

# Teoria testów #2

# Robert Kaszubowski

Dawniej Java Developer  
Tester automatyczny & manualny  
Test lead  
Test team leader i test manager  
People manager – mentor i coach  
Project coordinator  
Global head of testing automation CC in Epam Poland  
Trener ☺

<http://robertkaszubowski.com/>  
robertkaszubowski147@gmail.com

Hobby: sporty walki, trening siłowy, gry wideo, dobrej jakości sprzęt audio





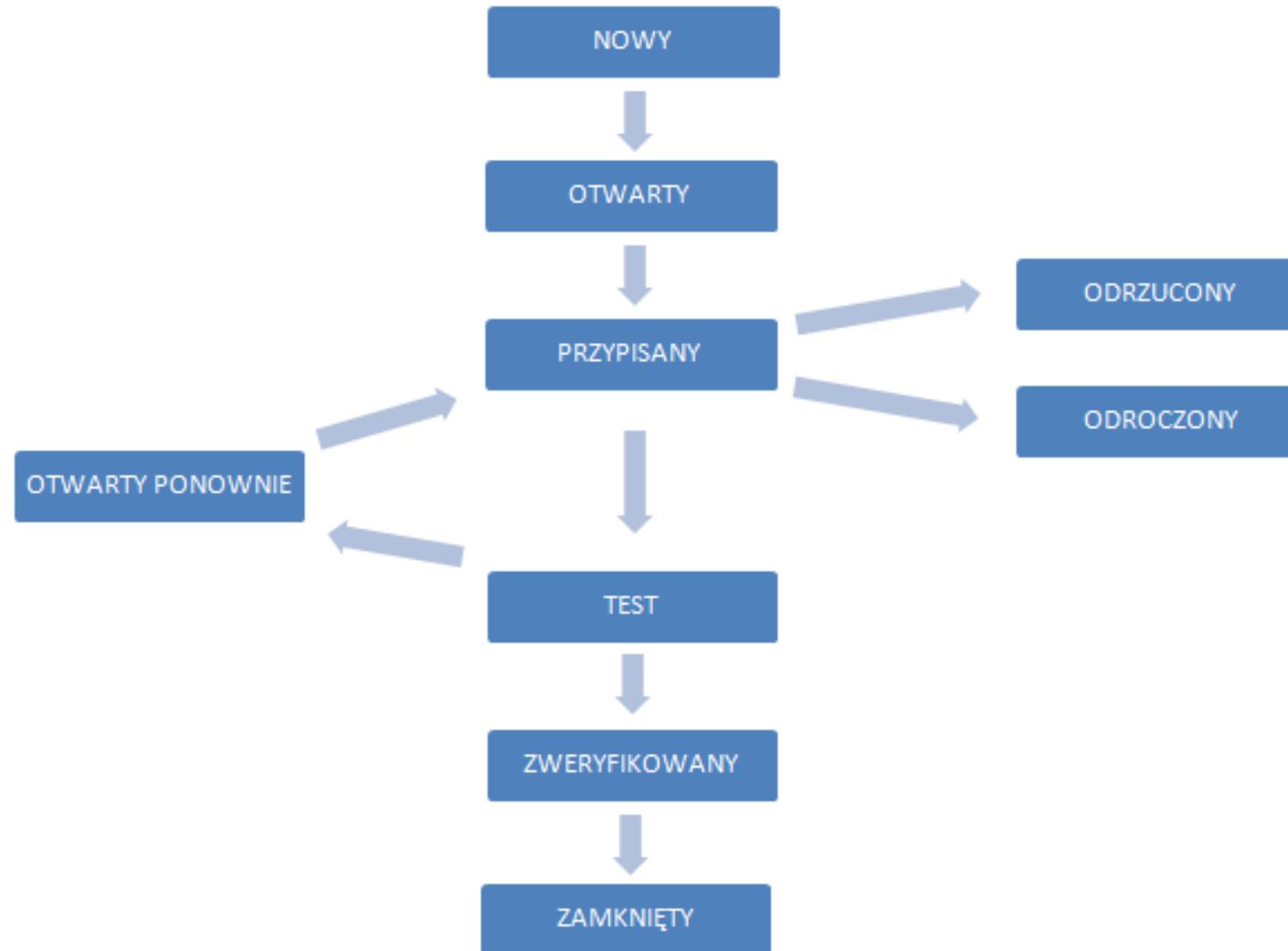
Jakie informacje powinien zawierać raport błędu?

