Love Ottosson Huskvarna loveo@kth.se

Teknisk Reflektion

Innan projektet så var mina kunskaper inom hårdvara allt annat än strålande. Jag hade aldrig gjort något mer praktiskt än labbarna i datorteknik eller läst om strömmar och spänning i elkrets. Därför blev det mer naturligt att jag ägnade mig mer åt programmering till Arduinon och i java än åt de andra delarna som att snickra och bygga kretsar.

Jag tror att det var ett ganska smart val med tanke på att jag har hållit på med programmering mycket längre än de andra i gruppen och behövde därför inte lägga ner lika mycket tid på t.ex. mottagardelen i java.

Det viktigaste jag har lärt mig från det här projektet är att aldrig lita på hårdvara. Om det står att en viss komponent ska kunna göra en viss sak med en viss precision, så ska man vara kritiskt till om komponenten verkligen klarar detta. Vi antog flera gånger att vissa komponenter skulle fungera felfritt och att de skulle göra precis som vi ville. Hade vi istället undersökt vad komponenter verkligen gör eller klarar av och hur andra har lyckats använda den tidigare så hade många av våra problem inte existerat.

En annan källa till många av våra problem var snålheten hos gruppen. Vi var rädda att spräcka budgeten så vi försökte komma undan på det billigaste sättet varje gång, vilket ledde till att de komponenterna vi köpte kanske inte var de bästa varje gång. Tipset till nästa gång är alltså att inte snåla på de viktigaste komponenterna som i vårt fall var magnetometer och metalldetektor.

Nog med saker som gick dåligt. Självklart så hade vi många lösningar som jag hade var med och kom på. Det bästa med många av dessa lösningar är att de är så pass enkla att till och med vi lyckades få de att funka. Det som jag tycker var den bästa lösningen var hur vi (jag och Anton) löste problemet med kodhärvan. Vi hade skrivit kod för alla olika delar som drivlina, interrupts, pingsensorer osv. Men dessa kodsnuttar var omöjliga att implementera till något större program som skulle efterlikna en minletare. Så vad gjorde vi som var så smart?

Vi skrev upp alla hårdvarukomponenter på tavlan och skrev upp funktioner som vi förväntade oss kunna använda mot alla komponenter som t.ex. driveForward på hjulen eller getDistance på pingsensorerna. Efter vi hade skrivit upp alla funktioner så skapade vi klasser till Arduinon som innehöll precis dessa funktioner. En klass för varje komponent. Efter så ritade vi upp ett minfält på tavlan och simulerade en robot som åkte. Medans roboten åkte framåt så skrev vi upp funktioner som den skulle kalla på som t.ex. checkForMine, driveDistance, checkHeading osv. Efter att roboten åkt hela fältet så hade vi alla funktioner som någonsin skulle bli kallade på. Då var det väldigt enkelt att para ihop funktioner som tillhörde samma klass. Vi skapade klasser med namn som gick att förstå och lade funktionerna i rätt klass. Sen började arbetet att fylla i funktionerna som mest innebar att kalla på hårdvarufunktionerna som vi skrivit tidigare. I slutändan fick vi en kod som var extremt lätt att följa och uppstod det ett fel med roboten så visste vi precis vart i koden som detta fel fanns.

Det var enligt mig det absolut bästa draget då vi gick ifrån kodhärva till färdig körkod för roboten på en förmiddag. Hela processen kändes väldigt ingenjörsmässig och resultatet var fantastiskt. Jag tror

att den här metoden går att använda på nästan alla projekt som liknar detta och jag kommer använda samma metod i fortsättningen.

En annan enkel och fyndig lösning som hjälpte oss väldigt mycket med den trådlösa kommunikationen var att skicka varje meddelande 5 gånger. Löjligt lätt att implementera på Arduinon och inte jättesvårt på java-sidan. Det vi behövde göra på java-sidan var att spara ner meddelanden med samma ID i en grupp och kolla hur många meddelanden som var likadana. Sen valde vi bara det meddelandet med rätt längd och mest ”träffar”. Genom att skicka allt 5 gånger så undvek vi nästan alla problem med brus och störningar på 433MHz bandet.

Det är just den typen av lösningar som jag tycker vi gjorde väldigt bra under projektet. Att hitta så enkla lösningar som möjligt. Båtlaget pratade om ”Keep It Stupid Simple”, vilket jag starkt rekommenderar. Att tänka lite extra och inte bara plocka första lösningen som dyker upp kan spara väldigt mycket tid och underlätta jobbet. Vi löste detta genom att tänka i grupp varje gång vi skulle bestämma något och alla fick komma med sitt egna förslag varav det bästa valdes.

Om det är några saker jag kommer tänka på till nästa projekt så är det att:

- Researcha ordentligt om det är saker jag inte hållit på med förut.
- Lägg tid på att hitta så enkla lösningar som möjligt.
- Snåla inte på kvalitet.
- Städa upp koden med ovannämnd metod.
- Var kritisk till specifikationerna för hårdvaran.