dr inż. Paweł Głuchowski (Politechnika Wrocławska, Wydział Informatyki i Telekomunikacji)

Inżynieria Oprogramowania

Laboratoria nr 2–3

wersja 1.0

Temat: Modelowanie wymagań.

Zadanie 1. (Projekt w Visual Paradigm)

Polecenie: Należy wykonać nowy projekt w programie Visual Paradigm. W nim będą umieszczane dokumenty tekstowe i diagramy modelujące system informatyczny, wykonywane podczas etapów 1—5.

Jeden członek zespołu powinien umieścić ten projekt w repozytorium Visual Paradigm i udostępnić go pozostałym członkom oraz prowadzącemu laboratoria.

Jak skorzystać z repozytorium Visual Paradigm

- Connect Visual Paradigm to VP Online for team collaboration
- Tworzenie współdzielonego projektu Visual Paradigm

Zadanie 2. (Opis „świata rzeczywistego”)

Polecenie: Należy wykonać krótki, szkicowy opis „świata rzeczywistego”, przekazywany przez klienta – zamawiającego oprogramowanie:

- opis zasobów ludzkich,
- przepisy i strategie,
- dane techniczne.

Objaśnienie: **Opis zasobów ludzkich:** Kim są i co robią przyszli użytkownicy oprogramowania.

W przypadku systemów wbudowanych ten opis dotyczy procesów technologicznych, które należy zautomatyzować.

Na podstawie tego opisu później określa się aktorów i wymagania funkcjonalne, czyli jakie funkcje powinno wykonywać tworzone oprogramowanie, aby wspomagać pracę jego użytkowników.

Przepisy i strategie: Co ogranicza działalność zamawiającego: przepisy, ustawy, zarządzenia, strategie zamawiającego itp.

Na podstawie tego opisu później określa się wymagania niefunkcjonalne, czyli ograniczenia dla wymagań funkcjonalnych i działania oprogramowania.

Dane techniczne: Jakie są dane związane z pracą projektowanego oprogramowania:

- liczba pracowników klienta, rodzaj i liczba przetwarzanych danych, częstość wykonywania operacji na danych itp.;
- umiejscowienie użytkownika (firmy) oraz charakterystyka jego klientów i pracowników;
- używany sprzęt i oprogramowanie, w tym przeznaczone na wdrożenie projektowanego oprogramowania.

Zadanie 3. (Wymagania funkcjonalne i niefunkcjonalne)

Polecenie: Należy wykonać numerowaną listę wymagań funkcjonalnych i numerowaną listę wymagań niefunkcjonalnych oprogramowania na podstawie opisu „świata rzeczywistego”. Wymagania powinny być szczegółowo zdefiniowane.

Objaśnienie: **Wymagania funkcjonalne:** co projektowane oprogramowanie powinno robić lub jakie cele osiągać, aby zautomatyzować procesy wykonywane przez pracowników klienta.

Wymagania niefunkcjonalne: jak lub w jakich warunkach system ma to osiągnąć (przy jakich ograniczeniach sprzętowych, organizacyjnych, prawnych itp.) oraz jakie zastosować rozwiązania technologiczne m.in. w zakresie: bezpieczeństwa, niezawodności, skalowalności i wydajności, aby spełnić oczekiwania użytkownika lub klienta.

Przykład definicji wymagań (bez podziału na funkcjonalne i niefunkcjonalne)

Projektowany system informatyczny (SI) automatyzuje kontrolę bezpiecznego poziomu promieniowania i reagowanie na jego przekroczenie w pewnej fabryce.

