Fd.o: Üljünk át egy szebb asztalhoz

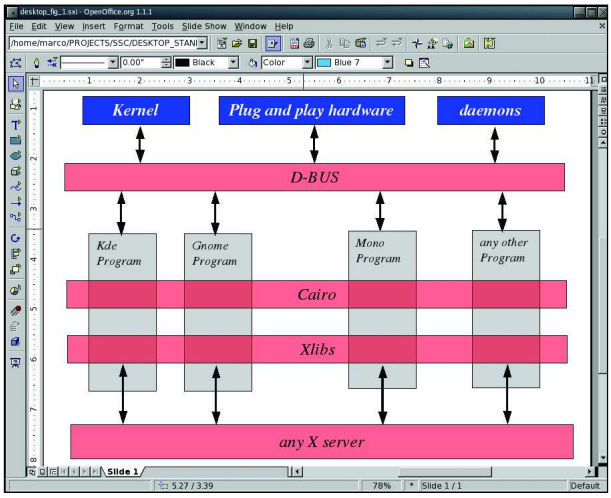
Senki ne gondolja, hogy az asztali környezetek között háború folyik. Az alkalmazások és az asztali környezetek a háttérfalak mögött egymással együttműködnek, és a való életben is helytállt szabványokat alkalmazva segédkeznek abban, hogy mindenki alkalmazásai gond nélkül működjenek, sőt, együttműködjenek.

A végfelhasználókat csak a kívánt feladatokat ellátó alkalmazások érdeklik. Azért választják a ***Linuxot***, hogy ezeket az alkalmazásokat szabadon, egyenként választhassák ki. Számukra az integrált asztali környezet tetszőleges programelegy kiválasztásának szabadságát jelenti, és a garanciát arra, hogy ennek összetevői működni fognak egymással. Egy monolitikus asztali környezet a programozókat is korlátozná. Saját kódunk más alkalmazásokkal való együttműködésének biztosítása alapvető követelmény, ha nem az első számú jellemző, mely a program hasznosságát megszabja. Ha célunk eléréséhez csak egy vagy két fejlesztői eszközláncot vehetünk igénybe, annak nem sok értelme van. Korábban az ***XFree86 GNU/Linux*** asztali környezet fejlesztése túlságosan lassan haladt, és teljesítménye sem volt kielégítő. Sok eszközt, a ***fontconfig***-tól egészen a ***zlibig*** a külső függőségek elkerülése miatt meg kellett kettőzni. Ha egy illesztőprogram megváltozott, az egész csomagot újra ki kellett adni. Mindezt tetézte, hogy az ***XFree86*** felhasználói szerződése a múlt évben úgy változott meg, hogy a ***GPL*** programok számára megtiltotta az új kódok becsatolását. A szerződés miatt számos terjesztés összeállítói egyszerűen az új változat elhagyása mellett döntöttek.

A ***Freedesktop.org (FD.o)*** 2000 márciusában alakult, elsődleges célja a fejlesztők segítése volt a fenti műszaki kérdések megoldásában. Alapítói olyan alaprendszert akarnak létrehozni, amelyre bármilyen asztali környezet felépíthető. A megoldást független specifikációk kidolgozásában látják, amelyeket szükség szerint működő kódok egészítenek ki. A formális szabványosítást más testületekre kívánják hagyni. Az előírásokat követve valódi együttműködést lehet megvalósítani az alkalmazások között, méghozzá fejlesztésük során a lehető leghamarabb – ideális esetben még annak megkezdése előtt. Minden program ***LGPL*** vagy ***X***-jellegű szerződés hatálya alá kerül. A ***FD.o*** számos rokonszenves tervezetet támogat, ám írásomban csak a fő eszközöket szeretném bemutatni, ezek alkotják az úgynevezett ***FD.o*** alaprendszert.

