

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِیْمِ



موسسه آموزش عالی کوشیار رشت

جزوه درس آزمایشگاه مهندسی نرم افزار

( نرم افزار رشنال رز )

مدرس: پوررجبی

## تعریف کلمات کلیدی و عناصر اصلی UML

مدل : علاوه بر تعریفی که در قسمت قبلی ارائه شد ، مدل از تعدادی "شی" تشکیل شده است.

شیء : هر یک از اجزاء نرم افزار یک "شی" نام دارد.

کلاس: مشخصه ها و تعاریف عمومی یک "شی" در کلاس مربوط به آن تعریف میشود . هر شیء نمونه ساخته شده از یک کلاس است . کلاس ، وظایف ، قابلیتها ، خصوصیت‌های آن موجودیت نرم افزاری را تعریف میکند و طراح با ساختن یک نمونه از کلاس ( شی ) و ارائه مقادیر مرتبط با محل خاص استفاده آن یا قرار دادن آن در مسیر وقایع لازم ، محیط مورد نظر خود را طراحی میکند . نمونه های مختلفی از یک کلاس در قالب اشیاء مختلف ساخته میشوند ، ضمن اینکه میتوان یک شیء را بطور همزمان از دو یا تعداد بیشتری کلاس ساخت . یعنی شی مذکور تمام مشخصه های کلاسهای فوق خود را به "ارث" خواهد برد.

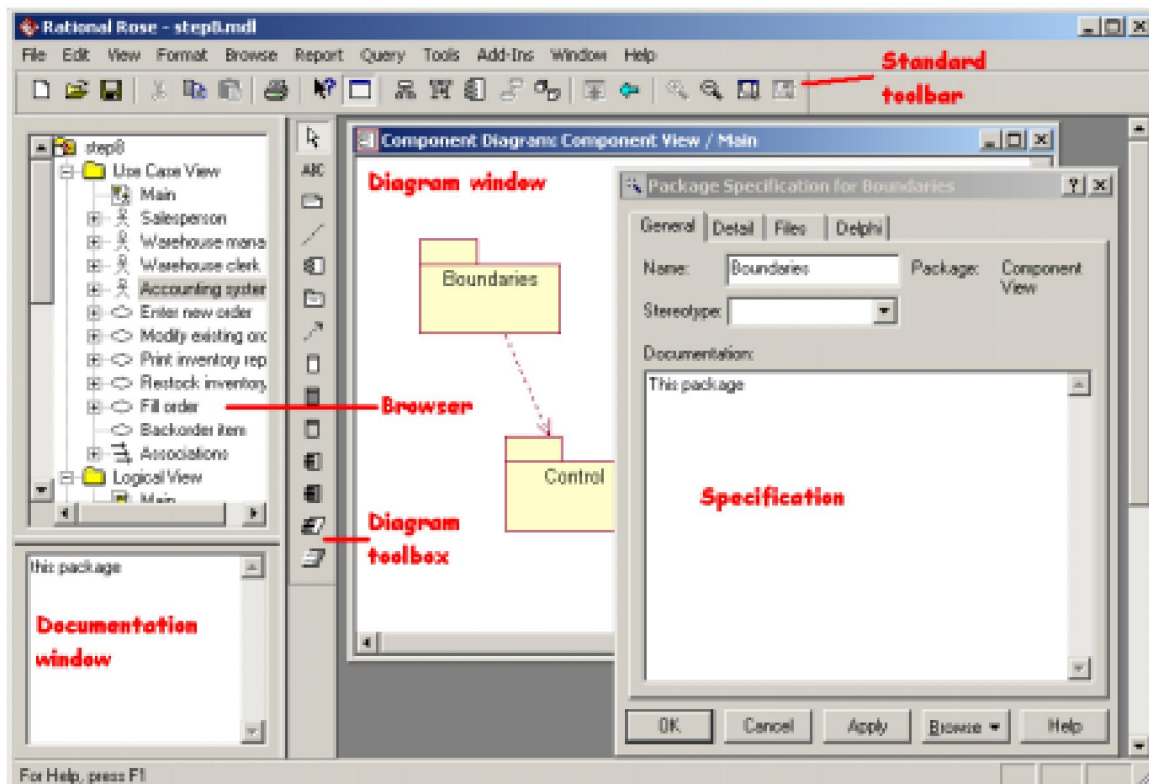
هر شی دارای تعدادی مشخصه یا Attribute است . مشخصه های یک شیء حاوی مقادیر مختلف اجزاء آن شیء هستند . به عنوان یک مثال ساده وقتی مختصات یک کلید روی فرم ۲۰ و ۴۰ است ، یعنی مشخصه های طول و ارتفاع آن ۲۰ و ۴۰ هستند.

اشیاء "میتوانند" کارهایی انجام دهند که به آنها اصطلاحاً Behavior "میگویند" . به عنوان مثال وقتی ماژول رمزنگاری نرم افزار شما میتواند درهم سازی با متد MD5 هم انجام دهد ، این توانائی یکی از Behavior های این شیء محسوب میگردد.

# معرفی محیط رشنال رز

## عناصر اصلی رشنال رز عبارتند از:

- ۱- **Standard toolbar**: که برای تمام دیاگرام‌ها مشترک است و در قسمت بالای پنجره واقع است.
- ۲- **Diagram toolbar**: که وابسته به پنجره‌ی دیاگرام فعال است و در سمت چپ پنجره‌ی دیاگرام واقع است.
- ۳- **Browser**: به شما اجازه می‌دهد تا بصورت یک ساختار درختی دیاگرام‌های موجود و عناصر مدل‌هایتان را مشاهده کنید.
- ۴- **Diagram window**: ساخت و ویرایش دیاگرام‌ها در این قسمت صورت می‌پذیرد.
- ۵- **Documentation window**: به شما اجازه می‌دهد تا به مدل‌هایتان مستندات لازم را نیز اضافه نمایید. می‌توانید مستنداتتان را در این قسمت یا در قسمت specification ویرایش نمایید.
- ۶- **Specification**: محیط ویرایشی برای اضافه کردن مستندات به مدل.



## معرفی نماهای رشنال رز

برای یک پروژه‌ی در حال ساخت نماهای مختلفی وجود دارد. رشنال رز نماهای زیر را برای یک پروژه فراهم می‌آورد که هر کدام یکی از جنبه‌های مختلف مدل را نمایش می‌دهند:

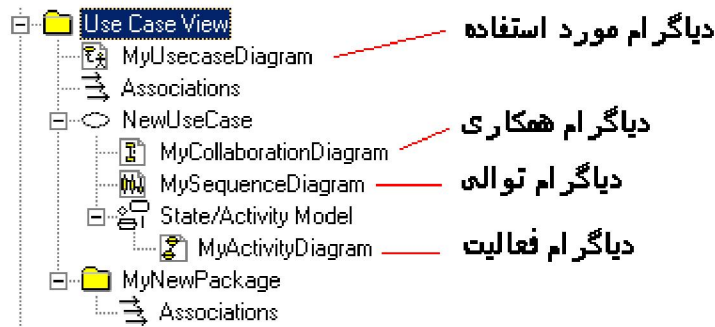


- Usecase view
- Logical view
- Component view
- Deployment view

### نمای مورد استفاده‌ی سیستم ( usecase view ) :

این نما جهت فهم و استفاده از سیستم پیش‌بینی شده است. در این نما که تشریح رفتار سیستم از دیدگاه کاربر است، فعل و انفعالات متقابل بازیگرها (actors) و موردهای استفاده نمایش داده می‌شود. در این نما چهار دیاگرام زیر وجود دارند :

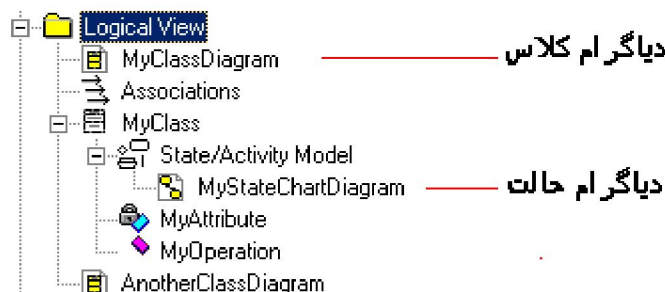
- دیاگرامهای موردهای استفاده (usecase diagrams)
- دیاگرامهای توالی (sequence diagrams)
- دیاگرامهای همکاری (collaboration diagrams)
- دیاگرامهای فعالیت (activity diagrams)



### نمای منطقی سیستم ( logical view ) :

این نما شامل نیازمندی‌های عملیاتی سیستم می‌باشد که به کلاسها و ارتباط بین آنها می‌پردازد. این نما شامل دو دیاگرام زیر می‌باشد:

- دیاگرامهای کلاسها (class diagrams)
- دیاگرامهای حالت (statechart diagrams)



### نمای اجزای سیستم ( component view ) :

این نما به سازمان سیستم می‌پردازد و اطلاعاتی در باره نرم افزار، اجزا قابل اجرا و کتابخانه‌های سیستم دارد. تنها دیاگرام موجود در این قسمت دیاگرام اجزا (component diagram) می‌باشد.



### نمای پیاده‌سازی سیستم ( deployment view ) :

این قسمت شامل نگاهی از فرایندهای موجود با سخت افزار سیستم می‌باشد. این قسمت بخصوص در جاهایی حایز اهمیت است که شما دارای برنامه‌ها و سرورهایی در قسمت‌های مختلف مکانی هستید و می‌خواهید به بهترین شیوه ساختار توزیعی محیط را نشان دهید.

این نما فقط شامل یک دیاگرام ( deployment diagram ) می‌باشد.



## معرفی دیاگرامهای رشنال رز

یک دیاگرام یک نمایش گرافیکی از عناصر سیستمتان می‌باشد. دیاگرامهای گوناگون به شما اجازه می‌دهند تا پرسپکتیوهای مختلف از سیستم‌تان را ببینید.

