

# Analýza a návrh systémů

© 2002 Jiří Sochor  
FI MU Brno

sochor@fi.muni.cz  
<http://www.fi.muni.cz/usr/sochor/>

## Software ?

**Software** je neviditelná, téměř éterická substance, která je obsažena v každém průmyslovém řídícím systému, v obchodních systémech, videohrách, komunikačních sítích, transportních systémech aj.

„**Software** je tvoren celkovým souhrnem počítačových programů, procedur, pravidel a průvodní dokumentace a dat, která náleží k provozu počítačového systému.“

IEEE Glossary of Software Engineering Terminology, 1983

**Softwarový produkt** je výrobek určený k předání uživateli.

## Charakteristiky software

1. Software je vyvíjen a řešen inženýrskými pracemi, není vyráběn v klasickém slova smyslu.
2. Software se fyzicky „neopotřebuje“.
3. Většina software je vyrobena na míru, málo produktů je sestaveno z předem existujících komponent.

## Kategorie SW produktů

### ◆ *Generické produkty:*

Samostatné systémy vytvářené výrobní organizací a prodávané na volném trhu libovolnému ze zákazníků, který si může výrobek kupit.

### ◆ *Smluvní, zakázkové, zákaznické produkty:*

Systémy objednané určitým zákazníkem. Software vyvíjí na zakázku smluvně vázaný dodavatel.

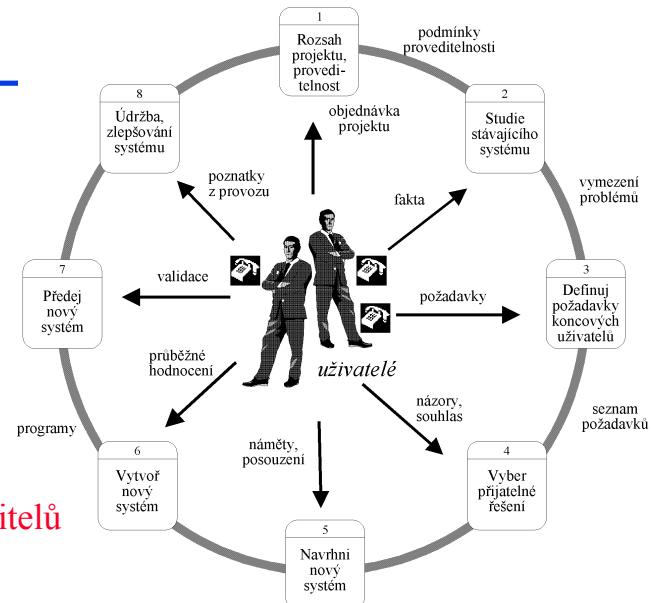
**Specifikace generických produktů vytváří (určuje) obchodní oddělení, jedná se o vnitřní záležitost SW producenta.**

**Specifikace zakázkového produktu tvoří důležitou součást kontraktu mezi zákazníkem a dodavatelem. Změny musí být odsouhlaseny a pečlivě oceněny.**

## Uživatel

5

interakce řešitelů  
a uživatelů

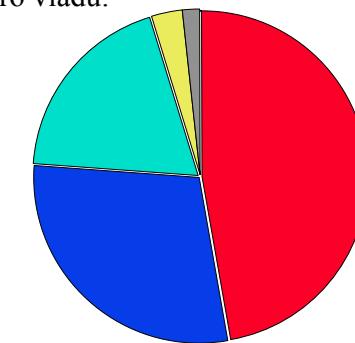


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Co dostane zákazník ?

Zakázky pro vládu:  
\$ 6.8 mil



- \$3.2 mil, dodán, nikdy nebyl úspěšně použit
- \$1.95, zaplacen, nedodán
- \$1.3 mil, po rozsáhlých změnách zahozen
- \$0.198, použit po úpravách
- \$0.119, použit beze změny

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno



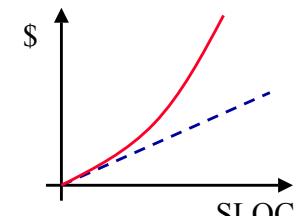
7

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Charakteristiky SW krize

- problémy s údržbou
- vysoká cena programů
- programátorská produktivita  
*extrémní individ. odchylky (1:28)*  
=> *odhad ceny, doby ... ?*
- invariance programovacího jazyka  
*složitost aplikace má větší vliv na produktivitu, než volba programovacího jazyka*
- ceny odstranění chyb



8

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Předmět našeho zájmu

### Softwarové inženýrství

je samostatný inženýrský obor, který řeší systematický přístup k vývoji, provozování, údržbě a nahrazování starého software.