Xlibek

Az ***X Window System*** egy átlátszó hálózati protokoll grafikus megjelenítéshez. A grafikus felületű alkalmazások az ***X***-et használják arra, hogy az ***X***-kiszolgálónak nevezett,



1. ábra Integráció a Freedesktop.org elképzelései szerint: kiszolgálók, bármely alkalmazás által használható könyvtárak és kommunikációs protokollok; mindez független attól, hogy az alkalmazások milyen asztali környezetben készültek.

a képernyőt ténylegesen kezelő programnak rajzolási utasításokat adjanak. Egészen a múlt évig a kiszolgálók és a könyvtárak jellemzően egyetlen monolitikus csomagba kerültek. ***FD.o*** részekre osztotta a csomagot, ezek a részek már önállóan is fejleszthetők és csomagba illeszthetők. Mindennek a fő előnye az, hogy a linuxos fejlesztők tetszésük szerint keverhetik össze és szabhatják testre az egyes részek megvalósításait.

További ***X***-es fejlesztés a fán belüli függőségek eltávolítása, az ***autotools*** használata fordítórendszerként, valamint az ***iconv*** könyvtár alkalmazása az ***Unicode*** és egyéb kódolások közötti átalakításokra. Az ***X*** protokollt burkoló könyvtárakat nevezzük ***Xlib***-eknek. A ***FD.o*** ezek első változatát 2004 januárjában adta ki. Követik az ***X*** szabvány előírásait, így bármely ***X***-kiszolgálóval képesek együttműködni.

A kiterjedt optimalizálások ellenére az ***Xlib***-ek mérete a kisebb teljesítményű gépeken továbbra is gondokat okozhat. További probléma, hogy bizonyos ***Xlib***-kérések blokkolódnak, amíg választ nem kapnak, függetlenül attól, hogy sok

esetben ez elkerülhető volna. Ez viszont ütközik a 2.6-os rendszermagok lappangási idők csökkentésére irányuló fejlesztéseivel. Az ***Xlib***-ek gyorsítótárazás, rétegezés és hasonló megoldások révén sokat tesznek a protokoll elrejtése érdekében is. Ezek a törekvések bizonyos esetekben előnyt, más esetekben csak többletterhelést jelentenek. Végül, de nem utolsóként meg kell említeni, hogy az X-bővítmények készítésének támogatása is korlátozott.

Az ***FD.o*** javaslata mindezen problémák megoldására az ***X C Binding***, röviden ***XCB***. Ez egy második könyvtár, és alkalmas arra, hogy új eszközkészletek és az ***Xlib API*** egyes részeinek egyszerűbb emulációja alapjául szolgáljon. Az ***XCB***-t úgy tervezték, hogy zökkenőmentes együttműködésre legyen képes a ***POSIX*** szálakra vagy egyetlen szálra épülő programokkal. A kód fenntartja a bináris kompatibilitást az ***Xlib*** bővítményekkel és alkalmazásokkal, vagyis használatához nincs szükség a bővítmények újrafordítására. Ezzel a megoldással a fokozatos ***Xlib - XCB*** áttérés könnyebben, a szolgáltatások egy részéről való lemondás nélkül is megvalósítható. A következő lépés ezen az úton az ***Xlibs Compatibility Layer (XCL)***, amelynek segítségével a meglévő, ***Xlibre*** alapuló alkalmazások is kihasználhatják az ***XCB*** előnyeit.

X-kiszolgálók

A ***FD.o*** az ***XFree86*** helyett két alternatívát támogat. Az első az ***XFree86 4.4-RC2*** kód egy mellékágaként indult, még a felhasználási szerződés megváltoztatása előtt. Ez a kiszolgáló az ***X.org***, és az ***XFree86***-tal azonos módon használható. A másik az ***Xserver***, hosszú távon ez tűnik a legígéretesebb választásnak. A ***Kdrive*** egy változata, amely több évvel ezelőtt szintén mint az ***XFree86*** egyszerűbb, erősen módosított kiadása indult. A ***Kdrive*** kisméretű, részben azért, mert kevés kódkettősséget mutat a rendszermaggal. A méretcsökkentést szolgálta néhány elavult szolgáltatás és illesztőprogram-modul eltávolítása is. Jóval kisebb méretének köszönhetően a ***Kdrive*** megfelelőbb alapot nyújt egy teljesen új kiszolgáló felépítéséhez.

