

Szegedi Tudományegyetem
Informatikai Tanszékcsoport

SZAKDOLGOZAT

Monda László

2006

**Szegedi Tudományegyetem
Informatikai Tanszékcsoport**

**Kétpaneles fájlkezelő elkészítése és a
fájlmenedzsment alternatív lehetőségeinek a
kutatása**

Szakdolgozat

Készítette:

Monda László

programozó matematikus
szakos hallgató

Témavezető:

Bilicki Vilmos

egyetemi tanársegéd

Szeged
2006

Feladatkiírás

Egy kétpaneles fájlkezelő elkészítése, amely potenciálisan több platformra átültethető és különösen jól megtervezett felhasználói felülettel rendelkezik.

A projekt infrastruktúrájának a kiépítése az Interneten elérhető szabad szolgáltatások maximális kihasználásával. Projekt honlap elkészítése, ami demonstrálja az alkalmazást és magára irányítja a felhasználók és a fejlesztők figyelmét.

A fájlmenedzsment alternatív lehetőségeinek a kutatása, amelyek meggyorsítják a fájlokkal történő munkát különös tekintettel az archivált fájlok menedzselése és a metaadatok kezelése tekintetében.

Tartalmi összefoglaló

- **A téma megnevezése:** Kétpaneles fájlkezelő elkészítése és a kapcsolódó feladatok elvégzése.
- **A megadott feladat megfogalmazása:** Kétpaneles fájlkezelő elkészítése, a projekt infrastruktúrájának a kiépítése és a fájlmenedzsment alternatív lehetőségeinek a kutatása.
- **A megoldási mód:** Mono környezet használata Gtk# grafikus felhasználói eszközkészlet segítségével Linux platformon, RAD fejlesztési metodológia alkalmazásával.
- **Alkalmazott eszközök, módszerek:** Mono környezet, Gtk# grafikus felhasználói eszközkészlet, Linux platform, RAD fejlesztési metodológia és fejlesztő eszközök alkalmazása. A SourceForge.net fejlesztői portál és sok másik szabadon igénybe vehető kapcsolódó szolgáltatás használata. A MediaWiki tartalomkezelő rendszer használata a projekt honlap részére.
- **Elért eredmények:** A fájlkezelő elkészítése pre-alpha állapotban, a projekt infrastruktúra erős kiépítése, a fájlmenedzsment alternatív területein végzett kutatási tapasztalat és a projekt jövőbeli direktíváinak definiálása.
- **Kulcsszavak:** Mono, Gtk#, fájlkezelő, metaadat, démon

Tartalomjegyzék

<u>Feladatkiírás.....</u>	<u>4</u>
<u>Tartalmi összefoglaló.....</u>	<u>5</u>
<u>Tartalomjegyzék.....</u>	<u>6</u>
<u>BEVEZETÉS.....</u>	<u>8</u>
<u>1. A FÁJLKEZELŐ.....</u>	<u>9</u>
<u>1.1. Hasonló alkalmazások.....</u>	<u>9</u>
<u>1.2 A kiterjeszhetőségről.....</u>	<u>10</u>
<u>1.3. Technológiai áttekintés.....</u>	<u>10</u>
<u>1.4. Felhasználói interfész tervezés.....</u>	<u>11</u>
<u>1.4.1. Egyablakos felület.....</u>	<u>11</u>
<u>1.4.2. Információs sáv.....</u>	<u>12</u>
<u>1.4.3. Keretek, vájatok és nézetek.....</u>	<u>12</u>
<u>1.4.4. Beágyazott panel figyelmeztetések.....</u>	<u>13</u>
<u>1.4.5. Új könyvtár létrehozása.....</u>	<u>14</u>
<u>1.4.6. Átnevezés.....</u>	<u>15</u>
<u>1.5. Interakció tervezés.....</u>	<u>16</u>
<u>1.5.1. Billentyűzettel történő vezérlés.....</u>	<u>16</u>
<u>1.5.2. Lebegő kijelölés.....</u>	<u>17</u>
<u>1.5.3. Élő panel konfiguráció.....</u>	<u>17</u>
<u>1.6. A projekt fogadtatása.....</u>	<u>17</u>
<u>1.7. Tapasztalatok.....</u>	<u>18</u>
<u>1.8. Jövőbeli tervek.....</u>	<u>20</u>
<u>2. A FÁJLMENEDZSMENT ALTERNATÍV LEHETŐSÉGEI.....</u>	<u>21</u>
<u>2.1. Integrált média katalógus.....</u>	<u>21</u>
<u>2.2. Globális metafájl adatbázis.....</u>	<u>22</u>
<u>2.3. Strukturált metaadat asszociáció és lekérdezés.....</u>	<u>23</u>
<u>2.4. Filmek vizuális asszociációja.....</u>	<u>23</u>
<u>3. A DEPOT DÉMON.....</u>	<u>26</u>
<u>3.1. Névterek és szignatúrák.....</u>	<u>27</u>
<u>3.2. A katalógus névtér.....</u>	<u>27</u>
<u>3.3. A globális névtér.....</u>	<u>27</u>

3.4. Strukturált metaadat asszociáció.....	29
3.5. Kapcsolódó projektek.....	30
4. PROJEKT ADMINISZTRÁCIÓ.....	31
4.1. SourceForge.net.....	31
4.1.1. Verziókövetés.....	31
4.1.2. Levelezőlisták.....	32
4.1.3. Követők.....	32
4.1.4. Szoftver térkép.....	32
4.1.5. Webtér.....	33
4.2. freenode.....	33
4.3. CIA.....	33
4.4. Gmane.....	34
4.5. freshmeat.....	34
5. PROJEKT MARKETING.....	35
5.1. Projekt logó.....	35
5.2. Flash videók.....	35
5.3. Barátságos arculat kialakítása.....	36
5.4. Bátorítás a közreműködők felé.....	36
6. HIVATKOZÁSOK.....	38
Nyilatkozat.....	41
Köszönetnyilvánítás.....	42

BEVEZETÉS

Ez a dolgozat három fő részre tagolható. Az első részben egy kétpaneles fájlkezelőről írok, amit elkezdtem a fejleszteni, nagyon jól megtervezett felülettel rendelkezik és potenciálisan több platformra portolható. A második rész a fájlmenedzsment alternatív lehetőségeiről szól, amelyek sokkal hatékonyabbá tehetnék a fájlokkal történő munkánkat különös tekintettel az archivált fájlok menedzselése és a metaadatok kezelése tekintetében. A dolgozatom harmadik része a projektmenedzsmenttel kapcsolatos adminisztratív és marketing megoldásokat mutatja be.

1. A fájlkezelő

Jelen fejezet a fájlkezelő alkalmazásról fog szólni, az azzal kapcsolatos lehetőségekről és a tervezéssel kapcsolatos kihívásokról.

1.1. Hasonló alkalmazások

Számtalan kétpaneles fájlkezelőt írtak már a legkülönbözőbb platformokra az első ilyen fájlkezelő, a Norton Commander [NC] megjelenése óta. Az ilyen típusú fájlkezelőket ortodox fájlkezelőnek [ORTHODOXFM] is hívják, amelyek általában a következő képességekkel rendelkeznek:

- Kétpaneles könyvtár nézet
- Közeli integráció a parancssorral
- A billentyűzetkombinációk erős alkalmazása

Ezeknek a képességeknek köszönhetően az ortodox fájlkezelőkkel sokkal hatékonyabban lehet dolgozni, mint a hagyományos Windows Explorer-szerű fájlkezelőkkel.

Jelenleg a legismertebb és legérettebb ilyen fájlkezelő Windows platformra írt a Total Commander [TC], amelynek a legközelebbi Linuxos alternatívája a Midnight Commander [MC] nevet viseli.

Sajnos a fájlkezelőkkel elég mostohán bántak a fejlesztők a portabilitás terén. Ez talán annak tudható be, hogy a különböző platformok eltérő fájlrendszereket használnak, amelyek különböző kiterjesztésekkel és képességekkel bírnak, így a portolással megnőne az adott szoftver karbantartási költsége és bonyolultsága.

Akárhogy is, a fájlkezelők listáját [FMLIST] átnézve, még nem született olyan ortodox fájlkezelő, amely több platformra portolható, szabad szoftver, jól megtervezett felhasználói felülettel rendelkezik és további komoly képességeket foglal magában, amiket a professzionális felhasználók elvárhatnak.

A projektemet, az Ultimate Commander fájlkezelőt azért indítottam újtára, hogy ezt az űrt betöltse. Az ultimate jelzővel próbáltam utalni a projekt jövőbeli elképzelt kaliberére, a commander jelzővel pedig a fájlkezelő ortodox jellegére.

1.2 A kiterjeszhetőségről

Sajnos nagyon kevés fájlkezelő létezik, amely valamilyen mechanizmus segítségével kiterjeszhető, pedig a jól megtervezett plugin interfészek óriási potenciált foglalnak magukban.

A kiterjeszhetőség erejét nem nehéz általánosságban sem belátni olyan projekteket megvizsgálva, mint például a Mozilla Firefox [FIREFOX] böngésző, vagy a GIMP képmánipuláló program [GIMP], mert ezek rengeteg pluginnal [FIREFOXPLUGINS, GIMPPLUGINS] rendelkeznek.

Egy a fájlkezeléshez közelebb álló kapcsolódó példa a Total Commander fájlkezelő óriási plugin leltára. A Total Commander pluginjai [TCPLUGINS] 4 kategóriába vannak szétosztva, ahol mindegyik kategória saját, jól definiált funkcionalitással és interfésszel rendelkezik. A plugin interfész jó tervezését bizonyítja a leltárban található pluginok száma, ami az általam látottak alapján körülbelül 200-ra tehető.

Az eddig leírtakon kívül van még két ok, amiért különösen elegáns megoldásnak tartom a pluginok használatát. Az egyik az, hogy elég egy minimális képességekészletet implementálni az alkalmazásunkban, ami a legtöbb felhasználót kielégíti. Ilyen módon a kezdő felhasználókat nem terheljük túl egy óriási képességekészlettel.

