

## بخش چهارم مهندسی نرم افزار شیء گرا

در این بخش از کتاب "مهندسی نرم افزار : رهیافتی برای یک اهل فن ، مفاهیم فنی ، شیوه ها و اندازه گیری هایی را که قابل به کارگیری در تحلیل ، طراحی ، آزمون نرم افزار شیء گرا هستند ، مورد بررسی قرار خواهیم داد. در فصل های آتی سؤالات زیر مورد توجه خواهند بود:

- مفاهیم و اصول اولیه ای که در یک تفکر شیء گرا مورد استفاده دارند ، کدامند؟
  - چه تفاوتی میان نگرش شیء گرا و نگرش های قراردادی و متداول وجود دارد؟
  - چگونه باید پروژهای نرم افزاری شیء گرا را طراحی و مدیریت نمود؟
  - تحلیل شیء گرا چیست؟ و مدل های مختلف آن چگونه مهندس نرم افزار را قادر می سازد راماها ، ارتباط میان آنها و اعمال آنها را درک کند؟
  - عناصر یک مدل طراحی شیء گرا کدامند؟
  - مفاهیم و اصول اولیه ای که برای آزمون نرم افزار شیء گرا قابل اجرا هستند ، کدامند؟
  - چگونه راهبردهای آزمون و شیوه های طراحی استفاده شده در آزمون ، هنگام بررسی نرم افزار شیء گرا تغییر می کنند؟
  - چه معیارهای فنی برای ارزیابی کیفیت نرم افزار شیء گرا موجود می باشد؟
- زمانی که به این سؤالات پاسخ داده شد ، متوجه می شوید که چگونه نرم افزار را با استفاده از پارادایم شیء گرا تحلیل ، طراحی ، پیاده سازی و آزمون کنید.

## فصل ۲۰ اصول و مفاهیم شیء‌گرا

### مفاهیم کلیدی (مرتب بر حروف الفبا)

اشیاء ، برآورد OO ، بسته بندی (کپسوله کردن) ، پیامها ، جارجوب مشترک ، فرآیند برای OO ، سلسله مراتب کلاسها ، صفات خاصه ، عملیات ، فرآیند بازگشتی / موازی ، کلاسها ، مدل فرآیند OO ، متریک های OO ، وراثت

### KEY CONCEPTS

Attributes , class hierarchy , classes , CPF for OO , encapsulation , inheritance , messages , OO estimation , OO metrics , OO process model , objects , operations , recursive/parallel process

### نگاه اجمالی

دیدگاه شیء‌گرا چیست؟ برای بررسی مسئله‌ای که قرار است با استفاده از راه‌حلی مبتنی بر نرم‌افزار حل شود، شیوه‌های متعددی وجود دارد. یک روش که در حل مسئله به دفعات مورد استفاده قرار می‌گیرد، دیدگاه شیء‌گراست. دامنه مسئله به وسیله یک سری اشیاء توصیف می‌شود که دارای صفات و رفتارهای خاص می‌باشند. اشیاء با مجموعه‌ای از عملکردها (به نام صفات خاصه، عملیات یا خدمات) دست‌کاری شده و از طریق یک پروتکل پیام‌بری با هم ارتباط برقرار می‌سازند. اشیاء به کلاس و زیرکلاسهای آنها تقسیم‌بندی خواهند شد.

چه کسی این کار را انجام می‌دهد؟ تعریف اشیاء در برگیرنده توصیفی از صفات خاصه، رفتارها، عملیات و پیام‌هاست. کاری که توسط مهندس نرم‌افزار انجام می‌پذیرد.

چرا این دیدگاه از اهمیت برخوردار است؟ یک شیء، دارای داده‌ها و روند پردازشی است که در آن داده‌ها به کار گرفته می‌شوند. این مشخصه مهم باعث ایجاد کلاسهای مختلف اشیاء شده است و به دنبال آن منجر به کتابخانه‌هایی از کلاسها و اشیای قابل استفاده مجدد خواهد شد. از آن روی که استفاده مجدد یک نگرش بسیار مهم در مهندسی نرم‌افزار است، معیار شیء‌گرا برای بسیاری از سازمان‌های تولید نرم‌افزار جذاب است. علاوه بر این، اجزاء نرم‌افزاری بدست آمده با استفاده از معیار شیء‌گرایی مشخصه‌هایی از طراحی (مانند استقلال عملکردی، پنهان‌سازی اطلاعات) را نشان می‌دهند که با کیفیت بسیاری بالای نرم‌افزار مرتبط خواهند بود.

مراحل کار چیست؟ مهندسی نرم افزار شیء گرا دارای همان مراحل است که در روش های متعارف طی می شود. تحلیل، اشیاء و کلاسهایی که مربوط به دامنه مسئله هستند که معین می کنند، طراحی معماری، رابط و جزئیات سطح اجزاء مشخص می گردد. پیاده سازی (با استفاده از یک زبان شیء گرا) طراحی را تبدیل به یک کد (برنامه) می نماید. آزمون معماری شیء گرا، رابطها و اجزاء را آزمون می نماید.

محصول کار چیست؟ مجموعه ای از مدل های شیء گرا ساخته می شود. این مدل ها نیازمندیها، طراحی و فرآیند آزمون برای یک سیستم یا محصول را توضیح می دهد. چگونه مطمئن شوم که این کار را درست انجام داده ام؟ در هر مرحله محصول کار شیء گرای از نظر بی نقص بودن، کامل بودن و هماهنگی با نیازهای مشتری و هماهنگی با دیگر قسمت ها مورد آزمون واقع می شود.

ما در دنیای اشیاء زندگی می کنیم. این اشیاء در طبیعت، صنایع ساخته شده توسط بشر، در کار و در محصولات می کنیم، وجود دارند. می توان آنها را ردیابی، توصیف، سازمان دهی، ترکیب، دست کاری و از نو ایجاد نمود. بنابراین پیشنهاد دیدگاه شیء گرا برای ایجاد نرم افزار کامپیوتر تعجب برانگیز نخواهد بود، نگرشی که مفهومی و اقتزاعی بوده و ما را قادر می سازد که جهان را به شیوه ای مدل سازی کنیم که درک و سیر بهتر آن ممکن باشد.

روش شیء گرا در تولید نرم افزار برای اولین بار در اواخر دهه ۶۰ پیشنهاد گردید. تقریباً ۲۰ سال به طول انجامید تا فناوری شیء گرا به طور گسترده مورد استفاده قرار گرفت. در سراسر دهه ۶۰ مهندسی نرم افزار شیء گرا تبدیل به معیاری برای انتخاب بسیاری از سازندگان محصولات نرم افزاری و تعداد بسیاری از سیستم های اطلاعاتی و نرم افزارهای تخصصی مهندسی گردید. با گذشت زمان، فناوری های مبتنی بر شیء، جایگزین روش های کلاسیک و سنتی نرم افزار شدند. سؤال مهمی که هم اکنون مطرح است اینست: چرا؟ پاسخ آن چون بسیاری دیگر از سؤالات مربوط به مهندسی نرم افزار، پاسخی ساده نخواهد بود. بعضی معتقدند که افراد متخصص صرفاً در جستجوی رهیافتی جدید بوده اند، اما این دیدگاه بسیار ساده لوحانه می نماید. فناوری های مبتنی بر شیء منجر به یک سری مزایای تفکیکناپذیر نشده که در هر دو سطح فنی و مدیریتی مزایای زیادی به همراه داشته است.

فناوری های مربوط به شیء منجر به کاربرد مجدد شده و استفاده مجدد (از اجزای برنامه) منجر به تولید نرم افزار سریعتر و برنامه های با کیفیت تر می شود. نگهداری نرم افزار شیء گرا ساده تر است زیرا ساختمان آن فی نفسه غیر مزدوج و یکپارچه است. این امر اثرات جانبی را در هنگام الزام ایجاد تغییر، کمتر نموده و ناراحتی کمتری را برای مهندس نرم افزار و مشتری ایجاد می کند. علاوه بر این، سیستم های شیء گرا راحت تر مورد تطابق و ارزیابی قرار می گیرند. (یعنی سیستم های بزرگ را می توان با مونتاژ سیستم های کوچک تر قابل استفاده مجدد، تولید نمود).

در این فصل اصول و مفاهیم مقدماتی را که پایه و اساس شناخت فناوری مربوط به شیء است،

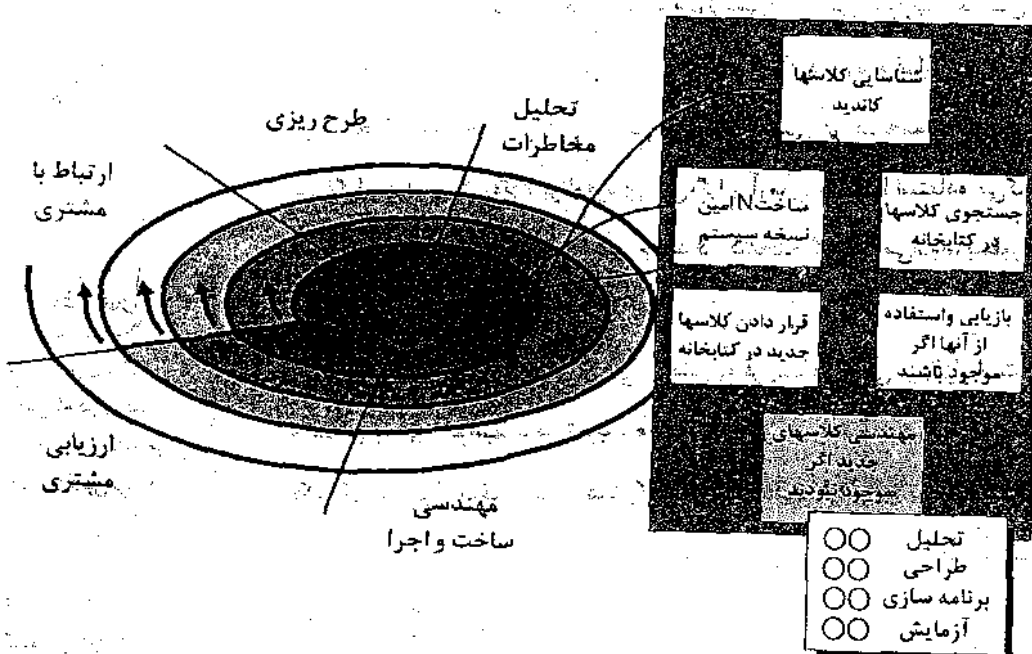
معرفی می‌کنیم.

## ۲۰-۱ بارادایم شیء‌گرا

نقل قول

واقعاً استاندارد است که پیمانه‌ها را برای سیستم‌های پیچیده با اشیاء بسازیم. بنابراین استفاده از برنامه‌سازی سلسله‌مراتبی دیوید تیلور

در طول سال‌های متعددی، اصطلاح شیء‌گرا (OO) برای اشاره به رهیافتی در تولید نرم‌افزار استفاده می‌شد که یک زبان برنامه‌نویسی شیء‌گرا را از میان چند زبان، مورد استفاده قرار می‌داد. امروزه معیار OO در برگیرنده دیدگاه کاملی از مهندسی نرم‌افزار است. ادوارد برارد این امر را زمانی بیان می‌دارد که می‌گوید: «مزایای فناوری شیء‌گرا در صورتی افزایش می‌یابد که در اوایل کار و در طول فرایند مهندسی نرم‌افزار مورد بررسی قرار گیرد. فناوری شیء‌گرای موردنظر باید اثرش را روی کل فرایند مهندسی نرم‌افزار، اعمال کند. به کارگیری برنامه‌نویسی شیء‌گرا به‌طور صرف، بهترین نتیجه را منعکس نمی‌کند. مهندس نرم‌افزار و مدیران آنها باید مواردی، هم‌چون تحلیل نیازمندی‌های شیء‌گرا (OORA)، طراحی شیء‌گرا (OOD)، تحلیل دامنه شیء‌گرا (OODA)، سیستم‌های پایگاه داده‌ای شیء‌گرا (OODBMS) و مهندسی نرم‌افزاری که کمک کامپیوتر شیء‌گرا (OOCASE) را در نظر بگیرند.»



شیء‌گرا : Object Oriented : OO

شکل ۲۰-۱ مدل پردازش شیء‌گرا

خواننده‌ای که با رهیافت متعارف در مهندسی نرم‌افزار آشناست (ارائه شده در بخش سوم این کتاب) ممکن است با بی‌تفاوتی در مقابل گفته بالا، عکس‌العمل نشان دهد: «این معامله بزرگ چیست؟ ما وقتی با استفاده از روش‌های قدیمی کار طراحی نرم‌افزار را انجام می‌دهیم، از تحلیل، طراحی، برنامه‌سازی، آزمون و فن‌آوری‌های مربوطه استفاده می‌کنیم. چرا فکر می‌کنید روش شیء‌گرا چیز متفاوتی است؟ در واقع، چرا باید OO متفاوت‌تر باشد؟» به‌طور مختصر می‌توان، خیر این روش فرقی با بقیه ندارد.

در فصل ۲ در مورد مدل‌های فرایند مختلف برای طراحی نرم‌افزار، بحث کردیم گرچه هر یک از این مدل‌ها را می‌توان برای استفاده در شیء‌گرایی انتخاب کرد، اما بهترین انتخاب تشخیص این است که سیستم‌های شیء‌گرا در طول زمان تکامل می‌یابند. بنابراین، مدل فرایند تکاملی همراه با رهیافتی که مونتاژ اجزاء را تشویق می‌نماید، بهترین الگو برای طراحی نرم‌افزاری شیء‌گراست. با توجه به شکل ۲-۱۰، مدل فرایند تولید بر مبنای جزء (فصل ۲) برای طراحی نرم‌افزاری شیء‌گرا، تدوین شده است.