Az ***Xserver*** ma elérhető változatát elsősorban tesztcélokra, az új bővítmények és szolgáltatások – mint például az átlátszóság vagy az ***OpenGL*** gyorsítás – kipróbálására használják. A memóriahasználatot úgy csökkentik, hogy sok számítást a program futási időben végez el ahelyett, hogy mindig a memóriában tartaná az eredményeket.

Az ***Xserver*** célja a sebesség növelése, és az egyéb olyan jelenségek (például villódzás) megszüntetése, amelyek miatt egyszerűen rossz ránézni a képernyőre. Egy új ***X***-bővítmény, a ***Composite*** lehetővé teszi a teljes képernyő kettős pufferelését. Természetesen egyetlen kiszolgáló sem lehet okosabb, mint a legbutább terminálja, ám az egyszerűbb felépítésnek köszönhetően könnyebb megtalálni és kijavítani a lassú kódrészeket, bárhol is legyenek. Az új kiszolgáló az eszközkészletek szintjén semmilyen hatással nem bír, hacsak a programozó nem akarja kifejezetten kiaknázni az új bővítmények által kínált lehetőségeket.

Cairo

A vektorgrafikánál a képek több-kevesebb bonyolult vonal, valamint az így kapott területek színekkel való kitöltése révén állnak elő. Az ilyen fájlok kisméretűek, és veszteség nélkül is tetszőleges felbontásra méretezhetők. Ez a megoldás tehát minden olyan felhasználó érdeklődésére számot tarthat, aki biztos akar lenni abban, hogy amit ki fog nyomtatni, az pontosan megegyezik azzal, amit lát. Sajnos az ***X*** bár képes kezelni a szövegek, négyszögek és egyebek képernyőn megjelenő bittérképeit, a nyomtatás és a vektorgrafika egyszerűen kimarad a látóteréből. Ez az egyik oka annak, hogy még jelenleg sem tökéletesen azonos az, amit a képernyőn látunk, amit a papíron kapunk, illetve amit a fájlokba elmentünk.

Az ***FD.o*** megoldása minderre a ***Cairo***, egy új ***2D*** vektorgrafikai könyvtár, mely többféle készüléken is biztosítja az egységes kimeneteket. Érthetőbben fogalmazva: minden adathordozón ugyanazt a kimenetet kapjuk. A külvilág felé a ***Cairo*** felhasználói szintű ***API***-kat biztosít, hasonlóan a ***PDF 1.4*** képkezelő modellhez.

Különféle háttérrendszerek segítségével a ***Cairo*** különböző kimeneti eszközöket képes támogatni. Az első eszközcsoportot a képernyők alkotják, ezek az ***Xlib*** vagy az ***XCB*** révén érhetők el. Ugyancsak fontos eszközt jelentenek a memóriában tárolt képpufferek, amelyeket fájlba lehet menteni vagy más alkalmazásoknak lehet átadni.

A ***PostScript*** és a ***PNG*** kimenet szintén megoldott, a ***PDF*** pedig tervezés alatt áll. Az ***OpenGL***-gyorsított kimenetet a ***Glitz*** nevű háttérrendszer fogja megvalósítani, illetve a ***Glitz*** önálló, ***OpenGL*** feletti rétegként is használható lesz.

A ***Cairo***-hoz nyelvi kötések ***C++***, ***Java***, ***Python***, ***Ruby*** és ***GTK+*** nyelvekhez léteznek.

Az ***OpenOffice.org*** fejlesztői az ***OOo 2.0***-ás változata után szintén tervezik a ***Cairo*** használatát, akár mint külön letölthető grafikai beépülő modulét. A ***Cairo*** jelenleg is fejlesztés alatt áll, ám mivel még nem rendelkezik tökéletesen üzembiztos ***AIP***-val, a hivatalos ***FD.o*** kiadásoknak egyelőre nem része.

D-BUS

A ***D-BUS*** egy bináris protokoll ***folyamatok közötti adatcsere (Interprocess Communication, IPC)*** céljára; itt a folyamatok azonos asztali munkamenet alatt futó alkalmazásokat jelentenek, illetve magát az operációs rendszert. A második esetbe tartoznak a felhasználóval folytatott párbeszédek is, amikor új hardverrel vagy szoftverrel bővül a számítógép.