A másik ok az, hogy a platformspecifikus képességeket külön platformspecifikus pluginokkal oldhatjuk meg. Például ha a kijelölt fájlokat CD-re vagy DVD-re akarjuk menteni, akkor az adott platformon jelenlevő CD / DVD-író alkalmazást hívhatjuk meg, esetleg a különböző platformokon jelenlevő függőségeknek megfelelően külön verziókat készíthetünk az adott pluginból.

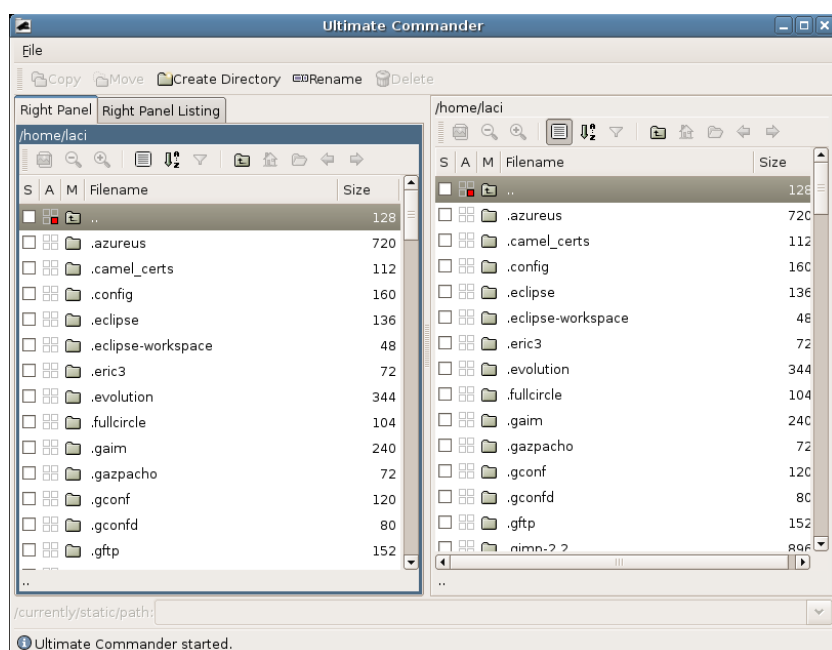
1.3. Technológiai áttekintés

A projekt Mono környezetre építve kerül fejlesztésre C# nyelven a Gtk# grafikus eszközkészletet felhasználva.

Ezen kívül a Glade [GLADE] grafikus interfész tervező alkalmazást használtam a grafikus felület megtervezésére. A Glade speciális GladeXML XML kódot generál, amely a felhasználói felületet írja le. A GladeXML futásidőben kerül elemzésre a Glade programkönyvtárak felhasználásával, amelyek könnyedén megkonstruálják a widget fát.

1.4. Felhasználói interfész tervezés

A felhasználói felület megtervezésére különösen nagy gondot fordítottam, mert rengeteg alkalmazást láttam, aminek a használhatóságát ez a terület nagyon komolyan korlátozta.



1. ábra

Az alábbiakban a felület egyes elemeit fogom felsorolni és az azokkal kapcsolatos tervezési okokra kitérni.

1.4.1. Egyablakos felület

Az 1. ábrán látható a fájlkezelő fő felülete. A tervezés során elsődleges szempontként lebegett előttem, hogy az alkalmazás ne használjon többet egy ablaknál.

Több olyan alkalmazás is létezik, mint például a GIMP [GIMP] képretszáló program, vagy a Glade [GLADE] felhasználói felület tervező, amelyek tipikusan 3-4 ablakot

használnak. Nagyon rossz tapasztalataim voltak ezekkel a programokkal ebben a tekintetben. Igazán zavaró volt, hogy annyi ablakkal kellett dolgoznom és azok átfedték egymást, így próbáltam ezt a tervezést olyannyira elkerülni, amennyire csak tudtam.

A másik kapcsolódó probléma a felbukkanó ablakoknál jelentkezik. A felbukkanó ablakok zavaróak, mert fölöslegesen jóvá kell hagyni őket az egér vagy a billentyűzet segítségével. Alternatív módszerekkel ki lehet váltani a funkciójukat, amint azt az alábbiakban példákkal is alátámasztom.

Bizonyos kivételes esetekben úgy hiszem használható a többablakos megoldás. Az egyetlen ilyen eset, ami eszembe jut, az a a segítség ablak. Ha a felhasználó a segítséget olvassa és rögtön ki akar próbálni egy benne leírt funkciót, akkor van értelme a segítség és az alkalmazásablak közötti váltogatásnak, és a későbbiekben lehetőséget is akarok teremteni arra, hogy a felület különböző részeit „ki lehessen tépni” a főablakból, de az egyablakos felület olyannyira jól működik, hogy ez a fenti képesség aligha nevezhető kritikusnak.

1.4.2. Információs sáv

Az 1. ábrán látható alkalmazásablak alján található az információs sáv. Ebben a sávban jelenik meg minden tájékoztató-, figyelmeztető- és hibüzenet, amely a felhasználói interakciókra reagál.

A tájékoztatóüzenetek esetében csak a szöveg jelenik meg az információs sávban, azonban a figyelmeztetőüzenetek és a hibüzenetek esetében a sáv sárga, illetve piros színnel villog, ezzel felhívva a figyelmet a benne foglalt üzenet jelentőségére.

A legtöbb hagyományos felülettel rendelkező alkalmazás ezt a problémát külön felbukkanó ablak feldobásával oldaná meg. Ez utóbbi megoldás megsértené az egyablakos tervezést, így a jelenlegi alternatívát választottam.

1.4.3. Keretek, vájatok és nézetek

Az 1. ábrán két keret látható. A bal keretben két nézet foglal helyet: az első a bal panel és a második jobb panel listázási konfigurátora, a jobb keretben pedig egy nézet foglal helyet, benne a jobb panellel.

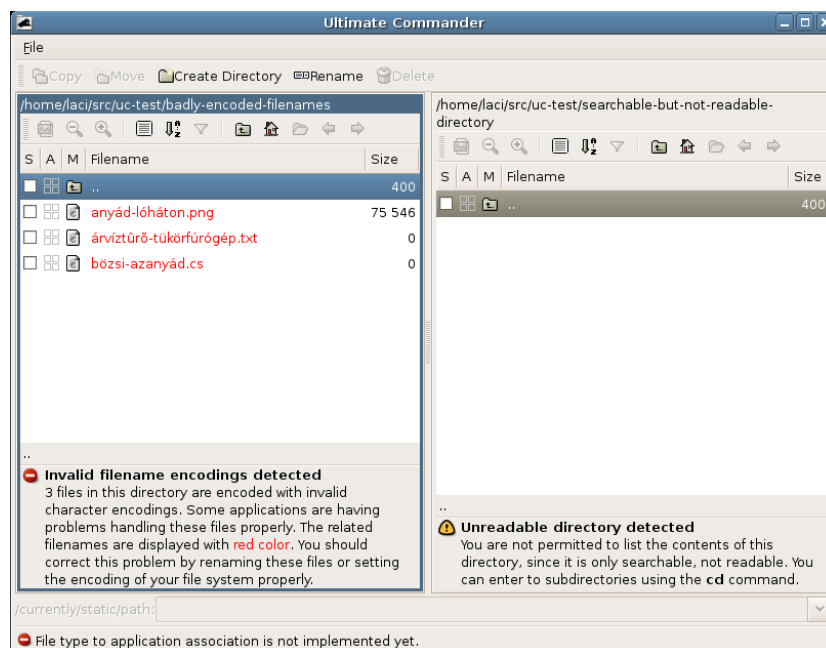
Ez a tervezés azon az ötleten alapul, hogy egy meghatározott számú keret létezik az alkalmazásablakban, egy adott keret pedig funkcionálisan összetartozó nézeteket tartalmaz.

Ha egy keret egy nézetet tartalmaz, akkor a keretnek értelemszerűen nincs füle, amellyel a benne foglalt nézeteket váltogatni lehet, ha pedig több nézetet tartalmaz, akkor azoknak megfelelő számú füle lesz.

Bármely pillanatban egy keret aktív és minden keretnek egy aktív nézete van, így bármely pillanatban egy aktív nézet van, az aktív kereten belüli aktív nézet.

Legvégül szót ejtek a vájatokról. Valójában minden nézet egy-egy vájatban foglal helyet, amely vájatnak az a feladata, hogy kijelyezze az adott nézet címét és annak az aktív, illetve inaktív állapotát.

1.4.4. Beágyazott panel figyelmeztetések



2. ábra

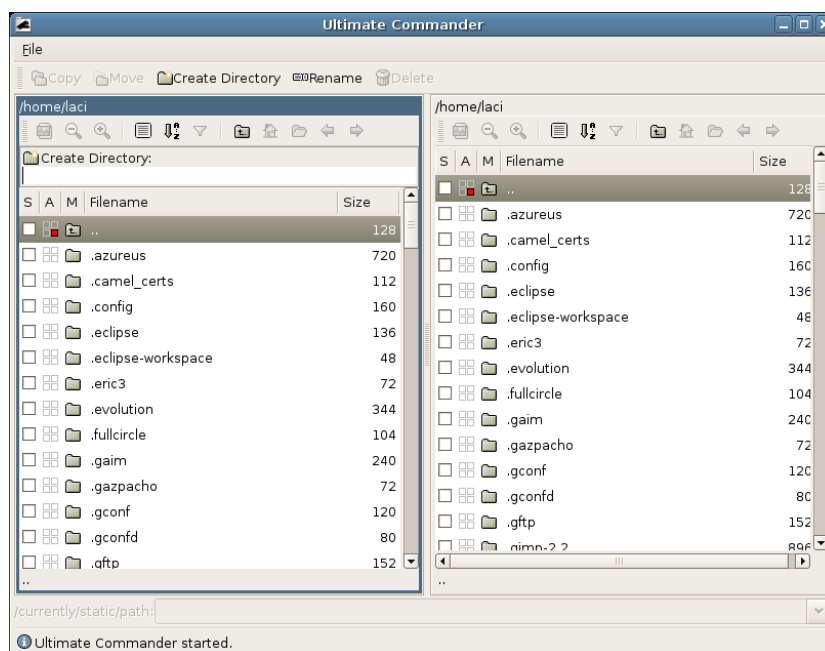
A figyelmeztetések a panel olyan alapértelmezésben rejtett részei, amelyek láthatóvá válnak bizonyos speciális állapot aktivizálódásakor. Jelenleg kétfajta figyelmeztetés van implementálva: a 2. ábrán a bal panelben a „hibás fájlnev enkódolás” figyelmeztetés látható, a jobb panelben pedig a „nem olvasható könyvtár” figyelmeztetés látható.

