INŻYNIERIA OPROGRAMOWANIA

wykład 4:
WYMAGANIA

dr inż. Leszek Grocholski

Zakład Inżynierii Oprogramowania
Instytut Informatyki
Uniwersytet Wrocławski

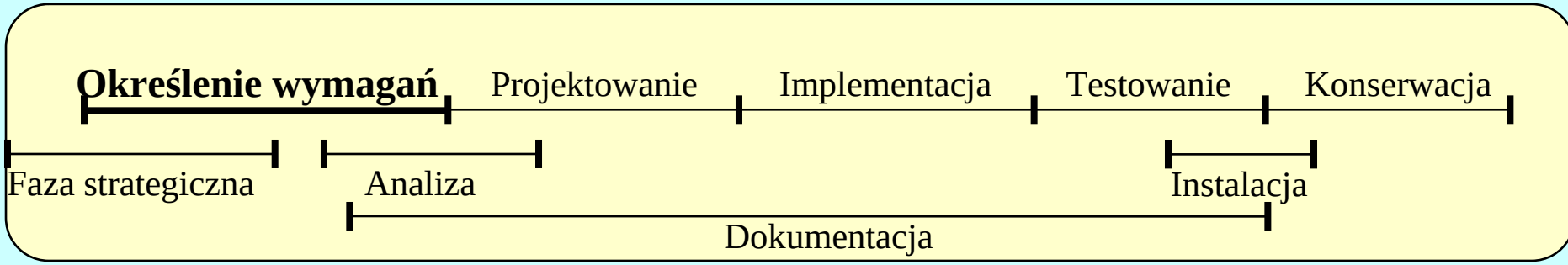
Plan wykładu

1. Problemy dot. określenia wymagań
2. Poziomy ogólności opisu wymagań
3. Ocena jakość opisu wymagań
4. Sposoby identyfikacji wymagań
5. Wymagania funkcjonalne
6. Porządkowanie wymagań wg. ich ważności
7. Wymagania niefunkcjonalne
8. Czynniki uwzględniane przy konstruowaniu wymagań
9. Dokument wymagań

Określenie wymagań

Celem fazy określenia wymagań jest ustalenie wymagań klienta wobec tworzonego systemu. Dokonywana jest zamiana celów klienta na konkretne wymagania zapewniające osiągnięcie tych celów.

Klient NIE ZAWSZE WIE, jakie wymagania zapewnią osiągniecie jego celów. Bardzo często wymagania są sformułowane na wysokim poziomie.
Ta faza nie jest więc prostym zbieraniem wymagań, lecz procesem, w którym klient wspólnie z przedstawicielem producenta konstruuje zbiór wymagań zgodnie z postawionymi celami.



W przypadku dedykowanego (wytwarzanego specjalnie na systemu na zamówienie) analitycy mają bezpośredni kontakt z przedstawicielami klienta. Faza ta wymaga **dużego zaangażowania ze strony klient**a, ze strony przyszłych użytkowników systemu i ekspertów w dziedzinie.

Problemy dot. określenia wymagań

Wymagań nawet dla „małych” systemów jest bardzo dużo. Bardzo trudno je wszystkie zidentyfikować, dokładnie opisać i zweryfikować.

Klient rzadko kiedy wie DOKŁADNIE w jaki sposób osiągnąć założone cele. Tymczasem cele klienta mogą być osiągnięte na wiele sposobów.

Duże systemy są wykorzystywane przez wielu użytkowników.
Ich cele są często sprzeczne.

Różni użytkownicy mogą posługiwać się RÓŻNĄ terminologią mówiąc o tych samych zagadnieniach.

Zleceniodawcy i użytkownicy to często inne osoby. Głos zleceniodawców może być w tej fazie decydujący, chociaż nie zawsze potrafią oni właściwie przewidzieć potrzeby przyszłych użytkowników.

Poziomy opisu wymagań

Definicja wymagań, to ogólny opis w języku naturalnym. Opis taki jest rezultatem wstępnych rozmów z klientem.

Specyfikacja wymagań, to częściowo ustrukturalizowany zapis wykorzystujący zarówno język naturalny, jak i proste, częściowo przynajmniej sformalizowane notacje.

Specyfikacja oprogramowania, to formalny opis wymagań.

Dokładne specyfikacja oznacza bardzo dokładne zdekomponowanie wymagań (najlepiej w pewnym formularzu) na krótkie punkty, których interpretacja nie powinna nastręczać trudności lub prowadzić do niejednoznaczności. Formalna specyfikacja powinna stanowić podstawę dla fazy testowania.

Ocena jakości opisu wymagań

Dobry opis wymagań powinien:

- Być kompletny oraz niesprzeczny.

- Opisywać zewnętrzne zachowanie się systemu a nie sposób jego realizacji.

- Obejmować ograniczenia przy jakich musi pracować system.

- Być łatwy w modyfikacji.

- Brać pod uwagę przyszłe możliwe zmiany wymagań wobec systemu.

- Opisywać zachowanie systemu w niepożądanych lub skrajnych sytuacjach.

Najbardziej typowy błąd w tej fazie: koncentrowanie się na sytuacjach typowych i pomijanie wyjątków oraz przypadków granicznych. Zarówno użytkownicy jak i analitycy mają tendencję do nie zauważania sytuacji nietypowych lub skrajnych.

Zalecenia dla fazy definicji wymagań

Wymagania użytkowników powinny być wyjaśniane poprzez:

-**krytykę,**

-**porównania z istniejącym oprogramowaniem i prototypami.**

Należy osiągnąć porozumienie - consensus pomiędzy projektantami i użytkownikami dotyczący projektowanego systemu.

Wiedza i doświadczenia potencjalnej organizacji podejmującej się rozwoju systemu powinny wspomóc studia nad osiągalnością systemu (omawiane feasibility study).

W wielu przypadkach dużym wspomaganiem jest budowa prototypów.

Uwaga: wymagania użytkowników powinny być: **jasne, jednoznaczne, weryfikowalne, kompletne, dokładne, realistyczne, osiągalne.**

Sposoby rozpoznania wymagań

Studia przeznaczenia systemu. Określenie realistycznych celów systemu, zakresu i metod ich osiągnięcia.

