# Petri Nets as Run-Time Models for Self-Adaptive Cyber-Physical Systems

Cost-Benefit Analysis of Structural Transformations

Lorenzo Capra<sup>1</sup>, Michael Köhler-Bußmeier<sup>2</sup>, Heiko Rölke<sup>3</sup> and Jan Sudeikat<sup>2</sup>

<sup>1</sup>Dipartimento di Informatica, Università degli Studi di Milano, Via Celoria 18, Milano, Italy <sup>2</sup>Hamburg University of Applied Sciences, Berliner Tor 7, D-20099 Hamburg, Germany <sup>3</sup>University of Applied Science of the Grisons, Pulvermühlestrasse 57, CH-7000 Chur, Switzerland

#### Abstract

In our general research we study adaptive systems of autonomous agents – with a strong emphasis on agents in cyber-physical systems (CPS). To enable adaptivity, we have a special interest in agents that are embedded into an overall structure, called: organisation. Our SONAR formalism is a specification of the organisational structure; it is based on Petri nets.

In this contribution we investigate the whole life-cycle of SONAR models, from (i) the analytical usage in the design phase, over (ii) their use as configuration data in the deployment phase, to (iii) their use as a digital twin used for cost-benefit reasoning at run-time during the SONAR-MAPE-Loop.

We also present a rule set of transformations for SONAR models. These rules are applicable at different phases of the life cycle: They are used by the modeller at design time, e.g. to refine an early, abstract model into a more concrete one; they are also used by the run-time engine, i.e., the SONAR-MAPE-Loop, as adaption operations during a self-modification.

Remarkably, these two phases – design and run-time – are intertwined in a cyclic way: Obviously, the SONARmodel (usually amended by an deployment profile, called cube-protocols) is used to instantiate the running multi-agent system; but, conversely, we use process mining techniques to (i) adjust the deployment profile in a re-configuration phase or (ii) to trigger a complete re-design phase of the SONAR-model.

#### Keywords

Self-adaptation, MAPE-loop, cyber physical systems, multi-agent systems, models@run.time, Petri nets

# 1. SONAR-Models for Cyber-Physical Systems

Our application are Cyber-Physical Systems (CPS) [1, 2, 3]. We follow an agent-oriented approach to CPS [4] since central concepts of agents (like autonomy, negotiation in case of conflicting goals, etc. [5]) fit very well. For example, agent based implementation approaches are studied within the manufacturing [6] and energy system [7] domains. Implementations of agents in industrial settings are supported by numerous works, e.g. deriving design patterns [8, 9], integration approaches [10, 11] and recommendations for deployment options [12]. These approaches enable autonomous, possibly goal-oriented, system elements in industrial and cyber-physical settings. For multi-agent systems (MAS) it is quite common to complement these concepts, which mainly address the micro level, i.e., the agent side of a MAS, with concepts for the macro-level, like hierarchies, roles, norms, teams, interaction protocols, etc. This approach is known as an organisation-based MAS (Org-MAS for short) [13, 14].

As a major challenge we consider the demand for adaptivity [15, 16] for these systems, especially if adaption has to happen at run-time. The basic pattern for adaption at run-time is known as the MAPE-loop (monitor, analyse, plan, execute) [17, 18]. It is widely assumed that it is beneficial to embed

PNSE'24, International Workshop on Petri Nets and Software Engineering, 2024

<sup>🛆</sup> capra@di.unimi.it (L. Capra); michael.koehler-bussmeier@haw-hamburg.de (M. Köhler-Bußmeier);

heiko.roelke@fhgr.ch (H. Rölke); jan.sudeikat@haw-hamburg.de (J. Sudeikat)

ttps://capra.di.unimi.it/ (L. Capra); https://www.haw-hamburg.de/michael-koehler-bussmeier (M. Köhler-Bußmeier); https://www.fhgr.ch/personen/person/roelke/ (H. Rölke); https://www.haw-hamburg.de/jan-sudeikat (J. Sudeikat)
 0000-0002-1029-1169 (L. Capra); 0000-0002-3074-4145 (M. Köhler-Bußmeier); 0000-0002-9141-0886 (H. Rölke); 0009-0007-7871-9006 (J. Sudeikat)

<sup>© 024</sup> Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

the system's model into the system so it can be used as knowledge during the MAPE-loop, an approach known as *models at run-time* [19, 20].

In previous work [21] we identified four research challenges RC1 - RC4 for organisation-based MAS in the CPS domain, which we recall here in short:

- Mining of Organisational Models (RC1): We will use techniques from process mining [22] to obtain organisation models. As a speciality we treat mining as an ongoing, i.e., online process, which takes execution data from logs together with the current organisational model and generates a more appropriate model as a modification of the current one using incremental techniques like [23, 24].
- Application Specific Support for Model Transformation (RC2): We need transformations, like refinement operations, which are specifically designed for organisation models in the domain of cyber-physical systems.
- Organisation Embedding and Execution Environment (RC3). Here, we study how we integrate organisation models into a heterogeneous MAS in a (semi-)automated way. We argue that, especially for adaptive systems, it is desirable to maintain the model as an run-time object to support reflection and self-modification. This approach is known as models@run.time [19, 20].
- Development Support (RC4): For the practical use we have a need for patterns of inter-agent structures (auctions, voting, social choice, etc.) in order to reflect best practices in industrial MAS development. We have to study whether it is possible to incorporate these patterns as a kind of profile when the organisational model is deployed into a MAS.

Here, we study these challenges in the context of our SONAR framework [25, 26]. A SONAR model is an Org-MAS specification that is designed to support the agents to deal with the system's tasks. A SONAR model describes the formal, organisation part of the MAS. More concretely, a SONAR model is deployed as an Org-MAS, i.e., a network of OPAs (organisational position agents). These OPAs represent the organisation as a macro structure; they enable the cooperation patterns pre-described by the organisation hierarchy, which comes with roles and obligations. However, the OPAs (organisational position agents) are not responsible to implement these roles and obligations, because this is the responsibility of the OMAs (organisational member agents). Each OPA acts as a proxy of its OMA into the organisation. In the context of CPS the OMA usually represents an entity like a machine.

**Example** Figure 1 shows our standard example Org-MAS to illustrate the OPA/OMA network. It consists of a formal organisation containing another organisation as a sub-part. We have the position *executive* represented by an OPA. This OPA is connected to Alice, which is an OMA implementing the position. Note that we allow for OMA-OMA interaction, e.g. between Alice and Bob, but his interaction is somehow private and has to be distinguished from an 'official' organisation interaction that takes places when Alice and Bob communicate via their OPAs *Executive Operator A*. Fiona is an example for an outside agent not belonging to the Org-MAS, i.e. an agent that neither is an OPA nor an OMA.



Figure 1: The Org-MAS deployed as an OPA/OMA network (adapted from [25])

**Aim of the Paper** In this paper we will address the research challenges from the perspective of the organisational SONAR-model underlying the systems. We present our vision how the four research challenges will be addressed in the life-cycle of SONAR-models in an *integrated* way.

In Section 2 we explain the general connection between the organisation model with a given underlying cyber-physical system (CPS). Section 3 describes the use of SONAR-Models, i.e. their life-cycle. As our MAS is self-adaptive we present basic transformation operations for SONAR-models in Section 4. Section 5 demonstrates how the self-adaption mechanism exploits our basic transformation operations in combination with given CPS key performance indicators (KPI) to establish a cost-benefit based reasoning to enable a goal-directed adaption at run-time. Section 6 illustrates the presented concepts by studying the price-of-anarchy in organisations for the example of a well-known coordination game: the battle-of-sexes scenario. The work ends with a conclusion and outlook.

