Bare Demo of IEEEtran.cls for IEEE Conferences

Franz-Dominik Dahmann
Master Informatik
Hochschule Bonn-Rhein-Sieg
https://www.h-brs.de/de
Grantham-Allee 20, 53757 Sankt Augustin
Email: franz.dahmann@smail.inf.h-brs.de

Jan Eric Müller
Master Informatik
Hochschule Bonn-Rhein-Sieg
https://www.h-brs.de/de
Grantham-Allee 20, 53757 Sankt Augustin
Email: jan-eric.mueller@smail.inf.h-brs.de

***Abstract*—Bei der Realisierung eines Softwareentwicklungsprojektes besteht hufig eine Kluft zwischen den Software-Architekten auf der einen Seite und den Requirements Engineers auf der anderen Seite. Software Architekten sind zum Beispiel hufig mit dem Problem konfrontiert, dass Anforderungsdokumente nicht ausreichend sind um weitreichende Architekturentscheidungen in Bezug auf den Softwareentwurf zu treffen. Daher ensteht in diesem Fall fr den Software-Architekten hufig der Mehraufwand, dass dieser ber weitere Interviews ein przises Bild von der gewnschten Software Architektur erhlt. Dies resultiert hufig in Verzgerungen die sich dann darin wiederspiegeln, dass Termine nicht eingehalten werden knnen. Betreibt der Software Architekt diesen Mehraufwand nicht und trifft eigene Annahmen bezglich der Architektur der Software kann dies wiederum zu einer geringeren Akzeptanz des Kunden und im schlimmsten Fall zum Auftragsverlust fhren. Die Requirements Engineers wissen auf der anderen Seite wiederum nicht, welche Anforderungen konkret wichtig fr den Software Architekten sind, da ihnen die entsprechende Fachkenntnis fehlt. Somit knnen diese nicht zielfhrend die notwendigen Informationen mit dem Kunden erarbeiten. Ohne diese Informationen ist eine ausreichende Grundlage der Anforderungen fr den Architekturentwurf nicht gegeben. Um diese Kluft zu berbrcken ist es notwendig Verfahren zu ermitteln, die es Requirements Engineers ermglicht die richtigen Informationen einzuholen und die Zusammenarbeit mit den Software Architekten zu optimieren.**

Im folgenden werden Verfahren untersucht die das Potenzial haben die Zusammenarbeit zwischen Requirements-Engineer und Software-Architekt zu verbessern. Ferner wird berprft wo diese Verfahren ihre Strken und Schwchen offenbaren und wie diese gewinnbringend kombiniert werden oder sich gegenseitig ergnzen knnen. Das zentrale Problem der Zusammenarbeit scheint eine mangelnde Verstndigung zwischen den Requirements Engineer und dem Software Architekten zu sein. Software Architekten sehen sich gezwungen, entweder eine Architektur anzunehmen, oder weiterfhrende Gesprche mit dem Kunden zu fhren. Requirements Engineers sehen sich mit dem Problem konfrontiert, dass sie nicht die Fachkenntnis haben, die architekturrelevanten Informationen mit dem Kunden zu erarbeiten. Um jedoch eine Optimierung zu ermglichen ist es zunchst notwendig zu identifizieren, warum die Anfordungsdokumente, die der Requirements Engineer generiert, nicht ausreichen um einen vollstndigen Architekturentwurf herzustellen und wie Erkenntnisse des Software-Architekten wieder in die Anforderungs-Spezifikationen einflieen knnen. Auerdem wird recherchiert ob Zusammenarbeit und Kommunikation durch eine passende Tool-Untersttzung verbessert werden kann.

Um eine Optimierung zu ermglichen bieten sich verschiedene Vorgehensweisen an. Im folgenden wird eine Kategorisierung von Anforderungen vorgenommen, die Relevant fr die Architektur einer Software sein knnen. Diese Kategorisierung bietet Requirements Engineers die Mglichkeit speziellere Fragen zu der gewnschten Architektur der Software zu stellen. Neben einer Kategorisierung wird ferner ein iteratives Vorgehen zum Design der Software Architektur vorgestellt. Mithilfe des Attribute driven Designs (ADD) soll es mglich sein eine feste Struktur zu haben, anhand derer eine korrekte Architektur entworfen werden kann. Desweiteren wird ein Ansatz vorgestellt und bewertet, welcher die Prozesse der Architektur-Entscheidung und des Requirements-Engineering erfahrungsgetrieben in einen gemeinsamen Prozess integriert. Dieser ermglicht eine engere Kommunikation zwischen Software-Architekt und Requirements Engineer. Anschlieend wird dieser Ansatz mit dem Twin Peaks Modell verglichen und beurteilt welches der beiden Verfahren den greren Mehrwert bringt oder ob diese sinnvoll kombiniert werden knnen.

Mit der Untersuchung der Verfahren soll erreicht werden, dass sowohl dem Requirements Engineer als auch dem Software Architekten, Potenziale aufgezeigt werden auf deren Basis die Zusammenarbeit verbessert werden kann. Durch die Beschreibung mehrerer Verfahren soll zudem die Mglichkeit gegeben sein, abzuwgen welche in der individuellen Situation am besten geeignet sind.

I. Introduction

This demo file is intended to serve as a "starter file" for IEEE conference papers produced under LaTeX using IEEEtran.cls version 1.8b and later. I wish you the best of success.

mds

August 26, 2015

A. Subsection Heading Here

Subsection text here.

1) Subsubsection Heading Here: Subsubsection text here.

II. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

REFERENCES

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.