

System modeling → بررسی سیستم از دیدگاههای

مختلف به طور ابزاری برای اینکه

چنی نی خواهیم در این مرحله پیجیدگی وارد سیستم کنیم

مزیت: ۱- تک در فهم نیازمندی ها

۲- ابزار بیار مناسب برای برقراری ارتباط با مستری یا بین

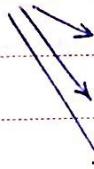
سیستم جو نیازمندی های دارد. Models of existing system

مصلحت با این کلیل نیازمندی های جزئی کسب کنیم.

در منسی نیازمندی ها آتفاقی افته Models of new system

perspective: دیدگاه

system perspectives → external



در رویداد های مختلف جو تبعیراتی → رفتاری: در سیستم خی ده

جو فعالیتی های در یک فرایند سیستم Activity

و وجود داره Diagram

● use case diagram → شان دهندهی تعامل سیستم

با محیط

● sequence diagram →

جزئی تراز

use case diagram

● class Diagram →

کلاس ها و ارتباطین آن ها را دار

P4P و جلوهه باهش نهاریم همان کار را انجام می دهد.

برنامه مان مشخصی کنیم

● state Diagram → قسمی آتفاق هایی در سیستم خی ده، جو تبعیراتی کنن سیستم

Subject  
Date

\* آنرا در مدل‌های بیانی کار استفاده کنیم، هم کامل باید باشد و هم دقیق

و محدوده سیستم مارکت شخصی کند و اینکه با چه چیزهایی  
ی حوار تعامل داشته باشد (system boundary)

## وحوش سیسم های با

تایپر زیادی روی سیارمندی‌های نرم افزار داره.  
(اینله حقیر از نیازمندی‌هارو حفظمن. هی خواهم پیاده کشم.)

\* مسائل سیاسی ی تونه تایپرلار باشد. اینله یه Developer خود یه پیاده سازی به خارج از سرکت ~~د~~ درز پیدا کنه ... یا سن Developer ها اختلاف تپریش بیاد.

مثل app تالیی → فرایند سب و کار : مثال  
ما قسمتی از اون

## \* توصیف فرایند در activity diagram

time ~~activity~~ diag <sup>ram</sup> 

(بعد از خط دو یعنی موازی با هم یعنی روند) fork

اطلاع به کجاز اقوام → inform next of kin

Social care  $\rightarrow$  نہاد دعالتی کی

## مِرَاقِدُ بَاسَةِ حَقَّقَ

San Francisco



۲۸

## جلسه (۱۹) خارج

interaction models → تعامل های درونی سیستم ② تعامل سیستم با محیط ①

1 use case diagram

چه مسیرهای ممکن رخ بده

و نرم افزارها با محیط

مطابق باشند ...

2 sequence diagram جزئیات پیش راه رودار توایی: نمودار توایی

Actor: بازیگر people  
other systems (\*)

use case شروع کنندی → یک use case  
case Actor  
است.

use case

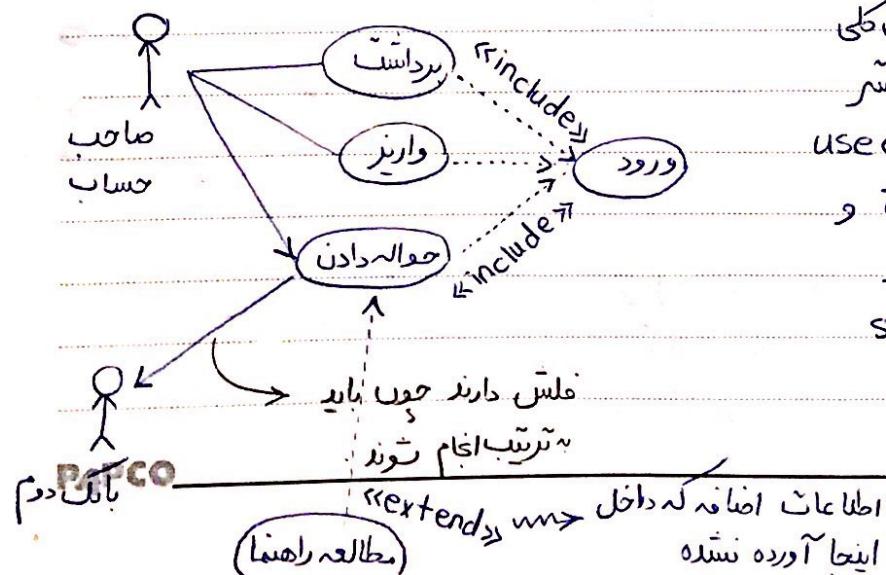
موارد کلی استفاده و هدف های کلی روشن شوند

use case هر

و Text با

و Table

sequence diagram



Subject  
Date

جزئی تر برای یک Actor خاص  $\rightarrow$  ص ۲۱  
use case

بعضی وقت‌ها  Actor  $\rightarrow$  سیستم خودرو زنگ به عنوان system نوشته شده است.  Case use diagram نیست تری تلسیم: ی شه منظور سیستم‌های خارج از نظر افزار ماست.

\* Sequence diagram →  مونitoring PRS: patient Record system

P: Patientinfo  
سی دی کالس → jl object بی  
class ~

لهم نوّتْه شَهْ يَوْم  
بِسَيْمَ خَارِجِيَّ لَهْ بِاسَيْمَ مَا دَرَ  
تَعْمَلْ يَوْمَه

ساختاری میانی سس متری زمان هنر از هر جا شروع شد. Alt: Alternation نتیجه رو با استون زمان هنر از هر جا شروع شد. این فعالیت از اون جا شروع می شود...

## • structural models

زمان دلیر شرایطی داره. بعدن شیء در آن ساختار طراحی سیستم - فعالیت

کلاس هار class diagram :

یک نمونه

جایی static چه اجزایی باهم در ارتباط اند.

class diagram: کلاس‌های

## اریاطس آن‌ها

در مراحل اولیه اخراج،  $\rightarrow$  یک تعریف کلی  $\rightarrow$  مفاهیم کلی روییت از دسته ای از اشیاء به عنوان کلاس تعریف کرد.

یک یا سیسرا : ۱۰۰ \*  
پین بلڈ نا ۴ نفر : ۱۰۰

\* صفات و وقارها رومی شه در class diagram مشخص کرد.

