

**VON DEN REQUIREMENTS  
ZUM TARGET CODE...**

**REQUIREMENTS**

**ANALYSIS**

**TOOL**

**SYSTEM**

**DESIGN**

**TOOLKIT**

**...TRACEABILITY  
ÜBER ALLE EBENEN**

Copyright © 2007 Stefan Sachs

UML ist ein eingetragenes Warenzeichen von Object Management Group, Inc. KORG ist ein eingetragenes Warenzeichen von Korg Inc. Visual Age ist ein eingetragenes Warenzeichen von IBM Corp. PARTS ist ein eingetragenes Warenzeichen von Digitalk Inc. Bean Builder ist ein eingetragenes Warenzeichen von Sun Microsystems, Inc.

Convert ist ein Produkt von Jesus Villena.

bmp2png ist ein Produkt von Myasaka Masaru.

Alle übrigen Produkte und Markennamen, die in dieser Broschüre genannt werden, sind eingetragene Warenzeichen der entsprechenden Rechtsinhaber.

Die Abbildung eines KORG Chromatic Tuner CA-20 erfolgt mit freundlicher Genehmigung von Korg Inc. Es sei an diese Stelle ausdrücklich darauf hingewiesen, daß das hier vorgestellte Entwurfswerkzeug nicht bei der Entwicklung dieses Geräts verwendet wurde.

# Warum Anforderungsanalysen?

Es gibt gute Gründe, die mühsame Arbeit eine Sammlung und Dokumentation aller bekannten Anforderungen an ein neues Produkt auf sich zu nehmen:

- Eine vollständige Anforderungsanalyse stellt sicher, daß das Ergebnis Ihrer Entwicklungsarbeit den Kundenwünschen entspricht
- Dokumentierte Anforderungen erleichtern eine nachhaltige Produktpflege
- Zulassungsbehörden fordern den durchgängigen Nachweis der Verfolgbarkeit von Anforderungen bis in den Code (Traceability)

**Anforderungsanalysen helfen Ihnen, mit Ihren Entwicklungen die Kundenwünsche zu erfüllen.**

**Anforderungsanalysen werden von Zulassungsbehörden vorgeschrieben**

## Anforderungen und Abstraktionsebenen

Die Formulierung von Anforderungen ist keine einfache Aufgabe. Die Beschreibung der Eigenschaften eines noch nicht existierenden Produkts stellt eine Gratwanderung zwischen phantasievoll kreativem Wunschdenken und realitätsbezogener Erbsenzählerei dar. Wenn dann noch mehrere Interessengruppen beteiligt sind, entstehen häufig Dokumente, die eher Staatsverträgen als Designdokumenten gleichen.

Aber gerade in Situationen, in denen die Mittel knapp und die Interessen konträr sind, ist es von entscheidender Bedeutung, Entscheidungen und ihre Konsequenzen sichtbar zu machen. Denn spätestens, wenn ein knapper Zeitplan zu weiteren Kompromissen zwingt, ist es von unschätzbarem Wert, für jede Designentscheidung die zugrundeliegenden Anforderungen (und umgekehrt) zu kennen.

**Das Sammeln von Anforderungen ist ein mühsamer, teurer Prozeß mit vielen Beteiligten**

**Wenn Anforderungen und die daraus folgenden Entscheidungen dokumentiert sind, dann werden Handlungsspielräume transparent**

**Schöpferisches  
Chaos beim  
Sammeln,  
einfaches  
Strukturieren bei  
der Weiterver-  
arbeitung**

Wie alle Entwurfsprozesse, kennt auch die Anforderungsanalyse mehrere Abstraktionsebenen, die möglichst nicht vermischt werden sollten.

Andererseits ist das Sammeln von Anforderungen an ein noch nicht existierendes Gerät ein Vorgang, der eben nicht linear und in hierarchischer Gliederung abläuft. Erfahrene Entwickler erkennen frühzeitig kritische Komponenten des Entwurfs und versuchen dann durch einen hohen Detaillierungsgrad das Projektrisiko zu reduzieren. Notwendig ist deshalb ein Werkzeug, das es ermöglicht, unstrukturierte Texte zu gliedern, Strukturen zu definieren und Verknüpfungen zu schaffen, ohne dem Benutzer eine bestimmte Vorgehensweise aufzudrängen.

## **Verknüpfungen sorgen für Nachhaltigkeit**

**Durch die enge  
Verbindung  
zwischen Modell,  
Anforderungen  
und generiertem  
Code kann die  
Dokumentation  
mit geringem  
Aufwand aktuell  
gehalten werden**

Dokumentationen von technischen Systemen aktuell zu halten ist keine leichte Aufgabe. Weil an einem technischen Projekt viele Personen mit recht unterschiedlichem Hintergrund beteiligt sind, ist es in der Praxis meist so, daß die Dokumentation zur Klärung der Grundanforderungen benötigt wird, die weiteren Schritte aber häufig aus Zeitmangel und Desinteresse nur sporadisch beschrieben werden. Wenn keine übergeordnete Institution die Forderung durchsetzt, sind die schriftlichen Aufzeichnungen beim Abschluß eines Projekts nicht mehr aktuell. Gerade von diesem Zeitpunkt ab steigt aber die Bedeutung der Dokumentation für eine bezahlbare Wartung und Modellpflege.

