TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik

TECHNISCHE BERICHTE
TECHNICAL REPORTS

ISSN 1430-211X

TUD-FI16-03-Oktober 2016

Prof. Dr. Frank J. Furrer, Georg Püschel (Eds.)
Institut für Software- und Multimediatechnik

Autonomic Computing:
State of the Art - Promises - Impact

Technische Universität Dresden
Fakultät Informatik
D–01062 Dresden
Germany
URL: http://st.inf.tu–dresden.de/

DRESDEN concept
Exzellenz aus Wissenschaft und Kultur

Hauptseminar im Sommersemester 2016

Prof. Dr. Frank J. Furrer

Autonomic Computing

State of the Art – Promises – Impact

Editors: Prof. Dr. Frank J. Furrer, Georg Püschel

Technische Universität Dresden
Technical Report TUD-FI16-03-Oktober 2016

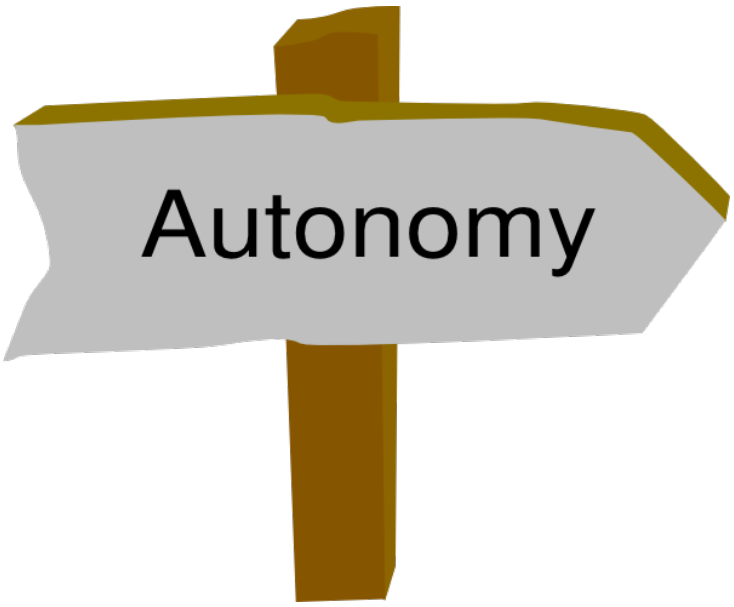

Table of Contents

Introduction **5**

1 What Knowledge Does a Taxi Need? – *Overview of Rule Based, Model Based and Reinforcement Learning Systems for Autonomic Computing*
Anja Reusch **11**

2 Chancen und Risiken von Virtual Assistent Systemen
Felix Hanspach **23**

3 Evolution einer Microservice Architektur zu Autonomic Computing
Ilja Bauer **37**

4 Mögliche Einflüsse von autonomen Informationsdiensten auf ihre Nutzer
Jan Engelmohr **49**

5 The Benefits of Resolving the Trust Issues between Autonomic Computing Systems and their Users
Marc Kandler **61**

Introduction

Prof. Dr. Frank J. Furrer

Context

Software has never been as important as today – and its impact on life, work and society is growing at an impressive rate. We are in the flow of a software-induced transformation of nearly all aspects of our way of life and work ([4], [5]). The dependence on software has become almost total. Malfunctions and unavailability may threaten vital areas of our society, life and work at any time.

The two massive challenges of software are one hand the complexity of the software and on the other hand the disruptive environment.

Complexity of the software is a result of the size, the continuously growing functionality, the more complicated technology and the growing networking. The unfortunate consequence is that complexity leads to many problems in design, development, evolution and operation of software-systems, especially of large software-systems.

All software-systems live in an environment. Many of today's environments can be disruptive and cause severe problems for the systems and their users. Examples of disruptions are attacks, failures of partner systems or networks, faults in communications or malicious activities.

Traditionally, both growing complexity and disruptions from the environment have been tackled by better and better software engineering. The development and operating processes are constantly being improved and more powerful engineering tools are introduced. For defending against disruptions, predictive methods – such as risk analysis or fault trees – are used. All this techniques are based on the ingenuity, experience and skills of the engineers!

However, the growing complexity and the increasing intensity of possible disruptions from the environment make it more and more questionable, if people are really able to successfully cope with this raising challenge in the future. Already, serious research suggests that this is not the case anymore and that we need assistance from the software-systems themselves!

Here enters "**autonomic computing** ([1], [2], [3])" – A promising branch of software science which enables software-systems with self-configuring, self-healing, self-optimization and self-protection capabilities. Autonomic computing systems are able to re-organize, optimize, defend and adapt themselves with no real-time human intervention. Autonomic computing relies on many branches of science – especially computer science, artificial intelligence, control theory, machine learning, multi-agent systems and more.

Autonomic computing is an active research field which currently transfers many of its results into software engineering and many applications. This Hauptseminar offered the opportunity to learn about the fascinating technology "autonomic computing" and to do some personal research guided by a professor and assisted by the seminar peers.

Seminar Work

The seminar worked on the central theme:

> *Which are the state of the art, the promises, and the impact of Autonomic Computing?*

Each participant chose one of the 3 questions:

Q1 Which are the promising software architectures and software technologies for Autonomic Computing?

Q2 How does Autonomic Computing enable future applications?

Q3 What is the impact of Autonomic Computing on people, work and society in 2025?

The Hauptseminar had 3 seminar days:

- An introduction day: Autonomic Computing was introduced in a lecture by Professor Dr. Frank J. Furrer, and the parts of the Hauptseminar (Paper, presentation) have been defined,
- Individual, guided research in the selected area and authoring of a scientific paper. Feedback from peer reviewers,
- A first seminar day: The participants presented their results and received feedback from the audience,
- Improvement of the paper and the presentation, based on the peer feedback,
- A second seminar day: The participants presented their improved results and received feedback from the audience,
- Delivery of the final paper (which are included in this proceedings volume).

The participants learned: (a) to do focused research in a specific area ("Autonomic Computing"), (b) to author a scientific paper, (c) to improve their LaTeX expertise, (d) to experience the peer-review process and (e) to hold convincing presentations, and (f) to benefit from a considerable broading of their perspective in the field of technology, software, applications, and impact.

As a final outcome of the seminar, this proceedings volume – including all the papers produced by the participants – has been assembled and made available in electronic form to anybody interested.

Seminar Schedule

Kick-Off Meeting (Introduction):	Wednesday, **April 20, 2016** / 11:10 – 12:40 in APB/INF 2101
Seminar Day 1:	Wednesday, **June 8, 2016** / 09:20 – 10:50 & 11:10 – 12:40 in APB/INF 2101
Seminar Day 2:	Wednesday, **July 13, 2016** / 09:20 – 10:50 & 11:10 – 12:40 in APB/INF 2101

References

1) Mandatory Reading

[1] The seminal work:
IBM Research Paper, 2001: **Autonomic Computing** – ***IBM's Perspective on the State of Information Technology***. Downloadable from: `http://people.scs.carleton.ca/~soma/biosec/readings/autonomic_computing.pdf` [last accessed: 2.2.2016]

[2] Introduction to the Architecture:
IBM White Paper: ***An architectural blueprint for autonomic computing***. 3rd edition, June 2005. Downloadable from: `http://www-03.ibm.com/autonomic/pdfs/AC%20Blueprint%20White%20Paper%20V7.pdf` [last accessed: 2.2.2016]

[3] The fundamental knowledge:
Philippe Lalanda, Julie A. McCann, Ada Diaconescu: **Autonomic Computing** – ***Principles, Design and Implementation***. Springer-Verlag, London UK, 2014. ISBN 978-1-4471-5006-0

2) Additional Reading

[4] Context:
Erik Brynjolfsson, Andrew McAfee: **The Second Machine Age** – ***Work, Progress, and Prosperity in a Time of Brilliant Technologies***. W.W. Norton & Company, Inc., N.Y., USA, 2014. ISBN 978-0-393-23935-5

[5] Context:
Klaus Schwab: **The Fourth Industrial Revolution**. World Economic Forum, Geneva, 2016. ISBN 978-1-944835-00-2

Additional Autonomic Computing References

[6] Hausi A. Müller, Liam O'Brien, Mark Klein, Bill Wood: **Autonomic Computing**. Carnegie Mellon University, Technical Note CMU/SEI-2006-TN-006, 2006. Downloadable from: `http://www.sei.cmu.edu/reports/06tn006.pdf` [last accessed 14.1.2016]

[7] Phan Cong-Vinh (Editor): **Formal and Practical Aspects of Autonomic Computing and Networking** – ***Specification, Development, and Verification***. Premier Reference Source, Information Science Reference Publishing, 2011. ISBN 978-1-60960-845-36]

[8] Salim Hariri, Manish Parashar (Editors): **Autonomic Computing - *Concepts, Infrastructure, and Applications***. CRC Press Inc., Boca Raton, USA, 2006. ISBN 978-0849393679

[9] Richard Murch: **Autonomic Computing**. IBM Press, Prentice Hall PTR, NJ, USA, 2004. ISBN 978-0-13-315319-3

[10] Daniel A. Menascé, Jeffrey O. Kephart: **Autonomic Computing** – ***Editor's Introduction***. IEEE Computer Society, 2007. Downloadable from: `https://www.computer.org/csdl/mags/ic/2007/01/w1018.pdf` [last accessed 12.3.2016]

[11] Ozalp Babaoglu, Mark Jelasity, Alberto Montresor, Christof Fetzer, Stefano Leonardi, Aad van Morsel, Maarten van Steen (Editors): **Self-star Properties in Complex Information Systems** – ***C*onceptual and Practical Foundations**. Lecture Notes in Computer Science / Theoretical Computer Science and General Issues, LNCS 4360, Springer Verlag Heidelberg, 2008. ISBN 978-3-540-26009-7

[12] Robert Morris: **Autonomic Computing**, IBM Almaden Research Center, 2001. Downloadable from: `http://www.almaden.ibm.com/almaden/talks/Morris_AC_10-02.pdf` [last accessed 12.3.2016]

[13] IEEE International Conference on Cloud and Autonomic Computing (ICCAC): `http://www.autonomic-conference.org/` [last accessed 12.3.2016]

[14] Simon Dobson, Roy Sterritt, Paddy Nixon, Mike Hinchey: **Fulfilling the Vision of Autonomic Computing**. IEEE Computer Society, January 2010. Downloadable from: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.376.1739&rep=rep1&type=pdf` [last accessed 12.3.2016]

Participants

Five participants took place in the Hauptseminar (see following table):

Name	**Topic**	**Paper Title**
Anja Reusch anja.reusch@tu-dresden.de	**Q1**	What Knowledge Does a Taxi Need? – *Overview of Rule Based, Model Based and Reinforcement Learning Systems for Autonomic Computing*
Felix Hanspach f.hanspach@gmail.com	**Q3**	Chancen und Risiken von Virtual Assistent Systemen
Ilja Bauer Ilja.Bauer@mailbox.tu-dresden.de	**Q1**	Evolution einer Microservice Architektur zu Autonomic Computing
Jan Engelmohr jan.engelmohr@mailbox.tu-dresden.de	**Q3**	Mögliche Einflüsse von autonomen Informationsdiensten auf ihre Nutzer
Marc Kandler marc.kandler@tu-dresden.de	**Q3**	The Benets of Resolving the Trust Issues between Autonomic Computing Systems and their Users

Their papers follow here in full length.

What Knowledge Does a Taxi Need

Overview of Rule Based, Model Based and Reinforcement Learning Systems for Autonomic Computing

Anja Reusch

Technische Universität Dresden, Germany

Abstract. Choosing the correct formalism to implement knowledge is a basic problem in autonomic computing. This paper describes and compares the approaches rule-based, model-based autonomic computing systems and systems that learn and gives an example applying each system to a autonomic taxi. We analyzed the advantages and disadvantages and found that rule-based systems can be used where memory and processing time is limited. An area of application for model-based systems are system that need to consider the past and, therefore, need an internal state. Reinforcement Learning can be used where it is necessary to learn new policies and adapt them flexibly. It is also a way to use an autonomic software system in an completely unknown environment.

1 Introduction

In the field of autonomic computing, it is important for a system to make useful decisions to guarantee the self*-properties. Making decision requires knowledge and an effective representation of it. There are several approaches how to represent knowledge, but to find the best-fitting one for a given purpose is a basic issue as we can see in the following example.

During this work we are going to use the example of an autonomic taxi whose purpose it is to carry passengers without human input to their desired destination. This taxi needs to collect and evaluate many information, e.g. data about the current position and distance to other objects, weather data, a map of the environment to compute the best route, data of the traffic regulations and more. All these data need to be stored and handled with a proper formalism. For the map it would be suitable to use a graph, but the traffic regulations could be displayed as a Condition-Action rules. Both approaches are not fitting for simple information about the current state like the position or weather data. In this paper we give an overview of the following formalisms of knowledge representation.

The simplest approach is a reflex agent which only uses Condition-Action rules to determine the next action. The model based autonomic computing system is also a strong tendency where the reflex agent keeps information about the state of the managed element. Another promising approach, reinforcement learning, is grounded in decision theory and tries to learn optimal policies. The simple reflex rules might not be flexible enough for complex problems, while Reinforcement Learning could be too overdone. We evaluate and investigate their advantages and disadvantages and, finally, give recommendations for the fields of application of the given approaches.

2 Background

2.1 State of the Art

In recent years there has been a huge effort to promote the development process in autonomic computing. There has been literature about autonomic computing in general which presents some representation approaches, but does not focus on them or compare them [1]. But on the other hand there has been work about knowledge representation in a broader sense, which does not consider the application of an autonomic system [2]. Some literature has given a short and brief explanation of the model based approach [3]. Research has been done in the management of servers and groups of servers using model based systems [4] [5]. In [5] the architectural model has been used as a basis for error detection and repair. They have also presented a framework of action functions, compared to goal and utility function policies with respect to, among others, CPU resource and response time. Reinforcement Learning has been also applied to autonomic computing. The approach has been used in an autonomic manager of a prototype data center [6]. We refer to the issues that have been found there in the Reinforcement Learning section in this work.

2.2 MAPE-K Architecture

MAPE-K is a reference architecture designed by IBM to structure automatic software systems and is an acronym for monitor, analyze, plan, execute and knowledge [7]. These components are combined in the autonomic manager which manages the autonomic element. This can be any hardware or software element. One run in the MAPE-K loop contains the monitoring, analyzing, planning and execution functions (Fig. 1).
During the monitoring function the system captures properties about the managed element by collecting information. The information is provided by sensors of the managed element or through specific interfaces provided by the computing environment. Interpreting these data the autonomic manager is able to decide whether there is a need to apply changes to the system.
In the analysis component the collected data is interpreted to evaluate the current state of the system. Therefore, the system's knowledge is involved in this progress. The analysis also determines a better state for the managed elements.
Now the planning function deals with finding the best way to reach this desired state. To do so, the component uses knowledge and reasoning to decide on the next action. Therefore, the planner contains a set of actions the managed element can perform.
Execution is the last activity of the MAPE-K loop and deals with the implementation of the actions plan determined by the planner. It uses the effectors provided by the managed element to execute the concrete action [1].

2.3 Example

During this work we are going to use the example of an autonomic taxi which is adapted from [8]. The goal of our autonomic taxi is to carry passengers to their desired destination. In order to reach the destination the taxi should also drive safely, i.e. for example not be

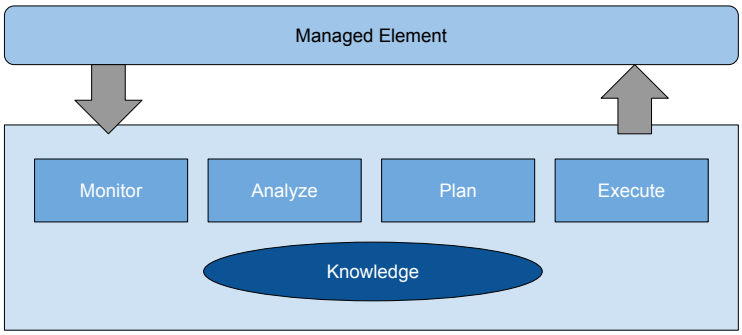

Fig. 1. Scheme of MAPE-K

involved in car accidents.
The environment of our car consists of roads in cities and highways between them, other cars, people crossing the roads and traffic lights. At the end of the trip our taxi gets tip in terms of money from the passengers.
The taxi has sensors like distance sensor, rain sensor, compass, cameras in every direction, which can be used to get information of the environment. We assume the taxi is build with the MAPE-K architecture.

3 Rule-based Autonomic Computing Systems

3.1 Overview

Rule-based autonomic computing systems define sets of *Condition-Action (CA) rules*, so called *reflex actions*. These rules are derived from system and business goals and describe the adaptation plans of the system. The system's knowledge is expressed as statements modeling the current state of the managed elements.

3.2 Architecture

Using a rule-based system the MAPE-K architecture can be modified in the following way as it is shown in Fig. 2. First, the system gets data from the monitoring function. During the analyze function we interpret the raw sensor data. So we receive the state of the system and the environment and can check whether a condition is fulfilled. During the planning we get the action of the satisfied condition and, finally, execute this action. So we do not need to change the monitoring or execution component.

3.3 Example

If our Autonomic Taxi is working with a rule-based system it would need some rules. These could be rules for the traffic regulations, e.g. "Stop at red traffic lights." or also

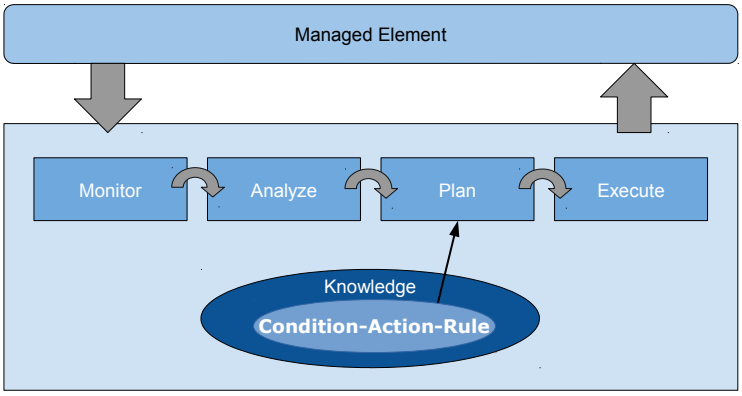

Fig. 2. Scheme of MAPE-K for Rule-Based System (adapted from [8] and [1])

rules which guarantee the safety of the passengers and the car itself, for example "Do not crash into other cars." or "Turn on lights at night." Now we need to modify these rules such that the system can understand and use them, that means we split them into two parts, the condition part and the action part. Our just mentioned rule are changed to these rules:

```
<condition> Traffic light is red
<action> Brake
<condition> Car in front brakes
<action> Brake
<condition> is Night
<action> Turn Car Headlights on
```

The rule in an autonomic system can be executed during the MAPE-K loop as illustrated in Fig. 3.

3.4 Evaluation

The advantage of the rule-based approach is clearly its promptness and simplicity. Without considering the past or future the decisions only depend on current state and are therefore quickly made. This also reduces the amount of knowledge to be expressed and maintained. This advantages come out clearly in a dangerous situation where we have to decide quickly what to do next. If the car in front of us suddenly needs to brake, it is important for us to stop immediately,too so that we do not crash into that car. Nevertheless the CA-rules are less smart, because they do not depend on the past resulting in for example *state flapping*, the phenomenon of flipping forward and backwards between states. This problem can be illustrated in the following simple example: Our Autonomic Taxi is accelerating until a speed of 50 km per hour with gear 3. At the time it reaches 50 km per hours, the autonomic manager tells it to change to gear 4. Since we are driving

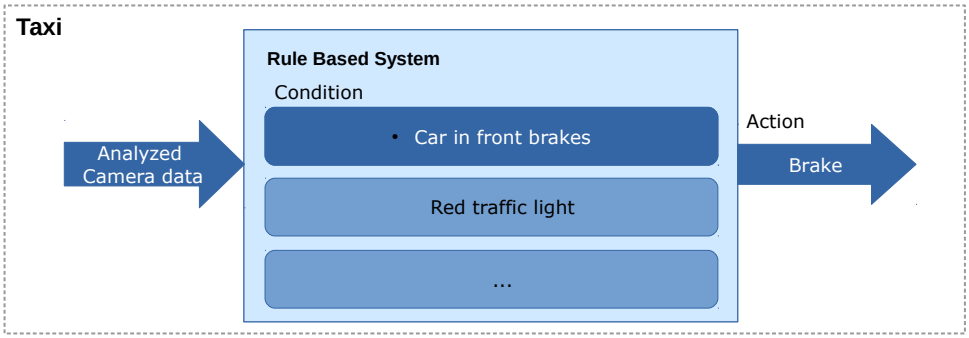

Fig. 3. Example of a Rule-Based System

in a city we do not want to accelerate anymore so we try to keep the speed. The car's velocity is around 50 km per hour, but varies a bit. If it gets lower than 50 km per hour, we need to gear down according to the rules. If the car gets higher than 50 km per hour we need to gear up again. The car gears up and down all the time which is not the desired behavior causing maybe damage to the clutch and the gear boxes. There can also be conflicts between rules which all have satisfied conditions, but are contradicting. For example: In rain drive 50 km per h, but on the highway drive 140 km per hour. What should the taxi do on the highway while it is raining? These conflicts might not be detected at the time of writing these rule, but only at runtime. In such a case, humans have to intervene which is by the nature of autonomic computing undesirable.

4 Model-based Autonomic Computing Systems

4.1 Overview

A *model-based autonomic computing system* keeps information about the managed element or the environment which is updated with for example sensor data. These information is called a *model*. Models are built with a clear purpose in mind and created in order to solve a problem or perform a specific task. Therefore, it is necessary for models to have clear semantics and be unique to avoid misunderstandings. Models are usually expressed as graphs, where nodes are concepts such as properties or files and an edges is a relationship between them. In order to get models readable by machines, we have to formulate a specification language that is clear, precise and unambiguous. An example for a specification languages that uses graphs is *UML*[1].

4.2 Architecture

The analyze function of the MAPE-K loop needs to be changed a lot for the model-based system. After measuring the sensor data, we need to interpret them. The system uses

information about the *last action* and the *state of itself and the world* to compute the new world. Therefore it needs knowledge about the world, the state and how a concrete action changes both. These three components are what we call a model. If we have analyzed how the new world looks like, we can find a new action during the planning. To do so, we can use condition-action rules as we did in the rule based system. But we can also use other approaches like utility or goal based policies which are not part of this work. Fig. 4 shows the structure of such a system.

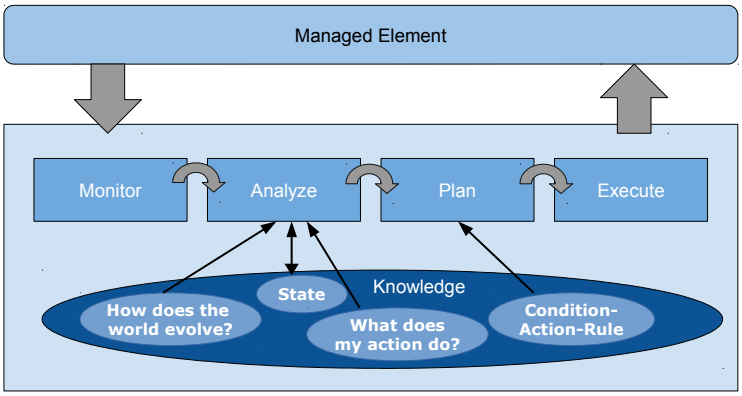

Fig. 4. Scheme of MAPE-K for Model Based Systems (adapted from [8] and [1])

4.3 Example

A model of our taxi's world could include a mathematical description (formulas) how the position or velocity of the car changes if it moves with a certain velocity or acceleration, or a map of the environment with cities, villages and roads between them.

Our taxi is in an specific state in which it can perform particular actions depending on that state. For example consider the taxi drives on an highway and is just switched to the left lane. The state of the taxi includes that the velocity is 100 km per hour as we can see in Fig. 5. In that case we want to accelerate the car and the autonomic system chooses the action to do so. After accelerating, the system computes that the velocity is now (for example) 150 km per hour.

4.4 Evaluation

Keeping information about the managed elements and changing them with time allows us to hold an inner state. Due to this inner state the model based system is able to make predictions about future behavior of the system. The system uses these data and the information collected about the past action to decide what to do next. Therefore, it can adapt to changes of the environment. For example, if a street is temporarily under

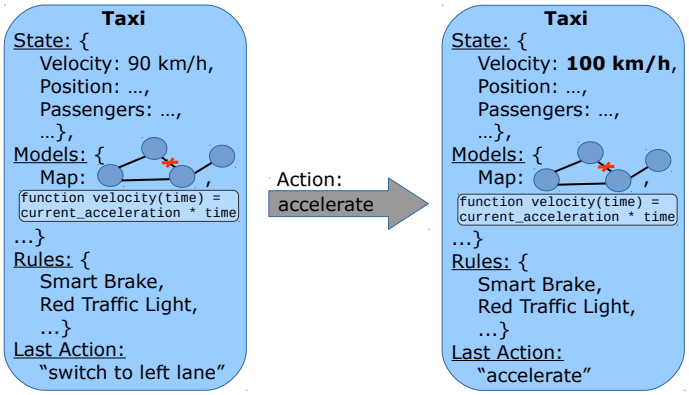

Fig. 5. Example of a Model-Based System

construction, we need to block this route in the map. Because no single model can capture all the information needed in a huge system to solve all kind of problems, we need many different models that are build for the same reality, but handle different aspects. This concept improves the problem-solving and we get smaller, simpler models. An issue is to synchronize these different models. In our case such an issue could be to update our models, if our position has changed. We need to notify the map, the model which stores our distance to other objects, maybe a model of the current route to the destination, or others, because they all should now our current position to work well. This could be done with a *central model*, the state of the taxi, and the other different models related to it. Models are always incomplete knowledge, because the managed artifacts are only partially observable. Thus, decisions based on that models have to be backed up by human beings. Another disadvantage appears from the wish of changing the business or domain oriented goals of the system. They are embedded in the problem solver and are therefore neither explicitly given nor they can be manipulated explicitly. Thus, changing a goal means changing the code of the problem solver, which is always a risky task[1].

