Analiza obiektowa wstęp do UML

Dr hab. inż. Ilona Bluemke

Plan wykładu

- Podstawy analizy obiektowej
- Historia powstania UML
- Perspektywy widzenia systemu przy projektowaniu w UML

Analiza systemu

Koncentruje się na zrozumieniu systemu oraz znalezieniu odpowiednich rozwiązań.

- V. Weinberg: "Analiza systemu jest badaniem problemów, celów, wymagań, priorytetów i ograniczeń wynikających ze środowiska wraz z szacowaniem kosztów, zysków i wymagań czasowych, w celu wypracowania pierwszych propozycji rozwiązań”

Analitycy biorą udział w:

- analizie wymagań,
- weryfikacji rozwiązań,
- ocenie technicznej projektu wstępnego i szczegółowego,
- walidacji i weryfikacji oprogramowania.
- Określają także wytyczne co do zasad pielęgnowania systemu.

Analiza obejmuje sfery:

- problemu (co ma być wykonane),
- wykonalności (czy realizacja jest możliwa),
- ekonomiczną (jakie będą koszty i zyski realizacji).

Zasady, sposoby przy podejściu do złożonego problemu:

- **Abstrakcja -** ignorowanie nieistotnych aspektów, skoncentrowanie się na istotnych.
- **Abstrakcja proceduralna** - dowolna operacje dająca określony efekt może być traktowana elementarnie, w rzeczywistości może być realizowana przez operacje niższych poziomów.
- **Abstrakcja danych** - definiowanie typu danych w sensie operacji dotyczących obiektów tego typu, z ograniczeniem takim, że wartości tych obiektów mogą być modyfikowane i odczytywane tylko za pomocą tych operacji.

Zasady, sposoby przy podejściu do złożonego problemu -2:

- **Ukrywanie informacji** - każdy składnik programu powinien ukrywać pojedyncze decyzje projektowe. Interfejs modułu jest tak projektowany aby jak najmniej odsłaniać sposób jego wewnętrznej pracy.
- **Dziedziczenie** - mechanizm wyrażania podobieństwa między klasami, ułatwiający definiowanie klas podobnych do już zdefiniowanych. Opisuje podział na cechy ogólne i szczegółowe, wyrażając atrybuty i usługi w hierarchii klas.
- **Skojarzenia** - łączenie idei.

Cechy modelu obiektowego

- abstrakcja,
- enkapsulacja,
- modularność,
- hierarchia.

Obiektowo zorientowane projektowanie

- Maksymalizuje ukrywanie informacji, może prowadzić do systemów bardzo spójnych, o mniejszym stopniu zależności elementów niż podejście funkcjonalne.
- System widziany jako **zbiór współdziałających obiektów.**

Zalety metody obiektowej

- wyeliminowane wspólne obszary danych (komunikacja poprzez przekazywanie komunikatów),
- obiekty są łatwo modyfikowalne, reprezentacja informacji jest wewnątrz nich, zmiany są lokalne i nie wpływają na inne obiekty,
- decyzje o implementacji sekwencyjnej lub równoległej nie muszą zapadać we wczesnej fazie projektowania.

Obiekt

- ma **indywidualność**
- ma **stan** - zawiera wszystkie statyczne cechy obiektu i dynamicznie się zmieniające wartości tych cech, skumulowane zachowanie obiektu,
- ma **zachowanie** (zmiany stanów, przekazywanie komunikatów).

Rola jaką obiekt może pełnić

- **Aktor** - aktywny, pracuje, steruje innymi, sam nigdy nie jest sterowany
- **Serwer** - sam innymi nie steruje, dostarcza usługi
- **Agent** - obie role, pracuje na innych, inne na nim

Analiza zorientowana obiektowo

Analiza obiektowa przebiega zazwyczaj w następujących krokach:

- Znajdowanie - identyfikacja obiektów,
- Organizowanie obiektów,
- Opis interakcji,
- Definicja operacji obiektu,
- Definicja wnętrza obiektu.

Identyfikacja obiektów

Ważny **podmiot (rzeczownik)** z dziedziny problemu jest kandydatem na obiekt.

