بسمه تعالی

دانشگاه صنعتی شریف

طراحی شی‌گرا

تمرین اول

استاد

دکتر رامسین

تهیه و تدوین

اشکان خادمیان (۹۸۱۰۵۷۳۸)

بهار ۱۴۰۲

فهرست

مقدمه 3
1. نمودار ساختاری 5
مقایسه از منظر مقصود و سطح انتزاع 5
مقایسه ساختاری 8
۲. نمودار تعامل 14
مقایسه از منظر مقصود 15
مقایسه ساختاری 16
۳. نمودار مورد کاربرد 19
مقایسه از منظر مقصود 19
مقایسه ساختاری 20
References 25

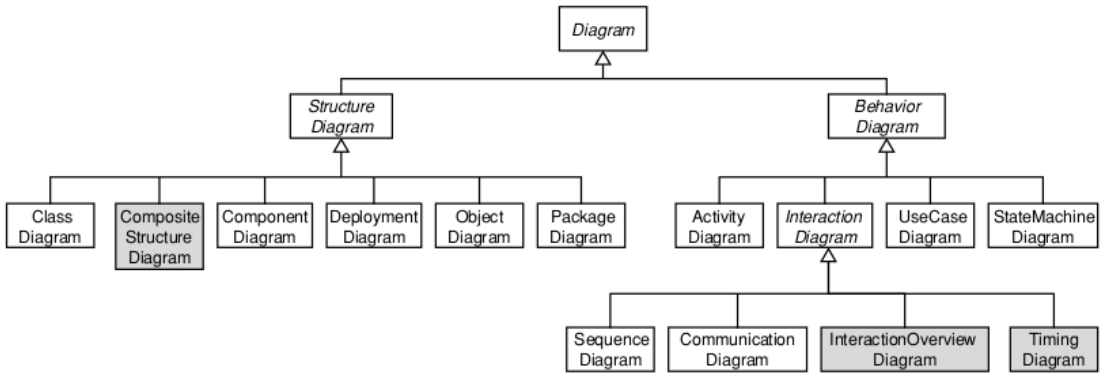
مقدمه

UML یک زبان مدل‌سازی بصری همه منظوره است که در سیستم‌های گوناگون مورد استفاده قرار می‌گیرد. این زبان برای توصیف، تجسم، ساختن و مستند سازی مصنوعات سیستم‌های نرم‌افزاری استاندارد شده است. با این حال از آنجایی که نمودارهای آن تا حد زیادی عمومیت دارند، برای عرصه‌هایی غیر از نرم‌افزار نیز کاربردی هستند.

در (۳)، ارزش چنین تلاشی برای داشتن یک روش متحد مدل‌سازی در مهندسی نرم‌افزار به وضوح بیان شده است.

> **UML نتیجه تلاش برای اتحاد زبان‌های مدل‌سازی بصری است که در متدلوژی‌های شی‌گرا استفاده می‌شدند؛ چنین اتحادی حاصل از این درک بوده است که تمامی این متدلوژی‌ها اگرچه دارای مدل فرایند و چرخه زندگی متفاوت بودند، اما بیشتر آن‌ها از نمودارهایی برای مدل‌سازی استفاده می‌کردند که در ذات یکسان بودند. از این رو، شروع ترند یکپارچگی و اتحاد زبان‌های مدل‌سازی یک انتخاب معقول به نظر می‌رسید. به همین سبب UML، به عنوان یک نقطه عطف برجسته در نظر گرفته می‌شود که پایان متدلوژی‌های بدوی و شروع مسرت یکپارچگی را رقم می‌زند.**

همانطور که تصویر ۰ نشان می‌دهد، UML سیزده نوع نمودار ارائه می‌دهد که هر کدام ساختار و هدف خاص خود را دارد.



تصویر ۰. انواع مختلف نمودار در UML 2، برگرفته از (۱).

در این گزارش ما قصد داریم با پاسخ به سه پرسش زیر در مفاهیم UML به درک عمیق‌تری دست یابیم.

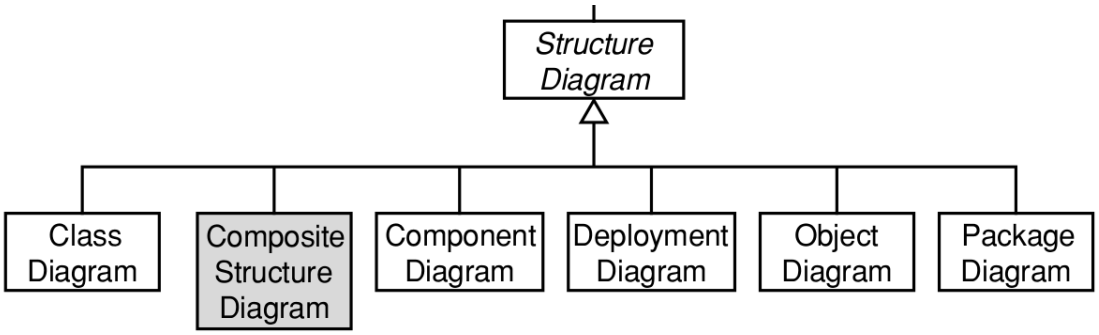
- نمودارهای ساختاری[1] را با یکدیگر از منظر مقصود، ساختار و انتزاع مقایسه کنید.
- نمودارهای تعامل[2] را با یکدیگر از منظر مقصود و ساختار مقایسه کنید. کدام یک از این نمودارها به منظور مدل‌سازی تحقق مورد کاربرد[3] استفاده می‌شود؟
- نمودار مورد کاربرد را با هر یک از نمودارهای رفتاری زیر از منظر ساختار و مقصود مقایسه کنید.
  - نمودار فعالیت[4]
  - ماشین حالت

[1] Structural diagrams
[2] Interaction diagrams
[3] Use case
[4] Activity diagram

1. نمودار ساختاری

نمودارهای ساختاری را با یکدیگر از منظر مقصود، ساختار و انتزاع مقایسه کنید.

به عنوان یادآوری خوب است که به تصویر ۱.۱ که شش نمودار ساختاری موجود در UML 2 را لیست کرده است توجه کنیم.



تصویر ۱.۱. نمودارهای ساختاری برگرفته از (۱). شش مدل نمودار ساختاری در UML 2 وجود دارد؛ نمودار کلاس (رده)، نمودار ساختار مرکب، نمودار مولفه، نمودار استقرار، نمودار شی و نمودار پکیج.

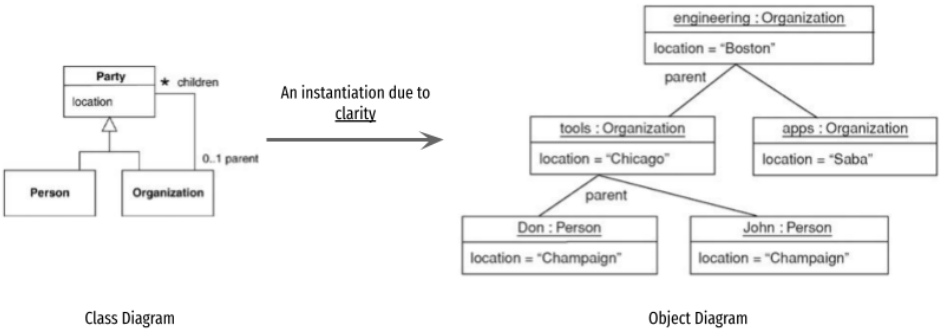
مقایسه از منظر مقصود و سطح انتزاع

نمودار کلاس (رده)، کلاس‌های موجود در سیستم و ارتباط آن‌ها با یکدیگر را نمایش می‌دهد و با توجه به آن، می‌توان دیدهای خارجی موجود برای هر کلاس را شناسایی کرد. این نوع نمودار، ابزار اصلی ما برای ساخت یک سیستم به روش شی‌گرا می‌باشد و می‌توان از آن در هر دو فضای مسئله و جواب (تحلیل و طراحی) بهره جست. البته باید توجه داشت که کلاس‌های مدل طراحی تا سه برابر کلاس‌های مدل تحلیل حجم اطلاعات و تعدد دارند؛ به عبارتی هر کلاس از فضای مسئله با یک یا چند کلاس در فضای جواب متناظر می‌شود و همچنین اطلاعات مختص فضای جواب به آن اضافه می‌شود (اطلاعات بیشتری در این باره، اینجا مطرح شده است).

