بخش دوم

تجزیه و تحلیل

شیءگرا

مبتنی بر زبان

UML

فهرست مطالب

فصل ۱

مقدمه

۱-۱ هدف

در این بخش از کتاب بازهم سعی شده تا مبتنی بر تحلیل مطالب گردآوری شده و برخلاف خواسته صاحبان علوم، در عمل و به صورت کاربردی روشی برای مهندسهای نرم افزار در جهت طراحی و پیاده سازی برنامه های کاربردی ارائه شود. در این راستا، روش پیشنهادی RPM[۱] که از زبان UML جهت بیان مدلها بهره می جوید، مطرح خواهد شد.. UML[۲] زبان و مجموعه ای است از علائم که برای مدلسازی سیستمها به روش شی ء گرا پیشنهاد شده است .

در طراحی یک سیستم اولین سوال تعیین وظایف اشیاء است . سوال دوم چگونگی همکاری بین کلاسها در جهت انجام وظایف کلی سیستم است . در واقع کلاسها مبین ابزار در یک مدل شی ء گرا هستند. از ارتباط بین این ابزار است که اهداف و وظایف سیستم به انجام می رسد. بنابراین ، در طراحی شی ء گرا، چگونگی و مسئولیت کلاسها به عنوان یک سوال اولیه مطرح است

مشکل اصلی در نگرش شی ء گرا تعیین عملکرد سیستمها از ارتباط اشیاء تشکیل دهنده آن است . تکه کردن سیستمها بر اساس اجزاء تشکیل دهنده نمی تواند روشی جامع برای شناخت باشد. چگونه می توان با تکه تکه نمودن به اجزاء دست ، پا و قلب، انسان را مورد شناسایی قرار داد. ترکیب اجزا خود هویتی جداگانه دارد. تکه کردن مغایر وحدت وجود است . تجزیه و تکه تکه کردن راه عقل برای شناخت است . پیروان نظریه وحدت رها کردن ذهن از بند تفکر مکانیکی و فریبهای عقل و در اصطلاح دیوانگی را راه شناخت وحدت وجود می دانند.

آزمودم عقل دور اندیش را بعد از این دیوانه سازم خویش را

---

[۱] Recommanded Process Model

[۲] Unified Modeling Language

به بیان دیگر جهت درک وحدت باید روش دیگری را با روش تفکر و استدلال جـایگزین نمـود. عارف راه دل را پیشنهاد می کند. اما چگونه می توان از موجودی که سراسر عمر خود را به عاشق نبودن گذرانده انتظار عشق داشت . پاسخ به این سوال را می توان در زندگی عرفا جستجو کرد. آنانکـه چـون مسیح به راه عشق رفتند و از وادی روشنایی برای بشـریت فرهنـگ ، هـنر و سـایر تجلیـات انسـانی را بـه ارمغان آوردند.

گر روی پاک و مجرد چو مسیحا به فلک از چراغ تو به خورشید رسد صد پرتو

در کیش مهر ، سرلوحه زندگی عشق است و عشق است که می تواند انسان را به وحـدت برسـاند. در این کیش هنگامی که انسان عاشق می شود با جهان به وحدت می رسد. آنگاه است که دریچه هـای نور به رویش باز می شود. سوال اینجا است . **آیا می توان عنصر عشق را در شـناخت سیسـتمها وارد نمود؟** در این کیش غیر ممکن وجود ندارد.

عرفا راهنمای خود را مکاشفه و شهود می دانند. اسـتدلال عرفـا بـر وحـدت وجـود از راه مسـاوی بودن وجود با وجوب ذاتی و مساوی بودن وجوب ذاتی با وحدت از جمیع الجهات است . بنابراین مـی توان نتیجه گرفت که وجود مساوی با وحدت از جمیع الجهات است .

وجود شی ء با علت خود یک رابطه و نسبت وجودی دارد که با هیچ چـیز دیگـری نـدارد. عـامل وجود دهنده فعل خویش است و مفهوم علیت هم چیز دیگری نیست

۱-۲ مروری بر روش شی ءگرا

لازمه ایجاد برنامه صحیح ، شناخت دقیق نیازمندیها است . باید مشخص نمود، هدف چیست و چه مسائلی مطرح می باشد. تعیین راه حل مقدمه ای بر ایجاد یک برنامه کامپیوتری است. راه حل را باید در قالب مجموعه أی از اشیاء دید که با همکاری یکدیگر، در جهت اهداف تعیین شده پاسخگوی مسائل و نیازهای مطرح می باشند. نگرش شی ءگرا به شناخت قلمرو مساله و یافتن راه حلهای منطقی بر پایه اشیاء تکیه دارد. باید توجه نمود که کلمه شیء ، به هر نوع موجودیت فیزیکی و یا مفهومی اطلاق می شود. در دیدگاه شی ء گرا عملکرد را در قالب مورد های استفاده از سیستم مورد شناخت و شناسایی قرار می دهند. برای نمونه توصیف در"خواست سفارش" به عنوان یک مورد استفاده، ابزاری جهت شناخت عملکرد و نیازمندیهای سیستم سفارشها است .

مورد استفاده : درخواست سفارش

شرح : این مورد استفاده، هنگامی که یک مشتری برای خرید کالا مراجعه می کند شروع می شود. درخواستهای مشتری همراه با اطلاعات محصول در یک سفارش جدید ثبت میشود. مشتری باید دارای پرونده باشد. در مورد مشتری جدید مدیریت باید تصمیم گیری کند. برای هر سفارش یک صورتحساب توسط واحد حسابداری تهیه می شود.

با تعیین موردهای استفاده از سیستمها در واقع فرآیندهای عملیاتی یک سیستم مشخص می شود. لازمه شناخت دقیق نیازمندیها، درک قلمرو و محیط عملیاتی سیستم مورد نظر است. عملیات را می توان در قالب سناریو یا در واقع داستان و فیلمنامه چگونگی استفاده از سیستمها مشخص نمود

اسناد مربوطه	تحلیل و طراحی شیء گرا	نتیجه
مورد استفاده	تحلیل نیازمندها	فرایندها کدامند؟

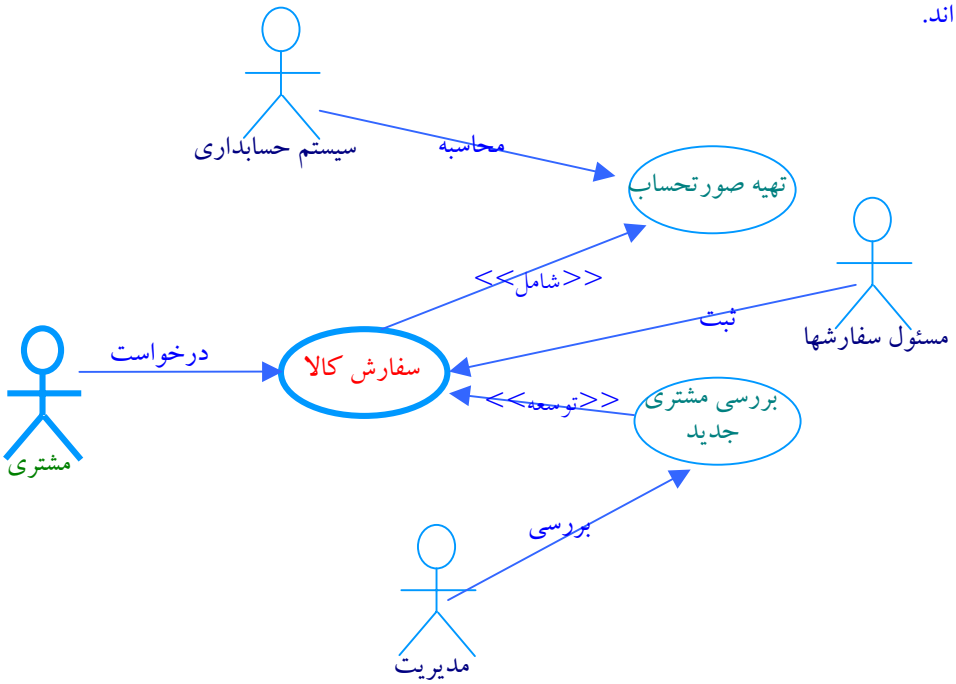
برای تعیین موردهای استفاده باید استفاده کننده ها را مشخص نمود. در واقع هدف استفاده کننده ها هستند و سیستم باید به آنها سرویس دهد. استفاده کننده ها را در اصطلاح بازیگر[۳] می نامند. در ارتباط با سیستم سفارش بازیگر اصلی سفارش دهنده کالا و زیر سیستمهای حسابداری و انبار بازیگرهای فرعی هستند. در واقع این دو زیر سیستم در ارتباط با تکمیل نمودن سفارش هستند.

پس از تعیین هر مورد استفاده باید بخشی از عملیات آنرا که قابل تفکیک و استفاده مجدد در موردهای دیگر است به عنوان مورد استفاده مجزا جدا نمود. برای نمونه در مورد سفارش کالا، می توان تولید صورتحساب را به عنوان یک مورد الحاقی جدا نمود. همانگونه که در بالا توضیح داده شده در

---

[۳] Actor

صورت نبودن سابقه مشتری در پرونده مشتری ها ، عملکرد مورد سفارش توسعه داده شده و مورد دیگری تحت عنوان "بررسی مشتری جدید" مطرح می شود. به این ترتیب با استفاده از دیاگرامی موسوم به دیاگرام مورد استفاده می توان یک مورد استفاده و عناصر مربوط به آنرا بطور خلاصه مشخص نمود. برای نمونه در شکل ۱-۱ مورد استفاده سفارش کالا و عناصر مربوط به آن مشخص شده اند.



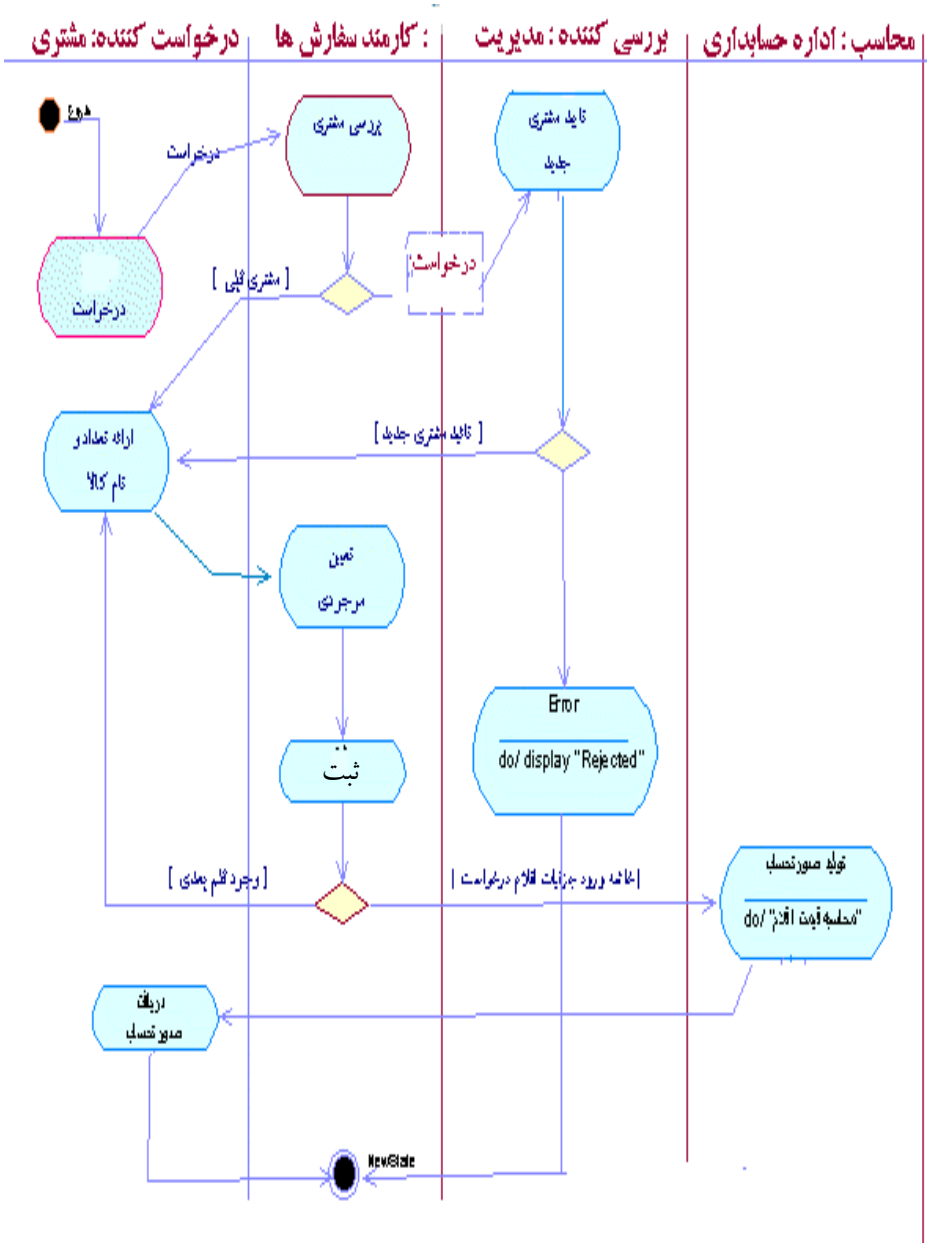
شکل ۱-۱ : مدل دیاگرام استفاده برای مورد سفارش کالا

عملکرد مورد استفاده را می توان توسط فیلمنامه یا در اصطلاح سناریوی برقراری ارتباط بین کاربر و سیستم مشخص نمود. در شکل ۱-۲ از دیاگرام گردش کار برای نمایش چگونگی مورد سفارش کالا استفاده شده است. دیاگرام فعالیت[٤] ابزاری مشابه با فلوچارت برای نمایش عملیات است. از دیاگرام فعالیت برای نمایش چگونگی گردش کار نیز می توان استفاده کرد . دیاگرام گردش کار[٥] ابزاری برای نمایش چگونگی همکاری واحدهای عملیاتی جهت انجام امور محوله می باشد. برای نمونه در شکل ۱-۲ برای ثبت سفارش مشتری مشخص شده است که درخواست از مشتری به مسئول سفارشها داده شده، از مسئول سفارشها به اداره حسابداری و نهایتا" از حسابداری به مشتری صورتحساب داده شده است.

---

[٤] Activity Diagram

[٥] WorkFlow



شکل ۱-۲ : دیاگرام فعالیت برای ثبت سفارشها

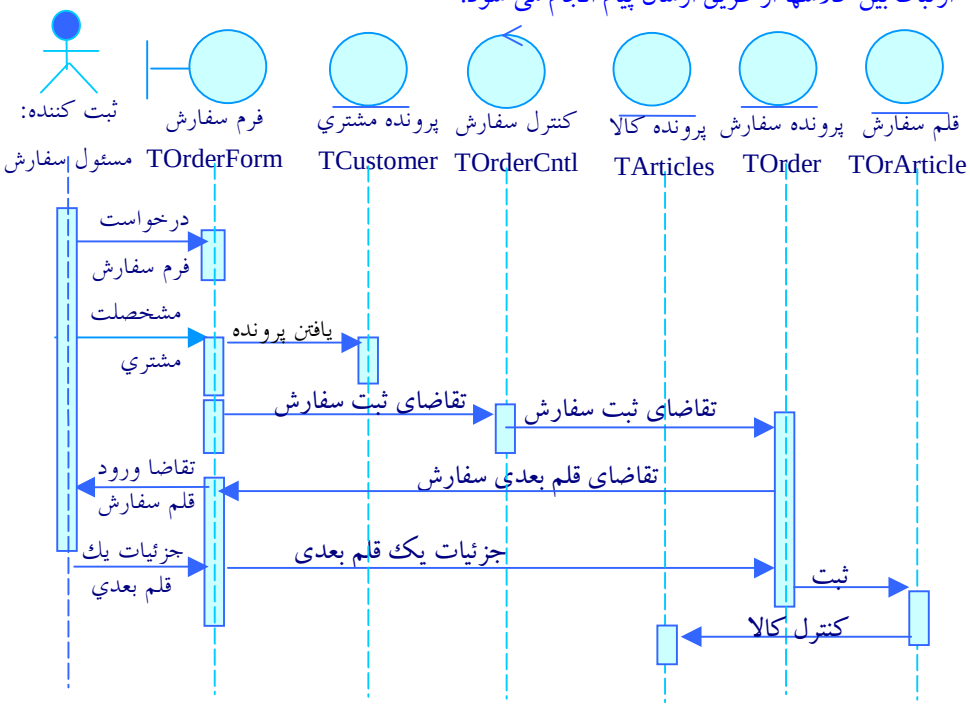
پس از تعیین موردهای استفاده ، یا در واقع سرویس های سیستم باید سرویس دهنده ها را مشخص نمود. در دیدگاه شی ءگرا ، مسئولیتها به اشیاء تفویذ می شوند. لذا، بر اساس دسته بندی مسئولیتها می توان اشیا، را مشخص نمود.

مشتری	سفارش	صورتحساب

اشیاء در ارتباط با یکدیگر عملیات سیستم را به انجام می رسانند. لذا، باید راه ارتباطی بین اشیاء را مشخص کرد. مدل ارتباطی کلاس اشیاء را در اصطلاح مدل ادراکی[۶] می نامند.

اسناد مربوطه	تحلیل و طراحی شیء گرا	نتیجه
Use case	تحلیل نیازمندها	فرایندها کدامند؟
مدل ادراکی	تحلیل قلمرو و محیط	سرویس دهندگان کدامند ؟

قبل از تعیین مدل ارتباطی اشیاء باید چگونگی همکاری بین اشیاء را مشخص کرد. همکاری بین اشیاء را در قالب دیاگرامهایی موسوم به دیاگرام محاوره[۷] مشخص می نمایند. دیاگرامهای محاوره شاخص چگونگی برقراری ارتباط در بین اشیاء در جهت رسیدن به اهداف یک مورد استفاده هستند. به عبارت دیگر هر دیاگرام محاوره عملکرد یک مورد استفاده را تصویر می کند. این عملکرد در قالب ارتباط بین کلاسها از طریق ارسال پیام انجام می شود.



شکل ۱-۳: دیاگرام اولیه توالی

---

[۶] Conceptual Model

[۷] Interaction

دیاگرامهای محاوره به دو صورت دیـاگـرام توالـی[۸] و دیـاگـرام همکـاری[۹] مشـخص مـی شـوند. دیاگرام توالی نوعی خاص ازدیاگرام محاوره است. در این نوع دیاگرام عملکرد یک مورد استفاده را به صورت تبادل دنباله ای از پیامهای متوالی بین اشیاء مشخص مـی نمـایند. در شـکل ۱-۳ دیـاگـرام اولیـه توالی برای عملیات سفارش کالا ارائه شده است. در این دیاگرام مسئول ثبت سفارش عمل ثبت سفارش را با تقاضا برای فرم سفارش آغاز می کند. شیء "فرم سفارش" وظیفـه نمـایش فـرم و دریـافت جزئیـات سفارش را بر عهده دارد. این شیء پس از نمایش فرم سفارش ، جزئیات متقاضی را دریافت و در پرونده مشتریان جستجو می نماید. در صورتیکه متقاضی دارای پرونده باشد تقاضای ثبت سـفارش را بـه شـیء کنترل سفارش می دهد. اصولا" برای هر مورد استفاده باید یک مرکز کنـترل در نظـر گرفـت. ایـن نـوع مراکز امکان هر گونه تصمیم گیری و کنترل بر عملیات یک مـورد اسـتفاده را بـه سـادگـی فراهـم مـی نمایند.

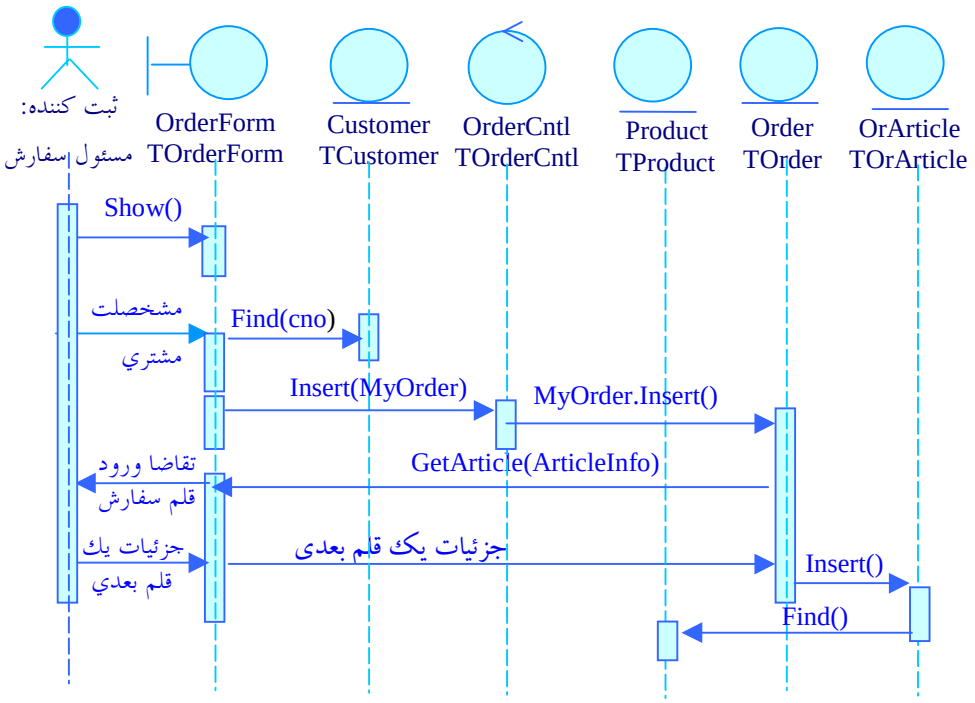
همانطور که در بالا توضیح داده شد، ایجاد دیاگرام توالی معمولا" در دو مرحله انجام می شود. در مرحله دوم تقریبا" به نوعی برنامه نویسی انجام می شود. برای نمونه بر اساس دیاگرام توالی ارائه شده در شکل ۱-۳ ، دیاگرام توالی دوم برای مورد سفارش کالا در شکل ۱-٤ ارائه شده است. در این دیـاگـرام توالی بر روی خطوط ارتباطی اسامی پیامها دقیقا"مشخص شده است. به این ترتیب در اولین گـام برنامـه به صورت دنباله ای از پیامها که در بین اشیاء مبادله می شوند مشخص شده است.. باید نوجـه نمـود کـه این پیامها معمولا" به صورت جملات فراخوانی توابع یـا در اصطـلاح متدهـای اشـیاء پیـاده سـاز ی مـی شوند. دیاگرام توالی ارائه شده در شکل ۱-٤ را می توان در قالب دنباله ای از جمله هـای فراخوانـی بـه شکل زیر مشخص نمود:

```
OrderForm.Show()
    Customer.Find(cno)
    OrderCntl.Insert(MyOrder)
        Order.Insert()
            OrArticle.insert()
                Article.find()
```

---

[۸] Sequence Diagram

[۹] Collaboration Diagram



شکل ۱-٤ : دیاگرام توالی برای پیاده سازی مورد سفارش کالا

در این مرحله باید چگونگی برقراری ارتباط در بین کلاسها را مشخص نمود. برای نمونه اگـر بـه جملات فراخوانی که در بالا ارائه شد توجه نمائید، مشـاهده مـی کنیـد کـه از داخـل تـابع Insert درون کلاس TOrAticle متـد FInd از کـلاس TArticles جـهت یـافتن مـیزان موجـودی قلـم کـالای مـورد سفارش فراخوانی می شود. بنابراین باید چگونگی برقراریارتباط بین کلاسـها را نـیز در مرحلـه طراحـی مشخص و پیاده سازی نمود. مدل ارتباطی کلاسها را مدل مفهومی[۱۰] نیز می نامند.

ارتباط بین کلاسها را در اصطلاح اجتماع بین کلاسها می نامند. با در نظر گرفتـن دیـاگـرام توالـی ارائه شده در شکل ۱-٤ نیز می توان اجتماع[۱۱] کلاسها را مشخص نمود. در شکل ۱-۵ کلاس Order یـا سفارش شامل قلم کالا است. به عبارت دیگر درون یک برگ سفارش تعداد یک یا چند قلم کالا قـرار دارد. به این نوع اجتماع کلاسها در اصطلاح تجمع[۱۲] گفته می شود. رابطه تجمع را بـه صـورت رابطـه "بخشی است از" نیز می توان خواند. بخاطر اهمیت این نوع رابطه در بین کلاسها علامـت لـوزی بـه آن

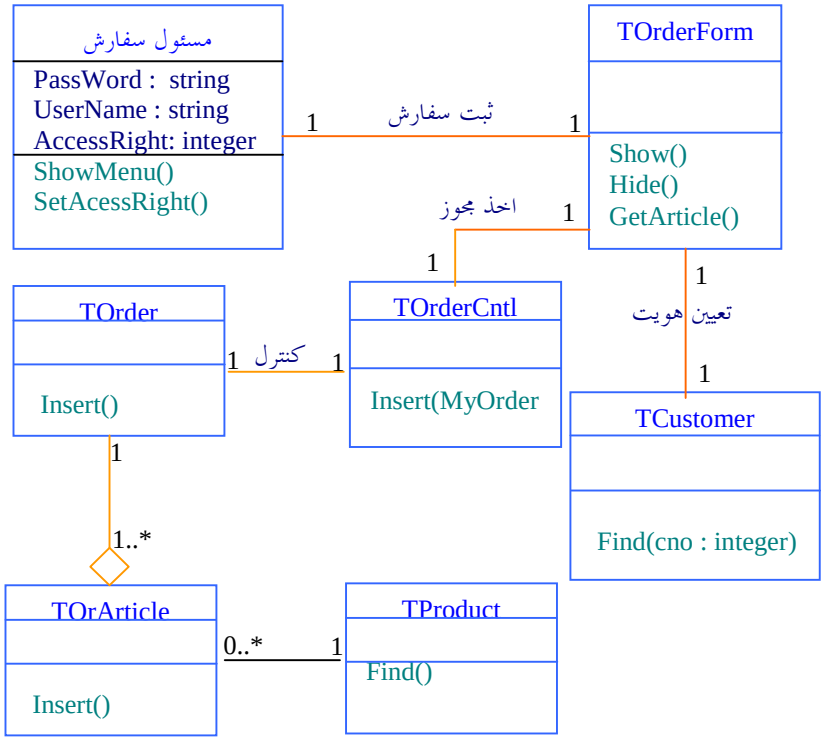
---

[۱۰] Conceptual Model

[۱۱] Association

[۱۲] Aggregation

تخصیص داده شده است.



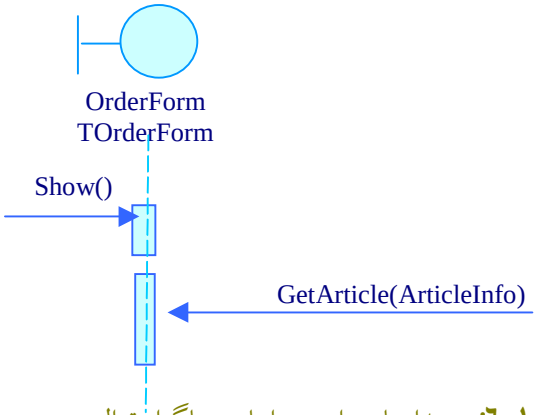
شکل ۱-۵ : مدل ارتباطی کلاسها برای سفارش کالا

با تعیین مدل ادراکی فضای دانش در ازتباط با سیستم مورد نظر تصویر می شود. در واقع با تعیین اشیاء مشخص می شود که چه عواملی در انجام یک مورد استفاده از سیستم باید با یکدیگر همکاری نمایند و با تعیین مدل ارتباطی کلاس اشیاء در واقع سیستم مکانیزه سازماندهی و روابط مشخص می شود.

شعور، واکنش در مقابل رویدادها است. موجود با شعور در مقابل رویدادها واکنش می کند. واکنش در مقابل رویداد با تغییر حالت همراه است. حالت را با ویژگیها می توان مشخص نمود. لذا، در دیدگاه شیء گرا مجموعه فیلدهای یک شیء را حالت شیء نیز می نامند. رفتار را می توان با تغییر حالت در مقابل رویدادها مشخص نمود. لذا، مدل رفتاری اشیاء را به صورت شبکه ای از حالات می توان مشخص کرد. رویدادها موجب گذر بین حالات می شوند. بنابر این پس از سازماندهی کلاسها در قالب مدل ارتباطی کلاسها باید برای هر کلاس رفتار و شعور را مشخص نمود.

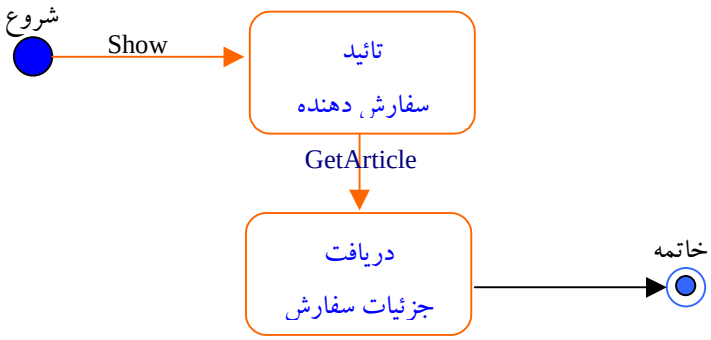
از آنجا که رویدادها عامل تغییر حالت موجودات با شعور هستند باید برای هر کلاس نیز رویدادهای وارده بر آن کلاس را مشخص نمود. رویدادهای وارده بر یک کلاس را می توان درداخل دیاگرامهای توالی متفاوت که آن کلاس در آن دخیل است جویا شد. برای نمونه با توجه به دیاگرام

توالی ارائه شده در شکل ۱-٤ رویدادهای وارده به کلاس TOrderForm مطابق شکل ۱-٦ می باشد. هر رویداد وارده موجب یک تغییر حالت کلاس می شود.



شکل ۱-٦: رویدادهای وارده بر اساس دیاگرام توالی

با توجه به شکل ۱-٦، بر طبق مورد سفارش کالا برای کلاس TorderForm دو رویداد ورودی و در نتیجه دو حالت متفاوت ایجاد می شود. بنابر این دیاگرام حالت[۱۳] برای این کلاس به صورت زیر است.



شکل ۱-۷: رفتار در قالب دیاگرام حالت

سوال اینجا است که این مدل رفتاری چگونه می تواند تبدیل به مدل برنامه شود؟ می توان هر حالت را بر مبنای دیاگرام توالی به توسط شبه دستورالعملها مشخص نمود. و بر اساس اینکه کدامیک از دو رویداد Show یا GetArticle برای کلاس TrrderForm رخ میدهد دقیقاً" واکنش این کلاس را در قالب کد برنامه پیاده سازی کرد. به این ترتیب کلاسها به صورت عوامل تصمیم گیرنده در مقابل رویدادهای جانبی خود عمل خواهند کرد. می توان رفتار یک کلاس را در قالب دیاگرام فعالیت به صورت دقیقتری مشخص نمود.

---

[۱۳] State Diagram

۱-۳ فرایند پیشنهادی نرم افزار

برای تولید نرم افزار در ادامه این کتاب روشی شناخته شده تحت عنوان فرایند پیشنهادی نرم افـزار[۱۴] استفاده شده است. در این بخش ، این فرایند پیشنهادی که توسط لارمن[۱۵] مطرح شده ، مـورد بررسـی قرار گرفته است. بر طبق پیشنهاد لارمن مراحل تولید نرم افزار به صورت زیر است:

۱- برنامه ریزی اولیه	۲- گزارش بررسی اولیه	**۳- تشریح نیازمندیها**
٤- ایجاد لغتنامه	٥- ایجاد نمونه اولیه برنامه	٦- تعریف موردهای استفاده
۷- ایجاد مدل ادراکی اولیه	۸- تعیین معماری اولیه	۹- اصلاح برنامه ریزی

شکل ۱-۸: فرایند برنامه ریزی در روش پیشنهادی

در مرحله برنامه ریزی باید زمانبندی پروژه و هزینه ها مشـخص شـود. در واقـع ایـن همـان مرحلـه امکان سنجی است. در ابتدا بطور تخمینی و بر اساس تجربیـات ممکـن اسـت یـک زمـانبندی مشـخص شود. سپس کار بررسی نیازها أغاز می شود. برای این منظور باید چارت سازمانی و حوزه وظایف افـراد مشخص شود. سپس با در نظر گرفتن حوزه وظـایف افـراد ، نیازمندیـهای عملیـاتی و کیفـی أنـها را در ارتباط با سیستم جدید مکانیزه باید جویا شد. نمونه سازی[۱۶] یک ابزار شناخت است. می تـوان بـا ایجـاد نمونه هایی از برنامه های اجرایی مشخص کرد که نیاز کاربر دقیقا" چه ی باشد و آیا سیستم مکـانیزه بـه صورت مطلوب پاسخگو است.

برای شناخت عملکرد سیستم باید استفاده کننده ها و موردهای استفاده أنها از سیسـتم را مشـخص

---

[۱۴] Recommended Process Model (RPM)

[۱۵] Craig Larman

[۱۶] pROTOTYPING

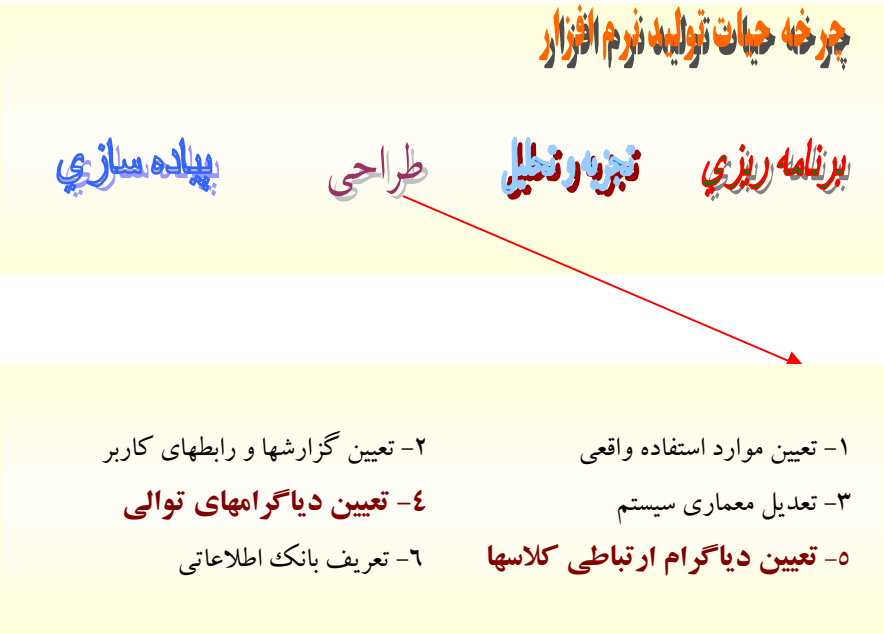
کرد. پس از تعیین موردهای استفاده می توان اشیاء و کلاسهایی که از همکاری أنها با یکدیگر عملیات سیستم انجام می شود را مشخص کرد. مدل ادراکی بیانگر ارتباط بین کلاسهایی است که در ابتدا و بدون در نظر گرفتن جزئیات سیستم مکانیزه مشخص می شود. در واقع این مدل بیانگر ساختار مفهومی سیستم است. پس از برنامه ریزی و زمانبندی پروژه و تشریح نیازمندی ها در روش پیشنهادی لارمن، مرحله تحلیل آغاز می شود.

شکل ۱-۹: فرایند آنالیز یا تجزیه و تحلیل در روش پیشنهادی لارمن

در مرحله تجزیه و تحلیل پس از تعیین موردهای استفاده ضروری و تعدیل و اصلاح دیاگرام های مورد استفاده، تاکید عمده بر تکمیل مدل ادراکی است. این مدل بیانگر رابطه بین مفاهیم یا در دیدگاه شیءگرا کلاسها است. برای متدهای هر کلاس قرارداد عملیاتی مشخص می شود. بر طبق این نوع قرار داد باید شرط قبلی[۱۷]، شرط بعدی و توصیف متد به صورت شبه دستورالعمل بیان شود. مرحله تجزیه و تحلیل در واقع مرحله شناخت نیازها است. پس از این مرحله می توان به طراحی سیستم پرداخت.

---

[۱۷] Pre



شکل ۱-۱۰ : مرحله طراحی در دیدگاه لارمن

مرحله تجزیه و تحلیل تاکید بر شناخت نیازها، مفاهیم یا کلاسها و عملیات موجود بر روی کلاسها دارد. جهت تجزیه و تحلیل و شناخت نیازمندیها ابزار زیر مورد استفاده قرار گرفتند:

۱. موردهای استفاده : جهت تعیین فرایندهای عملیاتی
۲. مدل ادراکی : جهت تعیین مفاهیم و یا به عبارت دیگر کلاسهای اصلی
۳. دیاگرام توالی سیستم : جهت تعیین رویدادها و عملیات سیستم
٤. قراردادها : جهت تعیین عملیات سیستم.