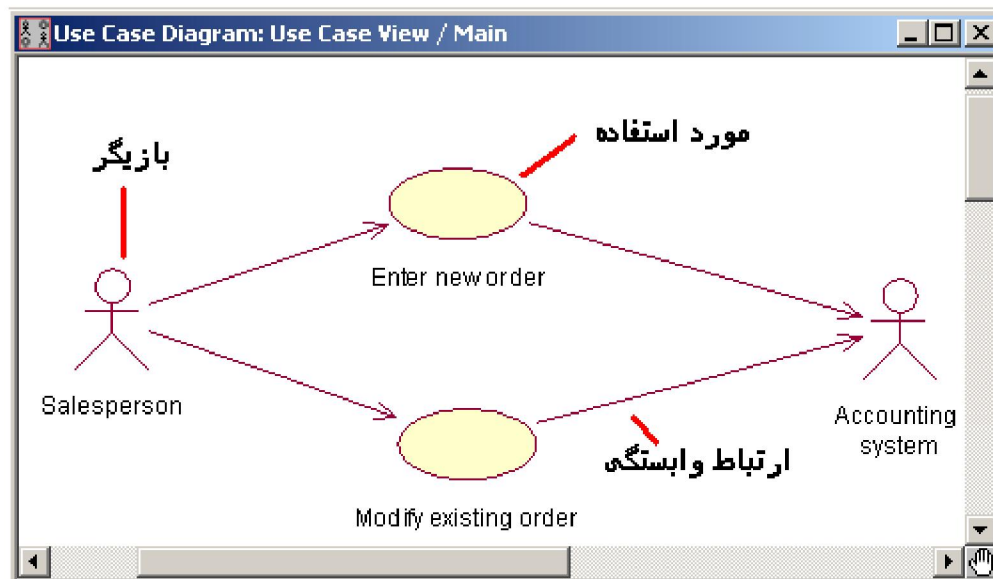
شما در رشنال رز قادر به ایجاد دیاگرامهای زیر می‌باشید:

- دیاگرام مورد استفاده (usecase diagram)
- دیاگرام کلاس (class diagram)
- دیاگرام توالی (sequence diagram)
- دیاگرام همکاری (collaboration diagram)
- دیاگرام فعالیت (activity diagram)
- دیاگرام حالت (statechart diagram)
- دیاگرام اجزا (component diagram)
- دیاگرام پیاده‌سازی (deployment diagram)

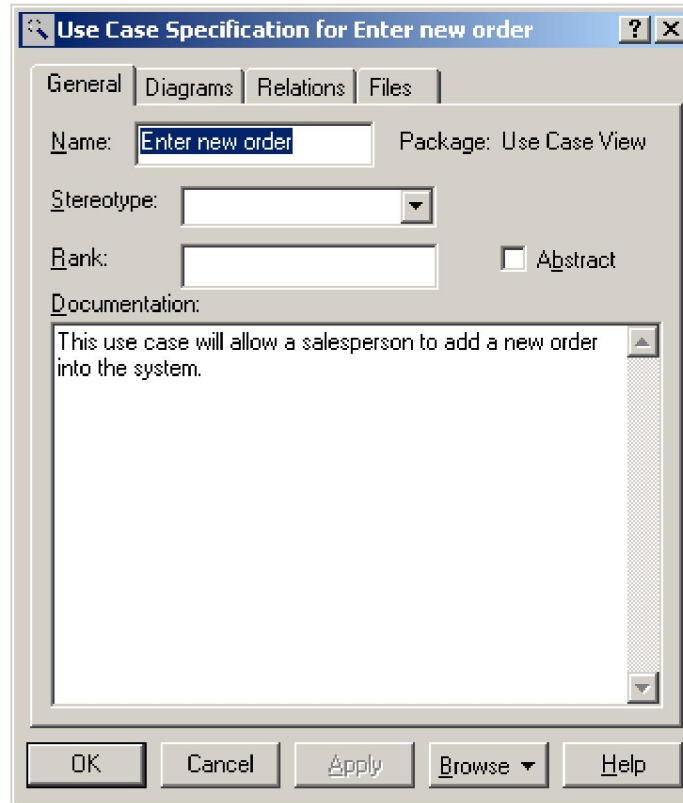
### دیاگرام مورد های استفاده (usecase diagram)

یک usecase رفتار سیستم را توصیف می‌کند، که شامل تقابل بین سیستم و بازیگران می‌باشد. عبارت کلی یک usecase یک الگوی رفتاری توانایی‌های سیستم و یک دنباله تراکنش‌های وابسته به هم می‌باشد، که بوسیله سیستم و بازیگران انجام می‌پذیرد و دیاگرام مورد های استفاده یک نمای سطح بالای سیستم از دید بازیگران سیستم به ما می‌دهد و نحوه‌ی برخورد آن با دنیای بیرون را مشخص می‌کنند. این دیاگرام در طول تحلیل سیستم برای بدست آوردن نیازمندی‌ها و نشان دادن چگونگی کارکرد سیستم بکار می‌رود.

**نکته:** در یک مدل واحد ممکن است چندین دیاگرام مورد استفاده داشته باشیم. معمولا در این صورت هر مجموعه‌ای از usecase های مرتبط به هم را در یک بسته قرار می‌دهیم. به این ترتیب کلیه usecase های سیستم در بسته‌های مرتبط به هم قرار می‌گیرند.



برای رسم یک دیاگرام مورد استفاده ابتدا در نمای usecase view روی آیکون main دو کلیک کنید تا دیاگرام مربوط به آن باز شود، کافی است که از جعبه ابزار سمت چپ، آیکون new use case را انتخاب کرده و بعد از قرار دادن روی دیاگرام با دو کلیک کردن روی آن در فرم specification نام، توضیحات و stereotype مربوط به آن را وارد نمایید.



اضافه کردن یک بازیگر هم مثل یک مورد استفاده می‌باشد. برای ساختن یک وابستگی جدید بین دو عنصر نیز روی آیکون association کلیک کرده و این وابستگی را با کلیک کردن روی عنصر اول و کشیدن تا عنصر دوم بسازید. سپس با دو کلیک کردن روی آن مشخصات آن را وارد نمایید.

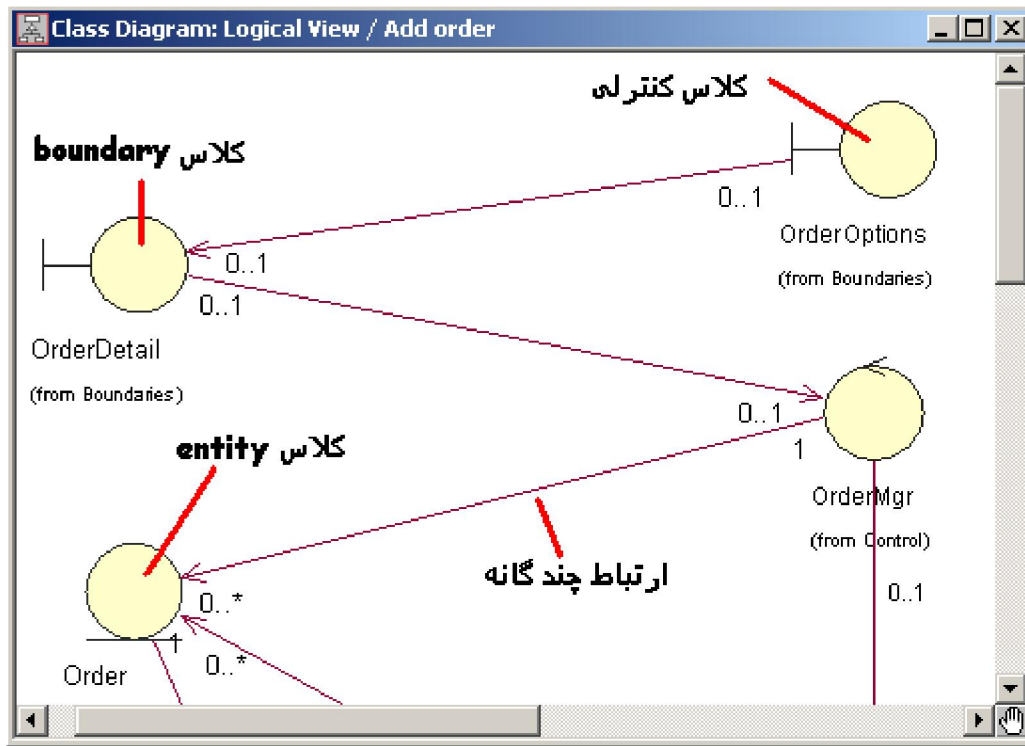
چگونگی اضافه کردن یک usecase موجود به یک دیاگرام دیگر: در browser دو دیاگرام مبدا و مقصد را باز (expand) کرده و سپس روی usecase مورد نظر کلیک کرده و آن را به دیاگرام مقصد بکشید. usecase مورد نظر در دیاگرام مقصد نیز ظاهر خواهد شد. اصولاً استفاده‌ی مجدد از عناصر موجود در سایر دیاگرامها نیز به همین شکل می‌باشد.

### دیاگرام کلاسها (class diagram)

این دیاگرام به شما کمک می‌کند تا نمای ساختاری سیستم‌تان را بصورت بصری (visual) در آورید. این دیاگرام از معمول‌ترین دیاگرامهای UML می‌باشد. این دیاگرامها جزئیات هر کلاس و ارتباطات بین آنها را نشان می‌دهد و پایه و اساس دیاگرامهای اجزا و پیاده‌سازی می‌باشد. در یک مدل واحد ممکن است چندین دیاگرام کلاس داشته باشیم.

در یک دیاگرام کلاس با کلاسهای با سه نوع stereotype متفاوت زیر سروکار داریم:

- boundary -
- control -
- entity -



**کلاسهای boundary** : اجزای لازم برای برقراری ارتباط سیستم با یک بازیگر را در خود دارند. این کلاسها می‌توانند پنجره ها، سنسورها، ترمینالها یا ... باشند. مثلا نجره‌ی گرفتن اسم رمز جهت ورود به برنامه، یک کلاس boundary می‌باشد. این نوع کلاسها معمولا جهت استفاده، با یک کلاس از نوع کنترلی در ارتباط هستند.

**کلاسهای کنترلی**: این کلاسها معمولا اشیا دیگر رفتارهای تعبیه شده در یک usecase را کنترل می‌کنند.

**کلاسهای entity** : این کلاسها اطلاعاتی را که باید توسط سیستم ذخیره گردند را در خود نگهداری می‌کنند. محل نگهداری ساختمان داده‌های منطقی سیستم، این کلاسها می‌باشند.

