

A Fast Convergence Coordination Protocol for Cooperative Open Real-Time Environments

Luís Nogueira, Luís Miguel Pinho

IPP Hurray Research Group
School of Engineering, Polytechnic Institute of Porto, Portugal
{luis, lpinho}@dei.isep.ipp.pt

Abstract. Although a lot of research has been conducted on the support of services' dynamic QoS reconfiguration under different constraints of resource availability, most has been focused on the runtime management of independent QoS attributes. This paper proposes support for adaptive cooperative coalitions of nodes, able to autonomously organise, regulate and optimise themselves without the user's intervention or any other central entity, even when services exhibit unrestricted distributed QoS inter-dependencies among their tasks.

Resumo Apesar da extensa investigação no suporte de uma reconfiguração dinâmica da QoS de serviços sob diversas restrições de disponibilidade de recursos, a maioria tem sido focada na gestão em tempo de execução de atributos de QoS independentes. Este artigo propõe suporte à adaptação de coligações cooperativas de nós, capazes de autonomamente se adaptarem, regularem e optimizarem sem a intervenção do utilizador ou outra qualquer entidade central, mesmo quando os serviços exibem inter-relações de dependência de QoS entre as suas tarefas.

1 Introduction

Adaptive real-time is a relatively new approach to open embedded systems design. It allows an online reaction to load variations, adapting the system to the specific constraints of devices and users, nature of executing tasks and dynamically changing environmental conditions. This ability is essential to efficiently manage the provided Quality-of-Service (QoS) in domains such as telecommunications, consumer electronics, industrial automation, and automotive systems.

Nevertheless, given the heterogeneity of services to be executed, users' quality preferences, underlying operating systems, networks, devices, and the dynamics of their resource usages, developing adaptive distributed cooperative systems presents a number of challenges. These include (i) defining a common understanding of how QoS should be specified; (ii) the provisioning of self-management actions that allow nodes to adapt to observed changes in the environment; (iii) resource management and scheduling strategies; and (iv) the design of an efficient coordination model that regulates individual autonomous adaptive actions. Here, the term *coordinated adaptation* refers to

the ability of a distributed adaptive system to invoke adaptive actions on multiple nodes in a coordinated manner so as to achieve a common goal.

While there has been a great deal of research in the former aspects of adaptation in embedded real-time systems, there has been little work that addresses the specific problem of coordinating individual autonomous adaptations in distributed environments. However, coordination is critical to maintain the correctness of a distributed application during adaptation [1,2] and also very challenging.

In particular, a QoS-aware adaptation in a service's distributed execution can require communication and synchronisation among nodes, used as a building block for a collective adaptation behaviour that emerges from local interactions among nodes. One challenge is controlling this exchange of information in order to achieve a convergence to a globally consistent solution without overflowing nodes with messages.

This paper shows how it is possible to achieve higher levels of self-adaptiveness in systems required to work in open real-time environments through a fast convergence decentralised coordination model. With the increasing size and complexity of open embedded systems the ability to build self-managed distributed systems using centralised coordination models is reaching its limits [3], as solutions they produce require too much global knowledge.

The one-step decentralised coordination model proposed in this paper defines a set of rules through which a coalition of nodes can be reconfigured, imposing restrictions on the way their members can self-adapt in response to changes in the environment. By exchanging feedback on the desired self-adaptive actions, nodes converge towards a global solution, even if that means not supplying their individually best solutions. As a result, each node, although autonomous, is influenced by, and can influence, the behaviour of other nodes in the system.

2 System model

Consider an open distributed system with several heterogeneous nodes, each with its specific set of resources R_i where independently developed services, some of them with real-time execution constraints, can appear while other are being executed, at any time, at any node. Due to these characteristics, resource availability is highly dynamic and unpredictable in advance.

It is assumed that each service S has a set of QoS parameters that can be changed in order to adapt the system's service provisioning to a dynamically changing environment. Each subset of parameters that relates to a single aspect of service quality is named as a *QoS dimension*. Each of these QoS dimensions has different resource requirements for each possible level of service. We make the reasonable assumption that services' execution modes associated with higher QoS levels require higher resource amounts.

There may exist QoS dependencies among two or more of the multiple QoS dimensions of a service S [4]. Given two QoS dimensions, Q_a and Q_b , a QoS dimension, Q_a , is said to be dependent on another dimension Q_b if a change along the dimension Q_b will increase the needed resource demand to achieve the quality level previously achieved along Q_a .

