

# Úvod do problematiky

Karel Richta

Katedra technických studií  
Vysoká škola polytechnická Jihlava

© Karel Richta, 2020

Softwarové inženýrství, SWI  
02/2020, Lekce 1

<https://moodle.vspj.cz/course/view.php?id=200214>



# Úvodem trochu historie

- Termín „software“ zavedl v roce 1958 statistik John Tukey (také autor termínu „bit“).
- Počítačů přibývalo, přibývalo i softwarových projektů, ale ubývalo úspěšně dokončených projektů. Někdy to došlo až na hranici únosnosti – software byl, nebo mohl být, příčinou havárií.
- Za okamžik zrození termínu „softwarové inženýrství“ se obvykle považuje rok 1968, kdy NATO sponzoruje první konferenci s tímto názvem a na toto téma.
- V roce 1972 vychází první časopis „Transactions on Software Engineering“ (IEEE Computer Society).
- V roce 1976 vytváří IEEE Computer Society první komisi, která by měla definovat obsah oboru „softwarové inženýrství“.

# Případ sondy Mariner I. (1962)

- Mariner I. byla sonda, která měla za cíl Venuši. Musela být zničena 293 sekund po startu.
- Příčinou byla hardwarová chyba v anténě, ta ale způsobila, že ovládání přešlo z počítače na zemi na lokální počítač rakety.
- A tam byla v softwaru chyba. Vznikla ručním přepisem vzorce, ve kterém programátor přehlédl aplikaci funkce (znázorněné nadepsanou čarou -  $\bar{o}$ ), což způsobilo chybu ve výpočtu, která způsobila odklon rakety z požadované dráhy.



## Případ Mercury (1962)

- Záměna čárky za tečku ve FORTRANu:
  - Místo "DO 17 I = 1,10", což je cykl,
  - bylo "DO17I = 1.10",
  - to je ale přiřazení hodnoty 1.10 proměnné DO17I
  - Pozn.: Mezery nejsou ve FORTRANu důležité.
- Naštěstí se ale chyba odhalila před startem dříve, než se mohla projevit. Iniciovala testování podle struktury programu, neboť výše uvedený cyklus nebyl v testování nikdy vyzkoušen.

## Případ Apollo 11 (1969)

- První přistání na Měsíci se v roce 1969 nezdařilo přesně podle představ.
- Přistávací modul Eagle se o 4 vteřiny odchýlil od plánované trajektorie.
- Mezi velké kameny poblíž kráteru dosednul pod manuálním řízením Neila Armstronga (míle daleko od zvoleného místa).
- Způsobil to zapnutý radar, který spotřeboval neplánovaný čas procesoru, místo aby se věnoval řízení.

### 3. světová válka ... téměř (září 1983)

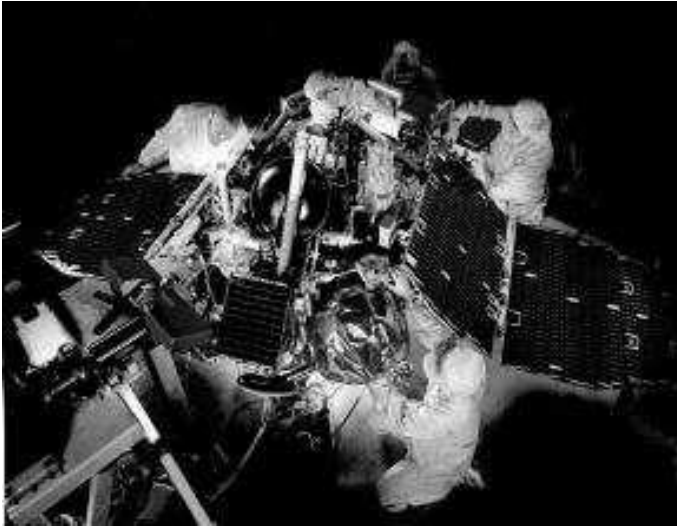
- Katastrofa: Chyba v sovětském softwaru nedokázala odfiltrovat detekci falešných střel způsobenou odrazy slunečního záření od mraků.
- Systém sovětského včasného varování falešně ukázal, že Spojené státy zahájily útok pomocí pěti balistických raket. Naštěstí úřadující sovětský důstojník (Stanislav Petrov) měl „podivné lechtání v žaludku“, že kdyby USA skutečně útočily, že by poslaly více než pět střel, takže ohlásil zjevný útok jako falešný poplach.

