

Version 0.1b

**Nils Faerber (nils.faerber@kernelconcepts.de)**

**© 2004 by Nils Faerber, GNU Documentation License**

**Das ist die Legalnotice: Er hörte leise Schritte hinter sich. Das bedeutete nichts Gutes. Wer würde ihm schon folgen, spät in der Nacht und dazu noch in dieser engen Gasse mitten im übel beleumundeten Hafenviertel?**

*Der Vortrag soll die Motivation und aktuelle Entwicklung von GPE, dem G-PDA Environment beleuchten. GPE ist der konsequente Schritt zum dritten Linux Zeitalter - die Verwendung offener Standards zur Entwicklung eines Desktop- und Applikationsstandards für embedded Systeme mit grafischer Benutzeroberfläche.*

## **Das dritte Linux-Zeitalter**

## Das erste Computer-Zeitalter

Zur Zeit erleben wir den Übergang zu dem, was man als drittes Computer-Zeitalter bezeichnen könnte.

Als vor gut 25 bis 30 Jahren Computer aus den Forschungslabors in die reale Welt traten und reale Aufgaben übernahmen, bestanden Sie noch aus riesigen grauen Kisten, die von einer kleinen eingeschworenen Gruppe von ebenfalls grau bekleideten Spezialisten umsorgt und gepflegt wurden. Dem normalen Menschen waren Sie ebensowenig zugänglich wie verständlich. Erste für den Laien sichtbaren Inkarnationen dieses beginnenden ersten Computer-Zeitalters waren Berge von Computerausdrucken und Gerüchte um die schier unfaßbaren Möglichkeiten dieser neuen Maschinen.

Die Hersteller dieser Maschinen waren zu diesem Zeitpunkt noch damit beschäftigt, ihren neuen Wunderwerken ersteinmal eine Existenzberechtigung zu verschaffen – denn eigentlich wurde jetzt vieles komplizierter als einfacher. An Standards oder gar Interoperabilität zwischen den herstellereigenen Systemen mochte keiner denken.

Dem Gros der normalen Menschen war dies auch noch reichlich egal, denn wer wollte schon freiwillig etwas mit diesen störrischen Maschinen zu tun haben. Maschinen deren primäre Aufgabe es war, eine kleine Menge an Aufgaben nach einem festen Schema zu lösen. Und außerdem, wer hat denn überhaupt schon damit zu tun? Angestellte in Behörden, Universitäten und größeren Konzernen vielleicht.

## Das zweite Computer-Zeitalter

Dies änderte sich jedoch bald. Die fortschreitende Entwicklung machte es möglich, daß Computer kleiner gebaut und universeller eingesetzt werden konnten. Zu den Großrechnern gesellten sich Terminals, die den Benutzern direkt vor die Nase gestellt werden konnten. Nicht mehr nur eine kleine eingeschworene Gemeinde von Graukitteln hatte nun direkten Kontakt mit dem Computer, sondern auch der Anwender. Genauer gesagt, dies war der Zeitpunkt zu dem der *Anwender* überhaupt erst geboren wurde.

Doch nicht nur die Großrechner bekamen dezentrale Ein-/Ausgabeeinheiten, sondern es wurde auch kleine Computer entwickelt, die Micro-Computer und IBM entwickelte den Personal-Computer (1981).

Nun war es also soweit. Der Computer eroberte die Schreibtische und bald fanden sich in fast jedem Büro der unvermeidliche Computer als universelles Gerät für viele Aufgaben des täglichen Geschäftslebens.

Heute ist der Computer aus dem Alltag nicht mehr hinwegzudenken und vom Handwerker bis zum Manager, früher oder später bekommen wir alle beruflich damit zu tun. Und je weiter die Durchdringung fortschreitet, desto größer wird die Notwendigkeit, daß alle diese Rechnersysteme miteinander interagieren können müssen.

## Das dritte Computer-Zeitalter

Noch blieb man von den offensichtlichen Computern im privaten Alltag verschont.

Noch.

Denn es beginnt das dritte Computer-Zeitalter. Der Computer dringt in unseren ganz persönlichen Alltag immer weiter vor. Es begann mit den immer komplizierter werden Video-Recordern, die auf einmal eine ganze Generation von Menschen nicht mehr bedienen konnte. Es wurde Konzepte vorausgesetzt die man nur verstehen konnte, wenn man ansatzweise die Funktionsweise von Computern verstanden hat. Video-Recorder sind überhaupt ein gutes Beispiel, denn diese waren auch die ersten Alltagsgeräte, die die Benutzer am Bildschirm bei der Bedienung führten.

Die bereits seit einigen Jahren im verborgenen arbeitenden Computer treten aus Ihrem Schattendasein hervor. Auf einmal wird jeder von uns zum Anwender, ob man möchte oder nicht. Von der Mikrowelle bis zur Waschmaschine, vom Mobiltelefon bis zum MP3-Player, vom Parkscheinautomat bis zum Auto-Navigationssystem, Computer sind allgegenwärtig geworden.

Auf den Höhepunkt des dritten Zeitalters wird jetzt und heute langsam hingearbeitet, indem die vorhandenen Computersysteme miteinander verbunden und vernetzt werden – die Waschmaschine der Mikrowelle mitteilt, daß das Essen jetzt angewärmt werden kann weil die Wäsche fertig ist, der MP3-Player eine entsprechende Meldung einspielt und im Display des Navigationssystems angezeigt wird, wann voraussichtlich zu Hause das Essen bereit steht.

