IO - inżynieria oprogramowania

dr inż. M. Żabińska,
e-mail: zabinska@agh.edu.pl
http://home.agh.edu.pl/~zabinska/

Modele cyklu życia

- **Modele cyklu życia SI/wytwarzania sofw.:** odwzorowują prowadzone działania i stany procesu wytwórczego (abstrakcyjna reprezentacja)
  - systematyzacja zagadnień dotyczących wytwarzania SI/oprogramowania
  - sposób prowadzenia przedsięwzięcia definiowany przez wybór modelu (fazy, kolejność)
  - dostarczają środki kontroli projektu
  - nie gwarantuje sukcesu (prawdopodobieństwo!)
  - dobór modelu – decyzja kierownictwa projektu – zarządzanie przedsięwzięciem projektowym

Cykl życia vs wytwarzania (1)

- Cykl **życia** systemu informatycznego

por.

- cykl **wytwarzania** SI

- **Ad 1. życie:** obejmuje całość działań od ujawnienia potrzeby budowy systemu aż do zakończenia jego wykorzystania (wycofanie); zobrazowane kolejne etapy rozwoju i eksploatacji systemu
  (+ kontekst, produkty, wzajemne relacje , zależnosci w czasie)
- **Ad 2. wytwarzanie:** obejmuje część życia: etapy wytwórcze (fazy, kroki, czynności wykonywane w fazach)

Cykl życia vs wytwarzania (2)

- **Cykl życia projektu:** okres czasu od początku do końca *projektu*
- Struktura procesu realizacji projektu – określona modelem cyklu życia
- **Model** – przybliżenie, eksponowanie czynników kosztem innych;
  - tu: rodzaj przewodnika, elastyczność kierownictwa projektu;
  - modyfikacja: uwzględnianie potrzeb i celów projektu, dostosowanie do sytuacji (przejście przez wszystkie ścieżki?)

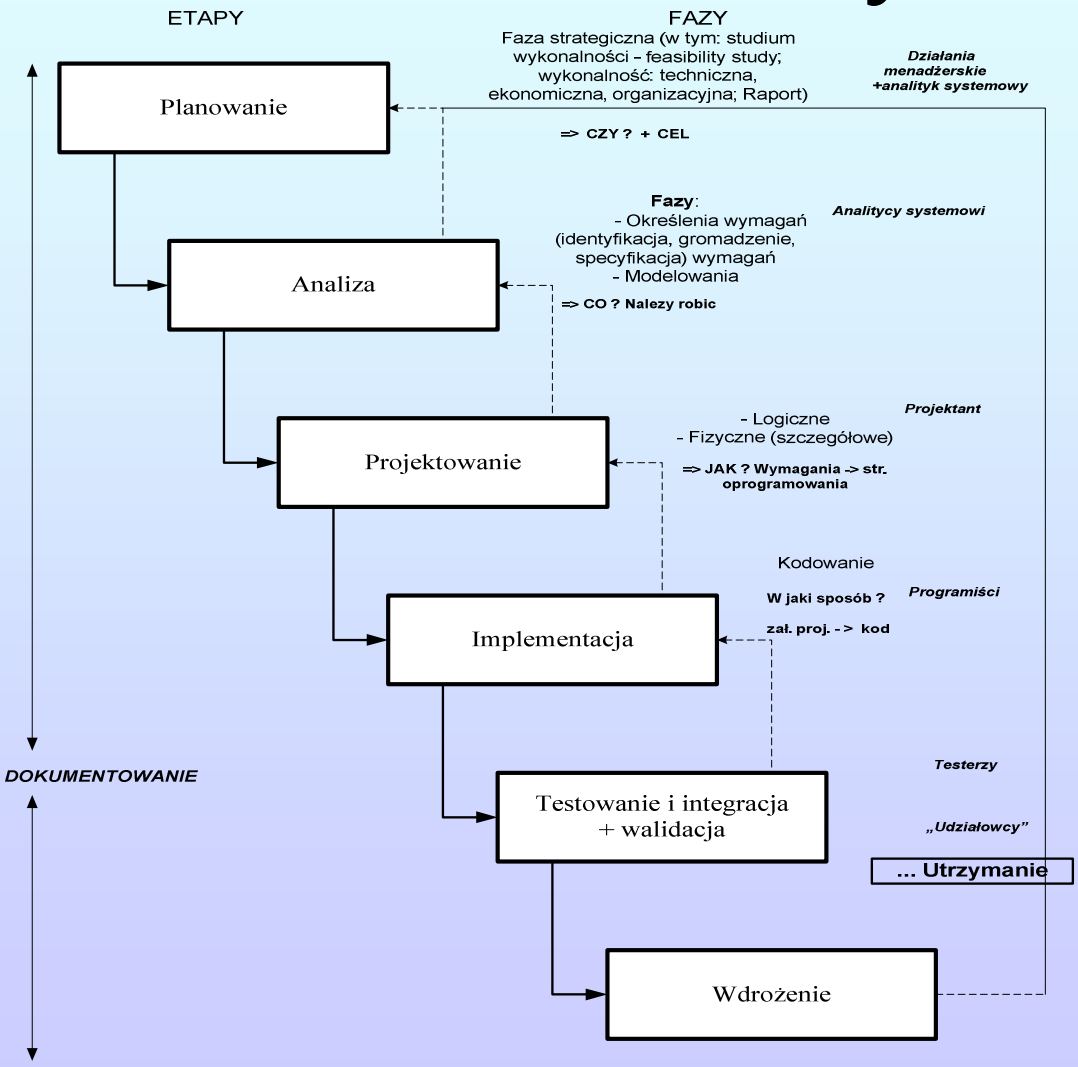
Model kaskadowy (1)

- **Model kaskadowy** *(ang.* ***waterfall model****)* – nazwa wprowadzona przez Winstona W. Royce w roku 1970, w artykule *"Managing the Development of Large Software Systems" („Zarządzanie tworzeniem dużych systemów informatycznych”);* inżynieria

Model kaskadowy (2)

- Wykonywanie czynności projektowych jako dobrze wyodrębnionych kroków **etapów** / **faz**, w określonej sekwencji, jedna po drugiej.
- Fazy, (patrz rys.), np.:
  - Faza strategiczna
  - Faza określania wymagań
- Modele procesu wytwórczego (kaskadowy, ewolucyjny, przyrostowy, spiralny, model RUP - iteracyjny i przyrostowy...)

