



dr inż. Michał Malinowski
michal.malinowski@warszawa.merito.pl

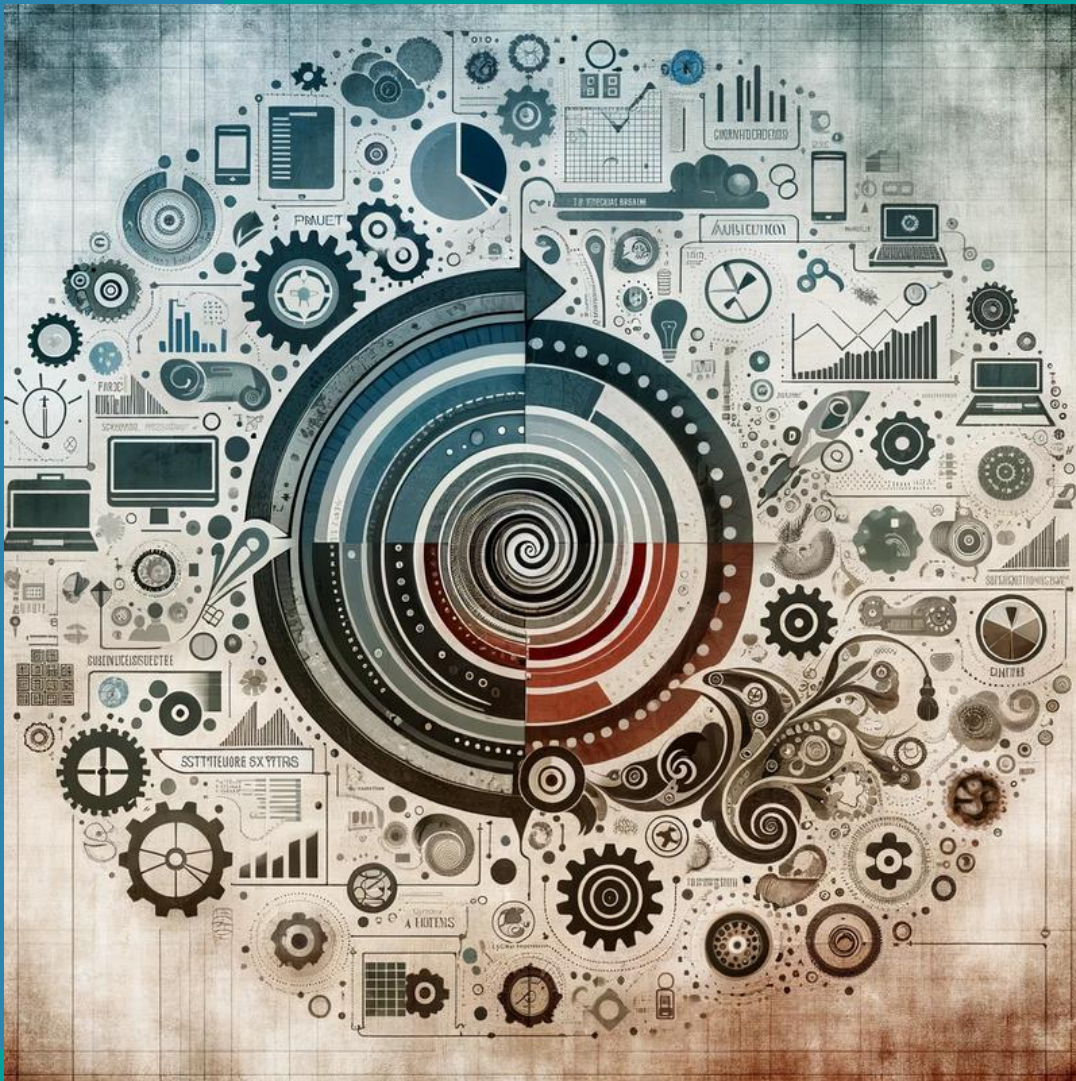


[https://www.linkedin.com/in/
michał-malinowski-malkon/](https://www.linkedin.com/in/michał-malinowski-malkon/)

Wprowadzenie do projektowania systemów informatycznych

Wykład 01

PROJEKTOWANIE SYSTEMÓW
INFORMATYCZNYCH



Zagadnienia (2h)

- [Projekt vs Proces](#)
- [Cykl życia oprogramowania](#)
- [Narzędzia CASE](#)
- [System informacyjny i informatyczny, oprogramowanie, aplikacja](#)
- [Rola systemów informatycznych](#)
- [Projektowania Architektury Oprogramowania](#)
- [Style programowania](#)

Projekt vs Proces

Projekt

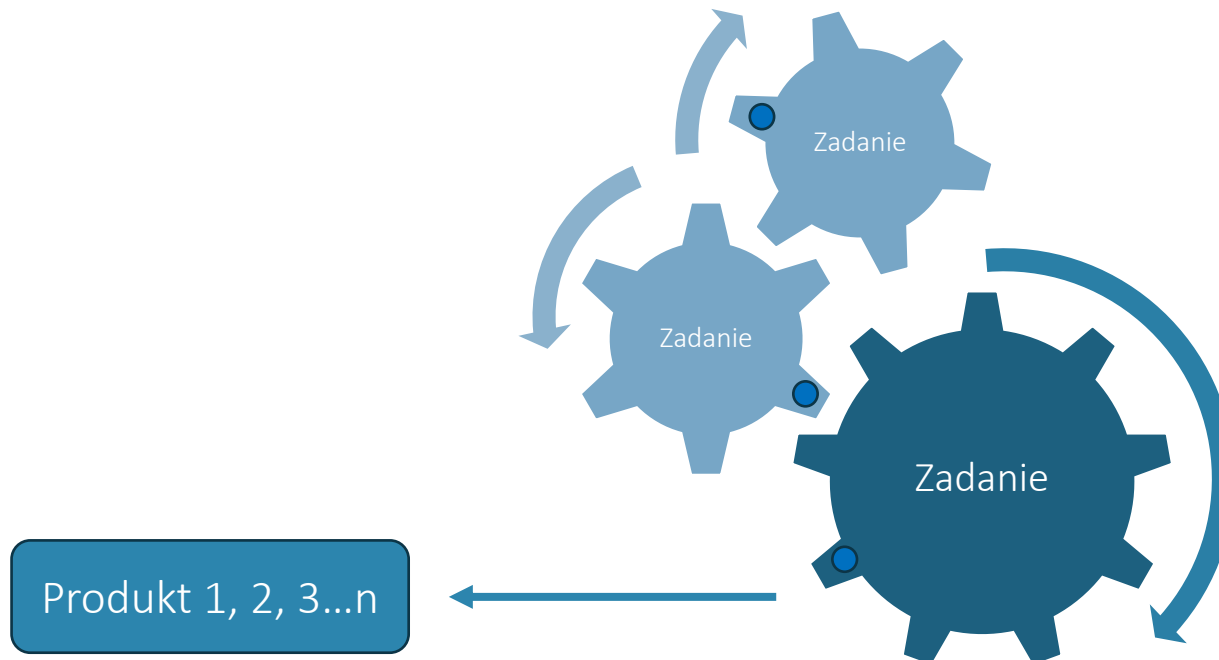
to zestaw unikatowych działań i zadań, mających na celu osiągnięcie określonych wyników (produktu) w określonym czasie i budżecie.



Projekt vs Proces

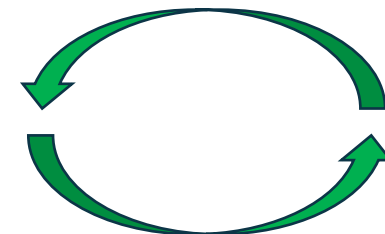
Proces

to sekwencja wzajemnie powiązanych zadań, które są wykonywane w celu osiągnięcia konkretnych produktów, rezultatów lub utrzymania usług.



Projekt vs Proces

Kryterium	Projekt	Proces
Czas trwania	<u>Ograniczony</u> , z jasno zdefiniowanym początkiem i końcem.	<u>Ciągły</u> , bez wyraźnie określonego końca.
Cel	Tworzenie nowego produktu, usługi lub wprowadzanie zmian.	Utrzymanie ciągłości działalności, efektywność i optymalizacja.
Wyniki	Konkretny produkt, usługa lub rezultat.	Efektywność i skuteczność wykonywania czynności ciągłych.
Przykłady	Rozwój nowego oprogramowania, budowa budynku, kampania marketingowa.	Procedury zakupowe, proces obsługi klienta, cykliczne raportowanie finansowe, monitorowanie systemów.



