بسمه‌تعالی

فصل چهارم

روش‌های شناسائی و یافتن کلاس‌ها

اهداف جلسه

- درک اهمیت طبقه‌بندی در OOAD
- آشنائی با روش‌های یافتن کلاس‌ها
- درک تفاوت بین روش‌های مبتنی بر داده و مبتنی بر وظیفه
- آشنائی با مدلسازی بوسیله کارت‌های CRC
- آشنائی با لایه‌بندی و نقش آن
- آشنائی با مقوله‌بندی و نقش آن
- کارت‌های CRC و مقوله‌بندی

فهرست مطالب

- طبقه‌بندی (*Classification*)
- روش‌های مبتنی بر داده (*Data-Driven*)
- روش‌های مبتنی بر وظیفه (*Responsibility-Driven*)
- روش‌های شناسائی کلاس‌های اولیه
- روش CRC
- لایه‌بندی (*Layering*)
- مقوله‌بندی (*Stereotyping*)

طبقه‌بندی (*Classification*)

- طبقه‌بندی ابزاری است که بوسیله آن دانش خود را مرتب می‌نماییم
- مسئله طبقه‌بندی در همه رشته‌های علوم محض، کاربردی و مهندسی مطرح است

کلاس‌های مطرح برای مدلسازی یک سیستم

بوسیله طبقه‌بندی

۱) کلاس‌های مناسب یا کلیدی
۲) غیر مناسب یا کم اهمیت

طبقه‌بندی – ویژگیها

۱) هر طبقه‌بندی با توجه به معیاری(معیارهایی) انجام می‌گیرد

- معیارهای متفاوت منجر به طبقه‌بندی‌های مختلف خواهد شد

۲) طبقه‌بندی ایده‌آل (یعنی بهترین طبقه‌بندی بدون توجه به شرایط موجود) وجود ندارد!

۳) فرآیند طبقه‌بندی یک فرآیند افزایشی و تکراری

ویژگیهای طبقه‌بندی در OOAD

۱) کلاس‌هایی باید انتخاب شوند که با توجه به معیارها و محدودیت‌های موجود (اقتصادی، تکنولوژی و ...) سازگار باشند

۲) نباید دنبال راه‌حل طلائی بلکه باید بدنبال راه‌حل مناسبتر باشیم!

ویژگیهای طبقه‌بندی در OOAD (ادامه)

۳) طبیعت افزایشی و تکراری فرآیند طبقه‌بندی به صورت زیر خود را نشان می دهد:

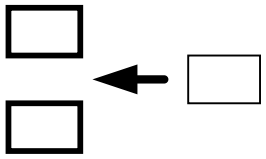
تعدادی کلاس کاندیدا (اولیه)

افزایش درک مسئله + تلاش برای تشریح ساختار و رفتار کلاس‌ها

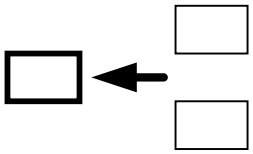
شناسائی کلاس‌های دیگر

بهبود کلاس‌های اولیه

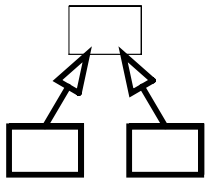
تقسیم یک کلاس بزرگتر به تعدادی کلاس کوچکتر



ترکیب تعدادی کلاس کوچکتر برای تشکیل دادن یک کلاس بزرگتر

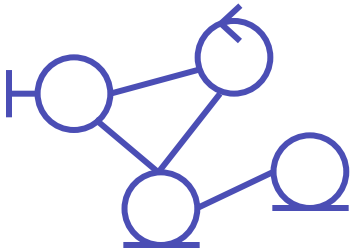


ایجاد زیرکلاس‌های جدید

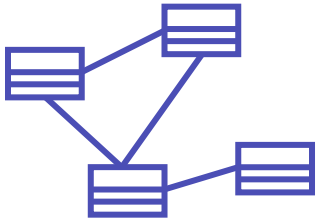


منابع تشخیص کلاس‌ها

- دو منبع اصلی برای تشخیص کلاس‌ها
  - فضای مسئله (*Problem Space*)
  - فضای راه‌حل (*Solution Space*)



مدل تحلیل ← توصیف فضای مسئله



مدل طراحی مفصل ← توصیف فضای راه‌حل

فرآیند شناسایی کلاس‌ها

فرآیند شناسائی کلاس‌ها شامل دو فعالیت

اکتشاف (*Discovery*) و ابداع (*Invention*) است

اکتشاف کلاس‌های موجود

ابداع کلاس‌هایی برای پیاده‌سازی کلاس‌های موجود

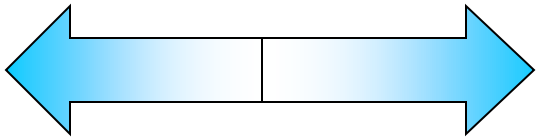
مثال: ساختمان داده‌ها (لیست، آرایه، درخت،...)، ارتباط با پایگاه داده‌ها، کلاس‌های کنترلی و هماهنگ کننده، ...

یافتن کلاس‌ها

- رهیافت‌ها

۱) روش‌های مبتنی بر داده (*Data Driven Approach*)

۲) روش‌های مبتنی بر رفتار (*Responsibility Driven Approach*)



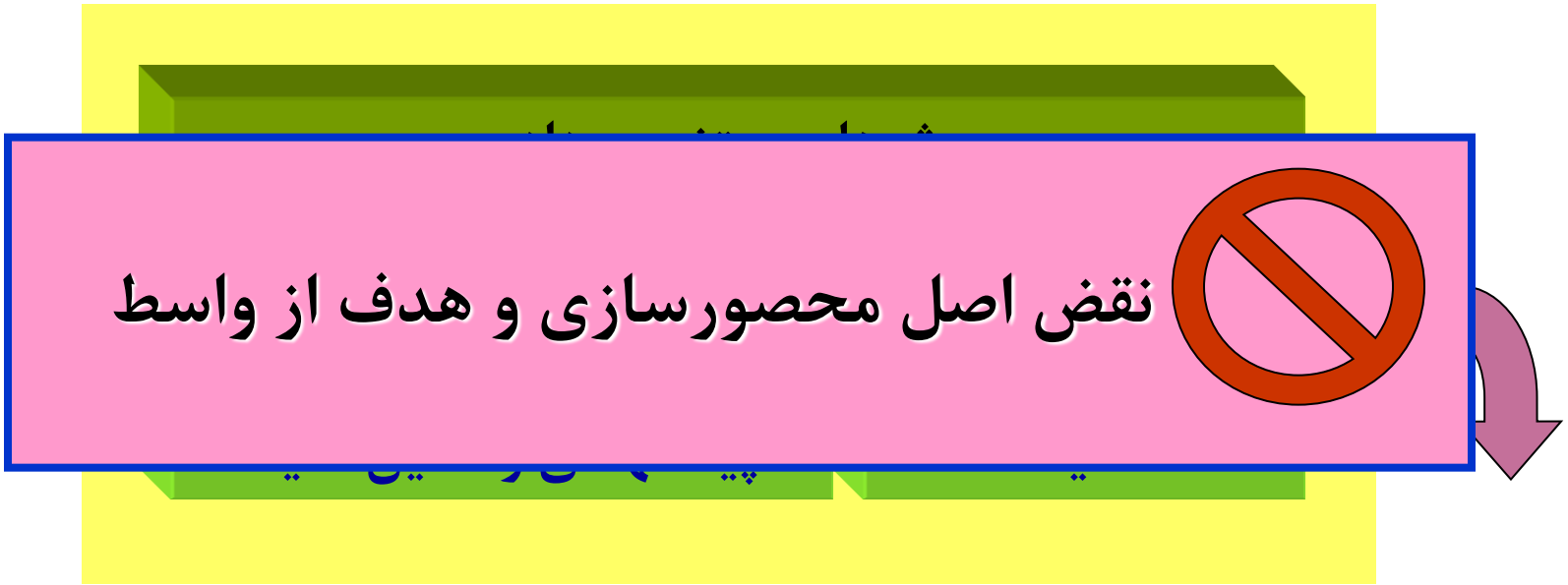
Responsibilities — ***Data***

اولویت با وظایف — اولویت با داده‌ها

روش‌های مبتنی بر داده

- در این روش‌ها مبنای شناسائی کلاس‌های مناسب سیستم، شناسائی ساختار داده‌های مورد نیاز هر کلاس است
- فرآیند تعیین کلاس‌ها با پرسیدن دو سوال صورت می‌گیرد
  ۱- ساختار هر کلاس چیست؟
  ۲- چه عملیاتی بوسیله هر کلاس انجام می گیرد؟
- مزیت
  - سهولت یادگیری مخصوصاً برای کسانی که پیش زمینه‌ای در روش‌های ساخت‌یافته و طراحی مبتنی بر داده‌ها را دارند