Model kaskadowy (3)

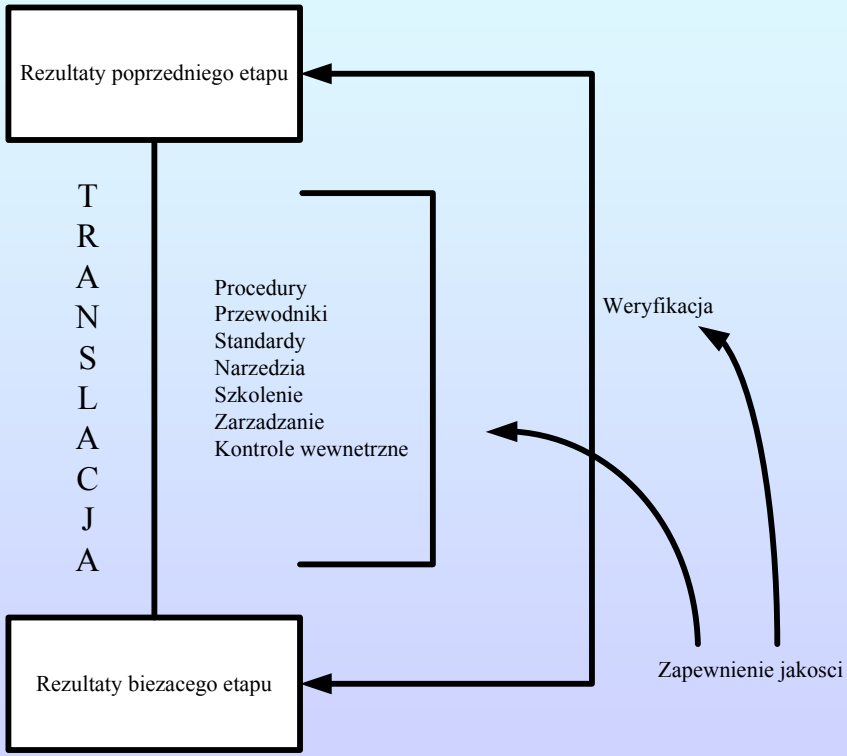


Kaskadowy (klasyczny, konwencjonalny) cykl tworzenia systemu informatycznego
(------ z iteracjami)

Model kaskadowy (4)

- Model kaskadowy: przejście do następnego etapu – wymaga ukończenia prac i dostarczenia produktów etapu poprzedniego (produkt wy => we), poprzedzone weryfikacją (element każdego etapu !)
- Wariant modelu: weryfikacja: iteracje w obrębie etapów sąsiadujących
  - *[Szejko], [Hoffer], [Jaszkiewicz], [Sommerville]*

Model kaskadowy (5)



Dzialania weryfikacyjne

Planowanie

- **Planowanie:** analiza celowości i potrzeb; uwzględnienie stanu aktualnego, działania menadżerskie (głównie)
- *Wyniki (do celu podjęcia decyzji):*
- ***Założenia dotyczące budowy systemu (wstępne i podstawowe !):***
- podst. informacje n.t. zamierzonego przedsięwzięcia,
- cele biznesowe, podstawowe wymagania (wstępne),
- ograniczenia, wstępna całościowa wizja SI (zadania, klasy użytkowników),
- perspektywy, założenia (podst.) jakościowe i parametry techniczne, standardy, organizacja i działanie systemu);

Studium wykonalności

- Raport wykonalności (ocena celowości i wykonalności – *założenia*):
  - wykonalność techniczna (warianty rozwiązania, ocena, zgodność ze standardami),
  - wykonalność ekonomiczna (analiza kosztu i zysku, zestawienia),
  - wykonalność organizacyjna (zmiany w organizacji, ryzyko, aspekty prawne)

Etap planowania - wyniki

- Opis i wstępny plan projektu (kontekst: cel, zakres, „udziałowcy”, uwarunkowania i ograniczenia); założenia: koszty, korzyści, standardy, oczekiwania, ryzyko, strategie prac, zasoby, organizacja zespołu, ramowy harmonogram, miary oceny, kryteria akceptacji)
- Podstawa do podjęcia decyzji o ustanowieniu projektu, punkt wyjścia do planowania szczegółowego

Analiza (1) – określanie wymagań

- **Analiza** (systemu, por. *faza analizy wymagań*) – ***"co"*** należy zrobić
- **Faza definiowania/ określania wymagań** (identyfikacja, gromadzenie, specyfikacja wymagań użytkowych: zakres, cele biznesowe, funkcjonalności, środowisko, jego cechy, ograniczenia użytkowe, parametry jakościowe, standardy, ch-ki systemu, por. def.systemu)
- **Wyniki:** dokument specyfikacji wymagań (F,NF), słownik (terminy dziedzinowe i informatyczne) – podlega weryfikacji przez zamawiajacych

Analiza (2) – modelowanie

- **Faza modelowania:** analiza wymagań i konstrukcja modelu logicznego (widoki)
- ***Wyniki:*** model logiczny systemu (forma graficzna: CASE – diagramy + opis), specyfikacja wymagań na oprogramowanie, szkic podręcznika użytkownika, rozwinięty słownik danych

Projektowanie

- **Etap projektowania** (odwzor. wymagań w metody, ***„jak”***)
- Faza projektowania logicznego (projektowanie wysokiego poziomu: struktura, komponenty, architektura); wymagania -> struktura oprogramownia i jego podstawowe komponenty
- ***Wyniki:*** dokumentacja projektu konstrukcyjnego systemu, słowniki
- Faza projektowania szczegółowego (szczegóły, interfejsy, sposób użytkowania systemu)
- ***Wyniki:*** dokumentacja projektu szczegółowego systemu, uaktualniony słownik danych, podręcznik użytkownika oprogramowania

Etap implementacji

- **Etap implementacji** (odwzor. proj.+ założeń -> kod, tworzenie kodu, uruchamianie, testowanie względem przyjętej specyfikacji projektowej)
- ***Wyniki:*** kod, opis, dokumentacja testów i wyników, podręczniki użytkowania, administrowania: w tym instalowania)

Testowanie i integracja

- **Etap testowania i integracji** (plan testowania, scalanie, sprawdzenie zgodności: funkcjonalności, użyteczności, – z oczekiwaniami oraz poprawności, efektywności)
- ***Wyniki:*** *oprogramowanie poprawione*, *dokumentacja,* opis ewentualnych zmian, dokumentacja testów i wyników, *podręczniki*: użytkowania i instalacji – poprawione

Etap wdrożenia i utrzymania

- **Etap wdrożenia systemu** (instalacja, przekazanie, przeprowadzenie szkoleń, ewentualna reorganizacja przedsiębiorstwa)
- ***Wyniki:*** dokumentacja – raporty
- **Etap utrzymania** (pielęgnacja, konserwacja): usuwanie defektów, poszerzanie funkcjonalności, dostosowywanie do zmian

Utrzymanie systemu

- Rodzaje:
  - utrzymanie naprawcze *(corrective maintenance)*,
  - adaptacja - zmiany środowiska (*adaptive maintenance*),
  - ulepszenia (*perfective maintenance*),
  - utrzymanie prewencyjne - przyszłe zmiany *(preventive maintenance).*
- ***Wyniki:*** dokumentacja – raporty
- Dbałość o ***aktualizację dokumentacji równolegle*** do prac !
- *Por. inne modele tradycyjne*

Model kaskadowy – wnioski (1)