# Úplně se to vždy nepovedlo (1996)

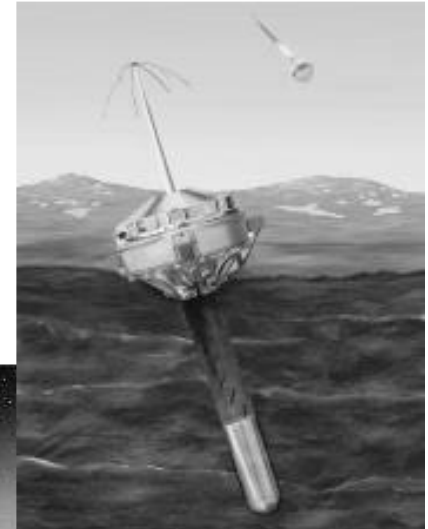
- Pád rakety Ariane 5 (1996):
  - Neobsloužená výjimka při operaci v pohyblivé řádové čárce
  - Ztráta 100 mil. \$, včetně družic Cluster 500 mil. \$.
  - Návrat programu Ariane o několik let zpět.
- Díky chybě dal řídicí počítač rakety příkaz k současnému vychýlení trysek urychlovacích bloků, tak i trysky motoru. Tím se kurs rakety prudce změnil a v důsledku aerodynamických sil se horní část rakety odlomila. Byl aktivován vlastní autodestrukční systém rakety a raketa se změnila v oblak hořících úlomků



# Přistávací modul na Marsu (1999)



Celková ztráta  
Mars Climate  
Orbiter  
125 mil. \$

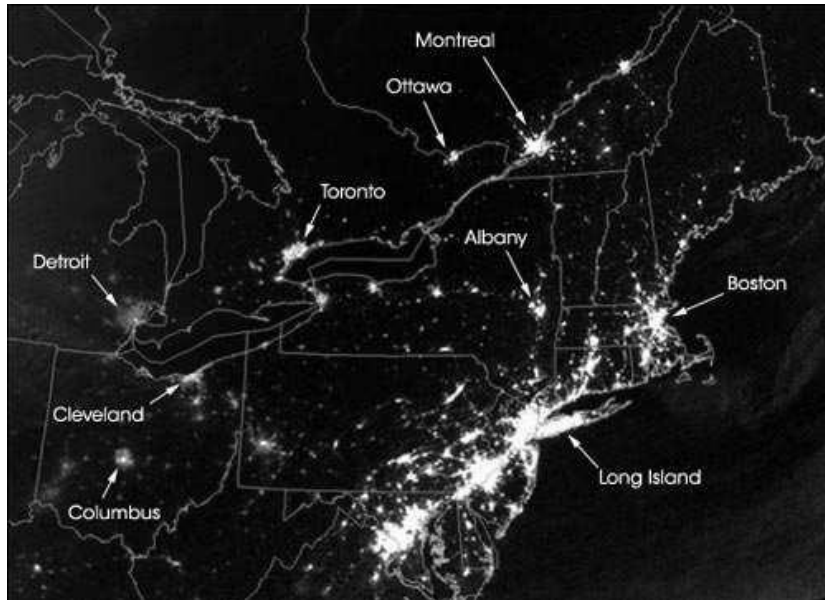


Problém komunikace mezi komponentami – uživatel rozhraní očekával hodnotu v kilometrech, poskytovatel ji udával v mílech. Ve výšce 57 km se nepodařilo modul ukotvit na oběžné dráze (měl být ve výšce 145 km).



