

Copyleft
En upphovsrättslig studie av ett
villkor i fria programvarulicenser

Examensarbete i juridik
Juridiska fakulteten vid Stockholms universitet

Love de Besche
HT 2011

1 Inledning

Svenska företag investerar årligen cirka 60 miljarder kronor i mjukvaruutveckling, en industri som beräknas hålla omkring 75 000 svenskar sysselsatta.¹ Hur mycket pengar som investeras internationellt är inte känt, men det är uppenbart att det rör sig om enorma summor. Det finns således ett kommersiellt intresse i att kunna förutse de juridiska risker som följer med investeringarna som görs i programmeringsprojekt.

Programmeringsindustrin har sedan länge haft en uppdelning mellan två typer av projekt: de som ger användaren rätt att ta del av källkoden till programmet (vanligast inom *F/OSS*² – se nedan 1.4) och de som licensieras utan tillgång till källkoden (*proprietär programvara*, se nedan 1.4). Valet av projekttyp grundar sig till stor del på vilken affärsmodell man vill använda sig av.

På senare tid har uppmärksamhet riktats mot de licenser som används till F/OSS-projekt. Anledningen är att det råder stor förvirring kring huruvida användandet av öppen källkod i programvara som i sin tur inte offentliggör sin källkod är tillåtet enligt licensavtalen för den öppna källkoden. Att praxis på området saknas underlättar inte situationen.

Osäkerheten kan exempelvis uppstå när ett programmeringsföretag investerat ett flertal miljoner kronor i skapandet av en programvara och man har räknat med att återfå utvecklingskostnaden genom försäljning av programvaran. Man har dock valt att integrera öppen källkod i sin programvara. Det fullt realistiska mardrömsscenarioet vore att tvingas välja mellan att följa den öppna källkodens licens genom att göra programvaran fritt tillgänglig för alla, eller att lägga dyrbara resurser på att granska källkoden rad för rad i syfte att avlägsna den öppna källkoden därifrån. Innehåller programmet företagshemligheter skulle ett offentliggörande kunna bli katastrofalt för de egna affärerna.

Licensen som är i fokus för studien – GNU General Public License³ [hädanefter ”GPL”] – är skriven utifrån ett amerikanskt rättsperspektiv vilket medför tolkningsproblem av den i ljuset av svensk rätt. Ett exempel på detta som kommer att utvecklas nedan (se 4.2) är hur termen ”derivative works” kan komma att tolkas enligt svensk rätt.

Ämnet utgör en kombination av problem av såväl juridisk som teknisk karaktär. Uppsatsen riktar sig till yrkesverksamma med en god förståelse för såväl tekniska som juridiska begrepp. Det författaren vill poängtera med detta är att höga krav ställs på läsarens tekniska förståelse, då det inte finns utrymme att förklara annat än de centrala begreppen.

¹ Se <http://www.swedsoft.se>.

² Free-/Open Source Software. ”Free” syftar här inte på ”gratis” utan ”fri”, för en fördjupning i distinktionen hänvisas till <http://www.gnu.org/philosophy/free-sw.html>.

³ Se <http://www.gnu.org/copyleft/gpl.html>.

1.1 Syfte

Syftet med denna uppsats är att redogöra för de juridiska frågeställningar och risker som uppstår genom att GPL-licensierad kod bearbetas och integreras i programmeringsprojekt av proprietär typ.

1.2 Avgränsning

Uppsatsen tar sikte på att beskriva rättsläget i Sverige. Detta kommer även att innefatta relevanta inslag av europarätten. Mjukvarupatent kommer att uteslutas från uppsatsen då dessa är av större betydelse i USA än i Sverige. Vidare syftar framställningen inte till att besvara vad som är ”bäst” av F/OSS och proprietär programvara. Framställningen kommer heller inte att behandla problematik förenad med att copyleftlicensierad mjukvara distribueras med hårdvara och där användaren inte tillåts förändra mjukvaran (TiVo-situationen).⁴

1.2.1 Val av licens

För att fokusera arbetet på problematiken kring copyleftklausuler snarare än att behandla alla till buds stående copyleftlicenser krävs att någon licens väljs ut. Författaren har här tagit sikte på två faktorer, nämligen att (1) licensen ska tillämpa stark copyleft, då copyleft-problematiken ställs på sin spets och att (2) den ska vara vanligt förekommande och därmed kommersiellt relevant. Båda villkoren uppfylls av en licens – GPL – som i sina olika versioner används i majoriteten av alla F/OSS-projekt.⁵ GPL förekommer idag i två versioner: version 2 [”GPLv2”] som togs i bruk 1991 och version 3 [”GPLv3”] som släpptes 2007. Den senare versionen är i grunden densamma som v2, men med flera betydelsefulla tillägg och klargöranden. De för uppsatsen relevanta frågorna kring när och hur copyleftklausulen ska tillämpas är i stort desamma i de båda versionerna.

GPLv2 talar om ”distribution of derivative works” medan GPLv3 använder termen ”conveying of a covered work”. Innebörden av ”derivative works” och ”covered work” är enligt författaren snarlik då de definieras som ”the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it [...]”⁶ respektive ”the unmodified Program or a work based on the Program”.⁷

Termen ”distribution” i GPLv2 har ersatts av ”conveying” i GPLv3. Detta skedde på grund av att ”distribution” inte definierades i GPLv2 och skapade därför en tolkningsfråga

⁴ Se <http://en.wikipedia.org/wiki/Tivoization>.

⁵ Andelen F/OSS-programvara licensierad under GPL 2.0 och 3.0 utgör tillsammans 51,84% av det totala, <http://www.blackducksoftware.com/oss/licenses#top20>.

⁶ Första stycket artikel 0 i GPLv2.

⁷ Femte stycket artikel 0 i GPLv3.

kring vad det omfattade.⁸ Termen existerar redan inom de flesta länders upphovsrättsliga lagstiftning. Risken var därför enligt FSF att termen skulle få olika innebörd beroende på vilket land GPL prövades i. FSF:s tanke att termen skulle förstås enligt dess innebörd inom amerikansk upphovsrätt.⁹ Termen ”convey” kom därför att få en uttrycklig definition i GPLv3 som lyder ”any kind of propagation that enables other parties to make or receive copies”.¹⁰ Definitionen är enligt författaren i överensstämmelse med 2 § tredje stycket p. 1 URL som anger när ett verk anses överfört till allmänheten. GPLv2 talar om ”distribution” och innebörden kommer att tolkas enligt tillämplig lag, i svensk rätt enligt 2 § URL. Författarens uppfattning är därför att de båda termerna tar sikte på samma situationer och aktualiserar samma lagrum inom svensk rätt. Med anledning härav kan det som sägs om GPLv2 i stort anses gälla även för GPLv3.¹¹ Uppsatsen kommer av pedagogiska och utrymmesbesparande skäl att analysera copyleftklausulen i GPLv2 om inget annat särskilt anges.

1.3 Metod

Sedvanlig juridisk metod har tillämpats. Fokus har legat på litteraturstudier. Frågan om copyleftklausulens effekt är främst debatterad av amerikanska programmerare och jurister, flera av dem tydligt styrda av bakomliggande ideologier. Detta innebär att majoriteten av tillgängliga artiklar rör sig på skalan ”partiska till starkt partiska”, varför ett stort mått av försiktighet måste iakttas vid hänvisningar till dessa såsom fakta.

1.4 Terminologi

För att underlätta analysen presumeras datorprogram i exemplen uppfylla artikel 1.3 i datorprogramsdirektivet¹² genom att anses vara ”originella i den meningen att det är uphovsmannens egen intellektuella skapelse”. Därmed erhåller det uphovsrättsligt skydd.

I uppsatsen kommer termerna ”program”, ”programvara” och ”mjukvara” användas parallellt för att beskriva kompilerad kod, det vill säga kod i maskinläsbar form, som exekveras genom ett operativsystem. Den sistnämnda delen av distinktionen är viktig av skäl som framgår under 2.1 nedan. Termen ”källkod” avser den kod som programmeraren skriver och som kan förstås och tolkas av människor.

⁸ Se <http://www.gnu.org/licenses/quick-guide-gplv3.html>.

⁹ Ibid.

¹⁰ Se sjunde stycket artikel 0 i GPLv3.

¹¹ Påståendet kan dock inte göras reservationslöst. FSF har sedan GPLv2 offentliggjordes gjort flera uttalanden, besvarat frågor (GPLv2:s FAQ exempelvis) där de ger sin syn på hur GPLv2 ska tolkas. Dessa skulle knappast kunna anses utgöra grund för tolkning av GPLv2 då de tillkommit efter licensen togs i bruk, men skulle kunna påverka en domstols tolkning av GPLv3 då de fanns tillgängliga vid dess offentliggörande.

¹² Europaparlamentets och rådets direktiv 2009/24/EC av den 23 april 2009 om rättsligt skydd för datorprogram.

Vidare kommer begreppet ”F/OSS” att användas som ett samlingsbegrepp för så väl ”fri mjukvara” som ”programvara med öppen källkod”. Begreppen är inte synonyma, främst på grund av att det finns flera ideologiska skillnader i synsätten mellan de båda.¹³ På motsvarande sätt kommer termen ”proprietär” användas för att beteckna programvara som inte är F/OSS, det vill säga där källkoden inte offentliggörs. Termerna ”GPL-licensierad” och ”GPL-skyddad” kommer användas synonymt för att beskriva källkod eller programvara som gjorts tillgängliga under GPL.

I största möjliga mån kommer svenska termer och uttryck att användas, men vissa termer – främst programmeringstekniska sådana – bedöms så pass etablerade inom området att dess engelska ord kommer att användas.

1.5 Svensk lags tillämplighet i GPL-sammanhang

Svensk lags tillämplighet är en förutsättning för analysen. Då GPL saknar lagvalsklausul så blir bestämmelserna om lagval i internationella privaträttsliga förhållanden tillämpliga. För att finna tillämplig lag krävs att en grundläggande fråga besvaras, nämligen vad tvisten gäller.

För det fall att tvisten gäller *avtalsförhållandet* kommer Rom I-förordningen¹⁴ att bli tillämplig. Lagvalet bestäms i första hand av artikel 4.2, vilken anger att ”det land[s lag] där den part som ska utföra avtalets karakteristiska prestation har sin vanliga vistelseort” ska vara avgörande. Detta torde i copyleftlicensens fall, där licenstagaren inte presterar mer än att följa licensens villkor, innebära att det är licensgivarens prestation – att utge licensen för programvaran – som blir avgörande, varför dennes hemlands lag ska tillämpas.

Situationen är den omvända om frågan gäller en *immaterialrätts existens eller innehåll*. I det fallet kommer nämligen skyddslandsprincipen (*lex loci protectionis*, artikel 8.1 i Rom II-förordningen)¹⁵ att tillämpas. I korthet innebär skyddslandsprincipen att det lands lag, inom vars territorium det immaterialrättsliga intrånget påstås begånget, ska tillämpas. Utfallet blir därför att tvister rörande avtalet skall avgöras enligt licensgivarens hemlands lagar.¹⁶

Då båda ovanstående situationerna kan medföra att svensk lag blir tillämplig kommer det i den kommande framställningen att presumeras att någon av dem föreligger.

¹³ Jämför FSF:s definition av ”free software” (not 2) med OSI:s definition av ”open source”, tillgänglig på <http://opensource.org/docs/osd>.

¹⁴ Europaparlamentets och rådets förordning (EG) nr 593/2008 av den 17 juni 2008 om tillämplig lag för avtalsförpliktelser.

¹⁵ Europaparlamentets och rådets förordning (EG) nr 864/2007 av den 11 juli 2007 om tillämplig lag för utomobligatoriska förpliktelser. Se även preamblens punkt 26.

¹⁶ Notera emellertid undantaget angivet under 4.4.

2 Allmänt om copyright och copyleft

När en programmerare skapar ett program så erhåller han enligt upphovsrättslagen¹⁷ [hädanefter ”URL”] ett omedelbart och formlöst upphovsrättsligt skydd. Det är dessa rättigheter som upphovsmannen kan förfoga över. Omfattningen av skyddet är begränsat till datorprogrammets uttrycksform och omfattar inte bakomliggande idéer, logik eller algoritmer.¹⁸ Emellertid erhålls skydd för ”förberedande designmaterial för datorprogram”,¹⁹ exempelvis flödesscheman, källkod och objektкод.²⁰ Skyddet innebär en exklusiv rätt att förfoga över det skyddade verket (de ekonomiska rättigheterna) samt ideella rättigheter. De senare består av respekträtten respektive namngivelsesrätten. Respekträtten innebär att upphovsmannens verk inte får användas eller ändras på ett sätt som är kränkande för dennes litterära anseende.²¹ Namngivelsesrätten stipulerar att om ett skyddat verk görs tillgängligt eller framställs ska upphovsmannen namnges i enlighet med god sed på området. Dessa rättigheter kan, till skillnad från de ekonomiska, normalt endast efterges i begränsad del.

