Softwarové inženýrství I

Životní cyklus SW a související aktivity. Specifikace požadavků a jejich typy. Štrukturované vs objektově orientované metody analýzy a návrhu.
PB007

Životný cyklus SW a súvisejúce aktivity

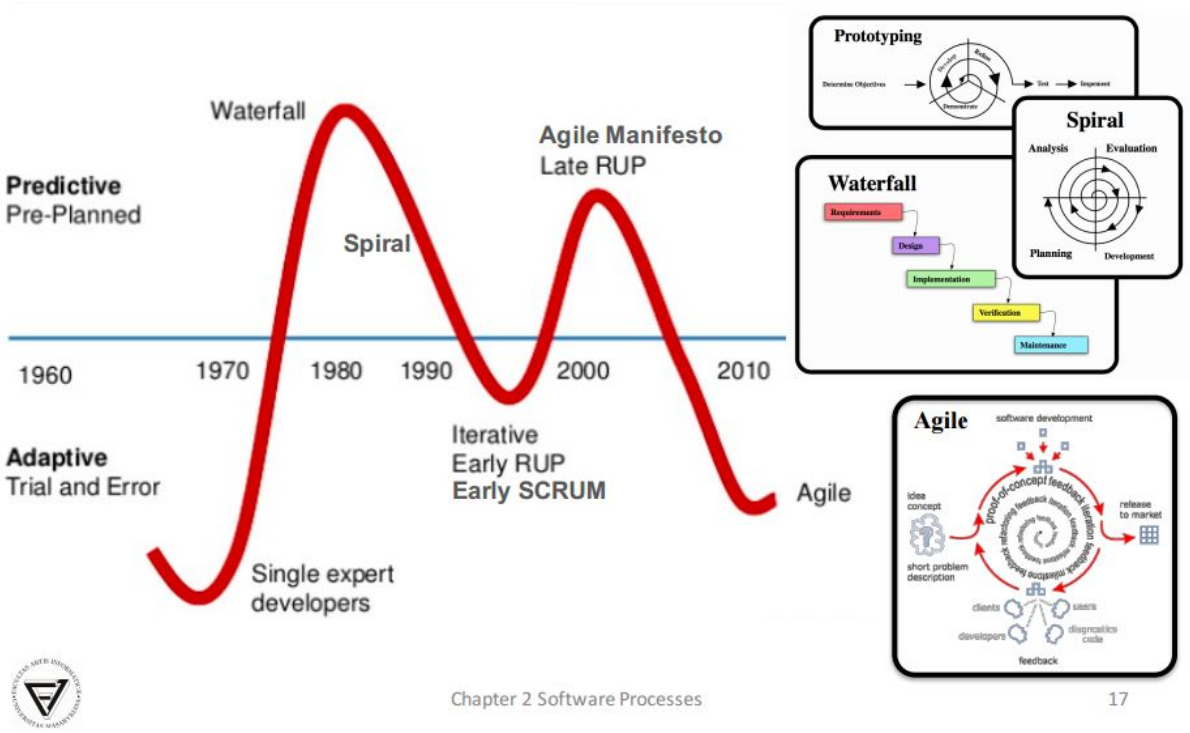
Proces tvorenia softwéru môže mať rôzne podoby no všetky zahŕňajú:

- Špecifikáciu požiadavkou
- Development:
    - Analýza a dizajn
    - Implementácia
- Validácia a verifikácia
- Zlepšovanie (Evolution)

2 základné skupiny softvéru:

- Generic products - produkty “v krabici”, ktoré si môže hocikto kúpiť a používať
- Customized products - software na mieru

- Spoľahlivosť a výkonnosť by mali byť dôležité pre každý systém
- Software by mal byť tovorený na základe predom stanoveného procesu



Špecifikácia požiadaviek

Požiadavky sú popisom systémových služieb (services) a záväzkov (constraints) ktoré sú kladené na vyvíjaný softvér. Môžu sa líšiť v konkrétnosti od slovného popisu cez štruktúrovaný jazyk, grafické notácie po matematickú špecifikáciu danej úlohy / požiadavku.

