Karlsruher Institut für Technologie

Proseminar: Optimierungsansätze für Adaptionsstrategien in SAS

Betreuerin: Dipl.-Inform. Martina Rapp

Tim Engbrocks | 22. 07. 2021

Inhaltsverzeichnis

Die Komplexität von Software

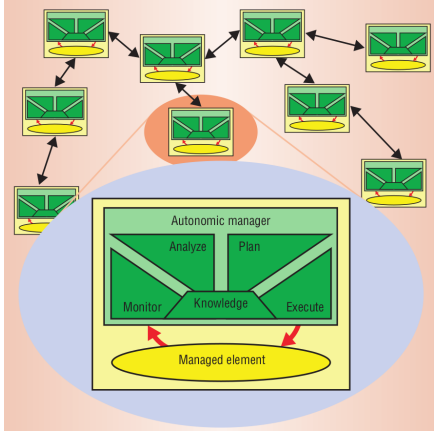
Die Komplexität von Software steigt stetig.
„The looming software crisis “- IBM, 2001

- Moderne Sprachmodelle:
  - GPT-3 von OpenAI: 175 Milliarden Parameter [Nvidia 2020]
  - Turing-NLG von Microsoft: 17 Milliarden Parameter [Microsoft 2020]
- Teslas Autopilot AI sagt 10.000 Parameter vorraus [Tesla 2021]

Netflix Microservice Architektur

[Evans 2016]

Die Lösung: Autonomes Management



[Kephart und Chess 2003]

Definition

Selbstadaptives System

Ein System, das sich autonom steuert durch das:

- *Beobachten* des Kontextes
- *Analysieren* von Veränderungen
- *Planen* von Anpassungen
- *Ausführen* von Anpassungen

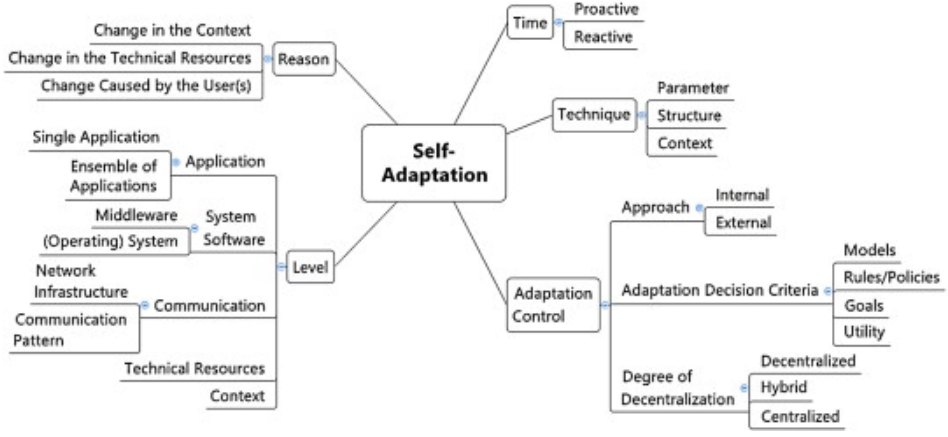
Beispiel: kommerzielles Web Angebot

- Szenario: System Administrator eines kommerziellen Web Angebots.
- Tägliche Aufgabe: System Parameter X basierend auf Metrik Y anpassen.

Dieses Szenario kann von einem Selbstadaptiven System profitieren:

- Generelle Anpassungs Regel: Wenn die Metrik Y den Schwellwert Z übersteigt: Passe den System Parameter X an
- Beispiel: Wenn die Serverlast zu hoch wird: Starte einen zusätzlichen Server

Selbstadaptive Systeme klassifizieren



[Krupitzer u. a. 2015]

Grenzen von Selbstadaptiven Systemen

Karlsruher Institut für Technologie

Unsicherheit

Änderungen im Kontext $\Rightarrow$ Unerwartete Ergebnisse von Anpassungen

Optimierungen für Selbstadaptive Systeme

Der meist verwendete Optimierungsansatz:

- Dynamisch Adaptations Regeln anpassen

Viele Ansätze verwenden maschinelles lernen, um neue Regeln zu generieren oder bestehende anzupassen.

