VI KKIO - Gdańsk

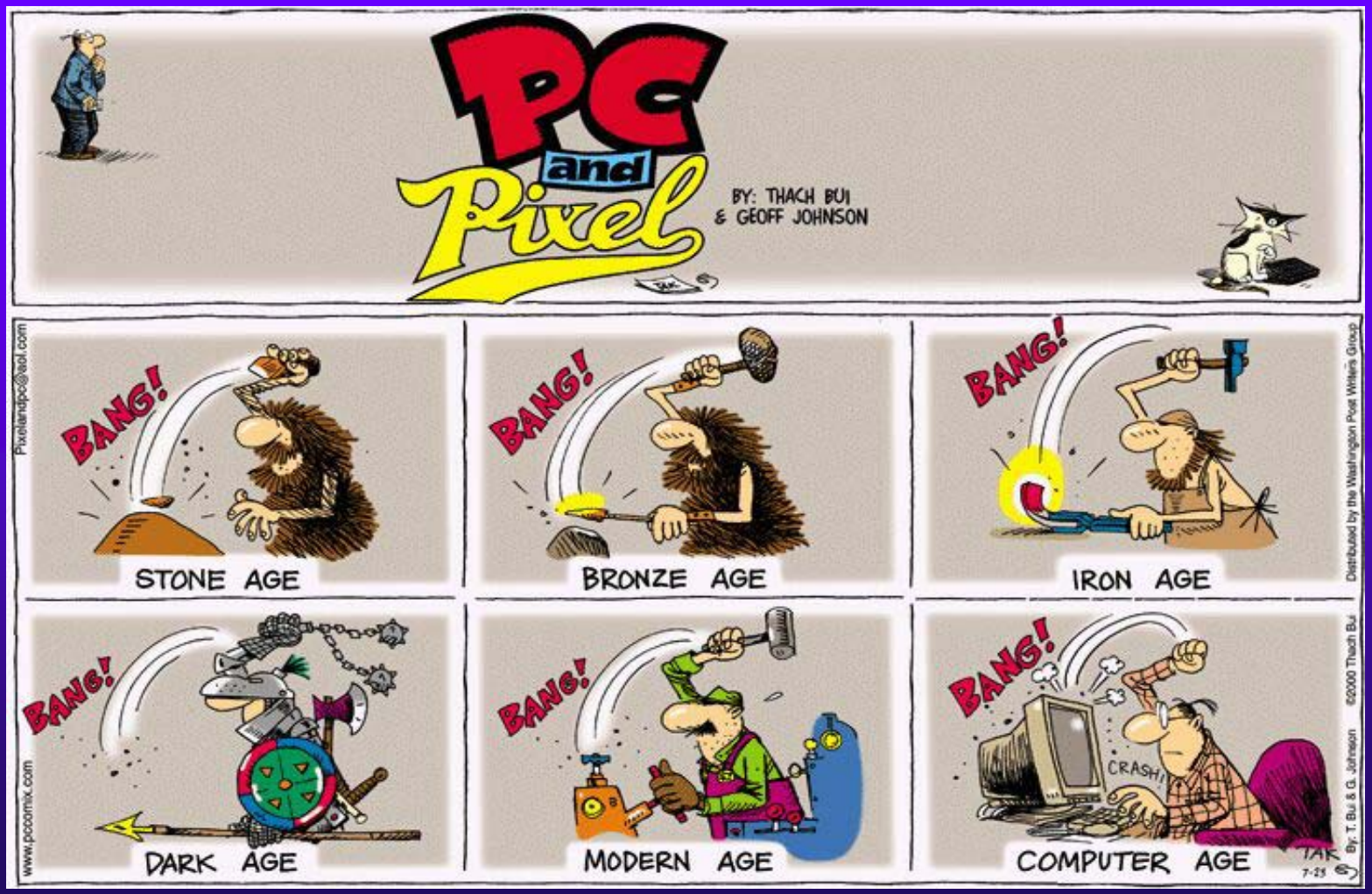
Inżynieria oprogramowania

na rozdrożu

Władysław M. Turski

Instytut Informatyki UW

Informatyka – ostatnim ogniwem rozwoju cywilizacji



- Куда стремиться капитализм?
- К неизбежной гибели!
- А социализм?
- Догнать и перегнать!

