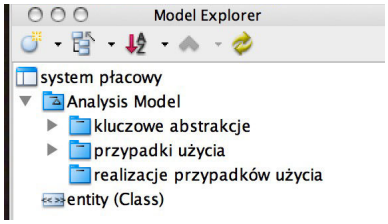
# Realizacje przypadków użycia – diagramy VoPC

To ćwiczenie rozpoczyna tworzenie realizacji przypadków użycia. Realizacje to nic innego jak pokazanie w jaki sposób (i z użyciem których klas) poszczególne przypadki użycia będą implementowane.

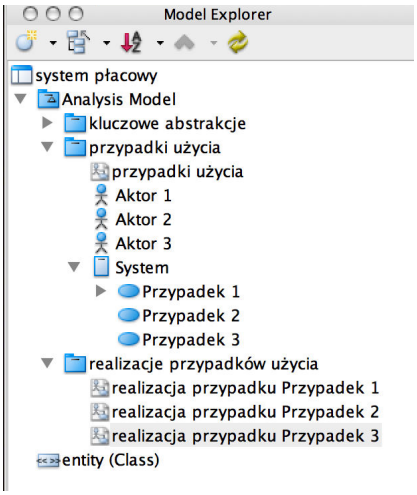
Jak zwykle, prace zaczynamy od wspólnego utworzenie przykładowej realizacji przypadku użycia, następnie wykonujemy samodzielnie realizacje dla wybranych podczas wcześniejszych zajęć przypadków modelowanego systemu.

## *Model systemu*

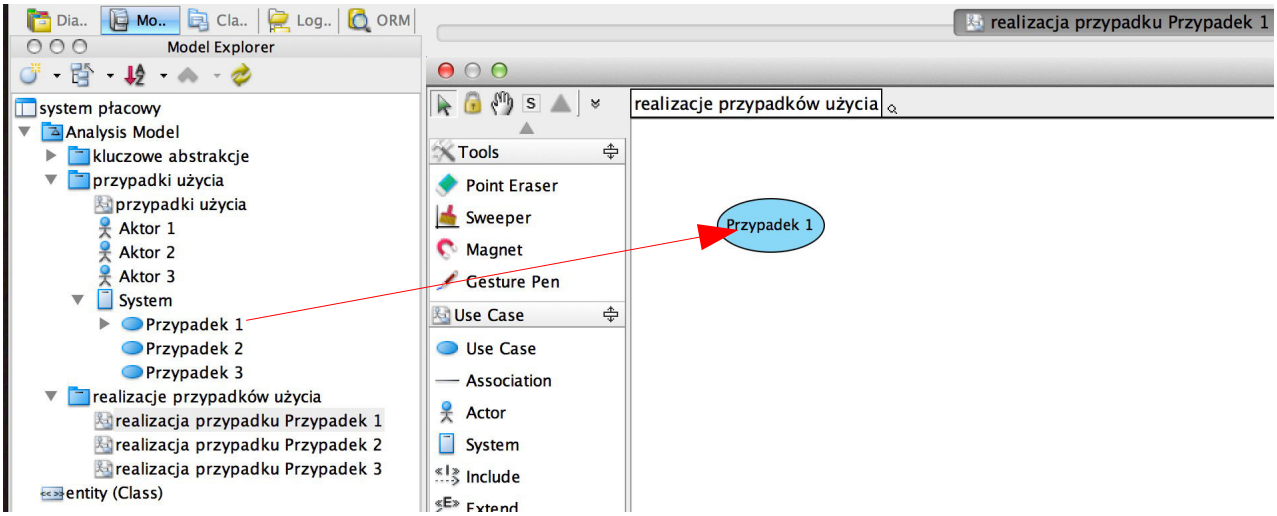
Prace rozpoczynamy od dodania do modelu nowego pakietu „realizacje przypadków użycia”:



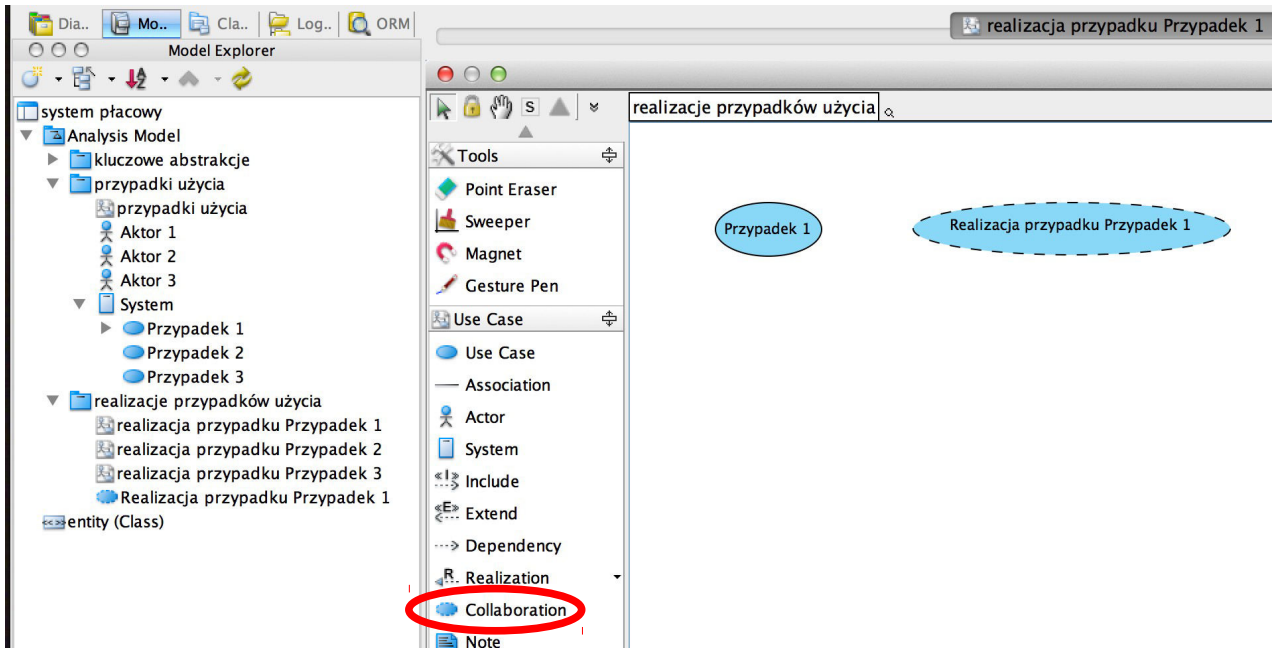

Do pakietu dodajemy nowy diagram przypadków użycia dla każdej realizacji (dla każdego ze zidentyfikowanych wcześniej przypadków użycia) o nazwie „realizacja przypadku <nazwa przypadku użycia>” (nazwy elementów modelu muszą być unikatowe):



