Podstawy modelowania w języku UML

dr hab. Bożena Woźna-Szcześniak, prof. UJD

Uniwersytet Humanistyczno-Przyrodniczy im. Jana Długosza w Częstochowie

Wykład 5

Plan wykładu

Diagram sekwencji

Diagramy komunikacji

Diagram sekwencji - wprowadzenie I

- **Diagram sekwencji** (ang. sequence diagram) jest jednym z czterech diagramów interakcji (ang. interaction diagram). Pozostałe to:
  - diagram komunikacji (ang. communication diagram),
  - diagramy czasowe (ang. timing diagram),
  - przeglądowe diagramy interakcji (ang. interaction overview diagrams).
- **Diagramy sekwencji** posiadają dwa wymiary:
  - wymiar pionowy - reprezentuje czas,
  - wymiar poziomy - reprezentuje różne obiekty (kolejność obiektów nie ma znaczenia).
- Orientację wymiarów można zmienić, tzn. :
  - wymiar pionowy - reprezentuje różne obiekty
  - wymiar poziomy - reprezentuje czas.

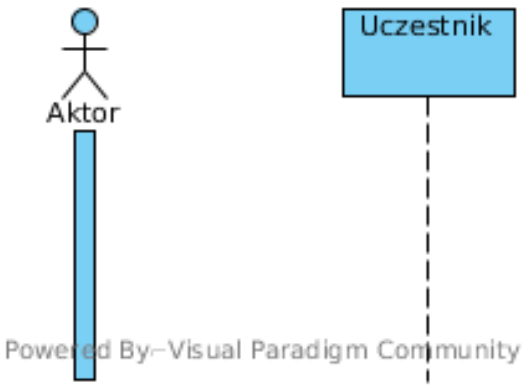
Diagram sekwencji - wprowadzenie II

- **Diagram sekwencji** ilustruje kolejność w czasie wysyłania komunikatów pomiędzy różnymi uczestnikami systemu; **Uwaga !** nie jest istotna rzeczywista miara czasu.
- Dla systemów zależnych od czasu, czas można reprezentować w pewnej mierzalnej skali.
- Diagramy sekwencji przydatne są do pokazywania uczestników, którzy komunikują się z innymi uczestnikami oraz do pokazywania widomości wywołujacych komunikację.
- Diagramy sekwencji rysuje się dla jednego przypadku użycia, przez co pokazuje się jak ten przypadek realizowany jest przez system.

Diagram sekwencji - uczestnicy i aktorzy I

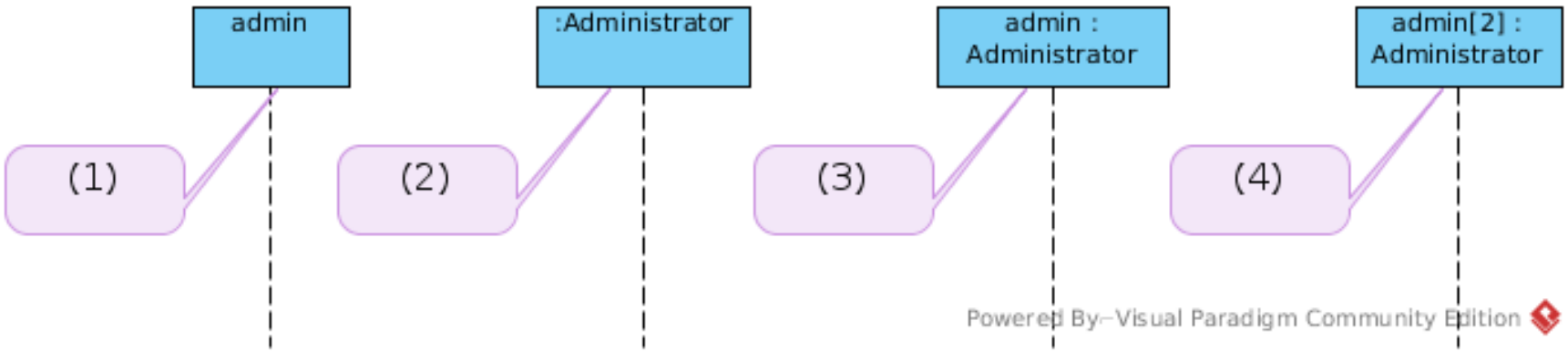
- Podstawowymi składowymi diagramów sekwencji są **uczestnicy** wymieniający między sobą komunikaty oraz **aktorzy** inicjujący komunikację.
- **Uczestnicy** są zaznaczani w postaci prostokątów z wpisaną wewnątrz nazwą uczestnika komunikacji.
- **Aktorzy** są zaznaczani tak samo jak w diagramach przypadków użycia.
- Każdy uczestnik posiada przerywaną linię reprezentująca jego *linię życia (czasu)* (ang. lifeline).
- Każdy aktor posiada pionowy słupek reprezentująca jego *linię życia (czasu)*.

Diagram sekwencji - uczestnicy i aktorzy II



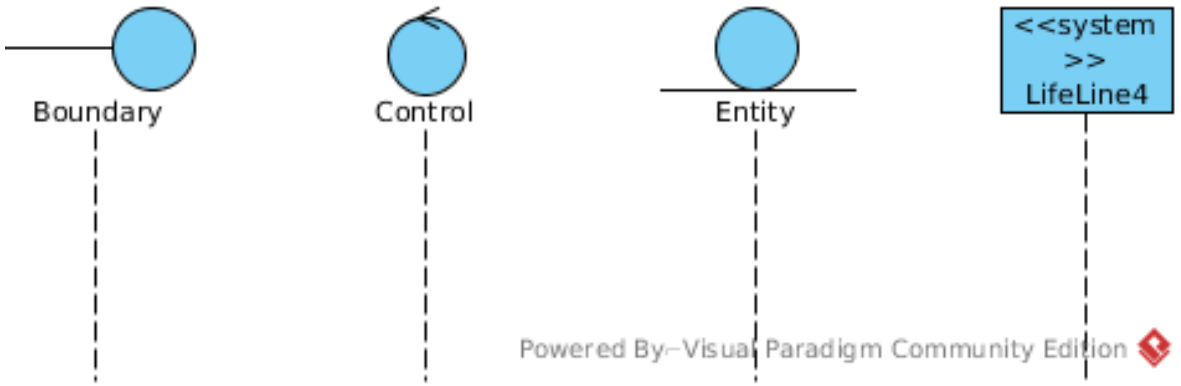
Rysunek: Aktor i uczestnik.

Diagram sekwencji - nazwy uczestników I



1. Uczestnik nosi nazwę **admin**, ale nie ma przypisanej klasy.
2. Klasa uczestnika nosi nazwę **Administrator**, ale uczestnik nie ma swojej własnej nazwy.
3. Uczestnik nosi nazwę **admin** i utworzony jest na podstawie klasy **Administartor**.
4. Uczestnik jest elementem tablicy o indeksie 2, utworzonej na podstawie klasy **Administartor**.

Diagram sekwencji - robustness diagrams



- Diagram sekwencji może mieć linię życia z symbolem reprezentującym elementy tzw. *robustness diagrams*: elementy brzegowe (ang. boundary), element sterowania (ang. control) oraz elementów jednostki (ang. entity).

Robustness diagrams I

Diagramy typu robustness są w zasadzie uproszczonymi diagramami komunikacji UML, które wykorzystuje symbole graficzne do reprezentacji:

- **aktorów** - to samo znaczenie co w UML
- elementów brzegowych (ang. **boundary elements**) zwanych również **interfejsem** - reprezentują one elementy oprogramowania takie jak: ekrany, raporty, strony HTML, lub interfejsy systemowe, które współdziałają z aktorami.
- **elementów sterujących** (ang. **control elements**), zwanych również kontrolerami - służą jako spoiwo między elementami brzegowymi i elementami jednostki oraz implementują logikę niezbędną do zarządzania różnych elementów i ich wzajemnych interakcji.

