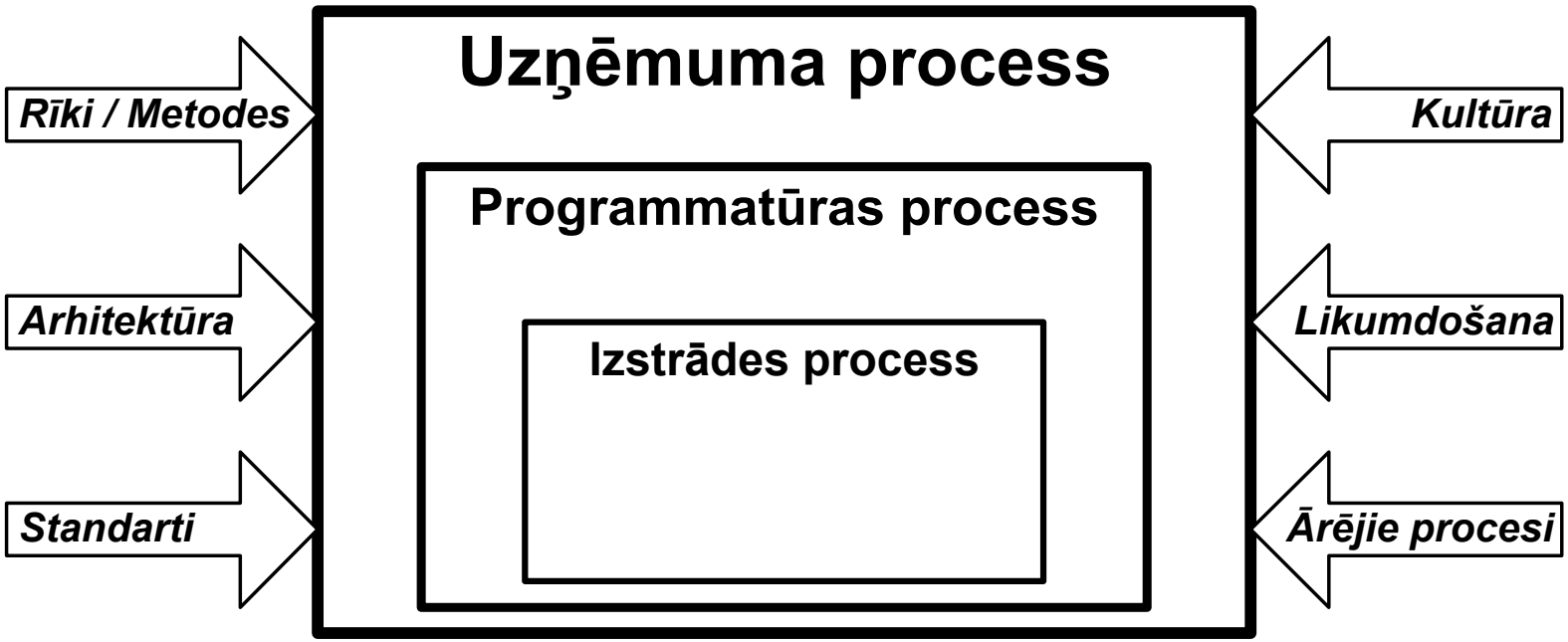
Programmatūras attīstības tehnoloģijas

Procesa uzlabošana
Standarti
Programmatūras attīstība

Dr.sc.ing., asoc. prof. Oksana Ņikiforova
DITF LDI
Lietišķo datorzinātņu katedra
Rīga - LV1048, Meža 1/3, 510.kab., tel.67 08 95 98
oksana.nikiforova@rtu.lv

Programmatūras izstrādi ietekmējoši faktori



Procesa uzlabošana

- Kāpēc programmatūras process ir jāuzlabo?
  - Nepārtraukta cīņa ar programmatūras veidošanu un uzturēšanu
    - Neizpildīta iecere par programmatūras produktivitātes un kvalitātes pieaugumu
    - Problēma ir "neiespējamība vadīt programmatūras izstrādi"
  - Uzlabots programmatūras process
    - Uzlabota programmatūras kvalitāte
    - Gatavs produkts laikā un budžeta robežās
- Problēmas risināšana
  - Standarti
  - Spējas gatavības modelis (capability maturity model - CMM) - 1986

Standarti

- Standarts ir oficiāls dokuments, kas nosaka prasības attiecībā pret dažāda veida objektiem un tehnoloģiskiem procesiem.
- ISO standarti
- LVS standarti, kas ir ANSI/IEEE standartu latviskojums
  - Esošo standartu adaptēšana latviešu valodai ir mazāk darbietilpīga nekā standartu izstrādāšana "no nulles"
  - Esošajos standartos ir ietverta labākā nozares prakse, kas ir vairāku gadu laikā pietiekami noslīpēta