Jako obor s certifikátem v USA uznáno v roce 1997.

*Chybí ucelená teorie, nejsme jednotní v základní terminologii !*



**Praktici používají kolekce technik, které se zdají být funkční.**

## Vývoj softwarového inženýrství

**1976-77:** Požadavky, specifikace, návrh. Výzkum fází před programováním. Návrhové techniky založené na *abstrakci a dekompozici*. Strukturogramy, metakód jako reprezentace návrhu. Snaha o integraci navazujících fází do životního cyklu.

**1978-80:** Rozšíření, asimilace. Zvýšené používání nástrojů pro automatizovaný vývoj software. *Kurzy SE*. Principy z let 69-71 přijaty v softwareovém průmyslu.

**1980-89:** Nástup *CASE a pracovních stanic* pro SE. Nástroje pro jednotlivé fáze SDLC na pracovních stanicích.

**1990-dále:** Aplikace technik expertních systémů v SE. Kombinace SE pracovních stanic, expertních systémů a automatizovaných technik se *uplatňuje v širší míře v SW průmyslu*.

## Vývoj softwarového inženýrství

**před 1969:** Vývoj SW nelze dál řídit, cenové kolapsy a chyby. Téma „Softwarové inženýrství“ na konf.NATO 1968,69

**1969-71:** První principy slušného chování. Výhody návrhu shora-dolů, vývoj po krocích, rozpoznání modularity. Pascal. Organizace týmu, tým šéfprogramátora.

**1972-73:** Strukturované programování, styl programování. Spory o GOTO. Roste *obecné vědomí o životním cyklu*, význam řízení práce, metody řízení.

**1974-75:** Systematické testování, kontrola kvality. Pojem formální korektnosti programu, modely chyb a spolehlivosti systému. První *systematické srovnání pracnosti a nákladů*.

## Základní atributy dobré řešeného SW

◆ **Udržovatelnost:** Měl by být umožněn vývoj SW podle měnících se potřeb zákazníka.

◆ **Spolehlivost:** Mezi atributy SW, na který je spolehnutí, patří spolehlivost, ochrana, a bezpečnost. SW by neměl při výpadku systému způsobit ani fyzické, ani ekonomické škody.

◆ **Efektivita:** SW by neměl plýtvat prostředky systému (paměť, čas procesoru a pod.)

◆ **Použitelnost:** SW by měl mít přiměřené uživatelské rozhraní a odpovídající dokumentaci.



PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Kritické faktory SW produktivity

- **složitost**

*lze charakterizovat nepřímo, některými viditelnými atributy programu (architektura, počet proměnných ...)*

- **velikost**

*programování v malém vs. programování ve velkém*

- **komunikace**

*jednotlivec, malý tým, velký tým*

- **čas, plán prací**

*6 měsíců +, 5 měsíců - !*

- **neviditelnost SW**

© J.Sochor, FI MU Brno

## Programování „v malém /ve velkém“

### Programování v malém

Ověřené techniky:

- shora-dolů, strukturované kódování, postupné zjemňování, zdokonalování (step-wise refinement)
- inspekce logiky a kódu
- nástroje (překladače, odvšivovače)

### Programování ve velkém

- plánovací mechanismy (dělení práce, harmonogram, zdroje)
- dokumentovaná specifikace
- strukturovaný tým
- formalizované soubory testů, testovací příklady

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Vývoj software jako proces ...

**Vývoj software je proces**, při němž jsou  
**uživatelovy potřeby**

transformovány na  
**požadavky na software**,  
tyto jsou transformovány na  
**návrh**,

návrh je implementován pomocí  
**kódu**,  
kód je testován, dokumentován a certifikován pro  
**operační použití**.

