

Inżynieria wymagań - Wykład 1

Agenda

- Inżynieria wymagań jako część inżynierii oprogramowania.
- Ważne procesy wytwarzania oprogramowania.
- Charakterystyka wymagań.
- Pozyskiwanie wymagań.
- Specyfikowanie wymagań.

Inżynieria wymagań jako wstęp inżynierii oprogramowania

Informatyka jest obejmuje teorie i podstawowe zasady działania oprogramowania.

Inżynieria oprogramowania obejmuje praktyczne problemy związane z tworzeniem oprogramowania w sposób systematyczny.

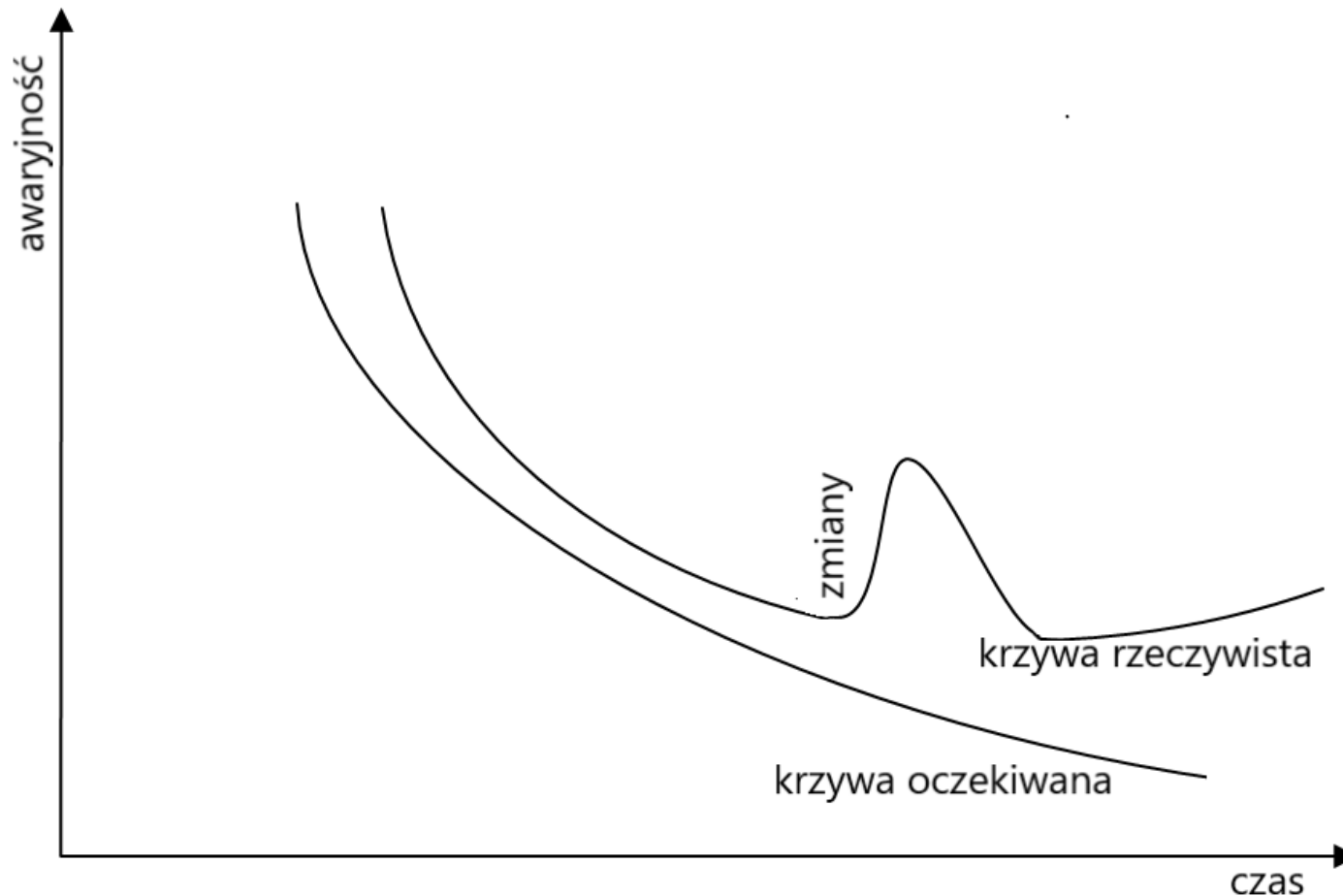
Raport Gartner (July 2023)

Table 1. Worldwide IT Spending Forecast (Millions of U.S. Dollars)

	2022 Spending	2022 Growth (%)	2023 Spending	2023 Growth (%)	2024 Spending	2024 Growth (%)
Data Center Systems	221,223	16.6	217,880	-1.5	235,530	8.1
Devices	766,279	-6.3	700,023	-8.6	748,150	6.9
Software	811,496	10.7	922,745	13.7	1,052,956	14.1
IT Services	1,305,699	7.5	1,420,905	8.8	1,585,373	11.6
Communications Services	1,423,075	-1.9	1,461,662	2.7	1,517,877	3.8
Overall IT	4,527,772	2.8	4,723,215	4.3	5,139,886	8.8

Source: Gartner (July 2023)

