Multi-objective Optimisation in Hybrid Collaborating Adaptive Systems

Veronika Lesch, Christian Krupitzer
Software Engineering Group
University of Würzburg
Würzburg, Germany
{veronika.lesch,christian.krupitzer}@uni-wuerzburg.de

Sven Tomforde
Intelligent Systems
University of Passau
Passau, Germany
sven.tomforde@uni-passau.de

***Abstract*—Allowing for self-adaptation in technical systems is intended to tackle the ever-increasing complexity resulting from the open, interconnected, and mobile characteristics of information and communication technology. Typically, self-adaptation is established by means of a feedback loop concept, e.g., in terms of the monitor-analyse-plan-execute (-knowledge) loop as known from the Autonomic Computing domain that acts on top of the productive part of a technical system. Two of the major parts of this loop are related to actually steering the behaviour of the productive part: planning what to do and executing this plan. In this paper, we present a novel concept for multi-objective optimisation-based planning of adaptations in autonomous self-adaptive systems. We focus on a subset of self-adaptive systems that deal with resource coordination problems and highlight issues between central planning and decentral execution of plans by autonomous resources. We discuss four application scenarios to illustrate the challenges and the benefits of our concept.**

***Index Terms*—adaptation, autonomous systems, coordination, optimisation**

I. Motivation

Recent developments in information and communication technology highlight that especially mobility of such systems is increasing, e.g., visible in the context of the Internet-of-Things [1]. A second trend is that these former small, closed systems become large-scale, open-world systems-of-systems [2]. Self-adaptive systems (SASs) [3] address these developments as they are able to change their behaviour at runtime to cope with changes in their environment or the system itself. These systems are the foundation of research communities for adaptive systems, such as Organic Computing (OC) [4] and Autonomic Computing (AC) [5]. Often, SASs reasons on adaptation following the MAPE model [5]: Monitoring the environment and the system, analysing, i.e., search for changes in the environment and the system, planning necessary adaptations, and control the execution of these adaptations. Alternative models with comparable functionality include the LRA-M model [6] or the Observer/Controller model [7].

Approaches for planning adaptation in SASs often rely on rules, goals, models, or utility functions (see [3] for an overview). Through the increasing computational power and its omnipresence (e.g., provided by clouds), approaches based on statistical optimisation procedures gained more importance recently. However, within large systems, these optimisation processes face a trade-off between finding a local vs. global optimal solution. A local optimal solution can be found in a decentralised fashion, i.e., each device can optimise itself autonomously. The shorter processing time and the individual optimal solution have the advantage of scalability but come with the cost of potentially contradicting adaptations. Global optimisation can overcome this issue with the downside of higher processing times. However, the central bottleneck can delay necessary adaptations. Often, researchers balance the trade-off by applying a global optimisation with heuristics to reduce the information load. However, hybrid solutions can be observed where a continuous decentralised optimisation process is augmented with an on-demand (semi-)centralised routine that runs with a larger time-horizon (e.g., [8]).

Another issue for adaptation arises through the systems-of-systems nature of SASs. These systems might be composed of collaborating autonomous systems [9]. For local optimisation, the entities are responsible for planning the adaptation, i.e., the plan complies with the individual preferences and goals. If we assume to have an instance of the system that is able to plan using global optimisation techniques, this does not guarantee that the autonomous entities of the system execute this plan as it might contradict to individual goals. Accordingly, in case of global optimisation, the system requires to guarantee the execution of the plan through coordination.

In this paper, we outline our vision for multi-objective optimisation in collaborating, autonomous SASs. Our approach targets the planning and executing of adaptations in the context of large-scale SASs with autonomous resources. This includes an evaluation of trade-offs between global and local decision making w.r.t. reaction time, costs, and coordination effort for adaptation. Further, our approach uses incentives, coordination, and regulations to enable adaptation.

Next, Section II describes the underlying system model and introduces platooning as running example for this paper. Section III describes the state-of-the-art in the field and identifies the research gap. Afterwards, Section IV explains the resulting research objectives. Section V derives specific challenges from the research statement. Following, Section VI maps the insights on a set of application scenarios and highlights how these applications could benefit from the corresponding efforts. Finally, Section VII concludes the paper.

II. Hybrid Collaborating (Self-)Adaptive Systems

Self-adaptive systems (SASs) are able to change their behaviour at runtime as a reaction to observed changes in the environment or the system itself [3]. These systems are composed of two major parts: an adaptation manager (AM) and managed resources (MR) [10]. The managed resources are a set of resources $MR_{SAS} = \{mr_1, ..., mr_n\}$ with mr_i as any kind of software and hardware, e.g., servers, laptops, smartphones, robots, or unmanned vehicles. The adaptation manager $AM_{SAS} = \{am_1, ..., am_n\}$ is a set of software modules am_i that implement the MAPE functionality. Hence, the self-adaptive system is defined as a tuple $SAS = (AM_{SAS}, MR_{SAS})$ with the adaptation manager AM_{SAS} and the managed resources MR_{SAS} [3].

In literature, different interaction patterns between the adaptation manager and the managed resources can be found, each manifesting a different degree of autonomy for the resources. In an SAS with a central adaptation manager, one instance controls the whole system. This allows for global, optimal planning. The contrary is a fully decentralised approach: each managed resource has its own dedicated adaptation manager.

