

References: • Software Engineering 8th edition ...

پیشنهاد ۳ نمره

• ?

پیشنهاد ۵ نمره

• ?

پیشنهاد ۸ نمره

حضور غایب + تمرین + مشاورت در کلاس ← ۴ نمره

نرم افزار خوب: امن - این - دارای رابط کاربری خوب (کیکاربر برای بار اول که برمایه را باز کنیم نمیتوانم) - توسعه پذیر - بینهایت - و با کیفیت

لیست، لیچ نشود که چه طور باید

از آن استفاده کند

تعداد مشخصی درخواست پاسخ بدید

که آن تمیز و با کیفیت و خوانا باشد - Usability خوب داشته باشد - برمایه برای

قابل استفاده مجدد باشد - به عنوان خیزی از یک نرم افزار بزرگتر سینم

قابل بله داری باشد -

نمودن افزارهای

خت و ب

عملانه هزینه‌ی نرم افزار یک وسیله از هزینه‌ی ساخت افزاری ش بیشتر باشد -

نماید  
یکسان  
باشد

از خود کامپیوتر بیشتر باشد -

هزینه‌ی عمره در سیستم‌های ساخت افزاری، در تولید شونه:

نرم افزاری، طراحی و تولید آن: هزینه‌ی تلیرش خیلی به نسبت

کمتر

کار را نمایند که نرم افزار چه قابلیت‌هایی داشته باشد.

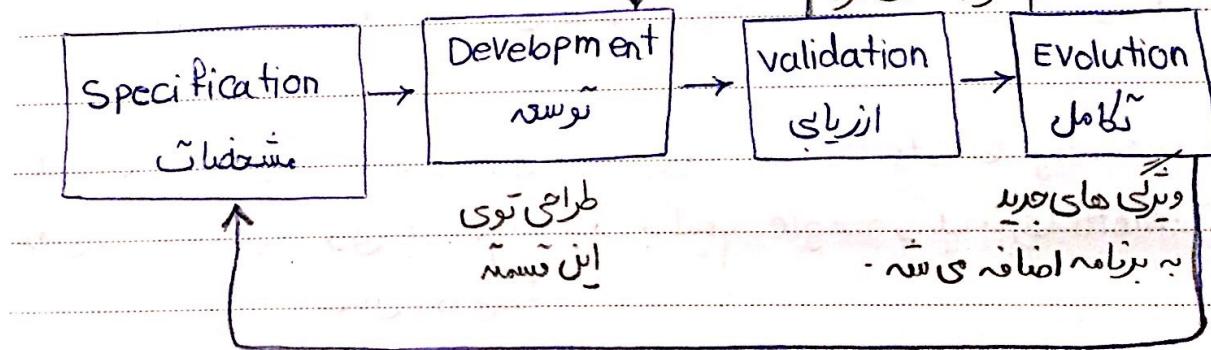
نرم افزار

ساخت و تولید کار کنند. مثلاً نرم افزارهای دولتی، یا تالیک دانشگاه

نرم افزار

اله ارزیابی مسئولیت دارند

در نرم افزار تکنولوژی



در تغییرات اعمالی روی نرم افزار  $\rightarrow$  سفارشی  $\leftarrow$  مشتری تصمیم‌گیرنده است.  
 سفارشی  $\leftarrow$  سازنده نرم افزار " " عمری  $\leftarrow$  "

فعالیت‌های حمله‌ی مهندسی نرم افزار  $\leftarrow$  مشخصات specification

سفارش‌دهنده و گروه سازنده دورهم بنتیجه در روی

مشخصاتی که نرم افزار باید داشته باشد فلکنند  $\leftarrow$  مشخص

(همان سکل ص ۲۶)

کردن محدودیت‌ها  $\leftarrow$  Development  $\leftarrow$  کلیل، طراحی، پیاده‌سازی

مشخصات مورد نظر معاشری شه زمان + هزینه  $\leftarrow$  validation  $\leftarrow$  نرم افزار توزع داده شده با

مشخصات  $\leftarrow$  evolution  $\leftarrow$  تغییرات اعمالی می‌شه تا با مشخصات

تطبیق داده شه  $\leftarrow$  تغییرات اعمالی می‌شه تا با مشخصات

تغییرات اعمالی می‌شه تا با مشخصات

روش کار ما وابسته به نوع نرم افزار ~~روش کار ما~~ است.

قابل آنکه reliable  $\rightarrow$  قابل آنکه (امنیت) امن  $\rightarrow$  قابل آنکه

safety  $\rightarrow$  این (اطینان)

reability  $\rightarrow$  قابل اسعاده بجد

maintain ability  $\rightarrow$  قابل نگهداری

بعدن

Stand-alone APPS  $\rightarrow$  نرم افزارهای مستقل  $\rightarrow$  مثلاً Word ...

به اینترنت نیاز نداشته و آن را برای هاسون روی همین سیستم نصب می‌شه و قابل دسترسی به.

دسته‌ای

یک سری ورودی که روشون کاری کنند و خروجی  $\rightarrow$  batch processing systems

رویی ده  $\leftarrow$  مثال: وقتی یک اسم search کی لیست در google و یک سری photo مرتبط با هاش

نداش داده می‌شه

... big data

نرم افزارهای اینترنتی برای معاشری دو فایل از داده و بررسی تطبیق شون

Data collection systems → google map اینلئه فرمه مالکای

نقشه ایم.

نرم افزار هواشناسی

\* درودی یک نرم افزار batch processing ای تونه خودش خودجی یک نرم افزار Data collection باشد.

fundamental principles → اصول بنیادی

نرم افزارها می تونن روش اجرا شن → یک

روی تبله ای web درخواست را قبول کنند و خودجی را نمایش دهند →

(درویدی) web

نرم افزار دهن → نرم افزار اجرایی

روی web " " web

(خودجی)

cloud computing → ?

خرابی و فرسودگی wear out:

web based systems → باید مرحله ب مرحله

بد استفاده کردن abuse:

impractical: توسعه داده شه قابل

عملی شدن نیست (incremental)

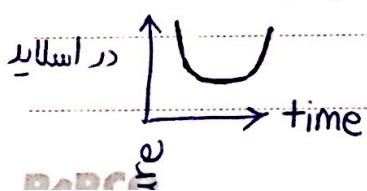
سخت افزار ← شرایط فیزیکی و نوع استفاده از شون در خراب و فرسوده شدن شون مؤثره.