Requirement engineering - proces stanovania požiadaviek.

Máme **2 typy** požiadaviek a to :

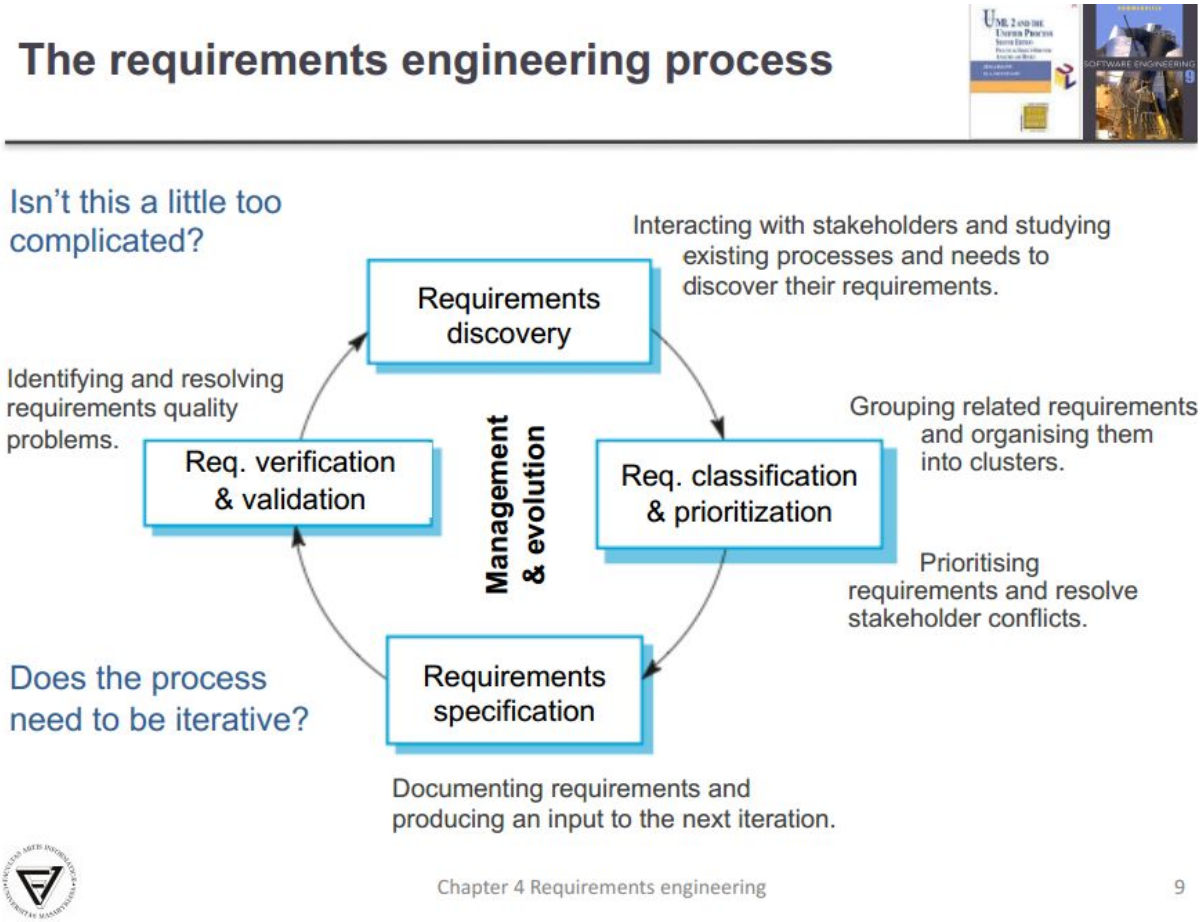
- Funkčné
    - Popisujú služby, ktoré systém poskytuje, popisujú ako systém má reagovať na konkrétne inputy a ako by sa mal správať v konkrétnych situáciách
    - Príklad : Užívateľ **by mal byť schopný** prehľadávať všetky svoje termíny vo všetkých klinikách kde je zaregistrovaný.
- Ne-funkčné požiadavky
    - Vlastnosti a záväzky kladené na služby poskytované systémom, ako napríklad časové obmedzenia, spoľahlivosť a bezpečnosť systému, požiadavky na vývojový proces, platforma, štandardy, atď.