**Aktualität der  
Dokumentation  
durch  
Unterstützung  
der Entwickler**

Für den einzelnen Entwickler war bisher der Nutzen der Dokumentation gering. Solange er am Projekt arbeitet, ist ihm die Spezifikation gegenwärtig, das Aktualisieren wird als lästig empfunden und Entwurfsänderungen lassen sich schneller ‚auf Zuruf‘ erledigen. Der später anfallende, höhere Aufwand bei Pflege und Erweiterung der Systeme ist wird selten den Verantwortlichen angelastet. während

die entwicklungsbegleitende Dokumentation offensichtlich teuer und lästig ist. Wer also an Nachhaltigkeit bei der Softwareerstellung interessiert ist, der muß für Unterstützung bei der Dokumentation sorgen.

Der natürliche Ansatz, dafür zu sorgen, daß Dokumentationen aktuell bleiben, besteht darin, die Trennung zwischen Spezifikation und Implementierung, wie sie das klassische Wasserfallmodell fordert, aufzuheben und eine enge Verknüpfung zwischen Spezifikationsprozeß und Implementierungsarbeit einzuführen. Das führt dazu, daß Wechselwirkungen zwischen Änderungen der Anforderungen und Änderungen der Implementierung sofort transparent werden.

## Werkzeuge

Für die Verwaltung der Anforderungen wird das Design Toolkit durch ein Fenster ergänzt, in dem Texte gegliedert und verwaltet werden.

In diesem Fenster werden Anforderungen, Spezifikationen und Testvorschriften erstellt oder aus bestehenden Dokumenten übernommen. Diese Texte können dann kategorisiert werden sowie miteinander und mit den Modellkomponenten verlinkt werden.

Mittels der Text- und Linkkategorien können unterschiedliche Sichten auf den Dokumenteninhalte aufgebaut werden. Die Anker von Links werden markiert und können durch alle Darstellungen der Werkzeuge verfolgt werden.

Linkketten können auf Durchgängigkeit und Vollständigkeit geprüft werden.

Der Benutzer kann neue Kategorien von Texten und von Links erstellen.

Für die Überprüfungen können Regeln definiert werden, so daß das Werkzeug an interne und externe Normen angepaßt werden kann.

**Spezifikationsänderungen können sofort in die Implementierung übernommen werden; Korrekturen der Implementierung sind einfach auf Rückwirkungen mit Anforderungen zu überprüfen**

**Die gesamte Dokumentation wird zentral verwaltet; alle benötigten Darstellungen können abgeleitet werden**

## HTML Export für die Dokumentation

Die gesamte Dokumentation inklusive der Verlinkungen kann als HTML Dokument exportiert werden und mit jedem beliebigen Browser betrachtet werden. Generierter Code kann ebenfalls in diese Exportfunktion einbezogen werden; sämtliche Links auf Modellebene werden dann im Code automatisch repliziert.

Weiter ist es möglich, das HTML Dokument durch generierte Java Applet Prototypen zu ergänzen, so daß eine vollständige Verifikation des Prototypen plattformübergreifend erfolgen kann.

## Systemanforderungen

### Anforderungen für das Toolkit

Für das Requirements Toolkit gelten die folgenden Mindestanforderungen:

- Windows NT, 9X oder XP
- Pentium III, 1GHz
- 500 MB RAM
- 500 MB freier Festplattenspeicher
- Grafikkarte mit 24Bit Grafik, mindestens 1024 x 768 (bedingt durch die vielen grafischen Darstellungen wird die Benutzung des Systems durch großzügige Bildschirme mit hoher Auflösung sehr erleichtert)

### Anforderungen für die die Einbeziehung von erzeugten Applet Prototypen

Zusätzlich wird für Applet Prototypen benötigt:

Java SE Development Kit (JDK) Version 6 für MS Windows, kostenlos erhältlich von [www.sun.com](http://www.sun.com)

bmp2png, Konvertierer für Bitmaps, Freeware © 1999-2005 Myasaka Masaru (wird mitgeliefert)

Convert, Konvertierer für Audio Files, Freeware © 1994 Jesus Villena (wird mitgeliefert).

Um die erzeugte Dokumentation auf einem Rechner anzuzeigen wird ein aktueller Webbrowser benötigt; soll eine Java Applet Prototyp integriert werden dann muß die Java Laufzeitumgebung , Version 6 installiert sein



Interaction Diagram Editor: E:\Dokumente und Einstellungen\Stefan\Desktop\SWT\Example... [F1] [Ctrl] [C]

File Use Case Component Interaction Extras Help

KorgFrqInp KorgTuner KorgMainCtrl MCalc Scale

ONOFFPressed OnOff  
DisplayOn displayOn  
erValue\* AlphaName:DisplayValue\*  
HorizontalSlideValue\* freq\*  
freq\* setF\*

Graph Table

TimerTick / DecCal, TimerShort, TimerStart

Up CalDown / TimerLong, TimerShort  
DownDown  
CalDown / DecCal, TimerStart

Generating Java Applet Code for Diagram KORGRT

Component: File: not changed  
E:\Programme\SWT\GeneratedApplets\KorgRT.java

main function  
Calculation Value MCalc  
Calculation Value Scale  
Counter Interface FPitch  
Realtime Data Source RD  
State Machine korgcalctd  
State Machine korgmainctrl  
State Machine korgEndEd3  
Table Lookup Notes  
Timer Interface 11  
Timer Interface 11

File Type: Java Applet

```
// korgrt.java: Applet
//
//
import java.applet.*;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.awt.geom.*;
//
// Main Class for applet Korgrt
//
```

KORG CHROMATIC TUNER CA-30

Copyright by KORG Inc.

event CALIBdown released

Stefan Sachs  
Dr. Ing.  
Beratender Ingenieur

Ringreiterweg 20  
23558 Lübeck

Tel. +49 (451) 8993444  
Fax +49 (451) 8993445  
E-Mail: [ssachs@acm.org](mailto:ssachs@acm.org)  
web: [www.ssachs.de](http://www.ssachs.de)