- *Model kaskadowy (Waterfall Model) –* ***<u>podsumowanie:</u>***
  - *główne procesy projektu – etapy realizowane w ściśle zdefiniowanej kolejności;*
  - *każdy etap musi być zakończony przed rozpoczęciem następnego;*
  - *nie występuje (wprost) weryfikacja;*
  - *działania weryfikacyjne w ramach kazdej fazy, oraz pomiędzy sąsiednimi – decyzja kierownictwa projektu.*

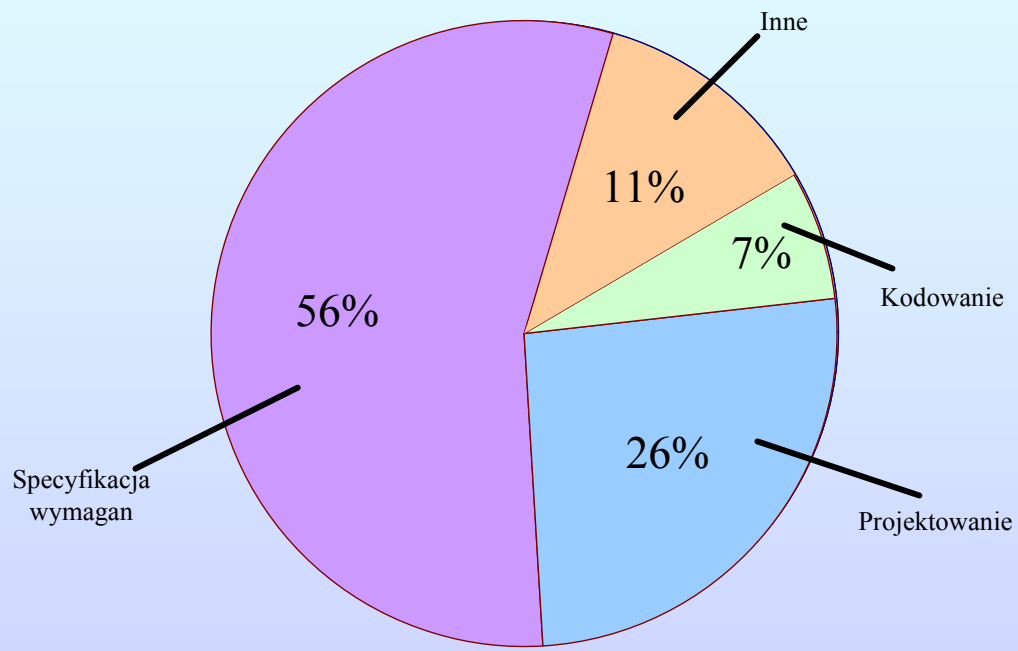
Model kaskadowy – wnioski (2)

- Konsekwencje: **zalety** – strukturalność modelu
- dobrze zdefiniowany ciąg kroków
- z ustalonymi produktami i dokumentami
- dobrze sprawdzony (używany)
- sprawdził się w wielu rzeczywistych projektach
- naturalny i zrozumiały
- możliwość wykorzystania różnych technologii (metodyk strukturalnych i obiektowych)
- propozycja standaryzacji modelu (US Dept. of Defence, European Space Agency, National Computing Centre - GB) *[Szejko2002: Metody wytwarzania oprogramowania]*

Model kaskadowy – wnioski (3)

- Konsekwencje: **wady** – strukturalność modelu
- każda modyfikacja wymaga cofnięcia się
- proces zależy w wysokim stopniu od stabilności wymagań – w praktyce trudne do uzyskania
- milczące założenie: można wytworzyć poprawną specyfikację wymagań już na początku projektu- w praktyce niemożliwe! trudności: pozyskiwanie wymagań; przygotowywanie specyfikacji; uświadomienie(użytkownicy) kompletu wymagań
- walidacja: na końcu - brak udziału użytkownika w projektowaniu i implementacji; koszty naprawy

Model kaskadowy – wnioski (4)



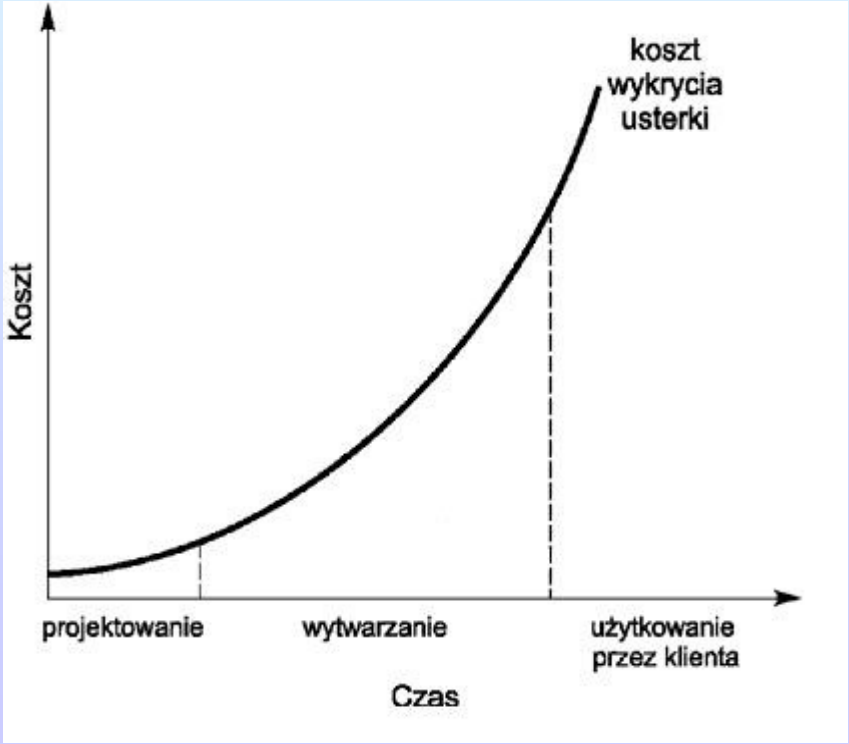
Rozklad bledow miedzy fazy klasycznego cyklu wytwarzania

Model kaskadowy – wnioski (5)

