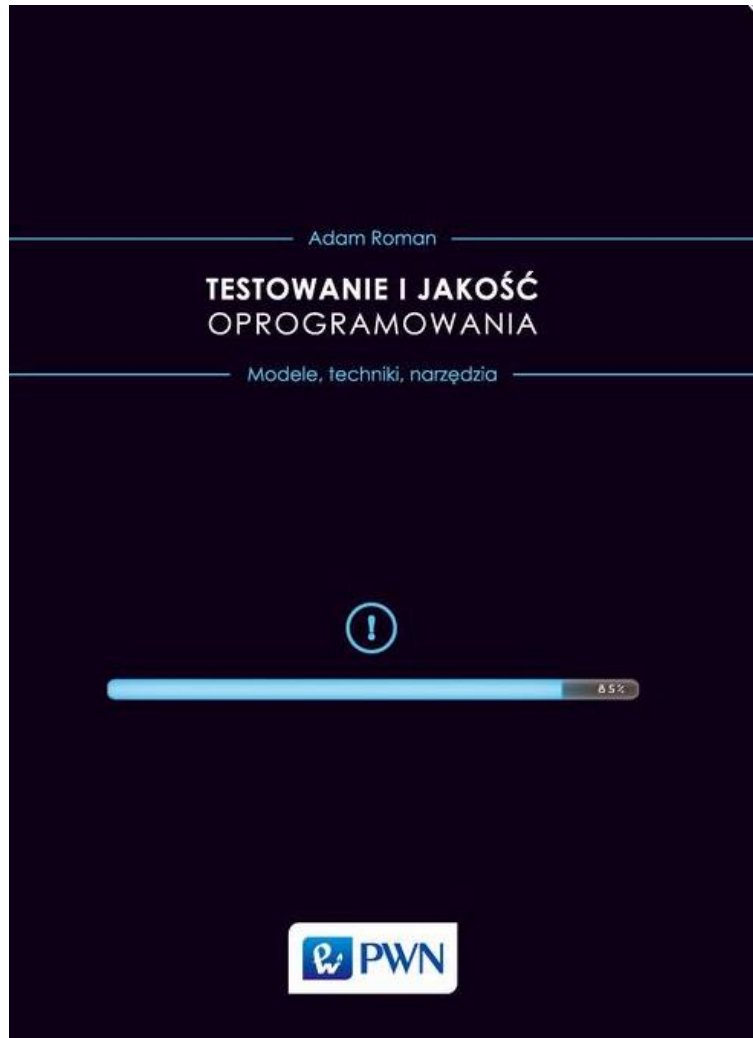


TESTOWANIE OPROGRAMOWANIA

WARSZTATY

Źródła wiedzy



http://sqam.org/wpcontent/uploads/2015/11/Sylabus_Podstawowy_2011.pdf

<http://testerzy.pl/>

<http://jakzdacistqb.blogspot.com/>

<http://sqa.stackexchange.com/>

<http://www.testowanie.net/>

<https://www.sogeti.com>

<http://www.softwaretestinghelp.com/>

<http://trojqa.pl/>

<http://www.testwarez.pl/>

<http://sjsi.org/>

I wiele, wiiiiiiiiie innych.....

Mel Silberman „**Credo uczącego się**”:

Gdy **usłyszę**, zapomnę.

Gdy **usłyszę** i **zobaczę**, zapamiętam.

Gdy **usłyszę**, **zobaczę** i **porozmawiam**, zrozumiem.

Gdy **usłyszę**, **zobaczę**, **porozmawiam** i **zrobię**, zdobywam
sprawność i wiedzę.

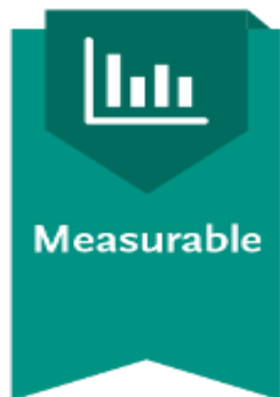
Gdy **uczę innych**, jeszcze lepiej opanowuję to, czego się
wcześniej nauczyłem.

S



Skonkretyzowany
i szczegółowy
(co?, po co?,
jak?...)

M



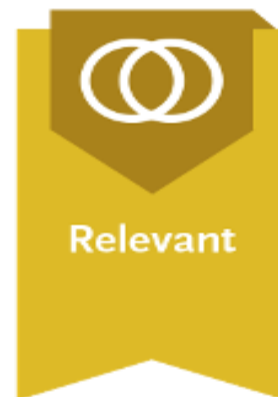
Mierzalny
(ile?, jak
długo?)

A



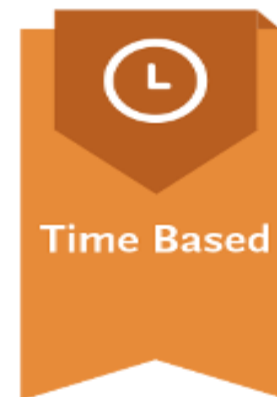
Ambitny i
osiągalny
(da się
osiągnąć
i można do
niego
dążyć)

R



Realistyczny
i istotny
(nie może być
za łatwy, ale
też nie zbyt
abstrakcyjny)

T



Terminowy
(dokładne
określenie
czasu, kiedy
chcesz osiągnąć
cel)

Wcześniejsze definicje

tester/testerka oprogramowania

to osoba związana z **wytwarzaniem oprogramowania**
odpowiadająca za wsparcie **procesów** zapewnienia
odpowiedniej **jakości** temu oprogramowaniu poprzez jego
weryfikację i walidację.

Rola testowania

Dostarczanie informacji

Podstawa do oceny ryzyka

Testerzy nie podejmują decyzji biznesowych



Cechy testera

Profesjonalny pesymizm

Wyważona ciekawość

Skupienie

Aktywność

Odwaga



Błąd – Defekt – Awaria

Pomyłka, błąd (mistake, error): Działanie człowieka powodujące powstanie nieprawidłowego wyniku.

Przyczyny:

- › Stres
- › Presja czasu
- › Niewystarczająca wiedza
- › Brak skupienia
- › Zmęczenie
- › Czynniki zewnętrzne



Defekt, usterka, pluskwa (defect, fault, bug):
Skutek błędu twórcy oprogramowania. Usterka może, ale nie musi spowodować awarii.

Typy defektów:

- › Literówka
- › Niezgodność z dokumentacją
- › Błąd logiczny
- › Zły typ danych
- › Niewłaściwe ułożenie elementów
- › Brak walidacji



Awaria (failure, problem, incident): Odchylenie od spodziewanego zachowania albo wyniku działania oprogramowania.

Typy awarii:

- › Program wykonał operację niezgodnie z oczekiwaniami
- › Program wykonał operację częściowo
- › Program przestał działać/odpowiadać
- › Program nie wykonał operacji



Człowiek popełnia **błąd (pomyłkę)**, której skutkiem jest **defekt (usterka, bug)**. Jeżeli kod z defektem zostanie wykonany, system nie zadziała zgodnie z oczekiwaniem, wywołując **awarię**.

