II UWr, inżynieria oprogramowania 2024-25, ćwiczenia-pracownia, cd.

**DRUGA FAZA WYTWARZANIA OPROGRAMOWANIA – FAZA OPRACOWANIA**

**ZADANIA DO WYKONANIA W LISTOPADZIE**

**1. Precyzyjniejsze określenie wymagań**

Uwaga. Przez **system prosty** należy tu rozumieć system, który ma około cztery nietrywialne funkcje wymagające wprowadzania przez użytkownika co najmniej czterech danych alfanumerycznych oraz którego relacyjna baza danych składa się z co najmniej czterech tabel.

W ramach niniejszego zadania **należy starannie sporządzić i pieczołowicie sprawdzić dokument o nazwie „Specyfikacja wymagań”**, zawierający następujące elementy:

1. Historyjki użytkownika

W tej części należy opisać projektowany system za pomocą nie mniej niż trzech historyjek użytkownika. Na końcu tego dokumentu przedstawiono podstawowe informacje dotyczące historyjek użytkownika.

1. Wymagania funkcjonalne

W przypadku systemu prostego należy sporządzić listę wszystkich jego funkcji wraz z ich krótkimi opisami. Dla systemów innych niż proste, jeżeli system składa się z pewnej liczby podsystemów, należy wymienić i krótko opisać wszystkie główne systemy składowe. W odniesieniu do systemu, który studenci w grupach obiorą za przedmiot swoich prac do końca semestru, należy sporządzić dokładną listę funkcji oraz zwięzłe i jasne ich opisy.

1. Wymagania niefunkcjonalne

Dla obranego przez Państwa systemu należy określić co najmniej 5 podstawowych wymagań niefunkcjonalnych, zaproponować ich jednoznaczne definicje i przedstawić propozycje zweryfikowania tego, czy dane wymaganie niefunkcjonalne będzie spełnione przez zaimplementowany system.

**2. Dalsze szczegóły**

W odniesieniu do systemów prostych dla każdej funkcji rozumianej jako odpowiednik **przy- padku użycia** należy napisać dokładne (umożliwiające późniejsze testowanie) scenariusze dialogu człowieka z komputerem ( tzw. **scenariusze przypadków użycia**).

W przypadku systemów innych niż proste scenariusze przypadków użycia należy napisać dla trzech funkcji, które wymagają wprowadzenia przez użytkownika co najmniej 4 danych.

Zaleca się zapoznanie się ze treścią stron WWW traktujących o tym, jak pisać dobre scena- riusze przypadków użycia, na przykład:

# Dokumentacja wymagań w oparciu o przypadki użycia

<https://wolski.pro/2015/05/dokumentacja-wymagan-w-oparciu-o-przypadki-uzycia/>

Przypadki użycia

https://www.ii.pwr.edu.pl/~kazienko/projektowanie/USE%20CASES%20-%20prezentacja\_KNABEL&KRYVETS.pdf

# 7 złych praktyk tworzenia przypadków użycia

# https://analizatozalezy.pl/index.php/2020/11/18/7-zlych-praktyk-tworzenia-przypadkow-uzycia/

**3. Projekty dialogów i dokumentów**

Dla funkcji, dla których sporządzono scenariusze przypadków użycia, należy zaprojektować dialogi z komputerem na poziomie fizycznym, czyli ukazywane ekrany („screeny”, formatki ekra-owe itp.) i jeżeli dany scenariusz przypadku użycia przewiduje drukowanie dokumentów (np. biletów lub faktur), trzeba również zaprojektować postać tych dokumentów.

**4. Zaprojektowanie architektury systemu**

W tej części zadania należy wymienić oraz przedstawić graficzne podstawowe elementy systemu oraz powiązania między nimi, w szczególności:

* przewidywane rozwiązania sprzętowe;
* oprogramowanie systemowe, bazy danych, narzędzia programistyczne;
* oprogramowanie do automatycznego testowania;
* strukturę logiczną oprogramowania (podział kodu na główne komponenty np. klasy).