مرور

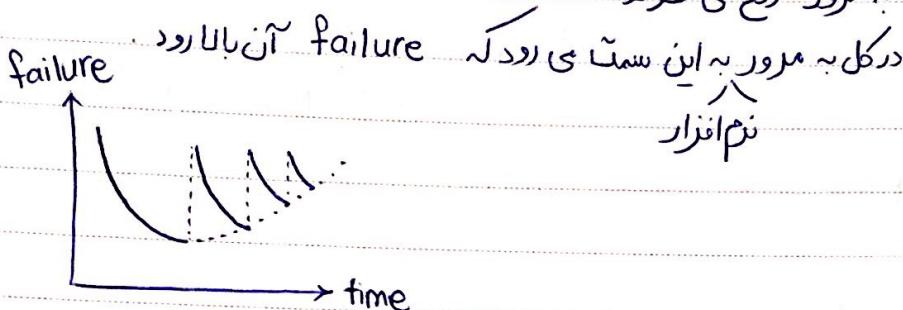
اول بروزه طراحی این را داری هست. به این ایندات لطف و حل می شوند.

بعد آن سخت افزار قابل استفاده می باشد. مناسب تا حدی است و بعد کم فرسوده می شه و همی

قطعاتش با هم خرابی شوند و دیگر قابل استفاده نیست. ← بودار



سخت افزار را معمولاً هیچ وقت نمی خصوصیاتش را عرض نماید و به روز نگیریم.  
لذا در نمودار چنانچه در نمودار failure-time نماین افزار، در هر باره روز رسانی  
دوباره ایجادی در آن ممکن است وجود داشته باشد لذا  
به مرور رفع نمایند.



### جلسه (۳)

## chp 2 processing models and A birthday party

۱- برنامه ریزی  $\leftarrow$  مکان و زمان - افراد - انتخاب برنامه - هزینه - تعیین وسائل مورد نیاز

۲- آرایه وسایل و هدایت  $\leftarrow$  آرایه حرکتی - آرایه لوازم تربیتی - آرایه لیک - آرایه هدایت - ...

۳- آماده سازی مکان

۴- اجرای برنامه

۵- تئیزتری

۶- ترقی بازخورد

فعالیت ها و ترتیب فعالیت ها = فرآیند

بعضی از process descriptions دارند. (برای مراحل بعد)

گاهی اوقات تقسیم وظایف نمایند.

برای بعضی فعالیت ها شرایط علی و بعدی داریم.

(ساخت یافته)

ی شهی پیشتر فرآیند را با برنامه ریزی  $\rightarrow$  برنامه ریزی شده اولیه مقایسه کرد.

یه برنامه اولیه مختصر داریم.

یه کاری رو پیش نماییم. بعد تقاضه هاش رو

برطرف نماییم. (کارهای روی سلولیم به بین فعالیت ها و مرحله به مرحله پیش نماییم)

PARCO

برنامه ریزی مرحله به مرحله است!

- \* معمولاً دریک فرایند از هر دروش استفاده می شد ← ترکی از هر دروش
- \* همچو روئی کاملاً درست و کاملاً علط نیست ← ولی باید درش مناسب هر فرایند رو بیا کرد.

### • مدل آشیاری : waterfall

interleaved : باهم در آن منعنه

incremental : مرحله به مرحله

incremental development → توسعه آش مرحله به مرحله است!

نه تزدیماً برنامه ریزیش! => یعنی تونه هم باش!

هر مرحله هر وقت کامل انجام شد تا به مرحله بعد!

۱- مشخص کردن نیازمندی ها

۲- طراحی سیستم و طراحی نرم افزار

۳- پیاده سازی و تست خری

۴

۵

\* مراحلس کاملاً قابل جیا سازی اند

\* عواملهای این روش ← مراحل رو بخی شه موازی پیش برد.  
\* عواملهای این روش ← تغیراتی در مسکله ← چون روش برای برگشت به فشتهای قبل تنظیم شده

\* هر مرحله به خوبی داره در مرحله بعد به عنوان ورودی ازش استفاده می شد.

\* نیازمندی ها باید کاملاً برآمدون تغییر مسکله باشند.

\* خوبی برای استفاده نیست چون معمولاً همی نیازمندی ها از اول مشخص نیست.

\* معمولاً برای بروزه هایی استفاده می شد که بین ترمه مختلف در حال کار درین راه اون درجا های مختلف باشند. ← تقسیم کارین سه های مختلف

مدیریت در چنین روئی ساده تر ← چون برنامه ریزی از اول انجام شده

در ضمن چون خوبی هر مرحله مشخصه دی شه بررسیش کرد.

(توسیه‌ی مرحله‌ای) : incremental

Development

ویرگاهای که مشخصه رعایل اجام‌ی دیم و ادن‌هایی که نامشخص بودن روی زاریم واسه بعد استفاده از نم افزار می‌توانه قبل از کامل شدن کل نم افزار شروع شه هزینه‌کمتر برای اعمال تغییرات ← دلیل: هر دفعه‌که روی ویرگاهای خود کار بشه، چون چون همه نیازمندی‌ها از اول  $\rightarrow$  تله‌ی توجهی هست، برسی‌ش ساده‌تره.

مشخصه نشده تغییرات چزو مرحله نیازمندی‌های بعی در تظر ترقیه می‌شه... دلیله‌یی رو لازم نیست تعاویت با آسیاری ← پیش‌رفت بروره رو نمی‌شه کامل مشاهده کرد (مستندات لست) عرض لیم تغییرات تله تله معلمه لینجیت کل نم افزار رو یا این بیاره  $\leftarrow$  معلمه لازم بشه

برای بالا بردن

کلیفت، به تغییر

کلی و هزینه زیاد

صرف بشه

در مثال تولد، الله تجربه‌مون توک تولد لرعن زیاد باشه، فرآند آسیاری بشه.

الله بار اول مون باشه تولدی لکیم یا ی خواهیم توی مسافرت تولد بلکیم، روشن دم بشه.