## Generalization

Super class  $\rightarrow \dots \rightarrow$  subclass

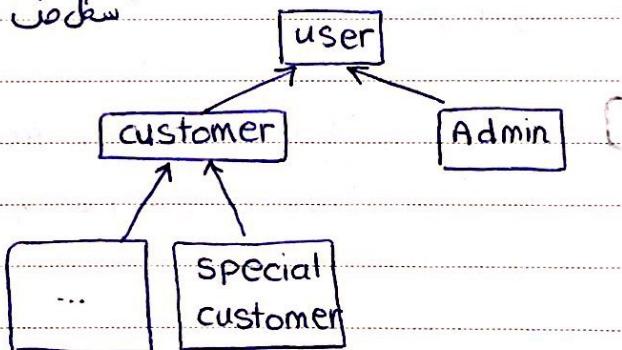
## و محرودی

اسلوب تغییر  $\rightarrow$  الله يه کلاسی جو لا تغیر کنه subclass فواید  
مشخص ہی شہ هاس هم باید تغیر کند اما superclass هاس  
لازم نیست تغیر کند

class → اسناده از وراثت (Inheritance)

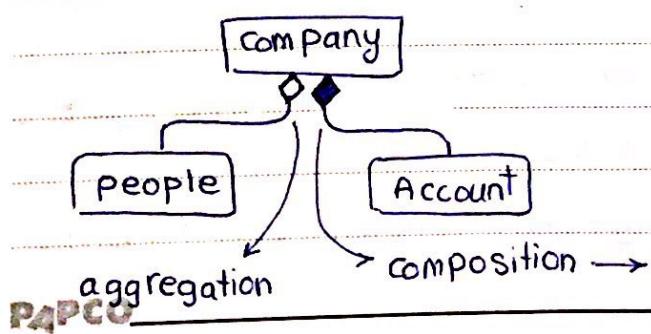
\* ملنه حلاس فزند ویری ای  
داسته باشه نه در طاس بدر و چود ندارد

٣٤



وقتی چند کلاس دارای یک سری  
ویرگی های بسان هستند، وارث  
یک کلاس مشترک به نام کلاس پدر  
خواهند شد.

اگر رابطہ کی کلاس ہو جوی ہے تو اس کے اسی کی آن کلاس  $\rightarrow$  aggregation models  
 اگر کلاس از بین رود، باقی ہے تو  $\rightarrow$  جمع



الله طاس company ازین بره، حساب های این شرکت هم ازین خواهد (زیست)

Subject \_\_\_\_\_  
Date \_\_\_\_\_

② با یک محرك داخلی یا خارجی، یه آهانی داخل سیستم می‌افتد.

Data - driven model:

Data & event

در زمانی کلیل سیاری از سیستم های کسب و کار  $\rightarrow$   
نیازمندی ها  $\rightarrow$  سیستم های مدیریت داده هستند.  
(یک سری داده های ورودی، پردازش، یک سری خروجی)  
خوبی های مولده بصری  
و نشون می‌ده

(Real-time)

event  $\rightarrow$  ② برای سیستم های لحظه ای مناسب

ص ۴۰ مناسب برای ... ② «تالید بر فعالیت ها»

ص ۴۱  $\leftarrow$  sequence diagram  $\leftarrow$  کارهای سلسله وار  $\leftarrow$  نمایش پردازش داده  
«تالید بر طلاس ها»

نمایش

ص ۴۲ استفاده از Activity diagram برای محیط کسب و کار

ص ۴۳  $\leftarrow$  استفاده از «بری شان دادن رفتار های لد

در سیستم ما آنهاقی افتاد

تعداد state های سیستم را بسته به یک event (محرك)

داخلی یا خارجی یه طریق عکس العمل نشون می‌ده

باشه تا بینه با تو به

اون ها و صفت سیستم رو بررسی کرد.

با هر event سیستم از یه حالت به یه حالت دیگه می‌رود

در همون حالت با یه مونه

شکل ص ۴۴ state diagram  $\leftarrow$  یک مارکو ویو

\* الین state پیچیده باشد و state های لوچر شلسن می شوند ← تال در صفحه ۴۷  
(تعریف سطح های مختلف از نمودار) در مورد پیچیده بودن

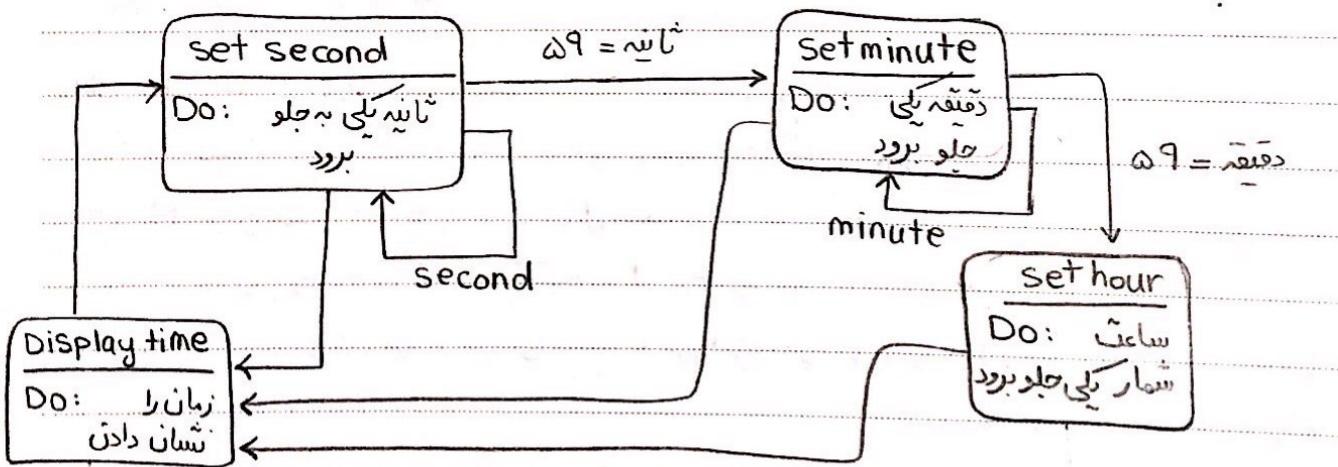
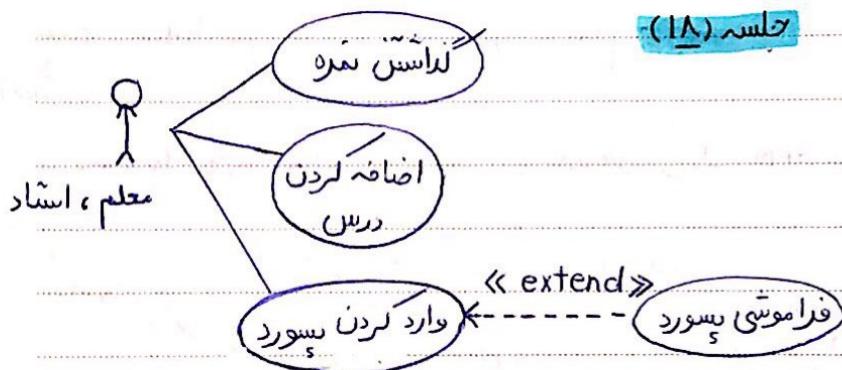
"operation" state

