XML

– Internets framtida format? –

Filosofie kandidatuppsats inom Datavetenskap

Författare: Håkan Bergh

Handledare: Bengt Carlsson

Ronneby maj 2001

SAMMANFATTNING

HTML (Hyper Text Markup Language) introducerades som ett enkelt och plattformsoberoende språk med syftet att göra information tillgänglig över hela världen och är idag det mest kända och använda språket på webben. När HTML skapades var det huvudsakliga användningsområdet publicering. Efter hand ökade mognaden hos användarna och behovet av mer avancerade tillämpningar ökade. Fler och fler började då inse begränsningarna med HTML och dess enkla struktur.

Metaspråket XML (eXtensible Markup Language) är ett universellt format för att strukturera dokument och data på Internet. Utvecklingen av XML startade 1996 och antogs som en W3C-rekommendation (World Wide Web Consortsium) i februari 1998. Syftet med XML var att skapa ett betydligt mer flexibelt språk än HTML, utan att för den skull göra det mer komplext.

Syftet med uppsatsen är att undersöka om det nödvändiga underhållet av presenterad information på webben (WWW) förbättras med hjälp av XML samt undersöka hur XML kan underlätta EDI (Electronic Data Interchange)via Internet. Uppsatsen fakta och påståenden bygger på en s k kvalitativ bearbetning d v s textmaterial i form av rapporter, specifikationer och böcker.

I min undersökning har jag kommit fram till att XML kommer att bli ett av morgondagens stora format, både vad gäller skapandet av Internetapplikationer och som meddelandeformat för informationsöverföring. Med XML kan allt från en enkel fil till avancerade affärsdokument skapas. Det som jag anser håller tillbaka XML i dagsläget är samtidigt en av dess styrkor, nämligen flexibiliteten, det vill säga möjligheten att deklarera egna märkord. Genom den här sortens flexibilitet blir även språket mer komplext jämfört med ett språk som t ex HTML. Ett av delmålen W3C satte upp vid skapandet var just att enkelheten skulle bibehållas. Detta delmål har delvis gått förlorat i XML. XML-familjen är idag relativt komplex. Samtidigt välkomnas förslagen till förbättringar och utbyggnad av utvecklarna, då det är kringstandarderna som ger XML flexibel funktionalitet. Den slutsats som kan dras av detta är att det kanske helt enkelt måste finnas en viss komplexitet i ett språk som ska kunna uträtta de saker som faktiskt XML gör.

ABSTRACT

HTML (Hyper Text Markup Language) was introduced as a simple and platform independent language with the purpose to make information accessible all over the world and today it's the most known and used language on the web. When HTML was created the main field of application was publishing. As time went by the degree of ripeness increased and the need of more advanced application increased as well. More and more people started to realise the limitations of the simple structure in HTML.

The meta language XML (eXtensible Markup Language) is a universal format to structure documents and data on the Internet. The development of XML began 1996 and was approved as a W3C-recommendation (World Wide Web Consortsium) in February 1998. The purpose with XML was to create a more flexible language then HTML, without making it more complex.

The purpose with this essay is to investigate if the necessary need of maintenance of presented information on the web (WWW) is improved with XML, and if XML can facilitate EDI (Electronic Data Interchange) on the Internet. The information and claims this essay contains is gathered from reports, specifications and books.

My investigation claims that XML is going to be one of big format of tomorrow. That goes for the creation of application on the Internet as well as format for electronic data interchange. One of the things that holds XML back is also one of the big advantage with the language, the flexibility, the possibility to declare your own markup-words. This flexibility makes the language more complicated then a language like HTML. One of the goals with XML was that it still should be simple. This goal is partly vanished. The family of XML is rather complex and contains several additions today. The developer accepts these additions. The conclusion is that a language has to contain some complexity if it's supposed to make complex things.

INNEHÅLLSFÖRTECKNING

1 INLEDNING

I detta kapitel introduceras det valda uppsatsämnet. Inledningsvis ges en bakgrundsbeskrivning som leder till problemdiskussion och syftesformulering.

1.1 Bakgrund

När HTML (Hyper Text Markup Language) och HTTP (Hyper Text Transport Protocol) introducerades kom detta att innebära starten för en explosionsartad utveckling av Internet som ledde till World Wide Web (WWW). Utvecklingen av HTML inleddes 1990 och HTML har sedan dess varit en W3C-rekommendation (standardiseringsorgan för Internet). Språket är leverantörsoberoende vilket gör att webbläsare och användare kan ta del av informationen världen över. (Statskontoret, 1998).

När webben (WWW) fick sitt stora genombrott i mitten på 90-talet var det huvudsakliga användningsområdet publicering. De flesta företag och efter hand även privatpersoner skulle ha en egen hemsida där man presenterade övergripande information om företaget eller sig själv. Efter hand ökade mognaden hos användarna och mer avancerade tillämpningar utvecklades. Fler och fler började upptäcka att HTML som språk hade ganska stora begränsningar i sin struktur och tanken på ett nytt flexiblare och mer avancerat språk växte fram. Detta kom att mynna ut i utvecklingsprocess som ledde fram till XML (eXstensible Markup Language).

XML startade som ett obestämt projekt av en liten grupp hängivna SGML-experter, (SGML beskrivs närmare i kapitel 3) som var övertygade om att världen behövde något mer kraftfullt än HTML. 1998 blev specifikationen till XML tillgänglig.

XML är ett språk som används för att definiera nya språk. Dessa skall sedan användas för att utbyta data på ett smartare sätt än dagens HTML-dokument som inte beskriver innehållet, och textfiler som inte beskriver strukturen i dokumentet. Statskontoret (1998) beskriver XML som ett mer avancerat språk, eller egentligen ett metaspråk, för att hantera informationen på ett strukturerat sätt.

1.2 Problemdiskussion

Det ursprungliga språket för att göra webbsidor, HTML, räcker inte längre till för att göra avancerade webbsidor. Långt mindre än tio procent av sidorna på webben följer HTML. De flesta webbsidesmakare lägger till nya funktioner som inte följer HTML utan kräver extra finesser i webbläsarna. Samtidigt tävlar företagen som gör webbläsare med att lägga in stöd för sådana nya funktioner som inte alls är standard, vilket lett till att vissa sidor endast fungerar i en specifik webbläsare. (Carlsson, 1999).

Ett behov har således växt fram av ett språk som är mer avancerat än HTML och som erbjuder större möjligheter vid utveckling av webbsidor. Här kan XML tillföra nya möjligheter och eventuellt ersätta HTML som det största språket på Internet. Jag vill med mitt arbete undersöka om det nödvändiga underhållet av presenterad information på WWW kan förbättras med hjälp av XML samt undersöka hur XML kan underlätta EDI via Internet.

1.3 Syfte

Mitt syfte med denna uppsats är att utifrån nedan beskrivna frågeställningar kunna besvara problemformuleringen från föregående stycke.

- HTMLs och XMLs struktur och syntax skiljer sig åt. Innebär skillnaderna några fördelar/nackdelar? Är det enklare att validera ett dokument skrivet i XML jämfört med HTML?

- Ett av de största problemen idag är säkerheten på Internet. Fler och fler företag väljer att skapa verksamhetskritiska system på webben. Löser XML säkerhetsproblemen? Finns det några fördelar jämfört med HTML vad gäller säkerhet?

