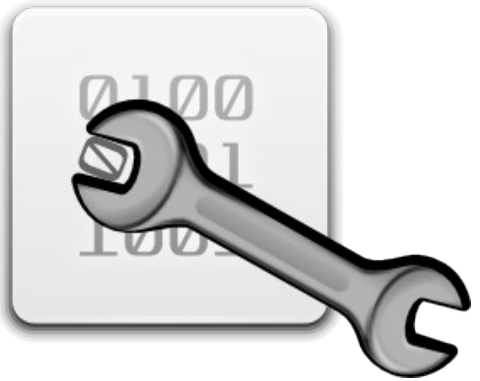
Podstawy inżynierii oprogramowania



Cykl życia oprogramowania

Aleksander Lamża
ZKSB · Instytut Informatyki
Uniwersytet Śląski w Katowicach

aleksander.lamza@us.edu.pl

Zawartość

Wszystko zaczyna się od pomysłu (albo potrzeby)

Czas i pieniądz

Z pewnością każdy klient zada te dwa pytania:

Ile to będzie kosztować?

I dlaczego tak drogo?

Kiedy będzie gotowe?

Ale ja to potrzebuję na już!

Doskonałe projekty

Doskonały rozwój oprogramowania prowadzi do dostarczenia

tego co potrzebne

na czas

po ustalonych kosztach

Jak doprowadzić do katastrofy

klient

Jak już mówiłem, aby nasza korporacja spełniała normę ISO 138534–13b, musimy mieć nowy system zarządzania zasobami ludzkimi. Przede wszystkim system musi zapewniać wydajny przepływ kluczowych informacji, a także integrować dobre praktyki zaimplementowane w poszczególnych działach naszej korporacji. Bla, bla bla...

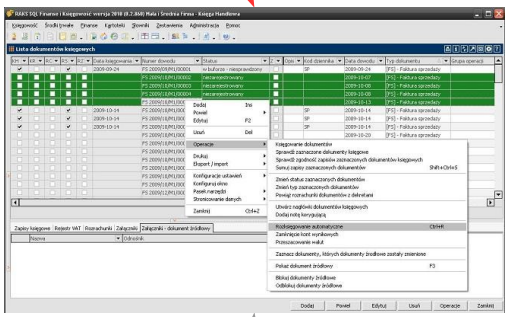
Tak, tak... Uhm... Oczywiście...

programista

Jak doprowadzić do katastrofy

Dobra, mniej więcej wiem o co chodzi,
więc nie ma na co czekać
– zabieram się za pisanie!

szał
programowania



Gotowe!!! Patrz i podziwiaj...

Jak doprowadzić do katastrofy

Przyznaję, **kod nie jest najpiękniejszy**, ale działa. **Trudno go też modyfikować i rozszerzać**, ale mam nadzieję, że aplikacja spodoba się klientowi, więc nie będę musiał już w tym grzebać.

Jak doprowadzić do katastrofy

Pójście na żywioł i spontaniczne programowanie przeważnie prowadzi do mniejszej lub większej katastrofy...