**Related Work** The central approach to adaptivity in software engineering is known as the MAPE-K approach, i.e., MAPE with knowledge K, where the knowledge has much in common with the organisational concepts [15, 27]. A rational approach to adaption based on cost-benefit reasoning is studied in [28]. The work presented here also has some connections to our research on adaption state spaces [29, 30], which we have studied for the self-modifying Petri net class of HORNETS [31, 32, 33].

Prominent examples of Org-MAS from the design and implementation perspective are AGR [34], MOISE [35], and OPERA [36]. An overview is given in the handbook of Dignum et al. [37]. A complementary approach to organisational design comes from Process Mining [22], which also includes the mining of structures.

### 2. Connecting a CPS with its Organisation

We follow an agent oriented approach to CPS [4]. The organisational network of OPAs can be seen as an overlay onto this CPS, which gives the system a desired structure. Agents are used to control and represent physical devices (e.g. according to [38]). Thus, each CPS component acts as a member agent, i.e. an OMA, of the Org-MAS. Let us have a look at the basic perspective on these CPS agents, i.e. to consider them as a network of autonomous components with sensors and actuators.

#### 2.1. The underlying CPS

The CPS defines some conditions on the systems as it provides the existing *sensors* and *actuators*. The world state is accessible only via the given sensors and the agents can manipulate the world only via the given actuators.

We assume that the MAS state is not directly observed, but observed. There are two classes:

- 1. Boolean observables  $\phi^{\mathbb{B}} : States \to \mathbb{B}$ , i.e.,  $\phi^{\mathbb{B}}(state)$  is a state predicate.
- 2. Quantified observables  $\phi^{\mathbb{R}}$ : *States*  $\to \mathbb{R}$ , i.e.,  $\phi^{\mathbb{R}}(state)$  is a key performance indicator (KPI).

Agents observe the system only by these sensors, i.e., they cannot distinguish states that are indistinguishable by sensors. For each MAS we assume a given set of state predicates and one of KPIs.

Actions define the operations in the teamwork protocols:  $act : States \rightarrow States$ . We assume a given set ACT of actions for our CPS.

**Definition 1.** A cyber-physical system (CPS) is modelled as the tuple

$$CPS = (States, \Phi^{\mathbb{B}}, \Phi^{\mathbb{R}}, ACT)$$

- States is the set of all system states.
- $\Phi^{\mathbb{B}}$  is a set of Boolean observables  $\phi^{\mathbb{B}} : States \to \mathbb{B}$ .
- $\Phi^{\mathbb{R}}$  is a set of quantified observables  $\phi^{\mathbb{R}}$ : States  $\to \mathbb{R}$  (performance indicators).
- ACT is a set of actions.

The agents' activities are structured by agent interaction protocols. It is convenient to use a multiparty variant of the well-known workflow nets [39, 39] to specify these protocols. These nets are called distributed workflow nets (DWFN) [40, 26]. We assume that each agent activity in the interaction protocol D is connected to an action, i.e., it has a specified effect on the observables.

These DWFN are interactions between roles  $R \subseteq \mathcal{R}$ , where a role induces a fragment D[R], i.e., a subnet of the workflow, which contains the part that is assigned to R. Agents must have the required capabilities to implement a role, and the role must own the required rights to execute the actions that are part of the role fragment.

#### 2.2. The Organisational Overlay: SONAR

In the following we consider a multi-agent perspective for CPS [4]. Additionally, we use organisations (here: organisation-based MAS, short: Org-MAS) to structure the systems in the large [3, 41]. This organisational structure of the MAS can be seen as a system overlay [3, 41]. We use SONAR as our organisation framework. In the following, we recall some basics and refer to [25, 26, 42, 43] for details.

**SONAR-Tasks** A central concept for SONAR is that of a *task*. A *task*  $p_0$  is an obligation for an agent (generated either from the environment or the MAS itself) to reach a state by taking actions in a way that (i) satisfies the goal  $goal(p_0)$ , which is a formula defined using the boolean observables, and (ii) optimises the effect on the KPI. We assume a given set  $P_0$  of initial tasks. An initial task is triggered by the observables by if-then rules, denoted as:

if  $\langle trigger \rangle$  then  $\langle create initial task p_0 \in P_0 \rangle$  end

Here, a *trigger* is an expression in propositional logic where we have the boolean observables  $\phi^{\mathbb{B}}$  and the inequalities  $(\phi^{\mathbb{R}} \leq x)$  for each quantified observable  $\phi^{\mathbb{R}}$  and each value  $x \in \mathbb{Q}$  as logical atoms. We use  $(\phi^{\mathbb{R}} \geq x), (\phi^{\mathbb{R}} = x), (\phi^{\mathbb{R}} < x)$ , etc. as abbreviations. A state *s* enables the task if its trigger evaluates to true in *s*.

**The Sonar-Organisation** We assume a given cyber-physical system CPS and a set TR of trigger rules. An organisation  $Org = (N, \mathcal{O}, \mathcal{R}, \mathcal{D})$  consists of a organisation net N, a set of OPAs  $\mathcal{O}$ , a set of roles  $\mathcal{R}$ , and a set of DWFN  $\mathcal{D}$  that is based on these roles. Our standard example for a SONAR organisation is shown in Figure 2. It has one initial task  $p = \text{task}_{PC[Prod, Cons]}^{O_1}$  for the OPA  $O_1$ , placed at the top. Here,  $O_1$  has the obligation to handle the task via the protocol PC[Prod, Cons]. In this case, Prod and Cons are all the roles in the DWFN PC. The highlighted part of the net describes a SONAR-Team that has been formed at run-time to handle the task. The existing conflicts have been resolved by reasoning of the OPAs. Note that conflicts are always internal to OPAs.

The organisation net N = (P, T, F) is a Petri net [45], where each place is of the form  $p = \operatorname{task}_{D[R]}^{a}$ , which describes a *task* or *sub-task* for the agent *a* to establish the role part *R* of the DWFN *D*. The tasks are either generated in the environment or they are sub-tasks, generated by the organisation itself. A place *p* is a task whenever  $\bullet p = \emptyset$  and we set  $P_0 = P_0(Org) := \circ P := \{p_0 \in P \mid \bullet p_0 = \emptyset\}$ .

Each task is handled by the transitions of the organisation net, which are called *team-operators*. We have four types of operators:

- 1. Delegate: The task to implement DWFN D[r] is delegated from agent a to b. Only the delegation operation delegates the ownership of a task.
- 2. Split: The task to implement  $R = \{r_1, \ldots, r_n\}$  in D is split into n sub-tasks to implement  $\{r_i\}$  in D.
- 3. Refinement: The DWFN D[r] is replaced by  $D'[r_1, \ldots, r_n]$ , which has to be a behaviour equivalent refinement, i.e. they are bisimilar.
- 4. Assign execution: The DWFN D[r] is assigned to an agent for execution.



Figure 2: A SONAR-Organisation Model; highlighted: a SONAR-Team (adapted from [44])

Each team-operator t also imposes a constraint  $\psi(t) \in \Psi$  onto the execution of the generated team-DWFN  $D_G$  where  $\Psi$  denotes a given set of constraints, which could be given in different ways, e.g. as a temporal logic specification or as stochastic constraints as we have done in [43].

