

dr inż. Michał Malinowski
michal.malinowski@uth.edu.pl

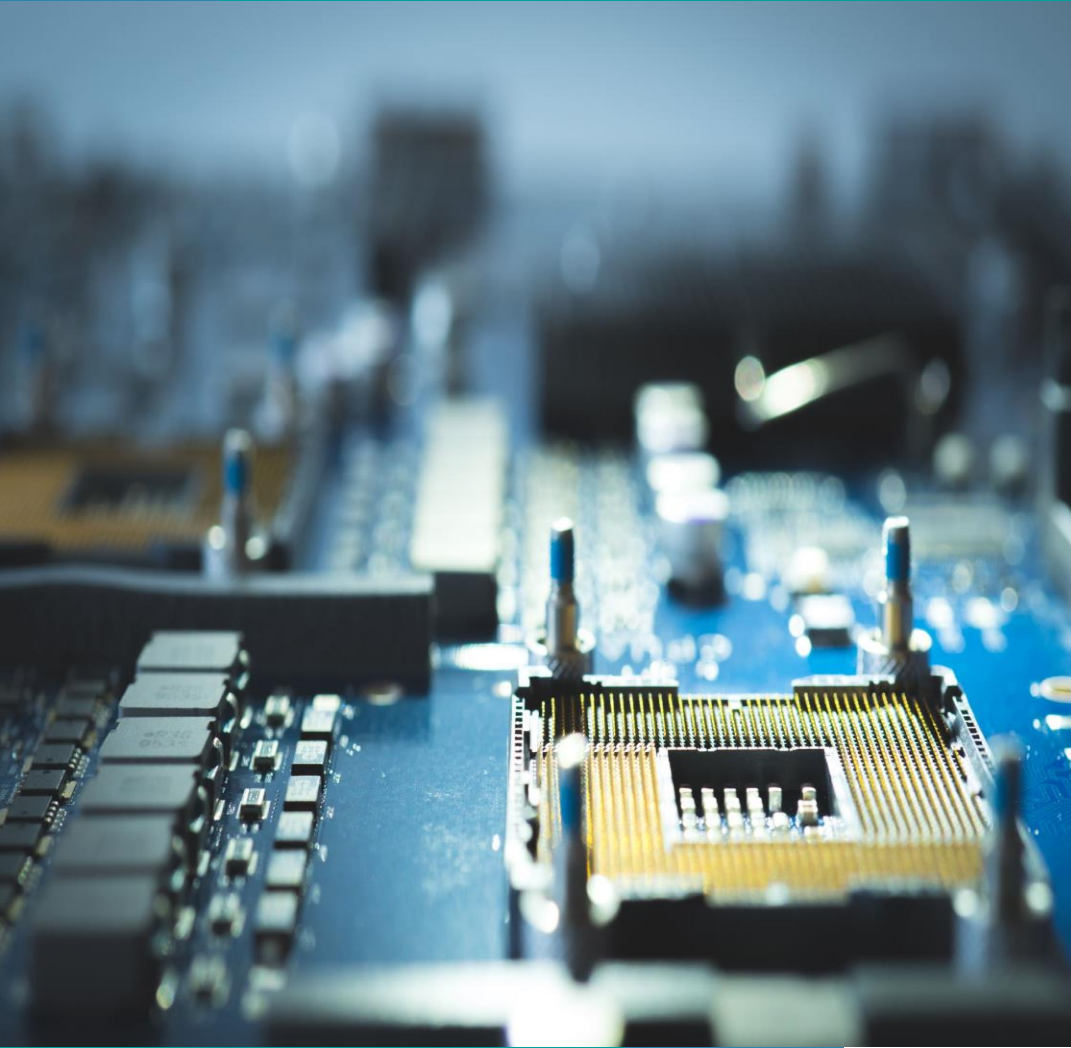


[https://www.linkedin.com/in/
michał-malinowski-malkon/](https://www.linkedin.com/in/michał-malinowski-malkon/)

Analiza Wymagań,
Harmonogram i Kosztorys
Projektu oraz Projektowanie
Architektury

Wykład 05

INŻYNIERIA
OPROGRAMOWANIA



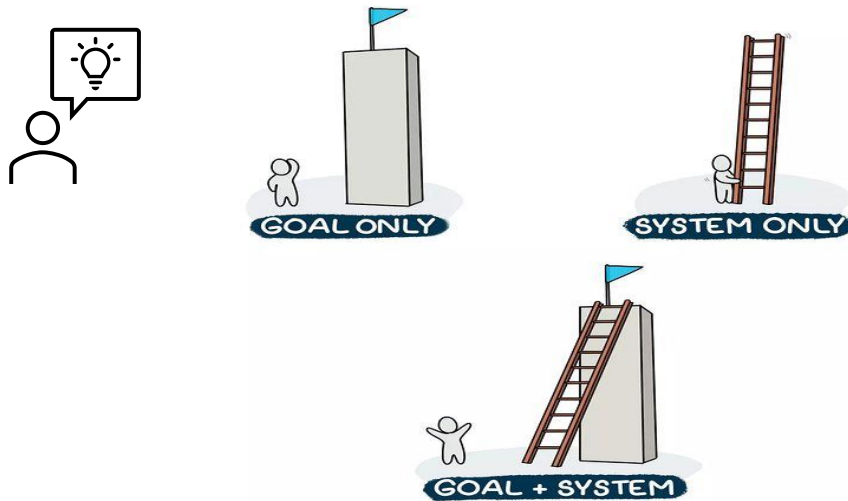
Zagadnienia (2h)

- [Wymagania dla oprogramowania](#)
- [Techniki zbierania, analizy i walidacji wymagań](#)
- [Harmonogram Projektu](#)
- [Kosztorys Projektu](#)
- [Projektowania Architektury Oprogramowania](#)

Wymagania dla oprogramowania

Wymagania

to zestaw specyfikacji, które określają, co system powinien lub nie powinien robić. Są one formalnym porozumieniem między interesariuszami odnośnie funkcji i ograniczeń systemu.