\*نام state ہالا یہ تعریف شوند و توضیح دادہ شوند ← مثل ص ۴۶

event →  un -

~~نحوی~~  $\rightarrow$  باستطیعه های کوئن منجز

مُسْرِصٌ ۖ اسْلَابٌ  
(model-driven)



\* دلیل اینله در state diagram گلاب ۲ استیت بلسان داریم، این است

Subject \_\_\_\_\_  
Date \_\_\_\_\_

## Features of executable UML →

در کلاس ها اسیا شون درجه و صنعتی می توانند تغیر نمایند و حجم تغییراتی در طول عمر شون ملته به و تبدیل می شوند.

مورد آخر ص ۵۲ (state model) ←

## chp6 : Architectural Design

مشخص کردن خود شون و رابطه شون → ساختار طی سیستم compontant & Subsystems → برای برآورده کردن نیازمندی ها

← سیلیال های لسی از کدام جزو به کدام جزو می ره (آنفال داده یا دستورات) (ص ۳)

مستقبل های کمربن سی نشان می دهد خودش یک subsystem است.

معاری ارتباط اجزای یک نرم افزار (دید بروج) →

دید بروج ← سیستم های بزرگ ← کاربرد نرم افزار های مختلف برای رسیدن به یک هدف ← کامپانی ← اجزاء خود شون سیستم اند.

## stakeholder communication

هزایا: ۱) لک به ذی نفعان ← مدل های برای بیت و لغت و لو system analysis

۲) دیکی و قابلیت های دارد بیاده سازی ویرایی های non-functional (یعنی شه

۳) بررسی سیمه که نرم افزارها ممکن استفاده در سطح کامپانی نرم افزار های سیمه هیں وجود داره یانه ← استفاده از نرم افزار های از پیش موجود

۴) شرکت هایی که برای کاربرهای متسابه ، نرم افزار های تولیدی نسخه (مثل سخت افزار که خط تولید دارد و اینبو تولید

(الله فراره ملکی برای بیت و لغت و لو باشد) حی الله

۵) نوع رابطه و ویرایی های اجزاء ← مشخص نیته در طراحی معماری و خودش باعث سرگردانی شه مگر اینکه قرار باشد پاییزی (خصوصاً برای شتری که از جملی چیزها سر در نمی آورد) مسندسازی فرار بلایه.

## Architectural design decisions:

طراحی معنی: خلاصه + نیازهای تجربی  
ویژگی های بازیه لیستی  
نحوه ساخته لیست

۱۱) سیستم ما چطور قراره

پرکاره شه روی core ها یا پردازندگان مختلف؟ → جواش تا شر زیادی بی لذاره دوی  
طراحی ← تغییر در ارتباط اجزای موجود

۱۲) چه سک معنی مناسب برای سیستم ما؟  
ب وجود ماد  
ی سه از جنگ آسا سک  
استفاده کرد

۱۳) سیستم به چه اجزایی ساخته شد

و در یک حوزه

\* سیستم های شبیه هم معمولاً

۱۴) چه اسناریوی برای لست اجزای در نظر  
آورده شد:

ر تمل قرارگیری موتور در مدل های تفاوت

ماشین ها . ماشین مسابقه محل موتور در وسط و ...)

سریع های مختلف هم

هیچ طور پیاده سازی کی لست

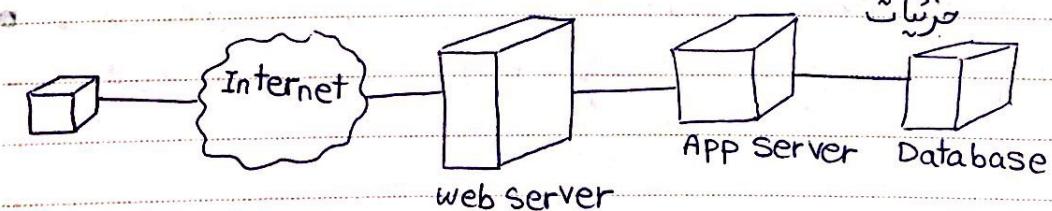
سوال ها در ص ۱۲ اسلامی ← مورد ۱ و ۲ و ۳ و ۴

Application product lines

\* بعضی سری های هسته ای نرم افزار را استخراج می کند و بنایه جزئیات نیازمندی های هر

دسته از مشتری ها، ارن را جداگانه پیاده سازی کند

جزئیات



منتظر گرفتن درخواست های HTTP

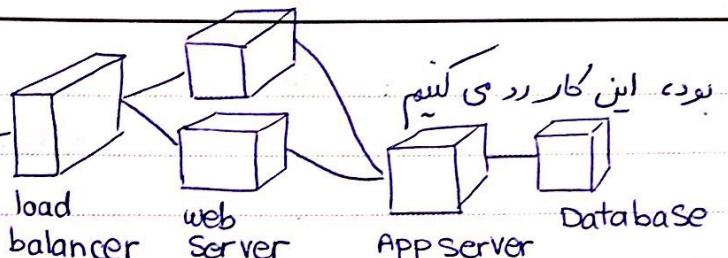
App Server درخواست ها را به است

ی فرسته و Database هم از App Server استفاده می کند

ال سرعت یافتن بوده، اینا برای کی لیست طوطا در کجا ایجاد شده است ← مثلاً اگر web طوطا  
Server

\* با وجود fault در سیستم، down نشوند

webserver



حیلی روی معادی طراحی می‌گذاره

عملکرد

در سیستم‌های که امنیت مردم، از لایه بندی استفاده می‌کند.  
نیز هر درخواستی که ارسال شده در هر لایه‌ی توان از یک سری نظر امنیتی را بچک کرد. ← مثال:



این ترتیب کس → همین‌چیز از نظر امنیت از خارج سیستم

اینی ← یک bug ایجاد هزینه کند. (این از داخل سیستم - امنیت از خارج سیستم)

حین‌چیز یک کار مشابه روتکار لسد + مکانیزم‌هایی برای تحمل خطا

درستی

الله نگیراری هم، به اجزای نیز تقسیم می‌کند و اجزای کوچک راحت‌تر

قابل تعریض و تعمیر و نگیراری اند.

