به نام خدا

جزوه کلاسی مهندسی نرم افزار ۲ استاد صادقی

دانشجویان :

محمد نیکروش

حسن حسن زاده

الهام عباسی

(رشته نرم افزار)

دانشگاه پیام نور قم - مرکز

سال تحصیلی ۹۳-۹۲

مقدمه:

بحران نرم‌افزار Software Crisis سال ۱۹۶۰

مهندسی نرم‌افزار:

ـ چگونگی نوشتن و ایجاد یک نرم‌افزار.

ـ نقشه راه تولید نرم‌افزار بصورت بهینه و مشتری‌پسند (کاربر پسند).

مراحل ساخت و تولید نرم‌افزار:

۱. تیم تحلیل‌گر (Analyzer):

شناخت کامل و جامع نرم‌افزار شامل:

ـ مشاهده

ـ مصاحبه (ثبت در فرم)

ـ مطالعه

۲. تیم طراح (Designer):

ـ حذف اطلاعات زائد و تعیین ابهامات نرم‌افزار

ـ تبدیل اطلاعات (نرم‌افزار) به اشکال و نمودارهای مختلف UML (مستندسازی نرم‌افزار)

ـ دامنه (چارچوب)

ـ امکان‌سنجی

ـ برآورد هزینه

ـ برآورد زمان

۳. تیم کدنویسی (پیاده‌ساز):

تبدیل نقشه به نرم‌افزار با استفاده از یک زبان برنامه‌نویسی.

٤. تیم تست‌گرا (Tester)

۵. تیم استقرار:

ـ نصب نرم‌افزار در محل.

ـ آموزش کاربران.

٦. تیم طراح:

ـ زبان سنتی

ـ زبان شیءگرا

ـ زبان تحت وب

فصل دوازدهم طراحی مبتنی بر الگوها

الگوهای طراحی:

الگوهای طراحی واسط میان یک حیطه معین، یک مسئله و یک راهکار می‌باشد. در طراحی نرم‌افزار، حیطه این امکان را می‌دهد تا محیطی را که مسئله در آن جای دارد درک کند.

مشخصات الگوهای طراحی از نظر کوپلین:

۱. تعیین الگو باید موجب حل مسئله باشد.
۲. الگو مفهومی اثبات‌شده می‌باشد. (راهکارهایی را نشان می‌دهند که سابقه داشته باشد)
۳. الگوها باید روابط را مشخص کنند و ساختار نرم‌افزار را توصیف کنند.
٤. الگوها باید ساده و دخالت انسان را به حداقل برسانند.

انواع الگوها:

۱. الگوهای غیر مولد: حیطه و مسئله را توصیف می‌کنند.
۲. الگوهای طراحی: وظیفه جستجو و مستندسازی الگوها را بر عهده دارند.
۳. الگوهای معماری: ساختار و معماری نرم‌افزار را تشریح می‌کند.
٤. الگوهای مؤلفه‌ای: وظیفه آن تعیین مسائل مربوط به مؤلفه‌های نرم‌افزار می‌باشد.
۵. الگوهای طراحی واسط کاربر: مانند منوها، کلیدهای تأیید، خروج و ... .

چارچوب (Framework):

برای کار طراحی نیاز به فراهم‌آوردن زیرساختار مختص به یک پیاده‌سازی می‌باشد که چارچوب نامیده می‌شود.

تفاوت میان الگو و چارچوب:

۱. الگوی طراحی انتزاعی (زیرمجموعه) از چارچوب است.
۲. الگوی طراحی عناصر معماری کوچکتری از چارچوب هستند. یک چارچوب حاوی چند الگوی معماری است ولی عکس آن درست نیست.
۳. تخصیص‌یافتگی الگوهای طراحی کمتر از چارچوب‌هاست.

الگوهای معماری:

چگونگی سبک و ساخت نرم‌افزار را نشان می‌دهند و شامل روش‌های ذیل می‌باشد:

۱. همروندی: بسیاری از برنامه‌های کاربردی باید وظایف چندگانه را به شیوه‌ای عهده‌دار شوند که موازی کاری را شبیه‌سازی کنند.

۲. توزیع: در این قسمت شیوه‌ی برقراری ارتباط سیستم‌ها یا مؤلفه‌های درون یک سیستم با یکدیگر مشخص می‌شوند.
۳. ماندگاری: داده‌ها در صوتی ماندگار خواهند بود که پس از اجرای فرآیند در یک بانک اطلاعاتی ذخیره و یا بعدها اصلاح شوند.

الگوهای طراحی برنامه‌های تحت وب:

۱. کانون طراحی: با نزدیک‌تر شدن به چگونگی ساخت نرم‌افزار، طراحی کاملتر و باریک‌تر می‌شود. در برنامه‌های تحت وب باید از الگوهای معماری اطلاعات، گشت و گذار و تعامل و ارائه استفاده نمود.
۲. دانه‌بندی: در برنامه‌نویسی وب باید مسائل بزرگ را به قسمت‌های مختلف تقسیم کرد و هر مسئله را جداگانه پیاده‌سازی کنیم. آنگاه این مسائل را به یکدیگر پیوند بزنیم. از الگوهای معماری، طراحی و مؤلفه‌ها استفاده می‌کنند.

فصل سیزدهم طراحی برنامه‌های تحت وب

طراحی برنامه‌های تحت وب:

شامل فعالیت‌های فنی و غیر فنی که عبارتند از:

تعیین ظاهر، ایجاد چیدمان زیبایی شناختی واسط کاری.

کیفیت برنامه‌های تحت وب:

۱. قابلیت استفاده: شامل قابلیت درک سایت در تمام جهان، ویژگی‌های راهنما و بازخورد بر خط، ویژگی‌های زیبایی‌شناختی و واسط، ویژگی‌های خاص.
۲. قابلیت عملیاتی: شامل قابلیت جستجو و بازاریابی، ویژگی‌های گشت و گذار و مرورگری، ویژگی‌های مرتبط با دامنه کاربرد.
۳. قابلیت اطمینان: شامل پردازش پیوندهای صحیح، رهایی از وضعیت خط، اعتبارسنجی و بازیابی ورودی کاربر.
٤. بازدهی: شامل کارآیی زمان پاسخ، سرعت ایجاد صفحات، سرعت ایجاد گرافیک‌ها.
٥. قابلیت نگهداری: شامل سهولت تصحیح، انطباق‌پذیری، بسط پذیری.

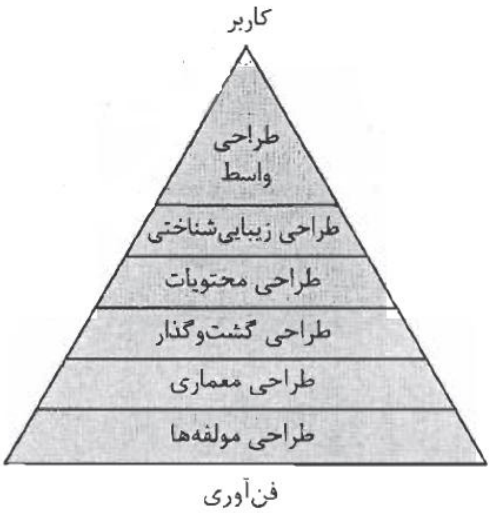
کیفیت برنامه‌های تحت وب از دیدگاه افوت:

۱. امنیت: ارائه همکاری‌هایی برای جلوگیری از حمله نفوذگران.
۲. دسترسی‌پذیری: میزانی از درصد زمان در دسترس‌بودن برنامه‌ی تحت وب برای استفاده است.
۳. گسترش‌پذیری: توانایی پاسخگویی به کاربران فراوان.
٤. زمان عرصه به بازار: از لحاظ تجاری، میزانی از کیفیت محسوب می‌شود.