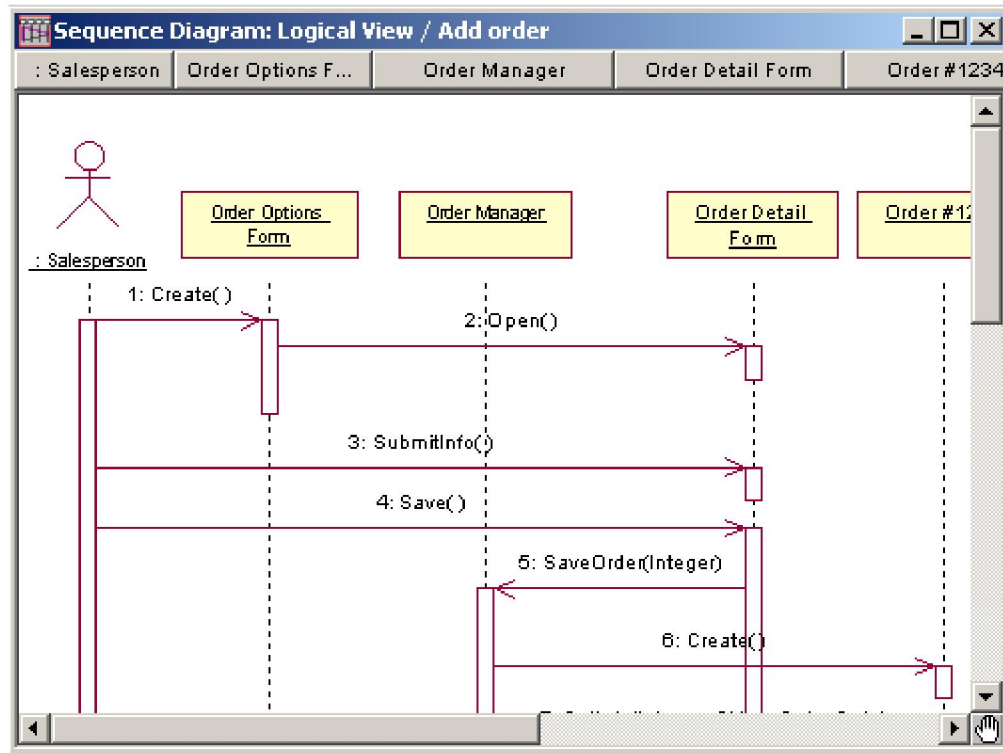
### دیاگرام توالی (sequence diagram)

دیاگرام توالی بصورت منظم و در یک توالی زمانی پشت سر هم ارتباطات متقابل اشیا را به ما نشان می‌دهد. این دیاگرام برای انجام عمل خاصی در یک usecase مشخص، مراحل انجام کار را مرحله به مرحله به شما نشان می‌دهد، یعنی دنباله‌ای از رویدادها را برای انجام یک عمل مشخص می‌سازد.



در مراحل تحلیل و طراحی برای فهم نحوه عملکرد سیستم از این دیاگرام استفاده می‌شود.

جهت ساخت یک دیاگرام توالی در قسمت browser رشنال رز در قسمت usecase view روی نود مورد استفاده‌ی مورد نظر راست کلیک کرده و new و سپس sequence diagram را انتخاب نمایید. روی نود جدید ایجاد شده دو کلیک کنید تا پنجره‌ی دیاگرام مربوط به آن باز شود.



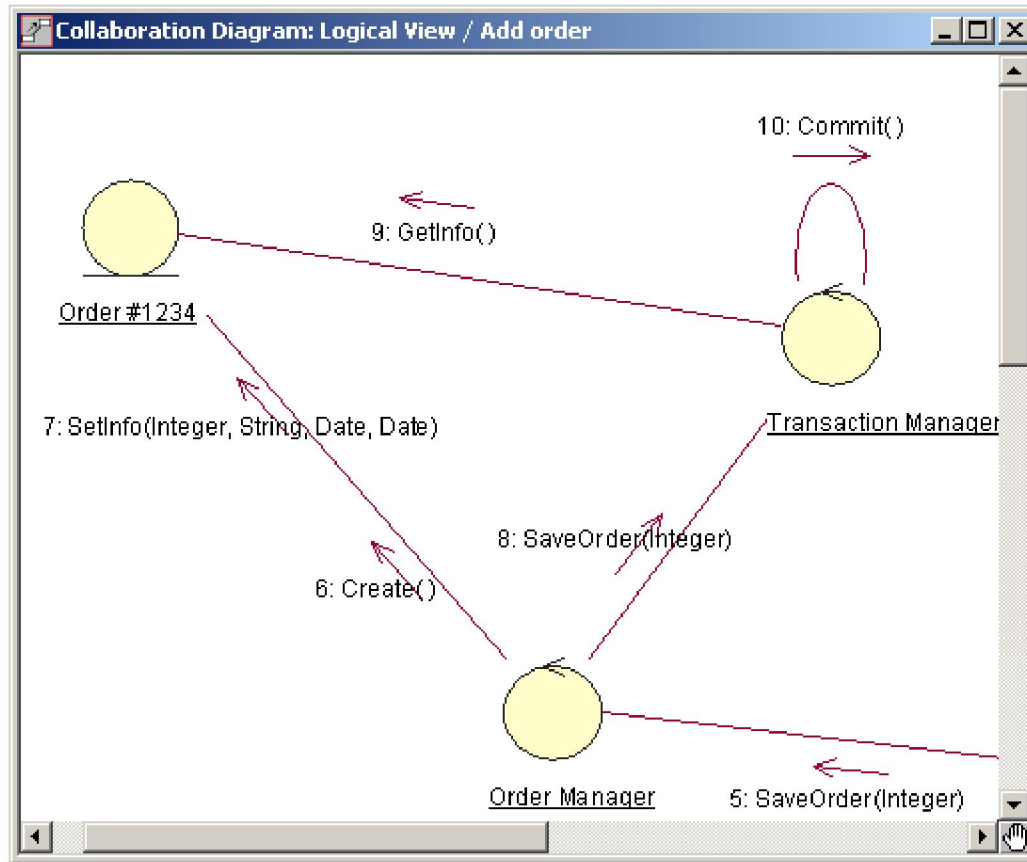
برای اضافه کردن کلاسها و یا بازیگرهای موجود به یک sequence diagram عناصر موجود در قسمت‌های قبلی را در browser کشیده و به داخل این دیاگرام بیندازید.

اگر می‌خواهید هنگام اضافه کردن پیغامها، عده‌هایی نیز به آنها اضافه گردد (همانند شکل فوق) تا توالی کار را بدرستی نشان دهد در منوی tools روی options کلیک کرده و به برگه‌ی دیاگرام رفته و جعبه‌ی انتخاب sequence numbering را علامت بزنید و سپس ok کنید. برای اضافه کردن پیغامها نیز می‌توانید از جعبه ابزار دیاگرام، آیکن object message استفاده کنید.

### دیاگرام همکاری (collaboration diagram)

یک نما از ارتباطات ساختاری بین اشیا در مدل فعلی را بما می‌دهد. تاکید این دیاگرام بر ارتباط بین اشیا است در حالیکه تاکید دیاگرامهای توالی بر روی دنباله‌ای از رویدادها بود. این نوع دیاگرام شامل اشیا، لینک‌ها، و پیغام‌ها می‌باشد. برای درک و فهم چگونگی رفتار سیستم واتخاذ هر گونه تصمیمی در این مورد از این دیاگرام استفاده کنید.

ساختن دیاگرامهای همکاری از دیاگرامهای توالی: رشنال رز این کار را بصورت خودکار برای شما انجام می‌دهد. کافی است از منوی browse گزینه‌ی create collaboration diagram را انتخاب نمایید، یا کلید F4 را بزنید. عکس این عمل هم، یعنی ساخت دیاگرام توالی از دیاگرام همکاری، با همین ترتیب فوق امکان‌پذیر است.



### دیاگرام فعالیت (activity diagram)

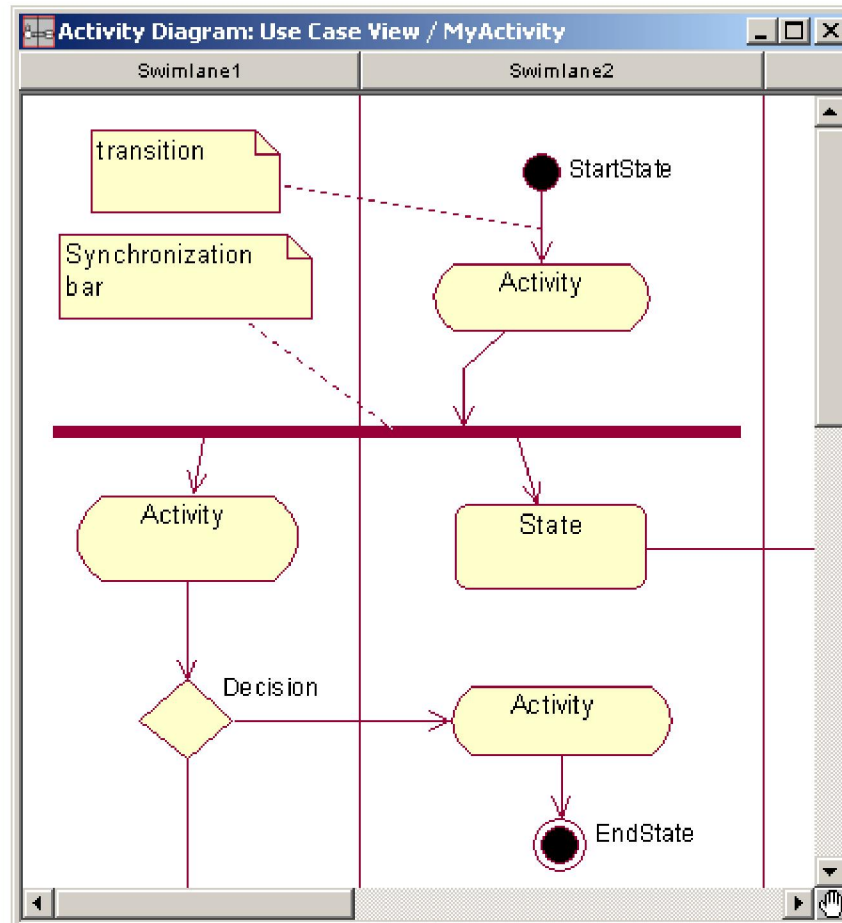
دیاگرام فعالیت، جریان کار و همچنین توالی فعالیت‌ها را در یک فرآیند مشخص می‌کند. این دیاگرام خیلی شبیه فلوچارت است، زیرا شما می‌توانید جریان کار را از یک فعالیت به فعالیت دیگر یا به حالتی دیگر، دنبال نمایید.

دیاگرامهای فعالیت همچنین در جاهایی که می‌خواهید رفتارهای موازی را توصیف کنید، یا چگونگی نشان دادن عکس العمل در مقابل یک وضعیت چندگانه را مشخص کنید، مفید هستند.

یک دیاگرام فعالیت می‌تواند عناصر زیر را داشت باشد:

- یک **start state** و یک **end state**
- **activity** ها، که مراحل را در جریان کاری نشان می‌دهند.
- **Transition** ها، که ترتیب کاری را نشان می‌دهند.

- **Decision** ها، که به شما اجازه‌ی تصمیم‌گیری‌ها را در دیاگرام می‌دهند.
- **Synchronization bar** ها : که اجازه‌ی نمایش کارهای موازی را در دیاگرام به شما می‌دهد.
- **Swimlane** ها : نقش‌های مسوول در یک فعالیت معین را مشخص می‌کند.



### ساخت یک دیاگرام فعالیت

در browser ، روی نود usecase view راست کلیک کرده و `new > activity diagram` را انتخاب نمایید. یک دیاگرام فعالیت جدید ظاهر می‌شود، کافی است روی آن دو کلیک نمایید تا دیاگرام مربوط به آن باز شود.