5 Reinforcement Learning

5.1 Overview

The purpose of autonomic computing system that can learn is to learn about their actions and to change knowledge supporting reasoning accordingly and, in some cases, the reasoning itself. The system has the ability to evaluate the usefulness of actions and to modify its knowledge based on that evaluation. In this case modifying knowledge means changing the models or utility function used to compare system states. One of these learning techniques is the *Reinforcement Learning*. Reinforcement Learning purposes to learn optimal management policies without explicit given system models or

domain specific initial knowledge. It derives relations and structures between agents and components that were unknown before.

One step in the progress of Reinforcement Learning would be:

1. Observe the state of the system and environment
2. Perform some action which are legal in the current state
3. Receive a reward
4. Observe the new state

The received *reward* expresses the usefulness of the performed action and is a value which we want to maximize. During the whole process reinforcement will add weights to the decision process and, thus, influence the trade-off in future[6].

5.2 Architecture

Applying Reinforcement Learning to an autonomic manager requires to add several components. There needs to be a *critic* component which is responsible for collecting information from the monitoring component about the reward we got from an action. For the analyzing use again the models to know how the states have been changed after executing an action. The planning component can use rules and goal or utility functions. We also have two new components: the *learning element* and the *problem generator*. The learning element uses the feedback from the critic and is responsible for the adaption of the knowledge of our autonomic manager. It can change the models or the planning component. The problem generator is responsible for the suggestion of new solutions which have not been considered so far. It adds therefore some randomness to the decision process. The system also needs a performance standard to know what reward is good and what is bad.

5.3 Example

Our taxi now uses Reinforcement Learning to improve its decisions. We can imagine the following situation: The taxi carries passengers from one city to another using the highway. The first time it does that it might not have a rule for the speed at the highway and tries to drive with 30 km per hour. As we can imagine the passengers are not really happy about that and at the end of the trip they give 0€ as tip. The autonomic manager can use the given tip as a reward and save the state and the value in the critic element. At the next time the taxi uses the highway it can get a suggestion from the problem generator which tells the planning element to drive with velocity 120 km per hour. After the trip the taxi got 5€ as tip. Because the performance standard says: "More tip is better.", the learning element can conclude that 120 km per hour at the highway is better then 30 km per hour. Thus it changes the planning element and adds a rule to drive 120 km per hour at the highway.

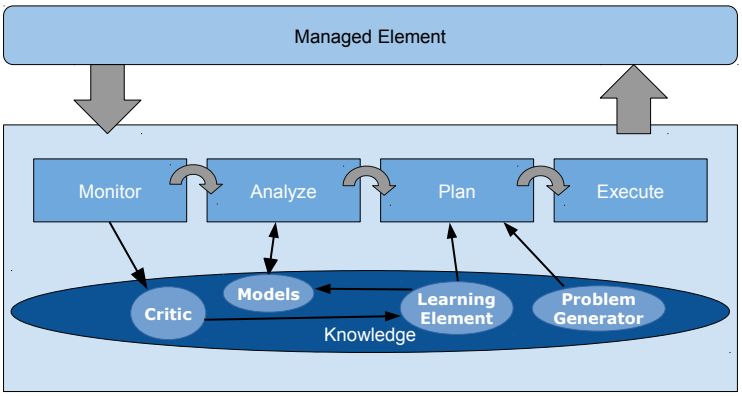

Fig. 6. Scheme of MAPE-K architecture using Reinforcement Learning (adapted from [8] and [1])

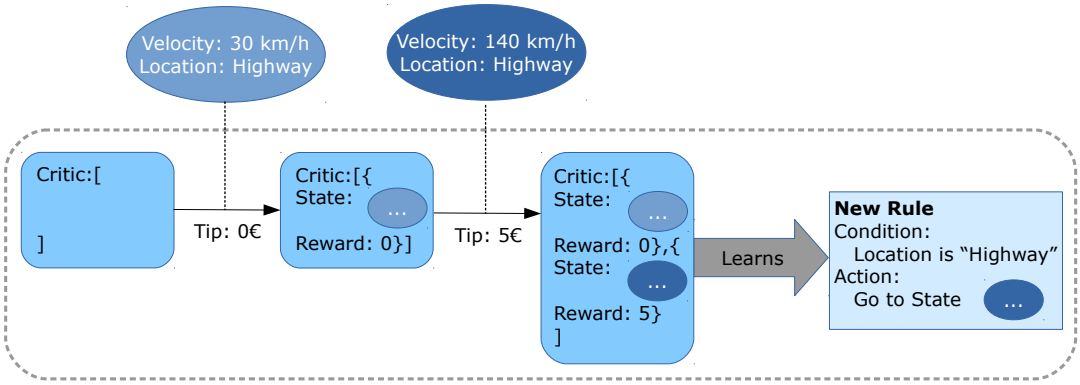

Fig. 7. Example of Reinforcement Learning

5.4 Evaluation

As we have seen in the example, the system can improve its actions according to the performance standard given before. If the autonomic manager "notices" that an other action would be better to get a better reward, he can adapt the models and rules. This means the system is also more flexible if changes occur and can decide on more useful action, but only if the critic element is trained. This is a small disadvantage of the system[6]: First, we need some time to train the systems decisions, because in the beginning we have an initial configuration for our models, which might not describe the current situation. Because the system can learn new policies, it is able to work in an completely unknown environment[8]. So if we would put our taxi on the moon, it as mentioned before would take some time, but in the end it would drive save. In the beginning the car would need to figure out how to actually drive with different material

on the ground or changed gravity. Then it should collect data from the cameras and distance sensors to build the model of a map. If the car now met passengers on the moon, it would drive save and tries to find the destination such that both goals are achieved. The policies learned by Reinforcement Learning can be characterized as reactive planners meaning that the decisions are made immediately without explicit search or forecasting of future states. Due to that, the system might not be able to find the most effective solution for problems that may require a more powerful approach of planning, but the autonomic agent is able to find an acceptable solution[6]. During the progress the critic element needs to store states and rewards for a lot of decisions. Hence, the system needs a large memory which is also fast accessible. Otherwise the system has to "forget" important decisions or rewards.

6 Findings and Recommendations

The rule-based autonomic computing system has the problem of conflicts between rules. Thus, it can only be used in small and easy understandable structures. Anyway, because of its lightweight this approach can be used in systems with limited processing power or little memory. An area of application could be for example autonomic software systems that run on micro-controllers. Furthermore, it does not need much time to make decisions which could be interesting for real-time multimedia application. It is not recommended to use rule-based autonomic software systems in an application that needs to make decisions depending on many factors, because the action-condition-rules can be to complex to handle and influence each other (state flapping). There is also work on action-condition-rules that consider past states, but currently it is not possible to build rule-based software systems that include past dependent rules.

The model-based autonomic computing systems have inner states whose advantage are the predictions of future behavior and the changes of the environment. That means the system can estimate what will happen if it performs a certain action. The systems models are flexible. They can be updated through the sensor measurements. Therefore the system can adapt to changes of the environment. An issue is still the adaptability of the system's goals. We also saw that synchronizing the different models of the system is an issue, but this is manageable, if considered at the beginning of construction. The model-based system can use the collected information to be more or less sensitive to changes in the data and, therefore, avoid state flapping in contrast to the rule-based system. These systems could be used for industrial robots. They have usually a fixed goal e.g. build the component according to the plan, but need to adapt to changes in the for example the construction plan.

The strengths of the autonomic computing system that uses Reinforcement Learning lie in the way of adaption of its knowledge. Because of its learning ability it can improve its decisions to fulfill the performance standard criteria. Furthermore it is able to work in an entire unknown environment, because it can evaluate the decisions made in the past. Therefore it needs intensive memory storage and processing usage. Systems that benefit from the power of the learning system could be virtual assistant systems or as already mentioned autonomic cars. The virtual assistant systems work in an initially unknown environment which means that they do not know anything about the preferences of their

user. First they need some time to learn. The same holds for autonomic cars. As we have seen we could use the learning capabilities for learning the preferences of the passengers, but also to detect routes with a higher potential of heavy traffic.

7 Conclusion

So far we introduced different approaches of knowledge representation which can be used in various kinds of autonomic systems. But now let us answer the question of this work: What knowledge does a taxi need? We have already seen how the approaches can be used during the trip of our example taxi. We found that the general system should be a Reinforcement Learning system such that the autonomic manager can learn what the passengers like best. That includes driving faster on the highways, more slowly in the city, but save all the time.

In addition the taxi needs models, for example one or more maps of the world to compute the correct route. Since the roads can be temporarily under construction it is useful that models are easily changeable. It is also important that the autonomic manager knows the impact of its actions to the real world. For that purpose models for the expected behavior of the car and the environment are necessary.

In cases of emergency it is absolutely obligatory for the system to react fast. Here we need our rule based system whose advantage is the promptness of its reactions. As shown on the example it system can break fast in case of a car stopping in front. Similarly the system can stop at red traffic lights or save the life of a person walking on the street.

We saw that we need to combine all tree systems to build a save and reliable complex autonomic computing system like the autonomic taxi. But as we showed in the previous section, this is not always necessary. It depends on the area of application what approach needs to be implemented.

8 Acknowledgment

This research was supported by Prof. Frank J. Furrer as my adviser. I would also like to thank my reviewers Ilja Bauer and Marc Kandler who provided comments that greatly improved the work.

References

1. P. Lalanda, J. A. McCann, and A. Diaconescu, *Autonomic Computing: Principles, design and implementation*. Springer Science & Business Media, 2013.
2. H. J. Levesque, "Knowledge representation and reasoning," *Annual review of computer science*, vol. 1, no. 1, pp. 255–287, 1986.
3. T. Vogel, S. Neumann, S. Hildebrandt, H. Giese, and B. Becker, "Model-driven architectural monitoring and adaptation for autonomic systems," in *Proceedings of the 6th international conference on Autonomic computing*. ACM, 2009, pp. 67–68.
4. L. Broto, D. Hagimont, E. Annoni, B. Combemale, and J.-P. Bahsoun, "Towards a model driven autonomic management system," in *Information Technology: New Generations, 2008. ITNG 2008. Fifth International Conference on*. IEEE, 2008, pp. 63–69.

5. S.-W. Cheng, D. Garlan, B. Schmerl, J. P. Sousa, B. Spitznagel, and P. Steenkiste, "Using architectural style as a basis for system self-repair," in *Software Architecture*. Springer, 2002, pp. 45–59.
6. G. Tesauro, "Reinforcement learning in autonomic computing: A manifesto and case studies," *Internet Computing, IEEE*, vol. 11, no. 1, pp. 22–30, 2007.
7. J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.
8. S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, *Artificial intelligence: a modern approach*. Prentice hall Upper Saddle River, 2003, vol. 2.
9. "An architectural blueprint for autonomic computing," *IBM White Paper*, 2006.
10. G. Antoniou and F. Van Harmelen, *A semantic web primer*. MIT press, 2004.
11. J. O. Kephart and W. E. Walsh, "An artificial intelligence perspective on autonomic computing policies," in *Policies for Distributed Systems and Networks, 2004. POLICY 2004. Proceedings. Fifth IEEE International Workshop on*. IEEE, 2004, pp. 3–12.
12. H. S. Nwana, "Software agents: An overview," *The knowledge engineering review*, vol. 11, no. 03, pp. 205–244, 1996.

Chancen und Risiken von Virtual Assistent Systemen

Felix Hanspach

Zusammenfassung. Intelligente Persönliche Assistenten sind autonome Softwaresysteme die auf natürlichsprachliche Eingaben setzen und schon heute eine Vielzahl von Aufgaben erfüllen können. Im Rahmen dieses Papers soll erörtert werden, welche Chancen und Risiken die gegenwärtige Entwicklung solcher IPA-Systeme aufzeigt. Dazu werden die Grundlagen erläutert, positive und problematische Aspekte erklärt und Lösungen für diese vorgestellt.

1 Einleitung

Die Flugbuchung durch eine simple Textnachricht im Smartphone. Uhrzeit, Strecke, Fluggesellschaft werden automatisch nach günstigem Preis und optimaler Zeitnutzung bestimmt. Die Buchung wird nur bestätigt, der Check-In wird ohne Zutun erledigt. Das Ticket kommt pünktlich am Vortag per E-Mail. Was klingt wie eine Zukunftsvision, könnte durch Intelligente Virtual Assistent Systeme schon bald zur Realität werden. Schon heute können Systeme wie *Siri*, *Google Now* oder *Cortana* einfache Anweisungen erledigen und Informationen abrufen.
Durch die Kombination von Eingaben durch Mitarbeitern, Lernalgorithmen und intelligenten Programmen werden die erfüllbaren Aufgaben immer komplexer. *Facebook M* wurde im August 2015 als ein solche hybrides Virtual Assistent System vorgestellt. Die Chancen die ein solches System mit sich bringen kann sind nahezu grenzenlos. Gleichzeitig birgt es allerdings auch Gefahren, da ein großer Einfluss auf Nutzer entsteht, der durch fragwürdige Geschäftsmethoden, falsch abgerufene Informationen oder uninformierte Benutzer ausgenutzt werden kann.

Dieses Paper soll die Chancen und Risiken von intelligenten Systemen, wie Personal Assistent Systemen, analysieren. Dazu wird im ersten Abschnitt

darauf eingegangen, was Recommender Systeme und Intelligent Personal Assistent Systeme sind und ein kurzer Überblick darüber gegeben wie sie funktionieren und wie diese zusammenspielen. Im darauf folgenden Abschnitt wird ein Ausblick auf eine mögliche Entwicklung und den damit zusammenhängenden Chancen und Risiken eingegangen. Abschließend werden zu den vorgestellten Problemstellungen Lösungsvorschläge angeboten und in einem abschließenden Fazit zusammengefasst.

2 Grundlagen von Intelligent Personal Assistent Systemen

Resultate von Intelligent Personal Assistent Systemen werden anhand von Daten, die allgemein bekannt, oder nutzerspezifisch sind ermittelt. Dabei haben sie eine ähnliche Rolle inne wie Recommender Systeme, welche aus dem Onlinehandel bekannt sind. In diesem Kapitel wird ein kurzer Überblick darüber gegeben was Intelligent Personal Assistent Systeme sind und wie Recommender Systeme als einfaches Beispiel für das Finden von Empfehlungen funktionieren.

2.1 Recommender Systeme

Recommender Systeme haben die Aufgabe aus Daten, die über einen Nutzer bekannt sind, Vorschläge zu generieren. Diese Empfehlungen sollen hierbei einerseits das Anwendererlebnis verbessern, andererseits betriebswirtschaftliche Vorteile für die Anbieter erzielen. Hierzu zählt die Erhöhung von Verkaufszahlen, der Verkauf von vielfältigeren Produkten und die Nutzerbindung erhöhen. Ein weiterer Vorteil des Einsatzes von Recommender Systemen für Dienstleister ist ein höheres Verständnis dafür erlangen was Nutzer wollen und welche Artikel oft in Verbindung gekauft werden. (Ricci u. a. 2011)
Die Empfehlungen sind nicht auf zum Verkauf angebotene Produkte beschränkt, sondern können ebenso Filme, Musik, Geschäfte, Restaurants, Bars oder Ähnliches sein. Die Internetdienste von Amazon, Netflix, IMDB, YouTube, LastFM oder Spotify bieten ihren Kunden Empfehlungen von Recommender Systemen an und sind Beispiele für den Einsatz von solchen Systemen.
Vorschläge können hierbei auf verschiedener Datenbasis erzeugt werden. Man unterscheidet zwischen:

- *Content-based*: Artikel werden anhand ihres Inhalts verglichen, beispielsweise indem ähnliche Objekte vorgeschlagen werden
- *Collaborative filtering*: Artikel die Nutzer wählten, die Ähnliches gesucht haben
- *Knowledge-based*; Empfehlungen nach Domänenwissen zu gesuchten Artikeln

- *Demographic*: Vorschläge nach demographischen Informationen wie dem Herkunftsland
- *Community-based*: Vorschläge nach Artikeln die Freunde des Nutzers bevorzugen (beispielsweise durch eine Social-Media Integration)
- *Hybrid*: Eine beliebige Kombination aus den vorher Genannten

(Ricci u. a. 2011)

Während sich Recommender Systeme darauf beschränken Objekte zu finden, an die der Nutzer wahrscheinlich am meisten interessiert ist, sind Intelligent Personal Assistent Systeme, Programme, die Empfehlungen von Recommender Systemen verwenden und dem Nutzer präsentieren.

2.2 Intelligent Personal Assistent Systeme

Als Intelligent Personal Assistent System wird ein Programm bezeichnet, welches den Nutzer beim erfüllen verschiedener Aufgaben wie der Planung von Terminen, dem Versenden und Empfangen von Nachrichten, oder dem Beantworten einfacher (z.B. *"Wie ist das Wetter in Berlin?"*) oder komplexeren (z.B. *"Wie komme ich nach Hause?"*, *"Was kann ich heute Abend machen?"*) Fragen unterstützen soll. Apples Assistent Siri versteht zur Zeit (Stand August 2016) etwa 1600 solcher Aufgaben.[1] Die Eingaben erfolgen in gesprochender oder geschriebener Textform. Weitere Beispiele für solche Personal Assistent Systeme sind *Google Now*, das in Entwicklung befindende *Facebook M* oder der Dienst *Cortana* von Microsoft.
Eine starke Verbreitung haben diese Assistenzsysteme vor allem im mobilen Bereich. So bieten Apple, Google und Microsoft auf ihren mobilen Betriebssystemen jeweils eine Integration mit einem Intelligent Personal Assistent System an. Mit der wachsenden Akzeptanz solcher Anwendungen (Vascellaro 2012) werden die Assistenten zunehmend auch im Desktop Bereich angeboten. So wurde *Cortana* in Windows 10 integriert und *Google Now* in Googles Webbrowser *Chrome*.
Um Empfehlungen abzugeben werden an Intelligent Personal Assistent Systeme Suchmaschinen-ähnliche Dienste angebunden die Mechanismen nutzen wie auch Recommender Systeme. (Good u. a. 1999) Die Auswahl der Empfehlungen hat hierbei das Ziel einer besseren Nutzererfahrung, kann jedoch für dieselben Einsatzzwecke verwendet werden wie die von Recommender Systemen.

Während klassische Intelligent Personal Assistent Systeme computergenerierte Ergebnisse anzeigen, gehen die Dienste von *Magic*[2] oder *Operator*[3] einen

[1] Gesammelt durch die Website `http://hey-siri.io/`
[2] https://getmagicnow.com/
[3] https://operator.com/

4

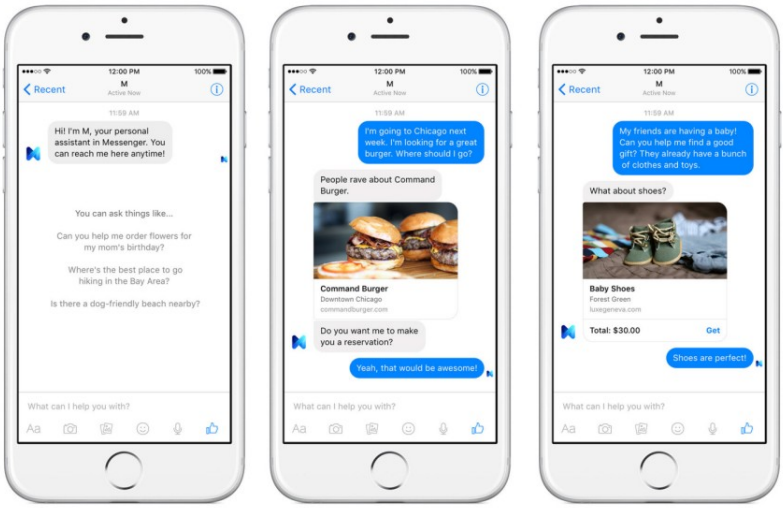

Abb. 0.1. Facebook M als Intelligent Personal Assistent in die Facebook Messenger App integriert ©Facebook

anderen Weg. Diese bieten zwar ähnliche Dienste an, die Anfragen werden jedoch durch Mitarbeiter beantwortet. Dadurch können Abfragen beantwortet werden, die voll-automatisch schwerer realisierbar sind. Da diese keine autonomen Systeme sind, werden sie in diesem Paper nicht näher untersucht.

Das im August 2015 angekündigte, in den Facebook Messegener integrierte, *Facebook M* geht einen hybriden Weg.[4] Die Anfragen werden teils von Mitarbeitern, teils automatisch von Chat-Bots beantwortet. (Abbildung 0.1) Der Nutzer erfährt in diesem Prozess nicht, ob die Antwort berechnet wurde, oder von einem Mitarbeiter beantwortet wurde. Die durch Facebook-Angestellten bearbeiteten Anfragen sollen verwendet werden um mit Machine-Learning Technologien zukünftig mehr und mehr Anfragen automatisch zu beantworten. Die von Facebook gespeicherten Daten zu den Nutzern sollen vorerst nicht für *Facebook M* verwendet werden.

2.3 Architektur

Eine Hauptaufgabe eines Persönlichen Assistenzsystems ist es, technische Interfaces soweit wie möglich zu verbergen, und alle Anfragen in natürlicher Sprache entgegenzunehmen. Deshalb ist eine *Natual Language Processing* Komponente ein zentraler Baustein der Software. Diese ermöglicht es, in text-

[4] http://www.wired.com/2015/08/facebook-launches-m-new-kind-virtual-assistant/

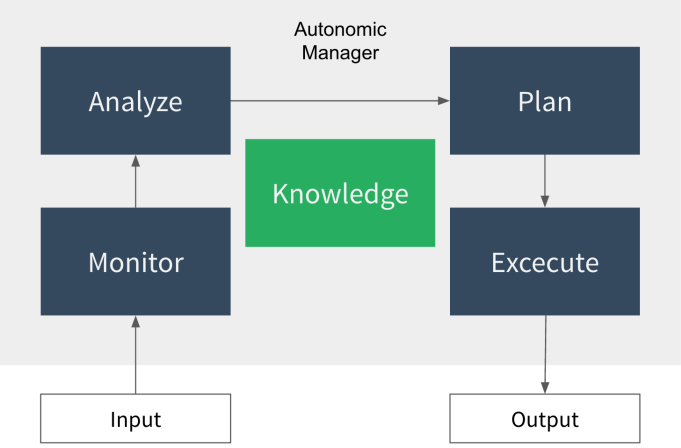

Abb. 0.2. Die klassische MAPE-K Architektur als Referenzarchitektur für Autonomic Computing Systeme

oder sprachform eingegebene Anfragen in eine maschinenleserliche Form umzuwandeln. (Rausch o.D.)
Die Wissensbasis des Assistenz Systems wird mithilfe sämtlicher Eingaben und Systemdaten (Dateisystem, E-Mails, Ortsdaten, etc.) aufgebaut und mithilfe von Reasoning-Technologien wie Machine-Learning erweitert und verfeinert. (Rausch o.D.)
Die nötigen Schritte zur Datenverarbeitung lassen sich grob auf die MAPE-K Architektur (Abbildung 0.6) (*"White Paper: An architectural blueprint for autonomic computing"* 2005) abbilden. Diese beschreibt einen iterativen Prozess in dem aus einem Eingabewert in den vier Schritten *Monitor*, *Analyze*, *Plan* und *Excecute* eine Ausgabe erzeugt wird. Diese Schritte werden auf Basis einer abgegrenzten Wissensdatenbank durchgeführt. Dieser Prozess wird von einem, von anderen abgegrenzten *Autonomic Manager* durchgeführt.

3 Verwandte Arbeiten

Auf die verschiedenen Probleme, die bei Big Data Anwendungen, zu denen auch Intelligent Personal Assistent Systeme gehören, wird in vielen Arbeiten eingegangen.
Jonas Rausch untersucht in *Influence of Ubiquitous Personal Assistants on the Society* (Rausch o.D.) ein ähnliches Thema wie in dieser. Es werden *Ubiquitous Personal Assistants* vorgestellt und verstärkt auf technische Aspekte wie Sicherheit, Automatisierung und Digitalisierung eingegangen. Es wird ebenso wie in diesem Paper ein Ausblick auf die Zukunft im Jahr 2025 gewagt und auf Probleme eingegangen.

Die einzelnen problematischen Aspekte wird in vielen Arbeiten beleuchtet. In *Data and Goliath* (Schneier 2015) wird insbesondere Datenschutz im Kontext von Big Data Applikationen eingegangen.
Das Problem welches auch bei Recommender Systemen auftritt, dass durch eine überpersonalisierung bestimmte Aspekte verloren gehen wurde unter dem Begriff *Filter Bubble* als Problem solcher Systeme wurde von Eli Pariser in *The filter bubble: What the Internet is hiding from you* (Pariser 2011) geprägt.