In these scenarios, the entities might collaborate fully autonomously: They (i) cooperate for fulfilling common tasks, (ii) compete for resources, or (iii) co-exist, i.e., their autonomously planned interactions are not coordinated but do not have conflicting goals. Diaconescu *et. al* refer to these extremes as entity versus collective awareness [11]. In-between these two extreme forms lies a hybrid approach for the adaptation manager. There, some of the MAPE functionality is centralised, others are decentralised. Weyns *et al.* proposed different decentralised control patterns for hybrid control [10]. Usually, scientists assume that the central elements have control over the implementation of central decisions, e.g., in case of central planning that the managed resources adapt according to the plan. However, with a higher degree of autonomy, the managed resources might act independently, i.e, they ignore the decision making [11]. Hence, conflicts arise between the central decision making and the autonomic decision execution. These conflicts may also arise in fully decentralised settings as well as in hierarchical settings with layered MAPE loops. Accordingly, appropriate decision execution approaches are required that force or motivate the adaptation of resources.

A. Coordination as Sub-problem of SAS

As outlined above, SASs define a very broad field of research. In order to focus the research efforts and to be able to come up with a general statement, we restrict our work to a sub-group of SASs: systems with ordering-based coordination problems. Therefore, we state several assumptions (cf. Figure 1) which we discuss in the following:

- An $AM_{SAS} = (G_{SAS}, AM_{ext,1}, \ldots, AM_{ext,l})$ has a set of certain goals G_{SAS} and one or several optional $AM_{ext,l}$ for achieving these goals.
- The $G_{SAS} = (g_1^{SAS}, \ldots, g_n^{SAS})$ may contain several subgoals g_n^{SAS}.

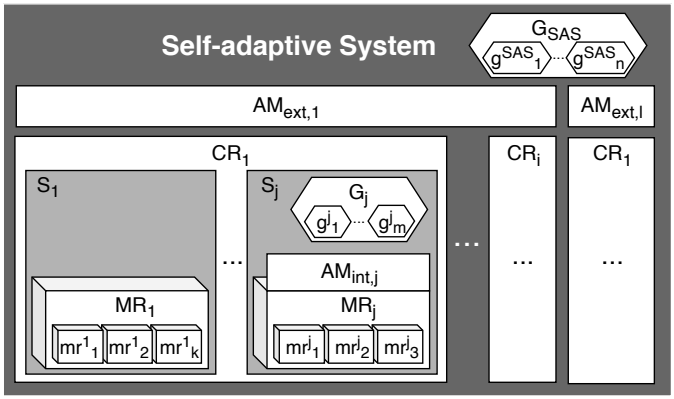


Fig. 1. System model for hybrid collaborating (self-)adaptive systems.

- The $AM_{SAS} = (CR_1, \ldots, CR_i)$ coordinates one or several coordinated resources CR_i.
- The $CR_i = (S_1, \ldots, S_j)$ consists of one or several subsystems S_j.
- Each subsystem $S_j = (MR_j, G_j, AM_{int,j})$ contains one managed resource MR_j and may have an optional set of goals G_j. In case the subsystem has certain goals, an additional internal adaptation manager $AM_{int,j}$ is responsible for coordinating the MR_j.
- The $G_j = (g_1^j, \ldots, g_m^j)$ has, as well as the G_{SAS}, several subgoals g_m^j.
- The $MR_j = (mr_1^j, mr_2^j, \ldots, mr_k^j)$ are composed of several hardware and software elements mr_k^j. In case the MR_j are fully controlled by the AM_{ext}, the AM_{int} just forwards the control signals.
- The position of each subsystem S_j, i.e., the position of each MR_j, within the CR_i group and the common properties of this group have direct impacts on the goal achievement of each subsystem and the overall SAS.
- There are a set of six standard processes that can be performed within such a group context:
  - For participation: 1) `initialiseGroup()`, 2) `dissolveGroup()`, 3) `joinGroup()`, 4) `leaveGroup()`
  - For organisation: 5) `changePosition()`, 6) `changeGroupProperties()`

These processes describe the basic formation and reorganisation behaviour of such a CR. Performing these processes can either be done externally (via AM_{ext}) or internally (via negotiations between the AM_{int}), both resulting in an optimisation problem. We introduce a running example for such a coordination problem in SAS and outline what these processes mean in the scenario within the next paragraphs.

B. Running Example: Platooning

Platooning is driving in convoys of (semi-)automated vehicles with inter-vehicle distances of only a few meters [12]. Accordingly, a platoon p_i is a vector $p_i = (vp_1^i, vp_2^i, ..., vp_n^i)$ with vehicles vp_i^p that are part of a platoon p_i. Traffic in total is a set of platoons $p = \{p_1, p_2, ...p_n\}$ and non-controllable

vehicles vn_i that cannot platoon, e.g., human-driven vehicles. Mapping to our system model, a platoon is a coordinated resource CR, composed of autonomic vehicles as MR_i.

In [13], we present the Platooning Coordination System (PCS) which coordinates the formation of platoons. It searches a suitable platoon and navigates the vehicles to the platoon. Therefore, each vehicle vp_i^p of a platoon p can be described by its input parameters: $vp_i^p = \{route, destination, objective, vehicle\_characteristics\}$. The PCS acts as adaptation manager: using global decision making algorithms it coordinates joining or leaving a platoon of vehicles as well as inter-platoon interactions, e.g., to overtake another platoon or merge with another platoon. Figure 2 shows platooning with the PCS.