A „hibás fájlnev enkódolás” figyelmeztetés akkor bukkan fel, ha az aktuális könyvtárban valamelyik fájlnev nem megfelelő enkódolással van kódolva. Linux alatt nincs szabványos fájlnev kódlap definiálva, hanem az alkalmazás lokális beállításai határozzák meg az alkalmazott kódolást, ezért tipikusan a rosszul beállított rendszereken, vagy egy átfogó fájlnev kódolási átállás esetén tapasztalható ez a jelenség. Bármely ilyen esetben hasznos ez a figyelmeztetés.

A „nem olvasható könyvtár” figyelmeztetés akkor bukkan fel, amikor az aktuális könyvtár csak futtatható/kereshető attribútummal rendelkezik, de nincs beállítva rajta az olvasható attribútum. Ilyen esetben a könyvtár tartalmát nem engedi listázni a rendszer.

A hagyományos alkalmazásokban a fenti figyelmeztetéseket felbukkanó ablakokkal implementálnák, ezért az egyablakos elvhez ragaszkodva a jelenlegi, alternatív megvalósítást alkalmaztam.

1.4.5. Új könyvtár létrehozása



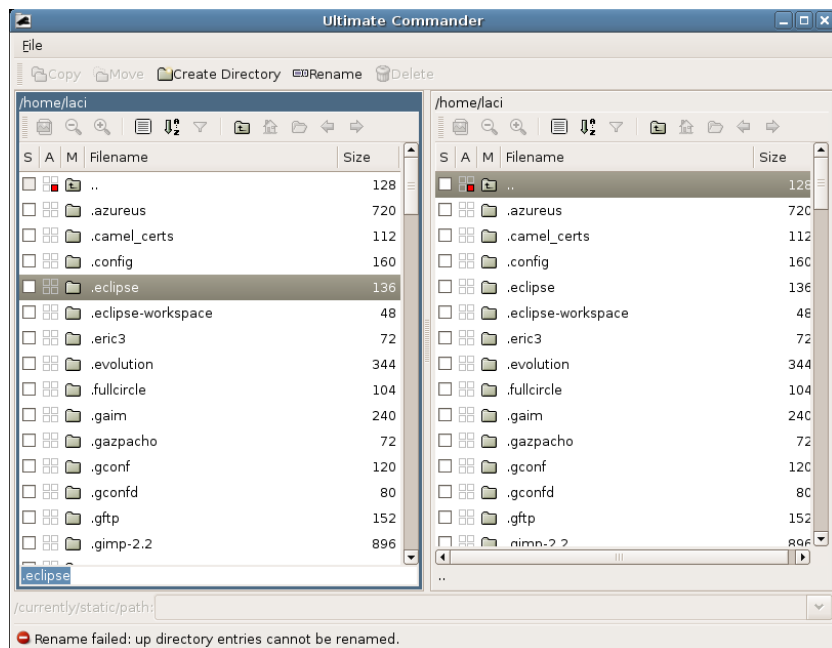
3. ábra

Új könyvtár létrehozását a menüből, az eszközsáv nyomógombjával, vagy pedig az Alt-E billentyűkombinációval lehet aktiválni.

A bevitel során a panelen belül előbukkan egy új beviteli rész, amely az új könyvtár nevét várja. Enter billentyűvel lehet a könyvtár nevet jóváhagyni, vagy Escape billentyűvel megszakítani a műveletet.

Bármely nem megengedett esetben, például ha olyan nevű könyvtárat akarunk létrehozni, amely név már létezik az adott könyvtárban, a hibáról az információs sáv értesíti a felhasználót.

1.4.6. Átnevezés



4. ábra

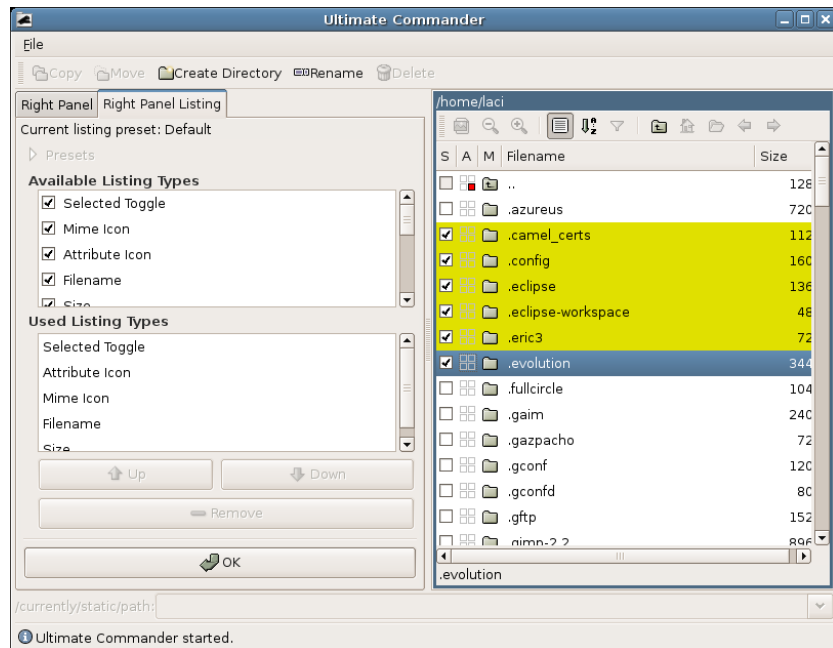
Az átnevezést a menüből, az eszközsáv nyomógombjával, vagy pedig az Alt-R billentyűkombinációval lehet aktiválni.

A bevitel során az aktív panel státuszsora érzékenyé válik és fókuszt kap. A felhasználó végső fájlnevet az Enter billentyűvel hagyhatja jóvá és az Escape billentyűvel szakíthatja meg a műveletet.

Bármely nem megengedett esetben, például ha olyan névre akarjuk átnevezni az aktuális fájlt, amilyen nevű fájl már eleve létezik a könyvtárban, vagy a felmenő könyvtárbejegyzést akarjuk átnevezni, a hibáról az információs sáv azonnal értesít minket.

1.5. Interakció tervezés

A jó interakció tervezés eredményeképpen az adott alkalmazással természetes, gördülékeny és problémamentes az kommunikáció, míg a rosszul megtervezett interakció sárba tiporhatja a felhasználói élményt, ezért ennek a területnek is a tőlem telhető legnagyobb figyelmet igyekeztem szentelni. Az alábbiakban részletezem, hogy mely elemeket tartottam fontosnak ebből a szempontból.



5. ábra

1.5.1. Billentyűzettel történő vezérlés

Joggal várhatjuk el, hogy az ortodox fájlkezelőkhöz hasonlóan képesek legyünk a jobb és bal panel között váltani a Tab billentyűvel, ezt azonban a gyakorlatban nem annyira egyértelmű megvalósítani.

A GTK alapértelmezésben a Tab hatására végiglépteti a fókuszt az összes vezérlőn. Ez a jelenlegi esetben azt jelenti, hogy miután az egyik panel aktív, a Tab hatására általában a másik panel státuszsorának az első nyomógombjára ugrik a fókuszt.

A kívánt hatás elérése érdekében felül kell definiálnunk a panelen belüli TreeView widget billentyű elfogó eseményét és meg kell akadályoznunk annak a propagálását, amivel a fókuszt tovább elváltaná.

1.5.2. Lebegő kijelölés

Bármelyik panelen belül bármely fájl fölött lenyomva a jobb egérgombot és egy másik fájl fölött elengedve azt, az összes közbülső fájl kijelölése invertált lesz. Az 5. ábra jobb panele például azt az esetet szemlélteti, amikor a felhasználó a „camel_certs” könyvtár fölött lenyomja a jobb egérgombot és az „evolution” könyvtár fölött elengedi azt. Ez az interakció természetesnek tűnik, de alapértelmezés szerint a GTK teljesen másképp kezeli a listák kijelölését.

A GTK alapértelmezett működése az, hogy a listaelemek kijelölhetők a Control billentyű segítségével, azonban a kijelölés az a kurzor újrapozicionálásakor azonnal elveszik. Ezt a viselkedését ebben az esetben explicit felül kell definiálni, hogy a kívánt hatást elérjük. A végeredmény egy sokkal kényelmesebb interakció.

1.5.3. Élő panel konfiguráció

Az 5. ábrán a bal keretben a jobb panel listázási konfigurátora látható. A konfigurátor változtatásával egy időben a jobb panel oszlopai is a módosításoknak megfelelően változnak.

A hagyományosan alkalmazott megoldás az lenne, hogy külön dialógusablakban megadjuk a kívánt listázási oszlopokat és az OK gomb megnyomása után az általunk megadott konfiguráció aktualizálódik a panelen. Ez a megközelítés azonban sokkal természetesebb és érthetőbb a felhasználó számára.