Users provide a single specification of their own range of QoS preferences Q_{Pref} for a complete service S , ranging from a desired QoS level $L_{desired}$ to the maximum tolerable service degradation, specified by a minimum acceptable QoS level $L_{minimum}$, without having to understand the individual components that make up the service. As a result, the user is able to express acceptable compromises in the desired QoS and assign utility values to QoS levels. Note that this assignment is decoupled from the process of establishing the supplied service QoS levels themselves and determining the resource requirements for each level.

For some of the system's nodes there may be a constraint on the type and size of services they can execute within the users' acceptable QoS levels. Therefore, the CooperatES framework [5,23] allows a distributed cooperative execution of resource intensive services in order to maximise the users' satisfaction with the obtained QoS. By redistributing the computational load across a set of nodes, a cooperative environment enables the execution of far more complex and resource-demanding services than those that otherwise would be able to be executed on a stand-alone basis.

Nodes dynamically group themselves into a new coalition, allocating resources to each new service and establishing an initial Service Level Agreement (SLA), a service description whose parameters are within the range of the user's desired QoS level $L_{desired}$ and the maximum tolerable service degradation, specified by a minimum acceptable QoS level $L_{minimum}$. The coalition is dynamically formed as the set of nodes which maximise the satisfaction of the QoS constraints associated with the new service and minimise the impact on the global QoS caused by the new service's arrival [5].

Nevertheless, short term dynamic environmental changes impose that the promised SLA for a service S_i can never be more than an expectation of a best-effort service quality during long term periods [6]. The system must be adaptive, that is, it must be able to adapt its timing expectations to the current conditions of the environment, possibly sacrificing the quality of other, non-time related, parameters. As such, once a SLA is admitted, it may be downgraded to a lower QoS level in order to accommodate new service requests with a higher utility or (re)upgraded when the needed resources become available.

As such, there will be a set of service's blocks being executed by a group of nodes, resulting from partitioning the resource intensive service S . Due to the existence of QoS dependencies, there are groups of service's tasks that must be executed in the same node. We refer to such groups of tasks as work units. A work unit $w_i = \tau_1, \tau_2, \dots, \tau_n$ is a set of one or more tasks τ_j of a service S that must be executed in the same node due to local QoS dependencies.

Let $W = \{w_1, \dots, w_n\}$ be the finite set of work units of a service S . We represent the set of inter-dependencies among work units $w_i \in W$ as a connected graph $\mathcal{G}_W = (\mathcal{V}_W, \mathcal{E}_W)$, on top of the service's distribution graph, where each vertex $v_i \in \mathcal{V}_W$ represents a work unit w_i and a directed edge $e_i \in \mathcal{E}_W$ from w_j to w_k indicates that w_k is functionally dependent on w_j . Within $\mathcal{G}_W = (\mathcal{V}_W, \mathcal{E}_W)$, we call *cut vertex* to a node $n_i \in \mathcal{V}_W$, if the removal of that node divides \mathcal{G}_W in two separated graphs.

We assume that each work unit $w_i \in W$ is being executed at its current QoS level Q_{val}^i at a node n_i , from a set of predefined QoS levels $\{Q_0, \dots, Q_n\}$, ranging from the user's desired QoS level $L_{desired}$ to the maximum tolerable service degradation,

specified by the minimum acceptable QoS level $L_{minimum}$. This relation is represented by a triple (n_i, w_i, Q_{val}^i) . Furthermore, for a given work unit $w_i \in W$, each node n_i knows the set $I_{w_i} = \{(n_j, w_j, Q_{val}^j), \dots, (n_k, w_k, Q_{val}^k)\}$, describing the quality of all the inputs related to work unit w_i coming from its adjacent nodes in $\mathcal{G}_{\mathcal{V}}$.

3 Autonomous cooperative systems

One of the key ideas of the CooperatES framework is that each coalition member should be able to take the initiative and to decide when and how to adapt to changes in the environment. However, whenever such autonomous adaptations have an impact on the outputted QoS of other coalition members the need of coordination arises: how to ensure that local, individual, adaptation actions of a node can produce a globally acceptable solution for the entire distributed service [7].

Yet, coordinating autonomous dependent adaptations in an open and dynamic system is challenging and may require complex communication and synchronisation strategies. According to [8] it borrows from as diverse areas as computer science, organisation theory, operations research, economics, linguistics, and psychology. This great diversity makes it very difficult to discuss every potential work that has some remote relation to coordination.