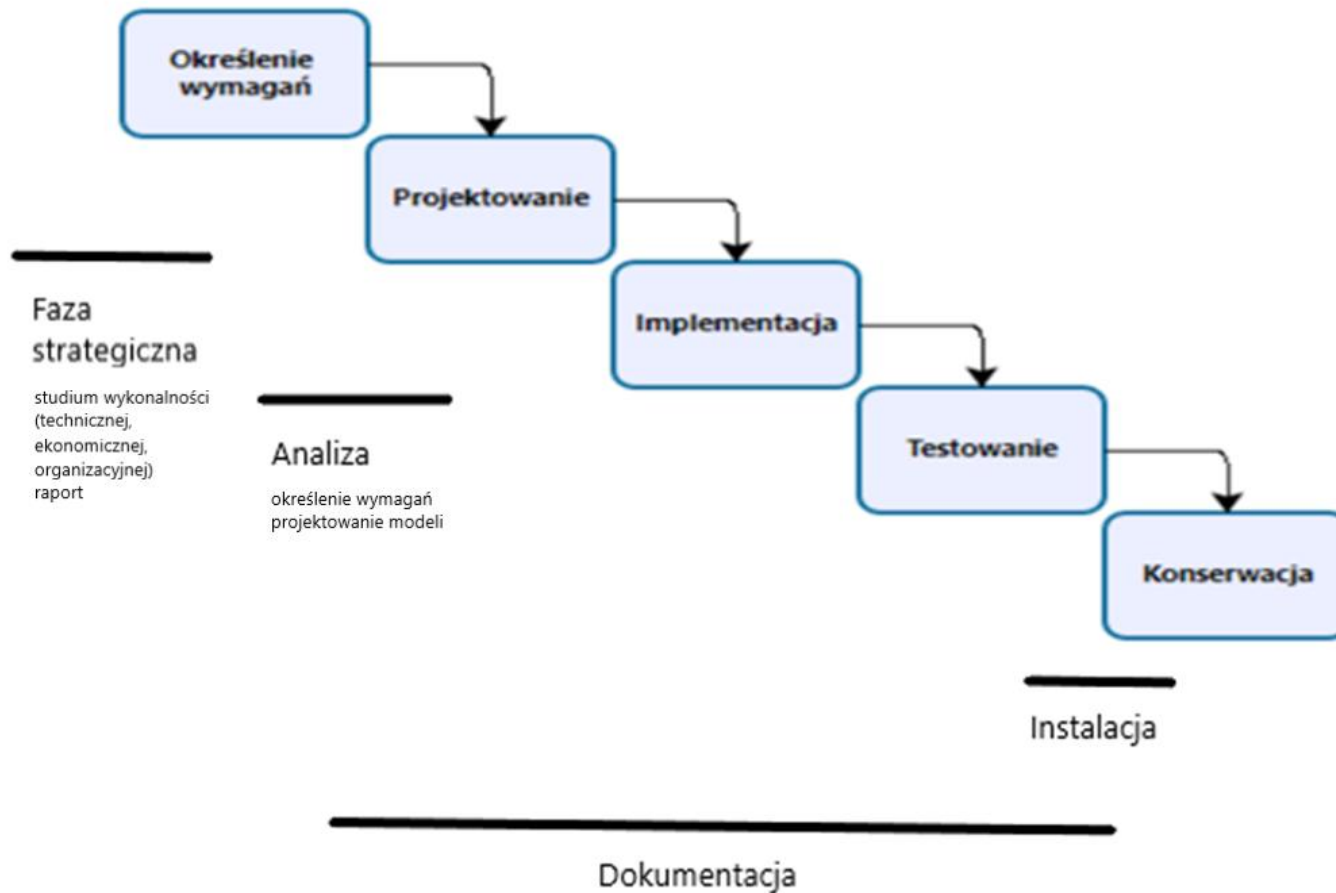
Awaryjność oprogramowania



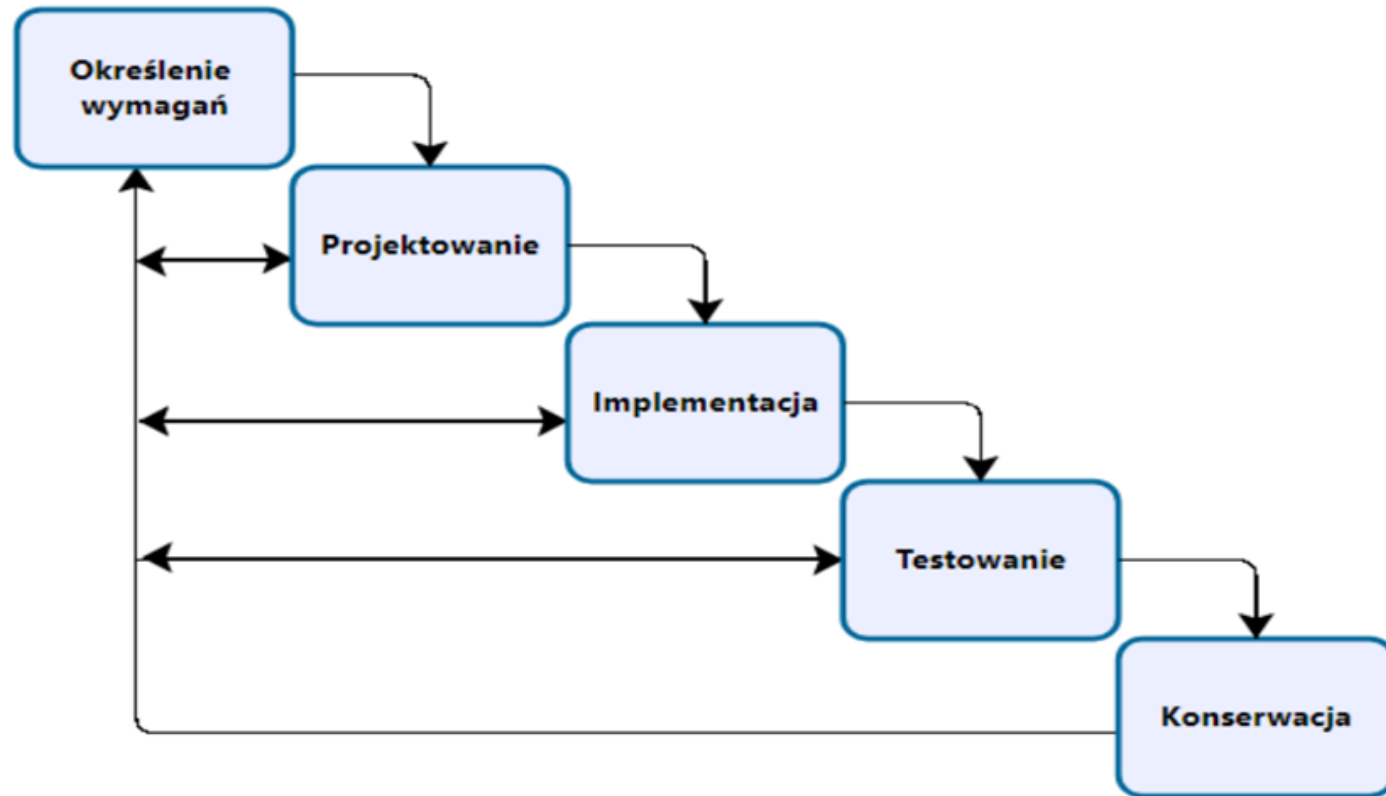
Procesy techniczne i zarządcze

Rzeczywiste procesy inżynierii oprogramowania to sekwencje przeplatających się zespołowych czynności technicznych i zarządczych, których ogólnym celem jest **specyfikacja założeń, zaprojektowanie, implementacja oraz przetestowanie żadanego systemu**. Pierwszym etapem, w którym dokonywane jest agregowanie poszukiwanych założeń projektowanego systemu, zajmuje się inżynieria wymagań.

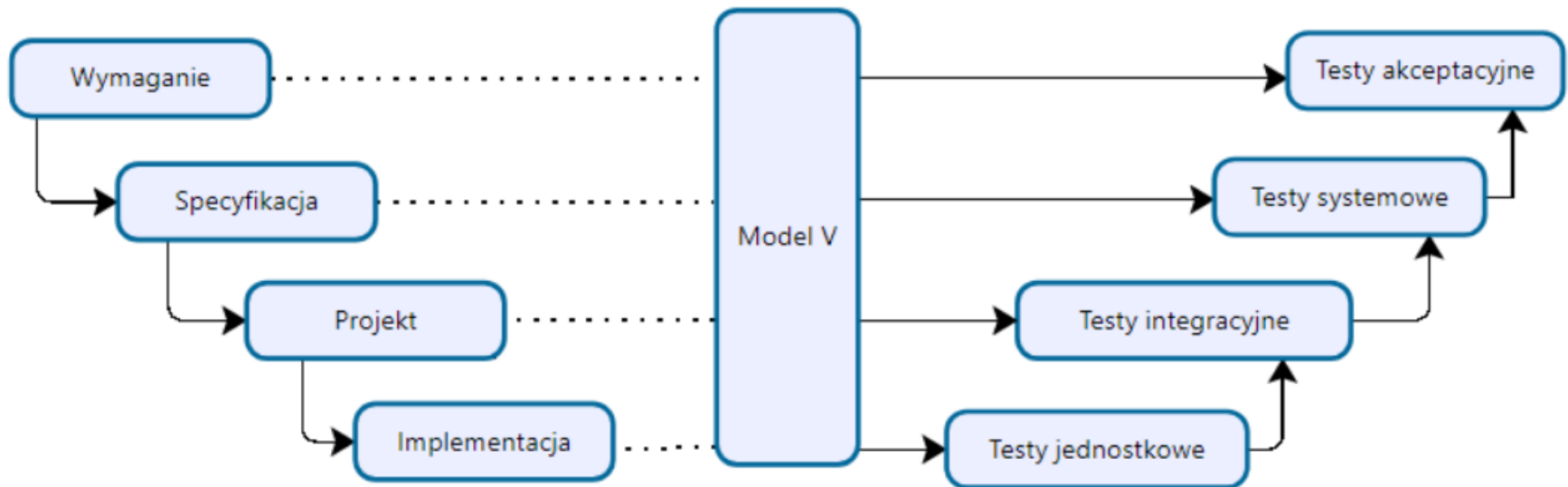
Procesy tworzenia – model kaskadowy



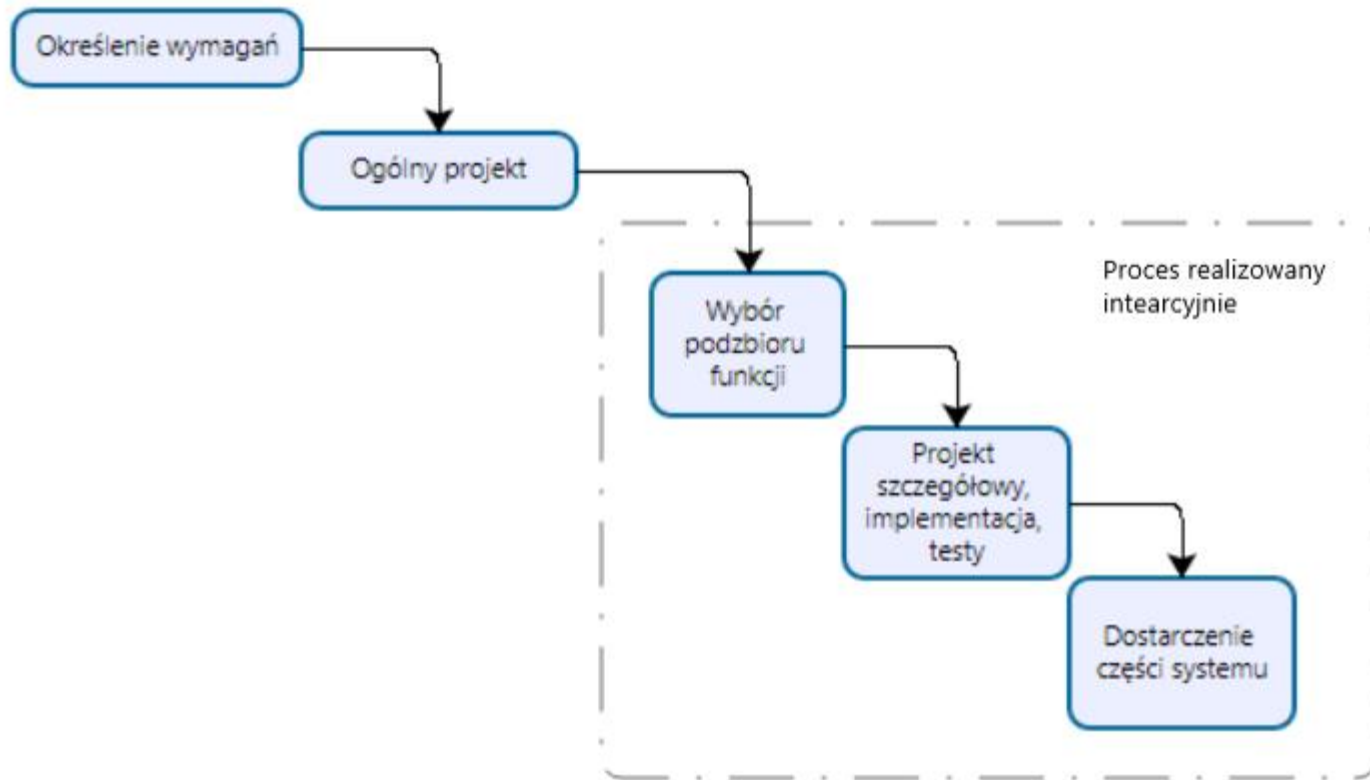
Procesy tworzenia – iteracje modelu kaskadowego



