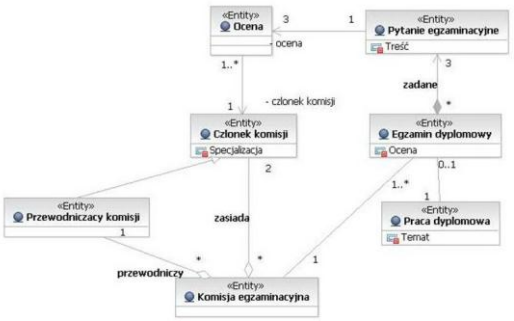
Aby przenieś model UML między narzędziami CASE powinny one być zgodne ze standardem **XMI**

Analizując poniższy diagram zmodyfikuj zapis krotności tak, aby pytania podczas egzaminu dyplomowego były losowane z listy 100 pytań. Poprawną definicję wpisz w polu tekstowym (pomijając spacje).



100{bag}

Atrybuty architektury to: Wybierz co najmniej jedną odpowiedź
- użyteczność
- statyka
- zachowanie
- piękno
- trwałość
- język

Celem wzorca "budowniczy" nie jest rozdzielenie sposobu tworzenia obiektów od ich reprezentacji. **Fałsz**

Czy prawą jest ze w podejściu translacyjnym (**w MDA**) zmiana platformy nie wymaga kodowania logiki aplikacyjne w języki nowej platformy? **TAK** bo działanie opisane jest dodatkowo za pomocą jeżyka semantyki akcji. Stąd czy potrzebujemy tworzyć PSM **NIE**

Czy klasa StawkaVat powinna być powiązana z klasą Towar? Odpowiedź brzmi nie, ponieważ klasa StawkaVat **enumeration**

Dane jest następujące wyrażenie OCL: 'kolekcja->forAll(var : Integer | var = 5)' . Zaznacz prawidłowe stwierdzenia
- Zmienna var zostanie podstawiona wartością 5.
- **Wartość danego wyrażenia jest typu Boolean.**
- Wartość wyrażenia 'var = 5' jest typu Boolean.
- Własność var wszystkich elementów kolekcji zostanie ustawiona wartością 5. (?)

Diagramy wymagań są poprawną techniką w **SysML**

Diagramy wymagań są poprawną techniką w:
- SoaML;
- **MDSD;**
- DSL;
- **UML**

Dopasuj charakterystyki do kwalifikatorów UML
- Określa czy własność jest wprowadzona, tzn jej wartość lub jej wartości można obliczyć z innych informacji zawartych w modelu **isDerived**
- W przypadku wystąpień wielokrotnych oznacza ze wartości w instancji danego elementu są sekwencyjna uporządkowane **isOredered**
- Własność może być tylko czytana **isReadOnly**
- W przypadku wystąpień wielokrotnych oznacza że wartości w instancji danego elementu są unikalne **isUnique(ok.)**

Dopasuj charakterystyki do typów OCL:
- ustalony porządek, możliwe powtórzenia **Sequence**
- brak porządku, brak powtórzeń **Set**
- ustalony porządek, brak powtórzeń **Orderset**
- brak porządku, możliwe powtórzenia **Bag**

- istnienie porządku i powtórzeń nieokreślone **Collection**
- istnienie porządku i powtórzeń **nieokreślone**

Dopasuj liczbę punktów widzenia (aspektów), do odpowiedniego paradygmatu modelowania:
- Obiektowy **2**
- Strukturalny **3**
- Zorientowany na usługi **1**

Dopasuj profil do metody (do wyboru: Brak metody MDSD MDS MDD)
- Profil SysML **MDSD**
- Profil SecureML **brak metody**
- Profil SoaML **MDD**
- Profil CimML **MDS**

Ekspozycja kandydatów na usługi odbywa się przez powiązanie możliwości (Capability) zależnością Expose z:
- Component
- ServiceContract
- Part
- **ServiceInterface**
- Collaboration
- Participant

Interfejs dostarczany uzyskamy poprzez zastosowanie do klasy interfejsu zależności:
- use
- include
- derive
- import
- **realization**

Interfejs wymagany uzyskamy poprzez zastosowanie, do klasy interfejsu zależności:
- realization
- derive
- import
- include
- **use**

Jaki typ relacji połączy klasę Osoba z klasą Produkt żywieniowy, jeżeli ma ona nazwę 'je'. **zależność**

Jawna konkretyzacja (wiązanie) wzorca LIST [ITEM] wymaga zastosowania zależności ze stereotypem, który wiąże wzorzec z jego parametrem. **Bind**

Jeśli pakiety na diagramie opisują warstwy, to asocjacje/zależności nie: **są poprawne**

Jeśli w klasie Osoba jej atrybut wiek będzie określony po znaku "/", tj. /wiek, to jest to atrybut (podaj nazwę typu)? **Pochodny**

Jeśli w klasie Pracownik jest atrybut staż i staż=dzis()-Pracownik.dataZatrudnienia; to w utwardzonej klasie projektowej ten atrybut powinien pozostać [Prawda], czy nie [Fałsz]? **Fałsz**

Jeżeli atrybut lub metodę poprzedzimy symbolem po lewej stronie, to jej widoczność będzie określona, jak po prawej stronie:
- ~ **package**
- # **protected**
- + **public**
- - **private**

Jeżeli atrybut Uwaga, klasy pracownik powinien być dostępny (widoczny) jedynie w tej klasie i jej podklasach (klasach potomnych), to oznaczamy go jako: (wpisz jeden znak) **#**

Jeżeli opisujemy złożone własności bytu projektowego (w odróżnieniu od prostych - {nazwa1 = wartość1}), to znaczniki (tagged value) rozdzielamy: (wpisz symbol oddzielający znaczniki bez zbędnych spacji): **, (przecinek)**

Jeżeli w klasie Pracownik, atrybut liczbaPracowników jest podkreślony to oznacza, że możemy się bezpośrednio odwołać do jego wartości [Prawda], czy musimy najpierw powołać jego obiekt [Fałsz]. **Prawda**

Klasy reprezentujące typy wyliczeniowe (**enumerations**) na diagramach klas z klasami opisującymi dziedzinę łączymy relacją **brak relacji**

Kształt systemu określa wynik modelowania:
- wymagań
- implementacji
- **architektury**
- analizy}

Którego ze stereotypów najlepiej użyć do zamodelowania pokrycia systemu testami na diagramie wymagań:
- copy
- refine
- satisfy
- **verifies**

Który z diagramów należy do SysML a nie UML: Wybierz co najmniej jedną odpowiedź **wymagań**

Manifest MDA bezpośrednio odwołuje się do: **Otwartych standardów, automatyzacji i bezpośredniej reprezentacji problemów**

MDA określa koncepcje
- Przekształcenia kodu źródłowego systemu do pisu właściwego platformie
- **Przekształcenia opisu systemu do opisu właściwego platformie**
- Przekszłacenia modelu w SysML do opisu UML
- **Wyboru platformy dla konkretnego systemu**
- **Opisu systemu niezależnego od platformy**
- **Opisu platformy**

Miejsce jest pojęciem w modelu:
- UML
- **SysML**
- PIM
- CIM
- SoaML

Modularność to:
- typ dekompozycji, w której wskazane części dobrze do siebie pasują
- **typ dekompozycji, w której łączymy podzespoły tworzące system**
- typ kompozycji, w której odwołujemy się do polimorfizmu wskazanych części
- typ kompozycji, w której zagregowane części są składane w całość

Na jakim poziomie modelu MDA opiszesz
- Oprogramowanie systemu **PSM**
- Wymagania **CIM**
- Organizacja biznesowa **CIM**