محرومیت های شه

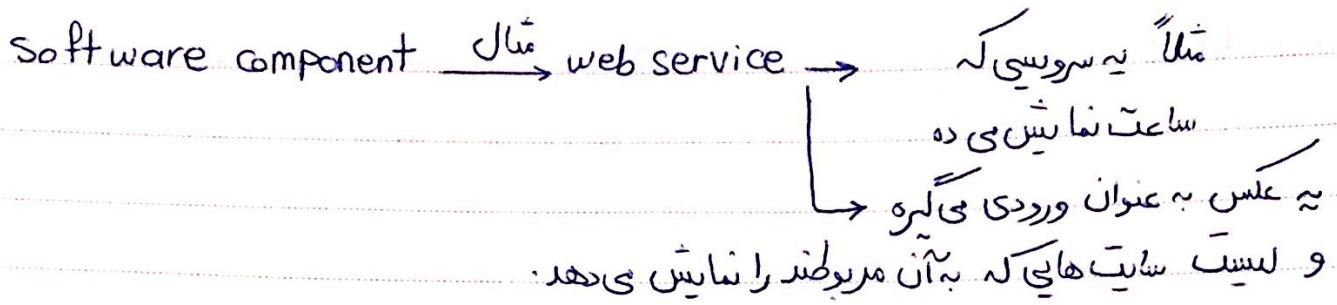
استفاده مجدد از نم افزارها  $\rightarrow$  حبسه بعد

جلسه (F)

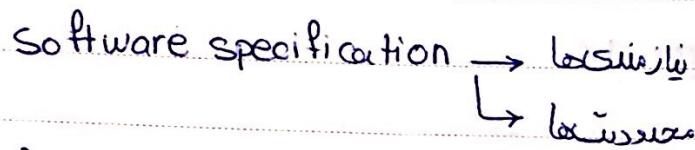
برای سیستم‌های این چنینی مناسب نیست  $\rightarrow$  بسیاری برای incremental development

re-use software  $\rightarrow$  نم افزاری که دوباره قابل استفاده باشد.

استخراج: animation, ... ک.



CMS  $\rightarrow$  Computer Management System



feasible  $\rightarrow$  معلم بودن یا نبودن ① Feasibility study  $\rightarrow$  آن مرحله سریع است (Development)

② Requirements  $\rightarrow$  استدلالی لیم خام است

Elicitation and Analysis  $\rightarrow$  ایجاد ایام بی‌نهی Survey

② Feasibility Report

③ Requirements Specification  $\rightarrow$  به صورت مشخص‌تر اطلاعاتی کسب کرده‌ایم

④ Requirement Document  $\rightarrow$  دیلیت نیازمندی‌ها کامل در اینجا مشخص شده است

Software Design  $\rightarrow$  معاری Implementation  $\rightarrow$  از معاری بی‌رسیم به پیاده‌سازی

\* Subsystem  $>$  component, module

(Architectural Design)

نیازمندی‌سیستم به زیرسیستم‌ها، کامپونت‌ها و...

Data collection

Data Base

Simulation

(Interface)

نیازمندی‌سیستم آنها

داده‌هم درین مرحله  $\rightarrow$  ③ طراحی زیرسیستم را اخراج می‌دهیم.

validation → چون معلمه تست ساختگی حواب → test باداده‌های واقعی روی ردی با test های واقعی error داشته باشد!

Acceptance testing → تست قبولی برای کاربر

تا آخرین

حلبیه بعد ← تکامل روی نه و ...

جلسه (۱)

AMD → Advanced Micro Devices → Intel رقیب  
با ارائه معماری Zen و بالا بردن سرعت CPU های Server از چند سال پیش  
تا حالا طی پیشرفت نه

software evolution: وقتی در نیازمندی‌های سیستم

تغییر ایجاد شد، (ین توسعه و نیازمندی  
تفاوت قابلی شد که در آن در هم نمی‌شوند). نیازمندی‌کاری  
جدید رو شناسایی می‌کنند و ...

Existing system → نرم افزار فعلی  
مرجعی در بازار

لکلولری‌های جدید ← مثلاً به الگریتم جدید تغییر شده که باعث بروز نرم افزار می‌شوند  
دلالی change  
محدودیت‌ها و قوانین جدید جامعه  
معلمه بیشتر شده بسته به لذت شده  
معلمه کلی که مانند افزار معرفی رونویسی شده ایم

نرم افزار ماهره بیانی (platform) تغییر کننده

لکلولری کاربر

بررسی نام

change avoidance

بیش بینی تغییرات از قبل و

برنامه زنی واسه‌ی اون بعضی

change tolerance

جا برای تغییرات احتمالی

در سیستم مون بذاریم.

با حدودی تظر → به مشتری‌ها → ایجاد کی

(بازخورد) مشتری از اون سیستم شفون بذنش ساده Prototype مشخص شده.

prototype → یک فرم هست برای شناسایی سر نیازمندی‌ها  
دارد ریزه کاری‌ها نمی‌شوند.

UI design → ی شده در نسخه اصلی اون → کاربر تطریه‌ای طری  
بنش ها بینید نظر کاربر شوند  
کاربر از اول رابط کاربری  
هیچ نظری نداشته!  
اون تاریخ prototype  
ایده‌ای به آنایی اولیه‌ی شوند و اسش  
خود developer و ... داشته‌اند.

\* مثلاً مدلن اون prototype از نیمه‌اشن، طری باشند. شیشه‌های سیاه باشند و توی ماسن دیده نشوند  
هر چند اون <sup>نکره</sup> prototype فقط نمای بیرونی ماسن بوده

ترکیبی نیازمندی‌های functional  $\leftarrow$  مسائل امنیتی، performance

Throw-away prototype → ها مناسب نیستند و اینه اینه پایه‌ی طراحی  
قرار گیرن و این ساخته شده اند که دور از اختره شوند. چون هی باید عرض شوند و ترکیبی‌های  
nonfunctional.

مدلن در prototype استانداردهای لازم رعایت نشده باشند: بیش به درد نمی‌خورند

incremental delivery → به قسیت از نرم افزار که ساخته شد، →  
کامل نه اون کارها مورد استفاده‌ی کاربر قرار گیرند تا قبل از  
(incremental development) اینه همی قسیت های ساخته شوند. ره اجام می‌دهند،

از قبل وجود داشته باشند،  
ازش در سطح کاربر اسکواده کرد  
PDRCO

Subject

Date

incremental  
delivery  
problems

ادن چیز که کاربری از این ارزش استفاده نماید

به دردش خود را

ایجاد شد در یک مرحله کار  
آسونی باشند

پایان 2

incremental delivery  
فایده

1- نسخه های اول مثل

Prototype می مونند

2- رسیل پروژه را به خاطر

فهم بتر نیازمندیها

جلسه (۴)