- Príklad: Systém by mal byť dostupný Pondelok až Piatok od 8:00 do 18:00 hod. S odstávkou nie dlhšou ako 5 sekúnd v hociktorom dni.

Pre **ohodnotenie kvality** stanovených požiadaviek je dvoležité sledovať nasledujúce body:

- Kompletnosť - je celá funkcionalita v požiadavkách zahrnutá?
- Konzistentnosť - nachádzajú sa v požiadavkách konflikty alebo kontadikcie?
- Presnosť/Konkrétnosť - existuje len jedno vyloženie požiadaviek v danom kontexte ?
- Overiteľnosť - dá sa splnenie požiadaviek skontrolovať?
- Realistickosť - je možné dané požiadavky splniť, sú reálne s danými zdrojmi ?

Validácia požiadaviek sa zaoberá tým, či systém ktorý je nimi definovaný naozaj odpovedá tomu čo zákazník chce/potrebuje. To môžeme docieliť napríklad pomocov reviews alebo poskytnutím prototypu či A/B testingom (dve verzie sú predstavene rôznym skupinám užívateľov a podla ich reakcií sa rozhodne ktorá sa vo výsledku použije, metódu je možné opakovať).



Softvéroví inžinieri komunikujú hlavne s tzv stakeholders a definujú hore popísané požiadavky, v tomto bode nie je dôležité ako systém bude pracovať (implementácia) ale čo bude robiť. Toto môžu praktizovať rôznymi spôosobmi pomocov interakcie s ľuďmi - áno aj to je informatika (SCARY!!), alebo analýzov axistujúcich procesov, ktoré by systém mal obsluhovať.

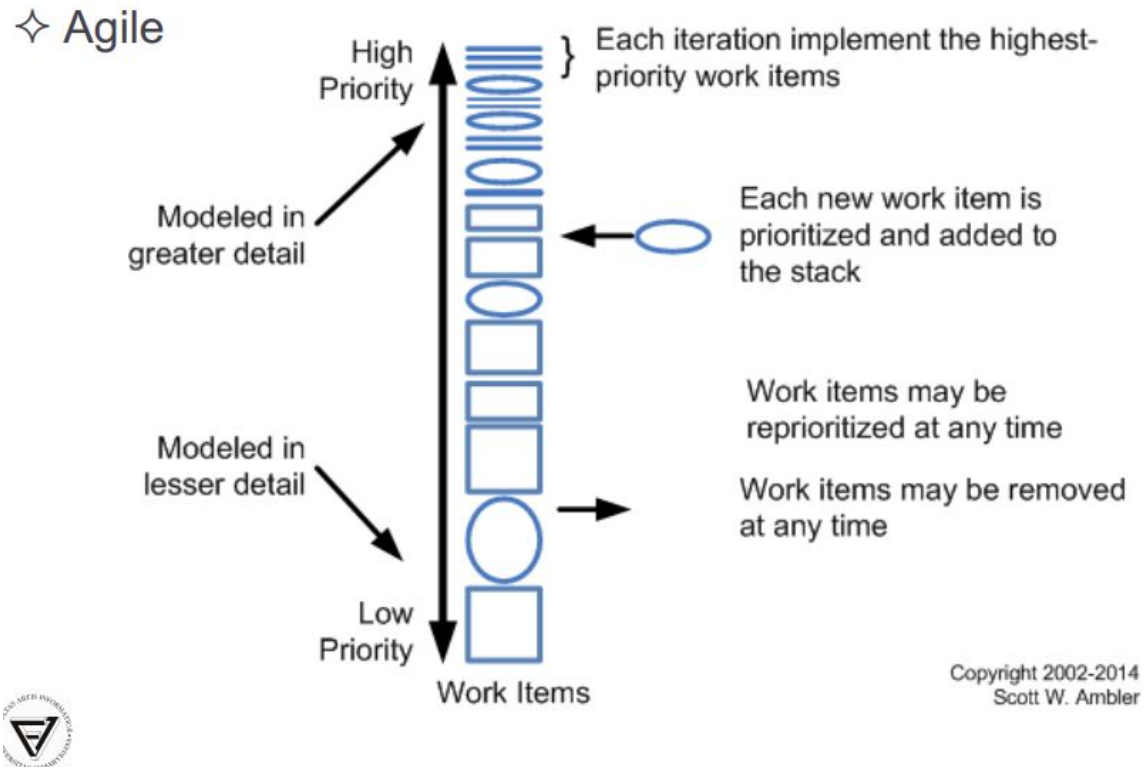
Klasifikácia a prioritizácia požiadavkov môže biť založená na rôznych princípoch