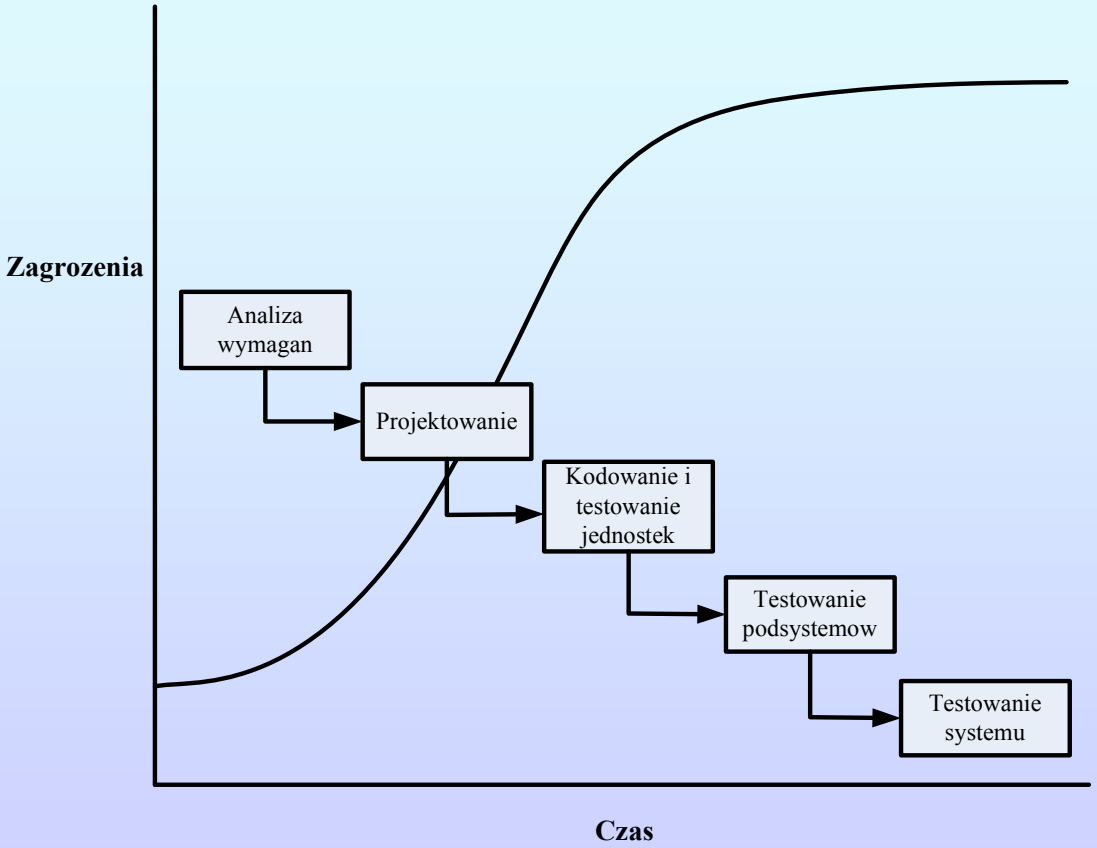
- Rozkład błędów między fazy... (Rys.)
  [Szejko: Metody wytwarzania oprogramowania]
- Znalezienie i usunięcie usterek jest o 100 do 1000 razy droższe po wdrożeniu !!!
  [Kruchten: RUP]
- Zagrożenia rosną z czasem (Rys.)
  [Kruchten: RUP]

Model kaskadowy – wnioski (6)

- Znalezienie i usunięcie usterek jest o 100 do 1000 razy droższe po wdrożeniu !!! *[Kruchten: RUP]*



Model kaskadowy – wnioski (7)



Kaskadowy okres trwania przedsiewziecia

Model kaskadowy – wnioski (8)

- Wg *[Cantor: Jak kierować zespołem programistów]* – w 1996 roku zespół *STSC – The Software Technology Support Center* (US Air Force) opublikował 2 tomy artykułów n.t. budowania dużych SI: „Ogólnie rzecz biorąc stosowania metody kaskadowej NIE zaleca się w poważnych przedsięwzięciach...” (*por. Wnioski 2*)

Model kaskadowy – wnioski (9)

- Uwagi praktyczne
- może być użyty z powodzeniem w krótkich projektach
- gdy wymagania dobrze określone i zrozumiane
- specyfikacja wymagań – w pełni przygotowana
- długie projekty: należy zapewnić właściwą procedurę zarządzania zmianami wymagań, dobrą współpracę z „udziałowcami”: klientem/użytkownikiem/kierownictwem projektu => zmiany oczekiwań powinny być odzwierciedlone w systemie
- inne modele procesu wytwórczego

Model kaskadowy – wariant: realizacja kierowana dokumentami

- *„Dokument-driven”* – lata 80-te, model zaproponowany przez armię amerykańską
- Duże wydatki na program „gwiezdnych wojen”; wkład w rozwój technik inż. opr.
- Wymagany sposób wykonywania oprogramowania dla armii am. standardy: DOD STD: 2167, 2167A, 7935A, 1703
- Opisywany model cyklu życia =*dd; później: MIL STD 498 (1994) -> IEEE12207 (1998)*

Realizacja kierowana dokumentami(1)

- Model *document-driven* (realizacja kier. dokumentami) – ścisła realizacja modelu kaskadowego: fazy następujące po sobie
- Każda faza kończy się opracowniem szeregu dokumentów – w pełni opisujących wyniki tej fazy
- Dokumenty powinny być podstawą do realizacji dalszych faz; Udostępniane klientowi; Po zaakceptowaniu dokumentacji przez klienta – kolejna faza

Realizacja kierowana dokumentami(2)

- Zalety (jak model kaskadowy,istota):
  - łatwe planowanie
  - łatwe harmonogramowanie
  - łatwe monitorowanie przedsięwzięcia
  - możliwość (przynajmniej teoretyczna) przerwania realizacji przedsięwzięcia w jednej firmie i wznowienia realizacji w innej firmie, po przekazaniu kompletu dokumentów

Realizacja kierowana dokumentami(3)

- Wady (jak model kaskadowy,istota) oraz:
  - duży nakład pracy na opracowanie dokumentów zgodnych ze standardem DOD STD 2167 – ponad 50 % całkowitych nakładów
  - przerwy (niezbędne, czas !) w realizacji przedsięwzięcia niezbędne dla weryfikacji dokumentów przez klienta
  - inne organizacje (np. *IEEE 12 207*) własne standardy realizacji przedsięwzięć na bazie DOD STD 2167; stos. uproszczona wersja

Inne modele (klasyczne) (1)

- Model procesu tworzenia – abstrakcyjna reprezentacja procesu
- Modele nie są ostatecznym opisem – użyteczne abstrakcje
- Zręby procesów a nie szczegóły czynności
- Dla wielu „dużych” systemów–tworzenie części systemu,użyto różnych procesów

Inne modele (klasyczne) (2)

- Wady modelu kaskadowego – tworzenie innych, nowych modeli
  - model przyrostowy (iteracyjność)
  - model spiralny (iteracyjność)
  - model z prototypowaniem (tworzenie ewolucyjne: wstępna implementacja, pokazanie użytkownikowi, udoskonalanie)

Nieklasyczny (na bazie doświadczeń):

- model iteracyjny i przyrostowy (*RUP*)

Iteracja procesu (1)

- Iteracja procesu: powtarzanie fragmentów procesu w miarę ewolucji wymagań stawianych systemowi
  - Prace projektowe i implementacyjne muszą być wykonywane ponownie, by spełnić zmienione wymagania
  - Różne podejścia do części systemu – modele hybrydowe wspomagające iterację procesu:
  - Tworzenie przyrostowe – ciąg przyrostów f.
  - Tworzenie spiralne – system rozwija się na zewnątrz od szkicu, do końcowego systemu

Iteracja procesu (2)

- Tworzenie przyrostowe:geneza, koncepcja
  - model kaskadowy wymusza zatwierdzanie zbioru wymagań przed projektowaniem
  - zmusza do zatwierdzania strategii projektowej przed przystąpieniem do implementacji
  - zmiany wymagań w trakcie tworzenia => powtórzenie prac nad wymaganiami, projektem i implementacją
  - Zaleta modelu kaskadowego: rozdzielenie projektowania i implementacji; prostota procesu zarządzania, ale...

