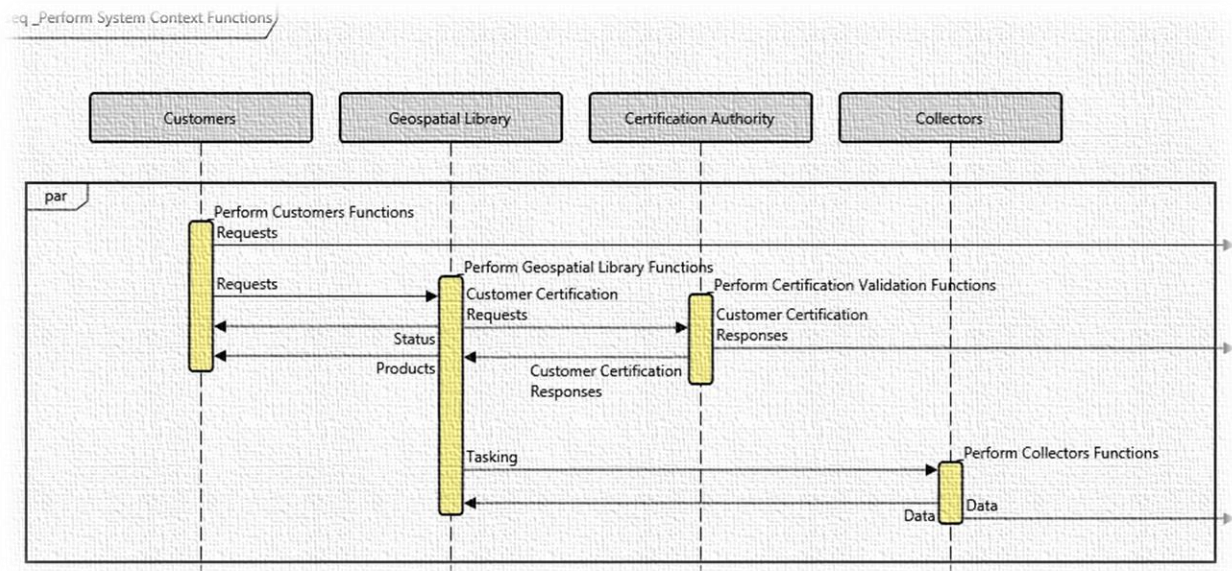


نمودار توالی یا Sequence diagram در UML



تهیه و تنظیم: پیمان مالکی



فهرست مطالب

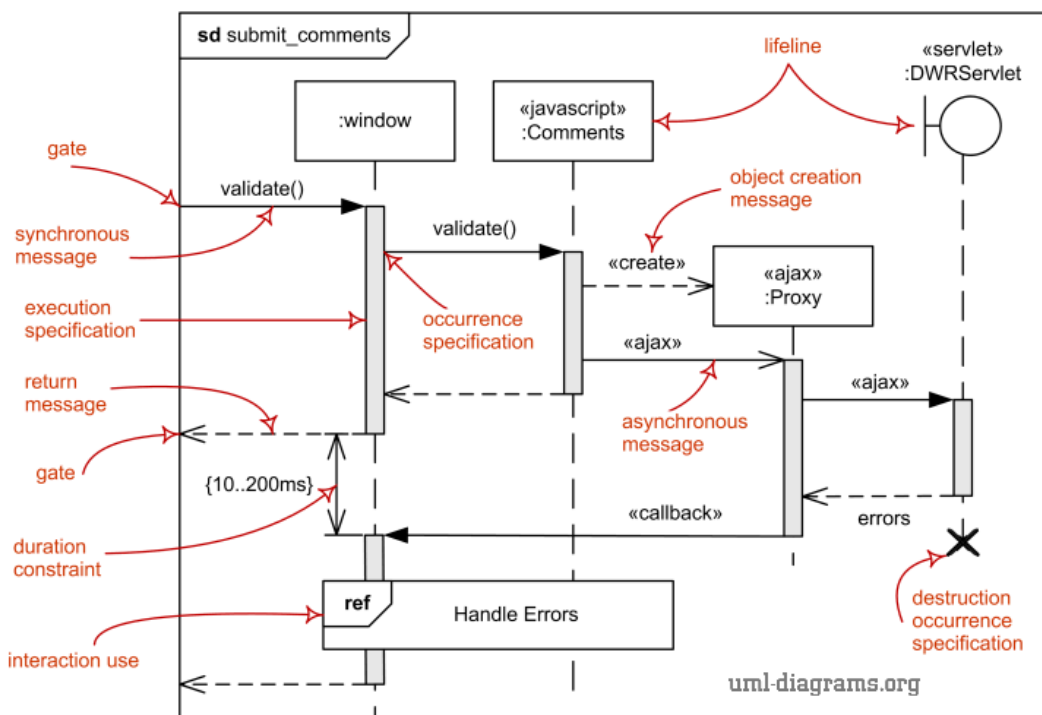
3	مقدمه
3	چه زمانی و به چه دلایلی تهیه این نمودار اهمیت خواهد داشت؟
5	Lifeline
7	Gate
8	Interaction Fragment
9	Occurrence
9	message occurrence
9	Destruction Occurrence
9	execution occurrence
10	execution
11	State Invariant
12	Interaction Use
14	Message
14	پیامها بر اساس نوع اقدام
14	Synchronous Call
15	asynchronous call
15	asynchronous signal
15	Create Message
16	Delete Message
16	Reply Message
16	پیامها بر اساس رویدادها
17	complete message
17	lost message
17	found message
17	unknown message
18	Combined Fragment
18	محدودیت‌های تعامل یا interaction constraint
18	alternatives :alt
19	option :opt
19	iteration :loop

20	break :break
21	parallel :par
22	strict sequencing :strict
22	weak sequencing :seq
22	critical region :critical
23	ignore :ignore
23	consider :consider
24	assertion :assert
24	negative :neg
25	خلاصه و چکیده
32	لغت نامه تصویری
32	Lifeline:
32	Message:
35	Lifeline Occurrence:
35	Message Occurrence:
35	Gate:
36	Trace:
38	Behavier Unit:
38	Interaction:
38	Interaction Fragment:
40	Combined Fragment:

نمودار توالی یا Sequence diagram رایج ترین نوع نمودار تعاملی است که بر تبادل پیام بین تعدادی از Lifeline ها متمرکز است. نمودار توالی یک تعامل را با تمرکز بر توالی پیام‌هایی که رد و بدل می‌شوند، همراه با مشخصات رویداد متناظر آنها در Lifeline ها توصیف می‌کند. گره‌ها و لبه‌های زیر معمولاً در نمودار توالی UML ترسیم می‌شوند:

- Lifeline
- execution specification
- message
- combined fragment
- interaction use
- state invariant
- continuation
- destruction occurrence

عناصر اصلی نمودار توالی در تصویر زیر نشان داده شده است.



چه زمانی و به چه دلایلی تهیه این نمودار اهمیت خواهد داشت؟

نمودار توالی نوعی از نمودار تعاملی-رفتاری است زیرا توضیح می‌دهد که چگونه و به چه ترتیبی، گروهی از اشیاء با هم کار می‌کنند. این نمودارها توسط توسعه دهندگان نرم‌افزار و یا متخصصان کسب‌وکار برای درک الزامات یک سیستم جدید یا مستندسازی یک فرآیند موجود استفاده می‌شود.

هنگامی که می‌خواهید به رفتار اشیاء در یک Use Case نگاه کنید، باید از نمودارهای توالی استفاده کنید. این نمودارها در نشان دادن همکاری بین اشیاء، خیلی کمک می‌کنند. اما این نمودارها فقط تعریف سطحی از رفتارهای اشیاء نشان می‌دهند و در تعریف دقیق رفتار آنها چندان مفید نیستند.

کارشناسان فنی یک سازمان ممکن است تهیه نمودارهای توالی را برای کشف یا نمایش نحوه رفتار یک سیستم مورد انتظار، مفید بدانند. در طول مرحله طراحی، این نمودارها می‌توانند ارتباطی فراگیر بین معماران و توسعه‌دهندگان بوجود آورند و طراحی کلی سیستم را کامل کنند.

Lifeline یک عنصر نامگذاری شده است که نشان دهنده یک مشارکت کننده منحصر به فرد در تعامل است. در حالی که بخش‌ها و ویژگی‌های ساختاری ممکن است منفرد نباشند، **Lifeline** ها تنها یک موجودیت تاثیرگذار را نشان می‌دهند. اگر عنصر مرتبط مورد نظر که به آن ارجاع شده است یک عنصر چند بخشی باشد (به عنوان مثال دارای یک $\text{multiplicity} < 1$) باشد، ممکن است **Lifeline** با یک عبارت **selector** یا همان (:) نشان داده شود که مشخص می‌کند کدام بخش خاص توسط این **Lifeline** نمایش داده می‌شود. اگر عبارت **selector** حذف شود، به این معنی است که عنصر مورد نظر می‌تواند یک جایگزین دلخواه از عنصر مرتبط چند بخشی را شامل شود. یک **Lifeline** با استفاده از یک نماد نشان داده می‌شود که شامل یک مستطیل است که "سر" آن را تشکیل می‌دهد و به دنبال آن یک خط عمودی (که ممکن است خط چین دار باشد) که نشان دهنده طول عمر مشارکت کننده است. اطلاعاتی که **Lifeline** را شناسایی می‌کند در الگوی زیر در مستطیل نمایش داده می‌شود:

lifeline-ident ::= [connectable-element-name ['[' selector ']']] [':' class-name] [decomposition] | 'self'

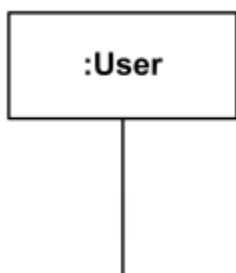
selector::= expression

decomposition ::= 'ref' interaction-ident ['strict']

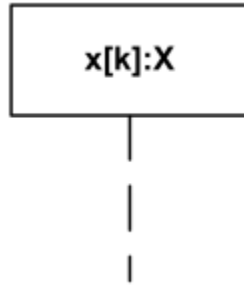
که در آن **class-name** نوعی است که توسط عنصر مرتبط ارائه شده، اشاره شده است. توجه داشته باشید که اگرچه قواعد نگارشی ذکر شده اجازه می‌دهد، **lifeline-ident** نمی‌تواند خالی باشد. سر **Lifeline** به شکلی ترسیم می‌شود که باعث طبقه بندی بخشی در زیر خود است که این خط زمانی را نشان می‌دهد. معمولاً این بخش مستطیل شکل سفید است که حاوی نام کلاس خواهد بود.



این **Lifeline** معرف بخش **data** از کلاس **Stock** است



معرف هر کاربری از کلاس **User** است



معرف عنصری به نام `x` که با پارامتر `[k]` از کلاس `X` انتخاب شده است

اگر نام کلمه کلیدی `self` باشد، Lifeline نشان دهنده شیء طبقه‌بندی کننده‌ای است که تعاملی را که مالک Lifeline است را شامل می‌شود.

Gate انتهای یک پیام است، نقطه اتصال داخلی یک قطعه تعاملی را با یک پیام خارج Gate می‌گویند. هدف از Gate ها و پیام‌های بین آنها، مشخص کردن فرستنده و گیرنده هر پیام است. Gate ها نقش‌های مختلفی را ایفا می‌کند:

- Gate های رسمی یا formal gates - در تعاملات یا interactions استفاده می‌شوند
- Gate های واقعی یا actual gates - در تعاملات کاربردی یا interaction uses استفاده می‌شوند
- Gate های دستوری یا expression gates - در combined fragment استفاده می‌شوند

Gate ها به طور ضمنی یا صریح نامگذاری می‌شوند. نام Gate ضمنی با افزودن جهت پیام ("in" یا "out") و نام پیام، ساخته می‌شود. به عنوان مثال، in_search که جهت آن از بیرون به درون است و پیام جستجو را شامل می‌شود، out_read که جهت آن از درون به بیرون است و پیام خوانده شدن را شامل می‌شود.

Gate ها دقیقاً به عنوان نقاط اتصال پیام به قاب یک fragment معرفی می‌شوند.

قطعه تعاملی یا Interaction Fragment یک عنصر نامگذاری شده است که عمومی ترین واحد تعامل را نشان می‌دهد. هر قطعه تعاملی از نظر مفهومی مانند یک تعامل مستقل است. هیچ علامت کلی برای یک قطعه تعاملی وجود ندارد. زیر کلاس‌های آن، به صورت باواسطه یا بی‌واسطه نماد آن یا نام آن را تعریف می‌کنند. نمونه‌هایی از قطعات تعاملی عبارتند از:

- رخداد یا occurrence
- اجرا یا execution
- وضعیت غیرقابل تغییر یا state invariant
- قطعه ترکیبی یا combined fragment
- تعاملات کاربردی یا interaction use

تعامل یا Interaction در UML چه معنایی دارد؟

یک تعامل، مشخص کننده رفتار و قطعه تعاملی است که نشان دهنده یک واحد رفتار است. این واحد رفتاری بر مبادله قابل مشاهده اطلاعات بین عناصر قابل اتصال، متمرکز است. تعامل، یک رفتار ضروری است. تعاملات بر انتقال اطلاعات با پیام‌ها بین عناصر قابل اتصال طبقه‌بندی کننده، متمرکز هستند.

یک تعامل در UML به شکل مجموعه‌ای از ردیابی‌ها معرفی می‌شود. این مجموعه‌ها در دو دسته، ردیابی معتبر و ردیابی نامعتبر تعریف می‌شوند. ردیابی‌هایی که در این دو مجموعه گنجانده نشده‌اند و معتبر یا نامعتبر بودن آنها معلوم نیست با تعامل توصیف نمی‌شوند. ردیابی، دنباله‌ای از رخدادهای حادث شده است که به شکل $\langle e1, e2, \dots, en \rangle$ نشان داده می‌شود و هر کدام با یک مشخصه رخداد مثل $e1$ ، توصیف می‌شوند.

هر رخداد تعامل، معمولاً با طول زمان صفر، تفسیر می‌شود. همیشه فرض بر این است که مدت زمان وقوع یک رخداد، از قبل اندازه گیری شده است و نیاز به محاسبه مجدد ندارد.