روش‌های مبتنی بر داده (ادامه)



سرویس‌های کلاس به ساختار داخلی آن وابسته خواهند بود

سرویس‌گیرندگان به ساختار داخلی کلاس وابسته خواهند شد

روش‌های مبتنی بر وظیفه

- در این روش‌ها مبنای شناسائی کلاس‌های مناسب سیستم، شناسائی مسئولیت‌های (*Responsibilities*) مورد نیاز هر کلاس است
- فرآیند تعیین کلاس‌ها با پرسیدن دو سوال صورت می‌گیرد

۱- هر کلاس چه مسئولیتی دارد؟ (چه عملیاتی بوسیله این کلاس انجام می‌گیرد؟)

۲- این شی، چه اطلاعاتی با بقیه اشیاء به اشتراک می‌گذارد؟

روش‌های مبتنی بر وظیفه (ادامه)

سرویس‌های کلاس به ساختار داخلی آن وابسته نخواهند بود

سرویس گیرندگان، مستقل از ساختار داخلی کلاس خواهند شد

روش‌های مبتنی بر وظیفه (ادامه)

- نگاه اصلی این روش به هر کلاس عبارت از موجودیتی است که در هر آن می‌تواند نقش سرویس‌دهنده یا نقش سرویس گیرنده را ایفا نماید
- هر کلاس در نقش سرویس‌دهنده می‌تواند فراهم‌کننده خدمات برای ۳ نوع سرویس‌گیرنده باشد

۱- سرویس‌گیرندگان خارجی (*External Clients*)

۲- سرویس‌گیرندگان مشتق شده (*Derived Clients*)

۳- خود کلاس (*Self Client*)

روش‌های مبتنی بر وظیفه (ادامه)

- مزیت
  - بزرگترین مزیت این روش به حداکثر رساندن محصورسازی در سطح طراحی است که باعث افزایش قابلیت نگهداری و انعطاف‌پذیری سیستم نسبت به تغییرات آتی خواهد گردید

روش‌های شناسائی کلاس‌های اولیه

۱) طبقه‌بندی‌های پیشنهاد شده بوسیله متدولوژی‌های OO

۲) تحلیل دامنه (*Domain Analysis*)

۳) تحلیل موارد کاربری (*Use Case Analysis*)

۴) تحلیل لغوی صورت مساله (*Problem Statement Analysis*)

۵) استفاده از الگوها (*Patterns*)

۶) کارت‌های CRC

طبقه‌بندی‌های پیشنهاد شده...

- منابع بالقوه زیر برای شناسائی کلاس‌ها پیشنهاد می‌شوند
  ۱) دستگاه‌ها(*Devices*): دستگاه‌هایی که برنامه با آنها تعامل دارد
  ۲) نقش‌ها: نقش‌های گوناگون که کاربر در تعامل با سیستم ایفا می‌نماید
  ۳) محل‌های فیزیکی (مانند دفاتر، سایت‌ها،...) که برای سیستم مهم هستند
  ۴) سازمان‌ها: مجموعه‌های سازماندهی شده (مردم، منابع، ابزار، ...) که دارای ماموریت‌های مشخصند

طبقه‌بندی‌های پیشنهاد شده...

۵) مفاهیم منطقی: اصول و ایده‌های منطقی که در منطق کاری سازمان بکار گرفته می‌شوند

۶) ساختار: همان روابط IS-A و PART-OF

۷) دیگر سیستم‌ها: سیستم‌های خارجی که برنامه با آنها تعامل دارد

تحلیل دامنه (Domain Analysis)

- عبارتست از شناسائی کلاس‌ها و اشیاء مشترک در همه برنامه‌های کاربردی (*Applications*) متعلق به یک دامنه مشخص (مانند کامپایلرها، سیستم‌های اطلاعاتی جغرافیایی، ...)
- در این روش، با دیدن سیستم‌های مرتبط و اسناد آنها و با صحبت با کارشناسان خبره در زمینه سیستم مورد نظر می‌توان کلاس‌های کلیدی یک سیستم را حدس زد

تحلیل موارد کاربری

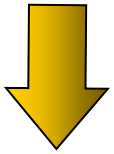
- دنباله‌ای از عملیات است که یک سیستم انجام می‌دهد تا یک نتیجه قابل مشاهده و ارزشمند برای کاربر فراهم نماید

تحلیل لغوی صورت مساله

- با تحلیل صورت مکتوب مساله می‌توان کلاس‌های اولیــه را بدست آورد

گام ۱: تعیین نام‌ها و فعل‌های موجود

گام ۲: حذف نام‌ها و فعل‌های غیرضروری

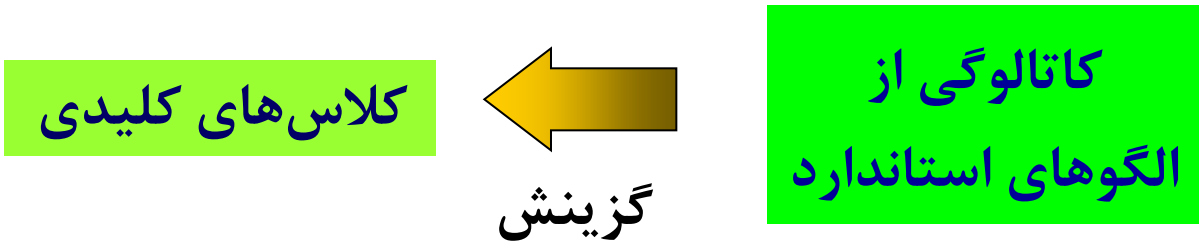


نام‌ها: کلاس‌های کاندیدا

فعل‌ها: عملیات کاندیدا

استفاده از الگوها

- یک الگو، یک مساله طراحی کـه در یـک زمینـه مشـخص مرتباً تکرار می‌گردد را توصیف کرده و سپس یـک راه‌حـل کلی و تکرارپذیر برای آن ارائه می‌کند



کارت‌های CRC

- روشی غیررسمی(*Informal*) برای شناسائی و توصیف کلاس‌ها، رفتار و مسئولیت‌های آنها و همکارانی (کلاس‌های دیگر) که به کمک آنها وظایف خود را انجام می‌دهند

کلمه	CRC
کلاس(رده)	Class
مسئولیت‌ها	Responsibilities
همکاران	Collaborators

کارت‌های CRC (ادامه)

Class Name	Collaborators
Responsibilities	

کارت‌های "۲×۳"

اسم کلاس :

مسئولیت‌ها :

همکاران :

دارا بودن مجموعه‌ای از اطلاعات لازم و کافی (به صورت *Abstract*) از هر عنصر در یک سیستم

- در چه رده‌ای ؟ (Class)
- با چه وظایفی؟ (Responsibilities)
- با چه همکارانی ؟ (Collaborators)

کارت‌های CRC (ادامه)

- هدف

- فراهم نمودن روشی برای آموزش مفاهیم شی‌گرا (کمک به پیاده نمودن شیوه‌ای که در آن اشیاء محور می‌باشند)
- حل مشکل شناسایی اشیاء برای مسائلی که آشنایی زیادی در مورد آنها نداریم