Denna korta inledning om upphovsrätten kommer att ligga till grund för den följande framställningen kring copyleftlicenser. Copyleftlicensernas funktion kommer att förklaras i abstrakt form för att sedan utvecklas och tillämpas konkret i samband med genomgången av GPL i kapitel 4 nedan.

2.1 F/OSS eller proprietär programvara?

Inledningsvis ska några ord ägnas åt den företagsekonomiska motiveringen till valet mellan de olika affärsmodellerna proprietär programvara respektive F/OSS. Om källkoden förblir hemlig för såväl användare som konkurrenter så skapas ett incitament till att göra investeringar i programvarans utveckling. Detta bygger på en affärsmässig förväntning om att kunna tjäna in utvecklingskostnaden genom försäljning av den färdiga proprietära programvaran. Finurliga lösningar på problem skapar konkurrensfördelar och därmed möjligheter att ta delar av marknaden. Så länge inga liknande eller mer effektiva lösningar finns så kan dessa konkurrensfördelar och därmed marknadsandelar bibehållas. Källkoden kan därför utgöra en viktig och skyddsvärd tillgång för ett företag. Av samma anledning kan tillgången till källkoden anses ha ett ekonomiskt värde i och med den tid en programmerare kan spara på att ha färdiga lösningar.

¹⁷ Lag (1960:729) om upphovsrätt till litterära och konstnärliga verk.

¹⁸ 2009/24/EC preambeln punkt 11 samt artikel 1.2 jämte URL 1 § p. 2. Att det förberedande materialet är upphovsrättsligt skyddat torde följa redan av första stycket, givet att de uppnår kravet på verkshöjd.

¹⁹ URL 1 § 3 st.

²⁰ Upphovsrättslagstiftningen, lagtextkommentar från Zeteos Internettjänst, Kommentar till 1 §.

²¹ Denna rätt torde vara av mindre intresse inom F/OSS, se Westman, D., *Återanvändning av programkod som utvecklas av eller för den offentliga sektorn*, IRI promemoria 2/2007 s. 5.

Den andra varianten är att göra källkoden fritt tillgänglig tillsammans med programvaran. Av naturliga skäl gör detta det svårt att ta betalt för programvaran men öppnar andra dörrar för att tjäna pengar. Några exempel är att man tar betalt för support eller genom att man erbjuder särskilda funktioner mot en kostnad (*open core*).²² En annan fördel med att tillgängliggöra källkoden är att både de egna programmerarna och användarna av programmet har tillgång till den. Detta leder exempelvis till att säkerhetshål eller buggar kan upptäckas och åtgärdas på ett förebyggande sätt.²³ Användarna av F/OSS får nämligen även rätten att modifiera programvaran, något som inte är det normala vid licensiering av proprietär programvara.²⁴

2.2 Upphovsrätt ett måste för att copyleftlicenser ska fungera

Upphovsrättslagen är vad som ger upphovsmannen de rättigheter som han kan förfoga över. Om en programmerare säljer en nyttjanderätt till ett program utan några särskilda villkor kommer upphovsrättslagen och de däri angivna villkoren att gälla. Som exempel kan nämnas att licenstagaren inte får framställa andra kopior av programvaran än en säkerhetskopia.²⁵ Eftersom upphovsrättslagen till stora delar är dispositiv så kan man genom avtal skapa egna villkor för hur och på vilket sätt ens verk får användas. Det är den rollen som licensavtal fyller.

För att de upphovsrättsliga huvudreglerna ska avtals bort så måste bundenhet enligt copyleftlicensen uppstå. Detta sker genom att en användare accepterar att copyleftlicensen ska tillämpas, vilket ofta sker när programmet laddas ned, installeras, körs eller modifieras.²⁶ Om en individ inte är bunden av copyleftlicensen så kan dess villkor heller inte tillämpas, varvid upphovsrättslagen träder in. Samma sak inträffar vid avtalsbrott, då konsekvensen av ett sådant är att licensen upphör, se vidare nedan 4.1.2.

2.3 Tanken bakom copyleft – ett krav på reciprocitet

Den ideologiska tanken bakom copyleft är enkelt sammanfattat en antirörelse mot upphovsrätten som den uttryckts i lagar världen över. Man vill frångå tanken om ensamrätt och exklusivitet till information och lösningar på problem för att garantera att de görs tillgängliga för alla och på lika villkor.

²² Se vidare http://guide.conecta.it/index.php/6._FLOSS-based_business_model. Open core används av bland annat MySQL som erbjuder en gratisversion av sin databasprogramvara samt en betalversion. Gratisversionen utgör grunden för båda varianterna och källkoden till den är fritt tillgänglig. Betalversionen innehåller flera funktioner som är proprietära och som MySQL tar betalt för.

²³ För en fördjupning i tanken bakom copyleft samt för- och nackdelar gentemot proprietär programvara hänvisas till Feller, J. et al, "Perspectives on Free and Open Source Software" s. 4 ff. samt s. 312 ff.

²⁴ Se dock ändringsrätten i 26 g § URL beskriven i avsnitt 5.1 nedan.

²⁵ URL 2 § 1 st, 12 § 2 st p. 2, 26 g § 2 st.

²⁶ För en redogörelse av hur bundenhetsmekanismen uppstår, se Andersson, M., a.a. s. 67 ff. och 4.3 nedan.

En copyleftlicens kan förenklat beskrivas som ett licensieringsalternativ där man istället för att hindra andra från att använda ens programvara vill garantera att så många som möjligt ska kunna använda den och bidra till att den utvecklas. Detta uppnås genom att körning,²⁷ kopiering, modifiering och tillgängliggörande tillåts så länge som det görs under samma licens och att den som i senare led mottar programmet garanteras samma rättigheter som den ursprungliga licenstagaren erhöll. Gemensamt för alla copyleftlicenser är nämligen att ingen licenstagare i senare led ska ges sämre rättigheter än den första licenstagaren. Att ändringar av programmet måste göras tillgängliga för andra bidrar enligt copyleftfilosofin även till att utvecklingen av programmet påskyndas.

Exempel: A skapar ett program som han, tillsammans med källkoden, gör tillgängligt under GPL. B finner programmet nyttigt, varvid han använder A:s källkod och lägger till en ny funktion i programmet. Om B vill göra den nya versionen tillgänglig för C så måste han licensiera användandet under GPL.

2.4 Svag och stark copyleft

Inom F/OSS-industrin har olika standardavtal för licensiering använts under lång tid. Bland dessa återfinns två huvudsakliga typer: copyleftlicenser samt licenser som saknar copyleftklausuler. Copyleftlicenserna kan vidare delas in i de som använder sig av svag respektive stark copyleft.

Svag copyleft skyddar modifikationer av programmet. Tanken är enkel – man vill möjliggöra distribution och användande av programmet, exempelvis i kombination med proprietär programvara, utan att det kombinerade programmet ska tvingas licensieras under en copyleftlicens. Man vill sprida användningen av det skyddade programmet, men tvingar modifieringar av det som görs tillgängliga för andra att licensieras under copyleftlicensen.

Exempel: A skapar och släpper ett bibliotek för kommunikation med LCD-skärmar under licensen LGPL.²⁸ B skriver ett program för felsökning av LCD-skärmar som i sin tur använder sig av A:s bibliotek. B behöver inte släppa sitt program under LGPL eller någon annan copyleftlicens. Hade han däremot modifierat A:s bibliotek hade denna modifikation behövt licensieras under LGPL.

Proprietär programvara kan därför integrera exempelvis LGPL-skyddade bibliotek utan att programmet i sin helhet måste göras tillgängligt under LGPL.²⁹

²⁷ "Körning" är sällan explicit uttryckt i licenserna men får anses ingå.

²⁸ GNU Lesser General Public License som använder svag copyleft, se <http://www.gnu.org/licenses/lgpl.html>.

²⁹ Användaren måste dock göras uppmärksam på att LGPL-skyddad källkod används, se vidare 4.1.2.

Den andra varianten av copyleftlicenser tar sikte på att ett program som använder sig av den licensierade programvaran måste göras tillgängligt under samma licens, så kallad stark copyleft. Vad som mer i detalj avses med ”använder sig av” diskuteras vidare i 5.3 nedan. Stark copyleft stannar inte, i motsats till svag copyleft, vid att kräva att modifikationer av den skyddade programvaran ska göras tillgängliga under samma licens som ”ursprungsprogramvaran”. Man kräver även att program som till del eller i sin helhet använder sig av den skyddade koden och görs tillgängliga för andra i sin helhet ska göras tillgängliga under ursprungslicensen. Denna skyldighet kommer i det följande att benämnas *copylefteffekten*.

Exempel: A skapar och släpper en programvara under GPL. B använder sig av en av modulerna från A:s källkod (utan att modifiera den) i sitt eget program. B:s programvara måste licensieras under GPL. Detsamma gäller även programvara som C skapar, där B:s källkod används (som ju i sin tur använder A:s).

Betänk situationen att ett företag ska utveckla en programvara som är tänkt att säljas under en proprietär licens. Tusentals arbetstimmar läggs på utvecklingen och man använder all den erfarenhet och expertis som man fått av att ha varit verksam inom denna marknad under lång tid. Denna erfarenhet uttrycks i unika metoder för att lösa problem och unika funktioner i programmet. När programmet väl börjar säljas så får företaget ett mejl, innehållande en varning om att programmet använder sig av GPL-skyddad kod och att det träffas av copylefteffekten. Givet att påståendet är sant så har företaget i detta läge tre val:

1. Ignorera varningen och bereda sig på att man blir stämd för intrånget (risk för rättegångskostnader samt skadestånd).
2. Dra tillbaka produkten och investera dyrbar tid till att ”sanera” källkoden från intrångsgörande kod (kostnad i form av ett stort antal arbetstimmar).
3. Följa licensen och göra programmet tillgängligt under GPL (uteblivna intäkter, företagshemligheter röjs och faller i konkurrenters händer).

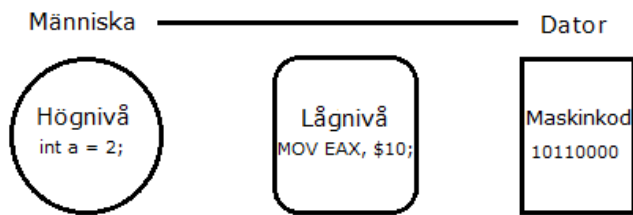
Oavsett valet så blir konsekvensen stora ekonomiska risker och potentiell badwill för företaget. Följaktligen är det av intresse för ett företag att ha kontroll över sin källkod och veta när copylefteffekten inträder. Denna fråga ska angripas efter en genomgång av de programmeringstekniska aspekterna.

3 Grundläggande programmeringsteknik

För att kunna se de juridiska problemen med mjukvarulicenser krävs en förståelse för programmeringsteknik. Det finns inte utrymme för en fullständig genomgång av ämnet inom ramen för detta arbete, varför den följande framställningen kommer att begränsas till den tekniska bakgrund som är relevant för uppsatsämnet. Den juridiska analysen i senare kapitel (se kapitel 4 och 5) kommer att bygga på denna bakgrund.

3.1 Vad är programmering?

En programmerare skriver källkod för att skapa programvaror. Precis som när människor ska göra sig förstådda sinsemellan så är valet av språk avgörande för att den eller det man talar med ska förstå. Källkoden kan därför skrivas i olika programmeringsspråk som utgör länken mellan människa och dator. Programmeringsspråk kan delas in i hög- respektive lågnivåspråk. Skillnaden kan förklaras med hur abstrakt språket är i förhållande till vad datorn förstår. ”Lågnivå” pekar på låg abstraktion, det vill säga svårt för en människa att förstå och vice versa med ”högnivå”. Se bilden nedan.



Programmeringsspråkets uppgift är att, precis som vilket annat språk som helst, skapa grundläggande regler och förhållningsramar för hur kommunikationen ska ske – vilka ord som kan användas, dess betydelse, struktur, med mera. Språket säkerställer att endast uttryck som en kompilator kan översätta till maskinkod kan användas.

