به نام خدا

میانترم اول ـ پاسخنامه

مهندسی نرم‌افزار – نیمسال اول ۴۰۲

- تاریخ آزمون: پنج‌شنبه ۹ / ۹ / ۴۰۲
- زمان شروع: ۹:۰۰ صبح
- زمان پایان: ۱۴:۰۰ ظهر

دانشکده مهندسی کامپیوتر – دانشگاه شریف

مدرس: دکتر مهران ریواده

بخش اول – سوالات تستی

۱- کدام یک از گزینه‌های زیر در ارتباط با ایجاد نرم‌افزار بر اساس مدل V صحیح است؟

a) صرفاً می‌تواند در بعضی از روش‌های چابک استفاده شود.

b) تمرکز خوبی روی تست دارد و به همین جهت، مورد علاقه Testerها است.

c) در این مدل، تست‌ها می‌توانند به طور مستقل از هم انجام شوند و هیچ گونه وابستگی به هم ندارند.

d) مانند مدل آبشاری، روشی قدیمی است و دیگر در صنعت کاربردی ندارد.

(صفحه ۴۲ و تصویر صفحه‌ی ۴۳_بخش ۴.۱.۱_کتاب پرسمن ادیشن ۸)

۲- کدام یک از موارد زیر، در مورد Extreme Programming (XP) صحیح نیست؟

a) توجه ویژه‌ای بر Pair Programming دارد.

b) از کارت‌های CRC برای طراحی استفاده می‌کند.

c) در هر چرخه، شامل فعالیت‌های کلیدی Planning, Design, Coding, Testing را دارد.

d) برخلاف اسکرام، جلسات Planning ندارد و تمرکز آن فقط بر روی Development است.

(صفحه ۷۲ تا ۷۵_بخش ۵.۴.۱_کتاب پرسمن ادیشن ۸)

۳- در زمینه سناریوهای سیستم، چه تعداد از موارد زیر معمولاً شامل می‌شود؟

- شرحی از آنچه سیستم و کاربران هنگام شروع سناریو انتظار دارند.
- شرحی از جریان عادی رویدادها در سناریو
- شرحی از مواردی که ممکن است اشتباه پیش برود و چگونه می توان با مشکلات ناشی از آن برخورد کرد.
- اطلاعاتی در مورد سایر فعالیت‌هایی که ممکن است همزمان در حال انجام باشند.
- شرح وضعیت سیستم زمانی که سناریو به پایان می‌رسد.

a) ۲ b) ۴

c) ۳ d) ۵

(صفحه ۱۱۹_بخش ۴.۳.۲_کتاب سامرویل ادیشن ۱۰ یا صفحه ۱۷۳ _ بخش ۹.۲_ کتاب پرسمن ادیشن ۸)

۴- مسئولیت‌های یک کلاس تحلیل توسط کدام یک از موارد زیر تعریف می‌شود؟

a) با Attributeهای کلاس

b) با Collaboratorهای کلاس

c) با Operationهای کلاس

d) گزینه‌های a و c

(صفحه ۱۹۲_بخش ۱۰.۴_کتاب پرسمن ادیشن ۸)

۵- کدام یک از موارد جز اهداف مدلسازی نیازمندی‌ها نیست؟

a) تعریف شدن مجموعه‌ای از نیازمندی‌های برای اینکه بتوان اعتبارسنجی انجام داد.

b) توصیف نیازمندی‌های مشتری

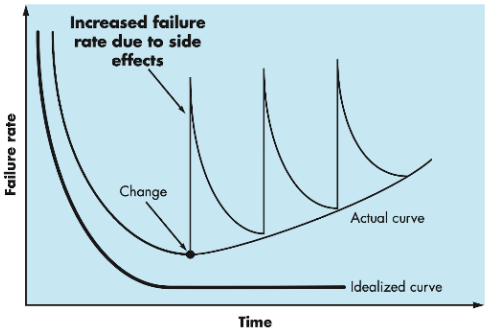
c) ایجاد یک راه حل خلاصه برای حل مساله مطرح شده

d) ایجاد مبنایی برای طراحی نرم‌افزار

(صفحه ۱۱۴ تا ۱۱۷_بخش ۷.۳.۳_کتاب پرسمن ادیشن ۸)

