CEUR-WS.org/Vol-3730/paper14.pdf

Design and Run-Time Analysis of Self-Adaption for Multi-Agent Organisations

Michael Köhler-Bußmeier[1], Heiko Rölke[2]

[1]Hamburg University of Applied Sciences, Berliner Tor 7, D-20099 Hamburg, Germany
[2]University of Applied Science of the Grisons, Pulvermühlestrasse 57, CH-7000 Chur, Switzerland

Abstract

In this contribution we study organisation-centred multi-agent systems (Org-MAS), i.e., MAS where the agents are embedded into an explicitly modelled organisational structure with positions, roles, hierarchies etc. Whenever an Org-MAS adapts this happens at two levels: At the micro-level the agents learn and at the macro-level the organisation adapts its structure. These two learning processes are intertwined and this co-learning is known as the micro-macro-link in sociology.

In previous work, we have developed an execution engine for Org-MAS, called Sonar, specified using Nets-within-Nets, i.e., Petri nets which contain other Petri nets as tokens. The learning part of the engine defines the micro-macro-link of two interconnected MAPE-like (monitor, analyse, plan, execute) learning processes.

The Sonar-engine uses a digital twin of itself, called the Sonar-MAPE-Loop@run.time, during the planning of MAPE to predict benefits and costs of applicable adaptation steps to allow for a goal-directed Adaptation. In this paper we study the behavioural impact of the meta-parameters that are used by the digital twin for the prediction. Additionally, we explore the aspect that meta-parameters of the twin correspond to deployment variants of the organisation. Here, we demonstrate how to use our twin model to find a 'good' organisation design using a sweep through the parameter space.

Keywords

multi-agent systems, organisations, co-learning, MAPE-K, self-Adaptation, Petri-Nets@run.time

1. Introduction

In general, we are interested in the design and analysis of self-adaptive systems [1], more specifically, self-adaptive and self-organising multi-agent-systems (MAS) [2] in the area of cyber-physical systems (CPS) [3, 4]. Usually, a self-adaptation is embedded within a goal directed process where the system compares the costs against the benefits of a potential adaptation step before executing it. In most cases one also considers more than one adaptation candidate and executes the one that has a good absolute cost-benefit ratio among all the candidates [5].

The central concepts of MAS (like autonomy, rationality, and cooperation) are complemented by organisational aspects (like roles, norms, positions, protocols, etc.). This combination is known as *organisation-centered MAS* (Org-MAS) [6, 7]. This is used to guarantee a certain

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

michael.koehler-bussmeier@haw-hamburg.de (M. Köhler-Bußmeier); heiko.roelke@fhgr.ch (H. Rölke)
https://www.haw-hamburg.de/michael-koehler-bussmeier (M. Köhler-Bußmeier);
https://www.fhgr.ch/personen/person/roelke/ (H. Rölke)
0000-0002-3074-4145 (M. Köhler-Bußmeier); 0000-0002-9141-0886 (H. Rölke)

© 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).
CEUR Workshop Proceedings (CEUR-WS.org)

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073

level of coherence for a system of autonomous entities, the agents. In the following we study adaptivity for organisation-centred MAS in the context of our formalism Sonar [8, 9], a Petri-net based formalism.

The Sonar-Execution Engine An MAS specified with Sonar is executed by an execution engine [10, 11], specified as a Hornet [12] – a nets-within-nets formalism [13] extended by algebraic operations that allow for structural modifications of nets at run-time; the adaptation support in Sonar exploits this feature. The execution engine itself is executed using the reference net simulator Renew [14].

This execution engine supports adaptation via a *monitor-analyse-plan-execute* loop, MAPE for short [5]. For the Sonar execution engine, we call these steps the *Sonar-MAPE-loop* [11]. The Sonar-MAPE-loop integrates the macro-level view, i.e., the organisation, with the micro-level view, i.e., the agents and their decision logic. The integration is also known a the micro-macro-link and is considered as a major ingredient to understand the emergent dynamics of complex MAS [15, 16].

For the adaptation of the Sonar organisation we have several means: We could adjust the organisational *structure* by adding new protocols or by structural modifications of the organisation net, i.e., adding delegation options, etc. A second approach is to modify the *state* μ of the organisation that controls team formation processes: For each task to be performed by an MAS an organisation model defines a whole set of possible teams. Which team is chosen for a concrete task at run-time depends on the organisational state. In this paper we concentrate on the second, simpler kind of modification of the organisational state; the structural adaptation is subject to ongoing work.

Using a Digital Twin during the Planning Step In our Sonar-MAPE-Loop we monitor the execution data and performance indicators at run-time to detect problematic spots, like bottle-necks, and we analyse, which parts of the organisation (the network, the interaction patterns, etc.) are interesting candidates for an adaptation [17]. In the planning phase we perform a cost-benefit reasoning for the most promising candidates. If the predicted cost-benefit ratio exceeds a certain threshold, we will execute this adaptation in the Execute-phase of MAPE using the Sonar engine. Here, adaptation costs are measured in the amount of changes needed to transform the organisation, while the benefit is measured as the relative increase of the indicator values (cf. [18] for more details).

In general, we cannot predict the change of indicator values for the Org-MAS candidate by a purely *static* analysis due to the complexity. Instead, we *simulate* a digital twin of the Org-MAS candidate, called the Sonar-MAPE-Loop@run.time, during the planning, i.e., we follow a *models@run.time* approach [19]. In summary, this leads to the following recursive structure of the Sonar-MAPE-loop:

- The Sonar-MAPE-Loop is the Petri net based specification of the execution engine (for more details cf. Sect. 3).
- The Sonar-MAPE-loop contains the Sonar-organisation net and the interaction protocols as net-tokens (for more details cf. Sect. 2).

- The planning step of the Sonar-MAPE-loop uses a digital twin of itself for prediction (cf. Sect. 4). Here, the digital twin is again a net-token within the Sonar-MAPE-loop.

The use of nets-as-tokens (another name for nets-within-nets [13]) allows for a simple runtime modification of the organisation or its interaction protocols. It also simplifies the execution of the digital twin within the system itself. Especially, the Hornet formalism [12] supports structural change of the net-token's topology with algebraic operations in a direct manner.

Meta-Parameters of the Twin From the perspective of our MAPE-loop, the environment and the organisational member agents are external, so that the organisation has no control over them. To obtain a closed system for prediction we have to model the environment and the decision logic of member agents. We extend the twin by models for both; the models are stochastic in nature and they assign probabilities to alternatives (usually estimated during the monitoring phase). We have three meta-parameters in the model (explained in more detail in Sect. 4): c_{org} defines the obligatory nature of organisational constraints; the organisational transparency c_{tra} defines how easy agents can incorporate organisational constraints into their decision logic; and η defines the agents' learning speed.

The meta-parameters correspond to a concrete deployment of our MAS; e.g., a low value c_{org} corresponds to a deployment where organisational constraints are, more or less, only recommendations for the member agents, while a high value of c_{org} corresponds to an enforcement of organisational constraints. These different deployments differ in their complexity, e.g., enforcement of organisational constraints has to be implemented by more complex techniques. Therefore, an analysis of the effect of the meta-parameters for a given organization model helps the designer to identify those deployment configurations that have a low complexity and enable a good performance at the same time.

Research Agenda On the one hand, an organisational design has to be restrictive in order to help agents to make good decisions (short-term reward); on the other hand, it must give the agents room for experiments to adapt to changes flexibly (long-term reward). Therefore, a central aspect of a 'good' Org-MAS design is the balance between the organisational constraints and the individual decisions within a long-term co-evolutionary adaptation process, i.e., the micro-macro dynamics. When considering the long-term behaviour we also favour 'robust' organisations, i.e., MAS where the behaviour does not depend critically on the given configuration. Instead we want that a variation of parameters has either little impact or the system will stabilise soon – maybe 'somewhere' else.

Additionally, quality is multi-dimensional and therefore a change in the organisation might be beneficial for one dimension, but worse for another. The central discovery of computational organisation theory [6] is that there is no organisation that performs well for each dimension; the question whether an Org-MAS is well balanced cannot be answered in general. The choice also depends on the environmental dynamics, the agents' learning capabilities and many other aspects. Here, we address the external aspects by the meta-parameters of our digital twin. We will use them twofold: Firstly, we will use them to formulate properties that help to identify a 'good' Org-MAS: the *quality impact* of an organisation; the *interchangability* of organisational guidance (macro) and individual learning (micro); the *indispensability* of the organisation (i.e., a weaker organisational impact could could not be compensated by a higher learning rate); and the *cumbersomeness* of obsolete constraints. Secondly, we use the space of meta-parameters for deployment. The search for a well balanced organisation design boils down to find a good setting of our meta-parameters. Therefore, we use also use the twin model of our *Sonar-MAPE-Loop* to perform a sweep through the meta-parameter space to identify 'good' candidates.

Structure of the Paper Our contribution has the following structure: We give a short introduction into the Sonar model in Section 2 and into the Sonar-MAPE-Loop in Section 3. Then, we will have a closer look at the digital twin of the Sonar-MAPE-Loop in Section 4. Here, we explain the meta-parameters and formulate properties that indicate a 'good' Org-MAS. In Section 5 we will describe some example scenarios of organisations. These scenarios are used as examples in Section 6 to demonstrate the usefulness of our properties. In Section 7 we use our digital twin model to perform a sweep through possible parameter settings to guide the modeller during the deployment phase. The work closes with a conclusion.

Related Work The most prominent paper to adaptive systems is given in the context of self-* properties (self-healing, self-organising, etc.) [20]. A related field to MAS are service-oriented architectures, where the orchestration and choreography of services is closely related to the intention behind organisational structures [21, 22].

The theoretical foundations of adaptivity are formulated for Complex Adaptive Systems (CAS) [23]. Among the theories of CAS are topics like Genetic Programming [24], Swarm Intelligence [25], Game Theory and Mechanism Design [26], and Network Analysis [27] – to mention a few.

The central approach to adaptivity in software engineering is known as the MAPE-K approach, i.e., MAPE with K (knowledge), where the knowledge has much in common with the organisational concepts [1, 28]. A rational approach to adaptation based on cost-benefit reasoning is studied in [5]. Digital twins and their use for analysis and planning at run-time are studied in [29, 19, 30, 31].

The work presented here also has some connections to our research on adaptation state spaces [32, 33], which we have studied for the self-modifying Petri net class of Hornets [12, 34].

Prominent examples of Org-MAS from the design and implementation perspective are AGR [35], MOISE [36], and OPERA [37]. Like Sonar, these approaches specify concepts like roles, goals, interaction protocols and the relationships between these concepts. Using Petri nets, Sonar emphasises the process nature of organisations while other approaches accentuate the logical aspects. An overview is given in the handbook of Dignum et al. [38]. A complementary approach to organisational design comes from Process Mining [39], which also includes the mining of structures. One direction of Org-MAS theory is called Computational Organisation Theory [40, 6]. In the context of Org-MAS, special attention is paid to the Micro-Macro-Link [15], which is especially studied in the area of *Socionics* [41], where Sociology and Informatics are combined.