Jak doprowadzić do katastrofy

Chwila prawdy – pokazujemy efekty naszej pracy klientowi

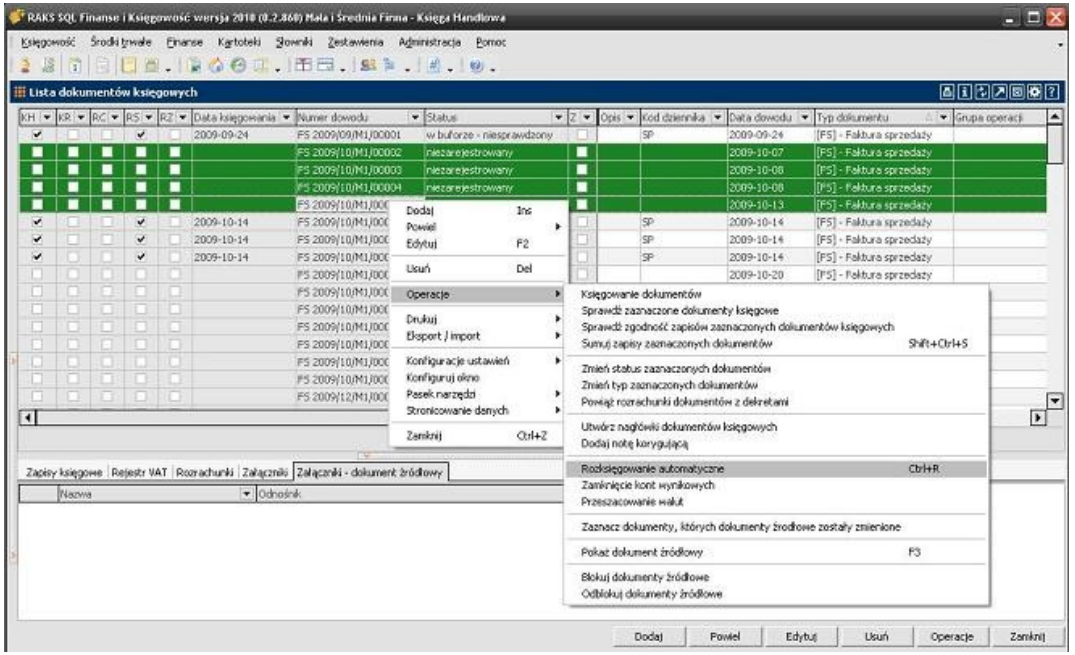
Jak doprowadzić do katastrofy

Zastanówmy się, co by zrobił rasowy inżynier...

ale taki prawdziwy, od dróg i mostów

Oczami inżyniera – budujemy mosty

Wytęż wzrok i znajdź różnice*

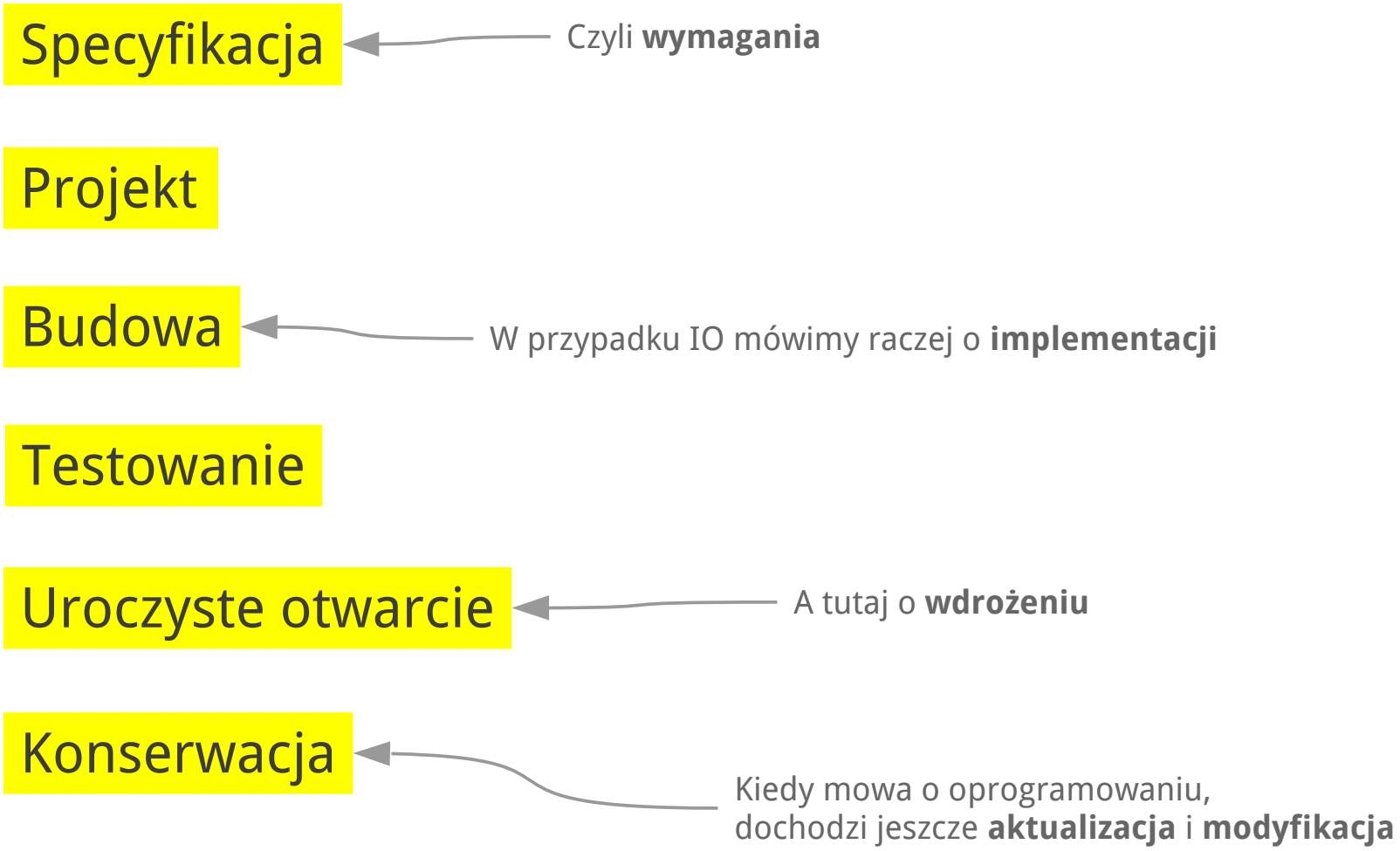


* Odpowiedź niebawem

Budujemy mosty

Gdybyś miał zbudować most, złapałbyś od razu nitownicę i wziął się do roboty? Raczej nie...