بخش دوم – سوالات تشریحی

۶- تفاوت اساسی بین نرم‌افزار و سخت‌افزار با توجه به زمان چیست؟



پاسخ: نرم‌افزار و سخت‌افزار هر دو در ابتدای حیات خود نرخ شکست (Failure rate) بالایی دارند. نقص‌های (Defects) احتمالی که در طراحی یا تولید یک سخت‌افزار وجود داشته است باعث بالا بودن نرخ شکست آن می‌شود. به مرور زمان و اصلاح این نقص‌ها، نرخ شکست کم شده و برای یک بازه زمانی در حالت پایدار قرار می‌گیرد. با گذشت زمان بیشتر، دوباره نرخ شکست شروع به افزایش کرده و سخت‌افزار شروع به فرسوده شدن می‌کند. این فرسوده شدن به دلیل شرایط محیطی مثل تغییرات دمای محیط، ارتعاشات تاثیرگذار بر سخت‌افزار و سایر شرایط رخ می‌دهد. وقتی سخت‌افزار فرسوده شد می‌توان آن را با یک قطعه یدکی جایگذاری و نگهداری کرد.

نرم‌افزارها همانند سایر ساخته‌های دست بشر یک سیستم فیزیکی نیستند، بلکه سیستم‌های منطقی‌اند، بنابراین تغییرات محیطی تاثیری در آن‌ها ندارد و هر چقدر کار کنند نه تنها فرسوده نشده بلکه مثل روز اول عمل می‌کنند (نمودار ایده‌آل نمایش داده شده در شکل بالا). اما موضوع دیگری در نرم‌افزارها مطرح است و آن درخواست‌های تغییری هستند که به صورت پیوسته به سمت نرم‌افزار می‌آید. وقتی تغییرات در نرم‌افزارها اعمال می‌شوند به دلیل عوارض جانبی (Side Effects) که دارند باعث افزایش مجدد نرخ شکست می‌شوند و پیش از آنکه این نرخ به حالت پایدار برسد درخواست تغییر بعدی می‌آید و تکرار این روند. این موضوع باعث می‌شود تا در واقعیت با گذشت زمان نمودار نرخ شکست، شکل ایده‌آل را به خود نگیرد و یک سطح حداقلی از شکست کم‌کم شروع به پیدایش کند. بنابراین تغییرات زیاد باعث می‌شوند تا نرم‌افزارها کیفیت خود را از دست داده و در نهایت به جای فرسوده شدن رو به نابودی بروند چرا که درخواست‌های تغییر نرم‌افزار را نمی‌توان با یک قطعه یدکی حل نمود و فرآیند نگهداری نرم‌افزار پیچیده‌تر از سخت‌افزار است.

(صفحه ۵ و ۶_بخش ۱.۱.۱_کتاب پرسمن ادیشن ۸)

۷- توضیح دهید Prototyping در چه مواقعی می‌تواند در پیشبرد پروژه موثر باشد؟

پاسخ: بعضی مواقع، یک مشتری مجموعه‌ای از اهداف کلی را برای نرم‌افزار تعریف می‌کند، اما الزامات دقیق برای عملکردها و ویژگی‌ها را مشخص نمی‌کند. در موارد دیگر، توسعه‌دهنده ممکن است از کارایی یک الگوریتم، سازگاری یک سیستم عامل یا شکلی که تعامل انسان و ماشین باید داشته باشد مطمئن نباشد. در این موارد و بسیاری از موقعیت‌های دیگر، یک الگوی نمونه‌سازی(Prototyping) ممکن است بهترین رویکرد را

ارائه دهد. یعنی توسعه‌دهنده با ارائه یک نمونه اولیه از محصول به مشتری، نظر او را جویا می‌شود و در ادامه، هم مشتری و هم توسعه‌دهنده به فهم خواسته واقعی مشتری نزدیک‌تر می‌شوند.
(صفحه ۴۵_بخش ۴.۱.۳_کتاب پرسمن ادیشن ۸)