Was hat das alles mit Linux zu tun?

## Die Evolution von Linux

Als Linus Torvalds im August 1991 seine Arbeit an Linux in der News-Group comp.os.minix bekannt gab<sup>1</sup>, war Linux nicht mehr als ein Hobby- und Hacker-Projekt. Die Idee eines freien Unix-ähnlichen Betriebssystems fand vor allem in Universitäten Anklang und so erfreute sich Linus und Linux bald einer aktiven Entwicklergemeinde. Die gerade aufkommende Verbreitung des Internet kam extrem hilfreich hinzu.

Die Faktoren Universitäten, freie Software und Entwicklung über das Internet führten zum ersten Linux-Zeitalter.

## Das erste Linux-Zeitalter

Universitäten haben seit jeher ein Problem: Sie haben nur geringe finanzielle Mittel zur Verfügung. Im aufkommenden Internet-Boom kam ihnen daher die Idee eines freien und damit weitgehend kostenlosen Betriebssystems sehr entgegen. Positiv hinzu kam, daß die meisten Linux-Entwickler ohnehin an den Universitäten zu finden waren, man also günstig an die Experten herankam.

---

1 <http://www.li.org/linuxhistory.php>

Dadurch, daß die Entwicklung von Linux über das Internet stattfand, wurde auch Linux selbst von Anfang an auf die Verwendbarkeit im und mit dem Internet getrimmt.

Mit weiterer freier Software, allen voran dem Apache-Webserver, wurde Linux zu einem der weitverbreitetsten Betriebssysteme für Internet-Server. Und wie beim ersten Computer-Zeitalter fand der erste Schritt eher im Verborgenen und hinter verschlossenen Türen statt – nur das die Leute mit den grauen Kitteln nun Computerfreaks waren, aber von den normalen Menschen ebensowenig verstanden wurden, wie die Graukittel vor 30 Jahren.

## **Das zweite Linux-Zeitalter**

Nachdem sich Linux als stabiles und offenes System für Server etabliert hatte, war der konsequente nächste Schritt, Linux auch auf den Arbeitsplatz-PC zu bringen. Genau wie der Schritt vom Großrechner zum Arbeitsplatz-PC, mußte die abstrakte Unix-Welt dem weniger versierten Benutzer begreiflich und benutzbar gemacht werden.

Neue Projekte der freien Software wurden ins Leben gerufen und schufen, neben vielen weiteren, KDE<sup>2</sup> und GNOME<sup>3</sup>. Beide haben den Anspruch, eine komfortable und für den normalen Anwender einfach zu verwendende grafische Benutzeroberfläche zu schaffen. Dazu gehört nicht nur ein schönes Aussehen, sondern vor allem Applikationen, die Hand in Hand miteinander arbeiten, Daten mit anderen Anwendungen nahtlos austauschen können und dem Benutzer Arbeit abnehmen.

Dank dieser Projekte hat Linux sein zweites Zeitalter erreicht und den Sprung von den Servern auf den Schreibtisch geschafft. Selbst Unix unerfahrene Benutzer können heute in vielen Bereichen mit Linux ebenso effektiv und effizient arbeiten, wie mit anderen Betriebssystemen.

## **Das dritte Linux-Zeitalter**

Bisher hat die Linux-Entwicklung der Entwicklung der Computer hinterher gehinkt – kein Wunder, denn seine Entwicklung hat viel später begonnen. Heute befinden wir uns allerdings an einer Art Break-Even-Point, die Linux- und Computer-Entwicklung liegen etwa gleichauf.

Die offensichtlichen Computer dringen mehr und mehr in unser Leben ein. Geräte des täglichen Lebens und Alltags werden mit immer mehr Eigenintelligenz ausgestattet, mit komplexen Computersystemen. Und in naher Zukunft werden viele davon miteinander verbunden und vernetzt sein.

Die ersten einfachen Computersteuerungen in Alltagsgeräten waren nicht viel mehr als einfache Zeitschaltuhren. Doch die Aufgaben werden stetig komplexer und die dafür benötigte Software immer komplizierter. Noch findet sich in der Mehrzahl dieser Geräte Software, die von den Herstellern vollständig selbst entwickelt wurde und als ihr „Intellectual Property“ (IP), als Teil ihrer

---

2 <http://www.kde.org/>

3 <http://www.gnome.org/>

Kernkompetenz, angesehen wird. Doch die Hersteller geraten zunehmend unter Druck. Je komplexer die Anforderungen werden, desto teurer wird die dafür notwendige Softwareentwicklung. Auf der anderen Seite drängt jedoch der Markt auf immer günstigere aber dennoch leistungsfähigere Produkte. Dies führt bereits heute zu teilweise desaströsen Zuständen, bei denen die Software immer schneller entwickelt und immer fehleranfälliger wird.