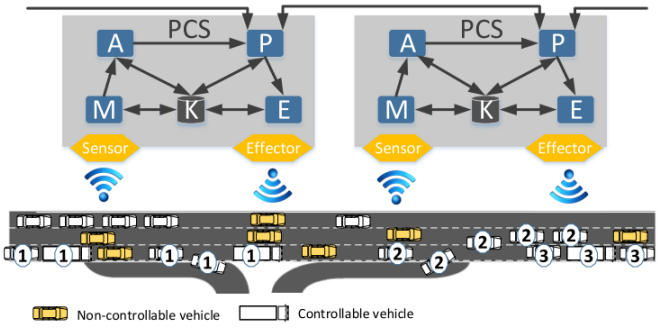


Fig. 2. The platooning process on a highway controlled by the PCS, shown as MAPE loops. A vehicle leaves `platoon(1)`, `platoon(2)` overtakes `platoon(3)`, and, additionally, a vehicle joins `platoon(2)`.

The definition of the six processes of CR problems are defined for platooning as follows:

1) **initialiseGroup()**: A set of vehicles are coordinated to form a new platoon.
2) **dissolveGroup()**: An existing platoon is dissolved.
3) **joinGroup()**: A vehicle or a group of vehicles (including platoons) becomes part of an existing platoon.
4) **leaveGroup()**: A vehicle is removed from the platoon.
5) **changePosition()**: The ordering within the platoon is modified, i.e., vehicles within the platoon change their positions.
6) **changeGroupProperties()**: The properties of the platoon are modified, e.g., speed.

However, as the vehicles are autonomous instances having their own goal function, they might disobey the commands of the PCS. This might be beneficial from the point of view of the vehicles in situations, in which the vehicle should act as platooning leader and would less benefit from platooning due to air drag while the vehicles behind benefit from slipstream effects. Accordingly, platooning is a good example to show our main challenges. On the one hand, our category of systems requires an optimisation approach that balances constraints and multiple objectives in scenarios with central decision making as well as collaborative decision making. On the other hand, it requires a decision execution mechanism with a compensation strategy to control the execution of adaptation plans.

III. State of the Art

Optimisation in SASs. To get an overview of commonly used optimisation techniques in the field of SASs, we analysed the published works of the last ten years in the ACM Autonomous and Adaptive System (TAAS) journal, the International Conference on Autonomic Computing and Communications (ICAC), the International Conferences on Self-Adaptive and Self-Organising Systems (SASO), the Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), and the Symposium on the Foundations of Software Engineering (FSE). We identified 54 publications. In these works, the planning functionality is supported by the following optimisation techniques[1]:

- **Probabilistic Approaches**: Bayesian Networks, Bayesian Optimisation, Simulated Annealing
- **Combinatorial Optimisation Approaches**: Cross-entropy Method for Combinatorial Optimisation, Decentralised Combinatorial Optimisation
- **Evolutionary Approaches**: Evolutionary Algorithm, Genetic Algorithm, Genetic Programming, Learning Classifier System, NSGA-II, SPEA2
- **Stochastic Approaches**: Greedy Algorithm, Markov Decision Process, Stochastic Approximation, Stochastic Programming, Variable Neighbourhood Search
- **Mathematical Optimisation Approaches**: Binary Programming, Integer Programming, Linear Programming, Sequential Quadratic Programming, Convex Optimisation Solver, Pattern Search Algorithm
- **Meta-Heuristics**: Heuristic Algorithm, Tabu Search
- **Others**: Canonical Correlation, Weighted Sum Model, Reinforcement Learning, Distributed Constraint Optimisation, Gradient Descent

The list is of exemplary nature; however, it shows the diversity of optimisation techniques used in SASs as the different optimisation techniques are applicable in different scenarios. With the exception of the approaches presented in [14]–[27], the approaches target optimisation in centralised systems. Hence, decentralised optimisation techniques as required in our targeted system domain is underrepresented. Further, the authors neglect the integration of coordination or negotiation techniques as they assume that instances obey the instruction of the adaptation plan. In our work, we target the study of the resulting issues between the planning phase based on optimisation and its execution on autonomous resources as well as how to support this through decentralised optimisation. A thoroughly analysis of the optimisation techniques used for planning in SASs is part of ongoing work.

Autonomous Self-coordination. Autonomous self-coordination mainly focused resource allocation without external intervention for finding an ordering of requesters by optimising a certain goal (e.g., priorities, cost, waiting times, or fairness). In the context of this paper, we consider three basic classes of autonomous self-coordination:

[1]Due to space constraints, the references can be found at: https://doi.org/10.5281/zenodo.2584266

a) Centralised approaches: This category of approaches uses algorithms such as leader election for choosing one specific node that acts on behalf of the group (see e.g., [28] for an overview of algorithms). Afterwards, the resource allocation or coordination problem is handled in a centralised manner with the leader deciding about the current strategy. There are further concepts for exchanging the leader if it does not act as expected, e.g., measured in terms of fairness metrics. Examples include [29] or [30].

b) Negotiation-based approaches: There are several reasons why a centralised (even a pseudo-centralised solution based on leader election) is not used in specific cases: single-point-of-failure, exploitation of power, communication overhead, or a variety of attack vectors. Consequently, decentralised approaches have been a promising alternative, see, e.g., [31]. Especially in the context of multi-agent systems, solutions among a group of autonomously acting agents that are considered to be equal have been investigated.