۸- چرا ممکن است یک سیستم با عمر طولانی به اسناد طراحی بیشتری نیاز داشته باشد؟

پاسخ: سیستم‌های با عمر طولانی ممکن است به اسناد طراحی بیشتری نیاز داشته باشند تا اهداف اصلی توسعه‌دهندگان سیستم را به تیم پشتیبانی منتقل کنند. این مستندات به حفظ درک روشنی از اهداف سیستم در یک دوره طولانی کمک می‌کند.
(صفحه ۹۲_بخش ۳.۴.۲_کتاب سامرویل ادیشن ۱۰)

۹- مدل‌های ایجاد نرم‌افزار را در نظر بگیرید:

A. چرا باید برای ایجاد یک نرم‌افزار بر اساس یک مدل پیش برویم و در طول پروژه پایبند به آن مدل باشیم؟

پاسخ: به دو علت اصلی که عبارتند از:
- کمک به ایجاد نرم‌افزارهایی با کیفیت خوب و کاهش اتلاف هزینه‌ها و زمان
- کمک به ایجاد نرم‌افزار به شکل سیستماتیک و منظم

(صفحه ۴۷ _ برگرفته از SafeHome _کتاب پرسمن ادیشن ۸)

B. از یک تیم مهندس نرم‌افزار برای پروژه‌ای در یک شرکت نفت بزرگ دعوت شده است. این شرکت چندین دپارتمان دارد و تیم مهندسی نرم‌افزار با دپارتمان مدیریت اطلاعات (MIS) تعامل می‌کند. سیستم MIS این شرکت موروثی (Legacy) است و هدف، انتقال دیتاها به یک سیستم جدید است (مهاجرت داده). فرآیندها، قراردادهای قانونی و معیارهای پذیرش این شرکت بسیار خاص و حساس هستند. به نظر شما چه مدل ایجاد نرم‌افزاری برای راه‌اندازی این سیستم انتقال داده را تیم مهندس نرم‌افزار انتخاب خواهد کرد؟ نام مدل و علت اصلی انتخاب آن کافی است.

پاسخ: مدل حلزونی (Spiral) – با توجه به حساسیت بالایی که این شرکت در دیتاهای خود دارد، پیشگری از مشکلات احتمالی که قبلاً رخ نداده‌اند حائز اهمیت است و مدل حلزونی به علت تحلیل ریسک، ارزیابی و پیش‌بینی جایگزین‌هایی برای این مشکلات مدل مناسبی خواهد بود.
(صفحه ۴۸_بخش ۴.۱.۳_کتاب پرسمن ادیشن ۸)

C. مهم‌ترین مشکلات مدل‌های سنتی (مثل مدل آبشاری) نسبت به مدل‌های چابک، چیست؟ (اشاره به ۳ مورد و توضیح کامل آن‌ها کفایت می‌کند.)

پاسخ: تفاوت اصلی بین مدل‌های سنتی مانند روش آبشاری (Waterfall) و مدل‌های چابک (Agile) در رویکرد آن‌ها نسبت به ایجاد نرم‌افزار است. در اینجا به ۳ مورد از آن‌ها اشاره می‌کنیم:

۱. **رویکرد توالی‌محور (Sequential) در مقابل رویکرد Iterative:**

- روش آبشاری: مدل آبشاری به روش خطی و توالی‌محور به ایجاد نرم‌افزار می‌پردازد. هر فاز از فازهای پنج‌گانه مدل آبشاری، به ترتیب یکی پس از دیگری کامل می‌شود و پیشرفت به صورت کاملاً خطی است. پس از اتمام یک فاز، پروژه به فاز بعدی منتقل می‌شود و نمی‌توان به عقب بازگشت و یا فیدبک داد.
- روش‌های چابک: روش‌های چابک بر رویکرد Iterative و Incremental تأکید می‌کنند. آنها فرایند ایجاد را به بخش‌های کوچکتر و قابل مدیریت تقسیم می‌کنند. همچنین، رویکرد چابک امکان انعطاف پذیری، بازبینی و فیدبک‌های مداوم و توانایی سازگاری با تغییرات و به آغوش کشیدن آنها را در طول پروژه را فراهم می‌کند.