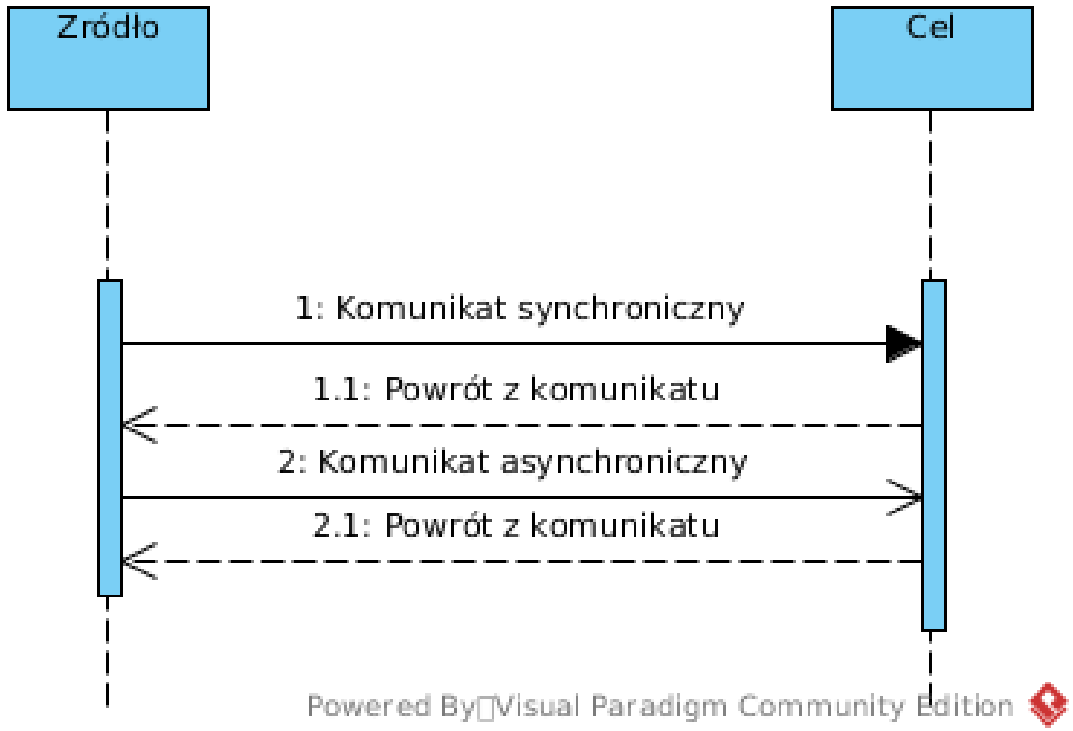
Robustness diagrams II

- **elementów jednostki** (ang. **entity elements**) - to typy jednostki, które zazwyczaj znajdują się w modelu koncepcyjnym, np. student, lista uzytkowników.
- Więcej infomacji na: `http://www.agilemodeling.com/artifacts/robustnessDiagram.htm`

Diagram sekwencji - komunikaty

- **Komunikaty** są reprezentowane jako strzałki łączące linie życia poszczególnych uczestników.
- Rodzaje komunikatów:
  - **Komunikaty** mogą być synchroniczne (oznaczane przez linie ciągłą zakończoną wypełnionym grotem) i asynchroniczne (oznaczane przez linie ciągłą zakończoną niewypełnionym grotem).
  - **Komunikaty** mogą być wywołaniem procedury (oznaczenie linia ciągła), powrotem z wywołania procedury (oznaczenie linia przerywana).
- Każdy komunikat wewnątrz interakcji opatrzony jest kolejnym numerem, co pozwala na łatwe śledzenie jej przebiegu

Diagram sekwencji - komunikaty



- Cienki prostokąt wzdłuż linii życie oznacza tzw. **słupek aktywności**.
- Słupek aktywności oznacza wystąpienie komunikacji.

Diagram sekwencji - komunikaty

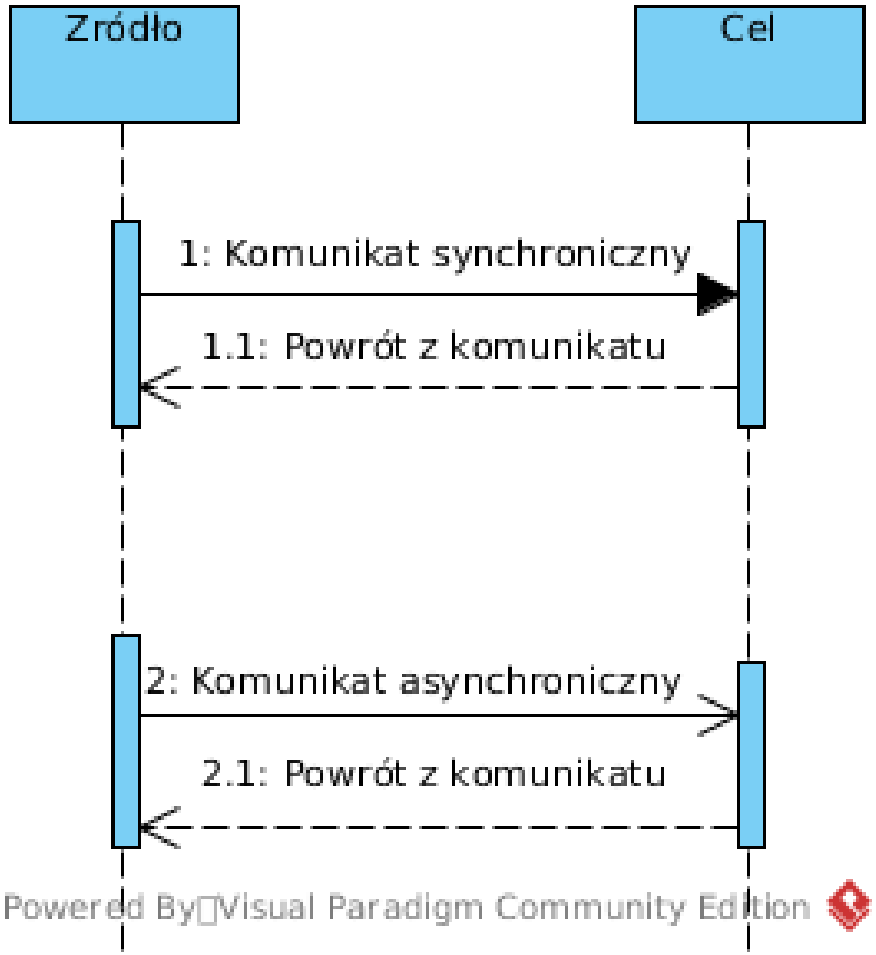


Diagram sekwencji - komunikaty

- Komunikat typu *self message* lub *recursion* może reprezentować fakt wywołania jednej metody przez inną metodę należącą do tego samego obiektu, lub rekurencyjne wywołanie pewnej operacji.