فرایند شیء‌گرا از یک مسیر تکاملی عبور می‌کند که از ارتباط با مشتری آغاز می‌شود. در اینجا است که دامنه مشکل تعریف شده و کلاسهای اولیه مسئله (که بعداً در این فصل مورد بحث قرار می‌گیرد) شناسایی می‌شوند. تحلیل خطرات و برنامه‌ریزی، پایه و بنیان طرح پروژه شیء‌گرا را تشکیل می‌دهند. کار فنی مربوط به مهندسی نرم‌افزار شیء‌گرا مسیر تکرار شونده نشان داده شده در محوطه سایه زده شده را طی می‌کند. این کار بر استفاده مجدد تأکید دارد. بنابراین کلاسها قبل از این‌که ساخته شوند در کتابخانه مورد جستجو قرار می‌گیرند. وقتی کلاسی در کتابخانه پیدا نشد، مهندس نرم‌افزار از تحلیل شیء‌گرا، طراحی شیء‌گرا، برنامه‌ریزی شیء‌گرا و آزمون شیء‌گرا استفاده می‌کند تا کلاس و اشیای مشتق از آن کلاس را ایجاد کند. این کلاس جدید در کتابخانه گذاشته می‌شود، به‌طوری‌که در آینده مورد استفاده مجدد قرار گیرد.

دیدگاه شیء‌گرا نیازمند نگرشی تکاملی به فرایند مهندسی نرم‌افزار است. همان‌گونه که در این فصل و فصول بعدی خواهیم دید، تعریف همه کلاس‌های مورد نیاز برای یک سیستم با محصول عمده تنها در یک تکرار، بسیار سخت است. همان‌گونه که مدل‌های طراحی و تحلیل شیء‌گرا تکامل می‌یابند، نیاز به کلاس‌های اضافه مشخص‌تر می‌گردد. به همین دلیل است که مدل توصیف شده فوق به بهترین نحو برای بازآرایی شیء‌گرا عمل می‌کند.

## ۲-۲۰ مفاهیم شیء‌گرا

هرگونه بحثی در مورد مهندسی نرم‌افزاری شیء‌گرا باید با مورد مخاطب قرار دادن عبارت شیء‌گرا، آغاز کرد. دیدگاه شیء‌گرا چیست؟ چرا شیوه‌ای را شیء‌گرا در نظر می‌گیریم؟ شیء چیست؟ در طول سالیان متمادی نظرات مختلفی در مورد پاسخ صحیح به این سؤالات وجود داشته است. در این مبحث، تلاش داریم که معمول‌ترین این نظرات را ادغام کنیم.



سیستم‌های شیء‌گرا (OO) با یک مدل فرایند اصلاحی مهندسی می‌شوند. در ادامه این فصل، ارجاعی به یک مدل موازی بازگشتی خواهیم داشت.

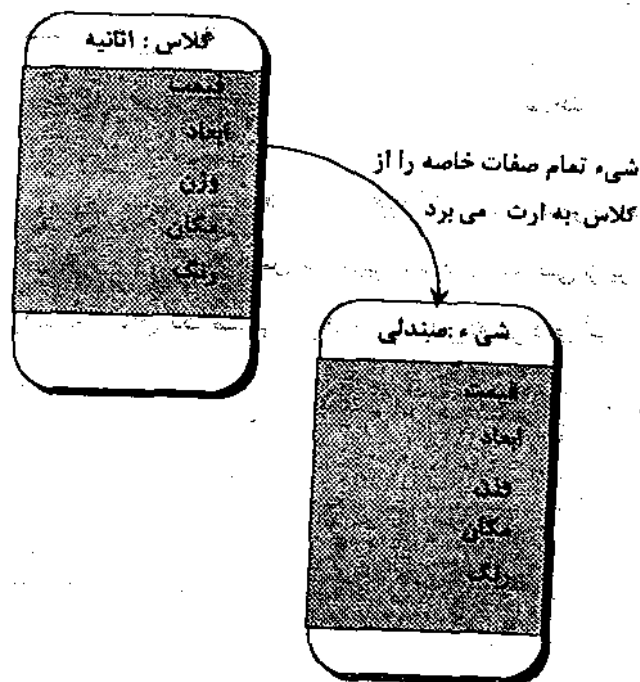


ارجاع به وب یک لیست کامل از منابع (OO) برآدرس زیر وجود دارد:  
[www.mini.net/cetus/software.htm](http://www.mini.net/cetus/software.htm)

## نقل قول

برنامه‌سازی شیء‌گرا، آنقدر که دارای فنون بسته‌بندی برنامه می‌باشد، از فنون برنامه نویسی برخوردار نیست. از اینرو راهی را فراگشوده که تهیه کنندگان، قابلیت‌های عملیاتی را برای انتقال به مشتریان بسته بندی (کپسوله) نمایند.  
براد کوکس

به منظور درک دیدگاه شیء‌گرا مثالی را از جهان واقعی مثل صندلی در نظر بگیرید، چیزی که الآن روی آن نشسته‌اید. صندلی متعلق به کلاس بزرگ‌تری از اشیاء است (به آن مصداق<sup>۱</sup> هم می‌گویند) که ما آن را اثنایه می‌نامیم. مجموعه‌ای از صفات کلی را می‌توان به هر شیء در این مجموعه اثنایه، اطلاق کرد. مثلاً، همه اثنایه دارای هزینه، ابعاد، وزن، محل و رنگ می‌باشند. این وقتی به کار می‌رود که بینیم آیا در مورد میز یا صندلی، یک کاناپه یا یک کمد لباس صحبت می‌کنیم. از آن‌جا که صندلی عضوی از اثنایه اشیاء تمام صفات<sup>۲</sup> تعریف شده برای این کلاس را به ارث<sup>۳</sup> می‌برد. این مفهوم بصورت شماتیک در شکل ۲-۲ با استفاده از عبارت شناخته شده UML، نشان داده شده است. در این شکل مربع علامت زده شده با گوشه‌ها تا خورده، با عبارتی در یک زبان برنامه‌نویسی نسبتی دارد. [BER93]<sup>۴</sup>، [TAY90]<sup>۵</sup>، [STR88]<sup>۶</sup> و [BOO86]<sup>۷</sup>



شکل ۲-۲ ارث بری شیء از کلاس

1. instance
2. attributes
3. inherits
4. Berard, E. V.
5. Taylor, D. A.
6. Stroustrup, B.
7. Booch, G.



وقتی کلاسی تعریف شد، به هنگام ایجاد کلاس جدیدی از مواد می‌توان از این صفات مجدداً استفاده نمود. مثلاً، فرض کنید که قرار بود شیء جدید به نام **Chable** (میز صندلی) را تعریف کنیم. (چیزی بین میز و صندلی **Table** و **Chair**) که عضوی از کلاس اثنائیه است. این **Chable** (میز صندلی) تمام صفات خاصه اثنائیه را دارا می‌باشد.

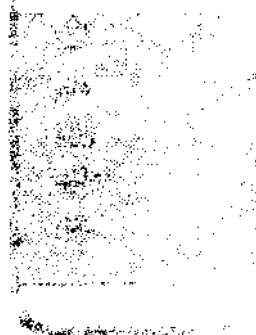


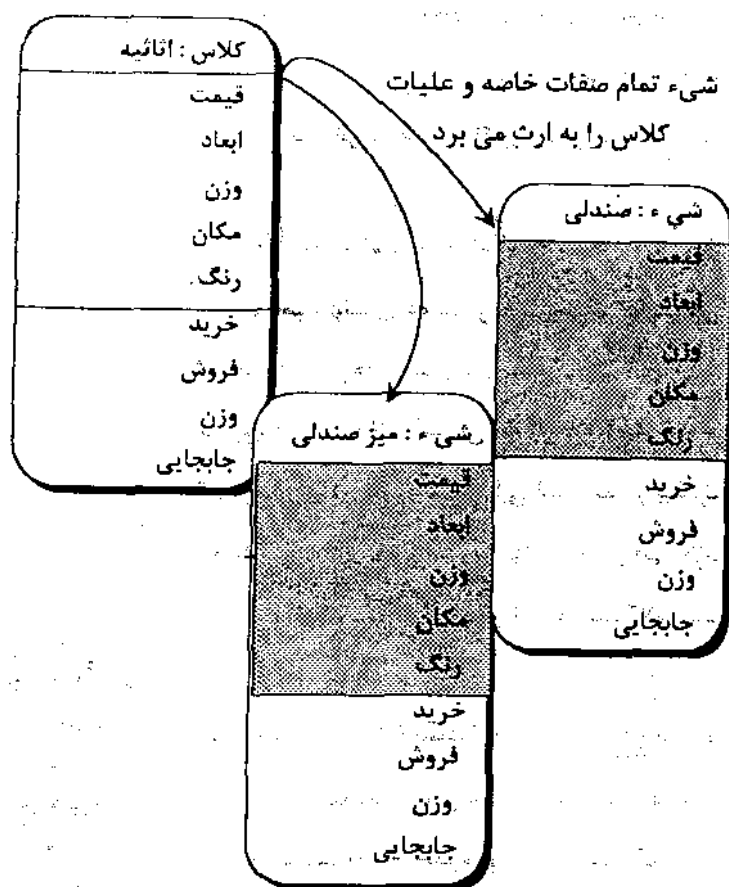
علامت مدل سازی داده می‌تواند برای بازتابی اشیاء و صفات خاصه آنها یکبار روند برای جزئیات بیشتر فصل ۱۲ را مرور کنید.

یا توصیف این صفات، برای تعریف حکایت‌گویی از این کلاس تلاش نمودیم، اما هنوز چیزی کم است. هر شیء، متعلق به کلاس اثنائیه را می‌توان به طرق مختلف تغییر داد. می‌توان آن را خرید و فروخت، به صورت فیزیکی تغییر داد. (مثلاً می‌توانید یک پایه آن را ببرید یا آن را قرمز کنید) یا از محلی به محل دیگر برد. هر کدام از این عملیات (یا عبارات دیگری مثل خدمات و شیوه‌ها) یک یا چند خصوصیت شیء را تغییر می‌دهد. مثلاً اگر صفات خاصه «محل» یک قلم عبارات داده‌ای مرکب باشد که به صورت زیر تعریف شده:

اتاق + طبقه + ساختمان = محل

پس عملیاتی به نام «انتقال» یا **Move** یک یا چند مورد را تغییر می‌دهد که صفت خاصه «محل» را می‌سازند. برای این کار، عمل جابه‌جایی باید دارای شناختی از این موارد باشد. عملیات انتقال و جابه‌جایی را می‌توان برای یک صندلی یا یک میز تا وقتی که هر دوی آنها مصداقی از کلاس اثنائیه هستند، استفاده نمود.





شکل ۲۰-۳ ارث بردن عملیات توسط اشیاء از کلاس

همه عملیات معتبر (مثل خرید، فروش، وزن) برای کلاس اناتیه مربوط به تعریف شیء است که در

شکل ۲۰-۳ نشان داده شده و به وسیله همه مصادیق و موارد این کلاس انتقال می یابند.

شیء مثل صندلی (و همه اشیاء به طور کلی) در برگیرنده اطلاعات (مثل ارزش های مشخصاتی که صندلی را تعریف می کنند)، عملیات (کارهای به کار گرفته شده برای تغییر مشخصه های صندلی)، دیگر اشیاء (اشیای مرکبی که می توان آن را تعریف نمود)، مقادیر ثابت (ارزش های مجموعه) و دیگر اطلاعات مربوطه است. خلاصه بندی کردن یا بست بندی کردن<sup>۴</sup> یعنی این که همه این اطلاعات تحت یک نام جمع بندی شده و می توان آن را به عنوان یک مشخصه یا جزئی از برنامه مجدداً استفاده کرد. [EVB89]<sup>۵</sup>

حالا که چند مفهوم مقدماتی معرفی شد، تعریف رسمی تری از شیء گرایی معنی دارتر جلوه می کند.

#### نقل قول

بسته بندی (کیسوله کردن) مانع از این می شود که وابستگی های داخلی یک برنامه موجب شود با تغییری اندک امواج عظیم پیامدها را شاهد باشیم. رامیاف و دیگران

1. buy

2. sell

3. weigh

4. Encapsulation

5. Object-Oriented requirement analysis (course notebook) EVB Software Engineering, 1989



کاد و یوردن این عبارت را به شکل زیر تعریف می‌کنند: [COA91]<sup>۱</sup>

شیء گرا = اشیاء + کلاس‌بندی + وراثت + ارتباط

سه تا از این مفاهیم هم‌اکنون معرفی شده‌اند. صحبت در مورد ارتباط را به بعد موکول می‌کنیم.

## ۲۰-۲-۱) کلاس‌ها و اشیاء

مفاهیم اولیه‌ای که منجر به طراحی یا کیفیت می‌شود، به‌طور یکسان در سیستم‌های توسعه یافته با استفاده از روش‌های متعارف و شیء‌گرا استفاده می‌شوند. به‌همین دلیل، مدل شیء‌گرای یک نرم‌افزار کامپیوتری باید حالات انتزاعی رویه‌ای و داده‌ها را که منجر به پیمانه‌سازی مؤثر می‌شود، نشان دهد. هر کلاس‌بندی یک مفهوم شیء‌گرایی است که در برگزیده انتزاع داده‌ای و رویه‌ای است که برای توصیف محتوا و رفتارهای برخی موجودیتهای جهان واقعی لازمند. تیلور [TAY90]<sup>۲</sup> از عبارت نشان داده شده در سمت راست شکل ۲۰-۴ برای توصیف یک کلاس استفاده می‌کند.