Ein ebenso prominentes wie beklagenswertes Beispiel hierfür sind Mobiltelefone. Der Mobiltelefonmarkt ist geprägt von extremem Preisdruck und extremer Kurzlebigkeit. Die Qualität der Software läßt immer weiter nach und die Liste der bekannten Fehler wächst, ohne das die Hersteller diese noch nennenswert verkürzen würden – bevor man lange einen Fehler sucht, ihn behebt und wieder Tests durchführen muß, ist bereits die neue Gerätegeneration am Start und die Softwareentwickler müssen neue Software entwickeln anstatt an alter weiter zu arbeiten.

Linux kann hier eine Alternative bieten!

Einige Projekte haben bereits hinreichend bewiesen, daß Linux auch in Geräte integriert (embedded) werden kann. Linux bietet für viele Probleme der Hersteller eine Lösung: Es ist portabel, es ist stabil, es ist erprobt, es bietet eine Vielzahl an Kommunikationsschnittstellen mit anderen Systemen, es orientiert sich an offenen Standards und es wird konstant weiterentwickelt.

Prominente Beispiele, um nur zwei zu nennen, sind die Linux Portierung von HP ehemals Compaq auf die Compaq/HP iPAQ PDAs sowie die heimliche Verwendung von Linux in einigen Netzwerk-Produkten. Gerade die Hersteller von Netzwerkkomponenten haben gesehen, daß der Softwareentwicklungsaufwand für eigene TCP/IP Router- und Firewall-Lösungen deutlich höher wäre, als eine bestehende, freie und erwiesenermaßen stabile Lösung zu übernehmen. Leider haben einige Hersteller das Prinzip der freien Software dabei nicht verstanden oder, freundlich ausgedrückt, wohl übersehen.

Mit der Linux Portierung auf PDAs der PocketPC-Klasse hat HP/Compaq eine Vorreiterrolle übernommen. Zum ersten mal war es damit der Community möglich, die Hürden von nicht dokumentierter Hardware zu überwinden und die Klasse der embedded-Geräte zu erobern.

Hier und heute beginnt es also, das dritte Linux-Zeitalter – Linux als standard Betriebssystem für embedded Geräte! Doch es ist gleichzeitig eine gewaltige Herausforderung. Die Notwendigkeiten für eine vereinheitlichte Softwareplattform für embedded Geräte wurde nicht erst jetzt erkannt. Andere Softwarehersteller versuchen mit all ihrer Marktmacht und in anderen Bereichen bereits marktbeherrschenden Stellung, in diesen Markt einzudringen und Fuß zu fassen. An für sich ist dagegen nichts einzuwenden, wenn sich nicht gezeigt hätte, daß beim Geld leider jede Freundschaft und Fairness endet. Einige Konkurrenten versuchen durch proprietäre „Erweiterungen“ Exklusivität in ihre Produkte zu bringen. Dies geht sogar so weit, daß bereits anerkannte Standards so „erweitert“ werden, daß der volle Funktionsumfang nur noch einzig und allein mit den Produkten eines bestimmten Herstellers erreicht werden kann.

Noch ist nichts entschieden. Noch können wir entscheiden, ob wir an offenen Standards mitarbeiten oder mit angeblich offenen Standards arbeiten wollen.

## GPE – Das G-PDA Environment

So wie es KDE und Gnome geschafft haben, Linux und Unix für den Endanwender benutzbar zu machen, so braucht es auch für embedded Geräte ebensolche Anstrengungen und Entwicklungen. Da jedoch auch die Anforderungen an embedded Software anders sind, als die an normale Desktop Software, braucht es dazu eine eigenständige Entwicklung.

Das GPE-Projekt<sup>4</sup> hat sich dies zum Ziel gesetzt. In Zusammenarbeit mit und in Anlehnung an bekannte Desktop-Systeme, ist GPE der konsequente nächste Schritt zur Verbreitung von Linux auf embedded Systemen mit grafischer Benutzeroberfläche und die Möglichkeit, einen wirklich offenen Standard dafür zu etablieren.

### GPE - Motivation

GPE wurde inspiriert durch die Linux Portierung auf die Compaq/HP iPAQ PDA Geräte. Anfänglich gab es zwar eine vollständige Linux Portierung mit lauffähigem Kernel, Framebuffer-Treiber und Unterstützung der grundlegenden Hardware wie Touchscreen und Tasten, doch es gab noch keine darauf aufbauende Software, die das Gerät sinnvoll verwendbar gemacht hätte.

Als grafische Oberfläche kam auf dem lediglich 240x320 Punkte großen Bildschirm eine angepaßte Version von XFree86 zum Einsatz. Schnell zeigten sich Unzulänglichkeiten: Der Framebuffer des iPAQ verwendete eine Auflösung von 320x240 Punkten, also „landscape“, die Hardware war jedoch für den 240x320 „portrait“ Modus ausgelegt – der Bildschirminhalt erschien also um 90 Grad gedreht. Normale XFree/X11 Applikationen konnten auch nicht sinnvoll verwendet werden, da die Fenster viel zu groß waren – ein heutiger Linux Desktop hat eine Auflösung von mindestens 1024x768 Punkten, der PDA hat gerade einmal ein neuntel dieser Auflösung! Ein Desktop besitzt eine Maus mit mindestens zwei, meist drei Tasten – der PDA hat nur einen Touchscreen mit absoluter Positionierung und nur einer „Taste“. Das ganze Fenster-Paradigma des grafischen Desktops paßt einfach nicht auf einen PDA – bei einer so geringen Auflösung, kann nicht sinnvoll mit gegeneinander verschiebbaren Fenstern gearbeitet werden.