برای ساخت swimlane روی آیکن مربوط به آن در جعبه ابزار دیاگرام کلیک کرده و سپس روی نمودار کلیک نمایید. Swimlane جدید ایجاد می‌شود. برای تنظیم فیلدهای آن کافی است روی آن دو کلیک کنید، فرم مربوط به specification آن باز می‌شود.

سایر قسمت‌ها نیز روی جعبه ابزار دیاگرام مشخص است و اضافه کردن آنها همانند قسمت‌های قبل می‌باشد.

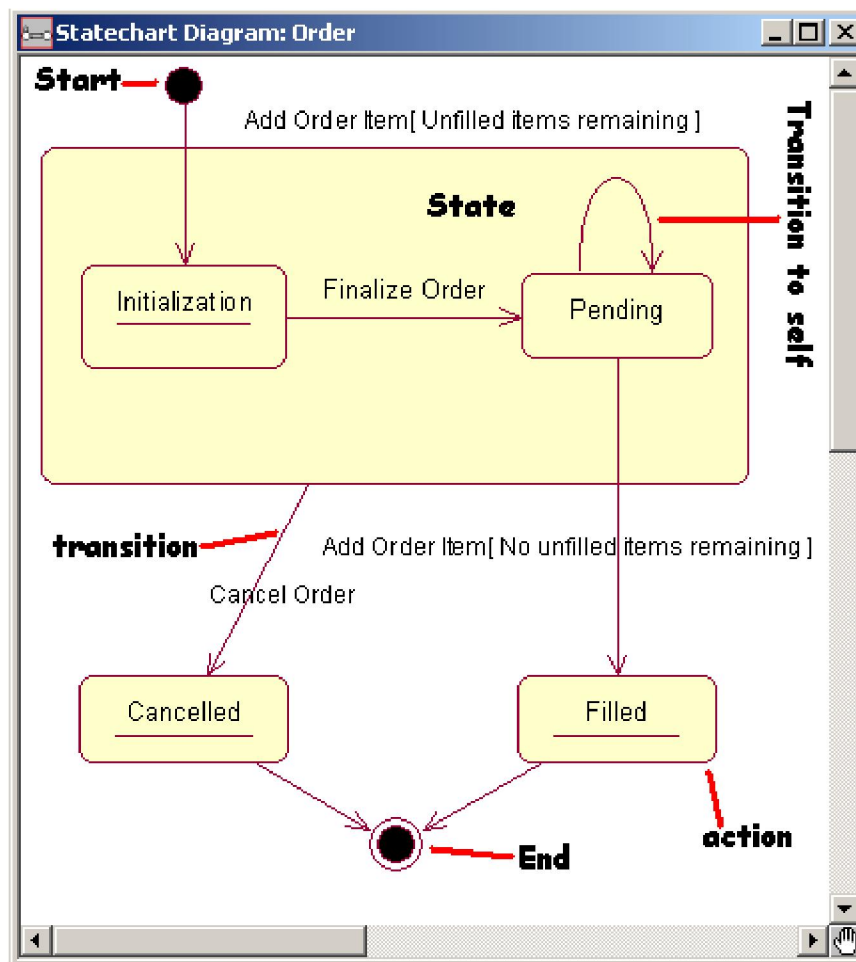
## دیاگرام حالت (statechart diagram)

می‌توانید دیاگرام‌های حالت را برای مدل کردن رفتار پویای کلاس‌ها یا اشیا انفرادی استفاده کنید. این دیاگرامها ترتیب حالاتی که یک شی می‌تواند داشته باشد، رویدادهایی را که موجب انتقال از یک حالت یا فعالیت به دیگری می‌شوند و نتایجی را که این انتقال بوجود می‌آورد، را نمایش می‌دهد.

یک دیاگرام حالت، معمولا برای مدل کردن مراحل گسسته‌ی چرخه‌ی حیات یک شی، بکار برده می‌شود، در حالیکه دیاگرام فعالیت بر توالی فعالیت‌ها یی در یک فرآیند، دلالت دارد.

عناصر اصلی یک دیاگرام حالت عبارتند از:

- state ها ( وضعیت‌های یک شی در طول حیات آن )
- حالت‌های شروع (start) و پایان (end)
- انتقال حالت‌ها (transitions)
- عملهای Entry ، Do و exit.



**نکته:** هر حالت در یک دیاگرام حالت می‌تواند شامل چندین عمل داخلی باشد. هر عمل، وظیفه‌ای است که در یک حالت رخ می‌دهد، که می‌تواند یکی از اعمال زیر باشد:

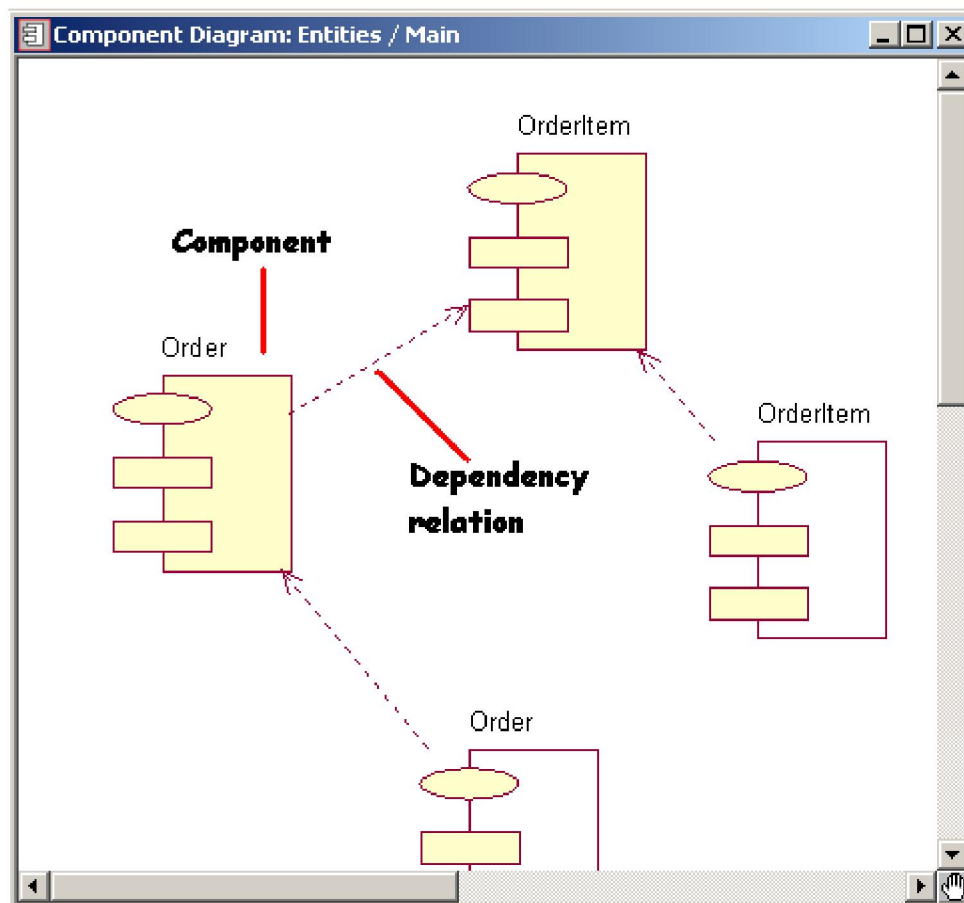
- OnEntry -
- OnExit -
- Do -
- OnEvent -

برای ساخت یک دیاگرام حالت برای یک کلاس کافی است روی کلاس مورد نظر در browser در نمای منطقی کلیک راست کرده و new statechart diagram را انتخاب کنید.

### دیاگرام اجزا (component diagram)

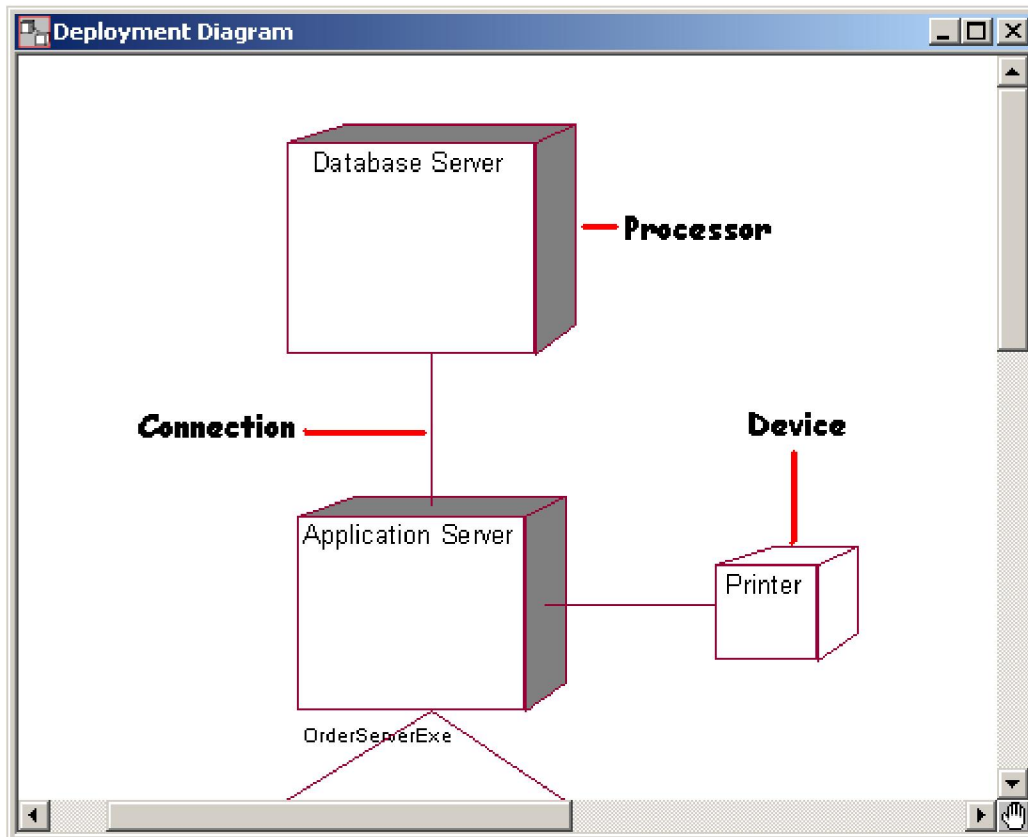
یک نمای فیزیکی از مدل جاری را به ما می‌دهد. این دیاگرام نحوه‌ی سازماندهی اجزای نرم‌افزار و وابستگی بین آنها را به ما می‌دهد، که شامل سورس کد، کد باینری و اجزای قابل اجرا می‌باشد. شما می‌توانید یک یا چند دیاگرام اجزا را برای مجسم ساختن اجزا وابسته‌ها و یا محتویات هر بسته به کار ببرید.

**نکته:** هر مدل می‌تواند شامل چندین component با زبانهای متفاوت باشد، ولی هر کلاس می‌تواند فقط به component هایی با یک زبان یکسان نسبت داده شود.



