناسب است که جریان کاری در آن‌ها راه‌اندازی نشده است و یا این نوع نرم‌افزار تجاری منطبق بر جریان کاری آن‌ها است و برای سازمان‌هایی که بسیار قدیمی هستند و روال‌کاری در آنها استاندارد نیست؛ از این نوع نرم‌افزار تجاری نمی‌توان استفاده نمود.

بیشترین کاربرد این نرم‌افزارها در پردازش کارهای تجاری است، برای مثال سیستم‌هایی از قبیل حقوق و دستمزد، سیستم‌های موجودی انبار، حضور و غیاب از این‌گونه هستند.

نرم‌افزارهای مهندسی و علمی (Engineering & Scientific Software)

این نرم‌افزارها به‌وسیله الگوریتم‌هایی که اعداد و ارقام را پردازش می‌کنند به‌وجود آمده‌اند و کاربردهای بسیاری در نجوم،

۱- قابل تذکر است که در بعضی از منابع سه عنصر برای نرم‌افزارهای بلادرنگ در نظر گرفته می‌شود. عنصر اول، عنصر جمع‌آوری داده‌ها است که اطلاعات از محیط خارجی جمع‌آوری و قالب‌بندی می‌نماید. عنصر دوم، عنصر کنترل/خروجی است که به محیط خارجی پاسخ می‌دهد و عنصر سوم، عنصر مدیریت است که بقیه مولفه‌ها را طوری زمان‌بندی می‌نماید تا بتوانند پاسخ بلادرنگ ایجاد نمایند.

زمین‌شناسی، مکانیزاسیون صنعتی و غیره دارند.

نکته: نرم‌افزارهای شبیه‌سازی در این گروه قرار می‌گیرند. به کمک نرم‌افزارهای شبیه‌سازی می‌توان فرآیندهای طبیعی را شبیه‌سازی نمود.

نرم‌افزارهای جاسازی شده (Embedded Software)

امروزه محصولات هوشمند بسیاری یافت می‌شوند که تمامی این محصولات از نرم‌افزارهای جاسازی شده در حافظه فقط خواندنی (ROM) خود استفاده می‌کنند. بیشتر این نرم‌افزارها مقاصد کنترلی و صنعتی دارند و حتی در بعضی از لوازم منزل نیز استفاده می‌شوند. نرم‌افزارهای کنترل آسانسور، کنترل چرخ‌ها، کنترل صفحه کلید یک مایکروفر نمونه‌هایی از نرم‌افزارهای جاسازی شده هستند.

نرم‌افزارهای کامپیوترهای شخصی (Personal Computer Software)

نام دیگر این نوع نرم‌افزار، نرم‌افزارهای desktop است. از جمله این نرم‌افزارها می‌توان نرم‌افزارهای گرافیک کامپیوتری، سرگرمی، واژه‌پردازی و مدیریت پایگاه داده‌ها را نام برد.

نرم‌افزارهای مبتنی بر وب (Web based)

می‌توان گفت هر صفحه وب نرم‌افزاری است که دستورات برنامه‌نویسی مثل زبان HTML و جاوا و داده‌هایی مانند تصاویر و فایل‌های صوتی را به هم مرتبط می‌سازد. به‌طورکلی می‌توان گفت شبکه وب به یک ابرکامپیوتر تبدیل شده که یک منبع نرم‌افزاری نامحدود را به‌وجود می‌آورد.

نرم‌افزارهای هوش مصنوعی (Artificial Intelligence)

این نرم‌افزارها از روش‌های غیرعددی برای حل مسائل پیچیده‌ای که با الگوریتم‌های عددی قابل حل نیستند استفاده می‌کنند مانند سیستم‌های خبره (Expert Systems)، پایگاه دانش (Knowledge-Based Systems) و تشخیص صدا و بازی‌ها و غیره.

نرم‌افزارهای منطق فازی (Fuzzy Logic)

منطق فازی در مقابل منطق قطعی قرار دارد. در جهان واقعیات، آدمی بسیاری از مفاهیم را به‌صورت فازی (به‌معنای غیردقیق، ناواضح و مبهم) درک می‌نماید.

به‌عنوان نمونه، هر چند کلمات و مفاهیمی همچون گرم، سرد، بلند، کوتاه، پیر، جوان و نظائر اینها به عدد خاص و دقیقی اشاره ندارند، ذهن انسان با سرعت و با انعطاف‌پذیری شگفت‌آوری همه را می‌فهمد و در تصمیمات و نتیجه‌گیری‌های خود به حساب می‌گیرد. این در حالی است که ماشین فقط اعداد را می‌فهمد و اهل دقت است. اهداف شیوه‌های نو در علوم کامپیوتر آن است که اولاً رمز و راز این‌گونه توانایی‌ها را از انسان بیاموزد و سپس آنها را تا حد امکان به ماشین یاد بدهد. در نرم‌افزارهای منطق فازی از این منطق استفاده می‌شود. از کاربردهای این گروه نرم‌افزار می‌توان به موارد زیر اشاره نمود:

۱- هدایت و کنترل هرگونه دستگاه و تاسیسات پویا و حرکت‌ساز را می‌توان با کمک منطق فازی به بهترین وجه اعمال نمود، از جمله ماشین لباسشویی، قطارها، ترمز ای‌بی‌اس خودرو، آسانسور، جرثقیل، تسمه نقاله، موتورهای احتراقی، نشست و برخاست خودکار هواپیما و غیره.

۲- آینده‌نگری نرم‌افزارها جهت جلوگیری از هنگ کردن سرورها. کنترل موتورهای جستجوگر در اینترنت. سیستم‌های نرم‌افزاری ترجمه، رباتیک و هوش مصنوعی.

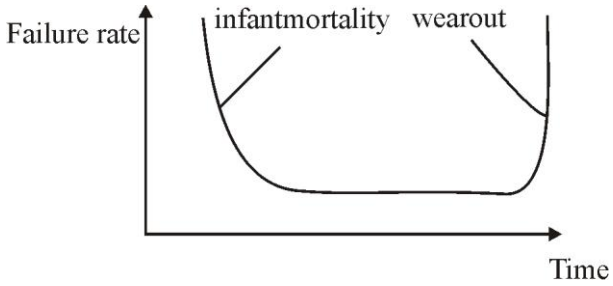
ویژگی‌های نرم‌افزار

تقریباً عموم ساخته‌های بشری حالت سخت‌افزاری و نمود فیزیکی دارند درحالی‌که نرم‌افزار یک سیستم منطقی است از این جهت نرم‌افزار دارای ویژگی‌هایی است که تفاوت زیادی با ویژگی‌های سخت‌افزاری دارد.

۱- مفهوم ساخت نرم‌افزار و سخت‌افزار تفاوت‌های بنیادی با هم دارند اگر چه شباهت‌هایی هم بین آن‌ها وجود دارد و باید گفت که هزینه‌های نرم‌افزار در مهندسی آن متمرکز است و پروژه‌های نرم‌افزاری را نمی‌توان مانند سایر پروژه‌های ساخت مدیریت نمود.

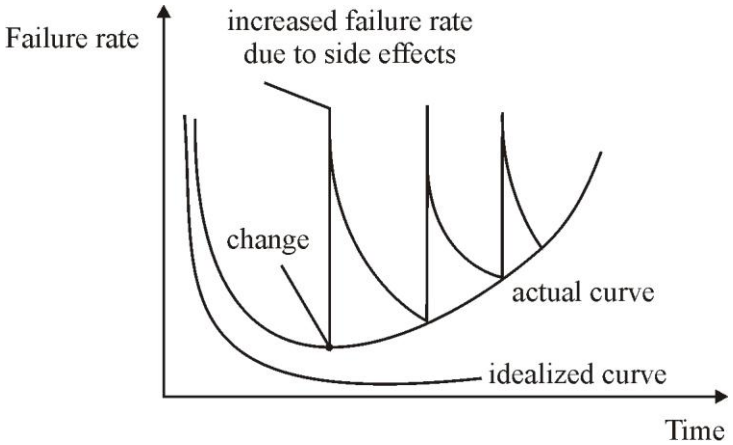
۲- اگر به سایر پروژه‌های سخت‌افزاری دقت کنیم می‌بینیم که در پروژه‌های سخت‌افزاری، قطعات فرسوده می‌شوند و نیاز به قطعات یدکی است ولی در پروژه‌های نرم‌افزاری این نکته بسیار مهم است که نرم‌افزار فرسوده نمی‌شود، بلکه نیازها تغییر می‌کند به همین دلیل نگهداری نرم‌افزار دربرگیرنده پیچیدگی بیشتری نسبت به سایر محصولات است. پس به‌طورکلی می‌توان گفت سخت‌افزار در تماس با محیط و با گذشت زمان دچار تخریب و فرسایش می‌شود درحالی‌که تغییرات باعث فرسایش نرم‌افزار می‌شود.

شکل ۱-۱ نمودار نرخ شکست محصول سخت‌افزاری را به‌صورت تابعی از زمان نشان می‌دهد. سخت‌افزار، در ابتدای عمر خود به‌دلیل عیوب طراحی و تولید نرخ شکست نسبتا شدیدی را از خود نشان می‌دهند. این عیوب تصحیح می‌شوند و نرخ شکست برای یک دوره زمانی ثابت می‌شود. با گذشت زمان سخت‌افزار دچار فرسایش می‌شود و دوباره نرخ شکست شدت می‌گیرد.



شکل ۱-۱. نمودار محصول سخت‌افزاری در گذر زمان

شکل ۱-۲ نمودار محصول نرم‌افزاری را در گذر زمان نشان می‌دهد.



شکل ۱-۲. نمودار محصول نرم‌افزاری در گذر زمان

همان‌طور که در شکل ۱-۲ مشاهده می‌شود به محض رخ دادن تغییر در نیازهای کاربران نرخ شکست افزایش پیدا می‌کند و با کاهش تغییرات، نرخ شکست کاهش می‌یابد و به منحنی واقعی مدل نرم‌افزار نزدیک‌تر می‌شود.