نگیراری

objects & classes، key abstraction

در زمان اجرا عده طراز اجزا باهم در ارتباط اند و

پرنسس‌ها در یک مرحله‌ی اجرا هستند؟

صف‌ها اسلاید

صف ۱۶ ← یک دیگاه معروف: ۱ + ۴

دیگاه ترکیب ← ؟

کوه‌های استفاده از فنریلی ← چه ساخت افزارهایی داریم و سیستم را در هر نیم افزارها حفظ نمایی آن‌ها سوار شده‌اند که مورد قبل از زیبایی ای لند.

RDCSO

## Design patterns ≠ Architectural pattern

برای این مشکل با این ترتیب  
تجربه ثابت کرده فلان روش و الگوریتم مناسب نیست.  
لکن برای حل این مسئله مناسب نباشد.

الگوهای معمولی بیان می‌کنند  
که خوب است از شون استفاده شد و کی خوب نیست.

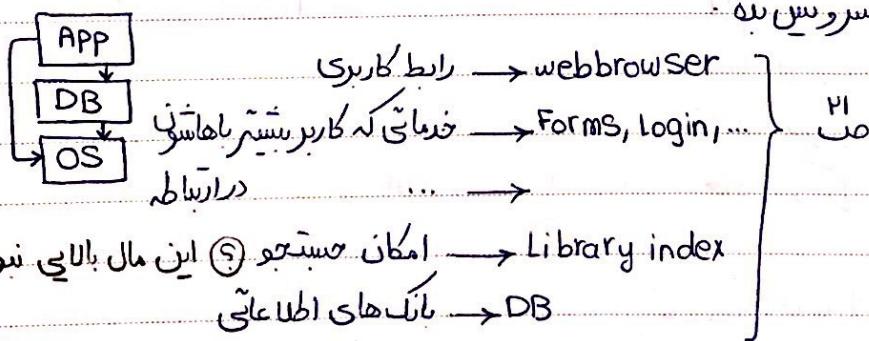
الگوهای روشنایی می‌دهند >  
< تراویل وار

الگوی لایایی ← اجزای سیستم‌ها هر کدام به اجزای دیگر خود را معرفه کند و باهم رابطه دارند.  
هر لایایی از لایایی زیری خود خود را معرفت می‌کند و...

۱ توسعه مرحله‌ای و  
 تقسیم‌کار راحت می‌شود  
۲ بالابردن امیت روی شه بیاده سازی نمود

۳ سطح تغییرات واضح می‌شود و همه قسمت‌های را بیان می‌کند

معایب: هیچ مدلی واقعی که بیان نمایند نداشته باشند از لایایی پیشی سرویس‌گیری و فقط به لایایی سرویس به...



امان حسب‌جویی ۴ این مال بالایی نبود؟  
→ بانک‌های اطلاعاتی  
→ DB

حجم

الگوی انبار (مقرن) ← ۵ قوی‌داده‌ها زیاده استفاده می‌شوند.

بطریکی > ۱ داده را برایم مستود می‌زیم  
۲ هر کلاسی (هر جزئی) داره خودش را دسترسی داشته باشد  
مدت طولایی تله‌داری  
کنیم (بخطایف big data)

و بعود داره

Subject  
Date

## \* گردی

مزیت ← حوزه داده هایم یک جانلبری تولد، هماهنگی و سازگاری داده ها وجود دارد

\* می تونے

\* فرمت نام

از دسترس خارج شد.

## 4.8 client-server

### جلسه بعد لوزیر داریم (همین فصل)

deos  $\frac{1}{2}$

\* شلک هم جزئی از اجزای سیستم client-server است. ← حسّاً باید وجود داشته باشد.

-Controller

\* client-server  $\rightarrow$  Single point of failure  $\rightarrow$  هر سرور کہ از سلسلہ خارج شود،  
کل سیستم دھار مسٹر نہیں۔ (برخلاف Repository)

## \* استفاده در

\* لـ نـسـمـيـلـرـا client

یک حالت خاص از web Service \*

## کلام سروری خواهد داده رد و بدل

معاری client-server است!

لند می تواند یک داده ای نمایند

سُكُل ص ۲۹ ← اون چس مُلزی

مدیریتی کے client و میں ہی ہے۔

اون نیش های دورش هم سرور نیستند

## ناره شباهی هم وجود نداره

تازه شبله‌ای هم وجود نداره که با شکل ص ۲۹ استیاه نشود

حصص اسند ← از ~~نیز~~ لست

\* در پردازش داده زیاد از مش اسپاوه می شو. Pipe & filter

(... Unix و Linux ...)

خروجی یک filter  $\leftarrow$  ورودی PIPE  $\leftarrow$  ...

هم بعد استفاده قرار یافته.

\* کاررویی داده شلسنه‌ی شه به مراحل زیر زیر و در هر مرحله فیلتر انجام می‌شود.

command<sub>1</sub> | command<sub>2</sub> | ... | command<sub>n</sub>

IS-1 | sort

cat input.txt | uniq | count

cat input.txt | count | uniq

\* می‌توانه پردازش به‌طور سری باشد یا

ب‌طور موازی  $\xrightarrow{\text{۳۲}} \leftarrow$  مثال صفت اسلاید

\* فرمیت نباید بی‌جایی باشد.

invoice: رسید

payments: پرداخت‌ها

→ فصل ۶ ناس اینجا خود رونم

(MVC)

→ هدف: جداسازی داده از نمایش و تعامل داده با کاربر

→ داده را نمایه‌داری و مدل‌سازی می‌کند و مدیریت داده

→ داده را نمایش می‌دهد

→ درودی را از کاربری لیده و ارتباط با کاربر را برقراری می‌کند و view و model ارتباط برقرار می‌کند.