### دیاگرام پیاده‌سازی (deployment diagram)

هر مدل شامل فقط یک دیاگرام پیاده‌سازی است، که نگاهی از فرآیندهای موجود با سخت‌افزار سیستم را نشان می‌دهد.



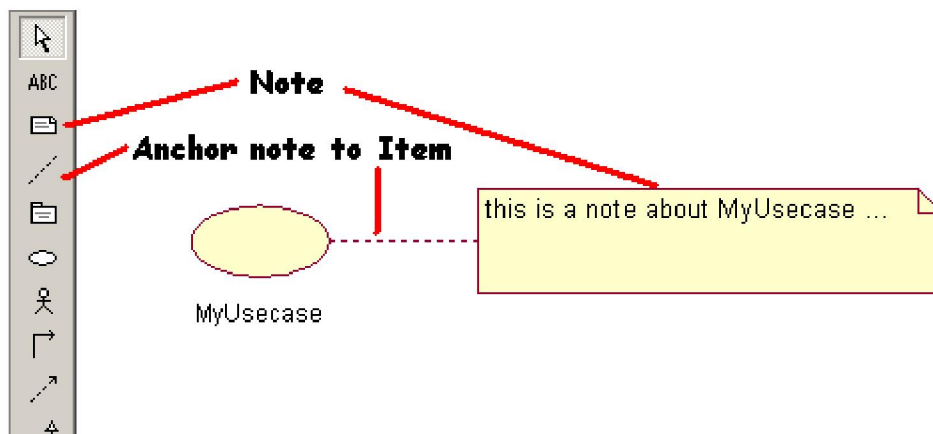
## مراحل کاری انجام یک پروژه در رشنال رز:

- ۱- شناخت سیستم موجود و آشنایی با پروژه و در صورت نیاز در سیستم‌های حجیم، مدل کردن سیستم موجود بصورت اجمالی شامل نمودارهای موردهای استفاده، نمودارهای فعالیت جاری و نمودار کلاس‌ها.
- ۲- مدل کردن نیازهای عملیاتی سیستم. که این بخش شامل تعیین بسته‌ها، رسم نمودارهای مورد استفاده و رسم دیاگرامهای فعالیت می‌باشد.
- ۳- ساخت مدل تحلیلی سیستم، شامل دیاگرامهای کلاس، توالی، همکاری و حالت.
- ۴- ساخت مدل پیاده‌سازی برنامه، شامل دیاگرامهای اجزا، پیاده‌سازی و بروز در آوری کد و مدل.
- ۵- ساخت پایگاه داده‌های سیستم.
- ۶- گزارشگیری و منتشر کردن پروژه.

حال پروژه‌ی خود را تعریف می‌کنیم و مراحل کاری فوق را تا تولید کد دلفی، در رشنال رز روی آن نشان می‌دهیم:

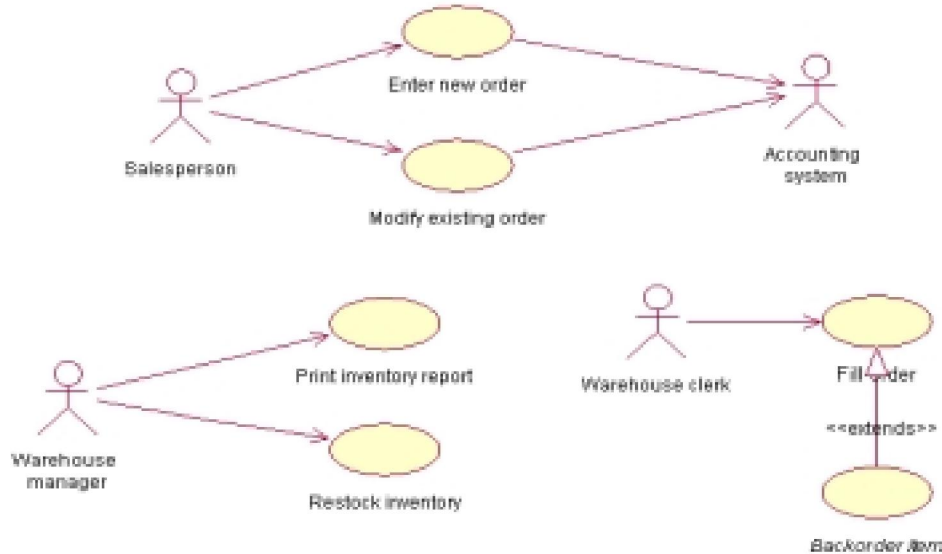
**تعریف پروژه:** این پروژه یک سیستم سفارش با یک کامپیوتر مرکزی و دو station برای گرفتن سفارشات مشتریان می‌باشد. سفارشات مشتریان در قسمت انبار توسط کارمندان مربوطه انجام می‌پذیرد. سیستم باید قادر باشد در هر لحظه گزارش فهرست کالاهای موجود در انبار را بدهد.

قبل از توضیح مراحل پروژه لازم است به این نکته اشاره شود که برای قابل درک و فهم ساختن اجزای نمودارهای مدل، بخصوص در جاهاییکه تعداد آنها زیاد است شما باید به آنها یادداشتهایی را اضافه کنید. برای اینکار در نمودار دیاگرامها در جعبه ابزار دیاگرام، آیکن Note را انتخاب کرده در صفحه‌ی دیاگرام قرار دهید. سپس با یک خط چین ارتباط این یادداشت را به تکه‌ی مورد نظر برقرار کنید. ولی در این پروژه بعلت کوچک بودن و برای جلوگیری از شلوغ شدن متن و پرداختن به نکات اصلی از آوردن یادداشتهای بصورت note خودداری شده است.



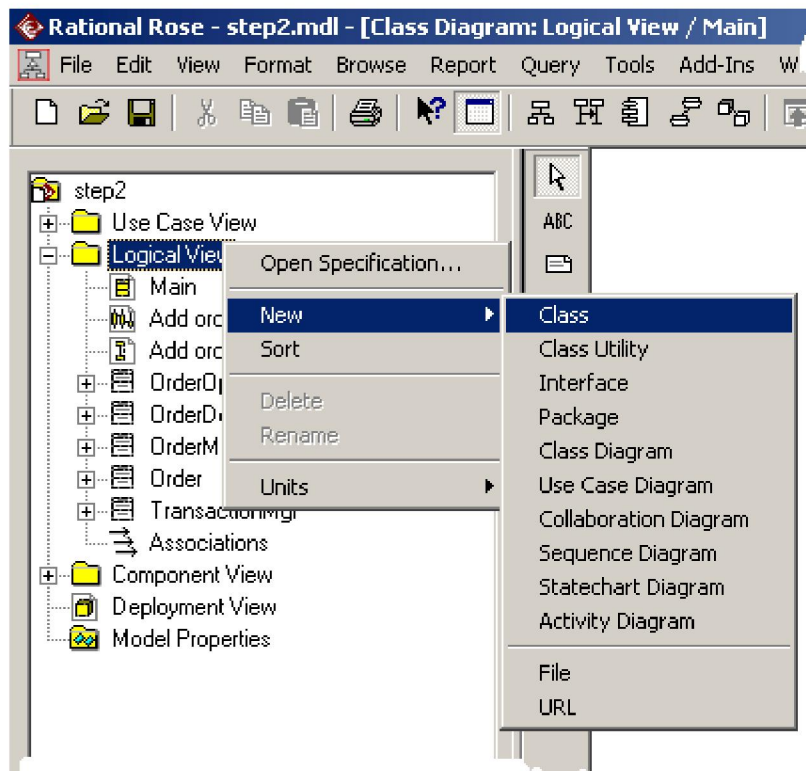
### مرحله ی اول پروژه ( فایل step1.mdl )

رسم نمودارهای usecase ها: همانطوریکه در قسمت قبل توضیح داده شد، در usecase view ما نمودار مورد استفاده ی خود را رسم می کنیم. برای پروژه ی ما این نمودار به شکل زیر در می آید:



### مرحله ی دوم پروژه ( فایل step2.mdl )

ابتدا کلاسهای مورد نظر را با در نظر گرفتن اشیا سیستم، مشخص می کنیم. سپس این کلاسها را به قسمت logical view اضافه می کنیم. اضافه کردن یک کلاس به این صورت می باشد که ابتدا روی نمود logical view راست کلیک می کنیم، سپس از منوی ظاهر شونده، new > class را انتخاب می نماییم. سپس مشخصات کلاس مورد نظر را وارد می نماییم.



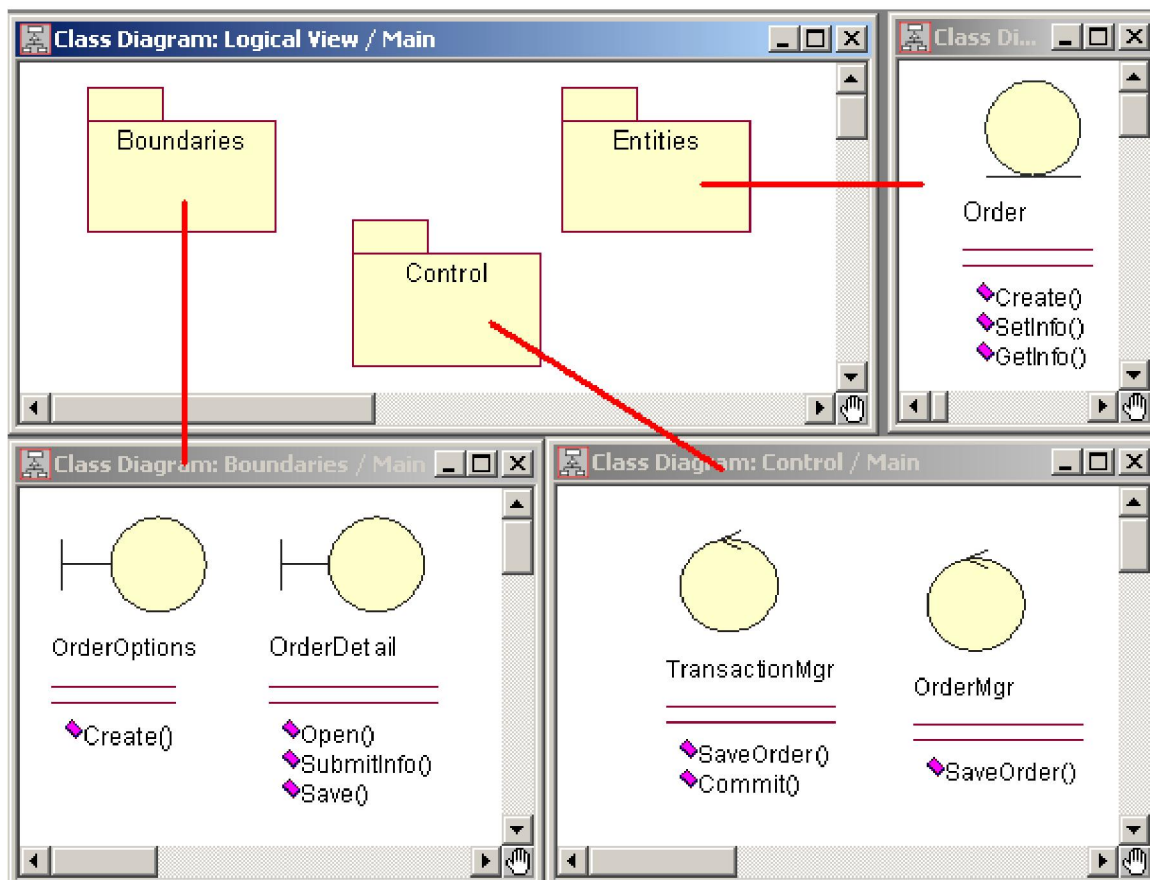


در ادامه‌ی این مرحله به رسم نمودارهای توالی می‌پردازیم. این نمودار با استفاده از اشیا موجود (بازیگرهای معرفی شده در دیاگرام موردهای استفاده و نمونه‌های کلاسهای معرفی شده در همین مرحله) ساخته می‌شود. کافی است اشیا را در قسمت browser انتخاب کرده و به داخل دیاگرام توالی بیندازیم. حال باید پیغامهای ارتباطی، را بین این اشیا برقرار کرد. پس از رسم این پیغامها با استفاده از این نمودار متدهای مربوط به هر کلاس را به آن add می‌کنیم. متد مربوط به هر پیکان در کلاس شی مقصد آن پیکان قرار می‌گیرد.