۳- یک قطعه نرم‌افزاری باید چنان طراحی شود که بتوان در برنامه‌های متفاوت از آن استفاده کرد، بیشتر نرم‌افزارها به جای اسمبل شدن از مولفه‌های موجود به‌صورت سفارشی ساخته می‌شوند درحالی‌که محصولات سخت‌افزاری عموماً توسط مولفه‌های موجود قابل تولید هستند.

✓**تست: کدام‌یک از عبارت‌های زیر در مورد نرم‌افزار صحیح می‌باشد؟** **(آزاد، فناوری اطلاعات ۸۹)**

۱) نرم‌افزار ساخته می‌شود.
۲) نرم‌افزار با گذشت زمان فرسوده می‌گردد.
۳) هزینه نرم‌افزار در مهندسی آن متمرکز است.
۴) نرم‌افزار یک عنصر سیستمی فیزیکی است نه منطقی

☑**پاسخ: گزینه «۳»**

دوره‌های تکامل نرم‌افزار

دوره اول – از دهه ۵۰ تا اواسط دهه ۶۰ میلادی: مشخصات دوره اول عبارت بودند از:

- رشد سریع سخت‌افزارها
- عدم نیاز به مدیریت نرم‌افزارها
- پردازش دسته‌ای و عدم وجود پردازش توزیع شده
- طراحی شدن نرم‌افزار توسط یک فرد یا سازمان و مدیریت آن توسط خود آن فرد یا سازمان.

دوره دوم– از اواسط دهه ۶۰ تا اواسط دهه ۷۰ میلادی: در این دوره بود که سیستم‌های چندبرنامه‌ای و چندکاربری به‌وجود آمدند. تکنیک‌های محاوره‌ای باعث شد تا کاربران جدید و سطوح مختلفی از پیچیدگی در نرم‌افزارها و سخت‌افزارها به‌وجود آید. یکی از مهم‌ترین مفاهیمی که در این دوره مطرح شد نگهداری نرم‌افزار (software maintenance) است، که شامل مجموعه فعالیت‌های زیر است:

۱- نرم‌افزارها در صورت بروز خطا اصلاح شوند.
۲- نرم‌افزارها متناسب با خواسته‌های جدید کاربران تصحیح شوند.
۳- نرم‌افزارها با سخت‌افزارهای جدید منطبق گردند.

نکته: در این دوره به‌دلیل مشکل افزایش غیرمنتظره هزینه و زمان نگهداری و توسعه نرم‌افزارها بحث بحران نرم‌افزار به‌وجود آمد.

دوره سوم: از اواسط دهه ۷۰ تا دهه ۸۰ میلادی: در این دوره سیستم‌های توزیع شده به‌وجود آمدند. منظور از این سیستم‌ها کامپیوترهای به هم متصلی بودند که توانایی این را داشتند تا عملیاتی را به‌صورت همزمان انجام دهند. در این دوره بود که کامپیوترهای شخصی به همراه ریزپردازنده‌ها ساخته شد و در تولید بسیاری از محصولات هوشمند از قبیل: ربات‌ها و اتومبیل‌ها به‌کار گرفته شدند. به‌وجود آمدن کامپیوترهای شخصی در این دوره باعث شد شرکت‌های نرم‌افزاری بسیاری به‌دلیل نیازهای گوناگون نرم‌افزاری این کامپیوترها رشد نمایند.

نمودار رشد کامپیوترهای شخصی کم‌کم به‌صورت یکنواخت تبدیل شد ولی نمودار رشد نرم‌افزار همچنان به‌صورت صعودی باقی ماند.

دوره چهارم: از اواسط دهه ۸۰ تاکنون: در این دوره بود که سیستم‌های رومیزی قدرتمند، همچنین فناوری برنامه‌نویسی شی‌گرایی (OOP) و کامپیوترهای شبکه‌ای و معماری‌های موازی مطرح شدند. سیستم‌های خبره و نرم‌افزارهای هوش مصنوعی به‌تدریج از محیط‌های آزمایشگاهی خارج شده و به سمت استفاده واقعی حرکت کردند.

بحران نرم‌افزاری (Software Crisis)

با توجه به توسعه و به‌وجود آمدن سخت‌افزارهای گوناگون تقاضا برای نوشتن نرم‌افزارهایی شکل گرفت که به‌راحتی بتواند این سخت‌افزارها را مدیریت کند و همین امر باعث ایجاد مشکلاتی شد که به آن بحران نرم‌افزاری می‌گویند و دلایل آن عبارت هستند از:

۱- توانایی تولید نرم‌افزار از میزان تقاضا جهت تولید کمتر است.
۲- پیچیدگی و قدرت سخت‌افزارها باعث شده نوشتن نرم‌افزاری که به‌صورت کامل بتواند از پتانسیل سخت‌افزاری استفاده کند مشکل گردد.
۳- توانایی نگهداری از نرم‌افزارها به‌دلیل ضعف در طراحی و مهندسی تولید آنها دشوار گردید.
۴- وجود خطا و اشکالات متعدد در نرم‌افزارهای تولید شده باعث ایجاد سه مشکل زیر می‌شود.

الف) افزایش هزینه
ب) افزایش زمان تولید
ج) پایین آمدن کیفیت نرم‌افزار

معیارهای ارزیابی نرم‌افزار

به‌طورکلی عوامل موثر در کیفیت و ارزیابی نرم‌افزار را می‌توان به دو دسته عوامل خارجی و داخلی تقسیم نمود.

الف) عوامل خارجی: این عوامل توسط کاربر نرم‌افزار تشخیص داده شده و مورد ارزیابی قرار می‌گیرند و شامل موارد زیر هستند:

۱- صحت برنامه (Correctness)
۲- استحکام (Robustness)
۳- قابلیت توسعه و تغییر(Extendibility)
۴- قابلیت اجرا روی سکوهای مختلف (Portability)
۵- کارایی (Efficiency)
۶- قابلیت استفاده مجدد (Reusability)
۷- سازگاری (Compatibility)

ب) عوامل داخلی: این عوامل توسط متخصصین کامپیوتر به‌کار گرفته می‌شوند و می‌توان گفت ابزارهای رسیدن به هدف هستند.

نکته: سه مشکل اصلی که در تولید و توسعه نرم‌افزار وجود دارد و یک مهندس نرم‌افزار در فرآیند تولید باید به آن‌ها توجه کند عبارت است از:

۱- ناهمگنی (Heterogeneity): راه‌حل این مشکل عبارت است از استفاده از تکنیک‌های مختلف در تولید نرم‌افزار به‌طوری که قابل اجرا روی سکوها و محیط‌های اجرایی مختلف باشد.
۲- اعتماد (reliability): راه‌حل این مشکل استفاده از تکنیک‌هایی است که نشان دهد نرم‌افزار ارائه شده می‌تواند از طرف کاربران آن مورد اعتماد قرارگیرد.
۳- تحویل (Delivery): راه‌حل این مشکل استفاده از تکنیک‌های مختلف برای سریع‌تر آماده کردن نرم‌افزار مورد نظر است.

با توجه به توضیحات فوق ۵ عامل اصلی که در تعیین کیفیت نرم‌افزارها مهم هستند، عبارت‌اند از:

۱- قابلیت پذیرش (Acceptability): نرم‌افزار باید به‌وسیله کاربری که آن را سفارش داده پذیرش و تأیید گردد یعنی باید قابل فهم مفید و سازگار با سایر سیستم‌ها باشد.

۲- درستی(Correctness): واضح است که یک نرم‌افزار باید درست کار کند در غیراین‌صورت ارزشی نخواهد داشت. اصلی‌ترین معیار برای اندازه‌گیری درست بودن یک نرم‌افزار تعداد خطاها و نقص به‌ازای هر هزار خطا است. منظور از خطا و نقص عدم تطبیق نتایج برنامه نوشته شده با خواسته‌های کاربران است.

۳- قابلیت نگهداری: منظور از نگهداری، سهولت در اصلاح و تغییر و پشتیبانی برنامه است که این تغییر و پشتیبانی در صورت بروز خطا و یا تطابق با تغییرات محیط یا بهبود برنامه با توجه به تغییر خواسته‌های مشتری انجام می‌گیرد.

نکته: برای اندازه‌گیری قابلیت نگهداری نرم‌افزار راه مستقیمی وجود ندارد، به همین دلیل از موازین غیرمستقیم استفاده می‌شود. یک معیار مبتنی بر زمان، میانگین زمان لازم برای تغییر MTTC است.

۴- یکپارچگی: منظور از یکپارچگی سیستم توانایی و تحمل سیستم در برابر حملاتی است که به‌صورت عمدی یا غیرعمدی، متوجه آن می‌شود. این حملات روی هر سه مولفه نرم‌افزار (برنامه‌ها، داده‌ها، مستندات) صورت می‌پذیرد. اندازه یکپارچگی معمولاً براساس دو صفت دیگر با عناوین تهدید و امنیت انجام می‌شود.

تهدید: احتمال وقوع حمله‌ای خاص در زمان معین که آن را با $P_i(T)$ نشان می‌دهند.

امنیت: به‌معنای احتمال دفع آن حمله است که آن را با $P_i(S)$ نشان می‌دهند.

طبق این دو تعریف یکپارچگی را با توسعه تهدید و امنیت روی انواع حملات می‌توان به‌صورت زیر تعریف کرد:

$$\text{یکپارچگی} = \sum_{i=1}^{n} (1 - P_i(T)) \times (1 - P_i(S))$$

که n تعداد انواع حملات است.

۵- قابلیت استفاده مجدد (Reusability): یکی از مهم‌ترین ویژگی‌های یک نرم‌افزار سهولت استفاده از آن است. در صورتی‌که استفاده از آن مشکل باشد، فروش نرم‌افزار شکست خواهد خورد. سهولت استفاده از نرم‌افزار براساس چهار خصوصیت زیر قابل‌اندازه‌گیری است:

۱- ارزیابی (نظرسنجی) موضوعی از احساس کاربران نسبت به سیستم

۲- زمان لازم برای بازدهی مناسب در به‌کارگیری سیستم

۳- مهارت‌های هوشی و فیزیکی برای یادگیری سیستم

۴- افزایش بهره‌وری سیستم نسبت به روشی که جایگزین می‌شود.