# Przykładowy defekt

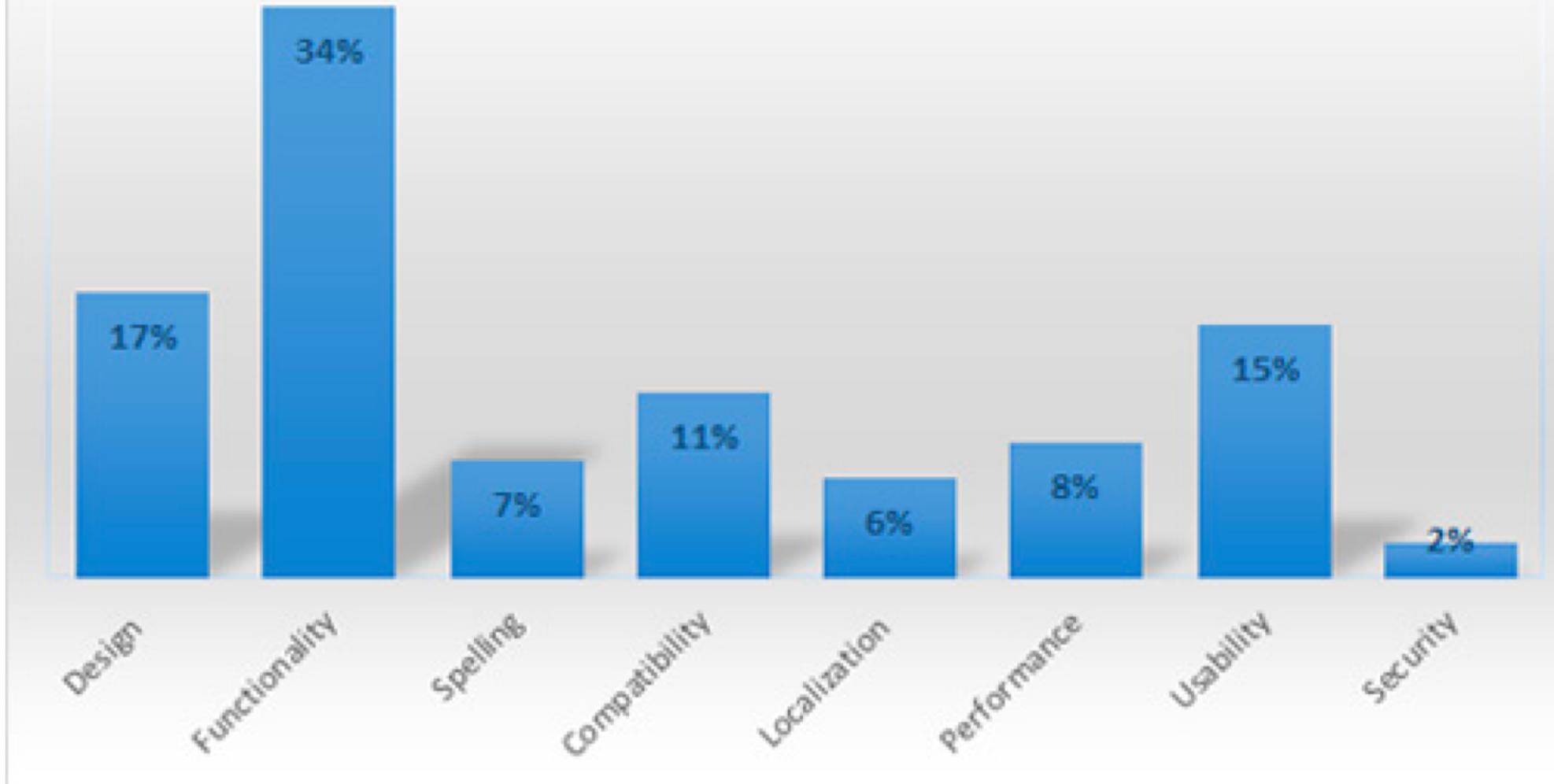
<b>Identyfikator defektu:</b>	D-1		
<b>Tytuł zgłoszenia</b>	Błąd 404 po wejściu na stronę główną aplikacji webowej		
<b>Twórca defektu:</b>	Tester	<b>Data wykrycia:</b>	11/12/2015
<b>System (przeglądarka) na której przeprowadzono test:</b>	Firefox 33	<b>Powiązany scenariusz testowy:</b>	TC-1
<b>Wersja testowanej aplikacji:</b>	1.5	<b>Priorytet:</b>	Wysoki
<b>Status:</b>	Nowy	<b>Kategoria:</b>	Funkcjonalny
<b>Opis zgłoszenia:</b>	Błąd 404 po wejściu na stronę główną aplikacji		
<b>Kroki do odtworzenia błędu:</b>	1. Otwórz stronę główną w przeglądarce - www.test.com.pl		
<b>Aktualny wynik testu:</b>	Po wejście na stronę główną aplikacji pojawia się błąd 404		
<b>Spodziewany wynik testu:</b>	Po wejście na stronę główną powinien pojawić się ekran logowania		