Procesy tworzenia – model V



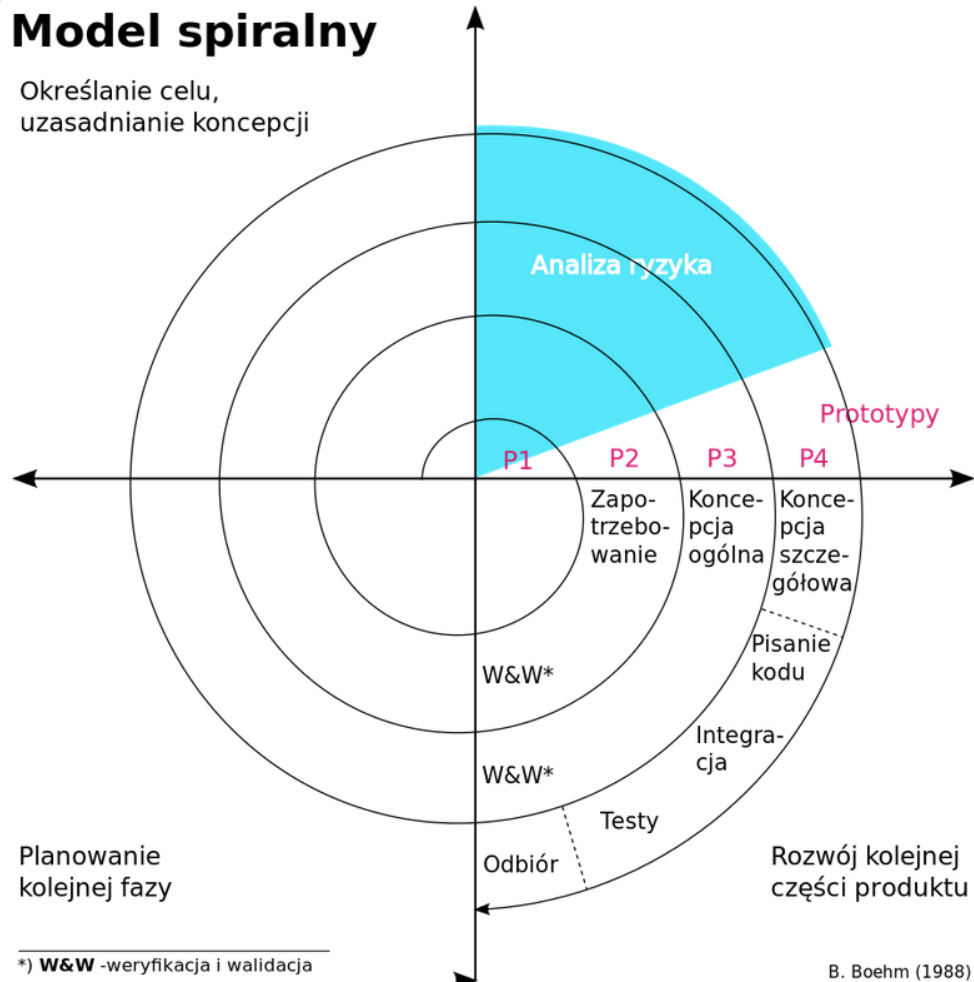
Procesy tworzenia – model przyrostowy



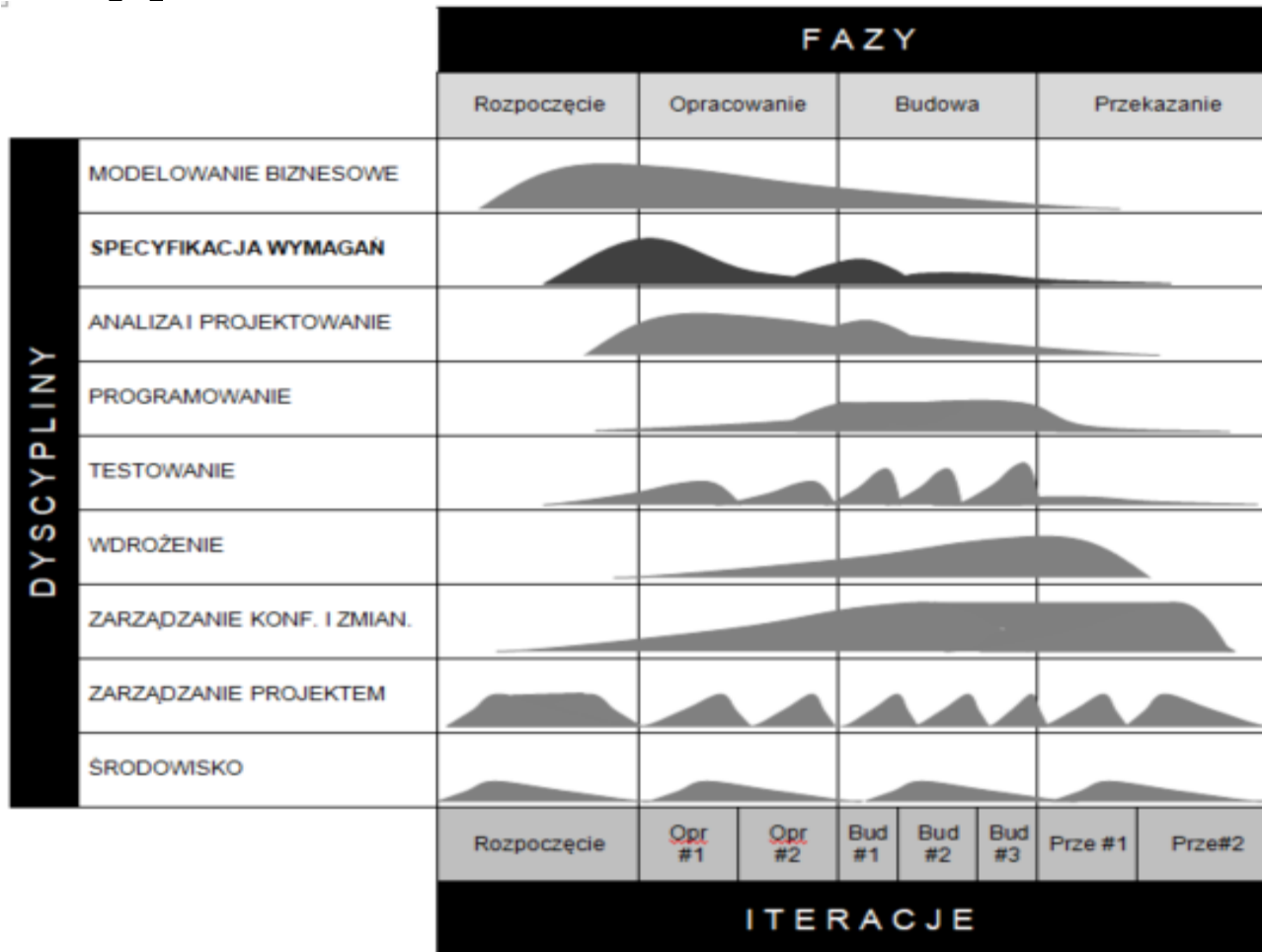
Procesy tworzenia – model spiralny

Model spiralny

Określanie celu,
uzasadnianie koncepcji



Procesy tworzenia – Rational Unified Process jako 9 dyscyplin w 4 fazach



Faza opracowania w RUP

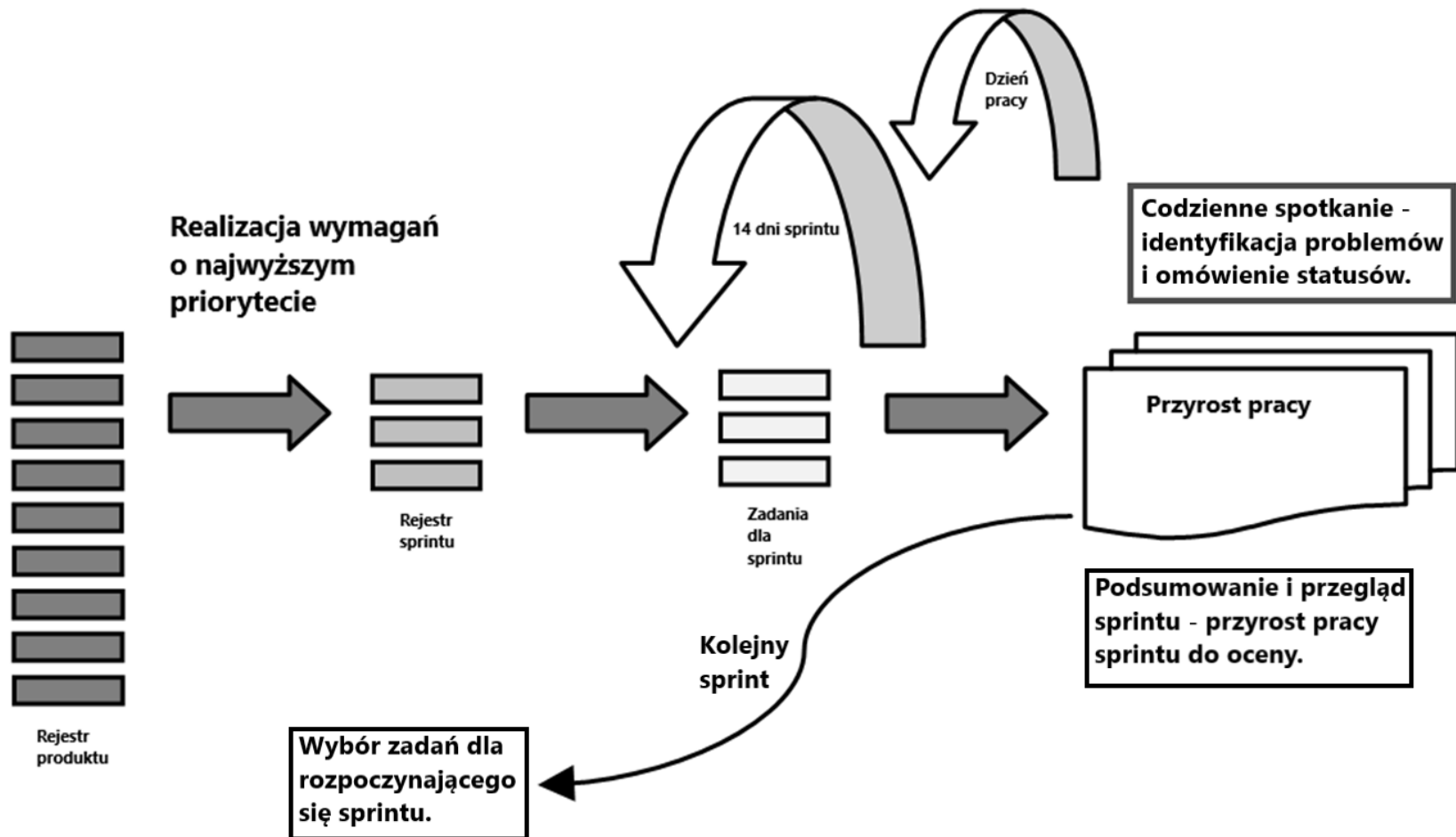
Faza opracowania w RUP jest główną fazą specyfikacji wymagań.

Wynikiem fazy opracowania są m.in.: pełen model przypadków użycia, wymagania niefunkcjonalne, interfejsy systemów, prototypy interfejsu użytkownika.

W przypadku braku zatwierdzenia zakresu można projekt w tej fazie zaniechać.

Uwaga: RUP nie narzuca stosowania całości metodyki – można wybrać i adaptować tylko praktyki niezbędnie potrzebne i możliwe do zastosowania w danym projekcie.

Scrum jako zwinna metodyka prowadzenia projektu - przykład.



<https://agilemanifesto.org/principles.html>

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Business people and developers must work together daily throughout the project.

Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Working software is the primary measure of progress.

Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

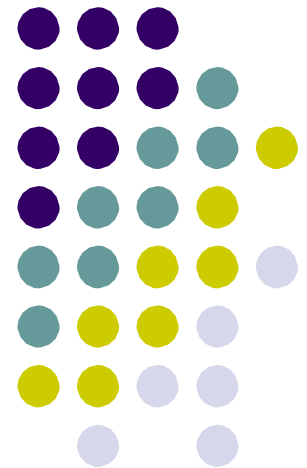
Continuous attention to technical excellence and good design enhances agility.

Simplicity--the art of maximizing the amount of work not done--is essential.

The best architectures, requirements, and designs emerge from self-organizing teams.

At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Charakterystyka wymagań



Definicje

System: spójny, dający się określić zestaw elementów, które poprzez skoordynowane działanie osiągają założony cel.

Interesariusz: osoba lub organizacja, która ma wpływ na wymagania na system lub na którą ten system ma wpływ.

Inżynieria wymagań: zdyscyplinowane podejście do wymagań w celu zrozumienie pragnień i potrzeb interesariuszy oraz zminimalizowanie ryzyka dostarczenia systemu, który nie spełnia tych pragnień i potrzeb.

Wymaganie

Zgodnie z IEEE 610, wymaganie można zdefiniować jako:

- 1) Warunek lub zdolność potrzebną interesariuszowi do rozwiązania danego problemu lub osiągnięcia celu.
- 2) Warunek, który musi zostać spełniony lub zdolność, którą musi posiadać dany system lub moduł systemu, by móc wypełnić dany kontrakt, standard, specyfikację lub inne formalnie nałożone dokumenty.
- 3) Udokumentowaną reprezentację warunku lub zdolności opisanych powyżej w punktach (1) i (2).

Wymaganie (2)

Może być definiowane:

... od abstrakcyjnego opisu usług lub ograniczeń systemu
... do ścisłej specyfikacji funkcjonalności.

Z punktu widzenia celu wymagania:

- może być podstawą oferty kontraktowej – czyli może być otwarte na interpretację.
- może być podstawą samego kontraktu – czyli musi być jednoznaczne i szczegółowe.

Wymaganie (3)

- Jest spójne i niczego nie pomija w swym opisie.
- Właściwie i kompletnie określa realną potrzebę.
- Funkcja i cel wymagania są testowalne.
- Jego cechę lub zachowanie można śledzić aż do końca projektu i jego finalnej realizacji.
- Jedno wymaganie powinno być w jednym zdaniu, np. „system powinien umożliwić zakup biletów komunikacji kolejowej”.