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Základní aktivity SW procesu

- ◆ **Specifikace SW**: Je třeba definovat funkcionality SW a operační omezení.
- ◆ **Vývoj SW**: Je třeba vytvořit SW, který splňuje požadavky kladené ve specifikaci.
- ◆ **Validace SW**: SW musí být validován („kolaudován“), aby bylo potvrzeno, že řeší právě to, co požaduje uživatel.
- ◆ **Evoluce SW**: SW musí být dále rozvíjen, aby vyhověl měnícím se požadavkům zákazníka.

## Viditelné znaky „výroby SW“

### *artefakty*

- výpis programů
- dokumentace
- data
- zdrojové soubory

### *procesy*

- pracovní postupy
- uznávaná pravidla (rules-of-thumb)
- interakce mezi členy týmu

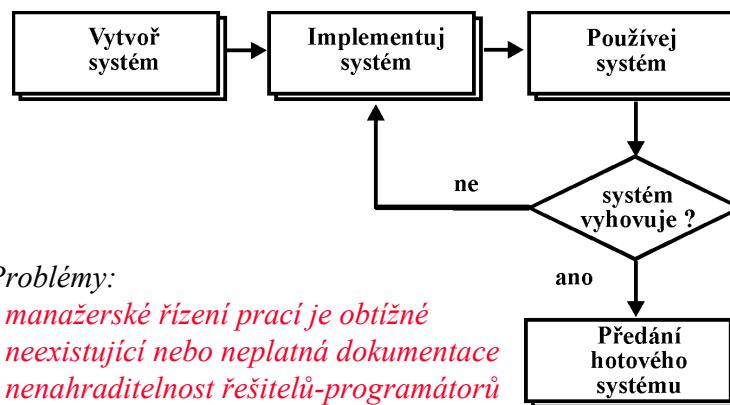
## Charakteristiky výrobního procesu SW - I

- ◆ **Srozumitelnost**: Je proces explicitně definován a je snadné porozumět definici procesu ?
- ◆ **Viditelnost**: Vyústují procesní aktivity ve zřetelné výsledky tak, že postup procesu je viditelný zvenčí ?
- ◆ **Spolehlivost**: Je proces navržen tak, že se lze chybám procesu vyhnout, nebo je zachytit dříve, než zaviní chyby ve výrobku ?
- ◆ **Přijatelnost**: Je definovaný proces přijatelný a použitelný inženýry zodpovídajícími za produkci ?

## Charakteristiky výrobního procesu SW - II

- ◆ **Robustnost**: Může proces pokračovat i v po výskytu neočekávaných problémů ?
- ◆ **Udržovatelnost**: Může se proces vyvíjet tak, aby odrážel měnící se organizační požadavky nebo identifikovaná zlepšení procesu ?
- ◆ **Rychlosť**: Jak rychle lze realizovat výrobní proces, který z dané specifikace vytvoří hotový systém předaný zákazníkovi ?
- ◆ **Podporovatelnost**: Do jaké míry lze aktivity procesu podpořit nástroji CASE ?

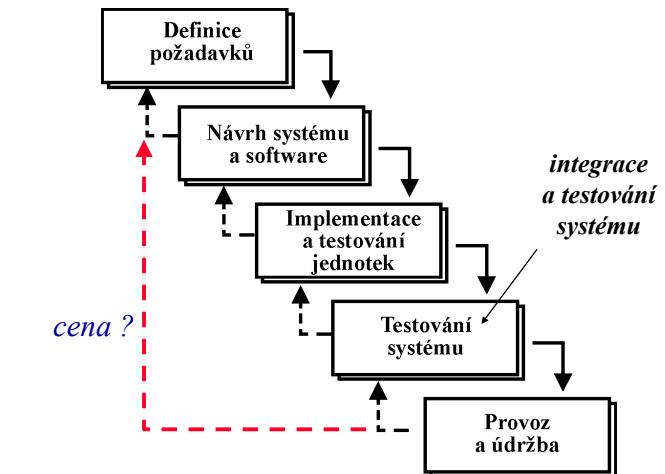
## Model životního cyklu - „výzkumník“



PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Model životního cyklu - „vodopád“



PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Model životního cyklu - „vodopád“

### Problémy:

1. Reálné projekty nedodržují jednotlivé kroky v předepsaném pořadí.
2. Uživatel nedokáže v počátečních etapách formulovat požadavky na systém zřetelně a přesně.
3. Zákazník musí být trpělivý. Pozdní odhalení nedostatků může vážně ohrozit celý projekt.