Historyjki użytkownika. Opowiadania użytkowników dot. oczekiwanego sposobu działania systemu.

Wywiady i przeglądy. Wywiady powinny być przygotowane (w postaci listy pytań) i podzielone na odrębne zagadnienia. Podział powinien przykrywać całość tematu a wywiady powinny być przeprowadzone na reprezentatywnej grupie użytkowników. Wywiady powinny doprowadzić do szerokiej zgody i akceptacji projektu.

Studia na istniejącym oprogramowaniem. Dość często nowe oprogramowanie zastępuje stare. Studia powinny ustalić wszystkie dobre i złe strony starego oprogramowania.

Studia wymagań systemowych. Dotyczy sytuacji, kiedy nowy system ma być częścią większego systemu.

Prototypowanie – analiza czy taki system jest potrzebny? Zbudowanie prototypu systemu działającego w zmniejszonej skali, z uproszczonymi interfejsami.

Wymagania funkcjonalne

Opisują „co system ma robić”.
Opisują funkcje (czynności, operacje) wykonywane przez system.
Funkcje te mogą być również wykonywane przy użyciu systemów zewnętrznych.

Określenie wymagania funkcjonalnych obejmuje następujące kwestie:

- Określenie wszystkich rodzajów użytkowników, **którzy będą korzystać** z systemu.

- Określenie wszystkich rodzajów użytkowników, którzy **są niezbędni do działania** systemu (obsługa, wprowadzanie danych, administracja).

- **Dla każdego rodzaju użytkownika określenie funkcji systemu oraz sposobów korzystania z planowanego systemu.**

- Określenie systemów zewnętrznych (obcych baz danych, sieci, Internetu), które będą wykorzystywane podczas działania systemu.

- Ustalenie struktur organizacyjnych, przepisów prawnych, statutów, zarządzeń, instrukcji, itd., które pośrednio lub bezpośrednio określają funkcje wykonywane przez planowany system.

Metody specyfikacji wymagań

Słowa, słowa, słowa …

Język naturalny - najczęściej stosowany. Wady: *niejednoznaczność* powodująca różne rozumienie tego samego tekstu; *elastyczność*, powodująca wyrazić te same treści na wiele sposobów. **Np. historyjki użytkownika.**

Formalizm arytmetyczny (matematyczny). Stosuje się rzadko (dla specyficznych celów).

Język naturalny strukturalny. Język naturalny z ograniczonym słownictwem i składnią. Tematy i zagadnienia wyspecyfikowane w punktach i podpunktach.

Tablice, formularze. Wyspecyfikowanie wymagań w postaci (zwykle dwuwymiarowych) tablic, kojarzących różne aspekty (np. tablica ustalająca zależność pomiędzy typem użytkownika i rodzajem usługi).

Różne rysunki (popularny UML):

- Diagramy blokowe: forma graficzna pokazująca cykl przetwarzania.

- Diagramy kontekstowe: ukazują system w postaci jednego bloku oraz jego powiązania z otoczeniem, wejściem i wyjściem.

- Diagramy przypadków użycia: poglądowy sposób przedstawienia aktorów i funkcji systemu.

Formularz wymagań

W formularzach zapis jest podzielony na konkretne pola, co pozwala na łatwe stwierdzenie kompletności opisu oraz na jednoznaczną interpretację.

Przykład:

Nazwa funkcji	*Edycja dochodów podatnika*
Opis	Funkcja pozwala edytować łączne dochody podatnika uzyskane w danym roku.
Dane wejściowe	Informacje o dochodach pracowników uzyskane uzyskanych z różnych źródeł: kwoty przychodów, koszty uzyskania przychodów oraz zapłaconych zaliczek na poczet podatku dochodowego. Informacje o dokumentach opisujących dochody z poszczególnych źródeł.
Źródło danych wejściowych	Dokumenty oraz informacje dostarczone przez podatnika. Dane wpisywane przez pracownika firmy podatkowej.
Wynik	
Warunek wstępny	Kwota dochodu = kwota przychodu - kwota kosztów (zarówno dla konkretnych dochodów, jak i dla łącznych dochodów podatnika). Łączne kwoty przychodów, kosztów uzyskania dochodów oraz zapłaconych zaliczek są sumami tych kwot dla dochodów z poszczególnych źródeł.
Warunek końcowy	Jak wyżej.
Efekty uboczne	Uaktualnienie podstawy opodatkowania.
Powód	Funkcja pomaga przyśpieszyć obsługę klientów oraz zmniejszyć ryzyko popełnienia błędów.