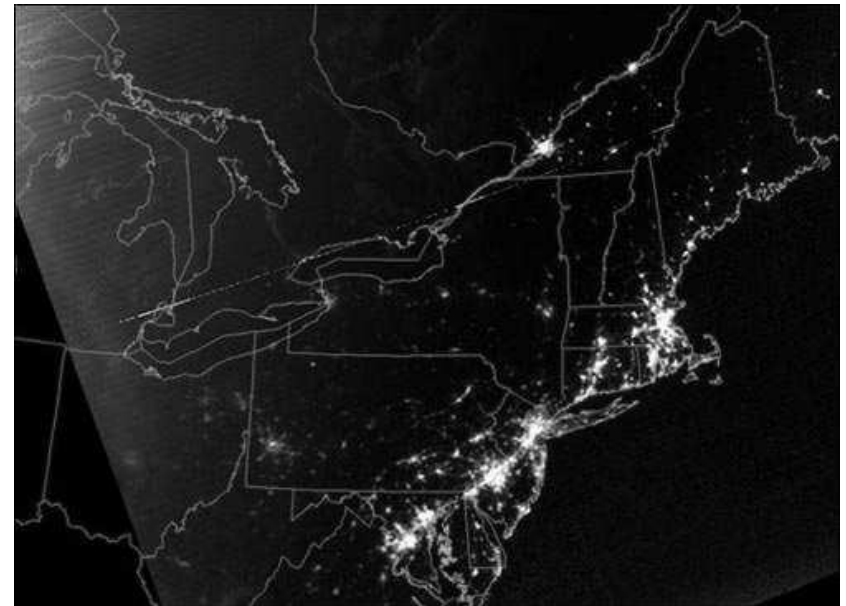


# Výpadek elektřiny USA (2003)



Postihlo to až 50 mil. obyvatel,  
zemřeli 3 lidé,  
škoda 6 mld. \$.

Způsoben neregistrovaným  
výpadkem automatizovaného  
hlásiče poruch v elektrárně  
u Niagary, který se kaskádně  
rozšířil sítí.



## Semaforey v Praze (2008)

- Závada počítačového programu zavinila v metropoli dlouhé kolony. Kvůli tomu se ucpala Severojižní magistrála nebo například ulice Ječná, Žitná, Vinohradská.
- Semaforey se vždy mezi šestou a sedmou hodinou ranní, podle toho, jak jsou nastaveny, přepnou z nočního do denního režimu. Došlo ale k výpadku softwaru, který řídí některé křižovatky a semaforey na nich se nepřepnuly.

# Výpadek služby Amazon S3 (2008)

- Amazon S3 (Simple Storage Service) - ukládání datových objektů na servery Amazonu s HTTP, SOAP a REST rozhraním.
- Ve 4:31 ráno 15.2.2008 překročilo množství autentizovaných požadavků kapacitu autentizačních serverů.
- Do 6:48 byla služba zcela nedostupná.
- Amazon plánuje lepší monitoring a stavové informace služby dostupné pro zákaznické aplikace spoléhající na 24/7 dostupnost.



# Centrální registr vozidel ČR (2012)

- V pondělí 9. července 2012 byla ministerstvem dopravy spuštěna nová aplikace Centrálního registru vozidel, ale během několika minut však došlo k výpadku a od té doby se dlouho potýkala s problémy.
- Ministerstvo dopravy stál software 37 milionů korun. 10 milionů korun by měly být náklady za nákup výpočetní techniky pro cca 1440 koncových uživatelů.

Výpadek Centrálního registru vozidel

Aktualizováno 11.01.2013

Z důvodu výpadku Centrálního registru vozidel nebude možné během dnešního dne provádět úkony s tímto registrem spojené. Pracovníci IT na tomto výpadku intenzívně pracují, avšak není možné určit, kdy se jim podaří závadu odstranit.

