Diagram sekwencji - przebiegu

UML 2.0 zawiera 4 rodzaje diagramów interakcji modelujące interakcje zachodzące w czasie działania systemu pomiędzy jego częściami, które to części wchodzą w skład **widoku logicznego modelu:**

- **Diagramy sekwencji** - opisują interakcje pomiędzy częściami systemu w postaci sekwencji komunikatów wymienianych między nimi
- **Diagramy komunikacji** - specyfikują strukturalne związki pomiędzy biorącymi udział w interakcji częściami oraz wymianę komunikatów pomiędzy tymi instancjami.
- **Diagramy czasowe** - reprezentują na osi czasu zmiany dopuszczalnych stanów, jakie może przyjmować uczestnik w interakcji
- **Przeglądowe diagramy interakcji** - udostępniają wysokiego poziomu widok wzajemnej współpracy kilku interakcji wykorzystujących w celu implementacji pewnej części systemu, na przykład danego przypadku użycia

Diagram sekwencji

Obrazuje kolejność w czasie wysyłania komunikatów pomiędzy różnymi obiektami w systemie

Uczestnicy na diagramie sekwencji

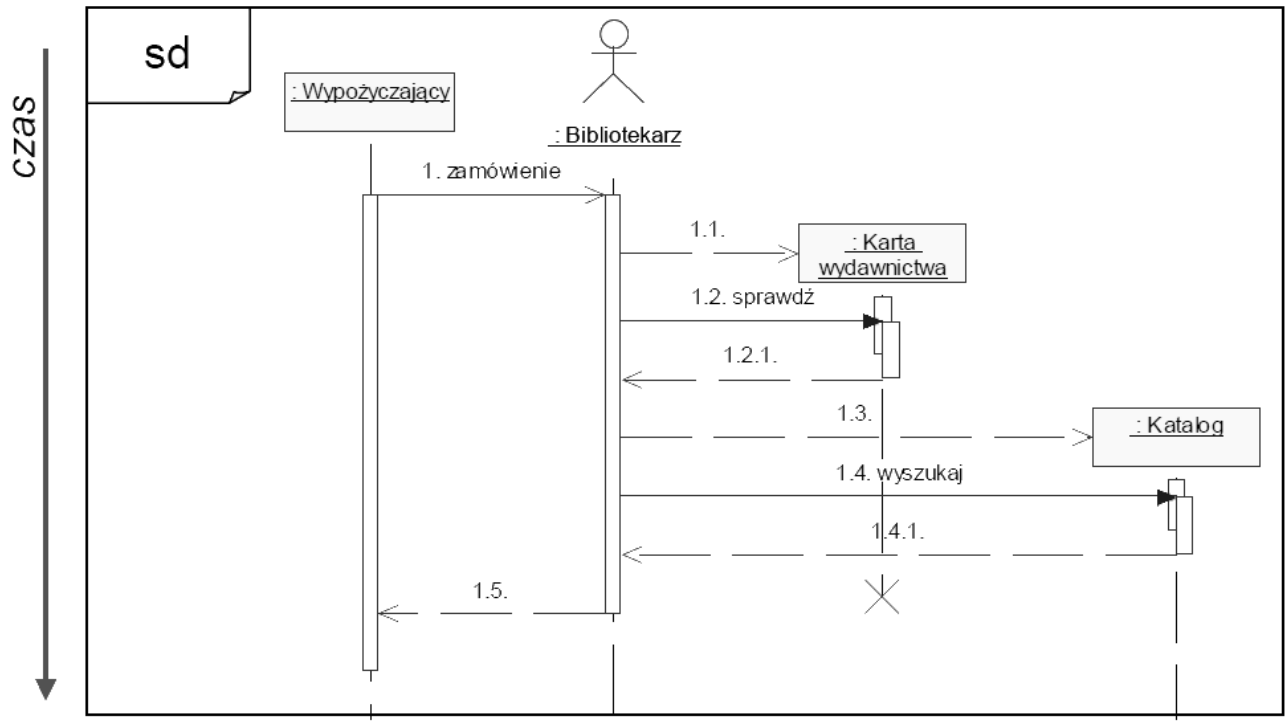
Standardowy format:

Nazwa `[ selektor]`:*nazwa klasy ref dekompozycja*

To jakie elementy z danego formatu zostaną wybrane dla określonego uczestnika, zależeć będzie od tego, jaki rodzaj informacji o nim mamy w danym momencie.