Severity vs priority

# Cykl życia defektu



## Type of bugs overall



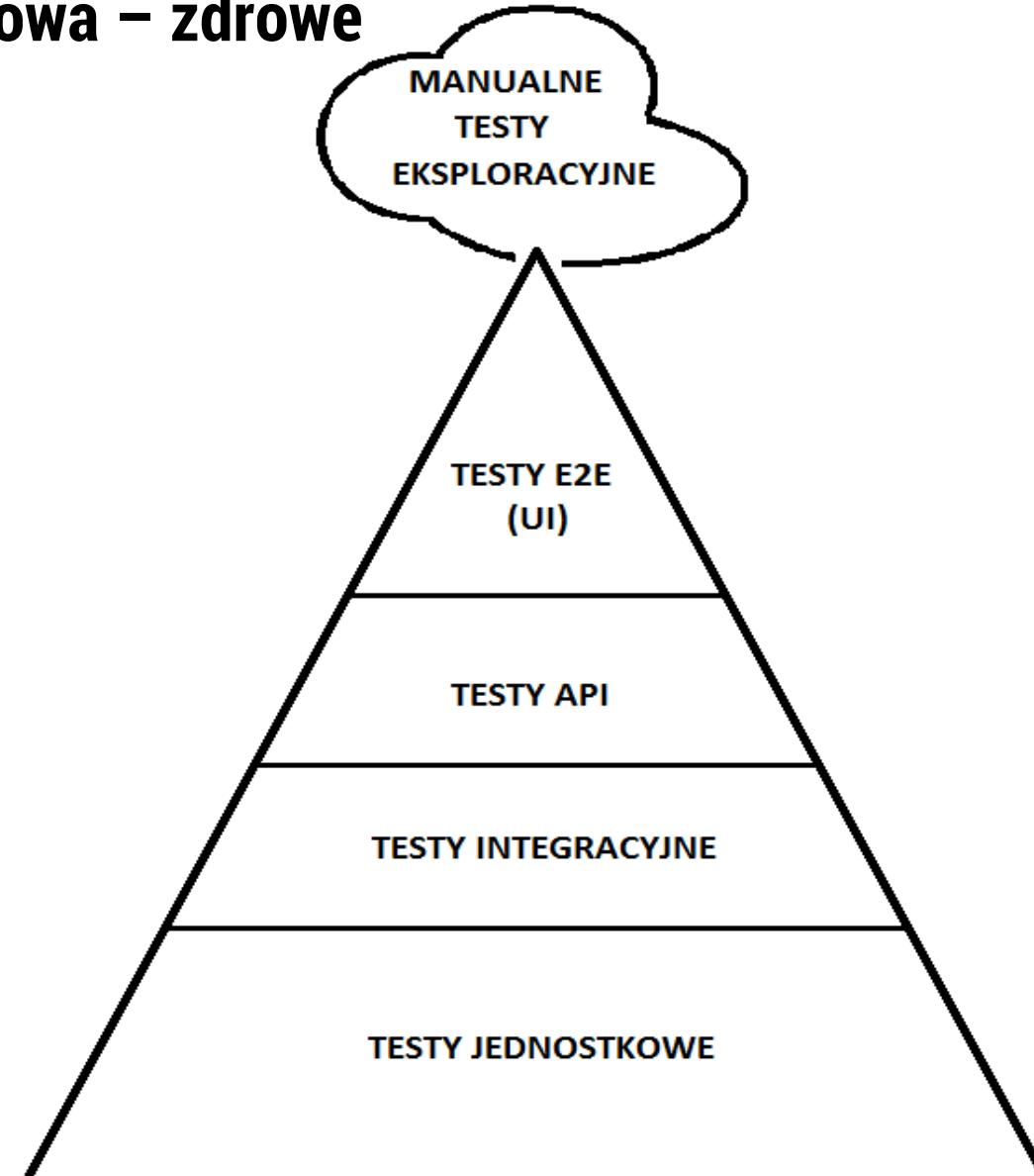
## Zadanie #2

Każdy wykonuje dokładne testy eksploracyjne aplikacji:  
<http://adam.goucher.ca/parkcalc/>

Starajcie się zepsuć aplikacje, bądźcie kreatywni.  
Znalezione błędy raportujemy lokalnie na dysku, tak by nie dawać  
ściągać ☺  
Na koniec dzielicie się swoimi bugami z sąsiadem i robicie sobie  
review.

Czas: 1.5h

# Piramida testowa – zdrowe podejście



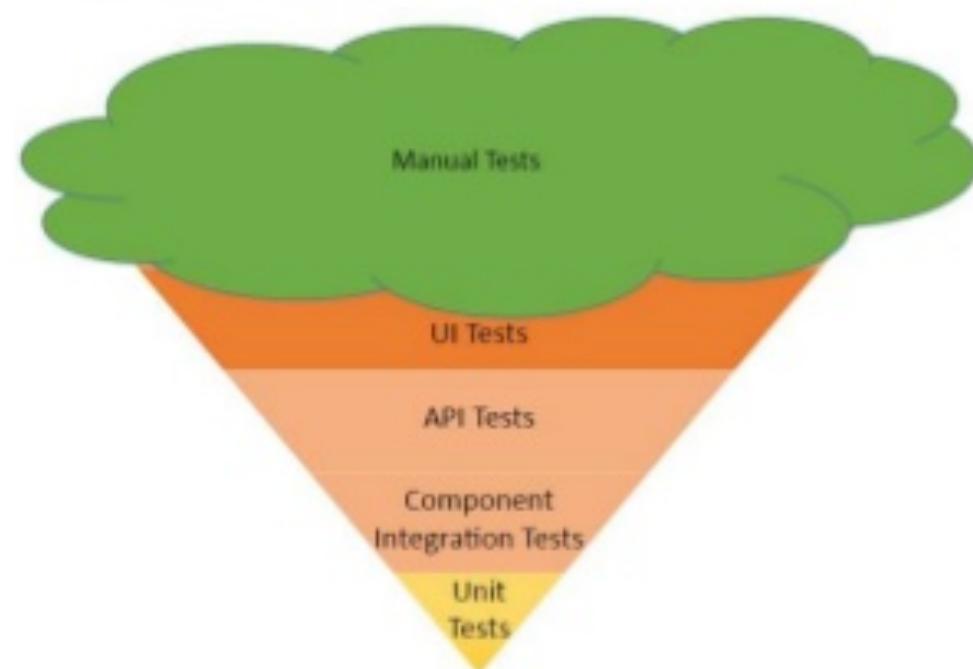
Znajduj bugi vs zapobiegaj ich powstaniu