Researchers are increasingly investigating decentralised coordination models to establish and maintain system properties [10,11,12,13,14]. A system built using a decentralised coordination model is a self-organising system whose system-wide behaviour is established and maintained solely by the interactions of its nodes that execute using only a partial view of the system [9]. How these nodes were to interact in order to solve the problem (and not what they were actually doing) became the focus of decentralised coordination research. Without a central coordination entity, the collective adaptation behaviour must emerge from local interactions among nodes. This is typically accomplished through the exchange of multiple messages to ensure that all involved nodes make the same decision about whether and how to adapt.

Bridges et al. [15] propose a framework based on Cactus [16] which supports adaptations that span multiple components and multiple nodes in a distributed system. The architecture supports coordinated adaptations across layers using a fuzzy logic based controller module and coordination across hosts using a prototype protocol designed for communication oriented services. However, the authors do not specifically propose a method to obtain a globally coordinated solution.

Ren et al. [17] present a real-time reconfigurable coordination model (RT-RCC) that decomposes dynamic real-time information systems based on the principle of separation of concerns, namely, functional actors which are responsible for accomplishing tasks, and non-functional coordinators which are responsible for coordination among the functional actors. High level language abstractions and a framework for actors and coordinators are provided to facilitate programming with the RT-RCC model.

A similar approach is followed by Kwiat et al. [18] who propose a coordination model for improving software system attack-tolerance and survivability in open hostile environments. The coordination model is based on three active entities: actors, roles,

and coordinators. Actors abstract the system's functionalities, while roles and coordinators statically encapsulate coordination constraints and dynamically propagate those constraints among themselves and onto the actors. Both the coordination constraints and coordination activities are distributed among the coordinators and roles, shielding the system from single points of failure.

However, none of these works proposes support for coordinating inter-dependent autonomous QoS adaptations in cooperative systems, which is the focus of our work. Furthermore, with some decentralised coordination models it becomes difficult to predict the exact behaviour of the system taken as a whole because of the large number of possible non-deterministic ways in which the system can behave [19].

Whenever real-time decision making is in order, a timely answer to events suggests that after some finite and bounded time we would expect the global adaptation process to converge to a consistent solution. Furthermore, optimal decentralised control is known to be computationally intractable [12], although near-optimal systems can be developed for certain classes of applications [20,21,14].

Our goal is then to achieve a fast convergence to a global solution through a regulated decentralised coordination without overflowing nodes with messages. The next section details the proposed one-step decentralised coordination model based on an effective feedback mechanism to reduce the complexity of the needed interactions among nodes until a collective adaptation behaviour is determined.

3.1 A one-step decentralised coordination model

The core idea behind the proposed decentralised coordination model is to support distributed systems composed of autonomous individual nodes working without any central control but still producing the desired function as a whole.

We model a self-managed coalition as a group of nodes that respond to environmental inter-dependent changes according to a distributed coordination protocol defined by the following phases:

1. **Coordination request.** Whenever $Q_{val'}^i$, the needed downgrade or desired upgrade of the currently outputted QoS Q_{val}^i for a work unit $w_i \in S$, has an impact on the currently supplied QoS level of other work units $w_j \in S$ being executed at other coalition members, a coordination request is sent to the affected partners.
2. **Local optimisation.** Affected partners become aware of the new output values $Q_{val'}^i$ of w_i and recompute their local set of SLAs in order to formulate the corresponding feedback on the requested adaptation action. We assume that coalition partners are willing to collaborate in order to achieve a global coalition's consistency, even if this might reduce the utility of their local optimisations. However, a node only agrees with the requested adaptive action if and only if its new local set of SLAs is feasible.
3. **Adaptive action.** If the requested adaptive action is accepted by all the affected nodes in the coalition, the new local set of SLAs is imposed at each of those coalition members. Otherwise, the currently global QoS level of service S remains unchanged.

As such, requests for coordination may dynamically arrive at any time, at any node n_j . We consider the existence of feasible QoS regions [22]. A region of output quality $[q(o)_1, q(o)_2]$ is defined as the QoS level that can be provided by a work unit when provided with sufficient input quality and resources. Within a QoS region, it may be possible to keep the current output quality by compensating for a decrease in input quality by an increase in the amount of used resources or vice versa.

