- # Title: Die digitale Gesellschaft: Software-Systeme in der Komplexitäts-Krise?

  Die Komplexität von Software Systemen hat in den letzten Jahren stetig zugenommen und gleichzeitig ist Software das Nervensystem unserer Gesellschaft geworden. In Anbetracht der zu erwartenden gewaltigen ökologischen, gesellschaftlichen, politischen und in Folge ökonomischen Herausforderungen der kommenden Jahrzehnte, ist es von fundamentaler Bedeutung diese zunehmend kritische Infrastruktur nachhaltig und vor allem resilient zu gestalten.

  Schon heute stecken viele von Software abhängige Organisationen und Unternehmen in der Krise: Fehler, im schlimmsten Fall Ausfälle, mangelnde Wartbarkeit, Sicherheitsprobleme, stetig steigende Betriebskosten aber auch ethische Probleme können zur existenziellen Bedrohung der Infrastruktur und damit auch der Betreiber werden.

  Wie sind wir in diese Komplexitäts-Falle geraten? Welche treibenden Faktoren kann man beobachten und, vor allem: welche Rolle spielen menschliche und organisatorische Faktoren um zu nachhaltigen und resilienten Softwaresystemen zu gelangen.

Einleitung

Vorstellung

- viele Jahre an der TU, Chemie, Software Engineering von komplexen Softwaresystemen
- Startup: Sophisystems
- Beratung Deloitte
- GF: Biac/TF – IT-Tochter der Vienna Insurance Group
- SBA und freidenkend

Kontakt / Infos

- https://www.schatten.info
- https://sichten.blogspot.com
- https://podcast.zukunft-denken.eu
  coming soon – stay tuned!
- https://twitter.com/alex_buzz
- http://photos.schatten.info

Zwei Meta-Gedanken

Wofür Konferenzen?

- nicht zur Informationsvermittlung: Dafür kaufe ich mir ein Buch, das ist besser und billiger bzw. schaue mir im schlimmsten Fall ein YouTube Video an
- Um Menschen mit ähnlichen Interessen oder Problemen kennenzulernen und neue Anregungen zu bekommen

    - Daher ist dieser Vortrag auch eine Angel-Expedition in wenig klaren Gewässern mit der Hoffnung den einen oder anderen dicken Karpfen an Land zu ziehen
- Viele ausgezeichnete technische Vorträge – versuch zwei Schritte zurück zu treten und eine andere Perspektive einzunehmen

Was ist das Problem?

- **Kern-These: Wir haben viele unserer komplexen IT-Systeme nicht mehr oder nicht hinreichend unter Kontrolle**

Wer ist *wir?*

- Unternehmen / Betreiber
- Technik(er) auf verschiedenster Ebene
- Nutzer / Gesellschaft
- Politik und ordnenden Strukturen

Wir beobachten zahlreiche Symptome (siehe Abbildung)

- Große Projekte scheitern (regelmäßig) oder verursachen dramatische Mehrkosten
  - VIG schreibt 180mio € ab (SAP)
  - Lidl schreibt 500mio € ab (SAP)
- Qualitätsprobleme
- Performance Probleme beziehungsweise hohe Betriebskosten
- »Bloatware« – mit trivialer Funktionalität
- Ständig steigende IT-Kosten ohne entsprechende Fortschritte: »Versteinerung der Systeme«
- Ethische und gesellschaftliche Probleme
  - mangelnder Datenschutz
  - Automatisierung (Finanzsysteme, Trading, Beurteilung von Schadensfällen, usw.)
  - Tracking
  - Spionage
  - Manipulation
  - Ablenkung
- **Security Vorfälle**
  - Deloitte hacked 2017
    https://www.theguardian.com/business/2017/sep/25/deloitte-hit-by-cyber-attack-revealing-clients-secret-emails
  - British Airways Leak 2018
  - Facebook 2018: large number of companies have full access to customer data
  - Marriott data theft 2018 (halbe Milliarde Kundendaten)

- 35C3 – Gesundheitsakte Deutschland
  https://media.ccc.de/v/35c3-9992-all_your_gesundheitsakten_are_belong_to_us

Blick mehr auf das Problemfeld, nicht die Symptome richten!