کارت‌های CRC – ویژگیها

- سادگی روش: بر اساس یک بازی ساده با کارت‌ها

- طبیعی بودن روند کار و نمایش سناریوهای واقعی

What if …

- فرایندگرایی براساس کار گروهی

فرآیند مدلسازی بوسیله کارت‌های CRC

۱- موارد کاربری کلیدی سیستم را مرور کنید

۲- در صورت نیاز یکی (یا ترکیبی) از تکنیک‌های یافتن کلاس‌های اولیه را برای شناسائی مجموعه‌ای از کلاس‌های کاندیدا بکار ببرید

۳- به ازای هر مورد کاربری گام‌های زیر را انجام دهید

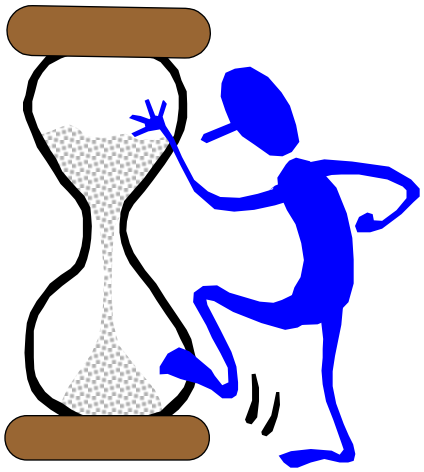
الف) از کلاس‌های موجود، کلاس‌هایی که مناسب این مورد کاربری است را مشخص نمایید

ب) اگر کلاس مناسبی وجود نداشته باشد پس کلاس جدیدی را ایجاد نمایید

فرآیند مدلسازی بوسیله ... (ادامه)

ج) مسئولیت‌های کلاس را تشخیص دهید:

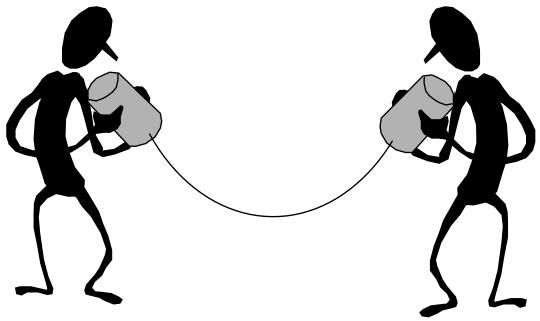
- این کلاس باید چه وظیفه‌ای را انجام دهد؟
- اگر وظیفه‌ای را در اختیار دارید، این وظیفه متعلق به کدام کلاس است؟
- بعضی از مسئولیت‌ها بوسیله همکاری کلاس با دیگر کلاس‌ها انجام می‌پذیرند، بنابراین عجله نکنید.



فرآیند مدلسازی بوسیله ... (ادامه)

د) همکاران کلاس را تشخیص دهید:

- سناریوی "What if...?" را اجرا کنید
- همکاری هنگامی رخ می‌دهد که یک کلاس نیازمند اطلاعاتی باشد که در اختیار ندارد
- همکاری هنگامی رخ می‌دهد که یک کلاس نیازمند به روزرسانی اطلاعاتی باشد که در اختیار ندارد
- در هر همکاری، حداقل یک کلاس آغازکننده باید وجود داشته باشد

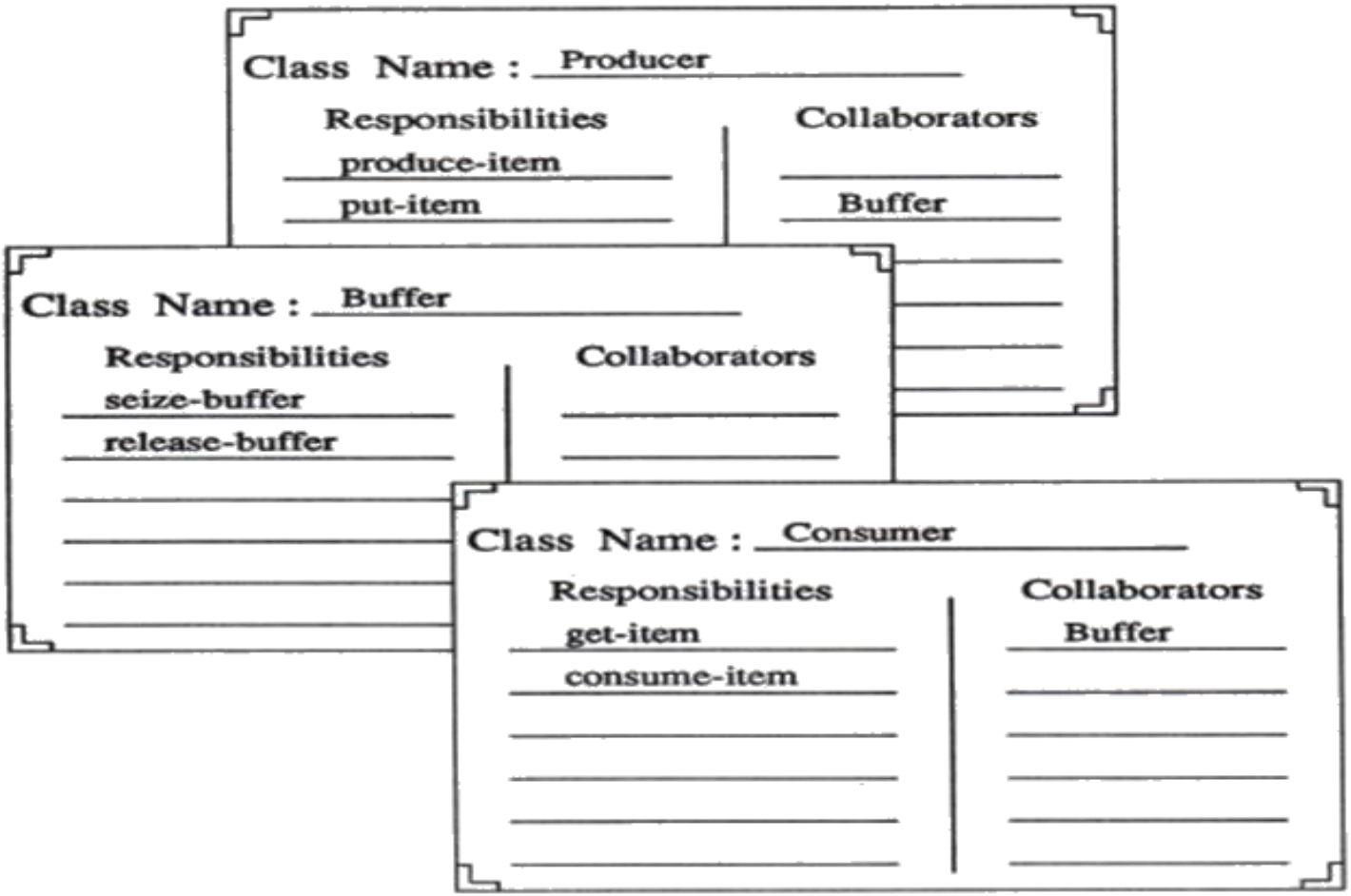


فرآیند مدلسازی بوسیله ... (ادامه)

