

Hat (I-) 4GL eine Zukunft?

• **JA!**

Entstehung einer offenen Standard 4gl-Sprache auf
Basis von Open-Source

Dipl. Ing. Alain Siverly, Gründer der VENTAS-Gruppe

Kurzer Lebenslauf aus 4gl-Sicht

Sendeplanung für
Radio ffn programmiert

1987

1988-89

Textillösung für die CS AG
programmiert

SDM GmbH gegründet und
Außenhandelslösung programmiert.
Heutige Kunden: Wünsche Gruppe,
Melitta Kaffee

1990

1993

New Era ausprobiert und
rechtzeitig losgelassen

Kurzer Lebenslauf aus 4gl-Sicht

Einsatz von 4js BDS.
Enger Kontakt zu den
Entwicklern und zu JGS

1995

1999

ERP mit Suse auf Linux portiert,
mit 4js Adabas D eingesetzt.
Suse setzt VENTAS als ERP bei
sich ein

Wir prüfen Genero von 4js

2003

2004

Einsatz von
Aubit4gl

Kurzer Lebenslauf aus 4gl-Sicht

prüfen wie es weiter geht
(Java Portierung, C++-Portierung usw.)
Entscheidung, eine GUI für Aubit4gl zu entwickeln

2006

2008

4GL Allianz mitgegründet, um A-4GL auf breiteren Füße zu stellen und professionellen Support in mehreren Ländern anzubieten

Die VENTAS Gruppe

- VENTAS Research & Support: SW-Entwicklungen
- VENTAS Net Solutions: Linux-Umfeld
- VENTAS AG: Produkt-Vertrieb

Die Technologie der Open Source 4GL

- 4GL (kompatibel zu I-4GL)
- C (4GL wird in C übersetzt und kompiliert)
- C++ (GUI als Thin Client)

hat diese Technologie eine Zukunft?

I-4GL, C, C++

- I-4GL, sehr gut für kaufmännische Anwendungen geeignet (kann man die Schwächen beseitigen?!)
- C als Basis gut geeignet. Kann bei Bedarf zu C++ werden
- C++ ist die vermutlich leistungsfähigste Sprache, sehr schnell in der Ausführung. Mit Qt kann eine zeitgemäße GUI plattformübergreifend eingesetzt werden.

Was ist an 4GL zu verbessern? Was lohnt sich zu implementieren?

- Vorab:

sogenannte Hype-Sprachen und

Objektorientiertes vs.
funktionsbasiertes bzw. prozedurales Programmieren

OOP - ein Allheilmittel?

- OOP ist 40 Jahre alt (s. Simula)
- OOP kann unübersichtlich werden
(zu viele Klassen und Hierarchien sollten vermieden werden)
- C++ oder Python offerieren OOP, bieten aber funktionsbasiertes Vorgehen ebenfalls an, da man mit OOP nicht alles am Besten lösen kann

Hype-Sprachen anders gesehen

- Java orientiert sich an Smalltalk (nichts Neues unter der Sonne?)
- Java ist „einfach“, im Vergleich zu C++, weil noch unvollständig. Als „Alleskönner-Sprache“ wird der Leistungsumfang in den nächsten Jahren vermutlich deutlich zunehmen

Hype-Sprachen anders gesehen

- C# ist proprietär. Braucht man heute eine proprietäre Sprache, die auf andere basiert und somit nicht wirklich besser ist?
- C# und Java benötigen eine komplexe Runtime-Umgebung. Also Updates vom eigenen Produkt UND der Runtime-Versionen stehen im Laufe der Zeit an
- C# und Java sind nicht schneller als C oder C++ (Ausführungs-Zeit ist auch Geld)
- Die Entwicklungszeit in C#, Java (und C++) ist deutlich höher als bei 4GL

Bemerkungen

- Hype-Sprachen sind nicht grundsätzlich gravierende Neuerungen
- Altes wird in der EDV-Welt oft überarbeitet und als Neuheit präsentiert. Beispiel: Nach der Fat-Server/Ascii-Client-Ära war vorübergehend Fat-Client („Client/Server“) angesagt. Später gab es wieder Fat-Server und Thin-Clients (oder auch „Internet“ genannt)