Let  $\mathcal{A}$  be a set of OPAs. We define  $\mathcal{T} := \mathcal{T}_{deleg} \cup \mathcal{T}_{split} \cup \mathcal{T}_{refine} \cup \mathcal{T}_{assign}$  as:

$$\begin{split} \mathcal{T}_{deleg} &:= & \left\{ \mathsf{d}^{\psi}(\{ \mathsf{task}_{D[r]}^{a} \}, \{ \mathsf{task}_{D[r]}^{b} \}) \\ & \mid D \in \mathcal{D} \land r \in R(D) \land a, b \in \mathcal{A} \land a \neq b \land \psi \in \Psi \right. \right\} \\ \mathcal{T}_{split} &:= & \left\{ \mathsf{s}^{\psi}(\{ \mathsf{task}_{D[r_1, \dots, r_n]}^{a} \}, \{ \mathsf{task}_{D[r_1]}^{a}, \dots, \mathsf{task}_{D[r_n]}^{a} \}) \\ & \mid D \in \mathcal{D} \land \{ r_1, \dots, r_n \} \subseteq R(D) \land n > 1 \land a \in \mathcal{A} \land \psi \in \Psi \right. \right\} \\ \mathcal{T}_{refine} &:= & \left\{ \mathsf{r}^{\psi}(\{ \mathsf{task}_{D[r]}^{a} \}, \{ \mathsf{task}_{D'[R]}^{a} \}) \\ & \mid D, D' \in \mathcal{D} \land D \neq D' \land D[r] \simeq D'[R] \land a \in \mathcal{A} \land \psi \in \Psi \right\} \\ \mathcal{T}_{assign} &:= & \left\{ \mathsf{e}^{\psi}(\{ \mathsf{task}_{D[r]}^{a} \}, \emptyset) \mid D, D' \in \mathcal{D} \land a \in \mathcal{A} \land \psi \in \Psi \right\} \end{split}$$

The set of all team operators is  $\mathcal{T}$ . Each set  $T \subseteq \mathcal{T}$  of these operators induces a *SONAR-Organisation* Net N = (P, T, F) where  $P = {}^{\bullet}T \cup T^{\bullet}$ . The flow F is also encoded in T, i.e., for an operator  $t = \operatorname{op}^{\psi}(X, Y) \in \mathcal{T}$  where  $\operatorname{op} \in \{\mathsf{d}, \mathsf{s}, \mathsf{r}, \mathsf{e}\}$  is given by  ${}^{\bullet}t := X$  and  $t^{\bullet} := Y$ .

**Team-Formation within the SONAR-MAPE-Loop** The semantics of a SONAR organisational model [40] (i.e. the execution loop, called SONAR-MAPE-Loop) is defined on top of the operational semantics of Petri nets. Whenever a task is triggered we put a token on the corresponding place. For each task we fire transitions in the organisation net to handle the task: we delegate it to another OPA, refine the DWFN, split the role in the DWFN into sub-roles, or simply execute the task. The resulting structure is called a team. Formally, a team is a partial run [46] of the organisation net. Since the team operators assign execution are the only transitions with empty post-set, the empty marking is only reached whenever all sub-tasks are assigned. The underlying partial run is called a *team* G. From this team G we can derive the DWFN D(G) that defines the interaction among the involved agents. Then, the team develops a *team plan* – based on  $D_G$  – via negotiation.

$$D(G) = (PC_3[Prod_1] \parallel PC_3[Prod_2] \parallel PC[Cons])$$

The transition labelled with light bulbs denote 'inner' agents. They do not participate directly in the team interaction, but, since the constraints of inner agents have to be fulfilled in the execution of D(G), too, these agents fulfil a coordinating purpose.

### 3. The Life-Cycle of Sonar-Models

The life-cycle of SONAR-models is shown in Figure 3. We start with some *design goals*. These are used in the *design* phase to develop a *SONAR-organisation model* and to generate some *assumptions about the environment*, i.e. the CPS. From these assumptions we derive a deployment profile (called Cube-Protocols) during the *configure* phase. For example, the SONAR-model specifies the obligatory nature of the organisational constraints. This might range of a 'liberal-mode' (i.e., organisational constraints are only recommendations for agents), over 'liberal-with-reputation' (i.e., agents are free in their decisions, but their behaviour is evaluated and a mismatch reduces their reputation), to 'restrictive' (i.e., for deviation against the organisational constraints is observed – using a sliding time window – and a violation is blocked). For this example the different deployment variants come along with an increasing implementation complexity.

Both parts, organisation model and deployment configuration profile are used to instantiate the Org-MAS (here: the OPA/OMA-MAS) together with the *digital twin* model of the organisation (cf. [47]). The latter is used during the planning phase of our SONAR-MAPE-Loop [43] to predict the costs and benefits of applying transformation during the *self-adaption* step of the MAPE-loop (cf. also Sect. 5).

The Loop generates log data, that is used for two purposes. We might update our assumptions about the environment, which will *trigger a re-configuration* of the deployment profile, while leaving the organisation model untouched. The second part uses process mining techniques to update the organisation model itself, which might *trigger a complete re-design*.

Note that we indeed have a live cycle here, as these two phases – design and run-time – are intertwined: Obviously, the SONAR-model is used to instantiate the running multi-agent system; but, conversely, we adjust the deployment profile in a re-configuration phase or even start a complete re-design phase of the SONAR-model.

The areas denoted RC1 to RC4 define research challenges for organisation-based MAS for CPS as identified in previous work [21]. From the life-cycle one observes that a SONAR-model is modified at different stages: during the design phase (cf. RC3) and during the re-design triggered by organisational mining (cf. RC1). Additionally, the SONAR-model is used as a part of the deployment stage (cf. RC4). The deployed SONAR-model is also object to transformations executed in a self-adaptive way (cf. RC2). These transformations are executed within the SONAR-MAPE-loop [43]. We study these transformations in more detail in Section 4.

Note that the research challenges nicely correspond to aspects of the SONAR live cycle. Therefore, we claim that the understanding of this live cycle is crucial to tackle our challenges.

### 4. Transformation of SONAR-Models by Refinement

The most elementary way to think about designing or re-designing a SONAR-model is given by *model transformations*, especially refinement (needed in the design and in the re-design phase) and abstraction (needed in the re-design phase mainly). We can identify transformations in all components of a SONAR-model. These transformations establish the basic transformations used in the self-adaption within the



Figure 3: The Life-Cycle of SONAR-Models

SONAR-MAPE-loop. Note that there is a notion of an organisation of being well-formed [40] and we require these transformations to *preserve* well-formedness.

For each aspect of the specification  $Org = (N, \mathcal{O}, \mathcal{R}, \mathcal{D})$  we have specific operations (the complete list is given in Figure 4). Note that in practice these transformations are triggered by different activities: A transformation of the observables  $\phi$  or positions  $\mathcal{O}$  is mainly triggered by changes in the underlying CPS; transformations of the DWFN in  $\mathcal{D}$  are usually triggered by process mining; and the change of the Network N (including modification of the constraints  $\psi(t)$ ) is triggered by self-observations of the Org-MAS within the MAPE-Loop.

Refinements are invertible by nature (maybe together with some book-keeping mechanism). Therefore, refinement is not only uni-directional from an ever more detailed model. Our approach also includes the aspects that whenever we have to undo a lot of 'older' refinement operations of the past before we can apply 'new' ones, i.e. whenever we have the current organisation  $Org = (\tau_n \circ \cdots \circ \tau_1)(Org_0)$ , then the transformation gets more involved and has the form:

$$\hat{\tau}(Org) = (\tau'_m \circ \cdots \circ \tau'_1)(Org) = \left((\tau'_m \circ \cdots \circ \tau'_1) \circ (\tau_1^{-1} \circ \cdots \circ \tau_n^{-1})\right)(Org_0)$$

We cannot give a detailed formal specification of all these transformations given in Fig 4. We will provide a precise notion of an organisation model as an abstract data type together with the specification of all the refinements mentioned here as operations expressed in Maude [48] in future work.