برای مثال وقتی سیستم یک کتابخانه را نرم‌افزاری می‌کنیم باید مراحلی که در امانت گرفتن یک کتاب توسط مشتری انجام می‌شود راحت‌تر و سریع‌تر از زمانی که تمامی کارها در محیط کتابخانه به‌صورت دستی بود، انجام گیرد.

فرآیند: منظور از یک فرآیند، نقشه مسیری است که به ساخت یک محصول و ایجاد نتایج در طی یک زمان معین است که به کیفیت بالا کمک می‌کند، به‌عبارت دیگر:

وقتی سیستمی را تولید می‌کنیم این امر مهم است که از یک سری مراحل قابل پیش‌بینی استفاده کنیم، نقشه مسیری که به ما کمک می‌کند تا نتایجی را در زمان معین و با کیفیت بالا تولید کنیم فرآیند نام دارد.

فرآیند نرم‌افزار یا مهندسی نرم‌افزار

عبارت است از یک رویکرد سیستماتیک، نظم‌یافته و قابل اندازه‌گیری برای طراحی و تولید نرم‌افزار و نگهداری آن. در واقع فرآیند نرم‌افزار یا مهندسی نرم‌افزار نظامی است که فرآیند، روش‌ها و ابزارها را به‌صورت یکپارچه با هم ترکیب می‌کند تا نرم‌افزاری فارغ از خطا در محدوده زمانی و بودجه پیش‌بینی شده به‌گونه‌ای که نیازهای کاربرانش را تامین کند تولید نماید.

✓تست: مهندسی نرم‌افزار: **(آزاد، کامپیوتر ۹۲)**

۱) یعنی به‌کارگیری عملی علوم کامپیوتر، مدیریت، اقتصاد و....در جهت طراحی و سخت سیستم‌های کلان

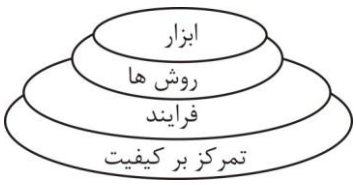
۲) به‌کارگیری علوم کامپیوتر در طراحی و ساخت سیستم‌های عامل

۳) یعنی برنامه‌ریزی (Micro Code) برای پردازنده اصلی کامپیوتر

۴) هیچ‌کدام از موارد بالا تعریف درستی از مهندسی نرم‌افزار نیست.

☑حل: گزینه «۴»

نکته: یکی از مهم‌ترین اهداف مهندسی نرم‌افزار تسهیل اصلاح نرم‌افزار است؛ به‌طوری‌که این اصلاح متناسب با نیازهای کاربران باشد.



شکل ۱-۳. ساختار لایه‌ای مهندسی نرم‌افزار

مهندسی نرم‌افزار یک ساختار لایه‌ای است که در شکل ۱-۳ نشان داده شده است. این لایه‌ها عبارتند از:

۱- لایه تمرکز بر کیفیت (کیفیت مداری): این لایه، پایین‌ترین لایه مهندسی نرم‌افزار است. به این معنی که لایه‌های دیگر بر آن استوار هستند. به‌عبارت دیگر در مهندسی نرم‌افزار روش‌ها باید بر کوشش سازمانی در جهت بهبود کیفیت محصول متکی باشند.

۲- لایه فرآیند: این لایه چارچوب است که تمامی فعالیت‌های مربوطه را به هم پیوند داده و شرایطی فراهم می‌آورد که باید برای تحویل موثر فناوری مهندسی نرم‌افزار وضع شوند.

این لایه به مانند یک واسط سایر لایه‌های ابزارها و روش‌ها را به هم پیوند داده و طراحی و ایجاد یک نرم‌افزار کامپیوتری را در یک زمان مشخص و در چارچوب کیفیتی مورد انتظار امکان‌پذیر می‌سازد.

۳- لایه روش‌ها: در این لایه روی چگونگی ساخت تکنیکی یک نرم‌افزار بحث می‌شود و در آن روش‌ها و دامنه وسیعی از وظایف و عملیات از جمله تحلیل نیازها، طراحی، ساخت برنامه، آزمون، چگونگی پشتیبانی و نگهداری را ارائه می‌دهد.

۴-لایه ابزارها: قبل از تعریف این لایه ابتدا باید با مفاهیم زیر آشنا شویم.

CASE (Computer Aided Software Engineering): ابزارهای اتوماتیک یا نیمه‌اتوماتیک را برای فرآیند یا متد فراهم می‌آورد.

CASEها نرم‌افزار، سخت‌افزار و یک پایگاه داده مهندسی نرم‌افزار را با هم ترکیب می‌کنند، تا یک محیط توسعه نرم‌افزار پدید آید. در تعریف کلی‌تر CASE عبارت است از مهندسی نرم‌افزار به کمک یک کامپیوتر.

CAD (Computer Aded Design): یا طراحی به کمک کامپیوتر شبیه به CASE است، اما برای طراحی سخت‌افزار استفاده می‌شود.

CAE (Computer Aided Engineerign): به معنی مهندسی به کمک کامپیوتر است در این جا به ترکیب سخت‌افزار یا نرم‌افزار اشاره نشده است.

در لایه ابزارها برای پشتیبانی خودکار یا نیمه‌خودکار از فرآیندها و روش‌های مهندسی نرم‌افزار به‌صورت یکپارچه از CASE استفاده می‌شود.

✓**تست: کدام‌یک از گزینه‌های زیر جزو لایه‌های مهندسی نرم‌افزار نمی‌باشد؟** **(آزاد، فناوری اطلاعات ۸۵)**

۱) فرآیند (Process)
۲) محصول (Product)
۳) روش‌ها (Method)
۴) ابزار (Tools)

☑**پاسخ: گزینه «۲»**

✓**تست: کدام‌یک از گزینه‌های زیر در مورد فرآیند مهندسی نرم‌افزار صحیح نیست؟**

(پیام نور،درس مهندسی نرم‌افزار ۸۸-۸۷)

۱) اساس مهندسی نرم‌افزار، یک لایه به نام فرآیند است.
۲) فرآیند نرم‌افزاری دربرگیرنده فناوری‌هایی است که در مهندسی نرم‌افزار وجود دارند یعنی روش‌های فنی و ابزار خودکار
۳) فرآیند مهندسی نرم‌افزار مانند چسبی است که لایه‌های فناوری را با هم نگه داشته است.
۴) فرآیند چارچوبی برای مجموعه‌ای از حوزه‌های کلیدی و اصلی خود مهیا می‌سازد.

☑**پاسخ: گزینه «۲»**

مراحل مهندسی نرم‌افزار

فعالیت‌های مرتبط با مهندسی نرم‌افزار را صرف‌نظر از اندازه و پیچیدگی و زمینه کاربردی آن می‌توان به سه فاز یا مرحله تقسیم کرد.

۱- مرحله تعریف (Definition Step): در این مرحله تولیدکننده نرم‌افزار سعی بر این دارد تا داده‌های مورد پردازش، وظایف، عملکردها، رفتار سیستمی مورد انتظار، واسط‌ها، قیود و شرایط طراحی و همچنین معیارهای تشخیص اعتبار سیستم را تعیین نماید.

سه فعالیت عمده‌ای که در این مرحله انجام می‌شود عبارت است از:

الف) مهندسی اطلاعات یا سیستم

ب) برنامه‌ریزی پروژه نرم‌افزاری

ج) تجزیه و تحلیل خواسته‌ها و نیازمندی‌ها

این مرحله بر تعیین چه (What) یعنی تعیین موجودیت نرم‌افزار تمرکز دارد.

۲- مرحله توسعه (Development step): در این مرحله سه فعالیت زیر انجام می‌شود.

الف) طراحی نرم‌افزار

ب) تولید کد و پیاده‌سازی نرم‌افزار

ج) آزمون نرم‌افزار تولید شده

در مرحله توسعه، مهندس نرم‌افزار سعی دارد چگونگی ساختمان داده‌ها، چگونگی پیاده‌سازی وظایف و جزئیات رویه‌ای، چگونگی واسطها و نحوه تبدیل طراحی به یک زبان برنامه‌نویسی مناسب و همچنین آزمون نرم‌افزار را به‌طور تفصیلی مشخص نماید. این مرحله تمرکز بر تعیین چگونگی (How) دارد.

۳- مرحله نگهداری (Maintenance Step): در این مرحله ممکن است مراحل تعریف و توسعه مجدداً به‌کار گرفته شود ولی تفاوت در این است که در مرحله نگهداری، مراحل محدود به نرم‌افزار موجود است درحالی‌که در مرحله توسعه و تعریف نرم‌افزاری موجود نیست.

در طول این مرحله با چهار نوع تغییر زیر مواجه می‌شویم:

الف) تصحیح: به‌معنی اصلاح نواقص نرم‌افزار است. نگهداری متناظر با این نوع تغییر نگهداری اصلاحی (corrective) نامیده می‌شود.

ب) تطبیق: به‌معنی وفق دادن یا تطبیق نرم‌افزار با تغییرات محیط پیرامون است. مانند تغییرات سخت‌افزاری، نگهداری متناظر با این نوع تغییر، نگهداری تطبیقی (Adaptive) نامیده می‌شود.

ج) بهسازی و تکمیل: به‌معنی افزودن نیازها و ایجاد تغییرات براساس خواسته‌های جدید مشتری است. نگهداری متناظر با این نوع تغییر، نگهداری تکمیلی نامیده می‌شود (Perfective).

د) پیشگیری: به معنی اعمال روش‌هایی از مهندسی نرم‌افزار است که در حین مرحله نگهداری، از کاهش کیفیت نرم‌افزار به‌واسطه تغییرات حاصل از سه مورد اول جلوگیری می‌کند.

نگهداری متناظر با این نوع تغییر نگهداری پیشگیرانه (Preventive) نامیده می‌شود.