4 Zukünftige Entwicklung und deren Chancen und Risiken

In diesem Kapitel wird ein Ausblick gewagt wie die zukünftige Entwicklung aussehen könnte und welche Möglichkeiten, aber auch, welche Probleme sich daraus ergeben könnten.
Die Richtung in die sich Intelligent Personal Assistent Systemen entwickeln könnten, wird am Beispiel von *Facebook M* erkennbar. Die Komplexität der erfüllbaren Aufgaben steigt weiter und weiter. Durch den hybriden Ansatz können erstmals die Vorteile eines Dienstes, welcher auf menschlicher Arbeit beruht, mit komplett autonom arbeitenden kombiniert werden. Um sowohl die Antwortzeit, als auch die notwendige Arbeitszeit der *Facebook M* Mitarbeiter gering zu halten, wird das System bei komplexen Aufgaben angelernt. Dadurch sollen mehr und mehr komplexe Aufgaben automatisch möglich sein.

4.1 Vereinfachung komplexer Prozesse

Intelligente persönliche Assistenten können weiterhin einen starke Vereinfachung in Bezug auf komplexe Prozesse, wie beispielsweise eine Flugbuchung, hervorbringen. Es ist vorstellbar, dass ein persönlicher intelligenter Assistent durch den Befehl *"Buche für nächsten Freitag einen Flug von Frankfurt nach Berlin."* alle Aufgaben wie, die Suche nach einer günstigen Fluggesellschaft, der Auswahl der Zeit, der Buchung an sich und dem Check-In am Vortag vollautomatisch erledigt. Der Nutzer muss sich somit nicht mit den Prozessen der Fluggesellschaften auseinandersetzen und bekommt lediglich eine Buchungsbestätigung und das Flugticket. Einige Intelligent Personal Assistent-Systeme wie *Magic*[5] bieten einen Flugbuchungsvorgang schon heute in den USA an. Dieser Vorgang wird jedoch durch einen Mitarbeiter, und nicht automatisiert, durchgeführt. *Facebook M* soll den Vorgang einer Restaurantreservierung ebenfalls unterstützen.
Es ist eine Vielzahl solcher Szenarien ist denkbar, da immer mehr Dienstleister ihre Leistungen online zur Verfügung stellen.

[5] https://getmagicnow.com/

Abb. 0.3. Schon heute können Intelligente Persönliche Assistenten komplexe und kontextspezifische Aufgaben lösen. Beispiel: Apples *Siri*

4.2 Bessere Vorschläge auf Nutzerdatenbasis

Auch wenn *Facebook M* zurzeit keinen Zugriff auf die von Facebook gespeicherten persönlichen Daten hat, so haben diese ein gewaltiges Potential. Durch die Art und die Menge der Daten könnte die Bearbeitung von privaten Anfragen, wie der Auswahl eines Geschenks für eine bestimmte Person oder ähnlichem möglich werden. Da Soziale Netzwerke üblicherweise speichern, was einzelnen Personen gefällt, können somit konkrete Vorschläge gemacht werden. Ähnliche Prozesse finden schon jetzt bei der Auswahl von personalisierter Werbung statt.
Weiterhin zeigt sich, dass *Community-based* Vorschläge die von Freunden gemacht werden häufiger angenommen werden als andere. (Sinha und Swearingen 2001) Die kompletten Daten von Freunden in die Berechnungen von Empfehlungen einzubeziehen könnte also einen großen qualitativen Sprung bedeuten.

4.3 Datenschutzrechtliche Probleme

Um einen funktionierenden Dienst zu ermöglichen, ist es üblich, dass viele private Informationen der Nutzer auf Seite der Anbieter verarbeitet und auch gespeichert werden. Diesem Vorgang stimmt man üblicherweise vor der Nutzung in Form einer Endbenutzer-Lizenzvereinbarung zu. In diesen stimmt man teils obskuren Bestimmungen, wie der Weitergabe privater Daten an Dritte[6] oder der Änderung der Lizenz ohne dass die Nutzer informiert werden zu.[7]

[6] *Facebook Data Policy* 2015.
[7] *Apple iOS Software License Agreement* o.D.

Die überwiegende Mehrheit der Benutzer setzen sich nicht mit dem Inhalt der Lizenzvereinbarung auseinander. So hat eine Studie in "*Trained to Accept?: A Field Experiment on Consent Dialogs*" (Böhme und Köpsell 2010) gezeigt, dass ein Dialog umso schneller übersprungen wird, je ähnlicher er einer Lizenzvereinbarung sieht.
Es ist somit anzunehmen, dass der Mehrheit der Nutzer ist somit nicht bewusst ist, was mit ihren persönlichen Daten geschieht. Ob, und in welcher Art und Weise die Daten bei der Weitergabe an Dritt-Unternehmen anonymisiert werden, ist nur schwer nachzuvollziehen.
Eine echte Anonymisierung nur kaum zu ermöglichen, da Nutzer ab einer gewissen Datenmenge eindeutig zuordenbar sind. (Schneier 2015) So kann eine Person ermittelt werden, indem man die 100 Lieblingsfilme dieser vergleicht. (Schneier 2015) In *Data and Goliath* schreibt der Autor Bruce Schneier, 95% der US-Amerikaner könnten anhand von nur vier Orten, die sie zu einer bestimmten Zeit besucht hatten, identifiziert werden. Eine echte Anonymisierung der Daten scheint dadurch nur schwer möglich zu sein.

4.4 Eingriff in die Privatsphäre der Nutzer

Ein autonomes Intelligent Personal Assistent System könnte ähnlich wie *Facebook* oder *Target* Annahmen über das Privatleben der Nutzer machen und so beispielsweise auf die Ethnie oder Sexuelle Präferenzen schließen. (Duhigg 2012) Dies kann insbesondere dann problematisch sein, wenn im Land des Nutzers diese unter Strafen gestellt werden. Weiterhin ist es in Zukunft denkbar, dass ein intelligenter beispielsweise eine Schwangerschaft vor dem Nutzer anhand dessen Verhalten erkennen könnte.

4.5 Betriebswirtschaftlich orientierte Vorschläge

Mit der steigenden Nutzung Intelligenter Assistenzsysteme, wird für Dienstleister wichtiger, in der Bewertung der Recommender Systeme von intelligenten persönlichen Assistenzsysteme hoch eingeordnet zu werden. Es ist zu erwarten, dass wie bei der Google Suche, Twitter oder Facebook bereits üblich, gesponserte Ergebnisse, wie normale dargestellt werden. Eine Studie aus dem Jahr 2015 zeigte, dass insbesondere Jugendliche Werbung und normale Suchergebnisse der Suchmaschine Google nicht unterscheiden. So konnten nur 31% der 12 bis 15-jährigen erkennen welche Suchergebnisse Werbung sind. (Ofcom 2015) Daher ist es besonders wichtig, dass jeder Nutzer eine klare Trennung zwischen finanzierten, und berechneten Beiträgen erhält, da die Präsentation der Daten in IPA-Systemen üblicherweise nur einen oder wenige Resultate darstellt.
Vielen Benutzern wollen nicht, dass Ihre Daten für Werbezwecke verwendet werden. So würden beispielsweise nur 9% der Befragten einer Umfrage von McDonald und Cranor (2010) personalisierte Werbung akzeptieren in einem

E-Mail Client, wenn er dafür kostenfrei wäre. Dies zeigt, dass Nutzern ihre privaten Daten als schützenswert erachten. Vielen ist jedoch nicht bewusst, dass sich Software durch genau diese Weitergabe der Daten finanziert. (McDonald und Cranor 2010)

4.6 Reinforced Feedback Loops und Filter Bubbles

Als *Feedback Loop* wird der, aus der Elektrotechnik bekannte Effekt bezeichnet, der auftritt wenn ein Signal auf sich selbst einwirkt. Dabei können verschiedene Effekte auftreten. Die Rückkopplung kann verstärkend oder dämpfend auf das Ursprungssignal wirken. Bei einem verstärkenden Feedback, spricht man von *Reinforced Feedback Loops*.
Ein ähnlicher Effekt kann auch bei intelligenten Filterungsalgorithmen auftreten. Sucht ein Benutzer wiederholt nach ähnlichen Datensätzen, so lernt der Algorithmus mehr und mehr das Nutzerverhalten an und bietet zunehmend einseitigere Ergebnisse an.(Abbildung 0.4)
Beispielsweise könnte ein Nutzer einmalig nach einem Buch aus dem Bereich der Verschwörungstheorien interessieren. Da der Nutzer nur nach diesem Buch gesucht hat nimmt das System nun an, dass der Nutzer großes Interesse an diesem Bereich hat. Dadurch werden ihm weitere entsprechende Bücher angeboten, welche möglicherweise wieder angesehen werden. So verstärkt sich der Effekt mehr und mehr selbst. Diese Überpersonalisierung der Ergebnisse wird auch als *Filter Bubble* bezeichnet.

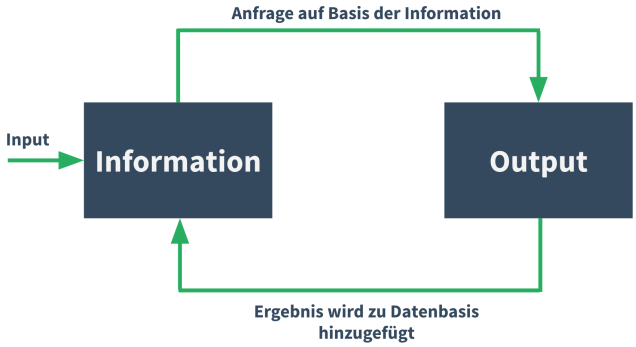

Abb. 0.4. Beispielhafter Ablauf eines Reinforced Feedback Loops

4.7 Fehlerhafte Informationsquellen

Ein weiteres großes Problem in Virtual Assistent Systemen können die eigentlichen Datenquellen sein. Selbst ein intelligentes System kann nur schwer entscheiden, ob Informationen vertrauenswürdig sind, oder nicht. Da die Daten

automatisch erfasst werden, erfolgt keine explizite Prüfung, ob gespeicherte Informationen korrekt sind, oder nicht. So ist es in der Vergangenheit bereits vorgekommen, dass Intelligente Assistenten falsche Informationen wiedergegeben haben. Appels *Siri* verwendet beispielsweise die Wikipedia-Enzyklopädie als Informationsquelle. Im September 2015 änderte ein Wikipedia Nutzer den Artikel zu Angela Merkel in einen beleidigenden Text. Obwohl die Änderung schnell rückgängig gemacht wurden, blieb der verfälschte Artikel in der *Siri*-Datenbank erhalten.(*Sprachassistentin Siri macht aus Merkel ein Ferkel* 2015) (Abbildung 0.5) Das Beispiel zeigt, wie einfach es ist, falsche Informationen in ein IPA-System einzuschleusen. Dies kann insbesondere bei Manipulationen ein Problem sein, die nicht offensichtlich als falsch erkannt werden können.

Abb. 0.5. Falsche Informationen die durch eine Manipulation eines Wikipedia-Artikels im September 2015 von Apples *Siri* über längere Zeit weitergegeben wurden, obwohl der Artikel in der Wikipedia korrigiert wurde.

5 Lösungsansätze

Der *datenschutzrechtliche Aspekt* von Intelligent Personal Assistent Systemen trifft selbstverständlich nicht ausschließlich auf diese Art Software zu. Ein Kernproblem ist hierbei, dass der Mehrheit der Nutzer das technische Verständnis dafür fehlt, was mit den eigenen Daten passiert. (McDonald und Cranor 2010) Weiterhin ist vielen nicht klar, dass eine Anonymisierung nur schwer möglich ist, und es somit kaum ein "Verschwinden in der Menge"gibt. (Schneier 2015) Hier ist eine deutlich höhere Transparenz nötig, aufbereitet in einer Form, die für einen interessierten aber technisch nicht versierten Endnutzer verständlich ist.
In Bezug auf datenschutzrechtliche Probleme ist ebenso eine transparente Präsentation der Lizenzvereinbarung und den Datenschutzbestimmungen. Ein

gutes Beispiel hierfür ist die Vereinbarung des Webservice 500px,[8] in der zu jedem Absatz in einfachen Sätzen erklärt ist, was dieser bedeutet.
Hinsichtlich *finanzierter Vorschläge* ist insbesondere eine klare Kennzeichnung nötig. Einem Nutzer muss sofort bewusst sein, dass er eine personalisierte Werbung sieht, und nicht denkt, es würde sich um ein normales Ergebnis handeln.
Das Problem von *fehlerhaften Informationsquellen* lässt sich nicht automatisiert lösen. Intelligente Assistenten sind auf ständig aktuelle Quellen wie beispielsweise Wikipedia oder Nachrichtenseiten angewiesen. Da die Daten (semi)-automatisch erhoben werden, ist eine Kontrolle der Quellen nur schwer möglich. Dennoch sollte beachtet werden, dass ausschließlich neutrale und vertrauenswürdige Quellen angebunden werden. Weiterhin wenn Benutzer fehlerhafte Informationen melden können, sodass der Betreiber der IPA-Software möglichst schnell reagieren kann, sobald ein Fehler auftaucht. Zurzeit haben Nutzer keine Möglichkeit Fehlerhafte Informationen zu melden.

Abb. 0.6. 1: Klassische Filter-Bubble. Nur die Kernelemente werden gefunden. 2: Filter Bubble mit allgemeineren Filtern. Es kommen unrelevante Ergebnisse hinzu, relevante Resultate außerhalb der Filter Bubble werden jedoch nicht gefunden. 3: Idealfall bei dem alle relevanten Ergebnisse gefunden wurden, jedoch kein irrelevantes

Reinforced Feedback Loops sind im Bereich der Recommender Systeme ein bekanntes Problem. (Nguyen u. a. 2014; Maccatrozzo 2012; Abbassi u. a. 2009) Es gibt eine Reihe von Ansätzen wie das Problem gelöst werden kann und Serendipitäten, also passende zufällige Vorschläge, ermöglicht werden können. Ein einfaches "verallgemeinern" von Filtern löst das Problem der *Filter Bubbles* nicht, sondern vergrößert nur die gefundene Blase. (Nguyen u. a. 2014) Es existieren jedoch diverse Lösungsansätze für dieses Problem. Beispielsweise versucht das Outside-The-Box Modell von Abbassi u. a. (2009) ein relevantes Ergebnis zu finden, jedoch gleichzeitig ein geringes Risiko eingegangen wird, dass das Ergebnis nicht relevant für den Nutzer ist. Ein anderer Ansatz für dasselbe Problem wird von Maccatrozzo (2012) vorgestellt. Hier wird versucht Serendipitäten mithilfe einer semantischen Modells erzeugt.

[8] https://about.500px.com/terms/

12

6 Fazit

Die Nutzungsmöglichkeiten von Intelligenten Persönlichen Assistenzsystemen und generellen (Big-)Dataanalysen sind breit gefächert und bieten ein gigantisches Potential an Möglichkeiten. Anwendungsfälle wie in Kapitel 4 beschrieben sind größtenteils schon heute technisch erfüllbar, jedoch noch nicht umgesetzt. Die Einbeziehnung von Social-Media Daten bietet eine weitere große Datengrundlage, die für die Verbesserung des Assistenzdienstes hinzugezogen werden kann.
Die aufgeführten Probleme sind bezeichnend für eine Vielzahl an Softwaresystemen, und müssen adressiert werden. Was die Preisgabe von persönlichen Informationen angeht, so gehen hier auch die Expertenmeinungen weit auseinander. (Schneier 2015) Es muss ein Weg gefunden werden, die gegebenen Möglichkeiten effektiv zu nutzen, gleichzeitig aber niemanden unfreiwillig in seiner Privatsphäre einzuschränken. Es ist ein hohes Maß an Transparenz der Betreiber von IPA-Software erforderlich, damit jeder Nutzer einen Einblick hat welche Daten preisgegeben werden und wie die Informationen ermittelt werden. Sollten Einträge finanziert worden sein, so ist eine gut erkennbare Markierung dieser erforderlich.
Die genannten Risiken sollten als abschreckend aufgefasst werden. Intelligente Persönliche Assistenten bieten ihren Nutzern viele nützliche Dienste, die sonst zeitlich Aufwändig sind oder komplex zu lösen sind.

7 Literatur

Abbassi, Z., S. Amer-Yahia, L. V. Lakshmanan, S. Vassilvitskii und C. Yu (2009). "Getting recommender systems to think outside the box". In: Proceedings of the third ACM conference on Recommender systems. ACM, S. 285–288.

Apple iOS Software License Agreement. Apple Inc. URL: http://images.apple.com/legal/sla/docs/iOS8.pdf.

Böhme, R. und S. Köpsell (2010). "Trained to Accept?: A Field Experiment on Consent Dialogs". In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. CHI '10. Atlanta, Georgia, USA: ACM, S. 2403–2406. URL: http://doi.acm.org/10.1145/1753326.1753689.

Duhigg, C. (2012). "How Companies Learn Your Secrets". In: New York Times. URL: http://www.nytimes.com/2012/02/19/magazine/shopping-habits.html?_r=0.

Facebook Data Policy (2015). Facebook Inc. URL: https://www.facebook.com/full_data_use_policy#ip.

Good, N., J. B. Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker und J. Riedl (1999). "Combining collaborative filtering with personal agents for better recommendations". In: AAAI/IAAI, S. 439–446.

Maccatrozzo, V. (2012). "Burst the filter bubble: using semantic web to enable serendipity". In: International Semantic Web Conference. Springer, S. 391–398.

McDonald, A. M. und L. F. Cranor (2010). "Beliefs and Behaviors: Internet Users' Understanding of Behavioral Advertising". In:

Nguyen, T. T., P.-M. Hui, F. M. Harper, L. Terveen und J. A. Konstan (2014). "Exploring the filter bubble: the effect of using recommender systems on content diversity". In: Proceedings of the 23rd international conference on World wide web. ACM, S. 677–686.

Ofcom (2015). "Children and Parents: Media Use and Attitudes Report 2015". In: Ofcom.

Pariser, E. (2011). *The filter bubble: What the Internet is hiding from you*. Penguin UK.

Rausch, J. "Functionality, Threats and Influence of Ubiquitous Personal Assistants with Regard to the Society". In: TECHNISCHE BERICHTE, S. 47.

Ricci, F., L. Rokach und B. Shapira (2011). *Introduction to recommender systems handbook*. Springer.

Schneier, B. (2015). *Data and Goliath: The Hidden Battles to Capture Your Data and Control Your World*. 1st. W. W. Norton & Company.

Sinha, R. R. und K. Swearingen (2001). "Comparing Recommendations Made by Online Systems and Friends." In: DELOS workshop: personalisation and recommender systems in digital libraries. Bd. 1.

Sprachassistentin Siri macht aus Merkel ein Ferkel (2015). URL: http://www.spiegel.de/netzwelt/gadgets/siri-beleidigt-angela-merkel-gefaelschter-wikipedia-eintrag-a-1054790.html.

14

Vascellaro, J. E. (2012). *The Wall Street Journal: Apple's Siri Gains Traction–For Some Things*. URL: http://blogs.wsj.com/digits/2012/03/26/apple%C3%A2%C2%80%C2%99s-siri-gains-traction-for-some-things/.

"White Paper: An architectural blueprint for autonomic computing" (2005).

Evolution einer Microservice Architektur zu Autonomic Computing

1 Einleitung

Microservices werden in vielen verteilten System wie Netflix und Airbnb eingesetzt. Sie ermöglichen eine bessere Wartbarkeit und Skalierbarkeit durch Aufteilung eines Systems in unabhängige, miteinander kommunizierende Dienste. Durch die wachsende Anzahl an zu verwaltenden Prozessen wächst jedoch die Komplexität in der Infrastruktur, insbesondere in der Einhaltung von operativen Eigenschaften wie Availability, Realibility, Durability, oder Performance. Bei der horizontalen und vertikalen Skalierung eines Dienstes wirken sich vielen Faktoren wie z.B. Replikationshäufigkeit, Standort (Rack, Region, Datencenter) oder Lokalität korrelativ auf die o.g. Eigenschaften aus. Diese Wechselbeziehungen sind stark domänen- und dienstspezifisch. Jede Änderung an einem Service kann eine Auswirkung haben. All diese Faktoren im Auge zu behalten wird mit wachsender Anzahl der Microservices unmöglich. Eine Lösung könnte Autonomic Computing darstellen. Hier werden die Dienste selbst von Software autonom und intelligent verwaltet.

Das Paper analysiert die aktuell genutzten Methoden zur Einhaltung operativer Eigenschaften bei der Verwaltung von Microservicesystemen. Dabei wird die Architektur des Dienstes Netflix als State of the Art eingeführt und die einzelnen Komponenten in den MAPE-K Feedbackloop des Autonomic Computing Ansatzes eingeordnet. Durch die Kategorisierung nach dem Adoption Model werden Verbesserungen , um eine höhere Kategorie zu erreichen. Am Schluss steht ein Ausblick auf Selbstadaptive Software am Beispiel des Forschungsprojektes MQuAT, welches die Definition und Einhaltung von SLAs ermöglicht (Götz 2013) und ein Fazit.

2 Grundlagen

2.1 Microservice

Anwendungen werden immer wieder an neue Geschäftsanforderungen angepasst. Neue Funktionen werden hinzugefügt, alte Funktionalitäten werden refaktorisiert und verändert. Die Codebasis einer monolithischen Anwendung wächst und wird immer komplexer. Trotz einer anfänglich guten, zukunftssicheren Architektur kann es bei diesem Änderungsprozess zum Durchbrechen der Kohärenz und somit zur Verletzung des Single-Responsibility-Prinzips kommen. Dabei werden Abhängigkeiten zwischen eigentlich getrennten Modulen eingeführt. Daraus resultiert vor allem eine verminderte Wartbarkeit, was die Umsetzung von neuen Anforderungen und die Beseitigung von Fehlern erschwert.

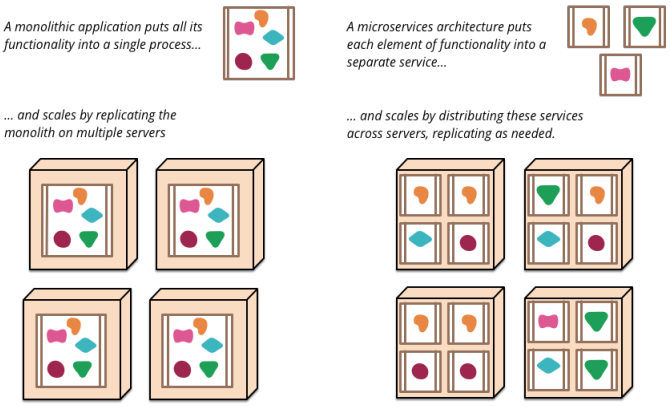

Abb. 1. Monolithen vs. Microservices

Hier setzt der Microservice Architekturstil an, wobei Software aus vielen kleinen, autonomen Diensten aufgebaut wird, wodurch ein emergentes System entsteht. Diese Dienste sind unabhängige Einheiten, welche oft als Betriebssystemprozesse oder Container auf Maschinen verteilt werden. Kommunikation erfolgt ausschließlich über Netzwerkprotokolle und APIs, z.B. HTTP / REST. Der Microservice-Ansatz ist somit eine konkrete Umsetzung einer Serviceorientierten Architektur (Newman 2015).

Wie in Abbildung 1 (rechts oben) veranschaulicht, implementiert jeder Dienst nur eine Geschäftsfähigkeit. Die Abgrenzungen des Service entsprechen somit den Grenzen der Geschäftstätigkeit. Durch die physische Trennung ist das Aufweichen der Modulgrenzen nicht mehr möglich und sorgt somit für eine verbesserte und nachhaltige Kohärenz. Eine Veränderung in einem Microservice sollte keine Änderung in einem anderen notwendig machen. Damit kann die Entwicklung besser an verschiedene Teams verteilt werden und die sonst notwendige Kommunikation und Koordination zwischen diesen auf ein Minimum reduziert. Auch Merge-Konflikte werden seltener, was zu Erhöhung der Produktivität und der Zufriedenheit der Entwickler führt.

Ein weiterer Vorteil ist die **Technologieheterogenität**. Jeder Service kann mit einer zum Problem passenden Technologie implementiert werden. Das führt zum einen zu einer erhöhten Agilität, zum anderen zu einer Verlangsamung des Alterungsprozesses. Sollten neuere, besser geeignete Technologien in Zukunft vorhanden sein, können diese in neuen Diensten eingesetzt werden. Auf Grund der kleinen Größe können zu einem geeigneten Zeitpunkt auch die „Legacy“ Dienste nachgezogen werden. Somit wird auch einem Mangel an Fachkräften vorgebeugt, der z.B. bei alten, mit COBOL umgesetzten Banksystemen eingetreten ist. Natürlich kann eine große Vielfalt an Technologien zu Problemen führen und die Komplexität erhöhen. Deshalb limitieren Firmen wie Netflix häufig die Technologien, z.B. auf JVM basierte Sprachen und Werkzeuge (Newman 2015).

Die **Skalierung** einer monolithischen Anwendung erfolgt immer als Ganzes (vgl. Abb. 1). Soll die Verfügbarkeit und Ausfallsicherheit erhöht werden, wird die ganze Anwendung auf mehreren Servern repliziert und es findet eine Lastverteilung statt. Fällt ein Teil des Monolithen aus, so fällt der ganze Monolith aus. Microservice können unabhängig von einander und feingranular skaliert werden (vgl. Abb. 1). Dienste mit höherer Last können repliziert werden, Dienste mit niedriger Last können unverändert bleiben. Fällt ein einzelner Service aus, führt dies zu einer Verminderung der Funktionalität, jedoch nicht zum Gesamtausfall. Somit hat das System eine höhere Resilienz.