1.6. A projekt fogadtatása

A projektről ezidáig csak egy szűk kör szerzett tudomást. Ahogy a projekt az alfa állapotot eléri, tehát a minimális alap funkcionalitással rendelkezni fog, ki fogom adni az első verziót és meg fogom hirdetni minden olyan csatornán, ahol potenciálisan figyelmet kaphat, de addig nem akarok egy félkész szoftverrel előállni és ezzel bizalmatlanságot ébreszteni a majdani fejlesztői és felhasználói bázisban.

Körülbelül 5 személyről tudok, akik kipróbálták a projektet. Ezek közül hárman külföldiek és az IRC csatornákról szereztek tudomást a munkámról, ketten pedig a szűk baráti körömből kerültek ki.

Az első próba után mindegyikük egyöntetűen dicsérte a felület minőségét. Egyik külföldi fejlesztő a projekt korai fázisa ellenére nagyon fellelkesedett és be akart szállni a fejlesztésbe, amire én boldogan lehetőséget is adtam neki és segítséget nyújtottam hozzá. Időközben sajnos kiderült, hogy az illetőnek nem voltak meg a kellő tapasztalatai a fejlesztéshez, de ez akkor is egy nagyon pozitív visszajelzés volt a számomra.

Őszintén hiszem, hogy az alfa állapot elérésével, az első verzió kiadásával és a projekt beharangozásával idővel egy jelentős fejlesztői gárda gyűlhetne össze a projekt körül és a jövőben azon fogok dolgozni, hogy eljussak eddig a célig és ezen is túl.

1.7. Tapasztalatok

Ahogy megismerkedtem a Linux operációs rendszerrel, rögtön éreztem a hiányát egy kiforrott, grafikus alapú, kétpaneles fájlkezelőnek. Akkor még vajmi kevés tudással rendelkeztem a Linuxszal és a programozással kapcsolatban, így aligha tudtam nekikezdeni egy ilyen fájlkezelő elkészítésének, azonban folyamatosan érlelődött bennem a gondolat.

A SourceForge.net-en a projektet 2003 elején regisztráltam, de akkor még nem volt egy átfogó vízióm, sem kellő tapasztalatom a megvalósításhoz. Kezdetben C nyelvet választottam és az Ncurses karaktergrafikus könyvtárral valósítottam meg a felületet.

Később belátva, hogy a karaktergrafikus felület mennyire korlátolt és a C nyelven mennyire nehézkes magasszintű alkalmazást fejleszteni, újrakezdtém a fejlesztést Python nyelven a GTK grafikus könyvtár felhasználásával. Így visszatekintve az akkori állapotra, akkor sem volt egy egységes, szilárd vízióm a projekt egészével kapcsolatban és sok helyütt elveszttem a részletekben, a speciális képességeken történő gondolkodásban és azok implementálásában, így a motivációm idővel lelankadt.

Az előbbi kudarcok ellenére 2005 vége felé újrakezdem a projektet ezen szakdolgozat keretein belül. Időközben platformot váltottam, a Mono környezetre esett a választásom, amely addigra már kellően éretté vált ahhoz, hogy lehessen rá építeni.

Kezdetben kissé aggódtam a C# nyelv jellege miatt. A Python maximális produktivitást adott és úgy éreztem, hogy a C# korlátozni fog ezen a téren. Eleinte még nem ismertem a C# nyelvet, de tudtam, hogy nagyon hasonló a Java nyelvhez és így jellegében hasonlít hozzá.

Az ilyen szigorúan típusos nyelvekben, mint a C#, vagy a Java a fejlesztőnek nagyon explicitnek kell lennie. Értem ezalatt azt például, hogy explicite be kell vezetnie az új változókat, és azoknak megadni a típusát. Ezzel szemben a Python gyengén típusos szkriptnyelv, ahol nem kell kifejezetten bevezetni a változókat, csak értéket adni egy addig még nem létező változónak, mire az létre jön. Python-ban a változók típusa sem kötött, így egy másik típusú érték is értékül adható ugyanannak a változónak.

Ahogy megismertem a C# nyelvet, elkezdtem benne fejleszteni és idővel hozzászoktam, rájöttem hogy a C# valójában jobb választás, mint a Python.

Python-ban a nyelvi egységek között „gyenge kötések” jönnek létre. Ezzel azt akarom mondani, hogy futásidőben dől el, hogy bizonyos interfészek léteznek -e azoknak az elérésekor. Például ha egy adott objektumnak az egyik metódusára hivatkozok, Python-ban megtehetem azt hogy egy nem létező metódusra hivatkozok és futás közben fog erre vonatkozó kivételt dobni az értelmező. C#-ban ez nem fordulhat elő, mert a fordító rögtön értesít erről a hibáról. Ezen jellegi különbségből fakadóan beláttam, hogy az erősen típusos nyelvek sokkal jobbák a robusztus projektek fejlesztéséhez, mint a gyengén típusos szkriptnyelvek.

A Mono további előnye a Python-hoz képest, hogy a kód jelentősen gyorsabban fut. Az általam olvasott statisztikák alapján úgy gondolom hogy ez a gyorsaság körülbelül ötszörös mértékű lehet a legtöbb esetben, bár meglehet hogy ez nem annyira érezhető, mert az alacsonyabb szintű API hívásokat C-ben implementálták és a program jelentős részét az I/O teszi ki.

Fejlesztői eszközök szempontjából szerencsére jelentős fejlődés ment végbe az utóbbi időkben. Kezdetben az Eclipse [ECLIPSE] integrált fejlesztőkörnyezetet használtam, ami csak egy alapvető keretrendszert adott C# szintaxiskiemeléssel az Improve C# plugin [IMPROVEPLUGIN] használatával.

Időközben a MonoDevelop integrált fejlesztőkörnyezet [MONODEVELOP] is elég éretté vált ahhoz hogy áttérjek rá, ami nagyon jelentősen megnövelte a produktívitásomat.

A MonoDevelop Stetic grafikus felhasználói felület építő komponensével még apróbb hibák miatt nem tudtam kiváltani a Glade-et, de a közeli jövőben ez a tervem, mert a Stetic tovább gyorsíthatná a fejlesztést az integrált képességei segítségével és gyorsabb kódot is eredményezne a generatív megközelítése révén. Addig is a felmerülő hibajelentéseket el fogom küldeni a MonoDevelop projektnek.

1.8. Jövőbeli tervek

Az alábbiakban a jövőben megvalósítandó képességeket sorolnám fel vázlatos formában:

1. További alapvető, nem-atomikus fájlműveletek implementációja: másolás, mozgatás, törlés. Ezek egy „job” elnevezésű új műveleti primitív segítségével lesznek implementálva, amely potenciálisan újrafelhasználható minden folyamat jellegű képesség implementálásakor. A job-ok vizualizációja a Mozilla Firefox [FIREFOX] böngésző Download Statusbar [DOWNLOADBAR] kiterjesztésének a mintájára történne.
2. Windowsra történő portolás. Március 13-án készítettem egy minimalizált verziót, a projektből, [UCWINVERSION] hogy kipróbáljam, hogy milyen a megjelenése Windows alatt, ami elég pozitív eredményeket adott. A jövőben be kell majd burkolni a natív .NET és a Unix-specifikus fájlkezelő mechanizmusokat ennek a célnak az elérése érdekében.
3. Globális VFS (viruális fájlrendszer) implementálása. A hagyományos VFS a hálózati protokollokat és a csomagolt fájlformátumokat kezeli. A globális VFS rendelkezik egy legfelső névtérrel, amelyből az elérhető protokollok nyílnak. Így például lehetőség lenne a felhasználó által gyakran használt FTP szervereket kilistázni az ftp:// útvonalon, vagy az alábbi „Depot démon” fejezetben leírt névterekbe leágazni.
4. Plugin interfész implementálása. A pluginokkal hatalmas mennyiségű egzotikus és / vagy platformspecifikus képességet lehetne implementálni.

2. A fájlmenedzsment alternatív lehetőségei

Napjaink számítógépei egyre kifinomultabb eszközöket nyújtanak számunkra információink kezeléséhez, mégis sokszor pokolinak tűnik ez a terület. A modern számítástechnika felhasználóinak gyakran komoly problémát okoz csupán egy-egy fájl megtalálása is, de ez még csak a jéghegy csúcsa. Ahogy egyre több információt halmozunk fel, egyre inkább szembesülünk azzal a problémával, hogy képtelenek vagyunk azokat megfelelően rendszerezni és előkeresni.

Bár ez a jelenség részben a felhasználók nemtörődömségének is köszönhető, hogy hagyják elhatalmasodni rendszereiken a káoszt, de az igazság az, hogy a jelenlegi eszközökkel nem is lehetséges igazán finoman rendszerezni és hatékonyan karbantartani adatainkat.

A strukturált információszervezés és keresés a számítástechnikának egy mostohán kezelt területe, amely egy nagyon mély és gazdag problémakört ölel fel. Ezen problémáknak egy része megoldott, mások részben megvalósítottak és egy jelentős hányadukra még nem született megoldás.

Ebben a fejezetben olyan képességeket sorolok fel, amelyek megkönnyítenék a fájlokkal való munkánkat és amelyeket eddig még nem valósítottak meg, vagy megvalósításuk hagy kívánnivalót maga után.

2.1. Integrált média katalógus

Definíció: *elsődleges tároló:* Ez a leggyorsabb és legdrágább adattároló. A mai számítógépekben ez a véletlen hozzáférésű memória (RAM).

Definíció: *másodlagos tároló:* Közepes sebességű és áru adattároló médium. A mai számítógépekben ez a merevlemez.

Definíció: *harmadlagos tároló:* Ez a médium rendelkezik a leggyengébb hozzáférési idővel és a legkedvezőbb tárolóegységenkénti árral. Manapság ezt a szerepet a DVD és CD lemezek töltik be.

Ahogy felhalmozuk az adatainkat, képtelenek vagyunk azokat kizárólag másodlagos tárolón tárolni annak korlátozott kapacitása miatt. Előbb, vagy utóbb harmadlagos tárolóra kényszerülünk menteni legalább egy részüket. A harmadlagos tárolóink nyilvántartása végett valamilyen katalogizáló megoldást kényszerülünk használni.