✓تست: کدام‌یک از مراحل زیر در مراحل تولید و به‌کارگیری نرم‌افزار به‌ترتیب دارای کمترین و بیشترین هزینه می‌باشند؟

(آزاد، فناوری اطلاعات ۸۴)

۱) تجزیه و تحلیل و پیاده‌سازی
۲) امکان‌پذیری و نگهداری
۳) طراحی و تجزیه و تحلیل
۴) تعریف و نگهداری

☑پاسخ: گزینه «۲»

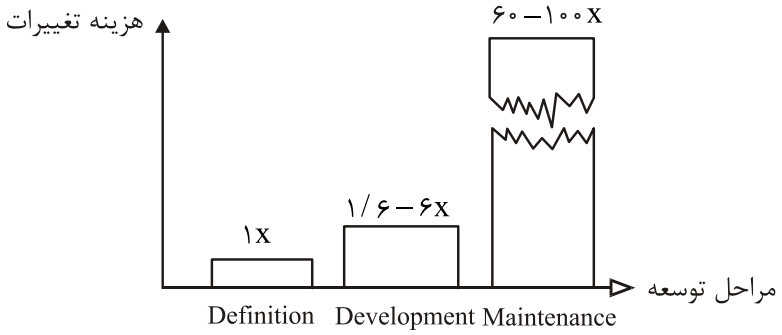
✓تست: کارهای مربوط به مهندسی نرم‌افزار را بدون توجه به حوزه کاربرد، اندازه پروژه یا پیچیدگی آن می‌توان به سه گروه کلی تقسیم کرد. کدام‌یک از گزینه‌های زیر جزو این سه گروه کلی است؟ (پیام نور، مهندسی نرم‌افزار ۸۸-۸۷)

۱) مرحله تعریف
۲) مرحله پشتیبانی
۳) مرحله خطی
۴) گزینه‌های الف و ب هر دو صحیح هستند.

☑پاسخ: گزینه «۴»

هزینه ناشی از تغییرات در چرخه توسعه نرم‌افزار

شکل ۱-٤ هزینه تغییرات را در مراحل مختلف توسعه نرم‌افزار نشان می‌دهد. در فاز تعریف، هزینه تغییرات ۱ برابر، در فاز توسعه هزینه تغییرات ۱/۵ تا ٦ برابر و در فاز نگهداری هزینه تغییرات ٦۰ تا ۱۰۰ برابر افزایش می‌یابد.



شکل۱-۴. مقایسه هزینه فازها

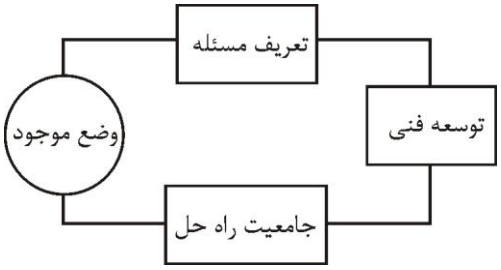
مدل‌های فرآیند نرم‌افزار

برای توسعه نرم‌افزارها نیاز به یک راهبرد توسعه است که دربرگیرنده لایه‌های فرآیند، روش‌ها، نرم‌افزارها و همچنین مراحل کلی مهندسی نرم‌افزار باشد.

این راهبرد را مدل فرآیند یا الگوی مهندس نرم‌افزار می‌نامند.

نکته: مدل فرآیند براساس ماهیت و نوع کاربرد روش‌ها و ابزارهای مورد استفاده و فرآورده‌های قابل تحویل پروژه انتخاب می‌شود.

مراحل توسعه نرم‌افزار را طبق شکل ۱-۵ می‌توان به‌عنوان یک حلقه حل مسئله نشان داد.



شکل ۱-۵. فازهای یک حلقه حل مسئله

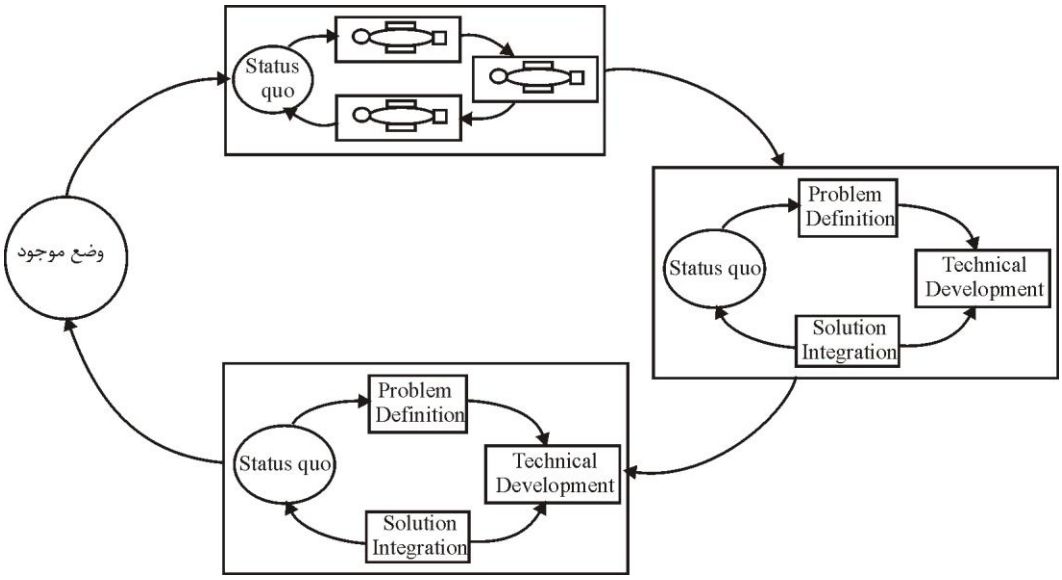
وضع موجود، وضع کنونی امور را نشان می‌دهد.

تعریف مسئله، موجودیت مسأله را مشخص می‌کند و تعیین می‌کند که چه مساله خاصی باید حل شود.

توسعه فنی، مسئله را با استفاده از فناوری حل می‌کند و به‌عبارت دیگر راه‌حل‌های مسئله را بررسی می‌کند و بهترین راه‌حل را انتخاب می‌کند.

جامعیت راه‌حل- نتایج (برنامه‌ها، مستندات، داده‌ها، محصول جدید) را تحویل متقاضی حل مسئله می‌دهد.

نکته: هریک از مراحل تعریف مسئله و جامعیت راه‌حل و توسعه فنی می‌تواند به‌صورت یک حلقه حل مسئله نوشته شود. این موضوع در شکل ۱-۶ نشان داده شده است.

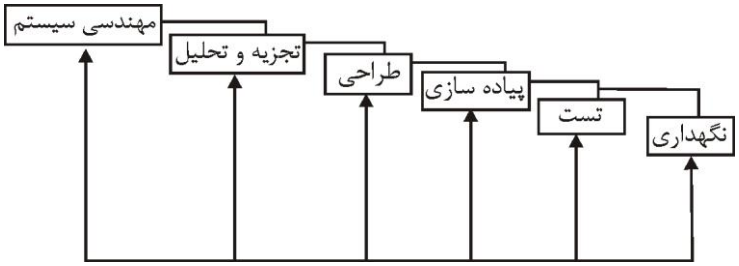


شکل ۱-۶. فازهای موجود در داخل فازهای حلقه حل مسئله

نکته: فرآیند نرم‌افزار مدل‌های گوناگونی دارد که همگی از حلقه حل مسئله تبعیت می‌کنند و مراحل کلی مهندسی نرم‌افزار را پوشش می‌دهند. در ادامه تعدادی از مدل‌های متداول فرآیند نرم‌افزار بررسی می‌شود.

مدل آبشاری (Water fall)

به این مدل، مدل ترتیبی خطی (Liner sequential model) یا چرخه حیات کلاسیک گفته می‌شود. در این مدل کار از سطح سیستم شروع شده و با طی مراحل تحلیل و طراحی، پیاده‌سازی، تست و نگهداری به‌طور کاملا ترتیبی پیش می‌رود. در شکل ۱-۷ روند کاری این مدل نشان داده شده است.



شکل ۱-۷. مدل آبشاری

۱- مهندسی سیستم: از آنجا که همیشه نرم‌افزار، قسمتی از یک سیستم بزرگ‌تر است، ابتدا باید نیازمندی‌های کل سیستم تعیین گردد و سپس خواسته‌های مربوط به نرم‌افزار شناسایی شود و به آن نسبت داده شود. این مرحله شامل جمع‌آوری نیازها در سطح سیستم و تجزیه و تحلیل سطح بالا روی آن است.

نکته: دلیل ضروری بودن این دیدگاه از سیستم تعامل نرم‌افزار با سایر مؤلفه‌های سیستم از قبیل افراد، سخت‌افزارها، پایگاه داده‌ها و غیره است.

۲- تجزیه و تحلیل: به‌معنی جمع‌آوری اطلاعات و نیازمندی‌هایی که تنها مربوط به نرم‌افزار است. برای فهمیدن برنامه‌ای که باید نوشته شود تحلیل‌گر نرم‌افزار (Analyst) باید دامنه اطلاعات، عملیات واسطها، ورودی و خروجی‌ها را به‌طورکامل مورد بررسی قرار دهد. نتایج کار در این مرحله به‌طور کامل مستندسازی شده و با مشتری بازنگری و اصلاح می‌شود.

نکته: تفاوت این مرحله با مهندسی سیستم در این است که در این مرحله تنها نرم‌افزار تحلیل می‌شود و نه کل سیستم.

۳- طراحی: فرآیندی چند مرحله‌ای است که روی چهار صفت متفاوت از برنامه تمرکز دارد.

۱- ساختمان داده‌ها ۲- معماری نرم‌افزار ۳- واسط‌ها ۴- جزئیات رویه‌ای (الگوریتم‌ها)

در واقع هدف از این مرحله تبدیل خواسته‌ها به نمودی از نرم‌افزار است به‌گونه‌ای که بتوان قبل از پیاده‌سازی و کدنویسی آن را ارزیابی کرد. طراحی نیز مانند بخش قبل مستندسازی می‌شود.

۴- پیاده‌سازی: پیاده‌سازی، یعنی طراحی باید به شکلی تبدیل شود که برای ماشین قابل فهم شود. درصورتی‌که طراحی با جزئیات کامل و به‌صورت تفصیلی انجام شده باشد آنگاه پیاده‌سازی را می‌توان با ابزارهای سریع‌تر و تا حد زیادی به‌صورت خودکار انجام داد.