Es sind nicht die Konzepte an sich, die erfolgreich sind, sondern was man damit lösen kann

Andere prozedurale Sprachen derzeit

- ABAP - die Cobol-ähnliche prozedurale Sprache von SAP ist die Sprache von SAP Core (die OOP-Erweiterung kam erst mit Version 6.10)
- C/AL von Navision ist ereignisgesteuert aber nicht objektorientiert (sie wird zukünftig in C# intern übersetzt)

„Warum muss ich es anders machen als SAP“?

Was spricht für 4GL heute und in Zukunft?

- Alleskönnersprachen sind bzw. werden komplex und dadurch langsamer erlernbar als 4GL
- OOP ist kein Muss, um beste kaufmännische Lösungen zu realisieren
- Prozedurale 4GL's findet man bei Marktführern kaufmännischer Lösungen

4GL: Konzentration auf das Wesentliche

- 4GL ist einfach weil spezialisiert
- Für „Rechnen und Daten verwalten“ ist 4GL optimal
- Bewiesen: mit 4GL können komplexe Anwendungen stabil und schnell laufen

Zeit ist Geld!

- Geld 1: 4GL ist schnell erlernt und SQL optimal integriert
- Geld 2: Keine Übermenge aus Funktionen und Leistungsmerkmalen, aus denen man die brauchbare Untermenge erst selektieren muss
- Geld 3: 4GL erlaubt nach wie vor schnelleres Programmieren als in Java, C# und Co.
- Geld 4: Fehlerfindung kann schneller sein als im evtl. abstrakten Klassen-/Objekt-Dschungel

Noch Fragen?

Was fehlt an 4GL, damit diese eine Zukunft hat? K.O.-Kriterien

- „Schicke“ GUI: die Qualität der GUI entscheidet zuerst über den Verkauf der Anwendung. Der Leistungsumfang folgt dann, da er vom Kunden oft nicht sofort ermessen werden kann
- Events und Multiple Inputs.
Ereignissteuerung und Drag&Drop sind seit langem marktüblich
- Zusätzliche Datenbanken anbinden
(JDBC spricht für Java z. B.)
- Besseres Reporting (PDF ist Minimum)
- Dabei sollte man auf Kompatibilität zu I-4GL achten und Befehl-Wildwuchs vermeiden
(Einfachheit bewahren, sonst ist der Vorteil hin)

Was fehlt an 4GL, damit diese eine Zukunft hat? Nice to have

- Nutzung der Anwendung im Webbrowser
- Interprogramm-Kommunikation
(andere Sprachen können es besser)
- Datenhaltung im RAM (SAPs-ABAP kann es besser)
- Für Neuentwicklungen begeistern
(die Summe der Merkmale muss attraktiv sein)
- Objektorientierung
- Code-Generator und Eclipse-Plugin
- Kompatibilität unter den *4GL-Anbietern. Normierung.
(vergl. Unix-Chaos in der Vergangenheit)

Der Weg zur Lösung: Die Ausgangslage in 2006 (Studie von Jean Georges Perrin)

- „I-4GL: Support durch IBM, jedoch keine Neuentwicklungen geplant“
- „Querix Hydra: sehr kompatibel zu I-4GL, gute Dokumentation, kein Web-Client“
- „Four J's: Marktführer, großer Leistungsumfang, BDS: sehr stabil, Genero: besseres Aussehen, Reporting: veraltet“
- 83,9% der 4GL-Anwender sind mit 4GL zufrieden (!)
- „Aubit4gl: sehr guten Support, keine leichte Installation, wenig Dokumentation, Open Source, GUI verbesserungswürdig“

Unsere Entscheidung:

eine bessere GUI für Aubit4gl zu programmieren

Was spricht für Open Source?