LVS standarti

- LVS 65:1996 Programmatūras kvalitātes nodrošināšanas plāns
- LVS 66:1996 Programmatūras lietotāja dokumentācija
- LVS 67:1996 Programmatūras projekta pārvaldības plāns
- LVS 68:1996 Programmatūras prasību specifikācijas ceļvedis
- LVS 69:1996 Konfigurācijas pārvaldības plāns
- LVS 70:1996 Programmatūras testēšanas dokumentācija
- LVS 71:1996 Programmatūras verifikācijas un validācijas plāns
- LVS 72:1996 Ieteicamā prakse programmatūras projektējuma aprakstīšanai
- LVS 73:1996 Programmatūras vienībtestēšana
- LVS 74:1996 Programmatūras apskates un auditēšanas
- LVS 75:1996 Sistēmas darbības koncepcijas apraksts

http://www.riti.lv/lv/metodnor/prog_doc_ieteik_b.htm

J. Borzovs, Ē. Viļums, R. Čevere

"Ieteikumi programmatūras dokumentācijas komplektam. Metodiskie norādījumi"

Papildus informācija

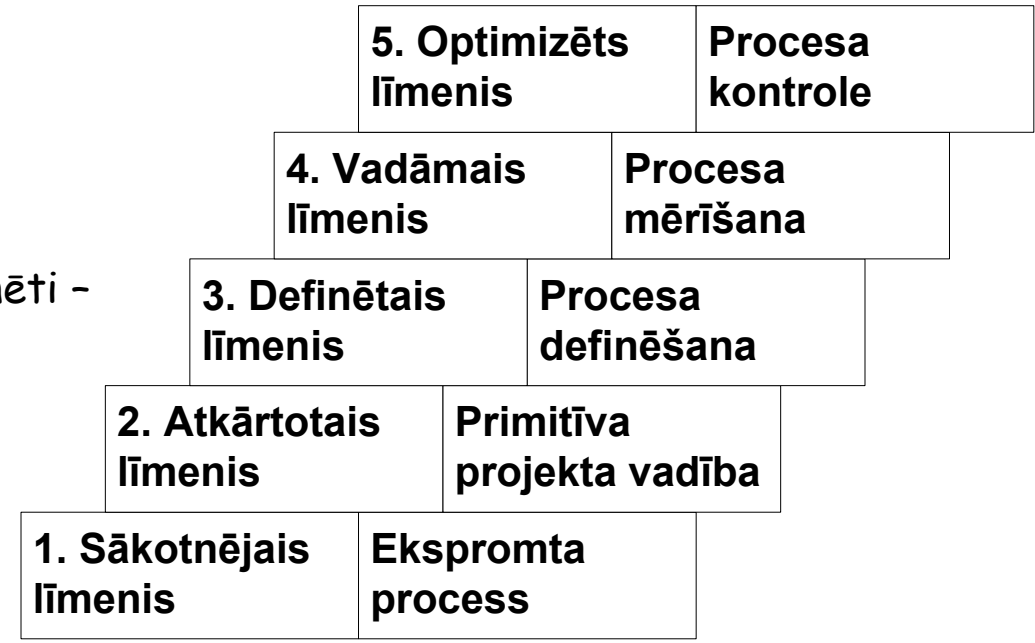
D. Šmite, D. Dosbergs, J. Borzovs

"Informācijas un komunikācijas tehnoloģijas nozares tiesību un standartu pamati",

LU Akadēmiskais apgāds, 2005

Spējas gatavības modelis Capability Maturity Model (CMM)

- Programmatūras procesa uzlabošanas stratēģija
  - Neatkarīga no dzīves cikla modeļa
- Galvenais princips
  - Problēma ir nevis programmatūras izstrādes tehnikas, bet vadība
  - Uzlabota vadība → uzlabotas tehnikas
- Izmaiņu vadība
  - Ieviest izmaiņas pakāpeniski
  - Pieci "gatavības" līmeņi ir definēti – kas ir gatavība?



Gatavības līmenis 1: Sākotnējais līmenis

- Ekspromta (ad hoc) pieeja
  - Neparedzams kopējs process
  - vadība = reaģēšana uz krīzes situācijām
- Tāds ir organizāciju vairākums visā pasaulē
  - Laika un izmaksu pārsniegums
- Veiksme ir pilnīgi atkarīga no darbiniekiem
  - Kā darbinieki rīkojas, tāds ir arī process

Gatavības līmenis 2: Atkārtojamības līmenis

- What is the key to this level?
  - metrics!
- Primitīva (basic) projekta vadība
  - planning, management based on past experience
  - measurements ("metrics")
    - realistic costs and schedules
    - can be used for predictions in next project
  - problems identified → take immediate corrective action

Gatavības līmenis 3: Definējams līmenis

- Software process is fully documented
  - managerial & technical aspects clearly defined
  - continual efforts to improve quality, productivity
  - reviews to improve software quality
  - CASE tools
- Many orgs have made it to level 2 & 3