Projekt vs Proces

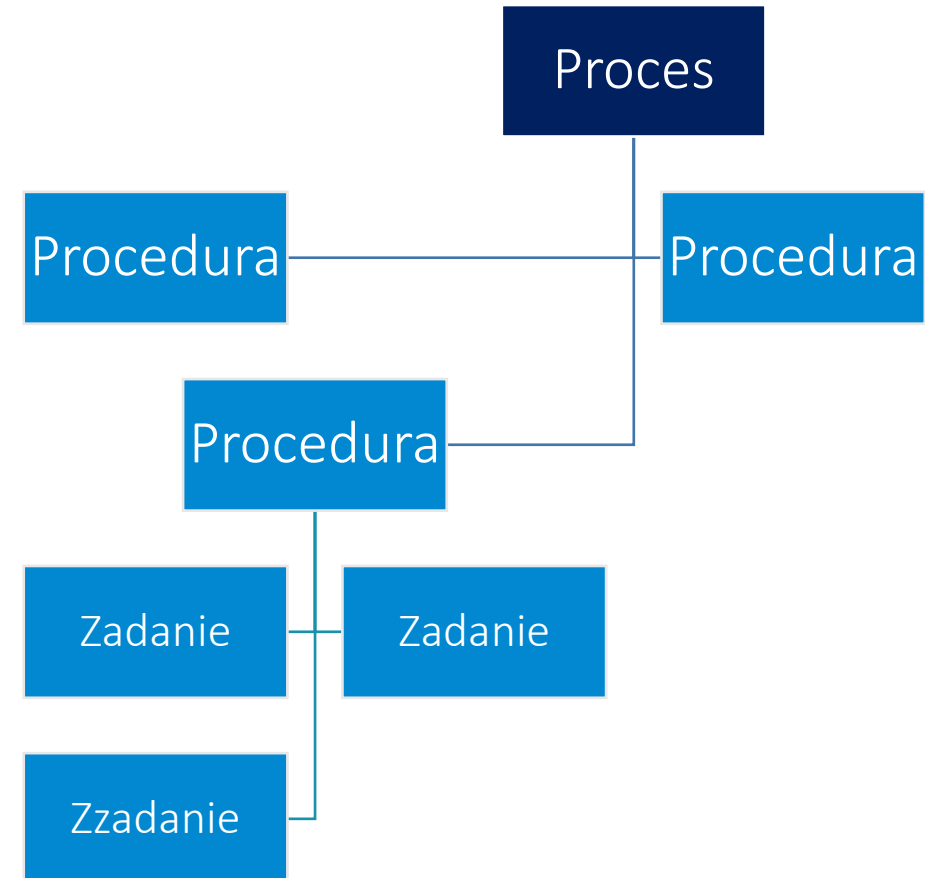
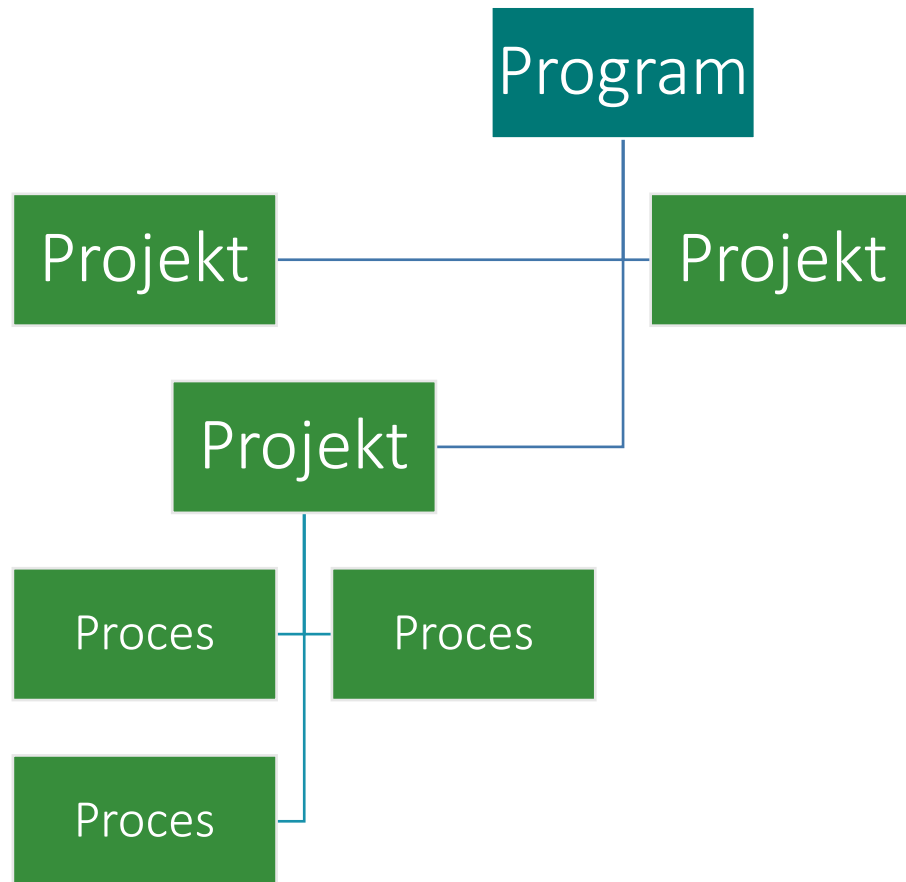
Projekt to tymczasowe przedsięwzięcie, które ma na celu stworzenie unikalnego produktu, usługi lub rezultatu. Projekty mają określony początek i koniec, oraz są zazwyczaj wykonywane w celu osiągnięcia konkretnych celów, które przynoszą zmiany lub dodatkową wartość.

Program w PRINCE2 to zbiór powiązanych projektów i działań zmian, które są zarządzane w sposób koordynowany. Programy są tworzone w celu dostarczenia strategicznych korzyści dla organizacji, które nie mogą być osiągnięte przez pojedyncze projekty działające niezależnie.

Proces to zbiór powiązanych zadań i czynności, które są wykonywane w celu osiągnięcia określonego rezultatu. Procesy są zazwyczaj stałymi, powtarzalnymi czynnościami w organizacji, które można standardyzować i optymalizować. Są one częścią codziennej działalności firmy i mają charakter ciągły.

Procedura to formalny, zdefiniowany zestaw kroków lub działań, które należy podjąć w celu wykonania określonego zadania lub procesu. Procedury są kluczowym elementem zarządzania operacyjnego w każdej organizacji, ponieważ zapewniają standardowe metody wykonywania powtarzalnych zadań, co przyczynia się do zwiększenia efektywności, spójności i przewidywalności działań.

Projekt vs Proces



Cykl życia oprogramowania

Inżynieria oprogramowania

to usystematyzowany, zorganizowany w czasie zbiór działań prowadzący do wytworzenia systemu informatycznego w szczególności oprogramowania w ramach projektu.