A média katalogizálás jelenleg csak alkalmazásszinten van megoldva. Léteznek katalogizáló programok, mint például a WhereIsIt [WHEREISIT] katalogizáló program, amelyeket pontosan erre a problémára fejlesztettek ki, de ezek az alábbi hátrányokkal rendelkeznek:

- zárt katalógus fájlformátum használata
- platformfüggőség
- integráció hiánya a külvilággal

Ezen katalogizáló alkalmazások integrációjának a hiányának különösen messze mutató korlátjai vannak. Kellő integráció meglétével a többi alkalmazás lekérdezhetné, hogy bármely a katalóguson kívüli fájl szerepel -e a katalógusban és azon belül hol. Továbbá katalógust a virtuális fájlrendszer részévé is lehetne tenni, ezzel mélyebben integrálva ezt a képességet a rendszerbe.

2.2. Globális metafájl adatbázis

Fájljaink egy része a másodlagos tárolóról közvetlenül elérhető, másik részük pedig harmadlagos tárolókon van archiválva. A másodlagos tárolón található fájljaink a fájlkezelőnk segítségével könnyedén elérhetőek, a harmadlagos tárolón található fájljaink viszont csak egy speciális katalogizáló alkalmazás segítségével böngészhetőek.

Ezen eltérő hozzáférési formák inkonzisztenciát okoznak fájljaink elérésének a tekintetében, ezért egy konzisztens hozzáférési mód nagyban megkönnyíteni az egységes elérésüket.

2.3. Strukturált metaadat asszociáció és lekérdezés

Sok alkalmazástípus rendel metaadatot az egyes fájlokhoz. Tipikusan a zenelejátszó programok lehetőséget adnak zenéink tetszési indexének megadására egy tízes skálán, vagy a fotó böngésző alkalmazásokkal hozzátársíthatunk különböző személyeket vagy helyszíneket az egyes képekhez.

Mivel a fájlokhoz történő metaadat asszociáció nem szabványosított, a különböző alkalmazások és platformok mind más-más utat választanak a megvalósításukra.

A Linux operációs rendszeren például az utóbbi években széleskörűen elérhetővé vált a fájlrendszer egy kibővítése, a kiterjesztett attribútumok. A kiterjesztett attribútumok segítségével név-érték párokat rendelhetünk fájljainkhoz. Azonban mi van olyankor ha nem Linuxot használunk?

Hiányzik egy olyan megoldás, amely a következő képességekkel bír:

- Potenciálisan platformfüggetlenül implementálható.
- A metaadatok sémája definiált, így azok relációs adatbázisba menthetők és lehetőség nyílik az összetett lekérdezésük hatékony kiszolgálására.
- Rendelkezik egy létező struktúrával a szabványos fájlformátumok metaadatainak a befogadása számára, és képes kicsomagolni azokat a kérdéses fájljokból.
- Lehetőséget ad a felhasználók számára, hogy a létező struktúrát kibővítsék nekik megfelelően.
- A fájlrendszeren végbement változásoknak megfelelően lehetőség szerint folyamatosan, vagy periodikusan aktualizálja a belső állapotát.

2.4. Filmek vizuális asszociációja

A modern operációs rendszerek segítséget próbálnak nyújtani abban, hogy a felhasználók könnyen felismerhessék filmjeiket. Teszik ezt úgy, hogy egy előre meghatározott pozícióból kiragadnak egy képet, amit lekicsinyítenek, amely alapján a felhasználó felismerheti a kérdéses filmet. A vizuális asszociáció ötlete nagyon jól működik és

természetes megoldás, de a jelenlegi implementációk nagyon szegényesen végzik a dolgukat.

A legnépszerűbb operációs rendszer, a Windows XP az első képkockát ragadja ki a filmből és ebből készíti kisképet. Mivel a felvételek első kockája az esetek többségében teljesen fekete, az így keletkező kép semmilyen többletinformációt nem szolgáltat a felhasználó számára a fájlneven túl.

A GNOME környezet Nautilus fájlkezelő alkalmazása is szinte ugyanezt a stratégiát használja annyi különbséggel, hogy nem a film elejébe, hanem az 1/3 részéig keres bele és az abban a pozícióban található képkockát menti le kisképbe. Bár ez a megoldás a gyakorlatban jobban működik, azonban az esetek többségében a kiragadott képkocka itt sem eléggé releváns ahhoz, hogy a kapcsolódó filmet könnyen fel lehessen ismerni az alapján.

A Windows XP és a GNOME környezet Nautilus fájlkezelőjét látva, ezeknek a kiskép megszerzési stratégiája aligha nevezhető sikeresnek. A teljes filmhossz képkockáinak csak egy bizonyos hányadát tudja az agyunk sikeresen hozzárendelni az adott filmhez, így alapvetően kétféle stratégiával érdemes próbálkozni:

- 1) A fejlesztő és a felhasználó számára is az egyszerűbb megoldás, hogy véletlenszerűen kiválasztunk x darab képet a filmből és azokat felkínáljuk a felhasználónak. Ezekből a képekből a felhasználó tetszőlegesen kiválaszthatja a neki legjobban megfelelőt, vagy ha egyik képet sem találja igazán relevánsnak, akkor újra kérheti az x darab kép véletlenszerű kiválasztását. A generált képek számától és a filmbeli releváns képkockát számától függően a gyakorlatban általában 10 - 20 generált kép közül van egy, ami megfelelő.
- 2) Bizonyos esetekben amikor a felhasználó ragaszkodik egy bizonyos képkockához, vagy az adott filmnek kevés egyedi képkockája van amire könnyen lehet asszociálni, érdemes egy másik utat választani. Ilyen esetekben hasznos lenne egy olyan minimális filmlejátszót konstruálni, amely képes kimenteni egy tetszőleges képet a filmből amit a felhasználó kijelöl. Erre a megközelítésre valószínűleg kevésbé lenne igény, mert a felhasználó számára lényegesen megterhelőbb ez a művelet.

Az első stratégia alapján készítettem egy implementációt. A Nautilus fájlkezelő külső programokat hív meg a kisképek készítéséhez. A filmek esetében ez a program a Totem videó lejátszó része, a totem-video-thumbnailer parancssori alkalmazás.

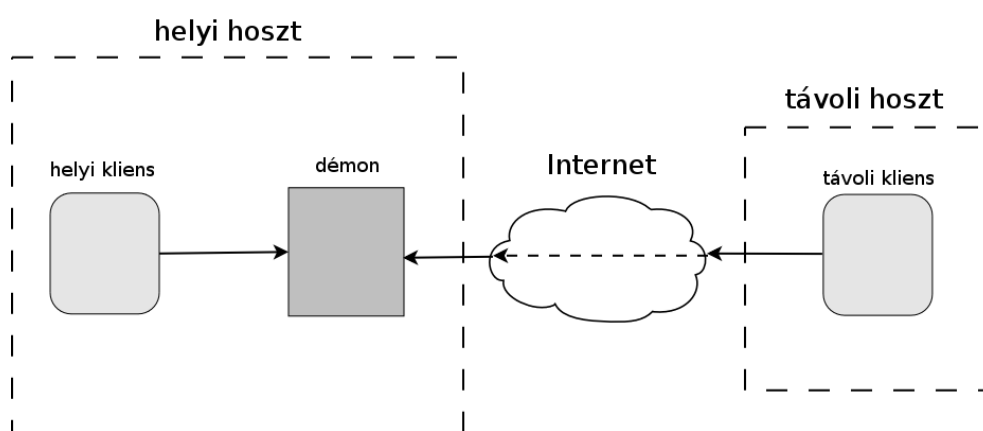
A totem-video-thumbnailer programot kiindulási alapként felhasználva, azt módosítottam, hogy a kívánt funkcionalitást elérjem:

- Először minden olyan részét eltávolítottam, amely nem szükséges az általam igényelt funkcionalitáshoz.
- Ezután kijavítottam egy apró hibát, amely hibás verziójú függőséget igényelt volna fordításkor, így az alkalmazást bizonyos esetekben nehézkes lett volna lefordítani.
- Következő lépésként bilineáris szűrő algoritmust használtam az eredetileg használt jóval gyengébb minőségű képméretező algoritmus helyett.
- Végül lépésként pedig implementáltam a csoportos véletlenszerű képmentést és az ahhoz tartozó parancssori opciókat, amelyekkel megadható, hogy hány kisképet generáljon a program és hogy milyen felbontásban tegye azt.

Mivel ez egy parancssori alkalmazás, a jövőben fogok írni hozzá egy grafikus előtétet ami keresztül a felhasználók kényelmesen ki tudják választani a számukra megfelelő képkockát.

3. A Depot démon

Az előző fejezetben felsorolt képességek olyan szolgáltatások, amelyeket minden alkalmazás számára elérhetővé kellene tenni. A fájlrendszer megfigyelése és a metaadatok változásának az aktualizálása aktív entitást kíván, amelyet legcélszerűbb rendszerdémonként implementálni, amely folyamatosan fut és a többi alkalmazás kéréseit kiszolgálja. Így a démon egy központi rendszerszolgáltatást biztosítana, amelyet bármely helyi, vagy távoli kliens elérhetne, ahogy az az alábbi ábrán látható:



A kliens lehet például egy fájlkezelő pluginja, vagy egy parancssori alkalmazás.

Az alábbiak egyszerűbb és egységesebb tárgyalása érdekében bevezetek 3 definíciót:

Definíció: *metafájl*: Egy olyan bejegyzés a fájlrendszerben, amihez nem tartozik valódi adat, csak metaadatok az adott fájlról.

Definíció: *online fájl*: Egy hagyományos, a másodlagos tárolón található fájl, ami közvetlenül elérhető.

Definíció: *offline fájl*: Egy archivált fájl, amely harmadlagos tárolóról elérhető és szerepel a katalógusban metafájlként.