W Wielkiej Brytanii wydatki na nowe systemy informatyczne w 2003 r. wyniosły

ok. 22,6 mlrd funtów (~150 mlrd zł).

Raport BCS & RAEng szacuje, że **tylko**

16% - 34% przedsięwzięć jest udane.

To znaczy, że **co najmniej**

15 mlrd funtów poszło na marne.

www.bcs.org/statements/royal

Wartość *całego* rynku informatycznego w Polsce w r. 2003 wyniosła

13,7 mlrd zł

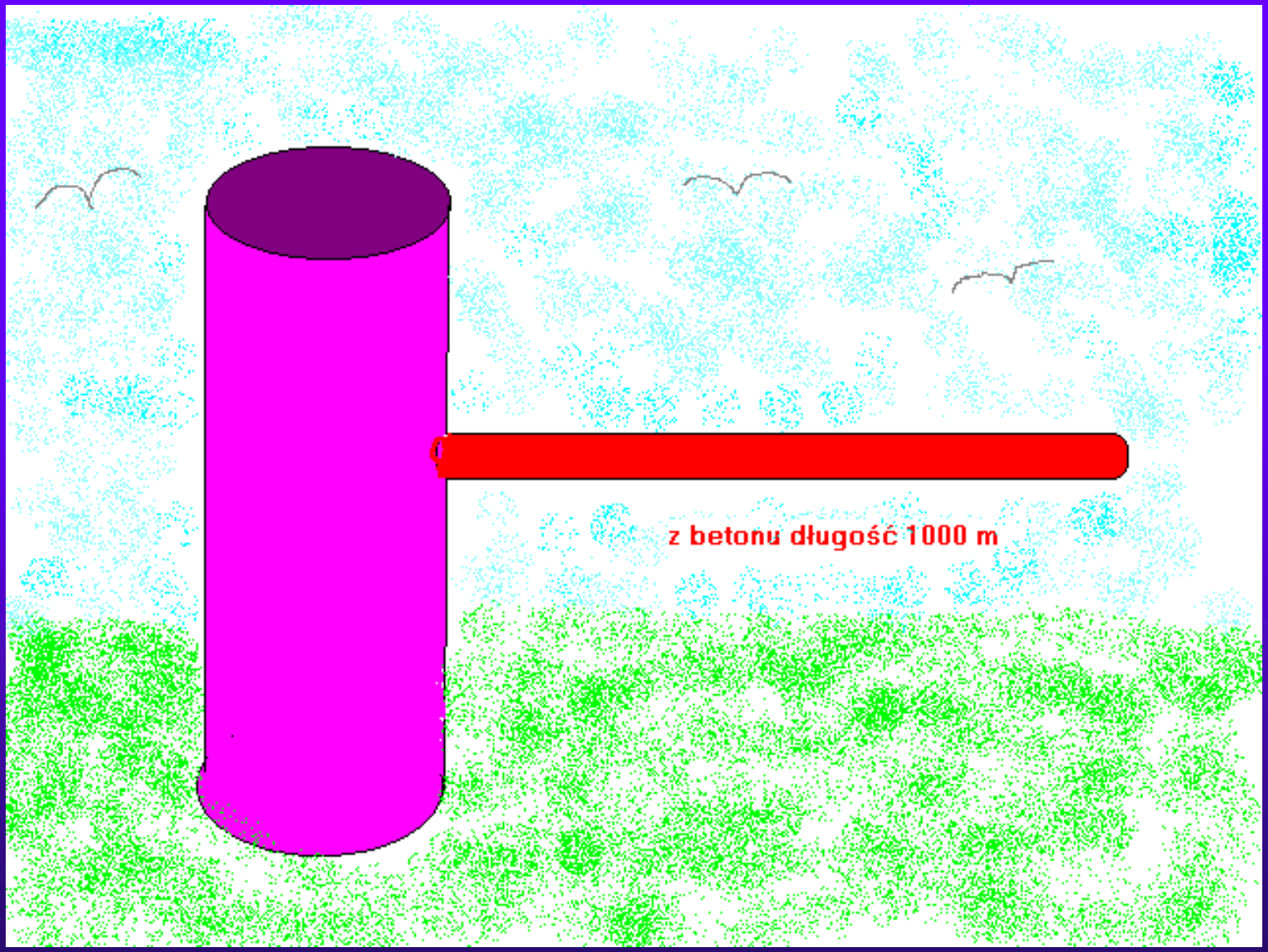
W tym:	sprzęt	47,2%
	usługi	31,4%
	oprogramowanie	21,5%

