**Software Quality Engineering (SQE)**

**Inžinierstvo kvality softvéru** – proces, ktorý

* hodnotí,
* oceňuje
* vylepšuje kvalitu softvéru.

Kvalita softvéru je často definovaná ako miera vyhovovania požiadavkám na softvér:

spoľahlivosť, udržovateľnosť, prenositeľnosť,

atď... v kontraste s funkcionalitou, produkciou a rozhraním požiadaviek, ktoré zodpovedajú výsledkom softvérového inžinierstva.

Chápanie kvality softvéru

* statický pohľad na chápanie kvality - posudzovanie softvérového produktu prostredníctvom **súboru charakteristík kvality**
* modely kvality:
  + ISO/IEC 9126
  + McCall

Softvér – IEEE definícia

Softvér je:

počítačový program, procedúry, možná pripojená dokumentácia a dáta týkajúce sa operácií v počítačovom systéme.

* Počítačový program (kód)
* Procedúry
* Dokumentácia
* Dáta potrebné k prevádzke softvérového systému

9 príčin softvérových chýb

* Chybne definované požiadavky
* Chyby komunikácie klient – vývojár
* Zámerné odchýlky od softvérových požiadaviek
* Logické návrhové chyby
* Chyby v kóde
* Nezhody s dokumentáciou a kódovými inštrukciami
* Nedostatky v testovacom procese
* Procesné chyby
* Dokumentačné chyby

Kvalita softvéru – IEEE definícia

Kvalita softvéru je:

1. Miera, v ktorej systém, **komponent** alebo **proces** spĺňa špecifikované

požiadavky.

1. Miera, v ktorej systém, **komponent** alebo **proces** spĺňa klientove alebo používateľove **potreby** alebo

očakávania.

Kvalita softvéru – definícia Pressman-a

**Kvalita softvéru je definovaná ako:**

Súhlas explicitne uvedených

funkcionálnych a splnených požiadaviek, explicitne dokumentovaný

štandardizovaný vývoj a implicitné charakteristiky, ktoré sú očakávané od každého profesionálne vyvíjaného softvéru.

Záruka kvality softvéru – IEEE definícia

Software quality assurance – SQA Záruka kvality softvéru je:

* Plánovaný a systematický vzor všetkých akcií potrebných pre poskytnutie adekvátnej istoty, že položky a/alebo produkty sa riadia stanovenými technickými požiadavkami.
* Množina aktivít navrhovaných na hodnotenie procesu, ktorým sú produkty navrhované alebo vyrábané. Kontrast s kontrolou kvality.

Záruka kvality softvéru – rozšírená definícia

Záruka kvality softvéru je:

Systematická a plánovaná množina aktivít potrebných pre poskytnutie adekvátnej

istoty, že podpora procesu vývoja softvéru

a/alebo proces vývoja a údržby softvérového systému sa riadi stanovenými technickými

požiadavkami a manažérskymi požiadavkami

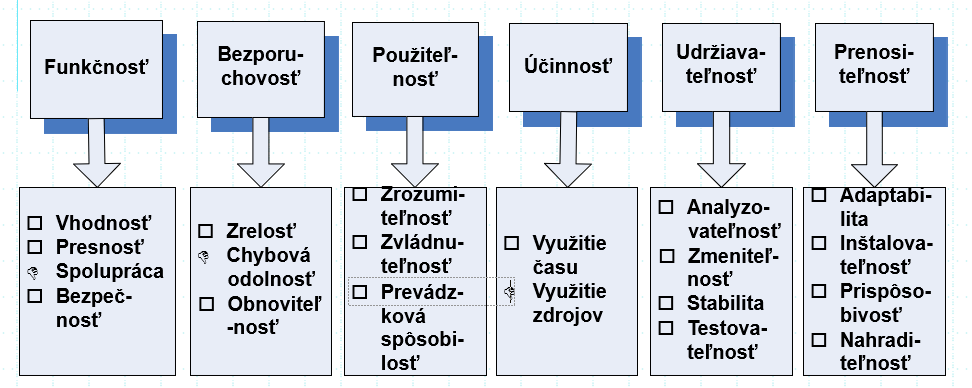
podporujúcimi plán a riadenie v hraniciach rozpočtu.

ISO-9126

Poskytuje hierarchickú štruktúru pre definíciu kvality organizovanú do charakteristík a subcharakteristík kvality.

Charakteristiky kvality podľa ISO/IEC-9126

* Funkčnosť (functionality)
* Bezporuchovosť (reliability)
* Použiteľnosť (usability)
* Účinnosť (efficiency)
* Udržiavateľnosť (maintainability)
* Prenositeľnosť (portability)
* **Kvalita softvérového produktu**



Software quality model (model kvality )

Productivity model ( model produktivity)

Certifikácie ISO na Slovensku

[http://www.snas.sk](http://www.snas.sk/)

SNAS Slovenská národná akreditačná služba

<http://www.tuvslovakia.sk/certifikacie.htm>

TÜV SÜD Slovakia s.r.o. je prvým slovenským neštátnym inšpekčným a certifikačným orgánom, ktorý v domácom trhovom prostredí, za domáce ceny umožní slovenským výrobcom a exportérom získať doklady a dokumenty signované značkou TÜV, značkou najväčšej európskej certifikačnej spoločnosti s dlhoročnou tradíciou.

Takéto dokumenty uľahčujú umiestnenie našich výrobkov na trhoch Európskej únie a na ďalších vyspelých trhoch sveta.

**Softvérové Metriky**

* Softvérové metriky možno definovať ako kritériá určujúce atribúty softvérového projektu.
* **Metriky pre výrobok** – opisujú vlastnosti produktu (veľkosť, zložitosť, spoľahlivosť, atď.).
* **Metriky pre proces** – sledujú vlastnosti procesu ako úsilie, náklady, čas, a pod.
* **Metriky pre zdroje**
* Softvérové metriky pre
  + Analýzu
  + Návrh
  + Implementáciu

Metriky v analýze

* Funkčné body
* Features points (body charakteristík)
* Uplatnenie pri odhadovaní finančných a časových nákladov

Funkčné body

* Albrecht 1979
* Vychádza zo špecifikácie požiadaviek
* Určuje počet a rozsah používateľovi prístupných služieb
* Možno stanoviť dosť presne už na začiatku

Výpočet

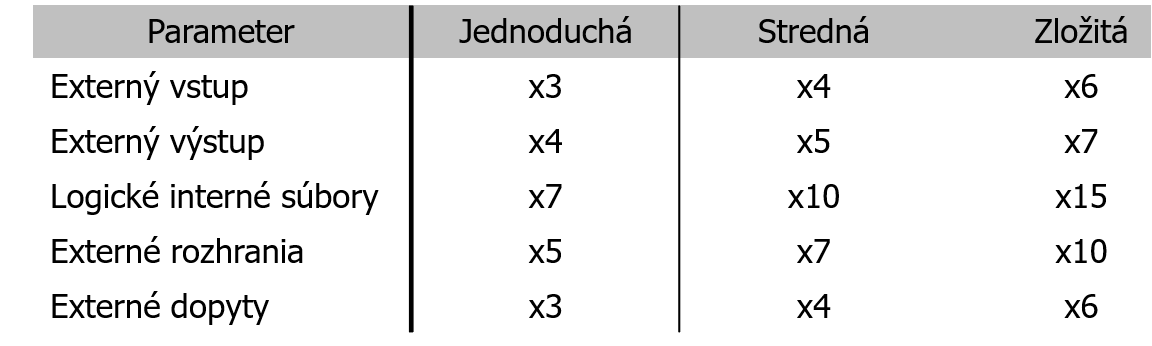
1. Určenie charakteristík softvérového projektu
2. Klasifikácia charakteristík
3. Neupravené funkčné body
4. Úprava funkčných bodov
5. Výpočet funkčných bodov

Určenie charakteristík

* Počet externých vstupov
* Počet externých výstupov
* Počet logických interných súborov
* Počet externých rozhraní
* Počet externých dopytov

Klasifikácia charakteristík

* Charakteristiky podľa zložitosti – jednoduchá, stredná, zložitá
* Metódou pokus-omyl určené váhové koeficienty



Neupravené funkčné body

* Výpočet podľa vzorca

*UFC* = å(*Parameter* \* *váhový* \_ *faktor* \_ *zložitosti*)

Úprava funkčných bodov

* Pre realistickejšie ohodnotenie konkrétneho projektu
* 14 charakteristík projektu, napr. či je výkonnosť systému kritická, či sa uvažuje jednoduchosť použitia, ...
* Každá charakteristika ohodnotená 0 až 5 v závislosti od vplyvu na systém

Výpočet funkčných bodov

* Koeficient úpravy funkčných bodov

*CAF* = 0,65 + 0,01\* å*vi*

* Výpočet funkčných bodov

*FP* = *UFC* \* *CAF*

Výhody metriky funkčných bodov

* Nezávislá na jazyku
* Nemusí byť použitá na kód
* Potrebné údaje sú dostupné v skorých fázach návrhu, požadovaná je detailná špecifikácia
* Presnejšia ako niektoré iné metriky (napr. počet riadkov kódu – LOC)

Nevýhody

* Subjektívne počítanie
* Ťažké zautomatizovať a vypočítať
* Ignoruje kvalitu výstupu
* Zameraná na tradičné „data processing“ aplikácie (aplikácie spracovávajúce veľké množstvá dát)

Features points

* Funkčné body nie sú vhodné pre systémy reálneho času, operačné systémy alebo vnorené systémy
* Pre systémy s veľkou vnútornou algoritmickou zložitosťou
* Vychádza z funkčných bodov
* Pridáva novú charakteristiku *algoritmus*

s váhou 3

Metriky v návrhu

* Zviazanosť modulov – miera, akou medzi sebou jednotlivé moduly závisia, vplyv na udržiavateľnosť

a testovateľnosť, snaha o čo najmenšiu

* Súdržnosť modulov – ako úlohy vykonávané modulom medzi sebou súvisia, snaha o čo najväčšiu

Zviazanosť – Metrika Fentona a Meltona (1)

* Pre každú dvojicu modulov sa určí miera zviazanosti

5. Obsahová zviazanosť

4. Globálna zviazanosť

3. Riadiaca zviazanosť

2. Typová zviazanosť

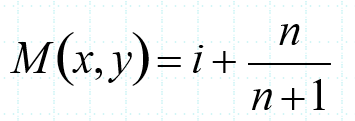
1. Dátová zviazanosť

0. žiadna zviazanosť

Zviazanosť – Metrika Fentona

a Meltona (2)

* Zviazanosť medzi dvoma modulmi



* i – maximálna trieda zviazanosti medzi x a y
* n – počet prepojení medzi modulmi
* Celková zviazanosť systému je medián zviazanosti všetkých modulov

Metrika zviazanosti na základe interakcií (1)

* Interakcie typu
  1. Dátová deklarácia – Dátová deklarácia (DD)

**Ak zmena deklarácie alebo použitia A zapríčiní potrebu zmeny deklarácie alebo použitia B.**

* 1. Dátová deklarácia – Podprogram (DS) **Ak dátová deklarácia interaguje s aspoň jednou dátovou deklaráciou podpogramu.**

Metrika zviazanosti na základe interakcií (2)

* Vstupná zviazanosť softvérovej súčiastky (import coupling)
* Počet interakcií DD medzi vonkajšími a vnútornými dátovými deklaráciami

vzhľadom na softvérovú súčiastku. Vysoká hodnota znižuje udržiavateľnosť, zrozumiteľnosť a znovupoužiteľnosť.

Metrika zviazanosti na základe interakcií (3)

* Výstupná zviazanosť softvérovej súčiastky (export coupling)
* Počet interakcií DD medzi vnútornými a vonkajšími dátovými deklaráciami

vzhľadom na softvérovú súčiastku. Vysoká výstupná zviazanosť nemusí byť automaticky ukazovateľom zlého návrhu.

Súdržnosť vo fáze návrhu

* Šesť úrovní súdržnosti medzi dvojicami výstupov z modulu
* Náhodná, Podmienečná, Iteračná, Komunikačná, Postupná, Funkcionálna
* Pre každú dvojicu najsilnejšia súdržnosť
* Súdržnosť modulu je najslabšia zo všetkých párov

Softvérové metriky v implementácii

* Počet riadkov zdrojového kódu
* Halsteadove metriky
* Cyklomatická zložitosť

Počet riadkov zdrojového kódu

* Jednoduchá metrika, jej vypočítanie sa dá ľahko zautomatizovať
* Počítanie fyzických riadkov – oddelených oddeľovačom riadkov, bez komentárov

a prázdnych riadkov

* Závisí od formátovania textu
* Počítanie logických príkazov – stanovenie začiatku a konca príkazu, napr. pomocou oddeľovačov (v C ;)

Halsteadove metriky

* Rozdelenie zdrojového kódu na
  + Operátory
  + Operandy
* Veľkosť programového slovníka

**n1 a n2 počet rôznych operátorov a oper***n***ndov***n*=*n*1+2*n*

* Dĺžka programu

**N1 a N2 celkový počet operátorov a operandov**

*N* = *N*1 + *N*2

Halsteadove metriky

* Odhad dĺžky programu

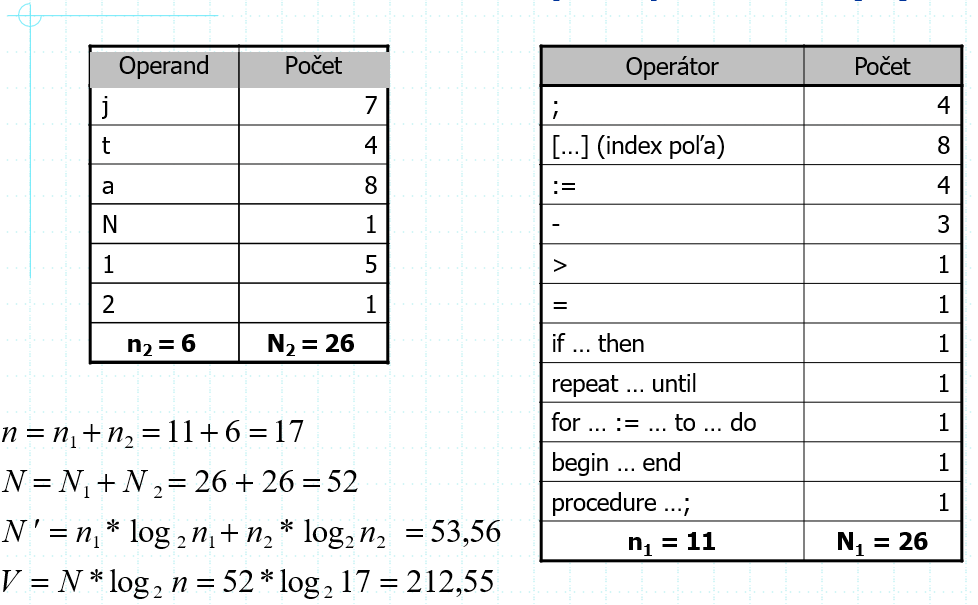
*N*'= *n*1 \*log2 *n*1 + *n*2 \*log2 *n*2

* Objem programu

*V* = *N* \* log 2 *n*

Halsteadove metriky – príklad (1)

Halsteadove metriky – príklad (2)



Cyklomatická zložitosť

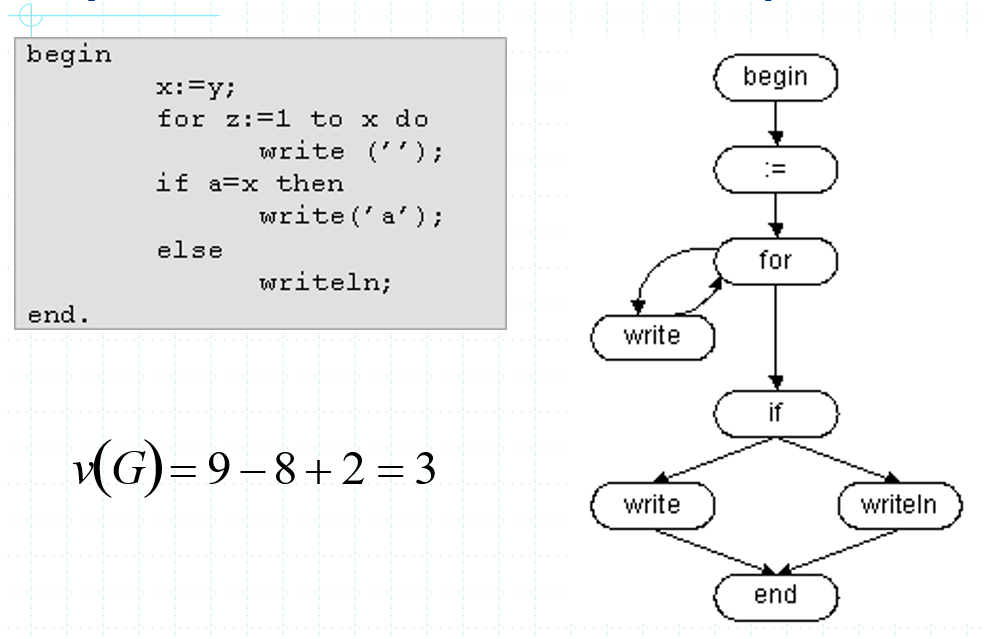
* Na posúdenie zložitosti programu, LOC sa ukázala ako nedostatočná
* Diagram toku riadenia
* Uzol je blok programu a hrana spája uzly, ak druhý uzol môže byť vykonávaný hneď po prvom
* Cyklomatická zložitosť – počet lineárne nezávislých ciest programu

Cyklomatická zložitosť

*v*(*G*) = *e* - *n* + 2

* e počet hrán
* n počet uzlov

Cyklomatická zložitosť – príklad



**Softvérové Metriky Objektovo-orientované**

* Vysvetlenie pojmu „softvérová metrika“
* Proces zavádzania metrík
* Objektovo-orientované metriky
* Interpretácia získaných hodnôt

Vysvetlenie pojmu „softvérová metrika“

**Definícia:**

Softvérová metrika je spôsob merania určitej softvérovej veličiny alebo softvérovej špecifikácie.