2. The Sonar-Model

In the following we give a short introduction into the Sonar formalism, which is used for analysis, as well as for implementing an Org-MAS. The Sonar organisational model [42] is based on Petri nets and its semantics (i.e. the execution loop) is defined on top of the operational semantics of Petri nets. We will recall some basics and refer to [42] for details.

In MAS, organisations are used to describe super-individual control concepts to complement the purely local perspective of agents and to overcome the so-called *price of anarchy*. These concepts are usually introduced in MAS organisations [7], a concept we studied before with respect to multi-party workflow nets within organisations [9].

The interaction protocols are defined by a *distributed workflow net* D, which is a multi-party version of the well-known workflow nets (WF nets; WFN) [43, 22] where the interacting parties are called *roles*. Let $\mathcal{R}$ be a universe of roles. Each transition of a distributed WFN is mapped by r to a role with the meaning that a transition t must be executed by an agent that implements the role $r(t)$. For a WFN D, a set of roles R induces a subnet $D[R]$, which is generated by restricting the net to those transitions that are mapped to R. Let $\mathcal{D}$ be a set of workflow nets.

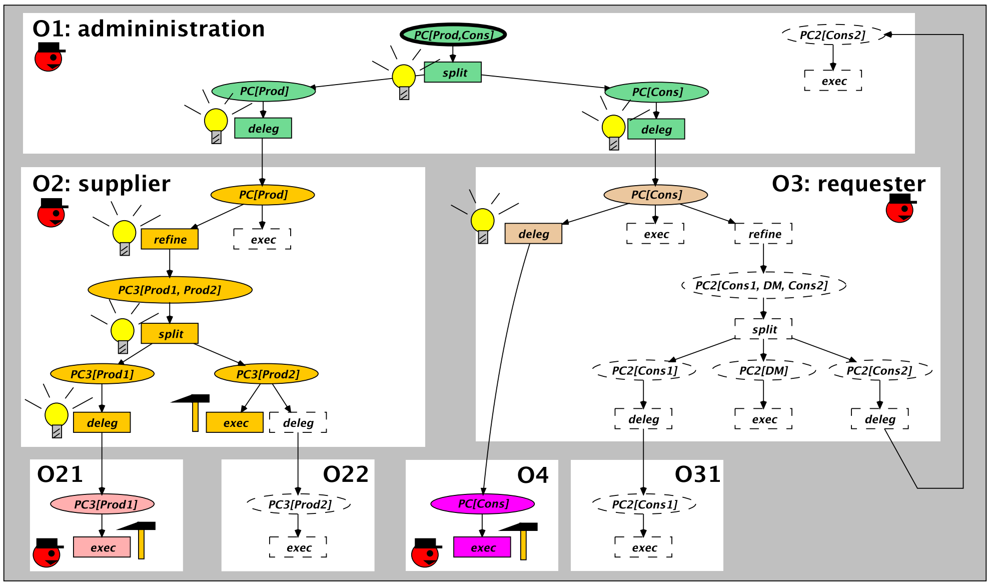


Figure 1: A Sonar-Organisation Model; highlighted: a Sonar-Team (adapted from [17])

In Sonar, the organisation is a Petri net $N = (P, T, F)$, where each place is of the form $p = \text{task}^{a}_{D[R]}$, which describes a *task* for the agent a to establish the behaviour that is described by the subnet $D[R]$. Our standard working example of an organisation net is shown in Figure 1. It will be explained in more detail in Section 3.

The tasks are either generated in the environment or are sub-tasks, generated by the organisation itself. The places with empty pre-set are those tasks that the organisation is responsible for, i.e., tasks that are generated externally: $P_0(Org) := {}^{\circ}P := \{p_0 \in P \mid {}^{\bullet}p_0 = \emptyset\}$.

Each task is handled by the transition of the organisation net, which are called *team-operators*.

We have four types of operators:

1. Delegate: The task to implement $D[r]$ 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: Here, $D[r]$ 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 WFN $D[r]$ is assigned to an agent for execution.

Each team-operator t also imposes a *constraint* $\psi(t)$ onto the execution of WFN D. Each subset T of these operators induces a *Sonar-Organisation Net* $N = (P, T, F)$ where $P = {}^\bullet T \cup T^\bullet$. The mapping $\alpha : \mathcal{P} \to \mathcal{A}$ returns the owner of a task: $\alpha(\text{task}^a_{D[R]}) := a$. The mapping α is extended to transitions by defining $\alpha(t) := \alpha(p)$ for the unique place p in the preset of t.

A *Sonar-Organisation* $Org = (N, \mathcal{O}, \mathcal{R}, \mathcal{D})$ is given by the organisation net N, the set of positions $\mathcal{O}$, which is the set of agents owning the operators: $\mathcal{O} = \alpha(T)$, the role set $\mathcal{R}$, and the set of WFN $\mathcal{D}$. The agents in $\mathcal{O}$ are called *organisation position agents (OPA)*.

3. The Sonar Run-Time Engine

The Sonar-engine defines an *execution loop* to carry out the organisational teamwork. The execution loop is carried out by a MAS, called the OPA-OMA-Network.

3.1. The OPA-OMA-Network

We use Sonar models to generate MAS architectures. This is done by parameterising a general Org-MAS engine with a concrete Sonar model. The Sonar-engine instantiates each position $O \in \mathcal{O}$ of the model by an agent: the *organisation position agent* (OPA). The network of OPAs as given in the model $Org = (N, \mathcal{O}, \mathcal{R}, \mathcal{D})$ defines the so-called *formal organisation.* An OPA roughly represents the *type* of agents expected at the given position. The implementation of this position is done by coupling another agent, the *organisational member agent (OMA)*, with the OPA. The OMA is not part of the formal model and is implemented separately. Note, that there is no need for additional concepts to integrate organisations into our MAS, as we specify the organisation as a network of OPAs and this network is just a part of the whole system. The OPA-OMA network represents the Sonar model at run-time.

3.2. The Sonar-Teamformation

Formally, teams are partial-order runs [44] of the organisation net. A configuration μ resolves conflicts in the organisation net and defines a partial-order run, called a *team group* (short: *team*). Therefore, μ defines the mapping of an initial task $p \in P_0(Org)$ to a team $G = \mu(p)$.

For the organisation in Figure 1 the highlighted nodes define such a partial-order run, i.e., a team G, for the initial task $p = \text{task}^{O_1}_{PC[Prod,Cons]}$. (Recall that this describes a task for the organisation agent O_1 that is handled by the interaction of the two roles $Prod$ and $Cons$ as specified in the WFN PC.) The three transitions labelled with a hammer at the agent O_{21}, O_2 and O_4 show the assignment operators, i.e., these agents really implement the roles in the team-WFN D. The team G shown here will execute a team-WFN $D(G)$, that is generated as the dynamic composition of the services provided by the three final transitions:

$$D(G) = (PC_3[\mathit{Prod}_1] \parallel PC_3[\mathit{Prod}_2] \parallel PC[\mathit{Cons}])$$

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

Simplifying a little, a central aspect of the Sonar-teamwork is to dynamically select the team-WFN $D(G)$ at run-time from the set of all WFN that are allowed by the organisation during a cooperative process among the agents; the organisational structure guarantees that $D(G)$ is a refinement of the initial WFN $D[R]$ for the considered initial task $p = \text{task}^{a}_{D[R]}$.

3.3. The Sonar-MAPE-Loop

The Sonar-engine defines a general *execution loop* where a *task* triggers the *formation* of a team G of agents. The Sonar-MAPE-Loop is presented in [11], where we studied the Micro-Macro-Dynamics between agents (micro level) and the organisation (macro level).[1]

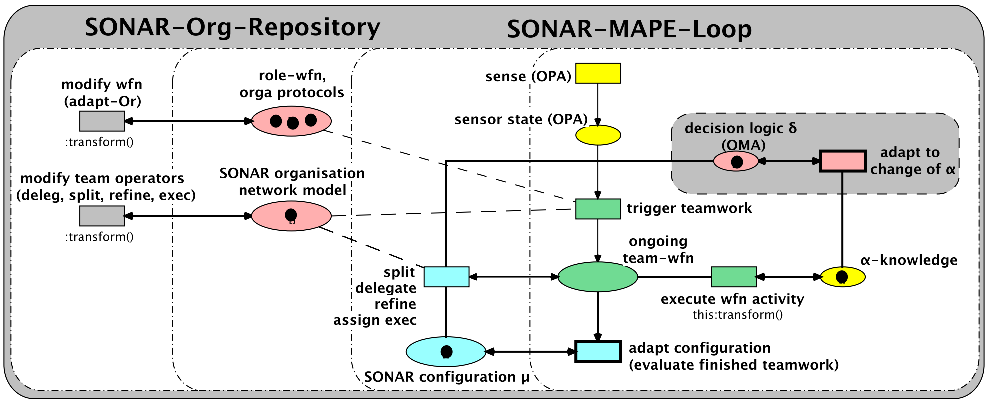


Figure 2: The Hornet-Model of our Sonar-MAPE-Loop (adapted from [11])

We use *nets-within-nets* [13] to specify the adapting MAPE-loop with the same formalism that is used to specify Sonar itself: Petri nets. We use the formalism of Hornets [12, 32] to enable high-level features as net-operators. Figure 2 shows a simplified version of the Hornet-Model of the Sonar-MAPE-loop. The model has two levels:

- The top-level net, called the system-net, defines the overall process of the Sonar-MAPE loop, while the so called net-tokens describe the organisation model, the WFN describing the interaction protocols, the configuration, and all the agents.

[1] This run-time engine is a revised version of a former approach presented in [45]. This earlier approach compiled a Sonar-model directly into the MAS, which complicates run-time adaptations as studied here.

- The organisation model N is a token on the place *Sonar organisation model*. Each WFN from the set $\mathcal{D}$ is a net-token on the place *role wfn*.

We will give a rather short description of the net, details are given in [11]. For simplicity we have chosen to represent the OPAs $\mathcal{O}$ only implicitly as part of the transition *sense*, which generates a *sensor state* and *triggers the teamwork*. The *ongoing teamwork* is defined by using the team-operators, (i.e. the transition *split, ...*). Which team operator is chosen depends on the side conditions, i.e., the organisation net N and the configuration μ. The *ongoing teamwork* is executed by the transition *execute wfn activity*. This updates the agent's *knowledge*. A significant change of the α *knowledge* leads to an *adaptation of the decision logic* δ. At the end of the WFN execution we *adapt the Sonar configuration* μ.

Note, that there is no need for additional features in our model to allow for transformations as the Hornet-model of the Sonar-MAPE-loop in Figure 2 contains two kinds of protocols (i.e. workflows): the normal first-order workflows, which modify the agents' environment (as discussed here); but, the model also contains *second-order workflows* (not discussed here), which modify the first-order workflows (using a synchronisation with the transition *modify wfn* in Fig. 2) and/or the Sonar-model itself (synchronising with the transition *modify team operators*). In principle, this hierarchy is unbounded, as we allow for third-order workflows to modify second-order workflows and so on.

For the second-order protocols, which are used for adaptation, we have to estimate the effect on the system (cost-benefit reasoning) . This is done in the planning phase of the MAPE-loop using a digital twin.

