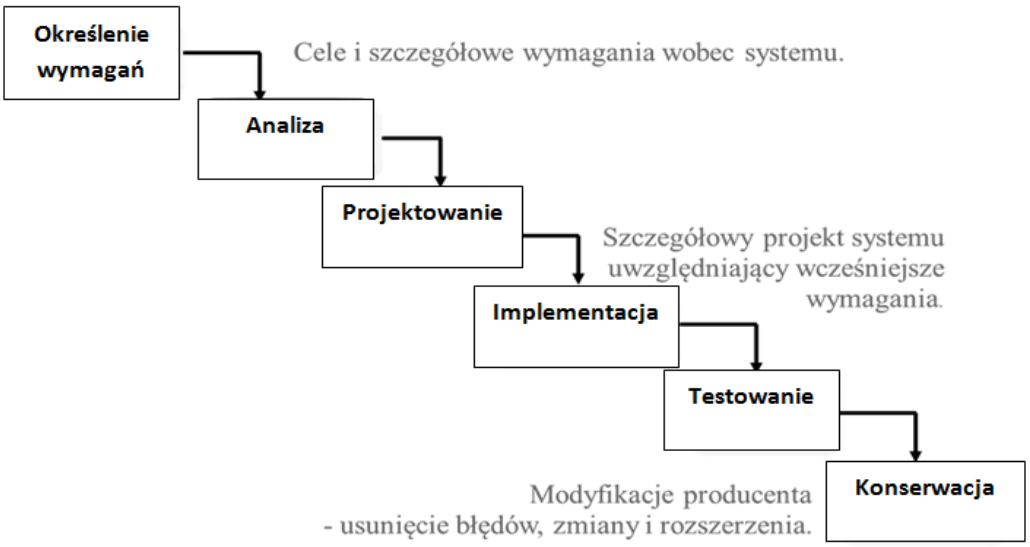
21. Modele cyklu życiowego oprogramowania, modele zarządzania projektem informatycznym

Modele Cyklu życiowego oprogramowania – modele generyczne

1. Model kaskadowy



Zalety

- przydatność w harmonogramowaniu i budżetowaniu przedsięwzięcia
- ułatwienie monitorowania postępu projektu: formalny odbiór rezultatów poszczególnych faz
- ułatwienie prowadzenia rozliczeń finansowych z klientem
- możliwość prowadzenia projektów w reżimie potokowym (przynajmniej teoretycznie)
- możliwość realizacji kierowanej dokumentami

Wady

- narzucona ścisła kolejność faz
- wysoki koszt błędów ponoszonych we wczesnych fazach cyklu
- problemy z wprowadzaniem zmian w projekcie
- długa przerwa w kontaktach z klientem

Model kaskadowy a zarządzanie projektem: realizacja kierowana dokumentami

- Model zarządzania przyjęty przez armię amerykańską dla realizacji projektów w języku Ada.
- Element kończący każdą fazę: sporządzenie szeregu dokumentów, w których opisuje się wyniki danej fazy.

Zalety:

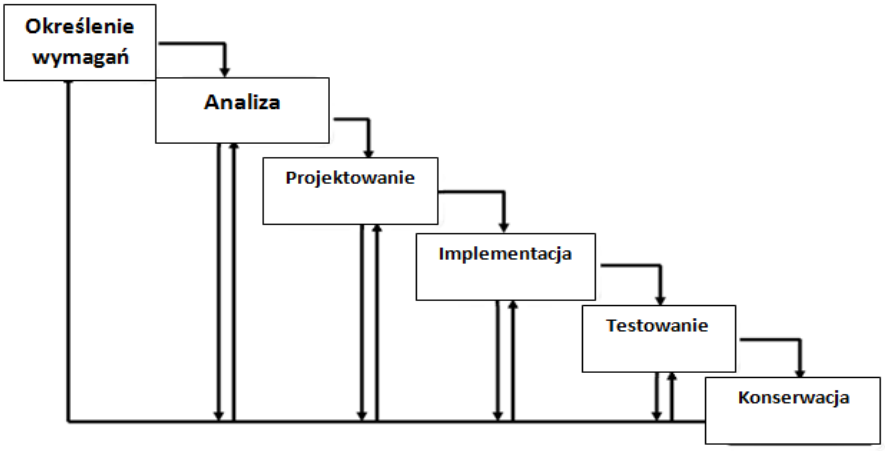
- Łatwe planowanie, harmonogramowanie, monitorowanie przedsięwzięcia.
- Możliwość (teoretyczna) realizacji dalszych faz przez inną firmę.