- Kan tiden som läggs ned på att anpassa och uppdatera befintliga webbsidor till nyare HTML-standarder minskas? Insatsen för att hålla en HTML-uppbyggd webbplats uppdaterad är både tids- och kostnadskrävande. Besparingar skulle kunna göras om tiden det tar att uppdatera kunde minskas och/eller arbetet delvis automatiseras.

- Dagens verksamhetskritiska system kräver ofta stor server-kapacitet. Kan man med hjälp av XML minska belastningen på servern jämfört med lösningar skapade med HTML

- Är XML ett effektivt hjälpmedel för att skapa mer avancerade webbapplikationer? Kan XML ersätta HTML? Kan man med hjälp av XML skapa bra lösningar för EDI över Internet?

1.4 Metod

I uppsatsen har jag använt mig en hel del av Internet som informationskälla. Anledningen till detta är att den senaste informationen om XML ännu inte hunnit ges ut i bokform. På Internet kan man erhålla den senaste informationen och uppdateringarna i ämnet. Generellt anses Internet vara en mindre tillförlitlig källa än t ex böcker. Jag har dock försökt att välja Internetkällor omsorgsfullt så att felaktiga påståenden i möjligaste mån kunnat undvikas. Vidare kan nämnas en muntlig diskussion kring berört ämne med XML Swedens VD Jan Östberg.

2 SGML

Kapitel 2,3 och 4 syftar till att skildra den teori uppsatsen grundar sig på. Föreliggande kapitel ger en beskrivning av uppmärkningsspråken SGML, HTML och XML.

SGML är en förkortning för Standard Generalized Markup Language, och är sedan 1986 en internationell ISO-standard. SGML var från början ett utvecklingsprojekt hos IBM, och målet med projektet var att hitta ett sätt att dela ett dokument med flera användare/plattformar med bibehållen information om dokumentets struktur.

Uttrycket markup härrör från tryckeribranschen där det syftar på de instruktioner till sättaren som var inlagda i manuskriptet angående t ex dokumentets typsnitt, struktur och storlek. Instruktionerna utgjordes av markeringar som sattes i manuskriptets verkliga innehåll så att sättaren visste vilken typ av typsnitt som skulle väljas. I SGML kallas dessa markeringar för ”taggar” och de omges av tecknen ”<” och ”>”.

Med SGML kan man själv göra sina egna generaliserade instruktioner för dokumentets struktur, men ändå göra det på ett standardiserat sätt. Instruktionerna läggs i en fil kallad DTD (Document Type Definition), vilken innehåller de specifika regler för hur dokumentet ska struktureras. I och med att DTDn styr hur innehållet presenteras för användaren kan man ha många typer av dokument med likadana taggar, men beroende på vilken DTD man använder få olika utseende på dokumenten.

SGML togs enligt Statskontoret (1998) fram för att lösa ett antal konkreta problem, nämligen:

- Underlätta utbytet av information mellan användare av olika datorer och plattformar
- Underlätta återanvändning av information
- Fungera som mall för att strukturera informationen
- Ge förutsättningar för att skapa mer intelligenta informationslösningar

2.1 Utbyta information

SGML skapades för att minska kompatibilitetsproblemen mellan olika system och program. Målet med detta var att säkerställa att information lagrad med hjälp av SGML skulle gå att läsa även efter ett antal år, och inte vara omöjligt att läsa p g a att en ny plattform eller ett nytt program installerats.

2.2 Underlätta återanvändning av information

SGML kan med fördel användas till att skapa avancerad teknisk dokumentation, vilket vanligtvis är mycket kostsamt. Fördelen med den uppmärkningsteknik som SGML representerar är att i och med att sakinformationen är separerad från informationen som styr layouten så kan man relativt enkelt återanvända sakinformation och publicera den på olika medier, t ex CD-ROM och WWW, utan större förändringar.

2.3 Mall för att strukturera information

I och med att SGML medger att man separerar sakinformationen från informationen om layouten kan man skapa mallar för att få en gemensam struktur inom t ex ett företag. Genom den gemensamma strukturen känner alltid läsaren igen sig. SGML stödjer också validering av den här typen av dokument.

2.4 Skapa mer intelligenta informationslösningar

Med intelligenta informationslösningar menar Statskontoret (1998) multimediapresentationer eller elektroniska manualer, dvs information som kan bearbetas av ett datorprogram. Ett pappersdokument betecknar man som ”ointelligent”. SGML är kodad på ett sätt som gör den mycket lämplig att använda för att skapa intelligenta informationslösningar.

2.5 Problem med SGML

SGML är som ovan beskrivits en mycket omfattande standard som klarar av det mesta vad gäller modularisering, strukturering och länkning av information. Det som kanske är den största nackdelen med SGML är att det inte är utvecklat för att användas på Internet. HTML-läsarna klarar inte av att tolka de märkord som finns i SGML-dokumenten. För att komma runt det problemet försöker man översätta de egendefinierade märkorden i SGML till de som finns tillgängliga i HTML. Nackdelen med att göra detta är att man går från ett rikare och mer strukturerat format till ett betydligt enklare format. Mycket av flexibiliteten går då förlorad.

En annan nackdel är att det är mycket svårt och dyrt att utveckla programvaror för SGML. Standarden är mycket komplex vilket medfört att billiga och bra utvecklingsverktyg inte nått marknaden.

3 HTML

När HTML och HTTP introducerades kom detta att innebära starten för en explosionsartad utveckling av Internet vilket ledde till det som idag kallas World Wide Web. HTML introducerades som ett enkelt och plattformsoberoende språk med syftet att göra information tillgänglig över hela världen och är idag det mest kända och använda språket på webben.

HTML blev en offentlig rekommendation i december 1990 och har sedan dess förfinats och förbättras i en rad olika versioner. Språket är uppbyggt av ett begränsat antal fördefinierade uppmärkningsord enligt en rekommendation av W3C, och är vad man kallar ett statiskt språk, d v s data kan inte ändras interaktivt. (Holzschlag & Oliver, 1998).

3.1 Problem med HTML

När HTML skapades var det huvudsakliga användningsområdet publicering på webben. De flesta företag och efter hand även privatpersoner skulle ha en egen hemsida där man presenterade övergripande information om företaget eller sig själv. Efter hand ökade mognaden hos användarna och därmed blev behovet av mer avancerade tillämpningar också större. Fler och fler insåg då begränsningarna med HTML och dess enkla struktur.

Som ovan beskrivits består språket av en fast uppsättning märkord vilket gör det betydligt mindre komplext än SGML, men å andra sidan går en stor del av flexibiliteten förlorad. Dessa märkord räcker helt enkelt inte till för att skapa mer avancerade presentationer och tillämpningar. HTML beskriver till skillnad från SGML enbart hur dokumentet skall se ut och innehåller ingen information om innehållet i dokumentet.

Ett annat problem är att vi idag har många verksamhetskritiska system på Internet som är beroende av att tekniken är pålitlig, att det är lätt att uppdatera informationen, att säkerheten är tillförlitlig och att systemen är skalbara när användarna blir fler. HTML klarar helt enkelt inte av att leva upp till alla dessa krav då det är utvecklat för att presentera relativt enkel information, inte för att bygga verksamhetskritiska tillämpningar på. Att utöka antalet märkord i HTML ger inte heller någon lösning på problemet eftersom varje verksamhet behöver sina egna märkord. (Statskontoret, 1998).

