Notebook statt Rechenzentrum

Jan Golka
Data Design & Management GmbH

Bauernwaldstraße 126
70195 Stuttgart
Telefon: + 49 – 711 – 69 70 71 – 10
E-Mail: info@d-d-m.de
www.d-d-m.de

Agenda

- Kurzvorstellung
- Einführung.
  - IT Landschaft.
- Vorstellung der Altanwendung.
  - Probleme der Altanwendung.
- Neuentwicklung: das Stammdaten-Managementsystem.
- Re-Engineering der Altanwendung.
- Beschreibung des realisierten Systems.
- Fazit.

Kurzvorstellung

- Wir sind ein kleines, seit über 20 Jahren existierendes Softwarehaus (bis 1998 unter anderem Namen). Seit ca. 1995 liegt unser Schwerpunkt auf der „Oracle Anwendungsentwicklung“.

- Geschäftsführer: Dr. Jan Golka

  - 1. Programmiersprache (Algol) : 1965.
  - Selbständige Beratertätigkeit seit 1987.
  - 1. Oracle RDBMS Version : 5.1B (30 Disketten, 1989).
  - Alle Oracle Versionen seit 7.0.
  - Bevorzugte Programmiersprache : PL/SQL.

Unternehmensphilosophie

- Laut Umfragen stehen die meisten deutschen Unternehmen dem Trend, die IT-Leistungen in Billiglohnländer zu verlagern, skeptisch gegenüber.
  - Auch wir unterstützen tatkräftig diese Meinung, indem wir in unserem Kerngeschäftsbereich Festpreis-Leistungen anbieten, die kein Offshoring unterbieten kann. Wir arbeiten schnell, kompetent und effizient mit einem kleinen Team hochkarätiger Mitarbeiter in Deutschland.
- Wir sind der Meinung, dass ein Softwareprojekt am effizientesten von einem *möglichst kleinen Team ausgezeichneter und hochmotivierter Spezialisten* zu realisieren ist.
  - Ein hochqualifizierter Entwickler, ausgerüstet mit den richtigen Tools, ist nicht nur wesentlich schneller sondern macht auch weniger Fehler als der Durchschnitt.
  - Statt Programme »gesund zu testen« versuchen wir, sie gleich fehlerfrei zu entwickeln. Per Saldo ist dieses Verfahren wesentlich effizienter, schneller und billiger als jeder Versuch, die Lohnkosten zu minimieren.
  - Wir sind der Meinung, dass ein erfahrener Softwareentwickler und Consultant imstande sein muss, eine gut formulierte Aufgabe in ein detailliertes Festpreisangebot umzusetzen.

Einführung

- Im Rahmen einer Neuentwicklung eines Systems für die Belieferung der Schnittstellenpartner mit Produktionsstammdaten sollte auch eine logistische Altanwendung, ein Hand-gesteuertes Workflow aus Shellskripten, C- und awk-Programmen, sowie SQL*Loader Skripten in das neue Stammdaten-Managementsystem übernommen werden.

