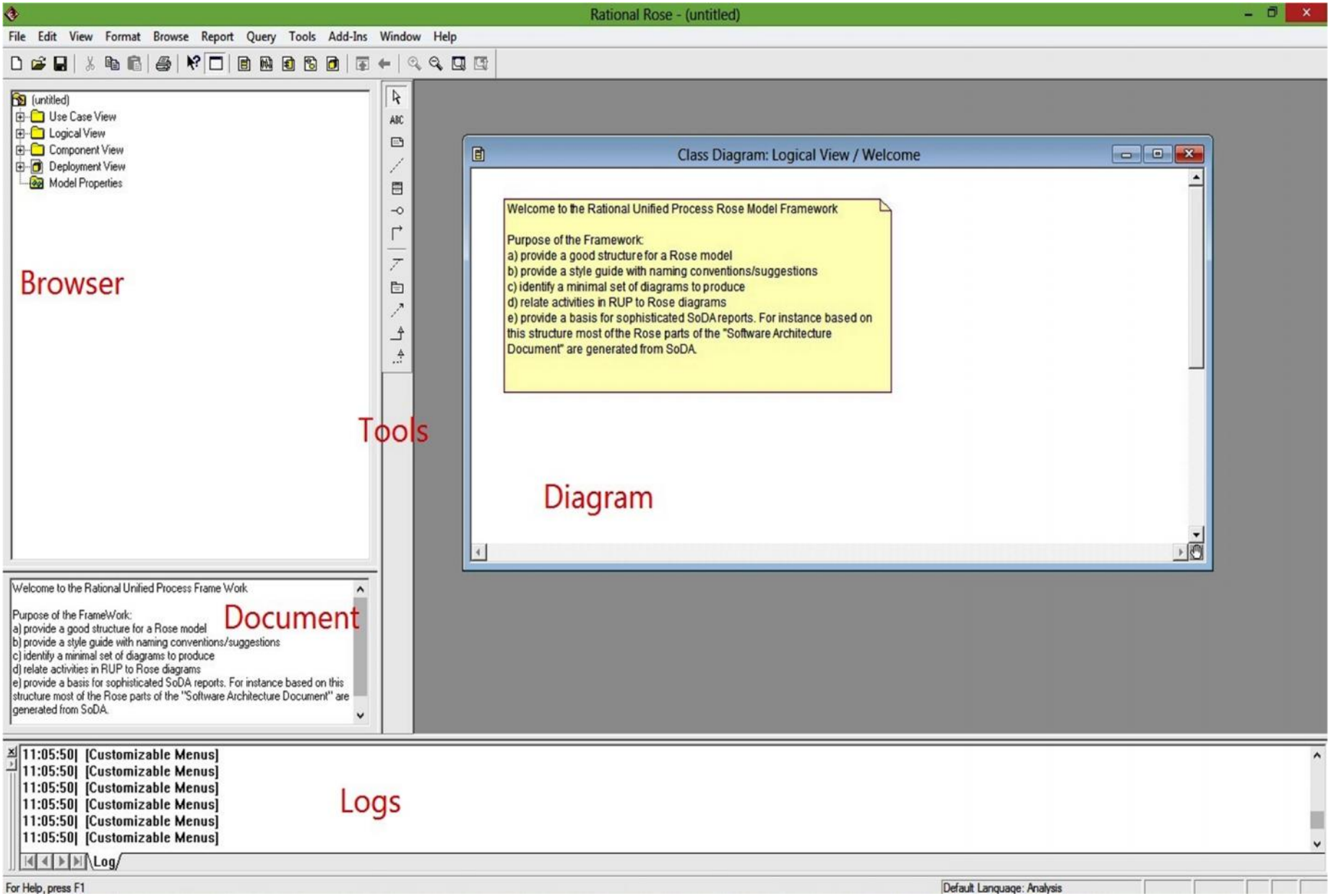
آموزش نرم افزار Rational Rose

مدرس: برادران هزاوه

پاییز ۹۲


بسم الله الرحمن الرحيم


Rational Rose - (untitled)
File Edit View Format Browse Report Query Tools Add-Ins Window Help
(untitled)
Use Case View
Logical View
Component View
Deployment View
Model Properties
Browser
Tools
Class Diagram: Logical View / Welcome
Welcome to the Rational Unified Process Rose Model Framework
Purpose of the Framework:
a) provide a good structure for a Rose model
b) provide a style guide with naming conventions/suggestions
c) identify a minimal set of diagrams to produce
d) relate activities in RUP to Rose diagrams
e) provide a basis for sophisticated SoDA reports. For instance based on this structure most of the Rose parts of the "Software Architecture Document" are generated from SoDA.
Diagram
Welcome to the Rational Unified Process Frame Work
Purpose of the FrameWork:
a) provide a good structure for a Rose model
b) provide a style guide with naming conventions/suggestions
c) identify a minimal set of diagrams to produce
d) relate activities in RUP to Rose diagrams
e) provide a basis for sophisticated SoDA reports. For instance based on this structure most of the Rose parts of the "Software Architecture Document" are generated from SoDA.
Document
11:05:50| [Customizable Menus]
11:05:50| [Customizable Menus]
11:05:50| [Customizable Menus]
11:05:50| [Customizable Menus]
11:05:50| [Customizable Menus]
11:05:50| [Customizable Menus]
Logs
Log
For Help, press F1
Default Language: Analysis

نمودار کلاس

این نمودار، کلاسها، وابط ها و همکاری و روابط بین آنها را نمایش می دهد و نمودار اصلی و مرکزی UML است که بیان کننده‌ساختار ایستای سیستم نرم افزاری است.

نمودار اشیا

این نمودار، اشیا سیستم و روابط بین آنها را نمایش می دهد. در واقع یک تصویر لحظه ای از نمودار کلاس می باشد. نرم افزار به اجزا مستقل تقسیم می‌شود و هر جز یا شی به‌طور کامل تعریف می‌شود.

نمودار موردکاربرد

این نمودار، تعامل کاربران خارجی و سیستم را مدل می کند و از جهاتی شبیه نمودار سطح صفرDFD می باشد که جنبه های رفتاری سیستم را نمایش می دهد. این نمودار نقطه ورودی برای تمامی نمودارهای دیگری است که به تشریح نیازمندیها و معماری و پیاده سازی سیستم می پردازد.

نمودار حالت

این نمودار، بیان کننده جنبه های رفتاری سیستم می باشد و در واقع توصیف رسمی یک کلاس بوده که شامل حالات، انتقال بین حالات، رخدادها و فعالیت ها می باشد. از این نمودارها برای نمایش دادن چرخه حیات اشیا یک کلاس خاص نیز می توان استفاده کرد.

نمودار فعالیت

نوع خاصی است از نمودارحالت که انتقال جریان از یک فعالیت به فعالیت دیگر را نمایش می دهد. این نمودار جنبه های پویایی یک سیستم را نمایش می دهد. در واقع حالات این نمودار گام های ترتیبی انجام یک عمل را نمایش می دهند.شامل رفتارها و عملکردهایی است که یک شی در حالت خاص اجرا می کند.

نمودار اجزا

از جمله نمودارهای پیاده سازی می باشد و سازماندهی و روابط بین مجموعه ای از اجزا را نمایش میدهد این نمودار، جنبه های ایستای پیاده سازی یک سیستم را مدل می کند. تمامی کتابخانه ها و فایل های مورد نیاز در زمان اجرای برنامه را تعیین می کند.

نمودار استقرار

پیکربندی گره های پردازشی زمان اجرا را نمایش می دهد که برای مدل کردن جنبه های ایستای به کارگماری یک معماری به کار می رود. هم چنین نمایش دهنده اجزای استفاده‌شده زمان اجرا مثل کتابخانه های DLL، فایل های اجرایی، کدهای مبدا و روابط بین آنها می باشد.

معماری فیزیکی یک سیستم در این نمودار مشخص می‌شود.تعداد کامپیوترها، نوع سیستم عامل ها، نرم افزارهای جانبی و چگونگی اتصالات آنها.

نمودار تعامل

بیان کننده تعامل هستند که‌شامل اشیا مختلف است و نیز روابط بین آنها و هم چنین پیغام هایی که بین آنها رد و بدل می‌شود. این نمودارها جنبه های پویایی یک سیستم را مدل می کنند و خود بر دو نوعند: 1. نمودار توالی 2. نمودار همکاری

نمودار توالی، ترتیب زمانی تعامل ها را نشان می دهد.

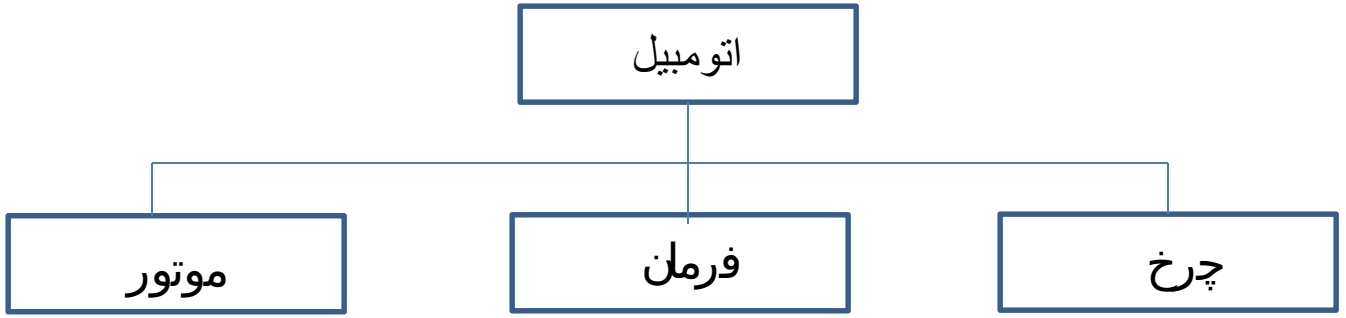
نمودار همکاری، تاکید بر نمایش ساختاری تعامل ها دارد.

نمودار شی

- نرم افزار کتابخانه کتاب، مشتری، مدیریت امانت و ....
- نرم افزار بیمارستان بیمار، پزشک، پرسنل و ....
- نرم افزار دانشگاه دانشجو، استاد، درس، پرسنل و ....

برای انتخاب اشیا باید نکات زیر را رعایت کرد:

1. اشیا باید ملموس باشند.
2. اشیا باید قابل درک باشند.
3. اشیا باید مورد فعل و فکر قرار گیرند.



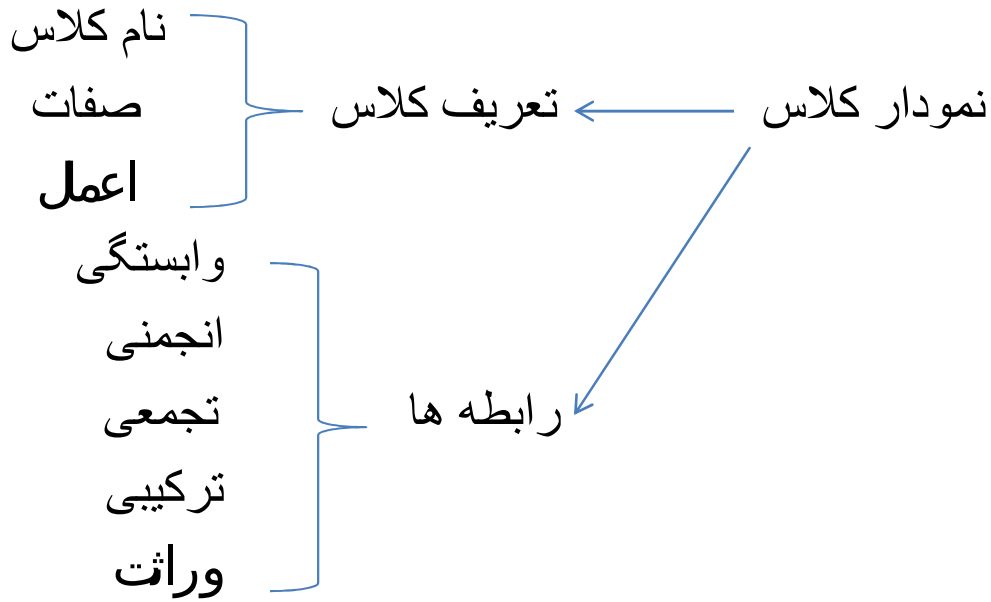
Class Diagram

این نمودار دیدگاه کلی از سیستم ارایه می دهد.

نکته:

نمودار کلاس، یک ابزار طراحی خوب برای تیم های نرم افزاری است. این نمودارها به برنامه نویسان کمک می کند تا ساختار سیستم را قبل از اینکه کدی نوشته شود، ببینند و طراحی کنند؛ هم چنین کمک می کند تا از طراحی خوب سیستم اطمینان حاصل کنند.

Student
Name Text[50]; Family Text[100]; ….
InsertCourse(); DeleteCourse(); Payment(); ….



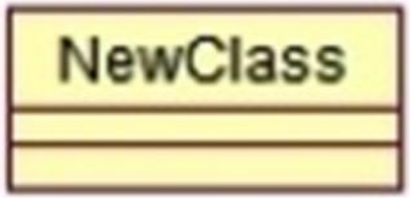
Class Diagram

- نمایش Class در UML
- روابط میان کلاسها
  - Association
  - Generalization
  - Aggregation ,Composition
  - Realization

نام کلاس
صفات و ویژگی ها
عملیات و رفتار شی

نمایش Class در Rational Rose

- در UML کلاس را با مربع نشان می دهند که شامل سه بخش است. در این بخشها به ترتیب از بالا به پایین: نام کلاس، صفات کلاس و متدهای کلاس قرار می گیرد.



- در قسمت اول مربع معرف کلاس، نام کلاس درج می‌شود. نام کلاس باید منحصر به‌فرد باشد و بیان کننده ماهیت کاری آن کلاس باشد.

  حرف اول نام کلاس باید با حروف بزرگ نوشته‌شود. اگر نام کلاس از دو کلمه یا بیشتر تشکیل‌شود، باید حرف اول هر بخش بزرگ درج گردد.
- در قسمت دوم مربع معرف کلاس، صفت و ویژگیهای اشیا مشخص می گردد. برای صفت باید نوع و تعداد آنها معین گردد. در تعیین صفت حروف بزرگ و کوچک فرقی ندارد.

  نکته:

  1. هیچ کلاسی بدون صفت نمی باشد.
  2. هر کلاسی حداقل یک کلید باید داشته باشد.(یعنی صفتی منحصر به‌فرد)
- در قسمت‌سوم مربع معرف کلاس، عملیات و رفتار اشیا مشخص می گردد. عملیات، کاری است که یک کلاس هس تواند انجام دهد.

  نام یک عملیات، با حروف کوچک نوشته می‌شود و اگر نام آن بیش از یک کلمه باشد، آنگاه کلمات بهم چسبیده و حروف اول هر کلمه بزرگ نوشته می‌شود. در داخل پرانتز، پارامتری که عملیاتی روی آن انجام می‌شود و یا تابعی که‌فراخوانده می‌شود، نوشته می گردد.

نمایش Class در Rational Rose (..)

- برای ایجاد صفات و یا متدهای یک کلاس، روی کلاس کلیلک راست نموده و به ترتیب گزینه های New Attribute و یا New Operation را انتخاب می نماییم.

سطوح دسترسی یا وسعت صفات:

- سطح عمومی

با نماد + نمایش داده می شود و مورد استفاده برای تمامی کلاس ها دارد.

- سطح حفاظت شده

با نماد # نمایش داده می شود و تنها برای کلاسهایی است که از کلاس اصلی ارث می برند.

- سطح خصوصی

با نماد – نمایش داده می شود و تنها در کلاس اصلی مورد استفاده قرار می گیرد.

سطوح دسترسی به اجزای کلاس

- سه سطح دسترسی وجود دارد:
  - **Public یا عمومی :** قابل استفاده توسط تمامی کلاس ها
  - **Protected یا محافظت شده:** قابل استفاده توسط کلاسهای فرزند
  - **Private یا خصوصی :** قابل استفاده تنها توسط همان کلاس

روابط بین کلاسها:

✓ ارتباط وابستگی Dependency

برای نشان دادن ارتباط میان دو کلاس که یکی از وجود دیگری آگاه است، یا از آن استفاده می کند، به کار می رود. اگر تغییری در کلاس مورد استفاده رخ دهد، کلاس استفاده کننده نیز تاثیر می پذیرد. این رابطه ضعیف ترین نوع رابطه است که می تواند یک طرفه یا دو طرفه باشد.

بیمار ⟵ پزشک

دارو ⟵

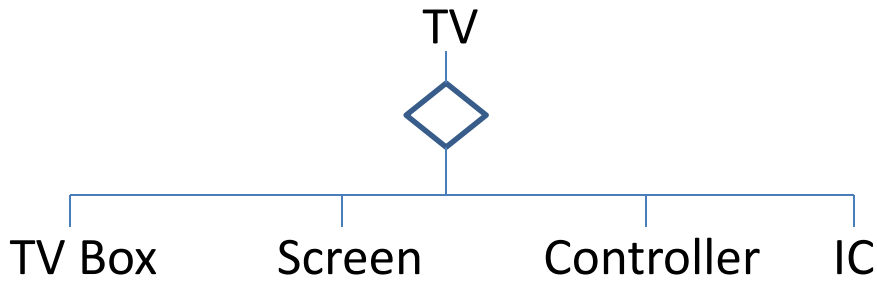
✓ ارتباط انجمنی یا تناظر Association

را بطه ای ساختاری است که نشان می دهد اجزای یک کلاس با اشیا دیگر ارتباط دارند و دو کلاس برای بازه ای از زمان می توانند با یکدیگر تعامل داشته باشند. این رابطه نسبت به رابطه قبل، قوی تر است و در مواقعی استفاده می‌شود که دو کلاس با هم ارتباط ساختاری دارند و در یک‌سطح از امنیت می باشند.

دریافت حقوق پرستار ⟷ دریافت حقوق کادر اداری

✓ ارتباط تجمعی ◇ Aggregation

گاهی اوقات یک کلاس از چند کلاس دیگر تشکیل‌شده است که نوع ویژه ای از ارتباط است که به آن تجمع گویند. کلاس کل، مالک کلاس های جز محسوب می‌شود اما کلاس جز می تواند به صورت مستقل و جدا نیز وجود داشته باشد و حتی با مرگ کلاس کل، اجزا می توانند به حیات خود ادامه دهند.

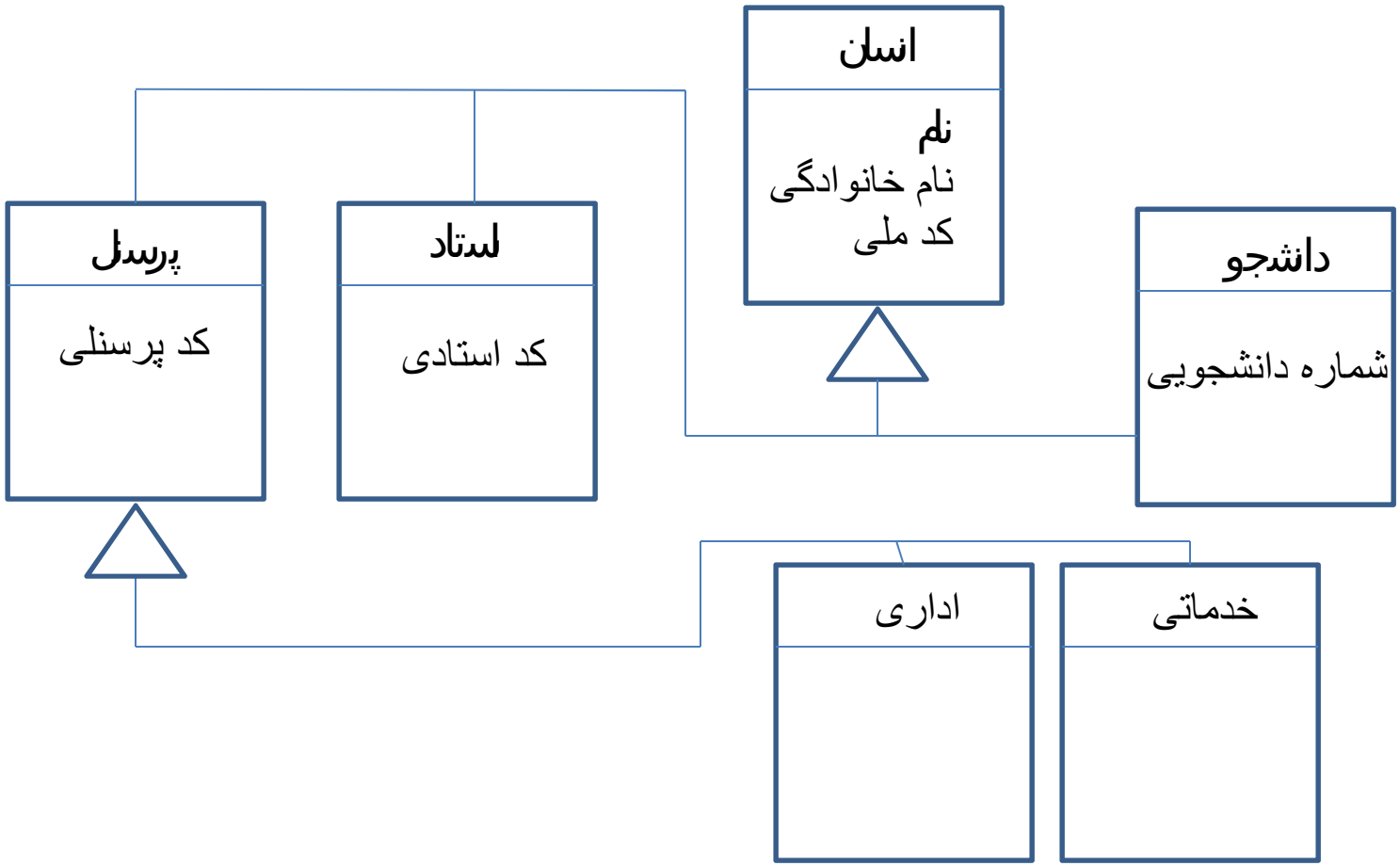


✓ ارتباط ترکیبی ◆ Composition

این رابطه نشان می دهد که یک کلاس ترکیبی از چند کلاس دیگر است به نحوی که با مرگ کلاس کل (اصلی)، اجزای آن نیز می میرند. این رابطه ها را رابطه کل به جز نیز گویند. کلاس اصلی مسئول ایجاد، مدیریت و مرگ اجزای خود است. (این رابطه قوی ترین نوع رابطه است)

✓ رابطه وراثت Generalization

وراثت یا تعمیم، ارث بری یا عمومیت، **با نماد** △ **مشخص** می **گردد و** این رابطه **نشان** می دهد که یک کلاس **حالت** خاصی (**فرزند**) **از** کلاس دیگر (**پدر**) **است** یعنی **علاوه** بر اینکه کلاس **فرزند** دارای **صفات** و عملیاتی **است**، می **تواند** برخی **صفات** و عملیات **را از** کلاس **پدر** به **ارث** برد.
برای پرهیز **از** تکرار **صفات**، **اعمال** و **ارتباطات** بین کلاسها در یک **نمودار از** این رابطه **استفاده** می‌**شود**.