نمودار شی به عنوان یک مکمل برای نمودار رده است. هر نمودار شی در واقع تصویر لحظه‌ای[5] از اشیای سیستم است؛ علی‌الخصوص لحظات استثنایی سیستم مانند ساختار مورد نیاز شی‌ها برای رخ دادن رویداد آفساید در یک سیستم شبیه‌ساز فوتبال. برای موارد استثنایی مشابه یا توضیح بخش‌هایی

[5] Snapshot

از نمودار رده‌ی سیستم که پیچیدگی بالایی به دلیل روابط بین‌کلاس‌ها دارند، می‌توان از این نمودار استفاده کرد تا نمایشی از نحوه استقرار شی‌ها در زمان اجرا[6]، به فهم ما از سیستم کمک کند. تصویر ۱.۲ تدقیق[7] یک مدل کلی از ساختار چند کلاس سیستم با استفاده از نمودار شی را به خوبی نمایش می‌دهد.



تصویر ۱.۲. نمودار شی. در (۲) مثالی ساده از تدقیق یک نمودار کلاس توسط نمودار شی با هدف شفاف‌سازی آورده شده است. نمودار شی و نمودار کلاس مربوطه با هم در تصویر فوق آمده‌اند.

در UML 2 یک مدل جدید به نام **نمودار ساختار مرکب** معرفی شده است که آن هم عمدتاً هدف تدقیق بخش‌های خاصی از نمودار رده را دنبال می‌کند. عمده دلیل استفاده از این نمودار شرایطی است که کلاس‌های حاضر در یک رابطه composition از نمودار رده، هر کدام دارای چند نقش مختلف‌اند که هر کدام از نقش‌ها معنا و محدودیت‌های خاص خود را دارند (انواع روابط در یک نمودار کلاس اینجا مطرح شده است). اگر خطر پادالگوی God Class را نادیده بگیریم، تصویر ۱.۵ مثال خوبی از چنین شرایطی است.

در دیدهای بالاتر نسبت به سیستم، می‌توان از **نمودار پکیج** استفاده کرد. این نوع نمودار به صورت غیر رسمی در ورژن‌های پیشین UML استفاده می‌شده است اما به صورت رسمی در ورژن دوم UML با هدف هرگونه مدیریت پیچیدگی (در هر نوع از نمودار UML) اضافه شد؛ اگرچه در عمل، تنها برای افراز کلاس‌های فضای مسئله استفاده می‌شود. یکی از اصلی‌ترین دلایل ورود نکردن این نمودار به فضای

[6] Run-time
[7] Refine

جواب در بسیاری از متدلوژی‌های نرم‌افزار مانند UP، احتمالاً نگاه مولفه‌محور[8] به قلمرو طراحی جواب مهندسین نرم‌افزار می‌باشد.

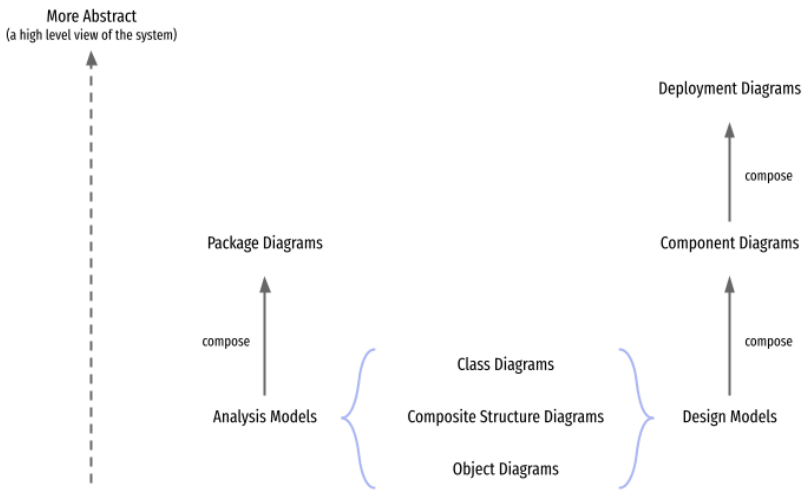
یک نگاه انتزاعی‌تر به فضای نرم‌افزار با هدف برقرار کردن اصل جداسازی دغدغه‌ها[9]، به طراحی مبتنی بر مولفه منتهی می‌شود. در واقع در قلب هر مولفه نرم‌افزاری، کلاس‌های قلمروی جواب در حال فعالیت هستند. برای مدل‌سازی این نوع نگاه، از **نمودار مولفه** استفاده می‌شود.

مفاهیم مولفه اولین بار توسط Brad Cox با واژه Software IC معرفی شد و پس از آن همواره مورد بحث در حوزه نرم‌افزار باقی ماند. لختی دیگر، در اینجا، به ویژگی‌های مولفه نرم‌افزاری خواهیم پرداخت.

در نهایت، **نمودار استقرار** ساختار توزیع مولفه‌های نرم‌افزاری روی سکو[10]های مختلف نرم‌افزاری یا سخت‌افزاری را نشان می‌دهد. (۴) هدف اصلی معماری نرم‌افزار را تسهیل چهار جنبه[11] سیستم نرم‌افزاری می‌نامد که یکی از آن‌ها "فرایند استقرار" است. از طرفی دیگر می‌دانیم معماری نرم‌افزار راجع به دو موضوع مهم مولفه‌ها و اتصال‌دهنده‌ها[12] صحبت می‌کند. همه این موارد اهمیت یک نگاه سطح بالا به سکو و مولفه برای ایجاد و نگهداری یک معماری صحیح را نشان می‌دهند.

با توجه به تمامی مواردی که گفته شد، از منظر عمومیت و اینکه هر نمودار چه نمودار دیگری را در بر می‌گیرد، می‌توان سلسله مراتبی شبیه به تصویر ۱.۳ را برای نمودارهای ساختاری در نظر گرفت.

[8] Component-based
[9] Separation of concerns
[10] Platform
[11] ایجاد، استقرار، عملکرد و نگهداری نرم‌افزار
[12] Components and Connectors



تصویر ۱.۳. یک انتخاب برای سطوح انتزاع نمودارهای ساختاری. از طرفی در فضای جواب هر پکیج در نمودارهای پکیج، افرازی از چند کلاس در قلمروی مسئله هستند و مولفه‌های موجود در نمودارهای مولفه‌ای، شامل چندین کلاس از قلمرو جواب می‌شوند. همچنین نمودار استقرار به نحوه‌ی چینش و توزیع بالارده‌ترین مولفه‌های‌های سیستم روی سکو می‌پردازد. از طرف دیگر نمودارهای ساختار مرکب و شی تدقیق شده‌ی بخش‌های کوچکی از یک نمودار رده می‌باشند و جامعیت کمتری دارند.

