Conway´s Law

Wechselwirkung zwischen Organisationsstruktur und Softwarearchitektur

99 FACETS of AGILE

codecentric

Agenda

1. Wer sind wir & wo kommen wir her?
2. Was ist dieses Conway´s Law?
3. Beispiele
4. Ergebnisse
5. Implikationen für Wipsys & andere

Referierende

Kristina Müller

Wirtschaftspsychologin (B. Sc.)
Systemische Beraterin (M. A. - i. A.)

99 FACETS of AGILE

Oliver Hoogvliet

Senior IT Consultant (codecentric AG)
Agile Coach

codecentric

Kontext - Agilität

Agilität ist „… Fähigkeit von Unternehmen zur **kontinuierlichen Anpassungsfähigkeit** an komplexe, turbulente und unsichere Umwelt“

(Goldman, Nagel & Preiss, 1995)

Achtung: Agilität meint damit **nicht** den umgangssprachlich definierten Eigenschaftsbegriff.

Kontext - VUCA

V = Volatil (Volatility)

U = Ungewiss (Uncertainty)

C= Komplex (Complexity)

A = Mehrdeutig (Ambiguity)

Quelle: https://hbr.org/2014/01/what-vuca-really-means-for-you

Kontext - Digitalisierung

- Digitalisierung von Information & Kommunikation
- Internet of Things (IoT)
- Disruptive & digitale Innovationen

Quelle: http://wirtschaftslexikon.gabler.de/Definition/digitalisierung.html

Conway’s Law

> *“... organizations which design systems (in a broad sense used here) are constrained to produce designs which are copies of the communication structures of these organizations.”* (Conway, 1968)

→ Organisationen produzieren Systeme, die eine Kopie ihrer Kommunikationsstrukturen darstellen

→ Bei Change-Projekten sind nicht nur Organisationsstruktur und die Menschen relevant. Die Softwarearchitektur kann unbewusster/unsichtbarer Showstopper sein.

Stellen wir uns mal vor…

Change mit Umstrukturierung

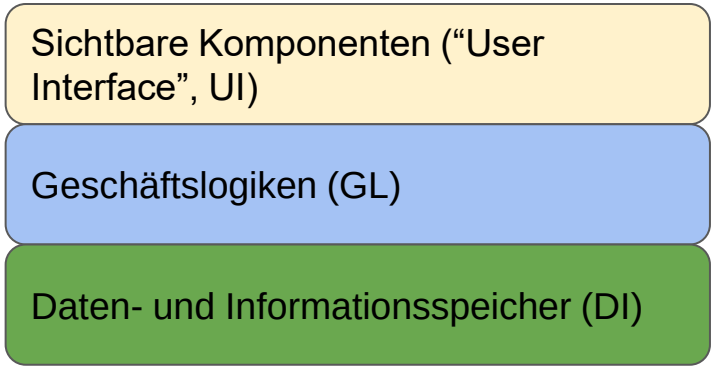
- Organigramm neu
- Teambuilding

Aber:

- Die Softwareentwicklung ist nach wie vor zu langsam
- liefert nicht und ist fehlerhaft
- Zusätzlich finden ständig Meetings zwischen den neu geschnittenen Teams statt.

Hypothese - “Monolith” ist schuld

- Schichten-Modell
- ein großes Stück Software (“Monolith”)
- feste Verdrahtung zwischen den Schichten
- feste Verdrahtung innerhalb der Schichten



Nachteile:

- Veränderungen in Teilen des Systems lösen Fehler in anderen Teilen des Systems aus
- Wartung des Systems wird unübersichtlich
- Unklare Zuständigkeiten innerhalb der Firma

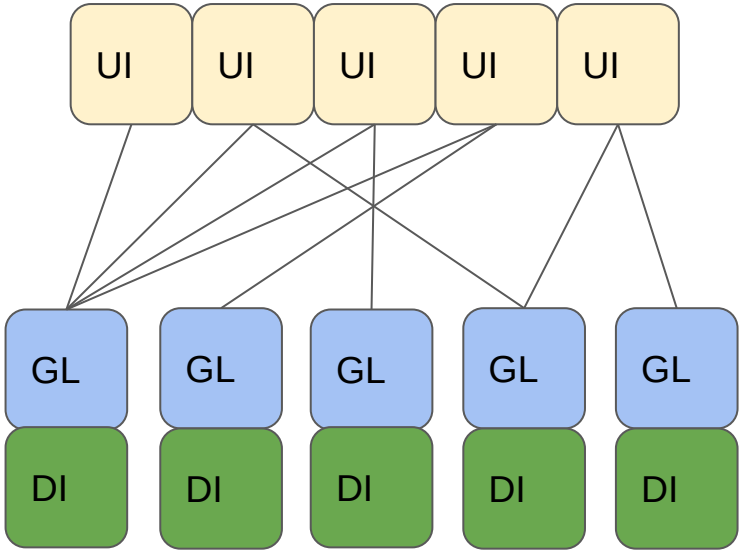
Schlussfolgerung

Softwarearchitektur muss flexibel

für Veränderungen sein!

