1. Softwarové inženýrství(SI)
   1. Definice: Softwarové inženýrství je systematický, disciplinovaný a kvalifikovaný přístup k vývoji, tvorbě a údržbě software
   2. Projekt je dobře definovaná posloupnost činností, která je zaměřena na dosažení nejistého cíle, má určen začátek a konec, a je uskutečňován pomocí zdrojů – lidí a prostředků
   3. Pokud plánovaná posloupnost činností nemá určen začátek a konec, jedná se o záměr
   4. Historie:
      1. Termín „Software“ zaveden roku 1958 – John Tukey
      2. Termín „Softwarové inženýrství“ 1968 - NATO sponzoruje první konferenci s tímto názvem a tímto tématem
      3. „Hardware“ původní název pro to co se prodává v železářství
      4. „Softwarové inženýrství je disciplína, která se zabývá zavedením a používáním řádných inženýrských principů do tvorby SW tak, abychom dosáhli ekonomické tvorby SW, který je spolehlivý a pracuje účinně na dostupných výpočetních prostředcích“
   5. Příčina vzniku
      1. Přibývalo SW projektů, ale zmenšovalo se procento dokončených projektů
      2. Docházelo i k velkým haváriím jako Mariner 1(Sonda, která počítal se špatnými proměnnými)
      3. Na programátory jako jednotlivce se dalo spolehnout do doby, než měli počítače určitý výkon po této fázi se programuje již v SW týmech.
   6. Moorův zákon
      1. „Počet tranzistorů, které mohou být umístěny na integrovaný obvod, se při zachování stejné ceny zhruba každý 18 měsíců zdvojnásobí“
      2. Dnes chápáno spíše ve významu exponenciálního růstu výpočetního výkonu
   7. Softwarová krize
      1. Hlavní příčinou softwarové krize byl nárůst výkonu hardware a nezkušeností sowftware profese
      2. Tento rozdíl způsobuje rozevírání nůžek a důsledkem pak jsou softwarová krize
      3. Projevy:
         * 1. Projekty překračují rozpočet a čas
           2. SW nemá dostatečnou kvalitu a neodpovídá požadavkům
           3. Projekt není dobře řiditelný a SW je obtížně udržovatelný
      4. Trvá dodnes
      5. SW expert již není schopen obsáhnout celý problém
      6. Proto se celý projekt začal řešit v týmech a začala se používat technika „ rozděluj a panuj“ – problém ale je v rozdrobení projektu na menší podproblémy a ty rozdělit
      7. Vše vedlo k tomu z oboru pro nadšence udělat inženýrskou disciplínu
   8. Inženýrská práce:
      1. Než vůbec začne cokoliv vyrábět nebo programovat namodeluje si a nakreslí si celý problém
      2. K tomu potřebuje zavedenou notaci a vědecky podložené metody a postupy
      3. Použitím vhodné technologie docílí toho, že je model efektivní a vydrží
      4. Programátor je pak schopen podle tohoto modelu programovat
   9. Znalostní oblasti SI
      1. Správa požadavků
      2. SW návrh
      3. Tvorba SW
      4. Testování SW
      5. Řízení vývoje
   10. Co přineslo SI
       1. SW profese a týmy
       2. Modely životního cyklu
       3. Oddělení správy dat od správy funkčnosti
       4. Objektově – orientované metody
   11. Co zmizelo díky SI
       1. Pocit opravdových programátorů
       2. Citát od neznámého programátora: „Softwarové inženýrství znamená zavedení disciplíny do volné tvorby software. Žádný opravdový programátor ho proto nemůže mít příliš v lásce, neboť jej nutí vytvářet nesmyslnou dokumentaci a další podobné artefakty
   12. Problémy vývoje softwaru
       1. Nepřesné pochopení požadavků koncového uživatele
       2. Neschopnost pružně reagovat na změnu požadavků
       3. Nízká kvalita a výkon sw
       4. Nekoordinovaná práce v týmu
       5. Příčiny:
          1. Nepřesná specifikace požadavků
          2. Málo precizní komunikace
          3. Obrovská komplexnost
          4. Nedostatečné testování
   13. SW proces
       1. SW proces je po částech uspořádaná množina kroků směřující k vytvoření nebo úpravě sw produktu
       2. Krokem může být aktivita nebo podproces
       3. Aktivity a podprocesy mohou probíhat v čase současně tudíž je vyžadována jejich kooordinace
       4. Je důležitá znovu použitelnost
       5. Statická struktura procesu definuje Kdo, co, jak, a kdy má vytvořit
       6. Vždy je vhodné mít na projektu nadefinovaný proces
       7. Inženýr nepíše inženýr kreslí
   14. Capability Maturity Model(CMM)
       1. Sw proces: neexistuje referenční proces, pro každý projekt je vhodné něco jiného podle: velikosti organizace, znalosti technologii, časový a finanční rámec, jako základní je brán vodopádový proces
       2. Úrovně vyspělost sw procesu
          1. Počáteční: firma nemá definován sw proces
          2. Opakovatelná: firma identifikoval v jednotlivých projektech opakovatelné postupy
          3. Definovaná: softwarový proces je definován
          4. Řízená: na základě definovaného sw procesu je firma schopna jeho řízení monitorovat
          5. Optimalizovaná: zpětnovazební informace získaná dlouhodobým procesem monitorováním
   15. Vodopádový proces:
       1. Rozdělení životního cyklu sw díla na 4–5 základních fází
          1. Analýza požadavků
          2. Návrh sw systému
          3. Implementace(kódování)
          4. Testování
          5. Provoz
       2. Následující množina činností spjatá s danou fází nemůže započít dříve, než skončí předchozí
       3. Výsledky předchozí fáze „Vtékají“ jako vstupy do fáze následující
       4. Základní nedostatky:
          1. Prodleva mezi zadáním projektu a vytvořením spustitelného systému je příliš dlouhá
          2. Nelze odhalit výslednou kvalitu produktu danou splněním všech požadavků, dokud není výsledný sw systém hotov
   16. Inkrementální
       1. Postupné vytváření verzí sw systému
       2. Zahrnuje stále širší funkcí definovaných postupně v průběhu jeho vytváření
       3. Řada menších vodopádků s kratším cyklem
   17. Spirálový
       1. Jako inkrementální
       2. Přidává mezi jednotlivé iterace vytvoření a hodnocení prototypu
   18. Průzkumové programování
       1. Snaha zjistit od zákazníka co vlastně chce, tím že mu předkládáme hotové systémy
       2. Analýza požadavků je odbytá hrubou specifikací a vše směřuje k rychlé realizaci sw ke které se zákazník vyjádří
   19. Moderní softwarové procesy
       1. Všechny moderní metodiky vycházejí z iterativního/inkrementálního vývoje sw
       2. RUP(Rarional unified process)
          1. Univerzílní proces
          2. Vhodnější pro větší týmy a projekty
          3. Odpověď na problémy vodopádového procesu
          4. Definuje disciplinovaný přístup k přiřazování úkolů a zodpovědností v rámci vývoje organizace
          5. Cílem je zajistit vytvoření produktu vysoké kvality požadované zákazníkem v rámci predikovatelného rozpočtu a časového rozvrhu
          6. Princip
             1. Sw je zde vyvíjen iterativně (v cyklech)
             2. Na konci každé iterace je spustitelný kód(verze)
             3. Při vývoji se využívá existujících komponent
             4. Snaha o skládání produktu z prefabrikovaných (ověřených) komponent
             5. Model sw systému je vizualizován pomocí UML
             6. Součástí RUP jsou i metody pro správu požadavků