روابط میان کلاسها

- کلاسهای سیستم از روی مذاکرات و صحبتهایی که مشتری مطرح می کند استخراج می شود. اسامی که شنیده می شود کاندیدی برای کلاس محسوب می شود به شرط اینکه بتوان چندین نمونه از آنرا در سیستم مشاهده کرد.
- اگر میان کلاسهای تعیین شده، رابطه ایی برقرار نباشد نمودار کلاس به فرهنگ لغات سیستم تبدیل می شود. بنابراین نیاز است روابط میان کلاسها را تعیین کرد.

رابطه تناظر (Association)

- یک رابطه ی ساده میان کلاسها را بیان می کند و با یک خط مستقیم نمایش داده می شود.

ویژگی های رابطه تناظر

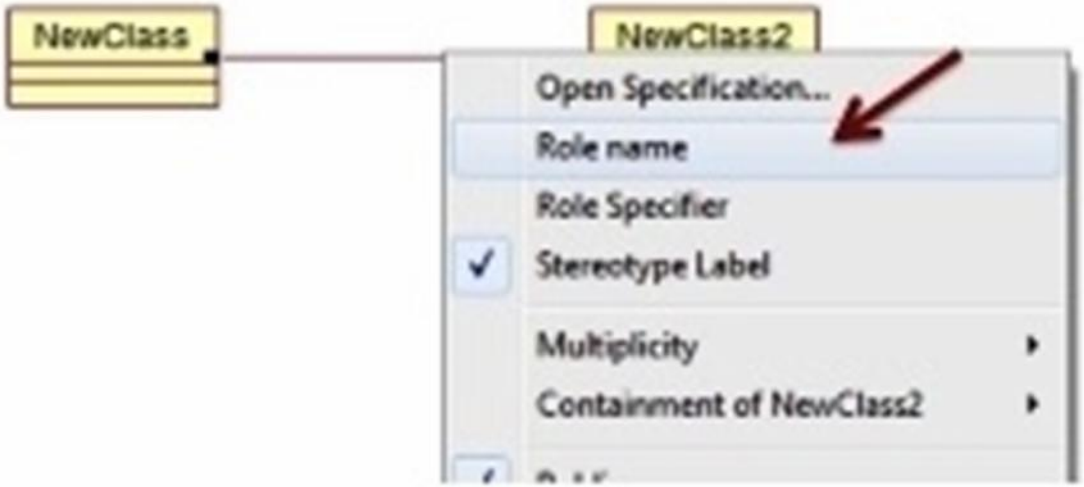
- **Multiplicity یا چند به چندی رابطه:**

نشان می دهد چند نمونه از یک کلاس با چند نمونه از کلاس دیگر در رابطه است. برای ایجاد چند به چندی یک رابطه روی رابطه کلیلک راست کرده و گزینه Multiplicity را انتخاب می کنیم.

ویژگی های رابطه تناظر (ادامه..)

- **Role Name یا نام نقش:**

در برخی از مواقع نقشهایی که هر یک از کلاسها در سیستم ایفا می کنند، در رابطه ی تناظر مشخص می شود. برای مشخص کردن نام نقش روی رابطه و در سمت کلاس مورد نظر، کلیلک راست کرده و گزینه Role Name را انتخاب می کنیم.

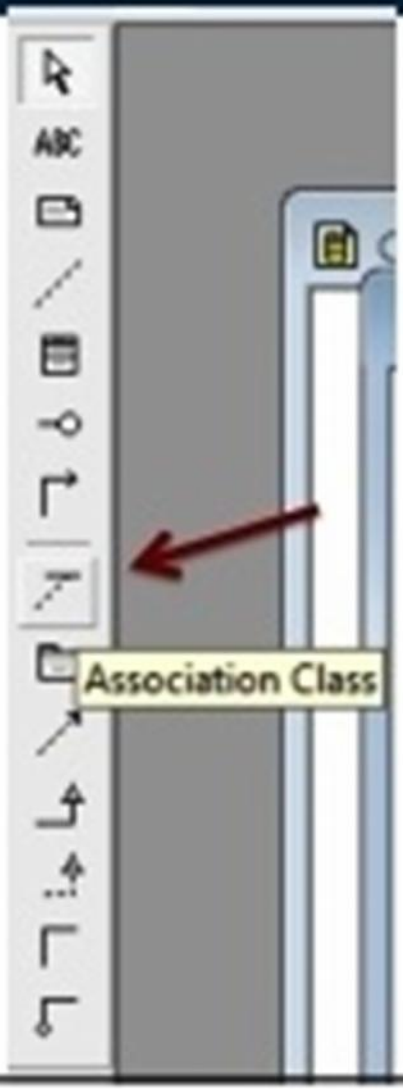


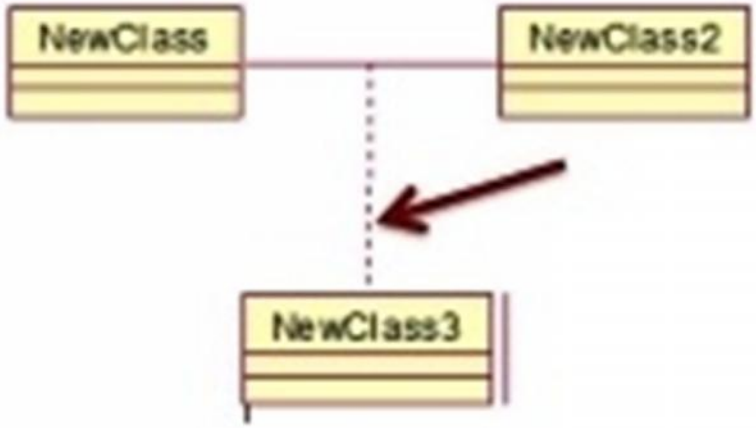
نکته 1

- **Association Class یا کلاس تناظر:**

در برخی از موارد رابطه ی تناظر خود در بر دارنده ی صفات و عملیات متعددی می باشد. در چنین مواردی می توان برای رابطه کلاسی در نظر گرفت و صفات و عملیات رابطه را در کلاس قرار داد. به چنین کلاسی، کلاس تناظر گفته می شود.با استفاده از ابزار Association Class می توان این کلاس را به رابطه متصل نمود.

Association Class نمایش

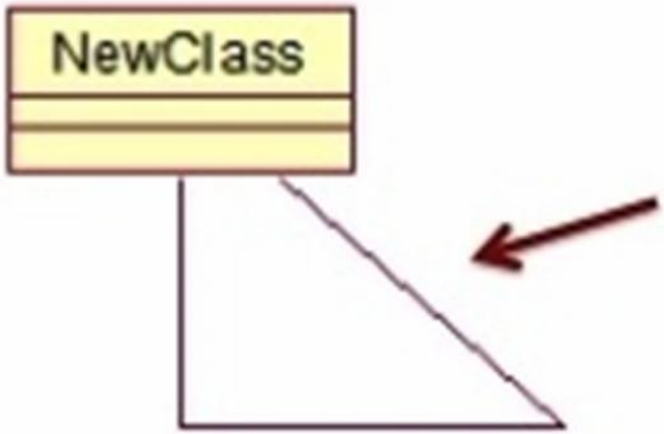




نکته 2

- **تناظر بازتابی**

رابطه ی میان کلاس با خودش را تناظر بازتابی گویند. معمولا زمانی مطرح می شود که یک کلاس اشیایی دارد که می تواند نقشهای مختلفی در سیستم ایفا کند. برای ایجاد چنین رابطه ایی کافی است رابطه ی Association را از یک کلاس به همان کلاس رسم نمود.



نکته 3

- **رابطه Navigate** (ماهیت و خاصیت)

رابطه ایی شبیه تناظر است با این تفاوت که صرفا یک طرفه و فقط از یک کلاس به کلاس دیگر رسم می شود. برای ایجاد آن از ابزار Unidirectional Association استفاده می شود.

(يك جهتی، تك سويی)

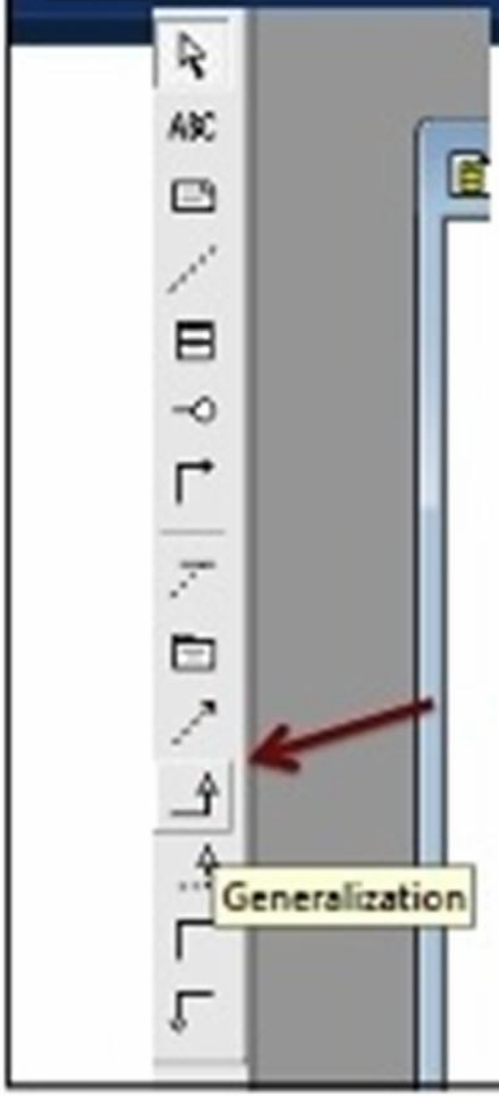
رابطه وراثت (Generalization)

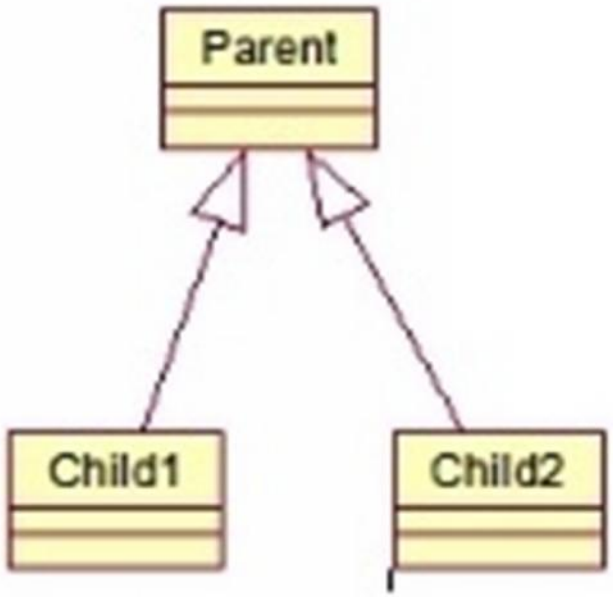
- این رابطه نوعی تناظر است که یک کلاس از صفات و متدهای کلاس دیگر استفاده می کند. این رابطه توسط تحلیل گر شناخته می شود. دو دیدگاه برای شناسایی این رابطه مطرح می شود:

  - تشخیص صفات و اعمال مشترک و جداسازی آن در کلاس پدر(تعمیم یا Generalization)

  - بازشناسی اعمال و صفات متفاوت بین نمونه های یک کلاس و جداسازی هریک از آنها در کلاس فرزند (تخصیص یا Specialization)

نمایش رابطه وراثت در Rational Rose



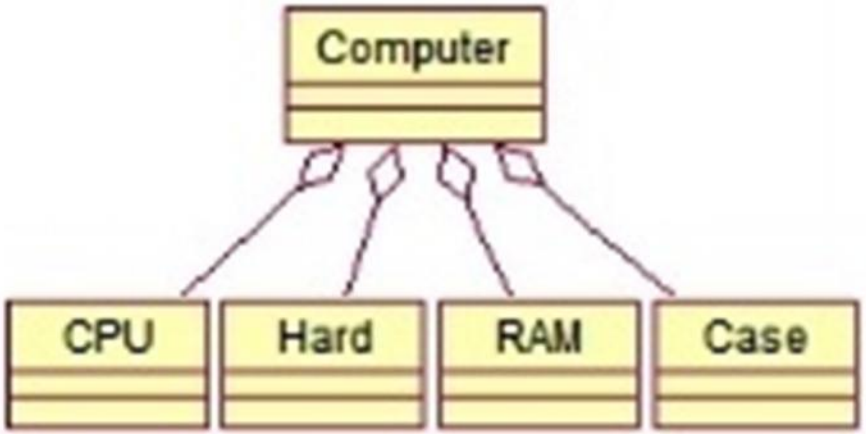


رابطه تجمع (Aggregation)

- این رابطه نوعی تناظر است که یک طرف رابطه از اهمیت بیشتری برخوردار است. به این رابطه، رابطه ی جزء – کل نیز گفته می شود. به این دلیل که رابطه تجمع بصورت سلسله مراتبی از کلاس کل در بالا و کلاسهای جزء در پایین نمودار می باشد.

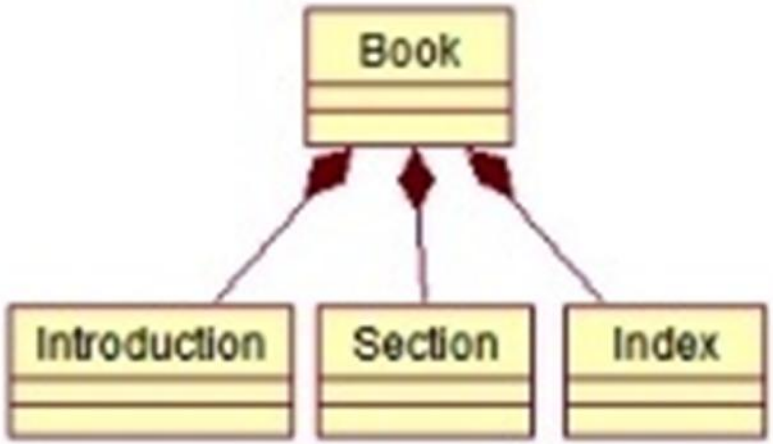
- مانند کلاس های CPU ، Hard، RAM و Case که جزعی از کلاس Computer هستند.

نمایش رابطه تجمع در Rational Rose



رابطه ترکیب (Composition)

- این رابطه نوعی رابطه تجمع قوی است بدین ترتیب که هر یک از کلاسهای جزء به تنهایی نمی تواند مطرح گردد بلکه همواره در ترکیب با کلاس کل در سیستم مطرح می گردند. به عبارت دیگر هر کلای جزء وابسته به کلاس کل است. بعنوان مثال کلاسهایی مثل مقدمه ، فهرست مطالب و فصل ترکیبی از کلاس کتاب هستند که به تنهایی مفهمومی ندارند.



راط Interface

مجموعه ای از عملیات است که برخی ابعاد رفتار یک کلاس را مشخص می نماید و مجموعه ای از عملیاتی است که یک کلاس به کلاس های دیگر نشان می دهد.

یک راط به همان شیوه یک کلاس مدلسازی می‌شود یعنی با شمایل مستطیل اما بدون هیچ صفتی. راط فقط عملیات است، صفت ندارد. روش دیگر نمایش آن، استفاده از یک دایره کوچک با یک خط متصل به کلاس است.

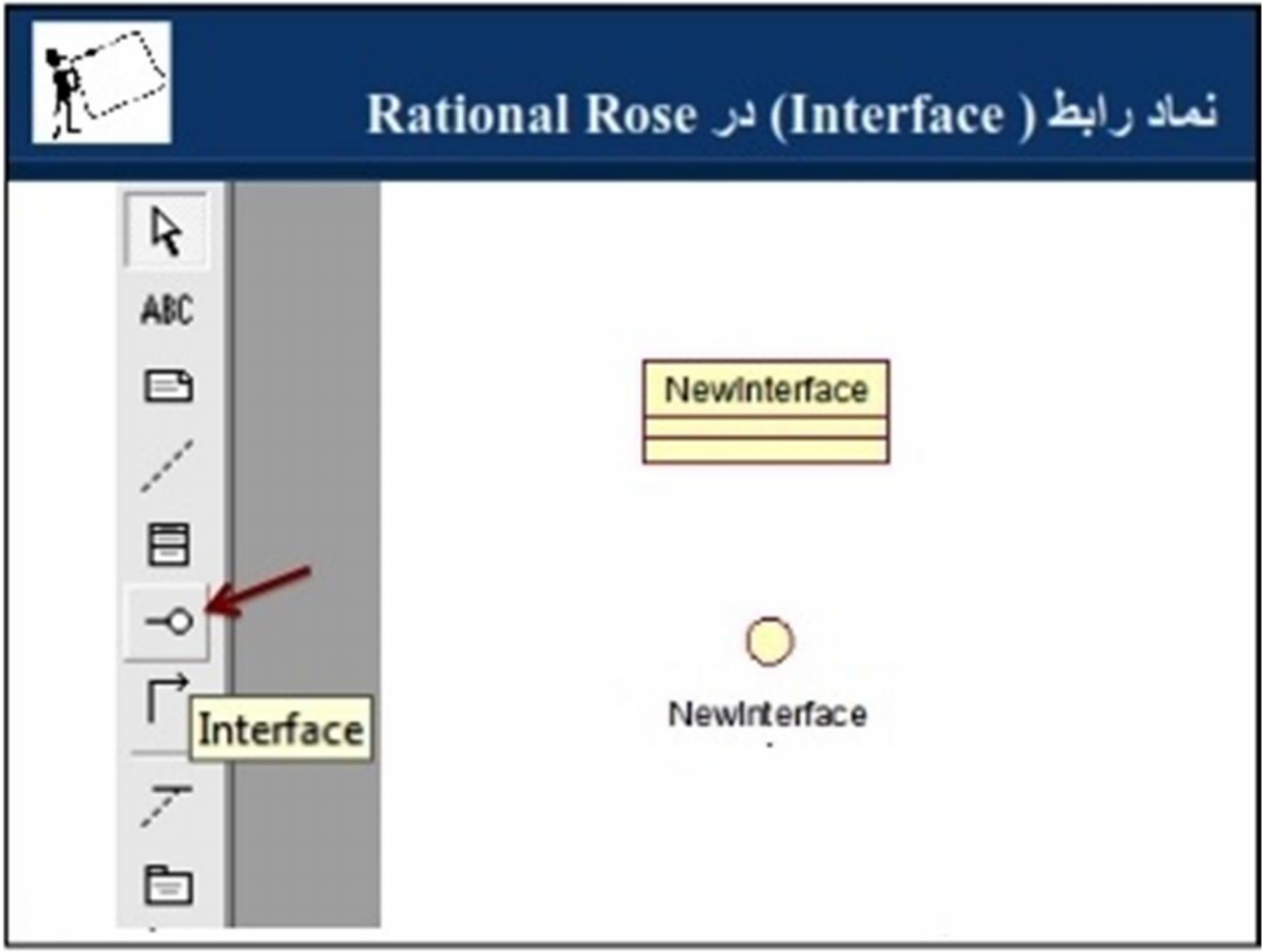
محقق سازی Realization

ارتباط بین یک کلاس و یک راط، محقق سازی نامیده می شود. این ارتباط به صورت یک خط چین با یک مثلث بزرگ توخالی که به راط متصل است و به آن اشاره می کند، مدلسازی می‌شود.

