

**LINKÖPINGS UNIVERSITET**  
**Institutionen för datavetenskap**  
**Uppgift**

**5 maj 2010**

Visning av webbtillgänglig data på särskild, därför avsedd hårdvara

## Den röda lådan

**Name** David Hall  
**E-mail** `davha@ida.liu.se`

## 1 Background

Forsknings- och utvecklingsavdelningen på Bonnier AB arrangerade den 24 mars 2010 en 24-timmars hackdag under namnet *Bonnier Hack Day*<sup>1</sup>. Syftet med denna dag var att inbjuda Bonnier-ägda företag (men även utomstående) att använda sig av varandras API-er till att skapa nya eller mer intressanta tjänster. En jury valde ut det bästa bidraget. Jag var inbjuden av Per Åström på TV4.se för att hjälpa till med ett av hans projekt: att visualisera data användbart för de anställda på TV4.se.

Alla hyperlänkar på TV4.se leds via ett speciellt skript, som ingår i systemet Linkpulse<sup>2</sup>, som upprätthåller statistik om antalet klick och med information om sidan man kom ifrån och annan manuellt tillagd information (exempelvis placering av länken på sidan) kan användas för att följa användarbeteenden på webbplatsen. För att ge en ögonblicksbild av graden av aktivitet på de olika webbplatserna (som tv4.se, fotbollskanalen.se, recept.nu m.fl.) som bara inte var antalet sidvisningar skulle antalet klick de senaste 15 minuterna visas tydligt på redaktionen.

Åström ville ha något annorlunda än bara ytterligare en datorskärm som visade siffror och något mer iögonenfallande och med ljus. Lösningen skulle helst också vara fristående, utan att någon extra hård- eller programvara utanför vår konstruktion skulle behövas.

## 2 Data format

Linkpulse-verktyget erbjuder en datafil i RSS-format (som är ett XML-baserat format för att beskriva uppdateringar på webbplatser och som blivit populärt för bloggar och nyhetswebbplatser under det senaste decenniet). Det finns åtminstone två konkurrerande RSS-standarder som är delvis kompatibla. Linkpulse levererar sitt data i form av RDF Site Summary<sup>3</sup> som uttrycker data i form av RDF<sup>4</sup> serialiserat i XML.

Listing 1: del av RSS-fil från Linkpulse

```
<item rdf:about="http://www.fotbollskanalen.se\  
  /1.1536086/2010/03/02/\  
  arsenal_tar_med_sig_klockan_fran_highbury">  
  <title>Arsenal tar med sig klockan från Highbury - Premier \  
    League - allt om engelsk fotboll - fotbollskanalen.se</title\  
  >  
  <link>http://www.fotbollskanalen.se/1.1536086/2010/03/02/\  
    arsenal_tar_med_sig_klockan_fran_highbury</link>  
  <description>avgclicks:33;stateclicks:75;totalClicks:122</\  
    description>  
</item>  
  
<item rdf:about="http://www.fotbollskanalen.se\  
  /1.1534692/2010/03/01/\  
  leonardo_stoppar_ronaldinho_far_inte_langre_sla_milans_straffar\  
  ">  
  <title>Leonardo stoppar Ronaldinho - får inte längre slå Milans \  
    straffar - Serie A - Allt om italienska ligan - \  
    fotbollskanalen.se</title>
```

<sup>1</sup><http://bonnierhackday.blogspot.com/2010/02/about-bonnier-hack-day.html> hämtad 2010-05-04

<sup>2</sup><http://www.onsite.no/linkpulse/> hämtad 2010-05-04

<sup>3</sup><http://web.resource.org/rss/1.0/spec> hämtad 2010-05-04

<sup>4</sup><http://www.w3.org/RDF/> hämtad 2010-05-04

```
<link>http://www.fotbollskanalen.se/1.1534692/2010/03/01/\
leonardo_stoppar_ronaldinho_far_inte_langre_sla_milans_straffar\
</link>
<description>avgclicks:15;stateclicks:74;totalClicks:45</\
description>
</item>
```

Listning ?? visar exempel på informationen för två länkar i filen från Linkpulse.

Filen erbjuds på Linkpulses webbserver och laddas alltså ner över HTTP. Varje webbplats hade varsin RSS-fil och för att visa det totala antalet klick behövde parametern stateclicks plockas för var och en av länkarna och alla dessa summeras. Filen var 250 KiB stor vilket inte skulle innebära något större problem på en vanlig PC men väl i en mer begränsad hårdvara.

### 3 Arduino

Arduino är en open-source-plattform för prototyp-utvecklande av elektronik. Den uttalade målgruppen är konstnärer, designers eller de som har som hobby att skapa interaktiva miljöer.<sup>5</sup> Arduino kan styra ljus, motorer och annan elektronik och är baserad på en mikrokontroller för vilken man skriver i ett C-liknande programmeringsspråk som kompileras på en vanlig PC innan det förs över till Arduinon.

