Model-driven Development of Self-organizing Control Applications

H.-U. Heiß, G. Mühl, J. Richling, H. Parzyjegla, J. Schönherr
Berlin Institute of Technology

T. Weis, A. Wacker
University Duisburg Essen

Scenario

Today, modern mobile end-user devices are ubiquitous in our environment. Currently, these devices are fully self-contained, duplicating hardware components in fact unnecessarily. It is our vision that they will become part of actuator and sensor networks (AS-Nets) consisting of devices that are functionally smaller than today, communicate wirelessly and interact with each other. Together, they cooperatively realize ubiquitous services that are able to take full advantage of all actuators and sensors in the neighborhood, enabling new kinds of applications.

MODOC Project

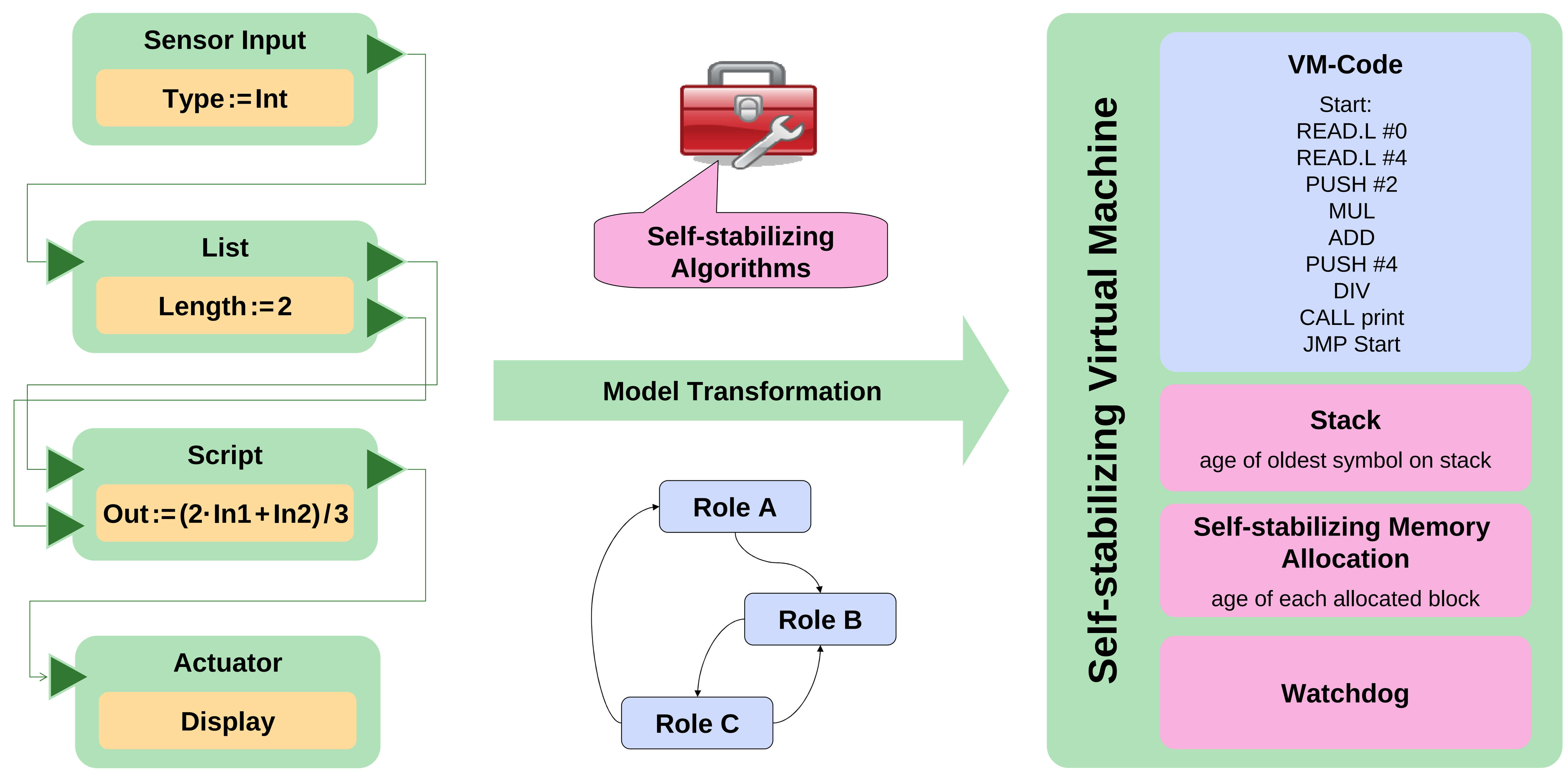
Developing applications for AS-Nets is challenging due to the complexity issues raised by the heterogeneity of devices, changing environment and the number of possible configurations and interaction patterns.