En kodrad består därför av text och tecken som följer det specifika programmeringsspråkets regler och syntaxer. Okända uttryck vet kompilatorn inte hur den ska hantera och därför kan den inte omvandla koden till instruktioner som datorn förstår. Källkoden är uppbyggd i *algoritmer*, det vill säga grundläggande logiska funktioner som använder *variabler* ("är X större än Y?").³⁰ Algoritmer utgör beståndsdelarna i *funktioner*³¹ som besvarar mer abstrakta frågor. Exempelvis kan ett program uppmana en användare att mata in två tal för att besvara vilket av dem som är störst. En funktion skulle kunna besvara

³⁰ X och Y är i detta fall variabler som måste tilldelas värden och algoritmen utgörs av jämförelsen av dem.

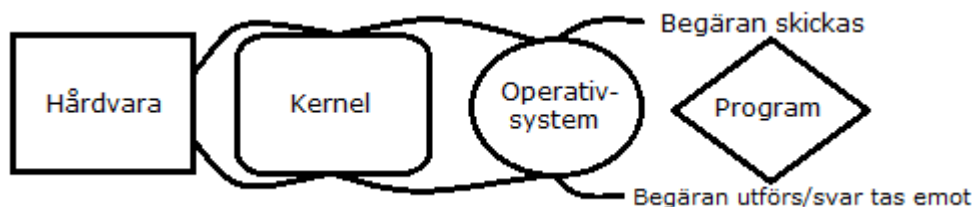
³¹ Beroende på vilket språk som används så kan en funktion även kallas för *metod* (inom objektorienterade språk).

frågan genom att ha en serie algoritmer som jämför de två talen för att ge resultatet till en annan funktion som skriver ut svaret på skärmen.

3.2 Vad kan programmeras?

I det följande skall en kort redogörelse ges för de typer av program som kan skapas.

Kärnan³² ("kernel") är den lägsta programmeringsbara³³ beståndsdel i datorn och är den komponent som hanterar hårdvaruresurserna (processor, minne och in-/utdataenheter). Kärnan måste få instruktioner om vad den ska ge hårdvaran för uppgifter att lösa. Dessa instruktioner får kärnan från operativsystemet, som den kommunicerar med genom ett *API*³⁴. Operativsystemet har ansvar för flera saker: exekvering av program, interrupts, minneshantering, virtuell minneshantering, filåtkomst, drivrutiner, grafiska gränssnitt, för att nämna några. Den virtuella minneshantering påverkar kärnan, som arbetar med två skilda områden: *kärnutrymme* (kernel space) och *användarutrymme* (user space) som är avskilda ifrån varandra. Kärnutrymmet är reserverat för att köra två typer av processer, kärnans egna samt drivrutiner för hårdvara. I användarutrymmet körs allt annat, bland annat all programvara som exekveras genom operativsystemet. Dessa behöver ta hårdvaruresurser i anspråk för att kunna lösa sina uppgifter. En sådan begäran skickas via operativsystemet till kärnan och kallas för *systemanrop*. Ett systemanrop kan exempelvis begära att få text utskriven på skärmen, registrera tangentnedtryckningar eller musförflyttningar, se bilden nedan.



Exempel: En användare vill öppna ett dokument och dubbelklickar på dess ikon. Operativsystemet tar hand om denna begäran, varpå flera systemanrop görs: (a) åtkomst begärs till den plats på hårddisken där programmet som ska öppna filen (exempelvis Word) finns lagrad, (b) åtkomst begärs till den plats på hårddisken där själva textfilen finns lagrad. Accepteras begäran kommer programvaran för att öppna filen att läsas in i användarutrymmet och textfilen läsas in. Ett nytt systemanrop kommer nu att göras via operativsystemet där en begäran att använda bildskärmen till att visa dokumentet genomförs.

³² Beskrivningen är inte allmängiltig då olika operativsystems kärnor arbetar på olika sätt. Den redogörelse som ges beskriver hur Linux kärna (som är av typen *module loading monolithic*) är uppbyggd.

³³ Med BIOS/EFI undantagna.

³⁴ Se http://sv.wikipedia.org/wiki/Application_Programming_Interface.

3.3 Hur programmeringsprocessen kan underlättas

Att skriva ett program från grunden innebär mycket jobb. För att spara tid kan man därför återanvända kod. Tanken bakom att använda redan skriven kod är enkel – det sparar tid att slippa uppfinna hjulet på nytt i varje program som skrivs. Källan till koden kan vara programmeraren själv, bestående i det som han har utvecklat i sina tidigare projekt. Har man inte själv skrivit någon kod som går att återanvända så finns det ytterligare källor som kan underlätta jobbet. Dessa källor kan samlas under beteckningen *extern källkod*.³⁵ Den första av dessa är sådan källkod som upphovsmannen gjort fri för alla att använda utan några begränsningar.³⁶ Den andra källan är den källkod som har gjorts tillgänglig under någon F/OSS-licens. Tack vare den oerhörda mängd extern källkod som finns på Internet är det osannolikt att det problem en programmerare står inför inte angripits eller lösts av någon som sedan gjort lösningen tillgänglig för andra.

Det man återanvänder kan vara så litet som en funktion, men oftare handlar det om metoder³⁷ eller *bibliotek*. Bibliotek är samlingar innehållande kod och information som syftar till att lösa specifika uppgifter. De är modulära i den bemärkelsen att de går att ”koppla på” en programvara för att få de funktioner som biblioteket i fråga erbjuder.³⁸ Biblioteken har en så kallad *headerfil* knuten till sig, i vilken kompilatorn återfinner de uttryck och symboler som är unika för biblioteket. Om man ser programmeringsspråkets regler och syntaxer som en faktabok så utgör en headerfil en påbyggnad till innehållsförteckningen. Den innehåller inte konkreta fakta utan talar bara om att de existerar.

Omfattningen av återanvändningen av kod beror på projektets karaktär och programmerarens egna preferenser. Som vi kommer att se nedan kan det även handla om risktaganden. Ett exempel är om programvaran innehåller företagshemligheter eller är avsedd att säljas när utvecklingen är klar, men man använt sig av copyleftlicensierad källkod.

3.4 Hur ett program blir till

En dator arbetar i grund och botten endast med att lösa matematiska uppgifter genom logik.³⁹ Människor använder logiken som ett hjälpmedel, men vi kommunicerar på ett mer abstrakt

³⁵ Med uttrycket ”extern källkod” avses källkod som tillförs och utvecklaren själv inte är upphovsman till.

³⁶ Jämför med *public domain* inom övriga upphovsrätten, http://en.wikipedia.org/wiki/Public_domain.

³⁷ Se ”Fast InvSqrt()” för ett exempel på hur tio rader kod kunde snabba upp 3D-beräkningar avsevärt, https://secure.wikimedia.org/wikipedia/en/wiki/Fast_inverse_square_root.

³⁸ Ett exempel är ett bibliotek som tar hand om kryptering/dekryptering av data som ska skickas/tas emot över ett nätverk. Denna funktion kan vara användbar om man utvecklar ett program som skickar/tar emot känslig data.

³⁹ Det kan i sammanhanget nämnas att man inom programmering räknar från noll till nio istället för ett till tio. Anledningen är att siffran noll har ett binärt värde, uttryckt i en byte är det 0000 0000. Därav att både Richard Stallmans ”fyra friheter” (se not 2) och GPL:s artiklar börjar sin numrering på noll.

sätt. En programmerare skriver kod i programmeringsspråk som en annan människa kan förstå.⁴⁰ En dator kan emellertid inte förstå detta språk.

För att omvandla källkoden till instruktioner som datorn kan köra behövs därför någon typ av "översättare". En sådan återfinns i form av en *kompilator*, en programvara som i olika steg omvandlar programmeringsspråk till maskinkod. Beroende på vilket programmeringsspråk kompilatorn är skriven för att översätta så kan vissa moment skilja sig något. Följande process beskriver förenklat en kompilering av källkod skriven i programmeringsspråket C.

- I. Källkoden läses in för *preprocessing*. Eventuella kommentarer i koden tas bort, programmet letar efter syntaxfel och felaktig kod och redovisar eventuella upptäckter för användaren.
- II. Resultatet (a) *kompileras* till *assemblerkod* (lågnivåspråk), samtidigt som (b) en *länkningsprocess* körs. Länkningsprocessen kopplar samman de symboler som är okända i programmeringsspråket men som finns definierade i header-filer knutna till bibliotek eller andra objektfiler som programmeraren hänvisat till.
- III. Assemblerkoden översätts till *maskinkod*.
- IV. Den färdiga programvaran sammanställs till exekverbara filer eller bibliotek.

Den ovan beskrivna metoden kommer att användas som utgångspunkt i den fortsatta framställningen. För att reda ut eventuella frågetecken till ska ges ett konkret exempel på hur kompilatorn genomför steg I i processen.

Exempel: Ett programs källkod består endast av följande kodrad: "int a = 5;" – vi deklarerar att variabeln/*integern*⁴¹ a tilldelas värdet 5. Vid kompilering av programmet så kommer raden att delas upp i sina minsta beståndsdelar, vilka blir fem till antalet: "int", "a", "=", "5" och ";".⁴² Varje beståndsdel måste nu översättas så att kompilatorn vet hur de ska användas vid en körning. Den använder sig av programmeringsspråkets regler för att veta detta. Hittar den ett uttryck eller en symbol som inte finns definierat så söker kompilatorn igenom header-filer från bibliotek eller andra objektfiler som finns länkade. Om en okänd symbol eller ett uttryck inte återfinns i någon av dessa filer så kommer kompilatorn att redovisa att ett fel har upptäckts, på vilken rad felet finns och vilken symbol som inte gått att översätta.

⁴⁰ Låt vara att man måste behärska programmeringsspråket i fråga för att kunna förstå det.

⁴¹ En integer deklarerar att 4 bytes i RAM-minnet ska reserveras för ett heltal.

⁴² I exemplet är "int" ett *keyword*, "a" en *variabel*, "=" är en *operator*, "5" är ett *decimaltal* och ";" är en *delimiter*.

3.5 Olika metoder för att sammanföra kod

Under 2.3 ovan har redogjorts för varför återanvändning av kod kan vara ett tidsbesparande och effektivt medel vid utveckling av mjukvara. Själva metoderna för hur detta kan ske samt deras innebörd ska förklaras i det följande, där hänvisningarna till kompileringsprocessen syftar till den i 3.4 ovan beskrivna. Beroende på i vilket läge den inlånade koden kommer i kontakt med den egna koden så används tre olika begrepp: *kompileringsprocessen* (compile time), för att beskriva att kontakten sker vid kompileringen (steg II.(a)), *länkningsprocessen* (linking time), när kontakten sker först vid länkningen (steg II.(b)), samt *run time* som tar sikte på när programmet exekveras av användaren. I det följande kommer ett antal olika typsituationer tas upp för att exemplifiera hur en programmerare på olika sätt kan använda sig av extern källkod. Listan är inte uttömmande utan har begränsat till de situationer som kan anses relevanta för den fortsatta juridiska analysen.

3.5.1 Inkorporering av extern källkod och makron

Den första situationen är den som uppstår när en programmerare kopierar källkod från en programvara till sin egen källkod, så kallad *copy-pasting*,⁴³ varefter källkoden kompileras till en enda programvara.

Exempel: Programmeraren A utvecklar programvaran X – en kalkylator. A vet inte hur han ska kunna beräkna kvadratroten ur ett tal, något som programvaran Y kan göra. A har tillgång till både Y och dess källkod. Han kopierar den metod från Y som löser räkneoperationen till sin egen källkod och gör vissa modifikationer så att den passar in i det egna programmet, varefter källkoden kompileras. A har nu en kalkylator som även kan beräkna kvadratrötter.

Ett makro används som en utrymmesbesparande åtgärd i källkod. Enkelt beskrivet så ger man ett alias till funktioner och metoder som används mer än en gång. Ett exempel är att man på flera platser i ett program vill använda en metod som sparar en skärmdump⁴⁴ av det som visas på skärmen. Används inte makron så måste metoden för skärmdumpen finnas på varje plats i källkoden där man vill kunna använda den. Med makron räcker det med att metoden skapas en gång för att sedan kunna anropas från valfri plats i programmet.

3.5.2 Statisk länkning

Begreppet avser den del av kompileringsprocessen som ovan beskrivits som länkningsprocessen. Statisk länkning innebär att man under länkningsprocessen kombinerar

⁴³ För en förklaring av termen, se http://en.wikipedia.org/wiki/Cut,_copy,_and_paste.

