مهندسی نرم افزار - استاد شاه‌ای

۱۳ / ۷ / ۰۰

از برنامه‌نویسی تفننی وارد برنامه‌نویس حرفه‌ای

دانشجو آروم آروم ... مطلبش باش و حرف‌هایش

کتاب آقای پریسمن، مهندسی نرم افزار ویرایش هشتم

یه پروژه‌ای باشه که کوچک هم نباشه، بزرگ و متوسط باشه و وظایم در این مورد صحبت کنم که چطور

A تا Z یه پروژه رو انجام بدیم. هم بحث مدیریت داده هم فنی، مدیریت ارتباط، مدیریت پیکربندی

مدیریت کیفیت، مدیریت ریسک و بحث مدل‌های فنی یه پروژه نرم افزاری که اشاره داره به ابزار و تکنولوژی‌ای

که برای توسعه نرم افزار نیاز هست.

برای کنکوری‌ها هم از این کتاب بحث‌هایی میاد و از زبان uml هم استفاده می‌کنیم.

اگه بهترین تعریف پروژه تعریف کنم که برای بعضی‌ها هم عنوان تعیین می‌کنم یا اصلا زبان نرم و مابقی

هم اصلی پایان نرم هست.

شروع درس

فصل ۱ درمورد مفهوم نرم افزار و ماهیت خود نرم افزار صحبت میکنه چون مفهوم نرم افزاری یا Software

کلمه اصلی رشته مهندسی نرم افزار هست و مفهوم هم که ترجمه کلمه product هست که

Software هست و هم هست که نرم افزار رو از لحاظ پیکربندی بشناسیم چون هر محصولی

اجزایی داره که شناخت این اجزا برای نحوه که توی درس میگم بعدا روی نرم افزار

دو تا بخش داده‌ای داریم و نرم افزاری داریم حالا اینجا گفت برنامه‌ها منظورش program

هست. مثلا یه نرم افزار ساده فروش کفش رو در نظر بگیریم تعریف اصلی داره: نرم افزاری که در حال

مهندسی نرم افزار - استاد دسته ای

از برنامه نویسی نترس و فقط برنامه نویس خوبی

۰۰/۷/۱۲

سلامی آدوم آدوم ... مطمئن باش وقتی هستی

کتاب آقای پریسمن، مهندسی نرم افزار ویرایش هشتم

یه پروژه ای باشه که کوچک هم نباشه و بزرگ و متوسط باشه و می خوام در این مورد صحبت کنم که چطور

A تا Z یه پروژه رو انجام بدیم هم بحث مدیریتی داره هم فنی، مدیریت ارتباط، مدیریت پیکربندی

و ریسک، مدیریت کیفیت و همینطور موضوعات فنی یه پروژه نرم افزاری که اشاره داره به ابزار و فناوری ای

که برای توسعه نرم افزار نیاز هست.

برای کنکور ارشد هم از این کتاب بحث هایی میاد و از زبان uml هم استفاده می کنیم.

اگه بخوام ترین پروژه تعریف کنم که برای بچه ها هم برن تعیین کنم یا اقلا زبان نرم و مباحث

هم اقلا پایان نرم هست.

شروع درس

فصل ۱ درمورد محصول نرم افزار و ماهیت خود نرم افزار صحبت میکنه چون محصول نرم افزاری یا software

کلمه اصلی رشته مهندسی نرم افزار هست و محصول هم که ترجمه کلمه product هست که

software هست، هم هست که نرم افزار رو از لحاظ پیکربندی بشناسیم چون هر محصولی

اجزایی داره که شناخت این اجزاء برای فهمه که توی این درس میگم یعنی نرم افزار

دو تا بخش داره ای داره و نرم افزاری داره حالا این اگفت برنامه ها منظورش program

هست. مثلا یه نرم افزار ساده فروش دفاتر رو در نظر بگیر یه دو بخش اصلی داره: نرم افزاری که در حال

اجرا هست یعنی یه فایل exe که داره اجرا میشه الان رو کامپیوتر بنده الان وب و کامپیوتر
باشه یه سایتی هست یا یه اپ توی موبایلی هست و روی موبایل اجرا میشه هم این و هم اون
داده‌هایی که در حقیقت برنامه روی Device شما اجرا میشه. مثلا وقتی سیستم خودش یه سری
داده داره دیگه و یه جایی نگهداری میشن دیگه پس بدون تداخل جزو دیگر بخوای تعامل نرم افزاری هست
میتونه روی خود Device یا تو برنامه‌های وب، در سرور نگهداری بشه و یا ترکیبی
باشه پس بخش نرم افزاری و داده‌ای داریم و خود بخش نرم افزاری هم تقسیم بندی داره یکی از
این‌ها که برای هست یک اند و خوانت اند که بخشی هست که کاربر و استفاده کننده از نرم
افزار با این در ارتباطه و خوانت اند یعنی همین که و ببینید و قسمت اعظمش میشه
UI و چون استفاده کننده بیشتر کاربر هست پس یکم واسط کاربری

ما غیر از کاربران چه استفاده کننده دیگه‌ای برای نرم افزار متصور میشیم؟ میتونه کلاس انسانی باشه مثل
کلوچه‌ها غیر از انسان‌ها استفاده کننده دیگه‌ای سخت‌افزاری میتونه متصور بشه برای نرم افزار؟
داریم در صورتی که موضوع سیستم نرم افزار کامپیوتر چه کسای نیست، خب یا یه عامل انسانی هست مثل یه فرد
و کاربر و هر جا یکم سیستم منظم سیستم نرم افزاری هست چون حتی وام داره ...
یا انسان هست یا یه نرم افزار هست و میتونه یه نرم افزار دیگه ازش هم استفاده بکنه. مثلا یه سیستم ایمیل
یه سیستم حسابداری داریم که اینا سیستم‌های خودشونو دارن و این یکی با این یکی ارتباط برقرار میکنه تا اطلاعات
کاربر بشه این یه ارتباط نرم افزاری هست کلمه Web Service رو هم حتی شنیدید و حتی برای

یعنی یه نرم افزاری یه سرویس نرم افزاری رو ارائه میکنه که نرم افزارهای دیگه میتونن از این استفاده بکنن و کلمه web service داره به تکنولوژی این اشاره میکنه یا نرم افزارهای دیگه ای هم داریم برای این نوع ارتباط، ماهیتش اینه که یه نرم افزاری یه چند دیگه از نرم افزاری رو مورد استفاده قرار میده

مثلا توی سیستم تعاملی این جوری رایجه که پروفن مثال یه سیستم تو فرودگاه هست که پیش کرملی میگی شماره رو پیش پرینتر تونه ایستاده چک میکنه با یه ارتباط نرم افزار با نرم افزار اتفاق میافته

حالا الان سیستمی داریم که متفاوت میکنه اگه پیش اطلاعات بدی

پس هوتا اینکه سوالی که الان مطرح بود قسمت کدومش اینه که یه سری کاربران استفاده میکنن اما به صورت یقین تر، نرم افزارهای دیگه هم میتونن کاربر یه نرم افزار باشن

مخصوصا تو عصر دیجیتالی، کلمه platform رو خیلی میشنید مثلا صنعت بیمه میخواد با خیلی از اینا اطلاعات برقرار کنه. مثلا با بلیت فرم پرواز، یا Delivery که خودش یه سرویس هست و برای خودشون نرم افزار دارن دیگه یا تلفن بیت توی عصر دیجیتال این ایجاد کانکشن بین نرم افزارها اهمیت بسزایی داره و یه مفهومی مثل API economy یعنی اقتصاد مبتنی بر API که مخفف App programe هست و به نرم افزارهای دیگه اجازه میدیه از این استفاده کنن برای استعمال اطلاعات به جای دیگه و لازم نیست صفر تا صد بشو انجام بدین

از اول هر نرم افزار ارتباط خودشو داره و با هم ارتباط برقرار میکنن ← صفر تا صد شو نه! ارتباطاتی خودشو تو کلاس

یک از ویژگی های مهم مهندسی نرم افزار بدون هست و کمتر با مدیریت

مهندسی رشته های دیگر فرق داره مثلاً توانایی نیوتن روش محاسبه میکنه ولی نرم افزاری مثلاً

میگن چقدر قابلیت اطمینان داره که چه جوابی مثل استحکام سازه نیست.

مراحل ایجاد نرم افزار: چرا این سوال موضوعه برای این که دید شما در مورد یه محصول نرم افزاری روشن

کنه چون همینطور که برپایی کردن یه محصول ساختمانی روش های مختلفی وجود داره برای محصول های

نرم افزاری هم همینه و برای مدیریت بزرگ و پیچیده این بیشتر مفید داره. دو تا چیز توانایی

رو دشوارتر میکنه: اندازه و پیچیدگی که وقتی مثلاً نرم افزاری تولید کنید برای

یه خوشگاه کوچک تک کاربره یا برای خوشگاه رفاه برای تعداد کاربرهای زیاد و پیچیدگی

هم همینه یه بار الگوریتم هاش ساده اس و یه بار نرم افزار کلی رفتار مشتری های پیچیده خود

بدست آوردن کلی کلی پیچیدگی داره. وقتی محصول ما پیچیده شد نمیشه مثل یه محصول

ساده باهاش برخورد کرد. باید یه process model تعریف کنیم که مشخص باشه مرحله اولش اتخاذ

نیازمندی هاس مثلاً ولی تو ایجاد یه سازه مثل پنج مراحل مطالعاتی تدریجه و باید فرایندهای تولید

نرم افزار رو به کار ببریم نمیشه طراحی و تیم تست، کیفیت کلی کنترل داریم.

شما تو مرحله اول باید نیازهای مشترک رو استخراج کنید خروجی این یه سند هست و باید لوتو کامل کنید

مسائل: یه زمانی میگیم فرایند تولید نرم افزار یا توسعه نرم افزار: دو دیدگاه یا تو سطح تو رشته ما محصول

تولید رو دیده به معنای اکسترس دارتقاء نداد! همین: تولید هست و البته همین اگه فرایند تولید نرم افزار

واسه همین ما کلمه software product رو میگیم software Development میگیم به معنای

توسعه نرم افزار همون یعنی رو بیان واسه همین به زبان فارسی ما میگیم دو لوپه

فرق فرایند توسعه نرم افزار با متدولوژی توسعه نرم افزار چی هست؟ process ≠ methodology

مثلا تو توسعه نرم افزار متدولوژی هایی مثل RUP یا Agile میگیم مثلا

جواب: process زیر مجموعه متدولوژی هست و یه تعریف اش متدولوژی هست و جزئیات بیشتری رو در بر داره

وقتی میگیم متدولوژی تولید نرم افزار باید process یعنی اول و دوم چکار کنیم این میشه و باید بگیریم

در مرحله از چی استفاده چه زبانی، چه فرمی چه سلسله ای استفاده کنیم یعنی علاوه بر process روش

ها و تکنیک ها رو هم در متدولوژی میگیم، به عبارت خلاصه: متدولوژی جامع تر از فرایند توسعه نرم افزار

هست و در آن و در متدولوژی توسعه نرم افزار علاوه بر ارائه اجزای کار روش ها، ابزار و

استانداردها، قالب ها و دیگر جزئیات مشخص می شوند.

نقش نرم افزار رو هم فوتش مطالعه کنید.

کاپیتر بعدی: دیتا هستش دیگه و هیچ داده رو دخیل میدونیم پس نقش اصلی نرم افزار علاوه بر

گرفتن داده، نقش پردازش هم داره که اینجا نقش نرم افزار در جایی داده رو جایی که میتونند

نقش گرفتن، دادن اطلاعات باشه مثلا نرم افزاری که توی سوئیچ یا روتر کار داره

نقش نرم افزار اینه که اینارو کنترل کنه، آدرس ها رو یا نقش تولیدی داشته باشه

مثلا فاکتور رو پیش میبره و اون برای شما دیتا رو ایجاد میکنه و یا یه هم زمونی روش

/ /

مهندسی نرم‌افزار بیشتر توی چه بازار و چه زمانی تولید شده؟

تاریخچه نرم‌افزار

از این بحث اهمیت داره که توش توجه کنیم یک مهندسی و تولید نرم‌افزار رو و چیزی که برای ما اهمیت داره سه اتفاقاتی که تو مهندسی نرم‌افزار رو بگیم اینکه چرا می‌خوام سیستم‌لوژیک بنا‌ایم و پایه‌های مدل‌سازی رو داشته باشیم؟

برای انجام موفقیت‌آمیز و انجام پروژه‌ها ما قاطع داریم ۵ یکی هزینه زمان کیفیت

چه پروژه نرم‌افزاری چه هر نرم‌افزار دیگه‌ای و هر کدوم از این اضلاع خراب بشه بقیه‌اش هم خراب میشه و دیگه قابلیت استفاده منظره آلد بررسی کنیم نتایج تولید نرم‌افزاری رو نشون میده که با گسترش سخت‌افزارها امکان اینکه بشه بقیه نرم‌افزارهای پیچیده و بزرگتر خراهم کنند و با این کار پروژه‌ها هم بزرگتر شد و وقتی بزرگ شد و حامل رشته‌های دیگه روش‌های تولید و مراحل مختلف نداشتیم میشه حدس زد که نرخ شکست تولید پروژه‌های نرم‌افزاری بالا میره این اتفاق هست که افتاده کلا نرخ شکستش بالاست و در حال حاضر پیشرفت به بقیه رشته‌های دیگه و این حال حاضر بیشتر به علت لوس بودن بازار داره یعنی موجودیت انتزاعی اش و رقابتی که وجود داره برمیگرده و فعلا این که شما مهندسی نرم‌افزار نداشید باعث شکست میشه اینجا نوشته هزینه بالای ایجاد نرم‌افزار و کیفیت پرهزینه بحث یکی هست، پیشرفت سریع مثل ایجاد میکرد چون نرم‌افزارهای تازه ایجاد میکردیم نرم‌افزارهای قبلی از رده خارج میشد و از بین میرفت

افزایش سیستمی محصولات ما مثلاً چطور کنیم که مثلاً عیب های جدیدی که توسط مشتری ها

پیش بینی رفتار کاربران و اینا عیب های جدید اورده هست مثلاً اگه کسی بخواد الان اینارو

ایجاد کنه بهت بد شده و بدردی نمیخوره تو کنفرانس سال ۱۹۶۸ اومد اجرا شد و ترم های مهندسی

نرم افزار ایجاد شد که بیایم اینارو بررسی کنیم توی اون دوره که با نام بحران نرم افزار ازش یاد

میکنن.

مثلاً برای یک مناسب روی موجودیت نرم افزار و ذات اون اومدن با سخت افزار مقایسه این

کردن و برای دلیل اتخاذ کردن.

مورد اول خیلی مهم هست نرم افزار توسعه و طراحی میشود اما ساخته نمیشود و اسمش رو

میذاریم توسعه نه تولید یه محصول فیزیکی نیست که براساس قوانین داخل فیزیک بسازیمش

باید طراحی و توسعه داده بشه فرق مهم دیگه اینکه سخت افزار فیزیک هست، استهلاک داره

و کهنه میشه و باعث از دست دادن قابلیت های اسقاط میشه ولی کوفتش مسنوخ میشه

به نظرتون چرا وقتی یه نرم افزار مسنوخ میشه و میخاب میشه بذاریمش کنار چرا نیستم

فروش چنین سال قبل انداشتیم کنار؟ کلمه Requirement یا نیازمندیت یعنی نتونه

نیازهای کاربرانش رو برآورده کنه یک نیازمندی هست مثلاً من انتظار دارم قابلیت داخل نلگرام بهت

افزار پیشرفت کردن یا User friend بودن. کلاً بحث با کیفیت بودن الان با ۲۰ سال قبل فرق میکنه

که باعث میشه همه اینا نیازمندی های کاربر شما عوض میشه و انتظاراتش انتظارات میلی

پیدا میشه و تکنولوژی ها جدیدتر میشن و تغییرات توی محیط که یکی کسب تکنولوژی هست.

بنابراین نرخ خرابی یا مرگ محصول در اثر زمان:

حالا سخت افزار مظفرش سخت افزار کامپیوتر هست هر چیز فیزیکی یه موبایل یه چیز گوشت این

میگه ابتدای کار، این خود لفقادمان و این هم نرخ مرگ یعنی بالاره خرابی ها بیشتر

پایین بشه خرابی ها کمتره میگه ابتدای کار اطلاع ندارن خرابی ها بیشتر باشه همینطوری ابتدای

کار هرچقدر هم تست بشه، تست ملقی توی استفاده طولانی مدت کاربر معلوم میشه و

همیشه تحقق درنیاد و میگه تقریبا به یه حدی رسید دیگه لطفا ثابت میشه و خرابی های ابتدا

تولید گذشته نرخ خرابی مرگ پایینه بعد با یه شیب کمی رو به پایین حرکت میکنه

اما آخرش که دیره بالا تو یه مدت چرا؟ چون فرسوده میشه این برای محصول

فیزیکی، سخت افزاری هست که الیتم اون قسمت آخر رو محصول نرم افزاری نداره یعنی

ایده آل اونش مثل اون باشه. فقط ایده آل نیست به کف لفزار قسمت کمتر شکل رو

نباید داشته باشه توی تئوری ولی فقط ملقی بالا هست. ایده این که یکلال این

همیشه نرخ خرابی بالاست. این وسط چه نقطه ای هست اگه نرم افزار برای نیازهای

جدید بروزرسانی نشه، منسوخ میشه و دیگه استفاده نمیشه نرخ خرابی نداره ولی استفاده ازش مجبور

به آپدیت میشیم و تقریبا به این حرکت که رفته بالا شکل ذهنی باید هست و ارورهای جدید

پیامدی اینکه از منسوخ بودن خارج میشه اما احتمال دوباره خطا ایجاد کرد به بخش های اضافه میشه و

تغییر داده میشه و ممکنه تو نقل کردنا یکی نخ خرابی یا لازم بوده ولی دوباره پیدا میاد پایین و

جدا شدن نیازمندی های جدید یعنی این اتفاق دوباره تکرار میشه ...

این بالا رفتن و پایین اومدن چه وقت باعث میشه به منفی ایده آل رسید یعنی به نرم

افزار رو بگیم تا ایده آپدیت کرد و باید پیش کاری داره استفاده اش رو از دست میده بیت

به سمت افزار ملتش چیه؟ علت اینکه شما اصلا داره به نرم افزار رو بگیر بدید تا نیاز

مندی های جدید رو بهش اضافه کنید ولی تغییر تکنولوژی ممکنه معماری با موقعیت های

روبرو کنه که دیگه قابلیت تغییر و معماری جدید رو نداره و نیازمندی هارو آپدیت کرد.

تقریبا دنیای کنونی که الان بهش خواهید هست مثلا به صورت یکپارچه خواهید میگیرن ولی قبلا اما الان

ماکرو هست و هر تیکه اش رو توسعه کنید یعنی بدون اینکه به قسمت دیگر کار بیاید ولی تیکه رو

درست می کنید و به این تیکه مونولیتیک (یکپارچه) این تیکه ماکروسرویس یکپارچه

فرق میکنه. مثلا اگه با مونو باشه دیگه نمیشه ماکرو اش کرد چون نیاز به تغییر معماری

و اساسی داره گرچه میشه آپدیت کرد اما معمولا به سمت منسوخ شدن میره.

و با آپدیت کردن زمان منسوخ شدن اش رو به تاخیر می اندازیه

۳- سخت افزار مدتها پیش به بازار آمد و تهیه هزینه و مقدار زیادی از آن ساخته میشد اما نرم افزار عموماً سفارشی و بنابر نیازهای

مشتری ارائه میگردید شکل می‌گرفت و ساخته میشد و اساساً هم به نرم افزار جدی توجه نمی‌شد چون نرم افزار

و تولید و سفارشی بودن از این کلمه استفاده میشد همین مفهوم تولید بود البته

کاربردهای نرم افزار

۱- اجرای روبطی یعنی فرض کنید یه نرم افزار بنام ۲ که یه روبه اس هم کار میکنه حالا یک نرم افزار

هوش مصنوعی نیاز هر اجراشون متفاوت باشد

۲- ارتباطات مالی و کلی باشد نرم افزار بکار میرفته باشد یا متخصصات یه وضعیتی رو بیشتر دیگه

مثل نرم افزار بلاد بنک ۰

این اومده یه تقسیم بندی روی انواع نرم افزار کرده از دید پرسی : اینا بعضیا هم به مرور زمان نوع

جدیدی پیدا میشن ایجاد میشه

۱- نرم افزارهای سیستمی : یه نرم افزارهای پایه سیستم هستن مثل درایورهای سیستم عامل و ویروس

و اپلیکیشن سرور مثلا لینوکس آپاچی دارید دیتابیس هم یه نرم افزار سیستمی یا کامپایلر یا یه

سخت افزار اینا نزدیک هستن و به نرم افزارهای پایه سرویس میدن و مثال شلوغش سیستم عامل هست

۲- نرم افزارهای Real time : مشخصه اون ها زمان پاسخ کوتاه هست و معمولا برای پیدا کنترل

خط های صنعتی هست که بیشتر به سیستم انسانی اتصال دارن و مرتبط به سنسورهای دما، فشار و

جمع آوری میکنن خصوصیتش همین زمان کوتاه هست که تو وضعیت خطرناک قرار گرفت یک مانع نزدیک بده

2000 نرم افزارهای اطلاعاتی

روی هم رفته

۱۰/۱

توی محیط‌های فیزیکی یا مجازی اینا مهم هستن و باید سریعاً پاسخ بدن (Response time های کوتاه‌تر و سریع‌تر هست)

۳- بیشتر نرم‌افزارهای دور و برمون از این نوع هست که ترجمه کلمه business هست تجاری. منظور فقط

تجاری بیزنس و پاداش نیست منظور هر چی که به کسب و کار و به تجارت رو هندل میکنه مثلاً یه

آموزشگاه یا درمانگاه یا حسابداری هست چون داره کسب و کار رو سازمان رو مدیریت و

هندل میکنه، بیزنس به معنی کسب و کار اینجا بودن و بیزنس به سازمان رو انجام میدن به صورت

کامپیوتری و تبادل اطلاعات زیاد هست، شرکت نرم‌افزارهای سرویس دادن به بقیه نرم‌افزارها

هست و دومی هم که گفتم ثبت اطلاعات جدید، گرفتن از اطلاعات و ...

و تمرکز اصلی مهندسی نرم‌افزار بیشتر روی تجاری هست

مثل اتوکد

طراحی سازه‌ها

۴- مهندسی داده‌ها: پایه‌گذاری اطلاعات سرورکار داشتن برای هوشمندی، طراحی استفاده میشه مثلاً برای

۵- تعبیه شده embedded device های یه‌ری کنترل شون خودکار برای کنترل هر چیز هوشمند یا متصل کنترل برقرار کنند

شن (IOT) اینترنت اشیاء

۶- متدین Overlap باشن توی دوتا گروه باشن جدا باشن جدا باشن روی کامپیوترهای شخصی دستی

برای PC طراحی شدن مثل ورد، دفترچه تلفن از این موردها مثلاً شخصی‌سازی اینه که برای کارهای شخصی استفاده میشه

۷- متدین پروسیجر بیشتر از دوره قبلی متدین وب بیس باشن یا نباشن چون به حل معماری

نرم‌افزار هست و زیرگونه‌ای از معماری کلاینت سروری هست که دو قسمت سمت سرور و کلاینت که

این‌ها باهم ارتباط دارن فقط نکته اینه توی وب بیس، توی مرورگر اجرا میشه یعنی جاهام از کلمه فرانت

اند یا بک اند استفاده میشه که متدین وب بیس باشه یا به اپلیکیشن باشه توی مرورگر اجرا شه مثل اپ تمت

موبایل

۸- نرم افزارهای مختلف اش رفتارهایی هست که پشت هوش انسان باشه که البته تعریف اش سلیقه ای هست

کمی ولی به صورت کلی همین تعریف اش هست مثلا نزدیک ترین فروشگاه و لوکیشن Based هستن

یا مثلا بازی شطرنج یا خیلی از بازی ها یا تحلیل رفتار مشتری

(IEEE موسسه خاص برای برق و کامپیوتر)

توضیحات این تست توضیحات خیلی خوبی هستن و موضوع اول ماهیت اش رو بهونم چیه.

افسانه های سیریتیا: یعنی مدیرها عموما این اشتباهات رو میکنن در مورد اول مهم رشته مهندسی نرم افزار هم

لیست بالایی از کتاب ها که مراحل کار رو توضیح داده دارن و استاندارد سازی شده اون و متدولوژی

های مختلفی که وجود دارن و گفته چرا افسانه های سیریتیا؟ آیا استفاده می شود؟ این ها همه

وجود داره ولی آیا مدیر یه پروژه روی این کتاب ها آشنایی داره و استفاده میشه؟

۲- هزینه کامپیوترهای جدید خیلی کمتر از هزینه نیروی انسانی و با به عنوان اساس قیمت کامپیوتر پایه و نرم افزار قبل نوشته

اگر برنامه نویس عقب باشه ← اینجوری پیش و با برنامه نویس درست نمیشه سعی عقب افتادگی به خاطر

استفاده از متدولوژی اشتباه، طراحی نرم افزار رو مستند نکردیم، تجزیه و تحلیل و پیاده سازی قبول نکردیم

و نیازمندی ها رو بررسی نکردیم. زمانی موفق هست که یه نرم افزار نیازمندی های کاربر رو برطرف کنه اچ اقتصاد ترمافیم؟

افسانه های مشتری: بعضی وقت ها کاربرها نرم افزار برای همه مشتری ها میکنن اما وقتی هست که برای کاربرهای خاص هست سفارش یه سازمان همه چیز خوبی کرد مانیه

پروژه ها و مدل ← بعد نیازهای مکن یا دولتی و ترند آپدیت میشه ← دگرگونی تحقیق

زمانی موفق میشه که خود کاربر یا مشتری هم شرکت میکنه مخصوصا وقتی خواسته های کلی پیچیده ماهیت مشتری یا یه چیزی مشارکت کنه؟ توی تعریف مسئله توی پایه اول

خروجی های مشکله که به نیازمندی ها و خواسته ها میپردازه.

در غالب رویکرد یک چیز پیدا نمیشه هر دو مهم هستن و فقط گفتن اهداف کلی نباشه باید جزئیات رو با

با در نظر گرفتن این افسانه که ما شرکت نکنیم خیلی پیش میاد

وقت تلخ بود و وجود نیازهایی که از این نرم افزار داریم صحبت کنن، اهداف کلی و که همه میدونن جزئیات همه

اصطلاحاً جزئیات نیازمندی ها و متخصص رو بیاورد b مشتری.

