

به نام خدا

تحلیل و طراحی سیستم ها

مدل سازی نیازمندی ها

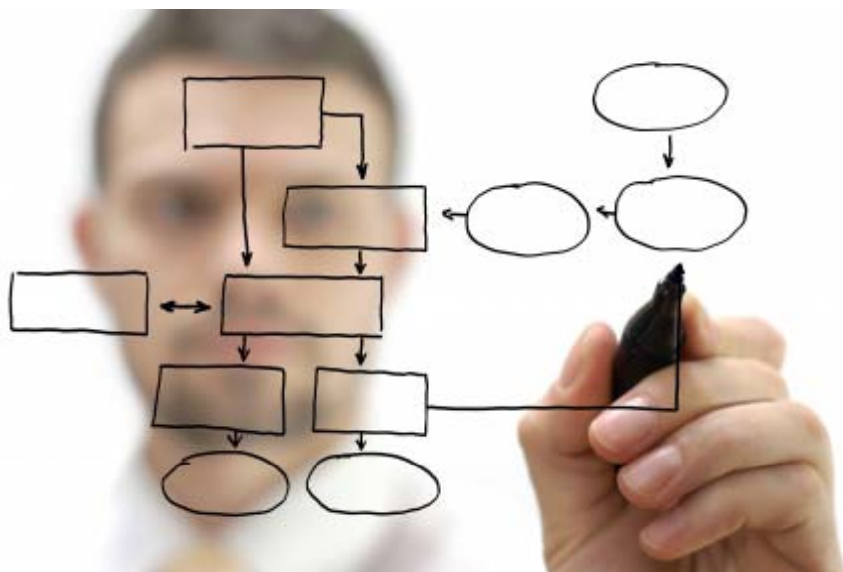
Requirements Modeling

# عناوین مطالب

- 1 مدل سازی نیازمندی ها و مزایای آن
- 2 نکاتی در مورد مدل سازی
- 3 مدل مورد استفاده (use-case)
- 4 مدل های مبتنی بر کلاس
- 5 نمودار فعالیت و swimlane
- 6 مدل های رفتاری
- 7 مدل های مبتنی بر جریان

# مدل سازی نیازمندی ها

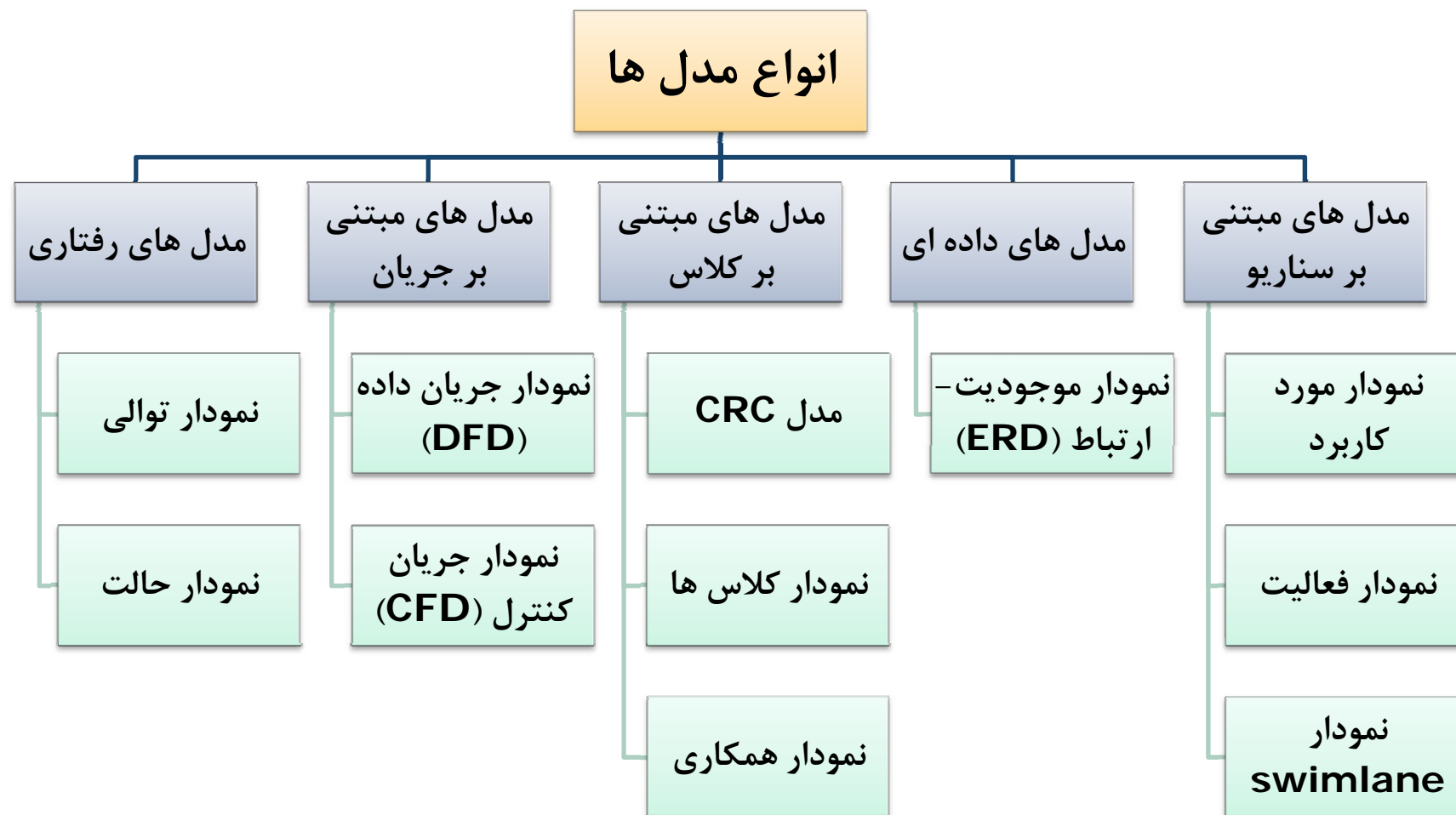
- ❖ مدل، یک ساده سازی از واقعیت دنیای بیرون است.
- ❖ در مدل سازی نیازمندی ها، با استفاده از ترکیب شکل و متن، نمودارهایی ایجاد می شود که نیازمندی های سیستم را تصویرسازی می کنند.
- ❖ گام سوم از مهندسی نیازمندی ها
- ❖ تبدیل نیازمندی های اولیه به نیازمندی های قابل انجام
- ❖ تشریح نیازمندی های مشتری
- ❖ یک رویکرد مبتنی بر تکرار



# مزایای مدل سازی

- ❖ دستیابی به درک بهتری از سیستم مورد نظر → تولید محصول با کیفیت بهتر
- ❖ نمایش نیازمندی ها از بعد های مختلف → افزایش احتمال کشف خطاها، ناسازگاری ها و کاستی ها
- ❖ ایجاد مبنایی برای طراحی نرم افزار
- ❖ ایجاد مبنایی برای اعتبار سنجی نرم افزار

# انواع مدل سازی نیازمندی ها



# نکاتی در مورد مدل سازی

- ❖ تنها نیازمندی هایی را تحلیل و مدل کنید که در دامنه کاربرد مساله قرار دارند.
  - حذف داده ها و اطلاعات غیر مرتبط با دامنه مسئله
- ❖ مدل مناسب، مدلی است که تطابق کامل با واقعیت داشته باشد. (تطابق با محصولی که در نهایت تولید می شود).
- ❖ مدل مناسب، مدلی است که به درک ما نسبت به نیازمندی های نرم افزار، بیافزاید.
- ❖ مدل باید برای افراد ذی نفع (از مشتری تا تیم توسعه دهنده) دارای ارزش باشد.
- ❖ نیاز به مدل سازی برای سیستم های پیچیده همیشه وجود دارد، زیرا نمی توان تمام پیچیدگی ها را به صورت مستقیم درک و پیاده سازی کرد.
- ❖ سادگی مدل را تا حد امکان حفظ کنید.
- ❖ ملاحظات مربوط به زیر ساخت ها مربوط به مرحله طراحی می شوند نه مدل سازی.

# نکاتی در مورد مدل سازی ...

- ❖ هیچ مدلی به تنهایی غالباً برای ساخت یک محصول کافی نیست و در اکثر اوقات، لازم است تا بیشتر از یک مدل ایجاد شود. (مجموعه ای از مدل‌های مختلف برای بخش‌های مختلف سیستم)
- ❖ انتخاب مدل‌هایی که برای ساخت یک محصول استفاده می شوند، کاملاً با فرآیند تولید و نوع محصول مرتبط است.
- ❖ هر مدل می تواند در سطوح دقت مختلفی ساخته شود.
- معمولاً در مراحل اولیه، مدل‌ها به صورت کلی ساخته می شوند و سپس به تدریج، مدل‌ها با جزئیات بیشتری ایجاد می شوند.

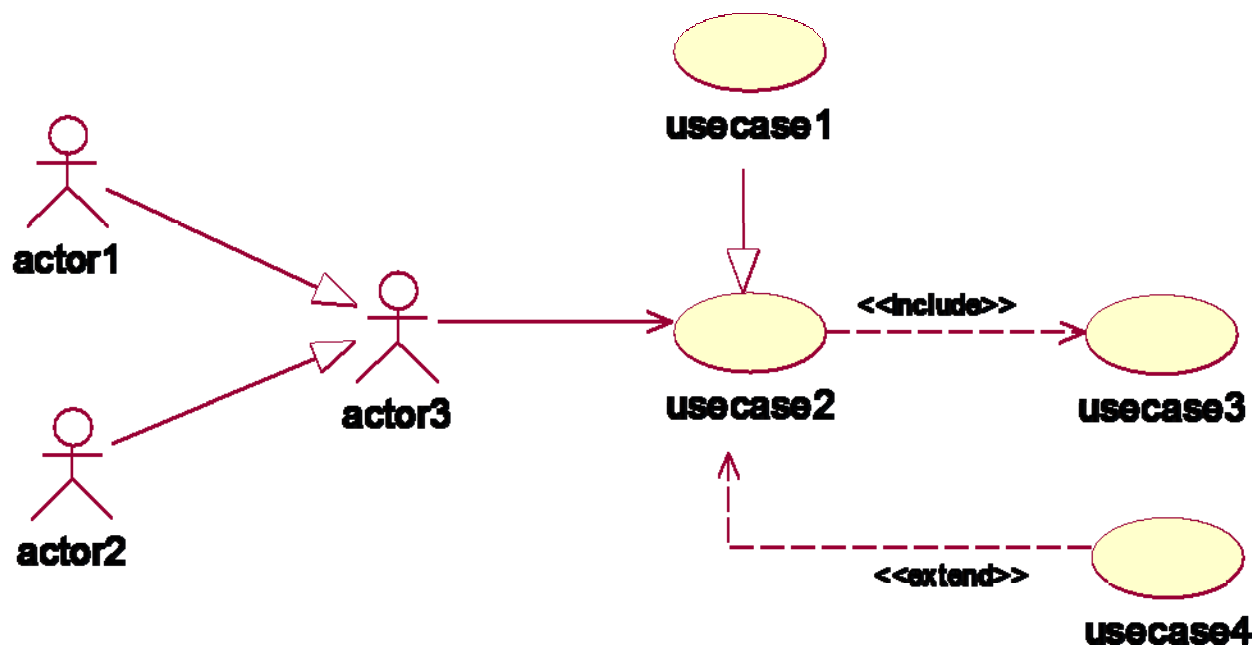
مدل مورد استفاده

**USE CASE MODEL**

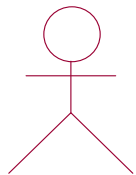


# مدل مورد استفاده (use case model)

- ❖ نمودار مورد استفاده، شامل عملیاتی است که سیستم انجام می‌دهد تا نتایج قابل مشاهده و ارزشمندی را برای استفاده کننده کنندگان از سیستم، فراهم کند.
- ❖ مجموعه ای از موارد استفاده (use case)، اکترها (actor) و روابط بین آنها
- ❖ توصیف رفتار سیستم از دید کاربر بدون توجه به چگونگی پیاده سازی آن



# کنشگر (Actor)



Actor

❖ کنشگر / بازیگر / عامل / اکر

▪ هر چیزی که بیرون از (خارج از مرز های) سیستم مورد نظر قرار دارد و از سیستم استفاده می کند.

• انسان ، سیستم نرم افزاری، سیستم سخت افزاری و...

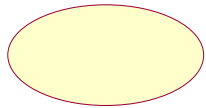
❖ یک اکر می تواند از سیستم سرویس بگیرد، سرویسی را به سیستم ارائه دهد و یا هر دو

❖ کسانی که از سیستم استفاده می کنند، می توانند نقش های متفاوتی داشته باشند اما هر اکر فقط یک نقش را می تواند بر عهده داشته باشد.

• برای مثال در یک گروه آموزشی، یک فرد می تواند هم برنامه ریز دروس باشد و هم مسئولیت ثبت نام دانشجویان را برعهده داشته باشد.

• برای هر نقش، باید یک اکر تعریف شود.

## مورد استفاده (use case)



UseCase

❖ یک رفتار دلخواه از سیستم بدون توجه به چگونگی پیاده سازی آن

❖ استفاده ای که یک اक्टर از سیستم می کند.

❖ سرویسی است که سیستم به اक्टर می دهد یا بر عکس.

❖ مثال

■ انتخاب درس، حذف درس، سفارش کالا، ایجاد حساب کاربری، مشاهده جزئیات پروفایل و ...

## شرح مورد استفاده

❖ هر use case، دارای یک شرح یا توضیح است که در آن رفتار use case با توصیف جریانی از رویداد ها (Flow of Events)، مشخص می شود.

- جریان اصلی رویداد ها / سناریو اصلی
- جریان فرعی رویداد ها / سناریو جایگزین / رویداد های استثنا

❖ روش های نوشتن شرح use case

- متن بدون ساختار
- متن ساخت یافته
- شبه کد (pseudo code)

## شرح مورد استفاده ...

❖ شرح use case با قالب متنی ساخت یافته

نام مورد استفاده	یک نام منحصر به فرد است.
توصیف مختصر	
اکترها	افراد یا چیزهایی که از این مورد استفاده، استفاده می کنند.
شرایط اولیه (Pre-Conditions)	حالتی که سیستم باید دارای آن باشد تا بتوان این مورد استفاده را اجرا کرد.
جریان اصلی رویدادها	توصیفی از آنچه سیستم برای اجرای مورد استفاده باید انجام دهد.
جریان فرعی رویدادها (استثناها)	وضعیتی که باعث می شود سیستم رفتار متفاوتی از آنچه که مورد انتظار است، از خود نشان دهد.
شرایط نهایی (Post-Conditions)	فهرستی از حالت هایی که سیستم پس از اجرای این مورد استفاده دارای یکی از آنها خواهد بود.
اولویت پیاده سازی	
و ...	

### شرح مورد استفاده ...

❖ برای شناسایی رویداد های استثنا می توان از سوالات زیر استفاده کرد:

- آیا شرایطی هست که use case نیاز به اعتبار سنجی داشته باشد و طی آن با خطا روبرو شود؟
- آیا شرایطی هست که در آن سیستم قادر به پاسخ دهی مناسب به درخواست کاربر نباشد؟
- آیا عملکرد ضعیف سیستم ممکن است به رفتار غیر منتظره از سوی کاربر منجر شود؟
- رفتار های اختیاری در طی اجرای use case

✓ در صورتی رویدادهای استثنا در شرح use case نوشته می شوند که سیستم قادر به تشخیص آنها باشد و بتواند اقدام مناسب را در برابر آنها انجام دهد.

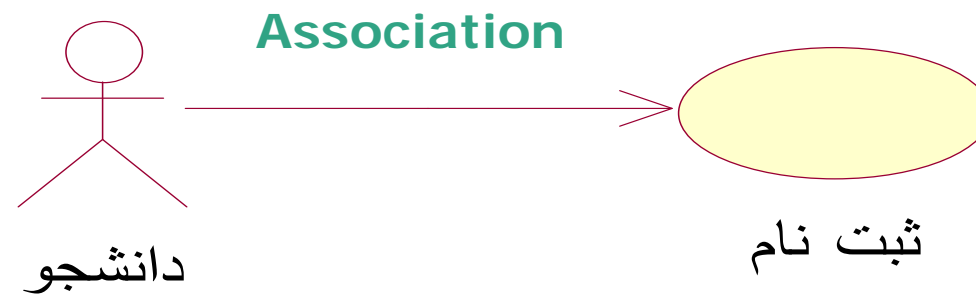
## شرح مورد استفاده ...

❖ مثال

نام مورد استفاده	اعتبار سنجی کاربر (Validate User)
توصیف مختصر	سیستم هویت کاربر را بررسی می کند و اجازه انجام عملیات را به او می دهد.
اکترها	مشتري
شرایط اولیه	نام کاربری و کلمه عبور مناسب در اختیار مشتری قرار دارد.
جریان اصلی رویدادها	<ol style="list-style-type: none"> <li>۱. کاربر، نام کاربری و کلمه عبور خود را وارد می کند و دکمه ورود را فشار می دهد.</li> <li>۲. سیستم نام کاربری و کلمه عبور را چک می کند.</li> <li>۳. سیستم یک پیغام خوشآمد گویی نمایش می دهد.</li> </ol>
جریان فرعی رویدادها	<ul style="list-style-type: none"> <li>• اگر کاربر دکمه لغو را فشار دهد، مورد استفاده «اعتبار سنجی کاربر» خاتمه پیدا می کند.</li> <li>• اگر نام کاربری یا کلمه عبور نامعتبر باشد، یک پیغام خطا نمایش داده می شود و مورد استفاده «اعتبار سنجی کاربر» دوباره فعال می شود.</li> <li>• اگر کاربر بیش از سه بار نام کاربری یا کلمه عبور نامعتبر وارد کرده باشد، کاربر تا ۱۰ دقیقه اجازه دسترسی به سیستم را نخواهد داشت.</li> </ul>
شرایط نهایی	کاربر وارد سیستم شده است و اجازه انجام عملیات خاصی به او داده می شود.

# ارتباط Association

❖ به ارتباط بین اکتور و مورد استفاده، association گفته می شود.





## مراحل مدل سازی

### ❖ مراحل

- شناسایی اکر ها و سازماندهی آنها
- شناسایی موارد استفاده و سازماندهی آنها
- تعیین ارتباط میان اکر ها و موارد استفاده
- بسته بندی اکر ها و موارد استفاده مرتبط
- رسم نمودار
- ارزیابی مدل

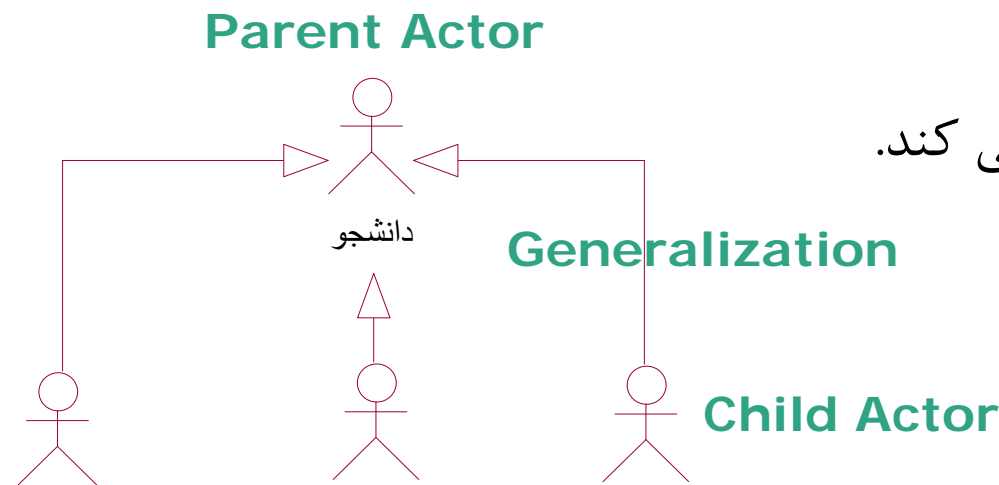
✓ مهمترین فعالیت در فرآیند ایجاد مدل، شناسایی اکر ها و موارد استفاده است.

# شناسایی و سازماندهی اکرها

❖ بر مبنای اهداف مسئله و نیازمندی های سیستم، کاربران و استفاده کنندگانی که با سیستم تعامل مستقیم دارند، شناسایی می شوند.

❖ در نهایت، فقط اکرهایی انتخاب می شوند که حداقل با یک use case در ارتباط باشند.

❖ سازماندهی اکرها با استفاده از رابطه Generalization



■ مکانیزم ارث بری

■ به کاهش حجم کد کمک می کند.

دانشجوی دکتری      دانشجوی کارشناسی      دانشجوی کارشناسی ارشد

## شناسایی موارد استفاده

### ❖ شناسایی موارد استفاده

- نحوه ارتباط کاربر با سیستم چگونه است؟
- کاربر چه سرویس‌هایی را از سیستم درخواست می کند؟
- ✓ آیا بوسیله موارد استفاده شناسایی شده، نیازمندی های مورد انتظار برآورده می شود؟

## سازماندهی موارد استفاده

❖ رابطه عام/ خاص (Generalization)

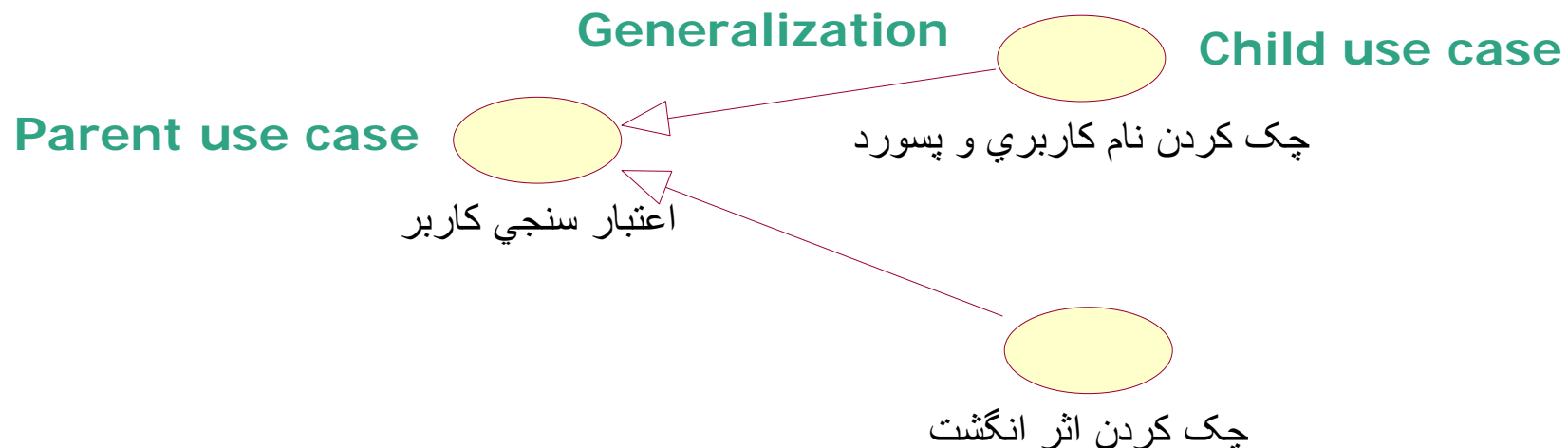
❖ رابطه شامل بودن (include)

❖ رابطه گسترش دادن (extend)

# سازماندهی موارد استفاده ...

### ❖ رابطه Generalization

- مورد استفاده فرزند، رفتار و معنای مورد استفاده والد را به ارث می برد.
- فرزند می تواند رفتار پدر را بازنویسی کند یا چیزی به آن اضافه کند. هر جایی که پدر حضور دارد، فرزند هم می تواند حضور داشته باشد.
- پدر و فرزند می توانند نمونه های مشترک داشته باشند.



# سازماندهی موارد استفاده ...

### ❖ رابطه شامل بودن (include)

- یک use case برای انجام بخشی از وظایف خود از use case دیگر استفاده می کند.
- یک use case، صریحا شامل رفتار یک use case دیگر هست.
- زمانی از این رابطه استفاده می شود که بخشی از وظایف چند use case، مشابه به یکدیگر باشد. در این حالت، برای جلوگیری از دوباره کاری، یک use case جدید برای وظایف مشترک تعریف می شود تا use case های دیگر، هر زمان که نیاز باشد، از آن استفاده کنند.

# سازماندهی موارد استفاده ...

### ❖ رابطه شامل بودن (include)

#### ■ مورد استفاده «پیگیری سفارش» سناریوی متنی بدون ساختار

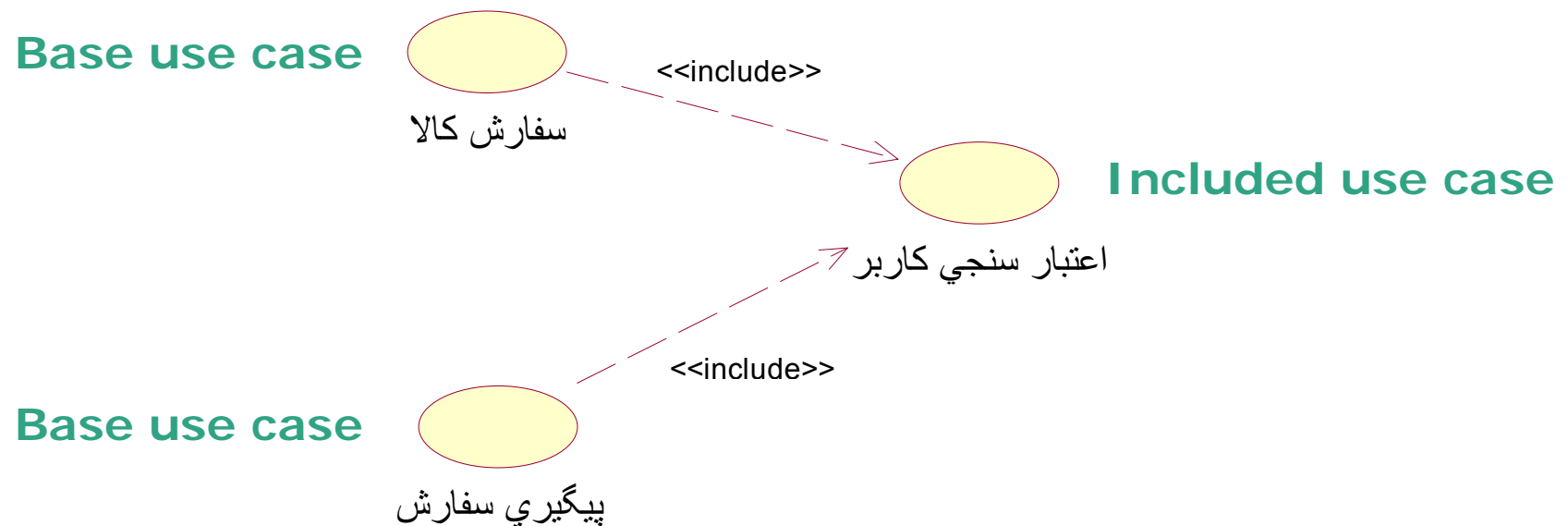
مشتری کد سفارش خود را وارد می کند، در صورتی که کد سفارش معتبر باشد، هویت مشتری بررسی می شود. اگر هویت او معتبر باشد، به ازای هر آیتم موجود در سفارش، وضعیت آن از پایگاه داده درخواست می شود. در نهایت، وضعیت آیتم های سفارش داده شده، به صورت یک فرم جدولی به مشتری نمایش داده می شود.