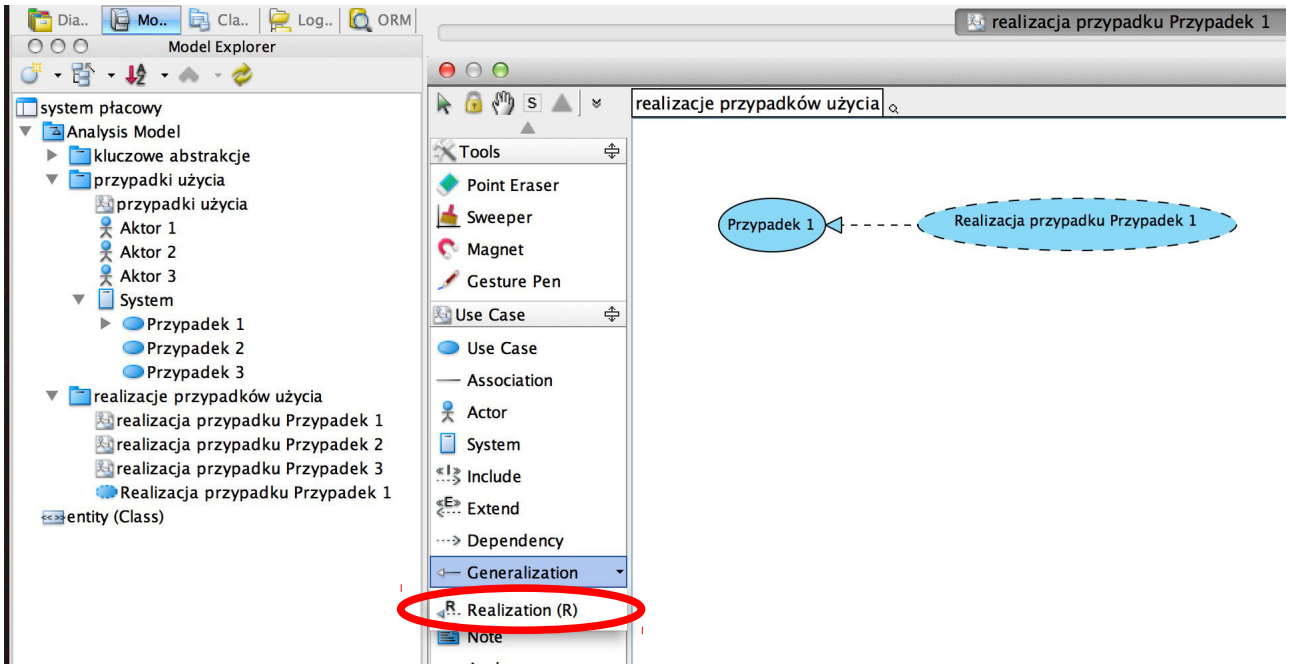

Sporządzimy związek, który będzie pokazywał zależność *traceability* między przypadkiem użycia a jego realizacją. Aby to osiągnąć, do diagramu przypadków użycia pokazującego realizację danego przypadku użycia przeciągamy ten przypadek z modelu (nie tworzymy nowego przypadku użycia):



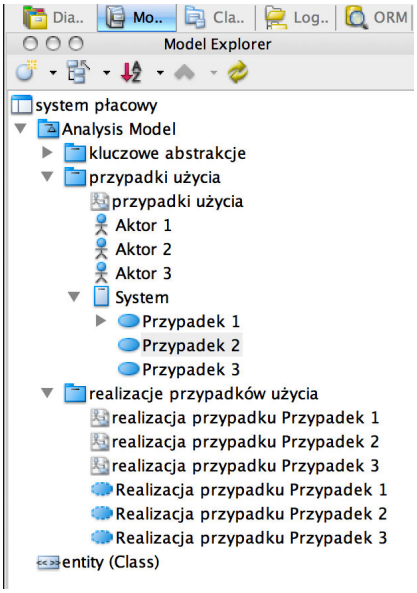

Następnie dodajemy kolaborację i nadajemy jej unikatową nazwę wynikającą z przypadku użycia:



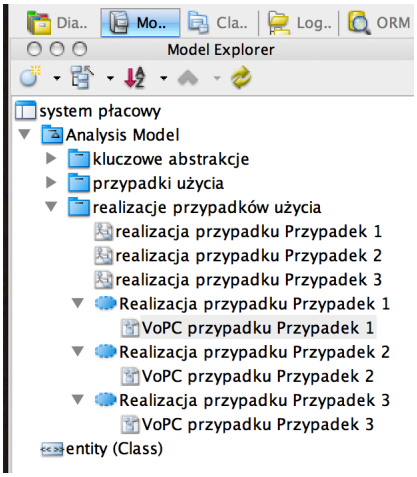

Przypadek z jego realizacją łączymy oczywiście związkiem realizacji, który wprost zapewnia zachowanie *traceability* między przypadkiem użycia a jego realizacją (rysunek poniżej).