رابط (Interface) و محقق سازی (Realization)

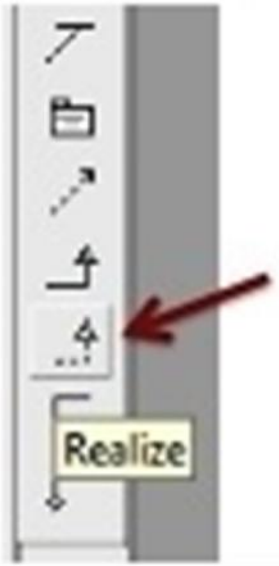
- رابطها (واسط ها) مجموعه ایی از عملیات است که رفتار یک کلاس را مشخص می کنند و توسط آن کلاس به کلاسهای دیگر عرضه می شود.

- در Rational Rose رابط ها همانند کلاس مدلسازی می شوند (به شکل مستطیل) و هیچ صفتی ندارند و فقط شامل عملیات اند. روش دیگر مدلسازی رابط با استفاده از نماد Interface می باشد.


نماد رابط (Interface) در Rational Rose
ABC
Interface
NewInterface
NewInterface

رابط (Interface) و محقق سازی (Realization)

- به منظور بکارگیری مجدد عملیات کلاسها از واسط ها استفاده می شود. عملیات در رابط تعریف شده و توسط سایر کلاسها مجددا استفاده می شود.
- رابطه ی میان یک کلاس با کلاس رابط ، محقق سازی (Realization) نامیده می شود . در Rational Rose با نماد Realize مشخص می گردد.



Use Case Diagram

- اجزای نمودار Use Case
  - Actor
  - Use Case
- ارتباط میان Actor و Use Case
- روابط میان Use Case ها
  - include
  - extend
  - generalization

نمودار مورد کاربر Use Case Diagram

این نمودار درسال 1992 توسط جاکوبسون یکی از بنیانگذاران UMLابداع شد. این نمودار کمک می کند، حصار میان برنامه نویسان و مشتریان شکسته‌شود. چرا که شرحی مفصل از اشیا مختلفی را که با سیستم تراکنش دارند، ارایه می کند.

Actor (بازیگر، کنشگر، هنرپیشه یا عامل) : نقشی که کاربر در ارتباط با سیستم ایفا می کند.

انواع عامل:
- عامل اصلی
- عامل فرعی
- سخت افزارهای خارجی
- سیستم های دیگر

عامل اصلی (Principal Actor): افرادی که از امکانات اصلی سیستم استفاده می کنند را گویند، به عنوان مثل در یک سیستم خودپرداز، عامل اصلی، مشتریان هستند.

عامل فرعی (Secondary actor): افرادی که مدیریت و نگهداری وضایف یک سیستم را انجام می دهند، به عنوان مثل، در سیستم خودپرداز، مسئول پرکردن صندوق اسکناس دستگاه، نقش تانویه یا فرعی را ایفا می کند.

سخت افزارهای خارجی (External Hardware):شامل ابزارها وسخت افزارهایی است که در

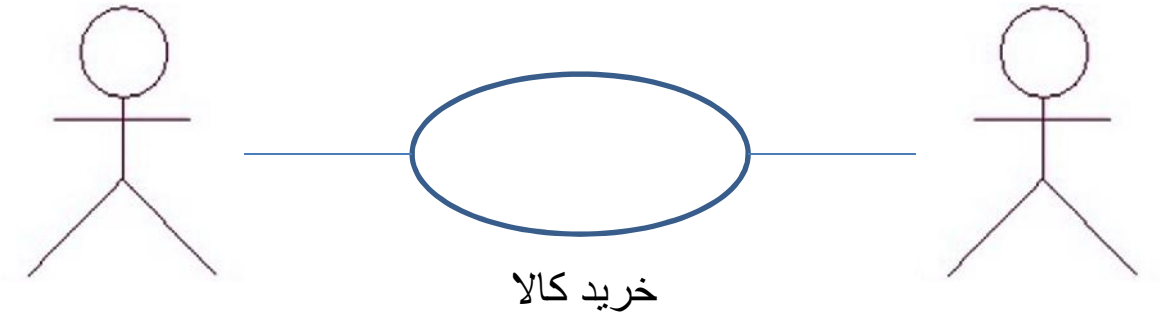
سیستم مورد نیاز است، مثلا در سیستم خودپرداز، چاپگری که صورتحساب را چاپ می کند نقش نرم افزار خارجی را ایفا می کند.

سیستم های دیگر (Other system): سیستم های دیگری که با سیستم مورد نظر ارتباط دارند، نقش سیستم های دیگر را ایفا می کنند. به عنوان مثال در سیستم خودپرداز، سایر سیستم هایی که از نظر مخابراتی و شبکه ای با خودپرداز ارتباط برقرار می کنند نقش سیستم های دیگر را ایفا می کنند.

نکته: شکل اصلی نمودار مورد کاربر به صورت بیضی است و نقش های اصلی در آن به صورت آدمک ترسیم می‌شود.

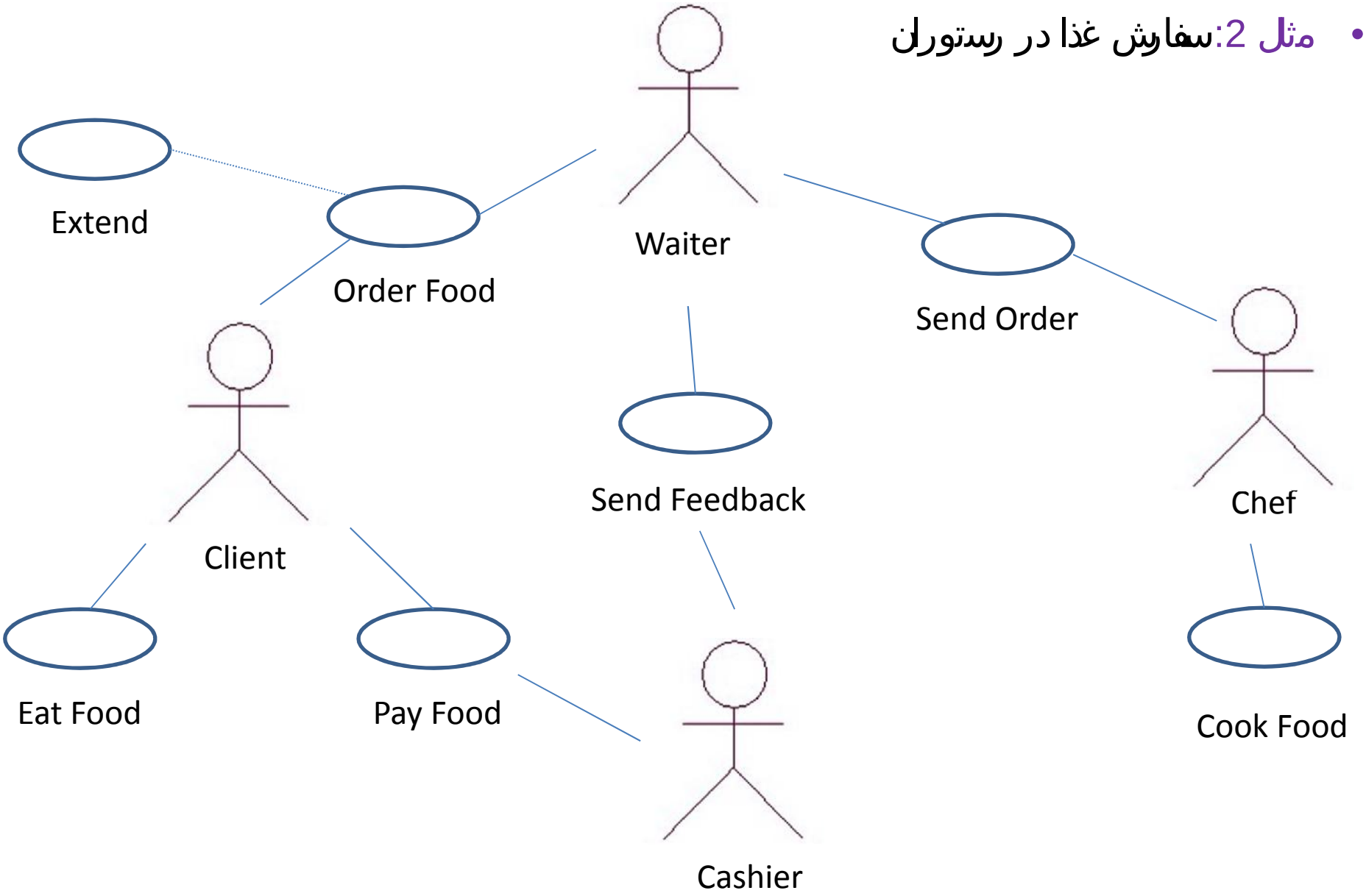
مثال: خرید کالا از فروشگاه

عامل اصلی: 1. مشتری 2. فروشنده



در کارهای دستی مورد کاربر، در داخل بیضی نوشته می‌شود اما در Rational Rose زیر نماد بیضی نوشته می‌شود.

- مثل 2:سفارش غذا در رستوران



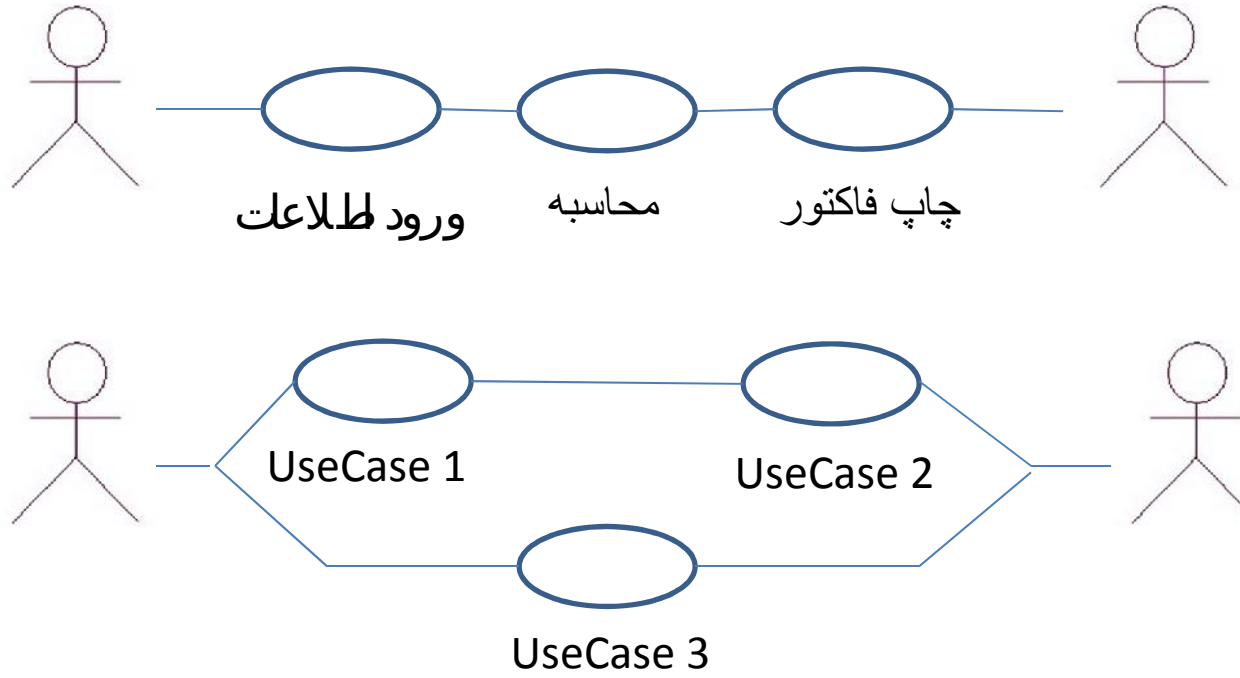
رابطه گسترش یا توسعه (Extend):

در اینجا مر حله ای وجود دارد، مثلا غذا به مشتری تحویل داده شده است و مشتری قصد تکرار سفارش دیگری را دارد. در این گونه موارد که قصد تکرار یک کاربردی میان دو عامل وجود دارد، از رابطه گسترش یا توسعه داده شده با فرمان Extend استفاده می شود. **در مثل فوق در «سفارش چای و قهوه» ار این رابطه استفاده شده است.**

Include:

گاهی اوقات میان دو عامل چندین مورد کاربرد وجود دارد که به آن Include گویند.

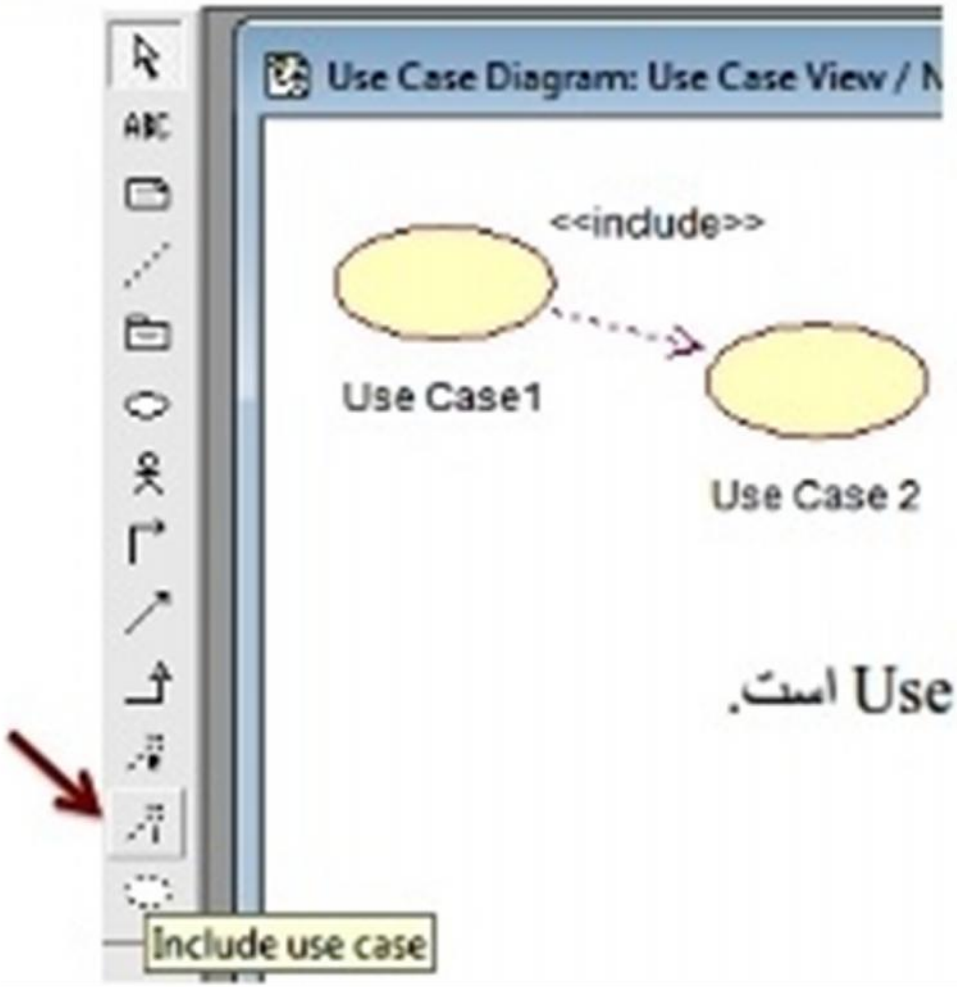
مثل: صدور فاکتور مشتری



include

- زمانی که یک Use case عملیات Use Case دیگر را مورد استفاده قرار میدهد این رابطه مطرح می شود. اصطلاحا گفته می شود عملیات تعریف شده در یک Use Case دربردارنده عملیات تعریف شده در Use Case دیگر است.
- به عبارت دیگر می توان عملیات مشترک را در یک Use Case تعریف کرد .سایر Use Case ها این عملیات مشترک را عینا مورد استفاده قرار می دهند.
- بعنوان مثال در سیستم کتابخانه، Use Case های *امانت کتاب* و *تمدید امانت* ، شامل Use Case *کنترل عضویت* می باشد.

نماد رابطه include در Rational Rose



Use Case1 شامل Use Case2 است.

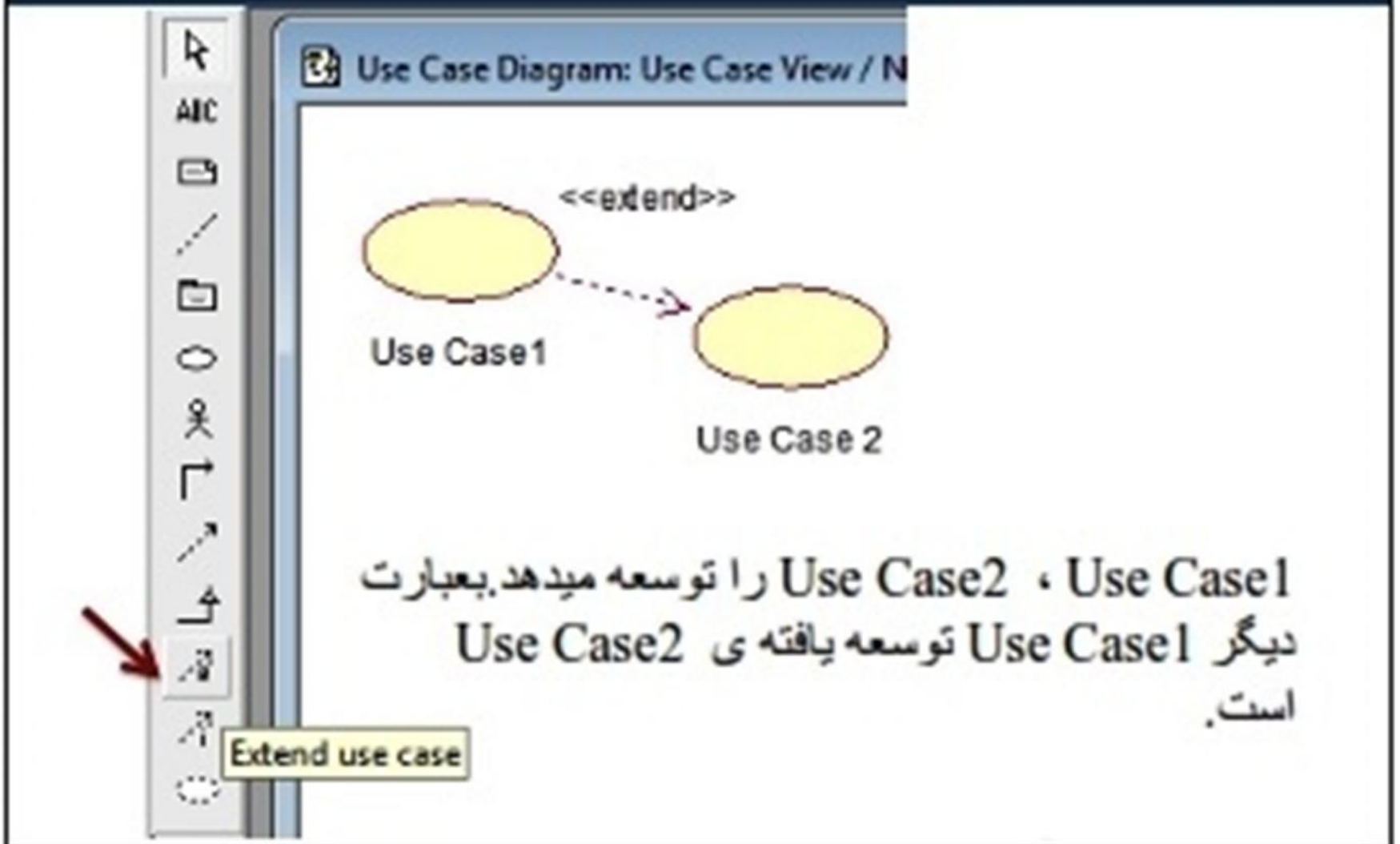
extend

- هرگاه یک Use Case ، Use Case دیگر را توسعه دهد، رابطه extend میان آنها برقرار می شود.

- Use Case های توسعه دهنده در واقع نوع خاصی از Use Case اصلی است. Use Case توسعه یافته بعنوان Use Case اجباری و هر یک از Use Case های توسعه دهنده بعنوان Use Case اختیاری محسوب می شوند که بر حسب نیاز کاربر اجرا می گردند.

- بعنوان مثال در سیستم بانک، *استعلام موجودی از حساب* توسط *چاپ موجودی* و *نمایش موجودی* توسعه داده شده است.

نماد رابطه extend در Rational Rose



Use Case1 ، Use Case2 را توسعه میدهد.بعبارت دیگر Use Case1 توسعه یافته ی Use Case2 است.