Several situations can occur, where agents may not agree, but still need to find a consensus, i.e., a solution that everyone accepts, even if it is not everyone's favourite choice. This helps to achieve overall system reliability in the presence of a number of disagreeing agents. In general, this is referred to as "consensus problem" [32]. Approaches to tackle this include protocols (e.g., the Terminating Reliable Broadcast protocol [33] or the Contract Net protocol [34]), negotiation techniques, or mechanisms such as auctions [35].

c) Emergent approaches: The third group of approaches does not make use of explicit coordination or management mechanisms. In turn, the system is fully decentralised as agents act fully autonomously without the usage of explicit coordination or negotiation techniques. For coordination purposes this generally refers to simple scheduling schemes, e.g., first-come-first-serve (see [36] for an overview). Alternative solutions include Organic Computing concepts, e.g., [37].

A. Identification of the Research Gap

In this vision paper, we focus the interaction of planning adaptation and its execution; two research streams that are often addressed in isolation. This might be valid if the adaptation manager has full control over the resources, e.g., in centralised settings or decentralised systems with fully cooperative decision making. However, in settings with autonomous resources, the aforementioned conflicts between decision making and adaptation execution may arise—which is not addressed explicitly in research so far.

Some researchers integrate techniques from the area of optimisation problems for planning of adaptation. However, often the researchers do not compare which technique might be the best for coping with specific situations of system / environmental states. Further, the types of coordinated systems we target might require decentralised, local decision making. Accordingly, we plan to research decentralised optimisation techniques. This includes an analysis of the required degree of information exchange and coordination. Further, we plan to analyse which techniques are superior in which situations. On the one hand, we analyse the trade-off between local, fast but potentially conflicting optimisation versus time-consuming but global optimisation. On the other hand, we compare the suitability of different techniques based on whether they deliver a usable result in situations where the optimisation procedure is stopped, i.e., they belong to the class of anytime algorithms and, hence, support anytime learning [38].

Regarding the execution of adaptation, research focuses on safe states for execution, i.e., guaranteeing that the system reached quiescence or at least tranquillity [39]. However, researchers often assume (implicitly) that a central instance can use optimisation techniques to find an adaptation plan that is then executed by the autonomous entities. Therefore, these autonomous entities require to behave cooperatively and coordinate the adaptation actions. This includes that entities might behave altruistically in case the global optimal adaptation plan decreases their individual utility. However, this changes if the autonomous entities become self-ish, i.e., they behave competitively. We will study the resulting issues from such situations. This includes the identification of mechanisms to reward those entities that have to decrease their utility for achieving global optimal behaviour as well as interaction mechanisms to control the execution of the adaptation plan.

IV. Research Statement

The identified research gap leads to several objectives and challenges that need to be addressed. A straight-forward approach would rely on a global optimisation of the system in terms of coordination. However, this is highly time-consuming, computational-intensive, and requires a significant load in terms of communication, while simultaneously decreasing the robustness of the system due to introducing a single point of failure. Further, the adaptation decisions are multi-objective and require a balance between different goals within a set of given constraints. Even if we assume that global optimisation is feasible, the autonomous nature of the agents does not guarantee the execution of the adaptation plan. As an alternative, purely decentralised decision making of autonomous entities might be beneficial, which comes with the cost of potentially conflicting decisions. However, (partly) centralised decision making of hybrid approaches requires, again, information exchange and coordination. Further, self-ish entities need to be convinced to act altruistically for global welfare as local optima require coordination to overcome the issue of conflicting adaptation plans.

As a result of these considerations, we propose to investigate the interplay of centralised and decentralised decision making for planning of adaptation and the execution of planned adaptation in coordination scenarios. In particular, we claim that an approach to balance the interests of the autonomous entities with the global optimal settings for the SAS as a whole is needed. This leads to the following research objectives:

Objective 1 Design and implementation of a support process for developers to derive appropriate decentralised and centralised decision making mechanisms for local and global optimisation of adaptations

Objective 2 Development of a centralised optimisation mechanism that balances conflicting goals and constraints

Objective 3 Improvement of the robustness of the solutions by providing degrees of freedom and alternative solutions for the execution-enforcement

Objective 4 Investigation of a subsequently distributed adaptation execution algorithm that controls the application of the plan within a set of autonomous entities by making use of the provided freedom

Objective 5 Analyse the relations of autonomic vs. heteronomic decisions of entities on the optimisation procedure

In the following, we illustrate the practical relevance of the defined objectives using the example of platooning presented in Section II-B. The self-driving vehicles act autonomously based on individual preferences of the driver. However, as they interact in a shared environment with specified interaction rules, this represents a setting of co-existing collaborating entities. To find a platoon, two approaches are feasible. First, the vehicles can collect information about available platoons themselves and decide locally which platoon to join. This requires a vehicle-to-vehicle communication infrastructure or intermediaries for the information exchange and the decisions will be locally optimised. This results in conflicts, e.g., each vehicle tries to optimise its position in a platoon for having the highest possible slipstream effect. Second, an intermediary system can collect the data and plan centrally, which platoon a vehicle should join. Our PCS follows this approach [13]. This enables globally optimal decisions for the multi-objective optimisation problem of balancing the interests of the individual vehicles, platoons, and the global traffic within given constraints. However, these vehicles get instructions on how to join but act autonomously, i.e., it is not guaranteed that they follow these instructions. Especially instructions such as joining a platoon at the front might be critical as being the platoon leader reduces the effects of fuel saving through slipstream effects. Accordingly, the system requires to control the adaptation execution and a reward system for the vehicles to motivate them to obey the instructions in an altruistic manner. In the following, we present several challenges for implementing our vision and how we plan to tackle them.