یک شیء، هم داده‌ها ( صفات مشخصه را) و هم فانکشن‌هایی ( عملیات شیوه‌ها یا خدمات) را بر روی داده‌ها کار می‌کنند، بسته‌بندی می‌کند.

تجزید داده‌ها که کلاس را تشریح می‌کند با "دیوار" از تجزید رویه‌ای احاطه شده است (که عملیات، شیوه‌ها<sup>۳</sup> یا خدمات<sup>۴</sup> نامیده می‌شود) و به نوعی داده‌ها را دستکاری می‌کند. تنها را دستیابی به این صفات، استفاده از یکی از روش‌هایی است که این دیواره را تشکیل می‌دهد. بنابراین این کلاس در برگزیده داده‌ها (داخل جداره) و پردازش است که داده‌ها را مورد تغییر قرار می‌دهد (روش‌هایی که دیواره را ایجاد می‌کنند). این کار باعث مخفی ماندن داده‌ها و کاهش تأثیر اثرات جانبی مربوط به تغییر می‌شود. از آنجا که این روش‌ها مایل به دست‌کاری تعداد محدودی از صفات خاصه می‌باشند، آنها منسجم‌اند و از آنجا که ارتباط تنها از طریق روش‌هایی واقع می‌شود که این دیواره را تشکیل می‌دهند، کلاس‌ها مایلند از دیگر عناصر سیستم جدا شوند. همه این مشخصه‌های طراحی منجر به نرم‌افزار با کیفیتی می‌شود.

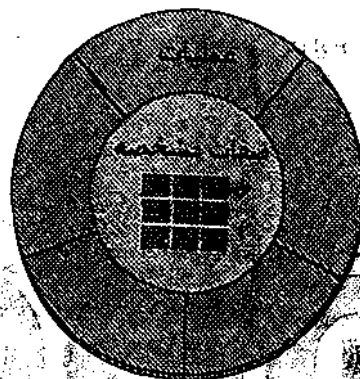
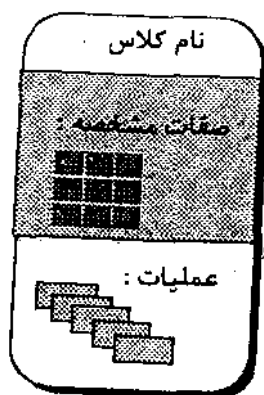
1. Coad, P. and E.

2. Taylor, D.A.

3. operations

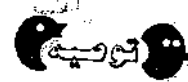
4. methodes

5. services



شکل ۲-۴ گزینه‌ها بی برای باز نمایی یک کلاس در بحث شیء گرای

اگر طور دیگری آن را بیان کنیم، هر کلاس یک توصیف کلی<sup>۱</sup> است (مثل لوح، الگو یا طرح کلی) که مجموعه‌ای از اشیای مشابه را توصیف می‌کند. در تعریف، همه اشیایی که در یک کلاس وجود دارند، صفات آن را به ارث برده و دارای همان عملیاتی هستند که برای تغییر صفات خاصه در دسترس می‌باشد. کلاس برتر<sup>۲</sup> مجموعه‌ای از کلاس‌هاست و یک کلاس فرعی<sup>۳</sup> نمونه‌ای از یک کلاس است. این تعاریف صرفاً نشان‌گر وجود سلسله مراتب یک کلاس است که در آن نگرش‌ها و عملیات کلاس بالاتر زیر مجموعه‌ای است که ممکن است هر کدام به شیوه‌ها و صفات اختصاصی اضافی بپردازند. یک سلسله مراتب صفات خاصه کلاس برای کلاس اثاثیه در شکل ۲-۵ آمده است.



یکی از نخستین اموری که هنگام ساخت سیستم (OO) به ذهن می‌رسد، چگونگی رده‌بندی اشیاء موجود در سیستم است.

#### ۲-۲-۲-۵ صفات خاصه

هم‌اکنون متوجه شده‌اید که صفات خاصه مشخصه‌ها به کلاس‌ها و اشیاء منصوب می‌شوند و کلاس‌ها شیء را به طریقی توصیف می‌کنند. مبحثی از این صفات خاصه توسط پوشامپ و همکارانش ارائه شده است: [CHA93]<sup>۴</sup>

شماهیت‌های جهان واقعی اغلب با کلیاتی توصیف می‌شوند که نشان‌گر مشخصه‌های ثابتی هستند. اکثر اشیای فیزیکی دارای مشخصه‌هایی همچون شکل، وزن، رنگ و نوع ماده هستند. انسان‌ها نیز دارای مشخصه‌هایی همچون تاریخ تولد، والدین، نام و رنگ چشم می‌باشند. ممکن است مشخصه‌ای مانند یک رابطه دودویی بین کلاس و داده معینی در نظر گرفته شود.


1. generalized description

2. superclass

3. subclass

4. de Champeaux, D.



  
مقادیری که به صفات  
خاصه بک شی  
تخصیص می یابند،  
(می توانند) آن را  
منحصر به فرد سازند.

[www.mitm-mobile.blogfa.com](http://www.mitm-mobile.blogfa.com)

## ۳-۲-۲۰ عملیات، شیوه ها و خدمات

هر شیء (که نمایانگر مجموعه‌ای از صفات خاصه است)، بسته‌بندی کننده داده‌ها و الگوریتم‌هایی است که این اطلاعات را پردازش می‌کنند. این الگوریتم‌ها را عملیات، شیوه ها و خدمات<sup>۱</sup> می‌نامند و می‌توان آنها را در حالت قراردادی به صورت پیمانه‌هایی مشاهده کرد.

هر یک از عملیات قرار گرفته در یک شیء، نمایانگر یکی از حالات آن شیء می‌باشد. مثلاً عملیات **Getcolor** در مورد خودرو، رنگ ذخیره شده در مشخصه رنگ را استخراج می‌کند. نشانه وجود این عملیات این است که خودرو برای دریافت محرکی (ما آن را پیام<sup>۲</sup> می‌نامیم) طراحی شده که نیازمند رنگ مدل خاصی از یک کلاس است. وقتی که شینی محرکی دریافت می‌کند، رفتار خاصی را شروع می‌نماید. این کار می‌تواند به سادگی باز یافت رنگ خودرو یا به پیچیدگی شروع یکسری محرک‌هایی باشد که در میان یکسری اشیای مختلف رواج یافته‌اند. در مورد بعدی، مثالی را در نظر بگیرید که در آن محرک اولیه توسط شیء شماره ۱ دریافت شده که منجر به تولید دو محرک دیگر می‌شود که به شیء ۲ و ۳ فرستاده می‌شوند. عملیات موجود در اشیای دوم و سوم روی محرک عمل کرده و اطلاعات لازم را به شیء اول می‌فرستند. سپس شیء شماره ۱ از اطلاعات ارجاع شده برای مرتفع نمودن حالت مورد تقاضای محرک اولیه، استفاده می‌کند.

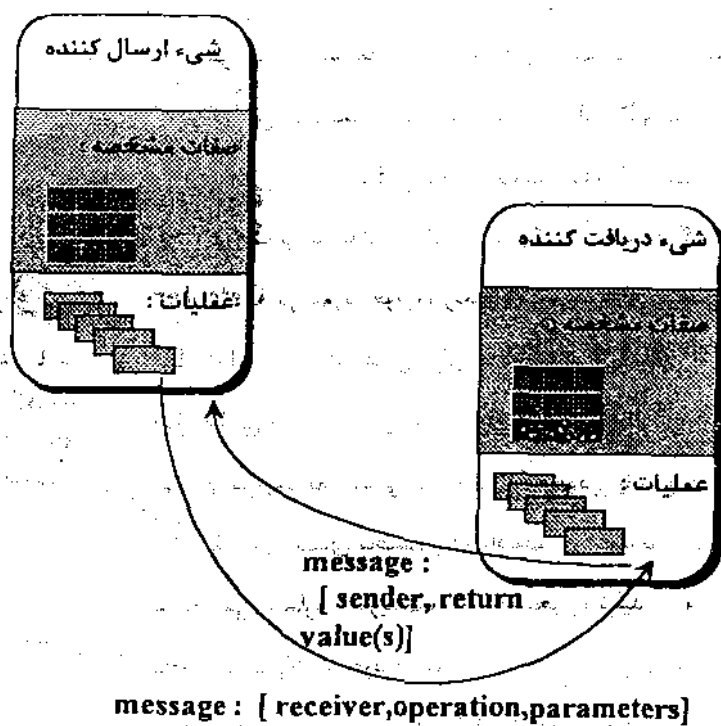


هرگاه یک شیء با یک پیام شبیه سازی شود، رفتارهایی را با اجرای عملیات از خود بروز خواهد داد.

## ۴-۲-۲۰ پیامها

پیامها وسیله‌ای هستند که توسط آن اشیاء به رابطه متقابل می‌پردازند. با استفاده از ترمینولوژی (واژگان) ارائه شده در بخش قبلی، یک پیام انجام رفتاری را در شیء گیرنده پیام، آنگاه می‌کند. این رفتار وقتی صورت می‌گیرد که عملیاتی اجرا می‌شود.

۱. در متن این بحث، ما اصطلاح عملیات را مورد استفاده قرار می‌دهیم، اما عموم افراد از واژگان شیوه و خدمات استفاده می‌کنند.



شکل ۶-۲۰ تبادل پیامها بین اشیاء

ارتباطات متقابل بین پیام‌ها به صورت شماتیک در شکل ۶-۲۰ آمده است. عملیاتی در یک شیء، فرستنده پیام، پیامی به صورت زیر ایجاد می‌کند:

**message: [destination, operation, parameters]**

پیام: (پارامترها، عملیات، مقصد)

#### نقل قول

پیامها و شیوه‌ها  
(عملیات) دوروی یک  
سکه می‌باشند. شیوه  
ها، رویه‌هایی هستند  
که یک شیء هنگام  
دریافت یک پیام آنها را  
فرا می‌خواند.  
گرم وس

که در آن مقصد<sup>۱</sup> مشخص کننده شیء دریافت کننده<sup>۲</sup> است که به وسیله پیام تحریک شده، عملیات به اقداماتی<sup>۳</sup> اشاره دارد که باید انجام گیرند و پارامتر اطلاعاتی مهیا می‌کند که برای موفقیت عملیات لازمند. به عنوان مثالی در مورد عبور پیام‌ها از یک سیستم شیء‌گرا، اشیایی را در نظر بگیرید که در شکل ۷-۲۰ نشان داده شده‌اند. چهار شیء A، B، C و D از طریق ارسال پیام‌ها با یکدیگر ارتباط برقرار می‌کنند. مثلاً اگر شیء B نیازمند پردازش مربوط به عملیات op10 شیء D باشد، پیامی به شکل زیر به D می‌فرستد:

**Message: [D, op10, {data}]**

به عنوان بخشی از اجرای op10، ممکن است شیء D پیامی به شکل زیر به C بفرستد:

1.destination

2.receiver object

3.operation refers

Message : [C, op8, {data}]

شیء C، op10 را پیدا کرده و آن را اجرا نموده و سپس مقدار مرجوعی مناسب را به D می‌فرستد.

عملیات op10 تکمیل شده و مقدار مرجوعی را به B می‌فرستد.

کاکس [COX86] تبادل بین اشیاء را به صورت زیر تعریف می‌کند:

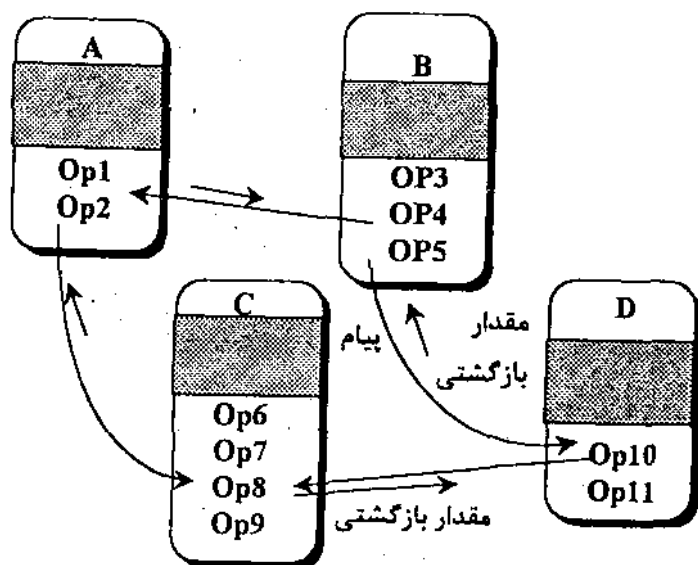
از شیئی می‌خواهند با دریافت پیامی که می‌گوید این شیء چه کاری انجام دهد، یکی از عملیات خود

را انجام دهد. دریافت کننده ابتدا با انتخاب عملیاتی که نام پیام را اجرا می‌کند، این عملیات را اجرا نموده و

سپس کنترل را به فراخواننده یا Caller می‌سپارد.

وضعیت ارسال و دریافت پیام، سیستم‌های شیءگرا را به یکدیگر متصل می‌کند. پیام‌ها دیدگاهی در

مورد وضعیت هر یک از اشیاء و سیستم شیءگرا به طور کلی، ارائه می‌دهند.



