Inżynieria oprogramowania

Skąd ta inżynieria?

- **Wcześniej**
  - Tylko małe programy,
  - Jedna osoba – pomysłodawcą, autorem i użytkownikiem,
- **Później**,
  - Chęć tworzenia większych rzeczy w krótszym czasie,
  - Nowsze języki umożliwiające tworzenie większych programów,
  - Konieczność pracy w grupie,
    - Komunikacja,
    - Dokumentacja,
    - Standaryzacja języków, notacji, narzędzi,
  - Nowe zagadnienia wynikające z tworzenie dla nowych klientów,
    - Zbieranie i analiza wymagań,
    - Projektowanie,
    - Interfejs użytkownika,

Przyczyny i początki ...

- Przyczyny powstania inżynierii oprogramowania
  - Rosnąca złożoność projektów informatycznych,
  - Brak standardów i narzędzi umożliwiających (ułatwiających) pracę w grupie programistycznej,
  - Nieumiejętność szacowania kosztów i czasochłonności projektów informatycznych.
  - Brak zrozumienia, że pracochłonność projektu nie rośnie liniowo ze wzrostem jego rozmiaru.
- Początki inżynierii oprogramowania datuje się na **lata sześćdziesiąte** począwszy od powstania pierwszych programów na rynek komercyjny,
  - Czas tzw. „**Kryzysu oprogramowania**”,
  - Czas gwałtownego rozwoju technologii,
  - Czas spadku cen i zwiększenia dostępności sprzętu komputerowego,

Historia programowania 1

- Programowanie liniowe
  - Wykonanie programu linia po linii,
  - Konieczność wykorzystania etykiet/numerowania linii kodu,
  - Konieczność wykorzystania instrukcji typu „goto”,
  - Bardzo trudny podział programu między programistów,
- Programowanie proceduralne
  - Możliwość wyznaczania bloków w kodzie źródłowym
    - Wcięcia, nawiasy klamrowe { }, konwencja begin/end,
  - Instrukcje sterujące
    - Instrukcje warunkowe,
    - Instrukcje powtórzenia/pętle,
  - Procedury i funkcje,
  - Możliwość dekompozycji kodu źródłowego,
  - Duża ilość zmiennych.

Historia programowania 2

- Programowanie strukturalne
  - Zmienne grupujące powiązane dane (struktury, rekordy),
  - Mała ilość zmiennych,
    - Ułatwione przekazywanie argumentów
  - Rozdział pomiędzy danymi i funkcjami,
- Programowanie obiektowe
  - Powiązanie danych i operacji na nich wykonywanych,
  - Abstrakcja - dostosowanie ilości składowych i metod do kontekstu,
  - Enkapsulacja – ukrywanie danych i operacji,
  - Dziedziczenie – rozszerzanie funkcjonalności,
  - Polimorfizm – wielopostaciowość, elastyczność programowania,
- Programowanie komponentowe
  - Ponowne wykorzystanie kodu,
  - Szybkość tworzenia aplikacji,
  - Zaawansowane narzędzia projektowe.

Języki programowania

- Liniowe
  - Basic – wczesne wersje i dialekty,
- Proceduralne
  - Bash
- Strukturalne
  - C
  - Ada
  - C++, PHP, Python, Swift – możliwość programowania strukturalnego,
- Obiektowe
  - C++, PHP, Python, Swift – możliwość programowania obiektowego,
  - Java, C# – języki zorientowane obiektowo,
- Środowiska komponentowe

Co to jest inżynieria oprogramowania?

- Jest to **dziedzina inżynierii**, która obejmuje **wszystkie aspekty tworzenia oprogramowania** od fazy początkowej do jego pielęgnacji/utrzymania,

- Inżynieria oprogramowania zajmuje się **teorią, metodami i narzędziami** związanymi z **wytwarzaniem oprogramowania**,

- **Ważne:**
  - Systematyzuje i porządkuje pracę nad oprogramowaniem,
  - Skutecznie zapewnia wysoką jakość oprogramowania,
  - Bardzo prężny rozwój przez ostatnie kilkadziesiąt lat,
  - Tworzenie oprogramowania to ważna gałąź gospodarki,

Oprogramowanie – co to jest?