As such, if a node n_j , despite the change in current quality of some or all of its inputs, is able to continue to produce its current QoS level there is no need to further propagate the required coordination request along the dependency graph $\mathcal{G}_{\mathcal{W}}$. Thus, a *cut-vertex* is a key node in our approach.

Consider that a node n_k proposes an upgrade to Q'_{val} for a work unit $w_k \in S$. It may happen that some other nodes, precedent in the path until the next cut-vertex n_c , may be able to upgrade to Q'_{val} and others may not. Whenever the cut-vertex n_c receives the upgrade request and its new set of inputs, if it is unable to upgrade to Q'_{val} then all the effort of previous nodes to upgrade is unnecessary and the global update fails. Otherwise, the upgrade coordination request continues along the graph, until the end-user node n_u is reached.

In the case of a downgrade to Q'_{val} initiated by node n_k , it may happen that some other nodes in the path to the next cut-vertex n_c may be able to continue to output their current QoS level despite the downgraded input and others may not. Again, if the cut-vertex is unable to keep outputting its current QoS level then all the precedent nodes which are compensating their downgraded inputs with an increased resource usage can downgrade to Q'_{val} since their effort is useless.

In these and all other cases, the formulation of the corresponding positive or negative feedback, at the *local optimisation* phase, must depend on the feasibility of the requested coordination action as a function of the quality of the node's new inputs I_{w_i} for the locally executed work unit w_i . Such feasibility is determined by an anytime local QoS optimisation algorithm [23] which aims to minimise the impact of the requested changes on the currently provided QoS level of other services.

Note that the node's local resource usage optimisation associated with the new local set of SLAs can be lower after the coordination request. Recall that we make the assumption that, in cooperative environments, coalition partners are willing to collaborate in order to achieve a global coalition consistency, even if this coordination might reduce the global utility of their local QoS optimisations.

If all the nodes affected by the requested adaptation sent by node n_i agree with its new service solution, the *adaptive action* phase takes place. A "commit" message is sent by node n_i to its direct neighbours in the dependency graph, which then propagate the message to all the involved nodes in the global adaptation process.

Decentralised control is then a self-organising emergent property of the system. The proposed coordination model is based on these two basic modes of interaction: positive and negative feedback. Negative feedback loops occur when a change in one coalition member triggers an opposing response that counteracts that change at other dependent node. On the other hand, positive feedback loops promote global adaptations. The snowballing effect of positive feedback takes an initial change in one node and reinforces that change in the same direction at all the affected partners. By exchanging

feedback on the performed self-adaptations, nodes converge towards a global solution, overcoming the lack of a central coordination and global knowledge.

Note that only one negotiation round is required between any pair of dependent nodes. As such, the uncertain outcome of iterative decentralised control models whose effect may not be observable until some unknowable time in the future is not present in the proposed regulated coordination model.

Also note that the normal operation of nodes continues in parallel with the *change acknowledge* and *local optimisation* phases. Every time a node recomputes its set of local SLAs, promised resources are pre-reserved until the global negotiation's outcome is known (or a timeout expires). As such, the currently provided QoS levels only actually changes at the *adaptive action* phase, as a result of a successful global coordination.

Due to the environment's dynamism, more than one coalition member can start an adaptation process that spans multiple nodes at a given time. Such request can either be a downgrade or an upgrade of its current SLA for a work unit w_i of service S . Even with multiple simultaneous negotiations for the same service S , only one of those will result in a successful adaptation at several nodes since, due to local resource limitations, only the minimum globally requested SLA will be accepted by all the negotiation participants. In order to manage these simultaneous negotiations, every negotiation has a unique identifier, generated by the requesting node.

4 Evaluation

The simulator used in the conducted experiments was custom built in Erlang, a functional programming language designed to be run in a distributed environment populated with resource constrained devices. An application that captures, compresses and transmits frames of video to end users, which may use a diversity of end devices and have different sets of QoS preferences, was used to evaluate the behaviour of the proposed decentralised coordination model, with a special attention being devoted to introduce a high variability in the characteristics of the considered scenarios. The application is composed by a set of source units to collect the data, a compression unit to gather and compress the data sent from multiple sources, a transmission unit to transmit the data over the network, a decompression unit to convert the data into the user's specified format, and an user unit to display the data in the end device.

The number of simultaneous nodes in the system randomly varied from 10 to 100. Each node was running a prototype implementation of the CooperatES framework, with a fixed set of mappings between requested QoS levels and resource requirements. The code bases needed to execute each of the application's units was loaded a priori in all the nodes.