شکل ۲۰-۷ مثالی از تبادل پیامها

#### ۲۰-۲-۵ بسته بندی، وراثت و چندریختی (پلی مورفیزم)

گرچه ساختار و ترمینولوژی معرفی شده در بخش‌های ۲۰-۲-۱ تا ۲۰-۲-۴ در سیستم‌های شیءگرا

نسبت به همتایان سنتی - متعارف و قراردادی‌شان (سیستم‌های ساخت یافته) تمایز ایجاد می‌کند، اما وجود

سه مشخصه در این سیستم‌ها آنها را منحصر بفرد می‌گرداند. همان گونه که توجه کردید، کلاس شیءگرا و

اشیای منشعب از این کلاس دارای داده‌ها و عملیاتی هستند که در یک بسته روی این اطلاعات کار

می‌کند. این عمل چند مزیت مهم را مهیا می‌سازد:

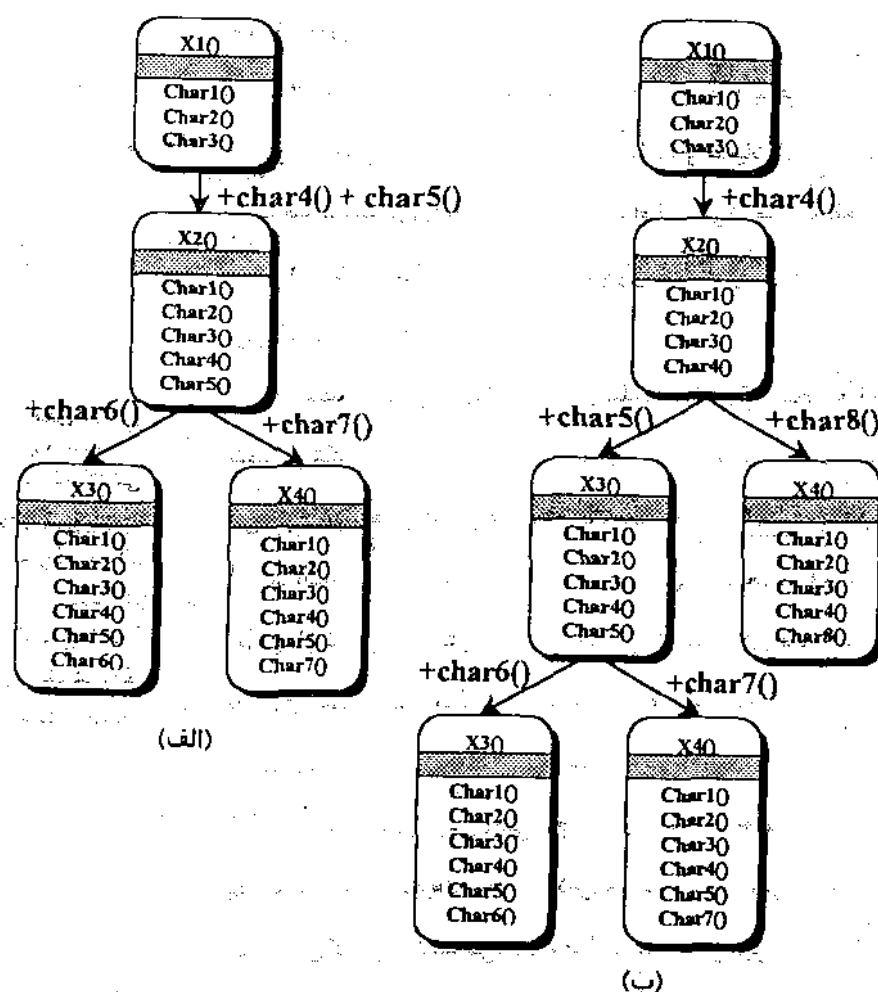




مزایای عمده یک  
معماری (OO)  
چیست؟

- جزئیات داخلی پیاده‌سازی اطلاعات و رویه‌ها که از چشم جهان خارج مخفی است (مخفی‌سازی اطلاعات). این کار اشاعه اثرات جانبی را در هنگام تغییر کاهش می‌دهد.
- ساختار داده‌ها و عملیاتی که آنها را تغییر می‌دهند در یک ماهیت نام‌گذاری شده ادغام می‌شوند که همان کلاس است. این کار استفاده مجدد از هر جزء را تسهیل می‌کند.
- رابط‌های میان اشیای تلفیق شده، ساده می‌شوند. شیئی که پیامی می‌فرستد، لازم نیست در مورد جزئیات ساختارهای اطلاعاتی داخلی، نگران باشد. بنابراین، ایجاد رابطه ساده شده و پیوستگی سیستم کاهش می‌یابد.

وراثت یکی از ایجاد تمایزکنندگان مهم بین سیستم‌های شیء‌گرا و قراردادی است. زیر زیرکلاس Y دارای تمام صفات خاصه و عملیات مربوط به زیرمجموعه X است. یعنی این که همه ساختارهای اطلاعاتی و الگوریتم‌ها که اساساً برای X طراحی و پیاده‌سازی شده‌اند، بلافاصله برای Y نیز مهیا هستند و کار بیشتری برای انجام دادن وجود ندارد. استفاده مجدد به‌طور مستقیم صورت می‌گیرد.



شکل ۲۰-۸ سلسله مراتبی از کلاسها:

(الف) اولیه

(ب) بازسازی شده

هرگونه تغییری در داده‌ها یا عملیات موجود در کلاس بالاتر بلافاصله به تمام زیر مجموعه‌هایی که از کلاسی اصلی مشتق شده‌اند، به ارث می‌رسد. بنابراین، سلسله مراتب کلاس مکانیزمی می‌شوند که در طول آن تغییرات (در سطح بالایی) می‌توانند از طریق یک سیستم منتشر شوند.

نکته مهم قبل توجه این است که در هر سطحی از سلسله مراتب کلاس، صفات خاصه و عملیات جدید، به آنهایی که از سطوح بالاتر سلسله مراتب آمده‌اند، ارث می‌رسد. در واقع، هرگاه کلاس جدیدی ایجاد می‌شود، مهندس نرم‌افزار چند گزینه در اختیار دارد:

۱. اصطلاح اولاد و اجداد [JAC92] برخی اوقات به جای کلاس فرعی و کلاس برتر به کار می‌رود.

• این کلاس را می‌توان از ابتدا و از روی پیش‌نویس، طراحی و تولید نمود. یعنی وراثت استفاده نمی‌شود.

• می‌توان سلسله مراتب کلاس‌بندی را مورد جستجو قرار داد تا مشخص شود آیا کلاسی که در مراتب بالاتر قرارداد حاوی اکثر صفات خاصه و عملیات لازم است یا خیر؟ کلاس جدید از کلاس بالاتر صفات خاصه‌ای را به ارث برده و موارد اضافی نیز در صورت تقاضا ممکن است افزایش یابند.

• سلسله مراتب هر کلاس را می‌توان مجدداً بازسازی نمود، به‌طوری‌که صفات خاصه و عملیات برای کلاس جدید پیاده‌سازی شود.

به‌منظور تشریح چگونگی بازسازی سلسله مراتبی که منجر به کلاس دلخواه می‌شود، مثال نشان داده شده در شکل ۸-۲۰ را در نظر بگیرید. این سلسله مراتب که در شکل (۸-۲۰ الف) تشریح شده ما را قادر می‌سازد. کلاس‌های X3 و X4 را با مشخصات ۱، ۲، ۳، ۴، ۵ و ۶ و ۱، ۲، ۳، ۴، ۵ و ۷ به‌ترتیب به‌دست آوریم اکنون فرض کنید که کلاس جدیدی تنها با مشخصه ۱، ۲، ۳، ۴ و ۸ موردنظر است. برای به‌دست آوردن آن که در این مثال آن را X2b می‌نامیم، ممکن است سلسله مراتب به‌صورت شکل (۸-۲۰ ب) بازسازی شوند. نکته مهم توجه به این نکته است، که بازسازی سلسله مراتب می‌تواند مشکل باشد و به‌همین دلیل، گاهی از باطل کردن استفاده می‌شود.

در اصل، باطل کردن وقتی رخ می‌دهد که صفت خاصه و عملیاتی به شیوه معمول به ارث برسند اما بعد تغییر کنند تا نیازهای خاص کلاس جدید را مرتفع سازند. همان‌گونه که جاکوب بیان می‌دارد وقتی از عمل باطل کردن استفاده می‌شود، «عمل وراثت انتقال‌پذیر نیست» [JAK92].<sup>۱</sup>

در بعضی موارد، انتقال بعضی از صفات خاصه و عملیات به همدیگر، وسوسه‌برانگیز است. این عمل را وراثت چندگانه<sup>۲</sup> می‌نامند و بحث‌برانگیز است. به‌طور کلی، وراثت چندگانه سلسله مراتب را پیچیده‌تر ساخته و مشکلات بالقوه‌ای در کنترل پی‌کریبندی ایجاد می‌کند. از آنجا که توالی وراثت چندگانه از نظر پی‌گیری مشکل‌تر است، تغییراتی در تعریف کلاس که در مراتب بالا قرار می‌گیرند، تأثیر بی‌هدفی بر کلاس‌های پایین‌تر معماری است.

پلی مورفیسم<sup>۴</sup> یا چند ریختی مشخصه‌ای است که تا حد زیادی کارهای لازم برای بسط سیستم شیء‌گرای موجود را کاهش می‌دهد. به‌منظور درک پلی مورفیسم، یک برنامه کاربردی قراردادی را در نظر بگیرید که باید چهار نوع نمودار مختلف را ترسیم کند: نمودارهای خطی، گرد یا مدور، هیستوگرام و دیاگرام‌های Kiviat (مطابق متن کتاب اصلی<sup>۳</sup>م)، به‌طور مطلوب، وقتی اطلاعات برای نوع به‌خصوصی از

### نقل قول

در حالیکه یک شیء موجودیتی صلب است با بعد زمانی و مکانی، یک رده (کلاس) تنها بیانگر نچرید است. جوهر یک شیء، آن‌گونه که هست، گریزی بوج

### نقل قول

هرچند ممکن است نام چندریختی زمخت و بی‌لطفات به نظر رسد، اما مکانیزمی لطیف و ظریف دارد.

1. overriding

2. Jakobson, I.

3. multiple inheritance

4. polymorphism

نمودار جمع‌آوری شدند، بهتر آن است که نمودار خودش را بتواند بکشد، برای نیل به این هدف در برنامه کاربردی (و حفظ انسجام پیمانه)، لازم است پیمانه‌های طراحی برای هر نوع نمودار ارائه داد. سپس در داخل طراحی هر نوع نمودار، منطق کنترلی مثل موارد زیر قرار می‌گیرد:

```
case of graphtype:
  if graphtype = linegraph then DrawLineGraph (data);
  if graphtype = plechart then DrawPieChart (data);
  if graphtype = histogram then DrawHisto (data);
  if graphtype = kiviatt then DrawKiviat (data);
end case;
```

گرچه این طراحی به‌صورت منطقی صحیح است، اما افزودن انواع نمودارهای جدید مشکل‌زا است. باید پیمانه ترسیم جدیدی برای هر نمودار ایجاد شده و سپس منطق کنترلی فوق باید برای هر نمودار به روز شود.

برای حل مشکل فوق، هر یک از نمودارهای مورد توجه، کلاسی فرعی از یک کلاس کلی به نام Graph می‌شود. با استفاده از مفهوم استفاده بیش از حد<sup>۱</sup> گراباری هر کلاس عملیاتی را به نام Draw (ترسیم) تعریف می‌کند. شیء می‌تواند یک پیام Draw برای هر یک از اشیای ذکر شده از هر نوع کلاس، ارسال کند. شیء دریافت‌کننده پیام، عملیات Draw خود را برای ایجاد نمودار مناسب تحریک می‌کند. بنابراین طراحی فوق به حالت زیر کاهش می‌یابد.

### Graphtype Draw

وقتی نموداری جدیدی قرار است به سیستم افزوده شود، یک مجموعه فرعی یا زیرمجموعه با عملیات ترسیم خودش ایجاد می‌شود. اما لازم نیست تغییراتی در خود شیء که می‌خواهد ترسیم<sup>۲</sup> شود، ایجاد گردد زیرا پیام Graphtype Draw بدون تغییر می‌ماند. به‌طور خلاصه، چند ریختی یا پلی مورفیزم تعدادی از عملیات‌های مختلف را قادر می‌سازد تا دارای اسم یکسانی باشند. این کار به نوبه خود اشیای را از یکدیگر جدا نموده و هر کدام را مستقل تر می‌سازد.

### ۲۰-۳ شناسایی عناصر مدل شیئی

عناصر یک مدل شیئی یعنی کلاس‌ها و اشیای، صفات خاصه، عملیات و پیام‌ها، هر کدام در بخش قبلی تعریف و مورد بحث قرار گرفتند. اما چگونه این عناصر را در مسئله واقعی شناسایی کنیم؟ بخش‌های بعدی نمایان‌گر یک سری از رهنمودهای غیررسمی است که به شناسایی عناصر مدل شیئی کمک می‌کنند.