اهداف طراحی:

۱. سادگی
۲. سازگاری
۳. هویت
٤. استحکام
٥. جاذبه‌ی بصری
٦. قابلیت گشت و گذار
۷. همسازمندی

هرم طراحی:



طراحی واسط:

ساختار و سازماندهی واسط کاربر را توصیف می‌کند و شامل نمایشی از چیدمان صفحه نمایش، تعریف شیوه‌های تعامل و توصیف سازوکارهای گشت و گذار می‌شود. این طراحی باعث می‌شود که بتوانیم کنترل ساختار نرم‌افزار را انجام دهیم.

طراحی زیبایی‌شناختی:

طراحی گرافیکی نامیده می‌شود، شکل و شمایل را توصیف می‌کند و شامل الگوهای رنگ چیدمان هندسی، اندازه، نوع فونت، محل متن، استفاده از تصاویر گرافیکی و ابعاد زیبایی‌شناختی می‌باشد.

طراحی محتویات:

چیدمان و ساختار و طرح‌بندی همه‌ی محتوای ارائه‌شده بعنوان بخشی از برنامه‌ی تحت وب را تعریف می‌کند و رابطه‌ی میان اشیای محتوایی و همبستگی میان آن‌ها را تعریف می‌کند.

طراحی گشت و گذار:

رابطه میان اشیای محتوایی که بعنوان بخشی از مدل خواسته‌ها برای برنامه‌ی تحت وب تعریف می‌شوند و برای کلیه قابلیت‌های عملیاتی در این طراحی انجام می‌شود.

طراحی معماری:

شامل دو بخش معماری محتوا و معماری برنامه‌ی تحت وب می‌شود. سبک‌های معماری شامل ساختارهای خطی، مشبک، سلسله مراتبی و شبکه‌ای می‌شود.

طراحی مؤلفه‌ها:

در این طراحی جزئیات منطق پردازش برای پیاده‌سازی مؤلفه‌های موجود در برنامه را شرح می‌دهد.

معماری محتوا:

این معماری شامل ساختارهای زیر می‌باشد:

طراحی محتویات:

۱. ساختار خطی: هنگامی مشاهده می‌شوند که دنباله‌ای قابل پیش‌بینی از تعامل‌ها متداول باشد.
۲. ساختارهای مشبک (کریت استراکچر): در هنگام سازماندهی محتوای برنامه‌ی تحت وب در دو بعد بکار می‌رود.

ساختار سلسله مراتبی:

متداول‌ترین معماری برنامه‌های تحت وب به شمار می‌رود که از جریان افقی و از میان شاخه‌های عمودی برای طراحی سایت استفاده می‌شود.

ساختارهای شبکه‌ای یا «وب ـ خالص»:

مشابه با بسیاری از شیوه‌های معماری است که برای سیستم‌های شیءگرا تکامل پیدا می‌کند.

معماری مدل ـ نما ـ کنترل‌گر:

یکی از چند مدل زیرساختی برای برنامه‌های تحت وب است که واسط کاربر را از قابلیت عملیاتی و متحوای اطلاعاتی آن منفک می‌سازد.

طراحی ابر رسانه‌ها:

به روش شیءگرا (OOHDM) یکی از چند روش پیشنهادشده برای طراحی برنامه تحت وب است که یک فرآیندی را پیشنهاد می‌کند که شامل طراحی مفهومی، طراحی گشت وگذار، طراحی واسط انتزاعی و پیاده‌سازی می‌شود.

فصل چهاردهم مفاهیم کیفیتی

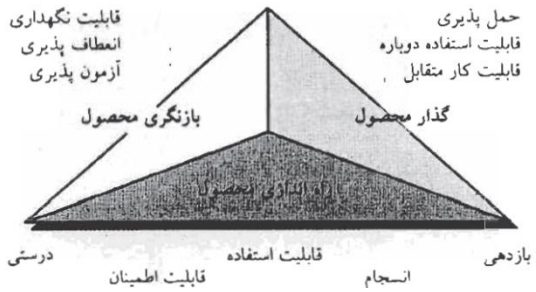
انواع کیفیت:

۱. کیفیت طراحی Design Quality : به خصوصیاتی اشاره دارد که طراح برای محصول مشخص می‌کند. کیفتی طراحی میزانی از دیده‌شدن قابلیت‌ها و ویژگی‌های مشخص‌شده در مدل خواسته توسط کیفیت طراحی است.

۲. کیفیت مطابعتی Quality Of Conformance : میزان مطابعت پیاده‌سازی از طراحی و برآورده‌شدن خواسته‌ها و اهداف کارآیی توسط سیستم کانون توجه قرار می‌گیرد.

"تحویل در زمانبندی و بودجه تعیین‌شده + کیفیت خوب + محصول مطابق با استاندارد = رضایت مشتری ←— کیفیت متابعتی"

عوامل کیفتی مک‌کال:



درستی:

حد برآورده‌شدن مشخصه‌های یک برنامه توسط آن و رسیدن به اهداف مشتری را گویند.

قابلیت استفاده:

کار لازم برای فراگیری، راه‌اندازی، آماده‌کردن ورودی و تفسیر خروجی برنامه را گویند.

قابلیت اطمینان:

حدی که می‌توان از برنامه انتظار داشت تا عملکردهای مورد نظر را با دقت لازم ارائه دهد.

انسجام:

حد کنترل دستیابی افراد غیر مجاز به نرم‌افزار یا داده‌ها.

بازدهی:

مقدار منابع کامپیوتری و کد لازم برای آنکه برنامه قادر به اجرای عملکردهای خود باشد.

قابلیت نگهداری:

کار لازم برای یافتن و تصحیح خطاهای برنامه.

انعطاف‌پذیری:

کار لازم برای اصلاح برنامه کامل‌شده.

آزمون‌پذیری:

کار لازم برای آزمودن برنامه برای اطمینان‌یافتن از اینکه عملکرد مورد نظر را به خوبی اجرا می‌کند.

حمل‌پذیری(Portability):

کار لازم برای انتقال‌دادن نرم‌افزار از یک سخت‌افزار و یا محیط سیستم نرم‌افزاری به دیگری.

قابلیت استفاده مجدد:

حدی که می‌توان یک برنامه یا بخش‌هایی از یک برنامه را دوباره در کاربردهای دیگر ـ مرتبط با پکیج‌سازی و دامنه عملیاتی که برنامه اجرا می‌کند ـ استفاده کرد.

قابلیت کار متقابل:

کار لازم برای جفت‌کردن یک سیستم به سیستم دیگر.

عوامل کیفیتی ISO ۹۱۲۶

قابلیت عملیاتی:

حدی که نرم‌افزار، نیازهای ذکرشده بر اساس این صفات را برآورده می‌کند: مناسب‌بودن، صحیح‌بودن، قابلیت کار متقابل، تطابق و امنیت.