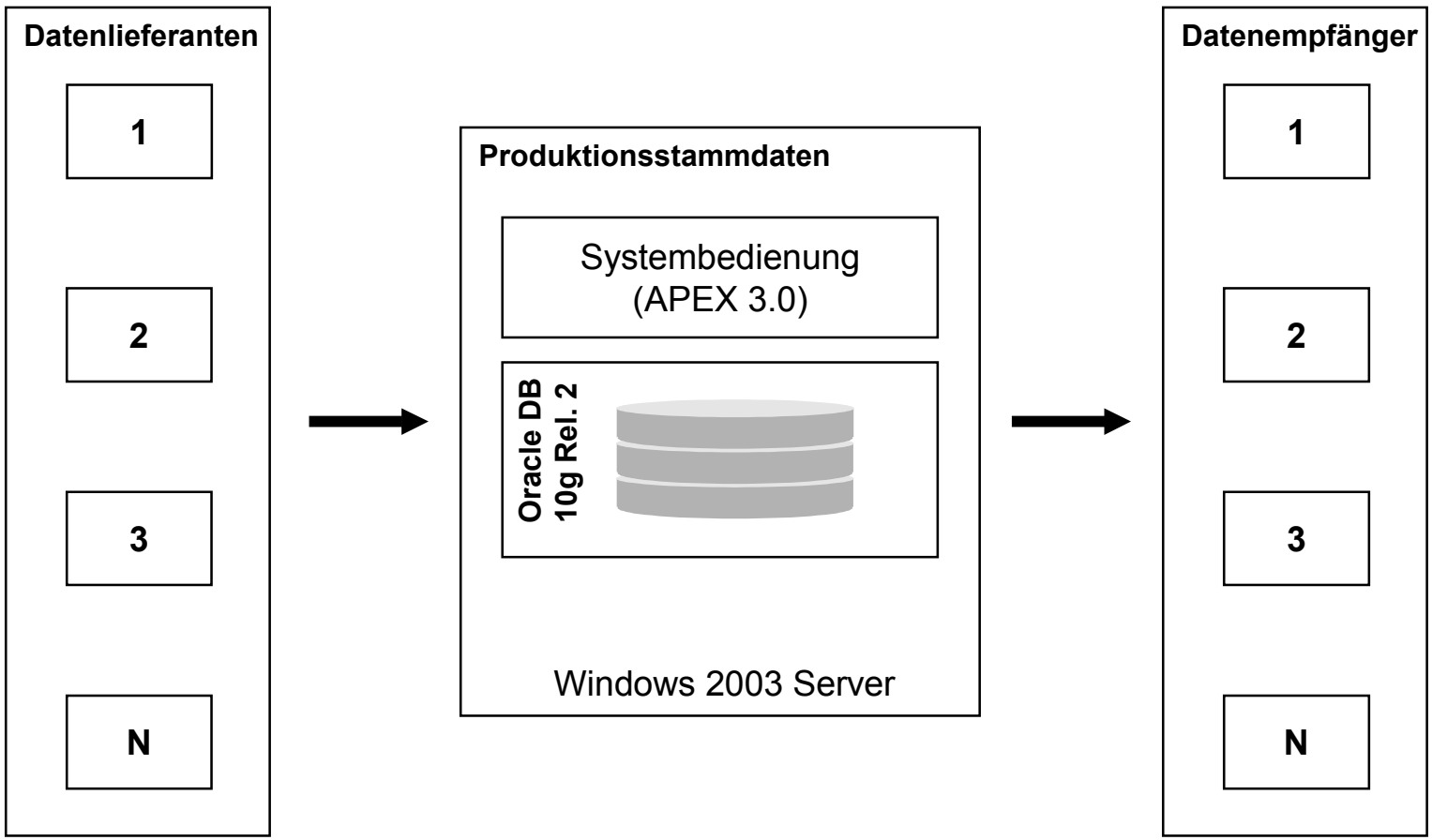
- Dieses neue System, ein Produktionsstammdaten-Managementsystem, besteht aus einer Oracle 10g Rel. 2 Datenbank und mehreren PL/SQL Programmpaketen. Die Bedienung des Systems erfolg über eine Intranet-Bedienoberfläche, die mit Hilfe von APEX erstellt wurde.

- Als *Datenlieferanten* dienen, zum einen, mehrere Oracle Datenbanken im WAN und, zum anderen, die Fachseite der Anwendung, die ihre spezifischen Daten über die Bedienoberfläche ins System einpflegt.