Taka kolejność działań byłaby chyba lepsza:



Budujemy mosty

Ile czasu mija od **określenia wymagań** do **uroczystego otwarcia**?

DUŻO

Czy istnieje możliwość **zmiany specyfikacji** po rozpoczęciu robót?

RACZEJ NIE

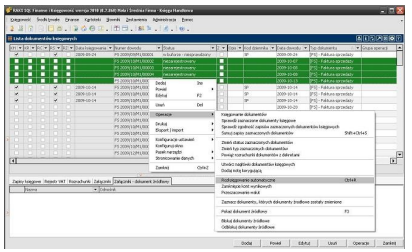
Czy po zakończeniu robót można **modyfikować** lub **aktualizować**?

NIE

Np. dobudować jezdnię

Budujemy mosty

Wracamy do naszych różnic



Ile czasu mija od **określenia wymagań** do **uroczystego otwarcia**?

~~DUŻO~~ **JAK NAJMNIEJ**

Czy istnieje możliwość **zmiany specyfikacji** po rozpoczęciu robót?

~~RACZEJ NIE~~ **ZDECYDOWANIE TAK**

Czy po zakończeniu robót można **modyfikować** lub **aktualizować**?

~~NIE~~ **KONIECZNIE**

Tego chcemy

Budujemy mosty

Taki model pracy jest nazywany **modelem kaskadowym**.

Sam pomysł takiego zorganizowania pracy nie jest zły, ale **nie nadaje się do tworzenia oprogramowania**.

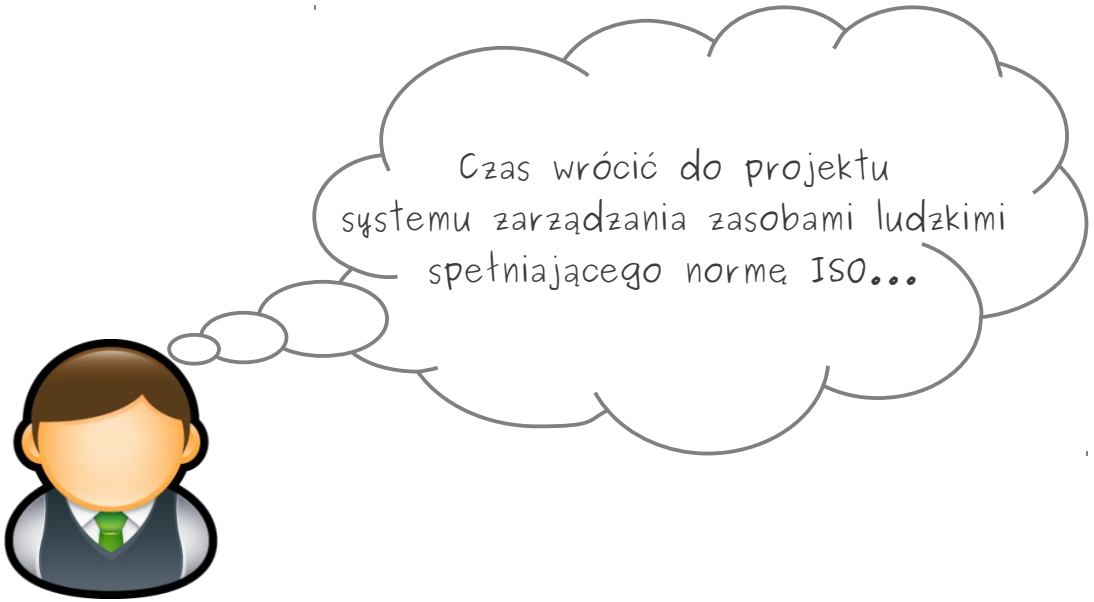
Dwa podstawowe problemy:

Czyli **zbyt długie przerwy** w kontaktach z klientem

Zbyt długi czas między określeniem wymagań a oddaniem produktu.

Niemożliwe lub utrudnione wprowadzanie zmian w projekcie.

A zmiany są nieuniknione!



Panta rhei, czyli wszystko się zmienia, a zwłaszcza wymagania

Po jakimś czasie...