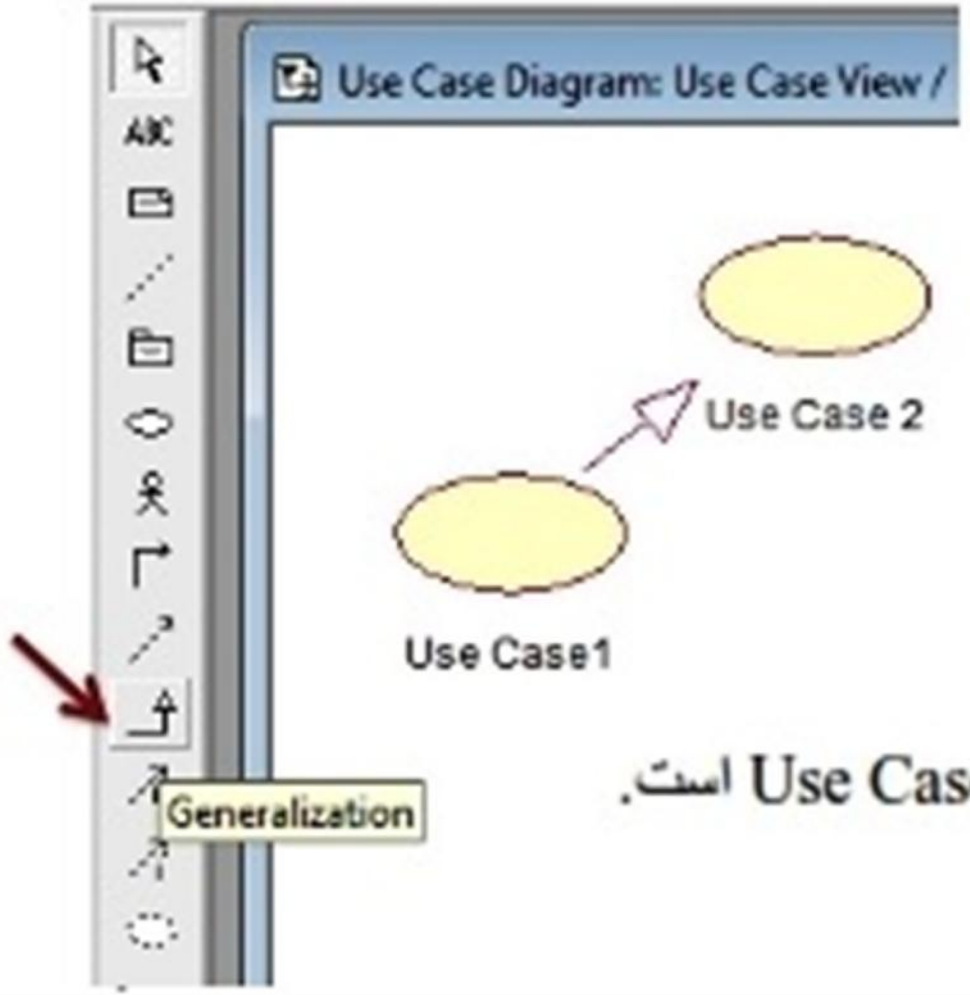
generalization

- زمانی که یک Use Case حالت کلی یک یا چند Use Case دیگر باشد ، رابطه ی وراثت میان آنها تعریف می گردد.

- بعنوان مثال در سیستم بانک *افتتاح حساب* یک Use Case عمومی است که خود در بر دارنده انواع افتتاح حساب مانند *افتتاح حساب جاری* ، *افتتاح حساب قرض* الحسنه می باشد.

- Use Case های فرزند مفهوم تعریف شده در Use case پدر را مورد استفاده قرار می دهند.

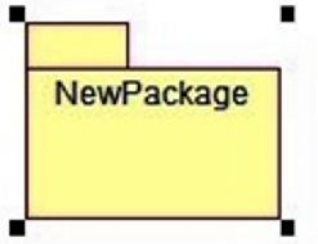
نماد رابطه Generalization در Rational Rose



Use Case1 ، نوعی از Use Case2 است.

گروهبندی Grouping

در برخی نمودارهای مورد کاربرد، ممکن است بخواهیم تعدادی از موردهای کاربرد را سازماندهی کنیم. این زمانی است که یک سیستم شامل شماری از زیر سیستم ها باشد. گروه مرتبط به مورد کاربرد را در داخل یک بسته Package (یک پوشه برچسب دار) قرار می دهیم.



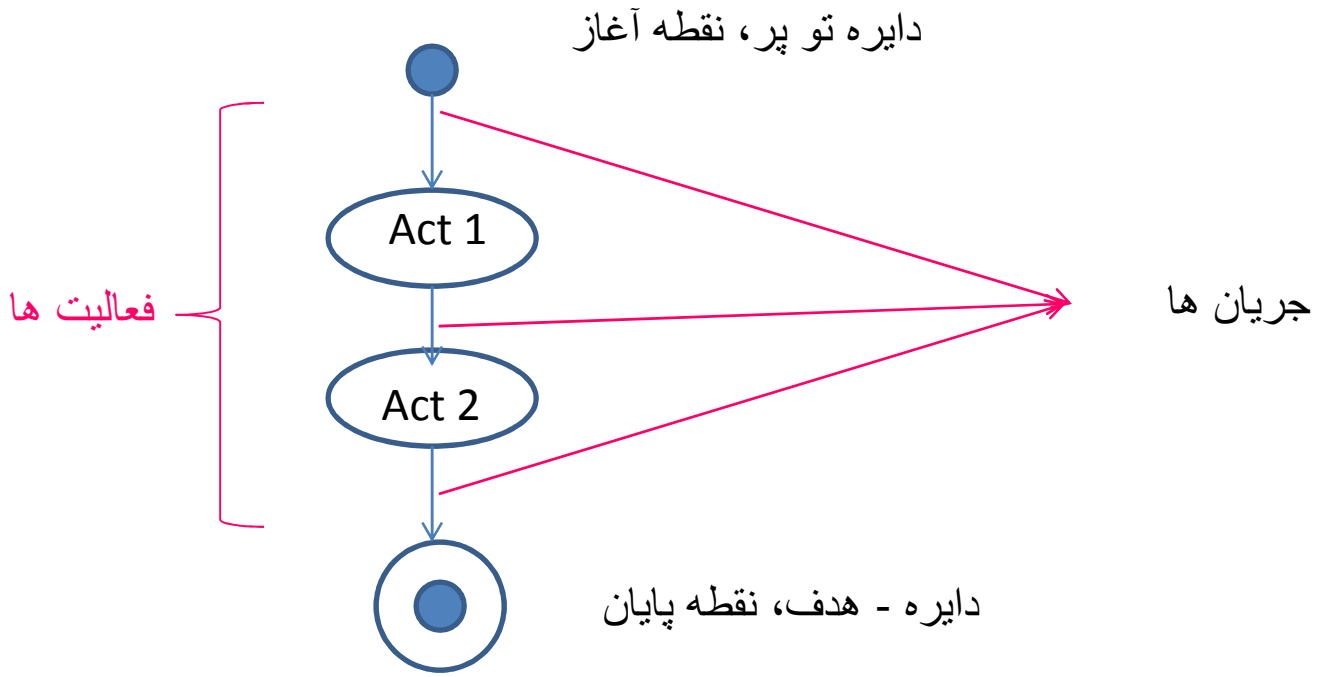
Activity Diagram

نمودار فعالیت Activity Diagram

این نمودار بیانگر یکسری از عملیات های پشت‌سر هم است که شباهت بسیاری با فلوچارت ها دارد. این نمودار نشان دهنده جریان گردش کار و عملیات، از رفتار یک سیستم است.

هرفعالیت به صورت یک بیضی نمایش داده می‌شود، ارتباط میان فعالیت ها از طریق خطوطی به نام جریان Transition انجام می گردد. زمانی که یک فعالیت تمام می‌شود، این جریان به کار افتاده و فعالیت بعدی را نشان می دهد. نقطه شروع هر فعالیت به صورت یک دایره توپر و نقطه پایانی به صورت یک دایره – هدف نمایش داده می‌شود.

نکته مهم: نقطه آغاز همیشه یکی است اما نقطه پایان می تواند بیش از یکی باشد.



❑ اجزای نمودار فعالیت:

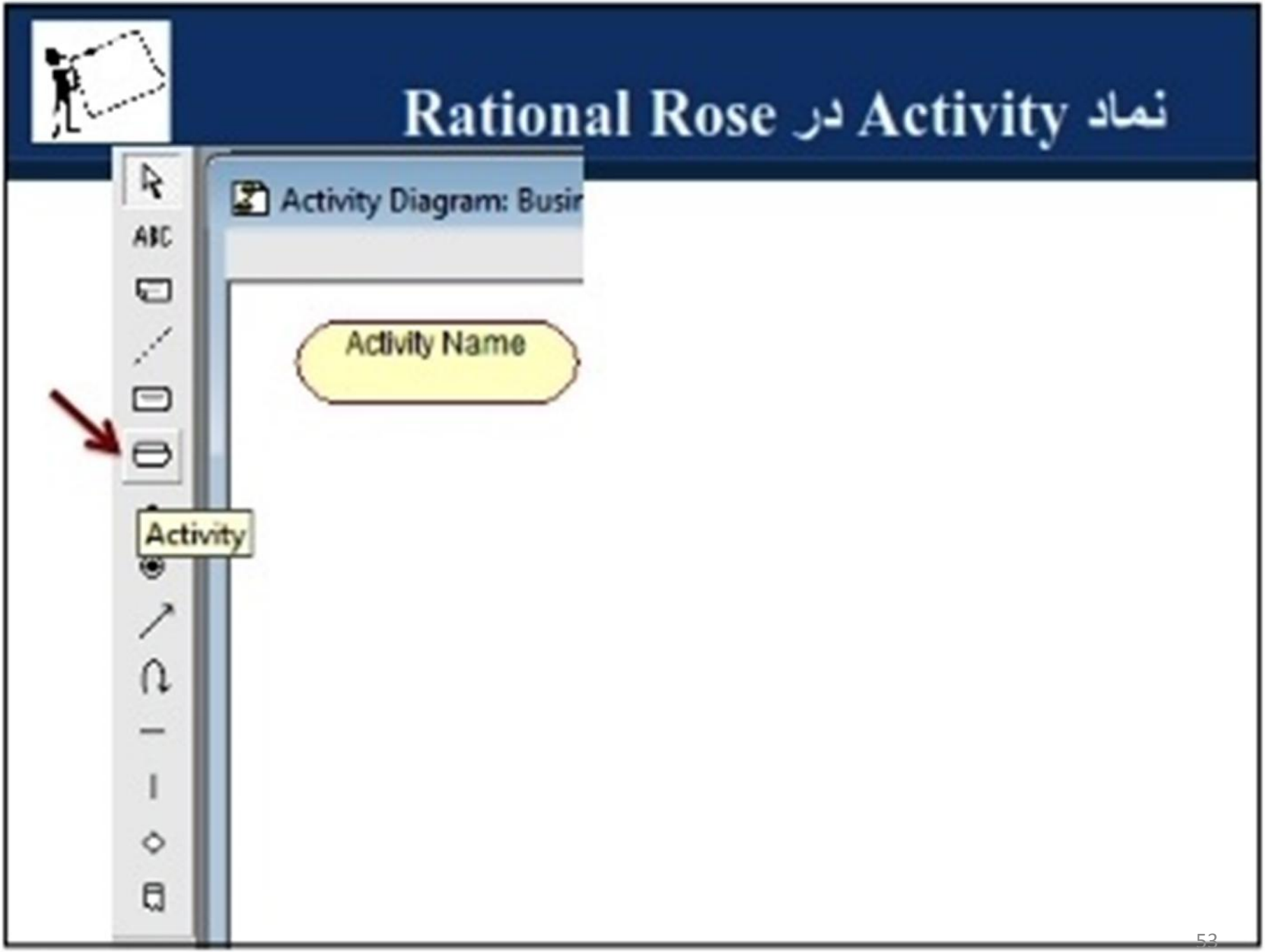
- Activity
- انتقال یا گذر (Transition)
- نقاط تصمیم
- نقاط آغازین و پایانی
- همگام سازی
- Swim lane

Activity

- نموار فعالیت مشابه فلوچارت است که برای نشان دادن جریان کار سیستم بکار می رود. (نمایش کنترل از یک فعالیت به فعالیت دیگر)

- هر فعالیت نشان دهنده ی یک عملکرد خاص است که در سیستم انجام می شود.

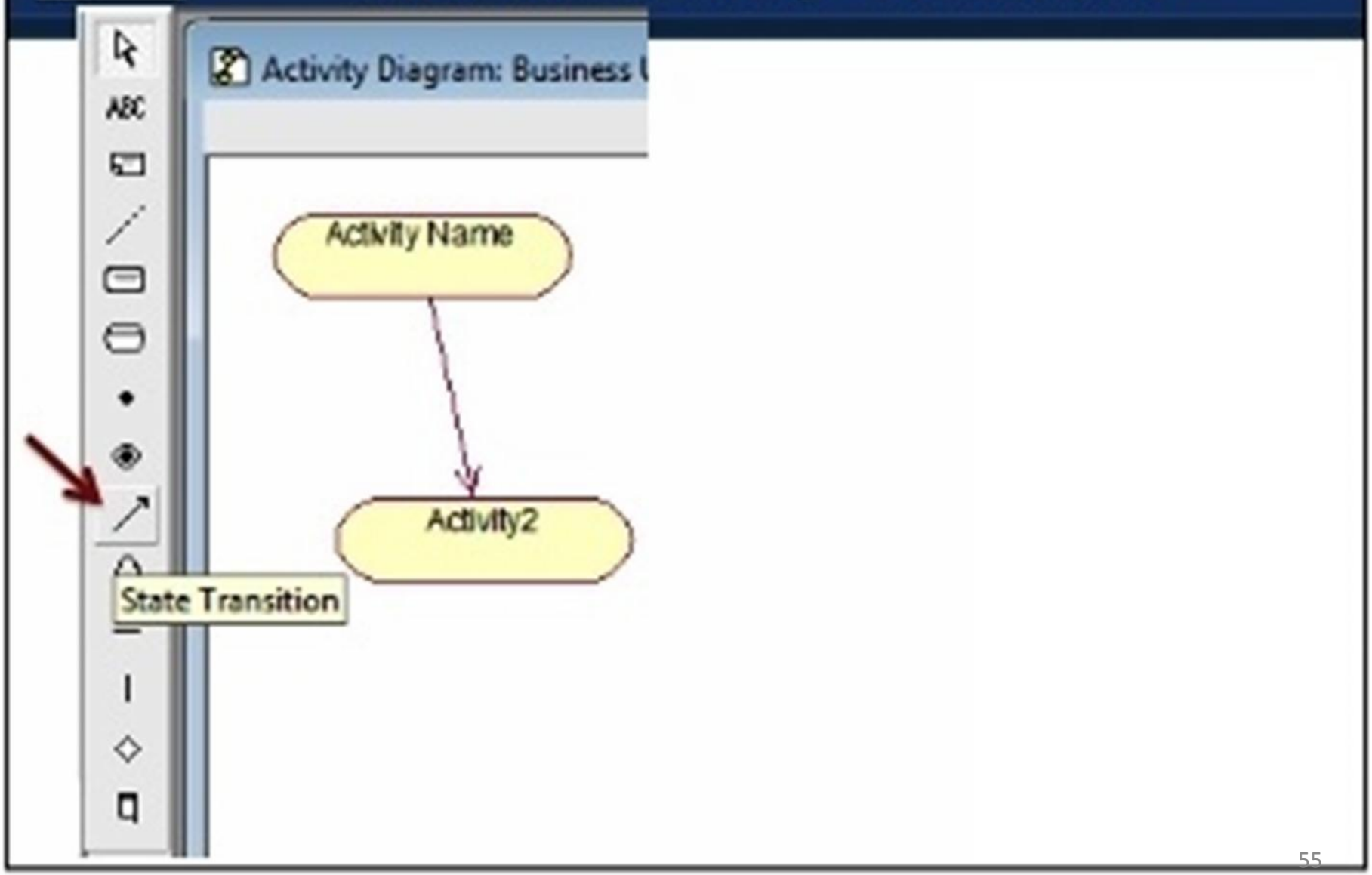
- در Rational Rose فعالیت با بیضی نشان داده می شود که نام فعالیت درون آن نوشته می شود.


Rational Rose در Activity نماد
Activity Diagram: Busir
Activity Name
Activity

انتقال یا گذر (Transition)

- برای نمایش جریان کنترل از یک فعالیت به فعالیت دیگر بکار می رود
- در Rational Rose گذر با یک فلش نمایش داده می شود.(ابزار (State Transition

نماد Transition در Rational Rose

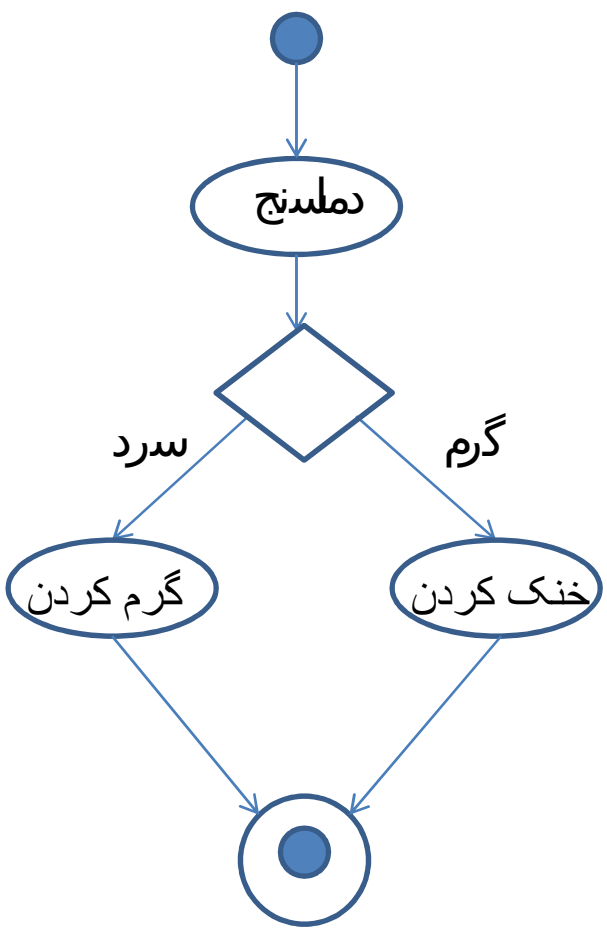


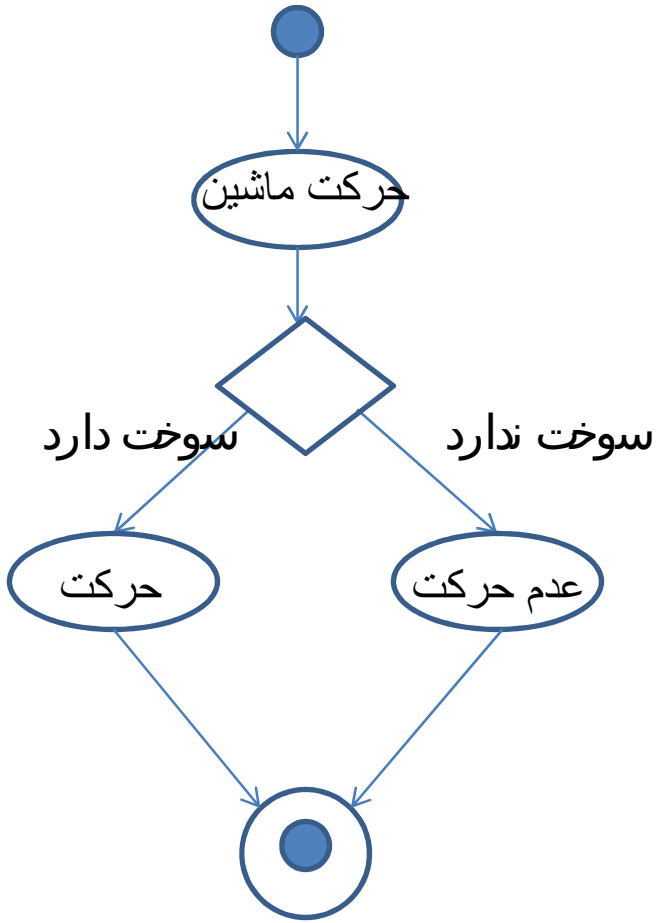
عنصر تصمیم گیری یا شرط Decision

در صورتی که بخواهیم از عبارات شرطی استفاده کنیم از عنصر شرط به شکل یک لوزی تو خالی استفاده می کنیم به نحوی که شروط در داخل کروشه ها و در بالای لوزی نوشته می شوند.

مثال1 : نرم افزاری برای متعادل نگه داشتن دمای اتاق

مثل 2: حرکت کردن ماشین

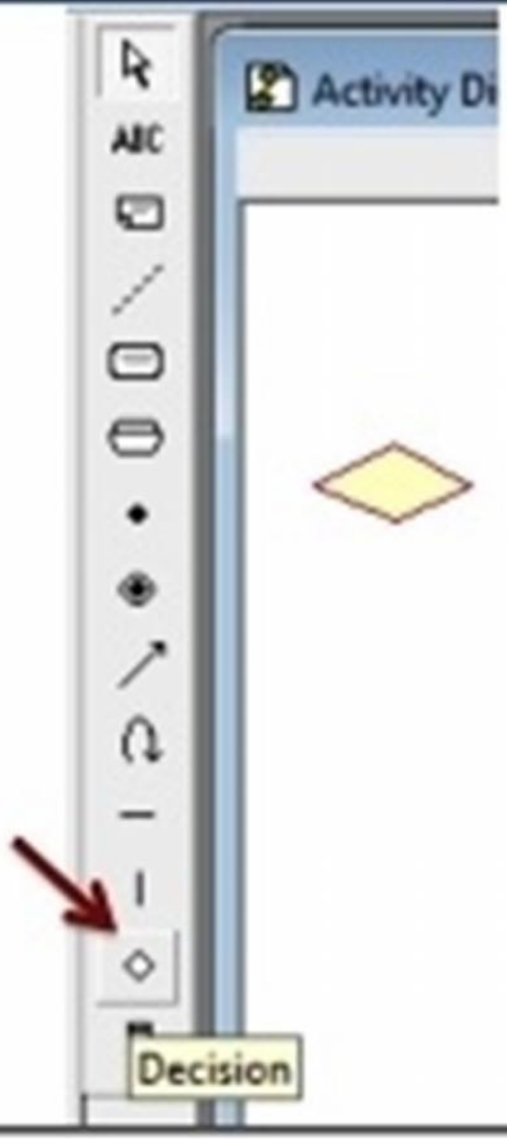




نقاط تصمیم

- به منظور شاخه شدن جریان کنترل در شرایط مختلف از ساختار تصمیم استفاده می شود. در واقع برای بررسی شرط های سیستم از این عنصر استفاده می شود.
- نقاط تصمیم در Rational Rose با لوزی نمایش داده می شود.