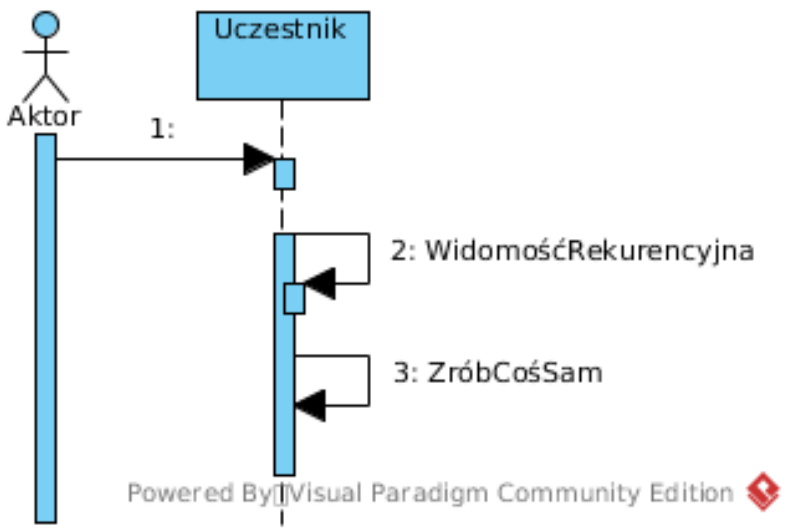


Diagram sekwencji - komunikaty zagubione i odnalezione

- **Komunikaty zagubione** (ang. lost messages) - komunikaty, które albo zastały wysłane i nie dotarły do adresata, albo zostały wysłane do odbiorcy, który nie jest pokazany na bieżącym diagramie.
- **Komunikaty odnalezione** (ang. found messages) - komunikaty wysłane przez nieznanego nadawcę lub przez nadawcę, który nie jest pokazany na bieżącym diagramie.

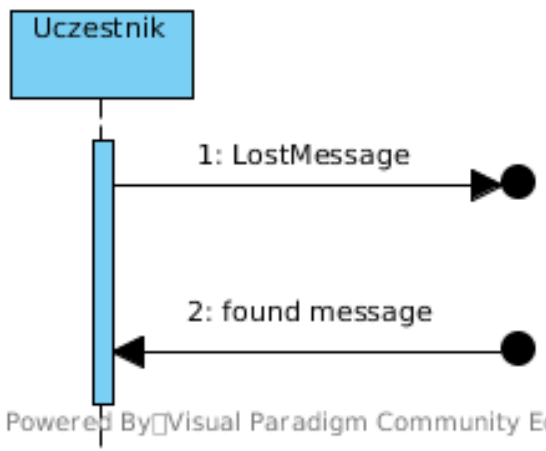


Diagram sekwencji

- Uczestnicy mogą być tworzeni lub niszczeni przez innego uczestnika.
- Linia życia zakończona symbolem krzyża reprezentuje zatrzymaną linię życia danego uczestnika.
- Uczestnik pokazany na poziomie słupka aktywności należącego do linii życia innego uczestnika, został przez tego uczestnika utworzony.
- Poniższy diagram pokazuje obiekt tworzony i niszczony.

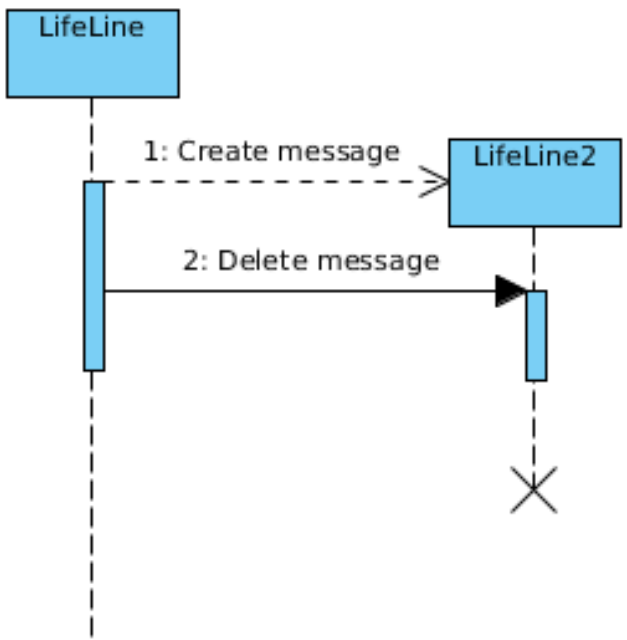


Diagram sekwencji - czas

- Domyślnie komunikaty ilustrowane są przy pomocy linii poziomej.
- Podczas modelowania systemów czasu rzeczywistego, czy też procesów biznesowych, ważne jest, aby wziąć pod uwagę czas potrzebny do wykonywania czynności.
- Komunikat z ograniczeniami czasowymi reprezentowany jest przez linię pochyłą.

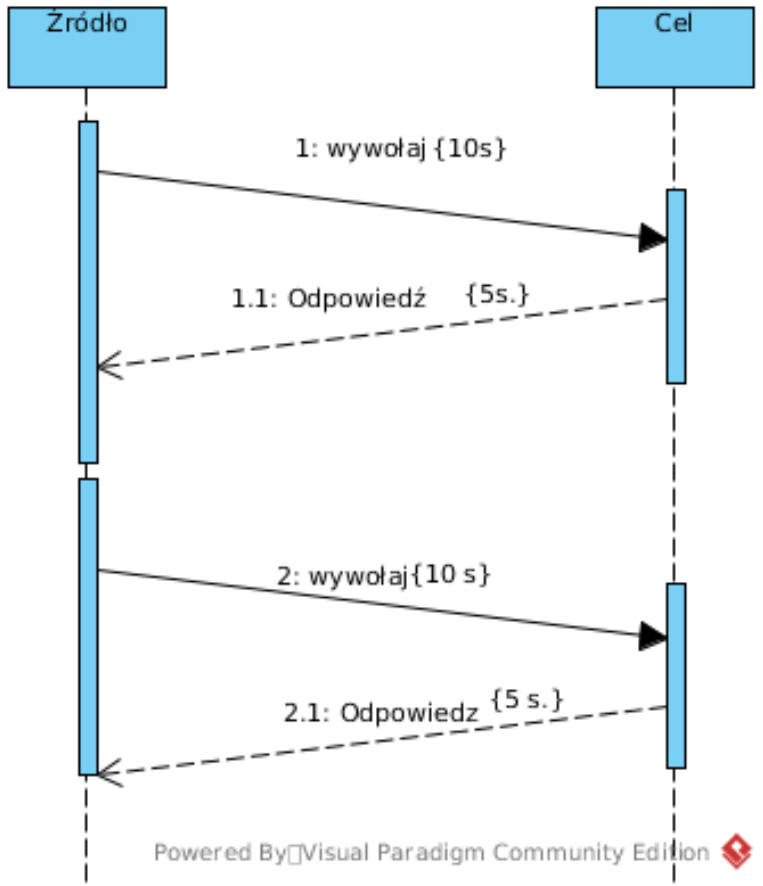


Diagram sekwencji - bloki I

- **Blok** definiuje grupę komunikatów wspólnie posiadającą pewną właściwość.
- Bloki obejmuje się prostokątem, w którego lewym górnym narożniku, w pięciokącie umieszcza się słowo kluczowe lub opis określający znaczenie danego bloku, tzw. **operator interakcji:**
  - **alt** (od alternative) - reprezentuje instrukcję *if-else*; warunek umieszcza się wewnątrz bloku w nawiasach kwadratowych
  - **opt** (od optional) - reprezentuje instrukcję *if* (bez else)
  - **break** - wykonanie fragmentu i zakończenie interakcji
  - **par** (od parallel) - nakazujący wykonać operacje równolegle

Diagram sekwencji - bloki II

- **seq** (od weak sequencing) - zamyka szereg sekwencji, dla których wszystkie komunikaty muszą być przetwarzane w danym segmencie zanim następny fragment diagramu może się wykonać;
- **strict** (od strict sequencing) - zamyka szereg komunikatów, które muszą być przetwarzane w podanej kolejności.
- **neg** (od negative) - zamyka nieprawidłową serię wiadomości.
- **critical** - oznacza obszar krytyczny
- **ignore/consider** - ignore(komunikat1, komunikat2, ...) oznacza, że na diagramie nie pokazano wymienionych komunikatów, choć mogą wystąpić. Consider = odwrotnie

