Electronic Communications of the EASST
Volume 27 (2010)

Workshop über Selbstorganisierende, adaptive, kontextsensitive verteilte Systeme (SAKS 2010)

Systemic Modeling of Agent Coaction: A Catalog of Decentralized Coordinating Processes

Jan Sudeikat[2] and Wolfgang Renz

12 pages

Guest Editors: Klaus David, Michael Zapf
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Systemic Modeling of Agent Coaction: A Catalog of Decentralized Coordinating Processes

Jan Sudeikat[1,2] **and Wolfgang Renz**[1]

[1] jan.sudeikat; wolfgang.renz @haw-hamburg.de
Multimedia Systems Laboratory, Faculty of Enigneering and Informatics,
Hamburg University of Applied Sciences, Berliner Tor 7, 20099 Hamburg, Germany

Abstract: Taking inspiration from natural self-organizing systems is a successful strategy to solve computational problems in distributed systems. Faced with a particular problem, application designers have to identify an appropriate dynamical behavior and decide how to induce similar behavioral modes. In order to consolidate these ad-hoc activities to a systematic dynamical design method, we discuss and exemplify a behavioral modeling approach that describes the macroscopic behavior of agent-based software systems. This formalism is used to catalog the dynamic behavior of prominent examples of natural self-organizing systems. These here presented models represent generic, reusable templates for decentralized system adaptations that serve as analysis templates for application designs. A tailored programming model allows to supplement these templates in agent-based software applications with minimal intervention in the agent models.

Keywords: Self-Organization, Systemic Modeling, Process Template

1 Introduction

The research project "Selbstorganisation durch Dezentrale Koordination in Verteilten Systemen"[3] (SodekoVS) addresses the preparation of nature-inspired dynamics as reusable design elements for software engineers. The conception of this project was reported in the last year's issue of this workshop [SBP+09]. Coordinating processes that have been found in natural self-organizing systems are understood as reusable templates that describe field-tested processes of inter-agent coordination. Within this research project, a tool set is elaborated (cf. Section 2) that allows to treat these processes as design elements, i.e. artifacts for the incremental revision and integration in applications. Part of this tool set is a modeling approach that expresses processes as structures of feedback loops among system entities.

Self-organizing phenomena can be explained by distributed feedbacks (e.g. [BDT99]) and here, we use a corresponding modeling stance to catalog nature-inspired, inter-agent coordination processes. This catalog aggregates processes that have been discussed in literature. The aim is an (initial) collection of generic processes that can be straightforward reused in MAS development. This is reflected by the abstraction level of these templates. Templates are described

[2] Jan Sudeikat is doctoral candidate at the Distributed Systems and Information Systems (VSIS) group, Department of Informatics, Faculty of Mathematics, Informatics and Natural Sciences, University of Hamburg, Vogt–Kölln–Str. 30, 22527 Hamburg, Germany, jan.sudeikat@informatik.uni-hamburg.de

[3] SelfOrganisation by Decentralized Coordination in Distributed Systems

as generic sets of inter-agent feedbacks. The participants within these feedbacks can be detailed to match application elements. The detailing and embedding of process models is exemplified ([SR09b, SR09c, SR09e], cf. Section 2.2) and systematized [SR09d] in earlier works.

This paper is structured as follows. In the next section, the conceptual model of a self-organizing application and the hitherto elaborated tool set is outlined. Afterwards, in Section 3, coordination templates are cataloged. Besides the structural and dynamic properties of these templates are contrasted and discussed. Finally, we conclude and give prospects for future work.

2 Systemic Modeling and Programming of Agent-Coaction

Decentralized, self-organizing processes are caused by the presence of mutual feedback loops among system components, e.g. discussed in [BDT99]. Early works focused on the analysis of self-organizing processes to facilitate the purposeful redesign of application to expedite the rise of intended system phenomena [SR08a]. These works lead to a *systemic* modeling approach to decentralized inter-agent coordination in Multi-agent Systems (MAS) [SR09b, RS08].

This modeling technique transfers *System Dynamics* [Ste00] concepts to the description of agent-based software systems. The macroscopic system configuration is denoted by a set of system variables. Each variable is represented by a graph node and represents the number of agents that exhibit a certain behavior or the quantitative values of environment properties. Different node types can be used to represent behavioral abstractions that are provided by agent-oriented design techniques, e.g. role or group concepts [SR09b]. In addition, interaction rates can be denoted by graph nodes as well. The dynamic relations between these variables are denoted by links, that represent either causal interdependencies or additive/subtractive influences. In, addition, it can be indicated whether an influence is based on an specific interaction mechanism, e.g. mediated by environment models [DH07] (cf. Section 2.1). In Section 3, the graph-based description level is used to illustrate template coordinating processes. The graphical notation is exemplified in the Figures 2, 3, and 4.