- zahlreiche Einflussfaktoren
- einer der wesentlichsten: **zunehmende Komplexität der Systeme** gepaart mit
  - ständiger Optimierung auf Kosten von **Resilienz**

Das ist allerdings kein neues Problem (Abbildung)

- ➔ »Software Crisis« **(1968)**
  - Nato Software Engineering Conference (1968): https://en.wikipedia.org/wiki/NATO_Software_Engineering_Conferences
  - Edsger Dijkstra, The Humble Programmer, ACM Turing Lectures (1972)
  - Melvin E. Conway, How do Committees Invent?, Datamation (1968)
- ➔ »Wicked Problems« **(1973)**
  - Rittel, Webber, Dilemmas in a General Theory of Planning, Policy Sciences 4 (1973)
  - Steve Easterbrook, From Computational Thinking to Systems Thinking, Proceedings of the 2nd International Conference on Information and Communication Technologies for Sustainability (ICT4S'2014)
  - s.u.
- Neue Fähigkeiten *werden offenbar hauptsächlich in neue Komplexität investiert, nicht bestehende Systeme zu verbessern*
  - Tools
  - Sprachen und Plattformen, Bibliotheken
  - Hardware-Leistungssteigerung
  - Virtualisierung, Cloud Systeme
  - Monitoring und Management-Systeme
  - *werden offenbar hauptsächlich in neue Komplexität investiert, nicht bestehende Systeme zu verbessern*

Was steht auf dem Spiel?

alles: IT ist das digitale Nervensystem unserer Gesellschaft

- »Software is eating the world«, Marc Andreessen
  https://www.wsj.com/articles/SB10001424053111903480904576512250915629460

Kritikalität von IT-Systemen nimmt daher ständig zu, auf unterschiedlichen Zeitachsen

- Kurzfristig existentielle Bedrohung wie
  - Lieferketten, Nahrungsmittel
  - Wasser, Stromversorgung
  - Mobilität
  - Medizinische Versorgung
  - Staatliche Strukturen (Polizei, Verwaltung, etc.)
  - etc.
- Mittelfristig gesellschaftliche und politische und ökonomische Bedrohung (s.o.)
- Langfristig
  - wesentliche Daten und Prozesse aus alten IT-Systemen in der Zukunft erhalten

- **Daten und Schutz von Daten ist eine wesentliche Herausforderung, aber die Bereitstellung der Prozess-Funktionalität ist ebenso von fundamentaler Bedeutung**
- **en und tatsächlichen Fortschritt (nicht »Innovation«) zurückerhalten**

Was ist *Komplexität?*

Versuch einer Definition

- Viele Teile / Teilsysteme **interagieren** miteinander, verändern einander dabei. Daraus entsteht *bottom up* die Gesamtfunktionalität des komplexen Systems
- Komplexe Systeme entwickeln ihre **Dynamik über die Zeit**, die **Vergangenheit** ist oft von erheblicher Bedeutung – statische Betrachtungen sind wenig sinnvoll
  - "Simple objects are more dependent on (physical) constraints than on history. As complexity increases, history plays the greater part.", Francois Jacob
- Nicht zu verwechseln (siehe **Abbildung**):
  - inspiriert von Rich Hickey: allerdings fokussiert er sich zu stark / ausschließlich auf die vermeidbare Komplexität!

    https://www.infoq.com/presentations/Simple-Made-Easy
    - **Complex**
    - **Complicated**
    - (Simple)
    - (Easy)
    - *Hickey fokussiert sich allerdings im wesentlichen auf Komplexität, die man sich überflüssigerweise eingetreten hat, nicht auf den Umgang mit unvermeidlicher Komplexität*