Andere Aspekte des Systems können auch optimiert werden:

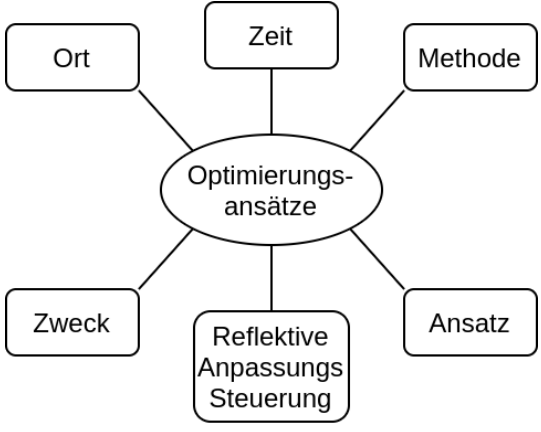
- Adaptations Steuerung: Wie
- Ebene: Wo
- Methode: Was

Eine Klassifizierung entwickeln

Die Klassifizierung basiert auf drei Konzepten:

- Reflektive Systeme [Weyns u. a. 2012]
- Die 6W Fragen [Salehie und Tahvildari 2009]
  - Wieso, Wo, Wann, Was, Wer und Wie
- Die Klassifizierung von Selbstadaptiven Systemen [Krupitzer u. a. 2015]

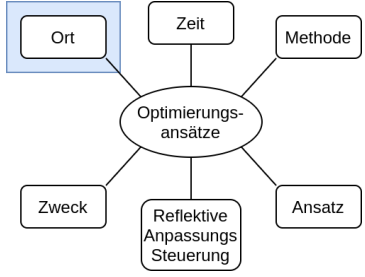
Vorschlag einer Klassifizierung



Klassifizierung - Ort

Ort

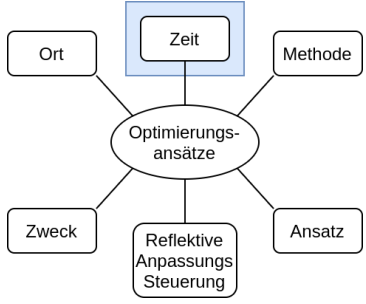
- Adaptations Steuerung
- Ebene
- Methode



Klassifizierung - Zeit

Zeit

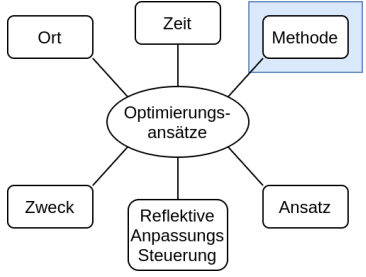
- Entwurfsphase
- Laufzeit / Online Phase
- Offline Phase



Klassifizierung - Methode

Methode

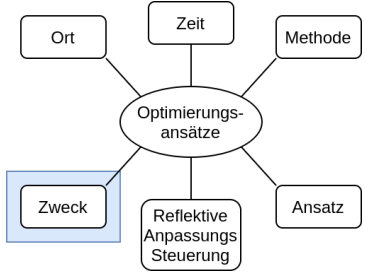
- Adaptations Regeln
- Wissen
- Adaptations Methode
- Ebene



Klassifizierung - Zweck

Zweck

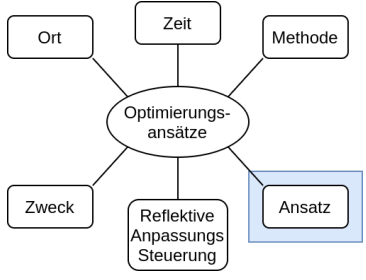
- Vorhandenes Wissen anpassen
- Adaptations Regeln an die neue Umgebung anpassen
- Ein Ziel erfüllen
- Eine Nutzenfunktion maximieren/minimieren