Typy obiektów:

- aktywne/pasywne
- fizyczne/konceptualne
- chwilowe/stałe
- prywatne/publiczne
- część/całość
- ogólne/specyficzne

Organizowanie obiektów

Kryteria organizowania obiektów:

- Jakie są cechy wspólne klas/obiektów - tworzona hierarchia dziedziczenia,
- Które obiekty współpracują ze sobą,
- Które obiekty są częścią innych obiektów,
- Jakie obiekty są zależne od siebie.

Organizowanie obiektów

Opisuje się różne **scenariusze** - przykłady użycia systemu (use case).

Z nich wynika które obiekty, i w jaki sposób mają się komunikować, czego obiekty oczekują po sobie.

Na tej podstawie można określić, interfejsy obiektów oraz które obiekty są częścią innych.

Definicja operacji i wnętrza obiektu

Definicja operacji obiektu

- Określenie **Co** obiekt ma wykonywać.
- Jeśli operacje są złożone to można identyfikować nowe obiekty.

Definicja wnętrza obiektu - implementacja

METODY ZORIENTOWANE OBIEKTOWO

- OOD Booch 1991
- Object Oriented System Analysis (OOSA)
- Shaer Mellor 1988
- OMT Object Modelling Technique (Rumbaugh et al 1991)
- OOA/OOD Coad & Yourdon 1991
- OOSE (I. Jacobson)
- HOOD 1989 (Hierarchical O-O Design)
- Responsibility Driven Design (Wirfs-Brock et al 1990)
- OORASS (O-O Role Analysis, Synthesis & Structuring)
- Reenskang et al 1990
- OOSD (O-O Structured Design)
- Wasserman et al 1989, 1990
- OSA O-O System Analysis (Embley et al 1992)
- OBA Object Behavior Analysis (Gibson 1990)
- Synthesis Page-Jones & Weiss 1989
- Fussion (HP)
- Merise

Przyczyny standaryzacji

UML - ang. Unified Modelling Language

Unifikacja metod modelowania została zapoczątkowana przez Booch'a i Rumbaugh.

- 1989 – 10 obiektowo zorientowanych metod projektowania i analizy
- 1994 – 50 metod (najbardziej znane to Booch, Jacobson's OOSE – Object Oriented Software Engineering, Rumbaugh's OMT – Object Modelling Technique, Fussion, Shlaer-Mellor, Coad-Yourdon,

Historia standaryzacji

- X 1994 Rumbaugh, Booch w firmie Rational
- X 1995 Unified Method version 0.8
- VI 1996 Unified Method version 0.9 (Jacobson dołączył do Rational)
- I 1997 UML 1.0 przedłożony do standaryzacji do OMG (Object Management Group)
- IX 1997 UML 1.1 zaakceptowany przez OMG
- VI 1998 UML 1.2
- 1998 UML 1.3
- 2003 UML 1.5
- 2006 UML 2.0
- 2007 UML 2.1