2.2 Autonomic Computing

Durch die Aufteilung einer Anwendung in viele einzelne Prozesse wird die Komplexität dieser auf die Infrastruktur verlagert. Es entsteht ein hochverteiltes System mit erheblichen Kommunikationsmehraufwand. Bei steigender Anzahl von Microservices, zum Vergleich: der Onlinefashionshop der Firma Gilt besteht beispielsweise aus mehr als 450 Diensten (Newman 2015), ist die Verwaltung und Skalierung durch Menschen mit hohen Kosten und Aufwand verbunden.

> Autonomic Computing helps to address complexity by using technology to manage technology [...] with minimal human intervention (Computing o.D.)

Um dieses Ziel zu erreichen, muss das System sich automatisiert und autonom konfigurieren, sowohl initial als auch bei sich wechselnder Umgebung. Eine manuelle Konfiguration ist bei steigender Komplexität und Anzahl der Dienste nicht mehr möglich. Vor allem da Änderungen in der Umgebung innerhalb weniger Minuten oder sogar Sekunden auftreten können (Horn 2001). Diese **self-configuration** Eigenschaft umfasst zum Beispiel das Hinzufügen oder Entfernen von neuen Diensten oder das Starten neuer Cloudinstanzen (Computing o.D.).

Im Falle eines Systemausfalls oder anderen operativen Problemen handelt das System selbstheilend (**self-healing**). Nach dem Feststellen eines Fehlers findet die autonome Diagnose statt. Es werden automatisierte Korrekturmaßnahmen eingeleitet, ohne dabei den Dienst als Ganzes zu unterbrechen. Ein Beispiel wäre das Neustarten eines fehlerhaften Prozesses nach dem Fail-Fast-Prinzip (Shore 2004).

Außerdem sollen sich autonome Systeme ständig selbst optimieren und anpassen (**self-optimizing**). Durch kontinuirliche Überwachung von Schlüsselmetriken wie CPU, RAM, Antowortzeiten oder auch Energieverbrauch passt das System seine Implementierung an und misst erneut um eine Verbesserung oder Verschlechterung festzustellen. Nur so kann das System den wachsenden Anforderungen an zukünftige Anwendungen gerecht werden (Horn 2001).

Sollte das System durch einen Angriff bedroht sein, etwa durch einen Virus oder eine DDoS Attacke, erkennt das System diese und leitet automatisiert Maßnahmen an. Im Besten Falle werden Angriffe im Vorraus erkannt und die Schwachstellen geschlossen. Diese **self-protecting** Eigenschaft liegt jedoch nicht im Fokus der Arbeit.

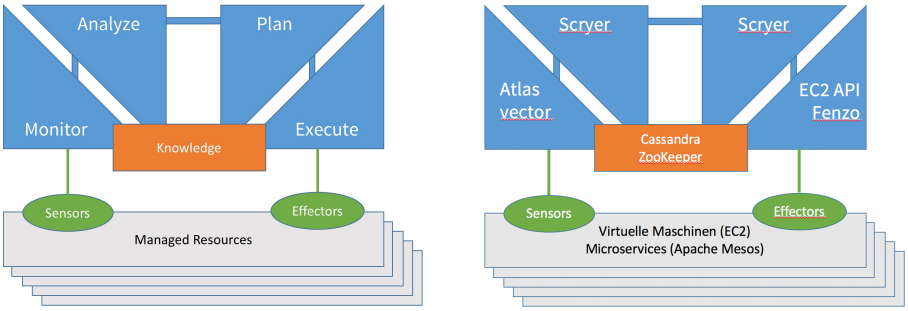

Abb. 2. links: MAPE-K Referenzarchitektur, rechts: Einordnung Netflix Architektur

3 State of the Art Architektur

Auf Grund der Neuheit des Microservice Ansatzes hat sich noch keine Standard Architektur durchgesetzt, so dass Firmen wie Netflix, Airbnb oder Otto.de eigene Lösungen schaffen und diese sehr offen online dokumentieren. Dieser Abschnitt zeigt eine Abbildung der Lösungen auf die MAPE-K Referenzarchitektur für Autonomic Computing am Beispiel der Netflix Architektur. Netflix, ein weltweiter Videostreaming Dienst, stellt mit dem Netflix Open Source Software Center[1] eine hohe Anzahl intern entwickelter Werkzeuge quelloffen zur Verfügung. Mit tausenden Microservices und zehntausenen VMs betreibt die Firma eine hochkomplexe und hochverteilte Infrstruktur. Diese ist verantwortlich für ein Drittel des Downloadtraffic in den USA und liefert pro Jahr weltweit 42,5 Milliarden Stunden (5 Millionen Jahre) an Videomaterial aus[2]. Diese Komplexität und Last ist für Menschen nur schwer verwaltbar und würde somit enorm von Autonomic Computing profitieren.

3.1 Einordnung in MAPE-K

Um Autonomic Computing zu ermöglichen, ist eine kontinuirliche Überwachung der Dienste, eine Erkennung von Situationen und Ereignissen, sowie die Planung und Ausführung von geeigneten Aktionen notwendig. Hierfür schlägt IBM in (Computing o.D.) die **MAPE-K Architektur** vor, welche aus den in Abb. 2 dargestellten Komponenten besteht.

Die unterste Ebene sind die verwalteten Elemente (**Managed Resources**). Sie stellen über die Sensoren Daten bereit und werden am Ende des Kreislaufes durch die Effektoren verändert. Bei Netflix entsprechen diese den verwalteten Microservices. Diese stellen Lognachrichten, Zugriffszeiten oder andere servicespezifische Metriken über (REST) Schnittstellen bereit. Aber auch die ausführenden virtuellen Maschinen stellen Daten wie Auslastung, belegten Arbeitsspeicher,

[1] https://netflix.github.io/
[2] http://de.slideshare.net/aspyker/netflix-and-containers-titus

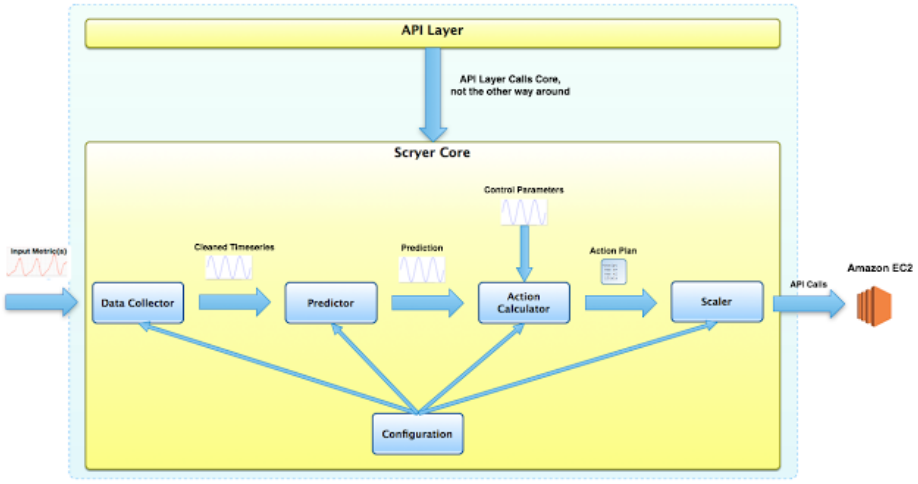

Abb. 3. Scryer Architektur

Datendurchsatz oder die momentanen Konfigurationsparameter über Amazons CloudWatch oder Apache Mesos zur Verfügung.

Diese Sensoren können von der **Monitor** Funktion überwacht werden. Die großen Datenmengen werden aggregiert und auf Korrelationen untersucht bis ein so genanntes Symptom gefunden wurde. Neflix sammelt Milliared von Metrikdaten pro Minute und mehrere TB pro Tag[3]. "It's been said that the Netflix microservices architecture is a metrics generator that occasionally streams movies." — dieses Zitat unterstreicht die Notwendigkeit eines Werkzeuges wie vector[4] oder Atlas[5] für die Überwachung und Auswertung der Daten. Dabei wird vector zum großen Teil von menschlichen Administratoren genutzt, um hochaufgelöste Metriken im Fehlerfall einer bestimmten Maschine oder eines Dienstes zu analysieren. Atlas hingegen ist ein System zum Speichern und Analysieren von großen, mehrdimensionalen Metrikdaten in nahezu Echtzeit. Es erlaubt komplexe Anfragen gegen Datenserien auszuführen. Es bietet sowohl eine Weboberfläche zur Darstellung und Auswertung der Daten, als auch eine REST Schnittstelle für die Anbindung weiterer Anwendungen. Durch die Stack Language[6] können spezifische voranalysierte und gefilterte Datenausschnitte mit einer URL referenziert werden. In Verbindung mit der REST Schnittstelle könnten so Symptome nach Außen zur Verfügung gestellt werden.

Die **Analyse** Funktion wertet Symptome aus und entscheidet, ob Maßnahmen ergriffen werden müssen, z.B. bei einer Verletzung eines Service Level Agree-

[3] http://techblog.netflix.com/2014/01/improving-netflixs-operational.html und http://techblog.netflix.com/2014/12/introducing-atlas-netflixs-primary.html

[4] https://github.com/Netflix/vector

[5] https://github.com/Netflix/atlas

[6] https://github.com/Netflix/atlas/wiki/Stack%20Language

ments. Dabei werden auch zukünftige Zeitreihen unter Bezugnahme vorheriger Situation und Domänenwissen (Knowledge) vorhergesagt. In det Netflix Architektur wird dieser Job vermutlich von Scryer[7] umgesetzt. Dieses Tool ist nicht als Open-Source veröffentlicht, jedoch wird die Architektur in einem Blog Post[8] mit Abbildung 3 dargestellt. Der Datenfluss von *Data Collector* ⇒ *Predictor* ⇒ *Action Calculator* ⇒ *Scaler* unter Verwendung der *Configuration* ist analog zu MAPE-K. An die *Data Collector* Komponente kann eine Quelle wie Atlas angebunden werden. Der *Action Calculator* enspricht der **Plan** Funktion. Diese generiert Schritte, um die nötigen Änderungen zu erreichen und erstellt daraus eine Ausführungsplan. Dieser kann aus einem einzelnen Kommando oder einem komplexen Workflow bestehen. Der generierte Ausführungsplan wird an den Execute Schritt übergeben. Dieser kann den Plan einplanen und umsetzen. Dabei könnten Resourcen gestartet oder umkonfiguriert werden. Dies entspricht der *Scaler* Komponente, welche an verschiedene API angebunden kann, wie zum Beispiel die EC2 API oder Netflix Fenzo[9] Scheduler in Verbindung mit Apache Mesos. Mesos ist eine Plattform zur Abstraktion und gesammelter Bereitstellung von Ressourcen eines Clusters (Hindman u. a. 2011). Es ermöglicht die verteilte Ausführung und Skalierung von Diensten und entspricht der Rolle des Effectors. Dabei senden Mesos Nodes Ressourcenangebote - auf welchen Knoten sind wieviele Ressourcen verfügbar - an registrierte Frameworks, wie zum Beispiel Fenzo. Dieses Framework verfügt über verschiedene Schdulingalgorithmen, um die benötigten Dienste zu verteilen. Dabei können Eigenschaften wie Lokalität oder Hardwarevorraussetzungen beachtet werden. Mit Bin Packing Algorhitmen ist es möglich, eine Ressource wie CPU oder RAM maximal auszunutzen. Während der Ausführung des Plans wird auch die Knowledge über das aufgetretene Symptom und die ausgeführten Schritte aktualisiert, so dass diese bei zukünftigen Entscheidungen mit berücksichtigt werden können. Diese **Knowledge** Komponente ist die zentrale Wissens- und Datenbasis, analog zur *Configuration*. Sie enthält durch Überwachung erhaltene Daten, Symptome und Änderungspläne. Darüber hinaus ist domänenspezifisches Wissen und bestimmte Regeln hinterlegt, z.B. als Model oder Zustandsmaschine. Zur Persistierung des Wissens kommt bei Netflix unter anderem Zookeeper oder Cassandra zum Einsatz.

Viele Komponenten und Werkzeuge aus der Netflix Architektur lassen sich in MAPE-K einordnen, obwohl diese keine direkte Implementierung der Referenzarchitektur ist. Der interne Aufbau von Scryer enthält einen ähnlichen, sich wiederholenden Kontrollkreislauf (Feedback Lopp), welcher unentbehrlich für Autonomic Computing ist. Vollständigkeitshalber seien hier andere Vorschläge zur Umsetzung erwähnt, wie der CADA loop (Dobson u. a. 2006) und der Kreislauf von (Oreizy u. a. 1999). Diese unterscheiden sich nur in Details, deshalb beziehen wir uns in dieser Arbeit auf die MAPE-K Referenzarchitektur.

[7] http://techblog.netflix.com/2013/11/scryer-netflixs-predictive-auto-scaling.html
[8] http://techblog.netflix.com/2013/12/scryer-netflixs-predictive-auto-scaling.html
[9] https://github.com/Netflix/Fenzo

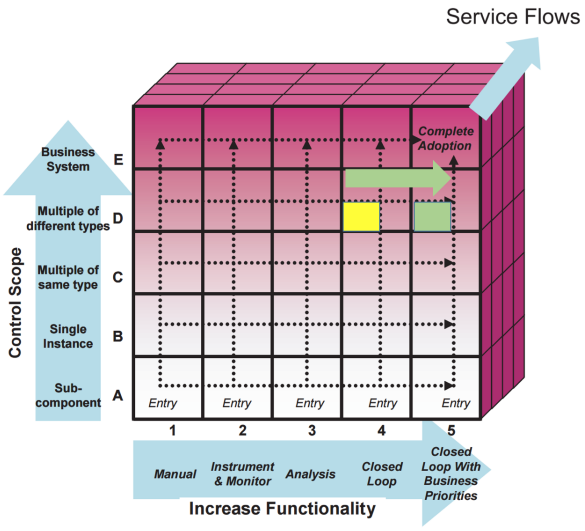

Abb. 4. Autonomic Computing Adoption Model

3.2 Einordnung in Autonomic Computing Adoption Model

Um den Grad der Autonomität der Netflix Infrastruktur zu beurteilen, kann man diese in das **Autonomic Computing Adoption Model** von IBM (Computing o.D.) einordnen. Dieses besteht aus drei Achsen, der horizontalen **functionality** Achse, der vertikalen **control scope** Achse und der **service flow** Achse (vgl. Abb.4). Die service flow Achse ist nicht im Fokus dieser Arbeit.

Die **functionality** Achse definiert den Grad der IT Automatisierung und ist in fünf Auspregungen eingeteilt, wobei in der Grundstufe **Manual** alle Managementaktivitäten von menschlichen Administratoren durchgeführt werden.

Die nächste Stufe **Instrument & Monitor** führt Werkzeuge ein, welche die Überwachung der verwalteten Resourcen und Sammlung von Messdaten erlauben. Diese Daten kann der Administrator verwenden, um das System zu verwalten. Netflix erfüllt diese Kategorie durch den Einsatz von Werkzeugen wie *Atlas* und *vector*.

Als Erweiterung führt die **Analysis** Stufe zusätzliche Analysewerkzeuge ein, welche Korrelation zwischen Messdaten aller verwalteten Resourcen feststellen können. Es können Muster erkannt werden und Handlungsvorschläge für den Administrator erstellt werden. Netflix erfüllt diese Stufe durch den Einsatz der Werkzeuge *Atlas* und *Scryer*. Beide Werkzeuge stellen eine Weboberfläche bereit, welche die Schnittstelle für den menschlichen Administrator bilden.

Ein **Closed Loop** erlaubt es einem automatisierten System anstatt einen menschlichen Administrators Aktionen auf Basis der Messdaten und Korrelationen vorzunehmen. Auch diese Stufe erfüllt Netflix mit den *Action Calculator* und *Scaler* Komponenten von *Scryer* und den Einsatz von Apache Mesos / Fenzo.

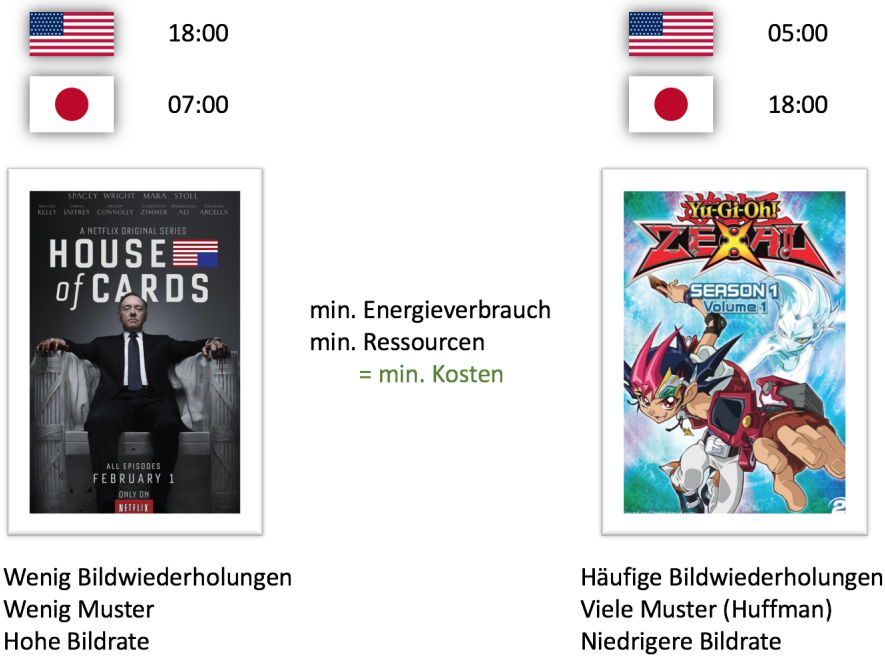

Abb. 5. Konstruiertes Beispiel für die verschiedenen Sehgwohnheiten von Netflix Benutzern

Der **Closed Loop With Business Priorities** erlaubt zusätzlich die Definition von domänenspezifischen Geschäftsregeln und Service Level Agreements. Der IT Administrator überwacht nun Business Prozesse und kann zum Ziele und Regeln anpassen. Zum Zeitpunkt dieser Arbeit ist nicht bekannt, dass Netflix die Regeln auf einer abstrakten Domänenspezifischen Ebene des Videostreamings spezifiziert. Somit erfüllt es diese Stufe der Autonomität nicht.

Die y-Achse des Adoption Models stellt den **Control Scope** dar. Die functionality - Achse stellt die Menge und den Grad der automatisierten Funktionalitäten dar. Der Control Scope hingegen die Art und Anzahl der automatisierten Ressourcen. In der niedrigsten Stufe wird nur eine Unterkomponente automatisiert und in der höchsten Stufe alle vorhanden Ressourcen samt den Business Prozessen in der Firma selbst. Netflix befindet sich hier auf der Stufe *Multiple of different types* und automatisiert verschiedene Ressourcen wie VMs, Container und Anwendungen. Zur Autonomität von internen Business Prozessen gibt es zum Zeitpunkt der Arbeit keine Quellen.

Dabei ergibt sich die in Abbildung 4 dargestellte Einordnung der Netflix Architektur.

4 Multi-Quality Autotuning

Um die nächste Stufe im Adoption Model zu erreichen, muss es möglich sein, domänenspezifische Regeln zu definieren. Als Beispiel wird die (konstruierten und vereinfachten) Sehgewohnheiten der Netflix Benutzer in USA und Japan (siehe Abbildung 5) eingeführt. Angenommen, um 18 Uhr amerikanischer Zeit schauen die meisten Nutzer in den USA Netflix, vorrangig Serien wie *House of Cards*. Zur gleichen Zeit ist es in Japan 7 Uhr am Morgen. Die meisten japanischen Nutzer frühstücken und gehen auf Arbeit. Dabei wird der Videostreamingdienst nicht genutzt. Somit kommt es zu unterschiedlichen regionalen Auslastungen. Elf Stunden später, ist es 18 Uhr in Japan. Die japansichen Nutzer benutzen den Videostreamingdienst um Ihre Lieblingsanimes zu schauen. Zu dieser Zeit ist es in den USA 5 Uhr in der Nacht. Nun liegt eine erhöhte Last in der japanischen Region. Aber auch der Inhalt unterscheidet sich regional. Während in den USA Realfilme und Serien im Fokus stehen, sind es in Japan animierte Serien. Realfilme benötigen eine Hohe Bildrate, haben wenig Bildwiederholungen, die Einzelbilder enthalten wenig Muster durch eine hohe Farbtiefe. Animierte Serien haben hingegen eine niedrigere Bildrate, viele Bildwiederholungen und enthalten viele Muster. Somit würden bei beiden Inhaltstypen verschiedene Komprimierungen zu besseren Ergebnissen führen. Durch eine veringerte Datengröße wird sowohl die Bandbreite des Nutzers weniger belastet, als auch die Traffickosten von Netflix verringert. Verschiedene Codierungen können auch die Energiekosten und Serverauslatung verringern. Dies führt wiederrum zu niedrigeren Kosten.

An diesem konstruierten Beispiel wird deutlich, dass es im Sinne der **self-optimizing** Eigenschaft sinnvoll ist, abhängig vom Inhalttyp, Region, Uhrzeit und Nutzerzusammensetzung verschiedene Implementierungen der Dienste im System einzusetzen. Führt ein System diese Optimierungen autonom aus, kann es in die Kategorie *Closed Loop With Business Priorities* des Adoption Models eingeordnet werden.

Mit der Optimierung von Systemen basierend auf mehreren Qualität beschäftigt sich das Projekt MQuAT Götz 2013. Dieses beschreibt ein Framework zur Beschreibung von Nutzeranfragen mit verschiedenen Qualitätsanforderungen. Die komponentenbasierte Software besitzt mehrere Implementierungen pro Komponente mit verschiedenen Anforderungen an die Hardware und bietet dafür verschiedene Qualitäten. Nun liegt ein Optimierungsproblem vor. Die Komponenten müssen so ausgewählt werden, dass die Qualitätsansprüche vom Nutzer erfüllt sind und dabei die kostengünstigste Lösung entsteht. Qualitäten sind dabei keine kontinuierliche Größe, da ein menschlicher Benutzer erst ab einer bestimmten Abweichung einen Unterschied erkennen kann (Weber-Konstante). Das Optimierungsproblem lässt sich mit Verfahren wie Integer Linear Programming exakt oder dem Ameisenalgorithmus annähernd lösen.

5 Fazit

Diese Arbeit hat die Motivation von Autonomic Computing anhand der verlagerten Komplexität von Microservices in die Infrastruktur gezeigt und die Net-

flix Architektur als State-of-the-Art eingeführt. Die zum Einsatz kommenden Werkzeuge und Komponenten konnten auf die von IBM vorgestellte MAPE-K Referenzarchitekur abgebildet werden. Da der Netflix Aufbau sich über die Jahre aus der Industrie herausentwickelt hat, ist dies ein Indiz dafür, dass MAPE-K bzw. der Feedback-Loop ein sinnvolles und notwendiges Konzept für die Umsetzung von Autonomic Computing ist. Dies wird bestärkt durch die Erkenntnis, dass Netflix *Scryer* intern einen ähnlichen Kreislauf besitzt.

Darüber hinaus wurde durch die Einordnung in das Autonomic Computing Adoption Model gezeigt, dass das Neflix System eine hohe Autonomität besitzt und einen *Closed Loop* implementiert. Aufgrund fehlender Quellen zu Definitionsmöglichkeiten von abstrakten, domänenspezifischen Businessregeln und Service Level Agreements, kann die Architektur nicht in die Kategorie *Closed Loop With Business Priorities* eingeordnet werden.

Als Ausblick auf den aktuellen Stand der Forschung wurde die Arbeit MQuAT Götz 2013 kurz eingeführt und ein Beispiel zum Einsatz im Videostreaming vorgestellt. Dieser Ansatz würde zu einer Erhöhung der Autonomität und Automatisierung führen und die self-optimizing Eigenschaft verbessern. Damit wäre eine Einordnung in die Kategorie *Closed Loop With Business Priorities* möglich.

Das MQuAT Projekt ist in seiner Konzeption stark gekoppelt mit dem Cool Component Model. Die Ansätze zur Definition der Qualitäten und verschiedenen Implementierung wären grundsätzlich auch mit Microservices umsetzbar. Eine Weiterentwicklung in diese Richtung würde die Forschungsarbeit mehr in den Kontext verteilter Systeme und den aktuellen Industriebedarf heben.

6 Literatur

Computing, A. “An architectural blueprint for autonomic computing”. In:

Dobson, S., S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, F. Saffre, N. Schmidt und F. Zambonelli (2006). “A survey of autonomic communications”. In: ACM Transactions on Autonomous and Adaptive Systems (TAAS) 1.2, S. 223–259.

Götz, S. (2013). *Multi-quality Auto-tuning by Contract Negotiation*. URL: `https://books.google.de/books?id=0d3roAEACAAJ`.

Hindman, B., A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. H. Katz, S. Shenker und I. Stoica (2011). “Mesos: A Platform for Fine-Grained Resource Sharing in the Data Center.” In: NSDI. Bd. 11, S. 22–22.

Horn, P. (2001). *Autonomic computing: IBM's Perspective on the State of Information Technology*.

Newman, S. (2015). *Building Microservices*. O'Reilly Media, Incorporated. URL: `https://books.google.de/books?id=1uUDoQEACAAJ`.

