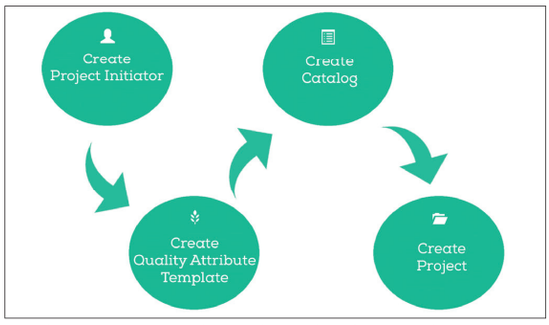
Emre
Avsar

Corina
Honegger

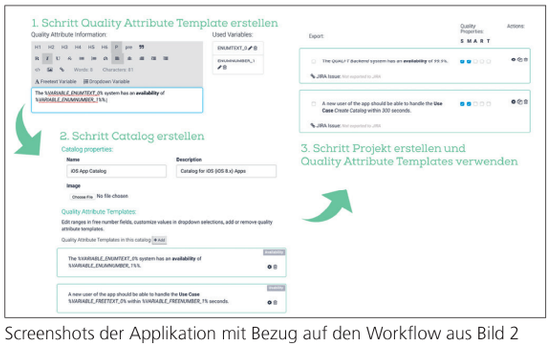
Diplomanden	Emre Avsar, Corina Honegger
Examinator	Prof. Dr. Olaf Zimmermann
Experte	Dr. Daniel Lübke, innoQ
Themengebiet	Software

QUALI-T – webbasierter Knowledge Manager für Architectural Analysis und Quality Attribute Elicitation

QUALI-T-Logo



Workflow zur Erstellung eines Projekts mit Zwischenschritten



Screenshots der Applikation mit Bezug auf den Workflow aus Bild 2

Ausgangslage: Im Rahmen des Requirements Engineering werden in Softwareentwicklungsprojekten zahlreiche nicht funktionale Anforderungen als Quality Attributes (QA) erhoben. Diese Architekturarbeit gliedert sich in drei Phasen, in denen die QAs teilweise als Input, teilweise als Output fungieren. In Phase eins, der Architectural Analysis, betrachtet man den gesamten Systemkontext sowie die funktionalen Anforderungen und leitet daraus die QAs ab. Anschliessend verfeinert man in Phase zwei, der Architectural Synthesis, diese QAs und kann auf deren Basis Architekturentscheidungen treffen. In Phase drei, der Architectural Evaluation, verifiziert und prüft man die QAs schliesslich, vorzugsweise durch eine neutrale, externe Stelle.

Vorgehen/Technologien: Die drei Phasen Architectural Analysis, Synthesis und Evaluation sind ohne Hilfsmittel schwierig zu realisieren. Etablierte Hilfsmittel stellen die Utility Trees und Quality Attribute Scenarios des Software Engineering Institute und zahlreiche ähnliche statische Dokumentvorlagen dar. Jedoch gibt es auf dem Markt aktuell kein Werkzeug, das den QA-Engineering-Prozess über alle drei Phasen unterstützt und die statischen QA-Dokumentvorlagen zum Leben erweckt. Das Ziel dieser Bachelorarbeit ist es, zu untersuchen, welche Eigenschaften in einem derartigen Werkzeug sinnvoll unterstützt werden können, und einen Prototyp eines solchen QA-Management-Werkzeugs zu entwickeln. Dazu erfassen wir zunächst über Personas die zentralen Aufgaben und Anforderungen in den drei Phasen der Architekturarbeit und leiten daraus Use Cases ab. Anschliessend fokussieren wir auf die QAs und leiten die wünschenswerten Werkzeugeigenschaften aus Literatur und eigenen Projektbeispielen ab. Der dritte Teil der Arbeit beinhaltet Design und Entwicklung einer passenden Software, um die Architectural Analysis, Synthesis und Evaluation zu unterstützen.

Ergebnis: Das Ergebnis der Arbeit ist die Client-/Server-Webapplikation QUALI-T, die serverseitig mittels Play Framework und clientseitig mit AngularJS entwickelt wurde. In QUALI-T legt man QAs als Vorlagen an und gliedert diese in Kategorien. Um die Qualität dieser QAs sicherzustellen, beurteilt man sie mithilfe von Quality Properties. Diese bestehen in QUALI-T standardmässig aus den SMART-Kriterien, nach denen ein QA Specified, Measurable, Agreed Upon, Realistic und Time-bound sein sollte; projektspezifische Erweiterungen sind möglich. Es besteht zudem die Möglichkeit, über eine Programmierschnittstelle Detektoren einzubauen, die den Text eines QAs überprüfen und Feedback dazu geben.