Wprowadzenie

Głównym celem zajęć jest sprawdzenie i poprawa praktycznych umiejętności w zakresie wybranych istotnych aspektów realizacji projektów informatycznych, t.j.:

- Zarządzania wymaganiami
- Planowania prac
- Wyboru architektury, technologii, projektu struktury aplikacji lub systemu
- Oceny i zarządzania ryzykiem w projekcie
- Wykorzystania narzędzi informatycznych

W celu bardziej realistycznego odwzorowania rzeczywistego procesu produkcji oprogramowania, wskazane jest aby zespoły realizujące każdy z projektów składały się z 3-4 osób. W ramach każdego zespołu osoby powinny się podzielić rolami, przejmując odpowiedzialność za poszczególne aspekty procesu. Chociaż wybór ról i zakresu odpowiedzialności może być (do pewnego stopnia) dowolnie kształtowany w ramach zespołu, jako przykład można wskazać następujące role:

a) Architekta
   Czyli osoby która projektuje konstrukcję programu, proponuje/wybiera rozwiązania, technologie itp. Odpowiada również za podział na modułu i ew. definicję interfejsów komunikacyjnych
b) Testera
   Osoby odpowiedzialnej za przygotowanie testów automatycznych, testowanie i raportowanie błędów oraz kontrolę jakości
c) Analityka / zarządzającego wymaganiami
   Osoba która odpowiada za zbieranie wymagań klienta oraz przełożenie ich na wymagania i konstrukcje programistyczne
d) Kontrolera
   Odpowiedzialnego za pilnowanie harmonogramu, zebranie wszystkich elementów projektu w całość itp.
e) Programisty
   Ta rola nie wymaga zapewne komentarza =)

Jedna osoba może (i w wielu przypadkach będzie musiała) pełnić więcej niż jedną rolę. Wskazane jest aby zespół podzielił się rolami już na początku prac, choć w trakcie realizacji projektu może się oczywiście okazać, że pewne zmiany są konieczne ze względu np. na braki kompetencyjne. Przy ocenie projektu będzie brane pod uwagę przede wszystkim inżynierskie podejście – odpowiedni projekt, szacowania (nakładu pracy, ryzyka itp.), analiza możliwych do wykorzystania technologii, ich wad i zalet itp. Samo osiągnięcie założonej funkcjonalności nie jest wystarczające.

Oceniana będzie:

- Prawidłowość zbierania i zarządzania wymaganiami – kompletność zidentyfikowanych wymagań, zarządzanie zmianami itp.
- Projekt realizowanego systemu / aplikacji – schemat/model funkcjonalny, struktury danych, stos technologiczny (krytyczna analiza technologii)
- Jakość projektu: ilość błędów, stopień pokrycia testami automatycznymi, dokumentacja itp.
- Stopień realizacji założonej funkcjonalności

Ocena odbywać się będzie na każdym z etapów projektu (patrz następna strona)

Etapy prac i kluczowe terminy

Etap: Wybór tematu (do 27.X)

Studenci mogą zaproponować realizację własnego tematu (aplikacji / systemu itp.) pod warunkiem jego odpowiedniej złożoności. Jako przykłady mogą posłużyć:

- Bank bitcoinów
- Sieciowa gra w brydża
- Platforma komunikacyjna – np. studenckiej wymiany książek, itp.

W zależności od ilości zgłoszonych / uzgodnionych projektów przygotuję listę tematów i przydzielone zostaną dla grup które nie miały własnego pomysłu.

Etap: Specyfikacja wymagań (17.XI)

Na pierwszym etapie jest zebranie i udokumentowanie wymagań wobec projektowanego systemu. Obejmuje to zarówno wymagania funkcjonalne jak i niefunkcjonalne. W zakresie wymagań funkcjonalnych zasadniczo należy opisać wszystkie główne przypadki użycia (*use cases*).

Wymagania należy skonsultować z prowadzącym. Interpretacja wymagań, w zakresie który nie został sprecyzowany, pozostaje w gestii prowadzącego zajęcia. Nie są dopuszczalne generalne określenia negatywne typu „żadna inna poza określoną powyżej funkcjonalnością nie będzie zrealizowana”. Zadaniem zespołu jest zidentyfikowanie potencjalnie wszystkich obszarów niejasności.