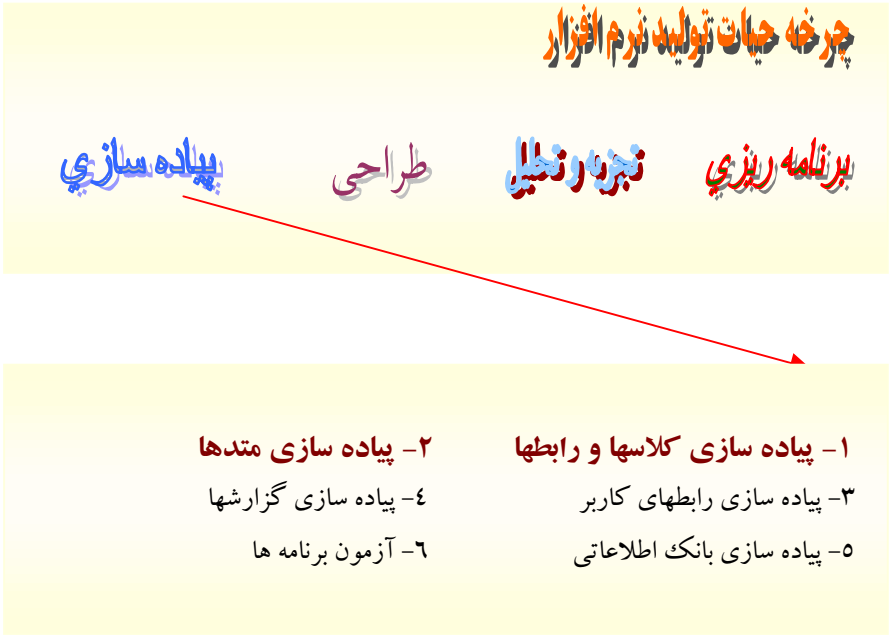
مرحله طراحی تاکید بر طرح ساختار و عملکرد منطقی سیستم دارد. منظور از طرح منطقی مشخص نمودن طرح سیستم بدون در نظر گرفتن محیط نرم افزاری، سخت افزاری و شبکه است. در این مرحله تنها منطق عملیات مشخص می شود.

در این مرحله موردهای استفاده مورد بررسی مجدد قرار گرفته ورودیها و خروجیها و فرمهای مربوطه دقیقا" طراحی و مشخص می شوند. بر این مبنا کار طراحی فیزیکی و برنامه سازی انجام می شود. مدل مفهومی تبدیل به مدل ارتباطی کلاسها می شود. کلاسها دقیقا" مشخص و دسته بندی می شوند. در هنگام دسته بندی کلاسها باید طرح سه لایه معماری نرم افزار را در نظر داشت. معماری سیستم بنا بر پیشنهاد لارمن به صورت سه لایه ایجاد می شود. طرح سه لایه به صورت زیر است:

۱- لایه رابط : شامل فرمهای ورودی / خروجی و ارتباطات با دنیای خارج
۲- لایه عملیات : شامل منطق عملیات

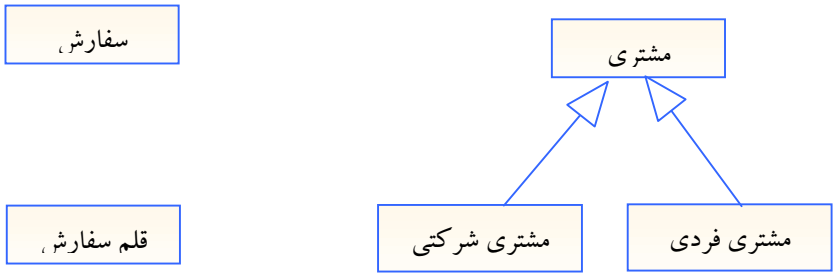
۳- لایه حافظه : شامل بانک اطلاعاتی و کلاسهای مربوطه است.

پس از مرحله طراحی منطقی طراحی فیزیکی و نهایتا" پیاده سازی سیستم انجام می شود. در ادامه مرحله پیاده سازی در فرایند پیشنهادی لارمن مشخص شده است.



شکل ۱-۱۱ : مرحله پیاده سازی در روش پیشنهادی لارمن

در دیدگاه شیء گرا عملیات از همکاری بین اشیاء به انجام می رسد. در ادامه روند تکاملی کلاسها در طی مراحل فرایند ایجاد نرم افزار مشخص شده است. برای نمونه سیستم سفارشات را در نظر بگیرید. یک مدل مفهومی ساده برای این سیستم شامل کلاس یا مفهوم سفارش ، قلم سفارش و دونوع مشتری شرکتی و فردی می تواند باشد.



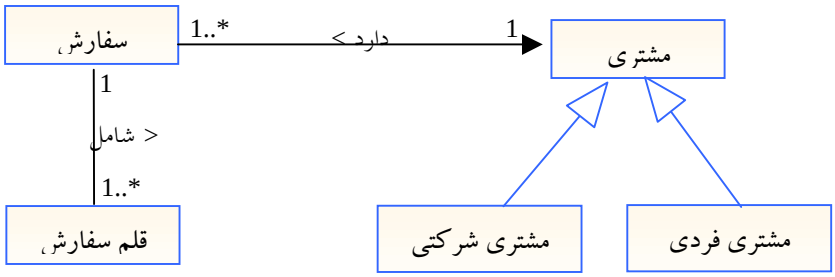
شکل ۱-۱۲: مدل اولیه ارتباط بین مفاهیم

پس از تعیین مفاهیم رابطه بین مفاهیم مشخص می شود. برای این منظور چنانچه A و B دو کلاس

باشند، رابطه بین آنها به صورت زیر است:

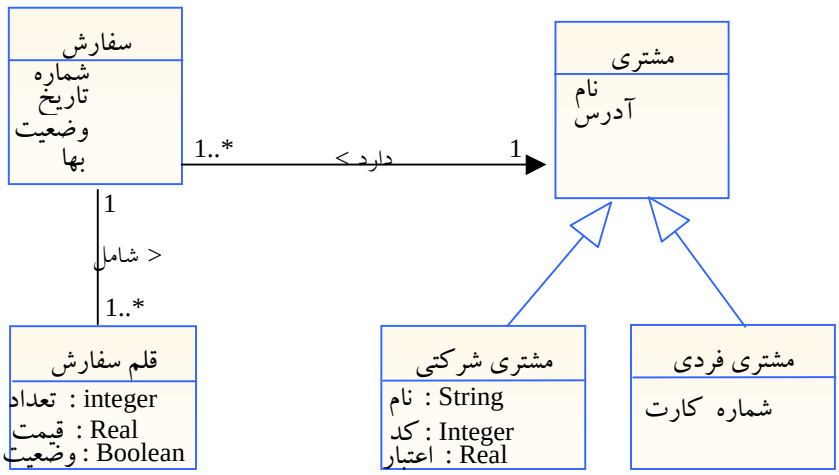
A بخشی از، شامل، در ارتباط با، مالک، نوعی از، توصیف کننده، استفاده کننده، غیره B است.

رابطه بین دو مفهوم را در اصطلاح اجتماع[۱۸] آنها می نامند. به این ترتیب با در نظر گرفتن انواع اجتماع مفاهیم مدل ارتباطی کلاسها یا در اصطلاح مدل ادراکی[۱۹] به صورت زیر ایجاد می شود.



شکل ۱-۱۳: مدل ادراکی

برای هر مفهوم باید در مرحله شناخت، ویژگیها را نیز مشخص کرد:



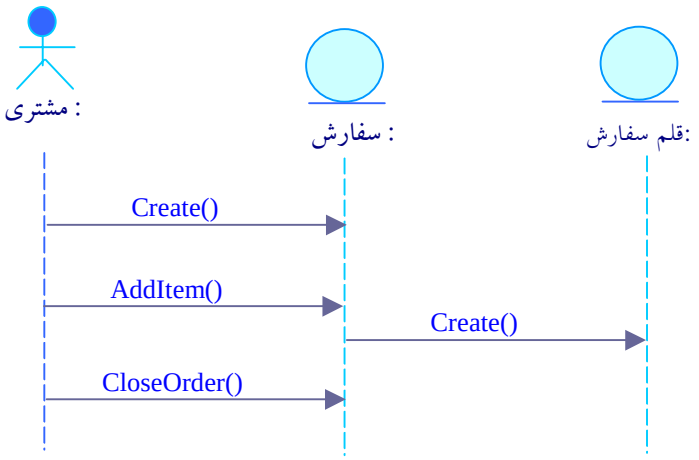
شکل ۱-۱٤: مدل ادراکی در خاتمه مرحله تجزیه و تحلیل

برای تعیین عملیات بر روی کلاسها از دیاگرامهایی موسوم به دیاگرام توالی استفاده می شود. برای نمونه در شکل ۱-۱۵ دیاگرام توالی برای عملیات سفارش کالا توسط مشتری ارائه شده است. توجه نمایید که در این دیاگرام عملیات در قالب اسامی توابع مشخص شده است.

---

[۱۸] Association

[۱۹] Conceptual Model



شکل ۱-۱۵: دیاگرام توالی

به این ترتیب می توان با در دست داشتن متدها که بر اساس دیـاگـرام توالـی بـرای مـورد اسـتفاده سفارش کالا مشخص شده، مدل ارتباطی کلاسها را تکمیل نمود. در شکل فوق ابتـدا از طریـق مشـتری متد Create() از کلاس سفارش فراخوانی می شود. پـس از ایجـاد بـرگ سـفارش، جـهت ایجـاد اقـلام سفارش متد دیگری تحت عنوان AddItem() فراخوانی می شود. که از داخل این متد در مرحله بعد تابع CloseOrder() فراخوانی می شود.

به این ترتیب پس از تکمیل مدل ارتباطی کلاسـها مـی تـوان در مرحلـه پیـاده سـازی بـه سـادگـی توصیف هر کلاس را تبدیل به کد جاوا به صورت زیر نمود:

```
public abstract class مشتری
   { private string name;
     private string address;
     public مشتری ( string n, string a) { name = 1; addres = a; }  ..}

public class مشتری شرکتی extends مشتری

public class مشتری فردی extends مشتری

public class order { private Data dateReceived;
                     private boolean isPrepaid;
                     private  سفارش قلم[] items;
                     private  int itemCount;
                     public order (boolean rePaid, string n)
                       {dateReceived=newDate(); isPrepaid=prepaid; number =n,}
                     public void addItem(int quantity, float unitPrice)
                       { orderLine = new سفارش قلم(quantity, unitPrice);
                         addItem(line); }
```

۱-۴ فرایند RUP

ابزار رَشنال رُز مبتنی بر فرایند و متدولوژی خاصی تحت عنوان RUP[۲۰] است. ایـن متدولـوژی نـیز همانند سایر متدولوژیها شامل چهار مرحله اصلی اکتساب یا شـناخت، تشـریح، ایجـاد و تحـول، مطـابق جدول شکل ۱-۱۶ است. این مراحل دقیقاً توضیح داده خواهند شد.

RUP		شناخت[۲۱]	تشریح[۲۲]	ایجاد[۲۳]			تحول[۲۴]
تلاش٪		٪۵	٪۲۰	٪۶۵			٪۱۰
مدت٪		٪۱۰	٪۳۰	٪۵۰			٪۱۰
آبشاری		امکان سنجی و تعیین نیاز	طراحی منطقی	طراحی فیزیکی	پیاده سازی و آزمون پیمانه	ترکیب پیمانه ها وآزمون	
	تلاش	٪۱۵-۲	٪۱۷	٪۶۴-۵۲	٪۳۱-۱۹	٪۲۰-۰	
	مدت٪	٪۲۴-۱۶	٪۲۸-۲۴	٪۵۶-۴۰	٪۳۲-۲۰	٪۲۰-۰	

شکل ۱-۱۶ : مقایسه مراحل و درصدهای تلاش و زمان در روشهای آبشاری و RUP

هر یک از مراحل چرخه حیات و فرایند نرم افزار در RUP از تعدادی فعالیت به شرح زیر تشـکیل شـده است:

۱- مهندسی امور : درک نیازهای کاری و حرفه ای متقاضی
۲- نیازها : تبدیل نیازهای کاری به رفتار و عملکرد سیستم مکانیزه مورد نیاز و مطلوب متقاضی
۳- تحلیل و طراحی : تبدیل نیازها به معماری نرم افزار
۴- پیاده سازی : ایجاد نرم افزار بر اساس معماری مطرح شده و مطابقت با رفتار و عملکرد مـورد نظر متقاضی
۵- آزمون : حصول اطمینان از پاسخگویی نرم افزار به نیازها و رفتار و عملکرد مطرح شده
۶- کنترل تغییرات : کنترل تغییرات و ردیابی ویرایشهای مختلف نرم افزار
۷- کنترل پروژه : کنترل زمانبندی و منابع تخصیصی به پروژه

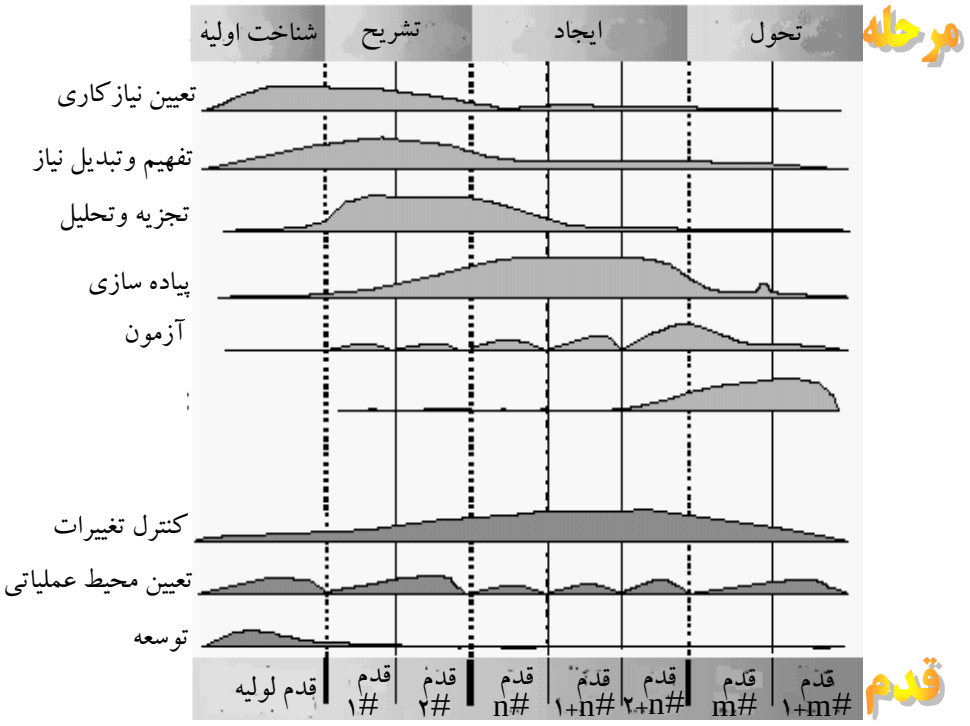
---

[۲۰] Rational Unified Process
[۲۱] Inception
[۲۲] Elaboration
[۲۳] Construction
[۲۴] Transition

۸- محیط عملیاتی : تعیین محیط توسعه نرم افرم افزار

۹- گسترش : کلیه عملیات لازم برای خاتمه فعالیتها

در شکل ۱-۱۷ نشان داده شده که در هر مرحله کلیه فعالیتهای فوق الذکر تکرار می شوند. هر مرحله از یک یا چند قدم تشکیل شده است. قدمهایی که ممکن است تکرار شوند. لذا، در این شکل پس از قدم ۲ ، قدم شماره n ظاهر شده است.



شکل ۱-۱۷ : شمای کلی فرایند یکسان شده رشنال، RUP

برای انجام یک پروژه بر طبق متدولوژی RUP ، باید هدف مهندس نرم افـزار در هر قدم از چرخه حیات، ایجاد و ارائه یک نمونـه از نـرم افـزار عملیـاتی بـرای کلیـه کاربرهای سیستم باشد. باید نرم افزار از دیدگاه کاربر مفهوم و قابل قبـول باشـد. لـذا، اعضا' تیم باید بطور همزمان با یکدیگر فعالیت نمایند. هر قدم باید در بـرگـیرنده کلیـه زیر سیستمها باشد. در هر قدم باید کلیه فعالیتهای پروژه را انجـام داد. هـر قـدم شـامل شناخت نیازهای کاری افراد باید باشد. شناخت نیازها مسلما" فعالیت تحلیـل نیازهـا را برای درک نیازها و بر این روال سایر فعالیتهای بعدی را به دنبال خواهد داشت.

نمی توان بطور دقیق زمان هر قدم را برای زمانبندی پروژه مشخص کرد. بهتر است کـه ابتـدا یـک قدم را به انجام برسانید. اگر زمان لازم جهت انجام این قدم برای نمونه دو هفته و تعداد قدمـهای پـروژه در مجموع ۳۰ قدم باشد، آنگاه بطور متوسط شصت هفته جهت اجرا پروژه زمان لازم خواهد بود. البتـه باید قبل از شروع هر قدم زمان را مشخص کرد. لذا، گام اول ممکن است دقیق نباشد. در ادامه بـه ذکـر مراحل پرداخته خواهد شد:

۱-۴-۱ مرحله شناخت اولیه

این مرحله شامل تعیین اهداف، امکان سنجی و زمانبندی پروژه است. مهمترین نکته در این مرحله شناسایی و تعیین اعضا' تیم است. همچنین در این مرحله اطلاعات ذیل گردآوری می شود:

- لیستی از نیازهای عملیاتی و کیفی
- معماری نرم افزار
- توصیف هدف
- برنامه ریزی اولیه پروژه
- موضوع پروژه

در خاتمه این مرحله باید اعضا' تیم مهندسی نرم افزار و کاربرهای سیستم مکانیزه مـورد نظـر، در مـوارد زیر به توافق برسند:

۱- نیازهای کاری افراد چیست و چه امکاناتی این نیازها را بر آورده می کند.

۲- زمانبندی اولیه پروژه

۳- معماری اولیه نرم افزار

۱-۴-۲ مرحله تشریح

در طی این مرحله باید کاربرها به مرور توسعه و سیر تکاملی نرم افزار را مشاهده کنند. بو هر قـدم جهت تکمیل و تایید بدنه ایجاد شده نرم افزار در مرحله قبلی باشـد. در هـر قـدم بـاید بتـوان آزمونـهای جدیدی برای تایید صحت عملکرد ایجاد کرد. باید، کاربر اطمینان حـاصل کنـد کـه زمـانبندی صحیـح است و پروژه بر طبق روال به خاتمه می رسد. لذا، مهندسهای نرم افزار می بایست در طی این مرحلـه بـا شناخت بیشتری که از سیستم پیدا کرده انـد مبـادرت بـه تصحیـح برنامـه ریـزی پـروژه نمـایند. در طـی قدمهای این مرحله اطلاعات بیشتری درارتباط با موارد ذیل گردآوری می شوند:

- تهیه شرحی کامل از مساله
- معماری بنیانی نرم افزار

- برنامه ریزی دقیق مراحل بعدی
- تعدیل فرایند و تیم نرم افزاری
- جلوگیری از عوامل ریسک و عدم موفقیت در اجرا به موقع پروژه

در خاتمه این مرحله باید اعضا' تیم مهندسی نرم افزار و کاربرهای سیستم مکانیزه مورد نظر، در موارد زیر به توافق برسند:

۱. موردهای استفاده از سیستم مطلوب مکانیزه بر طبق نیازهای کاری کاربرها مشخص شده اند،
۲. معماری انتخابی برای پیاده سازی نرم افزار مناسب است،
۳. عوامل اصلی در تعویق و به مخازره انداختن پروژه مشخص شده اند،
٤. برنامه ریزی پروژه منطقی و بر طبق زمانبندی به اجرا در خواهد آمد،

۱-۴-۳ مرحله ایجاد

قدمهای این مرحله تفاوت زیادی با مرحله قبل ندارند در هر تکرار از این مرحله بازهم سیر تکاملی در جهت تولید نرم افزار پیشرفت نموده ، ویژگیهای بیشتری به نرم افزار افزوده می شود. ویژگیها باید مورد تایید کاربر نیز قرار گیرد. این نکته اصلی در طراحی متمرکز بر خواسته های کاربر است. موردهای استفاده به مرور تکمیل و به نرم افزار اضافه می شوند. باید هر چه سریعتر کاربر بتواند از نرم افزار استفاده کند و عملکرد آنرا مشاهده نماید. بنابراین در حالیکه شما در حال تکمیل موردهای استفاده دیگر هستید ، کاربر در عمل موردهای تکمیل شده را استفاده می کند. در طی قدمهای این مرحله موارد ذیل تکمیل می شوند:

- سیستم کامل نرم افزاری
- تایید آزمونهای نرم افزار
- منوال کاربر

در این مرحله در اصطلاح نسخه بتا[۲۵] از نرم افزارآماده می شود. با خاتمه یافتن این مرحله، مرحله انتقال شروع می شود. در خاتمه این مرحله باید اعضا' تیم مهندسی نرم افزار و کاربرهای سیستم مکانیزه مورد نظر، در موارد زیر به توافق برسند:

۱. محصول نرم افزاری قابل استفاده است،
۲. محصول نرم افزاری ارزشمند است،
۳. هر دو طرف آمادگی برای انتقال نرم افزار را دارند.

---

[۲۵] Beta release

۱-۴-۴ مرحله تحول

در طی این مرحله نرم افزار بطور عملیاتی در اختیار کاربر قرار گرفته و کاربر در عمل آنرا استفاده می کند. لذا، در این مرحله معمولاً" برای توسعه و بهبود نرم افزار کاربر و منهندسهای نرم افزار با یکدیگر ملاقات می کنند. البته هدف در این گام بازهم رسیدن به اهداف تعیین شده در مرحله شناخت باشد. . در طی قدمهای این مرحله موارد ذیل تکمیل می شوند:

- سیستم کامل نرم افزاری
- تایید آزمونهای نرم افزار
- منوال کاربر

در خاتمه این مرحله باید اعضا' تیم مهندسی نرم افزار و کاربرهای سیستم مکانیزه مورد نظر، در موارد زیر به توافق برسند:

۱. پروژه به اهداف خود که در مرحله شناخت تعیین شده بودند، رسیده است،
۲. کاربرها راضی هستند.

۱-۴-۵ تحلیل متدولوژی

در دیدگاه شیء گرا عملیات در بین اشیاء تقسیم بندی می شود. لذا، در این دیدگاه تاکید بسیاری بر شناخت اشیاء تشکیل دهنده سیستمها به عنوان واحدهای عملیاتی وجود دارد. اما، از انجا که در سیستمها هدف عملیات و نه اشیاء انجام دهنده می باشد، باید جهت شناخت سیستمها ابتدا عملکردها را مشخص نمود تا بر مبنای عملکردها بتوان اشیاء در گیر در عملیات را مشخص کرد. بنابر این بازهم مطابق دیدگاه ساختیافته در اولین گام باید مشخص نمود که **فرایندهای کاری در سیستم کنونی به چه صورت هستند.**

در مرحله ۱ از چرخه حیات ، باید مشخص نمود، لازمه بقا و برقراری امور جاری یک واحد کاری مثل یک سازمان یا شرکت ، چه فرایندهای عملیاتی می باشد. برای نمونه یک واحد تولیدی برای انجام امور خود باید مواد اولیه تهیه کند ، محصول تولید کند ، به فروش برساند و حقوق کارکنان خود را بپردازد. اینها فرایندهای کاری برای گردش کار و برقراری امور و بقاء یک واحد تولیدی می باشند.

در دیدگاه شیءگرا فرایندهای کاری را در قالب افراد و استفاده های آنها از سیستم مشخص می کنند. برای این منظور باید

۱- چارت سازمانی را مشخص نمود.
۲- حوره وظایف و عملکرد افراد را مشخص نمود

۳- سرویس گیرنده ها برای هر وظیفه افراد را مشخص کرد (ابتدا خارجی ، سپس داخلی)

٤- نیازهای هر سرویس گیرنده در جهت بهبود سرویس و نیازهای هر سرویس دهنده در جهت بهبودشرایط برای ارائه سرویس کاری را جویا شد.

٥- در مجموع از همکاری بین افراد سرویس بهینه برای استفاده کننده ها یا سرویس گیرنده ها مشخص می شود.

٦- جهت شناخت سرویسها یا موردهای استفاده ، نیازهای افراد باید در حوزه وظایف آنها مطرح شود. باید پس از تعیین سرویس دهنده ها و سرویس گیرنده ها در یک نشست خواسته های سرویس گیرنده و پاسخهای سرویس دهنده را دریافت و سپس بر مبنای پاسخها نیازهای عملیاتی و کیفی سرویس گیرنده ها را تعیین نمود.

همانگونه که در شکل ۱-۱۷ مشاهده نمودید، کاربرنامه سازی از همان ابتدا آغاز می شود. باید با ایجاد کد اجرایی بلافاصله از ایجاد مستندات اضافی خود داری و صحت دریافت جزئیات و نیازها را آزمود و مطمئن شد که طراحی شما پاسخگوی نیازهای کاربر است.. این مشکلی است که متاسفانه روشهای ابشاری با آن مواجه هستند. و موجبات افزایش هزینه و طولانی شدن فرایند نرم افزار گردیده اند.

۱-۶ فلسفه مدل سازی

هدف از مدلسازی به تصویر در آوردن مفاهیم است. مفاهیمی که قابل تبدیل به کد برنامه باشند. ***لذا، اگر مدلی ترسیم می کنید، باید چگونگی تبدیل آن به کد برنامه را نیز مد نظر داشته باشید.*** برای ساختن مدل از فلسفه علم الوجود[۲٦] استفاده می شود. علم الوجود در فلسفه تحت عنوان تئوری وجود مطرح است . در هوش مصنوعی علم الوجود به مفهوم تئوری نمایش مدل ادراکی مطرح است . در مجموعه پایگاه دانش ، علم الوجود به عنوان مجموعه ای از مفاهیم و لغات مطرح است که برای ایجاد سیستمهای هوش مصنوعی بکار برده می شوند. علم الوجود، بیان دقیق مفاهیم و روابط است که در یک دامنه خاص وجود دارند. در علم الوجود از قید زبان آزاد و در ارتباط با مفاهیم و روابط آنها عمل می شود.

مهمترین ویژگیهای علم الوجود اشتراک و پالایش است . منظور از اشتراک توافق در درک مفاهیم است . برای نمونه مفهوم صورتحساب باید برای دو نفر که در مورد صورتحساب بحث می کنند یکی باشد. پالایش مفاهیم ما را به سمت تجرید سوق می دهد. ما مدلهایی از واقعیات را همواره در نظر

---

[۲٦] Ontology

می گیریم که این مدلها در واقع بخشی از واقعیت را در بر می گیرند. مفید بودن آنها در پـالایش و دور ریختن ویژگیهای غیر ضروری است . در واقع علم الوجود مشخص می کند که از درون یک سیستم چه باید خارج شود تا مدلی از آن ایجاد شود. البته مدل همواره در جهت درک هدفمند مفاهیم ، جنبه های معینی از یک مفهوم را تصویر می نماید. معمولاً در یک علم الوجود سه سطح اطلاعات موجود است :

۱. وابسته به مجموعه اصطلاحات : این در واقع مجموعه مفاهیم اصلی و رابطه بین آنها اسـت . در برخی موارد به آن لایه تعریف نیز اطلاق می شود.

۲. اثباتی : این بخش را لایه قضایا برای یک علم الوجود می نامند. در واقع این لایه شـامل اثبـات قضایا و شرایط موجود در ارتباط با مفاهیم و رابطه بین آنها است .

۳. عملی : این لایه را جعبه ابزار نیز می نامند. در واقع این لایه بیانگر عملیـاتی اسـت کـه بـر روی مفاهیم و روابط آنها صورت می گیرد.

بنابراین استفاده عمده یک علم الوجود ایجاد امکان جداسازی جنبه های مختلف مفاهیم است . هر جنبه ای از سیستمها توسط یک مدل تجرید می شود. مساله ارتباط مفهومی بین این مدلها است .

فصل دوم

شناخت

نیازمندیها

۲-۱ مقدمه

هدف از این فصل بیان چگونگی فرآیند تحلیل مشکلات ونیازمندیها در متدلوژی RPM می باشد. برای این منظور یک سیستم صندوق فروشگاه در نظر گرفته شده است. با توجه به فرایند پیشنهادی لاردن که در فصل ۱ توضیح داده شد و بر طبق شکل ۱-۸، تحلیل نیازها پس از یک برنامه ریزی و گزارش بررسی های اولیه انجام می شود. نیازها در دوقالب نیازهای عملیاتی و کیفی مشخص می شوند. نیازهای عملیاتی و کیفی در صفحه ۳۹ از بخش ۱ کتاب توضیح داده شده اند.

جهت بررسی و تفهیم نیازها لاردن پیشنهاد تهیه یک لغتنامه را داده است تا بتوان اصطلاحات بکار رفته را دقیقاً مشخص کرد. همچنین، می توان جهت حصول اطمینان از درک صحیح خواسته های کاربر برنامه های کوچک به صورت نمونه های سریع ایجاد نمود و با کاربر به توافق رسید. اینها نیازهای کاری کاربر و انتظارات وی از سیستم مکانیزه هستند. اما جهت درک نیاز اصلی یعنی سیستم مکانیزه، باید موردهای استفاده از سیستم مکانیزه را مشخص نمود. یعنی مشخص کرد کاربرهای سیستم چه کسانی هستند و چگونه سیستم را مورد استفاده قرار می دهند و در هر مورد چگونه سیستم را بکار می گیرند.

بطور خلاصه در متدولوژی لاردن اولین مرحله اول در فرایند توسعه نرم افزار برنامه ریزی و تشریح جزئیات سیستم است. این مرحله خود از مراحل زیر تشکیل می شود:

١. برنامه ریزی اولیه
٢. گزارش بررسی اولیه
٣. تعیین نیازمندیها
٤. ایجاد دیکشنری
٥. تولید برنامه های نمونه
٦. تعیین موردهای استفاده از سیستم
٧. ایجاد مدل ادراکی
٨. طرح اولیه معماری سیستم
٩. اصلاح برنامه ریزی

٢-٢ تعیین کلیات نیازمندیها

هدف مشخص نمودن صورت مساله است . در این راستا هنگامی که وارد یک سازمان می شوید ابتدا چارت سازمانی را مشخص می نمائید تا شناخت کلی از سیستم پیدا کرده ، مشخص شود که به چه واحدهای کاری باید رجوع نمود. سپس ، با درنظر گرفتن شرح وظایف و محدودیت افراد، برای هر واحد باید گزارش نیازمندیها شامل موارد زیر تهیه شود:

- **خلاصه عملکرد:** در حد یک پاراگراف نام پروژه مشخص می شود.
- **مشتری :** نام واحد متقاضی
- **اهداف :** به طور کلی عملیات سیستم مشخص می شود.
- **نیازهای عملیاتی** : لیستی از عملکردهای مورد نیاز سیستم مشخص می شود.
- **نیازهای کیفی :** لیستی از نیازهای کیفی سیستم تعیین می شود.

در واقع باید لیستی از عملیات و کیفیتهای مورد نظر را مشخص نمود. شاید روش قدیمی چارت عملیاتی سیستم که بر اساس شرح وظایف تهیه می شود، در کنار چارت سازمانی تا حد زیادی پاسخگوی این مرحله باشد. نیازهای عملیاتی و یا به عبارت دیگر توابع سیستم آنچه که از یک سیستم انتظار می رود را بیان می کنند. بطور خلاصه قالب بندی یک نیاز عملیاتی به صورت زیر است :

سیستم باید عمل <X> را انجام دهد

عملیات باید وابسته به نوع و واحدهای سازمانی یا واحدهای عملیاتی دسته بندی شوند. هر دسته در یک جدول مجزا قرار داده می شود. در هر ردیف این جدول یک تابع عملیاتی سیستم مورد نیاز با جزئیات زیر مشخص می شود:

الف – کد مرجع

ب – شرح نیاز در حد یک پاراگراف

ج دسته بندی یا رده نیاز به سه صورت آشکار(E) و یا پنهان (H) از دیدگاه کاربر

در جوار نیازهای عملیاتی ، نیازهای کیفی نیز مطرح می باشند. اینها بیانگر کیفیت عملیات مورد نیاز هستند. برای نمونه "سهولت استفاده" را می توان به عنوان یک نیاز کیفی نام برد. نیازهای کیفی را در حالت کلی می توان به صورت زیر دسته بندی نمود:

– سهولت در استفاده – تحمل خطا –زمان پاسخگویی

– محیط اجرائی – هزینه خرید – نوع رابط

پس از تعیین نیازهای عملی و کیفی ، باید مبادرت به تحلیل و تعیین مورد استفاده نیازهای مطرح شده نمی شودد. در واقع نیازها ، ویژگیهای سیستم نرم افزاری را مشخص می کنند. اما ، نکته اساسی عملکرد نرم افزار است که در قالب موردهای استفاده از آن مشخص می شود. پس از تعیین نیازها، گزارش اولیه شناخت به صورت ذیل مشخص می شود::

گزارش اولیه نیازمندیها

۱. خلاصه عملکرد سیستم : هدف ایجاد سیستم مکانیزه برای کنترل صندوق فروش است .

۲. مشتری : مجموعه فروشگاه های زنجیره ای عدالت

۳. اهداف : هدف اصلی تسریع عملیات وارسی مشتریان است که مسلما" سرویس دهی سریع و ارزان را برای فروشگاه به ارمغان خواهد آورد به طور خلاصه اهداف عبارتند از:

۱. وارسی سریع مشتریان

۲. تحلیل دقیق و سریع فروش

۳. کنترل اتوماتیک موجودی انبار

٤ . نیازهای عملیاتی سیستم : نیازهای عملیاتی در قالب سه جدول جداگانه برای سیستم صندوق فروشگاه مشخص می شوند. در این جداول نیازها به سه دسته E به معنای آشکار ، H به معنای پنهان و O به معنای اختیاری، از دیدگاه کاربر، رده بندی شده اند.

دسته ۱– جدول توابع اصلی :

رده	شرح تابع	مرجع
E	سیستم باید فروش جاری شامل اقلام خرید را ثبت کند	R1.1
E	سیستم باید کل فروش جاری را محاسبه کند.	R1.2

رده	شرح تابع	مرجع
H	سیستم بایدجزئیات قلم کالای مورد خرید از طریق کد کالا که توسط بارکدخوان با کد UPC خوانده مـی شـود، را دریـافت کند.	R1.3
H	سیستم باید موجودی کال ارا در هنگـام ورود هرقلـم فـروش کنترل کند	R1.4
H	سیستم باید بایگانی و ثبت سابقه فروش پس از تکمیل فروش را به انجام رساند.	R1.5
E	صندوقدار باید با تعیین یـک ID و کلمـه عبـور وارد سیسـتم شود	R1.6
H	تهیه امکانات حفظ و نگهداری دائم اطلاعات	R1.7
H	تهیه امکانات ارتباطی بین عملیاتی و بین سیستمی	R1.8
E	نمایش شرح و قیمت کالای ثبت شده	R1.8

شکل ۲-۱ جدول نیازهای پایه

دسته ۲- توابع پرداخت وجه

رده	شرح تابع	مرجع
E	کنترل پرداختهای نقدی و مبلغ بدهی	R2.1
E	کنترل پرداختها با کارت اعتباری یا از طریق کارت خوان و یـا دسـتی جزئیـات کـارت خوانـده شـده و از طریـق دسـتگاه Modem با سرویس صدور مجوز چک تمـاس حـاصل تـا از حساب مشتری چک در وجه فروشگاه صادر شود.	R2.2
E	کنترل پرداختها با چـک ، شـامل دریـافت کـارت شناسـایی و گرفتن مجوز دریافت چک از واحد مجوز فروشـگاه از طریـق Modem باید باشد.	R2.3
H	گزارش پرداختهای اعتباری برای محاسبه تراز	R2.4

شکل ۲-۲ جدول نیازهای پرداخت

٤. **ویژگیهای سیستم** : نیازهای کیفی یا در اصطلاح ویژگیهای سیستم صندوق فروشگاه در جدول زیر مشخص شده است .

جزئیات فاکتورهای کیفی

ویژگی	جزئیات و محدودیتها
زمان پاسخ	پس از خواندن کد کالا جزئیات آن حداکثر درطول ۵ ثانیه ظاهرمی شود
رابط	فرمها با شرح کامل همراه باشند. سهولت در استفاده از صفحه کلید.
تحمل خطا	پرداختهاي اعتباري در حداکثر ٢٤ ساعت بايد در سرفصل کل حسابهاي اعتباري دريافتي ثبت شود
سیستم عامل	WIN 95 ,NT

شکل ۱-۳: ویژگیهای کیفی

معمولاً، برای شناخت نیازمندیها در اولین گام عملکرد کنونی سیستم مشخص می شود تا بتوان نیازها را درک نمود. اما، متدولوژی پیشنهادی RPM پیشنهاد می کند که ابتدا نیازها را لیست نمائید و سپس به جزئیات بپردازید. توجه کنید که در واقع نیاز سیستم همان عملکرد سیستم است . نکته در واقع مرزی است که بین آنچه موجود است و آنچه که در آینده مورد نیاز خواهد بود باید قرار داده شود. جهت شناخت و واقع درک نیازهای مطرح شده اکنون باید تحلیلگر اقدام به شناخت سیستم نماید.