برای add کردن یک متد به یک کلاس کافی است، در browser روی آن راست کلیک کرده و `new > operation` انتخاب نماییم، سپس در کادر فعال شده مشخصات متد مورد نظر را وارد کنیم.

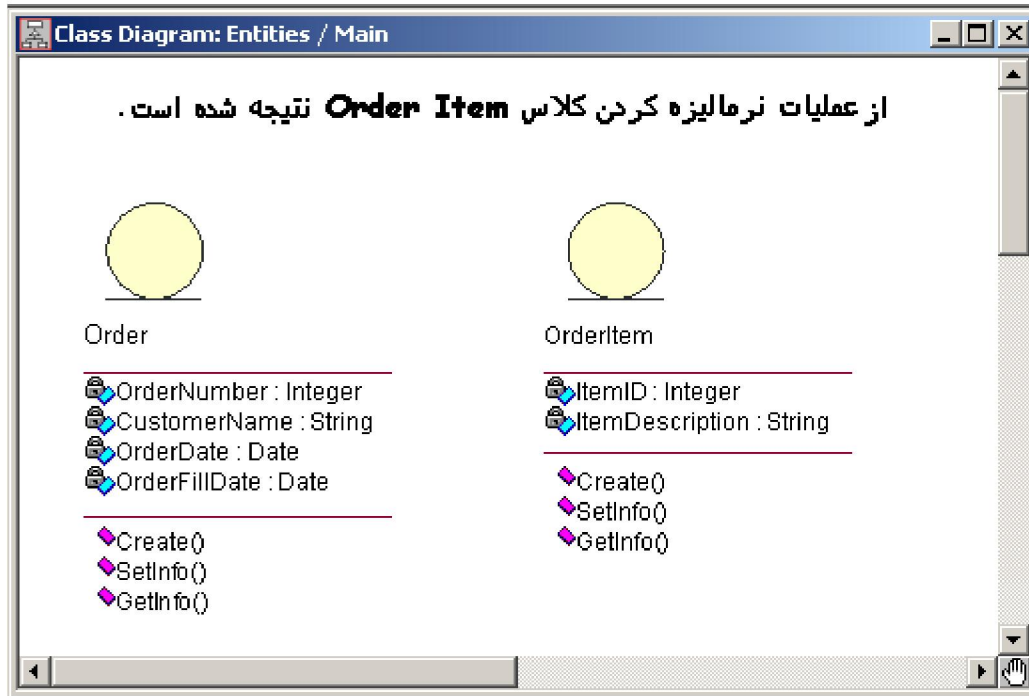
### مرحله‌ی سوم پروژه ( فایل step3.mdl )

حال در قسمت main مربوط به نمای منطقی سه بسته بعنوان سه عنصر معماری سه لایه قرار می‌دهیم و کلاسها را در بسته‌های مربوطه تقسیم بندی می‌کنیم. این کار با کشیدن هر کلاس و رها کردن آن روی بسته‌ی مورد نظر انجام می‌پذیرد. اکنون stereotype مربوط به هر کلاس را با دوکلیک کردن روی آن کلاس و با توجه به بسته‌ی قرار گرفته در آن تنظیم می‌نماییم.



### مرحله چهارم پروژه ( فایل step4.mdl )

حال باید کلاسهای بسته‌ی entity را نرمالیزه نماییم. بعد از نرمالیزه کردن باید در صورت تشکیل کلاسهای جدید، این کلاسهای تولیدی و کلیه‌ی attribute‌های مربوط به کلاسهای entity را وارد نماییم. وارد کردن یک attribute جدید به یک کلاس، با کلیک راست کردن روی آن و انتخاب گزینه‌ی `new > attribute` صورت می‌پذیرد.

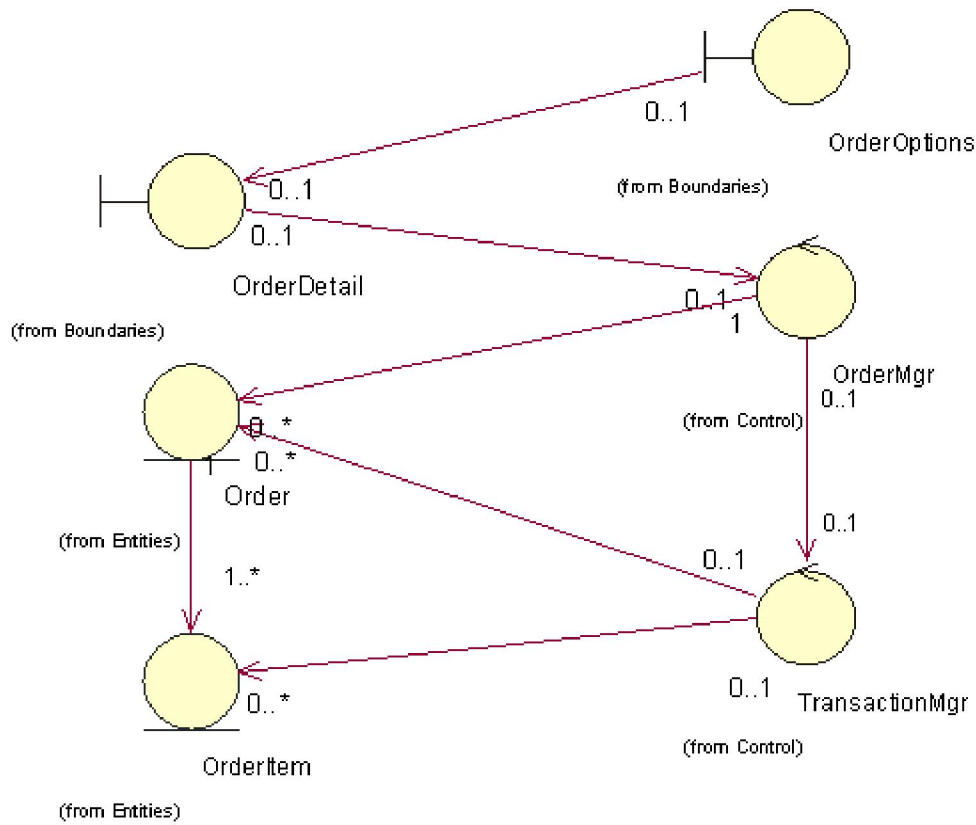


### مرحله پنجم پروژه ( فایل step5.mdl )

در کنار دیاگرام کلاس اصلی که بصورت سه بسته طراحی شد، ما باید یک ارتباط بین کلیه‌ی کلاسهای موجود را نشان دهیم، چون ممکن است ارتباطی بین کلاس موجود در یک بسته با کلاس موجود در بسته‌ی دیگر وجود داشته باشد، لذا ناچاریم دیاگرامهای کلاس دیگری نیز داشته باشیم که کلیه‌ی کلاسهای مربوط به هم را یکجا جمع‌آوری نموده و ارتباط بین آنها را مشخص نماید.

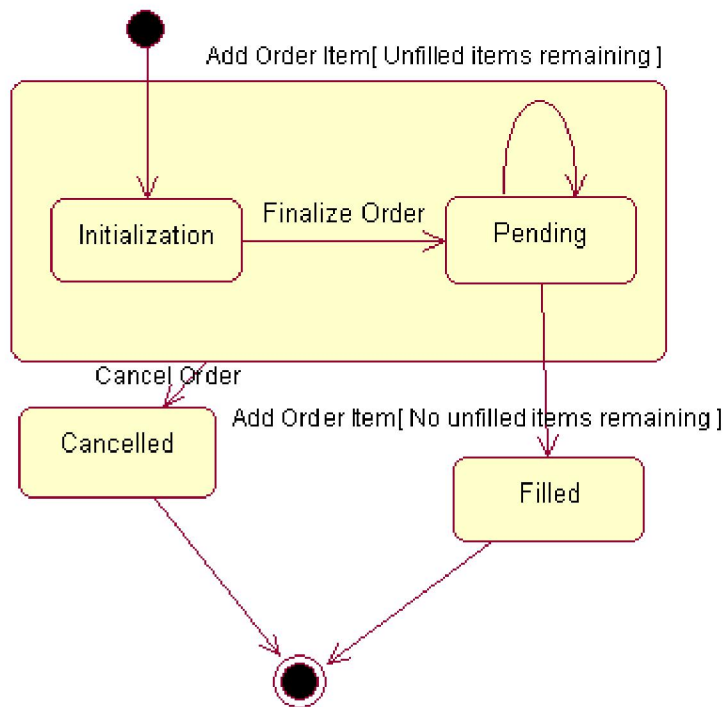
برای چگونگی مشخص کردن رابطه‌های چند گانه بین دو کلاس مشخص، روی فلش ارتباطی بین آنها کلیک کرده و بسته به هدف مورد نظر به یکی از برگه‌های Role A Detail و یا Role B Detail می‌رویم، سپس در فیلد multiplicity نوع رابطه را انتخاب می‌کنیم.