Określ jednym wyrazem jak w MDSD może być prowadzona dekompozycja **Rekursywnie**

Określ jednym wyrazem, jak w MDSD nazywa się najwyższy poziom modelu: **Context** (Context ->Analisis -> Design -> Implementation)

Określ kolejność działań w cyklu życia SOA: **Model, Montaż, Konfiguracja, Zarządzanie**

Określ liczbę punktów wejścia do SOA określoną w SOMA (wpisz liczbę): **5**

Określ liczbę widoków w modelu MDSD wiedząc, że dotyczny ona dwóch pierwszych poziomów modelu **11**

Określ poprawne powiązania (związki) między elementami modelu usług {<<ServiceInterface>>, <<Capability>>, <<Participant>>}: Wybierz co najmniej jedną odpowiedź
- Expose
- **Realization**
- Include
- Extend
- Generalization

Określ prawidłową notację tranzycji na diagramie maszyn stanowych UML: **wyzwalacz, wyzwalacz [wartość wyzwolenia] / akcja**

Określ stereotyp zależności między klasą ojciec a syn: **derive**

Określ wartość następującego wyrażenia OCL: -- a : Integer if a > 5 then a-5 else 0 endif
- 0
- Boolean.
- **Integer.**
- Wyrażenie nie ma wartości

Perspektywa architektury w SOA określa architektoniczny styl, który wymaga wystąpienia
- Klienta
- Wzorców
- Szkieletów
- Usługodawcy
- Opisu usług

Podaj nazwę ograniczenia atrybutu klasy pracownik, o nazwie NazwiskoRodowe, którego wartość nie może być zmodyfikowana po jej przypisaniu **readOnly**

Podaj nazwę ograniczenia atrybutu klasy pracownik o nazwie StanowiskoPracy, którego obiekty wewnątrz cechy mogą się powtarzać. **nonunique**

Ponieważ ograniczeniom zapisanym w **OCL** odpowiadając **asercje** w językach oprogramowania (C#: Debug.{**assert**}, Java:{**assert**}) to ich niespełnienie powinno skutkować:
- w języku programowania **wystąpieniem wyjątku**
- w UML'u **brakiem obsługi**

Prawdą jest, że:
- Architektura oprogramowania systemu ogranicza jego projekt i implementację
- Architektura oprogramowania systemu uszczegóławia wymagania na system oraz ułatwia procesy definiowania projektu
- Architektura oprogramowania systemu poprzedza działania analityczne i określa decyzje projektowe

Punkty wejścia do adaptacji SOA to **IT - Centric Entry Point,** które określają przyłączalność, wielokrotne użycie oraz: **Business Centry Entry Points** czyli Ludzie Procesy informacje

Rozszerzenie funkcjonalności klasy w trakcie działania programu a nie jego kompilacji umożliwia wzorzec: Wybierz co najmniej jedną odpowiedź
- Facade
- Message Flow
- Builder

- **Decorator**
- Proxy

Scenralizowany sposób zarządzania kompozycją usług, określa się jako (wpisz jeden wyraz) **orkiestracja**

Sposób na wystawienie spójnego interfejsu programistycznego to użycie wzorca:
- Facade
- Blackboard
- Composite
- Decorator
- Builder

Tworząc model architektury systemu koncentruj się na:
- Użyteczności
- Pięknie
- Trwałości
- Barwie
- Działaniu

Usługa w obiekcie uczestnik (Participant) jest reprezentowana przez: **port**

Usługi dostarczane przez usługodawców i widziane przez klientów powinny być **Gruboziarniste**

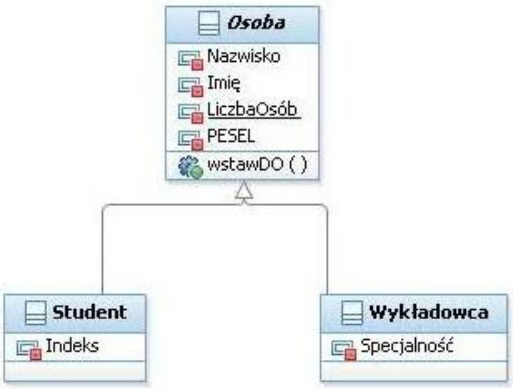
Użycie których związków na diagramie wymagań jest poprawne? **zagnieżdzenie, zależność**

Warstwy architektury oprogramowania są modelowane jako: Wybierz co najmniej jedną odpowiedź
- stereotypowane pakiety
- tory na diagramach aktywności
- słupki rozłączenia
- stereotypowane klasy
- zależności typu: <<layer>>

We wzorcu "obserwator": obserwowany wysyła komunikat do obserwującego kiedy zmieni się jego stan. **Prawda**

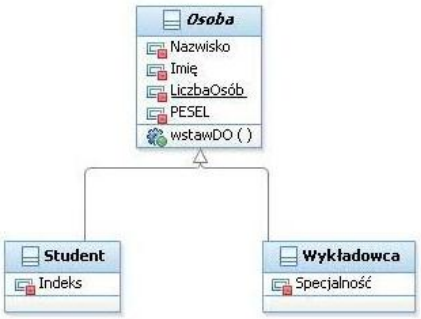
Wiedząc, że klasa Lista jest klasą szablonową, a klasa pracownik klasą powiązaną, to należy zastosować związek << >> (WPISZ NAZWĘ STEREOTYPU ZWIĄZKU), skierowany od klasy [pracownik] do klasy [Lista]. **bind**

Wiedząc że student może być wykładowcą, wpisz ograniczenie dotyczące specjalizacji



overlapping

Wiedząc, że w modelu może wystąpić klasa Dziekan, wpisz (bez zbędnych spacji) ograniczenie dotyczące specjalizacji:



isUnique

W języku UML w ver2.0 diagramy współpracy (collaboration diagram) zostały zastąpione przez diagramy: (wpisz jeden wyraz) **komunikacji**

W klasie Obywatel atrybut Pesel nie może zostać zmieniony. Jak to wyrazić w UML? Wpisz samą odpowiedź (bez uzasadnienia), wiedząc że będzie ona wstawiona w wierszu definicji atrybutu Pesel. **readOnly**

W MDA, PIM nie określa:
- Oprogramowania pośrednika (MiddleWare)
- Wymagań
- Uściślenia CIM
- Użytej infrastruktury
- Technologii

W modelu dziedzinowym poprawne jest użycie stereotypów klas: Wybierz co najmniej jedną odpowiedź
- Process
- Enumeration
- Provider
- Component
- Class
- Type

W modelu kandydatów na usługi (Capability) poprawne jest użycie
- Zależnosci derive
- Zależności include
- Zależnosci use
- Generalizacji
- Zależności extend
- Kompozycji
- Asocjacji

Wpisz nazwę wzorca, który realizuje poniższy kod: *class Wzorzec { private Wzorzec s; private int i; private Wzorzec(int x) { i = x; } public static Wzorzec getReference() { if (s == null) s = new Wzorzec(2); return s; } public int getValue() { return i; } public void setValue(int x) { i = x; } }*
Singleton

W modelu aktywności sygnał jest obiektem ze stereotypem **signal**

Wpisz nazwiska twórców manifestu MDA którzy współtworzyli UML **Rumbaugh, Booch**

Wpisz nazwy ograniczeń lub ograniczenia generalizacji między klasą Pracownik, a klasami Pracownik etatowy, Pracownik nieetatowy. Do wyboru z disjoint, overlapping, complete, incomplete. d**isjoint,incomplete**

Wpisz samą odpowiedź (bez zbędnych spacji), jak dla atrybutu "Pole : Integer" klasy Prostokąt, wprowadzisz jej uszczegółowienie in-line (w definicji atrybutu klasy) wiedząc, że pole jest większe od zera: **--pole jest większe od zera**