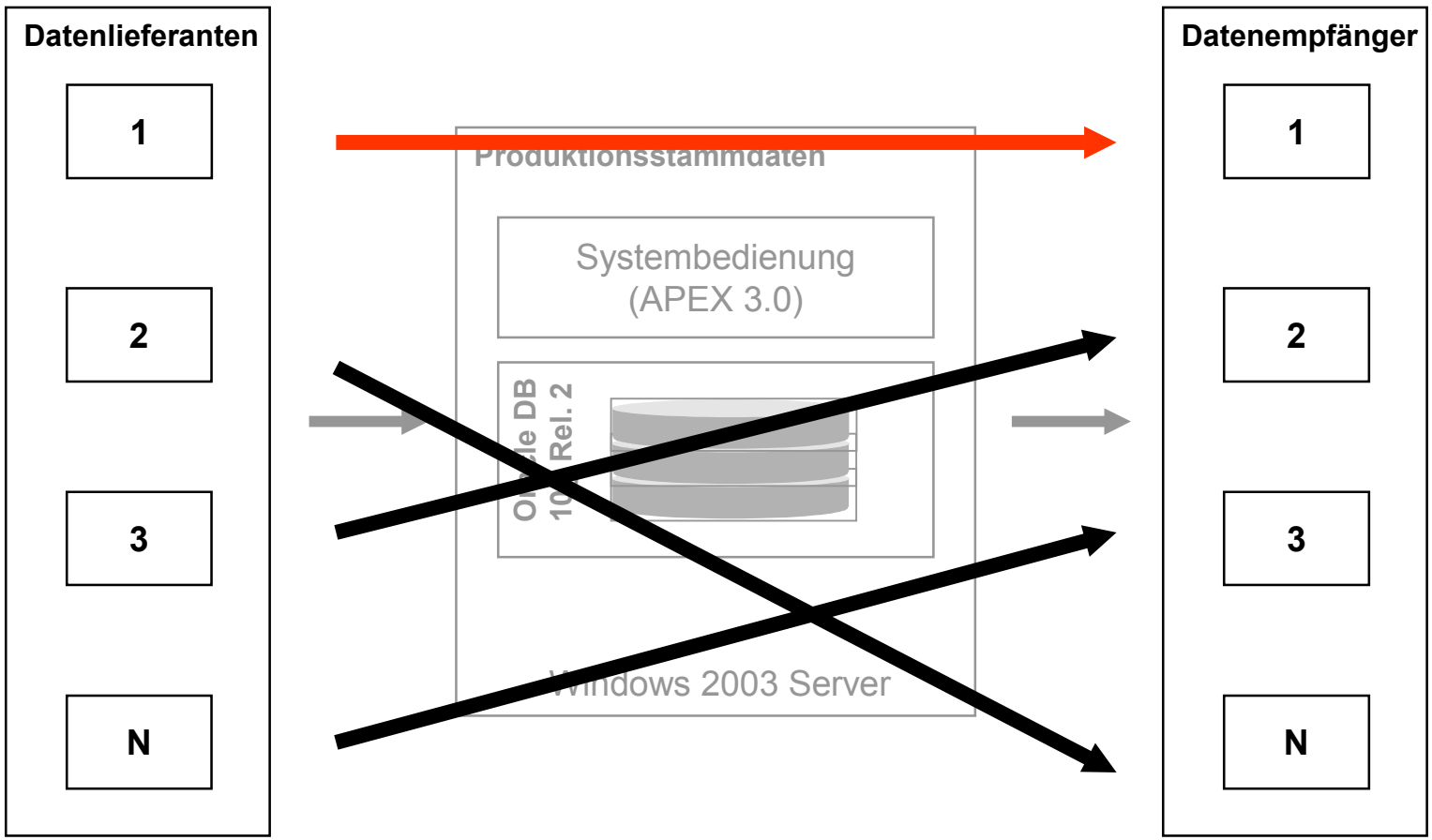
- Die *Datenempfänger* sind Produktionssysteme die entweder keine Datenbank verwenden oder keinen Direktzugriff auf ihre Datenbanken erlauben.

    - Ausgetauscht werden entweder XML Textdateien, CSV Dateien oder Dateien im Anwendungsspezifischen Format.

IT-Landschaft



Ursprüngliche IT-Landschaft



Altanwendung

- Wir reden hier über eine von diesen Datenschnittstellen, die gegenwärtig noch von einer Altanwendung bedient wird.