In the SodekoVS project, this modeling approach is transferred to a corresponding programming model that allows to describe decentralized processes, which affect the coordination of entity activities, as structures of influences among agent-behaviors. In [SR09b], a corresponding configuration language is discussed that allows to configure the exhibition of these processes. This language supports two description levels. First, inter-agent interdependencies are descried by directed graphs. These graphs describe application-independent structure of feedback loops among system entities that make up coordinating process. Secondly, the process models can be mapped to agent implementations and the dissemination of influences can be configured.

An architectural model for the enactment of the configured process instances is outlined in [SBP+09] and a realization is discussed in [SR09a]. This architecture follows a layered design and the enactment of inter-agent coordination is outsourced to a middleware layer that encapsulate the activities that are conceptually related to inter-agent coordination [SR09a]. The systematic utilization of this programming model is instructed by a set of reusable development activities [SR09d] that supplement conventional development, using *Method Engineering* [CGGS07].

2.1 Building-In Decentralized Coordination

The operation of autonomic computing systems can be distinguished, between (1) the *Objective* to be achieved by the management, (2) the *mechanisms* that are actuated to achieve the objective, and (3) *strategies* that prescribe and order the activations of mechanisms (e.g. in [JPR09]).

The design of a self-organizing application follows a comparable structure (cf. Figure 1). The design objective is to equip the developed *Application*, here a MAS, with an *Adaptation Dynamic*, i.e. a behavioral mode that makes the system adapt to external influences. Adaptivity is understood as a black-box property [Zad63] at the macroscopic system level that makes systems respond to changes in their execution context. Following a decentralized approach, the system is adjusted due to collective, concurrent adaptations of the individual system entities. This collective behavior is prescribed by a *Coordinating Process*. Systemic models (cf. Section 2) describe the structure of the inter-agent feedbacks that manifest these processes (*Systemic Process Model*). The described structures can give rise to a space of process realizations, due to varying parameterizations, e.g. interaction rates, thus the actually exhibited process is a concrete instantiation of the process structure. A successful approach to conceive these processes by resembling the dynamics in natural systems [MMTZ06]. The realization of these processes makes use of *Coordination Mechanisms* (e.g. reviewed in [SR08a] that provide models to control the *Information Exchange* [DH07] and *Local Entity Adaptation* [SGK06]. The former ones define how information is stochastically disseminated and diluted while the latter ones configure how individual entities respond to the communicated information. The enacted strategies define which mechanisms are to be embedded in the application and instruct their actuation.

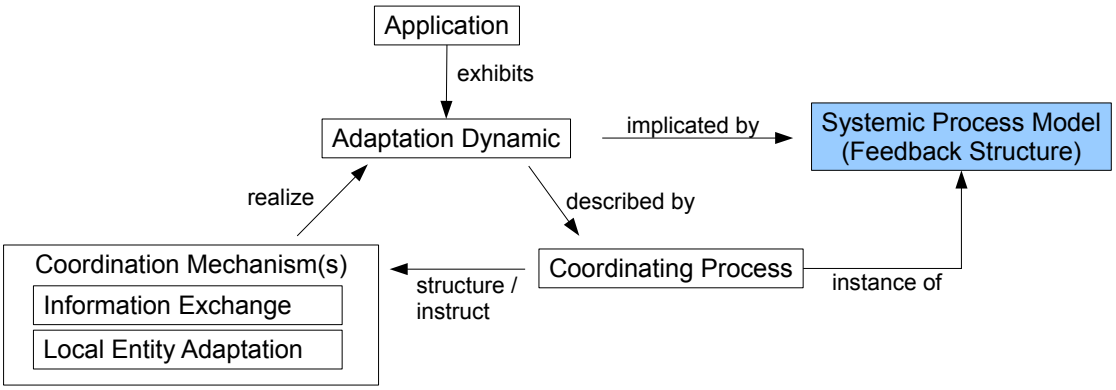

Figure 1: The relations between an Adaptation Dynamic, the prescribing Coordination Process, and the actuated Coordination Mechanisms.

2.2 Case Studies

The design approach and the tool set have been evaluated in several case studies. Here, we outline two studies that exemplify the minimal-invasive integration of pattern. The systematic conception of feedback loop structures is discussed in [SR09d, SR09c]. For the sake of brevity,

the integration of the pattern, using the outlined programming model is not demonstrated, but the systems objective and the rationale to integrate template processes are discussed. Details on the system realizations have been published in [SR09b, SR09c].