#### ■ مورد استفاده «سفارش کالا» سناریوی متنی بدون ساختار

ابتدا هویت مشتری بررسی می شود. اگر هویت او معتبر باشد، مشتری سبد خرید خود را مشاهده کرده و آن را بروز رسانی می کند. در صورت تایید سفارش، صورت حساب مشتری تهیه می شود و مشتری اقدام به پرداخت هزینه به صورت آنلاین می کند. در صورت پرداخت موفق، یک کد سفارش به مشتری داده می شود تا بر مبنای آن بتواند وضعیت سفارش را تا زمان تحویل محصول پیگیری کند.

## سازماندهی موارد استفاده ...

❖ رابطه شامل بودن (include)





# سازماندهی موارد استفاده ...

### ❖ رابطه گسترش دادن (extend)

- یک مورد استفاده پایه (base use case)، به صورت ضمنی شامل رفتار یک مورد استفاده دیگر (extension use case) می باشد.
- مورد استفاده پایه می تواند به تنهایی استفاده شود، اما در یک موقعیت خاص ممکن است رفتارش توسط یک use case دیگر گسترش پیدا کند. به این موقعیت های خاص ، نقاط گسترش (extension points) گفته می شود.
- زمانی از این رابطه استفاده می شود که بخواهیم یک رفتار اختیاری برای use case را نشان دهیم.

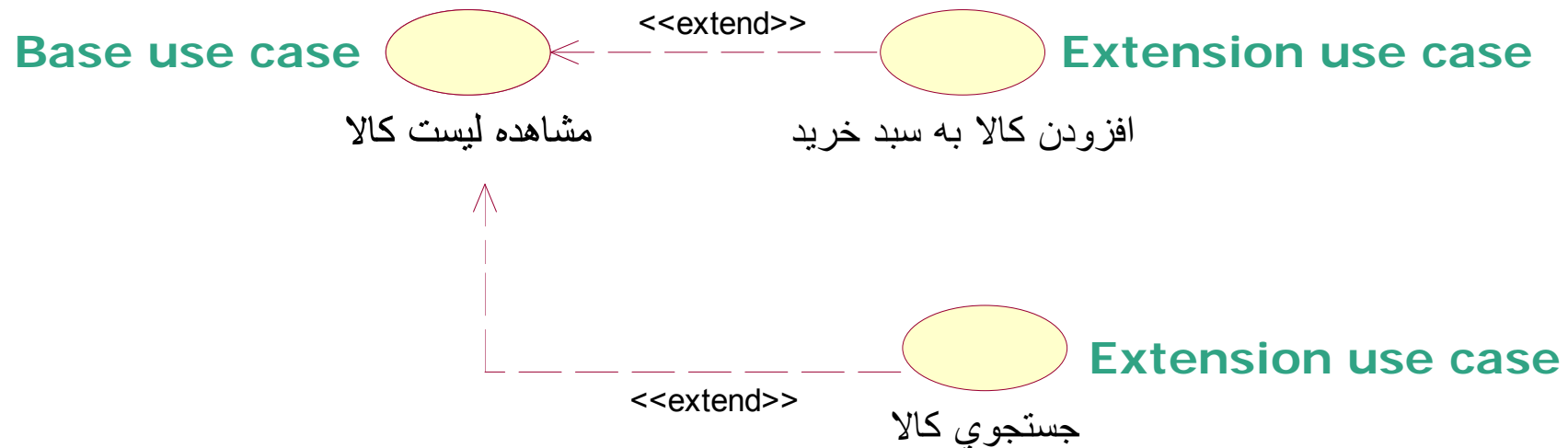
# مدل use case – مراحل مدل سازی

## شناسایی و سازماندهی موارد استفاده ...

### ❖ رابطه گسترش دادن (extend)

#### ■ مورد استفاده «مشاهده لیست کالا» سناریوی متنی بدون ساختار

لیستی از کالاهای فروشگاه به همراه یک سری اطلاعات در مورد هر کالا، به کاربر نمایش داده می شود. کاربر می تواند در صورت تمایل، کالای مورد نظر را به سبد خرید خود اضافه کند. همچنین امکان جستجوی کالاها نیز وجود دارد.



## مدل use case – مراحل مدل سازی

### تعیین ارتباط میان اکر ها و موارد استفاده

❖ هنگام شناسایی موارد استفاده، بهتر است عواملی که با آنها تعامل دارند را نیز شناسایی کنید.

اکر	مورد استفاده
کاربر جدید	مشاهده لیست کالا
کاربر جدید	ثبت نام
کاربر عضو	مشاهده لیست کالا
کاربر عضو	ثبت سفارش
کاربر عضو	پیگیری سفارش
مدیر فروشگاه	مدیریت کالاهای فروشگاه
مدیر فروشگاه	مدیریت پروفایل مشتریان
مدیر فروشگاه	مدیریت سفارشات
...	...

# مدل use case – مراحل مدل سازی

## رسم نمودار

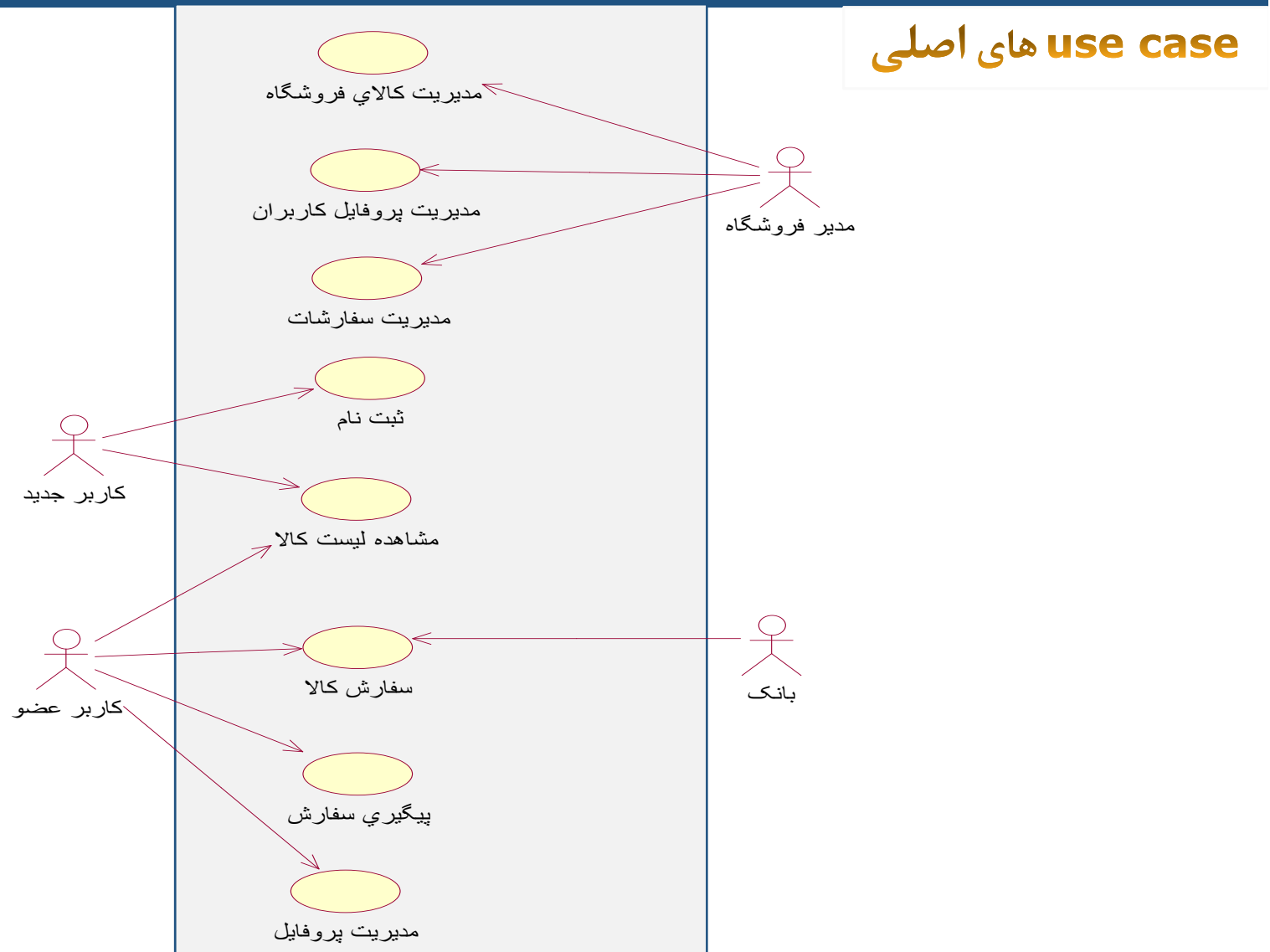
❖ مثال

### ■ فروشگاه اینترنتی

- هدف از این سیستم طراحی یک وب سایت برای مدیریت، عرضه و فروش کالاهای یک فروشگاه است. بدین ترتیب فروشندگان می توانند بدون محدودیت زمانی و مکانی کالاهای خود را به مشتریان عرضه کنند و مشتریان نیز می توانند کالاهای موردنظر را خود را سفارش داده و پس از پرداخت هزینه به صورت آنلاین، آنها را درب منزل تحویل بگیرند.

# مدل use case – مراحل مدل سازی

## رسم نمودار ...



## مدل use case – مراحل مدل سازی

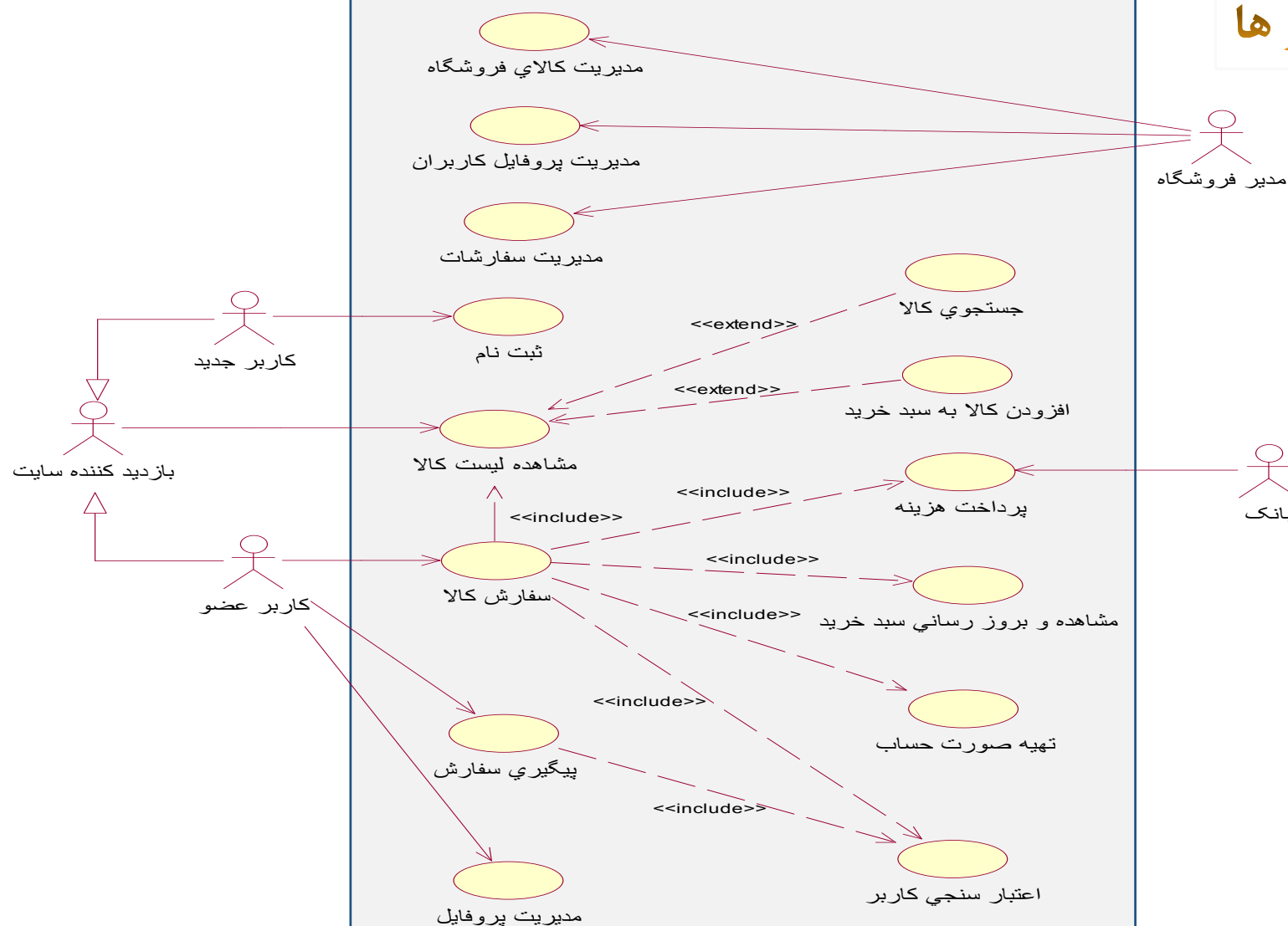
## رسم نمودار ...



# مدل use case – مراحل مدل سازی

## رسم نمودار ...

سازماندهی اکثرها



## مدل use case – مراحل مدل سازی

### بسته بندی اکتورها و موارد استفاده مرتبط

- ❖ بسته بندی مکانیزمی است برای گروه بندی عناصر منطقاً به هم مرتبط
- ❖ در مدل use case، یک بسته (Package) می تواند شامل تعدادی اکتور و موارد استفاده آن اکتورها باشد.
- ❖ درک و نگهداری مدل آسانتر می شود.

خرید آنلاین

مدیریت کالا

مدیریت پروفایل ها

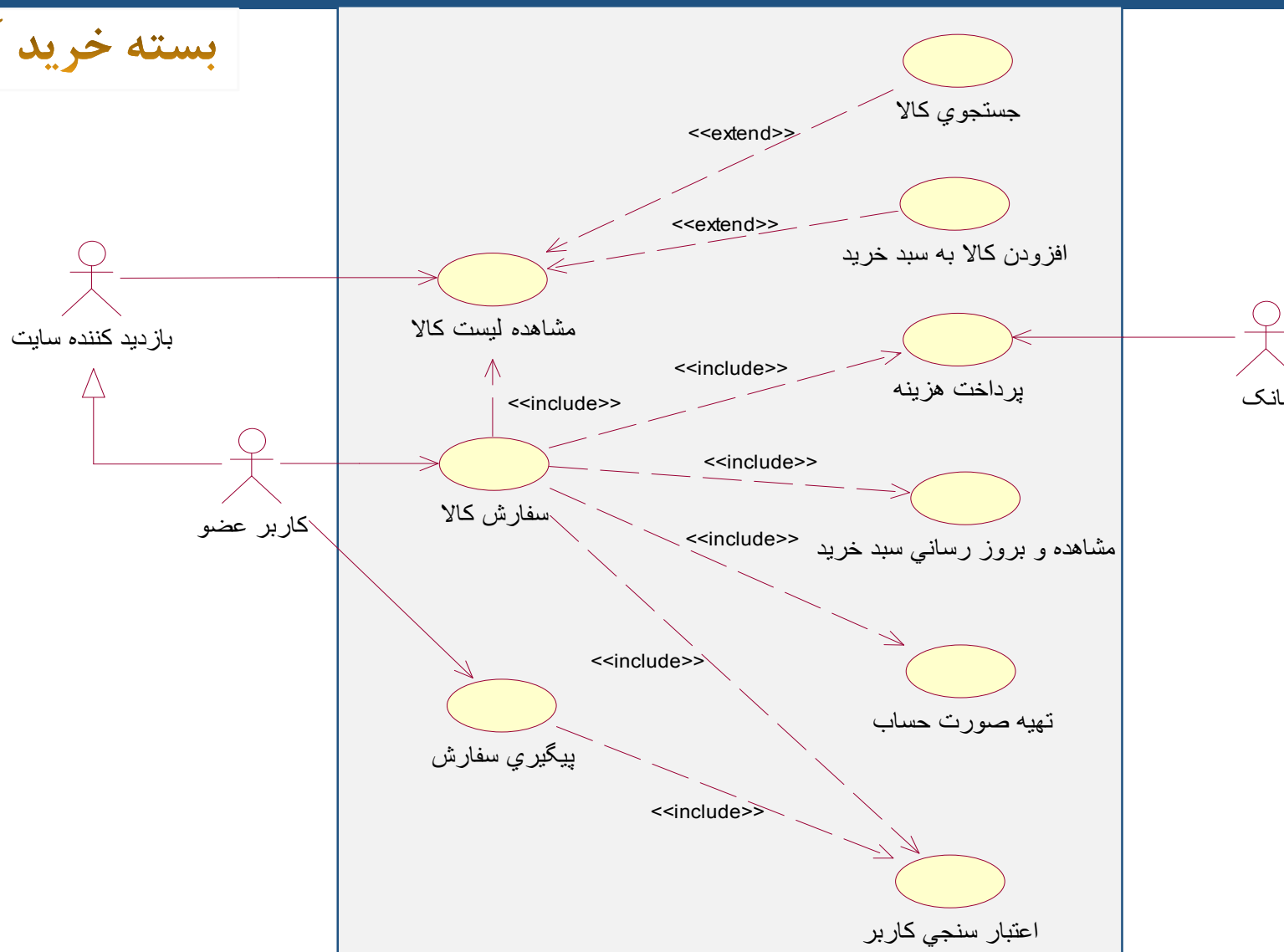
مدیریت سفارشات



# مدل use case – مراحل مدل سازی

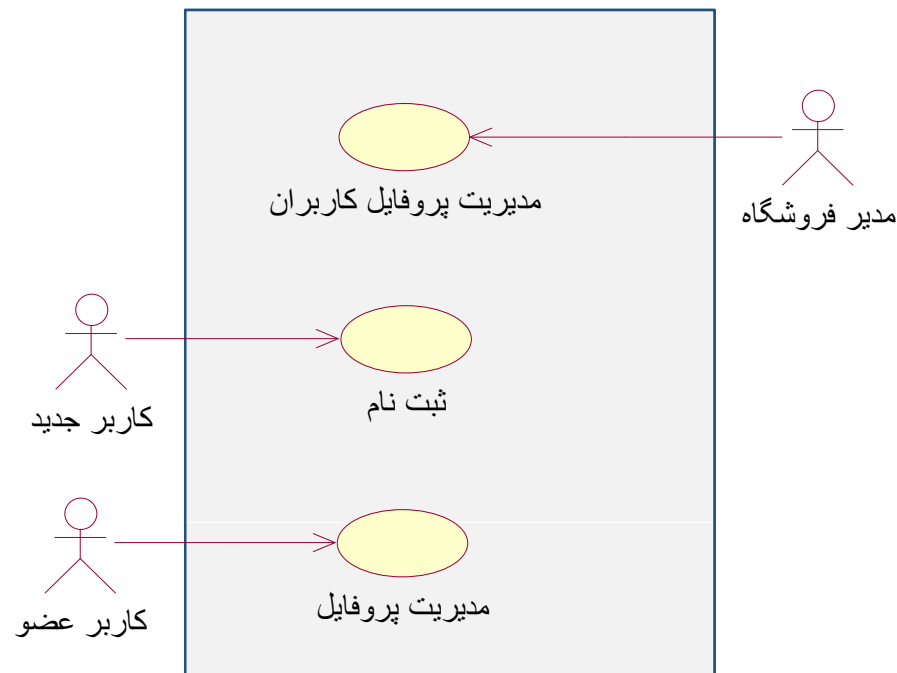
## بسته بندی اکتر ها و موارد استفاده

### بسته خرید آنلاین



## مدل use case – مراحل مدل سازی

### بسته بندی اکتر ها و موارد استفاده ...



**بسته مدیریت پروفایل**

### نکات مدل سازی

- ❖ فقط اکتر ها و use case هایی را مشخص کنید که در نهایت در بخشی از سیستم مورد استفاده قرار گرفته و پیاده سازی می شوند.
- ❖ برای اکتر ها و use case ها نام مناسب و با معنا انتخاب کنید تا بر اساس آن بتوان یک اطلاعات پایه از عملکرد آنها بدست آورد.
- ❖ نام اکتر ها و use case (حداقل در یک بسته) باید منحصر بفرد باشد.
- ❖ در نمودار use case نباید روابط جزئی را نشان داد؛ زیرا هدف اصلی از این نمودار، شناخت نیازمندی های سیستم است و به چگونگی پیاده سازی آنها و جزئیات پرداخته نمی شود.
- ❖ با استفاده از امکان یادداشت گذاری (Note)، می توان نیازمندی ها را در صورت لزوم به صورت شفاف تر توصیف کرد.

# مدل use case

## ❖ مزایا

- مدل کردن رفتار سیستم ، زیر سیستم یا کلاس
- به تولیدکنندگان و استفاده کنندگان از نرم افزار کمک می کند تا دید واحد و مشترکی در رابطه با امکانات نرم افزاری که قرار است ساخته شود، داشته باشند.
- بدست آوردن یک دید سطح بالا از سیستم و مشخص شدن مرز آن
- می توان مدل سازی use case را به موازات جمع آوری نیازمندی ها انجام داد.
- مبنایی برای نوشتن سناریوهای آزمون و همچنین آزمون اعتبار
- مبنایی برای طراحی اینترفیس کاربر

## ❖ نقاط ضعف

- با استفاده از این مدل، در برخی از موارد نمی توان اطلاعات مربوط به نیازمندی ها و جنبه های سیستم را به طور واضح بیان کرد.
- ترتیب اجرای عملیات را نشان نمی دهد.

# نمودار فعالیت

## ACTIVITY DIAGRAM

# نمودار فعالیت (Activity Diagram)

- ❖ نمایش مراحل انجام یک کار در زمان اجرا
- ❖ نمایش فعالیت ها و تصمیم گیری هایی را که در اجرای یک عملکرد رخ می دهند.
- ❖ مشابه به نمودار گردش (Flow Chart) است و جریان کنترل را از یک فعالیت به فعالیت دیگر نشان می دهد؛ با این تفاوت که می تواند جریان های همروند (موازی) را نیز نشان دهد.

## ❖ انواع جریان کنترلی

- ترتیبی (Sequential)
- شرطی (Conditional)
- موازی (Parallel)

## اجزای نمودار



❖ گره شروع



❖ گره پایان

Activity

❖ گره فعالیت

▪ یک عمل که توسط سیستم یا کاربر اجرا می شود.

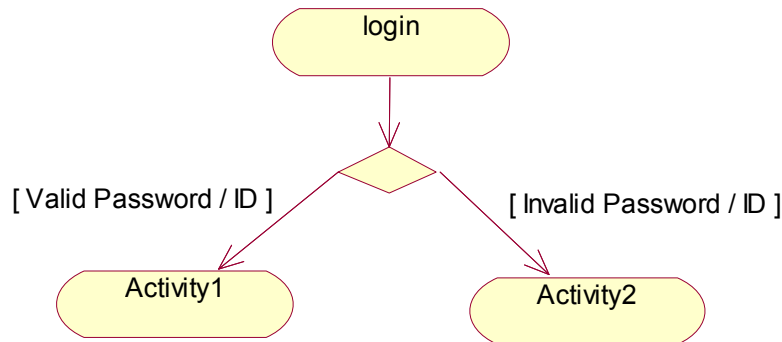
❖ انتقال (Transition)

▪ پیکان هایی که یک گره فعالیت را به گره دیگر متصل می کنند.

▪ پس از کامل شدن فعالیت اول، فعالیت دوم شروع می شود.



## اجزای نمودار ...



### ❖ گره تصمیم گیری (Decision)

#### ■ کاربرد ها

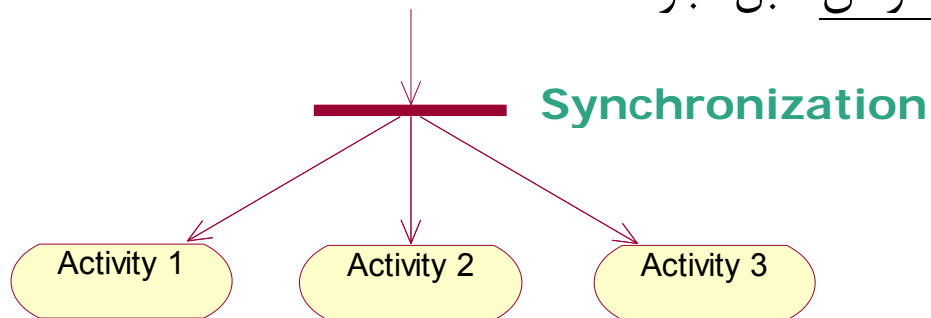
- نشان دادن جریان های شرطی
- تصمیم گیری در مورد انتخاب یک فعالیت از میان چند فعالیت
- در صورتی که چند فعالیت از مسیر های مختلف و به صورت غیر همزمان اجرا شوند و در نهایت به یک فعالیت مشترک برسند، برای ادغام (merge) آنها می توان از گره تصمیم گیری استفاده کرد.
- هر گره تصمیم گیری حداقل یک جریان ورودی و یک جریان خروجی دارد.
- جریان ها می توانند با استفاده از شرط های نگهبان (Guard Condition) برچسب گذاری شوند.
- شرط های نگهبان در داخل کروشه [ ] نوشته می شوند.
- در زمان اجرای عملکرد، در صورتی یک پیکان خروجی دنبال می شود که شرط نگهبان آن درست باشد.



## اجزای نمودار ...

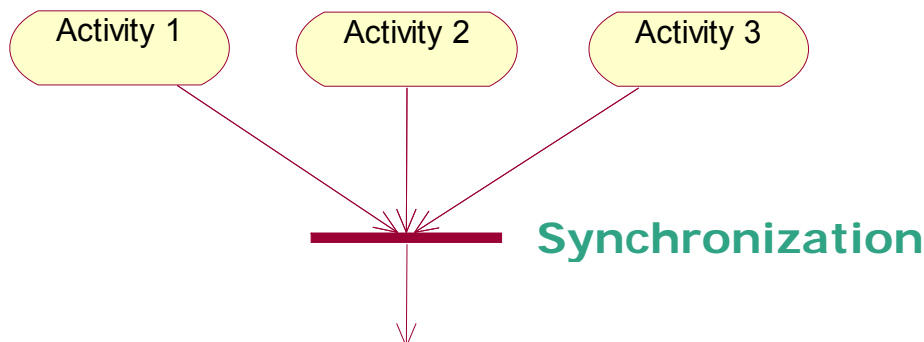
### ❖ انشعاب (Fork)

- نمایش دو یا چند فعالیت که به طور همزمان قابل اجرا هستند.