Produktem projektu informatycznego

jest system informatyczny w szczególności oprogramowanie

Inżynieria oprogramowania

jest dziedziną inżynierii

```
Inżynieria
├─ Inżynieria Mechaniczna
├─ Inżynieria Elektryczna
│   └─ Inżynieria Elektroniczna
├─ Inżynieria Informatyczna
│   └─ Inżynieria Oprogramowania
├─ Inżynieria Chemiczna
├─ Inżynieria Cywilna
├─ Inżynieria Biomedyczna
├─ Inżynieria Środowiskowa
├─ Inżynieria Materiałowa
├─ Inżynieria Energetyczna
├─ Inżynieria Lotnicza i Kosmiczna
└─ Inżynieria Automatyki i Robotyki
```

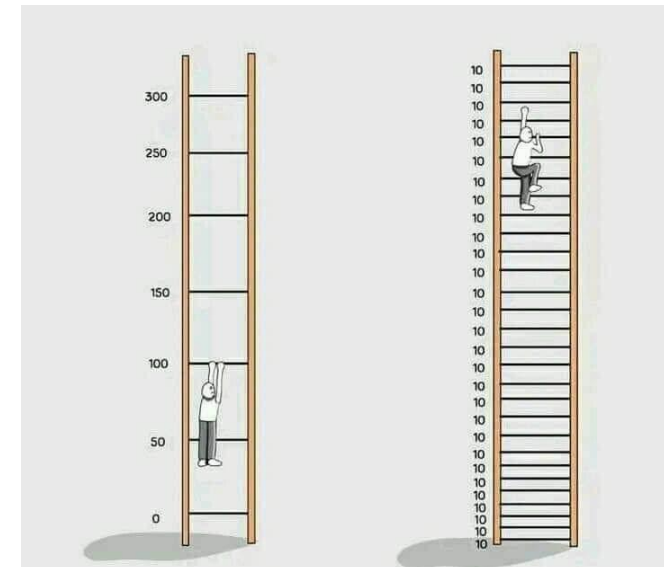
Cykl życia oprogramowania

Cykl Życia Oprogramowania (SDLC)

Sekwencja etapów, przez które przechodzi oprogramowanie od momentu jego koncepcji do wycofania z użycia.

Dlaczego to jest ważne?

Zarządzanie cyklem życia oprogramowania jest kluczowe dla realizacji celów projektu, takich jak jakość, efektywność i zgodność z wymogami.



Cykl życia oprogramowania

Kluczowe Etapy

Analiza Wymagań

Zrozumienie i dokumentacja potrzeb użytkowników.

Projektowanie

Określenie architektury i specyfikacji systemu.

Implementacja

Faza kodowania i testowania jednostkowego.

Testowanie

Weryfikacja i walidacja funkcjonalności systemu.

Wdrożenie

Instalacja i konfiguracja systemu w środowisku docelowym.

Utrzymanie

Aktualizacje, naprawy i wsparcie po wdrożeniu.

Cykl życia oprogramowania

Metodologie Cyklu Życia Oprogramowania

- Model Kaskadowy: Sekwencyjny proces, w którym każdy etap zależy od poprzedniego.
- Model Spiralny: Iteracyjny model, który łączy elementy modelu kaskadowego i prototypowania.
- Model Zwinny: zespoły pracują w krótkich cyklach, zwanych sprintami. W każdym sprincie zespół oprogramowania koncentruje się na ukończeniu określonego zestawu zadań.
- DevOps: to metodologia integrująca procesy rozwoju i operacji oprogramowania w celu przyspieszenia dostarczania aplikacji i usprawnienia ciągłej integracji oraz dostarczania.

Narzędzia CASE

CASE (Computer-Aided Software Engineering)

to zbiorcza nazwa oprogramowania, używanego do komputerowego wspomagania projektowania oprogramowania

Funkcje i Zastosowania

Automatyzacja Analizy Wymagań

Narzędzia do zbierania, analizy i zarządzania wymaganiami.

Projektowanie Architektury

Narzędzia do tworzenia diagramów UML, ERD, BPMN i innych modeli.

Generowanie Kodu

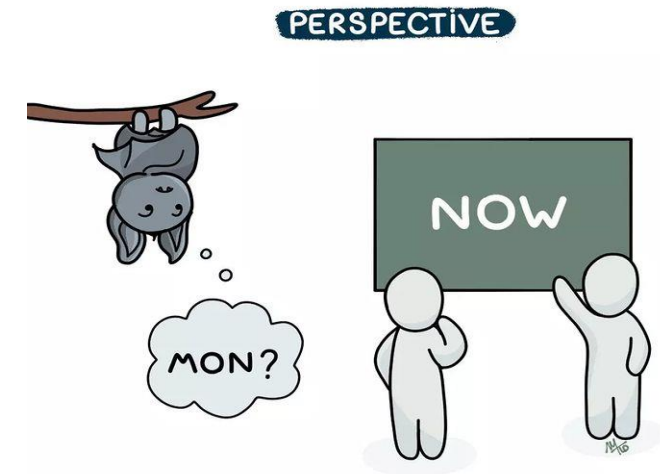
Automatyczne generowanie kodu na podstawie modeli i diagramów.

Testowanie i Debugowanie

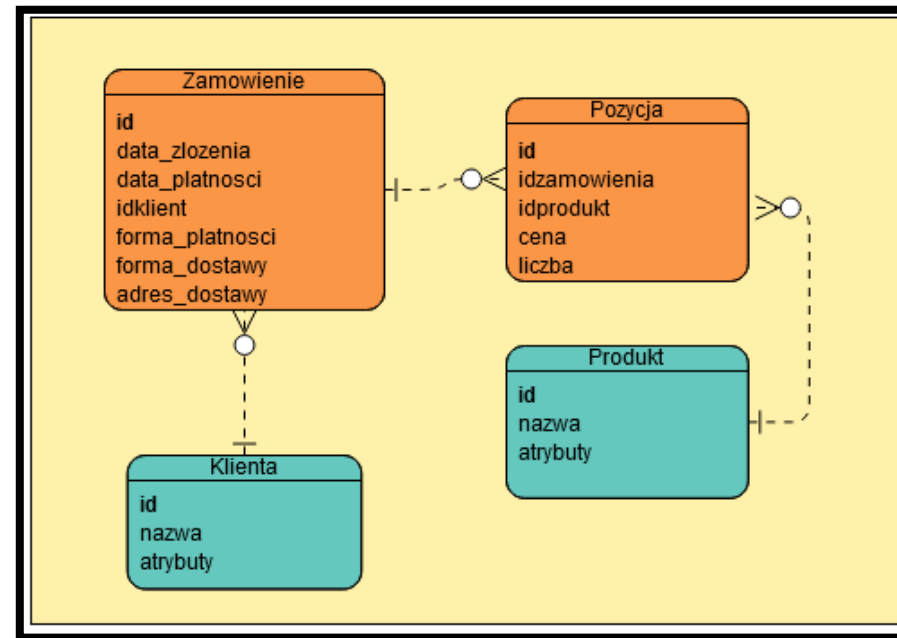
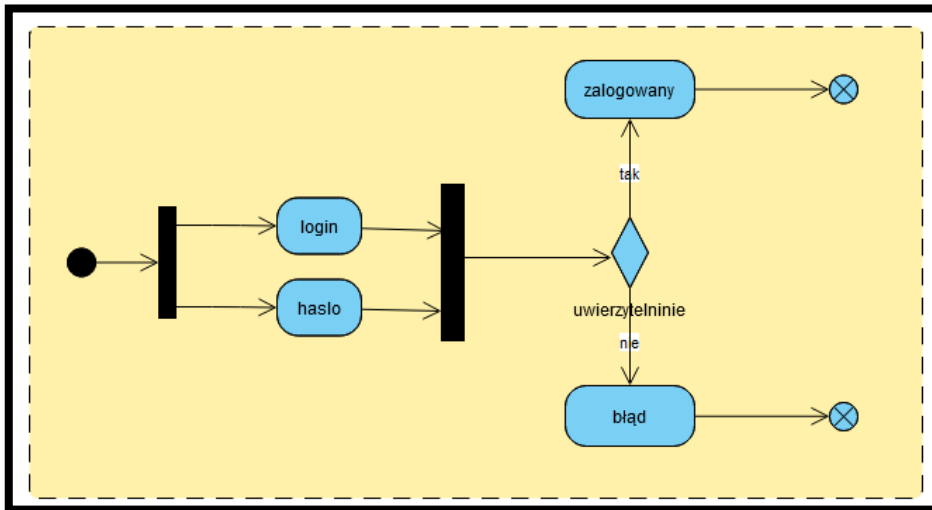
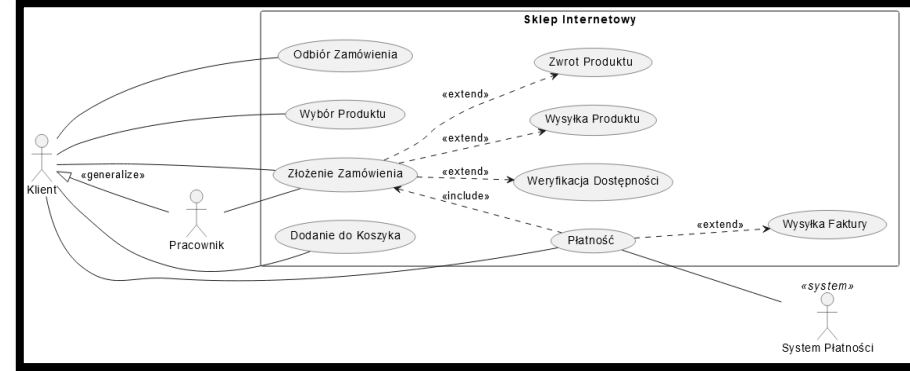
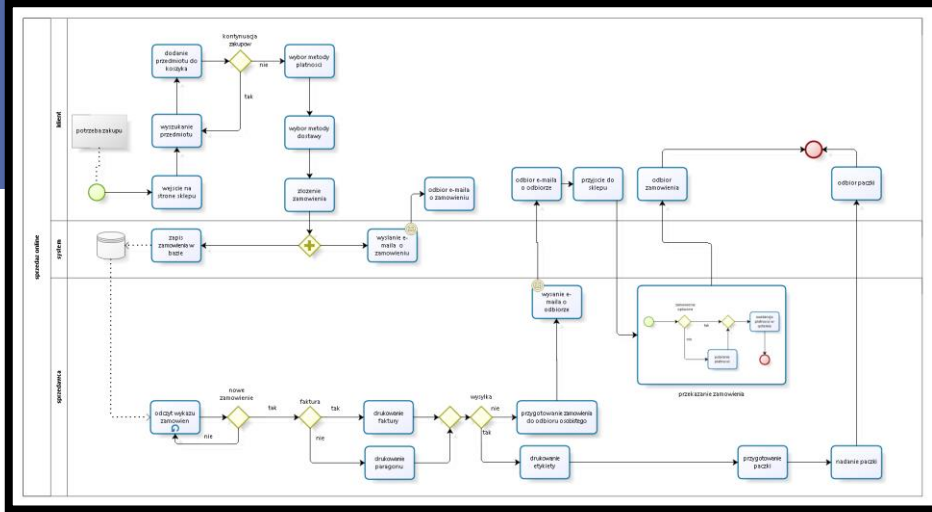
Narzędzia do automatycznego testowania i debugowania kodu.