\* استفاده در web-based apps: زیاد استفاده می‌شود

class MyClass{

private double age;

public double getAge();

:

}

سایت‌های وب‌یابی که استفاده می‌کنند.

controller  $\xrightarrow{\text{۳۳}} \leftarrow$  اسما

از client ها را دریافت می‌کند و ...

لیست

یک سری معادله‌های عمومی برای دسته‌ای از نرم افزارها استفاده می‌شوند.

۱- برای شروع معادله مون از شون کلک بلکم

۲- عنوان تیک  $\leftarrow$  بررسی هماهنگی checklist

۳- شناختن اجزای و نسیم‌بندی طار  
۴- نهادیم حقدار امکان داره از نرم افزارهای موبعد استفاده محدود نیم  
۵- ابزارک برای صحبت درباره اون دست نرم افزارها ← مفاهیم در اون حوزه  
استفاده از

پردازش داده ← pipe & filter ← میاد و معلومات فرستاده می شه

نحوه‌های از  
ترالس‌ها ← درخواست برای یک معلومات یا  
معلومات در یک طبقه داده

انواع نرم افزار  
پردازش آنفایات ← تغییر flag‌ها و... ← در سیستم‌های امنیتی،  
embedded systems

تعریف یک زبان خاص ← مثال: کامبایرها

متاسب با آنفای رخداده ← interpreter

پردازش‌هایی انجام می ده.  
تفسرها

(۳۸ آخر ص ۳۸ استاید)

جلسه (۲۱)

Transaction  
batch → سبب‌ای از داده  
تجارت الکترونیک، بانکداری، proccesing  
رزو کردن systems

یک سری عملیات سلسله مرتبی = ترالس  
درخواست‌ها عموماً به طور ناهمراه می‌شوند ← مرحله بعدی  
ایجاد

manager ← Transaction ← مرحله نه  
کار جلویی ره و

database ← الگ لازم باشے ← رابط  
تعامل داره: ← لایه کاربری

عمل نمودار "لایه‌ای" جا ← user communication

می‌مونه (شکل ص ۴۲)  
فقط لایه‌ها اتفاقی رسم شده‌اند

Authentication: ایزاز  
هیئت

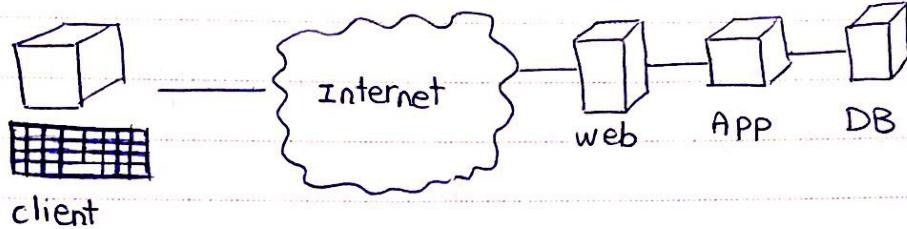
Authorization: کارهایی که مجاز و  
می‌باشد

login (role checking)

P4PCO

حال بسیتیم های مدیریت اطلاعات بر پایه وب هست

ص ۴۶



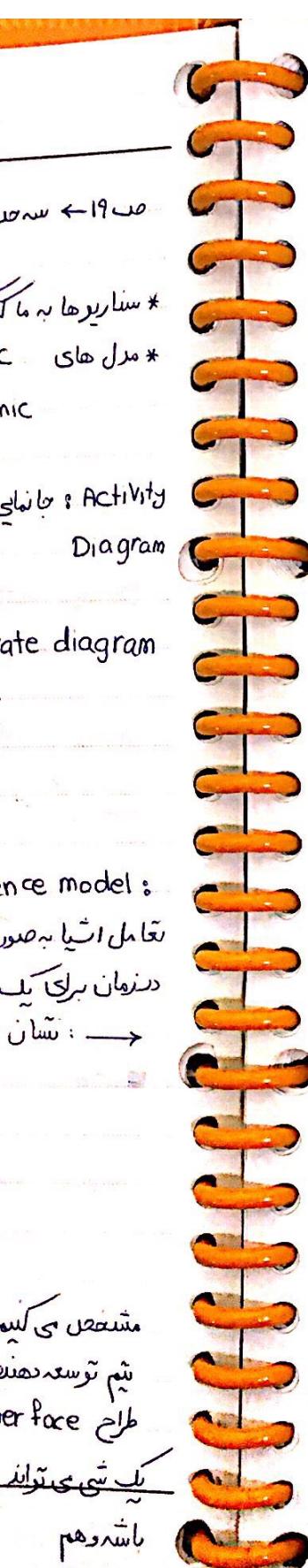
client

ص ۴۸ ← نرم افزار پردازش زبان  
زبان طبیعی (صعبت انسان) یا ساختی (برنامه نویسی و...)

ی شه برای این زبان ها نیز در نظر گرفت ← میباشد Alexa: تبدیل صوت به فرم ← نفعه منظور از درخواست چیزی  
حرجی به → پردازش لنه جا

نوشته sudo code

ادامه (Voice از الله)



Subject \_\_\_\_\_  
Date \_\_\_\_\_

اطمینان!

جلسه (۲۲)

۱۰

\* اجزا در نیس Design، لوگو از اجزا در نیس معماری هستند.

معایب استفاده از نرم افزارهای موجود → مملو م وجود باشد آنها به دروسی نشده باشد  
دیگر ویرگی های مورد توجه ما را در نیاسه باشد و محدود شیم  
تغییرش بدم

نیس، ترتیب کردن و تنظیم کردن کارها  
نرم افزار مناسب با ویرگی های  
سیستم مان (کد زدن نداره)

نوعی  
wordpress → content management system

Droopal → (جزئیات بیشتر و تغییرات)

(wordpress بیشتر سنت به)

حدود زمانی دستیاری → سیستم های در تعامل با  
نرم افزار مارو مشخصی نداشته اند.

modes of use → usecase diagram

یادآوردن کلاس های سیستم → ①

design models → sequence diagram, state diagram (ریزتر)

specify object interfaces → رابط کلاس ها

مغایضه / مغایر کلاس ها و روزنامه های method

به حیث صورت باشد

Users interactions  
systems =  
Actor

reconfigure → تغییر تنظیمات

instrument → کاربا سنسورهای هوشمندی

همی اطلاعات

broadcast

بصورت

۱) مشخص کردن کلاس ها ← یک راه حل

grammatical Approach است ( فعل: کارهای که

۲) هم اشیاء ملموسی وجود دارد. سیستم مان

انگلیسی آنها

نه! ← خوب!