۲-۳ شناخت جزئیات نیازمندیها

در دیدگاه شی ء گرا جهت شناخت یک سیستم مشخص می کنند که سیستم چه استفاده و بهره ای برای محیط خارج از خود دارد. برای این منظور ابتدا باید مشخص نمود که چه کسانی و یا سازمانهایی و یا دستگاههایی از سیستم استفاده می کنند. این استفاده کنندگان را در اصطلاح بازیگر[۲۷] می نامند. برای نمونه سیستم صندوق فروشگاه که در بالا مطرح شد را در نظر بگیرید. هدف ایجاد سیستمی کامپیوتری برای کنترل عملیات فروش است . به این ترتیب که با ورود شماره هر کالا از طریق بار کد خوان ، سیستم بهای آن کالا را مشخص نماید و عملیات فروش ، حسابداری فروش کنترل موجودی انبار و سایر اعمال مربوطه به صورت اتوماتیک انجام شود.

برای شناخت جزئیات نیازها که در واقع ویژگیهای سیستم مکانیزه هستند، باید ابتدا سیستم مورد نیاز را شناخت. برای این منظور موردهای استفاده و استفاده کننده ها از سیستم مطرح می شوند. استفاده

---

[۲۷] Actor

کننده ها را در اصطلاح بازیگرهای سیستم می گویند. بنابراین موردهای استفاده[۲۸] در سه مرحلـه اصلی که در زیر توضیح داده شده مشخص می شوند:

۱. تعیین بازیگرها : شامل لیست بازیگرها، موردهای استفاده بازیگرها از سیستم و نهایتا" دیاگرام موردهای استفاده
۲. تعریف : شامل شرحی از هدف و چگونگی عملکرد هر مورد استفاده بازیگرها از سیستم
۳. تعریف جزئیات موارد استفاده : شامل سناریوی استفده از سیستم

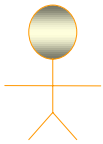
۲-۳-۱ تعیین بازیگرها

برای شناخت جزئیات نیازهای مشخص شده در جداول شکلهای ۲-۱ و ۲-۲، ابتـدا بـازیگر هـا یـا همانطور که در بالا توضیح دادیم استفاده کننده ها از سیستم را باید مشخص نمود. در ایـن مـورد چـهار بازیگر به شرح زیر مشخص می شوند

۱. صندوقدار (Cashier)
۲. مشتری (Customer)
۳. مدیر (Manager)
٤. مسئول راه اندازی سیستم کامپیوتری (System Administrator).

سپس ، طبق روش پیشنهادی باید موردهای استفاده هر یک از این بازیگرها را از سیسـتم صنـدوق فروشگاه یا در اصطلاح Post مشخص نمود.

بازیگر یک موجودیت خارجی و یا یک کاربر است که به طریقی در موردهای استفاده از سیسـتم دخیل است .یک بازیگر ممکن است استفاده کننده نباشد امـا ، در عملیـات و ارائـه و تـایید اطلاعـات مطرح باشد. بازیگرها بوسیله نقشهائی که در سیسـتم بـازی مـی کنندمعرفـی مـی شـوند مـانند مشـتری ، صندوقدار و غیره .در شکل ۲-٤ علامت بازیگر بصورت یک آدمک مشخص شده است .



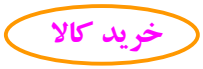
شکل ۲-٤ : نشانه بازیگر

---

[۲۸] Use Case

۲-۳-۲ تعیین کلیت موردهای استفاده ۸

موردهای استفاده ، ایزاری جهت شناخت سیستمها و تفهیم نیارها هستند. هر مورد استفاده شامل سناریو یا روایتی از فعالیتهای متوالی در جهت ارائه سرویسی خاص است. نیازها را می توان از متن موردهای استفاده استخراج نمود. برای نمونه برای مورد استفاده خرید نیاز است که سیستم قیمت کالا را مشخص کند و موجودی کالا را بروز رسانی کند. مورد استفاده را به صورت یک بیضی نشان می دهند. این بیضی یک مورد استفاده به نام خرید کالا را مشخص می کند.



شکل ۲-۵: نشانه مورداستفاده

برای تعیین موردهای استفاده دو روش رایج است .یک روش، مبتنی بر خواسته و عملکرد بازیگر است که ابتدا بازیگرهای مربوط به سیستم را تعیین و سپس برای هر بازیگر، فرآیند عملیاتی و موردهای استفاده وی از سیستم را مشخص می کنند. . روش دیگر تعیین موردهای استفاده بر پایه رُخداد است. به این ترتیب که ابتدا رُخدادهای خارجی که سیستم باید به آنها پاسخ دهد را تعیین و سپس ارتباط رُخدادها با بازیگرها بیان می شود.

روش پیشنهادی اینجانب استفاده از جداولی مطابق شکل ۲-٦ است که در اختیار افراد قرار داده می شود تا سرویسهای مورد نیاز خود و سرویس دهنده ها را مشخص کنند. سپس در یک نشست سرویس گیرنده ها در مقابل سرویس دهنده ها قرار می گیرند تا با کمک یکدیگر نیازهای عملیاتی و کیفی سیستم را مشخص نمایند.

ردیف	سرویس گیرنده	سرویس مورد نیاز	سرویس دهنده	هدف

شکل ۲-٦ : فرم تعیین مورهای استفاده و بازیگرها

نیازمندیها می توانند همرا با تحلیل موردهای استفاده استخراج شوند. برای نمونه برای مورد استفاده خرید کالا ، می توان نیازها را مشخص کرد و به جداول نیازهای عملیاتی و کیفی که در ابتدای این فصل ارائه شد افزود. مورد های استفاده نیازهای عملیاتی و نیازهای کیفی سیستم را تفهیم می کنند. نام یک مورد استفاده با یک فعل شروع می شود. موردهای استفاده بوسیله رخدادها تحریک و توسط بازیگرها ، فعال می شوند.

در ادامه برای سیستم صندوق ، مراحل تعیین و تعریف یک مورد استفاده همراه با مفاهیم مربوطه گام به گام بیان می شود :

الف – تعیین بازیگرها وموردهای استفاده

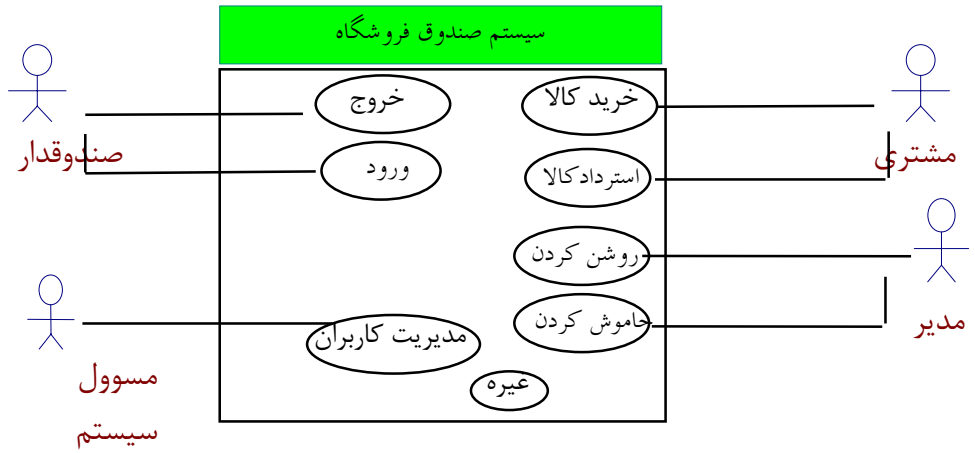
- تعیین حوزه سیستم به لحاظ سخت افزاری و نرم افزاری
- تعیین بازیگرها و رویدادهایی که آنها آغاز می کنند

رخدادها	بازیگر
خروج ورود	صندوقدار
خرید کالا – استرداد کالا	مشتري
روشن کردن – خاموش کردن	مدیر
اضافه کردن کاربران جدید	مسوول سیستم

شکل ۲–۷ جدول بازیگرهای سیستم

ب – ترسیم یک دیاگرام مورداستفاده

در این دیاگرام ملاحظه می شود که بازیگرهائی از قبیل مشتری و مدیر و غیره ،هرکدام در ارتباط با موردهای خاص خود قرار گرفته اند.



شکل ۲–۸ : دیاگرام مورد استفاده برای برنامه کاربردی صندوق

ج – نوشتن موردهای استفاده در فرمت سطح بالا

در مثال زیر مورد استفاده ای بنام خرید اقلام مشخص شده است .بازیگرها در اینجا صندوقدار و مشتری می باشند واین مورد استفاده اصلی یاPrimary می باشد.

مورد استفاده	خرید اقلام
بازیگر ها	مشتری ، صندوقدار

نوع	اصلی
	مشتری به صندوق نزدیک می شود و تقاضای خرید اجناس را با قرار دادن اقلام خرید در مقابل صندوقدار اظهار می دارد. صندوقدار اجناس مورد خرید را ثبت و یک فاکتور تهیه می کند.با اتمام کار، مشتری صندوق را ترک می کند

شکل ۲-۹ دیاگرام کلی مورد استفاده برای خرید اقلام کالا

توجه نمائید که جهت تفهیم نیازها باید موردهای استفاده از سیستم یا در اصطلاح Use Case ها مشخص شوند. سپس با ارجاع به متن هر مورد استفاده ، نیازها استخراج می شوند. در واقع با استخراج نیازها هدفهای سیستم مشخص می شود. توجه داشته باشید که ابتدا نیازها با در واقع توابع سیستم در جداول شکل ۲-۱ و شکل ۲-۲ مشخص شده اند.

۲-۳-۳ تعیین جزئیات موردهای استفاده

بعضی از موردهای استفاده مانند خرید اجناس (Buy Items) و یا فاکتور اجناس خریده شده، اصلی و مهم به شمار می روند . برای این موردهای استفاده باید نسبت به سایرین اولویت قائل شد. عملیات هر مورد استفاده به صورت لیستی از رویدادها و واکنش سیستم درمقابل هر رویداد مشخص می شود .در داخل یک مورد استفاده ممکن است به مورد استفاده دیگری ارجاع شود.برای نمونه در شکل زیر برای چگونگی پرداخت ،مورداستفاده جداگانه ای مشخص شده است . برای اینگونه موارد، فعالیتها به صورت مجزا در جدولی تحت عنوان زیر موارد استفاده یا Section مشخص می شود. توجه کنید که دنباله فعالیتهای مشخص شده برای یک مورد استفاده حالت مثبت را در نظر می گیرد و پاسخهای منفی و موارد استثنایی را بعداً مشخص می نمایند. همچنین موارد خلاف روال عادی مورد استفاده در بخش جداگانه ای تحت عنوان "مورد های دیگر" مشخص می شود.

خرید اجناس	مورد استفاده
مشتری ،صندوقدار	بازیگر
اصلی	نوع
ثبت یک فروش ویک پرداخت	هدف
مشتری به صندوق نزدیک می شود و تقاضای خرید اجناس را با قرار دادن اقلام خرید در مقابل صندوقدار اظهار می دارد. صندوقدار اجناس مورد خرید را ثبت و یک فاکتور تهیه می کند.با اتمام کار، مشتری صندوق را ترک می کند	شرح
R2.4 ç R2.3ç R2.1ç R1.9ç R1.7ç R1.3ç R1.2 çR1.1	نیازهای مرجع

رخدادها

واکنش سیستم	عمل بازیگر
	۱-این مورد زمانی که یک مشتری با اجناس خریداری شده به صندوق نزدیک می شود، شروع می شود
۳-بلافاصله با تعیین شدن هر کالا سیستم قیمت آن قلم شرح کالا را مشخص می نماید و اطلاعات مربوطه را برای اجرای تراکنش فروش موقتاً ثبت می کند.	۲-صندوقدار هر قلم کالای مورد خرید را ثبت می کند، اگر تعداد مورد در خواست از یک قلم کالا بیش از یک باشد،صندوقدار تعداد آنها را نیر وارد می کند.
۵-سیستم کل مبلغ فروش را محاسبه و نمایش می دهد	٤-برای تکمیل ورود کالاها ،صندوقدار با فشار دکمه جمع اقدام می کند.
	٦-صندوقدار به مشتری جمع مبلغ را اعلام می کند
	۷- مشتری نوع پرداخت را انتخاب می کند
	-اگر پول باشد: Paybycash
	-اگر کارت اعتباری باشد: Paybycredit
	-اگر چک باشد:ا Paybycheck
۸- سیستم سابقه فروش را ثبت می کند	
۹- بروزرسانی موجودی انبار	
۱۰-تهیه رسید	
۱۱-صندوقدار به مشتری رسید کالا را می دهد	
	۱۲- مشتری با کالای خود دور می شود.

موردهای دیگر	
خط ۲ .	اگر کد کالای غیرمعتبر وارد شد، اعلام خطا کن .
خط ۷.	در صورت عدم توانایی پرداخت مشتری ، تراکنش فروش حذف گردد

زیر سناریو	پرداخت نقدی	
پاسخ سیستم		**عمل بازیگر**
		۱. مشتری وجه را که معمولاً بیش از مبلغ مورد نیاز است پرداخت می کند.
۳- سیستم اضافه پول را مشخص می کند		۲- صندوقدار وجه پرداختی را وارد می کند
		٤- صندوقدار مابقی پول را تحویل می دهد.

شکل ۲-۱۰ دیاگرام مورد استفاده با جزئیات برای خرید اقلام کالا

در بالا یک توالی زمانی برای هر رویداد یا در واقع فعالیت تعیین شده است . موردهای استفاده در جهت اهداف خود مشخص می شوند و حالات استثنایی و پاسخهای منفی در آنها حتی المقدور مشخص نمی شود.

در شکل زیرروش دیگری برای تعیین اولیه مورد های استفاده مشخص شده است .در اینجا گام به گام مراحل خرید کالا آنگونه که در واقعیت رخ می دهد، مشخص شده است

سناریوی موفق اصلی

۱-خریدار با یک درخواست خرید، تلفن می زند.

۲-شرکت ، نام وآدرس وکالاهای درخواستی خریدار را می گیرد

۳-شرکت ، اطلاعات کالاها وقیمتها و تاریخ تولید را به خریدار ارائه می دهد.

۴-خریدار برای سفارش کالا،صحبت می کند

۵-شرکت اوراق سفارش به خریدار ارائه می دهد

۶-شرکت ،فاکتور به خریدار ارائه می دهد

۷-خریدار صورتحساب را می پردازد

متغیرها:

۱خریدار ممکن است ازطرق زیر خرید خود را انجام دهد :

-تلفن -فاکس -سفارش Web -مستقیم

۷. خریدار ممکن است به صورت زیر وجه را بپردازد

-پول -چک -کارت اعتباری

شکل ۲-۱۱ : یک مورد استفاده ساده توصیفی از خرید کالا

در هنگام تعیین نیازها و موردهای استفاده باید دقت نمود که:

۱. اولویتها برای کاربرها وابسته به نوع کارشان متفاوت است .

۲. کاربرها ممکن است معلم و یا بیان کننده های خوبی نبوده ، نتوانند آنچه را که می خواهند به درستی و بطور کامل بیان کنند.

۳. عدم آشنایی و شناخت قبلی مهندس نرم افزار از عملکرد سیستم خود می تواند مشکل ساز باشد. لذا، در گام اول فقط نیازها حعی المقدور لیست می شود. تا شناخت در حد نیاز یک مهندس کامپیوتر برای مستند کردن نیازمندیها حاصل گردد.

٤. معمولاً، تیم نرم افزاری در ارتباط با مدیران بخشها و واحدها ملاقات می کنند. مدیران دیدگاه دقیقی از عملکردها نداشته و اغلب خودشان هم حتی نمی دانند که نمی دانند!

برای نمونه یک سیستم کتابخانه دانشگاه را در نظر بگیرید. در ابتدا بر طبق معمول صورت مساله در قالب یک تقاضای مبهم مطرح می شود:

> – **شرح اولیه مساله** : هدف ایجاد یک سیستم ایده آل برای کتابخانه دانشگاه است . در حال حاضر یک برنامه ساده کامپیوتری جهت گردآوری اطلاعات کتب در کتابخانه فعال است .

اکنون باید صورت مساله را کمی مشخصتر نمود. برای همین منظور تحلیلگر به مطالعه سیستم و گردآوری ویژگیهای سیستم کامپیوتری می پردازد. شاید بهتر باشد در عوض کلمه نیاز، لفظ ویژگی را بکار ببریم .

مرحله اول تعیین نیازها با یافتن واقعیات در ارتباط با عملکرد سیستمها خاتمه می یابد. برای نمونه در ارتباط با سیستم کتابخانه واقعیات زیر مشخص می شود:

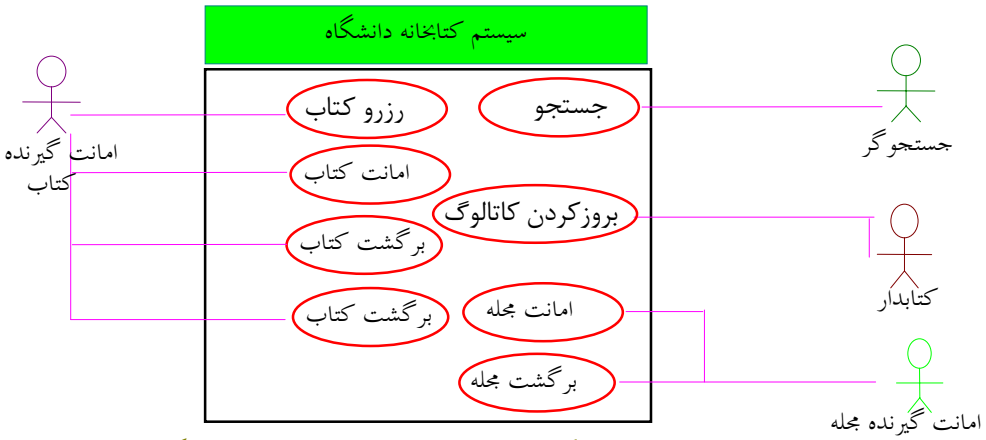
الف– کتابها و مجله ها: در کتابخانه ممکن است چندین کپی از یک کتاب موجود باشد. حداکثر برای مدت سه هفته می توان هر کتاب را به امانت گرفت . مجله را فقط کارمندان دانشگاه می توانند امانت بگیرند. عضوهای عادی یعنی دانشجویان حداکثر می توانند ٦ کتاب را در یک زمان در امانت داشته باشند. برای کارمندان حداکثر ۱۲ مورد امانت در یک زمان مجاز است . مجله ها و کتابها به طور معمول خریداری و به کتابخانه وارد می شوند. کتب قدیمی را به هر طریق ممکن از کتابخانه خارج می کنند. مجله های یک سال را در پایان سال به صورت یک مجموعه صحافی می کنند.

ب– امانت : سیستم باید در صورت تاخیر در برگشت هر مورد امانت گرفته شده ، یک نامه درخواست برگشت برای امانت گیرنده ایجاد کند.

ج– سرویس کاربران : سیستم باید امکان انواع جستجو موضوعی ، جستجو با تیتر و یا با استفاده از نام مولف را فراهم آورد. همچنین باید وضعیت کتاب را از لحاظ امانت گرفتن و یا

رزرو کردن مشخص نماید. هر فردی باید بتواند در کتابخانه با استفاده از کامپیوتر جستجو نماید.

اکنون جهت درک نیازها و تعیین جزئیات مربوطه باید موردهای استفاده را مشخص کود. برای این منظور ابتدا باید بازیگرها را مشخص نمود. منظور از بازیگر نقش افراد در ارتباط بـا اسـتفاده از سیستم است . افراد یا سازمانها یا چیزهایی که در حالت کلی در خارج از سیستم قرار مـی گـیرند و با آن تماس مستقیم دارند را بازیگر می نامند. سپس برای هر بـازیگر لیسـت مـوارد اسـتفاده وی از سیستم مشخص می شود. برای نمونـه در شـکل ۲-۱۲ دیـاگـرام موردهـای اسـتفاده بـرای سیسـتم کتابخانه دانشگاه ارائه شده است :



شکل ۲-۱۲ دیاگرام موردهای استفاده از کتابخانه دانشگاه

البته در بین موارد استفاده برخی ممکن اسـت از اولویـت بیشـتری بـرای بازیگرهـا برخـوردار باشند. برای نمونه چهار مورد استفاده در بالا از اهمیت بیشتری برخوردارند. این چهار مورد عبـارتند از:

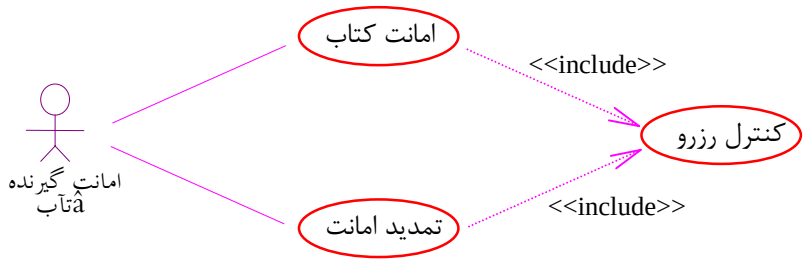
۱. امانت گرفتن یک کپی از کتاب
۲. برگشت دادن یک کپی از کتاب
۳. امانت گرفتن یک کپی از مجله
٤. برگشت دادن یک کپی ازمجله

۲-۴ موردهای استفاده به هم مرتبط

اصولاً، فرض بر این است که موردهای استفاده از طریق بانک اطلاعاتی بـا یکدیگـر مرتبـط مـی باشند و ارتباط مستقیم ندارند. علت این است که می خواهند عناصر سیستم به یکدیگر وابسته نباشند. در طرح معماری سیستمها سعی می شود که عملیات منسجم در یک واحد گردآوری شوند و واحدها حتی المقدور مستقل از یکدیگر عمل نمایند. برای نمونه واحد مالی و واحـد آمـوزش در دانشـگاه را در نظـر بگیرید. مزیت دیگر استقلال موارد استفاده از یکدیگر ، امکان ایجاد توسـعه مسـتقل موردهـای اسـتفاده است. توسعه می شود که موردهای استفاده را به صورت مجزا پس از تعیین اولویـت کـامل نمـوده و در اختیار استفاده کننده ها قرار داد.

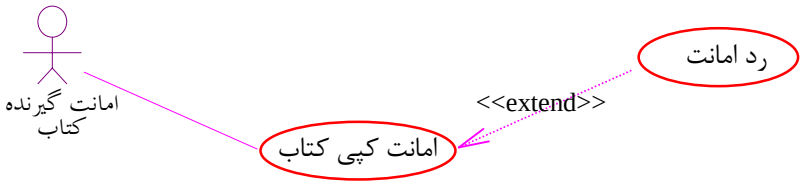
در طرح جامع سیستمها لازم است که موردهای استفاده بطـور کلـی بـا یکدیگـر مقایسـه و مـوارد مشترک و موردهای سرویس دهنده مشخص شوند. همچنین لازم است که بخشها یـا موردهـایی کـه در حالتهای خاص و تحت شرایط مشخص به یک مورد استفاده افزوده می شوند را تعیین کرد.

اصولاً، مساله استفاده مجدد در دیدگاه شی ء گرا از اهمیـت ویـژه ای برخـوردار اسـت . در ایـن راستا دو مقوله طراحی برای استفاده مجدد و طراحی با استفاده مجدد مطرح مـی باشـد. بـاید بتـوان در هنگام طراحی سیستم قطعه ها با امکان استفاده مجدد را مشخص نمود. بـرای نمونـه در شـکل ۲-۱۳ در سیستم کتابخانه دانشگاه برای دو مورد استفاده تمدید امانت و به امانت گرفتن کتاب مـی تـوان کنـترل وضعیت رزرو را به عنوان یک سناریو و قطعه مشترک مشخص نمود. مسلماً، مشترکات سیستمها قابلیت استفاده بالاتری نسبت به سایرین دارند.



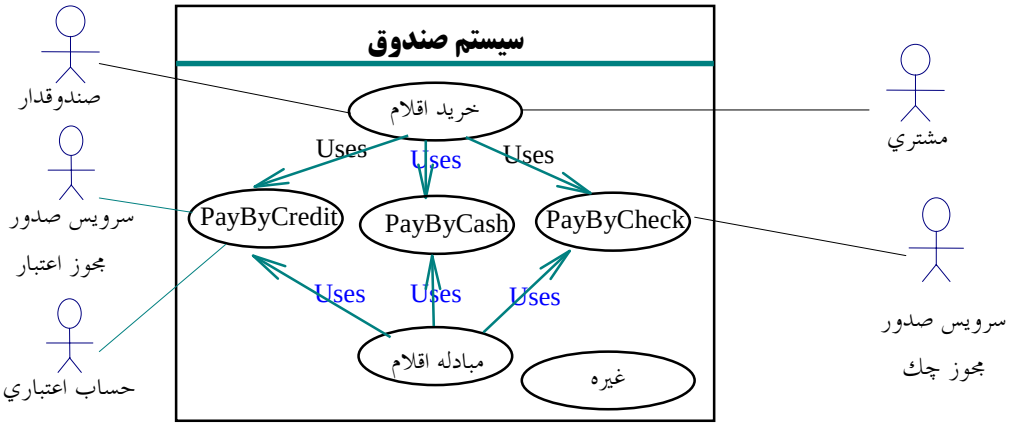
شکل ۲-۱۳: رابطه <<include>> بین موردهای استفاده

می توان رفتارهای متفاوت یک مورد استفاده که وابسته به شرایط مختلف مشاهده می شود را بـر اساس میزان اهمیت در قالب موردهای استفاده مجزا مشخص نمود. برای نمونه مـی تـوان وضعیـت غـیر عادی که کاربر مجاز به امانت گرفتن کتاب نیست را به صورت یک مورد جداگانه مشخص نمود.



شکل ۲-۱٤: رابطه <<extend>> بین موردهای استفاده

موردهای استفاده ممکن است در ارتباط نزدیک مفهومی با یکدیگر قرار داشته باشند . برای نمونه سه موردپرداخت نقدی ، با چک و توسط کارت اعتباری را در نظر بگیرید. این سه در ارتباط نزدیک مفهومی هستند. باید این ارتباط را مشخص نمود. برای مشخص نمودن ارتباط بین موردهای استفاده ، از لغت کلیدی Uses در RPM استفاده می شود. به این ترتیب پس از مشخص شدن روابط بین موردهای استفاده دیاگرام موردهای استفاده به صورتی کاملتر در زیر مشخص می شود:



شکل ۲-۱۵ : دیاگرام موردهای استفاده برای سناریوی پرداخت مبلغ خرید

در ارتباط با یک سیستم بزرگ مسلما" زیر سیستمها عاملی برای دسته بندی سناریوها هستند. اما، اینگونه دسته بندیها نهایتا" نباید عاملی برای تشخیص معماری باشد. هدف در روش شیء گرا شناخت مبتنی بر دسته بندی وظایف نیست. در دیدگاه شی ء گرا عامل تجزیه کننده سیستمها اشیاء هستند. لذا، اشیاء را باید دسته بندی نمود تا معماری مشخص شود. UML علائم خاصی برای نمایش روابط موردهای استفاده دارد.با استفاده از رابطه uses، موردهای استفاده را سازماندهی می کند.

چه موقع موردهای استفاده مجزا ایجاد می شوند؟ فعالیتهایی که در Use case چندین بار تعریف می شوند: فعالیتها پیچیده و طولانی هستندومجزا شدن آنها عاملی برای ه درک واحدهای عملیاتی می شود. برای مثال در برنامه کاربردی صندوق فروشگاه ، روشهای متفاوت پرداخت به

موردهای استفاده مجزا به نـام هـای payByCash ، payByCredit و payByCheck تفکیـک شـده است. پرداخت قیمت کالاهای می تواند از طریق وجه نقد،کارت اعتباری وچک باشد. برای دریافت وجه بنابراین مورد استفاده جدیدی بنام ExchangeItem مطرح می شود.

زمانیکه موردهای استفاده به صورت رابطه uses مشخص می شوند باید بتوان در متن توصیفی آنها به قسمی با استفاده از جملات این رابطه را مشـخص نمـود. بـرای نمونـه در بنـد ۷ از مـورد استفاده که در شکل ۲-۷ مشخص شد این رابطه به صورت زیر مشخص شده است . توجه کنید کـه لغت کلیدی initiate بیانگر فعال نمودن یک مورد استفاده مجزا است .

a.if cash payment ,initiate payByCash
b.if credit pament ,initiate payByCredit
c.if check payment ,initiate payByCheck

در مورد Buy Items در یـک زمـان فقـط یکـی از سـه فرآینـد payByCash، payByCredit، payByCheck انتخاب می شـود. بنـابراین Buy Items بـاید بـا یـک رابطـه uses بـا آنـها ارتبـاط برقرارکند.در زیر مشروح Buy items به طور کامل آمده است .

مورد استفاده	Buy Items
بازیگرها	مشتری و صندوقدار
شرح	مشتری به صندوق نزدیک می شود و تقاضای خرید اجناس را با قرار دادن اقـلام موردخریـد در مقابل صندوقدار اظهار می دارد. صندوقدار اجناس مورد خرید را ثبت و یک فاکتورتهیـه می کند. با اتمام کار، مشتری صندوق را ترک می کند

واکنش سیستم	**شرح فعالیت**
	۱. بانزدیک شدن مشتری به صنـدوق همـراه بـا اقلام انتخاب شده ، عملیات شروع می شود.
۳. سیستم قیمت کالا را مشخص و اطلاعات کالا را به تراکنش فروش جاری به طور موقت اضافه می کند.	۲. صندوقدار هر کالا را ثبت می کند. اگر بیش از یک کالا از یک نوع باشد ،صندوقدار تعدادکالا را وارد می کند
۵.کل فروش محاسبه و نمایش داده می شود.	٤.برای تکمیل فروش کالا ،صندوقدار دکمه جمع را فشار می دهد
	٦.. صندوقدار کل فروش را به مشتری می گوید
	۷.مشتری نوع پرداخت را انتخاب می کند

a.if cash payment ,initiate payByCash

b.if credit pament ,initiate payByCredit
c.if check payment ,initiate payByCheck

۸. سابقه فروش ثبت می شود

۹. رسید صادر می شود.

در زیر نوع پرداخت PayByCredit مشخص شده است.

۱.این مور با اعلان مشتری برای پرداخت با استفاده از کارت اعتباری آغاز می شود.

۲.مشتری بمنظور تکمیل پرداخت اعتبار، کارت خود را بر روی دستگاه کارت خوان اعتباری می کشد.

۳.یک درخواست پرداخت اعتباری آغاز شده وآنرا به سرویس مجوز کارت اعتباری می فرستد(با استفاده ازیک مودم و متصل به ترمینال صندوق)

٤. سیگنالی مبتنی بر جواب از سرویس اجازه کارت (CAS) دریافت می شود.

٥. صندوق ،پرداخت کارت اعتباری را ثبت می کند و اطلاعات را به سیستم دریافت حسابداری می دهد.

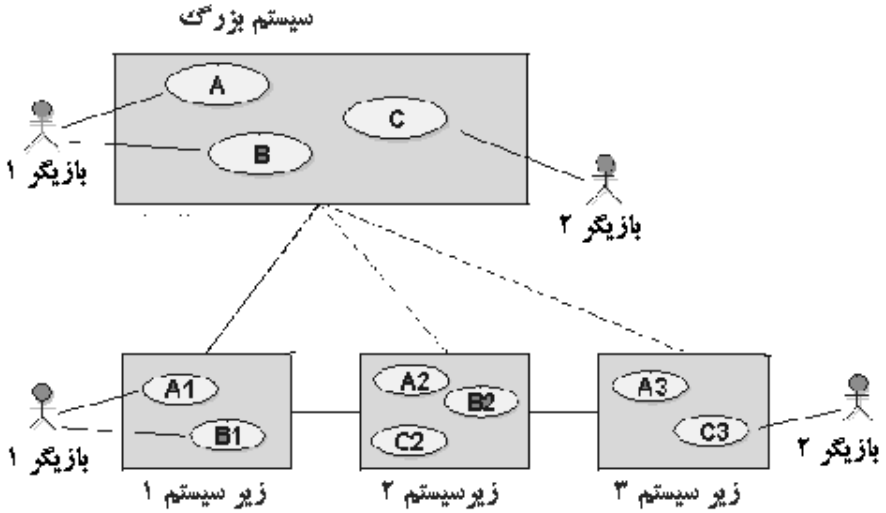
۶.پیامی مبنی بر موفقیت نمایش داده می شود.

شکل ۲-۱۷ : مورد استفاده کامل شده

۲-۷ : مساله شناخت سیستمهای بزرگ و طرح جامع

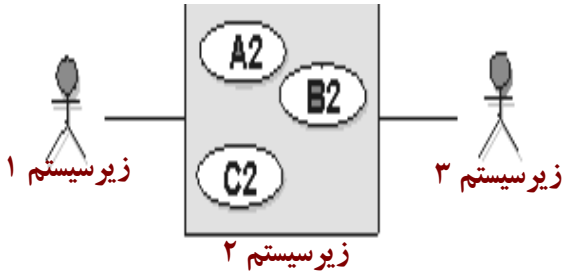
امروزه سوالی که در جوامع کامپیوتری در داخل مملکت مطرح است، پاسخگویی متدولـوژیـهای شیء گرا در مقابل طرح جامع برای سیستمهای بزرگ است. آنچه که مسـلم اسـت بـاید بتـوان گـردش کاری در سیستم موجود را دید، روابط را مشـخص کـرد، عملیـات را تحلیـل و روابـط را بهینـه کـرد. و گردش کاری سیستم جدید را با گردش کاری سیستم قبلی مقایسه نمود. نکته دیگری که مطـرح اسـت اغلب عدم وجود سیستم دستی و نیاز به ایجاد روش دستی و بهبود روشها است.

یک روش برای شناخت سیستمهای بزرگ در شکل ۲-۱۸ مشخص شده اسـت. همانگونـه کـه در این شکل مشخص شده، موارد استفاده از یک سیستم بزرگ خـود در بیـن زیـر سیسـتمها تقسـیم شـده است. برای هر زیر سیستم یک دیاگرام استفاده مجزا ترسیم می شود.

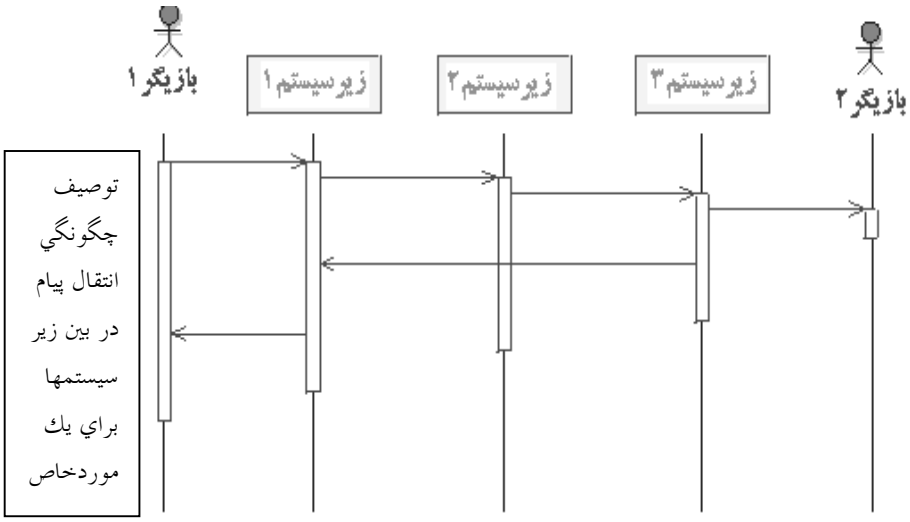


شکل ۲-۱۸ : برای سیستم بزرگ و زیر سیستمها دیاگرام جداگانه رسم می شود.

هر زیر سیستم به عنوان یک بازیگر برای زیر سیستم دیگر عمل می نماید. در اینجا مسـاله طراحـی رابط برای زیر سیستمها است. که معمولاً، این نوع رابطها توسط کلاسهای رابط مشخص می شـوند. در شکل زیر نشان داده شده که برای زیرسیستم ۲ دو زیر سیستم ۱ و ۳ به عنوان بازیگر عمل می کنند.