مقایسه ساختاری

از بین تمامی نمودارهای ساختاری UML، **نمودار کلاس** احتمالاً غنی‌ترین از نظر تعدد نمادها است. نمایشی که این نمودار برای مفهوم کلاس در اختیار ما قرار می‌دهد شامل نام کلاس، صفات[13] و عملیات‌ها[14] (در پیاده‌سازی‌های شی‌گرا با نام متد نیز شناخته می‌شوند) است. این نمودارها ساختار مناسب برای مدل‌سازی هر دو قلمروی مسئله و جواب را فراهم می‌سازند؛ چرا که از نمایش جزییات زائد وابسته به نوع پیاده‌سازی، مانند دسترسی‌ها[15] (public, private, protected) و جنس متغیرها، صفات و مقدار خروجی متدها، برای مدل‌سازی فضای مسئله صرف نظر می‌شود؛ این دسته جزییات را با در نظر گرفتن فناوری‌های نرم‌افزاری مورد استفاده، معماری نرم‌افزاری برگزیده و تصمیمات برجسته در سطح معماری تنها در قلمروی جواب نمایش می‌دهیم.

[13] Attributes
[14] Operations
[15] Access (aka. Visibility)

از طرف دیگر، خود ماهیت کلاس‌هایی که در نمودار کلاسیِ تحلیل مورد نمایش قرار می‌گیرند، بسیار انتزاعی است. در واقع این کلاس‌ها باید در سطحی از انتزاع قرار داشته باشند که مستقل از هرگونه پیاده‌سازی (حتی غیر نرم‌افزاری) موضوعیت داشته باشند. همچنین، کلاس‌های موجود در نمودار کلاسیِ طراحی باید به قدری دقیق باشند که یک مسئول ایجاد با در دست داشتن مدل‌های طراحی و دانش از تکنولوژی‌های مورد استفاده بتواند بدون هیچ ابهامی، یک پیاده‌سازی مابه ازای آن مدل‌ها تحویل دهد.

در نمادهای مورد استفاده نمودارهای رده، چندین نوع نماد برای نمایش روابط بین کلاس‌ها نیز موجود است. این روابط می‌توانند نشانگر موارد زیر باشند.

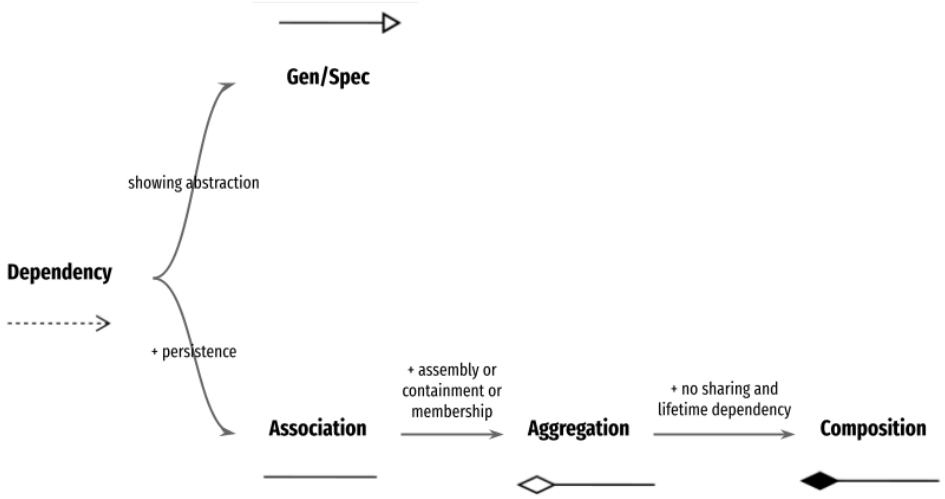
- دید مجاز از یک کلاس به کلاسی دیگر.
- سطوح انتزاعی مورد تصور برای کلاس‌ها.
- پیوند موجود بین کلاس‌ها.

ساده ترین نوع رابطه، رابطه وابستگی بین دو کلاس است. این نوع رابطه به دیدهای گذرای سیستم در زمان اجرا اشاره می‌کند؛ برای مثال فراخوانی یک عملیات از کلاس B توسط یک شی از کلاس A منجر به تصور یک رابطه وابستگی ساده در نمودار کلاسی این سیستم می‌شود که از سمت A به B است.

اگر این رابطه وابستگی مانا باشد، یعنی در طول زمان معتبر بماند، آن را رابطه انجمنی[16] می‌دانیم. همچنین اگر این مانایی تحت اثر سطوح انتزاع و تدقیق باشد آن را Gen/Spec می‌دانیم.

در نهایت اگر یکی از سه ویژگی گردآمدن، شامل بودن و عضویت برای یک رابطه انجمنی برقرار باشد آن را رابطه تجمعی[17] می‌نامیم. در روابط تجمعی همواره یک جزء و یک کل داریم. اگر کل‌ها بین اجزای یکدیگر اشتراک‌گذاری نکنند (هر جزء تنها به یک کل مربوط باشد) و همچنین عمر اجزا وابسته به عمر کل باشد، می‌گوییم رابطه از نوع ترکیب[18] است. چنین نوع نگاهی که همه روابط موجود را حاصل تدقیق رابطه وابستگی بدانیم در تصویر ۱.۴ رسم شده است.

[16] Association
[17] Aggregation
[18] Composition



تصویر ۱.۴. انواع روابط در نمودار کلاس. به طور عمده پنج نوع رابطه‌ی وابستگی، Gen/Spec، انجمنی، تجمعی و ترکیب در نمودار رده مورد استفاده قرار می‌گیرد.

در مقابل نمودار کلاس، که به عبارتی برای سیستم (مقیت در قلمروی تحلیل یا طراحی) عمومیت دارد، **نمودار شی** «تصویر لحظه‌ای» از اشیای مقیم در بخشی از سیستم در لحظه‌ای خاص را نمایش می‌دهد (در این باره به تفصیل در اینجا بحث شده است). این ذات تدقیق شده‌ی نمودار شی در نمادگذاری[19] این نمودار نیز مشهود است؛ توضیح اینکه، این نمودار نمادهای محدود و ساده‌ای دارد که عمدتاً از نماد شی، که شامل نام و جنس شی (کلاسی که آن شی را راه اندازی می‌کند) است، و پیوند بین اشیا تشکیل شده است.

همانطور که مشهود است برخلاف نمودار کلاس، این نمودار حاوی توصیفات بسیار چه در شی و چه در پیوند بین آن‌ها که باعث جامعیت نمودار در نمایش سیستم باشد نیست.

نمودار ساختار مرکب، که در UML 2 اضافه شده است، یک ساختار تو در تو[20] دارد؛ توجه داشته باشیم که دو نمودار ساختاری که پیش‌تر بررسی کردیم از هیچ نمادگذاری تو در تویی پشتیبانی نمی‌کنند و در آن‌ها پیچیدگی‌هایی که نیازمند نمایش تو در تو دارند را تا حد محدودی می‌توان با انواع روابطی که در اختیار مدل‌ساز قرار می‌دهند نمایش داد.