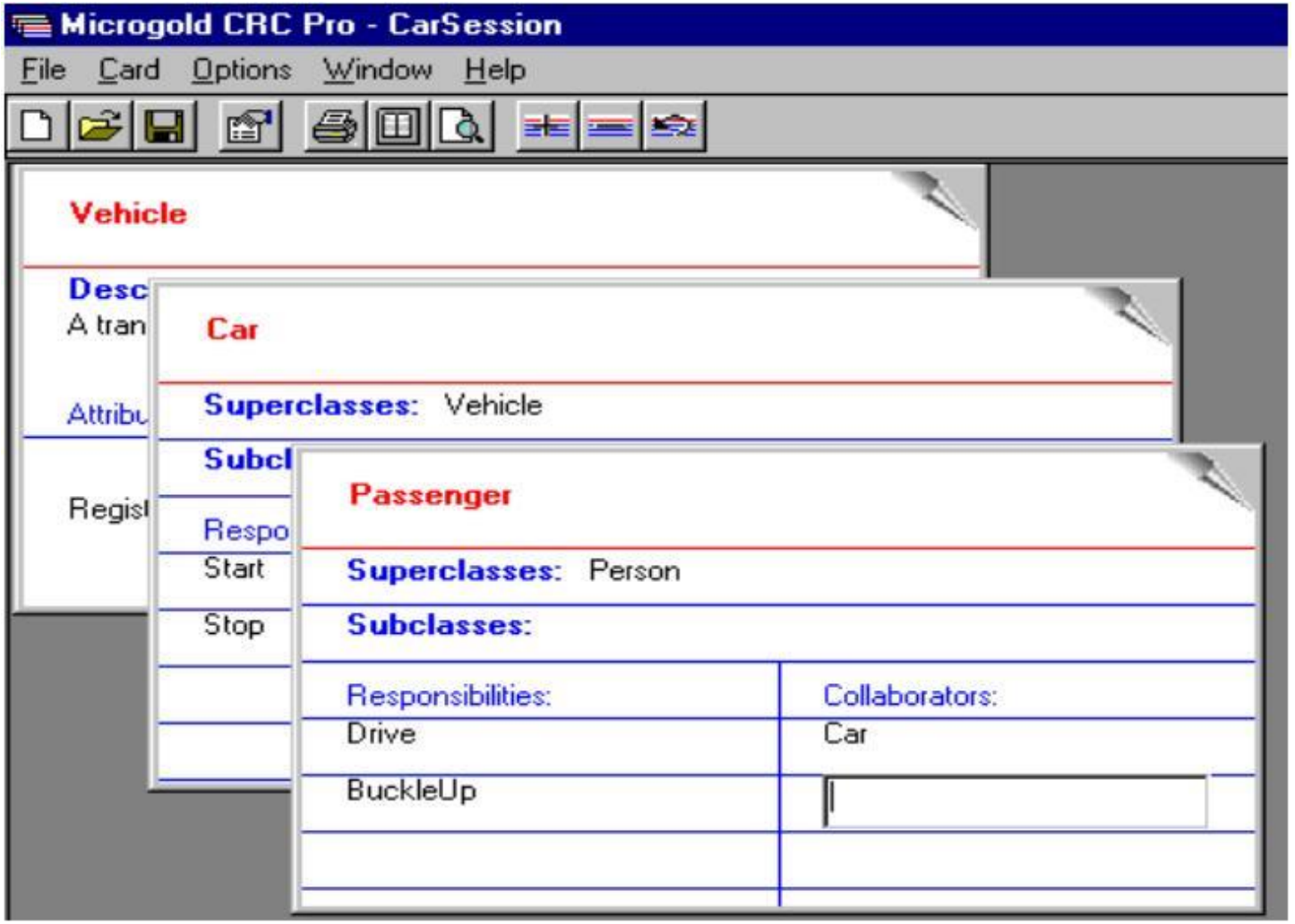
هـ) کارت‌های CRC را دور میز چرخش دهید:

- کارت‌های کلاس‌هایی که با یکدیگر همکاری دارند نزدیک هم قرار دهید
- هرچه همکاری قویتر باشد نزدیکی دو کلاس به یکدیگر می‌بایست بیشتر باشد
- کارت‌های پر(شلوغ) را در وسط میز قرار دهید
- کارت‌ها را دور میز بچرخانید
- از کسانی که در جلسه شرکت دارند بخواهید که به کارت‌های در حال چرخش توجه نمایند
- حاضرین در جلسه ارتباطات جدیدی بین کلاس‌ها تشخیص خواهند داد
- سناریوی «چه می‌شود اگر...؟» را اجرا نمایید

مثال: مسئله Producer/Consumer



نمونه نرم‌افزار کارت‌های CRC



کارت‌های CRC – مزایا

- نقطه مناسبی برای شروع تحلیل (جرقه ذهنی)
- قابلیت شبیه‌سازی رفتار سیستم (مرور سناریو)
- با نمودار کلاس‌ها سازگار است
- امکان کارگروهی

کارت‌های CRC – مزایا (ادامه)

- نداشتن مشکل سیم‌بندی (*Wire Syndrome*)
- مقوله‌بندی (*Stereotyping*)
- رضایت کاربران
- تحلیل بوسیله خبرگان

کارت‌های CRC – معایب

- مشکل برقراری ارتباط با کاربران
- کارت‌های CRC تنها بخشی از نیازمندی‌های یک سیستم شی‌گرا را تشکیل می‌دهند

لایه‌بندی و مقوله‌بندی

- یک سیستم پیچیده ابعاد گوناگونی دارد لذا باید از زوایای متفاوتی به آن نگریست
- مثال: ساختمان
  - یک ساختمان معمولاً از دید مشتری دارای یک بعد می‌باشد. اما معمار، ساختمان را از یک یا چند زاویه مختلف برای مشتری تصویر می‌کند

لایه‌بندی و مقوله‌بندی (ادامه)

- یک سیستم نرم‌افزاری نیز همانند یک ساختمان یک موجودیت واحد است، اما برای تولید و توسعه آن، توصیف کاملی از سیستم نیاز است که آن را از زوایای گوناگون مورد بررسی قرار می‌دهد

هرکدام از سهامداران به سیستم از یک زاویه دید معین می‌نگرند

کاربر نهائی، مشتری، مدیر پروژه،
تحلیلگر، طراح، معمار، برنامه‌نویس، ...

لایه‌بندی و مقوله‌بندی (ادامه)

- در توصیف یک سیستم نرم‌افزاری دو اصل وجود دارد:

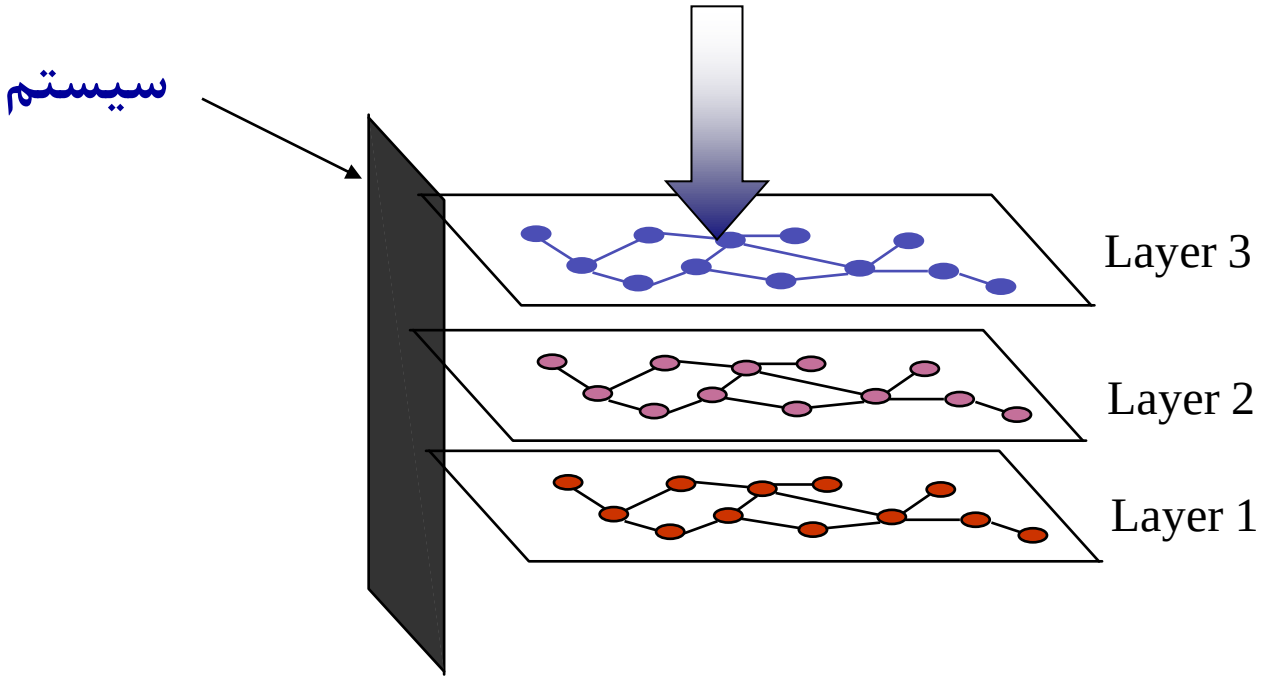
۱) از دید چه کسی این سیستم توصیف می‌گردد؟ ← لایه‌بندی