Wady

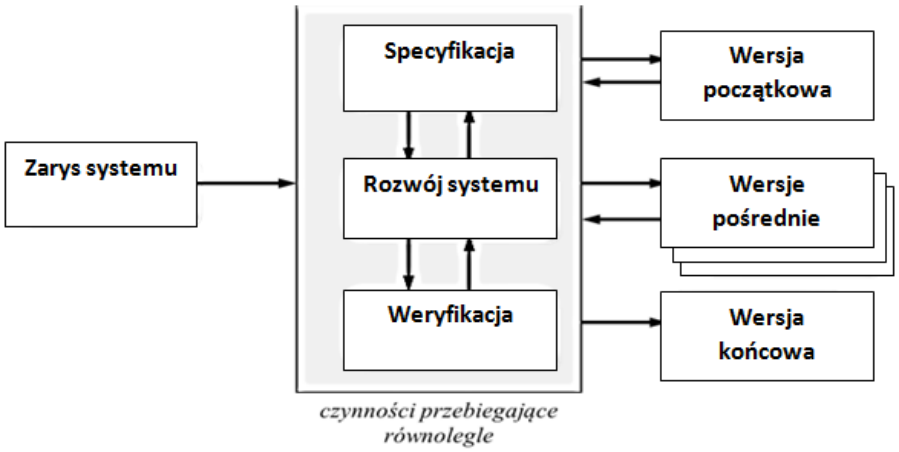
- Duży nakład pracy na opracowanie dokumentów zgodnych ze standardem (DOD STD 2167) - ponad 50% całkowitych nakładów.

Przerwy w realizacji niezbędne dla weryfikacji dokumentów przez klienta

2. Model kaskadowy z iteracjami



3. Rozwój ewolucyjny



Zalety

- b. dobry dla małych projektów lub części większej całości (np. interfejs użytkownika)
- dobry dla systemów o krótkim czasie życia (np. migracje danych)
- możliwość szybkiego startu projektu
  - przy dobrym zrozumieniu potrzeb użytkownika
  - przy słabo zdefiniowanych wymaganiach

Wady

- trudności z określeniem zaawansowania projektu
- problemy z budżetowaniem, harmonogramowaniem itp.
- złe ustrukturalizowanie wyprodukowanych systemów (najczęściej)
- częsta potrzeba specjalnych narzędzi ułatwiających budowę prototypów

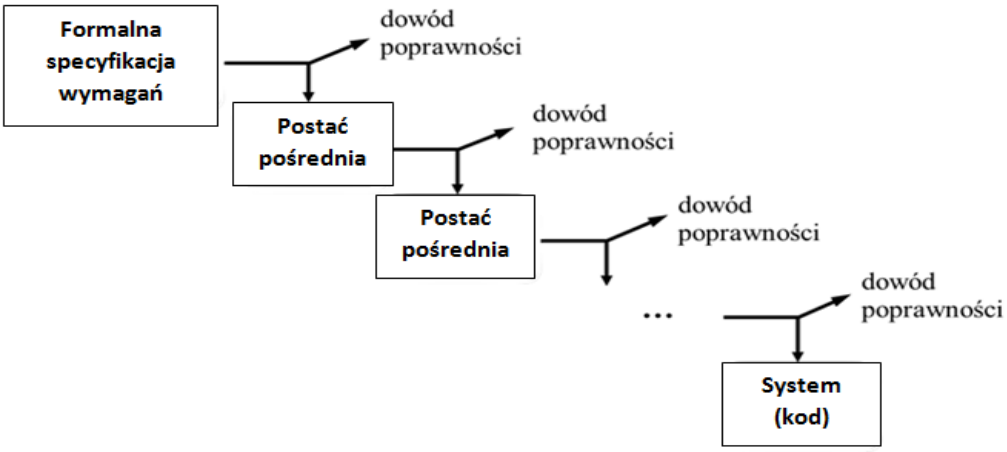
4. Prototypowanie

Sposób na uniknięcie zbyt wysokich kosztów błędów popełnionych w fazie określania wymagań. Zalecany w przypadku, gdy określenie początkowych wymagań jest stosunkowo łatwe.