افسانه های متخصص: یعنی ما یعنی قبلا بودن الان خیلی بهتر شده زودتر کمک رو شروع کن

اولی بدیه اصلی مشتری ایده ما

کار تمام میشه بله ولی برای پروژه های بزرگ و متوسط خیر چون کدنویسی رو براساس چی انجام

میدین؟ که اگه زود شروع کنید زود تمام میشه ما مرحله استخراج نیازمندی ها و ... داریم، تجزیه و تحلیل داریم

طراحی هارو داریم، تکنولوژی هایی که میخوایم استفاده کنیم. کار وقتی بزرگ میشه باید مراحل درست و خوبی هم تشخیص بدیم پس میگیم مهندسی نرم افزار

موقعیت قبل: چالش بودن متدولوژی به خطر رسیدن کار کردن بیشتر فقط افراد انجام کار چالش میشه / چرا گفتید زودتر شروع کنیم دیرتر تمام میشه؟ دوباره کاری

یا rework اتفاق میفته، اصلاح اساسی مثل معماری اتفاق میفته / مورد آخر: برای انجام پروژه هم برنامه تشخیص هم کلی مراحل دیگه کلی ارکان فنی لازم

فصل 2: فرایند process دارن.

software process

از مهم ترین فصل هاست و فصل سوال خیزی هست و از پایه ای هست این فصل در مهندسی نرم افزار

فرایند کاری ای که تولید پروژه نرم افزاری طی میشه تا به محصول میرسه. پروسس تولید یک محصول نرم افزاری

هست. تعریف خود proc رو گفته: نقشه ای که با پیروی ... مراحل کاری ای که نیاز هست انجام بدیم تا به محصول با کیفیت برسیم.

برای همه کسانی که درگیر پروژه نرم افزاری هستن اهمیت داره. کار پیشه دانشجوه استاد ولی صاحبان

دانشگاه هستند که صاحب پیشه اند چرا اهمیت داره؟ چون ما نیاز داریم وقتی محصول بزرگ میشه اگه فرایند درست پیروی نکنیم بخریده شکست پروژه

میشه حتما با کیفیت پایین یا زمان یا هزینه ما براساس فرایند انجام کار، سازمان دادن رو شکل میدیم اصطلاحاً پس مثلاً با زمان انجام کار

رو تعریف کنیم. مورد دیگه براساس همین می تونیم کارهامون رو پیگیری و کنترل کنیم میتونیم زمانبندی و افراد رو انجام بدیم سازمان دهی کنیم.

محصول فرایند: گفتیم software هست و به چیز دیگه اضافه کرده و اون اسناد هست و بخشی از اون

داکیومنت هایی که برای استفاده از نرم افزار نیاز مثل راهنمای کاربر بخشی از اون داده ها این ... و بخش مهم بود

هم بازده هم برنامه داره

(1) (2)

بیرتشون ability طرف در کامپیوتر اصطلاحات

هر مرحله یه خروجی یا محصولات میانی

داکیومنت های که در طول توسعه نرم افزار تولید میشن و کمک دارن های میانی هستن این خروجی ها عموماً داکیومنت هستن ، داکیومنت های تست و ... هم به عنوان خروجی کار هم در ارتباط با مراحل بعدی

همه کار دیگه و

چه پایه فیزیکی باشن یه هر حل جنس داده ای یت قابلیت نگهداری maintainability

دیگه استفاده میشه . یکی از ویژگی های اصلی ، قابلیت نگهداری نرم افزار هست اگر دیدن داکیومنت ها نمی تونیم سیستم نگهداری کنیم

کیفیت

تو مسئله ی هی جدید شل جابک ، داکیومنت حتی نداره بلکه سینیال رد شده بعضی ها میان هست ولی همیشه همشون طرف . هر چقدر اخروجی بزرگ و پیچیده تر نیاز به داکیومنت های بیشتر احساس میشه . پس برای قابلیت نگهداری حتماً باید مستندسازی بشه

نکته اطمینان از صحت و درستی کار : باید برای ارزیابی محصول تولید شده مراحل پایانی و خروجی مراحل قبلی هم مورد ارزیابی قرار میگیرن و داکیومنت ها و مدل ها ارزیابی میشه فقط محصول مورد آزمون قرار نمیگیره . دو مورد باید رعایت بشه

تعریف نرم افزار درست باشه ، طراحی هم درست باشه حالا درست هم پیاده سازی بشه هم باید نرم افزار درست

و خوب تعریف کنیم هم خوب پیاده سازی اش کنیم عموماً مراحل ارزیابی برای اینکه محصول رو درست تعریف

کنیم و تست های مراحل آخر برای اینکه نگاه کنیم درست محصول پیاده سازی شده یا نه . به تعریف از مهندسی نرم افزار اشاره کرده که این مهم هست .

۱- استفاده از اصول سازمان مهندسی : مهم ترین مشخصه ای که به روش و حقیقی میگه اینکه بشه برای بدست آوردن محصول توافق شده به روش مشخصی رو تعریف کرد که اندازه پذیر باشه مراحل مختلفش و مثلاً خروجی داریم برای

کیفیت پذیره شابه قطر سازه با این اوصاف هر روشی مراحلش مشخص و اندازه پذیر باشه پیش می رفتیم عقلانی

ولی مثلاً شعر اینطوری نیست چه فرایندی تعریف کنیم برایش . مقوله به وجود آوردن به بعضی چیزهای اشاره

کیفیت تضمین داره هزینه زمان کیفیت

میکنه ، مورد بعدی به کیفیت اشاره داره قابل اطمینان بودن و کیفیت داشتن و کارآمد بودن

تعریف دوم هم شابه تعریف اول هست و توسط مؤسسه IEEE که مثله بکار گیری ، سیستماتیک ، منظم ، کیفیت پذیر

تعریف رایج تر و مهم تر هست

ارزیابی پذیر

یعنی مهندسی باشه همه مراحل مشخص باشه.

رشته ما نسبت به رشته های دیگه کمتر مهندسی هست چون مدل ماتریکی نیست و از قوانین فیزیکی استفاده نمیکنه

مثلا نمیشه گفت قابلیت اطمینان نرم افزار اندازه بگیریم که تو تعریفش توافق هست ولی در عمل دقیق نداره

حالا برگردیم چی؟ برای توسعه، اجرا و پشتیبانی نرم افزار یعنی برای انجام A تا Z یه پروژه

نرم افزاری فقط برنامه نویسی نه کل مراحل. که اینجا توسعه گفت، اجرا، پیاده سازی، راه اندازی محیط

واقعی همه مراحل یعنی. هم بخش های فنی داره هم بخش های مدیریتی داره. نمیتونم همه مراحل یه پروژه نرم

افزاری رو پوشش بدم. پس یک قسمت اول: باید مهندسی باشه تمام همه مراحل رو پوشش بده.

سوال: گفتم توی هر مرحله یه خروجی تولید میشه که با لفظ تحویل داده میشن اون خروجی هایی که باید ارائه بشن مثل

داکیومنت، مدل، برنامه و چیزی. به انگلیسی چی میشه؟ Deliverable

تو چه سطحی باهم بحث کنیم؟ معمولا بحث های مهندسی نرم افزار که تئوری لایه بندی شده (layered) نوشته

شده. باید ببینم توی چه سطحی داره انجام میشه مثلا در مورد روش انجام کار، فرایند توسعه بعضی ها زیرمجموعه هستن.

اینجا به صورت یه فلوچارت لایه بندی شده هست لایه پایین: کیفیت quality - فرایندها process - شیوه Methods

و ابزارها tools. مثل اولی به صورت هم هست که مثلا اینجا اینطوری کشیده شده.

کیفیت یک زیربنای مهندسی نرم افزار هست بیشتر تمرکز روی کیفیت هست یعنی کیفیت رو تضمین کنیم.

توی لایه بعدی فرایندها هست که بعد از کیفیت که یک زیربنایی هست، بعدی process هست یعنی مدل

یعنی سطحی که بعد از کیفیت میاد

/ /

روش‌های فرایندی متفاوت برای بهبود و بهتر شدن مراحل کار و فرایند مشخص شده و باید بتونن روش‌ها

هست یعنی هر کدوم از مواردی که توی فرایند مشخص کرده حالا به چه روش‌هایی می‌تونیم انجام بدیم

پس مدل PROCESS به تولید مجری هر مرحله رو با چه روشی باید انجام بدیم مثلا از نمودارهای

Use Case استفاده می‌کنیم یا ماتریس نیازمندی‌ها این‌ها شیوه‌ها و روش‌های هستن که

برای هر مرحله می‌تونیم مشخص کنیم. مثلا تو مرحله طراحی از چی استفاده کنیم؟ مثلا از مدل مفهومی ER استفاده می‌کنیم.

مثلا تو تحلیل از دیاگرام‌های استفاده میشه...

مثلا تو ابزار از Rational Ros استفاده می‌کنیم. پس معلوم که دید به لایه‌های زیرساختی تری هست

که مدل بعد از اون مرحله رو بشه و بشه تا برسیم به مراحل بالاتر پس مدل لایه‌بندی مهندسی نرم

افزار. (کتاب اصول مدل هست). به ابزارهای کامپیوترهای در مهندسی نرم افزار گفته CASE

[Visio - Power designer - SA - EA - Archimate - Rational Ros] tools

برای مثال PRD در دیتابیس

غیر از ابزارهای هزینه پیاده سازی خیلی زیاد زبان‌های PM / دیتابیس اونا نرم افزارهای عمدتا فاز تحلیل و طراحی هست

که تلفیق

مهندسی نرم افزار با کمک کامپیوتر → case tools

CASE: Computer Aided Software Engineering. و تو فرمت هم انواع نرم افزار داریم.

زبان‌های مدلسازی استفاده میشه مثل UML و Archi دیگه فلوچارت خیلی استفاده نمیشه

اشاره کردیم مهندسی نرم افزار فقط برنامه‌نویسی نیست بلکه مراحل دیگه انجام کار رو هم پوشش میده سه مرحله داریم

تعریف what ، ایجاد اون توسعه دادنش و پشتیبانی کردنش نگهداری کردنش اینا مراحل کلی هست

فرایند فقط مراحل و تقدم و تاخر کارها رو مشخص نمیکنه جزئیات بیشتری داره هم فرایندها و مدل‌ها و روش‌ها و

ابزارها تعریف داده شده. و به صورت کلی این سه تا مرحله رو داریم. شناخت / توسعه (تولید) و طراحی / پیاده‌سازی

2000

اصلی، برای نمونه‌ی شوده

توی مرحله بارش یا ۶ تا ۵۰۰

توی مرحله تعریف به what می‌پردازیم یعنی اینکه سیستم نرم‌افزاری ما چه ویژگی‌هایی داره وقتی میگیم

متغیرات سیستم رو به من بگو یعنی متغیرات عددی داره. یکی متغیرات کارکردی هست

یا عملکردی هم پیش گفته میشه. ترجمه کلمه Functional هست. و غیر کارکردی هم non

functional هست. یه زمانی هست که میگیم اپ من باید زمان پاسخگویی به درخواست‌ها ثابت داشته

باشه یه زمانی هست میگیم این نرم‌افزار باید بتونه ۱۰۰۰ نفر رو تو ۳۵ پاسخگو باشه

این یک ویژگی غیر کارکردی و غیر عملکردی هست که توی طراحی نرم‌افزار قطعا تاثیر میذاره.

متغیرات غیر کارکردی اینا ویژگی‌هایی هستن نرم‌افزار پیاده‌سازی میشه به اینکه چه کاری انجام میده مثلا از نظر

امنیتی خوب باشه یا هر هفته رفع در دسترس باشه (غیر کارکردی) یا طاقت زیبا (غیر کارکردی)

سادگی میگیم باید این نرم‌افزار برای اطلاعات رو به چه شکل محاسبه کنه ← کارکردی

نیازمندی‌هایی که از جنس ویژگی هستن، یه هفته میبینیم می‌دیم این‌ها همه ویژگی

غیر کارکردی ← نیازمندی‌های ارتباطی (interface)

هاهستن. و مورد سوم ارتباطات لازم با دنیای خارج تعریف کنه، سرویس به ما سرویس

بده. یا شما سوبالی از کاربرم بگیم برای اون که ملی باشه. (شده تا: کارکردی، غیر کارکردی، ارتباطی)

مرحله توسعه سه تا کار عمده صورت می‌گیره، چیزی که میخام طراحی می‌کنم، توی مرحله بعد به

درون باید بپردازیم. الگوریتم‌ها، فرم‌ها، گرامرها، ساختار داده‌ها، اینا نوشته کدنویسی

فقط کدنویسی نیست. توسعه نرم‌افزار داره، مراحل کدنویسی و ... و نهایتاً آزمون.

آزمون هم سطح مختلفی داره. به هر تیکه نرم‌افزار آزمون اش چی می‌گفتیم؟ آزمون واحد / unit test / تست یکپارچگی / integration test

اصل طراحی، پشتیبانی، رفع خطا، ایجاد اصلاحاتی تو نرم‌افزار، تغییرات که نیاز داریم به

نرم‌افزار اضافه کنیم، خواسته‌های مشتری متفاوته. پشتیبانی هم تمرکزش رو change هست.

نیازمندی‌های جدید مثل اصلاح یا اشکالات کارکردی

چهار دسته رو شناسایی کرده، اصلاح، و تطابق که مثلاً با تغییرات محیطی خودمون تطابق بدیم

مورد بعدی بهبود و کیفیت هست یعنی خودمون به روز نگه داریم و افزایش امکانات بدیم. آخر شیفت‌ها

یا شیفت‌ها می‌کنم و برای بالا بردن کیفیت تست مقرعی رو دوباره کدنویسی مجدد می‌کنم.

پس یا مجبوریم که تغییر و اصلاح بدیم برای نیازمندی‌ها یا شرایط بیرون، یا به صورت کلی داریم

یا تغییر میدیم تا نرم‌افزارم منسوخ نشه.

تو این تست‌ها مدل‌های مختلف مهندسی نرم‌افزار رو تست کرده علاوه بر این سه مراحل عمومی که تست هم انجام

میشه به سری فعالیت‌ها توی پروژه‌های نرم‌افزاری داریم که بهش میگن فعالیت‌های چتری.

فعالیت‌های چتری تو کل زندگی پروژه وجود داشته پشتش به زندگی اشاره داره از شروع پروژه بدون تا انتهای پروژه

چون خیلی تعریف و توسعه و نگهداری می‌چینن برای فعالیت‌های چتری جمع بندی.

عموماً هم دیدیم هستن، تو کل زندگی پروژه وجود دارن. مورد بعدی تعریف کیفیت نرم‌افزار هست مثل آزمون و شناخت

نیازمندی جنس شون خیلی بهتر کلاً وجود داره.

2000

مرحله ۳ defined هست یعنی فرایند انجام کاری که تعریف شده است و تایید شده هست. و از یه متدولوژی تایید شده مثل RVP یا SCRUM استفاده کردم. این چهار مرحله تو دنیا می‌شناسن چون معروف و استاندارد شده است یا یه متدولوژی براساس فرم یه آدم با تجربه، فرم متدولوژی رو تعریف می‌کنم. اینم میشه فرایندمون تعریف شده است یه مورد هم ۴ مدیریت شده است. انجام کار طوری تعریف کردم که براساس این دلایل مختلف زمان افزایش و ارزیابی کنم و کیفیتشو بسنجم. یعنی فرایند انجام کار ما برای جزئیات زیاد تا کیفیت و اندازه‌گیری قابل انجام باشه. مرحله آخر ما بهینه‌سازی شده، یعنی چرخه بهبود مستمر داشته باشه. یعنی بازخورد بگیره از انجام کاره. مثل بلوغ قابلیت‌ها ← CMM

توی CMM یه چیزی به اسم KPA داریم یا حوزه کلیدی فرایند Key Process Area

برای هر سطح یه KPA تعریف می‌کنم اگه بخوام توی سطح 3 باشم این حوزه‌های کلیدی رو توی روند انجام کارم داشته باشم. هر کدوم از این چهار مرحله نداشته باشم نمی‌تونم بگم توی سطح 3 هستم.

مدیریت پیمان کاری: فرایند انجام روند یک دیگه انجام میشه. پیمان کار پایه. پایه‌ی نظارت مدیریت نیازمندی مدیریت تغییر فرایند به بهبود برساره. اگه بخوام برم سطح بعدی این ۷ تا مورد پایه اضافه شه سطح ۴ اندازه‌گیری هم بود یه مدیریت شده بود، KPA هایی که به اندازه‌گیری ربط داشت در نظر گرفته

فرایند انجام پروژه‌ی نرم‌افزاری یه سری مراحل داره که براش تایل و سیستم

هر رویدادی که روی موفقیت نرم افزار تأثیر بدی میذاره ریسک هست. ریسک یا فاکتور

ولی با problem فرق داره، ریسک هنوز اتفاق نیفتاده پایه بازبینی بشه به صورت دوره ای

ریسک اول اهمیت ریسک های بالا و پرخطر و با شدت زیاد هستند

و بعدش میگن action plan برای از دست دادن پروژه های کلیدی مثلا ۱ جایگزین

برای هر موضوع داریم و یه قسمت فقط یه نفر باشه. حالا برای وقوعش هم چیزی action plan

داشته.

ما چه مدل های فرایندی داریم حالا قبل از اینکه کتاب بیاد معرفی کند سطح رو بگیم

MM در CMM مخفف کلمه Maturity model مدلی که به شما میزان بلوغ رو میده اما در چی؟ مدل گروه

یعنی میان توی یه زمینه خاصی میزان بلوغ رو اندازه گیری می کنن. مثلا سایه بلوغ IT در سازمان،

روشنفکرانه که توش سطح بلوغ مقادیر تعریف کرده و مشخصا شده گفته که مثلا در سطح ۱ یا ۵ مثلا قرار

مدل بلوغ قابلیت

دارن. حالا اینجا کامل گفته ولی درستش بلوغه یعنی Capability Maturity Model

یعنی من برای فرایند توسعه نرم افزاری ام سطوح بلوغ مختلفی تعریف می کنم. این که هم فرایندی

مرحله ای نداشته باشید اون میشه مرحله ۰. بعد فرایند توسعه نرم افزار

سطح ۱ یعنی یه فرایندی تعریف کرده باشیم اما روش نامنظم و انفرادی باشه سطح ۲ یعنی

این فرایند توسعه از مرحله اولیه بالاتر بوده باشه استاندارد شده باشه تا بر اساس اون زمانم رو

تخمین بزنیم. و چیزی فرایند قابل تکرار بودن جزئیاتی اضافه شده که بتونیم تخمین زد

در مورد مدیریت طرح نرم افزاری یک از اساتیدی که توی ابتدای پروژه نرم افزاری هست یکی از این اسناد

مهم هست و نه پروژه های کوچک بلکه متوسط رو به بزرگ و توی ابتدای پروژه برنامه ریزی پروژه ایجاد میشه

توی کنترل پروژه، مدیریت کیفیت، تدارکات اینا بهتر میشه. آماده سازی اسناد: که گفتم

مدیریت استفاده مجدد: چیزی که توی سال های دیگه هم استفاده بشه

پایگاه های فنی و ...: مثل ریتاپیس اینا رو ارزیابی میکنیم.

و مدیریت خطر: چون ریسک هست، هر چیزی که روی موفقیت پروژه تاثیر بد میذاره.

بایه واحد مشخص براش داشته باشیم توی PMO مثلا شدت ریسک هزار و ... و دقت کنید ریسک

با Problem فرق داره. ریسک هنوز اتفاق نیافتاده. باید احتمال وقوعش ارزیابی بشه.

عبارت آخر کتاب به نظرتون چیه؟ تفاوت این مراحل چیزی هست.

در مورد فرایند توسعه نرم افزار صحبت کنیم: قبل این یه بحثی با نام سطوح تکامل نرم افزار گفت

/ /

نتیجه یکسری مراحل مشترک تو همه مدل های مراحل انجام پروژه نرم افزاری وجود نباید تعریف مراحل چیزی متفاوت باشد

مثلا تعریف نرم افزار نیست چرا متفاوته؟ چون فرایند انجام کار رو براساس شرایط می چینیم مثلا تو شرایطی

که فقط سرعت مهمه وقت که، یا سرعت مهم نیست میخوام این پروژه نرم افزار اینست و خطاش زیاد

مهم نباشه اینا شرایط مختلف هست دیگه. تو همه مراحل عمومی رو داریم ولی نحوه انجام مراحل مختلفه و مدل های

مختلف وجود داره. مثلا برای انجام پروژه های سریع تر چابک برای پروژه های تحقیقاتی تر، کلی توی

حوزه تحقیقی قرار میگیره که نتوی میدونه دقیقا چی میخواد تعریف مسأله رو طوری انجام میده که تحقیق باشه

این: اجالتی که تحقیق میدونه چی میخواد متفاوت هست، مدل های فرایند ارائه شده اینا به کار گیری شدن طی

زمان دوما برای هر شرایطی بعضی از این مدل ها قابلیت بکارگیری دارن بعضی ها ندارن و هر مدل توی

شرایط ویژه ای قابلیت استفاده داره.

اولین مدل رایج ترین مدل بوده که به دهه ۶۰ میرسه تا الان ولی توی خیلی از سال های گذشته این مدل رایج

پروژه های نرم افزاری هست، مدل خطی آبشاری Sequential یا ترتیبی. مراحل تولید نرم افزار پشت

سر هم انجام شه. هر مدلی که بتونا گفته (مرحله) بود یه مرحله بعدیشا نمیتونم انجام مدل خطی بود یعنی

ماهیت انجام روش کار به صورت خطی و پشت سر هم یعنی تموم میشه. انجام یه پروژه نرم افزاری

مدل خطی به ایراد بزرگی داره با توجه به اینکه خروجی های هر مرحله برای انجام مرحله بعد مورد نیاز هستا

کلیه: نیازها شناسایی میشن و کیفیت میشن و مدل میشه تا تایید فهم بشه یعنی از خروجی ها براساس مدل های

واسطی که هست میتونه ایجاد بشه که قابل فهم باشه.

دانش کارهای مختلفی دارد مثل تحلیل که Analysis هست فرض کنید خیلی چیزها دارد. یا تحلیل نیازهای

نرم‌افزاری که اینجا آمده اما قبلش تحلیل نیازهای کسب و کاری هست. که پیش‌نیاز Business Anal

تحلیل کسب و کار. هدفش اینه که: برای یک کسب و کار هست.

پس تحلیل هر کدوم دارد تحلیل قبل این چیزی که اینجا نوشته / تحلیل کسب و کار یعنی اولین

مرحله فرایندهای کاری دیدیم و ... و هدفش تحلیل نیازهای کسب و کار هست. بعد از این هر کدام نرم‌افزار

Software requirement Anal تحلیل نیازمندی‌های نرم‌افزاری که اینجا نظرش همون بوده.

را هم What هست. هر کدوم کتاب هم دارن مثلا برای Busine Analy دو دوره

استاندارد PMI و BABOK داریم. PMI - project managem Instit

Busin Analy Body of Knowledge به درد مدیریت کار میخوره.

تکنیک‌های خیلی کسب و کار خیلی تحلیل هستن مثلا به زبان uml، usCase هارو می‌گیریم

طراحی به پیوست نرم‌افزار جیپ‌ها. مرحله به مرحله پارسنوس و اگر فرض هم گفتیم

bpmn → آیا در سازمان فرایند کسب و کار

مدل خطی مدل اینه دیگه مثلا بعدش. مرحله پیشرفته‌ای هم هست آزمون فرایندی اضافه کن.

چیه مدل خطی چیه؟ چرا توسعه‌دهنده‌های کمتری استفاده می‌شه؟ مهم‌ترین حسن سادگی هست.

مدل نکات اصلی اینه مدیریت پروژه اسانی مشکل داره و نیازات نشده اعمال کردیاخیلی

و با اینکه توسعه‌دهنده‌ها نیازات دارد framework چیزهم. تغییرات توسعه‌دهنده‌های بزرگ بسیار پرهزینه.

مدیریت پروژه اسانی مشکل داره — 2000 چطوری بعد از مدل خطی استفاده شد که تغییرات کمتر اتفاق بیافتد تا

پروژه‌های کوچک و متوسط. مدل مرحله‌ای شکل اساسی پاچ نده یعنی پروژه‌هایی که تغییرات کم توسعه اتفاق بیافتد مدل

Year. Month. Date. Subject

حالا چه موقع و توی چه جایی یکی تحلیل در مدل خطی رو خوب انجام بدیم وقتی که درست بتونیم نیازمندی ها رو تعریف کنیم و یعنی گاهی برای شرکتی ممکن هست که نیازمندی های خودشو تعریف نکنه. این بعدا میشه دوباره کاری Rework و دوباره کاری. ولی چرا الان از رده خارج شده؟ چون مشتری دقیقاً میدونه چی میخواد و به تکنولوژی خاصی شدن رو میخوای به تکنولوژی جدید پیاده کنی تو اینجا استفاده از مدل خطی توجیه داره چون مرحله تحلیل رو دارم کامل. از لحاظ کارکردی، مدل جدید می‌خوام پس توجیه داره.

مثلا یه پروژه خیلی بزرگ دارم ولی دقیق میدونم چی می‌خوام تحلیل کامله ولی خیلی پروژه بزرگه آیا استفاده از مدل خطی توصیه میشه؟ خیر. چون دنیای کنونی تو حوزه‌های مختلف پیشرفت‌های سریع دارن اتفاق میافته مثل حوزه‌های کسب و کاری و تو حوزه‌ها، خواسته ارزایی سلامت مفاهیم جدید و روش‌های جدید دارن که تکنولوژی ساپورت شون میکنه تو دنیای کنونی هر وقت زمان طولانی میشه ایجاد تغییرات تو پروژه‌ها به صورت ذاتی ایجاد میشه. چون زمان با خودش تغییرات میاره.

گرچه استفاده نمیشه و خیلی کم استفاده میشه ولی اصلاً مبنای مدل‌های دیگه بوده و به خاطر عیب‌هایی که داشته سراغ مدل‌های دیگه رفتن بعیش هم گشتن چرخه حیات کلاسیک. مدل‌های قدیمی مثلا قبل سال 2000 همه اینا اینطوری ایجاد شدن و Rework هم خودن. و نکته دیگه change توی پروژه‌های نرم افزاری هست.

چرا متدولوژی‌های Agile مورد توجه‌اند؟ یکی چون تغییرات تعداد چون خیلی همه

برای برنامه ریزی کار، مثلا اگه خواستید مدل خطی رو هم دلیل به اول قسمت هایی رو تحلیل کنید که تغییرات کمتر باشد. خروجی های نهایی رو ارائه کنید و نظرسنجی کنید یکی از موضوعاتی که میتونه رسک change رو تو پروژه هاتون کمتر بکنه همراه کردن مشتری با خودتون هست.