MoSCoW kritériá :

- **M**ust have
- **S**hould have - dôležité, ale je možné ich opomenúť
- **C**ould have - naozaj len možnosti napr. Pre zvýšenie ceny projektu
- **W**ant to have - môže počkať na ďalšie releasy a pod.

Rational Unified Process (RUP) attributes - akési vlastnosti požiadaviek

- Status - Proposed/Approved/Rejected/Incorporated
- Benefit - Critical/Important/Useful
- Effort - počet hod. práce a pod.
- Risk - High/Medium/Low
- Stability - High/Medium/Low
- Target release - verzia produktu v ktorej má byť požiadavok splnený



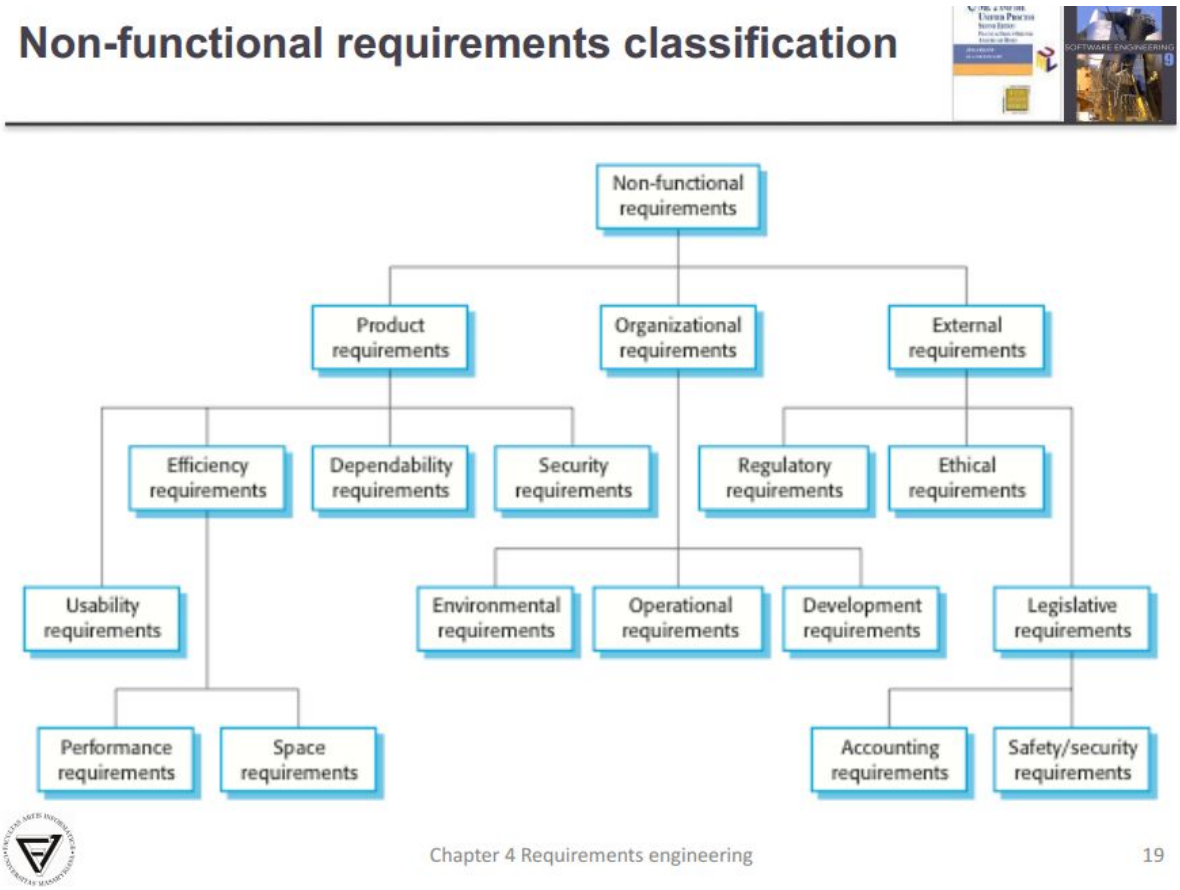
Manažment požiadaviek je proces správy požiadaviek ktoré sa menia počas ich zadávania ale aj developmentu systému. Každá meniaca sa požiadavka by mala byť analyzovaná pred tým než sa rozhodne, či sa zmení.