#### نقل قول

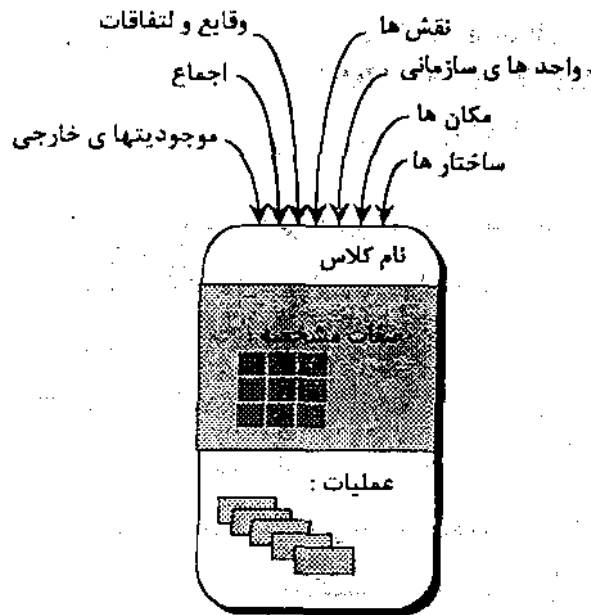
مسائل واقعا مشکل  
(در ۱۰۰ در همان  
مرحله نخست، با کشف  
اشیاء صحیح قابل حل  
خواهند شد.  
کارل آرجیلا

1. overloading

2. draw

## ۲۰-۳-۱ شناسایی کلاس‌ها و اشیاء

اگر به اطراف اتاق‌تان نگاهی بیندازید، مجموعه‌ای از اشیای فیزیکی وجود دارد که به راحتی شناسایی، کلاس‌بندی و تعریف می‌شوند (از نظر صفات خاصه و عملیاتشان). اما وقتی به جوانب یک مشکل مربوط به نرم‌افزار نگاه می‌کنید، درک مشکلات ممکن است سخت‌تر باشند.



شکل ۲۰-۹ اشیاء چگونه خود را جلوه گر می‌سازند

می‌توانیم شناسایی اشیاء<sup>۱</sup> را با بررسی وضعیت مسئله یا با انجام یک تجزیه گرامری روی روند پردازش سیستم در حال ساخت، آغاز کنیم. اشیاء یا مشخص کردن هر اسم یا عبارت اسمی و وارد کردن آن در یک جدول ساده، تعیین می‌شوند. باید به مترادفها توجه نمود. اگر لازم است شیئی راه‌حلی را پیاده‌سازی کند، پس این شیء بخشی از فضای راه‌حل است. در غیر این صورت، اگر لازم است شیئی تنها یک راه‌حل را توصیف کند، بخشی از فضای مسئله است. پس وقتی همه اسمها جدا شدند باید در جستجوی چه چیزی باشیم؟ اشیاء، خود را به صورت شیوه‌های به نمایش درآمده در شکل ۲۰-۹، جلوه‌گر می‌سازند. اشیاء می‌توانند به صورت زیر باشند:

- موجودیت‌های خارجی<sup>۲</sup> (مانند سیستمها، وسایل و افراد دیگر) که اطلاعات مورد استفاده توسط سیستم مبتنی بر کامپیوتر را تولید یا مصرف می‌کنند.
- اجسام<sup>۱</sup> که بخشی از دامنه اطلاعاتی برای مسئله هستند (مثل گزارشات، نمایش‌ها، نامه‌ها و علائم).



چگونه می‌توانم همراه با مطالعه مسئله مورد حل، اشیاء را نیز شناسایی نمایم؟

۱. در واقع، تحلیل شیء گرا عملاً سعی دارد که تعریفی از کلاسها ارائه نماید تا اشیاء در آن قرار گیرند. بنابراین، وقتی ما اشیاء بالقوه (کاندیدسم) را جدا می‌سازیم، اعضای بالقوه کلاسها را تعیین نموده ایم.

2 External entities

• وقایع و اتفاقات<sup>۲</sup> (مثل انتقال اموال یا تکمیل یک سری حرکات ربات) که در بطن عملیات

سیستم رخ می دهند.

• نقش های ایفا شده<sup>۳</sup> توسط افرادی که با سیستم در ارتباط متقابل هستند (مثل مدیر،

مهندس و فروشنده).

• واحدهای سازمانی<sup>۴</sup> (مثل بخش، گروه و تیم) که به برنامه ارتباط دارند.

• مکان ها<sup>۵</sup> (مثل محل ساخت یا اسکله بارگیری) که باعث مسئله را به وجود آورده و عملکرد

کلی سیستم را می سازند.

• ساختارها<sup>۶</sup> (مثل سنسورها، وسایط نقلیه چهار چرخ و کامپیوترها) که کلاسی از اشیا را

تعریف کرده یا در نهایت به کلاس بندی اشیا مرتبط می شوند.

طبقه بندی های فوق، تنها یکی از مولد بسیاری است که در اختیارات پیشنهاد می گردند.

نکته مهم مورد توجه دیگر این است که اشیا چه چیزی نیستند. به طور کلی، یک شیء نباید دارای

یک اسم به شیوه تحکمی یا امری باشد. مثلاً، اگر تولیدکنندگان یک نرم افزار برای یک سیستم

تصویربرداری پزشکی که به عنوان یک شیء تعریف شده اسم «معکوس سازی<sup>۷</sup> تصویر» یا را انتخاب کنند،

یک اشتباه ظریفی را مرتکب شده اند. البته تصویر بدست آمده از این نرم افزار می تواند یک شیء باشد. (این

جسمی است که بخشی از دامنه اطلاعات است). معکوس سازی تصویر عملیاتی است که در مورد شیء

به کار می رود: [BUD96]<sup>۸</sup>

احتمال دارد که معکوس سازی به عنوان عملیاتی برای تصویر شیء استفاده شود، اما به عنوان یک شیء

جداگانه برای القای «معکوس سازی تصویر» تعریف نمی شود. همان گونه که کشمن [CAS89]<sup>۹</sup> می گوید:

«هدف از شیء گرایی عبارتست از بستهبندی، اما هنوز اطلاعات و عملیات انجام شده روی آنها باید مجزا

باشند»

1. Things

2. Occurrences or events

در این متن، اصطلاح رویداد به هر وقعه ای اطلاق می شود. ضرورتی بر پیاده سازی کنترل های فصل ۱۲ وجود ندارد.

3. Roles

4. Organizational units

5. places

6. Structures

7. inversion

8. Budd, T.

9. Cashman, M.





یک تجزیه گرامری  
برای جداسازی بالقوه  
اشیاء ( اسمها ) و  
عملیات ( فعلها ) به کار  
می رود...

به منظور تشریح این که چگونه اشیاء ممکن است در طول مراحل اولیه تحلیل تعریف شوند، به مثال مطرح شده در مورد سیستم ایمنی خانه امن برمی گردیم. در فصل ۱۲، یک تجزیه دستوری روی شیوه پردازش سیستم خانه امن انجام دادیم. جریان پردازش به صورت زیر بازآفرینی می شود:

نرم افزار خانه امن کاربر خلنگی را قادر می سازد، بیکرندگی سیستم را هنگام نصب آن انجام دهد، تمام سنسورهای متصل به سیستم امنیتی را نظارت کرده و از طریق صفحه کلید و کلیدهای عملیاتی که در صفحه کنترل خانه امن در شکل ۲-۱۱ آمده با کاربر خلنگی ارتباط برقرار کند.

در طول نصب، صفحه کنترل خانه امن در برنامه استفاده شده و از سیستم استفاده می کند. هر سنسور دارای یک شماره و نوع است. یک کلمه عبور اصلی برای آماده سازی و غیر آماده سازی سیستم برنامه ریزی شده و شماره تلفن هایی برای تماس در هنگام رخ دادن مشکلی برای سنسور، در آن قرار داده شده اند.

وقتی مشکلی برای سنسور پیش آمده و نرم افزار آن را حس می کند، یک بوق هشدار دهنده که به سیستم متصل است به صدا در می آید. بعد از یک تأخیر زمانی که توسط کاربر در طول فعالیت های نصب و راه اندازی سیستم مشخص شده، نرم افزار شماره تلفن سرویس کنترل کننده را می گیرد و اطلاعاتی در مورد محل و ماهیت حادثه واقع شده را گزارش می کند. در صورت عدم تماس این شماره هر ۲۰ ثانیه دوباره گرفته می شود تا تماس برقرار شود.

تمام ارتباطات با خانه امن توسط سیستم فرعی ارتباطی کاربر سازمان دهنی و مدیریت می شود که داده های ورودی ارائه شده از طریق صفحه کلید و کلیدهای عملیاتی را خوانده، پیام های ارسالی را روی صفحه نمایش گر نمایش می دهد و اطلاعات وضعیت سیستم را روی صفحه می آورد. ارتباط از طریق صفحه کلید به شکل زیر صورت می گیرد.

با داشتن نامها می توانیم تعدادی اشیاء بالقوه را پیشنهاد نماییم:

شیء / کلاس بالقوه	کلاس بندی کلی
مالک خانه	نقش یا موجودیت خارجی
سنسور	موجودیت خارجی
صفحه کنترل	موجودیت خارجی
نصب	رخداد
سیستم (سیستم امنیتی)	چیز
شماره نوع	شیء نیست: صفت خاصه سنسور
رمز عبور اصلی	چیز
شماره تلفن	چیز
رویداد سنسور	رخداد
آژیر هشدار	موجودیت خارجی
خدمات نظارتی	واحد سازمانی یا موجودیت خارجی

فهرست فوق تا وقتی ادامه دارد که همه اسم‌های مورد پردازش در نظر گرفته شده باشند. توجه کنید که هر مدخل ورودی فهرست را یک شیء بالقوه می‌نامیم. باید قبل از تصمیم‌گیری نهایی هر کدام از آنها را بیشتر بررسی کنیم.

کُد و یوردون [COA91]<sup>۱</sup> شش مشخصه انتخابی را بیان می‌دارند که باید یک تحلیلگر در هنگام بررسی هر شیء بالقوه برای گنجاندن در مدل تحلیلی آنها را استفاده کند:

۱- اطلاعات حفظ شده. شیء بالقوه در طول تحلیل تنها در صورتی مفید خواهد بود که اطلاعات مربوط به آن یادآوری شوند به‌طوری‌که سیستم بتواند عمل کند.

۲- خدمات لازم. شیء بالقوه باید دارای مجموعه‌ای از عملیات قابل شناسایی باشد که بتواند ارزش صفات خاصه را به شکلی تغییر دهد.

۳- مشخصه‌های چندگانه. در طول تحلیل نیازمندیها، نقطه تمرکز باید روی اطلاعات عمده باشد. شینی با یک صفت خاصه ممکن است در واقع در طول طراحی مفید باشد، اما احتمالاً بهتر آن است که در طول کار تحلیل، به‌عنوان یک صفت‌ها خاصه شیء دیگر در نظر گرفته شود.

۴- مشخصه‌های مشترک. مجموعه‌ای از صفات خاصه را می‌توان برای شیء بالقوه تعریف نمود و این صفات خاصه در تمام وقایع مربوط به شیء به‌کار می‌روند.

۵- عملیات مشترک. مجموعه‌ای از عملیات را می‌توان برای شیء بالقوه تعریف نمود و این عملیات در همه وقایع مربوط به شیء به‌کار می‌روند.

۶- نیازمندیهای ضروری. موجودیتهای خارجی که در فضای مسئله ظاهر می‌شوند و اطلاعاتی تولید یا مصرف می‌کنند که برای عملیات هرگونه راه‌حلی در مورد سیستم ضروری هستند، تقریباً همیشه به‌عنوان اشیایی در مدل نیازمندیها تعریف می‌شوند.

اگر یک شیء بالقوه بخواهد به‌عنوان یک شیء قانونی برای قرار گرفتن در مدل نیازمندیها در نظر گرفته شود، باید تمام مشخصه‌های فوق را دارا باشد. تصمیم‌گیری در مورد اضافه‌کردن اشیای بالقوه در مدل تحلیلی تا حدی نظری است و تکامل بعدی ممکن است باعث شود که یک شیء کنار گذاشته شده یا به وضع اول باز گردد. اولین مرحله OOA باید تعریفی از اشیای بوده و تصمیماتی اتخاذ شود. با در نظر گرفتن اینها، باید مشخصه‌های انتخابی را در فهرستی از اشیای بالقوه خانه امن قرار دهیم:



چگونه اطمینان حاصل کنیم که یک شیء بالقوه، کاندید مناسبی برای یک سیستم (OO) می‌باشد؟



یک شیء بالقوه در صورتی که در مدل تحلیلی بکار می‌رود، باید تمام یا بیشتر ویژگیها را برآورده سازد

شیء / کلاس بالقوه : شماره ویژگی‌هایی که در بردارد

مالک خانه	مردود: عدم رعایت (۱ و ۲) حتی با رعایت ۳-۶
سنسور	پذیرفته: رعایت تمام موارد
صفحه کنترل	پذیرفته: رعایت تمام موارد
نصب	مردود
سیستم (سیستم امنیتی)	پذیرفته: رعایت تمام موارد
شماره نوع	مردود: عدم رعایت ۲، صفت خاصه سنسور
سنسور	مردود: عدم رعایت ۳
رمز عبور اصلی	عدم پذیرش: ۳
شماره تلفن	پذیرفته: تمام
رویداد سنسور	عدم پذیرش: ۳
آزیر هشدار	پذیرفته: رعایت ۲، ۳، ۴، ۵، ۶
خدمات نظارتی	مردود: عدم رعایت (۱ و ۲) حتی با رعایت ۳-۶

باید توجه داشت که: (۱) فهرست فوق در برگرفته همه چیز نیست، باید چند شیء اضافه برای تکمیل کردن مدل اضافه شود. (۲) بعضی از اشیای رد شده به صورت صفت‌های برای دیگر اشیای پذیرفته شده می‌شوند (مثلاً شماره و نوع، صفت‌های برای سنسور هستند و کلمه رمز و شماره تلفن صفت‌های برای سیستم می‌شوند). (۳) حالات مختلف مسئله ممکن است باعث تصمیمات مختلفی در مورد رد یا قبول آن شوند (مثلاً اگر هر کاربر خانگی دارای رمز منحصر به فرد بوده و نیز به وسیله نوع صدا شناسایی شود، شیء مالک‌خانه مشخصه‌های ۱ و ۲ را رعایت نموده و بدین ترتیب پذیرفته می‌شود).

