Nur Feiglinge testen! Testen von SAS Batch Programmen mit MS-Excel

Frank Hanke
viadee Unternehmensberatung GmbH
Anton-Bruchausen-Straße 8
48147 Münster
Frank.Hanke@viadee.de

Zusammenfassung

Nur Feiglinge testen - aber, wer testet spart meist viel Geld. Da ist man doch gerne ein wenig „Feigling“! In der praktischen Entwicklung von SAS-Programmen erstellt der Entwickler anhand eines Pflichtenheftes den SAS-Ablauf. Dabei erzeugt er sich für seine internen Tests Datenbestände, anhand derer er seine Verarbeitung testen kann. Meist handelt es sich hierbei um produktionsnahe Daten, die die meisten Datenkombinationen enthalten. Läuft das Programm fehlerfrei durch, ist der Entwicklertest abgeschlossen.
Aber gerade die Sonderfälle in Datenbeständen und die Ausreißer werden bei dieser Methode schnell übersehen und nicht getestet. Nutzt man einen produktionsnahen Bestand, sind die Ausreißer nur selten vorhanden. Diese im Test zu identifizieren und zu analysieren ist aufwändig und nicht immer trivial.
In den folgenden Kapiteln wird gezeigt, wie sich SAS-Programme systematisch mit synthetischen Testdaten testen lassen. Dabei werden Eingabe-, Ausgabe- und temporäre Daten automatisch dokumentiert, so dass sich jederzeit der Verarbeitungsweg nachvollziehen lässt. Durch die synthetischen Datenkombinationen können spezielle Testfälle entwickelt werden, die die Verarbeitung gerade in Sondersituationen testen und überprüfen. Dabei liegt der Schwerpunkt darauf, die Testdatenverwaltung so einfach wie möglich zu gestalten, damit diese Aufgabe auch von Fachbereichen oder Testcentern durchgeführt werden kann. Microsoft Excel wird als Tool zur Verwaltung von Testdatenbeständen genutzt. Über die SAS Integration Technologies werden Datenstrukturen synchronisiert, Daten in SAS-Bibliotheken geladen und die Programmausführung vorgenommen und dokumentiert. Auch das SAS Log wird im gleichen Zuge auf Fehler und Hinweise gescannt. Fehlersituationen können einfach reproduziert und das Ergebnis von Programmkorrekturen verifiziert werden.

Schlüsselwörter: Batchprogramm, MS-Excel, Referenztest, Funktionstest

1 Übersicht

Das eingesetzte Excel-Werkzeug wurde so implementiert, dass es sich in die vorhandene Infrastruktur einfügt und die etablierten Prozesse bzw. Vorgehensweisen beim Testen unterstützt. Um später einordnen zu können, warum gewisse Dinge so und nicht anders umgesetzt wurden, zeigt dieses Kapitel einen kurzen Überblick des Umfeldes sowie der vorhandenen Methodik beim Testen und erläutert einige Begriffe, welche sich in den Tabellenblättern wieder finden.

1.1 Umfeld – Was wird, wann getestet

Entwickelt und getestet werden SAS-Programme zum Aufbau eines DataWareHouses (DWH). Alle erforderlichen Eingangsdaten stehen als SAS-Tabellen zur Verfügung oder können über SAS/Access Schnittstellen verarbeitet werden. Es erfolgt kein Zugriff auf externe Bestände wie CSV Files oder seq. Files. Diese Programme werden in einer Verarbeitungskette (täglich, monatlich, wöchentlich, quartalsweise oder jährlich) automatisch ausgeführt, wobei Abhängigkeiten untereinander berücksichtigt werden.
Diese Programme sind nicht statisch und unterliegen den gleichen Veränderungen, wie in anderen Unternehmen auch. Kreative Fachseiten, technische Änderungen, oder gar der Gesetzgeber versorgen die Entwickler / Tester mit neuen An- und Herausforderungen. Innerhalb eines Release-Zyklus schließt sich nach der Programm-Entwicklung (inkl. Entwicklertest) die Phase Funktionstest an. Der Funktionstest ist eine Mischung aus Black & White Box Test, also ein erweiterter Modultest.

1.2 Vorgehensweise im Funktionstest

Die Herangehensweise beim Test von ganz neuen Programmen und geänderten Programmteilen unterscheiden sich lediglich in ein oder zwei Zwischenschritten. Alle Testergebnisse sind in einer vorgegebenen Verzeichnisstruktur auf einem Gruppenlaufwerk zu finden.

Die Ausgabe- und temporären Dateien der letzten / jüngsten Testausführung, bilden die Vergleichsgrundlage für den aktuellen Funktionstest. Häufig erfolgt vor dem Test der Programmänderung ein Programmdurchlauf mit unveränderten Programmquellen, um so sicherzustellen, dass die gesicherten Daten der letzten Testausführung noch aktuell und stimmig sind. Treten hier bereits Unterschiede auf, so können diese unabhängig von der Programmänderung im Vorfeld analysiert werden. Neue Testfälle werden dokumentiert bzw. fortgeschrieben; für komplexe Sachverhalte erfolgt eine tabellarische Testdatenermittlung, um bestimmte Grenzfälle im Funktionstest zu überprüfen.

