dr inż. Ilona Bluemke

Studia zaoczne inżynierskie
INŻYNIERIA OPROGRAMOWANIA

6 semestr 32 godziny wykładu

Przedmioty poprzedzające: Programowanie obiektowe
Przedmioty – następniki: Zarządzanie projektami

Słowa kluczowe:
obiektowe metody projektowania, UML, strukturalne metody projektowania, niezawodność, weryfikacja, walidacja, analiza wymagań, jakość oprogramowania, koszty oprogramowania, niezawodność oprogramowania

Cel przedmiotu

Celem przedmiotu jest zapoznanie studentów z zasadami i nowoczesnymi metodami projektowania i produkcji wysokiej jakości oprogramowania. Omawiane są problemy związane z procesem produkcji oprogramowania, specyfikacją wymagań, analizą, projektowaniem, implementacją, weryfikacją i walidacją, szacowaniem kosztów oraz zapewnieniem wysokiej jakości oprogramowania. Przedstawiony jest także UML – standard modelowania obiektowego.

Treść wykładu

Wprowadzenie. Cele, problemy i zakres inżynierii oprogramowania. Kryteria jakości oprogramowania.
Obiektowa analiza i projektowanie. Wprowadzenie – cechy i zalety modelu obiektowego, analiza obiektowa (identyfikacja obiektów, organizowanie obiektów, opis interakcji, definicja operacji).
UML – standard modelowania obiektowego: Poziomy modelu i typy diagramów. Przykłady użycia (use case)- elementy diagramów, organizowanie.
Model logiczny - diagramy klas (klasy – atrybuty i operacje, relacje agregacji, generalizacji, asocjacji, role, krotności, atrybuty powiązań, kwalifikacje, powiązania ternarne), diagramy sekwencji (typy komunikatów, aktywacja obiektów), diagramy zmian stanów (zdarzenia, warunki, akcje, strukturalizacja – generalizacja i agregacja, mechanizm historii), diagramy aktywności (decyzje, aktywności współbieżne i synchronizacje, wysyłanie i odbiór komunikatów), „tory wodne" (ang. swimlanes), diagramy współpracy (numerowanie komunikatów, warunkowe, sekwencyjne i równoległe wysyłanie komunikatów, synchronizacja komunikatów, obiekty aktywne.
Model implementacyjny – diagram komponentów (komponenty, podsystemy) i model fizyczny (diagram montażowy).
Metoda produkcji oprogramowania z zastosowaniem UML.
Modele cyklu życia oprogramowania (wodospadowy, ewolucyjny, spiralny), przyrostowa realizacja oprogramowania, montaż z gotowych elementów, prototypowanie. Lekkie metodyki programowania (XP programming). Czynniki nietechniczne w inżynierii oprogramowania (osobowości w grupie, organizacja zespołów).
Studium wykonalności. Decyzje podejmowane w studium wykonalności, czynniki sukcesu, rezultaty. Sieci zależności aktywności, harmonogramowanie, przydział zadań członkom zespołu.

Analiza i specyfikacja wymagań. Definicja i specyfikacja wymagań. Przykładowa struktura dokumentu definicji/specyfikacji wymagań. Wymagania funkcjonalne i niefunkcjonalne. Specyfikacje nieformalne i formalne:
Projektowanie oprogramowania. Jakość projektu, cechy „dobrego projektu". Walidacja i weryfikacja oprogramowania. Podejście obiektowe i funkcjonalne do projektowania.
Strukturalna analiza i projektowanie. Przydatność metod strukturalnych, narzędzia modelowania strukturalnego.
Diagramy przepływu danych (DFD) – procesy, magazyny danych, przepływy, terminatory. Specyfikacja procesów na najniższym poziomie (język naturalny, warunki początkowe i końcowe, tablice decyzyjne, diagramy przepływu sterowania, diagramy Nassi-Shneidermana). Diagramy związków encji (ERD) – typy związków, liczności, wskaźniki asocjowanych typów, nadtypy, podtypy, odmiany diagramów. Diagramy sieci przejść (STD) i diagramy struktury (structure charts).
Testowanie i inspekcja. Strategie testowania (zstępujące, wstępujące, wątków, stresowe, porównawcze). Testowanie ukierunkowane na wyszukiwanie defektów – funkcjonalne, strukturalne (złożoność cyklomatyczna McCabe, ścieżki niezależne), interfejsów (typy błędów). Testowanie obiektowe – aksjomaty, wpływ cech języków obiektowych, metody specjalizowane. Inspekcja kodu.
Estymacja kosztów. Metoda punktów funkcyjnych, model COCOMO, krzywa Rayleight'a.
Metryki i miary oprogramowania. Metryki oprogramowania obiektowego. Miary niezawodności oprogramowania i techniki programowania dla systemów o dużej niezawodności.

Literatura

[1] I. Bluemke „Inżynieria Oprogramowania", seria Pomocnicze materiały dydaktyczne WSISIZ 2003
[2] G. Booch, J. Rumbaugh, I. Jacobson:UML Przewodnik Użytkownika; WNT 2001
[3] A. Jaszkiewicz: Inżynieria oprogramowania, Helion 1997
[4] J. Górski i inni: Inżynieria oprogramowania w projekcie informatycznym; Mikom
[5] M.Fohler, K.Scot: UML w kropelce; LTP 2002
[6] E. Yourdon: Współczesna analiza strukturalna; WNT 1996
[7] I. Sommerville: Inżynieria oprogramowania, WNT 2003
[8] S. Szejko redakcja: Metody wytwarzania oprogramowania, Mikom 2002
[9] K. Subieta: Wprowadzenie do inżynierii oprogramowania, Wydawnictwo PJWSTK, 2002
[10] R.V. Binder: Testowanie systemów obiektowych. Modele , wzorce i narzędzia, WNT 2003
[11] S. Wrycza, B. Marcinkowski, K.Wyrzykowski : Język UML 2.0 w modelowaniu systemów informatycznych, Helion 2005

Uwagi:

Program przedmiotu w dużej mierze pokrywa zakres kursu Software Engineering ACM Computing Curricula. Układ wykładów wynika z konieczności równoległego prowadzenia laboratorium i korzystania z UML.