Kontext - Microservices

- Aufsplitten des Monoliths in kleine Teile (Module)
- “Microservices”: die Geschäftslogik und die Datenspeicherung wird in kleine Bauteile (Module) aufgeteilt
- ein Microservice ist für eine ganz bestimmte Aufgabe zuständig (“business concern”)
- keine Abhängigkeiten zwischen den Microservices
- die Services werden von in der UI nach Bedarf eingesetzt



Das Aufsplitten ermöglicht eine einfachere Wartung und Erweiterung der Software

Aber: WER ist für die Wartung der einzelnen Produkte zuständig?
WIE kommt man vom Monolithen zum Microservice?

Ergebnisse

- *„Produktinnovationen, welche die Architektur des Produktes ändern, (benötigen) eine **Änderung der Wissensarchitektur und Firmenstruktur**”* (Henderson & Clark , 1990)
- Produktqualität wird stark durch die Organisationsstruktur beeinflusst (Nagappan, Murphy & Basili, 2008)
- „…die Arbeit *an einem modularisierten System sollte derart verteilt werden, dass die **Trennung der Entwicklung der Aufteilung der Module** entspricht. Umgekehrt sollte die Entwicklung nur dann aufgeteilt werden, wenn die zu entwickelnden Produkte (oder Produktteile) gut verstanden werden, Pläne, Prozesse und Schnittstellen etabliert und stabil sind.”* (Herbsleb & Grinter, 1999)

Implikationen für Wipsys & andere

1. Sensibilität für Rolle der Technik
    a. Softwarearchitektur
    b. digitale Kommunikationstools
    c. System der Team-/Abteilungsstruktur bildet sind oft in der Softwarearchitektur bzw. in der Produktlandschaft ab
2. Bei Change Prozessen in Software-Unternehmen interdisziplinäre Zusammenarbeit (Techis mit reinnehmen)
3. Techis als mündige Stakeholder einbeziehen

Quellen

- Conway, Melvin E. (1968). How Do Committees Invent?. in F. D. Thompson Publications, Inc. (Hrsg.): Datamation. Band 14, Nr. 5., p. 28–31. abgerufen am 01.04.2017 von http://www.melconway.com/research/committees.html

- Goldman, S. L., Nagel, R. N. & Preiss, K. (1995). Agile Competitors and virtual organizations: Strategies for Enriching the Customer. New York: Van Nostrand Reinhold

- Herbsleb J. D. & Grinter, R. E. (1999). Splitting the organization and integrating the code: Conway´s law revisited. Procedings of the 21st international conference on software engineering. p. 85-95