⁴⁴ Se <http://sv.wikipedia.org/wiki/Sk%C3%A4rmdump>.

den egna programvaran samman med extern kod som kan behövas för att köra den färdiga programvaran. Som anført ovan (se 3.3) rör det sig vanligtvis om olika bibliotek eller drivrutiner. Man ”bakar ihop” biblioteken med det egna programmet, vilket skapar utrymmesmässigt stora filer. Resultatet blir en enda körbar fil. En stor fördel med tillvägagångssättet är att programmeraren vet att alla de bibliotek som behövs för ett fullt fungerande program finns med. Det finns två huvudsakliga tillvägagångssätt för att använda sig av statisk länkning. Det första är att ett färdigkompileerat bibliotek länkas utan att det först modifieras. Det andra är programmeraren modifierar bibliotekets källkod för att anpassa det till det egna programmet. Gemensamt för de båda metoderna är att de föregås av en granskning av bibliotekets källkod för att se hur och på vilket sätt biblioteket kan anropas. Vid kompileringen så kommer en statiskt länkad fil att, modifierad eller i sin helhet, vid länkningsprocessen att integreras i den sammanställda programvaran. Vid en körning av det sammanställda programmet kommer både programmet och de statiskt länkade filerna att läsas in tillsammans i datorns arbetsminne.

Exempel: Programmet A använder sig av biblioteken X och Y. Vid kompilering används statisk länkning, varför programmet A kommer bestå av en fil⁴⁵ som inkluderar de båda biblioteken.

3.5.3 Dynamisk länkning

Dynamisk länkning kan på många sätt ses som motsatsen till statisk länkning. Istället för att sammanföra hela bibliotek med programvaran vid länkningsprocessen så inkluderas endast referenser till biblioteken. Biblioteken som referenserna länkar till läses in först under runtime, det vill säga när programmet exekveras. Innebörden av detta är att varken källkoden till den dynamiskt länkade filen eller den faktiska filen finns i programmet som använder sig av den. Dessutom läses de vid en körning in i separata minnesområden (eftersom de inte körs samtidigt eller av samma process). En väsentlig skillnad mot statisk länkning är därför att biblioteken inte nödvändigtvis måste distribueras tillsammans med programvaran. Det blir upp till användaren som vill köra programmet att tillse att denne har de bibliotek som krävs. Ett vanligt sätt att lösa detta på är att programmeraren inkluderar ett steg i installationsprogrammet som erbjuder nedladdning, eller en hänvisning i en readme-fil som talar om var biblioteken går att hämta hem.

För att illustrera skillnaden kan en parallell dras till källhänvisningar i ett dokument. En källhänvisning kan exempelvis bestå av en länk till en webbsida där mer finns att läsa om ett uttrycks betydelse. Det som finns i dokumentet är endast en referens till webbsidan och

⁴⁵ Vars storlek i exekverbar form i princip blir (A+X+Y).

innehållet från den återges inte i dokumentet. Källhänvisningen talar endast om var informationen finns att återfinna, inte vad webbsidan faktiskt innehåller.

3.5.4 Remote procedure call⁴⁶ [hädanefter “RPC”]

RPC är en teknologi som möjliggör att program som körs på olika datorer utbyter information med varandra. Systemet bygger på att den ena datorn – klienten – kör en programvara som har en *IDL-fil*⁴⁷ knuten till sig. Klienten skickar en begäran till en annan dator – servern – som i sin tur kör ett program som klienten vill komma åt. IDL-filen fungerar på ett liknande sätt som headerfilen beskrivet ovan (se exemplet under 2.3). Likheten är att filen innehåller en lista över vad som kan anropas hos servern, men med den skillnaden att IDL-filen inte berättar något om symboler eller liknande. Den talar endast om för datorn att den ska leta efter innehållet på annat håll och hur den ska kommunicera för att få del av denna information.

⁴⁶ RPC är en underkategori till *inter-process communication*, se http://en.wikipedia.org/wiki/Inter-process_communication.

⁴⁷ Se http://en.wikipedia.org/wiki/Interface_description_language.

4 GPL

Det följande kapitlet kommer att redovisa GPL:s innehåll i de delar som är relevanta för uppsatsämnet, det vill säga vilka rättigheter och skyldigheter licensen medför, när dessa uppstår och deras innebörd. En mer ingående analys kommer att ske i det följande kapitlet. Slutligen kommer frågan om licensens giltighet i Sverige att behandlas.

4.1 Vad reglerar GPL?

GPL reglerar villkoren för att man ska få utnyttja den skyddade programvaran på ett sätt som utan licensen skulle anses intrångsgörande (se nedan 4.2). De viktigaste regleringarna är vad licensen skyddar och när bundenhet för licenstagare uppstår. Den tar även upp flera regleringar (exempelvis friskrivningsklausulerna i artikel 11-12) som är av juridiskt intresse men som faller utanför ramen för denna uppsats. I det följande kommer fokus därför att riktas på det för uppsatsämnet relevanta, nämligen vad man enligt avtalet får för rättigheter och under vilka förutsättningar skyldigheten att göra den egna källkoden tillgänglig under GPL uppstår.

4.1.1 Omfattning och bundenhet

Licensens omfattning regleras i artikel 0 andra stycket:

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

Licensen gäller endast kopiering, distribution ("offentliggörande" eller "spridning") samt sådana modifieringar ("bearbetningar") av programmet som görs offentliga. En slutsats av detta är att en person som endast kör programvaran inte är bunden av GPL. Bearbetningar som inte offentliggörs omfattas inte av copylefteffekten, vilket framgår genom ett motsatsslut av artikel 2 och 3. Den för uppsatsämnet relevanta bearbetningssituationen är därför den där bearbetningen också offentliggörs. Med anledning av detta kommer "modifiering" i det fortsatta att avse kombinationen av modifiering och offentliggörande om något annat inte anges särskilt.

4.1.2 Rättigheter och skyldigheter

Artikel 1 ger de grundläggande rättigheterna och har följande lydelse:

You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Första stycket ger licenstagaren rätt att kopiera och distribuera kopior av programmets källkod under förutsättning att han gör en potentiell mottagare medveten om att koden är skyddad under GPL. Andra stycket medger att licenstagaren kan ta betalt för de utgifter som kan uppstå om en licenstagare önskar erhålla programvaran på exempelvis CD-ROM.⁴⁸

Artikel 2 innehåller GPL:s copyleftklausul som är i fokus för denna studie lyder:⁴⁹

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

[...]

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

[...]

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

⁴⁸ Vidare kan man enligt GPL ta betalt för tillhandahållandet av fysiska kopior av manualer, support, och dylikt. Se preambelns andra stycke jämte artikel 1 andra stycket.

⁴⁹ Artikeln motsvaras av artikel 5.c) i GPLv3.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

Du medges rätt att modifiera det GPL-skyddade programmet och distribuera dina modifieringar, men modifikationerna och program som innehåller det licensierade programmet måste licensieras under GPL i sin helhet. För att förstå när denna copylefteffekt inträder måste artikeln läsas tillsammans med första stycket i artikel 0:

This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language.

Uttrycket "a work based on the Program" innefattar det GPL-licensierade programmet eller "any derivative work under copyright law". Sammanfattningsvis är det skyddas som enligt licensens ordalydelse följande:

1. Programmet som licensieras under GPL ("ursprungsprogrammet"),
2. Delar av ovanstående,
3. "Derivative works" av ursprungsprogrammet eller delar av det, samt
4. "Derivative works" i senare led av ovanstående.⁵⁰

Följden av detta blir att program som innehåller något av ovanstående träffas av copylefteffekten och måste i sin helhet göras tillgängligt under GPL. Huruvida detta påstående kan anses förenligt med svensk rätt kommer att tas upp i kapitel 5.

Artikel 3 är en utökning av rättigheterna och skyldigheterna i artikel 1 och 2 och tar sikte på situationen att man vill kopiera eller offentliggöra en GPL-skyddad programvara i objektods- eller körbar form.⁵¹ Villkoret är att licenstagaren bifogar källkoden i maskinläsbar form (vanligen i form av en textfil) eller erbjuder sig att ge ut en fysisk kopia⁵² av källkoden i minst tre år och i övrigt följer licensen.

Artikel 4 tar upp konsekvensen av att licensen inte följs:

⁵⁰ Definitionen i artikel 0 av "work based on the Program" skulle även kunna tolkas som att den omfattar programmet i sin helhet, det vill säga att termen även träffar ursprungsprogrammet. Det förefaller dock som en ologisk konstruktion och torde syfta på bearbetningar av såväl ursprungsprogrammet som i senare led.

⁵¹ "Motsatsen" till källkod, nämligen assemblerkod eller maskinkod. Se kompileringsprocessen i 3.4 ovan.

⁵² Det vill säga en textfil innehållandes källkod på exempelvis ett USB-minne, en diskett eller en CD-ROM.

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

För det första så medför ett kontraktsbrott, oavsett dess storlek, att licensen omedelbart återkallas från den kontraktsbrytande licenstagaren. Skäligheten i detta har ifrågasatts rörande gåvor⁵³ men torde bedömas som skäligt inom upphovsrätten.⁵⁴ För det andra så medför en sådan återkallelse inte att licenstagare i senare led får sin licens återkallad. Som tidigare beskrivits (se 2.2 ovan) så inträder URL:s bestämmelser när licensen återkallas.⁵⁵ Eftersom licensen återkallats står licenstagaren utan rätt att använda programvaran och dennes användande av programvaran utgör därför intrång i licensgivarens immaterialrätt.

Exempel: A skapar ett program som han, tillsammans med källkoden, gör tillgängligt till allmänheten genom att licensiera användandet med GPL. B finner programmet nyttigt och han gör en förbättring i det. B väljer att göra den uppdaterade versionen av A:s källkod tillgänglig enligt en annan licens än GPL. B bryter därmed mot GPL och hans licens till A:s program upphör omedelbart. B innehar upphovsrätten till sin förbättring, men inte till A:s ursprungliga verk (s.k. beroende upphovsrätt), varför hans publicering utgör ett brott mot URL.

Den sista artikeln som är av direkt intresse för ämnet är artikel 5 som lyder:

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

Artikeln konstaterar att du inte är tvingad att acceptera avtalet i och med att du inte signerat något. Däremot är bundenhet till licensen ett måste för att du ska få vidta åtgärder som utan licensen är att betrakta som upphovsrättsintrång. Vi kommer att återvända till denna artikel under avsnitten 4.3 och 5.1.

⁵³ Hellner et al. anser att en gåvotagares avvikelse från gåvovillkoren måste vara väsentligt för att kunna innebära en omedelbar återgång av gåvan. Åsikten motiveras med att gåvovärdet kan vara högt och innebära långtgående konsekvenser för gåvotagaren, se Hellner et. al. a.a. s. 268 och 4.2.1 nedan.

⁵⁴ Det nu sagda måste emellertid ställas emot förarbetena till URL där det anförs att "omedelbart upphörande får anses normalt", se prop. 1988/89:85 s. 15.

⁵⁵ Ett exempel ur praxis när denna modell tillämpats är NJA 1997 s. 109, *Partyman*.

4.1.3 Innebörden av ”derivative work”

GPL definierar aldrig vad ”derivative work” innebär. Termen är hämtad direkt från amerikansk upphovsrättslagstiftning.⁵⁶ Rent språkligt kan uttrycket översättas till ”bearbetat verk”, ett uttryck som återfinns i 4 § URL. Frågan är emellertid om de juridiska begreppen har samma innebörd. Som läsaren förstår så är detta en komplicerad fråga att besvara.

Enligt artikel 0 ska uttrycket förstås i enlighet med upphovsrättslig lagstiftning. Det finns åtminstone två olika tolkningar av detta. Den ena tolkningen är att uttrycket ska förstås i enlighet med amerikansk rätt.⁵⁷ Detta kan ha varit FSF:s intention i och med att man valt en terminologi som hämtats direkt från den amerikanska upphovsrätten. Den andra tolkningen är att begreppet ska tolkas i ljuset av *tillämplig* upphovsrättslig lagstiftning. Eftersom GPL saknar lagvalsklausul så kan nationella upphovsrättsliga bestämmelser bli tillämpliga och därmed också olika nationella tolkningar av innebörden.

Det har inom doktrinen påståtts att ”bearbetning” har en vidare betydelse inom europarätten⁵⁸ än ”derivative work” har inom den amerikanska rätten.⁵⁹ Inom svensk doktrin har det anförts att ”begrepp[et] borde i stort kunna jämföras med det svenska begreppet bearbetning, men det är möjligt att det kan finnas situationer där en tillämpning av de båda begreppen skulle leda fram till olika resultat”.⁶⁰ Då det inte finns möjlighet till någon vidare utredning av detta problem så utgår författaren från den citerade tesen i det följande.