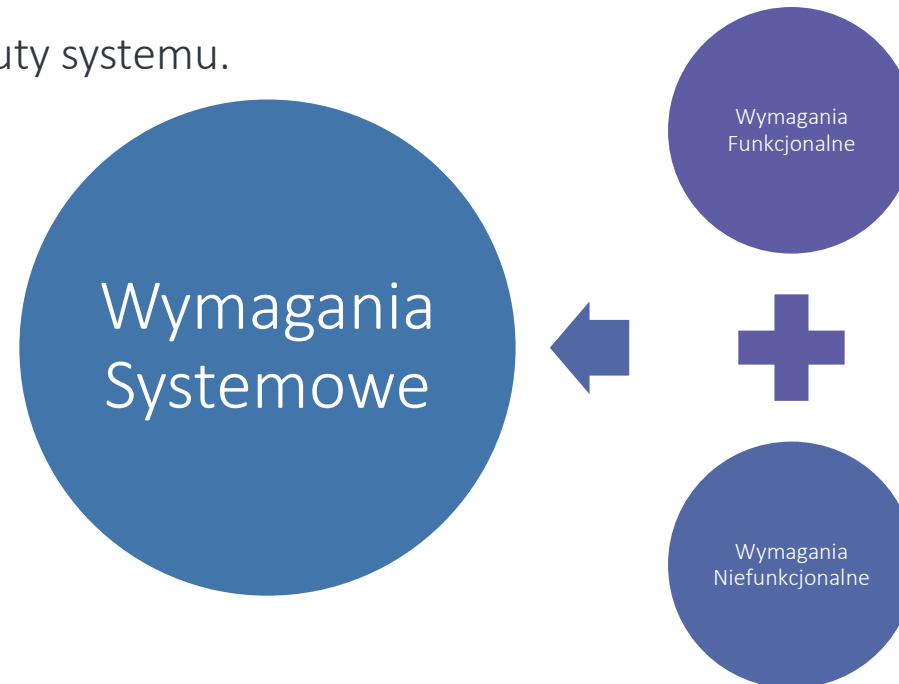
@melodie_tld



Wymagania dla oprogramowania

Wymagania systemowe

- Wymagania Funkcjonalne: Opisują konkretne funkcje lub działania systemu.
Przykłady: Logowanie użytkownika, zarządzanie bazą danych.
- Wymagania Niefunkcjonalne: Opisują jakościowe atrybuty systemu.
Przykłady: Wydajność, bezpieczeństwo, dostępność.



Wymagania dla oprogramowania

Znaczenie Wymagań

Podstawa dla Projektowania

Wymagania są wyjściowym punktem dla wszystkich decyzji projektowych.

Zrozumienie Kontekstu

Pomagają w zrozumieniu, co jest potrzebne do realizacji celów projektu.

Kryteria Akceptacji

Służą jako podstawa dla testowania i walidacji systemu.

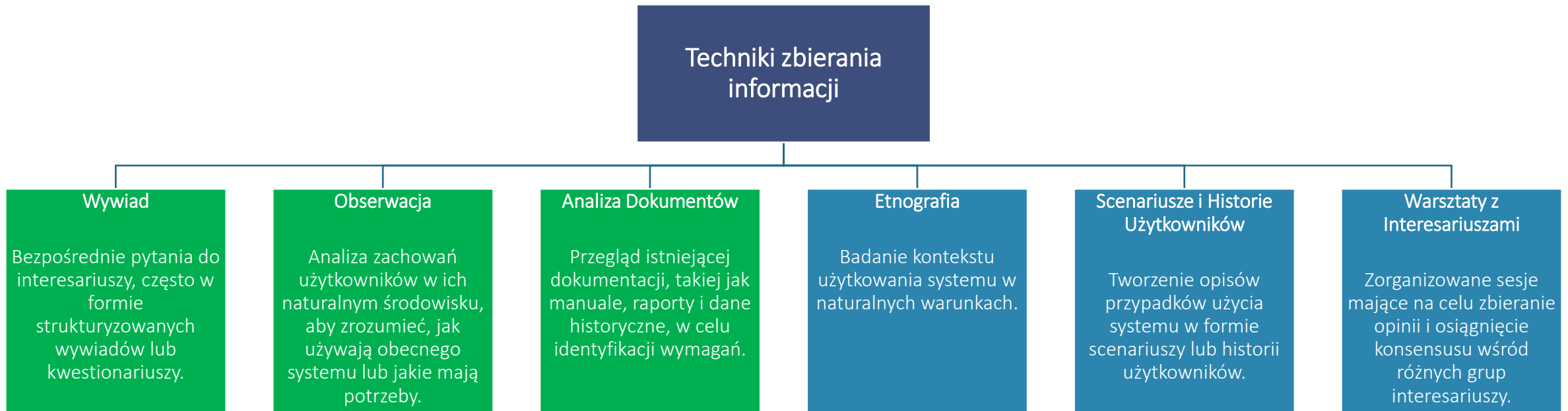
Komunikacja z Interesariuszami

Pomagają w zrozumieniu i komunikacji oczekiwań między różnymi grupami zaangażowanymi w projekcie.

Techniki zbierania, analizy i walidacji wymagań

Techniki zbierania wymagań

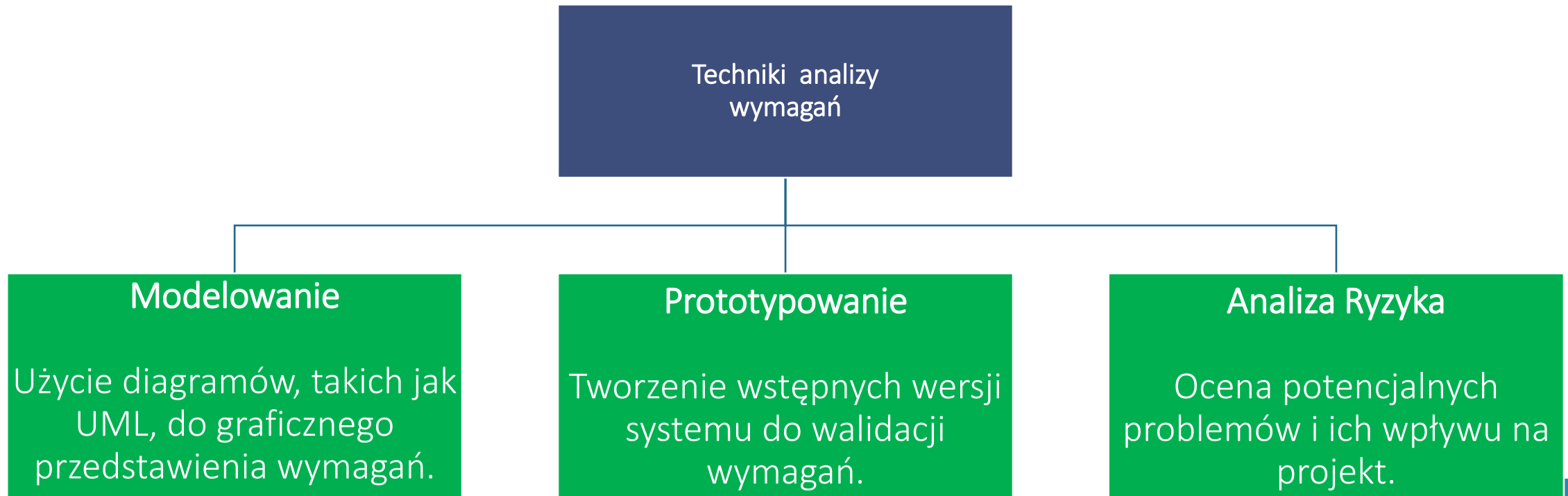
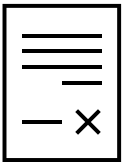
to metody i narzędzia używane do identyfikacji i dokumentowania wymagań oprogramowania.