Nach der Erfassung der Testdaten erfolgt die Testausführung. Die Ausgaben werden mit dem letzten vorhandenen Lauf mittels eines Vergleichstools [1] verglichen. Abweichungen werden dokumentiert und anhand der Programmänderungen erklärt. Alle Ergebnisse des Testlaufs (Ein- / Ausgabe, Logfiles, Testdokumentation usw.) werden in das Anforderungstool eingestellt und unveränderlich archiviert.

1.3 Begriffe

- Testobjekt
  In der Regel stellt das Testobjekt eine 1:1 Beziehung zu einem Programm dar. In seltenen Fällen kann das Testobjekt auch eine komplexe Bearbeitungskette abbilden.

[1] z.B. Beyond Compare der Firma ScooterSoftware (http://www.scootersoftware.com)

- Testlauf
  Unterschiedliche Testläufe bilden verschiedene Konstellation ab. Programme müssen beispielsweise andere Berechnungen / Verdichtungen für unterschiedliche Mandanten liefern oder es erfolgt ein bestimmter Andruck auf Nachweislisten, wenn keine Daten vorhanden sind.
- Testfall / Testdatenkombination (TDK): wird zur Strukturierung / Gruppierung und Organisation von Testdaten benutzt.
  Im Testframework ist dies als eine Dezimalzahl mit drei Vorkomma- (bezeichnet den Testfall) und zwei Nachkommastellen (bezeichnet die Testdatenkombination innerhalb des Testfalls) definiert. Mit der TDK werden in der Regel Schlüsselbeziehungen voneinander abhängigen Tabellen abgebildet bzw. berechnet.
- Technisches Teilprodukt
  Das Teilprodukt entspricht der Ausführungsumgebung, auf der das Programm ausgeführt werden soll. Damit wird die Testausführung von ganzen Programmketten (ETL über Veredelung bis zum Reporting) unterstützt.
- Scanlog
  Neben Fehlern und Warnungen, gibt es zahlreiche Hinweise, die auf mögliche Programmfehler deuten. Dazu wird das SAS-Log auf bestimmte Schlagwörter bzw. Kombinationen untersucht. Beispielmeldungen sind Kartesische Produkte oder Umwandlung von Character in numerische Werte.

2 Toolunterstützung beim Testen - Erwartungen

Welche Erwartungen werden an das „neue“ Tool gestellt?
Im Grunde ganz einfach, es soll genau so funktionieren, wie das alte Werkzeug, nur viel bunter, schöner, schneller, einfacher und komfortabler. Am besten mit einem großen grünen Button „Test“, der alles per Knopfdruck erledigt, also die Eierlegende Wollmilchsau[2].

Letztendlich geht es darum, Testdaten strukturiert und übersichtlich zu verwalten bzw. zu organisieren. Testläufe sollen jederzeit per Knopfdruck ausführbar sein und die gleichen Ergebnisse wie bei der letzten Ausführung liefern. Die entsprechenden Ein- / Ausgabedaten sowie Logfiles sollen automatisiert abgelegt werden. Da der Funktionstest i.d.R. durch einen größeren Personenkreis mit wechselnder Zuordnung erfolgt, ist es wichtig, dass Dritte sich schnell in einem Testobjekt zurechtfinden, wodurch letztendlich auch der administrative Aufwand verringert wird.

Das Tool soll die vielen kleinen, immer wiederkehrenden Schritte automatisieren, ohne dass der Benutzer manuell eingreifen muss. Der Anwender möchte seine Testdaten einfach und komfortabel erfassen, die Daten automatisiert in die erforderlichen Tabellen laden, den Programmablauf starten und die Ergebnisse auswerten und dokumentieren.

[2] Siehe http://de.wikipedia.org/wiki/Eierlegende_Wollmilchsau

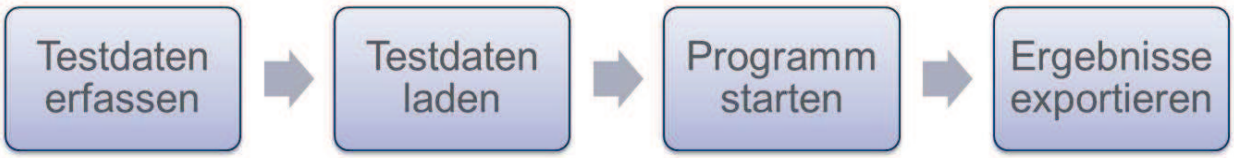


Abbildung 1: Ablauf

3 Vorstellung des Excel-Testframeworks

In den folgenden Unterkapiteln wird das eingesetzte Excel-Framework mit seinen wichtigsten Funktionen vorgestellt.