Zasadnicze uwarunkowania (1)

1. Wymagania mają nieść wartość biznesową, a poziom ryzyka biznesowego rozwiązania wpływa w odwrotnej proporcji na poziom zaangażowania finansowego klienta w pozyskanie wymagań. Wartość nakładów na pozyskanie wymagań jest kosztem, ale jego poniesienie jest krokiem w kierunku sukcesu przedsięwzięcia.
2. Inżynier wymagań nie może być tylko notariuszem potrzeb interesariuszy, naiwnie je kolekcjonując, ale musi aktywnie szukać istoty wymagania, ponieważ często to co przekazuje interesariusz nie jest do końca tym, co jest faktycznie potrzebne.

Zasadnicze uwarunkowania (2)

3. Sukces w odkrywaniu wymagań zależy tak od poziomu wiedzy interesariuszy (konieczna jest ich kategoryzacja wg możliwości oddziaływania na system) jak też od wzajemnego zaufania stron, popartego wcześniejszą współpracą, wiedzą dziedzinową, standardami i wartościami (jak w metodach zwinnych, gdzie nie stosuje się pełnego dokumentowania pozyskiwania wymagań).

Zasadnicze uwarunkowania (3)

4. Inżynieria wymagań jest dyscypliną, która dla skutecznego rozwiązywania problemów wymaga systematycznego prowadzenia prac przez techniki dopasowane do realiów konkretnego przypadku. Bagaż historii doświadczeń projektowych jest istotny i buduje pewność siebie, ale nie należy bezrefleksyjnie i bezkrytycznie powtarzać procesów i praktyk z poprzednich udanych działań.

Zasadnicze uwarunkowania (4)

5. Określenie wykonalnych wymagań dla systemu ma służyć skutecznemu rozwiązaniu problemu i przynieść wartość biznesową. Jednak fakt pozyskania wymagań w projektach, które skończyły się źle (np. realizowanych przez startupy) nie może być zawsze postrzegany negatywnie, ponieważ w takich sytuacjach dopiero konkretne techniczne realizacje były w stanie potwierdzić wykonalność albo niewykonalność wymagań.

Zasadnicze uwarunkowania (5)

6. Wymagania mogą się zmieniać w czasie życia systemu na skutek zmiany procesów organizacyjnych i technologii, a także zmian zachowania użytkowników i rozwoju produktów konkurencji. Można zauważyć w ewolucji wymagań zastosowanie cyklu Deminga (Plan-Do-Check-Act). Zmiany jako norma ujęte zostały jasno w manifeście Agile.

Zasadnicze uwarunkowania (6)

7. Projektowanie systemów musi być realizowane w związku z ich otoczeniem. Tam bowiem występują istotne zjawiska, mające wpływ na zachowanie się system. Przykładowe wymaganie użytkownika - np. „drzwi się otwierają tylko wtedy, gdy pojazd jest nieruchomy” – musi być wyrażone wymaganiem systemowym w postaci np.: „drzwi się otwierają tylko wtedy, gdy brak jest sygnału o ruchu z czujnika zewnętrznego”.

Zasadnicze uwarunkowania (7)

8. Konieczne są kryteria akceptacji na potrzeby walidacji. Przykład wymagania: "system powinien umożliwić wyszukanie połączeń kolejowych w czasie krótszym niż 10 sekund". Takie wymaganie jest testowalne w sposób jednoznaczny.

Podstawowe typy wymagań

- Wymagania biznesowe - w dokumencie wizji wskazują korzyści biznesowe z przedsięwzięcia. Odnoszą się do biznesu i są ogólne, ale w metodykach zwinnych sprowadza się je często do wymagań użytkownika. Zawsze mają nieść wartość biznesową.
Przykład: „firma wdroży system ERP dla zwiększenia wydajności swoich procesów”.
- Wymagania użytkownika - zdania języka naturalnego powiązane z diagramami ukazującymi usługi systemu wraz z ograniczeniami biznesowymi, jakie narzucono w kontekście produktu/projektu.
Ukierunkowane są na klientów. Zawierają 4 elementy: **użytkownika, oczekiwany wynik, przedmiot wymagania, ograniczenia biznesowe**.
Przykład: „pracownik sprzedaży powinien mieć informację o nowych produktach jeden dzień przed promocją”.

Wymagania funkcjonalne i нефункционалне

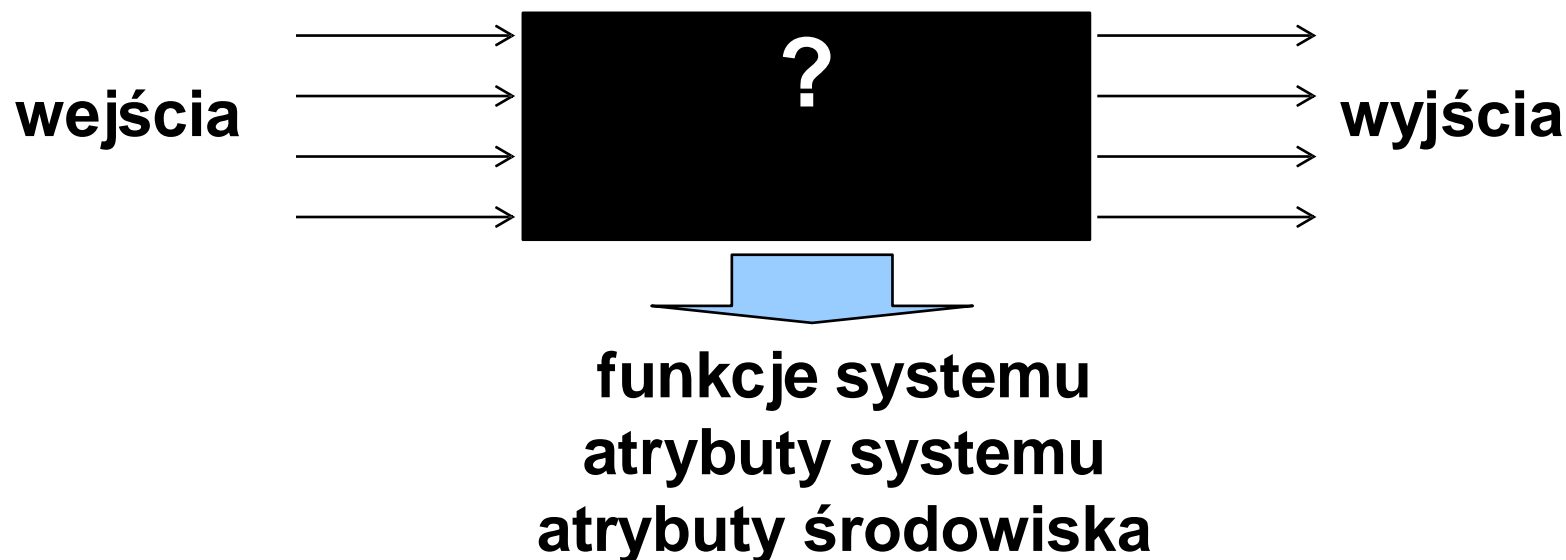
- Wymagania funkcjonalne – zachowanie systemu, czyli jakie akcje ma wykonywać system.
- Wymagania нефункционалне – ograniczenia, które mają wpływ na wykonywane zadania przez system (czy szybko, czy wydajnie itp.). Na jedno wymaganie funkcjonalne powinno przypadać minimum jedno wymaganie нефункционалне (ma pokazać jak działa wymaganie funkcjonalne).

Reguły biznesowe

- Reguły biznesowe nie są wymaganiami.
 - Reguły biznesowe mają dotyczyć bezpośrednio biznesu a nie środowiska aplikacji czy innych bytów wspierających biznes.
 - Reguły biznesowe mają być wyrażane w języku biznesu.
 - Reguły biznesowe są źródłem dla wymagań – zawierają terminy, fakty, reguły; definiują działania biznesowe (np. obliczenia), wskazują na polityki (np. bezpieczeństwa), standardy oraz przepisy prawa.
-
- Dobra reguła biznesowa, która też determinuje wyzwalacz działania:
„jeśli dane kontrahenta na FVAT są błędne, to należy wystawić notę korygującą”.
 - Zła reguła biznesowa (nie jest tu stosowany wyłącznie język biznesu):
„jeśli użytkownik 3 razy z rzędu wprowadzi złe dane logowania, to należy jego konto zablokować”.

Wymagania stawiane oprogramowaniu – charakterystyka wymagań systemowych

Wymagania systemowe są tymi „*rzeczami*”, które należy zdefiniować, aby w pełni opisać co robi system traktowany jako czarna skrzynka.



Ograniczenia projektowe

Ograniczenia nakładane są na projekt systemu lub proces, który używamy do budowy rozwiązania.

- *Produkt musi spełniać normę XYZ 601.*
- *Proces wytwarzania musi być zgodny ze standardem XYZ 1200-34.*

Mają często negatywny wpływ na elastyczność projektantów.

- *Użyj systemu Oracle, programuj w Visual Basic, użyj biblioteki klas XYZ.*

Definicje i specyfikacje

Definicja wymagań użytkownika:

17. Moduł oprogramowania musi udostępniać mechanizm reprezentacji i dostępu do zewnętrznych plików tworzonych przez inne narzędzia po ich zapisaniu.

Wymagania systemowe:

- 17.1 Użytkownik powinien mieć możliwość określenia typu zewnętrznego pliku.
- 17.2 Każdy zewnętrzny plik może mieć powiązane narzędzie, które może być do niego zastosowane.
- 17.3 Każdy typ zewnętrznego pliku powinien być reprezentowany przez specyficzną ikonę w interfejsie użytkownika.
- 17.4 Użytkownik powinien mieć możliwość definiowania ikony powiązanej z typem zewnętrznego pliku.
- 17.5 Efektem zaznaczenia przez użytkownika ikony reprezentującej zewnętrzny plik powinno być zastosowanie narzędzia powiązanego z typem zewnętrznego pliku.

Inny przykład...

Cechy (wymagania użytkownika):

Cecha 63 – system wykrywania błędów dostarczy informacji trendu, które pomogą użytkownikowi ocenić status przedsięwzięcia w danym czasie.

Wymagania systemowe:

WO63.1 Informacje trendu będą dostarczone w raporcie histogramu, gdzie czas oznaczono na osi x, a liczbę defektów na osi y.

WO63.2 Użytkownik może wprowadzić okres wyznaczania trendu w jednostkach dni, tygodni lub miesięcy.

WO63.3 Raport trendu powinien być zgodny z przykładem pokazanym na rysunku 3.1 (s. 56).

Wymagania funkcjonalne

Wymagania funkcjonalne w swym opisie:

- wskazują na zasadniczą funkcję wymagania – np. program obliczy podatek VAT od sprzedaży,
- ukazują merytoryczny sposób działania tej funkcji – np. program obliczy podatek VAT od sprzedaży przez przemnożenie wartości sprzedanego artykułu przez wartość dziesiętną jego stawki podatkowej VAT po zaistniałej transakcji,
- definiują jak reagować na określone dane wejściowe, a nawet czego system nie powinien robić
- posiadają cechy S.M.A.R.T: konkretny (*Specific*), mierzalny (ang. *Measurable*), osiągalny (*Attainable*), istotny (*Relevant*), określony w czasie (*Timebound*).

Przygotowanie do określenia wymagań funkcjonalnych

Określeniu wymagań funkcjonalnych towarzyszą zadania:

- 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 planowanego 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.