Techniki zbierania, analizy i walidacji wymagań

Analiza wymagań

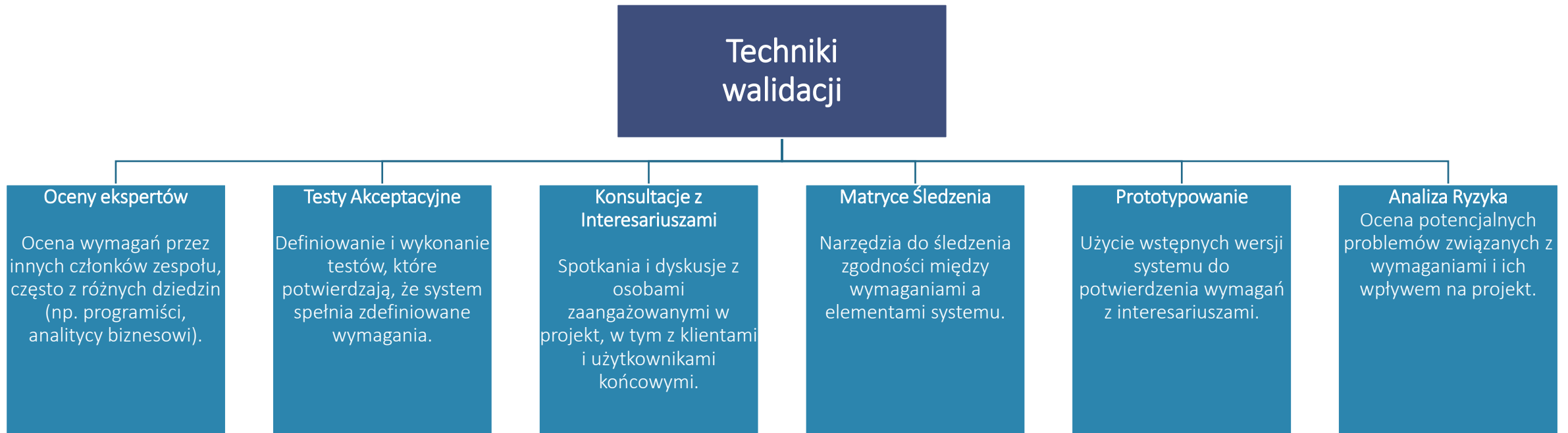
to proces identyfikacji, kategoryzacji, prioryteżacji i dokumentacji wymagań systemu.



Techniki zbierania, analizy i walidacji wymagań

Walidacja wymagań

to proces potwierdzenia, że zebrane wymagania odzwierciedlają rzeczywiste potrzeby i oczekiwania interesariuszy.



Techniki zbierania, analizy i walidacji wymagań

Elementy wymagania

Element	Opis
Numer Wymagania	Indywidualny numer wymagania - identyfikator
Wymaganie	Nazwa wymagania, ułatwiająca odniesienie i śledzenie.
Opis Wymagania	Szczegółowy opis wymagania, w tym co aplikacja powinna robić lub jakie funkcje powinna posiadać.
Priorytet	Określenie, czy wymaganie jest wysokiego, średniego, czy niskiego priorytetu. Może być używany do określenia, które funkcje są najważniejsze do wdrożenia.
Kategoria	Klasyfikacja wymagania, np. funkcjonalność, wydajność, bezpieczeństwo, interfejs użytkownika itp.
Źródło Wymagania	Informacja o tym, skąd pochodzi wymaganie (np. od klienta, zespołu, analizy rynku).
Status	Aktualny status wymagania, np. "Do rozpatrzenia", "Zaakceptowane", "W trakcie realizacji", "Zrealizowane".
Uwagi	Dodatkowe informacje lub komentarze dotyczące wymagania.
Data Dodania	Data, kiedy wymaganie zostało dodane do arkusza.
Odpowiedzialny	Osoba lub zespół odpowiedzialny za realizację wymagania.

Techniki zbierania, analizy i walidacji wymagań

Element	Wartość
Numer Wymagania	12.1
Wymaganie	Lista zadań
Opis Wymagania	Aplikacja powinna umożliwiać użytkownikom tworzenie i zarządzanie listami zadań. Użytkownicy powinni mieć możliwość dodawania, edytowania, usuwania i oznaczania zadań jako wykonane.
Priorytet	Wysoki
Kategoria	Funkcjonalność
Źródło Wymagania	Klient
Status	Do rozpatrzenia
Uwagi	Wymaganie to jest kluczowe dla podstawowej funkcjonalności aplikacji. Należy uwzględnić prostotę interfejsu użytkownika.
Data Dodania	2023-03-15
Odpowiedzialny	Anna Nowak (Zespół Projektowy)

Techniki zbierania, analizy i walidacji wymagań

Zagrożenia dla wymagań

Niedokładne lub niekompletne wymagania

Jeśli wymagania nie są dokładnie określone, mogą pojawić się problemy z interpretacją, co może prowadzić do nieodpowiedniego projektowania systemu.

Zmiana wymagań

Wymagania mogą zmieniać się w trakcie projektu, co jest naturalne, ale jeśli zmiany nie są odpowiednio zarządzane, może to prowadzić do chaosu i przekroczenia budżetu czy harmonogramu.

Nierozumienie kontekstu biznesowego

Wymagania powinny być zbierane z głębokim zrozumieniem kontekstu biznesowego, w którym oprogramowanie będzie funkcjonować. Brak tego zrozumienia może prowadzić do tworzenia funkcji, które nie spełniają rzeczywistych potrzeb użytkowników.

Komunikacja

Niewłaściwa komunikacja między wszystkimi zainteresowanymi stronami może prowadzić do nieporozumień i błędów w wymaganiach.

Założenia

Niepotwierdzone założenia mogą prowadzić do niewłaściwego zrozumienia potrzeb i oczekiwań użytkowników końcowych.

Techniki zbierania, analizy i walidacji wymagań

Zagrożenia dla wymagań

Nadmierna komplikacja

Próba zaspokojenia wszystkich możliwych wymagań może prowadzić do nadmiernie skomplikowanego systemu, który jest trudny w utrzymaniu i rozwijaniu.