Aplikácia kvantitatívnych metód v iných vedách priniesla zaujímavé a užitočné výsledky, preto sa softvéroví inžinieri rozhodli zaviesť podobné prístupy aj do vývoja softvéru.

Metriky poznáme tradičné, objektovo-orientované a procesné.

Výhody používania metrík

* získavanie údajov, ktoré nám v budúcnosti pomôžu ako referencia a dobrý, resp. zlý príklad
* viditeľnosť stavu projektu – aktuálny stav je nevyhnutný pre ďalšie plánovanie
* metriky neskvalitnia softvér, ale poskytnú informácie na zlepšenie jednotlivých procesov

Ako pristupovať k meraniu – GQM

* Výber reprezentatívnych veličín na meranie metódou GQM
* **GQM** – Goal, Question, Metric
* **Goal** – určenie cieľa (*Zníženie nákladov na údržbu o 50% za rok*)
* **Question** – otázky potrebné na dosiahnutie cieľa (*Koľko minieme mesačne na údržbu?*)
* **Metric** – zvolenie konkrétnych metrík, ktoré nám pomôžu nájsť odpovede na otázky (*Zoznam hodín strávených na jednotlivých častiach údržby*)

Zavádzanie metrík do povedomia zamestnancov

* Nevyhnutná podmienka úspešného uplatnenia metrík

**Problémy:**

* Strach zamestnancov, že budú trestaní alebo odmeňovaní na základe nameraných údajov
* Strach z nedostatku času na produktívnu činnosť

**Riešenia:**

* vzdelávanie a diskutovanie so zamestnancami
* používanie aplikácií na podporu zhromažďovania údajov

Rady pri zavádzaní metrík

* **Začínať pomaly**

Vhodne zvoliť menšiu množinu metrík a neskôr ju rozširovať. Metriky sa musia navzájom dopĺňať. Je napr. zbytočné monitorovanie produktivity, ak nie je doplnená sledovaním kvality.

* **Vysvetľovanie a vzdelávanie**

Všetkým členom tímu je dôležité vysvetliť zmysel metrík.

* **Analyzovanie údajov**

Diskutovanie celkových nameraných údajov s členmi tímu.

* **Definovanie dát, procedúr a procesov** Všetky pojmy, s ktorými sa človek v procese stretne, musia byť presne zadefinované.
* **Analyzovanie trendov a porovnávanie**

Dáta treba analyzovať nielen v kontexte daného projektu,

ale je nutné ich porovnávať aj s inými projektmi.

* **Nepresnosť metrík**

Pri analýze údajov treba mať na pamäti, že žiadna metrika

nie je dokonalá.

Weighted Methods per Class

WMC = suma zložitostí metód triedy

* Odhad času na vývoj, testovanie a údržbu
* Vplyv na triedy odvodené dedením
* Vyššie WMC → viac špecializovaná trieda

→ menej vhodná na znovupoužitie

Response for Class

RFC = počet metód, ktoré môžu byť vyvolané ako reakcia na prijatú správu

* Zložitosť komunikácie → zložitosť na pochopenie, testovanie, údržbu

a modifikáciu

* Čas na testovanie by mal byť úmerný najhoršiemu scenáru volania metód

Lack of Cohesion in Methods

LCOM = počet dvojíc metód, ktoré nepristupujú k tomu istému atribútu triedy

* Vysoká súdržnosť:
  + Podporuje zapúzdrenie
  + Menšia pravdepodobnosť chýb
  + Znovupoužiteľnosť

Coupling Between Object classes

CBO = počet tried, s ktorými je daná trieda zviazaná (*využíva metódy alebo atribúty*)

* Vysoká zviazanosť:
  + Znižuje modulárnosť
  + Znemožňuje znovupoužitie
  + Komplikuje testovanie, údržbu a modifikáciu

Depth of Inheritance Tree

DIT = max. vzdialenosť uzla stromu dedenia od koreňa

* Čím je trieda hlbšie, tým viac metód zdedila → zložitejšie testovanie
* Na druhej strane: hlbší strom dedenia ≈ viac znovupoužitého kódu

Number of Children

NOC = počet priamych potomkov triedy v hierarchii dedenia

* Viac dedenia ≈ viac znovupoužitého kódu
* Ale môže to indikovať aj:
  + nesprávna abstrakcia rodičovskej triedy
  + nevhodné použitie dedenia

Average Method Complexity

* *Priemerná zložitosť metódy = Suma cyklomatických zložitostí všetkých metód / Celkový počet metód v aplikácii*
* Vyššia zložitosť má za následok zhoršenie udržiavateľnosti, zhoršnie zrozumiteľnosti kódu, nepriaznivo ovplyvňuje spoľahlivosť a testovateľnosť aplikácie

Application Granularity

* *Granularita aplikácie = Celkový počet objektov / Celkový počet funkčných bodov*
* Cieľom OO návrhu je dosiahnuť jemnú granularitu
* Tým sa dosiahne jednoduchšia udržiavateľnosť, zlepšená znovupoužiteľnosť objektov, zlepšená zrozumiteľnosť kódu

Factoring Effectiveness

* *Efektivita faktorizácie = Počet jedinečných metód / Celkový počet metód*
* Čím je menší počet miest, ktoré sa ovplyvnia pri implementovaní zmeny, tým je menšia pravdepodobnosť vzniku chyby
* Vyššia faktorizovanosť má za následok zvýšenie spoľahlivosti a udržiavateľnosti
* Čím viac je faktorizovaná hierarchia dedičností, tým je vyšší stupeň, do akého môže nastať znovupoužitie metód

Degree of Reuse of Inheritance Methods

* *Percento využitia zdedených metód = (Počet využitých zdedených metód / Počet všetkých zdedených metód) × 100*
* *Percento preťažených zdedených metód = (Počet preťažených zdedených metód / Počet všetkých zdedených metód) × 100*
* Definovanie metód znovupoužiteľných cez dedičnosť negarantuje, že tieto metódy sú naozaj znovupoužité

Object Library Effectiveness

* *Priemerná hodnota = Celkový počet znovupoužití objektov / Celkový počet objektov v knižnici*
* Ak sú objekty navrhnuté tak, aby sa dali znovupoužiť vo viacerých aplikáciách, potom by sa to malo prejaviť v štatistikách pre použitie knižnice objektov

Nástroje pre softvérové meranie

Source monitor

súčasti a pluginy vývojových prostredí

JCSC

CheckStyle JavaNCSC JMT

CCCC

....

Nástroje pre softvérové meranie

**Testovanie V-model Kvality, V-**

Testovanie

**Testovanie softvéru** je súbor procesov slúžiacich na kontrolu kvality softvérového produktu, ktorých cieľom je dosiahnutie

požadovanej kvality softvéru z hľadiska

* funkčnosti,
* spoľahlivosti,
* výkonnosti,
* Použiteľnosti
* podporovateľnosti.

Kontrola kvality sa môže uskutočniť či už pre jednotlivé časti informačného systému alebo pre systém ako celok.

Testovanie možno vykonať:

verifikáciou

Alebo

Validáciou.

Aký je podľa vás rozdiel medzi verifikáciou a validáciou?

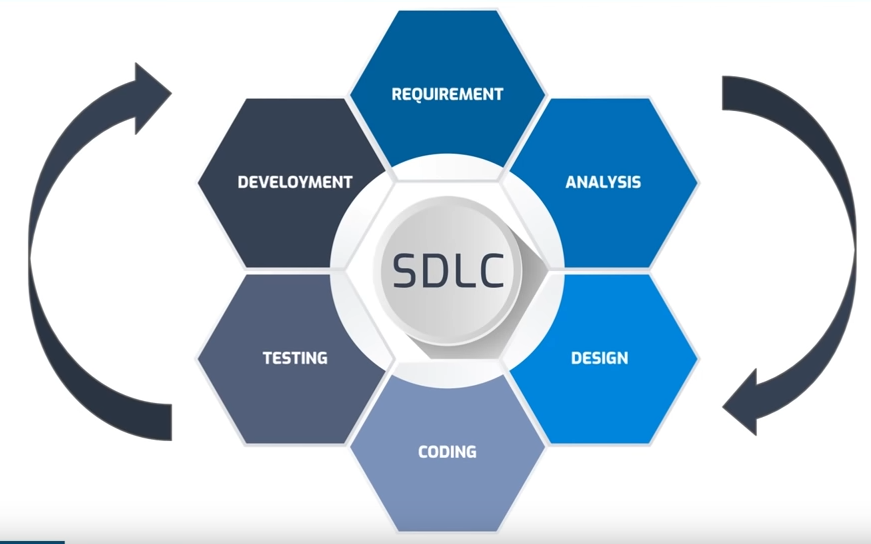
Verifikácia - voči niečomu čo existuje

Validácia - overovanie voči reálnej potrebe

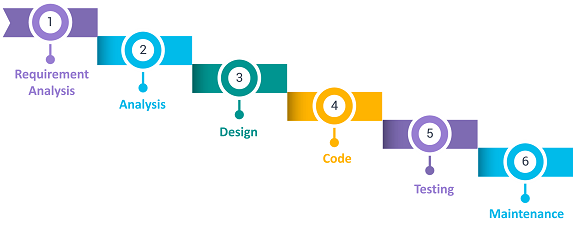
**Kto vykonáva testovanie SW:**

* Software tester
* Project Developers
* Project Lead
* Time Manager
* End User.

**SDLC – Softwere develompent Life-Cicle**



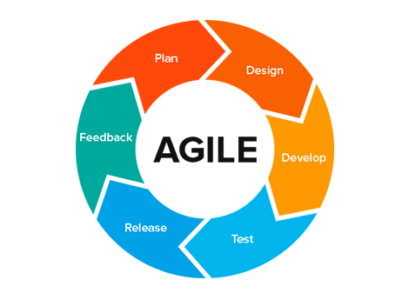
**Waterfall Model**



**Bohem Spiral Model**



**Waterfall Model vs Agile Model**



**Postup testovania softwaru**

* **Biela skrinka – white box testing (structural)**
* **Čierna skrinka – black box testing (functional)**
* **Šedá skrinka – gray box testing**

**Spôsob testovania softwaru**

* **Statické testovanie (revízia, vnútorný systém kvality)**
* **Dynamické testovanie (testovanie sw počas behu)**

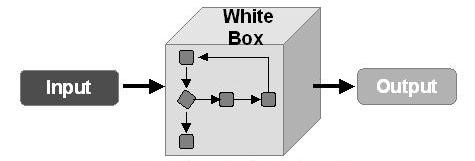
**Iný pohľad na testovanie softwaru**

* **Manuálne**
* **automatické**

**Nástroje automatického testovania**

1. **Selenium**
2. **TestingWhiz**
3. **HPE Unified Funcional Testing**
4. **TestComplete**
5. **Ranorex**
6. **Sahi**
7. **Vaitir**
8. **Tosca TestStuite**
9. **Telerik TestStudio**
10. **Waitin**

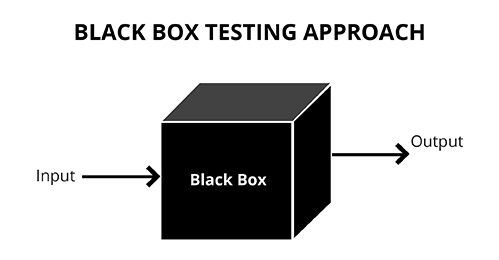
**Biela skrinka „White Box“**



**Test je vytváraný na základe vnútornej stavby SW.**

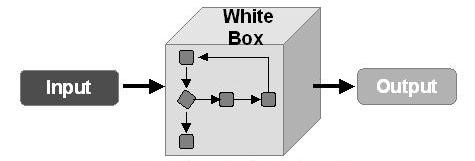
* **rozhodovacie bloky**
* **cykly**
* **podmienky**

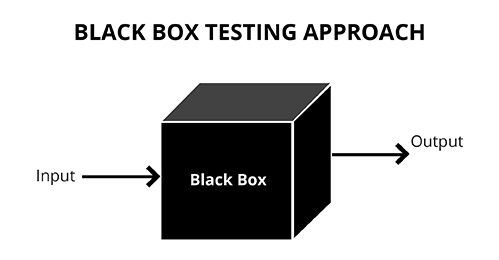
**Čierna skrinka „Black Box“**

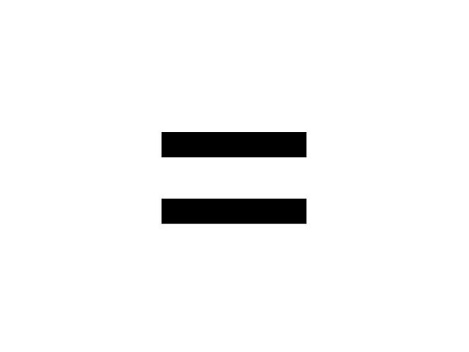


**Test je vytváraný na základe špecifikácie.**

**Šedá skrinka „Gray Box“**







**Unit testing Incremental integration testing**

**Integration testing Functional testing System testing End-to-end testing Sanity testing Regression testing Acceptance testing Load testing**

**Stress testing Performance testing Usability testing**

**Install/uninstall testing**

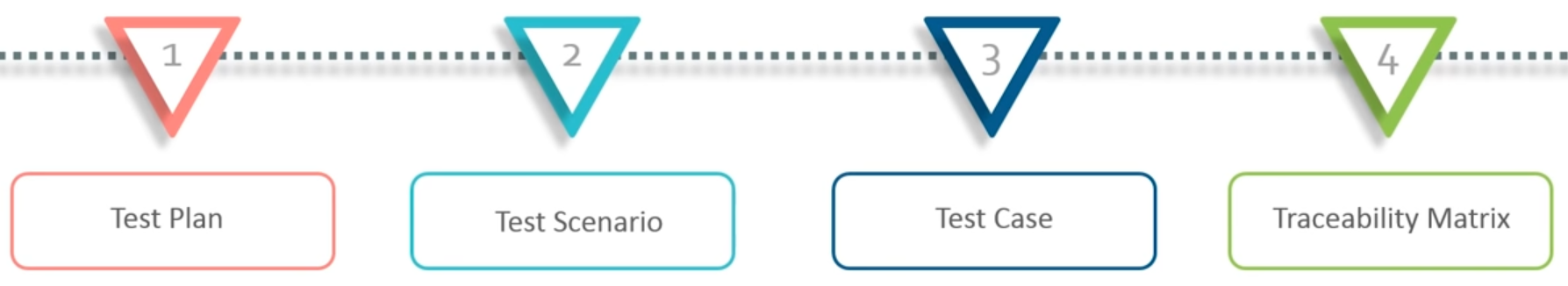
**Recovery testing Security testing Compatibility testing**

**Comparison testing**

**Alpha testing Beta testing**

**Hierarchia testov**

**Dokumentačné artefakty**



**Unit testing – jednotkové testovanie**

Testovanie izolovanej jednotky programu. Potláčame vplyv okolia

(volané metódy, súvisiace objekty, klientsky kód)

