

PMM (Process Maturity Metrics)

PMM är en metod för att mäta processmognad i utvecklingsprojekt. I korthet går metoden ut på att man utvärderar sin utvecklingsprocess med avseende på ett antal framgångsfaktorer som finns beskrivna nedan. Då utvärderingen gjorts har man underlag för att se vilka förbättringsåtgärder som ger störst effekt.

Metoden i sin helhet går igenom i Astrakans kurs A4602 Att planera och leda iterativa projekt (<http://www.astrakan.se/4602>).

Allmänt

Framgångsfaktorer är faktorer som samlats in från många projekt. Det är faktorer man pekat på som bidragande orsaker till att projekt lyckats. De 6 första kommer från utvecklingsprocessen UP (Unified Process). De sista 3 är vanor som är självklara men hanteras ofta inte alls eller bara delvis.

Mätetal för framgångsfaktorer

1. CM – konfigurationsstyrning

Att hantera versioner av filer, dokument, källkod, modeller är att ha ordning och reda och veta vad som gäller. Man vet vad man levererar och har kontroll på spårbarhet mellan krav och systemets komponenter. Endast i mycket små projekt med en utvecklare kan man möjligen klara sig utan CM (Configuration Management). En väl implementerad CM plan medför spårbarhet och möjligheter att samla in och mäta på utvecklingsprocessen.

Utvärderingsskala (1 till 5)

1. Ingen CM, filerna ligger på en disk och skrivs över vid varje ny version.
2. Ett CM verktyg har införskaffats och vissa eller alla projektfiler lagras i CM systemet. Det finns inte någon som ansvarar för versionshanteringen utan varje projektmedlem får själv ansvara för att filerna kommer in i systemet.
3. En CM ansvarig har utsetts och en CM-plan finns som beskriver processen. När milstolpar och iterationer passeras märks filerna som hör till leveransen så att man senare på ett kontrollerat sätt kan hämta ut exakt samma filer som tidigare levererats. Filer innehåller filhuvuden där CM systemet automatiskt lägger in versionsnummer.
4. Spårbarhet har upprättats och finns dokumenterad. Det finns stöd för att utifrån ett krav se vilka klasser som realiserar kravet. För en klass kan man se vilka krav som klassen uppfyller. Samt allt under punkt 3.
5. Metriker har identifierats och CM används för att samla in och leverera mätvärden från utvecklingen. Samt punkt 3 och 4 uppfylls.

2. Kravhantering

Det är viktigt att veta vad systemet skall göra vilket normalt beskrivs med hjälp av krav. Med kravhantering avses här även att man har en process att hantera kravändringar och att dessa påverkar planeringen av projektet.

Utvärderingsskala (1 till 5)

1. Ingen kravinsamling sker, man löser problemen som det kommer.
2. Kraven samlas in och dokumenteras i form av användningsfall eller en traditionell kravspecifikation. Andra former av kravfångst kan förekomma i form av XP's Customer Story Task Cards eller features enligt Feature Driven Development.
3. Kraven dokumenteras enligt något av alternativen i punkt 2 samt versionshanteras minst på nivå 3 enligt CM – konfigurationsstyrning.
4. Det finns en dokumenterad process att hantera ändringar i kraven som efterlevs. Varje kravändring medför en analys av hur den påverkar tidplan och resursbehov. Konsekvenser presenteras för beställaren som får ta ställning till om ändringen skall göras och i så fall ta konsekvenserna som presenterades i form av ändrad tidplan och kostnad. Samt punkt 3.
5. Spårbarhet och mätetal samlas in enligt CM punkt 5. I övrigt enligt punkt 4.

3. Iterationer

Iterationer medför bättre kontroll och lägre risk i projektet samt snabbare respons att man håller planen. Den ger korta mål som är hanterbara för varje projektmedlem samt ett tydligt resultat som skall produceras inom en relativt kort period. Jag brukar säga att den motverkar människans lathet och stimulerar till att skapa lite stress i projektet redan från början.