Tworzenie przyrostowe (1)

- Podejście przyrostowe (***incremental development***) – [*Mills i in., 1980*]
  - sposób na ograniczenie powtarzania całości prac w procesie tworzenia
  - możliwość odkładania decyzji –szczegółowe wymagania, do czasu zdobycia doświadczenia w pracy z systemem
  - tworzenie przyrostowe – pośrednie; łączy zalety podejścia kaskadowego i ewolucyjnego
  - stosuje się do przypadków, gdy dopuszczalna jest okrojona funkcjonalność systemu

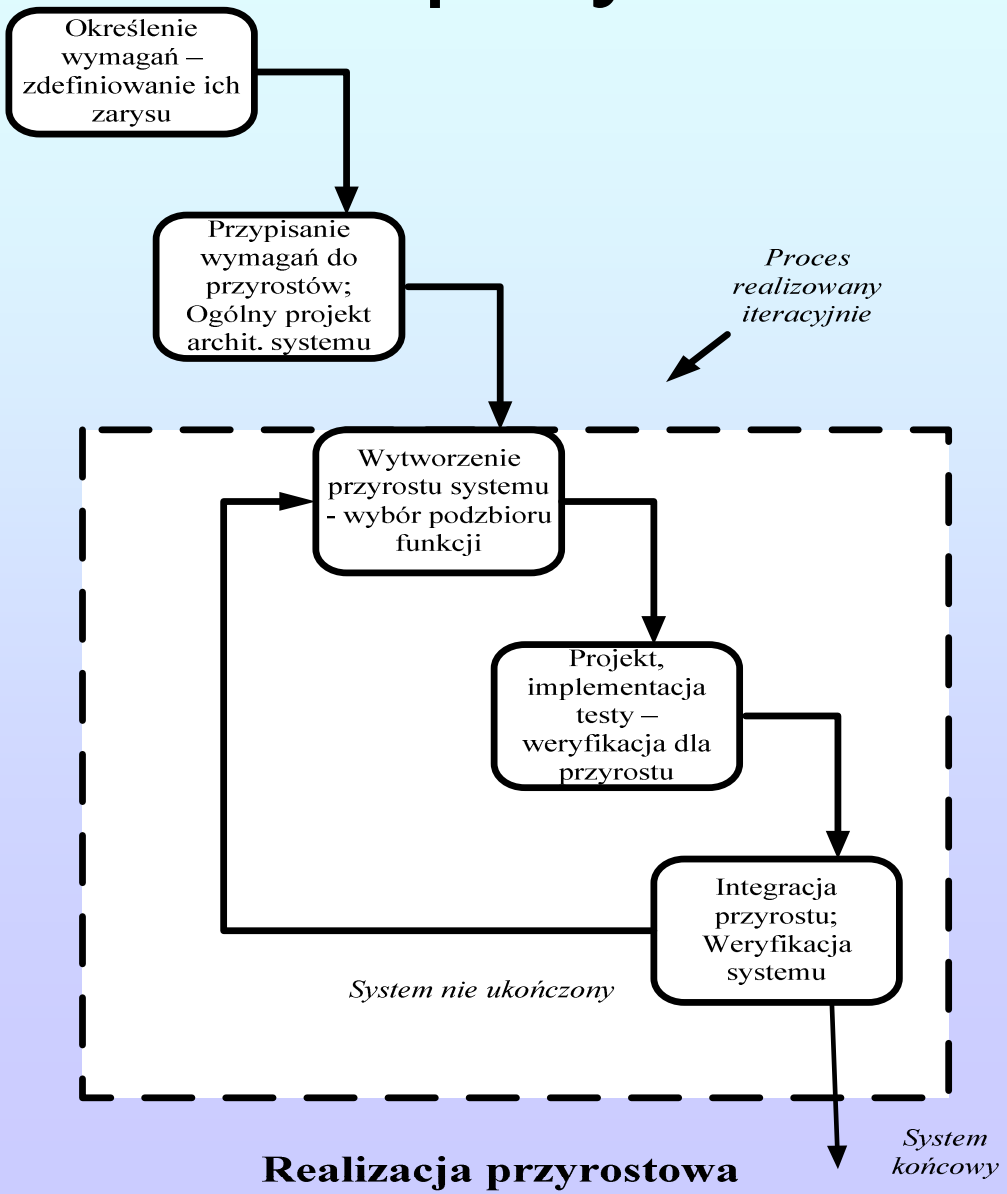
Tworzenie przyrostowe (2)

- Idea:
  - w procesie przyrostowym klienci identyfikują w zarysie usługi, które system ma oferować
  - wskazują hierarchię ważności usług
  - definiuje się pewną liczbę przyrostów, które mają być dostarczone
  - przyrost – część funkcjonalności systemu
  - przyporządkowanie usług do przyrostów, zależy od priorytetu tych usług
  - usługi najważniejsze dostarczane pierwsze

Tworzenie przyrostowe (3)

- Idea, c.d.:
  - gdy zdefiniuje się przyrosty, szczegółowo definiuje się wymagania stawiane usługom dostarczanym w pierwszym przyroście
  - przyrost jest tworzony za pomocą najbardziej odpowiedniego procesu, wybór modelu
  - w trakcie prowadzi się dalszą analizę wymagań dla następnych przyrostów
  - gdy przyrost gotowy – klienci uruchamiają: szybko – część funkcjonalności, ekperymenty
  - =>ustalanie nast.: wymagań i wersji przyrostu

Tworzenie przyrostowe (4)



Realizacja przyrostowa
(incremental development)

Tworzenie przyrostowe (5)

- Fazy:
  - rozpoczyna się od: określenia całości wymagań, wykonania wstępnego, ogólnego projektu całości systemu; potem:
  - wybór pewnego podzbioru funkcji systemu,
  - następnie: szczegółowy projekt (wg modelu kaskadowego) oraz implementacja części systemu realizującej wybrane funkcje
  - testowanie zrealizowanego fragmentu i dostarczenie go klientowi
  - powtarzanie procesu

Tworzenie przyrostowe (6)

- Uwagi:
  - po zakończeniu prac nad kolejnymi wymaganiami, integracja z istniejącymi przyrostami
  - funkcjonalność systemu poprawia się z każdym dostarczonym przyrostem
  - najczęściej używane usługi – najwcześniej lub implementowane przyrostowo w miarę realizacji wymagań kolejnych przyrostów (first-things-first)
  - tworzenie przyrostu – najbardziej odpowiedni proces: wybór; specyfikacja usług staranna w pewnym przyroście => model kaskadowy; niejasna => lepiej zastosować tworzenie ewolucyjne