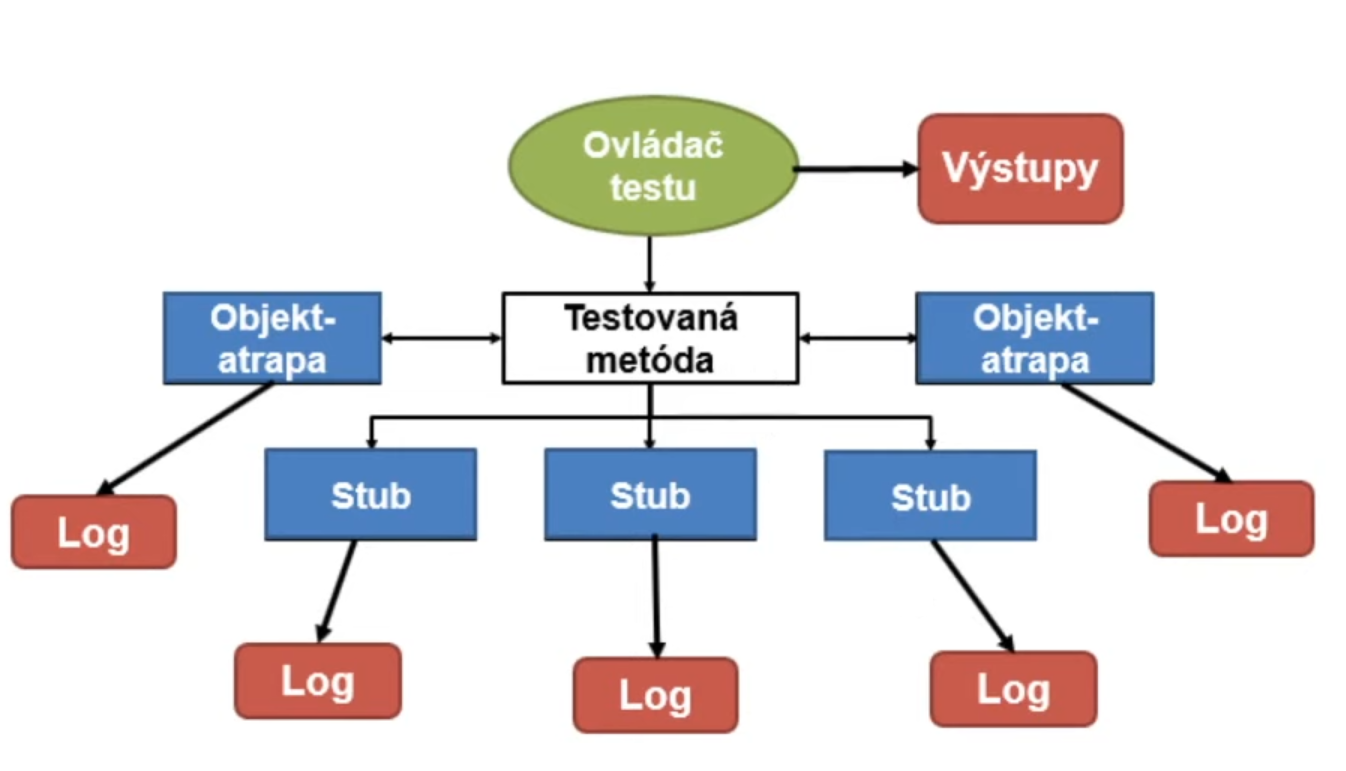
**Čo môže byť izolovaná jednotka?**

1. **Funkcia;**
2. **Modul;**
3. **Procedúra;**
4. **Trieda;**
5. **Metóda;**
6. **Komponent;**
7. **....**

**Kritériom je izolácia nie veľkosť jednotky;-)**

**Závislosti sa na účely testovania odstraňujú mocks a stubs**

**Jednotkový test**

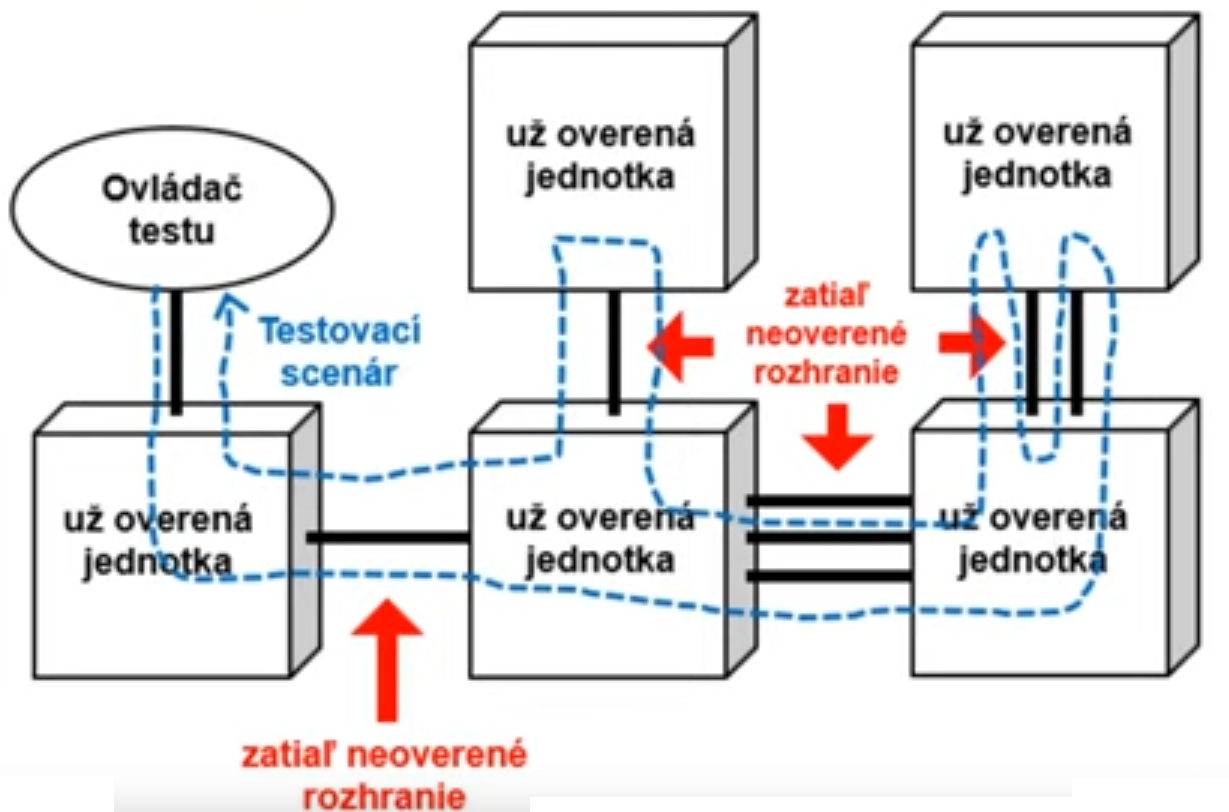


**Výhody**

* Nemusíme riešiť, čo sa deje inde
* Menej tesovacich scenárov
* Test priamo vyjadruje v ktorej jednotke je chyba

**Integration testing – integračné testovanie**

Technika systematického zostavovania softvérového systému počas vykonávania testov zameraných na odhalenie chýb spojených s rozhraniami. (spájanie súčiastok)



**System testing – systémové testovanie**

Komplexné testovanie vykonávané s cieľom validovať celý systém a jeho charakteristiky vychádzajúce z požiadaviek a návrhu.

* reálne podmienky
* nasadzovanie ako celok
* priamo overované požiadavky

Testovanie:

* Komunikácie (medzi pod-systémami)
* Modulov (pod systémov)
* Používateľských scenárov
* Použiteľnosti
* Bezpečnosti
* Hraničných hodnôt
* Spolupráca (kompatibilita)
* Výkonu
* Škálovateľnosti
* Stres test – maximálna záťaž
* Stability
* Regresov

**Acceptance testing – akceptačné testovanie**

Formálne testovanie vykonávané na určenie či systém zodpovedá akceptačným kritériám.

Vykonáva sa u zákazníka (nasadenie v produkčnom prostredí zákazníka)

Dodávateľ vykonáva školenie zákazníka

Porovnávajú sa akceptačné kritéria!!! – dôležitá komunikácia

1. Predpoklady testu
2. Scenár testu
3. Očakávaný stav

V-model

* systems development model
* používaný na definovanie uniformných systémových procedúr pri vývoji produktu
* grafická reprezentácia – systems development lifecycle
* German federal administration

VEE model

* V-model:
  + Je kompletný (Software Development,

Quality Assurance, Configuration Management

and Project Management)

* + Poskytuje konkrétnu podporu (inštrukcie, odporúčania)
* Je sofistikovaný
* Je vyvážený
* Podpora pri tendroch
* Verejne riadená aktualizácia
* Široké aplikačné spektrum

V-model

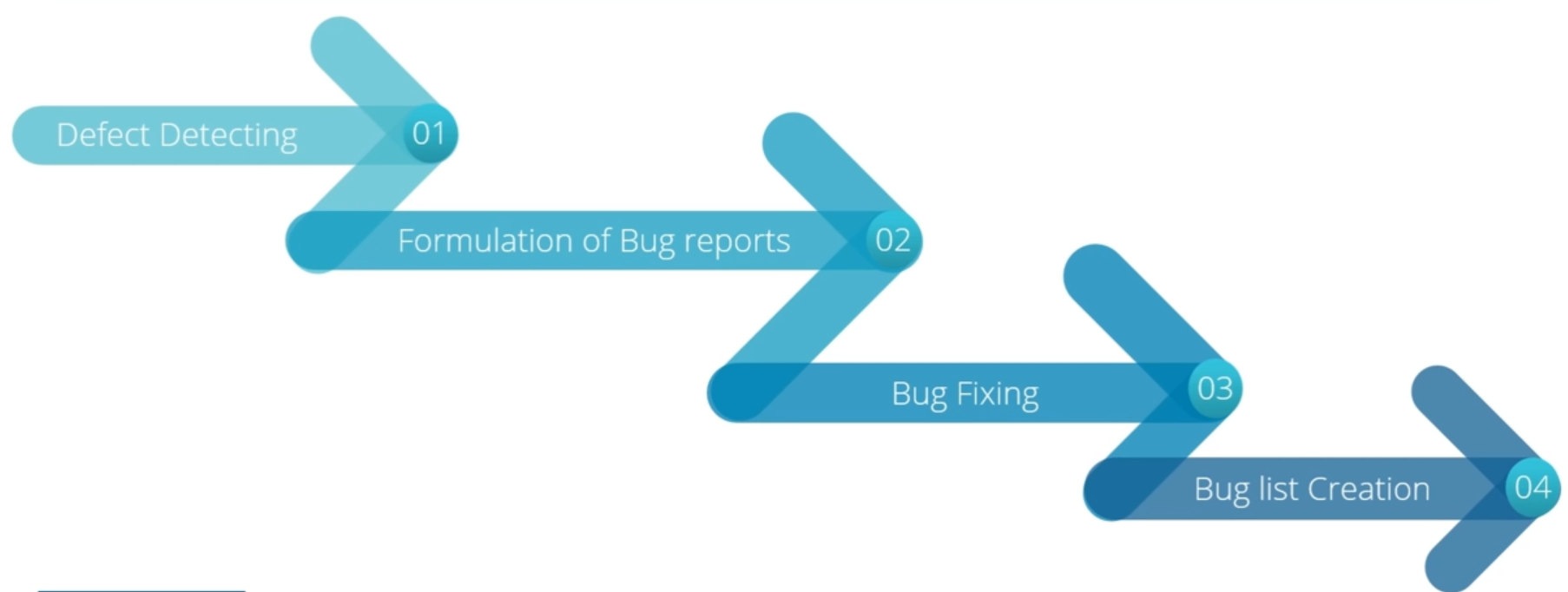
V-model testovania

X-model testovania

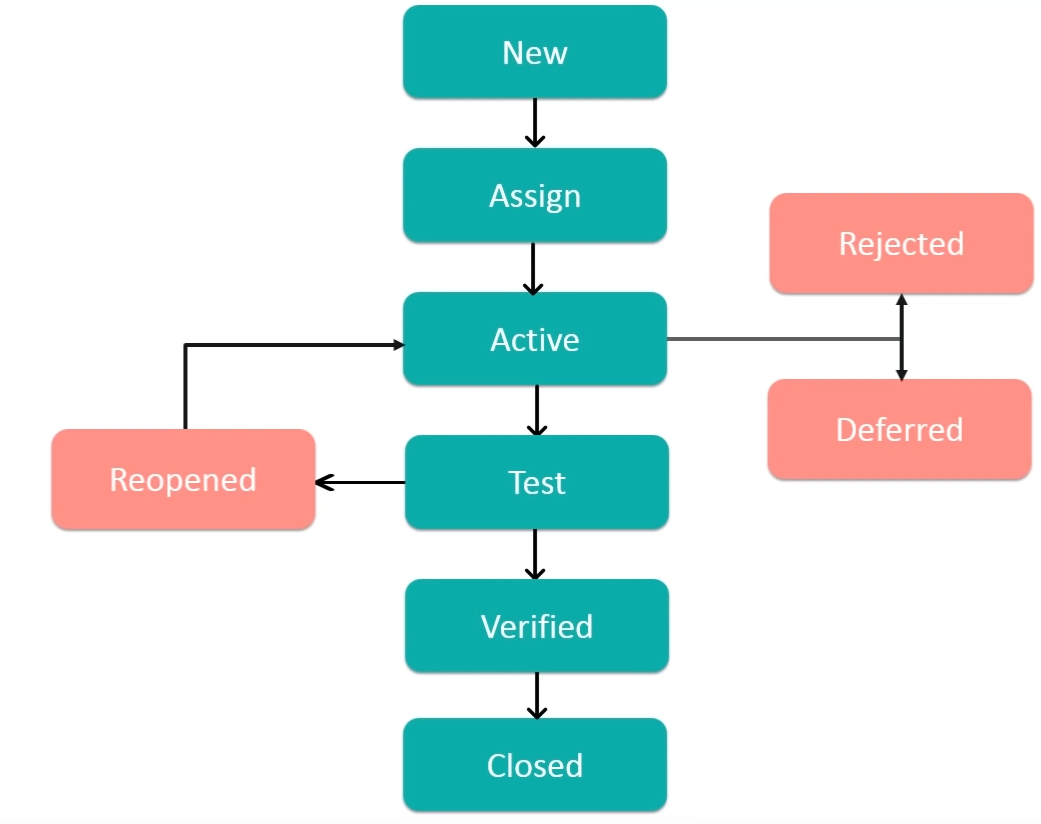
W-model testovania

**Defect Mangment process**

Možno ho definovať ako proces detekcie a odstránenia problému. Je potrebné povedať, že chyby vo vývoj SW vždy boli a budú.



Bug Life Cycle



Software Test Management,  
Test Documentation

* How to verify requirements?
* How to design tests for our project?
* What test design techniques to use?
* When to design and implement tests?
* When to execute tests?
* What to prepare for test execution?
* When to stop testing?
* Who to assign for testing? Test management?
* What to document during testing?
* How to document testing?
* ...

What is the purpose of testing?

* Finding defects
* Testing is a process
* Testing should be monitored and controlled
* Constraints – cost, schedule

ISO/IEC/IEEE 29119

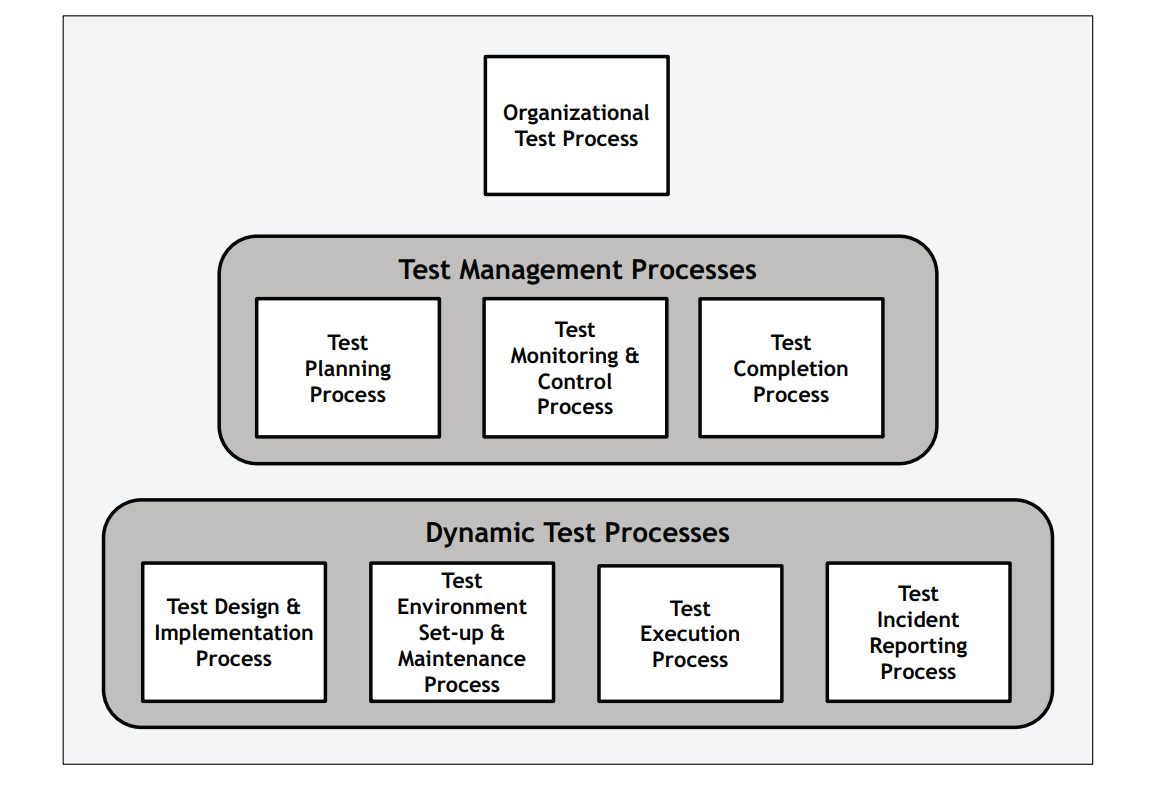
Covers the domain of software testing

1. Concepts and definitions
2. Test processes
3. Test documentation
4. Test techniques
5. Keyword-driven testing

Test Processes

* Dynamic test processes
  + Define test **sub-processes**
  + Sub-processes are created for different test **levels** (unit, integration, system, acceptance) or test **types** (functional, performance, usability, …)
  + A sub-process may address one or more levels/types
* Test management processes
  + Defines processes covering test management for the entire project or individual sub-processes
* Organizational test processes
  + Defines a process for creating/maintaining organizational test specifications – policies, strategies, …

Test Processes



Dynamic Test Processes

A screenshot of a cell phone

Description automatically generated

Test Design and Implementation

A picture containing text, screenshot

Description automatically generated

Test Design and Implementation

A picture containing text, screenshot

Description automatically generated

Test Design and Implementation

A picture containing text, screenshot

Description automatically generated

Test Design and Implementation

A picture containing text, screenshot

Description automatically generated

Common Items in Documents

* Unique identifier
  + Assigned to the entire document and individual items (e.g. test cases)
  + Automatic assignment of IDs recommended for a large volume of items
* Traceability
* Scope
  + Specifies what the document covers, assumptions, limitations
* References
  + to other related documents, sources, both inside and outside the organization – test plans, policies, …
* Glossary
* Change history

Test Documentation – Practices

Documents and their items may be combined for convenience.

