Rule-based Programming of Heterogeneous Sensor and Actor Networks

Falko Dressler, Abdalkarim Awad, Dongyu Wang, and Reinhard German
Computer Networks and Communication Systems
Dept. of Computer Sciences, University of Erlangen, Germany

***Abstract*—We present a rule-based programming approach for Sensor and Actor Networks (SANETs). The developed Rule-based Sensor Network (RSN) system provides network-centric communication and data processing with a very small footprint of necessary instructions. Based on the basic mechanism, rules can be exchanged among participating nodes in order to optimize the overall system behavior. In particular, we show the feasibility of data pre-processing mechanisms such as data aggregation as well as data-centric communication based on rules that are distributed throughout the entire network. The support of heterogeneous nodes is inherently integrated in terms of different hardware, e.g. sensors and actuators, and different data processing algorithms. RSN has been implemented as a simulation model as well as for experiments using BTnode sensors with attached sensors and actuators.**

I. Introduction

Sensor and Actor Networks (SANETs) represent a specific class of sensor networks enriched with network-inherent actuation facilities [1]. This is provided by actuation devices, which are often referred to as actors [2]. In addition to requirements known from sensor networks, real-time operation in massively distributed systems and coordination capabilities on a higher abstraction layer are necessary. Over the last decade, the need for network-centric data preprocessing has been identified as a key challenge due to the observation that communication is much more expensive in terms of energy requirements compared to local processing [2]. This includes communication constraints for network-wide coordination or, at least, local decision taking strategies that lead to an emergent behavior on a higher abstraction layer. Self-organizing algorithms have been developed relying for example on clustering and aggregation techniques to improve scalability and network lifetime [3].

In this work, we present Rule-based Sensor Network (RSN), a system for network-centric operation in SANETs, which addresses some of these challenging requirements. This approach provides the building blocks for developing network-centric operation and control techniques needed in SANETs. Basically, our rule-based programming approach is the result from studies in the context of bio-inspired networking – precisely, in the context of cellular signaling cascades [4]. In the following, we outline the concepts and principles of RSN. We implemented this system for the simulation framework OMNeT++ and for the BTnode sensor systems that we extended to support multiple sensors and actuators as shown in Figure 1. The main advantages of RSN are the small footprint of rules and the simple local programming of nodes – making self-organization possible even in large scale sensor and actor networks. In particular, this system allows the quick and heterogeneous reprogramming of (individual) nodes. Therefore, network-centric optimization of the placement of computational intensive rules becomes possible.

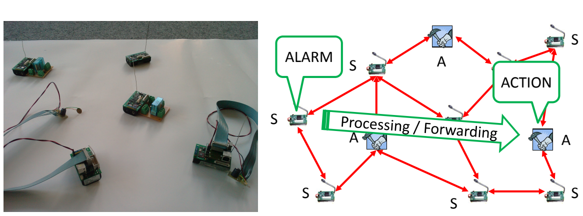

Fig. 1. Demo setup consisting of BTnodes with attached sensor and actuator hardware; RSN is used for network-centric data processing

II. Rule-based Sensor Network

Inspired by the capabilities of cellular signaling, i.e. the specific reaction to received information and the possibility to build signaling networks defining complex reaction pattern, we developed a rule-based programming system for application in SANETs. The primary design goals were a small footprint to enable the application of RSN on small embedded systems, easily transferable code, flexibility, and scalability for network-wide operations (basically, RSN provides the tools and concepts but the specific application needs to be designed properly as well). The rule-system greatly helps in designing distributed algorithms for use in self-organizing massively distributed systems. Additionally, RSN was inspired by early rule-based systems that have been developed in the context of active networking solutions. Examples are the mobile object system [5] and communicating rules [6].

The RSN architecture is depicted in Figure 2. It is based on *data-centric communication*, i.e. each message is self-describing to allow data-specific handling and processing without further knowledge, on *specific reaction on received data* using a rule-based programming scheme, and on *simple local behavior control* provided by state machines controlling the local behavior. Basically, all received messages are stored in a buffer (source set). Periodically, after a configurable timeout Δt, all these messages are processed by the instructions defined by the rules. Every rule has the form `if CONDITION then { ACTION(s) }` and selects a number of messages form the source set according to the condition and applies a (set of) actions to the selected messages.

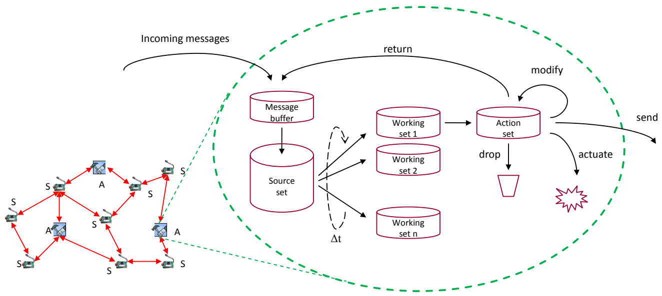

Fig. 2. Working behavior of a single RSN node; the rule-set is applied to received messages, which can be modified, forwarded, etc.