The characteristics of end devices and their more powerful neighbour nodes was randomly generated, creating a distributed heterogeneous environment. This non-equal partition of resources affected the ability of some nodes to singly execute some of the application's units and has driven nodes to a coalition formation for a cooperative service execution.

At randomly selected end devices, new service requests from 5 to 20 simultaneous users were randomly generated, dynamically generating different amounts of load and resource availability during the previously accepted services' execution.

Each service request was formulated as a set of random QoS levels, expressing the spectrum of the user's acceptable QoS levels in a qualitative way, ranging from a randomly generated desired QoS level to a randomly generated maximum tolerable service degradation. The relative decreasing order of importance imposed in dimensions, attributes and values was also randomly generated.

Based on each user's service request, coalitions of 4 to 20 nodes were formed using the anytime coalition formation and service proposal formulation algorithms proposed in [24,25]. Each node was connected at least to another node in the coalition. The maximum degree of each node, that is, the number of connections to a node was set to 3. After the coalition was formed, a randomly percentage of the connections among its members was selected as a QoS dependency among those work units.

To cope with such a dynamic environment, nodes were requested to adjust their local set of SLAs, either by lowering the currently provided QoS level of some services due to resource limitations or by (re)upgrading them when the needed resources become available [25].

The behaviour of the proposed decentralised one-step coordination model in highly dynamic scenarios was compared to a classic centralised optimal coordination model. With a centralised coordination model, all changes in the output quality of a work unit $w_{ij} \in S_i$ have to be communicated to a single entity with service-wide knowledge. Then, this central coordinator has to determine the impact of those changes in the overall coalition's QoS level and request the adaptation of the involved nodes. To evaluate the success or failure of such dependent adaptation, an adaptation request must be sequentially made along the dependency graph either until a common global service solution is found or one of the coalition member is unable to supply the new desired QoS values.

The conducted evaluation started by comparing the total number of messages that had to be exchanged among nodes when using both approaches to globally coordinate dependent autonomous self-adaptations. The average results of all simulation runs for different coalition sizes are plotted in Figure 1.

As expected, both coordination approaches require more messages to be exchanged among nodes as the complexity of the service's topology increases. Nevertheless, the proposed decentralised coordination model requires around 80% of the needed number of messages required by the centralised model until all the affected coalition members become aware of the coordination request's result.

Less messages should result in a faster convergence to a global common solution. To verify the veracity of such assumption a second study measured the needed average time from the moment a node issued a coordination request until the outcome of the global adaptation process was determined. The deadline used for the anytime local QoS adaptation at each node was set to one second. At the end of the algorithm's execution, the feasibility or unfeasibility of the received coordination request was determined by the node. The obtained results are plotted in Figure 2.

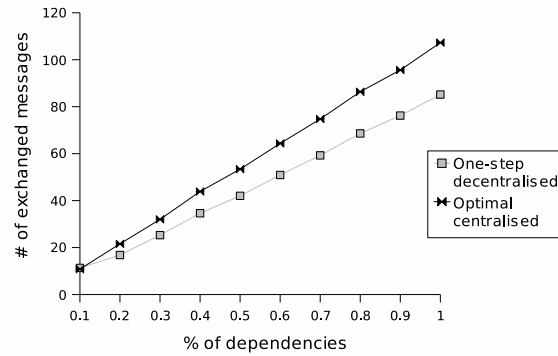


Fig. 1. Average number of exchanged messages

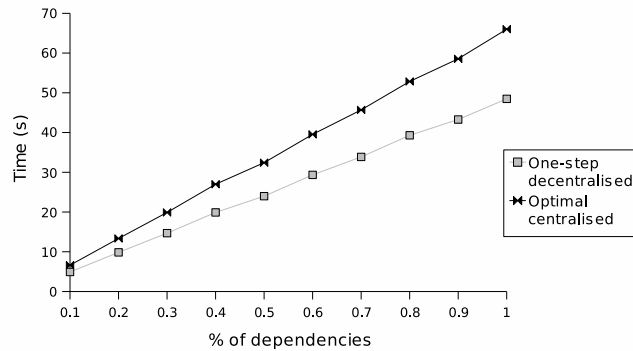


Fig. 2. Needed time until the global adaptation result is determined

Clearly, the proposed decentralised coordination model is faster to determine the overall coordination result in all the evaluated services' topologies, needing approximately 75% of the time spent by the centralised optimal model.