# Piramida testowa - antywzorce

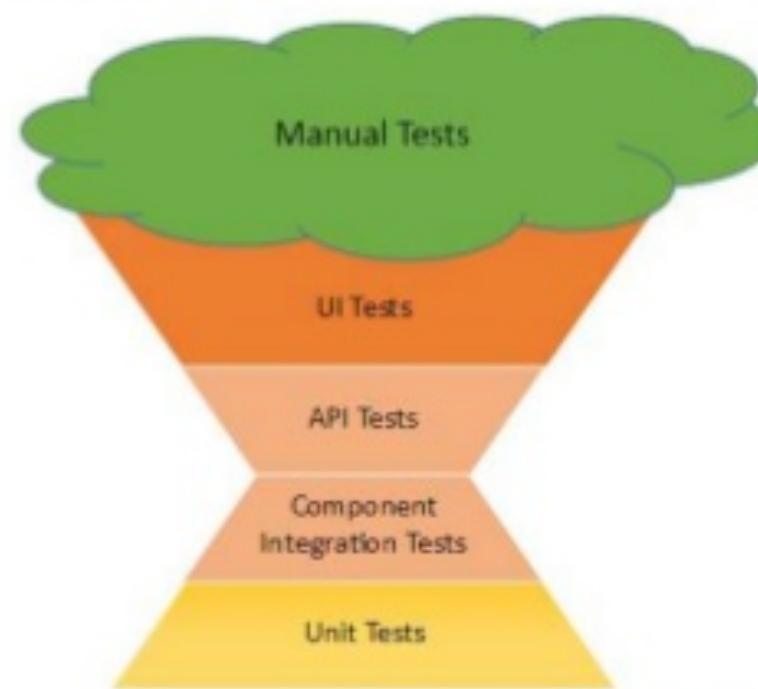


## Anti-patterns

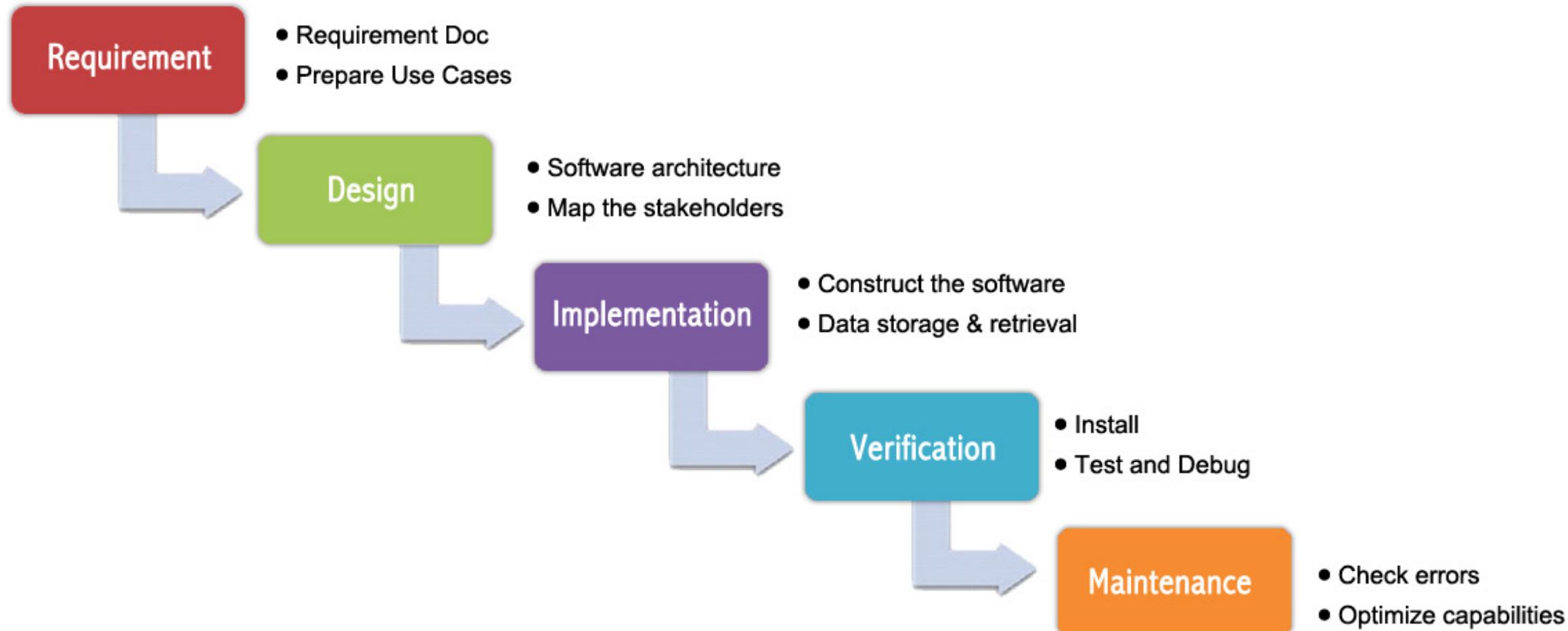
Ice cream cone



Hour-glass



# Waterfall

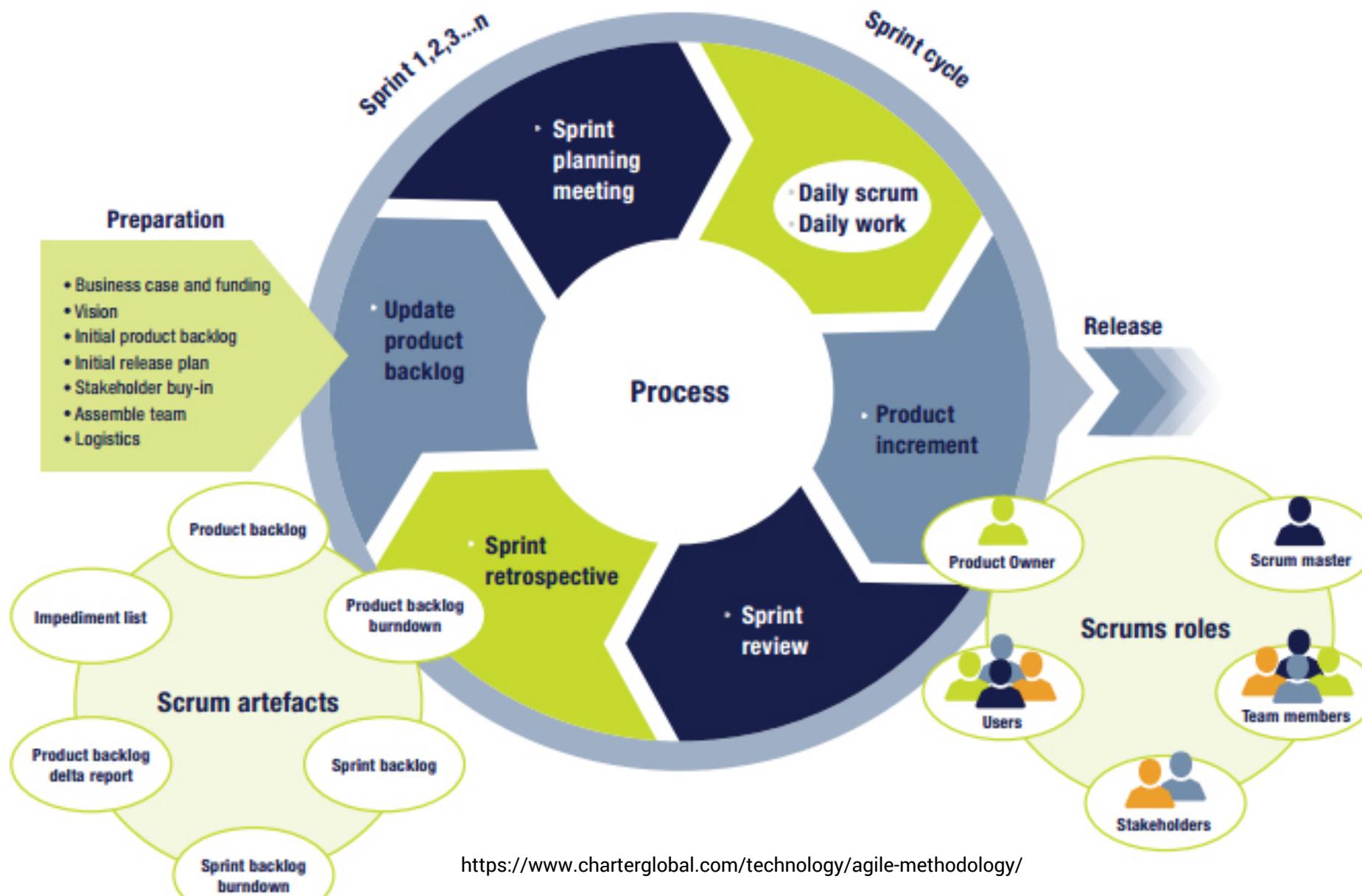




Jakie widzisz problemy z testami w tym podejściu?

# Agile

# Agile/Scrum



# Testowanie w Agile vs Waterfall

Agile	Waterfall
Black box and white box testing, deep knowledge of internal workings of the application	Black box testing, no need for deep knowledge of the internal workings of the application
Main function is to help produce killer applications	Main function is to certify the quality of the product
Work in parallel with development, testing as soon as new source code is produced	Work in batches at the end of milestones.
Heavily based on automated testing	Not much need for automated testing, if any. Only some UI automated testing is performed
Integrated with the development team. There is only one team	A completely separate team from development.
Key role interacting with the business. They make sure that the expectations from the customer (acceptance criteria) are met.	Not much interaction with the business. Their purpose is to make sure that the application meets whatever is specified in the requirements document.