چون مشتری کسب و کار رو خیلی تو دست میاره، خروجی های غلطی نظرم کسی نداره! شدت طلبه بذارم و مدل هارو بعدش نشون بدید توضیح بخواید.

الان که مدل کلاسیک (خطی) رو توضیح دادیم مدل های دیگه رو سریع تر بررسی می کنیم. یه روش دیگه هم پروتوتایپ هست. مدل فرایندی اش به این صورت هست که: یه مدل چرخشی بقیه اش مفید بعد

پروتوتایپ: یه مدل پیچیده که هنوز کار نمیکنه: محاوره ساخت نمونه (prototyp) ← ارزیابی مشتری

نمونه اولیه

prototype: یعنی یه خروجی مشابه نسخه نهایی که میتونه ناقص و باگ دار باشه هدف اینه

که مشتری بتونه ارزیابی کنه و نظر بده.

چه موقع استفاده؟ اگه اهداف کلی ذکر شود یعنی خود مشتری نمیدونه چی میخواد ما براش توضیح میدیم و نمیتونه بیارم

نمونه ۲ پیاده سازی اش دیگه کامل تر میشه

تو نمونه های اول پیاده سازی عمق نداره. نمونه proto خیلی کامل نیست چون همه مراحل رو مثل آرشیو نداره

و یک ایده هست، یا اینکه یه مدل تولید نرم افزار نیست شده از تکنیک های استخراج نیازمندی ها دست مشتری

فرایند

مراحله بعدی استفاده میشه

معایب: بعضی عیب ها رو میشه کنترل کرد و مشتری مثلا تو چه چیزی که بیش نشان میداری باشه عیب

اش اینه که نیازمندی ها اصلی رو یادش بره و باید اونا رو گوش کنه. مورد دوم ما می خوام تو این

مرحله کند باشیم و می خوام سریع باشیم و ما فقط اصلی ها رو پیاده کنیم و از روش های ناکارآمد استفاده کنیم

که مشکل کنترل کرد. معمولا پروژه های بزرگ بدون پیاده out sourcing میشن و بقیه شرکت ها انجام میدن

سند توصیف نرم افزاری هست.

مدل RAD: یه تطبیقی از مدل خطی هست ولی تغییری داده شده تا بیش سرعت اضافه بشه همون پیش

نیازهای ارتقاء در این مدل هست مثل بعضی نیازمندی هایی که دچار تغییرات میشن، و زمان اگه

طولانی بشه مشکل داره. و زمان کوتاه این مدل مناسبه. که بتونید یه تقسیم بندی انجام بدید تو کاربرتون و

هر کی یه کاری رو بسپارید پس شرایط به کار گرفتن مدل RAD چیه؟ نیازمندی ها قابل شناسایی اند کم

زمان کم / به بخش هایی شکوند که بشه به تیم های مستقل رو بدهد.

/ /

اینا موازی با هم کار میکنن و به مرحله یکپارچه سازی میرسن تا کارشون تموم شد کل این پروسه میشه.

این ۵ تا مرحله رو مثال زدم که دیدتون عوض شه قطعی نیست پس این اینه که تیم‌ها موازی

و مراحل خطی هست. اولی Business Model یعنی کسب و کار هست یه مدلی از این پروژه مجبوری

Data Model موجودیت های داده ای رو بدست میاری و اگه در ارتباط هستن مشخص کنی.

سه مرحله اول معادل what هست. این سه تا یعنی Business / Data / Proc و Application توی مرحله توسعه هست

بعدی Testing هست نرم افزار آموزش دیده تست و راه اندازی کنی. اگه میشه همیشه مدل خطی که

RAD میشه رو پیاده کرد چرا باید از RAD استفاده شه؟ چون نیازمندی های خودشون خواد تغییر کوتاه مدتی بشه کلا ازش دارن.

یه اشتباه: چون بیکاری زیاد هست کیفیت قحطی هست. برای مثال میدیم پروژه خیلی طول بکشه داریم

گفتم زود بشه روی یه کار و تخصص لازم بدست بیاریم و خوبی اش اینه که بدون نیرو رو براتون بازهسا

ایرادات این مدل مدل HR هست و بعدی توافق هر طرف. پس RAD از خطی هزینه بیشتری میشه.

Human Resource

هر موقع وارد سرعت می شیم برای پروژه های پیچیده قابل استفاده نیست و نیست فاکتور ریسک رو وارد

بین قضیه کرده یا وقتی کدوسک فنی داریم باید جای دیگه تست بشه و ریسک فنی بالا باشه این روش

خوب نیست.

RAD و خطی سه مدل خطی

دو مدل قبلی جنبه شون بازیست به دلیل تبلی نداری و مدل های تکاملی در مقابل داریم evolutional

هر مرحله یک بار انجام نمیشه بلکه تکرار میشه تو مدل خطی یه بار مرحله یکی رو داشتیم یا مدل RAD این مرحله

یه بار انجام میشه و تو یک زمان شروع و پایان مدل کسب و کار داری در مقابل تکاملی داخل کار تکرار میشه و پروتوتایپ

2000

هم تکاملی هست مثلا ساخت نمونه های تکرار میشه

مدل های تکاملی در دنیای الان رایج هست و قطعاً یکی از مدل های تکاملی را استفاده می کنید. و ذات مدل
مراحلی که باید انجام بدیم یک طراحی هم تکرار شه
های تکاملی این هست که شکل به مراحل کوچکتر و چرخه داریم تو مدل های تکاملی یا psycle
یا iterati داریم. یا قدم گذاشتن increment (افزایش) مدل های تکاملی هم تو زمان های
هزینه که برای تولید
مختلف تکرار میشه و نقطه قوت مدل های تکاملی تولید هست. مثل پیش مدل های RAD
تکاملی
هست هر کدوم از اینها مارو داره جلو. و بعضی ها اومده هزینه شده این Analys رو می بینید
اینجا توی تکرار دوم این جلوتر از اینه داریم اینو انجام میدیم به مراحل بالا رسیده. توی تکاملی
مثلا تقریبی
اگرچه increment به صورت خطی هست ولی بخاطر اینکه افزایش های بعدی اختلاف زمانی با قبلی دارن تو مرحله تکرار دوم میشم
شما افزایش 1
این به من کمک میکنه که نتایج کلیدی مراحل قبل دارم و از یه چیزی مطمئن شدم اگه اشکالی از اون
مراحل متوجه شدم میتونم اصلاح کنم. مثلا اشتباهاتی از مرحله آماده 1 بود توی افزایش 2 اصلاح می کنم.
پس از خروجی های مرحله قبلی در افزایش 1 توی افزایش 2 استفاده می کنم.

هر increm بره پایین تر، خروجی اش از مراحل بالایی بهتره. مثلا خروجی افزایش 5 از 4 بهتر و کامل تر
نسبت به مدل خطی این امتیاز که زودتر به مدل خطی میرسم. مدیرها درکی از مراحل کلیدی و طراحی
ندارن. مثلا ماه ششم هستیم و هنوز دو ماه که کلیدی و طراحی و 4 ماه پیاده سازی کردیم هنوز تمام نشده و میگه شما هنوز
یه هفته دیگه افزایش نشون ندادید. کلیدی و طراحی خوب تا 30، 40 درصد پروژه هست این روش خوبی است

اینکه یه نتیجه مثلا برای تست داره و ریسک های کار رو کم میکنه و یکی از دیگه از مزایا ریسک

های ناشی از تغییرات و ریسک های ارزیابی پیشرفت پروژه رو پایین میاره.

پس وقتی کدمن تیم یه خصوصی رو شروع بزنم. کلا همه هرچه زودتر نتایج ملموس به مدیران

بدیم و اینکه مدیرها آشنا باشند ریسک این مورد رو کم میکنه.

یکی دیگه از طراحی ها توی مدل های تکاملی کمک میکنه دیزاین ها تو زمان واقعی کار میکنن و برای

بیزنس یکی دیگه طراحی مثل کلا تعریف میکنه و تا انتهای زمان و برای Rework اعداد و

نگه میداریم مثلا این زمان رو بدیم همه بیزنس ها تو اقدامی ؟ تست می کنن هم کدینگ

توی 2 و یکی 3 و ... دیزاین ها رو جدیدتر به کار میبرن کریم و هر بیزنسی توی زمان مناسب

خودش روی پروژه کار میکنه. مدیریت منابع انسانی وقتی قوی باشه چون آدم های مختلف

تو زمان های مختلف میان. چند تا مدل دیگه هم وجود داره. ولی پس چیزهایی که تو خانه

مدیریت نرم افزار بود اینجا هستم.

نقطه قوت اصلی تکاملی: مدیریت کردن change

و پایین آوردن ریسک مثل ارزیابی پروژه

Year. Month. Date. Subject

ارتباط با مشتری منظورش فقط شناسایی نیازهاست. و بله با مشتری ها ارتباط داشته هم میتونه

جالب هم باشه هم فایده کاربران. مثلا میتونه فقط مشتری نباشه و چیزهای دیگه هم باشه مثلا از

سوابق فوتبال یا سوابق تیم ها و تجارب اون ها. توی فاز بعدی طرح دیتای فاکتور برای پیچیدگی

جاری که بیایم و نظام پیچیده طراحی رو انجام بدیم برنامه ریزی کنیم. بعدی تحلیل ریسک هست.

مدل‌های فرایند نرم افزار صحبت کردیم، گفتیم دو تقسیم‌بندی کلی می‌تونیم داشته باشیم مدل‌های خطی و تکاملی

تکاملی، مراحل تولید سیستم هی تکرار میشه یعنی اینه واقعی داریم فقط یه بار تکرار نمیشه چند بار تکرار میشه و این تکرار

مدل‌های مختلفی ایجاد میکنه مثلا مدل پیچیدگی یکی تموم شد بعدی شروع شد یا مدل overlap داشته

باشه چرخه‌ها، چرخه 1 تموم نشده بعدی شروع شن میگن مدل افزایشی.

تو مدل افزایشی ما چند تا iteration داریم و بعدی میگن increment. انتهای هر افزایش یه نسخه داریم مثلا

انتهای increment 1 هم نسخه‌ای برات میاد و بعدی که کامل‌تر از بالایی میشه.

یکی از خوبی‌های مدل تکاملی بیشتر به خاطر اینکه زودتر به چیزی می‌رسید که به عنوان محصول می‌تونید ارائه

اش بکنید برای کاربران، کاربرها که اونو ببینند و نظر بدن.

مدل مارپیچی: یا خودش به شکل دایره‌های پشت هم قرار گرفته میشه، یا رنگ‌های مختلف...

مدل مارپیچی صرف نظر از اینکه تو هر دایره چه مراحلی داشته باشه و — یه ایده هست که شما چرخه سیستم رو

تکرار بکنید. این تکرار به صورتی هست که یه تکرار باید کامل تموم بشه تا بعدی شروع بشه و overlap نداشته باشیم بودن

و قطعا تعداد پیچ‌ها خیلی زیاد میشه. تو حالت‌هایی دیگه خود مدل هرچی میتونه باشه مثلا مدل فیبی چی بود

آتاشه دیزاین و کدینگ و تست و برای مارپیچی باهمش مدل ارائه شده و هر چه قدر توانید سیستمی رو مدل مدل فیبی

رو می‌تونید اینجا بذارید. پس مدل 6 تا رو انجام میدن، مرحله به مرحله تکرار میشه اینکه چند تا بار شدن تا

بستگی به میزان بزرگی پروژه و موارد دیگه داره یعنی تعداد تکرارها iter و تو مدل فیبی incr ها و قطعا

تعدادش خیلی زیاد نیست چون بچه نیست و برای پروژه‌ای سخت میشه مثلا تا 10 تا.

ریسک کلمه همون انگلیسی و یا مخاطره هست ، هر چیزی که روی موفقیت آینده پروژه تأثیر منفی میذاره به عنوان ریسک شناخته میشه ، مثلا جای زلزله خیز باشه ، یا مدیریتی باشه برنامه ریزی درستی نشه ، یا نتونید یا فنی نفرات از ابتدا خوبی برقرار بکنید ، اینا ریسک های مدیریتی هستن ، یا ریسک های فنی ، مدیریتی ، محیطی و — فن مثل استفاده از تکنولوژی های قدیمی ، تو ریسک ، باید شناسایی بشن ، بعد اولویت بندی بشن با دو فاکتور : ۱- احتمال وقوع ۲- شدت ریسک . یعنی ریسک ها پایین و پروژه دچار خلل نمیشه ولی بدترین حالت اونی که شدت بالا (تأثیر زیاد رو پروژه) و احتمال وقوع شون هم بالاست ، بعدش چی ؟ باید برا شون action plan در نظر بگیریم ، که اتفاق افتادن چکار کنیم و جلوگیری از شکلگیری اینها قبل وقوع پیش بیاد چکار کنیم ...

مراحل بعدی مهندسی و ساخت و ارائه . مهندسی همون یعنی طراحی Analysis و design هست بعد به تیم توسعه میدیم . ساخت و هدف و تحویل development که دست بدست و ارزیابی مشتری ، تو محیط شبیه سازی شده مشتری نظر و بازخورد اعلام میکنه . و تکرار میشه بعد اتمام هر کدوم بعدی شروع میشه . اینا برای هدایت به هدفی قابل شناخت مثلا تو مثبت سفر ریسک ، پروژه ای نگهداری میوه است

مهایب روشاست و برایا همون ترتیبی مدل های تکاملی هست . تابع کردن مشتری ، که البته تا قبل مدل پیچیده ای هست هر مدلی که توش ارزیابی مشتری داشته باشه و بخوام مشتری رو دخیل بکنم این جزو دلیل ، علت است اصلاح تست کردن از نوادن جایی فاز ارزیابی تری هست که به عنوان ریسک شناخته میشه باید حسابت و پیشگیری های انجام میشه .

Subject

Year. Month. Date.

مدل مارپیچی Win_Win : این که فازهایی باشن مدل مارپیچی هست . win یعنی مشارکت تیم مشتری در کار استارت داره . پس اینو بچسبه میگه که مشتری کار پروسه تولید به صورت فعال شرکت بکنه . ما باید یه تیم توی از مشتری setup کنیم و وارد بیم . و نیازمندی ها رو بر طرف میشه و تغییر شرایط رو اشاره به نیازمندی ها داره توی هر پیچیدگی بخشی که باید به محصول اضافه شه مشخص میشه و نیازمندی ها رو دیدیت ها و ازالت رو

تو مرحله آخر چیزی که انجام میشه اعتبارسنجی هست . یکی verification یعنی صحت سنجی و Valid یعنی اعتبار سنجی ، ولی یعنی مطابق با طراحی ای که انجام شده هست محصول مورد نظر و با توجه به این اسناد هست می کنیم . این مراحل تایید شده یا نه . اعتبارسنجی یعنی آیا نیاز کاربر رو برطرف میکنه و نیاز کاربر رو رفع میکنه

مرحله 6 هست Valid و مرحله Verif تو مرحله قبل انجام شده .

مدل توسعه همروند : به عنوان یه مدل توسعه فرایند جامع . تو توسعه همروند concurrent مؤلفه های مختلف نرم افزاری به صورت موازی توسعه داده میشن مشخص تو پروژه ها میشه همه موازی سازی ها رو رعایت کرد . مثلا توی سیستم مالی این بخش پایه ای هست و باید زودتر انجام بشه . توی مدل توسعه همروند چیزی که مهمه اینه که کدوم مؤلفه های نرم افزاری رو موازی با هم توسعه بدیم . مثلا تو حوزه بانکی مؤلفه مشتریان خیلی پایه ای هست . مؤلفه های نرم افزاری که استقلال دارن رو توسعه به صورت موازی بدیم . ولی مثلا بعضی یه مؤلفه نرم افزاری رو مفروض پس یعنی مؤلفه هایی که جدا از هم توسعه میدیم شناسایی میکنیم

نکته : این مدل یه ایرادی هست که مؤلفه هایی که استقلال نسبی بیشتری با هم دارند به صورت موازی ازشون توسعه بدیم که مسائل مختلف پیش رو آورده . مؤلفه (Component)

Year. Month. Date. Subject

کنار قرار دادن IC ها در Component based و در software هم اومد واسه‌ی بیش بردن نرم‌افزاری

و قابلیت استفاده بود هم ترین ویژگی یک Compon هست reusability هست و توجه‌ی‌های مختلف

استفاده بشه به صورت جداگانه.

یه نمودار حالت هست پیدا کنید اینجا state chart

Compon یک مقدار هست توی مدل قبلی تمرکز روی معماری بود اما اینجا معماری انجام شدن مدنظر نیست

توی مدل A، معماری انجام شدن مدنظر نیست تمرکز روی Component based انجام شدن هست و اجزای

مجزا که کنار هم قرار میگیرن و شکست افراد یک شکل کلی میده. و این قسمت باگ، خرج و مصرف و

مشکل رو میگیره یه کامپونت. این مؤلفه‌ها یه نرم‌افزار و تولید کردن ایده‌آلی روش هست

توسعه قبلی بر اجزاء: این روش روی نرم‌افزار به شکل اجزاء ایده‌آلی روش هست و میتونه روی

هر مدل دیگه‌ای پیاده‌سازی بشه؟ کسی که این مدل رو پیاده کرده روی مدل خودش پیاده کرده و میتونه براساس نیاز

در پروژه تغییر کنه. پس هر تکرار میشه ساخت یه مؤلفه Compo

ارتباط باتری - طرح ریزی - کلیل ریک - کدنویسی - ساخت و تحلیل

مدل روش های رسمی Formal Met هست، که همین روش های مورد هم میگن به طور رسمی.

و فورمیش یک شاخه علوم کامپیوتر هست. به عنوان یه دوره اصلی علوم کامپیوتر کتاب های متعددی درباره

این نوشته شده. ایده رئال مدت ها اینه که نیازهای یک سیستم رو به صورت ریاضیاتی و دقیق تعریف

کنیم یک سیستم نیازمندی ای رو سلامت برطرف کرده یا نه رو اثبات بکنیم. و زبان هایی هم هست

بهش میگن زبان های ریاضیاتی. تو دوره های خاص مثل پروتکل های امنیتی استفاده میشه و تعریف بالایی هست. مثل دو

مورد آخر: مثلا پایین ترین سطح مثل زبان اسمبلی یا زبان ماشین کار میکنه. نزدیک به سخت افزار. هر چی

بیشتر فاصله بگیریم و به زبان انسان نزدیک میشیم زبان سطح بالا گفته میشه. ابزار مثل C که جزو از زبان های

سطح بالا هم سطح بالا تر هست. منظورمان اینه نیازمندی ها دقیق تر از یه نرم افزار رو در سطح بالا

ارائه. به یه نرم افزار می دیم و این نرم افزار رو در نظر تولید میکنه پس یه موجودیتی به نام مثل C

رو تولید میکنه. و به صورت اتوماتیک این نرم افزار برای من تولید میکنه. و امکان داره همه بخش

ها رو تولید نکنه یه بخش باشه. یکی از مثال هاش سایت سازها هستن. یعنی بخش هایی از یه نرم افزار رو

شما مشخص میکنید و اون برای من ایجاد میکنه. و جایی و زیبایی داره. مثلا پروژه های

بزرگ رو میشه و همه اش رو دستی بنویسن کسانی که این ها رو نوشتن برای نرم افزارهای ساده و شخصی

نوشتن مثلا ابزار سیستمی یا بانک ساز نداریم. مثلا یه نرم افزار بنویسه زبان های پیچیده ای مثل زبان فلو یا پایتون

ادعای مخالفین: فرایند مطلوب رو زبان گذاشته و تبدیل کنیم و کار آسونیه چون ساختن یه نرم افزار هست و یه دو اولگوریتم داره که بهتر نداریم چون

همه حالات رو باید پوشش بده و بحث همین دست زده. مثلا اگه شتری ابعاد داره در زمان طراحی از روش تکاملی.

چرا بحث مدیریت پروژه‌ای تو مهندسی نرم‌افزار زیاد؟ چون هدف انجام پروژه هست از A تا Z و باید همه رو پوشش بدیم

مجموعه وظایف مدیریت پروژه‌ای داشته باشیم ۱۰۰۰ هم بحث فنی داره هم بحث مدیریتی. صفر تا کلاس‌ها

plann و monitor هست. انحرافات در برنامه‌ریزی‌های فعالیت پروژه و بودجه و زمان و مدیریت کنترل

فرایند‌ها و رویدادها. مدیر فنی + مدیریت بخش‌های فنی مدیر ارشد و مدیران پروژه. که بالای مدیر فنی

هست. بالاترین قسمت مدیر پروژه هست و انواع مدیرهای دیگه هم هست. سطوح بعدی هم سطوح کارشناسی خواهد بود

مهندس نرم‌افزار

PMO

برای انجام مدیریت خوب باشد موضوع زمان و کیفیت خراب نشه. مؤلفه‌های مدیریت پروژه نرم‌افزاری 4P

این ۴ تا رکن رو: ۱- افراد ۲- محصول ۳- produ ۴- فرایند ۴- پروژه هست. ۱- افراد: یکی از ارکان

اصلی هست تو ایران که پیدا کردن نیروی خوب سخت شده. مثلاً مدل رشد براش جدی کرده. محصول که اساس

اون و نیازمندی‌ها و گرفتن تایم و زمان تأثیر اصلی داره

مدل بعدی فرایند هست مثلاً incremental هست یا خطی که دول گرفتن سناریوش فرق میکنه، وظایف و زمانبندی

و همه این‌ها فرق میکنه.

2:08:14

یکی بعدی محصول هست و فرایند و پروژه. پروژه براساس قرارداد‌های معقود شده انجام میدیم و براساس این ارتباط‌ها و شکل

هم پس این ۴ تا بحث تأثیر دارن روی فعالیت‌ها و چالش‌ها انجام این‌ها. حالا تو هر کدوم جداگانه توضیح داده. اجازه افراد

رو خودش مطالعه کنه. ۴ تا ویژگی مدیر پروژه رو آوردن. یک مدیر پروژه خوب کسی که تخصص بالا و مهارت بالا داشته باشه

مثل مثله به آیکو پرداخته، مسئله رو تجزیه کنی به قطعات کوچیک

جلسه پیش درمورد کلیات مدیریت پروژه‌های نرم‌افزاری صحبت کردیم.

سازمان‌دهی تیم‌های نرم‌افزاری: هر جایی که صحبت از سازمان پروژه یا چارت سازمانی پروژه همه به یه مفهوم اشاره دارن

و اینکه منابع انسانی قرار چه ساختاری تقسیم و دسته‌بندی بشن. حالا یا ساختار به سازمان، دپارتمان دولتی و

کلاسی ساختار به پروژه که اونجا یه فرق دارن، ساختار پروژه، تجزیه کارها، مکانیزم‌ها و... است.

از لحاظ تصمیم‌گیری و رهبری پروژه چند تا مدل اراده داره و از لحاظ مدیریت پروژه 8 عنصر تمرکز دموکراتیک و ...

عنصر تمرکز Decentralize (D) و تمرکز Centralize (C) اشاره داره.

دموکراتیک: از نظر چنین مدیریتی که آیا رهبری کار به صورت شورایی یا گروهی و یا یک نفره (دیکتاتوری) است

و تمرکز و عنصر تمرکز زیادی که برای حل مسائل گرفته شده. اگر به صورت گروهی باشد سیستم Decentra

۳. تمرکز کنترل شده: یعنی یه نفر مدیر پروژه داریم (کنترل شده)، تمرکز یعنی تصمیم‌گیری‌ها فردی هست نه گروهی

هم یه نفر مدیر پروژه هست هم تصمیم‌گیری‌ها یه نفر به صورت کلی تصمیم می‌گیره بر اساس تصمیمات کلی که صورت گرفته

۲. یعنی یه نفر مدیر پروژه باشد (کنترل شده) ولی حل مسائل گروهی هست مثلا شکل تصمیم‌گیری نرم‌افزاری پیش میاد

مدیرهای بخش‌ها جلسه تشکیل میدن و به صورت گروهی درمورد پروژه مشورت می‌کنن.

۱. هم رهبری کار غیر دموکراتیک هست هم تصمیم‌گیری‌ها و کلا این روش به صورت گروهی صورت می‌گیره.

عنصر تمرکز دموکراتیک و رهبری کار به صورت گروهی هست هم حل مشکلات تصمیم‌های کاری به صورت گروهی

و مرتبط و مناسب تر صورت می‌گیره. دموکراتیک لایه‌های بالا و عنصر تمرکز و کنترل لایه‌های پایین

پس ۲ تا حالت بوجود میاد. اسم حالت اصلا وجود نداره؟ هرکس دموکراتیک پس بیشتر که توی لایه‌های پایین‌تر مثل

به صورت فیزیکی لایه‌های بالاتر روی‌هم پس میشود ناوگان

سختی مشکلاتی که باید مورد توجه قرار بگیره. توی تست‌های امتحانی هم استفاده میشه. توی این حالت کسم سازمانهای و توجه کنید

چهار الگوی سازمانهایی که وجود داره. الگوی بسته مثل سازمانهایی هستن که از قبل معین شده با اختیارات

مشخص و از پیش تعریف شده هست و خارج از این ارکان پذیرفته نیست. بخش دلیل ارتباط نخواهد داشت

باشد باید با صدور این تخصص و اشراف بیاید. دقیقا این الگوی خوب هست برای یه کارخانه

صنعتی. قطعا باید توی یه ساختار باشه.

الگوی مکانیکی برای کارهایی مثل بخش بتن که نیاز به خلاقیت و معین کردن کارها از قبل داره. بدون اینکه

ببینیم توی چه بخشی کار میکنیم مثل الگوی مکانیکی و ط flat ندارم. و برای کارهایی خوبه که نیاز به ابتکار و خلاقیت

و مشارکت هستن و خارج از محدودیت‌های تعریف شده داشته باشید. الگوی ۳ باز به حالتی بین این دو تا هست

از طرفی ساختار ارتباط با بخش‌های یکه رو نمیده یکی جلبات از قبل تعریف شده داره و مکانیزم ارتباطی رو مشخص

کنه و تصمیم‌گیری‌ها براساس توافق آرا باشد. الگوی ۴: برای کارهایی که به صورت حواری صورت میگیره

مثلا سامانه‌ای دو تولید یکی که رسیده بخش داره و میخواد به شکل حواری پیش بره.