صف ۱۹ ← سه جدول پایین صفحه: **sensor**، **method**، **class diagram**

\* سناریوها به مکمل می‌کنند کلاس‌ها را استخراج کنند.

\* مدل‌های شان دهنده‌ی "ساختار" هستند. ← مثلاً **static**

**sequence diagram** ← سریز روی تعلیمات بین اجرا است. ← مثلاً **Dynamic**

**State diagram**: مثلاً ← **Activity Diagram**؛ جانمایی سیستم‌ها را مشخص می‌کند.

زمانی که یک آتفاق خیالی در **state diagram** شوند **object**‌ها **حکم** می‌کنند ← **state diagram**  
عوض می‌شوند ← در سیستم‌های بی‌حیله معمولاً استفاده می‌شود.

**aggregation model** →

مدل وراثت →

**subsystem models** →

دربیاده سازی مملکه اجزا نیارهم قرار نمی‌زند

**sequence model**:

معامل اشیا به صورت مرحله‌به مرحله در زمان برای یک هدف

→ شان دهنده‌ی تعلیمات با اسیاء سیستم را اجزا چه طور →  
    { به دخواست‌ها پاسخ می‌دهند.

می‌تواند در سطح بالابرای یک شیء مانند

→ **event** برای یک

سیستم

**Interface specification** → مشخص می‌کنیم صورت خارجی کلاس‌هایمان چه تفصیلی دارد.

این توسعه دهنده بر سبای **interface** ای تواند کار را شروع کند

طراح **interface** نماید کاری به دربیاده سازی داشته باشد.

یک شیء می‌توان **interface**‌های متعدد داشته باشد ← مثلاً **ماشین سواری** می‌توانه هم **rideable** و **co**

ماشدهم آما **ماشین** لستی فقط می‌تواند **rideable** باشد.

Subject \_\_\_\_\_  
Date \_\_\_\_\_

۴۸  استفاده ای کنیم. class diagram \* UML

## UML →

## تائید اسلامی مخصوص

## جلبہ بعد: الگوہای طراحی و سُلطاتِ پیادہ سازی

توسعه دهنده ≠ برنامه توییس

二

(۲۴) مل

برای یک مشکل یک الگوی کلی ارائه شده → الگوهای طراحتی  
له حل‌هایی که با تجربه ثابت شده مناسب هست. (انتشار معلومات برای همین همه)

محول از object oriented گنیم ① مسئل را تعیین کنیم ② راه حل به طور کلی توضیح داده شود ③ تفاودهای تابعیتی باشد ④ تغییرات مخلص در هر پردازه ممکن باشد ⑤ سخ مسئل

### ۳۰ اسم زاری اللوک

بیان محدودیت ها در استفاده از هر لغویا (F) 

## ۱۰۰ اثرات استفاده از

## ■ Observer Pattern

نمایش‌های مختلفی از یک داده که نیاز هست به روز رسانی هم

گاهی داده بینه که جدول، نمودار

→ مَسْنَى دریک loop

حل لئيمه تعسر رض داده بانه

حاب: مملوک از لحاظ

## new new performance

✓ → والدارکرن خبردادن → راه دوم

## رخداد تغییر به خود شیع

لارم نیست subject بارز

## بیانیه سازی

شون میاد پاس coupling ←

Attach → عضو شدن → پاسن میاد → در صرافی Performance  
 Detach → از عضویت درآمدن → مکانی getstate() → صرافی شفیده میان  
 است لازم نباشد تغییر کند  
 همه شان

■ Facade Pattern: یونان کرن بینظمی برای کار کردن راحت تر کاربر  
 نما

■ Iterator Pattern: ② در تیاری در طور توالی دار  
 بداده ها

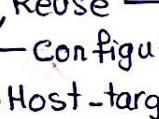
■ Decorator Pattern:

یک type را بر حسب نیاز به object ها درست → راه  
 اضافه می کنیم (مثال پنزا و مسیری که فلکل دلم  
 بخواهد : )

(keep track: دنبال کرن)

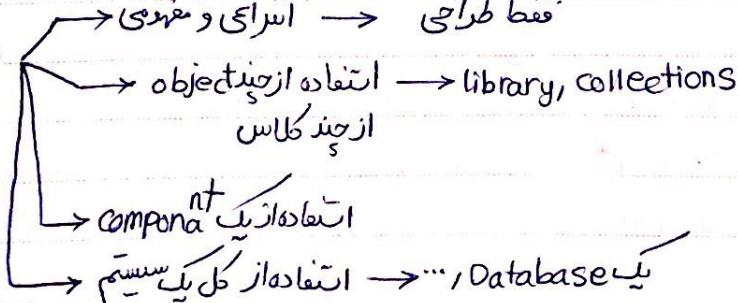
وقت

کاهش هزینه و Reuse →

\* Implementation Issues  سخن هایی که باهم → سخن هایی که باهم → هم خواهی طرند، باهم استفاده شوند.

stratch: از ۰ تا ۱۰۰

Reuse Levels



Subject \_\_\_\_\_  
Date \_\_\_\_\_

بعداً به روز نشانیم ④  
نم افزاری که از آن استفاده می‌کنیم ① → مسلطات  
هزینه ای استفاده از شون ④  
خوب document نشده باشد  
و وقت زیادی از مون بره تا بخواهیم متوجهش بشیم  
نیادی است (licence)

همانگاه کردن شون مملکت ④  
تضمین کردن شون مملکت ④  
لے تغیر دادن بر حسب نیاز پروره ما  
پروره مون  
با فیضی  
خیلی لذتی باید.