W jaki sposób zrozumieć części nazwy uczestnika

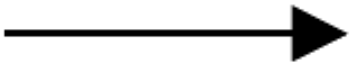
Przykładowa nazwa uczestnika	**Opis**
`Admin`	Część nosi nazwę admin, jednak w tym momencie nie ma przypisanej klasy.
`:ContentManagementSystem`	Klasa uczestnika nosi nazwę ContentManagementSystem, jednak w danym momencie część nie ma swojej własnej zawy
`admin:Administrator`	Jest to część nosząca nazwę admin, utworzona na podstawie klasy Administartor.
`eventHandlers[2]:EventHandler`	Jest to część zapisana jako element tablicy o indeksie 2, utworzona na podstawie klasy EventHandler.
`:ContentManagementSystem refcmsInteraction`	Uczestnik utworzony został na podstawie klasy ContentManagementSystem. Istnieje również inny diagram interakcji o nazwie cmsInteraction, przedstawiający wewnętrzne działanie danego uczestnika.



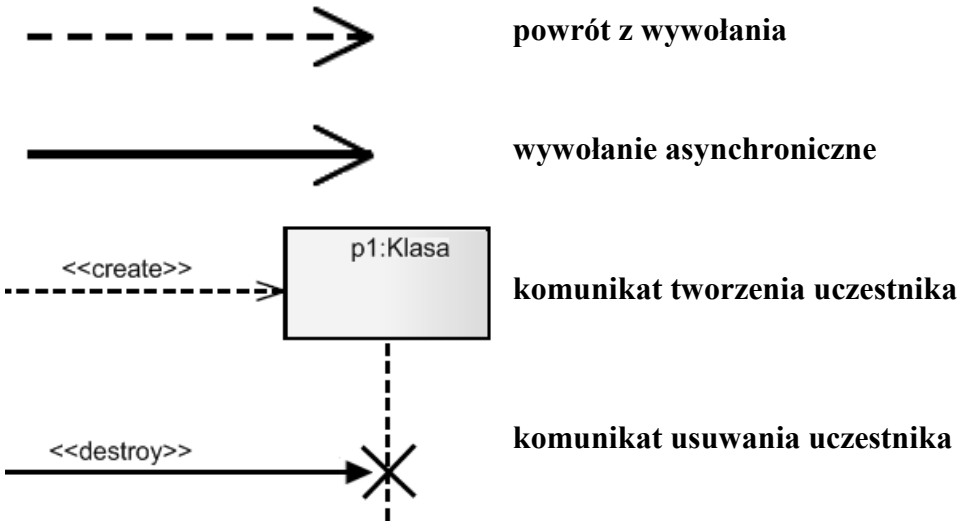
Diagramy sekwencji (ang. *sequence diagrams*) intuicyjnie prezentują kolejność wywołań operacji, przepływ sterowania pomiędzy obiektami oraz szablon realizowanego algorytmu. Pomijają natomiast całkowicie aspekt dostępu i operacji na danych, związany z komunikacją.

- Uczestnikami diagramów sekwencji są obiekty, opisane nazwą obiektu i jego klasą, które wymieniają między sobą komunikaty.
- Diagram sekwencji jest zapisany w prostokącie oznaczonym operatorem **sd** (od angielskiej nazwy diagramu) i składa się z **pionowych linii życia (ang. lifelines)** poszczególnych obiektów uczestniczących w interakcji oraz wymienianych między nimi komunikatów (wywołań operacji). Białe prostokąty umieszczone na linii życia obiektu oznaczają, że obiekt jest zajęty wykonywaniem pewnej czynności (natomiast nie mają bezpośredniego związku z istnieniem obiektu).
- Czas jest reprezentowany w postaci pionowej osi diagramu. Linia życia obiektu to czas, w którym konkretna instancja obiektu jest w stanie przyjmować komunikaty i wysyłać je. Innymi słowy, obejmuje ona czas istnienia obiektu w systemie.
- Obiekt jest tworzony poprzez wysłanie do niego komunikatu-konstruktora (Bibliotekarz tworzy obiekt klasy Karta Wydawnictwa), natomiast niekoniecznie jest fizycznie usuwany na końcu linii życia ? raczej przestaje być istotny. Fizycznie usunięcie obiektu można wprost oznaczyć jako znak X na linii życia (na przykład obiekt Karta wydawnictwa).

