Curses - Motif - Qt
Migration eines ERP Systems

Kurt Keller
Parity Software GmbH
www.parity-software.com
kk@parity-software.com

Auf dem Weg zu einem umfassenden ERP System unter Linux:

Im folgenden Vortrag möchte ich aufzeigen, wie ein recht umfangreiches Softwarepaket im Laufe der Zeit an verschiedenste Basistechnologien (Betriebssysteme, Oberflächen, Tools) angepasst wurde.

Die Software durchlebte beginnend mit ASCII Oberflächen, einigen Gehversuchen direkt unter X11, danach Motif und schließlich unter Qt verschiedene IT-Zeitalter

in grauer Vorzeit

(1980 bis 1987)

- Gründung der Parity Software GmbH
- Entwicklung von ERP Business Software
  - für proprietäre Betriebssysteme
  - in einer proprietären Programmiersprache (HAI Basic)
  - individuelle Programmanpassungen je Kunde

Die Rechner zur Zeit der Firmengründung (1980) hatten 2 8-Zoll Floppy-Laufwerke, mit jeweils 1.2 MByte Kapazität.

Hauptspeicher 64 KByte, Prozessor war ein Intel 8080 mit „schnellen“ 1 MHz

Die Entwicklung der Parity Anwendungen erfolgte in einem BASIC Dialekt, dem HAI-Basic, von Holland Automation

Aufgrund spezieller Sprachfeatures in diesem Business-BASIC Dialekt konnten (für damalige Verhältnisse) sehr benutzerfreundliche Anwendungen erstellt werden.

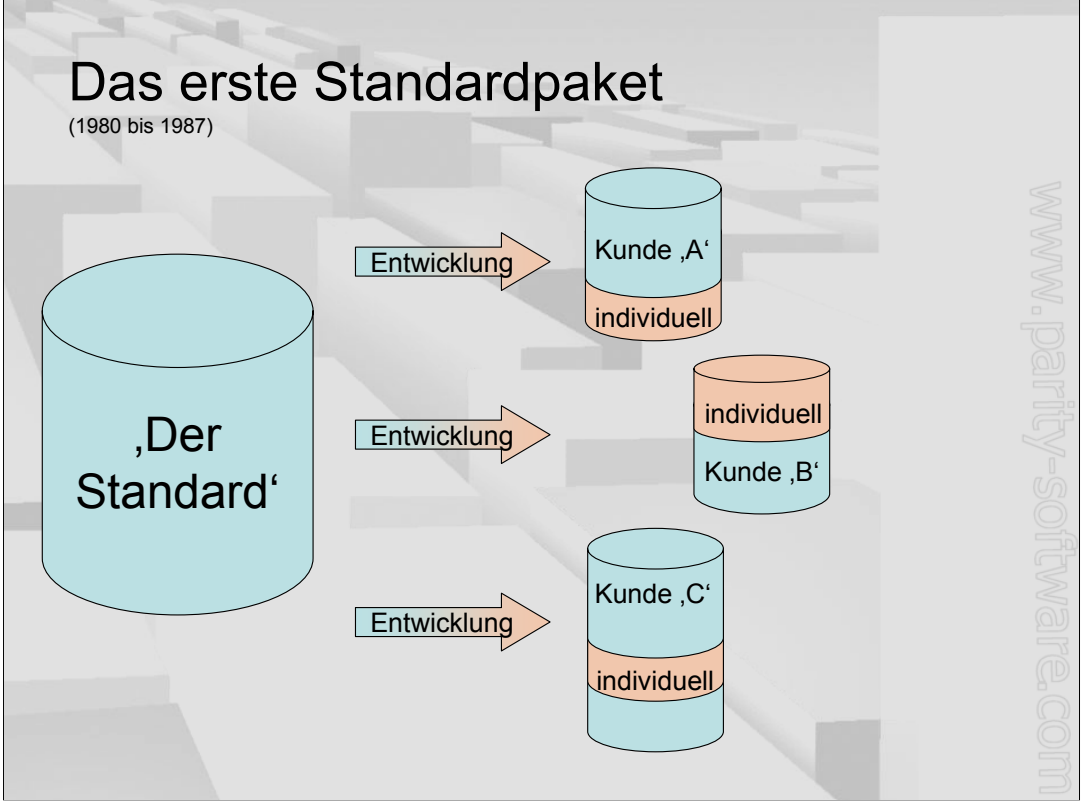
Das erste Standardpaket

(1980 bis 1987)

Parity entwickelte in BASIC ein Standardprogrammpaket für Fakturation und Auftragsbearbeitung

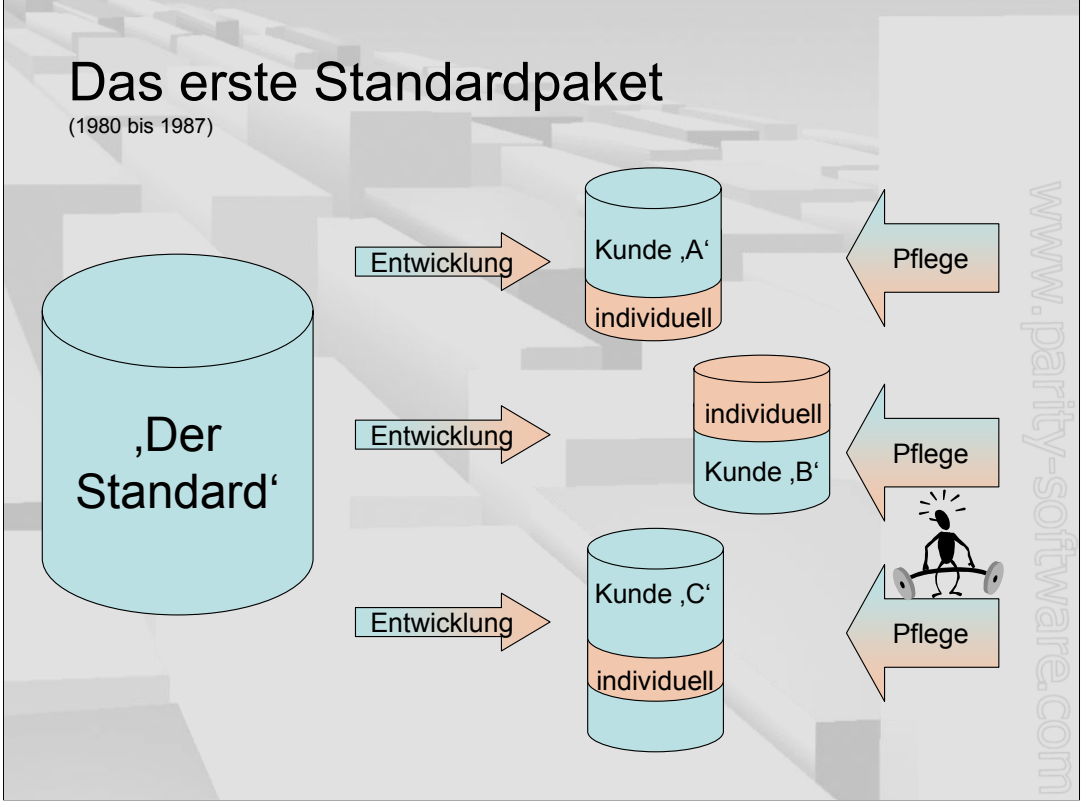
Der Standardlieferumfang dieser Software deckte alle Bereiche einer ‚Standard-Warenwirtschaft‘ ab.

