EntwicklerCamp 2012

„Hilfe – ich habe geerbt!“

Bernhard Köhler

Vorstellung

- Quereinsteiger: Studium Physik / Astronomie
- Seit 1992 Beschäftigung mit Notes
- 1996: Erstmals fremde Anwendung übernommen
- Seither viele fremde Anwendung überarbeitet, weiter entwickelt oder Zulieferungen übernommen
- Seit 2004 selbständig in Kooperation mit Kollegen aus Österreich und Deutschland

Um was geht es?

- Anwendungen, an denen man bisher nicht mitgearbeitet hat
- Komplexe Anwendungen
- Aus eigenem Haus oder gänzlich fremde

- Erweiterungen
- Verbesserungen
- „Aufräumen nach Katastrophen“

Inhalte

- Vermittlung von Erfahrungen aus 15 Jahren Arbeit mit fremden Codes und aus 19 Jahren Notes-Entwicklung
- Methoden der Planung
- Analysemöglichkeiten und Folgen
- Was dürfen, was können und was müssen wir verändern?
- Wie ersparen wir uns zukünftige Arbeit?

Bestandsaufnahme

- A und O ist die genauest mögliche Analyse des Erbstücks.
- Das wird viel zu oft unterschätzt!
- Schritt 1:
  Was wird erwartet – und was geboten (Zeit, Ressourcen)
- Damit erledigt sich ggf. Schritt 2
  (leider oder glücklicherweise)

Bestandsaufnahme

Hieraus folgt eine der wichtigsten Punkte:

- Es geht um Verantwortung (übernehmen)
- Es geht um Rückgrat
- Es geht um eine Wenn-Dann-Argumentation, die man nur mit sicheren Fakten und Wissen führen kann!

Bestandsaufnahme

Von vornherein muss klar sein:

Nach (immer viel zu) kurzer Zeit wird die übernommene Anwendung nur noch mit dem neuen Entwickler(team) assoziiert!
(vom Anwender bis zur Geschäftsleitung)

Es gibt kein „Herausreden“ mehr!

Bestandsaufnahme

- Schritt 2:
  Wie funktioniert die Anwendung? Was leistet sie?
  Hierbei nicht nur vertrauen auf
  - Dokumentationen (mit Seltenheitswert!)
  - Angaben der Verantwortlichen
  Statt dessen niemals vergessen:
  - Die Anwender mit einbeziehen!
  Nur aus diesem Mix ergibt sich ein halbwegs verlässliches Bild!

Bestandsaufnahme

- Schritt 3:
  Erst jetzt beginnt die technische Analyse
  - Welche Datenbanken / Anwendungen sind involviert?
  - Welche Dokumentationen gibt es?
  - Welche weitere Informationen gibt es?
  - Rückgriff auf die bisherigen Programmierer / Projektleiter / weiteres involviertes Personal

Bestandsaufnahme

Untersuchung der technischen Möglichkeiten:

- Design offen?
- Ehrliche Bereitschaft, das Niveau der vormaligen Entwickler mit dem eigenen zu vergleichen.
- Wenn erforderlich, sind verschiedene Experten einzubinden (Admins sowieso!):
  LotusScript, Web-Techniken / Java, 3rd party

Bestandsaufnahme

Rechte an den Sources??

- Hierzu ist ggf. rechtliche Beratung einzuholen – keine Experimente!
- Diese Frage MUSS vorab geklärt werden
- Offener Quellcode bedeutet keinen Freibrief für Veränderungen!
- Durchaus unterschiedlich:
  Bestehenden Code ändern oder durch eigenen ersetzen

Verstecktes Design

Ganz schlechte Karten ...

- Änderungen in der Anwendung selbst unmöglich
- Rechtlich wäre dies auch fraglich (hidden design als Achtungssignal!)
- Mögliche Alternative: Zugriff auf die Daten aus einer separaten Anwendung („Fernsteuerung der Applikation“)
- Übernahme / Bearbeitung / Rückübertragung

Ganz schlechte Karten!

Tiefe Analyse

Keine weitere Planung ohne tiefe Analyse!

- Ohne Werkzeuge nicht möglich
- Wo steht was? Wie funktioniert was?
- Tools:
  - Datenbank-Synopse
  - Freeware wie NotesHound
  - Punktuelle Scanner wie ScanEZ oder Teamstudio Configurator

Tiefe Analyse

Perfekt (meine Meinung):

- Teamstudio Analyzer
- Quellcode als Notes-DB
- Fulltext search, Filter etc.

