Autonomic Management of Component-Based Embedded Software

Authors

F. Romeo

F. Barbier

J.-M. Bruel

LIUPPA – Self-*

Université de Pau (FRANCE)

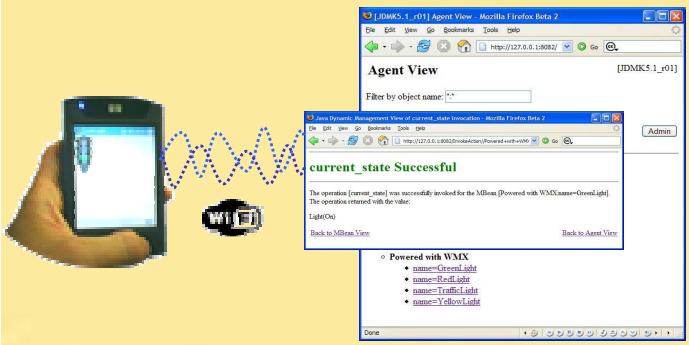
Abstract

Software components embedded in wireless devices are subject to behavior which cannot be fully and realistically predicted. This calls for **a runtime management infrastructure** that is able to observe and control the components' states and to make their behaviors explicit, tangible and understandable, in any case and at any time.

We propose a framework for remotely administrating the **functional behavior of software components** deployed on wireless nodes. This framework is based on components which are locally managed by internal managers on the wireless side. The controllable nature of components relies on **executable UML models that persist at runtime**. On the administration side, models are replicated and synchronized with the models that constitute the inner workings of the wireless components. This work has been validated and evaluated through the implementation of the **Wireless Management eXtensions** which enhances the spectrum width of the **PauWare** solution.

Wireless Management eXtensions (WMX)

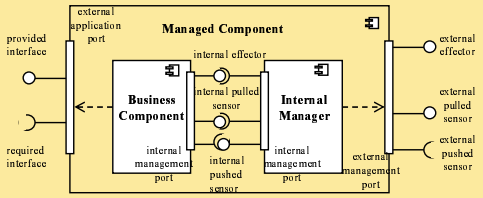
http://**www.univ-pau.fr/~fromeo/wmx**



wireless side **management system side**

On the wireless side:

- Each software component is controlled at runtime by an internal manager which drives the specified component's behavior
- Java ME implementation

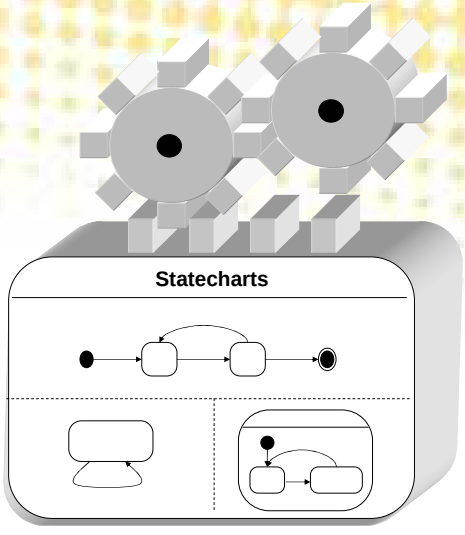


On the management system side:

- External managers remotely monitor and control the components through wireless communication (wifi, bluetooth, wma, …)
- Management operations are based on an abstract view of the components' behavior, which is provided by the internal managers
- Java SE implementation with JMX

Model Execution Engine

http://**www.pauware.com**



Model Driven Engineering philosophy:

- Components' behavior is designed with UML State Machine
- Direct execution and control of component's behavior by their internal managers which embed a PauWare Model Execution Engine
- The same models designed at development time are retrieved at runtime for management purpose

Statecharts for autonomic management policies:

- Both internal and external managers use PauWare State Machines (SM) to reify the components' behavior and their evolution is synchronized asynchronously
- Internal managers' SM evolve according to concrete phenomena of components' behavior, *i.e.* variable value changes and received events
- External managers' SM evolve according to an abstract view of the components' behavior which is provided by the internal managers (fired transitions)
- New transitions are added to external managers' SM in order to specify and organize management ECA (Event – Condition – Action) rules which are enforced on the components by their internal managers

Integrated Management 2007, Munich, Germany

http://**www.univ-pau.fr/~fromeo**

fabien.romeo@univ-pau.fr