- 1. Observables  $\phi$  and Positions  ${\cal O}$ 
  - a) take/loose responsibility for tasks
  - b) create/delete observables:  $\phi_{\mathbb{B}}$  and  $\phi_{\mathbb{R}}$
  - c) create/delete/split/join OPA's net parts
  - d) changes access rights to resources

Note: These transformations in the Org-MAS are mainly triggered by the underlying CPS-MAS.

- 2. Interaction Protocols/Workflows  ${\cal D}$ 
  - a) Refine/Abstract workflow net D
  - b) Modify role partition  ${\cal R}$  on activities

Note: These transformations in the Org-MAS are mainly triggered by the *organisation mining* (cf. RC1). 3. Organisational Network (constraint  $\psi$  modification):

- a) Modify the constraint  $\psi$  of an operation
- b) Escalate constraint  $\psi$  up- or downwards the network hierarchy

Note: These transformations in the Org-MAS are mainly triggered by self-observations of the Org-MAS. 4. Organisational Network (structural modifications of N):

- a) add/delete a task delegation operation  $d^{\psi}(\{task_{D[r]}^{a}\}, \{task_{D[r]}^{b}\})$
- b) add/delete a task split operation  $s^{\psi}(\{ task_{D[r_1,...,r_n]}^a\}, \{ task_{D[r_1]}^a, \ldots, task_{D[r_n]}^a\})$
- c) add/delete a task refinement operation  $r^{\psi}(\{ task_{D[r]}^{a}\}, \{ task_{D'[R]}^{a}\})$  for a refinement D'
- d) add/delete a task assignment operation  $e^{\psi}(\{task_{D[r]}^{a}\}, \emptyset)$  (provided that the OPA has the required capabilities/rights for the role r)

Note: These transformations in the Org-MAS are also triggered by self-observations of the Org-MAS, but usually are carried out with a lesser frequency.

Figure 4: Elementary Transformations of a SONAR-Model

## 5. Adaption of SONAR-Models as a Self-X Property

Our model allows for a goal-directed adaptation within the SONAR-MAPE-loop [43], which we coin as *Self-Modification at Run-Time (SM@RT)* in the following.

#### 5.1. Transformations as Second-Order Teamwork

As a special feature the execution of the team-plan may involve *transformation* statements, which modify the SONAR-model at run-time and therefore enable the *cooperative*, *self-organised adaption* of the organisation:

 $Organisation \rightarrow Team \rightarrow Plan \rightarrow Transformation \rightarrow new Organisation \rightarrow ...$ 

The formalism of SONAR [26] defines a notion of well-formedness to guarantee that each task may be handled by at least one team etc. Additionally, transformations in well-formed organisations preserve the well-formedness.

Our SONAR framework allows us to measure costs and benefits of a transformation. The set of all possible transformations defines a search space. Obviously, the search space considering all possible compositions of atomic transformations is much to big for an exhaustive search and we need more information to overcome this complexity.

Assume that all transformations are considered with decreasing priority in the cost-benefit reasoning:  $\hat{\tau}_1, \hat{\tau}_2, \ldots$  For SONAR, this order is given implicitly by probabilities assigned to teams and second-order workflows, since probabilities on team operators generate a probability on teams. Every team D defines a team workflow D(G) and the probabilities within the workflow induce a probability on interaction traces. Since every trace of a second-order workflow describes a transformation, we obtain a probability over transformations. Therefore, the enumeration order of transformations is simply from high to low probability.

#### 5.2. Costs and Benefits of a Transformation

Firstly, we have a *cost* measure of transformations. In SONAR a transformation is carried out at run-time using the standard teamwork; the main difference is that the team executes a *second-order DwFN*, where the actions attached to transitions execute atomic transformation operations as given in Section 4. A transformation is composed similarly to other interaction protocols using process operators (like sequence  $(\tau_1; \tau_2)$ , and-split  $(\tau_1 || \tau_2)$ , or-split  $(\tau_1 + \tau_2)$ , etc.) and atomic transformations. We denote these transformations as a complex composition in the following form:

$$\hat{\tau} = (\tau_2; \tau_5) \| (\tau_4 + (\tau_4; \tau_6)) \|$$

The costs of such a transformation  $\hat{\tau}$  could be measured by the number of transformation steps  $|\hat{\tau}|$  to be performed:

$$costs(\hat{\tau}) = const \cdot |\hat{\tau}| \tag{1}$$

This measurement treats all transformation steps uniformly; the approach can be generalised to the weighted sum of all transformation steps whenever we can assign individual costs to transformation steps.

As discussed above we will use process mining techniques to identify those transformations that are suitable to bring the current organisational model 'closer' to the observed behaviour as we assume that the observed behaviour contains valuable hints for the organisation originating from the OMA's reasoning processes. This mining will initialise the probabilities of the second-order operators and second-order workflows.<sup>1</sup> As our basic transformations are mainly refinements and their counterpart abstractions, we coin this idea under the slogan *Mine2Refine*.

Secondly, we use a *benefit* measure, which is the estimated average evaluation of the quantified observables, i.e., the KPIs  $\phi^{\mathbb{R}}(state)$ . Typical indicators include execution or production time, energy or resources needed, quality of the process, etc. We obtain the benefit using a simulation of a digital twin of the organisation model at run-time within the SONAR-MAPE-Loop [43]. The KPIs are defined by the underlying CPS.

Assume a SONAR-organisation model Org, which is the input of the analysis sketched in Figure 5. The organisation model is basis to deploy the SONAR-MAPE-Loop. Within the planning phase of the loop we use a digital twin of the organisation where the decision logic of each OMA is modelled as a stochastic distribution over actions. The twin model of the organisation is fed into a twin model of the Loop, which is configured by meta-parameters (cf. [43, 47]. The adaption dynamics of this *SONAR-MAPE-Loop twin* is considered in the analysis phase, as sketched in the lower part of Figure 5. Here, we extend a stochastic distribution over tasks to a distribution over teams; together with the stochastic behaviour model of OMAs (which is part of our digital twin model), we can compute the distribution over team protocol processes. To simplify the description of benefits, we make the assumption that the effect of actuators on KPIs is additive in the following. Therefore the concurrently running teams do not interfere and we can calculate the estimated KPI for the current organisation simply as a stochastic superposition of the KPI effect of all teams. This estimation of the effect on the *n* KPIs (key performance indicators) is the evaluation  $val(Org) \in \mathbb{R}^n$  of the current organisation.

This simulation evaluates the KPI for the resulting organisation  $\hat{\tau}(Org)$  (as described in [44]) as well as the complete dynamics including future transformations as well (cf. [30]).<sup>2</sup> This simulation also anticipates the uncertainty about the environment and the stochastic distributions used [49].<sup>3</sup>

The *absolute benefit* of changing *Org* to *Org'* is the vector:

$$benefit(Org, Org') := val(Org') - val(Org)$$
<sup>(2)</sup>

<sup>&</sup>lt;sup>1</sup>Note that it is possible to change these probabilities at run-time, too, by the use of second-order operators, teams, and workflows. This second-order transformations will require a third-order teamwork.

<sup>&</sup>lt;sup>2</sup>The study of adaption state spaces (like in [30]) in combination with stochastic distributions of adaptions is subject to ongoing research. The usage of the rewriting engine Maude [48] to support analysis is studied in [32, 33].

<sup>&</sup>lt;sup>3</sup>The same analysis technique is also used during the design and the deployment phase to select a deployment profile that corresponds to the best setting of meta-parameters used in the twin model [47].