Wspólne zaistnienie tych trzech czynników powoduje nieprawidłowe działanie testowanego produktu



Słowa **bug** w kontekście usterki użył Thomas Edison już w 1878.
Wprowadzenie do użycia tego terminu - pani admirał Grace Hopper.
Podczas prac prowadzonych 9 września 1947 r. z komputerem Harvard Mark II stwierdzono jego nieprawidłowe działanie, a po poszukiwaniach przyczyny operatorzy, znaleźli w przekaźniku 70. panelu F ćmę (ang. moth), która powodowała spięcie. Owad został usunięty i wklejony do dziennika z wpisem o 'bugu'.

Photo # NH 96566-KN First Computer "Bug", 1945

92

9/9

0800 Machine started
1000 " stopped - machine ✓

1300 (033) MP-MC 2.130476415
(033) PRO 2 2.130476415
convd 2.130476415

Relays 6-2 in 033 failed special speed test
in relay 11.00 test.


1100 Started Cosine Tape (Sine check)
1525 Started Multi-Adder Test.

1545

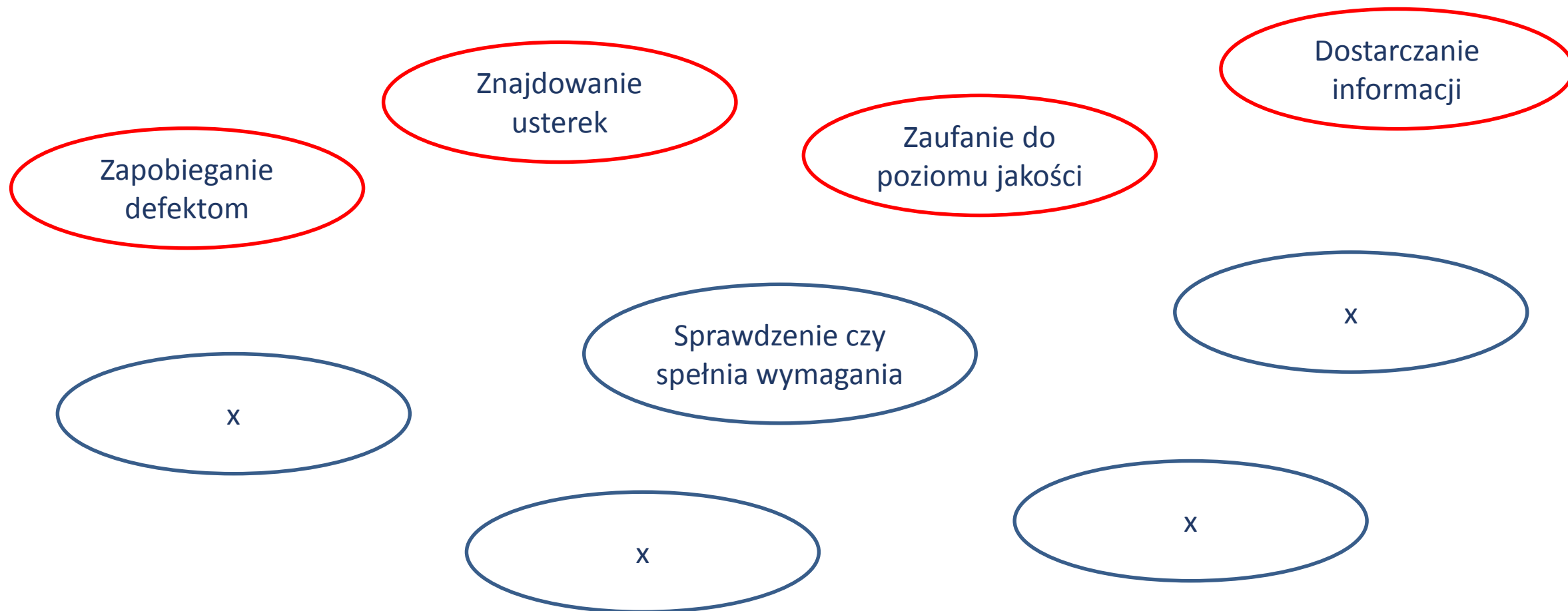
Relay #70 Panel F
(moth) in relay.

First actual case of bug being found.

1630 Machine started.
1700 closed down.



Po co testujemy?



Testowanie NIE jest:

**Debagowaniem
(*debugging*)**

(które jest procesem wyszukiwania,
analizowania i
usuwania przyczyn awarii
oprogramowania)
(testowanie zaś jest wyszukiwaniem
awarii, przez co
wykazuje ono obecność defektów)

Udowodnieniem,
że oprogramowanie działa
„bezbłędnie”;
(Bo tego się nie da
dowieść.)

Udowodnieniem,
że oprogramowanie nie
działa;