V. Research Roadmap

In order to achieve these objectives, we propose to investigate novel mechanisms from two perspectives in parallel and with increasing complexity: from a bottom-up (BU, i.e., from a purely decentralised perspective at the autonomous entities themselves) and a top-down (TD, i.e., from the centralised planning and optimisation) perspective. Therein, an increase of complexity of the optimisation problem refers to decreasing the assumption of benevolent behaviour, i.e., decreasing the level of compliance with the computed (TD) or negotiated (BU) solution. This results in a roadmap for tackling the described research statement. For each challenge, the according implementation of the system model from Section II is presented. The first part of challenges deals with the top-down perspective.

Challenge TD-1: Multi-objective central optimisation with constraints

- Goal: Provide the best possible solution within a given time constraint.
- Which are suitable basic optimisation techniques?
- How to balance the trade-off between fast response and optimal solutions?
- How to incorporate constraints (e.g., given by characteristics and preferences of entities)?

In the first setting, we look at a central adaptation manager with central, global optimisation (see Figure 3a). This means for the system model that there is only one AM_{ext} that is responsible for optimisation of the entire system. This optimisation includes single-objective and multi-objective optimisation. As the optimisation is running during the execution of the system, it might be necessary to sacrifice an optimal solution for using fast-responding heuristics. We plan to compare different optimisation categories such as evolutionary approaches, stochastic approaches, and mathematical approaches. Further, the approach integrates global constraints for the planning procedure. The managed resources send monitoring data and receive instructions for adaptation which they obey, i.e., they execute decisions heteronomically.

Challenge TD-2: Multi-objective central optimisation with adaptation freedom

- Goal: Provide solutions with alternative realisations and degrees of freedom for the adaptation execution.
- How to define ranges of allowed configurations for degrees of freedom for the adaptation execution?
- How to ensure compliance of autonomous adaptation execution?
- How to generate feedback from the adaptation enforcement that guides the search within the planning process?

The adaptation manager integrates central planning for global, multi-objective optimisation (see Figure 3b). Monitoring and execution are located at the resources, i.e., we have a combination of one AM_{ext} for analysing and planning and many AM_{int} for monitoring and execution. The local execution enables the integration of local constraints. As multi-objective optimisation usually generates a set of Pareto-optimal solutions, we have a many-objective approach as the adaptation freedom for the managed resources might lead to a situation in which the resources execute different planning aspects targeting different objectives. However, compliance of the adaptations must be ensured. Therefore, different procedures known from reinforcement learning might be feasible to either reward the instances or punishing them. Within the bottom-up stream, we have two corresponding challenges.

Challenge BU-1: Decentralised non-coordinated optimisation

- Goal: Negotiate solutions in open collections of autonomous entities.
- How to avoid re-optimisation if composition changes?
- How to incorporate multiple goals in the negotiation?
- How to act on local information sub-sets?

In both bottom-up settings, we do not integrate an external adaptation manager AM_{ext} (see Figure 3c). Rather, each subsystem has a AM_{int} and tries to optimise for its individual constraints and objectives. Similar to TD-1, the system uses single- and multi-objective optimisation. Using a local, smaller set of information can speed up the optimisation. Further, it eliminates the single point of failure of a central approach. The advantages raise several challenges that can result from the uncoordinated adaptation, such as concurrency of conflicting objectives, adaptation oscillation, and reasoning incomplete information.

Challenge BU-2: Decentralised coordinated multi-objective optimisation

- Goal: Negotiate solutions in open collections of autonomous entities.
- Which aspects need to be taken into account?
- How to integrate local constraints while obeying coordination results?
- How to include an invcentivation for coordination?

Contrary to the uncoordinated setting in BU-1, in challenge BU-2, the different AM_{int} coordinate the adaptation execution (see Figure 3d). Still, the planning takes place locally as each AM_{int} follows a multi-objective approach. However, coordination enables the fulfillment of several objectives of different AM_{int}. This represents a many-objective optimisation approach for planning. Accordingly, similar flexible optimisation procedures as in challenge BU-1 are required for flexibility through the adaptation execution. Efficient coordination approaches are necessary. The interplay of the TD and BU challenges result in some hybrid challenges (HY) targeting both categories:

Challenge HY: Distributed global, multi-objective optimisation

- Goal: Coupling of global planning and local execution.
- How to effectively coordinate a decentralised planning process within autonomous agents?
- How to enforce plans if collections change continuously?

This system setting combines the advantages of a global optimal, conflict-free optimisation for planning of adaptation with the data-reduced local decision making process which obeys local constraints. In contrast to the previous setting, coordination now targets the planning instead of the execution (see Figure 3e). Hence, this approach relies on a distributed optimisation approach, i.e. only AM_{int} are involved in the planning process. Planning is based on a coordinated optimisation procedure that runs distributed on several planners, hence, it eliminates the need of a central element for planning but requires exchange of monitoring and analysing information. For the execution, the system requires a coordination mechanism to ensure the execution of the plan, since resources might decide autonomously on the execution of the plans. Last, we define several cross-aspect challenges that for all categories:

Challenge Cross-1: Evaluation and generalisation

- Goal: In-depth analysis of the coupled behaviour between planning and execution.
- Which metrics are meaningful? Which scenarios need to be covered?
- Can we generalise the results to other use cases?

Challenge Cross-2: Decision support process for engineers

- Goal: Development of a decision support process that simplifies and improves the design process for engineers.
- In which cases are which solutions appropriate?
- How to trade-off between decentralised, centralised, and hybrid solutions?
- How to standardise the design process for these aspects?