Zarządzanie Projektem

Narzędzia do planowania, śledzenia i zarządzania zasobami i czasem.



@melodie_tld



Narzędzia CASE

Typowe moduły narzędzi CASE

Słownik Danych (Repozytorium)

baza danych o tworzonym systemie wraz z narzędziami edytującymi, zarządzającymi i wyszukującymi te dane.

Edytor Notacji Graficznych

program graficzny, umożliwiający tworzenie i edycję diagramów dla faz określania wymagań systemu, analizy i projektowania.

Moduł Kontroli Poprawności

narzędzie do wykrywania i poprawiania błędów w diagramach i repozytoriach.

Moduł Kontroli Jakości

narzędzie do oceny pewnych ustalonych miar jakości projektu – np. stopnia złożoności lub powiązań składowych modelu.

Generator Raportów

narzędzie tworzące dowolny raport na podstawie danych z repozytorium.

Generator Kodu

narzędzie transformujące projekt na szkielet kodu w wybranym języku programowania.

Generator Dokumentacji Technicznej

generator dokumentów zawierających specyfikację, opisy faz projektu, diagramy oraz wybrane raporty.

Moduł Projektowania Interfejsu Użytkownika

narzędzie do projektowania menu, okien dialogowych oraz innych elementów interfejsu użytkownika.

Moduł Inżynierii Odwrotnej

narzędzie pozwalające odtworzyć słownik danych lub/ oraz diagramy na podstawie kodu źródłowego lub struktury bazy danych.

Moduł Importu/Eksportu Danych

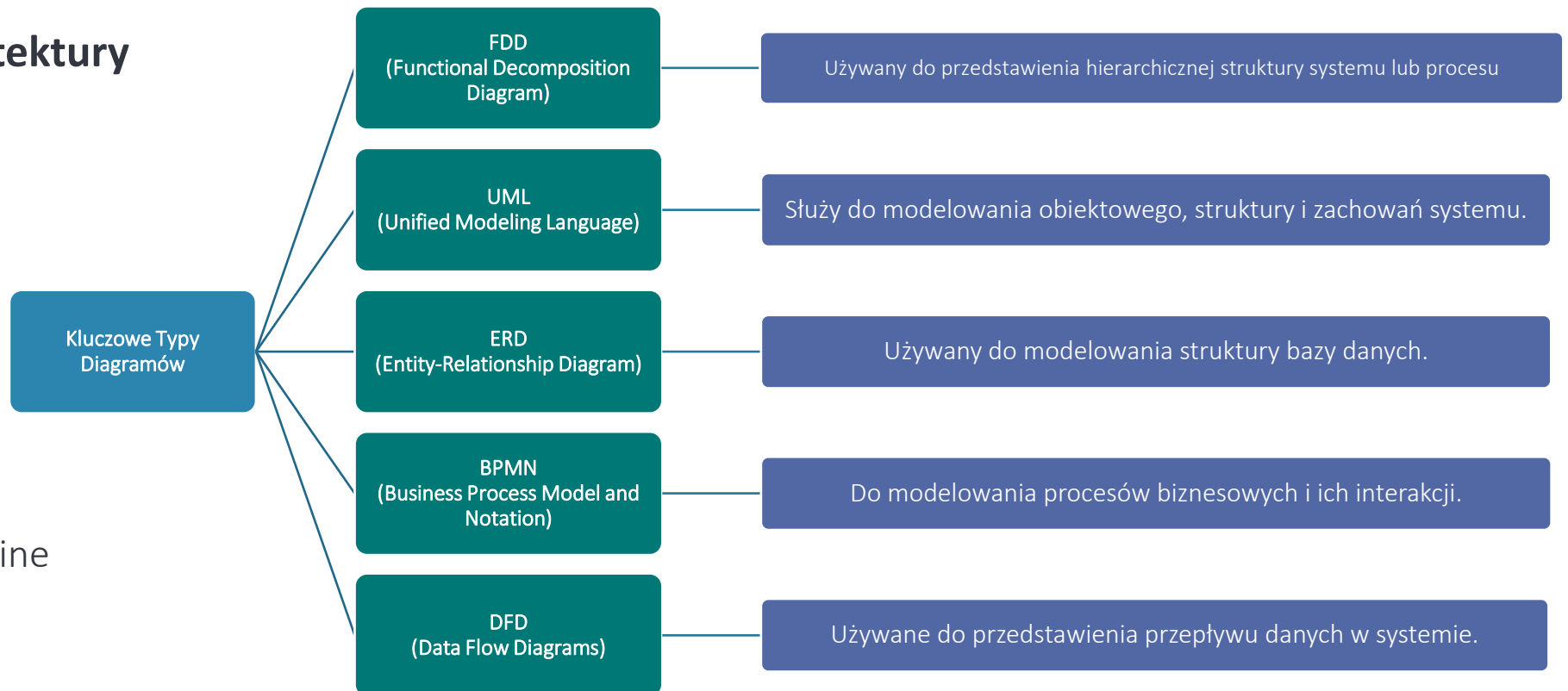
narzędzie służące do wymiany danych z innymi narzędziami CASE lub z innymi programami.

Moduł Zarządzania Pracą Grupową

narzędzie umożliwiające współpracę grupy osób podczas pracy nad projektem.