نماد نقاط تصمیم در Rational Rose



نقاط آغازین و پایانی

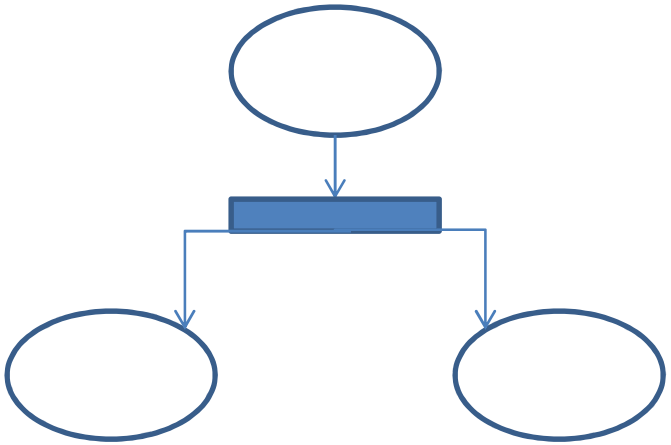
- برای نمایش نقاط شروع و پایان مجموعه فعالیتها بکار می رود.

- هر جریان کاری(مجموعه فعالیتها)، دارای یک نقطه ی شروع می باشد. همچنین هر جریان کاری می تواند در بخشهای مختلف و در شرایط گوناگون خاتمه یابد.

نماد نقاط شروع و پایان در Rational Rose

عنصر همزمانی یا همروندی Synchronization

این عنصر را با یک خط افقی تو پر نمایش می دهند و هنگامی مورد استفاده قرار می گیرد که بخواهیم از دو یا چند فعالیت به یک فعالیت برویم.

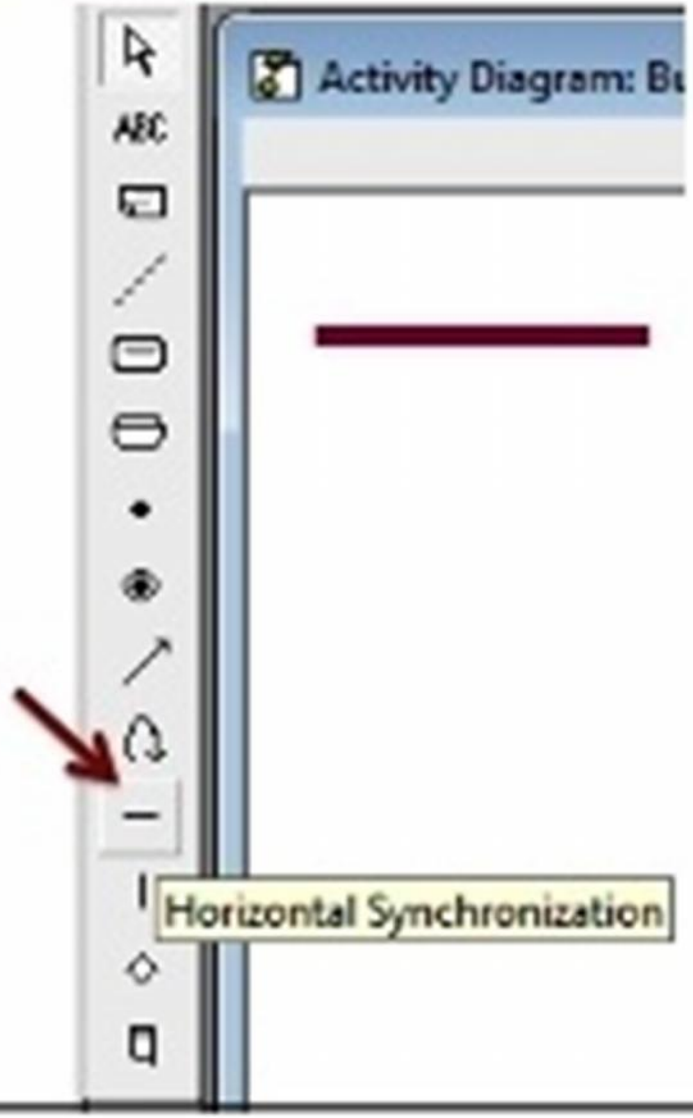


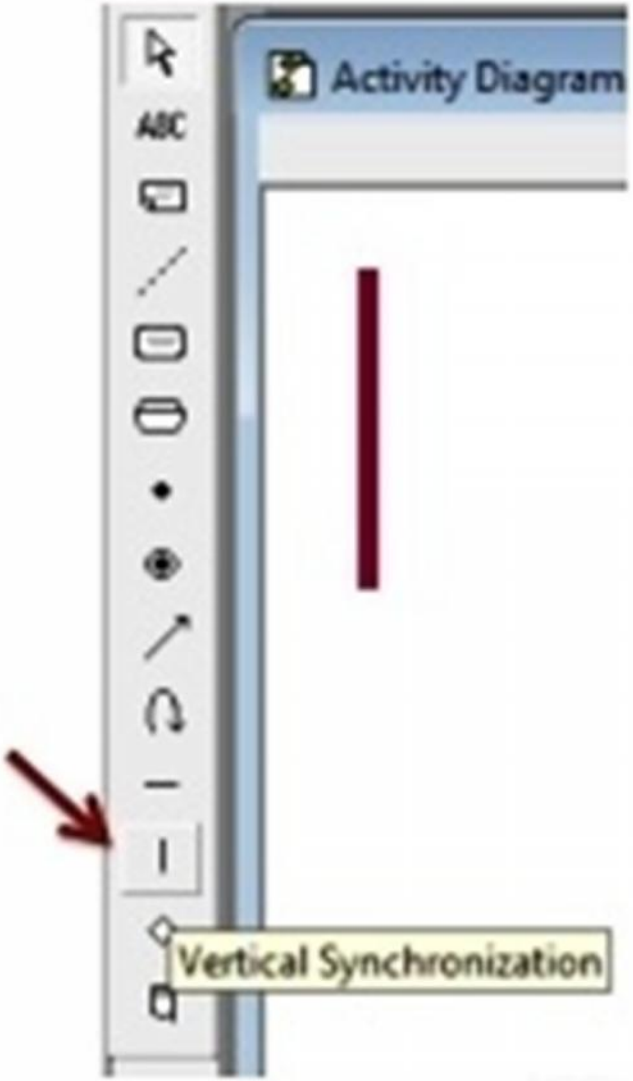
می توان از یک فعالیت به چند فعالیت رفت (join) و از چند فعالیت به یک فعالیت بازگشت (fork).

همگام سازی (Synchronization)

- برای نمایش فعالیتهایی که بطور موازی انجام می شود بکار می رود.
- در Rational Rose از میله های همگام سازی افقی یا عمودی (Horizontal synchronization یا Vertical synchronization) استفاده می شود.

نماد همگامسازی در Rational Rose



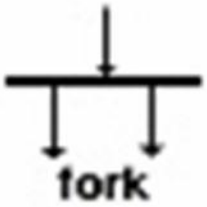


انواع همگام سازی (Synchronization)

- دو نوع همگامسازی مطرح می شود:

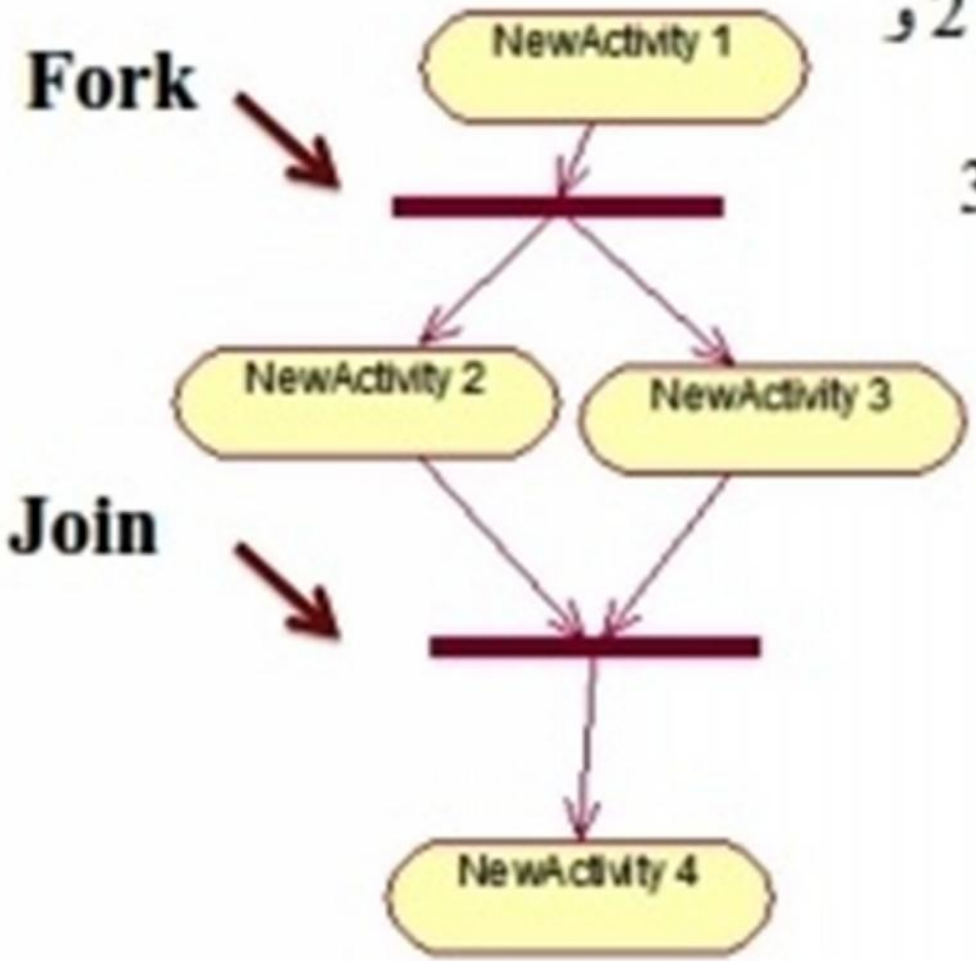
  - Join: با اتمام چند فعالیت، یک فعالیت شروع می شود.

  - Fork : با اتمام یک فعالیت، چند فعالیت شروع می شوند.

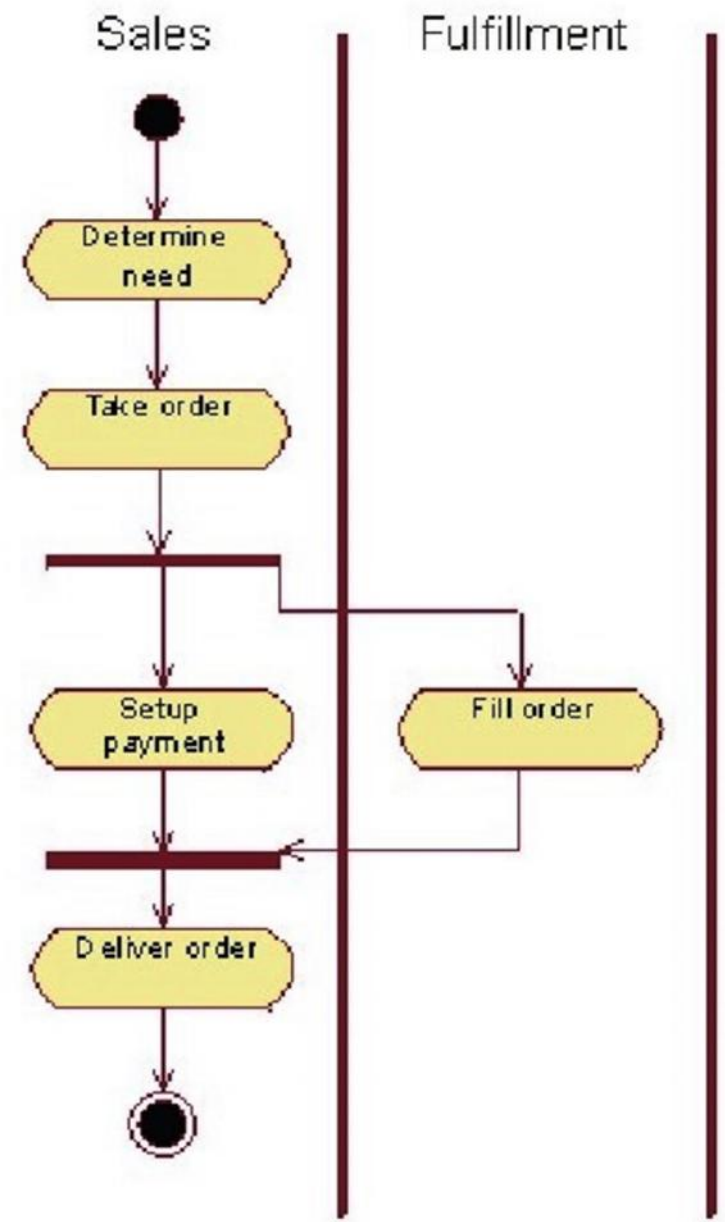


مثال

پس از انجام فعالیت 1، فعالیتهای 2 و 3 بطور همزمان انجام می شود.
همچنین پس اتمام فعالیتهای 2 و 3 فعالیت 4 انجام می شود.



عنصر خط شنا یا حوضچه Swim Lane



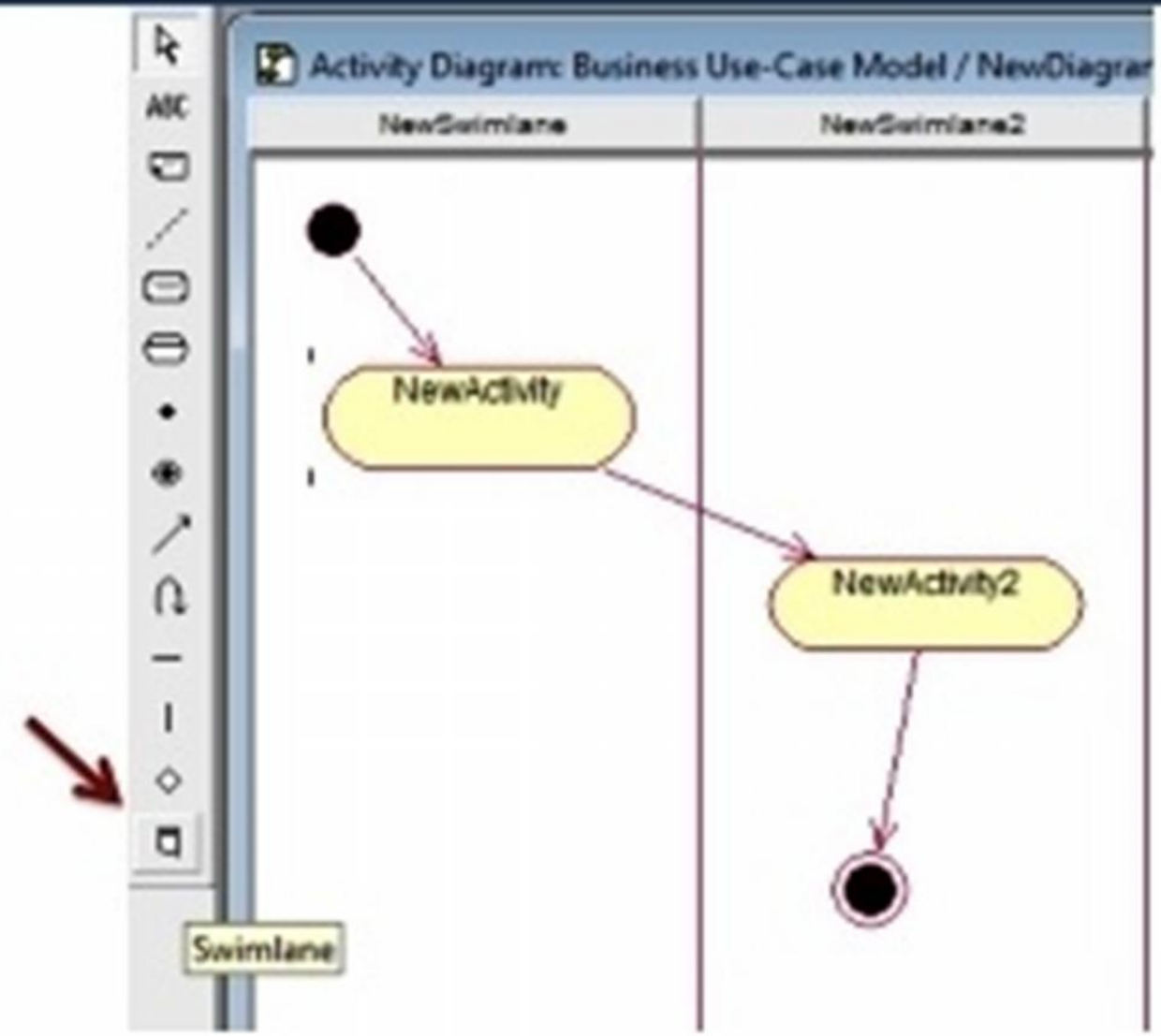
برای این که نشان دهیم فاعل یک فعالیت چه کسی است از نماد خط شنا استفاده می کنیم.
به این ترتیب که با ترسیم خط چین های عمودی و تفکیک نمودار به بخش های مختلف و سپس نام گذاری هر بخش با نام فاعل فعالیت های آن بخش و در نهایت قراردادن فعالیت های هر بخش در آن بخش، می توان این کار را انجام داد.

Swim lane

- برای قطعه بندی نمودار فعالیت مورد استفاده قرار می گیرد. منظور از قطعه بندی نمودار آن است که مشخص می کند چه عنصری مسئول یک فعالیت است.

- در Rational Rose از ابزار Swimlane برای این منظور استفاده می شود که با یک خط عمودی نمایش داده می شود. نام مسئول هر فعالیت در بالای خطوط نوشته می شود.

نماد swimlane در Rational Rose



تفاوت نمودار فعالیت با فلوچارت

در فلوچارت فقط فرایندهای پشت سرهم نشان داده می‌شود و قادر به نمایش همروند نیستیم. در حالی که نمودار فعالیت، صرفا توالی های ضروری که باید ادامه یابد را نشان می دهد و ترسیم گر را تشویق به نمایش رفتارهای موازی سیستم می کند که از توالی های غیر ضروری پرهیز شود. این نمودار را می توان برای تحلیل یک مورد-کاربرد و نیز تحلیل اعمال کلاس به خوبی به کار برد.

Sequence & Collaboration Diagrams

- نمودارهای توالی (Sequence) و همکاری (Collaboration) تعاملات میان اشیاء سیستم را نمایش می دهند. (همانطور که می دانیم اشیاء از طریق ارسال پیام با یکدیگر در تعامل اند).

- تفاوت این دو نمودار آن است که نمودار توالی، ترتیب زمانی ارتباطات میان اشیاء را بیان می کند، درحالیکه نمودار همکاری، صرفا ارتباطات اشیاء را بدون توجه به ترتیب زمانی ارسال پیامها، نمایش می دهد.

نمودار های تعامل

در هر سیستمی، اشیاء درون سیستم بوسیله‌ی ارسال پیام‌هایی با اشیاء دیگر تعامل و همکاری دارند. برای نمایش این نوع رفتار از نمودار تعامل استفاده می‌شود. نمودارهای تعامل ، مدل‌هایی هستند که چگونگی تعامل و همکاری مجموعه‌ای از اشیاء را از نظر رفتاری توصیف می‌کنند. در این نمودارها، پیام‌های ارسالی بین اشیاء مشخص می‌شود. نمودارهای تعامل به دو نوع نمودارهای توالی و نمودارهای همکاری تقسیم می‌شوند.