Fazy	Cele	Zalety
○ ogólne określenie wymagań ○ budowa prototypu ○ weryfikacja prototypu przez klienta ○ pełne określenie wymagań ○ realizacja pełnego systemu	○ wykrycie nieporozumień pomiędzy klientem a twórcami systemu ○ wykrycie brakujących funkcji ○ wykrycie trudnych usług ○ wykrycie braków w	○ możliwość demonstracji pracującej wersji systemu ○ możliwość szkoleń zanim zbudowany zostanie pełny system

zgodnie z modelem kaskadowym	specyfikacji wymagań	

5. Transformacje Formalne



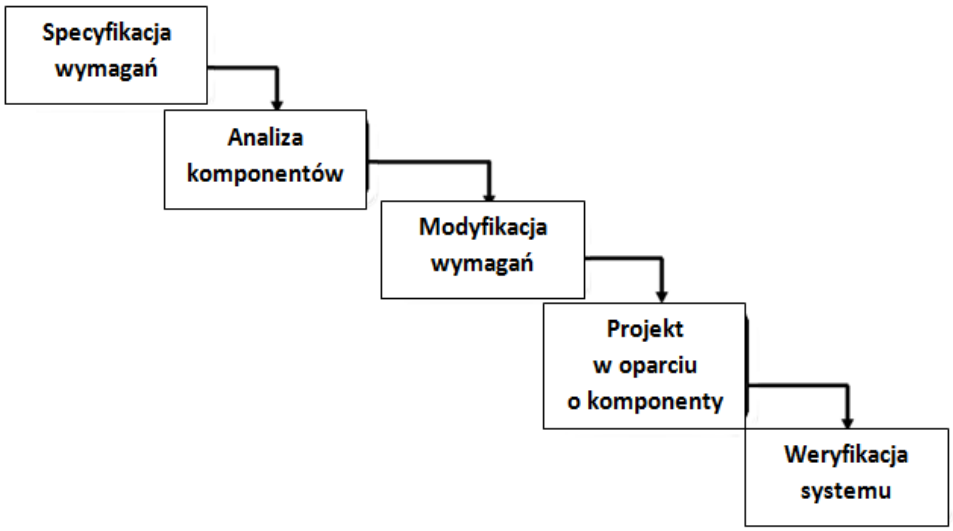
Zalety

- zagwarantowana poprawność

Wady

- trudności formalnej specyfikacji
- potrzebne wiele czasu
- kosztowny
- brak odpowiednich umiejętności matematycznych u wielu informatyków
- trudności w komunikacji z użytkownikiem
- mała efektywność kodu