Ograniczenia techniczne

Niewystarczająca wiedza o możliwościach technologicznych może prowadzić do ustanowienia wymagań, które są trudne lub niemożliwe do zrealizowania.

Zbyt ogólne wymagania niefunkcjonalne

Wymagania takie jak wydajność czy bezpieczeństwo muszą być specyficzne i mierzalne. Zbyt ogólne sformułowania mogą prowadzić do niejednoznacznych celów projektowych.

Zignorowanie ograniczeń budżetowych i czasowych

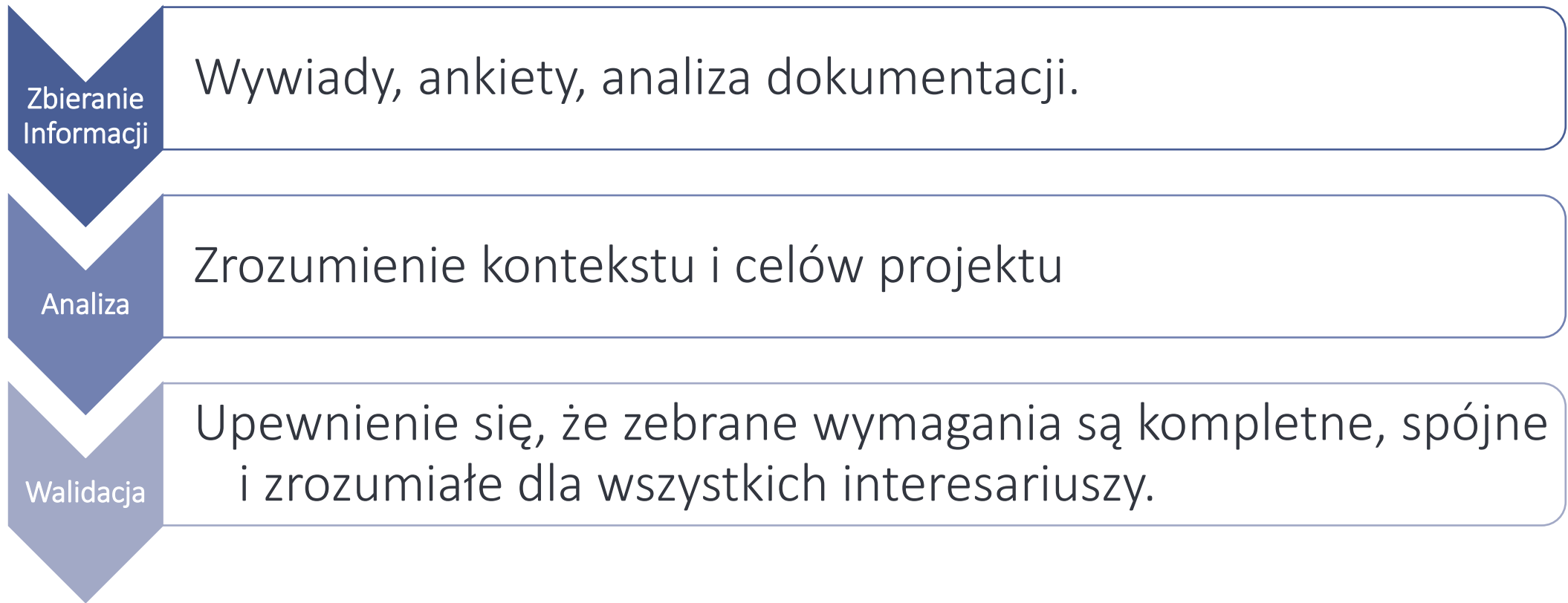
Wymagania muszą być realistyczne w kontekście dostępnych zasobów. Ignorowanie tych ograniczeń może prowadzić do niemożliwych do spełnienia celów.

Niewystarczająca analiza ryzyka

Każde wymaganie powinno być oceniane pod kątem potencjalnego ryzyka, które wprowadza do projektu. Nieuwzględnienie ryzyka może prowadzić do problemów w późniejszych etapach.

Techniki zbierania, analizy i walidacji wymagań

Podsumowanie

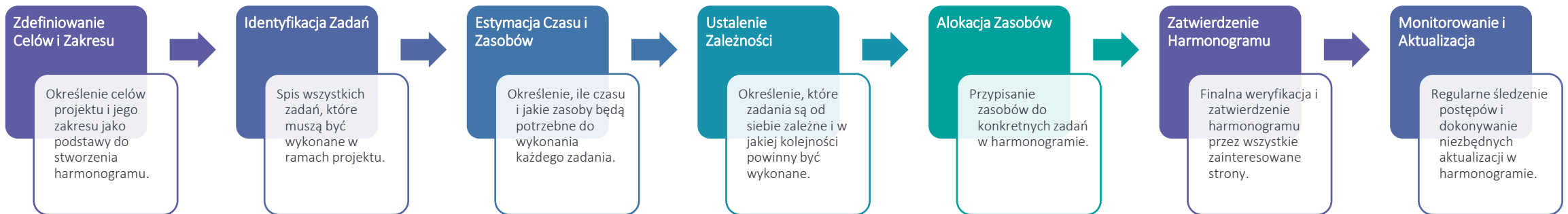


Harmonogram Projektu

Harmonogram

to planowany czas realizacji poszczególnych etapów projektu.

Kroki budowy i utrzymania harmonogramu



Harmonogram Projektu

Fazy harmonogramu

Lp.	Faza projektu	Kluczowe działania	Szacowany czas realizacji
1	Kick-off projektu	Rozpoczęcie projektu, zapoznanie zespołu z celami i zakresem.	1 tydzień
2	Analiza wymagań	Zbieranie i analiza wymagań od interesariuszy.	2-4 tygodnie
3	Projektowanie systemu	Tworzenie architektury, interfejsów i projektów baz danych.	3-6 tygodni
4	Planowanie zasobów	Przypisywanie kosztorys, personelu i zasobów technicznych do zadań.	1-2 tygodnie
5	Implementacja (kodowanie)	Pisanie kodu źródłowego oprogramowania.	2-6 miesięcy
6	Przeglądy kodu	Sesje analizy jakości kodu.	W trakcie implementacji
7	Testowanie	Wykonywanie różnych typów testów oprogramowania.	1-3 miesiące
8	Wdrożenie	Przekazywanie oprogramowania do środowiska produkcyjnego.	1-2 tygodnie
9	Szkolenia użytkowników	Przygotowanie użytkowników do pracy z systemem.	1-2 tygodnie
10	Utrzymanie i wsparcie	Działania utrzymaniowe po wdrożeniu oprogramowania.	Ciągłe po wdrożeniu
11	Kroki milowe	Punkty kontrolne wskazujące na zakończenie kluczowych faz.	Po każdej głównej fazie
12	Zarządzanie ryzykiem	Identyfikacja i minimalizacja ryzyk projektowych.	Ciągłe przez cały projekt
13	Zarządzanie zmianą	Zarządzanie zmianami w projekcie.	Ciągłe przez cały projekt
14	Przegląd postępów	Ocena postępu i doskonalenie procesu.	Po każdej głównej fazie
15	Aktualizacje i iteracje	Cykliczne opracowanie i doskonalenie oprogramowania.	W trakcie całego projektu
16	Zakończenie projektu	Formalne zamknięcie projektu, przekazanie dokumentacji.	1-2 tygodnie