Rent fysiskt består en Arduino av ett kretskort med mikrokontroller, nät-del, USB-port för koppling till dator samt ett antal digitala in- och utgångar och analoga ingångar. Dessa in- och utgångar kopplas sedan till kringliggande elektronik. Det finns även ett antal färdiga *shields* vilka enkelt kan kopplas in i en Arduino utan någon ledningsdragning.

En sådan shield är en Ethernet-shield<sup>6</sup> som innehåller en krets som man hittar i ett vanligt nätverkskort men även en egen IP-stack implementerad i hårdvara eftersom hårdvaran i en Arduino är för begränsad för att orka med att driva det.

Hårdvaran i den Arduino vi använde, Duemila Nove<sup>7</sup>, består av en ATmega168-kontroller med klockfrekvens på 16 MHz som internt har 32 KiB flashminne, 1 KiB SRAM arbetsminne och 512 bytes EEPROM-minne.

### 4 Visning

Som utenhet valde vi att använda oss av relativt stora segmentsiffror av den typen som finns i väckarklockor (se figur ??) Tre byggsatser från Velleman (K8063<sup>8</sup>) à två siffror var beställdes men enbart två byggsatser anlände. Dessa segmentsiffror kommunicerade med ett seriellt protokoll över RS-232 till vilka vi tänkt koppla Arduinon direkt. Utöver siffrorna (som alltså skulle vara summan över antalet klick) ville vi också visa om trenden var på uppåt- eller nedåtgående samt vilken webbplats det just nu visades statistik för.

<sup>5</sup><http://arduino.cc/> hämtad 2010-05-04

<sup>6</sup><http://www.arduino.cc/en/Main/ArduinoEthernetShield>, hämtad 2010-05-04

<sup>7</sup><http://arduino.cc/en/Main/ArduinoBoardDuemilanove> hämtad 2010-05-04

<sup>8</sup>[http://www.velleman.eu/downloads/0/k8063\\_serial\\_display.pdf](http://www.velleman.eu/downloads/0/k8063_serial_display.pdf) hämtad 2010-05-04



Figur 1: Segmentsiffror (och andra tecken) som kan visas med byggsatsen. Bild från Velleman.

## 5 Styrning

Det tänkta användarfallet för vår apparat innefattade att man skulle kunna välja vilken webbplats man såg statistik för. Alltså behövdes någon sorts inenhet. Eftersom vi såg fördelar med att inte begränsa oss till ett fixt antal webbplatser från början vore det bra med ett användargränssnitt som inte bestod av en knapp för vardera webbplats. Ett sätt skulle kunna vara att styra apparaten med två knappar, en för att byta webbplats nedåt och en uppåt. Eftersom det kit som medföljde Arduinon innehöll en potentiometer valde vi istället att basera gränssnittet på detta. Fördelen blev att man enbart har en kontroll att styra med samtidigt som det fortfarande är intuitivt att välja webbplats (jämför med frekvensväljaren gamla tiders radiomottagare).

## 6 Parsning av XML

RSS-datat vi skulle läsa in och behandla var runt 250 KiB stort och Arduinon har ett arbetsminne på 1 KiB fanns det inte möjlighet att snabbt läsa igenom och summera datat. Vid inledande tester kom vi fram till att inläsningen av filen skulle ta mellan två och tre minuter. Pratigheten i RSS- och XML-formaten är utformade för läslighet av människor. I tillägg till detta innehåller filen data som inte är intressant för oss i just denna tillämpning, t.ex. länkens URL och titel. I en normal PC med internminne betydligt större än filens storlek skulle tiden att läsa ut den intressanta informationen inte varit någon större faktor.

För att snabba på processen fick vi leverantören av Linkpulse att modifiera filen så att den fortfarande följde RSS-standarden men att den för oss intressanta informationen låg överst, färdigsummerad, i filen samt att viss redundant information togs bort.

Parsningen av XML skedde inte helt standardenligt heller. För att hålla koden så snabb och enkel som möjligt så skrev vi koden så att den letar efter förekomsten av `<description>` och sedan plockade ut sifvervärdet efter *stateclicks* med hjälp av biblioteket `TextFinder`<sup>9</sup>. Det finns även exempel på XML-parsning i Arduino<sup>10</sup> men så länge inga andra `description`-fält dyker upp (i t.ex. ett element med samma namn fast i annan namnrymd) i datafilen så är det ingen mening att ödsla processorkraft och minne på korrekthet i denna tillämpning.