3.1 Architekturübersicht

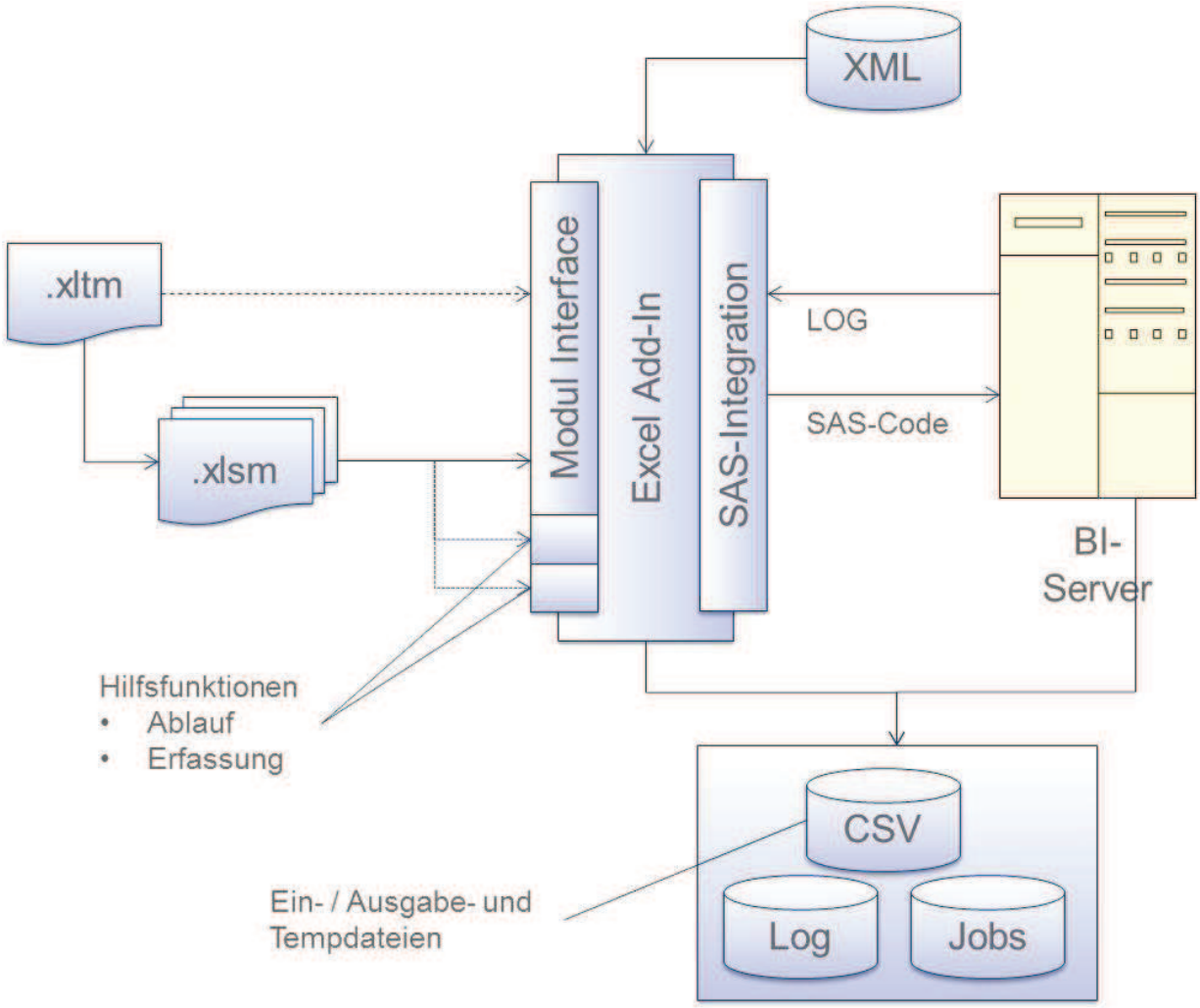


Abbildung 2: Architektur

Die Benutzeroberfläche ist als Excel-Vorlage mit Makros implementiert und wird jeweils vom Anwender instanziiert. Sämtliche Funktionalität, einschließlich des Eventmanagements ist in **ein** Excel-Addin ausgelagert. Innerhalb der Arbeitsmappe existieren nur noch rudimentäre Funktionen wie Automakros beim Öffnen bzw. Schließen einer

Arbeitsmappe. Hilfsfunktionen für die Definition / Erstellung des Programmablaufs und der Testdatenerfassung sind in zwei weiteren Funktionsmodulen gekapselt.
Die Anbindung an den SAS-Server erfolgt über SAS Integration Technologies. Die Anmeldedaten für die jeweilige Serverschicht sind in XML-Dateien auf einem Gruppenlaufwerk hinterlegt, so dass der Test durch jeden beteiligten Tester unter denselben Zugriffsbedingungen erfolgt. Die Excel-Arbeitsmappe dient letztendlich als Code-Generater, der alle Daten und Aktionen aus dem Testframework in SAS BASE-Code überführt. Dieser wird dann zum Server gesendet und dort ausgeführt. Die Ergebnisdaten werden entweder vom Server direkt (CSV-Dateien) oder dem Add-In (Log und Jobs) in die zentrale Verzeichnisstruktur exportiert.