Utvärderingsskala (1 till 5)

1. Iterationer används ej i praktiken.
2. Iterationer har planerats men inga releaser görs vid slutet av varje iteration. Arbetet bedrivs mot målen som satts upp för respektive iteration.
3. Iterationer har planerats översiktligt och varje iteration följs upp samtidigt som nästa iteration planeras. Varje iteration levererar en release som hanteras enligt minst nivå 3 i CM – konfigurationsstyrning.
4. Samma som i 3 men det har lagts in iterationer för felrättning och någon tom iterationer som buffert i planen.
5. Iterationer används fullt ut för att reducera risker och att kontinuerligt leverera systemet med mer och mer funktionalitet.

4. Verifiera kvalit 

Att verifiera kvalit  handlar om att testa och att testa kontinuerligt. Det handlar om att s  tidigt som m jligt identifiera brister i kvalit . Desto tidigare man identifierar brister desto mindre kostar det att  tg rda problemet.

Utvärderingsskala (1 till 5)

1. Inga tester görs.
2. Endast systemtest när systemet skall levereras.
3. Varje leverans testas, d.v.s interna leveranser från varje iteration testas.
4. Unittester görs och automatiseras för varje komponent. Ett felrapporteringsverktyg används för att samla in felrapporter och för att fördela arbetet, ta ut statistik mm.
5. Systemtester byggs upp och automatiseras i form av regressionstester. Testverktyg för belastning och gränssnittstest används för att spela in och spela upp tester. Unittester uppdateras för att framkalla varje fel som beskrivs i felrapporter. Processen Säkerställer att tidigare upptäckta fel aldrig kan återkomma i produkten.

5. Komponentbaserad arkitektur

Denna faktor är viktig för att kunna hantera större system som byggs upp av flera delar som samverkar. Detta medför att varje del kan utvecklas mer eller mindre oberoende av de andra delarna och även testas separat.

Utvärderingsskala (1 till 5)

1. Monolitiskt system
2. Systemet byggs i flera oberoende delar
3. Systemet har medvetet brutits ner i komponenter på ett sådant sätt att de har så liten koppling till andra komponenter som möjligt. Denna uppdelning gjordes tidigt innan man började bygga systemet.
4. Komponenter används för att fördela arbetet i projektet och dessa kan utvecklas oberoende av varandra.
5. Systemet består av oberoende autonoma komponenter som enkelt kan bytas ut och modifieras utan att andra delar av systemet behöver förändras. Det finns en arkitektur i form av ett ramverk som tillhandahåller mekanismer för att lägga till och ta bort komponenter.

6. Visuell modellering

Att göra ritningar på system är lika viktigt som att göra ritningar på hus som man skall bygga. Ritningarna görs helst i UML i och med att det är den mest spridda och använda standarden för modellering av objektorienterade system.

Utvärderingsskala (1 till 5)

1. Inga ritningar tas fram.
2. Skisser och diagram finns för slumpmässiga delar i någon icke standardiserad notation.
3. UML används för att dokumentera systemet.
4. En analysmodell eller domänmodell har tagits fram tidigt för att tolka kraven på systemet samt för att få en tidig förståelse för systemets struktur. De viktiga arkitekturella mekanismerna har ritningar och beskrivits i ett arkitekturdokument.
5. Det finns ett designdokument som med utvalda ritningar beskriver hur alla mekanismer i ramverket fungerar. Under de tidiga faserna har en omfattande modellering av systemet genomförts vilket medfört att projektmedlemmarna har samma idé om hur systemet skall byggas. Samt punkt 3 och 4.

7. Kompetens

Med kompetens avses att man har tagit hänsyn till den samlade kunskap och erfarenheten i projektet och tagit konsekvenserna av att analysera den. Denna dimension är tyvärr ofta en orsak till att projekt misslyckas och det är sällan som man ser att man analyserar vilken erfarenhet som finns och vilken som saknas och behöver tillföras projektet.