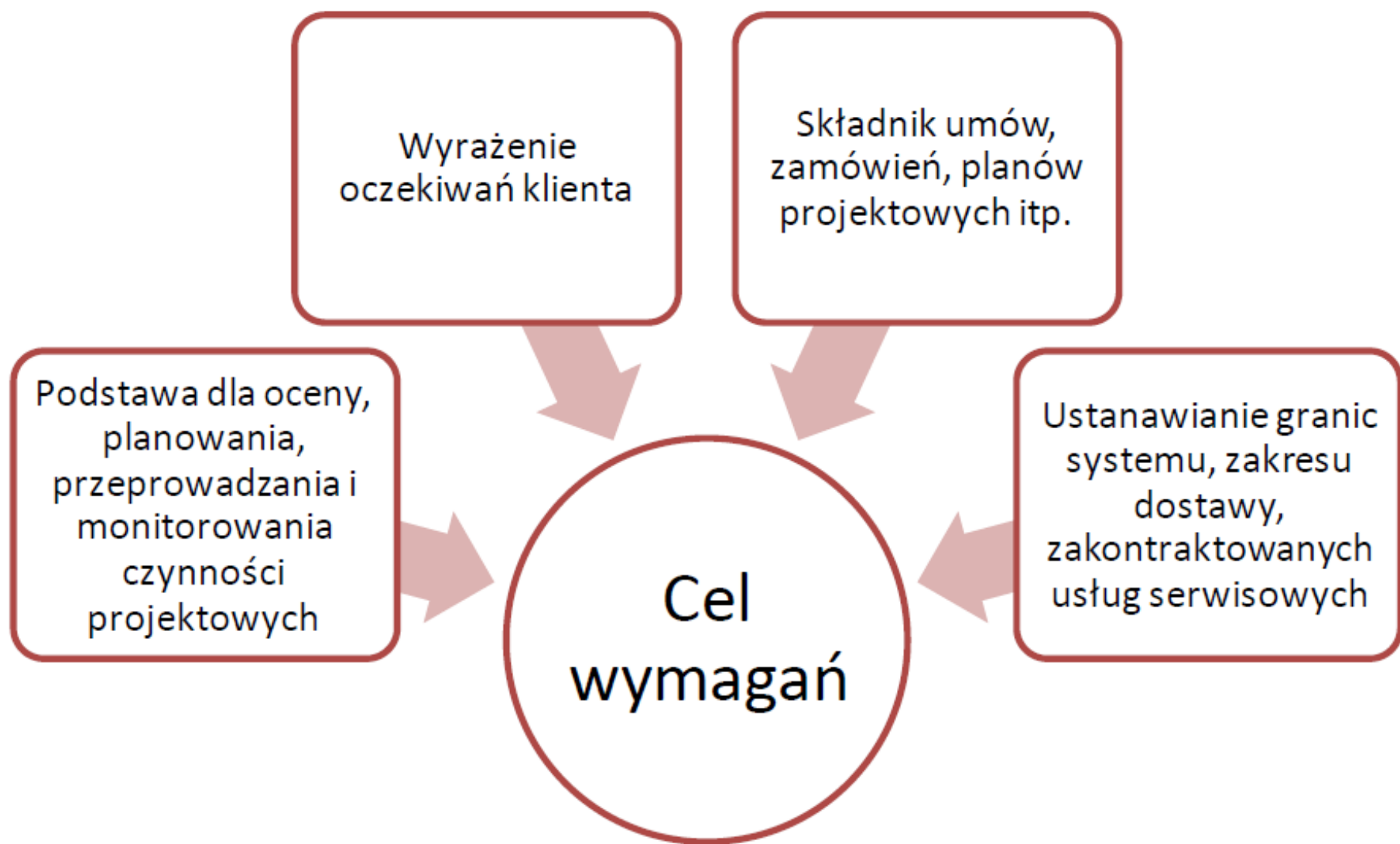
Wprowadzenie do wymagań

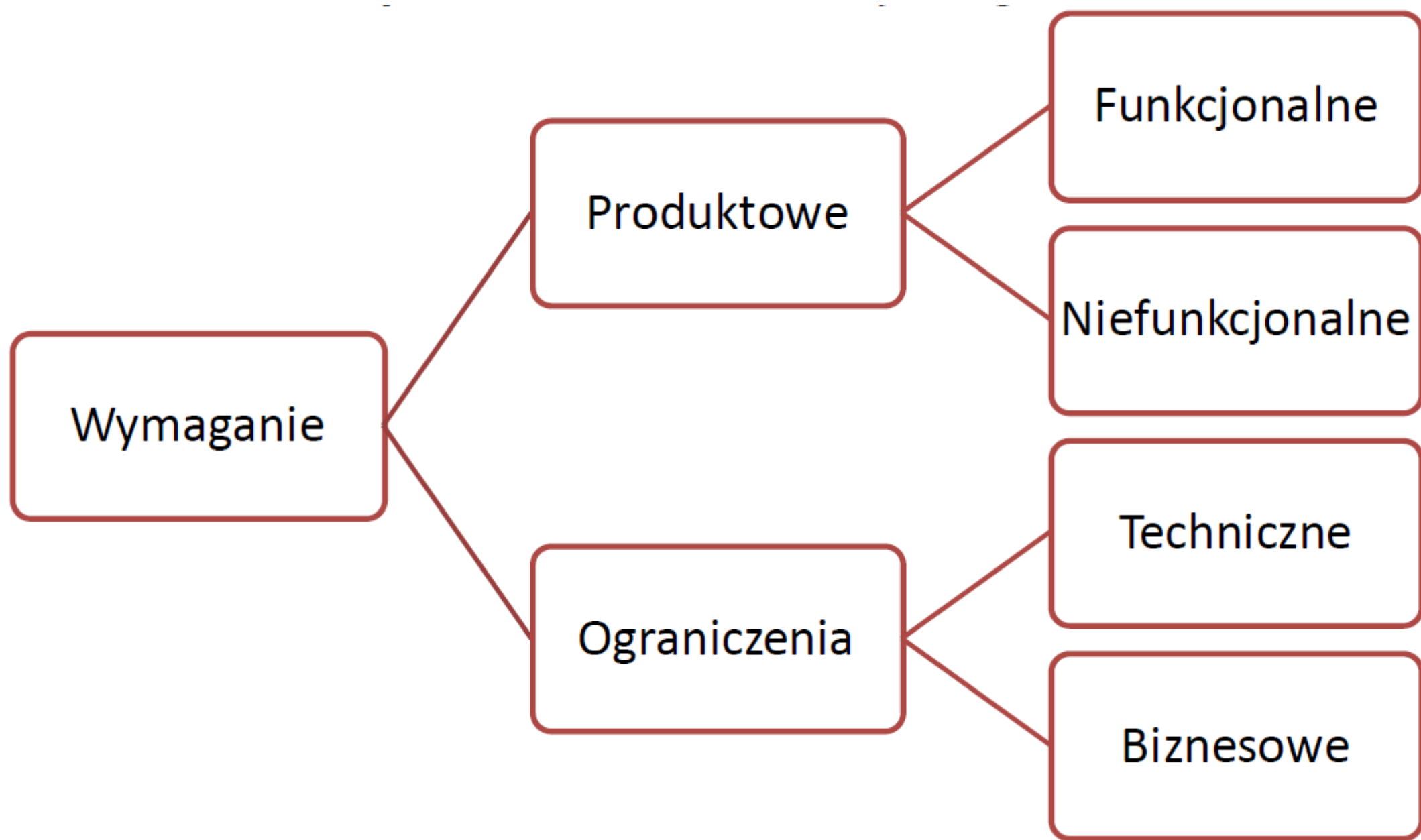
- Czym jest wymaganie?
- Skąd je brać?
- Jak je określać?
- Jak budować system IT mając wymagania?
- Czy wymagania dotyczą tylko IT?

Czym są wymagania biznesowe?

Wymaganie jest **pojedynczą**, udokumentowaną **potrzebą** określonego produktu czy usługi albo sposobu ich działania.

Wymagania pokazują jakie elementy są niezbędne w danym projekcie.





Wymagania

Są istotne dla każdego projektu
Po co są zbierane?

- pokazać czego chcą interesariusze
- dać interesariuszom szansę określenia czego chcą 😊
- reprezentować różne punkty widzenia
- sprawdzić wygląd
- mierzyć postęp prac
- mieć punkt odniesienia przy akceptacji produktu