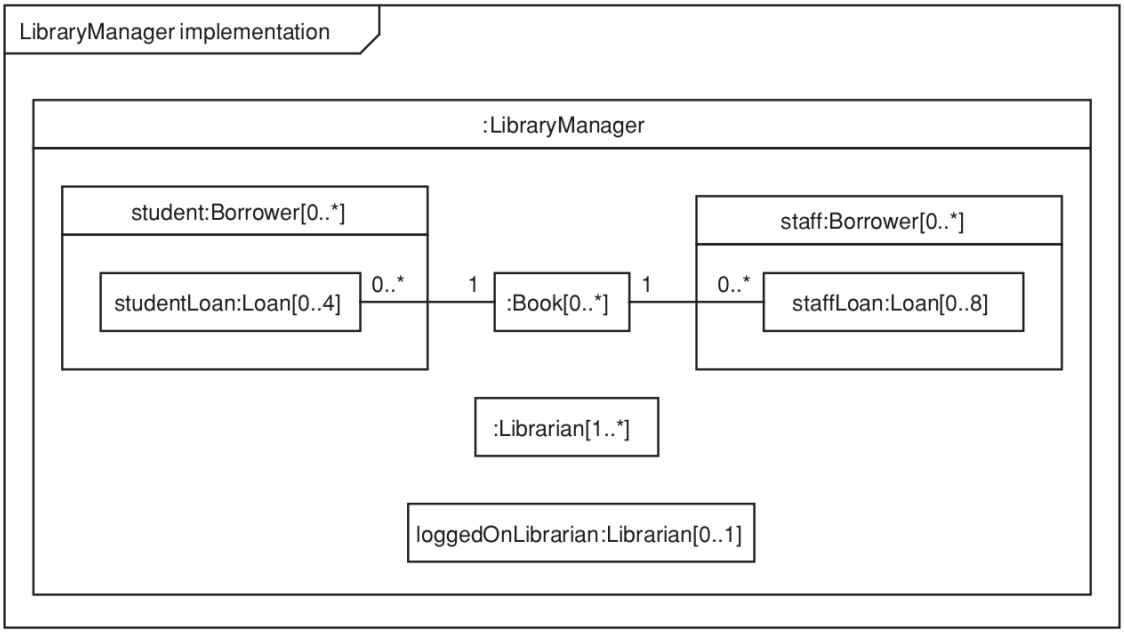
[19] Notation
[20] Nested

از نظر شکل نمادها اما، این نمودار چیزی ما بین نمودار کلاس و نمودار شی است. این نمودار نماد مربوط به نمایش شی را از نمودار شی و مربوط به نمایش محدودیت‌های حضور در یک رابطه را از نمودار کلاس به وام گرفته است.

در این نمودار نمایش هر شی موجود در ساختار مرکب علاوه بر جنس کلاس مربوط به خود، یک اسم به عنوان «نام مشارکت[21]» برای خود معرفی می‌کند. این نام مشارکت در واقع نقشی است که این شی در این ساختار بازی می‌کند. توانمندی مهمی که این مدل در اختیار دارد این است که در آن می‌توان برای اشیای شرکت کرده از یک کلاس مشابه، نقش‌های مختلفی تعیین کرد. برای مثال در تصویر ۱.۵، اشیای کلاس **Borrower** می‌توانند دو نقش مختلف در دل اشیای کلاس **LibraryManager** ایفا کنند: عضو پرسنل (staff) و دانشجو (student). نکته مهم دیگر آن است که اشیای مختلف، حتی اگر همجنس باشند، می‌توانند شروط متفاوتی را در قالب نقش‌های مختلف تجربه کنند. برای مثال در همین تصویر، به هر دانشجو حداکثر چهار کتاب و به هر عضو پرسنل حداکثر هشت کتاب قرض داده می‌شود. این شرایط درحالی است که جنس دانشجو و عضو پرسنل هر دو کلاس **Borrower** و جنس قرض کتاب برای دانشجو و برای پرسنل هر دو کلاس **Loan** می‌باشد.

چنین شرایطی با استفاده از رابطه ترکیب در نمودار کلاس قابل نمایش نیست زیرا از نمادی تحت عنوان نام مشارکت (که نشان‌دهنده نقش شی باشد) پشتیبانی نمی‌کند. همچنین، محدودیت‌های به نمایش گذاشته شده در این نمودار قابل بیان در یک نمودار شی نیست چرا که نمودار شی ذاتاً برای بیان‌ها عام نیست و صرفاً یک تصویر لحظه‌ای از نحوه استقرار اشیاست. به همین سبب وجود چنین نموداری لازم می‌شود.

[21] Partname



تصویر ۱.۵. مثالی برای نمودار ساختار مرکب از (1). دیدهای مانای بین اشیا به همراه محدودیت تعداد اشیای حاضر از هر نقش در هر پیوند ذکر شده است.

نمودارهای پکیج یک ساختار عمومی دارند و برای ثبت یک دید سطح بالا نسبت به کل سیستم و زیر سیستم‌های تشکیل‌دهنده آن‌ها استفاده می‌شوند. این نمودارها شامل پکیج‌های سیستم و وابستگی بین آن‌ها هستند. وابستگی بین پکیج با یک کمان خط چین نمایش داده می‌شود (این کمان ما را یاد نمایش وابستگی در نمودار رده نیز می‌اندازد). درون هر پکیج می‌تواند پکیج‌های دیگر قرار گیرد (مانند نمودار ساختار مرکب از ساختار تو در تو پشتیبانی می‌کند) و داخلی‌ترین جزء تجزیه‌ناپذیر[22] آن را کلاس‌های سیستم تشکیل می‌دهند؛ پیش‌تر در اینجا توضیح داده شد که این کلاس‌ها غالباً مربوط به فضای مسئله هستند.

از آنجایی که در شی‌گرایی، ما یک تفکر مبتنی بر مولفه برای تقسیم قلمروی جواب به ماژول‌های مستقل داریم، به جای نمودار پکیج از **نمودارهای مولفه** برای مدل‌سازی طراحی استفاده می‌کنیم. این نمودار از نمادی برای نمایش مولفه استفاده می‌کند که از متدلوژی Booch به وام گرفته شده است. مولفه‌ها نیز مانند پکیج‌های نمودار پکیج از ساختار تو در تو پشتیبانی می‌کنند اما داخلی‌ترین جزء تجزیه‌ناپذیر آن‌ها را کلاس‌های طراحی تشکیل می‌دهند. استفاده از کلاس‌های تحلیل در فضای نرم‌افزاریِ مولفه‌ها که صرفاً یک جواب برای قلمرو مسئله را توضیح می‌دهند، یقیناً موضوعیت ندارد.

[22] Atomic

این نمودار به قطع مانند اجزای دیگر طراحی شی‌گرا از روابط پشتیبانی می‌کند اما جنس این آن‌ها مانند رابطه بین کلاس‌ها نیست. مهم‌ترین تفاوت آن این است که کلاس‌ها می‌توانند نسبت به داخل یکدیگر دید برقرار کنند اما یکی از مهم‌ترین ویژگی‌های مولفه‌ها نبود دید خارجی برای آن‌هاست و تمامی ارتباط با آن‌ها از طریق واسط[23] هایی که به اشتراک می‌گذارند محقق می‌شود. این ویژگی در نمودار مولفه نیز منعکس شده است و تمامی ارتباطات بین مولفه‌ای با استفاده از واسط‌هایی که از یک سمت در اختیار گذاشته شده‌اند (نماد دایره) و از سمت دیگر درخواست شده اند (نماد کاسه) برقرار می‌شوند. این مورد یک تفاوت عمده نسبت به نمودارهای ساختاری‌ای اعم کلاس، شی و ساختار مرکب است که باید مورد توجه واقع شود تا مدل‌ساز به اشتباه سعی نکند بین مولفه‌ها دید برقرار کند.