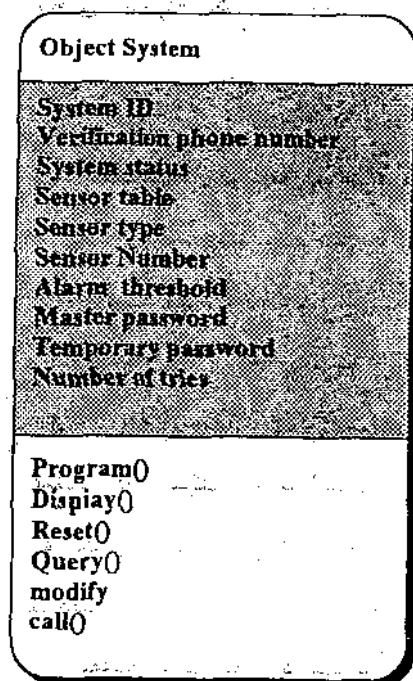
## ۲۰-۳-۲ مشخص سازی صفات خاصه

صفات خاصه شیء انتخابی برای قرار گرفتن در مدل تحلیلی را توصیف می‌کنند. در اصل، این صفات خاصه هستند که شیء را تعریف می‌کنند، این وضوح و روشنی چیزی است که منظور از شیء در بستر فضای مسئله است. مثلاً، اگر قرار بود سیستمی بسازیم که آمارهای بسپال را در مورد بازیکنان حرفه‌ای پی‌گیری کند، خصوصیات شیء **Player** (بازیکن) کاملاً از خصوصیات همان شیء وقتی که در بستر سیستم بازیابی بسپال حرفه‌ای به کار می‌رفت، متفاوت می‌شد. در مورد اولی، مشخصاتی هم‌چون اسم، پست، میانگین ضربات، درصد مورد استفاده قرار گرفتن در زمین، سال‌های بازی و بازی‌هایی که در آنها حضور داشته است، با مسئله در ارتباط می‌باشد. در مورد دوم، بعضی از این صفات خاصه دارای



صفات خاصه با آزمایش صورت مسئله به دست می‌آیند. در جستجوی چیزهایی که کاملاً یک شیء را تعریف کنند، و آنرا منحصر به فرد سازند.

مفهوم هستند اما بقیه را با صفات خاصه ای هم چون میانگین دستمزد، اعتبار نسبت به واگذاری کامل، گزینه های طرح بازنشستگی که انتخاب شده اند، آدرس پستی و غیره عوض می کنیم.



شکل ۲۰-۱۰ شیء سیستم به همراه عملیات الحاقی آن

به منظور ارائه مجموعه با مفهومی از صفات خاصه در مورد یک شیء، تحلیل گر می تواند دوباره شیوه حکایت پردازشی مسئله را بررسی نموده و چیزهایی را که به طور منطقی متعلق به شیء می باشند، بررسی کند. به علاوه، باید در مورد هر شیء به سؤال زیر پاسخ داد: چه قلم های اطلاعاتی (مرکب و/ یا ابتدایی) به طور کامل این شیئی را در بستر مسئله مورد بررسی تعریف می کنند؟

به منظور توضیح شیء سیستم (System) را در نظر می گیریم که برای خانه امن تعریف شده است. پیش تر متوجه شدیم که کاربر خانگی می تواند سیستم امنیتی را طوری پیکربندی کند تا اطلاعات سنسور، اطلاعات واکنش زنگ هشدار، اطلاعات فعال سازی/ غیر فعال سازی، اطلاعات شناسایی و غیره را منعکس کند. با استفاده از محتوای عبارت توصیفی تعریف شده برای فرهنگ داده و آنچه که در فصل ۱۲ آمده است، می توانیم این قلم های داده ای مرکب را به شکل زیر نشان دهیم:

آستانه زنگ خطر + شماره آن + نوع سنسور = اطلاعات سنسور

نوع زنگ خطر + شماره تلفن + زمان تأخیر = اطلاعات واکنش زنگ خطر

کلمه رمز موقت + تعداد آزمونهای مجاز + کلمه رمز اصلی = اطلاعات فعال سازی/ غیر فعال سازی

وضعیت سیستم + شماره تلفن صحت و تأییدیه + سیستم شناسایی = اطلاعات شناسایی

هر یک از قلم‌های داده‌ای در سمت راست علامت تساوی را در سطح مقدماتی می‌توان بیشتر تعریف نمود. اما از نظر اهداف ما، آنها یک فهرست منطقی از صفات خاصه برای شیء سیستم را تشکیل می‌دهند (بخش سایه زده شکل ۲۰-۱).

### ۳-۳-۲۰ تعریف عملیات

عملیات بیان گر رفتار یک شیء بوده و به شکلی مشخصه‌های شیء را تغییر می‌دهند. به‌طور دقیق‌تر، هر عملیات یک یا چند مقدار مشخصه را که در شیء وجود دارند، تغییر می‌دهد. بنابراین، باید یک عملیات از موجودیت مشخصات شیء شناخت داشته و باید به شیوه‌ای اجرا گردد که آن را قادر سازد در ساختار داده‌هایی که از این مشخصات بدست آمده‌اند، تغییر ایجاد کند.

گر چه انواع مختلفی از عملیات وجود دارد، اما معمولاً آنها را به سه کلاس تقسیم می‌کنند: (۱) عملیاتی که به شکلی داده‌ها را تغییر می‌دهد (مثل افزودن، حذف، قالب بندی مجدد، انتخاب) (۲) عملیاتی که محاسبه را انجام می‌دهند و (۳) عملیاتی که شیء را از نظر وقوع حادثه کنترلی، مورد مشاهده قرار می‌دهند.



آیا راه مستدلی برای طبقه بندی عملیات یک شیء وجود دارد؟

به‌عنوان اولین تکرار در بدست آوردن مجموعه‌ای از عملیاتی که برای اشیای مدل تحلیلی هستند، تحلیل گر دوباره می‌تواند وضعیت حکایت پردازشی را برای مسئله بررسی نموده و عملیاتی را انتخاب کند که به‌طور منطقی به شیء تعلق دارند. برای حصول این امر، تجزیه دستوری دوباره بررسی شده و فعل‌ها جدا می‌شوند. بعضی از این افعال همان عملیات قانونی هستند و می‌توانند به‌راحتی به یک شیء مشخص متصل شوند. مثلاً از روی شیوه حکایت پردازشی خانه امن که پیش‌تر در این فصل ارائه شد، می‌بینیم که سنسور دارای یک شماره و نوع است یا مسلح کردن و غیر مسلح کردن سیستم یک کلمه رمز اصلی برنامه‌ریزی شده است. این دو عبارت نشان گر یک سری چیزهاست:

- عملیات اختصاص با شیء سنسور مرتبط است.
- عملیات برنامه در شیء سیستم به‌کار می‌رود.
- عملیات مسلح و غیر مسلح کردن عملیاتی هستند که در سیستم به‌کار می‌روند.

(همین‌طور نهایتاً وضعیت سیستم با استفاده از عبارت فرهنگ‌نامه داده‌ای تعریف می‌شود)

مانند:

system status=[armed|disarmed]

[غیرمسلح / مسلح] = وضعیت سیستم

با تحقیق بیشتر این احتمال وجود دارد که عملیات برنامه<sup>۱</sup> به یکسری عملیات فرعی خاص تر تقسیم شود که برای طراحی سیستم ضروری است. مثلاً، برنامه بر روی شماره تلفن های مشخص عمل می کند. پیکربندی مشخصه های سیستم را انجام می دهد (مثل ایجاد جدول سنسور، وارد کردن مشخصه های رنگ خطر) و وارد کردن کلمات رمز را عهده دار است. اما هم اکنون ما برنامه را، به عنوان یک عملیات منفرد مشخص می نماییم.

#### ۲۰-۳-۴. نهایی کردن تعریف اشیاء

تعریف عملیات آخرین مرحله در تکمیل مشخص سازی یک شیء است. در بخش ۲۰-۳-۳، عملیات از روی تجزیه و تحلیل دستوری حکایت پردازشی سیستم، گلچین شده بودند. عملیات دیگر را ممکن است با در نظر گرفتن تاریخچه زندگی یک شیء و پیامهایی که در میان اشیای تعریف شده برای سیستم رد و بدل می شوند، [COA91]<sup>۱</sup> تعیین نمود.

تاریخچه کلی زندگی یک شیء را می توان با شناخت این مسئله که شیء باید ایجاد شده، تغییر یافته، اصلاح شده یا به طرق دیگری خوانده یا احتمالاً حذف شود، تعریف نمود. در مورد شیء سیستم، این کار را می توان بسط داد تا منعکس کننده فعالیت های شناخته شده ای باشد که در طول دوره آن رخ می دهد (در این مورد، در طول زمانی که خانه امن کار می کند). بعضی از عملیات را می توان از روی ارتباط احتمالی بین اشیاء حتمی نمود. مثلاً، رویداد گیرنده (حسگر) پیامی به سیستم می فرستد تا محل این حادثه و تعداد آن را مشخص سازد. صفحه کلید کنترل، یک پیام تنظیم مجدد برای سیستم می فرستد تا وضعیت سیستم را بهنگام یا به روز سازد. یک زنگ هشدار دهنده پیامی پرسشی ارسال می دارد. صفحه کلید کنترلی یک پیام اصلاحی برای تغییر یک یا چند صفت خاصه بدون تنظیم مجدد کل شیء سیستم می فرستد. بر اثر حادثه رخ داده برای سنسور یک پیام برای برقراری ارتباط تلفنی ارسال می شود که این شماره در شیء قرار دارد. دیگر پیام ها بررسی شده و عملیاتی حاصل می شوند. تعریف بدست آمده از شیء در شکل ۲۰-۱۰ آمده است.

ریافتی مشابه برای هر یک از اشیای تعریف شده برای خانه امن استفاده می شود. بعد از اینکه صفات خاصه و عملیات هریک از اشیاء تا این نقطه تعریف شد، کارهای اولیه مدل OOA آغاز می شود. بحث دقیق تری از تحلیل و مدل تحلیلی که به عنوان بخشی از OOA ایجاد شده در فصل ۲۱ آمده است.

1. program



## ۲۰-۴ مدیریت پروژه های نرم افزاری شی گرا

همان گونه که در بخش های یک و دو این کتاب بحث نمودیم، مدیریت پروژه نرم افزاری به صورت مدرن را می توان به فعالیت های زیر تقسیم بندی نمود:

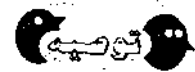
- ۱- ایجاد یک چارچوب پردازشی مشترک برای پروژه
- ۲- استفاده از متریک های تاریخی و معین برای بسط تلاش ها و تخمین زمان
- ۳- ایجاد محک های قابل ارائه ای که پیشروی را قابل ارزیابی می سازند.
- ۴- تعریف نقاط بازرسی برای سازمان دهی خطرات، تضمین کیفی و کنترل
- ۵- مدیریت تغییراتی که به طور یکنواخت با پیشروی پروژه رخ می دهند.
- ۶- پی گیری مشاهده و کنترل پیشرفت.

مدیر فنی که با پروژه شی گرا مواجه است این شش کار را در آن به کار می بندد. اما به خاطر ماهیت منحصر به فرد نرم افزار شی گرا، هر یک از این کارهای مدیریتی دارای احساس متفاوتی بوده و باید با استفاده از یک مجموعه فکری متفاوت اجرا گردد.

در بخش های بعدی، مدیریت پروژه نرم افزاری را از نظر پروژه های شی گرا بررسی می کنیم. اصول اصلی و بنیادی مدیریتی یکسان می باشند ولی تکنیک باید طوری انتخاب شود که پروژه OO به درستی سازمان دهی می شود.

## ۲۰-۴-۱ چارچوب فرآیند مشترک شی گرا

یک چارچوب پردازشی معمول یا مشترک (CPF)، رهیافت سازمان را در مورد طراحی نرم افزار بیان می دارد. این چارچوب، معیاری را شناسایی می کند که برای ساخت و نگهداری نرم افزار به کار رفته و همچنین کارها، معیارها و موارد قابل ارائه ای را بررسی می کند که لازم خواهند بود. این شیوه، درجه بندی از لحاظ شدت و بسختی به وجود می آورد که با آن انواع مختلف پروژه ها حاصل می گردند. CPF همیشه سازگار و انعطاف پذیر است. بنابراین نیازهای انفرادی یک تیم پروژه را برآورده می سازد. این مهم ترین مشخصه آن است.