# Termín softwarové inženýrství

- *„Softwarové inženýrství je disciplína, která se zabývá zavedením a používáním řádných inženýrských principů do tvorby software tak, abychom dosáhli ekonomické tvorby software, který je spolehlivý a pracuje účinně na dostupných výpočetních prostředcích.“*

*Konference „Softwarové inženýrství 1968“*

# Termín softwarové inženýrství

- *„Softwarové inženýrství je disciplína, která se zabývá zavedením a používáním řádných inženýrských principů do tvorby software tak, abychom dosáhli ekonomické tvorby software, který je spolehlivý a pracuje účinně na dostupných výpočetních prostředcích.“*

*Konference „Softwarové inženýrství 1968“*

# Termín softwarové inženýrství

- *„Softwarové inženýrství je disciplína, která se zabývá zavedením a používáním řádných inženýrských principů do tvorby software tak, abychom dosáhli ekonomické tvorby software, který je spolehlivý a pracuje účinně na dostupných výpočetních prostředcích.“*

*Konference „Softwarové inženýrství 1968“*

## Další historie – SWI jako profese

- V roce 1993 vznikají komise IEEE a ACM, které ústí do společného úsilí definovat softwarové inženýrství jako disciplinu. V roce 1998 společná komise IEEE a ACM definuje profesi softwarového inženýra.
- Nakonec v roce 2004 vzniká společný návrh „curricula“ pro výuku tohoto oboru, označovaného [SE2004](#).
- Tím se završilo uznání softwarového inženýrství jako discipliny, podobně jako CS1991 završilo uznání informatiky.
- Softwarové inženýrství je definované jako standard [IEEE 610.12](#).

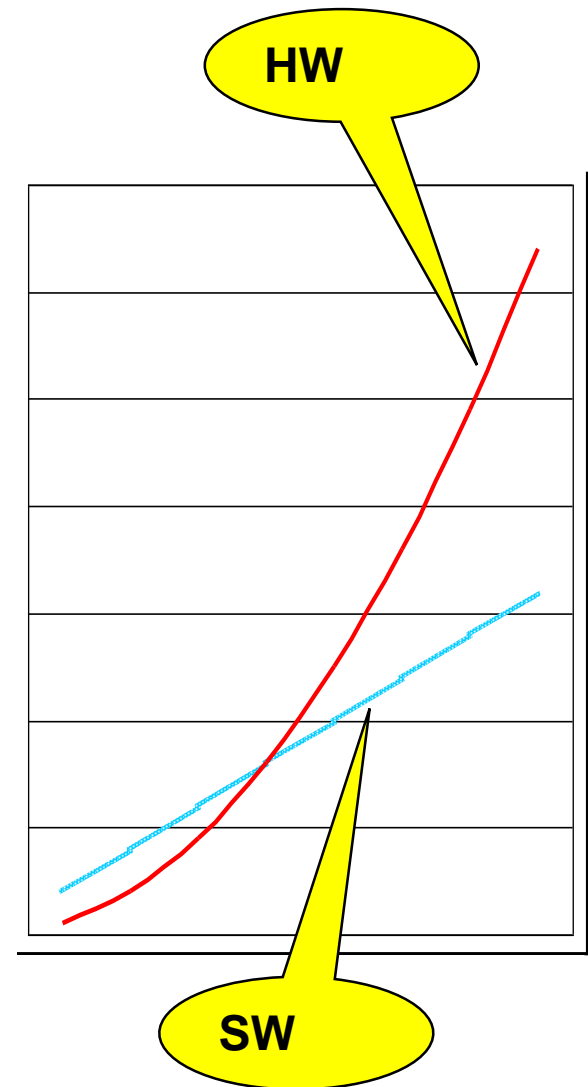


# Příčina vzniku SWI?

- Obvykle se říká, že to způsobila „**softwarová krize**“.
- Dokud výkon počítačů nepřesáhl určitý rozměr, bylo možno se spolehnout na programátorské „hvězdy“.
- Často se počítače se využívaly pro vědecko-technické výpočty, kde záleželo spíše na preciznosti řešení, než na efektivitě tvorby programů.
- Edsger W. Dijkstra: *„Hlavní příčinou softwarové krize byl nárůst výkonu hardware - programování nemělo problémy, dokud neexistovaly počítače. Dokud jsme měli slabé počítače, mělo programování jen snesitelně těžké problémy. Nyní máme gigantické počítače a k nim gigantické problémy se softwarem“.*

# Moorův zákon:

- **„Výkon hardwaru vzrůstá zhruba dvakrát za dva roky“.**
  - Přestože sám autor prohlásil svou extrapolaci jako „pěkně divokou“, zákon zhruba platí dodnes.
  - Firma Intel nedávno zveřejnila výsledky výzkumné zprávy uvádějící, že Moorův zákon pravděpodobně přestane platit až kolem roku 2021 (křemík se dostane na hranici svých možností).