Harmonogram Projektu

Struktura „tabeli” dla harmonogramu

Numer	Nazwa Zadania	Faza projektu	Opis Zadania	Zależności	Zasoby	Data Rozpoczęcia	Data Zakończenia	Czas Trwania	Stan Zadania	Osoba Odpowiedzialna	Uwagi	Procent Ukończenia	Priorytet
1.1	Inicjacja	Kick-off	Spotkanie inauguracyjne projektu	-	Zespół projektowy	2023-07-01	2023-07-01	1 dzień	Zakończone	Jan Kowalski	-	100%	Wysoki
2.1	Planowanie	Analiza wymagań	Zbieranie i analiza wymagań od interesariuszy	-	Analitycy	2023-07-02	2023-07-16	2 tygodnie	W toku	Anna Nowak	Wymaga potwierdzenia	50%	Wysoki
3.1	Wykonanie	Prototypowanie	Tworzenie prototypu interfejsu użytkownika	2.1	Projektanci UI	2023-07-17	2023-07-31	2 tygodnie	Nie rozpoczęte	Michał Wiśniewski	Oczekiwanie na wymagania	0%	Średni
4.1	Kontrola	Testy jednostkowe	Przeprowadzenie testów jednostkowych kodu	3.1	Testerzy	2023-08-01	2023-08-14	2 tygodnie	Nie rozpoczęte	Dorota Mazur	-	0%	Wysoki
5.1	Zamknięcie	Wdrożenie	Przeniesienie oprogramowania do produkcji	4.1	DevOps	2023-08-15	2023-08-29	2 tygodnie	Nie rozpoczęte	Piotr Zalewski	Potrzebny plan rollback	0%	Wysoki

Harmonogram Projektu

Zagrożenia dla harmonogramu

Zmiana zakresu projektu

Znana jako "creeping" lub "scope creep", jest to stopniowe rozszerzanie się lub zmiana zakresu projektu poza to, co zostało pierwotnie zaplanowane, co może prowadzić do znaczących opóźnień.

Niedoszacowanie złożoności

Niedocenienie złożoności zadań może prowadzić do ustalania nierealistycznych terminów, które są trudne lub niemożliwe do spełnienia.

Zasoby ludzkie

Niedobór lub fluktuacja personelu, niewłaściwe zarządzanie zasobami ludzkimi lub niedocenienie potrzebnych umiejętności może wpłynąć na tempo pracy.

Techniczne wyzwania

Nieprzewidziane problemy techniczne, takie jak trudności z integracją systemów lub błędy w oprogramowaniu, mogą spowodować opóźnienia.

Zależności zewnętrzne

Opóźnienia wynikające z zależności od dostawców, usług zewnętrznych lub integracji z innymi systemami często są poza kontrolą zespołu projektowego.

Zmiany w priorytetach biznesowych

Jeśli priorytety biznesowe zmieniają się w trakcie realizacji projektu, może to wymagać zmian w harmonogramie, które mogą spowodować opóźnienia.

Harmonogram Projektu

Zagrożenia dla harmonogramu

Niejasne wymagania

Jeśli wymagania nie są jasno zdefiniowane na wstępie, mogą pojawić się błędy i konieczność powtórzenia wykonania pracy, co wpływa na harmonogram.

Komunikacja

Niewystarczająca komunikacja w zespole i między zespołami może prowadzić do nieporozumień i błędów, które wymagają czasu na naprawienie.

Zarządzanie ryzykiem

Brak odpowiedniego planowania i reagowania na potencjalne ryzyka może skutkować nieoczekiwanymi opóźnieniami.

Problemy jakościowe

Wymagania dotyczące jakości, które nie są spełnione, mogą prowadzić do czasochłonnych cykli testowania i poprawek.

Zarządzanie projektami

Słabe praktyki zarządzania projektami, takie jak niewłaściwe planowanie, brak monitorowania postępów lub nieskuteczne zarządzanie zmianami mogą przyczyniać się do problemów z harmonogramem.

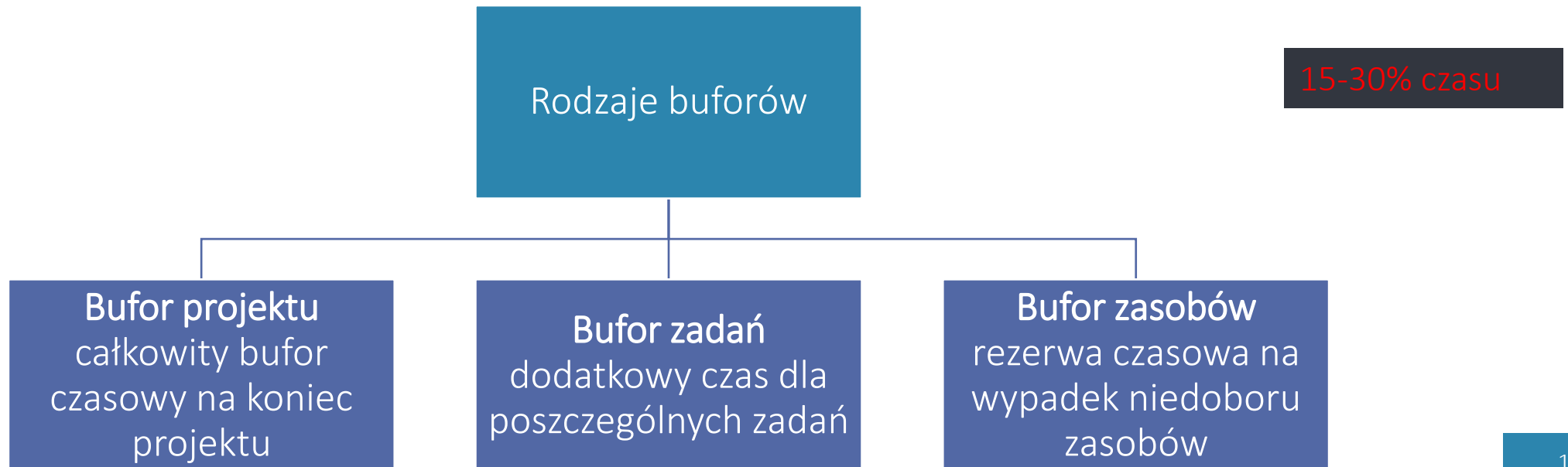
Warunki zewnętrzne

Czynniki zewnętrzne, takie jak zmiany w przepisach prawnych lub kryzysy ekonomiczne, mogą wpłynąć na harmonogram projektu.