Even if the proposed one-step decentralised model requires less messages and is faster than a centralised optimal model to determine a global solution it is still important to evaluate impact of an one-step coordination model on the achieved service solution's quality. Recall that, when adopting the proposed one-step coordination algorithm, if some other dependent node in the coalition is unable to supply the new requested values no other alternative solution is tried and the global adaptation process fails. On the other hand, with the centralised optimal coordination model, a node is able to reply with a service counter-proposal whenever it is unable to coordinate with the currently requested values. As such, it is possible that after some iterations, the node's best possible service solution can be accepted by all the dependent coalition partners as part of

a global SLA. Note that such intermediate service solution would not be achieved with the proposed one-step coordination model.

The reward of each determined SLA after a successful global coordination process was evaluated by computing, for each service's QoS dimension, a weighted sum of the differences between the user's preferred quality values and the proposed values [5,24]. The results were plotted, in Figure 3, by averaging the results over several independent runs of the simulation, divided in two categories: (i) when the average amount of available resources per node is greater than the average amount of resources demanded by the services being executed; and (ii) when the average amount of resources per node is smaller than the average amount of demanded resources.

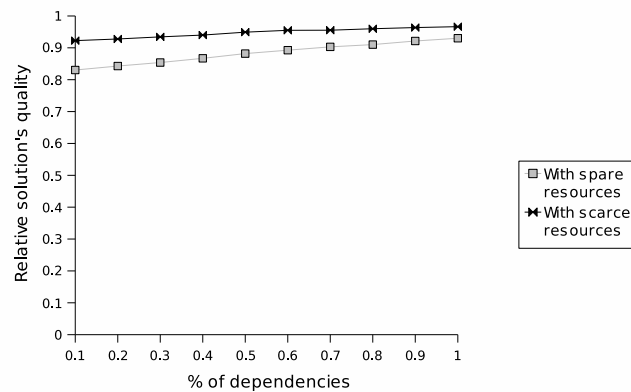


Fig. 3. Relative solution's utility as a function of available resources

As the coalition's topology complexity increases it is clearly noticeable, in both scenarios, that a near-optimal service solution's quality is achieved when using the one-step coordination model, despite its simpler approach and faster convergence to a common solution. The achieved results can be explained by the fact that as the coalition's topology complexity increases it also increases the probability of one of the involved nodes in the global adaptation process to be unable to use more than its current level of reserved resources for a work unit $w_i \in S$. As the achieved results clearly demonstrate, such probability is even greater when the resources are scarce.

5 Conclusion

In heterogeneous open distributed environments, computing devices can range from small, resource limited, mobile devices to stationary powerful servers. As such, for some of those nodes, there may be a constraint on the type and size of applications that can be executed with specific user's acceptable quality of service (QoS) and drive nodes to a coalition formation for a cooperative service execution.

This paper addressed the problem of coordinating autonomous dependent adaptations of resource-bounded nodes in dynamic open cooperative real-time environments. The proposed one-step decentralised coordination model imposes restrictions on the way members of dynamically created coalitions can self-adapt in response to changes in the environment.

The proposed coordination model is based on an effective feedback mechanism that reduces the complexity of the needed interactions among nodes. Feedback is formulated as a result of a QoS adaptation process that evaluates the feasibility of the new requested service solution. As the achieved results clearly demonstrate, the proposed coordination model has a reduced overhead and enables a faster convergence to a near-optimal global service solution.

Acknowledgements

This work was supported by FCT through the CISTER Research Unit - FCT UI 608 and the research project CooperatES - PTDC/EIA/71624/2006.