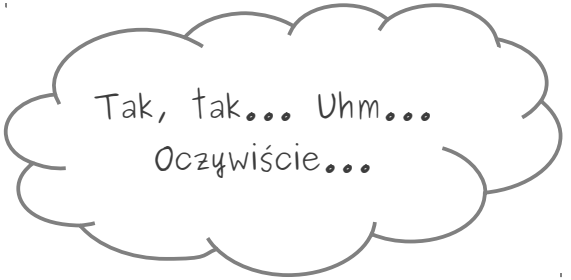
Już lepiej...
Problem w tym, że normę ISO 138534-13b
zastąpiła norma ISO 138534-13c.
W związku z tym **zmieniło się** kilka spraw...
Poza tym wolałbym, żeby ten przycisk...

W każdym projekcie można być pewnym tylko jednego
– tego, że nastąpią

ZMIANY

Problemy

Jakby tego było mało, dochodzą jeszcze inne problemy.



klient

Klient nie umie przekazać, na czym mu zależy.

Programista nie umie zadawać właściwych pytań.

Klient nie wie, na czym mu zależy.

Programista nie zna się na danej dziedzinie.

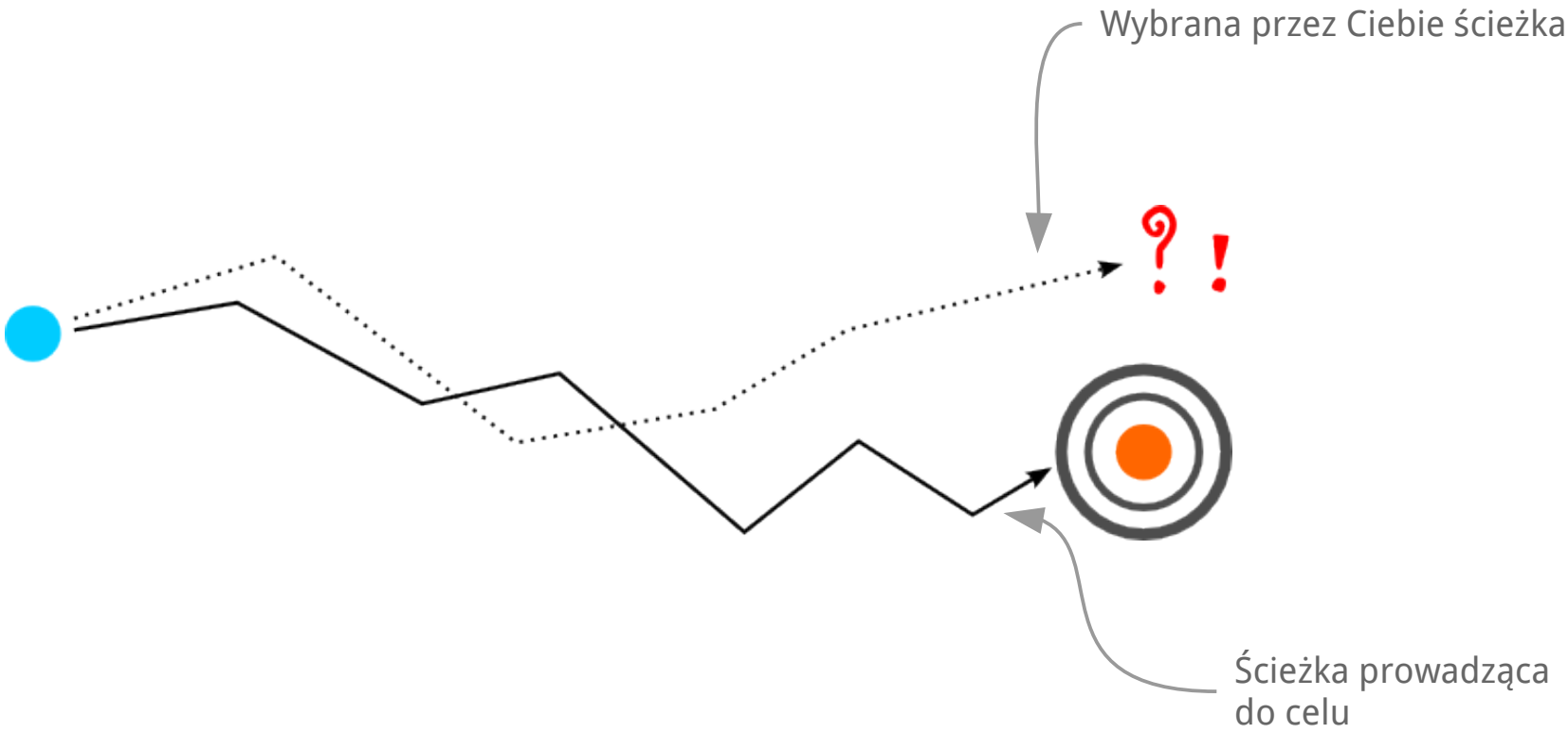
Klient nie zna wszystkich szczegółów.