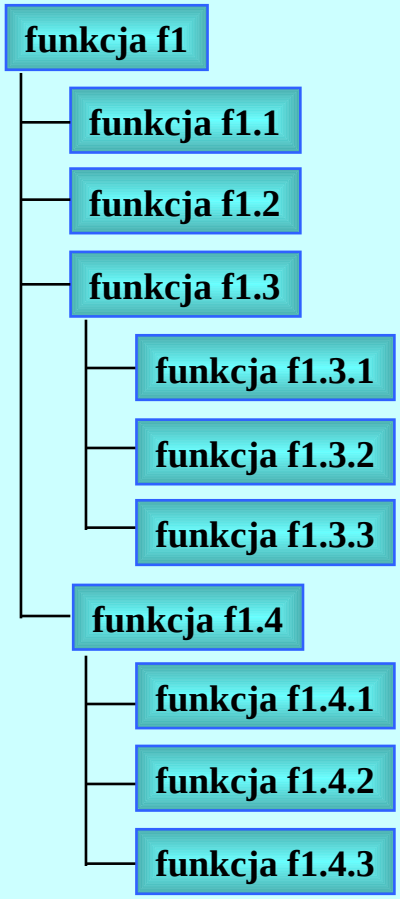
Porządkowanie wymagań

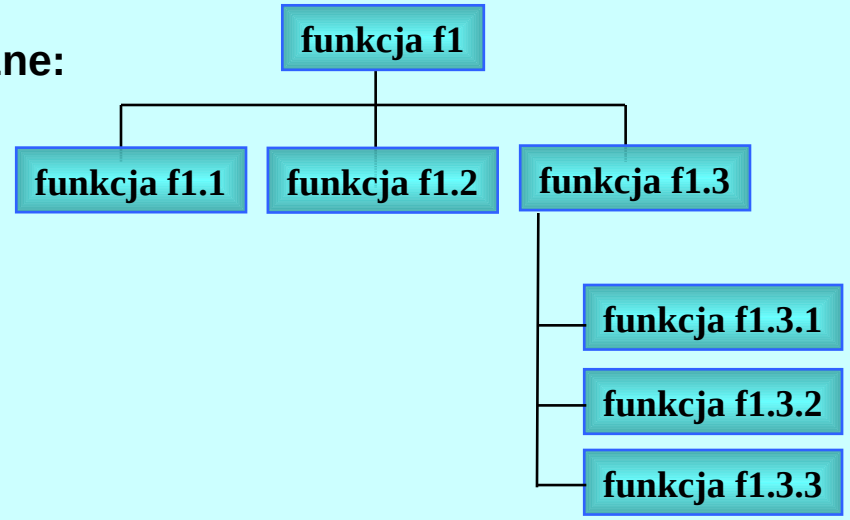
Hierarchia wymagań funkcjonalnych:
Z reguły funkcje można rozbić na podfunkcje.

Format tekstowy:

Funkcja nadrzędna f1
- funkcja f1.1
- funkcja f1.2
- funkcja f1.3
  - funkcja f1.3.1
  - funkcja f1.3.2
  - funkcja f1.3.3
- funkcja f1.4
  - funkcja f1.4.1
  - funkcja f1.4.2
  - funkcja f1.4.3

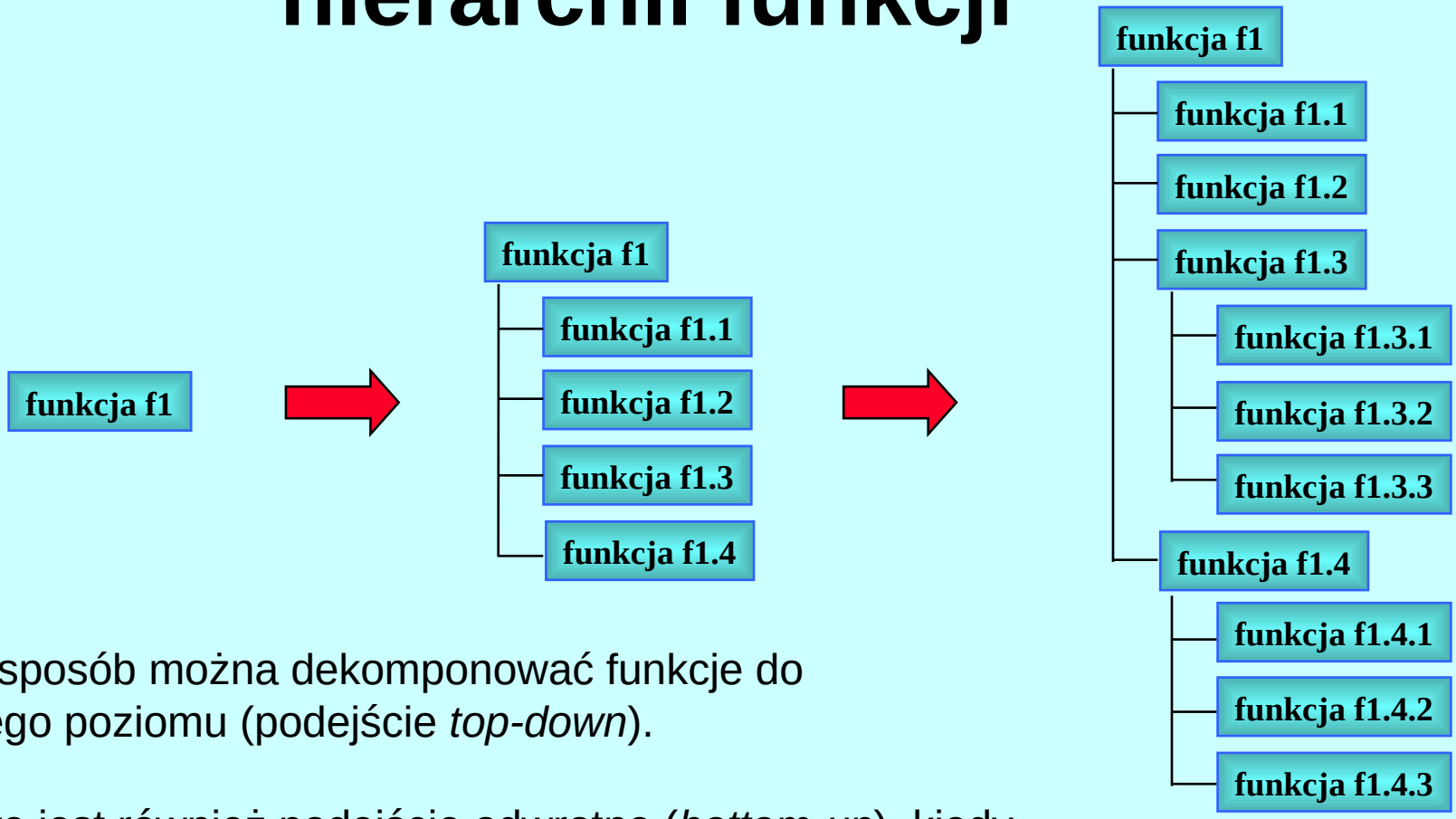
Notacje graficzne:





- Na każdym poziomie ten sam poziom szczegółowości.
- Kolejność funkcji nie ma znaczenia.
- Niektóre funkcje mogą być wykonywane wielokrotnie.
- Wyszczególniać tylko funkcje widoczne dla użytkowników.

Zstępujące konstruowanie hierarchii funkcji



- W ten sposób można dekomponować funkcje do dowolnego poziomu (podejście *top-down*).

- Możliwe jest również podejście odwrotne (*bottom-up*), kiedy składamy kilka funkcji bardziej elementarnych w jedną funkcje bardziej ogólną. Możliwa jest również technika mieszana.