۵- آزمون یا تست: بعد از مرحله پیاده‌سازی تست برنامه آغاز می‌شود که بر دو جنبه تأکید دارد.

الف) روال منطقی داخلی نرم‌افزار: یعنی حصول اطمینان از اینکه تمامی دستورالعمل‌ها و الگوریتم‌ها درست نوشته شده‌اند (تست جعبه سفید).

ب) خروجی صحیح برنامه: یعنی اطمینان از اینکه به‌ازای یک ورودی تعریف شده نتایج مورد نظر ایجاد می‌شود (تست جعبه سیاه).

۶- نگهداری: اکثر نرم‌افزارها بعد از تحویل به مشتری دستخوش تغییراتی می‌شوند (به‌غیر از نرم‌افزارهای جاسازی شده (embeded)) که دلایل این تغییرات عبارت هستند از:

۱- مشتری خواستار قابلیت‌های جدید است.

۲- خطاهای موجود در برنامه

۳- نرم‌افزار باید با تغییرات محیط بیرونی و تغییرات سخت‌افزاری منطبق شود.

نکته: در مرحله نگهداری نرم‌افزار، کلیه مراحل قبلی مدل را برای نرم‌افزارهای موجود (نه برنامه‌های جدید) دوباره به‌کار می‌گیرد.

خصوصیات مدل آبشاری

۱- یکی از مشکلات این مدل نرم‌افزاری این است که پروژه‌های نرم‌افزاری واقعی به‌ندرت ترتیب خطی پیشنهادی توسط این مدل را دنبال می‌کنند، به این معنی که ماهیت پروژه‌های نرم‌افزاری به‌گونه‌ای نیست که بتوان در آنها یک مرحله را به اتمام رساند و سپس مرحله دیگر را دنبال کرد؛ بلکه همیشه نیاز داریم به مرحله قبلی برگردیم و این امر باعث سردرگمی در مراحل مدل آبشاری می‌شود.

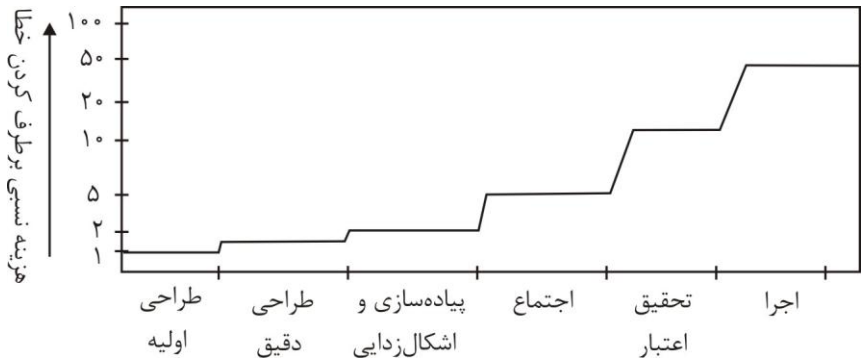
۲- طبق تعاریفی که از مدل آبشاری شد در این مدل نیاز است که تمامی نیازهای برنامه و نرم‌افزار از قبل مشخص شود درحالی‌که غالباً این امر برای مشتری دشوار است. بنابراین این مدل برای پروژه‌هایی که موارد غیرقطعی زیادی در ابتدا دارند استفاده چندانی ندارد و این یکی دیگر از معایب این مدل نرم‌افزاری است.

۳- در این مدل، مشتری برای اینکه بتواند یک نسخه کاری از نرم‌افزار را مشاهده کند باید تا پایان پروژه منتظر بماند. به این دلیل یک اشتباه بزرگ در پروژه نرم‌افزاری می‌تواند به شدت خطرناک باشد.

۴- مدل آبشاری قدیمی‌ترین و پرکاربردترین مدل فرآیند نرم‌افزاری است ولی با توجه به مواردی‌که گفته شد انتقادات زیادی در ارتباط با آن وجود دارد.

۵- ترتیبی بودن این مدل و این‌که نیازها در ابتدای پروژه مشخص نیستند باعث به‌وجود آمدن مشکلاتی نظیر عدم‌انعطاف‌پذیری کافی است. به این معنی که اعمال تغییرات پس از شروع پروژه مستلزم صرف زمان و هزینه زیادی است.

در شکل ۱-۸ ارتباط هزینه رفع خطاها نسبت به راه‌حل کشف آنها نشان داده شده است.



شکل ۱-۸. ارتباط هزینه رفع خط نسبت به راه‌حل کشف آن‌ها

✓**تست: به چه منظور از مدل آبشاری جهت توسعه نرم‌افزار استفاده می‌شود؟** **(آزاد، فناوری اطلاعات ۸۴)**

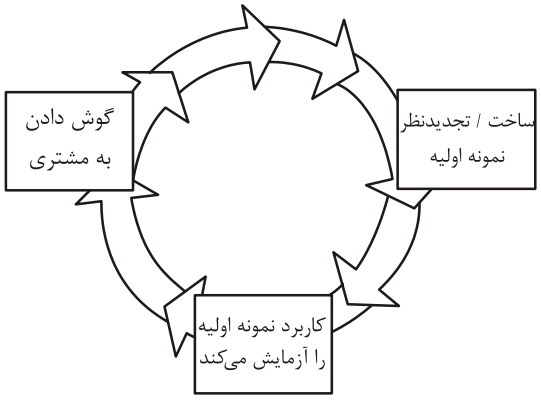
۱) جهت پایین آمدن هزینه نگهداری نرم‌افزار
۲) به‌منظور ساده کردن مدیریت فرآیند نرم‌افزار
۳) جهت جلوگیری از تکرار فرآیندها
۴) چون می‌تواند مدل‌های دیگر را در برگیرد.

☑**پاسخ: گزینه «۲»**

زیرا در مدل آبشاری مدیریت فرآیند توسعه نرم‌افزار ساده است.

مدل نمونه‌سازی (Prototyping Model)

در بعضی مواقع مشتری یکسری اهداف کلی را برای نرم‌افزار بیان می‌کند امّا جزئیات ورودی و خروجی‌ها را مشخص نمی‌کند و یا امکان دارد توسعه‌دهنده نرم‌افزار از کارایی یک الگوریتم (مناسب نبودن سیستم عامل و یا شکل محاوره انسان با کامپیوتر) مطمئن نباشد. در چنین مواردی روش نمونه‌سازی مناسب است. شکل ۱-۹ الگوی ساخت مدل نمونه‌سازی را نشان می‌دهد.



شکل ۱-۹. الگوی ساخت مدل نمونه‌سازی

این مدل با جمع‌آوری نیازها و خواسته‌های مشتری و به‌دنبال آن ساخت نمونه اولیه آغاز می‌شود. مشتری و تولیدکننده نرم‌افزار با هم ملاقات کرده و اهداف کلی نرم‌افزار را تعیین می‌کنند، سپس یک طرح سریع اتفاق می‌افتد تا آن دسته از ویژگی‌هایی که مشتری/کاربر در نظر دارد (مثل صفحات ورودی، اطلاعات و فرمت خروجی‌ها) سریعاً طراحی می‌گردد و به‌دنبال این طراحی، یک نمونه دیگر طراحی شده و توسط مشتری و کاربر مورد ارزیابی قرار می‌گیرد. مجدداً براساس نیازها نمونه‌های دیگری توسط تولیدکننده نرم‌افزار طراحی شده و در اختیار مشتری قرار داده می‌شود. این چرخه تا زمانی‌که نرم‌افزار مورد نظر ساخته شود تکرار می‌شود.

خصوصیات و ویژگی‌های مدل نمونه‌سازی

۱- همان‌طور که در تعریف این مدل بررسی بیان شد، این مدل برای پروژه‌هایی که در ابتدای آن‌ها موارد غیرقطعی زیادی وجود دارد مناسب است.
کلیه قوانینی که هر دو طرف پروژه باید بدانند بایستی در ابتدای کار وضع شوند. برای مثال: مشتری باید بداند که ساخت نمونه اولیه فقط به‌عنوان مکانیزمی جهت شناسایی نرم‌افزار بوده و نرم‌افزار نهایی حتماً دارای معیارهای کیفیتی خواهد بود.
۲- سازندگان نرم‌افزار ممکن است برای ساخت و به‌کارگیری هرچه سریع‌تر نمونه اولیه، در پیاده‌سازی آن کوتاهی کنند. برای مثال برای سرعت در پیاده‌سازی سیستم عامل زبانی را انتخاب کرده که در کیفیت ایده‌آل نرم‌افزار مناسب نباشد.
۳- یکی از معایب این است که مشتری تنها یک نسخه کاری از نرم‌افزار را می‌بیند و نمی‌داند این نمونه اولیه فقط یک ماکت است. مشتری به محض اطلاع از این مسئله ممکن است ناراحت شده و تقاضای تولید هرچه سریع‌تر یک نرم‌افزار را داشته باشد.