4. The Digital Twin of the Sonar-MAPE-Loop

During the adaptation of the organisation (as performed by the transition *adapt configuration* μ in Figure 2) we have to predict the impacts of modifying the Sonar configuration (the organisation state) μ to μ'. This is complicated by the fact that changes may lead to a different micro-macro dynamics as well. The transition uses a digital twin of the whole Sonar-MAPE-Loop (not shown in Fig. 2) to predict the effect of adaptations.

Since we use the digital twin for predicting the outcome of several adaptation candidates the simulation of the twin model has to be much faster than the system itself. Therefore, the twin has to be an abstraction. The digital twin also has to include the external elements, i.e., the environment and the member agents. Both are given as a *stochastic* model assigning probabilities to alternatives (instead of a complex decision logic). The interplay of organisation, member agents, and environment is modelled by our meta-parameters c_{org}, c_{tra}, and η, explained in the following subsections.

4.1. Modelling the Organisational Impact

To predict the resulting micro-macro dynamics we simulate a digital twin of the Sonar-MAPE-Loop. The digital twin is a simplified version of the original Sonar-MAPE-Loop net: We model all internal agents' logic by giving choices a probability. These choices are present in the team

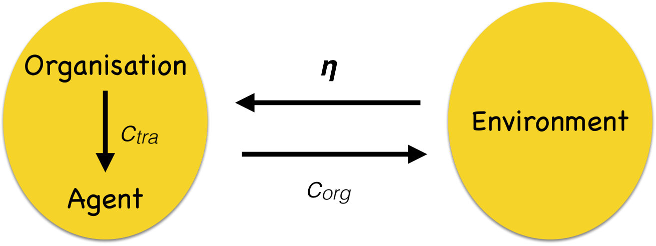


Figure 3: Relationships between the Meta-Parameters: organisational impact c_{org}, organisational transparency c_{tra}, and learning step size η

WFN (as xor-choices) and in the organisation net (as different branches for the team formation) as well.

Each organisational position agent (the OPA) is coupled with its member agents (the OMA) via the meta-parameter c_{org}: As the Sonar specification formulates constraints in a probabilistic way (i.e. defining a probability p_{OPA} for choices of the OPA) and the OMA is modelled as probabilistic actor (i.e. defining a probability p_{OMA}), too, the overall behaviour is simply a superposition of both:

$$p = c_{org} \cdot p_{OPA} + (1 - c_{org}) \cdot p_{OMA} \tag{1}$$

Here, the meta-parameter $c_{org} \in [0; 1]$ denotes the *organisational impact*, i.e., to which extend members are obliged to fulfill organisational requirements. Here, we have two borderline cases:

- $c_{org} = 0$: the organisation has no impact at all and only the choices of the member agents matter. This usually describes quite flexible systems, but often with only limited coherence.
- $c_{org} = 1$: the organisational constraints are strong obligations. This leads to maximal predictability, but the system is rather inflexible, because agents cannot deviate.

4.2. Modelling the Member Agents

The agents learn a decision probability p for each choice. In our model, agents learn by following an environmental feedback signal g, which they try minimise by going steps of size $\eta \geq 0$ (also known as the learning rate) into the opposite direction of the gradient g'. The agents also react on the organisational constraint $p_{org}(t)$ at time t by incorporating $c_{tra} \cdot p_{org}(t)$ into their value $p(t)$. The weight $c_{tra} \in [0; 1]$ describes the *transparency of the organisation* about its preferences $p_{org}(t)$. A high value describes an organisation that is quite clear in communicating the constraints and therefore the agents will incorporate them faster. The combination of theses two aspects leads to the learning formula:

$$p(t+1) = (1 - c_{tra}) \cdot \Big(p(t) - \eta \cdot g'\big(p(t)\big)\Big) + c_{tra} \cdot p_{org}(t) \tag{2}$$

Figure 3 summarises the relationships between the meta-parameters, i.e. the organisational impact c_{org}, the organisational transparency c_{tra}, and the learning step size η.

4.3. Properties of Organisational Dynamics

We are interested in different aspects of the Org-MAS behaviour w.r.t. the meta-parameters.

- **Quality Impact:** *For which configurations of the meta-parameters is the ratio of 'resulting value' to 'optimal value' greater than a given threshold?*
  This is the most obvious question as these configurations define the candidate set.
- **Interchangability:** *For which configurations do we obtain the same ratio of 'resulting' to 'optimal' behaviour?*
  This is interesting because for these configurations one can trade stronger impact of organisational constraints for a lesser individual learning rates or a lower transparency – or vice versa. For many scenarios the designer is faced with given learning capabilities of the member agents. For interchangeable configurations we can adjust the organisational impact accordingly.
  Extension: *Which interchangeable configurations have a kind of fixed 'exchange-rate' of learning/transparency and organisational constraints?*
  The designer can switch between those configurations. So, the designer can balance the costs for enforcing organisational constraints with the learning costs of members according to the concrete deployment environment.
- **Indispensability:** *For which configurations is it* not *possible to compensate a lower organisational impact by a higher learning rate?*
  In other words: Are there configurations where a lower organisational impact reduces the ratio, more or less, independently from the learning rate?
  These configurations are the contrary of the interchangeable ones. They indicate that organisational coordination is essential and the designer should not lower the impact value.
- **Visibility:** *Are there configurations such that a lower organisational impact may be compensated only if the agents have a high learning activity and, at the same time, the organisation has a strong transparency?*
  In this case the designer has to make sure that member agents are not only reacting to the environment but also incorporate organizational constraints into their decisions.
- **Cumbersomeness:** *Are there configurations where a higher organisational impact and/or incorporation activity is contra-productive (i.e., leads to a worse ratio)?*
  In this case the designer has to moderate the impact c_{org} of the organization and also the tendency c_{tra} of incorporating them.

In the following we study some examples to illustrate these properties.

5. Our Scenarios

In the following we explain our properties giving some examples where we vary the meta-parameters. The following scenarios are all executed as instances within our Sonar-MAPE-Loop. The whole system uses Renew [14] for simulation. The source files are available at: https://github.com/koehler-bussmeier/digitaltwin/. A description is given in the appendix.

5.1. Our Scenario S0 (Seasonal Production)

Two agents produce one of two possible goods: a or b. Each agent has a preference on how much he will produce of each good. The interaction protocol P defining the production process contains two roles: r_1 and r_2 for the two producers.

The organisation net N is a quite simple one. There is exactly one task, the production of resources a and b, and the organisation model induces exactly two teams for handling this task.

We have three OPAs: O_0, O_1 and O_2. Here, the main decision, which of the teams is chosen, is made by O_0. For both teams the agent O_1 is responsible for role r_1 and O_2 for r_2. The teams differ in the organisational constraints imposed on r_1 and r_2.

For S0 we assume that the environment exhibits a quite simple (and deterministic) behaviour: it oscillates each $s = 500$ production steps (which equals the number of executed loops in the WFN) between two different states (i.e., quite slowly). The OPA O_0 adapts to this external change by switching between the two teams. Essentially this means that the production probability constraint of good a is $p_{org} = 0.6$ for the first state and $p_{org} = 0.1$ for the second.

We have two OMAs (i.e., member agents). They start with an initial preference, too. The initial preference for good a is $p_1(t) = 0.7$ for the first agent and $p_2(t) = 0.5$ for the second at time $t = 0$. The member agents adapt at the end of each execution of the WFN. In this basic scenario an agent adapts its behaviour towards the organisational constraints:

$$p_i(t+1) = (1 - c_{tra}) \cdot p_i(t) + c_{tra} \cdot p_{org}(t), \quad i = 1,2 \tag{3}$$

Therefore, S0 is a special case of (2) from Section 4.2 with no learning from environmental feedback: $\eta = 0$.

5.2. Our Scenario S1 (Coordinated Production)

The second scenario is similar to S0. We still produce two kind of resources, and the desired production changes each $s = 500$ steps. We still oscillate between two states. The first state still describes an environment that favors a production outcome where resource a has a frequency of $q = 0.6$. But now the second states does no longer favor a single choice – now it prefers *two* alternative outcomes: Either there is a quite low frequency q_- or a quite high frequency q_+. These two values are the maximal values of our hat-like reward function are shown in Figure 4 (a).

The existence of two distinguishable options make the situation a kind of a coordination game (Therefore, our scenario bears a faint resemblance to the battle-of-sexes in game theory.): Both agents should make the same choice, since if they take opposite choices the resulting frequency will be around 50%, which is the worst outcome. We assume that a high learning rate is not able to compensate for a low organisational impact as this is also true for the battle-of-sexes game.

Our intention is the following: Since the member agents adapt independently, there is the risk that they will adapt their selection preference in an uncoordinated way, i.e., in different directions in our case. This risk is known as the *price-of-anarchy* in game theory. Organisations are supposed to *compensate* the unwanted effects of uncoordinated adaptation steps.

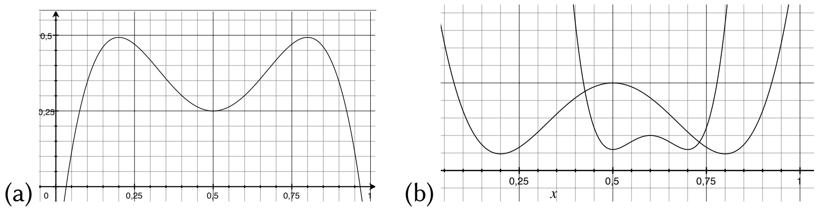


Figure 4: S1: (a) Shape of the Reward Function; (b) The Loss Functions

We will start our OMA for role r_1 with a rather high preference $p_1 = 0.7$ for resource a and the OMA for r_2 with low one of $p_2 = 0.4$. In this case it is quite likely that the OMA for r_1 will adapt towards the high optimum q_+, while the OMA for r_2 will adapt in the opposite direction towards the low optimum q_-.

For simplicity we assume that the two alternatives q_- and q_+ are equally attractive, i.e., we have the alternative minima $q_\pm = q_0 \pm a$. Moreover, we like the function to be symmetric. Here, we choose a reward function r of the following form to model the hat-like shape of Fig. 4 (a):

$$r(x) = -(x - q_0)^2 \cdot ((x - q_0)^2 - 2a^2)$$

Adaptation (maximising the reward r) is equivalent to searching for a minimum of the *loss* function $g(x) := -r(x)$.

For S1 the member agents adapt only via a gradient descent (of step size $\eta \geq 0$):

$$p_i(t + 1) = p_i(t) - \eta \cdot g'(p_i(t)), \quad i = 1, 2 \tag{4}$$

Therefore, S1 is a special case of the learning rule (2), where we ignore the incorporation of organisational constraints: $c_{tra} = 0$.

In this scenario will still toggle between two states each $s = 500$ time steps. For uniformity, we model also the first state (where $q = 0.6$ is desirable) by a function of the shape above, only with a very narrow distance between the two minima: We choose $q_0 = 0.6$ and $a = 0.1$, which leads to minima at $q_- = 0.5$ and $q_+ = 0.7$. For the second state, we have $q_0 = 0.5$ and $a = 0.3$ resulting in $q_- = 0.2$ and $q_+ = 0.8$. The loss functions for the two states are sketched in Figure 4 (b).