Gatavības līmenis 4: Vadāms līmenis

- Quality & productivity goals set for each project
- Quality, productivity continually monitored
  - corrective action taken upon deviation
  - rely on statistical quality controls
    - distinguish random deviations & meaningful violations
    - e.g., #faults detected per 1000 lines of code

Gatavības līmenis 5: Optimizējāms līmenis

- Continuous process improvement
- Statistical quality & process controls
- Feedback knowledge from project to next → leads to continued improvement
- How does level 5 differ from level 2?
  - level 2 tries to find & correct faults
  - level 5 orgs practice defect prevention
    - i.,e. ensure there are no faults in the first place

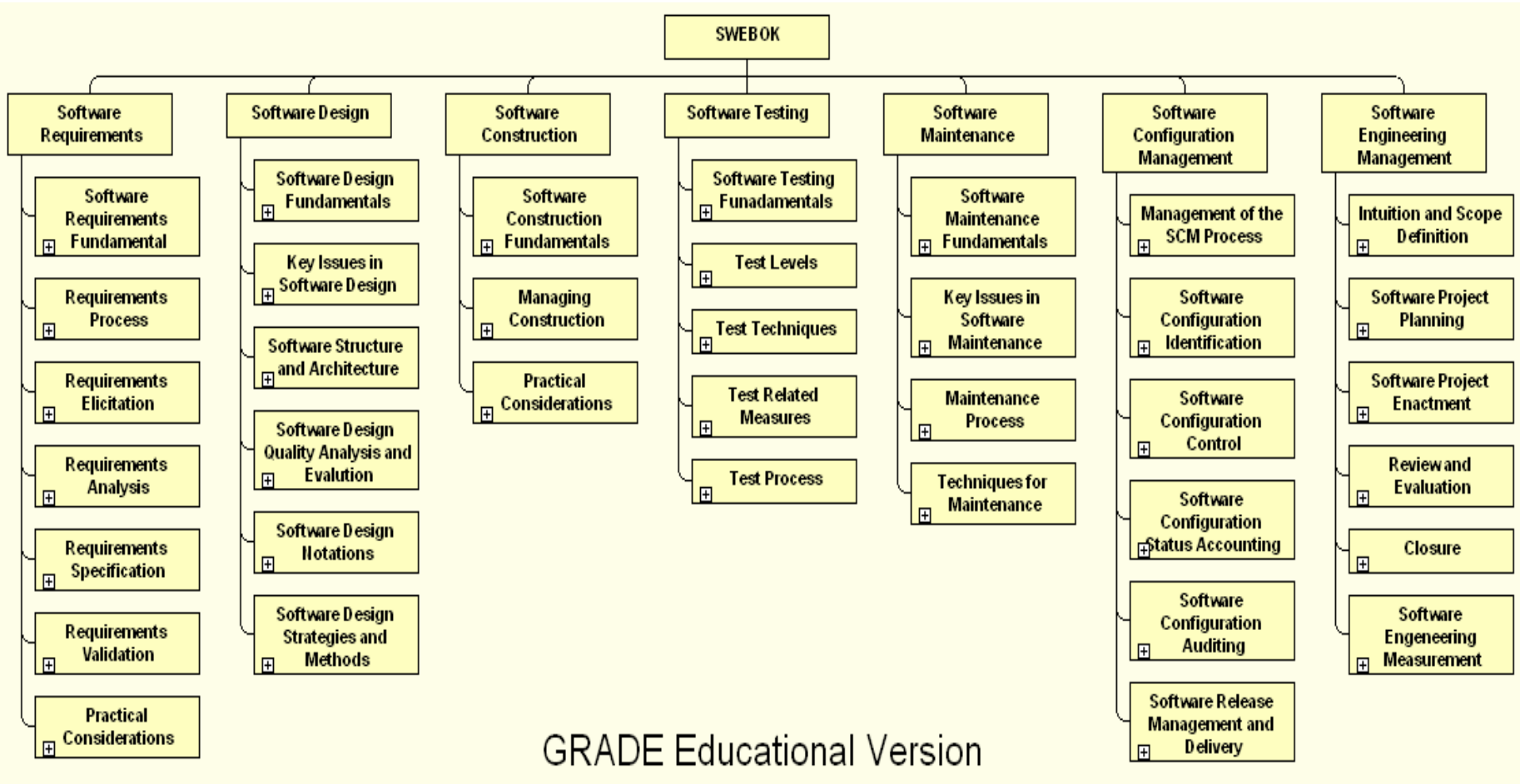
Climbing Levels & Case Studies

- Moving up the process hierarchy
  - 3 to 5 years to level 2
  - 1.5 to 3 years to level 3
  - SEI questionnaires highlight ways to improve process
- Experience
  - original idea: defense contracts only to capable firms
  - CMM has been profitable
    - Hughes Aircraft (Fullerton, CA) spent $500K (1987–90)
      - savings: $2M per year, moving from level 2 to level 3
    - Raytheon moved from level 1 in 1988 to level 3 in 1993
      - productivity doubled
      - return of $7.70 per dollar invested in process improvement