۲. **مشارکت مشتری: (Customer Involvement)**

- روش آبشاری: مشارکت مشتری معمولاً به مراحل اولیه (در حد استخراج اولیه نیازمندی‌ها) و یا انتهایی پروژه (مرور نهایی محصول) محدود است و در طول فرآیند ایجاد خیلی کمتر است.
- روش‌های چابک: برخلاف مدل‌های آبشاری، یکی از مهم‌ترین اصول مدل‌های چابک تضمین کردن مشارکت مداوم مشتریان و گرفتن فیدبک‌های آنها در طول دوره‌های ایجاد نرم‌افزار است. این مشارکت مشتریان یا به طور کل افراد ذینفع (Stakeholder) در طول فرآیند ایجاد، بازخورد مداومی را بر روی قسمت‌های مختلف محصول ارائه می‌دهد تا اطمینان حاصل شود که محصول با نیازهای متغیر مشتری همخوانی دارد و ایجاد نرم‌افزار از مسیر مطلوب خارج نشده است.

۳. **مستندسازی (Documentation):**

- روش آبشاری: روش‌های آبشاری معمولاً بر تولید مستندات جامع در هر مرحله ایجاد تأکید دارند. در هر مرحله، پیش از شروع آن فاز، مستندات حجیمی در مورد نحوه پیاده‌سازی، ریسک‌های احتمالی و ... تهیه می‌شود و نحوه ادامه مسیر آن فاز، از این مستندات تعیین می‌شود.
- روش‌های چابک: در حالی که روش‌های چابک به مستندسازی اهمیت می‌دهند، اما تمرکز اصلی آن‌ها بر روی نرم‌افزار قابل اجرا است و تأکید بر تولید مستندات "کافی و نه بیشتر" برای پشتیبانی از توسعه و همکاری دارند. این موضوع، همان اصل Working Software over Comprehensive Documentation از Agile manifesto است.

(فصل ۴ و ۵ _کتاب پرسمن ادیشن ۸)

۱۰- با در نظر گرفتن رویکرد چابک به سوالات زیر پاسخ دهید:

A. در اکثر پروژه‌های نرم‌افزاری پیش‌بینی موارد زیر سخت است:
   - اینکه کدام نیازمندی‌های مشتری تغییر خواهند کرد و کدام نیازمندی‌ها ثابت خواهند بود؟
   - اینکه به چه میزان طراحی پیش از پیاده‌سازی احتیاج داریم؟
   - و چه مقدار زمان از نظر برنامه‌ریزی برای تحلیل و طراحی، پیاده‌سازی و تست محصول نیاز خواهد بود؟

فرآیندهای چابک چگونه در جهت رفع این شرایط‌های نیاز به پیش‌بینی پاسخ می‌دهند؟

پاسخ: فرآیندهای چابک با رویکرد افزایشی و تکرارشونده خود با شرایط پروژه تطابق پیدا کرده و سازگار می‌شوند. این فرآیندها با ارائه Releaseهای منظم در بازه‌های کوتاه مدت که نمونه‌های قابل اجرا (Prototype) یا بخش‌هایی از یک سیستم عملیاتی هستند، بازخوردهای مشتری را دریافت می‌کنند و پس از ارزیابی و برنامه‌ریزی مناسب برای تغییرات جدید، در تکرار بعدی به تحلیل، طراحی و پیاده‌سازی لازم برای آن تغییرات می‌پردازند. بنابراین روال افزایشی و تکرار شونده تدریجی در این فرآیندها باعث می‌شود تا سرعت پروژه با تغییرات نیازمندی‌های مشتری هماهنگ شده و تطبیق یابد. ارزیابی‌های انجام شده پس از هر تکرار و دریافت بازخورد مشتری باعث می‌شود تا به تناسب تغییرات درخواست شده برای مدیریت زمان، تحلیل، طراحی و پیاده‌سازی برنامه‌ریزی صورت گیرد.