P1 program podatkowy

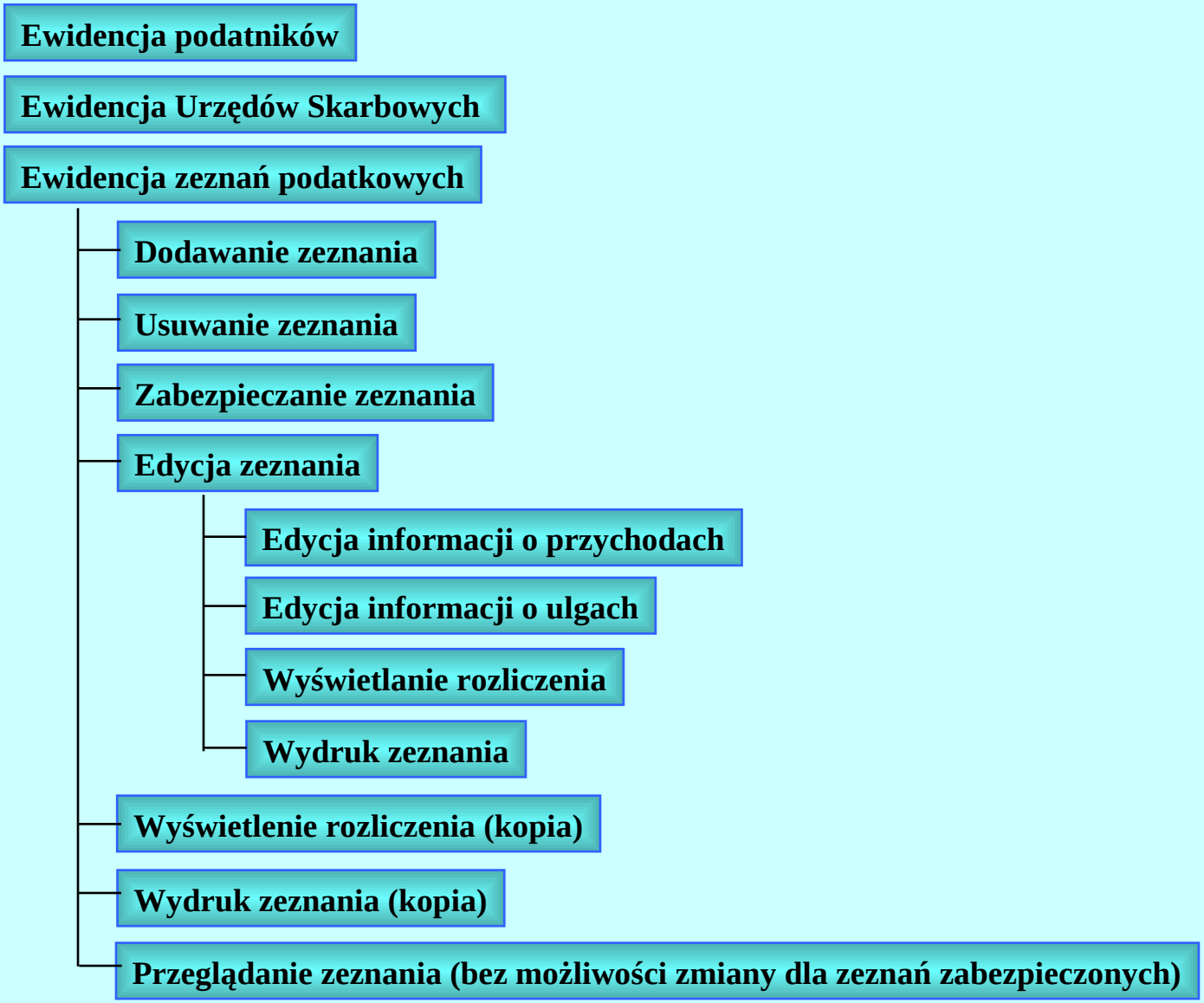
Przykład opowiadania (powieść?) użytkownika

Program ułatwia przygotowanie formularzy zeznań podatkowych (PIT-ów) oraz przechowanie informacji o źródłach przychodów i ulg. Zeznanie może być tworzone przez pojedynczego podatnika lub małżeństwa. Zeznania mogą obejmować informacje o rocznych przychodach (w przypadku małżeństwa z podziałem na przychody męża i żony) oraz o ulgach podatkowych. Przychody podzielone są na klasy ze względu na źródło uzyskania, np. poza-rolnicza działalność gospodarcza, wolny zawód,... . W ramach danej klasy przychodów podatnik mógł osiągnąć szereg przychodów z różnych źródeł.

Wszystkie przychody opisane są przez kwotę przychodu, kwotę kosztów, kwotę zapłaconych zaliczek oraz kwotę dochodu. Powyższe informacje pozwalają obliczyć należny podatek oraz kwotę do zapłaty lub zwrotu. Zeznanie zawiera także informację o podatniku oraz adres Urzędu Skarbowego.

System pozwala wydrukować wzorzec zeznania zawierający wszystkie informacje, jakie podatnik musi umieścić w formularzu. Zeznanie można zabezpieczyć przed dalszymi zmianami (po złożeniu w Urzędzie Skarbowym). System pozwala na tworzenie listy podatników oraz urzędów skarbowych, które mogą być pomocne przy tworzeniu nowego zeznania. Przechowuje także informację o wszystkich złożonych zeznaniach.