Charakteristiken und Systemgrenzen

- **Ursachen-/ Wirkungs- Beschreibungen** machen wenig Sinn
  - zyklische oder komplexe Abhängigkeiten
- Systeme moderner Gesellschaften sind komplexe Systeme von Systemen und können unerwartete **Skaleneffekte** (»Emergenzen«) zeigen; Beispiel:
  - ein Dorf ist keine große Familie
  - eine Stadt ist kein großes Dorf
  - ein Land ist keine große Stadt
  - die globale Gesellschaft ist kein großes Land
  - **➔ es gibt kein »global village«**, das ist fundamentaler und irreführender Unsinn
  - ähnliche Skaleneffekte / Emergenzen treten häufig auf, wenn Dinge in Raum oder Zeit skaliert, z.B. auch neue Technologien ausgerollt werden
- **Verhalten** komplexer Systeme ist nicht einfach vorhersagbar (Attraktoren, Tipping Points, etc)

- Wir neigen dazu unsere Fähigkeit komplexe Systeme zu kontrollieren weit zu überschätzen
- Siehe z.B. Philip Tetlock, Dan Gardner, Superforecasting – The Art and Science of Prediction

- Es gibt **systemische Archetypen:** typische systemische Muster, die in verschiedensten Kontexten auftauchen
  - z.B. https://thesystemsthinker.com/wp-content/uploads/2016/03/Systems-Archetypes-I-TRSA01_pk.pdf
    - fixes that fail
    - tragedy of the commons
  - Donella H. Meadows, Thinking in Systems: A Primer
  - Zoll, Healy, Resilience, When Things bounce back
  - Barnosky et al, Approaching a state shift in Earth's biosphere (Nature Review)
- Konkret bei Software: wer kann die **Eigenschaften** des aktuellen Systems, für das sie verantwortlich sind (vollständig) beschreiben?
  - ➔ Niemand
  - System-Funktionalität ist eine komplexe Mischung aus
    - Hardware
    - Software
      - verschiedenste Komponenten – technische Komplexität
      - an verschiedensten Orten und
      - verschiedenster Verantwortung
    - Entwicklern (und deren **implizitem** Wissen)
    - Betrieb (und deren **implizitem** Wissen)
    - Fachseite (und deren **implizitem** Wissen)
    - Kunden (und deren **impliziten** Anforderungen)
    - Management, das versucht die Kontrolle zu bewahren
    - … und der formelle und informelle **Interaktion** zwischen all diesen Gruppen

»Wicked Problems«

- Es gibt keine einfache Beschreibung des Problems
- Es gibt keine Abbruch-Regel, keine einfache “richtig/falsch” Lösung
  - Die Lösung wird beendet, wenn die Ressourcen zu Ende gehen

    - oder wenn die »gut genug«, »besser als zuvor« ist
- Jedes »wicked problem« ist im wesentlichen einzigartig
- Ein »wicked problem« verbreitert sich, ist also ein Symptom eines anderen Problems, oder kann durch verschiedenste Lösungsansätze angegangen werden

Was nun?

Zusammenfassend

- komplexe Softwaresysteme im genannten Sinn zeigen **emergentes Verhalten**, das wir noch nicht hinreichend verstehen. Klassische **Management-Ansätze** funktionieren nicht mehr
- ➔ Eine **statische Definition** von Funktionalität und Qualität macht daher bei heutigen Systemen im Kern **keinen Sinn** mehr
- **Problembewusstsein** ist ein erster sehr wichtiger Schritt

- **Viele Fragen offen: Wir müssen noch sehr viel über den Umgang mit komplexen Systemen lernen, ein paar Aspekte:**

1. Neue Phänomene brauchen neue Bilder

- zurück zum **Skaleneffekt!**

  Wir denken von Management, Architektur Seite usw. über Softwaresysteme immer noch so als wären die Stadt schlicht eine Summe kleiner Dörfer. Oder gar ganz Europa verhielte sich wie ein großes Dorf.

- ➔ biologisches / organisches Bild: **Garten, Ökosystem**

Vielleicht gibt es in wenigen Jahren Software-Psychologie / Soziologie?!