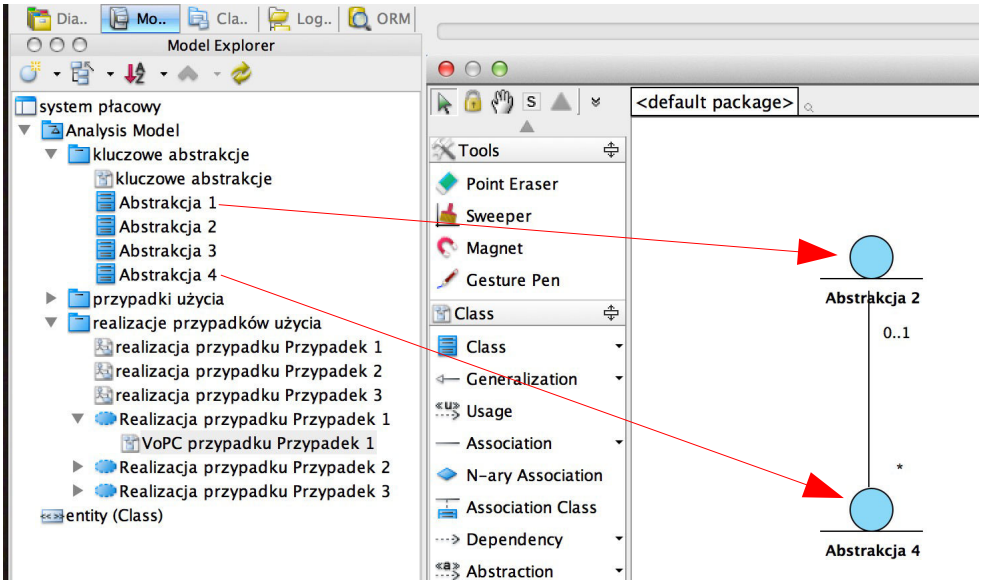

Czynności związane z dodawaniem diagramów realizacji powtarzamy dla wszystkich przypadków użycia (na potrzeby laboratorium – tych wybranych wcześniej do modelowania):




Każdy przypadek użycia będzie miał własny zestaw klas analizy, które będą potrzebne to jego realizacji (osiągnięcia założonego działania systemu w ramach tego przypadku użycia). Aby ten zestaw klas pokazać, pod każdą z realizacji (kolaboracje widoczne w modelu) podpinamy diagram klas i nazywamy go „VoPC przypadku <nazwa przypadku użycia>”, czego efekt pokazuje poniższy rysunek:




Dla każdej z realizacji decydujemy, które z kluczowych abstrakcji będą w niej uczestniczyły i przeciągamy na odpowiadający jej VoPC zidentyfikowane wcześniej kluczowe abstrakcje (klasy *entity*). Zwróćmy uwagę, że dzięki spójności modelu Visual Paradigm jest w stanie odtworzyć asocjacje, które wcześniej zdefiniowaliśmy na diagramie klas kluczowych abstrakcji:




Pora na dodanie pozostałych klas analizy – *boundary* i *control*.

> Pamiętajmy o regule, że każdy przypadek użycia ma jedną klasę *control* zawierającą w sobie całą logikę przypadku i po jednej klasie *boundary* na każdą parę aktor-system.