قابلیت اطمینان:

مدت زمانیکه نرم‌افزار بر اساس این صفات در دسترس است: بلوغ، تحمل در برابر خطاها، قابلیت رهایی‌یافتن از خطا.

قابلیت استفاده:

حد سهولت استفاده از نرم‌افزار بر اساس این صفات: قابلیت درک، قابلیت فراگیری و قابلیت کار با آن.

بازدهی:

حدی که نرم‌افزار بر اساس این صفات، از منابع سیستم استفاده بهینه بعمل می‌آورد: رفتار زمانی، رفتار منابعی.

قابلیت نگهداری:

سهولت ترمیم نرم‌افزار بر اساس این صفات: تحلیل‌پذیری، تغییرپذیری، پایداری و آزمون‌پذیری.

حمل‌پذیری:

سهولت انتقال نرم‌افزار از محیطی به محیط دیگر بر اساس این صفات: تحلیل‌پذیری، تغییرپذیری، پایداری و آزمون‌پذیری.

حمل‌پذیری:

سهولت انتقال نرم‌افزار از محیطی به محیط دیگر بر اساس این صفات: تطبیق‌پذیری، ناپایداری، مطابقت، قابلیت جایگزینی.

عوامل کیفیتی هدفمند:

۱. بصیرت‌گرایی: میزان پیروی واسط از الگوهای کاربر د مورد انتظار به طوریکه حتی یک کاربر تازه‌کار بتواند بدون نیاز به آموزش زیاد از آن استفاده کند.
۲. بازدهی: میزانی از امکان‌یافتن عملیات‌ها و اطلاعات یا استفاده از آن‌ها.
۳. استحکام: میزان اداره‌کردن داده‌های ورودی بد یا تعامل نامناسب کابر توسط نرم‌افزار.
٤. غنا: میزان ارائه مجموعه‌ای غنی از ویژگی‌ها بوسیله‌ی واسط.

کنترل کیفیت:

شامل مجموعه‌ای از کنش‌های مدیریت نرم‌افزار می‌شود که به کمک آنها می‌توان اطمینان حاصل کرد که هر محصول کاری اهداف کیفیتی‌اش را برآورده ساخته است.

تضمین کیفیت:

زیرساختی را تعیین می‌کند که روش‌های مهندسی نرم‌افزار، مدیریت پروژه و کنش‌های کنترل کیفیت را پشتیبانی می‌کند.

فصل پانزدهم تکنیک‌های مرور نرم‌افزار

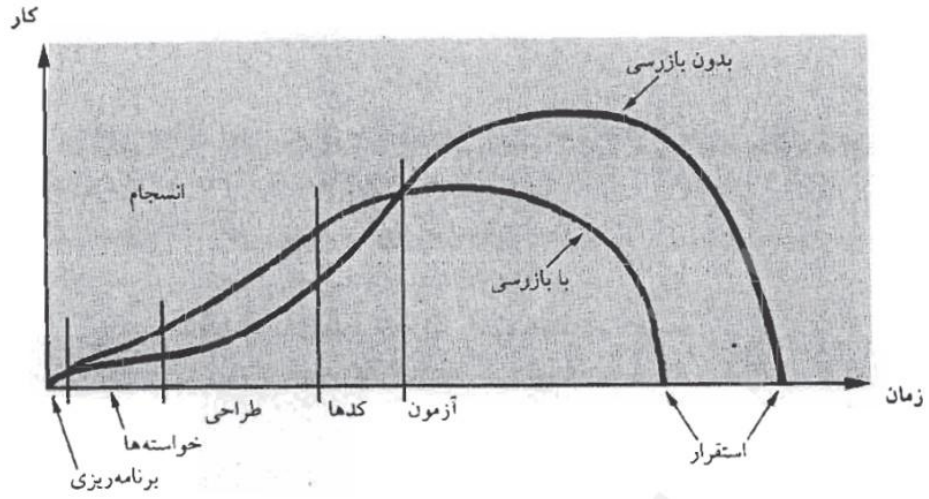
مرورهای نرم‌افزار:

اثربخش‌ترین ساز و کار برای یافتن زودهنگام خطاها در فرآیند نرم‌افزار است.
هدف اصلی آن یافتن خطاها در هنگام فرایند است بطوریکه پس از ارائه نرم‌افزار به نقص تبدیل نشود.

انواع مرورها:

۱. مرور غیر رسمی: شامل بررسی ساده رومیزی (Desk Check) یک محصول کاری مهندسی نرم‌افزار با همکاران دیگر است که منجر به یک جلسه اتفاقی برای بحث در مورد نرم‌افزار می‌شود. (نشست اتفاقی)
۲. مرور رسمی (FTR): یک جلسه سازمان‌یافته است که هدف آن بیان مشکلات و خطاهای نرم‌افزار و ارائه راه‌حل برای آنهاست. در مرور فنی رسمی باید کارهای ذیل انجام گردد:

- نشست مرور: در این جلسات تعداد ۳-۵ نفر حضور داشته و مدت زمان جلسه نباید بیشتر از ۲ ساعت شود. در پایان جلسه حاضران باید تصمیم بگیرند که باید محصول کاری را بدون اصلاح بپذیرند یا خطاهای نرم‌افزار را نادیده بگیرند.
- گزارش مرور: پس از انتهای نشست باید گزارش خلاصه‌ای از مرورهای فنی رسمی ارائه‌شده و راه‌حل‌های مناسب ارائه گردد سپس به امضاء حاضرین برسد. دستورالعمل‌های مربوط به اجرایء مرورها را باید از پیش وضع و در میان تمامی مسئولان بازرسی توزیع نمود، آنها را به تصویب رساند و سپس رعایت کرد.
- مرورهای نمونه‌محور: در فرآیند مرور نمونه‌محور برای اثربخشی باید تلاش به عمل آید که محصول کاری هدف اولیه برای مرور باشد و کارهایی بیش از همه استعداد خطا دارند در اولویت قرار گیرند.



معیارهای مرور:

۱. تلاش آماده‌سازی (EP): تلاش لازم برای مرور یک محصول قبل از جلسه‌ی مرور واقعی.
۲. تلاش ارزیابی (Ea): تلاش حذف‌شده طی مرور واقعی.
۳. تلاش دوباره‌کاری (Er): تلاش اختصاص داده‌شده به تصحیح آن دسته از خطاهایی که طی مرور کشف می‌شود.
٤. اندازه‌ی محصول کاری (WPS): میزانی از اندازه‌ی محصول که مرور شده است.
٥. خطاهای جزئی یافته‌شده (Err_{minor}) تعداد خطاهای یافته‌شده که می‌توان آنها را در خطاهای جزئی دسته‌بندی کرد.
٦. خطاهای عمده‌ی یافته‌شده (Err_{major}): تعداد خطاهای یافته‌شده که می‌توان آنها را در خطاهای عمده دسته‌بندی کرد.

$E_{review} = E_p + E_a + E_r$

$Err_{tot} = Err_{minor} + Err_{major}$

چگالی خطا = $\frac{Errtot}{WPS}$

مثال: در یک نرم‌افزار یک مجموعه مستندات ۳۲ صفحه‌ای حاوی ۱۸ نمودار UML است، پس از مرور، ۱۸ خطای جزئی و ٤ خطای عمده کشف شد.

