Projektowanie aplikacji

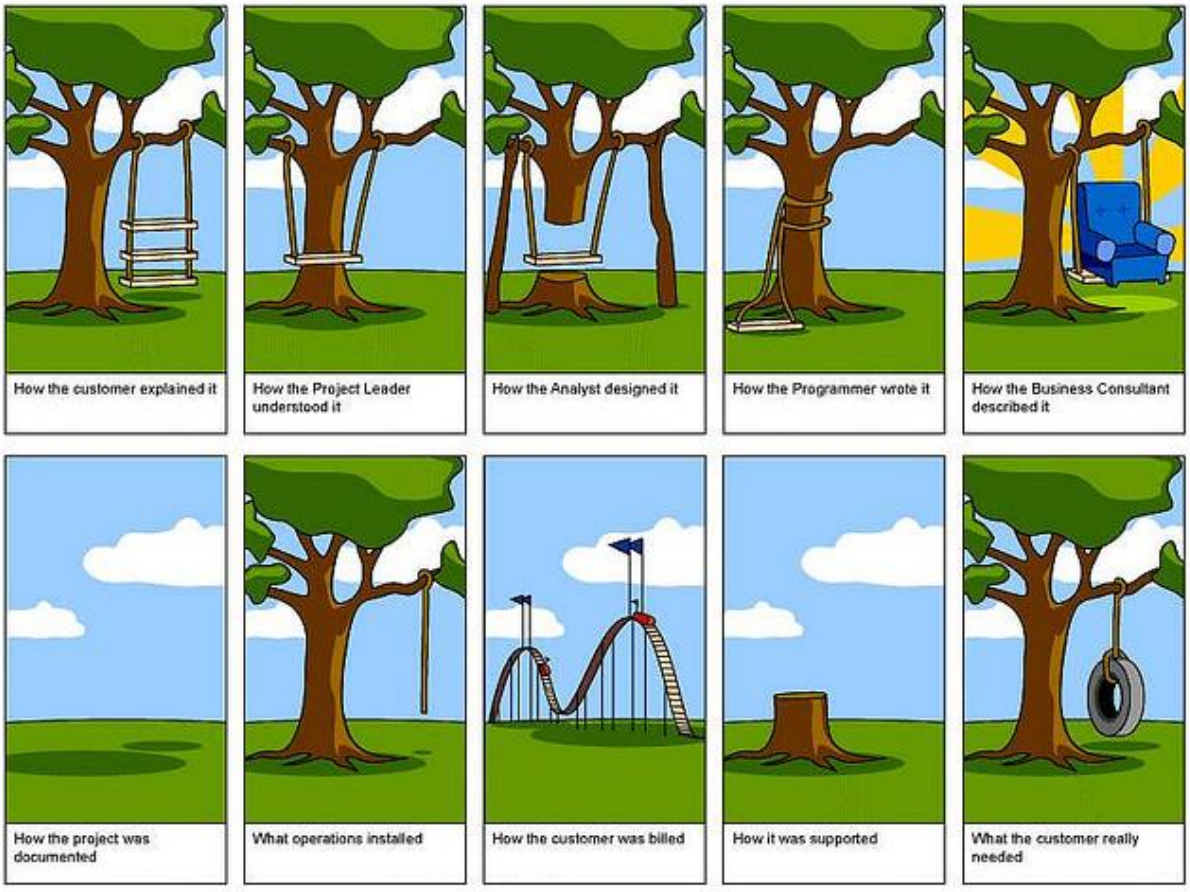
Analiza wymagań klienta

Podstawowa przyczyna problemów w komunikacji między Klientem a Dostawcą wynika z rozbieżności między tym, co klient mówi, a tym czego rzeczywiście potrzebuje.

W przypadków rozwiązań technologicznych Klient sugeruje się najczęściej rozwiązaniami, które obejrzał w Internecie, bądź próbuje przepisać dotychczasową funkcjonalność istniejącego systemu wraz z jej błędami i regułami.