در زمان بازنگری، مانند یک رفتار، یک تعامل نیز قابل ویژه‌سازی و تعریف مجدد است. ویژه‌سازی یک تعامل، اجازه می‌دهد تا ردیابی‌های بیشتری را به ردیابی اصلی اضافه کنید. ردیابی که به واسطه ویژه سازی تعریف شده، با ردیابی‌های قدیمی‌تر در یک تعامل با هم مجتمع می‌شوند. طبقه بندی کننده ای که دارای یک تعامل است ممکن است ویژه‌سازی شده باشد و در ویژه‌سازی، ممکن است تعامل دوباره تعریف شود. تعریف مجدد یک تعامل به معنای جایگزینی تعامل بازتعریف شده با تعامل تعریف شده جدید است.

نماد یک تعامل یک قاب مستطیلی یکپارچه است. پنج ضلعی در گوشه سمت چپ بالای مستطیل قرار می‌گیرد، حاوی کلمه کلیدی sd به دنبال نام تعامل و پارامترها است. از نماد عمومی برای نام رفتار استفاده می‌شود. مدل سازی درون قاب مستطیلی (تعامل) به یکی از اشکال زیر صورت می‌گیرد:

- sequence diagram
- communication diagram
- timing diagram
- interaction overview diagram

نمودار نمای کلی تعامل، ممکن است شامل فهرستی از Lifeline ها از طریق یک Lifeline جزء باشد. فهرست Lifeline، فهرستی از Lifeline ها است که در تعامل هستند. یک نمودار نمای کلی تعامل به خودی خود Lifeline های درگیر را نشان نمی‌دهد، حتی اگر Lifeline های محتمل به صراحت در تعاملات درون خطی در گره‌های نمودار حضور داشته باشند.

یک نمودار تعامل همچنین ممکن است شامل معرفی ویژگی‌های محلی با نگارش مشابه با ویژگی‌های نشان داده شده در بخش‌های کلاس باشد. این معرفی‌ها ممکن است در نزدیکی بالای کادر نمودار یا در نمادهای یادداشت در سایر مکان‌های نمودار ظاهر شوند.

رخداد که نام کامل آن مشخصات رخداد یا **occurrence specification** می‌باشد، (به عنوان مثال "Event" شرح) قطعه تعاملی است که یک لحظه در زمان (واقعه) را در ابتدا یا انتهای یک پیام یا در ابتدا یا انتهای یک اجرا نشان می‌دهد. مشخصه رخداد یکی از واحدهای معنایی اصلی تعاملات است. توصیف فعل و انفعالات با توالی رخدادهایی که با مشخصات رخداد توصیف می‌شوند بیان می‌شوند. هر مشخصه رخداد دقیقاً در یک **Lifeline** ظاهر می‌شود. مشخصات رخداد دستورات موجود در امتداد **Lifeline** هستند. مشخصه رخداد هیچ علامتی ندارد و فقط یک نقطه در ابتدا یا انتهای یک پیام یا در ابتدا یا انتهای یک مشخصات اجرایی است.

نمونه‌هایی از رخداد عبارتند از:

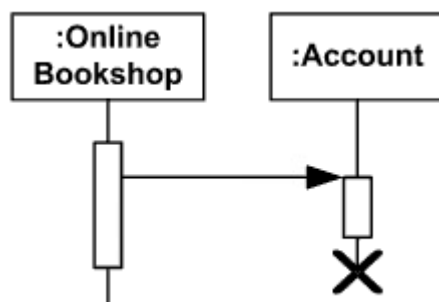
- رویداد پیام یا **message occurrence**
- رویداد اجرا یا **execution occurrence**

message occurrence

message occurrence که نام کامل آن مشخصات رویداد پیام یا **message occurrence specification** است، رخدادی است که نشان دهنده رویدادهایی مانند ارسال و دریافت سیگنال یا فراخوانی یا دریافت درخواست اجرای عملیاتی است.

Destruction Occurrence

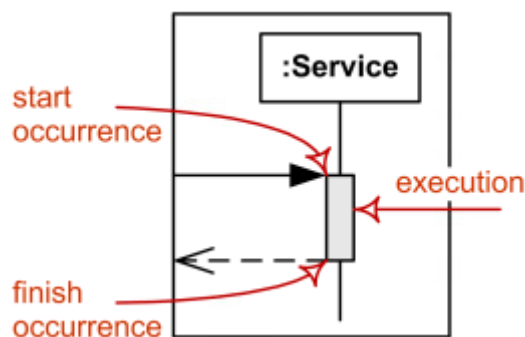
رویداد جمع‌آوری یک رخداد پیام است که نشان دهنده جمع‌آوری نمونه توصیف شده توسط **Lifeline** است. ممکن است منجر به جمع‌آوری بعدی اشیاء دیگری شود که این شیء متعلق به ترکیب آنها است. هیچ رویداد دیگری در زیر رویداد جمع‌آوری در یک **Lifeline** ظاهر نمی‌شود. نام کامل این رخداد، **destruction occurrence specification** است. تا قبل از **UML 2.4** به آن رویداد جمع‌آوری و قبل از آن - توقف نامیده می‌شد. جمع‌آوری نمونه، با یک علامت صلیب به شکل X در پایین یک **Lifeline** به تصویر کشیده شده است.



در تصویر بالا **Lifeline** به نام **Account** خاتمه یافته است

execution occurrence

رخداد اجرا که نام کامل آن مشخصات وقوع اجرا یا **execution occurrence specification** است، رخدادی است که لحظه‌ای از زمان را نشان می‌دهد که در آن اقدامات یا رفتارهای مشخصی، شروع یا پایان می‌یابند. رویداد اجرا، دقیقاً به یک مشخصه اجرایی یا **execution specification** اشاره می‌کند که اجرایی را که در این رویداد اجرا شروع شده یا به پایان می‌رسد را توصیف می‌کند.



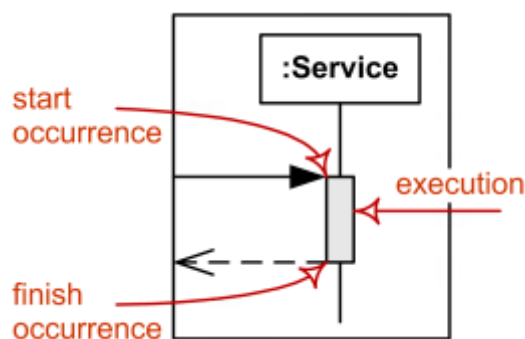
مدت زمان اجرا با دو رخداد اجرا نشان داده می‌شود - شروع و پایان.

execution

اجرا که نام کامل آن، مشخصات اجرا یا execution specification است و برخی مواقع، فعالیت یا activation نامیده می‌شود، قطعه تعاملی است که نشان دهنده یک دوره در طول عمر مشارکت کننده است که در آن ممکن است یکی از حالت زیر صادق باشد:

- اجرای یک واحد رفتاری یا عملیاتی در Lifeline
- ارسال سیگنال به مشارکت کننده دیگر
- منتظر پیام پاسخ از سوی مشارکت کننده دیگر

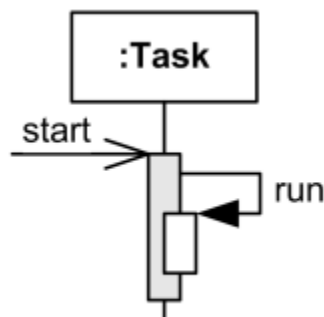
توجه داشته باشید که مشخصات اجرا، شامل مواردی است که معرف یک رفتار فعال نیست و فقط منتظر پاسخ است. مدت زمان اجرا در یک مشخصه اجرا، به فاصله زمانی بین دو رخداد اجرای شروع و پایان، بستگی دارد. اجرا به صورت یک مستطیل نازک خاکستری یا سفید روی Lifeline نشان داده می‌شود.



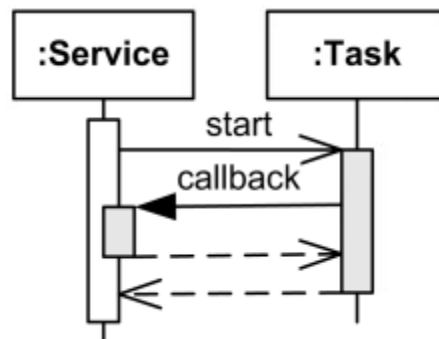
مشخصات اجرا را می‌توان با یک مستطیل پهن‌تر همراه با یک با برچسب نشان داد. در این حالت می‌توانید عملی را که اجرا شده است به کمک برچسب مورد نظر، مشخص کنید.



برای مشخصات اجرایی که به کنش‌های اتمیک مانند خواندن ویژگی‌های یک سیگنال (که توسط پیام منتقل می‌شود)، اشاره دارد، نماد عمل ممکن است با مشخصات رویداد دریافت با یک خط مرتبط شود تا تأکید شود که کل عمل فقط با یک رخداد مرتبط است. مشخصات (و پیوندهای شروع و پایان به مشخصات رویداد یکسانی اشاره دارد). مشخصات اجرای همپوشانی در همان Lifeline با مستطیل‌های همپوشانی نشان داده می‌شود.



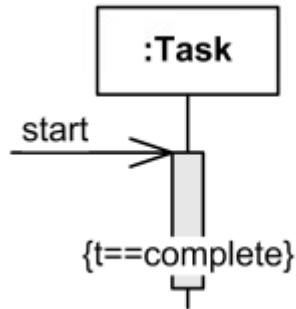
همپوشانی مشخصات اجرایی در همان Lifeline - پیام به خود یا Self Message



همپوشانی مشخصات اجرایی در همان Lifeline - پیام برگشت پیام یا Callback Message

State Invariant

وضعیت ثابت، یک قطعه تعاملی است که یک محدودیت زمان اجرا را برای مشارکت کنندگان تعامل نشان می‌دهد. وضعیت‌های ثابت ممکن است برای تعیین انواع مختلفی از محدودیت‌ها، مانند مقادیر ویژگی‌ها یا متغیرها، حالت‌های داخلی یا خارجی و غیره استفاده شود. محدودیت‌ها، بلافاصله قبل از اجرای مشخصه رویداد بعدی ارزیابی می‌شوند، به طوری که تمام اقداماتی که به طور صریح مدل‌سازی نشده‌اند، اجرا شده‌اند. اگر ارزیابی مورد نظر نشان دهد که قاعده درست است، ردیابی یک ردیابی معتبر است، در غیر این صورت ردیابی یک ردیابی نامعتبر است. حالت ثابت معمولاً به شکل براکت‌های باز و بسته بر روی Lifeline نشان داده می‌شود.



این تصویر نشان می‌دهد که ویژگی t مربوط به کلاس Task باید برابر با complete باشد.

همچنین می‌تواند به عنوان یک نماد حالت نشان داده شود که معادل یک محدودیت را نشان می‌دهد که وضعیت شیء نشان داده شده توسط Lifeline را بررسی می‌کند.



این تصویر نشان می‌دهد که Task باید در حالت Finished باشد

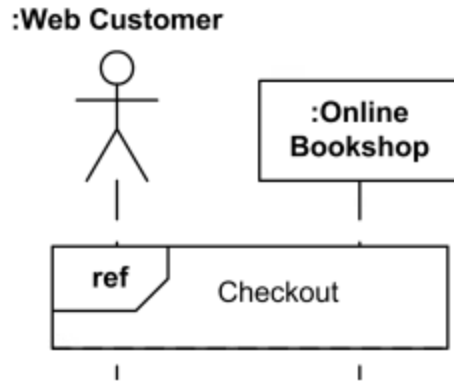
وضعیت ثابت همچنین می‌تواند به صورت اختیاری به عنوان یک یادداشت مرتبط با یک مشخصه رویداد نشان داده شود.

Interaction Use

تعامل کاربردی، یک قطعه تعاملی است که امکان استفاده (یا فراخوانی) تعامل دیگری را فراهم می‌کند. نمودارهای توالی بزرگ و پیچیده را می‌توان با استفاده از قطعات تعاملی، ساده کرد. همچنین استفاده مجدد از برخی تعاملات بین چندین تعامل دیگر معمول است. تعامل مرجع دارای Gate رسمی است. تعامل کاربردی، مجموعه‌ای از Gate های واقعی را ارائه می‌دهد که باید با Gate های رسمی، مطابقت داشته باشد. استفاده از قطعه تعاملی به صورت زیر عمل می‌کند:

- محتویات تعامل ارجاع شده را در جایی کپی می‌کنید که این تعامل باید استفاده شود
- پارامترهای رسمی را با آرگومان‌ها، جایگزین می‌کند
- Gate های رسمی را به Gate های واقعی وصل کنید

در تصویر زیر استفاده از قطعه تعاملی به عنوان یک قطعه ترکیبی با عملگر ref نشان داده شده است.



شیء Web Customer و Online Bookshop برای انجام عمل پرداخت یا همان Checkout از مرجع یا قطعه تعاملی ref استفاده می کنند

نحو نگارش interaction use به نام ref به صورت زیر است:

interaction-use ::= [attribute-name '='] [collaboration-use '.'] interaction-name [io-arguments] [':' return-value]

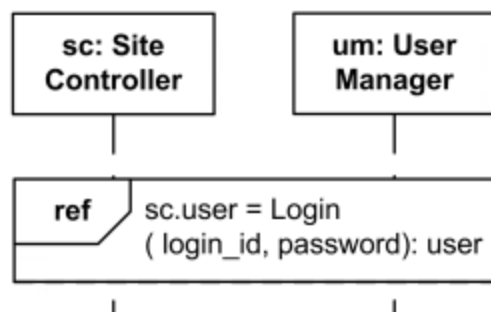
io-arguments ::= '(' io-argument [',' io-argument]* ')'

io-argument ::= in-argument | 'out' out-argument

attribute-name به ویژگی یکی از Lifeline ها در تعامل اشاره دارد که نتیجه تعامل را دریافت می کند. توجه داشته باشید که این امر باعث می شود، نتایج تعاملات فقط به ویژگی ها اختصاص داده شود. در زندگی واقعی، نتایج فراخوانی متد را می توان به متغیری از متد فراخوانی اختصاص داد.

collaboration-use، ویژگی استفاده شده در یک همکاری است که Lifeline های یک همکاری را به هم پیوند می دهد. -interaction-name در این شرایط، نام همکاری است.

io-arguments فهرستی از آرگومان های ورودی و/یا خروجی تعامل است.