$Err_{tot} = Err_{minor} + Err_{major}$ ➡ ۱۸ + ٤ = ۲۲

WPS = ۱۸

چگالی خطا = $\frac{۲۲}{۳۲}$ = ۰.٦۸

چگالی خطا نمودارها = $\frac{۲۲}{۱۸}$ = ۱.۲

فصل شانزدهم تضمین کیفیت نرم‌افزار

تضمین کیفیت نرم‌افزار:

یکی از فعالیت‌های چتری است که در هر مرحله نرم‌افزار اجرا می‌شود و شامل موارد ذیل است:

۱. تعریف وظایف: تضمین کیفیت و کنترل کیفیت نرم‌افزار است و همچنین برای کنترل تغییرات نیز دستورالعمل‌هایی دارد.

عناصر تضمین کیفیت نرم‌افزار SQR:

۱. استاندارد.
۲. مرورها و ممیزی‌ها.
۳. آزمون: یک هدف را دنبال می‌کند و آن یافتن خطاهاست.
٤. مدیریت تغییرات: تغییر یکی از مخرب‌ترین جنبه‌های هر پروژه نرم‌افزاری است.
۵. جمع‌آوری و تحلیل خطاهای نقایص: تنها راه بهبودبخشیدن، سنجیدن عملکرد است.
٦. آموزش: هر سازمان نرم‌افزاری مایل است کارهای مهندسی نرم‌افزار خود را بهبود بخشد.
۷. مدیریت منابع.
۸. مدیریت امنیت.
۹. ایمنی.
۱۰. مدیریت ریسک.

شش سیگما:

پرکابردترین راهبرد برای تضمین کیفیت آماری در صنعت کنونی است و سه مرحله اصلی دارد:

۱. تعریف خواسته‌های مشتری.
۲. اندازه‌گیری فرآیند ورودی و خروجی.
۳. تحلیل معیارهای نقص نرم‌افزار.

قابلیت اطمینان نرم‌افزار:

احتمال عملکرد بدون شکست یک برنامه در محیطی مشخص برای یک زمان معین.

قابلیت اطمینان = زمان میانگین شکست + زمان میانگین ترمیم

MTBF = MTTF + MTTR

دسترس‌پذیری نرم‌افزار:

عبارت است از احتمال کارکردن رسایل طبق خواسته‌ها در یک نقطه‌ی مشخص از زمان.

$$\text{دسترس‌پذیری} = \frac{MTTF}{MTTF+MTTR} \times ۱۰۰$$

ایمنی نرم‌افزار: یکی از فعالیت‌های تضمین کیفیت است که به شناسایی و سنجش ریسک‌های بالقوه تأکید دارد. ممکن است تأثیر منفی در نرم‌افزار داشته منجر به شکست کل سیستم می‌شود.

فصل هفدهم راهبردهای آزمون نرم‌افزار

آزمون:

مجموعه‌ای از فعالیت‌هاست که می‌توان از قبل برنامه‌ریزی کرد و بطور سیستماتیک اجرا نمود.

وارسی:

عبارتست از یک مجموعه فعالیت‌ها که پیاده‌سازی صحیح یک عملکرد خاص توسط نرم‌افزار را تضمین می‌کند.
فارسی ← آیا محصول را درست ساخته‌ایم؟

اعتبارسنجی:

عبارت است از مجموعه‌ای متفاوت از فعالیت‌ها که تضمین می‌کند نرم‌افزار ساخته‌شده با خواسته‌های مشتری مطابقت دارد. فارسی ← آیا محصول درست را ساخته‌ایم؟

راهبردهای آزمون نرم‌افزار:

آزمون واحدها در مرکز این مارپیچ قرار دارد که بر هر واحد از نرم‌افزار که در کد منبع پیاده‌سازی می‌شوند تمرکز دارد. آزمون با حرکت به طرف بیرون مارپیچ ادامه یافته و به آزمون انسجام می‌رسد. در آن به طراحی و ایجاد و ایجاد معماری نرم‌افزار تأکید می‌شود. با یک دور دیگر پیش رفتن به آزمون اعتبارسنجی می‌رسیم که در آن خواسته‌های مشتری بررسی می‌شود. سرانجام به آزمون سیستم می‌رسیم که در آن کل نرم‌افزار آزموده می‌شود.

آزمون واحدها:

تلاش‌های وارسی مربوط به کوچکترین واحد طراحی نرم‌افزار یعنی مؤلفه‌ها و پیمانه‌ها را کانون توجه قرار می‌دهد. وظایف این آزمون شامل:

۱. ارزیابی صحیح اطلاعات.

۲. اطمینان از عملکرد مناسب و درست نرم‌افزار.
۳. اجرای تمام آزمون‌های جعبه سفید.

در آزمون واحدها، مسیرهای اجرایی اهمیت ویژه‌ای دارند. این مسیرها شامل حلقه‌ها، شروط و فراخوانی هستند.

آزمون مرزی:

آخرین وظیفه در مرحله آزمون واحدهاست که مرزهای سیستم را کنترل می‌کند.

آزمون انسجام:

تکنیک سیستماتیک برای ایجاد ساختار برنامه و در عین حال اجرای آزمون‌هایی جهت کشف خطاها در ایجاد واسطه‌هاست.

انواع آزمون انسجام:

۱. انسجام بالا به پائین: یک روش تدریجی برای ایجاد ساختار برنامه است. پیمانه‌ها با حرکت به طرف پائین در سلسله مراتب کنترلی و با شروع از پیمانه اصلی مجتمع می‌شوند.
۲. انسجام پائین به بالا: ساخت و آزمون پیمانه‌ها را با پیمانه‌های ساده آغاز می‌کنند.

آزمون رگرسیون:

هر بار که یک پیمانه جدید بعنوان بخشی از آزمون انسجام افزوده می‌شود، نرم‌افزار تغییر می‌کند و مسیرهای جدید برقرار می‌شود و نیاز به ورودی ـ خروجی جدید دارد. این آزمون عبارت است از اجرای دوباره زیرمجموعه‌ای از آزمون‌ها که قبلاً اجرا شده‌اند تا اطمینان حاصل شود که تغییرات باعث انتشار اثرات جانبی شده است.

آزمون دود:

یک روش آزمون انسجام است که بطور متداول هنگام توسعه محصولات نرم‌افزاری بسته‌بندی‌شده بکار می‌رود.
نکته: آزمون واحدها: در نرم‌افزار شیءگرا در مباحث تعریف کلاس‌ها و اشیاء شناخته می‌شود و همچنین آزمون انسجام بالا به پائین و پائین به بالا در نرم‌افزار شیءگرا معنا ندارد. بنابراین از آزمون نخ‌ها و آزمون خوشه‌ای استفاده می‌شود.

آزمون اعتبارسنجی:

هنگامیکه تک‌تک مؤلفه‌ها بطور کامل مونتاژ شده‌اند و خطاهای آنها برطرف شد، اعمال می‌شوند.