نمودار توالی Sequence Diagram

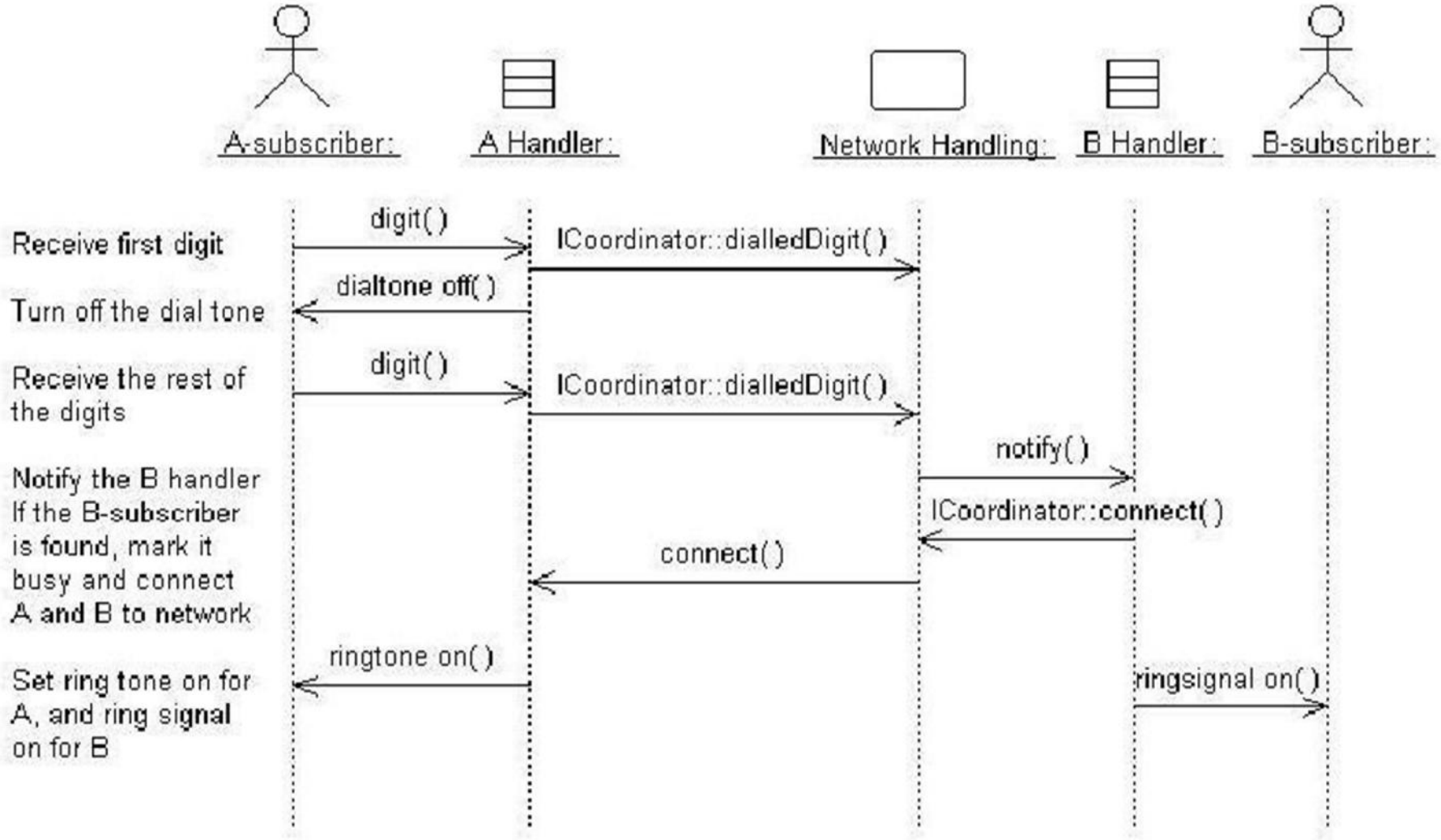
یکی از نمودارهای رفتاری است که نشان می دهد چه پیغام هایی با چه ترتیب زمانی میان اشیا در یک فرایند رد و بدل می شوند.

در واقع چگونگی انجام یک فرایند به صورت پشت سر هم در این نمودار ترسیم می شود.

در یک نمودار توالی، یک شیء را بالای یک خط‌چین عمودی به نمایش در می‌آورند. این خط‌چین عمودی به خط عمر معروف است و نشان دهنده حیات یک شیء در دوره‌ای از زمان است که در این تعامل شرکت دارد. پیام‌های ارسالی بین اشیاء با پیکان جهت‌دار نشان داده می‌شود. ترتیب ارسال پیام‌ها از بالا به پایین است.

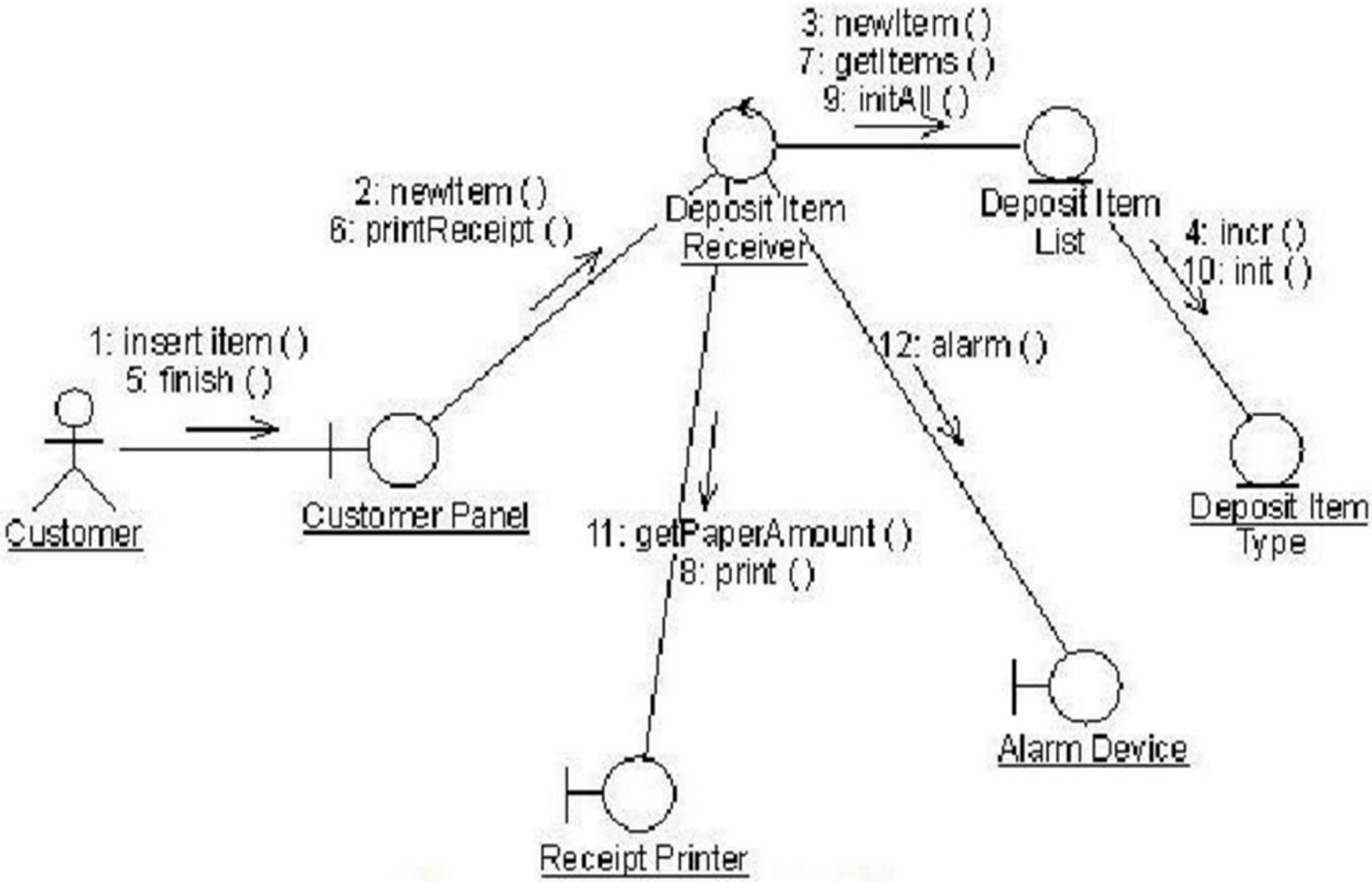
علاوه بر خط عمر که یکی از مشخصه‌های متمایز کننده نمودار توالی از همکاری است، کانون تمرکز نیز این گونه است. کانون تمرکز، مستطیل باریک و بلندی است که روی خط عمر قرار می‌گیرد و نشان دهنده دوره‌ای زمانی است که در حین آن شیء در حال انجام یک عمل است که این عمل می‌تواند مستقیما توسط یک رویه یا از طریق یک رویه تودرتو انجام شود. سر این مستطیل نشان دهنده نقطه شروع عمل است و انتهای آن نشان‌دهنده پایان یافتن یا کامل شدن عمل است.

نمودار توالی برقراری ارتباط تلفنی



نمودار همکاری Collaboration Diagram

در این نمودار، ابتدا اشیائی که از نظر ساختاری با هم مرتبط هستند به شکل یک نمودار شیء به نمایش در می‌آیند. سپس پیام‌های ارسالی بین آنها با ذکر ترتیب ارسال بوسیله شماره بر روی هر اتصال به نمایش در می‌آید. اگر پیام‌های این نمودار را از آن حذف کنیم، دقیقا به یک نمودار شیء می‌رسیم. کلیه توضیحاتی که راجع به پیام‌ها و ترتیب و شماره‌گذاری آنها پیش از این در نمودار توالی ذکر شد در این نوع نمودار نیز کاربرد دارد. به شکل بعد به‌عنوان مثالی از این نوع نمودارها توجه کنید. همان‌گونه که در این نمودار مشخص است، تأکید نمودار همکاری بر نمایش ارتباط استاتیکی اشیاء است.



مثالی از یک نمودار همکاری

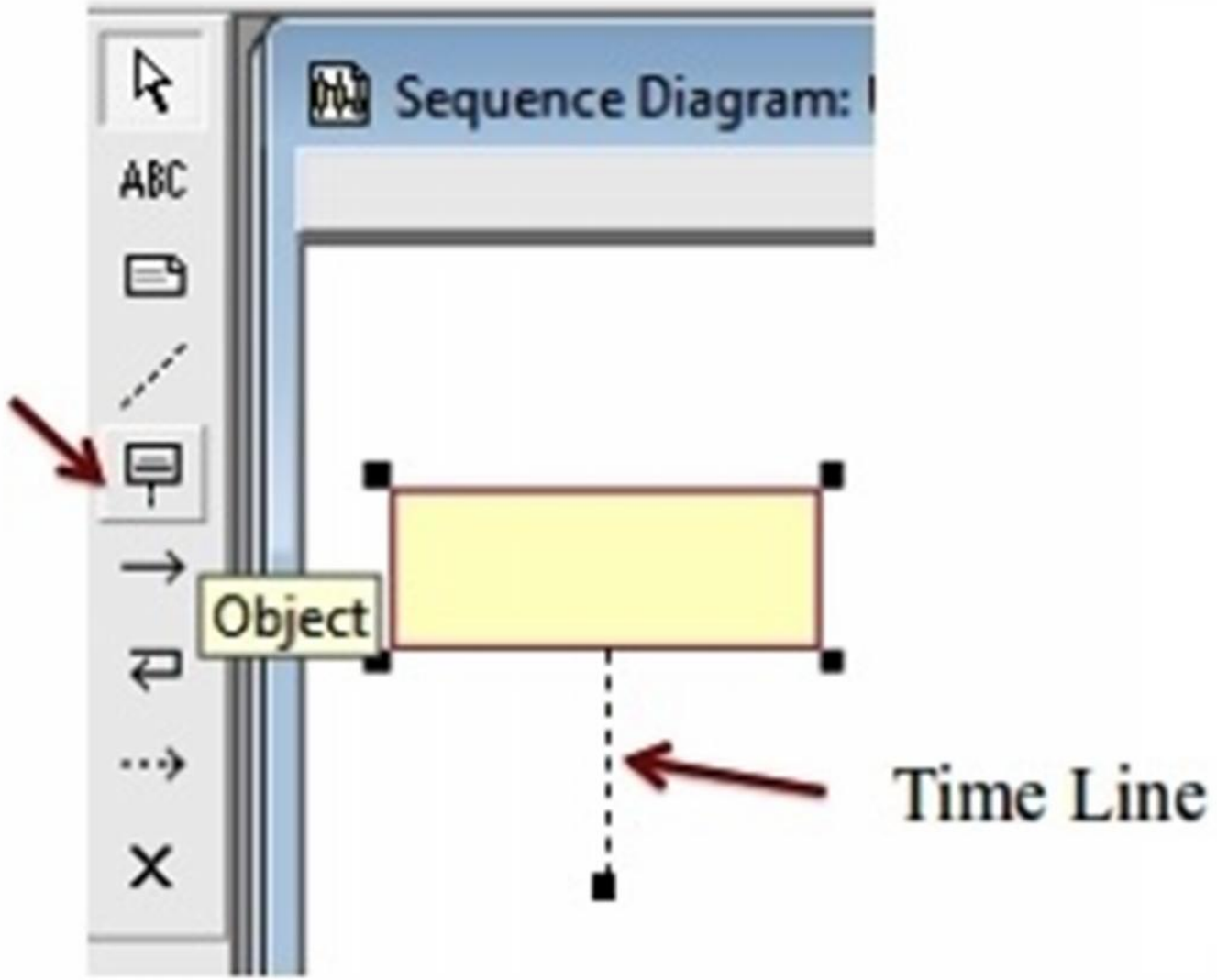
اجزای نمودار های همکاری و توالی

- Object: اشیاء همان نمونه های یک کلاس اند و در این نمودارها با یک مستطیل نمایش داده می شود که نام شئ بهمراه نام کلاس آن درون مستطیل ذکر می گردد.

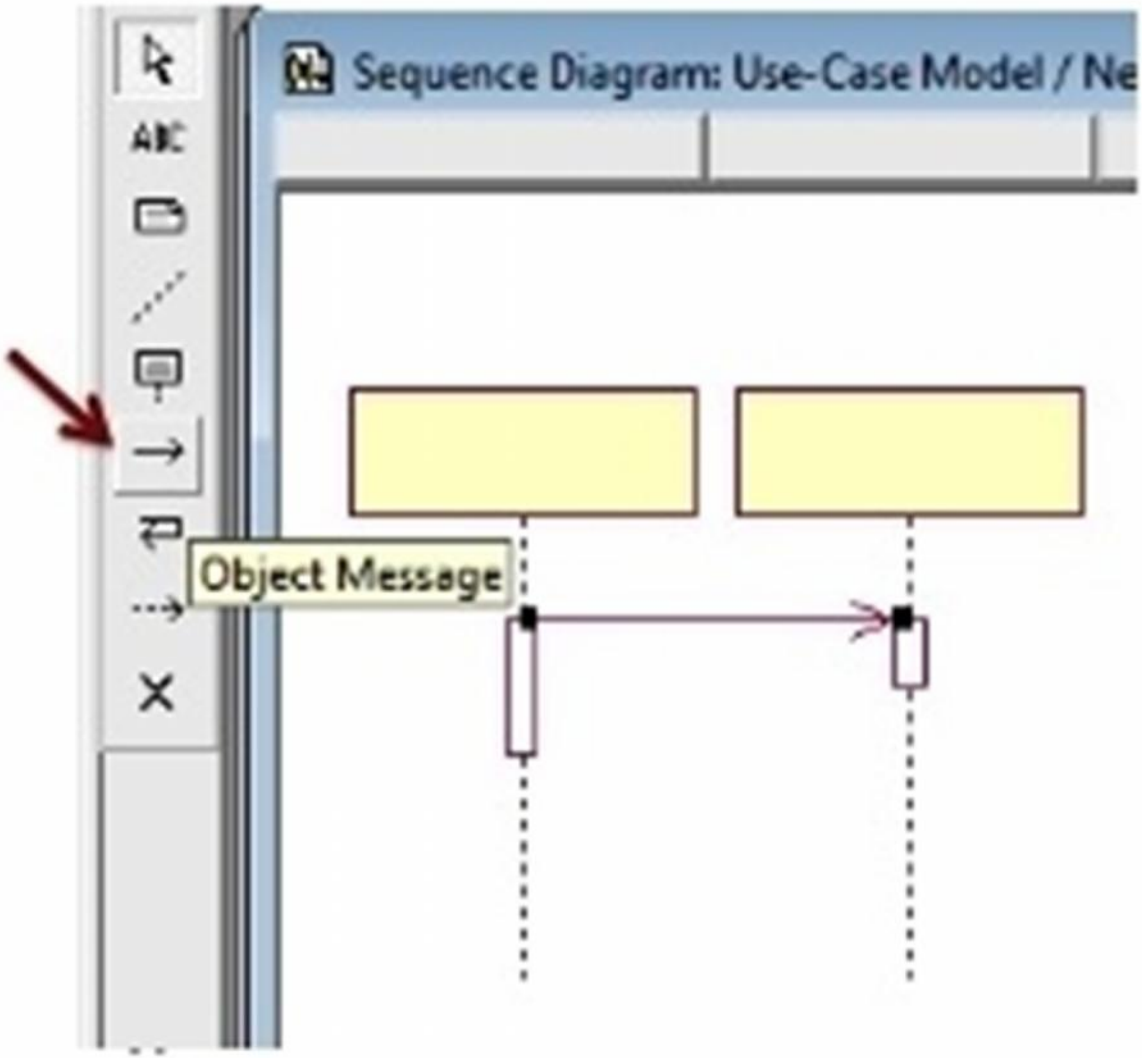
نکته: در نمودار توالی در انتهای مستطیل خط زمان (time line) وجود دارد که زمان حیات شئ را در ارتباط میان سایر اشیاء نشان می دهد.

- Message:ارتباط میان اشاء سیستم است که با فلش نشان داده می شود و از سمت فرستنده به گیرنده رسم می شوند.

نماد Object در Rational Rose

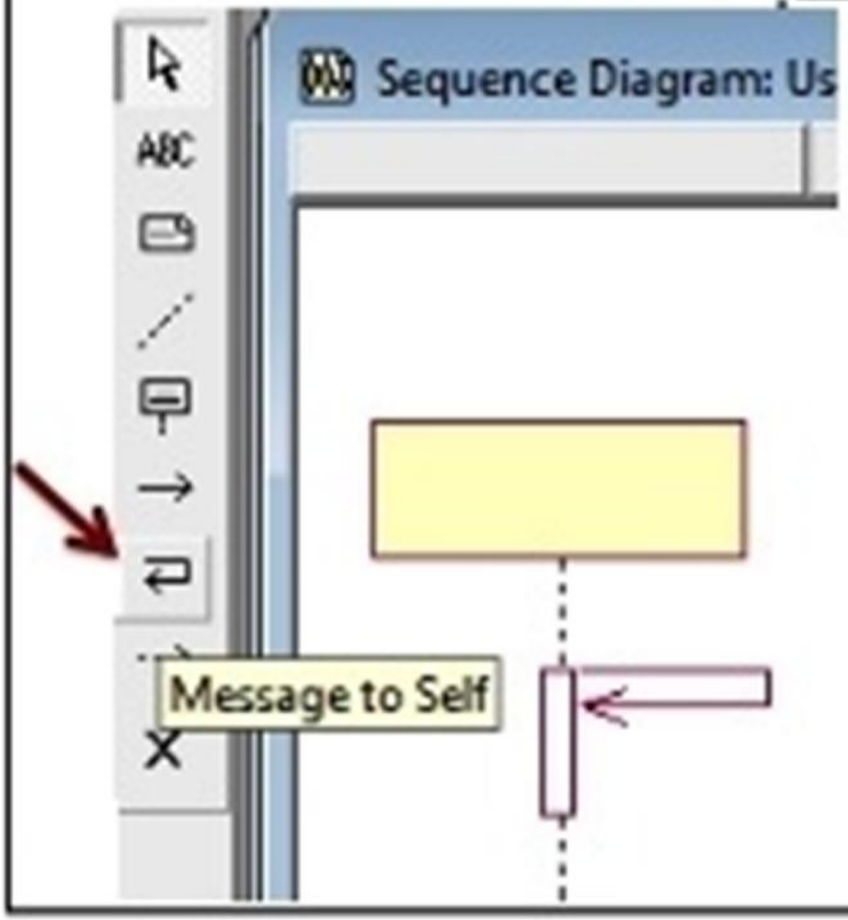


نماد Message در Rational Rose



نکات

- یک شئ می تواند به خودش پیام ارسال کند. به عبارت دیگر یک شئ می تواند متدهای خود را فراخوانی کند.

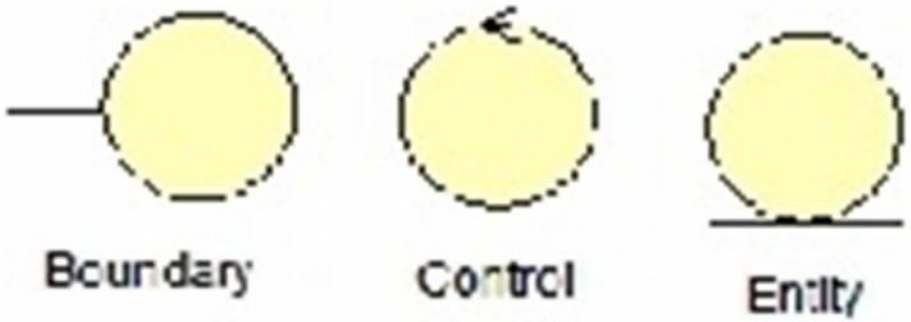


نکات (..)

- در نمودار توالی پیامهایی که در بالا ی نمودار رسم شده اند نسبت به پیامهای پایین تر زودتر ارسال شده اند.
- در Rational Rose هریک از نمودار های همکاری و توالی با فشردن دگمه ی F5 از روی دیگری ساخته می شود.