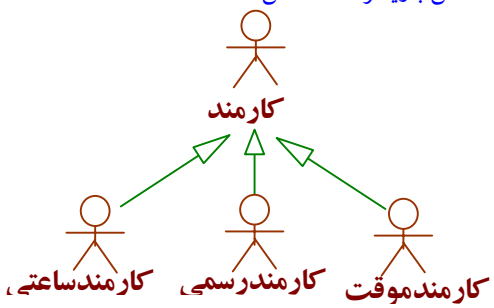
برای تشریح مورد های استفاده در سیستم اصلی می توان چگونگی برقراری ارتباط در بین زیر سیستمها را مشخص نمود. برای نمونه در شکل ۲-۱۹ برای نمایش چگونگی عملکرد مورد استفاده A چگونگی برقراری ارتباط در بین زیر سیستمهای آن مشخص شده است. در واقع خطوط افقی بیانگر بندهای سناریو برای مورد استفاده هستند. دیاگرام زیر مشخص می کند درجهت ارائه سرویس A چگونه زیر سیستمها با یکدیگر مرتبط می شوند.



شکل ۲-۱۹ : نمایش چگونگی ارسال پیام در بین زیر سیستمها جهت انجام مورد استفاده خاص

روش استاندارد برای تقسیم بندی سیستمها به زیر سیستمها استفاده از مفهومی Package است. به این ترتیب که هر زیر سیستم به عنوان یک بسته جداگانه مشخص می شود. برای یک Package در UML نشانه خاصی در نظر گرفته شده که توضیح داده خواهد شد.

تعداد بازیگرها برای یک سیستم بزرگ بسیار زیاد است. برای این منظور می توان کلاس بازیگرها یا مجموعه بازیگرها را مشخص نمود. اصولاً هر بازیگر خود به عنوان یک کلاس در محیط رَشنال رُز درنظر گرفته می شود. علت این است که در واقع برای هر کاربر سیستم در داخل سیستم یک کلاس به عنوان سرویس دهنده به آن بازیگر باید در نظر گرفت که شامل منوها و رابطهای خاص آن بازیگر است. در شکل زیر رابطه بین کلاس بازیگرها مشخص شده است.



۲-۸ شناخت سیستم موجود

برای شناخت نیازها در روشهای قدیمی، ابتدا سیستم موجود را مورد شناسایی قرار داده، عملکرد آنرا مدلسازی می نمودند. امروزه نیز مدیریت سنتی، در راس هرمهای سازمانی خواهان مستندات عملکرد سیستم کنونی سازمان خود می باشند. چرا که این مستندات می توانند مبنایی برای ارزیابی و سنجش میزان بهبود در عملکردها باشند. در ادامه ابتدا در بخش ۲-۸-۱ به تحلیل شناخت سیستم موجود پرداخته سپس ، روش پیشنهادی رَشنال رُز برای این منظور ارائه می شود.

۲-۸-۱ محدوده شناخت

منطق حکم می کند که در شناخت سیستمها، نیازها و سیستم آتی بررسی شود. اما ء سنت گرایان و پیروان روشهای قدیم دستی بدور از شناخت امکانات موجود برای برنامه سازی سریع، باز هم اسرار بر شناخت سیستمهای کنونی و ایجاد مستندات زیبا و دراکثر موارد به دور از واقعیات با بهایی گزاف دارند. چگونه می توان بدون هیچگونه آزمایش قطعی هزاران صفحه مستندات را در شناخت عملکردها ایجاد کرد. لذا، متدولوژی RUP پیشنهاد می کند که باید یک هسته اولیه از برنامه اجرایی را در طی مراحل آنالیز تولید نمود و همگام با مراحل بعدی ، توسعه داد.

امروزه پیشرفت سریع تکنولوژی ، شناخت و مستند کردن عملکرد کنونی سیستمها را نمی پذیرد. زیرا ، تا زمانیکه مستندات سیستم کنونی تهیه می شود سیستم دچار دگرگونی و تحولات زیادی شده است. امروزه کامپیوتر به عنوان یک عامل ارتباطی سریع، امکان برخورد مستقیم را ایجاد نموده ، هرمهای سازمانی و مدیریت سلسله مراتبی را واژگون نموده است.

وجود شبکه گسترده بانک اطلاعاتی، امکان تصمیم گیریهای بموقع و درمحل و بدور از اتلاف وقت، برای تایید مدیریت و بالنتیجه تسریع عملکرد سیستمها را فراهم نموده است. مدیریت می تواند درقالب شوراهای مدیریتی و با استفاده از تکنولوژی شبکه های گسترده کامپیوتری بر عملیات نظارت و اشراف داشته باشد. متاسفانه در این راستا چشم اندزهای بعضا" غیر واقعی و تخیلی زمینه های هوش مصنوعی به همان اندازه آنالیستهای بدون دانش به روز و تجربه مستمر برنامه نویسی، خطرناک و مضر بوده است.

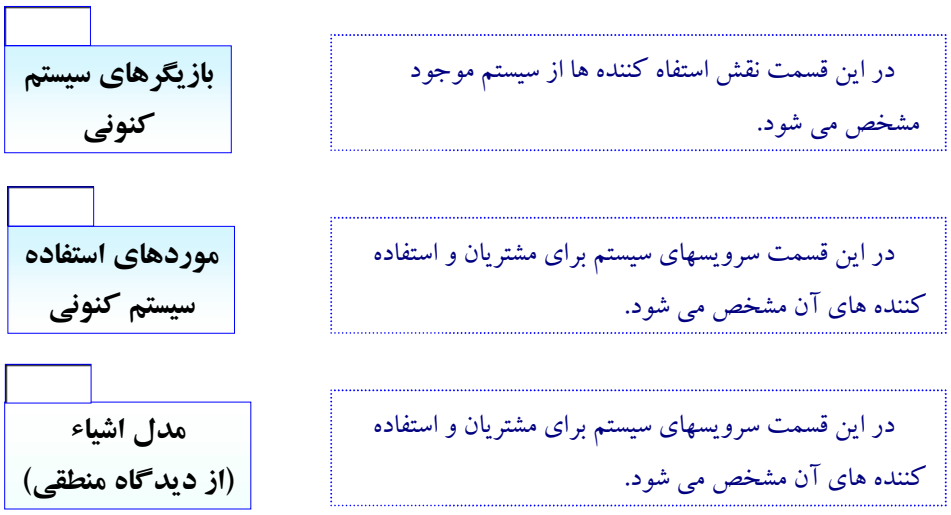
۲-۸-۲ مدلسازی سیستم کنونی در رَشنال رُز

برای مدلسازی عملکرد یک سیستم در حالت کلی سه بسته اصلی مطابق شکل ۲-۲۰ ایجاد می شود. یک بسته برای شناخت سیستم کنونی ، بسته دیگر برای مستندات آنالیزو سومین بسته برای

مستندات طراحی سیستم مکانیزه مطرح است. در ادامه با ذکر یک مثال مستندات شناخت سیستم موجود مورد بررسی قرار داده خواهد شد.

شکل ۲-۲۰ ساختار کلی مستندات تحلیل و طراحی

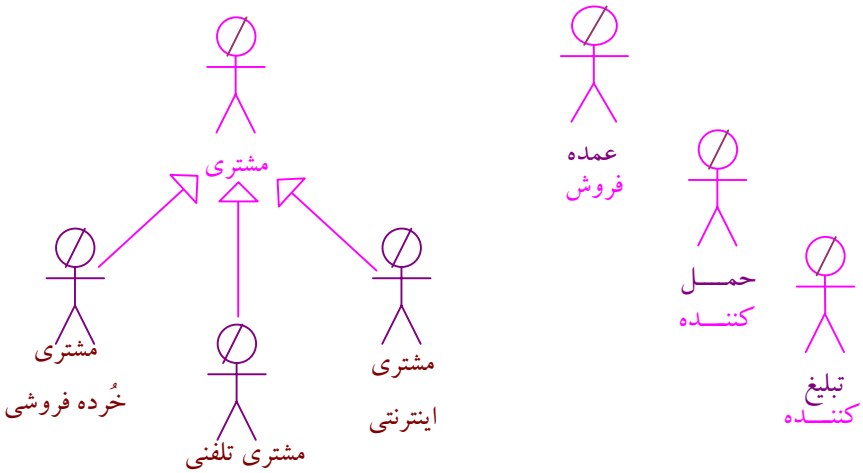
درون مستندات سیستم جاری سه بسته دیگر به صورت زیر، قرار می گیرد. بنا بر پیشنهاد شرکت رَشنال رُز مستندات سیستم کنونی شامل موردهای استفاده و مدل ارتباطی اشیاء مطرح در سیستم جاری است. این مدل در واقع توصیف ساده ای از سیستم موجو است، که در مرحله شناخت نیازمندیها می تواند تا حدی یاری دهنده آنالیست باشد.



شکل ۲-۲۱ : ساختار سیستم کنونی از دیدگاه موردهای استفاده

برای بسته بازیگرهای سیستم کنونی ، فقط سرویس گیرنده های سیستم مشخص و تا حد ی توصیف می شوند. علاوه بر این افراد ، زیر سیستمها و یا سیستمهای دیگر نیز که در ارتباط با سیستم هستند نیز به عنوان بازیگر در این بسته مشخص می شوند. با باز کردن بسته بازیگرها، مطابق شکل ۲-۲۲ کلیه موجودیتهای خارجی سیستم در قالب بازیگرها مشخص می شوند. شرحی از عملکرد هر یک بازیگرها در داخل محیط رَشنال رُز، قرار داده می شود. علاوه بر این هر بازیگر به عنوان یک کلاس مطرح می شود. درون این کلاس می توان منوهای مربوط به آن بازیگر جهت برقراری ارتباط با سیستم،

محدوده دسترسی به اطلاعات و سایر اطلات خـاص بـازیگر را گنجـاند. همانگونـه کـه در شـکل زیـر مشخص است برای بازیگرهای سیستم فیزیکی موجود[۲۹] علامت خاص در رَشنال رُز قرار داده اند.



ب: تعیین نوع توسط رابطه وراثت

الف : هر زیر سیستم بازیگری برای زیر سیستم دیگر است

شکل ۲-۲۲ : توصیف نقش بازیگرها در سیستم جاری

پس از تعیین بازیگرها، موردهای استفاده بازیگرها از سیستم مشخص می شود. باید توجـه داشـت که در هنگام تعیین این موردهای استفاده ارتباط زیـر سیسـتمها و ارتبـاط بـا خـارج مطـرح اسـت. بـرای مدیریتها به عنوان یک زیر سیستم معمولاً، کاربر و یا در اصطلاح بازیگر خاصی وجود ندارد. لـذا، در این راستا در عمل تئوری دچار مشکل گردیده ، موردهای استفاده بدون کاربر مشخص می شوند!!

در شکل ۲-۲۳، بسته موردهای استفاده از سیستم کنونی کـه در شـکل ۲-۲۱ مشـخص شـده بـود، توسعه داده شده است.

---

[۲۹] Business Actors

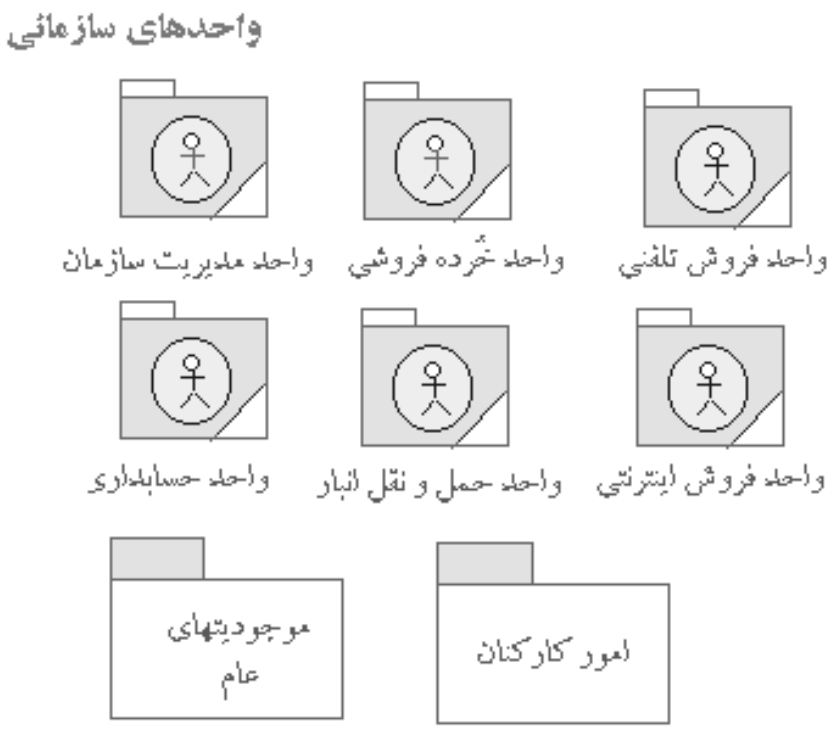
شکل ۲-۲۳ : دسته بندی موردهای استفاده از سیستم جاری بر اساس زیر سیستمها

جهت نمایش گردش کار برای هر یک از موارد استفاده از سیستم می توان دیاگرام فعالیت ترسیم نمود. دیاگرام فعالیت ابزاری جهت توصیف عملیات پیچیده است. قابلیت نمایش گردش کار را دارد. به این ترتیب که مشخص می کند جهت انجام یک کار چه مستنداتی بین چه افـراد و واحدهـای عملیـاتی مبادله می شود.

در دیدگاه منطقی[۳۰] ، زیر سیسـتم را بـر اسـاس واحدهـای سـازمانی تقسـیم بنـدی مـی کننـد. هـر واحدسازمانی نوعی خاص از بسته در UML است که با علامتی خاص کـه در شـکل ۲-۲٤ ارائـه شـده، تعیین می شود. این واحدهای سازمانی به صورت زیر سیستمها و در قالب بازیگرها نیز بعضاً در دیدگاه موردهای استفاده از سیستم در شکل ۲-۲۲ مشخص شدند. همانگونه کـه در شـکل ۲-۲٤ مشـخص مـی شده ، برای تطبیق با در نگرش شیء گرا ، عملکرد سیستم از همکاری واحدهای سازمانی به عنوان اشیاء عمده در سازمان حاصل می شود.

---

[۳۰] Logical View

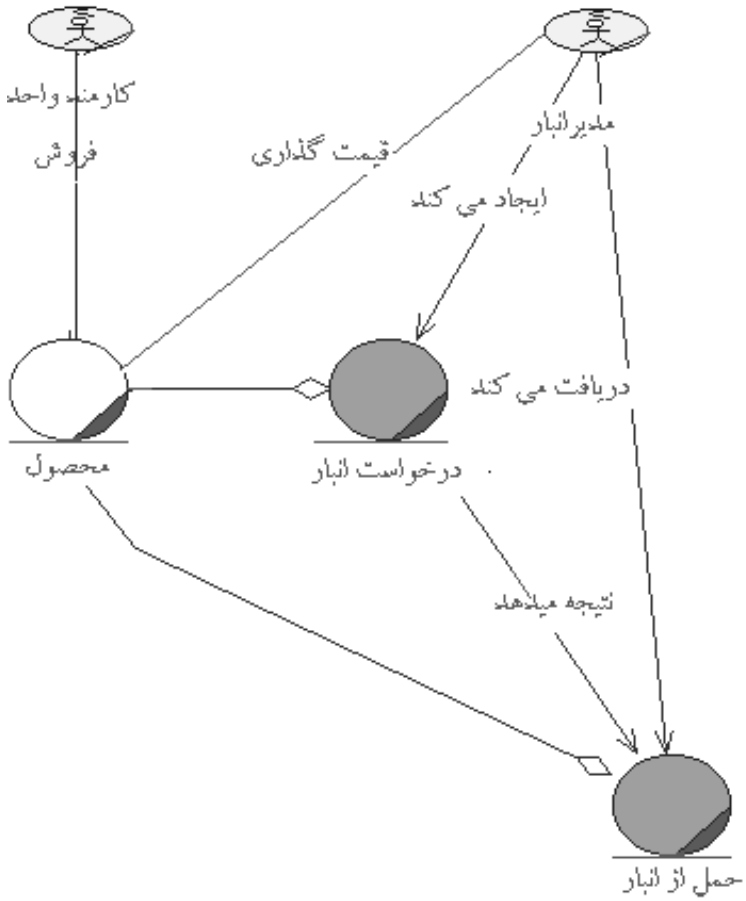


شکل ۲-۲٤: واحدهای سازمانی در قالب اشیاء عمده سیستم موجود

در دیدگاه منطقی برای سیستم موجود دو نوع کلاس در نظر گرفته می شود. یک دسته تحت عنوان کارمند[۳۱] و دیگری تحت عنوان موجودیت مشخص می شوند. کلاسهای کارمند با یک دایره که در مرکز آن آدمکی قرار گرفته مشخص می شوند. اینها در واقع امور سیستم جاری را به انجام می رسانند. کلاسهای موجودیت اطلاعات سیستم را درون خود نگهداری می نمایند. در واقع اینها مستنداتی هستند که توسط کارمندها ایجاد و مورد تبادل قرار می گیرند. این دسته از کلاسها با یک دایره که در زیر آن یک قطعه خط قرار داده شده مشخص می شوند. در شکل ۲-۲۵ ، مدل اشیاء سیستم جاری در قالب دیاگرام یا مدل ارتباطی کلاسها مشخص شده است.. مدل ارتباطی کلاسها تحت عنوان مدل ادراکی در فصل بعدی مورد بررسی قرار خواهد گرفت.

---

[۳۱] Business Worker



شکل ۲-۲۵ : اشیاء تشکیل دهنده سیستم

۲-۹ شناخت مسائل در نگرش شیء گرا

در سیر تکاملی شیء گرا به آنجا رسیده اند که نمی توان تنها بر اساس دسته بندی کلاسها و بر مبنای تجرید عملکردها ، فرایندها را شناسایی کرد. سیستمها طبیعتا" بر مبنای وظایف شناسایی و بر اساس وظایف سازماندهی و تقسیم کار انجام می شود. لذا، روش صحیح برای تقسیم بندی شناخت وظایف است. لازمه نگرش شیء گرا در شناخت پدیده ها ، تجرید و تعمیم یعنی مجرد کردن اشیاء از جزئیات و عمومیت دادن آنها در قالب کلاسها است. اما، مبنای تجرید، شناخت قبلی است. شما مبتنی بر شناخت قبلی خود از پرنده است که می توانید موجودی را که در آسمان پرواز می کند مستقل از متعلقاتش تجرید ، در قالب پرنده تعمیم و مورد شناسایی قرار دهید. لذا، در دیدگاه شیء گرا نیز مجبور به قبول این واقعیت گردیده، وظایف و عملکرد سیستمها را در قالب "موردهای استفاده" مبنای شناخت قرار دادند.

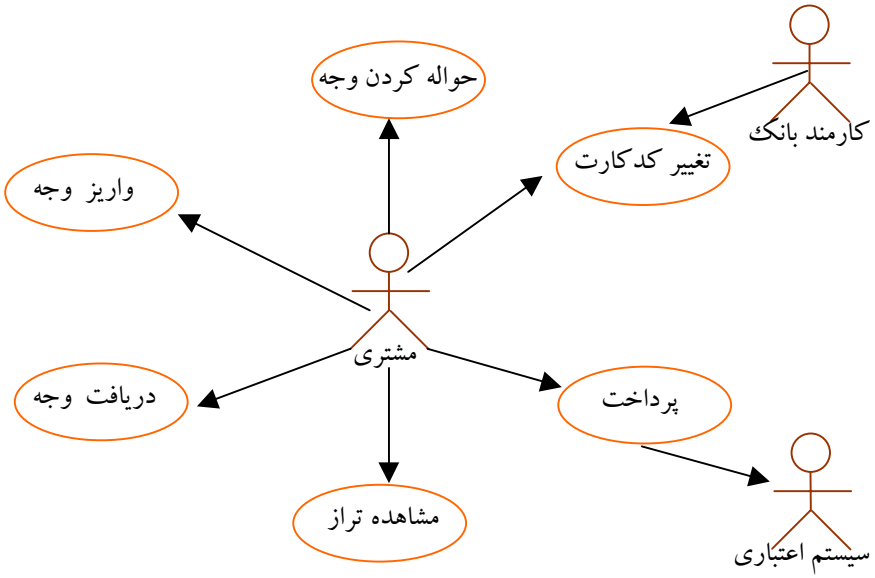
اصولا" شناخت نیازها مبتنی بر شناخت بایدها است. بایدها در مقابله با مشکلات متجلی می گردند. مشکلات را معمولا"باید در مقابل نبود سیستم مکانیزه شناخت. اما، وجود سیستم مکانیزه چگونه می تواند مشکلات کاربر را از میان بردارد؟ در اینجا است که دانش تحلیلگر از دنیای وسیع کامپیوتر می تواند مشکلات حاصل از عدم وجود آنرا برای کاربرها مشخص نماید. اما تحلیلگر باید سیستم را بشناسد تا بتواند بر مبنای شناخت خود مشکلات و در مقابله با آن باید ها را مشخص نماید.

بایدها را ابتدا کاربر مشخص می نماید و سپس آنالیست مبنایی برای شناخت قرار می دهد. برای شناخت بایدها باید سیستم را شناخت. سیستمها بزرگ هستند و نمی توان عملکرد آنها را تجرید و در قالب کلاسها مورد شناسایی قرار داد.

دردیدگاه شیء گرا بر خلاف دیدگاه ساخت یافته وظایف را از دید سیستم نمی شناسند. در دیدگاه شیء گرا وظایف از دیدگاه کاربرها مورد شناسایی قرار می گیرد. البته دیدگاه شیء گرا دیدگاه تحلیل وظایف هم نیست. در دیدگاه تحلیل وظایف جهت شناخت سیستمها وظایف افراد مورد تحلیل و بررسی قرار می گیرد. در اینجا وظایف سیستم مکانیزه از دیدگاه کاربر مورد بررسی قرار می گیرد. به عبارت دیگر از هر کاربر سوال می شود که چه انتظاراتی از سیستم مکانیزه در جهت رسیدن به اهداف خود دارد. لذا، مقوله مورد استفاده در دیدگاه شیء گرا در جهت بر آورده نمودن خواسته های کاربر مطرح می شود.

باید نقش کاربرها درون یک سیستم را شناخت تا بتوان در ارتباط با هر نقش و مسئولیت کاربر موردهای استفاده وی از سیستم ایده آل مکانیزه را مشخص نمود. کاربر می تواند یک فرد، یک

سازمان و یا یک شیء و بطور کلی هر چیزنیز باشد که از سیستم مکانیزه بهره خواهد برد. این بهره برها را در اصطلاح بازیگر یا اکتور می نامند.



شکل ۲-۲۰: دیاگرام مورد استفاده برای مشتری عابر بانک

همانگونه که در بالا مشاهده می نمایید در واقع موارد استفاده ابزاری برای نمایش سیستم از دیدگاه ناظر خارج از آن است. به عبارت دیگر دیاگرام متن را در بر می گیرد.

برای تعیین موارد استفاده ابتدا باید بازیگرها یا در اصطلاح اکتورها را مشخص نمود. بازیگر به هر چیزی اطلاق می شود که از سیستم بهره می برد. سپس برای هر بازیگر درون سیستم سوالات زیر مطرح می شود.

۱. هر بازیگر در ارتباط با شرح وظایف محوله ، نیاز به چه عملیاتی از سیستم خواهد داشت.
۲. آیا بازیگر در ارتباط با ایجاد، حذف، خواندن و تعویض اطلاعات عملی انجام می دهد.
۳. آیا بازیگر رویداد خارجی را به اطلاع سیستم می رساند.
٤. آیا نیاز است که سیستم رویدادها و برخی از تغییرات را به اطلاع بازیگر برساند.

پس از تعیین موارد استفاده برای حصول اطمینان از کامل بودن آنها باید تکات زیر را در نظرگرفت

۱. آیا نیازهای عملیاتی توسط موارد استفاده پوشانده شده اند
۲. آیا مشخص شده که هر بازیگری چگونه سیستم را مورد استفاده قرار می دهد
۳. آیا مشخص شده که بین سیستم و بازیگر چه اطلاعاتی مبادله می شود
٤. آیا پشتیبانی سیستم در نظر گرفته شده است.

معمولاً، موردهای استفاده را در چندین لایه می توان ترسیم نمود. لایه اولیه را تحت عنوان مورد استفاده اصلی یا در اصطلاح Main در محیط رَشنال رُز مشخص می نمایند. می توان با استفاده از روابطی مثل استفاه و توسعه یا در اصطلاح Uses و Extends یک مورد استفاده را با چند مورد دیگر ارتباط داد. این تنها حالتی است که دو مورد استفاده مستقیماً در ارتباط قرار می گیرند. در مجموع نباید بین موارد استفاده ارتباط مستقیم وجود داشته باشد. موارد استفاده خاص برای ثبت اطلاعات درون بانک اطلاعاتی باید در نظر گرفته شود. ارتباط مستقیم بین بازیگرها نباید وجود داشته باشد. هر مورد استفاده باید توسط یک بازیگر فعال شود. البته به استثنا موارد استفاده که مورد استفاده قرار می گیرند و یا اینکه توسعه دهنده یک مورد استفاده در حالت استثنایی هستند. بطور خلاصه در هنگام ترسیم دیاگرام موردهای استفاده نکت زیر را باید رعایت کرد:

۱- رابطه بین بازیگرها خارج از حوزه عملکرد سیستم و نباید در دیاگرام موردهای استفاده ترسیم شود.
۲- فرض بر این است که موردهای استفاده از طریق بانک اطلاعاتی مرتبط می شوند. پس به غیر ار حالات uses و extend رابطه مستقیم دیگری بین آنها وجود ندارد.
۳- کلیه موردهای استفاده باید توسط یک بلزیگر فعال شوند. مگر موردهایی که توسط روابط مستقیم با یک مورد اصلی مرتبط هستند.

فصل سوم

ساختن مدل ادراکی

۳-۱ مقدمه

درک مسائل بر اساس مفاهیم و روابط بین مفاهیم تشکیل دهنده، میسر می گردد. در دیدگاه شیء گرا مفاهیم بنیانی در قالب اشیاء مشخص می شوند. هدف از ایجاد یک مدل ادراکی سازماندهی مفاهیم است. مفاهیمی که از متن نیازها، موردهای استفاده و مسئولیتهای سیستم استخراج می شوند. در این فصل مراحل ایجاد مدل ادراکی و نهایتا" تبدیل آن به مدل ارتباطی کلاسها مطرح می شود.. مدل ادراکی ، ارتباط مفاهیم اساسی در قلمرو یک مساله را مشخص می نماید. تعیین و تعریف مفاهیم بخشی از تحقیق پیرامون قلمرو مساله است.

مدل ادراکی نقطه شروعی برای شناسایی و سازماندهی کلاسها است. دومین مرحله از فرایند نرم افزار در دیدگاه لاردن تجزیه و تحلیل است که شامل مراحل زیر است:

۱- تعیین موردهای استفاده ضروری

۲- تعدیل دیاگرام موردهای استفاده

۳- تعدیل مدل ادراکی

٤- تعدیل لعتنامه

٥- تعیین دیاگرام توالی

٦- تعیین قراردادهای عملیاتی

۷- تعیین دیاگرام حالات

مشاهده می کنید که دیاگرام ادراکی در این مرحله نیز تعدیل و اصلاح می شوند.

۳-۲ مدلهای مفهومی

از آنجائیکه مدلهای ادراکی از ارتباط بین مفاهیم ایجاد می شوند، می توان اشیاء تشکیل دهنده یک سیستم را به عنوان مفاهیم آن سیستم در نظر گرفت . در واقع مفاهیم یک سیستم بیانگر مولفه های دانش برای شناخت و یا طراحی آن سیستم هستند. با تعیین مفاهیم مطرح در قلمرو مسائل در دیدگاه شی ء گرا مسائل را تجزیه می نمایند. تقسیم و غلبه روشی رایج برای حل مسائل است . روش شناخت سیستمها رادر دیدگاه ساختیافته با تقسیم دامنه مسائل و سیستمها بر اساس توابع عملیاتی مبتنی نموده اند و دردیدگاه شی ءگرا تقسیم بندی بر اساس مفاهیم یا اشیاء است . همانطور که خواهید دید در UML یک مدل ادراکی از ارتباط بین اشیاءو در قالب مدلهای ساختاری نمایش داده می شود مدل ادراکی دربرگیرنده :

- مفاهیم یا کلاسها
- ارتباطات بین مفاهیم یا رابطه بین کلاسها
- صفات مفاهیم یا درواقع فیلدهای کلاسها

۳-۳ تعیین اشیاء

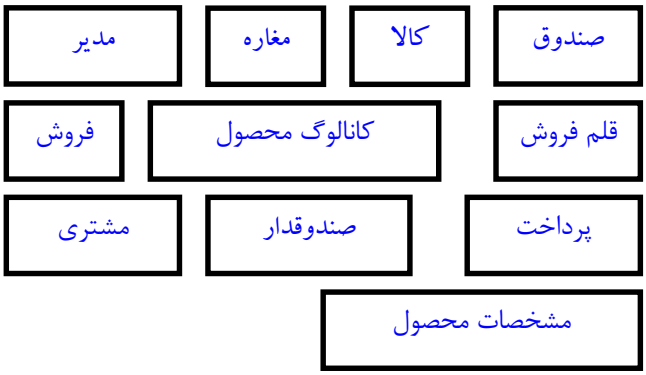
کلاس اشیاء را با تعیین اسامی که درون متن سناریوها ظاهر شده اند می توان مشخص نمود. روش دیگر برای تعیین کلاسها مشخص کردن مسئولیتها جهت انجام سرویسها و موردهای استفاده است . البته باید توجه داشت که اشیاء تنها فیزیکی نیستند. اشیاء ممکن است مفهومی نیز باشند. برای نمونه ، استخدام یک شی ء مفهومی است . در ادامه لیستی ازانواع کلاسها و نمونه های مشابه در سیستم فروشگاه مشخص شده است .

مثال	انواع اشیاء
POST یا صندوق	اشیاء فیزیکي
توصیف کالا	توصیف ، طرح
فروشگاه	مکانها
فروش ،پرداخت	تراکنشها
صندوقدار	نقش کاربر
واحد فروش	سازمان ها
فروش ،خرید	رخدادها
خرید	فرایندها

می توان کلاسها از متن موردهای استفاده با در نظر گرفتن انواع کلاسـها کـه در جـدول فـوق مشخص شده ، استخراج نمود. برای نمونه به مورد استفاده زیر توجه نمائید:

عمل کنشگر	پاسخ سیستم
۱-این مورد با رجوع مشتری همـراه بـا اجناس مورد خرید به صندوق فروشگاه آغاز می شود	
۲-صندوقـدار هـر جنـس را ثبـت مـی کند، اکـر بیـش از یـک جنـس باشـد، صندوقدار تعـداد آنـها را نـیر وارد مـی کند.	۳-تعیین قیمت کالا و اضافه کردن اطلاعات اشیاء بـرای اجرای تراکنش فروش
۴-برای تکمیل ورود کالاها ،صندوقـدار مشـخص مـی کندکـه ورود اجنـاس تکمیل شده است .	۵-کل فروش را محاسبه و نمایش می دهد.

اسامی کلاسها را از متن سناریو فوق می توان مشخص نمود. برخی از کلاسها را می توان از ارتبـاط بین کلاسهای موجود مشخص کرد. برای نمونه از ارتباط بیـن دو شـیء فـروش و مشـتری ، شـیء صندوق حاصل می شود. رابط بین دو شیء را در اصطلاح Link یا پیوند بین آن دو شیء می نـامند و رابطه بین دو کلاس را اجتماع یا Association گویند.



شکل ۳-۱: مفاهیم یا کلاسهای سیستم صندوق فروشگاه (POST)

یک روش برای تعیین کلاسها دسته بندی وظایف سیستم و مشخص کـردن مسئولیتها اسـت . هـر تعدادی از مسئولیتها را می توان برای یک کلاس مشخص کرد. معمولا"، حداکثر چهار مسـئولیت بـرای یک کلاس در نظر می گیرند. اگر کلاسی مسئولیت نداشـته باشـد دلیلـی هـم بـرای وجـود آن کـلاس

نیست. می توانید مسئولیتها را از متن موارد استفاده از کلاسها مشخص نمائید. .برای نمونه در سیستم مکانیزه برای مطب پزشک یک مسئولیت دادن نوبت به بیماران است . بنابراین می توان شی ء ای به نام نوبت را مشخص کرد. مسئولیت دیگر معرفی بیمار به پزشک است . برای سیستم کتابخانه دانشگاه که در فصل دوم مطرح شد، سه شی ءبا مسئولیتهای زیر مشخص شده است :

عضو کتابخانه	
مسئولیت	همکارها
نگهداری اطلاعات نسخه ها در امانت عضو دریافت تقاضای برگشت یا امانت نسخه ها	نسخه

کتاب	
مسئولیت	همکارها
نگهداری اطلاعات یک کتاب آگاهی از وجود نسخه قابل امانت	

نسخه	
مسئولیت	همکارها
نگهداری اطلاعات یک نسخه از کتاب اطلاع امانت یا برگشت نسخه به کتاب	کتاب

شکل ۳-۲: کارتهای تعیین مسئولیت و همکاری بین کلاسها (**CRC Cards**)

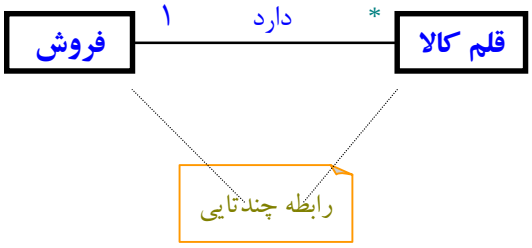
۳-۴ ساختن یک مدل ادراکی

هدف از تعیین چگونگی ارتباط بین کلاسها ایجاد امکان استفاده از امکانات یک کلاس از طریق کلاس دیگر است . بنابراین ارتباط بین اشیاء شاخص چگونگی فراخوانی متدها یا استفاده از فیلدهای یک کلاس از طریق کلاس دیگر است . این ارتباط ممکن است از طریق یک فیلد از نوع کلاس مورد نظر یا از طریق یک نشانگر و یا از طریق رابطه وراثت پیاده سازی شود. انواع روابط بین کلاسها را به صورت زیر می توان خلاصه نمود:

مثال	دسته
Drawer-POST(صندوق -کشو)	A یک بخش فیزیکی از B است
فروش جنس فروشی	A یک بخش منطقی از B است
کالا-قفسه ، صندوق فروشگاه	A به طور فیزیکی درون / روی B است

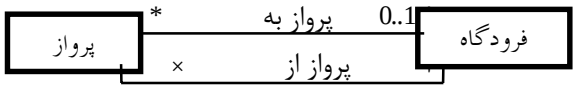
کاتالوگ کالا–شرح کالا	A به طور منطقی درون / روی B است
شرح کالا محصول	A توصیفی از B است
قلم فروش– فروش	A سطری از تراکنش / گزارش B است
فروش– فروشگاه	A ثبت /تراکنش/ گزارش/ نگهداری
فروش صندوق	شده توسط B است
صندوقدار–صندوق	A عضوی از B است
صندوقدار–صندوق	A ، B را استفاده یا مدیریت می کند
صندوقدار–مشتری	A باB ارتباط برقرار می کند
مشتری پرداخت	A مرتبط با تراکنش B است
پرداخت – فروش	A تراکنشی در ارتباط با تراکنشB است.
صندوق–فروشگاه	A در تملک B است

.در شکل زیر، فروش با کالا یک رابطه یک به چند دارد.یعنی هر کالای خاص متعلق به یک فروش است ولی هر فروش دارای یک یا چند کالا می باشد. در اجتماع دو کلاس هر کلاس ممکن است نقشی داشته باشد. برای نمونه رابطه فرد با کارخانه را در نظر بگیرید. در این اجتماع ، فرد در نقش کارگر و کارخانه در نقش کارفرما می توانند قرار بگیرند. برای یک اجتماع می توان تعداد در نظر گرفت. برای نمونه در اجتماع کالا با فروش یک رابطه یک به چند وجود دارد. یعنی اینکه یک مورد فروش می تواند شامل چند قلم کالا باشد. علامت ستاره در شکل زیر بیانگر رابطه چند تایی است.



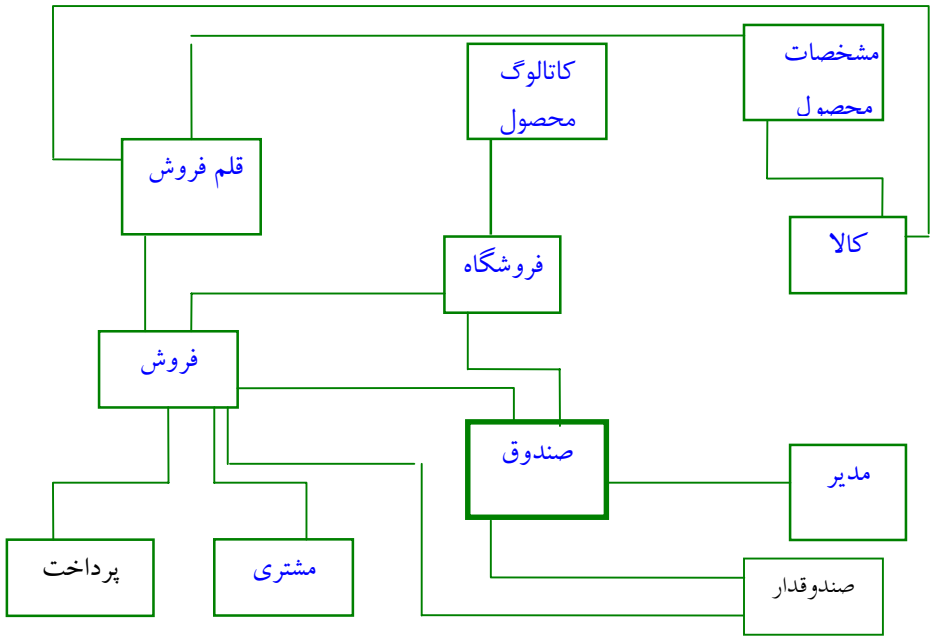
شکل ۳–۳ معرفی رابطه چندتائی

دو کلاس ممکن است بیش از یک رابطه با یکدیگر داشته باشند.ممکن است . در شکل ۳–٤ روابط "پرواز از" و "پرواز به" با دو خط مجزا مشخص شده است .