یک نکته مهم این است که مولفه‌ها نمایانگر قطعاتی هستند که می‌توان آن‌ها را مستقلاً خریداری کرد و یا ارتقا داد. (۲)

وقتی هدف نمایش، ساختار استقرار مولفه‌های نرم‌افزاری روی سکوهای مختلف باشد، استفاده از نمودار استقرار موضوعیت پیدا می‌کند. این نمودار شامل نمادهایی به نام گره[24] است که نشانگر سکوهای نرم‌افزاری/سخت‌افزاری هستند. هر گره می‌تواند یا یک دستگاه سخت‌افزاری باشد یا یک سکوی نرم‌افزاری به عنوان محیط اجرا. همچنین مصنوعات «به خود متکی[25]»، که معمولا از تیم‌های خارجی خریداری شده‌اند و کد مبدا آن‌ها قابل دسترسی نیست، به شکل‌های مختلفی می‌توانند نمایش داده شوند و به آنها «مصنوعات مستقر[26]» گفته می‌شود. مسیر ارتباطی بین گره‌ها به همراه اطلاعاتی همچون پروتکل ارتباطی (مانند REST, gRPC و ...) و بستر ارتباطی (مانند Lan, Internet و ...) نیز نمایش داده می‌شوند.

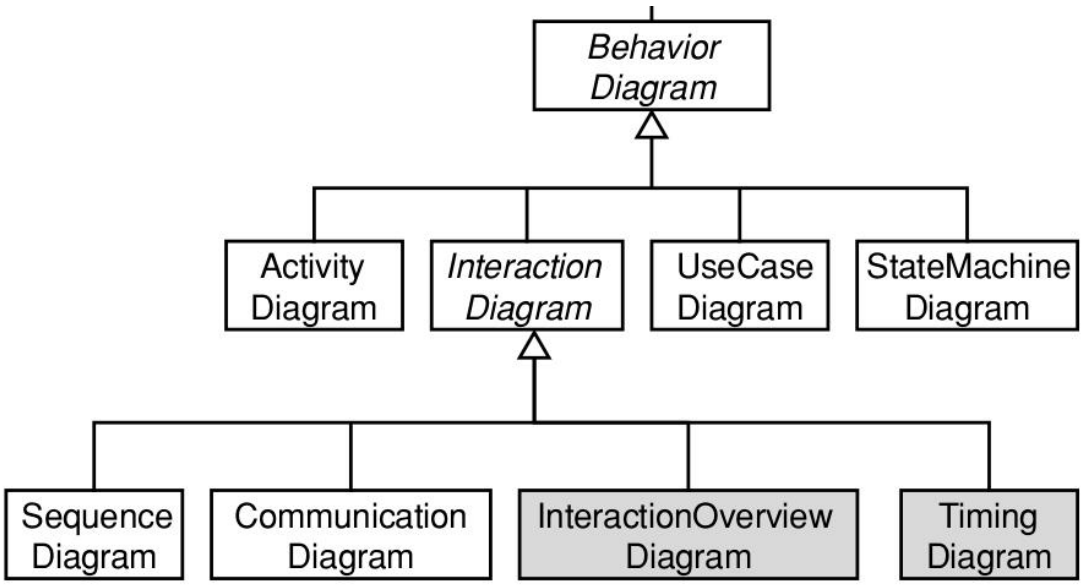
[23] Interface
[24] Node
[25] Standalone
[26] Deployed artifacts

۲. نمودار تعامل

نمودارهای تعاملی را از منظر ساختار و مقصود با یکدیگر مقایسه کنید. کدام یک از این نمودارها نوعاً برای مدل‌سازی تحقق مورد کاربرد استفاده می‌شود؟

همانطور که چیزهای ساختاری یک مدل UML همچون اسم‌های مدل هستند، چیزهای رفتاری مانند فعل‌های آن می‌باشند. (۱)

UML 2 سه روش برای مدل‌سازی رفتاری پیشنهاد می‌کند: نمودار تعامل، نمودار ماشین حالت و نمودار فعالیت. همانطور که تصویر ۲.۱ نشان می‌دهد، نمودار مورد کاربرد ذیل نمودارهای رفتاری دسته‌بندی شده است که کاملاً دقیق نیست. «رفتار» با ترتیبی از کنش‌ها و کارکرد[27] ها متناظر است. از آنجایی که نمودار مورد کاربرد صرفاً نیازمندی‌های کارکردی سیستم را مدل‌سازی می‌کند و ترتیب اجرای آنها را در نظر نمی‌گیرد، باید به عنوان یک نمودار وظیفه‌مندی(کارکردی)[28] طبقه‌بندی شود. اما همانطور که در ابتدای این گزارش و در تصویر ۰ مشخص است، جای این دسته از نمودارهای مدل‌سازی در UML خالی‌است که تصمیمی بوده است از سوی واضعان UML با هدف ساده‌سازی و برای مدت‌ها مورد بحث بوده است.



تصویر ۲.۱. نمودار های رفتاری برگرفته از (۱).

[27] Functionality
[28] Functional

در این بخش، قصد داریم به صورت اختصاصی به چهار مدل نمودارهای تعاملی بپردازیم. نمودارهای تعاملی برای مدل‌سازی تعاملات بین شی‌های زمان اجرا استفاده می‌شوند و نوعاً رفتار یک سیستم در یک سناریو را مجسم می‌سازند.

در چنین مواردی گزینش (سناریو) مهم می‌شود؛ جامعیت دشمن قابلیت درکِ مدل است. (۲)

یک نمودار تعاملی، با صرف نظر از نوع آن (توالی[29]، ارتباط/همکاری[30]، زمان‌بندی[31] و مرور تعاملات[32]) نمایانگر آن است که چگونه شی‌های مختلف سیستم با یکدیگر در یک رفتار خاص همکاری و با ارسال پیام تعامل می‌کنند، تا یک مورد کاربرد مشخص محقق شود. اگرچه، نمودارهای توالی و نمودارهای ارتباط نوعاً روش اصلی مدل‌سازی تحقق موردهای کاربرد هستند که در اینجا به تفصیل به آن‌ها پرداختیم. از آنجایی که نمودارهای زمان‌بندی به صورت عمده روی محدودیت‌های زمانی سیستم و نمودارهای مرور تعاملات روی ترتیب مجاز رخ دادن چند مورد کاربرد تاکید دارند، به اندازه دو نمودار پیشین در نمایش تحقق یک مورد کاربرد مفید واقع نمی‌شوند.

مقایسه از منظر مقصود

پر استفاده‌ترین نمودار تعاملی، **نمودار توالی** است. این نمودارها به عنوان پشتوانه‌ای برای نمودارهای کلاسی استفاده می‌شوند که مهمترین تعاملات شی‌های حاصل از آن کلاس‌ها را مدل‌سازی کنند و به همین ترتیب توان درک بیشتری برای مدل‌های سیستم فراهم می‌آورند. در متدلوژی‌های مورد کاربرد-رانه[33] مانند UP، که مدل‌سازی‌های دیگر از موارد کاربرد سیستم مشتق می‌شوند، این نمودار شی‌گرا حاصل یک «تغییر پارادیم[34]» از حوزه‌ی کارکردیِ نمودار مورد کاربرد است. این تغییر پارادایم با استفاده از نمودارهای فعالیت تحقق می‌یابد که در اینجا به اختصار درباره آن توضیح داده شده است.

نمودار ارتباط (که در UML 1 با نام نمودار همکاری شناخته می‌شد)، از نظر نشان دادن رفتار اشیای موجود در سیستم و ارتباطات آن‌ها از طریق ارسال پیام به منظور تحقق یک مورد کاربرد، در واقع یک هم‌ریخت (ایزومورف) برای نمودار توالی است. اگرچه، این نوع نمودار به جای تاکید روی صرف ارتباط و پیام‌های ارسالی، به روی لینک‌های داده‌ای موجود بین شی‌ها تمرکز دارد. از این رو، برای مدل‌سازی

[29] Sequence
[30] Communication/Collaboration
[31] Timing
[32] Interaction overview diagram
[33] Use case-driven
[34] Paradigm-shift

تعاملی یک سناریو، اگر تمرکز روی لینک‌ها است باید از نمودار ارتباط و اگر تمرکز روی ترتیب فراخوانی‌ها است باید از نمودار توالی استفاده شود.