3.2 Übersicht des Funktionsumfangs

- Definition unterschiedlicher Testläufe
  - Verschiedene Testläufe bilden unterschiedliche Datenkonstellationen ab. Beispielsweise gewollte Abbruchbedingungen.
- Import Tabellenstrukturen vom Server
  - Die für das Programm definierten Tabellenstrukturen vom SAS-Server in ein Tabellenblatt einlesen.
- Protokollierung von Strukturänderungen
  - Veränderungen gegenüber dem letzten Strukturimport, werden auf einem separaten Tabellenblatt dargestellt.
- Festlegen von Ablaufketten
  - Im Tabellenblatt „Ablauf“ können ganze Programmketten definiert werden. Die Programme werden auf der jeweiligen Schicht (technisches Teilprodukt) ausgeführt.
- Eingabeblätter generieren und Testdaten erfassen
  - Für alle als Eingabe definierten Tabellen, werden die Eingabeblätter mit den definierten Spalten erzeugt.
- Import CSV-Dateien
  - Testdaten können aus CSV-Datei eingelesen werden. Es müssen folgende Bedingungen erfüllt sein:
    - Unformatierte Daten (bezogen auf SAS-Formate)
    - Dateiname muss sich aus SASLIB und Tabelle zusammensetzen. <SASLIB>.<Tabelle>.csv
- Übersicht der Zuordnung von Tabelle / Testlauf / TDK
  - Aus dem Übersichtsblatt kann zu dem jeweiligen Testdatensatz einer Tabelle gesprungen werden (Hyperlink)
- Kopieren von TDKs
  - Um Schlüsselbeziehungen von Tabellen abzubilden, werden in den jeweiligen Tabellen die gleichen TDK-Nummern verwendet (Berechnung der Schlüssel anhand dieser Nummer). Diese TDK können per Knopfdruck als neue TDK kopiert werden, so dass ein weiterer Datensatz zur Verfügung steht.

- Hilfsfunktionen für Programmablauf / Testdatenerfassung
  - Für die Definition des Programmablaufes stehen dem Benutzer Hilfsfunktionen zur Verfügung, die über den Excel-Funktionswizzard aufgerufen werden können. Diese Funktionen liefern den notwendigen SAS-Code, ohne dass sich der Benutzer darum kümmern muss.
- Navigation innerhalb der Excel Mappe
  - Innerhalb der Tabellenblätter ist die Navigation via Hyperlinks realisiert.
- Kopieren in eine Referenzumgebung
  - Tabellenstrukturen können nach Programmlauf, wenn die Struktur durch das Programm geändert wurde, in Referenzumgebung kopiert werden.
- Migration – SAS-Libs / Tabellen- bzw. Spaltennamen werden geändert
  - Damit vorhandene Testdaten erhalten bleiben, können über ein spezielles Tabellenblatt die alten und neuen Bibliotheks- und Tabellennamen erfasst werden. Zusätzlich kann eine Übersetzungstabelle der Spaltennamen eingelesen werden. Per Knopfdruck erfolgt eine Übersetzung aller Namen auf den entsprechenden Tabellenblättern.

3.3 Ablauf / Methodik

Hier werden die üblichen Schritte für die Erstellung eines Testobjektes vorgestellt.

3.3.1 Erfassen der Ein- / Ausgabetabellen

Zunächst müssen die für den Test des Programms erforderlichen Tabellen im Testframework definiert werden (Darstellung erfolgt wegen der Breite zweigeteilt Abbildung 3 und 4). Ob es sich um eine Eingabe- bzw. Ausgabetabelle handelt, wird über den Eintrag in der Spalte „Laden=JA“ festgelegt. Über die Spalten CSV-Vorher / CSV-Nachher wird definiert, ob ein Export vor- oder nach Programmausführung testlaufabhängig erfolgen soll. Die Spalte „Worksheet“ wird automatisch befüllt. Eingabetabellen können hier per Doppelklick geöffnet werden. Ob eine Tabelle wieder in die Referenzumgebung zurückkopiert werden soll, wird über die Spalte „Export Ref“ gesteuert.

A	B	C	D	E	F	G
Testobjekt	PROG100	PR	12345-67	Tech. TP	Testlauf	
Tester	HaF	fachliches Teilprodukt	VI.KSFE2015			
Release	12015			VIADEE	R01	CSV-Impor

SAS Strukturimport | Defaultwerte bearbeiten... | Parameter aktualisieren | Generieren Testdaten-Sheets

Tabellen

TechTP	SASLIB	Tabelle	Worksheet	Löschen	Laden	CSV Vorher
VIADEE	SASUSER	CLASS	SASUSER.CLASS_011	JA	JA	R01
VIADEE	SASUSER	CLASS1	SASUSER.CLASS1_012	JA	JA	R01
VIADEE	SASUSER	CLASS3	SASUSER.CLASS3_013	JA	JA	R01,R03
VIADEE	SASUSER	CLASS4	SASUSER.CLASS4_014	NEIN	NEIN	NEIN

