Beslut och vägskäl

# NIO vs. IO

När det var dags att börja skriva på nätverksimplementationen ställdes vi kanske inför vårat största val hittills nämligen vilket av Javas APIer för nätverk som vi skulle använda. Skillnaden dem emellan är kanske, rent teknisk, inte allt för stor – [1] IO är Stream oriented samt blockerar sina trådar, medan NIO är Buffer oriented, non blocking samt använder sig av Selectors. Rent kodmässigt är det dock ett val som kommer ha stor inverkan, och därför tog det oss ett tag att slutligen bestämma oss.

Egentligen, efter lite efterforskning så borde vi insett att IO torde vara det rätta valet för våran applikation. Då NIO verkar vara att föredra och framförallt lyser när [1] applikationen ska handskas med många (hundra eller tusentals) anslutningar samtidigt medan IO fungerar bäst när applikationen ska handskas med få anslutningar som skickar mycket data hela tiden, vilket är så nära på en beskrivning av våran applikation som man kan komma.

Det som dessutom talar för IOs favör är att det generellt sett verkar vara det som rent APImässigt är enklast att använda, både enligt oss efter idogt prototypande och enligt andra [2].

Detta grusade dessvärre våra planer att använda KryoNet [3] som grundpelare, då det är det biblioteket som vi tidigare använt oss utav.

*Länkar aktiva i och med 4/3 2014*  
[1] <http://tutorials.jenkov.com/java-nio/nio-vs-io.html>  
[2] <http://stackoverflow.com/questions/12892536/how-to-choose-java-nio-vs-io>   
[3] <https://github.com/EsotericSoftware/kryonet>

# UDP vs. TCP

Egentligen var tanken från början att helt ignorera TCP och skriva funktioner över UDP för att få samma funktionalitet. Men när arbetet framför oss är så pass mycket och ibland överväldigande så valde vi ändå att implementera funktionalitet för TCP, så att vår klient och server kan skicka viktiga meddelanden mellan varandra. Till exempel när en ny klient vill ansluta.