Figure 5: Stochastic Analysis of SONAR-Models using the Twin of the SONAR-MAPE-Loop

As we define the benefit as a vector, a transformation may be an improvement in some dimensions, while being a worsening in others. This leads to the question under which circumstances a transformation might be considered as acceptable among the agents.

#### 5.3. Commonly Acceptable Transformation

As the team agents may have different partial views on the system, their evaluation may vary. Therefore, we only consider the sign of the absolute benefit to obtain a more stable measurement. For a given indicator k the sign of *benefit*(Org, Org')(k) indicates whether the adaption increases the KPI k:

$$\Delta(Org, Org')(k) := \operatorname{sgn}(benefit(Org, Org')(k))$$
(3)

Therefore, the vector  $\Delta(Org, Org') \in \{-1, 0, +1\}^n$  characterises the adaption w.r.t. all *n* KPI. We call  $\Delta(Org, Org')$  the *trend vector* of the adaption. Usually, an adaption leads to non-uniform changes, where the value increases for some indicators, while decreases for other, which leads to incomparable vectors (i.e. we have neither  $\Delta(Org, Org') \leq \mathbf{0}$  nor  $\Delta(Org, Org') \geq \mathbf{0}$ ).

The simplest adaptations from Org to Org' are those where the OPAs have a common understanding that effect is positive on all KPI, i.e., whenever  $\Delta(Org, Org') \geq 0$ . In this case the adaption is *indisputably acceptable* among the OPAs.

Whenever not all effects are positive, i.e.,  $\Delta(Org, Org') \geq \mathbf{0}$ , then the agents have to *negotiate* whether the adaptation is discarded. Whenever they discard it, the next adaption candidate Org' has to be considered; whenever the OPAs agree not to discard it, then they start to negotiate, i.e., they have to define additional relationships on incomparable vectors.

To establish a common believe that the adaption benefit should be considered as an improvement we have to re-interpret KPIs. This is done by a mapping r from the KPI vector to a lower dimensional one.

 $\frac{(0,0)}{(1,3)}$ 

A *KPI reduction* is a mapping  $r : \mathbb{R}^n \to \mathbb{R}^m$  that preserves the existing ordering (but may add them):

$$\mathbf{v}_1 \le \mathbf{v}_2 \Longrightarrow r(\mathbf{v}_1) \le r(\mathbf{v}_2) \tag{4}$$

Intuitively, whenever an OPA considers an adaption to be be an improvement, this will remain true under any reduction.

We extend the notation of benefit and trend to reductions:

$$benefit^{r}(Org, Org') := r(val(Org')) - r(val(Org))$$
(5)

$$\Delta^{r}(Org, Org')(k) := \operatorname{sgn}(benefit^{r}(Org, Org')(k))$$
(6)

The former notations could be considered as special cases, where the reduction is the identity function. A KPI reduction r with  $\Delta^r(Org, Org') \ge \mathbf{0}$  is called a *compromise*. Using this terminology negotiation among the OPAs is equivalent to agree upon a compromise r. Such a compromise must always exists as shown by the following example.

**Example** The most common approach to obtain a compromise is to define exchange rates  $a_i$  for each KPI, i.e. we have a money-like mechanism to compare KPI. This is expressed by a *linear* mapping:  $r_{\text{money}}(\mathbf{v}) = \mathbf{a} \cdot \mathbf{v} = \sum_{i=1}^{n} a_1 \cdot v_1 + \cdots + a_n \cdot v_n$ , where we require that the exchange rates are positive:  $\mathbf{a} = (a_1, \ldots, a_n) \ge \mathbf{0}$  to ensure monotonicity. In this case  $r_{\text{money}} : \mathbb{R}^n \to \mathbb{R}^m$  with m = 1, i.e., the reduction is maximal w.r.t. to the resulting dimension of the KPI vector.

So, we implement self-adaption as a distributed search among the OPAs for a transformation that has the best *balance* of costs and benefits. Of course, in practice, this self-organised transformation is restricted by the bounds of the agents, i.e. limited or outdated data, imprecise predictions about the environment, restricted computational power, etc.

### 6. Example: A Coordination Game

The following example for a transformation in SONAR is based on the battle-of-sexes scenario, which is well-known in game theory. Two agents must choose between two actions, labelled as *a* and *b*. They receive a positive reward if they choose the same action and zero otherwise. In this game, the first agent prefers action *a*, while the second prefers *b*. If we assume that the reward for the preferred outcome is three times higher than for the other, then the game is specified by the following pay-off matrix.

A game theoretical analysis shows that we have two Nash equilibria for pure	11
strategies: $(a, a)$ and $(b, b)$ . It can be shown that this game has a <i>unique</i>	
equilibrium for <i>mixed strategy</i> , where both agents choose their preferred	
action $p = 75\%$ of the time.	<u>b</u> (

The game is modelled by a very simple multi-party workflow net (shown in Fig. 6). We use RENEW syntax [50] to implement the nets. The sources are available at https://github.com/koehler-bussmeier/ bos. For each role (shown on the left and on the right side) the choice between the two options a and b is resolved by the agent's decision logic  $\delta$  of the agents and the organisation configuration  $\mu$ . More concretely, the choice is resolved randomly by the RENEW-inscription:

prob = 
$$c^*p + (1-c)^*q$$
; b = Math.random() < prob

Here, p denotes a probability as specified by the organisational  $\mu$  and q is a probability as specified by the member agents' decision logic  $\delta$ ; these two probabilities are combined using the so-called *organisation impact c*, which is a meta parameter of the twin of our SONAR-MAPE-Loop. These values are obtained by the calls this:getProb(task,p,q); this:getCorg(c).



**Figure 6:** Workflow Net with stochastic choices: Battle-of-the-Sexes (The shaded transition will be deleted by the second-order workflow given in Fig. 8.)

**The SONAR-Organisation** The situation described is known as a coordination game because the agents would benefit from coordinating their actions in advance. In this scenario, the social welfare, which is the sum of the individual pay-offs, is 3+1 = 1+3 = 4 in both cases. However, in uncoordinated games, a mixed strategy is the best choice, and agents only coordinate in  $0.75 \cdot 0.25 + 0.75 \cdot 0.25 = 0.375$  of the cases, leading to an expected pay-off of  $(0.75 \cdot 0.25 + 0.75 \cdot 0.25) \cdot 4 = 1.5$ . The ratio  $\frac{3+1}{1.5} = 2.66..$ , called *price-of-anarchy*, measures the need for an external coordination mechanism.



**Figure 7:** The SONAR-Organisation (The shaded nodes are those elements that are added by the second-order workflow given in Fig. 8.)

For MAS, this mechanism is termed an *organisation* [14]. The SONAR-Org-MAS of this scenario is depicted in Fig. 7. The primary purpose of the SONAR-Org-Model is to assemble a team in response to certain triggers. It determines the workflow net used for the response and the involved agents and sets constraints on the agents' decisions, represented by parameters  $\mu$  and  $\delta$ .

The Org-Model specifies three organisational agents (OPAs):  $O_0$ ,  $O_1$ , and  $O_2$ . The set of operators belonging to each OPA is shown by the labelled rectangles in the organisation net. The organisation manages two tasks which are the places on the top of Fig. 7. Tasks are generated by the environment by the transitions with the synchronisation channel :start(trigger,wfnName) on the top. It is the OPA  $O_0$ that will react to both triggers. The first task triggers the team-formation process for the battle-of-sexes interaction protocol P. The protocol P defines the interaction of two roles  $R_1$  and  $R_2$ . The team formation for this task is specified by the bold subnet on the left, which specifies that role  $R_1$  is finally assigned to the agent  $O_1$  and  $R_2$  to  $O_2$ . (Please ignore the shaded subnet in the middle for the moment, since it is not there initially, but will be added as the effect of the transformation.)