Von diesen „Unzulänglichkeiten“ abgesehen, bieten embedded System noch eine Reihe weiterer Herausforderungen. Der Arbeitsspeicher ist begrenzt, damals 32MB RAM, heute 64MB und teilweise 128MB RAM. Der permanente Speicher, vergleichbar der Festplatte, ist begrenzt, damals 16MB Flash, heute zumeist 32MB oder sogar bis zu 128MB Flash – nicht zu vergleichen mit den Gigabytes der Festplatten. Hinzu kommt das Problem, daß Flash-Speicher nicht unbegrenzt oft beschrieben bzw. gelöscht werden kann.

Es mußte also ein System entwickelt werden, welches zum einen diesen Herausforderungen gerecht wird, sich aber dennoch möglichst nahtlos in das bestehende Linux Desktop Konzept einfügt. Für Entwickler sollte es möglichst einfach sein, bestehende Linux/Unix Applikationen anzupassen und für die Benutzer sollte es ein einfach zu bedienendes, intuitives aber dennoch leistungsfähiges System sein. Es war von Anfang an erklärtes Ziel des GPE

---

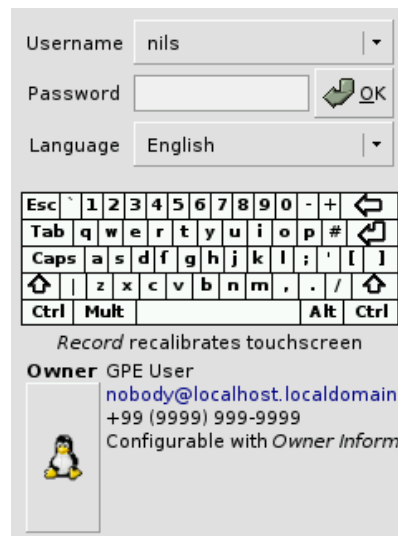
4 <http://gpe.linuxtogo.org/>

Projektes, sich an bestehenden Standards zu orientieren oder sie sogar, wo dies möglich ist, vollständig zu übernehmen.

Erst durch die Verwendung von offenen Standards kann dem dritten Computer- und Linux-Zeitalter Rechnung getragen werden, dem Zusammenwachsen aller Bereiche, in denen heute Computer eingesetzt werden.

## GPE - Hauptmerkmale

GPE verfolgt den von Compaq/HP eingeschlagenen Weg weiter und basiert auf X11/XFree86. Als X11 Toolkit wird GTK+ eingesetzt. Ein spezieller Window-Manager<sup>56</sup> sorgt für eine optimale Darstellung und verwendet Spezifikationen, die durch das freedesktop.org<sup>7</sup> Projekt entwickelt wurden und werden. Teil des Window-Manager Projektes ist ebenfalls eine Desktop-Komponente zur Auswahl und Start von Applikationen. Applikationen registrieren sich dort entsprechend den Desktop-Spezifikationen<sup>8</sup> von freedesktop.org. Konfigurationen bzgl. der grafischen Oberfläche werden durch XSettings<sup>9</sup> im X-Server allen Applikationen zur Verfügung gestellt und durch einen gconf-Konfigurationsserver gemeinsam für alle verwaltet und gespeichert. Zur Datenspeicherung wird der weit verbreitete SQL-Standard in Zusammenhang mit einer kleinen und schlanken SQL-Datenbank<sup>10</sup> verwendet. Nachrichtensbasierte Interprozeßkommunikation erfolgt über DBUS<sup>11</sup>. Hinzu kommen spezialisierte GPE-spezifische Funktionen, die in eigenen Bibliotheken gekapselt sind.



5 <http://www.freedesktop.org/Standards/wm-spec>

6 <http://www.freedesktop.org/Standards/startup-notification-spec>

7 <http://www.freedesktop.org/>

8 <http://www.freedesktop.org/Standards/desktop-entry-spec>

9 <http://www.freedesktop.org/Standards/xsettings-spec>

10 <http://www.hwaci.com/sw/sqlite/>

11 <http://www.freedesktop.org/Software/dbus>

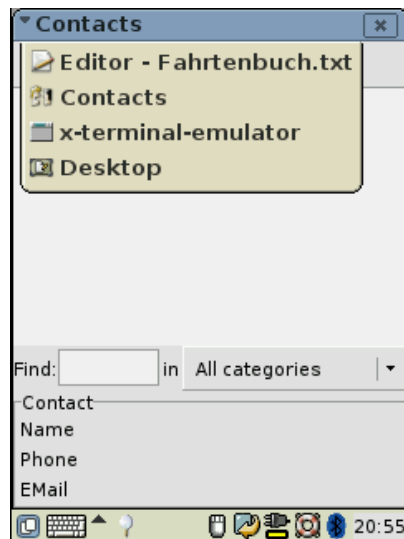
## Window-System, X11

X11 ist ein seit fast 20 Jahren gereiftes System für grafische Oberflächen. Die XFree86 Implementation ist zudem hinreichend ressourcenschonend, sodaß auch nur mit 32MB RAM ausgestattete Geräte ohne Probleme verwendet werden können. Die spezielle „kdrive“ X-Server Variante ist zudem für embedded-Geräte in punkto Ressourcenbedarf optimiert worden. In den vergangenen Monaten haben außerdem einige Entwickler den Quellcode von Altlasten befreit und so den Server als auch die XFree Bibliotheken erheblich verkleinert.