Program podatkowy: hierarchia funkcji



P2: system harmonogramowania

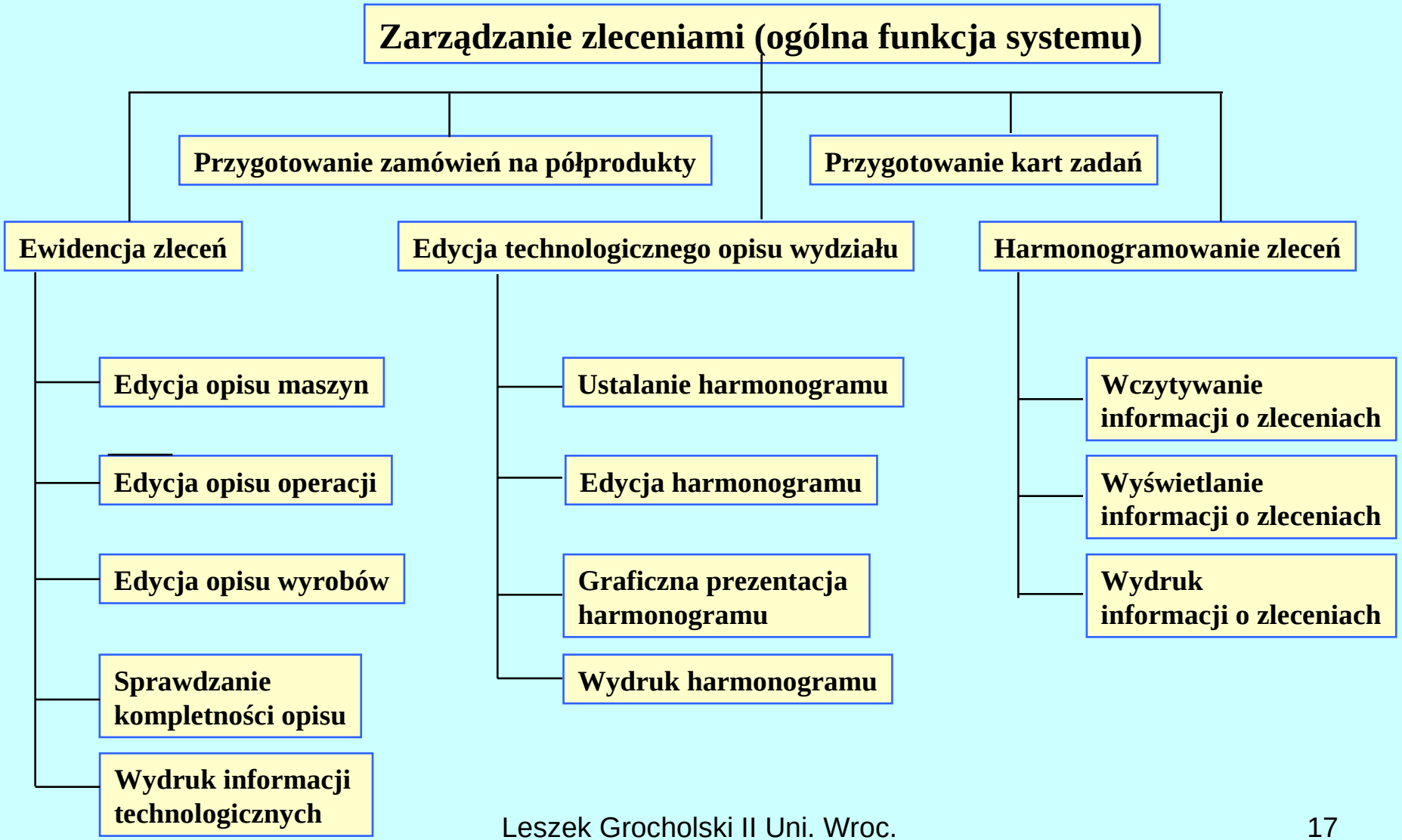
Opowiadanie (powieść?) użytkownika

Zlecenia dla wydziału przygotowywane są przez dział marketingu. Zlecenie oznacza konieczność wyprodukowania konkretnej ilości pewnego wyrobu przed upływem konkretnego terminu. Czasami może być określony najwcześniejszy pożądany termin realizacji. Dział marketingu wykorzystuje własny system informatyczny, w którym między innymi umieszczane są informacje o zleceniach, pożądane jest więc, aby system harmonogramowania zleceń automatycznie odczytywał te informacje.

Wyprodukowanie danego wyrobu (realizacja zlecenia) wymaga wykonania ciągu operacji. Pewne operacje mogą być wykonywane tylko na jednym urządzeniu; w innych przypadkach możliwe jest wykorzystanie kilku maszyn, które mogą różnić się efektywnością pracy. Po wykonaniu pewnych operacji może być konieczna przerwa, zanim rozpocznie się realizacja następnych; z drugiej strony taka przerwa może trwać przez ograniczony czas. Przestawienie maszyn na inne operacje może wymagać czasu. Co pewien czas (np. raz na miesiąc) ustalany jest harmonogram niezrealizowanych zleceń.

System powinien opracować harmonogramy w formie łatwej do wykorzystania przez kadrę kierowniczą wydziału oraz przygotowywać zamówienia do magazynu na półprodukty. Zlecenia wykonane są usuwane ze zbioru niezrealizowanych zleceń.

System harmonogramowania zleceń: funkcje



Diagramy przypadków użycia

Opis funkcji systemu z punktu widzenia jego użytkowników.
Opis wymagań poszczególnych użytkowników jest bardziej przejrzysty niż opis wszystkich wymagań wobec systemu.

Klasy użytkowników:

- projektant
- użytkownik - osoba przeglądająca mapę

Metoda modeluje **aktorów**, a nie konkretne **osoby**. Jedna osoba może pełnić rolę wielu aktorów; np. być jednocześnie projektantem i osobą przeglądającą mapę.
I odwrotnie, jeden aktor może odpowiadać wielu osobom, np. projektant.

Identyfikacja funkcji dla poszczególnych użytkowników.
Przeprowadzając wywiad z konkretna osobą należy koncentrować się na funkcjach istotnych z punktu widzenia roli (ról) odgrywanych przez tę osobę.

Metoda przypadków użycia nie jest sprzeczna z hierarchiczną dekompozycją funkcji.

Przykład: SIG

Opowiadanie (powieść?) użytkownika
SIG jest rodzajem mapy komputerowej, składającej się z tła (np. mapy fizycznej) oraz szeregu obiektów graficznych umieszczonych na tym tle.

Obiekt może być punktem (budynek, firma), linią (rzeka, kolej) lub obszarem (park, osiedle).

Każdy obiekt ma swoją nazwę i ewentualny opis, który może być wyświetlony na żądanie użytkownika. Obiekt można opisać szeregiem słów kluczowych.

Użytkownik ma możliwość wyświetlenia tylko tych obiektów, które opisano słowami kluczowymi.

