

Reengineering (von & mit) Architekturen

Dr. Frank Simon

Head of SQS Research & Innovation
 SQS Software Quality Systems AG
 Stollwerckstraße 11
 D-51149 Köln
 Frank.Simon@sqs.de

Dr. Kai-Uwe Gawlik

Head of Application Intelligence
 SQS Software Quality Systems AG
 Stollwerckstraße 11
 D-51149 Köln
 Kai-Uwe.Gawlik@sqs.de

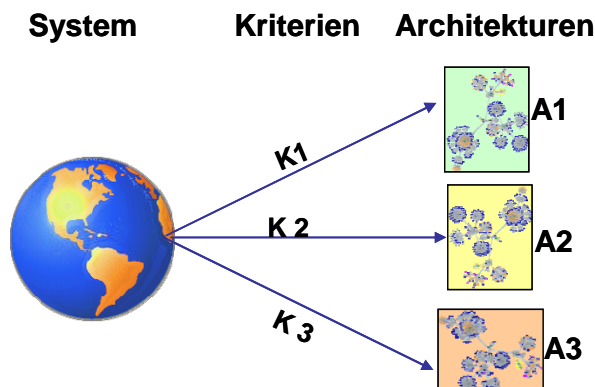
Zusammenfassung: Architekturen stellen prinzipiell ein mächtiges Mittel zur Unterstützung von Reengineering-Aktivitäten dar. In der Praxis scheitert dies allerdings häufig, da Architekturen nicht präzise definiert und sich insbesondere im völlig qualitätssicherungsfreien Raum befinden. In diesem Papier werden diejenigen qualitätssichernden Maßnahmen beschrieben, die VOR dem Einsatz von Architekturen für ein Reengineering zwangsläufig notwendig sind, um Architekturen tatsächlich gewinnbringend für das Reengineering einsetzen zu können.

Eine wesentliche Herausforderung des Reengineerings großer Systeme ist die Beherrschung der Systemkomplexität: Fehlende, falsche oder ungenügende Transparenz kann hier schnell zu schwerwiegenden Fehlentscheidungen führen. Ein wichtiges Hilfsmittel für ein planungssicheres Reengineering sind Architekturen, die wie folgt definiert werden können^[1]:

„Fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution.“

Die Mächtigkeit von Architekturen – auch für das Reengineering – wird aktuell in universellen Architekturkonzepten beschrieben und bearbeitet (z.B. im NATO Architecture Framework^[2] oder in SAGA^[3]). In der Praxis scheitert diese Möglichkeit allerdings häufig, da Architekturen unsauber definiert sind, das Zusammenspiel zwischen Architekturen und realen Systemen unklar ist, und Architekturen meist im „qualitätssicherungsfreien“ Raum existieren. Dieses Papier adressiert daher diejenigen qualitätssichernden Maßnahmen, die VOR dem Einsatz von Architekturen für ein Reengineering zwangsläufig notwendig sind. Als Grundlage erlaubt die obige Definition die Identifikation von 2 wesentlichen Architekturdimensionen:

- Betrachtete Systeme, die eine Dekomposition erfahren: Neben der Software besitzen z.B. auch Unternehmen als Organisation, Projekte, Geschäftsprozesse und Software-Entwicklungsprozesse die Möglichkeit der strukturierten Dekomposition und stellen damit Architekturen dar. Wichtig ist an dieser Stelle, dass Architekturen implizit immer schon bestehen! Lediglich die explizite Modellierung ist nicht in jedem Fall vorhanden.
- Verwendetes Dekompositionsprinzip: Selbst für ein konkret betrachtetes System sind unterschiedliche Architekturen zur Beschreibung möglich. So sind ausgehend von der Software z.B. technologische Architekturen möglich, bei denen die Technologie das Prinzip für die Komponentenbildung darstellt (.Net, SQL, Corba, etc.) oder auch funktionale Architekturen, bei denen der Zweck eines Moduls das Kriterium für die Komponentenbildung darstellt. Ein System besitzt daher eine Vielzahl unterschiedlicher Architekturen, wie in der nebenstehenden Abbildung dargestellt (in der Literatur häufig auch als spezifische Architektursichten bezeichnet).