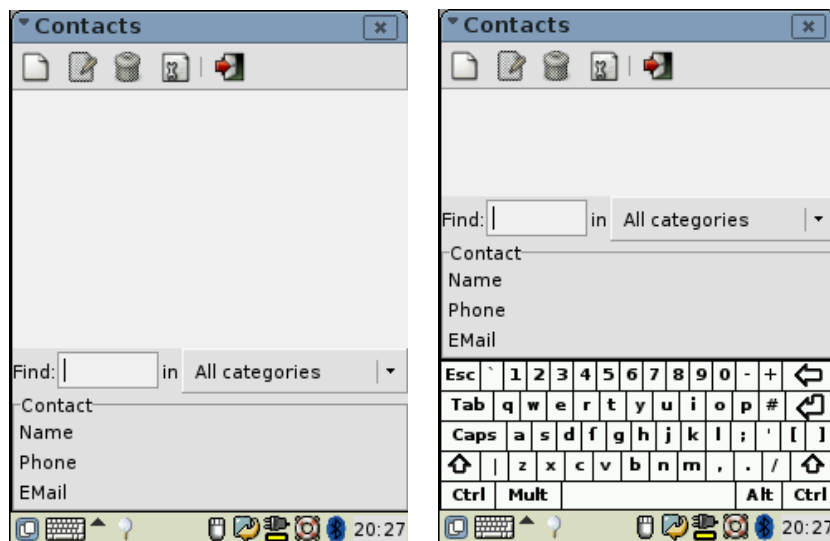
Durch die Verwendung von X11 ist GPE netzwerktransparent, d.h. GPE-Applikationen können, wie vom Desktop her gewohnt, auf anderen Displays gestartet werden oder es können Applikationen von entfernten Rechnern auf einem GPE-Gerät dargestellt werden. Dies eröffnet völlig neue Arten der Anwendung von PDAs!

Mit Hilfe der neuen XRandR (Rotate and Resize) Erweiterung des X-Servers ist es möglich, und auch von GPE unterstützt, daß Applikationen zu ihrer Laufzeit zwischen X11 Displays verschoben, also migriert, werden können! Eine mögliche zukünftige Anwendung könnte dann sein, daß PDAs die mit wireless Netzwerktechnologien ausgestattet sind, alle Fenster von gerade laufenden Applikationen auf ein Desktop-PC-Display migrieren, wenn man in dessen Nähe kommt. Die Applikationen werden über diese Migration ebenfalls informiert und können dann ihr Aussehen an die neuen Gegebenheiten optimal anpassen, bspw. mehr Buttons darstellen, größere Übersichten anzeigen, etc. Außerdem kann der Benutzer dann wesentlich komfortabler mit den Eingabemethoden des Desktop-PCs die PDA Applikationen bedienen und mit den Daten des PDA arbeiten. Es stellt sich dann tatsächlich die Frage, ob eine Synchronisation der Daten zwischen Desktop und PDA unabdingbar ist. Verläßt der Benutzer anschließend wieder die Umgebung des Desktop-PCs, so könnten automatisch alle Applikationsfenster zurück auf den PDA geholt werden.

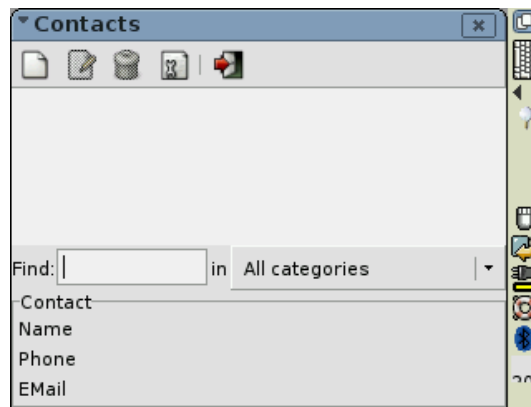
Das Window-Management übernimmt der speziell entwickelte Window-Manager „Matchbox“. Matchbox sorgt unter anderem dafür, daß Toplevel-Fenster auf die maximal darstellbare Fläche vergrößert werden. Spezielle Applikationen, wie bspw. eine virtuelle Bildschirm-Tastatur oder Handschrifterkennung, können ein- und ausgeblendet werden, wodurch automatisch die Größe der Applikationsfenster angepaßt werden. Der Window-Manager übernimmt auch die typische Fenster-Dekoration mit Rahmen und Titelzeile. Die Titelzeile dient bei Matchbox/GPE auch direkt zur Umschaltung zwischen mehreren aktiven Applikationen.



Bei der Verwaltung der Fenster verwendet Matchbox/GPE die durch Freedesktop.org festgelegten Standards, bspw. das Fenstereigenschaften durch sogenannte „Window-Hints“ gesetzt werden. Eine dieser Eigenschaften ist HINT\_TOOLBAR und zeigt an, daß es sich bei diesem Fenster um eine Toolbar-Applikation und nicht um ein normales Toplevel Window handelt. Solche Applikation würden unter GPE am unteren Bildschirmrand dargestellt und lassen sich platzsparend „einfalten“ - ein Beispiel hierfür ist die virtuelle Bildschirmstatur.