یقیناً این تفاوت در معنا و مقصود این دو نمودار، در ساختار و نمادگذاری‌های هر کدام از آن‌ها منعکس شده است که در بخش «مقایسه ساختاری» به آن خواهیم پرداخت.

نمودار مرور تعاملات نتیجه ادغام نمودار فعالیت با نمودارهای تعاملی (علی‌الخصوص نمودار توالی) می‌باشد. از طرفی، نمودار فعالیت به نوبه خود، نمایانگر ترتیب مجاز کنش[35]های سیستم است؛ از طرف دیگر، نمودار مرور تعاملات نشانگر ترتیب مجاز رخداد موردهای کاربرد می‌باشد. استفاده آن‌ها نیز به همین ترتیب در مواردی است که می‌خواهیم یک دید کلی از رخداد چندین سناریو در کنار هم و ترتیب وقوع آن‌ها را مجسم سازیم.

> **من از آنجایی که واقعاً حس می‌کنم این دو سبک به خوبی با هم ترکیب نمی‌شوند، با آن‌ها راحت نیستم.**
>
> مارتین فولر درباره نمودارهای مرور تعاملات در (۲).

وقتی مدل‌سازی نیاز دارد که روی محدودیت‌های زمانی سیستم تمرکز داشته باشد، **نمودارهای زمان‌بندی** مورد استفاده قرار می‌گیرند. این دسته از نمودارها، مشابه نمودارهای تعاملی دیگر، یک سناریو خاص که در آن یک مورد کاربرد در حال اجراست را منعکس می‌کنند و زمانی استفاده می‌شوند که زمان پاسخگویی[36] سیستم شدیداً مهم باشد (به عنوان مثال، سیستم‌های بلادرنگ).

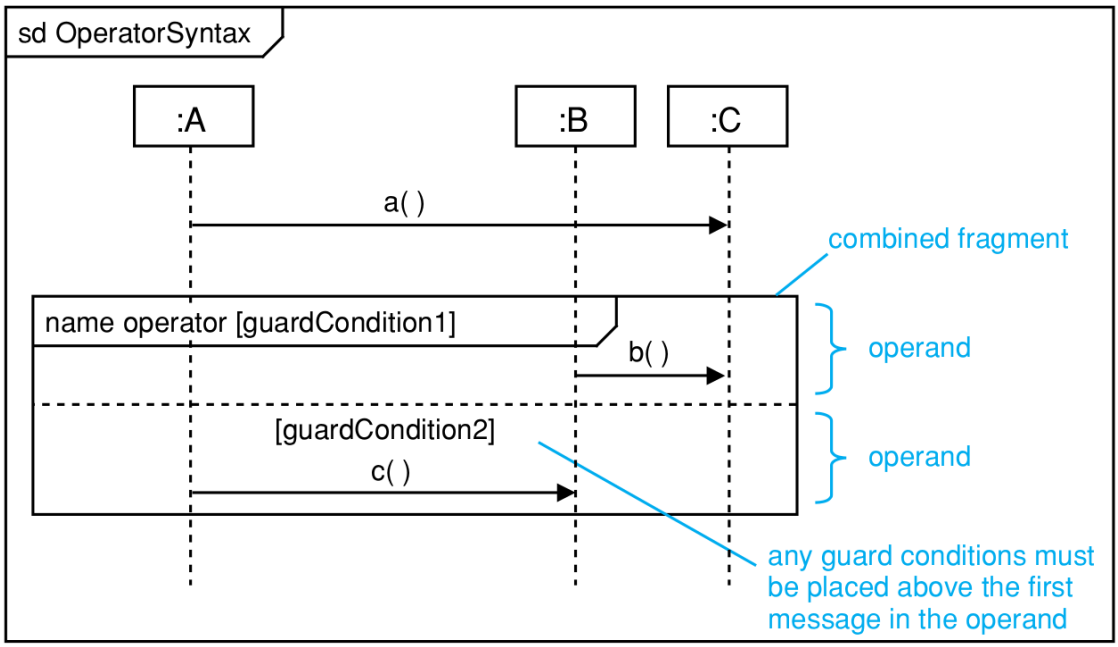
مقایسه ساختاری

نمودار توالی، اگرچه در نسخه‌های اولیه UML مبتدی بوده، یک نمادگذاری بسیار غنی در UML 2 پیدا کرده است. در نسخه دوم، هر شی شرکت‌کننده در تعامل را «خطِ عمر[37]» می‌نامیم. اشیا پس از ایجاد، دارای محور زمانی شده و این محور زمانی تا پایان عمرشان (که با × نشان داده می‌شود) ادامه می‌یابد. در نتیجه یک ارسال پیام، ممکن است خطِ عمری (یک شی) ساخته شود و اگر پیش از دریافت این پیام آن خط عمر زنده بوده است، آن خط عمر فعالسازی می‌شود که به معنای انجام فعالیتی می‌باشد.

[35] Action
[36] Response time
[37] Lifeline

از طرفی این محورهای زمانی در **نمودارهای ارتباط** وجود ندارند. پیام‌های رد و بدل شده میان شی‌ها و ترتیب ارسال آنها به صورت ضمنی در کنار لینک‌های داده‌ای بین اشیا مکتوب می‌شوند؛ به همین دلیل، درک ترتیب رویداد پیام‌ها در این مدل سخت تر از نمودار توالی است. این نمودار عیناً نماد نشان‌دهنده‌ی شی را از نمودار شی به وام گرفته و از همین رو شبیه یک نمودار شی می‌باشد که با نمایش رفتار تدقیق شده است.

شایان توجه است که نمودارهای توالی را می‌توان به نواحی‌ای با نام «قطعات ترکیبی[38]» تقسیم نمود. این مکانیزم جدید، مدل‌سازی رفتارهای پیچیده‌ای از قبیل هم‌روندی، حلقه و ... را تسهیل می‌کند. متاسفانه، نمودارهای ارتباط فاقد این مکانیزم غنی هستند. سینتکس عمومی این مکانیزم در تصویر ۲.۲ قرار گرفته است.



تصویر ۲.۲. سینتکس عمومی مکانیز قطعات ترکیبی.

همانطور که پیش‌تر در اینجا مطرح کردیم، وقتی که یک نگاه سطح بالا از چند مورد کاربرد و ترتیب تحقق آن‌ها به یکباره مد نظر است، از **نمودار مرور تعاملات** استفاده می‌شود. این دسته از نمودارها، ساختاری بسیار مشابه نمودارهای فعالیت دارند اگرچه بجای کنش‌ها، نمودارهای تعاملی دیگر در جایگاه گره‌ها قرار می‌گیرند.

[38] Combined fragments

همچنین با استفاده از نمودارهای مرور تعاملات دیگر بجای گره‌ها، این نمودار تنها مدلی در بین نمودارهای تعاملی خواهد بود که از ساختار تو در تو پشتیبانی می‌کند. البته این عمل تو در تو سازی به صورت صریح انجام نمی‌شود، بلکه باید به صورت ضمنی لینکی به یک نمودار مرور تعاملات دیگر در جایگاه گره‌ها ایجاد نمود. این موضوع موجب کنترل سطح پیچیدگی و خوانایی نمودار می‌شود.

لازم به ذکر است وجود انتقال پیام درون نمودارهای تعاملی موجود در هر گره است که باعث می‌شود این نمودار علاوه بر رفتاری بودن خود، تعاملی نیز محسوب شود.