2.2.1 Bee-Inspired Server Management

In [SR09e], a decentralized management architecture for J2EE application servers is presented. Glassfish[4] application servers are controlled by Jadex-based[5] agents via the *Appserver Management EXtensions*[6] (AMX). This exemplifies agent-based management approaches to the management of computational infrastructures. A bee-inspired mechanism for the dynamic allocation of (web) servers is presented in [NT04]. The design objective is to balance the deployments of services on servers with the fluctuating demands for the individual service types. In [SR09b], this scenario is adopted to exemplify the utilization of the systemic programming model (cf. Section 2) and the tool set is used to configure a decentralized process that resembles the dynamics of foraging bee societies. The corresponding template is discussed in Section 3.2 (S6).

Following the inspirational source, the agent society is separated into a population of scouting and foraging agents. Scouts metaphorically explore the environment by serving random client requests. When they observe increases in the demand of specific request types, this demand is gradually disseminated to the *foraging* servers that are allocated to a specific request type and autonomously reconfigure themselves, based on the perceived demands [SR09b].

2.2.2 Information Dissemination Processes

In [SR09c], the embedding of dissemination process [EGKM04, ST97] in MAS is discussed. These processes are integrated in a homogeneous MAS and is enacted without modification of the original agent models. The processes are integrated as background processes that modify the knowledge base of their surrounding agent coefficient to the agent execution. The first process concerns the integration of *convention emergence* [ST97]. Convention emergence describes phenomena, where agents mutually inform each other about their local configuration (cf. Section 3.3). Based on the perceived information and their local policy, e.g. a majority rule, agents adjust their configuration. When the communicativeness and the adaptivity of agents are well-matched, set so agents can agree on a coherent configuration value, solely based on their local interactions. Secondly, the *epidemic* spreading of information [EGKM04] (cf. Section 3.3) is exemplified. The metaphoric spreading of an infection is used to distribute patches in a set of agents and the competitive spreading of two dissemination processes is studied.

3 A Template Catalog

The design of Coordinating Processes (cf. Figure 1) on the drawing board is a laborious process that requires extensive system simulations to ensure that the conceived processes give rise to

[4] https://glassfish.dev.java.net/
[5] http://jadex.informatik.uni-hamburg.de
[6] https://glassfish.dev.java.net/javaee5/amx/index.html

the intended system phenomena. In this respect, nature-inspired processes can serve as field-tested templates. In order to prepare the reuse of these templates in application designs, these are modeled, using the systemic modeling technique from [SR09b].

The here presented catalog extends earlier works on the modeling of feedback loops structures in nature-inspired agent coordination [SR08b]. In alphabetic order, the considered templates are: *Brood Sorting* (S1) [MMTZ06], *Convention Emergence* (S2) [ST97], *Epidemics* (S3) [EGKM04], *Flocking* (S4) [MMTZ06], *Ant-based Foraging* (S5) [MMTZ06], *Bee-based Foraging* (S6) [NT04], *Molding* (S7) [MMTZ06], *Morphogenesis* (S8) [MMTZ06], *Nest-Building* (S9) [MMTZ06], *Quorum* (S10) [MMTZ06], and *Web Weaving* (S11) [MMTZ06].

The systemic modeling level expresses the structures of feedback loops that are present in the macroscopic observable system behavior (cf. Section 2). These feedbacks result from circular sequences of interdependences and influences. These are either *reinforcing* (+) as fluctuations of system variables are amplified or *balancing* (-), due to the attenuation of fluctuations. In the following, processes are distinguished by their *Regulation Polarity* (RP) that is given by their prevalent feedbacks. *Amplifying* processes contain a majority of reinforcing feedback while *compensating* processes contain a majority of balancing feedbacks. In *selective* processes, both types of feedbacks are evenly featured. This is a purely structural property.

		Decentralized Coordination Strategy										
		S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11
RP (I)	amplifying					$-/+^{2}$		$+$		$+^{2}$		$+$
	compensating	$-^{2}/+$			$-^{n}$		$-^{2}/+$		$-$			
	selective		$-/+$	$-^{(2)}/+$							$-/+$	
T	configuration		x	x	x		x	x	x		x	
	process	x				x				x		x
A	endogenous		x	x					x			x
	extrinsic	x			x	x	x	x		x	x	
S	differentiation		x	x	x		x		x			
	synchronization							x			x	
	external	x				x				x		x
C	coherence	x	x	x		x	x	x			x	x
	partitioned	x			x		x		x	x		

Table 1: Classification of Coordination Strategies, according to the regulating polarity of loop structures (RP) and the properties of the resulting structures (T,A,S,C).