- Diese Altanwendung ist ein Hand-gesteuertes Workflow aus Shellskripten, C- und awk-Programmen, sowie SQL*Loader Skripten. Diese über 15 Jahre alte Anwendung benutzt die Oracle Datenbank nur um die Umlaute zu expandieren und ihre Zwischendateien zu sortieren.

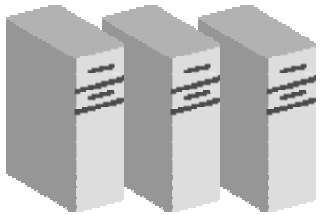
- Die Aufgabe der Anwendung ist es, aus einigen externen Stammdatendateien sowie mehreren variablen CSV Zusatzdateien einige Tausend (ca. 3.000) CSV Dateien und einem Gesamtvolumen von ca. 1,3 GB zu erstellen.

- Die variablen CSV Zusatzdateien bilden den Anwender-Input. Sie werden in Excel erstellt und gepflegt.

- Den Output, auch für den Anwender, bilden die ca. 3.000 CSV Dateien sowie eine große Anzahl an Verarbeitungsprotokollen. Die Letzteren werden nach jedem Einzelschritt vom Anwender überprüft, um eventuell die Inputdaten bzw. sonstigen Fehlerzustände zu korrigieren und die Anwendung nochmals zu starten.

Ablauf der Altanwendung

Fachlicher Benutzer



Rechenzentrum

Produktion

1. Input-Dateien in Excel vorbereiten, zum RZ übertragen und den Start der Anwendung beauftragen.
2. Verarbeitungsprotokoll untersuchen:
   - OK: Start des Folgeschrittes beauftragen
   - Sonst: Fehlerursache finden, Daten korrigieren und die bisherige Verarbeitung wiederholen.
3. Der Ablauf endet wenn alle Verarbeitungsschritte erfolgreich abgeschlossen sind und die Ergebnisdaten in die Produktionsanlage geladen wurden.

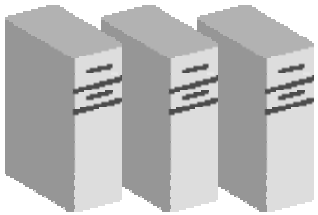
1. Der beauftragte Schritt wird (als Shellskript) gestartet und das Verarbeitungsprotokoll wird zurückgesendet.
2. Im Bedarfsfall: Unterstützung bei der Fehlerbeseitigung.
3. Nach Abschluss des letzten Schritts werden die Ergebnisdaten den Folgeanwendungen (der Produktion) bereitstellt.

1. Die gelieferten Daten werden in die Produktionsanlage geladen und dort bedarfsorientiert transformiert.
2. In diesem Prozess finden auch detaillierte Prüfungen statt.
3. Die hier bemerkten Fehler führen dazu, dass das ganze Prozedere wiederholt werden muss.

Altanwendung: Arbeitsaufwand

Fachlicher Benutzer

- Der Sachbearbeiter hat während des gesamten Arbeitszyklus (ca. 6 Wochen alle 3 Monate) ständig mit der Anwendung zu tun und mindestens zeitweise steht er für keine anderen Aufgaben zur Verfügung.



Rechenzentrum, IT

- Das RZ Personal startet und Überwacht die einzelnen Schritte der Anwendung.
- Bei Fehlern, die nicht auf den Inhalt der Ausgangsdaten zurückzuführen sind, muss u. U. ein Entwickler konsultiert werden.

Produktion

- Der Betriebslenker führt die Ladeoperation durch.
- Im Fehlerfall lädt er die alte Datenversion zurück und meldet den Vorfall.