« chp3 شروع

3- استفاده از نیم افزار رویتر شروع می شود

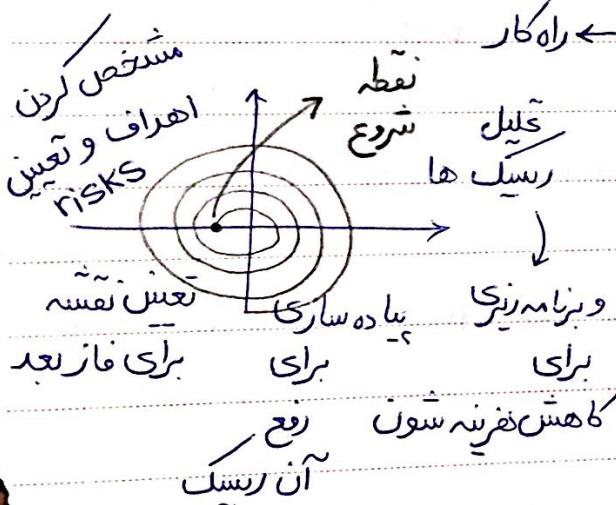
4- نیازمندی های بالاتر در نسخه های اول بیان شده اند: برای همین در نسخه های بعدی هم تست های توند و عیب یابی شون دقیق تر و بیشتر اجام می شوند.

Boehm's spiral model :

حلقه

loop که

روی risk ها تمرکز شود ← شناسایی ← ارزیابی ← راه کار ← است



risk = کامل نهایی نیازمندی  $\Rightarrow$  prototype طریق

risk = ?  $\Rightarrow$  water fall استفاده از

risk-driven

formal method:

روش ریاضی برای مدل

سازی

روش برای متعابله با

رسیل ها

برای پیشنهاد دادن

نیازمندیها

• Rational unified process → این هم یک روش <sup>گردنه</sup> generate  
UML: unify Modeling Language

3 generic process models

waterfall  
incremental

reused ?

inception: هر فاصله سیستم از دید چارکی بررسی می شود

Elaboration: نرم افزار از مهندسیات بیشتری شد + طراحی عمری → کاربری خریبات

Construction: نیازهای سازی → معماری نرم افزار → نیازهای سازی و باختن

Transition: انتقال به کاربر

متلاع: هزینه اش مملو است

آموزش به زیاد شده است!

تعداد زیادی کاربر

شکل ص ۵۶ ← فاز های مولده مرحله ای انجام می شوند

Dynamic ... Elaboration نیست بعد کامل شد ...

use cases → موارد استفاده ای Actor → کسی که با سیستم تعامل می کند  
سیستم

work flow آخوندی کاملی هستن

\* Business modeling

requirement ... کلی تراز

RUP فعالیت های سالن

نیازمندی ها را مشخص می کند

use component based architecture

آل کامپوننت ها را از قبل مشخص کرده باشیم، یعنی از

DAPCO

توسعه و تحریل سریع نرم افزار  $\rightarrow$  دلیل: الله دیر برسه به دست مشتری، مملکه یم هار کرده های دلیل تبلیش یکی با همون کاربرد مسازند و در شیوه نرم افزار ما استعمال چندانی ایش نشاند.

روش چاپک  $\leftarrow$  ترکیز روی پیاده سازی و کد هست نه طراحی  
 $\rightarrow$  به صورت مرحله به مرحله فعالیت ها تکراری شوند. (برگشت به مرحله قبل در دور بعدی ...)

manifesto: بیانیه

Agile manifesto  
تولید نرم افزار با کیفیت اولویت داره بر

پاسخگویی به تغییرات اولویت داره بر دنبال کردن یک نقصه  
و تعامل

روش های ا Formal  $\leftarrow$  ریاضی از روشن های ساده  $\rightarrow$  از حد امکان استفاده شد (ساده نویسی)

چیز روشن های ساخته یافته

اصلًا وجد دارند الله Agile این قدر قدری های حقیقی داره؟

لکچر حج

۱- نرم افزارهای بزرگ  $\leftarrow$  دلار هم باشد و دو ب دو با هم  
تعامل داشته باشد حون عوادشون زیاده تعامل ~~غیر رسمی هم~~ نیاز داره

۲- در حفظ مسائل امنیتی  
در تولید نرم افزارهایی مثل کامپایلر یک زبان بزرگ نویسی یا ناو دریایی نیفی شه نازه

۳- در مراحل بعیی خیلی تغییر داد Document

۴- الله مشتری تعامل زیاد خواهد داشت باشد  $\leftarrow$  راحت تر شد  
بالا به مستندات زیاد نیازه نداشته باشد

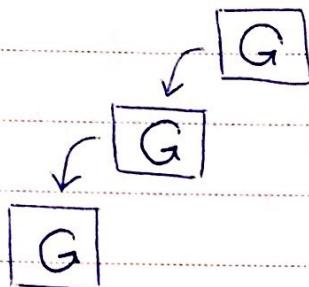
۵- اینله زود خود delivery داشته باشد، اولًا باید یه نرم افزار نیزه مستندات زیاری درین به روزه  
حین جمع کنند و ناینیا مملکه از الکترونیک های بینه استفاده نشاند حون روی طراحی خیلی ~~لهمه ای~~ کرد

وقت صرف نکرده اند.

۶- الله به یه نرم افزار زود به روز رسانی یا نهاده ای کند، حون کد های خیلی خونا نیستند،

کار ادن یعنی دوم خیلی سخت می‌شود

- \* پس به طور کلی مدل‌های ساخت یافته برای نرم افزارهای نیزگ مناسب نیست.
- \* در روش‌های چاپک هم برنامه ریزی‌هم بیاده سازی، مرحله به مرحله است.
- \* در هر مرحله از روش‌های ساخت یافته می‌توانیم چرخه داشته باشیم ولی در کل نباید مرحله‌ها رد برگردیم به عقب :



(شکل ۱۱) گویای این مطلب

در روش چاپک، نرم خیلی خفن و حرفه‌ای باید باشند دلیل بنون بعن داشن طراحی دینی،  
نرم افزار زد پیاده سازی کنن و خودشون اعضای  
طراحی رود یه جو ریکا خس بزین +  
چرخه‌ی زیاد داشته باشند تا مستندات  
بیشتری هم جمع آوری کنن و با  
روش‌های جمع آوری مستندات آسا  
باشند.

Extreme programming → XP

لے از نمونه‌های چاپک  
(تا سر XP توضیح داد)

### جلسه‌ی (۱)

XP ← توسعه‌ی مرحله‌ای روابط حد اعلایی رسونه! ← ورژن‌های جدید مملوه حذف تاتوی یه روز  
تولید شه!