Klad nebo zápor ? Manažeři tento model preferují !

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Viditelnost ž.cyklu „vodopád“

Aktivita	Výstupní dokumenty
Analýza požadavků	Studie proveditelnosti Obrysové požadavky
Definice požadavků	Dokument požadavků Funkční specifikace
Specifikace systému	Plán testování Návrh uživatelského manuálu
Návrh architektury	Specifikace architektury Plán testování systému
Návrh rozhraní	Specifikace rozhraní Plán integračních testů

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Viditelnost ž.cyklu „vodopád“

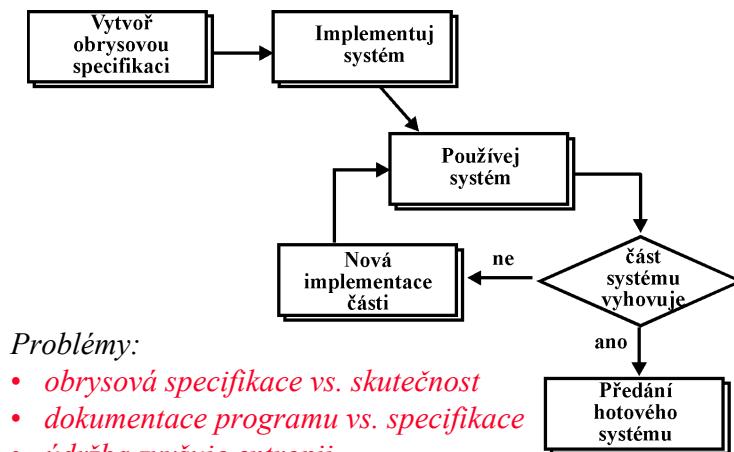
Aktivita	Výstupní dokumenty
Podrobný návrh	Specifikace návrhu
Kódování	Plán testování jednotek
Testování jednotek	Kód programu
Testování modulů	Protokol o testování jednotek
Integrační testování	Protokol o testování modulů
Testování systému	Protokol integračních testů
Přejímací testování	Konečný uživatelský manuál
	Protokol testování systému
	Konečný systém a dokumentace

## „Vývoj podle obrysové specifikace“

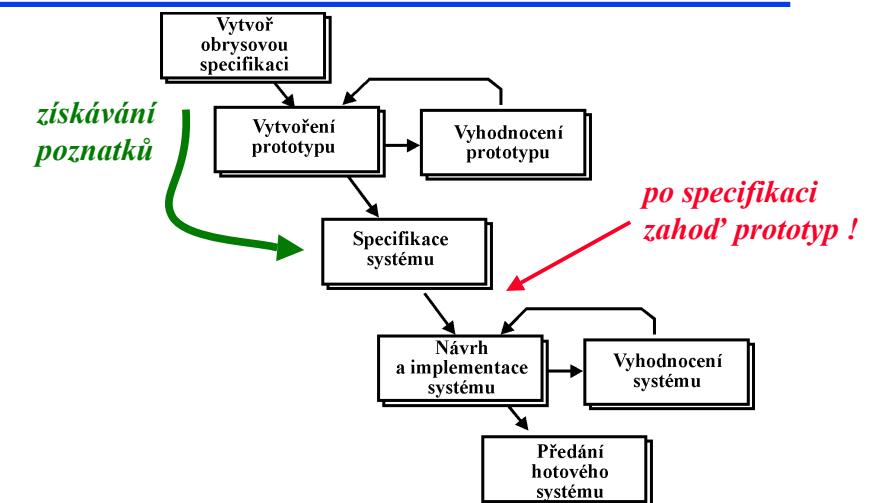
### Manažerský pohled:

- 1) *Proces není viditelný.* Manažer potřebuje pravidelné výstupy, aby mohl řídit proces. Při rychlém vývoji není efektivní produkovat dokumenty, které přesně zachycují každou verzi.
- 2) *Systémy jsou obvykle špatně strukturované.* Neustálé změny poškozují strukturu systému. Evoluce je obtížná a nákladná.
- 3) *Jsou vyžadováni vysoko talentovaní pracovníci.* Průměrný tým nelze použít pro tento typ vývoje. Úspěšné produkty byly takto vytvořeny malými týmy vysoko talentovaných řešitelů.

## „Vývoj podle obrysové specifikace“

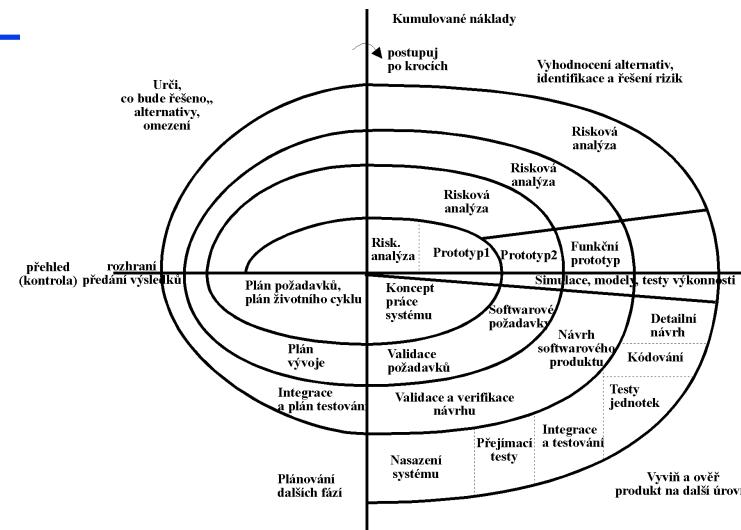


## Životní cyklus „prototypování“



## Spirálový model život. cyklu - Boehm, 1988

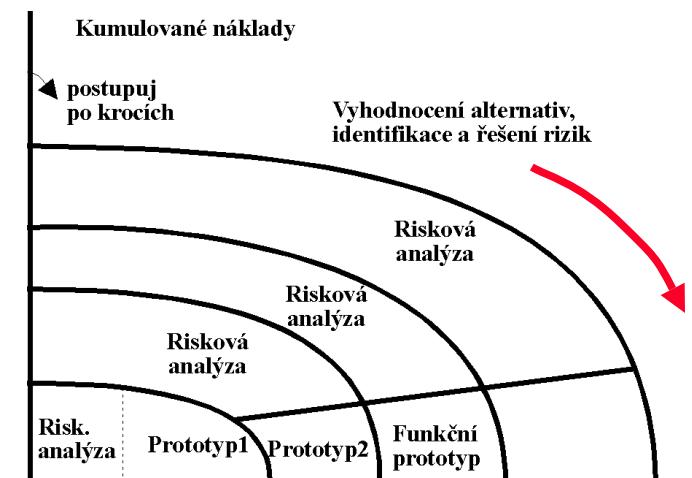
29



PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Spirálový model - I.

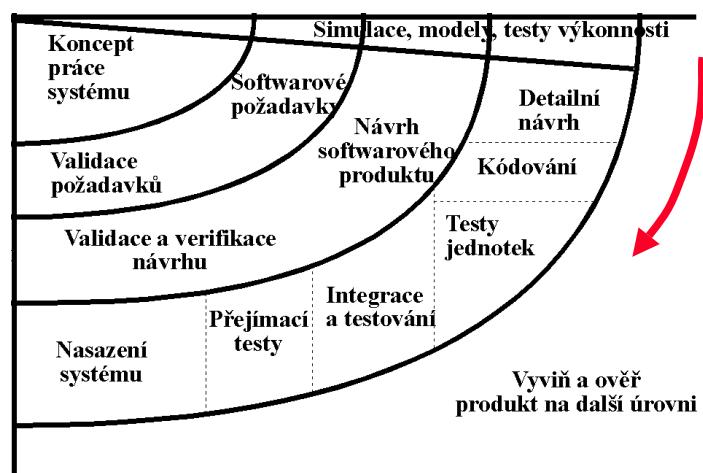


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Spirálový model - II.