Rodzaje komunikatów

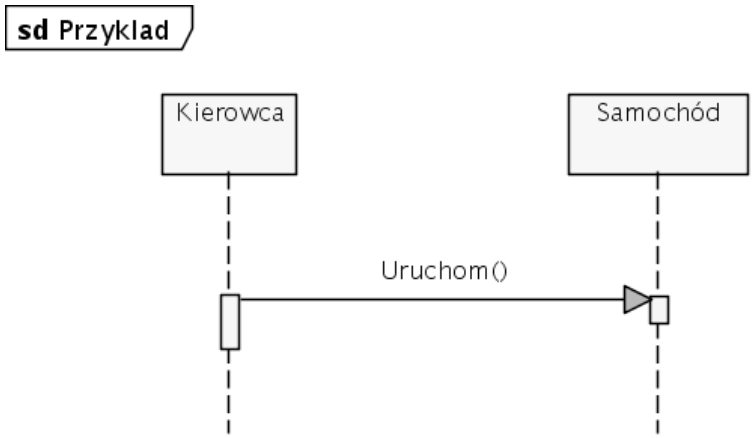


wywołanie procedury



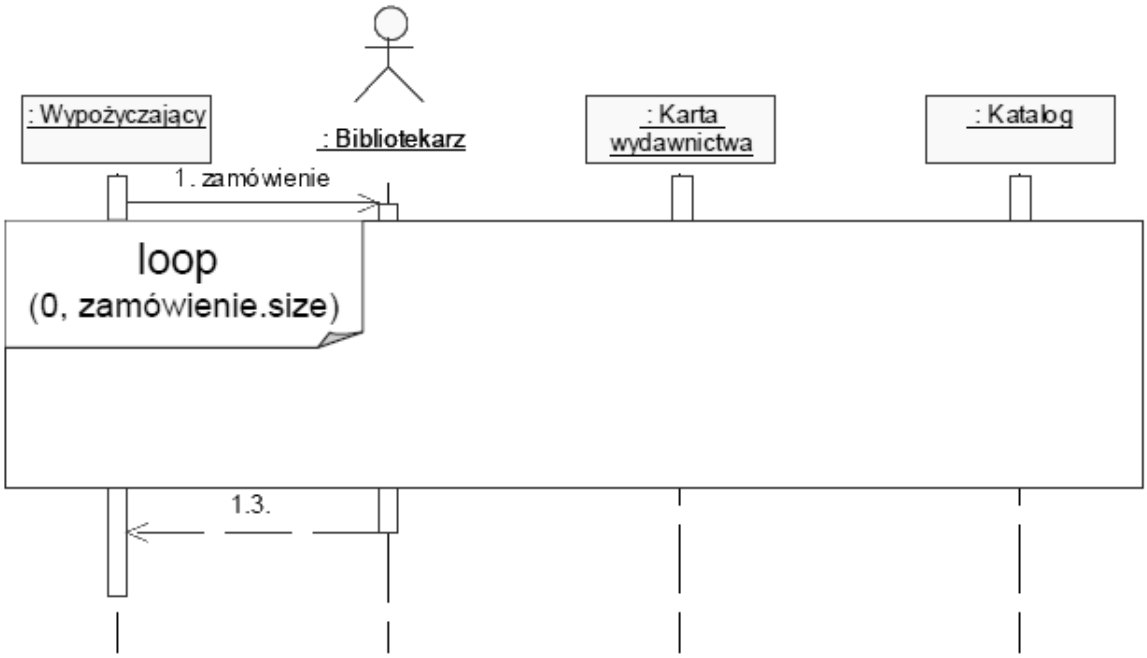
Komunikat to forma kontaktu pomiędzy obiektami, której efektem ma być podjęcie przez docelowy obiekt pewnej akcji. Otrzymanie komunikatu przez obiekt wiąże się z wykonaniem przez niego jego własnego kodu lub wysłaniem kolejnego komunikatu do innego obiektu w celu wykonania przez niego pewnej akcji.

Komunikaty w UML są reprezentowane przez strzałki łączące linie życia poszczególnych obiektów. Każdy komunikat wewnątrz interakcji opatrzony jest kolejnym numerem, co pozwala na łatwe śledzenie jej przebiegu. Istnieją trzy podstawowe komunikaty, jakie mogą zostać wymienione pomiędzy obiektami: **wywołanie procedury, powrót z niej oraz wywołanie asynchroniczne.**



Bloki - fragmenty wyodrębnione

Blok definiuje grupę komunikatów wspólnie posiadającą pewną właściwość.