# Techniki estymacji



WBS

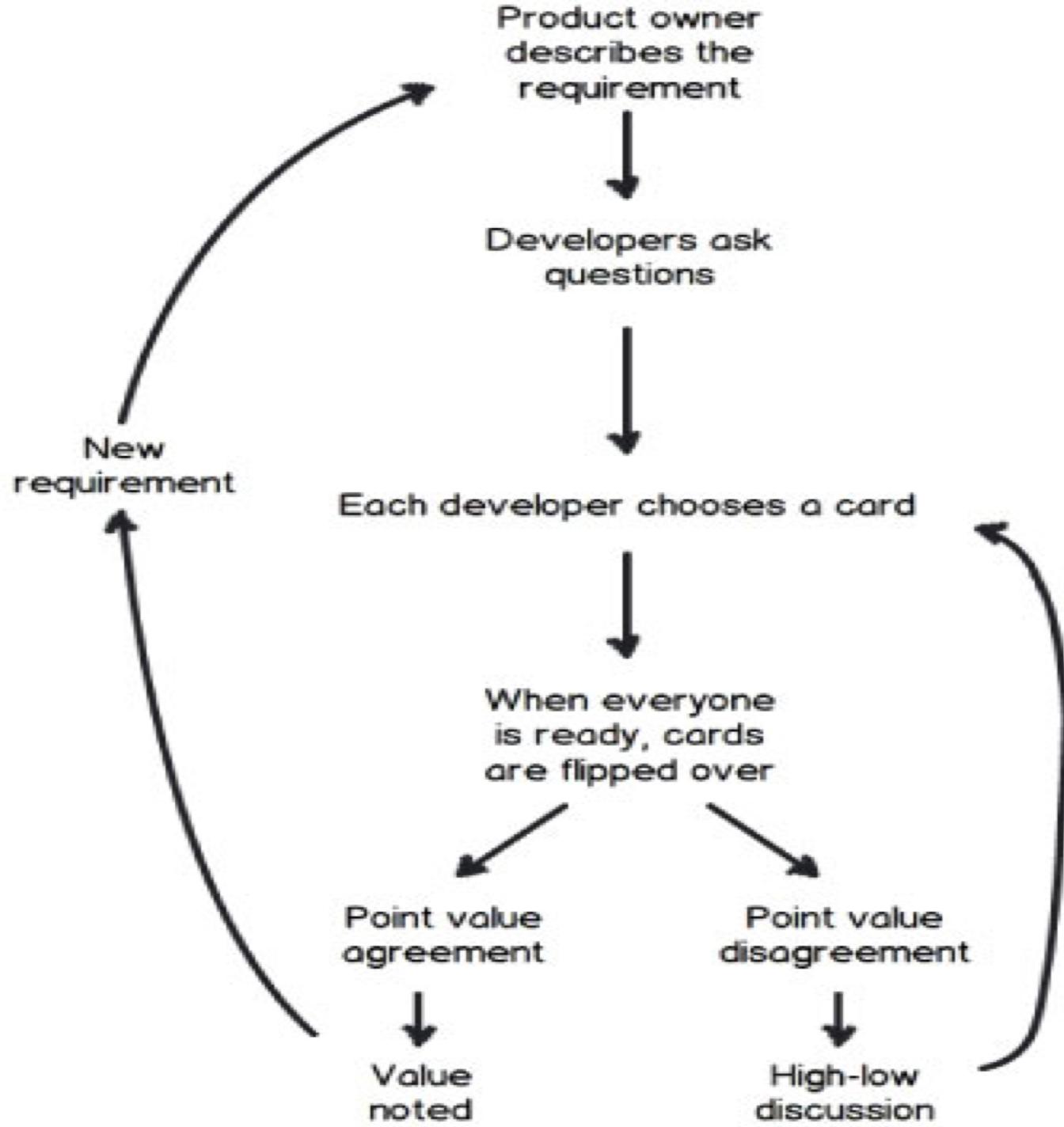
PERT

Delphi

Technika oparta o doświadczenie

Planning poker

Wideband Delphi Estimation Sheet						
Project:	<Project Name>		Estimation Units:	Person Hours		
Estimation Team Member:	<Name>			Date:	<MM-DD-YY>	
Task	Initial Estimate	Change 1	Change 2	Change 3	Change 4	Final
Task1	$n_1$					
Task2	$n_2$					
Task3	$n_3$					
Task4	$n_4$					
Task5	$n_5$					
Task6	$n_6$					
Task7	$n_7$					
Task8	$n_8$					
<b>Net Change</b>						
<b>Total</b>	$\Sigma n_i$					