Nové požiadavky vstávajú najmä z dôvodu zmien v biznise, organizácii a technológii.

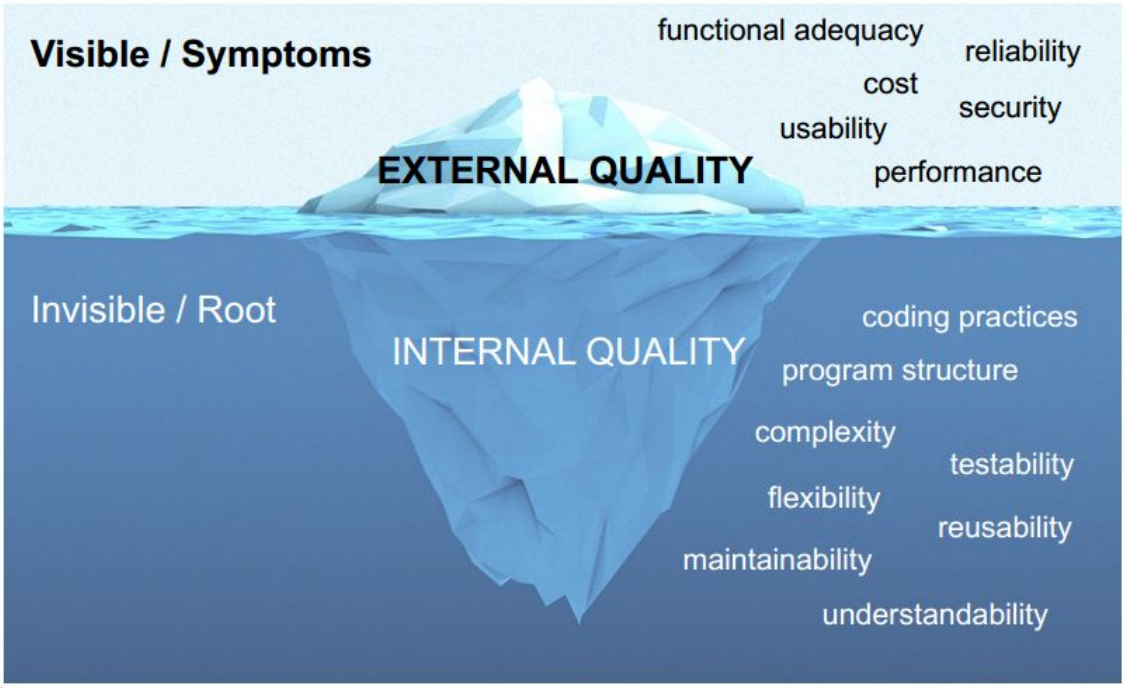
Je dôležité uchovávať závislosti medzi požiadavkami aby bolo jednoduchšie požiadavky meniť a mať pri tom prehľad o dôsledkoch zmien. Tejto vlastnosti sa hovorí Traceability.