شکل ۳–٤ رابطه چندگانه

در شکل ۳-۵ مدل کامل مفهومی یا ادراکی برای زیر سیستم صندوق فروشگاه مشخص شده است.



شکل ۳-۵ : یک مدل ادراکی از برنامه POST

پایانه صندوق یا در اصطلاح POST بوسیله مدیر سیستم أماده می شود. اطلاعات کالا براساس کدکالا دسترسی می شود . اشیاء از نوع پرداخت وفروش نتیجه اصلی فرایند فروش کالا را مشخص می کنند.

مشخصات یک کلاس شامل نام، صفات و متدها در سه ناحیه مختلف در داخل کادر شاخص کلاس مشخص می شوند. در واقع یک کلاس فرم کامل شده یک مفهوم در دیدگاه شیٔ گرا است.

نام : فروش
صفات : مبلغ
متدها : ثبت حذف

شکل ۳-٦ : نمایش بصری کلاس

فیلدها یا به عبارت دیگر صفتهای ارتباط دهنده بین مفاهیم را نباید در مدل ادراکی مشخص نمود. کلید خارجی در بانک اطلاعاتی یک فیلد ارتباط دهنده است که نباید در شرح کلاسها ظاهر شود. زیرا ، مدل ارتباطی کلاسها خود مبین این ارتباط است. البته، در هنگام پیاده سازی نیاز به این فیلدهای ارتباط دهنده خواهد بود. مدل ارتباطی کلاسها گویای این نوع فیلدهای ارتباط دهنده

است. برای نمونه در شکل زیر هر صندوقدار در ارتباط با یک صندوق است. نباید در مشخصات صندوقدار شماره صندوق متعلق به وی را مشخص نمود. زیرا، مدل گویای این نکته است.

شکل ۳-۷ : کلید خارجی را در داخل کلاس مشخص نکنید

باید صفتها یا فیلدها برای مفاهیم قلمرو مساله مشخص شوند. صفتها با خواندن مشخصات نیازمندیها ، توجه به مورد های استفاده و مستندات ، استخراج می شوند. .برا ی مثال ، ضروری می باشد که زمان وتاریخ فروش ثبت شود. پس مفهوم فروش نیازبه صفتهای تاریخ و زمان دارد. مجموعه نیازمندیهای قلمرو مساله زیر را در نظر بگیرید:

۱. برای هر فروشگاه از مجموعه فروشگاه های زنجیره ای شاخصی تحت عنوان کد تجاری ، تخصیص داده می شود.

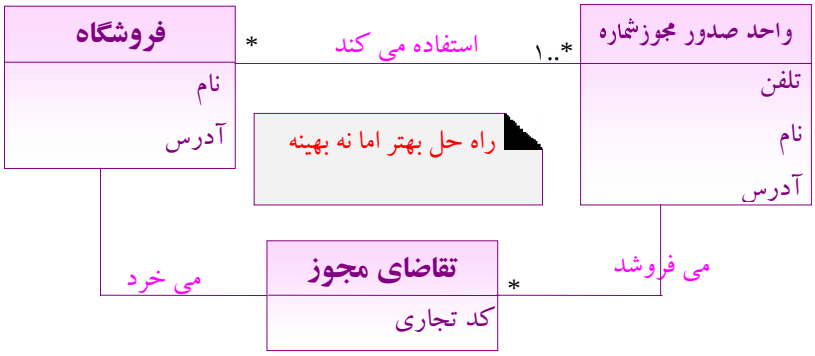
۲. این کد تجاری هویت فروشگاه را در هنگام تقاضای دریافت تائیدیه چک یا کارت اعتباری مشتری از واحد مرکزی مجوز، مشخص می نماید.

۳. هر فروشگاه یک شماره مجوز متفاوت برای هر سرویس خود دارد.

حال سوال اینجاست که در مدل ادراکی ، صفت کد تجاری باید در کجا قرار گیرد؟جایگذاری کد تجاری در کلاس فروشگاه صحیح نیست زیرا، یک فروشگاه می تواند بیش از یک کد تجاری داشته باشد. ازطرفی قراردادن آن شماره در کلاس مجوز نیزغلط است زیرا مجوز عمومی است. . پاسخ را در اصل زیر می توان یافت :

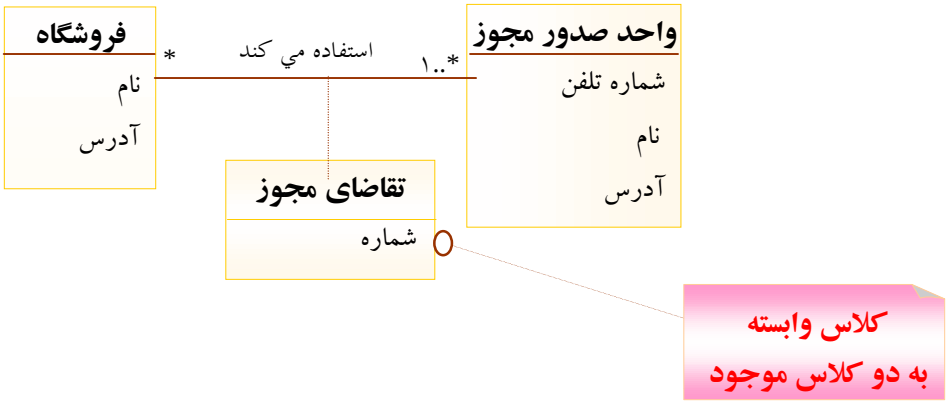
> **در یك مدل ادراكی ،اگر نوع T شامل تعداد بیش از یك عدد از صفت A است، صفت Aرا نباید در T قرار داد. صفت A را باید درنوع دیگری كه با T رابطه دارد قرار داد و به این ترتیب یك رابطه یك به چند ایجاد نمود.**

برای مثال یک شخص ممکن است دارای شماره تلفنهای زیادی باشد.شماره تلفن باید در نوع دیگری مانندPhonenumber قرار داده شود وسپس باید این نوع به نوع شخص مرتبط شود. در واقع رابطه یک به چند بین صفتهای یک کلاس موجب تفکیک آن کلاس به دو کلاس مجزاء می شود.



شکل ۳-۱۰: اولین تلاش در جهت مدلسازی مساله کد تجاری

توجه نمائید که وجود شیء از نوع تقاضای مجوز وابسته به وجود دو شیء فروشگاه و واحد صدور مجوز است به عبارت دیگر وجود شیء صدور مجوز وابسته به اجتماع دو شیء دیگر است. کد تجاری ابزاری جهت برقراری ارتباط بین دو شیء فروشگاه و واحد صدور مجوز است . کد تجاری نیز یک ویژگی حاصل از اجتماع دو شیء فروشگاه و واحد صدور مجوز است . لذا، کد تجاری صفت یا فیلدی از شیء تقاضای مجوز است . کلاس تقاضای مجوز را یک نوع اجتماع می نامند. برای نمایش این انواع از خطوط خط چین مشابه شکل ۳-۱۱ استفاده می شود.



شکل ۳-۱۱: یک نوع حاصل از اجتماع

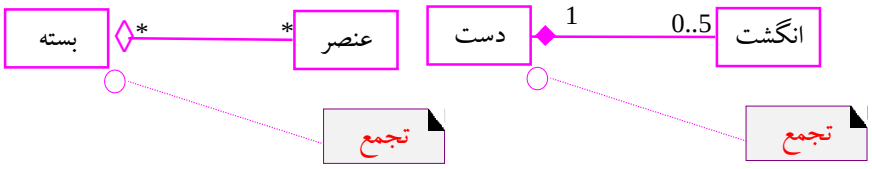
در شکل بالا یک نوع کلاس جدید تعریف شده است که صفتهایش وابسته به چگونگی اجتماع دو کلاس موجود است. چرخه حیات شیء جدید به این اجتماع بستگی دارد. اصولا"، زمانی یک رابطه اجتماع تولید کلاسی جدید می نماید که :

- یک رابطه چند به چند بین دو شی ء وجود دارد.
- اجتماع بین دو کلاس دارای چند ویژگی باشد.

۳-۵ رابطه تجمع[۳۲]

یک شیء ممکن است از کنار هم قرار گرفتن چند شیء دیگر ایجاد شود و یا شامل چند شیء دیگر باشد. این نوع رابطه ها را رابطه تجمع و یا کل به جزء می نامند. شیء کل را، مرکب نیز می گویند. اما جزء نام استاندارد ندارد ولی بیشتر به قطعه معروف می باشد. چنانچه جزء تشکیل دهنده کل باشد. رابطه تجمع را با لوزی توپر مشخص می کنند. برای نمونه رابطه دست با انگشت را در نظر بگیرید. انگشت جزیی از دست است و نمی توان به اشتراک آنرا قرار داد. بالعکس چنانچه کل شامل جز' اما مستقل از آن باشد، رابطه را با لوزی توخالی مشخص می کنند. می توان بجای قرار دادن لوزی با عبارت "بخش است از" نیز رابطه تجمع را مشخص نمود.

. رابط تجمع اشتراکی بدین معنی می باشد که تعدد در بخش مرکب ممکن است بیش از یک باشد. یعنی شیء جزء ممکن است بخشی از چند شیء کل باشد. این رابطه با یک لوزی توخالی نشان داده می شود. برای نمونه در شکل ۳-۱۲، یک دست بین ۰ تا ۵ انگشت دارد. در حالیکه یک بسته یا Package در UML تجمعی از چند عنصر است و هر عنصر UML می تواند در بیش از یک بسته قرار گیرد. اما، یک انگشت فقط در یک دست می تواند قرار بگیرد:



شکل ۳-۱۲: رابطه کل به جزء

ممکن است اجزاء بین کلهای متفاوت به صورت مشترک مورد استفاده قرار گیرند. ممکن است شی ء کل از مونتاژ و یا اسمبل کردن اجزاء ایجاد شود. برای نمونه ایجاد یک اتومبیل در CAD را در نظر بگیرید. ممکن است شی ء کل به صورت ظرفی دربر گیرنده اجزاء باشد. برای نمونه در پست الکترونیک ممکن است چند پیام قرار داشته باشد. هر پیام خود نیز شامل رشته آغازین ، بدنه و رشته انتهایی می باشد. ممکن است شی ء کل تکراری از اجزاء یکسان باشد. برای نمونه اشیاء گرافیکی را در نظر بگیرید. با وجود کتابخانه ای از اشکال هندسی مثل دایره ، خط و نقطه در واقع اجزاء در دسترس قرار دارند. هدف ایجاد اشکال کلی با استفاده از این اجزاء آماده است . اجزاء در واقع اشیایی هستند که بخاطر پنهان سازی جزئیات فیزیکی صفحه نمایش ، مثل فشردگی نقاط ، از مختصات مجازی استفاده

---

[۳۲] Aggregation

می کنند. کلاس پایه برای اشیاء گرافیکی ، متدهای عمومی مثل Draw برای ترسیم اشکالRotate حرکت دوران و متد دیگری به نام Dump را مشخص می نماید

```
abstract class GraphicalObject .  -
  {  abstract public void dump();
     abstract public void draw();
     abstract public void  rotate(int xc, int yc, double angle);
     ... }
```

برای نمونه کلاس Triangle برای اشکال مثلثی شکل را در نظر بگیرید. مثلث از اسمبلی سه نقطه که برروی یک خط قرار نمی گیرند، ایجاد می شود. بنابراین پیاده سازی کلاس Triangle وابسته به کلاس مشخص شده برای نقطه ، point ، است . در واقع مثلث کلی است که از اجزاء نقطه ایجاد می شود. عملیات روی مثلث را به این ترتیب باید بتوان با بهره بری از عملیات روی نقطه ایجاد نمود. برای نمونه چنانچه دوران یا rotate ، یکی از سرویسهای کلاس کل مثلث Triangle باشد ، باید بتوان با فراخوانی متد rotate متعلق به کلاس نقطه ، هر سه راس مثلث را دوران داد.

```
class point extends GraphicalObject
    { int x, y;
      public static double dist ( point p, point q)
        {double res = 0;
         if (p.x == q.x)  res = Math.abs( p.y - q.y)
          else { double dx  = p.x - q.x; double dy  = p.y - q.y;
                 res = Math.sqrt(dx*dx + dy*dy); }}
      public static boolean isCoLinear (point p, point q, point r)
        { double tmp = dist(p,q) - ( dist(p,r) + dist(r,q));
          if (math.abs(tmp) < EPSILON) return true; else return false; }
      public point( int xCoord, int yCoord) { x = xCoord; y = yCoord;}
      public
                                system.out.println
      public boolean isEqual( point aPoint)
        { return ( x == aPoint.x ) && ( y == aPoint.y)); }
      public void rotate(int xc, int yc, double angel)
        { if (is Equal(new point(xc, yc)) return;
          else { double cosa  = Math.cos(angel); double sina  = Math.sin(angel);
                 double dx = x - xc; double dy = y - yc;
                  x = (int) Math.round( cosa*dx - sina*dy + xc);
                  y = (int) Math.round( sina*dx + cosa*dy + yc); } } }
```

آزمون تشکیل مثلث توسط سه نقطه داده شده ، در متد سازنده کلاس مثلث باید انجام شود. برای این منظور باید متد آزمون همخطی تحت عنوان coLinear از کلاس نقطه مورد فراخوانی قرار گیرد. سه نقطه p ، q و r همخط هستند ، اگر فاصله بین دو نقطه مثلاً p و q مساوی با فاصله q و r به اضافه فاصله p و r باشد. کلاس مثلث به شرح زیر است :

```
class Triangel extends GraphicsObject
  { point p1, p2, p3;
```

```
public Triangel( point po1, point po2, point po3) throws PointsAreColinear
  {  if (point.isCoLinear(po1, po2, po3) throw PointsAreColinear;
     p1 = po1; p2 = po2; p3 = po3;}
public void dump()
  { System.out.println
    System.out.println(        1        1.dump();
    System.out.println(        2        2.dump();
    System.out.println(        3        3.dump(); }

public void  rotate( int xc, int yc, double angle)
   { p1.rotate(xc, yc, angle); p2.rotate(xc, yc, angle); p3.rotate(xc, yc, angle);  }
```

می توان با استفاده از نوع متغیر Vector مجموعه ای از اشکال گرافیکی را در قالب یک کلاس نگهداری نمود. این مجموعه در ادامه در قالب کلاس GroupObject معرفی شده است . در واقع این کلاس یک کل است که اجزاء آن از انواع متفاوت هستند. هر جزء این کلاس یک شکل خاص هندسی است .

```
class GroupObject extends Graphics object
  { private Vector members = new Vector();
    public int size(){ return members.size;}
    public GraphicsObject objectAt(int pos)
      { return (GraphicsObject)( members.elementAt)pos)); }
    public void addObject( GraphicsObject aShape)
      { members.addElement(aShape);}
    public void rotate(int xc, int yc, double angle)
      { for (int I=0; I< members.size(); I++)  objectAt(i).rotate(xc, yc, angle); }
    public void dum()
      {  System.out.println
         for (int I=0; I< members.size(); I++)  objectAt(i).dump(); }
```

فرض کنید که اکنون یک برنامه نویس می خواهد محیطی ایجاد کند که بتوان اشکال هندسی را ایجاد نموده ، تبدیل به یک گروه نمود و گروه اشکال را در حول نقطه (۰،۰) دوران داد. برای این منظور اکنون می توان کد زیر را ایجاد کرد

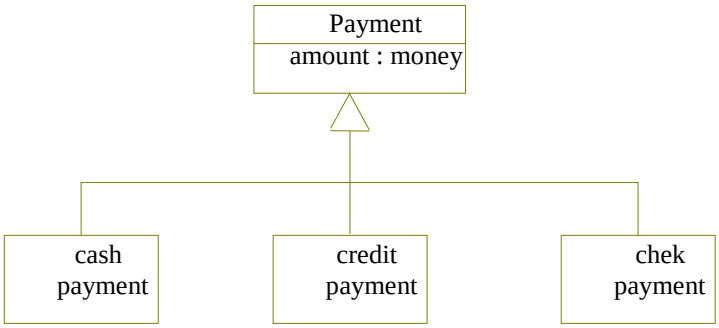
```
point  p1 = new point(10, 10);:
point  p2 = new point(10, 20);
point  p3 = new point(10, 20);
Triangle   t = new Triangle(p1,p2,p3);
Circle     c = new Circle(new point(0,0), 10);
Rectange r = new Rectangle(new point(-5, -5), new point(+5, +5));
Line l = new Line( new  point(1,1), new point(10,5));
GroupObject g = new GroupObject();
g.addObject( t );
g.addObject( c );
g.addObject( r );
g.addObject( l );
```

اشکال پیچیده تر را می توان به صورت ترکیبی از اشکال ساده فوق ایجاد نمـود. بـرای نمونـه یک دایره توخالی را می توان به صورت درخت دوتایی مشخص کرد به قسمیکه گـره سـمت چـپ نمایانگر دایره و گره سمت راست نمایانگر سوراخ مستطیلی شکل درون دایـره اسـت و گـره ریشـه حاوی عملگر منها است . در اینجا ساختار درختی برای تعیین اشکال پیچیده بر اساس اشکال سـاده تر در واقع یک ظرف است و گره های درخت که حاوی اشکال ساده هسـتند بیـانگر مظـروف مـی باشند.

۳-۶ رابطه وراثت

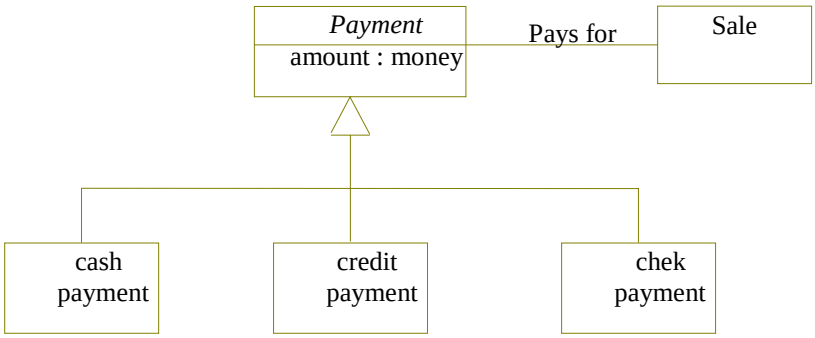
جنس کلاسها را در قالب کلاس مافوق برای آنها مشخص می کنند. در واقع یک کلاس مافوق بیانگر کلیت زیر کلاسهای خود است. رابطه بین کلاس مافوق و زیر کلاسها را رابطه "نوعـی اسـت از" نیز می نامند. برای نمونه رابطـه وراثـت بیـن کـلاس طـاووس و کـلاس پرنـده را بـه صـورت "طاووس توعی است از پرنده" می توان مشخص کرد. این رابطه مشخص می کند که طاووس دارای کلیه ویژگیهای یک پرنده است و یا به عبارت دیگر ویژگیهای پرنده بودن را به ارث می برد. بخـاطر اهمیت این نوع رابطه آنرا با علامت مثلث توخالی مشخص می کنند. کلاس مافوق شـاخص جنـس زیر کلاسها است. هر زیر کلاس خود یک نوع است. آنچه که زیر کلاسها را از یکدیگر متمـایز مـی نماید، در اصطلاح کلیت فصل گویند. در واقع فصل در مقابل جنـس همـانند نـور بـرای تشـخیص اشیاء است.

مشترکات کلاسها را به خاطر تفهیم مفهوم کلاس در قالب کلاس مـافوق تجریـد مـی نمـایند. مفاهیم pay by cash،pay by credit و pay by check همگی مشـترکا" انواعـی از جنـس Payment هستند.



شکل ۳-۱٤ نمودار سلسله مراتبی تعمیم سازی _ویژه کاری

تعمیم یا عمومیت دادن کلاسها در قالب کلاس مافوق روشی مـی باشـد کـه مفـاهیم را بـر اسـاس نمودار سلسله مراتبی طبقه بندی می کند. درuml رابطه تعمیم بین عناصر با یـک مثلـث نشـان داده می شود. در رابطه وراثت هرخاصیت کـلاس مـافوق بـرای زیـر کلاسـهای آن نـیز یـک خـاصیت محسوب می شود. برای نمونه صفت مبلغ در ارتباط با شی ء پرداخـت یـا Payment در ارتبـاط بـا فروش قرار گرفته و این صفت برای کلیه زیر کلاسها نیز مطـرح مـی باشـد. در شـکل زیـر کـلاس Payment به عنوان یک کلاس چکیده یا Abstract با حروف ایتالیک مشخص شده است :

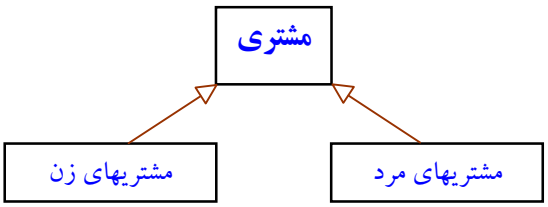


شکل ۳-۱۵: نمودار سلسله مراتبی نوع پرداخت

تعریف پرداخت بطور کلی تجزیه ناپذیرتر وعمومی تر ازتعریف پرداخت بوسیله کارت اعتبـاری است .همه زیر کلاسها ، از جنس کلاس مافوق خود می باشند.مثلا همه نمونه هـای پرداخـت بوسـیله کارت اعتباری ، از نوع پرداخت می باشند. در دیاگرام زیر ایـن مفـاهیم محسوسـتر مـی باشـند. قـانون دیگری بنام Is_a kind of وجود دارد که بیان می کند همـه عضوهـای یـک مجموعـه زیرکـلاس بـاید نوعی از کلاس مافوق باشند. باین رابطه به صورت زیر بیان می شود :

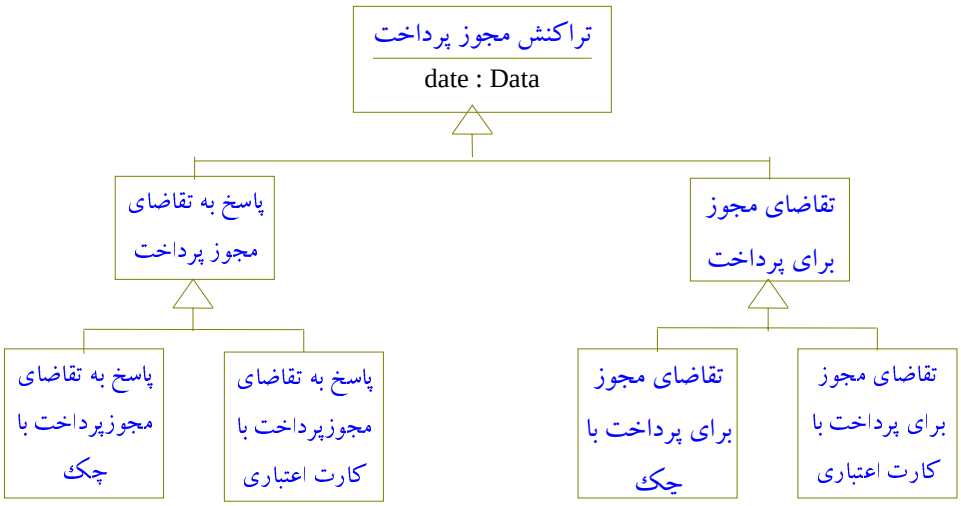
زیرکلاس نوعی از کلاس مافوق است

برای مثال اگر در برنامه کاربردی صندوق فروشـگاه ، مشـتری را براسـاس نـوع بـه مشـتریهای زن ومـرد تقسیم شوند ، آنگاه این تقسیم بندی با گرام سلسله مراتبی زیر نشان داده می یشود.



شکل ۳-۱۶ انواع مشتری

در شکل زیر تراکنش اجازه پرداخت در تاریخ و زمانی خاص شروع شده ، دو طریق پرداخت شامل استفاده از کارت اعتباری و چک را تصویب یا رد می کند.

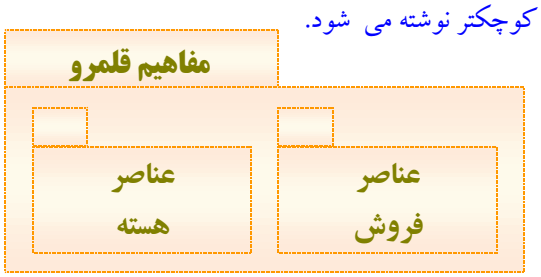


شکل ۳-۱۷: نمودار سلسله مراتبی برای تراکنشهای مجوزپرداخت

به دلیل تنوع مفاهیم در مدل ادراکی نیاز به ایجاد یک مکانیزمی هست تا بتوان این پیچیدگیها را کنترل کرد. به همین منظور از مکانیزم بسته بندی استفاده می شود.

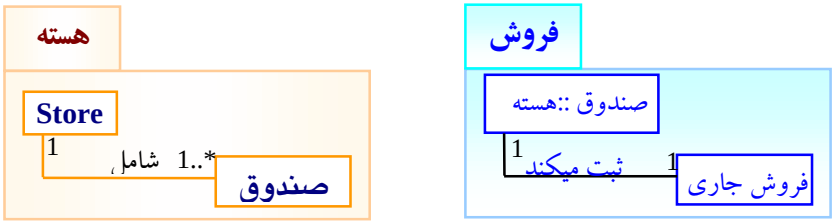
۳-۷ بسته

در روش ساختیافته سیستم ها را بر اساس توابع عملیاتی به زیر سیستمها تقسیم می کنند. این عمل تقسیم بندی و تعیین زیر سیستمها در نگرش شی ء گرا بر اساس دسته بندی کلاسها در قالب بسته ها صورت می گیرد. مدل ادراکی برای چرخه تولید نرم افزار بدلیل تنوع مفاهیم و عناصر عریض و گسترده است . بنابراین باید این عناصر بدرون بخشهای کوچکتر سازماندهی شوند. این سازماندهی باعث می شود در سطح بالای طراحی ،جزئیات کار نادیده گرفته شده ومدل کلان دیده شود. از نظر گرافیکی یک بسته با یک مستطیل نشان داده می شود که بر بالای سمت چپ آن مستطیلی کوچکتر واقع است . نام بسته در مستطیل کوچکتر نوشته می شود.



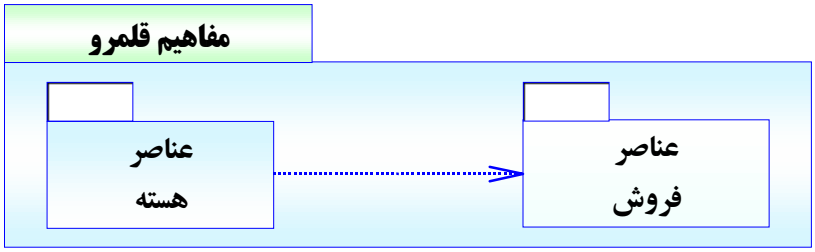
شکل ۳-۱۸: یک بسته UML

عمل بسته بندی تنها محدود به کلاسها نیست . ممکن است به یک کلاس از درون چند بسته ارجاع شود. در آن صورت نام کلاس بوسیله نام بسته در قالب مسیری مانند نام عنصر ::نام بسته می آید.. .



شکل ۳-۱۹: یک نوع مرجوع در یک بسته

در حالت کلی در UML وابستگی بین دو مدل را با فلش خط چین مشخص می کنند. اگر عنصری از یک بسته به هر طریقی با عنصری درون بسته دیگر اتصال داشته باشد، این اتصال را با فلش خط چین مشخص می کنند. برای نمونه بسته فروش در ارتباط با بسته هسته است .



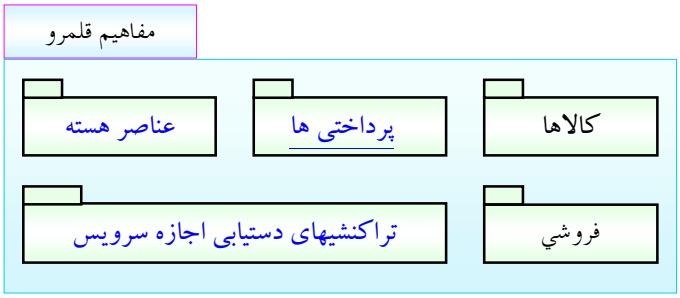
شکل ۳-۲۰: یک بسته UML

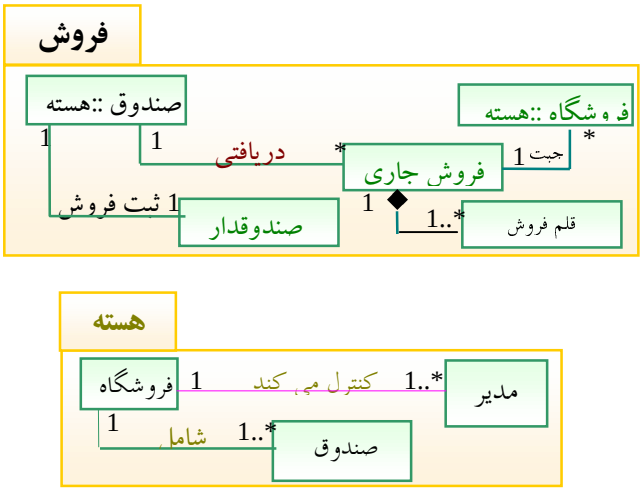
بعد از اینکه مدل ادراکی ایجاد شد، باید عناصر آنرا افراز و هر افراز را در قالب یک بسته مشخص نمود. در اینجا نکته معیارهای افراز بندی است . معمولا"، عناصری را در یک افراز قرار می دهند که :

- در ارتباط با یک موضوع خاص بوده ، هدف یا مفهومی آنها را مرتبط می نماید
- در یک سلسله مراتب وراثتی هستند
- در موارد استفاده مشابه شرکت می کنند
- اجتماع مستحکمی بین آنها برقرار است یا به عبارت دیگر اتصال زیادی با هم دارند

ضروری ترین کار در ابتدای یک بسته بندی ،این است که عناصر مدل ادراکی در یک بسته بنام مفاهیم قلمرو نشان داده شوند. سپس مفاهیم مشترک کلیه عناصر به شرط عمومی بودن در بسته دیگری بنام عناصر هسته ذکر شوند. بر اساس موارد بالا ،دسته بندی مدل ادراکی سیستم فروشگاه در زیر ارائه شده است . در ادامه هر یک از بسته ها و کلاسهای تشکیل دهنده آنها برای سیستم صندوق مشخص

شده است . یک بسته می تواند نمایانگر یک پیمانه یا Module از برنامه باشد. بسته ها عناصر معماری نرم افزار شی ء گرا هستند. لذا، مدل ارتباطی بسته ها شاخص معماری نرم افزار است.





شکل ۳-۲۱ : بسته بندی مدل ادراکی فروشگاه

۲-۶ طرح معماری

در حالت کلی معماری نرم افزارهای شی ء گرا را در سه لایه : ارائه ، منطق عملیات و ذخیره سازی یا به عبارت دیگر سه لایه رابط، عملیات و بانک مشخص می کنند.

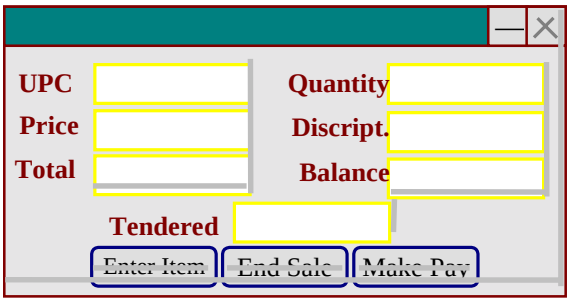
۱- لایه ارائه : شامل فرمهای ورودی / خروجی و ارتباط با محیط خارج است.

۲- لایه منطق عملیات : شامل بدنه اصلی و اشیاء تعریف شده در دامنه مساله مربوطه است . دو شی ء پرداخت و فروش در این لایه تعیین می شوند.

۲- لایه منطق کاربرد : شامل اشیاء سرویس دهنده است - سرویسهای پشتیبانی مثل ارتباط با DB . در این لایه رابط بانک اطلاعاتی و مدیریت امنیت سیستم مطرح می شود.

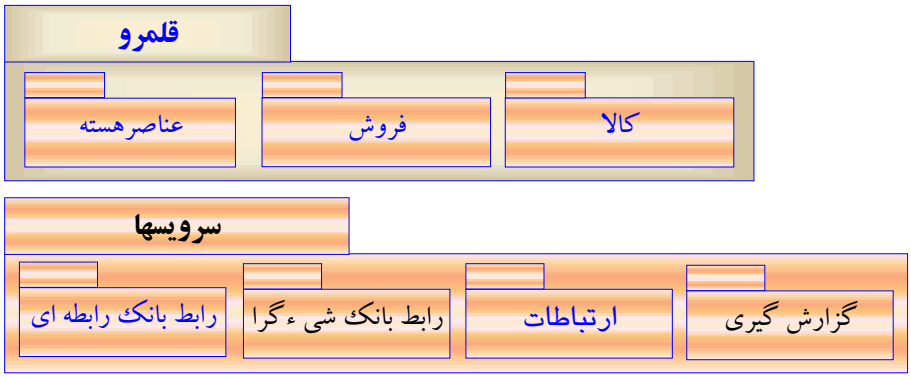
۳- لایه ذخیره سازی : شامل عناصر بانک اطلاعاتی است.

برای نمونه در شکل ۳-۲۲ لایه های معماری برای نرم افزار صندوق فروشگاه ، POST، ارائه شده است .



شکل ۳-۲۲ : دیدگاه کلاسیک معماری سه لایه

در سیستم صندوق فروشگاه لایه ارائه را کلاسی از نو اپلتهای جاوا پشتیبانی می کند. لایه منطق عملیات را دو کلاس فروش و پرداخت ، رابط بانک اطلاعاتی و مولد گزارش تکمیل می کنند. سیستم را بر اساس این سه لایه می توان به سه بسته یا Package اصلی تقسیم بندی نمود. براین اساس می توان سه بسته اصلی برای هر سیستم در نظر گرفت . هر بسته در ارتباط با عملیات یک لایه خواهد بود. در اینجا مساله چگونگی برقراری ارتباط در بین لایه ها است . معمولاً، دسته بندی عناصر در داخل یک بسته بر اساس میزان ارتباط بین عناصر است . عناصر داخل هر لایه فقط از طریق رابط آن لایه و به طور غیر مستقیم با لایه زیرین خود ارتباط برقرار می کنند. برای نمونه در شکل ۳-۲٤ دو لایه برای سیستم صندوق مشخص شده است :



شکل ۳-۲٤: لایه ها و افرازها

هربسته بعنوان یک بخش یا یک لایه در معماری سیستمها در نظرگرفتـه مـی شـود. ممکـن اسـت یک بسته ، کلاس یا بسته دیگری را دربر گیرد. در مثال فوق همه عناصر مدل ادراکی در بسته ای به نام قلمرو قرار گرفته اند. باید عناصری که در یک بسته قرار می گیرند در مجموع عمل و وظیفه مشـخصی را انجام دهند.

معماری لایه ای یک نوع خاص از معماری نرم افزار است. این نوع معماری بر اسـاس لایـه بنـدی نرم افزار ایجاد شده است. در معماری لایه ای هر لایه با استفاده از سرویسهای لایه زیرین خود ایجاد می شود. در ارتباط مستقیم با لایه زیرین قرار دارد و مستقل از لایه فوقانی ایجاد می شود. این نکته عمده ای است که باید در طرح این نوع معماری ها مد نظر داشت. باید استانداردی خاص و یکسان برای برقراری ارتباط در بین لایه ها ایجاد کرد، به قسمی که هر لایه با روشی مشخص و از طریق رابط مشخصی بتواند با لایه زیرین خود ارتباط برقرار کند.

باید توجه داشت که معماری سه لایه تنها معماری ممکن نیست. می توان در هر تعدادی از لایه ها نرم افزار را طراحی نمود. تکته اصلی در استقلال لایه زیرین از لایـه فوقـانی و ارائـه اسـتانداردی جـهت برقراری ارتباز بین یک لایه با لایه زیرین خود است. معماری خط لوله ، معماری تخته سـیاه و معمـاری براساس شبکه مرتبط از قطعه ها نیز از معماری های شناخته شده هستند که وابسته به نوع مساله می تـوان آنها را مورد استفاده قرار داد.