بحث ارتباط باز باشد باعث بحث کار از نظمش خارج شد و بحث کار شادروی بوجود میاد. صورت I جایگاه

این بازبودن و ارتباطات زیاد اندازه‌ی تیم رو نباید تلف کنه. و باعث اتلاف انرژی نشه و مورد ۲ هر تکنیک نباید غالب

شبه توی یه مجموعه حاکم شه. توی هر تکنیک باید پیروی شه و تقویت شه. توی موضوع دقتی بیشتر

و گردش کنیم. ۳- باید این ارتباط را هم بدهیم که لازمه بتوانیم کنیم. توسط ابزارهای مختلفی برای حفظ نیازی و اینا

رخ میدهد. پس اساس می‌کنیم استدلال شکل از روی نیازهایی که رو از دست میدن. مثلا فلسفه تولید به بچه‌های

یتیم اطلاق، بلکه ... ارتباط پیدا می‌کنه. باید هماهنگی داشت. حوزه‌ها یکی تعریف نشدن کارکنان، باید پول

انسانی هم بدی. و نباید بیشتر بیکار باشند و از کار بیاد؟ اینکه کیفیت باید پیگیر بکنه و نقش مفهوم می‌

باید کار شکل و تأیید تکنیکی چه کاری انجام بدهد. متدولوژی‌های مهم مثل RUP و Agile

به اینجا برگشت چون نقش‌ها مشخص شده. ۵ به مورد ۲ توجه نشده.

راه‌های جلوگیری از انتخاب‌های نادرست ... حوزه‌ها اینکه پیچیده به توجه مبتنی اگه به پروژه سخت باشد و نیاز

به مطالعه داشته باشد باید چه کار نیست پیچیده و فرایندی که دارید انعطاف‌پذیر باشد و نیاز به انعطاف

پایان در پروژه وجود نداره و هدف اینکه کار با کیفیت ارائه کنه. ۲- بعضی‌ها ... رشته‌ها ... هست.

ارتباط: رسانه‌ها (رسمی و غیررسمی) — غیررسمی — رسمی — قابل استفاده

Communication plan

رسانه‌های رسمی و غیررسمی: تو پروژه‌ها در ابتدا ایجاد و مشخص می‌کنن توی طرح ارتباطات مشخص می‌کنن هم پیشرفت‌های

رسمی که توی تصمیم‌گیری‌ها همیشه پیشون استفاده کرد مشخص می‌کنن. خروجی‌های پروژه اگه توسط لاین مدیریت

که تهیه کرده این نقشه این نقشه قابل استفاده برای کل پروژه خواهد بود ولی اگه به صورت کلی‌تر باشه دیگه

کنن اونطوری که تأیید شده قابل استفاده.

رسمی و بیان فرزی: مثل صرف جلسه‌های رسمی، یا جلسه‌های بازبینی در تقسیم کیفیت که اینجا پیشرفته شده اینجا

ملاک نیست رسمی و قابل استفاده هست. پیشرفت‌های غیررسمی: غیررسمی هست مثلا تیم تولید به جلسه بیاره تا اشکالات

رو بررسی کند

Year. Month. Date. Subject

اسلاید ۹ رفتیم. روابط الزامی و اختیاری مثل بندگیل زدن: ارتباطات بین افراد در پروژه و طبیعی است

اسلاید

درباره فعال و فرایند صحبت کردیم. حالا جلوتر بیشتر می‌پردازیم. یک شیء باید اطلاعات فعال قبل که مرتبط با مدیریت پروژه

این باید کامل بگیره. این فعال روی فعالیت‌ها تأثیر میذاره.

یکی دیگه از حوزه‌های مهم انجام پروژه، حاصله. ترجمه کلمه Scope هستا. چه چیزهایی داخل پروژه هست چه چیزهایی

نیست. و اندازه رو تعیین می‌کنه. و بافت Context: چون این هم یکی از بافت‌های از بافت‌ها

Constraint

برقرار میکنه و توسط کاربران بهتر استفاده میشه و به سری الزامات میاره (مثلاً این زمینه).

چه شناخت فعال و حاصله، تجزیه و تحلیل بودن مورد دیگه‌ای که تازه فرایند هست. باید مدل فرایند کاری

این رو و مدل‌های انجام کارشو تعیین کنه.

بعد از انتخاب مدل و طرح پروژه شروع میشه به انجام شدن و فعالیت‌ها رو می‌چینیم. و WBS کار

ساختار شکست کار

ارجاع شبیه مدیریت مطالب پیدا میکنیم. زیرشاخه‌های تجزیه هم باید مشخص بشه از نظر عملکرد شکست شبیه ابعادی باید فرایند مشخص باشه

مدیریت پروژه باید مدل‌های انجام کارشو تأیید کنه. باید مدل فرایند کاری این رو مشخص کنه مثلاً در پروژه‌های نرم‌افزاری و یا باید

شبیه نرم‌افزاری این وجود داره، کارفرما باید جواب بده که میشه انجام کارشون چجا هست. مدل RAD

prototype، طراحی و... مدیر پروژه باید این رو مشخص کنه چون فعالیت‌هاش رو میاد و براساس مراحل پیش می‌ره.

بعدی: باید ببینیم مدل زمانبندی مناسبه؟ براساس همون مواردی که در بالا گفتیم آیا زمان کم باشه مثلاً

مدل RAD. جایی که براساس فرایندی که سفارش میدن و کارفرما دوست داشته باشه برای استفاده از زمانبندی

خاص مثل Agile ، خیلی که شغل به کارهست ؛ یه دلیل همه کارهایی که توسعه میشه متدولوژی

خوب انتخاب باشه . ط) بعد انتخاب مثل طرح پروژه ... خب تا اینجا با تعهدات کمدل و بلند

و دورنمایی از خروجی ها و فرض ها معلوم شده . الان باید plan کنیم که چه کارهایی باید صورت

بگیره . و WBS کار دریاد هم دلیل رو انجام بدیم . WBS بشید فهرست کتاب هست بخش

متدولوژی و تعهد موقت مشخص میشه . SRS متدشخصات نیازمندی های نرم افزار مثلا مشخص کنید همه دلیل

باید مشخص باشد هر چقدر سطح بیشتر باشد WBS کار به فعالیت های شکسته شده و ریزتر شده .

تخمین و تحلیل انجام بدیم و فعالیت ها رو به لاک هر تخمین مشخص کنیم . تایمش محدود و قرار نیست نداریم

در کنارهم زیاد تا لیست فرضیه های محدود دریاد . چقدر طول میکشه این فرایند ؟ و کل سامانه

باید با سامانه ای که 90 درصد سیستم مشابه فرق داره . اغلب زمانی که اشتباه به مدیریت رسیدگی

پیگیره : کلاهی که ... در مواقع خطا قرار میگیرن خطرش RISK یا همون فاکتور هست .

کلاهی که همه پروژه حواسش باشد . مورد 1 یعنی نقص ها نیازهای شرکت رو درک نکنن ، چون توقع

خیلی باید نیازمندی هست باشه یا تعدیل خطر یه کمیته Ana که لازم باید اش نیازمندی ها

هست و اگه انجام نشه باشد کل پروژه خراب میشه . و خیلی اون این مورد نیازمندی ها

اگه نیازها رو درک نکنه انجام بدیم و بخش های غلط تو مراحل بعدی پرت اعتبار بخش مشخص شه و

دوباره کارها بوجود میاد و زمان پروژه طولانی و شکست . 2 درون پروژه ضعیف باشد یعنی

Year. Month. Date. Subject

وارد بازار نشده ولی توجیهی از محصول داده نشده یعنی شرح خدمات فنی نداده. نکته این کنسل چی

خواهد باشد و چی داده. هم نیازمندی ها باید مشخص باشد ولی باید در کل تیم داده بشه و الزامات

و محدودیت های اصلی و موارد تأثیرگذار در پلن مشخص بشه.

مورد ۳: تغییرات به خوبی کنترل بشه. یکی از مشکلات های اصلی change manage هست و باید

مدیریت با تعریف وظایف مشخص باشد و در متدولوژی RUP هم باید.

متدولوژی انتخاب شده: باید متدولوژی نرم افزار طوری انتخاب بشه که خیلی قیچی نباشه.

۵: نیازهای تجاری در حال تغییر هستند و اگر نیازهای بیزینسی در حال تغییر متناسب با این باید در

نظر گرفت. (یک روز بیزینس Digital transformation) یا چیزی داریم به اسم (KYC)

یعنی know your cutsh که به احراز هویت دیجیتالی و نرم افزاری اشاره داره یکی از سازوکار های رایج هست

برای سامانه های نیاز به احراز هویت دارند. که میتونه فرایند کاری به سازمان رو تحمیل کنه. مورد بعدی

مهلت ها غیر واقعی هستند... و کاربران متقاعد نشدند: یه کاربران همیشه مقاومت میکنند چون در

مقابل تغییر مقاومت میکنیم. وجود قواعد Role engin باید وجود داشته باشد. و اتوماسیون سازی

Automation و خودکارسازی داریم. حداکثر اتوماسیون سازی باید وجود داشته باشد لازم نیست باید

انجام بشه. مثل بانک که همه رو اتومات کردن. ۸- کمبود تیم تست... و مورد بعد: افراد

متخصص وجود ندارند. منبع انسانی مهم ترین چالش در پروژه های نرم افزاری هست. مورد بعدی: مدیران و کارمندان...

هنداری برای مدیریت هست برای میزان بازبودن ساختار پروژه. باید آیه دری اجازه پیشنهاد داده بشه

Honar

Year. ——— Month. ——— Date. ——— Subject ———

اسلاید بعدی رو نوشتن گزینه ۵ بعدی: WHS یعنی 5 تا W و 4 تا H وجود داره. میدم پروژه باید

کاملاً توجه بکنم. اولی چرا تیم تدوین میشه؟ به اهداف اشاره داره که باید اهداف رو به دونه چیز

اولویت و فعالیت‌ها رو تحت تأثیر قرار میده. مورد بعدی چه چیزی در چه موقع انجام شود؟ باید الزامات

زمانی محکم داشته باشه. مورد بعدی: چه شخصی مسئول چه کاری می‌باشد؟ اشاره به تعیین کردن

وظایف هست اینکه نقش‌ها چطور هست. وظایف و مسئولیت‌ها و ارتباطش با هم

مشخص باشه. از لحاظ سازمانی در کجا قرار داره؟ باید بدونیم چه واحدهایی هست و ...

از لحاظ تکنیکی؟ یعنی تکنولوژی کار. از نظر مبلغ ...؟ باید باید مشخص باشد

۲ تا برنامه‌نویس یا ۲۰ تا مثلاً. اسلاید آخر به شکوایی که لیست مهم‌ترین کارهایی

که تو مدیریت پروژه هم انجام میشه آورده. مهم‌ترین استانداردی که در حوزه مدیریت

پروژه وجود داره PMBOK هست. Program manag. body of Knowl یعنی

دانش مدیریت برنامه و این کتاب‌هاش هست و مرکز میده.

توی این شکوای این مدارک رو گفته ایده، خط (نقشه) و باید اینا انجام بشه و تخمین

داشته باشه بعدی زمان ایجاد گفته برخی دارن باهم و ... هست. و روش کار شدن.

مدیریت پروژه مبتنی بر فعالیت‌ها و متریک‌هاست بله. و به هر حال خوبه هم داره توی کتاب. یعنی چه موارد کلی

باید توی یه پروژه مورد بررسی انجام داده یا مثلاً این مؤلفه‌ها، شاخص‌ها و متریک‌ها رو بررسی کرده. رد لیدی هاردیه

Year: Month. Date. Subject

هیچ یک مطلب، البته باید در مورد کیفیت دقت باشیم. به آنجایی از بزنه زیر کالا فرادیمی عملکرد نیروی انسانی

که باید در نظر گرفته شه

Year. Month. Date. Subject

فصل 3 و 4 تموم کردیم فصل 5 حالا. این فصل به یکی از فعالیت های اصلی در مدیریت پروژه به نام طراحی و برنامه ریزی پروژه نرم افزار

می پردازه. تو کنکور هم ازش سوال میاد. دغدغش به بخشی به نام اندازه، نرم افزار، سایز نرم افزار هست که خیلی

برای planning و بقیه کارها مهم است. که به اندازه نرم افزار مرتبط فاهد بود خیلی از این ها.

فعالیت هایی مثل طراحی برنامه پروژه روی چه مواردی هست. پایه شروع و پایان فعالیت ها مشخص بشه

پس متدهای تخمین هست. estimation ما باید برآورد کنیم software stim proj

ما روی زمان، هزینه و روی منابع برآورد می کنیم. منابع میتونه سخت افزار باشه یا نیروی انسانی باشه

چقدر نفرات مورد نیاز یعنی چند نفر به چه درجه هایی مورد نیاز هست. و با نفر ماه اندازه گیری میشه یعنی

بگیم 72 نفر ماه یعنی اگه 1 نفر بخواد روی این پروژه کار بکنه 72 نفر ماه نیاز داره. و اگه به پروژه

48 نفر ماه نیاز داشته باشه، 1 نفر بخواد روی این پروژه کار بکنه چند ماه طول میکشه؟ 48 ماه

اگه دو نفر داشته باشیم هر ماه دو نفر کار بکنن 24 ماه طول میکشه حالا اگه 48 ماه نفر و 4 نفر داشته

باشیم این پروژه چند سال طول میکشه؟ 1 سال. 72 ماه. اگه 6 نفر داشته باشیم، 6 ماه طول

میکشه پس توی تخمین ها دو مورد که تخمین می زنیم مهم ترین هاش تایم، تلاش و هزینه

هست. مثلا اگه یک پروژه ای 48 نفر ماه، اگه هر نفر حقوق 5 میلیون بگیره از نظر نیروی

انسانی میشه 240 از کلا منابع انسانی میشه. مهم ترین بخش یه پروژه نرم افزاری منابع انسانی اش

هست. نیروی انسانی تشکیل میده، مالیات و موارد دیگه براش رد میشه. و هزینه سخت افزار

نرم افزار و ... که این موارد اینجا بحث نمی‌کنیم. وقتی بخواهیم یه جایی مثلاً پیشنهادی انجام بدیم بعد از

بدست آوردن مبلغ نیروی انسانی این رو در 2 یا 3 میکنیم و به کسانی پیشنهاد ارائه میدیم

چون پیشنهاد کار هست و به ترتیب سودی هم در نظر گرفته میشه. ما غیر از management پروژه‌های نرم افزاری

ما فعالیت‌های دیگه‌ای داریم مثل WBS، مدیریت ریسک‌ها، زمانبندی، Monitoring پروژه

و فعالیت‌های تضمین کیفیت. ما دایکومنت کردن پروژه یه مجموعه این فعالیت‌ها می‌گیم مدیریت

پیکربندی. اشتراکی از فعالیت‌ها رو توی مدیریت پروژه داریم. چطوری این کارو انجام میدیم؟ برآورد

برای مدیریت پروژه هست. و همیشه برآورد این موارد نیاز به این دارن که یه برآورد هم اندازه نرم افزار داشته

باشیم. برای همین گفتم مدیران نرم افزاری و خیلی مهمه که داده‌های تاریخی باشه یعنی بدونیم

پروژه‌های نرم افزاری قبلی چطور بودن. مثلاً ماه رو بگیریم 20 روز کاری و هر روز 50

خط کد میشه 1000 خط کد در ماه نیروی من می‌تونه بنویسه. اگه یه تخمینی از تعداد

خطوط کد نرم افزاری داشته باشم مثلاً 15000 خط کد دارم و این داده‌های تاریخی‌ام از پروژه‌های

قبلی‌ام و هر نفر 1000 خط بنویسه 15 ماه طول میکشه این پروژه. اصل کلی این فصل همین

هست تعداد خطوط کد یکی از مهم‌ترین توان بحث LOC. در اینجا 1000 یعنی

داده‌های تاریخی من بود. دلیل اهمیت تخمین: براساس این دما یزان زمان نوشتم

رو حساب می‌کنم که براساس این برنامه‌ریزی پروژه بدست می‌آد. در اصل تخمین اصل سایه نرم افزار باید بدست

Year. Month. Date. Subject

باید برای این علم دانش نرم افزار پایه حساب بشه و مشخص بشه scope ، چه چیزی داخل نرم افزار هست چه چیزی نیست بعد نرم افزار به بخش های کوچک شکسته بشه تا بتونیم تخمین بزنیم و میزان تغییرهایی که مورد نیاز داشت بدست بیاریم ، معمولا خیلی چیزها پایه هزینه، تلاش در زمان رو بدید میاره.

همیشه آورد خوب و تخمین خوب یکی از رسک های پروژه هست ، و اگه خوب تخمین نزنی هزینه رو ... دچار اشتباه میشه . پیچیدگی پروژه : مثلا توی دوتا نرم افزار با LOC حسابی داشتی می‌تونی نرم افزار دوم 700 خط که برای یه الگوریتم پیچیده هوش مصنوعی هست ولی اون یکی ساده و برای نوشتن اش 3 و 4 نفر تکمیل اش کردن ، اینکه پیچیدگی پروژه رو چطوری محاسبه کنیم مشکل هست و مشکله ، مورد بعدی اندازه بزرگی پروژه : اول داشتن خوب تشخیص داده بشه و بعضی وقت ها اینطوری نیست و هزینه زیادی بوجود میاد ، کلا استفاده به نرم افزار باعث تولید ده ها برابر یا ده برابر هزینه زیادی میشه یا بیشتر از اسکوپ و تشخیص بعد چطور تبدیل اش کنیم به خطوط که نیاز به تجربه و هیتوریکال دیتا داره ، روش های فنی مثل ... و دوتا مرحله که بیان شد در اکثر موارد پیدا می‌شن چون هر کسی تست کار رو از یه منظر خاص بدست میاره.

مورد سوم : میزان عدم قطعیت ... پایه چون نرم افزار موجودیت منطقی هست با اینکه طراحی و معماری ها و گزینه های مختلفی میتونه در نظر گرفته شده با سایر ملاک نرم افزار فرق کنه.

و مورد 4 دسترسی بدون ... مثلا هم نرم افزار کارکرد نوشته باید لازم فرق داشته و حتما نمی‌خواستند توی پروژه چه

چقدر کد بزنند. حالا یک دلیلی گفتند یک آخر نباید دیگر پروژه و دیوار ... نظرشان این نیست که بازاری

به تحقیق بدون نداره یعنی این اینه که اگه بخوایی دیوار به خرج بدی انقد پارامترهای بازاری رو

دربیاره نرم افزار هست که تحت نظر شد هم وقت و کارتون خیلی طول میکشه.

اسلاید بعدی: اصلا اینطور چیزی رو دیگه نگفتم. یعنی برای تعیین خوب به دلیل نرم افزار نیاز داریم

و از یه سری پارامترها مختلف شبیه یک اینکه ذی‌نفعان این پروژه کیا هستن و به صورت مستقیم

یا غیر مستقیم نفع می‌برند و ورودی‌ها و خروجی‌ها کیا هستن. کارایی و عملکرد هم نرم افزار خیلی مهم هست

مثلا اینکه نرم افزار در همه زمان دسترسی باشد یا نباشد فرق داره.

و کدوم‌هایی که بلدیم مثلا ریتاش بیاطلاعات و باید توسط تابع خاصی مدیریت شه.

رابط‌ها یا واسط‌ها (interfaces) باید سرویس‌هایی و چه سازمان‌هایی ارتباط برقرار کنه

چه اطلاعات جدیدی بیاد 4G یا 5G فرق داره. این دیدگاه‌ها نرم افزارهای

چه کارهایی انجام بدن و همه به دلیل نرم افزار ربط داره.

بدست آوردن ... : از اینا می‌تونیم کسب اطلاعات کنیم فرد شرکت و صاحبان سیستم و طلبه‌ها

باشرکت، پروژه‌های مشابه و استفاده از تجربه افراد.

اسلاید بعدی: ۲ و ۳ مورد علامت دهنده برای ارتباط با شرکت، امکان نیستی: که ترجمه کلمه

Feasibility هست. بررسی کنیم که امکان انجام دادن پروژه یا نه. شما رفتن به قرارداد پیش باید

قبل این پیشنهاد و لطفاش باید انجام بدین که براساس دانسته ای که تخمین دادیم به یه سری از زمان

بودجه ای، زمانی و ... دادیم. آیا امکانش رو دارم که انجام بدم. که آورده فرآورده و هزینه و ...

اینجا به شما ترتیب کردن خط مدل و فعل گفته (نیاز به حفظ برای امتحان نداره) اینجا دیگر نیست توضیح

دادن، تجزیه کردن و عملکرد اشون گفته فقط تقسیم بندی هاش برای ما مهمه. چه اطلاعاتی در خصوص

کارایی دادن تغییر مدیریت از 5 به 10 یا فاصله نداشتن جمعیت ها و ارتباطات و ...

اینکه چه اطلاعاتی بدیم تا اسکوپ به سیستم مشخص بشه. ایا به جدی ؟ منابع رو گفته

یکی از موارد مهم در تخمین، اسکوپ بود که یه سری اطلاعات در خصوص تخمین به ما میده مواد

مهم علاوه بر اسکوپ باید منابع هم مشخص بشه. یعنی منابع هم باید تخمین بکنیم.

اینکه منابع چی ها هستن باشن. گفته: سه تا منبع اصلی میتونم شناسایی بکنم

یک محیط توسعه که خوبه کامل ترین کنم فقط محیط توسعه ندارم انواع محیط توسعه دادم. مثلا

محیط تست، محیط برنامه نویس ها، محیط تست پیشرفته شده باشه یا محیط

تحلیلی باشه.

به محیط بیرونی هم داریم برای تست و انتقال به محیط عملیاتی.

ح) مقایسه کامپیوتری و هزینه یا نیازمندی به پارک کارکنان هزینه کامپیوتر اینو می‌خریم به جای وقت زیادی

و حدود بعدی مهمترین هست پیاده‌سازی اینجا اولیه یعنی کم‌کیفیت و ایجاد کارایی یعنی پایه‌ای.

و منظورش نیروی کار داخل پروژه هست و به شکل به هم، چون محیط هست که روش تمام انجام

هست بسته. حال اگه هزینه باید تعیین باشه مثلا هیچ چیز هست تعداد پروژه مثلا 4 فرد چی؟

یا 4 محیط توسعه مالیات ثبت باید توصیف اش کنم نیازمند چی داره این ابتکارات

و دوره حدود هم وضعیت دسترسی available بودنش چقدره 24 ساعت نیاز داره؟

حدود بعدی زمانی: مثلا از دسترسی و این 4 فرد تولید زمان تقریبی حدود نیاز داشتن

مدت زمانی که منبع به کار گرفته شده برای نیازمندی همه سالات در دسترس باشد.

و تولید زمانی که در دسترس هست زمانی که به کار گرفته شده چقدر هست.

و حدود بیانیه اساسی کلیه: براساس WBS درست میاریم و همه افراد چه پروژه و باید دقیقا

تعیین بزنیم. مثلا کسی که نرم افزارهای هست یا بتونه برنامه‌نویسی اش رو تمام باید

تعیین بکنیم. حدود بعدی با قابلیت گیرد (ورطهم) هست مثلا نرم افزارهای آماده

یا نیمه آماده اینا لحظه از سرش خرید کردن. مثلا بخشی ساخته شده حاضر و آماده هست

بخشی که داریم ... مثلا بیاد ایجاد کردیم با تغییرات استفاده اش می‌کنیم با تغییرات بیان

Year. Month. Date. Subject

با تغییرات زیاد، یا خودتون ایجاد کنید حتما خواهید بخرید. پارامترهای مهم در نرم‌افزار

همانند قیمت. و منبع نرم‌افزارهای مبتنی که خارج از نرم‌افزارهای که تولید می‌کنیم نیست

و نیز ساخت ما هم نرم‌افزار خواهد اما با اسم محیط توسعه دارند و کلا منبع نرم‌افزارهای متعددی در ساخت

پارامترهای اولیه. منبع محیطی رو توضیح بدهیم. برآورد پروژه‌های نرم‌افزاری: اومدن گفتن چند تا

گزینه وجود داره چی رو می‌خوایم برآورد کنیم: Time, effort, نیروی کار

که باید کل اندازه نرم‌افزار تخمین زده بشه. اولی میگه به حقیقت می‌خوایم دقیق‌تر بشه دل

به دل نخورد. ۲- این دو تا مبنا براساس پروژه‌های قبلی نیست یعنی قیمت گرفت باید

برای نرم‌افزار جاری بهتر گرفت. براساس پارامترهایی که براساس تجارت تاریخی پروژه‌های قبلی داریم

این ۴ تا چیز: پارامتر اندازه جاری نرم‌افزار و پارامتر ظرفیت‌های تاریخی

روش 3 و 3: از فنون تفکیک و ... تجزیه می‌کنیم تا مسئله آسون‌تر بشه و بعد جمع می‌کنیم و دومین تقسیم

دیگه هم توی برآورد از مدل‌های تجربی یعنی یه سری پارامتر از نرم‌افزار حاضر و یه سری تجربه از

پروژه‌های قبلی رو استاندارد با هم ترکیب کنند تا میزان تلاش و هزینه رو بدست بیاریم و براساس

$d = f(v)$ هست حالا این v ها میتونند پارامتر قبلی یا نرم‌افزار حاضر باشد و این d

پارامتر تعیین یا هزینه (تلاش) یعنی چقدر زمان ثابته مثلا یه گروه داریم چقدر از نفرات

میبرند میاد. برای اندازه نرم‌افزار که یکی از این پارامترهای موثر اصلی هست دو تا

Month. Date. Subject

گزینه اولی داریم یکی LOC هست مثلا 101000 خط کد یا function هست

یعنی امتیاز کارکردی. یکی از همین دوتا. مثلا 101000 هست یا 1000 تا یا قابلیت

پوینت هست مثلا امکان پرداختی یا هر چیزی که کدتان از پردازش داره.

و function point خیلی کم هست از LOC. و گفتم ابزار خودکار هم داریم. برای ابزار برآورد پروژه

هست software project estimation tools.

اینجا اومده درباره فنون تجزیه صحبت کرده.

4 مورد برای صحبت اش بیان کرده. 1) یعنی خروجی که ارائه می‌کنیم تا تخمینی برآورد رو به شکل

مناسب انجام بده. مثلا بیاید براساس 10 تا پروژه یا 200 تا پروژه فرق میکنه.

2) توانایی ترجمه! یعنی ورودی که ارائه می‌کنیم بتوانند پارامترهای درونظم تخمین بزنند.

3) توی این فرمول! یه سری پارامترها هست که دقت‌های تیم کاملا بشه چون روی زمان سوخت‌ها

تاثیر داره. مثلا 4 تا دانشی حرفه‌ای یا 3 تا حرفه‌ای و یه سال خیلی فرق میکنن.