## 7 Gränssnitt mot skylt

Skyltarna kommunicerade, som tidigare angivet, över ett seriellt protokoll med RS-232-anslutning. Varje skylt kan tilldelas en adress mellan 0 och 255 och sedan kopplar man ihop så många skyltar som behövs. För att bestämma vad som ska

<sup>9</sup><http://www.arduino.cc/playground/Code/TextFinder> hämtad 2010-05-05

<sup>10</sup><http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1253634941> hämtad 2010-05-05

visas på skyltarna skickar man några bytes data innehållande adress, kommando, parameter och en checksumma. Beskrivningen av databladet nämnde inte att adressen 0 inte fungerade exakt som de andra adresserna utan det visade sig vara någon form av adress för att kommunicera med alla skyltar oberoende av adress, något vi upptäckte när allt väl var inkopplat.

Väl på dagen insåg vi att Arduinon inte kunde kommunicera direkt med skyltarna eftersom Arduinons seriella gränssnitt arbetar med TTL-logik på 5 volt medan skyltarna förväntade sig kommunikation över RS-232 på 12 volt. Som tur var hade inte Arduinon förstörts vilket antagligen hade varit fallet om vi hade kopplat in dubbelriktad kommunikation och matat Arduinons seriella ingång med 12V. Därför blev det till att skynda sig till ELFA och köpa ytterligare komponenter och bygga upp elektroniken för att anpassa spänningen baserat på Maxims krets MAX232<sup>11</sup>. När den var inkopplad fungerade dock kommunikationen okej (förutom problemet med adressering enligt ovan).

Programvaran vi skrev fick rutiner för att skicka rätt kommando med korrekt uträknad checksumma och med två omsändningar (för att alla skyltar garanterat skulle ha uppfattat meddelandet). Eftersom Arduinon bara har en seriekanal som både matas ut/in på de digitala ut-/ingångarna (1 och 2, vilka var de vi hade kopplat till skyltarna) och USB-gränssnittet så hade vi inget annat sätt att skicka debug-data än via samma kanal. Eftersom vår debugdata saknade rätt checksumma så var risken för att dessa meddelanden felaktigt skulle tolkas som kommandon för skyltarna ytterst liten.

## 8 Programmering

Det C-liknande språket gjorde det enkelt att abstrahera bort saker som kommunikation med displayen, visa tal och läsa indata från potentiometern. I övrigt använde vi oss av färdiga bibliotek för nätverkskommunikation (dels Ethernet-biblioteket men även ett bibliotek för att över DHCP tilldela apparaten en IP-adress. Även det tidigare nämnda TextFinder-biblioteket användes och underlättade hantering av läsning av data.

Ett problem med att programmera på denna lägre nivå om man vant sig vid programspråk på högre nivå med stöd för flera trådar och som körs på snabbare processorer är att möjligheten till snabb respons snabbt blir en kritisk faktor. Håller apparaten på att ta emot data över nätverket så kan jag inte göra annat samtidigt. Använder man sig av en funktion som `delay` får man vara beredd på att processorn låses upp under den tiden och man bör fundera på att lösa sådant på annat sätt. Vår kod håller själv reda på hur lång tid som förflutit sedan förra uppdateringen av datat men tillåter att man använder potentiometern för att byta vilken webbplats man visar data för och uppdaterar i så fall skylten direkt. I vårt fall är det på bekostnad av att koden riskerar sluta fungera när den interna klockan slår över till noll igen, vilket är något oklart när exakt det sker. Dokumentationen säger efter cirka 50 dagar<sup>12</sup>, andra hävdar att det sker redan efter nio timmar<sup>13</sup>.

<sup>11</sup><http://sodoityourself.com/max232-serial-level-converter/> hämtad 2010-05-05

<sup>12</sup><http://www.arduino.cc/en/Reference/Millis> hämtad 2010-05-05

<sup>13</sup><http://www.cibomahto.com/2008/04/analysis-of-the-millis-function/> hämtad 2010-05-05

## 9 Slutsatser/Erfarenheter

Med begränsad hårdvara så väger nackdelarna med storleken på XML lätt över fördelarna med läsbarheten för människor och möjligheten till utökning av format. Parsning av data ska kunna ske snabbt och enkelt och därför skulle kanske ett binärformat eller ett mer rudimentärt textformat vara att föredra i dessa sammanhang.

Programmeringen sker någorlunda enkelt om man är van vid C men det är viktigt att ha i åtanke att man arbetar med en begränsad hårdvara och får prioritera minnesåtgång och hastighet i programkörningen.

Det är lätt att skapa en prototyp som kan göra saker. Det är förfining som tar tid.

## 10 Möjliga utökningar

Här följer en lista på saker som borde fixas i programvaran för skylten.

- Stöd för DNS (serverns IP-adress hårdkodad i dagsläget)
- Möjlighet att ändra vilken data som hämtas (filnamn hårdkodade i nuvarande kod)
- Fixa stöd för totalt sex siffror