Wpisz samą odpowiedź, jak dla operacji je(pożywienie: warzywa) klasy Osoba, wprowadzisz jej uszczegółowienie in-line: wegetarianin **--wegetarianin**

W podejściu translacyjnym (w MDA) udział człowieka ogranicza się do definicji modeli
- PSI
- **PIM**
- PSM **(?)**

W podejściu translacyjnym (w MDA) wykryty błąd w systemie na poziomie PIM skutkuje koniecznością poprawy w **PIM**

W relacjach między wymaganiami (na diagramie wymagań, które ze związków są poprawne **Zagnieżdżenie i zależność**

Wskaż wzorce behawioralne
- **State**
- Decorator
- **Iterator**
- **Medioator**
- Facade

Współpracę pomiędzy dwoma klasami o niekompatybilnych interfejsach umożliwia wzorzec: **Adapter**

W UML rozszerzenia definiowane są przez: Stereotypy Metki Ograniczenia i **profile**

Wybierz wzorce strukturalne:
- Builder
- **Adapter**
- Visitor
- **Proxy**
- **Facade**
- State
- **Decorator**
- Interpreter

Wynikiem identyfikacji procesów biznesowych w procesie SOMA, są: Wybierz co najmniej jedną odpowiedź:
- **Cele biznesowe**
- Encje biznesowe
- **Reguły biznesowe**
- Zadania biznesowe
- **Biznesowe przypadki użycia**
- Pracownicy

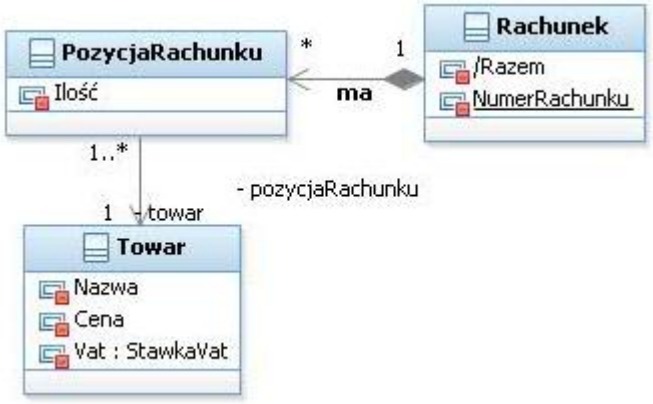
Wynikiem projektowania komponentów usług w procesie SOMA, są:
- **Dostawcy usług**
- Klienci usług
- **Komponenty usług**
- **Kanały usług**
- **Model usług**
- Architektura usług
- **Komunikaty**

Wzorzec "fabryki abstrakcyjnej" pozwala na stworzenie tylko jednego obiektu danej klasy. **Prawda**

Wzorzec proxy może być zastosowany w następującym modelu. Wybierz co najmniej jedną odpowiedź
- **Zdalny**
- **Wirtualny**
- Czekający
- **Ochraniający**
- Odpytujący
- Kompromisowy

- **Sprytne odwołanie**
- Wywłaszczający

Z analizy poniższego modelu wynika, że:



- NumerRachunku: **atrybut statyczny**
- Razem: **atrybut pochodny**
- Ilość: **atrybut**
- StawkaVAT: **dziedzina**
- Towar: **klasa**

Zapisz ograniczenie dotyczące sygnatury metody klasy Pracownik: zmieńDane(in danePracownika[2] : daneP); wiedząc, że nie może ona (metoda) modyfikować stanu obiektu klasy Pracownik Uwaga, Usuń z odpowiedzi wszystkie spacje. **isQuery**

Zaznacz prawidłowe stwierdzenie opisujące słowo self języka OCL
- **Jest to słowo kluczowe**
- Reprezentuje typ określony przez kontekst wyrażenia
- **Reprezentuje egzemplarz typu określonego przez kontekst wyrażenia**
- Jest to zmienna

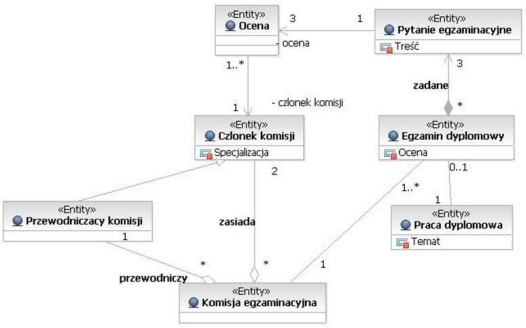
Zdecenralizowany sposób zarządzania kompozycją usług, określa się jako (wpisz jeden wyraz): **choreografia**

Aktor reprezentuje spójny zbiór ról, które są odgrywane przez użytkowników przypadku użycia: **Prawda**

Analizując poniższy diagram określ, czy stanowi on poprawny model analityczny (dla warstwy danych) czy nie? **Nie**

Analizując poniższy model, udziel odpowiedzi na następujące pytanie:

- Jeśli podczas obrony egzamin został odwołany, to ile pytań zadano podczas egzaminy, gdy zamienimy związki agregacji na asocjacje? **3**



- Zmodyfikuj zapis krotności tak, aby egzamin dyplomowy składał się z dokładnie 3 ponumerowanych i niepowtarzających się pytań. **3{ordered}**

Aplikacja wzorca projektowego do danego modelu oznacza realizację pewnej transformacji. Jej przykładem może być transformacja: **kodu do kodu, UML do UML**

Aplikacja wzorca projektowego do danego modelu oznacza realizację pewnej transformacji. Jej przykładem może być transformacja: **Modelu projektowego do kodu**

Audyt projektu, to jego ocena dokonana przez: **niezależny zespół**

BNF jest to: **zapis modelu struktury**

BNF to: **notacja zapisu modelu struktury**

Co oznacza tylda (~) przy atrybucie klasy **widoczność dla wszystkich elementów zawartych w tym samym pakiecie co klasa**

Co przedstawia diagram sekwencji: **Kolejność w jakiej obiekty współdziałają ze sobą w trakcie realizacji przypadku użycia.**

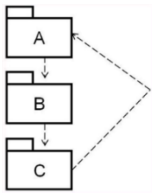
Co wchodzi w skład biznesowego modelu analitycznego? **system biznesowy (business system), realizacje biznesowych przypadków użycia, zasady biznesowe (business rules)**

(business rules), pracownicy biznesowi (business workers)

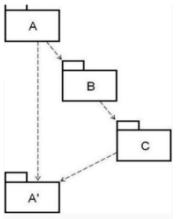
Czy dozwolone są asocjacje pomiędzy aktorami: **Fałsz**

Czy na diagramie wymagań SysML można zamodelować wymagania niefunkcjonalne: **Prawda**

Czy przedstawiony na diagramie model jest poprawny? **Fałsz**



Czy przedstawiony na diagramie model jest poprawny? **Prawda**



Czy usunięcie elementu z diagramu powoduje usunięcie elementu z modelu? **fałsz**

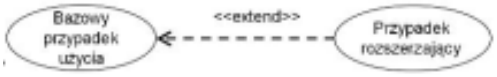
Czy usunięcie elementu z modelu powoduje usunięcie tegoż elementu z diagramu? **prawda**

Czy w węźle wyboru na diagramie aktywności można modelować więcej niż 2 wyjścia: **prawda**

Czy z interfejsu można powołać obiekt? **Fałsz**

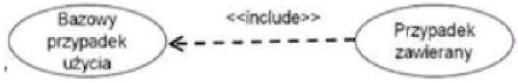
Czy zależność jest skierowana poprawnie? Prawda

Czy zależność na rysunku jest skierowana poprawnie?