ما در این مرحله بصورت زیر ارتباط منطقی بین کلاسها و رابطه‌ی آنها را مشخص می‌کنیم:



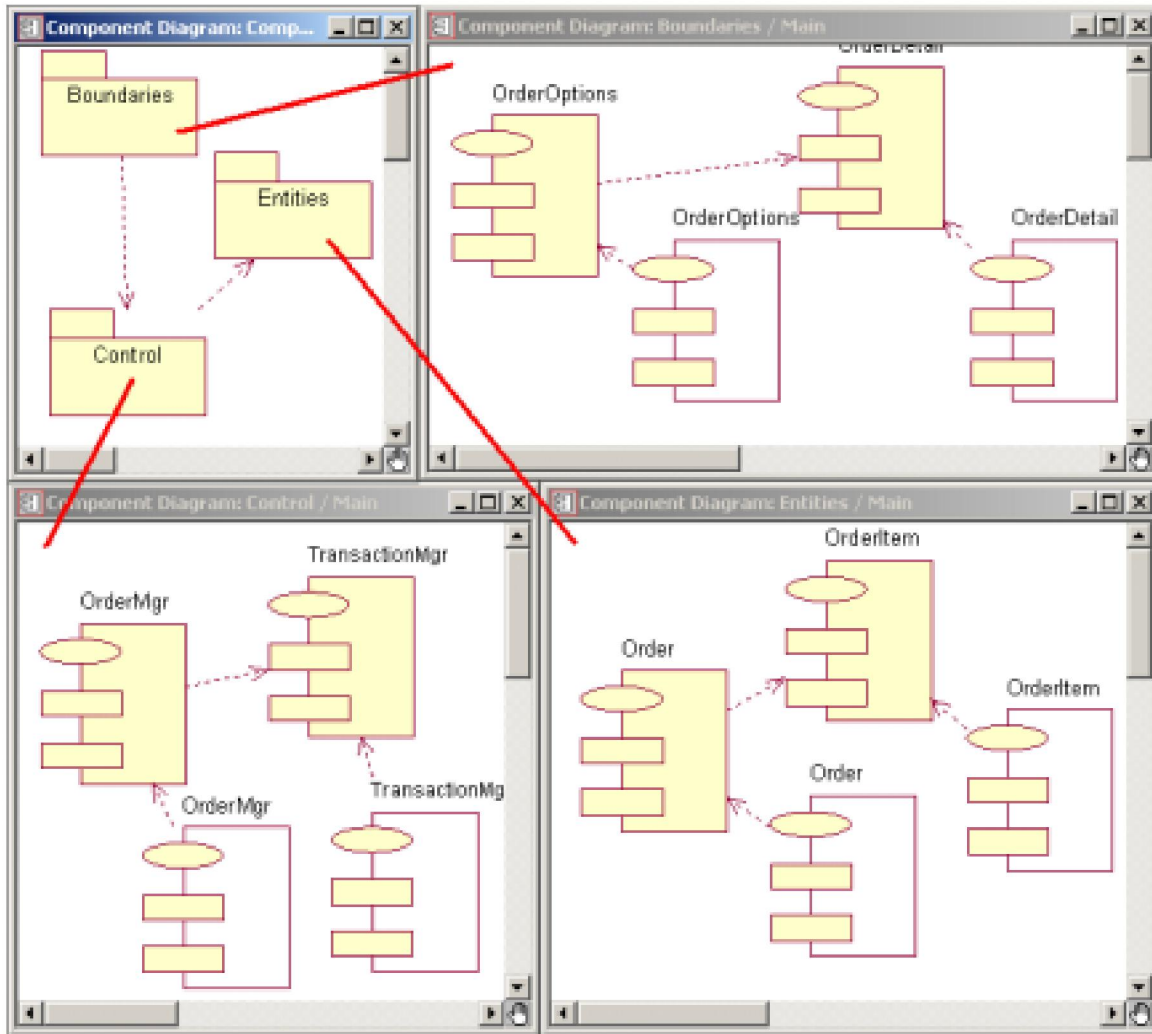
**مرحله‌ی ششم پروژه ( فایل step6.mdl )**

گفتیم که دیاگرام حالت وضعیت‌های مختلف یک شی را در طول عمر آن نمایش می‌دهد. در نتیجه برای اشیایی که لازم باشد (کلاسهای از نوع entity) در صورت لزوم دیاگرام حالت را رسم می‌کنیم.



**مرحله هفتم پروژه ( فایل step7.mdl )**

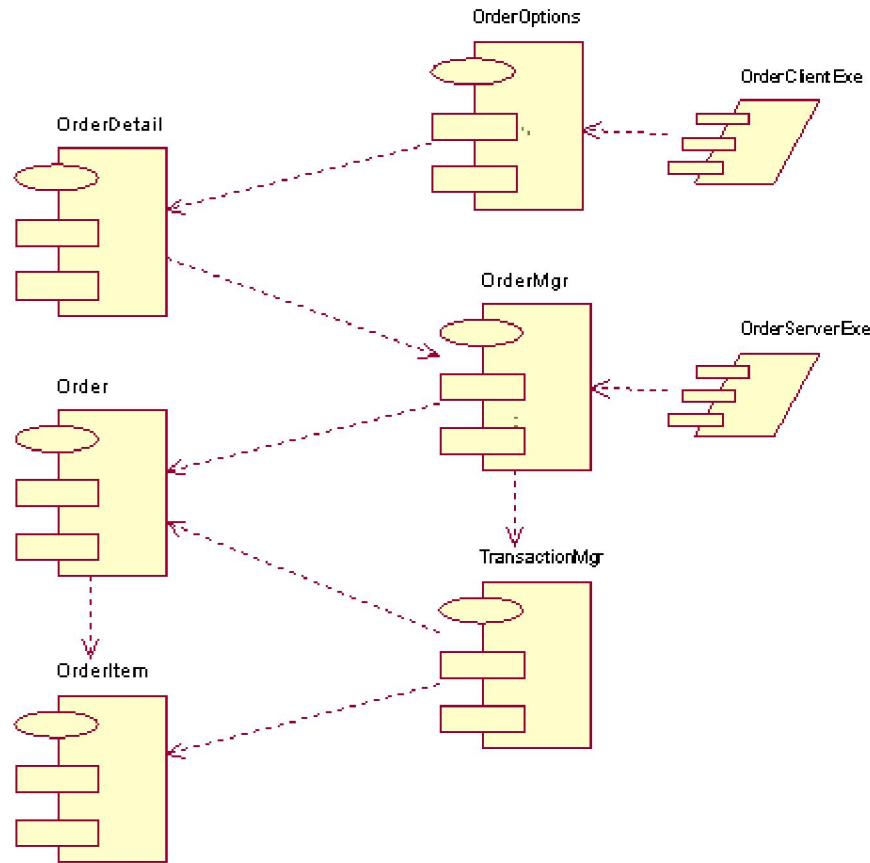
حال نوبت به بخش component view می‌رسد. ابتدا در دیاگرام اصلی این نما سب بسته‌ی اصلی معماری سه‌لایه را قرار داده و ارتباط منطقی بین آنها را برقرار می‌کنیم.



سپس یک نمودار component diagram جدید می‌سازیم (وجود این نمودار به این دلیل است، چون در main ما سه بسته‌ی جدا داریم، در نتیجه نمی‌توانیم ارتباط قسمت‌های مختلف بین دو بسته را برقرار کنیم)، حال به ازای هر کلاس که در دیاگرام اصلی کلاس‌هایمان داریم یک جز Package specification قرار می‌دهیم و ارتباط بین آنها را برقرار می‌کنیم.

چون برنامه‌ی ما یک برنامه‌ی Client/Server می‌باشد، دو جز Task specification در نمودارمان قرار می‌دهیم، یکی مربوط به برنامه‌ی اجرایی روی Client می‌باشد و دیگری مربوط به Server است.

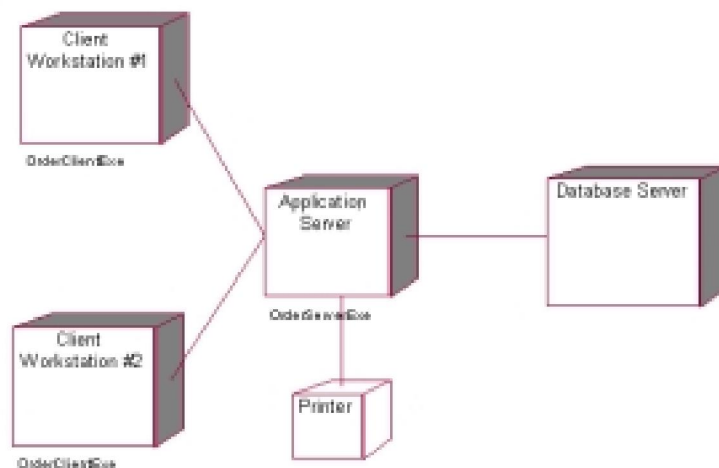
شکل مربوط به این دیاگرام در صفحه‌ی بعد آمده است :



دیاگرام ارتباطی اجزا

### مرحله‌ی هشتم پروژه ( فایل step8.mdl )

با توجه به تعریف پروژه به سادگی می‌توان نمودار پیاده‌سازی زیر را برای سیستم در نظر گرفت. تنها نکته‌ای که نباید فراموش گردد، نسبت دادن پروسه‌های اجرایی روی کامپیوترهای server و client می‌باشد. برای اینکار در browser روی نود مربوط به پردازشگر مورد نظر راست کلیک کرده و `new > process` را انتخاب نمایید و پروسه‌ی مورد نظر را به آن نسبت دهید.