The feedback models are used to describe templates of processes that, show self-organization, i.e. generate and maintain structures. For the selection and comparison of processes an additional set of classification criteria is presented that characterizes the types of structures and their run-time adjustments. First, two principal *Types* (T) of the self-organized structures can be distinguished. Decentralized processes can be utilized to either control the configurations of system entities, e.g. establish conventions and coalitions, or establish a collaborative *process* among the system elements, whereby the collaboration of agents is enforced by structuring the chronology of agent activities. The adjustment of the established structures is another fundamental property and this *Adaptivity* (A) is characterized by the loci of the causes for restructuring. *Endogenous* reconfigurations result from process internal processing and oppose the *extrinsic* adaptivity

where systems respond to external changes in the context of the software system. Furthermore, the Subject (S), i.e. the system quality, that are affected by Coordinating Processes are characterized. The *differentiation* refers to the partitioning of system elements, i.e. as the homogeneous MAS are specialized to distinct castes or segregated into distinct groups. The *synchronization* describes the creation of timely structures of agent activities and the *external* refers to the adjustment, i.e. structuring, of the systems environment. Finally, processes can be distinguished the the *Composition* (C) of the established and maintained structure. Structures either span the whole software system *coherence* or partition the system into locally coherent structures (*partitioned*). The attribution of processes to these criteria is summarized in Table 1.

3.1 Amplifying Processes

Amplifying processes contain a majority of reinforcing feedback loops, thus the adjustments of macroscopic structures are induced by the amplification of system variables that propagate through the system. Four examples are illustrated in Figure 2. The *Ant-based Foraging* (S5) template describes the metaphorical formation of trails between environment locations as found in ant colonies. Agents that are not bound to a trail are *Searching* the environment. The agents that transport a resources to their home base communicate their activity via a Coordination Mechanism [DH07], i.e. digital pheromones. This communications contribute to a globally observable *binding* rate as searching agents occasionally perceive these communications and get aware of resource locations. Aware agents follow pheromone gradient (*Trailing*), encounter a resource location (*pickup*) and subsequently *transport* resources to the home base (*delivery*). These transports are associated with a delay ($||$). The *Resource Availability* given by the environment state, contributes to the serendipitous encounter of resources by agents. The removal of resources in governed by two feedbacks (β, γ) that establishes a collaborative *process* among agents that repeatedly search and transport resources. These activities are controlled by an extrinsic factor, i.e. the availability of resources and the process addresses the *coherent* modification of the *external* environment as resources are gradually redeployed.

Molding (S7) is found in protozoic life forms. These individuals are either part of a larger cluster (*Aggregation*) or are individually foraging resources (*Autonomous Behavior*). The rates of *clustering* and leaving (*unclustering*) agents are controlled by the availability of resources in the systems environment. The exhibited process is controlled by a single feedback loop that controls the *coherent configuration* of agents according to the *extrinsic* resource availability. The changes of agent configurations are *synchronized*.

The Nest-building (S9) within termite colonies is a prominent example for distributed assembly processes. Agents utilize environment resources to generate building blocks (*Brick Creation*). Subsequently, individuals carry these bricks and search for locations for their deposit (*Brick Creation*). An effective mechanism is the enabling of bricks to communicate their placement, e.g. via digital pheromones (*Brick Communication*). Transporting agents are attracted and place building block nearby (*Brick Deposit*). The supply of building block is controlled by a feedback loop (α) that balances the generation of building blocks with the available resources. The sites of brick deposits compete for the generated resources and larger congregations of construction elements are enforced as deposits contribute to the communication, respectively attraction, of newly produced artifacts (β). The agent activities are arranged to show a collaborative *process*

that modifies the *extrinsic* system environment by transporting and depositing items. These activities responds to the availability of *external* resources. The process supports the diversity of structures in the environment *partitioned*, due to the emerging of differing deposit sites.