Diagram sekwencji - bloki III

- **loop** - definiuje pętlę typu for (o określonej z góry liczbie iteracji) lub while (wykonywanej dopóki pewien warunek jest prawdziwy)

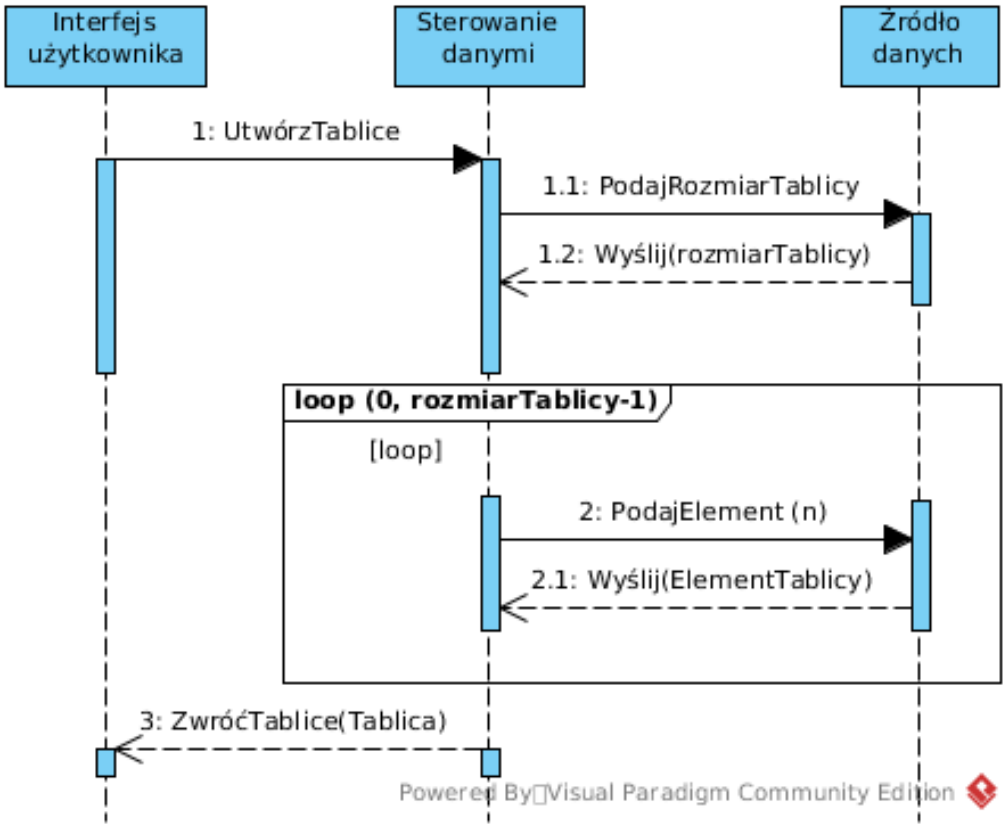


Diagram sekwencji - niezmiennik/kontynuacja

- Niezmiennik jest warunkiem umieszczonym na linii życie i musi być spełniony w czasie wykonywania.
- Niezmiennik jest przedstawiony jako prostokąt o półokrągłych końcach.
- Kontynuacja ma ten sam zapis jak niezmiennik, ale jest używana w połączeniu z blokami i może rozciągać się na więcej niż jedną linię życie.

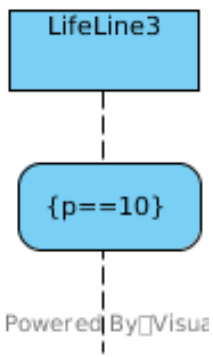


Diagram sekwencji- częściowa dekompozycja

- Częściowa dekompozycja (and. part decomposition) oznacza, że uczestnik może mieć więcej niż jedną linię życia.
- Częściowa dekompozycja pozwala na przesyłanie wiadomości między i wewnątrz uczestnikami.

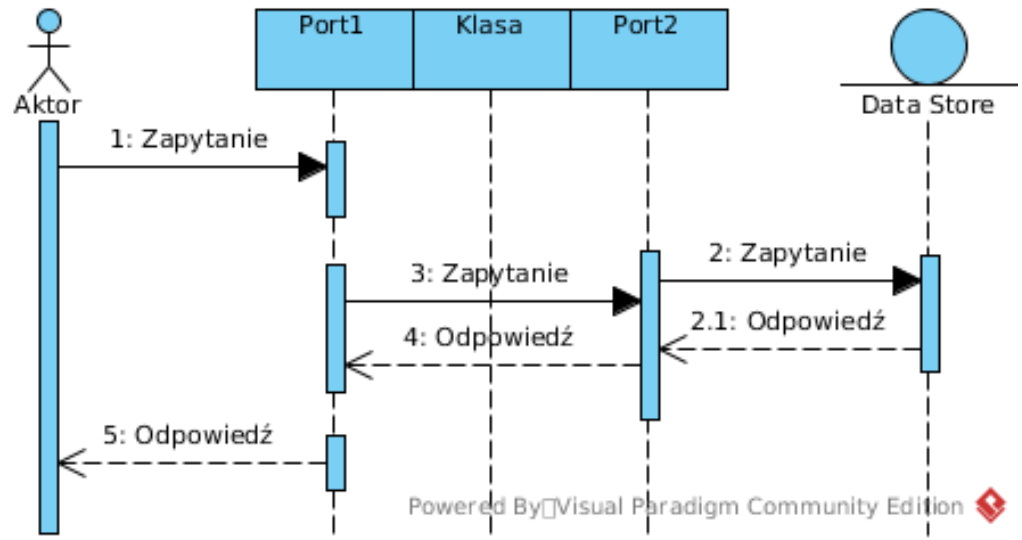


Diagram sekwencji- bramki

- Brama jest punktem łączenia komunikatów wewnątrz bloku z komunikatami poza blokiem.
- Bramka oznaczana jest przez mały kwadrat na ramie bloków.
- Bramki działają jak off-strony do diagramów sekwencji, reprezentujących źródło komunikatów przychodzących lub cel komunikatów wychodzących.