Narzędzia CASE

Projektowanie Architektury



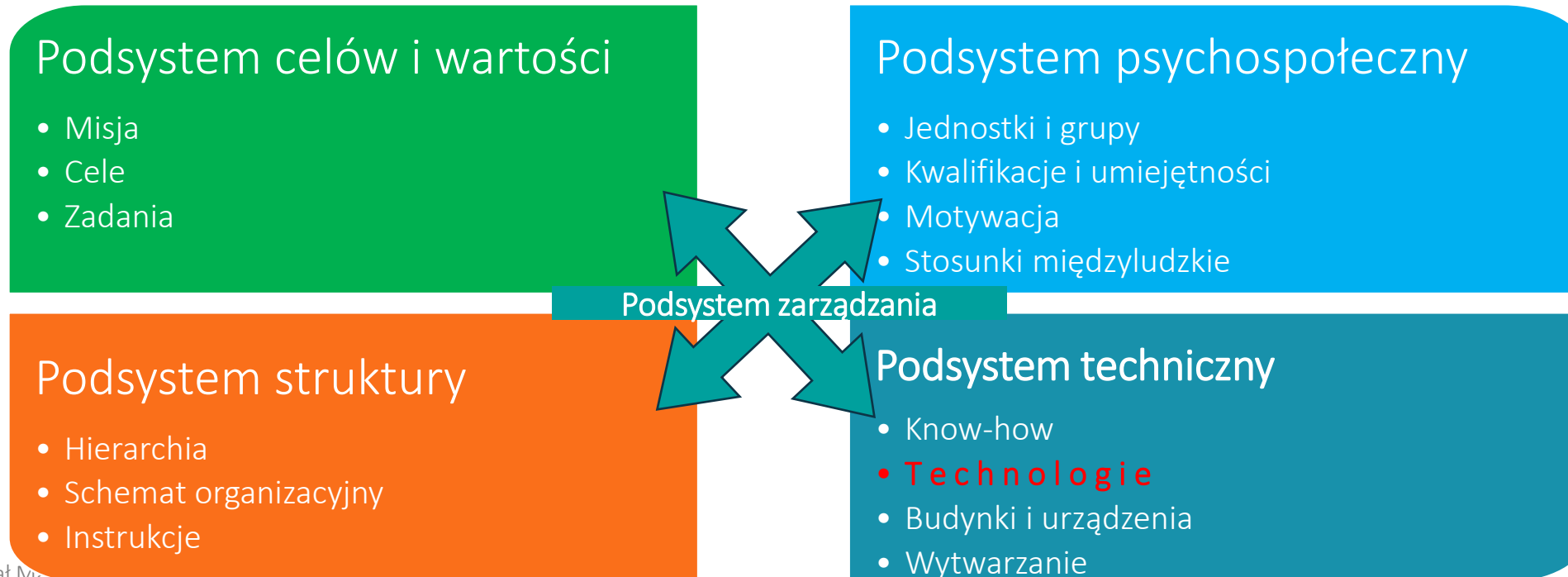
Popularne narzędzia

- Visual Paradigm Online
- Lucidchart
- Draw.io
- Enterprise Architect
- PowerPoint

System informacyjny i informatyczny, oprogramowanie, aplikacja

Organizacja

Ustrukturalizowany, czyli uporządkowany w pewien sposób system społeczno-techniczny złożony z pewnych podsystemów:



System informacyjny i informatyczny, oprogramowanie, aplikacja

System informacyjny

Układ odpowiednich elementów, którego zadaniem jest przetwarzanie danych.

Funkcje systemu informacyjnego

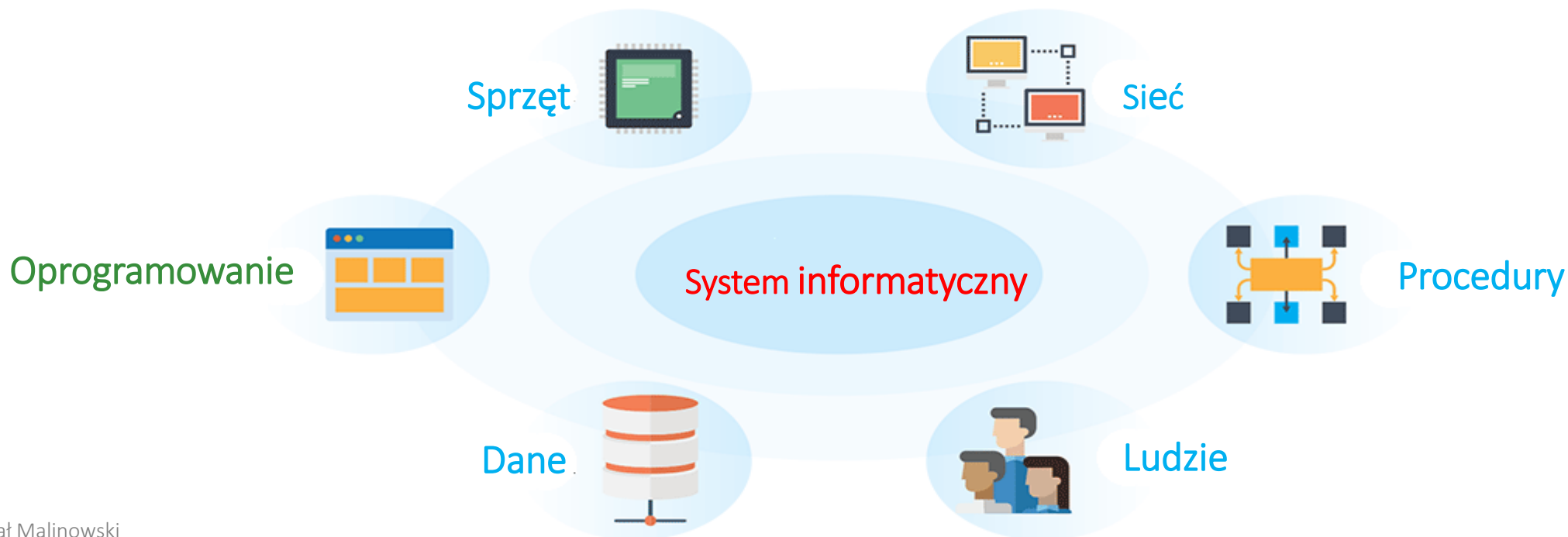
- gromadzenie informacji,
- przetwarzanie informacji,
- przechowywanie informacji,
- prezentacja informacji,
- przesyłanie informacji,



System informacyjny i informatyczny, oprogramowanie, aplikacja

System informatyczny

Wyodrębniona **część systemu informacyjnego**, w którym do przetwarzania danych zastosowano środki i metody informatyczne, a zwłaszcza sprzęt i oprogramowanie komputerów



System informacyjny i informatyczny, oprogramowanie, aplikacja

Oprogramowanie

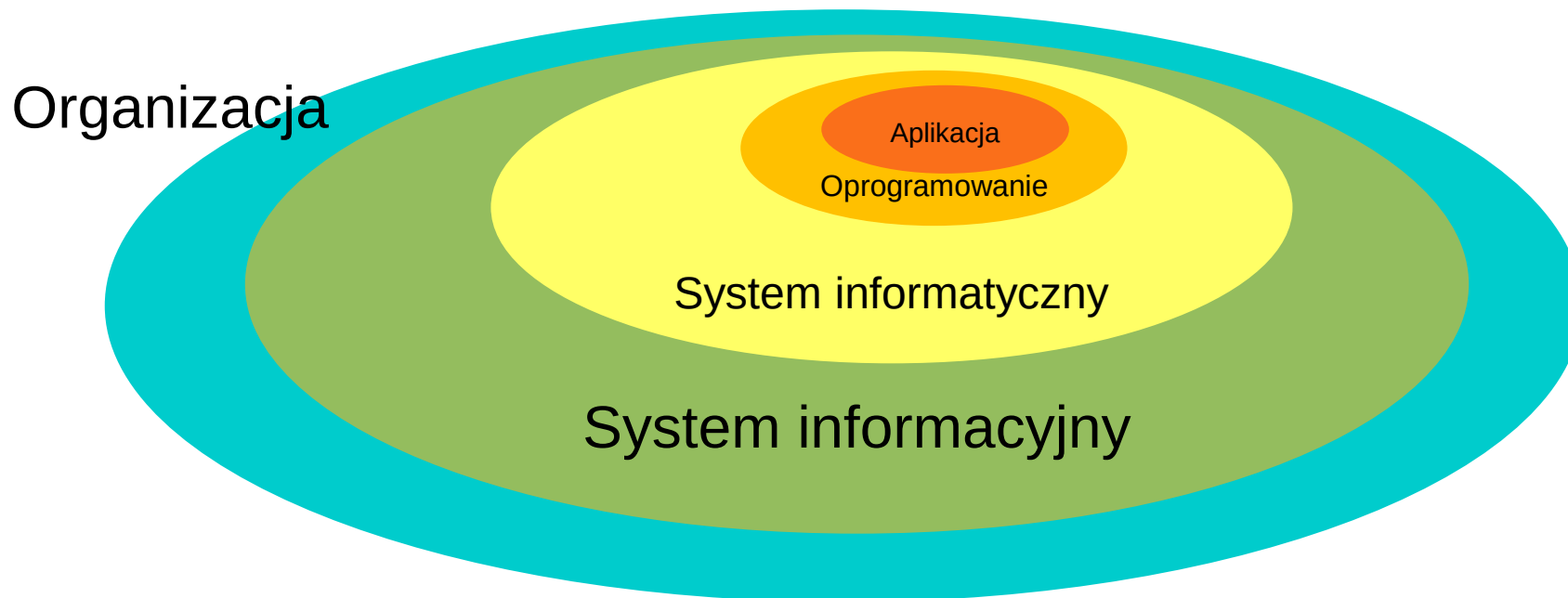
To zbiór instrukcji, które, gdy są wykonywane przez procesor komputera, powodują wykonanie określonej sekwencji operacji. Te instrukcje są zwykle napisane w językach programowania i mogą być interpretowane lub kompilowane do postaci, którą maszyna może bezpośrednio wykonać.