Es handelte sich um 60 Einzelprogramme mit ca. 100.000 Lines of Code



Leider benötigte keiner unserer Kunden eine ‚Standard‘ Warenwirtschaft

deshalb wurden aus dem Standard für jeden Anwender individuell angepasste Programmversionen ‚geschnitzt‘



Dies führte nach einiger Zeit zu ca. 200 individuell gepflegten Programmpaketen, die untereinander nicht mehr austauschbar waren

Jede Installation musste individuell gepflegt werden

Oft war nur der Autor der Änderungen in der Lage, weitere Erweiterungen für den Kunden zu entwickeln

Man kann sich vorstellen, was z.B. gesetzliche Änderungen bewirkten …

Was tun?

- Die Situation war mittlerweile ziemlich unhaltbar
- Zu viele Programmstände und viel zu wenige Programmierer
- Der Supportaufwand uferte aus
- Die Grenzen der proprietären Umgebung (Basic, Betriebssystem, Hardware) waren erreicht

Was tun?

- Der Markt wandelte sich:
  - Kunden waren nicht mehr bereit proprietäre Systeme und deren Preise zu akzeptieren
  - Es wurde zunehmend Standardhardware (auf PC Basis) verlangt
  - Es wurde zunehmend „echte“ Standardsoftware verlangt

Neuentwicklung unter Unix

(1988 bis 1993)

- Neuentwicklung der kompletten Software unter Unix SVR3 und unter BSD Unix
- Entwicklung in K&R „C“
- Zielsetzung:
  - Software muss sehr leicht an Kundenanforderungen anzupassen sein
  - Kein zusätzlicher Support- und Pflegeaufwand trotz individueller Anpassungen
  - Neue Releases müssen stets aufwärtskompatibel sein

Die Entwicklung der BASIS-Software fand in Kooperation mit 3 weiteren Softwarehäusern statt. (Gemeinsame Datenhaltung, Bildschirmtreiber usw)

Customizing

(1988 bis 1993)

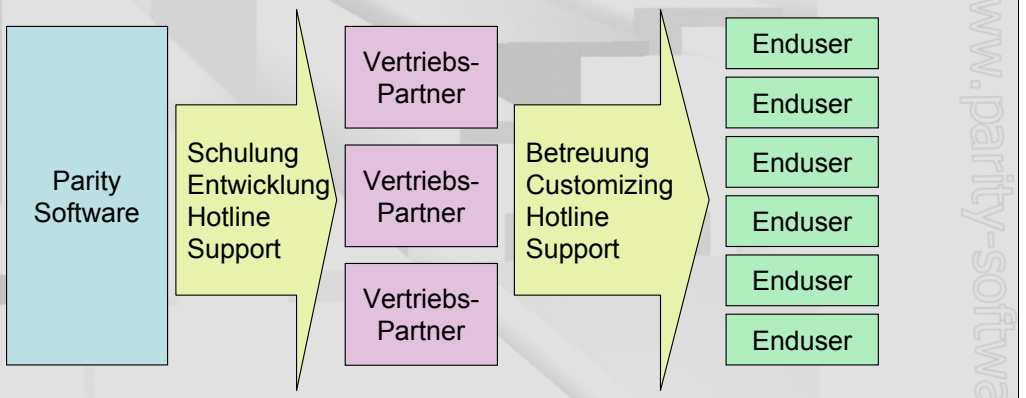
- Anpassungen an Kundenwünsche passieren ausschließlich ausserhalb des Programmcodes:
  - durch Änderung von Bildschirmmasken
  - durch Änderung des Datenbanklayouts
  - durch Änderung von Formulardefinitionen
  - durch Erstellen von Listen & Auswertungen
  - durch Scripting (später)
  - ...
- Customizing muss releasefähig sein!

Softwareanpassungen an individuelle Kundenwünsche nennen wir „Customizing“

Customizing wird durchgeführt ohne den Programmcode zu verändern

Releasefähig heisst: Auch jahrealte individuelle Anpassungen müssen nach dem Einspielen des nächsten Programmreleases automatisch wieder verfügbar sein

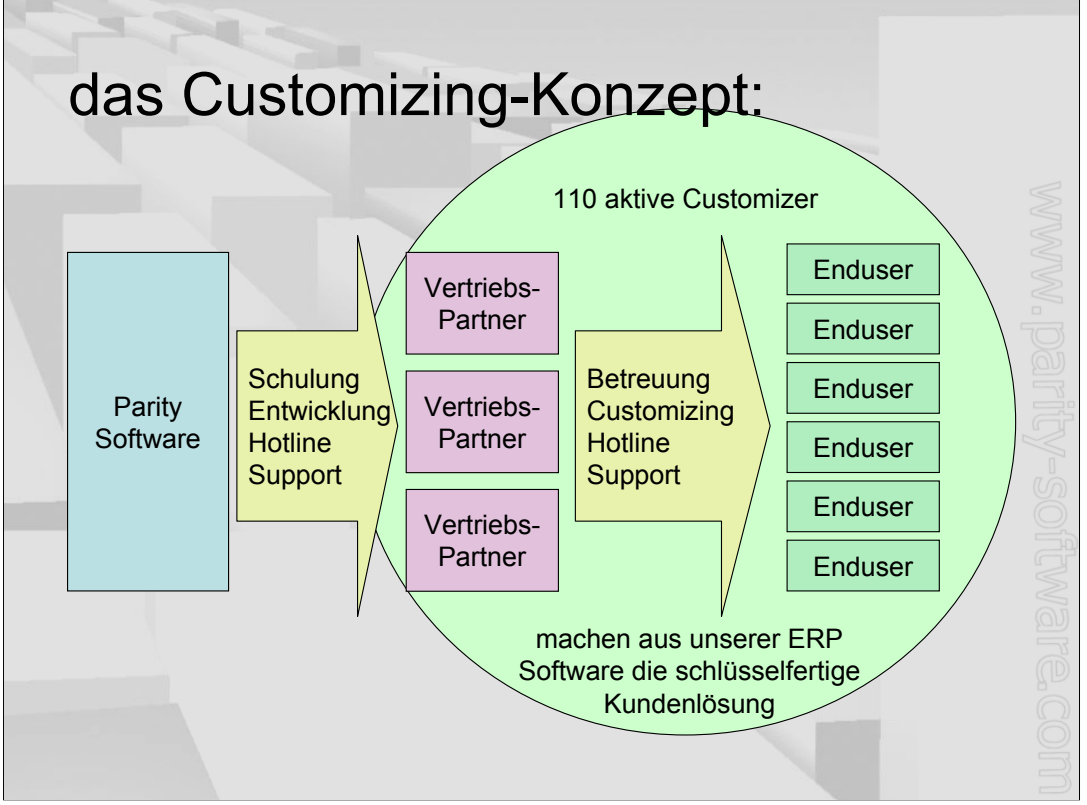
das Customizing-Konzept:



Parity stellt den Vertriebspartnern Standardlösung, Support, KnowHow und Marketinghilfsmittel bereit.