از قطعه تعاملی ورود به سیستم برای احراز هویت کاربر استفاده می شود و نتیجه به مشخصه کاربر کنترلر سایت بازگردانده می شود

یک محدودیت اجباری که گاهی اوقات رعایت آن دشوار است این است که تعامل کاربردی باید تمام Lifelin های درگیر ارائه شده در محدوده تعامل را پوشش دهد. این بدان معنی است که همه آن Lifeline ها باید به نحوی در نزدیکی یکدیگر قرار گیرند. اگر تعامل کاربردی دیگری در همان نمودار داشته باشیم، تنظیم مجدد همه Lifeline های درگیر، همانطور که توسط UML مورد نیاز است می تواند بسیار مشکل باشد.

پیام یک عنصر نامگذاری شده است که یک نوع ارتباط خاص را بین Lifeline یک تعامل تعریف می‌کند. پیام نه تنها نوع ارتباط، بلکه فرستنده و گیرنده را نیز مشخص می‌کند. فرستنده و گیرنده معمولاً دو مشخصه رویداد (نقاط انتهای پیام‌ها) هستند.

نگارش پیام به شکل زیر است:

```
message ::= [ attribute '=' ] signal-or-operation-name [ arguments ] [ ':' return-value ] | '*'
```

```
arguments ::= '(' [ argument [ ',' argument ]* ] ')'
```

```
argument ::= [ parameter-name '=' ] argument-value | attribute '=' out-parameter-name [ ':' argument-value ] | '-'
```

آرگومان‌های یک پیام فقط می‌توانند یکی از مقادیر زیر باشد:

- ویژگی‌های Lifeline ارسال کننده
- ثابت‌ها
- مقادیر نمادین (که مقادیر عام هستند که هر مقدار قانونی را نشان می‌دهند)
- پارامترهای صریح تعامل محصور کننده
- ویژگی‌های کلاس صاحب تعامل

یک پیام به صورت خطی است که از یک طرف به فرستنده پیام و از طرف دیگر به گیرنده پیام منتهی می‌شود. جهت ترسیم این خط باید به گونه‌ای باشد که همواره هر خط هنگام حرکت از رویداد ارسال به رویداد دریافت، افقی یا رو به پایین باشد. رویدادهای ارسال و دریافت، ممکن است هر دو در یک Lifeline باشند. شکل خط یا نوک پیکان ویژگی‌های پیام را مشخص می‌کند.

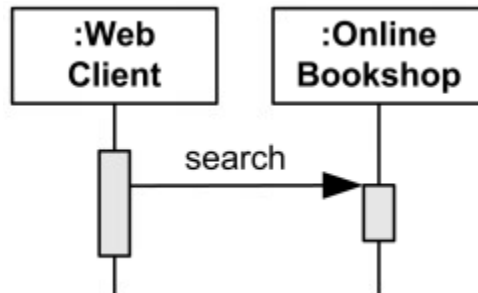
پیام‌ها بر اساس نوع اقدام

یک پیام یا یک فراخوانی، اجرا و شروع اجرا یا ارسال و دریافت سیگنال را منعکس می‌کند. هنگامی که یک پیام، یک فراخوان عملیات را نشان می‌دهد، آرگومان‌های پیام، آرگومان‌های عملیات هستند. هنگامی که یک پیام، یک سیگنال را نشان می‌دهد، آرگومان‌های پیام ویژگی‌های سیگنال هستند. بسته به نوع اقدامی که برای تولید پیام استفاده شده است، پیام می‌تواند یکی از موارد زیر باشد:

- فراخوانی همزمان یا synchronous call
- فراخوانی ناهمزمان یا asynchronous call
- سیگنال ناهمزمان یا asynchronous signal
- ایجاد کردن یا create
- حذف یا delete
- پاسخ یا reply

Synchronous Call

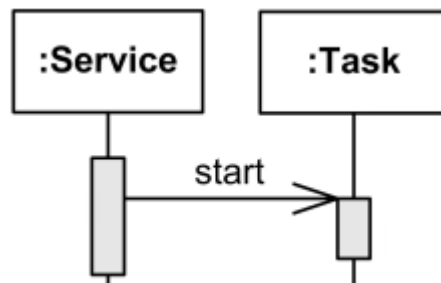
فراخوانی همزمان معمولاً نشان دهنده عملیات است که شامل ارسال پیام، تعلیق اجرا و منتظر ماندن برای دریافت پاسخ است. پیام‌های فراخوانی همزمان با سر فلش پر شده نشان داده می‌شوند.



شیء Web Client شیء Online Bookshop را جستجو می کند و منتظر نتایج می ماند

asynchronous call

فراخوانی ناهمزمان که نشان دهنده عملیاتی است که شامل ارسال پیام و بلافاصله بدون انتظار برای دریافت نتیجه، کار ادامه می یابد، است. پیام های ناهمزمان دارای سر پیکان باز هستند.



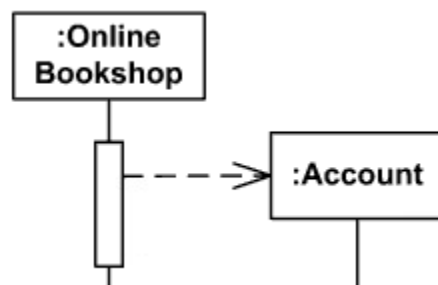
شیء Service شیء Task را شروع می کند و بلافاصله به صورت موازی ادامه می یابد

asynchronous signal

پیام سیگنال ناهمزمان مربوط به عملکرد سیگنال ارسال ناهمزمان است.

Create Message

پیام Create برای ایجاد یک شیء Lifeline به یک Lifeline ارسال می شود. به صورت یک خط چین با نوک پیکان باز (مانند پیام پاسخ) نشان داده می شود و به سر Lifeline ایجاد شده اشاره می کند.

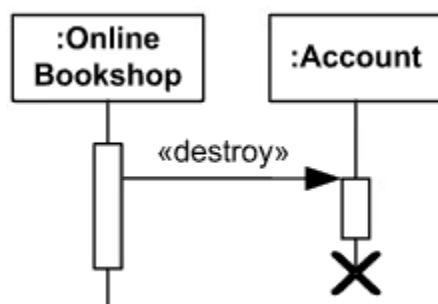


شیء Online Bookshop شیء Account را ایجاد می کند

توجه داشته باشید که این قرارداد عجیب و غریب برای ارسال پیام به یک شیء غیر موجود برای ایجاد آن شیء در UML 1.x و 2.x استفاده می‌شود. در Smalltalk-80 اشیاء جدید با ارسال پیام به کلاس‌ها ایجاد می‌شوند که نمونه‌ای از کلاس ایجاد شده و برگردانده می‌شود. بنابراین یک راه برای تفسیر UML ایجاد نماد پیام احتمالاً به عنوان میانبر برای این اقدامات است.

Delete Message

پیام حذف برای پایان دادن به Lifeline دیگری ارسال می‌شود. Lifeline معمولاً با یک ضربدر به شکل X در پایین خود، پایان می‌یابد که نشان دهنده رویداد جمع‌آوری یا destruction occurrence است.

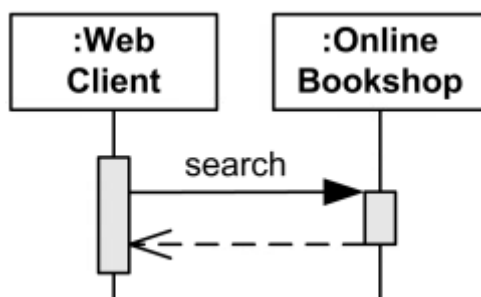


شیء Online Bookshop شیء Account را خاتمه می‌دهد

مشخصات UML 2.4 نه نماد خاصی برای حذف پیام و نه کلیشه‌ای ارائه می‌دهد. تا زمانی که آنها نمادی برای این موضوع ارائه کنند، می‌توانیم از کلیشه سفارشی «destroy» استفاده کنیم.

Reply Message

پیام پاسخ به یک فراخوانی عملیاتی به صورت یک خط چین با سر فلش باز نشان داده می‌شود (شبیه پیام ایجاد است).



شیء Web Client شیء Online Bookshop را جستجو می‌کند و منتظر بازگشت نتایج می‌ماند

پیام‌ها بر اساس رویدادها

بسته به اینکه آیا رویداد ارسال پیام و رویدادهای دریافت وجود دارد یا خیر، پیام می‌تواند یکی از موارد زیر باشد:

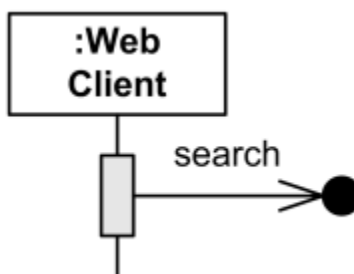
- پیام کامل شده یا complete message
- پیام گم شده یا lost message
- پیام پیدا شده یا found message
- پیام نامشخص یا unknown message (پیش فرض)

complete message

معنای یک پیام کامل شده، ردیابی <sendEvent, receiveEvent> است. هر دو sendEvent و receiveEvent در پیام حضور دارند.

lost message

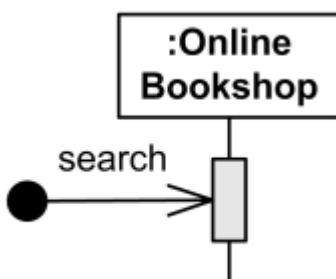
پیام گم شده، پیامی است که در آن رویداد ارسالی یا sendEvent مشخص است، اما رویداد دریافت یا receiveEvent وجود ندارد. این پیام به گونه‌ای تفسیر می‌شود که گویی پیام هرگز به مقصد خود نخواهد رسید. معنای این نوع پیام، رد پای <sendEvent> است، receiveEvent وجود ندارد. پیام‌های گم شده با یک دایره سیاه کوچک در انتهای پیکان پیام نشان داده می‌شوند.



شیء Web Client پیام جستجویی ارسال کرد که گم شد

found message

پیام یافت شده پیامی است که در آن رویداد دریافت یا receiveEvent مشخص است، اما رویداد ارسال یا sendEvent شناخته شده نیست یا وجود ندارد. به گونه‌ای تعبیر می‌شود که گویی مبدأ پیام خارج از محدوده توصیف است. این ممکن است برای مثال نویز یا فعالیت دیگری باشد که ما نمی‌خواهیم با جزئیات آن را توضیح دهیم. مفهوم آن به این معنی است: <receiveEvent>، در حالی که رویداد ارسال وجود ندارد. پیام‌های یافت شده با یک دایره سیاه کوچک در ابتدای پیکان پیام مشخص می‌شوند.



شیء Online Bookshop پیام جستجویی با منشأ ناشناخته دریافت می‌کند

unknown message

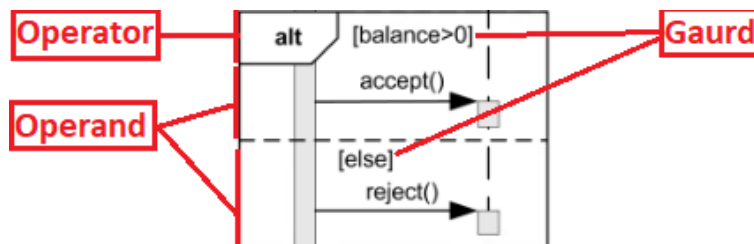
در این نوع پیام sendEvent و receiveEvent وجود ندارند (منطقاً این نوع از پیام نباید وجود داشته باشد).

قطعه ترکیبی یا Combined Fragment یک قطعه تعاملی است که ترکیبی از قطعات تعاملی را تعریف می‌کند. یک قطعه ترکیبی توسط یک عملگر تعاملی و عملوندهای تعاملی مربوطه تعریف می‌شود. با استفاده از قطعات ترکیبی، کاربر قادر خواهد بود تعدادی از ردیابی‌ها را به صورت فشرده و مختصر توصیف کند. قطعه ترکیبی ممکن است دارای محدودیت‌های تعاملی باشد که در UML 2.4 به نام guards نیز نامیده می‌شود. عملگر تعاملی می‌تواند یکی از موارد زیر باشد:

- **alt**: alternatives یا محدوده معرفی رفتارهای جایگزین
- **opt**: option یا محدوده معرفی گزینه‌ها
- **loop**: iteration یا محدوده معرفی تکرار رفتار
- **break**: break یا محدوده معرفی شکست رفتار
- **par**: parallel یا محدوده معرفی رفتارهای موازی
- **strict**: strict sequencing یا محدوده معرفی رفتارهایی که توالی دقیق دارند (قابل شمارش و مبتنی بر یک مقدار دقیق یا قابل پیش‌بینی)
- **seq**: weak sequencing یا محدوده معرفی رفتارهایی که توالی مشروط دارند (غیر مبتنی بر یک مقدار دقیق و غیر قابل پیش‌بینی)
- **critical**: critical region یا محدوده معرفی رفتارهای بحرانی
- **ignore**: ignore یا محدوده معرفی رفتار صرف نظر کردن از یک عملیات
- **consider**: consider یا محدوده معرفی فرضیات
- **assert**: assertion یا محدوده معرفی اعلان‌ها
- **neg**: negative یا محدوده معرفی وضعیت‌های نامعتبر

محدودیت‌های تعامل یا interaction constraint

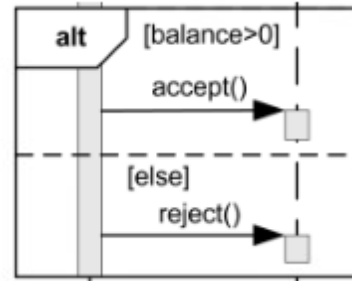
یک محدودیت تعاملی، محدودیت یا قیدی است که در تعاملات استفاده می‌شود. به عبارت دیگر عبارت **boolean** یا منطقی که به واسطه یک عملوند یک Combined Fragment را کنترل و مدیریت می‌کند. یک محدودیت تعاملی در داخل Lifeline که اولین رویداد موجود را پوشش می‌دهد، در بالای آن رویداد، درون براکتی نشان داده می‌شود. UML 2.4 اغلب از محدودیت تعامل به عنوان کنترل کننده اشاره می‌کند.



alternatives :alt

عملگر تعاملی **alt** به این معنی است که Combined Fragment مورد نظر یک Choice یا انتخاب یا جایگزین‌های رفتاری را معرفی می‌کند. حداکثر یکی از عملوندها انتخاب خواهد شد. عملوند انتخاب شده باید یک عبارت کنترلی بی‌واسطه یا باواسطه داشته باشد که در این نقطه از تعامل، صحیح ارزیابی شود. در صورتی که عملوند کنترلی نداشته باشد، یک کنترل واقعی باواسطه در نظر گرفته می‌شود. عملوند کنترل شده توسط **else** به معنای نفی انفصال همه کنترل‌های دیگر است. اگر هیچ یک از عملوندها کنترلی نداشته باشد که **true** ارزیابی شود، هیچ یک از عملوندها اجرا نمی‌شوند و باقیمانده Combined Fragment محصور شده اجرا می‌شود.

