

# Projektowanie Systemów Informatycznych

**Inżynieria oprogramowania** dotyczy stosowania empirycznego, naukowego podejścia do znajdowania wydajnych i ekonomicznych rozwiązań praktycznych problemów w dziedzinie oprogramowania.

**Projektowanie systemów informatycznych** to proces definiowania architektury, komponentów, modułów, interfejsów i danych tak, aby system spełniał określone wymagania użytkownika końcowego. Celem jest stworzenie dobrze zorganizowanej struktury, która spełnia zamierzony cel, jednocześnie biorąc pod uwagę takie czynniki, jak skalowalność, łatwość utrzymania i wydajność.

## Proces tworzenia oprogramowania

Każde oprogramowanie wytwarza się w czterech etapach:

**Planowanie → Analiza wymagań → Projektowanie → Implementacja**

### Etap 1. Planowanie (identyfikacja celów)

Odpowiedź na pytania: **po co** tworzymy to oprogramowanie i **jak** ono powstanie. Ustala się wartości biznesowe projektu, wykonalność wdrożenia oraz dostępne zasoby.

### Etap 2. Analiza wymagań (określenie funkcji systemu)

Ustala się: **kto** będzie używał systemu, **co** system ma robić, **gdzie i kiedy** będzie używany. System opisywany jest jako czarna skrzynka (black box), bez narzucania konkretnych technologii.

### Etap 3. Projektowanie (architektura i struktura danych)

Pierwsza decyzja o tym, jak system ma działać i jak ma spełniać wymagania użytkownika. Tu powstaje koncepcja wewnętrznej logiki systemu, jego architektura, interfejsy (w tym użytkownika).

Obecnie stosuje się metody obiektowe<sup>1</sup>, a projekt często stanowi abstrakcję przyszłego systemu: jest to najczęściej forma opisu słownego oraz wizualizacji (rysunki, schematy, diagramy).

### Etap 4. Implementacja (kodowanie)

Końcowa faza procesu tworzenia oprogramowania. Podczas implementacji zostaje zbudowana baza danych w wybranej technologii implementacyjnej i wytworzony kod aplikacji.

(tutaj się kończy samo wytwarzanie oprogramowania, ale zgodnie z fazami Cyklu życia systemu informatycznego, po implementacji dochodzi: Testowanie, Wdrożenie, Eksploatacja, Wycofanie. Czyli po wykonaniu testów systemu można wdrożyć system w warunkach rzeczywistych i sukcesywnie utrzymywać go w ciągłej pracy, aż do jego finalnego wycofania).

<sup>1</sup> Programowanie strukturalne a obiektowe. Programowanie strukturalne skupia się na pisaniu funkcji. W programowaniu obiektowym najważniejsze są dane.

# Cykl życia systemu informatycznego

---

## Etap 1. Planowanie (identyfikacja celów)

Odpowiedź na pytania: **po co** tworzymy to oprogramowanie i **jak** ono powstanie.  
Ustala się wartości biznesowe projektu, wykonalność wdrożenia oraz dostępne zasoby.

**Przykład:** Firma planuje stworzyć system zarządzania klientami CRM, opisuje jego najważniejsze cechy i zespół zaangażowany w jego wdrożenie.

## Etap 2. Analiza wymagań (określenie funkcji systemu)

Ustala się: **kto** będzie używał systemu, **co** system ma robić, **gdzie i kiedy** będzie używany.  
System opisywany jest jako czarna skrzynka (black box), bez narzucania konkretnych technologii.

**Przykład:** Firma sprawdza, czy zbudowanie systemu CRM jest warte kosztów i wysiłku oraz czy może on przynieść oczekiwane korzyści i czy spełnia oczekiwania użytkowników rynku.

## Etap 3. Projektowanie (architektura i struktura danych)

Pierwsza decyzja o tym, jak system ma działać i jak ma spełniać wymagania użytkownika.  
Tu powstaje koncepcja wewnętrznej logiki systemu, jego architektura, interfejsy (w tym użytkownika).

**Przykład:** Powstaje szczegółowy plan dla deweloperów, jak ma działać i wyglądać system CRM - analogicznie jak plany architektoniczne przed budową domu.