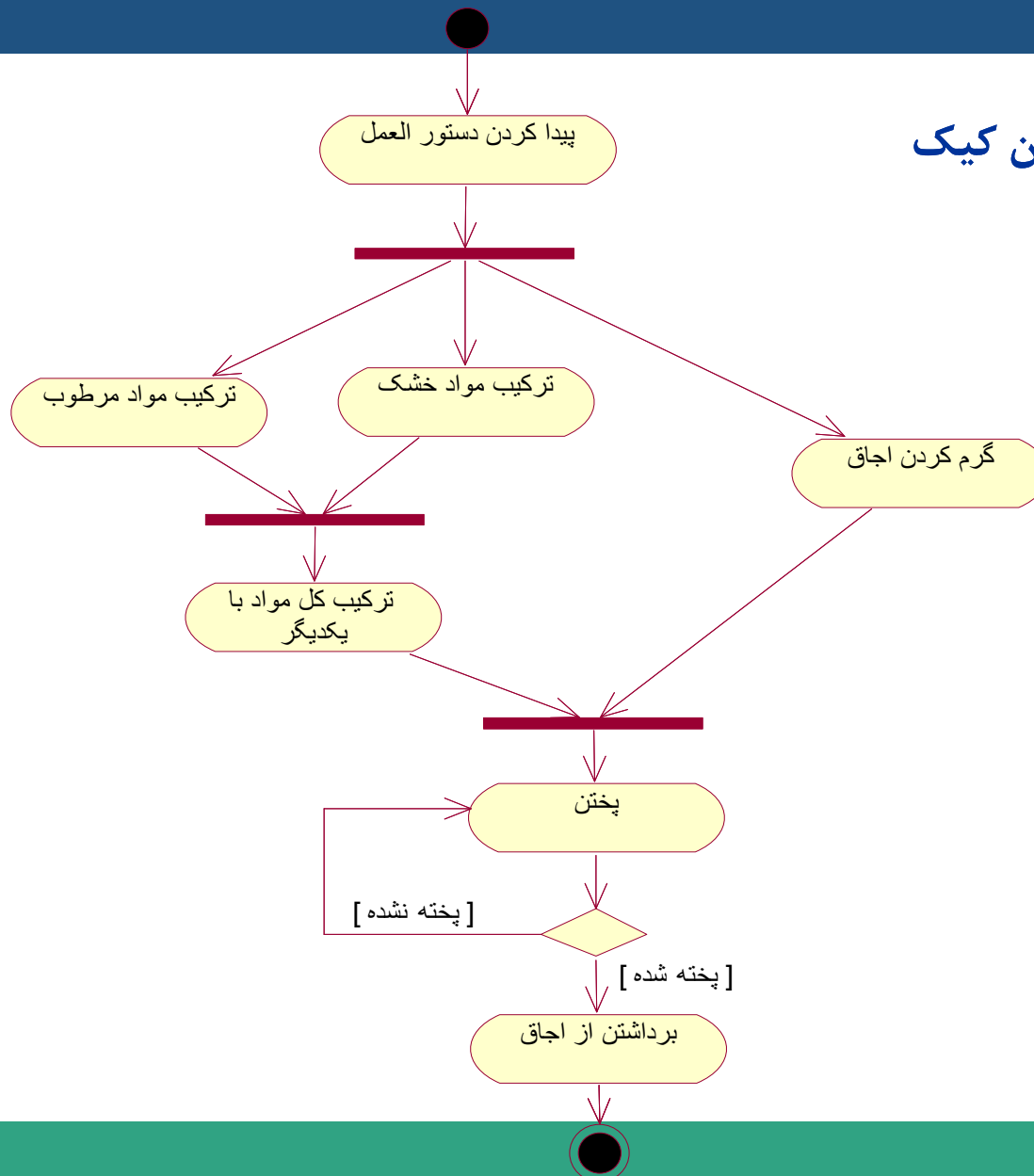
### ❖ اتصال (Join)

- همگام سازی دو یا چند فعالیت همروند
- فعالیت خروجی زمانی آغاز می شود که همه فعالیت های ورودی تمام شده باشند.



# نمودار فعالیت ...

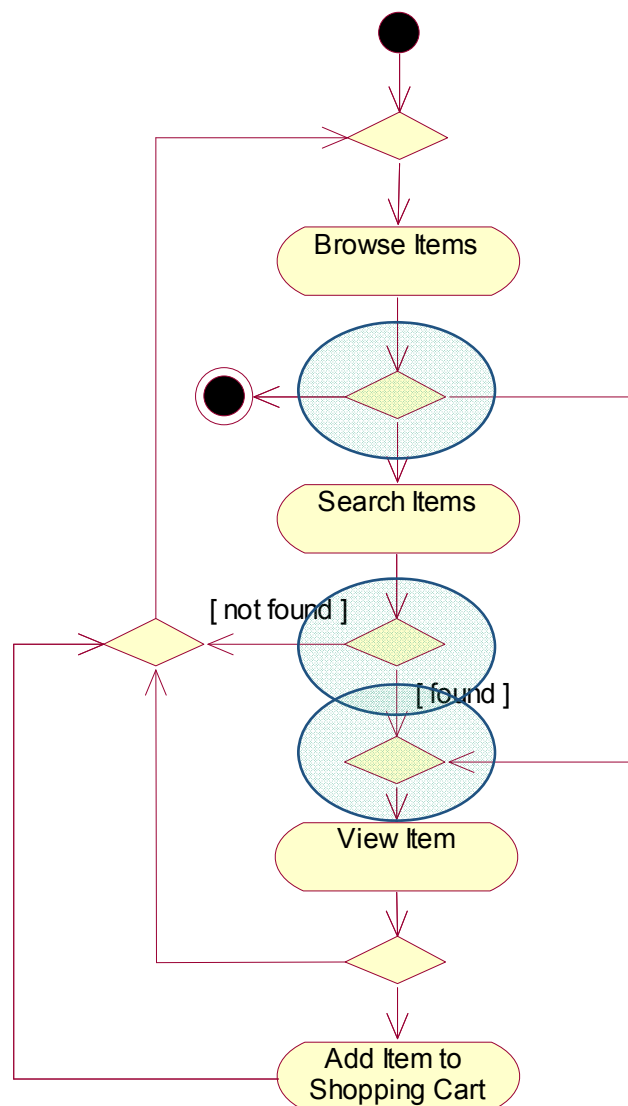
❖ نمودار فعالیت برای پختن کیک



# نمودار فعالیت ...

## ❖ مورد استفاده «مشاهده لیست کالا»

لیستی از کالاهای فروشگاه به همراه یک سری اطلاعات در مورد هر کالا، به کاربر نمایش داده می شود. کاربر می تواند در صورت تمایل، کالای مورد نظر را به سبد خرید خود اضافه کند. همچنین امکان جستجوی کالاها نیز وجود دارد.



# نمودار فعالیت ...

❖ در یک سیستم واقعی، مدلسازی همه جریان کارها در یک نمودار فعالیت امکان پذیر نیست. معمولاً، برای بیان چگونگی انجام عملکرد های پیچیده، به ازای هر عملکرد، یک نمودار فعالیت جداگانه رسم می شود.

## ❖ مزایا

- نمایش مراحل و ترتیب انجام کار در زمان اجرای سیستم
- نمایش جریان های موازی و شرطی
- مدل سازی و فهم بهتر جریان کار سیستم، جریان کار یک عملکرد خاص از سیستم، جریان کار یک کلاس یا use case

## ❖ نقاط ضعف

- عدم نمایش پیام ها و چگونگی تبادل آنها در حین انجام فعالیت ها
- نمودار فعالیت به طور مستقیم به کد تبدیل نمی شود.

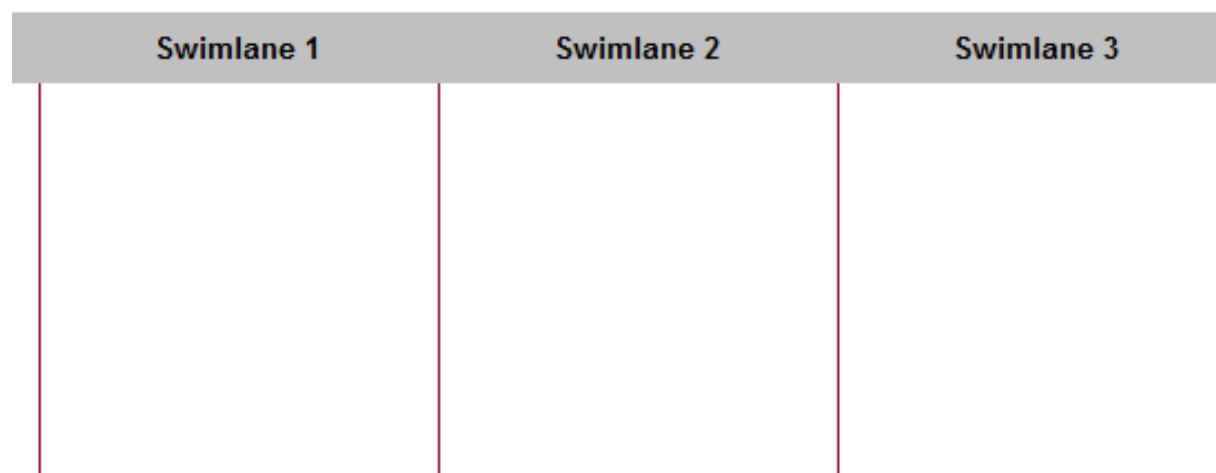
نمودار بخش بندی

**SWIMLANE DIAGRAM**

# نمودار بخش بندی (Swimlane Diagram)

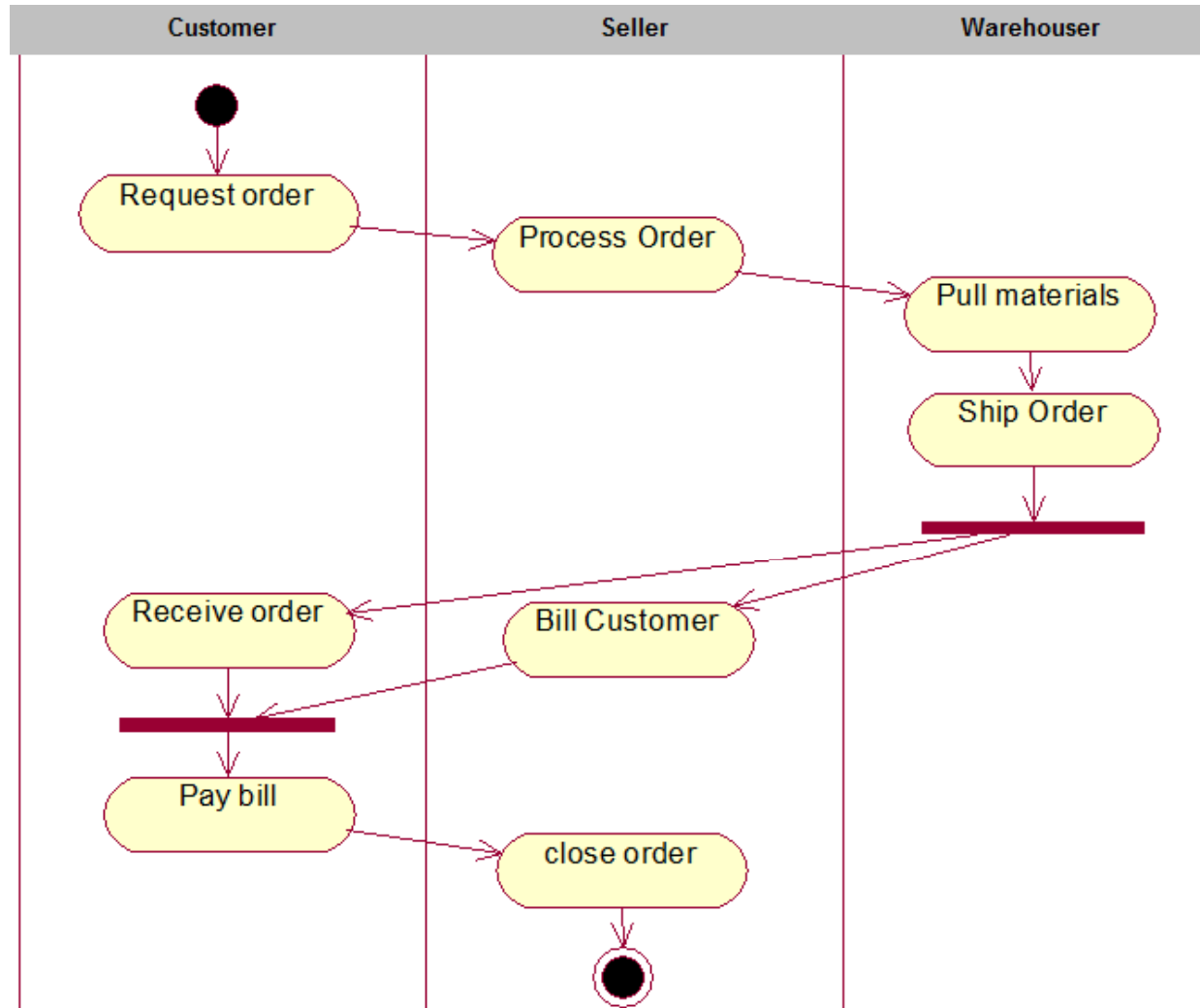
❖ شکل دیگری از نمودار فعالیت است با این تفاوت که علاوه بر فعالیت ها و جریان کنترل بین آنها، مشخص می کند چه کسی و یا چه کلاسی مسئول انجام هر فعالیت است.

❖ نمودار به چند بخش عمودی (Swimlane) تقسیم می شود. در هر بخش، فعالیت های مربوط به یک کنشگر یا کلاس قرار می گیرد.

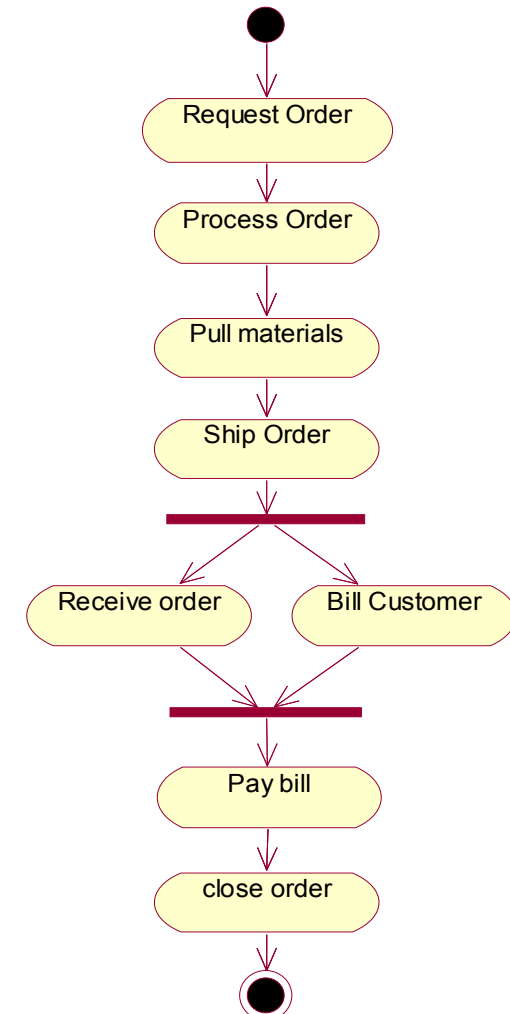


# نمودار بخش بندی ...

نمودار بخش بندی



نمودار فعالیت



نمودار کلاس

**CLASS DIAGRAM**



# نمودار کلاس (Class Diagram)

❖ کلاس، یکی از اجزای اصلی در متدولوژی شی گرا است.

✓ بسته بندی داده ها و عملیات مرتبط با آنها (کپسوله کردن)

✓ پنهان سازی اطلاعات

✓ طبقه بندی، وراثت، چند ریختی

## ❖ نمودار کلاس ها

■ مدل کردن صفات و عملیات کلاس ها و ارتباطات میان آنها

■ یک دید ایستا و ساختاری از کل سیستم را در اختیار تحلیلگر قرار می دهد.

# مفاهیم پایه

## ❖ شی (object)

- یک مفهوم کلی است بگونه‌ای که دارای هویت بوده و قادر به بروز رفتار و ثبت حالات (وضعیت) خود باشد.

## ❖ کلاس (class)

- مجموعه‌ای از اشیا که دارای صفات و رفتار مشترک هستند.
- هر شی، یک نمونه (instance) از یک کلاس است.

Class Name
attribute <sub>1</sub>
...
attribute <sub>n</sub>
operation <sub>1</sub>
...
operation <sub>m</sub>

- اجزای یک کلاس
  - نام کلاس
  - صفات
  - عملیات

# مفاهیم پایه ...

## ❖ صفات کلاس (Attributes)

■ ویژگی هایی که یک شی از کلاس را توصیف می کنند.

■ اجزای صفت

• نام (name)

• نوع (type)

• قابلیت مشاهده (visibility)

– عمومی (public) +

– خصوصی (private) –

– محافظت شده (protected) #

Student
-name : String
-family : String
-birthDate : Date

# مفاهیم پایه ...

## ❖ عملیات کلاس (Operation)

- وظایفی که اشیای کلاس قادر به انجام آن هستند.
- عملیات در قالب متد ها پیاده سازی می شوند.
- اجزا

Student
+getName() : String +getAge(currentDate : Date) : Integer +add()

- نام (name)
- نوع بازگشتی (return type)
- آرگومان ها (arguments)
- قابلیت مشاهده (visibility)









### ▪ Operation signature

- به مجموعه اجزای یک عمل، امضای آن عمل نیز گفته می شود.

# مفاهیم پایه ...

## تعریف کلاس

```
class Student
{
    private string name,family;
    private Date birthdate;
    ...
    public Student()
    {
        ...
    }
    public string getName()
    {
        ...
    }
    ...
}
```

Student	
	name
	family
	stdnumber
	birthdate
	add()
	delete()
	getName()
	setName()

## تعریف یک شی از کلاس

class

object

Student stu=new Student();

stu.getName();

# مراحل مدل سازی

## ❖ مراحل

- شناسایی کلاس ها
- تعیین صفات هر کلاس
- تعیین عملیات مربوط به هر کلاس
- تعیین ارتباطات میان کلاس ها
- رسم نمودار

✓ به کلاس هایی که در فاز تحلیل نیازمندی ها شناسایی و تعریف می شوند، کلاس های تحلیل (Analysis Class) گفته می شود.

# شناسایی کلاس ها

❖ کلاس های تحلیل می توانند به عنوان یکی از موارد زیر شناسایی شوند:

- **نقش ها:** نقش های گوناگون که کاربر در هنگام تعامل با سیستم ایفا می کند.
  - مدیر، مشتری، فروشنده و ...
- **مکان ها:** محل های فیزیکی یا واحد های سازمانی که اطلاعات آنها برای سیستم مهم است.
  - اتاق عمل، دفتر فروش و ...
- **مفاهیم منطقی:** مفاهیمی که در منطق کاری سازمان یا سیستم استفاده می شوند.
  - سفارش، درس و ...
- **وسایل:** وسایل و دستگاه هایی که برنامه با آنها تعامل دارد.
  - حسگر، دوربین و ...
- **سیستم ها:** سیستم های خارجی که برنامه با آنها تعامل دارد.
  - بانک و ...
- **سایر موارد (چیز ها):** که بخشی از دامنه اطلاعاتی مسئله یا سیستم هستند.
  - سیگنال، گزارش، فرم، نامه، کالا و ...

## شناسایی کلاس ها ...

### ❖ روش های شناسایی کلاس های تحلیل

#### ۱. تحلیل دامنه (Domain Analysis)

با بررسی سیستم های مرتبط و مستندات آنها و همچنین صحبت با افراد خبره در زمینه سیستم مورد نظر، می توان بخشی از کلاس های کلیدی سیستم را شناسایی کرد.

#### ۲. تحلیل و تجزیه گرامری use case ها و استخراج عبارات اسمی



## شناسایی کلاس ها ...

- تحلیل و تجزیه گرامری use case ها و استخراج عبارات اسمی

### کلاس های تحلیل کاندید

هویت
مشتری
سبد خرید
سفارش
صورتحساب
هزینه
کد
وضعیت

### مورد استفاده «سفارش کالا»

ابتدا هویت مشتری بررسی می شود. اگر هویت او معتبر باشد، مشتری سبد خرید خود را مشاهده کرده و در صورت لزوم آن را بروز رسانی می کند. در صورت تایید سفارش، صورتحساب مشتری تهیه می شود و مشتری اقدام به پرداخت هزینه به صورت آنلاین می کند. در صورت پرداخت موفق، یک کد سفارش به مشتری داده می شود تا بر مبنای آن بتواند وضعیت سفارش را تا زمان تحویل آن پیگیری کند.

## شناسایی کلاس ها ...

❖ کلاس های شناسایی شده در صورتی قابل قبول هستند، که دارای ویژگی های زیر باشند:

۱. کلاس باید حاوی اطلاعاتی باشد که ذخیره سازی و نگهداری آنها برای انجام بخشی از کارکرد ها و عملیات سیستم مورد نیاز و ضروری باشد.

۲. کلاس باید شامل مجموعه ای از سرویس ها باشد که مقادیر صفات کلاس را به طریقی تغییر دهند.

۳. داشتن بیش از یک صفت: کلاس هایی که فقط یک صفت دارند، ممکن است در فاز طراحی مفید باشند؛ اما در فاز تحلیل، نیازی به در نظر گرفتن چنین کلاس هایی نیست و معمولاً به عنوان صفت برای یک کلاس دیگر نمایش داده می شوند.

۴. مشترک بودن صفات در بین همه نمونه های کلاس

۵. مشترک بودن عملیات در بین همه نمونه های کلاس

✓ در هنگام انتخاب کلاس ها، اسامی مترادف را به عنوان یک کلاس در نظر بگیرید.

✓ دامنه ها و بیان های مختلف از یک مسئله منجر به انتخاب کلاس های متفاوت می شود.

## شناسایی کلاس ها ...

کلاس های تحلیل کاندید	رد/قبول	علت
هویت	رد	هویت معمولاً شامل نام کاربری و کلمه عبور می شود که می توانند به عنوان صفت برای کلاس مشتری تعریف شوند. اما اگر بخواهیم برای کاربران آنلاین یک کلاس جداگانه تعریف کنیم، می توان صفات نام کاربری و کلمه عبور را از کلاس مشتری جدا کرد و به عنوان یک کلاس جدید با نام کلاس کاربر در نظر گرفت.
مشتری	قبول	همه موارد درست است.
سبد خرید	قبول	همه موارد درست است. اما می توان سبد خرید را مترادف با سفارش نیز در نظر گرفت.
سفارش	قبول	همه موارد درست است.
صورت حساب	قبول	همه موارد درست است.
هزینه	رد	۳ درست نیست.
کد	رد	۳ درست نیست.
وضعیت	رد	۳ درست نیست.

## تعیین صفات هر کلاس

- ❖ مطالعه use case ها و انتخاب ویژگی هایی که به طور مستقیم به هر کلاس تعلق دارند.
- ❖ کدام عناصر داده ای در دامنه مساله وجود دارند که با استفاده از آنها می توان خصوصیات کلاس را به طور کامل تعریف کرد؟
- ❖ بعضی از عبارات اسمی در use case ها که به عنوان کلاس قابل قبول نیستند، می توانند به عنوان صفت برای کلاس های دیگر در نظر گرفته شوند؛ مانند وضعیت سفارش.

صفات	کلاس تحلیل
نام کاربری، آدرس، کد پستی، تاریخ عضویت، کلمه عبور و ...	مشتری
وضعیت، تاریخ ثبت، کد سفارش، تعداد کالاها، مبلغ کل و ...	سفارش
نام، کد کالا، رنگ، مبلغ، گروه و ...	کالا
شماره، مبلغ، نام بانک، تاریخ پرداخت و ...	صورت حساب

## تعیین عملیات مربوط به هر کلاس

### ❖ انواع عملیات

- دستکاری داده ها: درج، حذف، انتخاب، بروز رسانی
  - انجام محاسبات
  - بررسی وضعیت یک شی
  - نظارت بر یک شی برای وقوع یک رویداد
- ✓ عملیات بر روی صفات کلاس انجام می شود. همچنین ممکن است برای انجام بخشی از یک عمل، از کلاس های دیگر (کلاس های همکار) استفاده شود.

### ❖ روش های شناسایی عملیات

- تحلیل و تجزیه گرامری use case ها و استخراج عبارات فعلی
    - انتساب فعل های انتخاب شده به کلاس های مرتبط
  - در نظر گرفتن برخی از عملیات بر مبنای ارتباطات میان کلاس ها
- ✓ ممکن است لازم باشد بعضی از عملیات شناسایی شده، به چند عمل فرعی تقسیم شود.

## تعیین عملیات مربوط به هر کلاس ...

- تحلیل و تجزیه گرامری use case ها و استخراج عبارات فعلی

### مورد استفاده «سفارش کالا»

ابتدا هویت مشتری بررسی می شود. اگر هویت او معتبر باشد، مشتری سبد خرید خود را مشاهده کرده و در صورت لزوم آن را بروز رسانی می کند. در صورت تایید سفارش، صورت حساب مشتری تهیه می شود و مشتری اقدام به پرداخت هزینه به صورت آنلاین می کند. در صورت پرداخت موفق، یک کد سفارش به مشتری داده می شود تا بر مبنای آن بتواند وضعیت سفارش را تا زمان تحویل آن پیگیری کند.

عملیات	کلاس های تحلیل کاندید برای انجام عمل
اعتبار سنجی هویت کاربر	مشتری
بروز رسانی سبد خرید	سفارش، سبد خرید
پرداخت هزینه سفارش	سفارش
پیگیری وضعیت سفارش	سفارش

## ارتباطات میان کلاس ها

### ❖ انواع روابط میان کلاس ها (Relationships)

- رابطه همبستگی (Association)
- رابطه تجمعی (Aggregation)
- رابطه وابستگی (Dependency)
- رابطه وراثت (Generalization)

✓ برای تعیین روابط میان کلاس ها، می توان از نمودار های توالی، همکاری و همچنین کارت های CRC بهره برد.

# ارتباطات میان کلاس ها ...

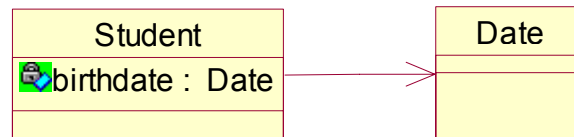
## ❖ رابطه همبستگی / انجمنی (Association)



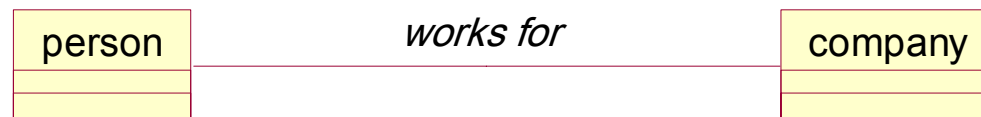
■ ارتباط بین نمونه های دو کلاس است.

■ در صورتی از این ارتباط استفاده می شود که یک نمونه از کلاس A برای انجام فعالیت هایش به اطلاعات یک نمونه از کلاس B نیاز داشته باشد.

• برای مثال، یک کلاس دارای صفتی از نوع کلاس دیگر باشد.



■ روابط همبستگی می توانند دارای نام باشند.



■ هر کدام از کلاس ها می توانند دارای یک نقش باشند.



## ارتباطات میان کلاس ها ...

### ❖ رابطه همبستگی ...

#### ■ جهت رابطه همبستگی (Navigation)

##### • همبستگی یک طرفه

– به ازای یک کاربر خاص (یک نمونه کاربر)، می توان به پسورد های او دست پیدا کرد، اما به ازای یک پسورد خاص، نمی توان کاربر صاحب آن را مشخص کرد.



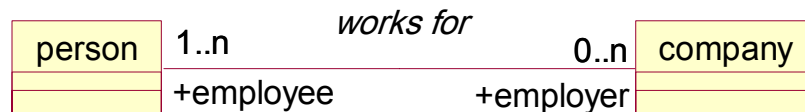
##### • همبستگی دو طرفه

– به ازای یک کارمند خاص (یک نمونه کارمند)، می توان اطلاعات شرکت هایی را که کارمند در آنها کار می کند، پیدا کرد و همچنین، به ازای یک شرکت خاص، می توان اطلاعات کارمندان آن شرکت را بدست آورد.



# ارتباطات میان کلاس ها ...

❖ رابطه همبستگی ...



■ چندی ارتباط (Multiplicity)

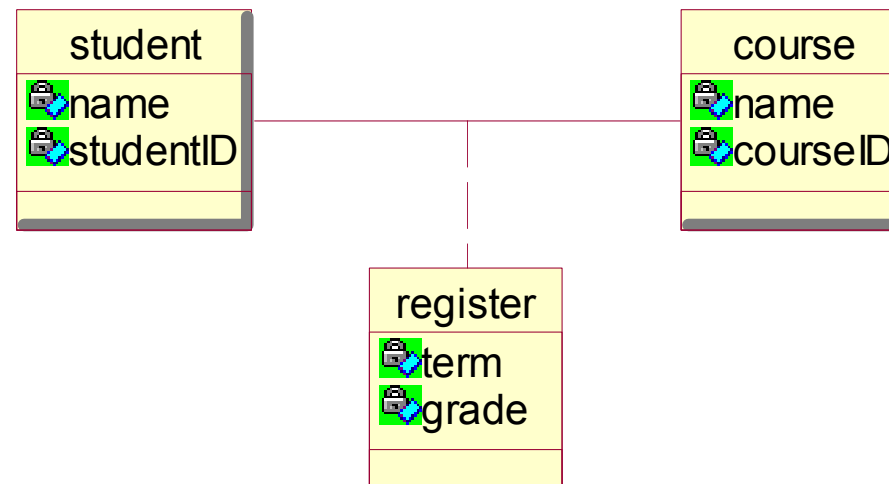
- تعداد نمونه های کلاس که در ارتباط شرکت دارند.
- ✓  $n \dots m$  :  $n$  تا  $m$  نمونه از کلاس در ارتباط شرکت دارند.