A ***D-BUS*** belső világát ***Robert Love*** részletesen ismertette ***„Get on the D-BUS”*** című írásában, a ***Linux Journal 2005. februári számában***. A felhasználót tekintve a ***D-BUS***-nak legalább olyan szolgáltatásszintet kell biztosítania, mint amilyet a ***GNOME*** és a ***KDE*** is nyújt. Ez egy ***message bus (üzenetbusz)*** nevű rendszerdémon és egy felhasználónkénti, munkamenethez kapcsolódó démon futtatásával válik lehetővé. A ***D-BUS***-on keresztül bármely két program kapcsolatba léphet egymással, amivel a lehető legjobb teljesítményt lehet elérni. Hasonlóan a teljesítmény javítását szolgálja az is, hogy mivel a ***D-BUS***-t igénybe vevő programok szinte mindig azonos gépen futnak, ezért egyszerű ***XML*** helyett bináris formátumot használunk.

Az üzenetbusz démon lényegében útválasztóként üzemel. Mivel az alkalmazások között nem bájtfolyamokat, hanem üzeneteket továbbít, szolgáltatásai a munkaasztalról is elérhetők. Normál esetben a démon több, egymástól független példányban fut. Az egyik példány rendszerszintű kommunikációra szolgál, esetében szigorú biztonsági előírások szabják meg a fogadható üzenetek körét. A többi példány

az egyes felhasználói munkamenetekhez jön létre, és a bennük futó alkalmazásokat szolgálja ki. A ***D-BUS*** rendszerpéldánya biztonsági kockázatot is jelenthet, ugyanis a rootként futó szolgáltatásoknak képeseknek kell lenniük adatok és események cseréjére a felhasználói alkalmazásokkal. Éppen ezért eleve korlátozott jogosultságokkal való használatra tervezték, és ***chroot jailben*** fut. A ***D-BUS***-szal kapcsolatos biztonsági előírások az ***Fd.o*** webhelyén (lásd az internetes forrásokat) találhatók meg.

A legtöbb programozónak nem kell közvetlenül szólnia a ***D-BUS*** protokollhoz, gyakorlatilag bármely keretrendszerhez és nyelvhez léteznek megfelelő burkolókönyvtárak. Ez azt is jelenti, hogy mindenki megtarthatja megszokott környezetét, csak az ***IPC*** miatt nem kell váltani. A végfelhasználók szintén jól járnak, hiszen a ***KDE***, a ***GNOME*** és a ***Mono*** alapú programok az eszközkészlettől függetlenül képesek lesznek kapcsolatba lépni egymással. Ahogy a ***Cairo***, úgy ***D-BUS*** is hiányzik az ***FD.o*** első változataiból, ugyanis ***API***-ja még nem minősül üzembiztosnak. Mindettől függetlenül a fejlesztők a ***D-BUS***-t a jövőbeli kiadások egyik sarokkövének tekintik. A ***D-BUS*** az elképzelések szerint a ***KDE 4***-ben található ***DCOP*** helyét is át fogja venni.

Biztosan ez a jó megoldás?

Csak az idő múlásával adhatunk egyértelmű választ arra, hogy az ***Fd.o*** első megvalósításai elég jók-e, a lefektetett irányelvek pedig megállják-e a helyüket. Utóbbi alatt itt azt kell érteni, hogy olyasvalamit állítottak-e össze a fejlesztők, amely nulláról indulva újra megvalósítható – ha valakinek ehhez támadna kedve. Én biztos vagyok abban, hogy maga a megközelítés helyes, és több lehetőséget tartogat, mint a jelenleg létező „teljes értékű munkakörnyezet” megoldások bármelyike.

A leggyakrabban két aggállyal találkozom: 1) a jelenlegi asztali környezetek elveszítik önazonosságukat, egyetlen feladatuk a felhasználói felület biztosítása lesz 2) az ***FD.o*** nem szabvány, csak egy újabb megvalósítás. Az első felvetésre személy szerint visszakérdeznék: biztos, hogy ez baj? A legtöbb végfelhasználó sosem fogja észrevenni, de ha mégis, hát nem fog foglalkozni vele. Egyszerűen tudomásul veszik a korábban említett fejlesztéseket, és többet nem foglalkoznak a kérdéssel. Ha biztosítjuk az alkalmazások együttműködésének lehetőségét, tekintet nélkül azok fejlesztésének módjára, akkor a ***Linux*** a vállalati körökben is elfogadhatóbbá válik, és a zárt megoldások mellett szóló érvek közül egy újabb veszti érvényét.