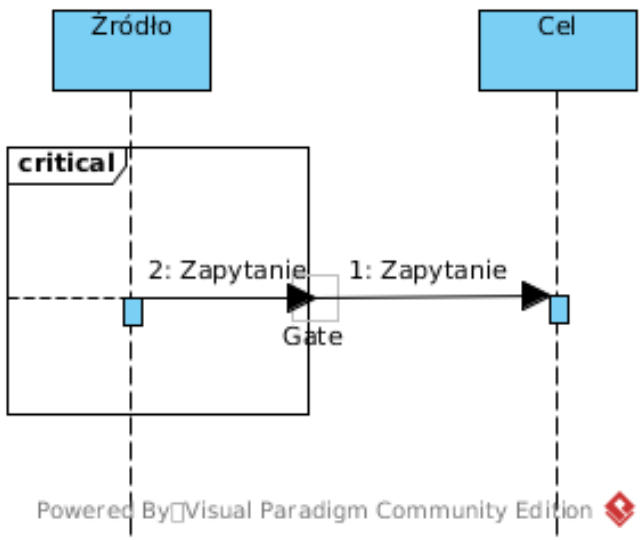


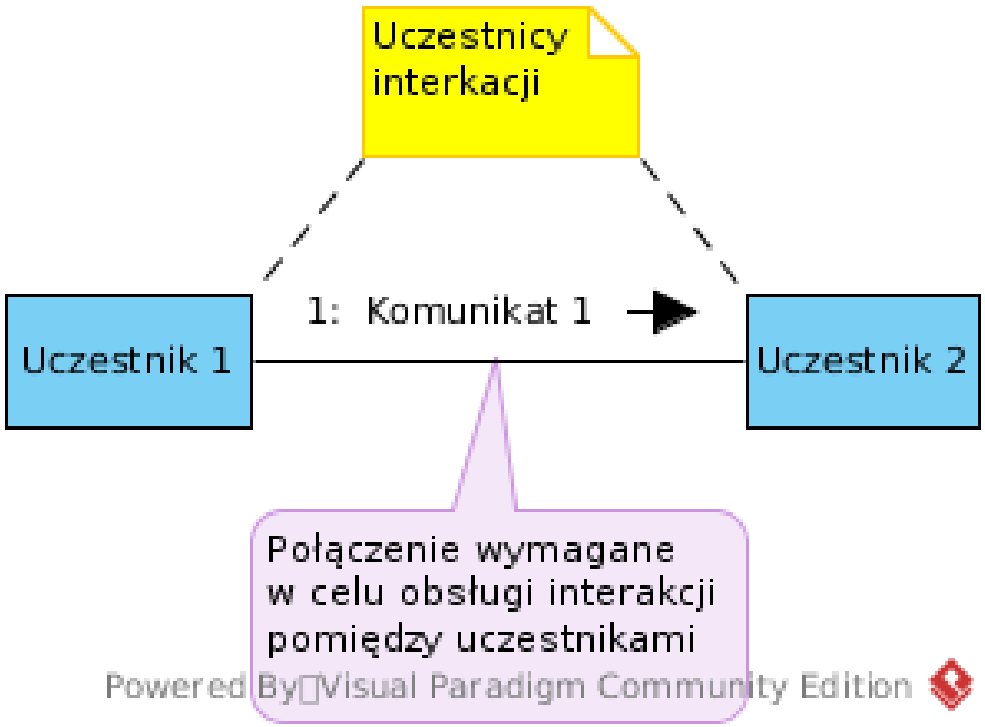
Diagram sekwencji a diagramy aktywności

- Diagram aktywności opisuje co będzie robione, ale nie wskazuje realizatora.
- Diagram sekwencji wskazuje realizatora zadania i opisuje współpracę realizotra zadania z otoczeniem podczas wykonywania danego zadania.

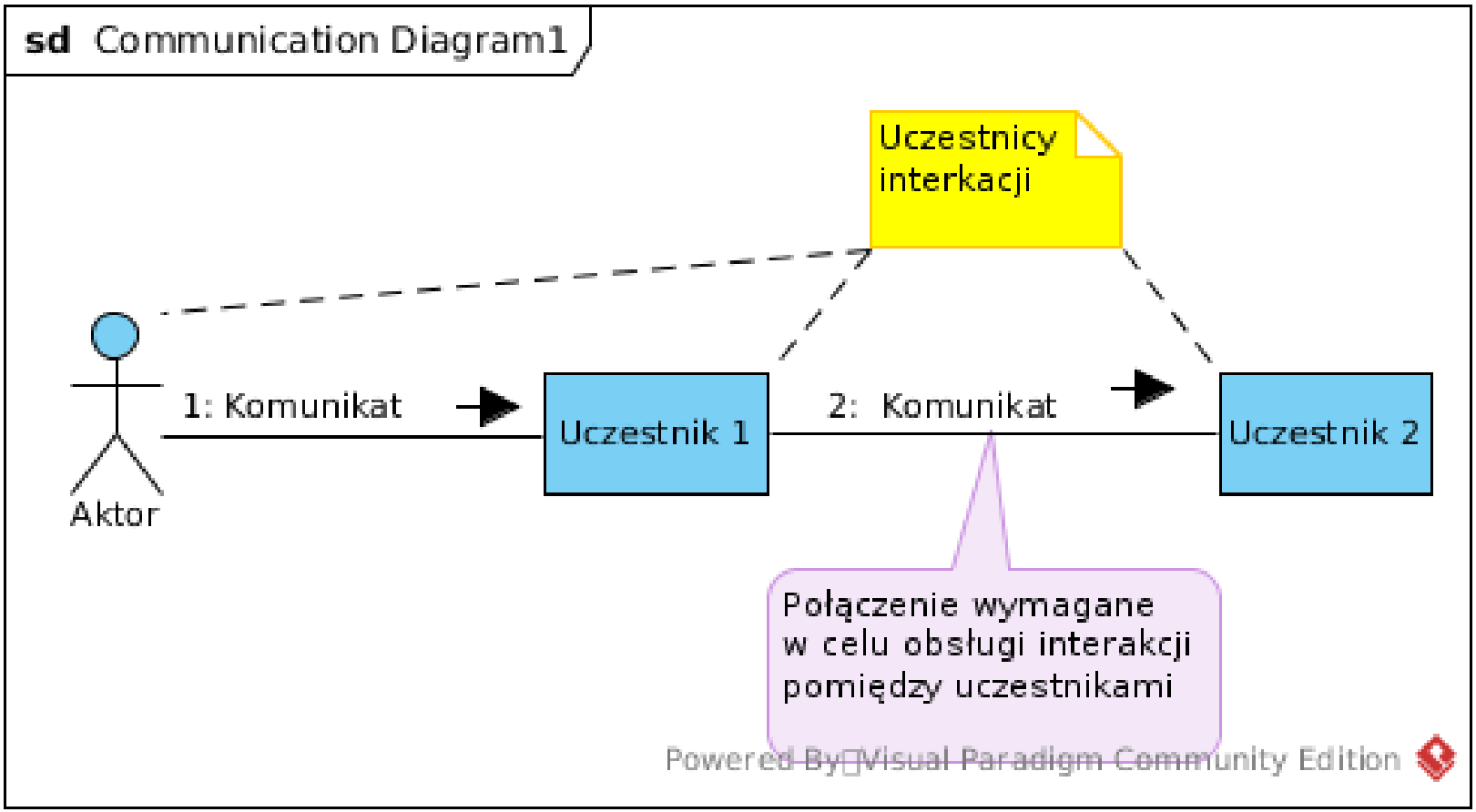
Diagramy komunikacji

- **Diagram komunikacji** (ang. communication diagram), dawniej nazywany **diagramem współpracy** (ang. collaboration diagram), jest jednym z czterech diagramów interakcji (ang. interaction diagram).
- **Diagram komunikacji** pokazuje informacje podobne do tych co diagram sekwencji, ale jego głównym celem jest ilustracja relacji pomiędzy uczestnikami komunikacji.
- Elementy diagramu komunikacji:
  - **uczestnicy** (aktorzy, obiekty, klasy biorące), nazywani również *lifelines*, ale nie posiadają pionowych linii życia, tylko same "głowy".
  - **asocjacje** - główny związek pomiędzy uczestnikami, reprezentowany przez linie łączące uczestników
  - **komunikaty** - realizacja interakcji, opisywane etykietowanymi krótkimi strzałkami (strzałka powinna wskazywać kierunek przepływu komunikatu.)