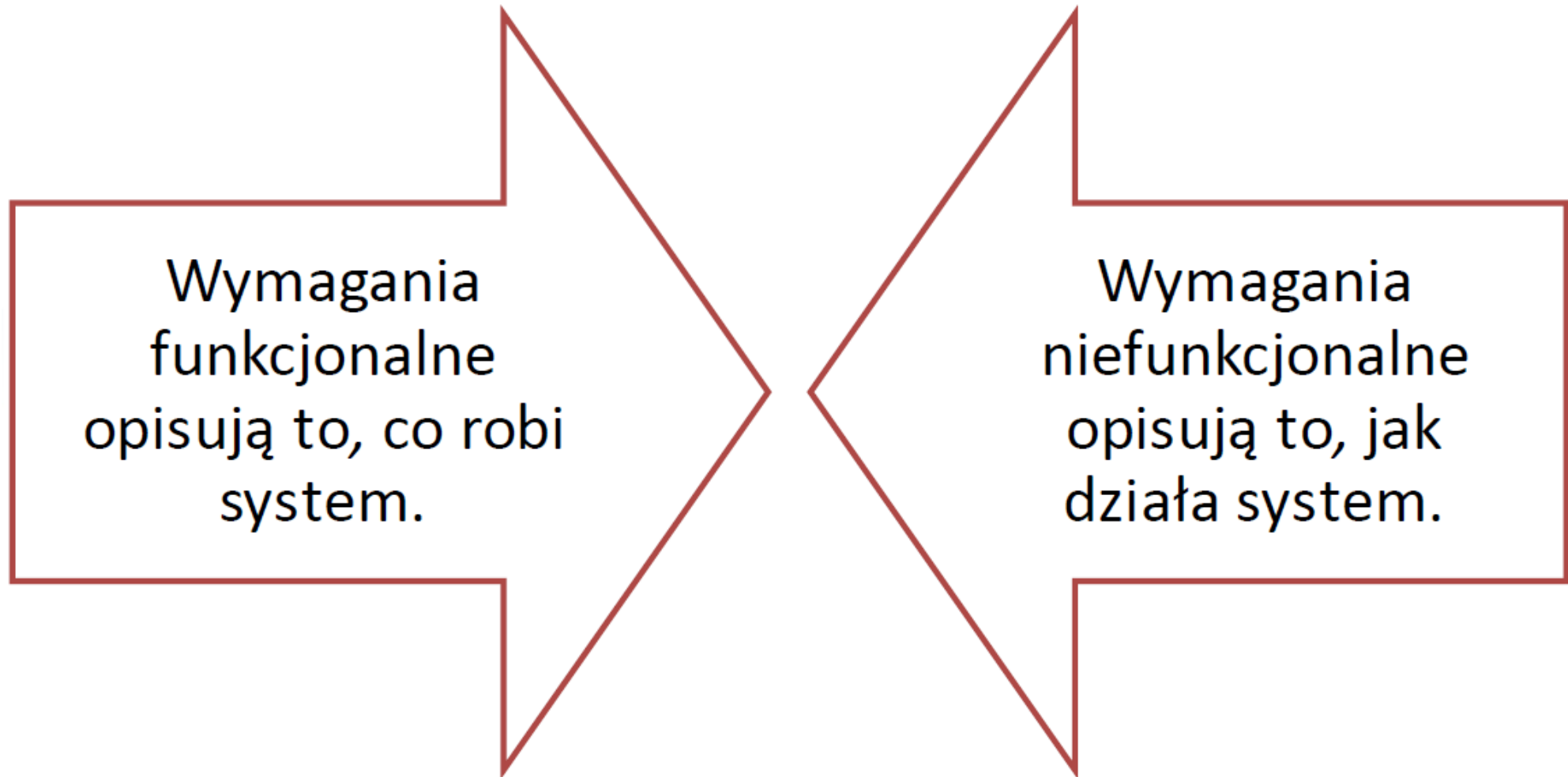
Specyfikacja powinna zawierać wymagania zaprezentowane z punktu widzenia klienta i w terminach dla niego zrozumiałych. Powinna stwierdzać,

CO ma zostać zrealizowane,

a **nie-jak** należy to zrobić.

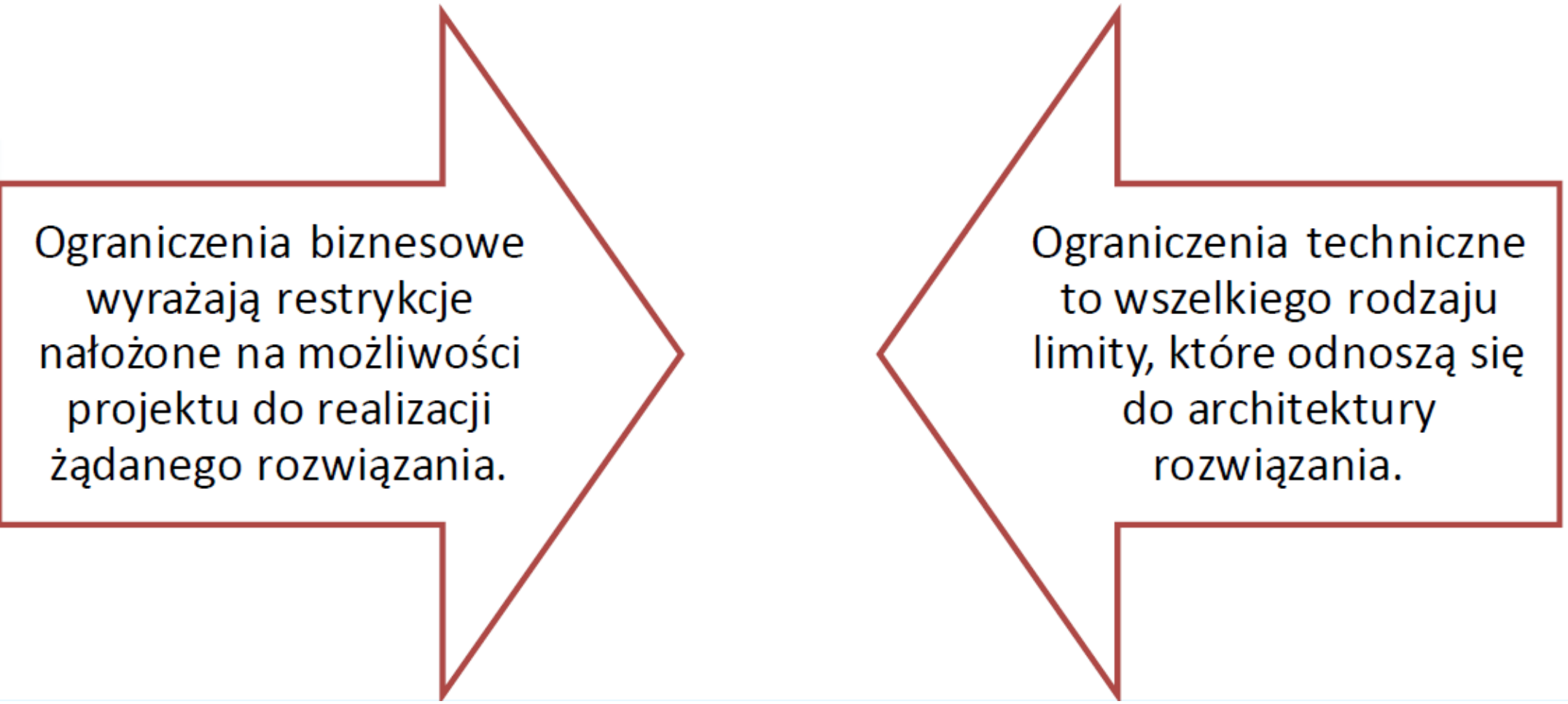
Specyfikacja nie może być więc próbą projektowania systemu.