Ha a protokollok és a formátumok többé nem meghatározott megvalósításokhoz vagy eszközkészletekhez kötődnek, akkor több asztali környezet között is meg lehet osztani őket. Ezzel a kód üzembiztossága és egyszerűsége terén csak nyerhetünk. A teljesen új programok fennakadások nélkül, azonnal kommunikálni tudnának a meglévőkkel. Bízom abban, hogy ez a szemlélet egyre nagyobb teret nyer, és egyre több alkalmazásfüggetlen szabvány kerül az ***FD.o*** szárnyai alá, ide értve a fájlformátumokat, a hangsémákat, a szín- és feladatbeállításokat egyaránt. Képzeljünk el egy levelezési beállító fájlt, amely tetszőleges levelező ügyfélprogrammal használható, az ***Evolution***-től egészen a ***mutt***-ig; vagy egy olyan könyvjelzőfájlt, amit a ***Mozilla*** és a ***Lynx*** egyaránt képes értelmezni.

Ami a második kifogást illeti – az ***FD.o*** nem szabvány, csak egy újabb megvalósítás –, nos, a szabad szoftverek és az ***RFC***-k világa pontosan így működik. Ha a szabályokat a kóddal együtt írjuk, akkor az elképzeléseket a lehető leghamarabb ki tudjuk próbálni a gyakorlatban is. Csak megemlíteném, hogy hasonló dolog történik az ***OO.o***-val és az ***OASIS*** irodai szabvánnyal. A fájlformátum a ***StarOffice***-on és az ***OO.o***-n belül született meg és finomodott ki, de ma már saját életet él. A bizottságban immár megtaláljuk a ***KOffice*** képviselőit is, és bármely jövőbeli irodai csomag használhatja natív formátumaként, akár nulláról indulva, kizárólag az előírásokat követve.

Persze ennek az útnak is vannak buktatói, de úgy látom, a fejlesztők tisztában vannak ezekkel, és képesek elkerülni őket. Megvan például a kockázata annak, hogy kizárólag ***Linux*** alatt működő szabványokat készítünk, a többi ***UNIX***-változatról elfeledkezünk. Hasonlóan ügyelni kell az erőforrás-használatra is, hiszen hiába a legvarázslatosabb elképzelés, ha kétszer annyi memória kell a zökkenőmentes működéséhez. Szerencsére – legalábbis egyelőre úgy látom – ilyesmitől nem kell tartani. Az viszont biztos, hogy itt az ideje csatlakozni ezekhez az erőfeszítésekhez. Kellemes buherálást!

Köszönetnyilvánítás

Köszönettel tartozom ***Waldo Bastian-nek***, ***Keith Packardnak***, ***Daniel Stone***-nak és ***Sander Vesiknek***, amiért magyarázataikkal segítették munkámat.

Linux Journal 2005. május, 133. szám

Marco Fioretti hardver-rendszermérnök, a szabad szoftverekkel mint EDA alaprendszerekkel és a hatékony asztali környezet létrehozására irányuló RULE Project vezetőjeként áll kapcsolatban. Marco Olaszországban, Rómában él a családjával.

KAPCSOLÓDÓ CÍMEK

XFree86: ➲ www.xfree86.org

X Protocol: ➲ www.x.org/X11_protocol.html

Libiconv: ➲ www.gnu.org/software/libiconv

Xlibs: ➲ www.freedesktop.org/Software/xlibs

Xserver: ➲ freedesktop.org/Software/xserver

Cairo: ➲ www.cairographics.org

Glitz: ➲ www.freedesktop.org/Software/glitz

D-BUS: ➲ www.freedesktop.org/Software/dbus

➲ www.linux.org.uk/~telsa/Trips/Talks/g3-swamp.html