Ocena z tego etapu to 10% oceny z przedmiotu

Etap: Projekt systemu (1.XII)

Projekt systemu powinien zawierać zarówno elementy związane z przygotowywanym systemem / aplikację jak i metodyką prowadzenia projektu, to jest:

- **Architekturę systemu**
  Przez architekturę rozumiana jest logiczna konstrukcja systemu – główne moduły systemu (w tym również elementy „zewnętrzne” - np. baza danych, frameworki czy główne biblioteki) oraz ich sposoby połączenia. Architektura typowego systemu będzie składać się z kilku-kilkunastu modułów. Wskazane jest przedstawienie ich powiązań w formie graficznej / diagramu.
- **Opis interfejsów**
  Na potrzeby zadania należy przyjąć że interfejsem jest każdy istotny punkt podziału logicznego aplikacji, w szczególności gdy służy on do komunikacji pomiędzy różnymi technologiami lub systemami. Interfejsem będzie więc np. połączenie z bazą danych, GUI, REST (i inne interfejsy sieciowe) a czasami również miejsca komunikacji wewnątrz aplikacji jeżeli ze względów projektowych istotna jest kontrola przepływu danych w danym punkcie systemu.
- **Listę wykorzystywanych technologii** (stos technologiczny)
  Należy uzasadnić wybór technologii
- **Projekt testów**

Projekt testów obejmuje w szczególności planowane do wykorzystania narzędzia (jUnit? Selenium? itp.), oraz zakres projektowanych testów – np. co będzie testowane testami jednostkowymi, co funkcjonalnymi, jak będą realizowane testy wydajnościowe (o ile będą) i inne.

- **Analiza ryzyka**
  Należy przygotować listę ryzyk, przedstawić macierz ryzyka oraz określić sposoby reagowania na przedstawione ryzyka
- **Listę narzędzi planowanych do użyci przy realizacji projektu**
  (issue tracking, repozytorium itp.)

W zależności od zakresu projektu/potrzeb/technologii, wskazane może być również opracowanie struktury bazy danych (diagramu), czy też opisu UML.

Ocena z tego etapu to 25% oceny z przedmiotu

Etap: Prezentacja prototypu (15.XII)

Na tym etapie zespół ma za zadanie przedstawić funkcjonalny prototyp systemu/aplikacji, obejmujący ok. 50% całej funkcjonalności. Należy również przygotować szkielet testów (pokrycie testami na poziomie ponad 30%) oraz szkielet dokumentacji. W czasie prezentacji grupa powinna przedstawić działającą aplikację (w zakresie zaimplementowanej funkcjonalności). Na tym etapie możliwa będzie też zmiana / modyfikacja wymagań.

Zarówno ten, jak i następny etap należy przedstawić w formie prezentacji (na rzutniku), obejmującej zarówno informacje o realizacji projektu, jak i pokaz działającego systemu/aplikacji.

Ocena z tej prezentacji to 30% oceny z przedmiotu

Etap: Dostarczenie kompletnego projektu (17.I) i prezentacja (19.I)

Na ostatnim etapie zespół ma przedstawić kompletny i funkcjonalny system/aplikację. Należy też poddać aplikację statycznej analizie kodu (przy pomocy np. SonarQube, PMD, FindBugs lub podobnych). Zespół powinien zaprezentować działającą aplikację oraz wnioski z wybranego sposobu tworzenia aplikacji – zidentyfikowane problemy, kluczowe decyzje, ryzyka, wybraną metodykę pracy oraz ich wpływ na projekt itp. Dwa dni wcześniej należy dostarczyć prowadzącemu zajęcia kod źródłowy systemu/aplikacji oraz dokumentację.

Istotne elementy oceny programu / aplikacji:

1. Funkcjonalność aplikacji
2. Jakość aplikacji (ilość błędów)
3. Zgodność ze wszystkimi wymaganiami (np. licencyjnymi)
4. Kompletność, czytelność i użyteczność dokumentacji (włączając w to ew. dokumentacje w kodzie)
5. Ergonomia instalacji i użytkowania
6. Zakres dołączonych testów automatycznych

Wymagania wobec kodu źródłowego

II. **Wykaz elementów składających się na oddawany projekt:**
    1. Kod źródłowy (wyczyszczony – bez zbędnych elementów typu pliki kontroli wersji itp.)
    2. Instrukcje kompilacji, instalacji i konfiguracji
    3. Dokumentacja użytkownika (czyli opis interfejsu - co gdzie i jak można zrobić)
    4. Dokumentacje techniczna - podział aplikacji na moduły, architektura, opis API i istotnych interfejsów itp.
    5. Dokumentacja w kodzie – komentarze
    6. Testy automatyczne (jednostkowe i funkcjonalne) aplikacji
III. **Wymagania licencyjne / prawne:**
    1. Kod przygotowany przez grupę w ramach projektu dostarczony na licencji zgodnej z **BSD** (informacje w nagłówkach)
    2. Wszystkie wykorzystane biblioteki / komponenty itp. - na licencji pozwalającej na nieograniczone kopiowanie i rozpowszechnianie

Ocena z ostatniego etapu to 35% oceny z przedmiotu