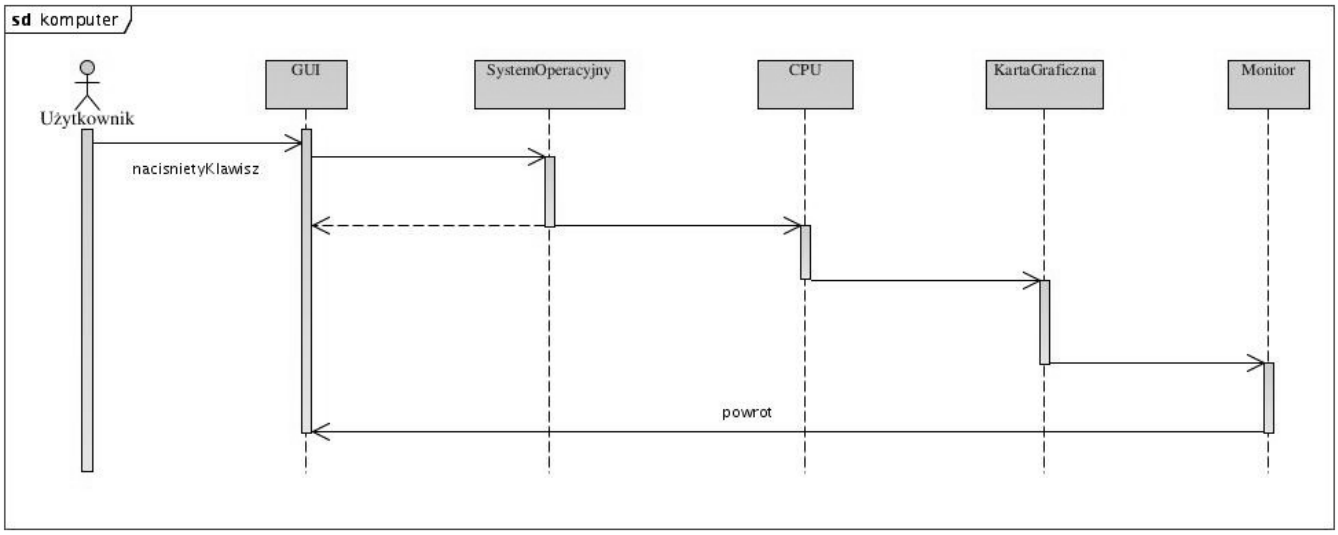
Bardzo często zachodzi konieczność wskazania specjalnej własności pewnej części interakcji, np. oznaczenie sekcji krytycznej czy zwyczajnej pętli. Na diagramach sekwencji taką grupę operacji obejmuje się prostokątem, w którego lewym górnym narożniku, w pięciokącie umieszcza się słowo kluczowe lub opis określający znaczenie danego bloku (tzw. operator interakcji)

Operatory interakcji

- **alt** (od *alternative*) - określający warunek wykonania bloku operacji, odpowiadający instrukcji *if-else* ; warunek umieszcza się wówczas wewnątrz bloku w nawiasach kwadratowych
- **opt** (od *optional*) - reprezentujący instrukcję *if* (bez *else*)
- **par** (od *parallel*) - nakazujący wykonać operacje równolegle
- **critical** - blok atomowy, oznaczający obszar krytyczny
- **loop** - definiujący pętlę typu *for* (o określonej z góry liczbie iteracji) lub *while* (wykonywanej dopóki pewien warunek jest prawdziwy)
- **break** - wykonanie fragmentu i zakończenie interakcji
- **seq** - słaba sekwencja (podobnie do współbieżności) dotyczy zdarzeń z kilku linii
- **ignore/consider** - ignore(komunikat1, komunikat2, ...) oznacza, że na diagramie nie pokazano wymienionych komunikatów, choć mogą wystąpić. Consider = odwrotnie