Die Partner betreuen den Endkunden (typisch: 6-40 Arbeitsplätze)

das Customizing-Konzept:

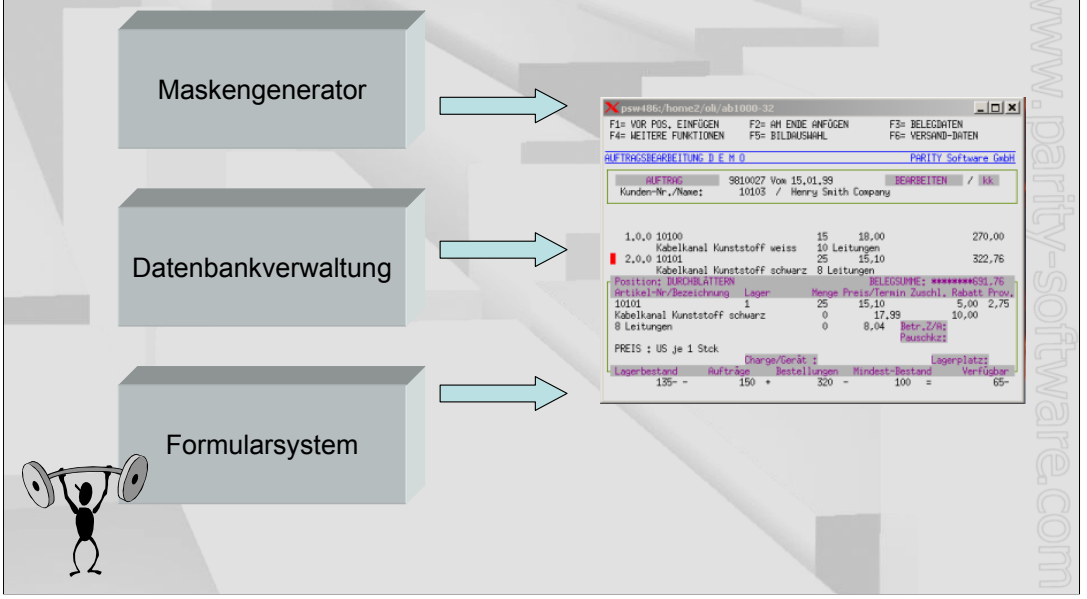


Die Partner führen das Customizing mittels Werkzeugen durch

An der Gesamtlösung macht der Anteil unserer Software-Erlöse häufig den geringeren Anteil aus.

benötigte Werkzeuge (BASIS-Software)

(1988 bis 1993)

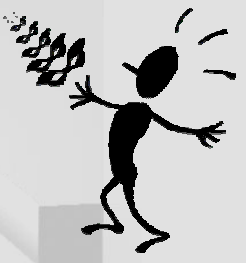


Der Ankauf fertiger Tools für Unix war nicht möglich, weil Tools entweder nicht verfügbar oder viel zu teuer waren

Maskengenerator:

(1988 bis 1993)

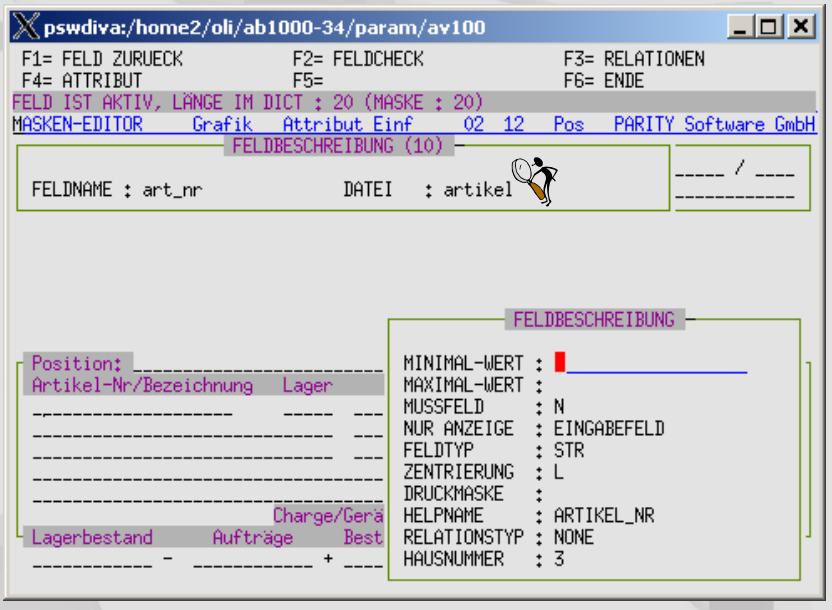
- Anpassen der Bildschirmmasken
- Hinzufügen und Entfernen von Eingabe- und Anzeigefeldern
- Hinzufügen und Entfernen von Bildern
- ohne Sourcecode zu verändern
- Feldprüfungen



Der Maskengenerator hieß damals zwar schon XForm, hatte mit X11 aber überhaupt nix zu tun

Maskengenerator:

(1988 bis 1993)



Maskengenerator beim Editieren einer Eingabemaske zur Verwaltung von Auftragspositionen

Datenbankverwaltung DB++

(1988 bis 1993)

- Hinzfügen und Entfernen von Datenbankfeldern und Tabellen über Oberfläche
- ohne den Sourcecode zu ändern
- ohne neu zu kompilieren
- keine ‚embedded SQL' mit Präcompiler
- sondern Prozedurale Schnittstelle

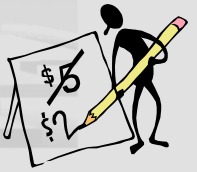
Als Datenbank wurde DB++ eingesetzt:

Mit DB++ waren wir in der Lage, auch auf Felder zuzugreifen, die gar nicht existierten. (Dann wurden von DB++ Defaultwerte geliefert!)

DB++ bot eine prozedurale Schnittstelle und verlangte nicht wie damals üblich, vorkompilierte SQL-Aufrufe hardcoded ins Programm zu schreiben

Formularsystem:

(1988 bis 1993)



- Drucker- und Treiberneutrale Druckausgaben
- Einfaches Anpassen von kaufmännischen Formularen (Rechnung, Lieferschein usw)
- ohne den Sourcecode zu ändern
- ohne neu zu kompilieren

Unser Formularsystem bestand aus einem selbstentwickeltem Basic PlugIn, mit dem man kaufmännische Formulare ‚programmieren' konnte

•Drucker- und Treiberneutrale Druckausgaben

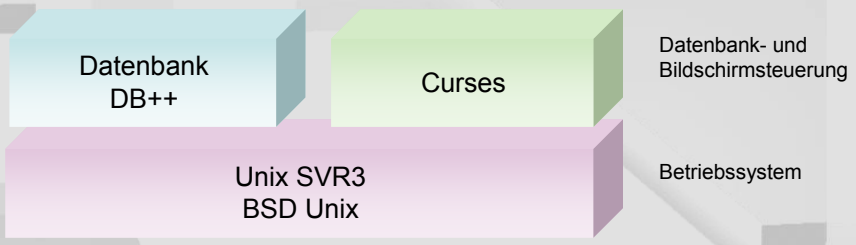
•Einfaches Anpassen von kaufmännischen Formularen (Rechnung, Lieferschein usw)

•ohne den Sourcecode zu ändern

•ohne neu zu kompilieren

Neuentwicklung unter Unix

(1988 bis 1993)

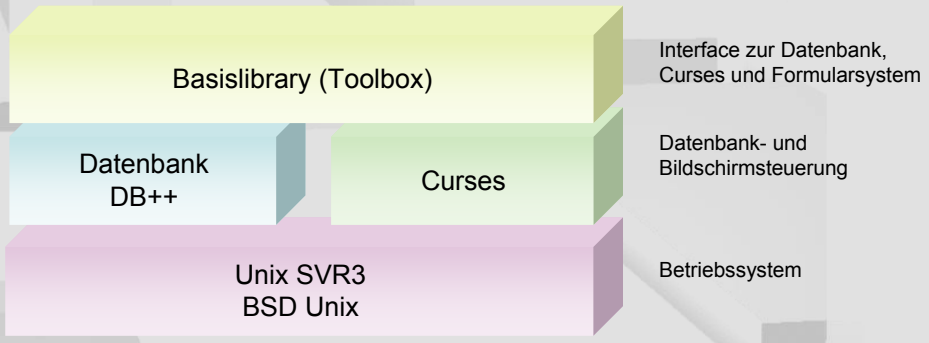


Basis der Entwicklung waren ein BSD basierendes Unix sowie SVR3 von AT&T. Beide Unixvarianten waren lizenzpflichtig.