3.1. Névterek és szignatúrák

Az előző fejezetben felvázolt integrált média katalógust és globális metafájl adatbázist célszerű lenne külön névtéreként implementálni. Ily módon standard URI hivatkozássémával mindkét névtér konzisztensen elérhető lenne, akár csak a fájlrendszer többi része.

Szükség van továbbá egy olyan mechanizmusra, amely lehetővé teszi, hogy a két névtér fájljait egymáshoz tudjuk rendelni, illetve a külső fájlokhoz is meg tudjuk feleltetni ezen névterek fájljait. Ebben a feladatban van kulcsfontosságú szerepe a szignatúráknak.

A szignatúra egy karakterisztikus érték, amely bármely fájlból kinyerhető egy nagyon rövid idő alatt. Ez az érték arra szolgál, hogy eldöntsük, hogy két fájl azonos -e vagy sem. A gyakorlatban annak a valószínűsége, hogy ugyanolyan fájlok különböző szignatúrával vannak jelen, szinte zérus.

A szignatúra kinyerésének a jelenlegi módja az, hogy az adott fájl közepéből kiolvassunk 4096 bájtot, vagy ha a fájl kisebb, mint 4096 bájt, akkor a teljes tartalmát vesszük és az így kapott bájtsorozat MD5 ellenőrzőértékét vesszük eredményül.

Ez a módszer nagyon hatékonynak és jól működőnek bizonyult a gyakorlatban.

3.2. A katalógus névtér

A katalógus névtér egy speciális, dedikált névtér, amelyből minden katalógusbeli fájl elérhető. Ezen névtér standard hivatkozássémája segítségével a katalogizált fájlokra általános formában lehet hivatkozni alkalmazástól függetlenül.

A névtér legfelső szintjének a könyvtárai a bekatalogizált médiumoknak felelnek meg, ezen könyvtáraknak a tartalma pedig a médiumok könyvtárszerkezetét leíró metafájl hierarchia.

3.3. A globális névtér

Az előző fejezetben szereplő globális metafájl adatbázis lényegében a fájlrendszer egy olyan jellegű kibővítése, amely online és offline fájlkat egyaránt tartalmaz, ahol az offline

fájlok meta-fájlok, amelyek a katalógus névtérbe mutatnak az aktuális fájlokra, amelyeket reprezentálnak.

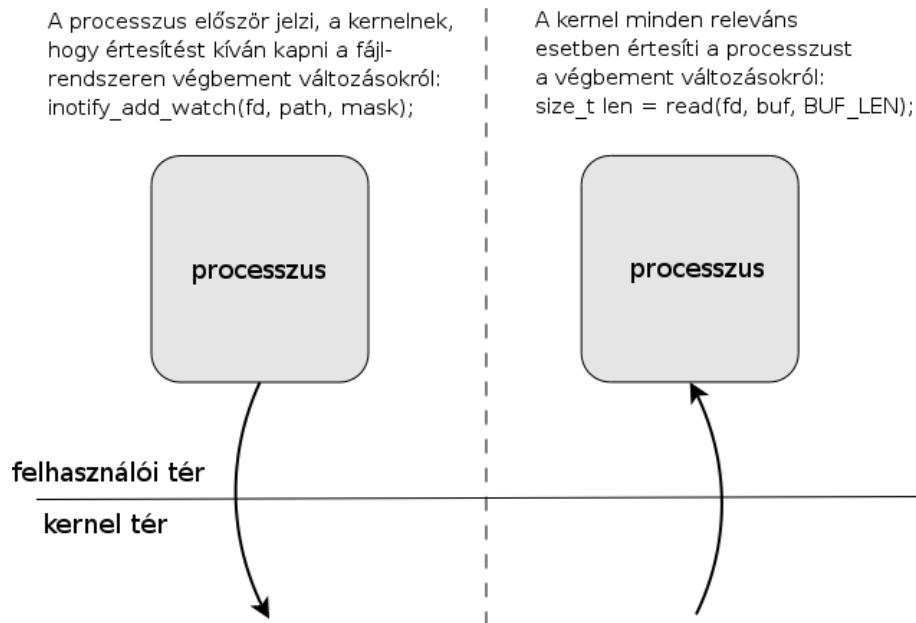
A globális névtér pedig egy speciális, dedikált névtér, amelyből minden globális meta-fájl adatbázisbeli fájl elérhető.

Ebben a kibővített modellben minden fájl az alábbi három metaállapot egyikbe tartozik:

<i>online?</i>	<i>offline?</i>	<i>metaállapot</i>	<i>megjegyzés</i>
igen	nem	besorolt	A fájl jelen van a globális névtérben és be van sorolva archiválásra.
nem	igen	katalogizált	A fájl már archiválásra került, de nem része a globális névtérnek.
igen	igen	kötött	A fájl archiválásra került és a globális névtérnek is része.

A globális névtér egy, a fájlrendszeren kijelölt könyvtárban helyezkedik el. Ezt a könyvtárat monitorozni kell valamilyen módon, hogy értesüljön a démon az újonnan érkezett online fájlokról, a fájlok mozgásáról, törléséről és bármilyen változásról.

A fájlrendszer monitorozását tipikusan periodikus ellenőrzéssel szokták megoldani, azonban Linux alatt nemrég elérhetővé vált egy új szolgáltatás, az inotify fájl-megfigyelő szolgáltatás. Bármely processzus megadhatja az elérési útvonalakat amelyeken belüli változásokra kíváncsi a fájlműveletekkel együtt, amelyeket figyelni kíván és a szolgáltatás a bekövetkezett változásról azonnal értesíti a processzust, ahogy azt a következő ábra is mutatja:



3.4. Strukturált metaadat asszociáció

A fájlok metaadatainak a strukturált tárolása egy osztályhierarchián keresztül történne. A hierarchia tetején az objektum osztály áll, amelyből a fájl és a könyvtár osztály származik. A fájl osztályból 4 további osztály származik, amelyek az adott fájl típusának felelnek meg, rendre szöveg, kép, audio és videó. Ezen tartalmakat jelölő osztályoknak pedig a létező MIME típusok a leszármazottjai.

Az osztályok attribútumai az alábbi típusokba tartozhatnak:

- A *boolean* típus logikai értéket tárol. Igaz vagy hamis értéket vehet fel.
- Az *int* típus egy 32 bites, előjeles egész numerikus értéket tárol.
- A *string* típus egy karakterláncot tárol.
- Az *enum* típus előredefiniált konstansok halmazából egy értéket vehet fel.
- A *set* típus előredefiniált konstansok halmazából vehet fel bármennyi értéket.

Saját osztályok a fenti osztályokból származtathatóak és a globális névtér bármely fájlja leszármaztatható a kívánt saját osztályba, amelynek az attribútumai ezután megadhatóak.

Ezen típusok segítségével definiálhatók a MIME típusokat leíró alapsztályok és a felhasználó által származtatott saját osztályok attribútumai.

3.5. Kapcsolódó projektek

Több megoldás is született már, amelyek valamiféle adatbázist készítenek a fájlrendszerből, annak a sokrétűbb és gyorsabb keresése érdekében.

A Depot ezektől lényegesen eltér a következő szempontokban:

- A strukturált metaadat hozzárendeléssel lehetőséget ad a széna szálak rendezésére, ahelyett, hogy csak keresne a szénakazalban.
- A katalógus névtér segítségével egy központi katalógus szolgáltatást biztosít
- A globális névtér segítségével konzisztens hozzáférést nyújt az online és offline fájljainkhoz egy névtér alatt.
- A szignatúrák segítségével lekérdezhető bármely külső fájl a katalógus és / vagy a globális névteren belüli útvonala.

A kapcsolódó projektek listája:

- Beagle [BEAGLE]: A Beagle keresőrendszer kereshetővé teszi a felhasználók személyi információs terének keresését a Lucene.Net [LUCENE] keresőmotor segítségével.
- SpotLight [SPOTLIGHT]: A SpotLight a Mac OS X Tiger keresőszolgáltatása
- WinFS [WINFS]: A WinFS a következő generációs Windows, a Vista metaadat fájlrendszere.
- SkyFS [SKYFS]: A SkyFS a SkyOS operációs rendszer metaadat fájlrendszere.
- Google Desktop Search [GOOGLEDESEARCH]: A Google Desktop Search a Google a Windowst futtató asztali gépekre szánt keresőrendszere.
- Kat [KAT]: A Kat a KDE [KDE] projekthez kapcsolódó keresőrendszer.
- GNOME Storage [STORAGE]: A GNOME Storage egy alternatív kereső rendszer, amely a GNOME [GNOME] projekt berkein belül indult útnak. A fejlesztése sajnos megszakadt mielőtt éretté vált volna.

4. PROJEKT ADMINISZTRÁCIÓ

A Szabad Szoftver Univerzum egyre bővülő szolgáltatáshalmazt kínál a fejlesztők részére. A különböző szolgáltatások eltérő értékkel rendelkeznek és mindegyik projekt szabadon választhatja meg, hogy mely szolgáltatásokat veszi igénybe. Az alábbiakban részletezem, hogy melyek azok a szolgáltatások, amelyeket igénybe vettem és milyen módon gazdagították a projekt infrastruktúráját.

4.1. SourceForge.net

A SourceForge.net [SOURCEFORGE] a világ legnagyobb nyílt forráskódú szoftverfejlesztői portálja, amely számtalan szolgáltatást nyújt több, mint 100 ezer nyílt forráskódú projekt részére. A portál egyedülállóan széles szolgáltatáspalettával rendelkezik, amelyet nem fogok részletekbe menően leírni. Csak azokra a szolgáltatásokra fogok kitérni, amelyeket felhasználtam.

4.1.1. Verziókövetés

A fejlesztés szempontjából a legkritikusabb szolgáltatás a verziókövetés. A projekt kezdetben CVS szolgáltatást használt, amellyel voltak kisebb problémáim. Egyrészt a szolgáltatásban voltak apróbb fennakadások, másrészt pedig a CVS korlátaiból fakadóan nem lehetett könyvtárakat, illetve modulokat eltávolítani, így néhány üres könyvtár fölöslegesen szennyezte a könyvtárhierarchiát.