۱. آزمون آلفا: در مکان سازنده‌ی نرم‌افزار توسط مشتری ساخته می‌شود. نرم‌افزار در شرایطی طبیعی انجام می‌شود و سازنده ناظر بر اجرای آزمون است و خطاها را ثبت می‌کند. این آزمون در محیطی کنترل‌شده است.
۲. آزمون بتا: یک کاربرد زنده از نرم‌افزار در محیطی است که سازنده قادر به کنترل آن نیست. مشتری مشکلات را ثبت و به سازنده گزارش می‌دهد.

آزمون سیستم:

هدف اصلی آن امتحان کل سیستم است و تلاش می‌کند تا عملکردهای سیستم درست انجام شود.

۱. آزمون ترمیم: نرم‌افزار را به طرق گوناگون وادار به شکست و سپس در مورد اجرای مناسب ترمیم تحقیق می‌کنند. اگر ترمیم بصورت خودکار باشد، چگونگی ترمیم را ارزیابی می‌کند. اگر ترمیم نیاز به دخالت انسان داشته باشد، زمان میانگین ترمیم تعیین می‌شود.
۲. آزمون امنیت: کوشش می‌کند (وارسی می‌کند) که راهکارهای محافظ تعبیه‌شده در داخل سیستم واقعاً آن را از نفوذ نامناسب حفظ می‌کند.
۳. آزمون فشار: سیستم را به شیوه‌ای اجرا می‌کند که منابع را به میزان غیر عادی و حجم غیر عادی طلب کند. نام دیگر این آزمون، آزمون حساسیت است.
٤. آزمون کارآیی: این آزمون به منظور کارآیی نرم‌افزار در زمان اجرا طراحی می‌شود.
۵. آزمون استقرار (پیکره‌بندی): در این آزمون همه مراحل نصب انجام شده و مستندات لازم برای معرفی نرم‌افزار به کاربران نهایی داده می‌شود.

هنر اشکال‌زدایی:

آزمون نرم‌افزار فرآیندی است که می‌توان آن را بطور سیستماتیک برنامه‌ریزی و مشخص نمود، اشکال‌زدایی در نتیجه آزمون موفق رخ می‌دهد و یک فرآیند منظم برای رفع خطا می‌باشد.

فرآیند اشکال‌زدایی تلاش می‌کند تا خطاهای سیستم را بدون آنکه خللی در قسمت‌های دیگر نرم‌افزار بوجود آید برطرف می‌کند.

سه روش اشکال‌زدایی پیشنهاد می‌گردد:

۱. جستجوی جامع
۲. عقب‌گرد
۳. حذف علت

فصل هجدهم آزمون برنامه‌های کاربردی سنتی

مبانی آزمون نرم‌افزار:

هدف آزمون یافتن خطاهاست، آزمون خوب آزمونی است که احتمال یافتن خطا را بالا می‌برد، لذا در طراحی و پیاده‌سازی نرم‌افزار طوری باید اعمال کرد که نرم‌افزار آزمون‌پذیری داشته باشد.

خصوصیات ذیل به نرم‌افزار آزمون‌پذیر منجر می‌شود:

۱. قابلیت کارکردن
۲. قابلیت مشاهده
۳. کنترل‌پذیری
٤. تجزیه‌پذیری
٥. سادگی
٦. پایداری
۷. درک‌پذیری

خصوصیات آزمون‌ها:

۱. با احتمال زیاد خطاها را کشف کنند.
۲. آزمون دارای زواید نباشد.
۳. بهترین آزمون انتخاب شود.

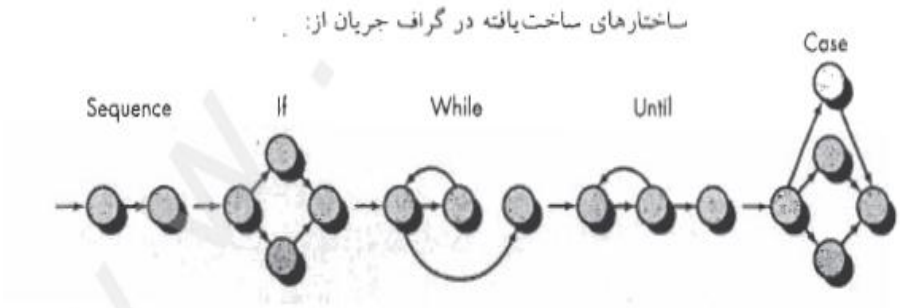
آزمون جعبه سفید (شیشه‌ای):

روش طراحی برای بدست‌آوردن خطاها از ساختار کنترلی برنامه است که تضمین می‌کنند که همه مسیرهای مستقل در یک پیمانه حداقل یکبار امتحان شده‌اند و همچنین همه تصمیم‌گیری‌های منطقی را در دو بخش درست و غلط امتحان می‌کنند و همه حلقه‌ها را در مرزها و در داخل مرزهای عملیاتی اجرا می‌کنند و ساختمان داده‌های داخلی را امتحان می‌کنند تا اعتبار آنها ثابت شود.

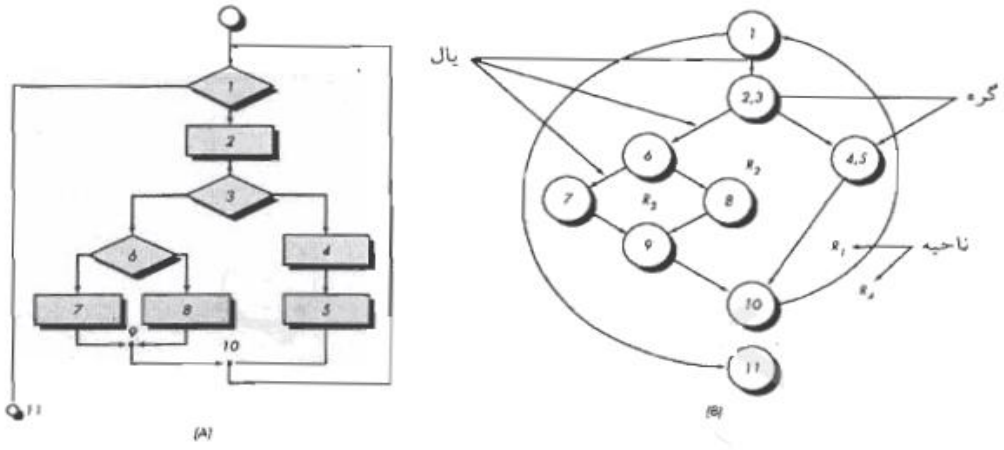
آزمون مسیریابی:

این آزمون با استفاده از گراف‌های برنامه مسیرهای مشخص نرم‌افزار را پیدا کرده و صحت کارکرد آنها را تضمین می‌کند، برای اینکار مراحل ذیل را انجام می‌دهند:

الف) نمادگذاری گراف جریان:



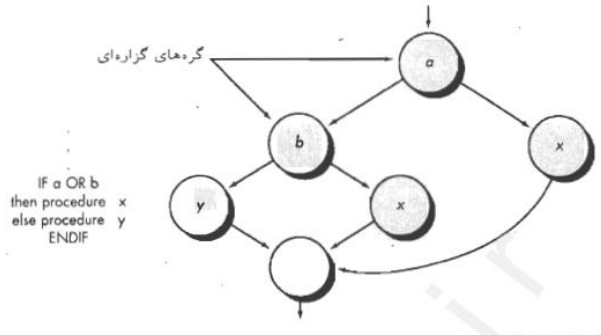
هر دایره نشان‌گر یک یا چند PDL یا دستور کد منبع است



(الف) نمودار گردشی (ب) گراف جریان.