1. SI jest obsługiwany przez kontrolera bezpieczeństwa i nadzorowany przez kierownika fabryki.
2. SI działa na podstawie poleceń wydawanych przez kontrolera bezpieczeństwa lub kierownika fabryki oraz odczytów stanu urządzeń pomiarowych i urządzeń alarmowych.
3. Kontroler bezpieczeństwa sprawdza sprawność systemu alarmowego przez przeprowadzenie jego testowego włączenia, opcjonalnie razem z ćwiczeniem ewakuacji pracowników fabryki przez testowe włączenie sygnalizacji nakazu ewakuacji.
4. Test systemu alarmowego nie może być wykonywany częściej niż raz na tydzień.
5. Kierownik fabryki ma te same uprawnienia do obsługi SI, co kontroler bezpieczeństwa, a dodatkowo na koniec każdego miesiąca otrzymuje z SI raport o pracy SI.
6. SI stale monitoruje poziom promieniowania w fabryce przy pomocy odczytów z urządzeń pomiarowych.
7. SI na bieżąco informuje kontrolera bezpieczeństwa o wzroście poziomu promieniowania w fabryce, jeśli ten wzrost utrzymuje się dłużej niż 1 minutę.
8. SI włącza w urządzeniach alarmowych alarm bezpieczeństwa i sygnalizuje nakaz ewakuacji z fabryki w przypadku przekroczenia bezpiecznego poziomu promieniowania oraz informuje o tym kontrolera bezpieczeństwa.
9. Kontroler bezpieczeństwa odwołuje alarm bezpieczeństwa i nakaz ewakuacji z fabryki, jeśli uzna taką potrzebę. Wówczas SI wyłącza w urządzeniach alarmowych alarm bezpieczeństwa i sygnalizację nakazu ewakuacji z fabryki.

Zadanie 4. (System i aktorzy)

Polecenie: Należy określić granice projektowanego oprogramowania i zdefiniować jego aktorów oraz jego system lub podsystemy.

Objaśnienie: Jeśli projektowane oprogramowanie ma składać się z odrębnych, współpracujących ze sobą części, to każdą część modleuje osobny podsystem.

Osoba, oprogramowanie, urządzenie, inny system itp. może być aktorem projektowanego oprogramowania, jeśli będzie mieć bezpośredni udział w jego działaniu, czyli bez pośrednictwa innego aktora.

Aktorzy współpracujący ze sobą poza projektowanym oprogramowaniem mogą być połączeni ze sobą relacją asocjacji.

Zadanie 5. (Przypadki użycia)

Polecenie: Należy określić przypadki użycia dla projektowanego oprogramowania, w tym 2 bardziej złożone lub 3 mniej złożone przypadki użycia.

Należy wykonać analizę wspólności i zmienności przypadków użycia w celu określenia:

– relacji między nimi (uogólnienie, zawieranie, rozszerzanie);
– relacji uogólnienia między aktorami, mającymi w nich udział.

Każdy rodzaj relacji (uogólnienie, zawieranie, rozszerzanie) powinien być zastosowany.

Należy połączyć aktorów relacjami asocjacji z przypadkami użycia, w których realizacji biorą udział.

Objaśnienie: **Przypadek użycia:** proces składający się z operacji wykonywanych przez projektowane oprogramowanie (jego funkcja), zmierzający do spełnienia jednego lub więcej wymagań funkcjonalnych.

Przypadek użycia powinien być możliwie najogólniej zdefiniowany tak, aby odpowiadał funkcji projektowanego oprogramowania, która spełnia jakieś wymaganie lub wymagania funkcjonalne z uwzględnieniem ograniczeń zawartych w wymaganiach niefunkcjonalnych.

Relacja uogólnienia: łączy przypadki użycia lub aktorów w związek *wersja podstawowa (ogólna) – wersja pochodna (szczególna)*.

Gdy dwaj aktorzy mają identyczny lub prawie identyczny udział w realizacji przypadku użycia, to łączy ich uogólnienie i tylko ogólny aktor jest połączony asocjacją z tym przypadkiem użycia.

Relacja zawierania («include»): łączy przypadki użycia w związek *zależna całość – obowiązkowa część*.

Gdy część realizacji przypadku użycia może być wykonywana osobno (nie tylko jako jego część) lub gdy może być też wykonywana jako część innego przypadku użycia, to należy ją wydzielić do osobnego przypadku użycia i przyłączyć go relacją zawierania.