2006 februárjában a SourceForge lehetőséget nyújtott a Subversion (SVN) verziókövető rendszer használatára, amelyet áprilisban használatba is vettem és ezzel minden fent említett problémám megoldódott. Az SVN egyértelműen egy evolúciós lépés a CVS után. A parancssori interfészük hasonló, de a belső struktúrájuk teljesen eltérő. SVN-ben konstans idő alatt el lehet érni bármely korábbi revíziót, ezzel szemben a CVS-ben minél korábbi revízióhoz akarunk hozzáférni, annál erőforrásigényesebb a hozzáférés. Továbbá SVN-ben a revíziók számai is sokkal tisztábbak, egyetlen számból állnak amelyekre nagyon kényelmes hivatkozni.

Az SVN-re történő költözéssel együtt egyúttal megoldottam annak a biztonsági mentését is. Az rsync inkrementális fájlátviteli protokollon keresztül letölthetőek a projektek SVN raktárai a SourceForge-ról. A saját gépemre a cron szolgáltatás használatával megoldottam az SVN raktár óránkénti biztonsági mentését a gépemre.

4.1.2. Levelezőlisták

A levelek elsődleges kommunikációs médiumként szolgálnak a projektek fejlesztői és felhasználói részére, így a levelezőlisták jelenléte kulcsfontosságú.

4 különböző levelezőlistát állítottam be: egyet a felhasználók, egyet a fejlesztők részére, egyet az SVN fejlesztői módosítások számára, egyet pedig az új verziók meghirdetésére.

A SourceForge rendelkezésre bocsájt egy hasonló szolgáltatást is, a fórumokat. Ez utóbbiak elsődlegesen web alapúak, ami miatt nem is használtam őket, mert erre a célra a levelezőlistákat jobb módszernek tartottam.

4.1.3. Követők

A követők (trackers) olyan eszközök, amelyekkel nyilván lehet tartani a különböző fejlesztéssel kapcsolatos egységeket, mint például a hibajelenségek, vagy a képesség igénylések. A levelezőlistákkal szembeni előnyük, hogy jobban szem előtt vannak és megmaradnak az explicit lezárásukig, tehát az igény kielégítéséig, ellentétben a levelezőlistákkal ahol az ilyen bejelentések könnyebben elveszhetnek.

4.1.4. Szoftver térkép

A SourceForge átfogó kategorizáló rendszere a szoftver térkép, ahol minden projekt jelen van különböző szempontok szerint, mint például a felhasznált programozási nyelv, a projekt érettsége, vagy az aktivitása. Ezen attribútumok megléte mellett nagyon kényelmes egy-egy szoftver kiválasztása a sok közül. Minden projekt rendelkezik egy kapcsolódó projekt lappal [UCSFPAGE] is, amiről összefoglaló információkat lehet kapni a kérdéses projektről és az összes kapcsolódó szolgáltatását el lehet érni.

4.1.5. Webtér

A webtérben az adott projekt honlapja helyezhető el. Lehetőség van PHP és MySQL használatára, de a gyakorlatban elég nehézkesen használható ez a szolgáltatás. Egyrészt a sebessége hagy némi kívánnivalót maga után, másrészt pedig a projekt web könyvtára közvetlenül nem írható. Az írást csak egy speciális könyvtárban lehet igénybe venni, ami nagyon megnehezíti a különböző webalkalmazások konfigurációját.

Idővel a MediaWiki tartalomkezelő rendszerre esett a választásom, mint a projekt portál motorja. Hosszú próbálkozás során sikerült feltelepítenem a MediaWiki-t a SourceForge webtérbe és írtam egy részletes hogyan dokumentumot ami ezt a műveletet részletezi [MEDIAWIKISF], hogy másoknak ne kelljen végigmenniük ezen a tortúrán.

Ironikus módon mivel a webszolgáltatás elég lassúnak bizonyult, nem sokkal a MediaWiki a SourceForge-ra történő gyötrelmes feltelepítése után, 2006 márciusában átköltöttem honlapot az általam regisztrált saját domainre [UCHOME] ami a MediaCenter [MEDIACENTER] szolgáltató egyik szerverén kapott helyet.

4.2. freenode

A freenode [FREENODE] IRC szolgáltatást nyújt a szabad és nyílt forráskódú projektek részére. A projektek regisztrálhatják magukat és dedikált csatornákat igényelhetnek. Az IRC nagyszerű kiegészítés a levelezőlisták mellé, mert bár alapjában véve ugyanazt a szerepet tölti be, de azonnali jellege miatt sokkal gyorsabb és az összeszokott csapatoknak nagyon hatékony kommunikációt biztosít.

Áprilisban igénybe vettem a freenode.net szolgáltatását és regisztráltam egy dedikált csatornát [UCIRC] a projektnek. Az adminisztrációs késés miatt a rendszergazdák azóta sem válaszoltak, de ettől függetlenül a csatornát addig is használatba vettem.

4.3. CIA

A CIA [CIA] valósidejű statisztikákat és IRC botokat nyújt a nyílt forráskódú projektek részére. A fejlesztők verziókövető rendszerekbe bevitt változtatásaik az aktuális változtatások azonnal megjelennek a CIA felületén, amiből statisztikát is készítenek, így

valós időben, pontosan nyomon lehet követni a projektek, és azokon belül az egyes fejlesztők aktivitását.

A CIA adminisztrátorokat meg lehet kérni, hogy az IRC botokat irányítsák át a projekt csatornára. Amint egy fejlesztő változtatást visz be a verziókövető rendszerbe, a csatornán állomásozó bot rögtön üzenetet küld amelyben szerepel, hogy melyik fejlesztő milyen változtatást hajtott végre, így az ott lévők azonnal tudomást szerezhetnek a forráskód változásáról.

2006 áprilisában regisztráltam a szolgáltatásukat [UCCIA] és egy üde színfoltnak tartom őket a többiek mellett.

4.4. Gmane

A Gmane [GMANE] egy email-news átjáró. Lehetővé teszi, hogy levelezőlisták tartalmát news szolgáltatáson keresztül is lehessen olvasni. Ezen kívül archivál is minden hozzá beérkezett levelet, alternatív webfelületeket biztosít az olvasásra, RSS formátumban elérhetővé teszi a listák tartalmát és a Mail Archive [MAILARCHIVE] szolgáltatásnak is átküldi őket további redundáns archiválásra.

2006 áprilisában vettem igénybe a szolgáltatásukat [UCGMANE, UCMAILARCHIVE].

4.5. freshmeat

A freshmeat [FRESHMEAT] tárolja a világ legnagyobb Unix és keresztplatform szoftver nyilvántartását. A szolgáltatás jellegében ugyanaz, mint a SourceForge szoftver térkép, de itt kritérium, hogy az adott szoftver fusson valamilyen Unix platformon.

2006 áprilisában regisztráltam [UCFRESHMEAT] a projektet a nyilvántartásukba.

5. PROJEKT MARKETING

Igyekeztem a projekt marketing részére maximális figyelmet fordítani. A kezdeti figyelem felkeltésének nagy szerepet tulajdonítok, mert ezen állhat vagy bukhat sok felhasználó megnyerése. Az alábbiakban leírom, hogy milyen elemeket használtam ebből a célból.

5.1. Projekt logó

A SourceForge.net „Help Wanted” szekciója lehetőséget biztosít a projekt adminisztrátorok részére, hogy segítséget kérjenek a projektjeikkel kapcsolatos munkálatokra. Éltem ezzel a lehetőséggel és grafikusokat kerestem a projekt logójának az elkészítéséhez. A felhívásomban leírtam a grafika stílusát, amit vártam és mellékeltem egy kép hivatkozást is, amin egy parancsnoki sapka volt látható.

Körülbelül egy héten belül hárman vették fel velem a kapcsolatot és az ő munkájukat [UCARTWORK] összeszerkesztve megszületett a projekt logó, ami nagyon jól sikerült és egyéni identitást adott a projektnek.

5.2. Flash videók

Szükségem volt egy eszközre, amellyel rögtön demonstrálhattam a projektet az újonnan érkező felhasználóknak. Egy kép többet ér ezer szónál, szokták mondani és a képernyőképek valóban kiváló demonstrációként szolgálnak és széles körben alkalmazottak.

Azonban létezik egy ennél is jobb eszköz: a videó. Az utóbbi években a sávszélesség rohamosan növekedett és a flash lejátszók képessé váltak videólejátszására. A vnc2swf program [VNC2SWF] pontosan ebből a célból született. A flash videóknak több előnyük is van. Egyrészt kevesebb tárterületet foglalnak, másrészt pedig több böngésző fel van készítve a flash lejátszásra, mint a különböző videóformátumok lejátszására. Írtam egy hogyan dokumentumot a webnaplómban [VNC2SWFHOWTO] a flash videók készítéséhez szükséges optimális környezet beállításáról és az új képességek

implementálásakor több ilyen videót is készítettem [UCSCREENCASTS], amelyeknek nagyon pozitív fogadtatása volt mindazok körében, akik látták.

5.3. Barátságos arculat kialakítása

A projekt adminisztrációjának minden lépésében arra törekedtem, hogy aki kapcsolatba lép a projekttel bármely kapcsolódó szolgáltatáson keresztül, az könnyen boldoguljon vele. Ez megnyilvánult például a levelezési listák leíró szövegében, a projekt logóban vagy abban, hogy minden releváns helyre betettem egy-egy beszédes képernyőképet, de a legfontosabb elem ebből a szempontból a projekt honlap volt.

Az első honlapot 2005 novemberében hoztam létre. Ez kezdetben DokuWiki [DOKUWIKI] tartalomkezelő rendszerre épült. Minimálisan átszabtam az eredeti megjelenést, hogy helyet adjak a projekt logónak, de azon kívül nem sok változtatást hajtottam rajta végre.