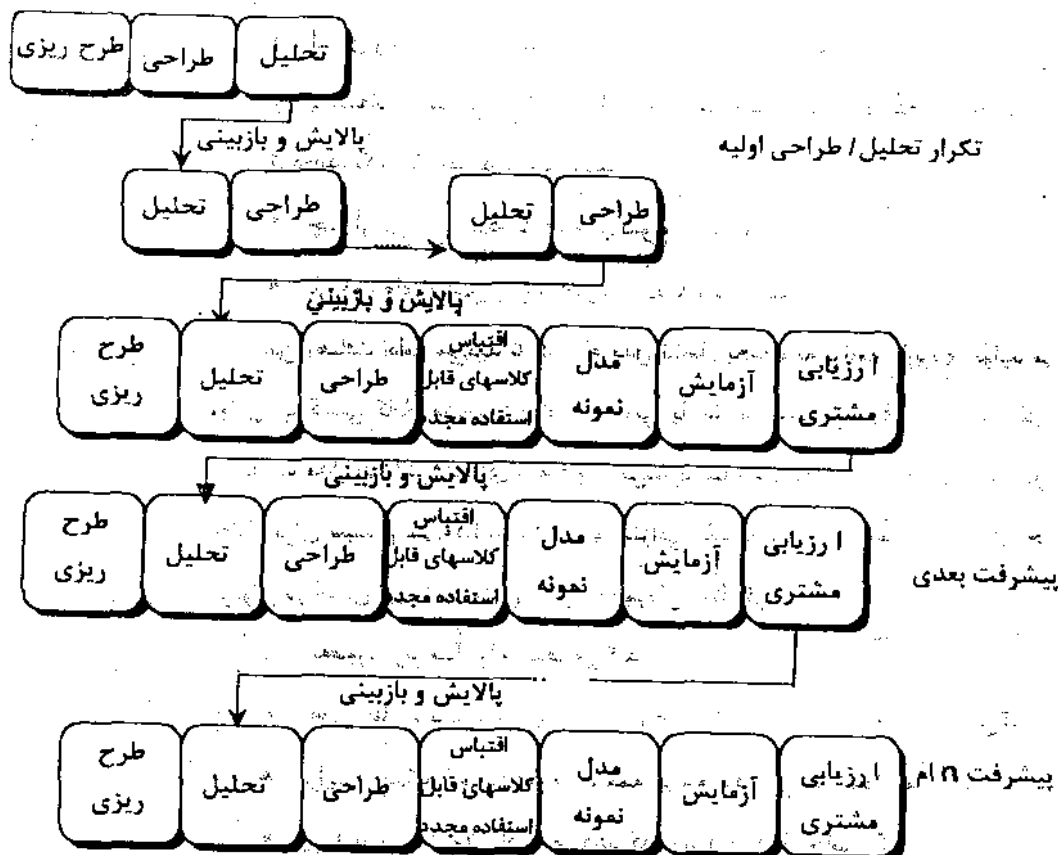
پروژه های OO نیز مانند پروژه های متعارف نرم افزاری به طرح ریزی، مدیریت و توجه نیاز دارند. بنابراین که OO مسئولیت شما را در این زمینه به نوعی کاهش خواهد داد



**ارجاع به وب**  
یک آموزش جامع در خصوص مدیریت پروژه های OO و مجموعه نشانگرها، در آدرس زیر وجود دارد:  
[www.mini.net/cetus/oo-project-](http://www.mini.net/cetus/oo-project-)



یک چارچوب فرآیند مشترک که فعالتهای پایه مهندسی نرم افزار را تعریف می کند، در فصل ۲ توضیح داده شده است.



شکل ۲۰-۱: رشته مرتبی از فرآیندهای نوعی یک پروژه شیء گرا

اد برارد و گاردی بوج [BER93]<sup>۱</sup> و [BOO91]<sup>۲</sup> از میان بقیه افراد پیشنهاد استفاده از مدل

موازی / Recursive را برای تولید نرم افزار شیء گرا پیشنهاد دادند. در اصل این مدل به شکل زیر عمل

می کند:

- تحلیل کافی انجام دهید تا کلاسها و ارتباطات عمده مسئله را جدا کنید.
- کمی کار طراحی انجام دهید تا تعیین کنید آیا کلاس بندیها و ارتباطات را می توان به شیوه ای عملی، پیاده سازی کرد یا خیر؟
- اشیای قابل استفاده مجدد را از کتابخانه ای استخراج کنید تا یک نمونه ساده بسازید.
- چند آزمون انجام دهید تا خطاهای نمونه را مشخص کنید.
- باز خورد و واکنش مشتریان را در مورد نمونه به دست آورید.
- مدل تحلیل را بر اساس آنچه که از روی الگو، طراحی و واکنش مشتری دریافته اید، تغییر داده و اصلاح کنید.
- طرح را از نو مورد تجدیدنظر قرار دهید تا با تغییرات شما تطابق یابد.

1. Berade, E. V.

2. Booch, G.

• اشیای ویژه را برنامه نویسی کنید (اشیایی که از کتابخانه بدست نمی‌آیند)

• با استفاده از اشیایی که از کتابخانه بدست می‌آیند و اشیای جدیدی که ایجاد

کرده‌اید، یک مدل جدید درست کنید.

• چند آزمون انجام داده تا خطاهای نمونه را مشخص کنید.

• باز خوردهای مشتریان را در مورد نمونه بدست آورید.

این رهیافت ادامه می‌یابد تا وقتی که مدل تکمیل یافته و به برنامه کاربردی تولید می‌رسد. این مدل

مولزی/ بازگشتی کاملاً مشابه یا پارادایم تکاملی یا حلزونی است. پیشرفت به صورت تکراری رخ می‌دهد.

آنچه که مدل بازگشتی/ مولزی را متفاوت می‌سازد عبارتست از: (۱) تشخیص این که مدل سازی طراحی و

تحلیل برای سیستم شیءگرا در یک سطح انتزاعی یکنواخت حاصل نمی‌گردد و (۲) تحلیل و طراحی را

می‌توان در اجزای سیستم مستقل به صورت همزمان به کار گرفت.

برارد به صورت زیر مدل را توصیف می‌کند:

• به طور نظام مند مسئله را به صورت اجزای کاملاً مستقل تجزیه و تفکیک می‌کند.

• فرایند تفکیک را مجدداً در مورد همه جزءها به کار می‌برد. (بخش مولزی)

• این فرایند را تا زمانی ادامه می‌دهد که معیارهای کامل حاصل گردند.

نکته مهم مورد توجه این است که فرایند تجزیه ذکر شده فوق در صورتی متوقف می‌گردد که

تحلیل گرا/ طراح متوجه شود که جزء یا زیر جزء مورد نیاز در یک کتابخانه قابل استفاده مجدد موجود

است.

به منظور کنترل چارچوب فرایند مولزی/ بازگشتی، مدیر پروژه باید بفهمد که پیشرفت به صورت

فرایند طراحی و ارزیابی شده یا خیر؟ یعنی کارهای پروژه و جدول برنامه ریزی آن شدیداً با اجزای مستقل

مرتبطند و میزان پیشرفت در مورد هر یک از این جزءها و اجرا به طور مستقل ارزیابی می‌شود.

هر تکرار فرایند مولزی/ بازگشتی نیازمند برنامه ریزی، کارهای مهندسی (مثل تحلیل، طراحی بدست

آوردن کلاس، الگوسازی و آزمون) و کارهای ارزیابی (شکل ۲۰-۱۱) است. در طول برنامه ریزی، کارهای

مربوط به هر یک از اجزای مستقل برنامه برنامه ریزی و زمان بندی می‌شوند. [نکته: زمان بندی با هر تکرار

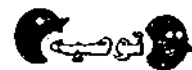
تنظیم می‌گردد تا با تغییرات مربوط به تکرار قبلی تطابق یابد]. در طول مراحل اولیه مهندسی، تحلیل و

طراحی به طور مکرر رخ می‌دهند. هدف جداسازی عناصر مهم مدل های طراحی و تحلیل شیءگراست. با

ادامه کارهای مهندسی، نسخه های تکمیلی از نرم افزار تولید می‌شوند. در طول ارزیابی، بازنگری ها، ارزیابی

مشتری و آزمون هایی برای هر مورد افزایش انجام می‌شوند که بازخورد آن بر فعالیت برنامه ریزی بعدی و

افزایش متعاقب آن اثر می‌گذارد.



معماری یک سیستم

OO از طرق مختلف

کار مولزی را تسهیل

می‌سازد. با این وجود،

اطمینان حاصل کنید

که هر وظیفه مولزی،

تعریف شده است. و از

این طریق پیشرفت

قابل ارزیابی خواهد

بود.

## ۲۰-۴-۲۰ متریک ها و برآورد پروژه شیءگرا

فنون قراردادی ارزیابی پروژه نرم افزاری نیازمند ارزیابی هایی از خطوط کد برنامه (LOC) یا امتیاز کارکردی (FP) به عنوان محرک اصلی کار ارزیابی است. از آن جا که یکی از اهداف موردنظر در مورد پروژه های شیءگرا استفاده مجدد است، تخمین های LOC حس و مفهوم کمتری دارند. تخمین های FP را می توان به طور مؤثری به کار گرفت. زیرا دامنه اطلاعات که در حال مورد نیازند از روی وضعیت مسئله قابل حصول است. تحلیل FP مقدار

و ارزشی برای تخمین پروژه های شیءگرا مهیا می کند، اما ارزیابی FP گرانگیزه کافی برای زمان بندی و تطبیقات لازمی که در طول الگوی موازی/ بازگشتی تکراری می کنیم، مهیا نمی سازد. لورنز و کید مجموعه متریک های زیر را در مورد پروژه بیان می دارند:

تعداد سناریوهای خلاصه. یک خلاصه سناریو<sup>۱</sup> (قابل مقایسه با توضیح مورد کاربرد در فصل ۱۱) یک توانایی دقیق از مراجعی است که روابط متقابل بین کاربر و برنامه کاربردی را نشان می دهد. هر نوشته از نظر شکلی به سه قسمت تقسیم می شود:

{initiator, action, participant} (آغازگر، عمل، شرکت کننده)

که در آن آغازگر شینی است که نیازمند خدماتی است (که پیام را آغاز می کند)، عمل<sup>۲</sup> نتیجه کار تقاضاست و شرکت کننده، شیء خادم است که نیاز را مرتفع می سازد. تعداد این خلاصه ها مستقیماً با اندازه برنامه و تعداد موارد آزمونی که باید برای آزمون سیستم در هنگام اجرا ارائه شوند، مرتبط است.

تعداد کلاس های کلیدی. کلاس های اصلی و کلیدی<sup>۳</sup> جزءهای بسیار مستقلی هستند که پیش تر در OOA تعریف شده اند. از آن جا که این کلاس در مرکز توجه دامنه مسئله قرار می گیرند، تعداد چنین کلاس هایی نشان گر میزان تلاش لازم برای توسعه نرم افزار بوده و هم چنین نشان گر میزان بالقوه کاربرد مجددی است که در طول تولید سیستم به کار گرفته می شود. [LOR94].<sup>۴</sup>

تعداد کلاس های پشتیبان. کلاس های پشتیبان<sup>۵</sup> برای اجرای سیستم لازمند، اما دقیقاً با حوزه مسئله مرتبط نمی باشند. به عنوان مثال می توان به کلاس های GUI، کلاس های تغییر و دسترسی به پایگاه داده ای و کلاس های محاسبه اشاره کرد. علاوه بر این، می توان کلاس های پشتیبان را برای هر یک از کلاس های اصلی درست نمود. کلاس های پشتیبان به صورت تکرار از سراسر فرایند موازی/ بازگشتی تعریف شده اند.

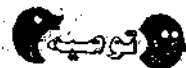
۱. در فصل ۲۴، متریک های فنی مربوط به سیستم های شیءگرا به تفصیل ارائه شده اند.

2. scenario script

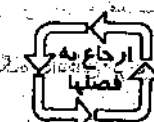
3. action

4. Key Classes

5. Lorenz, M. and J.



این متریک های می توانند بعنوان مکمل به متریک های FP (امتیاز کارکردی) الحاق شوند. آنها راهی را برای تعیین "سایز" یک پروژه OO فراهم می سازند.



یک توضیح تفصیلی از متریک های OO در فصل ۲۴ ارائه شده است.

تعداد کلاس های پشتیبان، نشان گر میزان تلاش لازم برای تولید نرم افزار و هم چنین میزان بالقوه استفاده مجددی است که در طول تولید سیستم به کار گرفته می شود.

میانگین تعداد کلاس های پشتیبان در هر کلاس کلیدی، به طور کلی، کلاس های اصلی پیش تر در پروژه شناسایی می شوند. کلاس های پشتیبان از هر جهت تعریف شده اند. اگر میانگین تعداد کلاس های پشتیبان در هر کلاس برای یک دامنه فرضی مسئله شناخته شونده انجام تخمین (بر اساس تعداد کل کلاس ها) بسیار ساده می شود.

لورنز و کید بیان داشتند که برنامه هایی با GUI بین ۲ تا ۳ برابر تعداد کلاس های پشتیبان، دارای کلاس اصلی هستند. برنامه های غیر GUI یک تا دو برابر تعداد کلاس پشتیبان، کلاس اصلی دارند.

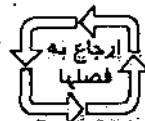
تعداد سیستم های فرعی، سیستم فرعی<sup>۲</sup> جمعیتی از کلاس هایی است که یک کارکرد را پشتیبانی می کنند که برای کاربر نهایی یک سیستم مشهود است. وقتی سیستم های فرعی شناسایی شدند، راحت تر می توان طرح بندی یک برنامه زمان بندی منطقی را طراحی نمود که در آن کار روی سیستم های فرعی، در میان کارکنان پروژه تقسیم می شود.