Ne-funkčné požiadavky

- Pomáhajú s definíciou kvality a definujú požiadavky okolia ako napríklad právne záväzky alebo požiadavky organizácie
- Ovplyvňujú celú architektúru systému
- Môžu generovať funkčné požiadavky alebo zamedziť už existujúce požiadavky



Produktové požiadavky špecifikujú kvalitu s akou musí systém pracovať (rýchlosť, spoľahlivosť,...)



- Dependability
  - Availability
  - Reliability
  - Safety
  - Security
- Efficiency
  - Performance
  - Space/resource utilization
- Usability
- Modifiability
- Testability
- Resilience
- Robustness
- Portability
- Adaptability
- Complexity
- Modularity
- Reusability
- Understandability
- Learnability

Availability (prístupnosť) je pravedpodobnosť, že systém bude v danom čase funkčný a pripravený poskytnúť požadovanú službu. Zaoberá sa tým ako dlho systém musí pracovať bezchybne a ako dlho je prípustné aby bol nedostupný (out of operation). Prístupnosť môžme vyjadriť ako MTTF/(MTTF + MTTR) (Mean Time To Failure/Repair)

Reliability(Spolahlivosť) je pravdepodobnosť behu systému bez chyby v danom prostredí po daný čas pre daný význam. Zaoberá sa tým ako systém detekuje chyby, ako často sa chyby vyskytujú a čo sa deje ak sa vyskytnú. Vyjadruje sa pomocou Pravdepodobnosti zlyhania (POFOD - Probability of failure on demand).

Term	Description
Human error or mistake	Human behavior that results in the introduction of faults into a system. *E.g., in the wilderness weather system, a programmer might decide that the way to compute the time for the next transmission is to add 1 hour to the current time. This works except when the transmission time is between 23.00 and midnight .*
System fault	A characteristic of a software system that can lead to a system error. *E.g., the inclusion of the code to add 1 hour to the time of the last transmission, without a check if the time is greater than or equal to 23.00.*
System error	An erroneous system state that can lead to system behavior that is unexpected by system users. *E.g., the value of transmission time is set incorrectly (to 24.XX rather than 00.XX) when the faulty code is executed.*
System failure	An event that occurs at some point in time when the system does not deliver a service as expected by its users. *E.g., no weather data is transmitted because the time is invalid.*

Safety (Bezpečnosť pre okolie) špecifikuje schopnosť systému pracovať bez poškodenia človeka alebo svojho okolia.

Term	Definition
Accident (or mishap)	An unplanned event or sequence of events which results in human death or injury, damage to property, or to the environment. *E.g. an overdose of insulin.*
Hazard	A condition with the potential for causing or contributing to an accident. *E.g. a failure of the sensor that measures blood glucose.*
Damage	A measure of the loss resulting from a mishap. Damage can range from many people being killed as a result of an accident to minor injury or property damage. *E.g., damage resulting from an overdose of insulin could be serious injury or the death of the user of the insulin pump.*
Hazard severity	An assessment of the worst possible damage that could result from a particular hazard. *E.g. when an individual death is a possibility, a reasonable assessment of hazard severity is 'very high'.*
Hazard probability	The probability of the events occurring which create a hazard. Probability values tend to be arbitrary but range from 'probable' (say 1/100 chance) to 'implausible'. *E.g. the probability of a sensor failure in the insulin pump that results in an overdose is probably low.*
Risk	This is a measure of the probability that the system will cause an accident. The risk is assessed by considering the hazard probability, the hazard severity, and the probability that the hazard will lead to an accident. *E.g. the risk of an insulin overdose is probably medium to low.*

Security (zabezpečenie) je vlastnosť systému odolávť vonkajším zásahom. Ide o ochrane systému na úrovni ochrany informácií pred neoprávneným prístupom alebo poškodením či ochrany pred zamedzením prístupu iným užívateľom. Je to dôležitý faktor ktorý je pre-requizitou aj pre availability, reliability a safety.