In order to tackle these challenges, the MODOC project follows a model-driven approach to the development of autonomous control applications supporting separation of concerns. To specify application logic, developers are provided with an easy-to-learn modeling language and a graphical development environment, while the implementation is generated by a model transformation taking care of non-functional properties. Generated applications work autonomously, organize and adapt themselves, and are guaranteed to be self-stabilizing.

Computational Model

We introduced a self-stabilizing Turing machine (STM) with the property that all programs executed on this STM are self-stabilizing. Thus, we were able to derive the complexity class that our approach can handle. Furthermore, multiple of these STMs can communicate with each other and form a distributed system that is self-stabilizing, too (assuming that the network stack and routers fail only temporarily). Unfortunately, using such STMs as an execution environment is very inefficient as the age of every tape symbol of a STM must be tracked in order to eventually erase old symbols.

For efficient execution we are developing a self-stabilizing virtual machine (VM) which can be almost as fast as normal Java or .NET VMs. The idea is to track the age of the stack and of allocated blocks on the heap. This way, tracking and aging each single byte (i. e., TM symbol) is avoided. The drawback is that it is not possible to guarantee self-stabilization for all possible VM programs – in contrast to STM programs. In order to still provide this guarantee, VM programs must follow certain constraints. Therefore, our model transformation ensures that all generated VM programs adhere to these constraints. Thus, we can still guarantee that every program generated by the MODOC tool chain is self-stabilizing.



Model Transformation: decomposition of developer supplied model into multiple roles and generation of per role VM-code.

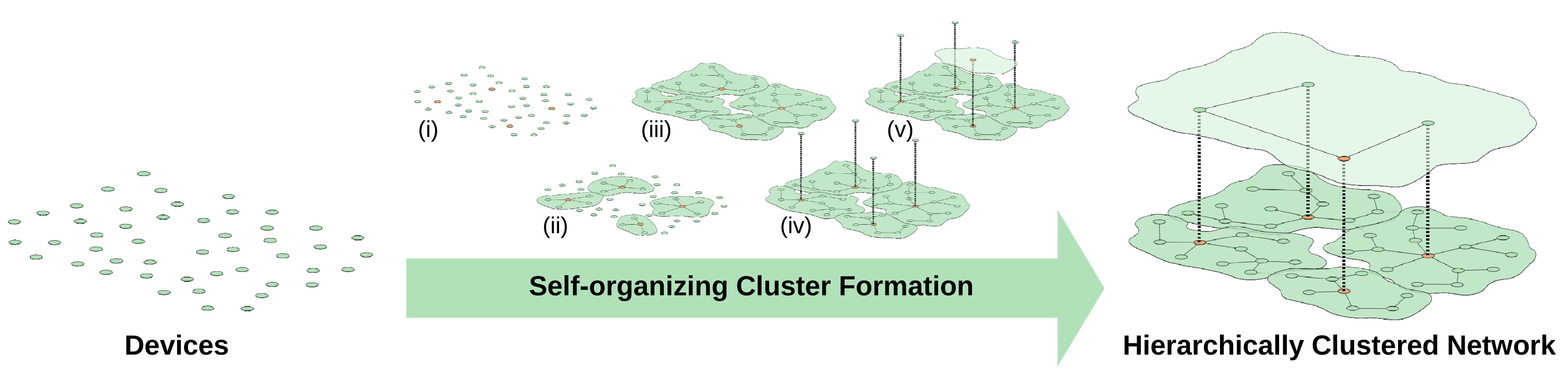
Non-functional Properties

Our model-driven development approach inherently supports separation of concerns regarding functional and non-functional properties. This way, developers can focus on application logic and rely on model transformation to fulfill desired non-functional requirements. When generating application code, MODOC automatically addresses properties such as self-organization and scalability, self-stabilization and fault-containment as well as self-optimization and adaptivity.