31

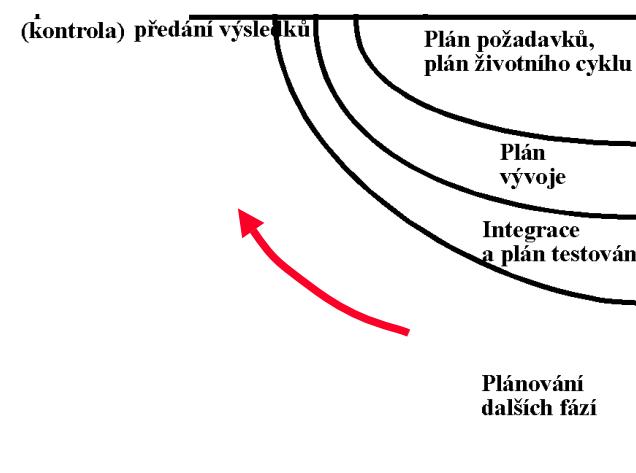


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Spirálový model - III.

32

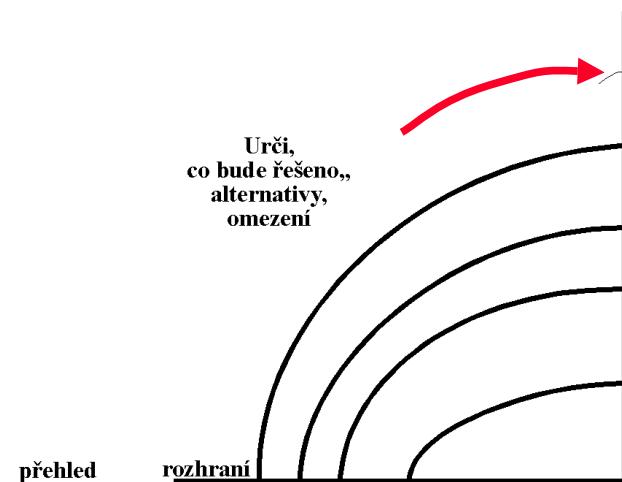


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Spirálový model - IV.

33

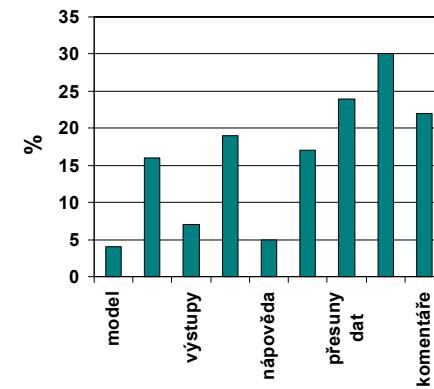


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Složení aplikace

- ◆ výpočty na modelu
- ◆ uživatelské vstupy
- ◆ uživatelské výstupy
- ◆ řízení
- ◆ návodě
- ◆ zpracování chyb
- ◆ přemístění dat (uvnitř)
- ◆ deklarace dat
- ◆ komentáře



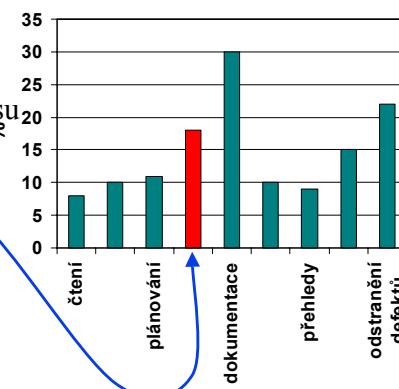
PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Typické aktivity programátorů

35

- ◆ čtení poznatků
- ◆ navrhování aplikace, komponent, dokumentace
- ◆ plánování přístupu, úloh, času
- ◆ programování
- ◆ tvorba dokumentace
- ◆ testování
- ◆ přehledy
- ◆ setkání
- ◆ odstranění defektů