Probleme der Altanwendung

- Alle diese Verarbeitungsschritte werden im RZ von Hand angesteuert. Auch wenn heute keine Magnetbänder mehr angelegt werden müssen und keine Protokolle per Fax ausgetauscht werden, ist der Betreuungsaufwand extrem hoch.
  - Eine Anwendung mit einem sehr hohen Betreuungsaufwand verursacht die entsprechend hohen Betriebskosten.
- Auch die Pflege und Wartung einer Anwendung dieser Art ist ein Alptraum:
  - Schon die kleinste Änderung der Laufzeitumgebung, z. B. ein neues Betriebssystem-Release, führt dazu, dass alle Skripte und Programme ordentlich durchgecheckt und eventuell angepasst werden müssen.
  - Ein Plattformwechsel verlangt, dass alles vollständig neu kompiliert und gründlich getestet werden muss.
  - In der Regel finden dann auch mehr oder weniger umfangreiche Anpassungen statt (weil z. B. die C Headerfiles anders definiert sind).
  - Mit jeder neuen Anforderung wird das Gesamtsystem komplexer und schwieriger zu beherrschen. Die Nebeneffekte bleiben völlig undurchsichtig.

Stammdaten-Managementsystem

- Mit der Entwicklung und der Einführung des Stammdaten-Managementsystems ist eine neue Situation entstanden:
  - Das Stammdaten-Managementsystem unterhält eine Online-Schnittstelle mit dem Hauptdatenlieferanten des Altsystems.
  - Alle Daten, die das Altsystem bisher nur einige Male im Jahr per Datei bekommen hat sind jetzt – tagesaktuell – in der Stammdaten-DB sofort verfügbar.
  - Auch die fachseitigen Daten sind im neuen System größtenteils schon vorhanden. Sie werden von den Anwendern regelmäßig gepflegt, da das Stammdatenmanagementsystem über eine Intranet-Bedienoberfläche verfügt.