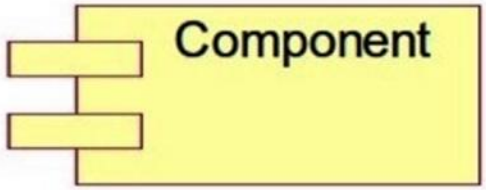
Stereotype

- **Stereotype** یا کلیشه ی یک کلاس به منظور گسترش نمودارهای UML مطرح می شود.
- برخی از کلیشه های پر کاربرد کلاس که در نمودار توالی و همکاری مورد استفاده قرار می گیرد عبارتند از:
  - **Boundary**: برقرار کننده ی تعامل میان کاربران و سیستم
  - **Control** : کنترل کننده ی اشیاء و رفتارهای سیستم
  - **Entity**: ذخیره کننده ی ساختمان داده های منطقی سیستم



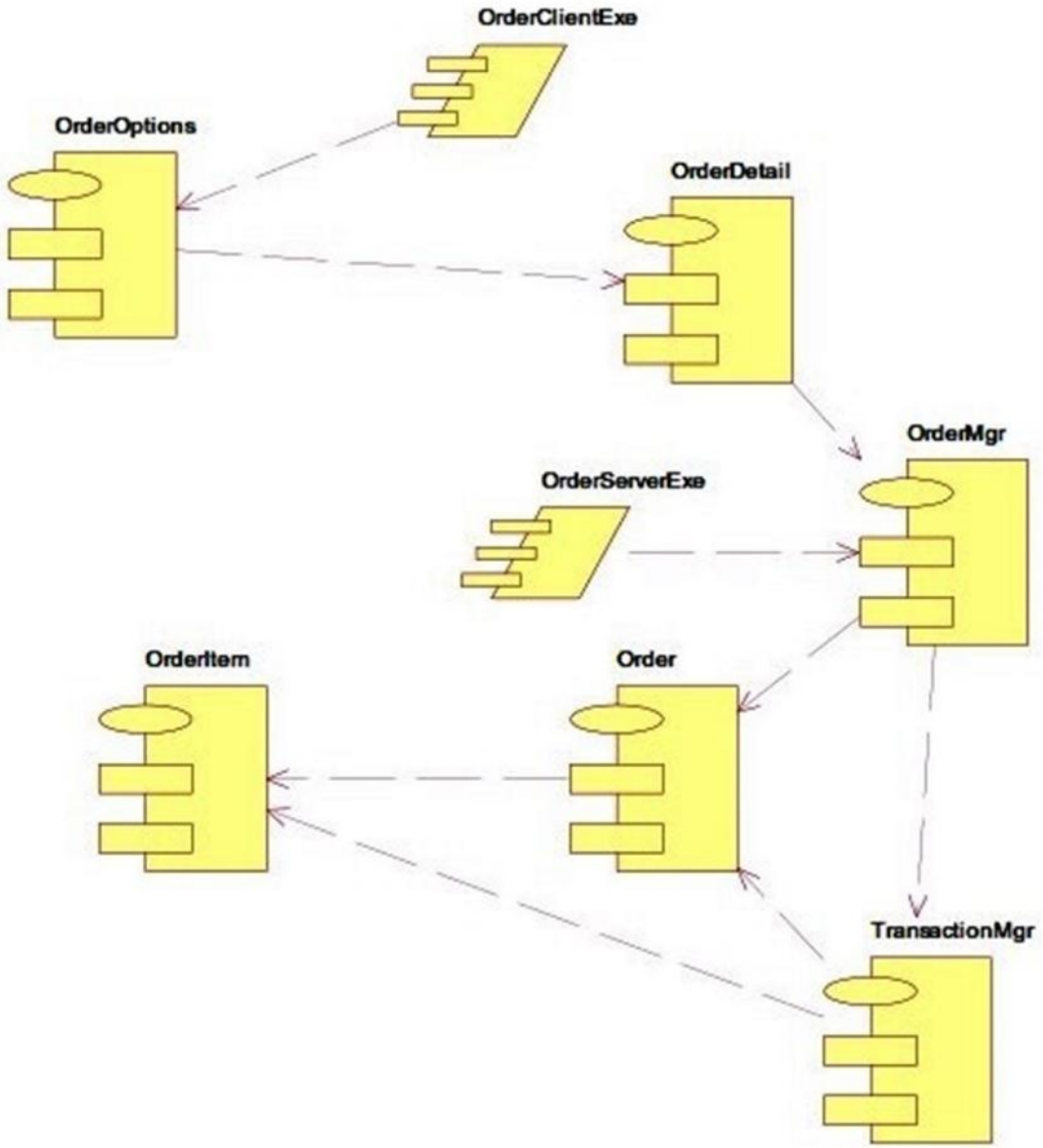
نمودار اجزا یا مولفه Component Diagram

مؤلفه (جزء) قطعه‌ای نرم‌افزاری، فیزیکی و قابل جایگزین است که مجموعه‌ای از یک یـا چند واسط را محقق می‌کند. بعنوان مثال، یک فایل اجرایی ، کتابخانه‌ای ، جداول و اسناد. هر مؤلفه دارای نامی است که او را از دیگر اجزاء متمایز می‌کند. نماد UML برای مؤلفه به صورت شکل زیر است.



نماد مؤلفه در UML

مؤلفه‌ها می‌توانند همانند هر عنصر دیگر UML، در بسته‌ها قرار گیرند و از پیچیدگی سیستم بکاهند. تنها رابطه‌ای که میان مؤلفه‌ها ممکن است وجود داشته باشد، رابطه‌ی وابستگی است. در شکل بعد مثالی از یک نمودار مؤلفه مربوط به یک سیستم ثبت و اجرای سفارش نمایش داده شده است. در این مثال OrderClientExe نمایانگر فایل اجرایی است که از کامپایل بسته‌ی کد OrderOptions حاصل می‌شود و در نتیجه به آن وابستگی دارد. بسته کد OrderOptions نیز به بسته‌کد OrderDetail وابستگی دارد. (نمایش مؤلفه‌های متناظر با فایل‌های EXE بصورت کج انتخاب ابزاری است که این نمودار با آن تهیه شده (Rational Rose) و نه استاندارد UML).



مثالی از یک نمودار مؤلفه

نمودار استقرار Deployment Diagram

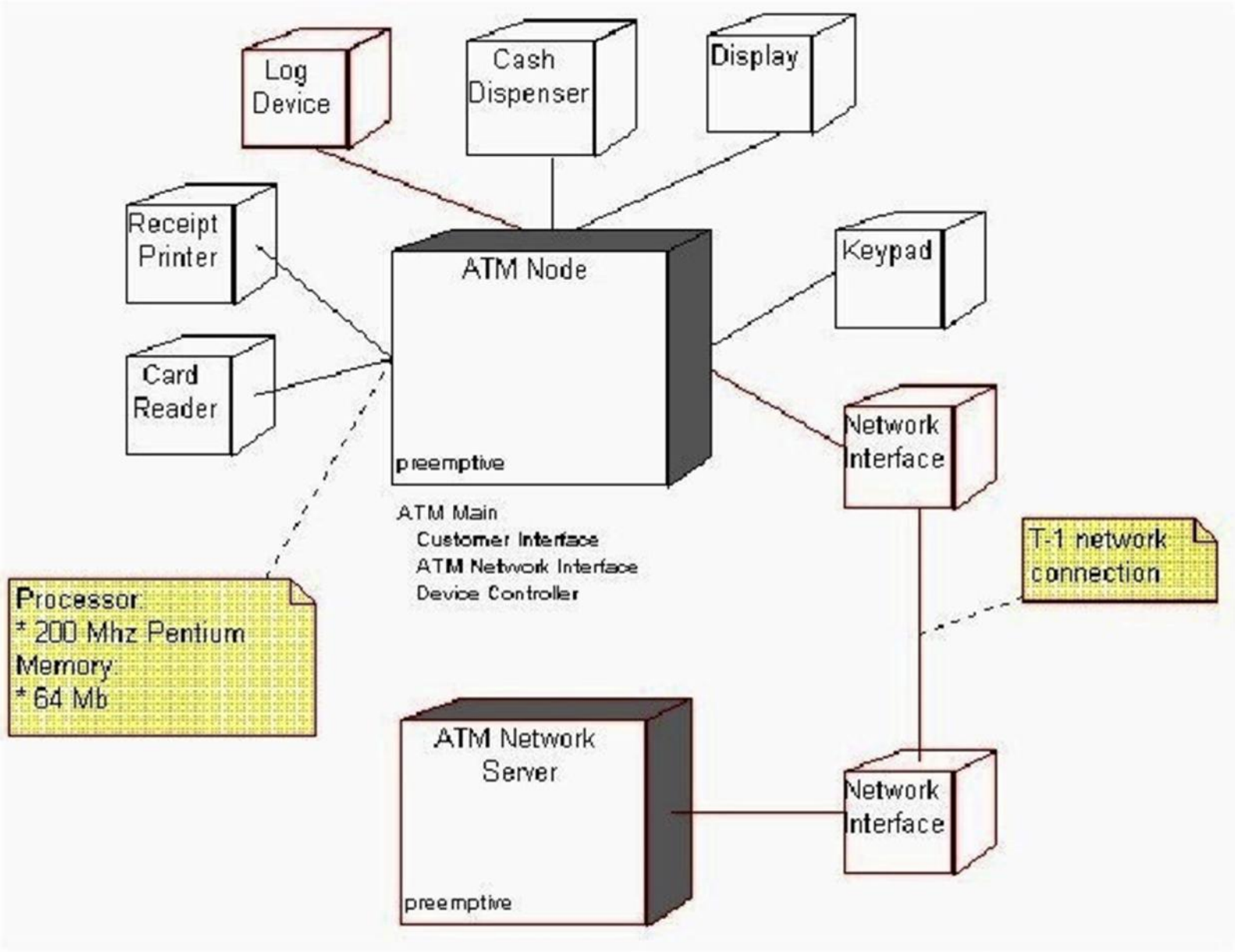
از نمودار استقرار به منظور نمایش جنبه معماری سیستم استفاده می‌شود. در نمودارهای اجزاء قسمت‌های نرم‌افزاری سیستم نهایی و اینکه هرکدام چه کلاس‌ها یا واسط‌هایی را محقق می‌سازند نمایش داده می‌شوند. در نمودار استقرار مشخص می‌شود که هر یک از اجزاء نرم‌افزاری که کل سیستم را می‌سازند، چگونه روی سخت‌افزارها توزیع می‌شوند. در این قسمت ابتدا نماد

سخت‌افزارهای سیستم را نشان می‌دهیم و سپس توضیح می‌دهیم که چگونه هر نرم‌افزاری می‌تواند بر روی یک سخت‌افزار مستقر شود. نمودار استقرار از گره‌های پردازشی، دستگاه‌ها و خطوط اتصال تشکیل شده است.

گره ، عنصری سخت‌افزاری و فیزیکی است که در زمان اجرا وجود دارد و پردازنده به حساب می‌آید. یک گره با یک مکعب مستطیل نمایش داده می‌شود و دارای نام می‌باشد. نام می‌تواند به همراه مسیر ذکر شود.

در نمودار استقرار دستگاه هایی که دارای قدرت پردازش نیستند(در سطح تجرد مدل شده) و گره‌های پردازشگر را پشتیبانی می‌کنند را نیز با مکعب نمایش می‌دهند.

ارتباطات میان گره‌های و دستگاه‌ها که نمایانگر جریان اطلاعات می‌باشند با خطوط صاف نمایش داده می‌شوند. در شکل بعد نمونه‌ای از یک نمودار استقرار مربوط به دستگاه‌های خودپرداز نمایش داده شده است.



مثالی از نمودار استقرار مربوط به دستگاه‌های خودپرداز (عابر بانک)

لازم به ذکر است که دو نمودار مؤلفه و استقرار برای مدل‌کردن چگونگی پیاده‌سازی سیستم بکار می‌روند. در واقع در UML این دو نمودار برای نمایش جنبه‌ی پیاده‌سازی سیستم بکار گرفته می‌شوند.

Rational Rose مرور کلی بر

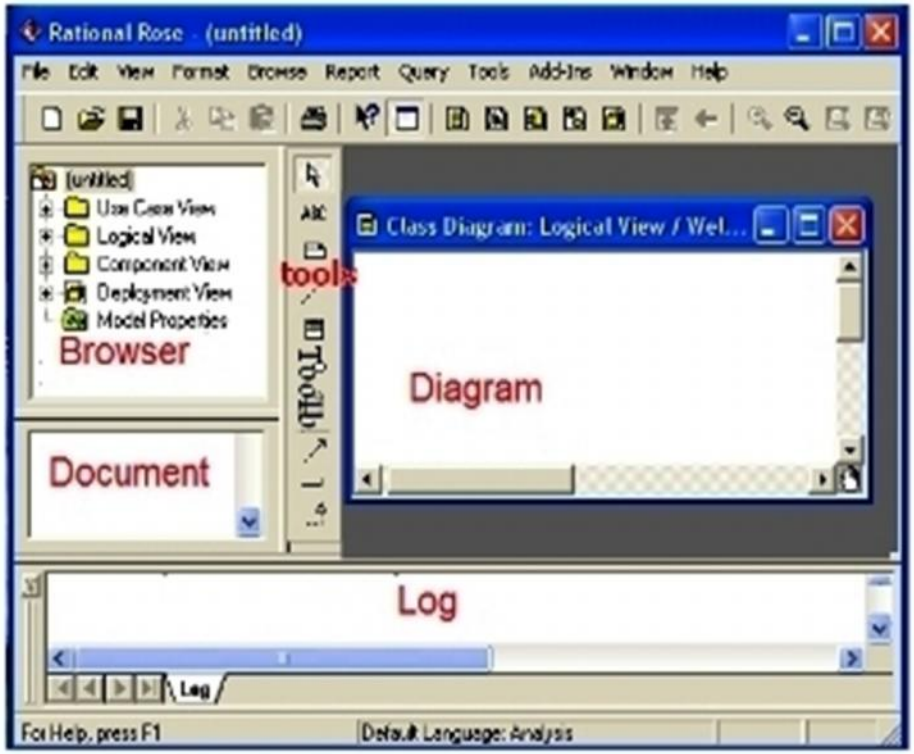
نرم‌افزار Rational Rose یک ابزار نیرومند برای تحلیل و طراحی سیستم‌های نرم‌افزاری از نوع شئ‌گرا به روش UML است. این نرم‌افزار به شما کمک می‌کند تا نمودارهای UML را به آسانی تولید و مدیریت کنید.

در این جزوه از علایم زیر استفاده شده است.

- R: یعنی راست کلیک کردن (Right click)
- C: یعنی کلیک کردن (Click)
- D: یعنی دابل کلیک کردن (Double click)
- your : یعنی مورد نظر. برای نمونه your usecase یعنی usecase مورد نظر
- mnu: یعنی منو. برای نمونه mnu file یعنی منوی فایل

محیط Rational Rose:

در شکل زیر پنجره های browser، tools و diagram را به ذهن بسپارید



افزودن ابزار مورد نظر به نوار tools:

R on tools bar > customize > available toolbar buttons > C on your tool > add > close

پنجره Browser

- در پنجره browser هر عنصری که در کنارش علامت جمع داشته باشد با کلیک بر علامت جمع باز می شود و گر نه با دابل کلیک باز می شود.

- پنجره Browser شامل چهار نما بشرح زیر می باشد:

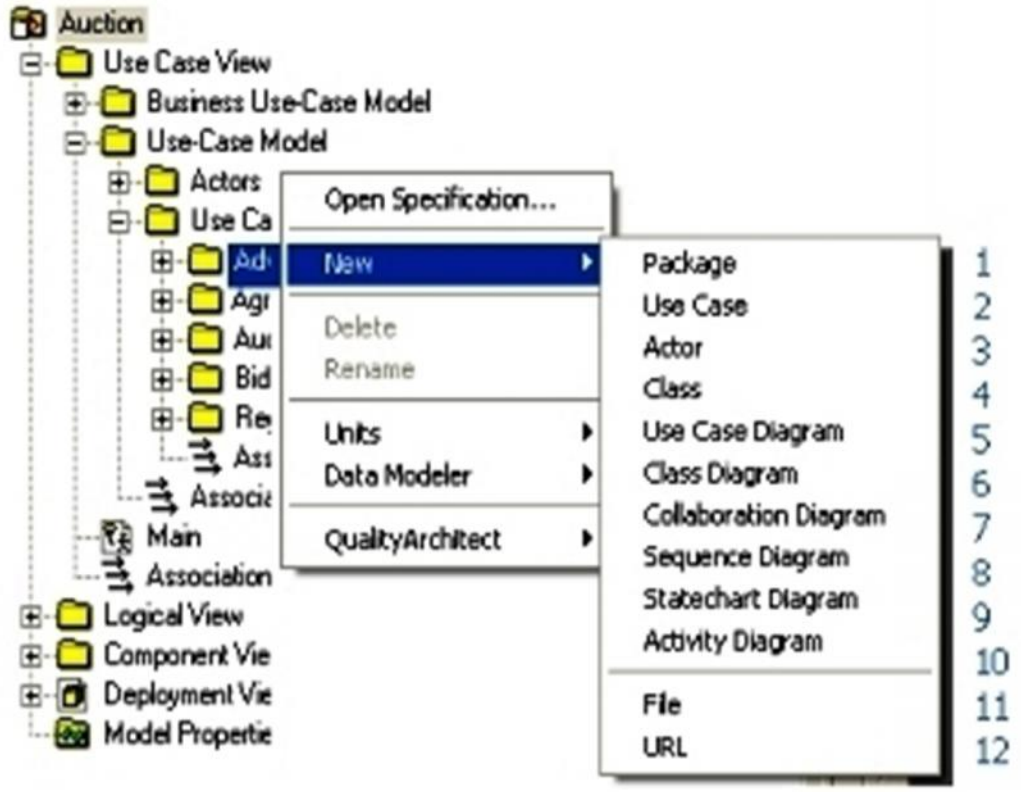
نما	عناصر UML
Usecase view	Actors , Usecases , Usecase Documents , Usecase Diagrams , Sequence Diagrams , Collaboration Diagrams , Packages
Logical view	Classes, Classes Diagrams, State Transition, Packages
Component view	Components , Component Diagrams , Packages
Deployment view	Process , processor , device , Deployment Diagrams

- برای گروه بندی عناصر UML ابزار package به کار می بریم. ابزار package در همه view ها کاربرد دارد.

- افزودن عناصر UML مورد نظر به مدل خود

Browser > R on your package > new > C on your UML item

مثال)



فایهای خارجی، توضیحات اضافی در مورد عناصر مدل هستند که معمولا در word نوشته می‌شوند

- حذف عناصر از مدل:

Browser > R on your item > delete

پنجره Diagram

- باز کردن نمودار:

browser > D on your Diagram

- افزودن عناصر به نمودار

برای این منظور ابتدا نمودار مورد نظر را باز می کنیم سپس عناصر مورد نظر را با ماوس از پنجره browser به نمودار می اندازیم.

- حذف عناصر از روی نمودار

in your diagram > R on your item > edit > delete

- نمایش مرتب عناصر درون نمودار

open your diagram > mnu format > layout diagram, autosize all

- افزودن توضیحات به عنصر مورد نظر در نمودار

tools > C note > diagram > C your place > typing your text on your note > tools >
> C anchor note to item > DD your note to your item.

به عبارتی

۱- از نوار tools ابزار note را انتخاب کنید.

۲- در جایی از diagram برای افزودن یادداشت کلیک کنید.

۳- در یادداشت ایجاد شده، متن خویش را تایپ کنید.

۴- از نوار tools ابزار anchor note to item را انتخاب کنید.

۵- ماوس را از یادداشت به اجزای مورد نظر بکشید.

- برچسب زدن به عنصر مورد نظر در نمودارها

tools > C textbox > diagram > C your place > typing your text on your textbox.

به عبارتی

۱- از نوار tools ابزار textbox را انتخاب کنید.

۲- در جایی از diagram برای افزودن textbox کلیک کنید.

۳- در کادر ایجاد شده، متن مورد نظر را تایپ کنید.

مثال)

Usecase View

نرم‌افزار Rational Rose با actor ها مانند کلاس برخورد می‌کند. بنابراین پنجره تنظیمات (specification) برای actor و class یکسان است.

تنظیمات یک actor

Browser > R on your actor > open specification >

> general > name = ? نامگذاری actor:

> general > documentation = ? توضیح در مورد actor:

> general > stereotype = actor تعیین کلاس از نوع actor:

> Relation مشاهده رابطه‌هایی که actor در آنها همکاری دارد:
در این برگه می‌توان رابطه ها را حذف و یا اضافه کرد.

> Detail > abstract = true تعیین actor از نوع abstract:

> Detail > Multiplicity =? تعیین تعداد نمونه های هر actor:

Multiplicity	معنی
*	بیشتر
0	صفر
1	یک
0..*	صفر یا بیشتر
1..*	یک یا بیشتر
0..1	صفر یا یک
1..1	تنها یکی
num1	تنها به اندازه num1
<num1>.. <num2>	بین num1 و num2
<num1>, <num2>.. <num3>	تنها به اندازه num1 یا بین num2 و num3
<num1>.. <num2>, <num3>.. <num4>	بین num1 وnum2 یا بین num3 وnum4

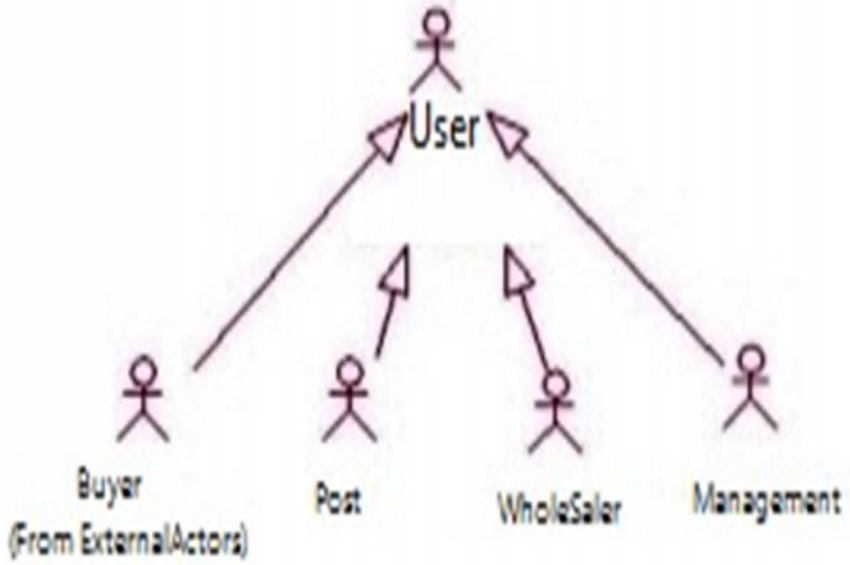
نشان دادن ارتباط ارث بری میان Actorها:

۱- یک usecase diagram ایجاد کرده و actor ها را از بخش browser بر روی آن Drag & Drop کنید.