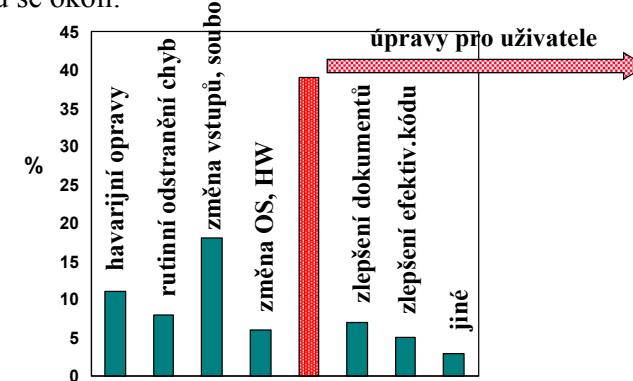


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Údržba

Údržba je modifikace SW produktu po předání zákazníkovi za účelem opravy chyb, zvýšení výkonnosti a přizpůsobení měnícímu se okolí.

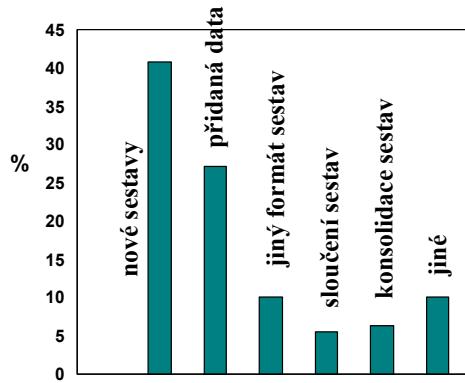


PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

36

## Úpravy pro uživatele



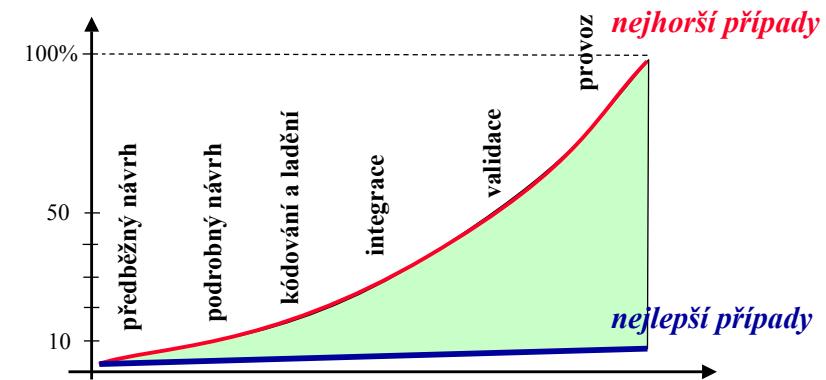
Víra: Investice do dražších nástrojů (generátory aplikací, 4GL,...) sníží náklady údržby.

## Proč máme utrácet za specifikace ?

### Hypotézy:

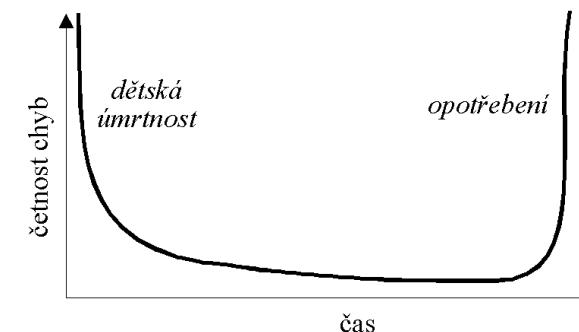
1. Čím později v životním cyklu detekujeme chybu, tím nákladnější bude její oprava.
2. Mnoho chyb zůstane skryto a je odhaleno až po ukončení fáze, v níž byly udělány.
3. V požadavcích je mnoho chyb.
4. Chyby v požadavcích jsou především **chybná fakta, opomenutí, rozpory a nejednoznačnosti**.
5. Chyby v požadavcích lze detektovat.