۲) میزان پرداختن به جزئیات چه قدر است؟ ← مقوله‌بندی

لایه‌بندی و مقوله‌بندی (ادامه)

در این لایه چه گروه‌هایی از کلاس‌ها را می‌توان یافت؟

مقوله‌بندی



لایه‌بندی

لایه‌بندی (Layering)

- در لایه‌بندی یک سیستم نرم‌افزاری به صورت تعدادی از لایه‌ها تقسیم‌بندی می‌گردد
- لایه‌بندی وابستگی‌ها را کاهش می‌دهد بطوریکه لایه‌های پایین‌تر از جزئیات و واسطهای لایه‌های بالاتر اطلاعی ندارند

لایه‌بندی (ادامه)

- یک برنامه کاربردی از نظر منطقی به سه لایه کلی تقسیم می‌شود
  ۱) واسط کاربر (*User Interface*)
  ۲) منطق حرفه (*Business Logic*)
  ۳) سرویس‌های داده‌ای (*Data Services*)

لایه‌بندی (ادامه)

- با توجه به تقسیم‌بندی قبل سه معماری وجود دارد:

۱) معماری متمرکز

۲) معماری *Client/Server*

۳) معماری *3-Tier*

معماری متمرکز

- در این معماری بخش‌های سه‌گانه برنامه شدیداً با یکـدیگر آمیخته‌اند. در واقع، تنها یک لایه وجود دارد.

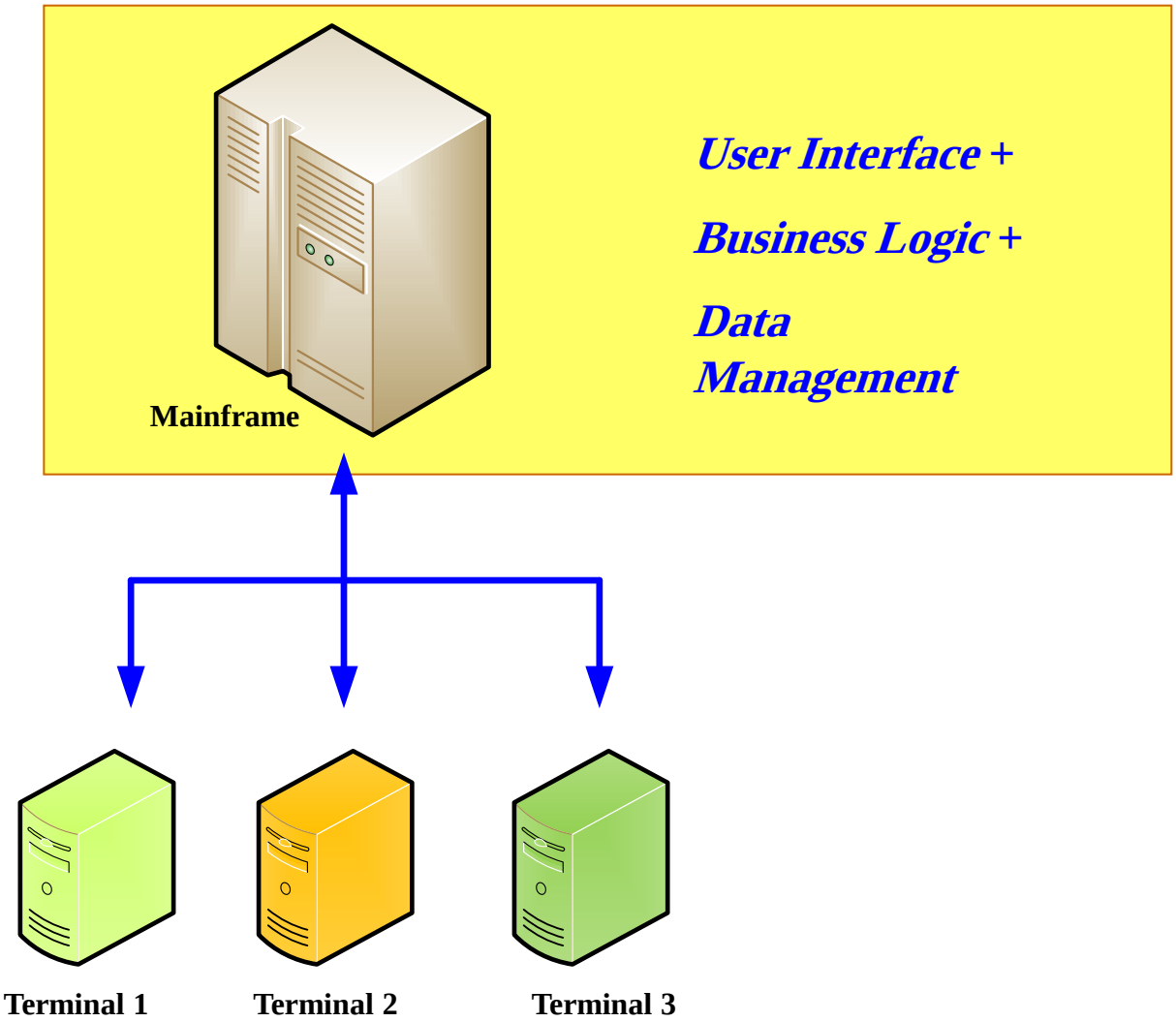
در صورت نیاز به قدرت محاسباتی بیشتر پدیده جزایر برنامه‌های کاربردی (*Applications Islands*) به وجود می‌آید

۱) عدم قابلیت استفاده مجدد

۲) عدم انعطاف‌پذیری و قابلیت تغییر پائین

۳) قابلیت توسعه‌پذیری و مقیاس‌پذیری پائین

معماری متمرکز (ادامه)