Oreizy, P., M. M. Gorlick, R. N. Taylor, D. Heimbigner, G. Johnson, N. Medvidovic, A. Quilici, D. S. Rosenblum und A. L. Wolf (1999). “An architecture-based approach to self-adaptive software”. In: IEEE Intelligent systems 14.3, S. 54–62.

Shore, J. (2004). “Fail fast [software debugging]”. In: IEEE Software 21.5, S. 21–25.

Mögliche Einflüsse von autonomen Informationsdiensten auf ihre Nutzer

Jan Engelmohr

Technische Universität Dresden, Fakultät Informatik

Zusammenfassung. Informationsdienste sind in der heutigen Zeit allgegenwärtig. Häufig verarbeiten sie Nutzerdaten autonom und passen ihre präsentierten Informationen an den Nutzer an. Im ersten Teil des Papers wird die grundlegende Funktionsweise solcher Algorithmen beleuchtet. Anschließend werden sie in die MAPE-K Referenzarchitektur eingeordnet, um ihre Autonomie aufzuzeigen. Auf dieser technischen Grundlage aufbauend werden im zweiten Teil des Papers sowohl die positiven als auch die negativen Implikationen dieser Algorithmen betrachtet. Dabei wird zwischen Implikationen für den individuellen Nutzer sowie für eine Gesellschaft als Ganzes unterschieden. Abschließend wird unter der Annahme, dass autonome Informationsdienste im Alltag präsenter werden ein möglicher Ausblick auf eine Gesellschaft im Jahr 2025 gegeben.

1 Einführung

Die Suche von Informationen im Internet ist für die meisten Menschen heutzutage ganz normal. Die dabei konsultierten Suchmaschinen suchen längst nicht mehr nur nach den eingegebenen Wörtern, sondern können in der Regel auch den Kontext der Suche mit einbeziehen. Dazu gehören Standortdaten, Nutzervorlieben, vergangene Suchanfragen und weitere Informationen, die der Nutzer preisgibt.
Der betriebene Aufwand ist berechtigt - Nutzer, die ihre Fragen adäquat beantwortet bekommen, werden vermutlich in Zukunft dieselbe Suchmaschine konsultieren. Da diese Suchmaschinen die vergangenen Eingaben des Nutzers in aktuelle Suchanfragen mit einbeziehen, kann man hier von einer Rückkopplung sprechen, welche dem Nutzer bei längerem Gebrauch immer genauerer Informationen liefert. Der Nutzer erhält also besser vorgefilterte Informationen und im Gegenzug wird sein Profil auf Seite der Suchmaschine immer genauer an ihn angepasst. Dabei ist ihm oft gar nicht bewusst, dass der konsultierte Dienst Informationen filtert und auf seine Interessen zuschneidet. Wer hat nicht schon mal schnell etwas „gegoogelt“ und das Resultat als objektive Antwort hingenommen? Die Filterung nach Vorlieben wird jedoch in der Regel als angenehm empfunden und auch ich als Autor bin oft begeistert von den präzisen und auf mich zugeschnittenen Ergebnissen.
Eine neue Form von Suchmaschinen stellen persönliche Assistenzsysteme dar. Sie gehen sogar noch einen Schritt weiter und bieten dem Nutzer proaktiv, also

ohne sein Zutun Informationen an. Diese Informationen beinhalten beispielsweise die Fahrzeit zur Arbeit oder Abfahrtszeiten öffentlicher Verkehrsmittel, um rechtzeitig zu einem Termin zu erscheinen.
Es werden jedoch auch Nachrichten vorgeschlagen. Ist der Nutzer technikaffin, erhält er Neuigkeiten über Smartphones. Interessiert er sich für Fußball, erhält er automatisch die Ergebnisse eines Länderspiels oder direkt einen Link zum Livestream. Beschäftigt er sich viel mit Ernährung, werden ihm möglicherweise Videos mit ausgewogenen Kochrezepten vorgeschlagen. Besucht er gerade eine fremde Stadt, bietet das Assistenzsystem Links zu Sehenswürdigkeiten in der Nähe an.
Die aktive Präsentation dieser Informationen wird wie bereits erwähnt in der Regel als sehr angenehm empfunden. Mit dem naiven Konsum dieser Informationen spielt der Nutzer jedoch eine passivere Rolle, was Risiken mit sich bringen kann. Es ist für den Nutzer einfacher, simple Fragen mit „Ja“ oder „Nein“ zu beantworten, als erst selbst eine Frage zu formulieren, diese dann einzutippen und ein befriedigendes Ergebnis zu suchen. Jedoch werden durch diese binäre Entscheidung oftmals alle weiteren Optionen - nämlich die, die gar nicht erst vorgeschlagen wurden - ausgeblendet.
Es wurde in Studien gezeigt, dass sich Manipulationen der präsentierten Informationen direkt auf den Nutzer und sein Verhalten auswirken. Facebook führte bei Wahlen im Jahr 2012 einen „I voted.“-Button ein. Diesen Button konnten Nutzer auf ihrem Profil anzeigen lassen, wenn sie angaben sich bei der Wahl beteiligt zu haben. Hochrechnungen zufolge kann dieser Button für eine um bis zu 0,4% höhere Wahlbeteiligung gesorgt haben [7].
Diese und weitere Seiteneffekte solcher Systeme sollen in diesem Paper näher betrachtet werden. Außerdem werden mögliche Auswirkungen - sowohl positiv als auch negativ - aufgezeigt.

2 Grundlagen und Definitionen

2.1 Autonomes System

Als autonomes System wird ein System bezeichnet, welches die folgenden Kriterien erfüllt [4].

Selbstkonfiguration Das autonome System ist der Lage, sich selbst zu konfigurieren, das heißt, dass es sein Verhalten auf ihm unbekannte Situationen adaptieren kann.

Selbstheilung Es kann Störungen oder andere Probleme erkennen und sein Verhalten so adaptieren, dass es weiterhin fehlerfrei funktioniert. Ist dies nicht möglich, ist es immer noch in der Lage, nach Beseitigung der Störung autonom in den Normalzustand zurückzukehren.

Abb. 1. Die vier Aspekte eines autonomen Systems. ©Jan Engelmohr

Selbstschutz Außerdem ist es einem autonomen System möglich, sich selbst adaptiv vor externen Angriffen oder Störungen zu schützen und somit seinen Service sicherzustellen.

Selbstoptimierung Zuletzt ist ein autonomes System in der Lage, sich selbst hinsichtlich neuer Anforderungen zu optimieren.

2.2 MAPE-K

Die MAPE-K Referenzarchitektur besteht aus den 5 Elementen **M**onitor, **A**nalyze, **P**lan, **E**xecute und **K**nowledge. Sie wurde von IBM im Jahr 2005 vorgestellt [3] und dient zur Veranschaulichung des allgemeinen Ablaufs eines autonomen Systems. Es handelt sich um eine Struktur, deren Ablauf zyklisch ist, deren Ausgaben also unter anderem als erneute Eingaben für den nächsten Zyklus dienen. Die Abbildung 2 zeigt den Aufbau dieser Architektur.
In der Monitor-Phase werden Eingaben mittels Sensoren eingelesen und in der Analyze-Phase analysiert und kontextuell eingeordnet. Die Plan-Phase beinhaltet die geplante Reaktion des Systems auf den eingelesenen Zustand, welche in der Execute-Phase mittels Aktoren ausgeführt wird. Diese Reaktion geht im nächsten Zyklus dann mittels Messungen der Sensoren wieder in die Monitor-Phase ein. Auf Grund dieses zyklischen Aufbaus ist es einem autonomen System nach MAPE-K Architektur möglich, Wissen (Knowledge) über seine Umwelt zu erlangen. Dieses Wissen nutzt das autonome System, um adäquat auf Veränderungen zu reagieren.

Letztlich ist die Aneignung von Wissen das, was ein autonomes System von Systemen unterscheidet, die lediglich Eingaben auswerten und mit immer gleichen Ausgaben reagieren. Das autonome System kann somit bei identischer Eingabe durch Sensoren verschiedene Ausgaben liefern, da es seine Wissensbasis mit in die Entscheidung einbezieht.

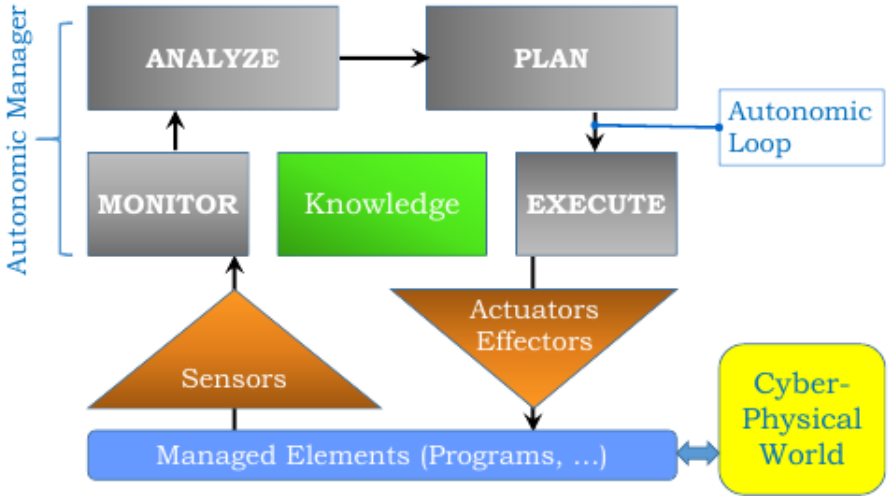

Abb. 2. MAPE-K Referenzarchitektur von IBM ©Frank J. Furrer

3 Autonome Informationsdienste in der heutigen Zeit

Informationsdienste wie 'Google Now' oder Facebooks 'M' sind heutzutage allgegenwärtig und werden von immer mehr Menschen benutzt. Diese Dienste stellen ebenfalls autonome Systeme dar, wie im Folgenden am Beispiel 'Google Now' gezeigt werden soll. Bevor das Beispiel jedoch näher beleuchtet wird, soll ein kurzer Abriss die Unterschiede zwischen autonomen Informationssystemen und einfacher Suchmaschine darstellen.

3.1 Klassische Suchmaschinen im Vergleich mit autonomen Informationssystemen

Klassische Suchmaschinen sind - verglichen mit autonomen Systemen - einfacher aufgebaut. Sie indizieren die zu durchsuchenden Daten in regelmäßigen Abständen und ordnen diese in einer internen Datenbank ein. Die verwendete Metrik kann sich unterscheiden, jedoch liegen die Daten immer irgendwie geordnet in der Datenbank vor. Die Metrik des früher von Google verwendeten

einfachen PageRank-Algorithmus bewertet beispielsweise eine Website, die von vielen anderen Websites verlinkt wird, hoch [5]. Der Gedanke dahinter ist, dass diese Website von vielen Menschen verwendet wird und daher höher in den Suchergebnissen angezeigt werden sollte.
Sucht ein Nutzer nun etwas - beispielsweise einen Text - wird die Suchmaschine die Texte in ihrer Datenbank mit der Eingabe vergleichen. Dann wendet sie eine Metrik wie PageRank an, um die Ergebnisse zu staffeln und liefert sie als Antworten zurück. An dieser Stelle ist die Aufgabe der klassischen Suchmaschine beendet. Eine erneute Suche nach demselben Text wird dieselben Ergebnisse zurückliefern. Abbildung 3 veranschaulicht diese Vorgehensweise. Es ist besonders zu beachten, dass jegliche Interaktion vom Nutzer ausgeht (vgl. Punkt 1: Nutzer stellt Anfrage).
An dieser Stelle sei gesagt, dass Googles PageRank natürlich weiterentwickelt wurde und heutzutage ebenfalls autonome Systeme für die Suche verwendet, aber dazu später mehr.
Besagte **Autonome Informationssysteme** gehen einige Schritte weiter als klassische Suchmaschinen. Sie beobachten, welche Information der Nutzer auswählt und reichern mit diesem Wissen ein internes Profil über den jeweiligen Nutzer an. Aber auch bei der Suche selbst verhalten sich diese Systeme anders. Fragt ein Nutzer wieder den beispielhaften Text an, zieht das System neben seiner Suchdatenbasis ebenfalls das Nutzerprofil zurate. Dieses dient als zusätzliche Metrik, um eine bessere Entscheidung darüber treffen zu können, was der Nutzer eigentlich wissen möchte. Die Konsequenz daraus ist, dass keine zwei Nutzer bei gleicher Eingabe identische Ergebnisse erhalten, wenn sich ihre Nutzerprofile unterscheiden.
Eine besonderes Ausprägung von autonomen Informationssystemen ist das **persönliche Assistenzsystem**, dass zusätzlich von sich aus Informationen bereitstellt. Abbildung 4 zeigt diesen Aufbau schematisch. Es ist zu beachten, dass das System aktiv die Initiative ergreift und berechnet, dass Nutzer Informationen benötigen könnte (vgl. Punkt 2). Anschließend stellt es ungefragt Informationen bereit (vgl. Punkt 5).

3.2 Beispiel Google Now

Googles Philosophie mit 'Now' ist es, Informationen für den Nutzer bereitzustellen, bevor dieser eine Frage stellt [2]. Dafür speichert Google die eingegebenen Suchanfragen und Klicks eines Nutzers, sowie seine Standortdaten, seine regelmäßig besuchten Orte, eingebuchte WLANs, Kontaktdaten seines Adressbuches und vieles mehr (vgl. *Monitor/Knowledge*), analysiert diese Daten hinsichtlich Stichworten und semantischen Themen (vgl. *Analyze*) und extrapoliert, welche Themen, Artikel oder Videos für den Nutzer relevant sein könnten (vgl. *Plan*). Die Relevanz der Themen wird dabei mit über 200 Faktoren bewertet [1] Schlussendlich werden diese neuen Informationen dem Nutzer präsentiert (vgl. *Execute*) und seine Reaktion erneut aufgezeichnet (vgl. *Monitor*). Abbildung 5 zeigt einen Screenshot des Dienstes auf einem Smartphone. Die Artikel im unteren Bereich

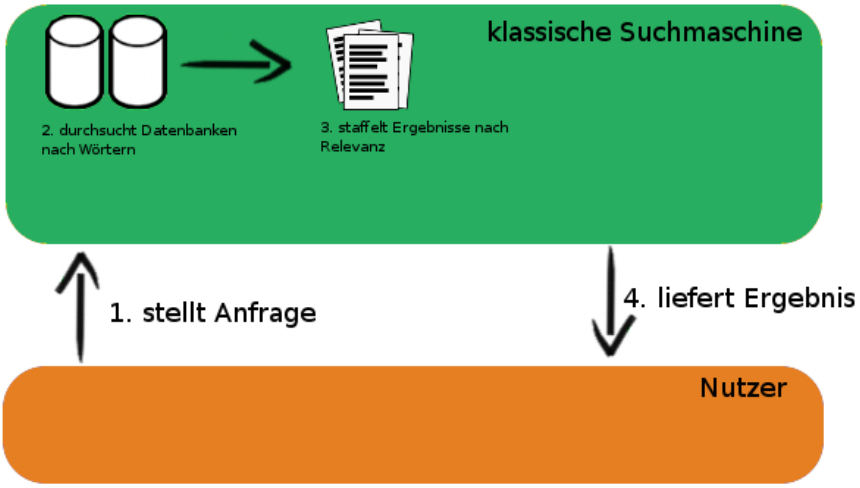

Abb. 3. Klassische Suchmaschine ©Jan Engelmohr

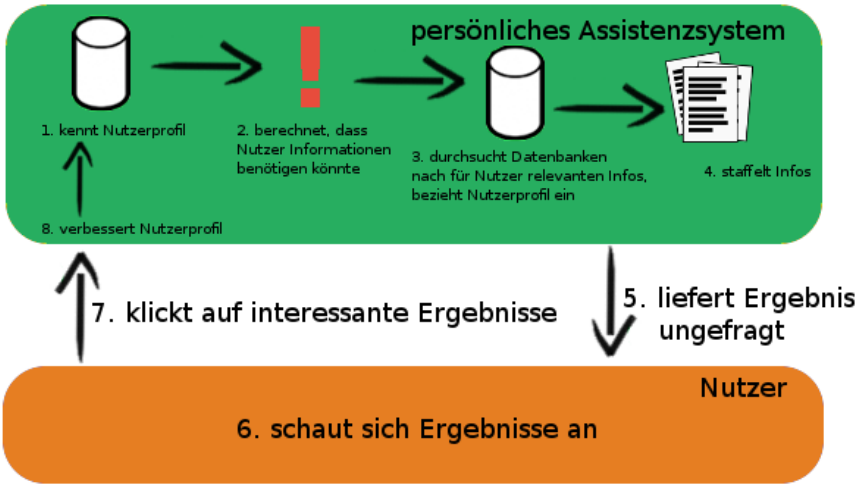

Abb. 4. Persönliches Assistenzsystem ©Jan Engelmohr

werden proaktiv von 'Now' vorgeschlagen, der Nutzer hat nicht danach gesucht.

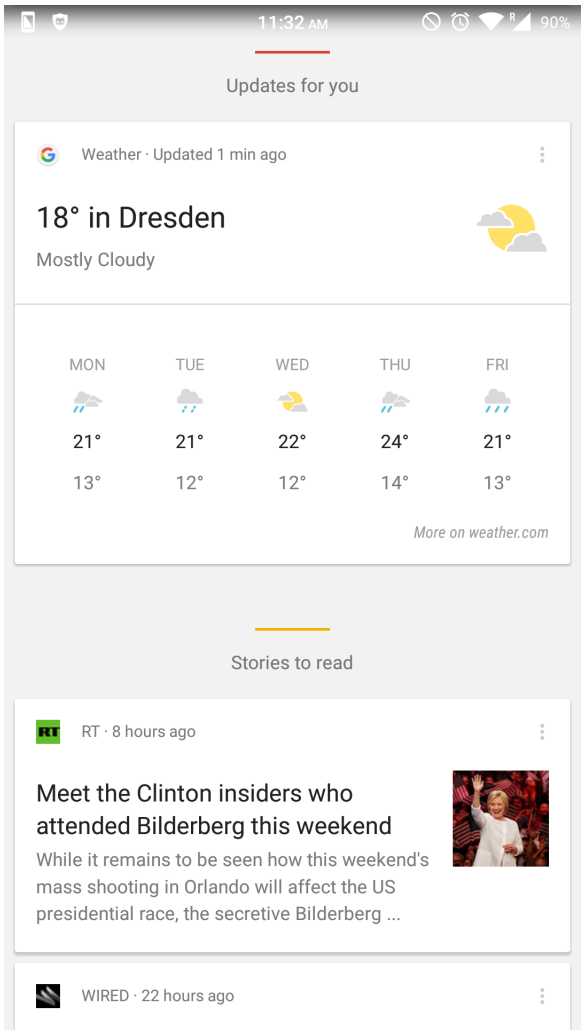

Abb. 5. Screenshot aus Dienst 'Google Now'. „Stories to read" beinhaltet Medienartikel, die vermutlich den Interessen des Nutzers entsprechen. ©Jan Engelmohr

4 Vorteile autonomer Informationsdienste für den Nutzer

Autonome Informationsdienste bieten dem Nutzer eine Reihe Vorteile.

4.1 Geschwindigkeit

Der offensichtlichste Vorteil ist das schnelle Finden von Informationen. Das System „weiß“ durch das vorhandene Nutzerprofil, welche Informationen der Nutzer sucht und kann diese gezielt bereitstellen.
Dadurch, dass es eigenständig Informationen in einen Kontext einordnet und mit verwandten Themen verknüpft, kann der Nutzer mit geringem Aufwand viele Informationen zu einem Thema finden. Durch eine auf die Vorlieben des Nutzers zugeschnittene Themenauswahl des autonomen Systems muss sich dieser nicht mit unnötigen oder für ihn irrelevanten Themen beschäftigen und hat somit mehr Zeit, sich zu Themen zu informieren, die ihn interessieren. Er kann sich leicht Hintergrundinformationen beschaffen, da das System diverse Internetseiten durchsucht und als Metainstanz Informationen mehrerer Quellen liefert.

4.2 Qualität der Ergebnisse

Als weiterer Vorteil ist der Ablauf des autonomen Systems zu nennen, da der Nutzer durch Klicks auf für ihn interessante Themen dem Systems kontinuierlich mitteilt, welche Interessen er besitzt und ihm das System daraufhin noch bessere Ergebnisse liefern kann.
Die Staffelung der Ergebnisse ist dadurch für den Nutzer besser, da sie bereits vorsortiert sind.

4.3 Erinnerungsfunktion

Persönliche Assistenzsysteme verfügen unter anderem auch über Funktionen, die den Nutzer an bevorstehende Ereignisse oder Termine erinnern. Weiß das System, dass es sich um ein Meeting handelt, blendet es Telefonnummern der Teilnehmer oder den Standort auf einer Landkarte ein. Das System bezieht bei räumlich entfernten Terminen sogar den Standort des Nutzers ein und erinnert ihn je nach Verkehrslage rechtzeitig an die Aufbruchszeit.
Es fungiert also wie ein Sekretär und wirkt unterstützend bei der Zeitplanung.

4.4 Eingabengenauigkeit

Da das System die Interessen des Nutzers und seinen Standort sowie seine Ansichten kennt, kann es auch bei ungenauen Eingaben wissen, was der Nutzer finden möchte.
Dies lässt das System menschlicher wirken, da Fragen nicht exakt sondern eben ungefähr formuliert werden können.

4.5 Zeitersparnis

Das Internet stellt heute sehr viele Informationen zur Verfügung. Oftmals ist es schlicht nicht möglich, aus dieser Fülle von Informationen manuell die relevanten Punkte herauszusuchen. Hier kann eine Vorfilterung gute Dienste leisten und dem Nutzer somit die Suche erleichtern.

5 Nachteile autonomer Systeme für den Nutzer

Die Funktionsweise des autonomen Informationssystems kann für den Nutzer ein Nachteil werden, wenn dieser das System allzu naiv nutzt.

5.1 Filterblasen

Auf Grund der immer genaueren Spezifizierung des Modells, welches über den Nutzer existiert, wird es zunehmend schwieriger, auf andere als vom Informationsdienst vorgefilterte Themen zu stoßen [6]. Auf die MAPE-K Architektur übertragen gewichtet der *Analyze*-Schritt die erhaltenen Daten stark in Richtung des vorhandenen Nutzermodells. Dies mag auf den ersten Blick nach einem Vorteil für den Nutzer klingen, jedoch beinhaltet diese Vorgehensweise auch Risiken. Werden Ergebnisse naiv konsumiert, kann leicht eine durch das System verzerrte Meinung zu einem Thema entstehen. Erhält der Nutzer nur auf seine Interessen zugeschnittene Ergebnisse und Informationen, ist es schwerer für ihn, unangenehmen Themen, oder Themen, zu denen er eine andere Meinung hat, zu finden [6]. Man spricht in diesem Zusammenhang auch bildlich von einer „Filterblase“, in der sich der Nutzer befindet. Diese kann dazu beitragen, dass sich seine Sicht auf die Welt stetig festigt, da er bei jeder Informationsdarbietung des Systems Inhalte erhält, die er sowieso schon präferiert. Dem Nutzer unangenehme Inhalte werden weggefiltert, sodass sich dieser diesen Themen möglicherweise nicht bewusst ist. Eine Auseinandersetzung mit neuen und unbekannten Sichtweisen oder unangenehmem Tagesgeschehen ist jedoch zur persönlichen sowie gesellschaftlichen Entwicklung unerlässlich [7].

Beispiel Sucht der Nutzer nach „Käse verursacht Krebs“ und klickt nur auf Links, die behaupten, dass Käse Krebs verursacht, lernt das System dieses „Interesse“. Bei neuen Suchanfragen bezüglich des Themas werden Verweise zu Websites, die das Gegenteil belegen gar nicht oder weit hinten angezeigt.
Hierbei handelt es sich jedoch nur um eine Gefahr, wenn der Nutzer naiv vorgeht und seine Meinung lediglich bestätigt haben will, statt wirklich Informationen zu erhalten.

5.2 Datenschutz und Quasi-Monopole

Hinzu kommt, dass Google aktuell die mit Abstand am meisten konsultierte Suchmaschine ist [8]. Damit verstärkt sich das Problem der Filter Bubbles, da die

Nutzer de facto meistens Google konsultieren wird und durch die Anpassungen die „guten“ Ergebnisse gewohnt ist. Ebenfalls ist es möglich, dass Nutzerdaten entwendet und zu schadhaften Zwecken missbraucht werden. Diese Punkte sollen jedoch nur in einer Randnotiz erwähnt werden, da sie keine Kritik an autonomen Informationsdiensten als solche darstellen.

6 Mögliche Einflüsse auf die Gesellschaft

Wie bereits gezeigt, gehen autonome Informationssysteme sowohl mit Vor- als auch mit Nachteilen für den Nutzer einher. Im Folgenden soll der Fokus auf einem möglichen gesellschaftlichen Einfluss solcher Systeme auf ein demokratisches Land im Jahr 2025 liegen. Dabei wird die Annahme getroffen, dass sich autonome Informationssysteme weiterhin ausdehnen und nahezu omnipräsent werden. Konkret heißt das, dass sich das Internet nicht mehr auf den virtuellen Raum beschränkt, sondern zunehmend durch Entwicklungen wie das Internet der Dinge Einzug in den physischen Raum hält.