نمودارهای زمانبندی، مدت‌ها در مهندسی برق مورد استفاده قرار می‌گرفتند. این نمودار، مشابه نمودارهای توالی و ارتباط، متشکل از خطِ عمرها (که مابه ازای اشیای شرکت‌کننده در تعامل هستند) و حالت‌های موجود برای هر کدام از آن‌هاست. محور زمان مدرجی که در زیر این نمودار رسم می‌شود - که نمایانگر محدودیت‌های زمانی موجود در سیستم است - عمده تفاوت ساختار این نمودار با نمودارهای توالی و ارتباط است. از طرف دیگر استفاده از نمودارهای توالی و ارتباط برای هر نوع از اشیایی که در یک تعامل مشارکت می‌کنند موضوعیت دارد اما این نوع نمودار ذاتاً برای خطوط عمری معنا پیدا می‌کنند که «دارای حالت[39]» باشند؛ در مورد حالت‌مندی اشیا، جلوتر در اینجا بیشتر می‌پردازیم.

[39] Stateful

۳. نمودار مورد کاربرد

نمودار مورد کاربرد را با هر یک از نمودارهای رفتاری زیر از منظر ساختار و مقصود مقایسه کنید.

- نمودار فعالیت[40]
- ماشین حالت

پیش‌تر در اینجا که بحث مقایسه نمودارهای تعاملی را باز کردیم، راجع به نمودارهای رفتاری UML توضیحات مقدماتی دادیم. در یک سیستم شی‌گرا، رفتار متناظر می‌شود با اینکه چگونه کارهای مختلف انجام می‌شوند. نمی‌شود از سیستم تنها با نمودارهای ایستا[41] (ساختاری) به درک عمیقی دست‌یافت؛ بلکه باید درباره نحوه همکاری ساختارهای مختلف و تعاملات آن‌ها با یکدیگر (به منظور انجام وظایف) اطلاعات داشت.

در این بخش قصد داریم دید خود را نسبت به سه مدل رفتاری UML عمیق‌تر کنیم: نمودار مورد کاربرد[42]، نمودار فعالیت، نمودار حالت.

مقایسه از منظر مقصود

نمودارهای مورد کاربرد برای مدت‌ها تنها نمودار وظیفه‌مندی مجاز در UML بوده‌اند و واضعان UML استفاده از هرگونه نمودار کارکردی دیگر (مانند نمودار جریان داده[43]) را برای مدل‌سازی وظیفه‌مندی سیستم‌های شی‌گرا شدیداً نکوهش کرده‌اند.

مورد کاربرد، یک فن برای مجسم‌سازی نیازهای کارکردی سیستم می‌باشد. این روش به ما کمک می‌کند چگونگی خدمت گرفتن کنشگران[44] مختلف سیستم از یک موضوع[45] (سرویس‌دهنده) را مدل‌سازی و تثبیت کنیم. (درباره نمادهای «موضوع» و «کنشگر» در اینجا بیشتر توضیح داده شده است).

یک موضوع قابل تامل درباره این نمودار، منجر شدن آن به درز پارادایمی بزرگی بین جهان شی‌گرای سیستم (و مدل‌سازی‌های مربوط به آن) و جهان کارکردی خود نمودار کاربرد است. از طرفی ما در

[40] Activity diagram

[41] Static

[42] همانطور که در اینجا اشاره کردیم، نمودار مورد کاربرد در ذات یک مدل وظیفه‌مندی است (و هیچ ربطی به رفتار ندارد)؛ با این حال در این گزارش ترجیح دادیم گاها خط مشی UML در این زمینه را دنبال کنیم.

[43] Data-flow diagram (DFD)

[44] Actors

[45] Subject

متدلوژی‌های نرم‌افزاری همواره در تلاشیم هیچ درز پارادایمی در منطق و پیاده‌سازی‌مان وجود نداشته باشد. این تغییر پارادایم با نمودار فعالیت ممکن می‌شود.

نمودار فعالیت یک فن برای توصیف منطق رویه‌ای، فرایند کسب‌وکار و جریان کاری است (۲). در مدل‌سازی شی‌گرا این نمودار مقصود مهمتری را برآورده می‌کند و آن تهیه روش آسانی برای طراحی نمودار توالی است که رفتار سیستم را در یک سناریو از تحقق یک مورد کاربرد نشان می‌دهد.

توجه داشته باشید که نمودار فعالیت نوعاً شی‌گرا نیست و تغییر پارادایم به شی‌گرایی با استفاده از ابزاری که به نوبه از المان‌های شی‌گرا پشتیبانی نمی‌کند چالش برانگیز خواهد بود. مکانیزم‌های مختلفی مانند swimlane در این چالش ما را یاری خواهند کرد که روش آن در اینجا به اختصار توضیح داده شده است.

در نهایت، **ماشین‌های حالت** یک فن قابل درک برای مدل‌سازی هر سیستم دارای حالت در هر سطحی از جزییات و انتزاع می‌باشند. حالت در حوزه طراحی سیستم‌ها، اشاره به وضعیتی دارد که کلیت رفتار سیستم در آن قابل توضیح و پیش‌بینی است؛ همچنین بعضاً می‌توان حالات آینده سیستم را با بررسی حالت فعلی یا توالی حالت‌های گذرانده شده توسط آن پیش‌بینی کرد. با چنین تعریفی، هر سیستم الزاماً دارای حالت نیست و به آن دسته دیگر سیستم‌های بی‌حالت[46] می‌گویند. این نمودار اساساً برای نمایش حالات و انتقال بین آن‌ها در یک سیستم رویداد-رانه[47] استفاده می‌شود؛ اما ما از آن برای مشخص کردن چرخه حیات شی‌های دارای حالت استفاده می‌کنیم. به نمودارهای ماشین حالتی که با چنین هدفی ساخته می‌شوند، نمودار چرخه‌ی حیات شی[48] گفته می‌شود.

شایان توجه است که بر خلاف نمودارهای فعالیت، توالی و ارتباط - که تنها یک سناریو در حال وقوع در سیستم را مجسم می‌سازند - این نمودار از نگاه شی، متناظر با چندین مورد کاربرد (و نتیجتاً چندین سناریوی محقق‌کننده هر مورد کاربرد) است چرا که هر شی در چرخه عمر خود می‌تواند در تحقق چندین مورد کاربرد ایفای نقش کند.

مقایسه ساختاری

نمودار مورد کاربرد به اندازه نیاز نمادهای مختلف دارد. چند نماد مهم از این ساختار شامل موضوع، مورد کاربرد و کنشگر می‌باشد.

[46] Stateless
[47] Event-driven
[48] Object lifecycle diagram (OLD)

موضوع غالباً به سیستمی اشاره می‌کند که کارکردهایی را برای کاربران مختلف انجام می‌دهد. این واژه در UML ۲ مطرح شد (سابقاً به آن مرز سیستم[49] گفته می‌شد). هدف این تغییر لفظ، تعمیم معنای این نماد به هر ماهیتی که بتواند موجب تحقق مورد کاربرد شود بوده است. به همین سبب، موضوع نمودار مورد کاربرد در UML 2 حتی یک کلاس سیستم می‌تواند باشد.

هر مورد کاربرد داخل موضوع، تجزیه‌ناپذیر بوده و با یک نیازمندی کارکردی منفرد از سیستم متناظر می‌شود؛ در نتیجه، نمی‌توان آن را به زیر-موردهای دیگری تجزیه کرد. به علاوه، هر مورد کاربرد الزاماً باید توسط یک کنشگر راه بیفتد (وگرنه نمایش آن در مدل موضوعیت ندارد).