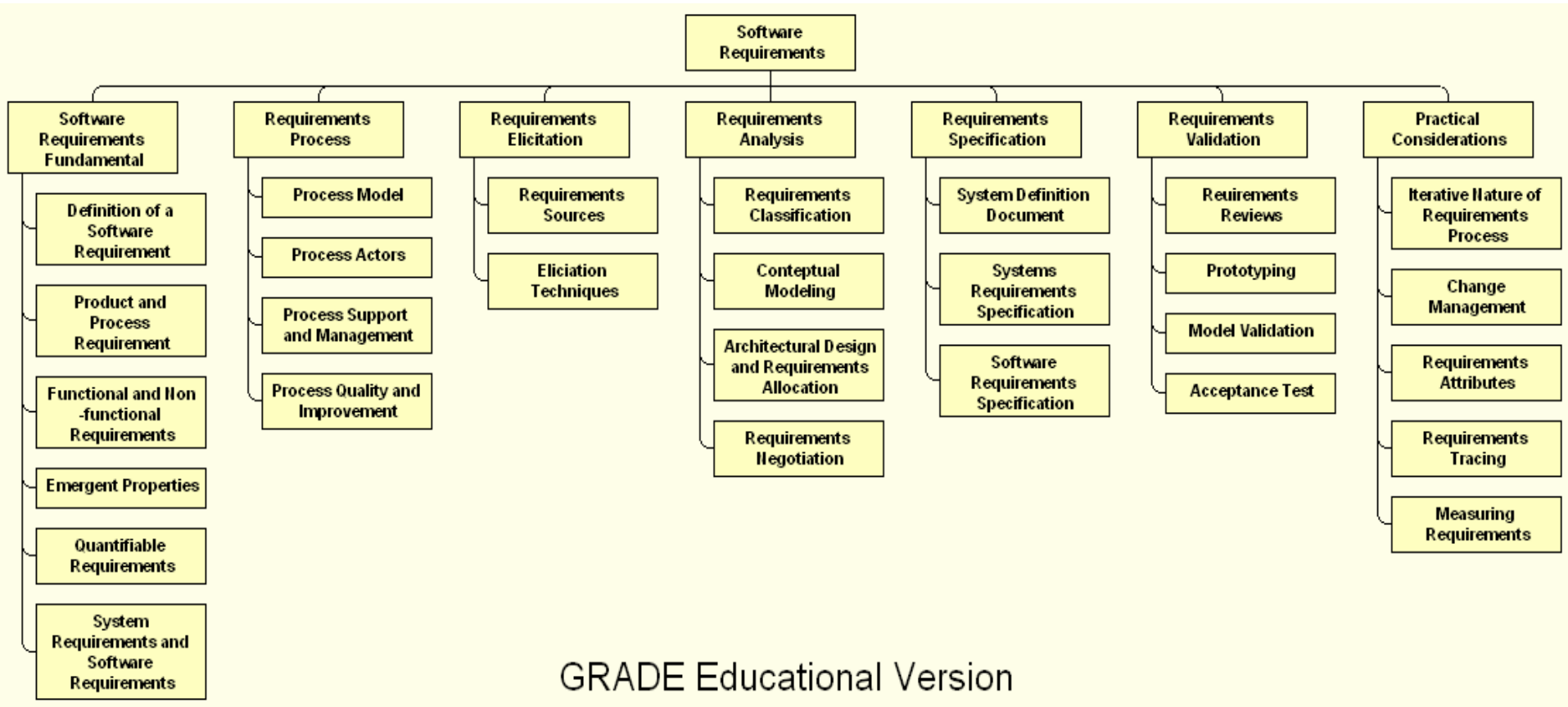
SWEBOK

- Programmatūras inženierijas jomas satura un struktūras apraksts – Programmatūras inženierijas zināšanu kopums - Software Engineering Body of Knowledge *SWEBOK*
- SWEBOK veidošanā daudzu gadu gaitā ir piedalījušies pasaules vadošie speciālisti un programmatūras izstrādes industrijas pārstāvji un tas atzīstams par šobrīd vispilnīgāko un kvalitatīvāko zināšanu un prasmju kopuma aprakstu programmatūras inženierijā.
- SWEBOK mērķis ir sniegt saskaņotu programmatūras inženierijas disciplīnas aptvērumu un šīs disciplīnas detalizētu zināšanu kopuma aprakstu.
- SWEBOK nekoncentrējas uz strauji mainīgajām tehnoloģijām, kaut arī to vispārīgie principi ir aprakstīti atbilstošajās zināšanu jomās.
- Jāievēro, ka SWEBOK aptver zināšanas, kas ir nepieciešamas, bet, iespējams, nav pietiekamas programmatūras inženierijas speciālistam.
- **www.swebok.org**