Dlatego umiejętne korzystanie z technik w procesie identyfikacji wymagań pozwoli na uniknięcie niedopowiedzeń czy domysłów, które najczęściej pojawiają się na ostatnim etapie projektu IT- testów przez Klienta.

Podczas testów Klientowi wydawało się, że coś zostanie zaimplementowane, Dostawca jest nie tyle zdziwiony, co poddenerwowany tym, że w trakcie rozmów dane wymaganie nie zostało przez Klienta wypowiedziane.



Czy podczas etapu identyfikacji wymagań muszę korzystać ze wszystkich dostępnych technik? Odpowiedziałabym jak typowy handlowiec – “to zależy”. Zależy od Klienta, specyfiki projektu, ale również od poziomu naszej wiedzy dziedzinowej. Istnieją jednak “żelazne zasady”, a raczej techniki, których użycie podczas spotkania z biznesem minimalizuje ryzyko pominięcia ważnych dla niego kwestii.

Wstępna analiza wymagań

- oszacować techniczne wymagania (środowisko działania, system operacyjny, wymagania sieciowe)
- określić aplikacje, systemy z którymi musimy zintegrować system
- wymagania funkcjonalne (spis oczekiwań klienta)
- (przykładowe) scenariusze użycia

Wymagania funkcjonalne

Analiza wymagań funkcjonalnych umożliwia zidentyfikowanie i opisanie pożądanego zachowania systemu. Zgodnie z jedną z definicji, wymaganie funkcjonalne to „stwierdzenie, jakie usługi ma oferować system, jak ma reagować na określone dane wejściowe oraz jak ma się zachowywać w określonych sytuacjach. W niektórych wypadkach wymagania funkcjonalne określają, czego system nie powinien robić.

Jakie działania będzie mógł wykonać użytkownik? Przykład:
Użytkownik może się zalogować, wydrukować wynik, wyświetlić listę plików.
Po wciśnięciu przycisku OK okno się zamyka i następuje powrót do okna głównego.

Jakie działanie wykonuje system? Przykład:
Cyklicznie (codziennie) wysyłana jest poczta do wskazanych użytkowników.

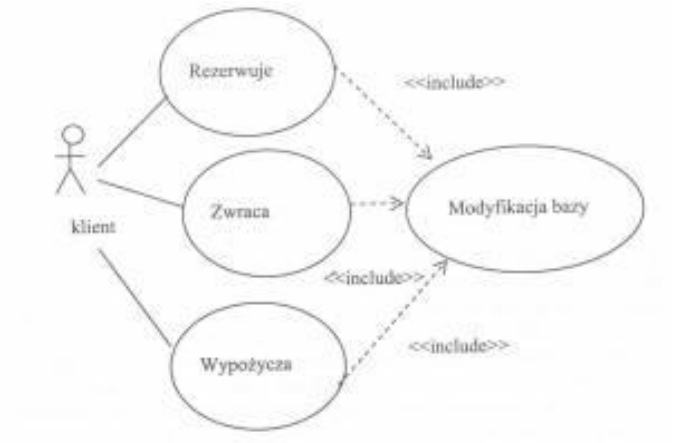
- dane wejściowe → działanie → dane wyjściowe
- ogólne – lista w języku naturalnym
- szczegółowa specyfikacja – dokładna lista, UML, grafy działania (zrobimy w późniejszej fazie projektowania)

Dobra dokumentacja wymagań nie powinna nadmiernie ograniczać projektu aplikacji, to znaczy narzucać konkretnego rozwiązania architektonicznego. Analityk powinien w taki sposób opisywać system, by prezentować dostępne funkcje i możliwości aplikacji bez zbędnego wnikania w szczegóły techniczne.