Klasyfikacja wymagań



Ograniczenia:

Pewne warunki limitujące możliwości procesu projektowania, działania rozwiązania lub jego cyklu życia.



Ograniczenia biznesowe
wyrażają restrykcje
nałożone na możliwości
projektu do realizacji
żądanego rozwiązania.

The diagram consists of two large, stylized arrows pointing towards each other. The left arrow is white with a dark red outline and contains text about business constraints. The right arrow is also white with a dark red outline and contains text about technical constraints. The arrows are positioned horizontally, with their points facing each other in the center.

Ograniczenia techniczne
to wszelkiego rodzaju
limity, które odnoszą się
do architektury
rozwiązania.

- Wymagania produktowe:
 - System powinien sprawdzać, czy operator banku ma możliwość dostępu do określonego raportu.
 - System powinien autoryzować użytkownika przed uzyskaniem dostępu do aplikacji.
- Ograniczenia:
 - Program ma być napisany w Javie.
 - Projekt ma być realizowany zgodnie z podejściem Prince2.

Wymagania

A co jeśli mamy długą listę życzeń od Klienta?

Albo nie będziemy w stanie dotrzymać uzgadnianych terminów?



Zobowiązanie (ang. commitment)

Stopień zobowiązania się do spełnienia wymagania.

Zwykle definiowane za pomocą słów kluczowych, przypisywanych do wymagań wysokiego poziomu.



Priorytet (ang. priority)

Ocena ważności biznesowej/pilności wymagania.

Im wyższy priorytet, tym bardziej istotne jest wymaganie dla realizacji ogólnych celów projektu.

Priorytet zwykle określa się przy użyciu skal, na przykład wysoki, średni, niski, lub skala liczbowa: 1, 2, 3 etc



Krytyczność (ang. criticality)

Ocena ryzyka/ szkody w przypadku nie spełnienia wymagania.

Wyrażana za pomocą skal - wyższy poziom krytyczności oznacza, że brak spełnienia wymagania będzie bardziej dotkliwy dla całości rozwiązania (rozwiązanie nie będzie mogło być pełne).

MoSCoW

Cechy dobrej specyfikacji

1. **poprawna** – przeanalizowana i potwierdzona przez klienta;
2. **kompletna** – zawiera pełen zbiór wymagań i każde z nich jest kompletnie opisane;
3. **jednoznaczna** – zawiera wymagania podlegające tylko jednej semantycznej interpretacji;
4. **spójna** – nie zawiera wymagań wzajemnie sprzecznych;
5. **weryfikowalna** – zawiera procedury sprawdzające, czy wymaganie zostało zrealizowane;
6. **modyfikowalna** – pozwala w łatwy sposób dokonać zmian w wymaganiach, przy zachowaniu kompatybilności i spójności między nimi.
7. **sprawdzalna** – pozwala na śledzenie wpływu wszelkich zmian.

Źródła wymagań

Warsztaty

Sugestie
klientów

Obserwacja
użytkowników w pracy

Porównanie z konkurencją

Kontrakt

Wywiady

Usprawnienia zrobione
przez użytkowników

Wejście w rolę
użytkownika

Trenerzy i konsultanci

Prototypy

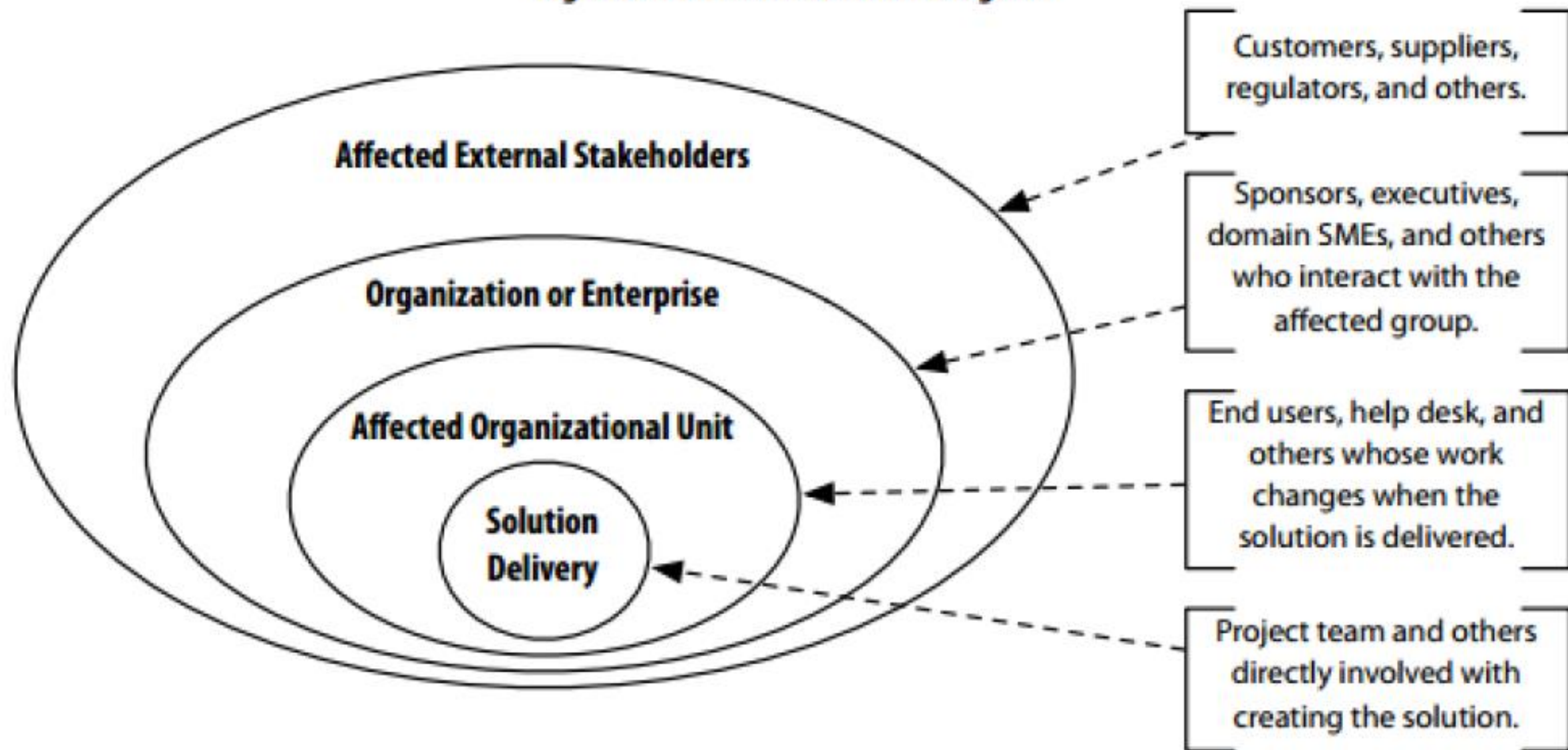
Koncepcja interesariusza

– Interesariusze:

- Osoby indywidualne, grupy lub organizacje, na które ma wpływ wynik projektu lub które są w jakiś sposób odpowiedzialne za ten wynik.
- Osoby indywidualne, grupy lub organizacje, które aktywnie uczestniczą w projekcie, lub na których interesy wynik realizacji projektu lub jego zakończenia mieć mieć jakiś wpływ.