6.1 Annahme: Positive Entwicklung

Die Grundlage dieser Aussicht beruht darauf, dass sich die Nutzer von Informationsdiensten bewusst sind, dass es sich bei diesen ebenfalls um Firmen handelt, welche eigene Interessen verfolgen. Folglich werden Nutzer unter dieser Annahme kritischer mit präsentierten Informationen umgehen und sie hinterfragen. Als nächsten Schritt wäre es denkbar, dass sich Nutzer bei mehreren Informationsquellen informieren, um die Entstehung von Filterblasen zu erschweren. Damit können sie jegliche Art von Informationssystemen nutzen und somit eine ungemeine Menge an Informationen abrufen. Dank des Wissens der Systeme werden den Nutzern genau zur richtigen Zeit Informationen präsentiert.
Nimmt man das Internet der Dinge in die Betrachtung auf, ist davon auszugehen, dass im Jahr 2025 fast jeder Mensch mindestens ein Gerät mit Internetanschluss bei sich hat. Diese sind dann voraussichtlich für autonome Assistenzsysteme entworfen, sodass sie dem Nutzer viel Verwaltungsaufwand abnehmen können. Nun ist denkbar, dass diese individuellen Geräte verstärkt zusammenarbeiten, um verknüpfte Assistenzsysteme mit noch genaueren Daten zu versorgen. Der Bürger im Jahr 2025 erhält somit präzise Informationen zu seinem aktuellen Standort und erhält diese zur richtigen Zeit. Assistenzsysteme planen seinen Tag optimal und erinnern an fällige Termine.
Weiterhin ist für dieses Szenario vorauszusetzen, dass die Firmen hinter den autonomen Assistenten keine Daten missbrauchen und für die Sicherheit der Nutzer garantieren.

6.2 Annahme: Negative Entwicklung

Dieses Szenario setzt voraus, dass der Nutzer einen Dienst besonders stark beansprucht und seine Antworten nicht oder nur unzureichend hinterfragt. Es könnte

so aussehen, dass sich der Nutzer Informationen noch stärker auf seine Interessen zuschneiden lässt. Diese Möglichkeit der Informationsbeschaffung ist für ihn natürlich in erster Linie bequem und schnell. Ihm werden die Wünsche von den Augen abgelesen. Allerdings setzt er sich auch nicht mehr so aktiv mit seiner Umwelt auseinander, bzw. behandelt nur Themen, die ihn sowieso bereits interessieren.
Interessiert sich ein Nutzer in seiner Freizeit für Volleyball und Kochen, schlägt ihm ein konsultiertes Videoportal Videos vor, die Volleyball und Kochen als Thema haben. Es ist durchaus möglich, dass er in diesen Themengebieten besonders tiefgreifend informiert ist und Expertise entwickelt, da die Filterblase immer weitere Informationen liefert.
Jedoch wird es ihm schwerer gemacht, neue Themen zu entdecken, die ihn vielleicht auch interessieren könnten. Das System schlägt ihm erst einmal altbekannte Dinge vor, da es von seinen neuen Interessen kein Wissen hat.
Neben Themengebieten werden auch Meinungen gefiltert. Deckt sich eine Meinung nicht mit der eigenen, filtert das System diese einfach weg. Dabei nimmt jedoch das kritische Hinterfragen von Meinungen und Inhalten ab - schlimmer noch: Die eigene Meinung festigt sich immer weiter in einer unendlichen Feedback-Schleife. Ist dem Nutzer an dieser Stelle die Filterung nicht bewusst, könnte er irrtümlich annehmen, dass ihm das „gesamte Internet“ zustimmt.
Andere Meinungen oder verstörende Bilder oder Ereignisse werden ebenso ausgeblendet. Der Nutzer kann somit verlernen, sich mit unbequemen Themen auseinanderzusetzen und verbleibt in seiner Filterblase, da diese ihm ständigen Zuspruch und eine Bestätigung liefert.
Betrachtet man die Problematik als gesellschaftliches Problem, könnte es passieren, dass ihre Mitglieder das Interesse an Diskussionen verlieren, da sie es schlicht und ergreifend kein Interesse daran haben, sich mit konträren Meinungen tiefergehend auseinanderzusetzen. Für die Evolution einer Gesellschaft ist jedoch Diskurs zwingend notwendig [7]. Es braucht gegensätzliche Meinungen und Streits, um einen Konsens zu finden. Ist niemand mehr bereit, sich mit anderen Meinungen auseinanderzusetzen, erstarrt diese Evolution. Die Geschichte hat gezeigt, dass immer wenn eine Institution viel Macht hatte, diese auch gegen das Wohl der Allgemeinheit missbraucht wurde. Ändert ein Dienst seine internen Algorithmen, sodass bestimmte Ergebnisse einfach für keinen Nutzer angezeigt werden, kann dieser die Nutzer tatsächlich manipulieren. So könnten bestimmte Meinungen gezielt gefiltert werden.

7 Fazit

Abschließend bleibt zu sagen, dass die Meinung des Autors gegenüber autonomen Assistenz- und Informationssystemen eher positiv ist. Es ist seiner Meinung nach nicht von einer völligen Zerstörung der Gesellschaft auszugehen, nur weil Assistenzsysteme marktreif werden. Beide Annahmen - also extrem positiv als auch extrem negativ - werden vermutlich nicht in der Reinform eintreten. Hier ist es wichtig, Abstufungen zwischen den Extremen zu beleuchten. Nicht alle

Menschen haben dieselben Voraussetzungen und während manche Suchergebnisse reflektieren, tun andere dies nicht.
Kritik an diesen Systeme ist zwar berechtigt, jedoch beruht sie meist auf der Annahme, dass der Nutzer dieser Systeme völlig hilflos und unreflektiert in ihrer Nutzung ist. Dies entspricht in der Regel nicht den Tatsachen und ist genau betrachtet keine wirkliche Kritik. Jedes Medium beeinflusst Menschen bewusst oder unbewusst. Es ist möglich, die Beeinflussung zu minimieren, indem beispielsweise mehrere Quellen zurate gezogen werden oder indem angebliche Fakten logisch hinterfragt werden. Diese Fähigkeit sollten Menschen jedoch ohnehin mitbringen, wenn sie Medien konsumieren.
Autonome Systeme werden nach Meinung des Autors die Gesellschaft in vielen Bereichen unterstützen, wenn sie richtig, also mit einem gewissen Medienverständnis genutzt werden.

Literatur

1. Google: How Google Search Works (2016), https://support.google.com/webmasters/answer/70897?hl=en
2. Google: Now Cards - The Google app (2016), https://www.google.com/landing/now/#whatisit
3. de la Iglesia, D.G.: MAPE-K Formal Templates for Self-Adaptive Systems: Specifications and Descriptions (2014), http://homepage.lnu.se/staff/digmsi/MFT/MAPEK-FT_documentation.pdf
4. Ltd, V.P.: Modes of Computing, http://www.users.globalnet.co.uk/~rxv/so/computing.htm
5. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab (November 1999), http://ilpubs.stanford.edu:8090/422/, previous number = SIDL-WP-1999-0120
6. Pariser, E.: Beware online filter bubbles"(2011), http://www.ted.com/talks/eli_pariser_beware_online_filter_bubbles
7. Schneier, B.: Data and Goliath S. 114, Z. 10ff. Norton & Company (2016)
8. Statistica: Marktanteile der Suchmaschinen weltweit nach mobiler und stationärer Nutzung im Dezember 2015, http://de.statista.com/statistik/daten/studie/222849/umfrage/marktanteile-der-suchmaschinen-weltweit/

The Benefits of Resolving the Trust Issues Between Autonomic Computing Systems and Their Users

Marc Kandler

Technische Universität Dresden: Andreas-Pfitzmann-Bau, Fakultät Informatik,
Nöthnitzer Str. 46, 01187 Dresden, Germany
marc.kandler@tu-dresden.de
http://www.inf.tu-dresden.de/

Abstract. The technological advancement of mankind progresses relentlessly. With the introduction of automation and computing systems, many task which could previously only be handled by humans were taken over by machines. We see a continuation of that flow and therefore the emergence of autonomic computing systems. However, in order to be able to fully utilize these systems, trust by their users will be required. In this work, we will show why trust is important for the success of autonomic computing systems. We will present the categories of technical factors, psychological factors, and the need for a foundation in law and society, which all are relevant for developing trust. We will choose a subset of factors from each category, which we deem to be vital for developing a trust relationship to these systems in the future. Furthermore, we will explain why each factor is relevant, and how to possibly cope with them. As an example of an autonomic computing system, the Google car will be presented, as well as how Google intends to deal with the presented factors. Finally, we will give an outlook on what impact of autonomic computing systems we can expect on society, work, and people by 2025.

Keywords: autonmic computing, autonomic computing systems, human-computer trust, google car, confidentiality, integrity, availability, experience, technical factors, psychological factors

1 Introduction

In the last decades, computing systems took over many tasks which could previously only be handled by humans. With technological advancements, said systems could even overcome human limitations, which led to many positive aspects like higher efficiency and higher productivity.

As the size as well as the complexity of computing systems keeps increasing, the tasks quickly outgrow the comprehension of their users. Therefore the concept of autonomic computing was introduced in 2001 by IBM [24] to tackle the upcoming challenges. Autonomic computing systems can supposedly manage themselves given only high-level objectives from their users or administrators

[26]. On one hand, those systems must actually be technically able to meet this requirement. On the other hand, a very important factor comes into play, which can make or break the usage and spread of autonomic computing systems. This factor is **trust** between the human user and the system. We distinguish three classes of factors, which have an impact on the trust relationship between autonomic computing systems and their users. The first class are the *technical factors*. These factors refer to the relationship from the side of the system. This means that technical factors have to be fulfilled by the system in order to lay the foundation for humans to trust it. Satisfying this requirement is a prerequisite, but no guarantee for trust. The relevance of each factor might vary from system to system, but in general all of these properties need to be addressed by the system and its developers.

The second class are *psychological factors*. Psychological factors describe the human side of this relationship. They possibly apply to any human and everybody might have other priorities regarding these factors. There is even the potential of some factors being unsatisfiable, as they might rely on the social surroundings of a user. It is only possible for the likes of developers, publishers, the responsible company, or the technology to influence these factors indirectly and only to a certain extent. Just as the technical factors, psychological factors are a prerequisite for building a trust relationship, but in this case, not necessarily all of them. Additionally, their priority might differ from person to person.

Finally, the *foundation in law and society* is to be seen as a necessary framework for the widespread and acceptance of autonomic computing systems. It is hardly possible for such systems to be widely used if there remain many unresolved matters within the law. Furthermore, once the government has approved of such a system by passing related laws and established other frameworks like certification processes, trust will increase naturally as long as research does not uncover critical flaws.

Trust is not a distinct to the field of autonomic computing, but has been researched for computing systems, mechanical machines, and automation in general [33, 39, 4]. Therefore many factors also apply for autonomic computing, but some new hurdles to overcome arise as well.
This paper will show possible benefits which await if we are able to resolve the existing trust issues between autonomic computing systems and their users. Trust is of extreme importance for further developments, as mankind advances technologically fast enough for (autonomic) computing systems to take over many tasks which have formerly been carried out by humans. The more the stakes rise, the more important trust becomes, as humanity will have increasingly often the opportunity to decide between relying on said systems (i.e. trust them) or to keep relying on humans, which they have always done in the past.

In section 2 we will elaborate the importance of trust for autonomic computing. We will provide a definition of trust in the context of computing systems in general, and derive a workable definition of trust with regard to autonomic computing systems. Section 3 gives an overview of a subset of factors which have an impact on the trust relationship between autonomic computing systems

and their users. We categorized them according to the above mentioned classes, technical factors, psychological factors, as well as the foundation in law and society. On the technical side, we will mainly examine the criteria confidentiality, integrity, and availability (CIA). The psychological factors will be represented by experience, feedback and own control, as well as risk and public information. We will illustrate each factor with an example, and additionally show possible approaches for handling and satisfying these factors in the future. In section 4 we will examine current applications, as well as relevant developments and how those systems gain or intend to gain the user's trust. As a result, we will give an outlook for the year 2025 in section 5 and show the possible impact of autonomic computing systems on society, work, and people. Section 6 concludes this work with a summary and recommendations on what to look out for in the future regarding trust towards autonomic computing systems.

2 Human-Computer Trust in Autonomic Computing

In this section, we will show the relevance of trust for the field of autonomic computing. This will be done by providing a typical definition of trust between humans and computers, especially with regard to automation. From this definition, we will derive a definition of trust from humans towards autonomic computing systems which will be the foundation for this work.

2.1 The Importance of Trust

Why does trust even play a key role? The more autonomous a system works, the less comprehensible are the actions taken by it for their users or humans in general. With increasing size and complexity, the tasks handled by such systems far surpass human capabilities. Even though humans mostly control them in general, they can only grasp a subset of the actions a system performs. This is also due to the fact that only some data is presented to the user, but mostly due to the sheer amount of data the system handles. In this work, we will focus on autonomic computing systems which can actually operate without human input, but allow a user to monitor the system in some way and even suspend its actions if necessary. We will use the term *user* to refer to anybody who interacts with the system or directly uses it for some purpose. This can be an operator monitoring the system or even elderly persons who use autonomous cars to reach their destination.

As autonomic computing systems become more and more relevant, be it the power management of data centers [25], or everyday life-related as in the case of smart homes [11], the more situations arise which require one or more humans to have trust in the system. The focus is clearly shifting from trusting the advice, which a system presents the user in order to guide his decision [34] towards trusting the decision and hence the system itself. If mankind is not able to take that leap, autonomic computing systems may never unfold their true potential. If a user does not trust a system, he will never use it. If a system is not at least

initially developed and started by humans, it can never run as a result. In real world applications mankind will increasingly often have the decision to either use classical systems, which have proven to work well, or rely on more autonomously working systems. The latter - of course - requires trust in the system [39].

With rising importance of the task, trust becomes more relevant. As risk and uncertainty increase respectively, the doubt towards the system climbs as well [14]. It is only natural for humans to question their relationship, be it with other humans or with a system, as the stakes rise. Our highest value, the own life, can only be entrusted to a system, if we can be sure that it can keep us safe.

2.2 A Definition of Trust

The topic of trust between humans and the (computing) systems they use is almost as old as the existence of systems with the ability to provide guidance in solving complex tasks or doing so themselves [38, 36].

In 2000, Madson and Gregor used the following definition of human-computer trust [34]:

> " [...] the extent to which a user is confident in, and willing to act on the basis of, the recommendations, actions, and decisions of an artificially intelligent decision aid."

This definition assumes that humans still are the ones to take the decision. With the aim of autonomic computing systems, this definition does not fully hold. Humans will become bystanders at most, monitoring the system, or just use it without taking part in the decision-making process. Still, a user needs to be confident in the system, its actions and decisions. Otherwise he may interrupt the system unnecessarily as defined in section 2.1, or desist from using it entirely.

In this work, a user might have trust in an autonomic computing system, if the following is true:

> *The system does, what it should do, and does not, what it should not do. Furthermore the user is provided constant feedback about the decision process and has the opportunity to interfere with the system or cease its activity if necessary.*

It has to be noted that this is only a prerequisite for the existence of trust. There are many more relevant factors, some of which we will address in section 3. Every human has own standards for developing trust, especially regarding technical systems. Therefore, we will not be able to present a perfect solution, but only some guidelines for dealing with this topic.

As simple definition for trust, which can also be seen as a foundation for this work, has been provided by Lee et al. [33]:

> *The attitude that an agent will help achieve an individual's goals in a situation characterized by uncertainty and vulnerability.*

In the context of this work, however, we assume that autonomic computing systems do not necessarily only help an individual, but may even do all the work.

Each individual has an own scale of trust for every system he interacts with. In this work we assume that every potential user decides whether the system is not trusted (and not used) at all (red), only partly trusted and used with caution (yellow), or fully trusted and used (green). However, the transitions vary as well as the impact of each factor on trust relationship. Figure 1 illustrates such a scale. It ranges from not trusting the system at all, to trusting it blindly. The upper half shows how user A views the system and at which point he starts to use and trust the system. The bottom half shows a slightly different distribution for user B. Therefore, user B has more doubts and is less likely to use the system. An example for a somebody being situated in the yellow area (system is used, but not to full potential due to missing trust) would be the usage of an autonomous car, where the user is constantly focusing on the road, hands at the steering wheel, highly tense and ready to take over on the first occasion of a discrepancy in the system's action.

At which point exactly an individual is on this scale and how the distribution and size of the three main areas (red, yellow, and green) is, depends on various factors. In fact, starting from our birth we collect information through our senses, perception and education which have an impact on that scale. This can be understood by comparing the willingness to use modern touch-based systems between current teenagers, and elderly who did not grow up with that kind of technology. Even though the second group might grow accustomed to the system and finally trust it, the path to that point might be long, as an elderly would most likely start in the red or yellow area. However, the teenager would already have a sufficient amount of trust in the system (green area) and use it right from the start.

Whether or not it is possible to get a user from having no trust (red) towards either having some (yellow), or full trust (green) in the system and therefore use it, depends on how well the factors of the defined classes are satisfied (see section 3. While the scale itself cannot be changed that easily (e.g. lowering the amount of trust needed for a user to use the system) due to it shaping with the beginning of our life, the point where the user is situated on that scale can be influenced. Some of the factors which have to be considered, as well as possible ways to address them, will be discussed in the following chapter.

3 Trust between autonomic computing systems and their users

This section will give an overview of a subset of factors with an influence on the trust relationship. Each of the named factors has an impact, even though the extent may differ. Each factor will be defined first, before the actual relevance for autonomic computing systems and the trust between their users and them is shown. Finally, we will suggest possible ways to deal with the named factors in

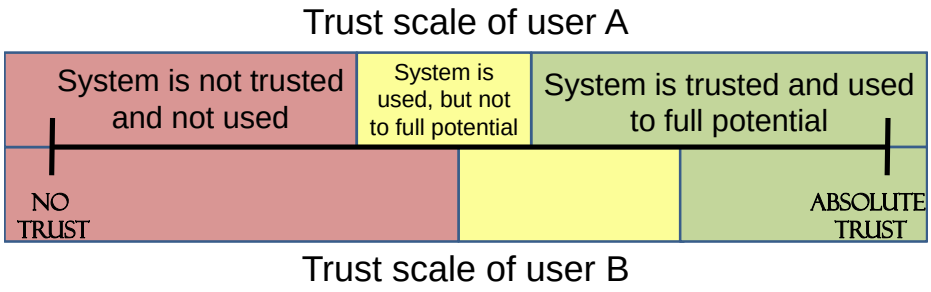

Fig. 1. An abstract scale of trust for users A and B regarding the same system.

autonomic computing systems in order to increase the distribution and effective usage of them.

In order to illustrate our view, we will use the example of an abstract autonomous car, which could exist in the future. There might be some correlation to the *Google Self-Driving Car Project* presented in section 4, but we will deviate whenever necessary without any notice. Our car will be able to operate fully autonomously, without human input, and drive on any street while acting law-abiding. Our car will either always have at least one human passenger, who will be able to monitor the car or intervene with its actions at any time, or some outside monitoring institution, which oversees the car's actions. Furthermore, GPS compability as well as an internet connection are required in order to receive map- and traffic updates, and to navigate accordingly. This connection does not necessarily need to be permanent.

We will briefly explain why we limited ourselves to the few factors from the following sections and name additional aspects which will not be further evaluated in this work. The presented factors are divided into three categories. First, the technical aspects which have to be resolved in order to achieve trust (section 3.1) and in section 3.2 the psychological factors. Finally, in section 3.3 we will give a short introduction into the relevance of establishing autonomic computing systems in law and society. Figure 2 shows our distribution of categories and the respective factors.

3.1 Technical Factors

In this section we will present a subset of factors with an impact on the trust relationship between autonomic computing systems and their users. We chose to work with the security of a system and its data, which can be seen as a composite of the attributes confidentiality, integrity, and availability (CIA) [5]. These properties will be addressed in order to show technical aspects of future developments. There are many more factors on the technical side which have to

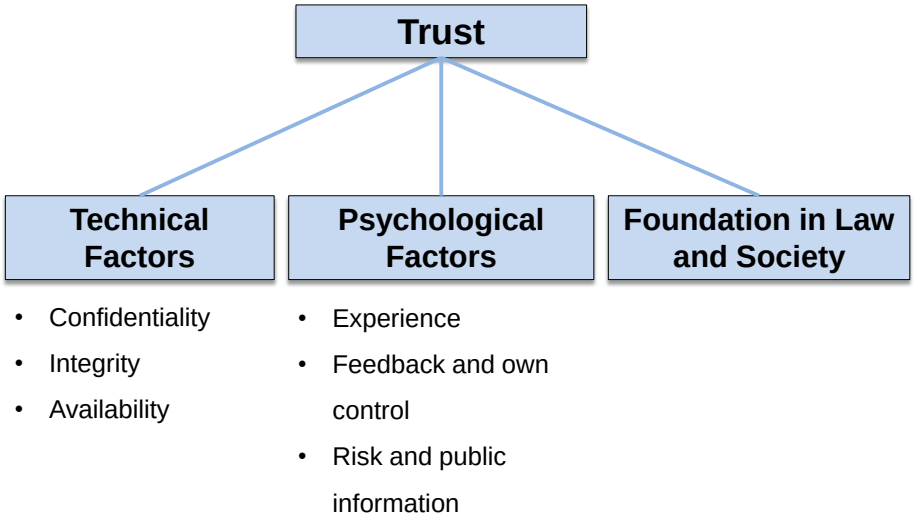

Fig. 2. An overview of how we view factors with an impact on the trust relationship between autonomic computing systems and their users. We see the three main categories *technical factors*, *psychological factors*, and the *foundation in law and society*.

be considered, e.g. accountability, assurance, reliability, safety, maintainability, and others [29, 5]. We decided to limit ourselves to the security properties, in order to 1) be able to address technological, as well as psychological factors in the scope of this work, and 2) highlight the importance of (data) security in future systems, in which CIA play a key role.

But why is security in the form of CIA relevant for human-computer trust, human-system trust, and even for the next step, relevant for the trust of humans in autonomic computing systems? This topic has been well researched in the past [33, 15, 53, 34], partly even before the introduction of autonomic computing. The importance of security in the context of human-computer trust has been highlighted and addressed as a central challenge.

This section provides definitions for the picked security properties. Each property will be presented in the context of autonomic computing systems with the focus on the impact on trust. The properties will be illustrated with the help of our introduced example. We will conclude the discussion of each property by providing approaches which deal with system security, and by giving leads as to how to possibly achieve the set goal of building or improving the trust relationship.

Confidentiality Confidentiality describes a system's capability to limit information access and disclosure to authorized users only [2]. In other words, the system is able to distinguish authorized from unauthorized clients and is therefore only able to grant access to information which is meant for the client. This

can be seen with regard to data, the respective transmission, and also includes the data of the system itself, e.g. the code, or more specifically, certain algorithms [8, 55].

In the context of autonomic computing systems, with our abstract example of the autonomously working car, confidentiality is also a key aspect. As there is a internet connection, there is also the possibility of access from outside. This could be abused by attackers, but can also be necessary for central controlling or remote help departments. As the passenger of our example car might not be able to intervene with the car's actions, for example due to disabilities, there could be a need for such remote help services. Therefore, such feature would have a justification. Furthermore, the data the car exchanges with some outside service could be intercepted and read by an attacker. This data could contain confidential information like the name of the owner of the car, the current geographical position, or the current destination. Such information could easily be abused by an attacker in order to e.g. plan terrorist actions against politicians. But there are many more fields, in which autonomic computing systems could be used, and would have to handle very personal data, which has to be dealt with very confidentially. One example would be the medial sector, where the medical history of a patient underlies very high security standards.

But how can confidentiality be achieved?
As it is a security property, the answer is simple: Better security. There are three dimensions, which have to be addressed:

1. **Confidential user data on the system**
   In general, it is necessary to encrypt confidential data stored on the system, in order to make it useless to third parties in the case of leaks or malicious attacks. This is already done in practice and has proven to be successful, even with some negative side effects [10]. The most common approach would be to use a symmetric key encryption method like AES with a key length of up to 256 Bit. In this way, confidentiality can be granted, even though the biggest problem remains the storing of the actual en-/decryption key. It has to be noted that the protection against malicious attacks is not the task of confidentiality. We listed this as possible threat to confidentiality for the sake of illustration. In general, autonomic computing system would therefore need to encrypt sensitive user data on the system in order to possibly gain a user's trust.
   Note: All of the above holds true for integrity as well.
2. **Transmission of confidential data**
   The second weak point is the communication with the outer world. As we stated in our example, an autonomous car might very well send confidential information e.g. in order to authenticate the client to enable him to use a road map updating service. This is also a central challenge to the Internet of Things (IoT) [46], as well as cloud computing [12], which are directly related to autonomic computing. Encrypted communication is already part of our every day internet usage, e.g. whenever sensitive data like a password is sent through the internet via the HTTPS protocol. This protocol grants reason-

able confidentiality, but is also partly susceptible to man-in-the-middle attacks [7, 17]. As the assumed communication of our example car would work via the internet, the usage of a secure protocol like HTTPS is likely. There is also fully asymmetric encryption possible, which would require a encryption key exchange over an other (secure) channel. An autonomic computing system would therefore need to enable and support encrypted communication in order to possibly gain a user's trust.
Note: All of the above holds true for integrity as well.