Die Datenbank DB++ war zuständig für die Datenhaltung, Die Curses-Library übernahm die Basis-Bildschirmsteuerung

Neuentwicklung unter Unix

(1988 bis 1993)

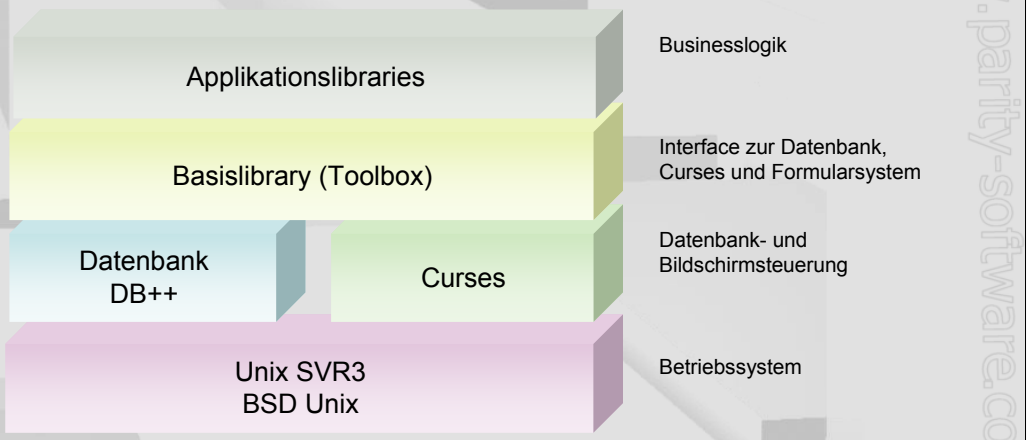


Eine Basislibrary stellt Funktionen zum Editieren von Eingabefeldern, Maskenfunktionalitäten und eine neutralisierte Schnittstelle zur Datenbank zur Verfügung.

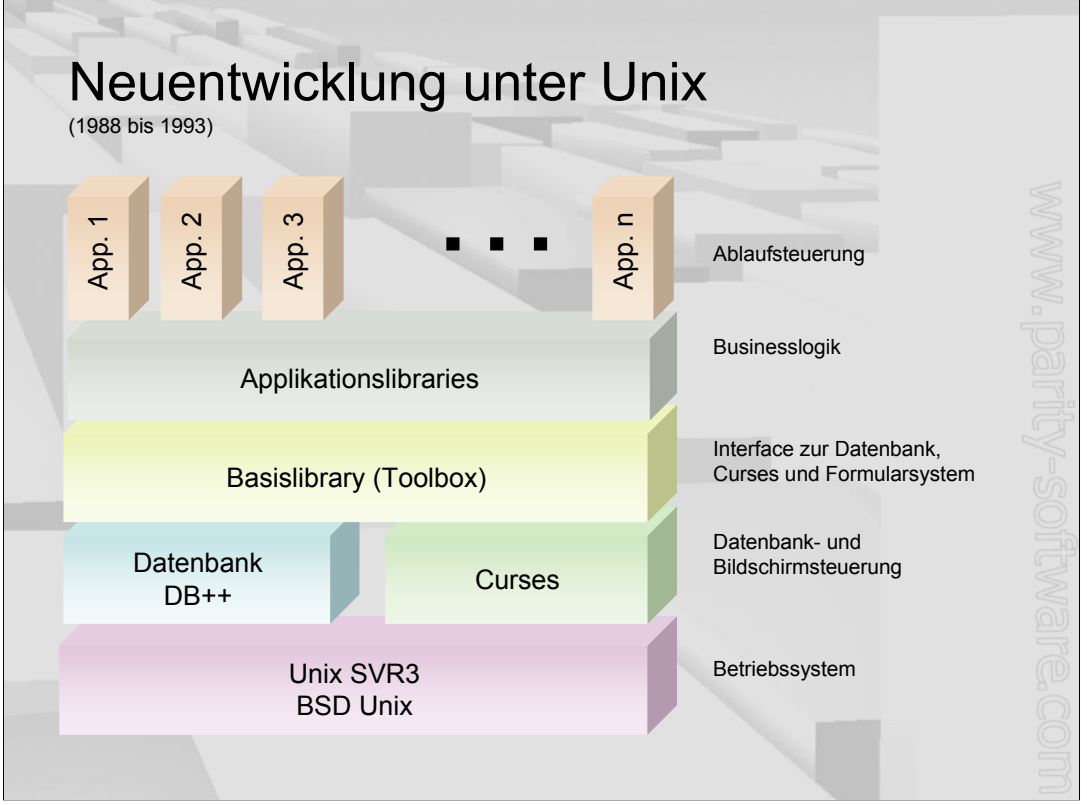
Unser Formularsystem war ebenfalls darin enthalten

Neuentwicklung unter Unix

(1988 bis 1993)



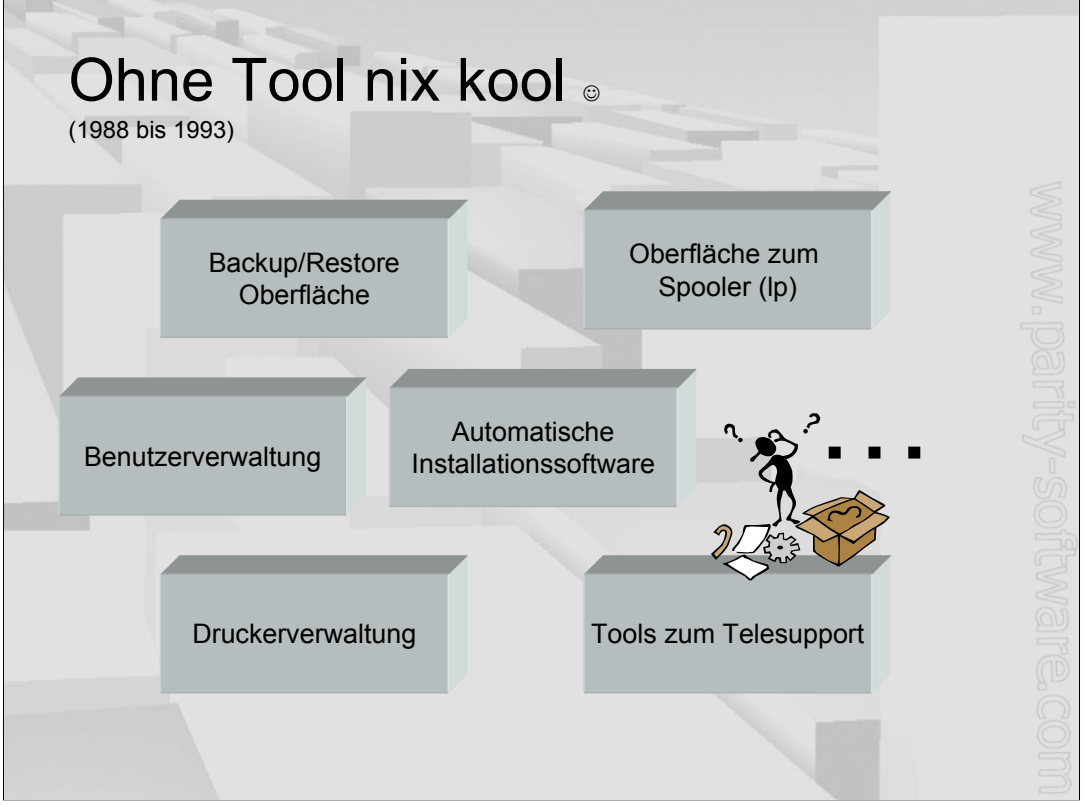
In den Applikationslibraries steckt die gesamte Businesslogik. (Lagerbuchung, Auftragspositionen verwalten, …)