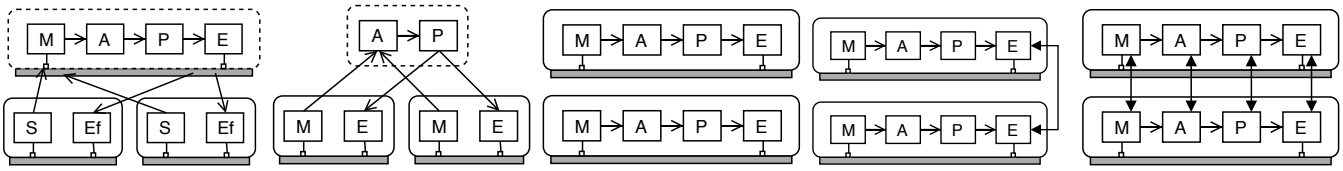
The challenges Cross-1 and Cross-2 concern all five system settings presented above and are underlying aspects concerning the evaluation of the mechanisms as well as the decision support delivered for application developers. Regarding the evaluation, the definition of metrics is important to compare different optimisation techniques or the coordination impact. Further, we need to apply the mechanisms in different use cases to avoid a restriction to a certain use cases. To simplify the application of the mechanisms, these should be integrated into frameworks or development processes. Additionally, researchers require development support in choosing between central, decentral, and hybrid approaches to find the optimal approach in terms of trade-offs as well as their (dis)advantages. The use of the different mechanisms should be integrated into a standardised process which makes them easily exchangeable.

TABLE I
DIMENSIONS OF DECISION EXECUTION AND DECISION MAKING WITHIN DIFFERENT PLANNED USE CASE SYSTEMS. IN PARENTHESIS, WE SPECIFY THE RELATED CHALLENGES FOR THE SCENARIOS.

	Autonomic Decision Execution	**Heteronomic Decision Execution**
Central Decision making	Highway platooning (TD-2, HY)	Industry 4.0 inventory management (TD-1)
Local Decision making	Code offloading (BU-1)	Inner-city platooning (BU-2, HY)

VI. APPLICATION SCENARIOS

In order to allow for a proof of general applicability of the developed methods, an investigation in different use cases is required. Therefore, we have to define use cases along two different axes: (i) from autonomic to heteronomic enforcement of planning results and (ii) from centralised to decentralised decision making. Dimension (i) targets the execution activity while dimension (ii) focuses on the planning step. Autonomic decision execution of the resources relates to the mentioned BU approaches; heteronomic decision execution symbolises the TD approaches. Similar, central decision making refers to the TD approaches; decentralised decision making are problems from the category of BU approaches. Table I provides an overview of the corresponding space with four possible use cases for each dimension conforming to the above described challenges. In the following, this section presents them.



(a) TD 1 (b) TD 2 (c) BU 1 (d) BU 2 (e) HY

Fig. 3. Overview on the challenges exemplified with systems having two resources. `M`, `A`, `P`, and `E` are the corresponding MAPE functions. `S` stand for the sensor, `Ef` for the effector interfaces. Dashed lines indicate an external AM, solied lines an internal AM.

A. Use Case 1: Platooning Coordination on Highways

The first use case represents autonomic decision / central decision making and focuses on the coordination of platooning on a highway by our PCS (cf. Section II-B). The PCS has to balance several (partly) conflicting objectives:

- Global optimisation for all traffic participants,
- optimisation of the interactions between platoons,
- optimisation of the vehicle order within a platoon, and
- individual optimisation objectives of drivers.

The execution of the adaptation plans as output of the central decision making approach is enforced on autonomously deciding vehicles. We assume fully autonomous driving; however, since vehicles act on behalf of their drivers', transparency of plans is required and autonomous behaviour may ignore the plan. Accordingly, the execution requires a mechanism with incentives. Alternatively, the PCS could force vehicles to join a specific platoon by omitting the possibility to choose the platoon. Additionally, uncertainty is added though non-controllable vehicles which are further constraints.

B. Use Case 2: Inner-city Platooning

Previous work defined a concept for a hybrid platooning approach which integrates platooning on highways with an urban traffic management system for achieving platooning in cities [13]. In contrast to platooning on highways, platoons in cities are loosely-coupled connections of vehicles. Their formation is controlled by the traffic management system and is established via different actuators, e.g., traffic light control, dynamic assignment of bus/taxi lanes, and re-routing of traffic. Consequently, this may serve as a representative of the category heteronomic enforcement / local decision making.

As a first step, the existing decentralised traffic management systems [40], [41] have to be integrated with the testbed for the PCS. Based on this integration, an extension of the system is required which is able to cope with the additional functionality, e.g., for dynamic re-routing through vehicle-to-infrastructure communication and adaptive signalling system.

The vehicles are forced to obey the traffic light control. As we assume to have self-driving vehicles present, we assume that they also obey the re-routing instructions.

C. Use Case 3: Industry 4.0 Inventory Management

In modern production facilities, often self-driving vehicles / robots are able to transport items or goods within the facility. As the systems share routes, coordination is necessary. Further, the flexibility demand for Industry 4.0 production processes requires also flexibility in planning the routes. This is a multi-dimensional optimisation problem as it has to integrate different aspects besides just finding the shortest path, e.g., prioritisation of items for premium customers. As this requires the integration of various data sources, the coordination of the vehicles is done centrally, i.e., central decision making is necessary. Further, the vehicle follows the instruction of the central system representing heteronomic decision execution.

We plan such a scenario for inventory management which requires the interaction of heteronomically coordinated vehicles with either employees or robots for loading items to/from the inventory shelves. The central coordination system and the robots should be simulated as well as emulated in a testbed.