Utvärderingsskala (1 till 5)

1. Erfarenhet och kompetens beaktas inte alls när man resurssätter projekt.
2. Roller används inom projektet för att fördela arbetsuppgifter där man gör enklare lämplighetskontroller när rollerna besätts, alternativt så väljer man den roll man tycker verkar vara kul.
3. För varje roll beaktar man projektets behov av erfarenhet och kunskap då den tillsätts.
4. Tekniker och processer som skall användas dokumenteras och resursernas kompetens och erfarenhet inom alla områden inventeras. Underlaget används för att identifiera luckor i kompetens och erfarenhet och resultatet används för att aktivt stärka upp projektet med ytterligare resurser.
5. Hantering av befintlig kompetens och erfarenhet har blivit strategisk. Organisationen arbetar aktivt med kompetensutveckling och placerar resurser för att kunna bemanna projekt på ett optimalt sätt för att lösa det aktuella problemet.

8. Enkelhet

Enkelhet är kanske även det en självklarhet men det ligger ofta mer arbete bakom en enkel lösning än bakom komplicerade lösningar. Det är viktigt att sträva efter enkla lösningar i och

med att systemen trots allt ofta blir komplexa ändå. Det är lättare att följa och förstå en enkel lösning än komplexa lösningar. Det blir lättare att lära upp nya resurser etc.

Utvärderingsskala (1 till 5)

1. Enkelhet beaktas inte.
2. Viss design görs men bara på delarna och inte på helheten.
3. Designmönster används i designen för att förenkla lösningen.
4. Ett ramverk används/utvecklas för att hantera gemensamma stödfunktioner på ett enhetligt sätt.
5. En genomtänkt arkitektur finns och är dokumenterad och dess mekanismer är enkla och kraftfulla. Man strävar att återanvända befintliga lösningar.

9. Hårt arbete med fokus på produkten

Ytterligare en självklarhet men ibland verkar det som om man tror att dyra verktyg och utvecklingsmiljöer skulle medföra mindre arbete, så är inte fallet. Man kan aldrig köpa sig från hårt arbete. Avancerade verktyg kräver ofta mycket arbete i form av att tillägnar sig kunskapen att använda dem, sedan skall man bygga ett avancerat system med hjälp av dem också. Detta handlar främst om en mental inställning att fokusera på att lösa ett problem. Att lägga hårt arbete på att lära sig och införa processer eller verktyg löser inte det egentliga problemet det tar tid och kraft från det egentliga problemet. Att använda dyra avancerade verktyg är en strategisk fråga där man tidigast efter 2 till 3 projekt möjligen kan se en produktivitetssökning. I det första projektet få man räkna med en vesäntlig produktivitetssänkning.

Utvärderingsskala (1 till 5)

1. Inget eller litet fokus på att lösa det egentliga problemet. Man arbetar med språkliga konstruktioner och hur man skall få verktygen att göra det man vill om man ens vet vad man vill att det skall göra.
2. Fokus ha förskjutits från verktyg och språkproblem till vad man skall göra och i vilken ordning. Här arbetar man ofta med att lösa problemet genom att införa en process vilket tar fokus från problemlösningen.
3. Nu har man lyckats få mer fokus på att lösa problemet och känner till de olika stegen man behöver ta i en process för att ta sig mot målet.
4. Iterationer används här för att planera aktiviteter på minst nivå 3. Verktygen är kända och orsakar sällan något problem och utvecklingsprocessen känns även den naturlig.
5. Fokus på att lösa det egentliga problemet och delar ur en eller flera processer används som en verktygslåda för att lösa problem som man stöter på. Alla känner till målet och vet vad dom skall göra. I stort sätt allt arbete är fokuserat på att producera systemet och processen stödjer det arbetet.