- Open Source ist in kritischen Umgebungen salonfähig (Bundestag, diverse Landesregierungen usw.)
- Vermeidung der Abhängigkeit von einem Anbieter (vermeiden von: nicht zeitgemäßem Leistungsumfang, zu langsamer Entwicklungsgeschwindigkeit, unerwünschten Richtungswechseln, nicht eingehaltenen Ankündigungen)
- Keine Gefahr durch finanzielle Situation des Anbieters oder Inhaber-Wechsel und deren Folgen
- Freiheit, das eigene Produkt zu optimieren
- Produkt wird i. d. R. durch eine größere Entwicklerzahl forciert als bei einem einzelnen Anbieter.

Was spricht für Open Source?

- Neue Ethik in der Gesellschaft: nicht nur Nehmen, auch Geben
- Vertrauen ist gut, auch wenn es Mut kostet
- Kooperation sichert den Erfolg, nicht Ausnutzen anderer

Grundsätzliche Einschränkungen von Open Source

- Niemand ist für Fehler Behebung oder Termineinhaltung verantwortlich zu machen
- Man kann Niemanden zwingen etwas Bestimmtes zu entwickeln
- Schlechte Dokumentation und „Handarbeit“ sind die Regel

Lösung? Duales Modell: Kosten- aber verantwortungsfrei vs. kostenpflichtig mit Verantwortungs-Verpflichtung

Die 4gl- Allianz: Ansatz, um von Open Source zu optimieren

- Professionelles Engagement beschleunigt die Entwicklung
- Das Ergebnis ist Open Source
- Professioneller Support bei Bedarf verfügbar
- Bessere Dokumentation im Rahmen kostenpflichtiger Leistungen