یک نمونه از کلاس A با هیچ یا فقط یک نمونه از کلاس B ارتباط دارد.		0..1
یک نمونه از کلاس A با صفر، یک یا بیشتر از یک نمونه از کلاس B ارتباط دارد (بدون محدودیت)		0..* یا 0..n یا *
یک نمونه از کلاس A دقیقا با یک نمونه از کلاس B ارتباط دارد.		1
یک نمونه از کلاس A با حداقل یک نمونه از کلاس B ارتباط دارد.		1..*

## ارتباطات میان کلاس ها ...

### ❖ کلاس همبستگی (Association Class)

- در ارتباط همبستگی بین دو کلاس، خود ارتباط ممکن است دارای صفات باشد.
- در این حالت، برای ارتباط یک کلاس تعریف می شود که به آن کلاس همبستگی گفته می شود.

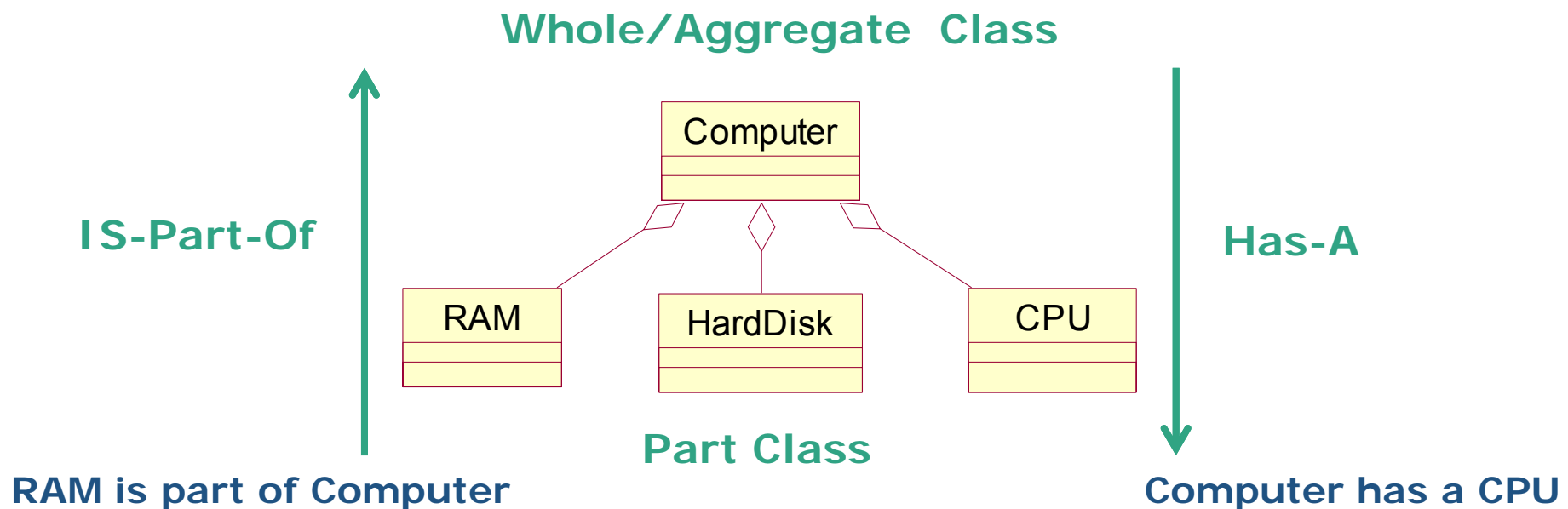


Association Class

## ارتباطات میان کلاس ها ...

### ❖ رابطه تجمعی/شمول (Aggregation)

- یک نوع خاص از ارتباط همبستگی است.
- زمانی از این رابطه استفاده می شود که یک کلاس از اجتماع تعدادی کلاس دیگر تشکیل شده باشد.
- به عبارت دیگر، کلاس کل (کلاس مجتمع)، شامل کلاس های جزء است.

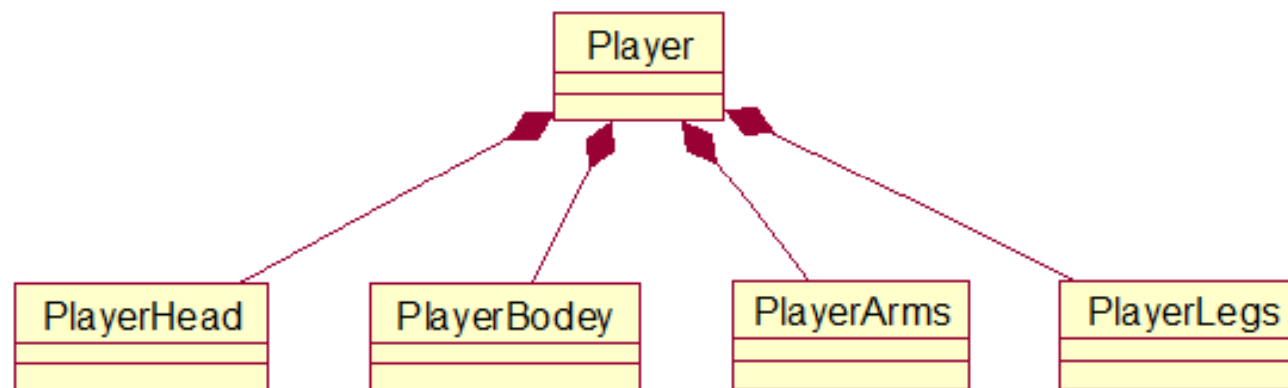


## ارتباطات میان کلاس ها ...

❖ رابطه تجمعی ...

### ▪ رابطه ترکیب (Composition)

- یک نوع خاص از رابطه تجمعی است.
- بین نمونه های کلاس جزء و کلاس کل وابستگی وجود دارد.
  - اگر شی کل حذف شود، شی جزء نیز باید حذف شود.
  - شی جزء فقط متعلق به یک شی کل است.



# ارتباطات میان کلاس ها ...

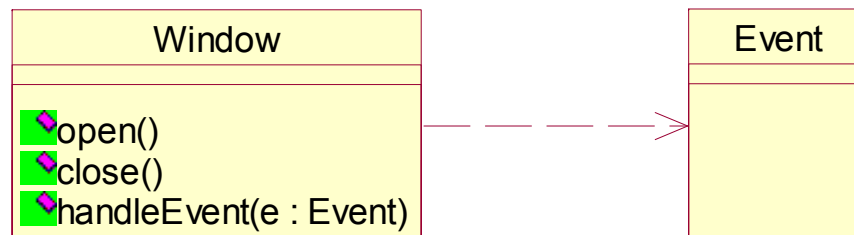
## ❖ رابطه وابستگی (Dependency)

- کلاس A به کلاس B وابسته است اگر کلاس A برای انجام بعضی از عملیاتش از کلاس B استفاده کند. در نتیجه، تغییر در کلاس B ممکن است منجر به تغییر در کلاس A شود.



- رابطه همبستگی به طور خودکار یک رابطه وابستگی نیز هست. اگر از قبل، بین دو کلاس رابطه همبستگی رسم شده باشد، دیگر نیازی به رسم رابطه وابستگی بین آن دو کلاس نیست.

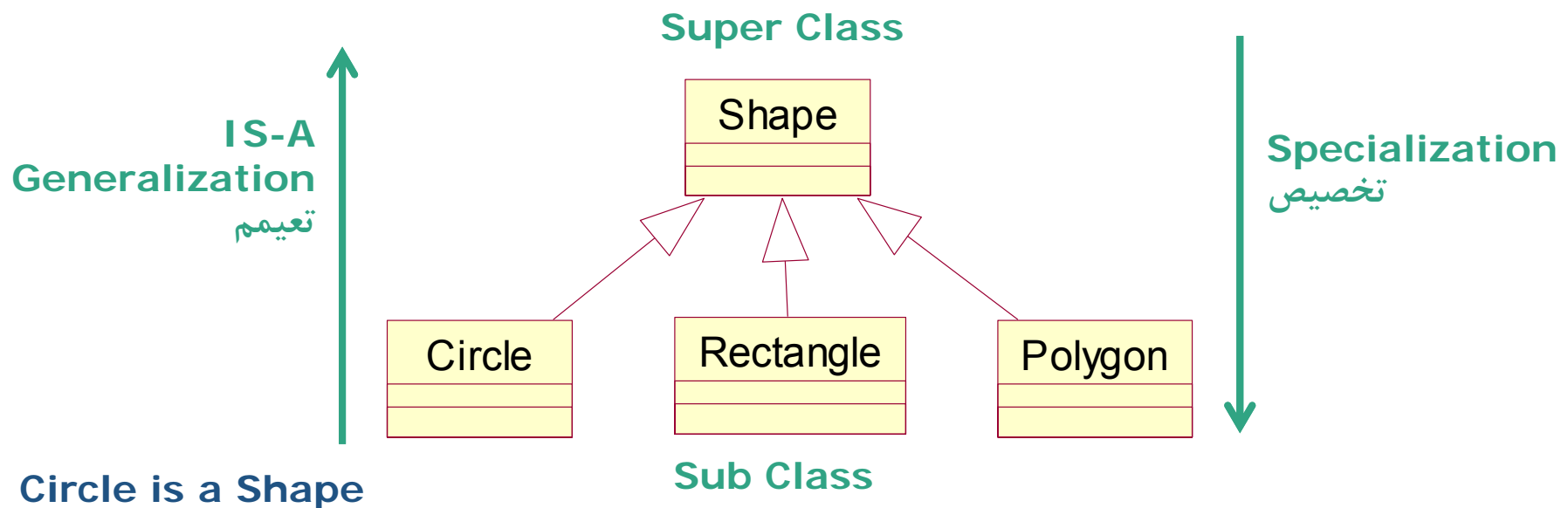
- معمولاً زمانی رابطه وابستگی بین دو کلاس رسم می شود که کلاس حاوی اتصال دراز مدت با کلاس دیگر نیست، اما گاهی از آن کلاس استفاده می کند. برای مثال، یکی از توابع کلاس A دارای آرگومانی از نوع کلاس B باشد؛ یا در یکی از توابع A، یک متغیر محلی از نوع کلاس B تعریف شده باشد.



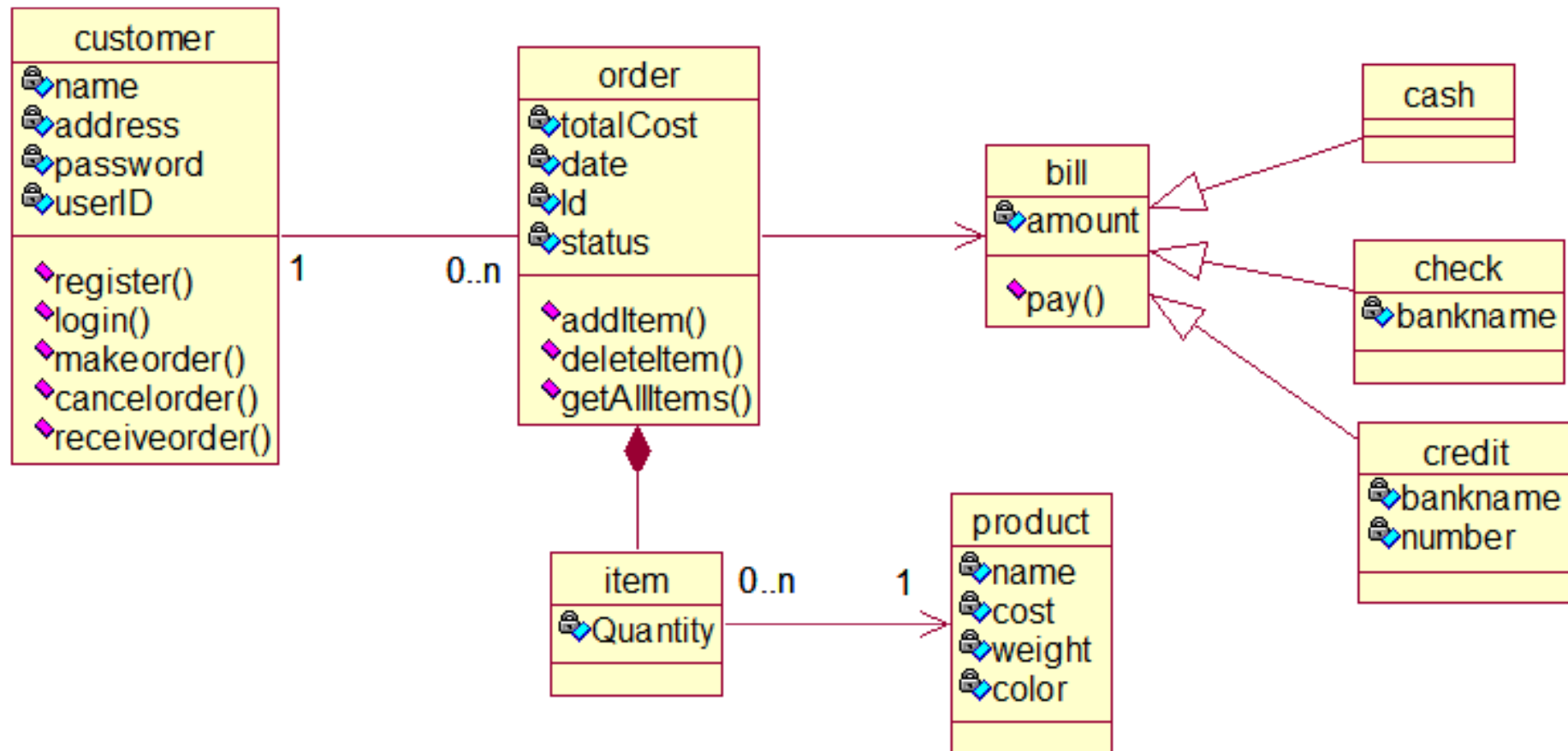
## ارتباطات میان کلاس ها ...

### ❖ رابطه Generalization

- یک رابطه بین یک کلاس عام (ابر کلاس) و یک کلاس خاص (زیر کلاس)
- کلاس خاص، صفات و / یا عملیات کلاس عام را به ارث (inheritance) می برد. علاوه براین، دارای صفات و عملیات اضافی (خاص خودش) است.
- یک نمونه از زیر کلاس، یک نمونه از ابر کلاس نیز می باشد.

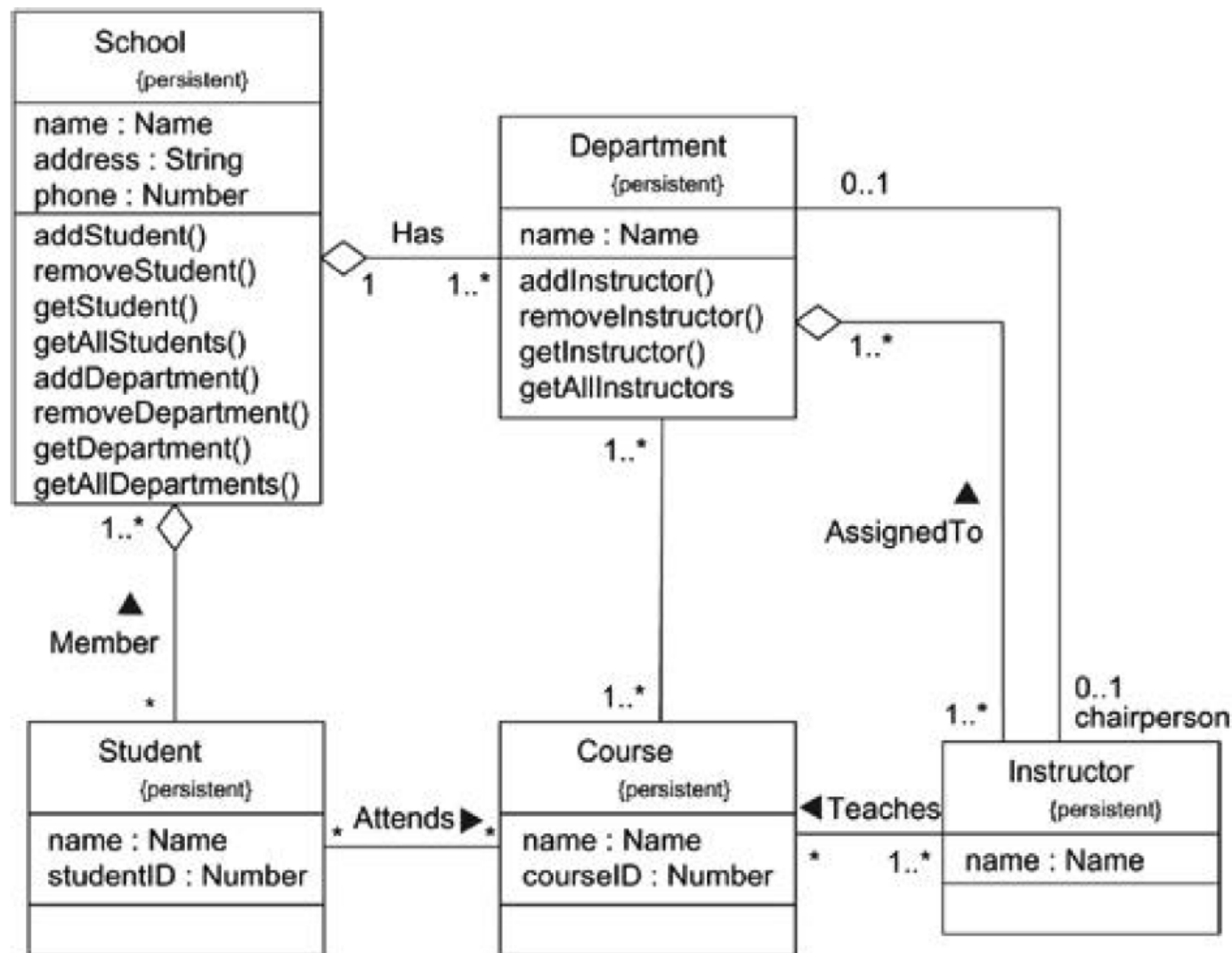


# رسم نمودار





# رسم نمودار ...

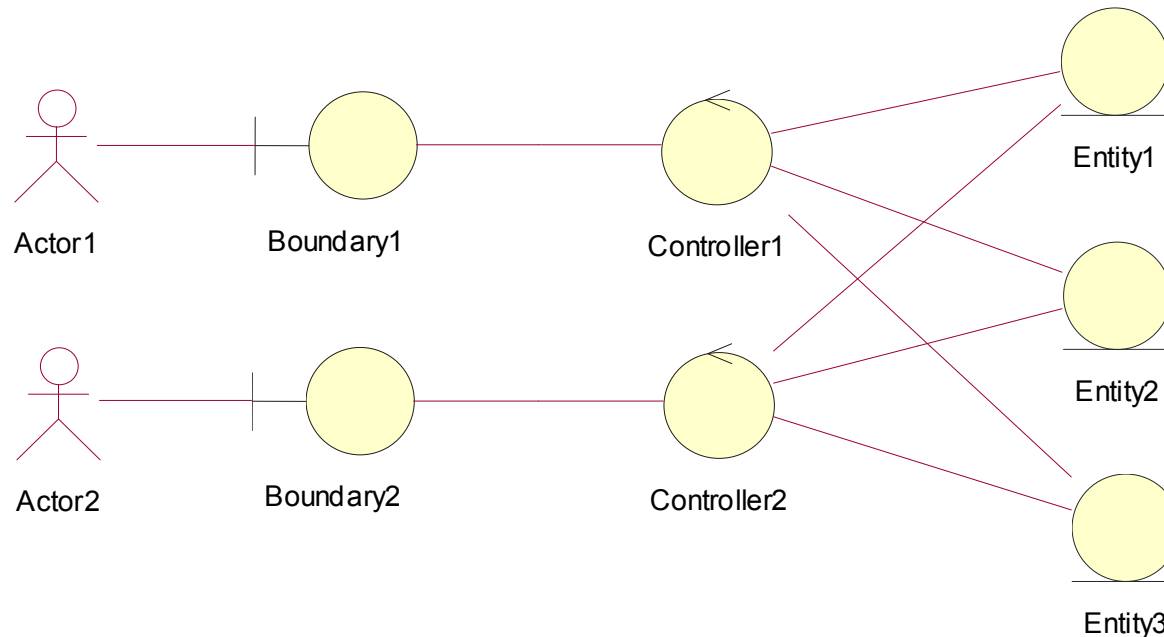


# انواع کلاس ها (Stereotype)

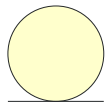
❖ در هنگام مدل سازی، می توان کلاس ها را به سه دسته کلی تقسیم کرد:

- کلاس های موجودیتی *Entity Class*
- کلاس های مرزی *Boundary Class*
- کلاس های کنترل کننده *Controller Class*

✓ می توان به جای استفاده از نماد عمومی کلاس، به ازای هر نوع کلاس از نماد خاص همان نوع استفاده کرد.



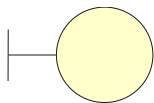
# انواع کلاس ها ...



EntityClass

## ❖ کلاس های موجودیتی

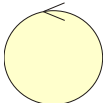
- این کلاس ها به طور مستقیم از دامنه مساله قابل شناسایی هستند.
- معمولا اشیایی را شامل می شوند که اطلاعات آنها می بایست در فایل ها یا پایگاه داده ها ذخیره شود و در طول مدت استفاده از نرم افزار ماندگار (persistent) باشد.



BoundaryClass

## ❖ کلاس های مرزی

- کلاس هایی هستند که واسطه های قابل مشاهده توسط کاربر را ایجاد و مدیریت می کنند؛ و کاربر در هنگام تعامل با نرم افزار، با آنها ارتباط برقرار می کند.
- مانند: پنجره های صفحه نمایش، گزارش ها، فرم های ورودی و خروجی و ...



ControllerClass

## ❖ کلاس های کنترل کننده

- کنترل، ایجاد و بروز رسانی اشیای کلاس های موجودیتی، تبادل اطلاعات میان اشیای کلاس های مرزی و موجودیتی، اعتبار سنجی داده های تبادل شده میان اشیا و یا میان کاربر و برنامه
- این کلاس ها معمولا در فاز طراحی شناسایی و تعریف می شوند.

# نکات

## ❖ نکات

- برای ساده سازی نمودار کلاس ها می توان مجموعه ای از کلاس ها را در یک بسته قرار داد.
- در نمودار کلاس ها، می توان تنها توابع و صفات مهم را نمایش داد.
- انتخاب و تعریف کلاس ها یک رویکرد مبتنی بر تکرار است و تا مرحله طراحی نیز ادامه دارد.

## ❖ مزایا

- از نمودار کلاس در مرحله ساخت و تولید کد استفاده می شود.

## ❖ نقاط ضعف

- تمام ارتباطات در این نمودار نمایش داده می شوند اما این که در هر ارتباط چه اتفاقی رخ می دهد مشخص نمی شود.
- عدم نمایش ماهیت پویای کلاس ها و چگونگی رفتار آنها در هنگام ارتباط با یکدیگر

**CRC** کارت های

**CLASS**

**RESPONSIBILITY**

**COLLABORATOR**

# کارت های CRC

❖ ابزاری ساده برای شناسایی و سازماندهی کلاس های مرتبط با نیازمندی های سیستم

❖ مجموعه ای از کارت های شاخص استاندارد

❖ هر کارت CRC شامل سه بخش اصلی می باشد:

▪ نام کلاس (class name)

▪ مسئولیت های کلاس (responsibilities)

▪ همکاران کلاس (collaborators)

Class Name	
Responsibility	Collaborator

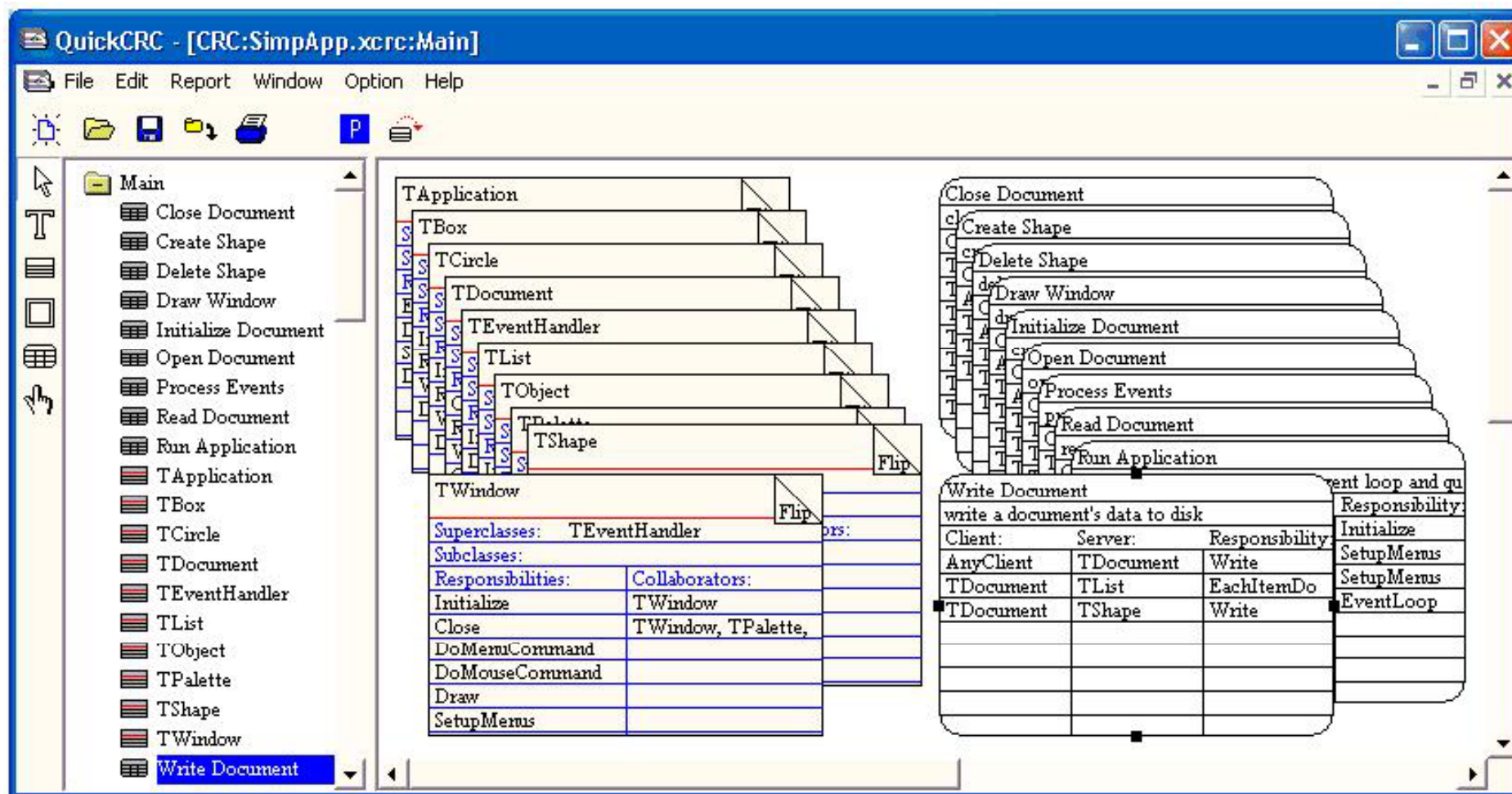
# کارت های CRC ...