We distinguish the following action categories: *rule execution*, i.e. operations on the received messages; *node control*, i.e. control of the local node behavior (e.g., addition of sensors); and *Debugging*, i.e. actions needed for experiment control without influence on the node behavior. Table I lists the actions that are currently available in our implementation. Each action is triggered by a condition based on a set of message parameters and local variables that reflect the state of the node. Therefore, each message is specifically encoded to determine the message attributes. All implemented message and node attributes as well as the available preprocessing operations are listed in Table II.

Extensive tests and experiments have been conducted based on an simulation model for different data aggregation algorithms, probabilistic data communication, and distributed actuation control [4]. In addition, we completed another implementation for BTnode sensors. The presented demonstration focuses on the experimental setup using the BTnode systems.

III. Experimental Setup

We demonstrate the applicability of RSN in an experimental setup. First, we explore network-centric data aggregation to improve the efficiency of probabilistic communication (gossiping) [7]. Secondly, we investigated the capabilities of network-centric actuation control in SANETs in terms of scalability and real-time behavior. The demo setup is depicted in Figure 1.

Aggregation as a major building block for efficient and scalable data communication and preprocessing in SANETs because communication is much more expensive (in terms of energy consumption) compared to processing. The following RSN rule allows to aggregate multiple messages into a more compact message:

```
if :count > 1 then {
  !send($hopCount := @minimum of $hopCount,
        $value := @average of $value);
}
```

The actuators have an even simpler programming. For each received message, a `THRESHOLD` is evaluated and, if necessary, local actuation is initiated.

```
if $value > THRESHOLD then {
  !actuate($type:=rsnActuatorLightSource,
           $value:=@average of $value);
}
!drop;
```

Rule execution	
`!stop`	Early termination of the rule execution, the next iteration will start with the first available rule
`!drop`	Erases all messages in the current set, needs to be called after messages have been processed
`!dropDupl`	All duplicates are discarded according to a unique identifier in each message
`!return`	A new message is created and appended to the source message set
`!returnAll`	Copies of all messages in the current set are created and stored in the source message set
`!send`	A new message is created and submitted to all neighboring nodes
`!sendAll`	Copies of all messages in the current set are created and submitted to neighboring nodes
`!actuate`	A message is sent to locally connected actuators
Node control	
`!ctrlSensor`	A control message can enable/disable sensors and update the type field or the sampling frequency
`!ctrlActuator`	This command controls locally attached actuators (enable/disable, update type field)
Debugging	
`!recordAll`	Statistics are recorded: message source, node-specific message ID, hop count, time, and delay

TABLE I
Currently implemented actions

Message attributes	
`$name`	Descriptive name of the message
`$type`	Type of the message; describes the content
`$hopCount`	Number of traversed nodes
`$priority`	Importance factor of this message
`$value`	Message type specific value
Node attributes	
`:count`	Number of messages in the current working set
`:hostName`	ID of the current host
`:random`	Random value for probabilistic decisions
Preprocessing operations	
`@minimum`	Minimum of the selected value
`@maximum`	Maximum of the selected value
`@sum`	Sum of the selected value
`@average`	Average of the selected value

TABLE II
Selected attributes and operations

References

[1] I. F. Akyildiz and I. H. Kasimoglu, "Wireless Sensor and Actor Networks: Research Challenges," *Elsevier Ad Hoc Networks*, vol. 2, pp. 351–367, October 2004.

[2] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz, "A Distributed Coordination Framework for Wireless Sensor and Actor Networks," in *6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (ACM Mobihoc 2005)*, Urbana-Champaign, Il, USA, May 2005, pp. 99–110.

[3] F. Dressler, "A Study of Self-Organization Mechanisms in Ad Hoc and Sensor Networks," *Elsevier Computer Communications*, 2008, to appear.

[4] F. Dressler, I. Dietrich, R. German, and B. Krüger, "Efficient Operation in Sensor and Actor Networks Inspired by Cellular Signaling Cascades," in *1st ACM/ICST International Conference on Autonomic Computing and Communication Systems (Autonomics)*, Rome, Italy, October 2007.

[5] J. Vitek and C. Tschudin, *Mobile Object Systems - Towards the Programmable Internet*. Springer, 1997, vol. LNCS 1222.

[6] L. F. Mackert and I. B. Neumeier-Mackert, "Communicating Rule Systems," in *7th IFIP International Conference on Protocol Specification, Testing and Verification*, 1987, pp. 77–88.

[7] Z. J. Haas, J. Y. Halpern, and L. Li, "Gossip-Based Ad Hoc Routing," in *21st IEEE Conference on Computer Communications (IEEE INFOCOM 2002)*, June 2002, pp. 1707–1716.