Kryteria jakości opisu wymagań funkcjonalnych (1)

- Poprawność – precyzja opisu cech dostarczanych w kontekście wymagania biznesowego z którego się wywodzi.
- Wykonalność – możliwość implementacji wymagania w obrębie znanych ograniczeń systemu lub środowiska.
- Abstrakcyjność – niezależność od sposobu implementacji.
- Konieczność – wyrażenie tego co niezbędne/rzeczywiście potrzebne dla spełnienia zewnętrznych wymagań, standardów lub interfejsów, a czego brak stanowiłby zauważalną lukę.
- Priorytetyzacja – określa wymiennie wartość jaką niesie dla klienta funkcjonalność.

Kryteria jakości opisu wymagań funkcjonalnych (2)

- Jednoznaczność – to samo rozumienie i taka sama, jedna interpretacja przedstawionego wymagania.
- Jednostkowość – opis wymagania nie może zawierać w sobie innych wymagań.
- Weryfikowalność – jasna ocena czy zostało poprawnie zaimplementowane poprzez mierzalne testowanie.
- Zgodność – brak sprzeczności z innymi wymaganiami.
- Zrozumiałość – opis wymagań językiem zrozumiałym dla odbiorcy, pozbawionym specjalistycznych formuł i skrótów myślowych.

Wymagania niefunkcjonalne

Opisują jak dane rozwiązanie wykonuje swoje funkcje.

- **Użyteczność** (*Usability*) – wymagany czas szkolenia, czas wykonania poszczególnych zadań, ergonomia interfejsu, pomoc, dokumentacja użytkownika.
- **Niezawodność** (*Reliability*) – dostępność, średni czas międzyawaryjny (MTBF), średni czas naprawy (MTTR), dokładność, maksymalna liczba błędów.
- **Efektywność** (*Performance*) – czas odpowiedzi, przepustowość, czas odpowiedzi, konsumpcja zasobów, pojemność.
- **Zarządzalność** (*Supportability*) – łatwość modyfikowania, skalowalność, weryfikowalność, kompatybilność, konfiguracja, serwis.
- **Przenaszalność** (*Portability*) – łatwość dostosowania do różnych środowisk lub platform.

Weryfikacja wymagań niefunkcjonalnych

Wymagania niefunkcjonalne powinny być weryfikowalne, tj. powinna istnieć możliwość sprawdzenia czy system je rzeczywiście spełnia. Np. wymaganie “system ma być łatwy w obsłudze” nie jest weryfikowalne.

Cecha	Weryfikowalne miary
Użyteczność	Ergonomia – np. system powinien umożliwiać dokonanie transakcji za pomocą najwyżej 10 operacji na klawiaturze bankomatu Czas niezbędny dla przeszkolenia użytkowników Liczba stron 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
Efektywność/Zasoby	Wymagana pamięć RAM Wymagana pamięć dyskowa
Wydajność	Liczba transakcji obsłużonych w ciągu sekundy Czas odpowiedzi (np. reakcja max 10s od czasu potwierdzenia transakcji) Szybkość odświeżania ekranu
Przenaszalność	Procent kodu zależnego od platformy docelowej Liczba platform docelowych Koszt przeniesienia na nową platformę

FURPS

Popularne podejście do klasyfikacji wymagań, grupuje wymagania funkcjonalne (**F**) i нефunkcjonalne (**URPS**) w następujące typy:

- Funkcjonalność (*Functionality*) – rozumiana jako zestaw wymagań funkcjonalnych + bezpieczeństwo.
- Użyteczność (*Usability*).
- Niezawodność (*Reliability*).
- Efektywność (*Performance*).
- Zarządzalność (*Supportability*).

Atrybuty wymagań (1)

Identyfikator - może być też jednocześnie powiązaniem lub relacją z innymi wymaganiami lub dokumentami projektowymi.

Źródło (zobowiązanie) – dokument lub interesariusz istotny w przypadku problemów z wymaganiami.

Typ – np. biznesowe, użytkownika, funkcjonalne, нефункциональные, produktowe, projektowe.

Status - nowe (proponowane), zatwierdzone, sprzeczne, zaimplementowane, zmodyfikowane, usunięte, wdrożone.

Atrybuty wymagań (2)

Zobowiązanie – stopień obligatoryjności wymagania (notacja MoSCoW – Must have, Should have, Could have, Will not have this time-but will have in the future)

Priorytet – określa ważność/pilność wymagania (obowiązkowy, bardzo pożądanym, pożądanym, opcjonalny)

Krytyczność – wynik oceny ryzyka szkód przy braku spełnienia wymagań (im wyższy poziom krytyczności, tym groźniejsze konsekwencje związane z usterkami w funkcjonalności)

Przykład priorytetyzacji – model Kano

Atrybuty podstawowe – produkt musi je posiadać aby spełnić wymagania klienta.

Atrybuty operacyjne – zwiększają zadowolenie klienta poprzez sposób w jaki produkt dostarcza klientowi możliwości w realizowanych funkcjonalnościach.

Atrybuty emocjonalne – nieprzewidziane przez klienta, które odkryte przez dostawcę dostarczają przewagi konkurencyjnej. Atrybut emocjonalny z czasem ulega rozmyciu.

Wymagania a inżynieria wymagań

Wymagania – opis usług i ograniczeń systemu generowanych w procesie inżynierii wymagań.

Inżynieria wymagań – proces pozyskiwania, analizowania, dokumentowania oraz weryfikacji wymagań

.... czyli zarządzania wymaganiami.

Dlaczego inżynieria wymagań? (1)

Niezbędna umiejętność – pozyskiwanie wymagań
od użytkowników i klientów.

Zamiana celów klienta na konkretne wymagania
zapewniające osiągnięcie tych celów.

Klient rzadko wie, jakie wymagania zapewnią osiągnięcie
jego celów.

Jest to tak naprawdę **proces konstrukcji zbioru
wymagań zgodnie z postawionymi celami.**

Dlaczego inżynieria wymagań? (2)

Organizacja i dokumentowanie wymagań

Setki, jeżeli nie tysiąca wymagań jest prawdopodobnie związanych z systemem

Według klienta prawdopodobnie wszystkie z nich są najważniejsze w 100%.

Większość z nas nie może pamiętać więcej niż kilkadziesiąt informacji jednocześnie.

Dlaczego inżynieria wymagań? (3)

Śledzenie, kontrola dostępu i weryfikacja wymagań:

- Którzy członkowie zespołu są odpowiedzialni za wymaganie nr 278, a którzy mogą je zmodyfikować lub usunąć?
- Jeżeli wymaganie nr 278 będzie zmodyfikowane, jaki to będzie miało wpływ na inne wymagania?
- Kiedy możemy być pewni, że ktoś napisał kod w systemie, spełniający wymaganie nr 278 i które testy z ogólnego zestawu testów są przeznaczone do sprawdzenia, że wymaganie rzeczywiście zostało spełnione?

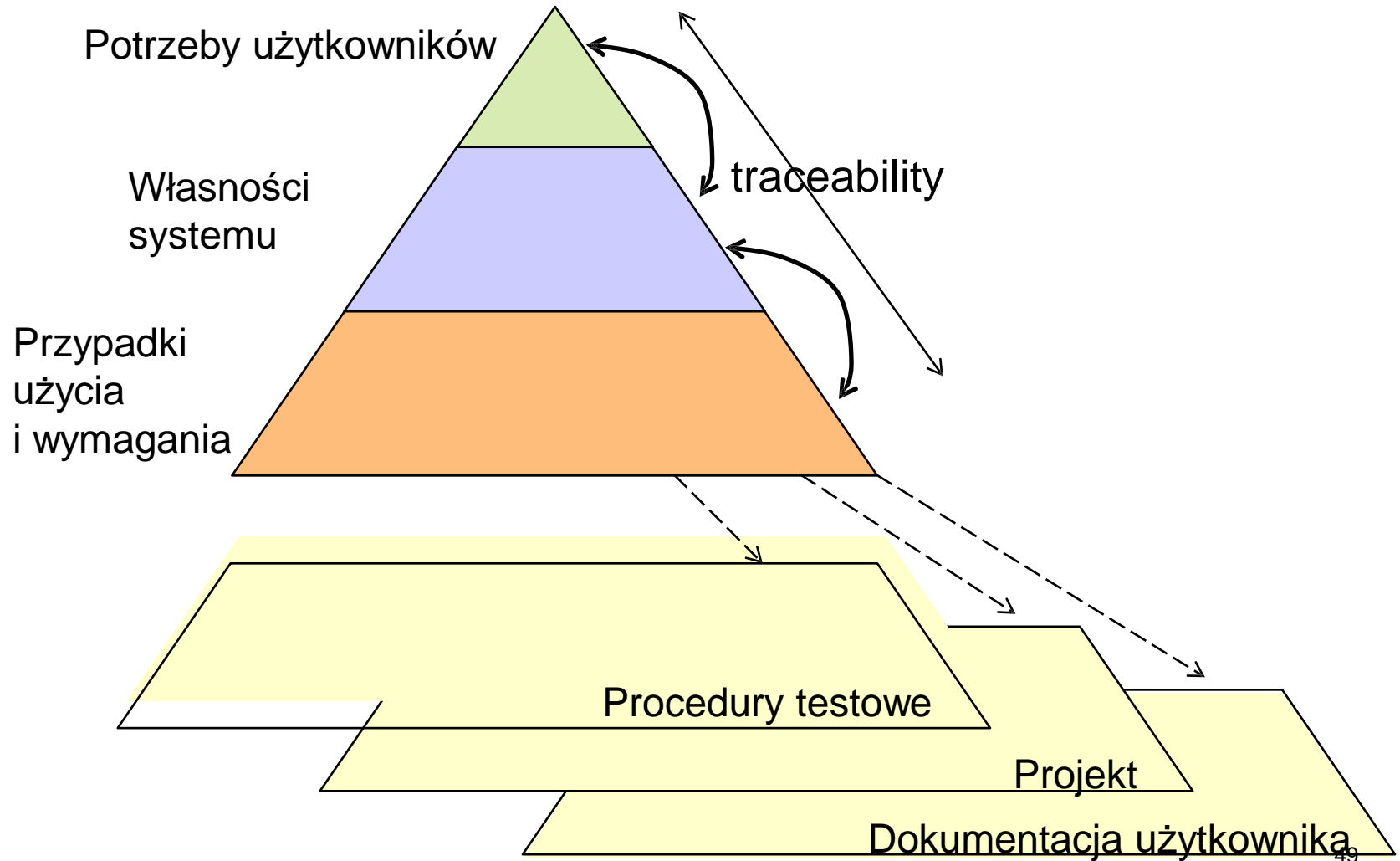
Zarządzanie wymaganiami

Zarządzanie wymaganiami dotyczy procesu translacji potrzeb klientów w zbiór kluczowych cech i własności systemu.