**5. Czynności i zadania uzupełniające**

Omówienie interfejsów z innymi systemami

5.A. Sporządzenie **modelu konceptualnego** rzeczywistości, której dotyczy system.

5.B. Sporządzenie **schematu bazy danych** systemu, tzn. diagramu pokazującego tabele, ich klucze i podstawowe pola oraz powiązania między tabelami i warunki integralności danych.

**6. Przedstawienie głównych zasad kodowania** na podstawie dostępnych materiałów (np. osiągalnych w Sieci).

7. Zidentyfikowanie zagrożeń i opracowanie zasad zarządzania **ryzykiem**. Uwaga: pomijamy ryzyko polegające na malej liczbie klientów, krachu świtowego lub braku pieniędzy.

8. Ocena zgodności wykonanych prac z wizją i założeniami przedstawionymi w **tablicy koncepcyjnej** i **specyfikacji wymagań**.

**Wykonanie powyższych zadań (od 2 do 5) powinno być udokumentowane w stosownym dokumencie o nazwie „Koncepcja wykonania systemu”.**

**\***

DODATEK. UWAGI O SPORZĄDZANIU HISTORYJEK UŻYTKOWNIKA

# Należy zapoznać się z filmem zatytułowanym

# Jak tworzyć prawdziwe Historie Użytkownika (User Stories)? https://www.youtube.com/watch?v=uEe0t6az7uA

**1. Podstawowe informacje dotyczące historyjek użytkownika**

**Historyjka (historia) użytkownika** (ang. *user story*) określa wysokopoziomowe wymaganie użytkownika lub wymaganie biznesowe i jest często używanym środkiem w zwinnym wytwa- rzaniu oprogramowania.

Historyjka użytkownika w typowych sytuacjach składa się z jednego lub kilku zdań wyra-żonych w języku potocznym lub żargonie biznesowym, opisujących funkcję lub cechę potrzebną użytkownikowi, kryteria niefunkcjonalne i (lub) kryteria akceptacji. Historyjki użytkownika nie powinny zawierać szczegółów technicznych, za to powinny być łatwe do napisania i w całości mieścić się na jednej kartce typu *post-it*.

Zadaniem analityka jest, oprócz opisania (włączając w to nagrania, rysunki, filmiki) historyjki użytkowania, znalezienie scenariuszy negatywnych oraz założeń do danej historyjki.

Według Roya Jeffersa twórcy koncepcji **programowania radykalnego** (ang. *eXtreme Programming*), historyjki użytkownika powinny składać się z trzech elementów: karty opisującej daną funkcję, konwersacji, tzn. zaproszenia do rozmowy o potrzebnej funkcji i (lub) wymaga- niach niefunkcjonalnych, oraz konfirmacji, czyli potwierdzenia weryfikującymi je testami.

Cechy dobrej historyjki opisuje akronim **INVEST** (ang. *Independent, Negotiable, Valuable, Estimable, Small, Testable*) wymyślony przez Billa Wake’a. Historyjki użytkownika powinny więc być:

* **Niezależne** – poszczególne historyjki powinny być niezależne od siebie, tak aby można je było łatwiej zaplanować.
* **Negocjowalne** – powinny dawać się łatwo zmieniać i dopasowywać), tworząc płaszczyznę porozumienia między wszystkimi zainteresowanymi stronami (tzw. interesariuszami).
* **Wartościowe** – powinny mieć jakąś wartość dla kogoś – niekoniecznie dla użytkownika – na przykład dla osoby testującej (testera).
* **Szacowalne** – dla każdej historyjki powinna istnieć rozsądnego określenia czasu..
* **Małe** na tyle, aby można określić potrzebne zadania oraz oszacować pracochłonność.
* **Testowalne** – należy zweryfikować historyjki pod kątem niejasności, sprzeczności, dwuzna- czności, niepewności.

Historyjki mogą mieć szablon, na przykład:

Jako <typ użytkownika> – dla kogo jest dana historia –

chce <jakiś cel> – co on chce zrobić,

żeby <jakiś powód> – dlaczego chce to zrobić.

LGR, II UWr 2024