هر دو هفته یکبار هم تقریباً نسخه جدیدی دست مشتری می‌دهند  
\* مراحل تولید هستند اما برعکس اند

Pair Programming →

لیکی لدی نویسی معقد نیستند.

collective ownership →

هر developer قدرکنن که کل نرم افزار را سه ای لدیه و کلمک اش بی‌کنند.

ادن و صاحب‌ش اونه. سه دست جمعی باهم کارکنند

PARCO

و آنگلست آنها می‌کنند که این را سه دست جمعی تغییر کنند. ← هر developer فقط روی یه قسمت کار

Subject  
Date

در  $XP$  ← برنامه زیری هم حتی مرحله هر مرحله است! به جویی جبران شه  
constant refactor → هر چیزی که اضافه می شه ، باید  $\rightarrow$  اسکرینری مجدد  
Simplicity بشه تا کیفیت لدیاسن نیاد. (بینه ترس نیم  $\leftarrow$  قابل درسته)  
باعث می شه نمود مستندات

ساختن تست ها → اول تست ها رای ساریم → Test - first development → بعد development رو ایجاد می دیم و هم مرحله به مرحله با این تست ها مطابقیت ای کنیم. بینیم درست جواب می دهیم است.

جون دراین روش نمی‌خوان → ساعت‌های طولانی بیشتر هم  
Pace → لزوماً در گواه مدت بیشتر  
On-site customer → کارتلند رعایتم کنن و محیله ندارند.

Task → بے کارہائی کو جوں کی لونڈی

مثلاً باروش مسندات  $\rightarrow$  حذف کردن مستندات  $\rightarrow$  test first  $\rightarrow$  Constant refactor

\* در روش XP نیاز به تعامل تنها تک مشترک با توجه دهنده‌ها است.

میں جیسے XP کو Testing \*

\* نوشتن بست قبل از کدنویسی، نیازمندی ها را قابل فرمتری کن.

\* برای معاینه نتت ها با بیاده سازی از نرم افزارهای از پیش تعیین شده استفاده کنیم.

↓ (frame work)

کاربر (مشتری) لام می‌لند در نوشتن سنت‌ها و اینله می‌بینند و ترکی‌های سنت Junit مثل

← در نویسنده نسبت از نزدیکی‌های از سیاست تعصی شده همچنین لفظی هم ایشان.



\* معلمه برنامه نویس‌ها،

کدنوشن رود به نست نوشن

ترجمه بدن و زمان زیادی روی

نست ندارن.

\* نه تهدید واسه این روش ← چون مسّری خودی پیاده‌سازی نم افشارش رو بی‌سنه،  
الله از این روش خوش‌نیاد، معلمه دیله خوب تعامل‌لنه و  
درباره‌ی تیازمندی‌ها تعلیت یکی → تطره‌ای مخالف منی بر روش پیاده‌سازی بده  
ی لنه

\* حکم‌لدن اینله نست‌هایی که نوشته این کافی هستند یانه، کار سختیه: اولاً هیچ وقت با نوشن  
هیچ مقید نست و معاویه اش پیاده‌سازی، نیش مطمئن شده پیاده‌سازی من ۵۰۰ بز بیون bug  
و مشکل باشه. ثانیاً تعداد نست‌های باید از یک حدی بیشتر شه. یه روش جد کردن این هست‌له  
بینیم نست‌های ما حین خط از لد رو نست‌ی لد:  
روش ۱: بودجه‌مون تقویت شه و بجور باشیم دیله کار نست  
نوشن رو تقویت لیم

### جلسه (۹)

\* در پیان ترم مباحث می‌امیم خلف‌نی شه  
ولی تمرکز روی مباحث پیان ترم

refactoring → بازنویسی  
pair worker → افراد مختلفی باهم همکاری شوند و  
روی نست‌های مختلف کاری کند => الله کسی از یتم خارج شه، هزینه اش کمتره چون  
همه افراد روی نست‌های مختلف یتم سلط دارند.

برنامه نویس‌ها باید به نسبت افراد قوی‌تری

باشد که بلند روی نست‌های مختلف بروزه کار کنند.

\* از معاویه ← pair workers ممکنه دتفه باهم کنار نیایند!

Agile project کار بروزه در زمان مناسب انجام شه و تحویل داده شه.

management با بودجه مشخص شده کار کنند

سامان دهی و مدیریت افراد یم

(کی کارها انجام شه و کی چه کاری رو انجام بده) ← واسه Agile نی شم!

Validation

پیش

فرق bug, error

Serum → ① تعیین اهداف کلی و

مشخص لردن معناری افزار  $\rightarrow$  micro service<sup>s</sup> مدل و ... ( ارتباط اجزا

چه طور باشند)

② Sprint cycles

↓ دوی سرعت

یه سری از نیازمندی ها

③ closure phase

فاز پیش

( تولید مستندات و ... )

انتخابی شد و پیاده سازی

و تستی شد

scrum story هر چندی  
است.

Sprint cycles \* معمولاً ۴ تا ۶ هفته طولی شون.

← لیست کارهای مونده ← product backlog

① جدا کردن تم از مسائل جانشی  $\rightarrow$  کسی که هدایت تم را بر عهده دارد  $\rightarrow$  Scrum master

و متعملز کردن افراد تم روی هدف ② خلاصات هفتگی  $\rightarrow$  برنامه نویس ها می گنند کار کردن و

کار برگزار می کنند چی کار قراره بلند و مشطاشون مناسب برای تم های کوچک و متوسط Scrum

چیزی را آن به منابع ملی یا سخت افزاری سینه نیاز دارند.

\* جلسات هر روز برگزار شد و مدتیش ۱۵ دقیقه باشد (15 min)

الله سریع چیز باید بیشتر کیش شد، فقط افزاد لازم

دفرهم جمع شن و یه جلسه دیله تسلیل بین.

نهایی اعضای تم بی فرم

20 min 15 min

کل بیرونی چه طور دره جلویی را

عادت های خوب از بین نزوند.

و در جریان کل بیرونی قرار یی لیزند

← سرقت آمن

\* الله کسی در جلسه حضوری شرکت نکرد، با skype

با هاشم در تماس باشیم

\* بین نفسمی واحد تلیه تلیم و همواره به دنبال ازتعای روش ها باشیم

\* جلسه  $\leftarrow$  استاده بیرونی شد

صحبت رو در رو  $\leftarrow$  فرم ا شباهات زودتر اجام می شد. سنت  $\leftarrow$  وقتی یک نفس  
 (متلاً ارتباط حمرو به حمرو باعث می شد) مستندات کار یکی دیگر رو  
 خواند و ا شباهت فرمید، وقتی باهم رو در رو درباره اش حرف بزند، کمتر  
 از این ا شباهات پیش می آید.)