4.1.4 Kompatibla licenser

Det framgår inte uttryckligen av GPL men programvara som licensierats under licensen får endast kombineras med annan programvara som licensierats under en *licens som är kompatibel med GPL*. Detta kan utläsas av artikel 2 och kravet på att det modifierade programmet som innehåller GPL-skyddad kod måste göras tillgängligt genom att licensieras under GPL.⁶¹ För att villkoret ska kunna uppfyllas så krävs att alla programmets beståndsdelar använder sig av en licens som tillåter att de släpps under GPL. Att en licens är kompatibel med GPL innebär helt enkelt att den tillåter att det nya programmet görs tillgänglig under GPL.⁶²

⁵⁶ Title 17 of the United States Code, Copyright Act, chapter 1, § 101.

⁵⁷ Se Välimäki, M., *GNU General Public License and the Distribution of Derivative Works*, Journal of Information, Law & Technology (JILT) 2005, s. 5.

⁵⁸ Syftande på Datorprogramdirektivet artikel 4.b) som talar om att ”varje annan ändring” är en bearbetning.

⁵⁹ Se Välimäki, M., a.a. s. 6.

⁶⁰ Olofsson, J., *Upphovsrättsliga aspekter på licenser för fri programvara och öppen källkod*, IRI rapport 2003:1 s. 41.

⁶¹ <http://www.gnu.org/licenses/old-licenses/gpl-2.0-faq.html#TOCWhatDoesCompatMean>.

⁶² För en lista över kompatibla licenser, se <http://www.gnu.org/licenses/license-list.html#GPLv2>. Här kan noteras att GPLv3 är kompatibel med fler licenser än GPLv2.

För att dra en parallell kan en GPL-licens betraktas som ett dominant arvsanlag och andra licenser som så väl dominanta som recessiva arvsanlag. Ett villkor för att GPL ska kunna kombineras med en annan licens är att den är recessiv gentemot GPL. En slutsats som därmed kan dras är att GPL-skyddad kod inte kan implementeras i en proprietär programvara utan att antingen begå ett brott mot licensen eller begå ett upphovsrättsligt intrång.

Exempel: Programmet A släpps under GPL. Programmet B licensieras under LGPL. Programmen A och B kan enligt GPL kombineras till programmet C eftersom LGPL är receptiv gentemot GPL. C måste då släppas under GPL.

4.2 Utgör GPL ett giltigt avtal?

Om en svensk domstol skulle behöva tolka GPL så skulle avtalsrättsliga aspekter aktualiseras, nämligen om (1) GPL är ett giltigt avtal och om (2) bundenhet har uppstått.

Huvudregeln i svensk rätt är att parterna är fria att ingå avtal om vad som helst och att de själva får avgöra vilka regler och villkor som ska gälla mellan dem. Vissa regler finns dock. Exempelvis måste parterna ha rättshandlingsförmåga för att kunna ingå avtal och avtalet får inte strida mot god moral (*pactum turpe*) eller lag (importförbud, försäljning av vapen, m.m.). Det kan även finnas formkrav som är tvingande för att ett avtal ska anses giltigt. Bryts det mot någon av dessa regler så kan avtalet vara ogiltigt. Vidare kan en domstol med stöd av 36 § AvtL⁶³ jämka eller ogiltigförklara hela eller delar av avtal som bedöms som oskäliga.

Mjukvarulicenser rör sig i det rättsliga gränslandet mellan upphovsrättslagen och avtalslagen och båda dessa områden präglas av avtalsfrihet. En upphovsman som använder GPL utnyttjar dispositiviteten som upphovsrättslagen medger. Något generellt hinder mot att GPL skulle vara ett giltigt avtal kan därför inte sägas finnas.⁶⁴ Det finns emellertid några tvingande regler rörande datorprogram. Dessa är rätten att framställa exemplar som är nödvändiga för att programmet ska kunna användas (26 g § 1 st URL), rätten att framställa en säkerhetskopior (26 g § 2 st URL) samt dekompileringsrätten (26 h § URL). Dessa rättigheter kan inte inskränkas. Det GPL gör är motsatsen, nämligen att utöka dessa rättigheter genom att ge licenstagaren en obegränsad rätt att kopiera programvaran och en garanterad tillgång till källkoden (vilket tar bort behovet av dekompilering). Det bör dock påminnas om att endast en domstol slutligt kan besvara frågan om GPL är ett giltigt avtal och att svensk praxis hittills saknas. GPL har dock varit i bruk i över 20 år och kan idag betraktas som ett standardavtal på

⁶³ Lag (1915:218) om avtal och andra rättshandlingar på förmögenhetsrättens område.

⁶⁴ Det kan emellertid ifrågasättas huruvida licensens samtliga klausuler kan anses vara skäliga, främst med hänvisning till friskrivningsklausulen i artikel 11-12. Klausulerna faller emellertid utanför uppsatsämnet varför läsaren hänvisas till Pawlo, M., *Något om fri programvara och öppen källkod – nya licenstyper för datorprogram*, NIR 4/2002, s. 387 ff. och Andersson, M., *Open source – Ur ett praktiskt juridiskt perspektiv*, Det IT-rättsliga observatoriets rapport 54/2002 s. 77.

F/OSS-området. Att någon praxis inte uppstått skulle därför kunna vara ett tecken på att dessa licensavtals giltighet inte ifrågasätts utan accepteras av marknadsaktörerna. Två tyska domstolar har dock konstaterat att GPL är ett giltigt avtal och accepterat dess bundenhetsmekanism.⁶⁵ Sammanfattningsvis har författaren svårt att tro att GPL inte skulle anses vara ett giltigt avtal om frågan prövades av en svensk domstol.

4.3 Uppstår bundenhet för licenstagaren och i sådant fall – när?

Utgångspunkten i avtalslagen är att ett bindande avtal uppstår genom att anbud och accept avges (1 § AvtL). Denna modell fungerar bra för avtal som bygger på skriftliga kontrakt. Det överlägset vanligast förekommande sättet att få tillgång till F/OSS-mjukvara på är emellertid genom att ladda ner programvaran via Internet utan att något kontrakt undertecknas. En förutsättning för att GPL ska kunna bli bindande är därför att licenstagaren fått möjlighet att ta del av avtalsinnehållet innan det att avtalet ingås.⁶⁶ Den hemsida som programmet laddas ned ifrån förses därför ofta med en upplysning om att programmet skyddas av GPL. Även själva filen som laddas ned (oftast ett filarkiv innehållandes flera filer) kan innehålla en fil som hänvisar till att programmet skyddas av GPL. Båda dessa situationer kan likställas med användandet av en referensklausul. Utgångspunkten för dessa i svensk rätt är att det räcker med att ett standardavtal gjorts lättillgängligt, exempelvis genom att det tydligt hänvisas till en hemsida där avtalet finns, för att de ska vara giltiga.⁶⁷ Detta krav lär uppfyllas i de allra flesta fall, men det lämnar ändå utrymme för osäkerhet (exempelvis att den tänkta licenstagaren hävdar att han inte sett hänvisningen och därför inte gjorts medveten om att GPL använts).

En annan möjlighet är därför att tvinga användaren att godkänna licensvillkoren innan han kan ladda ned programmet, så kallad *web-wrap*. Användaren får då se licensavtalet följt av att han måste klicka i en ruta där han godkänner avtalet för att kunna ladda ned programmet. Den fjärde möjligheten, *click-wrap*, är lik web-wrapping men användaren uppmärksammas på licensen och måste godkänna den under installationen av programvaran. Båda kontraktstyperna kan samlas under termen *adhesionskontrakt* och anses giltiga.⁶⁸

Licensen anger att du inte är tvingad att acceptera den (se ovan 4.1.1). Om en användare inte är bunden av licensen innebär det dock att upphovsrättslagens regler om upphovsmannens exklusivitet inträder. Detta medför att allt agerande som inte sker med stöd

⁶⁵ Landgericht München I, Entscheidung vom 19. Mai 2004, 21 O 6123/04 och Landgericht Frankfurt am Main vom 6. September 2006, 2-6 O 224/06.

⁶⁶ Ramberg, C., Ramberg, J., *Allmän avtalsrätt*, 7 u, Norstedts Juridik AB, 2007, s. 142.

⁶⁷ Ramberg, C., Ramberg, J., a.a. s. 143.

⁶⁸ Se Andersson, M., a.a. s. 67 ff.

av lagens undantag innebär upphovsrättsintrång. För att ha rätt att modifiera och distribuera programmet eller bearbetningar av det är det därför ett krav att licensen är bindande (se dock 5.1). Om någon använder en GPL-skyddad programvara på ett sätt som kräver bundenhet till licensen torde därför bundenhet genom konkludent handlande ha uppstått.

Tidpunkten för avtalets ingående beror enligt författaren på vilket sätt som licenstagaren görs medveten om licensen på. Den senaste möjliga tidpunkten för avtalets ingående bör vara när en användare uppvisar konkludent handlande enligt ovan. Är det web- eller click-wrap så bör avtalstidpunkten vara då man godkänner licensen i samband med nedladdning eller installation. Det är alltså innan dessa tidpunkter som användaren måste göras medveten om att GPL tillämpas och fått möjlighet att granska dess innehåll för att bundenhet till licensen ska kunna uppstå. I såväl click- som web-wrap-situationer bör det oftast inte råda något tvivel om att bundenhet uppstått och tidpunkten för det. I de andra situationerna krävs en bedömning av det specifika fallet för att kunna besvara frågan.⁶⁹

Det bör slutligen påminnas om att ett huvudsakligt syfte med copyleftlicenser i allmänhet (se 2.3 ovan) är att den som får gratis tillgång till programvaran ska återgälda detta genom att ”ge tillbaka det man fått”. Detta sker genom att man gör modifikationer och program som använder sig av programmet tillgängliga på samma vis. Detta stipuleras i andra stycket i GPL:s preambel⁷⁰ och bör betraktas som en del av avtalet – eller i vart fall som en viljeyttring – av betydelse vid tolkning av avtalets innehåll. Genom att den finns tillgänglig vid avtalets ingående så är detta något som är förutsebart för såväl licensgivare som licenstagare.

4.4 Vem är din avtalspart?

En annan fråga som kan vara av vikt är vilka som är avtalsparter. Frågan kan spela roll för avtalstolkningsfrågan i de situationer där den ena parten kan anses svagare än den andra. Den som väljer att licensiera en programvara under GPL gör detta ”i blindo” i den bemärkelsen att han inte kan förutse vem licenstagaren kommer att vara. Det kan vara en privatperson i Egypten lika gärna som ett företag i Kina. Vidare blir licensgivaren sällan medveten om att en licenstagare förbundet sig till licensen. Normalfallet vid avtalsingåenden är ju annars att respektive part blir varse den andres anbud/accept. Därtill skiljer sig avtalssituationen från den traditionella genom att licenstagaren är anonym för licensgivaren.

⁶⁹ Andersson hävdar dock att det inte spelar någon roll ifall licensavtalet inte blivit en del av avtalet. Se Andersson, M. a.a. s. 68.

⁷⁰ Mer specifikt avses andra meningens i andra stycket ur preambeln: ”Our General Public Licenses are designed to make sure that you have the freedom to [...] receive source code, [...] that you can change the software or use pieces of it *in new free programs*” (författarens kursivering).

När ett standardavtal används och licensgivaren inte vet vem licenstagaren är så kan det vara svårt att finna någon partsavsikt. Det finns dock en undantagssituation. FSF arbetar nämligen aktivt för att få upphovsmän att skriva över rättigheterna till sina program till FSF med hänvisning till att de som en större organisation bättre kan tillvarata upphovsmännens intressen i händelse av konflikter rörande brott mot GPL. Situationen som kan uppstå är därför att en licenstagare i en tvist kommer att möta FSF som motpart, som i sin tur formulerat avtalet. Detta skulle kunna aktualisera *oklarhetsprincipen (contra stipulatorem)*, innebärande att avtalet ska tolkas till nackdel mot den part som författat det och således till nackdel för FSF. Argumentet för denna åsikt är att FSF författat avtalet och att avtalet därmed kan anses vara ett uttryck för deras vilja.⁷¹ Denna tolkningsprincip ska dock ställas mot *specifikationsprincipen* som innebär att om det råder oklarhet kring vilka rättigheter som överlåtits eller upplåtits ska omfattningen av avtalet tolkas restriktivt till upphovsmännens fördel.⁷²

Slutligen ska även nämnas att om ett företag erbjuder programvaror under GPL så kan rättsförhållandet komma att betraktas som ett näringsidkare-/konsumentförhållande, givet att man uppfyller legaldefinitionen för de båda begreppen.⁷³ Ett sådant rättsförhållande medför åtminstone två effekter som man bör känna till. För det första så påverkas lagvalet enligt artikel 5.3 Romkonventionen på så sätt att det lands lag i vilket konsumenten har sin vanliga vistelseort blir tillämpligt. För det andra så kan AVLK⁷⁴ bli tillämplig. I 10 § AVLK anges att avtalsvillkor i konsumentförhållanden som inte varit föremål för individuell förhandling ska tolkas till konsumenternas förmån. Denna tolkningsregel skulle kunna medföra andra utfall än en objektiv tolkning av avtalsinnehållet ger. Det är dock endast en spekulation tills dess att en domstol prövat frågan.

