Zaurus mit Linux-Desktop-Anbindung

Ethernet over USB

Die Anbindung des Zaurus mittels USB over Ethernet wird in einem SuSE Artikel (Supportdatenbank Stichwort Zaurus) erklärt.

Gründsätzliches:

- --nur möglich mit Kernel ab 2.4.19 (ältere erfordern einen Patch)
- --Laden des Treibers: modprobe usbdnet
- (file:/lib/modules/2.4.19-4GB/kernel/drivers/usb/usbdnet.o)
- --Zaurus über Dockingstation an PC anschließen
- --IP-Adresse angeben: ifconfig usb0 192.168.10.1 netmask 255.255.255.0 up

QtDesktop

Qtdesktop von der Entwickler CD installieren (als root entsprechendes Startscript ausführen)

- --auf dem KDE Desktop eine Verknüpfung herstellen
  kdesu -n '/opt/ZaurusCD/zcd_qtopiadesktop'
  (Anmerkung: bei mir funktioniert Qtdesktop leider nur dann, wenn ich es als root starte
- --in den „Settings“ des Qtdesktop (Menü File) IP-Adresse des Zaurus einstellen, z.B. 192.168.10.99

Programme installieren

Einige Links zu Internetseiten, die ipk – Pakete anbieten:

http://www.crosswire.org/feeds/arm/zaurus/
http://opie.handhelds.org/feed/zaurus/
http://humphrey.applitec.com/zaurus/feed/
http://rio.vg/zaurus/files/ipks/
http://zaurus-ja.sourceforge.jp/feed/

Anmerkung: zuerst würde ich aus Administrationsgründen

servermanager.ipk (telnet auf dem Zaurus aktivieren)

terminal.ipk

filemanager.ipk installieren.

Zusätzlich sollte man

troll_ftpd.ipk installieren, damit der Datentransfer vom PC aus gesteuert werden kann.

Damit dies gelingt, muss nach der installation des ftpd.ipk auf dem Zaurus

/etc/inetd.conf angepasst werden !!

#Ausschnitt aus der /etc/inetd.conf

```
ftp             stream          tcp             nowait          root            /usr/sbin/ftpd
telnet          stream  tcp     nowait  root    /usr/sbin/in.telnetd
```

Adressbuch aus EVOLUTION zum Zaurus exportieren

Das Adressbuch für Evolution befindet sich unter
~/evolution/local/Contacts/adressbook.db

Das Adressbuch des Standardusers root befindet sich auf dem Zaurus unter
/home/root/Applications/adressbook/addressbook.xml

Vorgehensweise:

- Umwandlung des Evolution-Adressbuches in ein Zaurus-Adressbuch (adressbook.xml) mittels Perl-Script (s. u.)
- kopieren des konvertierten Adressbuches vom PC zum Zaurus
- !!!!! Softreset des Zaurus !!!! (Batteriefach entriegeln und nach kurzer Wartezeit wieder verriegeln) Dieser Schritt ist wichtig, da sonst das neue Adressbuch durch das „alte“ im ROM des Zaurus befindliche überschrieben wird !!!!!

Weitere nützliche (?) Links

http://www.killefiz.de/zaurus/
http://www.zaurus.com/dev/board/
http://www.zauruszone.com/
http://www.myzaurus.com/
http://zaurus.loveslinux.com/
http://sdb.suse.de/de/sdb/html/jreuter_zaurus_usb.html
http://www.zauruszone.com/howtos/servers_howto.shtml
http://humphrey.applitec.com/zaurus/
http://www.openzaurus.org/oz_website/content/overview.php
http://www.lisa.de/

Das PERL-Script :

Anmerkung:

Auf dem PC müssen die im Abschnitt „use“ eingetragenen Perl-Module installiert sein !!!
Erstellen Sie zusätzlich ein Verzeichnis ~/zaurus da das Perl-Script hier das neue addrbook.xml speichert.

```
#!/usr/bin/perl -w
#
# Evolution Addressbook 2 Zaurus
#

use strict;
use CGI qw( unescapeHTML );
use Data::Dumper qw( DumperX ); # for debugging
use DB_File;
use File::Copy;
use XML::Simple;

my $home = "$ENV{HOME}";

##############################################################################
# routines
##############################################################################

sub dump_field {
    my ( $hash, $field, $num ) = @_;

    my @data;

    # data
    if ( exists $hash->{$field} ) {
        # remove HTML
        my $data = unescapeHTML( $hash->{$field} );

        # split it up
        @data = split /;/, $data, $num;

        # count it up
        $num -= scalar @data;
    }

    # empty the rest
    while ( $num > 0 ) {
        push( @data, "" );
        $num--;
    }

    # cleanup
    foreach ( @data ) {
        # trim leading and trailing spaces
        s/^\s+//;
        s/\s+$//;
    }

    return @data;
}

##############################################################################
# address book
##############################################################################

my $evodbm = "$home/evolution/local/Contacts/addressbook.db";
my $tmpdbm = "evolution.db";
my $xmlfile = "$home/zaurus/addressbook.xml";

my %abook_db2xml = (
    # home contact
    'Title' => 'TITLE',
    'Nickname' => 'NICKNAME',
    'DefaultEmail' => 'EMAIL;INTERNET',
    'Emails' => 'EMAIL;INTERNET',
    'HomePhone' => 'TEL;HOME',
```

```
    'HomeWebPage' => 'URL',
# 'Gender' => '',
    'Spouse' => 'X-EVOLUTION-SPOUSE',
    'Children' => 'X-EVOLUTION-CHILDREN',
    'Birthday' => 'BDAY',
    'Anniversary' => 'X-EVOLUTION-ANNIVERSAY',
    'Notes' => 'NOTE;QUOTED-PRINTABLE',

    # business contact
    'BusinessPhone' => 'TEL;WORK;VOICE',
    'BusinessFax' => 'TEL;WORK;FAX',
    'BusinessMobile' => 'TEL;CELL',
    'BusinessPager' => 'TEL;PAGER',
    'BusinessWebPage' => 'URL',
    'JobTitle' => 'ROLE',
    'Profession' => 'ROLE',

    # evolution
    'FileAs' => 'X-EVOLUTION-FILE-AS',
# 'Categories' => '',
);

sub merge_addr {
    my ( $pobox, $addr1, $addr2 ) = @_;
    my $addr = "";

    $addr .= $pobox if length $pobox;
    $addr .= " $addr1" if length $addr1;
    $addr .= " $addr2" if length $addr2;

    return $addr;
}

sub dump_addrbook {
    my $ncontacts = 0;
    my $ngroups = 0;
    my %DBIN;

    print "Evolution AddressBook dump to Zaurus: ";

    # temporary copy
    chdir( "$home/tmp" );
    copy( $evodbm, $tmpdbm );

    # open db file for reading
    return 0 unless dbmopen( %DBIN, $tmpdbm, undef );

    # open xml file for writing
    return 0 unless open( XMLOUT, ">$xmlfile" );

    # header
    print XMLOUT "<?xml version=\"1.0\" encoding=\"UTF-8\"?><!DOCTYPE
Addressbook >\n";

    # output hash
    my $xmldata = {};
    $xmldata->{Contacts}{Contact} = [];
    $xmldata->{Groups}{Group} = [];

    # loop though each entry
    foreach my $key ( keys(%DBIN) ) {
        my $line = $DBIN{$key};

        next if $line !~ m/.*:.*/;

        my @list = split /\r\n/, $line;

        my %dbhash = ();
        my $contact = {};
```

```
        # get fields
        foreach my $item ( @list ) {
            if ( $item =~ m/^$/ ) {
                next;
            };
            if ( $item =~ m/^\x00$/ ) {
                next;
            };
            chomp $item;

            my ($field, $data) = split /:/, $item, 2;

            if ($field =~ m/TEL.*WORK.*FAX.*/){
                $field = "TEL;WORK;FAX";
            } elsif ($field =~ m/TEL.*WORK.*/){
                $field = "TEL;WORK;VOICE";
            } elsif ($field =~ m/TEL.*VOICE.*/) {
                $field = "TEL;HOME"
            } elsif ($field =~ m/TEL.*HOME.*/) {
                $field = "TEL;HOME"
            } elsif ($field =~ m/TEL.*CELL.*/) {
                $field = "TEL;CELL"
            } elsif ($field =~ m/TEL.*PAGER.*/) {
                $field = "TEL;PAGER"
            } elsif ($field =~ m/ADR.*WORK/) {
                $field = "ADR;WORK"
            } elsif ($field =~ m/ADR.*/) {
                $field = "ADR;HOME"
            };

            #print " data = $data -- field = $field\n";
            $dbhash{$field} = $data;
        };

        # read out data
        my ($lastname, $firstname, $middlename ) = dump_field( \%dbhash,
'N', 3 );
        $contact->{LastName} = $lastname;
        $contact->{FirstName} = $firstname;
        $contact->{MiddleName} = $middlename;

        # home address
        my ($pobox, $address2, $address1, $city, $state, $zip, $country)
= dump_field( \%dbhash, 'ADR;HOME', 7 );
        $contact->{HomeStreet} = merge_addr( $pobox, $address1,
$address2 );
# $contact->{HomeAddress} = $contact->{HomeStreet};
        $contact->{HomeCity} = $city;
        $contact->{HomeState} = $state;
        $contact->{HomeZip} = $zip;
        $contact->{HomeCountry} = $country;

        # business address
        ($pobox, $address2, $address1, $city, $state, $zip, $country) =
dump_field( \%dbhash, 'ADR;WORK', 7 );
        $contact->{BusinessStreet} = merge_addr( $pobox, $address1,
$address2);
# $contact->{BusinessAddress} = $contact->{BusinessStreet};
        $contact->{BusinessCity} = $city;
        $contact->{BusinessState} = $state;
        $contact->{BusinessZip} = $zip;
        $contact->{BusinessCountry} = $country;

        # business info
        my ( $company, $dept ) = dump_field( \%dbhash, 'ORG', 2 );
        $contact->{Company} = $company;
        $contact->{Department} = $dept;

        # copy data all other data
        foreach my $attr ( keys( %abook_db2xml ) ) {
            my ($val) = dump_field( \%dbhash, $abook_db2xml{$attr}, 1 );
            $contact->{$attr} = $val;
        }

        # cleanup (remove leading & trailing blanks)
```

```
        # is it a group (stored as XML)?
        if ( $contact->{Emails} =~ /^\<\?xml/ ) {
            my $xml = $contact->{Emails};

            # parse XML
            my $egroup = XMLin( $xml );

            # split out email addresses for group
            my $group = {};

            # add it to hash
            if ( scalar keys %$group ) {
                push( @{$xmldata->{Groups}{Group}}, $group );
                $ngroups++;
            }
        } else {
            # add it to hash
            push( @{$xmldata->{Contacts}{Contact}}, $contact );
            $ncontacts++;
        }
    }

    # dump XML
    print XMLOUT XMLout( $xmldata, rootname=>'AddressBook' );

    # close up
    close( XMLOUT );
    unlink( $tmpdbm );

    # success
    print "$ncontacts Contacts, $ngroups Groups dumped\n";
    return $ncontacts + $ngroups;
}

################################################################################
# calendar
################################################################################

################################################################################
# task list
################################################################################

################################################################################
# main
################################################################################

sub main {
    dump_addrbook();
}

&main();

0;
```