Architekturen sind damit generelle Ansätze der Komplexitätsreduktion, die eine Vielzahl von geschäftskritischen Fragen beantworten helfen. Im Folgenden wird gezeigt, dass sie zudem schätzenswerte Artefakte der Qualitätssicherung darstellen und als solche gepflegt werden müssen; geschieht dies nicht, ist der Wert einer Architektur unklar. In diesen Fällen muss ein Reengineering VON Architekturen dem Reengineering MIT Architekturen vorangestellt werden. Hierbei können folgende 3 Detailstufen einer Architektur betrachtet werden, die jeweils separate QS-Aktivitäten bedeuten, ohne die ein Reengineering mit Architekturen kaum möglich ist:

1. Die Architektur an und für sich (Syntaktische Architektur-QS): Diese Sicht adressiert die einzelne Dekomposition und ihre explizite Darstellung an und für sich. Typische Beispielanforderungen sind
 - Verständlichkeit und Konsistenz der verwendeten Notation (bestenfalls der Verweis auf einen etablierten Standard wie UML).
 - Wartbarkeit der Architekturdokumentation, d.h. die Möglichkeit, Änderungen an ihr (werkzeugbasiert) vorzunehmen.
 - Portierbarkeit der Architekturdokumentation, d.h. die Möglichkeit, die Architekturbeschreibung mit unterschiedlichen Werkzeugen bearbeiten und dazwischen austauschen zu können.
2. Eine Architektur und ihr Zusammenspiel mit dem System (Semantische Architektur QS): Diese Sicht adressiert eine einzelne Architektur und ihr Zusammenspiel mit dem realen System. Wesentliche Anforderungen sind:
 - Korrektheit der Architekturbeschreibung: Stimmt die Architekturexplicitierung mit dem realen System überein, d.h. existiert entlang des Dekompositionskriteriums für jedes Objekt der Architektur ein entsprechendes Objekt im realen System und umgekehrt?
 - Wert der Einzel-Architektur: Hilft die konkrete Architektur, die Geschäftsstrategie, die mit dem System und der dahinterliegenden konkreten Architektur verbunden ist, bzgl. des einen verwendeten Dekompositionsprinzips zu erfüllen?
3. Mehrere Architekturen und ihre Synchronisierbarkeit (Strategische Architektur QS): Diese Sicht hat die allumfassende Sicht auf die Gesamtarchitektur, bestehend aus einer Vielzahl explizit gemachter Teilarchitekturen. Eine wesentliche Anforderung dieser QS besteht darin, dass die unterschiedlichen Architektursichten aufeinander abbildbar sind und die so entstehende Kombination die Gesamt-Geschäftsstrategie unterstützt. Ein typisches Beispiel für diese Architekturebene ist die Anfang 2009 von der Burton-Group-Analystin Anne Thomas Manes veröffentlichte Feststellung „SOA ist tot.“^[4]. Im Kern geht es nicht um die syntaktische oder semantische Architekturebene, sondern vielmehr um eine fehlende strategische Architektur QS, in dem versucht wird, eine technische SOA-Architektur in einer Nicht-SOA-Organisation zu etablieren: „[...] dass man bei vielen Implementierungen von SOA nicht zu den grundlegenden Aspekten einer Architektur durchgedrungen ist, sondern sich zu stark auf Technologien konzentriert hat“^[5].

Nur wenn diese 3 Ebenen von Architektur-QS kontinuierlich berücksichtigt werden (und ggfs. zu einem Reengineering VON Architekturen führen) sind Architekturen geeignet, im Kontext von Reengineering-Projekten konstruktiv verwendet werden zu können.

Referenzen

- [1] IEEE 1471 – Recommended Practice for Architectural Description of Software-Intensive Systems
- [2] NATO Architecture Framework (NAF), verfügbar unter http://www.nhqc3s.nato.int/ARCHITECTURE/docs/NAF_v3/ANNEX1.pdf
- [3] Standards und Architekturen für E-Government-Anwendungen (SAGA), Version 4 verfügbar unter http://www.cio.bund.de/DE/Standards/SAGA/saga_node.html
- [4] <http://apsblog.burtongroup.com/2009/01/soa-is-dead-long-live-services.html>
- [5] Harald Gläser: „Ist SOA gescheitert?“, in Objektspektrum, Online-Ausgabe SOA 2009