بسته ها ابزاری برای مدیریت و کنترل عملکرد نرم افزار هستند. بسته ها را می توان جهت نمـایش زیر سیستمها مشخص کرد. بسته ها را می توان برای دسته بندی موردهای استفاده، کلاسها و قطعه هـای تشکیل دهنده یک سیستم مورد استفاده قرار داد. در طرح جامع سیستمها چگونگی برقراری ارتبـاط در بین بسته ها از اهمیت ویژه ای برخوردار است.

در طرح سه لایه نرم افزار که در UML مطرح شده است می توان سه بسته مجزا برای سه لایه رابط کاربر، عملیات سیستم و لایه سرویس بانک اطلاعاتی در نظر گرفت. لایه رابط شامل کلاسهای رابط یا در اصطلاح کلاسهای سرحدی است. این کلاسها ابزاری برای نمایش فرمهای ورودی / خروجی و یا ابزاری جهت برقراری ارتباط با دستگاههای جانبی و محیط جانبی سیستم مکانیزه هستند. بسته عملیات باید در برگیرنده کلیه عملیات داخلی سیستم باشد. لایه سرویس بانک شامل کلاسهای لازم جهت برقراری ارتباط با سیستم بانک اطلاعاتی است و توسط لایه بانک اطلاعاتی پشتیبانی می شود.

شکل ۳-۲۵: طرح معماری سه لایه

- لغتنامه مدل

در لغت نامه مدل ، همه واژه های موردنیاز جهت ارتباطات بیشتر ودرک بهتر از مدل تعریف می شوند. این لغت نامه در فاز تحلیل بوجود می آید و درتمام سیکل های توسعه سیستم ،پالایش و اصلاح می شود . لغت نامه مدل ، در واقع سندی است که مشخصات اشیاء قلمرو محیط را تعریف می کند.در زیر این لغت نامه برای سیستم صندوق فروشگاه ارائه شده است.

توضیحات	دسته	واژه
توصيف فرايند خريد كالا توسط مشتري	Use case	Buyitem
يك شرح اجمالي از كالا	صفت	productspecification
كالا براي فروش در يك فروشگاه	نوع	Item
پرداخت پول	نوع	Payment
قيمت كالا	صفت	productspecification.price : quntity
تعداد يك نوع كالا	صفت	Salelineitem.quntity : integer
يك تراكنش فروش	نوع	Sale
يك خط كالا براي يك كالاي مخصوص در طول فروش	نوع	Salelineitem
جائي كه فروش كالا در آن اتفاق ميافتد	نوع	Store
كل پول يك فروش	صفت	SaleTotal : quntity
مقدار پول تقديميشده از مشتري براي پرداخت	صفت	Payment.amount : quntity
كد جهاني محصول	صفت	Productspecification.upc : UPC

۳-۷ سیستم عابر بانک

در این قسمت به عنوان یک مثال عملی بخشی از عملیات یک سیستم عابر بانک مطرح شده است. ماشین عابر بانک یا در اصطلاح ATM دارای یک کارت خوان مغناطیسی است . همچنین یک کنسول برای کاربر دارد. می توان از طریق یک روزنه پول در آن قرار داد و یا اینکه از حساب خود پول برداشت نمود. ماشین عابر بانک در ارتباط با کامپیوتر بانک نیز قرار دارد. مراحل عبارتند از:

۳-۷-۱ تعیین نیازمندیها

الف – دستگاه ATM می بایست سرویسهای زیر را فراهم نماید:

۱. دستگاه عابر بانک باید قادر به دریافت چک ، حواله و وجه نقد باشد.

۲. دستگاه باید بتوان با کامپیوتر مرکزی ارتباط برقرار کند.

۳. دستگاه باید قادر به نمایش جزئیات حساب مشتریان باشد.

٤. دستگاه باید قادر به راهنمایی صاحبان کارت باشد.

۵. دستگاه باید قادر به ثبت درخواستها و تراکنشها باشد.

۱.دستگاه باید برای افراد رسید صادر نماید.

ب- صاحب کارت عابر بانک باید قادر به عملیات زیر باشد

۱. صاحبان حساب باید قادر به برداشت وجه به صورت ضرایبی از ۱۰۰۰ تومان از هر دستگاه متعلق به بانک باشند. قبل از اینکه بتوان پول را از دستگاه دریافت نمود، باید سیستم بانک مربوطه برداشت پول را تائید نماید.
۲. صاحب حساب باید قادر به ذخیره و پس انداز هر گونه وجهی به صورت چک ، حواله یا وجه نقد در داخل دستگاه عابر بانک باشد.
۳. مشتری باید قادر به انتقال پول در بین هر دو حساب خود که از طریق یک کارت قابل دسترس است باشد.
٤. صاحب حساب باید قادر به مشاهده تراز حساب خود باشد

۳-۷-۲ آنالیز

مستندات تحلیل یا آنالیز سیستم عبارتند از شرح موارد استفده از سیستم ، کلاسهای حاصل از آنالیز و دیاگرام حالت یا در اصطلاح State Chart. در ادامه مستندات مرحله آنالیز برای سیستم عابر بانک مشخص شده است . در مثال زیر منظور از PIN شماره یا کد کارت عابر بانک است . توجه نمائید که ابتدا جهت درک بهتر نیازهای سیستم که در بالا لیست شدند باید موردهای استفاده از سیستم مشخص شوند. در واقع موردهای استفاده مشخص می کنند که سیستم چه استفاده ای از نیازهای اعلان شده خود می برد. موردهای استفاده در شکل ۳-۲٦ مشخص شده اند. در ادامه شش مورد استفاده اجمالا" توضیح داده شده اند.

۱- راه اندازی

هنگامیکه اپراتور دستگاه را روشن می کند مقداری وجه در دستگاه قرار می دهد. و برای سیستم مشخص می کند که چقدر وجه اولیه قرار داده شده است. به این ترتیب سرویس دهی به مشتری آغاز می شود.

۲- اختتام

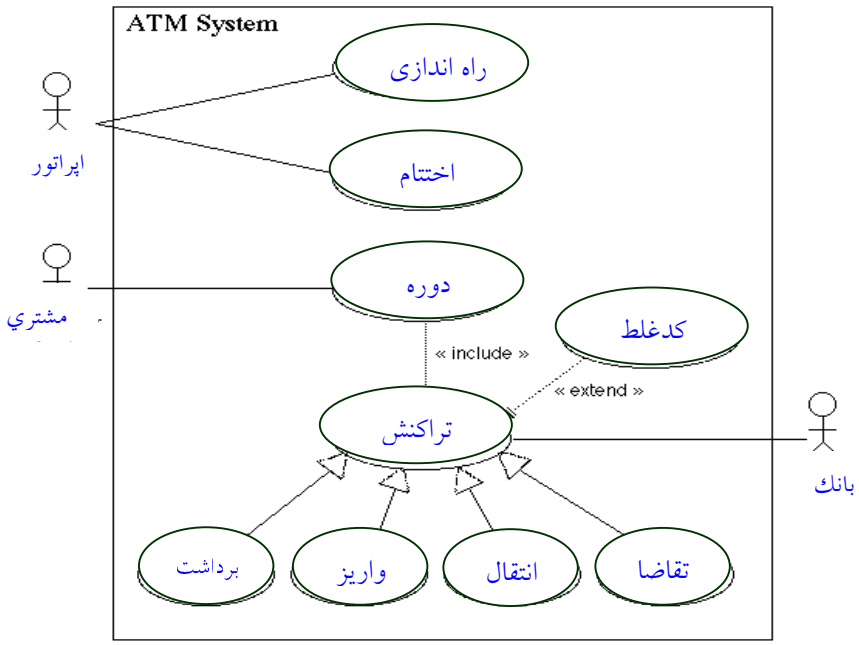
با پیچاندن دکمه ماشین عابر بانک در مکان خاموش دستگاه عابر بانک متوقف می شود. قبل از آن باید میزان وجه باقی مانده در دستگاه و نوار مبالغ برداشته شده از داخل دستگاه گرفته شود. به این ترتیب وجه درون دستگاه و مبالغ برداشتی یکبار قبل از خاموش کردن دشتگاه کنترل می شوند.

۳. مورد دوره

یک دوره عملیات یا در اصطلاح یک Session هنگامی آغاز می شـود کـه مشـتری کـارت عـابر بانک خود را درون دستگاه قرار می دهد.. سپس از Customer دستگاه تقاضای شـماره کـارت یـا Pin Number وی را می نماید. با انتخاب نوع Transaction مشتری می تواند کار خود را انجـام دهـد. دوره عملیات با ورود شماره کارت آغاز می شود و با فشردن دکمه خروج خاتمه می یابد.

٤. موردهای برداشت، واریز، انتقال و تقاضا

در یک دوره کار با دستگاه عابر بانک ، مشتری می توانـد بـا تعیـین نـوع حسـاب خـود اقـدام بـه برداشت پول نماید. برداشت با تعیین مبلغ آغاز مـی شـود و در صـورت وجـود وجـه سیسـتم پـول را در اختیار مشتری قرار داده ، رسید Receipt صادر می نماید.

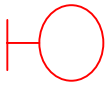


شکل ۳-۲٦: دیاگرام موردهای استفاده از سیستم عابر بانک

از متن توصیف موارد استفاده اسامی کلاسها را می توان استخراج نمود. به این ترتیب کلاسهای زیر مشخص می شوند:

۱. اجزاء تشکیل دهنده دستگاه ATM در قالب کلاسهای سرحدی کـه در واقـع جزئیـات سـخت افزار را از درون سیستم می پوشانند مشخص می شوند:

- کارت خوان
- پایانه مشتریها
- دستگاه عابر بانک
- قبول کننده پاکت وجه
- چاپگر رسید
- پنل اپراتور
- ارتباط با بانک



علامت اختصاری

کلاس سرحدی

۲. کلاسهای کنترلی (با علامت اختصاری)که برای کنترل عملیات هرمورد استفاده از سیستم مشخص شده اند:

- دوره
- برداشت وجه
- واریزوجه
- انتقال وجه
- تقاضای تراز

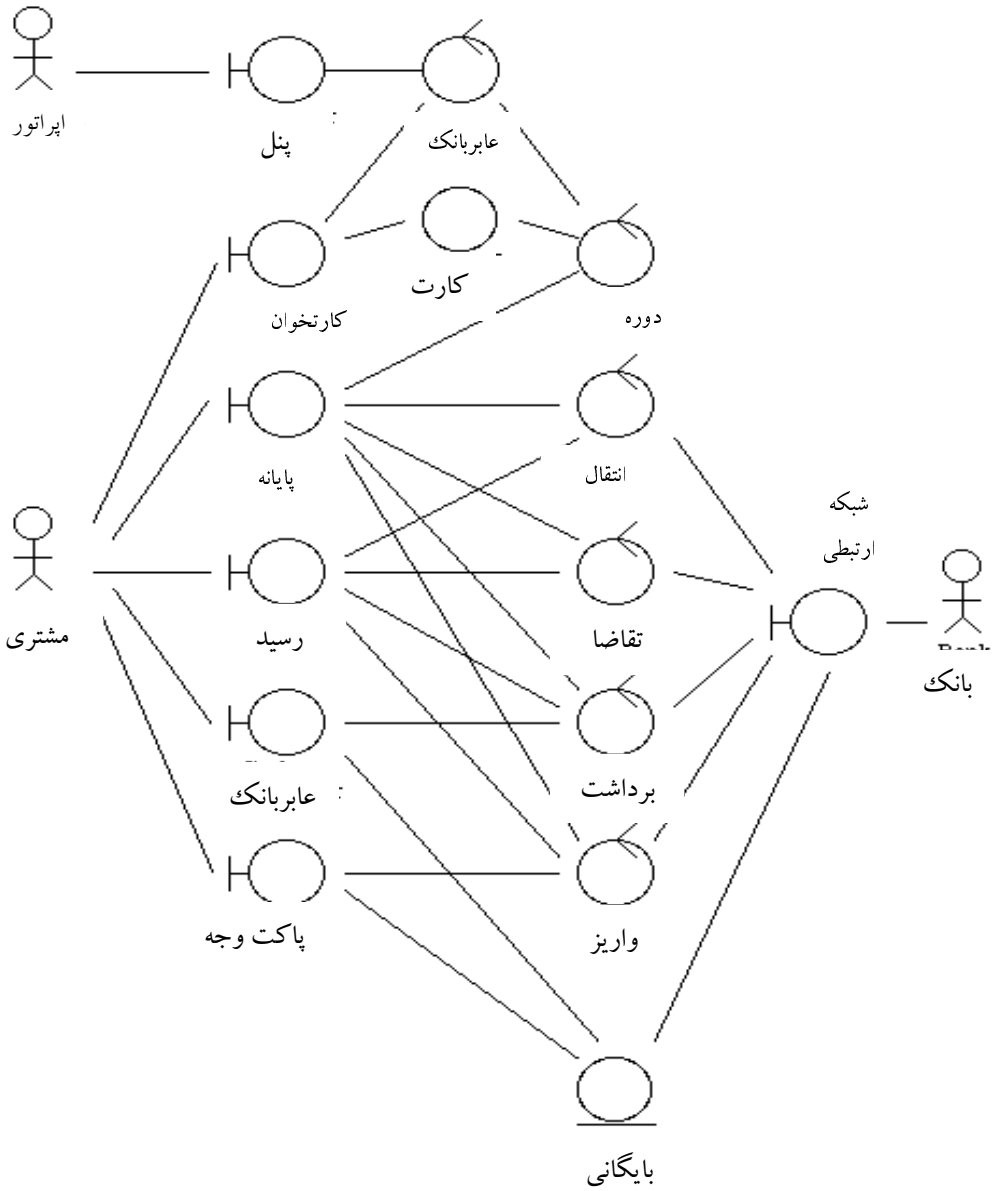
علامت اختصاری

کلاس سرحدی

۳. کلاسهای موجودیتی برای ذخیره سازی اطلاعات.

- کارت
- تراکنش

به ابن ترتیب دیاگرام آنالیز کلاسها مطابق شکل ۳-۲۵ استخراج می شود. هر یک از موارد استفاده ای که برای سیستم مشخص شد را می توان توسط دیاگرامهای همکاری بین اشیاء به شرح زیر مشخص نمود. پس از تعیین دیاگرامهای همکاری لیست مسئولیتها برای هر کلاس تهیه می شود. سپس، برای هر کلاس مسئولیتها لیست می شوند. و بر این اساس مدل ارتباطی کلاسها ترسیم می شود.



شکل ۳-۲۵: مدل ادراکی برای سیستم عابر بانک

فصل چهارم
مرحله طراحی

۴-۱ مقدمه

در مرحله تجزیه و تحلیل هدف شناخت نیازها ، مفاهیم و عملیات مر بوطه است. در مرحله طراحی هدف ارائه راه حل منطقی برای نیازهای شناسایی شده در مرحله شناخت است. در مرحله شناخت جهت تکمیل شناخت پاسخ به سوالهی زیر باید فراهم می شد:

۱- ویژگیهای سیستم ؟ (نیازهای کیفی و کمی)

۲- عملکرد سیستم ؟ (موردهای استفاده)

۳- مفاهیم و کلاسهای اصلی ؟ (مدل ادراکی)

٤- رویدادها و عملیات ؟ (دیاگرامهای همیاری)

٥- مفهوم عملیات ؟ (قراردادها)

در مرحله شناخت "چه" و در مرحله طراحی "چگونه" مطرح است. ابزار اصلی طراحی، دیاکرامهای همیاری میباشند. این دیاگرامها در این فصل مورد بررسی قرار خواهند گرفت. الگوهای طراحی امروزه ابزاری برای بکارگیری و استفاده مجدد از طراحی های قبلی است. در مرحله طراحی باید ابتدا کلاسها و مسئولیت کلاسها مشخص شود. بر مبنای مسئولیت واگذار شده به هر کلاس عملیات موردهای استفاده در بین کلاسها تقسیم می شود. در واقع از همکاری بین اشیاء است که موردهای استفاده ، به اجرا در می آیند. مراحل طراحی شامل :

۱. تعیین موردهای استفاده واقعی

۲. تعیین مسئولیتها

۳. تعیین معماری سیستم

٤. تعیین دیاگرامهای همیاری

٥. تعیین دیاگرام ارتباطی کلاسها

۴-۲ موردهای استفاده واقعی

موردهای استفاده در مرحله تحلیل جهت شناخت عملیات مورد نیاز و انتظارهای کاربر از سیستم مکانیزه مشخص شدند. نیازهای واقعی مبین طراحی موردهای استفاده از دیدگاه سیستم مکانیزه است. در واقع در موردهای استفاده واقعی اسامی کلاسها و متدهای مورد نیاز در متن سناریوها ظاهر شده، ورودی / خروجیها مورد تاکید قرار می گیرند. بنابر این طراح در این مرحله رابطهای کاربر را مشخص می کند. برای نمونه در اتباط با سیستم صندوق فروشگاه مورد خرید اقلام را در نظر بگیرید. این مورد در مرحله شناخت در شکل ۲-۱۰ توضیح داده شد. در این قسمت باید به طور دقیقتری در متن سناریو اسامی اشیاء و متدها را مشخص نموده، رابط کاربر را تصویر نمود.

مورد استفاده :	خرید اجناس
بازیگرها	مشتری ،صندوقدار
نوع	اصلی
هدف	ثبت یک فروش ویک پرداخت
شرح	مشتری به صندوق نزدیک می شود و تقاضای خرید اجناس را با قرار دادن اقلام خرید در مقابل صندوقدار، می نماید. صندوقدار اجناس مورد خرید را ثبت و رسید به مشتری می دهد. با اتمام کار، مشتری صندوق را ترک می کند
نیازهای مرجع	R2.4 ç R2.3ç R2.1ç R1.9ç R1.7ç R1.3ç R1.2 çR1.1

رخدادها

واکنش سیستم	عمل بازیگر
	۱-مشتری کالای مورد خریداری را به

صندوقدار می دهد.

۳- سیستم تراکنش فروش جاری را برای هر قلم کالا بلافاصله با تعیین کد کالا ایجادمی کند. قیمت و کد کالا بر روی فرم مطابق شکل فوق ظاهر می شود.

۵-سیستم کل مبلغ فروش را محاسبه و نمایش می دهد

۲-صندوقدار بارکد کالا را با استفاده اسکن نموده ، اکر تعداد مورد در خواست از یک قلم کالا بیش از یک باشد،صندوقدار تعداد آنها را نیر وارد می کند.

٤-برای تکمیل ورود کالاها ،صندوقدار با فشار دکمه "جمع کل" مطابق شکل عمل می کند.

٦-صندوقدار به مشتری جمع مبلغ

شکل ٤-١ : مورد استفاده واقعی از دیدگاه طراحی منطقی

۴-۲ دیاگرامهای توالی و همکاری

در دیدگاه شیء گرا وظایف از همکاری بین اشیاء به انجام می رسند. وظایف در قالب موردهای استفاده مشخص شدند. موردهای استفاده پیچیده در قالب سناریوها و به صورت دنباله ای از رویدادها مشخص شدند. با استفاده از دیاگرامهای همیاری می توان سناریوها را به صورت دنباله ای از پیامهای مبادله شونده بین اشیاء مشخص نمود. پیامها در واقع جملات فراخوانی هستند. فراخوانی برای متدها. لذا، دیاگرامهای همیاری ایزاری جهت نمایش دنباله ای از جملات فراخوانی، شیء فراخواننده و شیء فراخوانده شده در جهت انجام وظایف یک مورد استفاده مشخص هستند. دیاگرامهای همیاری به دو دسته دیاگرامهای توالی و همکاری تقسیم بندی می شوند. در محیط رَشنال رُز، با وجود دیاگرام توالی می توان دیاگرام همکاری را بصورت اتوماتیک ایجاد نمود.

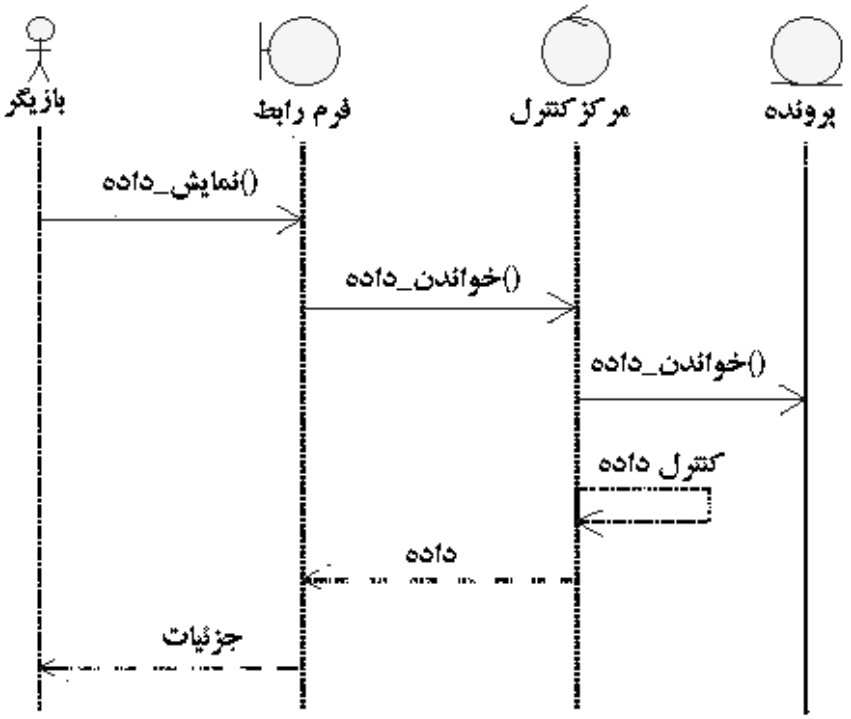
۴-۲-۱ فرم کلی دیاگرامهای توالی

دیاگرامهای توالی نمایانگر دنباله ای از عملیات جهت ارائه سرویس مورد نظر به استفاده کننده هستند. لذا، برای برقراری ارتباط با استفاده کننده معمولا" یک کلاس از نوع سرحدی[۳۳] برای برقراری ارتباط با کاربر و نمایش فرمهای ورودی / خروجی در نظر گرفته می شود. در زبانهای برنامه سازی مثل دلفی و امثال آن نیز فرمهای ورودی خروجی به عنوان یک کلاس مطرح هستند. اما، جهت کنترل عملیات نیز یک کلاس از نوع کنترلی در نظر گرفته می شود. این نوع کلاسها در واقع فقط تایید کننده

---

[۳۳] Boundary Class

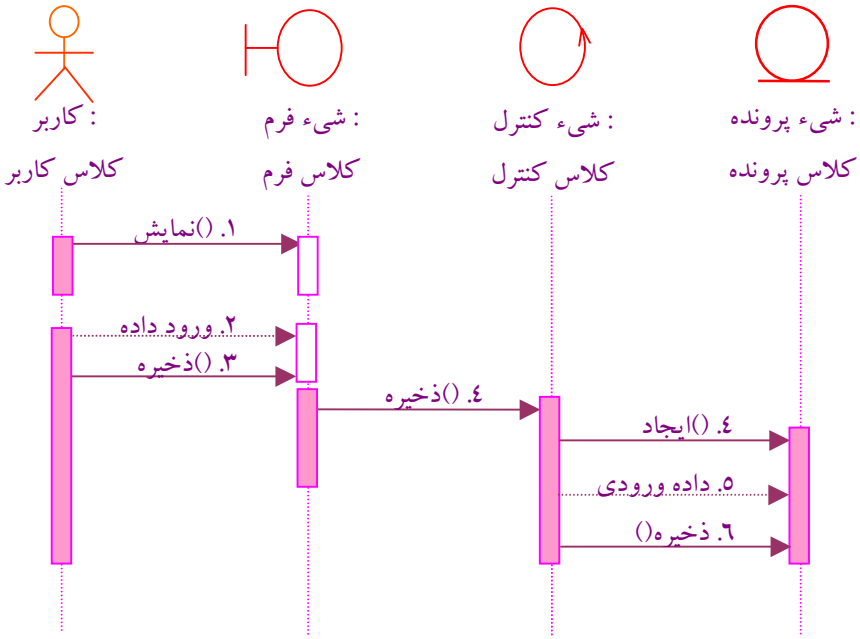
عملیات هستند و از خود هیچگونه عملی را موجب نمی شوند. جهت انجام هر عملی سایر اشیاء با این شیءابتدا تماس برقرار می کنند و این شیء در صورت تایید فرمان را به سایر اشیاء انتقال می دهد.



شکل ٤-٢ : ساختار یک دیاگرام توالی در فرم کلی

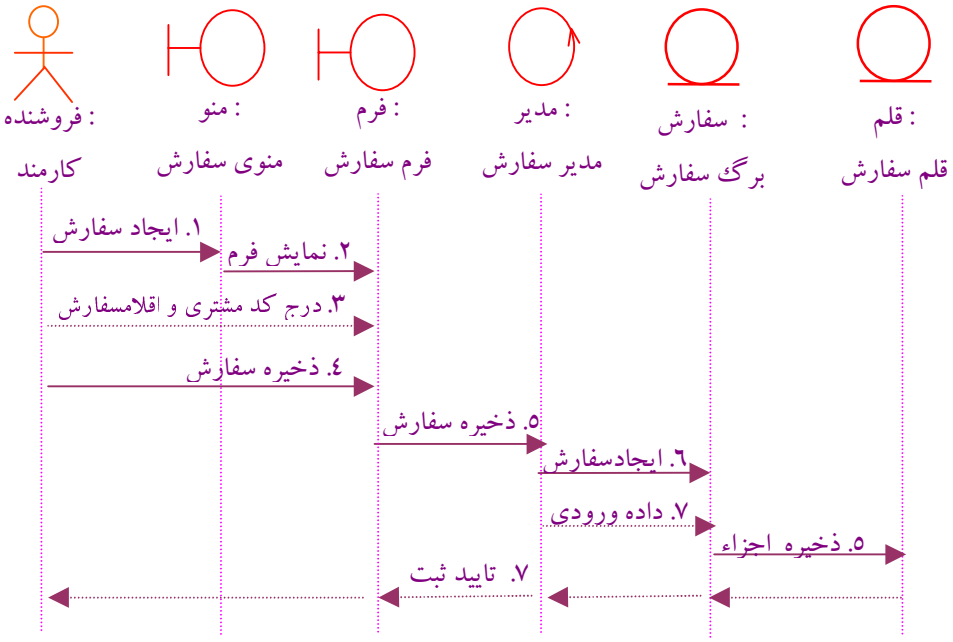
در شکل ٤-٢ یک نمونه از دیاگرام توالی برای خواندن داده از یک پرونده و نمایش آن بر روی فرم ارائه شد است. در این دیاگرام با فراخوانی متد "()نمایش_داده" از شیء فرم رابط ، مورد استفاده آغاز می شود. مرکز کنترل تنها عملیات کنترلی برای خواندن داده ها را اعمال می کند. در شکل فوق خطوط خط چین نمایانگر انتقال داده ها هستند.

در شکل ٤-٣ دیاگرام توالی برای عملیات ثبت اطلاعات درون یک پرونده مشخص شده است. دراین دیاگرام با فراخوانی متد "()نمایش" از شیء "فرم رابط" کار خواندن جزئیات از پرونده آغاز می شود. خواندن داده تحت نظارت شیء "مرکز کنترل" انجام می شود.



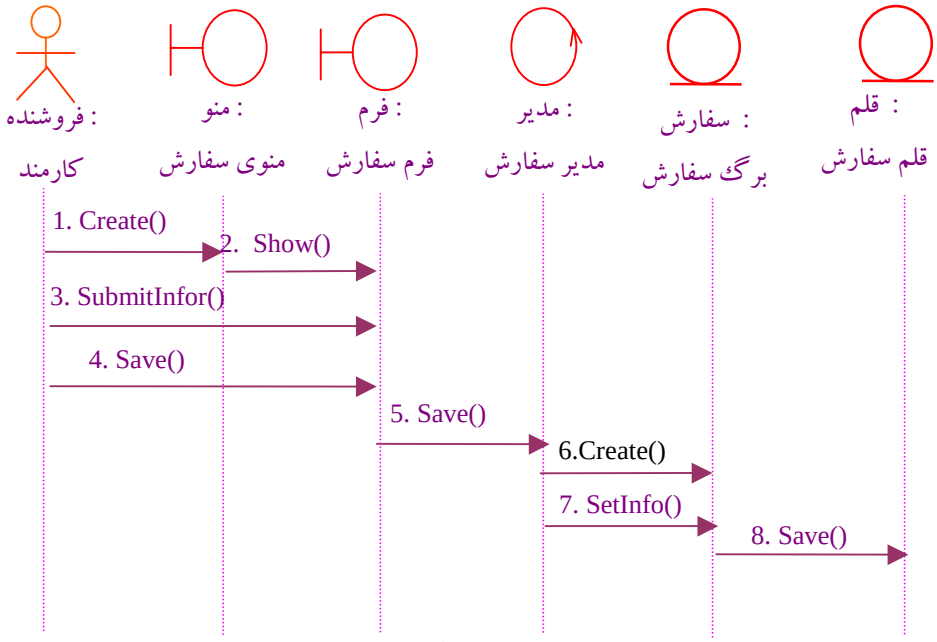
شکل ٤-٣: دیاگرام توالی برای ذخیره کردن داده درون یک پرونده

معمولاً، دیاگرام توالی در دو مرحله برای یک مورد استفاده ترسیم می شوند. در مرحله اول دیاکرام تولی دقیقاً مطابق با سناریو ایجاد می شود. یعنی هر خط رابط افقی مطابق با یک بند سناریو و به صورت یک عبارت مشخص می شود.



شکل ٤-٤: دیاگرام اولیه توالی

در مرحله دوم دیاگرام اولیه، تبدیل به مدل برنامه می شود. در این مدل بر روی خطوط افقی در عوض پیامها ، جمله های فراخوانی ظاهر می شود. در بالای خطوط نام شیء و کلاس آن ظاهر می شود. دیاگرام ایجاد شده در مرحله دوم در واقع مورد استفاده را در قالب دنباله ای از جملات فراخوانی برای متدهای اشیاء برنامه ریزی می کند. این دیاگرام بیشتر برای طراح قابل استفاده است و از دیدگاه کاربر و متقاضی سیستم نا مفهوم است. در شکل ٤-٥ ، در حالت کلی برای ذخیره سازی داده ها در داخل یک پرونده ، دیاگرام توالی ترسیم شده است.

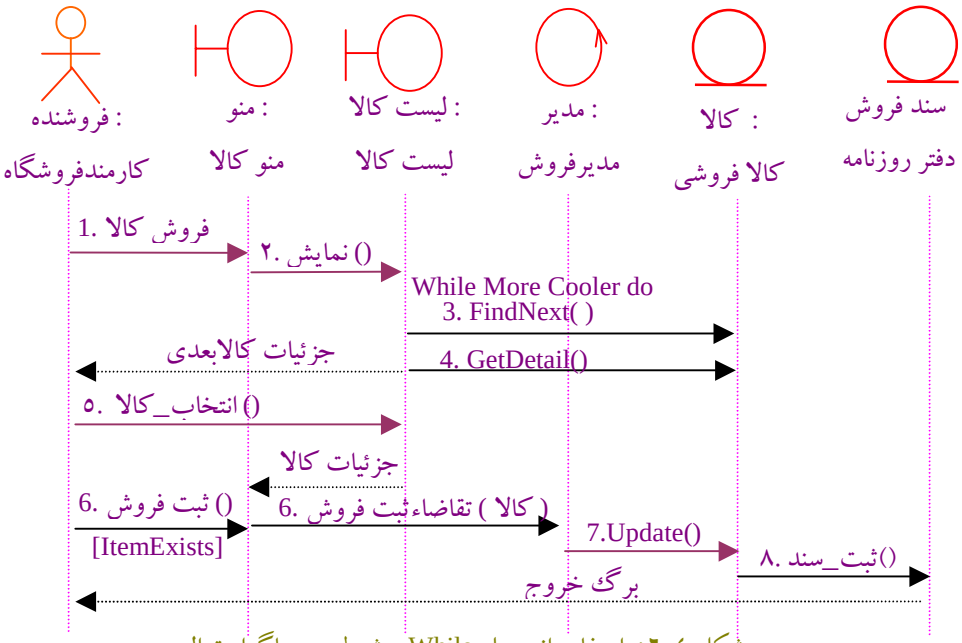


شکل ٤-٥ : دیاگرام ثانویه توالی

۴-۲-۲ دیاگرامهای توالی و همکاری

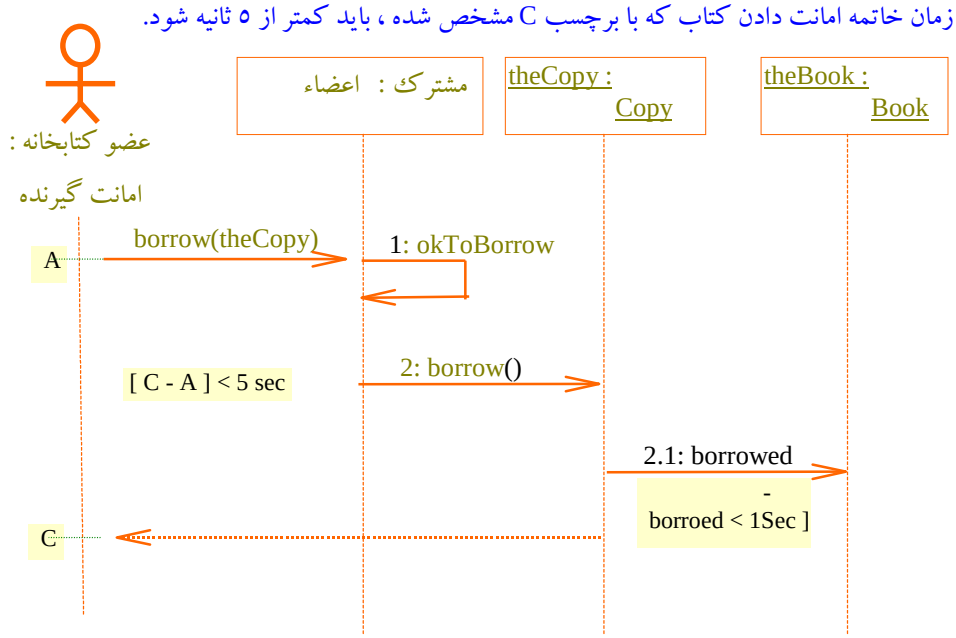
دیاگرامهای توالی یک دنباله پیوسته از رویدادها در جهت انجام یک مورد استفاده را تصویر می نمایند. هر رویداد در قالب یک پیام نمایش داده می شود. پیامها در نهایت تبدیل به جملات فراخوانی می شوند. هر جمله فراخوانی در واقع بیانگر فراخوانی متدی از یک شیء است. اسامی اشیاء و کلاس آنها بر روی خطوط عمودی در دیاگرام توالی مشخص می شود. ممکن است متدی مکررا" فراخوانی شود. تکرار را با از جمله While می توان در داخل دیاگرام مشخص نمود. مقدار برگشتی از جملات فراخوانی را نیز می توان در داخل دیاگرام توالی مشخص کرد. ممکن است یکپیام یا در واقع یک فراخوانی تحت شرایط صورت گیرد. در این حالت شرط را در داخل براکت

قرار می دهند. برای نمونه به شکل ٤-٦ توجه نمایید. توالی ابتدا مراحل زیر انجام می شود:



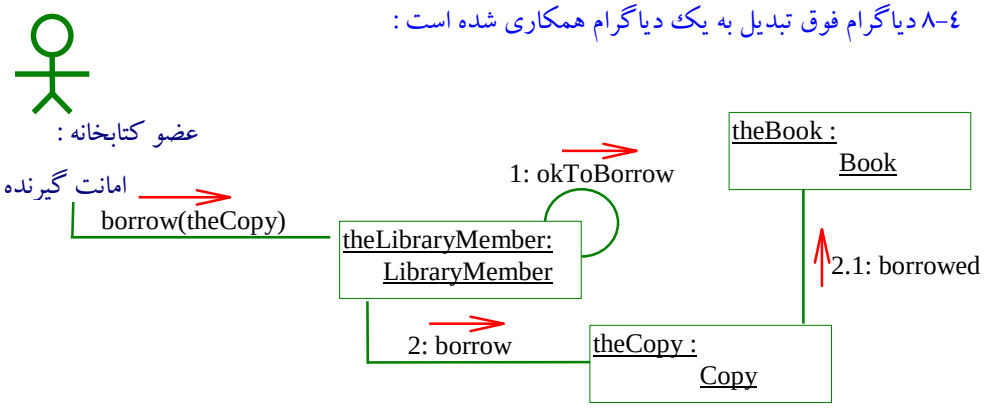
شکل ٤-٦ : استفاده از جمله While و شرط در دیاگرام توالی

در شکل ٤-٧ مشخص شده است که فاصله زمانی بین تقاضای امانت گرفتن کتاب در نقطـه A تـا زمان خاتمه امانت دادن کتاب که با برچسب C مشخص شده ، باید کمتر از ٥ ثانیه شود.