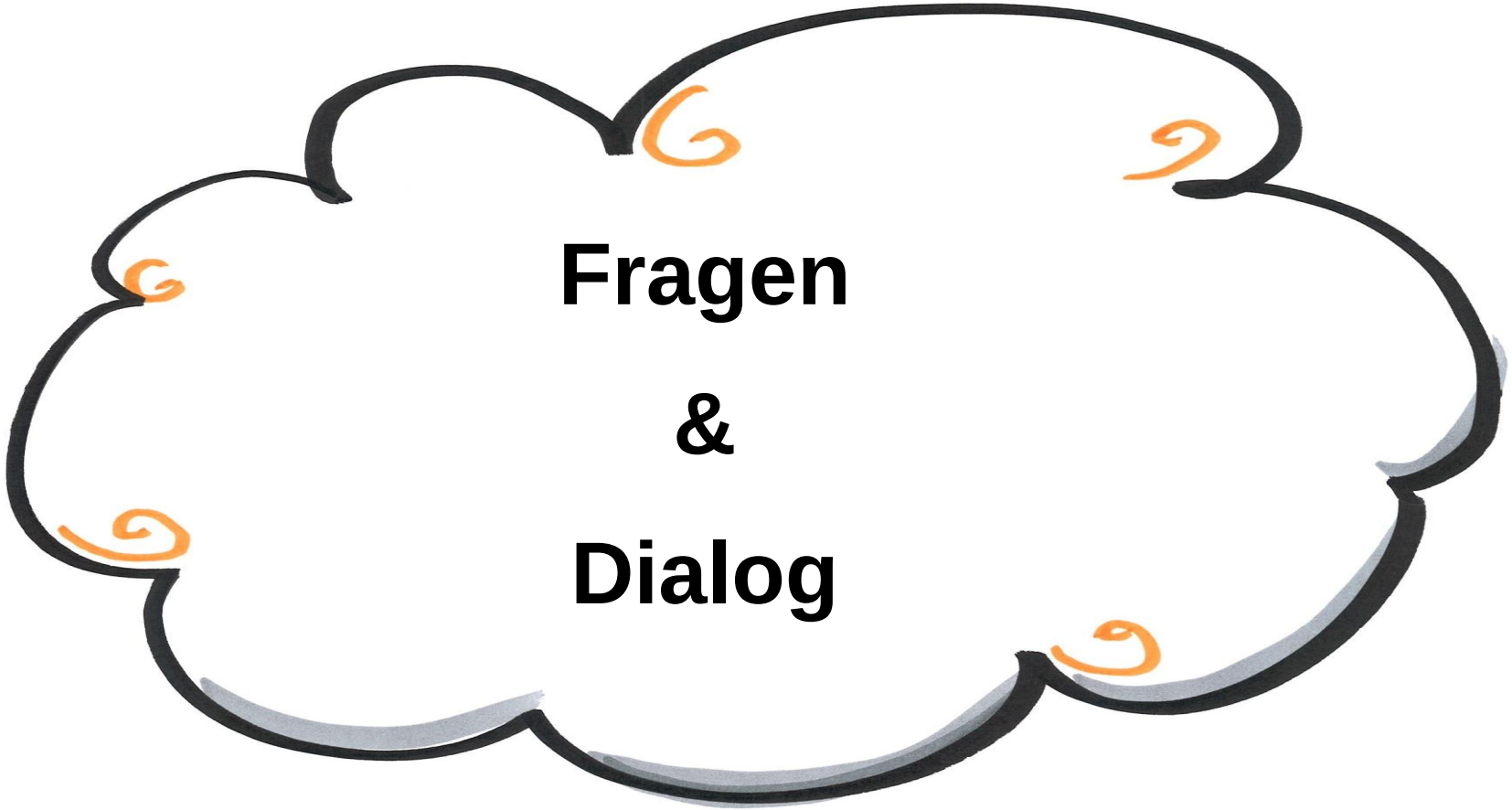
- Henderson, R. M. & Clark, K. B. (1990). Architectural Innovation: The regonfiguration of existing product technologies and the failure of established firms. Administrative Science Quarterly. Vol. 35, No. 1, Special Issue: Technology, Organizations & Innovation. p. 9-30. abgerufen am 01.04.2017 von http://www.edegan.com/pdfs/Henderson%20Clark%20(1990)%20-%20Architectural%20Innovation.pdf