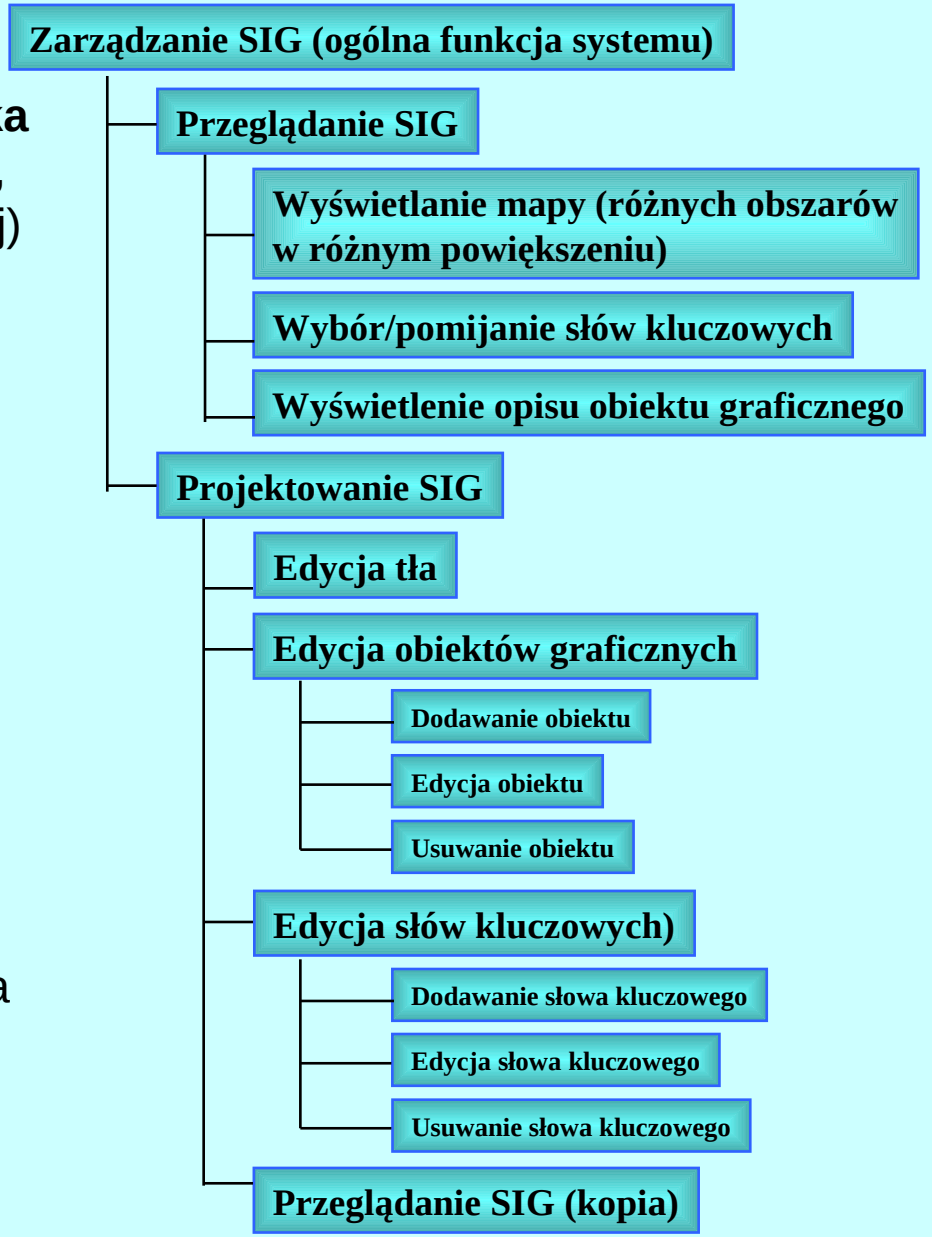
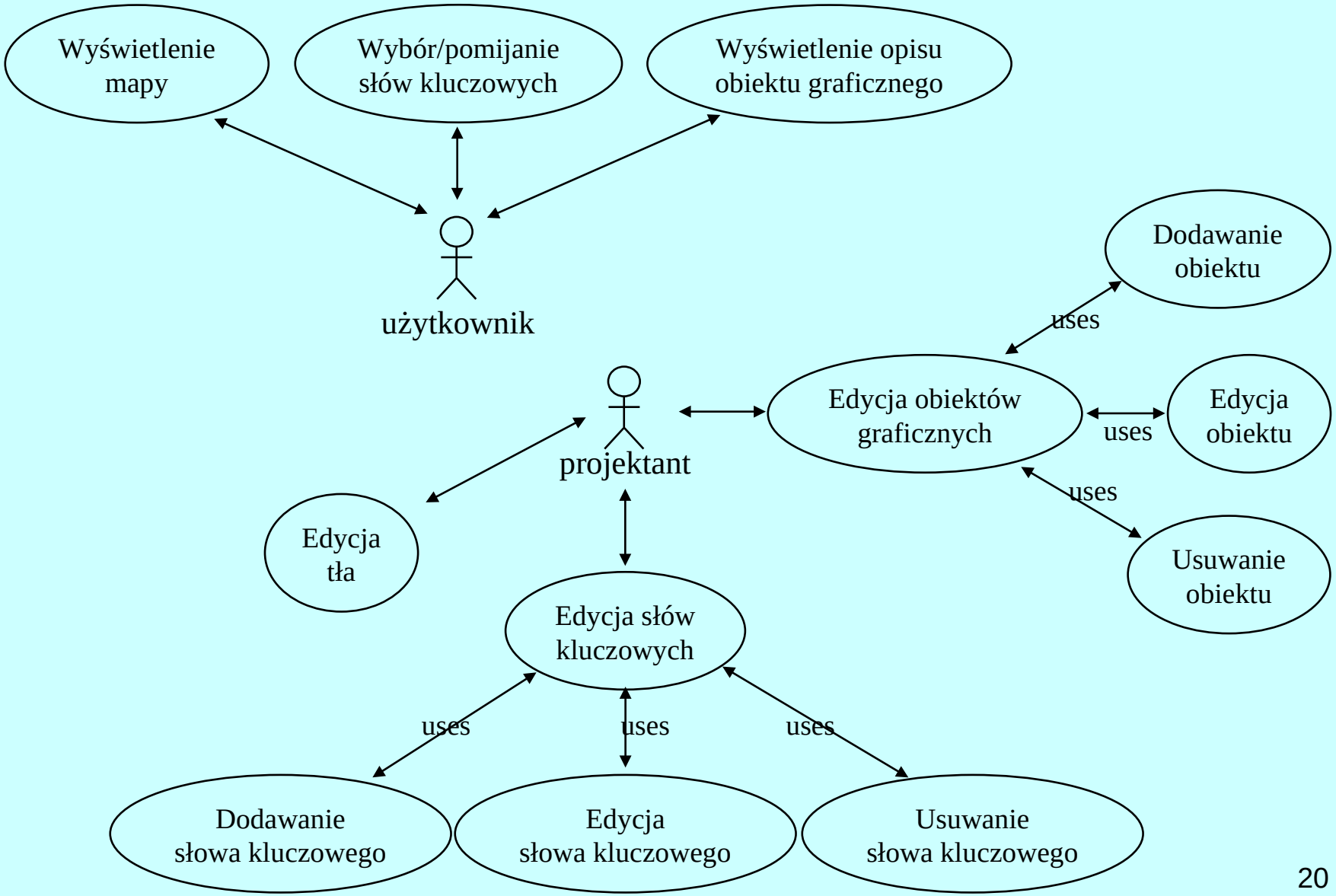