# Tvorba software ↔ inženýrství

- Všechny tyto problémy související se softwarovou krizí vedly tedy nakonec k pokusu udělat z vývoje produkováného nadšenci inženýrskou disciplinu.
- V 70-tých letech dochází k formulaci základních principů tohoto oboru.
- Vzniká také první generace nástrojů pro podporu této discipliny, které jsou označovány jako **CASE** (Computer Aided Software Engineering).

# Co to je inženýr?

- Ideální prototyp inženýra představuje C.Smith z Verneova románu Tajuplný ostrov. Uměl vše - vyrobit nitroglycerin i postavit loď.
- Takový inženýr všeuměl mohl možná existovat v 19. století. Dnes suma inženýrských poznatků již značně překračuje kapacitu šedé kůry mozkové jednoho individua.
- Univerzální inženýr je již jen romantickou představou a v našem století by působil spíše jako diletant.



# Etika osobnosti inženýra (FEANI)

## Inženýr:

- *vykonává svou činnost na co nejvyšší úrovni - respektujíc zákony země, v níž působí - tak, aby jím poskytované služby byly v souladu s tím, co je v jeho profesi považováno za úroveň odpovídající současnému stavu poznání,*
- *zachovává profesionální poctivost a intelektuální čest jako záruku nestrannosti v analýze a úsudku a v následném rozhodování,*
- *je vázán každou v dobré víře uzavřenou smlouvou, na kterou dobrovolně přistoupil,*
- *v souvislosti s výkonem své profese nepřijímá žádné peníze bez souhlasu svého zaměstnavatele,*
- *projevuje svou oddanost inženýrské profesi účastí na činnosti inženýrských organizací, zejména takových, které působí při ochraně profesních zájmů a přispívají k rozšiřování vědeckotechnických poznatků a k trvalému zvyšování odbornosti svých členů,*
- *používá pouze ty tituly a označení, na něž má právo.*

# Profesionální etika inženýra (FEANI)

## Inženýr:

- *může přijímat pouze takové úkoly a pověření, která odpovídají jeho kvalifikaci a oprávnění - při zajišťování činností ležících mimo tyto hranice spolupracuje s příslušnými odborníky,*
- *odpovídá za organizování a provádění úkolů, jejichž zajišťování převzal,*
- *zřetelně a úplně specifikuje služby, k jejichž provádění se zavázal,*
- *při plnění úkolů, jimiž je pověřen, činí veškeré nezbytné kroky k tomu, aby byla zajištěna bezpečnost osob a majetku,*
- *přijímá odměnu ve výši odpovídající poskytnutým službám a převzaté odpovědnosti,*
- *pečuje o to, aby každé odměňování, které souvisí s činností, za niž odpovídá, bylo přiměřené poskytnutým službám,*
- *usiluje o dosažení vysoké kvality technických řešení a přispívá ke zvyšování jejich úrovně,*
- *pečuje o vytváření zdravého a příjemného pracovního prostředí pro své spolupracovníky.*

# Rozmanitě inženýrské profese

- Stavební inženýr
- Chemický inženýr
- Inženýr architekt
- Genetický inženýr
- Softwarový inženýr
- ...
- Sociální inženýr?
- ...
- Byli stavitelé katedrál inženýři?



# Co to je softwarové inženýrství?

**Wikipedie:** Pojem „softwarové inženýrství“ není nijak jednotný, může mít víc významů:

- Obecný termín, který znamená mnoho činností, dříve označovaných jako programování? – **rozhodně ne**
- Obecný termín, který znamená praktickou činnost s počítači, na rozdíl od teoretického přístupu, který se nazývá informatika? – **rozhodně ne**
- Argument pro jisté přístupy k programování se zaměřením na inženýrskou profesi, nikoli jako pohled na programování jako druh umění, řemeslné zručnosti a kultury? – **částečně ano**
- Softwarové inženýrství je definované jako standard IEEE 610.12? – **spíše ano**