4 XML

XML är ett universellt format för att strukturera dokument och data på Internet. Utvecklingen av XML startade 1996 och antogs som en W3C-rekommendation i februari 1998. XML är ett ganska nytt språk, men tekniken bygger ursprungligen på SGML. De som utvecklade XML försökte skapa ett språk lika kraftfullt som SGML, men som var enklare att använda och innehöll betydligt mindre komplexitet. (Bos, 1999)

XML hämtar merparten av sitt ramverk ur SGML men utelämnar allt som inte är absolut nödvändigt. Vissa tillägg har även gjorts till XML som inte finns i SGML. XML kallas vanligtvis för en deluppsättning av SGML med det kan även sägas att XML är som SGML, fast i mindre skala. (North & Hermans, 1999)

4.1 XML – ett metaspråk

Både SGML och XML är metaspråk, vilket innebär att de är språk med vilka man kan specificera nya språk. SGML och XML används alltså för att definiera olika märkspråk. Det gör man med hjälp av dokumenttypsdefinitioner (DTD, förklaras närmare i stycke 5.4.2.3). Exempel på ett sådant märkspråk är HTML, som alltså är en applikation till SGML och som har definierats med en DTD. Det finns hundratals olika uppmärkta språk definierade i SGML. De flesta av dessa språk har inte fått några formella namn som t ex HTML, utan de använder namn baserat på deras syfte eller den standard de använder. (North & Hermans, 1999)

Många SGML-applikationer har tagit flera år att ta fram. De tas ofta fram speciellt för något industriföretag för att täcka deras behov av informationslagring. Denna lösning är dyr och komplicerad men mycket kraftfull.

4.2 Mål med XML

I detta samt nästkommande stycke behandlas några av de mål utvecklarna hade med XML samt några möjligheter som det erbjuder.

- XML ska kunna användas med existerande webbprotokoll som t ex HTTP utan ytterligare tillägg. XML har utvecklats med webben i åtanke. Funktioner som behövs på webben och som inte finns i SGML, har lagts till eller ärvts från existerande applikationer
- XML ska vara kompatibelt med SGML. De flesta SGML-applikationerna kan idag konverteras till XML
- Det ska vara lätt att skriva program som bearbetar XML-dokument eller som fungerar tillsammans med XML. Det finns redan ett antal XML-anpassade applikationer skrivna i bla Java, SmallTalk, C++, JavaScript och Perl
- XML-dokument ska vara lätta att skapa. Det är lättare att skapa ett XML-dokument än ett HTML-dokument eftersom du inte behöver lära dig några taggar utan du kan skapa dina egna. Det gör även språket flexibelt och tänjbart

(North & Hermans, 1999)

4.3 Möjligheter med XML

Enligt W3C möjliggör XML:

- internationaliserad elektronisk publicering som är oberoende av media.
- för industrier att definiera plattformsoberoende protokoll för utbyte av data.
- sändning av information som kan behandlas automatiskt efter mottagandet.
- att det ska bli enkelt att behandla data med billiga program
- för människor att publicera information på det sätt de vill, med hjälp av style sheets (förklaras närmare i stycke 5.4.2.1)
- att information kan visas efter egna önskemål
- tillhandahållande av metadata – data om data – som hjälper till vid sökning av information.

(Connolly, 2001)

4.4 Beståndsdelar i XML

I XML finns ingen fast uppsättning taggar utan XML bygger på att du skapar dina egna taggar där du själv väljer både vad de ska heta och hur de ska användas. Det går även att lägga till taggar i befintliga applikationer efter egna behov. (Åström, 1999)

En XML-applikation är ett märkspråk skrivet i XML. Varje gång du skapar en egen XML-sida skapar du ett eget märkspråk genom att du definierar egna taggar. En XML-applikation är en färdig uppsättning XML-taggar som skapats med de regler som gäller för XML. Du kan även låta andra använda ditt märkspråk genom att dokumentera ditt språk. Dokumentationen ska då beskriva hur dina taggar ska användas och vilka regler som är definierade. (Åström, 1999)

I nedanstående stycken kommer jag att ta upp de viktigaste delarna som kan användas vid skapandet av en egen XML-applikation. De består av:

- XML-filen där all data skrivs in
- CSS och XSL-mallar som kan användas för att presentera och formatera data
- DTDer och scheman som kan användas för att ange information och regler om de taggar du skapar

4.4.1 XML-filen

XML-filen är alltid grunden i en XML-applikation eftersom det är där all data skrivs in som ska presenteras. Filen byggs precis som i HTML upp med hjälp av taggar som innesluter innehållet med en start och sluttagg.

Den viktigaste skillnaden jämfört med HTML är att i HTML är alla taggar fördefinierade och det går inte att definiera egna taggar. En fördel med XML är att innehållet i dokumentet omsluts av taggar som beskriver vad det är, vilket bla ger helt andra sökmöjligheter i dokument. (Åström, 1999)

Exemplet nedan visar hur egna taggar kan användas i en XML-fil.

```
<adresspost>

        <namn>

                <förnamn>Sven</förnamn>
                <efternamn>Svensson</efternamn>

        </namn>

        <gatuadress>Stora vägen 2
        </gatuadress>

        <postadress>

                <postnummer>123 67</postnummer>
                <ort>Malmö</ort>

        </postadress>

</adresspost>
```

I exemplet ovan skapas en adresspost som innehåller ett antal personer (I det här fallet endast en men fler kan adderas vid behov) där varje person innefattas av olika element som t ex förnamn, efternamn och adress.

4.4.2 Presentation av XML-data

Som tidigare nämnts innehåller XML-filen enbart data som ska presenteras men talar inte om hur detta ska ske. För att kunna visa XML-filens data behövs en mall som anger hur innehållet ska formateras, d v s hur det ska visas. Det finns två olika typer av mallar för att åstadkomma detta, CSS-mallar och XSL-mallar. I XML-filen anges vilken mall som ska användas för formateringen av innehållet. (Statskontoret 1998)

4.4.2.1 Cascading Style Sheets (CSS)

CSS är stilmallar som används för att tala om för webbläsaren hur data ska presenteras. Det är en äldre teknik som fanns före XMLs uppkomst och används till att formatera HTML-dokument.

Det går att skriva CSS-mallen direkt i XML-filen, en sk intern CSS-mall. En intern CSS-mall kan användas om stilmallen enbart kommer att användas till en enda XML-fil. Annars används en extern CSS-mall, d v s en separat fil, för formatering. En fördel med externa CSS-mallar är att du kan använda en och samma fil för att formatera flera olika XML-filer. (Åström, 1999)

4.4.2.2 Extensible Stylesheet language (XSL)

XSL är en ny teknik som utvecklades samtidigt som XML och är helt anpassad för att formattera XML-filer. XSL är en kombination av CSS och Document Style Semantics and Specifikation Language (DSSSL), vilket är ett språk som används till att formatera SGML-data. (Statskontoret 2000). Till skillnad från CSS-mallar kan XSL inte användas till HTML-dokument. XSL-mallar ger större möjligheter än CSS-mallar att påverka innehållet i XML-filen eftersom de erbjuder mer funktionalitet. (Åström, 1999)

XSL är en applikation skriven i XML med en mycket omfattande uppsättning element och attribut för att beskriva presentationen av en XML-fil. XSL består av två delar, en större del för sidlayout och presentation samt en mindre del, XSLT, för transformering av ett XML-dokument till ett nytt XML-dokument. (Statskontoret, 2000)