Dla klas *control* stosujemy konwencję nazw w rodzaju „Zarządca <nazwy przypadku użycia>” lub „Menedżer <nazwy przypadku użycia>” (np. „Zarządca logowania”), natomiast dla klas *boundary* w przypadku:

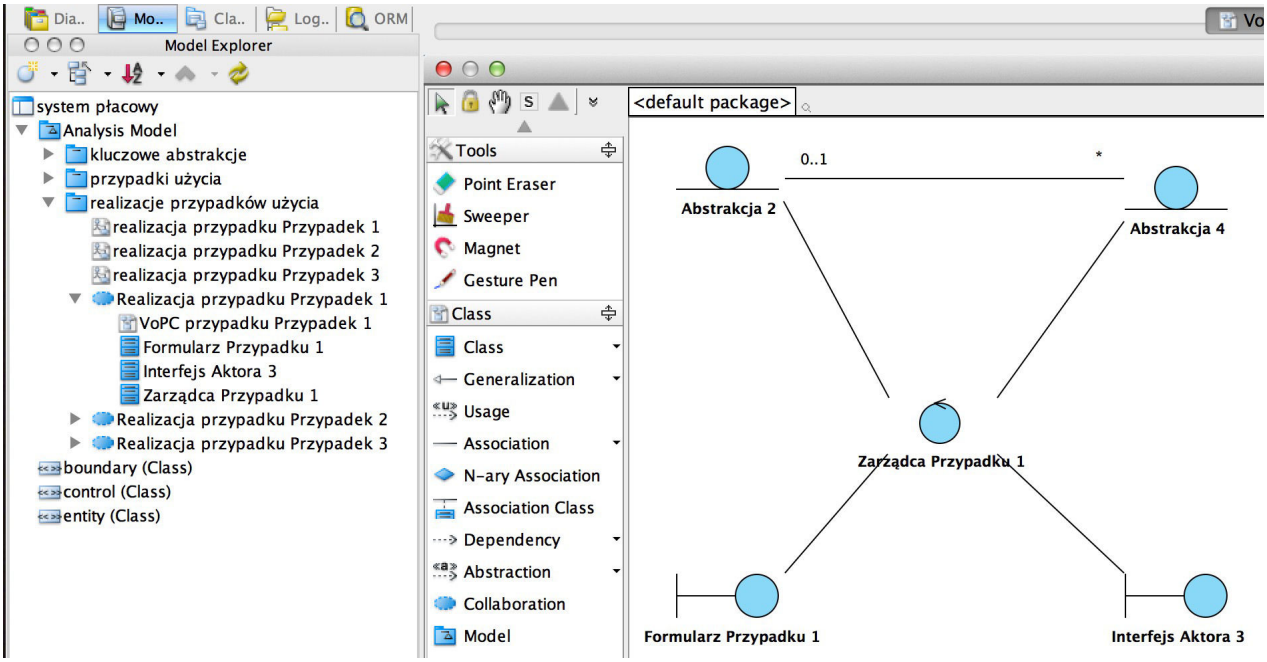
- aktorów ludzkich – „Formularz <nazwy przypadku użycia>” (np. „Formularz logowania”),
- urządzeń i systemów – „Interfejs <nazwy aktora> (np. „Interfejs drukarki”).

Możemy założyć, że pomimo oddzielnych klas *boundary* dla tego samego aktora w różnych przypadkach użycia (inny GUI logowania, inny GUI zarządzania profilem, itp.), urządzenia i systemy będą miały ten sam interfejs w różnych przypadkach użycia (np. ten sam interfejs drukarki używany do drukowania raportów i potwierdzeń przelewów bankowych).

Czyli dodajemy do każdego VoPC klasy *control* i *boundary*, nazywamy je zgodnie z konwencją, przypisujemy im stereotypy i dodajemy brakujące asocjacje (rysunek poniżej).

> Klasy *boundary* „widzą się” tylko z klasami *control*.
>
> Klasy *control* „widzą się” z klasami *boundary* i *entity*.
>
> Klasy *entity* „widzą się” tylko z klasami *control* i innymi klasami *entity* (mamy już te związki z diagramu klas kluczowych abstrakcji).

Zauważmy, że klasy *control* i *boundary* znajdują się bezpośrednio w danej realizacji przypadku użycia:




Diagramy VoPC sporządzamy dla realizacji wszystkich przypadków użycia (na laboratoriach – tylko tych wybranych wcześniej).