Zaawansowane zagadnienia projektowania obiektowego
Wskazówki projektowe

1 Struktura opracowania pisemnego

1. **Streszczenie**, w którym określamy cel projektu. Na przykład, obiektowy model topologii sieci komputerowej może służyć do wizualizacji rozmieszczenia jej węzłów, planowania routingu, itd. Staramy się umieścić nasz system w pewnej ogólnej klasie, np. typu klient–serwer, z bazą danych, sterowany zdarzeniowo, czasu rzeczywistego lub inny. Czy nasz system jest samodzielną aplikacją, fragmentem biblioteki klas, zrębu (ang. framework), komponentu, itp.? Jakie techniki zostały zastosowane (rodzaje dziedziczenia, składania, wymienić zastosowane wzorce projektowe). Wymienić sugerowane języki implementacji, środowiska i proponowane narzędzia. Streszczenie powinno być krótkie (punkty, hasła).

2. **Wstępny opis słowny** dotyczy tego **co** system robi, a nie **jak**. Jest to wyjściowy model słowny działania systemu, może być on wynikiem wywiadu z potencjalnymi użytkownikami. Zwracamy uwagę na ważne pojęcia, które pojawiły się w opisie, staramy się je ściślej zdefiniować i usystematyzować. Pojęcia te (głównie rzeczowniki) można przedstawić w formie słownika (glosariusza).

3. **Słownik pojęć z dziedziny problemu** ułatwia systematykę pojęć z dziedziny problemu, który rozwiązujemy. Przedstawia wyłowione z opisu słownego określenia, które przełożą się następnie na składniki oprogramowania takie jak np. klasy. Zdarza się, że do reprezentacji niektórych pojęć potrzeba kilku klas. Często w systemie pojawią się także klasy nie odpowiadające bezpośrednio terminom w słowniku, a dotyczące dziedziny implementacji.

4. **Analiza wymagań użytkownika**, w której można zastosować poznane w ramach przedmiotu "inżynieria oprogramowania" przypadki użycia (ang. use case). Ważne jest zrozumienie jak system jest widziany z punktu widzenia użytkownika. Można przedstawić w tym punkcie główny scenariusz działania oraz najciekawsze scenariusze poboczne (czyli tzw. nietypowe sytuacje).

5. **Modele systemu z różnych perspektyw** to najważniejsza część projektu, w której posiłkujemy się językiem UML do przedstawienia **struktury** i **zachowania** naszego systemu. Podstawowym diagramem do opisu struktury jest diagram klas, natomiast zachowanie systemu można zobrazować diagramem sekwencji. W razie potrzeby wykonujemy inne rysunki wyjaśniające nasze idee i koncepcje. Nie boimy się przedstawiać alternatywnych rozwiązań jakiegoś problemu i analizować konsekwencje. Rysunkom muszą towarzyszyć słowne komentarze i wyjaśnienia, dopiero te dwa elementy (grafika +

tekst) ułatwiają zrozumienie intencji autora. Wszelkie decyzje projektowe powinny być uzasadnione.

Właściwe projektowanie można zacząć od naszkicowania współpracy obiektów w celu realizacji scenariuszy działania systemu opisanych na etapie analizy. W dalszej kolejności przypisujemy obiekty do poszczególnych klas. Wtedy często może się okazać, że potrzebnych jest więcej klas niż założyliśmy wstępnie. Należy zwrócić uwagę aby modele systemu z różnych perspektyw były spójne.

6. **Kwestie implementacyjne** Na tym etapie zastanawiamy się nad wyborem języków implementacji, środowisk i narzędzi oraz organizacją projektu. Warto rozważyć i krótko opisać zastosowanie pewnych narzędzi ułatwiających pracę i automatyzację niektórych czynności (patrz wykaz źródeł).

7. **Podsumowanie i dyskusja krytyczna** Tu zamieszczamy spostrzeżenia na temat wykorzystania w projekcie ciekawych technik i co dzięki nim zyskaliśmy.

8. **Wykaz materiałów źródłowych** Podajemy źródła, z których korzystaliśmy (książki, artykuły, strony internetowe)

2 Przykładowe propozycje tematów

1. System monitorowania ruchu ulicznego
2. Nadzór ruchu narciarzy w dużym ośrodku
3. Inteligentny system kontroli prędkości pojazdów
4. System intensywnego nadzoru stanu pacjenta
5. System kompleksowej ochrony obiektu
6. Akwizycja i archiwizacja danych o stanie wód rzek
7. System akwizycji danych meteorologicznych
8. System do obsługi konferencji
9. Informatyczny system obsługi przychodni
10. Planowanie diety
11. Planowanie złożonego przedsięwzięcia
12. Wspomaganie konfiguracji systemu komputerowego
13. Zarządzanie rezerwacją miejsc w multikinie
14. Wspomaganie planu zajęć w uczelni
15. Obiektowy model rozmieszczenia węzłów sieci komputerowej
16. Wybrana prosta gra planszowa

17. Zbiór klas do realizacji złożonej struktury danych

18. Projekt i oprogramowanie wybranych wzorców projektowych

Literatura

[1] S. Chacon and B. Straub. Pro Git. Everything You Need to Know about Git, 2014. URL <url:http://git-scm.com/book/en/v2>.

[2] Doxygen: a tool for generating documentation from source code. <url:https://www.doxygen.nl/>. 2021.

[3] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Wzorce projektowe: elementy oprogramowania wielokrotnego użytku*. Inżynieria Oprogramowania. Wydawnictwa Naukowo-Techniczne, Warszawa, 2005.

[4] GoogleTest User's Guide. <url:https://google.github.io/googletest/>. 2021.

[5] JUnit 5: Testing Framework for Java and the JVM. <url:https://junit.org/junit5/>. 2021.

[6] PyUnit: Python Unit Testing Framework. <url:http://pyunit.sourceforge.net/pyunit.html>. 2021.

[7] S. Si Alhir. *UML. Wprowadzenie.* Helion, Gliwice, 2004.

Jacek Cichosz