Not all document items are required.

* ISO/IEC/IEEE 29119-3:2013, Annex B
  + *Shall* – required
  + *Should* – optional but recommended
  + *May* – optional

Test Design Specification

Identifies features to be tested

* Feature sets
  + Group of features to be tested, specified in the test plan
    - Hierarchically, by business processes, …
  + Feature sets may involve interaction with the entire system, not just software
* Test conditions
  + Specifies what can be tested
  + Format – tables, diagrams, …

Test Case Specification

Specifies individual cases to be tested

* Test coverage items
  + E.g. equivalence classes
* Test cases
  + Both valid and invalid inputs
  + Format depends on the test level or type
    - Scenarios
    - Tables of preconditions, inputs (actions) and expected outputs
    - …

Test Procedure Specification

Describes how test cases should be executed

* Execute test cases individually or in groups
* Execute in steps and examine the result of each step
* Test team – roles
* Post-execution wrap-up activities

Test Data

* Test data requirements
  + Describes properties of data for test execution
  + Artificial data, real data (without sensitive information), …
* Test data readiness report
  + Describes fulfillment of each test data requirement

Test Environment Set-Up and Maintenance

A screenshot of a cell phone

Description automatically generated

Test Environment Set-Up and Maintenance – Documents

* Test environment requirements
  + Required hardware, OS, configuration, constraints (e.g. security), testing tools
  + May simply reference other document items for more information
* Test environment readiness report
  + Describes fulfillment of each test environment requirement

Test Execution

A screenshot of a cell phone

Description automatically generated

Test Incident Reporting

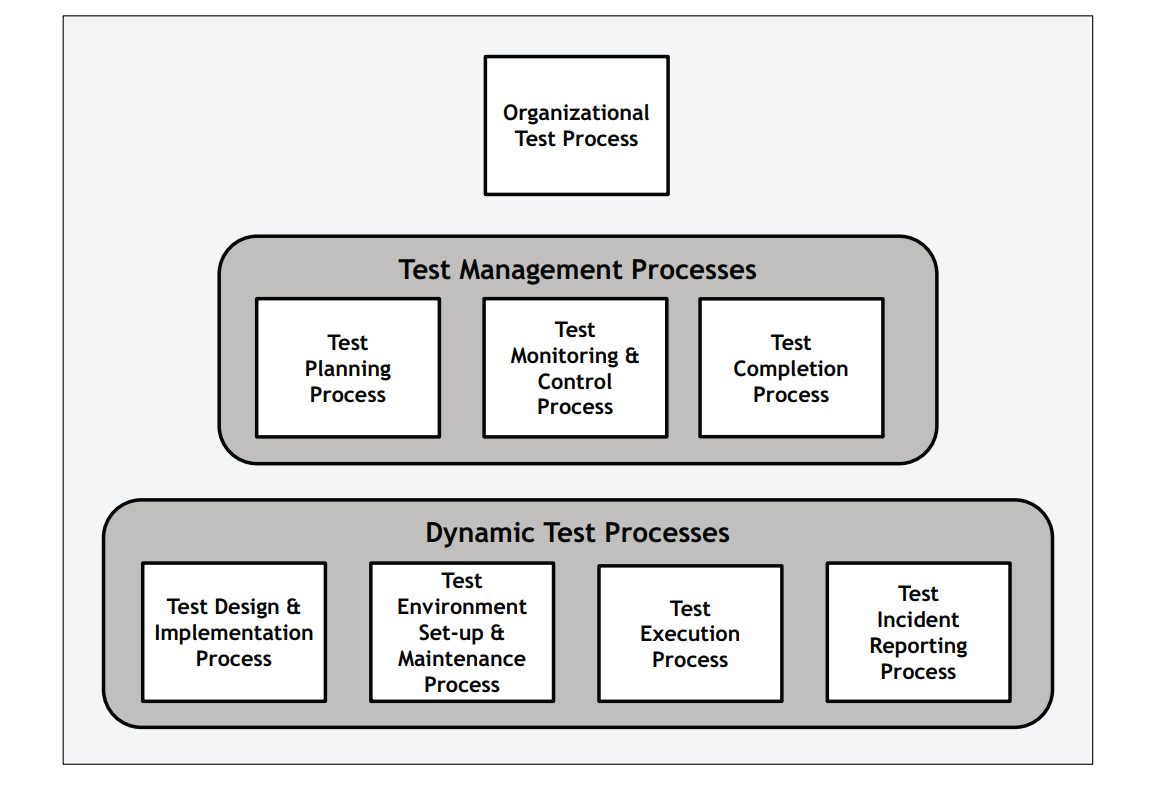
A screenshot of a cell phone

Description automatically generated

Test Execution and Reporting – Documents

* Test execution log
  + Records any useful information during test execution (e.g. that might affect further test executions)
* Test result
  + For each test case: pass/fail, approved/partially approved/fail
  + Fail – may be caused by anomalies
* Test incident reporting
  + Records any anomalies during test execution

Test Processes



Test Management Processes

A screenshot of a cell phone

Description automatically generated

Test Planning

A screenshot of a cell phone

Description automatically generated

Test Plan

* May range from a simple plan for one or more projects to detailed plans for individual test sub-processes
  + Project test plan, acceptance test plan, performance test plan, …
* Design Test Strategy (TP5)
  + Addresses both static testing (reviews) and dynamic testing

Specifies:

* Scope – features to be tested
* Test deliverables – test documents, reports
* Test design techniques to be used
* Risks
* Metrics to be collected
* Suspension and resumption criteria
* Testing activities and estimates
* Staffing
* Schedule
* Risks
  + Associated with the product, project, test sub-process itself, …
  + Common fields – probability, impact, prevention, mitigation
* Metrics
  + Number of passed test cases, failed test cases
  + Number of anomalies found per execution
  + Number of defects retested and closed
  + …
* Testing activities and estimates
  + Number of test cycles, when to stop and resume testing, …
  + May describe budget allocated to testing, cost estimates

Test Plan – Staffing (Roles in Testing)

* Tester
  + Develops tests
  + Test analyst, designer, executor, environment expert, …
* Test manager
  + Ensures conformance to the test management process, plans the dynamic test process
* Test strategist
  + Ensures conformance to the organizational test process

Test Monitoring and Control

A screenshot of a cell phone

Description automatically generated

Test Completion

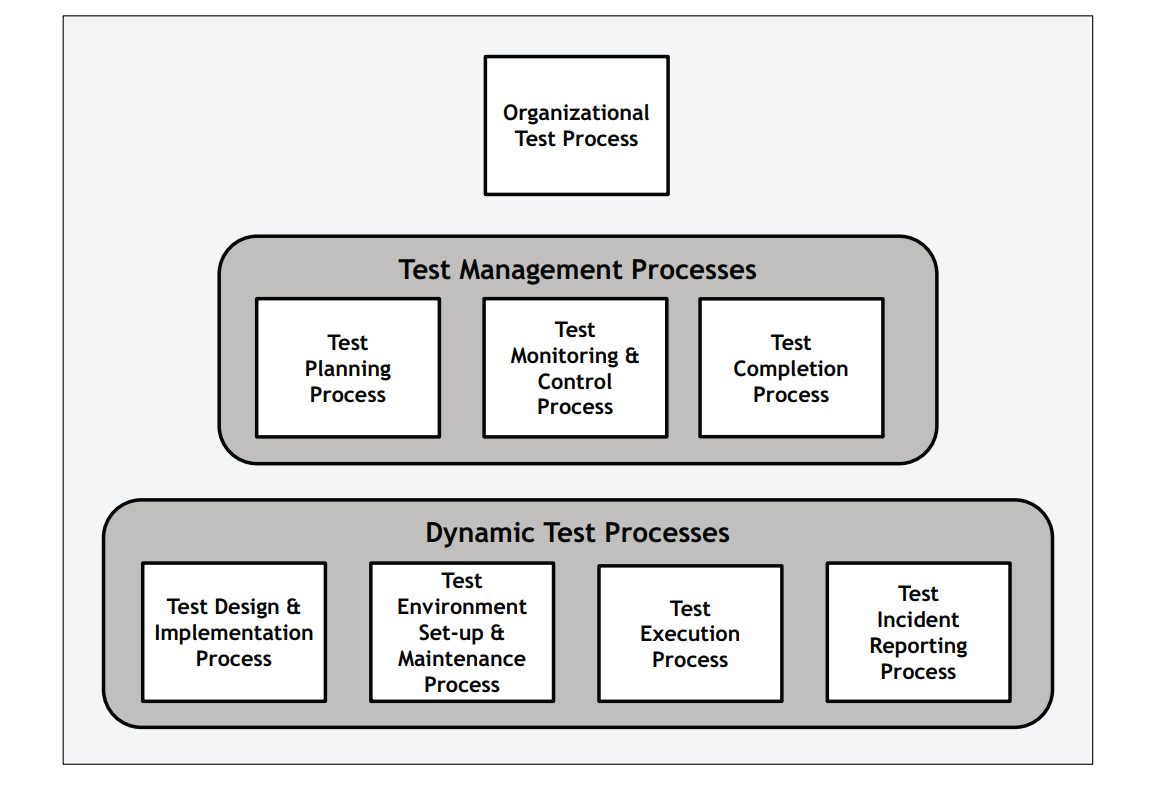
A screenshot of a cell phone

Description automatically generated

Test Monitoring, Control and Completion – Documents

* Test status report
  + Interim report on the tests under execution
  + Reports progress against the test plan
  + Test measures
  + Evaluation of tracked risks, identifying new risks
* Test completion report
  + Test measures
    - Number of test cases passed
    - Number of hours spent on test design, execution, …
  + Evaluation – customer accepted the product, …
  + Deviations from the test plan
  + Lessons learned

Test Processes



Organizational Test Process

A screenshot of a cell phone

Description automatically generated

Test Policy

* Defines what should be accomplished by testing
* Lays the groundwork for reviewing and improving the test policy
* Provides objectives of testing

Organizational Test Strategy

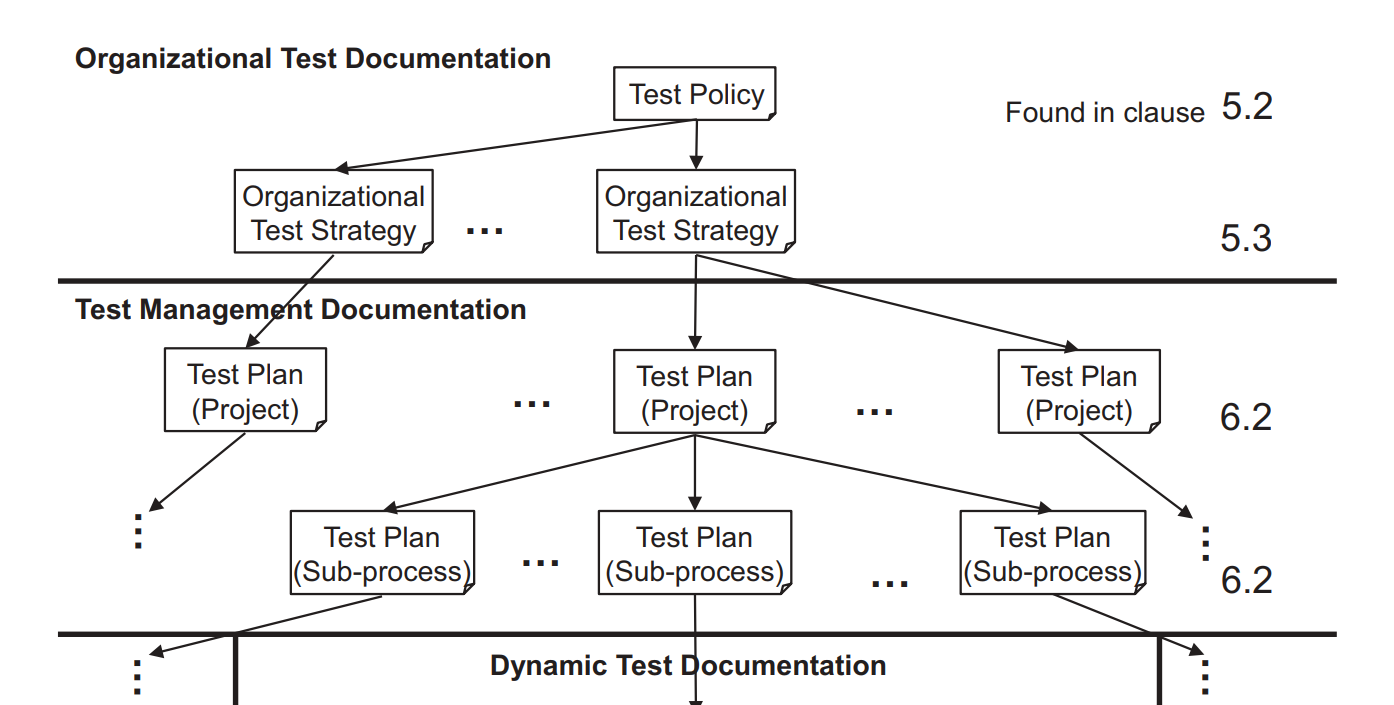
* “How to perform testing” in the entire organization
* How to achieve objectives specified in test policy
* Sections
  + Overall approach to testing
  + Degree of independence – test team from the development team, …
  + Risk management (generic, not yet tailored for testing)
  + Test organization structure – pool of testers, test lead, …
  + Test documentation strategy, requirements – what documents and items to include, conformation to standards
  + Test sub-processes
  + Test tools
  + Metrics to be collected

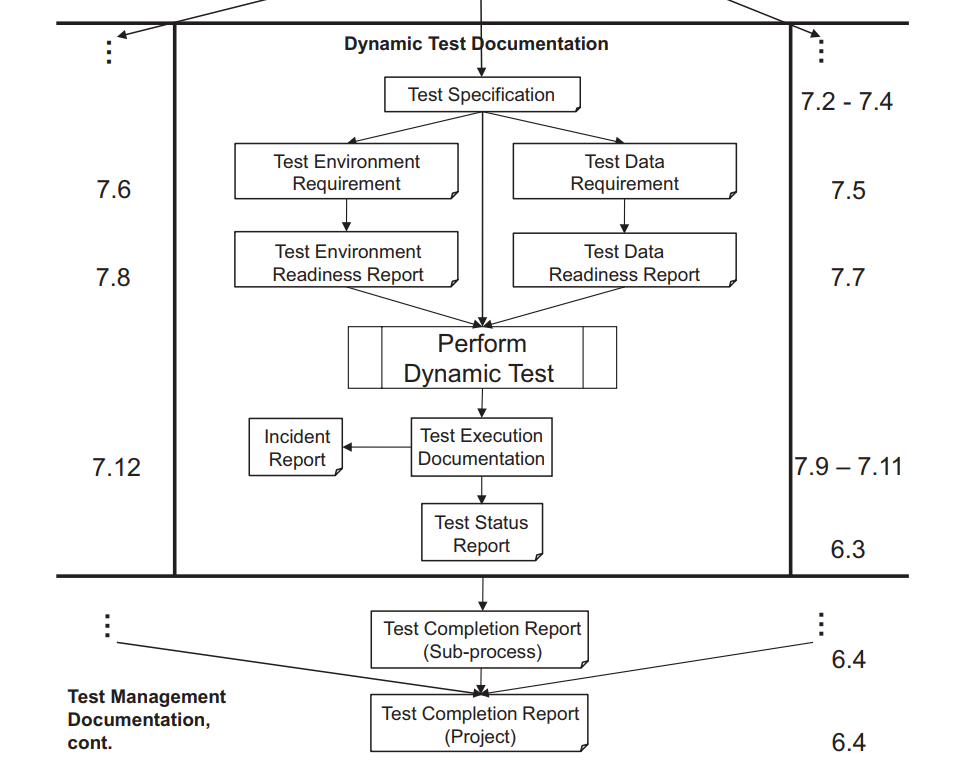
Organizational Test Process – Example Implementation

A screenshot of a cell phone

Description automatically generated

Test Documentation – Overview





Test Documentation – Overview

* Outline of documents:  
  ISO/IEC/IEEE 29119-3:2013, Annex A
* Examples of document contents:  
  ISO/IEC/IEEE 29119-3:2013, Annexes D to S

Test Management – Common Pitfalls

* Improper understanding of testing processes
  + No system is ever defect-free
  + Underestimation of required resources
* Inexperienced personnel
  + Testing of critical parts of a system
  + Outsourcing

Software Verification and Validation

Testing allows verifying/validating software or its components.

What defects can software testing detect/prevent?

What defects cannot software testing detect/prevent?