5.3. Our Scenario S2 (Organisational Learning)

For S1 we assumed that agents only follow the feedback from the environment. Scenario S2 will also incorporate the feedback signal from the organisation and we use our general update rule (2), i.e., we combine the learning functions (3) and (4) from S0 and S1. Therefore, S2 now relates all three meta-parameters at the same time.

5.4. Our Scenario S3 (Effect of Outdated Organisational Rules)

Scenario S3 modifies S2 slightly, such that the organisation still performs its role as it reduces complexity by breaking the symmetry towards the lower production frequency. But, we assume that we are now at the point where the environment has changed while the organisation is still adapted to a former situation not valid anymore.

In the previous scenarios the organisational decisions were always 'correct'. However, even if the organisation has managed to configure itself in such a perfect way, sooner or later the rewards will change due to a dynamic environment. Therefore, we have the risk that the organisation is not only badly adjusted; even worse, it will hamper the member agents, which are usually much quicker in adapting to a change in the environment. In such a situation the organisation rules are not the solution to overcome the price-of-anarchy; instead they are a millstone to adaptation. We assume, that a good compromise between these two extremes will position the value of c_{org} somewhere around $c_{org} \approx 0.5$, but, of course, the concrete value will depend much on the frequency and the strength of environmental changes. The parameter sweep in Sect. 7 will quantify this compromise region.

6. Evaluation of the Digital Twin Model

In the following we evaluate the simulation results with respect to their accordance to our hypotheses, which are about the general relationship between the meta-parameters. We evaluate the digital twin with a finite horizon h of oscillations. Here, we use $h = 8$ (which turned out to be computational feasible and big enough to observe the general trend), which leads to a number of $h \cdot s = 4000$ micro-adaptations of the OMAs.

Note, that for the scenarios S0 - S3 the desired behavior coincides with the organisational constraint, while for S5 the organization is slightly wrong. Thus, the designer can identify good values for the meta-parameters by the degree of how closely the systems' production follows the organisation line.

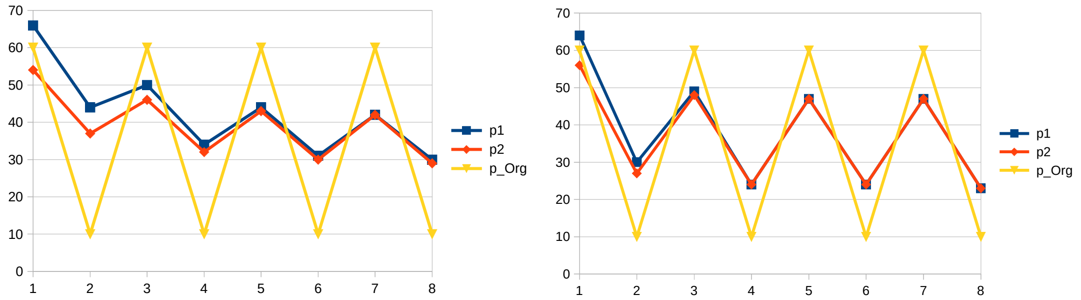


Figure 5: Adaptation of member agents' selection probability $p_i, i = 1, 2$ for resource a over time for $c_{tra} = 0.001$ (left) and $c_{tra} = 0.002$ (right) compared to the change of the organisational selection probability p_{org}

S0 In the scenario S0 we oscillate between two states which favour two distinct frequencies for production. In this scenario the member agents always adapt into the direction of organisational constraints, in the long run they behave as the organisation – even if they 'ignore' the organisational constraints (for $c_{org} \approx 0$) during their daily activity In the following we vary the

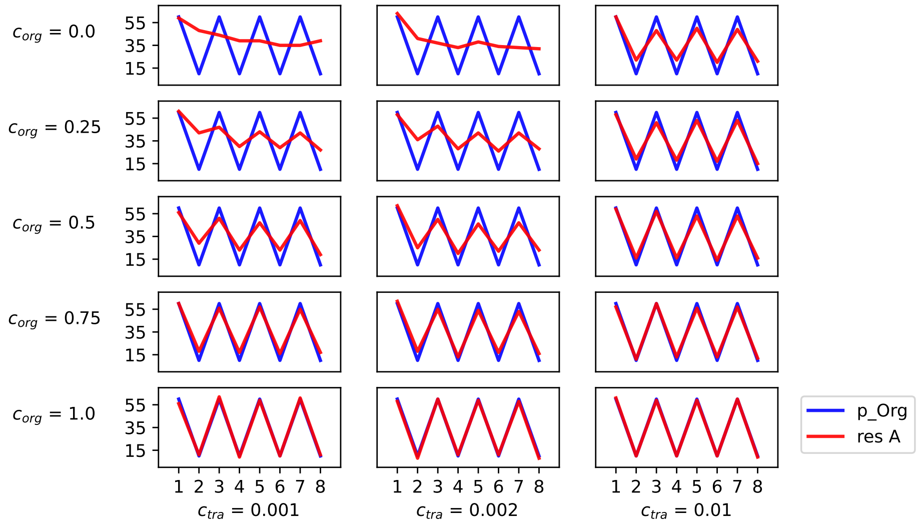


Figure 6: S0: Frequency of resource a over time compared to the dynamics of the organisational selection probability p_{org}

meta-parameters, especially the organisational impact c_{org} and the learning rate of member agents.

Figure 5 shows the adaptation of the member agents. It shows the selection probability $p_i, i = 1, 2$ for resource a over time for $c_{tra} = 0.001$ (left) and $c_{tra} = 0.002$ (middle) in comparison to the change of the organisational selection probability p_{org}. One can clearly observe that with the higher adaptation rate $c_{tra} = 0.002$ the member agents follow the organisational selection constraints more closely.

Figure 6 shows the impact of the micro/macro mixing-parameter c_{org}. The diagrams show the frequency of resource a taken for increasing values of c_{org}. The simulation uses $c_{org} = 1.0, 0.75, 0.5, 0.25, 0.0$ as values. We plot the the organisational constraint (blue) and the resulting production (red). As the organisational constraint is independent from the meta-parameters the blue line will be the same in all sub-graphs. Let us concentrate on both simulations with a low learning rate c_{tra} for the member agents (i.e., $c_{tra} = 0.001$ and $c_{tra} = 0.002$ on the left and in the middle). For all values of c_{tra} one can observe the same trend: Whenever the impact of the member agents is small ($c_{org} \approx 1$) we see that the resource production follows the organisational constraints very closely; whenever the impact of the organisation decreases (and member agents' impact increases) then the production frequency converges to the average $(0.6 + 0.1)/2 = 0.35$ of the two states since the agents could not adapt to the context change fast enough.

Since the organisation expresses the desired production, we obtain the result that for a less dominant organisation (smaller c_{org}) the system's performance (measured as the fit to the desired resource mix) decreases.

Whenever the organisation is less dominant it is interesting to study whether a higher transparency might compensate for this. We simulate our scenario with a third learning rate $c_{tra} = 0.01$, one magnitude bigger than the ones before. In this case the agents will follow the desired organisational dynamics almost immediately and a diagram analogously to Figure 5 will show indistinguishable curves. We study the impact of the organisational constraints on resource production for this scenario, too (cf. right column of Fig. 6). We can see that his high learning rate is able to compensate a less assertive organisation since the production follows the desired curve even if the organisational impact is set to $c_{org} = 0$. One can see that for a rate of $c_{tra} = 0.01$ an organisation with no impact (i.e., $c_{org} = 0$) is comparable to an organisation with a very high impact but only a small learning rate $c_{tra} = 0.001$ – as the diagram for $c_{tra} = 0.01$ and $c_{org} = 0$ looks roughly like that for $c_{tra} = 0.001$ and $c_{org} = 0.75$.

From the simulation results we can identify those configurations with a high *quality impact*: They have either a high value for c_{org} or forc_{tra} – or both. We can also identify *interchangability* as an increase of the transparency can compensate a decrease of organisational impact. In this simple scenario we cannot observe *indispensability* at all.

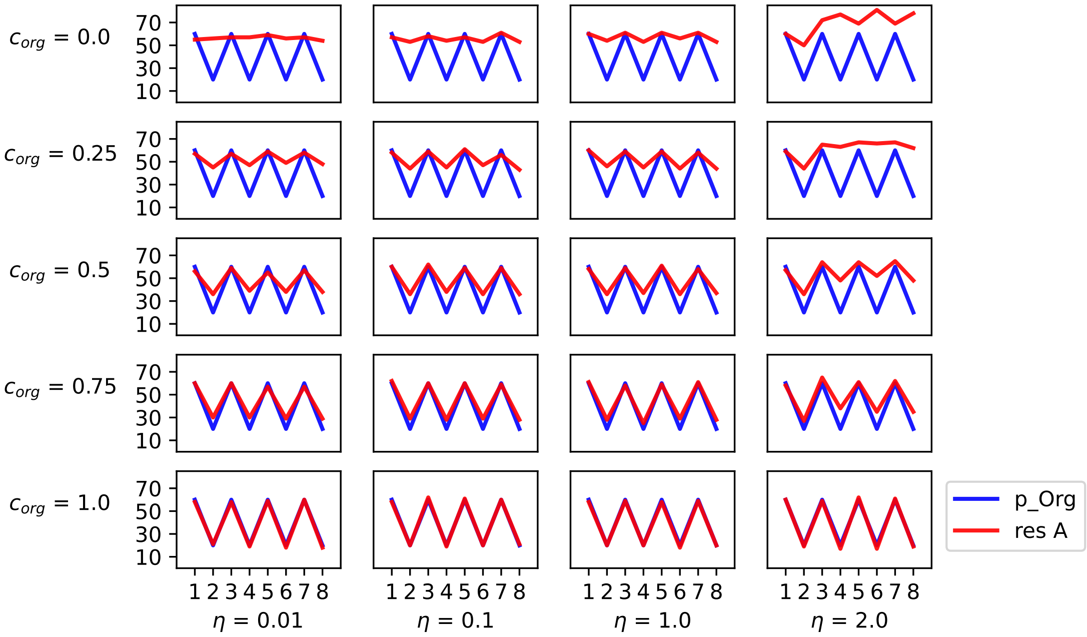


Figure 7: S1: Frequency of resource a over time compared to the dynamics of the organisational selection probability p_{org}

S1 In scenario S1 we still oscillate between two states, but the states require some symmetry breaking, i.e., some coordination between agents, since the desired outcome is either a rather low or a rather high frequency for the production of resource a. In other words: The agents should make the same decision. Since the WFN does not provide any coordination here, the only source of coordination is the organisation itself. The organisation constraints this situation to the low frequency decision.

The OMA for r_1 starts with a selection preference for resource a of $p_1 = 0.7$, which is likely to stabilise – in both states – in the left minima of Figure 4 (b), while the OMA for r_2 starts with $p_2 = 0.4$, which is likely to stabilise on the right side. Therefore, we expect that a gradient descent will lead the OMAs to different values.