(wg TELEINFO 500)

Za jedną z głównych przyczyn niepowodzeń raport BCS & RAEeng uznaje

brak profesjonalizmu projektantów i wykonawców oraz

brak standardów konstrukcyjnych (na podobieństwo prawa budowlanego)


z betonu długość 1000 m

Zdecydowana większość niepowodzeń ma swoje źródła w czynnikach ludzkich:

- błędy koncepcyjne
- błędy zarządzania (kierowania)
- zmienność poglądów i uleganie modzie

Pokory!

Czy i jak można temu zapobiec środkami inżynierii oprogramowania?

Ściśle powiedziawszy – nie!

Nadzieja

Ale można przeciwdziałać!

- na błędy koncepcyjne – makietowanie („rapid prototyping”)
- na złe zarządzanie – ścisłe przestrzeganie reguł SEI
- na zmienność zdania – formalizm specyfikowania i „kamieni milowych”

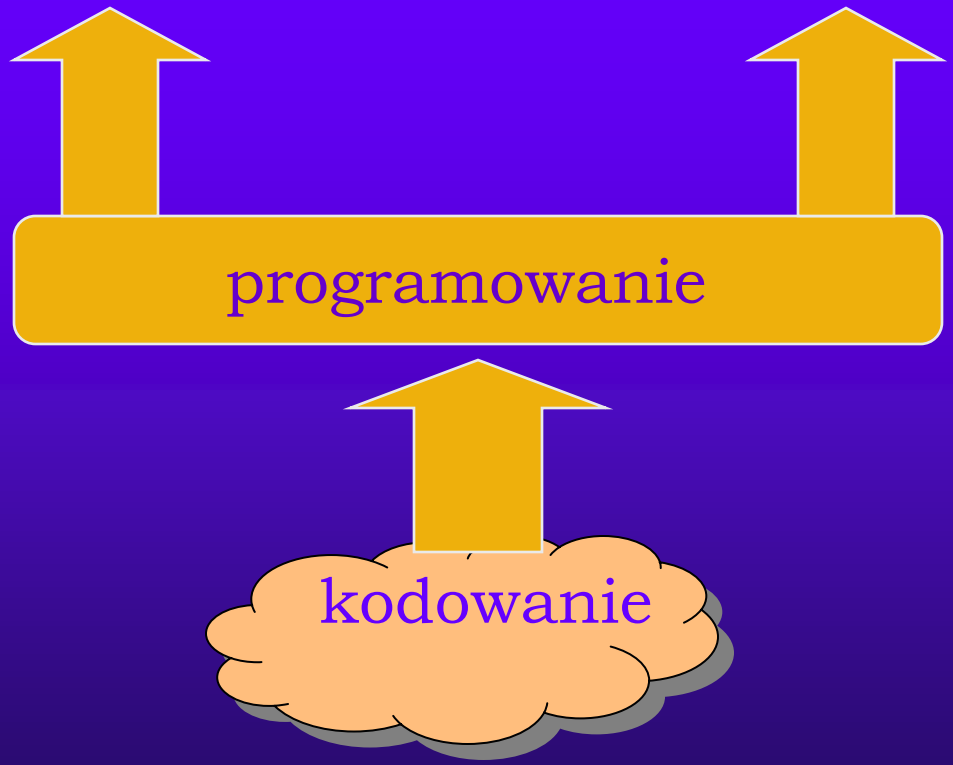
Wychowanie, głupku!