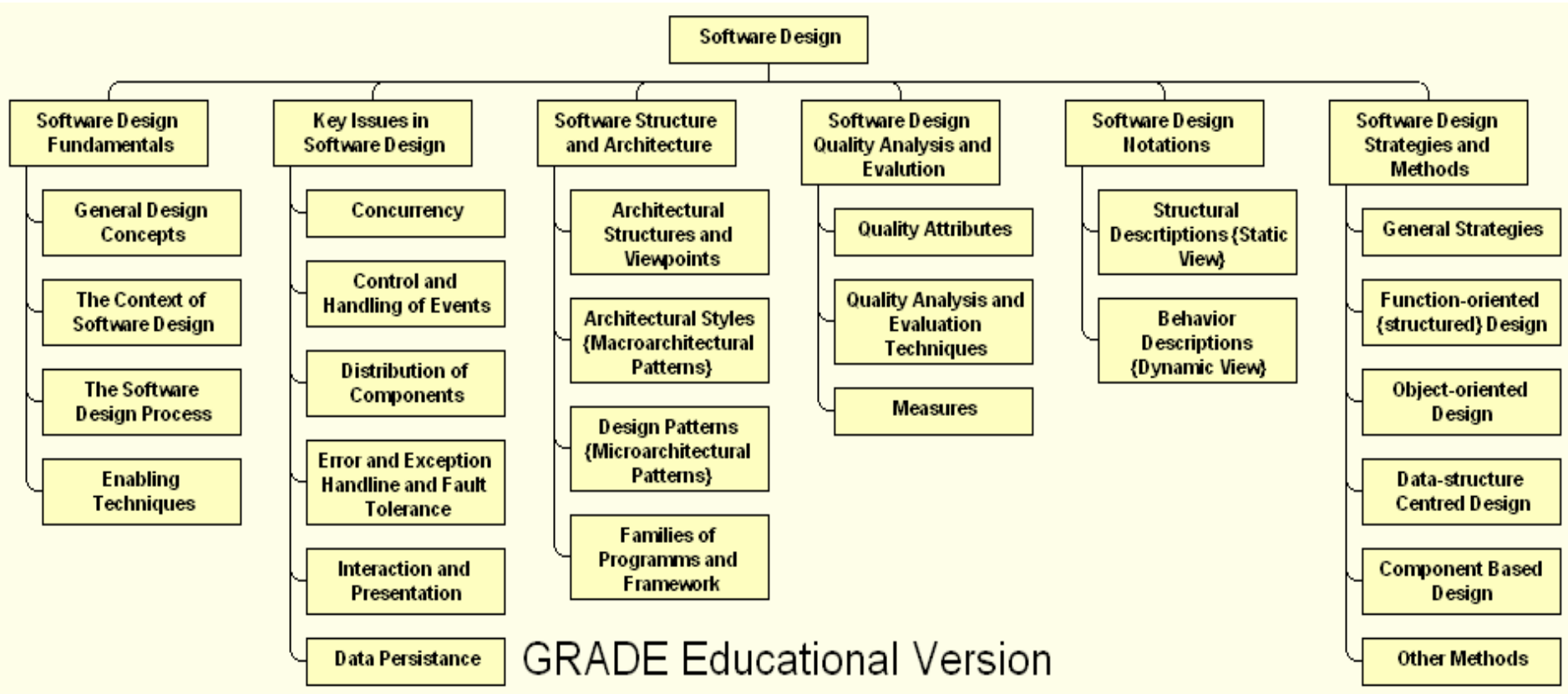
SWEBOK struktūra



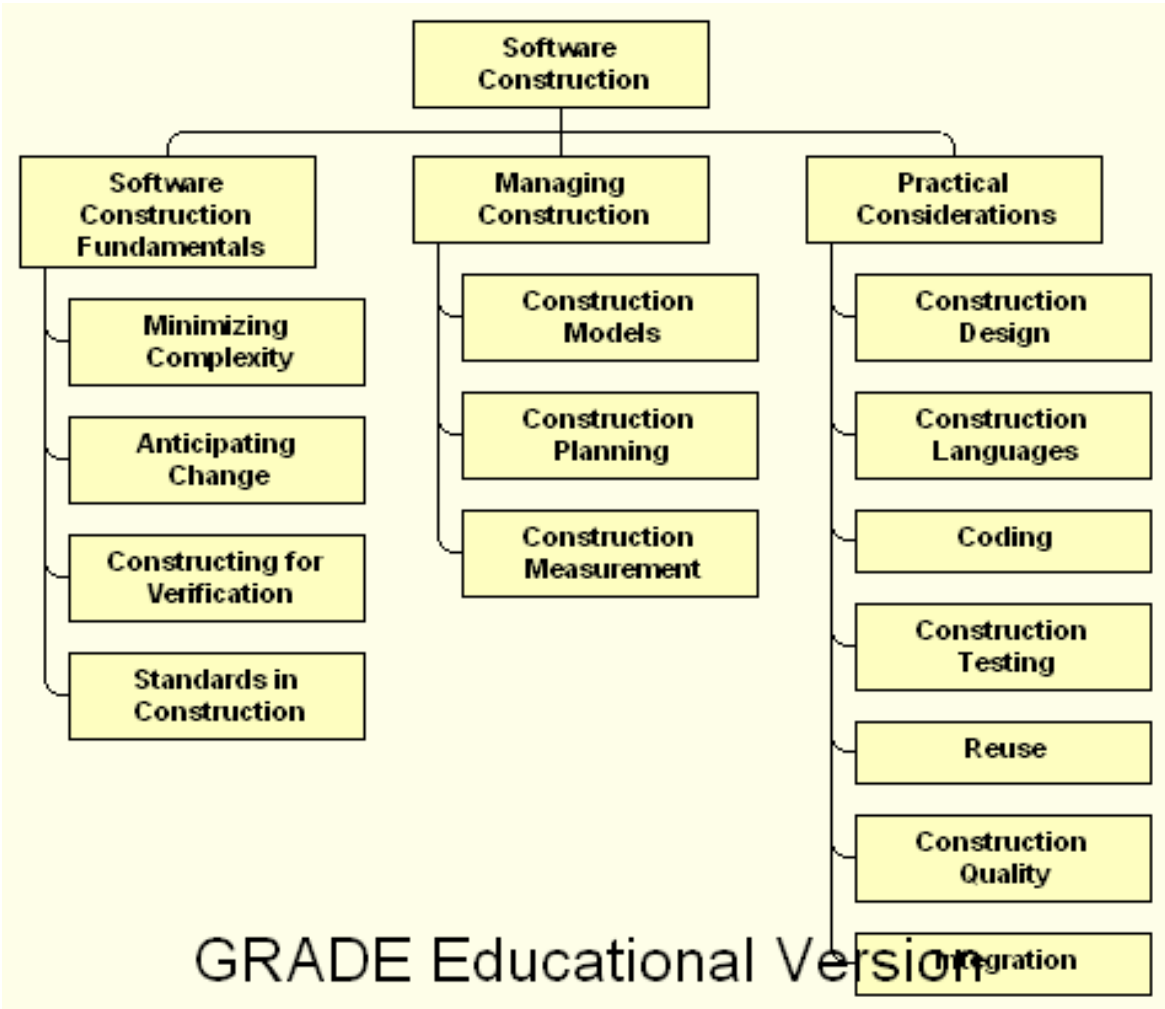
SWEBOK detalizācija "Software Requirements" nozarei



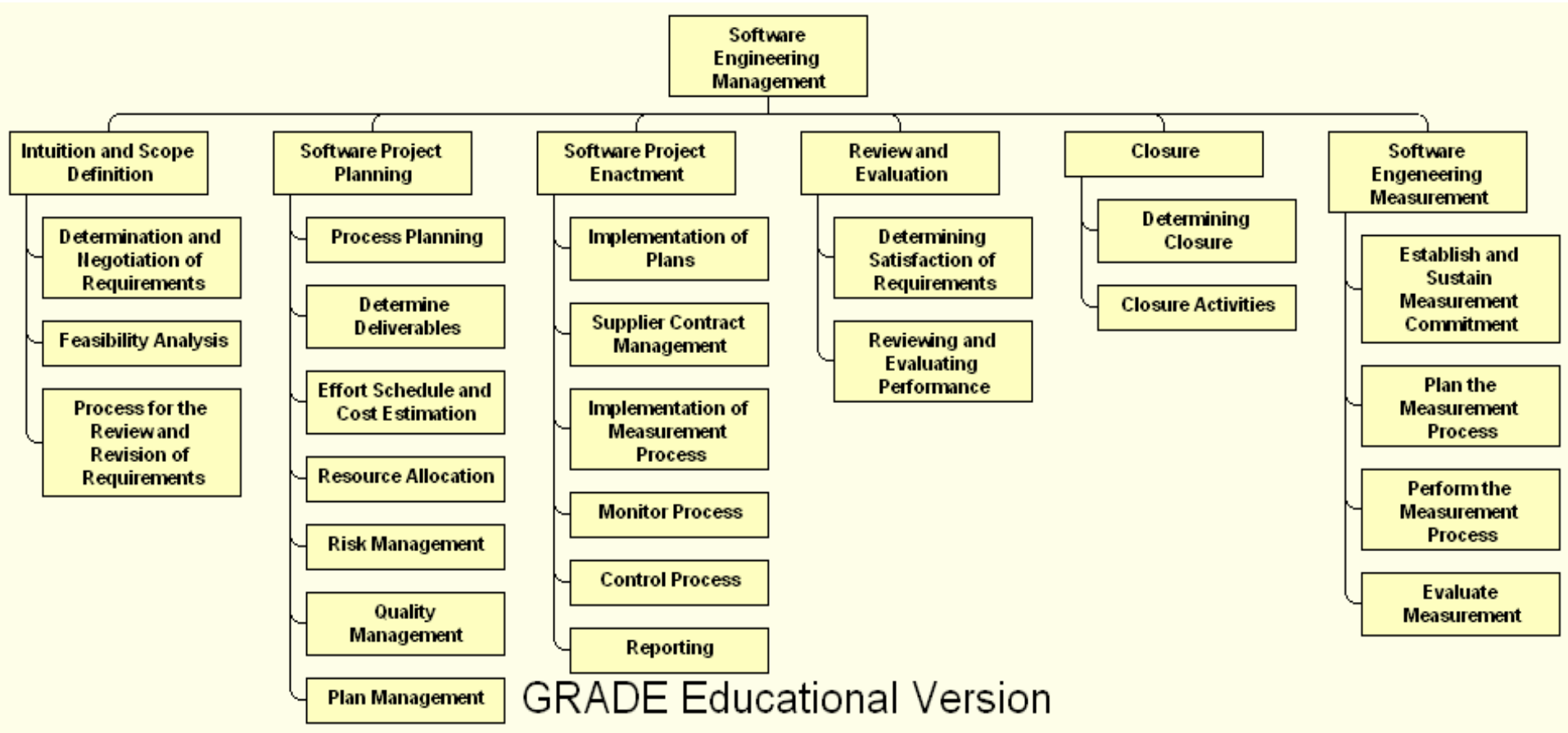
SWEBOK detalizācija "Software Design" nozarei