مدل توسعه سریع نرم‌افزار (RAD: Rapid Application Development)

این مدل بر یک چرخه تولید بسیار کوتاه تأکید دارد و یک مدل توسعه افزایشی است. مدل RAD شکل پرسرعت مدل آبشاری است و دلیل این سرعت بهره‌گیری از تکنیک ساخت مبتنی بر مؤلفه‌ها است.
نکته: در این مدل اگر دامنه پروژه کوچک باشد و نیازمندی‌ها به خوبی شناسایی شوند، می‌توان یک سیستم کاملاً عملیاتی را در مدت زمان بسیار کوتاه (مثلاً بین ۶۰ تا ۹۰ روز) تولید نمود.
مدل توسعه سریع نرم‌افزار، شامل مراحل زیر می‌باشد:
۱- مدل‌سازی تجاری یا کسب و کار (Business modeling): جریان‌های اطلاعات در این مرحله بین واحدهای تجاری به گونه‌ای مدل می‌شود که به سوالات زیر پاسخ داده شود.
الف) چه اطلاعاتی فرآیند کسب و کار را پیش می‌برد؟
ب) چه اطلاعاتی در وضعیت موجود تجاری تولید می‌شود؟
ج) مسئول تولید اطلاعات چه کسانی هستند؟
د) اطلاعات به کجا می‌روند؟
ه) چه کسی اطلاعات را پردازش می‌کند؟
۲- مدل‌سازی داده‌ای (Data Modeling): جریان اطلاعات به‌دست آمده از مراحل قبل به‌صورت کامل پالایش شده و اشیاء داده‌ای موردنیاز از آن استخراج می‌شوند صفات هر شیء مشخص شده و ارتباط اشیاء معین می‌گردد.
۳- مدل‌سازی فرآیند (Process Modeling): پردازش‌های موردنیاز برای افزودن، اصلاح، حذف، بازیابی کردن اشیاء داده‌ای تعریف می‌شوند.
۴- تولید برنامه‌های کاربردی (Application Generation): مدل RAD فرض را به استفاده از زبان‌های نسل چهارم گذاشته است و به‌جای نوشتن کد در این مدل سعی بر این است که از اجزاء و قطعات از قبل آماده استفاده شود.
۵- آزمایش و تحویل (Test and Turn Over): با توجه به اینکه مدل توسعه سریع نرم‌افزار از مؤلفه‌هایی که قبلاً تست شده‌اند استفاده می‌کند لذا زمان کلی آزمایش و بررسی خطا پایین آمده، اما این موضوعی دلیلی بر عدم آزمایش مولفه‌های موجود نیست، بلکه این مولفه‌ها به همراه کلیه واسطهای آنها باید آزمایش شوند.

تست: مدل ساخت سریع برنامه‌ها در برگیرنده کدام‌یک از مراحل نیست؟ **(پیام نور، مهندسی نرم‌افزار ۸۸-۸۷)**

۱) مدل‌سازی فرآیند
۲) تولید برنامه کاربردی
۳) مدل‌سازی نمونه‌سازی
۴) آزمون و گردش

پاسخ: گزینه «۳»

خصوصیات مدل توسعه سریع نرم‌افزار (RAD)

۱- در این مدل تولیدکنندگان و مشتریان باید نسبت به فعالیت‌های توسعه سریع نرم‌افزار در زمان کوتاه تعهد داشته باشند، کوتاهی هر یک از طرفین منجر به شکست پروژه خواهد گردید.

۲- برای پروژه‌های بزرگ باید منابع انسانی کافی باشد تا بتوان تیم‌های RAD را به درستی تشکیل داد. هر تیم مسئول بخشی از پروژه خواهد بود. تنها مشکلی که در این قسمت ممکن است وجود داشته باشد مسئله یکپارچه کردن نتایج تیم‌ها است که لازمه آن زمان و دقت کافی است.

۳- در تمامی کاربردها و برنامه‌ها نمی‌توان از این مدل استفاده نمود مانند:

الف) جایی‌که نیاز به‌کارایی (Performance) بالا است.

ب) جایی‌که نتوان سیستم را به‌صورت مناسب واحدبندی کرد.

ج) جایی‌که احتمال بروز خطرات فنی بالا باشد.

نکته: خطرات فنی زمانی رخ می‌دهد که نرم‌افزار جدید به‌طور گسترده از فناوری‌های جدید استفاده کند یا به درجه بالایی از همکاری و تعامل با سایر برنامه‌ها رخ دهد.

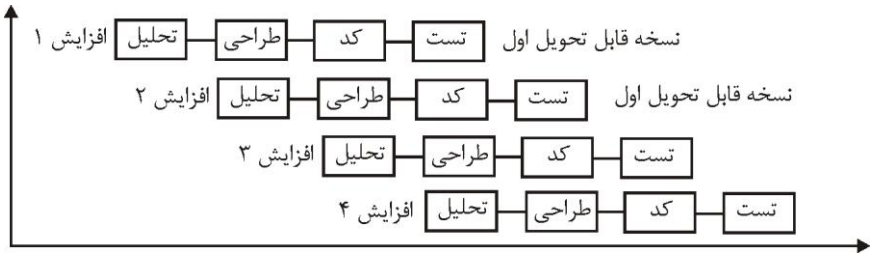
تست: در کدام‌یک از مدل‌های فرآیند نرم‌افزار بر کوتاه نمودن چرخه توسعه نرم‌افزار (توسعه سریع) تاکید می‌گردد؟ **(آزاد، فناوری اطلاعات ۸۵)**

۱) مدل Incremental
۲) مدل مارپیچی
۳) مدل Win Win
۴) مدل RAD

پاسخ: گزینه «۴»

مدل افزایشی (Incremental)

نام دیگر این مدل، مدل گام‌به‌گام است. در این مدل عناصر مدل آبشاری با روش تکرار در مدل نمونه‌سازی ترکیب شده است به این‌صورت که مراحل مدل آبشاری مرتباً تکرار می‌شود با این ویژگی که هر تکرار روی قسمت کوچکی از نرم‌افزار انجام می‌شود. هر کدام از این تکرارها یک افزایش تحویل دادنی را تولید می‌نماید. در شکل ۱-۱۰ مدل افزایشی و مراحل هر افزایش نشان داده شده است.



شکل ۱-۱۰. مدل افزایشی و مراحل هر افزایش

مثال: یک نرم‌افزار واژه‌پرداز (پردازشگر متن) را در نظر بگیرید.

با استفاده از مدل افزایشی مراحل توسعه و تکامل به شرح زیر است:

افزایش اول: عملیات پایه‌ای و اولیه از قبیل مدیریت فایل، ویرایش، عملیات تولید سند

افزایش دوم: قابلیت‌ها و ویرایش‌های پیچیده‌تر در تولید سند

افزایش سوم: کنترل املا و دستور و چک کردن نوشتار

افزایش چهارم: طراحی پیشرفته قالب و صفحه‌بندی و کنترل و طراحی ساختار چاپ

نکته: در مدل افزایشی، هسته محصول (core product) در افزایش اول ساخته می‌شود. به بیان دیگر در افزایش اول به خواسته‌های پایه و اصلی مشتریان پرداخته شده و از ویژگی‌های مکمل جنبی صرف‌نظر می‌گردد.

در مراحل بعد این مدل قابلیت‌ها و ویژگی‌های دیگر مورد انتظار به محصول هسته‌ای اضافه می‌شود.

خصوصیات مدل افزایشی:

۱- اگر مشکل کمبود نیروی انسانی وجود داشته باشد، استفاده از این مدل مفید خواهد بود، زیرا افزایش‌های اولیه را می‌توان با تعداد افراد کمتری پیاده‌سازی نمود.

۲- بر خلاف مدل نمونه‌سازی، در این مدل قطعات و نمونه‌های قابل تحویل ساخته می‌شوند. نمونه‌های اولیه، نسخه‌های ناقصی هستند از محصول نهایی که قادر به ارائه خدمات به‌کاربر هستند.

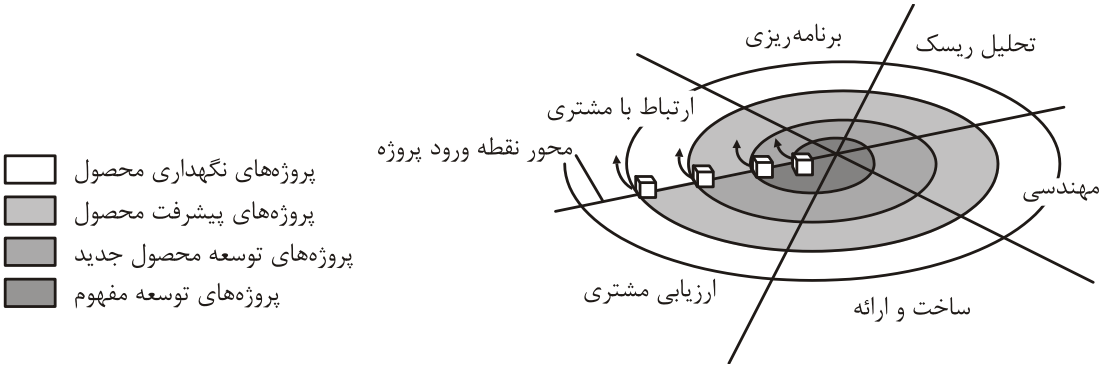
۳- این مدل به‌عنوان یک مدل تکاملی توسعه نرم‌افزار استفاده می‌شود.

۴- تفاوت اصلی این مدل با مدل نمونه‌سازی در ماهیت تکاملی این مدل در توسعه محصول است اما مدل نمونه‌سازی بر شناسایی تکاملی خواسته‌ها تاکید دارد نه توسعه تکاملی.

مدل حلزونی (Spiral)

این مدل همانند روش قبل ماهیت تکراری مدل نمونه‌سازی را با کنترل سیستماتیک مدل آبشاری ترکیب می‌کند.

با این روش نرم‌افزار به‌صورت یک مجموعه‌ای از نسخه‌های افزایشی توسعه پیدا می‌کند. در تکرارهای اولیه، نسخه تکاملی ممکن است یک مدل کاغذی یا یک نمونه اولیه (Prototype) باشد. در طول تکرارهای بعدی هر بار نسخه کامل‌تری از سیستم، مهندسی و تولید می‌شود.



شکل ۱-۱۱. مدل حلزونی

این مدل به تعدادی فعالیت تقسیم می‌شود که به آن‌ها نواحی وظیفه یا کاری (Task Region) می‌گویند که این ناحیه‌ها بین ۳ تا ۶ قسمت تقسیم می‌شوند.

نواحی کاری مدل حلزونی و مجموعه وظایف هر یک عبارتند از:

ارتباط با مشتری: مجموعه وظایف و فعالیت‌های موردنیاز برای برقراری ارتباط موثر بین مشتری و توسعه‌دهنده ارتباط با مشتری (customer communication)گفته می‌شود.

طرح‌ریزی یا برنامه‌ریزی: به مجموعه وظایف لازم برای تعریف منابع، برنامه‌ریزی و سایر اطلاعات مربوط به پروژه برنامه‌ریزی (planning) گفته می‌شود.