Diagram przypadków użycia dla SIG



Wymagania niefunkcjonalne

Określają „ w jaki sposób system ma realizować funkcje”
Inaczej - opisują ograniczenia, przy których system ma realizować swoje funkcje.

PODZIAŁ NA:

- Wymagania dotyczące produktu

Np. Musi istnieć możliwość operowania z systemem wyłącznie za pomocą klawiatury, np. bankomat.

- Wymagania dotyczące procesu

Np. Zarządzanie projektem ma się odbywać zgodnie z metodyką PRINCE 2.

- Wymagania zewnętrzne

Np. wykorzystanie standardów GPS, GLONASS (standard rosyjski nawigacji satelitarnej obejmujący zasięgiem całą kule ziemską).

Formularz zapisu wymagań nieunkcjonalnych

Nr	Data wprow.	Rozmówca	Wymaganie	Motywacja	Uwagi
1	99/04/14	A.Nowak J.Pietrzak	System powinien zwracać wynik zapytania po max 5-ciu sekundach przy 100 użytkownikach pracujących jednocześnie.	Inaczej system nie będzie konkurencyjny na rynku	Może być niestabilne
2	00/02/05	K.Lubicz	Każdy klient powinien mieć przypisany krótki numer identyfikacyjny	Inne identyfikatory niż PESEL np. numer telefonu są niestabilne, nie unikalne, lub za długie	
3	.....	.....	.....	....	.....

Weryfikowalność wymagań niefunkcjonalnych

Wymagania niefunkcjonalne powinny być **weryfikowalne -** czyli powinna istnieć możliwość sprawdzenia lub zmierzenia czy system je rzeczywiście spełnia.
Np. wymagania "*system ma być łatwy w obsłudze*", „*system ma być niezawodny*", „*system ma być dostatecznie szybki*", itd. nie są weryfikowalne.

Cecha	***Weryfikowalne miary***
Wydajność	Liczba transakcji obsłużonych w ciągu sekundy Czas odpowiedzi
Rozmiar	Liczba rekordów w bazie danych Wymagana pamięć dyskowa
Łatwość użytkowania	Czas niezbędny dla przeszkolenia użytkowników Rozmiar dokumentacji
Niezawodność	Prawdopodobieństwo błędu podczas realizacji transakcji Średni czas pomiędzy błędnymi wykonaniami Dostępność (procent czasu w którym system jest dostępny) Czas restartu po awarii systemu Prawdopodobieństwo zniszczenia danych w przypadku awarii
Przenaszalność	Procent kodu zależnego od platformy docelowej Liczba platform docelowych Koszt przeniesienia na nową platformę

Konstruowanie wymagań niefunkcjonalnych (1)

Możliwości systemu: Zestaw funkcji, które ma wykonywać system, uporządkowany hierarchicznie.

Objętość: Ilu użytkowników będzie pracować jednocześnie? Ile terminali ma być podłączone do systemu? Ile czujników będzie kontrolowanych jednocześnie? Ile danych będzie przechowywane?

Szybkość: Jak długo może trwać operacja lub sekwencja operacji? Liczba operacji na jednostkę czasu. Średni czas niezbędny dla jednej operacji.

Dokładność: Określenie stopnia precyzji pomiarów lub przetwarzania. Określenie wymaganej dokładności wyników. Zastąpienie wyników ilościowych jakościowymi lub odwrotnie.

Ograniczenia: ograniczenia na interfejsy, jakość, skalę czasową, sprzęt, oprogramowanie, skalowalność, itd.

Konstruowanie wymagań niefunkcjonalnych (2)

Interfejsy komunikacyjne: sieć, protokoły, wydajność sieci, poziom abstrakcji protokołów komunikacyjnych, itd.

Interfejsy sprzętowe: specyfikacja wszystkich elementów sprzętowych, które będą składały się na system, fizyczne ograniczenia (rozmiar, waga), wydajność (szybkość, RAM, dysk, inne pamięci), wymagania co do powierzchni lokalowych, wilgotności, temperatury i ciśnienia, itd.

Interfejsy oprogramowania: Określenie zgodności z innym oprogramowaniem, określenie systemów operacyjnych, języków programowania, kompilatorów, edytorów, systemów zarządzania bazą danych, itd.

Interakcja człowiek-maszyna: Wszystkie aspekty interfejsu użytkownika, rodzaj języka interakcji, rodzaj sprzętu (monitor, mysz, klawiatura), określenie formatów (układu raportów i ich zawartości), określenie komunikatów dla użytkowników (język, forma), pomocy, komunikatów o błędach, itd.

Konstruowanie wymagań niefunkcjonalnych (3)

Adaptowalność: Określenie w jaki sposób będzie organizowana reakcja na zmiany wymagań: dodanie nowej komendy, dodanie nowego okna interakcji, itd.