## Definice IEEE 610.12:

- *„Softwarové inženýrství je aplikace systematického, disciplinovaného, kvantifikovatelného přístupu k vývoji, provozu a údržbě softwaru, tj. aplikace inženýrství na software. Také je to studium postupů dle výše uvedeného.“*

# Je vývoj softwaru umění, věda nebo rutina?

Softwarové inženýrství má blízko k různým disciplínám:

- Na jedné straně je možné jej považovat za inženýrství, neboť se jedná o disciplinované využívání pragmatických zkušeností, tj. rutinní postupy, které se očekávají od inženýra.
- Na druhé straně je možné softwarové inženýrství považovat za vědu, neboť v sobě zahrnuje rozvoj matematických disciplin potřebných k řešení úloh.
- Na straně třetí lze softwarové inženýrství považovat za umění, neboť v sobě zahrnuje aspekty běžně přisuzované umění návrhu – návrh vzhledu, návrh ovládání apod.

# Lze SI srovnat s jinými inženýry?

- Srovnáme-li softwarového inženýra s inženýrem stavebním, pak stavební inženýr realizuje stavbu podle modelu, programátor programuje podle modelu.
- Model stavebnímu inženýrovi navrhl architekt (územní rozhodnutí, stavební povolení, realizace stavby), programátorovi softwarový architekt (úvodní studie, návrh architektury, konceptuální model) a návrhář (logický model).
- Chemický inženýr navrhuje postup výroby nějaké látky z ingrediencí, softwarový návrhář navrhuje skladbu celku z rozmanitých komponent.
- Postup přípravy látky v laboratoři je jiný, než postup výroby v továrně, záleží na použité technologii. Softwarový inženýr rovněž využívá různé technologie podle cílového prostředí.

# Jak se SI vyvíjí?

- Softwarové inženýrství nedávno oslavilo padesátiny. Zajímavá otázka je, jak se softwarové inženýrství za tuto dobu vyvinulo? Jaké trendy či změny jsou nejpodstatnější?
- Jiným zajímavým hlediskem je otázka, proč se některé trendy prosazují s určitým zpožděním, pokud je softwarové inženýrství srovnatelné s ostatními inženýrskými disciplinami.
- Co je v současnosti základní problém, se kterým se softwarové inženýrství potýká? Poučili jsme se již ze zkušeností, podobně jako se to stalo v jiných inženýrských disciplínách?

# Smysl předmětu SWI

- Seznámit se s metodami a nástroji používanými při modelování a realizaci programových systémů, protože to patří k běžnému vybavení absolventů universit v oboru softwarové inženýrství.
- Absolventi KTS VŠP by neměli být pozadu a měli by se umět domluvit s absolventy jiných škol, neboť se zdá, že tyto profese budou ještě určitý čas žádané.
- Předmět SWI se zabývá zejména sběrem požadavků, analýzou a modelováním.
- Návrhem a implementací programových systémů se zabývají jiné předměty.
- Řízením projektů se zabývají jiné předměty.

# Co tedy budeme probírat?

- Vyjdeme z definice toho, co by měl softwarový inženýr znát.
- Od toho odečteme to, co se učí jinde (v jiných předmětech) a zůstane nám obsah tohoto předmětu.
- Všechny materiály a podmínky pro absolvování najdeme na stránkách předmětu, které jsou umístěny na e-learningovém portálu Moodle, který škola pro výuku poskytuje. Adresa portálu je:  
<https://moodle.vspj.cz/course/view.php?id=200214>
- Pro přístup se musíte na portálu autentizovat.