System informacyjny i informatyczny, oprogramowanie, aplikacja

Aplikacja

Aplikacja to rodzaj oprogramowania zaprojektowanego do wykonywania konkretnej grupy zadań lub rozwiązywania określonych problemów użytkownika.



Rola systemów informatycznych

Kluczowe funkcje SI w organizacji

Automatyzacja Procesów

Zmniejszenie czasu i zasobów potrzebnych na rutynowe zadania, co prowadzi do zwiększenia efektywności operacyjnej.

Zarządzanie Danymi i Informacją

Efektywne gromadzenie, przechowywanie i analiza danych, co umożliwia lepsze zrozumienie rynku i podejmowanie informowanych decyzji.

Komunikacja i Współpraca

Ułatwienie komunikacji między pracownikami, działami i z partnerami zewnętrznymi, co zwiększa efektywność i jakość współpracy.

Wsparcie dla Podejmowania Decyzji

Dostarczanie narzędzi analitycznych i raportów, które ułatwiają zarządzanie i podejmowanie strategicznych decyzji w organizacji.

Bezpieczeństwo i Kontrola

Zabezpieczenie danych i informacji przed nieautoryzowanym dostępem, utratą czy naruszeniem, a także monitorowanie i audyty dla utrzymania zgodności z regulacjami i standardami.

Rola systemów informatycznych

Wpływ SI na organizację

Zwiększenie Efektywności

Zwiększenie produktywności i redukcja kosztów poprzez efektywniejsze wykorzystanie procesów i zasobów.

Wspomaganie podejmowania Decyzji

Dostęp do danych i analiz wspomaga procesy decyzyjne oraz lepsza kontrola i reakcja na potencjalne zagrożenia.

Konkurencyjność i Innowacyjność

Szybka adaptacja do zmian na rynku i innowacje poprzez wprowadzenie nowych produktów lub usług

Zarządzanie Wiedzą

Lepsze wykorzystanie wiedzy i umiejętności pracowników.

Rola systemów informatycznych

Wyzwania i zagrożenia

Bezpieczeństwo

Ryzyko
naruszeń
i ataków.

Zgodność i Regulacje

Należy
spełniać
wymagania
prawne
i standardy.

Koszty

Wysokie
koszty
wdrożenia
i utrzymania.

Rola systemów informatycznych

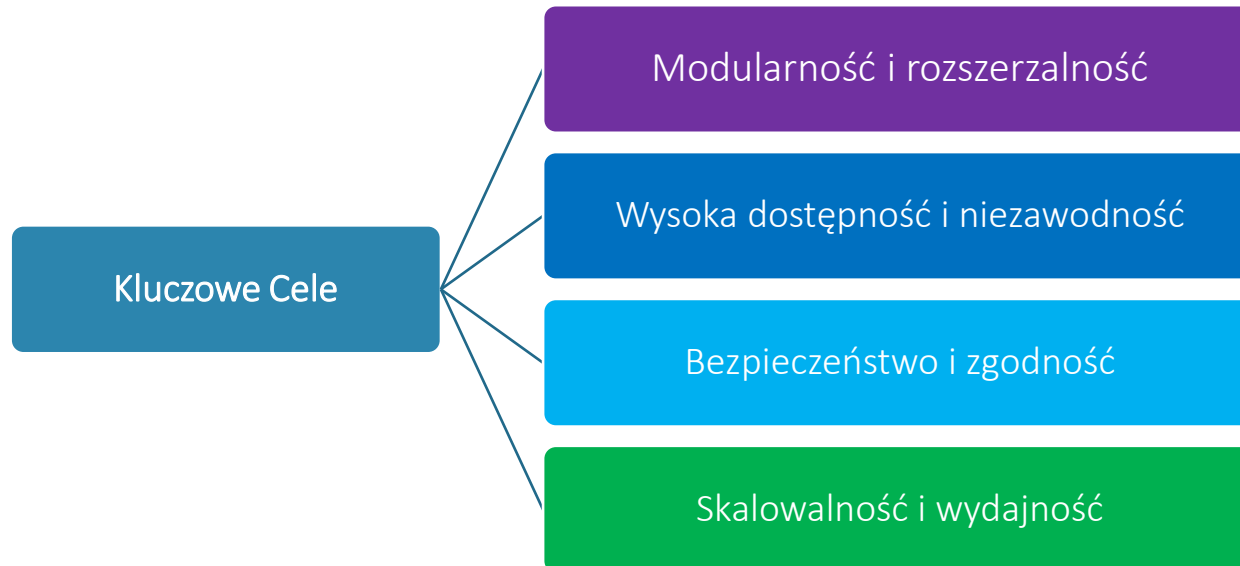
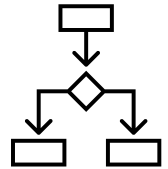
Standardy i Regulacje

- ISO 27001: międzynarodowy standard zarządzania bezpieczeństwem informacji.
- ISO 25010: międzynarodowy standard określający wymagania dla jakości oprogramowania.
- GDPR: europejskie rozporządzenie o ochronie danych osobowych.
- PCI DSS: standard bezpieczeństwa danych dla organizacji przetwarzających płatności kartą.
- CMMI: zbiór modeli dojrzałości dla procesów inżynierii oprogramowania.
- IEEE 828: standard określający wymagania dla dokumentacji oprogramowania.
- Ustawę z dnia 4 lutego 1994 r. o prawie autorskim i prawach pokrewnych
- Ustawa z dnia 18 lipca 2002 r. o świadczeniu usług drogą elektroniczną
- Ustawa z dnia 5 lipca 2018 r. o krajowym systemie cyberbezpieczeństwa
- Wewnętrzne procedury organizacji

Projektowania Architektury Oprogramowania

Projektowanie architektury oprogramowania

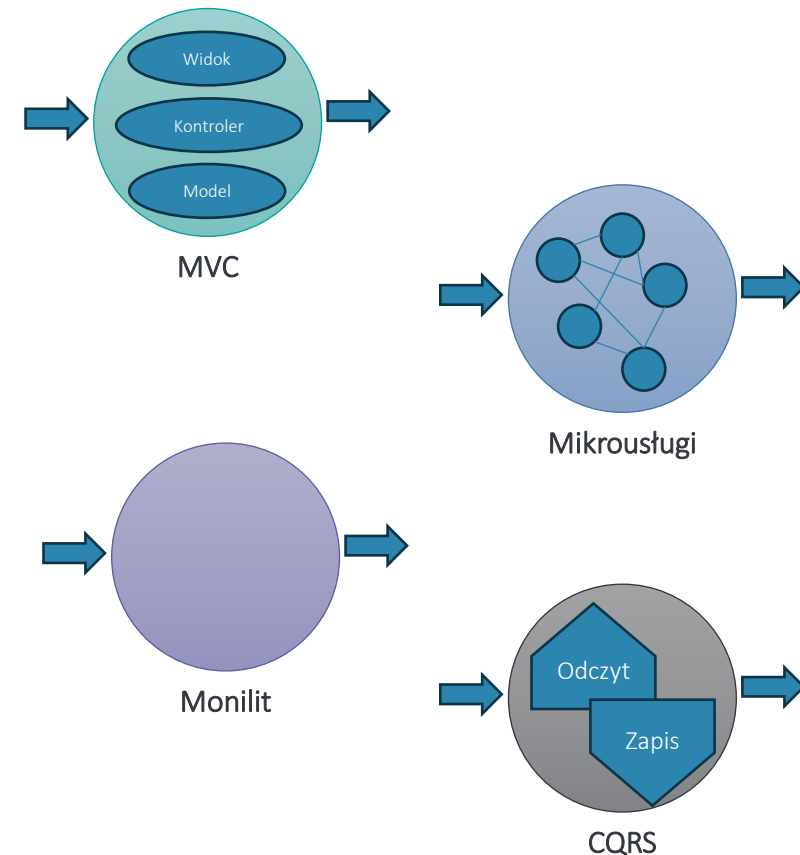
to proces definiowania strukturalnych elementów systemu, ich wzajemnych relacji oraz interakcji, aby osiągnąć określone cele jakościowe i funkcjonalne.