164-181



**Figure 8:** Second-order WFN: Team-based modification of the workflow given in Fig. 6 and the organisation given in Fig. 7

**Second-Order Workflow for Adaptation** Note that the agents can't do any better than to choose the mixed strategy of a Nash-equilibrium. But even then they have to pay a price, which is called the price-of-anarchy as we have seen above. Therefore, whenever the agents already are in a Nash-equilibrium then individual learning is not a source of improvement anymore – instead, we have to modify the organisational structure.

In our model the organisation may define constraints that drive the agents choices into a desired (i.e., cooperative) state. But this depends on the *organisational impact* mentioned in Sect. 3. Whenever the impact is low or moderate (which would correspond to the 'recommendation' or the 'reputation' mode mentioned above) then the only option of the organisation to improve the pay-off is a regulation of the available options, i.e., a structural modification of the interaction protocol.

The right side of the organisation in Fig. 7 handles a another task that triggers the adaptation. It initiates the team formation for a second-order workflow (cf. Fig. 8), which alters the role fragment so that the second agent always chooses option a (i.e., we force the second agent to deviate from its current, optimal mixed strategy). The resulting workflow net is named P' in the following. The construction essentially removes the dashed arcs and the right-most transition and re-normalises the probabilities in the workflow net of Fig. 6.

This structural modification is formalised by the subnet modify WFN. Furthermore, this second-order

workflow extends the original organisation model. The shaded area of the organisation net in Fig. 7 shows the nodes that will be added by the second-order protocol. The modification is formalised in the lower subnet (modify Organisation Model). The effect of this modification is that now there are two alternative teams that may be generated for the left task: one team will execute the original P and the new one P'. Note that for both teams it will be an interaction of the OPAs  $O_1$  and  $O_2$ .

Since we add new nodes to each agent block the second-order protocol has three roles and these are assigned to the three agents  $O_0$ ,  $O_1$ , and  $O_2$ . This is necessary since a modification of the subnet  $O_i$  the organisation net in Fig. 7 has to be executed by the OPA  $O_i$  itself, since no other agent is allowed to manipulate the structure of  $O_i$ . The second-order workflow net also has to synchronise the transformation steps, e.g. we have to add the operator for a task delegation from  $O_0$  to  $O_1$  (formally: the transition  $d^{\psi}(\{ task_{P'[R_1]}^{O_0} \}, \{ task_{P'[R_1]}^{O_1} \})$ ) into the subnet of  $O_0$  before  $O_1$  adds the operator for task assignment (formally: the transition  $e^{\psi}(\{ task_{P'[R_1]}^{O_1} \}, \emptyset)$ ) to its subnet. For this transformation the constraint  $\psi$  defines a probability how the conflict on the task place  $task_{P[R_1,R_2]}^{O_0}$  is resolved. For a better interpretation of the simulation results we set the probability to choose the new team is 100%.

The transformation defined by the subnet modify Organisation Model is constructed by elementary transformations as given in Figure 4 and has the form:

$$\begin{split} \pi_{org} &= & add \left( \mathsf{r}^{\psi}(\{ \mathsf{task}_{P[R_1,R_2]}^{O_0} \}, \{ \mathsf{task}_{P'[R_1,R_2]}^{O_0} \}) \right); \\ & add \left( \mathsf{s}^{\psi}(\{ \mathsf{task}_{P'[R_1,R_2]}^{O_0} \}, \{ \mathsf{task}_{P'[R_1]}^{O_0} \}, \mathsf{task}_{P'[R_2]}^{O_0} \}) \right); \\ & \left( \left( add \left( \mathsf{d}^{\psi}(\{ \mathsf{task}_{P'[R_1]}^{O_0} \}, \{ \mathsf{task}_{P'[R_1]}^{O_1} \}) \right); & add \left( \mathsf{e}^{\psi}(\{ \mathsf{task}_{P'[R_1]}^{O_1} \}, \emptyset) \right) \right) \\ & \parallel \\ & \left( add \left( \mathsf{d}^{\psi}(\{ \mathsf{task}_{P'[R_2]}^{O_0} \}, \{ \mathsf{task}_{P'[R_2]}^{O_2} \}) \right); & add \left( \mathsf{e}^{\psi}(\{ \mathsf{task}_{P'[R_2]}^{O_2} \}, \emptyset) \right) \right) \end{split} \end{split}$$

**Analysis of the Model** In the following we we will perform a stochastic analysis of our SONAR-model (i.e. an analysis of the SONAR-MAPE-Loop) as sketched in Figure 5. For a typical cyber-physical systems (CPS) we would evaluate our models using key performance indicators like throughput, occupation, etc. To simplify the presentation we use the price-of-anarchy as our only performance indicator. Therefore, the benefit of the adaptation modelled in this example is the change in this value and the costs are measured by the number of transitions in the second-order protocol in Fig. 8. As we have already a one-dimensional KPI vector there is no need for a reduction in this simple scenario.

During the planning phase of our MAPE-Loop we use a stochastic simulation of our model to predict the effects of the adaptation. The sources are available at https://github.com/koehler-bussmeier/bos. In the simulation, we have three phases. Firstly, we initiated the first-order battle-of-sexes workflow P(Fig. 6) 100 times to ensure some statistical stability. Secondly, we generated the adaptation tasks which triggers the team-based execution of the second-order workflow in Fig. 8. This adaptation results in a modified version P' of the first-order workflow P, where the second agent persistently chooses option a, and in an adjustment in the organisation so that this protocol would always be adopted from that point on. As explained above after the transformation the organisation is in a state such that the new protocol P' is always chosen. Finally, we generate another 100 trigger events and the protocol P' is executed the same number of times.

Typically, we would repeat the simulation multiple times to establish error ranges, but for this illustrative scenario, we decided to omit this step.

outcome	before and	after adaptation	before (expected)	after (expected)
(a,a)	20%	78%	18,75%	75%
(a,b) or $(b,a)$	62%	22%	62.50%	25%
(b,b)	18%	0%	18,75%	0%

Table 1

Simulation Results (left) and Expected Values (right) for the Pay-Off before and after our Adaptation

The simulation results (given in the left columns of Table 1) are in good alignment with the analytical results (right columns): In the initial configuration of this example, we can expect (a, a) in  $0.75 \cdot 0.25 = 18,75\%$  of the cases, with an outcome of (3, 1); for symmetry reasons, we have the same probability for (b, b). In 62.5% of times, the agents will choose opposite options.

After the adaptation has taken place, we expect that in  $0.75 \cdot 1$  of interactions, the agents will play (a, a) and in  $0.25 \cdot 1$  of interactions, the agents will play (b, a). This leads to the expected social welfare of  $0.75 \cdot (3+1) + 0.25 \cdot (0+0) = 3$ . So, the new price-of-anarchy is  $\frac{3+1}{3} = 1.33$ ..., which is a substantial improvement from the initial value of  $\frac{3+1}{1.5} = 2.66$ ... We have: benefit(Org, Org') = 2.66.. - 1.33.. = 1.33... > 0 and  $costs(\hat{\tau}) = const \cdot |\hat{\tau}| = const \cdot 8$ .

We could also consider an extension of the organisation model where we generate a new protocol P'' from P, which is obtained by deleting the option a for the second agent. The resulting second-order protocol looks almost identical to the presented one and therefore the transformation costs are identical, too.