The organisation is intended to reduce the price of anarchy by constraining the situation. Here, we set the organisational constraint to $q = 0.6$ for the first state, which in between the minima of this state, and to $q = 0.2$, which is the left minimum, for the second.

Similarly to S0 we can identify configurations in Figure 7 with a high *quality impact*. One can observe that a system without organisational control (i.e., for small c_{org}, which are shown at the top) does not follow the desired production profile at all. For all values of the learning rate η we see that increasing the organisational impact c_{org} (i.e., going down in the figure) leads to an ever closer behaviour. Additionally, we can observe that for a fixed c_{org} and a small learning rate η a change of η shows only insignificant effects, while a small increase of c_{org} really improves the quality. For very high values of η one can observe the phenomenon of over-shooting for gradient descent. Learning at high-rates introduces some kind of instability.

Remarkably, unlike in S0 we see *indispensability* for S1 in Fig. 7: For small values of c_{org} we have configurations where the learning step size η has almost no impact, i.e., we cannot trade organisational impact for an increased learning activity of the member agents.

The fact that a lower organisational impact could not be compensated by a higher learning activity indicates that organisational coordination is essential for S1 and the designer should not lower the impact value below a certain value.

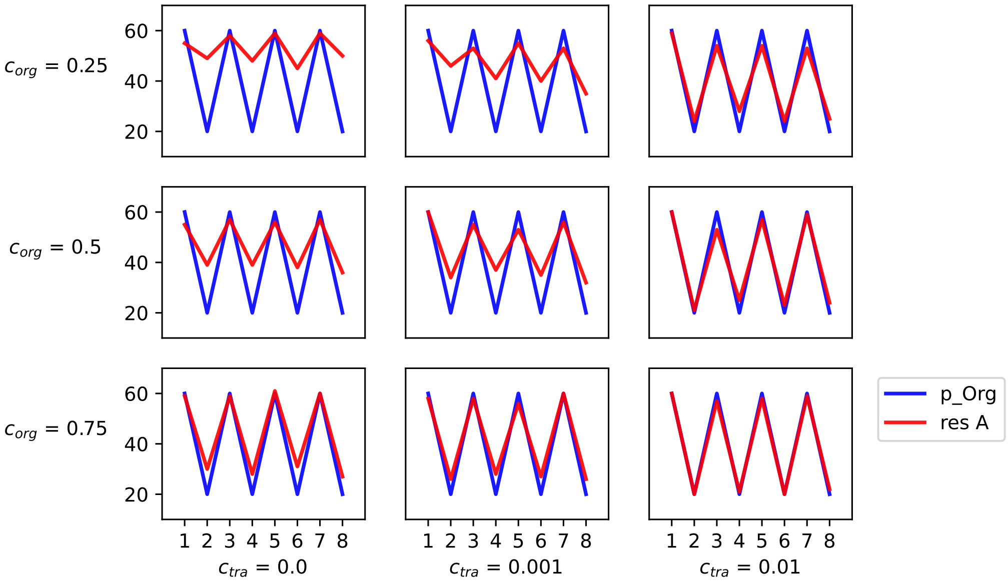


Figure 8: S2: Frequency of resource a over time for $\eta = 0.01$

S2 Scenario S2 is more complicated since we vary the organisational impact c_{org} against different transparency values and for different transparencies for a fixed η. Since we use our scenarios to demonstrate the concepts we will restrict ourselves to only one value for η.

In Figure 8 one can observe *interchangability*: a lesser organisational impact (i.e., going up in the figure) can be compensated by learning provided that the organisational constraints are transparent to the learning member agents (i.e., by going right in the figure). We can observe a kind of fixed *exchange-rate* of learning/transparency and organisational constraints as the diagram does not change much when going one step up and one step right at the same time.

A more detailed evaluation for different values of η (not shown here) also reveals the property of *visibility*: A lower organisational impact could not be compensated by a higher learning activity alone; it is necessary that the organisation additionally sends a strong transparency signal.

S3 In this scenario we assume the same rewards as for S1 and S2 (cf. Figure 4 (b)): For the first state we have minima at $q_- = 0.5$ and $q_+ = 0.7$ and for the second state we have $q_- = 0.2$ and $q_+ = 0.8$. In S1/S2 the organisation constraints the OMAs in the teams to $q = 0.6$ for the first state and to $q = 0.2$ for the second. As a difference to S1/S2, we have an organisation that is not well adjusted, i.e., has not adapted to the recent environmental changes yet. Concretely, we set the organisation constraints to $q = 0.6$ (just in between the minima) for the first state and to $q = 0.0$ (which is too low) for the second.

Note, that in S3 the desired production is unchanged but the blue line for p_{org} is now below that; e.g., for $c_{org} = 0.5$ we see that the smaller transparency value $c_{tra} = 0.001$ is better than the greater one $c_{tra} = 0.1$ since the production is closer to the desired outcome of 0.2.

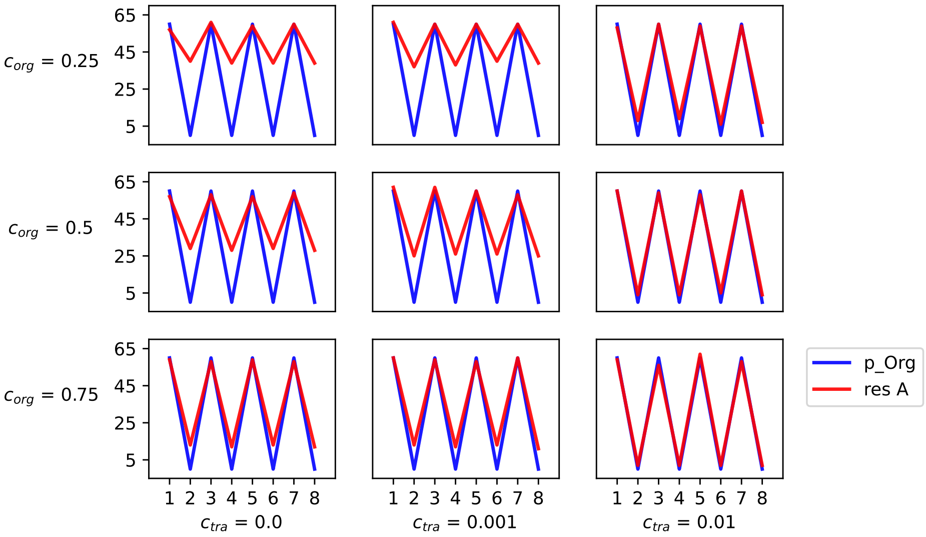


Figure 9: S3: Frequency of resource a over time for for the fixed learning rate $\eta = 0.1$

For the previous scenarios S0-S2 we could identify the *quality impact* by the rule of thumb: The more, the better. This is no longer true, as can be seen in Figure 9. It shows the frequency for a fixed learning rate $\eta = 0.1$, but for different values of c_{org} and c_{tra}. Here, one can observe that whenever the organisational constraints are 'wrong' somehow, then transparency value c_{tra} above a certain value (here: the right column of Fig. 9) is contra-productive (i.e., we observe *cumbersomeness*) since the member agents are not only driven into the direction of the environmental feedback but also into the wrong organisational one.

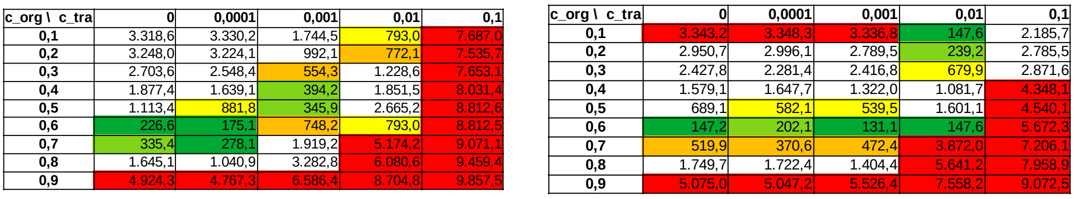
Therefore, we see *cumbersomeness* in S3, i.e., an outdated organisational signal cannot be compensated by increased individual learning activity.

7. Exploring Design Candidates of an Org-MAS

The main purpose of the twin model is to evaluate possible candidates for adapting the current organisational configuration μ w.r.t. costs and benefits. This evaluation is performed at run-time as part of our MAPE-Loop. For this evaluation we treat the organisation as fixed since we adapt an Org-MAS that is already deployed.

In the following we make use of the twin model outside the MAPE-Loop: We evaluate the current MAS by varying the meta-parameters themselves. Each meta parameter models the effect of several deployment decisions in the system; e.g., the organisational impact c_{org} could be implemented very strict by choreography rules or it could be implemented in a soft manner by a reputation mechanism. (These deployment options have been discussed as *teamwork parameters* in [9], which define a two-dimensional categorisation along the dimensions of team formation and team coordination.) If it turns out that e.g. the current deployment leads to a sub-optimal MAS then the designer might consider a re-deployment.

We could also use this idea even before deploying the system: We sweep through the space of all meta-parameter configurations to give feedback to the system's architect when finding a good deployment.



c_org \ c_tra	0	0,0001	0,001	0,01	0,1
0,1	3.318,6	3.330,2	1.744,5	793,0	7.687,0
0,2	3.248,0	3.224,1	992,1	772,1	7.535,7
0,3	2.703,6	2.548,4	554,3	1.228,6	7.653,1
0,4	1.877,4	1.639,1	394,2	1.851,5	8.031,4
0,5	1.113,4	881,8	345,9	2.665,2	8.812,6
0,6	226,6	175,1	748,2	793,0	8.812,5
0,7	335,4	278,1	1.919,2	5.174,2	9.071,1
0,8	1.645,1	1.040,9	3.282,8	6.080,6	9.459,4
0,9	4.924,3	4.767,3	6.586,4	8.704,8	9.857,5

$\eta = 0.01$

c_org \ c_tra	0	0,0001	0,001	0,01	0,1
0,1	3.343,2	3.348,3	3.336,8	147,6	2.185,7
0,2	2.950,7	2.996,1	2.789,5	239,2	2.785,5
0,3	2.427,8	2.281,4	2.416,8	679,9	2.871,6
0,4	1.579,1	1.647,7	1.322,0	1.081,7	4.348,1
0,5	689,1	582,1	539,5	1.601,1	4.540,1
0,6	147,2	202,1	131,1	147,6	5.672,3
0,7	519,9	370,6	472,4	3.872,0	7.206,1
0,8	1.749,7	1.722,4	1.404,4	5.641,2	7.958,9
0,9	5.075,0	5.047,2	5.526,4	7.558,2	9.072,5

$\eta = 0.1$

Figure 10: Heat-maps of the loss for all combinations of $c_{org} = 0.1, 0.2, \ldots 0.9$ and $c_{tra} = 0, 0.0001, 0.001, 0.01, 0.1$ for the learning rates $\eta = 0.01$ (left) and $\eta = 0.1$ (right)

Let us have a look at scenario S3 again: Assume a fixed learning rate η (which is given by the member agents and therefore cannot be influenced by our design), then a parameter sweep for all combinations of $c_{org} = 0.1, 0.2, \ldots 0.9$ and $c_{tra} = 0, 0.0001, 0.001, 0, 01, 0.1$ results in the heat-maps in Figure 10 where the accumulated loss for $\eta = 0.01$ and for $\eta = 0.1$ are presented. (In a more elaborate variant of this sweep we would increase the parameter selection recursively around the most promising candidates. Here, we skip this step.) We identify $(c_{org}, c_{tra}) = (0.6, 0.0001)$ for $\eta = 0.01$ and $(c_{org}, c_{tra}) = (0.6, 0.001)$ for $\eta = 0.1$ as optimal configurations.