## جلسه (۱)

- یک نفر یک نظرسنجی راجع به big Data داد.  
 - بعدش راجع به پروره و تشخیص نیازمندی ها در کلاس فعالیت Smily face را اجام دادیم.  
 لئے تعیین scrum master، ساختن prototype، نشان دادن prototype و  
 بعد ساختن tester، جلسات چند دفعه ای، ...

## جلسه (۲)

Scaling out & scaling up : صفحه اسلاید فعلی ۲

استفاده از Agile در یک سازمان بزرگ استفاده از توسعه نرم افزار بزرگ در Agile  
 (عملکرده نرم افزار کلک باش)

## شروع فعلی ۴

برنده سی نیازمندی ها پروسه ای که در آن نیازمندی ها به طور خلاصه متخصص می شود

مستندات نیازمندیها : خرچی

نرم افزار requirement: ویژگی های نرم افزار کارهایی که قراره ایجام بده  
 ی تونه سطح بالا باش، ی تونه دقیق و بیز باش  
 یا مناقصه ؟

user requirement

۱ استفاده طی  $\leftarrow$  ۱ به دید طی برای به مزایی دلستن  
 « جنی سطح بالا بیان می شد » a bid for construct

Subject

Date

نیاید بلند

functional requirement  $\rightarrow$  سیستم چی کار باید بلند و چی کار  $\rightarrow$  ورودی ها چی هستن -

non-functional "  $\rightarrow$  عملکرد کلی سیستم - درجه شرافتی  $\rightarrow$  به چی زبانی کاربر کار کنند  $\rightarrow$  مثلاً  $\rightarrow$  با نرم افزار

چون برای مشتری مخصوصه مبلغ سرعت و امنیت اش چه خوبی باشد  $\rightarrow$  response time

چون استانداردهایی روزگاری کنند  $\rightarrow$  reliability

حتی نیازمندی روزگاری  $\rightarrow$  storage requirement

Domain requirement  $\rightarrow$  اصطلاحات و ویرگی هایی که برای کاربر مخصوصه وی توجه دهنده باری خوب طوری باشد

توسعه دهنده ممکن عوایض  $\rightarrow$  یک خودی

Functional requirement  $\leftarrow$  سه مثال مثلاً  $\rightarrow$  سطح بالا  $\rightarrow$  سطح پایا  $\rightarrow$  آسیابه رو توونه !

مشخص نشدن دقیق نیازمندی ها  $\rightarrow$  هر کسی هر طور به نهش می رسید  $\rightarrow$  تفسیری کنند (کاربر یا توسعه دهنده)

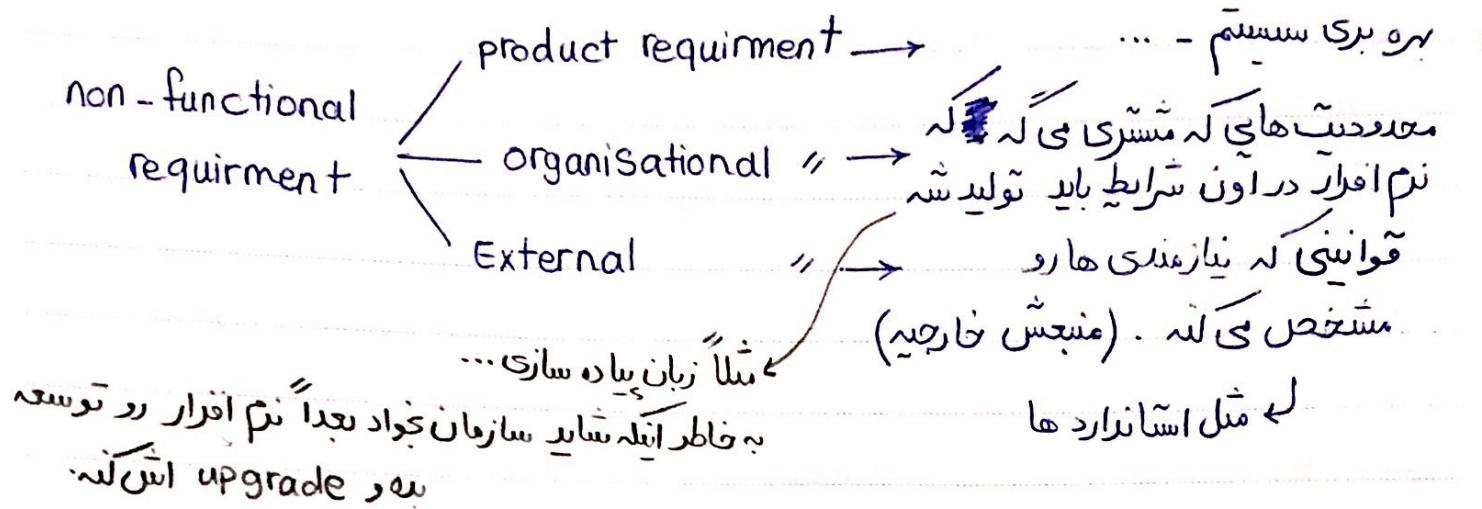
نیازمندی ها با هم همخوانی  $\rightarrow$  consistent

راسانه باشند و همگر رونق نقض نکنند.

\* ویرگی های non functional ممکنه حیاتی تراز ویرگی های functional باشند  $\rightarrow$  مثلاً: محاسبه زاویه مزد آمدن در یه هفتماً نباید خیلی طولانی شود ...

\* بعضی از ویرگی های non-functional ممکن خودش یه سری نیازمندی های functional دیلیه ایجاد کنند  $\rightarrow$  مثلاً: می خواهیم (سطح های مختلف دسترسی به یه سایت) ایجاد کنیم (برای بالا بردن امنیت)

non functional  $\leftarrow$  functional



- \* بعضی وقت ها اهداف و نيازمندي ها را هم آميشته مي شوند
- \* تعیین دقیق ویرایه های non-functional می توانه سخت باشه با این حال باید مشخص شوند

\* اين له حین درخواست رو می تونه تو به زمان جواب بره با response time فرق داره . نهی شده از روی اولی دعوی رو به دست آورد! چون مثنه درخواست ها مواری پيش بروند و آنلر حافظه وجود داشته باشد که درخواست ها مواری پيش بروند .

له ۱۰۰ درخواست در ۱۰۰۰ زمان هر درخواست ۵ سه ثانیه زمان هر درخواست ۵۰۰ می باشه!