۲- در نوار tools بر ابزار generalization کلیک کنید.

۳- ماوس را از actor های اصلی به سمت abstract actor بکشید.

مثال)



مشاهده نمودارهای مربوط به یک actor :

in your usecase diagram > C your actor > mnu Report > show instance

تنظیم یک usecase:

Browser > R on your usecase > open specification >

نامگذاری usecase:
> general > name = ?

توضیح usecase:
> general > documentation = ?

این توضیح در پنجره documentation دیده خواهد شد و در آنجا نیز قابل تغییر است.

طبقه‌بندی usecase:
> general > stereotype = ?

برای نمونه usecase مربوط به انبارداری.

تخصیص اولویت:
> general > rank = ?

با توجه به اولویت usecaseها، پروژه را ادامه می‌دهیم.

تعیین usecase از نوع abstract:
> general > abstract = true

مشاهده لیست رابطه‌هایی که usecase در آنها مشارکت دارد.
> relation

مشاهده نمودارهای یک usecase:

> diagram > D on your diagram

این نمودارها، در viewهای مختلف UML تولید خواهند شد. و تعدادی از آنها عبارتند از:

Sequence diagram, class diagram, usecase diagram, state transaction diagram

ایجاد usecase diagram

۱- یک usecase diagram خالی در package مورد نظر ایجاد می کنیم

۲- usecase ها و actor های مورد نظر را با ماوس از پنجره browser به usecase diagram می اندازیم

۳- روابط بین actor ها و usecase ها را طبق شرح زیر برقرار می کنیم.

۴- روابط include و extended را بین usecase ها طبق شرح زیر برقرار می کنیم.

روابط بین actor ها و usecase ها

۱- در نوار tools ابزار unidirectional association کلیک کنید.

۲- در usecase diagram ماوس را از actor مورد نظر به سوی usecase مورد نظر بکشید و رها کنید.

ایجاد ارتباط extend بین usecase ها

۱- در نوار tools ابزار dependency or instantiates کلیک کنید.

۲- در usecase diagram ماوس را از abstract usecase به سوی usecase اصلی بکشید و رها کنید.

۳- نوع رابطه را طبق دستور زیر تعیین کنید

Diagram > R on your arrow > open specification > general > stereotype = extend > ok

۴- برچسب << extend >> را طبق دستور زیر مخفی کنید

Diagram > R on your arrow > C stereotype label

ایجاد ارتباط include بین usecase ها:

۱- در نوار tools ابزار dependency or instantiates کلیک کنید.

۲- در usecase diagram ماوس را از usecase اصلی به سوی abstract usecase بکشید و رها کنید

۳- نوع رابطه را طبق دستور زیر تعیین کنید

Diagram > R on your arrow > open specification > general > stereotype = include > ok

۴- برچسب << include >> را طبق دستور زیر مخفی کنید.

Diagram > R on your arrow > C stereotype label

ایجاد sequence diagram برای یک usecase

۱-

Browser > R on your usecase > new > sequence diagram >

۲- در پنجره browser بر sequence diagram شده کلیک می کنیم تا باز شود

۳- actor ای که اجرا کننده usecase است از پنجره browser به sequence diagram درگ می کنیم.

۴- object های مورد نیاز برای اجرای usecase را در sequence diagram طبق شرح زیر ایجاد می کنیم.

۵- بین object ها پیغام های رد و بدل شده را طبق شرح زیر تعریف می کنیم.

ایجاد یک object به sequence diagram

۱- در نوار tools بر ابزار object کلیک کرده سپس در sequence diagram بر محل مورد نظر کلیک کنید.

۲- تنظیم خصوصیات object:

Diagram > R on your object > open specification >

نامگذاری object: > name = ?

نگاشت object به یک کلاس: > class = ?

بعد از نگاشت obj1 به cls1، بر جسب object در sequence diagram بصورت obj1 : cls1 نمایش داده می شود.

اگر کلاس مورد نظر در لیست class نباشد. باید ابتدا آن را در logical view ایجاد کرده سپس از این لیست انتخاب کنیم.

نحوه ایجاد class در فصل logical view توضیح داده شده است.

حذف نگاشت: > class = unspecification

افزودن توضیحات به یک object: > documentation = ?

این توضیحات و در کد ظاهر خواهد شد.

تنظیم پایداری object: > persistence = ?

persistent: شئ در حافظه جانبی مانند database نگهداری می‌شود. بنابراین اگر برنامه از حافظه اصلی خارج شود شئ از بین نخواهد رفت.

static: شئ تا زمانی که برنامه در حافظه اصلی است وجود خواهد داشت.

transient: شئ موقتی است و تنها برای مدت کوتاهی در حافظه اصلی باقی می‌ماند.

ایجاد یک نمونه چندگانه از یک کلاس: > multiple instance = true

اطمینان از اینکه همه object ها به class نگاشته شده اند:

تولید کد زمانی امکان پذیر است که همه object ها به class های مورد نظر نگاشته شده‌اند.

Mnu Report > show Unresolved object

ایجاد پیغام در sequence diagram بین object ها:

۱- اطمینان بیابید object های فرستنده و گیرنده به کلاس های مورد نظر نسبت داده شده اند.

۲- از نوار tools ابزار object message یا دکمه message to self را انتخاب کنید

۳- در sequence diagram ماوس را از خط عمر object فرستنده پیغام به object گیرنده پیغام بکشید.

۴- بر پیکان افقی ایجاد شده بین دو object راست کلیک کرده و گزینه < new operation > را انتخاب کنید.

۵- سپس در پنجره باز شده خصوصیات پیغام را طبق توضیحات زیر تنظیم کنید.

> Message specification

نامگذاری پیغام: > general > name = ?

افزودن توضیحات: > general > documentation =?

تنظیم تکرار message : > detail > frequency =?

Periodic: message در فواصل زمانی معین فرستاده می شود.

تنظیم همزمان‌سازی پیغام: > detail > synchronization =?

simple: ارسال پیغام یک جهته است.

synchronous: فرستنده منتظر گیرنده می‌ماند.

balking: اگر گیرنده آماده دریافت message نباشد، فرستنده متوقف می‌شود

timeout: فرستنده پیغام را می‌فرستد و زمان کوتاهی صبر می‌کند اگر در آن زمان گیرنده آماده دریافت massage نباشد فرستنده ارسال پیغام را متوقف می‌کند.

asynchronous: فرستنده منتظر گیرنده نمی‌ماند و پردازش خود را ادامه می‌دهد.

procedure calls

return

فعال کردن شماره‌گذاری پیغامها به طور خودکار:

Mnu Tools > options > diagram > display > sequence numbering = true

فعال کردن مشاهده مدت زمان اجرای پیغامها بر روی خط عمر object ها:

Mnu Tools > options > diagram > display > Focus of control = true

اضافه کردن عناصر توضیحاتی به sequence diagram

علاوه بر افزودن توضیحات در Note و textbox می‌توان توضیحات را در ابزار دیگری به نام script نمایش داد. ابزار script معمولا برای افزودن توضیحات به پیغامها روی sequence diagram به کار می‌رود. به عنوان نمونه برای نمایش شرایط ارسال یک پیغام این ابزار را به کار می‌بریم. این ابزار در سمت چپ diagram و روبروی message مورد نظر قرار می‌گیرد.

برای script مراحل زیر را انجام می دهیم:

۱- توضیح مورد نظر را textbox بنویسید

۲- textbox و پیکان مورد نظر را با هم به وسیله کلید ctrl انتخاب کنید

۳- از منوی edit گزینه Attach Script را کلیک کنید.

مشاهده شرکت کنندگان در usecase:

لیستی از تمام کلاسها و عملیات که در usecase مورد نظر شرکت دارند، را نشان می‌دهد.

In your usecase diagram > C your usecase > mnu Report > show participants

Logical View

ایجاد مدل معماری

در logical view یک نمودار class به نام welcome وجود دارد در این نمودار مدل معماری را طبق شرح زیر ایجاد می‌کنیم.

۱- package های مورد نظر را در logical view ایجاد می کنیم.

۲- نمودار welcome را باز می کنیم و با ماوس package ها به درون این نمودار می اندازیم.

۳- در نمودار welcome ارتباط بین package ها را برقرار می کنیم. این ارتباط از نوع هبستگی یا وابستگی است. که در بخش ارتباط بین کلاس ها توضیح داده شده است.

۴- در پنجره browser در درون هر package کلاس های مورد نظر را ایجاد می کنیم.

۵- در پنجره browser در درون هر package یک class diagram ایجاد می کنیم.

۶- نمودار class diagram هر package را باز می کنیم و با ماوس کلاس های package را به class diagram می اندازیم.

۷- در هر class diagram ارتباط بین کلاس ها را طبق توضیحاتی که در بخش ارتباط بین کلاس ها آمده است ایجاد می کنیم.

افزودن صفت به یک کلاس:

۱-

Logical view > R on your class > open specification > attribute > R on box > insert

۲- تایپ یک نام برای صفت

۳- بر نام صفت دابل کلیک می کنیم تا پنجره class attribute specification باز شود. در این پنجره نوع و مقدار پیشفرض و غیره را مشخص می کنیم.

تنظیم خصوصیات هر صفت:

Logical view > R on your class > open specification > attribute > D on your attribute >

- visibility: public , private , …
- محدودیتهای صفت: by value , by reference ,..
- به اشتراک گذاشتن یک صفت: static page 310
- صفت مشتق شده: derived

افزودن آرگومان به عملیات یک کلاس:

۱-

Logical view > R on your class > open specification > operation> D on your operation > Detail > R On box augment > insert >

۲- تایپ یک نام برای متغیر

۳- تعیین نوع متغیر

Logical view > R on your class > open specification > operation> D on your operation > detail > D on your augment > name = ?, type = ?, Default = ? > ok

۴- نمایش آرگومان در نمودار

mnu tools > options > diagram > message signatures > name and type = true

تنظیمات یک کلاس

logical view > R on your class > open specification >

> detail > abstract = true

تولید abstract class :

> detail > multiplicity = ?

تعیین تعداد نمونه های یک کلاس:

> general > stereotype =?

گروه بندی کلاسها:

برای نمونه یک stereotype به نام form می‌سازیم و همه پنجره‌ها را به این stereotype نسبت می‌دهیم. مقادیر stereotype به صورت زیر است:

Stereotype = Implementer | manager | access | helper | Boundary | Entity | Control | ...

مشاهده صفات و متدهای یک کلاس در class diagram

In class diagram > R on your class > options > suppress operation = false, suppress attribute = false, show operations = true

ایجاد ارتباط میان کلاس ها در class diagram

برای ایجاد انواع نمودارها به ابزارهای گوناگون نیاز داریم. برای افزودن ابزار مورد نیاز به نوار tools مراحل زیر را انجام می دهیم:

R on tools bar > customize > available toolbar buttons انتخاب ابزار مورد نظر از لیست > add > close

رابطه همبستگی

۱- از نوار tools ابزار unidirectional association را انتخاب کنید.

۲- در class diagram ماوس (خط association) را از کلاس مبداء به کلاس مقصد بکشید.

۳- تعیین دوطرفه بودن رابطه: R on relation > navigate

رابطه وابستگی

۱- از نوار tools ابزار dependency or instantiates را انتخاب کنید.

۲- در class diagram ماوس (خط dependency) را از کلاس مبداء به کلاس مقصد بکشید.

رابطه تجمع

۱- از نوار tools دکمه unidirectional aggregation را انتخاب کنید.

۲- در class diagram ماوس (خط aggregation) را از کلاس جزء به کلاس کل بکشید.

رابطه ترکیب

۱- از نوار tools دکمه aggregation را انتخاب کنید.

۲- در class diagram ماوس (خط aggregation) را از کلاس جزء به کلاس کل بکشید.

رابطه ارث بری

۱- از نوار tools ابزار generalization را انتخاب کنید.

۲- در class diagram ماوس (خط generalization) را از sub class به super Class بکشید.

تنظیم مشخصات یک رابطه

In your class diagram > R on your relation > open specification >
> general > name = ? نامگذاری:

نام یک رابطه بهتر است بصورت فعل باشد. این نام دلیل وجود رابطه و جهت رابطه را مشخص می‌کند. به عنوان نمونه دانشجو می‌تواند درس را اخذ کند ولی درس نمی‌تواند دانشجو را اخذ کند

توضیح: > general > documentation = typing description

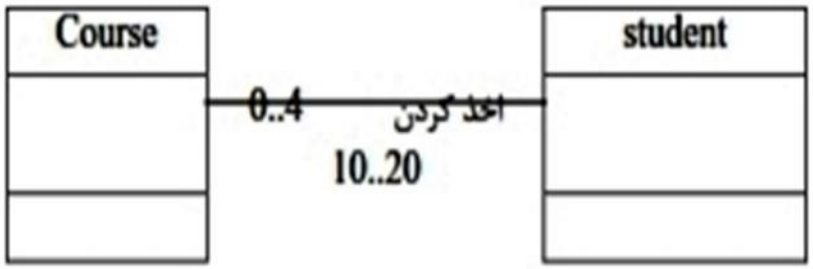
طبقه‌بندی: > general > stereotype = typing type

بین << >> قرار می‌گیرند.

تعیین روابط چند به چند بین کلاسها: > role detail > multiplicity =?

Multiplicity قانونهای کاری را مشخص می‌کند:

مثال) هر دانشجو می‌تواند صفرتا چهار درس را بگیرد، بنابراین فردی که یک ترم مرخصی می‌گیرد، هنوز به عنوان یک دانشجو در نظر گرفته می‌شود.



تغییر نحوه نمایش کلاس در class diagram:

In your class diagram > R on your class > options > Stereotype display=?

یا

Mnu tools > options > diagram > compartment > show stereotype
Mnu tools > options > diagram > stereotype display >

Component View

تنظیم مشخصات component:

R on your component > open specification >

- تعیین زبان برنامه‌نویسی: > language = ?
- تعیین نوع component: > stereotype = ?
- نسبت دادن کلاسها به component مورد نظر: > realizes > R on your class > assign

ایجاد نمودار component

۱- در قسمت component view بر نمودار main دابل کلیک می کنیم تا باز شود
۲- component های ایجاد شده را با ماوس از پنجره browser به نمودار main می اندازیم
۳- ارتباط بین component ها را (از نوع هبستگی یا وابستگی) ایجاد می کنیم.

Deployment View

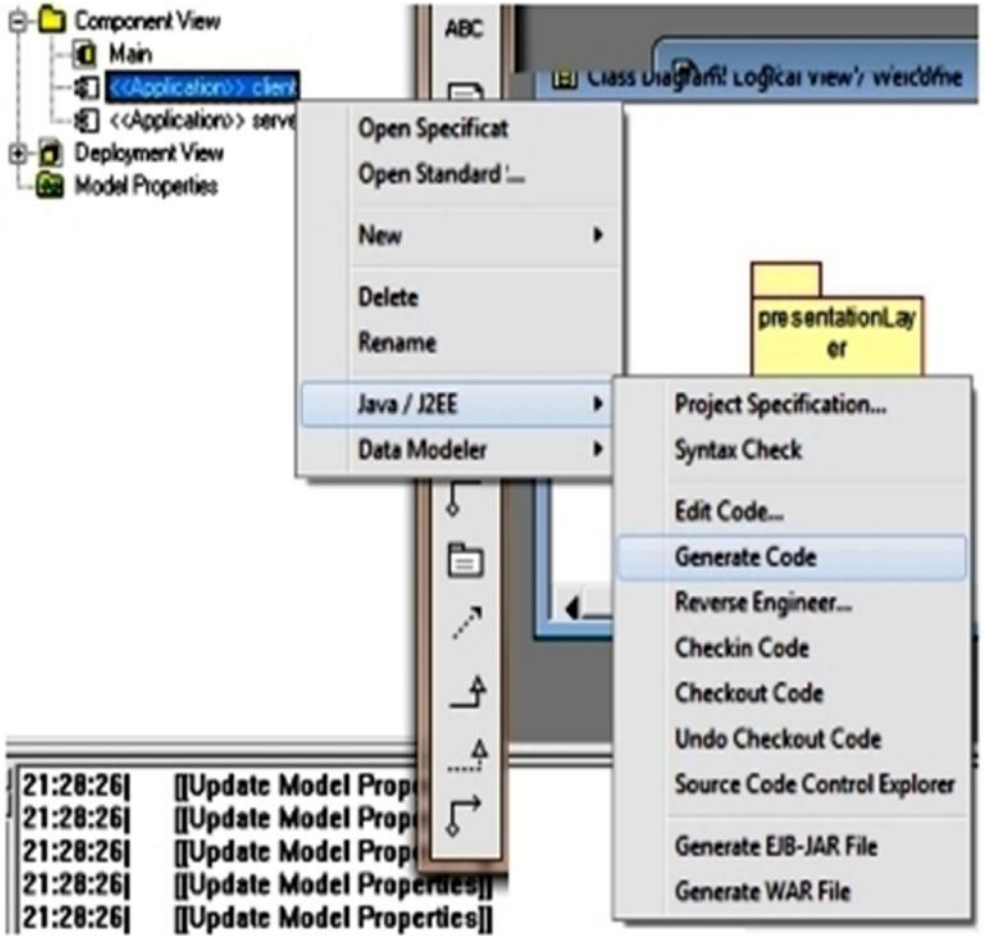
ایجاد نمودار deployment

۱- بر روی خود deployment view کلیک می کنیم
۲- با ماوس component ها را از پنجره browser به نمودار باز شده می اندازیم

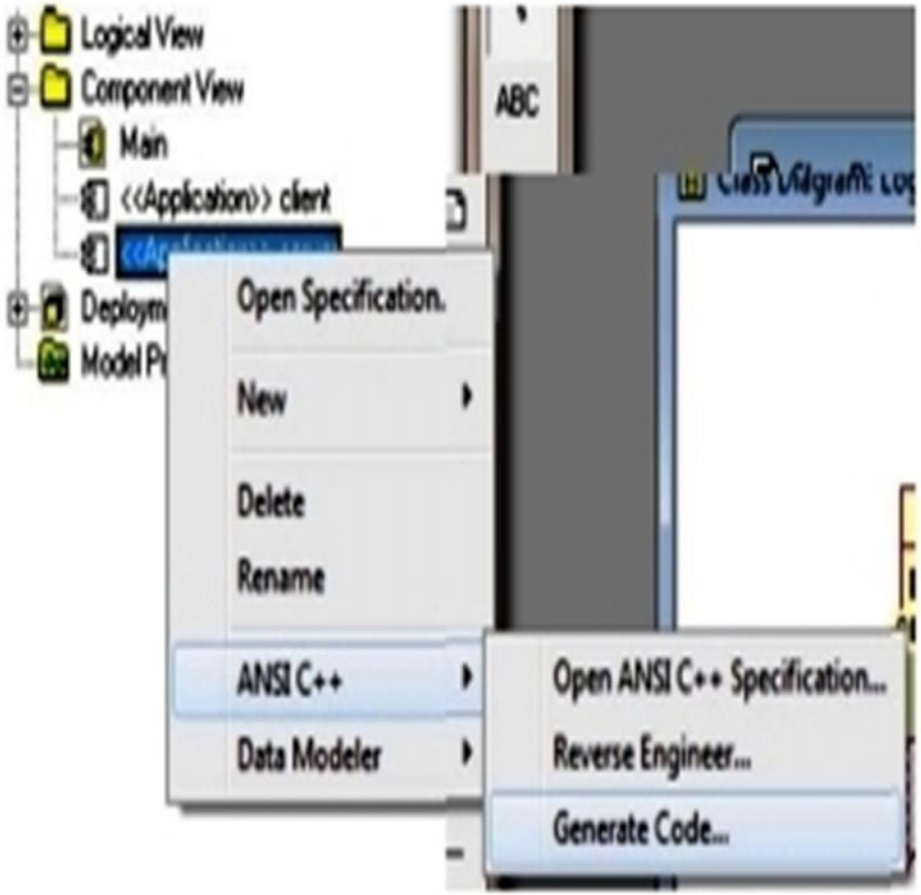
تولید کد

برای تولید کد کافیست در پنجره browser بر component مورد نظر راست کلیک کنیم و زبانی که قبلا برای آن component انتخاب کرده اید را کلیک کنید. همچنین درباره تولید کد می توانید از اینترنت کمک بگیرید.

مثال ۱)



مثال ۲)


Logical View
Component View
Main
<<Application>> client
Open Specification...
New
Delete
Rename
ANSI C++
Data Modeler
Open ANSI C++ Specification...
Reverse Engineer...
Generate Code...


موفق باشید