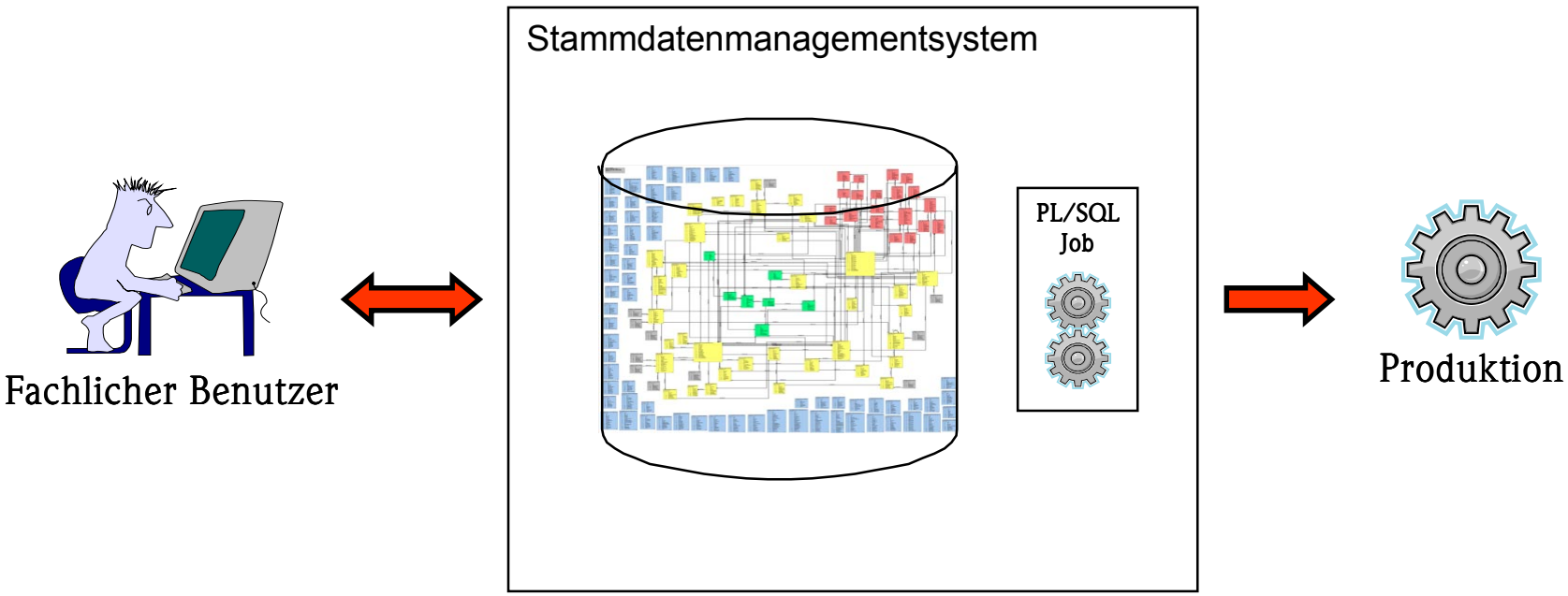
Re-Engineering der Altanwendung

- Die logische Konsequenz war daher, die Funktionalität der Altanwendung in das Stammdaten-Managementsystem zu übertragen.
  - Allerdings wäre es ziemlich absurd zu versuchen, die Altanwendung direkt in das Stammdaten-Managementsystem zu portieren, d. h., aus den Datenbankdaten Dateien zu erstellen um sie nachfolgend mit Hilfe von awk- bzw. C-Programmen zu bearbeiten, etc..
  - Die detaillierte Systemanalyse zeigte zudem, dass es keinen Sinn gemacht hätte, die einzelnen Verarbeitungsschritte der Altanwendung aus den Programmquellen zu wiederholen.
  - Stattdessen wurde entschieden, die alten Vorgehensweisen aufzugeben und für die Entwicklung der Neuanwendung ausschließlich die Vorgabe zu stellen, dass die Ergebnisdaten im Alt- und Neuverfahren identisch sein müssen.
- Die folgenden Folien dokumentieren das Re-Engineerig Verfahren, zeigen also, was und wie es gemacht wurde.

Was wollten wir erreichen?

- Der Gesamtablauf soll wesentlich vereinfacht werden:
  - Der fachliche Benutzer bereitet an der Bedienoberfläche seine Daten vor und startet die Dateiengenerierung.
  - Nach Abschluss der Generierung wird ein Protokoll erstellt (Mengengerüste, bearbeitete Sonderfälle usw.) sowie – im Fehlerfall – ein Fehlerprotokoll.
  - Der Bediener überprüft das Protokoll und gibt die Ergebnisdaten für die Produktion frei.
- Während der Entwicklung wurde noch ein weiteres Ziel gesetzt:
  - Neben den bisher verwendeten CSV Dateien sollten auch Produktionsdaten im Anlagenspezifischen Format bereitgestellt werden, um den Umwandlungsprozess in der Produktionsanlage zu ersparen.

Systemarchitektur



Technische Ziele

- Unser Ziel war es, ein robustes und effizientes Anwendungssystem zu erstellen:
  - Die gesamte Verarbeitung soll innerhalb der Datenbank stattfinden und verarbeitet werden sollen Daten in Datenbanktabellen und keine Dateien.
  - Keine redundante Datenspeicherung; die Redundanzen dürfen erst im letzten Moment, beim schreiben der Ergebnisdateien, entstehen.
  - Die gesamte Programmlogik soll in PL/SQL Packages gepackt werden. Dinge, die in PL/SQL (noch) nicht funktionieren, sollen in Java programmiert und in PL/SQL Programme integriert werden.
  - Der Ablauf soll komplett über die Bedienoberfläche, mit Hilfe von Oracle Job-Queues, gesteuert werden:
    - ohne Shell, SQL*Plus in Shell, die von Cron gestartet werden, etc.

Keine Shellskripte mit SQL*Plus Aufrufen etc. ...

```
sqlplus -s $USER/$PASS@CONN <<ENDSQL >> $LOG_FILE
   WHENEVER SQLERROR EXIT FAILURE
   ... <SQL / PL/SQL Code >
   EXIT;
ENDSQL
STATUS=$?
if [ $STATUS -ne 0 ]
then
   echo "FEHLER: ..." >>$LOG_FILE
   exit $STATUS
fi
...
```

Was macht man, wenn Oracle SQL*Plus doch abschaffen wird?