Configuration Management → ۴۲ کار

Version management → گروه ها و افراد مختلف خط های که را  
تغیر دهند

keep track of different versions → بررسی در زن های مختلف کار → git

System integration →

Problem tracking → تراش اینله چه کسی چه چیزی ممکن و بعد از که

Host - target development →

platform → سخت افزار → CPU - R , X86 , ...  
windows → نم افزار

editing tools → UML , XML برای

project support tools → مدیریتی  
فی

IDE → Integrated  
Development  
Environments

یک سی امکانات پلیارچی  
در اختیار تردد توسعه باشد

محصول یک زبان

باشد

فقط " " " باشد

high level

of  
communications

PAPCO

High Availability → نسخه های مختلفی از اجرا در یک لیست

اجرا را در یک سیستم  
یا سیستم های تردیل هم

open Source Development → معروف ترین Linux

مجانی ≠

Java, Apache webserver, MySQL

دسری بزیری → مسلطات امنیتی داشته باشد ① → چرا استفاده کنیم؟  
های ما بعضی های لئے → مسائل حقوقی ④ → ...  
جی خواهی از

با وجود اینترنت هی تون  
در جاهای مختلف دنیا  
باهم در ارتباط باشند.

نرم افزار openS  
ما استفاده کنی، نرم افزار توجه  
باید openS باشد.

مکله اجباره نهند  
 تمام یعنی های  
 تمام کاربردهای از

نرم افزار شون استفاده کند.

پول در میارن

از خدمای که به دیگران می دهند ① → از کجا پول در میارن → MySQL ②

بعضی قسمت هایشون ④ →  
... openS نیست.

{ GNU → تغیر ممکن نیست.  
BSD → ممکنه  
: [ مودودی ]

auditing systems →  
تیم وظایف شود دست و  
به مرقع انجام می دهند.

< تا آخر اسلاید ۵۵ >

## « chp 8 - Software Testing » جلسه (۲۴)

ازیابی نرم افزار

برای یه نرم افزار با سایز معقول بقایه لغت هیچ بگی نداره.

\* هدف ازیابی → نرم افزار کاری که قرار بوده، هی توانایام بده باخیر.  
با اختصار فلان بیشه  
لغت خط از اینقدر

کمتره ...  
PAPCO دلیل

1- درجه شرطی درست کاری کله.

2- درجه شرطی درست کاری کله.

(کامل در صنعت) اسلامی

artificial data

← "سخه" تغایری با جواب درست

روابط کرد

باعتبار بقایه درستی

یشه

Subject  
Date

دلیل بازیه مثال نقص می شد نادرستی رومطمیش شد.  
 تست کردن ویژگی های non-functional با سبیه سازی سوابط

تست کردن > Validation , Verification  
اعتبار ناید

برای هر کدام از نیازمندی ها حداقل ۱ تست لازمه  
برای تریب ویژگی ها لازمه تست های جدا انجام شوند.

منزیت دلیر  $\leftarrow$  test case ۱۴ جزء  
- لایر  $\leftarrow$  فرمیدن نیازمندی ها قبل از پیاده سازی ... (عن)...  
- درجه حالتی نم اقرار درست کار نمی کند. (defect testing)  
تعریف نیازمندیها ناچیز ممکن است اشتباه باشد که ویژگی non-functional  
ممکن است سریش تعریف نشده (نایه جای بیش برو بعد حافظه تمیز شد)  
ممکن است داده را corrupt (خراب) نمایند  $\leftarrow$  محاسبات اشتباه ...

در هدف اول: سیستم به صورتی که بیش بینی نمی شد اجرا شد  
در هدف دوم: یک حالت بیداشد که سیستم درست کار نمی کند

آیا سیستم درست پیاده سازی شده است  
آیا کار صحیح رو طریم اخواصی همیں  
هموینه که کاربری خواهد داشت  
(یعنی ممکن است نیازمندی ها خودسترن اشتباه  
نمایند که بوده اند)

soft ware purpose  
\* سطح اعتماد بالاتر نیازه  
\* سطح اعتماد برنامه های مختلف حیلی متفاوت (خلبان در اعتماد به هواییها،  
کاربر در اعتماد به ... editor)  
\* سطح توقع کاربران درجه هده  
\* قدراء زعیر خارج بازار شده یا ... (ادامه درص و اسلاید)

PAPCO

Subject (اجرایی کن) Dynamic vs static به صورت  $\left\{ \begin{array}{l} \text{خواندن} \\ \text{بازرسی} \end{array} \right.$

Inspection of testing  $\rightarrow$  خواندن (۱)  $\rightarrow$  بازرسی code  
 بررسی نمودارهای (۲)  $\rightarrow$  logical  
 مولته بی خنجری  $\rightarrow$  سسم شده مان  
 مثل معادل این، نمودارهای طراحی مون  $\rightarrow$  اشتباه پیدا شده  
 روش باشه

Testing & Inspection  $\rightarrow$  سکلچ

مزایای بازرسی: ۱) قبل از پیاده سازی ای تواند  
 نشت شود تا با وعده هایش همان ابتدا  
 برطرف شود. (من ۱۳ و ۱۴ ...)

Testing:  $\rightarrow$  ۱) Design Test cases

$\rightarrow$  ۲) prepare test Data

3)

4)

5)

تا آخر مرحله اسلامی

جلسه (۲۵)

Development testing  $\rightarrow$  در زمان توسعه دهنده  $\rightarrow$  غالباً توسط یک توسعه دهنده  
 انجام می شود

Release testing  $\rightarrow$  غالباً توسعه دهنده

یک نماینده دیگر  
 ایا نم اقرار آمده است  
 که به کاربر ارائه داده شود؟

User testing  $\rightarrow$  توسط کاربر

انواع  $\rightarrow$  ۱)  $\rightarrow$  unit testing  
 ۲)  $\rightarrow$  component testing  $\rightarrow$  Interfa<sup>ce</sup>  
 Development testing  $\rightarrow$  ۳)  $\rightarrow$  System testing  $\rightarrow$  component > unit

Unit testing  $\rightarrow$  قسمی از کد  $\rightarrow$  اجزای برنامه جدا جدا شسته  $\rightarrow$  unit:  $\rightarrow$  methods +  
 attributes  $\rightarrow$  از طریق object  $\rightarrow$  از یک کلاس یا که  $\rightarrow$  توزن  
 مثل یک method  $\rightarrow$  set, get



+ سیوف نزدن از محدودی type ها (int و ...)

برای خالی + چک نزدن محدودی آرایه ها + چک نزدن " " ها →  
بعد آرایه که در محدوده اش بیرون و ... در کد نزدیک.

