

Position Summary. Smart Messages: A System Architecture for Large Networks of Embedded Systems

Phillip Stanley-Marbell, Cristian Borcea, Kiran Nagaraja, Liviu Iftode
Department of Computer Science
Rutgers University
Piscataway, NJ 08854
{narteh@ece, borcea@cs, knagaraj@cs, iftode@cs}.rutgers.edu

We propose a system architecture and a computing model, based on *Smart Messages (SMs)*, for computation and communication in large networks of embedded systems. In this model, communication is realized by sending SMs in the network. These messages are comprised of code, which is executed at each hop in the path of the message, and data which the message carries in the network. The execution at each hop determines the next hop in the message's path – SMs are responsible for their own routing.

The nodes that support the execution of SMs are termed *Cooperative Nodes (CNs)*. The primary logical components of these nodes are a virtual machine that provides a hardware abstraction layer for executing SMs, and a *Tag Space* that provides a structured memory region consisting of tags persistent across the execution of SMs. Tags are used to store data that can be used for content-based addressing, routing, data sharing, or synchronization between SMs. An SM consists of code and data components. Upon admission at a CN, a task is created out of these components and executed on the virtual machine.

Figure 1 illustrates a network consisting of three types of nodes, represented with squares, circles and triangles. The nodes represented by squares are nodes of interest to an SM which is launched from the circular node in the lower left of Figure 1. The goal of the application implemented by this SM is to visit the five square nodes and to propagate a local data item of each node to the next one visited in order. The SM may use other nodes in the network, the circular and triangular nodes, as intermediates hops as it navigates through the network.

Admission at a CN is restricted based on tag availability, and resource demands of an SM. Tags can be used for synchronization between SMs executing on a CN: an SM may be de-scheduled on a read of a tag, pending a write on that tag by another SM, or the expiration of the tag in question. CNs employ a simple scheduling policy to accept and run multiple SMs. Once executing, an SM may create,

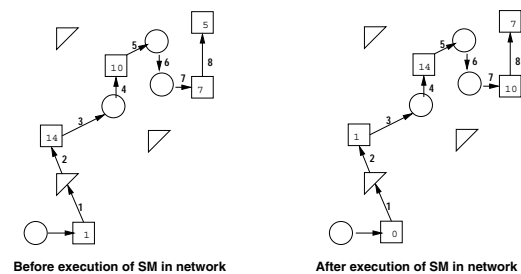


Figure 1. Smart Message Model Example

delete, read or write tags, either on the CN or in its own data component, subject to access restrictions and tag lifetimes. An SM may also create and send new SMs, building them out of its constituent code and data components, or it may migrate itself to another CN.

The Smart Message architecture is meant to provide a pervasive computing infrastructure for networks of embedded systems, such as sensor networks and computational fabrics – woven textiles with embedded computing elements. These networks will be inherently heterogeneous in their hardware architectures and inter-networking technologies, since each node will typically be specialized for performing a specific function, hence the need for a hardware abstraction layer such as a virtual machine, and will be volatile, due to node mobility and node failure. Applications utilizing these networks to perform a global task must be willing to accept partial results, or executions that satisfy a specific *Quality of Result (QoR)*.

Issues to be addressed in the architecture include: evaluating tradeoffs between flexibility and overhead of migration, defining a QoR for a partially successful execution, and CN security. A prototype implementation using Bluetooth technology for networking, and Sun Microsystem's KVM for the virtual machine is under development. More information can be found at:

<http://discolab.rutgers.edu/projects/sm.htm>.