Review delrapport 3

No Silver Bullet- Essence and Accident in Software Engineering.

Forfatteren til artiklen beskriver det største problem indenfor Software Engineering. Han bruger en metafor, der skaber en omliggende ramme for problemet artiklen omfatter. Metaforen Silver Bullet referer til det middel, i gamle folkemyter, som en varulv kunne bekæmpes med. Varulven repræsenterer noget der med det samme kan gå fra et stadie til et andet. Noget harmløst og kendt, til et monster uden kendte grænser. – **citat s 181**

Denne Silver Bullet er således et desperat kald fra software verdenen om at nedlægge problemet ved Software Engineering, nemlig hvordan en udviklingsproces i form af softwaren, hvad enten det er nye begreber, mennesker eller fysiske projekter, kan optimeres.

Artiklen tager sit standpunkt i fire hovedområder som forfatteren mener kan mindske problematikken. Disse fire punkter knytter sig til den essentielle del af software Engineering og ikke den acidentale. Forskellen på disse er at finde i en konceptuel mening. Den essentielle del af software Engineering som problemet er at finde i, skal forstås som de problematikker der ligger begravet i hele naturen af Software. Den acidentale del af softwaren betegner de problematikker som opstår gennem produktionen af software, men ikke tilhører den bagvedliggende natur. – **Citat s 182**

De fire punkter der nævnes er:

Udnyttelse af hele software markedets ressourcer

Brugen af prototyper som en del af iterations-design for at skabe software krav

Konceptet med at software gror og udfolder sig organisk

Opdage og udvikle de fremragende konceptuelle designer i kommende generationer

**- Citat s. 180**

Igennem artiklen beskrives den essentielle del, og den acidentale del mere udførligt. De fire punkter bruges som argumenter igennem artiklen og opsættes således at man kan se hvordan de skal løse problemet.   
Vi ikke har en Silver Bullet på nuværende tidspunkt, (vi får næppe en heller) men benytter vi punkterne kan vi komme tættere på målet. ”There is no royal road, but there is a road”. **– Citat s. 181**

Artiklen er meget spændende, da den opstiller et hidtil uopklaret problem. Men problemet viser sig at være det mest interessante problem i vores historie indenfor Software udvikling.

Andre kildeartikler som f.eks. a *rational design process: why and how to fake it* og *Designing for Usability: Key Principles and What Designers Think,* har i et mindre omfang også beskæftiget sig med dette problem. Men de fraviger fra den egentlige problemstilling som artiklen her beskæftiger sig med, og ser på den acidentale del af problem, fremfor den essentielle del.

Også bogen OOSE adresserer problemet implicit i hele dens omfang. Visse af bogens kapitler benytter sig af metodikker og teoretiske virkemidler til at hjælpe med den praktiske udvikling af at software system. Metodikkerne og teorien bag disse virkemidler er meget tæt knyttet til det essentielle Software Engineering problem som denne artikel adresserer. Et eksempel kunne være: recall punkt tre og fire, i samspil. Det vælger vi at sætte op imod en agil arbejdsproces. Her vil både softwaren og menneskerne gro sideløbende som et projekt forløber. Alle ressourcer bliver udnyttet til fulde, og man gennemgår en cyklus – heraf er nu de to første punkter fra artiklen også blevet inddraget i projektforløbet. Alle dele implicit, men de fremmer softwareudviklingen frugtbart, fordi de alle er nødvendige.

Det samme oplever vi også praktisk i vores projekt. Hvordan en arbejdscyklus giver designeren mulighed for at udfolde sig selv, men også udfolde systemet løbende i processen.

Det er ikke muligt konkret at sætte et projekt som vores i reel kontekst til problemet der adresseres i artiklen, da det er på et mere overordnet plan. Som tidligere nævnt, har vi på nuværende tidspunkt ikke en silver bullet til rådighed, men hvis vi udnytter alle vores ressourcer på en hensigtsmæssig måde, vil projektet også blive mest frugtbart.