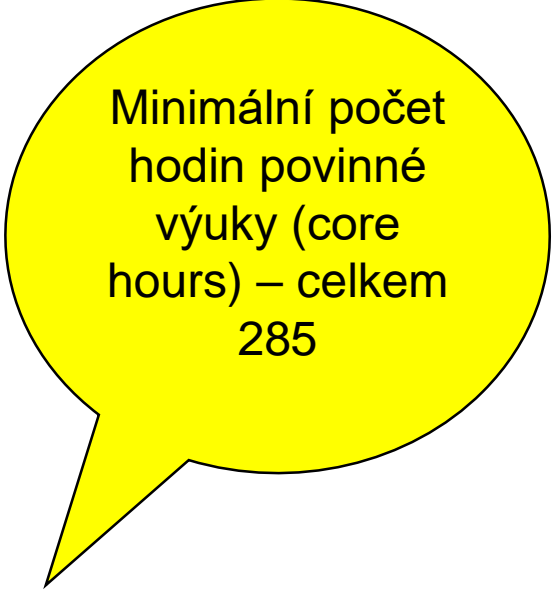
# Sada znalostí softwarového inženýra

- [SWEBOK](#) (Software Engineering Body of Knowledge – IEEE a ACM 2004)
- SEEK (Software Engineering Education Knowledge - [SE2004](#))
  - Co se má učit v bakalářských programech (undergraduate).
  - Pozn.: Na přípravě se podílejí známá jména jako Pressman, Sommerville, McConnell, na revizi ale také Jan Pavelka, Mária Bieliková, Pavol Návrat.
- [GSwE2009](#)
  - Totéž pro magisterské programy (Graduate SwE).



# Základní členění informatiky (CS)

- Diskrétní struktury (43)  
Základy programování (38)  
Algoritmy a složitost (31)  
Architektura a organizace (36)  
Operační systémy (18)  
Výpočty orientované na síť (15)  
Programovací jazyky (21)  
Styk člověka s počítačem (8)  
Grafika a vizualizace (3)  
Inteligentní systémy (10)  
Správa informací (10)  
Sociální a profesionální otázky (16)  
**Softwarové inženýrství (31 – 11%)**  
Počítačová věda a numerické metody (0)



Minimální počet  
hodin povinné  
výuky (core  
hours) – celkem  
285

Zdroj: CS2001 (CS2008)

# Z toho softwarové inženýrství

Povinný základ:

Návrh (8)

Programová rozhraní (API) (5)

Softwarové nástroje a prostředí (3)

Softwarové procesy (2)

Požadavky a specifikace (4)

Validace softwaru (3)

Vývoj softwaru (3)

Řízení softwarových projektů (3)

Volitelné doplňky:

Komponentový vývoj

Formální metody

Spolehlivost softwaru

Vývoj specializovaných systémů

Zdroj: SE2004

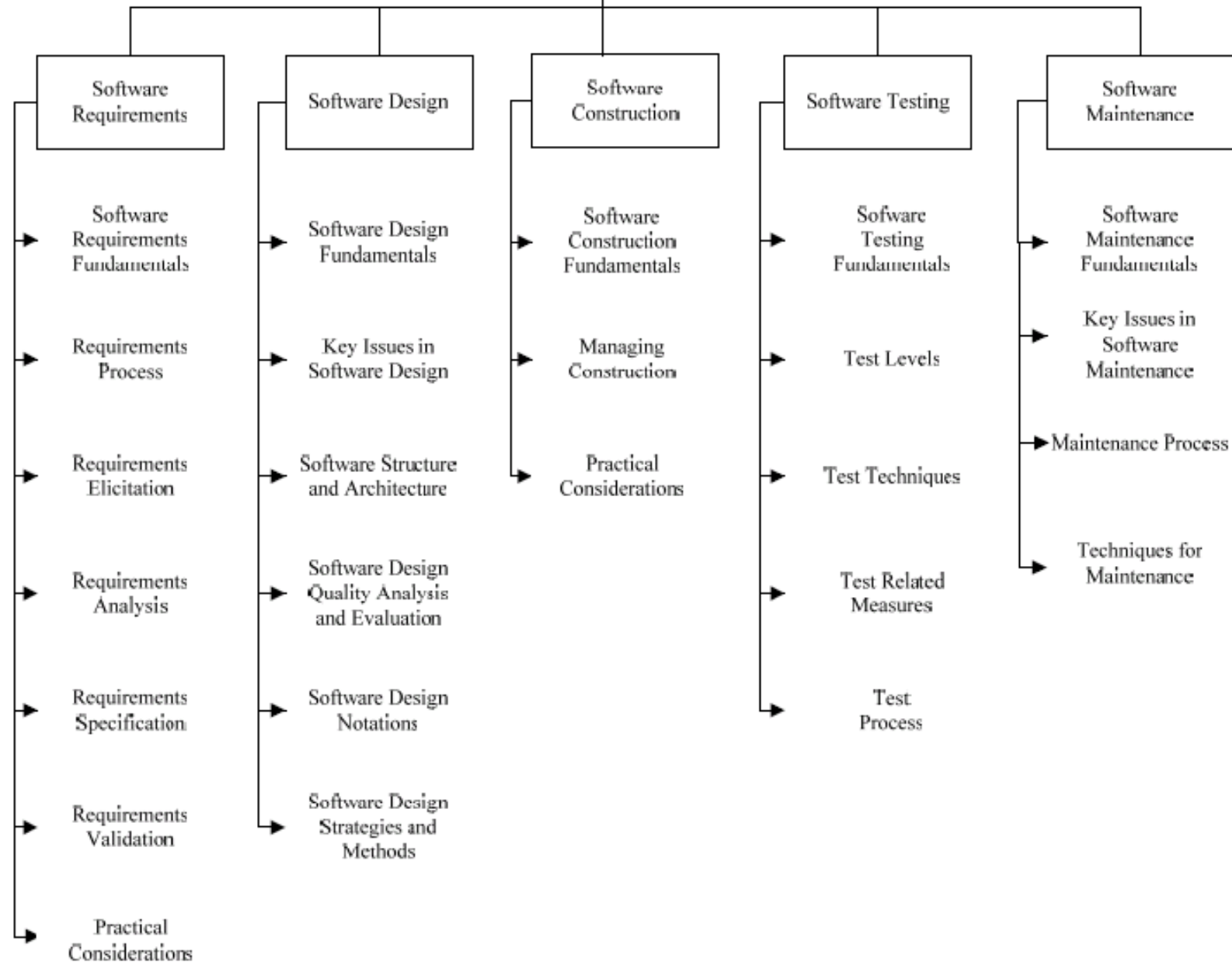
# Základní znalostní oblasti SWI

- Správa požadavků (Software requirements)
- Softwarový návrh (Software design)
- Tvorba softwaru (Software construction)
- Testování softwaru (Software testing)
- Údržba softwaru (Software maintenance)
- Správa konfigurací (Software configuration management)
- Řízení vývoje (Software engineering management)
- Softwarový proces (Software engineering process)
- Nástroje a metody softwarového inženýrství (Software engineering tools and methods)
- Kvalita softwaru (Software quality)

Zdroj: SE2004

# Guide to the Software Engineering Body of Knowledge

## 2004 Version



# Osnova přednášek

- Úvod do softwarového inženýrství
- Plánování projektů
- Modelování požadavků (CIM)
- Analýza (PIM)
- Architektura SW, MDA
- Návrh (PSM)
- Návrhové vzory
- Metodiky
- Testování

# Softwarové týmy a softwarové profese

- Jedním z projevů přechodu od ruční výroby k manuatuře je definice softwarových profesí.
- Řešení velkých projektů vyžaduje spolupráci mnoha řešitelů a práci je nutno rozdělit.
- Dělbba práce vyžaduje organizaci týmů řešících větší softwarové projekty.
- Týmy lze organizovat jako strukturované nebo nestrukturované.

# Organizace týmů

## Nestrukturované týmy

- Dělí práci podle objemu.
- Mohou být organizovány jako:
  - „Osamělí vlci“
  - „Horda“
  - „Demokratická skupina“

## Strukturované týmy

- Dělí práci podle profese.
- Mohou být organizovány jako:
  - „Chirurgický tým“
  - „Tým hlavního programátora“
  - „Agilní skupina“
  - „Více-týmová organizace“

# Kterou organizaci zvolit?

- Volba organizace je dána rozsahem projektu.
- Na větší projekty je třeba více-týmová organizace.
- Pro větší projekty se samozřejmě hodí strukturované týmy.
- Máme-li dost prostředků, je nejvýkonnější chirurgický tým.
- Nemáme-li, nejefektivnější může být agilní skupina.



**The End**