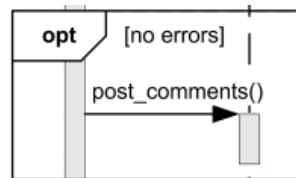
یک مسیر جایگزین به واسطه بررسی و
ارزیابی شرط‌های محدودکننده موجود
در هر بخش عملوند، انتخاب می‌شود.
این بررسی از بالا به پایین بین شرط‌های
محدودکننده صورت می‌گیرد.



اگر $0 < \text{balance}$ باشد، متد `Accept()` فراخوانی می‌شود، در غیر این صورت متد `reject()` فراخوانی می‌گردد

option :opt

عملگر تعاملی `opt` به این معنی است که Combined Fragment یک `choice` یا انتخاب از رفتار را نشان می‌دهد که در آن یا عملوند منحصر به فرد (`sole`) اتفاق می‌افتد یا هیچ اتفاقی نمی‌افتد. یک `option` از نظر معنایی معادل یک Combined Fragment از نوع `alt` است که در آن یک عملوند با محتوای غیر خالی وجود دارد و عملوند دوم خالی است.



اگر خطایی وجود نداشت متد `post_comments()` فراخوانی می‌شود

iteration :loop

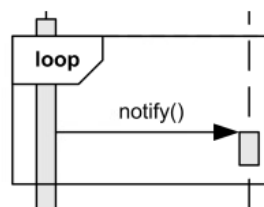
عملگر تعاملی `loop` به این معنی است که Combined Fragment یک حلقه را نشان می‌دهد. عملوند حلقه چندین بار تکرار خواهد شد. ساختار حلقه یک کاربرد بازگشتی از عملگر `seq` را نشان می‌دهد که در آن عملوند حلقه پس از دریافت نتیجه تکرارهای قبلی، تکرار می‌شود. مشخصات UML 2.4 توضیحات عجیبی از عملگر `loop` با مثال‌های عجیب و غریب ارائه می‌دهد. من سعی خواهم کرد در اینجا مقداری از آن را شفاف کنم. حلقه را می‌توان با محدوده‌های تکرار و یا یک کنترل‌کننده، کنترل کرد. عملوند `loop` می‌تواند دارای محدوده‌های تکرار باشد که ممکن است شامل حداقل تکرار و حداکثر تکرار حلقه باشد. نگارش متنی حلقه به صورت زیر است:

`loop-operand ::= loop ['(' min-int [',' max-int ']')]`

`min-int ::= non-negative-integer`

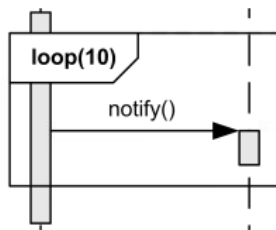
`max-int ::= positive-integer | '*'`

اگر حلقه هیچ محدوده مشخصی نداشته باشد، به معنای حلقه بی‌نهایت با صفر به عنوان حداقل و بی‌نهایت به عنوان حداکثر تکرار است.



حلقه بالقوه بی‌نهایت

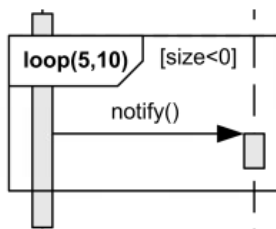
اگر فقط min-int مشخص شده باشد، به این معنی است که حداکثر برابر با حداقل است و حلقه دقیقاً به تعداد دفعات مشخص شده اجرا می‌شود.



حلقه تا دقیقاً 10 بار اجرا شود

اگر max-int مشخص شده باشد، باید بزرگتر یا مساوی min-int باشد. حلقه حداقل تعداد min-int و حداکثر تعداد max-int را تکرار می‌کند. علاوه بر محدوده‌های تکرار، حلقه می‌تواند یک محدودیت تعاملی نیز داشته باشد. یک محدودیت تعاملی، یک عبارت boolean یا منطقی است که در براکت نوشته می‌شود. برای ایجاد پیچیدگی‌های بیشتر، UML 2.4 هر دوی آنها را کنترل‌کننده می‌نامد.

UML سعی کرده‌است که ساده‌ترین اشکال حلقه for و while را با هم ترکیب کند که باعث معنایی عجیب حلقه UML 2.3 در صفحه 488 [UML 2.3 - Superstructure] شده است: "پس از حلقه به اندازه حداقل تکرار، اجرا شد و عبارت Boolean، نادرست بود، حلقه خاتمه می‌یابد." همانطور که به نظر می‌رسد این تعریف یک حلقه do...while است.



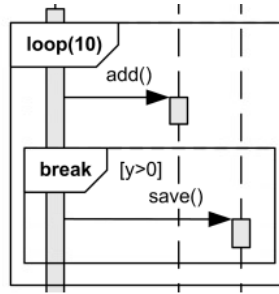
ممکن است حدس بزنیم که طبق UML 2.3، انتظار می‌رود، حلقه حداقل 5 بار و حداکثر 10 بار اجرا شود

اگر شرط کنترل [size<0] مقدار false شود، صرف نظر از حداقل تعداد تکرار مشخص شده حلقه خاتمه می‌یابد

(پس چرا به آن عدد حداقل نیاز داریم که مشخص شود؟!)

break :break

عملگر تعاملی break نشان دهنده یک سناریوی شکست یا استثنایی است که به جای باقیمانده Combined Fragment انجام می‌شود. زمانی که کنترل مقدار true باشد، عملگر break با کنترل انتخاب می‌شود. در این مورد بقیه Combined Fragment که مستقیماً محصور شده است؛ نادیده گرفته می‌شود. هنگامی که کنترل کننده عملوند break نادرست است، عملوند break نادیده گرفته می‌شود و بقیه Combined Fragment محصور شده؛ ادامه می‌یابد.

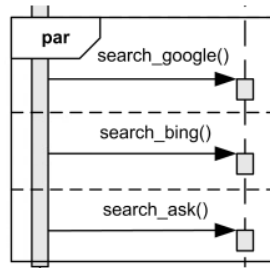


اگر $y > 0$ حلقه محصور شده، خاتمه می‌یابد

یک Combined Fragment با عملگر break باید تمام Lifeline های قطعه تعامل محصور کننده را پوشش دهد. توجه داشته باشید، UML اجازه می‌دهد تنها یک سطح، که به طور مستقیم بخش تعامل را در برمی‌گیرد، رها یا شکسته شود. اگر حلقه یا حلقه تودرتو با قطعات ترکیبی دیگر شکسته شود، می‌تواند واقعا آزار دهنده باشد. UML 2.3 بیان می‌کند که وقتی عملوند break، کنترل ندارد، انتخاب بین عملوند break و بقیه بخش تعامل محصور شده به صورت "غیر قطعی" انجام می‌شود که به احتمال زیاد به معنای "غیر قابل پیش‌بینی" است.

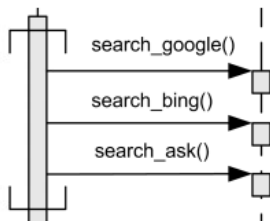
parallel:par

عملگر تعاملی par، اجرای موازی بالقوه رفتارهای عملوندهای قطعه ترکیبی را تعریف می‌کند. تا زمانی که ترتیب تحمیل شده توسط هر عملوند حفظ شود، عملوندهای مختلف را می‌توان به هر طریقی در هم آمیخت. مجموعه‌ای از ردیابی عملگر موازی، تمام راه‌ها یا ترکیب‌های ممکن را توصیف می‌کند که ممکن است مشخصات رخداد عملوندها بدون تغییر ترتیب در هر عملوند به هم متصل شوند.



Google، Bing و Ask را به هر ترتیبی، احتمالاً موازی، جستجو کنید

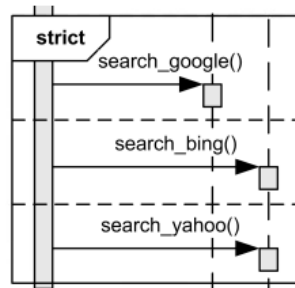
قطعه ترکیبی par برای موقعیت‌های که اهمیت ترتیب رویدادها در یک Lifeline ناچیز است، قابل استفاده است. در یک منطقه Ecoregion یا همترازی از یک Lifeline که توسط براکت‌های افقی محدود شده است، همه قطعات محدود شده، مستقیماً به عنوان عملوندهای جداگانه یک قطعه ترکیبی par در نظر گرفته می‌شوند.



Ecoregion یا همترازی: Google، Bing و Ask را به هر ترتیبی، احتمالاً موازی، جستجو کنید

strict sequencing :strict

عملگر تعامل به یک strict (ترتیبی) دقیق عملوندها در سطح اول در Combined Fragment نیاز دارد.



Google, Bing و yahoo را دقیقاً به ترتیب جستجو کنید

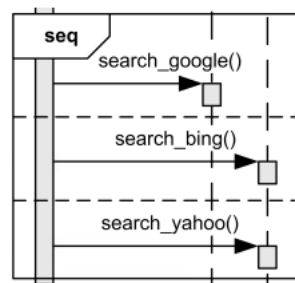
عملگرهای سطوح پایین‌تر در Combined Fragment موجود به طور مستقیم با سایر مشخصات رخداد Combined Fragment محصور مقایسه نمی‌شوند. وقتی صحبت از کلمه دقیقاً می‌شود، به این معنی است که مختصات عمودی قطعات موجود نه تنها در یک Lifeline بلکه در کل محدوده Combined Fragment مهم است.

weak sequencing :seq

عملگر تعاملی seq به این معنی است که Combined Fragment یک توالی ضعیف بین رفتار عملوندها را نشان می‌دهد. توالی ضعیف با مجموعه‌ای از ردیابی‌ها با این ویژگی‌ها تعریف می‌شود:

- ترتیب مشخصات رویداد در هر یک از عملوندها نگاه‌داشته می‌شود.
- مشخصات رویداد در Lifeline‌های مختلف از عملوندهای مختلف ممکن است به هر ترتیبی باشد.
- مشخصات رخداد روی Lifeline‌های مشابه از عملوندهای مختلف به گونه‌ای مرتب شده‌اند که مشخصات رویداد عملوند اول قبل از عملوند دوم باشد.

زمانی که عملوندها روی مجموعه‌های مختلف مشارکت کنندگان قرار دارند، عملگر تعاملی seq، به ادغام par کاهش می‌یابد. وقتی عملوندها روی یک مشارکت کننده کار می‌کنند، عملگر تعاملی seq، به توالی strict کاهش می‌یابد.

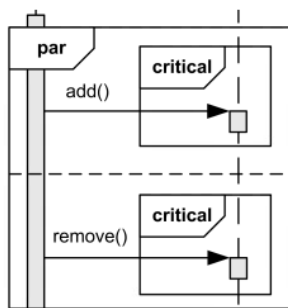


احتمالاً همانطور که Google را جستجو می‌کنید؛ به موازات آن، Bing و yahoo را نیز جستجو کنید، اما قبل از یاهو، بینگ را جستجو کنید

critical region :critical

عملگر تعاملی critical تعریف می‌کند که Combined Fragment یک منطقه بحرانی را نشان می‌دهد. یک منطقه بحرانی منطقه‌ای است با ردیابی‌هایی که توسط سایر مشخصات رویداد (روی Lifeline درون منطقه) قابل اقدام نیست. این بدان معنی است که این ناحیه به صورت

اتمیک و مستقل، توسط قطعه محصور کننده مورد نظر، اجرا می‌شود و نمی‌توان آن را در قطعات دیگر ترکیب نمود، به عنوان مثال. توسط عملگر **par**



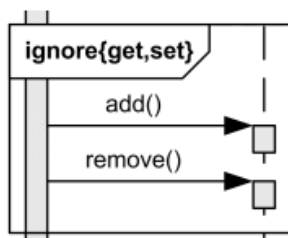
متدهای **Add()** یا **remove()** را می‌توان به صورت موازی فراخوانی کرد، اما هر کدام باید به عنوان یک محدوده بحرانی اجرا شوند.

ignore: ignore

مفهوم، توصیف و هدف عملگر تعاملی **ignore** مبهم است. UML 2.3 این معنا را اینگونه تعریف می‌کند: "برخی از انواع پیام وجود دارد که در این **Combined Fragment** نشان داده نمی‌شوند. این انواع پیام را می‌توان ناچیز در نظر گرفت و اگر در اجرای متناظر ظاهر شوند به طور ضمنی نادیده گرفته می‌شوند. از سوی دیگر، می‌توان نادیده گرفتن را به این معنی دانست که انواع پیام‌هایی که نادیده گرفته می‌شوند می‌توانند در هر جایی از ردیابی ظاهر شوند."

از سوی دیگر، توضیحات شکل 14.25 در ص. 530 [UML 2.3 - Superstructure] به این معنی است که این نوع تعامل می‌تواند برای تعیین آزمایش یک سیستم موجود، استفاده شود. در زمان اجرا، پیام‌هایی که در تست‌ها نادیده گرفته می‌شوند، "البته توسط سیستم در حال اجرا به نحوی مدیریت می‌شوند".