<u>Drawing</u> Holds Figures.  Accumulates updates, refreshes on demand.	Figure Drawing View Drawing Controller
--	--

Class: FloorPlan	
Description	
Responsibility:	Collaborator:
Defines floor plan name/type	
Manages floor plan positioning	
Scales floor plan for display	
Scales floor plan for display	
Incorporates walls, doors, and windows	Wall
Shows position of video cameras	Camera

Student	
Super Classes: Person	
Sub Classes:	
Responsibility	Collaborator
Student Name	
Student ID	
Register Course	Course
Drop Course	Course

# کارت های CRC ...





# مسئولیت های کلاس

## ❖ مسئولیت های کلاس

■ هر چیزی که کلاس می داند یا انجام می دهد.

■ صفات کلاس + عملیات کلاس

صفات	{	Student	
		Responsibility	Collaborator
		Student Name	
		Student ID	
عمليات	{	Register Course	
		Drop Course	

# مسئولیت های کلاس ...

➤ اصول تخصیص مسئولیت ها به کلاس ها

▪ هوشمندی سیستم باید به نحوی در میان کلاس ها توزیع شود که به بهترین وجه پاسخگوی نیاز های مساله باشد.

- **هوشمندی سیستم:** آنچه که سیستم می داند و آنچه که سیستم قادر به انجام آن است
- روش های توزیع هوشمندی (مسئولیت ها) بین کلاس ها

## ۱. توزیع غیر یکنواخت

- کلاس های خینگ (dumb) و کلاس های باهوش (smart)
- همه هوشمندی در چند کلاس خاص متمرکز می شود.

## ۲. توزیع یکنواخت

- تعداد مسئولیت ها به طور تقریبا یکسان بین کلاس ها توزیع شود.
- مسئولیت های یک کلاس، در سطح انتزاع یکسانی قرار داشته باشند.
- کمک به یکپارچگی سیستم و افزایش قابلیت نگهداری

# مسئولیت های کلاس ...

➤ اصول تخصیص مسئولیت ها به کلاس ها ...

- اطلاعات و رفتار مرتبط با آن، باید در یک کلاس قرار داشته باشد.
  - رعایت اصل پنهان سازی اطلاعات در متدولوژی شی گرا
- اطلاعات مربوط به یک چیز باید تنها در یک کلاس قرار داشته باشد و نباید در میان چند کلاس توزیع شده باشد.
  - افزایش قابلیت تست و نگهداری نرم افزار
- بعضی از مسئولیت ها را در صورت لزوم باید در میان کلاس های مرتبط به اشتراک گذاشت.
- مسئولیت های کلی/عمومی (هم صفات و هم عملیات) باید در بالای سلسله مراتب کلاس ها قرار داده شوند تا در زیر کلاس ها بتوان از آنها استفاده کرد.

## همکاران کلاس

❖ یک کلاس ممکن است در انجام بخشی از مسئولیت هایش به همکاری کلاس های دیگر نیاز داشته باشد.

❖ انواع همکاری

- درخواست برای دریافت اطلاعاتی که کلاس آنها را در اختیار ندارد.
- درخواست برای بروز رسانی اطلاعاتی که کلاس آنها را در اختیار ندارد.
- درخواست برای انجام عملی که از مسئولیت های یک کلاس دیگر است.

❖ کلاس همکار: کلاسی که اطلاعات لازم برای انجام یکی از مسئولیت های کلاس را فراهم می کند.

✓ هر گونه همکاری بین دو کلاس، معادل با یک نوع ارتباط میان آنها است که در نمودار کلاس نمایش داده می شود.

❖ در کارت CRC، نام کلاس همکار در مقابل مسئولیتی نوشته می شود که به همکاری آن کلاس نیاز دارد.

# کارت های CRC ...

Customer	
Responsibility	Collaborator
Name	
Address	
Make Order	Order
Cancel Order	Order
Get Name	

OrderItem	
Responsibility	Collaborator
Quantity	
Product	
Calculate cost	Product

Order	
Responsibility	Collaborator
Order ID	
Date ordered	
Order items	
Calculate order total cost	OrderItem

Product	
Responsibility	Collaborator
Name	
Cost	
Color	
Show specification	
Get Cost	

# بازبینی کارت ها

❖ اجرای سناریو « what if ... ? »

- اعضای تیم بازبینی در دور یک میز جمع می شوند.
- به هر یک از اعضا یک مجموعه از کارت های CRC داده می شود.
- ✓ یک فرد نباید دارای دو کارت کلاس باشد که با هم همکاری دارند.
- مدیر تیم بازبینی شروع به خواندن یک use case می کند. هر زمان که مدیر تیم به یک شی نام گذاری شده برسد، فردی که کارت کلاس مربوط به آن شی را در دست دارد، فرآیند خواندن را ادامه می دهد.
- ✓ در این زمان، گروه تیم بازبینی بررسی می کنند که آیا کلاس دارای مسئولیتی هست که پاسخگویی نیازمندی مطرح شده در use case باشد یا خیر؟
- ✓ در صورتی که در کنار مسئولیت مورد نظر، نام یک کلاس همکار نوشته شده باشد، ادامه کار به فردی سپرده می شود که کارت کلاس همکار در دست اوست.

## بازبینی کارت ها ...

❖ اجرای سناریو « what if ... ? » ...

▪ اگر مسئولیت ها و همکاری های تعیین شده در کارت های CRC قادر به پاسخگویی به نیازمندی های **use case** نباشند، اصلاحاتی در کارت ها اعمال می شود؛ مانند تعریف کلاس های جدید، بازبینی مشخصات یک مسئولیت یا همکاری در کارت های موجود و ...

▪ این کار آن قدر ادامه پیدا می کند تا خواندن **use case** به اتمام برسد.

□ بازبینی کارت های CRC به ازای هر **use case** انجام می شود.

□ این روش، راه مناسبی برای پیدا کردن خطاها یا کاستی ها در مدل کلاس ها است.

نمودار اشیا

**OBJECT DIAGRAM**



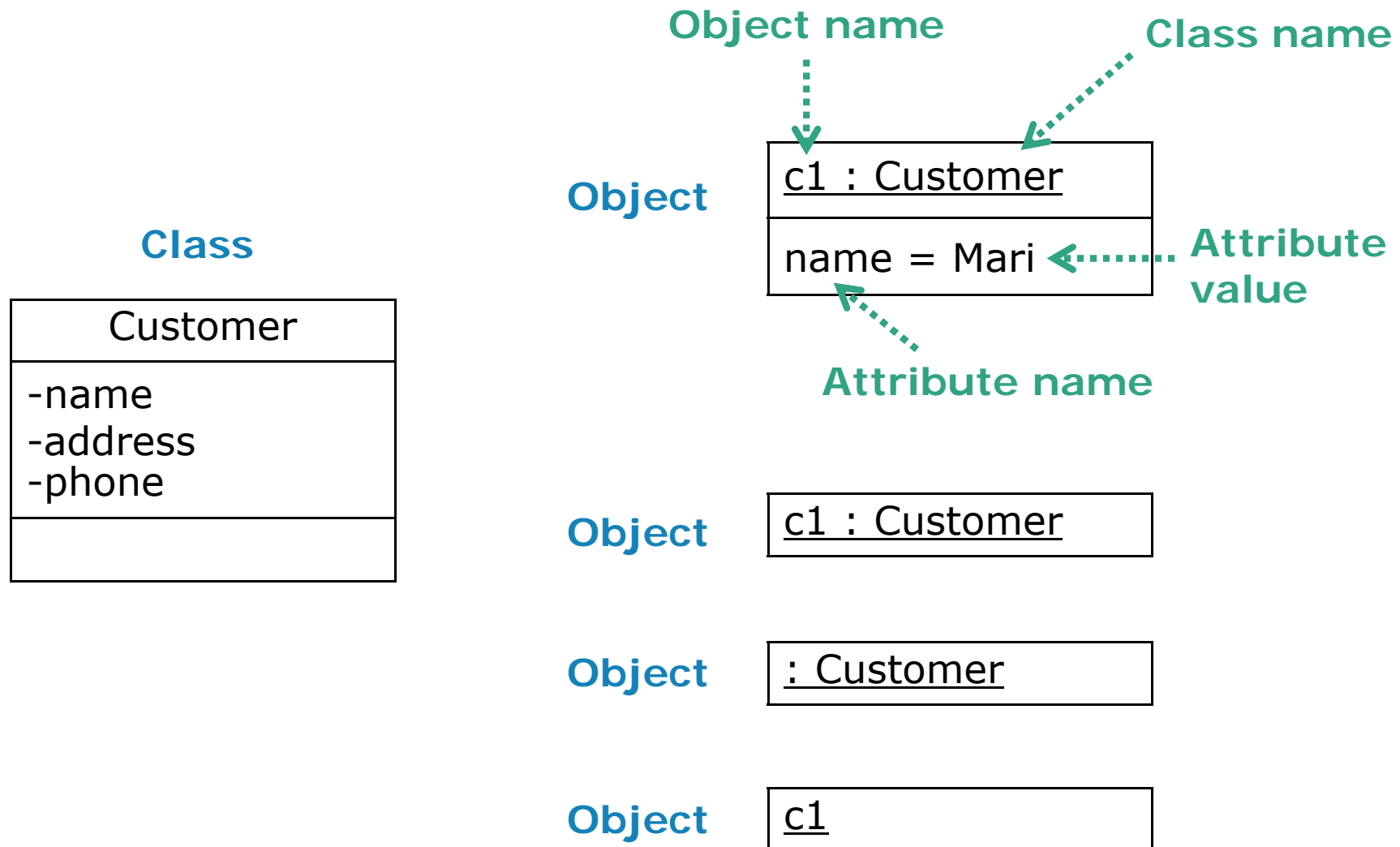
# نمودار اشیا (Object Diagram)

- ❖ نمودار اشیا یک نمونه (instance) از نمودار کلاس است.
- ❖ نمایش یک دید ثابت (ایستا) از ساختار سیستم در یک لحظه از زمان اجرا
- ❖ نمایش مجموعه ای از اشیا و روابط بین آنها

## ❖ اجزای نمودار

- شی (Object) : نمونه ای از یک کلاس
- پیوند (Link) : ارتباط بین دو شی

# نحوه نمایش یک شی



## مراحل مدل سازی

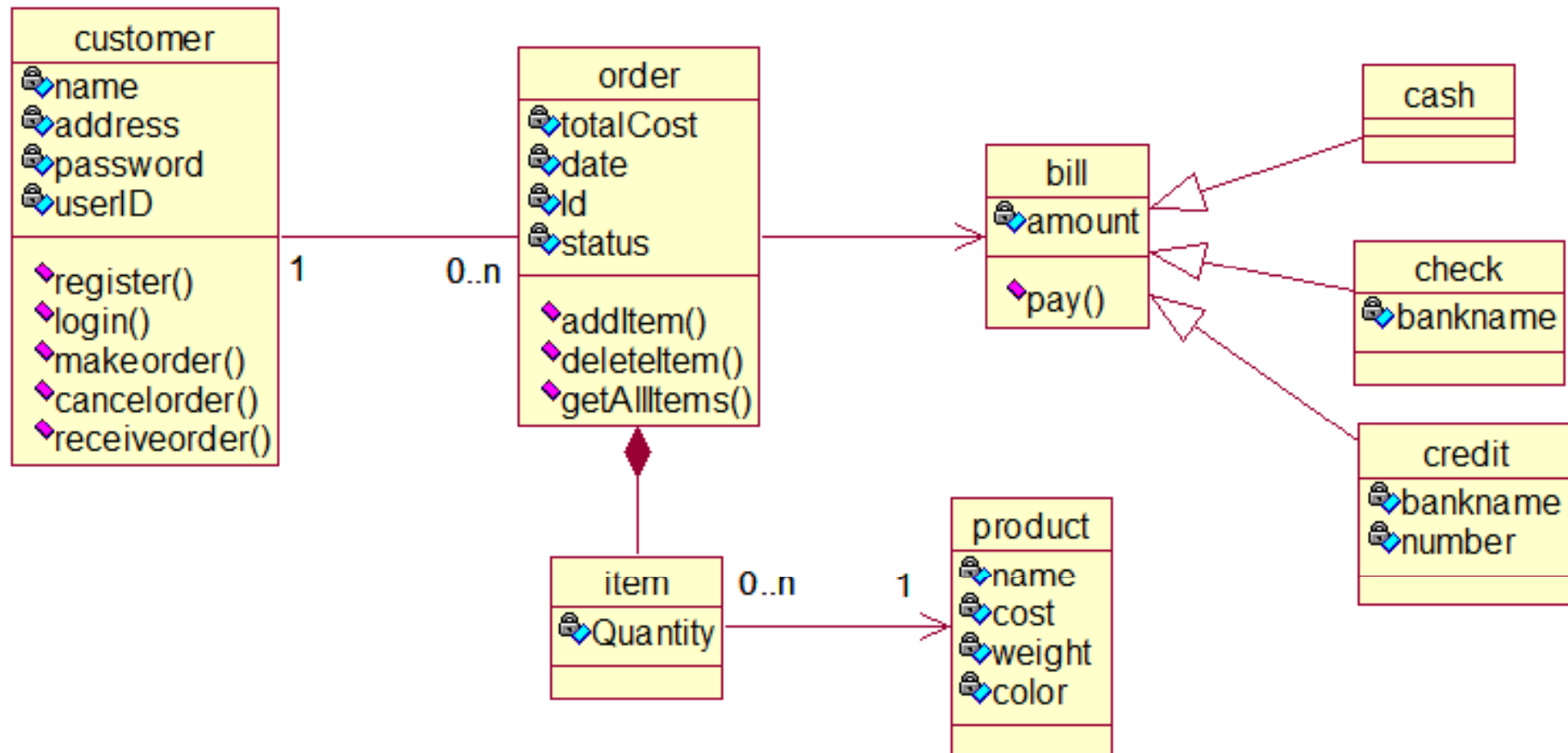
- ✓ نمودار اشیا بر اساس نمودار کلاس بدست می آید.
- ✓ به ازای یک سیستم، تعداد نمودار های کلاس محدود است اما تعداد نامحدودی نمودار اشیا می تواند وجود داشته باشد.
- ✓ یک نمودار اشیا نمی تواند همه اشیا سیستم را نمایش دهد.

### ❖ مراحل مدل سازی اشیا

- انتخاب توابع یا بخشی از سیستم که می خواهید رفتار آن را در زمان اجرا مدل کنید.
- انتخاب کلاس ها و روابطی که قرار است در مدل نمایش داده شوند.
- یک سناریو از اجرای عملکرد را در نظر بگیرید و به ازای یک لحظه از اجرای آن، اشیا و روابط بین آنها را مشخص کنید.
- مشخص کردن مقادیر صفات و وضعیت اشیا
- تعیین لینک های میان اشیا

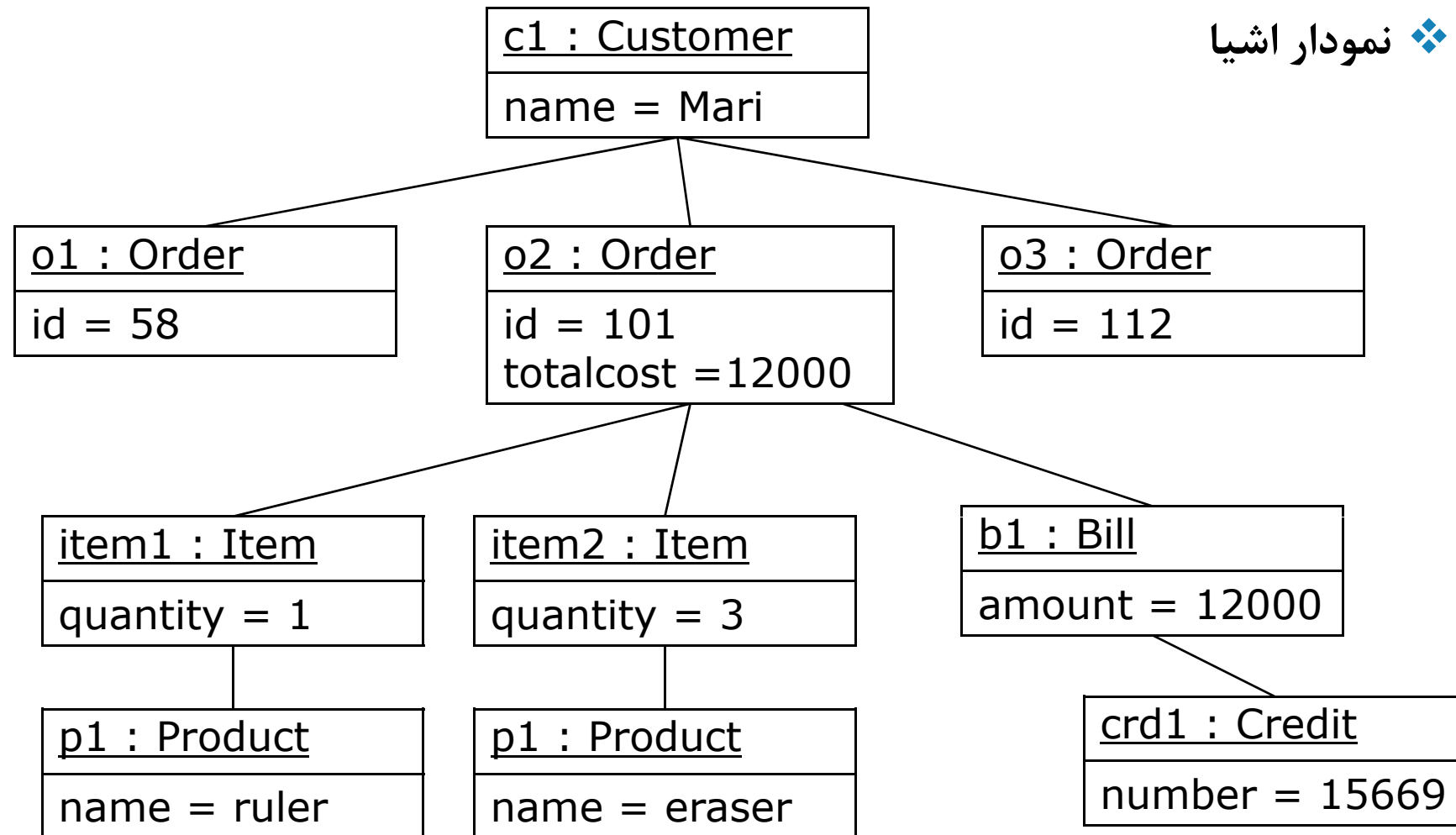
# مراحل مدل سازی ...

❖ نمودار کلاس



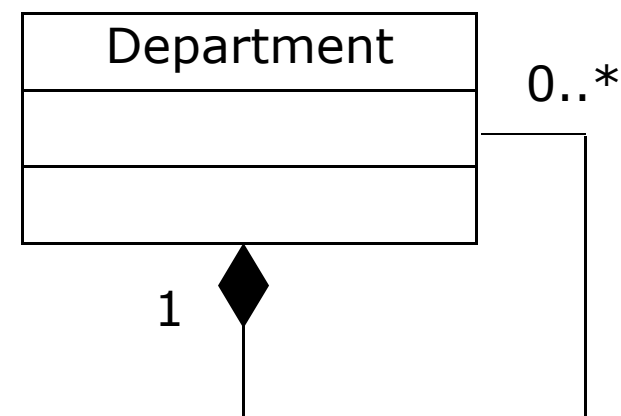
## مراحل مدل سازی ...

❖ نمودار اشیا

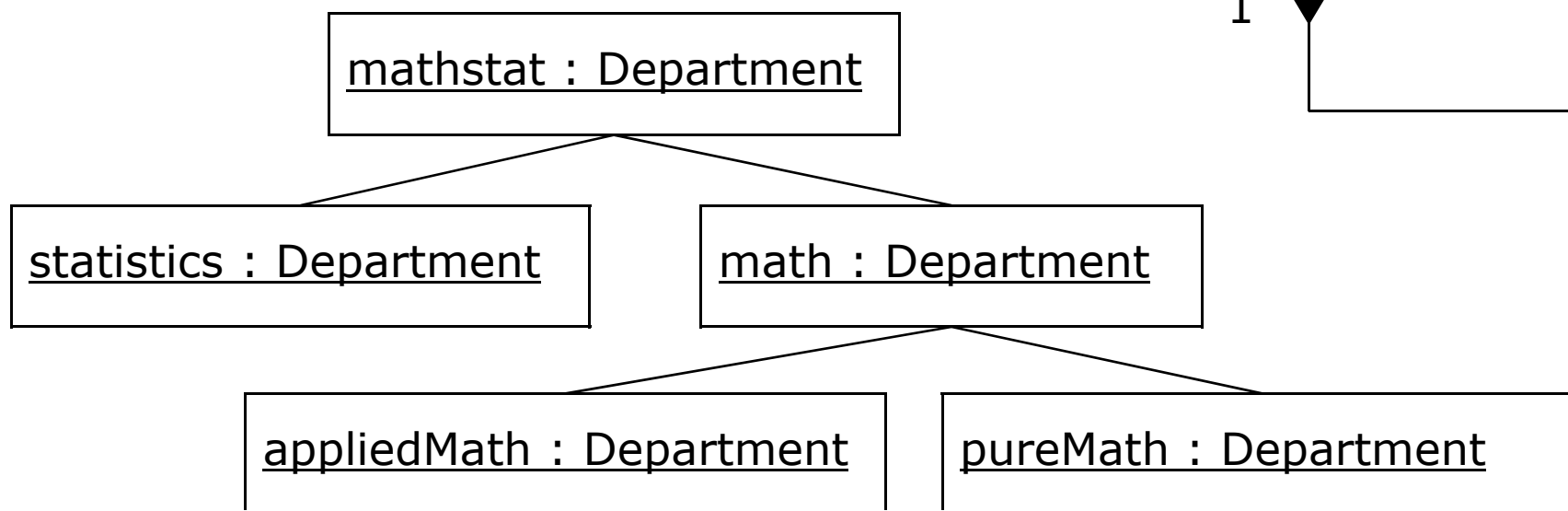


# مراحل مدل سازی ...

نمودار کلاس



نمودار اشیا



## نمودار اشیا ...

- نمودار کلاس یک مدل انتزاعی از کلاس ها و روابط بین آنها را نمایش می دهد اما نمودار اشیا یک نمونه واقعی از کلاس ها را در یک لحظه از عملکرد سیستم نشان می دهد.
- نمودار اشیا در مقایسه با نمودار کلاس به رفتار واقعی سیستم نزدیک تر است.
- نمودار اشیا به فهم بهتر روابط بین کلاس ها و چگونگی پیاده سازی آنها کمک می کند.
- نمودار اشیا معمولا برای مدل کردن بخش هایی از کد یا نمودار کلاس که دارای پیچیدگی نسبتا بالایی هستند، به کار می رود.

# نمودار های تعاملی

## INTERACTION DIAGRAMS



# نمودار های تعاملی (Interaction Diagrams)

❖ نمودار تعاملی شامل مجموعه ای از اشیا و پیام های مبادله شده میان آنها می باشد.

- نمایش تعامل میان اشیا در هنگام انجام یک عمل

- مدل سازی رفتار پویای سیستم

- در رفتار پویای سیستم، نحوه و ترتیب ارتباط اجزای مختلف با یکدیگر برای انجام وظایف سیستم نمایش داده می شود.

- رفتار پویای سیستم مبتنی بر ساختار ایستای آن است.

## ❖ انواع نمودار تعاملی

- نمودار توالی

- نمودار همکاری

✓ نمودار های توالی و همکاری به راحتی و بدون از دست رفتن اطلاعات قابل تبدیل به هم هستند (isomorphic)

## تبادل پیام میان اشیا (Message Passing)

- ✓ پیام (Message): ارسال یک درخواست از طرف سرویس گیرنده به سرویس دهنده برای انجام یک عمل
- ✓ تبادل پیام میان اشیا از طریق فراخوانی عملیات آنها (Calling Operation) انجام می شود.
- ✓ درخواست باید مطابق با امضای عمل (Operation Signature) ارسال شود.

## تبادل پیام میان اشیا ...

Class A

{

...

Func1()

{

B objectB=new B();

objectB.Func2();

}

...

}

A objectA=new A();

objectA.Func1();

Message Passing

Func2()

objectA : A

objectB : B

Client

Server

# نمودار توالی

## SEQUENCE DIAGRAM

# نمودار توالی (Sequence Diagram)

❖ نمودار توالی (ترتیب) یک نمودار تعاملی است که بر روی ترتیب زمانی تبادل پیام ها تاکید دارد.

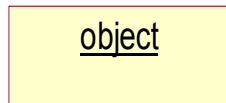
- عملیات چگونه انجام می شوند.
- چه پیام هایی برای انجام عملیات ارسال می شوند. (محور افقی: از چپ به راست)
- پیام ها با چه ترتیب زمانی ارسال می شوند. (محور عمودی: از بالا به پایین)

❖ چگونگی گذر از یک شی به شی دیگر

❖ نمایش جریان عملیات در Use Case ها

# نمودار توالی ...

## ❖ اجزای نمودار



- شی (object) // البته می تواند کلاس یا هر چیز دیگری نیز باشد

- پیام ارسالی به یک شی (object message)

- پیام بازگشتی از یک شی (return message)

- خط حیات (Lifeline)

- یک خط مقطع که از زمان ایجاد شی تا زمان تخریب آن ادامه دارد.

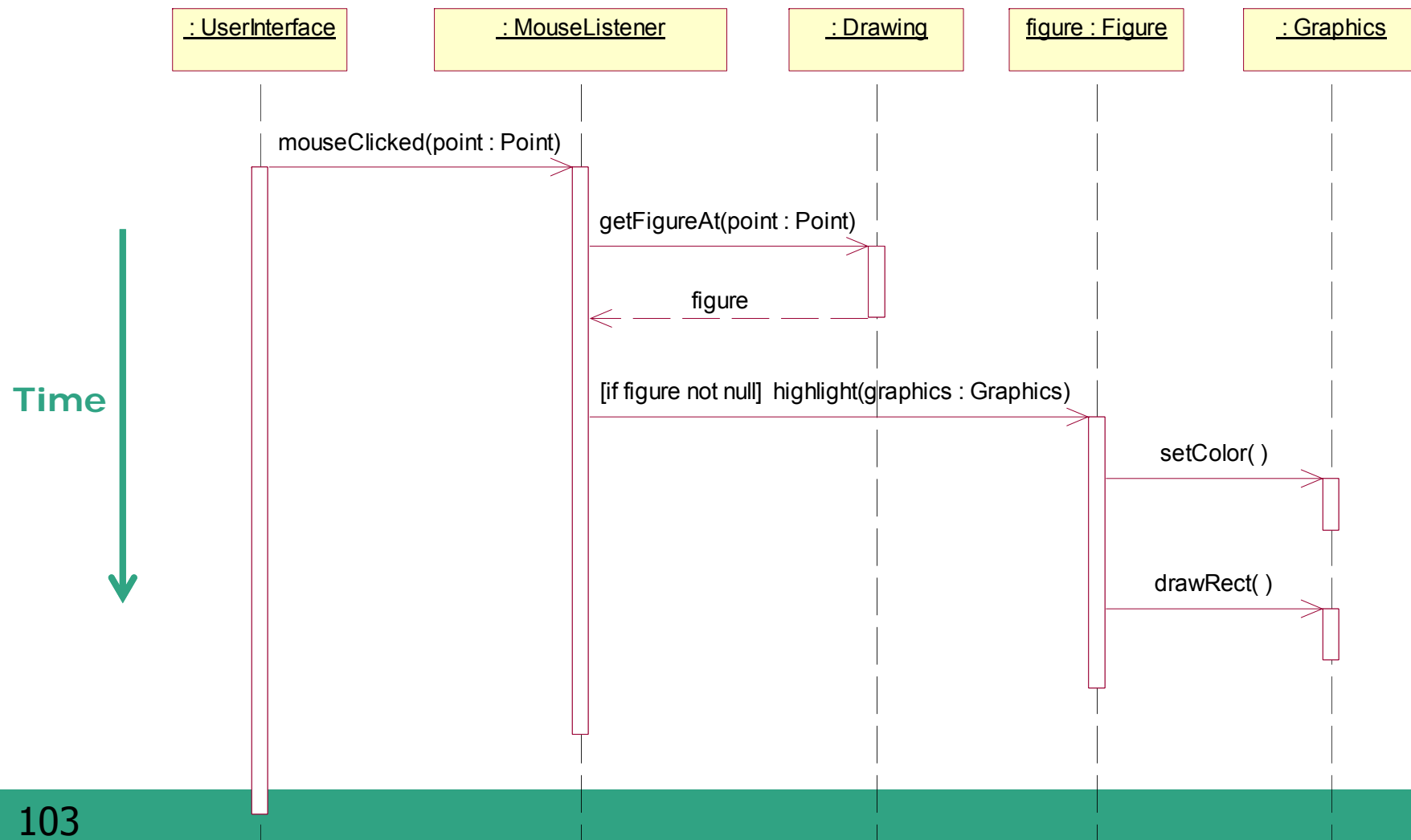
- نوار فعال سازی (focus of control)

- یک مستطیل سفید رنگ است که مدت اجرای یک عملکرد توسط یک شی را (از بالا به پایین) نشان می دهد.