#### ۲۰-۴-۳ یک رهیافت زمان بندی و برآورد شیءگرا

کار تخمین در پروژه نرم افزاری بیشتر هنر است تا علم. در هر صورت این کار به هیچ وجه مانع استفاده از رهیافتی نظام مند نمی شود. به منظور ارائه تخمین های دقیق، لازم است نقاط چندگانه داده ای را ایجاد نمود. یعنی، تخمین ها باید با استفاده از فنون مختلفی حاصل گردند. تخمین میزان کار و مدت استفاده شده برای توسعه متعارف نرم افزار، در اشیاء شیءگرای قابل اجرا است، اما پایگاه داده ای برای پروژه های شیءگرا برای بسیاری از سازمان ها نسبتاً کوچک است. بنابراین، برآورد هزینه نرم افزار قراردادی مکمل با روشی ارزشمند است که منحصر برای نرم افزار شیءگرا طراحی شده است.

لورنز و کید روش زیر را پیشنهاد می کنند:

- ۱- تخمین ها را با استفاده از تجزیه کارها، تحلیل FP و هر روش دیگری که در مورد برنامه های کاربردی متعارف قابل اجراست، بدست آورید.
- ۲- با استفاده از OOA (فصل ۲۱) خلاصه سناریوهایی نوشته و یک شماره را تعیین کنید. متوجه باشید که تعداد سناریوها ممکن است با جلو رفتن برنامه تغییر کنند.
- ۳- با استفاده از OOA تعداد کلاس های اصلی (کلید) را تعیین کنید.
- ۴- نوع رابط را برای برنامه، طبقه بندی نموده و یک ضریب برای کلاس های پشتیبان ایجاد کنید.



تعدادی از فنون برآورد پروژه های نرم افزاری به تفصیل در فصل ۵ ملاحظه می شود.

1. Support Classes

2. subsystem

interface type	multiplier	
NO GUI	2.0	(بدون رابط گرافیکی)
Text-based user interface	2.25	(رابط گرافیکی متنی)
GUI	2.5	(رابط گرافیکی ساده)
Complex GUI	3.0	(رابط گرافیکی پیچیده)

تعداد کلاس‌های اصلی (مرحله ۳) را در ضرب، ضرب کنید تا مقدار تخمینی برای تعداد

کلاس‌های پشتیبان بدست آورید.

۵- تعداد کل کلاس‌ها (اصلی + فرعی) را در میانگین تعداد واحدهای کاری در هر کلاس

ضرب کنید. لورنز و کید ۱۵ تا ۲۰ نفر - روز در هر کلاس را بیان می‌دارند.

۶- تخمین مبتنی بر کلاس را با ضرب تعداد میانگین واحدهای کاری در سناریوها، چک

کنید.

زمان‌بندی پروژه‌های شیءگرا به‌خاطر ماهیت تکرار شونده چارچوب فرایند، پیچیده می‌شود.

لورنز و کید مجموعه‌ای از معیارها را بیان می‌دارند که ممکن است در طول زمان‌بندی پروژه کمک

کند.

تعداد تکرارهای عمده، مدل حلزونی را به‌خاطر بیاورید (فصل ۲)، تکرار عمده مربوط به یک

برگشت ۲۶۰ درجه‌ای پیچشی است. مدل پردازش موازی/ بازگشتی چند تارپیچ کوچک به‌وجود می‌آورد.

(تکرارهای تعیین محل شده) که با پیش‌روی تکرار عمده به‌وجود می‌آیند. لورنز و کید بیان می‌دارند که

تکرارهایی که از نظر طولی در فاصله ۲/۵ تا ۴ ماه هستند از نظر پی‌گیری و سازمان‌دهی از آسان‌ترین نوع

هستند.

تعداد قراردادهای تکمیل شده، یک قرارداد مجموعه‌ای از مسئولیت‌های عمومی مربوطه است

که توسط سیستم‌ها و کلاس‌های فرعی برای مشتریان مهیا شده‌اند. قرارداد یک سنگ محک عالی است و

حداقل یک قرارداد باید با هر تکرار پروژه مرتبط شود. مدیر پروژه می‌تواند از قراردادهای تکمیلی به‌عنوان

یک نشان‌گر خوب برای پیشرفت در مورد پروژه شیءگرا استفاده کند.

#### ۴-۴-۲۰ ردگیری پیشرفت یک پروژه شیءگرا

گرچه مدل پردازش موازی/ بازگشتی بهترین چارچوب برای یک پروژه شیءگراست، موازی‌گرایی،

ردیابی پروژه را مشکل می‌سازد. مدیر پروژه ممکن است در ایجاد معیارهایی با مفهوم برای پروژه شیءگرا



دچار مشکل شود، زیرا در یک لحظه چند چیز مختلف با هم رخ می‌دهند. به‌طور کلی، معیارهای عمده زیر را باید وقتی موارد مورد اشاره فوق تأمین شدند، تکمیل شده پنداشت:

#### نقطه عطف فنی: تحلیل شیء گرا تکمیل شده<sup>۱</sup>

- تمام کلاس‌ها و سلسله مراتب کلاس تعریف شده و بازنگری شده‌اند.
- عملیات و صفات خاصه کلاس در رابطه با هر کلاس تعریف و بازنگری شده‌اند.
- روابط کلاس (فصل ۲۱) ایجاد و بازنگری شده‌اند.
- مدل رفتاری ایجاد و بازنگری شده‌اند.
- کلاس‌های قبل استفاده مجدد مورد توجه قرار گرفته‌اند.

#### نقطه عطف فنی: طراحی شیء گرا تکمیل شده<sup>۲</sup>

- مجموعه سیستم‌های فرعی تعیین و بازنگری شده‌اند.
- کلاس‌ها به سیستم‌های فرعی تخصیص یافته و بازنگری شده‌اند.
- تخصیص کار انجام و بازنگری شده‌است.
- مسئولیت‌ها و همکاری‌ها شناسایی شده‌اند.
- صفات خاصه و عملیات طراحی و بازنگری شده‌اند.
- مدل پیغام‌ری ایجاد و بازنگری شده‌است.

#### نقطه عطف فنی: برنامه‌نویسی شیء گرا تکمیل شده<sup>۳</sup>

- هر کلاس جدید، در برنامه‌ای که از مدل طراحی گرفته شده، پیاده‌سازی گردیده است.
- کلاس‌بندی‌های استخراجی (از کتابخانه قابل استفاده مجدد) پیاده‌سازی شده‌اند.
- مدل نمونه یا افزایش ساخته می‌شود.

#### نقطه عطف فنی: آزمون شیء گرا<sup>۴</sup>

- درستی و کامل بودن تحلیل شیء گرا و مدل‌های طراحی بازنگری شده‌اند.
- شبکه‌ای از کلاس - مسئولیت - همکاری (فصل ۲۳) ارائه و بازنگری شده است.
- مولود آزمون طراحی و آزمون‌هایی در سطح کلاس برای هر یک اجرا می‌شود.
- مولود آزمون طراحی شده و آزمون گروهی تکمیل می‌گردد و در آخر کلاس‌ها با هم

ترکیب می‌شوند.

- آزمون‌های سطح سیستم تکمیل می‌گردند.

1. Technical milestone: OO analysis completed

2. Technical milestone: OO design completed

3. Technical milestone: OO programming completed

4. Technical milestone: OO testing

با یادآوری مدل پردازش موازی/ بازگشتی که پیش‌تر در این فصل مورد بحث قرار گرفت نکته مهم قابل توجه این است که هر یک از این نقاط عطف را ممکن است با وقوع افزایش‌های متعدد که در اختیار مشتری قرار می‌گیرند، مجدداً ببینیم.

#### ۴-۵ خلاصه

فن‌آوری‌های شیءگرا، نمایان‌گر دیدگاهی طبیعی از جهان است. اشیاء به کلاس‌ها و سلسله مراتب کلاس طبقه‌بندی می‌شوند. هر کلاس شامل مجموعه‌ای از صفات خاصه است که آن را توصیف کرده و مجموعه‌ای از عملیات که رفتار آن را تعریف می‌نمایند. اشیاء تقریباً هرگونه جنبه قابل شناسایی حوزه مربوط به مسئله را مدل‌سازی می‌کنند. موجودیت‌های خارجی، اشیاء، رویدادها، نقش‌ها، واحدهای سازمانی، مکان‌ها و ساختارها همگی را می‌توان به‌عنوان شیء نشان داد. نکته‌ی مهم دیگر این است که اشیاء (و کلاس‌هایی که این اشیاء از آنها مشتق می‌گردند) در برگزیده داده‌ها و فرآیند است. عملیات پردازشی بخشی از شیء بوده و با رسیدن پیامی به شیء آغاز می‌گردند. تعریف کلاس، زمانی که تعریف گردید، پایه و اساس قابلیت استفاده مجدد را در مدل‌سازی، طراحی و پیاده‌سازی تشکیل می‌دهد. اشیاء جدید را می‌توان از یک کلاس به‌دست آورد. سه مفهوم مهم باعث تفاوت رهیافت شیءگرا از مهندسی نرم‌افزار متعارف می‌شود: بسته‌بندی داده‌ها و عملیاتی که داده‌ها را در یک شیء نام‌گذاری شده، تغییر می‌دهند، وراثت باعث می‌شود، صفات خاصه و عملیات یک کلاس به تمام کلاس‌های فرعی رسیده و به اشیایی که از آنها برگرفته شده‌اند تخصیص یابند، چندریختی باعث می‌شود که چند عملیات مختلف دارای نامی یکسان بوده که این کار تعداد خطوط کد برنامه لازم برای اجرای سیستم را کاهش داده و تغییرات را در هنگام پیاده‌سازی آنها تسهیل می‌کند.

سیستم‌ها و محصولات شیءگرا با استفاده از مدل تکاملی که گاهی به آن مدل موازی/ بازگشتی می‌گویند مهندسی طراحی می‌شوند. نرم‌افزار شیءگرا به‌صورت تکراری تکامل یافته و باید با توجه به این که محصول نهایی در طول یک‌سری فعالیت‌های افزایشی با تکاملی تولید می‌گردد، نیازمان‌دهی شود.

### مسایل و نکاتی برای تفکر و تعمق بیشتر

- ۱-۲۰ مهندسی نرم افزار شیء گرا به سرعت رهیافتهای توسعه سنتی نرم افزار را دگرگون می سازد. با این وجود شیء گرایی نیز چون هر فناوری دیگری خالی از اشکال نیست. با استفاده از منابع اینترنتی و مکتوب کتابخانه خود مقاله کوتاهی تهیه کنید که شیء گرایی را نقد کرده، به چالش بکشد. و بحث کنید که چرا منتقدین دقت در کاربرد پارادایم شیء گرایی را الزامی می دانند.
- ۲-۲۰ در این فصل، به وضعیتی که یک شیء جدید نیازمند صفت خاصه یا عملیات غیر موجود در کلاس خود باشد اشاره نکردیم. فکر می کنید چگونه راه حلی می توان ارائه نمود؟
- ۳-۲۰ پژوهشی انجام دهید و پاسخی واقعی برای مسئله ۲-۲۰ بیابید.
- ۴-۲۰ به زبان خودتان و چند مثال، اصطلاحات کلاس، بسته بندی (کپسوله کردن)، توارث و چندریختی را بیان کنید.
- ۵-۲۰ اشیایی را که ممکن است در یک سیستم رزرو بلیط هواپیما ظاهر شوند، از چه ویژگیها برخوردارند. چه صفات خاصه ای خواهند داشت؟
- ۶-۲۰ اشیای تعریف شده برای سیستم خانه امن را مرور کنید. آیا اشیای دیگری نیز وجود دارند که گمان دارید در آغاز مدل سازی باید تعریف شوند؟
- ۷-۲۰ یک واسط کاربر گرافیکی معمولی را در نظر بگیرید مجموعه ای از کلاس ها و (زیر کلاس ها) را برای موجودیتهای رابط که معمولاً در رابط گرافیکی کاربر ظاهر می شوند را تعریف کنید.
- ۸-۲۰ مثالی برای یک شیء مرکب بیاورید.
- ۹-۲۰ کار مهندسی یک واژه پرداز نرم افزاری جدید به شما واگذار شده است. کلاسی با نام document تعریف می شود. صفات خاصه و عملیاتی را که به document مربوط می شوند، تعریف کنید.
- ۱۰-۲۰ دو زبان مختلف برنامه نویسی شیء گرا را مورد پژوهش قرار داده و نشان دهید پیامها چگونه در ترکیب زبان پیاده سازی می شوند. چند مثال برای هر زبان ارائه کنید.
- ۱۱-۲۰ مثالی واضح از تجدید ساختار سلسله مراتب کلاس ها بیاورید. آنگونه که در شکل ۸-۲۰ توضیح داده شد.
- ۱۲-۲۰ مثال واضحی از وراثت چندگانه ارائه دهید. یک یا چند مقاله را در این موضوع تحقیق کرده نظرات موافق و مخالف وراثت چندگانه را بیان کنید.
- ۱۳-۲۰ برای سیستمی که استادان معین می دارد یک توضیح دامنه (جمله دامنه) بسازید. با استفاده از تجزیه گرامری کلاس های کاندید، صفات خاصه و عملیات را برای سیستم تعیین کنید. با استفاده از معیارهای توضیح داده شده در بخش ۲-۳-۱، تعیین کنید که آیا ضرورتی دارد که کلاس در مدل تحلیل مورد استفاده واقع شود.

۲۰-۱۴ به زبان خودتان شرح دهید که چرا مدل فرآیند بازگشتی/ موازی برای سیستم‌های شیء

گرا مناسب می باشد.

۲۰-۱۵ سه یا چهار مثال مشخص از کلاس کلیدی و کلاس پشتیبان که در بخش ۲۰-۴ بحث

شده تهیه کنید.