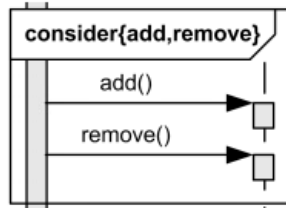
لیست پیام‌های نادیده گرفته شده در یک جفت آکولاد "{" و "}" محصور خواهند شد. عملیات نادیده گرفتن معمولاً با عملیات دیگری مانند **"assert ignore {m, s}"** ترکیب می‌شود.



پیام‌های **get()** و **set()** را نادیده گرفته شده‌اند

consider: consider

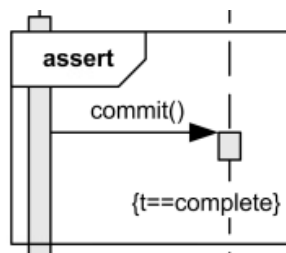
عملگر تعاملی **consider**، تعریف می‌کند که کدام پیام باید در این **Combined Fragment** در نظر گرفته شود، به عبارت دیگر هر پیام دیگری نادیده گرفته می‌شود. لیست پیام‌های در نظر گرفته شده در یک جفت آکولاد "{" و "}" محصور خواهند شد. عملیات در نظر گرفتن معمولاً با عملیات دیگری مانند "اظهار در نظر گرفتن {s, m}" ترکیب می‌شود.



فقط پیام‌های `add()` یا `remove()` را در نظر بگیرید، هر پیام دیگری را نادیده بگیرید

assert :assert

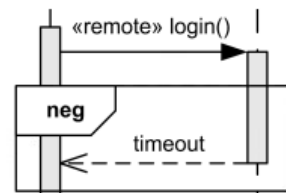
عملگر تعاملی `assert` در یک `Combined Fragment` تأکیدی را نشان می‌دهد؛ به این معنی که دنباله‌های عملوند `assert` تنها ادامه معتبر هستند (باید با طراحی مسیر مورد انتظار یا به عبارت دیگر بهترین مسیر سیستم هم‌راستا باشد). همه ادامه‌های دیگر یک مسیر جایگزین هستند و منجر به یک ردیابی نامعتبر می‌شوند.





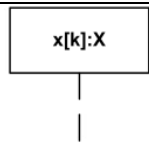
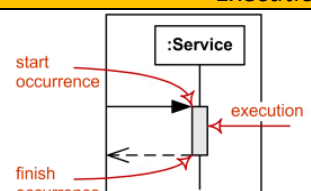

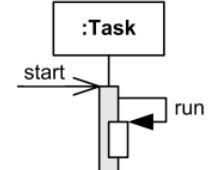
پیام `Commit()` باید در این نقطه رخ دهد و سپس وضعیت با یک مقدار ثابت، ارزیابی شود

negative :neg

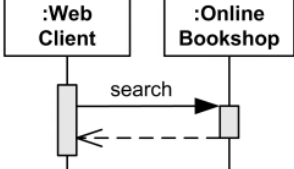


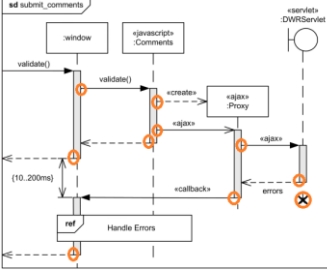
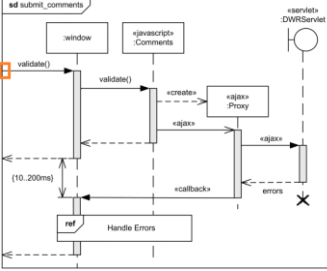
عملگر تعاملی `neg` که `Combined Fragment` از ردیابی‌ها را توصیف می‌کند که منفی (نامعتبر) تعریف شده‌اند. ردیابی‌های منفی، ردیابی‌هایی هستند که در هنگام از کار افتادن سیستم رخ می‌دهند. تمام قطعات تعاملی که غیر منفی هستند مثبت در نظر گرفته می‌شوند، به این معنی که آنها ردیابی را توصیف می‌کنند که معتبر هستند و باید امکان پذیر باشند.



اگر پیام `timeout` دریافت کنیم، به این معنی است که سیستم از کار افتاده است

شرح	Notation
Lifeline	
یک Lifeline با استفاده از نمادی نشان داده می‌شود که شامل یک مستطیل است که "سر" آن را تشکیل می‌دهد و به دنبال آن یک خط عمودی (که ممکن است خط چین دار باشد) که نشان دهنده طول عمر مشارکت کننده است.	 <p>Stock با نام "data" از کلاس Stock</p>
Anonymous lifeline نامی ندارد - نماینده از یک کلاس نامشخص.	 <p>User Lifeline ناشناس کلاس User</p>
Selector می‌تواند نمونه‌ای را به عنوان Lifeline از مجموعه نمونه‌ها، فیلتر کند.	 <p>Lifeline به نام X نمونه‌ای از کلاس X با پارامتر [k]</p>
Execution	
اجرا که نام کامل آن execution specification یا مشخصات اجرا است و به طور غیررسمی به آن فعالیت هم گفته می‌شود، interaction fragment یا یک قطعه تعاملی است که نشان دهنده یک مرحله در طول عمر مشارکت کننده است که ممکن است موارد زیر در آن وقوع یابد:	 <p>مشخصات اجرایی به صورت مستطیل خاکستری در Lifeline سرویس، نشان داده شده است</p>
<ul style="list-style-type: none"> اجرای یک واحد رفتار یا عمل در Lifeline ارسال سیگنال به مشارکت کننده دیگر منتظر پیام پاسخ از سوی مشارکت کننده دیگر <p>مدت زمان اجرا با دو execution occurrences یا رویداد اجرا نشان داده می‌شود: رویداد شروع و رویداد پایان. اجرا به صورت یک مستطیل نازک خاکستری یا سفید روی Lifeline نشان داده می‌شود.</p>	
اجرا را می‌توان با یک مستطیل پهن‌تر به همراه برجسب نشان داد، در اینجا برجسب معمولاً عملی را که اجرا شده است، مشخص می‌کند.	 <p>اجرا به صورت مستطیل پهن‌تری نشان داده می‌شود که برجسب عملیات مورد نظر را نشان می‌دهد</p>
اجراهای همپوشانی شده روی یک Lifeline با مستطیل‌های روی مستطیل اصلی نشان داده می‌شوند. این حالت، وضعیتی را نشان می‌دهد که از بخشی از شیء مورد نظر، بخشی دیگر از همان شیء فراخوانی می‌شود و در بخش اول، کار یا عملیات متوقف می‌ماند تا کار یا عملیات در بخش دوم خاتمه یابد و نتیجه به بخش اول ارائه گردد و سپس کار و عملیات در بخش اول ادامه می‌یابد.	

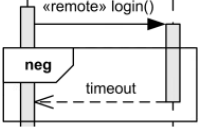
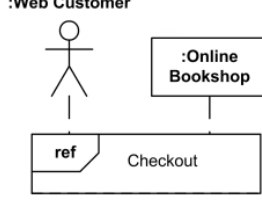
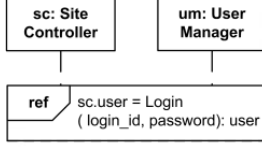
	<p>مشخصه اجرایی همپوشانی شده در همان Lifeline یا به عبارت دیگر پیام به خود</p> <p>مشخصه اجرایی همپوشانی شده در همان Lifeline یا به عبارت دیگر فراخوانی پیام برگشت به فراخوانی کننده</p>
<p>فراخوانی بازگشتی، روشی برای ساده کردن طراحی می‌باشد. از نظر معنی شناختی، این رفتار تعاملی به گونه‌ای توصیف می‌شود که در آن فراخوانی شونده، بنا به شرایطی، فراخوان خود را مجدد فراخوانی می‌کند و این کار ممکن است به صورت متوالی تکرار شود.</p>	
Message	
<p>Synchronous call یا فراخوانی همزمان، معمولاً نشان دهنده فراخوانی عملیات است که از منظر دیگر حالتی شبیه به ارسال پیام و تعلیق اجرا در حالی که منتظر پاسخ هستید، می‌باشد. پیام‌های همزمان با سر فلش پر شده نشان داده می‌شوند. ارسال کننده پیام تا زمان دریافت پاسخ، اجرای خود را متوقف می‌نماید و پس از دریافت پاسخ، اجرای خود را ادامه می‌دهد. به عبارت دیگر در این حالت، اجرا یا مشخصات اجرا بهتر بود که خالی می‌شد ولی توافق بر این است که انتظار اجرا مانند اجرا در خط زمانی، نمایش داده شود.</p>	<p>شیء Web Client شیء Online Bookshop را جستجو می‌کند و منتظر نتایج می‌ماند</p>
<p>Asynchronous Call یا فراخوانی ناهمزمان، معمولاً نشان دهنده فراخوانی عملیات است که از منظر دیگر حالتی شبیه به ارسال پیام و ادامه اجرا، می‌باشد. پیام‌های ناهمزمان دارای سر پیکان باز هستند. ارسال کننده پیام بلافاصله پس از ارسال پیام، اجرای خود را ادامه می‌دهد و منتظر دریافت پاسخ نمی‌ماند.</p>	<p>شیء Service شیء Task را شروع می‌کند و بلافاصله به صورت موازی ادامه می‌یابد</p>
<p>Create message پیامی است که به Lifeline ارسال می‌شود تا خودش را ایجاد کند. به عبارت دیگر با فرض اینکه یک Lifeline نماینده یک شیء از یک کلاس است، این پیام به کلاس ارسال می‌شود تا یک نمونه از خود را بسازد تا Lifeline نمایندگی آن را داشته باشد. توجه داشته باشید که در OOAD، ارسال پیام ایجاد به یک شیء غیر موجود برای ایجاد خود، امری عجیب اما رایج است. اما در OOP یکی از اصول اصلی تولید اشیاء محسوب می‌شود. در زندگی واقعی، پیام ایجاد به برخی از محیط‌های زمان اجرا ارسال می‌شود. Create message به صورت یک خط چین با سر پیکان باز (مانند Reply message) نشان داده می‌شود و همواره به بالای یک Lifeline ایجاد شده اشاره می‌کند.</p>	<p>شیء Online Bookshop شیء Account را ایجاد می‌کند</p>
<p>Delete message که در نسخه‌های قبلی UML، stop نامیده می‌شود پیامی است که برای پایان دادن به Lifeline دیگری ارسال می‌شود. Lifeline دریافت کننده این پیام معمولاً با یک ضربدر به شکل X در پایین خود، پایان می‌یابد که نشان دهنده رویداد جمع‌آوری است. مشخصات UML 2.3 نه نماد خاصی برای Delete message و نه کلیشه‌ای ارائه می‌دهد. تا زمانی که آنها نمادی برای این منظور ارائه کنند، می‌توانیم از کلیشه سفارشی «destroy» استفاده کنیم.</p>	<p>شیء Online Bookshop شیء Account را خاتمه می‌دهد</p>

<p>Reply message یک پیام پاسخ به یک فراخوانی عملیاتی است که به صورت یک خط چین با سر فلش باز نشان داده می‌شود.</p>	 <p>شیء Web Client شیء Online Bookshop را جستجو می‌کند و منتظر پیام پاسخ می‌ماند</p>
<p>Lost Message پیامی است که در آن رویداد ارسالی مشخص است، اما رویداد دریافتی وجود ندارد. به گونه‌ای تفسیر می‌شود که گویی پیام هرگز به مقصد خود نرسیده است. Lost Message با یک دایره سیاه کوچک در انتهای پیکان پیام نشان داده می‌شوند.</p>	 <p>شیء Web Client پیام جستجویی ارسال کرد که گم شد</p>
<p>Found Message پیامی است که در آن رویداد دریافتی وجود دارد، اما رویداد ارسالی وجود ندارد یا نامعلوم است. به گونه‌ای تعبیر می‌شود که گویی مبدأ پیام خارج از محدوده توصیف شده، قرار دارد. این ممکن است به عنوان مثال نویز یا فعالیت دیگری باشد که ما نمی‌خواهیم با جزئیات توضیح دهیم. Found Message با یک دایره سیاه کوچک در ابتدای پیام، مشخص می‌شوند.</p>	 <p>شیء Online Bookshop پیام جستجویی با منشاء ناشناخته دریافت می‌کند</p>
<p style="text-align: center;">Occurrence specification</p>	
<p>یک Occurrence specification، رویدادی است که در جریان یک توالی تعاملی، فعال‌کننده (Trigger) یک رفتار مشخص است. این رویدادها به Lifeline های درون نمودار مرتبط هستند و در زمان طراحی یک نمودار تعاملی خاص، رویدادهایی که در سایر نمودارهای تعاملی قرار دارند از منظر این نمودار یک Occurrence specification در نظر گرفته نمی‌شوند. از جمله این رفتارها می‌توان به موارد زیر اشاره کرد:</p> <ul style="list-style-type: none"> • فراخوانی متد • بازگشت نتیجه • ایجاد شیء • جمع‌آوری شیء 	 <p>رویدادهایی که هر یک آغازگر یک پیام یا فراخوانی یک متد هستند</p>
<p style="text-align: center;">gate</p>	
<p>نقطه اتصال یک پیام خارج از یک Combined Fragment با یک پیام در داخل آن را Gate می‌گویند. هدف از Gate ها و پیام‌های بین آنها، مشخص کردن فرستنده و گیرنده هر پیام است. Gate ها دقیقاً به عنوان نقاط اتصال پیام در یک fragment معرفی می‌شوند. Gate ها به صورت بی‌واسطه یا باواسطه نامگذاری می‌شوند. نام Gate باواسطه با افزودن جهت پیام ("in" یا "out") و نام پیام، ساخته می‌شود. پسوند in معرف Gate ورودی است. در مثال تصویر، Gate مورد نظر in_validate نام خواهد داشت.</p>	 <p>یک Gate ورودی محل فراخوانی آغازین یک Combined Fragment است</p>