- Wenn wir die Idee der Emergenz Ernst nehmen erleben wir gerade das Entstehen von emergentem Verhalten.
- Dieses wird sich – wie in allen anderen Schichten nicht aus den unteren Schichten alleine heraus erklären lassen.
- Chemie ist keine bessere Physik und die Psychologie keine angewandte Biologie ist.
- Komplexe Software-Systeme sind daher etwas anderes als eine Summe von Programmen

2. *Systemisches Denken* statt *Command and Control*

oder: mehr »bottom up« statt »top down«

komplexe Systeme brauchen neue Methoden der »Steuerung«

- **Software als Projekt ist der erste Schritt in den Untergang**
- ➔ Agil
- ➔ Selbstorganisierende Einheiten stoßen häufig an traditionelle Management-Ideen
- *Ständige* inkrementelle und **evolutionäre Weiterentwicklung**
- ~~Wasserfall-Management und Budgets~~ ➔ z.B. Beyond Budget

- ~~Management KPIs und Boni~~ ➔ neue Anreizmodelle, vielleicht *objectives and key results (OKR)*

Evolutionäre Architektur

Martin Fowler…

- **unnötige Komplexität** vermeiden
- **notwendige Komplexität** im Griff behalten

Denken in Dynamik – Laufzeit

- Chaos Engineering ➔ Denken in der Laufzeit, nicht im Testsystem

3. Neuer Umgang mit Unsicherheit und Veränderung

- ~~vermeintliche Stabilität (aus Komponentensicht)~~ ➔ **Resilienz der Kern-Funktionalität**
- ~~klassisches Risikomanagement~~ ➔ **Ausfall ist die Norm** und darf nie das gesamte System bedrohen
- Kompartmentalisieren, Sandboxing, etc.
- *möglicher Angriffs- und Ausfallsschemen sollten ständig im Produktiv-System simuliert werden*

Und weiter…?