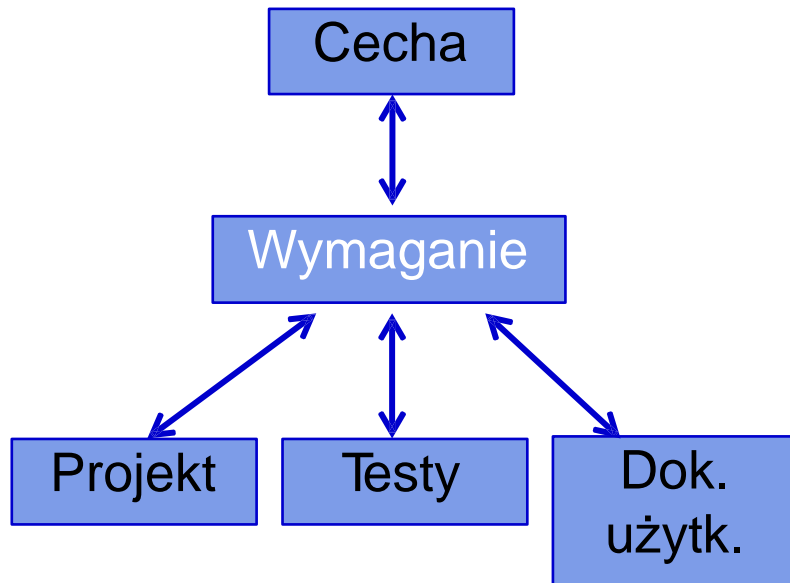
Następnie ten zbiór jest przekształcany w specyfikację funkcjonalnych i niefunkcjonalnych wymagań.

Specyfikacja jest następnie przekształcana w projekt, procedury testowe i dokumentację użytkownika.

Zarządzanie wymaganiami i traceability



Traceability



- Oszacowanie wpływu zmiany w wymaganiach na projekt.
- Oszacowanie wpływu jaki nowe wymagania będzie miał „zawalony” test (jeżeli system nie przeszedł testu, wymagania mogą nie być spełnione)
- Zarządzanie ramami projektu
- Weryfikacja czy wszystkie wymagania zostały spełnione przez implementację systemu
- Zweryfikowanie czy system robi tylko to co miał robić.
- Zarządzanie zmianami.

Traceability (c.d.)

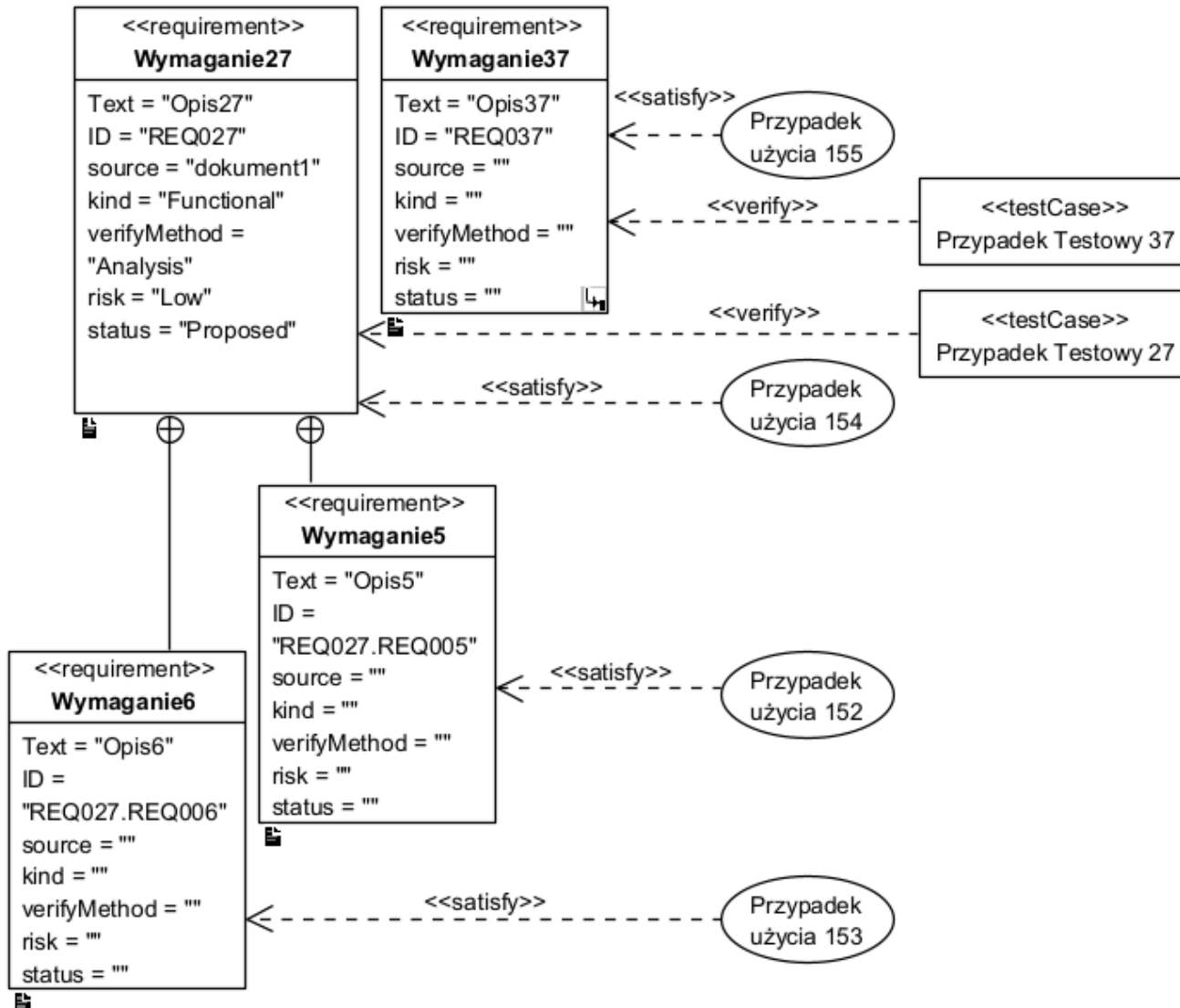
- Dekompozycja wymagań złożonych.
- Zbalansowany poziom szczegółowości wymagań.
- Śledzenie niezbędnych relacji między wymaganiami po dekompozycji.
- Relacje poprzez unikalne identyfikatory wymagań o strukturze hierarchicznej.
- Analiza wpływu.
- Demaskowanie luk w wymaganiach.
- Wyszukiwanie sprzeczności.

1. Funkcja 1

1.1 podfunkcja 1

1.2 podfunkcja 2

Przykładowe śledzenie wymagań (1)



Przykładowa śledzenia wymagań (2)

Wymaganie biznesowe

Bank XYZ poprawi wydajność rozpatrywania wniosków kredytowych przez stworzenie dedykowanej temu aplikacji.

Wymagania użytkownika

1. Wniosek kredytowy.

System musi obsługiwać proces składania wniosku kredytowego przez osoby nowe i nie korzystające jeszcze z usług banku oraz użytkowników będących już klientami banku. Obie grupy powinny mieć szybką ścieżkę dotarcia do informacji kredytowej i używać mechanizmu złożenia wniosku. Pracownik banku powinien być skutecznie powiadomiony o zmianach lub nowych wymaganiach kredytowych przez system i całościowo rozpatrywać wniosek klienta (akceptować albo odrzucić).

Wymagania funkcjonalne

1. Wniosek kredytowy

1.1 Nowi użytkownicy

Nowy użytkownik powinien mieć na stronie banku natychmiastowy dostęp do formularza rejestracji po użyciu widocznej możliwości "Wniosek o kredyt".

Na formularzu rejestracji nowy użytkownik powinien podać nazwisko, imię, adres, data urodzenia, numer dowodu osobistego, telefon, adres email obowiązkowo.

Nowy użytkownik może zrezygnować z w/w rejestracji i będzie miał dostęp tylko do informacji opisowych o zasadach udzielenia kredytu jako gość.

Po wprowadzaniu koniecznych dla procesu rejestracji danych nowy użytkownik ustali hasło i otrzyma ID dostępu.

1.2 Istniejący użytkownicy

Istniejący użytkownicy banku powinni złożyć wniosek kredytowy bez wypełniania danych osobowych. Wypełnianie wniosku kredytowego we właściwym formularzu użytkownik może przerwać i powrócić do niego w nowej sesji bez straty poprzednich zapisów. Wszyscy, którzy złożyli wniosek kredytowy mają możliwość zarządzania nim.

Użytkownik powinien mieć informacje o tym, na jakim etapie jest rozpatrywanie jego wniosku.

W momencie zakolejkowania do rozpatrywania wniosek kredytowy jest oznaczony statusem "Przesłany do rozpatrzenia".

Po rozpatrzeniu wniosek kredytowy zmienia status na "Rozpatrzony i zaakceptowany" albo "Rozpatrzony i odrzucony" zgodnie z decyzją pracownika banku.

Przykładowa macierz śledzenia wymagań (2 c.d.)

Testowanie wymagań	
ID	Test akceptacyjny
TS_001	Walidacja cech procesu "Wniosek kredytowy" dla nowego użytkownika
TS_002	Walidacja cech procesu "Wniosek kredytowy" dla już istniejącego użytkownika
TS_003	Dla nowego użytkownika w procesie "Wniosek kredytowy" sprawdź opcję gościa i dostępne dla niego elementy procesu
TS_004	Dla nowego użytkownika w procesie "Wniosek kredytowy" sprawdź opcję rejestracji i dostępne dla niego elementy procesu
TS_005	Zaloguj się jako zarejestrowany użytkownik z pozytywnie rozpatrzonym wnioskiem kredytowym i sprawdź dostępne informacje
TS_006	Sprawdź wniosek o statusie "Przesłany do rozpatrzenia"
TS_007	Sprawdź wniosek o statusie "Rozpatrzony i zaakceptowany"
TS_008	Sprawdź wniosek o statusie "Rozpatrzony i odrzucony"

Przykładowa macierz śledzenia wymagań					
Wymagania użytkownika	Wymagania funkcjonalne	Test akceptacyjny	Przypadek testowy	Status	Errors
1. Wniosek kredytowy	1.1 Nowy użytkownik	TS_001 - Walidacja cech procesu "Wniosek kredytowy" dla nowego użytkownika	TC_newUser_01	Passed	
			TC_newUser_02	Passed	
			TC_newUser_03	Not Passed	Error_01, Error_02
			TC_newUser_04	Passed	
		TS_002 - Walidacja cech procesu "Wniosek kredytowy" dla już istniejącego użytkownika	TC_newUser_05	Not Passed	Error_01
			TC_newUser_06	Not Passed	Error_03
			TC_newUser_07	Passed	
			TC_newUser_08	Passed	
	1.2 Istniejący użytkownik	TS_003 - Dla nowego użytkownika w procesie "Wniosek kredytowy" sprawdź opcję gościa i dostępne dla niej elementy procesu	TC_newUser_09	Passed	
		TS_004 - Dla nowego użytkownika w procesie "Wniosek kredytowy" sprawdź opcję rejestracji i dostępne dla niej elementy procesu			
		TS_005 - Zaloguj się jako zarejestrowany użytkownik z pozytywnie rozpatrzonym wnioskiem kredytowym i sprawdź dostępne informacje	TC_existUser_01	Passed	
			TC_existUser_02	Passed	
			TC_existUser_03	Passed	
			TC_existUser_04	Passed	
		TS_006 - Sprawdź wniosek o statusie "Przesłany do rozpatrzenia"			
		TS_007 - Sprawdź wniosek o statusie "Rozpatrzony i zaakceptowany"			
		TS_008 - Sprawdź wniosek o statusie "Rozpatrzony i odrzucony"			

Weryfikacja i walidacja wymagań

- Weryfikacja – odpowiada zagadnieniu czy tworzymy produkt zgodnie ze specyfikacją („czy właściwie tworzymy produkt?”).
- Walidacja - odpowiada zagadnieniu czy tworzymy prawidłowy produkt („czy tworzymy właściwy produkt?”). Po analizie dokumentacji walidacji rozważane są sugestie odnośnie zmian w wymaganiach.
- Weryfikacja i walidacja wymagań z założenia powinny być ciągłym procesem w cyklu procesu wytwórczego.