<p>نقطه اتصال یک پیام خارج از یک Combined Fragment با یک پیام در داخل آن را Gate می‌گویند. هدف از Gate ها و پیام‌های بین آنها، مشخص کردن فرستنده و گیرنده هر پیام است. Gate ها دقیقاً به عنوان نقاط اتصال پیام در یک fragment معرفی می‌شوند. Gate ها به صورت بی‌واسطه یا باواسطه نامگذاری می‌شوند. نام Gate باواسطه با افزودن جهت پیام ("in" یا "out") و نام پیام، ساخته می‌شود. پسوند out معرف Gate خروجی است. در مثال تصویر، Gate اول از بالا out_resault نام خواهد داشت و Gate دومی out_error خواهد بود.</p>	<p>یک Gate خروجی محل بازگشت نتیجه رفتار یک Combined Fragment است</p>
<p style="text-align: center;">Destruction Occurrence</p>	
<p>Lifeline Destruction occurrence یک رویداد پیام است که نشان دهنده جمع‌آوری نمونه توصیف شده توسط Lifeline مورد نظر است. این رویداد ممکن است در مراحل بعدی، منجر به جمع‌آوری اشیاء دیگری بشود که این شیء در ترکیبی از آنها شرکت دارد. در پایین Destruction occurrence در یک Lifeline هیچ رویداد دیگری ظاهر نمی‌شود. نام کامل این رویداد در UML، destruction occurrence specification است. تا قبل از UML 2.4 به آن Destruction Event و قبل از آن stop نامیده می‌شد. جمع‌آوری نمونه با یک صلیب به شکل X در پایین یک Lifeline به تصویر کشیده می‌شود.</p>	<p>Lifeline به نام Account جمع‌آوری شده است</p>
<p style="text-align: center;">State Invariant</p>	
<p>یک State Invariant یک قطعه تعاملی است که یک محدوده زمانی در زمان اجرا را برای شرکت کنندگان در تعامل نشان می‌دهد. State Invariant ها ممکن است برای کنترل و بررسی انواع مختلفی از محدوده‌ها، مثل محدوده‌هایی که بررسی مقادیر ویژگی‌ها یا متغیرها را مشخص می‌کنند، بررسی حالت‌های داخلی یا خارجی را مشخص می‌کنند و غیره استفاده شوند. State Invariant معمولاً داخل یک مجموعه آکولاد "{" و "}" در داخل یک Lifeline محدود می‌شود. به عبارت دیگر یک State Invariant مکانی است که شروط مختلفی که ممکن است ریشه داخلی یا خارجی نسبت به Lifeline داشته باشند، در آن بررسی می‌شود.</p>	<p>مشخصه t مربوط به کلاس Task باید برابر با complete باشد</p>
<p>یک State Invariant همچنین می‌تواند به عنوان یک نماد که حالت یا وضعیت را نشان می‌دهد، استفاده شود؛ که البته در این حالت، وضعیت شیء مرتبط با Lifeline را در یک محدوده، نمایش می‌دهد. یک State Invariant می‌تواند ترتیب داخلی رفتارهای ترتیبی مرتبط با یک Lifeline را همانطور که باید باشد یا وضعیت‌های خارجی نسبت به یک Lifeline را در قالبی شبیه به یک "جعبه سیاه" در داخل آن Lifeline نشان دهد.</p>	<p>Task باید در حالت Finished باشد</p>
<p style="text-align: center;">Combined Fragment</p>	
<p>عملگر تعاملی alt در یک Combined Fragment، معرف این است که تعامل مورد نظر، یک انتخاب یا جایگزین‌های رفتاری را در قالب عملوندها و محیط‌های محدود شده آنها، نشان می‌دهد. حداکثر یکی از عملوندها، انتخاب خواهد شد. از طرف دیگر عملوندهای قابل انتخاب، باید یک عبارت تضمین شده باواسطه یا بی‌واسطه داشته باشند که در این نقطه از تعامل، حداقل یکی از آنها true ارزیابی شوند. وقتی صحبت از عبارت‌های تضمین شده می‌شود؛ منظور این است که عبارت‌ها به گونه‌ای باشند که اجرای کار را بر اساس رفتار صحیح، ادامه دهند. به طور مثال وجود [balance>0] و [else] در کنار هم باعث می‌شود که حداقل یکی از این عبارت‌ها true ارزیابی شود.</p>	<p>اگر $balance > 0$ باشد، متد <code>Accept()</code> در غیر این صورت متد <code>reject()</code> فراخوانی می‌گردد</p>
<p>عملگر تعاملی opt در یک Combined Fragment، معرف این است که تعامل مورد نظر، یک انتخاب رفتار است که در آن یا یک عملوند منفرد اتفاق می‌افتد یا هیچ اتفاقی نمی‌افتد. یک عملگر تعاملی opt از نظر معنایی معادل یک عملگر تعاملی alt است که در آن یک عملوند با محتوای و عملوند دوم بدون محتوای یا خالی است.</p>	<p>اگر خطایی وجود نداشت متد <code>post_comments()</code> فراخوانی می‌شود</p>

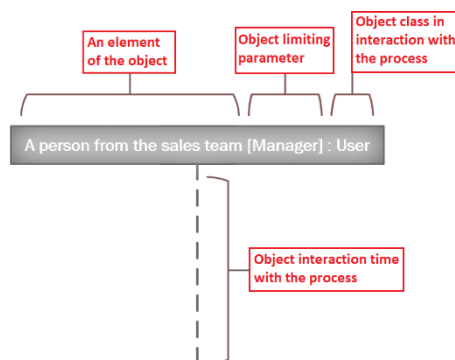
<p>عملگر تعاملی loop در این حالت هیچ مرز مشخصی ندارد، در این حالت، حلقه بالقوه بی‌نهایت است و حداقل صفر دفعه و حداکثر بی‌نهایت دفعه تکرار خواهد شد.</p>	 <p>حلقه بالقوه بی‌نهایت</p>
<p>عملگر تعاملی loop اگر فقط عدد حداقل (min-int) مشخص کرده باشد، به این معنی است که مقدار حداکثر با حداقل برابر است و حلقه دقیقاً به تعداد دفعات مشخص شده اجرا می‌شود.</p>	 <p>حلقه دقیقاً 10 دفعه اجرا می‌شود</p>
<p>عملگر تعاملی loop اگر هر دو عدد حداقل و حداکثر، مشخص کرده باشند، حلقه، حداقل تعداد min-int و حداکثر تعداد max-int دفعه تکرار می‌شود. علاوه بر تکرارهای یک حلقه که به واسطه حداقل و یا حداکثر مشخص شده می‌باشد، محدوده تکرار حلقه می‌تواند یک ارزیابی تعاملی نیز داشته باشد. ارزیابی تعاملی، یک عبارت Boolean یا منطقی است که در یک براکت قرار می‌گیرد. UML 2.3 برای پشتیبانی از حالت‌ها و الگوریتم‌های پیچیده‌تر هر دوی آنها را عبارت یا موارد تضمینی می‌نامد. UML سعی می‌کند با ترکیب ساده‌ترین شکل حلقه for و while معنای عجیب حلقه UML 2.3 در صفحه 488 را که بیان زیر آن را شرح می‌دهد، معرفی کند:</p> <p>پس از انجام حداقل تعداد تکرار و نادرست بودن عبارت Boolean حلقه خاتمه می‌یابد.</p> <p>این عبارت با معنای معکوس در صفحه بعد همین مستند به این صورت زیر بیان شده است:</p> <p>حلقه تنها در صورتی ادامه می‌یابد که آن مشخصات در طول اجرا بدون توجه به حداقل تعداد تکرارهای مشخص شده در حلقه، درست ارزیابی شود.</p>	 <p>ممکن است حدس بزنیم که طبق UML 2.3، انتظار می‌رود، حلقه حداقل 5 و حداکثر 10 دفعه اجرا شود. اگر شرط کنترل [size<0] مقدار false شود، صرف نظر از حداقل تعداد تکرار مشخص شده، حلقه خاتمه می‌یابد. (پس چرا به آن عدد حداقل نیاز داریم که مشخص شود؟!)</p>
<p>عملگر تعاملی loop در این حالت هیچ مرز بی‌واسطه‌ای ندارد و مرز آن به واسطه ارزیابی یک عبارت تضمینی مشخص شده‌است، این Combined Fragment حداقل صفر دفعه و حداکثر تا زمانی‌که ارزیابی عبارت تضمینی true باشد تکرار خواهد شد.</p>	 <p>حلقه بالقوه بی‌نهایت اگر size<0 بشود خاتمه می‌یابد</p>
<p>عملگر تعاملی break نشان دهنده یک سناریوی شکست یا استثنایی است که به جای باقیمانده قطعه تعاملی محصور کننده عملگر مورد نظر، انجام می‌شود. UML اجازه می‌دهد تنها یک سطح، که به طور مستقیم بخش تعامل را در بر می‌گیرد، رها شود. اگر حلقه یا حلقه‌های تودرتو با Combined Fragment های مختلف وجود داشته باشد عملگر تعاملی break، می‌تواند واقعا آزاد دهنده باشد.</p> <p>توجه داشته باشید که عملگر تعاملی break می‌تواند جایگزین خوبی برای حالت قبلی باشد (حلقه بالقوه بی‌نهایت مشروط) با این تفاوت که در این حالت در هنگام خاتمه Combined Fragment مورد نظر که در این مثال یک عملگر تعاملی loop است، می‌توان قطعه تعاملی دیگری را نیز اجرا نمود.</p>	 <p>اگر $y > 0$ حلقه محصور شده، خاتمه می‌یابد</p>
<p>عملگر تعاملی par، اجرای بالقوه موازی رفتارهای عملوندهای یک Combined Fragment را تعریف می‌کند. تا زمانی که ترتیب اجرای هر عملوند بی‌اهمیت باشد، عملوندهای مختلف را می‌توان به هر ترتیبی در هم آمیخت.</p>	 <p>Google, Bing و Ask را به هر ترتیبی، احتمالاً موازی، جستجو کنید</p>
<p>Combined Fragment موازی برای موقعیت‌های رایجی که ترتیب رویدادها در یک Lifeline اهمیت نداشته باشد، یک تصویر خلاصه شده نمادین است. در یک Coregion از یک Lifeline که توسط براکت‌های افقی محدود شده است، همه قطعات دربرگیرنده، مستقیماً به عنوان عملوندهای جداگانه یک Combined Fragment موازی در نظر گرفته می‌شوند.</p>	

	<p>Google, Bing و Ask را به هر ترتیبی، احتمالاً موازی، جستجو کنید</p>
<p>عملگر تعاملی strict، ترتیب توالی دقیق و قطعی عملوندها در سطح اول Combined Fragment را تضمین می‌کند.</p>	 <p>Google, Bing و yahoo را دقیقاً به ترتیب جستجو کنید</p>
<p>عملگر تعاملی seq، ترتیب توالی ضعیف یا غیر قطعی مجموعه‌ای از عملوندها را با شرایط زیر محقق می‌کند:</p> <ul style="list-style-type: none"> • ترتیب رویدادهای هر یک از عملوندها در عملوند حفظ می‌شود. • رویدادهای مختلف در Lifeline های مختلف که در عملوندهای مختلف قرار دارند ممکن است به هر ترتیبی، محقق شوند. • رویدادهای یک Lifeline یکسان که در عملوندهای مختلف قرار دارند به گونه‌ای مرتب می‌شوند که رویدادهای عملوند اول قبل از عملوند دوم محقق می‌شود. <p>بنابراین، زمانی که عملوندها روی مجموعه‌های مجزا از شرکت‌کنندگان (Lifeline های مجزا) قرار دارند، عملگر تعاملی seq، به عملگر تعاملی par، تبدیل می‌شود. زمانی که عملوندها فقط روی یک شرکت‌کننده (Lifeline) قرار دارند، عملگر تعاملی seq به عملگر تعاملی strict، تبدیل می‌شود.</p>	 <p>احتمالاً همانطور که Google را جستجو می‌کنید؛ به موازات آن، Bing و yahoo را نیز جستجو کنید، اما قبل از Bing, yahoo را جستجو کنید</p>
<p>عملگر تعاملی critical یک Combined Fragment که یک منطقه بحرانی را نشان می‌دهد، تعریف می‌کند. یک منطقه بحرانی، منطقه‌ای است با عملوندهایی که توسط سایر رویدادها، (روی Lifeline های درون همان منطقه) قابل تعامل نیستند. این بدان معنی است که این ناحیه به صورت مستقل توسط قطعه محصور کننده، اجرا می‌شود و نمی‌توان آن را در هم آمیخت، به عنوان مثال، توسط عملگر تعاملی par</p> <p>مثالی که می‌توان برای این موضوع آورد، طراحی تراکنش‌ها است که حتماً به صورت مستقل و فارق از جریان اصلی، اجرا می‌شوند و الگوریتم آنها را نمی‌توان در جریان اصلی ترکیب کرد.</p>	 <p>متمدهای Add() یا remove() می‌توان به صورت موازی فراخوانی کرد، اما هر کدام باید به عنوان یک محدوده بحرانی اجرا شوند</p>
<p>عملگر تعاملی ignore به این معنی است که برخی از پیام‌ها در این Combined Fragment از دسترس خارج خواهند بود. این نوع پیام‌ها را می‌توان بی‌ارزش در نظر گرفت و اگر در اجرای واقعی ظاهر شوند به طور ضمنی نادیده گرفته می‌شوند.</p> <p>لیست پیام‌های نادیده گرفته شده، در مقابل نام عملگر Combined Fragment در یک جفت آکولاد "{" و "}" قرار می‌گیرند. عملیات ignore از نظر مفهومی معمولاً با عملیات دیگری مثل "اقدام به نادیده گرفتن" مشابه است.</p> <p>$ignore\{get, set\} \cong assert\ ignore\{get, set\}$</p>	 <p>در صورت وجود، پیام‌های get() و set() آنها را نادیده بگیرید</p>
<p>عملگر تعاملی consider، تعریف می‌کند که کدام پیام باید در این Combined Fragment در نظر گرفته شود، به این معنی که هر پیام دیگری نادیده گرفته می‌شود. لیست پیام‌های در نظر گرفته شده در مقابل نام عملگر Combined Fragment در یک جفت آکولاد "{" و "}" قرار می‌گیرد. عملیات consider معمولاً با عملیات دیگری مانند "اقدام به در نظر گرفتن" مشابه است.</p> <p>$consider\{add, remove\} \cong assert\ consider\{add, remove\}$</p>	 <p>فقط پیام‌های add() یا remove() را در نظر بگیرید، هر پیام دیگری را نادیده بگیرید</p>
<p>عملگر تعاملی assert، یک Combined Fragment تضمین را نشان می‌دهد که در آن فرض بر این است که جریان عملوند تضمین تنها جریان معتبر است (باید با طراحی صحیح سیستم این حالت را نشان دهید). سایر جریان‌ها منجر به یک مسیر نامعتبر می‌شوند. در این نوع Combined Fragment حتماً باید شرط اعتبار جریان ذکر شود که در مثال ذکر شده شرط زیر می‌باشد:</p> <p>{t==complete}</p>	 <p>پیام Commit() باید در این نقطه رخ دهد و سپس وضعیت با یک مقدار ثابت، ارزیابی شود</p>