Relacja rozszerzania («extend»): łączy przypadki użycia w związek *niezależna całość – opcjonalna lub alternatywna część*.

Gdy część realizacji przypadku użycia może być wykonywana opcjonalnie (losowo lub zależnie od spełnienia jakiegoś warunku) lub alternatywnie z inną lub innymi częściami jego realizacji, to należy ją wydzielić do osobnego przypadku użycia i przyłączyć go relacją rozszerzania.

Złożony przypadek użycia: zawiera w sobie lub jest rozszerzany przez inne przypadki użycia.

Za bardziej złożony przypadek użycia należy przyjąć taki, który sam składa się ze złożonych przypadków użycia.

Relacja asocjacji: łączy aktora z tym przypadkiem użycia, którego realizację ten aktor inicjuje lub który inicjuje z tym aktorem przepływ danych lub sterowania. Relacja ta może być skierowana i mieć określoną liczność.

Zadanie 6. (Diagram przypadków użycia)

Polecenie: Należy wykonać diagram przypadków użycia, zawierający:

- system lub jego podsystemy;
- aktorów i relacje między nimi (uogólnienie, asocjacja);
- przypadki użycia i relacje między nimi (uogólnienie, zawieranie, rozszerzanie) oraz między nimi a aktorami (asocjacja);
- inne elementy, w tym notatki (opcjonalnie).

Objaśnienie: Diagram przypadków użycia nie pokazuje kolejności realizacji przypadków użycia.

Zadanie 7. (Opis przypadków użycia)

Polecenie: Należy wykonać opis każdego przypadku użycia, który znajduje się na wykonanym diagramie.

Należy zdefiniować udział (bezpośredni i pośredni) aktorów w realizacji przypadków użycia w postaci tabeli lub listy.

Objaśnienie: Opis przypadku użycia należy wykonać w odpowiednich zakładkach okna “Use Case Details” programu Visual Paradigm.

Elementy opisu przypadku użycia:

- **numer:** identyfikator;
- **nazwa:** krótka, mająca formę wykonywanej operacji, a NIE rzeczy, zaczynająca się od numeru przypadku użycia; musi być identyczna z nazwą pokazaną na diagramie;
- **cel:** dłuższe wyjaśnienie, jeśli nazwa nie wyjaśnia przeznaczenia przypadku użycia;
- **warunki wstępne:** określają możliwość realizacji przypadku użycia; np. wymagania, których wcześniejsze spełnienie lub przypadki użycia, których wcześniejsze zrealizowanie pozwala rozpocząć wykonywanie opisywanego przypadku użycia;
- **warunki końcowe:** określają poprawne zrealizowanie przypadku użycia; np. wymagania spełnione lub przypadki użycia zrealizowane przez opisywany przypadek użycia;
- **scenariusz:** przepływ zdarzeń, które składają się na realizację przypadku użycia (algorytm); w postaci szczegółowego pseudokodu w języku naturalnym;
- **alternatywne scenariusz:** alternatywne przepływy zdarzeń (jeśli istnieją) z podaniem ich warunków wstępnych.

Scenariusz pokazuje:

- miejsce realizacji przypadku użycia, który jest w realizacji zawierania z opisywanym przypadkiem użycia;
- miejsce i warunek (opcjonalność, losowość, alternatywność) realizacji przypadku użycia, który jest w realizacji rozszerzania z opisywanym przypadkiem użycia.

Scenariusz nie pokazuje scenariusza innego przypadku użycia, w szczególności takiego, który jest połączony relacją zawierania lub rozszerzania.

Opracowano na podstawie instrukcji 2 i instrukcji 3 „Laboratorium z przedmiotu: Inżynieria Oprogramowania W04ITE-SI0011G”, autorstwa dr inż. Zofii Kruczkiewicz.
Te instrukcje zawierają też przykłady wykonania podobnych zadań i totoriale do programu Visual Paradigm.