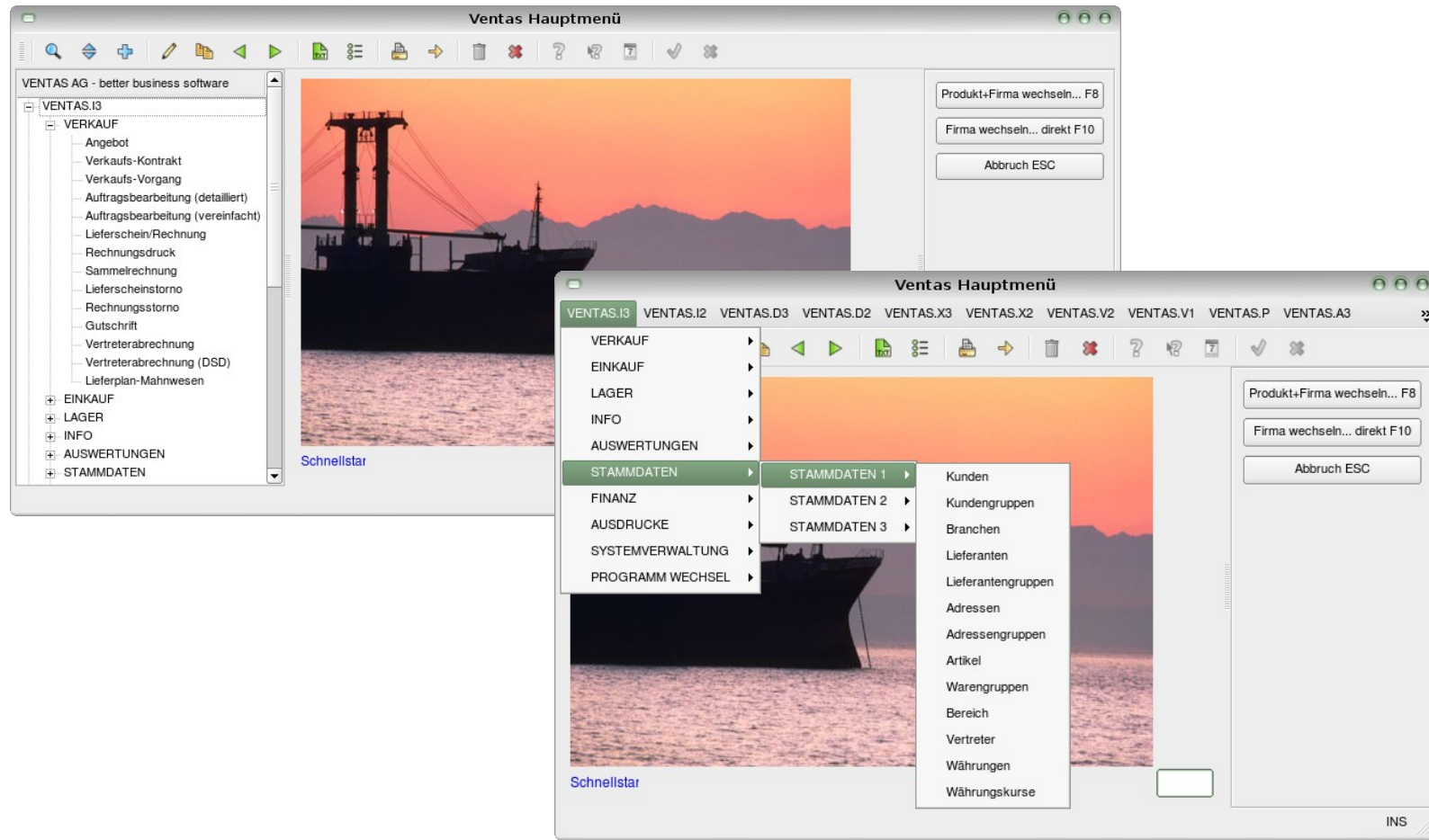
Ausblick: Die Zukunft von 4GL

- Für bestehende I-4GL-Applikationen: Modernisierung ohne die Risiken und Kosten einer Umsetzung in 3GL
- Für neue Applikationen: Reduzierung der Entwicklungszeit mit gleichzeitiger Akzeptanz der User durch schöne und ergonomische GUI

daher:

- Die 4GL Allianz hat sich vorgenommen, die Sprache zu modernisieren und attraktiv zu machen, dabei Bewährtes erhalten

Bilder-Galerie (der Stand per heute)



Bilder-Galerie (der Stand per heute)

The screenshot displays the VENTAS software interface with several overlapping windows:

- w_m_vkopf**: Main header window containing fields for 'Bearbeiter' (dbu), 'Kostenstelle', 'Auftrags-Nr.', 'vom', 'VK-Kontrakt', 'Vorgangs-Nr.', 'Referenz-Nr.', 'Lagerort', 'Kunden-Nr.', 'Sprache', 'Sammelrech.', 'Ansprechpartner', 'Lieferterm.' (27.10.2008), 'Währung', 'Betreff', 'Lieferung', 'Lief.anschr', 'Rchg.anschr', 'Text Kopf', 'Text Fuss', and 'Versand'. It also features a calendar for October 2008.
- POSITIONEN BEARBEITEN**: Window for processing positions, showing 'Auftrag 48', 'Kunde MEYER', 'Ges.Rabatt 4 %', and a table with columns: Pos. Nr., Artikel, Bezeichnung, T, Anz, Rab, VK-Preis, P, D, Liefer.
- VERSAND**: Shipping details window with fields for 'Lief.anschr', 'Versand', 'Tour', 'Versanddatum', 'Etiketten', 'Dampfer/LKW', 'Geschäftsart', 'DTG', 'Akkreditiv', 'Zuschlag', 'Versandzone', 'Vertreter1', 'Vertreter2', 'Währung', 'Gesamtwert' (0,00), and 'Gewicht'.
- Dialog Box**: A 'VENTAS' dialog box with a question mark icon asking 'Möchten Sie den Bearbeitungsschritt wirklich abbrechen?' with 'Ja' and 'Nein' buttons.

Wie mitmachen, wie testen?

- Die Client-Site: **www.ventas.de/opensource**
- Die aubit4gl-Site: **aubit4gl.sourceforge.net**
- Die Studie von J. G. Perrin befindet sich auf der Client-Site
- Zukünftige Sites der 4gl Alliance: www.4gl-alliance.com und www.4gl-alliance.eu

Willkommen!