Term	Definition
Asset	Something of value which has to be protected (e.g. *patients records in a health-care system*). The asset may be the system itself or data used by that system.
Exposure	Possible loss or harm to a computing system (e.g. *financial loss from patients' legal action or loss of reputation*). This can be loss or damage to data, or can be a loss of time and effort if recovery is necessary after a security breach.
Vulnerability	A weakness in a computer-based system that may be exploited to cause loss or harm (e.g. *weak password*).
Attack	An exploitation of a system's vulnerability. Generally, this is from outside the system and is a deliberate attempt to cause some damage.
Threats	Circumstances that have potential to cause loss or harm. You can think of these as a system vulnerability that is subjected to an attack (e.g. *guessing the weak password*).
Control	A protective measure that reduces a system's vulnerability. E.g. *encryption is an example of a control that reduces a vulnerability of a weak access control system, or a password checking system in our example*.

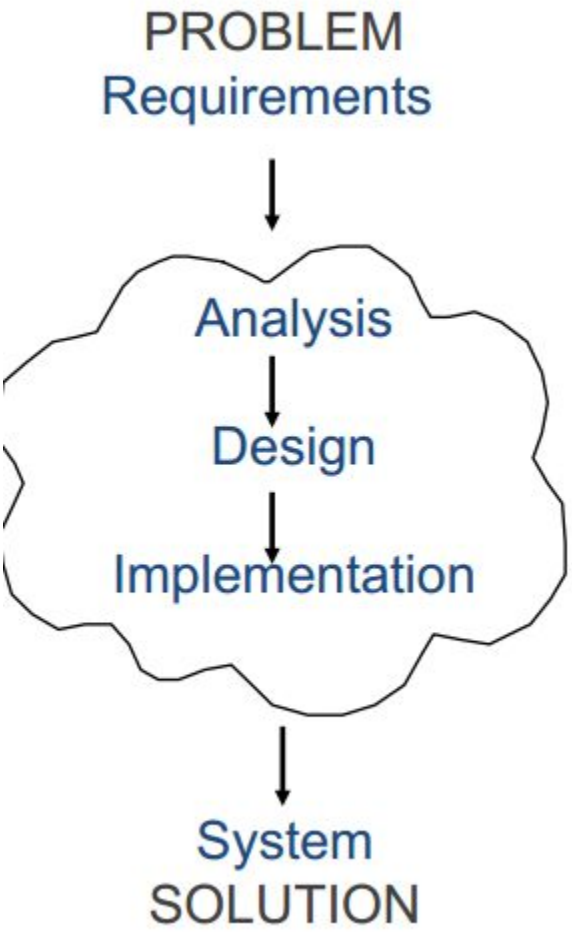
Organizačné požiadavky vstávajú z organizačnej politiky a procedúr ktoré systém musí dodržiavať.

Externé požiadavky sú definované na základe vonkajších faktorov a z procesu tvorby systému. Môžu nimi byť napríklad spomínané právne normy a pod.

Analíza, design a implementácia

- Zjendotene nazývame Software developement

Počas **analízi** sa snažíme identifikovať softvérové procesy, entity a ich vzťahty.
Dizajn popisuje analitické modely implementačnými detajlami.
Implementácia je realizáciou dizajnu.