- **Oprogramowanie są to programy komputerowe wraz z całym środowiskiem je otaczającym** (dokumentacja, konfiguracja, często nierozerwalne ze sprzętem),
- **„Oprogramowanie jest wszędzie”** – komputery, automatyka, sprzęt domowy, samochody,

- Rodzaje oprogramowania
  - Oprogramowanie **gotowe / powszechne**,
    - Skierowane do dużej liczby użytkowników,
    - Niska cena,
    - Dostępność „od ręki”,
  - Oprogramowanie na **zamówienie / dostosowane**,
    - Dostosowane do indywidualnych potrzeb,
    - Wysoka cena,
    - Długi czas przygotowania,

Proces tworzenia oprogramowania

- Zbiór czynności (etapów) i związanych z nimi wynikami cząstkowymi, które zmierzają do przygotowania gotowego produktu czyli oprogramowania,

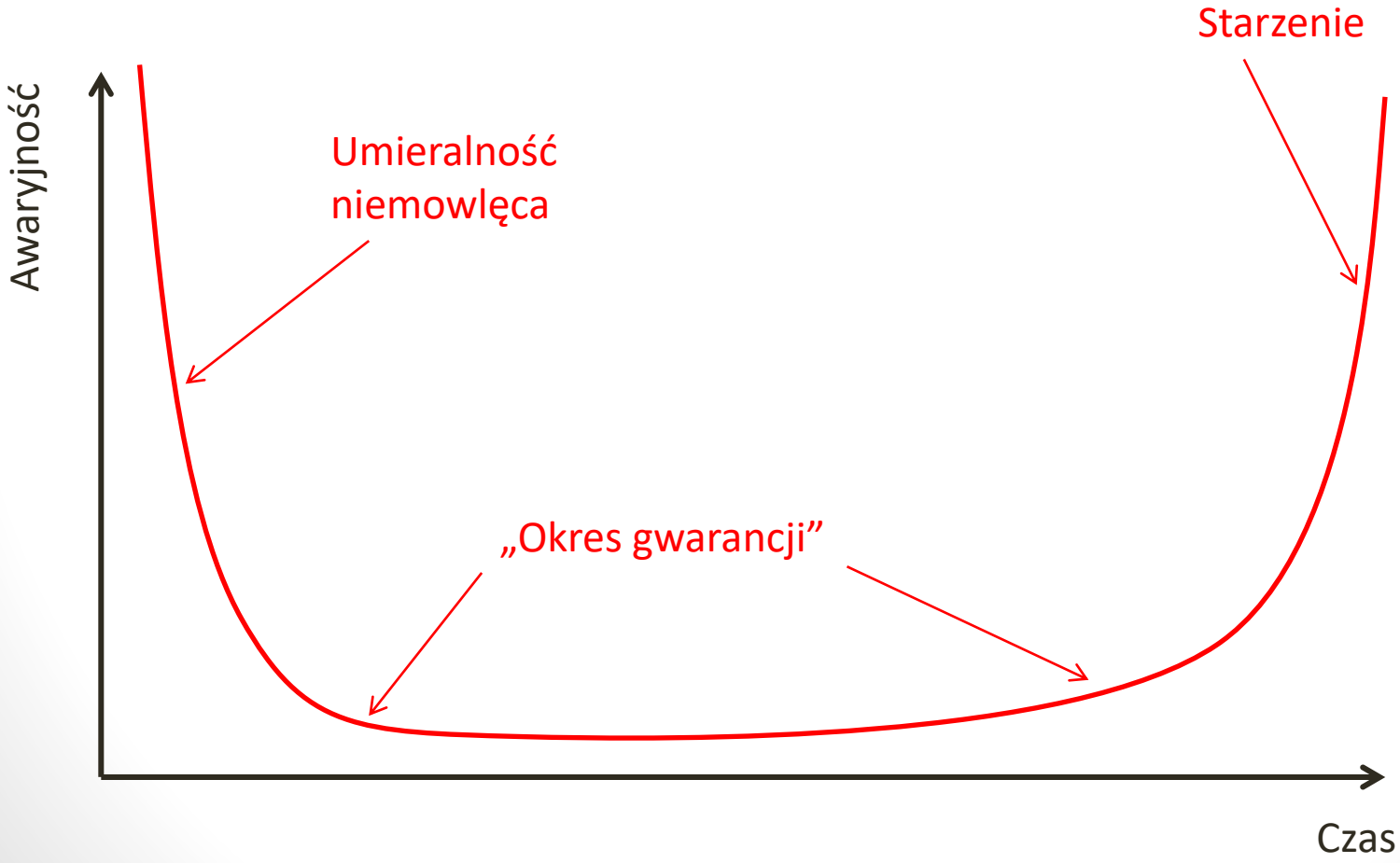
- Podstawowe czynności / etapy
  - Specyfikacja i analiza oprogramowania,
  - Projektowanie oprogramowania,
  - Tworzenie oprogramowania,
  - Utrzymanie i ewolucja oprogramowania,