Abbildung 3: Eingabeblatt - Basis - Teil I

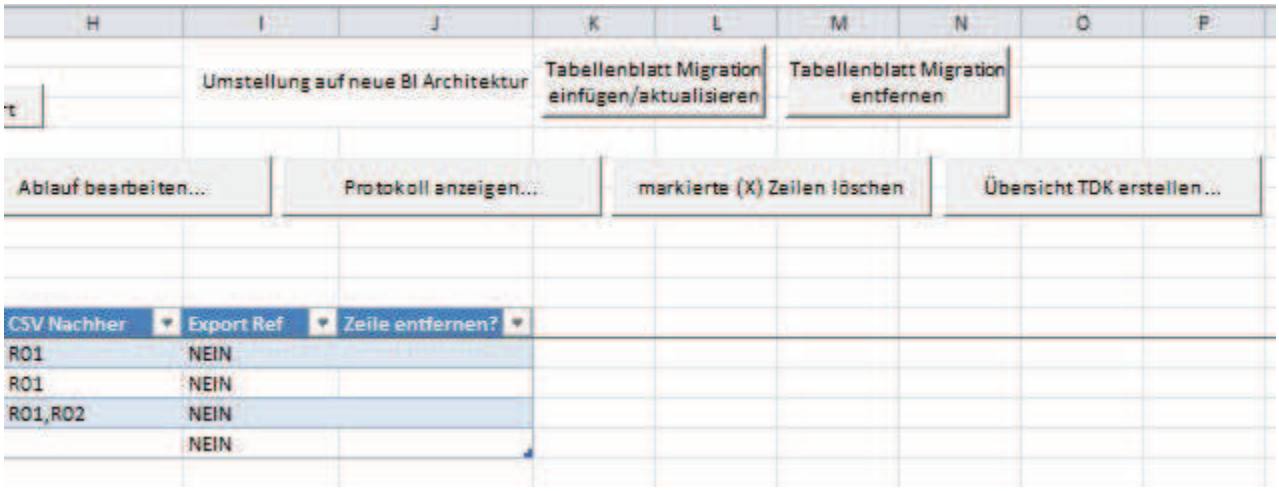


Abbildung 4: Eingabeblatt - Basis - Teil II

3.3.2 Festlegen der Spalten und Standard-Werte

Nicht immer werden alle Spalten einer Tabelle benötigt. Damit die Eingabeblätter übersichtlich bleiben, kann der Anwender im folgenden Schritt die für das Programm bzw. den Test relevanten Spalten festlegen (wegen der Breite erfolgt die Darstellung zweigeteilt in Abbildung 5 und 6 auf den nächsten Seiten). Die Navigation innerhalb des Tabellenblatts erfolgt über die Combo-Box, in der alle Tabellen alphabetisch sortiert gelistet sind.
Die Spalten A bis H (außer B) werden über den Import vom SAS-Server geliefert. Sofern Beschriftungen vorhanden sind, werden diese in der Spalte „Label“ dargestellt. Ob eine Spalte im Testdatenblatt dargestellt werden soll, wird über die Spalte „Relevant?“ gesteuert. Zusätzlich kann in der Spalte „Default-Wert“ ein Standard-Wert eingetragen werden, der für alle Testdaten dieser Tabelle gilt, sofern dieser nicht überschrieben wird. Spalten ohne Relevanz und Default-Wert, werden in den SQL-Anweisungen nicht berücksichtigt.
Die Spalte „Status“ zeigt an, ob Spalten neu hinzugekommen sind. Weggefallene Spalten sind im Protokollblatt zu finden. Ein erneuter Import ersetzt „Neu“ durch ein „X“.
Hinweis: Standard-Werte werden auch ohne Relevanz berücksichtigt.

3.3.3 Importprotokoll

Nach einem Strukturimport, Migration oder CSV-Import werden die Veränderungen an den Tabellen protokolliert. Der Anwender hat eine Übersicht, ob wichtige Spalten womöglich mit Standard-Werten belegt, entfallen sind. Im Fall eine Migration (Bibliotheks- / Tabellen- / Spaltennamenänderungen) wird der neue Spaltenname ebenfalls ausgegeben. Der aktuellste Eintrag wird immer oben dargestellt.

	A	B	C	D	E	F	G	H
1	SASUSER.CLASS	STATUS	TYP	LENGTH	FORMAT	FD	FL	LABEL
2								
3	Tabelle SASUSER.CLASS							
4	AGE	X	1	8		0	0	
5	HEIGHT	X	1	8		0	0	
6	NAME	X	2	8		0	0	
7	SEX	X	2	1		0	0	
8	WEIGHT	X	1	8		0	0	
9								
10	Tabelle SASUSER.CLASS1							
11	AGE	X	1	8		0	0	
12	HEIGHT	X	1	8		0	0	
13	NAME	X	2	8		0	0	
14	SEX	X	2	1		0	0	
15	WEIGHT	X	1	8		0	0	
16								
17								
18	Tabelle SASUSER.CLASS3							
19	AGE	NEU	1	8		0	0	
20	HEIGHT	NEU	1	8		0	0	
21	NAME	NEU	2	8		0	0	
22	SEX	NEU	2	1		0	0	
23	WEIGHT	NEU	1	8		0	0	
24								