تحلیل ریسک (Risk Analysis): مجموعه وظایف موردنیاز برای ارزیابی و تعیین مخاطرات مدیریتی و فنی

مهندسی (Engineering): مجموعه وظایف موردنیاز برای ساخت یک چند ارائه از برنامه

ساخت و ارائه (constraction and Release): مجموعه وظایف مربوط به ساخت آزمون، نصب، تست و پشتیبانی

ارزیابی مشتری (customer Evaluation): مجموعه وظایف و فعالیت‌ها برای به‌دست آوردن نظرات مشتری

نکته: با شروع این مدل تیم مهندسی نرم‌افزار حول شکل حلزونی در جهت عکس عقربه‌های ساعت حرکت می‌کند، در اولین دور ممکن است فقط تعریف محصول ایجاد شود.

خصوصیات مدل حلزونی:

۱- می‌توان برای بکارگیری در سراسر عمر نرم‌افزار تطبیق داد.

۲- گرفتار مشکل متقاعد کردن مشتری به این واقعیت است که مدل تکاملی قابل‌کنترل است.

۳- به مهارت فراوانی برای تحلیل ریسک نیاز دارد و موفقیت خود را در گرو این مسئله گذاشته است.

۴- هنوز به اندازه مدل نمونه‌سازی و آبشاری استفاده نشده است.

۵- از الگوی نمونه‌سازی به‌عنوان راه‌کاری جهت کاهش ریسک پروژه استفاده می‌نماید.

۶- روشی واقع‌بینانه است که برای توسعه نرم‌افزارهای بزرگ استفاده می‌شود زیرا نرم‌افزار در هر بار پیشرفت فرآیند تکامل می‌یابد و در هر تکرار مخاطرات احتمالی بررسی و شناسایی می‌شوند.

تست: دو پروژه را در نظر بگیرید: **(آزاد، فناوری اطلاعات ۹۰)**

یک. مشتری تاکید بر انجام سریع پروژه در مدت ۳ ماه می‌نماید. تغییرات در درخواست مشتری اندک بوده و امکان تقسیم پروژه به قسمت‌های مختلف وجود دارد.

دو. پروژه نیازمند عواملی می‌باشد که در طول انجام پروژه امکان دارد تغییر نماید و تا چند روز به عرضه نهایی محصول قطعی نمی‌گردد.

۱) مدل RAD و مدل Spiral
۲) مدل RAD و مدل Linear
۳) مدل Spiral و مدل RAD
۴) مدل Spiral و مدل Spiral

پاسخ: گزینه «۱»

مدل حلزونی Win – Win

هدف از بررسی مدل حلزونی این بود که فعالیتی را تحت عنوان ارتباط با مشتری معرفی می‌نماید که در آن مشتری بازخوردی را از سیستم ارائه می‌دهد. در حالت کلی توسعه‌دهنده نرم‌افزار از مشتری آنچه را که نرم‌افزار نیاز دارد را می‌پرسد و مشتری جزئیاتی دقیق را ارائه می‌کند. ذکر این نکته بدیهی است که به‌ندرت پیش می‌آید اطلاعاتی که مشتری در اختیار می‌گذارد صورت دقیق باشد. از مشتری ممکن است خواسته شود تا عملکرد کارایی و سایر خصوصیات نرم‌افزار را در برابر هزینه و زمان تحویل به بازار متعادل سازد و در واقع هر آنچه گفت اجرا نشود.

بهترین مذاکره سعی می‌کند نتایج "برنده – برنده" را تولید کند یعنی هر دو طرف (مشتری و سازنده) در داخل یک فرآیند مذاکره قرار می‌گیرند تا در مورد خواسته‌های طرفین تصمیم بگیرند.

که سود مشتری یک سیستم عملیاتی رضایت‌بخش و سود سازنده انجام یک پروژه موفق واقعی و مقرون به صرفه است.

در مدل حلزونی Win – Win فعالیت‌های زیر انجام می‌شود:

۱- مشخص کردن ذینفع‌های اصلی سیستم و یا زیر سیستم‌ها (stake holder)

۲- مشخص کردن شرایط سود برای ذینفع‌ها

۳- مذاکره در مورد شرایط برد ذی‌نفع‌ها به‌منظور توافق آنها و حصول شرایط برد – برد برای همه

نکته: مدل حلزونی Win – Win علاوه بر تاکیدی که بر سریع بودن مذاکرات دارد سه مرحله فرآیندی دارد که به آنها لنگرگاه گفته می‌شود. نقاط لنگرگاه سه دیدگاه متفاوت از پیشرفت پروژه را به‌شرح زیر تبیین می‌کنند.

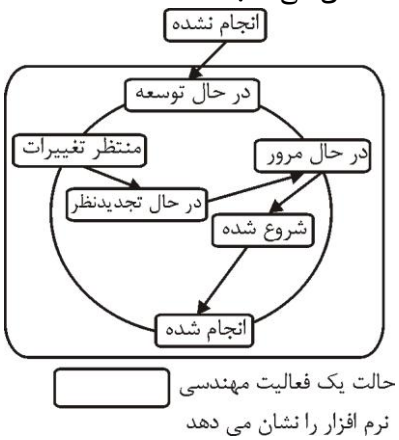
الف) اهداف چرخه حیات (LCO): مجموعه‌ای از اهداف را برای فعالیت‌های عمده در مهندسی نرم‌افزار تعیین می‌کند.

ب) معماری چرخه حیات (LCA): مجموعه اهدافی را تعیین می‌کند که باید به موازات تعریف معماری سیستم و نرم‌افزار برآورده شوند.

ج) قابلیت عملیاتی اولیه (ICO): مجموعه‌ای از اهداف را در ارتباط با نصب، تحویل و نگهداری نرم‌افزار مشخص می‌کند.

مدل توسعه همروند (Concurrent Development Model)

این مدل که مهندسی همزمانی نامیده می‌شود به‌صورت شیمیایی از یک سری فعالیت‌هایی تشکیل شده و هر فعالیت دارای وضعیت خاص خود است. برای مثال در مدل حلزونی فعالیت‌های مهندسی تعریف شده عبارت هستند از نمونه سازی، مدل‌سازی تحلیل، تعیین نیازمندی‌ها و طراحی که در هر لحظه می‌تواند در یکی از حالت‌های نشان داده شده باشد. شکل ۱-۱۲ وضعیت‌های مختلف را برای هر فعالیت نشان می‌دهد.



شکل ۱-۱۲. مدل توسعه همروند

به‌عنوان مثال وقتی فعالیت ارتباط با کاربر انجام شد و در وضعیت منتظر تغییرات قرار گرفت، فعالیت تحلیل وارد وضعیت در حال توسعه می‌شود. در این موقع اگر کاربر تغییراتی را به اطلاع برساند وضعیت فعالیت تحلیل به منتظر تغییرات تبدیل می‌شود و در واقع از وضعیت در حال توسعه به وضعیت منتظر تغییرات انتقال می‌یابد.

در این مدل مجموعه‌ای از رویدادها تعریف می‌شود که وقوع آن‌ها باعث گذار از حالتی به حالت دیگر برای هر یک از فعالیت‌های مهندسی نرم‌افزار می‌شود.

خصوصیات مدل توسعه همروند:

الف) این مدل اغلب برای توسعه نرم‌افزارهای مشتری/ کارگزار (Client / server) استفاده می‌شود.

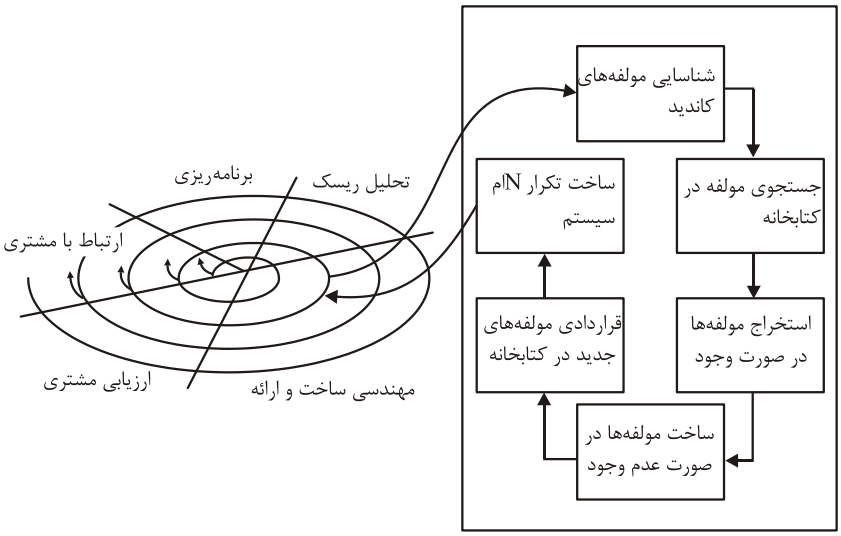
ب) در این مدل همه فعالیت‌ها به‌صورت موازی انجام می‌شوند و رخدادهای یک فعالیت روی فعالیت دیگر در شبکه تاثیر می‌گذارد.

ج) این مدل را می‌توان در مورد همه انواع نرم‌افزارها به‌کار گرفت و این مدل تصویری صحیح از حالت جاری پروژه را ارائه می‌کند.

مدل توسعه بر مبنای مولفه (Component base Development)

تکنولوژی‌های شیگرایی زیرساخت‌های فنی خاصی را برای بهره‌گیری از این مدل در مهندسی نرم‌افزار فراهم می‌نمایند. در روش شیگرایی کلاس‌ها می‌توانند داده‌ها و الگوریتم‌ها را کپسوله‌سازی نمایند. اگر این کلاس‌ها به‌صورت مناسبی نوشته شده باشند می‌توان از آنها برای کاربردهای مختلف دیگر و در معماری‌های سیستم‌های کامپیوتری استفاده مجدد نمود.

نمایی از مدل توسعه برمبنای مولفه در شکل ۱-۱۳ آمده است:



شکل ۱-۱۳. مدل توسعه بر مبنای مولفه

از جمله مزایایی این مدل این است که نرم‌افزارها را با استفاده از مونتاژ قطعات از پیش‌ساخته و از پیش بسته‌بندی شده تولید می‌کند. یکی از این مزایا کاهش اثرات و تبعات بحران نرم‌افزار است.

براساس یک تحقیق استفاده مجدد در نرم‌افزار باعث کاهش زمان توسعه به میزان ۷۰٪ کاهش هزینه پروژه به میزان ۸٤٪ و افزایش بهره‌وری به میزان ۲٦/۲٪ می‌شود.