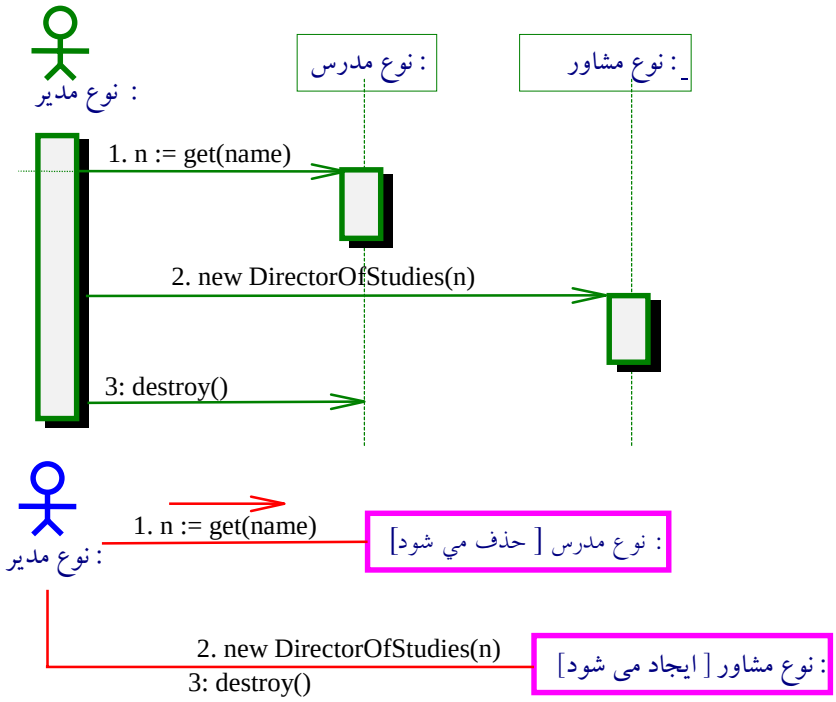
شکل ٤-٧: فعل و انفعال مصور در یک دیاگرام والی

مزیت دیاگرامهای توالی نمایش توالی زمانی است هر گونه شرایطی برای ارسال پیام بین دو شیء را می توان در دیاگرام توالی مشخص نمود. در شکل فوق متقاضی بیش از ٥ ثانیه نبایستی که منتظر بماند و اینکه زمان ارسال پیام ۲،۱ یعنی اولین پیام در ارتباط با پیام ۲ حداکثر یک ثانیه است . در شکل ٤-٨ دیاگرام فوق تبدیل به یک دیاگرام همکاری شده است :



شکل ٤-٨ : دیاگرام همکاری

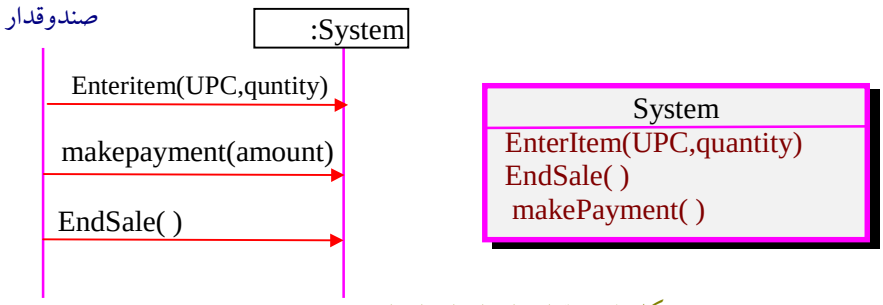
در شکل ٤-٩ مقدار برگشتی از تابع مشخص شده است. در این شکل مدیر دانشکده یک عضو هیئت علمی را از سمت مدرس حذف و به عنوان مشاور درسی تعیین می کند.



شکل ٤-٩ : دیاگرام توالی و دیاگرام همکاری مشابه آن

۴-۲-۳ قراردادها

برای تکمیل دیاگرامهای توالی، باید عملکرد توابع مورد فراخوانی در این دیاگرامها را تشریح نمود. به این ترتیب با مشخص نمودن این توابع نه تنها مفهوم دیاگرامهای توالی بلکه مفهوم کلاسها نیز تکمیل می شود. برای توصیف عملکد متدها از ساختاری تحت عنوان قرار داد استفاده می شود.



شکل ٤-٥: قرار دادها برای انجام مورد خرید

بطور دقیقتر، قرارداد وورود کالا در موردخرید به صورت زیر مشخص می شود:

نام قرارداد : (enterItem(UPC,quantity).

نوع : سیستمی

ارجاعات : توابع عملیاتی R1.1 ، R1.3 و R1.9

موضوع : - دریافت کد یک کالا و نمایش توصیف و قیمت آن کالا
- درج سابقه فروش کلا مشخص شده در پروند فروش

حالت استثنایی : - چنانچه کد کالا مشخص نباشد ، باید اعلام خطا کند.

شرایط اولیه: - قیمت واحد کالا باید مشخص باشد.

شرایط بعدی :
- سابقه فروش هر قلم کالای فروخته شده ثبت شده است.
- در پرونده صندوق نیز سابقه فروش قلم کالا درج شده است.
- قلم فروخته شده از موجودی کسر شده است..

منظور از شرایط اولیه ، پیش فرضها و شرایط لازم برای اجراء یک قرار داد است . شرایط بعدی شاخص وضعیت پیش بینی شده حاصل از اجراء صحیح قرار داد هستند برای ساخت دیاگرام همکاری مراحل زیر اجرا می شود:

۱. برای هر مورد استفاده در صورت لزوم در هنگام طراحی ،یک دیاگرام مجزا ایجاد می شود.
۲. اگر دیاگرام پیچیده است ، باید به دیاگرامهای کوچکتر تقسیم شود.

۴-۳ مدلسازی رفتاری

رفتار را با تعیین واکنش در مقابل رویدادها مشخص می کنند. موجود با شعور در مقابل رویدادها واکنش از خود نشان میدهد. لذا، با تعیین واکنش اشیاء در مقابل رویدادها برای آنها شعور و در واقع مرکز تصمیم گیری عملیات ایجاد می نمایند. در اصطلاح شیء گرایی قطعات[۳٤] در واقع کلاسهای با شعور هستند. زیرا برای آنها واکنش در مقابل رویدادها نیز مشخص می شود. رویدادها به دو دسته تقسیم می شوند. یک دسته از رویداده توسط سیستم مشخص می شوند. دسته دیگر توسط برنامه نویس برای قطعه مورد نظر وی مشخص می شود.

شکل ٤-۱۰ : رویدادهای تعیین شده در دلفی برای یک گلاس دکمه فشاری[۳۵]

در شکل ٤-۱۰ برای نمونه ، رویدادهای استاندارد برای یک کلاس دکمه فشاری مشخص شده است. این رویدادها توسط کامپایلر زبان شناخته شده می باشندو دسته ای دیگر از رویدادها توسط برنامه

---

[۳٤] Components

[۳۵] PushButton

نویس ممکن است مشخص شوند. برای نمونه رویدادی مثل OnHighPresure ممکن است در شرایط که فشار دیگ بخار از حدی بالاتر رفته باشد از طریقه یک وقفه به اطلاع برنامه برسد. در این صورت با وقوع این رویداد بلافاصله اشیاء مورد نظر در داخل برنامه باید واکنش نمایند. واکنششیء با فراخوانی اتوماتیکشیء در مقابل آن رویداد تعیین می شود. برای نمونه، در دلفی با کلیل کردن دکمه ماس بر روی هر یک از رویدادهای فوق برای دکمه فشاری متدی به صورت زیر به برنامه اضافه می شود:

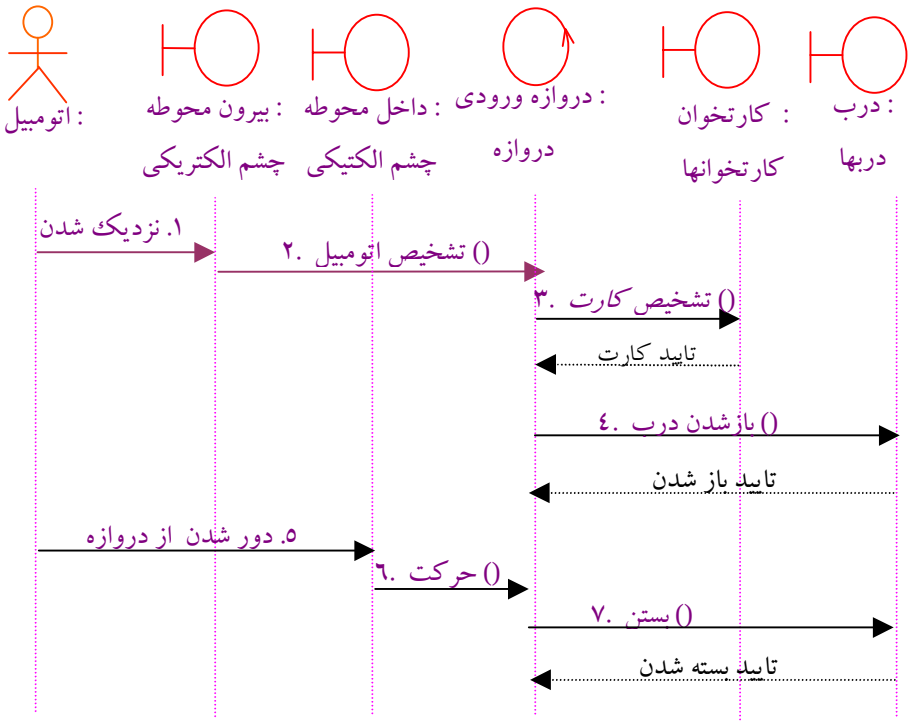
```
Procedure TForm1.Button1Click(Sender: TObject
  Begin

  End
```

به این ترتیب می تواند برای شی، ، دکمه فشاری با نام Button1 شعور و قوه تصمیم گیری مشخص کند . برای این منظور باید واکنش شیء در مقابل رویداد را با یک متد برای آن مشخص کند.. قطعه سازی به صورت استاندارد در کلیه زبانهای تصویری تحت قالب ActiveX رایج است.

باید تحلیلگر قادر به تعیین رویدادها و واکنش اشیاء در مقابل رویدادها باشد، تا بتواند کلاسها را به صورت با شعور و در قالب قطعات پیاده سازی نماید. به این ترتیب می توان از مزایای قطعات در برنامه ها استفاده نموده ، اصول برنامه سازی بر اساس رویدادها را رعایت نمود. برای این منظور دیاگرامهای حالت و فعالیت مورد استفاده قرار می گیرند.

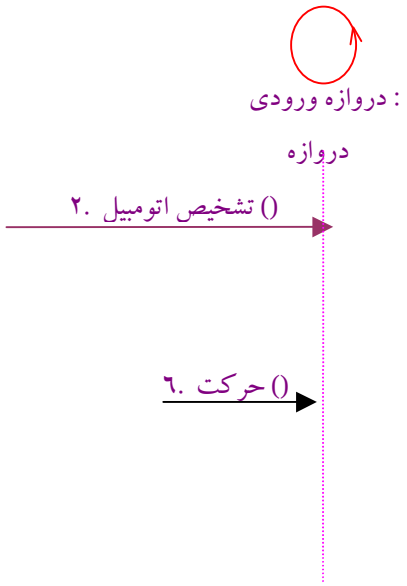
۴-۳-۱ دیاگرام حالت

رفتار را می توان به صورت واکنش در مقابل رویدادها مشخص نمود. هر رویداد موثر موجب تغییر حالت شیء از حالت کنونی به حالت دیگر می شود. لذا، برای مدلسازی رفتار باید رویدادها و تغییر حالت اشیاء را بواسطه رویدادها مشخص کرد. دیاگرام حالت در واقع یک گراف است که گره های آنرا حالات مختلف یک شیء مشخص می کنند. رویدادها موجب گذر بین حالات و به صورت برچسب کمانهای گراف مشخص می شوند بنابر این برای تعیین رفتار اشیاء باید رویدادهای موثر بر شیء و حالتهای شیء را مشخص نمود. برای این منظور به دیاگرامهای مختلف توالی که در آنها شیء مورد نظر قرار دارد باید رجوع نمود. هر رویداد وارده بر شیء در دیاگرام توالی یک تغییر حالت را برای آنشیء موجب می شود. لذا، فاصله بین هر دو رویداد متوالی وارده بر یک شیء را می توان یک حالت شیء در نظر گرفت. برای نمونه به دیاگرام توالی ارائه شده در شکل ٤-١١ توجه نمائید.



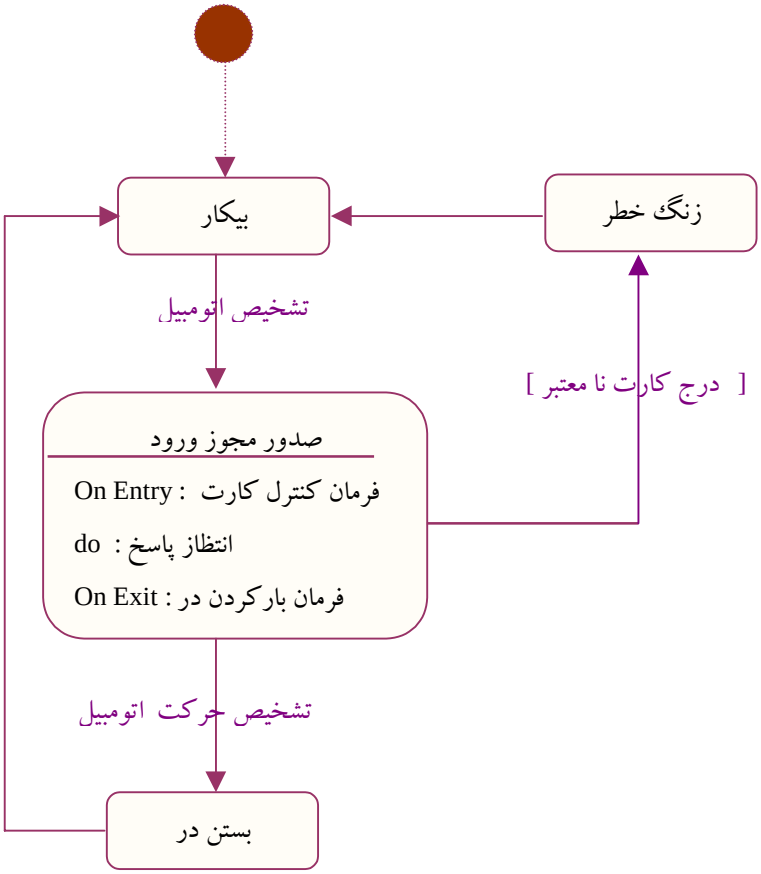
شکل ٤-١١ : سیستم کنترل درب ورودی ساختمان

دیاگرام حالت برای کلاس دروازه در شکل فوق را می توان با در نظر گرفتن رویدادهای ورودی به این کلاس مطابق شکل ٤-١٢ ایجاد نمود.



شکل ٤-١٢ : رویدادهای واردهبر کلاس دروازه

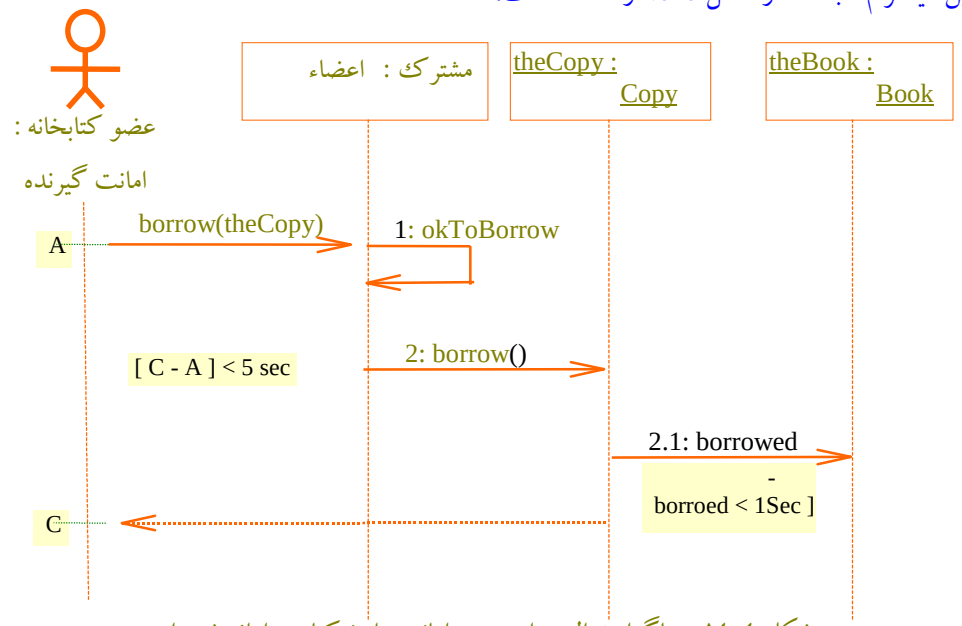
اکنون با در نظر گرفتن هر رویداد وارد شونده به کلاس دروازه ، به عنوان یک عامل تغییر حـالت می توان دیاگرام حالت را برای این کلاس به صورت زیر مشخص نمود.



شکل ٤-١٣ دیاگرام حالت برای کلاس دروازه

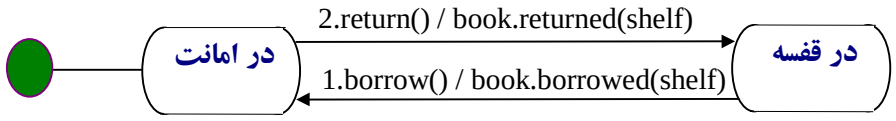
همانگونه که در شکل فوق مشخص شده است برای یک حالت سه نوع فغالیت در نظر گرفته شده است. با ورود به حالت "صدور مجوز ورود" بلافاصله فرمان کنترل کارت صادر می شـود. در هنگـامی که شیء درون حات قرار دارد در انتظار پاسخ کارتخوان است. با خروج از حالت فرمان باز کردن درب ورودی صادر می شود. لغات On Entry ، Do و OnExit به ترتیب برای تعیین فعالیت در هنگام ورود، در صمن گذراندن و در هنگام خروج از حالت مشخص می شود. استفاده از این لغات کلیدی اختیـاری است.

ورود به یک حالت نیز ممکن است با فعالیت خاص انجام شود و همچنین برای ورود به حالت ممکن است شرایطی لازم باشد. برای نمونه دیاگرام توالی برای به امانت گرفتن کتاب را در نظر بگیرید. این دیاگرام مجدداً در شکل ٤-١٤ ارائه شده است.



شکل ٤-١٤ : دیاگرام توالی برای مورد امانت دادن کتاب دارائه شده است.

بر ظبق شکل ٤-١٤ تنها یک ر.ویداد Borrow() بر شیء امانت وارد می شود. با در نظر گرفتن اینکه در حالت عادی نسخه کتاب در قفسه کتابها است ، دیاگرام حالت برای کلاس Copy به صورت زیر خواهد بود.



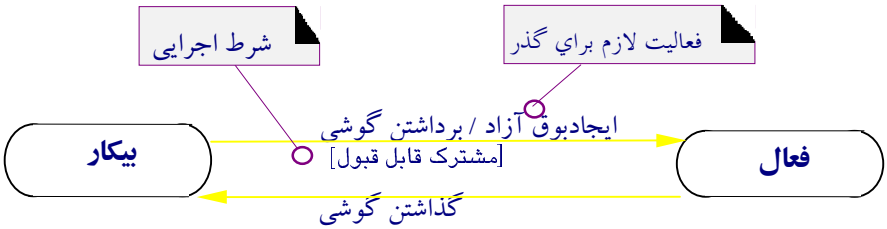
شکل ٤-١٥ : دیاگرام حالت برای کلاس Copy

در شکل ٤-١٥ ، مورد برگشت دادن کتاب به کتابخانه نیز مشخص شده است. فعالیت لازم برای انجام عمل امانت و در واقع فعالیت لازم برای گذر کتاب از حالت "در قفسه" به حالت "در امانت"، طبق شکل ٤-١٤ انجام فعالیت Borrowed بر روی کلاس Book است. لذا، این فعالیت جهت تغییر حالت کلاس مشخص شده است. در حالت کلی یک رویداد ممکن است تحت شرایط خاص و با انجام یک فعالیت بتواند موجب تغییر حالت یک کلاس بشود. بنابر این در حالت کلی کمانهای گراف و یا به عبارت دیگر گذر بین حالات در گراف دیاگرام حالت به صورت زیر است:

فعالیت شرط اجرایی لیست پارامترهای رویداد) نام _رویداد

در شکل زیر علامت مورد استفاده برای نمایش یک حالت در UML مشخص شده است.

در شکل ٤-١٠ ، شرط فعال شدن تلفن ، قابل قبول بودن مشترک برای مکالمه تلفنی است .



شکل ٤-١٠: دیاکرام حالت برای خط تلفن

دیاگرامهای حالت را برای بیان رفتار موردهای استفاده نیز می توان به عنوان یک بزار مدلسازی مورد استفاده قرار داد. این دیاگرامها، توالی زمانی در وقوع رویدادها را نیز مشخص می کنند. این توالی در متن سناریوها نیز مشخص بود. انواع رخدادهایی که در یک دیاگرام حالت می توان مشخص نمود شامل :

١. رخدادهای خارجی

این نوع رخداد به رخدادسیستمی معروف است و بوسیله عامل های خارج از حوزه سیستم تاثیر می گذارد(مانند بازیگرها). دیاگرامهای توالی ، رخدادهای خارجی را به نمایش می گذارند. رخدادهای خارجی در فراخوانی عملیات سیستم بکار می روند.. برای مثال وقتی که صندوقدار کلید ورود کالا را در برنامه صندوق فروشگاه فشار می دهد، یک رخداد خارجی اتفاق افتاده است .

٢. رخداد داخلی

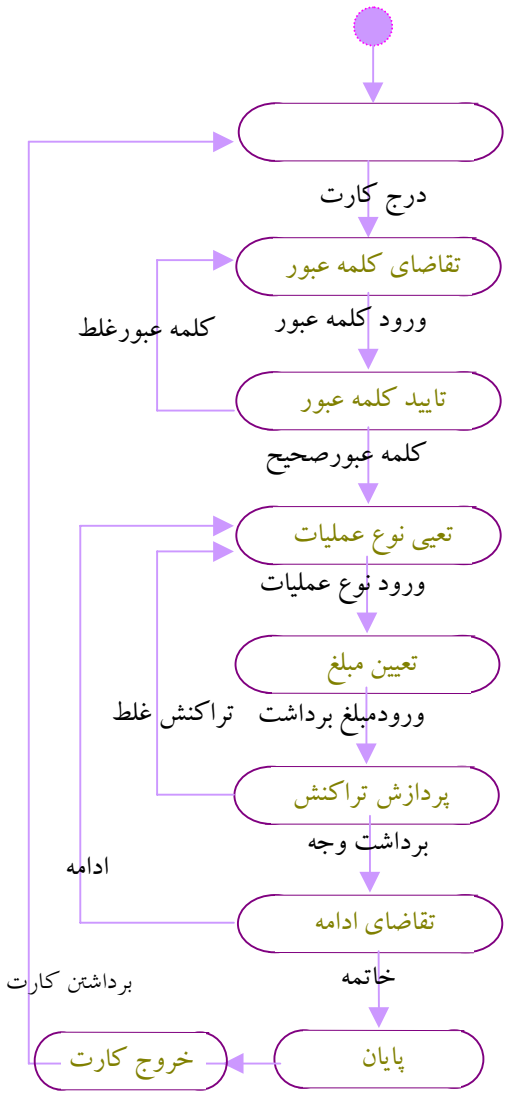
این رخداد بوسیله اشیاء داخل حوزه سیستم به وقوع می پیوندند . در طرح نرم افزار یک رخداد داخلی وقتی بوجود میی آید که پیام ویا سیگنالی در بین اشیاء مبادله می شود. پیامها در دیاگرامهای همکاری ،رخدادهای داخلی محسوب می شوند.

٣. رخداد کمکی

شرایط خاص یا حالت خاص و یا رسیدن به مقدار و یا زمان مورد نظر، صورت می گیرد. در مفاهیم نرم افزاری ،یک رخداد کمکی با یک ساعت بلادرنگ شبیه سازی می شود. برای نمونه در پایان هر ماه باید لیست خقوق ایجاد شود. پایان هر ماه در اینجا یک رویداد زمانی است.

۴-۳-۲ استفاده از دیاگرام حالت

هدف از ایجاد دیاگرام حالت برای کلاسها تعیین لیستی از رویدادهـای موثـر بـر کـلاس و تعییـن متدهایی از کلاس است که در مقابل این رویدادها واکنش می کنند. به این ترتیب می تـوان کلاسـها را به صورت قطعه های[۳۶] با شعور در داخل برنامه ایجاد نمود. البته کل دیاگرام حالت را نیز مـی تـوان بـه عنوان یک متد کنترلی برای کلاس مورد نظر تعیین نمود. برای نمونه به دیاگرام حالت زیر و کد معـادل آن توجه نمایید:




```
Do forever
    Display  صفحه اصلي
    Read  كارت عابربانك
    Repeat
        تقاضاي كلمه عبور
        Read  كلمه عبور
        تاييد كلمه عبور
    Until كلمه عبور تاييد شود
    Repeat
      Repeat
        تقاضاي نوع تراكنش
        Read نوع عمليات
        تقاضاي مبلغ
        شروع تراكنش
        انتظار براي تكميل تراكنش
      Until  تراكنش كامل شده
    Until كاربر تقاضاي خاتمه دهد
    Eject Card
    انتظار براي برداشتن كارت
```

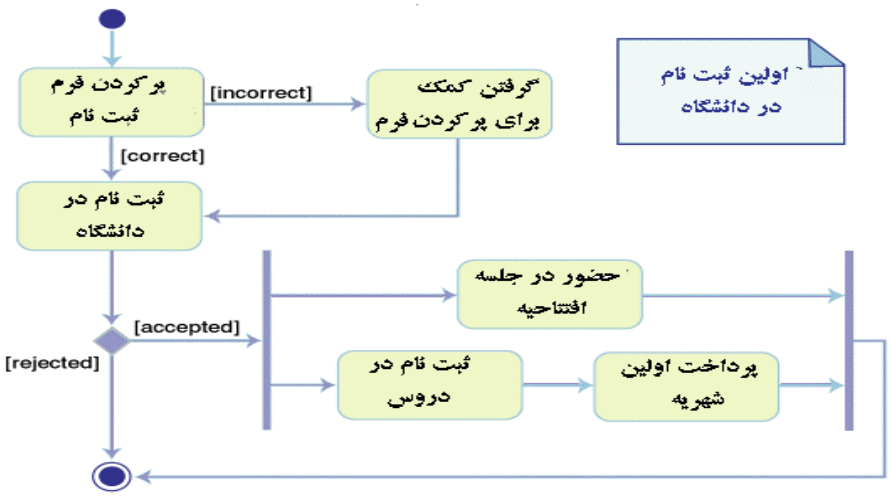
شکل ٤-١١ دیاگرام توالی برای سیستم عابر بانک

---

[۳۶] Component

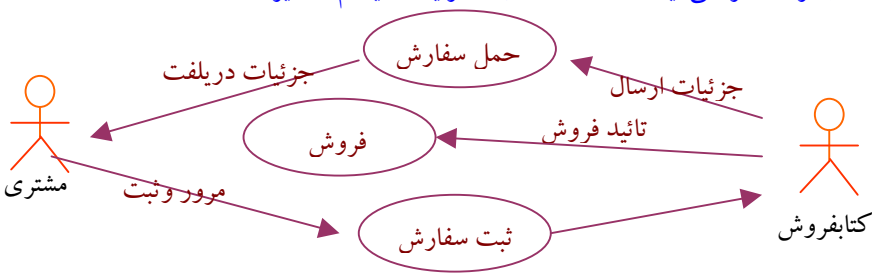
۴-۳-۳ دیاگرامهای فعالیت

دیاگرامهای فعالیت ابزار دقیقتری از دیاگرامهای حالت هستند. این دیاگرامها بیانگر مفهوم و یا منطق عملیاتی یک عمل ، متد، مورد استفاده و یا یک کلاس هستند. سعی کنید بوسیله یک یادداشت جانبی عملکرد دیاگرام فعالیت را خلاصه کنید. برای هر دیاگرام فعالیت باید نقطه شروع و خاتمه مشخص باشد. نقطه شروع با دایره پر و نقطه خاتمه با دو دایره تودرتو مشخص می شود. برای فعالیتهای دائمی، وجود نقاط خاتمه ضروری نیست. هر دیاگرام فعالیت دارای تعدادی فعالیت است. اولین فعالیت توسط یک بازیگر معمولا" فعال می شود. همزمانی و یا توازی در اجرا فعالیتها را می توان در دیاگرام فعالیت مشخص کرد. در دیاگرام فعالیت نقاط تصمیم گیری بصورت لوزی مشخص می شوند. برای نمونه به دیاگرام شکل ٤-١٢ برای ثبت نام دانشجویی توجه نمایید.



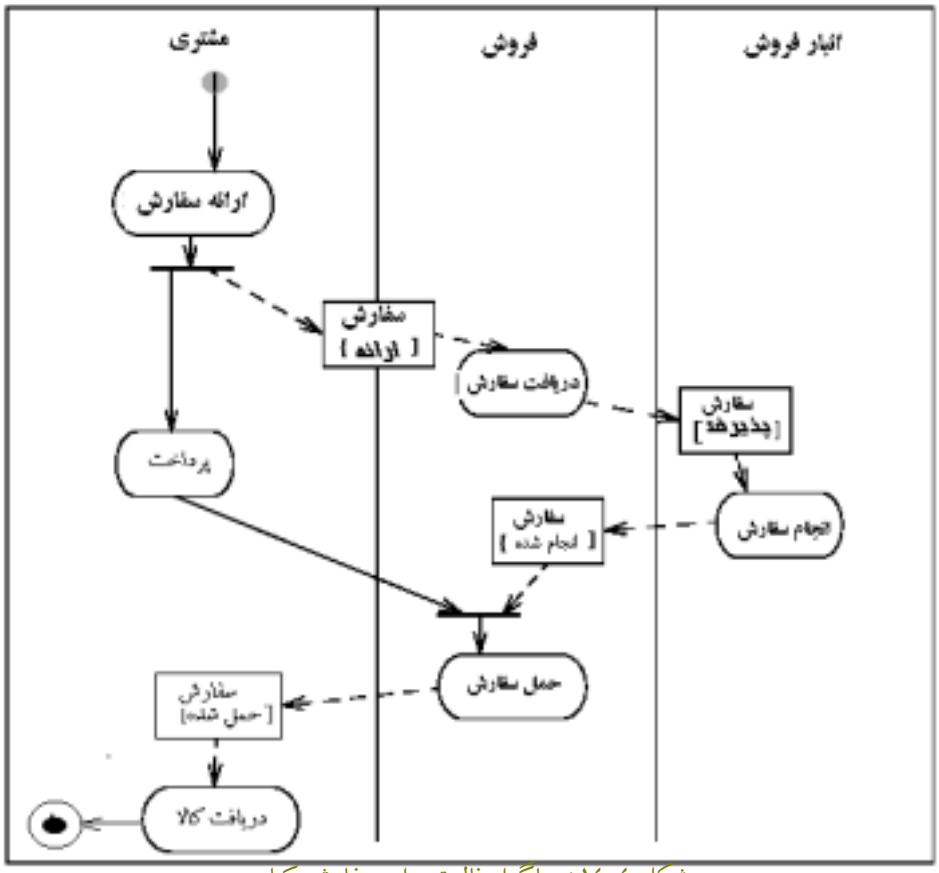
شکل ١٢-٤ : دیاگرام فعالیت برای نمایش عملیات ثبت نام

در شکل زیر دیاگرام مورد استفاده برای سیستم سفارشها ارائه شده است. این دیاگرام مورد استفاده در مرحله طراحی ایجاد شده لذا، حاوی جزئیات سیستم مکانیزه است.



شکل ١٣-٤ : دیاگرام مورد استفاده برای ثبت سفارش و دریافت کتاب

در شکل زیر عملکرد مورد ثبت سفارش با استفاده از یک دیاگرام فعالیت مشخص شده اســت. در ایـن دیاگرام ، گردش کار نیز مشخص شده است. ابتدا مشتری سفارش خود را ارائه می دهد. مسئول فروش سفارش را دریافت می کند و بخش انبار سفارش را انجام مـی دهـد. در ضمـن اینکـه مشـتری بـه واحـد فروش سفارش خود را ارائه می کند، عمل پرداخت وجه را نیز انجام می دهد. همزمـانی در عملیـات را نیز می توان توسط این نوع دیاگرامها بوسیله قطعه خطهای افقی و عمودی مشخص نمود.



شکل ٤-١٤ : دیاگرام فالیت برای سفارش کتاب

فصل پنجم

طراحی فیزیکی

۵-۱ مقدمه:

یکی از ابتکارهای خارق العاده UML ارائه نشانه ها و علائم استاندارد جهت تعیین مدل برنامه ها و طرح فیزیکی سیستم مکانیزه است. از نکات قابل توجه دیگر در نرم افزار رَشنال رُز امکان وارد کردن مدل در زبان برنامه سازی و بالعکس استفاده از زبان برنامه سازی برای ایجاد مدل است که در این بخش مورد بررسی قرار خواهد گرفت.

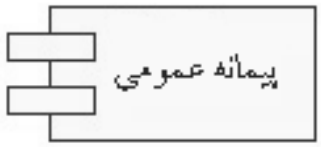
۵-۲ : پیمانه کردن برنامه ها

پس از تعیین کلاسها باید بر طبق اصول پیمانه بندی[۳۷] که چسبندگی ، استقلال و مفهوم بودن پیمانه های کد برنامه را پیشنهاد می کند، مبادرت به دسته بندی کلاسها در قالب پیمانه های برنامه نمود. در محیط رَشنال رُز لفظ قطعه را برای پیمانه ها بکار برده اند. هر پیمانه در واقع مجموعه ای از کلاسهایی است که در داخل برنامه مورد استفاده قرار می گیرند. در ادامه انواع قطعه ها بررسی خواهد شد:

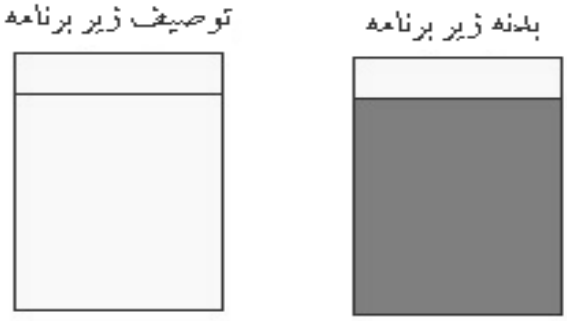
۱- **قطعه عمومی :** در حالت کلی منظور از قطعه پیمانه یا یک مدول از برنامه است. یک قطعه می تواند از نوع ActiveX ، DLL ، اپلت ، و یا Application و یا هر پیمانه ای از کد باشد. یک قطعه عمومی در UML به صورت زیر مشخص می شود.

---

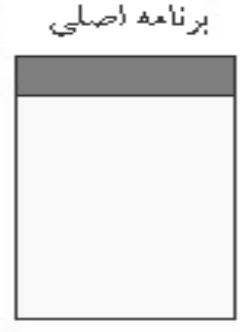
[۳۷] Modularity



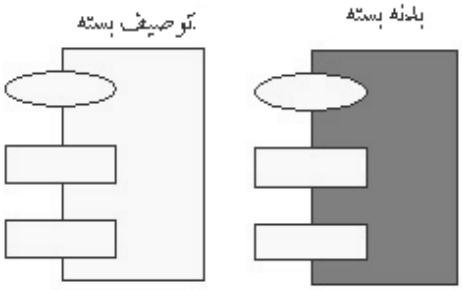
۳. **زیر برنامه ها** : زیربرنامه ها و روالهایی که متعلق به هیچ کلاسی نیستند را درون پیمانه هایی از نوع قطعه زیر برنامه قرار می دهند.. می توان توصیف زیر برنامه را در یک پیمانه و پیاده سازی آنرا در پیمانه دیگر قرار داد. این پیمانه ها شامل کلاس نیستند.



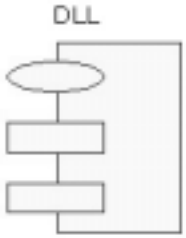
٤. **برنامه اصلی** : بدنه اصلی برنامه را در پیمانه ای تحت عنوان "برنامه اصلی می توان قرار داد.



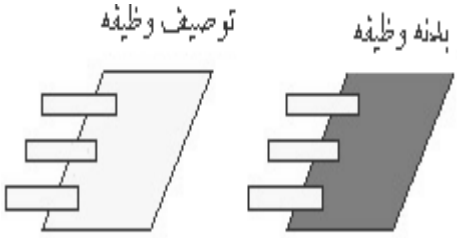
۵. **بسته** : هر بسته شامل کد پیاده سازی برای یک کلاس است. می توان توصیف کلاس را در یک بسته و پیاده سازی آنرا در بسته دیگر قرار داد.