4.4.2.3 Document Type Definition (DTD)

DTD är en teknik för att beskriva en XML-applikation. Det är en fil som innehåller information om de olika taggarna du skapar i XML-filen, bla vilka taggar som får vara med, vad de heter, vilka attribut som är tillåtna och hur de får användas tillsammans. Syftet med DTDn är att se till att XML-filerna i en applikation utformas enligt uppsatta krav och regler. DTD är ett arv från SGML, vilket gjort att det blivit det naturliga sättet att definiera en XML-applikation. (Statskontoret, 1998)

En XML-fil behöver inte kontrolleras mot en DTD utan det går bra att skapa egna taggar direkt i XML-filen. Användandet av DTD behövs främst vid skapandet av större XML-applikationer eller om det vid skapandet av XML-filer är viktigt att en enhetlig standard följs. (Åström, 1999)

När man läser om XML stöter man ofta på uttrycken ”valid” och ”well-formed”. Om ett XML-dokument är helt korrekt avseende syntaxen samt att det följer en viss DTD säger man att det är ”valid”. Med ”well-formed” menas att dokumentet är helt korrekt syntaktiskt men inte följer någon angiven DTD. (Statskontoret, 1998)

4.4.2.4 XML-Schema

Det finns en annan teknik för att utföra samma sak som med DTDn, nämligen scheman. Scheman är en teknik som Microsoft började med att utveckla. En skillnad mellan DTD och scheman är att DTDn använder en helt egen syntax medan scheman är specialskrivna XML-filer. En ytterligare skillnad är att scheman har stöd för olika datatyper, d v s du kan ange om ett element ska innehålla text eller om det ska vara ett heltal. (Åström, 1999)

5 BAKGRUND EDI

Nedanstående fakta syftar till att ge läsaren en inblick i EDI utifrån uppsatsens perspektiv. All information kapitlet baseras på är hämtat från "The XML/EDI Group", (http://www.xmledi-group.org).

5.1 Business-to-business Electronic Data Interchange

EDI har använts för Business-To-Business (B2B) kommunikation i nästan 25 år. De inledande insatserna innehöll överenskommelser på hur man utbytte data mellan företag. Från början överfördes information från band, för att sedan skicka meddelanden över dedikerade kommunikationskanaler. För att undvika att behöva använda olika protokoll för att flytta data mellan olika företag så kom olika industrigrupper överens om hur standarder för deras meddelanden skulle kunna överföras. Dessa standarder skulle underlätta för företag inom samma bransch att överföra data mellan varandra.

Problemen med detta sätt brukar man dela upp i två kategorier:

- Företagen måste anpassa sina meddelanden till en redan vedertagen standard som kanske inte till fullo är anpassad för deras behov
- Standarden är strikt skriven vilket innebär att företagen måste följa en exakt standard som bestämmer precis hur de ska flytta data till och från fördefinierade EDI-format. Samtidigt var kostnaden för att anpassa sig till dessa system relativt hög.

Omvärlden har förändrats sedan de första EDI-lösningarna såg dagens ljus, och idag krävs mer dynamiska lösningar som stämmer överens med företagens organisatoriska uppbyggnad. Internet har förändrat människors sätt att mötas, köpa och sälja samt utbyta varor och tjänster. Framförallt visar Internet oss att EDI inte bara är intressant vad gäller B2B-lösningar. Samma lösningar är också intressanta för alla consumer-to-supplier relationer, oavsett om konsumenten är en slutanvändare, en återförsäljare, en serviceorganisation som t ex ett hotell, en statlig organisation eller en virtuell organisation.

5.2 Electronic business transaction

När Internet i slutet av 1990-talet slog igenom genomgick mönstret för elektronisk handel en dramatisk förändring. Framförallt introducerade Internet nya vägar för elektronisk handel mellan de parter som inte hade haft råd med detta tidigare.

Tidigare har utbyte av data skett genom automatiserade rutiner utan inblandning av människan. Med Internet introducerades ett helt nytt sätt att bedriva elektronisk handel, s k webbaserad handel, där en människas beslut, till skillnad från innan, sätter igång en process. Den här nya modellen baseras på att en rad av interaktiva val görs i form av elektroniska formulär.

För att klara av att skapa väl fungerande interaktiva lösningar måste man:

- Förstå hela affärsprocessen för att på så sätt se vilka informationsmängder som måste utbytas mellan systemen
- Skapa affärsspecifika regler för hur informationselement skall utbytas

För att kunna göra ovanstående saker måste man kunna:

- Identifiera varje del av information som skall utbytas och dess roll i utbytet.
- Identifiera källan till varje del som utbyts.
- Identifiera vilka element som skall finnas i informationen som utbyts, och om det behövs ange i vilken ordning de skall presenteras.
- Identifiera vem som har som har skapat, överfört, tagit emot och tittat på varje meddelande, samt vilka program som skall sköta den här kontrollen.
- Identifiera vilka regler som skall kontrollera att relevanta formulär av data har blivit utbytta i affärsprocessen.

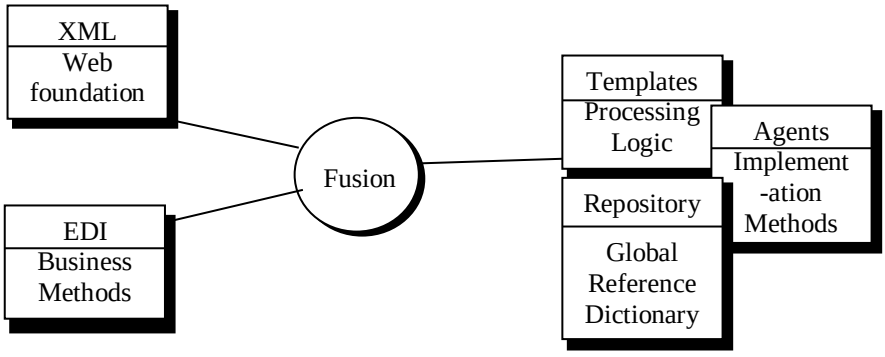
5.3 Teknologin för XML/EDI

XML/EDI är ihopsatt av många olika koncept. XML/EDI:

- Använder XML-protokollet som sitt ”dataöverföringslager”
- Använder XSL-protokollet som sitt ”presentationslager”
- Kan bli integrerat med traditionella metoder för EDI
- Kan användas med de olika standarder för överföring som finns idag som IP-routing, HTTP, FTP och SMTP.
- Använder sig av moderna programmeringsverktyg som Java och ActiveX för att underlätta att data delas mellan program.

XML/EDI kan ses som en sammansättning av fem olika existerande teknologier:

1. Webbdatautbyte baserat på *XML*
2. Existerande affärsmetoder och meddelandestrukturer för *EDI*
3. Kunskapsmallar (*templates*) som tillhandahåller kontroll för processer
4. DataBots, (*data manipulation agents*), som utför specialfunktioner
5. En samlad datamängd (*repository*) som ser till att relationer vidmakthålls.