Die Applikationen selbst stellen nur die Ablaufsteuerung zur Verfügung.
Die einzelnen Applikationen waren aufgrund dieser Struktur relativ klein

Ohne Tool nix kool ☺

(1988 bis 1993)



Für Unix existierten damals nur Commandline Tools zur Systemverwaltung.

Eine Oberfläche zur Systemverwaltung war nicht verfügbar.

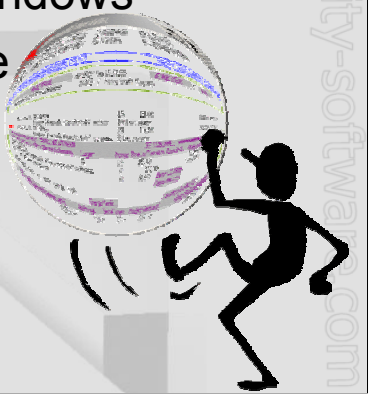
Unsere User hatten aber keinerlei Unix Know How

Deshalb wurden unter unserer Cursesbasierenden Benutzeroberfläche verschiedene Tools zur Systemverwaltung entwickelt

Vertriebliche Anforderungen

(Ende 1993)

- Ende 1993 wurden auch unter Unix zunehmend grafische Benutzerinterfaces nachgefragt
- O-Ton: Das ist aber nicht Windows
- => eine grafische Oberfläche musste her!



Obwohl die Softwarelösung mittlerweile ziemlich umfassend war, gab es zunehmend Akteptanzprobleme

Die Interessenten forderten ein GUI

O-Ton: „Die Software muss wie Windows aussehen“, „Das ist aber nicht Windows – oder?

Was tun?

- sollten wir ca 1,5 MLines of Code „in die Tonne“ werfen???
- und dabei alle bestehenden Kunden mit inkompatibler Software verärgern???

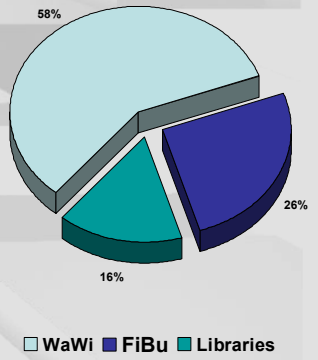
Was tun?

- sollten wir ca 1,5 MLines of Code „in die Tonne“ werfen???
- und dabei alle bestehenden Kunden mit inkompatibler Software verärgern???
- aber wie verpasst man einer ASCII basierenden Software ein GUI???

Vorgaben für die Umstellung

(ab 1994)

- 80% der Sourcen enthielten das Applikations KnowHow
- 20% entfielen auf die Basisbibliotheken
- Softwareänderungen dürfen nur im Bereich der Basislibraries erfolgen



Aus Support- und Distributionsgründen musste der Source- und Objektcode für beide Oberflächen identisch sein

d.h. alles oberhalb der BASIS-Library (mehr als 80% unserers Sourcecodes) blieb unverändert!

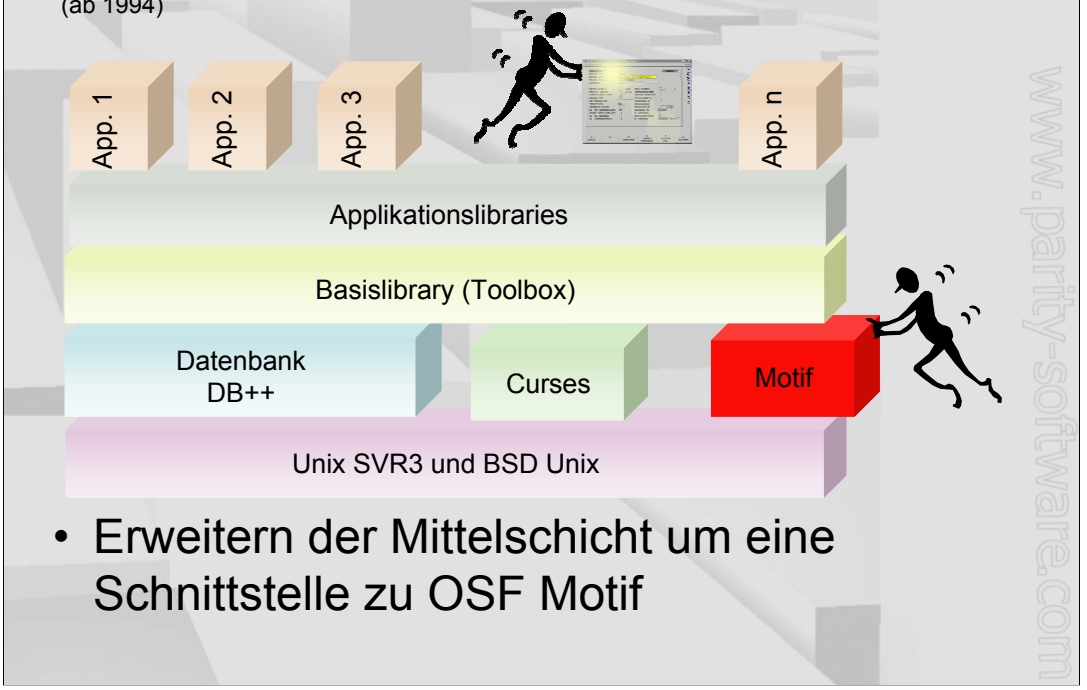
Durch Customizing entstandene Anpassungen für Endkunden (mittlerweile ca. 600) durften von der Änderung nicht beeinträchtigt werden!

Die Applikationslogik durfte nicht verändert werden.

Kundenupdates mussten automatisch laufen

OSF Motif als Toolkit

(ab 1994)



- Erweitern der Mittelschicht um eine Schnittstelle zu OSF Motif

Es kam (wegen Unix) nur eine X11 Umgebung in Frage

OSF Motif war damals das am weitesten entwickelte Toolkit

Deshalb wurden die Maskenfunktionen in der Basisbibliothek wurden um eine Schnittstelle zu OSF Motif erweitert.

sieht einfacher aus als es war :-)

(ab 1994)

- ein Beispiel:
  - Ein GUI Benutzer kann jederzeit in jedes Feld klicken!
  - Ein ASCII User muss jedes Feld mit der Entertaste bestätigen!