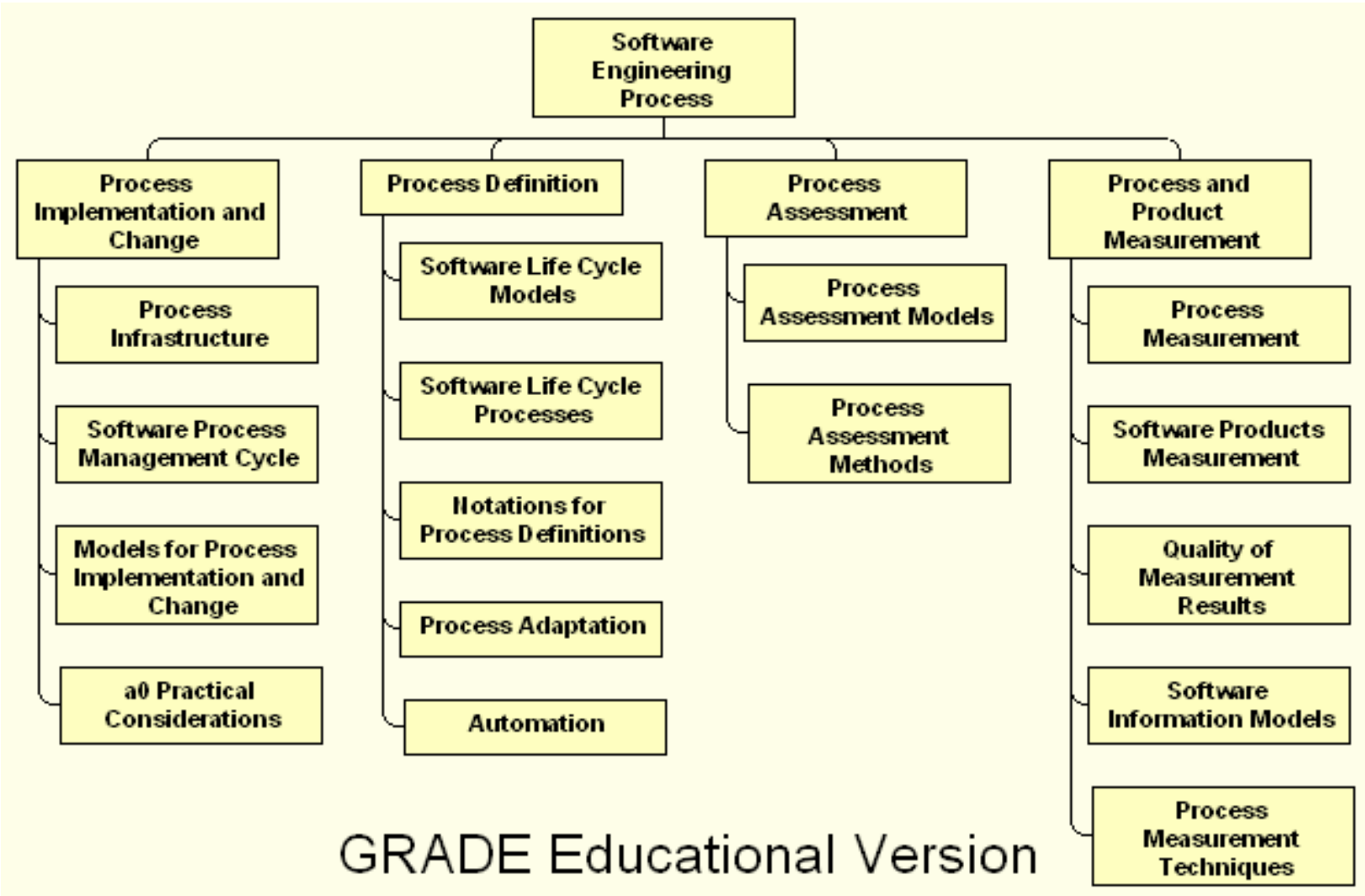
SWEBOK detalizācija "Software Construction" nozarei