4) این هم مهم. با موضوع جدول نیازمندی های پروژه داریم این برآورد رو انجام میدیم. و

اثبات ما هم باید تکرارپذیر و دقت‌های جدید بدست بیاد. هم ترتیب چیز ورودی پارامترها

هم ابزار هست که شامل بودجه، تیم و ... نیاز داریم و وابسته به تخمین دقت

پروژه ها. چهار روش مختلف برای مدیریت مطرح میکنه: یا اندازه هم ابزار یا

Honar

منطق فازی یعنی به صورت قاطع نباشد و براساس یه توزیع آماری باشد. مثلا احتمال 30٪ اینقدر COC

و 50٪ اینقدر ... یعنی به صورت فازی (0 و 1 نباشد) ما توی این مدل از فازی استفاده

نمی‌کنیم دقیق در بیاریم. مورد دوم 12 امتیازات عملکردی function point، مدرنی برای نرم افزار

که به پرازش بیشتر این نشده و برای پروژه‌های مدرن مورد استفاده قرار میگیرد. گزارش هم میتونن بلکه بیشتر

به معیار Fump که هست ایم. یا تکنیکات، یا الگوریتم‌ها مثلا. یعنی هرچی که باید پیش‌بینی تعداد کد نوشته شده

تا بجا باشه. ۳) اینجا COC رو میگن ۴) یعنی به نرم افزار دلیم وقتی تخمین زدیم 90٪

این باید دقت بیشتر برای نرم افزار بدیم.

استاندارد بعدی: دو تا رویکرد اصلی برای تخمین اندازه نرم افزار داریم: یکی مبتنی بر مسئله problem based و

مبتنی بر فرایند process based و تاریخ داده. مبتنی بر مسئله یعنی زوم می‌کنیم

روی مسئله چقدر ابعاد داره چقدر کار نیاز داره و مثلا 10,000 کد داره و نیاز به اینقدر نفر-ماه

داره یعنی از تجزیه مسئله ساخته میشه. این فعل کافی من چی داره برای ایجاد داده‌ها

چقدر نیاز داریم این مسئله خیلی فرایند یعنی چه کارهایی باید انجام بدیم که بتونیم فرایندی

خطی و ... پیاده کنیم و باید بگیم چه نفر-ماه نیاز داریم. پس از روی فعل به اینکه

برسیم به نفرماه و از روی فرایند نرم افزار به نفرماه برسیم. بعضی وقت‌ها ترکیبی

و هر دو رو استفاده میشه. تقریبی بخش اول در مورد مبتنی بر مسئله صحبت می‌کنیم.

اینجا تو مرحله اول به این صورت هست که اندازه SDK براساس LOC یا امتیازات کاربردی الگوریتمی و عملکردی

انجام میشه. اول براساس یکی از این دو پارامتر بدست میاریم بعد هم براساس داده های تاریخی

قبلی که مشخص نموده سازمان قبلی نرم افزار هست مثلا هر یک نفر چقدر میتونند کد بزنند.

برفرض مثال میدونند تحلیل کردیم براساس یه روش بدست میاریم که این پروژه انقدر خط

کد هست. این همون متریک بهره وری هست یعنی LOC باشد بهره وری بیشتر

و کمتر باشد کمتر پس عدد LOC یعنی متریک بهره وری مثلا تیم تولید یا افراد پروژه ای

که معمولا به صورت Funct po یا LOC pm هست یعنی تعداد خط کد برازای تعداد نفرات

و Funpo به ازای تعداد نفرات. بعد متریک های بهره برداری رو محاسبه کنیم. مثلا:

میاییم پروژه قبلی 20K LOC خط کد بوده. 10 نفر بوده و 5 نفرماه هم کار کردیم

و 5 نفر مثلا 5 ماه کار کردیم تا تمام شد 25 نفرماه یعنی و میشه 20K/25 LOC یعنی تقسیم کنم

که همون متریک بهره وری هست یعنی براساس LOC pm گفتم. اگر همون تیم پروژه

جدید انجام بدن عدد LOC بیشتر میشه یا تازه کارمون کمتر بهتر هست. و خیلی

مهم داده های تاریخی رو براساس این در نظر بگیریم.

اینجا یک نکته دیگر روش Problem قبلی یک مسئله برای تخمین چیه. اول اندازه نرم افزار

بعد با فرض که از داده های تاریخی استفاده میکنیم تا نفرماه به دست بیاریم

نکته‌ای که هست اینه که می‌تونیم حالت بدبینانه / متوسط و خوش‌بینانه رو در نظر بگیریم.

یعنی توی اندازه‌گیری‌ها نرم‌افزار اینا رو براساس پارامترهای انجام شده تجزیه می‌کنه. در حقیقت به یکی دیگه که حالت

داره و حتی از پارامترهای داده تاریخی کمک می‌گیره که هر هفته چقدر کد زده در ماه این حالت تعداد داده‌ها

و عددهای متفاوتی ایجاد میکنه. مثلاً تعداد نفرماه شده 80 نفر ماه، 50 نفر ماه توی حالت‌های مختلف

و عدد مختلفی رو میده. و یکی بدحالت بدبینانه ایستاده / خوش‌بینانه هم 1 واحد و یکی

حالت عادی مثل 4 واحدی باشه. اگه op باشه 50 و مثلاً 60 و بدبینانه

80 باشه عدد مختلفی میشه: یعنی 50 نفرماه / 60 نفرماه 80 نفرماه

$$S = \left(\underbrace{S_{opt}}_{50} + 4\underbrace{S_{m}}_{60} + \underbrace{S_{pess}}_{80}\right)/6 \longrightarrow 61.6$$

Year. Month. Date. Subject

به جای به نمودار پروژه رسیدیم که ادامه می‌دهیم. در مورد این صحبت کردیم که اساس اش چی هست پایه پارامترهای تاثیر گذار روی برنامه‌ریزی زمان، هزینه، تلاش نفرماه رو بدست بیاریم و بالاخره هزینه پروژه رو بدست بیاریم. نیاز به این داره که اندازه نرم افزار رو بدست بیاریم. اندازه نرم افزار رو با استفاده از پیاده‌های حل IOC یا امتیاز کارکردی یا امتیاز عملکردی Func po می‌سنجیم. IOC خطوط برنامه و FP تعداد F های نرم افزار هست. IOC معمولی بزرگتری از FP میشه. اگه IOC و پارامترهای بهره‌وری رو داشته باشیم تقسیم کنیم تعداد نفرماه های پروژه بدست میاد. نرخ منفی باشه از روش تایم و تایم مثبت باشه تعداد نفرات. مثلا: خط IOC = 24000 و پارامتر بهره‌وری: 500 (PROD)

و هر نفر توی یک ماه میتونه 500 خط بزنه یعنی اش اینه. 24000/500 = 48 pm میشه.

این نیاز تلاش ما هست effort وقتی نیاز تلاش بدست میاد تایم یا ریسورس بدست میاد مثلا پروژه یه ساله یعنی تایم ما چون اینجا نفرماه هست اینجا هم به ماه میگیم 12 ماه هست و effort/time که شد 48/12 که شد 4 person مقدار منابع ما هست. حالا اگه تعداد نفرات ثابت باشه time بدست میاد مثلا تعداد منابع 6 pers باشد 48/6 = 8 پس 8 month طول میکشه تمام

$E = T \times R$

هزینه زمان نفرات

و فرمول های دیگه‌ای هم داریم مثلا برای IOC که وقتی تقسیم بر میزان بهره‌برداری کنیم E بدست میاد برای هر پارامتری که استفاده میشه گفتیم E هست $E = IOC / PROD$ (بهره‌برداری) اینجا E بر اساس IOC و پارامتر بهره‌برداری بدست آوردیم. حتی میشه یه سری موارد تخمین زده شه بر اساس اون یه سری چیزهای دیگه تخمین زده شه

Year. Month. Date. Subject

توی یک تخمین حالت بدبینانه ۱ محتمل و خوشبینانه رو داریم. و با فرض یه حالت از این‌ها رو بدست میاریم.

(مربوط به فصل 5 بود) همونطور که جلسات پیش گفتیم تخمین‌های شما میتونه دقیق باشه اولش دقیق

برسه یعنی اندازه نرم‌افزار و از روی این بدیم بیشتر چیزی رو بدست بیاریم نفرماه و زمان و هزینه

لازمه برای 10C هست که به نرم‌افزار بدیم.

اینجا تو صفحه 67 مثالی که زده رو حل می‌کنیم. رفته نرم‌افزار رو تجزیه کرده این تیکه‌ها رو من فلسفه کلیس

هندسی و ... برای تخمین زدن این‌ها رو داریم. یعنی باید بیاید این کارها رو انجام بدیا اینا رو داریم.

همه رو جمع زده و بعد آخر حساب کرده 33200 10C شده. با استفاده از فرمول S میانگین گیری کرده برای

بدست آوردن کدهای طول هر قسمت و بعد اینا رو جمع زده و 10C نرم‌افزار بدست آورده. نوشته طبق

سابقه سازمان از روی داده‌های قدیمی اینا رو بدست آورده و پارامتر بهره‌بری این 620

10C به ازای هر نفرماه هست و این پروژه چقدر طول میکشه؟ PROD : 620 10C/pm

33200/620 = 54 و گردش کرده 53-54 رو براساس رهکرد قبلی برسانه بود.

حالا من براساس فانکشن پوینت رفتم جلو اول گفته به بخوای نرم‌افزاری داریم که براشون کد می‌نویسم

مناسب‌تره اطلاعاتی که اینا به سری دادن طرف فرم‌های اطلاعاتی که باید کد نوشته بشه. دیدیم

خروجی‌هایی که گزارش‌ها فایل ایجاد به فایل منابع به سیستم خارجی و تعریف به فایل بیرونی. پرس‌وجوها

قسمت جستجو داریم و ترجمه کلید query هست. هر جا نیاز به نوشتن کدی داریم به عنوان item‌های برای

Year. Month. Date. Subject

محاسبه FP می‌دونیم به دست میاد. تعداد پروژه‌ها (Profile) مثلا به ازای هر پروفایل چند به

عنوان ورودی یا خروجی یا تعامل استعلامات هر کدوم یه پروفایل هست. دیتای فرم ارتباط می‌گیره سرویس

می‌ده سرویس می‌گیره تعداد رابط‌های خارجی هست. یا به اداره بیمه فایلشو ارسال میکنه و ترجمه کلمه

external interface هست در قالب وب سرویس ما هست. برای هر کدوم از این‌ها حالت خوش‌بینانه

محتمل و بدبینانه در نظر گرفته. مثلا گفته خوش‌بینانه 90 تا فرم داریم و تو حالت محتمل 74 و در بدترین

حالت 72 و پروژه‌ها به محتمل ترینشون چون ترینش 4 هست یا تعداد خروجی‌ها 76 هم با این روش.

و به اینا وزن داده مثلا ایجاد یه پروفایل یا ایجاد یه گزارش یا وب سرویس فرق داره. این تعداد هر آیتم

برای محاسبه FP می‌باشد × وزنش امتیاز عملیاتی (کارکردی) به دست میاده سوال چرا اینجا شده 15

دلیل؟ یا اینجا شده 78؟ نه چون اشاره شده مثلا وزنش اون دقیق رو ننوشته و جزء اعداد اشاره

شده 75 و برا اینکه ظاهری میاریم که رند بیاد. و همین 320 FP به دست میاد

حالا باید بگیم هر نفر تو یه ماه چقدر تا FP میتونه تولید کند؟ اینجا گفته 6.5 آدم بلدم 6 و هر فرم امتیاز

4 بود پس حالا که تو هر ماه 6.5 تا FP داریم یعنی 7.5 فرم قوی یه ماه میتونه تولید کنه. یا

هر وب سرویس وزنش 7 هست پس 1 × 1 وب سرویس میشه یعنی یه وب سرویس هر نفر

قوی یه ماه میتونه. prod براساس FP هست. حال FP / به prod تا نفرماه به دست بیاد

و 320 / به 6.5 میشه 58 نفرماه. دلیل اینکه بالا 57 شد این 58 همین تخمین هست که مبنی بر هزینه‌ها

پس تخمین می‌زنیم و فیلاس نیست. برای به دست آوردن تخمین‌های ممکن یا نتیجه‌ترکیبی از این‌ها

نکته‌ای که قرار بود بگیم: وقتی FP کلی برنامه زیاد می‌شه 370 هست. معمولاً تعداد کل FP چندین

برابر می‌شه اینکه چطور برنامه زیاد رو نمیدونم ولی بعدش می‌گیم ضریب تخمین یا ضریب تعدیل بهش

تخمین‌های از هم تفاوت هست که بعضی توی تخمین‌های پیچیده input و output رو نمی‌ذاشتند

یا نیاز داده در جریان، ضرب بشه تا بتونن موارد تفاوت پوشش داده بشه توی امتحان بگیم قیمتم در نظر

نگیریم فقط برنامه‌نویسی یا ضریب تعدیل FP یا تخمین FP رو بدیم چیزی اینجا ضرب در ضریب

1.17 شده این شده 1.17 برابر افزایش کرده. پس بعد از اینکه FP تعداد کلش به دست اومد لازمه ضرب در

ضریب بدیم تا کارکردی کنیم و این تعداد کل x اون ترتیب شبیه 1.17 شبیه از این کردیم تا این

کردیم این شده و حساب هم این بوده و حالا بعد 375 ر برابر 0.5 شبیه 58

توی روش اول براساس میزان تخمین خود نرم‌افزار حرکت کردیم چه تخمین‌های داده‌ها شبیه 150 داده و براساس

اون نقطه به دست آوردیم ولی توی روش دوم خواسته دقیقی برخواسته براساس کارهایی که قرار اجام بدیم

میزان نقطه رو محاسبه می‌کنیم. باید لیست کنیم چه کارایی توی پروژه قرار اجام بشه و مدل ذاتی هم

اینجا مطرح میشه مثلاً مدل حلزونی پیچیده که در کدوم مراحلی داده و براساس اون تعداد لازمی و مدل

و هر کارتونی هر مرحله نوشته شه مثلاً طراحی، اگه تیم بزرگ باشه تکنیک شبیه به زیرسیستم‌ها و بعد طراحی دیتابیس

ها، الگوریتم‌ها و محاسبات. برای هر فعالیت براساس دیتای تاریخی باید چقدر نقطه بیاره تا بهش تخصیص

Year. Month. Date. Subject

این مسئله بستگی برخواسته. توی مسئله قبلی براساس EP یا IC بود اما اینجا براساس فعالیت‌هایی که لازم

واسه انجام کار مشخص میکنه. پس همه پروژه‌های سناریوی خطی عددش یکه؟ نه چون هر سیستم، زیرسیستم‌هایی

درست به جش هست زیرسیستم‌هایی اندازه‌شون یکی نیست رفتن به پروژه مستقلی گذاشته بود اینجا.

مثال زده گفته که: معمولاً هم توی پروژه‌ها برای تخمین سریع این روش خوبه. Activ و نوشته

براساس سناریوی. که اینجا خروجی یا پیچیدگی هست. CC یعنی چی؟ Customer Connection

دقت بدی تجربه خود نرم‌افزار هست آیا الان با دیزاین کل نرم‌افزار چقدر طول میکشه. و مثلا اینجا گفته 0/5

نفرماه و نصف ماه طول میکشه یا مثلا 2.5 نفرماه دو نفر باشه 1 ماه و 0/25 ماه اگه یه نفر باشه 2.5

ماه طول میکشه. N/A یعنی نیاز نبود، چرا توی ریسک و آیا الان شاخص زده؟ چون فعالیت‌های

مدیریتی هستن که روی کل نرم‌افزار صورت میگیره به شکل بخش بخش جواب نمیده.

25 نفرماه ارتباط با مشتری، 25 نفرماه برنامه‌ریزی و ... تا آخر در مجموع 96 نفرماه نیاز داره

قسمت بعد به سری مدل‌های تجربی رو آوردن به سازمانی نه خرابی پروژه‌هایی رو بررسی کرده و پارامترهایی

موثر آوردن و تبدیل به فرمول کرده. E نفرماه A و B به ضریب هست و X ضریب هست

و EV چون اندازه نرم‌افزار هست به متغیر C که عدد ثابته. EV یا IC یا FP هست

پایین هم دوباره اونو نوشته. حساب کردن مثال اون هم پایین:

به این حساب کنم مثل بالا: IC = 33200 و K لاین اف‌سی یعنی ۱ به ۱۰۰۰ بیشه و حساب کرد

یعنی $10C = 33.2K$ اول $(33.2)^{0.96}$ و $0.73 \times$ و به علاوه 5.5 میشه 98

الان برای یه سازمان با چهار روش (تعیین بر مبنا) $10C = 58$ $FP = 59$ و برای حتی برنامه

96 و مدل تجربی 98 در آمده. مدل های تجربی یکه هم همینه بودم را ... اینا مدل های هستن که بر اساس 10C

هستن مدل های FP هم داریم. مثلاً مدل کدنه هست خودش پایین با مدل های یکه. چقدر این فرمول ها اعتبار

دارن؟ چون متفاوت هستن هر کسی اومده برای خودش بر اساس پروژه ها بدست آورده و در ازای اون فا

ربط دهاد و نوشته. مثلاً برای اون بخام بگم که FP بود 375 تقریبی حالا اینجا پایداری می کنم به دست میاد

دفعه بعد 375 رو در خودش ۳ بار ضرب کنم.

مدل کدکوچه: مدلی که خیلی پیچیده مدل problem base هست نیست 10C و FP هست. از اون یکه مدل هزینه

سازی گرفته شده. این مدل ساده راحت بود برای زمان های مختلف پروژه مدلش متفاوت بود.

تفاوت هم زمانی زمانی از اونکه بر اساس 10C و FP بود یه چیزی به اسم OP هم اضافه شده و فرمولش

هست برخلاف 10C با استفاده از شمارش اون ۳۰ تا بدست میاد

NOP = new obj point

یعنی هر چقدر دهدرات تعداد بیشتر باشه این ایتم کمتر شده دالانه پایدار یعنی بدیم

دقیقه اش مثل قبل هست NOP تقسیم بر بازدهی تا Estimant effort بدست میاد.

حدس کنید 70 تا اکثر شرط ما زمان میانگین و 70 تا رپورت ساده و 3 تا هم رپورت مشکل

با 4 تا هم کامپوننت داشته باشیم که کامپوننت با کلاس حساب میشن و Prodn هم متوسط باشه.

Year. Month. Date. Subject

در واقع یعنی 30٪ احتمال دیدیم simph باشد. 70٪ dif باشد و درآوردیم برای حالت 0.3 —

و برای dif هم انقدر ۵۵۵ اعشار هزینه مورد انتظار توی حالت build رو حساب میکنه آورده

و جمع این باید شه 700 .

تغییرات ساده و پیچیده ، تغییرات کم و زیاد

Expeted tim (Build) = 0.3 × 380K + 0.7 × 450K = 424K

و گزینه سوم 267K از همه بهتر است

حاسبه بعد در صفحه بعد نوشته شده است

دو تا فایل داریم یه فایل یعنی نصف های پرس من که 7 تا فصل دادم و 4 تا رو تعلق کردم یعنی 5,2,3,1

و فایل زبان uml

زبان برنامه‌سازی پایتون (۲) (مقدمه)

توی چند جلسه به صورت خیلی کلی می‌گم این زبان چه و اول در مورد شیءگرایی صحبت می‌کنیم یعنی چون زبان VHL شیءگرایی

دارد. توضیحات بعدی مدل و مرور روی زبان VHL به صورت کلی و دیاگرام‌ها چند تا از پروژه‌ها رو بررسی می‌کنیم

(object oriented paradigm)

برخلاف paradigm که شما با دستورات کلاسیک برنامه‌نویسی پایتون پارادایم‌های دیگر رو می‌بینید و واحد اصلی که توی برنامه‌نویسی این

زبان C توی یه تابع main داشتیم دقیقاً بعد تابع دیگه صدا زدیم یه همین‌جا تابع بود

زبان فقط و فقط تابع است. ولی توی شیءگرا با موجودیتی به نام کلاس سر کار داریم. و داخل همه یه کلاس که

هدفی که می‌خوایم — جداگانه متغیرهایی درباره موجودیت و توابع برای کارهای روی خود موجودیت روی کار آیه

داریم متغیر و کیفیت هست. برخلاف زبان‌های رویه‌ای که تابع واحد اصلی هست در تابع ما یه سری

ورودی که داخل تابع تعریف می‌شه داریم و یه سری دستورات که کاری که تابع انجام میده تعریف می‌شه و خروجی

داخل تابع و... مثلاً توی C یه return داشتیم که انتهای تابع بود نتیجه رو برمی‌گردونه. ما هر چیزی که از دنیا که داریم

داریم رو نگه‌داری کنه. یه سری موجودیت (ویژگی) که پیاده کلاسی که در حقیقت این موجودیت هست رو انجام بدیم

(تابع) توی شیءگرایی انگیزه ما به دنیای واقع نزدیک‌تر می‌کنم من یه دانشجو می‌بینم که یه سری مشخصات

این یه موجودیت واحد هست

داره و یه سری توابع پس داخل داخل نگه‌داشتن به نام کپسوله کردن. یعنی یه چیزی رو داخل

کپسول گذاشتن داخل. کپسول اینجا هست توی ∞ که انگیزه اونا رو ببینیم داخل این؟ انگیزه

کلاس یعنی با استفاده از یه کلاس یه موجودیت رو تعریف می‌کنم هم یه سری

اطلاعات داره که موجودیت رو تعریف می‌کنه هم یه سری تابع که رفتار شو تعیین می‌کنه.

یعنی اگر یه موجودیتی به اسم دانشجو داریم همه پارامترها برای نمایش مشخصات داخل یه پارامتر توی کلاس داده‌ای یه کلاس

می‌توانیم برای انجام کارهای روی دانشجو توابعی مثل رفتارها و رفتارها را روی همین خط یه کلاس بنویسیم و اسم کلاس رو بذاریم

دانشجو و هر دانشجویی که بخوایم تعریف کنیم کافیه یه متغیر تعریف کنیم بگیم از نوع این کلاس هست. همه داده‌ها و متدها براش

قابل دسترس هست پس هر کلاس از نوع یک داده‌ای (اطلاعاتی) و رفتاری تشکیل شده و توی یک

اطلاعاتی چیزهایی که تعریف میشه پارامترهایی که به صورت اول رو تعریف میکنه مثلا سال ورودی یه

string چهار رقمی هست. توی یک رفتاری توابعی از جنس تابع وجود دارند مثلا تابع register. پس کلاسی به نام

student درست میکنیم احمد یه متغیر تعریف می‌کنیم و توش کلیه پارامترهای داده‌ای و رفتاری رو تعریف کردیم و اینو می‌نویسیم

Student Ahmad; یعنی احمد الان یک متغیری هست از نوع کلاس student یعنی همه اون مشخصات

تعریف شده توی کلاس رو داره. یعنی یه پارامتر مثل name و یه پارامتر تابع

مثلا برای دسترسی به اون کاپسول می‌نویسیم Ahmad.name = "Ahmad" یا Ahmad.Lastname = "shokri"

Ahmad. Register(...); همچنین می‌تونیم متدها و توابع رو صدا بزنیم.

سوال: اینجا student یک کلاسی هست که تعریف کردم Ahmad رو چی میگیم؟ یک شئ

به این پارامتر میگیم کپسوله‌سازی یا انکاپسولاسیون که همه چیز موجود شده و رفتار داریم گفتم یعنی این داده‌ها توی

یک دسته‌بندی یک کپسول قرار می‌دیم اسمشو میذاریم کلاس. و یه شئ پارامترهای خودشو داره. تعریف دلخواه

به جزء مشخصات خودش هم برای خودش هست.

اگر می‌خوایم توی این دوره برنامه کار کنیم بهترین چیز اینه: یک زبان برنامه‌نویسی به روز + یک دیتابیس به روز

مثلا زبان .net رو بلدیم یا دیتابیس SQL یا جاوا و mysql

Year. Month. Date. Subject

البته زبان C رو که همه باید بلد باشن. این امکان یه کلاس دیاگرام هست. و بنابین هم نرم افزاری هست برای تعریف ساختار داده ها در قالب جداول. روی اطلاعات نامگذاری شده و نمایش متن. نرم افزار هم از اون استفاده کند و مباحثی رو توی درس پایگاه داده می خونید.

مزایای OO که دوستش اریه بودی: پارادایم دوم توی OO، ارث بری هست یعنی یه کلاس رو از روی تعریف یه کلاس دیگه ساخت یه کلاس بلدید ولی یه کلاس دیگه ایجاد می کنید که از اون ارث ببره و کپی های کلاس اول توی اون کپی میشه ولی یه تغییراتی هم داره هم موارد کلاس اول رو بدارث می بره و تا تایم اضافه تر داره یا میتونه برخی از ویژگی ها که تعریف کرده رو تغییر بده به این میگن رابطه ارث بری. و برای نزدیک کردن به دنیای واقعی هست این هم. مثلا رده بندی های توی زیست شناسی مثلا برای موجودات زنده گیاهان، میوانات و حیوانات یا خزندگان ملخ پستانداران ملخ این دگه از پایین بخوریم پایینی یه نوع خاصی از بالایی هست یا خزندگان نوع خاصی از حیوانات هست. اگر بالایی رو در قالب یه کلاس تعریف کرده باشیم اون پایینی همه بالایی هارو داره ولی امکان داره اضافه تر یا تغییر یافته بالایی رو داشته باشه مثلا دانشجوهای مقاطع مختلف همه توی دانشجوی بودن مشترک هستن. توی uml یک قسمت از سمت کلاس که بدارث برده از کلاس اصلی اینجا wind7 داره خاص از کلاس Rootwind هست. به این رابطه میگیم Genera (تعمیم) یعنی W7 هرچی Root داره خودش هم داره یه چیزهایی هم اضافه تر (اینجا بحث ارث خونده) مثلا خیلی ها تو یه چیزایی مشترک هستن یه بدی چیزهای مشترک دارن مثلا عنوان دارن طول و عرض دارن.

و بخش رفتاری مثلا به شکل فرم ها که به سری تابع بودن این به event هست. هر جایی حجم نوع خاص بودن از

یه چیزی رو میخوایم ارث ببری هست. poly هم به پیاده سازی های مختلف اشاره میکنه. کپسوله سازی که به مفهوم کلاس

وارث بری سلسله مراتب کلاس ها هست و بخش اشاره میکنه. به بخش لیست طبقه ای دیگه ای، مشخصات attribute

گفته میشه و چند شکل هم از نوع دیتا تایپ هست مثلا str و ... و یه پکیجی math میگن برای بخش دوم که

شامل توابع هستن.