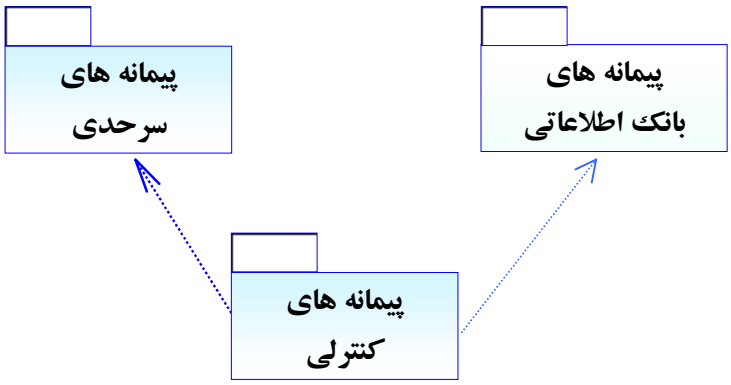
٦. **فایلهای DLL** : کتابخانه های پویا یا ادر اصطلاح DLL ها را در پیمانه ای جداگانـه بـه صـورت زیر می توان نگهداری نمود.



۷. **وظایف** : یک وظیفه به بسته ای اطلاق می شود که می تواند به صورت همزمان با سایر بخشـهای برنامه به اجزا در آید.



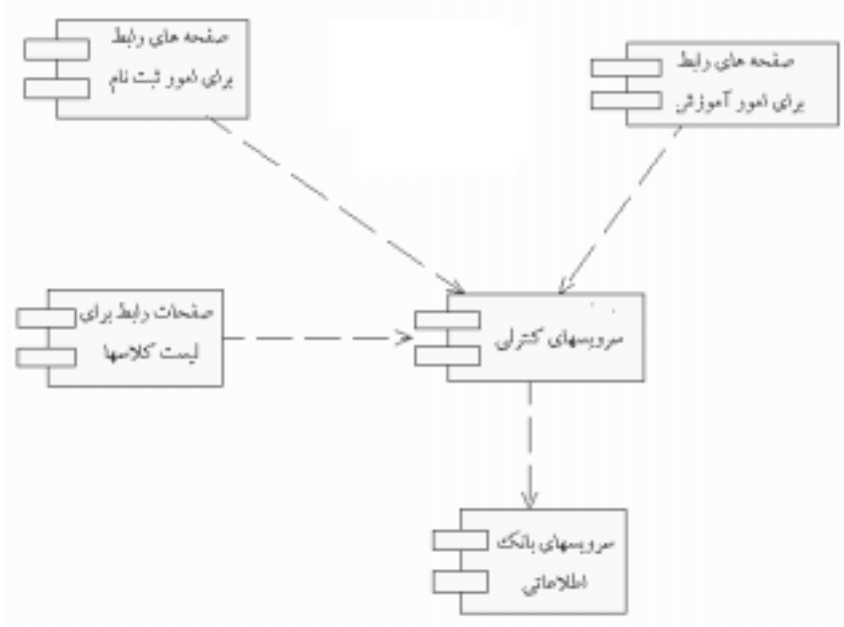
در بین قطعه ها باید روابط را با خطوط خط چین مشخص نمود. البته باید توجه داشـت باشـید کـه این رابطه بیانگر پیوند بین قطعات کد برنامه است که برای کامپایل کـردن کـد برنامـه هـا حـائز اهمیـت است. در حالت کلی جهت ایجاد معماری سه لایه پیشنهادی نـرم افـزار ، سیسـتم مکـانیزه را در حـالت کلی به صورت سه بسته مطابق شکل زیر در نظر می گیرند:



شکل ۵-۱ : معماری سه لایه برای تقسیم بندی نرم افزار

در شکل ۵-۲ پیمانه ها برای یک برنامه کاربردی سیستم آموزش دانشگاه ارائه شده اسـت. پیمانـه ها در این شکل با در نظر گرفتن طرح سه لایه نرم افزار ایجاد شده اند. یک دسته از پیمانه ها لایه رابـط

کاربر ، یک پیمانه کار کنترل عملیات و پیمانه دیگر وظیفه برقراری ارتباط با بانک اطلاعاتی را بر عهده دارد



شکل ۵-۲ : پیمانه بندی برنامه سیستم آموزش

برای هرپیمانه از برنامه می توان در رَشنال رُز ، زبان برنامه نویسی برای آن پیمانه را مشخص نمود. پس از تعیین زبان برنامه نویسی با انتخاب گزینه Tools از داخل محیط رشنال گزینه مربوط به زبان را انتخاب و کد برای پیمانه ها تولید نمود. کد تولیدی شامل تعاریف ارائه شده برای کلاسها و متدها خواهد بود. با یک برنامه ساده مبتنی بر مباحث آنالیز جریان داده ها در کامپایلرها، امکان مهندسی معکوس یٔ تبدیل کد برنامه به مدل نیز در رَشنال رُز فراهم شده است.

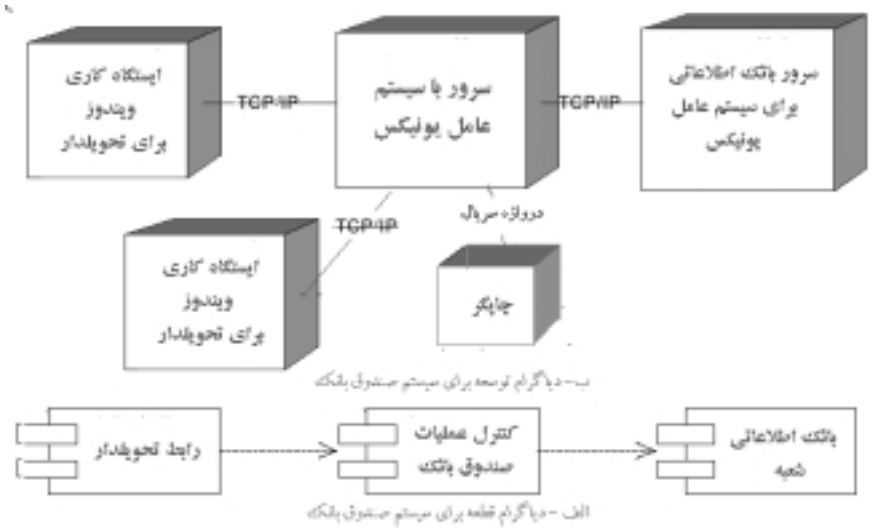
۵-۳ طراحی فیزیکی سیستم مکانیزه

پس از پیمانه بندی برنامه ها ، باید محل کامپیوترها و دستگاه های جانبی آنها و در مجموع طرح شبکه را مشخص نموده ، تعیین کرد که هر پیمانه از برنامه در کدامیک از محلهای فیزیکی تعیین شده باید نصب شود. برای این منظور نوعی خاص از دیاگرام به نام دیاگرام توسعه[۳۸] در UML مطرح شده

---

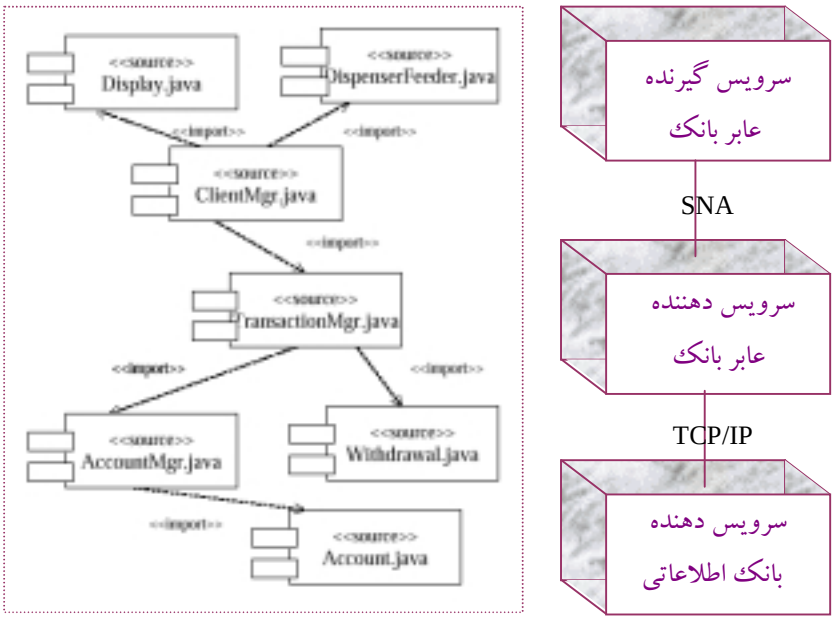
[۳۸] Deployment Diagram

است. در دیاگرام توسعه ، خطوط ارتباطی بین گره ها نمایانگر کابلهای فیزیکی ارتباطی بین دستگاه های جانبی مثل چاپگر با کامپیوتر و خطوط ارتباطی شبکه هستند. برای نمونه در شکل ٥-٣ برای یک سیستم بانکی پیمانه ها و طرح فیزیکی سیستم مکانیزه مشخص شده است.



شکل ٥-٣ : دیاگرامهای قطعه و توسعه برای سیستم صندوق بانک

در شکل ٥-٤ ، دیاگرام قطعات و توسعه برای سیستم عابر بانک ارائه شده است.



شکل ٥-٤ دیاگرام قطعات و توسعه برای سیستم عابر بانک

۵-۴ دسترسی به مدل درون برنامه

برای دسترسی به مدلهای ایجاد شده درون رَشنال رُز ، از طریق برنامه VB باید ابتدا در داخل محیط ویژوال بیسیک منوی Project را انتخاب کنید. از منوی Project گزینه References را انتخاب نمایید. در داخل لیست Rferences گزینه های مربوط به رَشنال رُز را انتخاب کنید. به این ترتیب محیط ویژوآل بیسیک برای ایجاد برنامه شما جهت دسترسی به مدلهای ایجاد شده درون رَشنال رُز، آماده می شود.

حالا می توانید برای نمونه قطعه کد زیر را ایجاد کنید. در این قطعه کد، نام مدلی که در داخل برنامهVB به آن دسترسی می شود، به عنوان نمونه F:\MyRational\MyModel.mdl انتخاب شده است.

```
Private Sub Command1_Click()
    Dim roseApp As New RoseApplication
    Dim mdl As RoseModel
    Set mdl = roseApp.OpenModel("D:\test\matab.mdl")
    MsgBox ("Hello")
    MsgBox mdl.GetAllClasses.GetAt(1).Name
End Sub
```

در مثال فوق ، نام اولین کلاس از کلاسهای مدل در خروجی چاپ می شود. می توان هر گونه ویژگی از یک مدل را داخل برنامه مورد دسترسی قرار داد.

۵-۵ : ایجاد کد دلفی

محیط رَشنال رُز مستقیماً کد دلفی را پشتیبانی نمی نماید. اما، می توان با نصب نرم افزاری تحت عنوان دلفی لینک کد دلفی را ایجاد نمود. این نرم افزار را می توانید از آدرس :

http://www.rational.com/support/downloadcenter/addins/rose/index.jsp

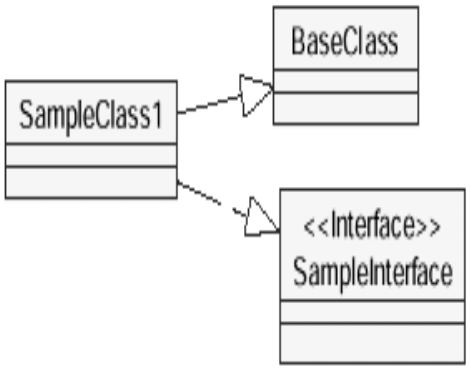
در اینترنت بدست آورید. پس از نصب این نرم افزار در داخل منوی tools در رَشنال رُز گزینه ای تحت عنوان Ensemble Tools ایجاد می شود. با انتخاب گزینه Rose Delphi Link از داخل این منو فرم شکل ۵-۵ بر روی صفحه کامپیوتر ظاهر می شود. از منوی File گزینه New را انتخاب نمایید. حالا می توانید نام پروژه یا برنامه دلفی جدیدی را که قرار است از مدل تولید شود وارد کنید. با تعیین نام برنامه

بلافاصله کلاسها از داخل محیط رشنال به کلاسها داخل برنامه دلفی تبدیل می شوند. توجه کنید که در هنگام ایجاد قطعات باید زبان برنامه سازی برای هر قطعه درون دیاگرام قطعه را دلفی انتخاب کنید.

شکل ٥-٥ : محیط تولید کد و مهندسی معکوس برای زبان دلفی

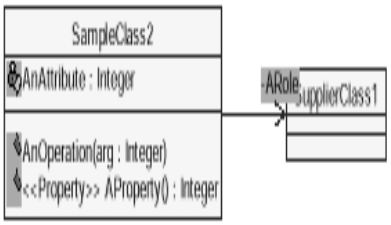
در ادامه کد تولیدی دلفی برای انواع اجتماع کلاسها ارائه شده است:

١- رابطه وراثت : در شکل زیر کلاس SimpleClass1 دارای دو کلاس مافوق است.



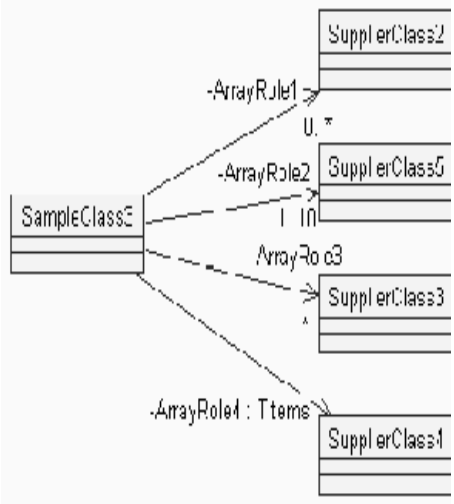

```
type SampleClass1 = class
  (BaseClass, SampleInterface)
  {...}
end;
```

٢. رابطه معمولی :




```
type SampleClass2 = class
private
  AnAttribute : Integer;
  ARole : SupplierClass1;
 Public
  procedure AnOperation
                        (arg : Integer);
 property
    AProperty : Integer  index 2
                 read AnAttribute
                write AnOperation;
end;
```

٤. رابطه چند تایی




```
type SampleClass3 = class
private
  ArrayRole1 : array of SupplierClass2;
  ArrayRole2 : array [1..10] of
  SupplierClass5;
  ArrayRole3 : array [SampleRange] of
  SupplierClass3;
  ArrayRole4 : TItems;
end;
```

هر پیمانه یا در اصطلاح قطعه از دیاگرام قطعه تبدیل به یک Unit دلفی می شود.

فصل ششم

پرسش و پاسخ

۶-۱ مقدمه

تجربه نشان داده که دانشجویان بخصوص در یادگیری مفاهیم و مدلهای مورد استفاده در UML با مشکل مواجه می باشند. یک عامل کاربردی بودن مطالب تحلیل و طراحی است. باید تولید سیستم مکانیزه و برنامه نویسی و اشکالات ناشی از آن در عمل را تجربه کرد تا ارزش مفاهیم مهندسی نرم افزار را بتوان حس کرد. لذا، تجربه برنامه نویسی انگیزه برای یادگیری متدولوژی بوجود می آورد. معمولاً ، افراد ترجیح می دهند مستقیماً برنامه نویسی کنند تا اینکه مدل برنامه ها را نقاشی کنند.

در این فصل سوالات و مشکلات مشاهده شده در کلاسها ودر اغلب دانشجویان مطرح ، پاسخ مناسب ارائه شده است. امیدوارم که از این طریق بتوانم پاسخگوی مشکلات شما برای درک علت ایجاد و چگونگی استفاده از مفاهیم مطرح در بخش دوم از این کتاب باشم.

۶-۲ سوالهای دانشجویان

سوال ۱- چگونه می توان نیازهای عملیاتی و کیفی را مشخص نمود ؟

پاسخ : در ابتدا به سراغ کاربرها می روید و تا جایی که می توانید نیازهای آنها را مشخص می کنید. روش دیگر تکثیر فرمی است به صورت زیر در بین افراد و درخواست از افراد برای پر کردن آن:

سرویس گیرنده	سرویس دهنده	سرویس	فعالیتها	شرایط
مشتری بانک	متصدی حساب	افتتاح حساب جاری	۱. بررسی درخواست	۱. باید معرف معتبر باشد.

			۲. اســـــتعلام از بانک مرکزی	۲. در حداکـــثر ۲٤ ساعت

به این ترتیب سرویس گیرنده به عنوان بازیگر اصلی ، سرویس دهنده ها بـه عنـوان بازیگرهـای فرعـی ، سرویسها به عنوان موردهای استفاده و شرایط به عنوان نیازهای کیفی و نیازهـای عملیـاتی مشـخص مـی شوند.

سوال ۲- کاربرها معمولا" پاسخ صحیح نمی دهند می گویند ما دستی هم کارهایمـان انجـام مـی شـود. در اینگونه موارد چه باید کرد؟

پاسخ : در این صورت به سراغ مسئولین آنها بروید. البته در ضمن یافتن موردهای استفاده شـما شـرایط لازم برای هر مورد استفاده را به عنوان نیازهای عملیاتی و کیفی مشخص می کنیـد. بـرای نمونـه فـرض کنید مورد استفاده "تحویل کالا" باشد، در این صورت با در نظـر گرفتـن ایـن مـورد مـی تـوان دو نیـاز عملیاتی را به صورت زیر مشخص نمود:

۱. سیستم باید درخواستهای بیش از X تومان را پس از تایید مدیریت بپذیرد.
۲. سیستم باید درخواستهای الگو را نگهداری نماید.

سوال ۳- موردهای استفاده را چگونه باید مشخص کرد؟

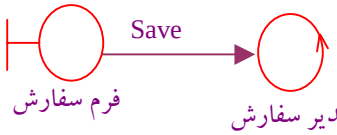
پاسخ : به هر مورد استفاده می توان به عنوان یک گزینه منوی سیستم نگاه کرد. حالا موردهـای اسـتفاده گزینه های منوی شما هستند و نیازهای عملیاتی شرایط لازم برای اجرا صحیح این گزینه ها.

سوال ٤ به نظر من اگر ما برنامه را می نوشتیم خیلی سریعتر کارها انجام می شد تا اینکه "مورد استفاده" تعیین کنیم و نقاشی بکشیم ؟!

پاسخ : بله بعضا"حرف شما صحیح است. برای همین است کـه در متدولـوژی RUP مراحـل تکـرار مـی شوند . طبق منحنی ارائه شده در شکل ۱-۱۷ ، از ابتدای آنـالیز بـاید هسـته برنامـه را ایجـاد کـرد. مـورد استفاده پیچیده را باید دقیق مشخص و مدل کنید تا بعدا" با مشـاهده آن عملکـرد برنامـه را درک کنیـد. حاصل کار شما یک برنامه کاملا" شیء گرا خواهد بود.

سوال ۵- من نمیدانم اولا" چگونه باید مدل ارتباطی کلاسها را کشید و ثانیا" نمیدانم بـه چـه دردی مـی خورد؟

پاسخ : بهتر است که به دیاگرام توالی ارجاع کنید برای نمونه بـه دیـاگـرام توالـی در شـکل ٤-٥ توجـه نمائید. برای این دیاگرام مدل ارتباطی کلاسها را می توانید بر اسـاس ارتبـاط بیـن کلاسـها در دیـاگـرام توالی ایجاد کنید. برای نمونه در این دیاگرام متـد Save از کـلاس مدیرسـفارش توسـط کـلاس فـرم سفارش مورد دسترسی قرار گرفته است. بنابراین بین این دو کلاس در مدل ارتباطی کلاسها یک رابطـه به شکل زیر ایجاد می شود:



هنگام تولید کد توسط رَشنال رزِ در داخل کلاس "فرم سـفارش" فیلـد یـا شـیء ای از نـوع "مدیـر سفارش" اضافه می شود. اگـر نـام ایـن فیلـد OrderManager باشـد مـی تـوان در داخـل کـلاس "فـرم سفارش" تابع OrderManager.Save() را مورد فراخوانی قرار داد.

سوال ٦: برای ما ترسیم دیاگرام توالی بسیار مشکل است . چگونه باید عمل کنیم ؟

پاسخ : دیاگرام توالی را بر اساس سناریوی مورد استفاده ایجـاد کنیـد. بـرای مثـال بـه شـکل ٤-٤ توجـه کنید. ابتدا فرض کنید در مقابل کامپیوتر قرار گرفته اید. برای شما به عنوان یک بازیگر کلاسی وجـود دارد که حاوی منوهای مربوط به شما است. از داخل این منو گزینه ای را انتخاب می کنید. بلافاصله متد Show از یک فرم ورودی / خروجی باید فعال شود تا آن فرم بر روی صفحه طاهر شود. برای نمونه فرم درخواست از انبار ممکن است ظاهر شود. پس اولیـن شـیء یـک فـرم ورودی / خروجـی اسـت. حـالا جزئیات را وارد می کنید و با فشردن دکمه ای بر روی فرم عملی را تقاضا می کنید. به این ترتیب بـا در نظر گرفتن سناریو و با تصور ارتباط سیستم با کاربر می توانید، در یک سیر تکاملی ، به مرور کلاسـها و متدهای مورد نیاز را مشخص نموده ، دیاگرام توالی را ترسیم کنید.

سوال ٧: شما می گویید باید تقوی علمی داشـته باشـیم و هـر چـیزی را بـدون دلیـل قبـول نکنیـم. ایـن دیاگرام حالت چه استفاده ای دارد؟

پاسخ : ببینید فردی از داخل خیابان دارد عبور می کند. ماشینی با سرعت می آید. بلافاصله فرد به سـمت دیگر میدود تا ماشین به وی برخورد نکند. این واکنش در مقابل رویداد با انجـام فعـل دویـدن مشـخص شد. بنابراین می توان گفت شعور با چگونگی واکنش در مقابل رویدادها مشخص می شود. یک قطعه[٣٩]

---

[٣٩] Component

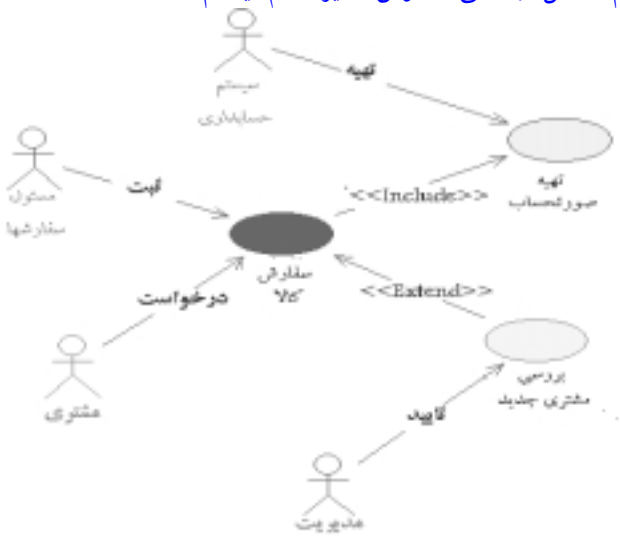
در واقع یک کلاس با شعور است. برای ایجاد قطعه با شعور باید رویدادهای موثر بر کلاس را مشخص نمود و تعیین کرد که چه متدی از کلاس با وقوع هر رویداد، به عنوان واکنش آن کلاس در مقابل رویداد فعال می شود. لیست رویدادهای موثر بر یک کلاس و واکنش کلاس ر مقابل هر رویداد را می توان درون دیاگرام حالت برای آن کلاس جویا شد. در هنگام ترسیم دیاگرام حالت باید هدف و نهایتا" استفاده از دیاگرام حالت در تولید کد را مد نظر داشت.

سوال ۸: در شرکت ما گروه های طراحی کار تحلیل و طراحی را انجام داده و حاصل طراحی را به برنامه نویسها تحویل میدهند. در روش شیء گرا چه چیزی به برنامه نویسها تحویل می شود؟

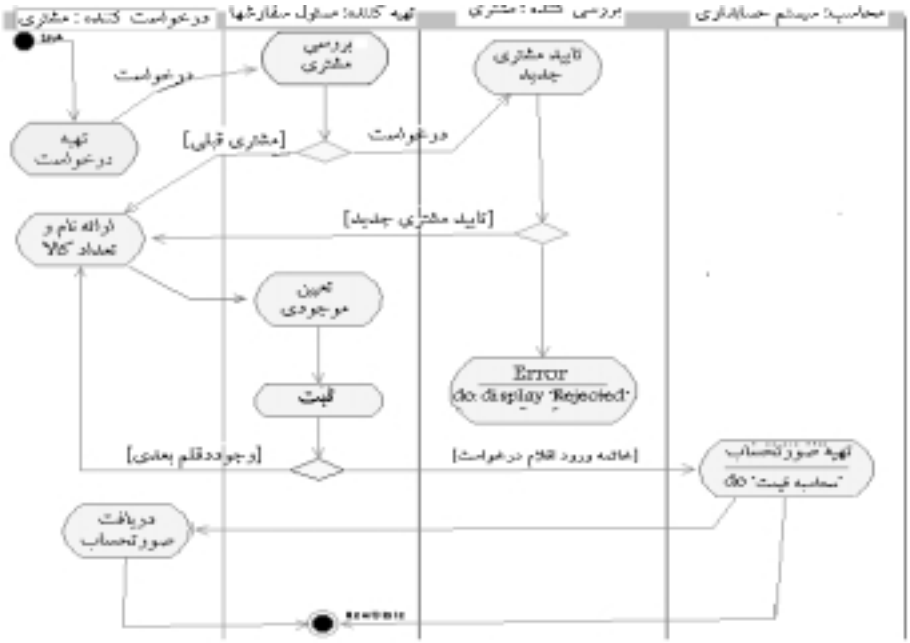
پاسخ : در روش شیء گرا این دو مرحله همزمان عمل می شوند. برای موارد پیچیده مدل ایجاد می کنند و موارد عادی را مستقیما" پس از یک شرح مختصر به کد تبدیل می کنند. باید دقت کنید که دیاگرام توالی به مرور موجب یافتن کلاسها و متدهای کلاسها می شود. بر این مبنا می توان برای یک مورد استفاده کد ایجاد نمود. در واقع یک دیاگرام توالی را همانگونه که قبلا" نیز نشان داده شد، می توان به صورت دنباله ای از جمله های فراخوانی تو در تو تبدیل نمود. علاوه بر این دیاگرام مدل ارتباطی کلاسها ها نیز مستقیما" به کد برنامه تبدیل می شود. دیاگرام حالت ابزاری برای، تبدیل کلاسها به قطعه های با شعور مثل ActiveX و یا قطعه های دلفی و یا Beans در جاوا است که با وقوع رویداد در مقابل آن واکنش می کنند.

سوال ۹- می توانید با یک مثال کوچک مراحل کار را به ما نشان دهید؟

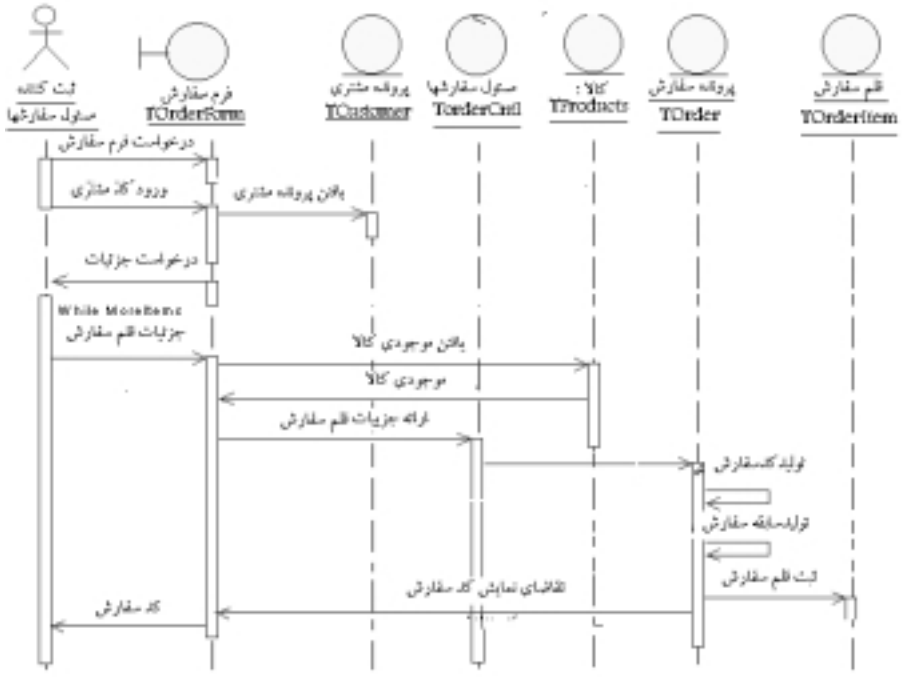
پاسخ : بله در ادامه یک مورد را در ارتباط با سفارش کالا مشخص نموده، کد دلفی و VB برای آن ایجاد می کنم و عمل مهندسی معکوس را نیز انجام میدهم.



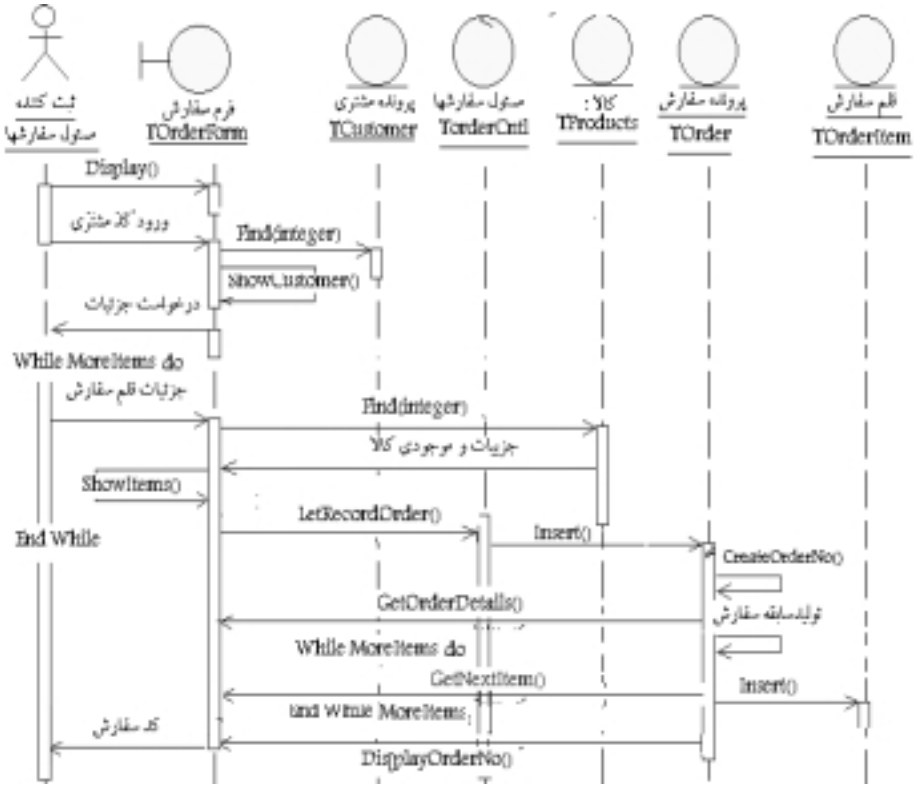
مفهوم مورد استفاده سفارش کالا را می توان با یک دیاگرام فعالیت به صورت زیر مشخص نمود.



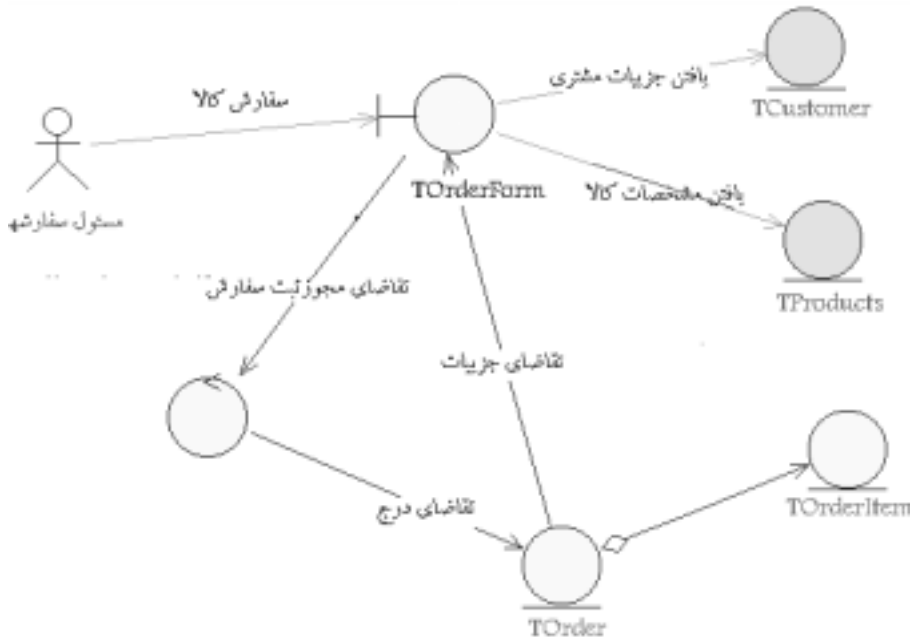
بر اساس توصیف موردهای استفاده اکنون دیاگرام اولیه توالی به صورت زیر استخراج می شود:



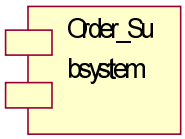
دیاگرام توالی نهایتا" به صورت زیر تبدیل می شود. این دیاگرام عاملی برای تعیین متدهای مورد نیاز برای هر کلاس نیز است.



اکنون با در نظر گرفتن ارتباط بین کلاسها در دیاگرام توالی فوق می توان مدل ارتباطی کلاسها را به صورت زیر استخراج نمود. در واقع این دیاگرام نشان می دهد که کدام کلاسها ناز به دسترسی به متدها و یا فیلدهای کلاس دیگری دارند..



در مرحله بعد مبتنی برنگرش قطعات[٤٠]، یک پیمانه برای زیر سیستم سفارش در نظر گرفته می شود. کلیه کلاسهایی که در مدل فوق ظاهر شده اند به این پیمانه اضافه می شود.



ابتدا برای پیمانه فوق گد دلفی تولید می شود. برای تولید کد دلفی به نکات ذکر شده در فصل قبل ارجاع کنید.

بخشی از کد دلفی تولید شده

```
// زیر سیستم سفارشها
unit Order_Subsystem;
interface
type
  TOrderForm = class;
  TOrderCntl = class;
  Tcustomer = class;
  TGoods = class;
  TOrder = class;
```

---

[٤٠] Component View

```
  TOrderItem = class;
// کلاس فرم سفارش
  TOrderForm = class
  private
    // شماره سفارش که توسط سیستم به طور اتوماتیک تولید می شود.
   // بهتر است شماره سال روز و شماره سفارش در آن روز در آن مستتر باشد .
    No : integer;
    Date : integer;    // تاریخ سفارش
    Supplier : string; // تهیه کننده
    Total : real; // جمع کل مبلغ
  public
    theTcustomer : Tcustomer;
    theTGoods : TGoods;
    theTOrderCntl : TorderCntl;
    procedure Display; ;        // نمایش فرم سفارش
    procedure ShowCustomer; ;      // نمایش جزئیات مشتري
    procedure ShowItem;    // نمایش یك قلم
   function getOrderDetail : TOrder; // گرفتن جزئیات سفارش
    // گرفتن قلم بعدي
    function GetNextItem : TOrderItem;
    procedure DisplayOrderNo (// شماره سفارش
                        orderNo : integer);
  end;
```

بخشی از کد ویژوال بیسیک تولید شده

```
    'حاوی جزئیات مشتریها است.
    Option Explicit
    'شماره مشتری
    '##ModelId=3B7F37880366
    Private No As Integer

    'تاریخ پیوستن به لیست مشتریها
    '##ModelId=3B7F37A30348
    Private Date As integer

    'نام مشتری
    '##ModelId=3B7F37CC00D2
    Private Name As char[40]

    '##ModelId=3B7F37EC0262
```

```
        Private Address As char[40]

        'شماره تلفن
        '##ModelId=3B7F38060172
        Private tel As Integer

        'شماره موبایل
        '##ModelId=3B7F381D014A
        Private Mobile As Long

        'شماره فکس
        '##ModelId=3B7F3835030C
        Private Fax As Integer

        'جمع کل مبلغ خرید
        '##ModelId=3B7F384D0118
        Private TotalPurchase As Float

        'کار این تابع یافتن مشتری است.
        '##ModelId=3B7E3EC00230
        Public Function Find(CustomerNo As Integer) As Boolean
            On Error GoTo FindErr

            '## Your code goes here ...

            Exit Function
        FindErr:
            Call RaiseError(MyUnhandledError, "Find Function")
    End Function
```