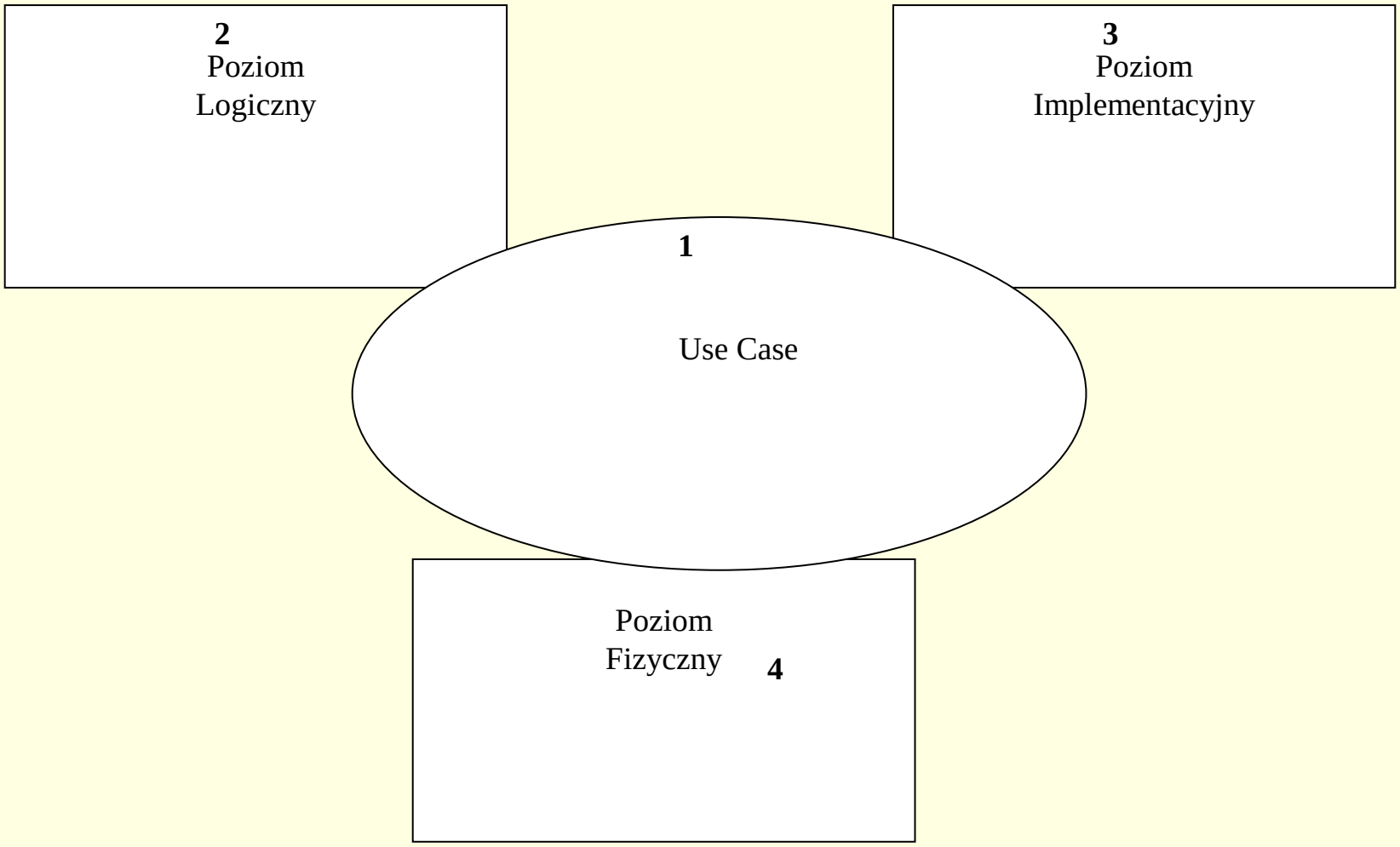
Unifikacja metod projektowania

Unifikacja metod modelowania została zapoczątkowana przez Jacobsona, Boocha i Rumbaugh.

Podstawowe składniki UML:

- pojęcie podsystemu, kategorie podsystemów - Booch
- przykłady użycia - Use Case - Jacobson
- pojęcie asocjacji - Rumbaugh
- pojedyncze klasy, obiekty – kompozycje - Embley
- opisy operacji, numerowanie komunikatów - Fusion
- diagramy stanów - Harel
- warunki przed i po operacji - Meyer
- dynamiczna klasyfikacja, nacisk na znaczenie zdarzeń - Odell
- cykle życia obiektów – Shlaer/Mellor

PERSPEKTYWY



Model Use Case

- Przedstawia system z punktu widzenia użytkownika (różnych klas użytkowników systemu).
- Modeluje zachowanie systemu w odpowiedzi na polecenia użytkownika.

Na tym etapie tworzone są diagramy „Use Case”
Posłużą one do następnych etapów projektowania oraz do końcowego testowania systemu pod kątem spełniania wymogów użytkowników.

Określa **CO** system robi

Model logiczny

Przedstawia system w postaci klas, powiązań i interakcji między nimi, zachowań obiektów należących do tych klas oraz sekwencji działań systemu.

Na tym etapie tworzy się następujące diagramy:

- klas, obiektów
- sekwencji (interakcji)
- współpracy
- przejść stanów

Określa **CO jest** w systemie, **JAK** system działa

Model implementacyjny i wdrożeniowy

Model implementacyjny

- Przedstawia system jako moduły, podsystemy, zadania. Na tym etapie powstaje diagram komponentów.

Model wdrożeniowy **(Deployment)**

- Modeluje fizyczne rozmieszczenie modułów systemu na komputerach. Uwzględnia wymagania sprzętowe, obszary krytyczne.
- Na tym etapie tworzymy diagramy rozmieszczenia (deployment).