\* یک نست طریق است که تمام اورهایی که غیری کنیم ممکن و وجود داشته باشد در اون بیاد.  
\* اگر یک آرایه بدرست ببرایم بدهیم، هی طور عمل خواهد کرد.

interface → محل دادن لوگو داده است

محل درگیری کلاس ها → استفاده از یک دستگاه  
mem مشارک مثل

interface → یک سری هاروی method هایی  
معمولاً خارجی صدای زده ای شوند.  
نیاز دارند.

برخواست یک خدمت داده شود  
خارجی دارد

interface errors → misuse: interface از اشتباه است.

استفاده درست ولی در جای اشتباه

misunderstanding → Timing errors →

متلاع اول client اجرا شد

server بعد

و client به اطلاعات سرور دست پیدا کند قبل از اینکه

(داده ای هنوز وجود ندارد ولی Server مأمور شد و اجازه بده ...)

(لیکی قصد خوندن شود)

جلسه (۲۶)

۳:۳۰ ← حضور اخباری

(راهنمای کلاسی برد اموز و ...)

۱- تست کردن اول و آخر range

۲- نک تست null بفرستم بینم درست کارمی کنم یا نه

۳- تست کردن حالاتی که ممکن exception ایجاد کنم

۴- تست های stress test ایجاد کنم مثال: ... باز

۵- ص ۴۴ اسلامی

نست system برای use case  
استفاده می شون testing  
component های مختلف رو در لیر کنم  
interface شکل روی component  
را بطری بین component ها  
system testing

\* تست ایندیکاتورین درودی اشتباه وارد کنم  
متلاً قراره ctrl + c بلوغ: اشتباهی ممکن  
دستش بخوبه یه کلمه دیگه رو هم بلوغ

TDD سکل ها

Test Driven Development

نیازمندی های بسیار pass

code coverage → TDD

ویری های جدید رو که اضافه می کنیم، تست های قبلی رو هم run کنیم  
و ویری های قبلی هم تست شوند (چون با افروزن تست های جدید را تحسین و  
مکنند ویری های قبلی هم دچار مشکل شوند)  
متندسازی و  
مشخص کردن عملکرد

دلیل راحت شدن debug → آن bug برخوردیم،

با احتمال جیلی زیاد برای تست های جدید نیز

اضافه کردم

«آخر ص ۴۳»

نود تست ها  
مثل مستندسازی مونه

P4PCO

مکرر و هدف: نرم افزار مطابق با نیازمندی ها شده باشد

محول: یک تیم معاونت این ارزیابی را انجام می دهد  
که با تفکر جمیع دهنده

System testing → شرکت بر پایه  
کردن bug ها

طیافی

یک روش برای شیوه سازی سناریوهای لر  
نیازمندی ها توش دلیری شن  
و مشاهده لیم سسته و چور  
عملی لر

ملهه یک جزء بهتری رست کارنده ولی حا  
در یک فرایند نه!

release testing → performance testing → non-functional requirements

testing

وقتی نیسته جک کرکه سسته بیاده ساریش تعمیم شده باشد

شست باریکه بسستم واردی کیم و رخداد رفته رفته بالا برداشته باشند عرضه سقف حدی که دیله  
نی ده لجاست

غشای بار چند برابر توان سستم توری به شست واردی کیم و باید بینیم آتفاق نالواری لر قابل پیش بینی  
نیست نیفه

user testing →  $\alpha$  testing → کرده لوحی از کاربران سستم را

$\beta$  testing → مورد ارزیابی آثاری دهند

acceptance testing → در هون حیط تورعه

↓  
وقتی یک سفارش دهنده  
و این سسته وجود داره  
P4PCO

مولی  
بعداز  
 $\alpha$  testing

این مورد الزامیه

ص ۳۵ → مراحل

negotiate: مذاکره

در مدل چارک، کاربر با یکم توسعه در ارتباط است. همون تست هایی که ارن زمان کاربر تضمیم می کنند تا روی سیستم اجرا شوند، می توانند به عنوان acceptance testing مود استفاده قرار بگیرند.

اون کاربرها مملو نظر شون با تغییری ذی تغییر دارند.  $\rightarrow$  ولی توی اون بیکی  $\rightarrow$  معاشر مدل تست های زمان توسعه  $\rightarrow$  نظری دهنده ها متعاون باشند!

## chp 9 - Software Evolution :

همی نرم افزارها تغییرات توشن  $\rightarrow$  انتخاب ناپذیر: همی نرم افزارها تغییرات توشن  $\rightarrow$  انتخاب ناپذیر:  $\rightarrow$  بالاخره باید موقتی لازم باشد!

\* بعضی نرم افزارها چندین سال از شون استفاده می شون (مثال: هواپیمایی، نظایری، ...)  $\leftarrow$  بوجهی زیادی  $\rightarrow$  تا به روز نگهش  $\rightarrow$  توسعه می دارند

دیگه نیازمندی جدیدی  $\rightarrow$  سرتیفیکی  $\rightarrow$  ص ۵ نرم افزارهای generic  $\rightarrow$  ص ۶ servKing: مدل word معمولاً این جوری هستند.  $\leftarrow$  مدل bug ها شد و برطرف می شوند فقط این استفاده  $\rightarrow$  کی لست دلخواه bug ها شد  $\rightarrow$  توسعه ای اینست که تغییراتی در مدل داشتند!  $\leftarrow$  توسعه ای اینست که تغییراتی در مدل داشتند!

\* software evolution  $\rightarrow$  سیستمی ب  $\rightarrow$  ۱ type of software روش های استفاده شده در ۲ توسعه مهارت و تجربه ای افراد نیاز نیست (یعنی توسعه؟)  $\rightarrow$  آنرا تغییر را باید روایی کنیم!

Subject: \_\_\_\_\_  
Date: \_\_\_\_\_

قابل استفاده بودن نرم افزار در <sup>(۱)</sup> دسته بندی انواع تغییرات

Fault repair  
platform Adaption  
System Enhancement

چهار platform

آل شیم پایه کشیده کی تغییرات  
با تیم توسعه یابی بناسه  
program understanding → شامل <sup>۳</sup> صیغه  
نیاز دارد.  
آراین مورد تغییرات

محب خواهی قسمت های دلیلی نرم افزاری شدیانه.

(تآخیری اسلاید)

PAPCO