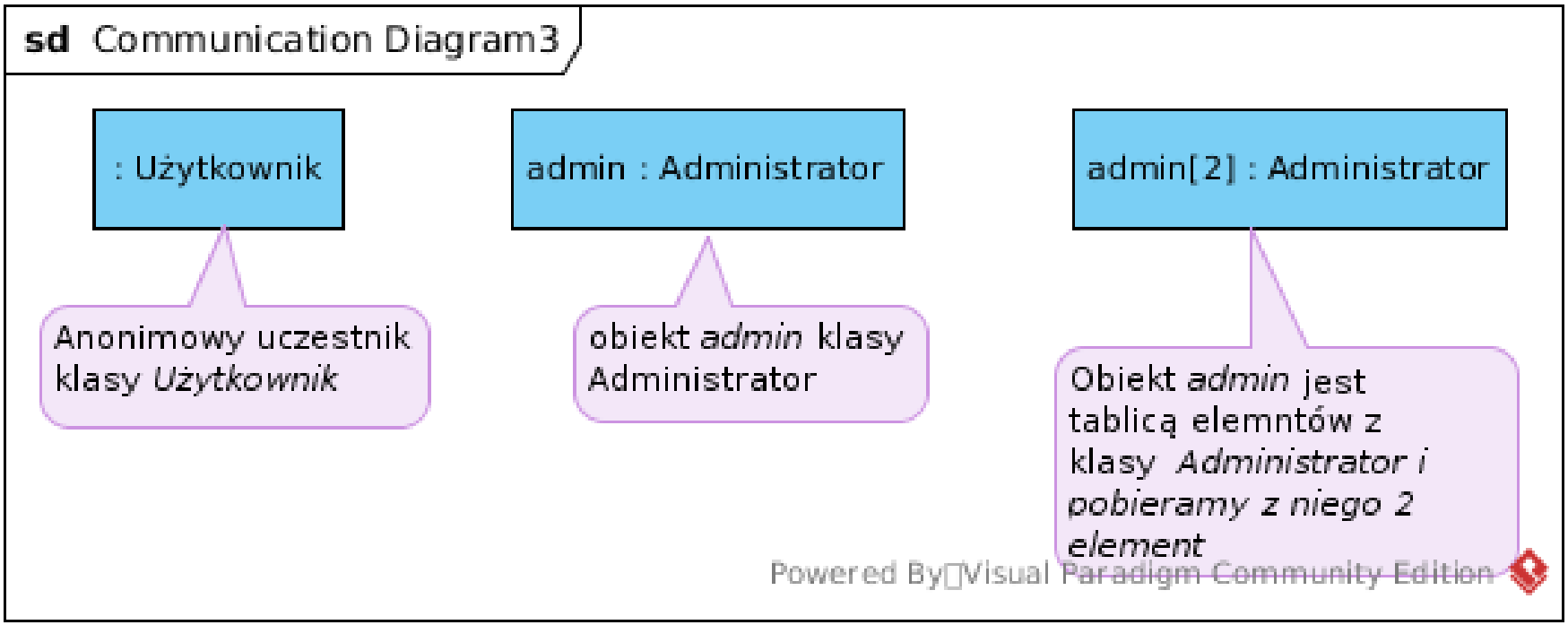
Diagramy komunikacji - przykład



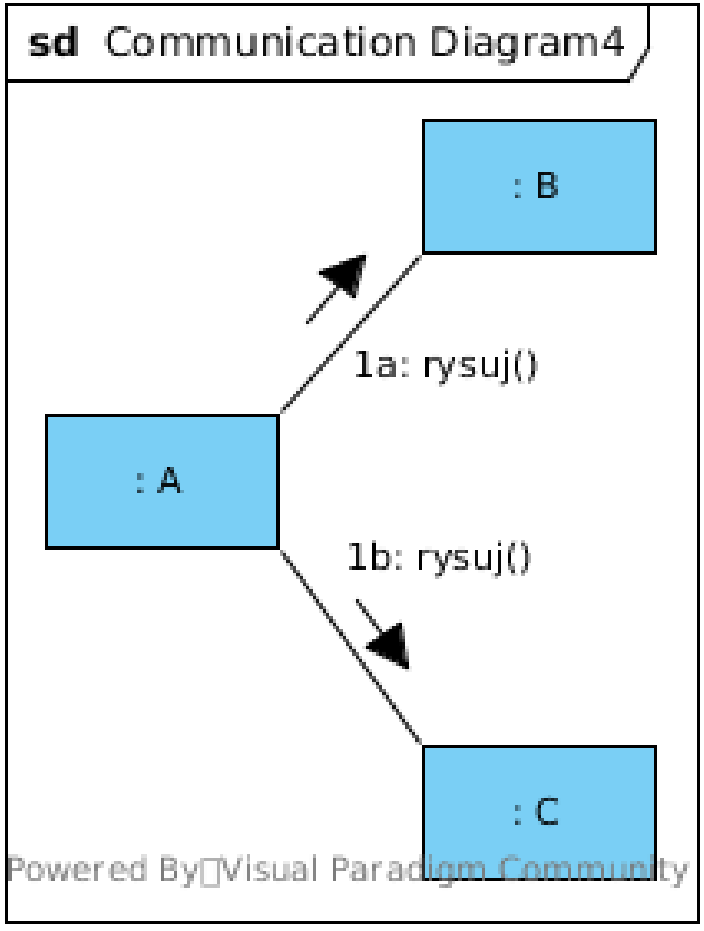
Diagramy komunikacji - przykład



Diagramy komunikacji - nazwy uczestników

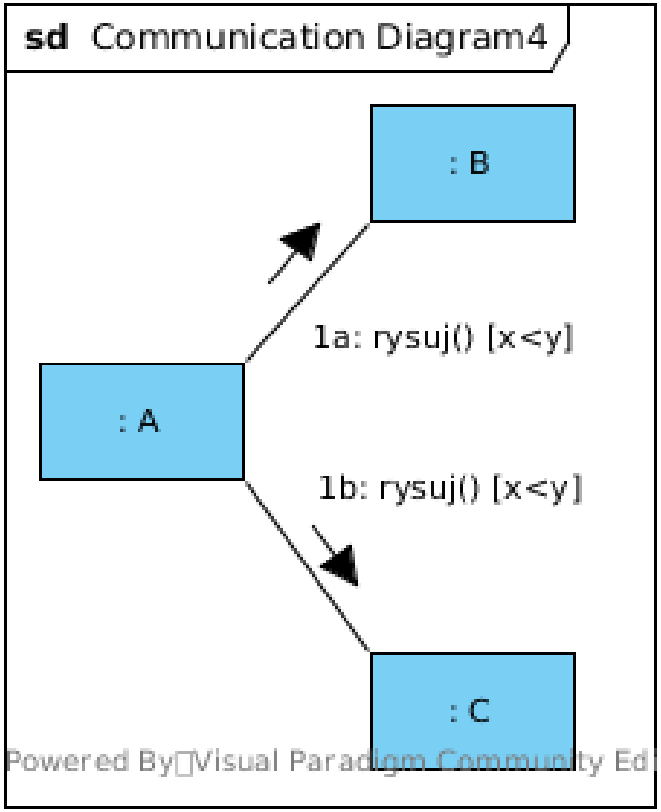


Diagramy komunikacji - modelowanie współbeżności



- Instancja klasy A wysyła komunikat *rysuj()* jednocześnie (współbieżnie) do instancji klasy B i instancji klasy C.