Abbildung 5: Spalten und Standard-Werte - Teil I

H	I	J	K	L
	Relevant?	Default-Wert	zur Hauptseite...	
	Tabelle	Tabelle		
	X	35		
	X			
	X			
	X			
	X	80		
	Tabelle	Tabelle		
	X	40		
	X			
	X			
	X			
	X	80		
	Tabelle	Tabelle		
		20		
	X			
	X			
	X			
	X			

Abbildung 6: Spalten und Standard-Werte - Teil II

Zeitstempel	Release	PR-Nummer	Tabelle	Spalte	Status	Relevanz	Default	neue Spalte
SAS-IMP-01.03.2015 13:45:03	37100	33567-23	SASUSER.CLASS3	HEIGHT	NEU	FALSCH		
				Hoehe	DEL	WAHR	3	
SAS-IMP-27.02.2015 08:11:48	37100	33567-23	SASUSER.CLASS3	AGE	NEU	FALSCH		
				HEIGHT	NEU	FALSCH		
				NAME	NEU	FALSCH		
				SEX	NEU	FALSCH		
				WEIGHT	NEU	FALSCH		
SAS-IMP-26.02.2015 16:30:36	37100	33567-23	SASUSER.CLASS	AGE	NEU	FALSCH		
				HEIGHT	NEU	FALSCH		
				NAME	NEU	FALSCH		
				SEX	NEU	FALSCH		
				WEIGHT	NEU	FALSCH		
			SASUSER.CLASS1	AGE	NEU	FALSCH		
				HEIGHT	NEU	FALSCH		
				NAME	NEU	FALSCH		
				SEX	NEU	FALSCH		
				WEIGHT	NEU	FALSCH		

Abbildung 7: Spalten- und Standardwerte - Teil III

3.3.4 Eingabeblätter erzeugen

Nach der Definition aller Spalten erzeugt der Anwender über die Hauptmaske die leeren Tabellenblätter für Eingabe und erfasst die Testdaten. In diesem Beispiel für die Tabellen Class, Class1 und Class3. Class4 ist aus der Ausgabetabelle definiert.

F3 ⨍ₓ ="N_" & TEXT(ZZTF(A3);"000") &"-" & TEXT(ZZTDK(A3);"00")

	A	B	C	D	E	F	G	H
1	[Basis]	[Übersicht]	Defaults	35				80
2	TDK	Lfd-Nr.	Testläufe	AGE	HEIGHT	NAME	SEX	WEIGHT
3	001,01	1	R01	40	1	N_001-01	W	75
4	001,02	2	R01,R02	42	1	N_001-02	W	
5	002,03	3		$MISSING	1	N_002-03	W	
6	003,15	4	R01,R03		1	N_003-15	W	63

Abbildung 8: Tabelle class mit Testdaten

	A	B	C	D	E	F	G	H
1	[Basis]	[Übersicht]	Defaults	40				80
2	TDK	Lfd-Nr.	Testläufe	AGE	HEIGHT	NAME	SEX	WEIGHT
3	001,01	1	R01		2	N_001-01	X	
4	001,02	2	R01,R02		2	N_001-02	X	
5	002,03	3			2	N_002-03	X	85
6	003,15	4	R01,R03		2	N_003-15	X	

Abbildung 9: Tabelle class1 mit Testdaten

A	B	C	D	E	F	G
[Basis]	[Übersicht]	Defaults				
TDK	Lfd-Nr.	Testläufe	HEIGHT	NAME	SEX	WEIGHT