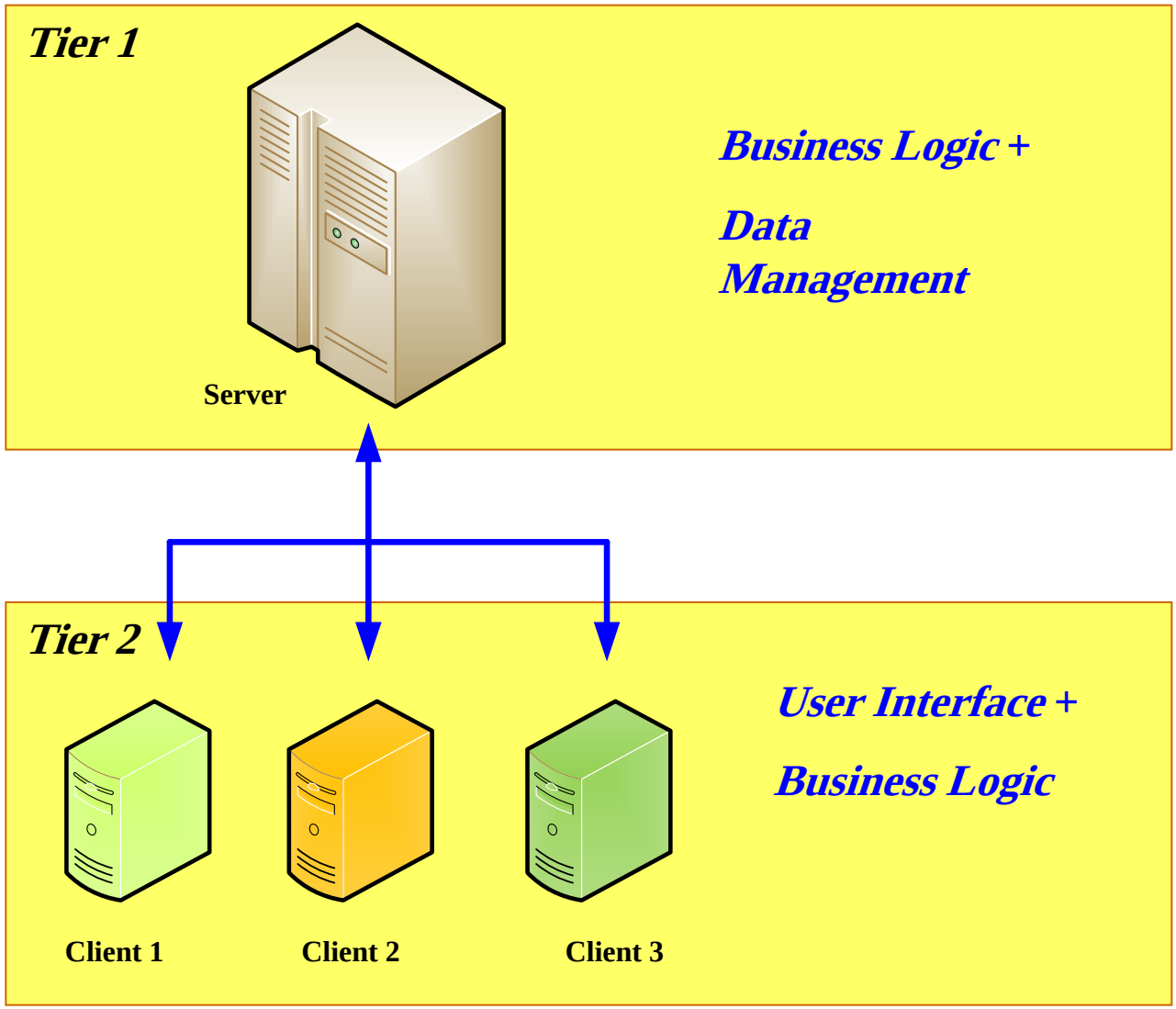
معماری Client/Server

- در این معماری بخش‌های سه‌گانه برنامه در دو لایــه اصــلی قرار می‌گیرند
- لایه‌ها

۱) سرویس‌گیرنده: واسط کاربر + بخشی از منطق حرفه

۲) سرویس‌دهنده: سرویس‌های داده‌ای + قسمت اعظم منطق حرفه

معماری Client/Server (ادامه)



معماری Client/Server (ادامه)

- مزایا
  - واسط کاربر قابل استفاده مجدد است
  - وجود محیط‌های برنامه‌سازی یکپارچه و قوی مانند Delphi، Visual Studio .NET، و ... که امکان دسترسی به داده‌های قدیمی و جدید سازمان به صورت یکپارچه را فراهم می‌کنند
  - امکان استفاده مجدد جزئی از منطق حرفه در قسمت سرویس‌دهنده

معماری Client/Server (ادامه)

- معایب
  - محل منطق حرفه مشخص نیست
  - تنها بخشی از منطق حرفه قابل استفاده مجدد است
  - منطق حرفه قابل استفاده مجدد، معمولاً، به صورت روال‌های ذخیره شده که متعلق به یک پایگاه داده معینی است، وجود دارد. به علت این وابستگی منطق حرفه قابلیت استفاده مجدد را در سطح تمام سازمان نخواهد داشت

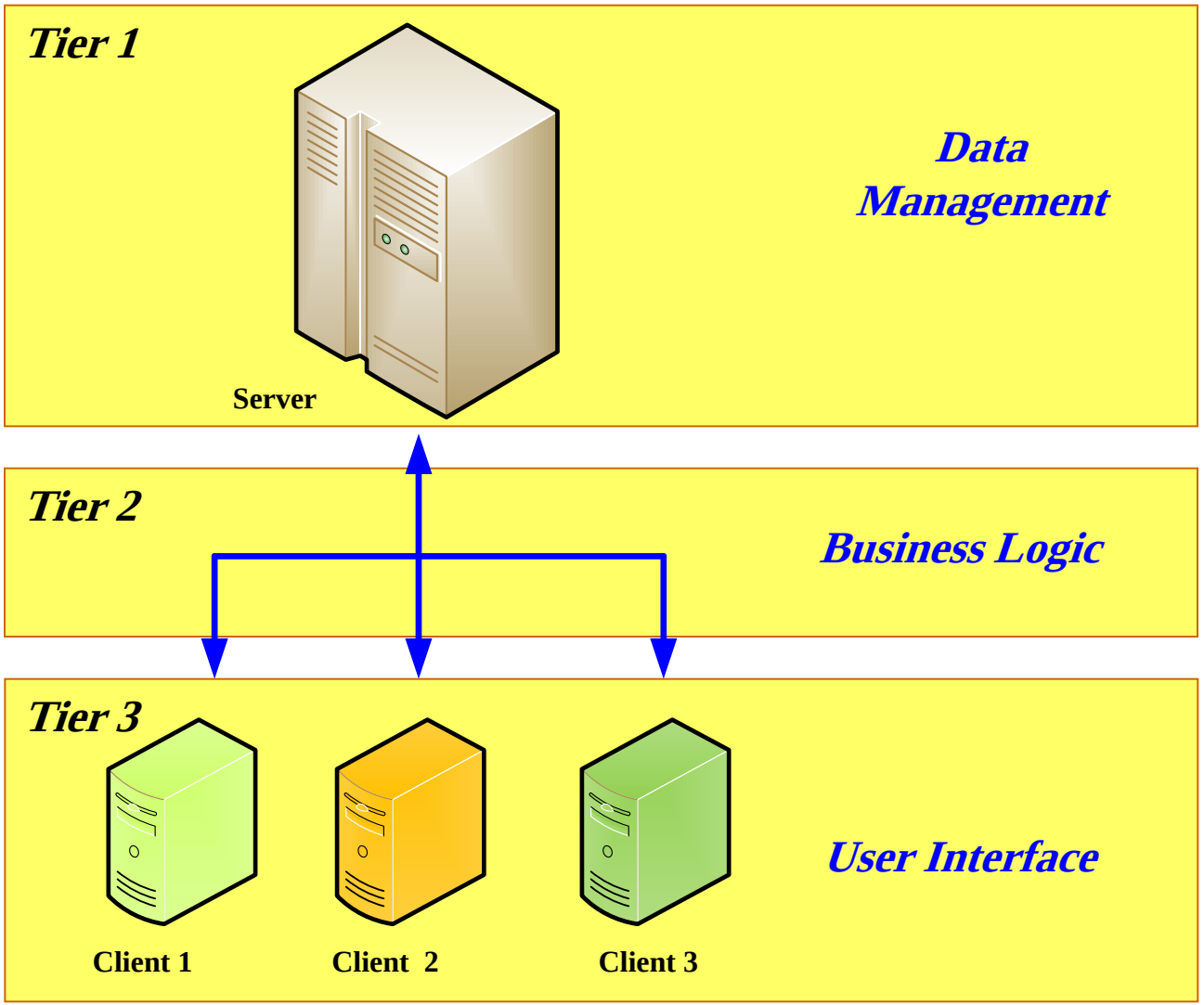
معماری 3-Tier

- در این معماری بخش‌های سه‌گانه برنامه در سه لایه اصلی قرار می‌گیرند
- منطق حرفه بر روی سرویس‌دهنده مستقلی که به سرویس‌دهنده کاربردی (*Application Server*) معروف است، قرار می‌گیرد