Prawda

Czy zależność na rysunku jest skierowana poprawnie?



Fałsz

Dekompozycja i abstrakcja służy: **opanowaniu złożoności problemów projektowych**

Diagram komunikacji przedstawia: **interakcje**

Dlaczego modelujemy?: **aby pomóc zwizualizować system, aby dostarczyć szablon dla budowania systemu, aby dokumentować podjęte decyzje, aby redukować złożoność projektu**

Do czego może najlepiej posłużyć format XMI: **do wymiany danych pomiędzy różnymi narzędziami UML**

Do grupowania elementów służą: **Pakiety**

Do zorganizowania elementów w grupy użyłbyś? **Pakietów**

Dokument "Wizja" opracowywany jest przez **Zespół Projektowy**

Dopasuj charakterystyki do terminów. Klasa...

- odpowiadająca za komunikację z otoczeniem systemu to **klasa graniczna (boundary)**
- odpowiadająca za logikę biznesową aplikacji to **klasa sterująca (control)**
- odpowiadająca za przechowywanie informacji to **klasa danych (entity)**

Dopasuj diagramy do opisu:

- ilustruje interakcje otoczenia systemu z systemem: **diagram przypadków użycia**
- ilustruje logiczną strukturę systemu **diagram klas**
- ilustruje umieszczenie komponentów oprogramowania w architekturze sprzętowej **diagram wdrożenia**
- ilustruje przepływ zdarzeń **diagram aktywności**
- ilustruje interakcję pomiędzy obiektami **diagram komunikacji (współpracy)**
- ilustruje przebieg w czasie komunikatów pomiędzy obiektami: **diagram sekwencji**

Dopasuj diagramy do opisu:

- ilustrują logiczną strukturę systemu: **diagram klas**
- ilustruje przebieg w czasie komunikatów pomiędzy obiektami: **diagram sekwencji**
- ilustruje sekwencję (kolejność) przepływających komunikatów pomiędzy obiektami: **diagram komunikacji**
- ilustruje rozmieszczenie artefaktów w strukturze systemu (jego węzłach): **diagram wdrożenia**
- stanowi meta scenariusz: **diagram aktywności**
- ilustruje wartości dostarczane przez system na rzecz jego otoczenia: **diagram przypadków użycia**

Dopasuj odpowiednią cechę do właściwej kategorii mechanizmów architektonicznych:

- Mechanizmy analityczne **konceptualne**
- Mechanizmy wymagań **brak**
- Przykład mechanizmu implementacyjnego **rozproszenie**
- Mechanizmy projektowe **konkretne**
- Przykład mechanizmu projektowego **baza danych**
- Przykład mechanizmu analitycznego **COBRA**
- Mechanizmy implementacyjne **aktualne**

Dopasuj odpowiedź. W modelu zależności między aktorami systemu sprzedaży użyty zostanie związek:

- Administrator, użytkownik **generalizacji**

- Sprzedawca Klient **komunikacji**
- Kooperant, Klient **żaden**

Dopasuj odpowiedź

- W modelu UC, związki między aktorami systemu obsługi wypożyczenia książek (Bibliotekarz, Kierownik wypożyczalni) **użyję związek - komunikuje się**
- W modelu UC, związki między aktorami systemu rejestracji na studia (Dziekanat, Kandydat) **użyję związek - komunikuje się**
- W modelu UC, związki między aktorami systemu sprzedaży (Kooperant, Klient) **aktorzy nie będą ze sobą powiązani**

Dopasuj przykłady do wymagań.

- System umożliwi współpracę z systemem bankomatowym Sieci Banknetu przez sieć Elzam. **FEAT (cecha)**
- Przeglądanie dokumentów. **UC (w funkcjonalne)**
- System nie może zajmować więcej niż 4Mb pamięci. **SR, NFR (niefunkcjonalne)**
- Skrócenie czasu dostępu do dokumentów magazynowych **STRQ (żądanie udziałowca)**

Dopasuj przykłady do wymagań.

- Przyjmij zapłatę **Wymaganie funkcjonalne (UC)**
- System dostarczy nowoczesny interfejs o bardzo któtkim czasie dostępu do przetwarzanych informacji **Błędne wymaganie**
- Skrócenie czasu obsługi studenta podczas jego rejestracji **Żądanie udziałowca (STRQ)**
- System umożliwi wyświetlenie informacji o ocenach studentów. Student uzyska dostęp tylko do swoich ocen. Oceny będą wyświetlone w postaci graficznej. **Wymaganie typu cecha (FEAT)**
- System nie może zajmować więcej niż 4mb pamięci. **Wymaganie niefunkcjonalne (SR)**

Dopasuj w logiczną całość poniższe wyrażenia wiedząc, że dotyczą one modelu analitycznego:

- odpowiada za komunikację z otoczeniem systemu: **boundary**
- odpowiada za logikę biznesową aplikacji: **control**
- odpowiada za przechowanie informacji: **entity**
- stanowi element modelu, o semantyce określonej przez specjalizowany profil (poza base UML): **player**

Doprecyzuj krotność związku 0..5{} w języku UML2.0 wiedząc, że dotyczy ona asocjacji samochód osobowy - Koło po stronie Koło **5**

Do uwypuklenia następujących aspektów, zastosujesz odpowiednio:

- upływ czasu, (2) struktura powiązań: **(1) d.sekwencji (2) d.współpracy**
- trwanie działań, (2) przepływ danych **(1) d.stanów (2) d.aktywności**
- upływ czasu,(2) współbieżność działań **(1) d.sekwencji (2) d.aktywności**

Do zaprojektowania systemu obsługi nadzoru elektrowni jądrowej należy posłużyć się cyklem życia: **kaskadowy**

Do zorganizowania elementów w grupy użyłbyś? **Pakietów**

Enkapsulacja... **Często jest przedstawiana jako ukrywanie informacji**

Identyfikacja klas metodą lingwistyczną polega na wykonaniu następujących czynności: **wykreśl powtarzające się rzeczowniki, wykreśl rzeczowniki nazbyt opisowe, podkreśl wszystkie rzeczowniki w tekście źródłowym**

Ile klas granicznych (zgodnie z RUP) przypada na każdą realizację przypadku użycia, jeżeli liczba aktorów z nim powiązanych wynosi 3? **3**

Ile klas sterujących (zgodnie z RUP) przypada na każdą realizację przypadku użycia (w modelu VOPC)? **1**

Ile osób liczy komisja egzaminacyjna? **3**

Ile punktów wejściowych do SOA jest określonych w SOMA? **5**

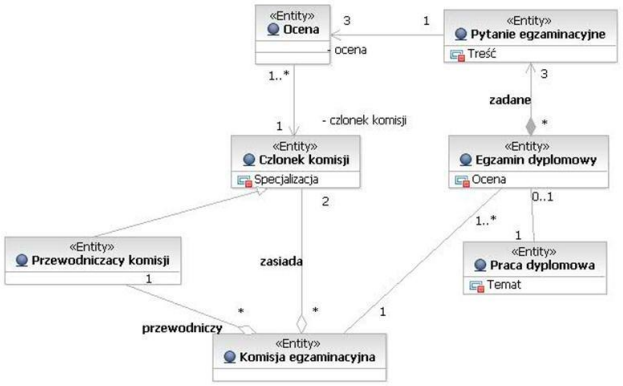
Jakimi nawiasami należy ująć parametr ograniczenia? **{}**

Jak na diagramach UML2.0 reprezentowane są parametry wzorców projektowych? (co najmniej 1 odp.) **Jako stereotypowany pakiet**

Jak na diagramach UML2.0 reprezentowane są wzorce projektowe? **Jako parametryzowana współpraca**

Jeśli podczas obrony egzamin został odwołany, to ile pytań zadano podczas egzaminu: **0**

Jeśli podczas obrony egzamin został odwołany, to ile pytań zadano podczas egzaminu gdy zmienimy związki kompozycji na asocjacje: **3**



Jeżeli wielokrotność jest związana z elementem, którego notacja jest wyrażona ciągiem znakowym (np. atrybut) wówczas UML pozwala na określenie wielokrotności w obrębie tego ciągu znakowego. Jakimi nawiasami należy ująć parametr wielokrotności? **[]**

Kiedy zastosujesz diagram sekwencji (1) a kiedy diagram aktywności (2)?