Harmonogram Projektu

Bufor bezpieczeństwa („bufor kontyngencyjny”, „bufor ryzyka”)

to dodatkowy czas włączony do harmonogramu projektu, który nie jest przypisany do konkretnych zadań. Jest to rodzaj "rezerwy czasowej", która może być wykorzystana w razie potrzeby, aby złagodzić wpływ opóźnień lub innych problemów.



Harmonogram Projektu

Łańcuch krytyczny

to sekwencja zadań w projekcie, która określa minimalny czas potrzebny do zakończenia projektu. Obejmuje ona zarówno zależności między zadaniami, jak i ograniczenia związane z dostępnością zasobów.

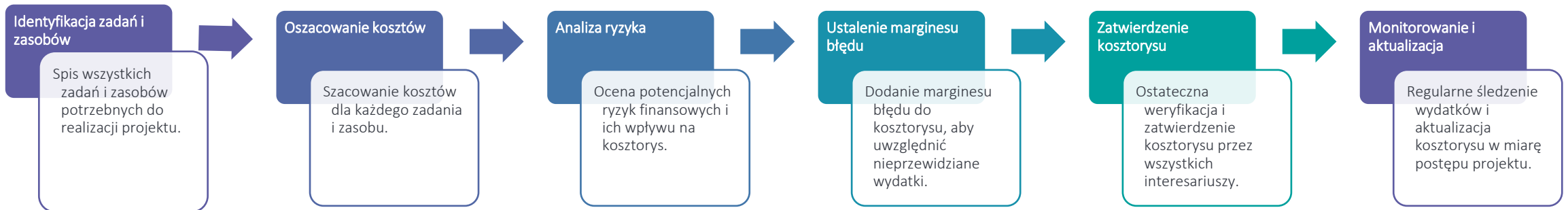
Faza Projektowania	Faza Kodowania	Faza Testowania	Faza Wdrożenia
Zadanie A: Zebranie wymagań	Zadanie D: Implementacja modułu 1	Zadanie G: Testy jednostkowe	Zadanie J: Przygotowanie środowiska
Zadanie B: Opracowanie specyfikacji	Zadanie E: Implementacja modułu 2	Zadanie H: Testy integracyjne	Zadanie K: Wdrożenie systemu
Zadanie C: Tworzenie architektury	Zadanie F: Integracja modułów	Zadanie I: Testy akceptacyjne	Zadanie L: Szkolenie użytkowników

Kosztorys Projektu

Kosztorys

to szacunkowa kalkulacja wszystkich kosztów związanych z realizacją projektu.

Kroki przygotowania i utrzymania kosztorysu



Kosztorys Projektu

Elementy kosztorysu

Lp.	Element Kosztorysu	Opis	Szacowane Koszty (PLN)
1	Zasoby Ludzkie	Wynagrodzenia programistów, projektantów, testerów, menedżerów projektu, koszty szkoleń	500 000
2	Oprogramowanie	Licencje, narzędzia programistyczne, subskrypcje	50 000
3	Sprzęt	Komputery, serwery, urządzenia sieciowe	100 000
4	Infrastruktura i Hosting	Hosting, serwery, usługi chmurowe	30 000
5	Materiały i Usługi Zewnętrzne	Zakup materiałów, komponentów, konsultacje zewnętrzne	80 000
6	Podróże i Wydatki Związane z Miejscem Pracy	Podróże służbowe, zakwaterowanie, koszty przestrzeni biurowej	20 000
7	Szkolenia i Rozwój	Koszty szkoleń technicznych, kursy zawodowe	15 000
8	Marketing i Komunikacja	Materiały promocyjne, reklama, PR	40 000
9	Zarządzanie Ryzykiem i Rezerwy na Nieprzewidziane Wydatki	Fundusze na nieoczekiwane koszty	25 000
10	Koszty Administracyjne i Zarządcze	Zarządzanie projektem, księgowość, usługi prawne	60 000
11	Testowanie i Kontrola Jakości	Narzędzia do testowania, testy zewnętrzne	35 000
12	Wdrożenie i Szkolenia Użytkowników Końcowych	Koszty wdrożenia systemu, szkolenia użytkowników	25 000
13	Utrzymanie i Wsparcie Po Wdrożeniu	Utrzymanie oprogramowania, wsparcie techniczne, aktualizacje	50 000

Kosztorys Projektu

Struktura „tabeli” dla kosztorysu

Numer	Element Kosztorysu	Szczegół	Szacowane Koszty (PLN)
1	Zasoby Ludzkie	Wynagrodzenie programisty Java (3 osoby na 6 miesięcy)	360 000
1.1		Wynagrodzenie projektanta UI/UX (1 osoba na 3 miesiące)	45 000
1.2		Wynagrodzenie testera oprogramowania (1 osoba na 4 miesiące)	48 000
1.3		Wynagrodzenie menedżera projektu (1 osoba na 6 miesięcy)	150 000
2	Oprogramowanie	Licencje na środowisko programistyczne IntelliJ IDEA (5 licencji)	10 000
2.1		System kontroli wersji GitLab (roczna subskrypcja)	15 000
2.2		Narzędzia do zarządzania projektem Jira (roczna subskrypcja)	10 000
2.3		Pakiet bezpieczeństwa i szyfrowania danych	15 000
3	Sprzęt	Serwery deweloperskie (2 serwery)	40 000
3.1		Stacje robocze dla zespołu (5 stacji)	40 000
3.2		Urządzenia sieciowe i peryferia	20 000
3.3		UPS-y i zabezpieczenia prądowe	10 000

Kosztorys Projektu

Zagrożenia dla kosztorysu

Wzrost cen

Zmiany w cenach rynkowych, zarówno w zakresie kosztów pracy (np. wzrost wynagrodzeń) jak i ceny narzędzi, oprogramowania czy licencji mogą nieoczekiwanie zwiększyć koszty projektu.

Zmiany technologii

Szybka ewolucja technologii może sprawić, że zakupione rozwiązania lub narzędzia szybko się dezaktualizują, co może wymagać dodatkowych inwestycji.