Please visit our Sponsor ;-)

Tiefe Analyse

Aber:

Kein technisches Hilfsmittel ersetzt das Nachvollziehen der Logik!

- Stimmen die Algorithmen?
- Passen die Rückgabewerte von Funktionen / Properties von Klassen? IMMER?
- Nur Code trauen, den man VERSTANDEN hat!
- „Doppelter Code“ bedeutet nicht Identizität!

Tiefe Analyse

Code nachvollziehen:

- Bei Verständnisproblemen Zeit mit dem Debugger einplanen!
- Alternativ / ergänzend: „Debug-Code“ einsetzen
- Tipp: „Debug-Code“ immer (!) eindeutig und identisch kennzeichnen, um ihn nach Abschluss der Analyse wieder entfernen zu können
- Dazu braucht es keinerlei zusätzliche Tools!

Vorstufe zur Realisierung

Ehrlich sein und Rückgrat beweisen!

- Es dauert nur wenige Wochen und wenige Änderungen, und SIE werden als „Macher" dieser Anwendung angesehen!
- Dokumentation Ist, Soll und FOLGEN
- „Budget" ausloten
- Hat jeder alle Konsequenzen verstanden?

Und notfalls: ABSAGEN! Rückgrat beweisen!

Realisierung

Von Anbeginn an: Dokumentieren!

- Was?
- Warum?
- Wieso?
- Hintergründe / Nebenwirkungen!

Das ist auch des Entwicklers Selbstschutz und „Billigmacher“!

Auch nachträgliche Dokumentation!

BASICS

(ALT und NEU / unfertig / Test müssen parallel existieren)

„Auf Almen darf man sorglos lieben, denn im Herbst wird abgetrieben!“

„Klimbim“, ARD 1847 (gefühlt)

Realisierung: Basics (1)

- Komplette Trennung von Entwicklungs- und Produktiv-umgebung. Ohne Wenn und Aber!
  - Eigene Domäne (Certifier, User).
  - Möglichst eigene reale Server- und Client-Umgebung
  - Keine Querzulassung
  - Positiver Nebeneffekt: Entwickler administriert (sollte den Admin aber „sein“ System prüfen lassen)
- Niemals mit Repliken aus Produktivsystem arbeiten – auch nicht in der Entwicklungsumgebung!

Realisierung: Basics (2)

Die „kleine Lösung“:

- Domino auf Entwickler-Kiste (keine Verbindung zu anderem System)
- Eigener Client (nicht nur getrennte Arbeitsumgebung!)
- Verbindung zum Test-Domino via 127.0.0.0
- Clients bis Version 7 konnten beliebig kopiert werden (geringe Anpassungen in NOTES.INI und AU)
- Gilt auch noch für Clients bis 8.5.2 im Basic-Mode!

Realisierung konkret:

- Wir verlassen das Feld allgemeingültiger Grundsätze und Erfahrungswerte!
- Jedes „Erbstück“ ist anders
- Immer gilt: Denken Sie an IHRE Nachfolger!
- Immer gilt: Frust auf die Vorgänger ist weder
  - hilfreich
  - produktivitätssteigernd
  - zukunftsweisend (und wie vermeide ich gleiches?)

Realisierung: Beispiele

Immer daran denken: Wir übernehmen jetzt selbst Verantwortung, daher

- NULL neues hard coding
- Altes hard-coding entfernen!
- Nochmals: Externe und interne Dokumentation auch für Alt-Code nachtragen. Nochmals: Selbstschutz!
- „Alte Zöpfe" DURCHGEHEND entfernen!