Podczas opisywania wymagań funkcjonalnych (zarówno listy wymagań, jak i szczegółowej specyfikacji), należy posługiwać się prostymi, jednoznacznymi i zrozumiałymi stwierdzeniami

Diagramy użycia

Przykładowy scenariusz użycia:



Przypadek użycia powinien przedstawiać podstawowy przebieg operacji, tzw. szczęśliwą ścieżkę wydarzeń („basic flow”, „happy flow”)

Przykład:

1. System prosi Użytkownika o zalogowanie
2. Użytkownik podaje swój numer identyfikacyjny oraz hasło
3. System stwierdza poprawność danych
4. Użytkownik zostaje zalogowany do systemu

Ścieżki alternatywne - sytuacje, gdy nie zachodzi ścieżka optymalna.
Dla punktu 3 może zajść, np.:

- System odrzuca podane dane
- Powrót do kroku 1.

Wymagania niefunkcjonalne

Ograniczenia w jakich system ma pracować, standardy jakie spełnia, itp.

Jaki ten system powinien być ?
Co jest interfejsem ? (strona www, GUI, konsola, inne aplikacje)
Czy jest przeznaczona dla pojedynczego użytkownika?
Na jakim systemie operacyjnym działa?
Jakie mam wymagania (dodatkowe oprogramowanie, dodatkowy sprzęt)?

- wydajność
- sklalowalność
- otwartość, możliwość rozbudowy
- odporność na awarie
- bezpieczeństwo

Metoda FURPS

- **Functionality** funkcjonalność w rozumieniu zestawu funkcji uwzględniająca również bezpieczeństwo, możliwości systemu
- **Usability** użyteczność jako zestaw wizualnych aspektów oprogramowania, estetyka, dokumentacja
- **Reliability** niezawodność, będąca mierzona np. częstością występowania błędów
- **Performance** wydajność aplikacji określana również jako czas odpowiedzi lub użycie zasobów
- **Supportability** przenosność, rozszerzalność, nie dająca się łatwo przetłumaczyć “wspieralność” uwzględniająca zdolność aplikacji do instalacji na różnych platformach, łatwość testowania itd.

Schemat dokumentu

1. **Wprowadzenie**
    - krótki opis celu systemu.
2. **Słownik** – definicja wszystkich technicznych i specyficznych terminów użytych w dokumencie.
3. **Ogólny opis systemu**
    - opis najważniejszych cech systemu
    - modele systemu - wyodrębnienie i opis najważniejszych modułów systemu, relacje między elementami systemu i środowiskiem systemu (np. w formie graficznej - diagram) .
      O ile uda się już na tym etapie taki model sporządzić. Na początek ogólny model (lub brak modelu).
    - grupy użytkowników i ich atrybuty (użytkownik, administrator)
    - środowisko pracy systemu (system operacyjny, sprzęt), ogólny opis interfejsów
4. **Wymagania funkcjonalne** – możliwie pełny spis usług systemu, przy czym nie muszą one jeszcze być opisywane bardzo szczegółowo (szczegółowy opis dodamy później).
    - Zwykle opis wymagań funkcjonalnych powinien zawierać następujące punkty:
        - Nazwa funkcjonalności
        - Opis – krótki opis funkcjonalności.
        - Wejście – definicja danych wejściowych i ich ewentualnych ograniczeń.
        - Wyjście – definicja zwracanych rezultatów.
        - Efekty uboczne – określenie dodatkowych czynności wykonywanych dla tej funkcjonalności, np. interakcji z innymi funkcjami.
    - można zawrzeć diagramy (przypadki) użycia.
5. **Wymagania niefunkcjonalne** – w tej sekcji przede wszystkim należy wypisać ograniczenia wynikające z oczekiwań klienta co do technologii, w której ma to być realizowane. Część z nich już mogła pojawić się we wcześniejszym opisie.
    - środowisko pracy systemu (system operacyjny), komunikacja z innymi elementami środowiska
    - wykorzystywane technologie
    - ograniczenia sprzętowe
    - ograniczenia prawne, licencje
    - inne …
    - założenia dotyczące bezpieczeństwa, wydajności , wymagania pamięciowe, itp.