Ve všech aktivitách se neustále ověřuje kvalita sw produktu

RUP zahrnuje i princip řízení změn

* + - 1. Výhody
         1. Obecnost a mohutnost
         2. Iterativní přístup – včasné odhalení rizik
         3. Provázanost s notací UML, dokumentace
      2. Nevýhody
         1. Komerční, placený produkt
         2. Rozsáhlost RUP může být na škodu u malých týmů – tým stráví spoustu času implementací metodiky
         3. Její použití vyžaduje hloubkové studium, týká se i projektových manažerů
      3. Cyklus vývoje v RUP
         1. Každý cyklus vede k vytvoření verze systému, kterou lze předat uživatelům a může být s nimi konzultována
         2. Cyklus probíhá těmito fázemi

Zadání: transformace původní myšlenky do vize koncového produktu

Rozpracování: dochází k analýze požadavků, tvorbě podrobných specifikací a návrhu architektury sw

Tvorba: sw je naimplantován a otestován

Předání: zhotovená verze systému je předána uživateli do užívání, zahrnuje beta testování a zaškolení

* + - * 1. Každá fáze může být dále rozložena do několika iterací

Iterace vývoje(toky): je úplná vývojová smyčka vedoucí k vytvoření spustitelné verze systému reprezentující podmnožinu vyvíjeného cílového produktu a která je postupně rozšiřována až do výsledné podoby

V rámci každé iterace probíhají činnosti, které jsou spořádány do toků charakteristickým účelem základních toků

Výsledkem iterace je část sw produktu (artefakt)

Základní toky

Byznys modelování – popisující strukturu a dynamiku procesu podniku či organizace

Specifikace požadavků – definující prostřednictvím specifikace tzv. případů použití sw systému jeho funkcionalitu

Analýza a návrh – zaměřené na specifikaci architektury sw produktu

Implementace – reprezentující vlastní tvorbu sw, testování komponent a jejich integraci

Testování: Zaměřené na činnosti spjaté s ověřením správnosti řešení sw v celé jeho složitosti

Rozmístění: Zabývající se problematikou konfigurace výsledného produktu na cílové počítačové infrastruktuře

Podpůrné toky

Mimo základní toky existují i toky podpůrné, které přímo nevytváří hodnotu, ale jsou nutné pro realizaci základních toků

Řízení změn a konfigurací

Projektové řízení

Prostředí a jeho správa

* + 1. Agilní metody
       1. Snaha o nejrychlejší dodání produktu(ne nutně finálního)
       2. Extrémní programování
       3. Scrum
       4. Test driven development
  1. Business
     1. V sw inženýrství se často operuje se slovem business
     2. Nesouvisí to nijak s komercí
     3. Jedná se o operační procesy, kterými se řídí fungování zákazníka
     4. Business modelování ve škole
        1. Kdo je učitel, kdo žák, kdo inspektor, jak se známkuje
  2. Doména:
     1. Oblast znalostí
     2. Doménová znalost je schopnost chápání procesů a terminologie v oblasti, pro kterou je sw vyvíjen(doménová znalost jaderná fyzika)
     3. Doménová znalost je doplňková znalost nutná k vývoji sw