Kommunikation – eine andere Betrachtung

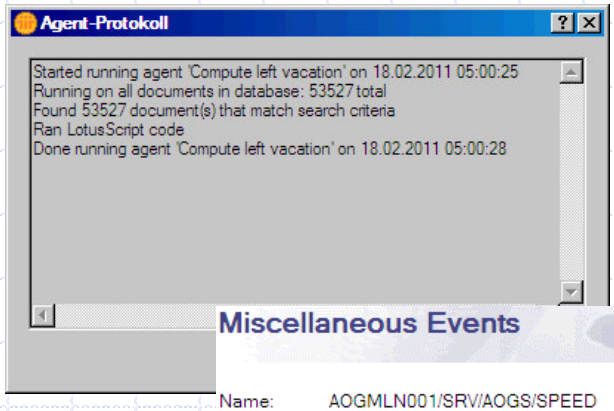
Häufig fehlen dem Administrator Rückmeldungen der Anwendung:

- Was haben die Agents über Nacht gemacht?
- Laufen die Agents fehlerfrei?
- Traten bei Anwendern Fehler auf? Wenn ja: Wo und warum?

Dies ist auch durchaus im Interesse des Entwicklers!

Agents und Protokollierung

- Agents sind „Black Boxes“
- Bordmittel wenig aussagekräftig:



Miscellaneous Events

Name: AOGMLN001/SRV/AOGS/SPEED
Time: 18.02 04:59:47 - 18.02 05:11:19

Miscellaneous Events:

18.02.2011 04:59:47 Replicator updated 1 document(s) in CLUSTER AOGMLN003/SRV/AOEU/SPEED iQSuite\g_log.nsf from iQSuite\g_log.nsf
18.02.2011 05:00:02 SMTP Server: Mail for [redacted] rejected for policy reasons. Recipient could not be found in the Domino Directory.
18.02.2011 05:00:03 Pushing iQSuite\g_log.nsf to CLUSTER AOGMLN003/SRV/AOEU/SPEED iQSuite\g_log.nsf
18.02.2011 05:00:03 Replicator updated 1 document(s) in CLUSTER AOGMLN003/SRV/AOEU/SPEED iQSuite\g_log.nsf from iQSuite\g_log.nsf
18.02.2011 05:00:25 AMgr: Start executing agent 'Compute left vacation' in 'HR\TimeManagementMN.nsf' by Executive '5'
18.02.2011 05:00:25 AMgr: [redacted]USER/AOGS/SPEED' is the agent signer of agent 'Compute left vacation' in 'HR\TimeManagementMN.nsf'
18.02.2011 05:00:25 AMgr: 'Agent 'Compute left vacation' in 'HR\TimeManagementMN.nsf' will run on behalf of 'Alpine Development/USER/AOGS/SPEED'
18.02.2011 05:00:25 Starting replication with server AOUKCV001/SRV/AOEU/SPEED
18.02.2011 05:00:26 Pulling AOUKCV001/SRV/AOEU/SPEED ddm.nsf from ddm.nsf
18.02.2011 05:00:26 Replicator updated 4 document(s) in ddm.nsf from AOUKCV001/SRV/AOEU/SPEED ddm.nsf
18.02.2011 05:00:26 Pushing ddm.nsf to AOUKCV001/SRV/AOEU/SPEED ddm.nsf
18.02.2011 05:00:26 Finished replication with server AOUKCV001/SRV/AOEU/SPEED

Agents und Protokollierung

Ist dieser Agent nun gelaufen?

- Keine Meldung im Agentprotokoll?
- Keine Aussage ausser „gestartet“ in der LOG.NSF
- Anwender klagen jedoch über fehlerhafte Werte!

Alternativen?

Agents und Protokollierung

Eine ganz schlechte Lösung: Print-Statements

„Keine (ich betone: KEINE) Print-Statements aus Agents – die müllen das Log voll!“ (Alexander S., Admin, Wien)

Formelsprache: Keine Möglichkeiten

LotusScript-Klasse NotesLog

- Beschränkte Möglichkeiten (Grösse begrenzt, entweder/oder-Problematik)
- Aber „besser als nichts“

