

Report on "The Omni Macroprogramming Environment for Sensor Networks - Asad Awan et al"

Soham Chakraborty

Summary.

This paper describes a programming environment, called Omni, which is designed to facilitate the development of efficient macro level applications for sensor networks. The Omni system can be used to develop a variety of DDD applications. In this paper, wireless sensor networks designed to monitor the structural integrity of buildings and bridges has been used as a test case for the Omni system.

The paper points out several shortcomings in the existing systems that facilitate the development of applications for sensor networks. Current sensor network operating systems allow programmers to write efficient programs for individual nodes in the sensor network but does not directly support the development of applications for the network as a whole. Using high level domain specific languages abstracts the low level details, which may need to be fine-tuned to improve the system efficiency.

The Omni programming Environment provides an easy-to-understand programming interface for writing applications for the network as a whole and a set of operating system services that map the high level application functions to the network. The Omni OS separates the applications, OS services and the kernel. The OS services and the kernel control individual nodes in the network, whereas the applications define the functionalities of the entire network as a whole.

Applications are conceptually based on the box and arc model, where each processing element (PE) is represented by a box and the connection between two PEs is depicted by an arc. The system supports reconfiguration of this model at runtime, allowing both the PEs and the inter-PE links to be modified.

Omni PEs are viewed as single processing units which communicates with other PEs only through the input-output channels. The design

specification of a PE mentions only its functions and its input-output streams along with their datatypes. Applications can be developed from a collection of such PEs by creating a connection diagram depicting the links between the PEs. The connection diagram is checked for "typed-correctness" before running the application. This frees up the system from having to do the same at runtime. At runtime each PE behaves as a transaction which reads from input streams and writes into output streams. After successful execution of a task, a PE can ask the system to commit. A commit involves flushing the old data from the input queues and making the output queue available for consumption. The inter-PE links are implemented as single producer multiple consumer queues which are designed to lock free.

The Omni OS services are also designed based on the box and arc model. The channels between the services are determined by the arcs in the connection diagram. The OS services are primarily designed to control the hardware at each node.

The Omni system system is currently being developed and tested on the AVR and POSIX platforms.

Analysis.

The paper mentions that the connection diagram for the applications will be statically verified for typed-correctness and to see if the design meets the goals of the envisaged system. However the system is designed to reconfigurable at runtime, ie PEs can be replaced and links can be reordered. Such a system should have a mechanism for checking the correctness of the application on the fly, each time it is reconfigured, to prevent an inconsistencies from creeping in.