- **Dla przedstawienia pojedynczego przebiegu; (2) Dla przedstawienia meta przebiegu,**
- **(1), Dla przedstawienia interakcji między obiektami; (2) Dla przedstawienia odpowiedzialności ról określonych przez tory,**

Kiedy zastosujesz diagram sekwencji (1) a kiedy diagram współpracy(2)?

- **(1)Dla uwypuklanie aspektu upływu czasu. (2) Dla przedstawienia działań zachodzących współbieżnie..**
- **(1)Dla uwypuklanie trwania działań. (2) Dla uwypuklenia przepływów danych.**

Klasa to **abstrakcyjna definicja obiektu**

Kompozycja jest to zależność: **agregacji określająca czas życia „części"**

Konstrukcja modelu zachowania obiektu posługując się UML odbywa się poprzez wyszczególnienie jego... **operacji**

Która fraza najlepiej opisuje relację generalizacji? **Jest rodzajem.**

Która rola z wymienionych poniżej pozwala na modelowanie, symulacje i monitorowanie procesu wytwórcy oprogramowania: **analityk**

Które typy wyrażeń OCL służą do definiowania zmiennych pomocniczych? **let, def**

Które z wymienionych są diagramami modelującymi statykę systemu: **klas, komponentów, struktur złożonych, wdrożenia, obiektów, pakietów**

Które z wymienionych są diagramami modelującymi zachowanie systemu: **aktywności, interakcji, przypadków użycia, maszyny stanów**

Które z zasad modelowania są poprawne? **Model, który tworzysz wpływa jak problemy będą rozwiązywane, Najlepsze modele to te, które pozwolą na wybór poziomu szczegółowości, Najlepsze modele związane są z opisem rzeczywistości**

Którego ze stereotypów najlepiej użyć do zamodelowania pokrycia systemu testami na diagramami wymagań **Verifies**

Model UC nie jest poprawny jeżeli: **każdy aktor nie jest powiązany z przynajmniej jedną usługą lub każda konkretna usługa nie jest powiązana z przynajmniej jednym aktorem, istnieje aktor nie powiązany z przynajmniej jedną realizacją usługi**

Modelujemy aby: **dokonań zobrazowania, dokumentować podjęte decyzje, dostarczyć szablon do budowy rozwiązania**

Modularność systemu... **Rozbija złożoność systemu na kierowalne części**

Na diagramach usług można użyć związków: **Generalizacji, Realizacji, Extend**

Na diagramach usług niepoprawne jest użycie zależności: **kompozycji, agregacji**

Na diagramie stanów zmiana stanu systemu jest możliwa, gdy: **wystąpiło zdarzenie określone dla danego przejścia między stanami i spełnione są ograniczenia.**

OCL to: **Rozszerzanie UML o możliwość doprecyzowania ograniczeń**

Odpowiedzialność klasy: **definiują jej operacje, to atrybuty i metody**

Odpowiedzialność klasy jest zdefiniowana przez posiadane: **operacje**

Określ prawdziwość zdania: Przedstawiony na diagramie związek jest poprawny. **Fałsz**

Określ prawdziwość zdania: Przypadek bazowy nie wie o usłudze rozszerzającej: **Prawda**

Określ prawdziwość zdania: Związek widoczny na diagramie jest określony poprawnie: **Fałsz**

Określ stereotyp zależności między klasą bank a bankomat: **use**

Określ stereotyp zależności między klasą ojciec a syn: **create**

Określ typ, nazwę, krotność związku i rolę klasy w tym związku na diagramie klas pomiędzy klasami: Wykładowca i Wydział Uczelni: **1...2, Dziekan, asocjacja, zatrudniony**

Określ: typ związku, nazwę związku, krotność na diagramie klas między klasą: wydział uczelnie a wykładowca **asocjacja, zatrudniony, 0…n**

Polecenie przelewu z operacją bankową łączy związek **generalizacja**

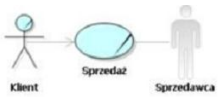
Polimorfizm można scharakteryzować jako: **ukrywanie wielu różnych implementacji za jednym interfejsem** , **jest dziedziczeniem**

Poprawność modelu usług jest uwarunkowana spełnieniem warunku: **każdy aktor powiązany jest przynajmniej z jedną usługą a każda usługa z przynajmniej jednym aktorem**

Pożądane cechy specyfikacji wymagań to: **jednoznaczność, kompletność, spójność wymagań**

Przedstaw kluczowy argument uzasadniający potrzebą modelowania SI? **Złożoność SI, dlatego model stanowi uproszczenie rzeczywistości.**

Przyjrzyj się uważnie poniższemu diagramowi. Następnie zaznacz poprawne stwierdzenie: diagram jest **błędny**



Repozytorium projektu to: **baza danych projektowych**

Scenariusz UML: **pozwala na opisanie interakcji między obiektami, opisanie różnych interakcji na jednym diagramie**

Scenariusz w UML **Opisuje diagram współdziałania, przedstawia jeden konkretny przebieg**

Specyfikacja wymagań powinna być: **jednoznaczna, kompletna i spójna**

Stan obiektu... **Jest zdefiniowany przez wartości cech obiektu i relacje z innymi obiektami**

Stereotyp include: **określa strukturalną zależność między usługami, tworzy tzw. grupę wielokrotnego użycia**

Testy akceptacyjne wytwarzane są zaraz po: **zatwierdzeniu wymagań na system**

Testy jednostkowe wytwarzane są zaraz po: **implementacji każdej konstrukcji**

Tranzycja występuje: **Na diagramach stanów**

Typ wymagania w RUP: **Przypadek użycia (UC), Cecha systemu (FEAT), Żądania udziałowca (STRQ),Słownik (TERM)**

UML to: **język programowania, sposób komunikowania**

W architekturze trójwarstwowej klasa danych łączy się bezpośrednio z klasą interfejsu: **fałsz**

W której fazie cyklu życia systemu opracowany jest model organizacji Zamawiającego: **analizy biznesowej**

W modelu zależności pomiędzy aktorami systemu sprzedaży (kooperant, klient) **aktorzy nie będą ze sobą powiązani**

W modelu zależności pomiędzy aktorami systemu sprzedaży (sprzedawca, klient) **użyje związek komunikuje się**

W modelu zależności pomiędzy aktorami systemu Windows (administrator, użytkownik) **użyje generalizacji**

Wskaż które z poniższych określeń stanowi typ wymagania w RUP: **przypadek użycia (UC), cecha systemu (FEAT), żądania udziałowca (STRQ), słownik (TERM)**

Wskaż typ wymagań (zgodnych z RUP), które bezpośrednio nie stanowią opisu budowanego systemu: **SR, CRUD, FURPS**

W modelu aktywności sygnał jest obiektem ze stereotypem **signal**

W OCL co oznacza self: **egzemplarz typu określony przez konkretne wyrażenie, jest to słowo kluczowe**

W systemie obsługi bankomatu dopasuj odpowiedzi:

- Wpłatę z wypłatą łączy związek **brak związku**
- Wypłatę z poleceniem przelewu łączy związek **brak związku**

- Wypłatę z operacją bankową łączy związek **generalizacja**
- Polecenie przelewu z operacją bankową łączy związek: **gerneralizacja**

Wybierz poprawne:

- mechanizmy implementacyjne – **konkretne**
- mechanizmy wymagań –**konkretne**
- mechanizmy analityczne – **konceptualne**
- mechanizmy projektowe – **konceptualne**

Wybierz poprawne stwierdzenie. Klasa. **jest to abstrakcyjna definicja obiektu**

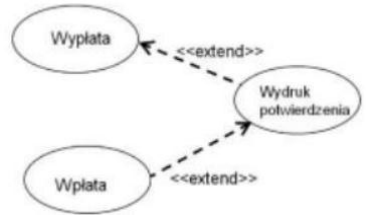
Wymagania funkcjonalne: **określają jak system ma się zachowywać w określonych sytuacjach, pochodzą z dziedziny zastosowania, mogą dotyczyć dziedziny przedsięwzięcia**

Zaznacz te pary spośród niżej podanych usług, które uczestniczą w relacji generalizacji. **Operacja bankowa, Polecenie przelewu, Operacja bankowa, wypłata**

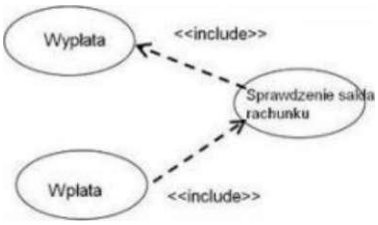
Zaznacz poprawne stwierdzenie. Modularność systemu. **rozbija złożoność systemu na kierowalne części**

Związek całość-część obłożony zastrzeżeniem, że część nie może istnieć poza całością to: **kompozycja**

Związek include: **określa strukturalną zależność między usługami, tworzy tzw, grupę wielokrotnego użycia**



- Zależność do wypłaty jest skierowana **poprawnie**
- Zależność od wpłaty jest skierowana **niepoprawnie**
- Zależność do wydruku potwierdzenia jest skierowania **niepoprawnie**
- Zależność od wydruku potwierdzenia jest skierowana **poprawnie**



- Zależność do wypłaty jest skierowana **niepoprawnie**
- Zależność od sprawdzenia salda rachunku jest skierowana **niepoprawnie**
- Zależność do sprawdzenia salda rachunku jest skierowania **niepoprawnie**
- Zależność od wpłaty jest skierowana **niepoprawnie**

Związek klas, w którym jedna składa się z innych i nie jest egzystencjalnie odpowiedzialna za swoje części to: **agregacja**

PYTANIA BEZ ODPOWIEDZI LUB PODAWANE Z PAMIĘCI

czym nazywamy usługi: **WSDL**
czym opisujemy uslugi: **XML**
jaki protokół wykorzystujemy do wysyłania usług: **SOAP**
za pomocą jakiego rejestru możemy szukać usług po nazwach(WSDL'ach): **UDDI**

Z każdą notką biograficzną (klasa: Notka) pracy dyplomowej (klasa Dyplom) związane są słowa kluczowe (klasa Klucz). Notka zawiera statyczny atrybut Ilosc, który jest zwiększany dla każdej nowej notki. Korzystając z języka OCL, zapisz następujące ograniczenie: - liczba wszystkich słów kluczow. Kształt systemu określa wynik modelowaniaych jest nie mniejsza niż liczba notek biograficznych (ilosc). **????**

Zapisz ograniczenie dotyczące sygnatury metody klasy Pracownik: zmieńDane(in danePracownika[2] : daneP); wiedząc, że nie może ona (metoda) modyfikować stanu obiektu klasy Pracownik Uwaga, Usuń z odpowiedzi wszystkie spacje. **Pracownik: zmieńDane(in danePracownika[2] : daneP){readOnly} ???**

SŁOWNICZEK

Agregacja – rodzaj powiązania, które określa związek całość-część między agregatem (całością) a składnikiem (częścią), przy czym część może należeć do wielu całości; związek między klasą a komponentami

Aktor – rola rozpoczynająca interakcję,

Aktor – w odniesieniu do UML jest rozumiany jako rola grana przez zewnętrzny podmiot (użytkownika, inny system, sprzęt, itp.) będący w interakcji z opisywanym systemem.

Aktywacja – przedstawianie operacji wykonywanej przez obiekt; długość określa czas jej trwania;

Artefakt – w odniesieniu do UML rozumiany jest jako element (przedmiot) wytworzony podczas procesu projektowania pomagający opisać strukturę lub zachowanie obiektu, systemu bądź samego procesu projektowania.

Asocjacja – odzwierciedla relację pomiędzy dwoma elementami modelu.

Asocjacja skierowana – asocjacja wskazująca kierunek przepływu sterowania lub danych.

Atrybut – nieodłączna cecha klasy obiektów, określająca zbiór wartości, jakie można przypisać do poszczególnych jego egzemplarzy. Ogólna postać zapisu atrybutu jest następująca:

widoczność nazwa : typ [krotność] {ograniczenia}=wartość domyślna,

Szczególnym przypadkiem jest atrybut wywiedziony (pochodny), który jest zależny od innych atrybutów i jego wartość zazwyczaj oblicza się na podstawie wartości innych atrybutów. Na diagramach UML oznacza się go ukośnikiem '/' poprzedzającym jego nazwę,

Bramka – element stanowiący łącznik pomiędzy diagramami sekwencji lub ich fragmentami. Jest elementem granicznym, który łączy interakcję z jej uszczegółowiającym fragmentem.

Diagram klas – podstawowy diagram UML służący do przedstawienia struktury systemu, ukazuje klasy i związki między nimi oraz z innymi klasyfikatorami,

Diagram obiektów – pomocniczy diagram UML służący do przedstawienia struktury systemu, przez wskazanie konkretnych egzemplarzy klas i związków między nimi (zwanych wiązaniami); diagram, który ukazuje możliwą konfigurację obiektów w określonym momencie działania systemu,

Diagram przypadków użycia – graficzny opis funkcjonalności opisywanego systemu w zakresie interakcji aktorów z systemem, które służą realizacji celów aktorów (reprezentowanych przez przypadki użycia) oraz ukazaniu zależności między przypadkami użycia. Diagram ten stanowi jedynie pogląd na wymagania.

Diagram sekwencji (diagram przebiegu) – jeden z diagramów zachowania w UML, pokazujący komunikację pomiędzy obiektami w celu realizacji poszczególnych funkcji systemu. Przedstawia scenariusz będący rozwinięciem przypadku użycia, uwypuklając kolejność w jakiej obiekty współdziałają ze sobą w trakcie jego realizacji;

Dziedziczenie – mechanizm, dzięki któremu byty szczegółowe przejmują strukturę i zachowanie bytów ogólnych; diagramy UML ukazują hierarchię dziedziczenia,