Bezpieczeństwo: założenia co do poufności, prywatności, integralności, odporności na hakerów, wirusy, wandalizm, sabotaż, itd.

Odporność na awarie: konsekwencje błędów w oprogramowaniu, przerwy w zasilaniu, kopie zabezpieczające, częstotliwości składowania, dziennika zmian, itd.

Standardy: Określenie dokumentów standardyzacyjnych, które mają zastosowanie do systemu: formaty plików, normy czcionek, polonizacja, standardy procesów i produktów, itd.

Zasoby: Określenie ograniczeń finansowych, ludzkich i materiałowych.

Skala czasowa: ograniczenia na czas wykonania systemu, czas szkolenia, wdrażania, itd.

Dokument wymagań

- Wymagania powinny być zebrane w dokumencie - opisie wymagań.
- Dokument ten powinien być podstawą szczegółowego kontraktu między klientem a producentem oprogramowania.
- Powinien także pozwalać na weryfikację stwierdzającą, czy wykonany system rzeczywiście spełnia postawione wymagania.
- Powinien to być dokument zrozumiały dla obydwu stron.

Uwaga: Zdarza się, że producenci nie są zainteresowaniu w precyzyjnym formułowaniu wymagań, które pozwoliłoby na rzeczywistą weryfikację powstałego systemu.
Tego rodzaju polityka lub niedbałość może prowadzić do konfliktów.

Zawartość dokumentu SW (1)

Informacje organizacyjne

Streszczenie (maksymalnie 200 słów)
Spis treści
Status dokumentu (autorzy, firmy, daty, podpisy, itd.)
Zmiany w stosunku do wersji poprzedniej

Zasadnicza zawartość dokumentu

1. Wstęp
- 1.1. Cel
- 1.2. Zakres
- 1.3. Definicje, akronimy i skróty
- 1.4. Referencje, odsyłacze do innych dokumentów
- 1.5. Krótki przegląd

2. Ogólny opis
- 2.1. Walory użytkowe i przydatność projektowanego systemu
- 2.2. Ogólne możliwości projektowanego systemu
- 2.3. Ogólne ograniczenia
- 2.4. Charakterystyka użytkowników
- 2.5. Środowisko operacyjne
- 2.6. Założenia i zależności

3. Specyficzne wymagania
- 3.1. Wymagania funkcjonalne (funkcje systemu)
- 3.2. Wymagania niefunkcjonalne (ograniczenia).

Dodatki

Norma
ANSI/IEEE Std 830-1993
„Recommended Practice for Software Requirements Specifications”

Zawartość dokumentu SW (2)

Kolejność i numeracja punktów w przedstawionym spisie treści powinna być zachowana. W przypadku gdy pewien punkt nie zawiera żadnej informacji należy wyraźnie to zasygnalizować przez umieszczenie napisu „Nie dotyczy”.

Dla każdego wymagania powinien być podany powód jego wprowadzenia: cele przedsięwzięcia, których osiągnięcie jest uwarunkowane danym wymaganiem.

Wszelki materiał nie mieszczący się w podanych punktach należy umieszczać w dodatkach.

Często spotykane dodatki

- wymagania sprzętowe
- wymagania dotyczące bazy danych
- model (architektura) systemu
- słownik terminów (użyte terminy, akronimy i skróty z wyjaśnieniem)
- indeks pomocny w wyszukiwaniu w dokumencie konkretnych informacji (dla dokumentów dłuższych niż 80 stron)

Jakość dokumentu wymagań

Styl

Jasność: jednoznaczne sformułowania, zrozumiały dla użytkowników i projektantów. Strukturalna organizacja dokumentu.
Spójność: brak konfliktów w wymaganiach.
Modyfikowalność: wszystkie wymagania są sformułowane w jasnych punktach, które mogą być wyizolowane z kontekstu i zastąpione przez inne.

Ewolucja

Możliwość dodawania nowych wymagań, możliwość ich modyfikacji.

Odpowiedzialność

Określenie kto jest odpowiedzialny za całość dokumentu wymagań.
Określenie kto lub co jest przyczyną sformułowania danego wymagania, istotne w przypadku modyfikacji, np. zmiany zakresu lub kontekstu systemu.

Medium

Dokument papierowy lub elektroniczny.
Staranne kontrolowanie wersji dokumentu.

Słownik

Opis wymagań może zawierać terminy niezrozumiałe dla jednej ze stron. Mogą to być **terminy informatyczne (niezrozumiałe dla klienta)** lub **terminy dotyczące dziedziny zastosowań (niezrozumiałe dla przedstawicieli producenta**).

Wszystkie specyficzne terminy powinny być umieszczone w słowniku, wraz z wyjaśnieniem. Słownik powinien precyzować terminy niejednoznaczne i określać ich znaczenie w kontekście tego dokument (być może nieco węższe).

Termin	**Objaśnienie**	**Synonimy (nie zalecane)**
konto	Nazwana ograniczona przestrzeń dyskowa, gdzie użytkownik może przechowywać swoje dane. Konta są powiązane z określonymi usługami, np. pocztą komputerową oraz z prawami dostępu.	katalog użytkownika
konto bankowe	Sekwencja cyfr oddzielona myślnikami, identyfikująca stan zasobów finansowych oraz operacji dla pojedynczego klienta banku.	konto
klient	Jednostka sprzętowa instalowana w biurach urzędu, poprzez którą następuje interakcja użytkownika końcowego z systemem.	stacja robocza
użytkownik	Osoba używająca systemu dla własnych celów biznesowych nie związanych z obsługą lub administracją systemu.	operator (klient)

Kluczowe czynniki sukcesu

- Zaangażowanie właściwych osób ze strony klienta

- Pełne rozpoznanie wymagań, wykrycie przypadków i dziedzin szczególnych i nietypowych. Błąd popełniany w tej fazie polega na koncentrowaniu się na sytuacjach typowych.

- Sprawdzenie kompletności i spójności wymagań. Przed przystąpieniem do dalszych prac, wymagania powinny być przejrzane pod kątem ich kompletności i spójności.

- Określenie wymagań niefunkcjonalnych w sposób umożliwiający ich weryfikację.

INŻYNIERIA OPROGRAMOWANIA

Dziękuję za uwagę