From the heat-maps we can see – at least for the chosen parameter selection – that this environmental dynamics and the giving learning rate η is best balanced by an organisation with stronger impact, i.e., where $0.6 \leq c_{org} \leq 0.7$ and a quite moderate transparency. The first aspect seems at least plausible, since one might expect a good compromise around a ‘middle’ value of $c_{org} \approx 0.5$. Here, we see that the ‘compromise area’ is in fact a little bit higher: $c_{org} \approx 0.6$. For the transparency value c_{tra} we observe that quite moderate values are already sufficient.

Interestingly, we also have a ‘second-best’ cluster at $c_{tra} = 0.01$ and $c_{org} \leq 0.2$, i.e., for a high learning impact c_{tra}, that is also a good candidate when combined with small values of organisational impact c_{org}. This seems to be counter-intuitive, but for S3 the organisational constraint doesn’t match the environmental signals well and therefore it is maybe a good idea *not* to follow the organisation too much and better invest in learning activities. Therefore, the good configurations of this scenario roughly follow a curve where the product of learning impact c_{org} and transparency c_{tra} remains constant. Of course, the concrete reason why a combination of the meta-parameters results in a good performance requires an in-depth analysis of the model’s behaviour.

8. Conclusion

We studied the planning phase of a MAPE-loop for Org-MAS where we use a digital twin to predict the benefit of adaptations of the organisation model at run-time. The demand for a recursive formalism together with the fact that Sonar is a Petri net based formalism makes nets-within-nets highly interesting for our purpose here.

In this paper we studied the impact of the meta-parameters of the digital twin. We introduced properties of the organisational behaviour with respect to our meta-parameters (like quality impact, interchangeability, indispensability, visibility, and cumbersomeness) and studied the for example scenarios of a coordinated production.

The twin model gives us the opportunity for *quantitative* planning of adaptations, i.e. the Sonar-MAPE-Loop can decide whether a possible adaptation results into an organisation model that will perform better than the current one (i.e., we evaluate the benefit) and whether the amount of transformations (i.e., the costs) will pay off in the long run.

We have also seen that our digital twin model can be used to improve the deployment of the Org-MAS. If we find out that another setting of the meta-parameters is more beneficial, we can try to change the implementation in such a way that it is an counterpart of this better setting.

At the time being, we make an ‘educated guess’ how a concrete Org-MAS architecture has to be translated into a meta-parameter setting of the digital twin. Especially, we concentrate on our so-called *teamwork parameters* as mentioned above. In the case of a parameter sweep we are faced with the opposite problem, i.e., we have to translate a good setting of the meta-parameters into concrete deployment details of the Org-MAS. In current work, we are deepening our understanding which design choices of our concrete applications are expressed by which parameter settings.

Additionally, we would like to integrate a feedback into these settings: We will monitor the real system for a while and fit the meta-parameters according to our observations. Then we collect some more data, adjust the parameters, collect some more data, and so on. Of course, this a kind of chicken-and-egg situation and we have to start somewhere without any prior observation data. So a good understanding of the relationship between the real architecture and the meta-parameters remains essential.

References

[1] B. H. C. Cheng, (et al.), 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.

[2] G. Weiß (Ed.), Multiagent systems: A modern approach to Distributed Artificial Intelligence, MIT Press, 1999.

[3] 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.

[4] J. Sudeikat, M. Köhler-Bußmeier, On combining domain modeling and organizational modeling for developing adaptive cyber-physical systems, in: ICAART'22, 2022.

[5] 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.

[6] K. M. Carley, L. Gasser, Computational organisation theory, in: [2], 1999, pp. 229–330.

[7] 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.

[8] 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.

[9] 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.

[10] M. Köhler-Bußmeier, M. Wester-Ebbinghaus, Model-driven middleware support for team-oriented process management, Transactions on Petri Nets and Other Models of Concurrency 8 (2013) 159–179.

[11] 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, 2024.

[12] 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.

[13] R. Valk, Object Petri nets: Using the nets-within-nets paradigm, in: J. Desel, W. Reisig, G. Rozenberg (Eds.), Advanced Course on Petri Nets 2003, volume 3098 of *Lecture Notes in Computer Science*, Springer-Verlag, 2003, pp. 819–848.

[14] 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.

[15] M. Schillo, K. Fischer, C. Klein, The micro-macro link in DAI and sociology, in: S. Moss, P. Davidsson (Eds.), Second International Workshop on Multi-Agent Based Simulation, volume 1979 of *Lecture Notes in Computer Science*, Springer-Verlag, 2000, pp. 133–148.

[16] M. Köhler, D. Moldt, H. Rölke, R. Valk, Linking micro and macro description of scalable social systems using reference nets, in: K. Fischer, M. Florian, T. Malsch (Eds.), Socionics: Sociability of Complex Social Systems, volume 3413 of *Lecture Notes in Artificial Intelligence*, Springer-Verlag, 2005, pp. 51–67.

[17] 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.

[18] M. Köhler-Bußmeier, J. Sudeikat, Defining adaption measures for organizational multi-agent systems, International Journal of Parallel, Emergent and Distributed Systems (2023).

[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] J. O. Kephart, D. M. Chess, The vision of autonomic computing, IEEE Computer 36 (2003) 41–50.

[21] C. Peltz, Web services orchestration and choreography, IEEE Computer 36 (2003) 46–52.

[22] W. M. P. van der Aalst, N. Lohmann, P. Massuthe, C. Stahl, K. Wolf, Multiparty Contracts: Agreeing and Implementing Interorganizational Processes, The Computer Journal 53 (2010) 90–106.

[23] C. Gros, Complex and Adaptive Dynamical Systems: A Primer, Springer complexity, Springer-Verlag, 2013.

[24] J. H. Holland, Hidden Order: How Adaptation builds complexity, Helix Books, 1995.

[25] J. Kennedy, R. C. Eberhart, Swarm intelligence, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2001.

[26] Y. Shoham, K. Leyton-Brown, Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations, Cambridge University Press, New York, 2008.

[27] M. E. J. Newman, Networks: an introduction, Oxford University Press, Oxford; New York, 2010.

[28] 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.

[29] M. Simeoni, S. Balsamo, P. Inverardi, A. Di Marco, Model-based performance prediction in software development: A survey, IEEE Transactions on Software Engineering 30 (2004) 295–310. doi:doi.ieeecomputersociety.org/10.1109/TSE.2004.9.

[30] R. Calinescu, C. Ghezzi, M. Kwiatkowska, R. Mirandola, Self-adaptive software needs quantitative verification at runtime, Commun. ACM 55 (2012) 69–77. doi:10.1145/2330667.2330686.

[31] 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.

[32] 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.

[33] M. Köhler-Bußmeier, H. Rölke, Analysing adaption processes of Hornets, Transactions on Petri Nets and Other Models of Concurrency XVII 14150 (2023).

[34] 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.

[35] J. Ferber, O. Gutknecht, F. Michel, From agents to organizations: An organizational view of multi-agent systems, in: P. Giorgini, J. P. Müller, J. Odell (Eds.), Agent-Oriented Software Engineering IV, volume 2935, 2003, pp. 214–230.

[36] 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.

[37] 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.

[38] V. Dignum (Ed.), Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models, IGI Global, Information Science Reference, 2009.

[39] 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.

[40] K. M. Carley, Computational and mathematical organization theory: Perspective and directions, Computational and Mathematical Organization Theory 1 (1995) 39–56.

[41] T. Malsch, I. Schulz-Schaeffer, Socionics: Sociological concepts for social systems of artificial (and human) agents, Journal of Artificial Societies and Social Simulation 10 (2007) 11. URL: http://jasss.soc.surrey.ac.uk/10/1/11.html.

[42] M. Köhler, A formal model of multi-agent organisations, Fundamenta Informaticae 79 (2007) 415 – 430.

[43] 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.

[44] J. Esparza, K. Heljanko, Unfoldings - A Partial-Order Approach to Model Checking, EATCS Monographs in Theoretical Computer Science, Springer-Verlag, 2008.

[45] M. Köhler-Bußmeier, M. Wester-Ebbinghaus, D. Moldt, Generating executable MAS-prototypes from Sonar specifications, in: M. De Vos, N. Fornara, J. V. Pitt, G. A. Vouros (Eds.), Workshop on Coordination, Organizations, Institutions, and Norms in Agent Systems, COIN'10, volume 6541 of *Lecture Notes in Artificial Intelligence*, 2010, pp. 21–38.

A. The Renew-Model

The Renew-Model of the complete Twin of our Sonar-MAPE-Loop is quite large and consists of several components. These components are connected using the Renew-mechanism of synchronisation channels. The overall view is shown in Figure 11. In the following we present a 'zoom' into the Petri net model, presenting all the model's parts.

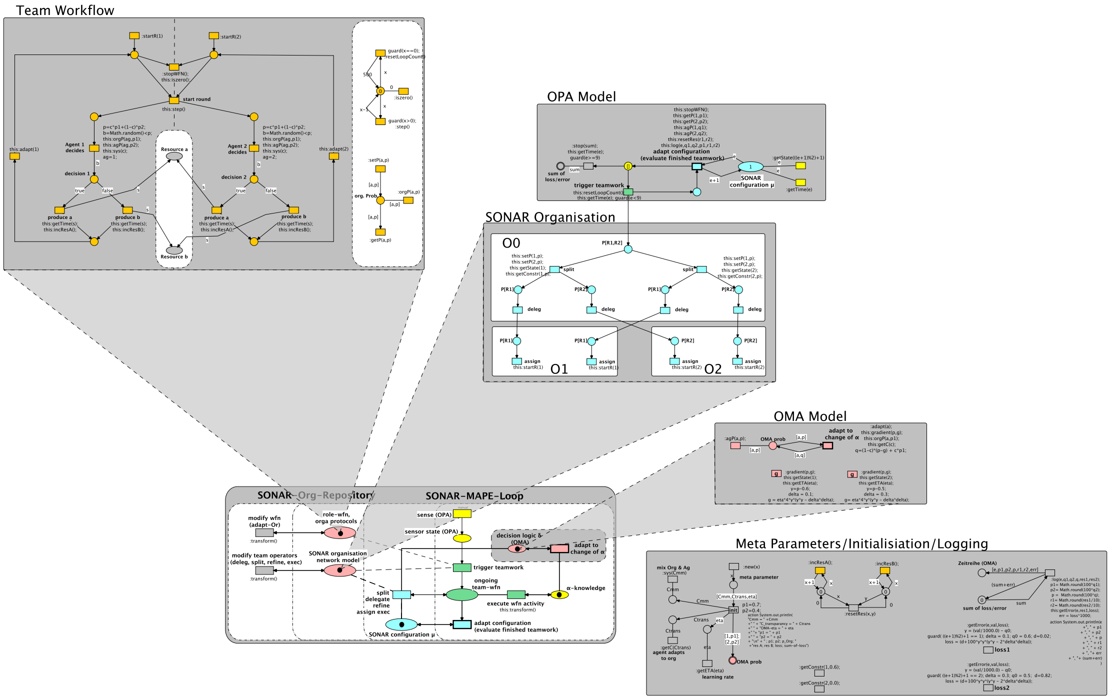


Figure 11: The Renew-Model of the Digital Twin

The source files are available at: https://github.com/koehler-bussmeier/digitaltwin/. The Renew simulator is available at: https://www.informatik.uni-hamburg.de/TGI/renew/.