ب) مسیرهای مستقل برنامه: مسیر مستقل مسیری است که حداقل یک مجموعه جدید از دستورهای پردازش یا یک دستور شرطی را معرفی می‌کند.

پیچیدگی سیکلوماتیک: ریشه در نظریه گراف‌ها دارد و یک معیار نرم‌افزاری سودمند را فراهم می‌کند.



شکل ۳-۱۸ منطق مرکب.

سه روش برای پیچیدگی سیکلوماتیک

۱. تعداد نواحی گراف جریان متناظر با پیچیدگی سیکلوماتیک.
۲. پیچیدگی سیکلوماتیک، V (G)، برای یک گراف جریان G بصورت V(G) = E – n + ۲ تعریف می‌شود که در آن E تعداد یال‌های گراف جریان و N تعداد گره‌های آن است.
۳. پیچیدگی سیکلوماتیک، V (G)، برای یک گراف جریان G بصورت V(G) = P + ۱ نیز تعریف می‌شود که در آن P تعداد گره‌های گزاره‌ای موجود در گراف جریان G است.

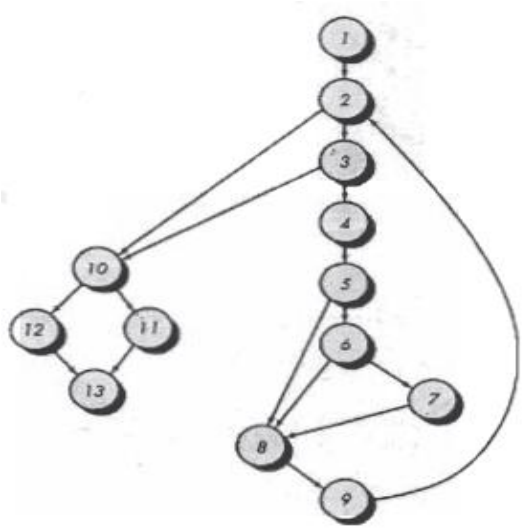
به عنوان مثال: مسیرهای مستقل ذیل در شکل فوق «ب) گراف جریان»:

۱-۱۱ ۱-۲-۳-٤-٥-۱۰-۱-۱۱ ۱-۲-۳-٦-۸-۹-۱۰-۱-۱۱ ۱-۲-۳-٦-۷-۹-۱۰-۱-۱۱

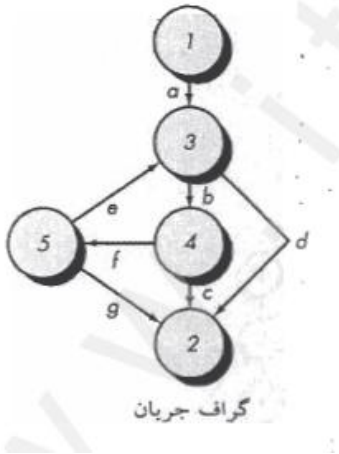
نواحی: V(G) = ٤ V(G) = ۱۱ – ۹ + ۲ = ٤

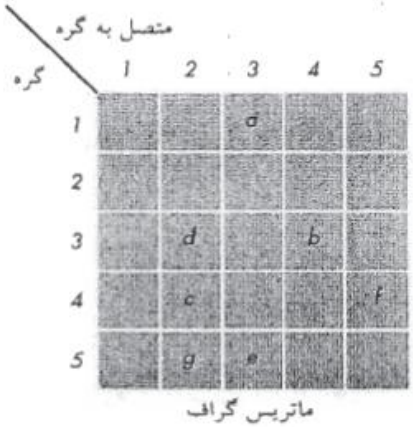
مثال:

V (G) = ٥ + ۱ = ٦



ج) ماتریس گراف:





آزمون ساختار کنترلی:

این آزمون تمامی شرط‌ها و حلقه‌های موجود در برنامه ار تست می‌کند.
الف) آزمون شرط‌ها: این آزمون تمامی شرط‌های منطقی موجود در یک برنامه را امتحان می‌کند.
ب) آزمون جریان داده‌ها: در این روش مسیرهای آزمون یک برنامه طبق موقعیت تعاریف و کاربردهای متغیرها انتخاب می‌شود.
ج) آزمون حلقه‌ها: بر اعتبار ساختمان حلقه تکیه دارد.

آزمون جعبه سیاه (رفتاری):

این آزمون بر خواسته‌های عملیاتی نرم‌افزار تکیه دارد.

۱. افزار هم‌ارزی: یکی از روش‌های آزمون جعبه سیاه است که دامنه‌های ورودی یک برنامه را به دسته‌هایی از داده‌ها تقسیم می‌کند و ورودی‌های مختلفی را اعمال می‌کند.
۲. آزمون آرایه‌های متعامد دامنه: یعنی تعداد پارامترهای ورودی و مقادیری که هر یک از پارامترها به خود می‌گیرند مورد آزمون واقع می‌شود.

آزمون محیط‌ها، معماری‌ها و برنامه‌های کاربردی تخصص‌یافته:

الف) آزمون واسط گرافیکی کاربرد (GUI): در این آزمون محیط نرم‌افزار و چگونگی شکل و شمایل آن بررسی می‌شود.
ب) آزمون معماری‌های کلاینت ـ سرور: در این آزمون نرم‌افزار از لحاظ شبکه‌ای و ارتباطات آن با بانک اطلاعاتی بررسی می‌شود.
ج) آزمون مستندات راهنما: خطاهای موجود در راهنمای برنامه کشف و رفع می‌شود.
د) آزمون سیستم‌های بلادرنگ: پاسخگویی نرم‌افزار به تراکنش‌ها از لحاظ زمانی در این آزمون بررسی می‌شود.

فصل نوزدهم آزمون برنامه‌های شیءگرا

آزمون‌های شیءگرا:

از نظر راهبردی مشابه آزمون سیستم‌های سنتی است ولی از لحاظ تاکتیکی تفاوت دارد، چون مدل‌های شیءگرا از لحاظ ساختاری متفاوت می‌باشند.
هنگامیکه کدها نوشته می‌شود آزمون شیءگرا بصورت کوچک با آزمون کلاس‌ها شروع می‌شود، آزمون‌های بعدی بررسی می‌کنند که آیا در ارتباط کلاس با کلاس دیگر خطا وجود دارد یا خیر، پس از اجتماع کلاس‌ها، آزمون نخ، آزمون مبتنی بر کاربرد و آزمون خوشه‌ای اعمال می‌شود.

سازگاری مدل‌های شی‌گرا:

* برای ارزیابی، سازگاری هر کلاس و اتصالات آن با کلاس‌های دیگر را باید بررسی کرد.
{مدل کلاس، مسئولیت همکاری} (CRC) و نمودار روابط میان اشیاء برای انجام این فعالیت استفاده می‌شود.
برای ارزیابی مدل کلاس‌ها مراحل زیر توصیه می‌گردد:

۱. مرور مدل CRC و مدل رابطه میان اشیاء.
۲. بازرسی توصیف هر یک از کارت‌های شاخص CRC برای تعیین این که مسئولیت‌ها درست تعریف شده است.
۳. معکوس کردن اتصال برای اطمینان از اینکه همکاری و روابط درست تعریف شده است.
٤. ادغام برخی مسئولیت‌ها در کلاس‌های دیگر.

