Metoden

* Använda Action Research tillsammans med Design Science, inspirirat av Technical Action Research (TAR)
* Börja med ett problem, som vi fastställt i en tidigare fas
* Applicera en artefakt på problemet för att försöka lösa delar eller hela
* Reflektera över vad som gick bra och dåligt, mäta

Specifikt:

Steg 1 - Utveckla en plattform som utgör grundkonceptet och grundproblemet så idealt som möjligt, i det här fallet, ett spel i 3d-miljö med de vanliga FPS-elementen. Det vill säga redan här begränsa området till en specifk genre.

Steg 2 - Applicera ett ramverk (artefakten) som separerar Gameplay- och Nätverks-kod så väl som möjligt på plattformen (problemet)

Iterationbasis –

1. Implementera en feature (krav från stakeholders) i plattformen, sedan addera nätverk till featuren. Features identifieras av stakeholders för att sedan väljas slumpmässigt

2. Mäta tidsåtgång, rader kod (Metrics Reloaded plugin till IntelliJ), förändringar i koden mer specifikt (fungerar seperationen).

3. Ta lärdom av fasen och reflektera. Eventuellt förbättra ramverket. Har det uppkommit nya problem och genomfördes det några trade-offs?

Detta sker hela tiden under ideala förhållanden i så kallad laborationsmiljö där mening är att låta ramverket mogna för att sedan bli testat i en mer realistisk miljö. (Uppskalning)

Moget ramverk – Applicera ramverket på ett verkligt problemområde (ett spel specifikt utvecklat för singleplayer)

Valideringsproblem

Under itarationerna kommer problemområdet husera under laborationslika förhållande där utvecklarna är fria att manipulera både ramverk och plattform för att passa ett specifikt syfte.

Vi som utvecklare tar på oss flertalet roller (eller hattar) och är därför benägna till att bli påverkade av våra andra jag.

Kanske säger mätningarna inte allt för mycket om effektiviteten i sig. Inte heller tidsåtgången då den kommer variera från feature till feature.