Verification and Validation (V&V) Methods

* Functional, non-functional testing – dynamic V&V, also referred to as dynamic testing
* Static review, verification, analysis – static V&V, static testing
* Formal methods

Verification

Definition from IEEE Std 1012-2016:

*The process of providing objective evidence that the system, software, or hardware and its associated products conform to requirements for all life cycle activities during each life cycle process.*

* *“conform to requirements*” – developing the product right

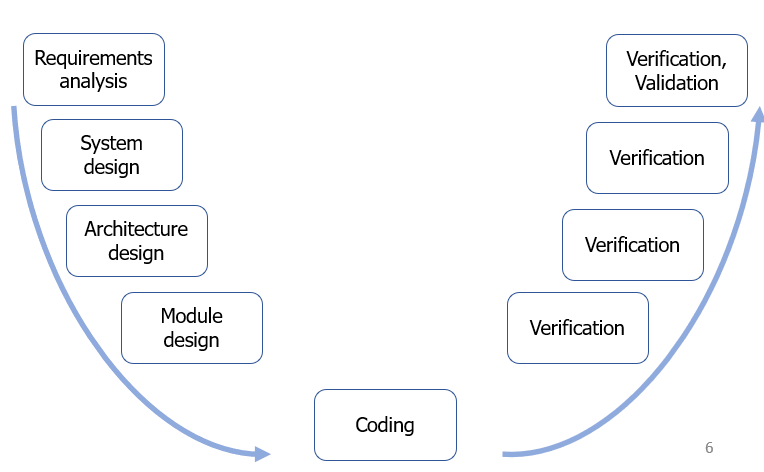
Validation

Definition from IEEE Std 1012-2016:

*The process of providing evidence that the system, software, or hardware and its associated products satisfy requirements allocated to it at the end of each life cycle activity, solve the right problem and satisfy intended use and user needs.*

* *“solve the right problem*” – e.g. correctly model physical laws, implement business rules, use the proper system assumptions, …
* *“satisfy intended use and user needs*” – developing the right product

Verification and Validation Methods



Static V&V Methods

* Code review
* Informal verification
* Walkthrough
* Technical review
* Inspection

Software artifacts under V&V

* Source code
* Software architecture, design
  + Models, diagrams, documents
* User guide
* API documentation
* Requirements specification
* Software test documentation
* Documentation of project processes
* …

Static Code Analysis

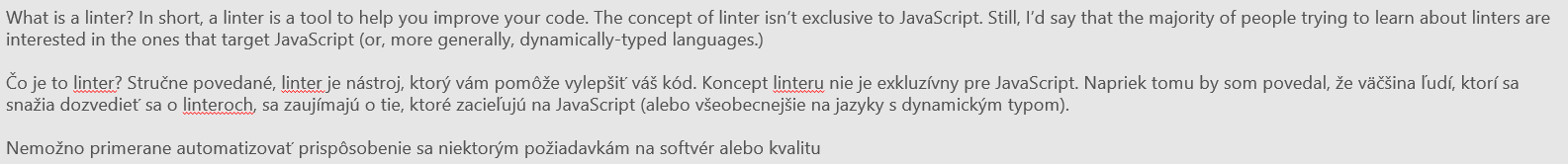
Provides automated support for determining:

* Compiler errors
* Potential program failures
  + E.g. uninitialized pointer in use
* Deviations from code style, coding conventions
* Deprecated API
* Software metrics
* ...

Static Code Analysis

* Tools – built-in analyzers in IDEs, third-party “linters”
* Static code analysis does not substitute reviews
  + Cannot reasonably automate conformation to some software or quality requirements

Linter



Review Methods – Common Objectives

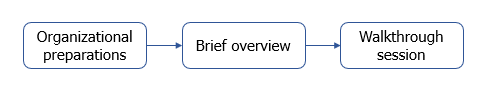
* Identify defects (errors) in artifacts
* Identify inconsistencies, deviations from templates, conventions
* Identify risks that may affect project completion
* Approve the artifact under review

Informal Verification

* Main purpose: detecting potential defects
* Possible additional purposes: generating new ideas or solutions, quickly solving minor problems
* Not based on a formal (documented) process
* May not involve a review meeting
* May be performed by a colleague of the author (buddy check) or by more people

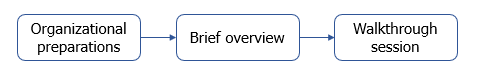
Walkthrough

* Involves “walking through” the artifact under review
  + E.g. source code – simulation (“mental execution”) of test cases
* Allows exchanging knowledge about the artifact and the methods and techniques used



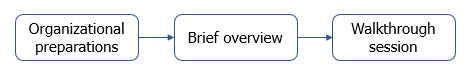
Walkthrough –  
Organizational Preparations

* Prepare a walkthrough team – 2-7 participants
* Prepare the artifact under review
* Prepare a presentation



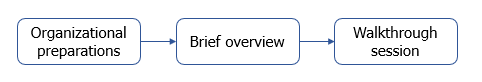
Walkthrough – Roles

* Walkthrough leader
  + Prepares, schedules and conducts walkthrough sessions
* Author
  + Author of the artifact
  + Presents the artifact
* Recorder
* Team member(s)
  + Identify and describe defects in the artifact during a session



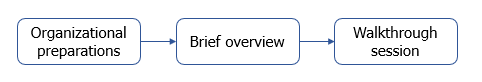
Walkthrough – Brief Overview

* Each team member **briefly** examines the artifact
* Each team member prepares a list of items (defects found)
* Walkthrough leader classifies the defects found for a more efficient walkthrough session



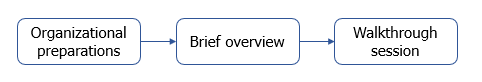
Walkthrough – Walkthrough Session

* Author presents the artifact
  + Order – sequential, data flow, control flow, … – agreed by the team at the beginning of the session
* Team members raise the prepared list of items and discuss additional defects found during the session
* Output:
  + Walkthrough session findings report



Walkthrough – Data Collection

* Optional
* Serves to improve the quality of the artifact, software development processes and the walkthrough process itself



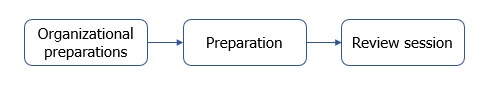
Technical Review

* Main objective: Determine conformation to specification or standards
* Allows reviewing larger volumes of artifacts than inspections or walkthroughs
* Allows examining alternatives to satisfying specification or standards



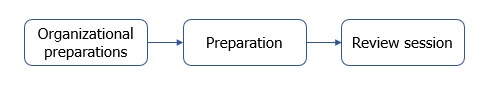
Technical Review – Organizational Preparations

* Prepare a review team – 3 or more participants
* Prepare the artifact under review
* Prepare a presentation



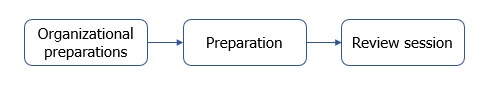
Technical Review – Roles

* Review leader
  + Prepares, schedules, conducts review sessions
  + Determines who shall present the artifact during sessions
* Decision maker
  + Determines if the review objectives have been met
* Recorder
* Technical reviewer(s)
  + Trained personnel evaluating the artifact



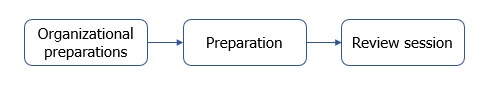
Technical Review – Preparation

* Each reviewer examines the artifact
* Each reviewer documents any defects found
* Defects are sent to the author



Technical Review – Review Session

* Identifies defects (anomalies) and change requests
* Determines if the artifact is ready for the next activity
* More than one session may be conducted for an artifact
* Output:
  + Session report – documents the meeting



Inspection

* More formal compared to technical reviews and walkthroughs
* Shown to be more efficient in finding defects



Additional objectives:

* Exchange knowledge about methods, techniques, artifact under review
* Lay the groundwork for improvements in software development processes
* Lay the groundwork for improving the inspection process itself (follow-ups)



Inspection –  
Organizational Preparations

* Prepare an inspection team – 3-6 participants
* Prepare the artifact under review
* Prepare a presentation

Inspection Team – Roles

* Inspection leader (moderator)
  + Guides the review process, selects reviewers, schedules meetings
* Author
  + of the artifact under review
* Reader (presenter)
  + A person other than the author presenting the artifact
  + Studies the artifact beforehand
  + Having another person understand the artifact allows disseminating knowledge in the team/organization
* Recorder
* Reviewer(s) (inspectors)
  + Specialized for the artifact under review
* Observers
  + Passive observers wanting to learn about the artifact

Inspection – Artifact Review

* Each reviewer **thoroughly** examines the artifact independently
* Each reviewer prepares a potential list of questions, changes and improvements



Inspection – Inspection Session

* Author gives an overview
* Presenter presents the artifact
  + Code, document – present line by line or blocks of text/lines
* Reviewers may make general suggestions



Inspection – Inspection Session

* Decision about the artifact:
  + Full approval
    - Accept with or without minor corrections
  + Partial approval
    - An inspection team member verifies corrections
  + Denial of approval
    - Artifact requires reinspection
* Outputs:
  + Inspection session report – decision, list of proposed changes
  + Inspection summary report – basis for follow-ups



Inspection –  
Corrections and Reworking

* Author takes corrective measures based on reviewers’ general suggestions (change requests)

Inspection – Follow-up

Review the inspection process, suggest improvements

* Add/modify checklists
  + Allowing reviewers to quickly find typical defects
* Track process metrics
  + Number of lines of code (LOC) reviewed per hour
  + Number of change requests generated per one KLOC (1000 LOC), per hour
  + Total number of hours spent on inspection
  + …
* Classify defects found, e.g. by severity

Review Methods – Summary



More detailed comparison in IEEE 1028-2008, Annex A

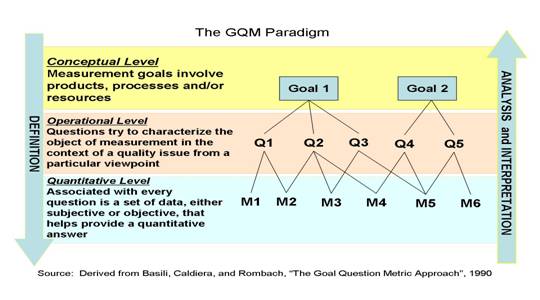
Static testing (reviews) and dynamic testing focus on defect detection and removal.

Understandably, defect prevention can reduce the number of defects inserted during software development processes.

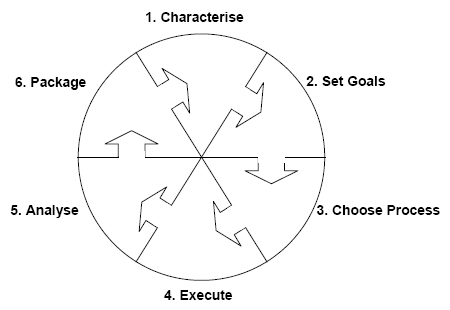
Goal-Question-Metric

„GQM je prístup zhora-nadol na vytvorenie meracieho systému založeného na cieľoch pre vývoj softvéru tak, že tím začne s organizačnými cieľmi, definuje ciele merania, kladie otázky na ciele a identifikuje metriky, ktoré poskytnú odpovede na otázky.“

Úrovne GQM



GQM Basili (1/2)



GQM Basili (2/2)

Kroky:

* Charakterizovanie projektu
* Definícia cieľov
* Výber procesu
* Vykonanie merania
* Analýza dát
* Spracovanie a uloženie výsledkov

GQM Berghout, Solingen

Základné fázy metódy GQM:

* Plánovanie
* Definícia
* Zbieranie dát
* Interpretácia

Fáza: Plánovanie

* Zostavenie GQM tímu
* Výber oblasti zlepšenia
* Výber projektu a zostavenie projektového tímu
* Vytvorenie projektového plánu
* Školenie a prezentácie

Fáza: Definovanie

* Je definovaný merací program
* Sú vypracované 3 dokumenty
  + GQM plán
  + plán merania
  + plán analýzy
* Fáza má definovaných 11 činností

Fáza: Definovanie – činnosti (1/3)

* Definovanie cieľov merania
  + odvodené z vylepšených cieľov vo fáze plánovania

a opísané v projektovom pláne

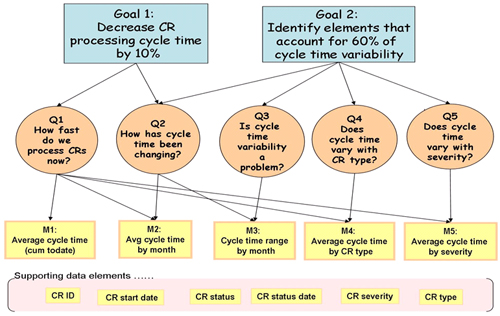
* + má svoju vlastnú štruktúru (šablóny)
* Revízia alebo vytvorenie procesných softvérových modelov
* Vykonanie GQM rozhovorov
  + pre podporu komunikácie medzi GQM tímom a projektovým tímom
  + používajú sa tzv. abstrakčné listy (angl. Abstraction sheets)
  + listy sa delia na štyri sekcie: kvalitatívne hľadisko, základné hypotézy, premenlivé faktory a dopad na základné hypotézy
* Definícia otázok a hypotéz

Fáza: Definovanie – činnosti (2/3)

* Zhodnotenie otázok a hypotéz
  + otázky sú základným spojivom medzi cieľmi a metrikami
  + dôležitá rola nielen počas fázy definície, ale aj počas fázy interpretácie
* Definícia metrík
  + metriky by mali byť definované tak, aby poskytli všetky kvantitatívne informácie k odpovedi na otázku
  + mali by sme mať k dispozícii dostatočné informácie na zodpovedanie otázok
* Kontrola konzistencie a kompletnosti metrík
  + pre zabezpečenie musí byť vykonávaná kontrola počas celej definičnej fázy
  + ak sa počas kontroly nájdu chýbajúce, neúplné alebo nezlučiteľné definície, prispôsobia sa definície modelu postupu alebo naopak

Fáza: Definovanie – činnosti (3/3)

* Vytvorenie GQM plánu
  + obsahuje ciele, otázky a metriky pre program merania
  + slúži ako príručka pre interpretáciu dát a poskytuje základ pre následný vývoj plánu merania a plán analýz
* Vytvorenie plánu merania
  + poskytuje formálne definície priameho merania
  + identifikuje člena, ktorý zbiera jednotlivé merania
  + definuje, ktoré médium (nástroj, forma) má člen použiť k priamemu meraniu
* Vytvorenie plánu analýz
  + dokument, ktorý simuluje interpretáciu dát podľa GQM plánu
  + hlavným účelom plánu analýz je opísať, ako významná je meraná informácia
  + informácia je zapracovaná takým spôsobom, že môže byť ľahko interpretovaná projektovým tímom
* Zhodnotenie plánov



Fáza: Zbieranie dát (1/2)

* Cieľ: vykonať merania v súlade s GQM plánom a plánom merania
* Fáza pozostáva z:
  + úvodných činností pre zberanie dát
  + úvodnej sekcie (angl. kick-off)
  + skúšobnej doby (angl. trial period)
  + trénovania (angl. training)

Fáza: Zbieranie dát (2/2)

* Činnosti pre zbieranie dát
  + definujú všetky hľadiská, ktoré sú nevyhnutné na uskutočnenie úloh pre zbieranie dát
  + forma činností (manuálna, elektronická)
* Úvodná sekcia
  + získať súhlas projektového tímu
  + ešte raz vysvetliť ciele merania programu
  + monitorovanie GQM tímu projektovým tímom
* Skúšobná doba
  + testovanie definovaných dát, činností, nástrojov a foriem
  + vykonáva iba jeden alebo dvaja členovia projektu

Fáza: Interpretácia

* Pokus o nájdenie odpovedí na otázky
* Bez tejto fázy je celé úsilie zbytočné
* Výsledky sú prediskutované na porade (angl. feedback session)

Porada (1/3)

* Príprava porady
  + zozbierané dáta transformovať do podoby na prezentovanie
  + materiál musí byť v podobe, že je možné určiť, či sa ciele dosiahli
  + poradu zabezpečuje GQM tím s príp. zainteresovaním projektového tímu

Porada (2/3)

* Porada
  + 1.5 – 2 hod. (max 3. hod.)
  + 15 – 20 „slidov“ (obrázky, grafy)
  + úloha GQM tímu: motivovať projektový tím
  + fázy
    - otvorenie porady
    - prehľad GQM
    - predmet merania
    - prezentovanie materiálov a odpovedanie na otázky
    - vyhodnotenie

Porada (3/3)

* Oboznámenie s interpretáciou nameraných dát
  + GQM tím napíše správu a rozdistribuuje ju
  + pri posielaní správy manažmentu hrozí zlá interpretácia
  + analýza nákladov/výhod
  + + (zvýšený odbyt výrobku, lepšie pochopenie manažmentu zdrojov)
  + - (čas strávený zbieraním dát a vypĺňaním formulárov, poradami a stretnutiami)

Príklad (1/5)

* Merací program zameraný na zistenie vplyvu prerušení pracovníka pri práci
* Negatívny dopad
* Problémy s plánovaním projektu
* Stres pracovníkov
* Problémy s kvalitou výrobku
* Odloženie dodania produktu

Príklad (2/5)

* Plánovanie
  + GQM už existoval
  + určenie, čím sa bude program zaoberať (prerušeniami)
  + školenie projektového tímu GQM tímom
  + určenie cieľov
    - lepšie pochopenie vplyvu prerušení na pracovníka

Príklad (3/5)