D. Use Case 4: Code Offloading with the Tasklet System

The two trends of mobility of resources as well as the omnipresence of computation resources through Cloud Computing converge to the new trend of code offloading. Mobile devices – with potentially limited computational capacity – benefit from code offloading as they can offload computation-intensive tasks and, through that, save energy. Different approaches exist for that purpose. The Tasklet system [42] focuses on code offloading for edge and IoT devices.

The idea of the Tasklet system is to provide a middleware-based infrastructure for distributed computing on heterogeneous devices. Three classes of entities exist: resource providers, resource consumers, and resource brokers. Resource providers offer the possibility to execute the offloaded code. Resource consumers contact one or several resource broker(s) for finding suitable resource providers. Brokers form a peer-to-peer overlay network. The requested broker(s) reply a list of resource providers, out of which the resource consumer can choose one based on different metrics such as costs.

The Tasklet use case is an example of local decision making. Each broker has a limited set of registered resource providers only. Further, each resource consumer decides on its own to which resource provider it will offload the tasks, i.e., follows an autonomic decision execution. Since some resource providers may be superior to others, they may be flooded with requests. This can be avoided by a more heteronomic decision execution, i.e., through negotiations between the brokers but also between resource providers and consumers.

VII. Conclusion

Self-adaptive and self-organising systems attracted increasing research activities due to the need of mastering complexity in interconnected and autonomous systems. These adaptation capabilities are established by means of a feedback loop concept that performs monitoring, analysing, planning, and execution steps. However, planning and execution are typically considered in an isolated manner. Consequently, this paper proposes to shift research attention towards a better integration of these two steps. We therefore defined a system model that defines a sub-class of SASs with coordination tasks. Based on this model, we discussed the objectives required to close the gap in research and described the most important challenges. We explained the necessary efforts in terms of four different case studies. Next we will address these challenges.

References

[1] J. A. Stankovic, "Research directions for the internet of things," *IEEE IoTJ*, vol. 1, no. 1, pp. 3–9, 2014.

[2] K. L. Bellman, S. Tomforde, and R. P. Würtz, "Interwoven systems: Self-improving systems integration," in *Proc. SASOW*, 2014, pp. 123–127.

[3] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, and C. Becker, "A Survey on Engineering Approaches for Self-Adaptive Systems," *PMCJ*, vol. 17, no. Part B, pp. 184–206, 2015.

[4] C. Müller-Schloer and S. Tomforde, *Organic Computing – Techncial Systems for Survival in the Real World*. Birkhäuser Verlag, 2017.

[5] J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," *IEEE Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[6] S. Kounev, P. Lewis, K. L. Bellman, N. Bencomo, J. Camara, A. Diaconescu, L. Esterle, K. Geihs, H. Giese, S. Götz, P. Inverardi, J. O. Kephart, and A. Zisman, "The Notion of Self-aware Computing," in *Self-Aware Computing Systems*. Springer, 2017, pp. 3–16.

[7] S. Tomforde, H. Prothmann, J. Branke, J. Hähner, M. Mnif, C. Müller-Schloer, U. Richter, and H. Schmeck, "Observation and Control of Organic Systems," in *Organic Computing - A Paradigm Shift for Complex Systems*. Birkhäuser Verlag, 2011, pp. 325 – 338.

[8] S. Tomforde, H. Prothmann, J. Branke, J. Hähner, C. Müller-Schloer, and H. Schmeck, "Possibilities and Limitations of Decentralised Traffic Control Systems," in *Proc. WCCI*, 2010, pp. 3298–3306.

[9] A. Diaconescu, S. Frey, C. Müller-Schloer, J. Pitt, and S. Tomforde, "Goal-oriented Holonics for Complex System (Self-)Integration: Concepts and Case Studies," in *Proc. SASO*, 2016, pp. 100 – 109.

[10] D. Weyns, B. R. Schmerl, V. Grassi, S. Malek, R. Mirandola, C. Prehofer, J. Wuttke, J. Andersson, H. Giese, and K. M. Göschka, "On Patterns for Decentralized Control in Self-Adaptive Systems," in *Software Engineering for Self-Adaptive Systems II*. Springer, 2013, pp. 76–107.

[11] A. Diaconescu, K. L. Bellman, L. Esterle, H. Giese, S. Götz, P. Lewis, and A. Zisman, *Architectures for Collective Self-aware Computing Systems*. Springer, 2017, pp. 191–235.

[12] T. Robinson, E. Chan, and E. Coelingh, "Operating Platoons On Public Motorways: An Introduction To The SARTRE Platooning Programme," in *Proc. ITSWC*, 2010.

[13] C. Krupitzer, M. Segata, M. Breitbach, S. S. El-Tawab, S. Tomforde, and C. Becker, "Towards Infrastructure-Aided Self-Organized hybrid platooning," in *Proc. GCIoT*, Alexandria, Egypt, 2018.

[14] V. Nallur, N. Cardozo, and S. Clarke, "Clonal plasticity: A method for decentralized adaptation in multi-agent systems," in *Proc. SEAMS*, 2016, pp. 122–128.

[15] P. Pilgerstorfer and E. Pournaras, "Self-adaptive learning in decentralized combinatorial optimization - a design paradigm for sharing economies," in *Proc. SEAMS*, 2017, pp. 54–64.

[16] M. A. M. de Oca, T. Stuetzle, M. Birattari, and M. Dorigo, "Incremental social learning applied to a decentralized decision-making mechanism: Collective learning made faster," in *Proc. SASO*, 2010, pp. 243–252.