Abbildung 10: Tabelle class3 noch ohne Testdaten

3.3.5 Übersichtsblatt

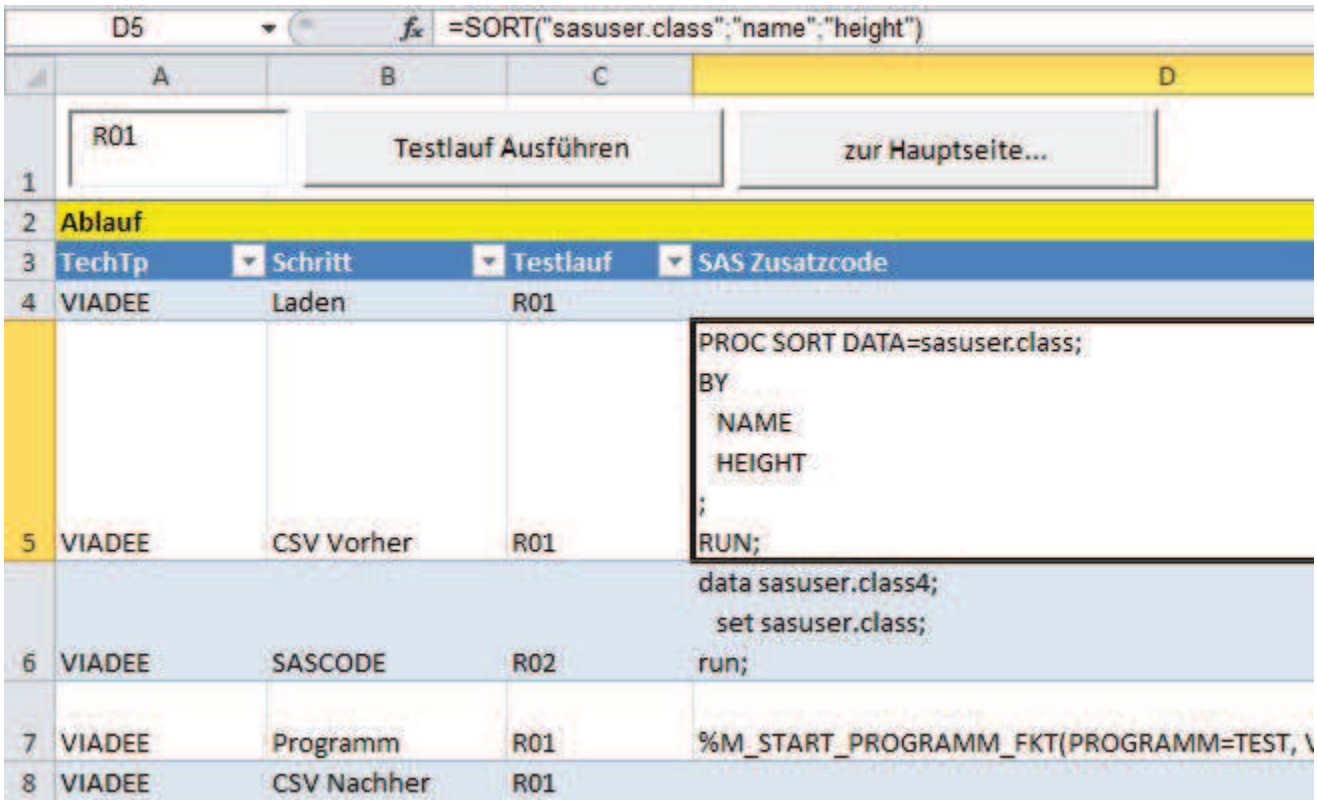
Das Tabellenblatt liefert eine Übersicht der definierten TDKs und in welchen Testläufen dieser verwendet werden. Über die Hyperlinks kann der Anwender direkt an die Stelle in der Tabelle springen. Wurden zu einer Tabelle keine Daten erfasst, wird dies in der Übersicht angezeigt und kann somit schnell geprüft werden.

A	B	C	D	E	F	G
[Basis]		Tabellen				
copy	SASUSER.CLASS	SASUSER.CLASS1	SASUSER.CLASS3			
TDK						
leer			!NODATA!			
001,01	R01	R01				
001,02	R01,R02	R01,R02				
002,03	Alle	Alle				
003,15	R01,R03	R01,R03				

Abbildung 11: Übersicht Testdaten

3.3.6 Festlegen des Programmablauf und Ausführung

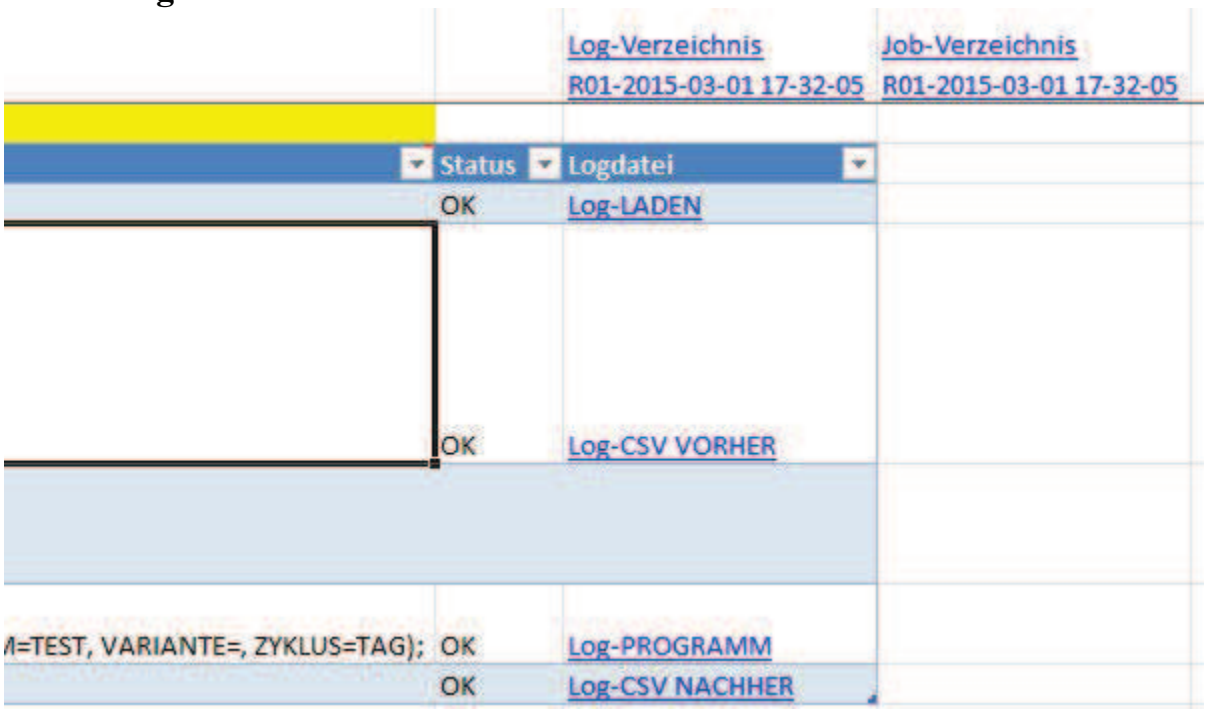
In den folgenden Abbildungen 11 und 12 legt der Anwender den Ablauf fest. Zur Unterstützung stehen dem Benutzer einige Hilfsfunktionen zur Verfügung, die über den Funktionswizzard aufgerufen werden können und SAS-Code zurückliefern (Beispiel Sort). In der Spalte „SAS Zusatzcode“ kann gültiger SAS-Code eingegeben werden, der **vor** dem Schritt ausgeführt wird. Jeder Schritt wird in einer eigenen SAS-Session ausgeführt und schreibt sein eigenes Protokoll. Fehlermeldung oder Warnungen werden in Kurzform in der Spalte „Status“ ausgegeben. Über die Hyperlinks kann das Logfile direkt geöffnet werden.