⁷¹ Lindberg, A., Westman, D., *Praktisk IT-rätt* (cit. Lindberg/Westman) s. 387.

⁷² Lindberg/Westman s. 251.

⁷³ Se exempelvis prop. 1984/85:110 s. 139 ff., prop. 1994/95:17 s. 87 f., samt prop. 1989/90:89 s. 59 ff.

⁷⁴ Lag (1994:1512) om avtalsvillkor i konsumentförhållanden.

5 Analys

Som visat ovan (se 4.1.2) skapar GPL ett antal olika situationer då copylefteffekten inträder. I copyleftsammanhang skapar detta frågan: orsakar allt användande av GPL-skyddad kod att copylefteffekten inträder? Kan man tänka sig situationer där användande av GPL-skyddad kod kan ske utan att copylefteffekten inträder? Det saknas praxis på när en bearbetning anses ske enligt GPL. Den följande analysen avser därför att ta upp några av de frågor som kan aktualiseras vid en bedömning av denna fråga.

5.1 Vad skyddas enligt URL?

Svensk rätt ger som tidigare nämnt (se 2.1 ovan) upphovsmannen till ett verk ensamrätt till detta. Vad krävs då för att få detta skydd? I korthet är inte kravet ”verkshöjd” som inom andra områden inom upphovsrätten, utan verket behöver endast vara ”originellt”.⁷⁵ I kapitel 2 nämndes att inte logik eller grundläggande algoritmer skyddas. Ett exempel är en grundläggande BubbleSort-algoritm.⁷⁶ Den kan uttryckas på olika sätt och variabler kan ges olika namn men den bakomliggande logiken är alltid densamma eftersom uppgiften bara kan lösas på ett sätt. Innebörden är att funktionen som sådan inte kan anses upphovsrättsligt skyddad, även om dess uttrycksform kan uppnå den originalitet som krävs.

Skyddet som erhålls har en yttre gräns kring vad det omfattar. Vid bedömningen om ett intrång föreligger slås först fast vad som skyddas och därefter hur närliggande kopian är. Man kan tänka sig att bedömningen sker på en skala där ”intrång” utgör den ena extrempunkten och där ”självständigt verk” utgör den andra. Om ett påstått intrång är identiskt med originalet är det lätt att se likheterna och därmed också att bedöma det förra som ett intrång. Verkligheten är inte alltid så enkel, utan många gånger man sig i en gråzon kring om det är ett intrång eller inte.

Undantag från ensamrätten finns i 2 kap. URL. Undantagen som är relevanta för datorprogram är främst citaträtten i 12 § URL, ändringsrätten i 26 g § URL och dekompileringsrätten i 26 h § URL. Den sistnämnda är av mindre i F/OSS-sammanhang eftersom källkoden alltid finns fritt tillgänglig och dekompilering därmed inte behövs. Om någon bryter ensamrätten utan att något av undantagen är tillämpligt så anses den personen göra intrång i upphovsmannens immaterialrätt. Om någon tillämpar ett av undantagen men överskrider den rätt som medges så innebär även det att ett intrång begås.

⁷⁵ Artikel 1.3 i datorprogramdirektivet, se 1.4 ovan.

⁷⁶ För en förklaring av termen och ett exempel på dess funktion, se http://en.wikipedia.org/wiki/Bubble_sort.

En citering med stöd av 22 § URL måste följa god sed på området samt motiveras av ändamålet,⁷⁷ exempelvis att det används för att belysa, kritisera eller utveckla det citerade.⁷⁸ Varken partiella eller hela återgivningar av andras verk anses förenliga med citaträtten om syftet är att krydda eller förgylla det egna verket.⁷⁹ Därtill ska den citerade upphovsmannen anges. Med tanke på hur datorprogram är uppbyggda är det svårt att se hur citering kan ske på detta område på ett sätt som överensstämmer med god sed. Problemet kommer att undersökas vidare i 5.2.1 nedan.

Ändringsrätten i 26 g § 1 st URL är också av intresse. Att köra ett GPL-skyddat program omfattas inte av licensen, men modifiering anses enligt artikel 5 utgöra en presumtion för att man accepterat bundenhet till licensen. Därtill påstås i samma artikel att all modifiering (med mera) är lagstridig utan licensen. Du får emellertid köra ett GPL-skyddat program utan att acceptera bundenhet till licensen. Samtidigt ger dig 26 g § 1 st URL rätten att utföra ändringar samt rättelser av fel i program om syftet är att det ska göra så att programmet kan användas för sitt avsedda ändamål. Detta torde alltså kunna utgöra en situation där ditt utövande av en lagenlig rätt gör att ditt handlande inte kan anses vara ett tecken på att du accepterat bundenhet till GPL.

5.2 Vad utgör en bearbetning enligt URL?

Termen ”bearbetning” återfinns i 4 § URL. Paragrafen innehåller en negativ definition där ett verk som inte är ”självständigt” eller kan anses uppkommet ”i fri anslutning” till ett verk är att betrakta som en bearbetning. Därtill måste verket som bearbetats uppnå verkshöjd (originalitetskravet) för att bestämmelsen ska aktualiseras. Om ett program är en bearbetning av ett annat så är upphovsrätten till bearbetningen en så kallad beroende upphovsrätt. Med detta menas att man förvisso erhåller upphovsrättsligt skydd för sin bearbetning (givet att den uppfyller originalitetskravet) men den får bara användas med den ursprungliga upphovsmannens samtycke. I det konkreta fallet med GPL innebär detta att en bearbetning av ett GPL-skyddat program träffas av copylefteffekten.

Anta att en programmerare skriver programmet X bestående av flera hundra tusen rader kod. Han infogar sedan fem rader kod från det GPL-skyddade programmet Z. Kan programmet X betraktas som en bearbetning av Z? Gränsdragningen mellan vad som är en fri respektive beroende bearbetning är ett av upphovsrättens största problem.⁸⁰ Det går inte att

⁷⁷ Koktvedgaard, M., Levin, M., *Lärobok i Immaterialrätt*, 8 u, Norstedts Juridik AB, 2004, s. 187.

⁷⁸ Se Olsson, H., ”Upphovsrättslagstiftningen” tillgänglig på Zeteeo (version den 1 maj 2011), kommentaren till 2 kap. 22 §.

⁷⁹ Koktvedgaard, M., Levin, M., a.a. s. 187.

⁸⁰ Koktvedgaard, M., Levin, M., a.a. s. 71, 160 f.

svara generellt på när en bearbetning av ett datorprogram kan anses föreligga, utan bedömningen får göras från fall till fall med hjälp av en *intrångsbedömning*.⁸¹ En sådan bedömning kan förenklat beskrivas som en sakkunnig jämförelse mellan programmets källkod. Det avgörande för bedömningen är om det ursprungliga verkets *identitet* framträder i det påstått intrångsgörande verket. Någon närmre precisering av vad begreppet innebär går inte att göra, utan *in casu*-bedömningar krävs.

Förutom gränsdragningen mellan fria och beroende bearbetningar så finns även möjligheten att man genom att sammanfoga upphovsrättsligt skyddade verk åstadkommit ett samlingsverk (5 § URL). Precis som med bearbetningarna är de en beroende upphovsrätt. Den största skillnaden mellan de båda är att de verk som samlingsverkets innehåller måste förhålla sig passivt till varandra (som en encyklopedi). För datorprogram innebär detta att ett program bestående av flera moduler som interagerar med varandra inte kan utgöra ett samlingsverk.⁸²

5.3 Vad utgör en bearbetning enligt GPL?

Licensen talar om att copylefteffekten inträder om ett program ”innehåller” eller ”härrör” från ett GPL-licensierat program. Om vi börjar med begreppet ”innehåller” så syftar det på att programmet innehåller hela eller delar av ett GPL-licensierat program i antingen källkods- eller maskinkodsform. Om vi jämför det med det nyss sagda om det upphovsrättsliga skyddets omfattning så kan konstateras att detta svårligen kan anses överensstämma med svensk rätt. Anledningen till detta är att även om en källkod skulle innehålla en del av ett GPL-skyddat program så är det inte säkert att den inlånade biten anses vara upphovsrättsligt skyddad. Ett tänkbart exempel vore att en algoritm lånades in men där exempelvis variabelnamn byttes ut. Att kategoriskt påstå att ett sådant program ”innehåller” det GPL-skyddade programmet och därmed ska träffas av copylefteffekten framstår som onyanserat och direkt felaktigt.

Begreppet ”härrör från” tar sikte på modifikationer av ursprungsprogrammet, det vill säga frågan om när en bearbetning av ett program uppstår. Inledningsvis bör sägas att vad som faktiskt utgör en bearbetning i GPL:s mening är en synnerligen omtvistad fråga som ännu inte har någon lösning. Enligt FSF:s mening utgör i stort sett allt som använder GPL-skyddad kod ett ”derivative work” och ska därför träffas av copylefteffekten. Ett undantag är att GPL-skyddade plug-ins får användas av proprietär programvara.⁸³ Så länge dessa plug-ins endast anropas med grundläggande anrop (fork, exec, kill, med flera) anses de vara separata program

⁸¹ Se Lindberg/Westman s. 288 ff.

⁸² Se Lindberg/Westman s. 230 f.

⁸³ <http://www.gnu.org/licenses/gpl-faq.html#NFUseGPLPlugins>.

och därför inte falla under kravet att behöva licensieras under GPL. Detta undantag kan komma att ha betydelse för hur vissa tekniska lösningar ska bedömas, se nedan 5.4.3.

GPL klargör även i artikel 2 andra stycket andra meningen att copylefteffekten inte inträder om programmet inte anses härröra från det GPL-skyddade programmet:

Om identifierbara delar av verket inte härrör från Programvaran och skäligen kan anses vara fristående och självständiga verk i sig, då skall dessa licensvillkor inte gälla i de delarna när de distribueras som egna verk.⁸⁴

Här knyter man alltså an till det som kan anses gälla enligt svensk upphovsrätt, se 5.1 ovan.

För att kunna ge mer pragmatiska svar på frågan om copylefteffektens inträde krävs emellertid en teknisk granskning av vilka metoder som använts. Några metoder kan nämligen anses vara av sådan natur att de är mer närliggande att konstituera bearbetningar än. Vi kommer i det följande att granska några av dessa tekniska lösningar och se hur de samspelar med juridiken.

5.4 De olika tekniska gränsdragningarna

I det följande ska gås på djupet med de mer uppmärksammade tekniska gränsdragningarna som rör frågan om när en bearbetning anses uppstå. Syftet är att redogöra för författarens uppfattning om hur olika tekniska metoder ska betraktas i ljuset av svensk upphovsrätt. Framställningen kommer att bygga på det under kapitel 3 redovisade om arbetsmetodik och programmeringsteknik. De programmeringstekniska metoderna har redogjorts för under 3.5.1 till 3.5.4 och kommer att utvecklas i den mån det är relevant. Något som är viktigt att hålla i åtanke är att följande analyser och ställningstaganden är generella och att *in casu*-bedömningar är av största vikt på detta område.

5.4.1 Inkorporering av främmande källkod samt makron

Vi börjar med den enklaste situationen, nämligen den då en programmerare kommer över GPL-skyddad källkod som han integrerar i sin egen källkod. En sedvanlig intrångsbedömning ska därvid göras, där huvudfrågan blir om den inlånade källkoden har ett skydd och i sådant fall hur brett det skyddet är.