و منابع

(۲۵ اسلاید)

جلسه (۱۲)

چه تغییراتی لرده

preface: ... در ورژن کي مختلف → پيش لغتار

introduction: اين سسitem چرا نياز به وجود بياه →

چه کاري به طور کلي مي لنه و با خود

نم افرازها يا سسitem هاي دليله اي

كار مي لنه

system architecture:

به طور سطح بالا از جم نوع معماري

استفاده شده optional هست

Glossary →

اصطلاحات استفاده شده باید

می تونه لغته شه

PAPCO

تعريف و توضیح داده شوند

system requirement specifications

Subject

Date

requirement → what to do

design → how to do

جدلناپذیراند

و روی هم  
نایس لدارند

Natural language → کاربر مسْتَفْعِمَيْ تَوْنَه ازش - فرمش آسونه رساست  
اسْتَفَادَه کَنَه

computer jargon → اصطلاحات

پیجینی کامپیوْتری

elicitation: استخراج

negotiation: مناظر

پا آخر ص ۱۵

requirement discovery: الگاف نیازمندی ها

system requirement: نیازمندی های سیستم

Stakeholders: ذی تعان درحال  
تیمارستان یک مرضی را زیر نظر بگیرد -

برستارهای که باید با سیاران سروکار نهاید و پرسنل درمان را پیاده سازی می کنند.  
سیم سپیسیانی برای بضب و تهیاری نرم افزار receptionist -  
سیارهای را اضافه کند و با آن کار کند. (IT)

... ← ص ۵۵ health care managers

سوالات از قبل نویته → (سپاه)  
closed مصاحبه باز و مشخص است.

براساس هر گفت و گو →  
جدا کانه سوالاتی که پیش بی آید پرسیده می شود.

\* به طور کلی موقع مصاحبه: پیش فرض هارا از ذهن گون پاک نماید.  
بله از دید صحت عذری ادامه پیدا نماید. تا از داخلش معلومات به دست آید ← بخورد مناسب با منزه

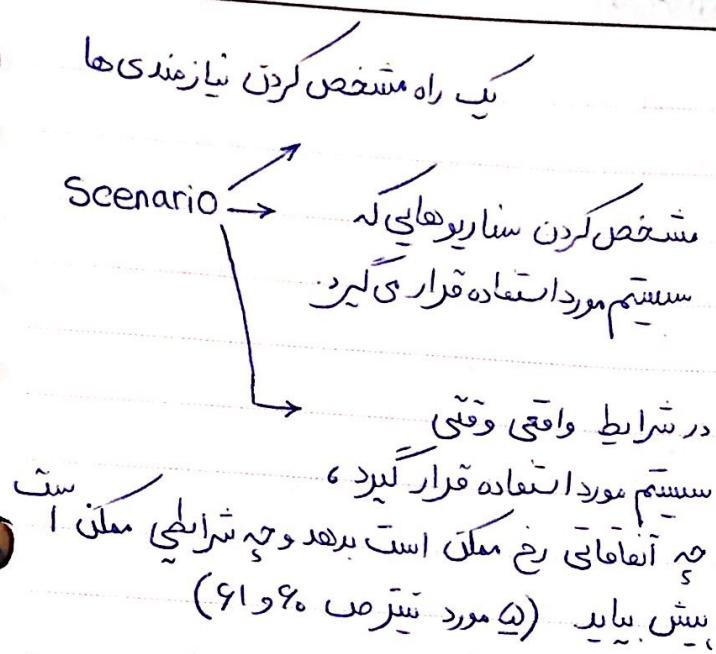
\* در عمل مصاحبه مخلوطی از روش باز و بده است. ← چون همهی یزده کاری ها را قبل از مصاحبه نمی دانسته اید که بخواهید سوال مطلع کنید.

\* مصاحبه باید شامل چهی حوان با سیستم چی کار نماید + چه طوری خواهد با سیستم کار کند.

ل ← چنی روش مناسب نیست ← چون دقیق نیست  
قراره چی کاری را با این نرم افزار ساده نماید

## موقع مسئله سازی

\* راه حل را از مشکل تغییل کنید.



یک دیاگرام براساس → یک روش برای یکی → موارد استفاده سیستم ساریوی UML می شوند.

لئام تعامل ها رسم خواهند شد. ← آنچه خیلی زیاد باشند use case اون ها رو دسته بندی می کنند و هر دسته رو جدا ی کنند. (هر دسته اون هایی که بیشتر به هم مرتبط هستند.)

## جلسه (۱۴)

در use case Diagram را با بیضی شان بی دهند و user دارند (یا سطحی که اطلاعات سهی از اون user بی دهند) authorize: تایید purchase: خرید برای chapter ⑨ ← use case

لئاب باید خوانده شود. (۵-۶ صفحه است)

برای کا بقیه کی قسمت های درس اسلاید ها کافیه

requirement

اطمینان از صحیت نیازمندی ها : Validation

PAPCO

با توجه به اینله آنر نیازمندی اسیده عوینده شه هزینه‌ی زیادی برای تصحیح‌ش نیاز است، از همان اینا با ہیسلیری ی لست requirement validation

هم (consistency)

پرداخت شدی یا آنلاین → نیازمندی‌ها تضاد نداشته باشد با ①

استخراج همه نیازمندی‌ها رو در tap30 ②

نیازمندی‌ها اجام داده باشیم و چیزی از ملم نیغناه باشے (Completeness)

نیازمندی قابلیت ③

اطمینان داره باخیر

یک نیازمندی واقع‌الاینه است ④

یا خیر (با توجه به

کنکوئری فعلی و بودجه‌ی موجود و ...)

۵) آیا یک نیازمندی قابل اندازه‌گیری (Verifiability)

و ارزیابی هست؟ ← باید سبب شوند

تسی براش ISA fuse مثلاً سرعت نرم افزار ...

تعریف کرد.

راهکار ← بازسینی نیازمندی‌ها (هر چند وقت یلبار)

اسفراوه شه (که بینن یه نیازمندی قابل پیاده‌سازیه یانه) prototype

ساخته شه Test case

در ابتدایی ترین زمان ممکن باید فرم مون از نیازمندی‌ها کامل شه! ← کاربرو ۶) تیم برنامه نویس

باید با ہم فرم‌شون رو کامل

کند و با ہم در ارتباط

باشند.