[17] A. J. Ramirez, B. H. C. Cheng, and P. K. McKinley, "An evolutionary approach to network self-organization and resilient data diffusion," in *Proc. ICAC*, 2011, pp. 198–207.

[18] C. Lee and J. Suzuki, "An immunologically-inspired autonomic framework for self-organizing and evolvable network applications," *ACM TAAS*, vol. 4, no. 4, pp. 22:1–22:34, 2009.

[19] C. J. v. Leeuwen, K. S. Yildirim, and P. Pawelczak, "Self adaptive safe provisioning of wireless power using dcops," in *Proc. SASO*, 2017, pp. 71–80.

[20] K. Zhang, E. G. Collins, Jr., and D. Shi, "Centralized and distributed task allocation in multi-robot teams via a stochastic clustering auction," *ACM TASS*, vol. 7, no. 2, pp. 21:1–21:22, 2012.

[21] Y.-X. Wang, Q.-L. Xiang, and Z.-D. Zhao, "Particle swarm optimizer with adaptive tabu and mutation: A unified framework for efficient mutation operators," *ACM TAAS*, vol. 5, no. 1, pp. 1:1–1:27, 2010.

[22] J. Hao and H.-F. Leung, "Achieving socially optimal outcomes in multiagent systems with reinforcement social learning," *ACM TAAS*, vol. 8, no. 3, pp. 15:1–15:23, 2013.

[23] L. Kraemer and B. Banerjee, "Reinforcement learning of informed initial policies for decentralized planning," *ACM TAAS*, vol. 9, no. 4, pp. 18:1–18:32, 2014.

[24] E. Pournaras, P. Pilgerstorfer, and T. Asikis, "Decentralized collective learning for self-managed sharing economies," *ACM TAAS*, vol. 13, no. 2, pp. 10:1–10:33, 2018.

[25] H. Wang, X. Chen, Q. Wu, Q. Yu, X. Hu, Z. Zheng, and A. Bouguettaya, "Integrating reinforcement learning with multi-agent techniques for adaptive service composition," *ACM TAAS*, vol. 12, no. 2, pp. 8:1–8:42, 2017.

[26] A. Marinescu, I. Dusparic, and S. Clarke, "Prediction-based multi-agent reinforcement learning in inherently non-stationary environments," *ACM TAAS*, vol. 12, no. 2, pp. 9:1–9:23, 2017.

[27] I. Dusparic and V. Cahill, "Distributed w-learning: Multi-policy optimization in self-organizing systems," in *Proc. SASO*, 2009, pp. 20–29.

[28] S. Dolev, A. Israeli, and S. Moran, "Uniform dynamic self-stabilizing leader election," *IEEE TPDS*, vol. 8, no. 4, pp. 424–440, 1997.

[29] S. Edenhofer, S. Tomforde, J. Kantert, L. Klejnowski, Y. Bernard, J. Hähner, and C. Müller-Schloer, "Trust communities: An open, self-organised social infrastructure of autonomous agents," in *Trustworthy Open Self-Organising Systems*, 2016, pp. 127–152.

[30] M. Ji, A. Muhammad, and M. Egerstedt, "Leader-based multi-agent coordination: Controllability and optimal control," in *Proc. Am. Ctrl. Conf.*, 2006, pp. 6–pp.

[31] M. Weißbach, P. Chrszon, T. Springer, and A. Schill, "Decentrally coordinated execution of adaptations in distributed self-adaptive software systems," in *Proc. SASO*, 2017, pp. 111–120.

[32] G. F. Coulouris, J. Dollimore, and T. Kindberg, *Distributed systems: concepts and design*. pearson education, 2005.

[33] G. Bracha, "Asynchronous byzantine agreement protocols," *Information and Computation*, vol. 75, no. 2, pp. 130–143, 1987.

[34] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Trans. Computers*, no. 12, pp. 1104–1113, 1980.

[35] N. R. Jennings, P. Faratin, A. R. Lomuscio, S. Parsons, M. J. Wooldridge, and C. Sierra, "Automated negotiation: prospects, methods and challenges," *Group Decision and Negotiation*, vol. 10, no. 2, pp. 199–215, 2001.

[36] M. L. Pinedo, *Scheduling: theory, algorithms, and systems*. Springer, 2016.

[37] A. Scheidler, D. Merkle, and M. Middendorf, "Congestion control in ant like moving agent systems," in *Proc. Biologically-Inspired Collaborative Computing*, 2008, pp. 33–43.

[38] J. J. Grefenstette and C. L. Ramsey", "An approach to anytime learning," in *Machine Learning Proceedings 1992*. Morgan Kaufmann, 1992, pp. 189 – 195.

[39] A. J. Ramirez, B. H. Cheng, P. K. McKinley, and B. E. Beckmann, "Automatically Generating Adaptive Logic to Balance Non-functional Tradeoffs During Reconfiguration," in *Proc. ICAC*, 2010, pp. 225–234.

[40] H. Prothmann, S. Tomforde, J. Branke, J. Hähner, C. Müller-Schloer, and H. Schmeck, "Organic Traffic Control," in *Organic Computing — A Paradigm Shift for Complex Systems*, 2011, pp. 431–446.

[41] M. Sommer, S. Tomforde, and J. Hähner, "An organic computing approach to resilient traffic management," in *Autonomic Road Transport Support Systems*, 2016, pp. 113–130.

[42] D. Schäfer, J. Edinger, J. M. Paluska, S. VanSyckel, and C. Becker, "Tasklets: "Better than Best-Effort" Computing," in *Proc. ICCCN*, 2016, pp. 1–11.