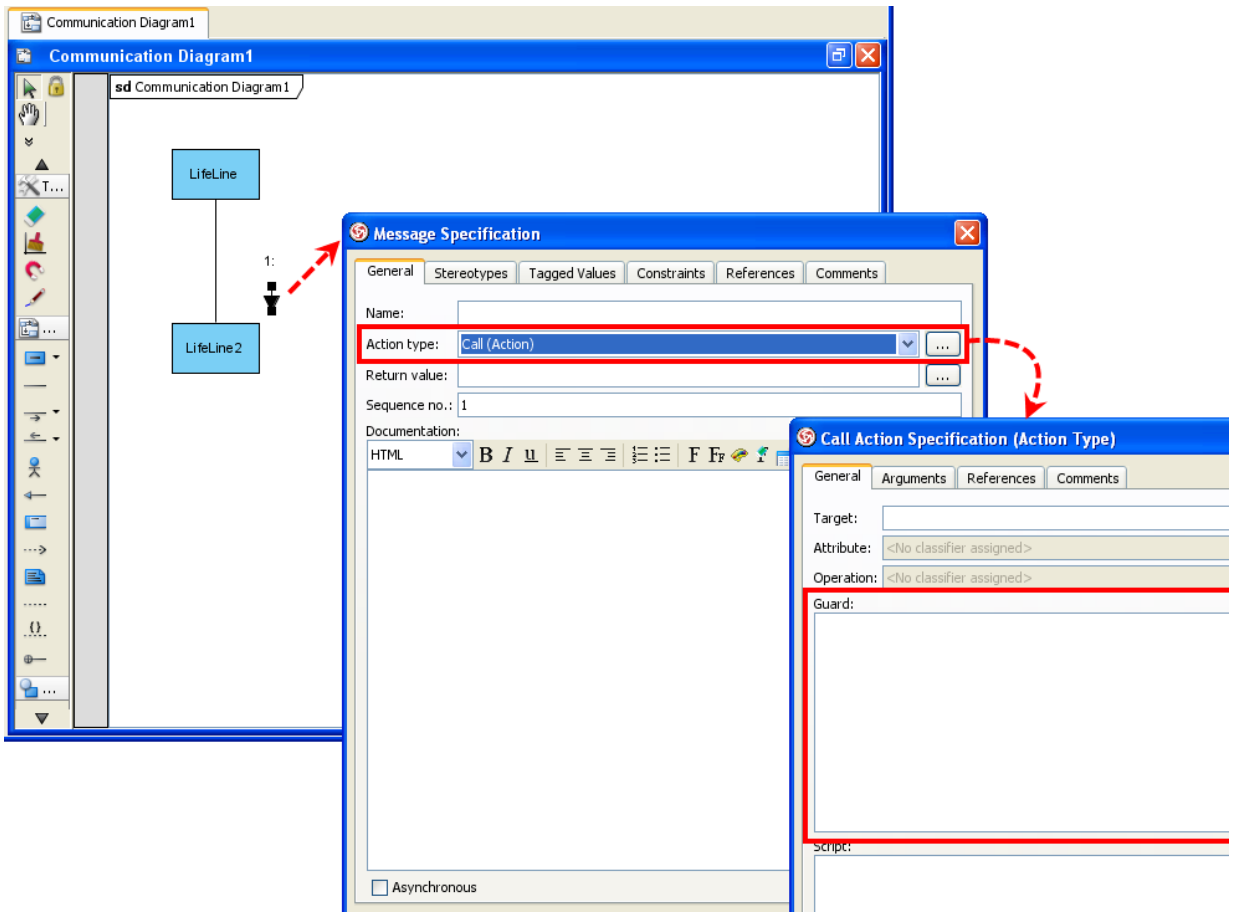
Diagramy komunikacji - modelowanie komunikatów z ograniczeniami



- Instancja klasy A wysyła komunikat *rysuj()* jednocześnie (współbieżnie) do instancji klasy B i instancji klasy C, jeśli $x > y$.

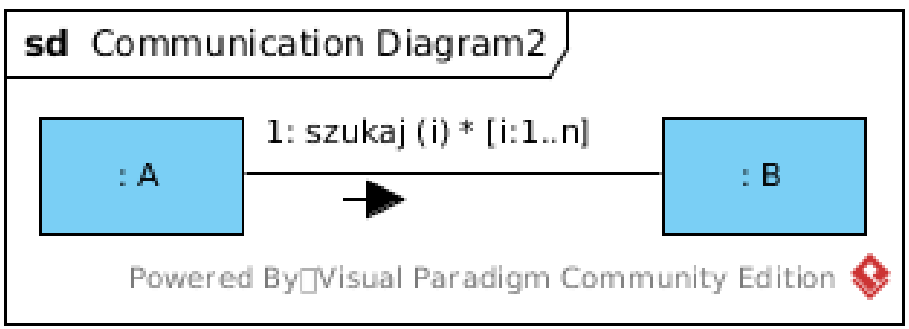
Diagramy komunikacji - modelowanie komunikatów z ograniczeniami

- Wprowadzenie ograniczeń w VP:



Diagramy komunikacji - modelowanie komunikatów z iteracją

- Komunikat *szukaj()* będzie wykonany *n*razy



Diagramy komunikacji I

- Przy pomocy diagramiu komunikacji można modelować podział systemu na komponenty. Służą do tego tzw. swimlanes.
- Kolejny slajd pokazuje system oparcji transferu pieniędzy, który podzielony jest na dwa podsystemy: *Client* oraz *Main frame*.
- Kroki do powstania diagramiu komunikacji dla oparcji transferu pieniędzy:
  - Aby utworzyć diagram komunikacji, wybierz z menu **Diagram** > **New**.
  - W oknie **New Diagram**, wybierz **Communication Diagram** i zatwierdź z nazwą *Transfer Money*.
  - Dodaj dwie **swimlanes**, aby zamodelować podział systemu na aplikację klienta *Client* oraz system po stronie banku *Main frame*.

Diagramy komunikacji II

- Po stronie klienta utwórz aktora o nazwie *User*.
- *User* kontaktuje się z systemem poprzez swoje konto internetowe *account page*.
- Kontakt *User*a z *account page* jest modelowany poprzez wysłanie komunikatu (**Message -> LifeLine**) z opisem *visit*.
- Konto internetowe skieruje wniosek użytkownika o przekazanie pieniędzy do systemu bankowego do zatwierdzania i realizacji poprzez moduł *Transaction*, reprezentowany jako uczestnik.
- komunikat wysłany od *Account Page* do *Transaction* to *transfer (targetAccount, amount)*.