itd., itp...

programista

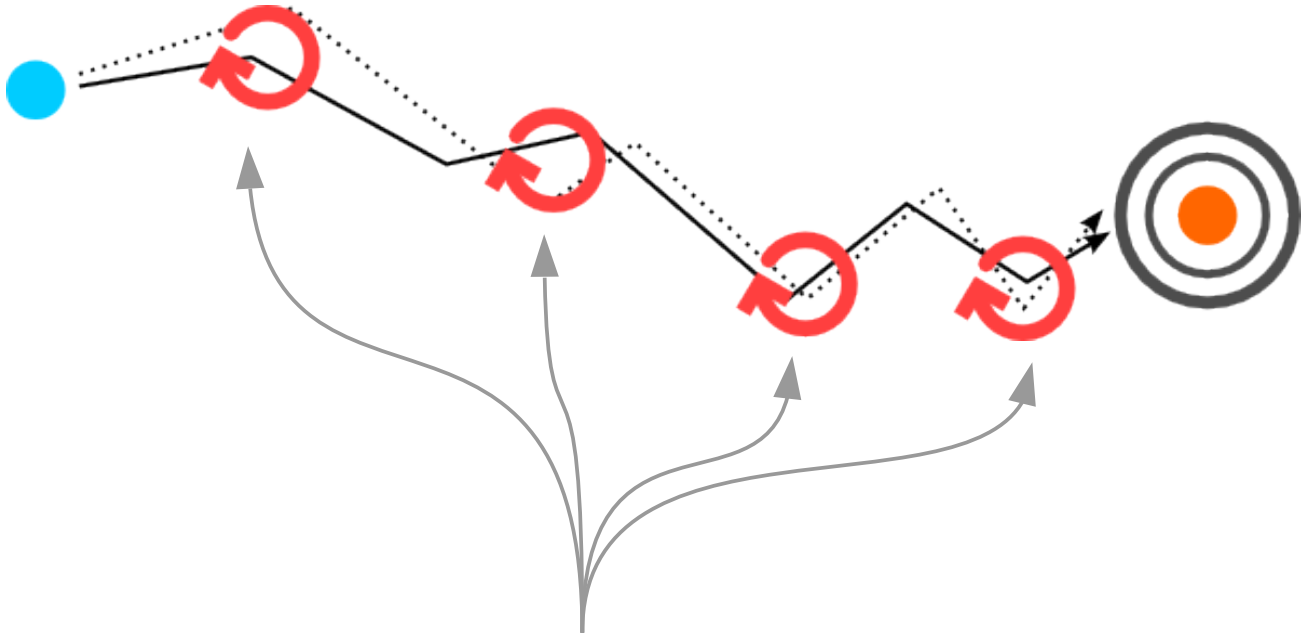
Problemy

W takich warunkach trudno trafić do celu...



Rozwiązanie (niektórych) problemów

A gdyby tak częściej widywać się z klientem i pytać go o zdanie?



Co jakiś czas pokazujesz klientowi aktualną wersję
i pytasz, czy projekt zmierza we właściwą stronę.

Iteracje

Chodzi więc tylko o to, by cyklicznie spotykać się z klientem?

NIE

Chodzi o to, by **cały proces** tworzenia oprogramowania zorganizować w oparciu o **cykliczne** wykonywanie tych samych bloków zadań, a inaczej mówiąc

ITERACJI

...czyli ?

Iteracje

Każda iteracja jest miniprojektem, na którą składają się cztery główne elementy:

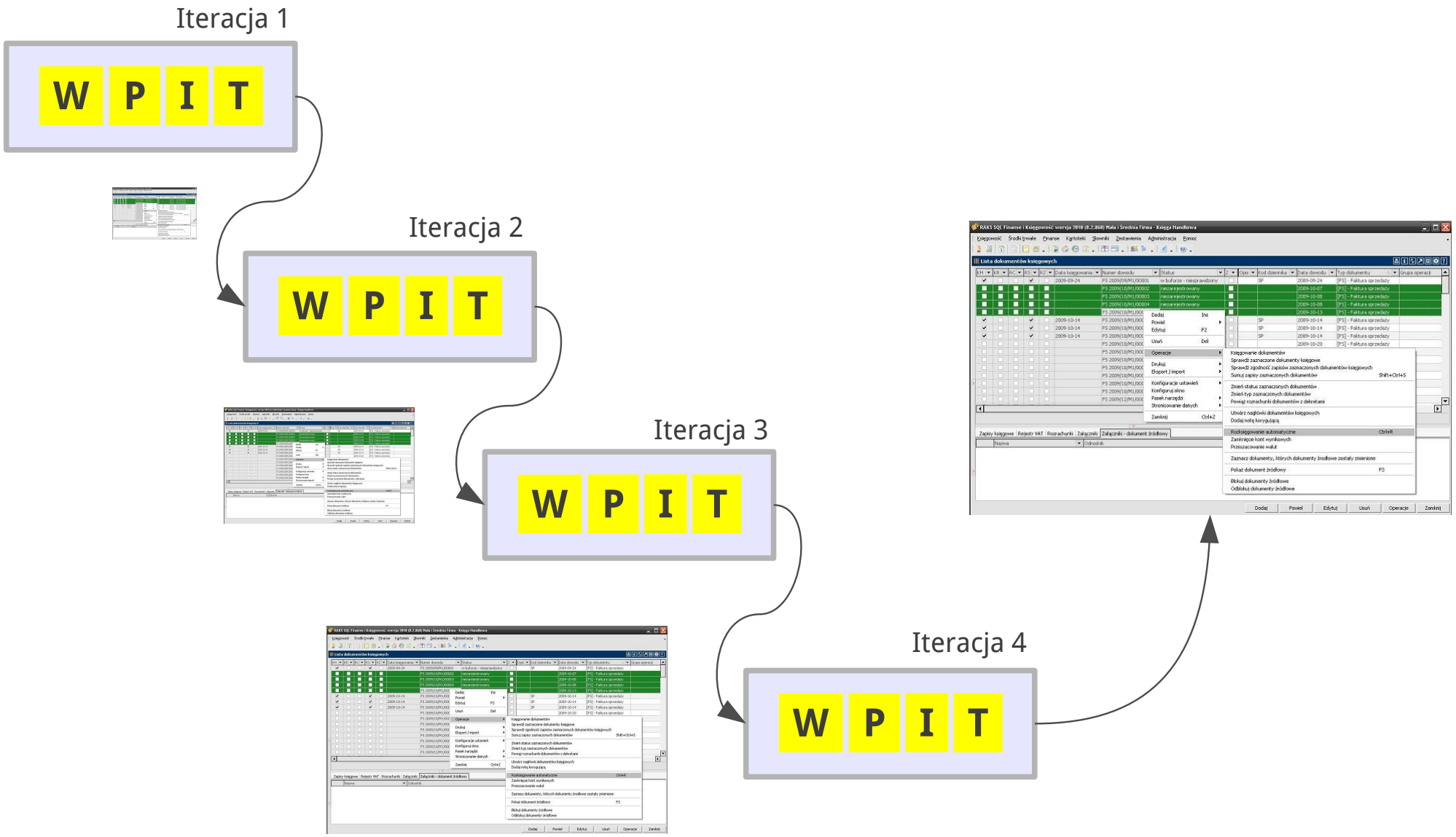
wymagania **projekt** **implementacja** **testy**

Czy to Ci czegoś nie przypomina?
Pamiętasz mosty?

Na każdym etapie prac skupiamy się na **wybranych wymaganiach** i tworzymy **działające oprogramowanie** spełniające te wymagania.

Iteracje

Proces rozwoju oprogramowania wygląda więc następująco:



Iteracje

Jak wybierać wymagania?

w skali od 10 do 50
(10 – najważniejsze)

Decydującą rolę odgrywa klient – to on wskazuje **priorytety** poszczególnych wymagań.

Zadaniem zespołu programistów jest **oszacowanie czasu** potrzebnego na zrealizowanie danej funkcji.

Wprowadzanie danych

Priorytet: 10
Czas: 10 dni

Generowanie raportów

Priorytet: 20
Czas: 10 dni

Eksportowanie danych do systemu księgowego

Priorytet: 40
Czas: 2 dni

Integracja z programem pocztowym

Priorytet: 50
Czas: 7 dni

System pomocy

Priorytet: 50
Czas: 10 dni

Wewnętrzna poczta

Priorytet: 20
Czas: 9 dni

Edytowanie danych

Priorytet: 10
Czas: 5 dni

Lista zadań do wykonania

Priorytet: 10
Czas: 3 dni

Te przykłady funkcji to duże uproszczenie (a nawet przekłamanie).
Więcej na temat określania wymagań powiem na innym wykładzie.