Das grosse Problem war die Ereignisorientierung unter grafischen Oberflächen:

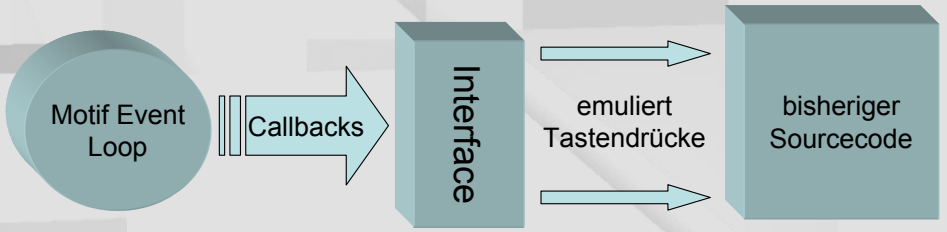
Ein Motifbenutzer konnte zu jedem Zeitpunkt mit der Maus in ein beliebiges Feld klicken

Ein ASCII Benutzer konnte das nicht, er musste Feld für Feld mit der ENTER-Taste ‚durchtasten‘

sieht einfacher aus als es war :-)

(ab 1994)

- ein Beispiel:
  - Ein GUI Benutzer kann jederzeit in jedes Feld klicken!
  - Ein ASCII User muss jedes Feld mit der Entertaste bestätigen!



Das grosse Problem war die Ereignisorientierung unter grafischen Oberflächen:

Ein Motifbenutzer konnte zu jedem Zeitpunkt mit der Maus in ein beliebiges Feld klicken

Ein ASCII Benutzer konnte das nicht, er musste Feld für Feld mit der ENTER-Taste ‚durchtasten‘

GUI und ASCII Oberflächen

(ab 1995)

entweder HUI ...

psw486:/home2/oli/ab1000-32

```
F1= VOR POS. EINFÜGEN      F2= AM ENDE ANFÜGEN       F3= BELEGDATEN
F4= WEITERE FUNKTIONEN     F5= BILDAUSWAHL           F6= VERSAND-DATEN

AUFTRAGSBEARBEITUNG D E M O                          PARITY Software GmbH

   AUFTRAG          9810027 Vom 15.01.99        BEARBEITEN  /  kk
 Kunden-Nr./Name:     10103  /  Henry Smith Company

 1.0.0 10100                          15      18,00              270,00
       Kabelkanal Kunststoff weiss    10 Leitungen
 2.0.0 10101                          25      15,10              322,76
       Kabelkanal Kunststoff schwarz  8 Leitungen
 Position: DURCHBLÄTTERN                      BELEGSUMME: ********691,76
 Artikel-Nr/Bezeichnung   Lager      Menge Preis/Termin Zuschl. Rabatt Prov.
 10101                     1           25      15,10              5,00  2,75
 Kabelkanal Kunststoff schwarz          0       17.99             10,00
 8 Leitungen                            0       8,04   Betr.Z/A:
                                                       Pauschkz:
 PREIS : US je 1 Stck
                                Charge/Gerät :                Lagerplatz:
 Lagerbestand      Aufträge     Bestellungen   Mindest-Bestand     Verfügbar
        135- -          150  +         320  -          100   =          65-
```

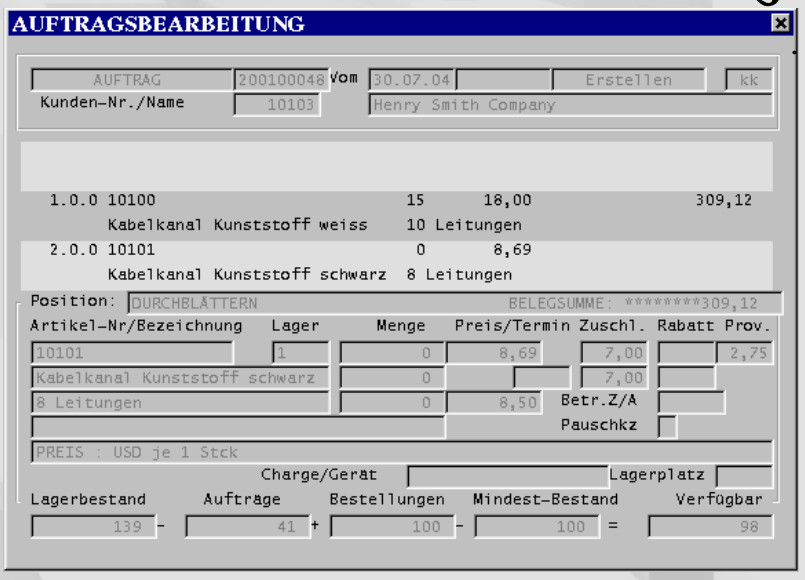

Die Anwendungen liefen nach wie vor auch unter VT100/VT220 Ascii-Terminals

die Kunden die nur ASCII Terminals einsetzten mussten ebenfalls mit den neuen Releases versorgt werden können

GUI und ASCII Oberflächen

(ab 1995)

... oder GUI



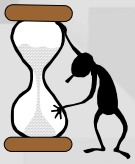
Dasselbe Programm, d.h. derselbe Objectcode lief auch unter X11/Motif

Das bedeutete

•jede Erweiterung der Anwendungsprogramme musste nur einmal entwickelt werden

•jede Maskenanpassung (Customizing) muss nur einmal gemacht werden.

es sollte jedoch nicht lange dauern ...



es sollte jedoch nicht lange dauern ...

... bis unsere Kunden wieder
neue Ideen hatten

neue Herausforderungen

- Die Applikationen wurden komplexer
- aus einzelnen Maskenbilder konnten bis zu 200 Widgets generiert werden
- da war Motif recht langsam
- Kein Windows Look & Feel (z.B. wegen fixed Fonts)
- Immer mehr Interessenten fragten nach Windows Plattformen

Unsere Motif Oberfläche konnte sich nicht flächendeckend durchsetzen:

•Zu viel Installationsaufwand, um einen X Server zu konfigurieren

•Zu viele ‚Unwägbarkeiten' bezüglich Fonts, Installationsumfeld

•X-Server für Windows PC's waren teuer und instabil

Was tun?

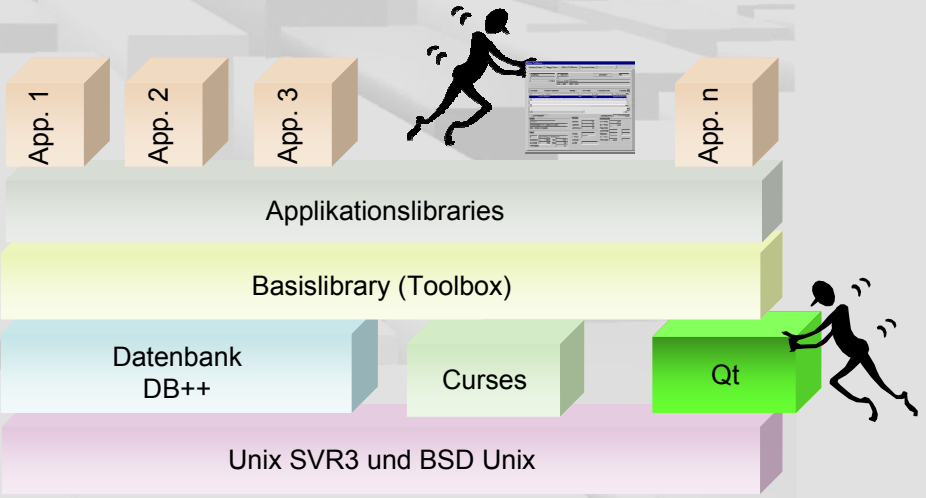
- Für die Microsoft Umgebungen gab es keine überzeugenden Lösungen unter Motif
- also doch: alles neu entwickeln?
  - vielleicht in Java oder VBasic?