-۷-

جن مخصوص کردن نیازمندی‌ها یا زمان توسعه نرم افزار حس

مدیریت تغییرات نیازمندی‌ها

- دلیل به وجود آمدن changing requirements
- آنکه اسنادهای از نرم افزار با ~~تامین شده~~ بودجه تامین شده بیایند
  - سیستم تعداد کاربر زیادی داشته باشد و اختلاف نظر روی نیازمندی‌های سیستم داشته باشد.

نیازمندی‌ها قابل ارجاع باشند و هر کدام شخصاً شماره داشته باشد  $\Leftrightarrow$  ارتباط و تأثیرپذیری نیازمندی‌ها

در تامین مستندات هم مفیده

مشخصه ای شده changing requirement واضع تری شده و ...

راهکارهای

۷۷  $\leftarrow$  requirement management

Tools

- 1) در پروسه تغییر نیازمندی‌ها اول باید بررسی کنیم که نیازی به تغییر هست یا مستحب است که نیاز به تغییر در نیازمندی باشد یا نه.  $\leftarrow$  2) چه تغییراتی در نیازمندی‌های دلیل لازمه و حدود اون‌ها رو تکت تأثیری ذاره  $\leftarrow$  برآمد هزینه‌اش
- 3) اعمال تغییرات  $\leftarrow$  اصلی هزینه‌اش، این تغییر قابل قبوله یا نه  $\leftarrow$  با توجه به

هزینه‌اش، این تغییر دهنده بوده توی محیطی که توسعه دهنده بوده توی محیطی که به نرم افزار مانیزه و مستندات و ... کسب کنند.

اصحاحیه  $\rightarrow$  آشنایی نیازمندی‌ها  $\rightarrow$  ساریو و use case  $\leftarrow$  ethnography  $\leftarrow$  خود

اهمیت ارزیابی نیازمندی‌ها  $\leftarrow$  key points

P4PCO

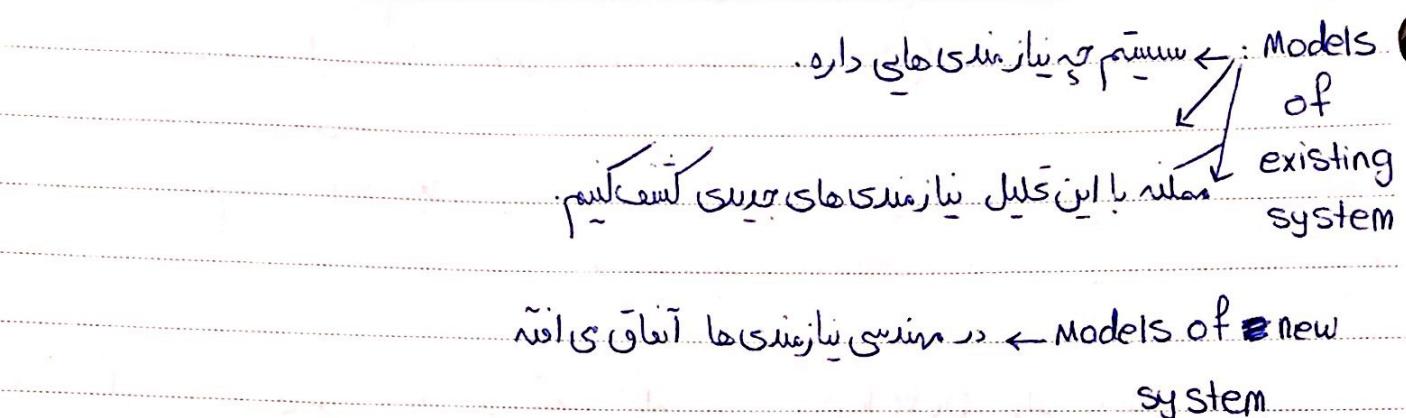
جلسه بعد: روش‌های مدل‌لردن و دیگر ام‌های شون و ...

امتحان تا آخر این جلسه

۸۰ را آخرین اسالای

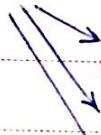
System modeling → بررسی سیستم از دیدگاه‌های مختلف به طور انتزاعی برای اینکه چیزی نی فراهم در این مرحله بیچیدگی وارد سیستم نشود

- منزیت: ۱- تک در فهم نیازمندی‌ها  
۲- ابزار بیار مناسب برای برقراری ارتباط بامسّری بین Developer‌ها



دیدگاه:

system perspectives → external



در رویداد‌های مختلف چه تعبیراتی → رفتاری:

در سیستم رخداده:

چه ممکنی‌هایی در یک فرایند سیستم

وجود داره Diagram

• use case diagram → (کلی و سطح بالا) نشان دهنده تأثیرات سیستم →  
بامحیط

• sequence diagram → جزئی تراز Use case diagram

• class diagram → کلاس‌ها و ارتباطات آن‌ها را در برنامه مان مشخص می‌کنند

P4P رحلونه راهنمایی نهاد همان کار را انجام می‌دهد.

• state Diagram → چه تعبیراتی که سیستم داشته باشد

وچی اتفاقی‌هایی در سیستم داشته باشند

Scanned by CamScanner

\* آگر ار مدل ها به عنوان پایه کار استفاده کنیم، هم کامل باید باشند و هم دقیق

Context models → سیستم ها به شکل  → نوع تعامل ها × → تعیین سیستم ها فقط تأثیری که تعامل شون با سیستم ها مشخص بیشتر افرار مارکیط داره

و محدوده سیستم مارو مشخص بیکند و اینله با چیزهایی که خواهد تعامل داشته باشند (system boundary)

تأثیر زیادی روی بیازندی ها که نرم افزار داره (اینله حقدار از بیازندی ها و خودمون بی خواهیم بیاده کنیم)

\* مسائل سیاسی بی تونه تأثیر لدار باشند. اینله بی دکلیر هستند خارج از سرکت  در پیدا کنند ... باین Developer ها اختلاف تپریش بیاد

process model → مثال app تالکی  شکل

یا فرمتی از اون

\* توصیف فرایند در

UML activity diagram

صلیل  activity diag  ram هست

(بعد از خط دویس موازی با هم پیش بروند)

اطلاع به نیک از آنها  next of kin

Social care  یک نیاد دولتی که

مراقبت با شه حقیق

سفار روانی  نیش

System  
MHC-PMS

سیستم های  
دیلیت نه تعامل  
داشته اند با سیستم های

● → شروع  
○ → پایان  
□ → activity

↑ → نهایی جریان کار

— → فلش هایی که بین join

واردی شوند باید اول همی

تمام شوند بعد فلش های خارج شده از آن همراه عمل نند.