Cechy dobrego oprogramowania

- **Utrzymywalność,**
  - Maksymalizacja czasu użytkowania,
  - Możliwość ewolucji zgodnie z potrzebami użytkownika,
- **Ufność / Niezawodność,**
  - Minimalizacja kosztów w przypadku wystąpienia awarii,
  - Oprogramowanie nie powinno powodować fizycznych i ekonomicznych strat w razie wystąpienia awarii,
- **Efektywność / Wydajność,**
  - Oprogramowania powinno racjonalnie wykorzystywać dostępne zasoby sprzętowe (pamięć, moc obliczeniową),
- **Użyteczność / Użytkowość,**
  - Przejrzysty i intuicyjny interfejs,
  - Dobra jakość dokumentacji,

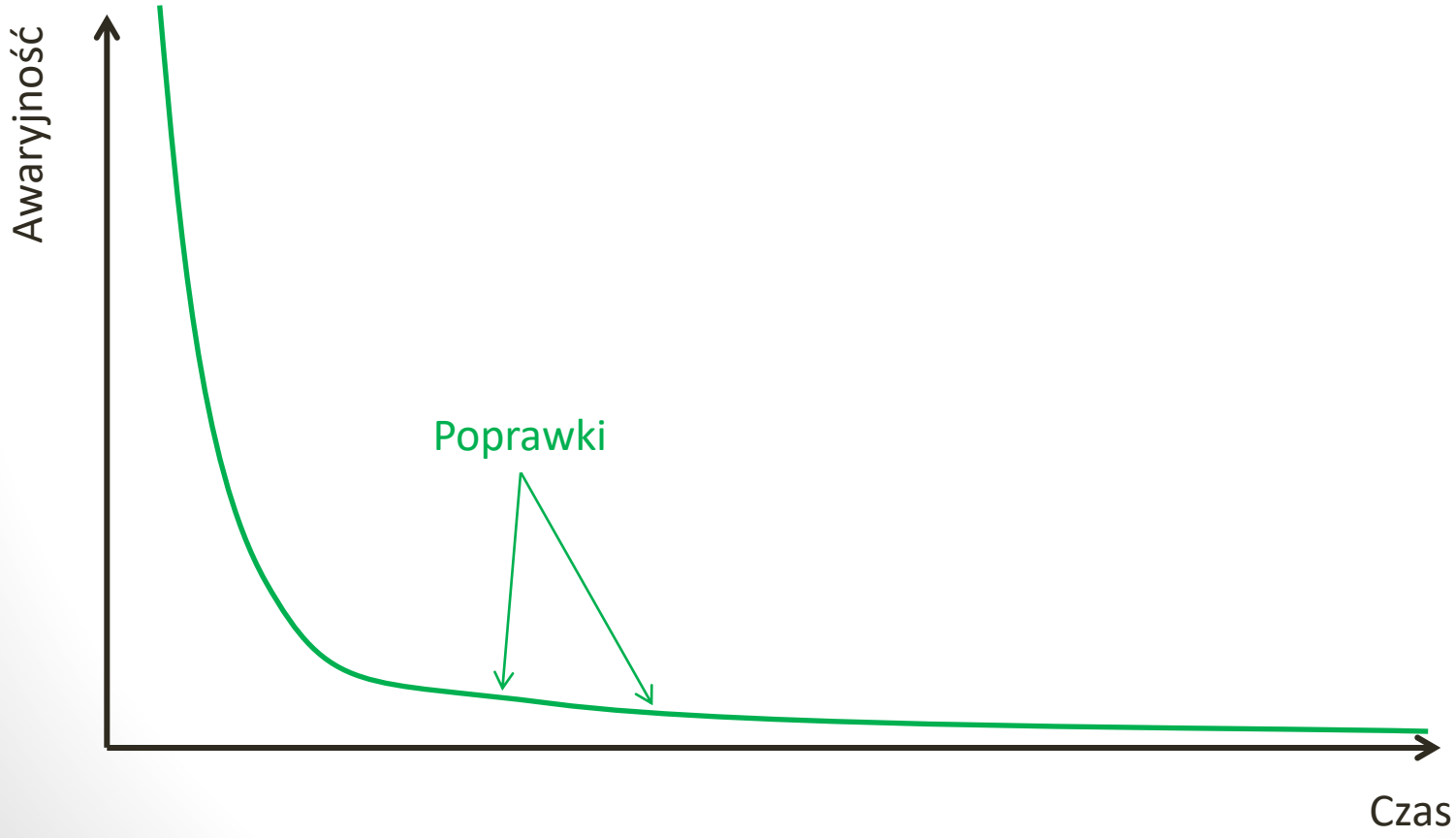
Czy programy się starzeją?

- Oprogramowanie się nie starzeje ale dezaktualizuje
- Krzywa starzenia dla **sprzętu**



Czy programy się starzeją?

- Oprogramowanie się nie starzeje ale dezaktualizuje
- Krzywa starzenia dla oprogramowania - **idealna**



Czy programy się starzeją?

- Oprogramowanie się nie starzeje ale dezaktualizuje
- Krzywa starzenia dla oprogramowania - **realna**,