Eine zusätzliche Funktion der bereits erwähnten XRandR Erweiterung ist die Möglichkeit den Bildschirminhalt zur Laufzeit um je 90 Grad in jede beliebige Richtung rotieren zu können. Dadurch ist es möglich, zur Laufzeit von der normalen Portrait-Ansicht (hochkant) zu einer Landscape-Ansicht (Querformat) umzuschalten. Der Window-Manager greift hier ebenfalls unterstützend ein und übernimmt die Anpassung der Fenstergrößen.



## Toolkit GTK+ V2

GPE Applikationen verwenden gemeinsam das GTK+ Toolkit<sup>12</sup> in der Version zwei. Die Wahl fiel aus mehreren Gründen auf GTK+. Zum einen erlaubt GTK+ die Verwendung unter einer freizügigen Lizenz, der LGPL. Damit ist es möglich, basierend auf GTK+, Applikationen zu entwickeln, die nicht notwendigerweise automatisch unter die GPL oder LGPL fallen – im Unterschied bspw. zu Qt/embedded. Desweiteren werden GTK+ Applikationen üblicherweise in der Programmiersprache C entwickelt. C erlaubt es, im Gegensatz zu C++, etwas kompakteren und ressourcenschonenderen Code zu produzieren. Neben diesen Vorzügen bietet GTK+ in Zusammenhang mit seinen Hilfsbibliotheken GLIB, GDK, ATK (Accessibility ToolKit) und Pango Unterstützung für eine Reihe wichtiger Funktionen.

So ist es mit GTK+ im Zusammenspiel mit Pango und der XRender Extension möglich, TrueType-Schriften mit korrektem Anti-Aliasing zu versehen. Sogar Subpixel-Rendering zur deutlichen Verbesserung der Schriftdarstellung gerade auf LC-Displays, wird direkt durch das Toolkit unterstützt, ohne das die Entwickler dafür spezielle Vorkehrungen treffen müssen. Und dies gilt nicht nur für einige wenige Applikationen, sondern für alle GTK+ und damit alle GPE Applikationen! Mit Hilfe von Pango ist es zusätzlich möglich, auch andere Schriftformen als die gewohnte von links-nach-rechts Form zu unterstützen, Beispiele sind Chinesisch oder Hebräisch. Dies ist insbesondere für die zukünftige Internationalisierung von GPE besonders wichtig.

## Datenspeicherung – SQLite

Die meisten auf einem PDA zu speichernden Daten lassen strukturiert und in einzelnen Datensätzen darstellen. Als Folge liegt es nahe, diese Daten in einer strukturierten Datenbank abzulegen. Als Standard für Datenbankzugriffe hat sich in den letzten Jahren SQL etabliert und ist sowohl bei Entwicklern als auch Benutzern anerkannt und bekannt.

Das GPE-Projekt verwendet daher zur Datenspeicherung die kleine aber leistungsfähige SQL Datenbank SQLite. Zur Zeit wird SQLite von den GPE Personal Information Management (PIM) Applikationen wie Adreßbuch, Kalender und Notizbuch, sowie von einigen Konfigurationsapplikationen verwendet. Applikationen können durch die Ausführung von einfachen SQL-Anweisungen sehr komplexe Operationen auf Ihren Datenbeständen ausführen, ohne diese

---

<sup>12</sup> <http://www.gtk.org/>

komplexe Logik neu implementieren zu müssen. Für den Applikationsentwickler entfällt ebenfalls die oft mühselige und lästige Arbeit, ein spezielles und eigenes Dateiformat mit den dafür notwendigen Schreib- und Leseroutinen zu entwickeln.

## **Personal Information Management – PIM**

Die PIM-Applikationen sind zentraler Bestandteil für PDAs. Das GPE Projekt hat die typischen Vertreter ebenfalls implementiert, also Kalender, Kontakte und To-Do-Liste. Alle diese Applikationen verwenden zur Datenspeicherung die SQL Datenbank. Hiermit soll es zukünftig auch möglich sein, mehrere Datenquellen für die Applikationen transparent verwalten und kombinieren zu können, d.h. die Applikation führt nur einen SQL-Query aus, es werden aber mehrere Datenbanken gleichzeitig abgefragt. Dies wird besonders dann interessant, wenn externe Datenquellen auf zusätzlichen Speichermedien, wie SD/MCC- oder CF-Karten, dem System hinzugefügt werden. Denkbar sind natürlich auch auch Datenquellen, die über ein Netzwerk angesprochen werden.

Bei der Gestaltung der Applikationen wurde Wert darauf gelegt, daß alle Applikationen früher oder später einmal miteinander verzahnt werden sollen. So soll es möglich sein, im Kalender einen Termin mit einer Person aus der Kontaktdatenbank einzufügen oder einen Termin für einen Punkt aus der To-Do-Liste in den Kalender einzutragen. Alle PIM-Applikationen haben daher Ihre Datenbankschnittstelle in einer separaten Bibliothek gekapselt, sodaß diese von allen Applikationen gemeinsam verwendet werden können.

Leider sind noch nicht alle dieser Funktionen implementiert. Bereits realisiert ist aber die gemeinsame Verwaltung von Kategorien, denen einzelne Einträge zugeordnet werden können (wie „geschäftlich“, „privat“, etc.).