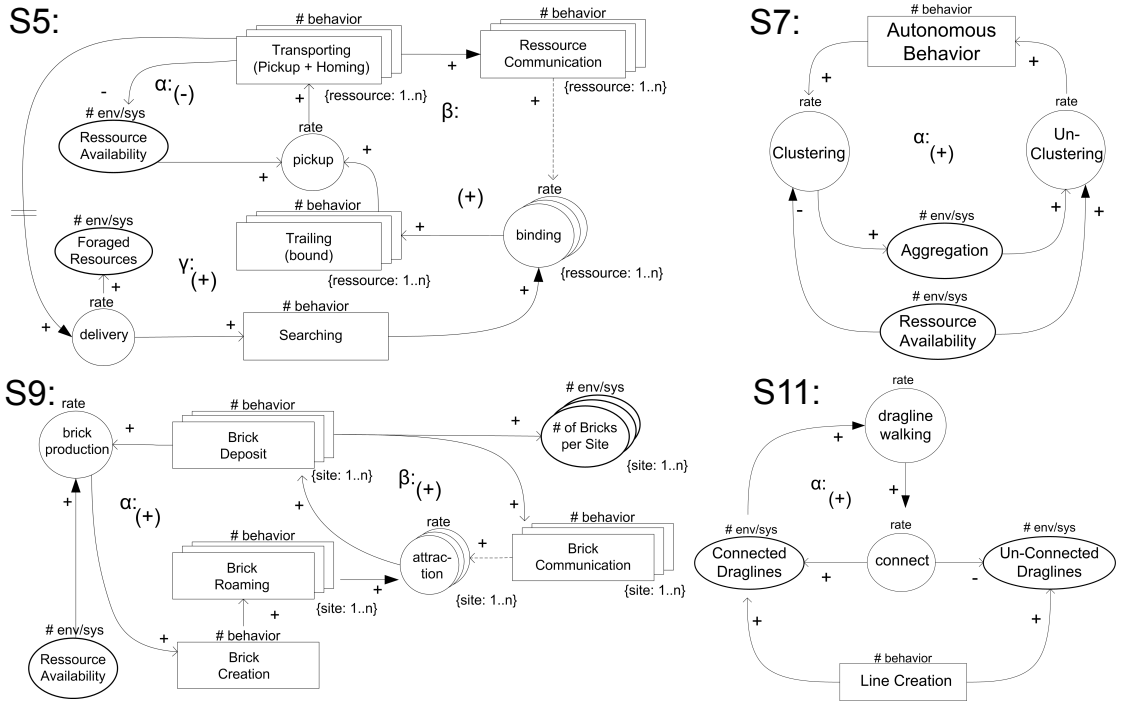

Figure 2: Amplifying Pattern.

Web Weaving (S11) resembles the network creation by spider species, due to the sequential connecting of ground locations with draglines. The basic activity is the creation of lines (*Line Creation*). These lines are used to connect locations that are already reachable in the network (*Connected Draglines*) as well as so far unconnected locations (*Un-Connected Draglines*). Since spiders prefer to walk within their already established network of connections (*dragline walking*), the connection of connected locations is enforced. The web creation is a *process* that is governed by a single feedback loop. The created graph is an *coherent* element in the *external* environment that results from an *endogenous* stimulus, i.e. the creation activity of agents.

3.2 Compensating Processes

These template processes compensate fluctuations of system variables, thus these template processes are applicable to the maintenance of continuously perturbed structures. The first example is the *Brood Sorting* (S1) that is exhibited by insect colonies. Agents randomly explore their nest (*Wandering*) and occasionally encounter environment elements, e.g. offspring in different stages (egg, larvae, etc.). Isolated elements are picked up and transported (*Transportation*) till similar items are encountered. The deposits reduce the dispersion of environment elements. The agent

activities manifests a balancing feedback (α) that affects the systems environment. Two auxiliary feedbacks control the movement of items. Transports are balanced with the amount of available agents (γ) and are enforced by the deposits of items (β). Transports form a *process* within the agent population that modifies the extrinsic environment and responds to the perception of the *external* diversity of elements. The deposit logic can be configured to allow for *coherent* or *partitioned* structures.

The *Flocking* (S4), a.k.a. *schooling* or *herding*, is a prominent self-organizing phenomenon that describes movement pattern of bird and fish swarms. These pattern emerge when individuals maintain sets of highly fluctuating properties. Agents mutually observe a properties of their neighboring agents, e.g. their speed and heading, and adjust their local configuration (*Adjustment*) to minimize deviations (*disagreement*). In addition, certain invariants, e.g. the minimal distance to neighbors is maintained (*Maintenance*) and the corresponding corrections introduce additional deviations ((*self.property != destination.property*)) and imply delays (||). The system exhibits a number of *n* feedbacks, one feedback per maintained property (α) and two feedbacks per maintained invariant (β, γ). The subsequent adjustments by agents are *synchronized* to control the *configurations* of individuals and responds to *extrinsic* fluctuations. Consequently, the system continuously approaches a *coherent* configuration.