Figur 1 XML/EDI byggstenar

5.4 EDI över Internet med XML

Styrkan hos XML bygger på att all information som överförs är plattformsoberoende och uppmärkt på ett sätt som är analyserbar för mottagaren. Till skillnad från HTML, där minsta beståndsdelen information just var HTML-dokumentet, är att XMLs minsta beståndsdel består av varje enskilt dataelement i dokumentet. Trots sin relativa enkelhet kan XMLs syntax beskriva allt från ett enkelt textdokument till databasliknande information. Detta gör det idealt för informationsöverföring genom att det inte längre krävs konverteringar för att kunna hantera information mellan olika system och program. (Norén, 2001)

XML är skapat för användning på Internet, vilket enligt många bedömare kommer att vara den klart dominerande distributionskanalen för information inom en överskådlig framtid. Dessutom är de kommunikationslösningar som grundar sig på Internet klart mer konkurrenskraftiga vad gäller kostnader än övriga alternativ. Med övriga alternativ menas t ex dedikerade kommunikationskanaler mellan två parter. (Norén, 2001)

5.5 Företags affärssystem och deras webbplats

Nästan alla företag eller organisationer idag har ett datorsystem som man lagrar affärsinformation i. Vinsten i tid och pengar uppstår enligt Fredholm (1999) inte förrän det interna affärssystemet integreras med företagets webbplats eller att olika affärssystem sammankopplas för elektronisk samverkan.

5.5.1 Affärssystem till affärssystem

Att implementera och sätta igång ett system som kopplar ihop olika företags affärssystem och som bygger på EDI är både kostsamt och kräver mycket manuellt arbete. Det finns ett antal möjligheter med XML:

- reducera kostnaden vid affärssamverkan
- reducera kostnaden vid uppstart av EDI
- är lätt att lära sig och använda
- ökar dataintegriteten och åtkomligheten
- tillgodoser säkerhet och kontrollmöjlighet
- skalbar
- går att integrera med dagens affärssystem
- använder en öppen standard
- globalt implementeringsbar

(Webber, 1998)

XML kan både ersätta och komplettera EDI. Startas en elektronisk affärssamverkan i dagens läge byggs denna på utbyte av XML-dokument som följer en överenskommen DTD. Detta ger ett flexibelt, billigt och lättarbetat system. Men de företag och organisationer som har lagt ned mycket pengar på utvecklandet av EDI-system vill använda dessa som grund för sin del av affärssamverkan. Att bygga på gamla EDI-system är möjligt på grund av att det går att översätta EDI-meddelanden till XML och skicka dessa över Internet. På så vis kan de företag som har EDI bedriva elektronisk affärssamverkan med de som också har EDI men också med de företag som inte har EDI. Även små företag som ej har haft råd med EDI tidigare kan, genom att samverkan sker via Internet, ha råd att utveckla EDI. (Webber, 1998)

5.5.2 Affärssystem till webbhandelsplats

För att kunna få en effektiv webbhandelsplats måste det manuella arbetet minimeras. De data och den information som finns i det befintliga affärssystem som företagen har måste användas med automatik av webbhandelsplatsen. XML Swedens VD Jan Östberg menar att kopplingen mellan affärssystem och webbhandelsplats går att genomföra utan att företagen måste byta till ett affärssystem som kan klara att hantera XML-dokument. Det tillvägagångssätt som används är ungefär det samma som med EDI-fallen. Data från affärssystemen plockas ur den befintliga databasen och körs genom en sk metamotor (ett program som översätter ursprunglig data till XML) och kan sedan presenteras på valfritt sätt på webbhandelsplatsen. På liknande sätt konverteras indata till affärssystemet, från XML till det ursprungliga formatat med hjälp av en metamotor.

XML Solutions (1999) tar upp fyra olika scenarier som ovan nämnda koppling med fördel kan tillämpas på:

- Framtagningsprocessen för företagens produktkataloger förkortas på grund av snabbare datauppdatering och mindre manuella editeringar och felkontroller. Vänteprocessen på datautbyte mellan företagsavdelningar minskar.
- Företagens webbhandelsplats kan bättre reflektera ändringar i produktsaldo och priser. Detta blir ännu viktigare då komplexa prissättningsstrategier och produktkonfigurationer används. Mänskliga inmatningsfel, katalogförseningar och andra faktorer som kan förstöra produktkataloger reduceras.
- Uppstart av elektronisk affärssamverkan kan förenklas. De störningar som kan finnas initialt vid uppstart minimeras.
- Ett rapporteringssystem baserat på förekomst eller ej förekomst av data eller händelse är lätta att implementera. Detta skulle kunna ge sofistikerade system som kan hantera semantiskt innehåll, t ex sortera E-post.

5.5.3 Slutsats XML/EDI

Det som gör XML till ett kraftfullt hjälpmedel för EDI på Internet är enligt min mening att:

- XML tillåter de som skapar dokument att tydligt märka upp varje element i dokumentet på ett sätt som både dator och människa förstår.
- XML tillåter de som skapar dokumenten att tala om vad som ska visas för mottagaren och i vilken ordning det skall presenteras.
- Genom särskilda metadata fält i XML-dokumenten kan man på ett enkelt sätt se vem som är ansvarig för att ha skapat, skickat, tagit emot samt tittat på varje enskilt meddelande, vilket innebär att man tillgodoser de säkerhets- och kontrollaspekter som dagens verksamhetskritiska system kräver.
- Är en öppen standard som stöds av de flesta större mjukvaruleverantörer.
- Inmatad information är lätt att validera.

Det har enligt mina källstudier visat sig att det finns en rad fördelar med XML inom EDI-lösningar. Det som dock måste poängteras är att ett av problemen men HTML var att det inte försåg användarna med tillräckligt mycket stöd för att skapa avancerade applikationer. HTML fick utökas med olika tilläggsprogram i diverse programspråk, vilket sågs som en stor nackdel, inte minst därför att utvecklarna av webbläsarna hela tiden låg steget efter i sin utveckling. När nu XML introduceras som en kraftfull teknik inom EDI är det tillsammans med en rad olika tekniker. Att lösa såpass komplicerade uppgifter som ett företags EDI-system med enbart ”ren” XML är kanske omöjligt, men man bör åtminstone ha problemen med HTML i åtanke när man skapar sina XML/EDI-lösningar.

6 JÄMFÖRELSE AV HTML OCH XML

Till skillnad från redovisningen av empirin som presenterar bakgrunden till, och uppbyggnaden av tre olika uppmärkningsspråk, SGML, HTML och XML så baseras min analys på jämförelser mellan framförallt de två sistnämnda. Analysen syftar dessutom till att göra återkopplingar till den teori som beskrivits under metodkapitlet. Nästföljande kapitel, Diskussion och slutsatser, summerar analysen i ett antal slutsatser vilka får illustrera de mest framträdande resultaten från min undersökning.