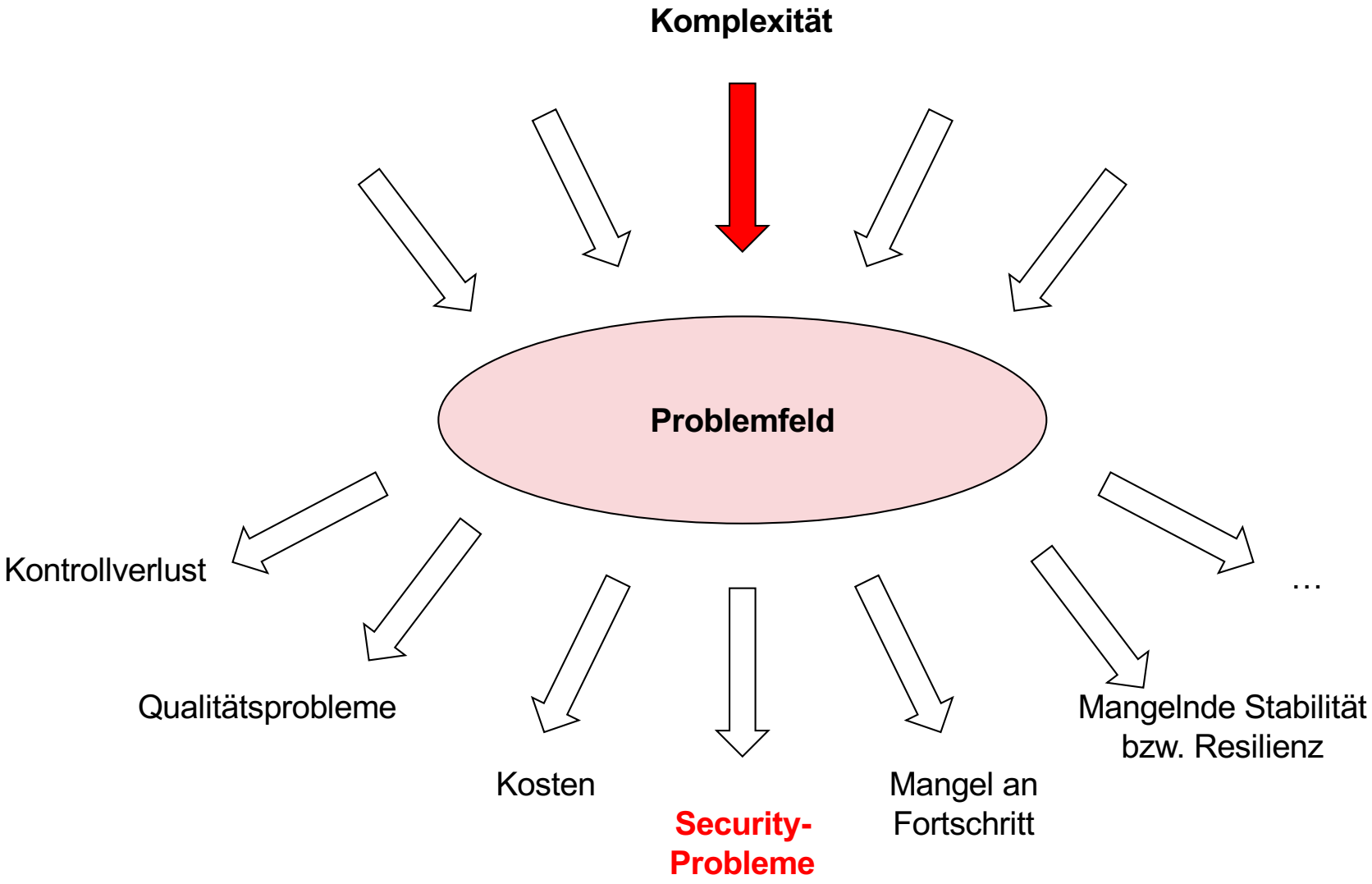
- **An dieser Stelle kann das Problem natürlich nur angerissen werden**
- **Freue mich auf persönliches Gespräch – besonders auch abseits der Konferenz**
  - Kontakt s.o.
- **SBA**
  - Beratung auch bei systemischen Problemen im
    - Software Engineering und
    - Management von Software
  - Bootcamp – Beyond Coding (siehe Abbildung)

Die digitale Gesellschaft: Software-Systeme in der Komplexitäts-Krise?

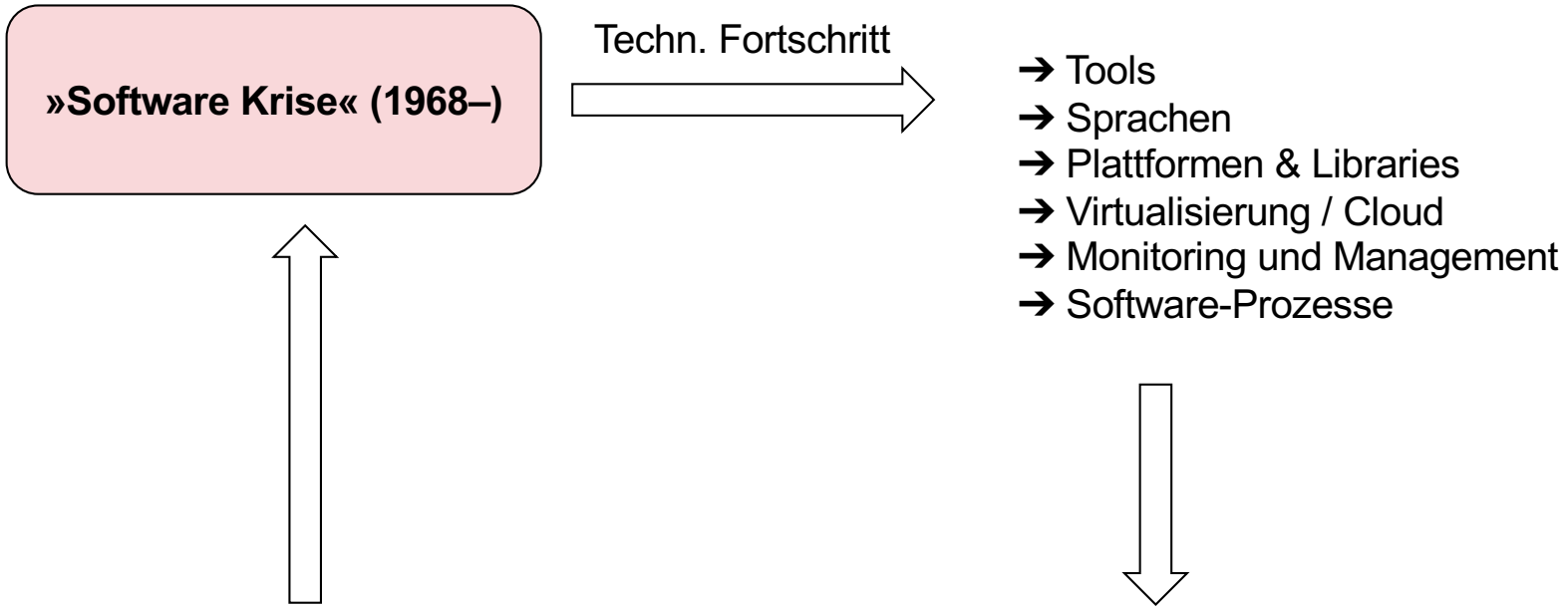
Alexander Schatten
aschatten@sba-research.org
www.schatten.info