- Różni interesariusze mogą mieć różne potrzeby i oczekiwania dotyczące planowanego rozwiązania.
- Ważna uwaga:
 - Zidentyfikuj wszystkich interesariuszy oraz ich potrzeby
 - Znajdź wspólne zrozumienie celu systemu

Figure 2–6: Stakeholder Onion Diagram



Pisanie dobrych wymagań

ID	
Nazwa	
Priorytet	
Krytyczność	
Wykonalność	
Ryzyko	
Źródło	
Właściciel	
Typ	

Jako [użytkownik], chcę
[potrzeba], by [cel]

Ryzyko

Ryzyko – czynnik, który w przyszłości może skutkować negatywnymi konsekwencjami; zazwyczaj opisywany jako **wpływ** oraz **prawdopodobieństwo jego wystąpienia**.

Zasada I –

- jeśli coś w projekcie może pójść niezgodnie z planem, to należy oczekiwać, że taka sytuacja będzie miała miejsce.

Zasada II

- lepiej unikać ryzyka niż nim zarządzać.

Ryzyko produktowe i projektowe

Ryzyko produktowe – jest to ryzyko bezpośrednio powiązane z przedmiotem testów.

Ryzyko projektowe – jest to ryzyko związane z zarządzaniem i kontrolą projektu; ryzyko zdolności projektu do osiągnięcia celów.

Ryzyko produktowe

- Źle dobrane testy
- Złożoność oprogramowania
- Niejasna dokumentacja

projektowe

- Problemy z firmami zewnętrznymi
- Czynniki techniczne
- Zmieniające się wymagania
- Braki zasobów
- Sztywny harmonogram
- Złe przywództwo
- Rozproszenie zespołu

Testowanie na podstawie ryzyka

Warto identyfikować ryzyka aby:

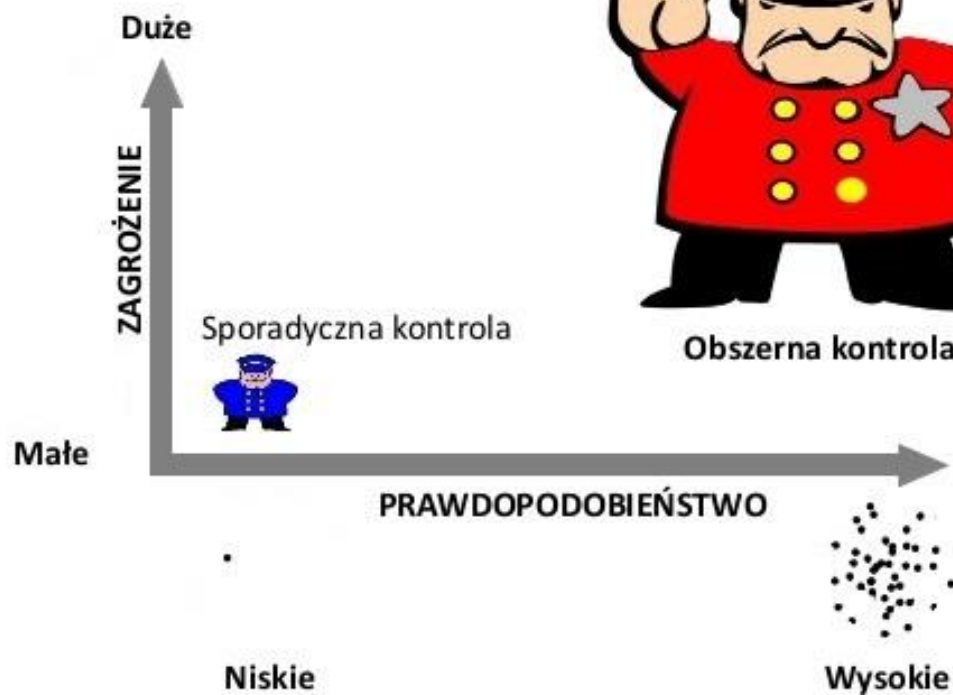
określić **jakie techniki** testowania będą najodpowiedniejsze

- **zdefiniować** odpowiedni **zakres** testów
- **uporządkować testy** pod względem ich wag, aby móc jak najwcześniej znaleźć najistotniejsze defekty
- określić co można zrobić, aby **zminimalizować prawdopodobieństwo wystąpienia ryzyka** lub jego potencjalny skutek

Testowanie na podstawie ryzyka

Testowanie na podstawie ryzyka - jest specyficznym rodzajem testowania, nastawionym na wykrycie i dostarczenie informacji o ryzykach.

Jest to jedna ze strategii testowych (metoda analityczna) w której testowanie koncentrowane jest w obszarach najwyższego zagrożenia.



Uproszczona macierz ryzyka

wpływ prawdopodobieństwo	Niski 1	Średni 2	Wysoki 3
Wysokie 3	3	6	9
Średnie 2	2	4	6
Niskie 1	1	2	3

Te ryzyka monitorujemy i planujemy akcje awaryjne

Te ryzyka ignorujemy

Te ryzyka monitorujemy

Kiedy zakończyć testowanie

Wiemy już **dłaczego testujemy**, wiemy **jakie są cele** testowania...

A **kiedy** należy **zakończyć testowanie**?

Podczas podejmowania decyzji **jak dużo** testów należy wykonać, powinno się wziąć pod uwagę poziom ryzyka, włączając w to ryzyko techniczne, ryzyko związane z bezpieczeństwem, a także ryzyko biznesowe oraz ograniczenia projektowe takie jak czas i budżet.

Testowanie powinno dostarczać interesariuszom wystarczającej ilości informacji do podjęcia świadomych decyzji o dopuszczeniu testowanego oprogramowania lub systemu do następnej fazy rozwoju lub przekazaniu go klientowi.