Weryfikacja wymagań

Weryfikacja dostarcza wyników w punktach kontrolnych - wybrane produkty pośrednie lub rezultaty są weryfikowane w celu potwierdzenia spełnienia przez nie wymagań.

Głównym celem weryfikacji jest zapewnienie kompletnych i spójnych ocen czy wymagania systemowe są spełnione – czy produkt został wykonany zgodnie z wymaganiami.

Techniki stosowane w weryfikacji wymagań to:

- eksperckie przeglądy wymagań
- demonstracje prototypów lub symulacje
- testowanie (czarnoskrzynkowe, białoskrzynkowe, regresji)

Walidacja wymagań

Czy wymaganie spełnia oczekiwania interesariuszy?
Czy wymaganie faktycznie rozwiązuje problem?

Walidacja ma charakter konfrontacji z wyobrażeniami:

- wskazuje, czy wymagania nie są wzajemnie sprzeczne,
- wykrywa błędy i niejednoznaczności tak wcześnie jak można, aby zminimalizować wpływ błędu na projekt.

Formy walidacji: opinia ekspercka, inspekcja wymagań, przegląd koleżeński, czytanie wymagań w konkretnym kontekście, przez prototypowanie, z użyciem list kontrolnych, przez budowę i wykonanie akceptacyjnych scenariuszy testowych.

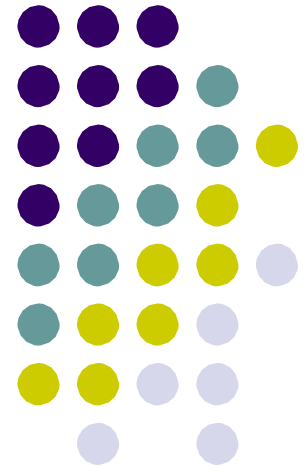
Wybrane formy walidacji - Inspekcja

Fazy działań metody **Inspekcja**:

- planowanie - pokazanie celów, formy wyników, wskazanie ról inspektorów przewidzianych do inspekcji,
- przeglądanie – referowanie inspektorom wymagań przez analityka,
- detekcja błędów – wnikliwa ocena dokumentacji przez inspektorów wraz z dokumentowaniem odkrytych błędów,
- konsolidowanie błędów – zakres od analizy i krytycznego przetwarzanie ujawnionych błędów (m.in. wykrywanie niepoprawnie zakwalifikowanych jako błędne, co jest możliwe w natłoku często setek wymagań) po dokumentowanie i wskazanie działań naprawczych.

Pozyskiwanie wymagań

W czym tkwi problem i jak sobie z nim poradzić?



Inżynieria wymagań – proces odkrywania, dokumentowania i zarządzania wymaganiami dla systemu

- 4 fazy opracowywania wymagań:
 - identyfikacja (pozyskiwanie)
 - analiza
 - specyfikowanie
 - walidacja
- 3 fazy zarządzania wymaganiami:
 - śledzenie (traceability)
 - gwarantowanie jakości
 - zarządzanie zmianami

Alegoria „problem kamienia”



Efekt...




Modyfikacja...




Modyfikacja...





Ci programiści po prostu tego nie rozumieją...

Sam nie wie, czego chce, ciągle zmienia zdanie...



W końcu może się okazać, że klient cały czas myślał o małym szarym kawałku granitu

Może nie był pewien, czego chce, poza tym, że był to mały szary kawałek granitu?

A może jego wymagania uległy zmianie pomiędzy dostawą pierwszego (dużego)

Źródła wymagań

- **Interesariusze**
- **Dokumenty** związane z zagadnieniem:
 - dokument wizji projektu
 - regulaminy
 - akty prawne i rozporządzenia
 - reguły biznesowe
 - oferta przetargowa
- **Systemy** będące w użyciu – także **konkurencyjne**.

Barierzy pozyskiwania wymagań

- Syndrom „tak, ale”

„Tak, ale hmmm, teraz kiedy go już widzę, czy będzie można...? Czy nie lepiej byłoby, gdyby...? Przecież jeżeli..., to trudno będzie...”

- Syndrom „nieodkrytych ruin”

Nigdy nie wiadomo, które z wymagań zostało „nieodkryte”

- Syndrom „wiedzy kompletnej”

Strony zainteresowane zakładają posiadanie pełni wiedzy („prawidłowej”) o rozwiązaniu.

- Syndrom „użytkownik i programista”

Użytkownicy, nie wiedzą, czego chcą, lub wiedzą, co chcą, ale nie mogą tego wyrazić.

Użytkownicy uważają, że wiedzą, czego chcą, dopóki programiści nie dadzą im tego, o czym mówili, że chcą.

Analitycy uważają, że rozumieją problemy użytkownika lepiej niż on sam.

7 nawyków skutecznego działania

(Steven R. Covey)

- Zależność - obarczanie innych zadaniem opieki nad sobą.
- Niezależność - wiara w siebie i we własne siły.
- Współzależność – synergia działań ku wzajemnej korzyści pozwala osiągnąć więcej.

Cechy pomagające pogłębiać nawiązywane relacje: otwartość, aktywność, umiejętność perswazji, dar przekonywania, ale też zawsze wiedza, jakość wykonywania prac, wysokie standardy moralne.

Proces pozyskiwania wymagań

- Tradycyjne podejście do opracowywania specyfikacji wymagań kładzie nacisk na sukces w pierwszym podejściu, co z podstawowych przyczyn ponosi porażkę.
- W nowoczesnych technikach produkcji oprogramowania powszechna jest świadomość, że specyfikacja wymagań będzie długo w procesie jej opracowywania dyskutowana, ponieważ należy starać się odnaleźć najlepsze sposoby obniżenia ryzyka w miarę postępu prac.
- Nie należy traktować w kategoriach porażki sytuacji, gdy klient odrzuca zaprezentowaną grupę wymagań – każde odrzucenie to nowe informacje o tym co istotne dla klienta.

Potrzeby vs cechy

- W procesie odkrywania celu działania często zamiast określenia „co jest potrzebne” (dla określenia celu biznesowego) często podawana jest cecha „rozwiązania”.
- Dla zrozumienia potrzeby kryjącej się za podaną cechą, należy dokonać analizy problemu, czy jest to rzeczywista potrzeba, np. ponieważ faktycznie chcemy wiedzieć, czy w systemie usterki pojawiają się szybciej niż są usuwane, to podana cecha „system śledzenia usterek powinien zapewnić raport trendu statusu” jest wyrażona poprawnie.
- Tak potrzeby jak cechy muszą spełniać zasady S.M.A.R.T..
- Należy odnotowywać każde żądanie (potrzebę czy cechę) i przeprowadzić je przez standardowy proces akceptacji.

Identyfikacja interesariuszy

- Wyszukiwanie przez analizę struktury organizacyjnej oraz procesów biznesowych z ich właścicielami.
- Grupowanie interesariuszy i wyłanianie ich reprezentantów.
- Określenie relacji i identyfikacja potencjalnych konfliktów.
- Znajdowanie możliwości realizacji strategii win-win w analizie konfliktów.
- Określenie perspektyw interesariuszy i znajdowanie takich, którzy minimalizują ryzyka przez ich większy udział w czynnościach projektowych.

Przykładowe techniki pozyskiwania wymagań

- Śledzenie (*Shadowing*)
- Wywiady (*Interviewing*)
- Warsztaty pozyskiwania wymagań (*Focus groups*)
- Przeglądy i ankiety (*Surveys*)
- Instrukowanie przez użytkowników (*User instructions*)
- Prototypowanie (*Prototyping*)

Kryteria wyboru techniki pozyskiwania wymagań

- Rodzaj tworzonego systemu.
- Typy wiedzy – różne metody pozyskują różne typy wiedzy (zachowanie, procesy, dane).
- Wewnętrzne filtrowanie wiedzy – ze względu na różny sposób wydobywania z pamięci różnych typów wiedzy.
- Kontekst pozyskiwania w organizacji – środowisko może mieć istotny wpływ na techniki pozyskiwania wymagań.

***Shadowing* - śledzenie**

Polega na obserwowaniu użytkownika podczas wykonywania przez niego codziennych zadań w rzeczywistym środowisku.

Rodzaje – pasywne i aktywne

- **Zalety**

- Informacja z pierwszej ręki w odpowiednim kontekście
- Łatwiejsze zrozumienie celu określonego zadania
- Możliwość zebrania nie tylko informacji, ale i innych elementów środowiska (np. dokumenty, zrzuty ekranowe istniejącego rozwiązania)
- Zebranie informacji na temat istniejącego rozwiązania oraz tego czy i w jaki sposób jest ono frustrujące dla użytkowników

- **Wady**

- Nieodpowiednie dla zadań wykonywanych sporadycznie, zadań związanych

Interviewing - wywiady

Spotkanie członka zespołu projektowego z użytkownikiem lub klientem.

Zalety

- Można uzyskać dużo informacji o problemach i ograniczeniach obecnej sytuacji, która ma być zmieniona przez nowy system.
- Daje możliwość uzyskania wielu informacji, które niekoniecznie można uzyskać przy wykorzystaniu techniki śledzenia.

Wady

- Zależna od umiejętności i zaangażowania obu stron.

Focus groups - warsztaty pozyskiwania wymagań

Sesja w której wymagania ustala się w większej grupie (np. burza mózgów, odgrywanie ról, itp.)

- **Zalety:**

Pozwala na uzyskanie szczegółowych informacji o szerszym kontekście aktywności biznesowych. Brak informacji u jednego z uczestników może być uzupełniona przez pozostałych.

- **Wady:**

Jest kosztowne ze względu na oderwanie osób od pracy.

Wymaga od prowadzącego umiejętności prowadzenia dyskusji.

***Surveys* – przeglądy, pomiary (1)**

Zbiór pytań utworzony w celu zebrania określonych informacji.

Kwestionariusze

- Wymagane przy okazji np. rejestracji.
- Informacje zwrotne.
- Arkusze badania poziomu zadowolenia z produktu.

Zalety

- Anonimowe wyrażanie swojego zdania.
- Odpowiedzi tabelaryzowane i łatwe w analizie.

Wady

- Pracochłonne.
- Wymagana profesjonalna wiedza w celu tworzenia i analizy.

***Surveys* – przeglądy, pomiary (2)**

Może być pomocne w pozyskaniu następujących Informacji:

- Struktura organizacyjna, polityka działania, praktyki stosowane w pracy.
- Frustracje związane z wykonywaną pracą.
- Wymagania specjalne związane z oprogramowaniem, sprzętem.
- Efektywność programu szkoleniowego.
- Stopień zadowolenia z produktu.