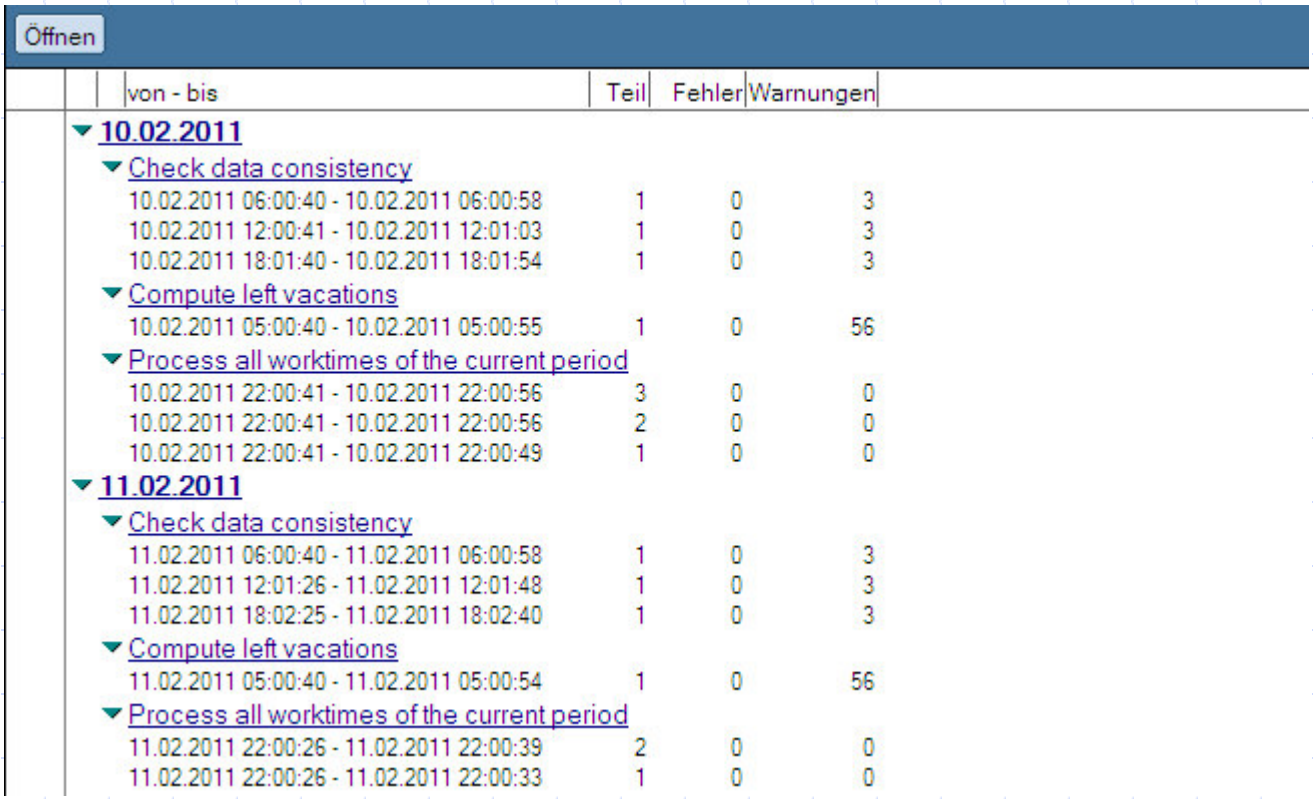
Agents und Protokollierung

Ideale Lösung: Eigene Routinen

- Speicherung beliebig grosser Protokolle
- Speicherung in beliebiger Datenbank
- Meldungen, Warnungen und Fehler
- Optional: Mailbenachrichtung an beliebige Personen („keine Mail vom Agent – alles i.O“)
- Maske zur Darstellung der Protokolle
- Ansicht für eine schnelle Übersicht
- Ideal wäre zwar RichText, aber unperformant!

Agents und Protokollierung

Ansicht:



von - bis	Teil	Fehler	Warnungen
10.02.2011			
Check data consistency			
10.02.2011 06:00:40 - 10.02.2011 06:00:58	1	0	3
10.02.2011 12:00:41 - 10.02.2011 12:01:03	1	0	3
10.02.2011 18:01:40 - 10.02.2011 18:01:54	1	0	3
Compute left vacations			
10.02.2011 05:00:40 - 10.02.2011 05:00:55	1	0	56
Process all worktimes of the current period			
10.02.2011 22:00:41 - 10.02.2011 22:00:56	3	0	0
10.02.2011 22:00:41 - 10.02.2011 22:00:56	2	0	0
10.02.2011 22:00:41 - 10.02.2011 22:00:49	1	0	0
11.02.2011			
Check data consistency			
11.02.2011 06:00:40 - 11.02.2011 06:00:58	1	0	3
11.02.2011 12:01:26 - 11.02.2011 12:01:48	1	0	3
11.02.2011 18:02:25 - 11.02.2011 18:02:40	1	0	3
Compute left vacations			
11.02.2011 05:00:40 - 11.02.2011 05:00:54	1	0	56
Process all worktimes of the current period			
11.02.2011 22:00:26 - 11.02.2011 22:00:39	2	0	0
11.02.2011 22:00:26 - 11.02.2011 22:00:33	1	0	0