<p>عملگر تعاملی neg یک Combined Fragment است که جریان‌های نامعتبر را تعریف می‌کند. جریان‌های نامعتبر، جریان‌هایی هستند که در هنگام از کار افتادن سیستم رخ می‌دهند. تمام Combined Fragment که neg نیستند معتبر در نظر گرفته می‌شوند، به عبارت دیگر، آنها جریان‌هایی را توصیف می‌کنند که معتبر هستند و باید امکان پذیر باشند.</p>	 <p>اگر پیام timeout دریافت کنیم، به این معنی است که سیستم از کار افتاده است</p>
Interaction Use	
<p>عملگر تعاملی ref یک Combined Fragment است که امکان استفاده (یا فراخوانی) تعامل دیگری را در داخل جریان تعاملی موجود، فراهم می‌کند. sequence diagram های بزرگ و پیچیده را می‌توان با استفاده از عملگر تعاملی ref، ساده کرد. استفاده مجدد از برخی تعاملات بین تعاملات دیگر رایج است. استفاده مجدد از تعامل به شکل یک Combined Fragment با عملگر ref نشان داده می‌شود.</p>	 <p>شیء Web Customer و Online Bookshop برای انجام عمل پرداخت یا همان Checkout از مرجع یا قطعه تعاملی ref استفاده می‌کنند که ممکن است در Combined Fragment محاط کننده وجود نداشته باشد</p>
<p>نگارش استفاده از عملگر تعاملی ref به صورت زیر است:</p> <pre> interaction-use ::= [attribute-name '='] [collaboration-use '.'] interaction-name [io-arguments] [':' return-value] io-arguments ::= '(' io-argument [',' io-argument]* ')' io-argument ::= in-argument 'out' out-argument </pre> <p>تفسیر مثال مورد نظر بر اساس نگارش بالا به شکل زیر است:</p> <pre> [attribute-name '='] →→→ sc.user [collaboration-use '.'] interaction-name →→→ Login [io-arguments] →→→ (login_id, password) [':' return-value] →→→ user </pre>	 <p>از قطعه تعاملی ورود به سیستم برای احراز هویت کاربر استفاده می‌شود و نتیجه به ویژگی کاربر کنترلر سایت بازگردانده می‌شود</p>

Lifeline:

نماینده شیء که در تعامل قرار دارد



Message:

ارتباط خاصی بین اشیاء فرستنده و گیرنده یک نمودار توالی که مبتنی بر فراخوانی متد و یا سیگنالی است.

Message $\stackrel{\text{def}}{=}$ Accept

Message $\stackrel{\text{def}}{=}$ getBalance (accountNumber) : balance

پارامترهای یک پیام می‌تواند یکی از موارد زیر باشد:

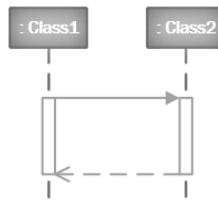
- یکی از مشخصات یا فیلدهای شیء ارسال کننده پیام
- یکی از مشخصات عمومی قطعه تعاملی که در اکثر مواقع به آن پارامتر قطعه تعاملی می‌گویند
- یکی از مشخصات یا متغیرهای صاحب تعامل که در واقع متدی است که فراخوانی از داخل آن صورت می‌گیرد
- یک مقدار ثابت عددی یا رشته‌ای
- یک مقدار فراگیر (مثل: man یا Validated)

فراخوانی متد و یا سیگنال شامل موارد زیر است:

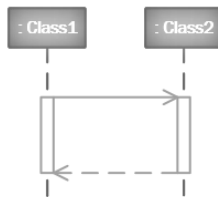
- فراخوانی متد خارجی (Call external method)
- بازگرداندن نتیجه اجرای متد (Return result executed method)
- فراخوانی متد ایجاد شیء (Call object creation method)
- برچیدن شیء (Destruct object)
- فراخوانی متد درونی (Call internal method)

نوع فراخوانی متد و یا سیگنال شامل موارد زیر است:

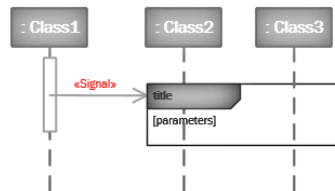
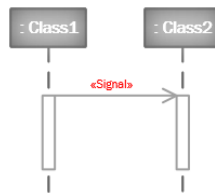
- فراخوانی همزمان (synchronous call)



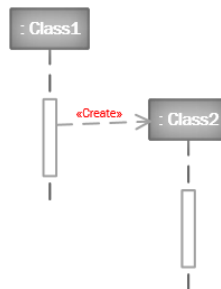
- فراخوانی ناهمزمان (asynchronous call)



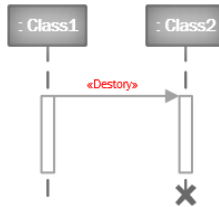
- سیگنال ناهمزمان (asynchronous signal)



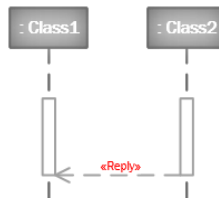
- ایجاد (create)



- حذف (delete)

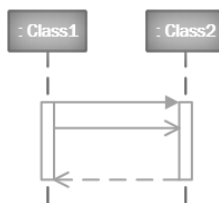


- پاسخ (reply)

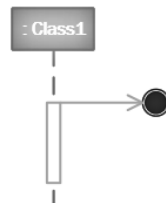


پیام‌ها از نظر اشیاء فرستنده و گیرنده می‌توانند حالت‌های زیر را داشته باشند:

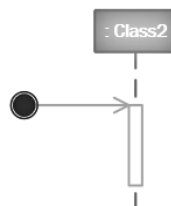
- پیام کامل (complete message)

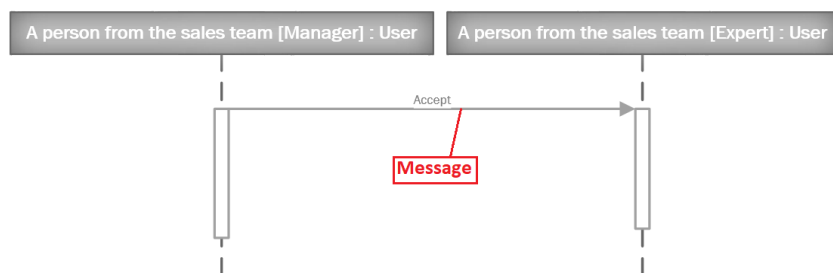


- پیام گم شده (lost message)



- پیام پیدا شده (found message)





Lifeline Occurrence:

رویدادهای مربوط به تعاملات یک شیء شامل موارد زیر است:

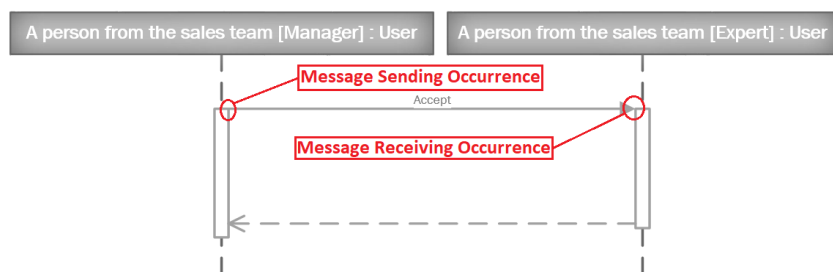
- رویداد آغاز کار مربوط به شیء (Start Occurrence)
- رویداد پایان کار مربوط به شیء (Finish Occurrence)
- رویداد اجرایی کار مربوط به شیء (Execution Occurrence)
- رویداد ایجاد شدن شیء (Creation Occurrence)
- رویداد برچیده شدن شیء (Destruction Occurrence)



Message Occurrence:

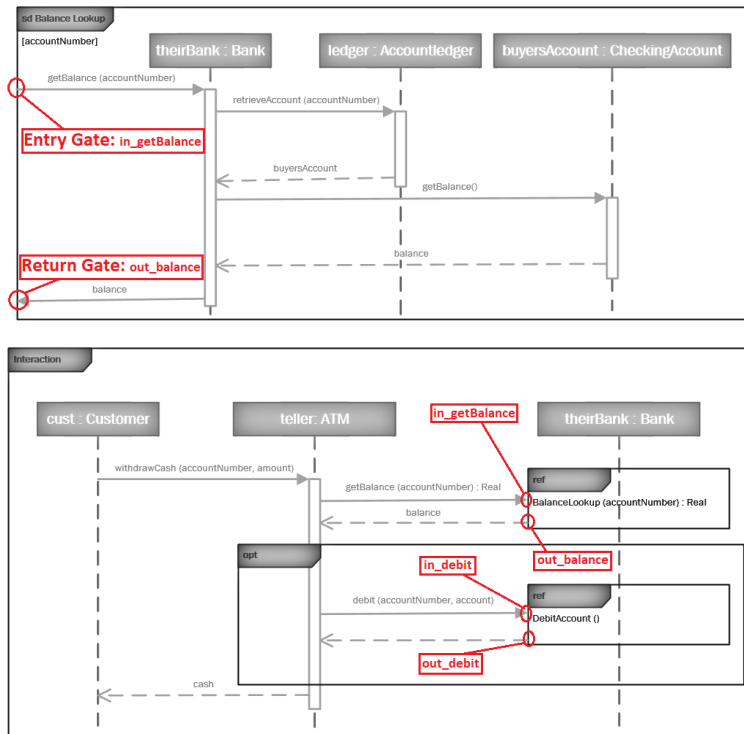
رویدادهای مربوط به پیام که شامل موارد زیر است:

- رویداد ارسال پیام (Message Sending Occurrence)
- رویداد دریافت پیام (Message Receiving Occurrence)



Gate:

ارتباط خاصی بین نمودارهای توالی یا قطعات تعاملی درون یک نمودار توالی، که تقریباً مشابه پیام، مبتنی بر فراخوانی متد است



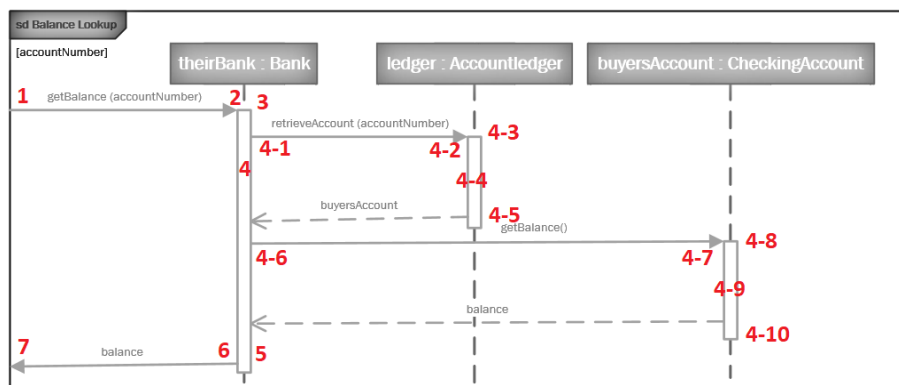
Trace:

مسیر ردیابی یا جریان پیگیری رویدادها است که با قالب زیر تعریف می‌شود:

<event1, event2, ... , eventN>

رویدادهای جریان همانطور که قبلاً توضیح داده شد شامل موارد زیر است:

- رویداد ارسال پیام (Message Sending Occurrence)
- رویداد دریافت پیام (Message Receiving Occurrence)
- رویداد آغاز کار مربوط به شیء (Start Occurrence)
- رویداد پایان کار مربوط به شیء (Finish Occurrence)
- رویداد اجرایی کار مربوط به شیء (Execution Occurrence)
- رویداد ایجاد شدن شیء (Creation Occurrence)
- رویداد برچیده شدن شیء (Destruction Occurrence)



اگر بخواهیم مسیر ردیابی این نمودار را بنویسیم به شکل زیر می‌شود:

Trace $\stackrel{\text{def}}{=} \langle e1, e2, e3, e4-1, e4-2, e4-3, e4-4, e4-5, e4-6, e4-7, e4-8, e4-9, e4-10, e5, e6, e7 \rangle$

که به زبان طبیعی به شکل زیر نوشته می‌شود:

- **e1:** ارسال پیام درخواست، بازیابی اطلاعات تراز حساب مربوط به شماره حساب مورد نظر از مقتضای به بانک مربوطه
- **e2:** دریافت پیام درخواست، بازیابی اطلاعات تراز حساب مربوط به شماره حساب مورد نظر توسط بانک مربوطه
- **e3:** آغاز بررسی پیام دریافت شده توسط بانک مربوطه (e3)
- **e4-1:** ارسال پیام درخواست، بازیابی اطلاعات حساب مربوط به شماره حساب مورد نظر، توسط بانک مربوطه به دفتر حساب‌ها
- **e4-2:** دریافت پیام درخواست، بازیابی اطلاعات حساب مربوط به شماره حساب مورد نظر، توسط دفتر حساب‌ها
- **e4-3:** آغاز بررسی پیام دریافت شده توسط دفتر حساب‌ها
- **e4-4:** استخراج اطلاعات حساب مربوط به شماره حساب مورد نظر توسط دفتر حساب‌ها (زمانبر است)
- **e4-5:** پایان بررسی پیام دریافت شده توسط دفتر حساب‌ها
- **e4-6:** ارسال پیام درخواست، بازیابی اطلاعات تراز حساب مربوط به شماره حساب مورد نظر از بانک مربوطه به بررسی کننده حساب‌ها
- **e4-7:** دریافت پیام درخواست، بازیابی اطلاعات تراز حساب مربوط به شماره حساب مورد نظر توسط بررسی کننده حساب‌ها
- **e4-8:** آغاز محاسبه تراز حساب مربوط به شماره حساب مورد نظر توسط بررسی کننده حساب‌ها
- **e4-9:** محاسبه تراز حساب مربوط به شماره حساب مورد نظر توسط بررسی کننده حساب‌ها (زمانبر است)
- **e4-10:** پایان محاسبه تراز حساب مربوط به شماره حساب مورد نظر توسط بررسی کننده حساب‌ها
- **e5:** پایان بررسی پیام دریافت شده توسط بانک مربوطه
- **e6:** ارسال پیام حاصل، محاسبه تراز حساب مربوط به شماره حساب مورد نظر از بانک مربوطه به مقتضای
- **e7:** دریافت پیام حاصل، محاسبه تراز حساب مربوط به شماره حساب مورد نظر توسط مقتضای

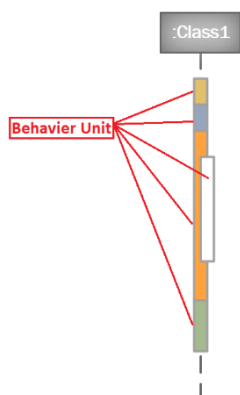
همانطور که مشخص است، شیء مقتضای در این نمودار یک شیء مفهومی است و از طریق درگاه‌های مشخص شده با این نمودار ارتباط دارد.