Najważniejsza jednak jest edukacja

- precz z „jakoś to będzie”!
- precz z niechlujstwem!
- precz z akceptowaniem źle udokumentowanych świetnych pomysłów!
- precz z wszelkimi formami indywidualizmu!
- precz z kultem nowości!

Metodologia programowania

Inżynieria oprogramowania



Inżynieria oprogramowania

- Specyfikowanie systemu
- Projektowanie systemu
- (Wykonanie elementów)
- Montaż systemu
- Testowanie systemu
- Konserwacja systemu
- Modernizacja systemu

Metodologia programowania

- Poprawne wykonanie elementów
- Optymalizacja
  - czasowa
  - pamięciowa
- Standaryzacja (!!!)
- Testowanie (?)

Tradycyjny pogląd nie rozgranicza inżynierii oprogramowania (IO) i metodologii programowania (MP).

Czasem IO $\subset$ MP, czasem MP $\subset$ IO, a czasem IO = MP.

Dlaczego takie poglądy nie są już adekwatne?

- coraz rzadziej wykonuje się systemy „od zera”

modernizacja, rozbudowa, adaptacja, integracja, przenosiny na inną platformę (pełne, częściowe), ...

Dlaczego takie poglądy nie są już adekwatne?

- coraz rzadziej wykonuje się systemy „od zera”
- coraz rzadziej wykonanie systemu obejmuje znacząca ilość swobodnego programowania

Dlaczego takie poglądy nie są już adekwatne?

- coraz rzadziej wykonuje się systemy „od zera”
- coraz rzadziej wykonanie systemu obejmuje znacząca ilość swobodnego programowania
- coraz częściej systemy montuje się z gotowych części (COTS)

Dlaczego takie poglądy nie są już adekwatne?

- coraz rzadziej wykonuje się systemy „od zera”
- coraz rzadziej wykonanie systemu obejmuje znacząca ilość swobodnego programowania
- coraz częściej systemy montuje się z gotowych części (COTS)
- coraz częściej wykorzystuje się poprzednie wersje systemu, a nawet inne systemy („legacy code”, integracja)

Takiemu to dobrze!

Inżynier budowy mostów nie zajmuje się

- wytopem i walcowaniem stali
- produkcją lin, nitów, śrub i nakrętek
- produkcją maszyn budowlanych
- itp.

Wszystko to „kupuje” z katalogu, wiedząc

- że nakrętki na pewno pasują do śrub.
- że szyny stalowe na pewno maja wskazaną wytrzymałość itp.
- że maszyny funkcjonują zgodnie z katalogowymi specyfikacjami.
- itp.

IT – usługą masową

Ale bajzel!

Różne „filozofie”

Obliczenie

- poprawne
- skończone
- „optymalne”
- . . .

Usługa

- niezawodna
- dostępna
- przystępna
- . . .

Różne „ filozofie”

Wszystko dla wszystkich	Każdemu, co mu trzeba
Windows	Systemy otwarte
Centralne serwery	Grid computing
…	…
*	*
Coraz więcej sprzętu i coraz bardziej skomplikowane systemy	Coraz większe wymagania względem sieci i jej bezpieczeństwa

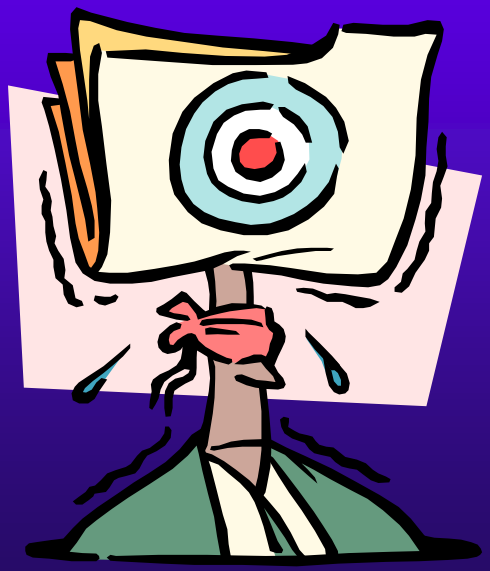
Różne „filozofie”

to różne kierunki rozwoju inżynierii oprogramowania.

