

## System- och användarkrav för programvaruutveckling - 4 dagar

*kurser 218*

- Du får lära dig att**
- Ta fram krav för mjukvaruintensiva system med hjälp av beprövade metoder
  - Skapa en kravmodell baserad på användningsfall
  - Skriva användarberättelser och korta, lättfattliga, fullt utvecklade användningsfall
  - Förbättra och förfina användningsfall med hjälp av ett upprepande tillvägagångssätt
  - Utforma användargränssnitt med hjälp av fabricerade modeller (mock-ups) och utveckla en datamodell
  - Validera krav, hantera förändringar och behålla spårbarheten
- Sammanfattning** Kravinsamling utgör grunden i alla projekt för programvaruutveckling. Under kursen får du de kunskaper och färdigheter som är nödvändiga för att dra nytta av programvarulösningar genom tydligt definierade processer. Du lär dig ange användar- och systemkrav, anpassa processen till projektets storlek samt tillämpa kvalitets- och följdriktighetstest på kravmodellen.
- Vem bör delta** De som utvecklar, designar, testar eller leder utvecklingen av ett programvarusystem. Tidigare erfarenhet av UML är inte nödvändig. De som ansvarar för att identifiera användarkrav, men inte inom programvaruutveckling, bör gå kurs 315, "Utveckla användarkrav".
- Workshops** Omfattande PC-baserade övningar under kursens gång gör att du får uppleva en realistisk användarkravmiljö. Detta ger dig praktisk erfarenhet av att konstruera en modell för användarkrav för programvaruutveckling. Övningarna omfattar:
- Skapa användarberättelser utifrån videoscenarion som placerar dig vid mötesbordet
  - Utforma krav med UML-diagram med hjälp av ett ledande CASE-verktyg
  - Skapa, strukturera och förfina användarfall i en simulerad, realistisk miljö
  - Utveckla skärmmodeller med en gränssnittssimulering
  - Producera en kravdatamodell i UML
  - Validera krav med hjälp av en standardiserad IEEE-checklista
  - Utföra granskningar av verkliga dokument över användningsfall

## System- och användarkrav för programvaruutveckling - 4 dagar

*kurser 218*

### Programvarukravens betydelse

#### Programvaruutvecklingens livscykel

- Definiera och skilja mellan olika kravtyper
- Söka efter kravkällor
- Utvecklingsstrategier

#### Presentera programvarukrav

- Strukturera kravdokumentet
- Kravkomponenter: text, diagram, data

### Strukturera projektet

#### Anpassa metoden till projektets storlek

- Anpassa processen till små, mellanstora och komplexa system
- Skilja lättroliga (agile) metoder från standardmetoder

#### Analysera intressenternas input

- Identifiera och prioritera intressenter
- Få fram krav utifrån dokument med indata
- Upprepa kraven genom samarbete

#### Tillämpa kravprocessen

- Framkallande
- Analys
- Specifikation
- Validering
- IEEE
- SWEBOK
- Unified Process

### Skapa och förfina användarbeskrivningar

#### Skriva användarberättelser

- Skripta användarberättelser och korta versioner av användningsfall
- Uppprepning och progressivt framställande av användningsfall

#### Skapa strukturerade användningsfall

- Användarbeskrivningar som beteendekrav
- Identifiera intressenter och aktörer
- Namnge användarbeskrivningar och ange vad de ska omfatta
- Författa scenarion: huvudscenarion samt alternativ
- Lägga till förhandsvillkor och garantier

#### Upprepa användarbeskrivningar

- Förbättra användarbeskrivningar med intressenter
- Sortera ut vanliga steg
- Upptäcka utökningsscenario
- Verifiera att användarbeskrivningar är fullständiga

#### Organisera användarbeskrivningar

PMI R.E.P.-logotypen är ett registrerat varumärke som ägs av Project Management Institute, Inc.

- Skapa diagram över scenarion med UML
- Välja mellan fri text och formell notation för användarbeskrivningar

### Skapa gränssnittskrav

#### Integrera gränssnittskrav

- Stödja användarbeskrivningar med fabricerade modeller (mock-ups) av användargränssnitt
- Jämföra olika typer av gränssnitt

#### Skapa gränssnittsmodeller

- Skapa skisser (storyboarding) och prototyper
- Utforma gränssnitt med UML-statusdiagram och navigeringskartor

### Datakrav

#### Analysera datakrav

- Utforska användningsfallen och gränssnittet
- Bestämna datastyrda verksamhetsregler

#### Skapa en datamodell för krav

- Återge datamodeller i UML-klassdiagram
- Storheter (Entities)
- Attribut
- Associationer
- Lägga till associationers multiplicitet
- Underhålla ordlistan

### Icke-funktionella krav

#### Samla in icke-funktionella krav

- Erhålla volymetrisk information
- Klassificera icke-funktionella krav med hjälp av FURPS

#### Dokumentera icke-funktionella krav

- Systemets pålitlighet: tillgänglighet, korrekthet och misslyckanden
- Inrikta sig på pålitlighet, tillgänglighet, säkerhet o.s.v.

### Validera krav och skapa testscenarion

#### Utföra kravvalidering

- Åstadkomma välformulerade krav genom validering
- Gå igenom krav med genomgångar
- Verifiera krav med inspektioner

#### Generera användarbeskrivningstest utifrån krav

- Säkerställa att kraven går att testa
- Extrapolera testskript och testscenarion utifrån krav
- Relatera krav till system- och UA-testning

### Hantera ändrade krav

- Utveckla en process för att hantera krav
- Hantera förändringar med hjälp av ett Change Control Board (CCB)
- Bekräfta krav genom en spårbarhetsmatris