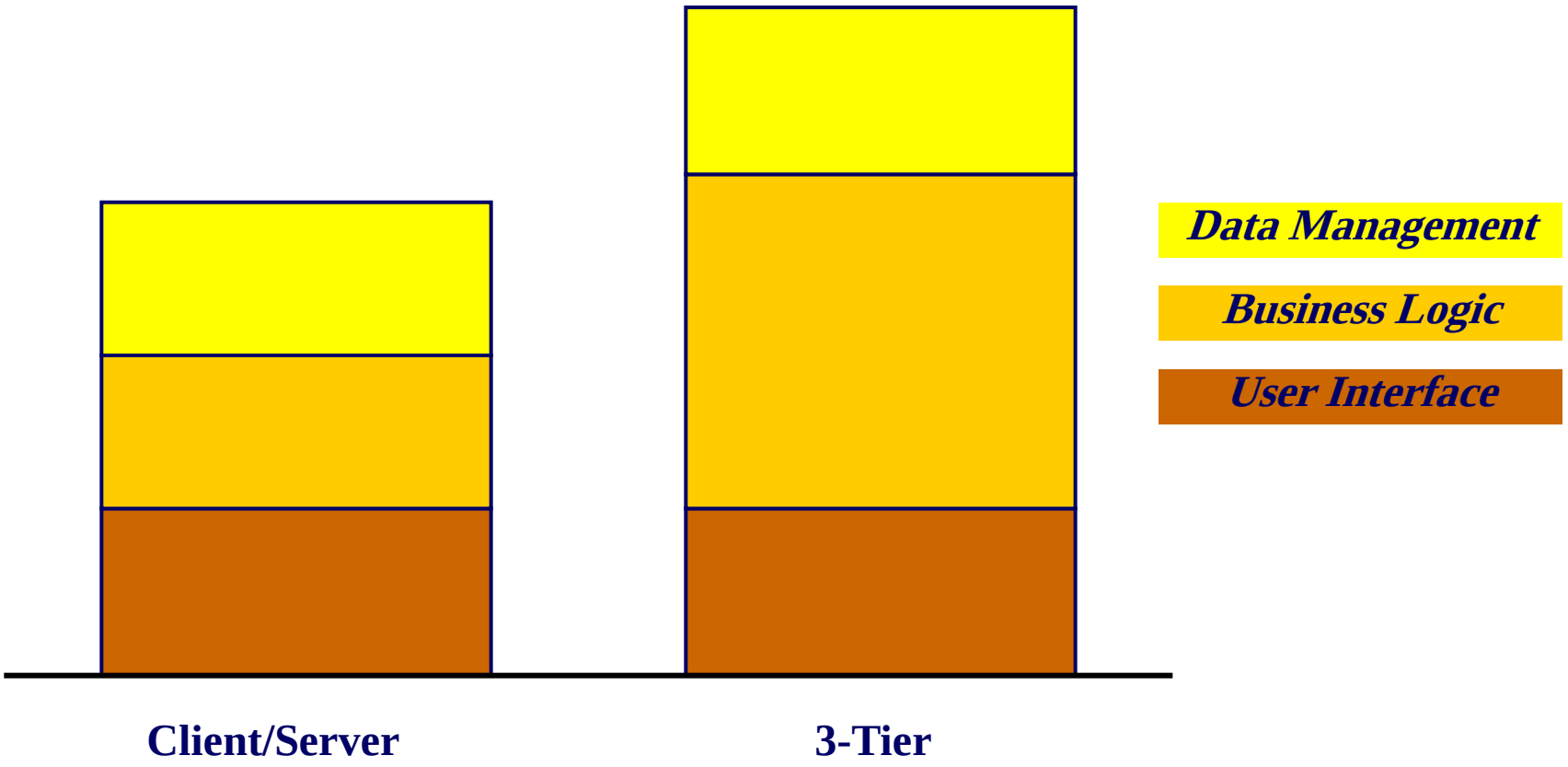
معماری 3-Tier (ادامه)

- مزایا
  - معماری ۳-لایه‌ای از تقسیم‌بندی منطقی برنامه تبعیت می‌کند
  - جداسازی منطق حرفه از بقیه قسمت‌های برنامه باعث به حداکثر رساندن قابلیت استفاده مجدد آن است
- معایب
  - پیچیدگی در طراحی و پیاده سازی

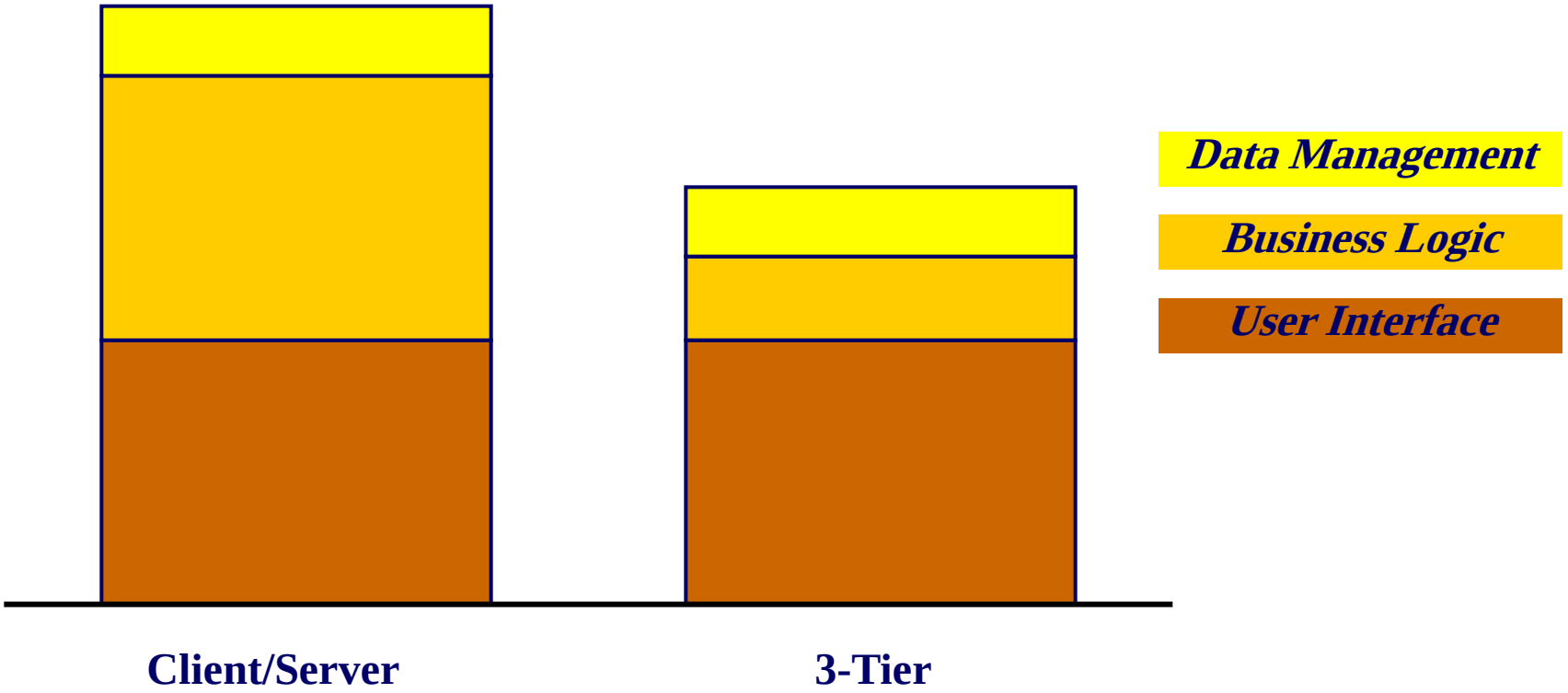
معماری 3-Tier (ادامه)