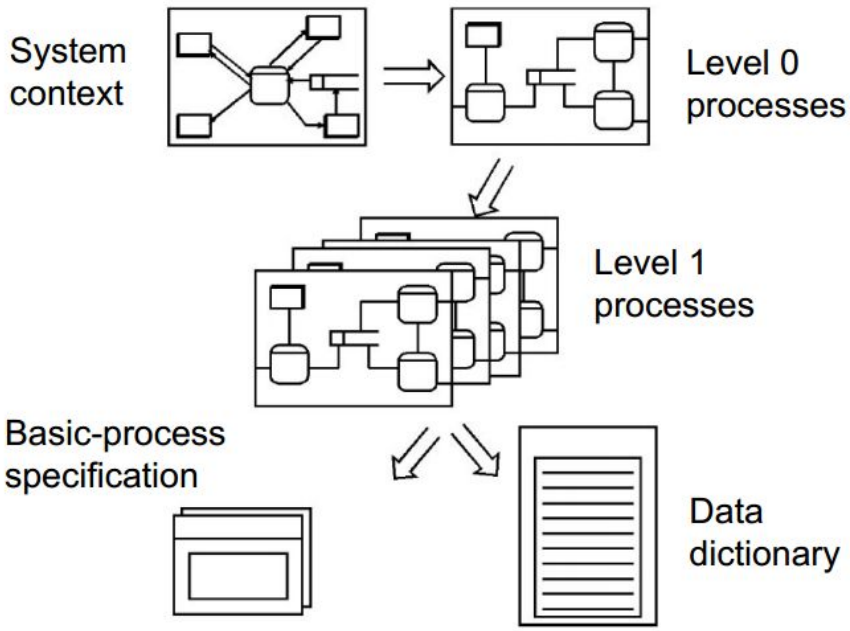
Štrukturované VS Ojektovo orientované metódy

Existuje niekoľko pohľadov na softvérové systémy.
Funkcionálne orientovaný pohľad hľadí na systém ako na súbor funkcií ktoré komunikujú prostredníctvom dát a control flow.
Dátovo orientovaný pohľad hľadá základné dátové štruktúry systému. Funkcionálne aspekty systému sú menej dôležité.
Objektovo orientovnaý pohľad pozerá na systém ako na súbor objektov, ktoré zahŕňajú ako dáta tak operácie nad nimi.

Štrukturovaná analýza využíva funkcionálne orientovaný pohľad v spojení s datovo orientovaným cez koncept dátovej dekompozície. Objektovo orientovná analýza je postavená na objektovo orientovanom pohľade.

Štrukturovaná analýza a dizajn

Rozdeľuje projekt na malé, dobre definované aktivity a definuje poradie a interakcie aktivít. Používa hierarchické grafické techniky pre zobrazenie svojich zámerov. Je efektívna pre štruktúrovanie projektu na menšie časti ktoré zjednodušujú manažment projektu.



Examplary method (Gane-Sarson)

1. Define system context and create initial system DFD.
2. Draft initial data model (ERD).
3. Analyze data entities and relationships into final ERD.
4. Refine DFD according to the ERD data model (create logical process model).
5. Decompose logical process model into procedural elements.
6. Specify the details of each individual procedural element.

Objektovo orientovaná analýza a dizajn

Prístup ktorý modeluje systém ako skupinu objekotov. Každý objekt reprezentuje nejakú entitu v systéme ktorý modelujeme a je charakterizovaná svojou classou, stavom a správaním.

Examplary method (Unified Process, analysis and design excerpt)

1. Requirements
   - System boundary, actors and requirements modelling with **Use Case diagram**.
2. Analysis
   - Identification of analysis classes, relationships, inheritance and polymorphism, and their documentation with a **Class diagram**.
   - Use Case realization with **Interaction** and **Activity diagrams**.
3. Design
   - Design classes, interfaces and components, resulting in refined **Class diagrams** and **Component diagrams**.
   - Detailed Use Case realization with **Interaction** and **State diagrams**.

	Structured analysis	Object-oriented analysis
System boundary	Context diagram	Use case diagram
Functionality	Data flow diagram	Activity diagram Interaction diagrams
Data	Entity-relationship diagram	Class and Object diagram
Control	State diagram	State diagram