چرا مدل میکنیم؟ چون موقع یه چیز پیچیده و اطلاعاتش بازسازی کنم، نمیتونم بدم که این قابل فهمه

باشه که چی بودن تا بخش دیگه هم منظورشو بفهمه مثلا برای تولید یک ساختمان باید چی تهیه کنید؟

باید یه مدلی از این فعال ارائه کرد یعنی نقشه ساختمان. نیاز داریم برای توضیح دادن چون پیچیده

که تولید یک محصول ای از آدم ها هستن و مدل کشیدن عملا ساده کردن یه چیز پیچیده است.

مدلی که کامل کامل باشه میشه خود فعال. مدل می کشیم که به سری اطلاعات بخشی از اطلاعات براساس

یه سری قرارداد کشیده میشه. یعنی یه مدل فقط نمیتونه بیشتر مشخصات بکشه. نیاز به چند مدل از چند دیدگاه

داریم. با کشیدن مدل های مختلف مدل تجزیه میشه از منظرهای مختلف میشه. توی uml اینه که همه میتونن اونو

فهمیدن ما پیچیده است تری رشته ما نیازمند نرم افزار یک سیستم های متوسط و بزرگ هست ما دخیل میکنه طراحی و ...

داریم اینا خروجی هایی دارن. از زبان های واحد هم برای مدلسازی uml هست.

اهداف مدلینگ رو گفتن. ارتباط گروه های کاری با هم و کمونیکشن و ... مدل سازی کنه انتخاب مدل بدی نتیجه

Year. Month. Date. Subject

کاملی تأثیر دارن مدل به انتخاب کم تأثیر دارن. 2) هر مدل یه جنبه رو نشون میده نرم افزار شما منظر بیرونی داره

درون داره، یه دیتابیس داره و داخلش یه سری محاسبات انجام میده مدل های خاص لازم میکنم. 3) بهترین مدل

شبیه ترین به واقعیت هست. 4) دیدگاه که ترجمه کلمه view هست. یا aspect هم میگن. از مدل های مختلف

مدل کلا صحبت کردن یه سری اطلاعات رو نظر نظرن اطلاعات هم قرارداد.

دیدگاه های مختلف Uml (view) : 5 دیدگاه داریم که از طریق این نرم افزار رو می بینیم و توی هر view

هم مدل های مختلفی وجود داره. یعنی من با این 5 دیدگاه نرم افزار رو برای شما Hight نشون میدم.

اولی مورد کاربرد Use Case نیازمندی ها بلهایش کار داره و تمرکزش روی نیازمندی ها و قابلیت

های نرم افزار هست نظر بیرونی سیستم. 2) دیدگاه Design یعنی درون نرم افزار رو مدل کنم خود

استاده طرح ها و برنامه نویس ها و تمرکزش روی درون نرم افزار هست دیدگاه سوم process view. منظور از فرایند

فرایندهای کسب و کار هست اشاره به multithread چندنخی داره موقع اجرای نرم افزار چیزها کار به صورت موازی

انجام بشن این ها مورد دید 4) مدل هایی که برای پیاده سازی دستی تمرکز روی برنامه نویس و تکنولوژی

های مورد نیاز هست. و آخرین view 5 استقرار هست باید اینجا بیان سیستم مدل دیتابیس

اپلیکیشن مورد نیاز هست. دید استقرار هست اینا به زیرساخت فنی برگزیده شده نرم افزاری که با دیدگاه ها

و تکنولوژیت ها سروکار داره اونهایی که زیرساخت مشخص میکنن به جای hardware باید می نوشت زیرساخت

info structure ها به عنوان تکنیک و طرح اگه می خوام کار کنم که uml بیشتر اینجا به کار گرفته میشه

Year. Month. Date. Subject

مباحثی برنامه‌نویسی خودمون: بحث اول مدل‌های کوتاهی که برای پیاده‌سازی وجود داشت مثل waterfall و dev

پس یه نرم‌افزار از ویژگی‌های مختلف مدل‌های مختلف کار می‌کنه اینجا وجود داره که اون ویژگی رو نشون میده.

اسلایدبعدی: توی شکل ویژگی‌های مختلف نشون داده فقط زمانی که use case و سرعت قرار گرفته شده است اینه که

این ویژگی رو نمی‌تونستیم از خط بیرون بیاریم روی نیازمندی‌ها و قابلیت‌ها تمرکز داره. اینجا روی ۲ ویژگی

داره ورود میشه. اطلاعات براساس اون چیزی که از نیازمندی‌ها بدست میاد دقت مثل view

design با نیازمندی‌ها دارد پیاده‌سازی می‌کنیم. مثلاً اینکه use case ویژگی قابلیت‌ها و چیزهای

نرم‌افزار باید چه قابلیت‌های دیگری خواهد بود. تا پیچیده زبان uml گفتند چون دیدن شد بگیم که

ما این زبان رو تبعیتی از یکی از زبان‌های uml هست که ایجاد کردن زبان‌های مختلف

که هر کدوم زبان خاص خودشون رو طراحی کرده بودن و unifi یکپارچه بودن از این زبان‌ها زبان

طی سالیان 94 تولید شد در این دنیا و تکمیل‌هایی داشت سال 1997 زبان uml

توسط موسسه OMG منتشر شد و مفهوم مهم هست توی بحث استاندارد سازی نرم‌افزار و uml و it

زبان مدل‌سازی‌های وابسته‌گرا و کاربرد است. آخرین ویرایشی که از این منتشر شده ویرایش 2.5

هست و این اسلایدها هم بر اساس آخرین ویژگی طراحی شده. اینجا در مورد اینکه علاوه بر اینکه یه دلیل

تاریخی داره این رو نفهمیدن درست کردن استفاده از لفظ uml استفاده می‌کنیم چیزهای نوشته اولین

دلیل اینه که ما چرخه حیات توسعه سیستم رو کلاس رو پوشش دیم development life cycle

استفاده اینا کردن برای طول چرخه حیات توسعه سیستم مدل دادم مورد دوم داشتن اپلیکیشن دیتا ارکاش اینه من نرم افزار

های از انواع مختلف رو می‌تونم مدل کنم با این روش تجاری رایگان و ... مورد سوم یعنی پلتفرم ها و زبان های

برنامه‌نویسی و یعنی به این ها و برنامه‌نویسی زبان خاصی وابسته نیست قابل استفاده هست مورد 4) یعنی به عنوان

یک ابزار از فرایند توسعه نرم افزار حمایت میکنه و فرایندهای کلاری رو پشتیبانی میکنه.

مباحث اصلی توی خود زبان uml که چهار دارد: 1) بلوک های سازنده یعنی چه شکل هایی می‌تونیم استفاده

بکنیم. واژگان زبان uml که یک زبان نمادینه هست شکل های مورد استفاده هست. 2) قوانین

3) مکانیزم های رایج، سازوکارهای رایج و تونیم از مواد اضافی استفاده کنیم که قوانین رفت و برگشت دادن تربیه

مورد بعدی: تجاری که معمول نود و قبلی 5 وجود هست. توی این شکل ص 14 نوشتن به عنوان مثال

توی زبان مدلسازی شون نشون داده و گفتند بلاک های سازنده چیا هستن این الان نیه عنوان کلاس هست.

از چند تا شکل استفاده شده از یه سری مربع ها که کلاس هستن و اسمش اومده ای / رفتاری هست.

خود بلوک های سازنده هم به دسته هستن اینا الان بلوک های سازنده هستن که به دسته Things / relationsh

Degram داخل Thin اشکال هستن مثل اینا اینه و با یه سری خط وصل میشن relat ارتباط پس

یه شکلی رو می‌بینم از uml غیر از یه سری Thin و relat چیز دیگری نیست. بدون Thin داریم اینا

یعنی یه نوع و انواعی کلاس ولی رابطه اینا 3 تا هستن (ولی دو نوع) ↑ gene و associat —

و یه سری دو نوع خاص associ compo هست و اگه براساس uml برخی از Thin ها رو با یه سری روابط نشون بدی

ما اینرو به اسم دیاگرام می‌شناسیم و 5 تا Thi به نوع داریم از نوع کلاس و ما هر Thi رو هرجور خواستیم با Thi دیگه می‌تونیم فامیل

کنیم براین اساس که چه چیزهایی از تم افزار بخام نشون بدم یه دیاگرام می‌سازیم انواع مختلف توی uml مختص هست.

الان می‌خوام سراغ اینا و یکی از جلسه‌هایی که باید توضیح داده بشه هست.

Thiهای منتسب رو اینجا آوردم. یه 9 دسته دسته‌بندی داریم: ساختاری / رفتاری / دسته‌بندی / توضیحی

مستقیماً دوتای اولی اهمیت بیشتری دارن. توی ساختاری ما کلاس داریم، واسط داریم، همکاری داریم

مورد کاربرد داریم و کلاس فعال و جزء نرم‌افزاری داریم و نود داریم و توی رفتاری پیام، حالت، پکیج

و توی توضیح‌ها هم یادداشت توضیح داریم. توی رابطه‌ها ما 4 نوع داریم با یکی 4 تا پیچیده‌ترین

چیزها رو توصیف نرم‌افزار میشه پیاده کرد. یکی رابط استفاده کردن رو میشه dependency هست (ارتباط داشتن)

یعنی ساختاری associat و depe توقع هست که یه Thi از یه Thi دیگه استفاده کنه ولی توی

asso رابط ساختاری داره از این استفاده شده نمیگه یه Thi و Gen یکی رقم یافته دیگری هست

و Reali یعنی تحقق بخشیدن ما از رابطه Reali استفاده می‌کنیم. از کلاس دیاگرام‌ها هم

11 تا دیاگرام اصلی توی uml داریم. 11 نوع ولی 4 تای اول توی یک نوع دسته‌بندی میشه

ولی 7 تا دیاگرام دیگه. نمودار توالی، نمودار همکاری یا ارتباط، نمودار زمان‌بندی، نمودار مورد کاربری

تعامل، نمودار حالت ماشین، نمودار جزء نرم‌افزاری، نمودار استقرار، نمودار ساختار ترکیبی

نمودار پروفایل از این نمودارها 5 نمودار بیشتر مهم هستن و 90 درصد هم توی این 5 تا هست بیشتر اینجا

Year. Month. Date. Subject

اول نمودار useca بعدی نمودار کلاس و اگر بگم 5 تا مهم ترین: نمودار activity ، نمودار seq و نمودار stat هست. این ها مهم ترین ها هستن.

سازه کارها: مواردی که میشه در شکل ها استفاده بشه که شکل ها غنی تر بشه + تا دیدیم و در کتاب شکل ها اومده صفحه 19 مجک sperift هست یعنی بعد اینکه نمودار رو کشیدید به صورت متنی هم میشه توضیحی از مشخصات اون Thi بدین. مثلا یه کلاس ایجاد کردید یه توضیح متنی هست مثلا می گید هدف از این کلاس چی هست یا مثلا useca deposi داریم. بیایید راجع به این توضیحاتی بدید هدفش چیه و حتی میتونید روی relat هم باشه.

گفتیم که فقط نمودار دیدیم ولی با ابزار کشیده میشه نه با دست مثلا یکی از اون ها visio / SA / EA یا rationalRos یا Power designer که همین نسخه فریبدی uml هستن. محصولات دیگه ای هم دیدیم به اینا میگیم Casetool و الان هم نرم افزارهای مهندسی نرم افزار از uml ساپورت میکنن.

البته خودمون میشه زینت و تزئین

مورد بعدی adorm (جزئیات) هست مثلا برای یه تیم جزئیاتی رو اضافه می کنه مثلا توی قسمت لول بخش بالایی رو آوردم و قسمت دومش که رفتاری هست متدها رو مشخص کردن. و از سمبل های دیگه هم گفتم.

Constaint یعنی محدودیت داخل گروشه میذاریم و محدودیت میذاره مثلا کلاس بانک اکانت هم به per هم به Corp ولی با سود بالا امنش کردیم محدودش کردیم بعدی رو دیدیم بعدا مورد بعدی یعنی کلاون برنج اون Tnin مشخصات دقیق تری ازش بدید توی Tay و که داخل گروشه هست توضیحات ویژه

باشه یک ابزاری رو یا مثلا روش باید این نسخه فصل باشه میشه مقدار پیش فرض زدن شده یه توضیح اضافه

و STV یعنی یه Thin انواع مختلفی داشته باشه مثلا actor که اون هم Thin هست میتونه هم سیمی باشه هم آدم

باشه پس از این نوع حالا بعضی ها دنبالت هستن بعضی ها هنوز رفاقت کنن یعنی یه STV چند تا

بشن. اول واحد Thin ها میشیم بعد از اون role و بقیه رو توضیح میدم.

کتاب منبع برای uml : uml in practice

جلسه بعد : همونطور که جلسه پیش گفتیم یه سری اجزای سازنده داره و تقسیم بندی هاش رو گفتیم. یکی Thin اجزایی که

بین role ها هست و ارتباط ها و دیاگرام ها. اولین Thing که وجود داره، چیزی ساختاری هست کلاس هست

به همین خوبیم که در برنامه نویسی شیء گرا / رفاقت یک موجودیت اصلی داره نام / صفت ها / اوپریشن ها یعنی متدها

و به زبان ساده بالاتر یعنی عملیات یک کلاس که رفتار یک کلاس هست و بعدش قسمت توضیحی که میتونه اضافه شه یا خیر

و از روی نیازمندی ها حتی بعضی ها هنوز ok و غیر ممکن شدن اینو توضیح میدیم و از روی اون نوشته میشه.

یک کلاس باید یه موجودیت مستقل رو نشون بده که دیگه تاثیر نخواهد باشه که اینجا به صورت نکته آورده و مجموعه ای از

کلاس ها باید تعریف بشه

مورد بعدی : واسط . ممکنه کلاس رو پیاده سازی کنه ولی نمیخواد همه این متدها رو در کلاس های دیگه قرار بدیم و برخی

برای خود کلاس و داخلش هست مثلا همون شبکه متد جراغ شبکه رو برگردونه فاصله نقطه چند تا متد دیگه از این

کلاس صادر شده اون متدهای یک جلسه های داخلی مختص هست و نوع که جراغ رو برگردونه برای همگان یکسانه

و اصطلاحا private هستن. اون متدها که برای داخل هر کلاس هستن و اونایی که بقیه کلاس ها هم میتونن

همانطور بزنن میشه interface. هرکی با استفاده از این int چیزی رو بفرستن میتونن پیاده بکنن. inte در دسترس

همه قرار میگیره. نکته: چون هر چیز داخل کلاس قابل تعریف داخلش یه inter میتونم جدید اونا خودشون شدن. Win کلاس

اسلایدبعدی usecase: این Thi های تبدیل داخل نرم افزاری بودن ولی مفهوم use داخل نیازمندی هاست

و مفهوم بیرونی هست در حدود تعریف کردن هست. مجموعه عملیاتی که تعریف میشه ما یه چیزی رو در اختیار

Actor قرار میدیم. usec. Actor کسانی که با سیستم های ما تعامل دارن اطلاعات میدن یا میگیرن

این دیگه ما یه الگوری داریم که شکلش کشیدیم اینو وصل کردیم به usecase، placeorder یعنی با این تعامل

داره Acto من، و نکته مهم کلاس مفهوم داخل نرم افزاری اما use بیرون نیست از بیرون

کار داریم، کلاس در زمان طراحی ولی use در زمان آنالیز و دیزاین هست.

هر usec ها funct نرم افزار خاص مشخص میکنه. هر Act میتونه به یه کاری داره Active یعنی تعامل داره میتونه

سیستم باشه یا فرد مثلا یه سیستم دیگه بیاد کمک بگیره! کمک گرفتن هست مثلا یه سنسور هست و یه سیگنالی میفرسته

خود بعدی Collolo هست یعنی ما یه عملیاتی که می خوام پیاده سازی شه توی سیستم ما چند تا سیستم دیگه میشن. کلاس

درس. کلاس استاد چند تا کلاس درگیر میشن اینو با Collo نشون میدیم که با خط چین به کلاس

وصل میشه یعنی بوسیله این کلاس ها پیاده سازی میشه. نکته: رابطه شکل 1 به 1 هست معمولا

این یعنی مثلا یک سفارش به Collo بعد میدیم که معلوم شه با چه کلاس های پیاده سازی میشه ولی

رابطه کلی اش یه رابطه 1 به 1 هست ولی uns میتونه عملیاتی بزنه که به چند جزء تقسیم شه ولیکن

نیاز مواقع برای یه use یه call در نظر میگیرن و رابطه call با Class 1 به n هست چون call

دردس نرم افزار و US بعدش نرم افزار پس ممکنه برای پیاده سازی این چند کلاس دیگه بشن یک call

n کلاس رو دیگه بکنه بردلش ؛ کلاس با call رابطه اش 1 به n هست . یکی معمولی کلاس های

کلاس یعنی 1 کلاس میتونه n call مشارکت بکنه مثلا یه کلاس مثل دانشجو فقط توی یه call حضور نداره

چون تدریس باید کلاس معمولا برای یه use یه call میگیریم پس نکته اول ضمنا هم مشخص هست که اسمش اینه

یعنی تحقق بخشیدن use case چون نشون میده برای هر US چه کلاس هایی درگیر هستن و چطوری

میان چطوری رو نمیگه . یعنی use که یه مفهوم بیرونی نسبتی هست توسط چه اجزایی داخل سیستم نرم افزاری

که یه کلاس هست پیاده سازی میشه .

Thing یکی Component هست ؛ یه سری کلاس که داریم هر کدوم که یه فایل میشن ، یه کامپونت میشن

یه جزء فیزیکی ای هست که مربوط به چیزی تحت بوده که کامپونت مثلا خاص برای بارگذاری هست .

میتونن رابطه use کردن با هم داشته باشن یه فایل یه چندین فایل یه هم وصل میشن

از این طریق و بین فایل ها هست فیزیکی هست مثل use شیء و قابلیت بایندینگ

داره مثلا comp بارگذاری رو معرفی کنه و مجموعه ای از inter ها هست کلاس های هستن سند و یونیک ها

که برای کامپایل کردن و تولید کردن لازمه .

Thing یکی node هست یه عنصر سخت افزاری . comp فقط برای معرفی های نرم افزاری و نود اسم خاص

Year. Month. Date. Subject

بحث نظارتی هست اینجا ۳ تا سرور هست مثلا یه مدیریت نظارتی بنام ریتابیس سرور یعنی نرم افزارهای ما یه نام نرم افزار

های کنترل سیستم مثل اوراکل دومی اپلیکیشن سرور یا یه روش ...

اینکه یک ملاک کنیم باید داخلش این Compon رو کشیده یعنی چی؟ inter بود یعنی یه جزء نرم افزاری یعنی

روی این کامپوننت App سرور یه نرم افزار هست و این روش نصب شده پس بالا نیست چون نمیتونه

دست شما هست چطوری مثل کنید.

رفتاری

مورد بعدی message هست. متدهای هر کلاس میتونه فراخوانی بشه توی زمان برای نرم افزار. این رفتاری کردنم

افزاره افزودنش نشون میده حاصل فراخوانی متدهاست و کلاس ها متدهای هم میتونه رو فراخوانی بکنن و

برای نرم افزار چیزی جز فراخوانی متدهای کلاس های مختلف نیست. این که یه کلاس متد یه کلاس دیگه رو

فراخوانی کند میگیم یه Message این شکل اول Collob دیاگرام هست این کلاس اولی گفته

get name از کلاس دوم میاش بدنه خود کلاس دوم هم میتونه متدهای کلاس های دیگه رو فراخوانی کند و اینا در واقع

یه فلش نشون داده میشه Messag میش میگیم. آیا فراخوانی متد یه کلاس دیگه انواع mes دیگه

هم داریم؟ بله. مثلا فرض کنیم فقط فراخوانی نیست اما کلاس ها میتونن طوری که اونی هم با هم ارتباط داشته باشن

همزمان و غیر همزمان: همزمان یعنی به محض اینکه از کلاس ۱ به کلاس دیگه ای ازش رو صدا زده

Asyc یعنی همزمان اینارو انجام بده. بقیه هم که Crea به نوع کال برای ایجاد شیء خاصی هست

Delet حذف یک زمان و چیزی برگردونه بیگه مدلش هم زیاد شبیه replay

Year. Month. Date. Subject

ماشین حالت: وضعیت های مختلف یک نرم افزار رو توی حالت های مختلف و رفتارشون به حالت های مختلف

نشون بدیم. هرکدوم از این باکس های کوچیک یک حالت هستن، AT توی شکل نشون وضعیت دستگاه

و وارد کردن کارت یک رویداد هست که شماره از sta انتظار برای ورود کارت شماره به st. انتقال

زمان می‌بره. مثلا دیگر زمان تازه انتقال شد این یه رویداد هست معمولا جایی که AT با سیستم تماس گرفته

رویداد اتفاق میافته. وضعیت های نرم افزار رو نشون می‌دیم دیگر انتقال این یه رویداد هست شماره

می‌بره به وضعیت انتظار برای ورود دستور یا انتقال دستور.

stat ماشین حالت خاص هست برای هر جا پیت برای موجودیت های سیستم بود

این خطی که از open به full کشیده شده یک event هست که تکمیل ظرفیت اتفاق بیافته full میشه

یا میره closed یا جدید یعنی زمان - enrolled تموم شده میشه تو وضعیت بعدی.

خطی که از full به open رفته چه نوع رویدادی هست؟ یعنی یکی اومده حذف کرده کلاس تکمیل ظرفیت بوده یکی

میاد حذف میکنه. Thing های ساختاری که Thin های مهم بودن نوع بعدی server دسته بندی کردن

بودن اولی packa هست متفاوتی هم فایل ها توی فولدر ها دسته بندی میشن pa هر کی

هر اپلیکیشن توی uml توی پوشه‌ای یا پکیجی ترکیب می‌کنه. 70 تا فولدر مدلی 70 تا فولدر کشیده و مطلب

می‌تونید 70 تا پکیج بذارید و اسمشو بذارید As-is وضعیت جدید و مطلب هم to-be

اینارو با پکیج نشون می‌دیم. تو درس هم می‌تونش باشن

Honar

Year. Month. Date. Subject

Thing بعدی Note هست از نوع توضیحی هست Anno. زبان uml انواع Th هارو در اختیار داره اما باز

شاید بخوایم یه توضیحی به یه عنصری مثلاً Thi اضافه کنیم از Note استفاده میشه و بنابه خیلی طولانی باشه.

ولی توضیحاتی که روی خود مدل و دیاگرام به هم میتونیم از این استفاده کنیم.

Relationsh : رابطه های بین Thin ها رو تشکیل میده و خیلی مهم 4 تا رابطه داریم 4 نوع.

از 4 تا اولی از نوع Associ هست (تخصیص) ۲ در اصل روابط برای بین کلاس ها استفاده میشه جاهایی که هم

هست البته و اصلاً رابطه Asso از نوع رابطه ای قوی هست و یعنی object های اینها با هم ارتباط دارند.

حواستون باشه این با این ارتباط داره و ساختاری هست مثلاً فرض کنید کلاسی داریم به اسم Stude و کلاس درس

آیا باهم ارتباط دارن؟ آیا ایشاد کلاس st با آیشاد کلاس درس ارتباط دارن؟ مثلاً اعم را از کلاس st

و معدلی نم اقرار مثلاً از کلاس درس بین این ها ارتباطی وجود داره؟ توی سیستم آموزشی داریم کار می کنیم.

بله ارتباط دارن. یعنی مثلاً رضا دو تا درس برداره توی یه ترمی پس بین شون ارتباط وجود داره. آیا ساختاری

هست یعنی آیا بنا شه هویت کلاس دچار دستکاری میشه؟ بله در سیستم ها کارش همینه نمیتونه ارتباط بین دانشجو و درس

از بین بره و کنش یعنی از هویت دانشجو داخل این سیستم دستکاری میشه. مثال شون بیت هر کدوم از

چه فقط برای کلاس استفاده کرده و رابطه ساختاری دارن توی هر دو کلاس یا کلاس Font

1) میگه به ارتباط بین کلاس ها که نشان می دهد برخی ارتباطات خود ساخته و باطنی

2) یه رابطه ساختاری که شرح می دهد یه مجموعه از پیوندها که یه لینک ارتباط بین اشیاء از این نوع کلاس هست

Honar

Obj های این ها با هم رابطه دارن. اگه رابطه ساختاری از کلاس اصل حذف بشه فرزندهاش دچار مشکل میشن یه مثال میزنم

و برای تیم هم هست و نمیشه حذفش کرد. ص 35: توی رابطه ساختاری یه طرف کلاس مثل A و کلاس B

برای رابطه Ass (خط وصل) هست اگه ازین دو سیم متصل، label اسمشه و A و B

سمت هر کلاس یعنی دو طرف این رابطه کلاس های A و B هر کدوم چند نقش بازی میکنن. برای کلاس ها دو تا چیز

مثل A mult و B هم نوشته میشه. نشون میده که یه Obj از طرف دوم که B باشه چند Obj از طرف A

در ارتباط هست و یا ~~[illegible]~~ سمت A مثلا سمت B درس رابطه هم اتفاق میفته یا درس خاص از

طرف Obj به نام کلاس n با چند Obj از طرف A رابطه نوع کردن و فقط چند مثلا میتونه اتفاق بیفته مدل 0

مدل n پس می‌نویسن 0..n کلاس B ؟ 1 دانشجوی خاص مدل و دکتر با چند درس رابطه اتفاق

کردن مثلا ؟ اینجا رابطه بین اینا many هست. A mult و B هم اندازه اینور که 1

هست یعنی 1 استاد خاص فقط با یک (one) در ارتباط هست هر استاد فقط یک کورس ولی یکی یعنی

یه درس خاص مثل تیم هم استاد و دکتر n استاد رابطه هست یه درس رو یک درباوش کلاس های مختلف بدن

36 mult رو توضیح مثال زدن کلاس g1 و f1 اگه 2 ای 6 یعنی یک کلاس مثلا یک تیم مثلا 2 دکتر 6

یعنی یه پلیس دقیقا باید تیم در ارتباط باشن مثلا فرزند ← یه مادر مثلا با یک دکتر با چند بچه در ارتباط یه

بچه دقیقا با 1 مادر در ارتباط هست. یه بچه میتونه بیش از 1 مادر در ارتباط باشد اینا غیر مثلا دیگه شاید اینه.