On the one hand, this adaptation removes an option and therefore the coordination becomes simpler. On the other hand, the coordinated outcome is less desirable. A similar calculation shows that restricting the agents' options is not beneficial: Removing option a for the second agent results in a welfare of  $0.25 \cdot (3+1) + 0.75 \cdot (0+0) = 1$ . This leads to a new price-of-anarchy of  $\frac{3+1}{1} = 4$ . In this case the benefit is negative:

$$benefit(Org, Org') = 2.66.. - 4.0 = -1.33.. < 0$$

So, we can observe that simply enforcing some coordinated behaviour is not a good idea unless we have considered the concrete change for the expected pay-offs.

### 7. Conclusion

In this paper, we addressed four central research challenges for CPS (which we identified in a previous publication) from the perspective of the life cycle of SONAR-models. Here, we have shown that the life cycle addresses all challenges in a uniform and integrated way. Remarkably, design phase and execution are intertwined in a cyclic way: On the one hand, a SONAR-model is used to instantiate the running multi-agent system; but, conversely, we use process mining techniques to adjust the deployment at run-time.

A special focus was put on basic transformation operations as they are used throughout the life cycle, mainly during design time, but also as the basic operations during the self-adaption, which is performed at run-time as part of the SONAR-MAPE-Loop. In current work, we integrate the transformation steps into our development tools. Organisations are represented by an abstract data type and the refinements are specified as algebraic operations. We are working on a prototype variant expressed in Maude [48].

In future work we would like to study to which extend it is possible to avoid the 'expensive' analysis of the adaption state space and replace it by a static analysis of the organisation net itself.

In another thread of research, we consider the evaluation of MAS designs per-se, i.e., the organisation models themselves. As stated above, for SONAR the probability distributions on second-order teams induces a probability over transformations. Obviously, a well-designed organisation is supposed to generate those transformations that are 'cheap' and 'beneficial' at the same time with a higher probability. This idea could be used to evaluate the design space of all organisations, i.e., an organisation is better than another one whenever is has a better expected cost-benefit ratio in average. In future work we like to interpret the cost-benefit ratio as the reward of a transformation, which will turn the adaption state space (where states are organisation models and transformations are transitions) into a Markov Decision Process (MDP).

### References

 E. R. Griffor, C. Greer, D. A. Wollman, M. J. Burns, Framework for cyber-physical systems: volume 1, overview (2017). doi:10.6028/nist.sp.1500-201.

- [2] P. Leitao, S. Karnouskos, L. Ribeiro, J. Lee, T. Strasser, A. W. Colombo, Smart agents in industrial cyber-physical systems, Proceedings of the IEEE 104 (2016) 1086–1101.
- [3] J. Sudeikat, M. Köhler-Bußmeier, On combining domain modeling and organizational modeling for developing adaptive cyber-physical systems, in: ICAART'22, 2022.
- [4] P. Leitao, S. Karnouskos, Industrial Agents: Emerging Applications of Software Agents in Industry, 1st ed., Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 2015.
- [5] G. Weiß (Ed.), Multiagent systems: A modern approach to Distributed Artificial Intelligence, MIT Press, 1999.
- [6] T. Pulikottil, L. A. Estrada-Jimenez, H. Ur Rehman, F. Mo, S. Nikghadam-Hojjati, J. Barata, Agentbased manufacturing – review and expert evaluation, The International Journal of Advanced Manufacturing Technology 127 (2023) 2151–2180. doi:10.1007/s00170-023-11517-8.
- [7] R. Yao, Y. Hu, L. Varga, Applications of agent-based methods in multi-energy systems—a systematic literature review, Energies 16 (2023) 2456. doi:10.3390/en16052456.
- [8] L. A. C. Salazar, D. Ryashentseva, A. Lüder, B. Vogel-Heuser, Cyber-physical production systems architecture based on multi-agent's design pattern—comparison of selected approaches mapping four agent patterns, The International Journal of Advanced Manufacturing Technology 105 (2019) 4005–4034. doi:10.1007/s00170-019-03800-4.
- [9] A. Lopez, O. Casquero, M. Marcos, Design patterns for the implementation of industrial agent-based aass (2021). doi:10.1109/ICPS49255.2021.9468129.
- [10] A. López, O. Casquero, E. Estévez, A. Armentia, D. Orive, M. Marcos, An industrial agent-based customizable platform for i4.0 manufacturing systems, Computers in Industry 146 (2023) 103859. doi:10.1016/j.compind.2023.103859.
- [11] A. Lober, J. Lehmann, J. Reichwald, L. Ollinger, H. Baumgärtel, Flexible skill-based production systems through novel opc ua design approaches, IFAC-PapersOnLine 56 (2023) 3654–3659. doi:10.1016/j.ifacol.2023.10.1529.
- [12] IEEE Standards Committee, J. Olszewska, IEEE recommended practice for industrial agents: Integration of software agents and low-level automation functions: IEEE standard 2660.1-2020, IEEE Std 2660.1-2020 (2021) 1–43. doi:10.1109/IEEESTD.2021.9340089.
- [13] K. M. Carley, L. Gasser, Computational organisation theory, in: [5], 1999, pp. 229–330.
- [14] V. Dignum, J. Padget, Multiagent organizations, in: G. Weiss (Ed.), Multiagent Systems, 2nd ed., Intelligent Robotics; Autonomous Agents Series, MIT Press, 2013, pp. 51–98.
- [15] B. H. C. Cheng, R. de Lemos, H. Giese, P. Inverardi, J. Magee, J. Andersson, B. Becker, N. Bencomo, Y. Brun, B. Cukic, G. Di Marzo Serugendo, S. Dustdar, A. Finkelstein, C. Gacek, K. Geihs, V. Grassi, G. Karsai, H. M. Kienle, J. Kramer, M. Litoiu, S. Malek, R. Mirandola, H. A. Müller, S. Park, M. Shaw, M. Tichy, M. Tivoli, D. Weyns, J. Whittle, Software engineering for self-adaptive systems: A research roadmap, in: B. H. C. Cheng, R. de Lemos, H. Giese, P. Inverardi, J. Magee (Eds.), Software Engineering for Self-Adaptive Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 1–26.
- [16] D. Weyns, I. Gerostathopoulos, N. Abbas, J. Andersson, S. Biffl, P. Brada, T. Bures, A. D. Salle, M. Galster, P. Lago, G. Lewis, M. Litoiu, A. Musil, J. Musil, P. Patros, P. Pelliccione, Self-adaptation in industry: A survey, ACM Transactions on Autonomous and Adaptive Systems 18 (2023) 1–44. doi:10.1145/3589227.
- [17] J. Kephart, D. Chess, The vision of autonomic computing, Computer 36 (2003) 41–50. doi:10. 1109/mc.2003.1160055.
- [18] D. Weyns, An Introduction to Self-Adaptive Systems: A Contemporary Software Engineering Perspective, John Wiley & Sons Ltd, 2020.
- [19] N. Bencomo, R. France, B. Cheng, U. Aßmann (Eds.), Models@run.time: foundations, applications, and roadmaps, Lecture Notes in Computer Science, Springer, Germany, 2014. doi:10.1007/ 978-3-319-08915-7.
- [20] N. Bencomo, S. Götz, H. Song, Models@run.time: a guided tour of the state of the art and research challenges, Software & Systems Modeling 18 (2019) 3049–3082.
- [21] J. Sudeikat, M. Köhler-Bußmeier, Controlled run-time adaptivity in industrial agent systems:

challenges and research prospects, in: Informatik 2023 - Designing Futures: Zukünfte gestalten, volume 337 of *Lecture Notes in Informatics*, 2023, pp. 2003–2014.