Később látva a Novell számos projektjének honlapját [NOVELLPROJECTS], úgy döntöttem, hogy a MediaWiki tartalomkezelő rendszert használom, annak robusztus felépítése, elegáns megjelenése és könnyű témázhatósága miatt.

2006 márciusában lecseréltem a DokuWiki-t MediaWiki-re, amit telepítése után rögtön személyre szabtam. Először is beleintegráltam az elkészült logót, másodszor pedig egyedi formára szabtam az alapértelmezett sablont.

A fejléctet és lábléctet a Beagle [BEAGLE] projekt honlapjáról mintáztam, a CSS többi részét pedig egy eleve elkészített MediaWiki bőrből [MWFRAATMANSKIN] vettem át.

5.4. Bátorítás a közreműködők felé

A projekt honlap minden pontján igyekeztem olyan hangnemet megfogni, amely barátságosan fogadja az újonnan érkezőket és bátorítja őket a csatlakozásra, illetve a részükről felmerülő kérdések feltételére.

Ennek a stratégiának a része a kommunikációs infrastruktúra politikájának a felállítása. A csatlakozás lapon [UCCONTACT] felsoroltam azokat a csatornákat, amelyeken a projekt elérhető és egyben egy politikát is meghatároztam, hogy az illető milyen esetben melyik csatornát vegye igénybe.

Ezen kívül a webnaplómban külön kategóriát hoztam létre a projekt részére [UCBLOG], amelyben rendszeresen írok a fejlesztés menetéről.

6. HIVATKOZÁSOK

[NC]	A Norton Commander, az első kétpaneles fájlkezelő: http://en.wikipedia.org/wiki/Norton_Commander
[ORTHODOXFM]	Ortodox fájlkezelők a Wikipédián: http://en.wikipedia.org/wiki/Orthodox_File_Manager
[TC]	A Total Commander fájlkezelő: http://www.ghisler.com/
[TCPLUGINS]	A Total Commander fájlkezelő pluginjai: http://www.totalcmd.net/
[MC]	A Midnight Commander fájlkezelő: http://www.ibiblio.org/mc/
[FMLIST]	Fájlkezelők listája a Wikipédián: http://en.wikipedia.org/wiki/List_of_file_managers
[GLADE]	A Glade felhasználói felület tervező alkalmazás: http://glade.gnome.org/
[ECLIPSE]	Az Eclipse projekt honlapja: http://www.eclipse.org
[IMPROVEPLUGIN]	Improve C# plugin az Eclipse-hez: http://www.improve-technologies.com/alpha/esharp/
[MONODEVELOP]	A MonoDevelop integrált fejlesztői környezet: http://www.monodevelop.com
[FIREFOX]	A Mozilla Firefox webböngésző: http://www.mozilla.com/firefox/
[FIREFOXPLUGINS]	A Mozilla Firefox webböngésző kiterjesztései: https://addons.mozilla.org/firefox/extensions/
[DOWNLOADBAR]	A Mozilla Firefox böngésző Download Statusbar kiterjesztése: http://downloadstatusbar.mozdev.org/
[UCWINVERSION]	Készítettem egy minimális Windows verziót a projektből a felhasználói interfész megtekintése végett: http://laci.monda.hu/blog/2006/03/13/uc-windows-port/
[GIMP]	A GIMP képmanipuláló alkalmazás: http://www.gimp.org/
[GIMPPLUGINS]	A GIMP képmanipuláló alkalmazás pluginjai: http://registry.gimp.org/index.jsp
[WHEREISIT]	A WhereIsIt média katalogizáló alkalmazás. http://www.wheredit-soft.com/
[SOURCEFORGE]	A SourceForge.net a világ legnagyobb fejlesztői portálja a nyílt forráskódú projektek részére: http://sourceforge.net
[UCSFPAGE]	A projekt SourceForge lapja: http://sourceforge.net/projects/ulc

[MEDIAWIKISF]	A hogyan dokumentum, amit írtam a MediaWiki a SourceForge-on történő telepítéséről: http://laci.monda.hu/blog/2006/03/03/mediawiki-on-sourceforge-the-howto/
[UCHOME]	A projekt honlapja: http://ultimatecommander.org
[MEDIACENTER]	A MediaCenter webszolgáltató honlapja: http://mediacenter.hu
[FREENODE]	A freenode IRC szolgáltatás honlapja: http://freenode.net
[UCIRC]	A projekt IRC csatornája megtalálható a freenode.net IRC hálózaton az #uc csatornán.
[CIA]	A CIA szolgáltatás valós idejű statisztikákat és IRC botokat bocsájt a nyílt forráskódú projektek részére: http://cia.navi.cx/
[UCCIA]	A projekt CIA lapja: http://cia.navi.cx/stats/project/ulc
[GMANE]	A Gmane szolgáltatás elérhetővé teszi a levelező listák tartalmát alternatív news és RSS felületeken keresztül: http://gmane.org/
[MAILARCHIVE]	A Mail Archive archiváló szolgáltatást biztosít levelezőlisták részére: http://mail-archive.com/
[UCGMANE]	A projekt levelező listái a Gmane-en: http://dir.gmane.org/search.php?match=gmane.comp.gnome.apps.ulc
[UCMAILARCHIVE]	A projekt levelező listái a Mail Archive szolgáltatáson: http://www.mail-archive.com/index.php?hunt=ulc-
[FRESHMEAT]	A freshmeat, a világ legnagyobb Unix szoftver leltára: http://freshmeat.net
[UCFRESHMEAT]	A projekt honlap a freshmeat-en: http://freshmeat.net/projects/ulc/
[UCARTWORK]	A projekt logó grafikák, amiket a közreműködők készítettek: http://ultimatecommander.org/Artwork
[VNC2SWF]	A vnc2swf program, amellyel flash videók készíthetők: http://www.unixuser.org/~euske/vnc2swf/
[VNC2SWFHOWTO]	A hogyan dokumentum, amit amiben leírtam, hogy hogyan állítható be a GNOME környezet optimális módon a demó videók elkészítéséhez: http://laci.monda.hu/blog/2006/03/01/making-flash-screencasts-under-gnome/
[UCSCREENCASTS]	A projekt demó videók: http://ultimatecommander.org/Demo_Section
[DOKUWIKI]	A DokuWiki tartalomkezelő rendszer, amely kezdetben a projekt honlap alapját szolgáltatta: http://wiki.splitbrain.org/wiki:dokuwiki
[NOVELLPROJECTS]	A Novell néhány projekt honlapja, amelyek nyilvánvalóvá tették számomra, hogy mennyire jól alkalmazható a MediaWiki tartalomkezelő rendszer erre a célra: http://beagle-project.org , http://mono-project.com , http://tango-project.org , http://betterdesktop.org , http://hula-project.org

[BEAGLE]	A Beagle kereső démon honlapja: http://beaglewiki.org
[SPOTLIGHT]	A SpotLight keresőrendszer honlapja: http://www.apple.com/macosx/features/spotlight
[WINFS]	A WinFS fájlrendszer honlapja: http://msdn.microsoft.com/data/winfs/
[SKYFS]	A SkyFS fájlrendszer a Wikipédián: http://en.wikipedia.org/wiki/Skyfs
[GOOGLEDSEARCH]	A Google Desktop Search honlapja: http://desktop.google.com/
[KAT]	A Kat keresőrendszer honlapja: http://kat.mandriva.com/
[STORAGE]	A GNOME Storage keresőrendszer honlapja: http://www.gnome.org/~seth/storage/
[LUCENE]	A Lucene.Net keresőmotor honlapja: http://www.dotlucene.net/
[GNOME]	A GNOME projekt honlapja: http://www.gnome.org
[KDE]	A KDE projekt honlapja: http://www.kde.org
[MWFRA TMANSKIN]	A MediaWiki FratMan bőr, amelyet a projekt honlap saját stílusához használtam fel: http://meta.wikimedia.org/wiki/Gallery_of_user_styles#FratMan
[UCCONTACT]	A projekt kapcsolatfelvételi lapja: http://ultimatecommander.org/Contact_Us
[UCBLOG]	A projekt fejlesztői webnaplója: http://laci.monda.hu/blog/category/hacking/ultimate-commander/

Nyilatkozat

Alulírott szakos hallgató, kijelentem, hogy a dolgozatomat a Szegedi Tudományegyetem, Informatikai Tanszékcsoport, Tanszékén készítettem, diploma megszerzése érdekében.

Kijelentem, hogy a dolgozatot más szakon korábban nem védtem meg, saját munkám eredménye, és csak a hivatkozott forrásokat (szakirodalom, eszközök, stb.) használtam fel.

Tudomásul veszem, hogy szakdolgozatomat a Szegedi Tudományegyetem könyvtárában, a kölcsönözhető könyvek között helyezik el.

Szeged, 2005. május 12.

.....
Monda László

Köszönetnyilvánítás

Szeretném megköszönni Bilicki Vilmosnak az összes konstruktív kritikát, segítséget és bátorítást amit adott és hogy elvállalta, hogy a témavezetőm legyen.

Ezen kívül szeretnék köszönetet mondani minden fejlesztőnek, aki segítséget adott az IRC csatornákon és a levelező listákon a fejlesztés folyamán, különösen az alábbiaknak:

- Bauer Mirco <<http://www.meebey.net>> a C# nyelvvel kapcsolatos segítségéért.
- Paco Martinez <<http://www.mfconsulting.com/blog/>> a Windows specifikus problémákkal kapcsolatos segítségéért.
- Jonathan Pryor <<http://www.jpri.com/>> a Mono Unix-specifikus szolgáltatásaival kapcsolatos segítségéért.
- Köszönöm a projekt logó elkészítésében nyújtott segítségüket: Herbert Williams [espamaccount at yahoo dot com], Pete [pete at webring dot cc], Andrei Popa [andypopa at gmail dot com].

és mindenki másnak akinek a nevére nem emlékszem.