Struktura organizacyjna firmy

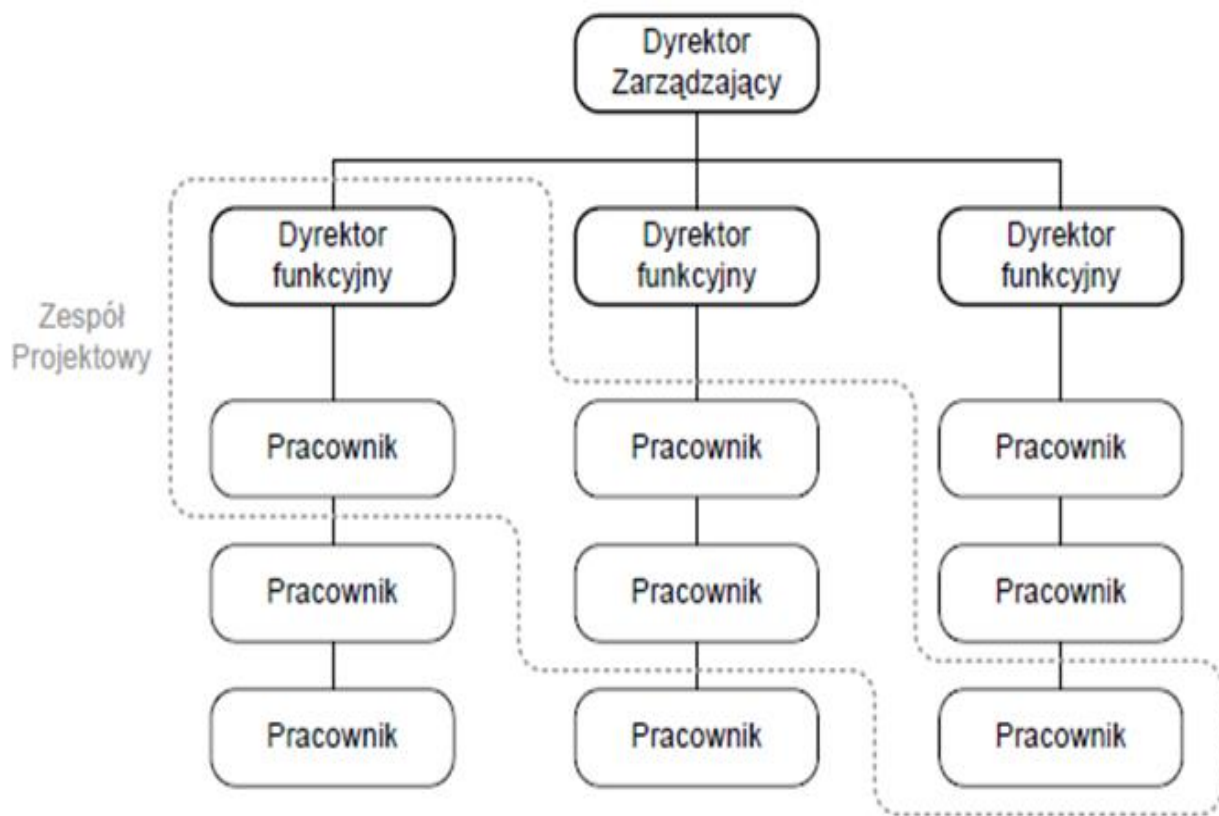
Zastanówmy się wspólnie, gdzie w organizacji (firmie) znajdują się testerzy.

Jaka jest rola testera w organizacji?

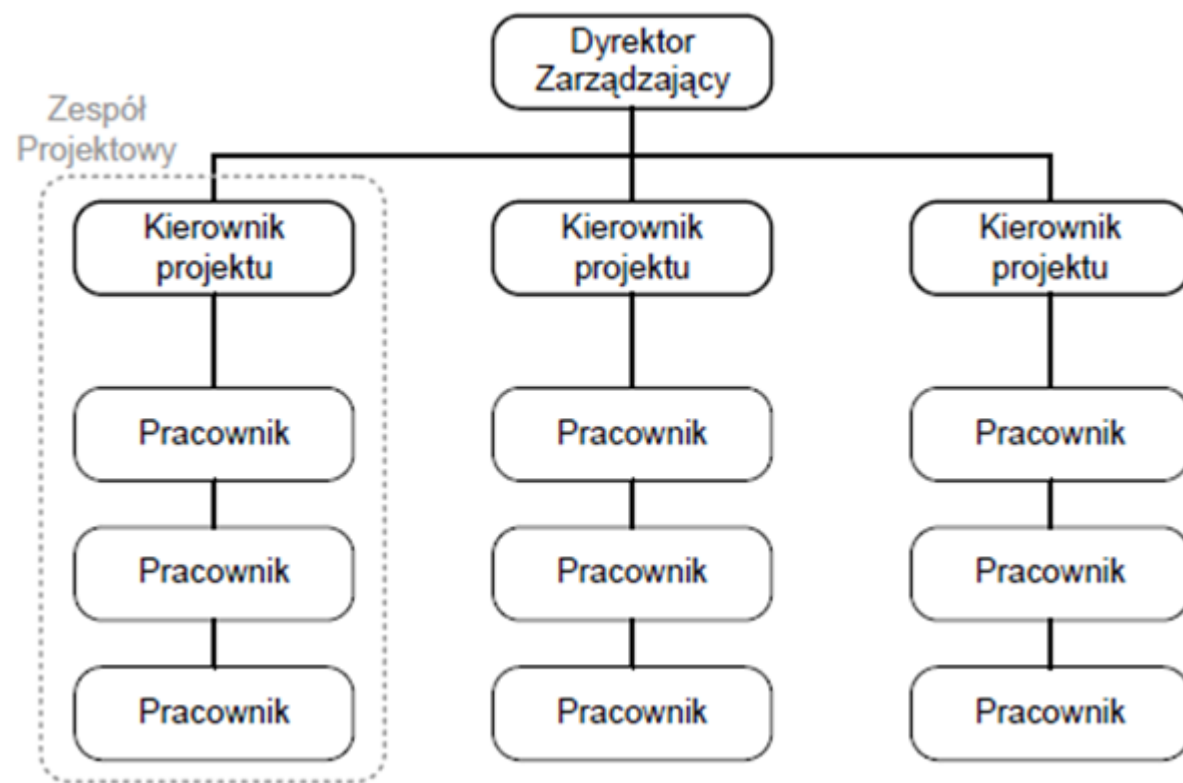
Gdzie w strukturze projektu umieścić testera?

Jakie znacie (wydaje Wam się, że mogą być) role związane z testowaniem?

Struktura macierzowa



Struktura projektowa



Struktura macierzowa

Macierzową (matrycową) strukturę organizacyjną buduje się wokół problemów lub projektów. Kolumny macierzy są odpowiednikami stałych, powtarzalnych funkcji (np. programowania, analizy, testów).

Zalety struktury macierzowej:

- tworzenie warunków do pracy interdyscyplinarnej (w jednym zespole pracują analitycy, programiści, testerzy, itd.),
- elastyczność,
- sprzyjanie wysokiej identyfikacji pracowników z celami (działamy w zespole dla wspólnego celu, widać postęp i wkład każdej z osób),
- samoczynne mechanizmy koordynacji,
- sprzyjanie powstawaniu zjawiska synergii (bliska współpraca np. projektantów z analitykami może tworzyć produkty jeszcze wyższej jakości – razem stworzycie coś, na co nie wpadłoby żadne z osobna)

Struktura projektowa

Przydział kompetencji jedynie do realizowanych projektów. Cechy:

- duża elastyczność i możliwość szybkiego reagowania na pojawiające się nowe problemy,
- zarządzaniem zajmuje się kierownik projektu, znika przełożony z działu funkcjonalnego.
- rywalizacja wewnątrz przedsiębiorstwa pomiędzy poszczególnymi projektami,
- brak poczucia stabilności i niepewność członków zespołów, co do sytuacji w firmie po zakończeniu projektu,
- możliwość pokrywania się niektórych działań, kiedy realizowanych jest równocześnie kilka projektów

Zespół projektowy

Zespół projektowy to grupa ludzi współpracujących ze sobą po to, by zrealizować określony cel.