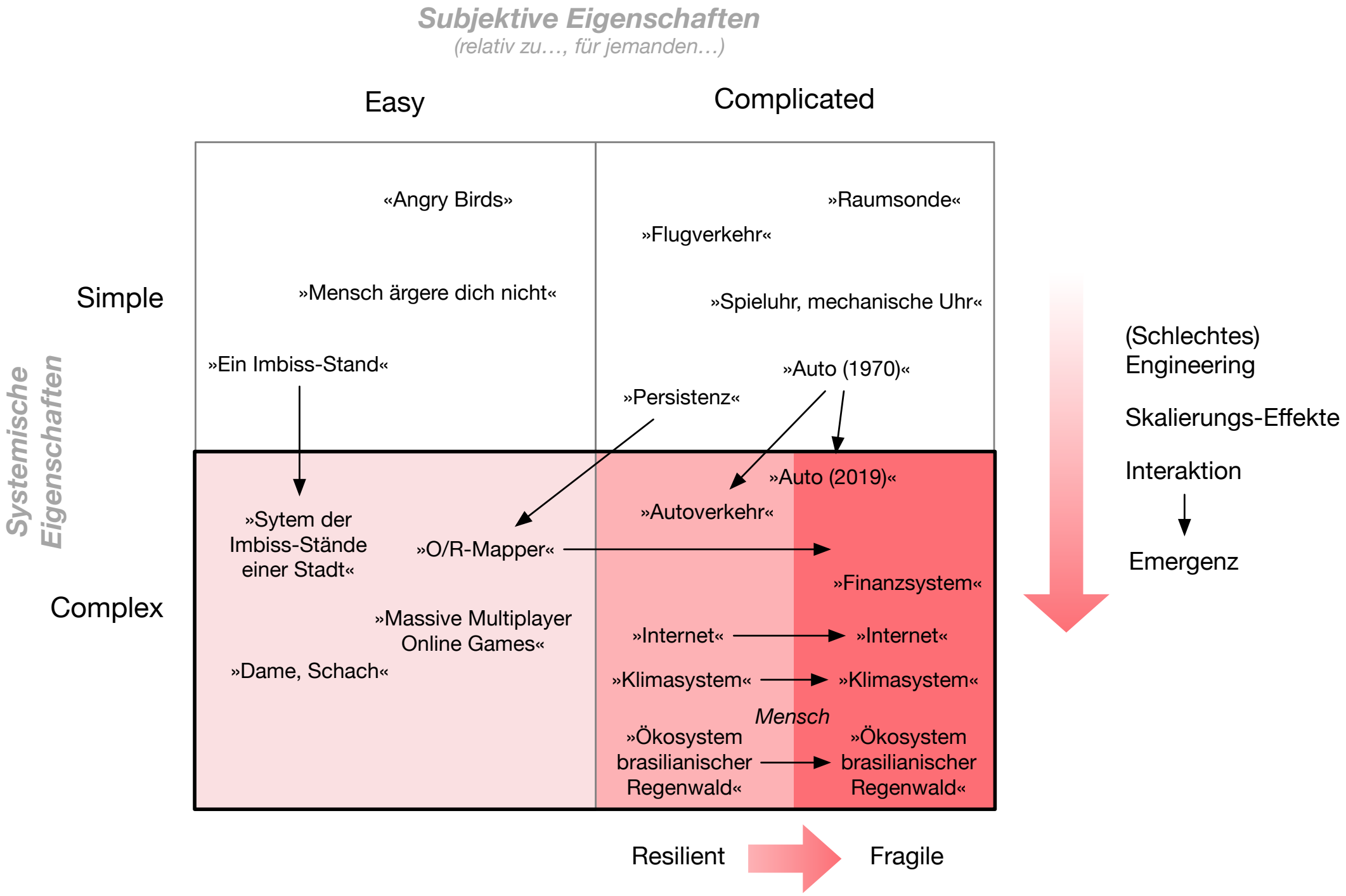
„Die Selektion hat uns Anschauungsformen gemäß den Aufgaben in noch höchst einfachen Lebensbereichen eingebaut. Und mit Anschauungen von gestern unterwerfen wir uns eine Welt von morgen.“