- [22] W. M. P. van der Aalst, Process Mining: Data Science in Action, 2 ed., Springer, Heidelberg, 2016. doi:10.1007/978-3-662-49851-4.
- [23] D. Schuster, S. J. van Zelst, W. M. P. van der Aalst, Freezing sub-models during incremental process discovery, in: A. Ghose, J. Horkoff, V. E. Silva Souza, J. Parsons, J. Evermann (Eds.), Conceptual Modeling, Springer International Publishing, Cham, 2021, pp. 14–24.
- [24] D. Schuster, S. J. van Zelst, W. M. P. van der Aalst, Incremental discovery of hierarchical process models, in: F. Dalpiaz, J. Zdravkovic, P. Loucopoulos (Eds.), Research Challenges in Information Science, Springer International Publishing, Cham, 2020, pp. 417–433.
- [25] M. Köhler-Bußmeier, M. Wester-Ebbinghaus, Sonar\*: A multi-agent infrastructure for active application architectures and inter-organisational information systems, in: L. Braubach, W. van der Hoek, P. Petta, A. Pokahr (Eds.), Conference on Multi-Agent System Technologies, MATES 2009, volume 5774 of *Lecture Notes in Artificial Intelligence*, 2009, pp. 248–257.
- [26] M. Köhler-Bußmeier, M. Wester-Ebbinghaus, D. Moldt, A formal model for organisational structures behind process-aware information systems, Transactions on Petri Nets and Other Models of Concurrency. Special Issue on Concurrency in Process-Aware Information Systems 5460 (2009) 98–114.
- [27] C. Krupitzer, F. M. Roth, S. VanSyckel, G. Schiele, C. Becker, A survey on engineering approaches for self-adaptive systems, Pervasive Mob. Comput. 17 (2015) 184–206. doi:10.1016/j.pmcj. 2014.09.009.
- [28] J. V. D. Donckt, D. Weyns, M. U. Iftikhar, S. S. Buqar, Effective decision making in self-adaptive systems using cost-benefit analysis at runtime and online learning of adaptation spaces, in: Best papers of ENASE'18, Springer-Verlag, 2018.
- [29] M. Köhler-Bußmeier, Restricting Hornets to support adaptive systems, in: W. van der Aalst, E. Best (Eds.), PETRI NETS 2017, Lecture Notes in Computer Science, Springer-Verlag, 2017.
- [30] M. Köhler-Bußmeier, H. Rölke, Analysing adaption processes of Hornets, Transactions on Petri Nets and Other Models of Concurrency XVII 14150 (2023).
- [31] M. Köhler-Bußmeier, Hornets: Nets within nets combined with net algebra, in: K. Wolf, G. Franceschinis (Eds.), International Conference on Application and Theory of Petri Nets (ICATPN'2009), volume 5606 of *Lecture Notes in Computer Science*, Springer-Verlag, 2009, pp. 243–262.
- [32] L. Capra, M. Köhler-Bußmeier, Modelling adaptive systems with nets-within-nets in maude, in: Proceedings of the 18th International Conference on Evaluation of Novel Approaches to Software Engineering - ENASE, Prague, Czech Republic, INSTICC, SciTePress, 2023, pp. 487–496. doi:10.5220/0011860000003464.
- [33] L. Capra, M. Köhler-Bußmeier, Modular rewritable Petri nets: an efficient model for dynamic distributed systems, Theoretical Computer Science 990 (2024) 114397. doi:https://doi.org/ 10.1016/j.tcs.2024.114397.
- [34] J. Ferber, O. Gutknecht, F. Michel, From agents to organizations: An organizational view of multiagent systems, in: P. Giorgini, J. P. Müller, J. Odell (Eds.), Agent-Oriented Software Engineering IV, volume 2935, 2003, pp. 214–230.
- [35] O. Boissier, J. F. Hübner, J. S. Sichman, Using the moise+ for a cooperative framework of MAS reorganisation, in: A. L. C. Bazzan, S. Labidi (Eds.), Advances in Artificial Intelligence - SBIA 2004, volume 3171 of *Lecture Notes in Computer Science*, Springer-Verlag, 2004, pp. 506–515.
- [36] V. Dignum, F. Dignum, J.-J. Meyer, An agent-mediated approach to the support of knowledge sharing in organizations, Knowledge Engineering Review 19 (2004) 147–174.
- [37] V. Dignum (Ed.), Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models, IGI Global, Information Science Reference, 2009.
- [38] Standards Committee of the IEEE Industrial Electronics Society, IEEE Recommended Practice for Industrial Agents: Integration of Software Agents and Low-Level Automation Functions, Technical Report EEE Std 2660.1<sup>™</sup>-2020, IEEE Industrial Electronics Society, 2020.
- [39] W. v. d. Aalst, Verification of workflow nets, in: P. Azeme, G. Balbo (Eds.), Application and theory

of Petri nets, volume 1248 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin Heidelberg New York, 1997, pp. 407–426.

- [40] M. Köhler, A formal model of multi-agent organisations, Fundamenta Informaticae 79 (2007) 415 430.
- [41] J. Sudeikat, M. Köhler-Bußmeier, Towards integrating multi-agent organizations in OPC UA for developing adaptive cyber-physical systems, in: D. Demmler, D. Krupka, H. Federrath (Eds.), 52. Jahrestagung der Gesellschaft für Informatik, INFORMATIK 2022, Informatik in den Naturwissenschaften, 26. - 30. September 2022, Hamburg, volume P-326 of *LNI*, Gesellschaft für Informatik, Bonn, 2022, pp. 1565–1570. doi:10.18420/inf2022\\_135.
- [42] M. Köhler-Bußmeier, J. Sudeikat, Defining adaption measures for organizational multi-agent systems, International Journal of Parallel, Emergent and Distributed Systems (2023).
- [43] M. Köhler-Bußmeier, J. Sudeikat, Studying the micro-macro-dynamics in MAPE-like adaption processes, in: 16th International Symposium on Intelligent Distributed Computing (IDC'23), Springer-Verlag, 2023. (to appear).
- [44] M. Köhler-Bußmeier, J. Sudeikat, Balance vs. contingency: Adaption measures for organizational multi-agent systems, in: K. Jander, L. Braubach, C. Badica (Eds.), 15th International Symposium on Intelligent Distributed Computing (IDC'22), volume 1089 of *Studies in Computational Intelligence*, Springer-Verlag, 2023, pp. 224–233.
- [45] W. Reisig, Petri Nets: An Introduction, Springer-Verlag, Heidelberg, 1985.
- [46] J. Esparza, K. Heljanko, Unfoldings A Partial-Order Approach to Model Checking, EATCS Monographs in Theoretical Computer Science, Springer-Verlag, 2008.
- [47] M. Köhler-Bußmeier, H. Rölke, Run-time analysis of adaption in multi-agent organisations, in: Petri Nets and Software Engineering 2024, PNSE'24, CEUR, 2024.
- [48] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, C. L. Talcott (Eds.), All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic, volume 4350 of *Lecture Notes in Computer Science*, Springer-Verlag, 2007.
- [49] M. Köhler-Bußmeier, H. Rölke, Petri-nets@run.time: Handling uncertainty during run-time adaptation using digital twins, in: Petri Nets and Software Engineering 2023, PNSE'23, volume 3430, CEUR, 2023, pp. 34–52. URL: http://ceur-ws.org/Vol-3430/.
- [50] O. Kummer, F. Wienberg, M. Duvigneau, J. Schumacher, M. Köhler, D. Moldt, H. Rölke, R. Valk, An extensible editor and simulation engine for Petri nets: Renew, in: J. Cortadella, W. Reisig (Eds.), International Conference on Application and Theory of Petri Nets 2004, volume 3099 of *Lecture Notes in Computer Science*, Springer-Verlag, 2004, pp. 484 493.