Klassifizierung - Ansatz

Ansatz

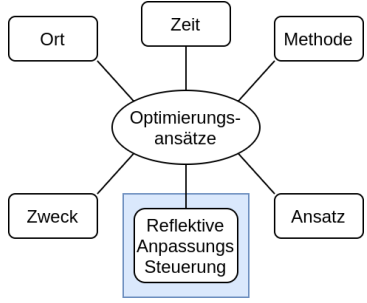
- Intern / Extern
- Grad der Dezentralisierung:
  - Komplett zentral
  - Komplett dezentral
  - Sowohl zentral als auch dezentral



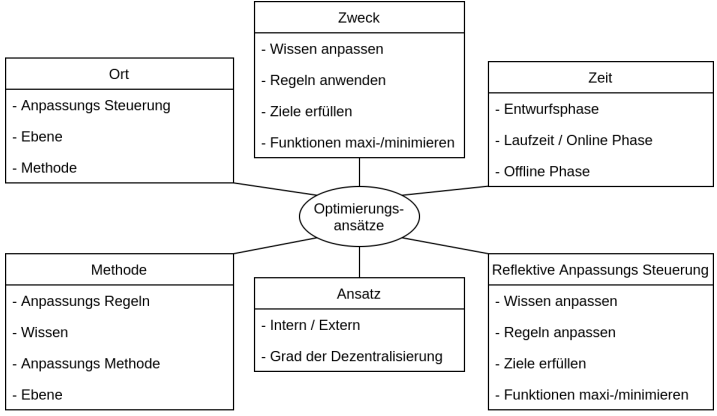
Klassifizierung - Reflektive Anpassungs Steuerung

Reflektive Anpassungs Steuerung

- Modelle anpassen
- Adaptations Regeln anpassen
- Ziele erfüllen
- Nutzenfunktionen maximieren/minimieren



Vorschlag einer Klassifizierung



Schlussfolgerungen

Nur wenige Optimierungsansätze verwenden:

- Zeit: Entwurfsphase und Offline Phase
- Ort: Ebene und Methode
- Ansatz: Dezentral

Weitere Forschung

Folgende Bereiche benötigen weitere Forschung:

- Die vorherigen Dimensionen erforschen
- Eine Methode, um den Nutzen von Optimierungsansätzen zu quantifizieren
- Selbstadaptive Systeme und Optimierungsansätze, die unabhängig von der Domäne sind

Literatur I

[1] Josh Evans. „Josh Evans, QCon 2016“. In: (2016). Accessed: 2021-07-01. URL: https://www.infoq.com/presentations/netflix-chaos-microservices/.

[2] Jeffrey Kephart und David Chess. „The vision of autonomic computing“. In: *Computer* 36.1 (2003), S. 41–50. DOI: 10.1109/MC.2003.1160055.

[3] Christian Krupitzer u. a. „A survey on engineering approaches for self-adaptive systems“. In: *Pervasive and Mobile Computing* 17.B (2015), S. 184–206. DOI: 10.1016/j.pmcj.2014.09.009.

[4] Microsoft. „Turing-NLG by Microsoft“. In: (2020). Zugriff: 01.07.2021. URL: https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/.

[5] Nvidia. „OpenAI Presents GPT-3, a 175 Billion Parameters Language Model“. In: (2020). Zugriff: 01.07.2021. URL: https://developer.nvidia.com/blog/openai-presents-gpt-3-a-175-billion-parameters-language-model/.

Literatur

Literatur II

[6] Mazeiar Salehie und Ladan Tahvildari. „Self-adaptive software: Landscape and research challenges“. In: *ACM Transactions on Autonomous and Adaptive Systems* 4.2 (2009), S. 1–42. DOI: 10.1145/1516533.1516538.

[7] Tesla. „Tesla Autopilot AI“. In: (2021). Zugriff: 01.07.2021. URL: https://www.tesla.com/de_DE/autopilotAI.

[8] Danny Weyns, Sam Malek und Jesper Andersson. „FORMS: Unifying reference model for formal specification of distributed self-adaptive systems“. In: *ACM Transactions on Autonomous and Adaptive Systems* 7.1 (2012), S. 1–61. DOI: 10.1145/2168260.2168268.