***User instructions* – instruowanie przez użytkowników (1)**

W technice tej użytkownicy instruują przeprowadzającego badanie w sposobie w jaki wykonują określone zadania.

Zalety

- Widzenie procesu z perspektywy użytkownika.
- Zebranie informacji na temat doświadczenia pojedynczych osób.

Wady

- Może być czasochłonne.
- Może być frustrująca dla badacza, jeżeli użytkownik nie jest przyzwyczajony do uczenia kogoś.
- Różne osoby mogą wykonywać to samo zadanie w różny sposób.

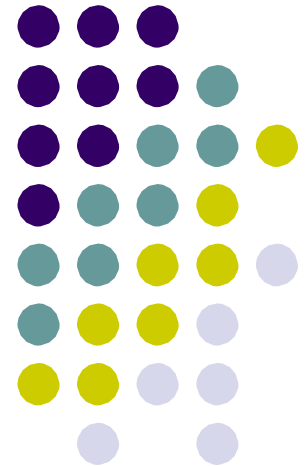
***User instructions* – instruowanie**

Może być pomocne w pozyskaniu następujących informacji:

- projekt interfejsu użytkownika,
- wymagania związane z procesem szkoleniowym,
- kryteria wydajnościowe systemu,
- wpływ środowiska na wykonywane zadania.

Specyfikacja wymagań

Czy wymaganie nr. 31.2 jest sprzeczne z wymaganiem nr. 34.3, a wymaganie 22.1 jest powiązane z wymaganiem 14.2?



Metody specyfikacji wymagań (1)

- Język naturalny - najczęściej stosowany. Wady:
 - *niejednoznaczność* powodująca różne rozumienie tego samego tekstu;
 - *elastyczność*, powodująca wyrazić te same treści na wiele sposobów. Utrudnia to wykrycie powiązanych wymagań i powoduje trudności w wykryciu sprzeczności.
- Formalizm 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.

Metody specyfikacji wymagań (2)

- Tablice, formularze. Wyszpecyfikowanie wymagań w postaci (zwykle dwuwymiarowych) tablic, kojarzących różne aspekty.
- Diagramy blokowe: forma graficzna ukazująca cykl obróbki.
- Diagramy kontekstowe: ukazują system jako jeden blok oraz jego powiązania z otoczeniem, wejściem i wyjściem.
- Model przypadków użycia: przedstawia w poglądowo aktorów i funkcje systemu. Uważa się go za dobry sposób specyfikacji wymagań funkcjonalnych.

Wymagania w metodykach zwinnych

Historyjka użytkownika (*user story*) powinna być INVEST:

- niepowiązana (*Independent*) z innymi, operująca na „jej” funkcjonalności,
- negocjowalna (*Negotiable*) w zakresie operacji na „jej” funkcjonalności,
- wartościowa (*Valuable*) w zakresie „po co?”, wyrażającym sens funkcjonalności,
- estymowalna (*Estimable*) w ocenie kosztu wykonania „jej” funkcjonalności,
- krótka (*Sized Appropriately*) na tyle aby implementacja odbyła się w jednym sprincie,
- weryfikowalna (*Testable*) w procesie testowania.

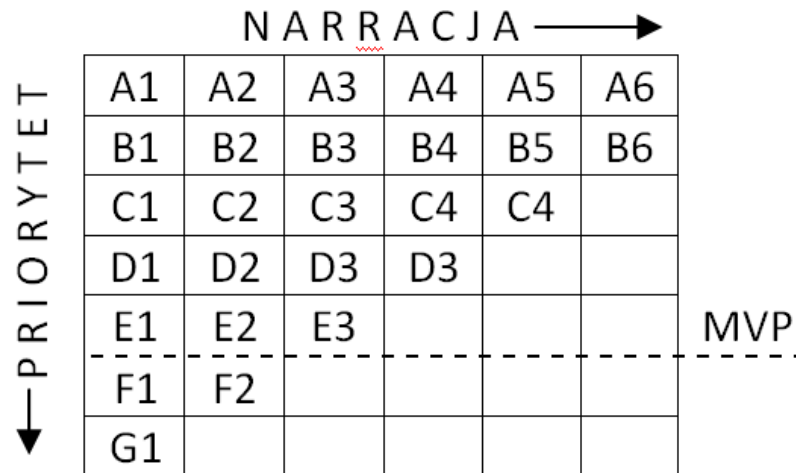
Opis wymagania typu user story

- Struktura zdania opisu powinna być w postaci:
„jako [tu: rola użytkownika]
chcę [tu: możliwość, czynność],
aby [tu: powód lub cel]”.

Przykład: „jako uczeń chcę przeglądać ofertę kursów, aby zapisać się na wybrany zestaw”.

Technika story mapping

- Opisy wymagań typu user stories powinny układać się w spriorytetyzowane warstwowo, logiczne ciągi opowieści o procesach – to tzw. **story mapping** – ułożone w dwóch wymiarach, gdzie najbardziej wartościowe procesy są w warstwach powyżej linii MVP (źródło: <https://4ba.pl/story-mapping-nieco-szersze-spojrzenie>).



Scenariusze w modelu przypadków użycia.

- Przypadek użycia (*use case*) odpowiada konkretnemu wymaganiu funkcjonalnemu.
- Przypadek użycia zawiera zestaw scenariusza głównego oraz alternatywnych powiązane ze sobą wspólnym celem dla użytkownika.
- Scenariusz to ciąg kroków (od 5 do 9) opisujących interakcje (np. pomiędzy użytkownikiem a systemem), opis jednej, konkretnej sytuacji.
- Scenariusz może opisywać alternatywne zdarzenie (np. zakończone niepowodzeniem) jako oddzielny scenariusz.
- W scenariuszu należy odejść od stosowania pojęć żargonu technicznego.

Dokumenty/Narzędzia

- Rekomendowana przez IEEE struktura dokumentu wymagań (*IEEE recommended practice for software requirements specifications IEEE Std 830-1998*)
- Rational Requisite Pro www-306.ibm.com/software/rational — narzędzie do zarządzania wymaganiami.

Dokument Specyfikacji Wymagań Oprogramowania (SWO)

- Wymagania powinny być zebrane w dokumencie – specyfikacji wymagań oprogramowania.
- Dokument ten powinien być podstawą do szczegółowego kontraktu między klientem a producentem oprogramowania.
- Powinien to być dokument zrozumiały dla obydwu stron.
- Tekstowy dokument SWO jest najczęściej powiązany z innymi formami specyfikacji wymagań.
- Powinien także pozwalać na weryfikację, czy wykonany system rzeczywiście spełnia postawione wymagania.

Zawartość dokumentu SWO

Informacje
organizacyjne

Streszczenie
Spis treści
Status dokumentu (roboczy, 1-sza faza, zatwierdzony, ...)
Zmiany w stosunku do wersji poprzedniej

Przykładowy
spis treści

1. Wstęp
 1. Cel
 2. Zakres
 3. Definicje, akronimy i skróty
 4. Referencje, odsyłacze do innych dokumentów
 5. Krótki przegląd
2. Ogólny opis
 1. Walory użytkowe i przydatność projektowanego systemu
 2. Ogólne możliwości projektowanego systemu
 3. Ogólne ograniczenia
 4. Charakterystyka użytkowników
 5. Środowisko operacyjne
 6. Założenia i zależności
3. Specyficzne wymagania
 1. Wymagania co do możliwości systemu
 2. Przyjęte lub narzucone ograniczenia.

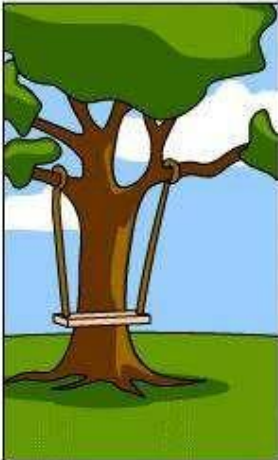
Pomocna literatura

- Dąbrowski, Subieta: *Podstawy Inżynierii Oprogramowania*, rozdział 3.
- Wiecej nt. zarządzania wymaganiami:
Leffingwell D., Widrig D., *Zarządzanie wymaganiami*, WNT, Wa-wa, 2003.

Specyfikacja oprogramowania. Inżynieria wymagań.
Karl Wiegers, Joy Beatty. Helion. 2014.



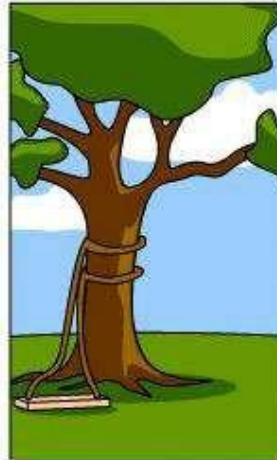
How the customer explained it



How the Project Leader understood it



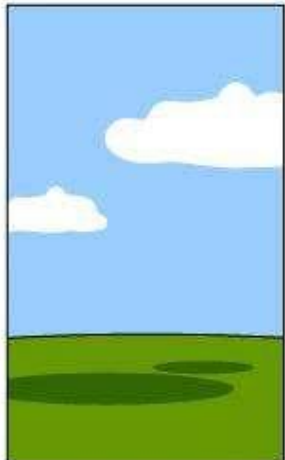
How the Analyst designed it



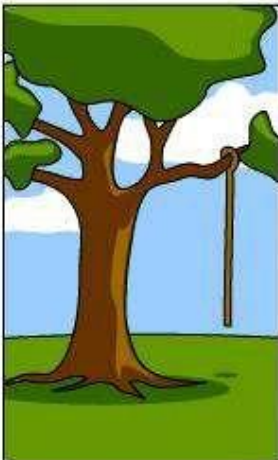
How the Programmer wrote it



How the Business Consultant described it



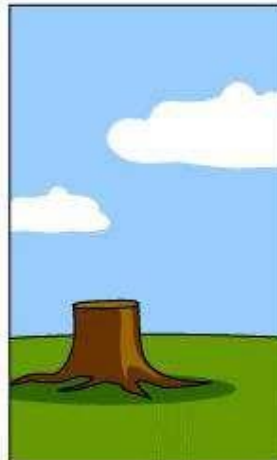
How the project was documented



What operations installed



How the customer was billed



How it was supported



What the customer really needed



What the customer wanted



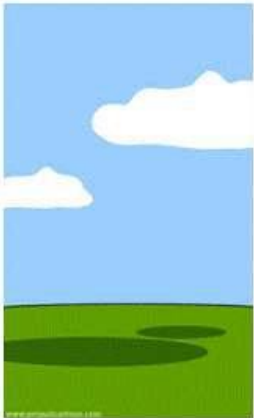
How the business consultant described it



What the design team came up with



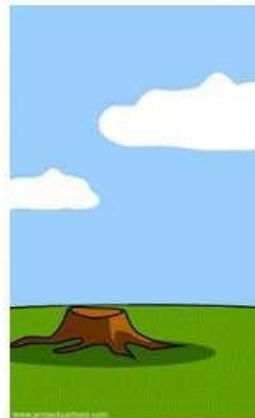
The first test version



How the project was documented



The second test version



Customer Support



Performance and durability