SWEBOK detalizācija "Software Engineering Management" nozarei



SWEBOK detalizācija "Software Engineering Process" nozarei



Programmatūras attīstības tendences (1.daļa)

- Programmatūra kā produkts
  - Programmatūras produkcijas ražošana kļuva par lielu izstrādātāju firmu uzdevumu
  - Programmatūras attīstību veicina programmatūras izstrādes firmu konkurence
  - Nekomerciāla produkta daļa samazināšana
- Programmatūras īpašības
  - Programmatūras funkcionalitāte
  - Uztverams, ērts, intuitīvi saprotams un pierasts lietotājam interfeiss
  - Programmatūras apmācības vienkāršība pat iesācējam
  - Programmatūras drošums

Programmatūras attīstības tendences (2.daļa)

- Programmatūras jaudas paaugstināšana
  - Iespēja apstrādāt ātrāk lielāku datu apjomu ar plašāku datu apstrādes funkciju spektru
  - Funkciju integrācija
  - Aparatūras prasību paaugstināšana
- Lietotāja interfeisa ērtums
  - Programmas pieņemšana
  - Standarta interfeisa organizācija
- Standartizācija
  - Programmatūras izstrādātāju mijiedarbības rezultāts – atsevišķu interfeisa elementu, datu formātu standartizācija
  - ISO 9001 – programmatūras produkta standarts, kas nodrošina produkta kvalitāti
  - Programmatūras savietojamība

Produkcijas ieviešana izmantošanā

- WCPA – Web-Centric Production Acceptance
- Izvēršanas (jeb izvietošanas) uzdevums – deployment
- Lai nodrošinātu izvēršanas uzdevuma veiksmīgu izpildīšanu visa programmatūras attīstības procesa gaitā jārūpējas par sekojošas "aptaujas lapas" aizpildīšanu
  - Sistēmas darbības operētājvide:
    - Aparatūra (CPU, memory, disk, network)
    - Operētājsistēma
    - Failu sistēmas tabula
    - Datu bāzu prasības
  - Lietojumsistēmas prasības:
    - Lietojumsistēmas mijiedarbība ar citām sistēmām
    - Ietilpības plānošana
    - Sistēmas pieejamība
    - Lietotāju un IT nodrošinātāju apmācība
    - Lietošanas dokumentācija
    - Administrēšanas procesi
  - Izņēmuma gadījumu procedūras un specializētās prasības:
    - Aizsardzība (security)
    - Sistēmas pārvietošana un modificēšana
    - Kritiskie sistēmas ziņojumi
    - Failu sistēmas dublēšana (backup) un arhivēšana
    - Reģenerācija "nelaimes" gadījumā

Programmatūras produkta izplatīšanas veidi:

- Komercprodukts – lai iegādātos šo produktu sākuma ir jāsamaksā noteikta naudas summa
- Freeware – bezmaksas programmas – tiek izplatītas bezmaksas
  - parasti programmatūras izstrādātāja produkts sākotnēji tika radīts pašu vajadzībām un pēc tam tika pilnveidots izplatīšanai
  - bezmaksas individuāliem lietotājiem un par maksu lietošanai organizācijās
- Shareware – nosacīti bezmaksas programmas – var iegūt un aprobēt bez maksas, sistemātiskai lietošanai nepieciešams samaksāt noteiktu naudas summu

Programmatūras versiju numerācija

- Populāras programmas attīstoties netiek nosauktas ar jauniem vārdiem, bet tiek nosauktas par sākotnējas programmas versijām
- Programmas versiju apzīmēšanā parasti tiek izmantoti decimālie skaitļi (piem. UML 1.3)
  - Būtiskās izmaiņas tiek atspoguļotas ciparos pirms punkta, nebūtiskās izmaiņas un izlabotas kļūdas ciparos pēc punkta
  - Piemēram, sākotnēja programmas versija ir 1.0, versija ar dažiem labojumiem ir 1.2 un būtiskās pilnveidošanas rezultāta programmas versijai tiks piešķirts numurs 2.0
- Eksistē arī programmu numerācija pēc izlaiduma gada, piemēram Windows 98
- Jaunākas produkta versijas tiek pārdotas ar būtiskām atlaidēm (vai pat bezmaksas) klientiem, kas nopirka agrāko programmas versiju

Parazītiskās programmatūras

- Adware – any software application in which advertising banners are displayed while the program is running
- Spyware – any software that transmits information back to a third party without notifying the user