En möjlig invändning mot ett påstått intrång vore att hävda att man endast använt sig av citaträtten i 22 § URL. Åsikten att det ska tillåtas inom programutveckling har stöd bland annat i den amerikanska *fair use-doktrinen*⁸⁵ och har av vissa ansetts vara i linje med syftet

⁸⁴ Pawlo, M., Inofficiell svensk översättning av GPLv2, idag tillgänglig på <http://www.danielnylander.se/gpl/>.

⁸⁵ Se <http://www.copyright.gov/fls/fl102.html>.

bakom F/OSS-licensieringen.⁸⁶ Vissa har dragit analogier till den klassiska upphovsrätten och påstått att så länge inte mer än 5-6% av det citerade programmets källkod använts så ska det inte anses utgöra ett intrång.⁸⁷ Andra har ansett att citering inom programmering aldrig kan anses följa god sed.⁸⁸

Författarens åsikt är att svensk rätt innebär att citaträtten inom programmeringen är mycket begränsad. En tänkbar acceptabel citering skulle vara då man i sin källkod citerar en annan lösning på ett problem som man själv löst, i syfte att belysa skillnader eller att den andra lösningen vore möjlig. Citeringen bör då enligt författaren ske som en kommentar i källkoden och man bör inte låta det citerade få existera som en fungerande algoritm i programmet. Givet att den inlånade koden omfattas av det upphovsrättsliga skyddet och inte kan anses vara ett citat så utgör det ett intrång i den ursprunglige upphovsmannens immaterialrätt. Med beaktande av de ovan angivna bedömningsgrunderna vid intrång medför inlåning av främmande GPL-skyddad källkod högst troligen att verket i GPL:s mening ”innehåller” programmet eller delar av det, varför copylefteffekten inträder.

En annan tänkbar invändning vore att hänvisa till 26 g § 4 st URL och hävda att man endast tittat på källkoden till det skyddade programmet och hämtat inspiration därifrån. Källkoden är fritt tillgänglig och får läsas utan att bundenhet till licensen uppstår. Det blir emellertid en intrångsbedömning som avgör sannolikheten även i en sådan invändning. Om man *kopierat* den GPL-skyddade koden så råder knappast något tvivel om att en intrångssituation föreligger (givet att den kopierade delen anses skyddad, se 5.1 ovan). Har man däremot skapat en identisk algoritm men exempelvis bytt ut variabelnamnen så är det betydligt mer osäkert om ett intrång föreligger då ju algoritmer i sig inte kan skyddas.

5.4.2 Statisk länkning

Biblioteken inklusive dess headerfil (se 3.3 ovan) kan presumeras uppnå originalitetskravet varför de är upphovsrättsligt skyddade. Den första kopian⁸⁹ av filen uppstår därför hos den som kompilerar programmet och kopian återfinns i det sammanställda programmet. Vid spridning av detta program kommer således upphovsrättsligt skyddad kod att finnas i programmet, på ett liknande sätt som vid integrerandet av främmande kod under 5.3.1 ovan. I och med att det GPL-licensierade programmet integreras i sin helhet i det nya programmet råder det enligt författarens mening ingen tvekan om att det faller inom GPL:s

⁸⁶ Paldam Folker, E., *Open source-licenser i ophavsretlig belysning*, NIR 2/2005, s. 174.

⁸⁷ Paldam Folker, E., a.a.s. 175.

⁸⁸ Olofsson, J., a.a. s. 23.

⁸⁹ Vilket troligen är den andra kopian med tanke på att programmeraren i fråga har tillgång till filen/källkoden för att över huvud taget kunna kompilera den.

tillämpningsområde (se artikel 0 och artikel 2). FSF:s och de flesta andra aktörers åsikt är att statisk länkning innebär att copylefteffekten inträder. Ett undantag är den amerikanska juristen Andrew Katz som inte anser att något annat än modifikationer av den GPL-skyddade programvaran (jämför med LGPL och dess svaga copyleft) medför copylefteffekt.⁹⁰ Argumentet som åsikten grundas på är att det skapade verket, som alltså innehåller både GPL-skyddad kod och proprietär kod, ska betraktas som ett samlingsverk och inte som en bearbetning av GPL-koden. Konsekvensen av detta är att GPL måste följas i den del som rör det statiskt länkade (och GPL-skyddade) biblioteket. Däremot så behöver inte det proprietära programmet följa GPL och därmed heller inte licensieras under GPL. Skulle man godkänna den typen av resonemang så skulle copylefteffekten begränsas till att endast omfatta den GPL-skyddade koden. Detta skulle vara i strid med så väl licensens avsikt som dess lydelse (se artikel 0 och artikel 5).

Vidare så kan argumentet avfärdas på rent programmeringsteknisk nivå. I och med att den GPL-skyddade koden sammanblandas med annan kod vid kompileringstidpunkten så kan den betraktas som modifierad och därmed träffas av copylefteffekten. I vart fall så finns (delar av) den GPL-skyddade koden i det nya programmet och utgör en del av det, varför copylefteffekten inträder. Därtill kan filernas inbördes närhet tas upp som ett argument. I och med att de läses in i datorns arbetsminne samtidigt och dessutom delar utrymme så kan de anses utgöra en helhet som ett program. Att påstå att det är ett samlingsverk lär generellt inte hålla i svensk rätt. Detta med hänvisning till att ett samlingsverks beståndsdelar måste förhålla sig passiva till varandra (se 5.2 ovan). Precis som i 5.4.1 ovan krävs dock att den använda koden faktiskt erhållit ett upphovsrättsligt skydd. Om de påstått intrångsgörande delarna bedöms som skapade i fri anslutning till originalprogrammet så skulle något intrång inte föreligga.

Författarens svar på frågan om statisk länkning medför att copylefteffekten inträder kan sammanfattningsvis besvaras jakande.

5.4.3 Dynamisk länkning

Det största praktiska användningsområdet är även här bibliotek. Från teknisk synvinkel så är det svårt att motivera hur copylefteffekten generellt skulle kunna anses inträda vid dynamisk länkning. Givet att en programmerare inte väljer att modifiera ett GPL-licensierat bibliotek⁹¹ så är frågan huruvida copylefteffekten inträder betydligt svårare att avgöra än vid statisk

⁹⁰ Katz, A., *GPL – the linking debate*, Magazine of the society for computers and law vol 18 issue 3 2007, s. 13 ff.

⁹¹ Vilket som ovan visat ovillkorligen skulle få copylefteffekten att inträda, i vart fall vad gäller det GPL-skyddade programmet.

länkning. Därtill har FSF själva sagt att viss dynamisk länkning aldrig kan medföra copylefteffekt (se 5.3 ovan). Åsikten att dynamisk länkning generellt inte medför copylefteffekten har visst stöd i doktrinen.⁹² En viktig fråga för den juridiska bedömningen är därför hur de dynamiska filerna används samt hur de distribueras.

Ett tänkbart argument vore att den offentliggjorda källkoden varit en förutsättning för att en programmerare skulle kunna skapa den dynamiska länkningen till programmet. Motargumentet skulle troligen vara att det inte alls var en förutsättning, utan att interaktionen gick att uppnå genom att endast utnyttja 26 g § 4 st URL, se argumentationen under 5.1 ovan.

Om man skapar ett proprietärt program som vid installationen även installerar det GPL-skyddade biblioteket så borde detta kunna betraktas som *ett* program och därmed träffas av copylefteffekten. Argumentet för detta är att användaren inte ges möjlighet att välja bort det GPL-skyddade biblioteket som distribueras tillsammans med det proprietära programmet. Det sker därmed en distribution av den GPL-skyddade programvaran. Dessutom har programmeraren själv valt att skapa detta beroendeförhållande. Detta innebär i sin tur att det kombinerade programmet kan betraktas som ett program bestående av det proprietära och det GPL-skyddade programmet. Denna omständighet i sig talar för att copylefteffekten inträder i detta fall.

En annan tänkbar situation är att ett program görs beroende av ett GPL-skyddat bibliotek men att detta inte installeras tillsammans med programmet. Istället kan användaren bli hänvisad till en hemsida där biblioteket finns att ladda ned och installera detta. Detta skulle förskjuta ansvaret för installationen till användaren och denne skulle köra den GPL-skyddade applikationen i enlighet med licensen. Det GPL-skyddade programmet utgör dock en förutsättning för att det proprietära programmet ska fungera som avsett. Beroendeförhållandet mellan programmet och biblioteket skulle därför kvarstå från föregående exempel. Författarens åsikt är därför att även denna situation träffas av copylefteffekten.

En tredje tänkbar situation är att det finns olika bibliotek tillgängliga under olika licenser som alla kan användas för att lösa samma sak i ett program. Detta skulle kunna innebära att användaren vid installationen av ett program får flera valmöjligheter till vilket bibliotek han vill installera och uppmärksammas på vilka licenser som används. Denna distributionsform medför större svårigheter att ge ett generellt svar kring copylefteffekten då både valet av bibliotek och installationen av dem läggs på användaren. Beroendeförhållandet från de tidigare exemplen saknas här, vilket får författaren att luta åt att copylefteffekten generellt sett inte kan sägas inträda i dessa fall.

⁹² Paldam Folker, E., a.a. s. 177.

Ett fjärde tänkbart exempel är en variation på det ovanstående, nämligen att en programmerare skapar ett program som kan använda flera olika bibliotek (under olika licenser) men inte skickar med något i installationen. På detta sätt lämnas det helt åt användaren att välja vad denne vill använda. Det är nu svårt att påstå att det existerar något som helst beroendeförhållande som kunde anmärkas på i de första två exemplen. I detta fall talar starkt för att copylefteffekten inte kan anses inträda.

Som tumregel bör en analogi kunna användas. En professor skriver en studiebok om hur Ekelöfs Rättegång I-V ska förstås. Boken innehåller referenser till Ekelöfs böcker. Boken saknar värde om den inte läses ihop med Ekelöfs böcker, men är knappast att betrakta som en bearbetning av dem.

Sammanfattningsvis är författarens uppfattning att dynamisk länkning befinner sig i en gråzon för bedömningen enligt så väl GPL som URL och att varje fall måste bedömas för sig.

5.4.4 Remote procedure calls

RPC är en teknik som används, som namnet antyder, vid kommunikation mellan datorer. Tekniken använder sig av en struktur där en dator, servern, erbjuder tjänster till klienter. Ett exempel på ett användningsområde är *Software as a Service*-lösningar⁹³ som går ut på att man istället för att köra ett program på sin egen dator eller server låter ett företag göra det på sina servrar. Man ansluter sedan över Internet till dessa servrar för att kunna använda applikationen. Anknytningen till GPL och copylefteffekten ska förklaras med ett exempel.

Om ett företag skulle skapa ett program, programmet X1, där man integrerat GPL-skyddad källkod så skulle det högst troligen (se 5.4.1 ovan) träffas av copylefteffekten och omöjliggöra försäljning av programmet om det inte gjordes tillgängligt under GPL. Detta bygger på distributionsmodellen att den som köper programmet också får det i digital form som installeras på en eller flera datorer, i och med att det då skulle röra sig om distribution (och exemplarframställning) av den skyddade koden.

Betänk istället situationen att företaget som utvecklar X1 gör om det till en serverprogramvara, programmet X2, som man hostar på en server. Man skapar även en klientapplikation, programmet Y, som inte innehåller någon GPL-skyddad kod och vars största uppgift består i att ansluta till servern. När Y väl ansluter till servern så får användaren tillgång till programmet X2 och dess funktioner och därmed till alla funktioner som X1 kunde erbjuda. Har man på detta sätt lyckats kringgå GPL?

Abstrakt så handlar det alltså om två program som utbyter data sinsemellan över ett nätverk. Det man åstadkommit är att den GPL-skyddade koden endast används i ett program,

⁹³ http://en.wikipedia.org/wiki/Software_as_a_service.

X2, som inte distribueras. Någon källkod sprids inte till Y utan finns bara i X2 och företaget kan därför påstå sig utnyttja rätten att köra och modifiera programvaran i enlighet med GPL artikel 0 i och med att någon distribution inte sker. Det kan därför argumenteras för att någon copylefteffekt inte uppstår och att användandet sker i enlighet med GPL.

FSF har själva uttryckt att RPC-tjänster är en gråzon⁹⁴ och att enkla interaktioner mellan program måste vara tillåtna utan att copylefteffekten inträder.⁹⁵ När FSF insåg att det gick att kringgå GPL genom nätverkskommunikation så skapade man licensen AGPL.⁹⁶ I dess artikel 13 anges att modifikationer av GPL-skyddad programvara som användare kan komma åt över nätverk (RPC-situationen) måste göras tillgängliga under (A)GPL.