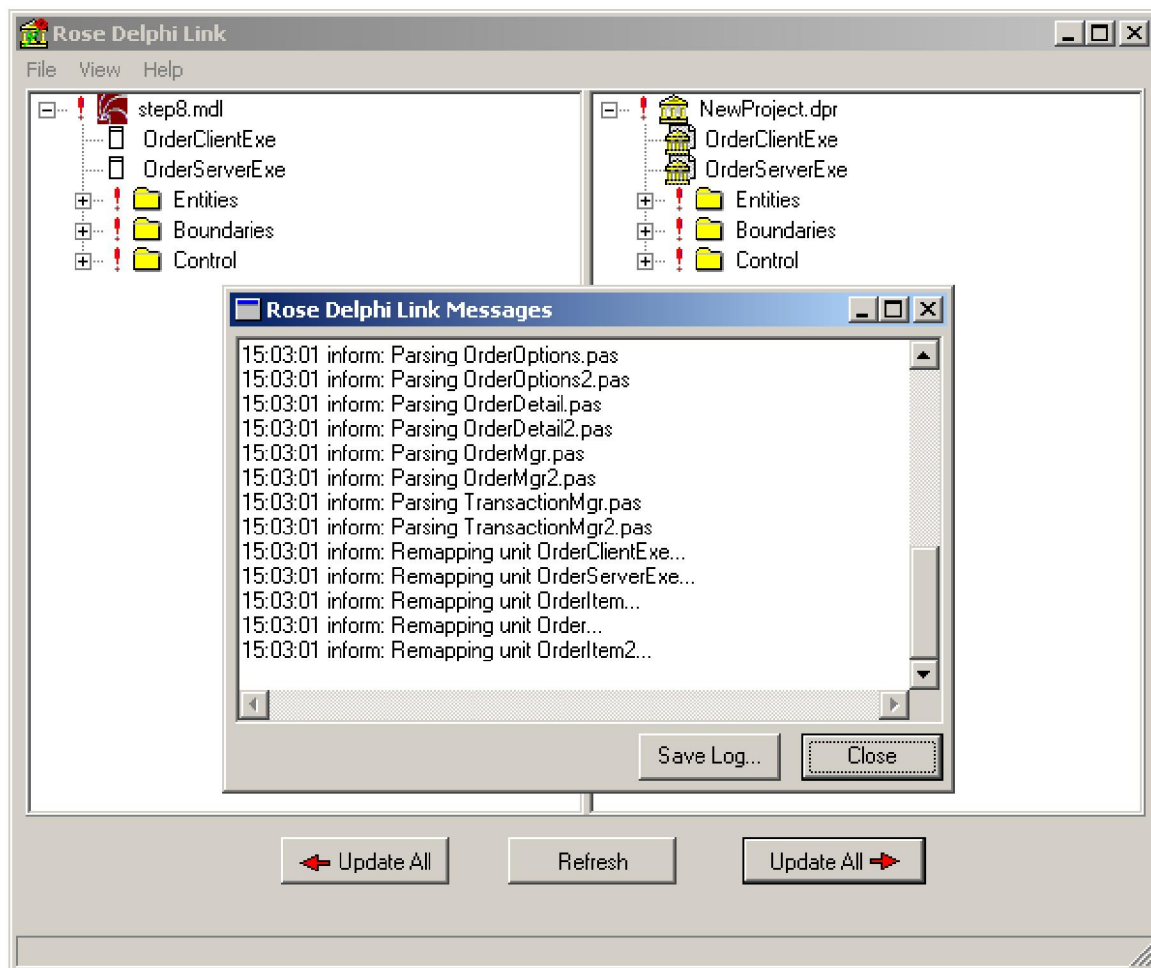


### چگونگی تولید کد دلفی از مدل و یا عکس عمل فوق:

ابتدا روی componentهای مورد نظر دو کلیک نمایید. سپس در specification مربوط به آن در قسمت language زبان دلفی را انتخاب نمایید.

حال از منوی tools گزینهی Ensemble Tools > Rose Delphi Link ... را انتخاب نمایید. اکنون برای برو ز در آوری کد مورد نظر و یا update کردن مدل با توجه به کد موجود، پروژهی مورد نظر را باز کرده و با توجه به جهت نشان داده شده، عملیات مورد نظر را انجام دهید.

اگر هدف شما تولید کد می باشد و پروژه را قبلا نساخته اید، ابتدا می بایستی از منوی File گزینهی new project را انتخاب نمایید. ولی بجای اینکار، پیشنهاد می کنیم که ابتدا یک پروژهی خالی را توسط دلفی ساخته و بجای انتخاب این گزینه، پروژهی ساخته شده را باز نمایید.

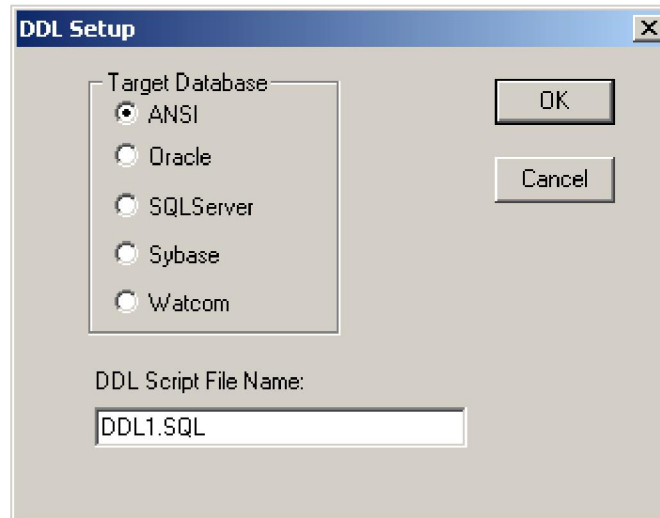


### نحوه ی تولید پایگاه داده ی مربوط به پروژه :

می دانیم که بانک اطلاعات پروژه از کلاسهای entity بدست می آیند.. برای تولی تولید کد SQL مربوطه، مطابق با هر کدام از بانکهای موجود نظیر access ، SQL Server ، oracle ، یا ... ، ابتدا باید کلاسهای entity خود را به صورت

persistant در آوریم. این کلاسها، همانند دیگر کلاسها بصورت پیش فرض transient هستند. برای این تغییر کافی است روی هر کلاس دو کلیک کرده و به در فرم specification به برگه‌ی Detail بروید و تغییرات لازم را اعمال کنید.

بعد از این تبدیل کلاسهایی را که می‌خواهید برای آنها کد SQL تولید گردد، را انتخاب نمایید. سپس در منوی tools گزینه‌ی DDL > Generate code را انتخاب نمایید. کد مورد نظر در فایل انتخابی شما ذخیره می‌گردد.



برای مشاهده‌ی کد تولید شده می‌توانید از منوی tools گزینه‌ی DDL > Browse DDL را انتخاب نمایید.

```

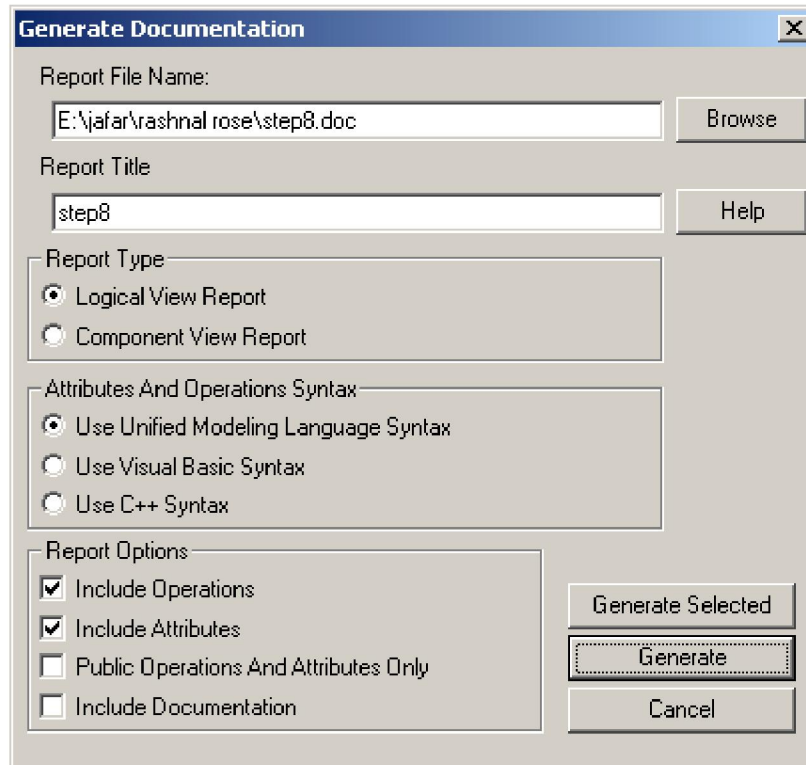
CREATE TABLE T_OrderItem(
  ItemID VARCHAR,
  ItemDescription VARCHAR,
  OrderItemId NUMBER(5),
  PRIMARY KEY(OrderItemId))

CREATE TABLE T_Order(
  OrderNumber VARCHAR,
  CustomerName VARCHAR,
  OrderDate VARCHAR,
  OrderFillDate VARCHAR,
  OrderItemId NUMBER(5) REFERENCES T_OrderItem(OrderItemId),
  OrderId NUMBER(5),
  PRIMARY KEY(OrderId))

```

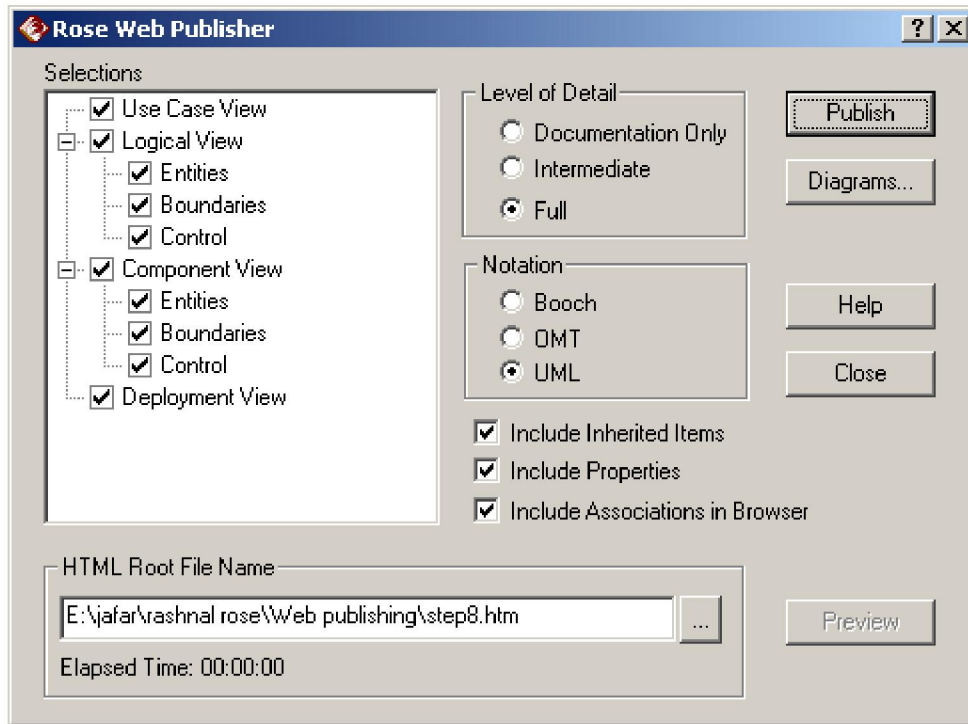
**نحوه‌ی گزارش گرفتن از پروژه:**

برای تولید مستندات مدل‌تان ( نماهای منطقی و اجزا) و مشاهده‌ی آنچه که رشنال رز برای شما تولید کرده است، از منوی reeport گزینه‌ی Documentation report ... را بکار ببرید. گزارشهای تولیدی برای پروژه‌ی ما در انتهای نوشتار ضمیمه می‌باشد.

**نحوه‌ی انتشار پروژه:**

در منوی tools گزینه‌ی Web publisher... عملیات انتشار را برای پروژه‌ی شما انجام می‌دهد. این عمل یک نسخه‌ی html از تمامی مدل شما همراه با مستندات تهیه می‌کند، تا سایر کاربران با دیگر browserهای موجود نیز بتوانند مدل شما را ببینند. فرم مربوطه در صفحه‌ی بعدمی‌باشد و فایل html تولیدی نیز در دیسکت ضمیمه موجود می‌باشد.





فرم تولید فایل html مربوط به مدل