6.1 Struktur

Den största skillnaden mellan XML och HTML är hur strukturen är uppbyggd. Nedan följer beskrivning av strukturen uppbyggd i HTML jämfört med XML:

Exempel på struktur i HTML

```
<UL>
        <LI>Stekpanna
        <LI>Bord
        <LI>Diskett
        <LI>Stol
</UL>
```

Samma exempel skrivet i XML:

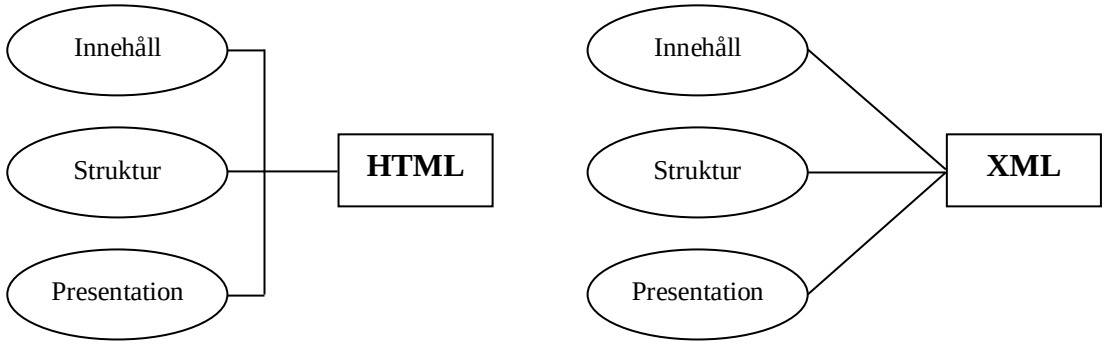
```
<Håkans saker>
        <Köksredskap>       Stekpanna   </Köksredskap>
        <Möbler>            Bord        </Möbler>
        <Datortillbehör>    Diskett     </Datortillbehör>
        <Möbler>            Stol        </Möbler>
</Håkans saker>
```

6.1.1 Jämförelse av struktur

HTML-exemplet är uppbyggt med generella taggar (märkord) som <UL> och <LI>. Taggarna i HTML-koden beskriver endast strukturen på dokumentet och presenterar detta på ett snyggt sätt. Någon information om vad de olika elementen i listan är för något definieras inte av HTML. I ett HTML-dokument kan innehållet, strukturen och presentationen sägas vara integrerade till en enhet. (Statskontoret, 1999)

XML-dokumentet, till skillnad från exemplet med HTML, består av egendefinierade taggar som avslöjar vad varje element är för något. XML-koden innehåller enbart data som ska presenteras, men talar inte om hur den skall presenteras. För att kunna visa XML-kodens data behövs en mall som anger hur innehållet ska formateras, d v s hur det ska visas. XML separerar alltså innehållet, strukturen och presentationen ifrån varandra till skillnad från HTML.

Detta medför bla att man i XML-dokumentet, till skillnad från HTML-dokumentet, kan göra en sökning efter t ex "köksredskap" vilket är omöjligt på en HTML-uppbyggd sida. (Bosak & Bray, 1999)



Figur 2 HTMLs respektive XMLs uppbyggnad

6.1.2 Slutsatser struktur

Skillnaden i struktur mellan HTML och XML är kanske den mest påtagliga olikheten. HTMLs struktur är väldigt enkel medan XML har en betydligt mer komplex och uppdelad uppbyggnad. En stor del av XMLs fördelar, som ökade möjligheter till sökning, enklare uppdateringsmöjligheter etc gentemot HTML går att hänföra till just den här skillnaden. XML är betydligt mer genomtänkt i sin struktur än vad HTML är. XML är inte bara skapat för att presentera information på Internet utan är tänkt att fungera som ett format för ett företags samtliga dokument.

6.2 Serverprestanda

Utvecklingen och mognaden på webben har ökat sedan 1990 då HTML lanserades. Idag skapar man verksamhetskritiska system på Internet. Dessa kräver att systemen är säkra, snabba, tillförlitliga och lätta att uppdatera. (Statskontoret, 1998). Systemen måste vara tillgängliga dygnet runt varje dag i veckan och klara av många användare samtidigt. Användarna vill ha presentation av webbsidor, söka bland webbsidor samt fylla i formulär och ställa frågor gentemot webbservern. (Bosak, 1997). Jag har utifrån detta valt att dela in analysen av serverprestanda i inmatning, sökning i databas samt utmatning/presentation.

6.2.1 Inmatning

En av de saker som skiljer XML ifrån HTML är att man på enkelt sätt kan kontrollera vad som matas in och vad som skrivs ut. När inmatning sker genom en webbplats är det enligt Bosak (1997) två saker som är viktiga att kontrollera:

- Är de uppgifter som skrivs in korrekta, d v s följer de reglerna för vad som får matas in
- Är uppgifterna inmatade i rätt ordning

XMLs struktur är, som tidigare beskrivits, uppbyggd av en DTD. I DTDn lagras regler för hur information är strukturerad. Fördelen jämfört med HTML ligger i att när inmatningen sker görs en validering (kontroll) gentemot DTDn av det som skrivs in. På så sätt följer inmatningen de regler man ställt upp för dokumentet och fel p g a inmatning minskar. I HTML kan en liknande kontroll göras, men då är man tvungen att använda sig av speciallösningar i andra programspråk. (Statskontoret, 1998).

Sett ur ett serverperspektiv kan även detta minska belastningen på servern. Kontrollen av inmatningen i formulär sker på klientsidan istället för på serversidan genom att klienten laddar ner XML-dokumentets DTD och kör en kontroll utifrån denna på inmatningsformuläret med hjälp av ett program som är standard i webbläsaren.

6.2.2 Sökning i databas

En av de saker som tar servertid i anspråk är sökningar i databaser. Bosak & Bray (1999) ger ett exempel på hur detta går till i HTML och XML:

Tänk dig att du skall beställa en flygbiljett mellan London och New York ett specifikt datum från en webbplats som har en rad olika flygbolags turer inlagda. Med HTML går sökningen till på följande vis:

- Användaren fyller i ett HTML-formulär
- Formuläret görs om till en SQL-fråga
- SQL-frågan körs mot databasen
- Svaret på frågan görs om till HTML
- HTML koden skickas till användarens webbläsare

Antagligen kommer listan du får tillbaka innehålla en rad olika alternativ. Vill du sedan förfina din sökning med att ange t ex avresetid och flygbolag måste du göra om ovanstående procedur.

Exemplet ovan, gjort i XML hade utförts på liknande sätt med inmatning i formulär, SQL-frågor etc. Skillnaden med en sökning i XML är att man från webbplatsen skickar med ett litet Java-program. I ovanstående exempel lagras all information från databasen i Java-programmet och den förfinade sökningen sker endast i det Java-programmet och man behöver inte göra ytterligare sökningar/förfrågningar gentemot databasen, vilket minskar belastningen på servern. Multipliceras ovanstående lösning med miljontals användare är effektivitetsökningen stor samtidigt som nätverksbelastningen minskar. (Bosak & Bray, 1999). En vidareutveckling av detta är att Java-programmet kan integreras i webbläsaren och sedan användas med XML-dokumentets DTD för att kunna göra en dokumentspecifik sökning.

Ett annat problem är att vi idag har många verksamhetskritiska system på Internet som är beroende av att tekniken är pålitlig, att det är lätt att uppdatera informationen, att säkerheten är tillförlitlig och att systemen är skalbara när användarna blir fler. HTML klarar helt enkelt inte av att leva upp till alla dessa krav då det är utvecklat för att presentera relativt enkel information, inte för att bygga verksamhetskritiska tillämpningar på. Att utöka antalet märkord i HTML ger inte heller någon lösning på problemet eftersom varje verksamhet behöver sina egna märkord. (Statskontoret, 1998)

6.2.3 Utmatning/Presentation

XML-filen innehåller ingen information om hur innehållet ska presenteras. Istället använder man sig av layoutmallar (stylesheets) som är egna filer med information om layouten.