Ein wichtiger Bestandteil des PIM ist auch die Synchronisation der Datenbestände mit anderen Applikationen. GPE verfolgt hierzu den Ansatz über MultiSync<sup>13</sup>. MultiSync ist eine modulare Applikation zur Synchronisation zwischen Paaren von Datenquellen. Die jeweilige Datenquelle wird als Plugin realisiert. Zur Zeit sind Plugins für verschiedene PDAs und Mobiltelefone, sowie für Gnome-Evolution vorhanden. Das erste Synchronisationsziel für GPE ist daher die Verbindung zu Gnome-Evolution. Erste Ansätze in diese Richtung existieren bereits.

## **DBUS**

### **GPE Bibliothek(en)**

Die GPE spezifische Bibliothek libgpewidget begann als Ergänzung zu GTK+ und sorgte für die Anpassung einiger GTK+ Widgets und Funktionen an die Gegebenheiten von PDAs. So implementiert die libgpewidget einen eigenen, deutlich kleineren, Fileselektor. Zusätzlich werden einige neue Widgets zur Verfügung gestellt, die die Anwendung komfortabler gestalten sollen, so wie ein

---

<sup>13</sup> <http://multisync.sourceforge.net/>

einfaches Datums- und Uhrzeit-Widget sowie ein platzsparender Farbauswahldialog. [XXX Screenshots].

Eine weitere Funktion der libgpewidget ist das Ausblenden des bei Stiftbedienung sonst störenden und unnötigen Mauszeigers.

Von den großen Desktops ist es der Benutzer heute gewohnt, kurze Hilfen angezeigt zu bekommen, wenn man mit dem Mauszeiger über einem Element der grafischen Oberfläche stehen bleibt. Da jedoch der Mauszeiger ausgeschaltet ist und ein einfaches Zeigen ohne gleichzeitiges Auslösen eines Button-Press-Events mit einem Touchscreen nicht möglich ist, wurde für GPE eine neue Methode entwickelt. In der Panel-Leiste befindet sich das Symbol eines Rettungsrings. Aktiviert man diese Hilfe-Funktion und zeigt anschließend auf das fragliche Oberflächenelement, so wird nun die entsprechende Hilfe angezeigt.

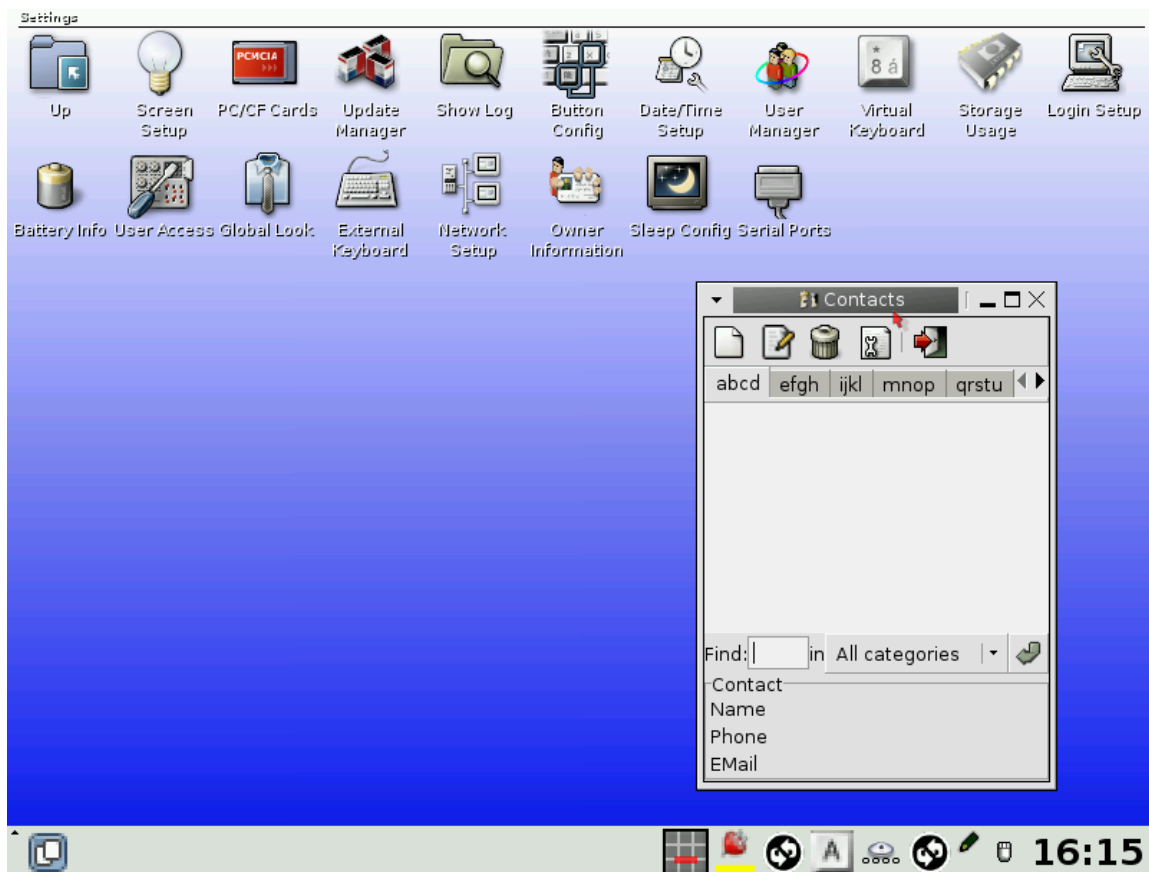
Und für den Entwickler verhält sich diese Hilfe genau wie die bereits bekannte Bubble-Help von den Desktops! Grundsätzlich gibt es bei GPE eigentlich nur wenige Details, die die Entwicklung zwischen GTK+-Desktop und GPE Applikationen unterscheidet.