* Definícia
  + vytvorenie skupiniek (2-3 osoby)
  + definovanie otázok, metrík a hypotéz
  + osobitné stretnutie s projektovým manažérom
  + otázky súvisiace s cieľom
    - Aké faktory ovplyvňujú vyžadované úsilie pri riešení prerušenia?
    - Je možné zabrániť prerušeniu?
    - Aký vplyv má prerušenie na prácu, ktorá bola prerušená?
    - Koľko nákladov si vyžadujú prerušenia?
  + definovanie hypotéz a ich ohodnotenie (1-10)
* Zbieranie dát

Príklad (4/5)

* Interpretácia
  + zistenia
    - dvojnásobne väčší počet prerušení ako predpokladaný
    - príliš časté osobné návštevy (55%)
    - vynaloženie úsilia
      * osobné návštevy – 13 minút
      * telefónny rozhovor – 10 minút
      * e-mail – 8 minút
  + opatrenia
    - zníženie osobných návštev
    - preferovanie e-mailu
    - uvedomenie si, že prerušenie je záťaž

Príklad (5/5)

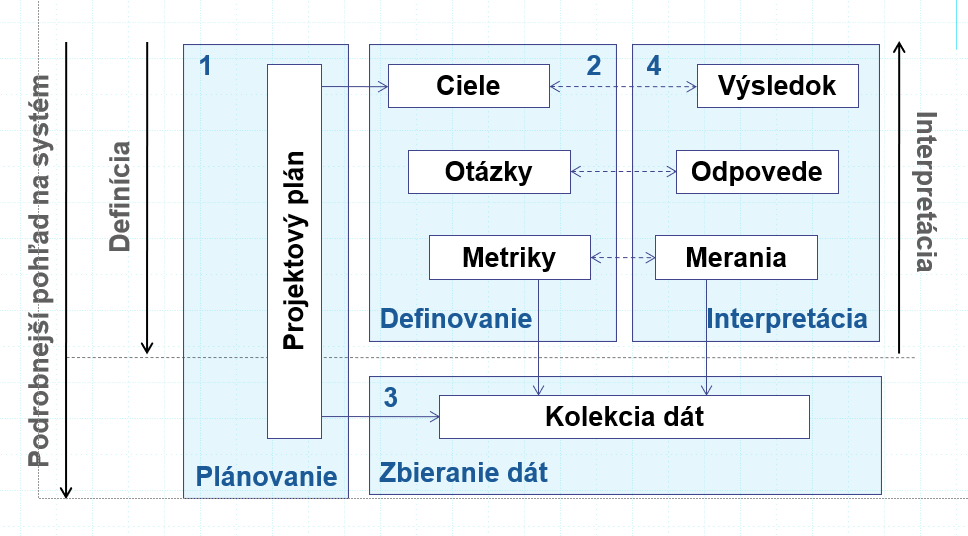
* Záver meracieho programu
  + Ak je prerušení viac ako 10 za deň, čas je neefektívne využitý.
  + Problém s projektovým plánom (zväčšiť čas na projekt, zmenšiť počet prerušení).
  + Prerušenia počas stretnutí a porád by mali byť minimálne.

Úrovne, na ktorých GQM prebieha



Fázy GQM

* Na zodpovedajúcich úrovniach je potrebné výsledky interpretovať:



Aplikácia postupu v praxi

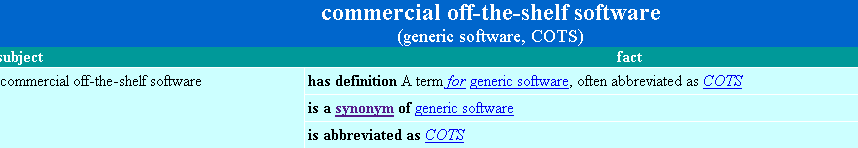


Zhrnutie

* Metóda GQM odporúča pri vývoji špecifikovať ciele, stanoviť zodpovedajúce otázky a definovať metriky, ktoré nám na určené otázky odpovedia.
* Objektívne a subjektívne merania sa používajú pri cieľoch najčastejšie definovaných ako produkty, procesy alebo zdroje.
* Postupuje sa určením biznis cieľov, ktoré sa konvertujú na merateľné ciele, definujú sa otázky, ktoré s týmito cieľmi súvisia, použitím vhodných metrík sa získajú dáta, ktoré na definované otázky odpovedajú a na základe výsledkov sa aktualizuje projektový plán. Výsledky sa prezentujú a vyhodnotia.
* Výhodou je zlepšený prehľad o systéme, podrobná kvantifikácia zmien

v projekte a určenie nekvantifikovateľných požiadaviek, odhalenie možností a obmedzení systému a docielenie dobrej porovnateľnosti výsledkov.

COTS – Commercial Off The Shelf



COTS (Commercial Off-The-Shelf) softvér je systém vybudovaný použitím viacerých iných hotových COTS komponentov. Vývojár je v tomto prípade integrátorom rozdielnych technológií od nezávislých dodávateľov.

COTS produkty predstavujú alternatívu ku komponentom vyvíjaným priamo vo firme. Majú potenciál urýchliť vývoj softvéru, znížiť námahu a zvýšiť kvalitu. Produkty sú dané k dispozícii, a tým pádom nie je potreba vynakladať úsilie na vývoj ich funkcií. Toto môže spôsobiť zlacnenie procesu, nakoľko modifikácia COTS produktu je menej nákladná ako vývoj produktu s rovnakou funkcionalitou od základu.

COTS softvér sa podobá na vývoj softvéru so znovupoužitím komponentov, ale nie je to úplne to isté. Oba modely síce používajú komponenty na zaručenie znovupoužiteľnosti a flexibility, ale v prípade COTS nemusíme komponenty vytvárať sami. Namiesto toho potrebné komponenty jednoducho nakúpime od ďalších dodávateľov.

Na COTS softvér sa viažu tieto charakteristiky:

1. COTS softvér nie je modifikovaný, ale je použitý v takej forme ako je získaný od dodávateľa. Toto je hlavná črta COTS softvéru. Vývojár jednoducho zakúpi hotový produkt a integruje ho do svojho systému bez toho, aby ho nejakým spôsobom pozmeňoval. Výnimkou sú iba minimálne úpravy nutné na zlepenie komponentov do výsledného produktu.
2. Ku zdrojovým textom a vývojovej dokumentácii COTS komponentov je iba minimálny prístup.
3. Vývojár nemá kontrolu nad životným cyklom jednotlivých COTS komponentov. Táto zodpovednosť je práve na dodávateľovi, ktorý musí zabezpečovať budúci vývoj a udržiavanie jeho softvéru.

V súčasnosti je väčšina softvérových produktov zložitá zmes

viacerých systémov, práve preto potrebujeme použiť COTS prístup

pri tvorbe veľkých produktov.

**Softvér môžeme rozdeliť na dva typy:**

1. COTS silné – v takýchto systémoch neexistuje žiadny hlavný komponent, ktorý by tvoril základ systému. Naopak systém je zložený z mnohých komponentov.
2. COTS riešenie – je založené na jednom hlavnom COTS komponente, ostatné komponenty iba rozširujú alebo upravujú jeho funkcionalitu.

Definícia COTS

* Znovupoužitie na úrovni samostatných systémov
* Systémy s definovaným rozhraním

a funkcionalitou

* Vytváranie aplikácií integráciou

a využívaním COTS

Charakteristiky COTS

* Pôvod – vplýva na cenu

a modifikovateľnosť

* Modifikovateľnosť
* Cena
* Funkcionalita
* Distribuovateľnosť

Využitie COTS



* Využitie produktov
* Integrácia s dominantným COTS
* Integrácia viacerých rovnocenných COTS

Výhody používania COTS

* Zvýšenie spoľahlivosti
* Redukcia procesných rizík
* Zrýchlenie vývoja
* Efektívne využitie ľudských zdrojov
* Zachovanie kompatibility, štandardov

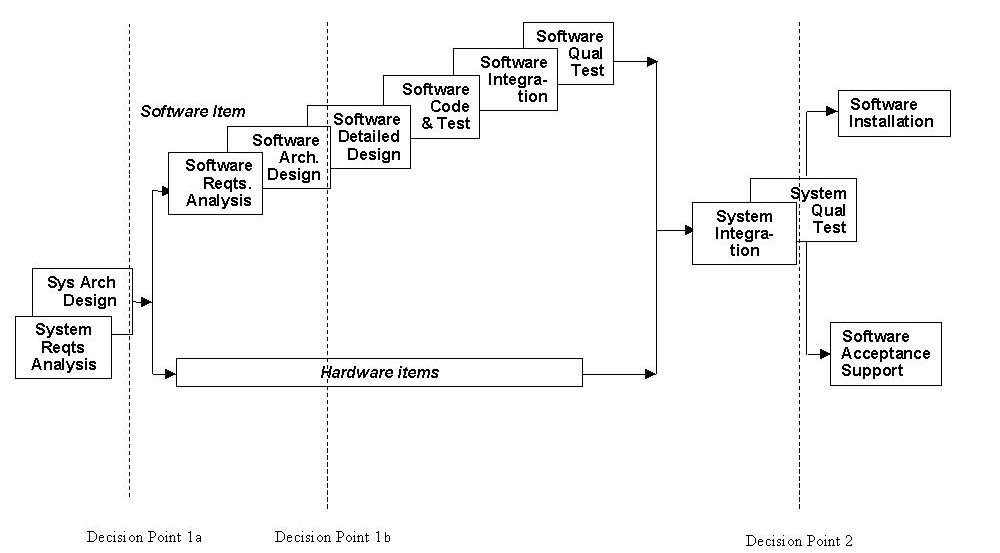
Nevýhody použitia COTS

* Nedostatok kontroly na produktom
* Integrácia s COTS
* Závislosť od dodávateľa
* Oboznámenie sa s produktom

Ako vytvoriť kvalitný softvér ...

* … pri použití COTS komponentov?
* Kvalita produktu je závislá od kvality COTS komponentov
* Potreba zakomponovania merania kvality komponentov do procesu vývoja softvéru
* Zabezpečenie kvality:
  + výber
  + vyhodnotenie
  + kvalifikácia

Proces vývoja softvéru s COTS



Životný cyklus

* tzv. „decision point“
  + zozbierané požiadavky na systém sú rozpracované do jednotlivých úloh a delegované do ďalších etáp
* Tradičný spôsob vývoja softvéru:
  + „decision point“ je identifikovaný po fáze architektonického návrhu systému
* Vývoj pomocou COTS:
  + potreba súbežného vykonania viacerých fáz naraz
  + úvahy o architektonických komponentoch medzi skupinami viacerých odborníkov
  + rozhodnutie, či komponent vyrobiť alebo kúpiť
  + „decision point“ je realizovaný po fáze architektonického návrhu softvéru

Rozdiely oproti tradičnému procesu

* Pridané aktivity
  + vyhodnocovanie produktov, zoznamovanie sa

s produktmi, interakcia s výrobcom (COTS)

* Obmedzené aktivity
  + kódovanie, odstraňovanie chýb, tzv. „unit“ testovanie, prehliadky kódu
* Modifikované aktivity
  + návrh systému zameraný viac na precízne skladanie existujúcich súčastí ako na prácu na interných moduloch
  + dôraz na architektúru systému – potreba kompatibility, konfigurovateľnosti a integrovateľnosti

Výber, vyhodnotenie a kvalifikácia COTS

* Identifikácia rolí a zodpovedností
* Vstupné kritériá
* Vstupy procesu
* Úlohy v procese
* Výstupy procesu
* Výstupné kritériá
* Metriky

Roly a zodpovednosti

* Projektový manažér
  + celková zodpovednosť
* Produktoví inžinieri (a.k.a. vedúci tímov)
  + zodpovednosť za definovanie relevantných kritérií
* Testovacia skupina
  + testovanie voči kvalifikačným kritériám
* Tím manažmentu rizík
  + identifikácia a odstraňovanie rizík v celom procese
* Vedúci manažmentu konfigurácií
  + kontrola konfigurácií v celom procese
* Manažér kvality
  + vyhodnocovanie aktivít súvisiacich s procesom voči projektovému plánu

Vstupy procesu

* Dokumentované systémové/softvérové požiadavky
* Dokumentovaná systémová/softvérová architektúra
* Definované ohraničenia jednotlivých úloh
  + časový plán
  + personál
  + ...

Proces výberu COTS

* Proces pozostáva zo siedmych krokov
* Plánovanie
  + zaznamenanie udalostí, plánovanie

a odhady

* + vytvorenie tímu pre manažment rizík
  + vyhodnotenie požiadaviek na komponenty

Metodológia hodnotenia

* Vytvorenie vyhodnocovacej metodológie
  + určenie výberových kritérií
  + stanovenie váhy pre kritériá
  + určenie hodnotiaceho algoritmu
  + definovanie prahových hodnôt pre záverečné filtrovanie kandidátov
  + pre každý komponent architektonického návrhu

Vyhľadanie a ohodnocovanie

* Vyhľadávanie
  + v COTS/GOTS doméne
  + znovupoužiteľné knižnice
  + zhromažďovanie dát
* Ohodnocovanie
  + roztriedenie zhromaždených dát
  + vytvorenie súčtového hodnotenia pre každého kandidáta
  + akceptačné kritériá

Analýza a porovnanie kandidátov

* Aplikácia akceptačných prahových hodnôt
* Cenový odhad, veľkosť spojovacieho kódu
* Vytvorenie stratégie na zmiernenie rizík pre každého kandidáta
  + založená na výsledkoch skúšobných testov, referenciách, stratách výpočtových zdrojov, cene licencií a aktualizácií

Obstarávanie a kvalifikácia

* Obstarávanie
  + určenie, či daný komponent arch. návrhu bude zakúpený (COTS) alebo vytvorený
  + vytvorenie záložného plánu pre prípad zlyhania COTS produktu
* Kvalifikácia
  + umiestnenie COTS do systému pre správu verzií
  + vytvorenie a vykonanie akceptačných testov
  + v prípade potreby prechod na záložný plán

Vstupné kritériá a metriky

* Ukončenie procesu
  + úspešná kvalifikácia COTS produktov
  + umiestnenie COTS do systému pre správu verzií
  + správa o výbere, metriky a odporúčania na zlepšenie boli zaznamenané
* Metriky
  + dáta z plánovania, vynaložené úsilie a cena
  + získané hodnoty z plánovacieho systému

Výberové kritériá pre produkt

* Výkon (priepustnosť; pamäťové, ... nároky)
* Transparentnosť
* Bezpečnosť
* Aktualizačný cyklus
* Kompatibilita revízií
* Udržiavateľnosť
* Interoperabilita
* ...

Kritériá pre dodávateľa

* Reputácia
* Technická podpora
* Ochota rokovať o zmenách
* Školenia
* Referencie od komerčných zákazníkov
* ...

Štandardy pre management kvality softvéru

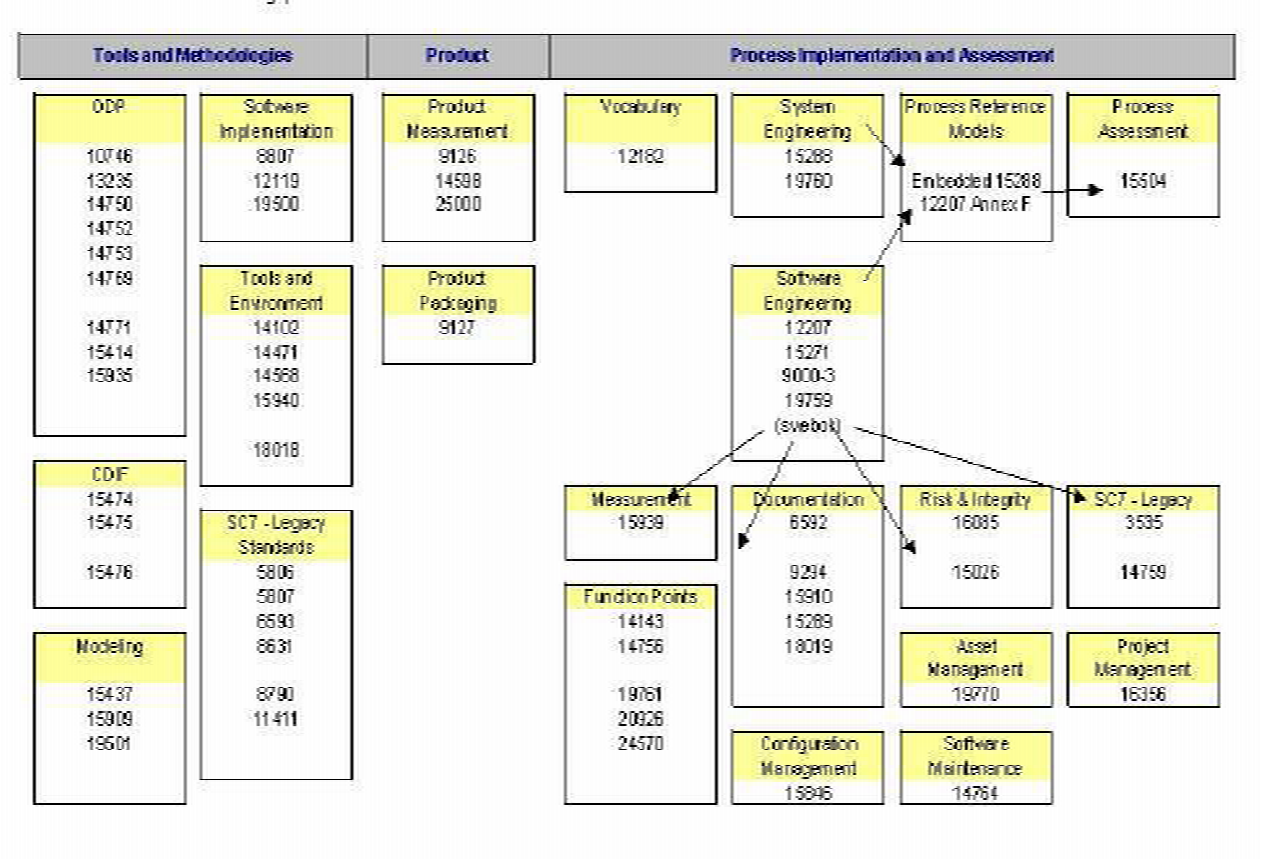
Štandard ISO 9126, ISO 2502x

* rastúci dôraz na kvalitu
* náročnosť porovnania kvality dvoch produktov
* štandardy, ISO
* ISO 9126 pre kvalitu softvérových výrobkov