Iteracje

Teraz grupujemy funkcje w iteracje

długość iteracji może być różna, ale często przyjmuje się 20 dni

Iteracja 1

Wprowadzanie danych

Priorytet: 10
Czas: 10 dni

Edytowanie danych

Priorytet: 10
Czas: 5 dni

Lista zadań do wykonania

Priorytet: 10
Czas: 3 dni

Iteracja 2

Generowanie raportów

Priorytet: 20
Czas: 8 dni

Wewnętrzna poczta

Priorytet: 20
Czas: 9 dni

Iteracja 3

Eksportowanie danych do systemu księgowego

Priorytet: 40
Czas: 2 dni

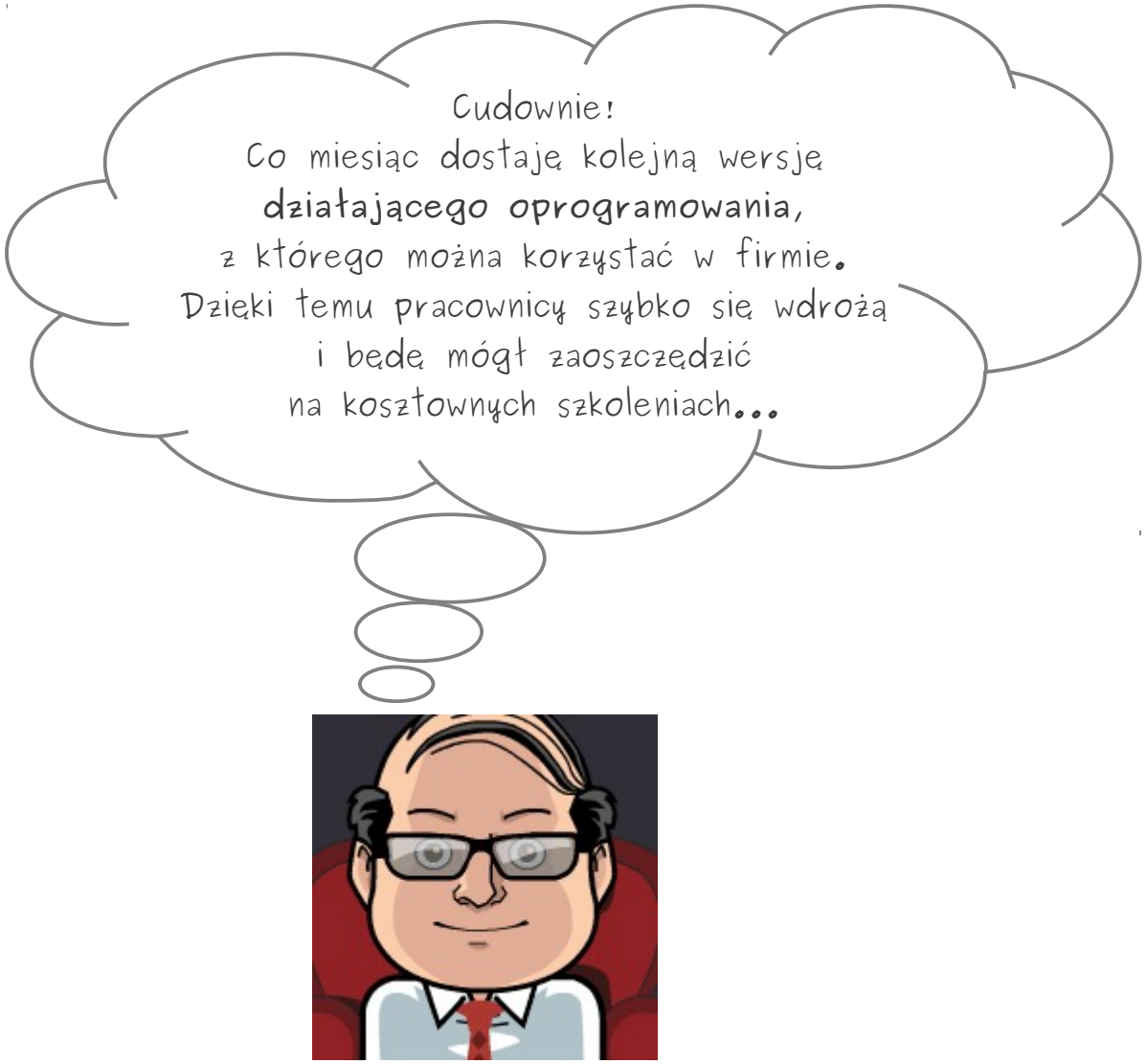
Integracja z programem pocztowym

Priorytet: 50
Czas: 7 dni

System pomocy

Priorytet: 50
Czas: 10 dni

Iteracje


Cudownie!
Co miesiąc dostaję kolejną wersję
działającego oprogramowania,
z którego można korzystać w firmie.
Dzięki temu pracownicy szybko się wdrożą
i będę mógł zaoszczędzić
na kosztownych szkoleniach...