Złe estymacje kosztów

Niewłaściwa ocena kosztów projektu na etapie planowania, która nie bierze pod uwagę wszystkich potencjalnych wydatków, może prowadzić do poważnych przekroczeń budżetu.

Zmiany w przepisach prawnych lub podatkowych

Nowe regulacje prawne mogą wprowadzić konieczność dodatkowych działań lub zmian w projekcie, co generuje dodatkowe koszty.

Niedocenienie trudności zadania przez zespół

Brak doświadczenia i specjalistycznej wiedzy wśród członków zespołu może prowadzić do błędów, które będą kosztowne w naprawie.

Brak zabezpieczenia budżetowego na nieprzewidziane wydatki

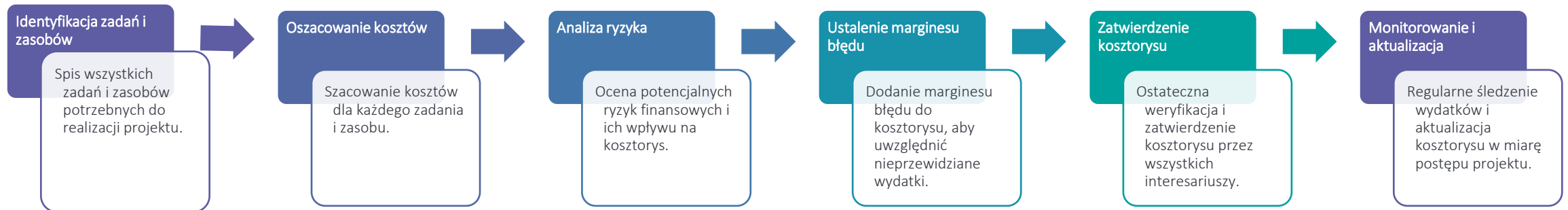
Brak środków rezerwowych na nieoczekiwane zdarzenia lub problemy może prowadzić do sytuacji, w której dodatkowe koszty nie są pokryte.

Kosztorys Projektu

Kosztorys

to szacunkowa kalkulacja wszystkich kosztów związanych z realizacją projektu.

Kroki przygotowania i utrzymania kosztorysu





thinking



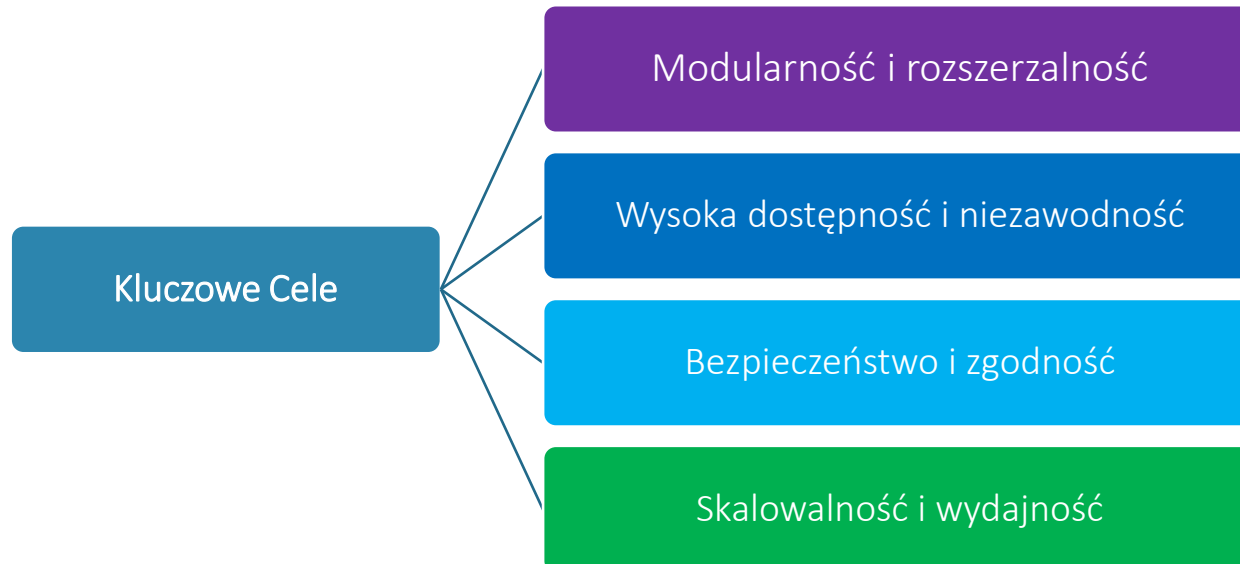
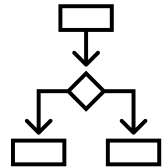
writing

@MILANICREATIVE

Projektowania Architektury Oprogramowania

Projektowanie architektury oprogramowania

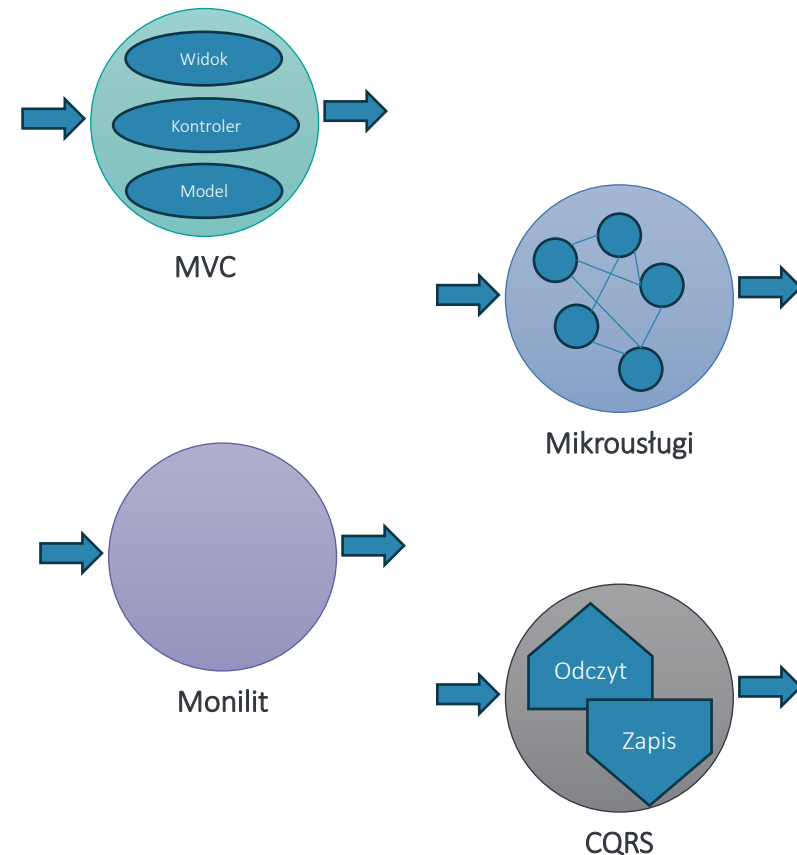
to proces definiowania strukturalnych elementów systemu, ich wzajemnych relacji oraz interakcji, aby osiągnąć określone cele jakościowe i funkcjonalne.