The Sonar-MAPE-Loop

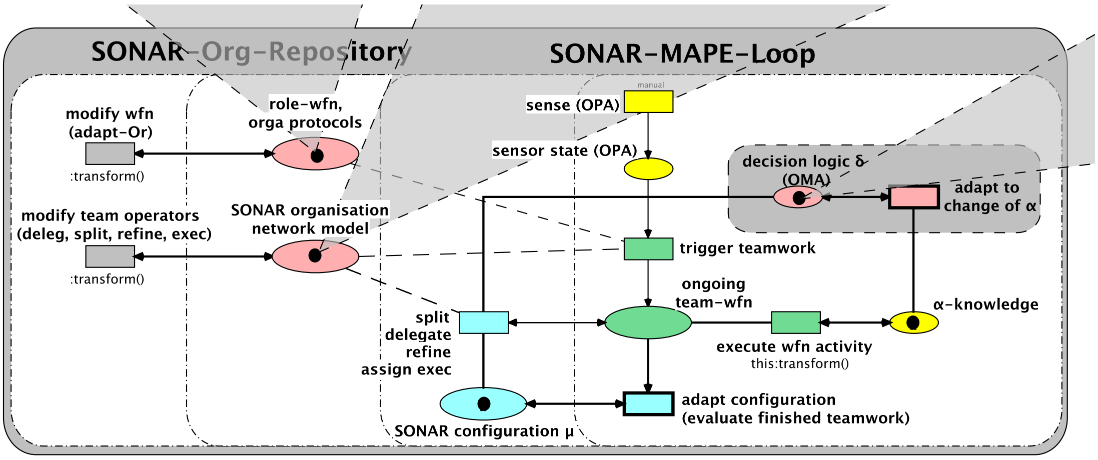


Figure 12: Model: The Sonar-MAPE-Loop

The core of the Sonar-engine is defined by the general *execution loop* (cf. Fig. 12). This net defines the overall process of the Sonar-MAPE loop, i.e., the formation of a team and the accompanied adaptation activities, i.e., adapt to change of α and adapt configuration μ.

Its main activity, the formation of teams, is started by the transition sense (OPA). It has an empty preset, as it is activated by the environment. Here, a *task* triggers the *formation* of a team G of agents. The transition trigger teamwork takes this sensor state (i.e. a trigger) and generates a team-wfn, i.e., a pair of a team and a workflow net (WFN) to handle the task that is triggered. The transition has the places role-wfn, orga protocols and SONAR organisation network model as side conditions, which are used as a 'database' to make a look-up. The organisation model N is a token on the place *Sonar organisation network model.* Each WFN from the set $\mathcal{D}$ is a net-token on the place *role wfn/protocols.*

The Sonar Position Agent (OPA)

The Position agent model is shown in Fig. 13. The main purpose of this net is to trigger the production teamwork, which is then handled by the organisation net. It also contains the logic that the environment oscillates each $s = 500$ production steps between the two different states, which have a different reward signal for the production (cf. Figure 4 (b)).

The Workflow P with Roles r_1 and r_2

The workflow net P (i.e. the interaction protocol) is shown in Fig. 14. It defines the interaction of two roles: r_1 and r_2 The left part of P belongs to r_1 and defines $P[r_1]$; the right part constitutes $P[r_2]$. In a team agents will be assigned to these roles.

Each role models a producer which has the choice between two possible goods: a or b. The Renew-model generates a random number (calling Math.random()) to solve the conflict

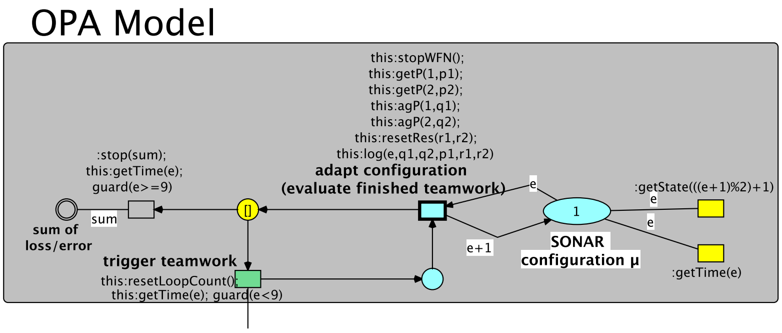


Figure 13: Model: The Sonar Position Agent (OPA)

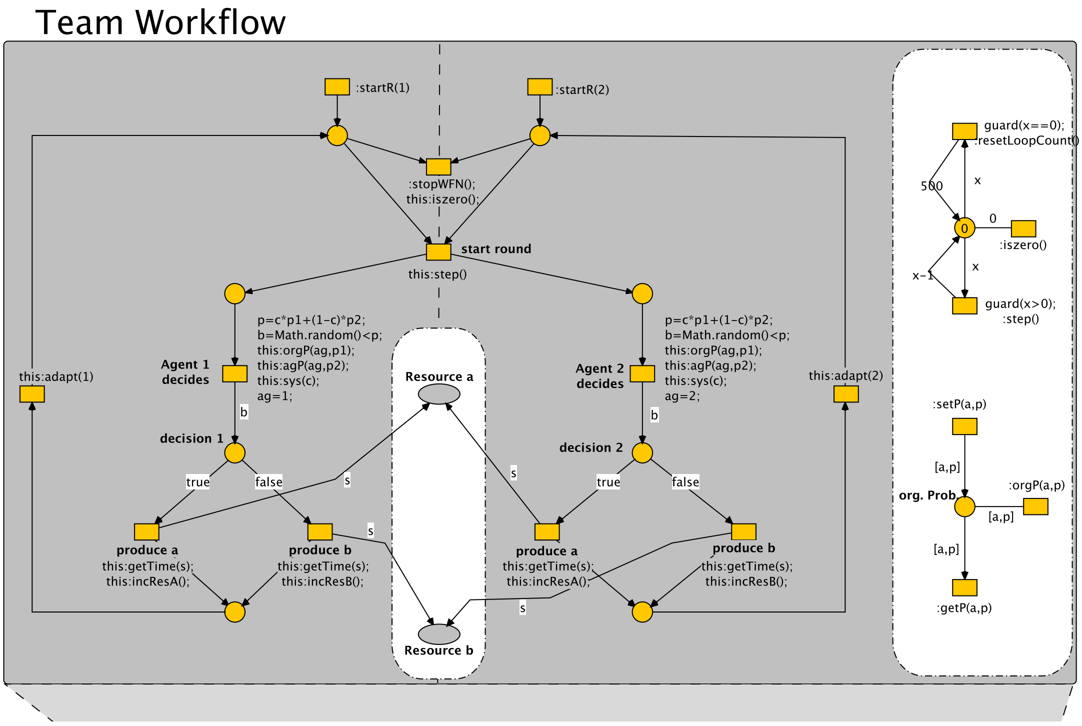


Figure 14: Model: The Workflow Net

between a and b. An agent implementing one of these roles has a preference order on these goods, modelled as a probability over a and b. Here, p_2 denotes the probability that the agent will choose to produce a. Additionally, the organisation defines a constraint on the production in the team formation; here, p_1 denotes the organisational constraint to produce a. Both probabilities are combined according to (1):

$$p = c_{org} \cdot p_{OPA} + (1 - c_{org}) \cdot p_{OMA}$$

In the net, the decision conflict is resolved randomly by the inscription Math.random() $<$ p.

The production steps of both roles are executed for $s = 500$ steps; the upper right part contains a counter place for the number of steps. After each step both agents will adapt their decision probabilities over the channel this:adapt.

The SONAR-Organisation Net

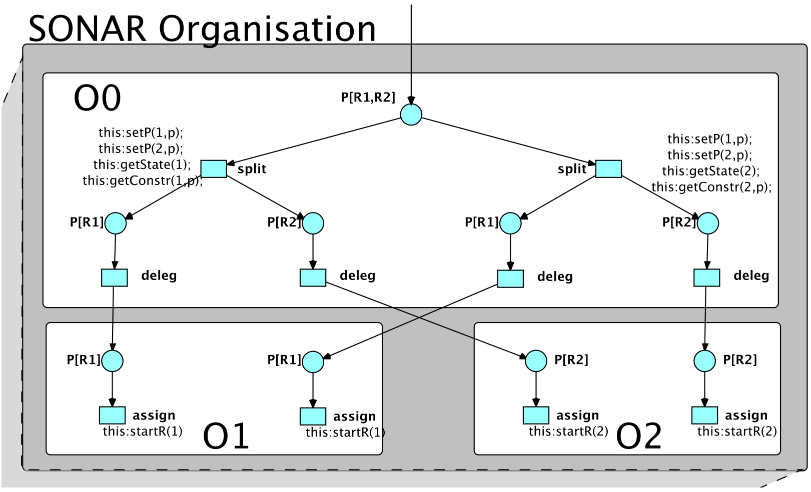


Figure 15: Model: The SONAR-Organisation Net

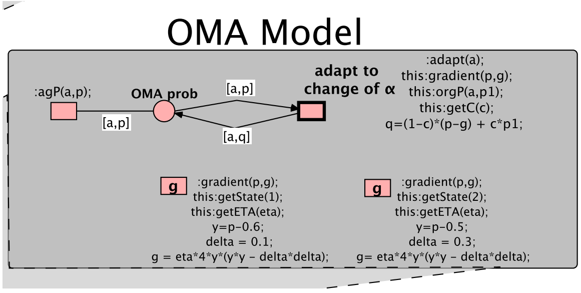
The organisation net N of Fig. 15. reacts upon exactly one task, which initiates the production of resources a and b. We have three OPAs: O_0, O_1 and O_2.

Whenever a task triggers a team-formation, the team is generated as an *ongoing* interaction between the MAPE-loop and an instance (i.e., the team) of the organisation net. This interaction takes place using the transition split, delegate, refine, assign exec.

The inner state of the team decides which concrete team construction steps are enabled Usually, we have more than one enabled transition, which, in general, leads to a combinatoric explosion of possible teams that could be generated by the organisation for a given trigger/task.

During this creation of teams the agents will be assigned to the roles of the team workflow, since, sooner or later, each of these formation processes must end with several firings of transitions named assign, drawn at the bottom of the organisation net.