## Etap 4. Implementacja (kodowanie)

Końcowa faza procesu tworzenia oprogramowania.  
Podczas implementacji zostaje zbudowana baza danych w wybranej technologii implementacyjnej i wytworzony kod aplikacji.

**Przykład:** Kodowane są funkcje systemu CRM: profile klientów, pulpity nawigacyjne itp.

## Etap 5. Testowanie

Weryfikacja, czy system spełnia określone wymagania i czy wszystko działa zgodnie z projektem.

**Przykład:** System CRM poddawany jest różnym testom, np. testy jednostkowe, testy integracyjne i testy akceptacji użytkownika, aby zapewnić jego funkcjonalność, wydajność i bezpieczeństwo.

## Etap 6. Wdrożenie (uruchomienie systemu na produkcji, deployment/release)

Przeniesienie systemu ze środowiska testowego do produkcyjnego i udostępnienie faktycznym użytkownikom.

## Etap 7. Eksploatacja i utrzymanie (konserwacja, monitorowanie, aktualizacje)

Regularne aktualizacje, poprawki błędów i wsparcie użytkowników w dostosowywaniu systemu do zmieniających się wymagań biznesowych oraz w rozwiązywaniu pojawiających się problemów.

## Etap 8. Wycofanie

Zamknięcie systemu.

## Cykl życia systemu vs. Proces tworzenia systemu

Aspekt	Cykl życia systemu informatycznego	Proces tworzenia oprogramowania
<b>Definicja</b>	Kompleksowe ramy obejmujące cały proces rozwoju systemu.	Podzbiór zajmujący się konkretnie projektowaniem i wykonaniem systemu
<b>Zakres</b>	Cały cykl życia: od planowania, przez uruchomienie, do wycofania.	Przed wszystkim aspekty projektowe systemu.
<b>Fazy</b>	Planowanie, analiza, projektowanie, implementacja, testowanie, wdrożenie, eksploatacja, wycofanie.	Planowanie, analiza wymagań, projektowanie, implementacja.
<b>Centrum</b>	Całościowy proces rozwoju: planowanie, wdrażanie, testowanie, konserwacja.	Etap projektowania: szczegółowy opis budowy i działania systemu.
<b>Zamiar</b>	Prowadzi zespół przez cały proces: od koncepcji po wsparcie powdrożeniowe.	Plan budowy systemu w oparciu o określone wymagania projektowe.

### Typowe wyzwania projektowe

- Niejasne lub niejednoznaczne wymagania na etapie startu projektu.
- Zmieniające się wymagania w trakcie procesu projektowania.
- Szybki postęp technologiczny utrudniający dobór właściwych narzędzi.
- Złożona integracja komponentów z różnych technologii i platform.
- Ograniczenia budżetowe utrudniające pełną realizację wymagań.

## Cykl życia procesu analizy danych w Data Science

Projektowanie nowoczesnych systemów informatycznych coraz częściej obejmuje wsparcie analizy danych i procesów decyzyjnych opartych na algorytmach data science.

- 1. Zdefiniuj cel** - *Jaki problem staram się rozwiązać?*
- 2. Zgromadź dane i zarządzaj nimi** - *Jakie informacje są mi potrzebne?*
- 3. Zbuduj model** - *Znajdź w danych wzorce prowadzące do rozwiązań.*
- 4. Oceń model i poddaj go krytyce** - *Czy model rozwiązuje mój problem?*
- 5. Zaprezentuj wyniki i udokumentuj je** - *Udowodnij, że możesz rozwiązać problem i pokaż, jak tego dokonasz.*
- 6. Wdróż model** - *Wdróż model w środowisku produkcyjnym i utrzymuj go.*

## Bibliografia

- Farley D., Nowoczesna inżynieria oprogramowania. Stosowanie skutecznych technik szybszego rozwoju oprogramowania wyższej jakości, Helion 2023
- Śmiałek M., Rybiński K., Inżynieria oprogramowania w praktyce. Od wymagań do kodu z językiem UML, Helion 2024
- Wrycza S., Maślankowski J., Informatyka ekonomiczna. Teoria i zastosowania, PWN 2019
- Zumel N., Mount J., Język R i analiza danych w praktyce, Helion 2021