Rupert Riedl, “Evolution und Erkenntnis”, Piper (1985)


Komplexität
Problemfeld
Kontrollverlust
Qualitätsprobleme
Kosten
Security-
Probleme
Mangel an
Fortschritt
Mangelnde Stabilität
bzw. Resilienz
…

»Balance des Schreckens«




Subjektive Eigenschaften
(relativ zu…, für jemanden…)
Easy
Complicated
Simple
Complex
Systemische Eigenschaften
«Angry Birds»
»Raumsonde«
»Flugverkehr«
»Mensch ärgere dich nicht«
»Spieluhr, mechanische Uhr«
»Ein Imbiss-Stand«
»Auto (1970)«
»Persistenz«
»Auto (2019)«
»Autoverkehr«
»Sytem der Imbiss-Stände einer Stadt«
»O/R-Mapper«
»Finanzsystem«
»Massive Multiplayer Online Games«
»Internet«
»Internet«
»Dame, Schach«
»Klimasystem«
»Klimasystem«
Mensch
»Ökosystem brasilianischer Regenwald«
»Ökosystem brasilianischer Regenwald«
(Schlechtes) Engineering
Skalierungs-Effekte
Interaktion
Emergenz
Resilient
Fragile

Boot Camp: »Beyond Coding«

- **Weiterentwicklung** für in den Grundlagen erfahrene und motivierte Entwickler und Techniker – *keine »Basisschulung von der Stange«*
- **9 Wochen interaktiver Co-Working Space** mit gemeinsamen **thematischen Schwerpunkten und Trainings**, die für Unternehmen in hohem Maß **finanziell unterstützt** werden
- Kleine Gruppen, Anwendung des gelernten auf die eigenen Probleme und Austausch mit anderen
- **Schwerpunkte**
  - Software Development Strategies
  - Sustainability in Software
  - New Technologies
- Antrag bis **29.5.2019**

Was nun?

Kontakt
Dr. Alexander Schatten
SBA-Research
aschatten@sba-research.org
www.schatten.info

- Beratung (auch) bei systemischen Problemen im
  - Software Engineering und
  - Management von Softwaresystemen
- Bootcamp – »Beyond Coding« – **29.5.2019**