یا به صورت خلاصه شده:

Trace $\stackrel{\text{def}}{=} \langle e1, e2, e3, e4, e5, e6, e7 \rangle$

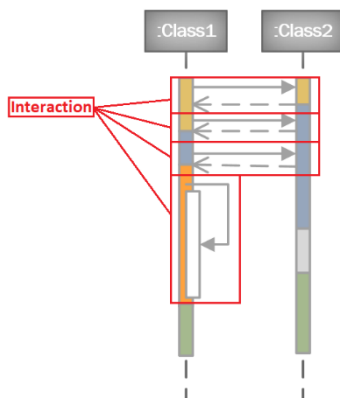
Behavior Unit:

یک واحد رفتاری، در واقع یک جمله دستوری است که فقط یک فعل عملی و اصلی دارد. این فعل، نماینده عملیاتی است که در این جمله به عنوان یک واحد رفتاری معرفی می‌شود.



Interaction:

یک تعامل، در واقع نماینده مجموعه‌ای از واحد رفتاری خاص و الزامی است که بر مبادله قابل مشاهده اطلاعات بین عناصر قابل اتصال، متمرکز است. تعاملات برای انتقال اطلاعات بین این عناصر از پیام‌ها استفاده می‌کنند. این پیام‌ها از منطری دیگر رویدادهای یک تعامل محسوب می‌شوند و در معادله مسیر ردیابی و پیگیری معرفی می‌شوند.



Interaction Fragment:

یک قطعه تعاملی، متناظر با یک واحد تعاملی است که می‌تواند در یک نمودار به صورت تجزیه‌شده و یا مبهم نمایش داده شود و می‌تواند بر اساس شاخص‌های زیر طبقه‌بندی شده باشد:

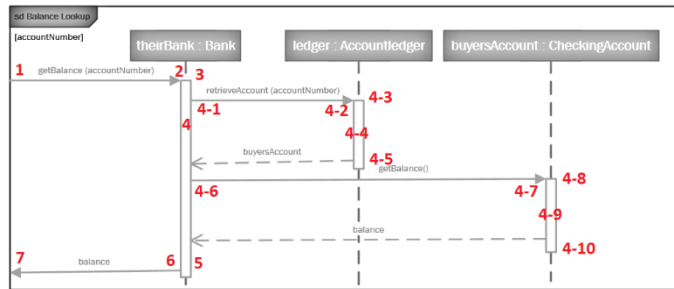
- رخدادها

Interaction Fragment $\stackrel{\text{def}}{=}$ <e1, e2, e3, e4, e5, e6, e7>

Interaction Fragment $\stackrel{\text{def}}{=}$ <e2, e3, e4-1, e4-2, e4-3, e4-4, e4-5>

Interaction Fragment $\stackrel{\text{def}}{=}$ <e4-1, e4-2, e4-3, e4-4, e4-5>

Interaction Fragment $\stackrel{\text{def}}{=} \langle e4-6, e4-7, e4-8, e4-9, e4-10 \rangle$



• اجراها

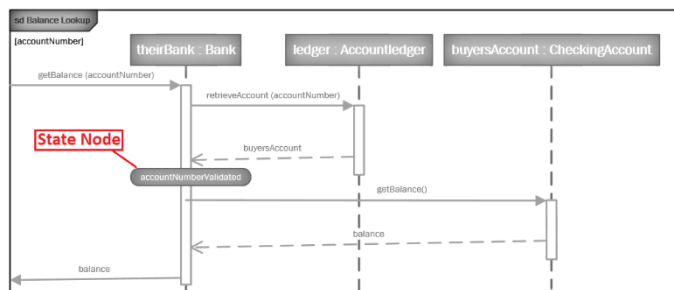
Interaction Fragment $\stackrel{\text{def}}{=} \langle e4, e4-4 \rangle$

• وضعیت‌های ثابت

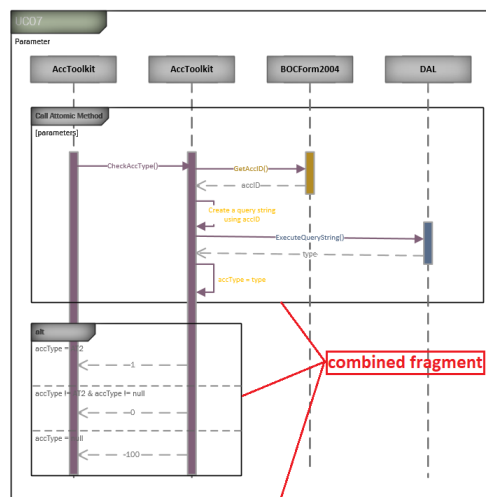
Interaction Fragment $\stackrel{\text{def}}{=} \{t==completed\}$

Interaction Fragment $\stackrel{\text{def}}{=} \{accountNumberValidated\}$

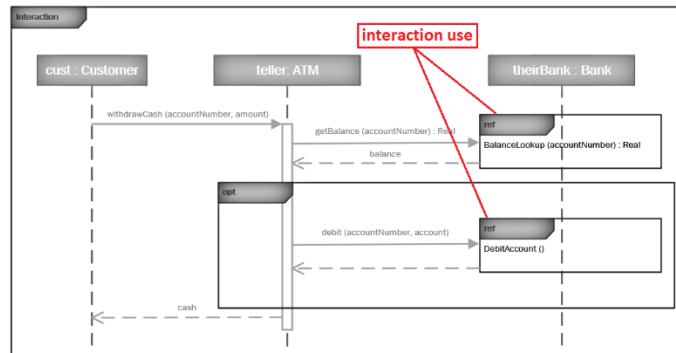
Interaction Fragment $\stackrel{\text{def}}{=} \{accountNumberRejected\}$



• قطعات ترکیبی



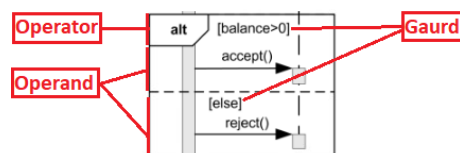
- تعاملات کاربردی



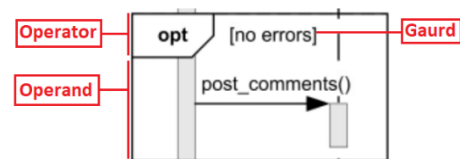
Combined Fragment:

یک قطعه ترکیبی، یک قطعه تعاملی است که با جزئیات کامل، تعاملات بین اشیاء تعامل را نمایش می‌دهد. این قطعه تعاملی دارای عملگر و عملوند است. عملوندهای آن می‌توانند مجموعه‌ای از محدودکننده‌ها باشند و عملگرهای آن در زیر بیان شده است:

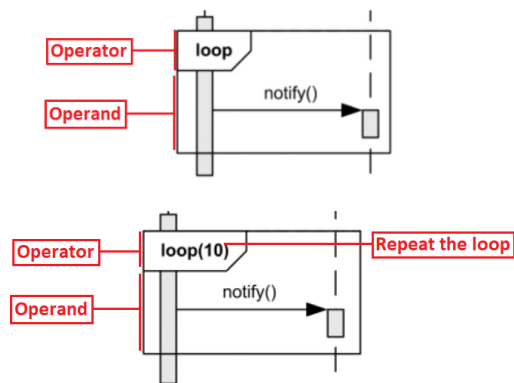
- **alt** (محدوده معرفی رفتارهای جایگزین)

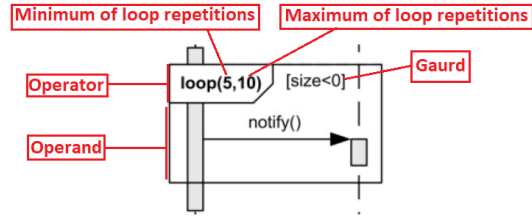


- **opt** (محدوده معرفی گزینه‌ها)

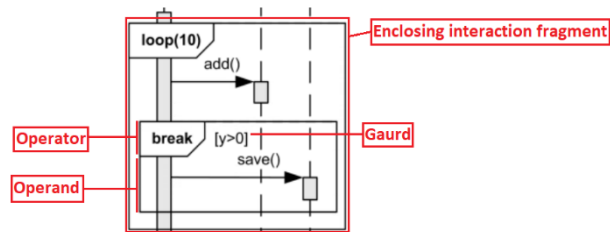


- **loop** (محدوده معرفی تکرار رفتار)

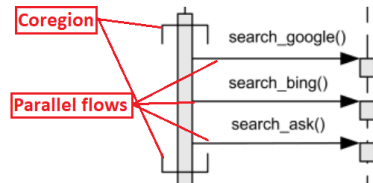
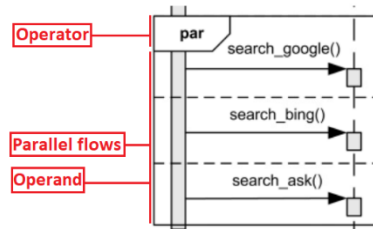




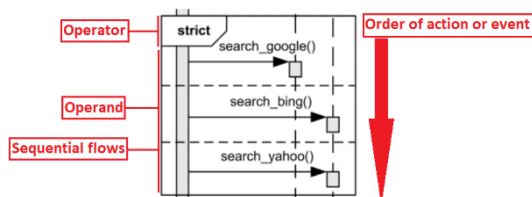
- **break** (محدوده معرفی شکست رفتار)



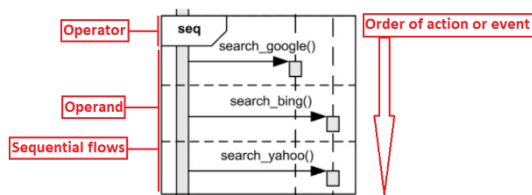
- **par** (محدوده معرفی رفتارهای موازی)



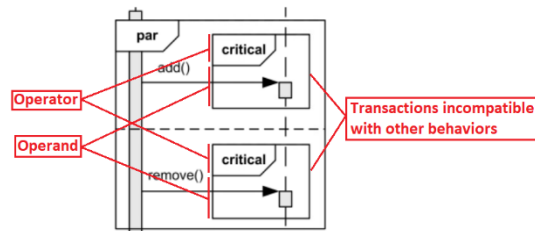
- **strict** (محدوده معرفی رفتارهایی که توالی قطعی دارند و رفتار بعدی پس از خاتمه رفتار قبلی، شروع می شود)



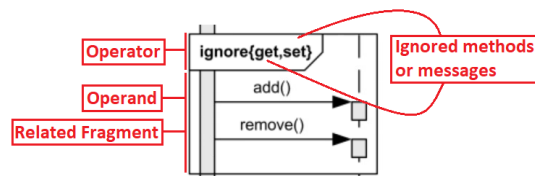
- **seq** (محدوده معرفی رفتارهایی که توالی غیر قطعی دارند و رفتار بعدی می تواند قبل از خاتمه رفتار قبلی، شروع شود)



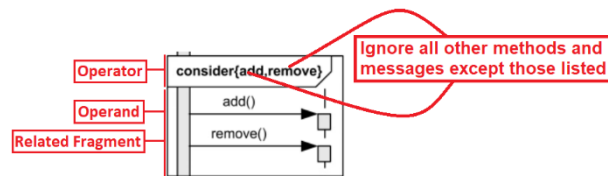
- **critical** (محدوده معرفی رفتارهای بحرانی که در هیچ ترتیب توالی و یا موازی با رفتارهای دیگر قابل ترکیب نیستند)



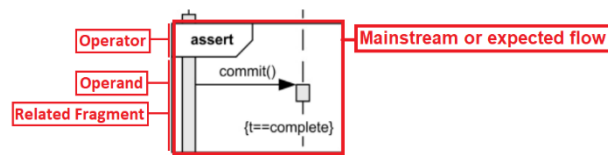
- **ignore** (این محدوده مشخص می کند از برخی پیام ها، عملیات یا رفتارهای موجود، برای نمایش در نمودار فعلی صرف نظر می شود)



- **consider** (محدوده معرفی فرضیات)



- **assert** (محدوده معرفی مسیر مورد انتظار یا مسیر اصلی کسب و کار در جهت خلق ارزش)



- **neg** (محدوده معرفی وضعیت های نامعتبر)