Neuentwicklung hätte einen Bruch in der Kompatibilität bedeutet

Der Entwicklungsaufwand wäre sehr hoch gewesen

Umstieg auf QT Version 1.21

(ab 1997)



- Motif Toolkit wurde durch Qt abgelöst

Wir „fanden“ durch Zufall Qt

sehr positive Evaluierungsphase (Qt 1.2)

Die Motif Schicht wurde durch Qt ersetzt.

Die Schnittstellen-Library (Toolbox) wurde in grossen Teilen auf C++ umgestellt

Aufwand zunächst ca. 5 Mann-Monate

Unsere Erfahrungen mit Qt

(seit 1997)

- Unmittelbare Vorteile:
  - Performanceverbesserungen
  - Es waren nur sehr wenige Änderungen an den Applikationen erforderlich
  - mehr als 80% des Sourcecodes blieb unverändert
  - daher: schnelle Umstellung
  - Qt war leicht auf unsere Zielplattform (UnixWare) zu portieren

Darüberhinaus:

-gibt es die Möglichkeit, denselben Sourcecode auch auf Windowsplattformen zu kompilieren

Wir entschlossen uns daher, gleichzeitig mit dem Umstieg von Motif nach Qt auch die Windowsplattform zu unterstützen

Damit einher ging die Umstellung von „serverbased computing“ auf „client server computing“

Unsere Erfahrungen mit Qt

(seit 1997)

- Jedesmal wenn Trolltech eine neue Qt Version freigab konnten wir einen Teil unserer Basislibrary wieder ‚einstampfen‘
  - Farbauswahldialog
  - Mehrsprachigkeit
  - Toolbars
  - usw. usw.
- Im ersten Moment ziemlich nervig
- Aber: jedesmal ein weiterer Schritt zur Standardisierung

Unsere Erfahrungen mit Qt

(seit 1997)

- Qt (und als Folge davon unsere Software) ist ein stabiles Produkt mit guter Performance
- Sehr gute Weiterentwicklung und Produktpflege (Bsp. Unicode Umstellung)
- Die Implementierung unserer Software unter Windows ist einfach
- Ein Source-Code sowohl für Linux als auch für Windows

Unsere Erfahrungen mit Qt

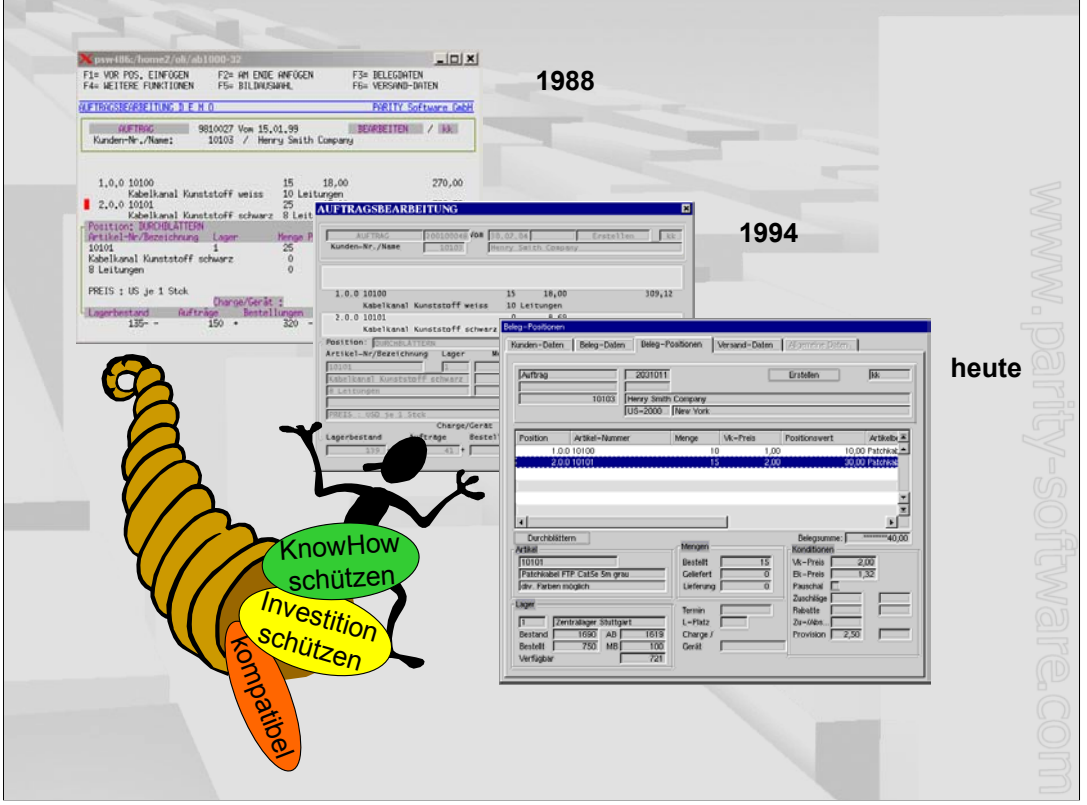
(seit 1997)

- wir sind mit Qt sehr zufrieden

It‘s Linux time

(Anfang 1999)

- 1999 wurde unsere komplette Softwarepalette auch unter Linux bereit gestellt
- Seit 2001 konzentrieren wir uns ausschließlich auf Linux- und Windows.



Seit mehr als 15 Jahren aufwärtskompatibel

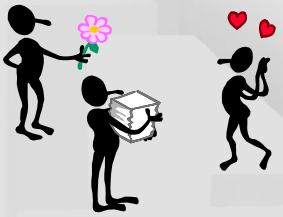
komplett Releasefähig, d.h. alle individuellen Änderungen bleiben auch nach einem Update des Standards erhalten!

Oberfläche ist nicht alles:

- Heute ist die Software komponentenbasierend
- Es gibt das komplette Framework als Perl-Plugin.
- Komfortabler Maskendesigner
- Derzeit wird ein umfassendes CRM Modul (in C++) entwickelt
- ... (siehe www.parity-software.com)

ist soweit doch alles perfekt
oder nicht ?

ist soweit doch alles perfekt
oder nicht ?



