پاسخ تمرینات- مهدی شعبانی

تمرین ۱ - موسسه SEI چیست ؟

پاسخ: انستیتوی مهندسی نرم افزار دانشگاه کارنگی ملون، برنامه ها، فعالیتها و ابتکارات مربوط به فعالیتهای مهندسی نرم افزار تعریف کرده و حمایت می کند. دانشگاه کارنگی ملون برترین دانشگاه جهان در حوزه مهندسی و علوم کامپیوتر است. موسسه SEI تحت حمایت و کمک مالی در حوزه تحقیق و توسعه از طرف دولت و وزارت دفاع آمریکاست. این انستیتو به صورت ویژه در حوزه امنیت سایبری، تضمین نرم افزار و مهندسی نرم افزار فعالیت می کند. این موسسه با نوآوری در فراهم کردن ابزارها، تکنولوژیها و تجربیات فنی در دنیا پیشرو است.

تمرین ۲ - تفاوت نمونه سازی (prototyping) اکتشافی (Exploratory) و تکاملی(Evolutionary)

پاسخ: پروتوتایپ یک پیش نمایش ، پیش نمونه، طرح و الگوی اولیه از نرم افزار در جهت دریافت بازخورد از بیننده و کاربر است و برای بررسی Feasibility و هزینه فایده کردن و تخمین کارا بودن استفاده می شود. در واقع مدل ملموس برای ارزیابی در اختیار قرار می دهد. کاربرد ارتباطات و اشتراک مساعی و آموزش نیز دارد.
بر اساس اهدافی که پروتوتایپینگ دارد سه روش اصلی دارد: *اکتشافی، تجربی، تکاملی*

روش اکتشافی برای هندل کردن چالشهای ارتباطی بین کاربران و توسعه دهندگان استفاده می شود. در فاز اولیه ی توسعه برای شناخت نیازمندیها استفاده می شود. بعنوان یک آزمون کوچک برای بررسی برخی فرضیات کلیدی در مورد عملکرد یا تکنولوژی مورد استفاده در پروژه کاربرد دارد.بعضا برای بررسی کارکرد یک مولفه نرم افزاری یا سخت افزاری نیز از این پرو تو تایپ استفاده می شود.بصورت غیر رسمی و informal نوشته می شود و معمولا از چند صد خط کد تجاوز نمی کند و نیزبعد از اینکه کار مورد نظر انجام گرفت دیگر کاری با آن نخواهیم داشت.

روش تکاملی که به صورت تدریجی رشد می نماید تا به سیستم واقعی تبدیل شود.در حقیقت در طی تکرار ها رشد و نمو نموده تا به نمونه واقعی و نهایی تبدیل شود . به همین دلیل باید بصورت رسمی و اصولی نوشته وطراحی شود تا مدیریت و تست آن امکان پذیر باشد. و اضح است که در طی تکرار ها ، کد ها و طراحی این پروتوتایپ ها تغییر می کند تا به نمونه نهایی نزدیک و نزدیک تر شود.

تمرین ۳ - دیدگاه ۱+۴

پاسخ: طبق این دیدکاه، آقای کروچن بیان کرد که هیچ کدام ازساختارها با هم تناقض ندارند و همه آنها در جهت بیان کردن نیازمندیهای سیستم هستند ولی بهتر است از موارد کاربری مهم به منظور انتخاب ساختارها استفاده شود. بدین منظور یک دیدگاه به نام ۱+۴ ارایه کردند که دیدهای آن به شرح زیر هستند:

دید منطقی، شامل کلاسها و اشیا است که در شی گرایی جهت دسترسی به حداکثر تجرید استفاده می شوند. در واقع همان ساختار ماژول است. دید منطقی درگیر فابلیت هایی است که سیستم برای کاربران نهایی فراهم می اورد. برای نمایش دید منطقی از نمودار های کلاس، Communication، و نمودار Sequenceاستفاده می شود.