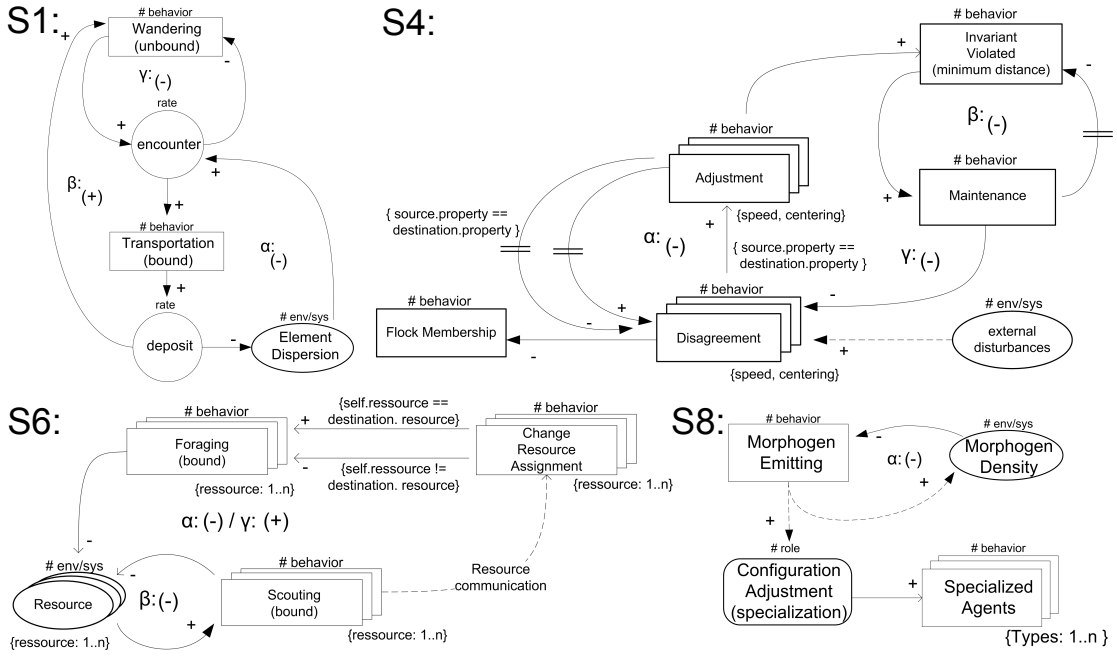

Figure 3: Compensating Pattern.

Bee-based Foraging (S6) resembles the resource gathering by honey-bee colonies. Scouts-bees wander the environment and search for resource locations (*Scouting*) while Forager-bees are associated to specific resource locations and repeatedly transport resources to the home base (*Scouting*). When resource locations are serendipitously encountered, scouts return to the

nest and communicate their findings via *waggle dances* to foragers. Foraging agents decide autonomously (*Change Resource Assignment*), e.g. based on communicated quality criteria, if they adjust their association to a depletion site (*Foraging*). The change of an association is modeled as the removal of foragers with other previous associations (*self.resource != destination.resource*) and the addition of foragers with the updated association (*self.resource == destination.resource*). The removal or resources is governed by two feedback loops (α, β). In addition, the allocation of foragers to resource locations are adjusted by a reinforcing feedback (γ). This template prescribes the *differentiation* of agent *configurations*, i.e. the associations of forgers. These associations are updated, due to the *extrinsic* availability of resources. The process can be configured to exhibit both the convergence to one globally *coherent* association of foragers, or their segregation to differing locations (*partitioned*).

Morphogenesis (S8) is another approach to control the differentiation of agents. An initially homogeneous set of agents is influenced by a subset of agents that distribute messenger substances (*Morphogen Emitting*). The emitters maintain a certain density at their position. Other agents sense these substance and infer their relative position, based on the gradients of the perceived morphogens. Based on this information they configure themselves (*Configuration Adjustment*) and convert to specialized agent-instances (*Specialized Agents*). This template manifests a feedback loop that controls the density of the morphogens. Supporting different types of messenger substances multiplies this feedback. The reconfiguration of agents is a result of the locally perceived substance density. Thus the *endogenous differentiation* of agents is controlled, leading to *partitioned* structures of agent *configurations*.

3.3 Selective Processes

These process templates prescribe balanced sets of feedback types, thus the system history decides which system properties are amplified and discriminated. *Convention Emergence* (S2) describes the establishment of global consensus on a particular value. The operation of agents (*Activity*) is influenced by their local configuration value. The process template prescribes that a side effect of the triggering of these activities, e..g when these are participate in interactions, is the communication of their local configuration to other agents. The receivers of these values adjust their local configuration (*Convention Adjustment*) in order to agree with the majority of agents in their neighborhood. Adjustment affect the operations of agents. Figure 4 (S2) illustrated the template for two values. One feedback (α) amplifies the spread of a certain value, while the opposing feedback (β) discriminates the other values. A wider range of values would introduce additional balancing feedbacks but the theme of the template is that a randomly selected value is enforced. Thus a *coherent configuration* of agents is generated, due to the *endogenous* process that concerns the *differentiation* of agents.