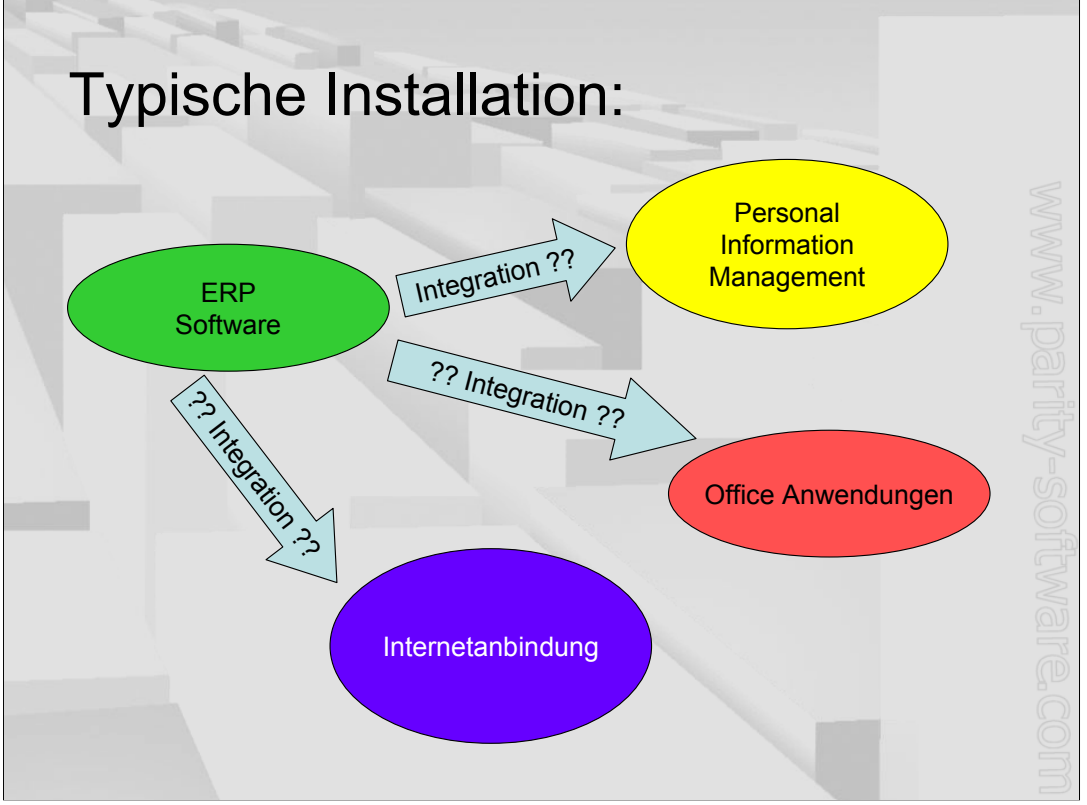
Nicht ganz, es gibt so einiges was wir
uns noch wünschen würden ...

- derzeit arbeiten mehr als 30.000 Menschen mit unserer Software

- derzeit arbeiten mehr als 30.000 Menschen mit unserer Software
- weniger als 1% davon arbeiten an einem Linux Desktop
- obwohl die Mehrzahl der Installationen auf Linux Servern basiert

- derzeit arbeiten mehr als 30.000 Menschen mit unserer Software
- weniger als 1% davon arbeiten an einem Linux Desktop
- obwohl die Mehrzahl unserer Installationen auf Linux Servern basiert

Woran liegt das?



In einer typischen Installation sind mehrere Module gefordert (ERP, PIM, Internetzugriff, Office)

prinzipiell ist das alles auch unter Linux vorhanden

aber ohne standardisierte Integrationsmöglichkeiten:

•es fehlen Schnittstellen oder sie sind uneinheitlich und nicht standardisiert (dcop bei KMail, aber wie funktionierts mit OpenOffice)

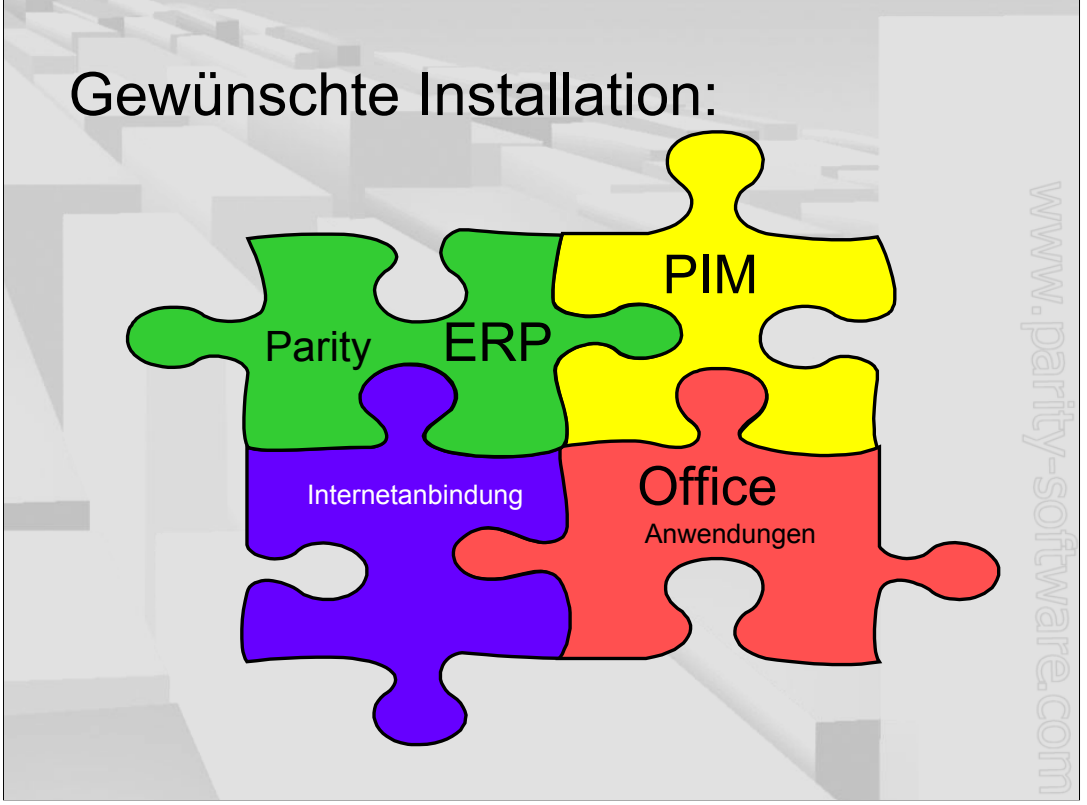
•Die Anwender möchten keine Scripts starten, um Daten an einen Kalender zu übertragen

•die Horizontalsoftware (PIM, Office) muss nahtlos in die ERP Software eingebettet werden

•KOffice User können nur manche MS-Office Dokumente verarbeiten

Das Problem ist, daß jeder Anwender andere Vorstellungen hat.

Es ist nicht möglich, diese Integrationen ‚hardcoded' zu realisieren



Anwender fordern:

„Wenn ich während der Auftragserfassung feststelle, daß die gewünschte Ware nicht verfügbar ist, dann möchte ich sofort einen Hinweis in meinen Aufgaben haben“

„Während ich mit einem Interessenten telefoniere möchte ich den gerade kalkulierten Preis sofort in einen Auftrag übernehmen können“

„Wenn jemand über unseren Internet Shop bestellt, dann möchte ich seine E-Mail Adresse automatisch in meinem Adressbuch haben“

„Wenn jemand den Artikel 4711 bestellt, dann soll die dazugehörige Dokumentation automatisch per E-Mail an den Kunden verschickt werden“

Diese Integrationsdichte ist derzeit leider nur auf Windows Plattformen erreichbar!

Was wir uns wünschen...

- konsequente Weiterführung der mit LSB (Linux Standard Base) begonnenen Entwicklung
- Weiterentwicklung der Integrationsschnittstellen in den Desktopanwendungen
- Weiterentwicklung der KOffice Im- und Exportfilter
- verlässliche Standardisierung und Dokumentation dieser Schnittstellen


PIM
ERP
Office
Internet
www.parity-software.com

Danke

für Ihre Aufmerksamkeit