The given organisation model can create exactly two teams for handling this task – one for each state of the environment. The main decision during the team formation (i.e. the decision which of the teams is chosen) is made by O_0. For both possible teams the agent O_1 is responsible for role r_1 and O_2 for r_2. The teams differ in the organisational constraints imposed on r_1 and r_2. The OPA O_0 adapts to the external change by switching between the two teams. Essentially this means that the production probability of good a is $p_{org} = 0.6$ for the first state and $p_{org} = 0.1$ for the second.



```
:adapt(a);
this:gradient(p,g);
this:orgP(a,p1);
this:getC(c);
q=(1-c)*(p-g) + c*p1;
```

```
:gradient(p,g);
this:getState(1);
this:getETA(eta);
y=p-0.6;
delta = 0.1;
g = eta*4*y*(y*y - delta*delta);
```

```
:gradient(p,g);
this:getState(2);
this:getETA(eta);
y=p-0.5;
delta = 0.3;
g= eta*4*y*(y*y - delta*delta);
```

Figure 16: Model: The Sonar Member Agent (OMA)

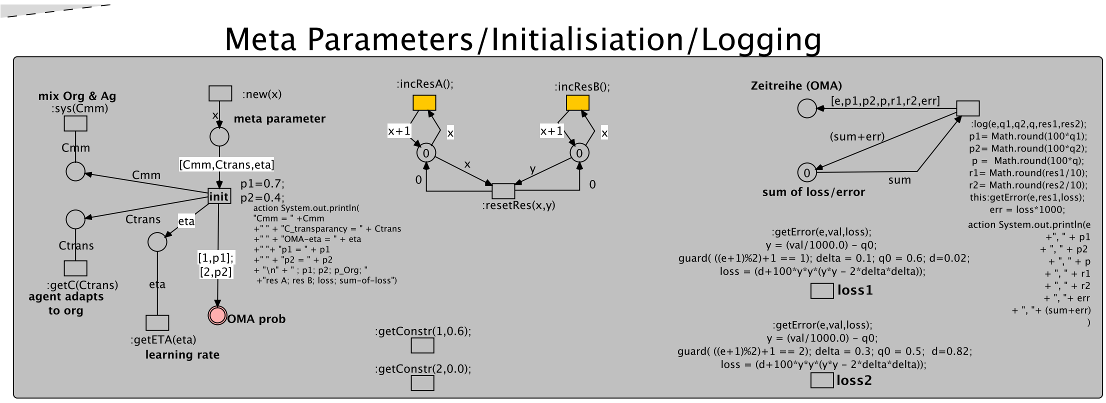
The Member Agent (OMA)

The member agent net is given in Fig. 16. Its main purpose is to implement the member agents' learning process as defined in (2):

$$p_i(t+1) = (1 - c_{tra}) \cdot \Big(p_i(t) - \eta \cdot g'(p_i(t))\Big) + c_{tra} \cdot p_{org}(t)$$

The two two transitions named g are used to compute the current gradient. We have two transitions here to handel the different parameters of the reward functions that we used to model the two states between which the environment oscillates.

Initialisation and Logging



```
p1=0.7;
p2=0.4;
  action System.out.println(
  "Cmm = " +Cmm
  +" " + "C_transparancy = " + Ctrans
  +" " + "OMA-eta = " + eta
  +" "+ "p1 = " + p1
  +" " + "p2 = " + p2
  + "\n" + " ; p1; p2; p_Org; "
   +"res A; res B; loss; sum-of-loss")
```

```
:log(e,q1,q2,q,res1,res2);
p1= Math.round(100*q1);
p2= Math.round(100*q2);
p =  Math.round(100*q);
r1= Math.round(res1/10);
r2= Math.round(res2/10);
this:getError(e,res1,loss);
err = loss*1000;
action System.out.println(e
                       +", " + p1
                      + ", " + p2
                       + ", " + p
                       + ", " + r1
                       + ", " + r2
                       + ", "+ err
                  + ", "+ (sum+err)
                                  )
```

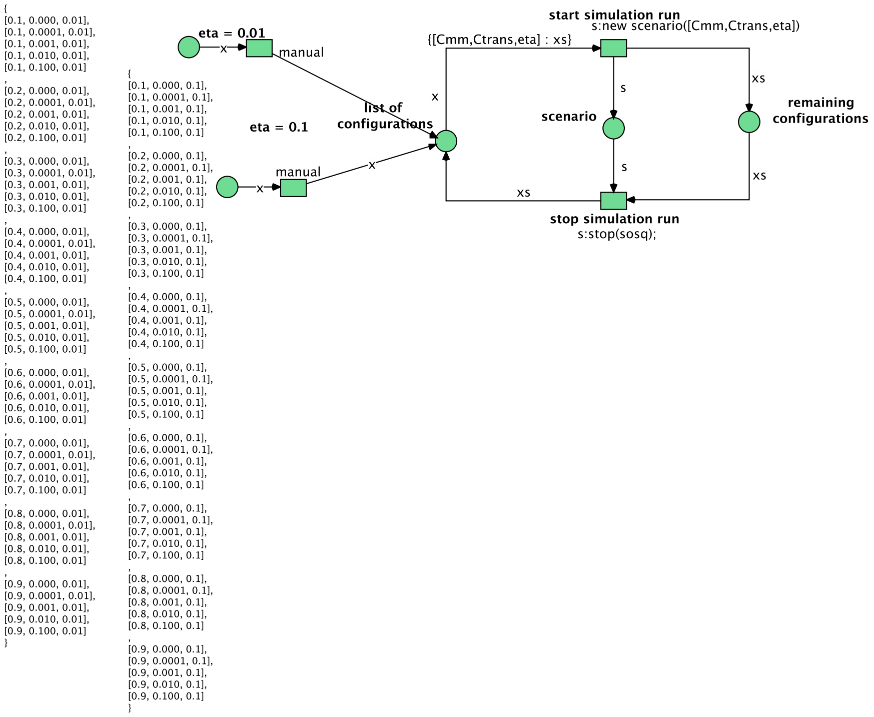
```
:getError(e,val,loss);
y = (val/1000.0) - q0;
guard( ((e+1)%2)+1 == 1); delta = 0.1; q0 = 0.6; d=0.02;
loss = (d+100*y*y*(y*y - 2*delta*delta));
```

```
:getError(e,val,loss);
y = (val/1000.0) - q0;
guard( ((e+1)%2)+1 == 2); delta = 0.3; q0 = 0.5;  d=0.82;
loss = (d+100*y*y*(y*y - 2*delta*delta));
```

Figure 17: Model: Initialisation and Logging

The model also contains helper code to set-up the meta-parameters c_{org}, c_{tra}, and η (left part of Fig. 17) and to report the number of produced goods as an output to stdout (right part of Fig. 17). This log data is used to produce the charts in Section 6.

The Experimental Setup: Parameter Configurations



```
{
[0.1, 0.000, 0.01],
[0.1, 0.0001, 0.01],
[0.1, 0.001, 0.01],
[0.1, 0.010, 0.01],
[0.1, 0.100, 0.01]
,
[0.2, 0.000, 0.01],
[0.2, 0.0001, 0.01],
[0.2, 0.001, 0.01],
[0.2, 0.010, 0.01],
[0.2, 0.100, 0.01]
,
[0.3, 0.000, 0.01],
[0.3, 0.0001, 0.01],
[0.3, 0.001, 0.01],
[0.3, 0.010, 0.01],
[0.3, 0.100, 0.01]
,
[0.4, 0.000, 0.01],
[0.4, 0.0001, 0.01],
[0.4, 0.001, 0.01],
[0.4, 0.010, 0.01],
[0.4, 0.100, 0.01]
,
[0.5, 0.000, 0.01],
[0.5, 0.0001, 0.01],
[0.5, 0.001, 0.01],
[0.5, 0.010, 0.01],
[0.5, 0.100, 0.01]
,
[0.6, 0.000, 0.01],
[0.6, 0.0001, 0.01],
[0.6, 0.001, 0.01],
[0.6, 0.010, 0.01],
[0.6, 0.100, 0.01]
,
[0.7, 0.000, 0.01],
[0.7, 0.0001, 0.01],
[0.7, 0.001, 0.01],
[0.7, 0.010, 0.01],
[0.7, 0.100, 0.01]
,
[0.8, 0.000, 0.01],
[0.8, 0.0001, 0.01],
[0.8, 0.001, 0.01],
[0.8, 0.010, 0.01],
[0.8, 0.100, 0.01]
,
[0.9, 0.000, 0.01],
[0.9, 0.0001, 0.01],
[0.9, 0.001, 0.01],
[0.9, 0.010, 0.01],
[0.9, 0.100, 0.01]
}
```

```
{
[0.1, 0.000, 0.1],
[0.1, 0.0001, 0.1],
[0.1, 0.001, 0.1],
[0.1, 0.010, 0.1],
[0.1, 0.100, 0.1]
,
[0.2, 0.000, 0.1],
[0.2, 0.0001, 0.1],
[0.2, 0.001, 0.1],
[0.2, 0.010, 0.1],
[0.2, 0.100, 0.1]
,
[0.3, 0.000, 0.1],
[0.3, 0.0001, 0.1],
[0.3, 0.001, 0.1],
[0.3, 0.010, 0.1],
[0.3, 0.100, 0.1]
,
[0.4, 0.000, 0.1],
[0.4, 0.0001, 0.1],
[0.4, 0.001, 0.1],
[0.4, 0.010, 0.1],
[0.4, 0.100, 0.1]
,
[0.5, 0.000, 0.1],
[0.5, 0.0001, 0.1],
[0.5, 0.001, 0.1],
[0.5, 0.010, 0.1],
[0.5, 0.100, 0.1]
,
[0.6, 0.000, 0.1],
[0.6, 0.0001, 0.1],
[0.6, 0.001, 0.1],
[0.6, 0.010, 0.1],
[0.6, 0.100, 0.1]
,
[0.7, 0.000, 0.1],
[0.7, 0.0001, 0.1],
[0.7, 0.001, 0.1],
[0.7, 0.010, 0.1],
[0.7, 0.100, 0.1]
,
[0.8, 0.000, 0.1],
[0.8, 0.0001, 0.1],
[0.8, 0.001, 0.1],
[0.8, 0.010, 0.1],
[0.8, 0.100, 0.1]
,
[0.9, 0.000, 0.1],
[0.9, 0.0001, 0.1],
[0.9, 0.001, 0.1],
[0.9, 0.010, 0.1],
[0.9, 0.100, 0.1]
}
```

Figure 18: Model: Sweep of Parameters

For the exploration of design candidates (as done in Section 7) we generate a simulation run for each configuration of the meta-parameters – as given in the initial marking.

We use another Petri net to manage this batch of experiments (shown in Fig. 18). Here, we evaluate all combinations for a parameter sweep for all combinations of $c_{org} = 0.1 \ldots, 0.9$ and $c_{tra} = 0, 0.0001, 0.001, 0, 01, 0.1$ – once for $\eta = 0.01$ and then for $\eta = 0.1$. For each of these configurations a fresh instance of the simulation is started and evaluated.