- به ازای هر فراخوانی تابع، یک نوار فعال سازی رسم می شود.

# نمودار توالی ...

usecase: با فرض اینکه در محیط گرافیکی، مجموعه ای از اشکال رسم شده است؛ کاربر می تواند با کلیک بر روی یک شکل، آن را برجسته کرده و در حالت انتخاب قرار دهد.



# نمودار توالی ...

```
Class MouseListener
```

```
{  
    ...  
    void mouseClicked (Point point)  
    {  
        ...  
        Figure fig=drawing.getFigureAt(point);  
        if(fig!=null)  
        {  
            fig.highlight(graphics);  
        }  
        ...  
    }  
    ...  
}
```

```
Class Figure
```

```
{  
    ...  
    highligh(Graphics g)  
    {  
        ...  
        g.setColor(...);  
        ...  
        g.drawRect(...);  
        ...  
    }  
}
```

```
Class Graphics
```

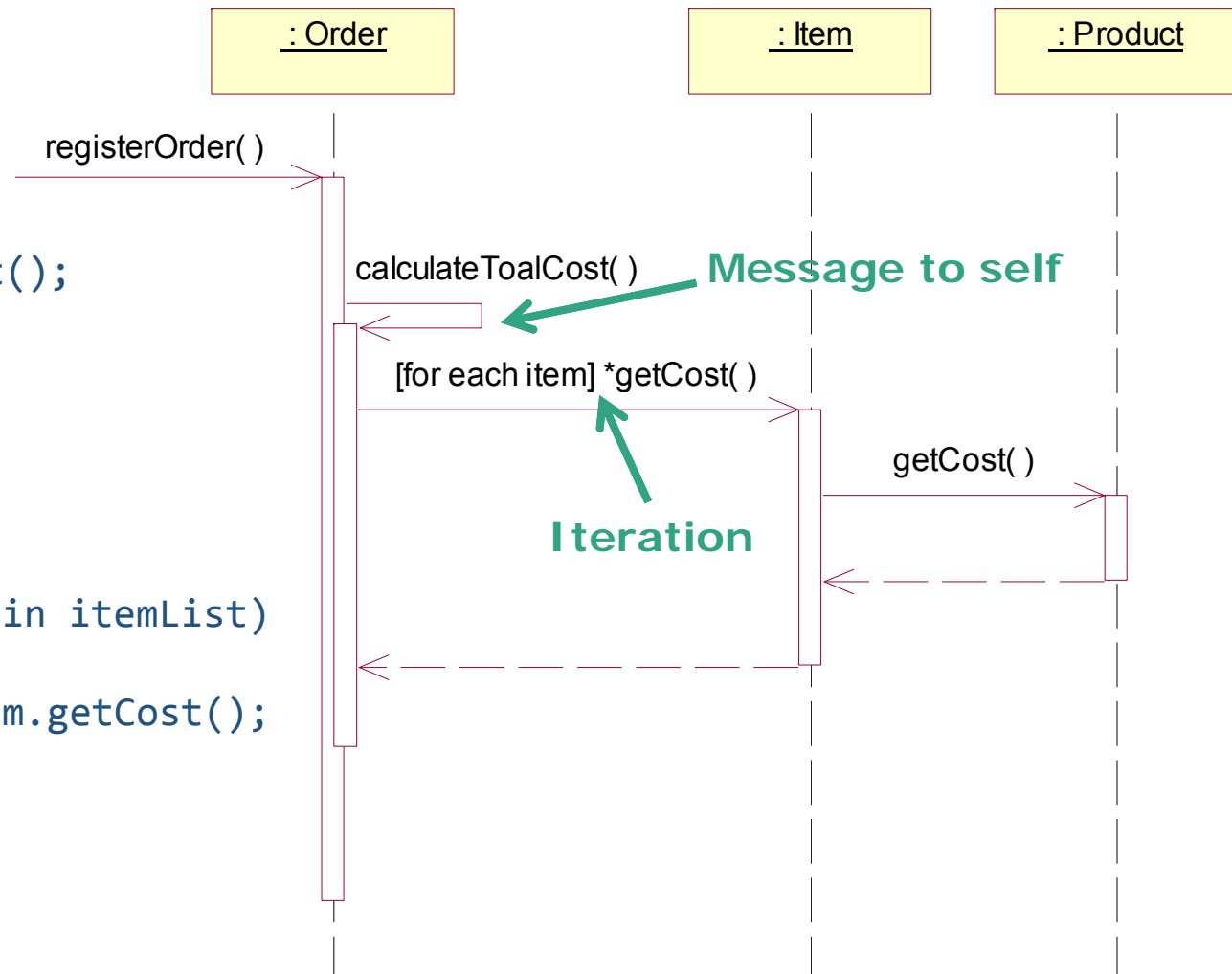
```
{  
    ...  
    void setColor (Color c) { ... }  
    void drawRect(int x, int y, int w, int h){ ... }  
    ...  
}
```



# نمودار توالی ...

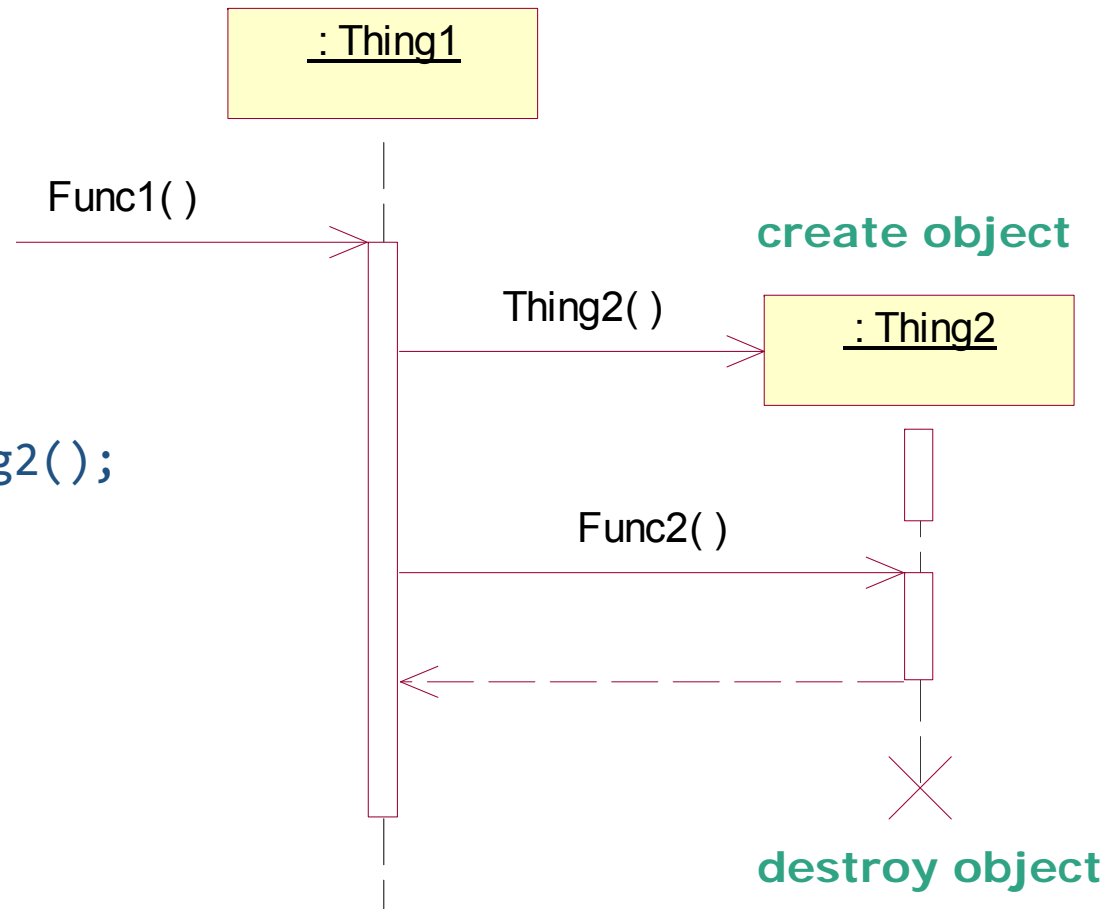
Class Order

```
{  
  ...  
  registerOrder()  
  {  
    ...  
    calculateTotalCost();  
    ...  
  }  
  calculateTotalCost()  
  {  
    ...  
    long totalcost=0;  
    foreach(Item item in itemList)  
    {  
      totalcost+=item.getCost();  
    }  
    ...  
  }  
  ...  
}
```

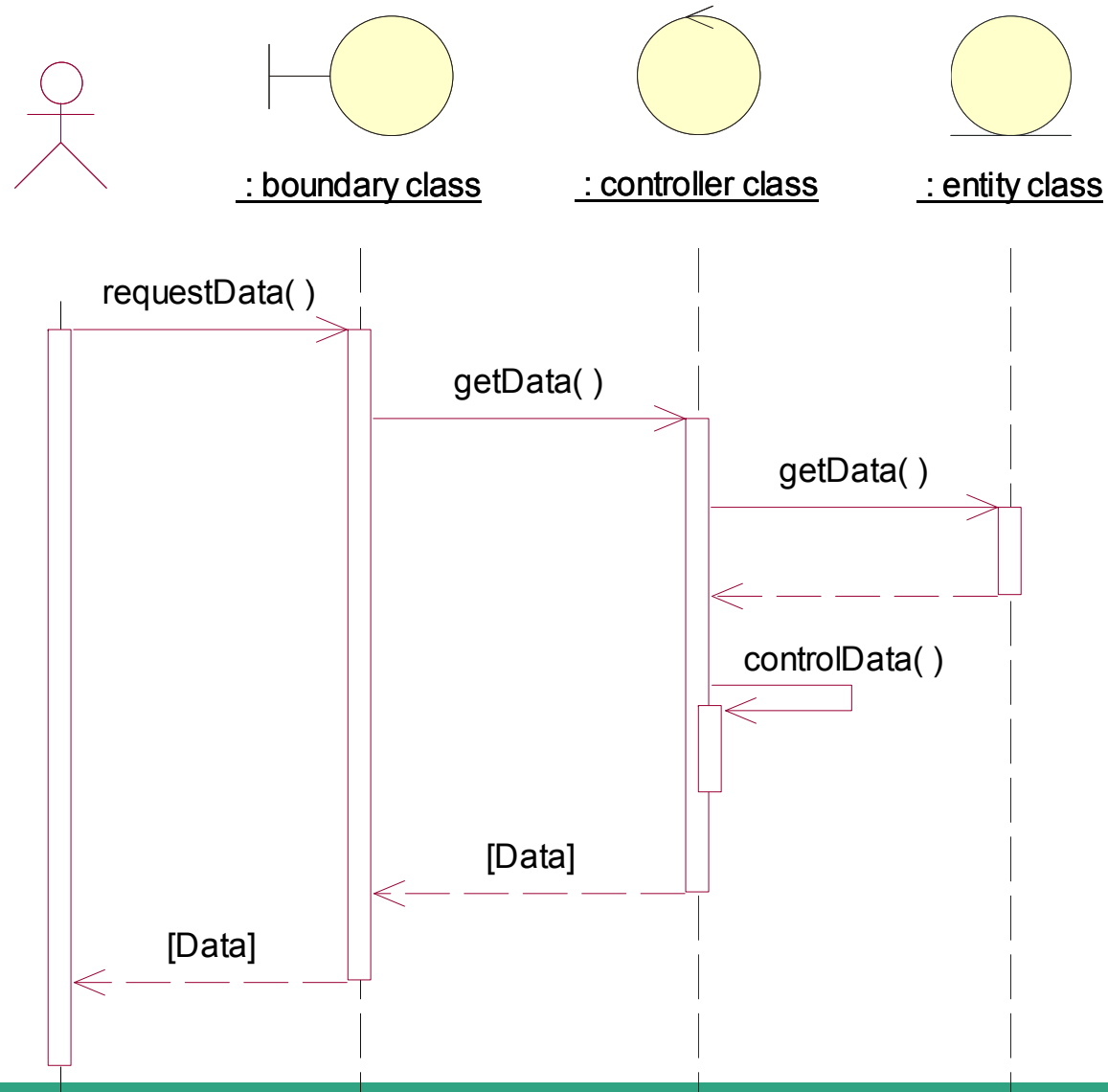


# نمودار توالی ...

```
Class Thing1
{
    ...
    Func1()
    {
        ...
        Thing2 obj2 =new Thing2();
        ...
        obj2.Func2();
        ...
    }
    ...
}
```



# نمودار توالی ...



# نمودار همکاری

## COLLABORATION DIAGRAM

# نمودار همکاری (Collaboration Diagram)

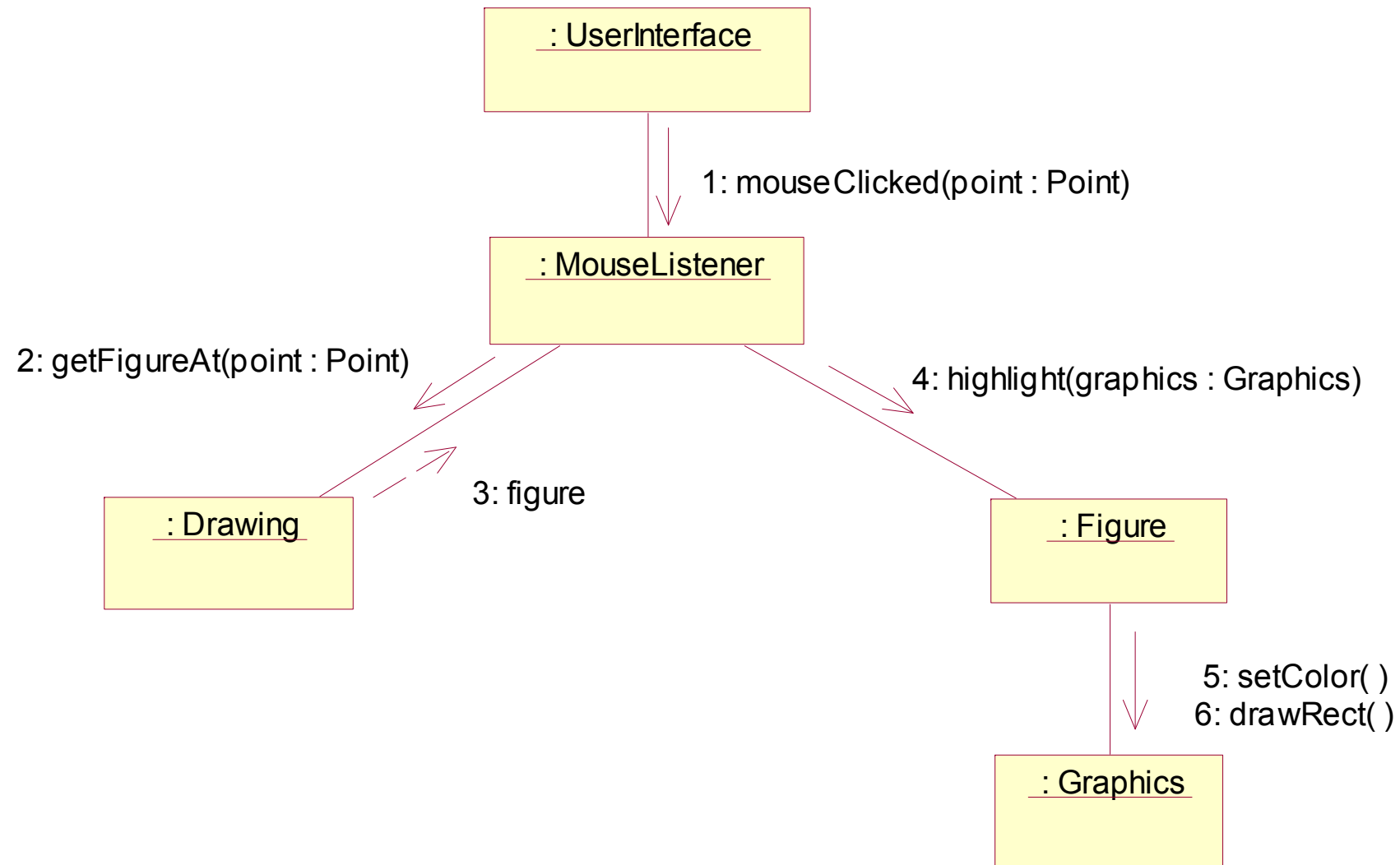
❖ نمودار همکاری (ارتباطات) یک نمودار تعاملی است که بر نحوه سازماندهی اشیا در  
حین تبادل پیام تاکید دارد.

❖ نمایش چگونگی همکاری میان اشیا

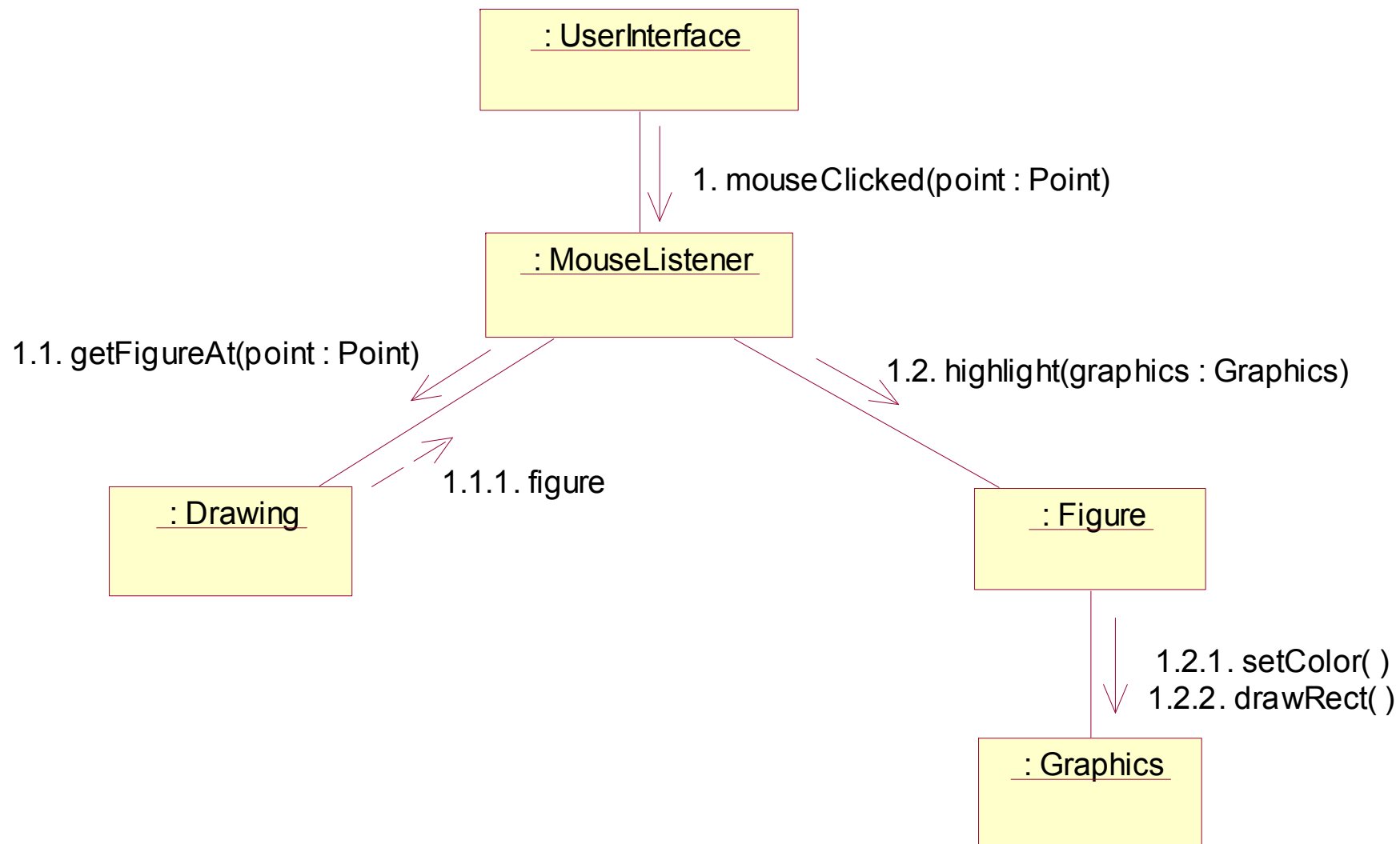
## ❖ اجزای نمودار

- اشیا
- لینک های میان اشیا که شامل اطلاعات زیر است:
  - پیام مبادله شده بین دو شی
  - جهت تبادل پیام
  - شماره پیام: پیام ها به صورت ترتیبی یا سلسله مراتبی شماره گذاری می شوند.

# نمودار همکاری ...



# نمودار همکاری ...



# نمودار حالت

## STATECHART DIAGRAM



# نمودار حالت (Statechart Diagram)

- ❖ مدل کردن رفتار یک نمونه کلاس، usecase و یا سیستم
- ❖ نمایش حالات مختلف یک شی و شرایط لازم برای انتقال از یک حالت به حالت دیگر
- ❖ مدل سازی چرخه زندگی یک شی (از لحظه ایجاد تا تخریب آن)
  - یک شی در طی چرخه زندگی خود چه حالاتی (state) می تواند داشته باشد؟
  - چه رویداد هایی (event) منجر به انتقال (transition) از یک حالت به حالت دیگر می شوند؟
  - به ازای هر تغییر حالت، چه کارهایی (action/activity) انجام می شود؟

# نمودار حالت ...

## ❖ رفتار شی

- چگونگی عمل و عکس‌العمل یک شی در مقابل دریافت و یا ارسال پیام
- ✓ رفتار شی ممکن است وابسته به حالت شی باشد.

## ❖ حالت شی (state)

- مقادیر صفات شی در یک لحظه خاص
- یک موقعیت خاص در طی چرخه زندگی شی که در آن موقعیت:
  - شی دارای شرایط خاصی است،
  - فعالیت‌های خاصی را انجام می‌دهد،
  - یا منتظر وقوع یک رویداد خاص می‌باشد.

State

# نمودار حالت ...

## ❖ رویداد (event)

- هر چیزی که ممکن است در محیط عملیاتی اتفاق بیافتد:
  - سیگنال (signal)
  - فراخوانی یک تابع (call)
  - گذر زمان (passing of time)
  - برآوردن شدن یک شرط یا تغییر یک وضعیت (change)
- ✓ یک رویداد ممکن است موجب انتقال از یک حالت به حالت دیگر شود.

## ❖ کنش (action)

- عملی است که در صورت اجرا شدن، باید طور کامل اجرا شود و نمی توان آن را متوقف کرد (atomic).

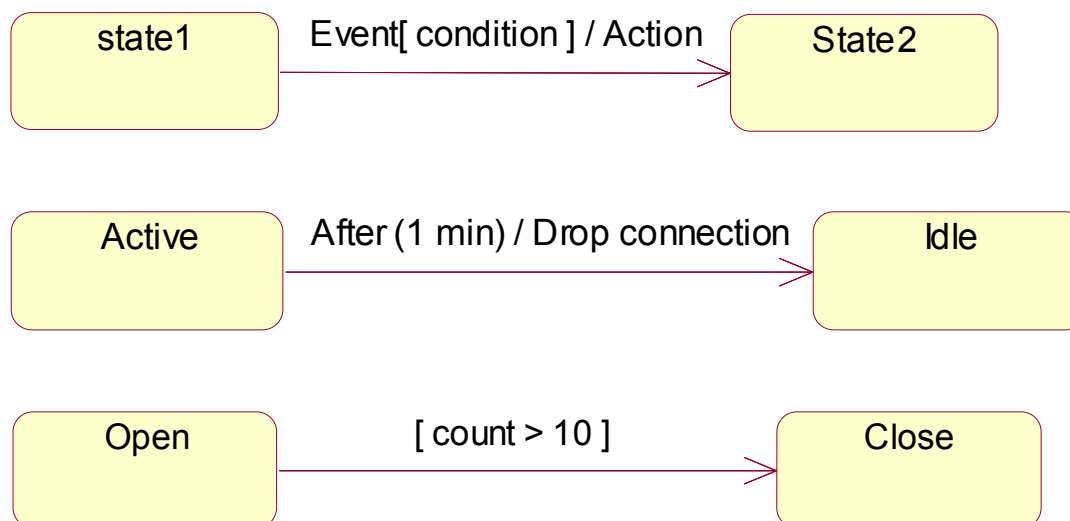
## ❖ فعالیت (activity)

- عملی است که در صورت اجرا شدن، ممکن است با وقوع یک رویداد، متوقف شود و به طور کامل اجرا نشود (non atomic).

# نمودار حالت ...

## ❖ انتقال (Transition)

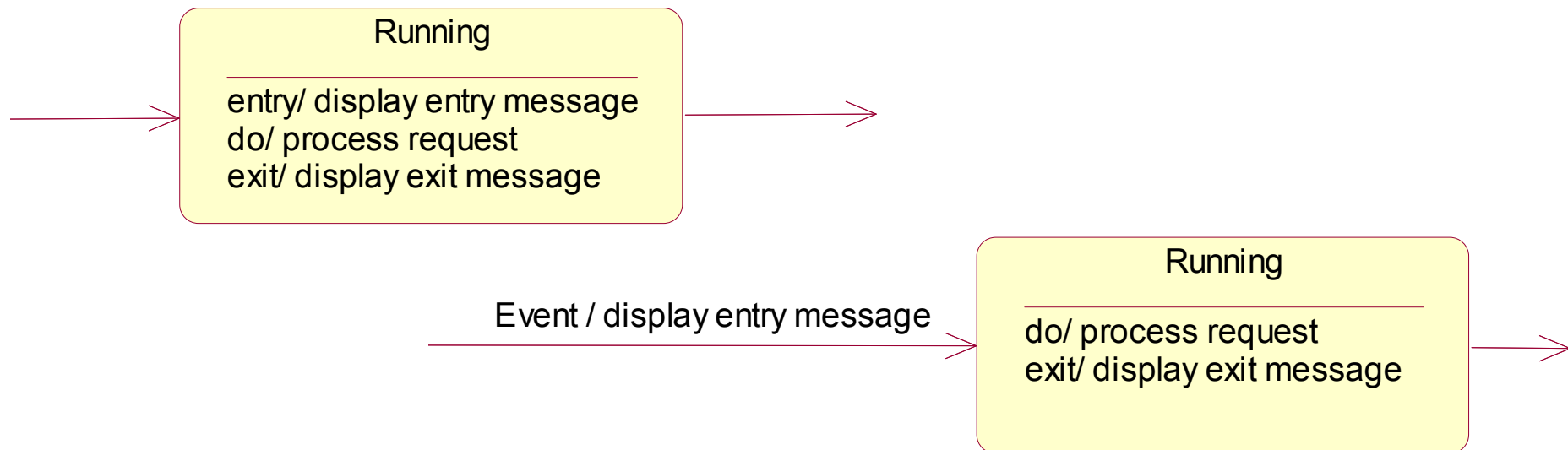
- ارتباط بین دو حالت
- اگر رویداد event رخ دهد و شرایط condition برقرار باشد، شی که در حالت state1 قرار داد، ابتدا عمل action را انجام می دهد و سپس وارد حالت state2 می شود.