Czy stać nas na wszystko?

Czy stać nas na wszystko?

- Jakich „nas”?
- Co to znaczy „wszystko”?
- Co to znaczy „stać na coś”?



Głupie pytania

Co warto, czego nie warto?

Warto

- Standaryzować, przede wszystkim pośrednictwa i ich opisy.
- Rozwijać metody i środki formalnego opisu rzeczywistości.

Nie warto

- Patentować koła.
- Zawracać kijem Wisły.
- Udawać Greka *vel* rżnąć głupa.
- Obiecywać gruszek na wierzbie.
- Szukać kamienia filozoficznego.

Co warto, czego nie warto?

Warto

- Rozwijać metody kalkulacyjnej weryfikacji
  - opisów
  - dekompozycji
  - zestawień

Obliczać konstrukcje

Nie warto

- Patentować koła.
- Zawracać kijem Wisły.
- Udawać Greka *vel* rżnąć głupa.
- Obiecywać gruszek na wierzbie.
- Szukać kamienia filozoficznego.

Co warto, czego nie warto?

Warto

- Wdrażać procedury postępowania
  - kontraktowanie
  - kontrola wewnętrzna i zewnętrzna
  - panowanie nad postępem i wariantami
  - itd.

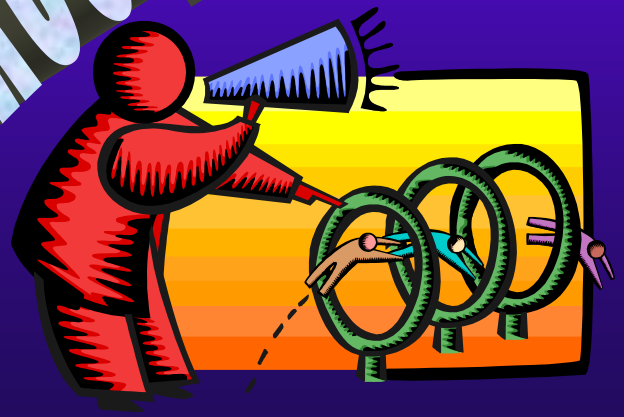
Nie warto

- Patentować koła.
- Zawracać kijem Wisły.
- Udawać Greka *vel* rżnąć głupa.
- Obiecywać gruszek na wierzbie.
- Szukać kamienia filozoficznego.

No łaak

Jakość firmy

to jakość procesu



Wstępująco (bottom-up)

czy

zstępująco (top-down) ?

To pytanie nie ma już sensu !

Inwestor: zbudujcie (przeróbcie zamek na) czterogwiazdkowy hotel na 200 łóżek.

Architekt (konserwator zabytków)

Projektant

Majster Podwykonawcy

Inspektorzy

Robotnicy Robotnicy

Inspektorzy

Dobrze określone role

Zadanie

- stan wiedzy (jak to się robi? dostępne prefabrykaty i narzędzia)
- możliwości i konsekwencje, koszty i ryzyko
  - samodzielnego wykonania
  - użycia prefabrykatów
  - adaptacji

O rany!

Mija czas „Zoś-samoś”, Mädchen für alles i magii.

Idzie czas specjalizacji w wykonywaniu dobrze określonych zadań cząstkowych i scalających.

Prawdopodobnie nie ma najlepszej metody tworzenia oprogramowania.

Na pewno wysiłek poświęcony na jej poszukiwanie jest marnowaniem energii, a próby przekonania innych, że się taką metodę znalazło, świadczą o nikłej wiedzy i nadmiernym samouwielbieniu proponującego.

Lektura prac i – tym bardziej – książek na temat nowych rewelacyjnych metod tworzenia oprogramowania jest stratą czasu.

Kluczem do sukcesu nie jest tworzenie nowych metod lecz systematyczne stosowanie metod właściwych dla wykonywanych zdań.

Oczywiście zakładam, że dysponuje się wiedzą co do tego, które metody są właściwe dla których zadań.

Nareszcie

koniec