Modelowanie Systemu Informatycznego

prezentacja 1

Wprowadzenie

wersja 1.2

dr inż. Paweł Głuchowski

Wydział Informatyki i Telekomunikacji, Politechnika Wrocławska

Treść prezentacji

1. Co to jest inżynieria oprogramowania
2. Plan wykładów
3. Spis prezentacji
4. Pomocne książki i strony internetowe

1

Co to jest inżynieria oprogramowania

Inżynieria oprogramowania

- **Wiedza i praktyka techniczna:**
  - dotycząca wszystkich faz cyklu życia oprogramowania;
  - aby uzyskać produkt (np. SI) wysokiej jakości (tańszy, mniej zawodny, efektywniej działający);
  - systemowe podejście z ilościową oceną.

Składniki inżynierii oprogramowania

- **Zarządzanie projektem informatycznym** – organizacja i kierowanie realizacją projektu (np. mającego wykonać i wdrożyć oprogramowanie).
- **Określenie środowiska programistycznego** – wybór procedur organizacyjnych, sprzętu i oprogramowania dla realizacji projektu informatycznego.
- **Analiza i model oprogramowania:**
  - model odbiorcy – opis struktury i dynamiki jego organizacji / przedsiębiorstwa,
  - analiza wymagań (w tym analiza przypadków użycia),
  - model struktury i zachowania (w tym model oprogramowania systemu i jego wdrożenia).

Składniki inżynierii oprogramowania

- **Implementacja** – stworzenie oprogramowania (w tym kodu).
- **Testowanie** – testy jednostkowe, funkcjonalne, integracyjne, akceptacyjne i inne:
  - na etapie implementacji i potem,
  - przygotowanie danych testowych,
  - wybór procedur i metryk poprawności.
- **Wdrożenie** – konfiguracja wykonanego oprogramowania i jego uruchomienie u odbiorcy.
- **Konserwacja** – zarządzanie zmianami, dbanie o spójność elementów, poprawa bezpieczeństwa.

Warstwy inżynierii oprogramowania

- **Dbanie o jakość** – kompleksowe zarządzania jakością.
  - ∈ **Proces wytwórczy** – spojenie elementów kolejnych warstw, aby racjonalnie i terminowo wytworzyć oprogramowanie.
    - ∈ **Metody** – modelowanie, projektowanie, programowanie, testowanie i pielęgnacja.
      - ∈ **Narzędzia** – środowisko CASE (Computer-Aided Systems Engineering): rozwiązania sprzętowe, programy komputerowe, bazy danych…

Inżynieria Oprogramowania – ogólne spojrzenie

na podst. Inżynieria Oprogramowania, Z. Kruczkiewicz, PWr

- Jaki problem rozwiązać ?
- Jakie cechy produktu umożliwiają rozwiązanie problemu ?
- Jak produkt ma wyglądać (rozwiązanie problemu) ?
- Jak skonstruować taki produkt ?
- Jak wykrywać błędy w projekcie lub podczas jego konstrukcji ?
- Jak obsługiwać i pielęgnować gotowy produkt ?
  - jak uwzględniać uwagi i żądania jego użytkowników:
    - poprawianie, adaptowanie, rozszerzanie;
    - zapobieganie (aby łatwo poprawiać, adaptować i rozszerzać).

2

Plan wykładów

Spodziewane efekty

- **Z zakresu wiedzy:**
  - student zna metody specyfikacji wymagań oprogramowania za pomocą diagramów przypadków użycia i diagramów czynności języka UML;
  - student zna zasady wyrażania struktury oprogramowania za pomocą diagramów klas i pakietów tego języka;
    zna kontekst użycia projektowych wzorców strukturalnych i wytwórczych;
  - student zna zasady opisywania dynamiki oprogramowania za pomocą diagramów sekwencji, czynności i maszyn stanowych języka UML;
    zna kontekst użycia projektowych wzorców zachowania;
  - opanowanie podstaw wiedzy z zakresu kierowania projektami programistycznymi;
  - nabycie wiedzy z obszaru strukturalnych metod analizy i projektowania;
  - zdobycie wiedzy z zakresów testowania, weryfikacji i walidacji oprogramowania.