راهبردهای آزمون نرم‌افزار:

آزمون واحد: در نرم‌افزارهای شیءگرا مفهوم واحد فرق می‌کند، شامل واحد کلاس‌ها، اشیاء و بسته‌بندی است. این بدان معناست که هرکلاس و هر نمونه از صفات و رفتار آن‌ها را دستکاری می‌کند و ممکن است صفات و رفتار اضافی را حذف و یا اضافه کند.
آزمون انسجام: از آنجائیکه نرم‌افزارهای شیءگرا فاقد ساختار کنترلی سلسله مراتبی‌اند، راهبردهای بالا به پایین و پائین به بالا چندان معنایی ندارند، لذا از آزمون‌های زیر استفاده می‌گردد:

۱. آزمون نخ‌ها: مجموعه‌ای از کلاس‌های لازم برای پاسخ‌دهی به یک ورودی یا رویداد سیستم مجتمع می‌کند. هر نخ بطور انفرادی مجتمع و آزموده می‌شود.

۲. آزمون مبتنی بر کاربرد: ساخت سیستم را با آزمون آن دسته از کلاس‌هایی آغاز می‌کند که از تعداد کلاس‌های اندکی استفاده می‌کند. پس از آنکه کلاس‌های مستقلی آزمون شدند لایه بعدی کلاس‌ها که دارای کلاس‌های مستقل هستند مورد آزمون قرار می‌گیرند.
۳. آزمون خوشه‌ای: در اینجا خوشه‌ای از کلاس‌های همکار با طراحی آزمونی که سعی در کشف خطاهای موجود در مشارکت‌ها دارند امتحان می‌شود.
٤. آزمون اعتبارسنجی: کارکرد کلی نرم‌افزار را از طریق آزمون‌های جعبه سیاه می‌آزمایند.
۵. آزمون مبتنی بر خطا: هدف از این آزمون طراحی آزمونی است که احتمال بالایی در کشف خطاهای ممکن دارد. در این آزمون ممکن است به خطاهای زیر برخورد کنیم:
- نتیجه غیر منتظره
- استفاده از پیام یا عملیات اشتباه
- فراخوانی نادرست
٦. آزمون ساختار سطحی و ساختار عمیق: ساختار سطحی به ساختار بیرونی و قابل مشاهده برنامه شیءگرا اشاره دارد. یعنی ساختاری که بلافاصله در معرض دید کاربر نهایی است. ساختار عمیق به جزئیات داخلی برنامه شی‌گرا می‌پردازد. یعنی ساختاری که با بررسی طراحی و یا کد درک می‌شود. این آزمون برای امتحان‌کردن وابستگی‌ها، رفتارها و سازوکارهای ارتباطی که بعنوان بخشی از طراحی سیستم و اشیاء در نرم‌افزار شیءگرا وضع می‌شوند، طراحی می‌گردد.

فصل بیستم آزمون برنامه‌های کاربردی تحت وب

آزمون برنامه‌های کاربردی تحت وب:

آزمون‌های تحت وب نیز بدنبال یافتن خطاهاست تا تلاش کند کیفیت یک نرم‌افزار را بالا ببرد. آنچه در این آزمون‌ها کانون توجه قرار می‌گیرد؛ محتوا، قابلیت‌های عملیاتی، ساختار، قابلیت گشت و گذار و امنیت است. راهبرد این آزمون‌ها همه ابعاد کیفیت را باید درنظر بگیرد.

انواع آزمون‌ها:

۱. آزمون محتوا (Content): در این آزمون تلاش می‌شود که مشکلات نحوی و معنایی مورد توجه قرار گیرد و سه هدف را دنبال می‌کند:
   - کشف خطاهای نحوی و گرامری.
   - کشف خطاهای معنایی و محتوایی.
   - یافتن خطاهای موجود در سازماندهی.

الف) آزمون نحوی و معنایی.

ب) آزمون بانک اطلاعاتی: در این آزمون‌ها، سیستم‌های مدیریت بانک‌های اطلاعاتی و نوع رابطه بین جداول بانکی مورد بررسی قرار می‌گیرد.

۲. آزمون واسط کاربر (Inter Face): در این آزمون مدل واسط مرور می‌شود تا اطمینان حاصل شود که با خواسته‌ها همخوانی دارد. راهبردی که درنظر گرفته می‌شود شامل:
   - کشف خطاهای مرتبط با سازوکارهای ویژه.
   - کشف خطاهای موجود در روش پیاده‌سازی و گشت و گذار.

الف) آزمون معناشناختی واسط: تعیین می‌کند که طراحی تا چه حد مورد نظر کاربران است و با آنها سازگاری دارد و بازخوردهایی را تحویل می‌دهد.

ب) آزمون قابلیت استفاده: در این آزمون میزان اثربخشی تعامل کاربران با برنامه تحت وب را ارزیابی می‌کند.

ج) آزمون سازگاری: در این آزمون صفحات وب بر روی سیستم عامل‌ها و مرورگرهای مختلف تست و ارزیابی می‌شود تا خطاهای آنها پیدا شود.

۳. آزمون در سطح مؤلفه‌ها: مجموعه‌ای از آزمون‌ها که سعی در کشف خطاهای موجود در توابع برنامه‌های تحت وب دارد. این آزمون با ورود به پیاده‌سازی سیستم مسیرها و مقادیر فرضی را تست می‌کند.

٤. آزمون گشت و گذار: این آزمون اطمینان می‌دهد که کاربر به درستی می‌تواند در صفحه وب گشت و گذار کند. نخستین مرحله از این آزمون، اجرای آزمون واسط است. همچنین این آزمون تمامی پیوندهای موجود در وب و مسیرها را ارزیابی می‌نماید.
٥. آزمون پیکربندی: در این آزمون چگونگی برقراری برنامه با سرور و انتقال اطلاعات تست می‌شود.
٦. آزمون امنیت: برنامه از لحاظ نفوذ و آسیب‌پذیری ارزیابی می‌شود تا حفره‌های آن کشف شود. برای محافظت در مقابل این آسیب‌پذیری‌ها، یک یا چند عنصر امنیتی پیاده‌سازی می‌شود که شامل:

- دیوار آتش
- احراز هویت
- پنهان‌سازی
- اخذ مجوز

٧. آزمون کارایی: این آزمون برای تعیین اینکه محیط برنامه چه مقدار ظرفیت را می‌تواند کنترل کند استفاده می‌شود.