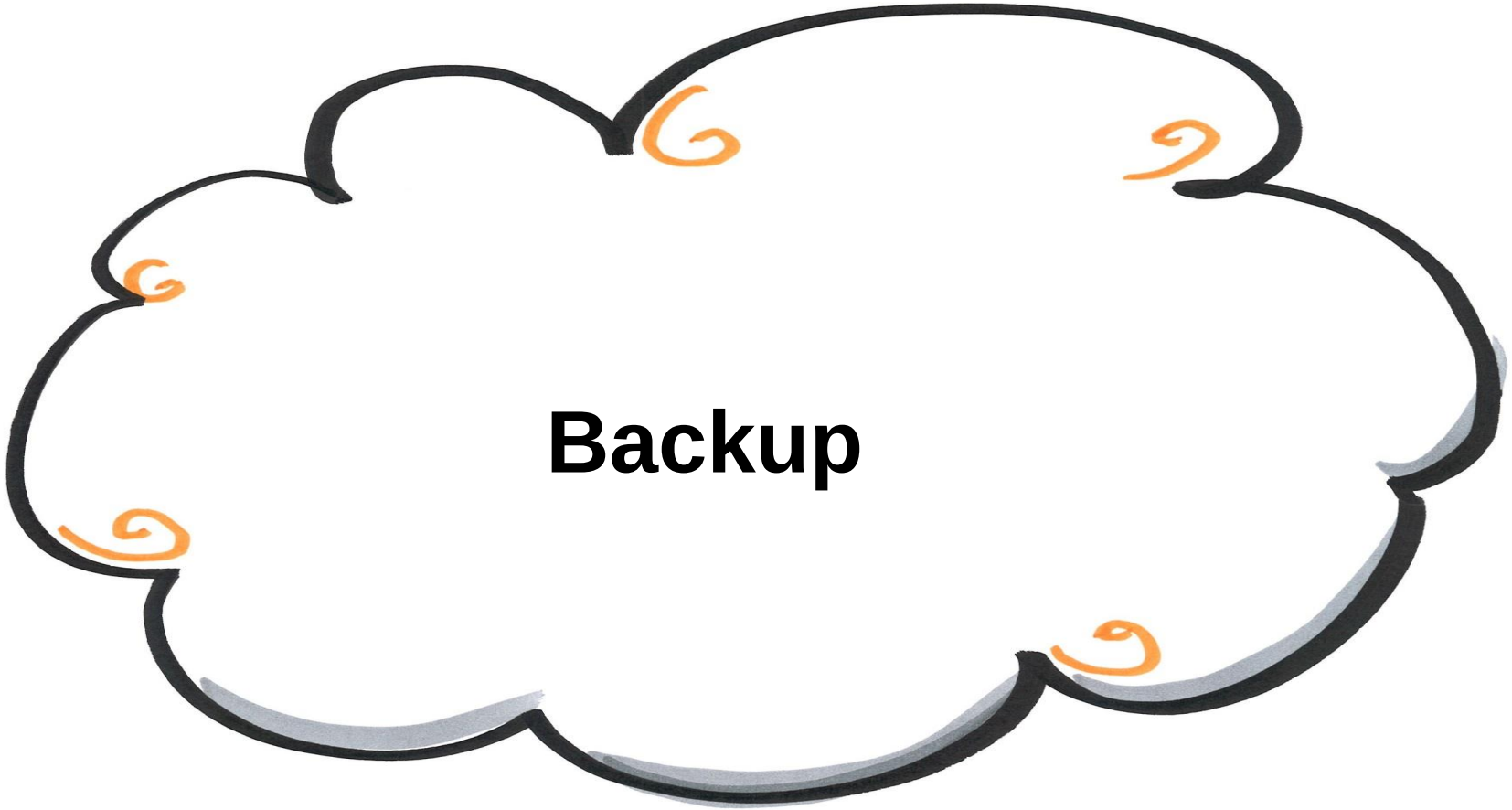
- Nagappan, N., Murphy, B., & Basili, V. R. (2008). The influence of organizational structure on software quality: an emirical case study. abgerufen am 01.04.2017 von https://www.microsoft.com/en-us/research/publication/the-influence-of-organizational-structure-on-software-quality-an-empirical-case-study/#

Danke :)

kristina@99facetsofagile.de

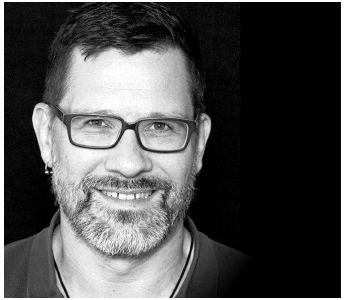
oliver.hoogvliet@codecentric.de


Fragen
&
Dialog


Backup

Referierende

Kristina Müller ist Wirtschaftspsychologin (B. Sc.) und angehende Systemische Beraterin (M. A.). Sie begleitet als freiberufliche Agile Consultant & Coach Unternehmen auf dem Weg zu mehr Agilität. Hierbei ist sie spezialisiert auf agile Methoden/Frameworks wie Scrum und Kanban, laterale und situative Führung, Team- und Organisationsentwicklung sowie Agile HR. kristina@99facetsofagile.de



Oliver Hoogvliet ist Diplom-Biologe und arbeitet seit 20 Jahren in der Softwareentwicklung. Seine Begeisterung für die agile Softwareentwickung entwicklelte sich durch die Arbeit in unterschiedlichen Rollen in agilen Teams. Aktuell ist er als Senior IT Consultant bei der codecentric AG tätig. Seine Schwerpunktthemen sind agile Softwareentwicklung, agile Methodiken, Continuous Delivery, Spring Boot, Java, Groovy und DevOps. oliver.hoogvliet@codecentric.de