Agents und Protokollierung

Maske:

Report-Dokument

Aktion:	Compute left vacations
Report Nr.:	1
Aktion gestartet:	18.02.2011 05:00:25
Aktion beendet:	18.02.2011 05:00:28
Anzahl Fehler:	1 / 1
Anzahl Warnungen:	9 / 9

```
18.02.2011 05:00:25:     Agent started
18.02.2011 05:00:25: >   Pablo Avila: No proofed contingent document found!
18.02.2011 05:00:26: >   Magdolna Balint: No proofed contingent document found!
18.02.2011 05:00:26: >   Baerbel Becker: No proofed contingent document found!
18.02.2011 05:00:26: >   Ralf Anton Beier: No proofed contingent document found!
18.02.2011 05:00:26: >   Sami Bejaoui: No contingent document found! (No Contingent document found for Sami Bejaoui / 2011!)
18.02.2011 05:00:26: >   Tilo Boeer: No contingent document found! (No Contingent document found for Tilo Boeer / 2011!)
18.02.2011 05:00:27: >   Markus Buesgen: No contingent document found! (No Contingent document found for Markus Buesgen / 2011!)
18.02.2011 05:00:27: >   Michael Burka: No proofed contingent document found!
18.02.2011 05:00:27: >   Liviu Cristea: No contingent document found! (No Contingent document found for Liviu Cristea / 2011!)
18.02.2011 05:00:28: !!  Laufzeitfehler in Modul <Compute left vacation>
Fehler <Variant does not contain a container> (Nummer 184 in Zeile 143)
```

Wir sehen: Der Agent ist nicht erfolgreich abgeschlossen worden!

Agents und Protokollierung

Mail:


```
This is an automated message from the database  'Time Management'
This notification is connected to the document behind this link: (Document
link: )

Mitteilung:
In folgender Datenbank ist ein Problem aufgetreten:
Datenbank:      Time Management
Server:         AOGMLN001/SRV/AOGS/SPEED
Datei:          HR\TimeManagementMN.nsf

Die Meldung lautet:
Agent 'Compute left vacations' reports a serious error: Compute left
vacation
```

Agents und Protokollierung

Kein Zauberwerk!

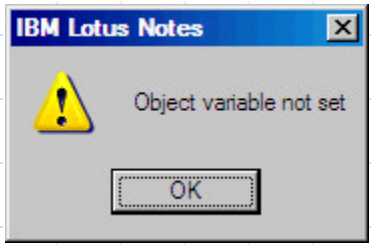
- Eine (einfache) Maske mit Readers Item
- Eine (einfache) Ansicht
- Eine ScriptLibrary mit Routinen für
  - Initialisierung der Routine (Objekte)
  - Erzeugen Report-Dokument
  - Schreiben der Meldungen incl. Erkennung eines (Textfeld-)Überlaufs
- Bei gesetztem ErrorCode wird optional die Mail erzeugt.

Agents und Protokollierung

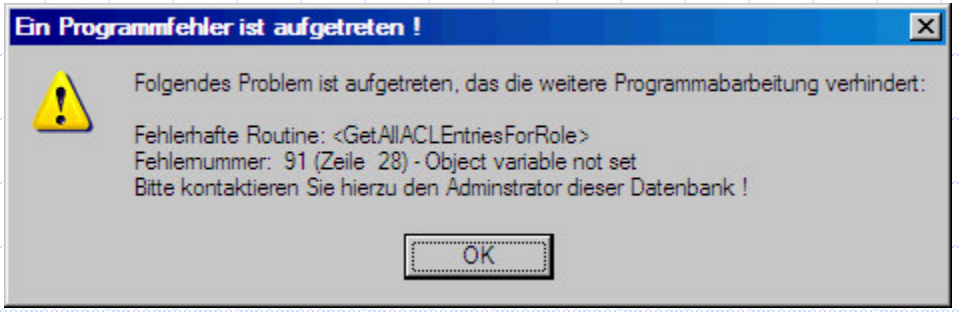
Und jeder Admin freut sich, wenn der Programmierer im Setup eine Einstellung anbieten, nach wieviel Tagen alte Protokolle gelöscht werden sollen!