En stor fördel med att särskilja informationen i dokumentet från utseendet är att samma information kan presenteras på olika sätt utan att man behöver ändra i XML-dokumentet. Man länkar istället olika layoutmallar till olika presentationstillfällen. Den här uppdelningen gör också att man kan välja vilken information som ska presenteras på olika medier. Ett företag kan exempelvis ha fullständig produktinformation internt med priser, antal produkter i lager osv. Ur samma informationsmängd kan man plocka ut den information kunderna ska se på företagets hemsida.

6.2.4 Slutsats serverprestanda

Som redogjorts för i styckena ovan minskar XML belastningen på servern avsevärt. HTML skapades helt enkelt inte för att man skulle kunna göra avancerade sökningar gentemot stora databaser utan är ett instrument för att presentera information på ett snyggt sätt på nätet. XML däremot är skapat för att möta de ökade kraven på åtkomlighet som krävs i dagens system. Ovanstående resonemang leder till slutsatsen att belastningen på servern kan minskas med XML-lösningar jämfört med lösningar gjorda i HTML.

6.3 Uppdateringar

Informationen i ett företag eller en organisation ändras ständigt. Detta gäller även för företagens eller organisationens webbsidor. De kan ge information om företag eller företagets produkter. Webbsidorna kan även vara ett instrument eller kanal för att sälja företagets produkter. Vad som gäller för alla dessa tillämpningar av webbsidor är att de måste vara aktuella och innehålla riktig information. (Degnan, 1999)

Att hålla en webbplats uppdaterad är ett av de mest tids- och kostnadskrävande momenten för en webbadministratör. (Bosak & Bray, 1999). Jag har därför valt att studera om det är enklare att uppdatera dokument gjorda i XML jämfört med HTML.

HTML är, som tidigare redan påpekats, ett verktyg för att presentera information på webben. Sett ur ett större perspektiv innebär detta ett problem, då man tvingas separera information som ska presenteras på Internet, och information som man t ex använder internt i ett företag. Eftersom HTML bygger på en enkel struktur med information och presentation ihopkopplade blir det omöjligt att använda samma dokument som skickas internt i företaget till presentation på företagets webbplats.

XML däremot, med presentation och information separerade, kan genom att endast ändra formatmall använda sig av en samlad informationsmängd för presentation både internt och på företagets webbplats. En av grundtankarna med XML var just att all information bara skall finnas på ett ställe i företaget.

Till hjälp just för att underlätta underhåll av webbplatser har man i XML skapat ett verktyg kallat XLink. XLink placerar alla länkar som finns på ett företags hemsidor i en fil vilket även detta medför en förenkling av uppdatering. Jämfört med en webbplats uppbyggd med hjälp av HTML, där länkar till andra sidor ligger spridda i koden, samlas all information på ett ställe med XML. Detta gör att tiden det tar att uppdatera webbplatsens olika delar kan minskas med XML. (Arciniegas, 2000)

6.3.1 Slutsatser uppdateringar

Även i det här avseendet är XML en självklar vinnare. Lösningar med HTML täcker endast in publicering på webben och skapar inga förutsättningar för att ha en och samma informationsmängd på olika medier inom t ex ett företag. XML däremot kan användas både som lagringsformat för dokument och som presentationsverktyg på Internet vilket gör att när ett dokument skall ändras, behöver det endast ske på ett ställe vilket i sin tur leder till att uppdateringen tar mindre tid och risken för redundant information elimineras.

6.4 Säkerhet

Utvecklingen av XML tog fart när man upptäckte att mer avancerade tillämpningar krävde mer komplicerad teknik, snabbare uppdateringsmöjligheter och ökad säkerhet. HTML fick efter hand som kraven ökade använda sig av tilläggsprogram i olika programspråk för att möta kraven på säkerhet. Enligt XML Swedens VD Jan Östberg är det idag fullt möjligt att bygga lösningar som uppfyller de säkerhetskrav man kan kräva i en normal verksamhet. Det är dock inte enbart lösningar med HTML utan som ovan nämnts HTML i kombination med något annat programspråk som gör det möjligt.

Många är idag oroliga för säkerheten i utvecklandet av XML på Internet. Till exempel är det lätt att återanvända formatmallar som är sparade i system utanför den direkta kontrollen för användaren. Görs ändringar i ”originalmallen” kan detta få stora konsekvenser på hur dokument presenteras, uppgifter kan med lätthet ändras och leda till att hela innehållet på en sida förändras. (Laurent, 2000). Jan Östberg uttrycker det på följande sätt: *”Utan särskild kompetens inom XML är verksamhetskritiska företag körda”*. Laurent (2000) hävdar vidare i sin artikel att man genom att lägga till deklarationer i koden kan förstöra valideringen som görs i DTDn. En liten förändring som genererar ett allvarligt fel i en DTD kan stanna upp en hel process. Ett annat exempel på detta kan vara när en extern DTD anropas via URLs och dessa förstörs, flyttas eller blir oåtkomliga.

6.4.1 Slutsatser säkerhet

Slutsatserna för ovan beskrivna del är att varken lösningar gjorda i HTML eller XML löser de säkerhetsproblem man faktiskt har idag på Internet. Med HTML kan man idag lösa en del av dessa säkerhetsproblem genom att använda sig av tilläggsprogram i olika programmeringsspråk, men HTML lider fortfarande av sin enkla och statiska struktur. XML skapades till stor del för att HTMLs struktur inte fungerade väl för verksamhetskritiska system, men trots detta har man inte lyckats att minimera de säkerhetsproblem dessa lider av.

6.5 Validering

HTMLs och XMLs valideringsmetoder skiljer sig en del åt. När HTML-dokument valideras får man p g a strukturen kontrollera hela HTML-filen. Ett program som t ex HTML Validator (http://validator.w3.org/) sköter den här kontrollen på ett bra sätt.

XML med sin uppdelning av struktur, information och presentation valideras på ett annorlunda sätt. En programvara kallad parser kontrollerar att den struktur man skrivit i XML-dokumentet överensstämmer med DTDn. DTDn och parsern säkerställer att alla dokument följer den givna strukturen, vilket har stor betydelse vid maskinell behandling av dokumenten. (Statskontoret, 1998)

6.5.1 Slutsatser validering

Både HTML och XML har väl fungerande validering av sina dokument. Skillnaden är att valideringen i HTML görs genom att hela dokumentet kontrolleras, medan XML kontrollerar att strukturen och syntaxen följer den DTD man använder. Någon direkt för- eller nackdel för respektive metod har ej kunnat påvisas i den här undersökningen.

