Jag vill först säga att jag tyckte föreläsningen var mycket intressant och jag tog till mig mycket av vad som sades. Om jag inte glömt bort att titta på föreläsningen hade jag säkerligen tillämpat mycket mer om vad som sades i mitt projekt. Jag kan också nu säga att det inte blev mycket av ett typiskt referat, utan mer lösryckta saker som togs upp under föreläsningen med egna tankar och erfarenheter...

Vi som människor har mycket lätt att överskatta risker helt i onödan, i många fall är riskerna inte alls så stora som man tror. Med risker kommer också förmåga på tal. Det händer ofta att nybörjare på ett område överskattar sin förmåga (vilket jag märkt av...), medan mer erfarna personer underskattar sin förmåga.

Jag fastnade speciellt när just det här med ”att zona” (vilket som nybörjare på ett område inte är lätt), eller flow som jag föredrar att kalla det, kom på tal. Det är ett enligt mig mycket intressant fenomen som jag gjort arbete på som mitt projektarbete under min gymnasietid. Just varför vi sollar har en teknisk orsak, hjärnan klarar bara av att hantera ett ex antal saker i taget. När stor koncentration läggs på en uppgift tas mycket av hjärnans kraft, och därmed sollas en del saker bort (jag kan skriva så pass mycket mer om detta, men det var inte huvudtemat på föreläsningen).

I ungefär samma stuk har vi det här med hur vi väljer beroende på ett antal faktorer, till exempel exemplet med Rom och London, vilket kan tillämpas även på utveckling då man kanske inte alltid väljer den bästa lösningen utan man väljer det som känns mest ”lockande”(Boken ”Thinking, Fast and Slow” av Daniel Kahneman tar upp detta väldigt bra).

Huvudtemat på föreläsningen var ju ändå om testdriven utveckling, vilket jag tyckte var mycket intressant och hoppas jag kommer att använda mig av. Utvecklingsprocessen baseras på tester och bara funktionalitet som har med just testerna att göra skall implementeras, exempel get och set funktionalitet på sina ställen. Enligt testet behövs kanske bara det ena, och då skall bara det implementeras. Genom till exempel ett interaktionstest kan man också dra slutsatser om precis vilken funktionalitet som behövs, och i vilken ordning den bör implementeras. Detta leder också, tillsammans med refaktorering, till bättre kod genom bättre struktur och onödig implementation osm inte görs. Testerna gäller den publika delen av koden, medan de privata delarna kan justeras hur mycket man vill utan att de påverkar testerna, i alla fall till viss mån där någon privat del inte påverkar en publik del. Att också kommentera koden är ett åerkommande dilemma, men ett bra tips togs upp. Kommentera med att den här biten löser det här testet, detta ger mycket bättre tydlighet om man varit borta från koden ett tag än ett par utryckta kommentarer.