## Relativní cena údržby (změn)



## Opotřebení HW produktů

### HW



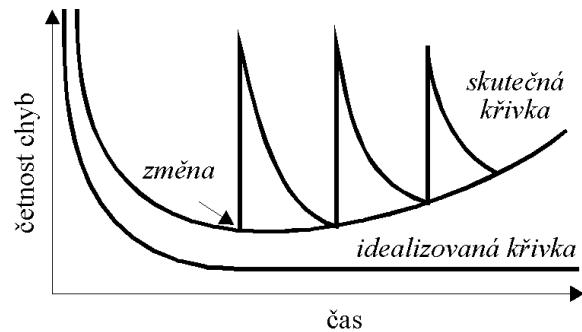
dlouholetá praxe !



PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

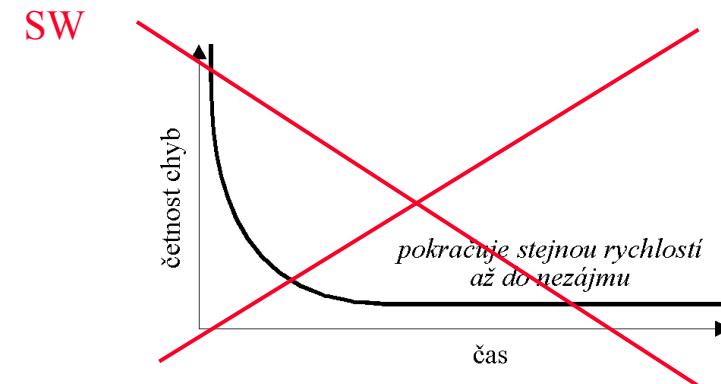
### Chyby a stárnoucí SW



PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

### Opotřebení SW produktů



Ideál a nesplnitelný cíl !

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

### Nová profese - „údržbář programů“

1970 - 50%  
1980 - 70%  
1990 - ?

Složitost údržby ovlivňuje

- špatná dokumentace
- vylepšování (?) na žádost zákazníka
- časové limity (údržba a změny se dělají „rychle“)

PB007: Analýza a návrh systémů, 23.9. 2003

© J.Sochor, FI MU Brno

## Lehmanovy „zákony“ (1980,1985)

1. **Z. trvalé proměny:** Systém používaný v reálném prostředí se neustále mění, dokud není levnější systém restrukturalizovat, nebo nahradit zcela novou verzí.
2. **Z. rostoucí složitosti:** Při evolučních změnách je program stále méně strukturovaný a vzrůstá jeho vnitřní složitost. Odstranění narušující složitosti vyžaduje dodatečné úsilí.
3. **Z. vývoje programu:** Rychlosť změn globálních atributů systému se může jevit v omezeném časovém intervalu jako náhodná. V dlouhodobém pohledu se však jedná o seberegulující proces, který lze statisticky sledovat a předvídat.

## Lehmanovy „zákony“ (1980,1985)

4. **Z. invariantní spotřeby práce:** Celkový pokrok při vývoji projektů je statisticky invariantní. Jinak řečeno, rychlosť vývoje programu je přibližně konstantní a nekoreluje s vynaloženými prostředky.
5. **Z. omezené velikosti příručku:** Systém určuje přípustnou velikost příručku v nových verzích. Pokud je limita překročena, objeví se závažné problémy týkající se kvality a použitelnosti systému.

## Programování v týmu

LOC - velikost programu ve zdrojových řádcích

E - pracovní úsilí (doba v měsících)      *Lines of Code*

PP - produktivita programátora      *Effort*

PP=LOC/E (řádky kódu za měsíc)      *Programmer's productivity*

GPP = LOC/(E+λN<sup>2</sup>)      *Group Prog.Productivity*

N programátorů => N.(N-1)/2 interakcí ~ N<sup>2</sup>

GPP = LOC/(E+λN<sup>2</sup>)      skupinová produktivita

GPP/PP = E /(E+λN<sup>2</sup>)      *úsilí na komunikaci*

**Brooksův zákon:** *Přidání řešitelské kapacity u opožděného projektu zvětší jeho zpoždění.*