الف) آزمون ازدحام بار: هدف این آزمون تعیین چگونگی پاسخ‌دهی برنامه تحت وب و محیط سرور به انواع شرایط ازدحام است. توان عملیاتی سایت از فرمول زیر محاسبه می‌شود:

$P = N \times T \times D$

- N (تعداد کاربران زمان)
- T (تعداد تراکنش‌های آنلاین در واحد زمان)
- D (ازدحام بار داده توسط سرور به ازای هر تراکنش)

مثال: یک سایت خبری ورزشی پر طرفدار در یک لحظه بیست‌هزار کاربر همزمان به طور میانگین هر دو دقیقه یک درخواست تسلیم می‌کنند، هر تراکنش مستلزم آن است که برنامه تحت وب، مقاله جدیدی به طول میانگین سه کیلو بایت دانلود کند، توان عملیاتی آن را حساب کنید؟

$N = ۲۰,۰۰۰$

$T = \frac{۱}{۱۲۰}$

$D = ۳\ Kb$

$P = ۲۰,۰۰۰ \times \frac{۱}{۱۲۰} \times ۳ = ۵۰۰\ kb/s$

ب) آزمون فشار: این آزمون، ادامه آزمون ازدحام بار است که ظرفیت برنامه، تجهیزات، سرورها و تراکنش‌ها را ارزیابی می‌کند.

فصل بیست و یکم مدلسازی و وارسی رسمی

مدل‌سازی و وارسی رسمی:

مهندسی نرم‌افزار اتاق تمیز یک رویکرد رسمی در توسعه رویکردهاست که می‌تواند منجر به تولید نرم‌افزار با کیفیت شود. در این رویکرد از مشخصه‌های ساختاری چهارگوش برای مدلسازی طراحی و تحلیل استفاده می‌شود که همان لحظه خطاها را پیدا و حذف می‌کند و بر آزمون تأکید ندارد.

رویکرد اتاق تمیز با مدل‌های تحیل و طراحی آغاز می‌شود که از نمایش ساختارهای چهارگوش در آن استفاده می‌شود. هر چهارگوش سیستم را در سطح معینی پنهان‌سازی می‌کند و چهارگوش‌های سیاه‌رنگ رفتار را نشان می‌دهد. چهارگوش‌های حالت، حالت اشیاء را نشان داده و چهارگوش شفاف، مدلسازی را نشان می‌دهد.

وارسی هنگامی بکار می‌رود که طراحی ساختارهای چهارگوش کامل باشد. هنگامیکه وارسی تمام شد، آزمون کاربردی آماری آغاز می‌شود.

زبان قید و بند اشیاء (OCL): یک نمادگذاری رسمی است که طوری توسعه یافته است که کاربران UML می‌توانند دقت بیشتری به مشخصات خود بیافزایند. همه‌ی قدرت منطق و ریاضیات گسسته در این زبان در دسترس قرار دارد.

فصل بیست و دوم مدیریت پیکربندی نرم‌افزار

یک فعالیت چتری است که در سرتاسر فرآیند نرم‌افزار اجرا می‌شود و وظیفه آن شناسایی، کنترل، ممیزی (SCM) و گزارش اصلاحاتی است که پس از ارائه نرم‌افزار به مشتری رخ می‌دهد. پیکربندی به شیوه‌ای سازمان‌دهی می‌شود که کنترل منظم تغییر را میسر می‌سازد.

خروجی فرآیند نرم‌افزار، اطلاعاتی است که به سه گروه عمده تقسیم می‌شود:

– برنامه‌های کامپیوتری. – مستنداتی که این برنامه‌های کامپوتری را توصیف می‌کنند. – داده‌ها.

❋ خط مبنا:

تغییر حقیقت انکارناپذیری در توسعه نرم‌افزار است. مشتریان خواسته‌ها را اصلاح کنند، بنابراین باید بدانیم کدام قسمت نرم‌افزار را تغییر دهیم. خط مبنا یک مفهوم مدیریت پیکربندی نرم‌افزار است که به کنترل تغییرات کمک می‌کند.

مخزن SCM : مجموعه‌ای از سازوکارها و ساختمان داده‌هاست که به تیم نرم‌افزاری این امکان را می‌دهد تا تغییرات را به شیوه‌ای اثربخش مدیریت کند.

فرآیند SCM : برای اجرای فرآیند SCM باید مراحل زیر را انجام دهیم:

۱. شناسایی اشیاء در پیکربندی نرم‌افزار: در این مرحله باید اشیاء و آیتم‌های مورد نظر را به دقت شناسایی و جدا کرد.
۲. کنترل نسخه‌ها: یک فعالیت رویه‌ای است که تضمین کیفیت را به موازات اعمال تغییرات در یک شیء پیکربندی را بر عهده دارد و تمام تغییرات را ثبت و شماره‌گذاری می‌کند.
۳. کنترل تغییرات: درخواست تغییر پس از تسلیم مورد ارزیابی قرار می‌گیرد تا شایستگی فنی، اثرات جانبی بالقوه و تأثیرات کلی آن بر دیگر اشیاء سنجیده می‌شود، نتایج این ارزیابی بصورت یک گزارش تغییر ارائه می‌شود که مورد استفاده مسئول کنترل تغییر (CCA) قرار می‌گیرد که تصمیم نهایی با این مسئول است. برای هر تغییر مصوب یک سفارش تغییر مهندسی (ECO) تولید می‌شود، ECO تغییری را که باید اعمال شود محدودیت آن و چگونگی تغییر آن را شرح می‌دهد. اشیائی که قرار است تغییر داده شوند توسط مهندسی مربوطه تغییر داده می‌شوند و سپس توسط مسئول کنترل نسخه بروز می‌شود.
٤. ممیزی پیکربندی: شناسایی، کنترل نسخه و کنترل تغییرات به نرم‌افزارنویس کمک می‌کند تا نظم امور را حفظ کند.
۵. گزارش وضعیت: یک فعالیت SCM است که در آن گزارش تغییرات و اتفاقات رخ‌داده ثبت می‌شود.

فصل بیست و سوم معیارهای محصول

معیارهای محصول:

معیارهای نرم‌افزار یک راه کمی برای ارزیابی کیفیت صفات داخلی محصول بدست می‌دهد و امکان می‌سازد تا کیفیت را پیش از تولید محصول ارزیابی کند. معیارها دید لازم برای ایجاد مدل‌های تحلیل، طراحی، کدها و آزمون‌ها فراهم می‌کند. معیارهای نرم‌افزاری برای اینکه در جهان واقعی مفید باشد باید ساده و قابل محاسبه و سازگار باشد.

معیارهایی برای مدل خواسته‌ها:

مهندسی نرم‌افزار با تحلیل آغاز می‌شود، در این مرحله خواسته‌های مشتری بدست می‌آید، برای کنترل این خواسته‌ها از معیار زیر استفاده می‌گردد:

معیار مبتنی بر عملکرد FP : بر اساس رابطه تجربی تعداد ورودی‌ها، تعداد خروجی‌ها، تعداد درخواست‌ها، تعداد فایل‌ها و تعداد فایل‌های واسطه خارجی اندازه‌گیری می‌شود.

$FP = \text{شمارش کل} \times [۰,٦٥ + ٪۱ \times ٤(fi)]$

در معیارهای مربوط به طراحی جنبه‌های ساختاری مدل طراحی مدنظر قرار می‌گیرد. در این معیارها یک پارچگی، اتصال، پیچیدگی و کیفیت سنجیده می‌شود.

معیارهای مربوط به سیستم‌های شیءگرا اندازه‌گیری‌هایی را کانون توجه قرار می‌دهد که می‌توان آنها را در خصوصیات کلاس‌ها بکار برد. این خصوصیات عبارتند از:

- محلی‌سازی.
- پنهان‌سازی اطلاعات.
- وراثت.

.

.

.

و من ا... توفیق.