Projektowania Architektury Oprogramowania

Wzorce Architektoniczne

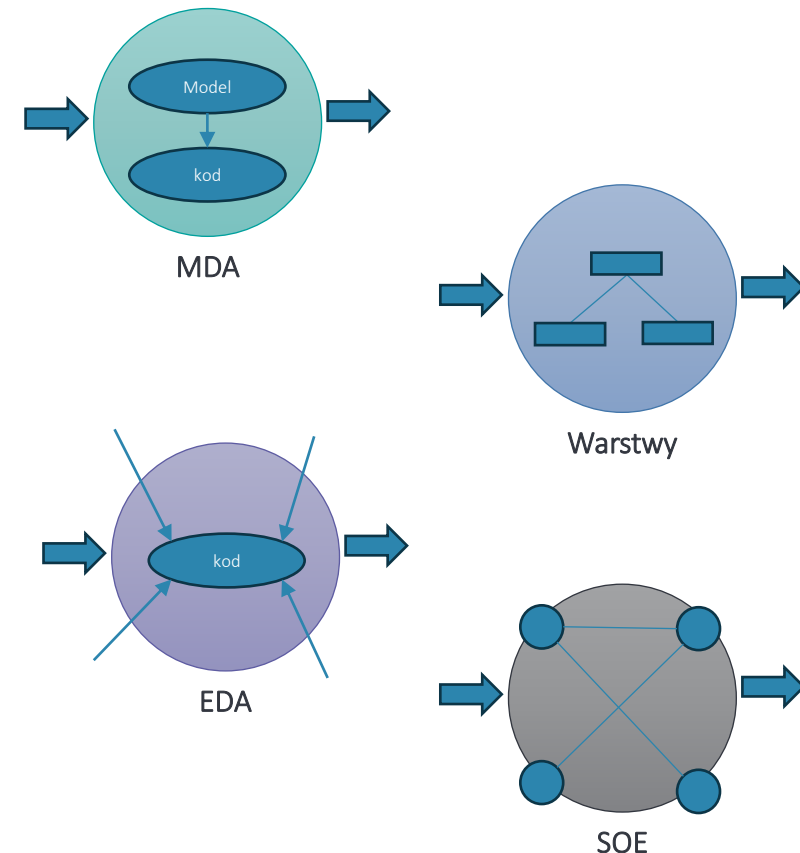
- MVC (Model-View-Controller)
Strukturyzuje aplikację na trzy główne komponenty: Model, Widok i Kontroler. Ułatwia separację logiki biznesowej od interfejsu użytkownika.
- Mikroserwisy (Microservices)
Podział aplikacji na niezależne, małe serwisy komunikujące się przez API. Umożliwia skalowanie i rozwijanie poszczególnych części systemu niezależnie.
- Monolit (Monolithic)
Cała logika aplikacji jest zawarta w jednym, dużym bloku kodu. Prostsze do zrozumienia i wdrożenia, ale trudniejsze do skalowania.
- CQRS (Command Query Responsibility Segregation)
Oddzielenie operacji odczytu i zapisu w systemie. Umożliwia optymalizację wydajności i zwiększa elastyczność systemu.



Projektowania Architektury Oprogramowania

Wzorce Architektoniczne

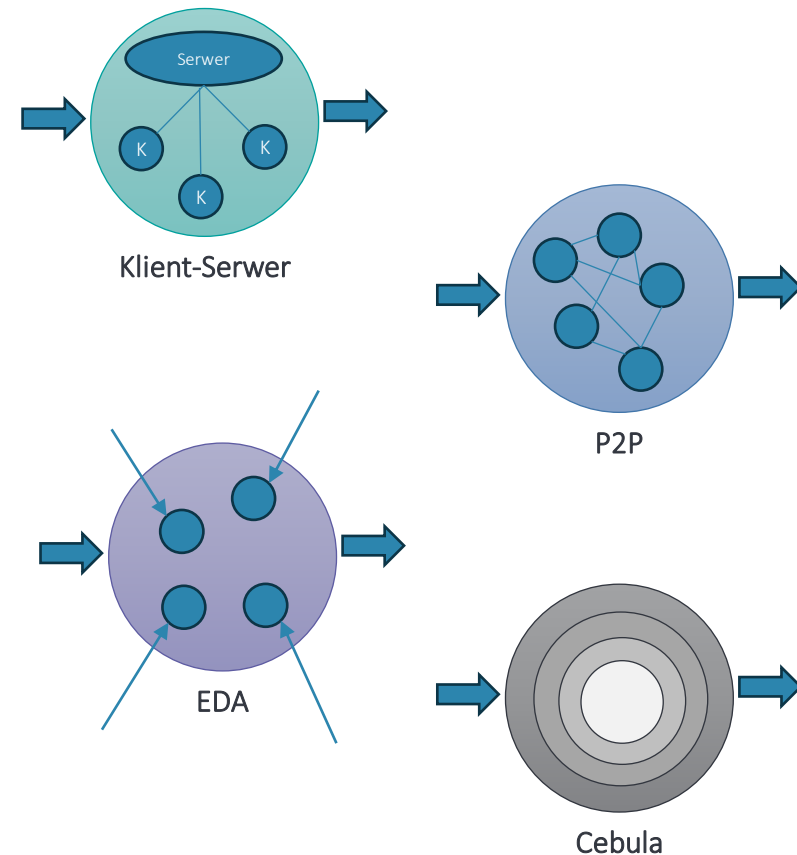
- Model Driven Architecture (MDA)
Podejście oparte na modelach, które są transformowane w kod źródłowy. Umożliwia automatyzację i standaryzację procesu projektowania.
- Architektura Warstwowa (Layered Architecture)
Podział systemu na warstwy, zwykle z hierarchią zależności. Każda warstwa ma określone zadania i komunikuje się tylko z sąsiednimi warstwami.
- Architektura Event-Driven (EDA)
Przepływ logiki programu jest sterowany przez zdarzenia, takie jak akcje użytkownika czy komunikaty z innych systemów.
- Architektura Orientowana na Usługi (SOA)
Podział funkcjonalności na niezależne usługi, które są dostępne przez sieć i komunikują się między sobą z wykorzystaniem protokołów.



Projektowania Architektury Oprogramowania

Wzorce Architektoniczne

- Architektura Klient-Serwer
Podział, w którym jeden serwer dostarcza zasoby lub usługi dla wielu klientów.
- Peer-to-Peer (P2P)
Każdy węzeł może działać zarówno jako klient, jak i serwer, umożliwiając bezpośrednią wymianę zasobów między sobą.
- Architektura Event-Driven (EDA)
Komponenty systemu reagują na zdarzenia, co umożliwia asynchroniczną komunikację i lepszą skalowalność.
- Architektura Cebulowa (Onion Architecture)
Organizuje aplikację wokół jej rdzenia biznesowego, dodając kolejne warstwy, takie jak logika aplikacji i interfejs użytkownika, w formie "obręczy".