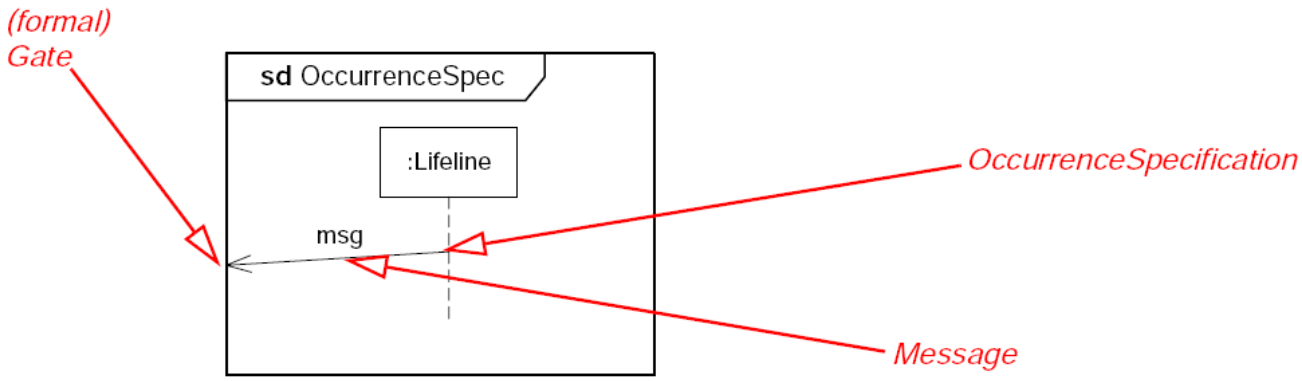
Aktorzy na diagramach interakcji

Po lewej stronie diagramu można dodawać aktora, inicjującego dany ciąg działań. Można również narysować jego linię życia



Dekompozycja

- W celu upraszczania skomplikowanych diagramów interakcji wprowadzono jeszcze jeden mechanizm składania diagramów.
- Uczestnik (obiekt) na diagramie kooperacji sam może reprezentować diagram kooperacji. W takim wypadku po nazwie stosuje się: `ref nazwa_diagramu`
- Komunikaty przychodzące do zdekomponowanej linii życia, jak również z niej wychodzące traktowane są jak bramki. Bramki te muszą mieć odpowiedniki na diagramie do którego się odnosimy.
- Domyślnie nazwa bramki oparta jest na kierunku (in, out) i przesyłanym komunikacje (np. out_weryfikacja). Można jednak bramkom nadawać inne nazwy



Techniki tworzenia diagramu sekwencji

- Ustalenie otoczenia (operacja, scenariusz PU itp.)
- Zidentyfikowanie obiektów biorących udział w operacji (po lewej umieszczane obiekty najważniejsze)
- Narysowanie linii życia obiektów (jeśli są tworzone i niszczone - dodanie odpowiednich komunikatów)

- Dodanie komunikatu inicjującego, kolejne komunikaty pod nim (dodanie parametrów do komunikatów, tam gdzie potrzeba)
- Dodanie aktywacji (ośrodka sterowania)
- Dodawanie ograniczeń

Zasady i porady

- Nazywaj aktorów w sposób spójny z diagramami przypadków użycia
- Nazywaj obiekty/klasy w sposób spójny z diagramami klas
- Po lewej stronie umieszczaj aktorów, którzy są ludźmi lub organizacjami
- Po prawej stronie umieszczaj aktorów, którzy są innymi systemami reagującymi na działanie modelowanego systemu
- Po lewej stronie umieszczaj aktorów, którzy są systemami inicjującymi interakcją z modelowanym
- Aktor może mieć tą samą nazwę co klasa
- Jeżeli nie jest to niezbędne - nie umieszczaj niszczenia (destroy) obiektu
- Umieszczaj nazwę obiektu jeżeli występuje odwołanie do niej w komunikacie
- Umieszczaj nazwę obiektu jeżeli istnieje wiele obiektów tej samej klasy
- Jeżeli umieszczono kilka obiektów o tej samej nazwie, muszą pochodzić z różnych klas
- Przy aktorach i obiektach/klasach umieszczaj stereotypy informujące o warstwie systemu, w której one działają
- Umieszczaj najważniejsze komunikaty (zwykle)
- Jeżeli komunikat jest wysyłany do obiektu/klasy, który jest implementowany jako składnik oprogramowania (klasa, interfejs, komponent), nazywaj komunikat z użyciem składni języka programowania
- Jeżeli komunikat wymaga przekazania parametrów, podaj ich nazwy, a nie same typy danych
- Komunikaty wychodzące od aktorów, którzy są osobami lub organizacjami, nazywaj w sposób opisowy (zdaniem)
- Komunikaty przesyłane do klas (nie obiektów) implementowane są jako metody statyczne
- Przy włączaniu innych przypadków użycia do aktualnie modelowanego stosuj komunikaty ze stereotypem << include >>
- Komunikat tworzący obiekt oznaczaj stereotypem << create >>
- Aktywacja nie jest obowiązkowym elementem linii życia, ale ułatwia zrozumienie diagramu - podkreśla czas trwania danej operacji