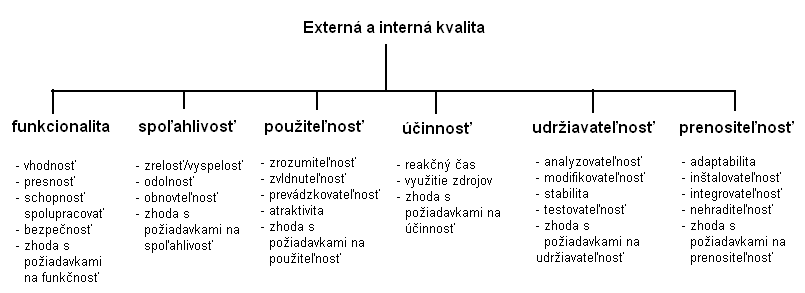
ISO 9126

* prvé vydanie 1991, revidovaný 2000
* členenie:
  + ISO 9126-1:2001 – Model kvality
  + ISO 9126-2:2000 – Externé metriky
  + ISO 9126-3:2003 – Interné metriky
  + ISO 9126-4:2004 – Použitie metrík akosti
* postavenie v hierarchii ISO štandardov ->

Postavenie v hierarchii ISO



ISO 9126-1:2001 Model kvality



Funkčnosť

* Je spôsobilosť softvérového produktu poskytovať funkcie zhodné s definovanými a vyplývajúcimi požiadavkami, pokiaľ je softvérový produkt používaný v definovanom prostredí.
* Subcharakteristiky:
  + **vhodnosť** – Dokáže softvér vykonávať požadované úlohy?
  + **presnosť** – Je výsledok taký, aký sme predpokladali?
  + **schopnosť spolupracovať** – Dokáže systém spolupracovať s inými systémami?
  + **bezpečnosť** – Zabraňuje softvér neautorizovaným prístupom?
  + **zhoda s požiadavkami na funkčnosť**

Spoľahlivosť

* Je spôsobilosť softvérového produktu udržiavať určitú hladinu výkonnosti softvérového produktu pri dodržaní definovaných prevádzkových podmienok.
* Subcharakteristiky:
  + **zrelosť/vyspelosť** – Bola väčšina chýb softvéru už eliminovaná?
  + **odolnosť** – Dokáže sa softvér vysporiadať s chybami?
  + **obnoviteľnosť** – Je softvér schopný pokračovať v činnosti a obnoviť dáta po výskyte chyby?
  + **zhoda s požiadavkami na spôsobilosť**

Použiteľnosť

* Je spôsobilosť softvérového produktu byť pochopiteľný, zvládnuteľný, použiteľný a atraktívny pri dodržaní definovaných prevádzkových podmienok.
* Subcharakteristiky:
  + **zrozumiteľnosť** – Zvláda používateľ ovládať systém jednoducho?
  + **zvládnuteľnosť** – Dokáže sa používateľ ľahko naučiť so systémom pracovať?
  + **prevádzkovateľnosť** – Dokáže používateľ pracovať so systémom bez zbytočnej námahy?
  + **atraktivita** – Vyzerá používateľské rozhranie zaujímavo?
  + **zhoda s požiadavkami na použiteľnosť**

Účinnosť

* Spôsobilosť softvérového produktu poskytnúť používateľovi adekvátny výkon vzhľadom na množstvo vynaložených zdrojov pri dodržaní definovaných prevádzkových podmienok.
* Subcharakteristiky:
  + **reakčný** **čas** – Aká je odozva systému?
  + **využitie** **zdrojov** – Využíva systém zdroje efektívne?
  + **zhoda s požiadavkami na účinnosť**

Udržiavateľnosť

* Je spôsobilosť softvérového produktu byť modifikovaný. Zmeny môžu zahrňovať opravy, zlepšenia a adaptáciu softvérového produktu na zmeny v prostredí, požiadavkách alebo funkčných špecifikáciách.
* Subcharakteristiky:
  + **analyzovateľnosť** – Dajú sa chyby ľahko diagnostikovať?
  + **modifikovateľnosť** – Je systém ľahko modifikovateľný?
  + **stabilita** – Pokračuje softvér v práci aj v prípade vykonaných zmien?
  + **testovateľnosť** – Dá sa softvér ľahko testovať?
  + **zhoda s požiadavkami na udržiavateľnosť**

Prenositeľnosť

* Je spôsobilosť softvérového produktu byť transformovaný z jedného prevádzkového prostredia do iného.
* Subcharakteristiky:
  + **adaptabilita** – Môže byť softvér premiestnený na iné prostredia?
  + **inštalovateľnosť** – Je softvér jednoducho inštalovateľný?
  + **integrovateľnosť** – Zodpovedá softvér štandardom portability?
  + **nahraditeľnosť** – Dokáže softvér nahradiť iný softvér?
  + **zhoda s požiadavkami na prenositeľnosť**

ISO 9126-2:2000 – Externé metriky

* slúžia na získavanie externých atribútov softvérového produktu
* testovanie, vykonávanie, pozorovanie
* „blackbox“
* nevýhoda – len na hotový produkt
* použitie – nezávislé testovanie

ISO 9126-3:2003 – Interné metriky

* slúžia na získavanie interných atribútov softvérového produktu
* aplikácia na špecifikáciu alebo zdrojový kód
* „whitebox“
* nevýhoda – nie na všetky charakteristiky

ISO 9126-4:2004 – Použitie metrík akosti

* nemeria sa kvalita produktu
* meria sa kvalita pre rôznych používateľov a scenáre
* použitie – aj na informačný systém ako celok

Príklad

* Charakteristika účinnosť, subcharakteristika reakčný čas
* externá -> doba odozvy

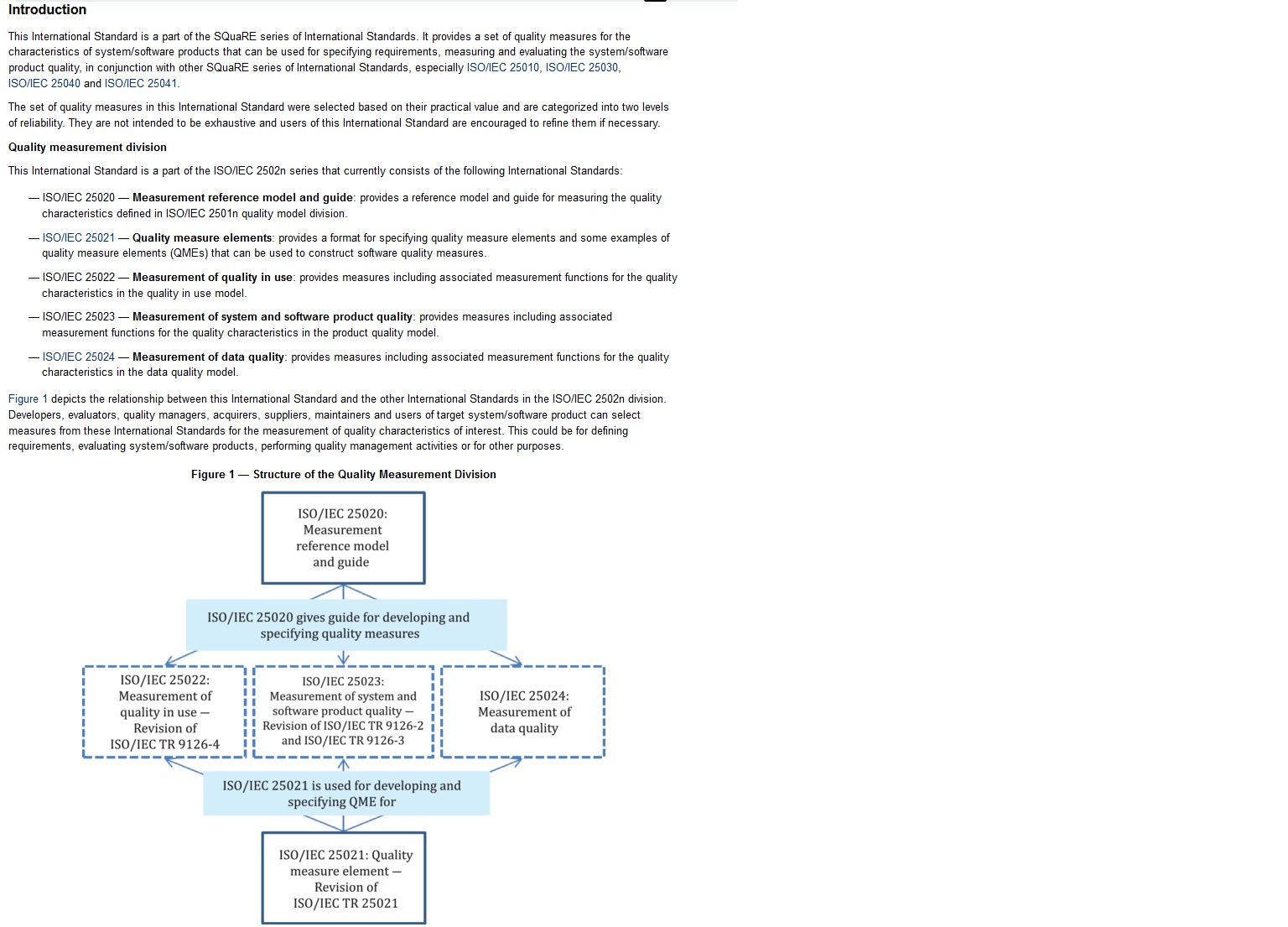
T = (čas získania výsledku) – (čas ukončenia vloženia úlohy)

* interná -> časová priepustnosť

Výsledok = počet špecifikovaných operácií vykonateľných za jednotku času

Štruktúra a obsah noriem radu ISO 9126





CMMI  
(Capability Maturity Model Integration)

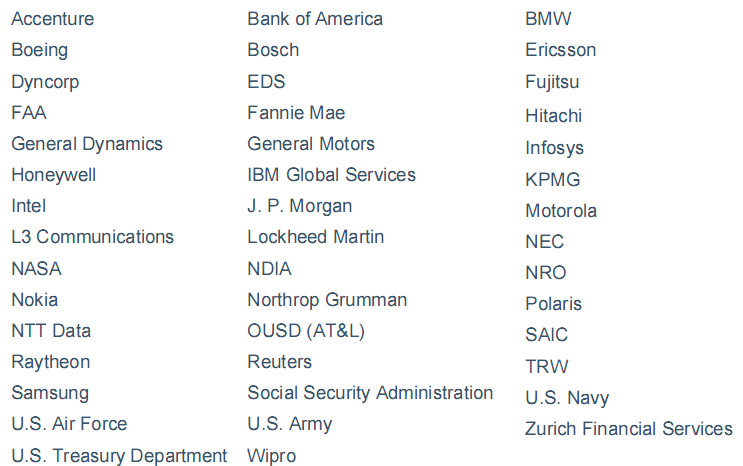
Myšlienka

* Detailnejšie pokrytie životného cyklu produktu
* Zabezpečiť elimináciu problémov
* Integrovať softvérové a systémové inžinierstvo
* Vytvoriť niečo hodnotné pre organizáciu
* Umožniť voľbu reprezentácie podľa obchodného zámeru

História

* 1987 – 1997: CMM modely
  + 1987: SEI-87-TR-24 (SW-CMM questionnaire), released
  + 1989: Managing the Software Process, published
  + 1991: SW-CMM v1.0 (for Software Engineering)
  + 1996: SECM V1.1 (Systems Engineering Capability Model)
  + 1997: IPD-CMM V0.98 (for Integrated Product Development)
* 2000 – 2007: CMM Integration
  + 2000: CMMI-SE/SW v1.02, released
  + 2002: CMMI v1.1, released
  + 2006: CMMI v1.2, released
    - currently, just CMMI Development exist
    - in 2007, planned release of
      * CMMI Acquisition
      * CMMI Services

Organizácie používajúce CMMI



Procesné oblasti

* System engineering – SE (systémové inžinierstvo)
* Software engineering – SW (softvérové inžinierstvo)
* Integrated product and process development – IPPD (integrovaný produktový a procesný vývoj)
* Supplier sourcing – SS (zaisťovanie dodávok)

CMMI-SE

* Zahŕňa vývoj celého systému (nie len SW)
* Transformácia požiadaviek zákazníka
* Vlastnosti a obmedzenia výsledného produktu
* Podpora počas nasadenia

CMMI-SW

* Vývoj softvérových systémov
* Aplikovanie systematických

a merateľných prístupov

* + Vývoj
  + Obsluha
  + Údržba

CMMI-IPPD

* Podporná oblasť pre CMMI disciplíny
  + Softvérové inžinierstvo
  + Systémové inžinierstvo
* Efektívne využitie zdrojov a aktív
* Formulácia postupov pre maximálnu spokojnosť zákazníka

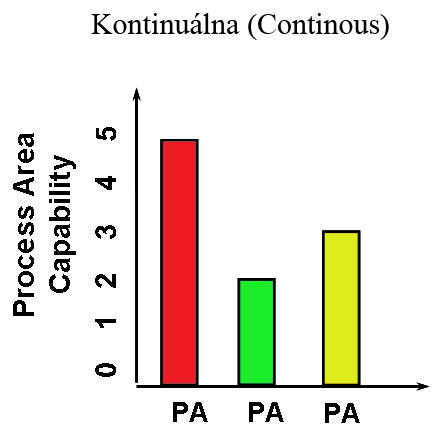
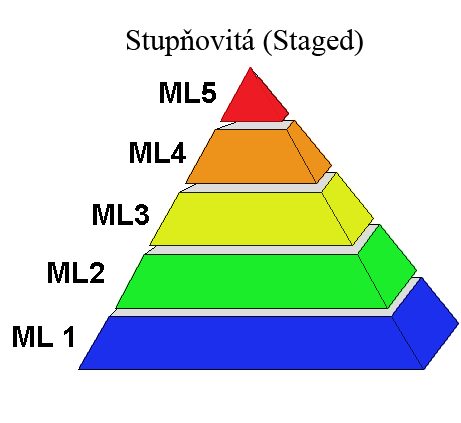
CMMI-SS

* Externí dodávatelia
* Monitorovanie aktivít dodávateľa
* Súbor opatrení

CMMI reprezentácie

* **Stupňovitá**
  + Zameraná na celkovú úroveň
  + Maturity levels
  + Podobná štruktúra ako v SW-CMM
* **Kontinuálna**
  + Zameraná na procesy
  + Capability levels
  + Podobná štruktúra ako v SE-CMM

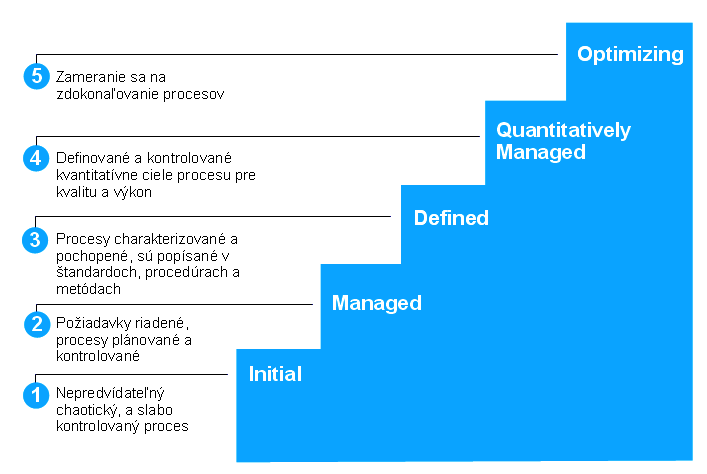
CMMI reprezentácie



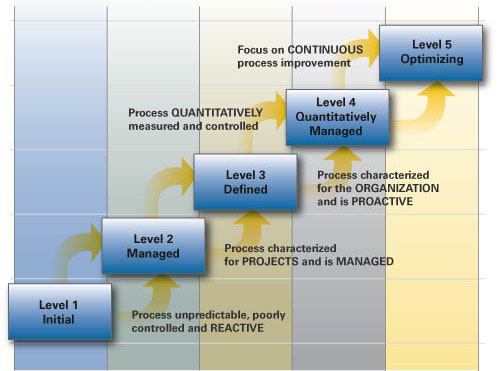
Stupňovitý model

* Procesné oblasti organizované do 5-tich úrovní (maturity levels)
* Úroveň má preddefinovanú sadu procesných oblastí
* Poskytuje odskúšanú cestu zlepšenia
* Jednoznačné označenie úrovne