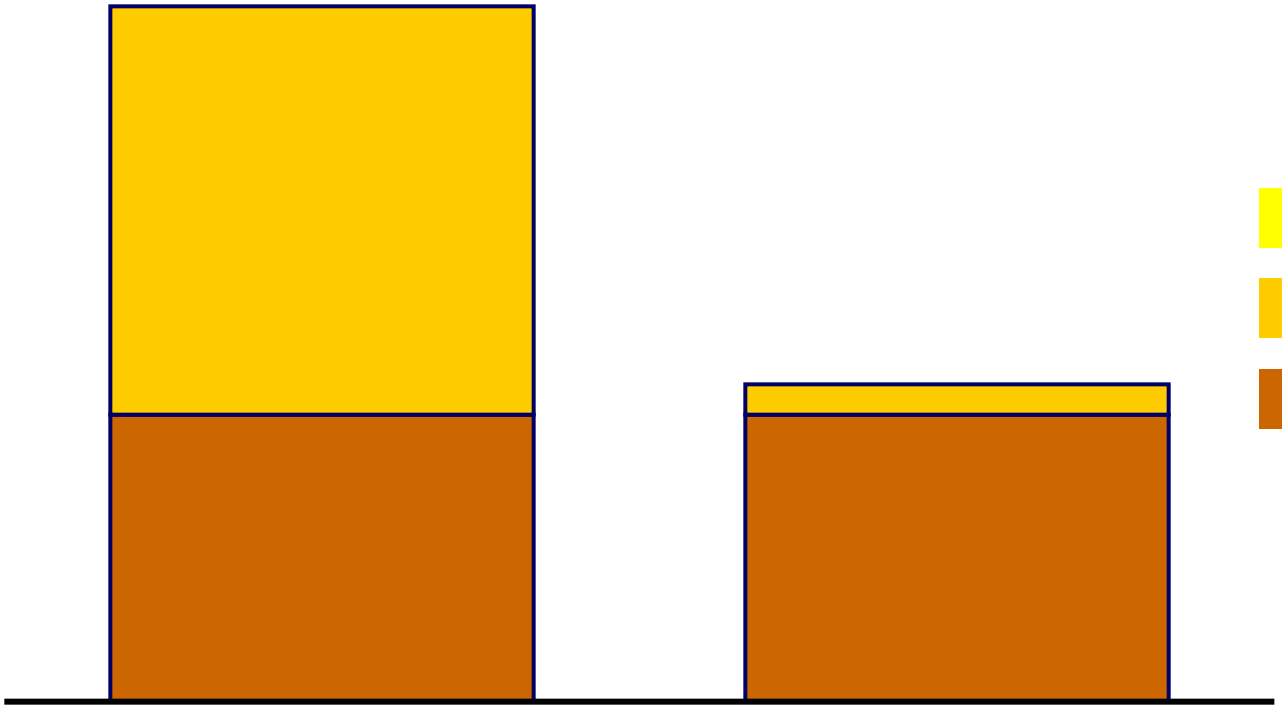
مقایسه معماری C/S و 3-Tier



مقایسه معماری C/S و 3-Tier



مقایسه معماری C/S و 3-Tier



Client Tool Migration

مقوله‌بندی (Stereotyping)

- مقوله‌بندی اشیاء کمک می‌کند که سطوح تجرید در یک سیستم نرم‌افزاری را شناخته شده و بدین صورت تمام

مقوله‌بندی روشی برای تنظیم سطح تجرید است

(*Abstraction Level Tuner*)

مقوله‌بندی (ادامه)

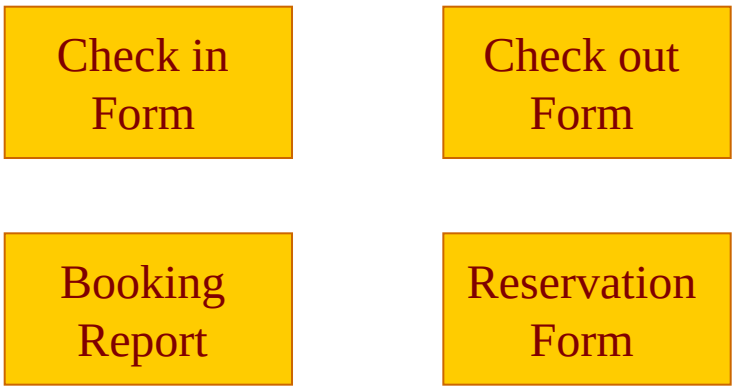
- مثال: مقوله‌بندی در متدولوژی‌های
  - ***Unified Software Development Process*** **(USDP)**
  - ***Select Perspective***
  - ***Responsibility Driven Design***

مقوله‌بندی در USDP

- در USDP کلاس‌ها به سه گروه زیر تقسیم می‌شوند
  - کلاس‌های مرزی (*Boundary Classes*)
    - مثال: واسط کاربر، پروتکل‌های ارتباطی، واسط چاپگر، حسگرها و ...
  - کلاس‌های کنترلی (*Control Classes*)
  - کلاس‌های موجودیتی (*Entity Classes*)
    - کلاس‌های موجودیتی مفاهیم کلیدی در سیستم در حال توسعه را نمایش می‌دهند

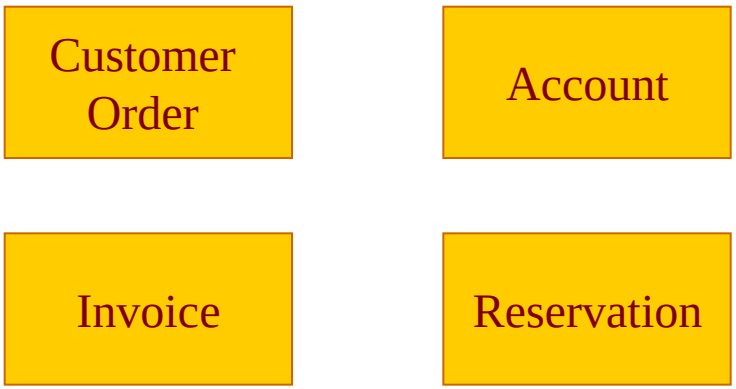
مقوله‌بندی در Perspective

- کلاس‌های واسط (*Interface Classes*)
  - در سیستم هتل‌داری: اپراتوری که با سیستم کار می‌کند با فرم‌هایی(همان اشیاء واسط) ارتباط دارد مانند: فرم رزرو اتاق، فرم حساب شخص، گزارشی از تعداد افراد و ...



مقوله‌بندی در Perspective (ادامه)

- کلاس‌های حرفه (*Business Classes*)
  - در سیستم هتل‌داری: اشیائی مانند رزرو کردن و درخواست مشتری، فاکتور، حساب و ... مشاهده می‌شوند که سرویس‌های مورد انتظار از سیستم را فراهم می‌نمایند



مقوله‌بندی در Perspective (ادامه)

- کلاس‌های داده‌ای (*Data Classes*)
  - در سیستم هتل‌داری: اشیائی مانند مدیر داده‌های حساب، تبدیل‌کننده داده‌های حساب، دستیابی به داده‌های حساب و... مشاهده می‌شوند که سرویس‌های داده‌ای مورد انتظار اشیاء دیگر را فراهم می‌نمایند

Account Data Conversion

Account Data Manager

Account Data Access

مقوله‌بندی در RDD

- در RDD کلاس‌ها به ۶ گروه زیر تقسیم می‌شوند
  - کلاس‌های کنترل کننده (*Controller Classes*)
  - کلاس‌های هماهنگ کننده (*Coordinator Classes*)
  - کلاس‌های واسط (*Interface Classes*)
  - کلاس‌های فراهم‌کننده سرویس (*Service Provider Classes*)
  - کلاس‌های نگهدارنده اطلاعات (*Information Holder Classes*)
  - کلاس‌های ساختاری (*Structure Classes*)

کارت‌های CRC و مقوله‌بندی

- از کارت‌های CRC می‌توان برای تشخیص طبقه‌بندی (مقوله‌بندی) اشیاء استفاده نمود
  - مثال: در یک سیستم بانکی مشتری می‌خواهد مبلغی از حساب خود برداشت نماید. این کار بوسیله ماشین پرداخت پول(*Automatic Teller Machine*) انجام می‌گیرد.

کارت‌های CRC و مقوله‌بندی (ادامه)

Withdrawal Transaction	
Supercalss: Financial Transaction	
Responsibilities	**Collaborators**
Knows account Knows amount Performs Withdraw Logs Transaction Initiates Dispensing cash	Cash Dispenser

کارت‌های CRC و مقوله‌بندی (ادامه)

Withdrawal Transaction	
Purpose	Withdraws cash from an account and dispense it
Stereotypes	Service Provider, Coordinator, Business Object

پشت کارت Withdrawal Transaction که در آن طبقه‌بندی کلاس دیده می‌شود

پرسش و پاسخ