Iteracje – podsumowanie

Iteracje pozwalają zachować **prawidłowy kierunek prac**.

A więc podążamy wprost do celu, jakim jest utworzenie **oprogramowania zgodnego z wymaganiami.**

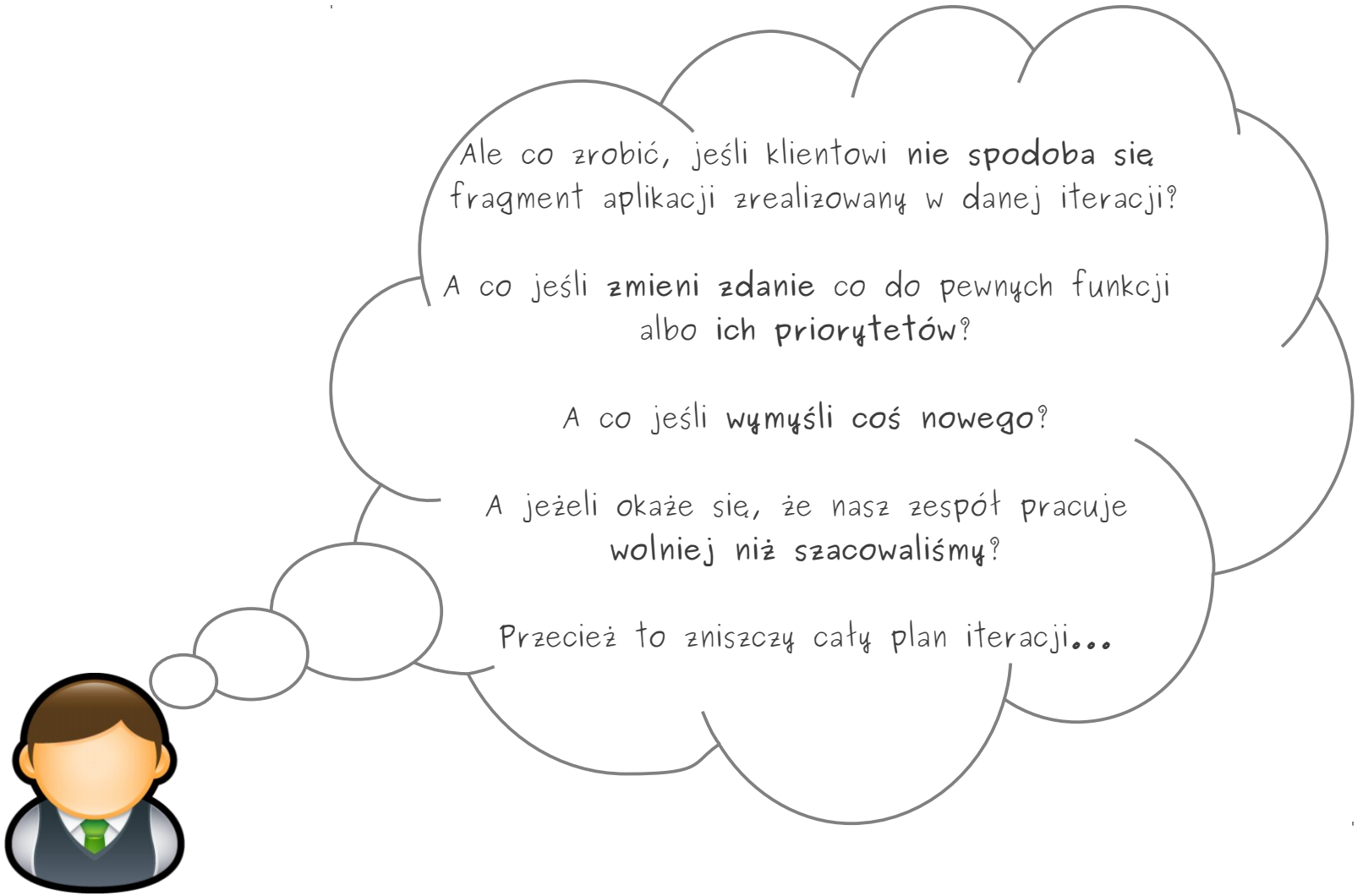
Na każdym etapie rozwoju projektu iteracje prowadzą do **utworzenia działającego oprogramowania**.

Klient może już korzystać z częściowo stworzonego oprogramowania.
Na każdym etapie dokładamy kolejne funkcje.

W każdej iteracji uzyskujemy **informacje zwrotne od klienta**.

Klient może szybko **sygnalizować swoje uwagi** oraz **informować o zmianach.**

Zawsze jest jakieś „ale”...



O tym na następnym wykładzie.