کنشگران سیستم نقش‌های مختلفی هستند که ممکن است توسط کاربران یا حتی سیستم‌های دیگر بازی شوند. آن‌ها یک خدمتی را از موضوعِ نمودار درخواست می‌کنند.

اگرچه موردهای کاربرد تجزیه‌ناپذیرند، اما آن‌ها گام‌هایی برای تحقق دارند که محتوایشان را توضیح می‌دهند. بر این اساس، به همراه این نمودارها عموماً یک متن توصیفی به نام «جریانِ رویدادها[50]» تعبیه می‌شود. یک مثال ساده از جریان رویدادهای یک مورد کاربرد از (۱) در تصویر ۳.۱ قرار گرفته است.

[49] System boundary

[50] Flow of events

Use case: ManageBasket
ID: 2
Brief description: The Customer changes the quantity of an item in the basket.
Primary actors: Customer
Secondary actors: None.
Preconditions: 1. The shopping basket contents are visible.
Main flow: 1. The use case starts when the Customer selects an item in the basket. 2. If the Customer selects "delete item" 2.1 The system removes the item from the basket. 3. If the Customer types in a new quantity 3.1 The system updates the quantity of the item in the basket.
Postconditions: None.
Alternative flows: None.

تصویر ۳.۱. یک مثال چکیده از جریان رویدادهای مورد کاربرد مدیریت سبد خرید (Manage Basket) از (۱). برای هر مورد کاربرد تعدادی کنشگر اولیه مورد کاربرد را راه اندازی می‌کنند. ممکن است تعدادی کنشگر ثانویه نیز در مسیر تحقق مورد کاربرد درگیر شوند که در این متن لیست می‌شوند. همچنین در این متن می‌توان پیش‌شرط‌ها و پس‌شرط‌ها را به نمایش گذاشت.

نمودار مورد کاربرد از منظر سناریوهای مختلف سیستم، ساختاری تدقیق نشده دارد که به آن کمک می‌کند یک نمایش[51] عمومی از سیستم را مجسم سازد. در واقع این نمایش برای تمامی سناریوهای مختلفی که موضوعِ نمودار تجربه می‌کند برقرار است. برای مثال به مورد کاربردِ «ورود به سامانه» توجه کنید؛ سناریوهای مختلفی برای همین یک مورد کاربرد متصور است مانند: «رمز عبور اشتباه»، «نام‌کاربری اشتباه»، «ورودِ موفقیت‌آمیز». با صرف نظر از سناریوهای ممکن، مورد کاربرد مذکور در نمودار مربوط به این سامانه حضور خواهد داشت.

نمودارهای فعالیت عمدتاً متشکل از کنش‌ها و جریان ها هستند. هر کنش یک کار تجزیه‌ناپذیر است که توسط کنشگرهای سیستم انجام می‌شوند. این کنشگرها با همین المان از نمودارهای مورد کاربرد نظیر می‌شوند و در این نمودار با نام swimlane شناخته می‌شوند. بعد از انجام هر کنش، یک

[51] Representation

جریان خروجی، قرائت‌کننده نمودار را به کنش بعدی (یا دیگر نمادها مانند شاخه) هدایت می‌کند. این کنش‌ها به همراه جریان‌های خروجی‌شان (که ترتیب عملیات‌ها را نشان می‌دهد) در مجموع یک فعالیت را می‌سازند. فعالیت‌ها به کنش‌های تشکیل دهنده تجزیه‌پذیرند و از ساختار تو در تو پشتیبانی می‌کنند (فعالیت‌های درونی یک نمودار فعالیت با نماد شن‌کش[52] نشان داده می‌شوند). این موضوع در تقابل با ساختار نمودار مورد کاربرد است، اگرچه موارد کاربرد ذاتاً نیازمند ساختار تو در تو نیستند.

در متدلوژی‌های مورد کاربرد-رانه مانند UP، بسیاری از نمودارهای دیگر از نمودار مورد کاربرد مشتق می‌شوند. پس ما در اولین گام این نمودار را به همراه جریان رویدادهای آن (که متن توصیف‌کننده‌ی گام‌ها و محتوای آن است) تهیه می‌کنیم.

سپس جزییات گام‌ها و توضیحات جریان رویدادهای هر مورد کاربرد را استخراج کرده و در ذیل سناریوهای مختلف به صورت نمودار فعالیت طراحی می‌کنیم. کنشگرهای راه‌انداز آن مورد کاربرد که در نمودار مورد کاربرد مربوطه حضور داشتند، در این نمودار فعالیت نیز به صورت swimlane های آغازکننده فعالیت منعکس می‌شوند. همچنین کنشگران ثانویه ذکر شده در جریان رویدادهای مورد کاربرد، به عنوان swimlane های دیگری که جریان کاری وارد آن‌ها می‌شود حضور دارند.

در گام آخر هر کدام از این swimlane ها را به swimlane های ریزتر می‌شکنیم تا در نهایت به کلاس‌های تعریف شده در مدل ایستای خود برسیم. در هر کدام از این swimlane های ریز شده شی‌های زمان اجرا از آن کلاس وجود دارند. جریان‌های نمودار فعالیت جای خود را با پیام‌های بین اشیا عوض می‌کنند. در نتیجه، چندین نمودار توالی که با ارسال پیام به یکدیگر متصل‌اند به دست می‌آید. در آخر، این نمودارهای توالی، مورد کاربرد اولیه را به یک شکل شی‌گرا محقق می‌سازند.

برای تاکید بر رفتارهای تنها یک شیِ دارای حالت، از ماشین حالت استفاده می‌شود. این نمودارها در نمادگذاری آن قدر غنی هستند که یک فصل کامل از کتاب (۱) به توضیح استفاده پیشرفته از این نمودار تخصیص یافته است (فصل ۲۲، ماشین حالت پیشرفته) اما یک ورژن چکیده‌تر آن شامل حالات شی، کمان‌های انتقال بین آن حالات و توضیح هر انتقال می‌باشد.

هر انتقال غالباً شامل موارد زیر می‌شود؛

- یک رویداد که آن را فعال می‌کند.
- یک شرط نگهبان[53] که باید ارضا شود تا انتقال صورت پذیرد.
- یک کنش که همزمان با انتقال انجام می‌شود.

[52] Rake
[53] Condition guard

با این وجود، این خصوصیات اختیاری بوده و در صورت عدم موضوعیت هر ویژگی برای یک انتقال، قاعدتاً می‌توان از ذکر آن صرف نظر کرد. برای مثال انتقالی را در نظر بگیرید که در صورت رخداد یک واقعه، بدون هیچ شرطی همواره صورت پذیرد؛ در این مورد نیازی به ذکر شرط نگهبان نیست.

از سوی دیگر، درون نماد مربوط به «حالت» می‌توان فعالیت در حال انجام در آن حالت را ذکر نمود. شایان ذکر است که این فعالیت‌ها می‌توانند با یک نمودار فعالیت و توالی که نشانگر رفتار آن شی در آن حالت است، حمایت شوند. همچنین در مدل‌سازی پیشرفته‌تر نمودار حالت، مانند نمودار فعالیت، می‌توان ساختارهای تو در تو داشت (ماشین حالت درون ماشین حالت).

References

1. Arlow, J., Neustadt, I., UML 2 and the Unified Process: Practical Object-Oriented Analysis and Design, 2nd Edition.

2. Fowler, M., UML Distilled, 3rd Edition.

3. Ramsin, R., Paige, R. F., Process-Centered Review of Object Oriented Software Development Methodologies.

4. Martin, R. C., Clean Architecture: A Craftsman's Guide to Software Structure and Design.