## **Portabilität**

GPE hat bereits bewiesen, daß sein System durchaus ohne übermäßigen Aufwand zwischen unterschiedlichen Plattformen portabel ist. Grundsätzlich sind alle Komponenten in der Programmiersprache C geschrieben und verwenden keine Architekturspezifika (bspw. keine Endianess- oder Alignment-Probleme) und lassen sich daher problemlos für andere Architekturen übersetzen. Außerdem ist das Gesamtsystem so angelegt, daß plattformspezifische Anpassungen leicht vorgenommen werden können und nur wenige Komponenten betreffen.

GPE wurde bereits erfolgreich auf Compaq/HP iPAQ PDAs eingesetzt – gleichzeitig die erste GPE-Zielplattform überhaupt. Außerdem wurde GPE an die neueren Sharp Zaurus C-7xx Geräte angepaßt, ohne daß größere Änderungen notwendig waren.

Doch auch größere Vertreter der embedded Hardware bereiten GPE keine besonderen Schwierigkeiten. Mit nur wenigen Tagen Arbeitsaufwand konnten GPE an Siemens SIMpad angepaßt werden. Die größte Herausforderung dabei war es, einen anderen Window-Manager als Matchbox zu verwenden. Das SIMpad verfügt über eine Bildschirmauflösung von 800x600 Punkten. Alle Applikationen auf volle Bildschirmgröße anzupassen macht hier keinen Sinn mehr:



Alle weiteren Applikationen und Bestandteile des GPE System blieben weitgehend unangetastet und konnten unverändert übernommen werden.

Die Anpassungen an neue Geräte hat damit gezeigt und bewiesen, daß das Konzept von GPE grundsätzlich richtig und das Ziel der Universalität erreicht wurde.

## Ausblick

GPE befindet sich in stetiger Entwicklung. Ein Hauptziel, eine funktionsfähige grafische Oberfläche und Umgebung für PDAs zu schaffen, ist erreicht. Viele Details müssen noch erarbeitet werden.

Ein wichtiges nächstes Ziel ist die Vervollständigung der Integration von GPE mit dem Linux Desktop. Als erstes soll dies mit dem Gnome-Desktop erfolgen und dort insbesondere mit Evolution.

Aber nicht nur auf der Ebene der Applikationen gibt es noch viel zu tun.

GPE ist zur Zeit in die Familiar-Distribution von Handhelds.org integriert und damit für die HP/Compaq iPAQ Geräte optimal verfügbar. Die Unterstützung andere Plattformen ist zwar grundsätzlich möglich, doch in Familiar nicht integriert.

Ein vordringliches Ziel der nahen Zukunft ist dazu beizutragen, eine Vereinheitlichung, sprich Standardisierung, im Bereich der embedded Linux Distributionen und Entwicklungswerkzeuge zu erreichen. Dies betrifft vor allem den Prozeß zur Erzeugung der Distribution selbst. Ein vielversprechender Ansatz hierfür ist das aus dem OpenZaurus Projekt hervorgegangene Open-Embedded Projekt [XXX URL]. Im Gegensatz zur Familiar Distribution bei der jedes Paket

von einem Maintainer als Binärdatei beigetragen wird, ist Open-Embedded eine vollständige Entwicklungsumgebung. Innerhalb dieser Umgebung wird die komplette Distribution aus Quellcode vollständig neu übersetzt und in einem Build-Prozeß die architekturenspezifischen Boot-Images und alle zusätzlichen Pakete generiert.

Doch fast ebenso wichtig wie die Unterstützung der Entwicklung von GPE, ist die Unterstützung seiner Verbreitung, sogar die Unterstützung der Verbreitung von Linux auf embedded Plattformen überhaupt.

Linux und GPE haben zur Zeit noch keine Marketing-Abteilung, die pro-aktiv bei potentiellen Hardwareherstellern vorstellig werden kann und diese von Linux und GPE überzeugen kann – ganz im Gegensatz zu hier nicht genannten Unternehmen.

Daher ist es gerade jetzt an der Schwelle zum dritten Linux-Zeitalter so wichtig, Hardwareherstellern zu beweisen, daß es funktioniert, daß Linux eine ernsthafte Alternative sein kann, daß Linux und GPE sogar wirtschaftlicher sein können und das die Benutzer, *und das könnten Sie sein*, ein ernsthaftes Interesse an Linux und GPE auf diesen Geräten haben.

Erst wenn der Leidensdruck der Hersteller groß genug ist, werden wir Erfolg haben.

In diesem Sinne, ein erfolgreiches drittes Zeitalter

*nils faerber*