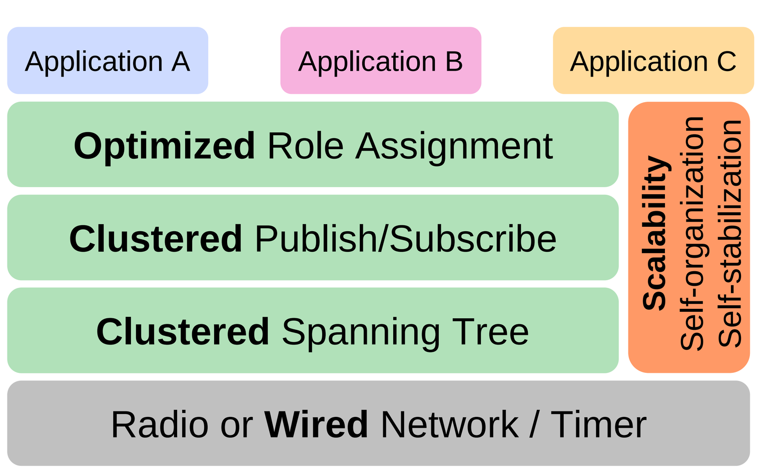
The employed content-based publish/subscribe algorithm establishes a flexible and extendible communication basis for devices, applications, and roles enabling an event-driven style of interaction. To support hundreds and even more physical nodes, a novel clustering scheme was developed which interconnects many small publish/subscribe networks to form hierarchies of independent clusters. The scheme is applicable for wireless ad-hoc networks, but can also take advantage of fixed infrastructures.

To cope with transient network faults such as temporary link or node failures as well as lost, delayed, or duplicated messages, all algorithms of our communication stack are proven to be self-stabilizing. Furthermore, additionally robustness is achieved by containing the effects of faults within affected clusters by exploiting cluster borders as fault barriers. However, a small fraction of faults cannot be contained and must be handled on higher hierarchy levels, where enough redundancy in form of alternative nodes and routing paths exists.

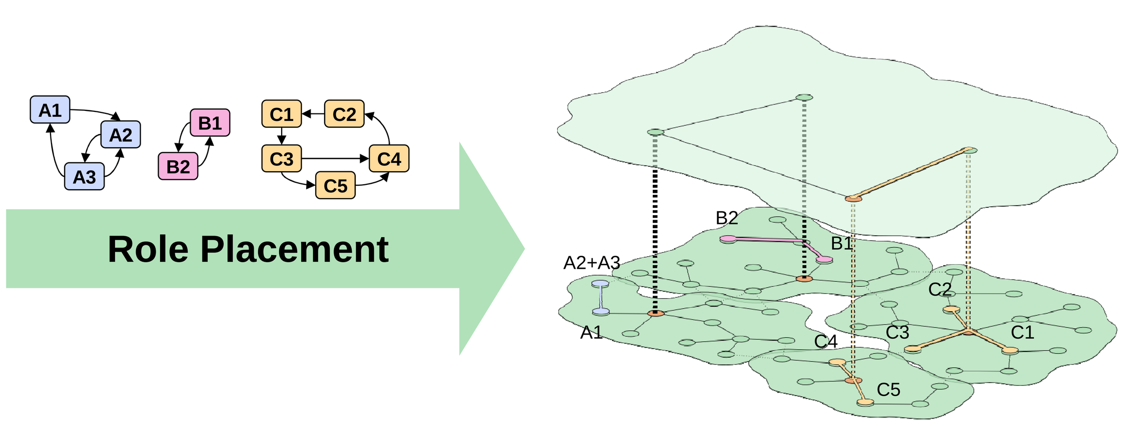
MODOC autonomously assigns application roles to devices capable of serving them. The role assignment algorithm has been improved so that application-specific quality of service (QoS) requirements are respected and related roles are placed close to each other in order to reduce the overall network traffic. As AS-Nets dynamically evolve over time (e. g., nodes are mobile, devices are added or removed) the current role assignment is monitored and adapted if necessary.



Clustering of devices to form one large clustered publish/subscribe network.



Revised algorithm stack.



Intelligent placement of roles in the clustered publish/subscribe network.

Summary and Outlook

The project MODOC is about developing highly distributed applications without the need to deal with issues like heterogeneity of devices, actual distribution, deployment and possibly hundreds of different types of faults. This is made possible by strict application of the model driven approach: Expert knowledge is encapsulated in the model transformation, allowing developers to focus on application logic. This way, important non-functional properties of the generated applications such as self-stabilization, fault-containment and self-optimization are guaranteed by construction.

For future work, we have three main goals: First, we want to widen the range of applications by including safety critical applications. Second, we will support large scenarios more efficiently by using an adaptive, fine grained selection of the routing strategies as well as support for composite events which will offload computation logic from applications to the network itself. Third, we want to further ease the work of the developer by introducing a new data-flow-based modeling language that provides a more intuitive way of writing self-x applications as well as by round-trip modeling that gives the developer feedback from the network based on reflective mechanisms to check whether applications behave as intended.

Web: http://www.kbs.tu-berlin.de/modoc E-mail: modoc@kbs.cs.tu-berlin.de