Sondern Oracle Job Scheduling ...

```
DBMS_SCHEDULER.CREATE_JOB
      ( job_name        => 'XMLSPOOL'
      , job_type        => 'STORED_PROCEDURE'
      , job_action      => 'XMLUSR.xml_gen_all.spool_all_xml' -- Schema.Package.Proc.
      , start_date       => NULL                              -- Sofort
      , repeat_interval => 'FREQ=MINUTELY;INTERVAL=10;'     -- alle 10 min.
      , enabled         => TRUE                              -- Enabled
      , comments        => 'XML-Spooling Prozedur'
      ) ;
```

Was wurde gemacht?

- Die Analyse der Ziele der Anwendung sowie der Beziehungen zwischen den Ergebnis- und Ausgangsdaten führte zu der Spezifikation der Geschäftsregeln des zukünftigen Systems.
  - Dies war die längste und die schwierigste Aufgabe im gesamten Entwicklungsprozess!
- Das Ergebnis dieser Analyse waren, zum einen, eine Erweiterung des Datenmodells des Stammdaten-Managementsystems und, zum anderen, algorithmischen Regeln für die Verarbeitung der betroffenen Daten.
- Allerdings wurden auf Grund der Komplexität der Aufgabe wurden sowohl die Spezifikation als auch das Datenmodell und die Algorithmen während der gesamten Entwicklungsdauer kontinuierlich korrigiert.
- Ein nicht minder herausfordernde Aufgabe war der Gesamttest sowie die Abnahme: Verfahrensbedingt sind die Ergebnisdaten der beiden Anwendungen nicht vollständig identisch, so dass letztendlich alle berechtigten Abweichungen einzeln nachgewiesen werden mussten.

Erweiterungen des Datenmodells

- Das Stammdatenmanagementsystem basiert auf einem streng relationalem Datenmodell mit ca. 150 Tabellen. Bis auf die wenigen Ausnahmen sind alle Daten historisiert.

- Das Datenmodell wurde um mehrere Tabellen erweitert und auch einige der bestehenden Tabellen mussten um neue Attribute erweitert werden.

- Letztendlich zeigte sich aber, dass es unmöglich ist, alle notwendigen Daten sinngemäß in einem relationalen Datenmodell zu unterzubringen:

  - Es wurde notwendig, eine „schmutzige Ecke“ mit ca. 5 nicht in das Datenmodell eingebundenen Tabellen zu zulassen. Diese Daten passten nirgends logisch mit anderen zusammen.

  - Eine davon speichert Daten, die jedem Modellierungsversuch wiederstreben und am Ende der Verarbeitung in das Ergebnis beigemengt werden.

  - Glücklicherweise sind das alles in allem nur wenige und relativ stabile Datensätze, so dass ihre Pflege unproblematisch ist.

Ablauf der Anwendung: Datenaufbereitung

- Die Ausgangsdaten sind historisiert, d. h., der Basissatz hat immer den aktuellen Stand, die Änderungen sowie die Löschungen sind in Historietabellen untergebracht.
- Gemäß den Vorgaben war es nötig, einen konsistenten Stand zu einem definierten Zeitpunkt in der Vergangenheit zu verarbeiten.
- Der Aufbau dieses konsistenten Bestandes ist der erste Schritt der Anwendung. Dies geschieht mit Hilfe eines PL/SQL Packages „TMP_DATEN“ mit ca. 3.000 Codezeilen.
  - Mit dessen Hilfe wird ein konsistenter Datenbestand von ca. 30 Tabellen gebildet.
  - Sofern zweckmäßig, werden BULK Operationen verwendet.
  - Die historischen Daten werden mit Hilfe von MERGE „eingemerged“.