دید فرایند نحوه همزمانی و توزیع وظیفه مندیها را نشان می دهد.(ساختار مولفه ها و اتصال ها). دید فرایند، ، درگیر وجهه پویای سیستم است. پروسه های سیستم را توصیف کرده و نحوه تعامل ان ها با هم. همچنین بر رفتار سیستم در زمان اجرا

تمرکز دارد .دید فرایند همزمانی(concurrentcy) ، توزیع، کارایی و مقیاس پذیری را پوشش می دهد. در UML از نمودار های Activity برای نمایش ان استفاده می شود.
دید توسعه، سازماندهی ماژولها، کتابخانه ها، زیرسیستم ها و واحدهای توسعه را نمایش می دهد که در واقع ساختار تخصیص را نمایش می دهد. دید توسعه برای تشریح سیستم از دید یک برنامه نویس است و درگیر مدیریت نرم‌افزار است. به این Viewهمچنین Implementation View هم می گویند. در UMLاز نمودار Component برای توصیف کمپوننت های سیستم استفاده می کند .برای نمایش این دید در UML از نمودار Package استفاده می کنیم.
دید فیزیکی، نحوه نگاشت اجزای سیستم را به پردازنده ها و گره های ارتباطی مشخص می کند. این ساختار مشابه ساختار تخصیص است اما بیشتر جنبه استقرار مد نظر آن است. دید فیزیکی سیستم را از دید یک مهندس سیستم نمایش می دهد. این دید درگیر توپولوژی کمپوننت های نرم‌افزاری در لایه فیزیکی است، بعلاوه ارتباطات فیزیکی بین این کمپوننت ها. در UMLاز نمودار های Deployment برای نمایش لایه فیزیکی استفاده می شود.
دید سناریو به خودی خود یک دید نیست درواقع همان موارد کاربرد است. توصیف معماری با استفاده از مجموعه ای از useکaseها ، سناریوها انجام می شود. این سناریو ها توصیفگر توالی ارتباطات بین اشیا و پروسه ها هستند

تمرین ۴ - تعریف مفاهیم زیر:
Class Diagram – Use Case Diagram – Sequence Diagram – Deployment Diagram

پاسخ: در UML 2ما نموداری بنام "نمودار ساختار" نداریم، بلکه از آن تنها برای دسته بندی نمودار ها استفاده می شود. بجای آن **نمودار کلاس (class diagram)**وجود دارد که مجموعه ای از قوانین و عناصر نشانه گذاری را در اختیار ما قرار می دهد که تقریبا در تمامی نمودار های ساختار مورد استفاده قرار می گیرند. در کلاس دیاگرام چهار نوع رابطه وجود دارد که می توانیم آنها را بین کلاسها برقرار کنیم و عبارتند از , associationdependency, aggregation generalization
در واقع نمودار کلاس، نموداری است که می تواند معرف توصیف تمام یا بخشی از یک سیستم و ارتباطات انها باشد.
Use-Caseها دنباله ای از فعالیت ها را توصیف می کنند که سیستم آن ها را انجام می دهد تا یک نتیجه قابل مشاهده از یک مقدار را به یک عامل تحویل دهد. این نمودار نیازهای سیستم را بیان می‌کند و هر شخص با مرور ان می‌تواند بفهمد سیستم درگیر ایجاد چه چیزی است . در ساده ترین حالتUse-Case ها به وسیله مصاحبه با کاربران و انتظاراتی که انها از سیستم دارند تعریف می‌شود. برای نمایش اطلاعات مربوط به نیازهای کاربران که در قالبUse-Case ها جمع‌اوری شده است از نمودارUse-Case استفاده می‌شود. Use-Case ها که در فاز آنالیز پروژه برای شناساسیی و تقسیم بندی فعالیت های سیستم استفاده می شوند و می توننند به عنوان سرویس ها یا کارکردهایی که سیستم برای کاربران خودش فراهم می کنه نیز توصیف بشوند. دو دیدگاه وجود دارد: یکی داخلی , دید ساختاری و دیگری خارجی و دید وظیفه گرایی(task Oriented)
در دیدگاه اول ما باید کلاسها و متدها را تعریف کنیم و سپس واسطهای کاربری (user interface)را تعریف کنیم. مشکل اینجاست که برای کاربر مهمترین چیز رفتار سیستم است ولی واسطهای کاربری تنها قسمت آخر فرآیند را تعریف می کنند.
در دید دوم ،سیستم از Actor ها و فعالیتها و کلاسهایی که به فعالیتها وصل شده اند پشتیبانی می کند.در این دیدگاه هیچ کار ناخواسته ای وجود ندارد و سیستم تمام فعالیتهای کاربر را پشتیبانی می کند که همه آنها در Diagram Use case نمایش داده می شود.
دیاگرامهای use case با استفاده از Use case و Actor عملکرد (Functionality) سیستم رامدلسازی می کنند.
نمودارهای توالی برای نشان دادن ترتیب فرایندهای صورت گرفته در یک گردش کاری استفاده می شوند. هر گردش کار از تعدادی فرایند تشکیل شده است.
نمودار استقرار یا توزیع و یا بکارگیری در uml ، معماری یک سیستم متکی به رایانه را به صورت فیزیکی نمایش می دهد.