Epidemics (S3) is nature-inspired approach to the dissemination of information in distributed systems. Agents are either susceptible or infectious. Infections metaphorically describe the transfer of information between agents and a macroscopic infection rate describes spreading within a set of agents. Gradually, the susceptible agent configurations are removed (α) and become infectious (β). The system tends towards a globally *coherent configuration* where all agents are infected. The infections are triggered by the *endogenous* functioning of the process and agents are *differentiated*. An important variation of this scenario is the introduction of a

recovering process. Infectious agents recover after infections and are afterwards insusceptible to infections. The recovery resembles the processing of perceived information (infections). This introduces an additional balancing feedback (γ). Consequently the system tends toward a global configuration where all agents are recovered. The behavioral properties of the process are not effected by this extension.

Quorum is a phenomenon that can be observed when the activities of individuals stimulate coherent activities of neighboring agents. *External perturbations* drive agents out of sync (*Non-Coherent*). The coherence of activities is reestablished as agents perceive the activities of their neighbors and align themselves (*Adjustment*). A balancing feedback (α) counter-balanced perturbations. Thus the system responds to *extrinsic* influences by reestablishing a *coherent* configuration of agents that manifests the *synchronization* of agent activities.

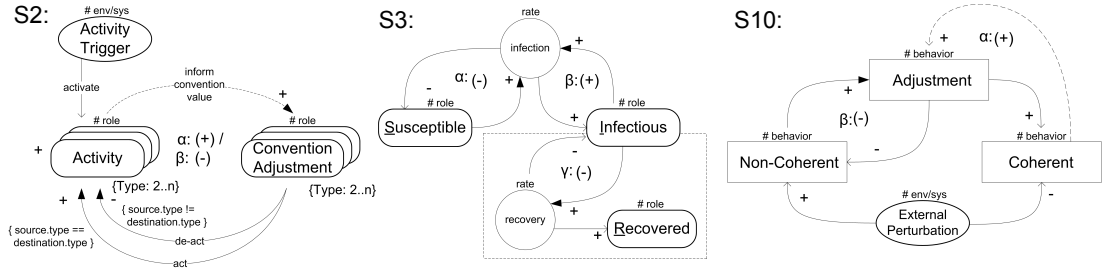

Figure 4: Selective Processes.

3.4 Associating Template Processes with Behavioral Properties

The presented template-processes factor out the details of natural self-organizing systems but provide abstract representations of decentralized processes that developers can embed in distributed applicators to realize decentralized coordination. These templates are associated to behavioral properties to support their comparison and selection.

The presented classification criteria justify the adoption of the bee-based foraging template, as outlined in Section 2.2.1. The deployments of a web-services changes the local configuration of the system entities, here servers. These *differentiate* by their local set of services that are offered. Since these configurations are to be structured and adjusted, in response to demand changes, process-establishing templates are not applicable. Unpredictable fluctuations of demands are an *extrinsic* influence that the management has to respond to. The redeployments of services are internal to the system and do not affect the systems environment. Thus process that concern the modification of *external* environments, e.g. system resources are not applicable as well. Finally, the applied coordination processes has to allow for *partitioned* structures, since the allocations of different service types are required.

The examined information dissemination processes (cf. Section 2.2.2) share the same properties of the prescribed structure-establishment. Since both processes prescribe an *endogenous differentiation* of agents, these are appropriate to be embedded as background processes within MAS. Equipping MAS with these processes can be used to ensure the *coherent* distribution of information, independent from the application-domain and the data that are exchanged.

4 Conclusions

In this paper, prominent examples for nature-inspired self-organizing processes are cataloged according to their underlying feedback structure. These templates show the principled structure of the processes and guide their integration, using a systemic modeling approach (outlined in Section 2). These processes describe structure-establishing dynamics and phenomenologic criteria for the characterization of the adaptive structuring are presented as well. These criteria facilitate the selection of processes for specific applications. Future work, concerns guidelines for the systematic selection and combination of process templates.

Acknowledgements: We would like to thank the *Distributed Systems and Information Systems* (VSIS) group at Hamburg University, particularly Winfried Lamersdorf, Lars Braubach and Alexander Pokahr, and Ante Vilenica for inspiring discussion and encouragement.

The project *Selbstorganisation durch Dezentrale Koordination in Verteilten Systemen* (SodekoVS) is funded by the German Research Council (Deutsche Forschungsgemeinschaft, DFG).

Bibliography

[BDT99] E. Bonabeau, M. Dorigo, G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Santa Fe Institute Studies on the Sciences of Complexity. Oxford University Press, 1999.