مهم اهمیت نداره فاکتیم اصلی مثلا 1 یعنی دقیقا 1 و 2 یعنی دقیقا مثلا و بعدی گفته AC بعد اینا رو میشم

Year. Month. Date. Subject

obj های فیلم، فیلم های ذخیره شده بازیگری کردنی رو باز کرده * گذاشتن یعنی میگیم دکلرش با چه Ac و چه Fi

گذاشتن همون Ali هست که به بازیگر میتونه توی چند جا و به فیلم چند بار بره

چهار کلاس

3/5 : رابطه چه از چه نوعی هست توی اینجا چهار تا فلش دادیم؟ associati همون تا الان

اینو گفتم. بین کلاس store و post ← cont که همه از نوع assoc هستن. الان میبینیش میم

از کلاس یافته ماشین نیاز داریم که توی مثلا sto باشه فقط post باشه محصولهای یعنی فروش

بدون store حداقل یک 1 و حداکثر n و به بیت فروش خاص فقط برای یه sto هستش n

هست تعدادش توی cap به فروش خاص دقیقا توی اینه یه فروش یه بار پرداخت

یه sale خاص به یه pay m خاص وصل هست یه چیزا نمیتونه وصل باشد البته به کلیدش دارن دیگه لطفا

1 pay دقیقا به 1 sale ربط داره

یک کورس خاص mi با n و m و st افتاده، یک st حداقل 0 و حداکثر n کورس میتونه اخذ کند

حالا این ارتباط رو نوشته اخذ کردن Takeas و obj هارو نوشته Ali و DB و Ali

این دو تا درس اخذ کردن منا هم این دو تا درس رو و یه دونه st میتونه با چند تا کلاس Cont obj

ارتباط داشته باشد این ساختاری هست. این نباید سیستم چهار عملگر باشه.

Year. Month. Date. Subject

گفتم چهارتا خط دادم: اولین رابطه که محکم‌ترین رابطه هست رابطه بختاری یا assoc هست.

زمانی استفاده کنیم بین موجودیت‌های یک کلاس رابطه وجود داشته باشد و برای من مهم باشد اطلاع داشتن در مورد اون مثلا کلاس درس

و دانشجو توی سیستم‌ها برای من هست بدونم چه درسی رو میاد Reza اینا موجودی هست یا DB وقتی بخوام بدونم

هرکس چه درسی اخذ کرده ارتباط بین این موجودیت‌ها برای من مهم است. این رابطه Ass هست که در مورد شما

صحبت کردم علاوه بر خطی که بین دو کلاس کشیده شده یه خط کشیده شده و mul هم نوشته شده اینکه چند موجودیت یک موجودیت درس

با صفر و یک چند موجودیت st در ارتباط هست رو با mul st می‌نویسیم و برعکس درست درس

و همچنین این 0 و 1 شرط تیره است که گفتیم می‌تونیم اهمیت کم و کاردینالیتی اهمیت بیشتر داشته و تعداد رابطه 1 به

1 و 1 به 1 براساس کاردینالیتی‌ها هست که نوشته میشه مثلا کاردینالیتی 1 و n Sa هست یک رابطه Comp ass

تو رابطه Assoc ما دو تا رابطه خاص داریم یعنی برخی از روابط assoc از این نوع هستن یکی agg و یکی دیگه

Comp یا ترکیب هست. وجود ارتباط خاص از Ass هستن به شرط دیگه‌ای هم باید برقرار باشد.

وقتی که Assoc داشته باشیم هر کدوم از دو تا بخش موجودیت‌های فرعی هستن. Aggrea زمانی تعیین کردیم رابطه کل-جزء

whole-part داریم. یعنی کل از یه سری جزء تشکیل شده که کل بدون جزء‌ها نمیتونن هویت داشته باشند

مثل تجمیع Aggrea که نداشته باشیم میشه ترکیب. هر نوع رابطه Agg از نوع Ass هست و برای کسی که

اینطوری تعریف کرده رابطه کل جزء یعنی نمونه‌های یک سمت تجمیع مثل نمونه‌های سمت دیگه تجمیع اون هست

سمت کل ما هست یه سمت part ما هست

Year. Month. Date. Subject

یک Assoc که مثل میگه رابطه Has-a رو و وجود یک اشیاء و توش به صورت مستقل از طرف یک

وجود داشته باشن یعنی رابطه کل جزئی که طرفین هویت مستقل دارند. مثال: مدرسه و دانش آموز

مثلا سیستمی که اطلاعات دانش آموزها و دارن پس کلاس school داره و میخوایم از مدرسه هایی

سنی و دانش آموزها بدست بیاد. برای اهمیت داره که بدونیم چه دانش آموزی توی چه مدرسه ای

هست و چه دانش آموزهایی دارن و اینجا یه رابطه Asso بین نقل قدت (حفظ بدون حمید)

و بینهم 0 و ماکزیمم 1 و اینجا یه دانش آموز حداقل و حداکثر با 1 مدرسه در ارتباط هست

ولی کیس Asso مربوط به تحلیل کردن هست. این 0 و چه تعم یعنی یک sch حداقل

0 و حداکثر 1 sch با هاش در ارتباط هست یعنی sth میاد یا وارد نشده باشه یا چند sch در

خدش. اگه فقط یه رابطه خط معلوم بود درست بود و این نوری داره اطلاعات بیشتری

بدون یکه از نوع aggr هست. اینجا مدرسه کل و دانش جزء هست

اگه طرفین کل جزء مستقل هویت مستقل دارند و اجزاء از بین رن کل وجود دارن Agg دست

در صورتی که باشه ترکیب هست. اینجا school بدون sth میتونه وجود داشته باشه؟ بله. از لحاظ فیزیکی هم هست دیدن بدون

اسم دیگر Ag هست Consist of یا is part of هست. و هم obj هم یکی از سمت دیدن نیست هر سمت

میتونه مستقل وجود داشته باشه. sth هست school is part of یا برعکس sch Consi sth

یعنی سمت اجزای اگر در رابطه کل، جزء قبل به اشتراک گذاری نباشن مثال: school و depart اگه sch باشد

یعنی کالج باشد که دپارتمان های مختلفی داره. یعنی مثال قبل مثل این یک رابطه 1 به n است

یه school هست 1 و دیگری n dep

n depa و school سمت 1 باگزیم n هست و رابطه assoc هست اما به خرجی داره سمت sch به

صورت مستقل میتونه باشد یا سمت dep میتونه جدا باشد از این sch بره به sc دیگه اما تو مثال قبلی داشتیم

اجزای نمی‌تونستند. پس سمت پارت part و honsha هست. who بدون part میتونه وجود داشته باشد

رابطه از نوع comp هست ولی کمی قوی‌تر هست: compos چون علاوه این قبلی‌ها رو داره به

چیزایی هم اضافه کرده که یعنی بدون این نمیتونه وجود داشته باشد. ترکیب یه رابطه

غیر متقارن هست Agg هم یه رابطه غیر متقارن هست به طرف whole و به طرف part هست که هر دو w هستند

. اگه تو کلاس بگن یعنی اگه A با B رابطه com داشته باشد و B با C هم پس A با C هم داره

اگه فقط Asso رو مشخص کنم و خط نمی‌کشم در مورد این دو تا نمی‌تونم اشکال داره؟ نه هر طوری از جزئیات رو میخوایم

می‌تونیم نشون بدیم. و هر رابطه assoc جزء این دو تا است میتونه ass باشد ولی نه Ag باشد نه com

باشد. براساس طراحی و تحلیل این ها کشیده میشه توی زبان uml.

چه فرقی داره شکل Agg با Com؟ در مفهوم Com قوی‌تر هست. چون قوی‌تر هست. مثلا صحنه 4D

خوبی مفهوم هست. به مثال زده گفته: org و به Tea و به per دارم. سازمان، تیم و شخص

دارم. اصل داره به بیرون هست و تیم های کاری باشن به سری اطلاعات دقیق‌تر از شخص داده‌ای دیگر

Year. Month. Date. Subject

حالا فرض کنید هر سازمان چه تیم های داره پس رابطه Asso هست و توی هر تیم چه کسانی کامیتر

بین زوج هاش یه ارتباطی وجود داره. هر چند Asso و اعلی ارتفع Com و دهی ارتفع Avg

یه سازمان از تیم های کاری تشکیل شده و بدون تداخل وجودی ندارن. دهی اما اگه افراد

متین جداشن بدن به تیم دیگه پس Agg هست.

رابطه دوم depend & هست. خیلی ساده رابطه use کردن هست استفاده کردن. هر وقت یه

سمت از سمت دیگه استفاده می کند بین اون دو رابطه depe یا use کردن کشیده میشه تا کلاس

هم وظایفش هست از سمت چیزی که استفاده می کند به سمت چیزی که استفاده می شود.

وقتی که توی یه کلاس، متد یه کلاس دیگه فراخوانی میشه رابطه use کردن هست و کلاس A متد B

رو فراخوانی میکنه اینجا Asso نیست چون تو نمودار نمی گم اطلاعات شو. من تا حد متدی ازش رد

میشم و فقط بین کلاس ها هم نیست هستند درمورد چیزهایی که هم باشه وقتی بین

دو تا کلاس رابطه us کردن میکنیم یعنی کلاس Gr از کلاس poi استفاده می کند یعنی متدی

از کلاس هم فراخوانی شده. چه زمانی توی کلاس اول درکی متدی از کلاس دوم رو صدا می زنم؟

هر کلاس یه سری متد داره هر متد یه تابع هست یه کدی هم داره توی اون کدی که توی اون تابع نوشته شده

لوکال داره ازش استفاده کنیم پس بین اینا رابطه use کردن هست. کلاس اول استفاده کننده کلاس دوم

استفاده شونده.

مثلا در کلاس Giv و یه کلاس Om داریم مثلا برای نرم افزار گرافیکی بودن و قرار کنند Giv یه متد draw داره.

باید یه سری نقطه بذاره تا رسم بشه پس برای متد draw از یه کلاس دوم استفاده میکنیم پس اینجا

رابطه use کردن هست. تو مثال دوم یه فرم داریم و یه کلید روی فرم این اطلاعات رو داخل دیتابیس

میذاره که یه متد داریم که فقط توی همین یه کلاس Info داریم مثلا که کلاس Re رو صدا میزنه.

نکته: تغییر در کلاس اول بعد روی کلاس دوم تاثیر نداره. مثلا متدی که توی کلاس اول باشه اگه عوض بشه مثلا

از دو تا شیء دیگه استفاده بشه اون چیزی که از کلاس دوم توسط کلاس اول دیده شه اگه تغییر بکنه داره

شیء رو تاثیر میذاره. اگه مثلا int یه شیء رو تغییر بده این کلاس دوم متدهایی رو به دنیای خارج داده

شیء int اگه تغییر کنه این کلاس اول که داره ازش استفاده میکنه دچار مشکل میشه.

رابطه generalization یا تعمیم: یه رابطه isa هست و به اسم inherit ارث بری هم شناخته میشه

اگه یه جزئی نوع خاصی از جزء دیگه باشد رابطه بالا بر قرار میشه. مثلا توی دنیای موجودات زنده

خزندگان هم جزء حیوانات هستن و پستانداران هم جزء اون. این یه رابطه isa هست

یعنی is-a (یک) اینجا حالت خاص از یک چیز بزرگتر حیوانات هستن به اون

بالایی میگیم شیء پدر و شیء فرزند یعنی داره از پایین به بالا پر کله هست این رابطه.

از عنوان خاص تر به عنوان بزرگتر تعمیم داده میشه. راه های بعد خاص تری (فرزند) نمونه های

برای اشیاء بعد هست که بالایی رو میگیم والد. اینجا کلاس های بالایی والد و اشیاء

پایتی child متن از پایین به بالای فرم ؟ Circle is a shape . اینا داده ها usecase

متن یعنی عملیاتی که مجزا به انتقال میشه . کلاس مستطیل یا بیضی حالت خاصی از شکل هستند .

کلاس عمومی form مثلا که همه متدها و داده های عمومی اونجا هست تا کلاس جدید بهش برسه و اونا که ارث

ارث بری میکنن توی فرزند و چیزای خاص بهشون اضافه میشه . توی کلاس ها اطلاعات و رفتار داده

به فرزند به ارث میرسد .

رابطه تحقق Realiz : وقتی یک سمت وظیفه پیاده سازی سمت دیگه رو داره . یه سمت به قرارداد

سمت دیگه پایبندی واقعی که اون قرارداد رو پیاده سازی میکنه یک رابطه معنایی بین دو عنصر

که در آن یک عنصر یا یک جزء تعهد میکنه پیاده سازی چیزی که توسط جزء دیگه انتظار می رود .

کمتر ازش استفاده می کنیم . توی هر حالت نمیتونه دیده بشه . وقتی یک سمت interface یک سمت

کلاس باشه . یعنی Reg وظیفه پیاده سازی IRunM رو داره . یک کلاس وظیفه پیاده سازی یک inte

رو داره . مثلا Gen ولی خط چین داره به جای فلش هم مثلث داره . از سمت پیاده سازی کننده به

سمت پیاده سازی شونده هست . توی uml جای دیگه که می بینید بین یوزکیس و Collab هستش

که چون یوزکیس هایی که باهم همکاری می کردن بود پس اون به یوزکیس سمت راست که

وظیفه پیاده سازی سمت چپ رو داره .

اینا 44 بعد تقریبا دیاگرام های uml رو گفته ما هم اون دیاگرام هایی توی uml 2.5 ، دیاگرام های ساختاری

Year. Month. Date. Subject

structure دارد و بعضی نمودار رفتاری behav . در زمانی قسمت ساختاری چیزی که استاتیک هست توضیح

می‌دهد مثلا مدل و معماری ابزوره ساختارش هست چه چیزی کجا قرار داره چه ارتباط‌هایی با هم دارن .

مثلا کلاس دیاگرام جزء اصلی‌ترین قسمت‌های ساختاری هست یعنی این بلیس نقشه‌ش چی هست ساختارش چی

هست . از نمودارهای دیگر obj و پکیج و ... این شکل کرنگ‌ها یعنی توی روزش اصلی نیست و بعدا اضافه کردن .

توی نمودار استقرار depl چی می‌کشیم؟ سرورها و ارتباط‌شون رو می‌کشیم . این نمودار استقرار ساختار رو توضیح

میده یا رفتار؟ ساختار پس جزء نمودارهای ساختاری هست . و نمودارهای ساختاری هم سمت چپ هستن

حالت اجرایی و برنامه‌نویسی و رفتاری رو توضیح نمیده . توی قسمت رفتاری یوزکیس دیاگرام، sta، act و st

mes هاستن . ۵ تا نمودار هست که لازمه بلد باشیم: ۱- یوزکیس دیاگرام ۲- کلاس دیاگرام ۳- inter دیاگرام

و activ دیاگرام ۵- statem دیاگرام

این خط ← در سمت چپ کدوم رابطه هست؟ یک رابطه gener هست . می‌خواد بگه ۴ نوع خاص داره

و آخرشم وصل کردن به دیاگرام beh یعنی چه . این ۴ حالت خاصی از هم هست . بین یوزکیس دیاگرام

و کلاس دیاگرام رابطه gener کشیده .

یوزکیس دیاگرام: نمودار مورد کاربرد زمانی کشیده میشه که تو مرحله کلیات نیازمندی‌ها هستیم یعنی محدود و نیازمندی‌های

سیستم و رفتاری که از خارجش نشون میده و ما به بیرون کار داریم نمودار رفتار بیرونی سیستم هست .

یوزکیس ویو شما مرکز بود قبلا و بقیه کنارش بودن . اصلی‌ترین Thin های که همه نمودار use cas هست دو تا چیز

Actor و use که هرکدوم رو بگیم، use case میتونند از عملیات دیگر سیستم جدا بشه و جایی کار نامه

یه use case هست و تعریف حساب یه یوزکیس هست. اکتور: کسی که با سیستم تعامل داره آدم یا سیستم باشه

مثلا مشتری شعبه شرکتی جدید ایجاد میکنه ایجاد شرکتی یه یوزکیس اکتوری که تعامل داره با این یوزکیس یه اکتور

هست و یه نقش هست. اکتور در انجام یوزکیس شرکت میکنه یا اطلاعات میده یا دریافت میکنه یا هر دو

و برای به انجام رسیدن اون یوزکیس این تعامل لازم هست.

توی یوزکیس دیاگرام موظفیم طبق تصاویر پیش برویم. اکتورها و یوزکیس ها. سوال: یوزکیس دیاگرام در حقیقت

یه نرم افزار به سوال what می پردازه یا how؟ چه چیزی هست این نرم افزار چه کاری به ما میده؟

رفتار بیرونی رو توضیح میده یا درون نرم افزار چه اتفاقاتی میافتد؟ بیرون پس what

توی طراحی به how می پردازیم متدها، الگوریتم ها و کلاس ها اون جاست.

یوزکیس دیاگرام به what می پردازه. یوزکیس دیاگرام یه نمودار ساختاری هست یا رفتاری؟ رفتاری دیاگرام

چون رفتاری هست به این که مثلا هر کنش چه کارهایی میکنه مثلا درست اش، رفتارش و...

داخل سیستم نمودار کاربرد اینه که نیازمندی ها رو انجام بدیم اول و زبان مدلسازی هست.

اکتورها ارتباط شون با یوزکیس ها چیه؟ اینجا توی شکل 48 از نوع Assoc هست.

باید ببینید چطوری باهاش ارتباط داره؟ استفاده اش میکنه؟ پیاده سازی میکنه؟ یا رابطه ساختاری

داره. چون رابطه as بین کلاس ها توضیح دادیم ولی بین چیزهای دیگه هم میتونه برقرار باشه اگه رابطه

Year. Month. Date. Subject

اکتور با یوزکیس قطع بشه بعدش دیگه اجرا نمیشه و مثلا اکتورها حذف بشه لنتاج حساب استان نمیافته.

اگه قطع بشه توی تعریف یوزکیس دچار مشکل میشی رابطه اکتور و یوزکیس همیشه 0..* هست

یعنی جلوی فلش هارو اصلا نمی‌کشم. همیشه معمولا جهت استفاده اطلاعات هست اگه دادن اطلاعات

هستش اکتور اطلاعات میده به سیستم سفارش میگه پس اینوره اگه اکتور اطلاعات دریافت کند از طرفی

ولی چون معمولا هر دو هست پس خط بدون فلش می‌کشن. پس معمولا هر دو حالت هست.

بین اکتورها میتونن رابطه gen برقرار باشه مثل مثال جلسه 48 مثلا اکتور اپراتور و مدیر

فرض کنیم. هر دو رو تعمیم میدیم از نوع کاربر باشن که میتونن login کنن

پس به این معنی که اپراتور و مدیر هم میتونن لاگین کنن. ببینم اگه بین مثال رابطه gen هم بزنه

مورد بعدی 49: هر کدوم از اکتورها به کدوم یوزکیس وصله.

لیست یوزکیس هارو درآوردیم اکتور هارو مشخص کردیم و دیدیم کجا تعامل دارن وصل بشکل کنیم

این یه سیستم آموزشی هست اینجا چندتا اکتور شناسایی کردن؟ 5تا همه این اکتورها انسانی هستن؟

Office منظورش یه عامل انسانی هست. اکتور یا عامل انسانی یا غیر انسانی یا غیر افراد هست ولی اینجا

منظورش مشخص و خاص کدوم کاتالوگ هست. همه اینجا عامل انسانی ان. این شکل‌ها خیلی

بیشتر مفاهیم هست و شکلش کردیم در ادامه دوره نوعی اکتور مشخص میشه. st یه تا یوزکیس داره

یا یعنی prof یه تا یوزکیس میشه. آیا یه اکتور میتونه به چند تا یوزکیس وصل بشه؟ بله

به بردن پوزیشن میتونه چند تا اکشن فعال بشه ۳ بله این پوزیشن CloseR هم بدیم آموزش بالانس مقابل بلو

هم بدیم قیمت مالی ۰ و پوزیشن R با Bill ارتباط داره. بدیم ثبت نام CIO میکنه ولی منتهی این کلا به قیمت

مالی اعلام میشه ولی به تتیجه‌ای رو می‌گیره تو اتاقش داخلی نداره اکشن آخر

پس بیشتر این سه تا فنش رو فعال کن یه پوزیشن gen کن راوتیه log فعال کن.

Year. Month. Date. Subject

برای کیس use باید actor ها رو شناسایی کنیم و با خط‌های Asso به این خط‌های use وصل کنیم.

هر use مجموعه‌ای از عملیاتی هست که توسط سیستم یا یک actor ها تحقق پیدا می‌کنه و نتایجی رو خواهد داشت. این معادل هست که بیایم به ازاش کاری انجام بدیم یا به دست آوردن اطلاعات یا دادن و ... بین actor ها می‌تونه رابطه تعمیم gen باشد یعنی دو تا اکتور عمومی‌تری از اون act باشن و به این وصل باشن اینجا هم OPK هم پیش‌فرض به یه اکتور کاربر تعمیم داده شده و اکتور کاربر به یوزکیس‌ها کلیک متصل هستن.

بین خود use که رابطه تعمیم هست بین use ها هم می‌تونه وجود داشته باشه اگه یوزکیس حالت خاص از یکی دیگه باشه یا رابطه include با هم متصل بشن مثل مدیر حالت تو صفحه 50 یعنی این اتفاقاتی که توی والد یا پدر توی فرزند هم هست اما متون عملیات خاص خودشون رو هم اضافه یا کم کن در فرزند اگه زیاد باشن با استفاده از pk دسته‌بندی می‌شن استفاده از st هم زمانی پیش میاد که هست یعنی مثلا باید اینا یوزکیس‌های دیگه‌ان یا اینا سیستم‌ان و میتونید به یوزکیس‌ها اضافه کنید.

این st رو بیشتر برای همه به کار برده. Business یوزکیس یعنی یوزکیس که در ارتباط با کل

مثلا st تایپ بیزنس یوزکیس یعنی طرز عمل فرایندهای که ممکن استفاده بشه اینا پیش فرض هستن مثلا ویژگی‌های

الف و ب اهمیت کمتر و چ هم نره که اینا بین یوزکیس‌ها که رابطه تعمیم باشه و نکته بعدی هم اینه که

act ها و یوزکیس‌ها رابطه بین شون از نوع Ass هست. یوزکیس کریتیکال اهمیت اکتور پولیدر هست

و گفتیم یوزکیس ما که هدفش ایجاد ویژگی هست و دو راه هست که شما گفت ولی اینجا مدل ندارید تحقق پیدا نمیکند پس

کنید این رابطه گذاری ندارم

رابطه Assoc اینجا وجود نداره. دوره بعدی اینکه رابطه Asso میتونه جهت داشته باشه یعنی اگر

Art کلاس انجام بشه، اطلاعاتی در اختیار سیستم قرار میده اینجا بایت رو از اکتور به سمت UC می‌گیره.

مثلا سیر فروش یا صدور فاکتور، اینکه میخوای هست که کاری با سیستم انجام بده، اطلاعاتی میده

که فقط به تحقق بفرستد UC میشه پس از سمت AC به سمت US هست که شیء کلاس دو لیفاف

Art To کلاس داده بشه وقتی اون ارتباط می‌شه و نتیجه این در اختیار سیستم اکانتینگ قرار میگیره

و رابطه بین این دو تا رو جلوتر می‌گیم. این یوزکیس تولید سند حسابداری هست که نتیجه این

به سیستم حسابداری فرستاده میشه. پس این قسمتی از یوزکیس یادگیری هست. صادر شدن

فاکتور فقط به صادر شدن میشه. سیستم فروش سند حسابداری صادر کند بفرستد به سیستم اکانتینگ

اکتور حسابداری اینجا سیستم و نرم افزار هست (اگر سیستم هم تعامل انسانی هم میتوانند) اینجا دیگه یک یوزکیس داریم

که نتیجه این در اختیار اکانت سیستم که یه نرم افزار هست قرار داده میشه و یا سیستم ما تعامل داره.

ما بین یوزکیس‌ها می‌تونیم رابطه depen داشته باشیم اینجا یوزکیس صدور فاکتور و صدور سند حسابداری

به dep بین شکل نسبت یعنی وقتی یوزکیس صدور فاکتور انجام میشه داخل هدفش یوزکیس فاکتور VSC رو

هم داره میکنه و نتیجه این به acc sy فرستاده میشه اینجاست که رابطه dep بین این دو تا برقرار هست.

رابطه یوز کردن بین دیتا خیلی چیز استوار بر قرارداد یا پلیس بین یوزکیس هست

Year. Month. Date. Subject

بعد فاکتور بالا رفتن از سند حسابداری استفاده میکنه و هر کدوم یوزکیس جزء اگه جدا کانتکس استفاده میشه هست.

اما توی یوزکیس صدور فاکتور داخل خودش به این یوزکیس نیاز پیدا میکنه و از این استفاده میکنه جهت طلبش رو نگاه حالا یا اول یا وسط یا آخر اجرای صدور فاکتور، از صدور سند حسابداری استفاده میشه اما دوستش در اختیار هست

ما دو تا حالت داریم یکی به این صورت که یوزکیس A و B داریم رابطه dep بین شون برقرار هست و با بازیچه include رئیم روشن، عملیات B حین انجام عملیات A صورت می‌پذیرد عملیات صدور سند حسابداری B حین عملیات صدور فاکتور A انجام می‌شود. حالت دوم: با پرچم یا استریوتایپ extem هست. جهت این پرچم هست A یوزکیس اصلی در ابتدا اجرا میشه و B در انتها اجرا میشه توی include اجباری هست داخل متن هر B رو انجام میده ولی پایین اختیاری هست.

یوزکیس B به صورت اجباری انجام میشه پس از اجرای عملیات A انجام میشه و B به صورت اختیاری A رو اضافه میکنه

مثلا یوزکیس A صدور فاکتور داریم توی متن صدور فاکتور یه رکن هم میاره که به انتخاب کاربر پرینت بگیره یا نگیره پس یه یوز کیس پرینت فاکتور داریم ولی همیشه اجرا نمیشه پس یوزکیس B که پرینت فاکتور باشه برای یوزکیس A حالت اجباری و اختیاری داره. پس dep رابطه این دو حالت داره و include میشه در حین انجام A، B انجام میشه و extend بنابه شرایط میتونه انجام بشه B.