این نمودار قادر به نمایش رایانه و دستگاه های مربوط به آن و نیز ارتباطاتی که این دستگاه ها با هم دارند و نیز نرم افزاری که روی هر ماشین قرار دارد، می باشد. هر رایانه توسط یک مکعب نمایش داده می شود و ارتباط آن نیز با رایانه های دیگر توسط خطوط ارتباطی ارائه می گردند.
تعریف دیگری که می توان ارائه نمود بدین صورت است: سخت‌افزار بکار رفته در پیاده‌سازی سیستم و همچنین محیط‌های اجرا و سایر مولفه هایی که بایستی بر روی این سخت‌افزار مستقر شوند را شرح می دهد.
بنابراین نمودار استقرار جهت توصیف دید ایستای استقرار یک سیستم بکار می رود. نمودارهای استقرار از گره ها (node) و رابطه ی بین آن ها تشکیل می شود.

تمرین ۵ -تعریف و تفاوت های GIT وSVN

پاسخ: Subversionکه به اختصار SVN نیز خوانده می شود یک سیستم کنترل نسخه ی اصطلاحاً Centralized یا «متمرکز» است. به عبارت دیگر، تمامی اعضای تیم توسعه ی نرم‌افزار روی یک نسخه ی واحد که روی سروری خاص قرار دارد کار می کنند. وقتی توسعه‌دهنده ای اقدام به دریافت یک نسخه از پروژه از روی سرور می کند، اس وی ان آخرین نسخه از پروژه را در اختیار وی قرار خواهد داد. در سیستم ورژن کنترل Git که یک سیستم اصطلاحاً Distributed یا «نامتمرکز «است، شرایط تا حدودی متفاوت تر است. زمانی که یکی از توسعه دهندگان پروژه ای را از روی سرور اصطلاحاً Clone) کلون به معنی دریافت کردن) می کند، وی نسخه‌ای کامل و جامع از آنچه روی سرور قرار دارد را در دست خواهد گرفت که شامل تاریخچه ی تغییرات پروژه نیز می باشد.
مزایای مدل توزیع شده: عدم نیاز به شبکه- سرعت بیشتر - مشارکت در پروژه بدون نیاز به دسترسی کامیت - کاهش خطر از دست رفتن اطلاعات- پشتیبانی از مدل‌های کاری (workflow)متعدد - امکان داشتن مخزن خصوصی از یک پروژه- تمیزتر بودن مخزن اصلی
معایب مدل توزیع شده: مشکل درک مفاهیم- مشکل کنترل دسترسی-
مزایایGit : سرعت فوق العاده- حجم کم- پایداری - راحتی کار با شاخه ها (branch)- دنبال کردن محتوای فایل به جای نام فایل- کامیت با جزئیات بیشتر- شلوغ نکردن working directory با پوشه‌هایgit. - نمایش میزان پیشرفت عملیات
معایبGit : کدهای طولانی SHA1 به جای شماره‌های متوالی - عدم سازگاری دستورات با سیستم‌های سنتی- امکان checkoutجزئی وجود ندارد- امکان افزودن پوشه خالی به مخزن وجود ندارد- عدم سازگاری با ویندوز- کمبود رابط گرافیکی قوی

تمرین ۶ Metamorphic Testing - چیست ؟

پاسخ: مواردی از تست ها هست که یا تست کردن خروجی امکان پذیر نیست یا خیلی خیلی سخت و زمانبر و پرهزینه است. در متامورفیک تست، موارد تست از روی ویژگیهای خاصی از صورت مسله که قبلا در تست ها نتایج مشخصی از آنها بدست آمده تولید می شود و با اجرای آن موارد تست، نتیجه تست به موارد سخت بسط داده میشود. مثلا در تست خروجی تابع سینوس به ازای یک زاویه ی غیرنرمال و ناشناخته، به تست خروجی تایج سینوس بر روی مقادیر شنلخته شده یا کمان های مضربی از آنها بسنده می کنیم. این تست به رفتار های خاص موجود در مساله استناد می کند.