6.6 Syntax

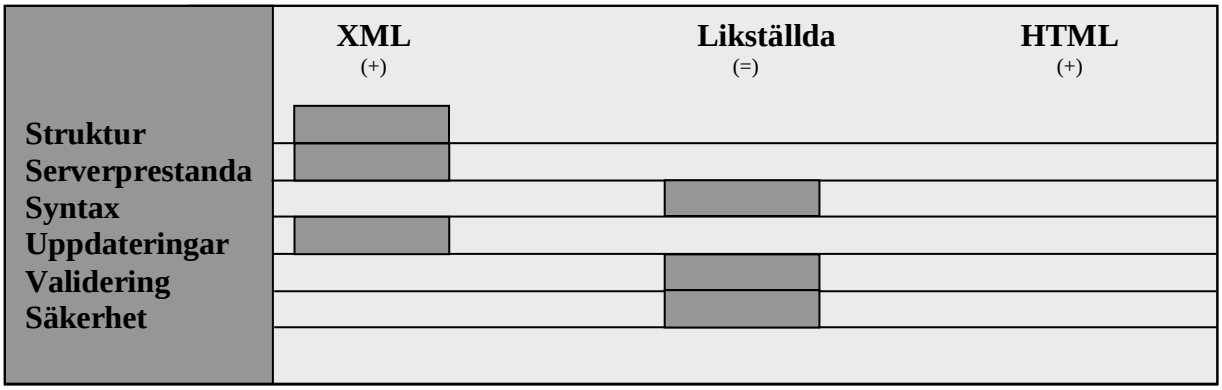
HTML är som tidigare nämnts uppbyggt av en relativt enkel syntax med fördefinierade uppmärkningsord. Syntaxen är dock trots sin enkelhet kraftfull vad gäller att snabbt skapa webbsidor som layoutmässigt ser bra ut.

XML är betydligt mer komplex i sin syntax. Eftersom strukturen är uppdelad i olika delar, så måste man som användare av XML både kunna skriva DTDer och formatmallar för att skapa välskrivna dokument.

6.6.1 Slutsatser syntax

XML-dokument ska vara lätta att skapa. Det är lättare att skapa ett XML-dokument än ett HTML-dokument eftersom du inte behöver lära dig några taggar utan du kan skapa dina egna. Det gör även språket flexibelt och tänjbart. (North & Hermans, 1999). Personligen kan jag inte dra riktigt samma slutsatser. HTML anser jag med sin sammansatta struktur vara relativt enkelt att skapa snygga webbsidor med. Detta gäller dock enbart om målet med webbsidan är att presentera information och inte skapa mer avancerade tillämpningar. I XML bör du kunna skapa både DTD’er och formatmallar för att skapa snygga och välstrukturerade hemsidor. Slutsatsen av den här jämförelsen blir alltså att HTMLs syntax är enklare att lära sig när det handlar om grundläggande webbprogrammering, men när mer avancerade tillämpningar ska göras blir detta enklare i XML.

7 SLUTSATSER



Figur 3 Sammanställning av jämförelse HTML/XML

Som bilden ovan avspeglar är XML bättre eller likvärdig på de punkter som tagits upp under kapitel 8, *Jämförelser av HTML och XML*. XML är ett mycket kraftigt hjälpmedel, och kommer med all säkerhet att användas allt mer, både som internt format och för att skapa avancerade webbapplikationer med. Det som är lite oroväckande är dock att man inte har lyckats lösa de säkerhetsproblem som är enormt viktiga att ta itu med om verksamhetskritiska system ska fungera på ett bra sätt. Anledningen till att ”Syntax” har hamnat i mitten är att språket i HTML duger alldeles utmärkt att skapa hemsidor med. Då är det ett betydligt enklare och effektivare hjälpmedel än XML. Ska man skapa mer avancerade tillämpningar är dock XML det självklara valet.

Den samlade bilden är att XML kommer att bli ett av morgondagens stora format, både vad gäller skapandet av Internetapplikationer och som meddelandeformat för informationsöverföring. Med XML kan allt från en enkel fil till avancerade affärsdokument skapas.

Det som jag anser håller tillbaka XML i dagsläget är samtidigt en av dess styrkor, nämligen flexibiliteten, det vill säga möjligheten att deklarera egna märkord. Genom den här sortens flexibilitet blir även språket mer komplext jämfört med ett språk som t ex HTML. Ett av delmålen W3C satte upp vid skapandet var just att enkelheten skulle bibehållas. Detta delmål har delvis gått förlorat i XML. XML-familjen är idag relativt komplex. Samtidigt välkomnas förslagen till förbättringar och utbyggnad av utvecklarna, då det är kringstandarderna som ger XML flexibel funktionalitet. Den slutsats som kan dras av detta är att det kanske helt enkelt måste finnas en viss komplexitet i ett språk som ska kunna uträtta de saker som faktiskt XML gör.

8 REFERENSLISTA

Böcker/utredningar

Fredholm, P., (1999). *Elektroniska affärer.* Studentlitteratur. Lund

Holzschlag, M. & Oliver, D., (1998). *Lär dig HTML 4 på 24 timmar.* Pagina Förlags AB. Sundbyberg.

Lekvall, P. & Wahlbin, C., (1993). *Information för marknadsbeslut.* IHM förlag. Göteborg.

Lundahl, U. & Skärvad, P., (1999). *Utredningsmetodik för samhällsvetare och ekonomer*. Studentlitteratur. Lund.

North, S. & Hermans, P., (1999). *Lär dig XML på 3 veckor*. Pagina Förlags AB. Sundbyberg.

Statskontoret, (1998). *Vad är XML?.* Statskontoret. Solna.
Finns att tillgå online på:
< http://www.statskontoret.se/pdf/199806.pdf > [2001-05-13]

Statskontoret, (2000). XML-familjen – Vad är XML formatmallar?. Statskontoret. Solna.
Finns att tillgå online på:
< http://www.statskontoret.se/pdf/200033.pdf > [2001-05-14]

Webber, D., (1998). *Introducing XML/EDI Frameworks.* Electronic Markets vol:8 nr:1

Åström, P., (1999). *Databoken XML, Exstensible Markup Language.* IHM förlag. Göteborg.

Intervjuer

Östberg, J., XML Sweden, Järfälla, 2001-04-03

Internetkällor

Arciniegas, F., (2000). *What is Xlink?.*
< http://www.xml.com/pub/a/2000/09/xlink/index.html > [2001-05-14]

Bos, B., (1999). *XML in 10 points.* W3C.
< http://www.w3.org/XML/1999/XML-in-10-points > [2001-05-13]

Bosak, J., (1997). *XML, Java, and the future of the Web.*
< http://metalab.unc.edu/bosak/xml/why/xmlapps.htm > [2001-05-14]

Bosak, J. & Bray, T., (1999). *XML and the Second-Generation Web.* Scientific American.
< http://www.sciam.com/1999/0599issue/0599bosak.html > [2001-05-14]

Carlsson, T., (1999). *Webbspråket som förenar alla datorer.* Ny Teknik.
< http://www.nyteknik.se/pub/ipsart.asp?art_id=7767 > [2001-05-13]

Connolly, D., (2001). *Activity Statement*.
< http://www.w3.org/XML/Activity.html > [2001-05-13]

Degnan, C., (1999). *Migrating Web sites to XML.* eWeek.
< http://www.zdnet.com/eweek/stories/general/0,14352,406676,00.html > [2001-04-23]

Laurent, S., (2000). *When XML gets Ugly*. Xtech.
< http://www.xml.com/pub/a/2000/02/xtech/megginson.html > [2001-05-14]

Norén, K-J., (2001). *Babels torn eller tusen blommor – XML, CSS och framtiden.* Pagina Förlags AB. Sundbyberg
< http://nezzo.nu/np.asp?id1=1&id2=2155 > [2001-05-14]

XML Solutions., (1999). *XML White Paper: Legacy System Integration Using XML.*
< http://www.xmls.com/resources/whitepapers/XMLSolutions_Legacy.pdf > [2001-05-14]

W3C, World Wide Webb Consortsium., (2001). *XML/EDI Group.*
< http://www.xmledi-group.org > [2001-05-14]