مثل بعدی: نوشتن یوزکیس ص 53 بیگ لس وظایف اسلاید قبلی همه اینا وجود دارن یه اکتور داریم اکتور Ac و Sc و همه این ها gen میشن به اکتور user. که به یوزکیس لاگین متصل هست یه اکتوری بیشتر اینا رو gen به اکتور یوز که به یوزکیس لاگین متصل هست یعنی این دو تا وقتی لاگین کنن

Honar

Sale موتور ISS بکند و سند تاکتیک صادر کند سند حسابداری نتیجه اش به SGA به ACC سیستم صادر میشه

و در فاکتور ساعت فروش و در سند حسابداری هم انجام میشه

پس رابط صدور صورت حساب با صدور سند حسابداری یه رابطه inclu هست از طرفی رابطه صدور صورت

حساب و صورت حساب ext هست یعنی بعد از صدور صورت حساب بنابر شرایطی print bar میتونه انجام بشه

اینجا دوتا یوزکیس فرض کشیده on prin و on whi که نوع خاصی از use پرنت فاکتور

هست و با هم متفاوت هستن. مثلا یهی جاها صورت حساب از قبل طراحی شده

به صورت پیش فرض یه زبانی روی کاغذ بیفته اولی یه سری اطلاعات خاص مثلا مالیات و ارزش افزوده

باید بیفته بعد در مورد این خاصیت های اطلاعاتو چک میکنه و اگر نیاز دیگر رو کاغذ نیست فروش باید چاپ کنه

این فلش ها تایپ هست در صفحه 54 جزوه نداریم.

بیزنس رو دلیلم توضیح میدم یه قسمت هایی سیستمی و دیگر باید

یه سیستم در حقیقت، یعنی یه یوزکیس کشیده و یه آرایش هواپیمایی هست و درجه stree زده اینجا الان

یه تیکه هست که طبقش این یوزکیس ها هستن و اسمشو گذاشتیم آرایش هواپیمایی و یه قسمت هایی

دستی باشه. این ها بیزنس یوزکیس هستن. این خط های ارتباطی این اکتورها هم یا نقش برچسب نقش بزنند

هدف این سیستم نیست هدف فرایند هست اینجا دوتا اکتور شناسایی کردیم یه tour op هست یکی

passeng هست یعنی مسافر فرد Tour رو به pas یعنی دلال یعنی نوع خاصی از pa هست

و tourg هم یه use کیس های pass دسترسی خواهد داشت. پس یه هر یوزکیسی دسترسی

داشته باشه و چهار تا یوزکیس هم شناسایی کرده.

گروب چیک یعنی چیک کردن دست جمعی مثلا اختیاری یا اجباری هر دو نوشته و روب

Subject

Year. Month. Date.

inclu داره به indiv چیکار یعنی وقتی ترتیب انجام میشه چیکار میخوایم انجام میشه Tow میبینی دیدرتمی داره

این داره شکل میده pas هم خودش به صورت فرعی میتونه indi چیکار انجام بده این رابطه ass اینجا

خود بلد رابطه exte هست Bag رو چی هست؟ همون اینه یعنی میتونه اون نیازمندی های IA نیاز به

bag چیکار میشه یا نیاز به این کار داره. پس به ازای چیکار هر فرد میتونه باشه یا نباشه ex شده هر pa

هم باید sec پشت رو بسازه اینا.

بین اکتورها و یوزکیس ها رابطه multip هم میتونیم بذاریم رو سیستم ...

پیچیدگی به نظر میرسه یوزکیس سیستم یا خزانه انجام کار بیزنس یوزکیس

مثال بعدی: نمودار یوزکیس سیستم. نمودار یوزکیس مناسب برای توضیح سیستم هست نه خزانه کاری برای مدلسازی و ایده

سلوهای بهتری مثل bpmn دارم.

یه بخش از خود کاربرد سیستم به سیستم مثلاً نمودار این ۴ تا یوزکیس رو کشیده اسمشو نوشته چک اوت نمودار سیستم پیشنهاد

و یکی نیست نمودار مربوط به سایت سیستم چک اوت هست چهار تا یوزکیس برای قابل شدن ... و Help

و دو تا اکتور در نظر گرفته، گفته Fast میتونه chec کنه و che هم توی سیستم چک کردن شرکت میکنه

سیستم هست هم یوزکیس هم اکتورها میتونن یعنی با به سیستم نرم افزاری قابل دادن

توی چک اوت کردن یه سری جزئیات هست هم کاستومر هم کارگر، این یه نمودار هست فقط

به ازای هر یوزکیس یه سند باید دربیاره که اسمش هست use case specification (یعنی جزئیات خود کاربرد) و سناریو بیزنسیه

اون تسته در کنار این مدل نیازمندی ها و کلاس دیاگرام شکل میده. حالا بریم توضیح بیشتر این نمودار رو ببینیم.

فرض کنید همین قسمت کمک بکنه به کاستومر برای چک اوت کردن و در نهایت اینه help بکنه بیش ex داره

اصلی داره بکنه کار نیازمند help بلکه باید باشه برای نیازمندی هست رابطه dep یا pa و che

هست یعنی قطعاً عبارت pay هست بلکه حتماً chec و اجباری هم pa به جای همین میدم دارم

Honar

مثلاً در کلاس بانکی IPG

Year. Month. Date. Subject

برای انجام یوزکیس Paym ، اکتور Paym که میتونه دو تا اون ایراتور بانک باشد خارجی میشه.

یه اکتور دیگه هم گرفتم به اسم admins که به یوزکیس mana وصله که یه پیز yen هست. کاربرها سطح استفاده

کاربرها و دسترسی دیگه اینجا فقط usea منبع بروزرسانی هست.

بعد از مدل اکتیویتی دیاگرام رسم شد

بعد از فعالیت ها بیزینس فلو چارت هست، هرجایی که بخوایم یه جریان کاری رو انجام بدن دیدی رو مدل کنیم و دقیق

چطوری انجام کار

هاش نشون بدیم از این استفاده می کنیم. پس برای پیاده سازی جریان کاری هست. میتونه تو سطح فرایند باشد. خزانه کلی

پیاده سازی مثلا یکی از راه های توسعه نرم افزار

یه سیستم یا زیرسیستم. بخوایم یک یوزکیس رو توصیف بدیم. نقشه فعالیت دیاگرام هست یه راه یا برای نوشتن

جریان کاری و توصیف این

فعالیت در سطح یه کلاس یا یه متد. چند پیاده سازی سیستم رو توصیف میده و جاهایی فرایند میتونه کشیده بشه

رفتاری

مثل یه self service هست. میتونم اینو که مثلا ای از چند مدل یا خروجی و دیاگرام های استفاده کنم یه تعریف متنی و تعریف هر دیاگرام

زبان uml

این نقشه میتونه تو سطح فرایند کشیده شه، ما توی اکتیویتی دیاگرام یه سری المنت داریم مهم ترین اکتیویتی

SVD

اکتیویتی یه فعالیت هست

المنت

هست که شلوغش به این صورت هست. میتونه این فعالیت دو حالت داشته یا ریزدانه باشد یعنی میتونه

بعد از شلوغش باشد و قابل تجزیه باشد یا نباشد و زیر کارهایی باشد یا نباشد قابل اینترایت باشد یکی اش میتونه توجه کلی

خیلی ها باشد. در سطح متد یه کلاس یا oool که اینجا ریزدانه تره پس به دست کلی توی uml استاندارد

دو تا اکسیچ state و act رو تعریف کردن که sta گفتیم

Honar

Year. Month. Date. Subject

Action رو تعریف کرده مثلا کوشه پیاده اینک . دل رفت اینا در دیتابیس . جمع کردن چیزی رفتن در میتونه تمرکز روی اکتیو قسمت
روی فعلی . act و actio فارسی میشه

Ration Role بیشتر Activ دیاگرام می‌بینیم Action . اسم Activist هست . Activ فعلی هست و شکل بیشتر بالایی

متدلید نظر باین هست

بیشتر Acti به چیزی بر اسم stat هم داریم یه وضعیت رو اشاره می‌کنیم مثلا الان در وضعیت ... درست ولی

کلاس روی Activ توجه می‌کنیم . پس فرق این دو تا در درست و اشتباه هست acti تایل ترجیح میشه کلاس نمیشه stat
acti توجه می‌کنیم

این نسخه اسلایه 58 و 59 رو رد می‌کنیم . تست بعدی ...

60 چیزی که ما داریم برنده Transit هست از یک sta | Acti به یک sta | Acti رفته دو حالت
Tri یعنی هم شرط نداریم cati قبل تمام شده بعدی ولی gua
داریم Trigless هست یا guar | هست یعنی شما شرط می‌ذاریم وقتی قبلی انجام شد شرط برقرار بود

می‌ایم روی بعدی . و وقتی روی این فلش گارد کلیشش بتونیم .

پس Tran بین act و start میتونه بدون شرط باشد یا با شرط gu هست . نقطه O یعنی شروع

Tv

مورد بعدی Bran هست . با چه مسیر میتونید از این خارج بکنید زمانی که به ازای شرط مسیر جریان
یک خارجی
کلاک متصل میشه . اینجا مقصد داره روی نتیجه Verify pi داره یا اگه inc باشد از این مسیر اگه cor باشد
اینجا یک dec و شرطش باید بنویسید و

از این مسیر می‌ریم . ما acti داریم sta داریم Tran داریم بدون شرط یا با شرط و یه dec
داریم زمانی که در نتیجه فعالیت فرضیان یه چیزی رو چک کنن در مسیر جریان کلاک تو تعریف میشه

61 توی اینتیتی دیاگرام تعریف for و هر داشته باشید دو تا RR اسمش هر فاز هست زمانی که به جایی
رسیدیم که چند تا فایت مختلف هم ← for اگه انجام شد اگه نظام دارای یک جریان بزرگ و اولید بیم ← Join Honar
مثلا ۲ و ۳ و ۴ و مثلا انجام شن for بعد اینکه کیفیت اتم شد join میشه و حرکت بعدی

Year. Month. Date. Subject

نکته آخر: اکتیویتی های موازی باید توی SWI های مختلف صورت بگیرن یعنی توسط انجام دهنده های مختلف صورت بگیره تو ایلاید

fork join

یعنی دوتا فرم موازی توسط انجام دهنده های مختلف صورت بگیره ولی یکی از دو فرم مورد بعدی باید منتظر بمونن تا نکته بعدی

تمیز و واضح اطلاعاتی داشته باشن مستقل نباشن از هم

swim lane ها می‌خوان به اکتیویتی دیاگرام بگن که انجام دهنده‌هاش باهم متفاوت هستن شرکت برای تفکیک وظایف انجام میشه

مثلا میگه swim کاستومر اینو انجام میده swim lane customer → call sales

انجام میشه

یه اکتیویتی دیاگرام هست که یه زیرسیستم رو توضیح میده چون درست طرف هست و بزرگ میشه اینجا رو تایلیشم و هر کسم

روتوی swim انجام دهنده‌اش میکیم swim ها میتونن اشاره به موجودیت های مختلف یکی توی

هر سطحی که یکی اینارو / اینجا رو swi میذاریم / یه اکتیویتی رو توی SW انجام دهنده‌اش می‌کیم. / swi میشه

اکتور باشه یا فلوهای کسب و کار یه سازمان باشه و هر swi نشون میده که هر کس چه وظایفی انجام میده

خیلی کمک به پوزیشن دیاگرام داره اکتیویتی ها، پوزیشن فروش و ... مثلا داره ساعات فروش رو مدیریت میکنه

بالای پوزیشن دیاگرام ها C

دوتا پوزیشن Take order و Collect F داریم. دوتا پوزیشن هم اینور داریم ship part و make داریم.

بیشتر بازی بی‌ثباتی

این نمودار اکتیویتی دیاگرام پایین نشون میده که پوزیشن بالا هم ربط داره؟ الیتم اکتیویتی دیاگرام رو بهتر سطح

کشیده میشه پوزیشن اومد از کلاس، زیرسیستم، کل سیستم. این اکتیویتی دیاگرام میخواد هر پوزیشن ship part

رو نشون بده برای هر پوزیشن یه اکتیویتی دیاگرام کشیده و برای پوزیشن VC اینو کشیده

اول انتقال قطعه از تولید به حمل و نقل بعد توی مالیات پاکیج پات بعد یه راهی هم میره

برای اون شرط دو حالت اتفاق میافته یا ship next day حمل در روز بعد و در حالی که

ship second Air میشه یعنی دو روز بعد به صورت هوایی صورت می‌گیره

این یه قسمتی از سیستم هست. اول PVC نوشته شده چند تا نقشه بعد این رو حل می‌کنه بعد یه جا حل بکنیم میتونن چطوری صورت بگیرن یکی یکی بالای CVR و بعدی ship هست لیستی حل و بعد این رو تا چطوری انجام بشن بهم گره شده join صورت می‌گیره هر دوی این‌ها انجام می‌دادن، یه‌جایی می‌گم کاستمر رو به اطلاعی به کاستمر بدیم، یه تا خطا یا — هم کلید همزمانی هست.

9:47

Year. Month. Date. Subject

بله ← این کار در زبان ولاسیک صورت می‌گیره

exit ← موقع خروج مثل از یک ایستگاه

enter موقع ورود wrap part صورت می‌گیره

بله به آدرس این شکل میشه

جدا این شرط روی weighpa اتفاق بیفته

توی Trans بله ← یک gnar

تو acti پایان ، Lepive صورت می‌گیره تمام نهایی

64 بلکه انتخاب

65 این یه مثال خوبه مدیریت صنعت هست ، مثلا سندهای مختلف تولید شه دیاگرامی و یا تبدیل فرآیند

پس سیستم‌های مدیریت دانش این‌ها رو بلدیم این کاره ایشاره توضیح میده ، چهار تا انجام دهنده‌های نهایی این هست

Author , reviewer , approver , owner

فعالیت Route ، نرم‌افزار انجام میده یعنی بررسی درست می‌کنه

یه این مسیر یه زنگ

یک حامل این فعالیت یه شیئی به نام Document از نوع Draft شده که دارنده

reviewdr مقاله گرفته و بینگارد review انجام میده . کوشش‌ها یا activ هستن ،

کوشش‌های دیگه این

Document (Review)

عرضه‌کننده هوز استفاده‌کننده نشده خیلی این شکل

Honar قبلی وضعیت این زود تلاش عمومی زمان می‌گیرد حالا این

Year. Month. Date. Subject

زمانه در اختیار app قرار می‌گیره و خروجی این دیاگرام Defi که موجودیت هست و به صورت
کارد روی Trans کشیده شده، و اسم جریان کاری مختلف داره میره به سمت auth
و بررسی می‌کنه اگه appro باشه داکیومنتی که وضعیتش شده effecti میره اینجا
تا بازیابی‌های رو انجام بده. اینجا هم میتونه نتیجه changen باشه یا absul باشه
اگه cha باشه این chan هایی که ver نوشته زمانی که
may Requ هست و میره به سمت Anthor که باید upd کنه و اگه abso باشه
میره سراغ own و arch میگه و نتیجه‌ی داکیومنت آرشیو شده
این، swim تمرکز کرده هر کاری رو چه کاری انجام بده مرحله‌ی ما چطوری
هست وضعیتش چطوری تغییر میکنه و ... و lifetim شئ داکیومنت رو در چرخه
حیاتش نشان میده.

GG موقعی که بخوایم نقاط تصمیم ثبت بشه این دو جریان کاری موازی هست
یکی برای جمع تصمیم یکی برای صورت حساب این کمک میکنه fork و join
کشیده

یه نکته: اینجا این یه branch هست ◇ یه میره ورودی و دوتا میره خروجی داره
یه ورودی و دوتا چند تا خروجی داره آخری ◇ نیست یک نقطه اتصال هست join
اگه چند تا جریان کاری مستقل باشن چند تا جریان کاری به جریان کاری

Year. ——— Month. ——— Date. ——— Subject ———

موارد زیر نیست ولی تغییر ساختار مطالب و مفاهیم رو متوجه بشید با فرض جزوه ۵ و ۶ هستیم

Year. Month. Date. Subject

کلاس دیاگرام اصلی توان حاصل هست

کلاس دیاگرام های کامل

کلاس دیاگرام در دو مرحله، آنالیز و طراحی رفتیم تو مرحله طراحی بودیم نمودارهای جدیدی تو آنالیز نیازمندی نبود.

تو مرحله طراحی دیگر جزئیات کلاس ها و ارتباط هاشون مشخص میشه، طراحی میشه و مدل اصلی که توی فاز آنالیز شکل گرفته میشه کلاس دیاگرام هست و کلاس دیاگرام جزء نمودارهای ایستایی و اساسی هست.

توی دیاگرام ها interval diagram - sequence diagram و دیاگرام state diagram

استفاده میشه کلاس دیاگرام ها برای مدل کردن: شکل و ظاهر نقش ها و مسئولیت های کلاس

موجودیت هایی که رفتار سیستم را ایجاد می کنند.

اگر قرار باشد ساختار کلاس ها که معماری سیستم رو شکل میده، و کلاس ها با ویژگی جزئیات دیاگرام ها، ارتباط هاشون مشخص بشه، تو مرحله آنالیز کلاس های اصلی بدون جزئیات و کم کم تو این مرحله کلاس ها به طور کامل در میاد.

پس کلاس ها زیاد هست توی چندتا پکیج پخش کردیم از اون زیرهای منطقی و زیرسیستم ها

برای پیوند اینترنتی هست

هستن ممکنه به زیرسیستم ها شلوغش و نقش های متفاوت داشته باشن یعنی فرم متفاوت

شکل میده. اینایی کلاس هایی که مربوط به پیوند اینترنتی ها هستن.

مراقبت اند سمت اینترنتی و بانکی، بلکه مربوط به پرسنل و سمت سرور هست.

Honar

scope لذا نوع کار رو مشخص می کند

Month. Date. Subject

کلاس هاشو توی 5 تا پکیج از کلاس ها تعریف کرده که توی یک پکیج یکی کلاس باشه مثلا اول چیزی رو براش دیدگاه تم انتخاب میکنه یعنی وظایف زیر نظر به چه وصله از لحاظ کارش بشوکیند یکی کس یه نرم افزار و بعدی بیزینس سرویس ها و کلاس های کارهای داده ای توی دیتابیس کاردوی داده شرکت انجام بدیم و بعدی کلاس دیتابیس هایی که به اشیا بیزینس سیستم اشاره میکنه کلاس دیتا بیس برای کارهای عمومی همه از نوع رابطه depen

چند رفتاری: مثلا چرخه میتونه یکی از این سه تا باشه /

یه کلاس comm و dep داره و یه کلاس off و یه کلاس head یعنی هر دو (سازمان) داره و خودش پدر هست و per و con داریم و یه کارهای دیتابیس هست. comp رابطه 1 به چند با این دو تا داره و رابطه dep با of که چند به چند هست. رابطه dep با خودش رابطه ساختاری داره یعنی میتونه از چند تا dep تشکیل بشه یا به تنهایی dep بالایی اش داشته باشه یک رابطه 1 به n به ترتیب (رابطه بازگشتی) و به سلسله مراتب اشاره داره.

و رابطه هم با head و of داریم چون head یه نوع of هست پس دو تا کلاس میتونند رفتار رابطه as از لحاظ تمامیت مجموعه داشته باشند. یه رابطه 1 به n داریم (members) البته با گذاشتن دو طرف چند داره میگه یه per میتونه توی چند تا dep باشه. این کمک رابطه 1 به چند هست man یعنی یه per، یه ma یه dep هست. یعنی dep یه per به عنوان مدیر داره یا یه pe میتونه مدیر چند تا dep باشه. ارتباط مدیر بودن زید خودش ارتباط کارمند بودن یه dep هست یعنی مدیر باید یه mem کارمند dep هم باشه.

کلاس دیاگرام: یعنی وارد نموده طراحی میشیم، مدل های قبل توجه که آنالیز نیازمندی ها کشیده میشد این به طراحی کشیده چه کلاس های داره چه عملیات هایی داره، چه ارتباط هایی با هم دارن کلاس دیاگرام های اولیه تو مدل آنالیز کشیده میشد ولی تو مدل دیزاین کشیده میشه. یه کلاس دیاگرام عوناری هست که نشون میده یه نمونه از کلاس ها، همکاری ها، ارتباط های لازم داره. نوتاسیون خودش اش با ابجکت دیاگرام هست. کلاس دیاگرام یه نظر ساختاری هست (درون نمایی) گفتم مدل بود. این چند استاتیک مدل سیستم رفتار سیستم و حالت اجرا رو. اگر نمودار کلاس دیاگرام به پیشرفت پکیج ها شکسته شده. البته اگر داره اینا با هم ارتباط داشته باشن.

Honar

Year. ——— Month. ——— Date. ——— Subject ———

پس این کلاس ها باهم فرق دارند اینجا کلاس ها رو توی ۵ تا پکیج پیش رفتیم

بنابراین اول براساس دیدگاه نرم افزار سه لایه هست اشاره به این دارد که وظایف نرم افزار

از لحاظ منطقی ها به چند سطح میشکنند. یک نرم افزار بانک بخش رو پس های کپ دلار

کلاس هایی که برای حسابات هست، کلاس هایی که مربوط به کارهای داده ای هستند توی این قسمت

مربوط به یه موجودیت خاص مانند حساب معلومیت مشترک پس باید روی داده مشترک کارهایی

رو انجام بدیم و کلاس. یکم دیتابیس کمپیوتر نیم هست و کلاس عمومی داریم که

ما کنیم اینا اینجا میان چون خیلی هم از فرم ها و گزارشات از این استفاده کنیم هم برای

این فقط لایه های دسترسی نداره. و دیگاه پایه و سطوح های کلاس ها رو نشون ندادم

که: label هست و دو تا textbox و یه تا button هست. برای فرم کلاس خواد

اسمش product فرم هست این یه کلاس مربوط به interface هست یعنی قرارداد با کاربر ارتباط پیدا کند

کلاس عمومی نرم هست و همه اون ها از کلاس پروداکت فرم به ارث می برند و از نوع عضو هست.

پس توی تعریف ساختار این پروداکت فرم داریم از یه کلاس دیگه استفاده می کنیم توی تعریف ساختارش توجه

کنید یعنی اگه butt ضرب شه تعریف ساختارش چه جوری میشه پس این رابطه ass

هست و جهت طبق به اینجا چون توی تعریف پروداکت فرم از VB استفاده کردیم و پروداکت فرم به بانک نیاز داره و Tex هم به طور آخری use کردیم هست و توی کلاس فرم از

ریکورد ست استفاده نکردیم پس مثلا وقتی روی فرم دکمه اضافه یا حذف رو می زنید مستقیما جلوش توابع مدیریت داده خواد

یه رکورد اضافه باهم کنه کلی که این کار اضافه ای رو انجام بده کلاس رکورد دست ابجکت میدهیم

داخل این متدها به کلاس رکورد دست نیاز داشتیم پس داخل این متدها ازش استفاده کردیم.

ساختار بیت چون از کلاس رکورد ارثبری نمیکنه کلاس فرم ارثبری نمیکنه فقط متدهاش رو

میگیره.

70) برای کلاس ها visibility تعریف میکنیم قابلیت رؤیت. هر کلاس یه کلاس بالاتر رو دیدن میبینه. باعث

مسئولیت کارشون بهتر میشه و درست میبینن و درست میدن.

برای کلاس ها Mult داریم. چیزی رو که داخل خودش میتونه از ش ایجاد میکنه مثلا کلاس های وارد در الگوریتم

مثل فدرلان در اکس خودمون که باز میشه پس min0 و max1 هست یا فرم اصلی فرم لغزنده

سند و لیبل ها مثل فرم پس معمولا 1 هست میشه. برای یه کلاس توحالت کلی n تا کلاس میتونن داشته باشند

مثلا منتلیس att ها اینطوری میتونن visibility داشته باشن معمولا priv میگیرن و بنابراین فقط متدهاش میگیرند

نام میتونن داشته باشند attr هم میتونن multi داشته باشن (به معنای آرایه بودن) و میتونن Type

داشته باشن قابل بشه و prop باید مقدار میتونن مخفی کنند یعنی یه att میتونه صحت نداشته باشه

frozen constant هست و add فقط یه مقدار میتونن اضافه کرد و باید بیشتر از آیا شد نظر

و برای متدها هم منتلیس قائل میشیم. اونی که داخل کروشه هستن اختیاری هستن ابجکت دهنده

یه مقدار داشته باشد مثل دهنده و خروجی و هر چیز دیگه. مثال زیدش هم هست

اعضای متد یا لیست از نوع Veth Name رو میگیره و میگیره اعضا. اگه اعضای متد تعریف شده

باید روی صفحه چیزی باشد.

مطرح شد که چطور در مورد کلاس ها بتوانیم توی کلاس ها رابطه تعریف کنیم. در تدریس طراحی از الگوهای کلاس ها از این ایده استفاده می‌کنیم
اول یه مدل کلاس داریم و توی کلاس مدل فرم ها قطعه می‌کنیم. (سوپر کلاس یا پدر) و (ساب کلاس یا فرزند)
این کلاس ها را رابطه هست چون فقط اسم دارن خصوصیات مشترک ندارن پس تو مرحله آنالیز هستیم.
رابطه تعریف میدیم شکل های مختلفی داشته باشه مثلا یه کلاس گرفتیم به نام Asset این اشکال برای یه کلاس
باشن که بودن. در مدل ساب کلاس دارن اینجا Asset پدر و این سه تا فرزند هستن
رابطه بین sec و ass این وراثت تکی هست. ولی تو vecale هم از as ارث بره
هم بانک الحاقت هم ویندوز. و دوتا کلاس هم میشه multiple inhe
بانک الحاقت هم ویندوز. و دوتا کلاس هم میشه
یعنی در مورد رابطه associa لازمه تیغ بدن:

74 صفحه 75

76: در تاحالت میشیم

77: یه حالت خاص توی کلاس یا از از veal وراثت داشته باشه لازم قطعی کردیم

77: یه حالت خاص توی کلاس یا از از veal وراثت داشته باشه لازم قطعی کردیم

interfa: توی int اصلاحی رو می‌ذاریم که از دنیای خارج قابل دسترسی هستن

acco: هستش رابطه کلاس که بدا این را ستاره int رو پیاده سازی میکنه این کلاس

veali: بین شون نسبت وابسته کلاس بدن این in رو پیاده سازی میکنه یا اینجا

76: توی درس دنیایی دوتا جدول وقتی رابطه n به n داشت یه جدول واسطه گرفتیم

اینجا یه کلاس associals داریم که رابطه n به n داره دوتا کلاس و خودش رابطه

نیاز به اطلاعات کلاس یه کلاس می‌کنیم و به این رابطه مصل می‌کنیم یعنی کلاس که دارا رابطه

یه کلاس associa رو نگهداری میکنه و رابطه n به n هست یه عضو

توی چند تا سفارش به یک کلاس هست. باید اطلاعات شامل تعداد و یا حجم داشته باشه