تمرین ۷ Spring Framework - چیست؟

پاسخ: فریم‌ورک اسپرینگ (Spring Framework) یکی از چهارچوب‌های برنامه‌نویسی معروف و پرکاربرد در زبان جاواست. این فریم‌ورک متن‌باز (Open Source) بوده و اولین نسخه آن در ماه ژوئن سال ۲۰۰۳ میلادی عرضه شده است. اسپرینگ از ابتدای ارائه تا به امروز تغییرات زیادی داشته و امکانات و ابزارهای بسیار قدرتمندی را در اختیار برنامه‌نویسان قرار می‌دهد. هدف اصلی اسپرینگ توسعه‌ی برنامه‌های Enterprise است و بیشتر در پلتفرم Java EE مورد استفاده قرار می‌گیرد.
اسپرینگ یک فریم‌ورک ماژولار بوده و ماژول‌های آن در دسته‌بندی زیر قرار می‌گیرند:
Core Container:شامل ماژول‌های Bean و Core است که هسته اصلی اسپرینگ را تشکیل می‌دهند.
AOP and Instrumentation:یک پیاده‌سازی از برنامه‌نویسی جنبه‌گرا (AOP) را فراهم می‌کند؛ در واقع این ماژول یک interceptorاست.
Messaging:ماژول مربوط به MessagePassing در برنامه‌نویسی شی‌گرا.
Data Access/Integration:ماژول دسترسی به DataSource و پایگاه داده.
Web:برای طراحی web بر اساس مدل MVC به کار می‌رود.
Test:این ماژول استفاده از ابزارهایی مانند JUnit و... را برای تست برنامه‌ها به ما می‌دهد.

تمرین ۸ - مدل Furps چیست؟

پاسخ: مدل FURPS توسط رابرت گریدی و شرکت هیلوت پاکارد پیشنهاد شد. خصوصیات کیفی، با استفاده از دو مجموعه متفاوت نیازمندیها بررسی شدند: نیازمندیهای کارکردی (F) که بوسیله ورودی و خروجی تعریف می‌شود و نیازمندیهای غیرکارکردی که بوسیله قابلیت استفاده، قابلیت اطمینان، کارائی و قابلیت پشتیبانی تعریف می‌شود. زمانی که این مدل استفاده می‌شود دو گام در نظر گرفته می‌شود: مشخص نمودن اولویت‌ها و تعریف صفات کیفی که می‌بایست اندازه‌گیری شود

تمرین ۹ – سرویس choreography را بررسی نمایید.

پاسخ: Choreographyرویکردی است که برای سناریو های شامل فرآیندهای پیچیده، سیستم های مبتنی بر رخداد و مبتنی بر عامل مناسب است. در رویکرد Choreographyقوانینی وضع می شوند که رفتار هر بخش از فرآیند را بصورت جداگانه مشخص می کنند. در این رویکرد رفتار کلی فرآیند از طریق برقراری ارتباط بین زیر بخش های آن حاصل می شود، هر بخش بصورت خود مختار تنها طبق قوانین خود عمل می کند. دو رویکرد اصلی برای پیاده سازی وجود دارد: مبتنی بر پیغام (Message Component)و مبتنی بر اجزاء کاری.(Work Component)
به طور خلاصه:

- رفتار کلی فرآیند از بخش های درون آن نشأت می گیرد (رویکرد پایین به بالا) و نیازی به یک دید یکپارچه بصورت کلی از فرآیندها نیست.
- فرآیندهای پیچیده به کارهای کوچک تر با دستورالعمل مجزا تقسیم می شوند و هر بخش دستورالعمل خود را کنترل می کند.
- قابل نگاشت به سیستم های مبتنی بر عامل و رخداد است.

- معمولاً راه اندازی آن ها ساده نیست ولی برای تبدیل به فرآیندهای پیچیده تر راحت تر هستند.
- می توان از فرآیند یک شمای گرافیکی تهیه کرد. به طور مثال: مدل جریان عملگرهادر choreography برای بدست آوردن یک Composite service طی یک فرایند، هر سرویس مثل قطعه ای از یک پازل بخشی از فرایند رو در دست میگیرد و از سرویس های متعامل دیگر اطلاع دارد و با آنها در تعامل و تکامل است.

تمرین ۱۰ - پروتکل های SOAPو REST را بررسی نمایید.

پاسخ: بر اساس پروتکل دستیابی ساده به شیء است که تمامی خدمات وب به گردش در می‌آید. منظور از پروتکل، مجموعه شرح قواعد و فرمت‌های مربوط به ارسال پیامهای مخابراتی از یک ماشین به ماشین دیگر است SOAP .پروتکل ساده‌ای ست که به منظور سازگاری (compatibility) با سکوهای (platforms) مختلف و نیز سیستم‌های عامل (operating systems)گوناگون نوشته شده است.

که مخفف Representational State Transfer میباشد یک معماری وب سرویس است که از HTTP برای انتقال اطلاعات میان کلاینت و سرور استفاده میکند کار کردن با REST بسیار ساده تر از وب سرویس های پیچیده ای مانند SOAPمیباشد.