Spodziewane efekty

- **Z zakresu umiejętności:**
  - nabycie umiejętności opracowania specyfikacji wymagań za pomocą diagramów przypadków użycia i diagramów czynności języka UML;
  - nabycie umiejętności wyrażania struktury systemu za pomocą diagramów klas i pakietów tego języka;
    student potrafi zastosować projektowe wzorce wytwórcze i strukturalne zgodnie z ich kontekstem użycia;
  - zdobycie umiejętności opisywania dynamiki systemów za pomocą diagramów sekwencji, czynności i maszyn stanowych języka UML;
    student potrafi zastosować projektowe wzorce zachowania zgodnie z ich kontekstem użycia;
  - opanowanie umiejętności przygotowywania testów akceptacyjnych oraz funkcjonalnych za pomocą narzędzi FitNesse oraz Selenium;
  - nabycie umiejętności przygotowywania testów jednostkowych za pomocą narzędzia JUnit.

Spodziewane efekty

- **Z zakresu kompetencji społecznych:**
  - umiejętność pracy w dwuosobowym zespole przygotowującym specyfikacje wymagań, modele struktury i dynamiki oprogramowania oraz testów akceptacyjnych, funkcjonalnych i jednostkowych.

Plan wykładów

- **Wykład 1:**
  - Modelowanie w projekcie programistycznym.
- **Wykład 2:**
  - Modelowanie wymagań. Diagramy: wymagań i przypadków użycia.
- **Wykład 3:**
  - Modelowanie zachowania. Diagramy: czynności i stanów.
- **Wykład 4:**
  - Modelowanie struktury. Diagramy: pakietów, klas, obiektów, struktur złożonych, komponentów i wdrożenia.
- **Wykład 5:**
  - Wzorce projektowe.
  - Wielowarstwowy system informatyczny.
- **Wykład 6:**
  - Modelowanie zachowania. Diagramy: komunikacji, sekwencji, przeglądu interakcji i czasowe.

Plan wykładów (druga część kursu)

- **Wykłady 7–8:**
  - Testowanie oprogramowania. Rodzaje testów, techniki projektowania testów, programowania przez testy. Testy: jednostkowe, akceptacyjne i funkcjonalne.
- **Wykład 9:**
  - Zarządzanie projektem.
- **Wykład 10:**
  - Modele cyklu życia systemu.
- **Wykłady 11–12:**
  - Analiza strukturalna. Diagramy: ERD, DFD, stanów.
- **Wykłady 13–14:**
  - Zapewnienie jakości w projekcie.
  - Metody weryfikacji i walidacji.
- **Wykład 15:**
  - Bezpieczeństwo i konserwacja oprogramowania.

3

Spis prezentacji

Spis prezentacji

1. Wprowadzenie
2. Modelowanie w projekcie programistycznym i język UML
3. Modelowanie wymagań – diagram wymagań i diagram przypadków użycia
4. Modelowanie zachowania – diagram czynności
5. Modelowanie zachowania – diagram stanów
6. Modelowanie struktury – diagram klas
7. Modelowanie struktury – diagram pakietów, diagram obiektów i diagram struktur złożonych
8. Modelowanie struktury – diagram komponentów i diagram wdrożenia
9. Wzorce projektowe
10. Modelowanie zachowania – diagram komunikacji, diagram sekwencji, diagram przeglądu interakcji i diagram czasowy

4

Pomocne książki i strony internetowe

Pomocne książki i strony internetowe

Inżynieria Oprogramowania

- **Specyfikacja wymagań oprogramowania. Kluczowe praktyki analizy biznesowej**
  – K. Wiegers, C. Hokanson; wyd. Helion 2024
- **Inżynieria oprogramowania w praktyce. Od wymagań do kodu z językiem UML**
  – M. Śmiałek, K. Rybiński; wyd. Helion 2023
- **Inżynieria oprogramowania**
  – A. Jaszkiewicz; wyd. Helion 2011
- **Inżynieria oprogramowania**
  – K. Sacha; wyd. PWN 2010
- **Inżynieria oprogramowania**
  – J. Somerville; wyd. WNT 2003
- **Inżynieria oprogramowania w projekcie informatycznym**
  – J. Górski; wyd. Mikom 1999