Budując zespół, a potem zarządzając nim, powinno się kierować zasadą maksymalizowania jego użyteczności dla projektu.

Zespół projektowy służy określonemu celowi produkcyjnemu, a dodatkowo, jeśli jest dobry i zgrany to staje się podtrzymującą się strukturą, która potrafi rozwiązywać problemy i wspierać swoich członków.

Interesariusze projektu

Osoby lub inne organizacje, które uczestniczą w tworzeniu projektu (biorą czynny udział w jego realizacji) lub są bezpośrednio zainteresowane wynikami jego wdrożenia. Interesariusze mogą wywierać wpływ na daną organizację.

Przykład interesariuszy: udziałowcy, pracownicy, klienci, dostawcy, konkurenci, władze państwowe itd.

Role związane z testowaniem

Pojęcie roli nie jest tożsame z pojęciem stanowiska w firmie. Jedna **osoba**, będąc **zatrudniona na konkretnym stanowisku może odgrywać różne role w procesie testowym.**

Typowe role związane z testowaniem:

- dyrektor testów

- menedżer testów

- kierownik testów (lider testów)

- architekt testów

- analityk testów

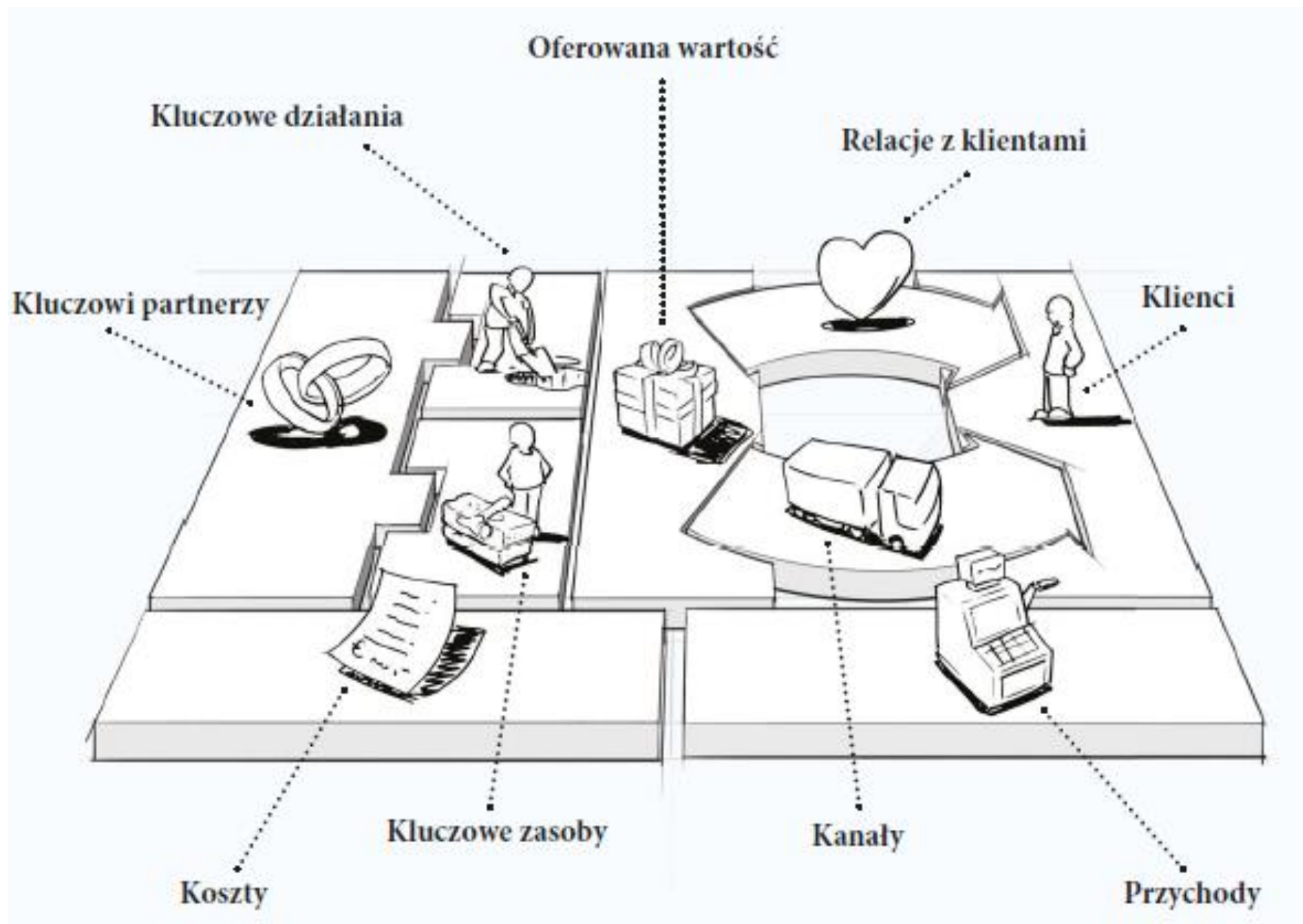
- inż. ds. automatyzacji testów

- administrator testów

Model biznesowy firm z branży IT

Model biznesowy – jest to plan, który tworzy przedsiębiorstwo w celu wygenerowania przychodu i maksymalizacji zysku operacyjnego. Określa relacje pomiędzy uczestnikami rynku, informuje jak przedsiębiorstwa działają, tj. w jaki sposób tworzą wartość dla klientów, towary i usługi oraz z czego czerpią zyski. (Źródło: <http://it-consulting.pl/>)

Innymi słowy: jak optymalnie wykorzystać posiadane zasoby, aby dostarczyć klientowi wartość i jednocześnie zapewnić własnej firmie rentowność.



Typy kontraktów w branży IT

„**Time and material**” kontra „**fixed price**”

Są to dwa najpopularniejsze typy kontraktów. Określają w jaki sposób firma świadczy swoje usługi.

W rozliczeniu typu „fixed price” **cena** usługi jest określana **na początku projektu**, na podstawie specyfikacji wymagań.

W rozliczeniu typu „time and material” **klient płaci** za czas pracy kapitału ludzkiego, najczęściej **wg stawek godzinowych lub dziennych**.

Time and material – wady i zalety

Zalety:

- Nie trzeba czekać na ukończenie specyfikacji
- Łatwy sposób tworzenia dynamicznych struktur projektowych
- Mniejsze ryzyko dostawcy implikuje niższą cenę
- Łatwość zmiany wymagań w trakcie trwania projektu

Wady:

- Całkowita cena jest trudna do przewidzenia
- Klient przejmuje na siebie ryzyko błędnej **estymacji**

Fixed price – wady i zalety

Zalety:

- Cena z góry znana za całość prac
- Ryzyko niedoszacowania prac jest po stronie dostawcy, a nie po stronie Klienta

Wady:

- Zakres trudno zmienić w czasie trwania projektu
- Dostawca uwzględni wyższe ryzyko w cenie