Style programowania

Styl programowania

to zorganizowany zestaw konwencji, zasad i praktyk, które określają sposób strukturyzowania, formatowania i dokumentowania kodu źródłowego w celu zapewnienia jego spójności, czytelności i łatwości utrzymania.



Określenie "styl programowania" może być zastąpione terminami takimi jak:

- "konwencje kodowania" (Coding Conventions)
- "standardy programistyczne" (Programming Standards)
- "wytyczne programistyczne" (Coding Guidelines)
- "paradygmat programowania" (Programming Paradigm)

Określenie "programowanie" może być zastąpione terminami takimi jak:

- "implementacja" (Implementation)
- "kodowanie" (Coding)
- "skryptowanie" (Scripting)

Style programowania

Programowanie proceduralne

to styl programowania, w którym kod jest podzielony na funkcje (procedury). Każda funkcja wykonuje określoną czynność. Programowanie proceduralne jest często używane w językach programowania, takich jak C, C++, Java, Python czy JavaScript.

C

```
int main() {  
    int a = 10;  
    int b = 20;  
  
    int c = add(a, b);  
  
    printf("c = %d\n", c);  
  
    return 0;  
}  
  
int add(int a, int b) {  
    return a + b;  
}
```

Style programowania

Programowanie obiektowe

to styl programowania, w którym kod jest podzielony na obiekty. Obiekt to jednostka kodu, która zawiera dane i funkcje. Programowanie obiektowe jest często używane w językach programowania, takich jak Java, C#, Python czy JavaScript.

Java

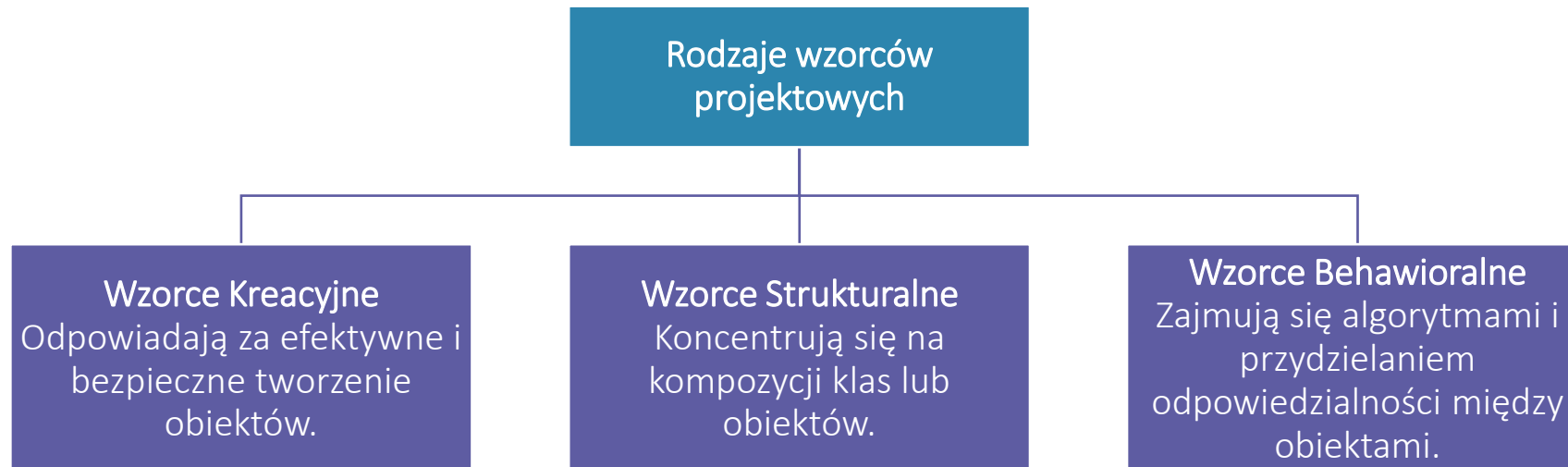
```
public class Calculator {  
  
    public static void main(String[] args) {  
        Calculator calc = new Calculator();  
  
        int c = calc.add(10, 20);  
  
        System.out.println("c = " + c);  
    }  
  
    public int add(int a, int b) {  
        return a + b;  
    }  
}
```

Style programowania

Wzorce projektowe (design patterns)

to sprawdzone rozwiązania do często występujących problemów w projektowaniu oprogramowania. Służą jako szablony, które można dostosować do konkretnych potrzeb projektu. Są opisem rozwiązania, a nie jego implementacją.

Wzorce projektowe stosowane są w projektach wykorzystujących programowanie obiektowe.



Style programowania

Programowanie deklaratywne

to styl programowania, w którym kod skupia się na opisie pożądanego wyniku, a nie na tym, w jaki sposób jest on uzyskiwany. Oznacza to, że programista nie musi martwić się o szczegóły implementacji. Programowanie deklaratywne jest często używane w językach, takich jak SQL, HTML czy CSS

HTML

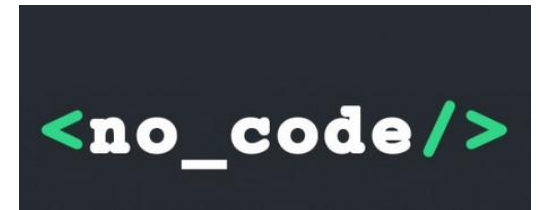
```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Deklarative HTML</title>
</head>
<body>
  <h1>This is a heading</h1>
  <p>This is a paragraph</p>
  <ul>
    <li>This is a list item</li>
    <li>This is another list item</li>
  </ul>
</body>
</html>
```

Style programowania

no-code

to termin odnoszący się do platform i narzędzi, które umożliwiają tworzenie aplikacji komputerowych i cyfrowych rozwiązań bez konieczności pisania tradycyjnego kodu programistycznego.

Są to zazwyczaj graficzne interfejsy użytkownika (GUI), gdzie użytkownicy mogą przeciągać i upuszczać komponenty, aby budować aplikacje, procesy lub systemy.



Łagodniejsza wersja to **low-code**

Tworzenie Stron
i Sklepów
Internetowych

Automatyzacja
Procesów
Biznesowych

Rozwój Aplikacji
Wewnętrznych
i Narzędzi
Biznesowych

Style programowania

Zalety no-code

Łatwość Użycia

No-code umożliwia tworzenie aplikacji bez głębokiej wiedzy technicznej. Użytkownicy mogą korzystać z interfejsów typu "przeciągnij i upuść", co znacznie upraszcza proces tworzenia.

Szybszy Rozwój

Pozwala na szybsze tworzenie i wdrażanie aplikacji, co jest szczególnie ważne w szybko zmieniających się warunkach biznesowych.

Obniżone Koszty

Redukcja potrzeby zaawansowanych umiejętności programistycznych może obniżyć koszty rozwoju i utrzymania aplikacji.

Demokratyzacja Tworzenia Aplikacji

Umożliwia niemal każdemu tworzenie aplikacji, co otwiera drzwi do innowacji i kreatywności osób, które nie są programistami.

Elastyczność i Skalowalność

Wiele narzędzi no-code oferuje łatwe skalowanie i dostosowywanie aplikacji do zmieniających się potrzeb biznesowych.

Style programowania

Wady no-code

Ograniczenia Dostosowania

Chociaż no-code oferuje wiele funkcji, może być ograniczający w przypadku bardziej złożonych lub specyficznych wymagań. Dostosowanie poza standardowymi opcjami może być trudne lub niemożliwe.

Problemy z Wydajnością

Aplikacje tworzone przy użyciu narzędzi no-code mogą nie być tak wydajne jak te stworzone tradycyjnymi metodami programowania, szczególnie w przypadku obsługi dużej liczby użytkowników lub danych.

Zależność od Dostawcy

Użytkowanie platformy no-code często wiąże się z zależnością od dostawcy narzędzia, zarówno pod względem możliwości technicznych, jak i cen.

Bezpieczeństwo i Prywatność

Aplikacje no-code mogą stwarzać wyzwania związane z bezpieczeństwem i prywatnością, zwłaszcza jeśli platforma nie przestrzega rygorystycznych standardów branżowych.

Skalowalność

Mimo że narzędzia no-code są skalowalne do pewnego stopnia, mogą pojawić się problemy przy bardzo dużych lub złożonych projektach.