Diagramy komunikacji III

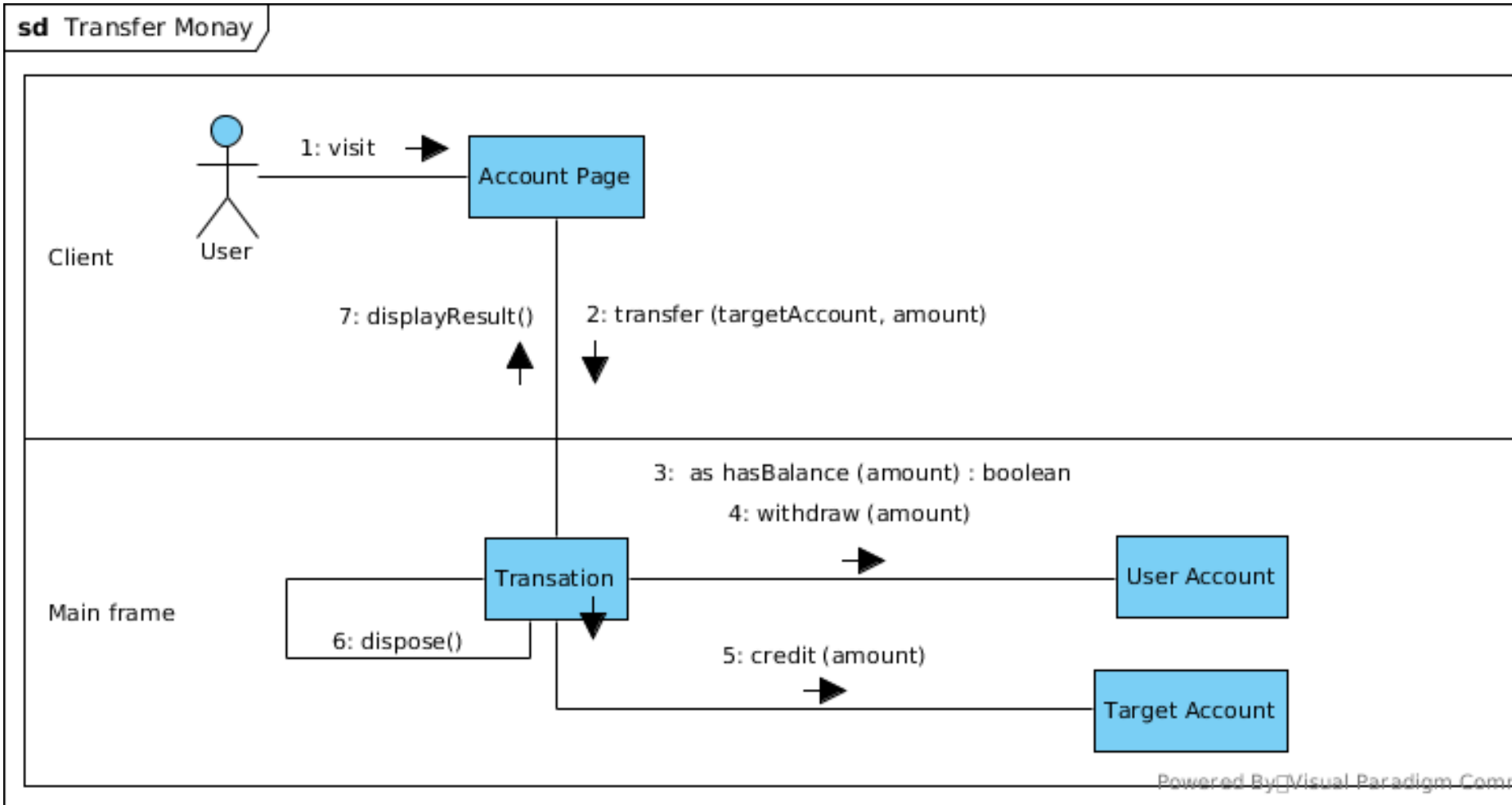
- Proces transferu pieniędzy polega na wypłacanie pieniędzy z konta użytkownika, a następnie wplaceniu tych pieniędzy na konto docelowe. Zanim ta operacja zostanie wykonana, musimy mieć pewność, że na koncie użytkownika jest wystarczająco dużo pieniędzy.
- Aby zamodleować powyższą sytuację, tworzymy uczestnika po stronie systemu bankowego o nazwie *User Account* i wysyłamy do niego od uczestnika *Transaction* komunikat: *hasBalance (amount) : boolean*.
- Gdy saldo konta użytkawnika zostanie sprawdzone, możemy wypłacić pieniądze z jego konta. Można to zamodelować wysyłając komunikat od *Transaction* do *User Accoun* o treści *withdraw (amount)*.

Diagramy komunikacji IV

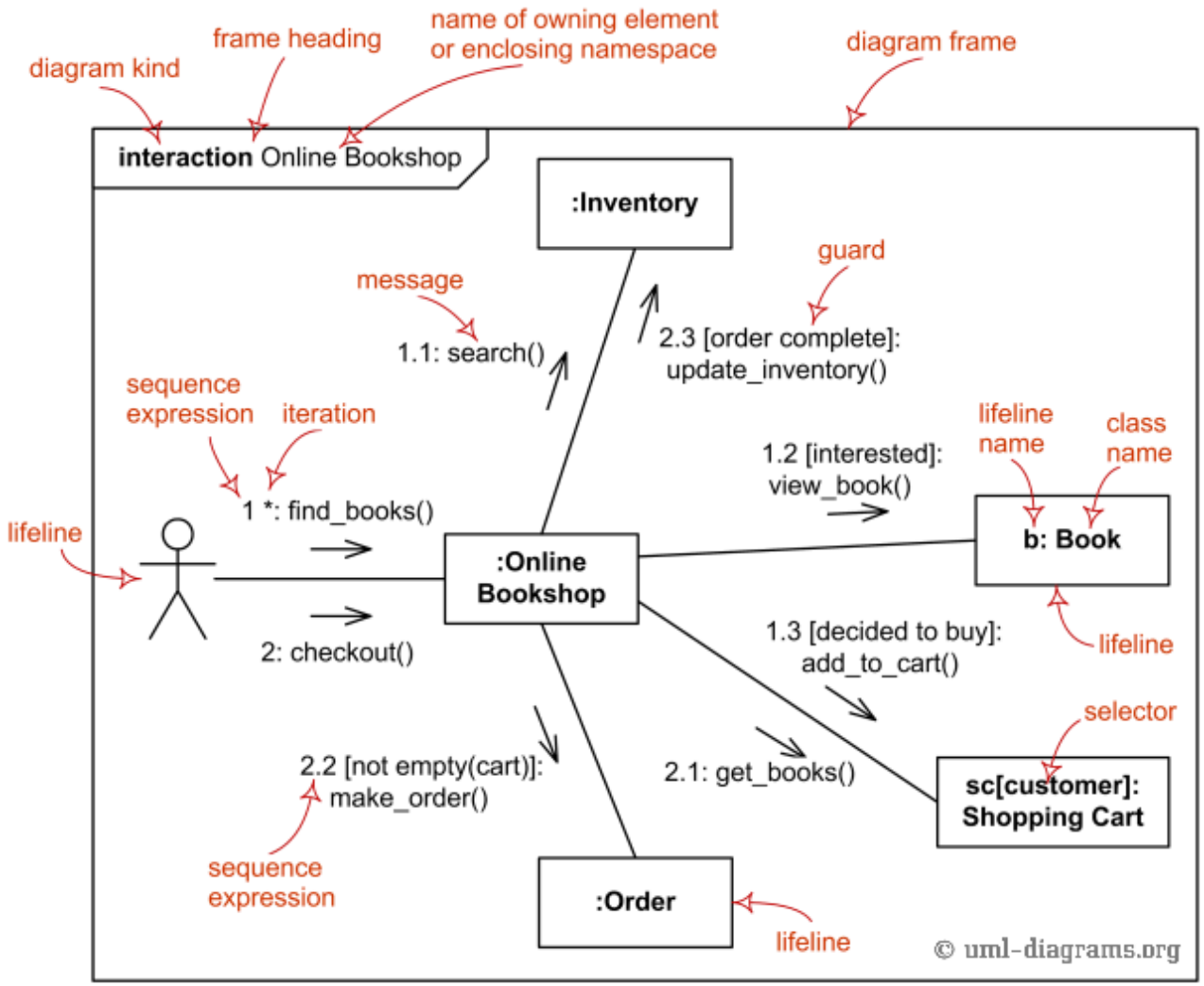
- Aby zamodelować wpłacanie pieniędzy na konto docelowe, można utworzyć uczestnika *Target Account* i wysłać do niego komunikat *credit (amount)* od uczestnika *Transaction*.
- Any zamodelować w systemie informacje o zrealizowanej tranzakcji, uczestnik *Transaction* może wysłać komunikat do siebie o treści *dispose*.
- Na koniec, aby poinformować użytkownika, że transakcja jest zakończona, wysyłamy komunikat *displayResult()* od uczestnika *Transaction* do uczestnika *Account Page*.

Diagramy komunikacji - operacja transferu pieniędzy online

Diagram powstały na podstawie tutorialu: `http://www.visual-paradigm.com/tutorials/communicationdiagram.jsp`



Diagramy komunikacji



Źródło:http://www.uml-diagrams.org/communication-diagrams.html

Różnice między diagramami komunikacji a sekwencji

- Diagramów sekwencji używa się, gdy zainteresowani jesteśmy głównie przepływem komunikatów w danej interakcji.
- Diagramów komunikatów używa się, gdy chcemy się skoncentrować głownie na połączeniach pomiędzy uczestnikami danej interakcji.