**Inżynieria oprogramowania 2020-21**

**ćwiczenia/pracowania**

**II FAZA WYTWARZANIA OPROGRAMOWANIA**

**- FAZA OPRACOWANIA -**

**ZADANIA DO WYKONANIA W LISTOPADZIE**

1. Bardziej precyzyjne określenie wymagań

Uwaga:

Przez system prosty należy rozumieć system, który NIE POSIADA co najmniej 3

funkcji, które wymagają wprowadzenia przez użytkownika co najmniej 4 danych

alfanumerycznych oraz którego relacyjna baza danych składa się z co najmniej

z 4 tabel.

1. Historyjki użytkownika

Wymagania należy opisać w postaci nie mniej niż 3 historyjek użytkownika.

Na końcu dokumentu przedstawione są podstawowe informacje nt historyjek użytkownika.

1. Wymagania funkcjonalne

Dla systemów prostych należy sporządzić listę wszystkich funkcji i ich krótki opis.

Dla systemów innych niż proste:

- Jeżeli system składa się z systemów składowych to należy wymienić i krótko opisać

wszystkie główne systemy składowe.

- Dla tego systemu którym będziecie się Państwo zajmować do końca semestru

należy sporządzić dokładną listę funkcji oraz krótki ich opis.

1. Wymagania niefunkcjonalne

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

**Wykonanie tego zadania powinno być udokumentowane w odpowiednim dokumencie o nazwie Specyfikacja wymagań.**

2.

Dla systemów prostych dla każdej funkcji rozumianej jako odpowiednik przypadku 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 3 funkcji, które wymagają wprowadzenia przez użytkownika co najmniej 4 danych.

Zaleca się zapoznanie się ze treścią stron www traktujących o tym jak pisać dobre scenariusze przypadków użycia, np.:

http://arturgula.aretech.pl/2016/06/05/10-wskazowek-jak-skutecznie-pisac-przypadki-uzycia/

czy

<http://analizait.pl/2012/jak-opisywac-przypadki-uzycia/>

czy

<https://www.ii.pwr.edu.pl/~kazienko/projektowanie/USE%20CASES%20-%20prezentacja_KNABEL&KRYVETS.pdf>

czy

http://www.cs.put.poznan.pl/bmichalik/doc/wymagania.pdf

3. Dla funkcji, dla których sporządzono scenariusze przypadków użycia należy projektować na

poziomie fizycznym dialog z komputerem czyli ukazujące się ekrany ( screeny, formatki

ekranowe) i jeżeli dany scenariusz przypadku użycia przewiduje drukowanie dokumentów

np. biletów, faktur to trzeba te dokumenty też zaprojektować.

4.. Zaprojektowanie architektury:

Wymienienie oraz przedstawienie graficzne podstawowych elementów oraz powiązań

między nimi: - sprzęt,

- oprogramowanie systemowe, bazy danych, narzędzia

programistyczne, oprogramowanie do automatycznego testowania.

- struktura logiczna oprogramowania

( podział kodu na główne komponenty np. klasy )

Omówienie interfejsów z innymi systemami.

4.A Sporządzenie modelu konceptualnego rzeczywistości, której dotyczy system.

4.B Sporządzenie schematu bazy danych systemu tzn diagramu pokazującego tabele,

ich klucze i podstawowe pola oraz powiazania między tabelami i warunki integralności

danych.

5. Przedstawienie głównych zasad kodowania

( oczywiście na podstawie dostępnych np. w sieci materiałów).

6. Identyfikacja ryzyka i opracowanie zasad zarządzania ryzykiem.

Uwaga: pomijamy ryzyko polegające na malej liczbie klientów czy braku pieniędzy.

7. Ocena zgodności wykonanych prac z wizją przedstawioną w tablicy koncepcyjnej i

specyfikacji wymagań.

**Wykonanie powyższych tzn od 2 do 7 zadań powinno być udokumentowane w odpowiednim dokumencie o nazwie Koncepcja wykonania systemu.**

**--------**

**HISTORYJKA UŻYTKOWNIKA**

Należy zapoznać się z filmem Altkom Akademia:

User Story – zwinne definiowanie wymagań

<https://www.youtube.com/watch?v=NRiaTM9t5oc&t=2175s>

**Podstawowe informacje nt historyjki użytkownika**

**Historyjka użytkownika (ang. user story)** – wysokopoziomowe wymaganie użytkownika lub wymaganie biznesowe, często używane w zwinnym wytwarzaniu oprogramowania.

W typowych sytuacjach składa się z jednego lub kilku zdań w codziennym lub biznesowym języku, opisujące funkcjonalność potrzebna użytkownikowi opisujące funkcjonalność potrzebna użytkownikowi, kryteria niefunkcjonalne oraz kryteria akceptacji.

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 ( też nagrania, rysunki, filmiki) historyjki użytkowania – znalezienie scenariuszy negatywnych oraz założeń do danej historyjki.

Wg Roya Jeffersa twórcy eXtreme Programming, historyjki użytkownika powinny składać się z 3 komponentów: karty opisującej funkcjonalność, konwersacji (ma być zaproszeniem do rozmowy o funkcjonalności i wymaganiach niefunkcjonalnych ) oraz konfirmacji (inaczej: potwierdzenia czyli testów, które je zweryfikują).

Cechy dobrej historyjki opisuje **akronim INVEST (Independent, Negotiable, Valuable, Estimable, Small, Testable)** wymyślone przez Billa Wake’a.

**Niezależne** – poszczególne historyjki powinny być niezależne od siebie tak, aby można je było łatwiej zaplanować.

**Negocjowalne** – powinny móc podlegać różnego rodzaju zmianom i tworzyć płaszczyznę porozumienia między wszystkimi interesariuszami.

**Wartościowe** – powinny nieść jakaś wartość dla kogoś, niekoniecznie dla użytkownika – może być np. wartość dla testera.

**Szacowalne** – dla każdej historyjki powinna istnieć możliwość określenia z rozsądnym przybliżeniem czasy

**Małe** – tak aby można określić potrzebne zadania oraz oszacować pracochłonność.

**Testowalne** – należy zweryfikować historyjki pod katem niejasności, sprzeczności, dwuznaczności, niepewności.

Historyjki mogą mieć szablon, np.:

Jako <typ użytkownika> dla kogo jest ta historia

chce <jakiś cel> co on chce zrobić

żeby <jakiś powód> dlaczego chce to zrobić