یک سرویس به اصطلاح RESTful عموما بر روی پروتکل HTTP و تمام افعال استاندارد این پروتکل را که توسط مرورگرهای وب قابل درک هستند کار میکند مانند(GET, POST, PUT, DELETE)

معماری REST لازم است شرایط زیر را داشته باشند:

کلاینت سرور (client-server) باشد.

بدون حالت (stateless) باشد.

قابلیت cache داشته باشد.

سیستم لایه‌بندی شده داشته باشد.

واسط یکنواخت داشته باشد.

دارای قابلیت کد در صورت نیاز باشد.

از لحاظ رویکرد برنامه نویسی REST جایگزینی ساده برای سرویس‌های وب است .توسعه‌پذیری در تعاملات میان اجزا، عمومیت واسط ها، توسعه مستقل اجزا و استفاده از واسطه ها از کلیدی ترین اهداف معماری REST می‌باشد و همچنین استفاده از معماری REST در برنامه‌نویسی کارایی، سادگی، انعطاف‌پذیری، امکان مشاهده و نظارت، قابلیت حمل و قابلیت اطمینان را افزایش می دهد.

یک وب سرویس REST دارای مشخصات زیر است

بوسیله URI کار میکند یعنی ریسورس ها و کالکشن های خود را به صورت http://example.com/resources دریافت میکند

اطلاعات را به صورت عموما JSON دریافت میکند البته میتواند اطلاعات به صورت XML هم برگردانده شود

برخلاف وب سرویس های برپایه SOAP هیچ استاندارد رسمی برای وب سرویس های RESTوجود ندارد به دلیل اینکه RESTیک معماری است در حالی که SOAP یک پروتکل وب سرویس است.

تمرین ۱۱ - فرق sporadic و stochastic چیست؟

پاسخ: در stochastic یک روبداد به صورت تصادفی و غیر پیش بینی و مبتنی بر یک تابع توزیع روی می دهد. در Sporadic ، رویدادهایی مد نظر هستند که نه به صورت periodic و نه به صورت stochastic هستند. به عنوان مثال وقتی نمیدانیم که یک کاربر چه زمانهایی یک درخواست را میفرستد در تخمین sporadic میتوانیم بگوییم که مثلا قطعا در یک ساعت بیشتر از ۱۰ درخواست نمیتواند بفرستد. این تخمین را میتوانیم بر اساس محدودیتهای محیط یا خود عامل که اینجا کاربر است حدس بزنیم.

تمرین ۱۲ - تعریف Dependency Injection ؟

پاسخ: به صورت خلاصه ترزیق وابستگی و یا dependency injection ، الگویی است جهت تزریق وابستگی‌های خارجی یک کلاس به آن، بجای استفاده مستقیم از آن‌ها در درون کلاس. تزریق وابستگی ها یک الگوی طراحی نرم افزار است که با ما امکان می دهد که در زمان کامپایل و هم چنین زمان اجرا انتخاب کنیم که چه کامپوننت های بایستی ساخته شده و مورد استفاده قرار گیرند.
در واقع تزریق وابستگی ها مجموعه ای از قوانین و الگو های طراحی نرم افزار است که ما را برای توسعه کدی با همبستگی پایین توانمند می سازد. برای مثال شخصی را در نظر بگیرید که قصد خرید دارد. این شخص می‌تواند به سادگی با کمک یک خودرو خود را به اولین محل خرید مورد نظر برساند. حال تصور کنید که ۷ نفر عضو یک گروه، با هم قصد خرید دارند. خوشبختانه چون تمام خودروها یک اینترفیس مشخصی داشته و کار کردن با آن‌ها تقریبا شبیه به یکدیگر است، حتی اگر از یک ون هم جهت رسیدن به مقصد استفاده شود، امکان استفاده و راندن آن همانند سایر خودروها می‌باشد و این دقیقا همان مطلبی است که هدف غایی الگوی تزریق وابستگی‌ها است. بجای این‌که همیشه محدود به یک خودرو برای استفاده باشیم، بنابر شرایط، خودروی متناسبی را نیز می‌توان مورد استفاده قرار داد.

تمرین ۱۳ – مثال از Maintain multiple copy of data و Maintain multiple copy of computation ؟

پاسخ: چند کپی از دیتا مانند فرایند های cache . چه Cache server ها و چه Cache memoryها و دیسک های RAID از نوع Mirror
چند کپی از محاسبات هم انواع سرور های لود بالانس شده و تعداد CPU های موازی موجود در سرور ها