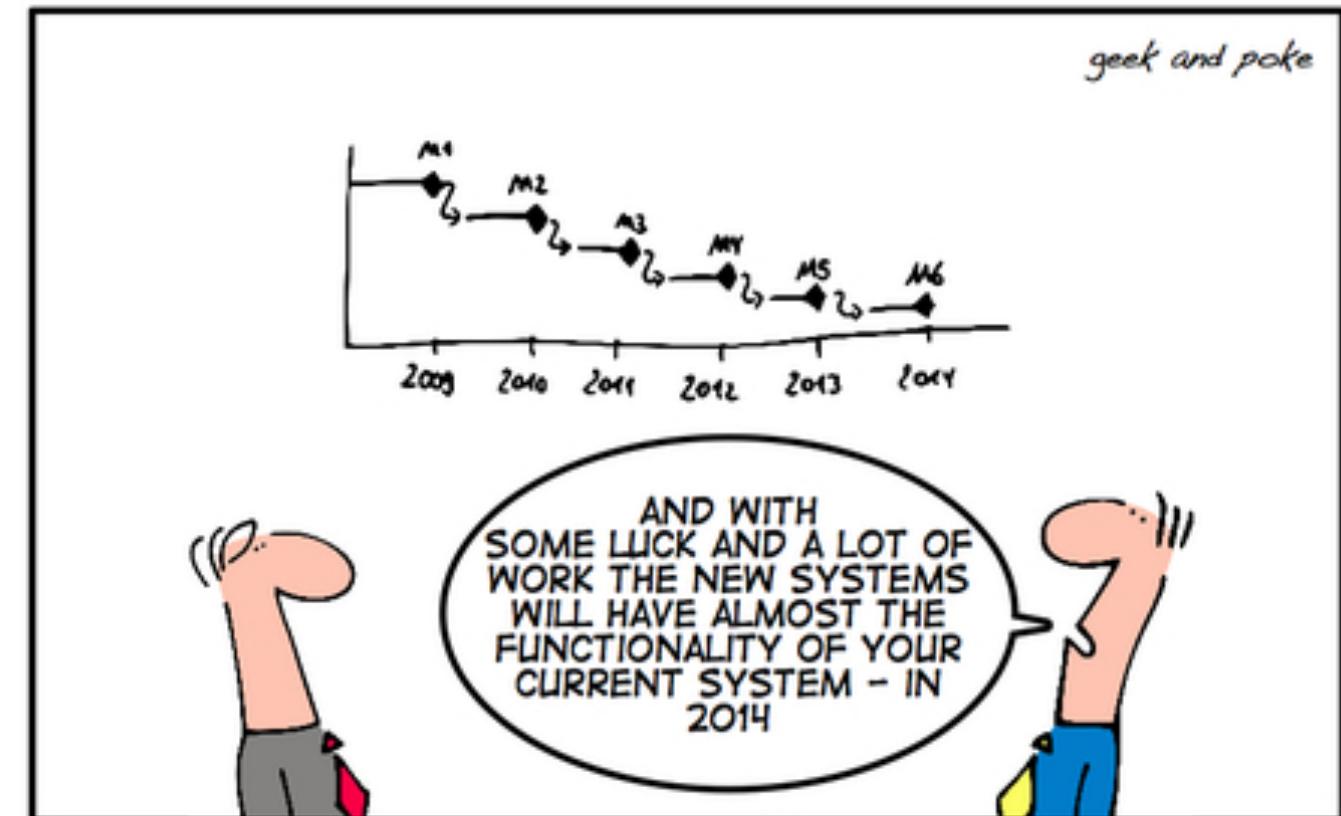
Co to jest Test Plan?

Co powinien zawierać?

Po co to robić?

# Scope

Scenariusze testowe  
Gole do zweryfikowania  
Jakie wymagania pokrywamy  
ROI



ON YEAR IN A IT PROJECT - DAY 22:  
NEVER RAISE EXPECTATIONS

# Out of scope

Czego nie będziemy testować  
Jakich wymagań nie pokrywamy



# Assumptions

Założenia, które muszą być spełnione by testy zostały wykonane poprawnie

## Przykłady założeń:

- Środowiska testowe wg. Specyfikacji muszą być dostępne przed <data>
- Testowany kod jest pokryty w minimum 80% przez testy jednostkowe
- Testerzy przeszli szkolenia domenowe
- Wiele innych uzgodnionych z managementem i biznesem...



# ASSUMPTIONS

NEVER ASSUME WHAT YOU'RE TRYING TO PROVE,  
UNLESS YOU'RE TRYING TO PROVE YOU'RE A BONEHEAD.

# Schedules

Planowane terminy zakończenia poszczególnych faz (Waterfall)

Czas testów

Ile cykli

Czas rozpoczęcia poszczególnych cykli

Ooh I just checked my schedule, looks like I don't have time for your bullshit!!



# Roles and responsibilities

Członkowie zespołu

Kto jest za co odpowiedzialny, może być RACI matrix

Informacje kontaktowe



# Deliverables

Jakie dokumenty (artefakty testowe) będą dostarczone  
Co jest oczekiwane w każdym z dokumentów/raportów