Fragment (**blok**) – zamknięta części diagramu sekwencji, rozszerzająca możliwości obejmowanego przez siebie obszaru tego diagramu. Za pomocą fragmentów można przedstawiać: pętle, powtórzenia, scenariusze alternatywne, zrównoleglenia, wskazywać poziom abstrakcji modelowanego fragmentu itp. UML określa następujące rodzaje fragmentów:

alt – dzieli fragment interakcji na dwa alternatywne scenariusze; każda z alternatyw musi być opatrzona warunkiem dozoru, którego spełnienie gwarantuje wykonanie danej alternatywy,

assert – prezentuje fragment interakcji, który musi być wykonany zgodnie z założonymi warunkami i komunikatami,

break – wskazuje fragment diagramu sekwencji, który realizowany jest po spełnieniu warunku dozoru; spełnienie warunku dozoru skutkuje wykonaniem sekwencji zawartej we fragmencie, a następnie wyjściem ze scenariusza; w przypadku, gdy warunek dozoru nie jest spełniony, sekwencja opisana we fragmencie jest pomijana,

consider – wskazuje fragment z listą nazw komunikatów, które są wyselekcjonowane w tej części interakcji; oznacza to, że mimo innych komunikatów, które znajdują się w danej części interakcji, pokazane zostaną tylko te, które są wylistowane ze słówkiem kluczowym consider,

critical – wskazuje, że dany fragment diagramu sekwencji nie może być przerwany przez inny proces,

ignore – wskazuje, że w tym fragmencie interakcji znajdują się wiadomości, które zostały pominięte, gdyż ich widoczność nie zmienia zachowania systemu; zignorowane wiadomości są wylistowane ze słówkiem kluczowym ignore;

loop – powtórzenie fragmentu interakcji określoną (warunkiem) liczbę razy ,

neg – fragment prezentujący jedną lub więcej wiadomości, które są prawdopodobnie nieprawidłowe,

opt – wskazuje opcjonalny fragment interakcji, który jest wykonywany po spełnieniu warunku dozoru,

par – wskazuje równoległy przepływ komunikatów,

seq – prezentuje słabo uszczegółowiony (ogólny) fragment sekwencji,

stricte – prezentuje uszczegółowiony fragment sekwencji.

FURPS (and. Functionality, Usability, Reliability, Performance, Supportability) – akronim pomagający zapamiętać różnicę między wymaganiami funkcjonalnymi (F) a niefunkcjonalnymi (URPS).

Granice systemu – reprezentują typ podziału pomiędzy aktorami a częścią systemu (bądź całym systemem) reprezentowaną przez przypadki użycia.

Interfejs – zestaw operacji, które wyznaczają usługi wyznaczane przez komponent (lub klasę). Najczęściej wykorzystywane są do prezentowania komunikacji między komponentami; element notacji UML stanowiący deklarację operacji zdefiniowanych w innej klasie (lub komponencie),
kierunek [in, out, inout, return] nazwa typ [krotność]=wartość domyślna,

Klasa – opis zbioru obiektów, które mają takie same atrybuty, operacje, związki i znaczenie; szablon do tworzenia obiektów; opis kategorii obiektów,

Klasa abstrakcyjna – klasa deklarująca wspólne cechy grupy klas, jednak nie mogąca posiadać swoich egzemplarzy,

Klasa danych – reprezentuje przechowalnie danych wykorzystywanych w systemie (nie udostępnia przetwarzania).

Klasa graniczna – reprezentuje interfejs udostępniający funkcje systemu bytom poza systemem (odpowiedzialna jest zatem za komunikację),

Klasa powiązań (asocjacyjna) – byt o właściwościach zarówno klasy, jak i powiązania,

Klasa sterująca – reprezentuje proces w systemie (przetwarzanie), jest łącznikiem pomiędzy klasami interfejsu i danych. Na każdy przypadek użycia przypada jedna klasa sterująca (musi zatem wystąpić na diagramie sekwencji),

Klasyfikator – blok konstrukcyjny, który służy do opisania cech strukturalnych i czynnościowych. Klasyfikatorami są klasy, interfejsy, typy danych, sygnały, komponenty, węzły, przypadki użycia i podsystemy,

Kompozycja (agregacja całkowita) – rodzaj agregacji, który charakteryzuje się tym, że część należy wyłącznie do jednej całości i czas jej życia jest ściśle uzależniony od czasu życia całości;

Komunikat - informacja przekazywana bezpośrednio pomiędzy obiektami, zazwyczaj przenosząca zlecenie wykonania określonej czynności;

Kwalifikator – w UML jest to informacja identyfikująca (identyfikator), wyrażona symbolem niewielkiego prostokąta, umieszczona na linii powiązania od strony klasy dokonującej wyszukiwania,

Liczność (**krotność**) **roli powiązania** – służy do określenia ile obiektów może być połączonych przez dane powiązanie:

1 – dokładnie jeden,

0..1 – opcjonalnie jeden,

1..* – przynajmniej jeden,

1,3,5 – konkretna liczba obiektów,

dolna granica..górna granica – przedział od – do,

* – dowolna liczba obiektów.

Linia życia – reprezentacja roli uczestnika interakcji, określająca interesujący, z punktu widzenia modelowania, czas życia obiektu,

Metka – jeden mechanizmów profilujących w UML, stanowiący znacznik, który pozwala zmienić wartość danego elementu projektowego (nie jest atrybutem ale metadaną) – np. {liczba_procesorów=3}, {komponent_występuje_tylko_na_serwerze}

Metoda – implementacja (realizacja) operacji,

Model trójwarstwowy – struktura logiczna aplikacji (głównie webowych), w której wyróżnia się warstwę danych, warstwę logiki aplikacji (sterującą) oraz warstwę prezentacji (interfejsu użytkownika). W języku UML model taki przedstawia się z wykorzystaniem następujących elementów projektowych:

klasy danych «entity»

klasy sterującej «control»

klasy granicznej(interfejsu) «boundary»

Nawigowalność – określa wiedzę o sobie nawzajem obiektów uczestniczących w relacji,

Obiekt (egzemplarz) – konkretny reprezentant klasy; konkretne wystąpienie klasy, egzemplarz należący do kategorii,

Ograniczenia dotyczące atrybutów:

{ordered} – obiekty wewnątrz cechy są uporządkowane,

{unordered} – obiekty są nieuporządkowane,

{unique} – obiekty wewnątrz cechy nie powtarzają się,

{nonunique} – obiekty wewnątrz cechy mogą się powtarzać,

{readOnly} – wartość atrybutu służy tylko do odczytu,

{frozen} – wartość atrybutu nie może być zmodyfikowana po jej przypisaniu.

Ograniczenie – jeden mechanizmów profilujących w UML, stanowiący warunek, który musi zostać spełniony, by model mógł być uznany za poprawny (dany element konstrukcyjny ma sens) – np. {połączenie_szyfrowane}, {osoba.żona.płeć = kobieta}

Operacja – usługa, którą oferuje klasa; czynność, którą klasa może wykonać lub którą na danej klasie może wykonać inna klasa. Ogólna postać zapisu operacji jest następująca:

widoczność nazwa (parametr1, parametr2, …) : typ {ograniczenia}, przy czym parametry można precyzować w sposób następujący:

kierunek [in, out, inout, return] nazwa typ [krotność]=wartość domyślna,

Pakiet – element notacji UML, służący do grupowania elementów modelu w dowolnym celu

Podsystem – grupa bytów, z których część stanowi specyfikację zachowania pozostałych; element notacji UML stanowiący skrzyżowanie pakietu z klasą, realizujący co najmniej jeden interfejs opisujący jego zachowanie,

Powiązanie (**asocjacja**) – związek strukturalny określający zbiór połączeń między obiektami; związek znaczeniowy między co najmniej dwoma klasyfikatorami, który określa połączenia między egzemplarzami. Asocjacje w UML są równoważne atrybutom klas, zwykle jednak atrybut reprezentuje typ prosty (lub wartość), asocjacja natomiast typ złożony,

Poziomy widoczności:

- public (ozn. +) – element jest widoczny z każdego miejsca w systemie,
- private (ozn. –) – element jest widoczny tylko we własnej klasie,
- protected (ozn. #) – element jest widoczny we własnej klasie i w klasach potomnych,
- package (ozn. ~) – element jest widoczny tylko wewnątrz pakietu

Profile UML - mechanizmy rozszerzeń języka UML, które pozwalają dopasować model do konkretnej dziedziny lub platformy. Do tych mechanizmów zalicza się: stereotypy, metki i ograniczenia;

Projekt – techniczna reprezentacja czegoś co należy zbudować.