3. **Confidentiality of the system itself, code confidentiality**
Lastly, the system itself and its code has to be kept confidentially. This can be of utmost importance to any autonomic systems used in the finance sector, as there are easily millions of dollars on the line. In order to achieve something like that, it is possible to encrypt the code during runtime, making it resistant against static analysis [8]. The typical byte code format is not enough to protect against e.g. reverse engineering, as all the relevant information is still accessible. Another approach is the obfuscation of the code [3]. This means applying semantic-preserving transformations to the code, which would in turn increase the difficulty of automatically extracting code or relevant information from the heap. An autonomic computing system would therefore need to take measures protecting its own code, in order to gain the trust of potential users. In this case, this would most likely be companies which want to keep their methods and algorithms a secret.
Note: All of the above holds true for integrity as well.

Integrity Integrity describes a system's capability to preserve the structure and content of information resources [2]. In other words, the system has to be able to protect information handled or transmitted by it from alteration or deletion by unauthorized clients. This can be seen as the integrity of the data handled by the system, but also the system's logic and code itself [8, 55].

In the context of autonomic computing systems, with our abstract example of the autonomously working car, integrity is also a key aspect. As information is the key resource in our example, this information has to be accurate on one hand, but also unaltered. The autonomic system decides with the information at hand, and therefore could very well take wrong actions if the information given to it is wrong. An example of such behavior in our system would be altered map data. The car would try to reach the given destination with the available data. In the least problematic case, the destination would simply not be reached, as the car drives in a totally wrong direction. But this can quickly escalate into life threatening situations, even though the system decided "correctly" according to the given information. Another example would be a wrong decision based on altered information in the context of energy management, which could cost a large technology company easily millions of dollars. The same problem occurs if the system itself is altered.

How can integrity be achieved?
As it is a security property, just like confidentiality, the same reasoning applies

(see section 3.1, Confidentiality). We will therefore describe the same three dimensions in short, and refer to the arguments presented for confidentiality:

1. **Integrity of the user data on the system**
   The integrity of the user data on the system is going hand in hand with its confidentiality. If the proposed encryption mechanisms are used, any altering of the stored data becomes obvious. Therefore, the correctness of the user data can be guaranteed to a certain degree.
2. **Integrity of transmitted, especially received, data**
   The integrity regarding the transmission of data is key aspect, partly distinct to the description given on confidentiality. In this situation, mechanisms need to be applied which can ensure that the received data has not been tampered with. This is also not new to the field of autonomic computing, but gains an even higher priority, as this information is the key factor of such a system. In order to achieve this goal, the described concept of asymmetrical encryption can be used (e.g. RSA). In this case, the public key provides the means for digital signatures, as only the intended recipient can decrypt the message. This ensures that the data has not been altered and is therefore (most likely) correct. Therefore, an autonomic computing system would need to enable and support asymmetrically encrypted communication in order to possibly gain a user's trust.
3. **Integrity of the system itself, code integrity**
   The integrity of the system refers to the code and respective algorithms being protected against tampering and altering. In order to achieve this goal, the described mechanisms have been successfully implemented and used. Additionally, there are many current or past developments of mechanisms which can ensure or increase code integrity [54, 1, 43, 44].

Availability Availability describes a system's "capability of guaranteeing continuous access to data and resources by authorized clients" [2]. This can also be applied for the system itself, hence stating availability as the property of a system to be available (i.e. accessible and usable) upon demand. This should be true for any authorized client, and includes the system's ability to carry out any of the actions it is possibly able to perform, even in the case of a security breach or other unexpected events [55].

In the context of autonomic computing systems, with our abstract example of the autonomously working car, availability is also a key aspect. Especially in the current age of constant technological advancements and high-availability services, autonomic computing systems can certainly not afford to fall behind. In our example, it would be very important to 1) always keep the user informed about current actions (data availability), e.g. the calculated route, and 2) prevent and faults or malfunctions of the system (system availability). If an autonomous car would suddenly stop in the middle of a highway due to malfunction, the results are possibly far beyond horrific. On the other hand, if the car would not be available whenever a user sees the need to use it, he might refrain from using it entirely.

But how can availability be achieved?
There are two dimensions, which have to be addressed:

1. **Availability of the data on the system**
   The availability of the data refers to it being extractable or readable whenever needed. This is obviously highly dependent on the availability of the system itself (see next point). In order to guarantee the availability of the data, it is necessary to prevent any kind of corruption. This has partly already been described in the sections *Confidentiality* and *Integrity*. Additionally, it is necessary for a user to be able to access information on demand. This can be any kind of feedback, either provided by the system by default, or specially requested. This goes for any information in the context of the purpose of the system, but within the boundaries of confidentiality and integrity. Additionally, the interfaces have to be well defined. This goes for the user interface, which enables the user to monitor the system, as well as for interfaces regarding inside and outside connections and transactions of the system.
   It can be concluded that the trust of a user towards an autonomic computing system can exist, if the technical availability of the data can be ensured. This includes proper presentation through a user interface, but is not restricted to human-computer interaction. If one system in an network of autonomic computing system does fail to provide promised data, this might reflect on the other systems.
2. **Availability of the system itself**
   The availability of the system refers to it being usable or reachable whenever needed. This can be another system requesting a service, or a user wanting to use it. In order to increase the availability, we have to consider potential threats to it.
   The main threat can be located in actual hardware failures (e.g. in single devices), which force the system to cease activity. One possible approach is to harden the drivers against such device failure [27]. Many hardware problems can be handled through appropriate software, which prevents the complete system shutdown. But even then the system might not be fully usable anymore. Therefore weakened and soon-to-be problematic parts have to be detected, and appropriate actions, like contacting a maintenance service, have to be taken.
   Another threat can be seen in malicious software (malware) [40]. Malicious software has the goal to damage or perform unwanted actions on a system. Examples would be the disruption operations, gathering of sensitive information, or gaining access to the system itself. Therefore, we partly addressed this problem in the sections *Confidentiality* and *Integrity*. In general, the most effective way to prevent availability problems due to malware is to prevent its intrusion into the system. If this is not possible, a system should constantly monitor its own actions and memory in order to detect any malware signs and take elaborate countermeasures.
   A third threat are software failures. These can be summarized in five categories [40]:

(a) Resource exhaustion (e.g. Memory leaks)
(b) Computational/logic errors (e.g. corrupt pointers)
(c) System overload (e.g. synchronization errors)
(d) Recovery code (e.g. fault-recovery routines may fail)
(e) Failed upgrades or downtime during updates

In order to prevent such failures, it is necessary that the system is well constructed against them. Most of them can be foreseen and therefore be considered during the development. Obviously, the system has to be able to deal with such problems even during runtime by allocating resources appropriately and managing itself.
In general, it is necessary for autonomic computing systems to be available anytime. As they are supposed to run autonomously, it can be concluded that they are supposed to be available (e.g. to run, to be reachable, etc.) when needed and without any announcement. One general point of focus should be the reliance on other systems. The more independent an autonomic computing system is from other systems, the more availability can be guaranteed. In the context of our example car, the environment does play a crucial role as well. If the car would not work with the slightest change in environment (e.g. rain) which would not bother a human driver, this would have a highly negative impact on the trust of a user towards the system.

Conclusion In this section, we presented the security properties confidentiality, integrity, and availability (CIA). We illustrated the importance of them for establishing trust by describing their relevance in an autonomously driving car. Furthermore, we presented the respective dimensions, which have to be addressed in autonomic computing systems, as well as how they are currently handled in partly comparable systems.

We found that all three properties have an impact on the trust relationship, even though there might be differences depending on the system and the domain. Since there is no confirmation of a human necessary in order for the system to take action, there has to be a special focus on the security of the system and the information it autonomously handles. As autonomic computing systems supposedly possess the four self-* properties (namely self-configuring, self-healing, self-optimizing, self-protecting), besides other properties, the CIA properties should be fulfilled in any autonomous system, as they are included in the self-* properties. Comparing the results of this chapter to the definition given in chapter 2.1, we conclude that the named properties fall under the aspect of "The system does, what it should do, and does not, what it should not do".

3.2 Psychological Factors

In this section, we are going to provide an overview over some psychology-based factors with an impact on the trust relationship between autonomic computing systems and their users. We picked these factors in order to show how important the devotion towards the human user is. The technological factors may be

perfectly satisfied, yet every human might evaluate the system in question differently. Therefore, these factors have to be addressed. Unlike the technological factors, the fulfillment of the psychological factors is not completely obligatory in order for trust to exist. Every human being does have other mental barriers which have to be overcome in order for trust to develop. Therefore, trust is not a property of the system, but a state which the system has in the mind of every person in contact with it.

This section provides definitions for the addressed psychological factors. Each of them will be presented in the context of autonomic computing systems with the focus on the impact on trust. Just as with the technological factors, we will illustrate their importance with the help of our introduced example. We will conclude the discussion of each factor by providing approaches of how to handle them, and by giving leads as to how to possibly achieve the set goal of building or improving the trust relationship. We note that some of these factors are closely related to the trust in automation in general and have therefore already been discussed in that context by researchers.

Experience Experience, which includes factors like expectation and familiarity, can be viewed as a crucial factor in every interaction between humans as well as human-machine interaction. If we have successfully worked with somebody else in the past, we are more inclined to do so again in the future. The same is true with the usage of systems which perform tasks automatically or even autonomously. When, at the beginning, humans were reserved regarding new automation systems, their trust steadily increased with the growing experience and satisfying results. Of course, while our own past experience does play a crucial role, we also consider experiences of others [33], especially when they are close to us in a social context. While we might not directly buy an article just because some other user wrote a positive review about it, a comparable report of a member of our family might influence us on a bigger scale. Of course, also strangers who write reviews online might impact our decision more if they have a good reputation (e.g. many good reviews written in the past, high rating by other users) [13]. Carlson et al. [9] have shown that the factor experience (*Your past experience with the car*) does score in the top third of most important factors that influence trust in automated systems in the automotive domain. Even though this could slightly differ for autonomous systems, and therefore autonomic computing systems as well, we believe experience to be very important nevertheless. However, they have also shown that experience is domain specific. In the medical domain, experience does rank in the bottom of the middle third.

In the context of autonomic computing systems, especially our autonomous car, experience is also of utmost importance. While it is obvious that one might continue to use a system if the past usages had a positive outcome, the crucial part lies in the persuasion of "first-time users". Why would an elderly person use an autonomous vehicle if a driver's license as well as the car itself are available? As we tend to stick to what we know and do not like to take risks (see section

3.2), a person is unlikely to change behavior unless there is a pressing issue at hand.

But experience does not only have an impact on the usage of the same system. Past interactions with comparable systems, be it technologically or functionally, also influence the amount of trust we have. A teenager nowadays who grew up with touch-based systems would faster adopt to using a new system of this sort than an elderly who had no prior experiences with them. We note that a positive experience is highly linked to the *predictability* of the system's actions [18]. If the system does not respond deterministically, or even worse, acts in a wrong or unexpected way, one can hardly build any trust.

To tackle this problem with the introduction of autonomic computing systems, it is very important to allow potential users to grow accustomed to them. One would obviously not right off the bat jump into an autonomous vehicle and relax while it drives on a highway. However, if people have the chance to gain experience with the system, and therefore catch a glimpse of the possible improvements and gains, this problem might be solvable.

Another way is to take the route over autonomic or semi-autonomous systems in a comparable domain. For example, there already exist vehicles which offer autonomic or even semi-autonomous driving assistance systems [6] and therefore already allow drivers (and potential users of autonomously driving cars) to gain some experience with the technology. If these vehicles are able to keep on driving close to without any accidents, and allow their users to see and make use of the benefits they possibly offer, autonomously driving cars might have a better standing in the future.

Feedback and Own Control As we already stated in section 2.2, a user should be constantly provided with feedback about the system's actions and the possibility to interfere with its actions at any time. This is going to be a necessity for trust to be established in autonomic computing systems. If these systems were to be a black box which only presents us final results, but no way to either investigate the interior of it or gain some information about ongoing processes, one would have a hard time to establish trust. Therefore, it is of utmost importance to present the user on one hand useful information and on the other hand also the information he might demand. Additionally, the possibility to intercept the system it necessary in order to provide a feeling of own control [18]. In the worst case, the system runs berserk and the user would just be able to sit by idly and watch how his life is possibly endangered. Such a vision is certainly enough for many people to turn tail on the usage of such a system.

In the context of autonomic computing systems, especially our autonomous car, the provided statements still hold strong. For example, the user might want to be sure that the car is actually driving towards the correct destination. This could easily be achieved by constantly showing the current route on a monitor, just like navigation devices would do. In the case of a route change, due to the car obtaining data about a traffic jam, the user would probably feel at unease if the car were to just change directions without informing him of the reason. This

list can be extended to pretty much every decision of the car, be it the change of speed, or the reason for opening the windows. Experiments have shown that non-communicated braking of an autonomous car can have significant negative consequences for the driver [31]. But with the introduction of voice warnings, it was also shown that anxiety can be alleviated, alertness can be increased, and the user can gain a sense of control back due to being informed about the current vehicle status.

A central challenge in that context is the choice of data and the method of presentation. As it is impossible to keep the user up to date with every piece of data the car obtains, creates, or handles, the focus has to be on the most important pieces. Additionally, there will have to be taken special measures as to how to inform the user about any unexpected events or system failures which require him to take action. If the passenger of an autonomous car were to read a book while the vehicle attempts to transport him to his destination, a simple red warning symbol on a display would certainly not be enough to catch his attention.

As the choice of data might be highly dependent on the current situation and status of the vehicle, we will not further elaborate on the process of choosing correct data. However, even beyond default feedback, the user should be able to access data which is relevant to him, but not displayed by default. However, as for how to face the problem of providing feedback appropriately, one path has already been touched upon previously. It can be of immense help to provide feedback over various output channels [37]. While the default channel would be the visual representation of information, other channels like auditive, haptic, or a combination of them (multimedia), can be used to generate the desired effect. However, when to use multiple channels, has to be chosen wisely. On one hand, a constant overflow of information does not help to generate trust. On the other hand, the positive effects of using multiple channels are highly dependent on the task, and redundant feedback can even be harmful [35].

In order for a human to interfere with the system, interfaces which allow for that are necessary. As for autonomously driving car, instruments like a steering wheel or a gas pedal should still exist and be usable. However, as it is not necessarily a given that an autonomous vehicle has a human aboard, there should also be possibilities for remote access. We assume that with a growing number of fully autonomous vehicles, also a respective number of monitoring personnel exists, which oversee the traffic and take control of certain vehicles if required.

Risk and Public Information Whenever we perform an action, we consider and weigh, mostly subconsciously, possible gains and risks. The same applies for the usage of an autonomic computing system. Before a user does initially, for example, trust his life to such a system, a considerable amount of thought has gone into this decision. The higher the risk involved is, the higher the amount of trust is needed in order to actually rely on the system. However, risk does only include what would happen in the worst case, but also the probability of this event occurring. While it is an unchangeable fact that a user would is some

way need to entrust an autonomously driving car with his life, the risk can be lowered by reducing the probability of failure (e.g. prevent accidents).

Depending on the situation of the individual, the risks can differ even with regard to the same system. For example, the creator of such a system does have his reputation, job security and the finances of his employer on the line. The final end user of the system however, puts his personal safety, finances, and property on the line [32]. Therefore, the user's relationship with the system, be it as active profiteer of it, or just through passive interaction (e.g. other road users which use a non-autonomous vehicle but drive next to an autonomous one), has to be considered while addressing the risk factor. It has been proven that the subjective probability for an even does often not correlate with the objective probability. For example, humans tend to overestimate the chance of rare events (e.g. winning the lottery), while also underestimating the objective probability in general [28]. Our decisions are therefore sometimes not rationally sound, but are subject to various disturbances, both psychologically and environmentally. Additionally, humans tend to value possible losses higher than comparable gains, which is in research addressed as *risk aversion*. In general, it states that an individual usually favors choices with less risk (i.e. less possible losses) over the chance to gain more, but also accept a higher risk respectively.

The main question is how the probability of negative events occurring (e.g. system failure, road accident) can be reduced. As already stated, it is not sufficient to minimize the objective (i.e. actual) probability, but it is also necessary to address the subjective probability which is unique to every individual. It has to be noted that risk actually involves all the technological and psychological factors in some way. The objective risk is mostly evaluated in accordance to the technological side of the system. The subjective risk is evaluated considering the technological factors, but also with the influence of past experiences with the system, education, and many more.

To address the subjective risk, it is of utmost importance to actually prove a low chance for a negative event to occur. With respect to our autonomous car, this could involve providing statistics, which show that these vehicles are less accident-prone than vehicles driven by humans. Carlson et al. have also shown that there are many factors which are considered by humans in order to develop trust in an autonomous system in the automotive domain [9]. With regard to risk, main influence factors are *statistics of the car's past performance*, *Extent of research on the car's reliability*, and the *existence of error indicators*. They considered many more factors, which partly have also an impact on the perceived risk of an individual, but these will not be addressed in this section.

In general, it is necessary to provide information and statistics in order to prove the system's reliability and trustworthiness. This includes a comprehensive amount of test data and transparency of the process of the creation and the test of the system. Furthermore, it is also shown that the trust in the system can be influenced by it being produced by a brand versus a non-brand [9]. If public figures and authorities were to accept such a system and promote it, the end-

user's trust would probably also be affected. The same goes for the government which can aid the spread and usage of such systems by providing a framework, for example through foundation in the law or credible certifications. Last but not least, as every decision is influenced by weighing possible gains versus losses, there exists also the path of fundamentally increasing the gains. If autonomic computing systems were to drastically improve the aspects relevant for society as a whole, work, or even a single individual, they might be worth it to take the risk of uncertainty at the beginning.

Conclusion In this section, we presented the psychological factors *experience*, *feedback and own control*, as well as *risk and public information*. We illustrated the importance of them for establishing trust by describing their relevance in an autonomously driving car. Furthermore, we presented possible basic approaches which could be used to address these factors.

We found that all of these factors are of utmost importance for trust to be established. While they are not mandatory to be fulfilled in order for trust to exist, their impact is not to be neglected. As we already stated, the necessity of certain factors being satisfied varies for each individual and the system in question. However, taking the whole field of possible adoptions for autonomic computing systems into account, we assume that all these factors will have to be addressed in order to be able to reap all possible benefits from these systems in the future.

It has to be noted that typical problems which exist with automation like *automation complacency* (the system is not monitored and controlled appropriately), *automation bias* (the system's actions and outputs are seen as impeccable), and the *cry-wolf effect* (insufficient trust in the system), have their own role in the field of autonomic computing systems.

Automation complacency is in the context of our autonomous car partly a given. First of all, it is not possible to monitor every action appropriately. Furthermore, the goal of autonomous vehicles should be to have them drive even without human passenger. As we already stated, in such a case a monitoring institution would likely be in charge. However, this institution would not be able to monitor every action the car performs.

Automation bias has to exist in order for the concept of autonomic computing systems to bear fruit. As one of the traits of these systems would be the analysis and evaluation of humongous amounts of data, it would be extremely harmful to the efficiency of the systems if their users were to doubt them needlessly. Of course, if we were to allow automation bias to unfold altogether, we might face irreversible damage. If the user of an autonomous vehicle sees his car heading straight for a cliff, he might want to question the systems action. On the other hand, if humans were to stop a system constantly and manually follow the decision process in order to determine if the system's actions were correct, the benefits of using the system would be marginal. Therefore, we need some overreaching trust (automation bias), but not in an unlimited amount.

Lastly, the cry-wolf effect is counterproductive to the existence of autonomic computing systems. If the user would not trust the system, reaping its benefits would hardly be possible.

3.3 Foundation in Law and Society

The third key aspect is the foundation in law and society. If there is no legal framework for the usage of autonomic computing systems, users are hardly going to trust them if they are allowed to use them at all. Besides answering questions like who is responsible for system failures or accidents cause, society as a whole is also obliged to dispute about the acceptance of such systems. Besides laws which - for instance - actually permit the operation of autonomous vehicles on roads as the state of Nevada did in 2011 [30], it is also necessary to have groups which follow the leads of consumer protection. It is necessary to constantly weigh the upsides and downsides of these systems, and also to install an independent institution to monitor the companies and the autonomic computing systems they produce. Additionally, it is highly advised to install certifications and review processes in order to certify and guarantee the proper functionality of the system and prove this to the public.

For further research, we refer to the work of Hildebrandt and Rouvroy, who addressed the topic of how law and technology can be brought into accordance with each other while especially considering the legal factors of autonomic computing [23].

4 Current Development - Google Self-Driving Car Project

This section will provide information about an example for an autonomic computing system, namely the *Google Self-Driving Car Project*. This autonomous vehicle (*Google car*) is one of the most prominent representatives of currently developed autonomic systems. We will give a brief introduction on the technical aspect of the vehicle, as well as some further background information on the project itself. Additionally, we will depict how the factors described in section 3 are dealt with, or are to be dealt with in the future. We would like to note that Google is not the only company researching and investing in autonomous vehicles. An incomplete list includes the likes of General Motors (GM), Mercedes Benz, and BMW [41].

4.1 Background

Google has made its involvement with autonomous cars public back in 2010 [47]. At that point in time, the *Google Self-Driving Car Project* had already been in development for some years and Google could therefore already show a prototype. They wanted to tackle two major challenges in the automobile industry: Safety and efficiency. Their main goal is to prevent traffic accidents as

well as possible, free up the time of people which they spend driving, and reduce carbon emissions.

Their approach to reach these goals, namely the development of an autonomously driving car, has proven to be a huge project with estimations of the completion at up to thirty years from now [19]. One year after the initial revelation, Google also promised that the car would be able to "drive anywhere a car can legally drive" [19]. The world is tensely waiting for any news on the development of autonomous vehicles, which goes to show that the direction taken is of big interest in general.

The Google car itself does by now not differ as significantly as one could expect from currently developed cars with automatic driving assistance systems. It is equipped with video cameras, a variety of sensors, and most importantly, a laser range finder. The car integrates hardware and software alike to allow for correct perception of the surroundings, as well as to perform the required calculations to finally decide on the course of action [41]. The laser range finder, which is mounted on top of the car, allows for real-time environment analysis and is therefore one of the most important parts. The Google car uses the hardware and software to generate a three dimensional map of the environment, which is the foundation for the car's decisions and actions. It recognizes a variety of possible obstacles and objects and reacts accordingly, while always adjusting for local traffic laws and aiming to never break them. Even real-time adjustments are possible in order to take other driver's decisions and actions into account. Furthermore, the car is connected to its environment through GPS and the internet. Figure 3 shows the Google car and its main components.

Even though the project started before 2010, it is still in a stage of testing and adjusting. By now, there are constantly test cars driving on the streets in the USA, both manually and autonomously. However, the car does always have a driver, even if it drives autonomously. These test drivers monitor the car's actions and provide feedback to the engineers on the behavior shown. They go through "rigorous training" in order to prepare them for any eventual situation that may arise, including extreme and complicated driving situations. Unlike an average driver, they also know intimately how the car works. If the situation calls for it, the test drivers may take over the car manually at any time.

According to the official webpage of the project[1], the Google car drives by constantly answering the following four questions:

1. Where am I?
2. What is around me?
3. What will happen next?
4. What should I do?

As of the end of July 2016, the Google cars have in total driven 1.199.427 miles (1.930.291 kilometers) in manual mode, and 1.842.496 miles (2.965.210 kilometers) in the autonomous mode since the start of the project's test drives

[1] `https://www.google.com/selfdrivingcar/how/`

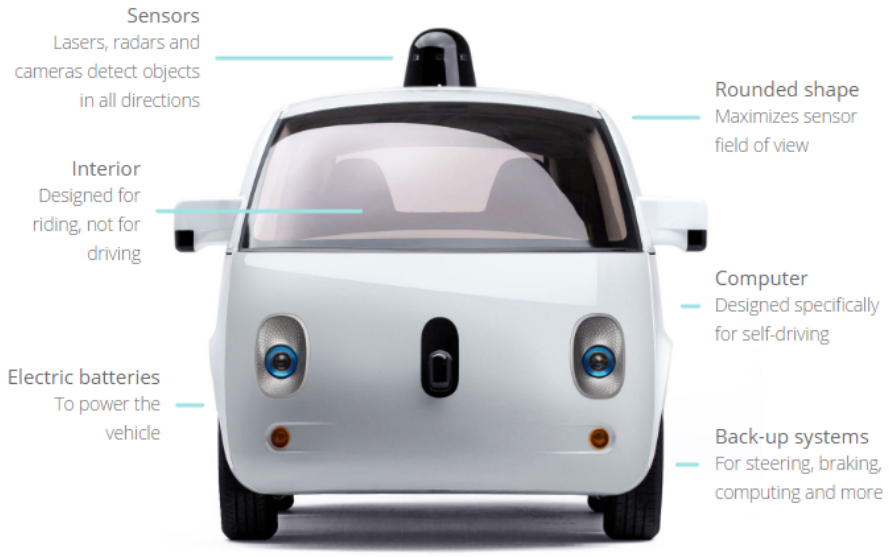

Fig. 3. The Google car with its main components shown from the front, as presented on the official project page.

in 2009 [20]. There are currently 58 vehicles on public roads, while some tests are also conducted on closed roads.

4.2 Google Car and the Technical Factors

As we already stated in section 3.1, security in form of the three main factors confidentiality, integrity, and availability (CIA) is a crucial aspect to the trust relationship between autonomic computing systems and their users. Therefore, the Google car cannot allow to neglect them, which is all the more true the closer the project is to the commercial release. However, research shows there is still a lack of attention on the topic of security for autonomous vehicles. Most approaches simply focus on encrypting the data, while possibly neglecting the special circumstances which undoubtedly come with this partially new field [51]. There are many known threats to the security of cars which are currently out on the roads, and these therefore possibly apply to the Google car as well [52]. However, as Google has the opportunity to create the car practically from scratch. Therefore, everything related to the internal network and infrastructure of the car itself is potentially free from all current implications which we can see in cars. However, as the car still uses interfaces to communicate with the outside world, it has to abide by the respective rules. We conclude that this is still an area which will see many changes before the project is finished, and we will therefore leave it for future research.