نکته: مدل توسعه بر مبنای مولفه از بسیاری از خصوصیات روش حلزونی استفاده می‌نماید. این مدل از نظر ماهیت، مدل تکاملی است و ساختاری تکرارشونده را برای توسعه نرم‌افزار در پیش می‌گیرد.

تست: فرآیند یکپارچه توسعه نرم‌افزار (USDP) به‌عنوان نماینده کدام‌یک از مدل‌های فرآیند نرم‌افزار معرفی می‌گردد؟

(آزاد، فناوری اطلاعات ۸۶)

۱) مدل توسعه هم‌روند

۲) مدل مارپیچی برنده- برنده (Win- Win)

۳) مدل روش‌های رسمی

۴) مدل توسعه بر مبنای مولفه (Component Base Development)

پاسخ: گزینه «۴»

فرآیند یکپارچه توسعه نرم‌افزار (Unified Software development Process)، نماینده تعدادی از مدل‌های تولید بر پایه اجزا است که در صنعت پیشنهاد شده‌اند. با استفاده از زبان مدل‌سازی یکپارچه (UML)، فرآیند یکپارچه جزءهایی را تعریف می‌نماید که برای ساختن سیستم استفاده شده و رابط‌هایی که این جزءها را به هم وصل می‌کنند. با استفاده از تولید فزآینده و تکراری، فرآیند یکپارچه تابع سیستم را با به‌کارگیری رهیافت مبتنی بر سناریو تعریف می‌کند. (از دیدگاه کاربر). سپس تابع را با یک چارچوب معماری و ساختاری مرتبط می‌نماید که فرم نرم‌افزار را مشخص می‌سازد.

تست: با استفاده از زبان مدل‌سازی یکپارچه (UML) کدام‌یک از مدل‌های تکاملی فرآیند نرم‌افزار زیر را می‌توان تعریف کرد؟

(پیام نور، درس مهندسی نرم‌افزار ۸۸-۸۷)

۱) مدل توسعه مبتنی بر اجزا

۲) مدل حلزونی Win-Win

۳) مدل آبشاری

۴) مدل حلزونی

پاسخ: گزینه «۱»

مدل روش‌های رسمی یا فرمال (Formal Methods)

این مدل شامل مجموعه‌ای از فعالیت‌هایی است که نرم‌افزار را به‌صورت ریاضی و رسمی تعریف می‌نماید. در این روش یک مهندس نرم‌افزار می‌تواند با استفاده از علائم ریاضی یک سیستم کامپیوتری را تعریف، پیاده‌سازی و وارسی کند.

نکته: به گونه‌ای از روش‌های فرمال، روش مهندسی نرم‌افزار اتاق تمیز (Clean Room) می‌گویند.

تست: مهندسی نرم‌افزار اتاق تمیز (Clean Room) به کدام دسته از مدل‌های فرآیند نرم‌افزار تعلق دارد؟

(آزاد، فناوری اطلاعات ۸۸)

۱) مدل بسط همزمان
۲) مدل مار پیچی Win-Win
۳) مدل توسعه مبتنی بر مولفه
۴) مدل روش‌های رسمی

پاسخ: گزینه «۴»

خصوصیات این مدل عبارت هستند از:

۱- تولید مدل‌های رسمی کار بسیار وقت‌گیر و پرهزینه‌ای است.

۲- در این مدل مشتری حتماً باید دید فنی داشته باشد، در غیراین‌صورت مدل ناکارآمدی است.

۳- به‌علت اینکه تعداد کمی از تولیدکنندگان نرم‌افزار پیش زمینه خاصی در مورد به‌کار بردن روش‌های فرمال دارند، استفاده از این روش‌ها نیازمند آموزش گسترده و هزینه زیاد است.

۴- احتمال اینکه این مدل در آینده، برای نرم‌افزارهایی که از نوع بحرانی امنیتی یا اقتصادی باشند طرفدار پیدا کند، زیاد است.

۵- این مدل نویدی برای رسیدن به نرم‌افزارهای بی عیب و نقص است.

۶- این مدل امکان کشف و اصلاح خطاهای زمان اجرا را در طول مراحل اولیه تولید نرم‌افزار فراهم می‌آورد.

تست: کدام‌یک از مدل‌های فرآیند زیر بر اصول ریاضی در توسعه نرم‌افزار تاکید دارد؟ **(آزاد، فناوری اطلاعات ۸۹)**

۱) مدل گام‌به‌گام (Incremental Model)
۲) مدل روش‌های رسمی
۳) مدل مارپیچی
۴) مدل بسط همزمان

پاسخ: گزینه «۲»

تست: کدام‌یک از ویژگی‌های زیر جزو خصایص مدل‌های رسمی نمی‌باشد؟ **(آزاد، فناوری اطلاعات ۹۰)**

۱) از آنجا که به‌صورت کلیشه‌ای می‌باشد، هزینه زمانی اندکی دارد.

۲) با اعمال یک نظم ریاضی شدید سیستم کامپیوتری را توسعه می‌دهد.

۳) در نرم‌افزارهای بحرانی- امنیتی (مثلا نرم‌افزارهای مرتبط به پزشکی و هوافضا) کاربرد فراوانی دارد.

۴) استفاده از این مدل به‌عنوان راهکار ارتباطی با مشتریانی که دید فنی ندارند، دشوار است.

پاسخ: گزینه «۱»

تکنیک‌های نسل چهارم (Fourth Generation Techniques)

شامل مجموع وسیعی از ابزارهای نرم‌افزاری است که مهندس نرم‌افزار را قادر می‌سازد تا برخی از خصوصیات نرم‌افزار را در سطح بالایی تعریف کند.

این عقیده وجود دارد که هر چه سطح توصیف و تشریح نرم‌افزار بالاتر باشد آن را می‌توان سریع‌تر ساخت. به همین خاطر تکنیک‌های نسل چهارم بر توصیف نرم‌افزار توسط اشکال و علائم گرافیکی و یا زبان‌های سطح بالای نزدیک به زبان طبیعی تاکید فراوان دارد.

در محیط‌های توسعه نرم‌افزار که از 4GT حمایت می‌کنند و از ابزارها و ویژگی‌های زیر در آنها استفاده می‌شود:

۱- زبان‌های غیررویه‌ای جهت پرس‌وجو از پایگاه داده

۲- دستکاری داده‌ها

۳- تولید گزارش

۴- تعریف صفحات نمایش و محاوره با آن‌ها

۵- تولید کد
۶- قابلیت گرافیکی و تصویری سطح بالا
۷- قابلیت کار با صفحات گسترده Speard Sheet
۸- تولید خودکار کد HTML در ابزارهای حرفه‌ای ساخت وب سایت

به‌طورکلی می‌توان مراحل الگوی 4GT را در چهار مرحله به‌صورت زیر خلاصه‌سازی نمود:
۱- جمع‌آوری نیازها
۲- راهبرد طراحی
۳- پیاده‌سازی با 4GT
۴- یکپارچه‌سازی محصول

خصوصیت تکنیک‌های نسل چهارم به شرح زیر است:
۱- در پروژه‌های کوچک می‌توان مرحله راهبرد طراحی را حذف نمود ولی در پروژه‌های بزرگ خیر. اگر در پروژه‌های بزرگ چنین کاری انجام شود محصول نهایی دارای کیفیت بسیار پایینی خواهد بود.
۲- طرفداران 4GT بر این باورند که با استفاده از این الگو زمان توسعه نرم‌افزار کاهش و میزان تولید و بهره‌وری افراد سازنده تا حد زیادی افزایش می‌یابد.
۳- مخالفان 4GT بر این باورند که ابزارهای کنونی 4GT، نسبت به زبان‌های برنامه‌نویسی آسان‌تر نیست و کد خودکاری که تولید می‌شود اساساً ناکارامد بوده و قابلیت نگهداری محصول نیز مورد تردید است.
۴- این روش در کنار ابزارهای CASE و مولد کد قادر به ارائه راه‌حل قانع‌کننده‌ای برای اکثر مشکلات است.
۵- بررسی شرکت‌هایی که از این مدل استفاده کرده‌اند نشان می‌دهد که زمان لازم برای تحلیل طراحی و تولید نرم‌افزارهای کوچک تا حد زیادی کاهش می‌یابد که این کاهش در مرحله تولید محسوس‌تر است.
۶- استفاده از الگوی 4GT برای پروژه‌های بزرگ ممکن است زمان مراحل تحلیل طراحی و آزمون را افزایش دهد. اما از این جهت که زمان کدنویسی تقریباً حذف می‌شود لذا در زمان کل صرفه‌جویی می‌گردد.
۷- اگر این الگو با روش توسعه بر مبنای مولفه‌ها همراه شود می‌تواند در آینده به روش حاکم در توسعه نرم‌افزار تبدیل شود.

تیم نرم‌افزاری

منظور از تیم نرم‌افزاری تخصیص و جایگذاری نیروی انسانی موجود برای توسعه نرم‌افزار است.
برای تخصیص n نفر به مدت k سال برای یک پروژه می‌توانیم از حالات زیر استفاده کنیم.
الف) n نفر را به m کار عملیاتی مختلف به‌گونه‌ای تخصیص دهیم که کار ترکیبی کمی اتفاق افتد. وظیفه هماهنگی n نفر بر عهده مدیر پروژه خواهد بود.
ب) n نفر را در قالب تیم‌های داخلی غیررسمی به m کار عملیاتی مختلف تخصیص دهیم. هر تیم دارای مدیر خود خواهد بود ولی وظیفه هماهنگی بین این مدیران بر عهده مدیر پروژه است.
ج) n نفر را در قالب p تیم سازماندهی نموده و به هر تیم یک یا بیشتر از یک کار عملیاتی اختصاص دهیم.

عواملی که در نحوه سازماندهی تیم نرم‌افزاری موثر هستند:
۱- سبک مدیریت سازمان
۲- تعداد افراد مشارکت‌کننده در تیم
۳- سطح مهارت توانایی آن‌ها
۴- هزینه پروژه
۵- محدودیت‌های زمانی