Tworzenie przyrostowe (7)

- Zalety:
  - częste kontakty z klientem (skrócenie przerw por. model kaskadowy), używanie: dośw., szkolenia
  - brak konieczności zdef. z góry całości wymagań
  - wczesne wykorzystanie przez klienta fragmentów systemu (testowanie funkcjonal.), mniejsze ryzyko
  - pierwszy przyrost spełnia najważniejsze wymagania, można używać systemu (prototyp ?)
  - potencjalne opóźnienia: możliwość elastycznego reagowania – opóźnienie realizacji fragmentu – przyspieszenie prac nad inną/innymi częściami (sumarycznie – bez opóźnienia całości przedsięwzięcia projektowego)

Tworzenie przyrostowe (8)

- Wady:
  - przyrosty powinny być małe (nie więcej niż 20000 wierszy kodu)
  - każdy przyrost powinien dostarczać pewną funkcjonalność systemu=>może być trudno przypisać wymagania klienta do przyrostów, potencjalne trudności z wycinaniem podzbioru funkcji w pełni niezależnych
  - trudno zdefiniować wspólne udogodnienia potrzebne wszystkim przyrostom
    - dodatkowy koszt konieczność implementacji szkieletów (interfejs zgodny z docelowym systemem) ryzyko nie wykrycia błędów w fazie testowania

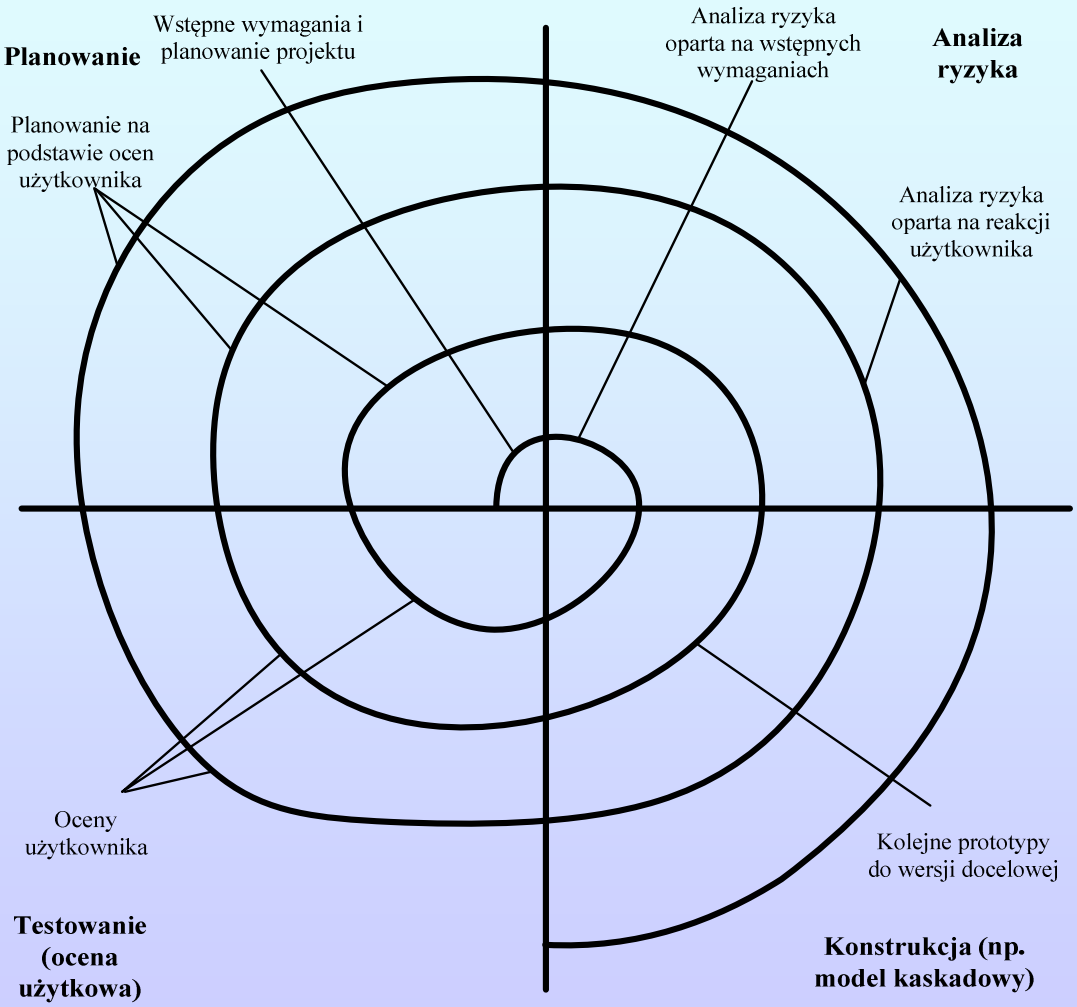
Tworzenie przyrostowe (9)

- Konsekwencje:
  - Ewolucja podejścia przyrostowego doprowadziła do powstania „programowania ekstremalnego” [*Beck, 1999*] =>
    - tworzenie i dostarczanie bardzo małych przyrostów funkcjonalności
    - włączenie klienta w cały proces
    - ustawiczne poprawianie kodu
    - programowanie pozbawione indywidualizmu (2-10)
    - wymagania: słabo sprecyzowane lub szybko się zmieniające

Tworzenie spiralne (1)

- Ogólny model, twórca [*Boehm 1988*], idea:
  - proces nie jest przedstawiony jako ciąg czynności z nawrotami, lecz ma postać spirali
  - każda pętla obejmuje cztery fazy/ sektory wykonywane cyklicznie
    - Planowanie (nowej wersji),
    - Analiza ryzyka,
    - Konstrukcja,
    - Testowanie.

Tworzenie spiralne (2)



Model spiralny

Tworzenie spiralne (3)

- *Planowanie:*
  - ustalane są generalne cele produkcji kolejnej wersji systemu
  - identyfikuje się ograniczenia (proces i produkt)
  - szczegółowe plany zarządzania
  - rozpoznanie zagrożeń

Tworzenie spiralne (4)

- *Analiza ryzyka:*
  - rozważane – ogólne opcje budowy nowej wersji systemu
  - analiza możliwości w aspekcie ryzyka związanego z realizacją (ocena rozwiązań alternatywnych, identyfikacja i oszacowanie związanego ryzyka: techniczne, ekonomiczne, organizacyjne dotyczące realizacji nowej wersji systemu):
    - szczegółowa analiza rozpoznanych zagrożeń
    - kroki zmierzające do redukcji tych zagrożeń
    - faza ta może obejmować także budowę prototypu

Tworzenie spiralne (5)