# نمودار حالت ...

## ❖ اجزای حالت

- کنش ورودی (entry): به هنگام وارد شدن به یک حالت اجرا می شود.
  - کنش خروجی (exit): به هنگام خروج از یک حالت اجرا می شود.
  - فعالیت (do): پس از اجرای کنش ورودی (در صورت وجود)، فعالیت مورد نظر اجرا می شود. اما در صورت وقوع یک رویداد، اجرای فعالیت متوقف خواهد شد.
- ✓ می توان به جای تعریف کنش ورودی، آن را به عنوان کنش انتقال تعریف کرد.

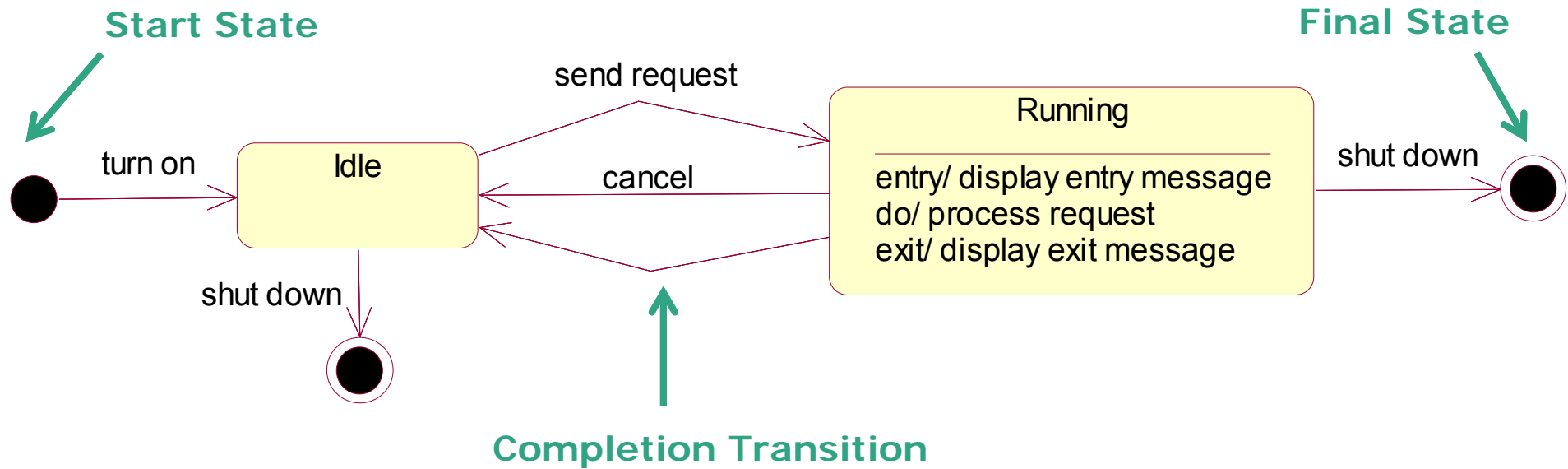


# نمودار حالت ...

## ❖ مراحل مدل سازی

- تعیین محدوده مدل سازی: شی، use case یا سیستم
- تعیین حالت‌ها
- تعیین رویدادها
  - رویداد های ناشی از تعامل کاربر با سیستم (واسط کاربری)
  - رویداد های ناشی از روابط میان اشیا
- رسم انتقال‌ها میان حالات
- افزودن کنش‌ها و فعالیت‌ها (در صورت لزوم)
- گسترش یک حالت به چند زیر حالت و یا بر عکس (در صورت لزوم)

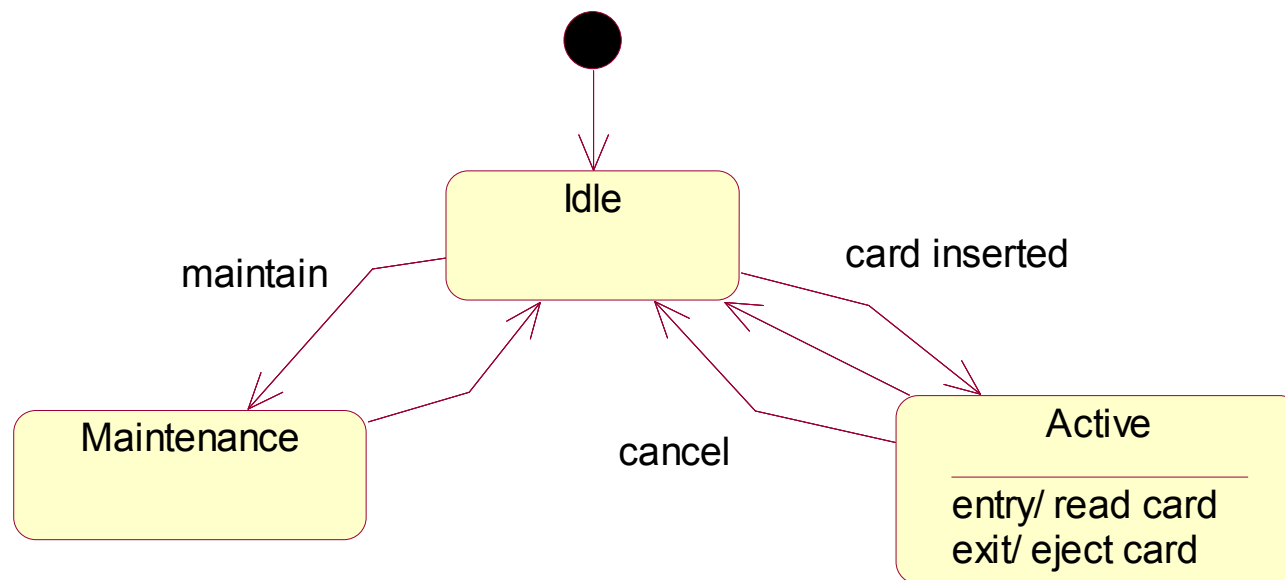
# نمودار حالت ...



# نمودار حالت ...

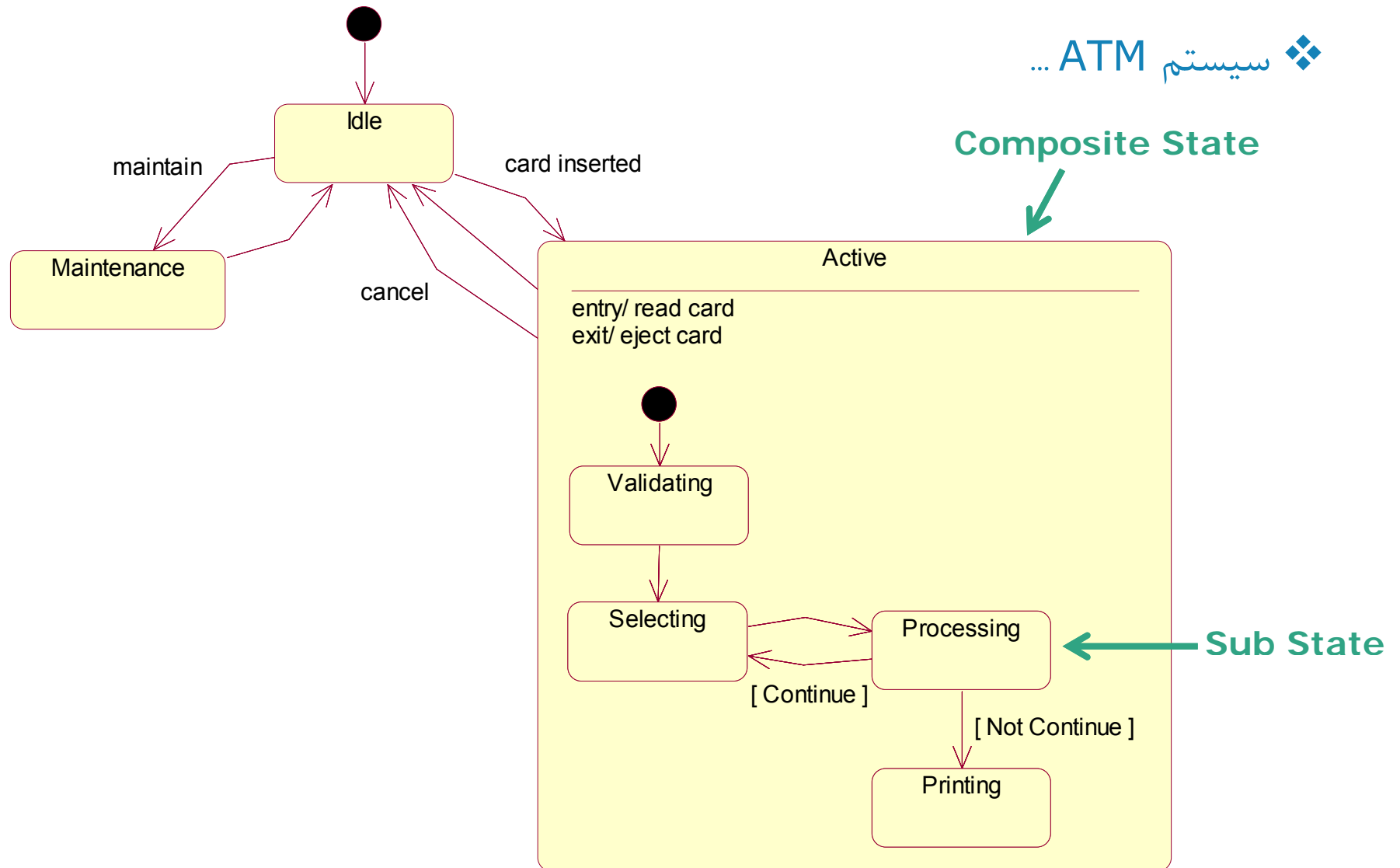
## ❖ سیستم ATM

- **use case:** مشتری کارت خود را وارد سیستم می کند. سیستم کارت را خوانده و اعتبار آن را بررسی می کند. در صورت معتبر بودن کارت، مشتری یک درخواست را انتخاب می کند. پس از انجام تراکنش، مشتری می تواند در خواست دیگری را به سیستم اعلام کند و یا از ادامه کار صرف نظر کند. در صورت انصراف، رسید یا گزارش فعالیت مشتری چاپ می شود و سپس کارت به او تحویل داده می شود. مشتری این امکان دارد که در هر مرحله از فرآیند، از ادامه کار انصراف دهد. همچنین سیستم قابلیت نگهداری دارد: مانند جایگزینی پول، تعمیر و ... )

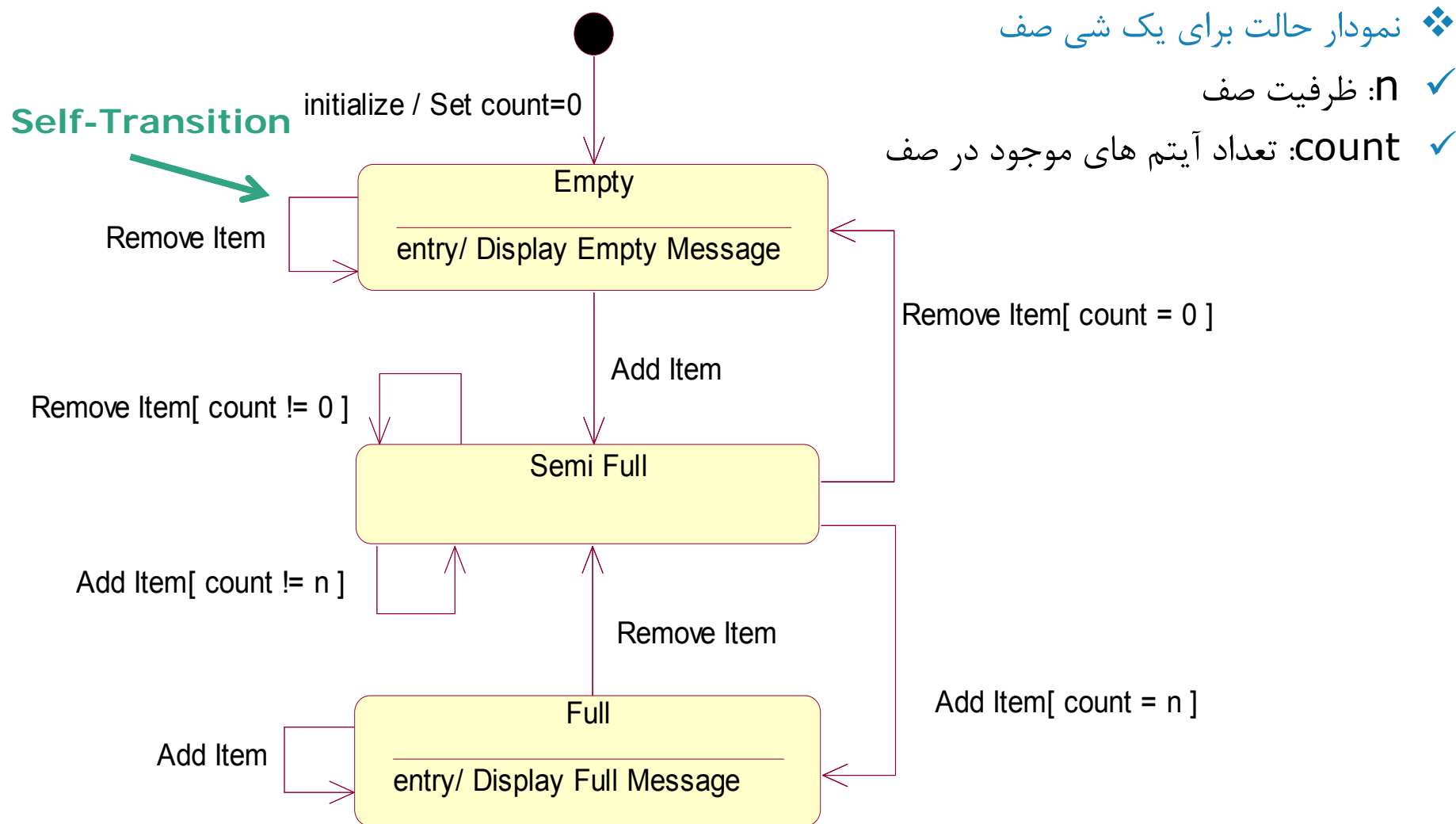




# نمودار حالت ...



# نمودار حالت ...



## نمودار حالت ...

```
Class Queue
{
    int state,n,count;
    int Empty=0,Ready=1,Full=2;
    Item[] itemList;
    public Queue(int capacity)
    {
        n=capacity;
        state=Empty;
        count=0;
        itemList=new Item[n];
    }

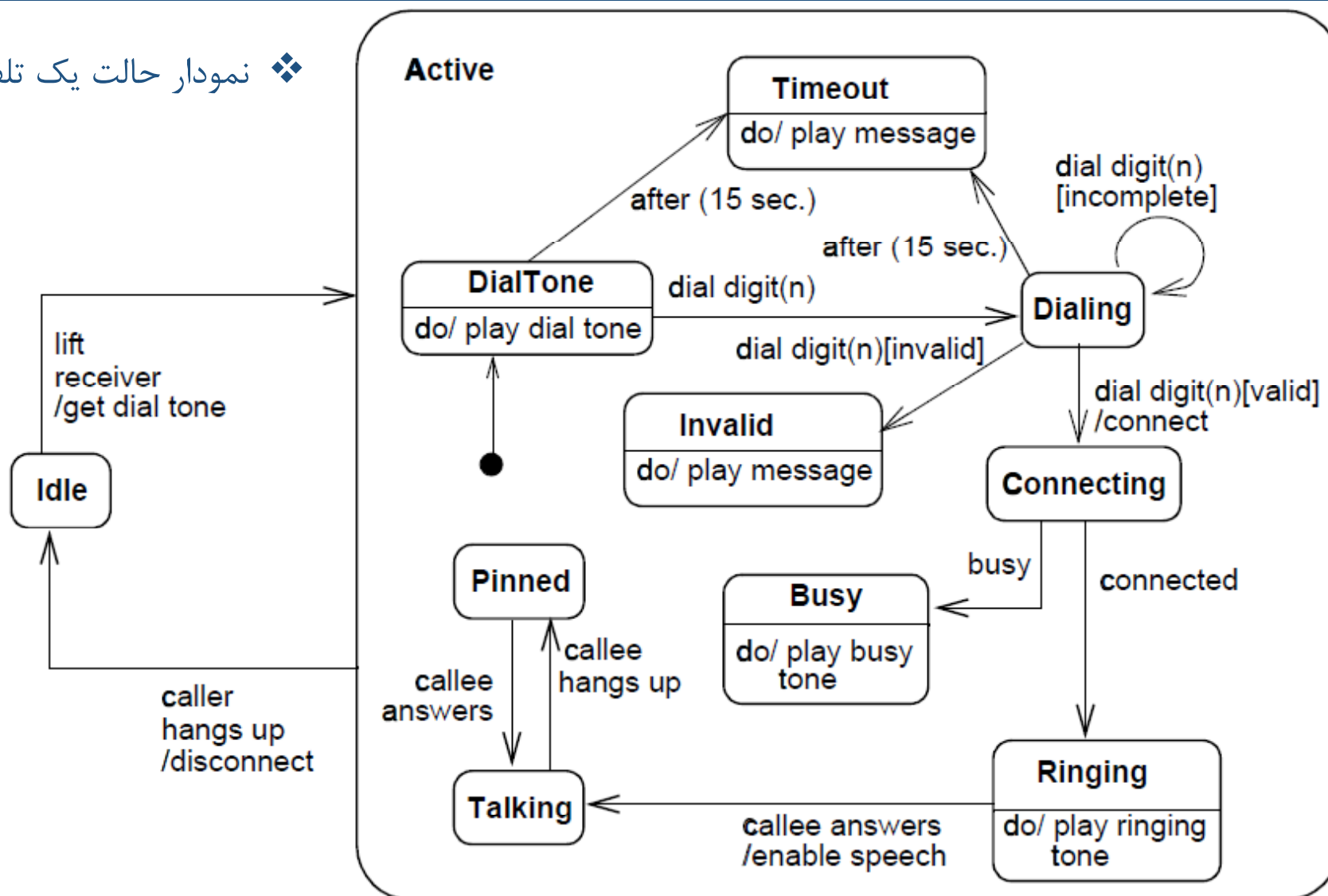
    ...
}
```

```
public void AddItem(Item item)
{
    if(state==Full)
        print("Queue is Full");
    else {
        itemList[count++]=item;
        if(count==n)
            state=Full;
        else
            state=Ready;
    }
}

...
} //end of class
```

# نمودار حالت ...

❖ نمودار حالت یک تلفن



# نمودار حالت ...

## ❖ کاربردها

- مدل سازی سیستم های رویداد محور (event-oriented) یا تعاملی
- مدل سازی رفتار یک کلاس، use case یا سیستم
- مدل سازی اشیایی که رفتار آنها به حالات گذشته وابسته است یا باید به سیگنال های خارجی پاسخ دهند.
- نمایش چرخه عمر یک شی

## ❖ نکات

- نمودارهای تعاملی رفتار سیستم را با توجه به ساختار کلاس ها نشان می دهند اما نمودار حالت توجهی به ساختار کلاس ها ندارد.
- نمودار های تعاملی رفتار مجموعه ای از اشیا را مدل می کنند اما نمودار حالت رفتار یک شی را مدل می کند.
- نمودار فعالیت یک نوع خاص از نمودار حالت است.

مدل سازی داده ها

**DATA MODELING**

# مدل سازی داده ها ...

❖ اگر نیازمندی های نرم افزار شامل ایجاد، گسترش یا برقراری ارتباط با یک پایگاه داده، ساختمان داده ها یا ساختار فایل های پیچیده باشد، نیاز به مدل سازی داده ها داریم.

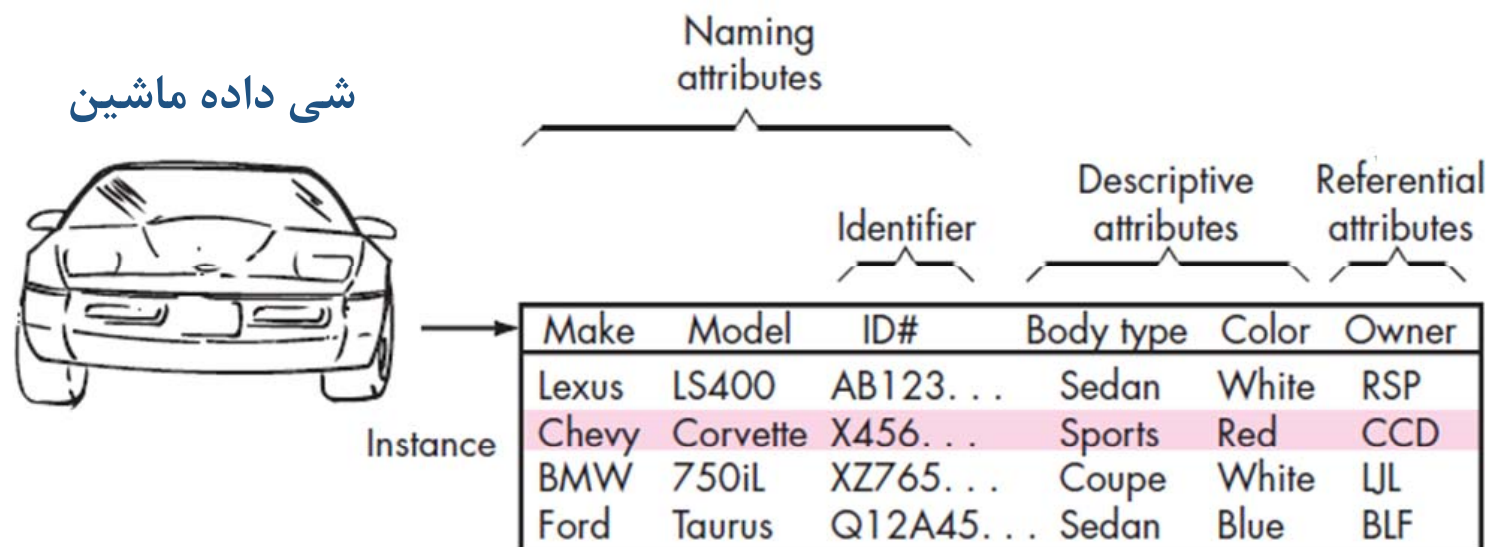
## ❖ مدل سازی داده ها

- تعیین اشیای داده و صفات آنها
- تعیین روابط میان اشیای داده

## ❖ شی داده (Data Object)

- نمایشی از هر گونه اطلاعات ترکیبی که باید نرم افزار آنها را درک کند.
- شامل مجموعه ای از صفات مختلف
- شی داده، فقط داده ها را پنهان سازی می کند و شامل عملیات قابل انجام بر روی داده نیست.

# مدل سازی داده ها ...



## ❖ انواع صفات

- صفت (صفات) شناسه (Identifier): مقدار آن باید در بین نمونه های شی داده منحصر بفرد باشد.
- صفات نام گذاری (Naming): نام گذاری یک نمونه از شی داده
- صفات توصیفی (Descriptive): توصیف یک نمونه از شی داده
- صفات ارجاعی (Referential): ارجاع به یک شی داده (جدول) دیگر

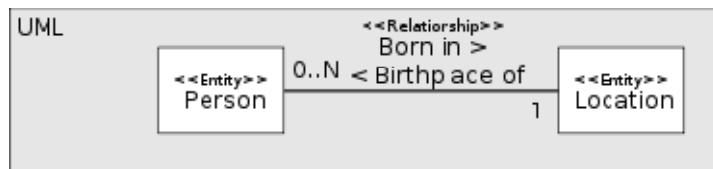
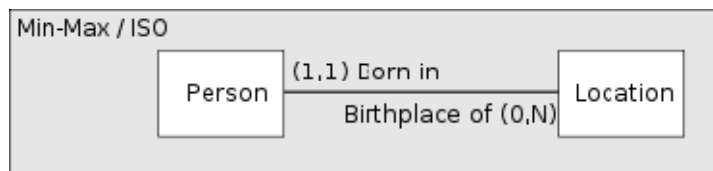
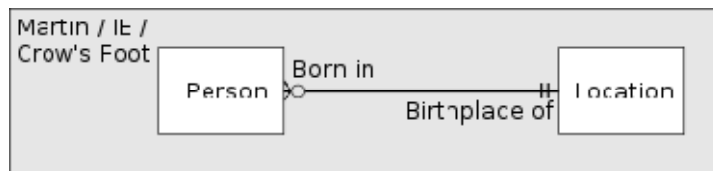
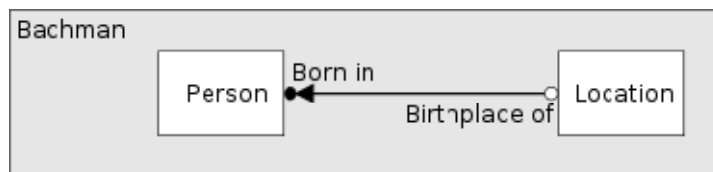
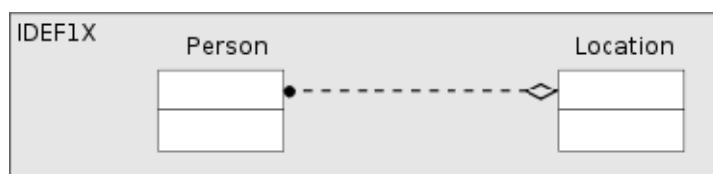
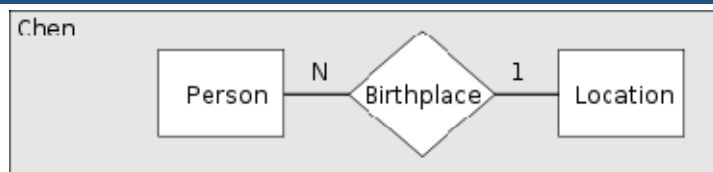


# مدل سازی داده ها ...

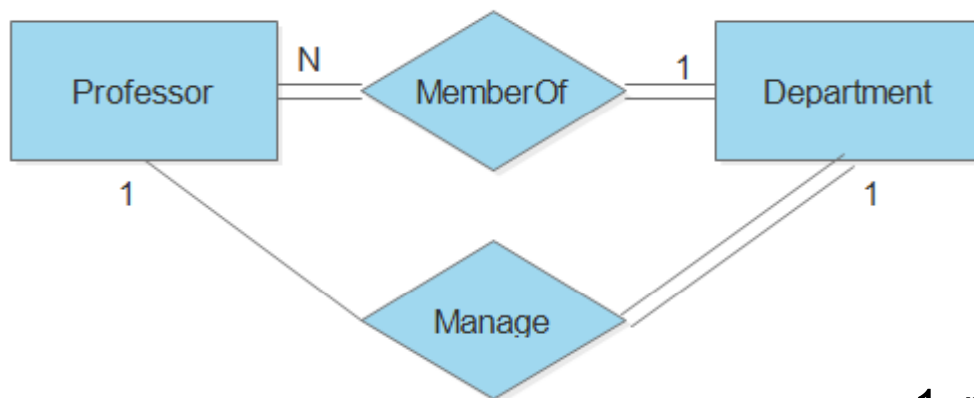
## ❖ روش های مدل سازی

■ نمودار ER (Entity-Relationship Diagram)

■ از نمودار کلاس ها در استاندارد UML نیز می توان برای مدل سازی داده ها استفاده کرد.