Ablauf der Anwendung: algorithmische Verarbeitung

- Im zweiten Schritt wurden die temporären Daten Schrittweise verarbeitet mit dem Ziel, alle spezifizierten Bedingungen nach und nach zu erfüllen.
- Diese Verarbeitung besteht im Wesentlichen daraus, dass mit Hilfe der programmatisch manipulierten DDL Constraints sowie prozeduraler Verarbeitung der Ausgangsbestand in den gewünschten Ergebnisbestand transformiert wurde.
- Zuletzt wurden die restlichen fachseitigen Zusatzdaten hinzugefügt und dem gleichen Prozedere unterzogen.
  - Als Ergebnis entstand eine einzige Tabelle die alle Ergebnisdaten (nichtredundant) speichert.
  - Unter „nichtredundant“ wird hier verstanden, dass ein Datensatz, der in alle Dateien gehen soll, ist dort nur ein mal enthalten ist.
- Diese Verarbeitung sichert ein PL/SQL Package mit ca. 4.500 Codezeilen sowie 5 Hilfstabellen.

Ablauf der Anwendung: Dateienerstellung

- Zuletzt werden alle Arten der Ergebnisdateien erstellt. Dies geschieht mit Hilfe des dritten PL/SQL Packages (ca. 1.600 Codezeilen).
- Für die Verzeichnismanipulationen auf Bettriebssystemebene (Verzeichnis anlegen bzw. löschen, Verzeichnisbaum auflisten bzw. löschen) wurde ein Java Package in PL/SQL verwendet.
- Auch die Erstellung der ZIP-Archive wurde mit Hilfe eines Java Packages realisiert. Auch diese Funktionen wurden direkt aus PL/SQL aufgerufen.
- Die Anwendung ist so konfiguriert, dass – abhängig von den Einstellungen an der Bedienoberfläche – entweder alle Zwischendateien produziert werden oder nur das Endergebnis. Der Bediener kann auch entscheiden, ob er die im früheren Lauf erstellten Dateien beibehalten oder löschen möchte.

Anwendung: Ergebnis

- Die neue Anwendung produziert alles in allem die folgenden Datenmengen:
  - Mehrere ZIP-Archive mit Gesamtdatenvolumen von ca. 450 MB.
    - Eine Archivdatei enthält 4 Dateien mir einem unkomprimierten Volumen von ca. 150 MB.
  - Über 3000 CSV Dateien in mehreren Verzeichnissen mit einem Gesamtvolumen von 1,3 GB.
  - Knapp 40 weitere Hilfsdateien, Verarbeitungsprotokolle usw. mit einem Gesamtvolumen von ca. 20 MB oder mehr (falls ein Trace-Log geführt wird).
- Alle diese Daten stammen aus einer einzigen Tabelle mit ca. 1,4 Mio. Zeilen und einem Datenvolumen von etwa 80 MB.
- Die Gesamtlaufzeit der Anwendung auf einem einfachen Notebook beträgt, in Abhängigkeit von den gewählten Optionen, 2 bis 2,5 Stunden.

Schlussfolgerungen

- Eine Anwendung, die als ein Workflow aus Shell, SQL*Loader, awk und C Programmen, im Rechenzentrum ca. 6 Wochen Zeit brauchte, ist jetzt auf einem Notebook in 2 bis 2,5 Stunden fertig.

- Die Anwendung besitzt jetzt eine Intranet-Bedienoberfläche die die komplizierte Excel-Verarbeitung ersetzt.

- Für den Betrieb der Anwendung ist keine Hilfe des administrativen RZ Personals notwendig: der Anwender editiert seine Daten und startet die Verarbeitung – alles direkt auf der Bedienoberfläche.

- Die Wartung des Systems wurde grundsätzlich vereinfacht:

  - keine Abhängigkeit von der Betriebssystemplattform mehr.
  - eine PL/SQL Anwendung läuft überall dort, wo Oracle vorhanden ist.
  - auch eine Oracle-Migration ist für eine PL/SQL Anwendung i.d.R. unproblematisch.

Fazit

- Es lohnt sich, über ein Re-Engineeering von Altanwendungen nachzudenken.
- Insbesondere die Übernahme einer Anwendung in die Oracle Datenbank ist, wie unser Beispiel zeigt, eine sehr lohnende Investition und kann zu einem traumhaften Return-On-Investment führen.