Maturity levels



Prechod medzi úrovňami



Opakovateľná (managed)

* Riadenie požiadaviek (Requirements Management)
* Plánovanie projektov (Project Planning)
* Sledovanie a riadenie priebehu projektu (Project Monitoring and Control)
* Riadenie subdodávok (Supplier Agreement Management)

Opakovateľná (managed)

* Meranie a analýza (Measurement and Analysis)
* Kontrola kvality procesov a produktov (Process and Product Quality Assurance)
* Konfiguračné riadenie (Configuration Management)

Definovaná (defined)

* Spracovanie požiadaviek (Requirements Development)
* Technické riešenie (Technical Solution)
* Integrácia (Product Integration)
* Verifikácia (Verification)
* Validácia (Validation)
* Vývoj procesov organizácie (Organizational Process Focus)
* Formulácia procesov (Organizational Process Definition)
* Školenie organizácie (Organizational Training)
* Jednotné projektové riadenie (Integrated Project Management)
* Riadenie rizík (Risk management)
* Zlepšenie organizačnej štruktúry (Integrated Teaming)
* Integrované riadenie dodávateľov (Integrated Supplier Management)
* Analýza problémov a prevencia (Decision Analysis and Resolution)
* Organizačné prostredie pre integráciu (Organizational Environment for Integration)

Riadená (quantitatively managed)

* Sledovanie výkonu procesov (Organizational Process Performance)
* Riadenie projektov podľa výkonnosti (Quantitative Project Management)

Optimalizujúca (optimizing)

* Inovácie organizačných procesov

a technológií (Organizational Innovation and Deployment)

* Analýza a zdôvodnenie rozhodovania (Causal Analysis and Resolution)

Výhody

* V danej oblasti je preddefinovaný postup
* Zamerané na vylepšenie organizácie ako celku
* Viacero výsledkov sumarizovaných v jednej úrovni
* Jednoznačné zaradenie organizácie
* Celkový pohľad na organizáciu

Kontinuálna reprezentácia

* Procesné oblasti organizované podľa kategórií
  + Process Management
  + Project Management
  + Engineering
  + Support
* Sloboda výberu procesnej oblasti
* Procesná oblasť – 6 úrovní (capability levels)
* Jednoznačné vyhodnotenie každej procesnej úrovne

Capability levels



Capability levels

0. Neúplná (Incomplete)

* + Nevykonávaný alebo neúplný proces

1. Vykonávaná (Performed)

* + Umožňuje prácu potrebnú na produkciu produktu

2. Opakovateľná (Managed)

* + Infraštruktúra na jeho podporu
  + Plánovaný, vykonávaný, monitorovaný a vyhodnocovaný

3. Definovaná (Defined)

* + Spadajúce pod organizačnú množinu štandardných procesov

4. Riadená (Quantitatively Managed)

* + Kontrolované pomocou štatistických

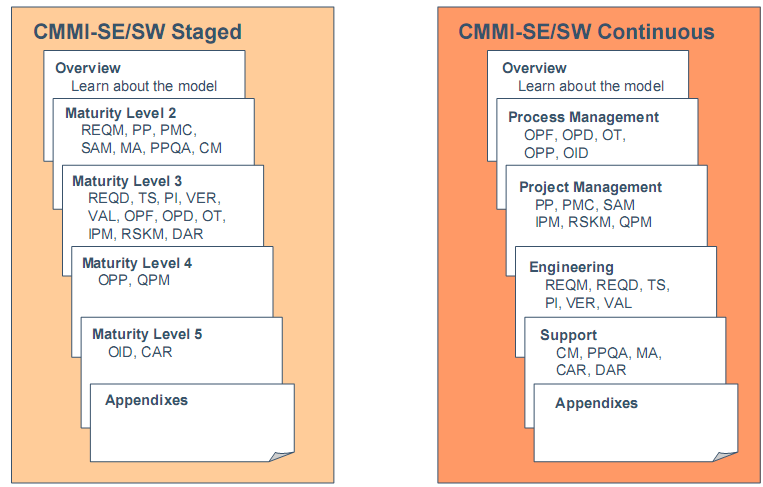
a iných kvantitatívnych techník

5. Optimalizujúca (Optimizing)

* + Zlepšované na základe pochopenia dôvodov zmien obsiahnutých v procese

Výhody

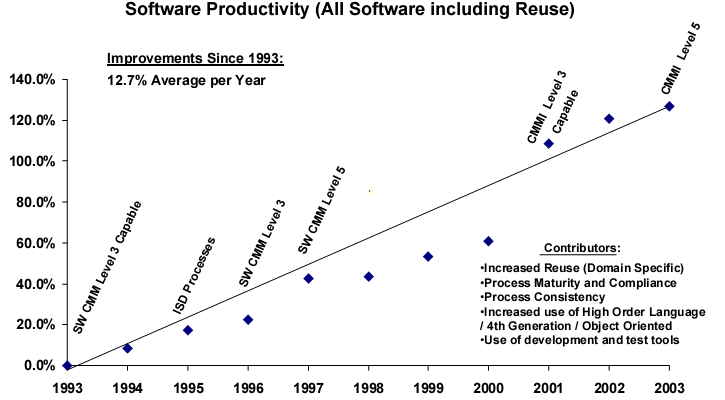
* Maximálna flexibilita pre možné zoradenie procesných vylepšení
* Zamerané na zlepšenie vo vnútri procesných oblastí
* Procesy môžu napredovať rôznym tempom
* Je možné zamerať sa na rizikové oblasti v ľubovoľnej úrovni
* Vhodnejší pre špecializovaných odborníkov

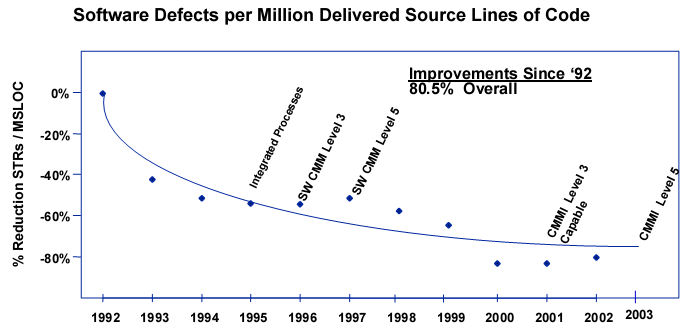
Jeden model, dva pohľady

Lockheed Martin

* Americká zbrojárska firma
* 1993 – maturity level 3 CMM
* 2003 – maturity level 5 CMMI

Lockheed Martin (produktivita tvorby SW)



Lockheed Martin (znižovanie počtu chýb) 

Zhrnutie

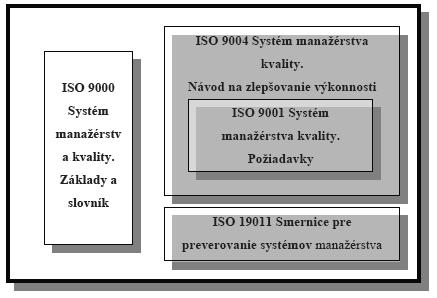
* Prínos CMMI
  + Zlepšenie predpovede plánovania a rozpočtu
  + Zlepšenie znovupoužitia získaných znalostí
  + Zvýšená produktivita
  + Vylepšená kvalita (ako miera zhoršenia)
  + Väčšie uspokojenie zákazníka
  + Vyššia morálka zamestnancov
  + Väčšia návratnosť investícií
  + Zníženie nákladov na kvalitu

ISO 9000

Rodina noriem ISO 9000

* ISO 9000:2005
  + základy a slovník
* ISO 9001:2000
  + požiadavky
* ISO 9004:2000
  + smernica na zabezpečenie zlepšovania

Štruktúra súboru noriem ISO 9000:2000



Zásady manažérstva kvality

* Orientácia na zákazníka
* Vodcovstvo
* Angažovanosť pracovníkov
* Procesný prístup
* Systémový prístup
* Neustále zlepšovanie
* Prijímanie rozhodnutí na základe faktov
* Navzájom výhodné vzťahy s dodávateľmi

Orientácia na zákazníka

* systematické preskúmavanie požiadaviek zákazníkov
* rýchle a efektívne uspokojovanie týchto požiadaviek
* previazanosť cieľov kvality s potrebami

a očakávaním zákazníkov

* systematické meranie spokojnosti a lojality zákazníkov
* rozvoj a riadenie vzťahov so zákazníkmi
* rozvoj vzťahov i s ďalšími zainteresovanými stranami

Vodcovstvo

* deklarovanie vízie, politiky, cieľov organizácie v súlade s požiadavkami zákazníkov

a zainteresovaných strán

* vytváranie pracovného prostredia vzájomnej dôvery medzi zamestnancami a manažmentom
* poskytovanie príležitostí zamestnancom

k vlastnej aktívnej a tvorivej práci vrátane určenia zodpovednosti a právomoci

* motivácia zamestnancov k tímovej práci

a k zlepšovaniu

Zapojenie ľudí

* vysvetľovanie dôležitosti všetkých činností svojich zamestnancov pre dosahovanie výsledkov organizácie
* vedenie zamestnancov k otvorenej diskusii

o ich nedostatkoch a hľadanie riešenia

* trvalé vzdelávanie zamestnancov na všetkých úrovniach
* odmeňovanie zamestnancov za pracovné výkony
* motivovanie zamestnancov k lepším výkonom

Procesný prístup

* definovanie procesov pre dosiahnutie požadovaných výsledkov
* identifikácia a meranie vstupov a výstupov z procesov za súčasného identifikovania externých a interných dodávateľov a zákazníkov
* vyhodnotenie možných rizík, súvislostí a vplyvov procesov na zákazníkov a všetky ostatné zúčastnené strany
* stanovenie jasných zodpovedností a právomoci pre riadenie procesov
* identifikovanie interných a externých zákazníkov, dodávateľov a ostatných účastníkov procesov

Systémový prístup

* identifikovanie, porozumenie a riadenie systému vzájomne súvisiacich procesov zameraných na daný cieľ prispieva k efektívnosti a účinnosti organizácie pri dosahovaní jej cieľov
* vyžaduje činnosti ako pochopenie vzájomných väzieb medzi procesmi
* neustále zlepšovanie systému prostredníctvom merania a vyhodnocovania

Neustále zlepšovanie

* plánovať (Plan) – stanovenie cieľov a procesov
* vykonať (Do) – zavedenie procesov
* kontrolovať (Check) – monitorovanie a meranie procesov a produktov

Orientácia na fakty pri rozhodovaní

* uskutočnenie zberu dostatočne presných

a spoľahlivých dát z jednotlivých procesov v organizácii

* využitie vhodných štatistických metód zberu a analýzy
* vedenie ľudí k využitiu štatistických metód
* zabezpečenie ochoty manažérov analyzované dáta využívať pri rozhodovaní
* pokiaľ je to možné, sprístupniť výsledky analýzy dát zamestnancom

Vzájomná prospešnosť vzťahov  
s dodávateľmi

* výber kľúčových, resp. strategicky významných dodávateľov
* pravidelné hodnotenie dodávateľov
* poskytovanie rôznych druhov pomoci
* efektívna komunikácia a obojstranná dôvera
* motivácia dodávateľov k zlepšovaniu a pod.

Ako implementovať ISO 9000   
v praxi?

* Vôľa manažmentu
* Nutnosť
* Zostavenie tímu kvality
* Predmet systému manažérstva kvality
* Politika kvality
* Ciele kvality
* Príručka kvality
* Ďalšia dokumentácia

Manažment kvality

* Určenie zodpovedných osôb
* Manažér kvality
* Predstaviteľ manažmentu pre kvalitu
* Tím kvality

Manažér kvality

1. Zabezpečuje kontrolu plnenia politiky a cieľov kvality

v spoločnosti.

1. Zostavuje tím kvality.
2. Vedie tím kvality.
3. Zostavuje tím interných audítorov kvality.
4. Vedie tím interných audítorov kvality.
5. Predkladá návrh plánu interných auditov kvality manažmentu.
6. Kontroluje dodržiavanie systému manažérstva kvality

a následne overuje jeho funkčnosť pomocou interných auditov.

1. Určuje konkrétne opatrenia k náprave a preventívne opatrenia v systéme manažérstva kvality.
2. Zabezpečuje kontrolu opatrení pri nápravnej a preventívnej činnosti.
3. Vykonáva a organizuje externé audity systému manažérstva kvality u dodávateľov.
4. Rokuje za spoločnosť s dodávateľmi v oblasti kvality.
5. Schvaľuje a vyhodnocuje návrhy na stanovenie politiky a cieľov kvality.
6. Podáva informácie a správy riaditeľovi spoločnosti o fungovaní systému manažérstva kvality a o akýchkoľvek potrebách na zlepšenie.
7. Navrhuje zmeny a úpravy dokumentov týkajúcich sa systému manažérstva kvality.
8. Zabezpečuje vykonávanie rozborov a vyhodnocovanie vývoja kvality v spoločnosti.
9. Činnosť predstaviteľa manažmentu pre kvalitu vykonáva v súlade s medzinárodnou normou ISO 9001.
10. Vykonáva metodickú činnosť pre zamestnancov v rámci ďalšieho zlepšovania systému manažérstva kvality podľa medzinárodných noriem radu ISO 9000 a zabezpečuje podporu povedomia o zameraní na zákazníka v rámci spoločnosti.

Tím kvality

* vytvorenie potrebnej dokumentácie pre jemu podriadený alebo určený útvar
* zjednodušenie a prepracovanie vyplývajúcej dokumentácie vzhľadom k medzinárodnej norme ISO 9001
* plnenie politiky a cieľov kvality v jemu danom rozsahu, osvojenie si a poznanie dokumentácie systému riadenia kvality
* implementácia jednotlivých postupov v danom útvare a zabezpečenie poznania a osvojenia si dokumentácie u svojich podriadených
* vykonávanie jednotlivých činností podľa aktuálnej dokumentácie u svojich podriadených

Politika kvality

1. Naše produkty a služby zameriavame na potreby zákazníkov.
2. Svojou činnosťou uspokojujeme potreby zákazníkov, partnerov, zamestnancov i majiteľov.
3. Trvalý rast a pokrok dosahujeme neustálym zvyšovaním kvality produktov a služieb.
4. Kvalitu našich produktov a služieb vytvárajú vzdelaní, tvoriví a angažovaní zamestnanci.
5. Meradlom kvality sú pre nás svetové štandardy a špičkové technológie.

Ciele kvality

Strategické ciele:

1. Udržať dlhodobý rast objemu tržieb za poskytnuté služby nad úrovňou 15%.
2. Posilniť pozíciu spoločnosti na trhu ako dodávateľa komplexných riešení najmä v oblastiach dodávania hardvéru, vlastného softvéru a konzultačných služieb v oblasti IT.
3. Skvalitniť riadenie spoločnosti na úrovni vedenia, ako aj manažérov pre projekty dlhodobým manažérskym vzdelávaním. Zvýšiť odborné vzdelanie zamestnancov a hĺbku ich poznatkov.
4. Trvalo zvyšovať hodnotu spoločnosti a jej atraktívnosť pre zákazníkov, ako aj pre zamestnancov.
5. Pripraviť spoločnosť na akvizíciu spoločnosti XY a transformáciu na akciovú spoločnosť.

Krátkodobé ciele:

1. Zvýšiť tržby spoločnosti o 20% oproti predchádzajúcemu roku.
2. Rozšíriť počet interných audítorov podľa ISO 9001:2000.
3. Vyškoliť 1 pracovníka s certifikátom Microsoft pre vývoj softvéru.
4. Nájsť u kľúčových zákazníkov oblasť pre rozšírenie pôsobenia – portfólia spoločnosti.
5. Uzavrieť minimálne jednu zákazku s novým potenciálne strategickým partnerom.

Interný audit

* Požiadavka na neustále zlepšovanie
* Dokument na riadenie interných auditov
* Audítorský tím
* Plán interného auditu
* Audit je zameraný smerom ku konštruktívnym riešeniam
* Nielen ako sankčná akcia

Ciele auditu

* Nájsť nezhody s požiadavkami ISO 9001
* Nájsť riešenie týchto nezhôd
* Nájsť priestory na zlepšenie systému manažérstva kvality

Typy nezhôd

**Realizačná nezhoda** – ide napr. o nevykonanie niektorých zo stanovených akcií (činností, úkonov, požiadaviek). Vzniká väčšinou pri zavalení pracovníkov operatívou, zabudnutím alebo zanedbaním niektorých zo stanovených požiadaviek. Takáto nezhoda negeneruje ďalšie nezhody.

**Systémová nezhoda** – ide napr. o nezavedenie

a nerealizovanie niektorej z požiadaviek normy, poprípade združenie niekoľkých realizačných nezhôd vyskytujúcich sa u viacerých auditovaných organizačných jednotiek. Systémové nezhody generujú ďalšie nezhody a opakovanie nezhôd, to znamená nedodržanie ďalších požiadaviek normy ISO 9001 alebo iných požiadaviek stanovených a vyžadovaných spoločnosťou.

Certifikácia

* Externý audit
* Akreditovaná certifikačná autorita
* Plnenie požiadaviek podľa ISO 9001
* Dodržiavanie pravidiel a postupov
* Znalosti zamestnancov o systéme kvality