6. Montaż z gotowych komponentów



Modele Cyklu życiowego oprogramowania – modele hybrydowe

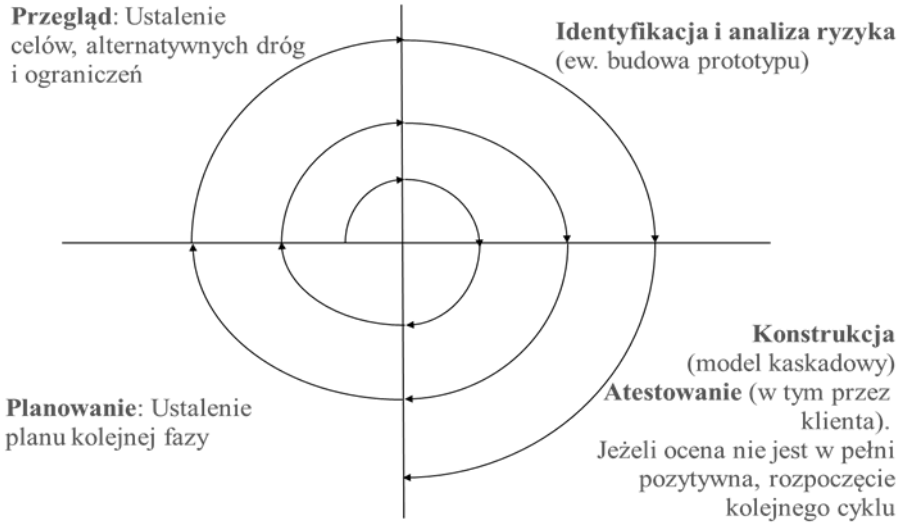
7. Model Spiralny

Zalety

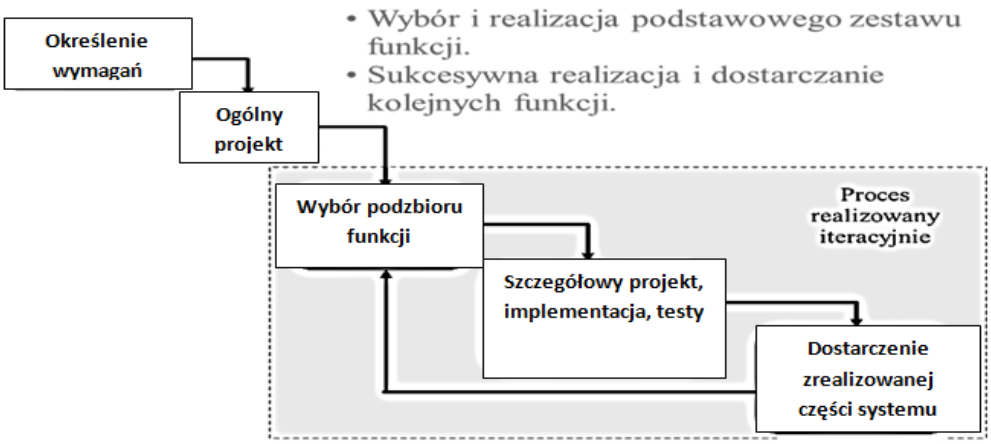
- zmniejszenie ryzyka
- elastyczność – dla każdego z zagadnień można stosować odmienny model rozwoju
- przeglądy kończące poszczególne fazy - ułatwienie szybkiego wykrywania błędów

Wady

- kontrakty często zawierają specyfikację, harmonogram oraz sposób wytwarzania dostarczanych składowych systemu → możliwe ograniczenie zastosowania modelu
- wymagane ekspertyzy z zakresu zarządzania ryzykiem



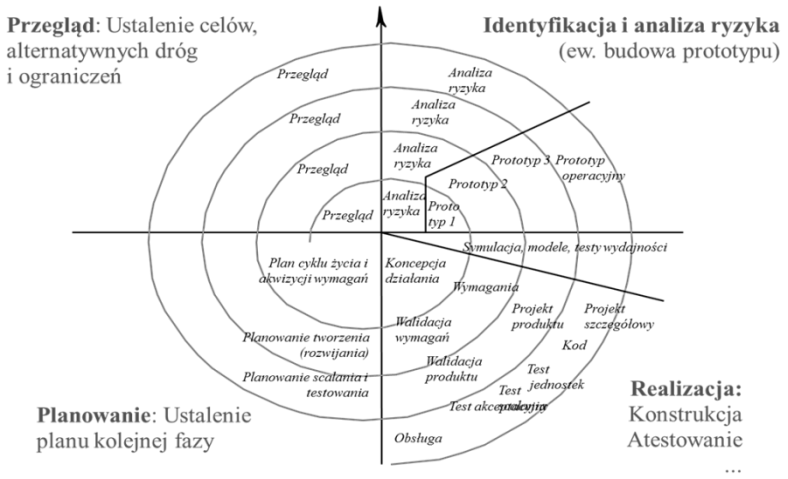
8. Model iteracyjny (realizacja przyrostowa) – odmiana modelu spiralnego



Zarządzanie projektem - model spiralny Boehma

- **Przypomnienie:** przyczyna powstania modeli hybrydowych cyklu życiowego (wytwórczego) – uwzględnienie ryzyka w związku z zarządzaniem projektem
- **Przy realizacji dużych projektów** – dostosowanie czynności merytorycznych do okresów budżetowych.