- *Konstruowanie:*
  - tworzenie kolejnej wersji – przybliżenia systemu (wg wybranego modelu, np. w sposób zgodny z modelem kaskadowym)
  - wybór modelu tworzenia systemu: np. najpoważniejsze zagrożenie związane z interfejsem użytkownika => prototypowanie ewolucyjne; główne ryzyko związane z integracją podsystemów => model kaskadowy

Tworzenie spiralne (6)

- *Testowanie (ocena użytkowa):*
  - recenzowanie przedsięwzięcia
  - ocena przez klienta kolejnej wersji systemu (jeśli nie jest w pełni pozytywna – rozpoczynany jest kolejny cykl)
  - Uwaga:
  - nie ma staych faz; model obejmuje inne procesy; w pętlach można użyć różnych modeli: np. prototypowania w celu zmniejszenia zagrożeń (niejasne wymagania);

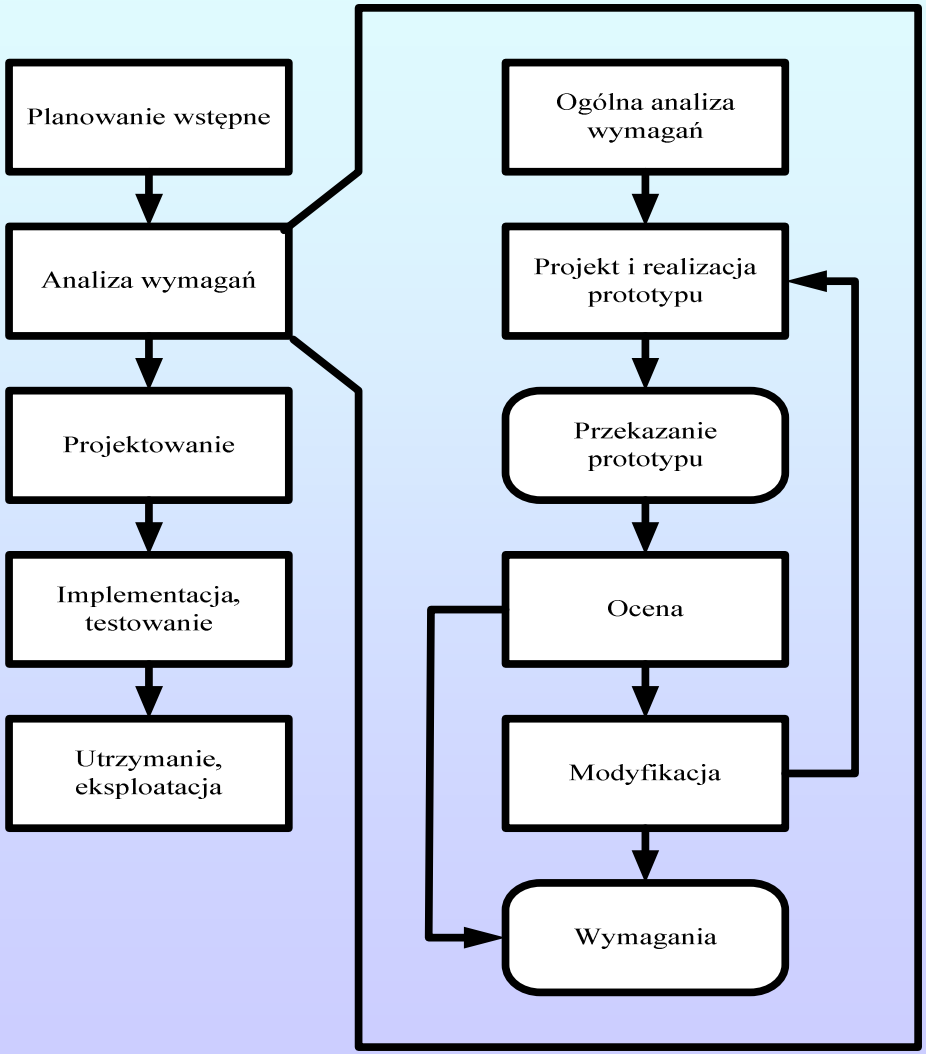
Tworzenie spiralne (7)

- Zalety:
  - jawne potraktowanie zagrożeń
  - zmniejszenie ryzyka
  - wprowadzenie oceny przez użytkownika
- Wady:
  - przydatny dla systemów, które mogą być wdrażane przy okrojonej funkcjonalności i obniżonej jakości
  - wersja docelowa – długi czas osiągnięcia

Tworzenie ewolucyjne (1)

- Cechy:
  - opracowanie wstępnej implementacji
  - pokazanie jej użytkownikowi
  - udoskonalanie w wielu wersjach aż do powstania odpowiedniego systemu
  - eksperymentowanie z tymi wymaganiami, które są niejasne =>
    - prototypowanie z porzuceniem

Prototypowanie z porzuceniem (1)



Schemat prototypowania

Prototypowanie z porzuceniem (2)

- Cel: minimalizacja ryzyka związanego z niewłaściwym określeniem wymagań
  - lepsze określenie wymagań,
  - wykrycie nieporozumień pomiędzy klientem a twórcami systemu,
  - wykrycie brakujących funkcji,
  - wykrycie trudnych usług,
  - wykrycie braków w specyfikacji wymagań.

Prototypowanie z porzuceniem (3)

- Fazy:
  - ogólne określenie wymagań,
  - budowa prototypu,
  - weryfikacja prototypu przez klienta,
  - pełne określenie wymagań,
  - realizacja pełnego systemu zgodnie z modelem kaskadowym,
  - określenie wymagań.