En möjlig invändning ur upphovsrättslig synpunkt vore att resultatet av en körning av ett datorprogram skulle kunna erhålla skydd. Ett sådant resultat av ett programs output ska bedömas enligt allmänna upphovsrättsliga regler.⁹⁷ Detta innebär att situationer där ett GPL-skyddat program görs om till en serverprogramvara men där användargränssnittet lämnas orört kan anses utgöra intrång.

Sammanfattningsvis kan konstateras att man i RPC-kommunikation kan ha en server som erbjuder klienter tillgång till GPL-skyddad programvara utan att den skyddade programvaran framställs på klientdatorerna. Den GPL-skyddade koden är inte en del av klientapplikationen, utan förekommer bara på servern. Oavsett om källkoden modifieras eller inte så sker ingen distribution eftersom klientdatorerna inte har möjlighet att kopiera källkoden eller programmet. Då ingen exemplarframställning av källkoden eller programmet sker så får användandet anses röra sig inom det av licensen tillåtna. Författarens slutsats av detta är att RPC-kommunikation som huvudregel inte medför att copylefteffekten inträder, med reservation för att det skyddade programmets output kan vara skyddat och därmed förbjudet att göra tillgängligt.

⁹⁴ <http://www.gnu.org/licenses/gpl-faq.html#MereAggregation>.

⁹⁵ FSF har även tillsatt en arbetsgrupp för att utreda bland annat frågan om RPC-interaktion kan medföra copylefteffekt. Se <https://wiki.fsfe.org/EuropeanLegalNetwork/LinkingDocument>.

⁹⁶ GNU Affero General Public License, se <http://www.gnu.org/licenses/agpl.html>.

⁹⁷ Lindberg/Westman s. 234.

6 Avslutning

Tidigare i uppsatsen drogs en parallell mellan copylefteffekten och arvsanlag. För att återvända till den så är copylefteffekten inte att betrakta som en smitta, utan som ett arv. Arvet inträder endast om det GPL-skyddade programmet modifieras eller återfinns till en så pass stor eller viktig del i det nya programmet att det inte kan anses utgöra ett fritt och självständigt verk. Det man vill skydda sig mot är att proprietära program använder sig av de GPL-skyddade programmen på ett sätt som kan liknas vid ett beroendeförhållande.

För en programmerare handlar det om att effektivisera utvecklingstiden. Att involvera GPL-skyddad kod innebär därför ett medvetet risktagande, där risken är att man ställs inför de tre alternativen i 2.4 ovan. Konsekvensen kan bli mycket stor för ett företag som investerat pengar i utvecklingen av en programvara men där hanteringen av extern kod och deras licenser varit bristfällig. Det finns med andra ord all anledning till att ta fram principer för om och hur extern källkod får användas, vilka licenser som är inblandade och hur det i sådant fall ska dokumenteras.

GPL är en giltig licens i Sverige. Enkelt sammanfattat så bör FSF:s tämligen extensiva bild av när ett verk används eller bearbetas tas med en stor nypa salt. Detta då licensen ska tolkas enligt svensk rätt som inte kan anses i fullständig överensstämmelse FSF:s uppfattning. Ett exempel på detta är att förekomsten av några delar av ett GPL-skyddat program i ett proprietärt program inte kategoriskt kan anses medföra att det senare utgör en bearbetning av det förra. En inträngsbedömning måste göras och det kan mycket väl vara så att programmen i fråga har snarlika funktioner men att dessa givits olika uttryck. Ett annat alternativ vore att den kopierade koden inte är upphovsrättsligt skyddad, exempelvis när det bara finns en algoritm som kan lösa ett visst problem.

Copylefteffekten i sig möter inget hinder men frågan är om den kan anses inträda på det sätt som FSF påstått. Svaret är nej, då FSF:s påståenden gjorts tämligen nyanslöst. Istället är svaret att det beror på vilken typ av teknisk lösning som använts och hur den använts i det specifika fallet. Nedan redovisas slutsatserna i tabellform.

Att några definitiva svar inte kan lämnas kan förklaras av tre saker; licensens oklara formuleringar, bristen på relevanta motivuttalanden och bristen på praxis. Det första problemet behöver knappast förklaras ytterligare. Propositionen till URL kom 1960 och datorprogram var inte något av relevans vid denna tidpunkt. I och med att datorprogram togs in som ett skyddsobjekt i URL utan några större särregleringar ansågs analogier kunna dras mellan datorprogram och den analoga upphovsrätten. Vidare lämnades det till praxis att vidare definiera termen "bearbetning" i datorprogramssammanhang. Eftersom man från domstolarnas håll inte givit sig i kast med några betydelsefulla tekniska distinktioner av

begreppet sedan dess så blir författarens uttalanden av generell natur och beroende av bedömningar i de enskilda fallen. Författarens förhoppning är emellertid att någon typ av vägledning ska gå att finna i de redogörelser som gjorts i de olika programmeringstekniska situationerna.

Teknisk lösning	Inträder copylefteffekten enligt FSF?	Inträder copylefteffekten enligt författaren?
Copy-paste/makron	Ja	Huvudregel: ja, men beror på omfattningen
Statisk länkning	Ja	Huvudregel: ja, men beror på omfattningen
Dynamisk länkning	Oftast	Huvudregel: nej, men undantag finns
RPC	Det beror på	Huvudregel: nej, men undantag finns

Källförteckning

Litteratur

- Andersson, Mattias, *Open source – Ur ett praktiskt juridiskt perspektiv*, Det IT-rättsliga observatoriets rapport 54/2002.
- Feller, Joseph, Fitzgerald, Brian, Hissam, Scott, Lakhani, Karim, *Perspectives on Free and Open Source Software*, The MIT Press, 2007.
- Hellner, Jan, Hager, Richard, Persson, Annina, *Speciell avtalsrätt II, Kontraktsrätt*, 1 häftet, 4 u, Norstedts Juridik AB, 2005.
- Katz, Andrew, *GPL – the linking debate*, Magazine of the Society for Computers and Law, vol 18 issue 3, 2007.
- Koktvedgaard, Mogens, Levin, Marianne, *Lärobok i Immaterialrätt*, 8 u, Norstedts Juridik AB, 2004.
- Lindberg, Agne, Westman, Daniel, *Praktisk IT-rätt*, 3 u, Advokatfirman Delphi & Co och Norstedts Juridik AB, 2001.
- Olofsson, Jessica, *Upphovsrättsliga aspekter på licenser för fri programvara och öppen källkod*, IRI rapport 2003:1, Institutet för rättsinformatik, juridiska fakulteten vid Stockholms universitet, 2003.
- Olsson, Henry, *Upphovsrättslagstiftningen*, tillgänglig på Zeteo, version av den 1 maj 2011, Norstedts Juridik AB, 2011.
- Paldam Folker, Emil, *Open source-licenser i ophavsretlig belysning*, NIR 2/2005.
- Pawlo, Mikael, *Något om fri programvara och öppen källkod – nya licenstyper för datorprogram*, NIR 4/2002.
- Ramberg, Christina, Ramberg, Jan, *Allmän avtalsrätt*, 7 u, Norstedts Juridik AB, 2007.
- Välimäki, Mikko, *GNU General Public License and the Distribution of Derivative Works*, Journal of Information, Law & Technology (JILT), 2005.
- Westman, Daniel, *Återanvändning av programkod som utvecklas av eller för den offentliga sektorn*, IRI promemoria 2/2007, Institutet för rättsinformatik, juridiska fakulteten vid Stockholms universitet, 2007.

Offentligt tryck

- Lag (1915:218) om avtal och andra rättshandlingar på förmögenhetsrättens område.
- Lag (1960:729) om upphovsrätt till litterära och konstnärliga verk.
- Lag (1994:1512) om avtalsvillkor i konsumentförhållanden.

Offentliga utredningar

SOU 1985:51 Upphovsrätt och dator teknik. Delbetänkande 3 av upphovsrättsutredningen.

Propositioner

Proposition 1984/85:110 om konsumenttjänstlag.

Proposition 1988/89:85 om upphovsrätt och datorer.

Proposition 1989/90:89 om ny konsumentköplag.

Proposition 1994/95:17 Oskäligen avtalsvillkor m.m. Införlivande med svensk rätt av EG:s direktiv om oskäligen avtalsvillkor i konsumentförhållanden.

EU-rättsligt material

Förordningar

Europaparlamentets och rådets förordning (EG) nr 864/2007 av den 11 juli 2007 om tillämplig lag för utomobligatoriska förpliktelser.

Europaparlamentets och rådets förordning (EG) nr 593/2008 av den 17 juni 2008 om tillämplig lag för avtalsförpliktelser.

Direktiv

Europaparlamentet och rådets direktiv 2009/24/EC av den 23 april 2009 om rättsligt skydd för datorprogram.

Svenska rättsfall

NJA 1997 s. 109.

Utländska rättsfall

Landgericht München I, Entscheidung vom 19. Mai 2004, 21 O 6123/04, ”*netfilter/iptables*”.

Landgericht Frankfurt am Main vom 6. September 2006, 2-6 O 224/06, ”*d-link*”.

Webbsidor

Black Duck Software, *Top 20 Most Commonly Used Licenses in Open Source Projects*, <http://www.blackducksoftware.com/oss/licenses#top20> (2011-07-28).

FLOSSMETRICS/OpenTTT, *6. FLOSS-based business models*,

http://guide.conecta.it/index.php/6._FLOSS-based_business_models (2011-09-21).

GNU Operating System – Home of the Free Software Foundation

The AGPL license, <http://www.gnu.org/licenses/agpl.html> (2011-10-30)

The LGPL license, <http://www.gnu.org/licenses/lgpl.html> (2011-08-03).

GPL FAQ: What does it mean to say a license is “compatible with the GPL”,
<http://www.gnu.org/licenses/old-licenses/gpl-2.0-faq.html#WhatDoesCompatMean>
(2011-08-04).

GPL FAQ: Can I release a non-free program that's designed to load a GPL-covered plug-in?, <http://www.gnu.org/licenses/gpl-faq.html#NFUseGPLPlugins>, (2011-08-14).

GPL FAQ: What is the difference between an “aggregate” and other kinds of “modified versions”?, <http://www.gnu.org/licenses/gpl-faq.html#MereAggregation> (2011-08-11).

The Free Software Definition, <http://www.gnu.org/philosophy/free-sw.html> (2011-07-29).

GNU General Public License, <http://www.gnu.org/copyleft/gpl.html> (2011-07-29).

Various Licenses and Comments about Them, <http://www.gnu.org/licenses/license-list.html#GPLv2> (2011-08-22).

Pawlo, Mikael, *Inofficiell svensk översättning av GPLv2*, <http://www.danielnylander.se/gpl>
(2011-09-03).

Swedsoft, *Vad är mjukvara?*, <http://www.swedsoft.se> (2011-09-09).

The Open Source Initiative, *The Open Source Definition*, <http://opensource.org/docs/osd>
(2011-07-29).

U.S. Copyright Office, *Fair use*, <http://www.copyright.gov/fls/fl102.html> (2011-08-05).

Wikipedia

Tivoization, <http://en.wikipedia.org/wiki/Tivoization>, (2011-07-30).

Application Programming Interface (API),
http://sv.wikipedia.org/wiki/Application_Programming_Interface (2011-08-17).

Bubble sort, http://en.wikipedia.org/wiki/Bubble_sort (2011-08-10).

Copy and paste, http://en.wikipedia.org/wiki/Cut,_copy,_and_paste 2011-08-13).

Fast inverse square root,
https://secure.wikimedia.org/wikipedia/en/wiki/Fast_inverse_square_root (2011-08-09).

Inter-process communication, http://en.wikipedia.org/wiki/Inter-process_communication
(2011-08-14).

Interface description language,
http://en.wikipedia.org/wiki/Interface_description_language (2011-09-06).

Public domain, http://en.wikipedia.org/wiki/Public_domain (2011-08-04).

Skärmdump, <http://sv.wikipedia.org/wiki/Sk%C3%A4rmdump> (2011-08-04).

Software as a Service, http://en.wikipedia.org/wiki/Software_as_a_service (2011-10-14).

Appendix – GNU GPL

Version 2, June 1991

Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a)** You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- b)** You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- c)** If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

- a)** Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- b)** Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
- c)** Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE

LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and an idea of what it does.
 Copyright (C) yyyy name of author

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) *year name of author*
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type `show w'. This is free software, and you are welcome
to redistribute it under certain conditions; type `show c'
for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items-- whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright  
interest in the program `Gnomovision'  
(which makes passes at compilers) written  
by James Hacker.
```

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License.