- **Konsekwencja:** przeglądy, analiza i rozstrzyganie ryzyka, konstrukcja, planowanie – stosowane do poszczególnych etapów cyklu wytwórczego
- **Rozszerzenie metody** na produkt całościowy (wersje) i elementy składowe (komponenty – podproblemy)



Zarządzanie projektem informatycznym

Zarządzanie projektem informatycznym - działalność mająca na celu:

- zapewnienie terminowej realizacji projektu,
- spełnienie przyjętych przez wytwórcę zasad tworzenia oprogramowania,
- spełnienie oczekiwań i wymagań strony zamawiającej.

Czynności wchodzące w zakres zarządzania

- Opracowanie ofert
- Wykonanie kosztorysów
- **Planowanie i harmonogramowanie**
- Nadzór nad realizacją projektu
- **Dobór osób i organizacja pracy**
- **Raportowanie, gromadzenie dokumentacji, prezentacje**

Planowanie projektu

- Planowanie - najbardziej czasochłonna czynność związana z zarządzaniem projektem.
- Faktyczny początek procesu szczegółowego planowania i przygotowania harmonogramu - jeszcze przed zawarciem umowy
- Kontynuacja - przez cały czas realizacji przedsięwzięcia.
- Planowanie - proces o przebiegu cyklicznym: W miarę realizacji kolejnych etapów przedsięwzięcia otrzymuje się informacje mające wpływ na planowanie (lub zmianę planów) następnych prac pozostających do wykonania.
- Planowanie wymaga umiejętności
  - rozsądnego gospodarowania zasobami,
  - synchronizacji działań
  - przewidywania zagrożeń.

Planowanie projektu (rodzaje planów)

Plan zarządzania jakością	Opis procedur zapewnienia jakości oraz standardy przyjęte przy realizacji projektu

Plan testów	Opis podejścia, zaangażowanych zasobów, przyjętych schematów i zasad wykonywania testów
Plan zarządzania konfiguracją	Opis wszystkich obiektów pojawiających się w projekcie (dowolne kombinacje sprzętu komputerowego, oprogramowania, usług i szkoleń)
Plan utrzymania systemu	Określenie wymagań związanych z utrzymaniem systemu.
Plan rozwoju kwalifikacji	Określenie, w jaki sposób będą podnoszone kwalifikacje osób pracujących nad projektem.

Organizacja działań

- Zakończenie każdego działania (każdej czynności) - wymierny efekt (artefakt): dokument lub element oprogramowania.
- Koniec działania (lub zespołu działań) wyznaczony przez tzw. kamienie milowe (milestones).
- Określenie:
    - zbioru zasobów potrzebnych dla każdego działania,
    - stopnia wykorzystania zasobów.
- Stosowanie odpowiednich metod weryfikacji wyników działania.

Harmonogramowanie prac

Zakres czynności

- Podział projektu na zadania.
- Oszacowanie czasu potrzebnego do realizacji zadań.
- Przypisanie zasobów potrzebnych do realizacji każdego zadania.
- Uszeregowanie zadań z uwzględnieniem kryterium optymalnego wykorzystania zasobów.
- Minimalizacja zależności pomiędzy zadaniami z uwzględnieniem kryterium czasowego.
- W przypadku wystąpienia kolizji i problemów – modyfikacja założeń w zakresie terminów i zasobów.
- Przekazanie zespołowi ustaleń wynikających z opracowanego harmonogramu.

Problemy

- Oszacowanie złożoności zadań - rzecz niezmiernie trudna; czynniki wpływające w dużej mierze na dokładność szacunku: intuicja i doświadczenie osoby zarządzającej projektem.
- Zależność wydajności od wielkości zaangażowanych zasobów - nieliniowa.
- Angażowanie dodatkowych zasobów nie powoduje bezpośredniego wzrostu wydajności (wdrożenie osoby
  w projekt, nauka posługiwania się nowym narzędziem, itp.)
- Przy układaniu harmonogramu - konieczność zachowania pewnego marginesu bezpieczeństwa („Nieoczekiwane zawsze się zdarza”).

Narzędzia

- Arkusz zasobów MS Project
- Wykres Gantta MS Project
- Diagram sieciowy MS Project