Projektowania Architektury Oprogramowania

Wzorce Architektoniczne

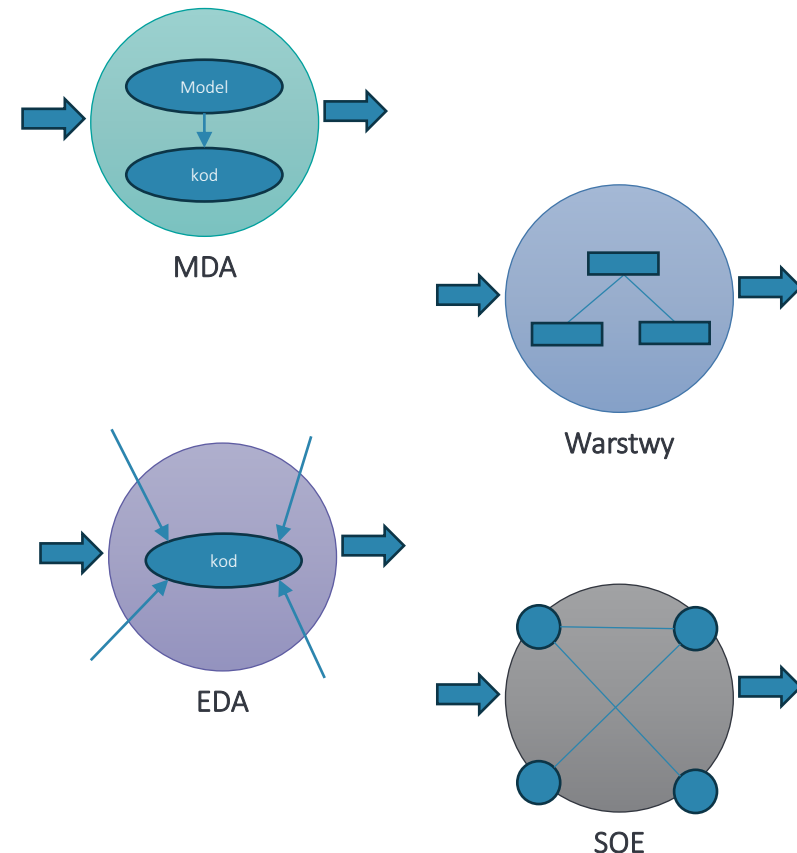
- MVC (Model-View-Controller)
Strukturyzuje aplikację na trzy główne komponenty: Model, Widok i Kontroler. Ułatwia separację logiki biznesowej od interfejsu użytkownika.
- Mikroserwisy (Microservices)
Podział aplikacji na niezależne, małe serwisy komunikujące się przez API. Umożliwia skalowanie i rozwijanie 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

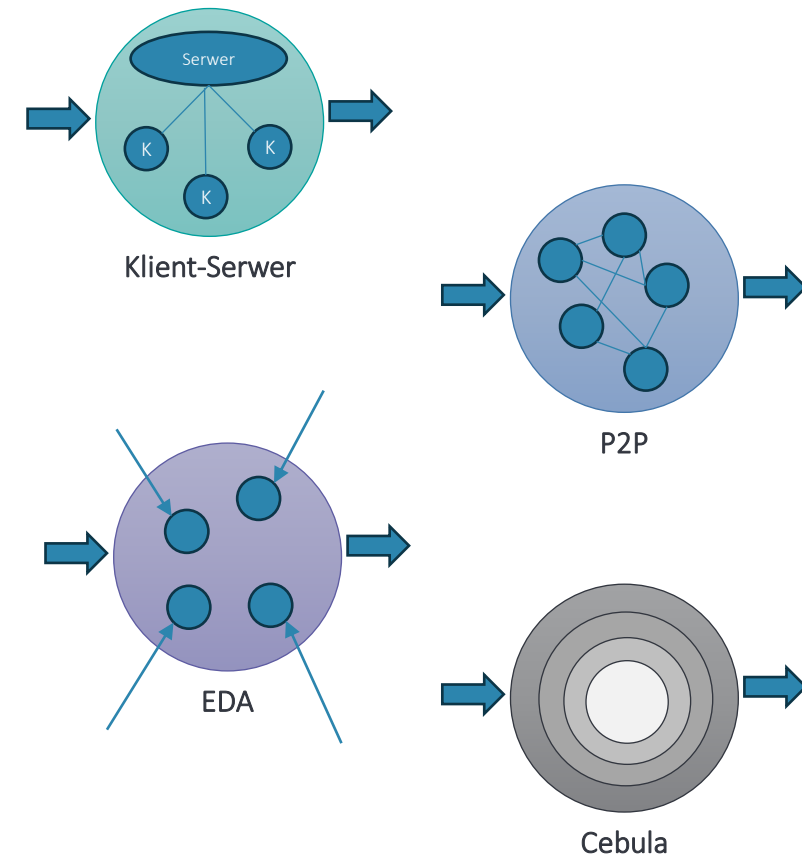
- Model Driven Architecture (MDA)
Podejście oparte na modelach, które są transformowane w kod źródłowy. Umożliwia automatyzację i standaryzację procesu projektowania.
- Architektura Warstwowa (Layered Architecture)
Podział systemu na warstwy, zwykle z hierarchią zależności. Każda warstwa ma określone 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".

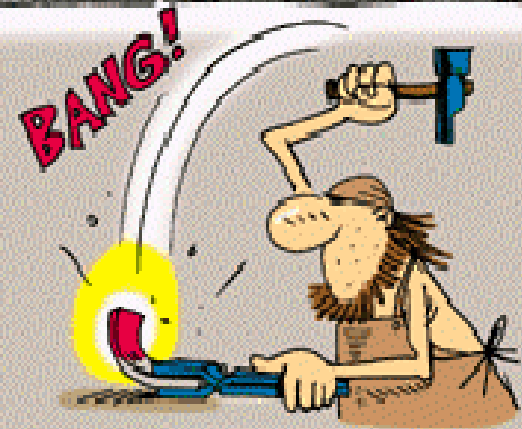




STONE AGE



BRONZE AGE



IRON AGE



DARK AGE



MODERN AGE



COMPUTER AGE

TAK