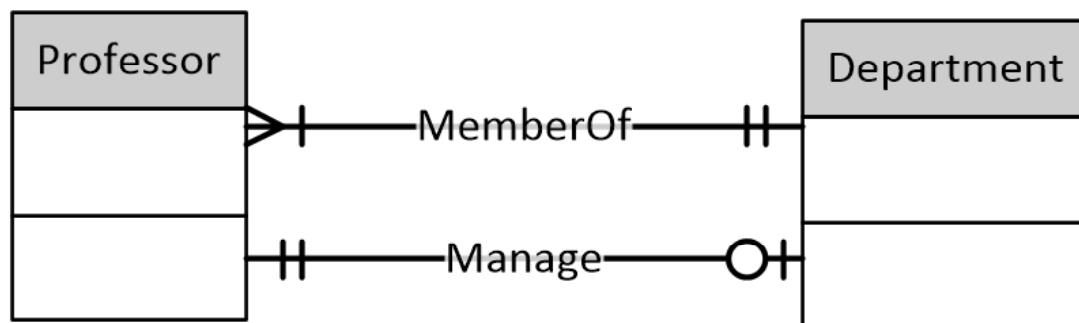
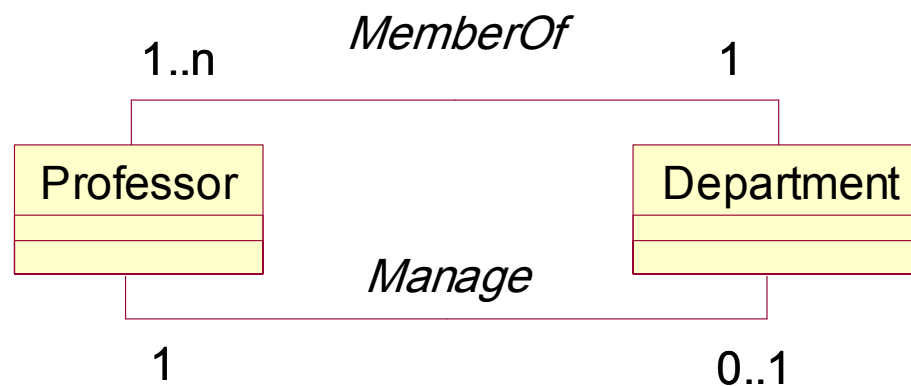


# مدل سازی داده ها ...



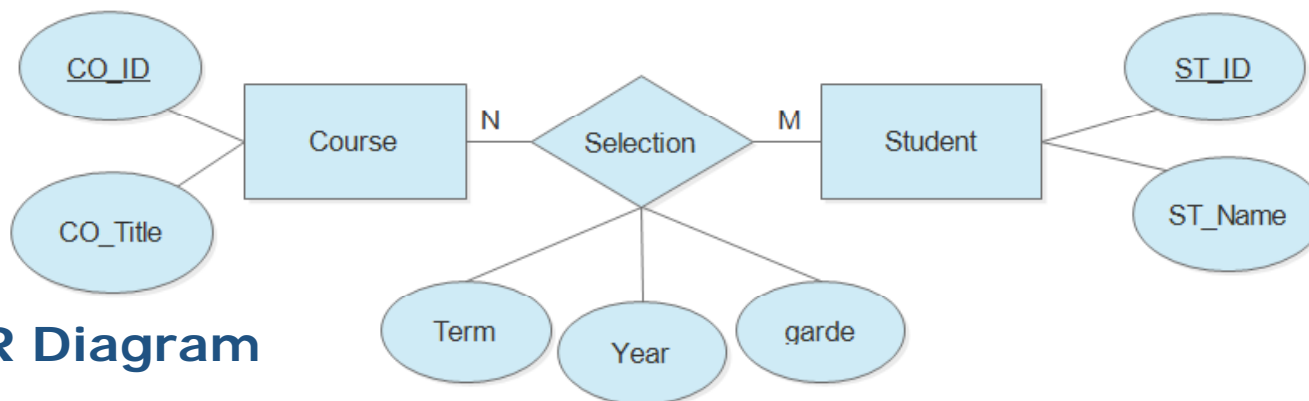
Chen E-R Diagram

UML Class Diagram



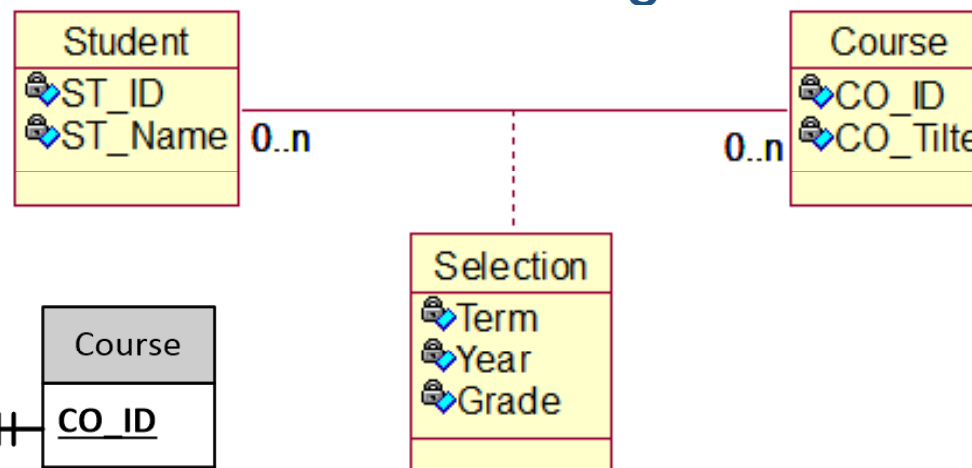
Crow's Foot E-R Diagram

# مدل سازی داده ها ...

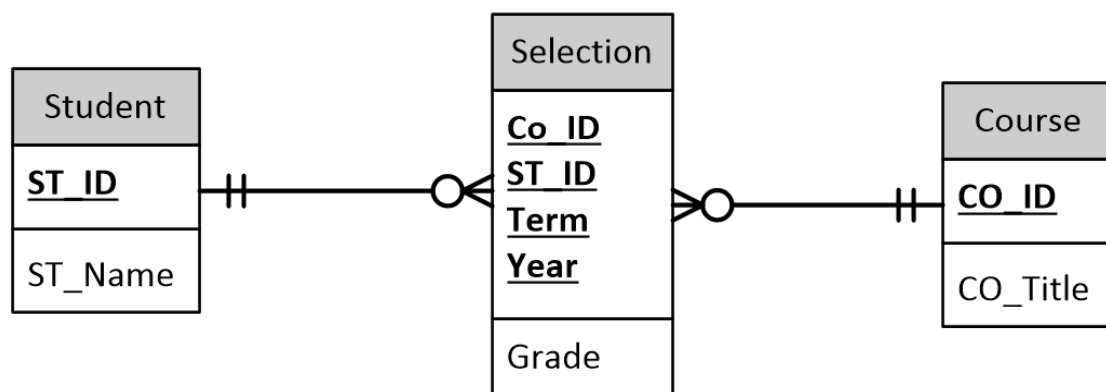


Chen E-R Diagram

UML Class Diagram



Crow's Foot E-R Diagram

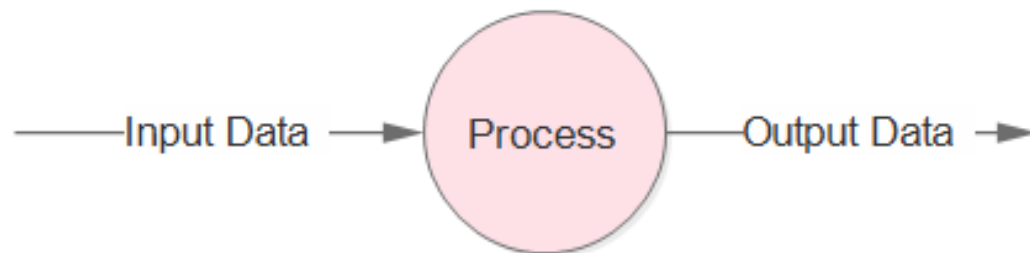


نمودار جریان داده ها

**DATA FLOW DIAGRAM**

# نمودار جریان داده ها (DFD)

- ❖ تجزیه سیستم از نظر عملیاتی
- ❖ مدل سازی سیستم به صورت مجموعه ای از داده ها ، پردازش ها و روابط بین پردازش ها
- ❖ نمایش سیستم بر اساس مدل ورودی-پردازش-خروجی (IPO)
  - مجموعه ای از داده ها به داخل سیستم جریان پیدا می کنند.
  - پردازش ها (تبدیلات) بر روی داده ها انجام می شوند.
  - داده های حاصل، به خارج از سیستم جریان پیدا می کنند.
- ✓ هسته اصلی متدولوژی ساخت یافته
- ✓ عدم نمایش اولویت و ترتیب اجرای پردازش ها



## اجزای نمودار

### ❖ پردازش

- یک یا چند جریان داده ورودی را به یک یا چند جریان داده خروجی تبدیل می کند.

### ❖ جریان داده

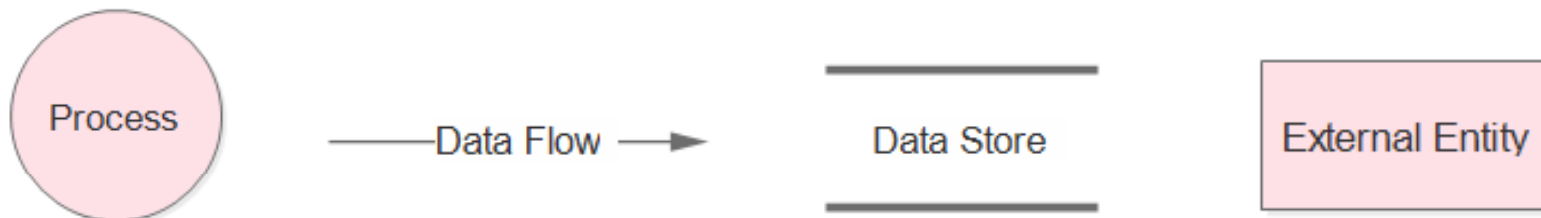
- داده هایی که بین پردازش ها مبادله می شود.

### ❖ مخزن داده

- مکانی برای ذخیره داده ها و اطلاعات سیستم

### ❖ پایانه / موجودیت خارجی

- نرم افزار، سخت افزار، افراد و سیستم های دیگری که با سیستم مورد نظر تبادل داده دارند.
- بخشی از خود سیستم یا سیستم دیگر است که اطلاعات تولید شده توسط سیستم را مصرف می کند و یا اطلاعاتی را برای سیستم تولید می کند.



## مراحل مدل سازی

### (۱) شناخت و تعیین اجزای نمودار

#### ■ تجزیه گرامری use case ها

- عبارت مصدری / فعلی: پردازش ها
- عبارت اسمی: جریان داده ها، مخزن های داده ای و یا موجودیت های خارجی

### (۲) رسم سلسله مراتبی نمودار

- نمودار DFD معمولاً در چند سطح رسم می شود و با افزایش سطوح، جزئیات بیشتری به آن افزوده می شود.

## مراحل مدل سازی ...

### ❖ DFD سطح صفر (Context Diagram)

- نمودار فقط یک پردازش دارد که همان سیستم است.
- نمایش سیستم، ورودی ها و خروجی های اصلی سیستم
- تعیین مرز سیستم

### ❖ DFD سطح $i$ ام ( $i \geq 1$ )

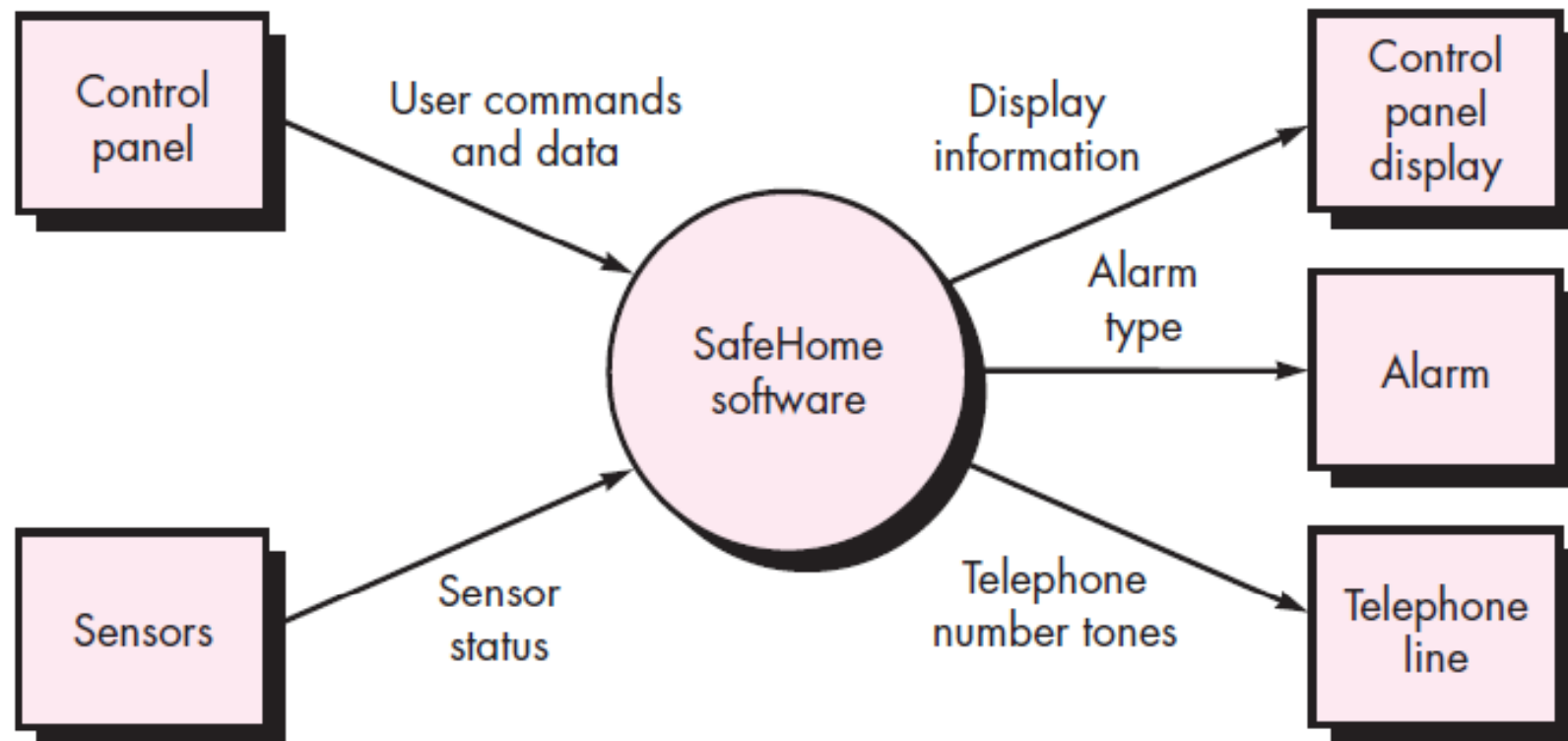
- در DFD سطح  $i$  ام، یک پردازش از سطح قبل در صورت امکان به پردازش های کوچکتر تقسیم می شود. (پالایش)

- ✓ برای هر پردازش موجود در DFD سطح  $i-1$  ام، می توان یک DFD سطح  $i$  ام رسم کرد.
- ✓ پیوستگی (توازن) جریان ها از یک سطح به سطح دیگر باید حفظ شود. ورودی ها و خروجی هر پردازش در سطح  $i$  ام باید برابر با ورودی ها و خروجی های آن در سطح  $i-1$  ام باشد.
- ✓ پالایش DFD ها تا زمانی می تواند ادامه پیدا کند که هر پردازش فقط معادل با یک عملکرد از سیستم باشد (به راحتی به عنوان یک مولفه برنامه قابل پیاده سازی باشد)



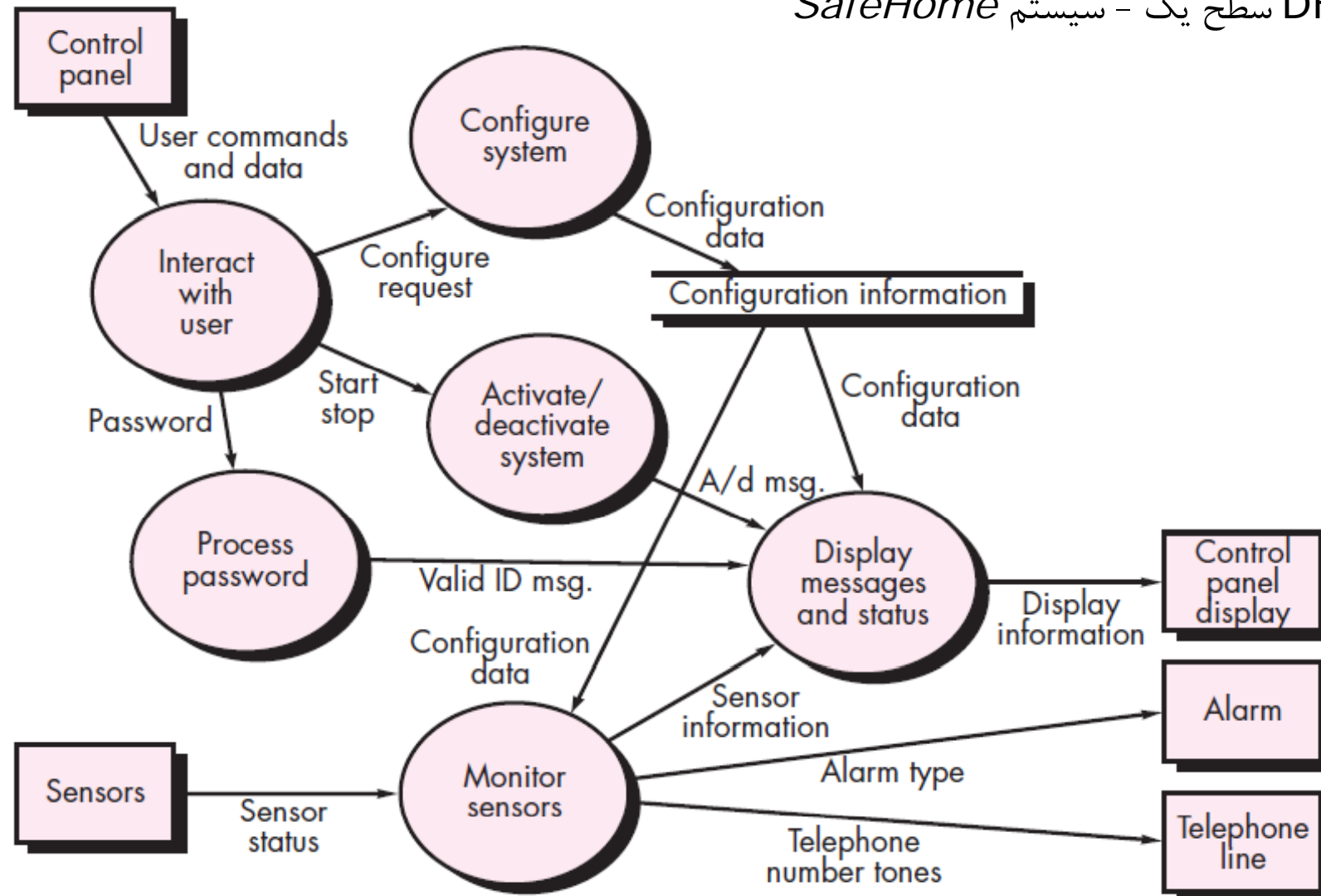
## مراحل مدل سازی ...

DFD سطح صفر - سیستم SafeHome



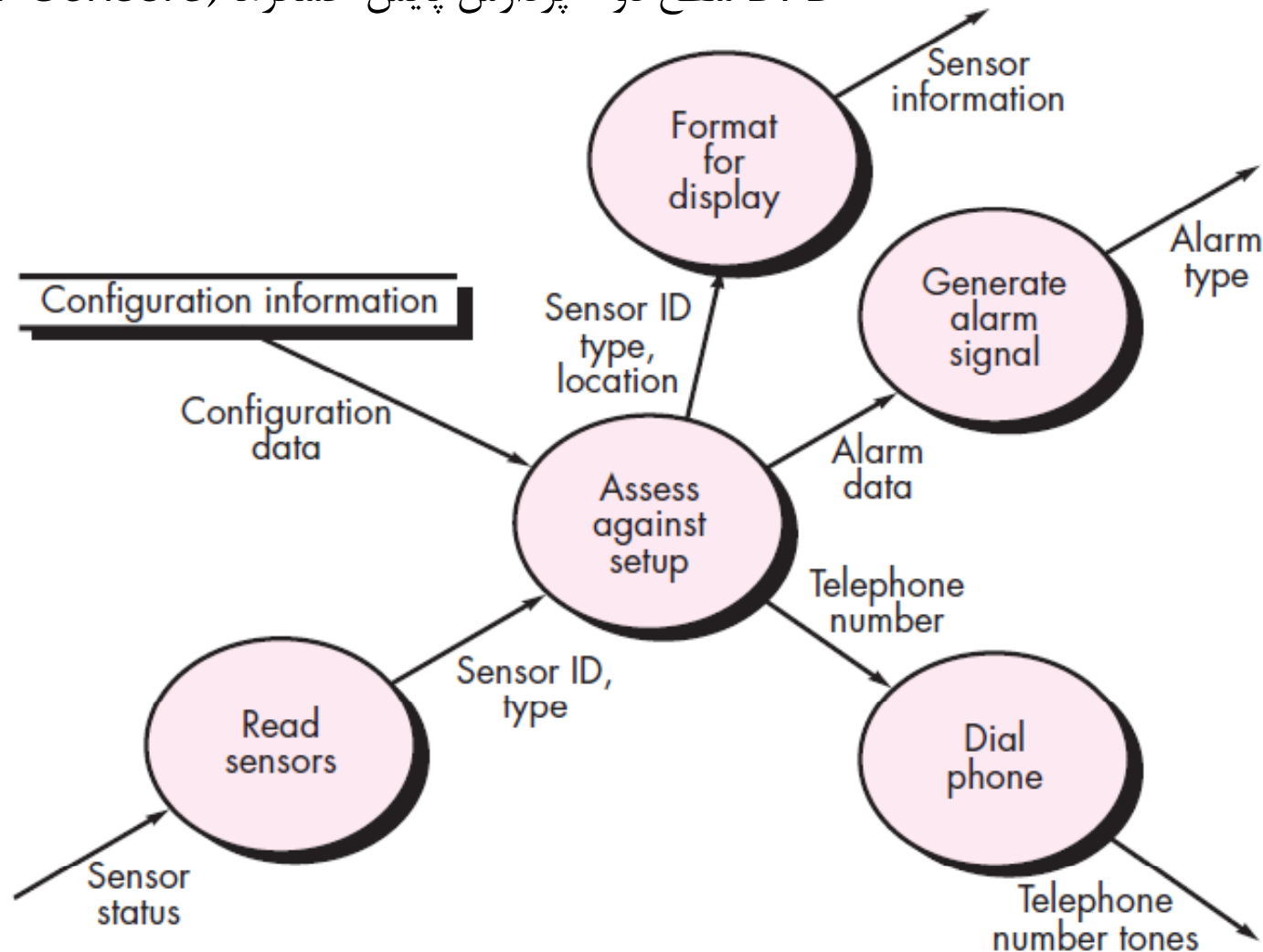
# مراحل مدل سازی ...

DFD سطح یک - سیستم SafeHome



# مراحل مدل سازی ...

DFD سطح دو - پردازش پایش حسگرها (*Monitor Sensors*)



## نکات مدل سازی

- ✓ همه اجزای نمودار باید دارای نام باشند.
- ✓ نام پردازش ها باید نام منحصر بفرد باشد (در یک سطح از نمودار)
- ✓ در یک سطح از نمودار، یکسان بودن نام دو جریان، به معنای یکسان بودن عملکرد آنها است.
- ✓ هر پردازش باید حداقل دارای یک ورودی و یک خروجی باشد.
- ✓ اگر پردازش فقط دارای ورودی و یا فقط دارای خروجی باشد، باید آن را به صورت پایانه نمایش داد.
- ✓ جریان های ورودی یک پردازش با جریان های خروجی آن باید متفاوت باشد.
- ✓ جریان داده ای باید یک طرفه باشد.
- ✓ یک جریان نباید به طور مستقیم به پردازشی که از آن خارج شده است، دوباره وارد شود.
- ✓ نمی توان دو پایانه، دو مخزن داده، و یا یک پایانه و یک مخزن داده را به طور مستقیم از طریق رسم یک جریان داده به هم وصل کرد. حتما باید یک پردازش بین آنها وجود داشته باشد.

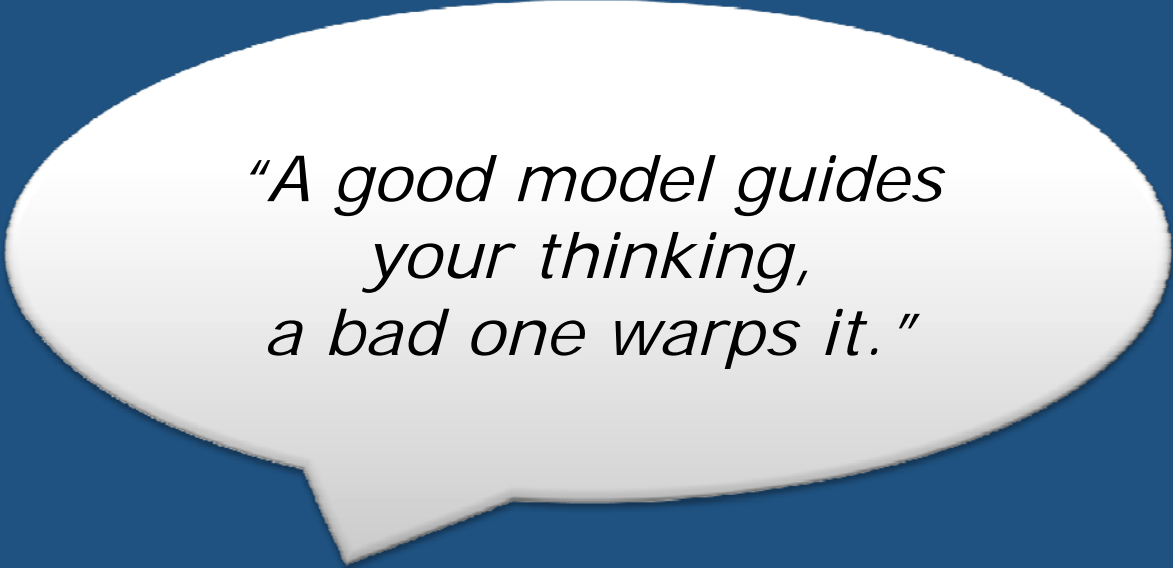
# روش های کلی تحلیل نیازمندی ها

## ❖ تحلیل ساخت یافته

- مدل سازی اشیا داده ای، صفات و روابط بین آنها (نمودار ER)
- مدل سازی پردازش هایی که اشیای داده ای را دستکاری می کنند (نمودار جریان داده و نمودار حالت)

## ❖ تحلیل شی گرا

- تعیین کلاس ها و شیوه همکاری بین آنها
- استفاده از UML برای مدل سازی نیازمندی ها
  - نمودار های مبتنی بر سناریو، رفتاری و مبتنی بر کلاس
- تاکید بر سه جنبه مختلف سیستم (عملیاتی، پویا و ایستا)



*"A good model guides  
your thinking,  
a bad one warps it."*

# Question ?