4.3 Google Car and the Psychological Factors

In order for a positive trust relationship between autonomic computing systems and their users to exist, the psychological aspects as stated in section 3.2 have to be addressed as well. If one fears the unknown, why would he, despite existing fear, decide to trust the Google car and let it drive him? While this question may not arise with the test drivers, it will eventually become a crucial aspect when the commercial release is planned. However, even at this point in time the appearance of the Google car impacts other road users, as it is quite conspicuous visually. In this section, we will briefly depict how the Google car tackles the factors *experience*, *feedback and own control*, as well as *risk and public information.*

Google Car - Experience As we already stated, experience is of utmost importance to humans for deciding whether to trust a system or not. However, the Google car does not allow public test drivers to participate in the project. As a result, your average person and potential user is not able to gain experience with the system. The other way, even though it is not directly related to Google and therefore actually not part of this section, is to let users grow accustomed to automatic driving assistance systems. These are already widely used today, and in some cases close to working autonomously. With this path, which seems to be the way of the world as of now, there is a possibility that the experience factor does not become a problem, as the transition from highly automatic to autonomous is smooth.

Google Car - Feedback and Own Control As the Google car is still far from being commercialized, it is hard to tell which forms of feedback will be provided to the user, and whether or not manual control will finally be possible. However, Google is already planning to develop their autonomous cars without ordinary tools for human intervention like a steering wheel or a accelerator pedal [50]. As of now, their cars still have these tools, which is also partly due to the law of the states their tests are conducted in [45]. Another reason for this is the fact that their test drivers are still supposed to take over if desired or necessary. However, Google fully intends to remove these if the law changes accordingly and their technology is advanced enough to deem human intervention unnecessary [21]. We believe that this step would be highly questionable for building up a trust relationship. Then again, it has also been shown that an intervention of the user is less likely, if the vehicle does not have traditional control elements like a steering wheel, but instead only a red stop-button [42]. This also goes in accordance with the fact, that especially with regard to autonomic computing systems, any manual intervention would cause more risk and complexity, and is therefore undesired.

We conclude that removing the possibility to manually control the vehicle might prove to be a problem for establishing trust. On the other hand, if a brand like Google takes such a leap, it can be seen as a huge statement for the confidence they have in their product. And as Carlson et al. have shown in their

study [9], people tend to have considerably more trust just because a product was developed by a brand like Google. This does prove that reputation of a company or their product is a crucial aspect to building trust, which can be seen part of one's experience and publicly available information.

Google Car - Risk and public information Google makes great effort to reduce the risk of its autonomous car for its users and other road users by conducting countless tests and employing top engineers in their project. While smaller accidents still occur, the project is continuously advancing towards on one hand preventing human errors by not letting them make mistakes, and on the other hand detecting and eliminating flaws in their software and hardware.

As for public information, Google also tries to promote their successes and statistics to the public in the form of monthly reports [22]. On their project page they also provide various information about the car, how it works, and the benefits of their dream of driverless vehicles. It has to be noted that Google does still not really advertise the car and and therefore only promotes it in a way to satisfy public interest. As the initial steps for commercialization are planned for 2020 [16], their is yet no need to provide too much information and possibly giving leads for malicious players for future attacks. However, it is predictable that Google will reduce the risk considerably before introducing their car to the market. In order to promote their success in doing so, we will most likely also see more information and statistics published. As long as during the coming years no major accidents and human deaths happen with a direct involvement of the car, it can be expected that a commercial user is very likely to develop trust in the vehicle due to the very reduced risk and a variety of publicly available information about the car's functionality and statistics.

5 What awaits us in 2025...

In this section we will examine possible benefits of autonomic computing system in the year 2025. We will depict the impact of these systems on society as whole, work, and people. We consider the two main possiblities, which we expect to possibly unfold until 2025: 1) Mankind is *not* able to overcome current or future hurdles regarding trust, which prevents an optimal usage of autonomic computing systems and results in missing out on possible benefits which come with the systems (section 5.1). 2) Mankind is (mostly) able to overcome current trust issues and use autonomic computing systems to their full potential (section 5.2).

In order to depict our train of thought, we use the abstract scale seen in figure 4. We use three main interaction possibilities with the system which represent the amount of trust a user has in it, as well as to which amount he can profit from the benefits the system can possibly provide. Red symbolizes that the system is not trusted, and therefore not used. Yellow is representative for partial trust, which allows for usage, but the user remains highly doubtful or cautious, trying to monitor the system too closely and possibly even restricting its actions. Green represents the area in which the user fully trusts the system and is also able to

fully reap the benefits it offers. The upper half shows a possible scale if many trust issues remain. While already a considerable amount of trust is needed to work with the system at all (i.e. leave the red area), an even larger amount is needed in order to leave the yellow area (partial trust) and therefore fully utilize the system. The lower half of the figure shows the best possible case of mostly resolving the current trust issues. While still a noticeable amount of trust is needed to leave the red area (no trust in the system), the yellow area has shrunk considerably and the green area has increased respectively.

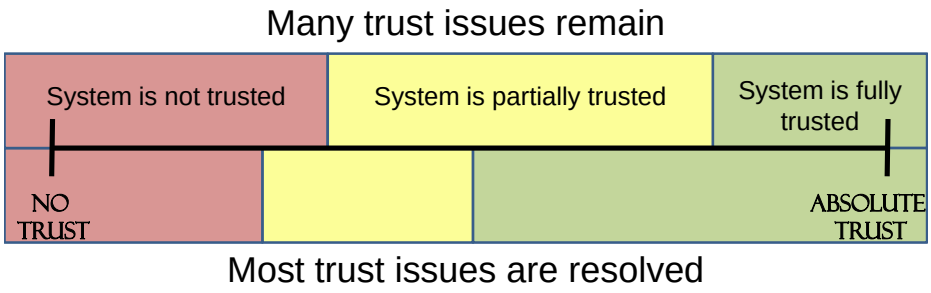

Fig. 4. An abstract scale showing how much trust is needed for a user to trust a system to which amount. The upper half represents a scale if many trust issues remain. The lower half represents a scale which could exist if most of the trust issues are resolved by 2025.

5.1 If the Trust Issues Can be Partially Resolved

Considering the relatively small timeframe of 9 years from now, it is more likely that the trust issues can either be not resolved at all, or only be partially resolved. While with the increased usage of automation, be it in the automobile industry, in the health industry, or in the finance sector, people already develop trust in said systems. As they become more and more a part of our everyday life, their spread and usage will only increase. Once a user has understood how much such a system can help him, he does not want to miss it anymore. However, the gap from automatic to autonomic computing systems is still large.

Impact on Society Even though autonomic computing systems might hold the power to change society in a way the world wide web or smartphones did, it is unlikely for them to gain enough prominence by 2025 to achieve the same. This is all the more true if trust issues still remain. While certain parts of society might indulge in autonomic computing systems, most groups are not ready yet to adapt

to changes on such a scale, even in our constantly changing world. We expect technology by 2025 to advance enough to potentially allow the introduction of the first autonomic computing systems in major fields which affect a society as a whole. Examples would be public transport, cars and comparable vehicles, healthcare, and so on. However, by that point in time the main question will still be whether we are ready for such systems or not. There will still be need of a legal framework in many cases, and society as whole still has to decide whether or not to accept such systems in situations where lives or enormous financial values are at stake.

Impact on Work The impact on work might be considerably high even if trust is still an issue by 2025. As corporations are mostly profit-oriented, they might consider the usage of autonomic computing systems even before society as a whole accepted them. However, as trust issues still remain, we expect the deciding positions in a company to introduce them incrementally. This means that a longer period of test runs await, while the systems are only run with a low risk - small reward attitude. While this may lay the foundation for further spread of these systems, we expect the impact to be relatively small compared to the potential autonomic computing systems hold. However, once the first competitor is able to transfer complex or expensive tasks to these systems, and is therefore able to save money and resources, many might follow their lead.

Impact on People The impact on people might be the most interesting aspect by 2025. While many are still overwhelmed by the current technological advancements, autonomic computing systems will reach yet again a new stage in that aspect. While the usage of a smartphone is mostly a matter of growing accustomed to it, trusting an autonomous system is a far more essential decision. While the trust needed to use such a system to full potential might in this case not be a given, people will still be confronted with the essential question on how to think about and possibly interact with them.

The gains versus risk debate will become an important aspect, as many autonomous systems will be in development or close to be released by 2025. Therefore, the possible benefits will be promoted as well, possibly shaking the very mindset of somebody towards these systems as they understand what they might miss out on. Some people might very well be left behind, just like some were with the introduction of personal computers or smartphones.

Furthermore, interactions which are not face to face will more often be doubted. As autonomic computing systems might very well be used for customer interactions, said customers are more inclined to doubt whether the communication partner is actually human or not.

Lastly, we believe that as long as the trust issues remain, every crucial failure of an autonomic computing system will have tremendous impact on whether an individual is going to trust them in the future or not. Even if the overall results by these systems are better than anything humans could ever achieve, single mistakes can ruin all the reputation which was built up. There will always be

the question of whether a human would have made the same mistake, even if the system has beforehand avoided crucial mistakes which a human would have made.

5.2 If the Trust Issues Can be Resolved

The best possible case would undoubtedly be to fully trust autonomic computing systems, if their technical ability allows for better results than humans could provide. With the ongoing trend, to which the development of autonomous vehicles as described in section 4 belongs, it is just a matter of time until more and more tasks are handled by autonomic computing systems. Even though we do not think that people will be able to fully utilize autonomic computing systems by 2025, we will still give a short overview on what benefits we can expect if the trust issues can be resolved.

Impact on Society In the event that society is able to accept and trust autonomic computing systems as a whole, many possible new paths open up. Next to the introduction and usage of autonomous cars as we described, many other areas could profit as well. The workload can be better balanced according to the strengths of humans and of autonomic computing systems. Besides autonomous cars and other road vehicles, public transport by trains, trams or buses could also see changes in an autonomous direction.

The benefits of autonomous cars are for society are already assumable. The U.S. Department of Transportation reported that 94% of traffic accidents are due to driver (i.e. human) errors [48]. After all, 33.000 people die every year on America's roads, even though 55% of all crashes only result in minor damages and no injuries [49]. Unfortunately, these crashes are not even reported, making it harder to understand these accidents. However, as we stated in section 4, Google has already amassed a considerable amount of kilometers driven with their cars. And the result, as of 2015, was that their cars were only involved in 11 minor accidents with light damage and no injuries, while the Google car was not once the cause of the accident [49]. So we see a great opportunity to make roads a safer place, while reducing the amount of deaths, injuries, and even material damage and the respective costs. On the other hand, the car would permit people who are unable to drive to more mobility, increasing the overall benefit for society as well.

Similar effects are expected in other application areas of autonomic computing systems. Of course, this assumes that they are able to perform better than a human at the task, but as they are designed to exactly do that, this can be seen as a given. Services would be faster and of better quality, it would be possible to provide more comfort due to the user not always needing to oversee the system, and it would be possible to adjust better to human needs. Additionally, it would be possible to tackle new arising challenges, as some complex tasks are still limited to human capabilities and boundaries.

Impact on Work Autonomic computing systems will likely also have an impact on work. While by 2025, this impact will be relatively small compared to the potential which exists, but we will try to depict possible scenarios. First of all, these system could be used to execute tasks which are boring to humans and unnecessarily cost intensive. This could be the scraping of data, and making decisions according to the results. Autonomic computing systems could even be used to lead a group of people by deciding on the next task to be handled in a project or by assigning tasks according to current situation of a worker. Furthermore, whole interactions between companies could be handled autonomously. With huge sums of money involved, and also huge amounts of data, it would be highly beneficial to eliminate the human as a source of error. However, we do not see this happen by the timeframe we regard in this work.

Overall, jobs can be done faster, with better results, more efficiently, and even generate more revenue for the companies due to less resources needed. On a downside, we see the possibility of many low-income sectors using these systems to replace their workers and therefore eliminate their workplace. As we did show in this work, it is in the realm of possible events to have autonomous cars by 2025. This would also include trucks used for the transportation of goods, and therefore these systems would effectively replace the driver. Many other sectors, in which either the human is a hindrance due to his introduction of errors, or the human traits are not necessary to solve the task at hand, could follow the lead and replace the human workers.

Impact on People Lastly, each of us will also be affected by autonomic computing systems. Even if we are not using them, chances are that they will exist somewhere in our environment, be it the autonomous car in the next lane, or the system which analyzes the own job application for relevant information. As the Google car already aims at providing the benefits of more security and a better spending of time, it is only natural to assume that this would be possible by 2025. Therefore, just an autonomous car could provide more comfort, an actual gain of lifetime by freeing the driver from driving, and more security.

In general, humans need to handle less tedious tasks, which would also result in a better feeling overall. This would also allow individuals to focus their resources on what they are good at, rather than what they need to do due to their circumstances. However, by 2025 we also expect that there will be a critical debate and grappling with about the usage of autonomic computing systems, which will affect everybody in one way or another.

As a downside, we will most likely rely on these systems (too much?), which might lead to false security. If the autonomous vehicles would still need human intervention in some cases, the driver would possibly focus his attention on reading a book, as he assumes that the rare event of him needing to intervene does not occur.

6 Conclusion

In this work, we gave an overview about some of the most important factors with an impact on the trust relationship between autonomic computing systems. We started by outlining the importance of trust (section 2.1) and by providing a definition of trust for human-computer trust (section 2.2). From that definition, we derived a definition of trust in the context of this work, which was from that point on used. In section 3, we specially focused on the relationship between autonomic computing systems and their users. We chose the factors confidentiality, integrity, and availability as technical factors, which are important for the trust relationship (section 3.1). In section 3.2, we chose the factors *experience*, *feedback and own control*, as well as *risk and public information*. Each of the named factors was thoroughly defined, explained and with examples depicted. As a representative of current developments, we chose the *Google car* (section 4), which we described in general. Furthermore, we did show how Google intends to deal with the technical and psychological factors. Finally, we gave an outlook on what impact of autonomic computing systems we can expect on society, work, and people in 2025 (section 5).

We see that with the increase of automation and especially with current efforts of global players like Google with autonomic computing systems, more and more such systems will be developed and used. Therefore, it is a central challenge to deal with the one who has to interact with the system and handle all the upsides, as well as the downsides - the human. Trust in the system is essential for it to work without any unnecessary disruption by the user, if he considers to use it at all. We see the need for technology to be advanced enough to be able to actually guarantee, e.g., confidentiality, integrity, and availability, if the user does not have a away to actually decide what happens once a "start button" was pressed. But as that alone is not enough, there is still a considerable amount of work necessary to convince people (i.e. to help them to develop trust) to use the system, and use it to the full potential.

Even though Google does intend to remove the possibility for humans to interfere with the car's action, we still see a strong need for such a fallback. It is hard enough to trust something a user cannot possibly understand in total, but it might be too far-fetched to rob the user of the last resort, and a way to exert actually any control. By 2025 we see some changes and further autonomic computing systems being developed or finished, but we consider most trust issues to still remain. As we did show in the upper half of figure 4, we expect that users in general will be able to trust autonomic computing systems to an extent, but unfortunately not enough to use them to their full potential and reap the respective benefits.

The way to convince somebody to trust an autonomous vehicle would be to first guarantee the functionality of the technical aspects and reducing the risk involved. This goes hand in hand with providing the respective information of technical attributes and test conducted to the public. And before an autonomous vehicle is used, users may also grow accustomed to leaving some work to a system by using automatic driving assistance systems. Finally, a legal framework, as

well as support by the public is necessary in order to influence the position an individual has on the depicted trust scale. However, the most crucial time will probably be in the next 10 years. As these systems are on the rise right now, and pioneers like the Google car are expected to hit the market during that time, every decision made by the companies, and any negative or positive headline will hugely influence the trust towards the upcoming systems.

References

1. Andriesse, D., Bos, H., Slowinska, A.: Parallax: Implicit code integrity verification using return-oriented programming. In: 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks. pp. 125–135. IEEE (2015)
2. Anisetti, M., Ardagna, C.A., Damiani, E., Saonara, F.: A test-based security certification scheme for web services. ACM Transactions on the Web (TWEB) 7(2), 5 (2013)
3. Armoogum, S., Caully, A.: Obfuscation techniques for mobile agent code confidentiality. Journal of Information & Systems Management 1(1), 83–94 (2011)
4. Atoyan, H., Duquet, J.R., Robert, J.M.: Trust in new decision aid systems. In: Proceedings of the 18th Conference on l'Interaction Homme-Machine. pp. 115–122. ACM (2006)
5. Avizienis, A., Laprie, J.C., Randell, B., Landwehr, C.: Basic concepts and taxonomy of dependable and secure computing. IEEE transactions on dependable and secure computing 1(1), 11–33 (2004)
6. Bajpai, J.N.: Emerging vehicle technologies & the search for urban mobility solutions. Urban, Planning and Transport Research 4(1), 83–100 (2016)
7. Callegati, F., Cerroni, W., Ramilli, M.: Man-in-the-middle attack to the https protocol. IEEE Security and Privacy 7(1), 78–81 (2009)
8. Cappaert, J., Kisserli, N., Schellekens, D., Preneel, B.: Self-encrypting code to protect against analysis and tampering. In: 1st Benelux Workshop Inf. Syst. Security (2006)
9. Carlson, M.S., Desai, M., Drury, J.L., Kwak, H., Yanco, H.A.: Identifying factors that influence trust in automated cars and medical diagnosis systems. In: AAAI Symposium on The Intersection of Robust Intelligence and Trust in Autonomous Systems. pp. 20–27 (2014)
10. Casey, E., Stellatos, G.J.: The impact of full disk encryption on digital forensics. ACM SIGOPS Operating Systems Review 42(3), 93–98 (2008)
11. Cetina, C., Giner, P., Fons, J., Pelechano, V.: Autonomic computing through reuse of variability models at runtime: The case of smart homes. Computer 42(10), 37–43 (2009)
12. Chen, D., Zhao, H.: Data security and privacy protection issues in cloud computing. In: Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on. vol. 1, pp. 647–651. IEEE (2012)
13. Chen, P.Y., Dhanasobhon, S., Smith, M.D.: All reviews are not created equal: The disaggregate impact of reviews and reviewers at amazon. com. Com (May 2008) (2008)
14. Cheshire, C.: Online trust, trustworthiness, or assurance? Daedalus 140(4), 49–58 (2011)
15. Costante, E., Hartog, J., Petković, M.: Understanding perceived trust to reduce regret. Computational Intelligence 31(2), 327–347 (2015)

16. Daziano, R.A., Leard, B.: Are consumers willing to pay for letting the car drive for them? analyzing response to autonomous navigation. In: IATBR 2015-WINDSOR (2015)
17. Durumeric, Z., Kasten, J., Bailey, M., Halderman, J.A.: Analysis of the https certificate ecosystem. In: Proceedings of the 2013 conference on Internet measurement conference. pp. 291–304. ACM (2013)
18. Franke, J.L., Zaychik, V., Spura, T.M., Alves, E.E.: Inverting the operator/vehicle ratio: Approaches to next generation uav command and control. Proceedings of AUVSI Unmanned Systems North America 2005 (2005)
19. Gomes, L.: When will google's self-driving car really be ready? it depends on where you live and what you mean by" ready"[news]. IEEE Spectrum 53(5), 13–14 (2016)
20. Google: Google self-driving car project monthly report (2016, Last accessed: 20th August 2016), https://static.googleusercontent.com/media/www.google.com/de//selfdrivingcar/files/reports/report-0716.pdf
21. Google: Faq (Last accessed: 24th August 2016), https://www.google.com/selfdrivingcar/faq/
22. Google: Monthly reports (Last accessed: 24th August 2016), https://www.google.com/selfdrivingcar/reports/
23. Hildebrandt, M., Rouvroy, A.: Law, human agency and autonomic computing: The philosophy of law meets the philosophy of technology. Routledge (2011)
24. Horn, P.: Autonomic computing: Ibm\'s perspective on the state of information technology (2001)
25. Huebscher, M.C., McCann, J.A.: A survey of autonomic computing-degrees, models, and applications. ACM Computing Surveys (CSUR) 40(3), 7 (2008)
26. Jeffrey, O., David, M.: The vision of autonomic computing
27. Kadav, A., Renzelmann, M.J., Swift, M.M.: Tolerating hardware device failures in software. In: Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles. pp. 59–72. ACM (2009)
28. Kahneman, D., Tversky, A.: Choices, values, and frames. American psychologist 39(4), 341 (1984)
29. Khalil, I.M., Khreishah, A., Azeem, M.: Cloud computing security: a survey. Computers 3(1), 1–35 (2014)
30. Knapp, A.: Nevada passes law authorizing driverless cars. Forbes, June 25 (2011)
31. Koo, J., Shin, D., Steinert, M., Leifer, L.: Understanding driver responses to voice alerts of autonomous car operations. International Journal of Vehicle Design 70(4), 377–392 (2016)
32. Lacher, A., Grabowski, R., Cook, S.: Autonomy, trust, and transportation. In: 2014 AAAI Spring Symposium Series (2014)
33. Lee, J.D., See, K.A.: Trust in automation: Designing for appropriate reliance. Human Factors: The Journal of the Human Factors and Ergonomics Society 46(1), 50–80 (2004)
34. Madsen, M., Gregor, S.: Measuring human-computer trust. In: 11th australasian conference on information systems. vol. 53, pp. 6–8. Citeseer (2000)
35. Mayer, R.E., Heiser, J., Lonn, S.: Cognitive constraints on multimedia learning: When presenting more material results in less understanding. Journal of educational psychology 93(1), 187 (2001)
36. Moor, J.H.: What is computer ethics?*. Metaphilosophy 16(4), 266–275 (1985)
37. Mousavi, S.Y., Low, R., Sweller, J.: Reducing cognitive load by mixing auditory and visual presentation modes. Journal of educational psychology 87(2), 319 (1995)
38. Muir, B.M.: Trust between humans and machines, and the design of decision aids. International Journal of Man-Machine Studies 27(5-6), 527–539 (1987)

39. Parasuraman, R., Riley, V.: Humans and automation: Use, misuse, disuse, abuse. Human Factors: The Journal of the Human Factors and Ergonomics Society 39(2), 230–253 (1997)
40. Pertet, S., Narasimhan, P.: Causes of failure in web applications (cmu-pdl-05-109). Parallel Data Laboratory p. 48 (2005)
41. Poczter, S.L., Jankovic, L.M.: The google car: Driving toward a better future? Journal of Business Case Studies (Online) 10(1), 7 (2014)
42. Schaefer, K.E., Straub, E.R.: Will passengers trust driverless vehicles? removing the steering wheel and pedals. In: 2016 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA). pp. 159–165. IEEE (2016)
43. Seshadri, A., Luk, M., Qu, N., Perrig, A.: Secvisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity oses. In: ACM SIGOPS Operating Systems Review. vol. 41, pp. 335–350. ACM (2007)
44. Seshadri, A., Luk, M., Shi, E., Perrig, A., van Doorn, L., Khosla, P.: Pioneer: verifying code integrity and enforcing untampered code execution on legacy systems. In: ACM SIGOPS Operating Systems Review. vol. 39, pp. 1–16. ACM (2005)
45. Sikdar, S.K.: Is the software industry environmentally benign? Clean Technologies and Environmental Policy 17(4), 821–822 (2015)
46. Suo, H., Wan, J., Zou, C., Liu, J.: Security in the internet of things: a review. In: Computer Science and Electronics Engineering (ICCSEE), 2012 International Conference on. vol. 3, pp. 648–651. IEEE (2012)
47. Thrun, S.: X-frame-options, or solving the wrong problem (2010, Last accessed: 24th August 2016), `https://googleblog.blogspot.de/2010/10/what-were-driving-at.html`
48. of Transportation, U.D.: Critical reasons for crashes investigated in the national motor vehicle crash causation survey (Last accessed: 24th August 2016), `https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115`
49. Urmson, C.: The view from the front seat of the google self-driving car (11052015, Last accessed: 24th August 2016), `https://backchannel.com/the-view-from-the-front-seat-of-the-google-self-driving-car-46fc9f3e6088#.6qyrhbh9a`
50. Urmson, C.: Just press go: designing a self-driving vehicle (2014, Last accessed: 24th August 2016), `https://googleblog.blogspot.de/2014/05/just-press-go-designing-self-driving.html`
51. Wyglinski, A.M., Huang, X., Padir, T., Lai, L., Eisenbarth, T.R., Venkatasubramanian, K.: Security of autonomous systems employing embedded computing and sensors. IEEE Micro 33(1), 80–86 (2013)
52. Yağdereli, E., Gemci, C., Aktaş, A.Z.: A study on cyber-security of autonomous and unmanned vehicles. The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology p. 1548512915575803 (2015)
53. Yan, Z., Zhang, P., Vasilakos, A.V.: A survey on trust management for internet of things. Journal of network and computer applications 42, 120–134 (2014)
54. Zhang, M., Sekar, R.: Control flow and code integrity for cots binaries: An effective defense against real-world rop attacks. In: Proceedings of the 31st Annual Computer Security Applications Conference. pp. 91–100. ACM (2015)
55. Zissis, D., Lekkas, D.: Addressing cloud computing security issues. Future Generation computer systems 28(3), 583–592 (2012)