References

1. Allen, G., Dramlitsch, T., Foster, I., Karonis, N.T., Ripeanu, M., Seidel, E., Toonen, B.: Supporting efficient execution in heterogeneous distributed computing environments with cactus and globus. In: Proceedings of the 2001 ACM/IEEE conference on Supercomputing. (November 2001) 52–52
2. Ensink, B., Adve, V.: Coordinating adaptations in distributed systems. In: Proceedings of the 24th International Conference on Distributed Computing Systems, Tokyo, Japan (March 2004) 446–455
3. Montresor, A., Meling, H., Babaoglu, Ö.: Toward self-organizing, self-repairing and resilient distributed systems. In: Future Directions in Distributed Computing. (2003) 119–126
4. Rajkumar, R., Lee, C., Lehoczky, J., Siewiorek, D.: A resource allocation model for qos management. In: Proceedings of the 18th IEEE Real-Time Systems Symposium, IEEE Computer Society (1997) 298
5. Nogueira, L., Pinho, L.M.: Dynamic qos-aware coalition formation. In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium, Denver, Colorado (April 2005) 135
6. Burgess, M.: On the theory of system administration. *Science of Computer Programming* **49** (2003) 1
7. Gelernter, D., Carriero, N.: Coordination languages and their significance. *Communications of the ACM* **35**(2) (1992) 96–107
8. Malone, T.W., Crowston, K.: The interdisciplinary study of coordination. *ACM Computing Surveys* **26**(1) (1994) 87–119
9. Goldin, D., Keil, D.: Toward domain-independent formalization of indirect interaction. In: Proceedings of the 13th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Washington, DC, USA, IEEE Computer Society (2004) 393–394
10. Dorigo, M., Caro, G.D.: The ant colony optimization meta-heuristic. *New ideas in optimization* (1999) 11–32

11. Graupner, S., Andrzejak, A., Kotov, V., Trinks, H.: Adaptive control overlay for service management. In: First Workshop on the Design of Self-Managing Systems, San Francisco, USA (June 2003)
12. De Wolf, T., Holvoet, T.: Towards autonomic computing: agent-based modelling, dynamical systems analysis, and decentralised control. Proceedings of the IEEE International Conference on Industrial Informatics (August 2003) 470–479
13. Boutilier, C., Das, R., Kephart, J.O., Tesauro, G., Walsh, W.E.: Cooperative negotiation in autonomic systems using incremental utility elicitation. In: In Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence, Acapulco, Mexico (August 2003) 89–97
14. Dowling, J., Haridi, S.: in Reinforcement Learning: Theory and Applications. In: Decentralized Reinforcement Learning for the Online Optimization of Distributed Systems. I-Tech Education and Publishing, Vienna, Austria (2008) 142–167
15. Bridges, P., Chen, W.K., Hiltunen, M., Schlichting, R.: Supporting coordinated adaptation in networked systems. In: Proceedings of the Eighth Workshop on Hot Topics in Operating Systems, Oberbayern, Germany (May 2001) 162
16. The University of Arizona, C.S.D.: The cactus project. Available at <http://www.cs.arizona.edu/projects/cactus/>
17. Ren, S., Shen, L., Tsai, J.: Reconfigurable coordination model for dynamic autonomous real-time systems. In: Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, Taichung, Taiwan (June 2006) 60–67
18. Kwiat, K., Ren, S.: A coordination model for improving software system attack-tolerance and survivability in open hostile environments. In: Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, Taichung, Taiwan (June 2006) 394–402
19. Serugendo, G.D.M.: Handbook of Research on Nature Inspired Computing for Economy and Management. In: Autonomous Systems with Emergent Behaviour. Idea Group, Inc., Hershey-PA, USA (September 2006) 429–443
20. Jelasity, M., Montresor, A., Babaoglu, O.: A modular paradigm for building self-organizing peer-to-peer applications. In: In Engineering Self-Organising Systems, G. Di Marzo Serugendo, Springer (2004) 265–282
21. Dusparic, I., Cahill, V.: Research issues in multiple policy optimization using collaborative reinforcement learning. In: Proceedings of the 2007 International Workshop on Software Engineering for Adaptive and Self-Managing Systems, Washington, DC, USA, IEEE Computer Society (2007) 18
22. Shankar, M., de Miguel, M., Liu, J.W.S.: An end-to-end qos management architecture. In: Proceedings of the 5th IEEE Real-Time Technology and Applications Symposium, Washington, DC, USA, IEEE Computer Society (1999) 176–191
23. Nogueira, L., Pinho, L.M.: Time-bounded distributed qos-aware service configuration in heterogeneous cooperative environments. Journal of Parallel and Distributed Computing **69**(6) (June 2009) 491–507
24. Nogueira, L., Pinho, L.M.: Iterative refinement approach for qos-aware service configuration. IFIP From Model-Driven Design to Resource Management for Distributed Embedded Systems **225** (2006) 155–164
25. Nogueira, L., Pinho, L.M.: Dynamic qos adaptation of inter-dependent task sets in cooperative embedded systems. In: Proceedings of the 2nd ACM International Conference on Autonomic Computing and Communication Systems, Turin, Italy (September 2008) 97