D5 =SORT("sasuser.class","name","height")

	A	B	C	D
1	R01	Testlauf Ausführen		zur Hauptseite...
2	Ablauf			
3	TechTp	Schritt	Testlauf	SAS Zusatzcode
4	VIADEE	Laden	R01	
5	VIADEE	CSV Vorher	R01	PROC SORT DATA=sasuser.class; BY NAME HEIGHT ; RUN;
6	VIADEE	SASCODE	R02	data sasuser.class4; set sasuser.class; run;
7	VIADEE	Programm	R01	%M_START_PROGRAMM_FKT(PROGRAMM=TEST, V
8	VIADEE	CSV Nachher	R01	

Abbildung 12: Ablauf - Teil I



		Log-Verzeichnis	Job-Verzeichnis
		R01-2015-03-01 17-32-05	R01-2015-03-01 17-32-05
	Status	Logdatei	
	OK	Log-LADEN	
	OK	Log-CSV VORHER	
M=TEST, VARIANTE=, ZYKLUS=TAG);	OK	Log-PROGRAMM	
	OK	Log-CSV NACHHER	

Abbildung 13: Ablauf - Teil II

4 Fallstricke

Bei der Umstellung der vorhandenen Testobjekte und im Laufe des Pilotbetriebes sind einige Punkte aufgefallen, die man bei der Implementierung so nicht erwartet hat:

- Kalender-Excel-SAS:
  In den vorhandenen Testdaten (SAS Tabellen Export) beziehen sich viele Datumswerte auf den 01.01.1900. Excel kann Datumswerte aber erst ab dem 03.01.1900 richtig darstellen. Beim CSV-Import wird geprüft, ob der SAS-Wert kleiner als -21855 ist. In diesem Fall wird das eingelesene Datum auf den Wert -21855 gesetzt.
- 31.12.9999:
  Ist ein sehr wichtiges Datum in den Daten. Dieser Wert kennzeichnet ein offenes Konto. Diese Werte wurden bei der Umrechnung zwischen SAS und Excel permanent falsch dargestellt. Weil dieses Datum so wichtig ist, wird hier der maximale SAS-Wert fix gesetzt (gilt auch für den Typ DateTime).
- Dezimalpunkte:
  Beim Erstellen der SQL-Anweisungen musste das Dezimal-Trennzeichen in einen Punkt umgewandelt werden. Damit Excel beim Import Dezimalzahlen nicht als Datum darstellt, musste hier der Dezimalpunkt in den CSV-Dateien durch ein Komma ersetzt werden.

5 Fazit - Test mit Excel

Mit Excel, VBA und SAS Integration Technologies lässt sich ein schlankes Werkzeug für die Testdatenverwaltung und Testdurchführung mit moderatem Aufwand erstellen. Normalerweise steht Excel auf nahezu allen Rechnern eines Unternehmens zur Verfügung. Visual Basic for Applications, kurz VBA, ist leicht erlernbar und im Internet sehr gut dokumentiert. Das Rad muss folglich nicht zwingend neu erfunden werden.

Ein großer Vorteil ist, dass eine Anwendung auf der Basis von Excel / Access ganz genau auf die Methoden und Prozesse im Unternehmen zu geschnitten werden kann. Tools von der „Stange“, wie SQS-Test, decken natürlich ein breiteres Spektrum an Funktionalitäten ab, sind aber in Lizenzkosten und Unterhalt wesentlich teurer. Auch ist fraglich, was genau von diesem Funktionsumfang letztendlich genutzt wird.

Erfahrungsgemäß ist für derartige Tools Spezialwissen notwendig, wenn es ums „Customizing“ geht. Hier hilft das Internet meist auch nicht weiter.