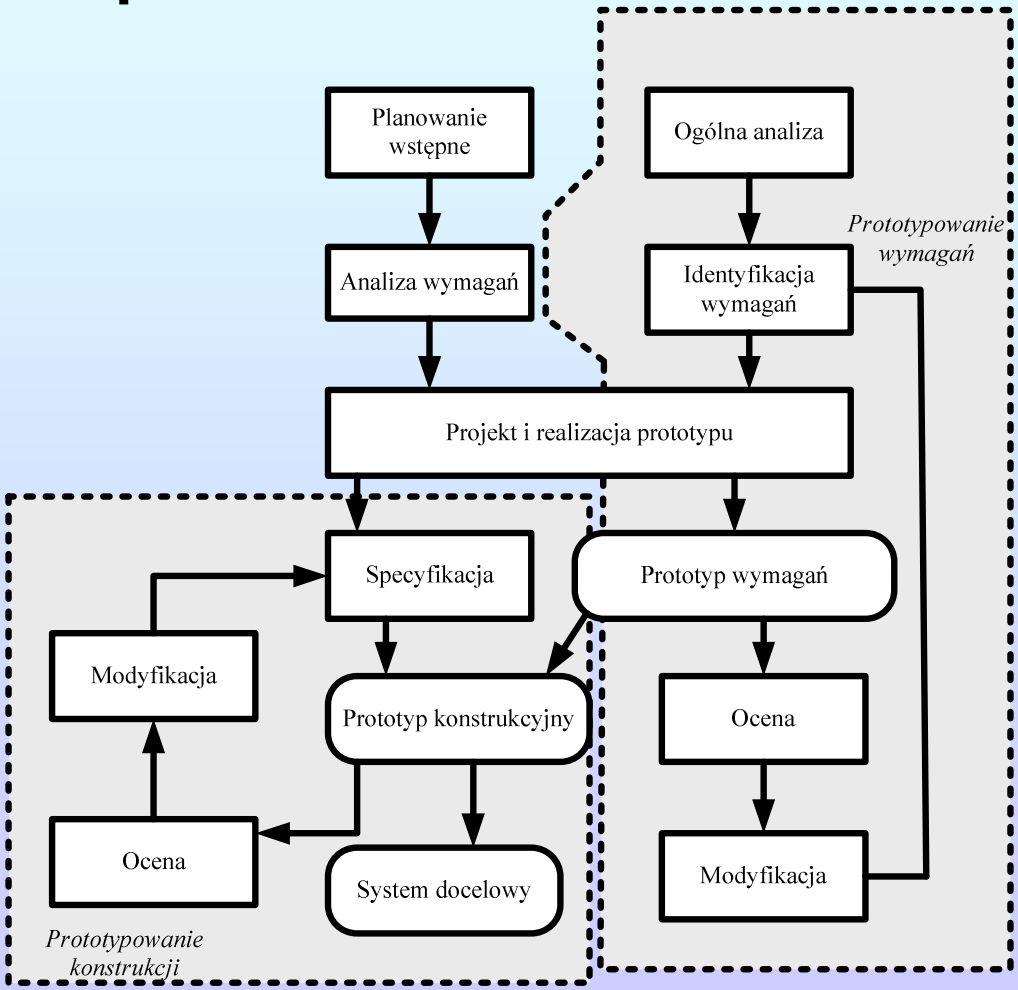
Prototypowanie z porzuceniem (4)

- Zalety:
  - możliwość szybkiej demonstracji pracującej wersji systemu
  - mozliwość szkoleń przed zbudowaniem pełnego systemu
- Wady:
  - dodatkowy koszt budowy prototypu
  - potencjalne zdziwienie klienta (czas i koszt budowy prototypu/ por. pełnego systemu)

Prototypowanie z porzuceniem (5)

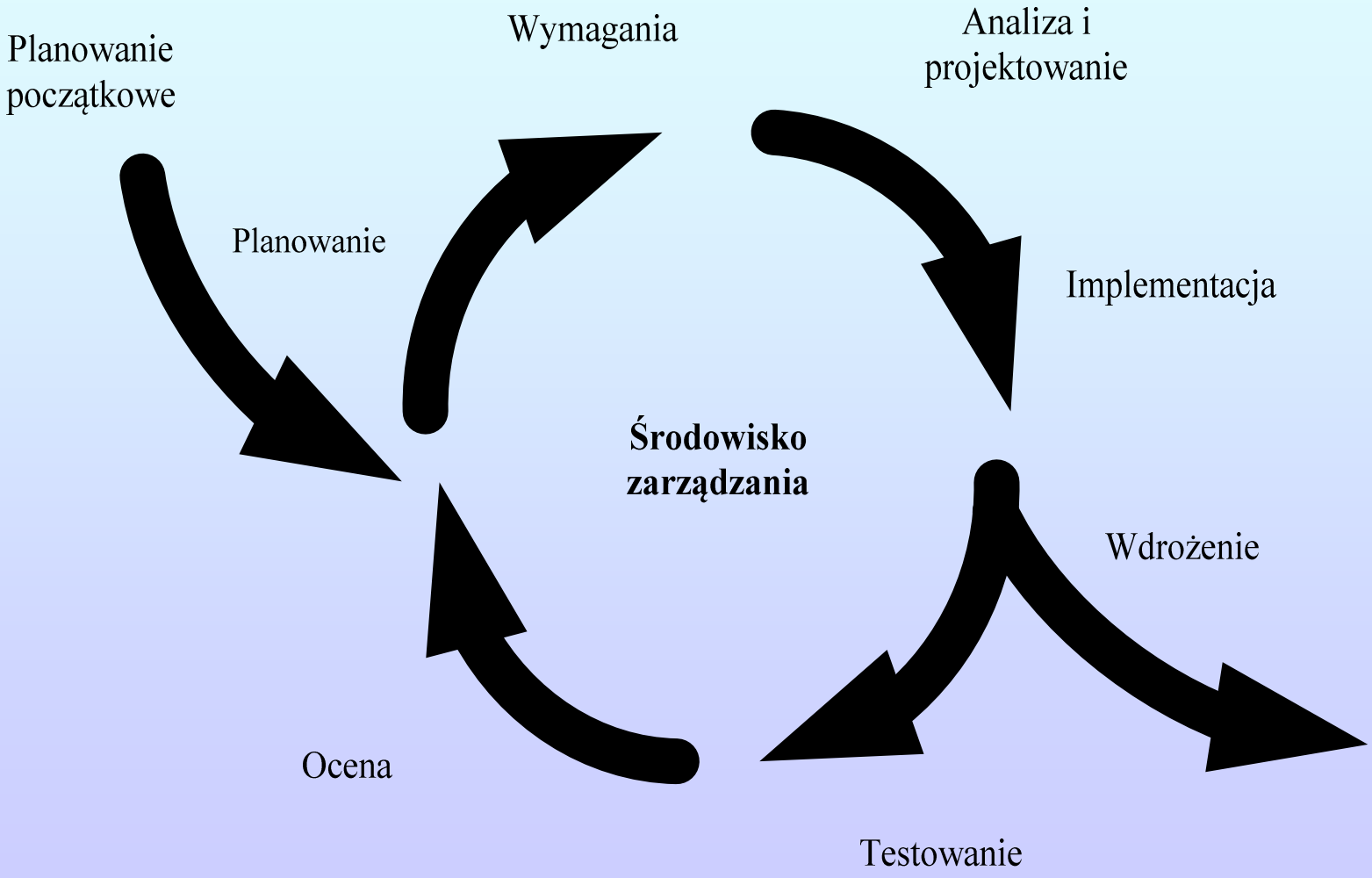
- Uwagi:
  - prototyp nie jest częścią przyszłego pełnego systemu (wykorzystanie fragmentów)
  - dążenie do szybkiej realizacji (często kosztem niezawodności i efektywności)
  - pomoc w lepszym określeniu wymagań
  - specyficzny sposób realizacji fazy określenia wymagań (dla modelu kaskadowego)
  - dalszy przebieg prac wg tego modelu. budowy prototypu/ por. pełnego systemu)

Tworzenie ewolucyjne (badawcze) – praca z klientem (3)



Procesy prototypowania wytwórczego

Inne modele (nie klasyczne)



Proces iteracyjny i przyrostowy

Literatura

- Literatura:
  - ogólna (lista, cytowania do wykładu)
  - *Hoffer, Sommerville, Szejko, Kruchten, (Jaszkiewicz)*

Podsumowanie

Koniec

Podsumowanie