(صفحه ۷۰_بخش ۵.۳_کتاب پرسمن ادیشن ۸)

B. اگر برای سیستم‌های بزرگ و با عمر طولانی که توسط یک شرکت نرم‌افزاری برای مشتری‌های خارجی توسعه داده می‌شوند، از رویکرد چابک استفاده شود، چه مشکلاتی ممکن است بوجود آید؟ ۳ مورد از مشکلات ممکن را ذکر کنید.

پاسخ:

- غیر رسمی بودن رویکرد چابک با رویکرد حقوقی تعریف قرارداد که معمولاً در شرکت‌های بزرگ استفاده می‌شود، ناسازگار است.
- روش‌های چابک بیشتر برای ایجاد نرم‌افزارهای جدید مناسب هستند تا برای نگهداری نرم افزار؛ این در حالی است که اکثر هزینه‌های نرم‌افزاری در شرکت‌های بزرگ ناشی از حفظ سیستم‌های نرم‌افزاری موجود آنهاست.
- روش‌های چابک برای تیم‌های کوچک طراحی شده‌اند، با این حال بسیاری از پروژه‌های ایجاد نرم‌افزار، اکنون شامل تیم‌های توزیع‌شده در سراسر جهان می‌شوند

(صفحه ۸۹_بخش ۳.۴.۱_کتاب سامرویل ادیشن ۱۰)

C. فکر می‌کنید مدل‌های چابک خود چه مشکلاتی داشته باشند؟ (حداقل ۴ مورد)

پاسخ: از مشکلاتی که در مدل‌های چابک می‌توان اشاره نمود عبارتند از:

- از آن‌جایی که در این مدل‌ها نیاز به ارتباط پیوسته با مشتریان وجود دارد، اینکه علاقه مشتری را به مشارکت در طول فرآیند نگهداریم سخت است.
- اعضای تیم نرم‌افزار ممکن است متناسب با شدت مشارکتی که باید در طول فرآیند باشد، نباشند. این به این معناست که بعضی از افراد ممکن است راحت نباشند تا در یک محیط سریع، مشارکتی کار کنند یا ترجیح دهند به تنهایی کار کنند، از یک برنامه‌ریزی دقیق پیروی کنند. با تغییر مرتب برنامه‌ریزی‌ها حس راحتی نداشته و انعطاف‌پذیر نباشند. بنابراین ترجیحات افراد می‌تواند با اصول چابک کار تیمی تضاد داشته باشد و فرد به مرور زمان احساس بی‌انگیزگی کرده یا در عملکردش خدشه وارد شده و کار با کیفیتی ارایه نکند.
- تغییر اولویت‌بندی‌ها در مواردی که تعداد ذینفعان زیاد باشد می‌تواند سخت باشد.
- نگهداشتن اصل سادگی در طول فرآیند احتیاج به کار بیشتری دارد. یکی از اصول چابک سادگی است به معنای حداکثر کردن تعداد کارهایی که انجام نشوند. این یعنی تسک‌های پیچیده و غیر ضروری انجام نشود، تعامل‌ها با اعضای تیم و ذینفعان به نحو موثری باشد. تصمیم‌گیری‌ها، اولویت‌بندی نیازمندی‌ها با دقت و سخت‌گیری انجام شود و احتیاج به مهارت بالا و خلاقیت داشته باشد. بنابراین کم کردن مواردی که مفید نیستند، کم کردن پیچیدگی‌ها و مدیریت زمان برای حفظ سادگی احتیاج به کار اضافه دارد.

ذکر موارد صحیح دیگر نیز قابل پذیرش است.

(با توجه به فصل ۵، مطالب مطرح شده در کلاس و دانشی که از مدل‌های چابک دریافتید.)

موفق باشید

تیم آموزش مهندسی نرم‌افزار