Pomocne książki i strony internetowe

Inżynieria Oprogramowania

- **Inżynieria Oprogramowania,**
  Modelowanie i analiza systemów informatycznych,
  Projektowanie oprogramowania
  – prezentacje wykładowe; Z. Kruczkiewicz; PWr
  – zofia.kruczkiewicz.staff.iiar.pwr.edu.pl/index.php?id=dydaktyka
- **Inżynieria oprogramowania,**
  Kierowanie projektem programistycznym,
  – prezentacje wykładowe; O. Unold; PWr
  – olgierd.unold.staff.iiar.pwr.wroc.pl/dydaktyka
- **PSK - projektowanie systemów komputerowych**
  – B. Frączak; 2009
  – brasil.cel.agh.edu.pl/~09sbfraczek
- **Learning Guides**
  – Visual Paradigm
  – www.visual-paradigm.com/guide

Pomocne książki i strony internetowe

Wzorce projektowe

- **Wzorce projektowe. Rusz głową!**
  – E. Freeman, E. Robson; wyd. Helion 2023
- **Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku**
  – E. Gamma, R. Helm, R. Johnson, J. Vlissides; wyd. Helion 2021
- **Tablice Informatyczne. Wzorce projektowe**
  – D. Krasnokucki; wyd. Helion 2017
- **Projektowanie zorientowane obiektowo. Wzorce projektowe**
  – A. Shalloway, J.R. Trott; wyd. Helion 2005
- **J2EE. Wzorce projektowe**
  – D. Alur, J. Crupi, D. Malks; wyd. Helion 2004
- **Wzorce projektowe**
  – Refactoring.Guru
  – refactoring.guru/pl/design-patterns

Pomocne książki i strony internetowe

UML i SysML

- **Język inżynierii systemów SysML. Architektura i zastosowania. Profile UML 2.x w praktyce**
  – S. Wrycza, B. Marcinkowski; wyd. Helion 2013
- **UML 2.0. Almanach**
  – D. Pilone, N. Pitman; wyd. Helion 2007
- **Język UML 2.0 w modelowaniu systemów informatycznych**
  – S. Wrycza, B. Marcinkowski, K. Wyrzykowski; wyd. Helion 2006
- **UML w kropelce, wersja 2.0**
  – M. Fowler, wyd. LTP 2005
- **The Unified Modeling Language User Guide**
  – G. Booch, J. Rumbaugh, I. Jacobson; wyd. Addison-Wesley Professional 2005
- **UML Przewodnik użytkownika**
  – G. Booch, J. Rumbaugh, I. Jacobson; wyd. WNT 2012

Pomocne książki i strony internetowe

UML i SysML

- **Diagramy UML**
  – M. Wolski
  – wolski.pro/diagramy-uml
- **Notationsübersicht UML 2.5**
  – oose Innovative Informatik;
  – www.oose.de/wp-content/uploads/2012/05/UML-Notations%c3%bcbersicht-2.5.pdf
- **UML**
  – Visual Paradigm
  – guides.visual-paradigm.com/category/uml
- **UML Modeling**
  – Visual Paradigm
  – www.visual-paradigm.com/VPGallery/diagrams
- **UML – Some metamodels**
  – kompilatory.iiar.pwr.edu.pl/wiki/index.php/UML/Some_metamodels
- **SysML Diagram Tutorial**
  – SysML.org
  – sysml.org/tutorials/sysml-diagram-tutorial

Pomocne książki i strony internetowe

UML i SysML

- **The OMG Specification Catalog**
  – www.omg.org/spec
- **MDA Specifications**
  – www.omg.org/mda/specs.htm
- **Unified Modeling Language (UML)**
  – www.omg.org/spec/UML

Pomocne książki i strony internetowe

Visual Paradigm

- **Visual Paradigm Quick Start**
  – wyd. Visual Paradigm 2017
- **Visual Paradigm Tutorials**
  – Visual Paradigm
  – www.visual-paradigm.com/tutorials
- **Examples**
  – Visual Paradigm Community Circle
  – circle.visual-paradigm.com/diagram-examples

Temat następnej prezentacji

Modelowanie w projekcie programistycznym i język UML