Przypadek użycia – w odniesieniu do UML jest rozumiany jako zachowanie opisywanego systemu, będące sekwencją czynności wykonywanych przez system, gwarantującą aktorowi wykonanie przez system usługi realizującej cel aktora. Innymi słowy, każdy przypadek użycia opisuje funkcjonalność systemu.

Role obiektów określane są przez typy klas:

Rozszerzenie – relacja zależności wskazująca, że dany przypadek użycia może być poszerzony o dodatkowe zachowanie, zdefiniowane w rozszerzającym przypadku użycia. Rozszerzenia wykorzystuje się do modelowania reakcji na spełnienie warunków, występowanie sytuacji wyjątkowych, lub błędów.

Scenariusz – sekwencja czynności wykonywanych przez system, gwarantująca aktorowi wykonanie przez system usługi realizującej cel aktora; w UML stanowi rozwinięcie przypadku użycia;

Sekwencja – układ elementów następujących po sobie w określonej kolejności, stanowiący pewną zamkniętą całość;

Specyfikacja wymagań – zbiór wymagań funkcjonalnych; opisuje funkcjonalność produktu. Metodą specyfikacji wymagań są np. przypadki użycia.

Stereotyp – jeden mechanizmów profilujących w UML, stanowiący znacznik, który pozwala zmienić znaczenie danego elementu projektowego (tworzyć nowe elementy wywodzące się z już istniejących) – np. «entity»; często reprezentowany jest przez piktogram;

Uogólnienie (Jestem w rodzaju …) – związek pomiędzy dwoma bytami: ogólnym (przodkiem) i szczegółowym (potomkiem). Obiekt bytu szczegółowego może być używany w zastępstwie obiektu bytu ogólnego;

VOPC (ang. View of Participating Classes) – widok klas uczestniczących – diagram klas, który zawiera wszystkie klasy, które biorą udział realizacja przypadku użycia (tj. które występują na diagramie sekwencji bądź współpracy),

widoczność nazwa (parametr1, parametr2, …) : typ {ograniczenia}, przy czym parametry można precyzować w sposób następujący:

Wizja – dokument służący przedstawieniu przeznaczenia tworzonego produktu, jego głównych cech i ograniczeń. Powinien zawierać następujące informacje:

- nazwę projektu opisującego dany produkt,
- autora, datę oraz numer wersji dokumentu,
- wprowadzenie,
- cel projektu (tzw. potrzeby biznesowe), czyli jaki problem ma rozwiązać produkt opisywany niniejszym projektem,
- opis docelowych odbiorców (użytkowników) produktu,
- rozwiązania alternatywne (jeśli istnieją podobne produkty) lub wzorce,
- zasadniczy słowny opis produktu,
- zakres i ograniczenia produktu.

Język dokumentu wizji powinien być zrozumiały zarówno dla klienta biznesowego, jak i dla zespołu projektowego, gdyż wizja stanowi pomost pomiędzy tym co klient wymaga, a ich techniczną reprezentacją np. przypadkami użycia.

Wymaganie funkcjonalne – pojedyncza potrzeba (usługa, funkcja), którą produkt ma zaspokoić (realizować); zazwyczaj odpowiada na pytanie co? Zbiór wymagań funkcjonalnych tworzy specyfikację wymagań.

Wymaganie niefunkcjonalne – pojedyncza właściwość produktu dotycząca takich aspektów, jak użyteczność, niezawodność, wydajność, zdolność do utrzymywania a także bezpieczeństwo (por. FURPS); zazwyczaj odpowiada na pytanie jak? Zbiór wymagań niefunkcjonalnych tworzy ograniczenia produktu.

Zależność – relacja (skierowana) określająca fakt, że dany element (tzw. klient) jest zależny od innego elementu (tzw. dostawcy). Kierunek zależności określa, że zmiany we wskazywanym elemencie mogą mieć wpływ na drugi element. Zawieranie i rozszerzenie jest typem relacji zależności.

Zależność – rodzaj powiązania ukazujący wpływ na siebie powiązanych klas (np. «call», «create», «instantiate», «use»),

Zawieranie – relacja zależności wskazująca, że część przypadku użycia realizowana jest zawsze w innym przypadku użycia. Zawarty przypadek użycie nie występuje samodzielnie, lecz tylko w związku z realizacją przypadku użycia, który go zawiera.

Termin “Service Oriented Architecture” jest określeniem architektury systemów rozproszonych, w których wyróżnia się:
usługobiorcę – klienta korzystającego z usług
dostawcę usług – usługą jest realizacja pewnego przetwarzania z wykorzystaniem dostarczonych danych
rejestr usług – miejsce, gdzie klient uzyskuje informacje o potrzebnych mu usługach

Podstawą SOA jest wykorzystanie przesyłania komunikatów do wymiany informacji między uczestnikami przetwarzania. Architektury usługowe odróżnia się od architektur obiektowych i architektur komponentowych.

Dostarczając usługę w ramach architektury usługowej należy w ogólnie znanym i dostępnym rejestrze dokonać zgłoszenia:
miejsca dostępności usługi
sposobu korzystania z usługi

Klient chcący korzystać z określonej usługi:
wyszukuje w ogólnie znanych i dostępnych rejestrach opisy świadczonych usług
wybiera najbardziej mu odpowiadającą
kontaktuje się z serwerem usługi w celu realizacji przetwarzania

Sposób korzystania z usługi jest równoważny interfejsowi oprogramowania realizującego usługę. Interfejs taki powinien być napisany w sposób niezależny od:
języka programowania
systemu operacyjnego
sprzętu realizującego obliczenia

Sposób powiązania usługodawcy z usługobiorcą za pomocą tak określonych interfejsów określa się jako luźny (loosely coupled). Usługi zgłoszone w systemach SOA mogą być jednocześnie klientami innych usług. W ten sposób można tworzyć złożone schematy przetwarzania (workflows) składające się z wielu, odpowiednio zorganizowanych usług. Architektura usługowa (SOA) jest w tym ujęciu przeciwstawiona architekturze komponentowej (CBD).
ten sam cel - ponowne wykorzystanie oprogramowania
SOA - realizacja za pomocą organizacji usług sieciowych
CBD - lokalne komponowanie dostarczonych fragmentów oprogramowania
WebServices

SOA i CBSE są konkurencyjnymi technikami dostarczania gotowych „cegieł” do budowania programów. Komponenty są implementacjami interfejsów dla konkretnego środowiska (nie ma wspólnego języka definiowania interfejsów), elementy SOA realizują opublikowaną funkcjonalność (języki opisu z założenia mają być powszechnymi standardami). Pakowanie i montaż komponentów są czynnościami fazy tworzenia oprogramowania, kombinacja wykonawców usług odbywa się dynamicznie, w trakcie realizacji programu. Realizacja usług korzysta ze standardowych protokołów sieciowych, komponenty są zazwyczaj silniej zintegrowane.

Wzorce Kreacyjne:
Abstract factory
Builder
Factory method
Lazy initalization
Multiton
Object pool
Prototype
Resource acquisition is initalizing
Singleton

Wzorce Strukturalne
Adapter / Wrapper / Translator
Bridge
Composite
Decorator
Facade
Front Controller
Flywegiht
Proxy
Module

Wzorce Behawioralne
Blackboard
Chain of responosobility
Command
Interpreter
Iterator
Mediator
Memento
Null object
Observer / Publish / Subscribe
Servant
Specyfication
State
Strategy
Template method
Visitor