Agents und Protokollierung

Visualisierung von Client-Fehlermeldungen:



Übel ...



Hilfreich für

- den Anwender
- den Administrator

und vor allem für den Entwickler!

Agents und Protokollierung

Kein Zauberwerk!

- On Error Goto ErrorRoutine
- ErrorRoutine:
  - Erzeugen der Meldung mit Err, Error$ und Erl
  - Prüfung, ob an ausgewählte Personen eine Mail erzeugt werden soll

Nicht vergessen, Err wieder zurückzusetzen!

Vor dem Erreichen von ErrorRoutine Codelauf beenden!

Realisierung: Weitere Beispiele

Nochmals: Wir übernehmen jetzt selbst Verantwortung, daher

- Datentypen stringent speichern!
- Falls erforderlich, Umstellung nicht scheuen und Bestandsdaten umstellen
- Tipp: Externe Datenbank für Datenbestands-Updates einmalig schreiben und situationsbezogen anpassen: Ein Aufwand – vielfacher Nutzen

Realisierung: Weitere Beispiele

- Kann bekannter / eigener vielfach erprobter Code eingesetzt werden?
  - Wenn ja: Umsetzen!
  - Wenn nein: Immer versuchen, hier neu erforderliche Routinen allgemeingültig zu erstellen

-> Auch die ungeliebte Anpassung / Verbesserung des „Erbstücks“ kann für andere (alt-)Anwendungen erneut Nutzen bringen!

Realisierung: Weitere Beispiele

Performance-Analysen:

- Anwender-Erfahrungen einholen!

Agents:

- Agent-Laufzeiten ermitteln, Relation zur Menge bearbeiteter Daten herstellen
- „Zeitfresser" ermitteln und auf diese konzentrieren
- Algorithmen in Frage stellen!
  (schon das Verfahren kann falsch sein)
- Problem neu durchdenken

Realisierung: Weitere Beispiele

Performance-Analysen:

Ansichten (1):

- Tatsächlich benötigte Ansichten ermitteln
  (Anwender, Log befragen, Code analysieren – Bezüge finden)
- Arbeitsweise der Anwender ermitteln – auch dies ist immer zu hinterfragen („Das war schon immer so!“)
- Ansichten - > Reports machbar?
  (Eigene Reports, Excel o.a.)

Realisierung: Weitere Beispiele

Performance-Analysen:

Ansichten (2):

- Performancekiller finden
  unnötig aufgeblähte Indizes
  komplizierte Spaltenformeln
  umsortierbare Spalten (ggf. ist eine weitere Ansicht besser!)

- vorab in der alten Anwendung: updall –R (Indizes löschen)
  Oft ist ein Uralt-Index schon die Ursache!

Realisierung: Weitere Beispiele

Performance-Analysen:

Masken (1):

- Performancekiller finden

  Permanente Lookups, auch im EditMode

  Vermeidbare automatische Feldaktualisierung

  Code in PostRecalc untersuchen!

  Code In QuerySave / PostSave / QueryClose untersuchen

Realisierung: Weitere Beispiele

Performance-Analysen:

Masken (2):

- Performancekiller finden

  Aktionen in Masken analysieren

  - Dialoglisten, Radio-/Checkboxes: Was passiert da?
  - Aktionsleiste
  - Schaltflächen
  - Weitere Aktionen

Realisierung: Weitere Beispiele

Masken (3):

- Validierung untersuchen
  Können wir konsistente Daten sicherstellen?
  - erforderliche Felder ausgefüllt?
  - sind die Daten valide?
  - Felder, die in Ansichten benötigt werden
    (Beispiel: Zeilenschaltungen verhindern)
  - Erneuter Hinweis: Passende Datentypen!!!

Realisierung: Weitere Beispiele

Masken (4):

- Frontend versus Backend:

  Item „SaveOptions“, ggf. auch „MailOptions“:

  Sicher verwendet?

  Rutscht uns unter bestimmten Voraussetzungen mal ein SaveOptions = „1“ ins Backend?

  RichText:

  Beispiel: Dokument im EditMode

  NotesDocument.Save ohne NotesUIDocument.Save

  → Änderungen in RTF sind über den Jordan!