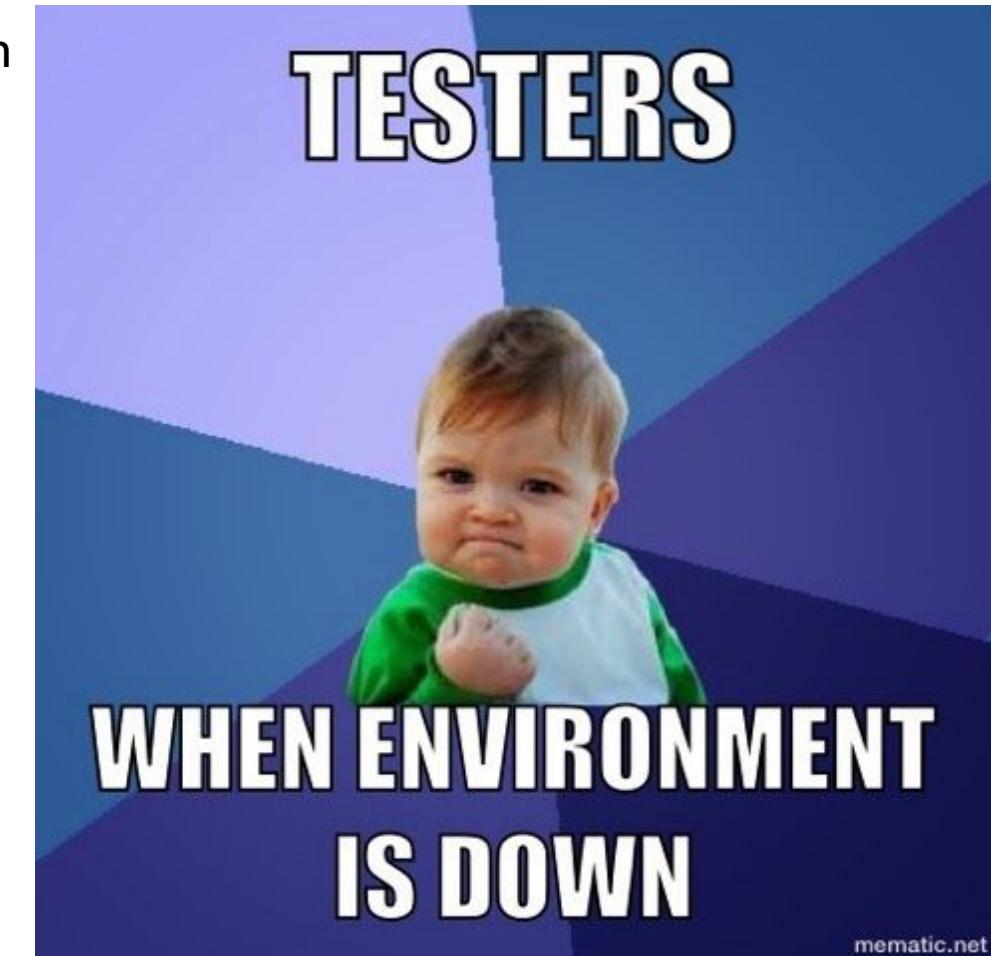
# Environment

Jakie wymagania istnieją odnośnie środowisk testowych

Konfiguracje

Kto jest odpowiedzialny

Co robić w przypadku problemów



# Tools

Jakich narzędzi będziemy używać (np. Jira, TestLink, Selenium itp.)

Prosty opis i instrukcje użytkowania

Konfiguracje

Plusy, minusy, zagrożenia



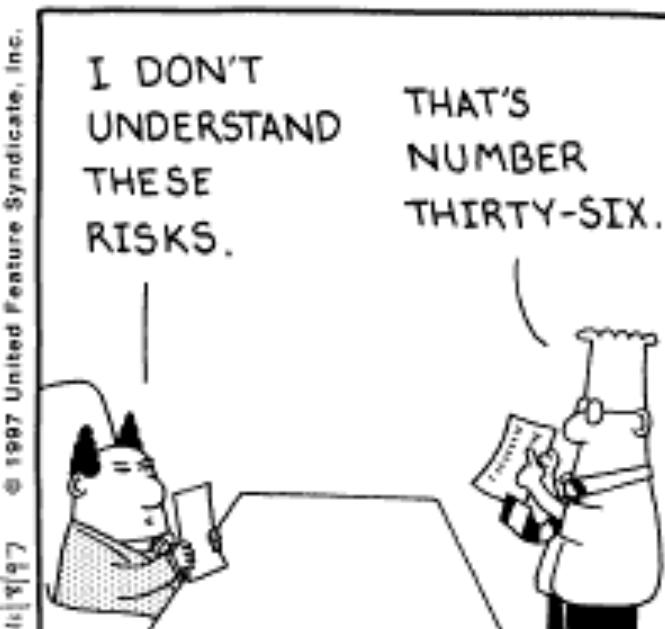
# Defect Management

Do kogo raportujemy błędy  
Jaki mamy flow obsługi błędów  
Jak będziemy raportować  
Czy mamy odpowiednie formatki  
Proces



# Risk Management

Wystawione ryzyka  
Analiza ryzyk  
Plan działania  
Proces



# Exit criteria

Kiedy kończymy testowanie  
Przekazanie wiedzy  
Opis procesu utrzymywania



"I'm your exit strategy."

## Zadanie #3

Każda para tworzy Test Plan dla pierwszej aplikacji jaką testowaliście na zajęciach.

<http://www.phptravels.net/supplier>

Email [supplier@phptravels.com](mailto:supplier@phptravels.com)

Password demosupplier

Przemyślcie co chcecie tam zawrzeć. Co dla Was jest istotne? Jak podzielicie się pracą? Na koniec wymienicie się planem z inną parą i zrobicie review.

Czas: 2h