[CGGS07] M. Cossentino, S. Gaglio, A. Garro, V. Seidita. Method fragments for agent design methodologies: from standardisation to research. *Int. J. Agent-Oriented Software Engineering* 1(1):91–121, 2007.

[DH07] T. DeWolf, T. Holvoet. Decentralised Coordination Mechanisms as Design Patterns for Self-Organising Emergent Systems. In *Engineering Self-Organising Systems*. Volume 4335/2007, pp. 28–49. 2007.

[EGKM04] P. T. Eugster, R. Guerraoui, A.-M. Kermarrec, L. Massoulieacute;. Epidemic Information Dissemination in Distributed Systems. *Computer* 37(5):60–67, 2004.

[JPR09] S. Jha, M. Parashar, O. Rana. Self-Adaptive Architectures for Autonomic Computational Science. In *Proceedings of the Workshop on Self-Organizing Architectures*. 2009.

[MMTZ06] M. Mamei, R. Menezes, R. Tolksdorf, F. Zambonelli. Case studies for self-organization in computer science. *J. Syst. Archit.* 52(8):443–460, 2006.

[NT04] S. Nakrani, C. Tovey. On Honey Bees and Dynamic Server Allocation in Internet Hosting Centers. *Adaptive Behavior* 12(3-4):223–240, 2004.

[RS08] W. Renz, J. Sudeikat. Modeling Feedback within MAS: A Systemic Approach to Organizational Dynamics. In *Organised Adaptation in Multi-Agent Systems, First*

International Workshop, OAMAS 2008, Estoril Portugal, May 2008 Revised and Invited Papers. LNAI 5368, pp. 72–89. 2008.

[SBP+09] J. Sudeikat, L. Braubach, A. Pokahr, W. Renz, W. Lamersdorf. Systematically Engineering SelfOrganizing Systems: The SodekoVS Approach. *Electronic Communications of the EASST* 17, 2009. ISSN 1863-2122.

[SGK06] G. D. M. Serugendo, M. P. Gleizes, A. Karageorgos. Self-Organisation and Emergence in MAS: An Overview. In *Informatica*. Volume 30, pp. 45–54. 2006.

[SR08a] J. Sudeikat, W. Renz. *Applications of Complex Adaptive Systems*. Chapter Building Complex Adaptive Systems: On Engineering Self–Organizing Multi–Agent Systems, pp. 229–256. IGI Global, 2008.

[SR08b] J. Sudeikat, W. Renz. Toward Systemic MAS Development: Enforcing Decentralized Self-Organization by Composition and Refinement of Archetype Dynamics. In *Proceedings of Engineering Environment–Mediated Multiagent Systems*. LNCS, pp. 39–57. Springer, 2008.

[SR09a] J. Sudeikat, W. Renz. DeCoMAS: An Architecture for Supplementing MAS with Systemic Models of Decentralized Agent Coordination. In *Proc. of the 2009 IEEE/WIC/ACM Int. Conf. on Intelligent Agent Technology*. Pp. 104–107. IEEE Computer Society Press, 2009.

[SR09b] J. Sudeikat, W. Renz. MASDynamics: Toward Systemic Modeling of Decentralized Agent Coordination. In David and Geihs (eds.), *Kommunikation in Verteilten Systemen*. Informatik aktuell, pp. 79–90. 2009.

[SR09c] J. Sudeikat, W. Renz. On the Modeling, Refinement and Integration of Decentralized Agent Coordination – A Case Study on Dissemination Processes in Networks. In *Proceedings of the Workshop on Self-Organizing Architectures*. 2009.

[SR09d] J. Sudeikat, W. Renz. Programming Adaptivity by Complementing Agent Function with Agent Coordination: A Systemic Programming Model and Development Methodology Integration. *Communications of SIWN* 7:91–102, may 2009. ISSN 1757-4439.

[SR09e] J. Sudeikat, W. Renz. Shoaling Glassfishes: Enabling Decentralized Web Service Management. In *3rd International Conference in Sef-Adaptive and Self-Organizing Systems*. Pp. 291–292. IEEE, Los Alamitos, CA, USA, 2009. (short paper).

[ST97] Y. Shoham, M. Tennenholtz. On the emergence of social conventions: modeling, analysis, and simulations. *Artif. Intell.* 94(1-2):139–166, 1997.

[Ste00] J. D. Sterman. *Business Dynamics - Systems Thinking and Modeling for a Complex World*. McGraw–Hill, 2000.

[Zad63] L. A. Zadeh. On the definition of adaptivity. *Proceedings of the IEEE* 51(3):469 – 470, 1963.