CHARIOT: A DOMAIN SPECIFIC LANGUAGE FOR EXTENSIBLE CYBER-PHYSICAL SYSTEMS

Presented at DSM 2015, Pittsburgh, PA

Subhav Pradhan, Abhishek Dubey, Aniruddha Gokhale, and Martin Lehofer
subhav.m.pradhan@vanderbilt.edu

Institute for Software Integrated Systems
Department of Electrical Engineering and Computer Science
Vanderbilt University, Nashville, TN, USA

This work was supported in part by research grant from Siemens Corporate Technology
Extensible System Evolution

- **Mid 1990s**: Distributed System
- **Mid 2000s**: Grid Computing
- **Late 2000s**: Cloud Computing
- **Present**: Ubiquitous Computing

**Multi-domain CPS**

- Complex edge devices with purpose of connecting physical world with cyber world. E.g.: smart sensors
- Future of ubiquitous systems is cyber-physical
• What are CPS?
  – Integration of cyber (computation, communication) and physical components
  – Cyber components use sensors and actuators as interfaces to monitor and control physical environments
  – Traditionally, designed as static, single purpose systems

• What are Extensible CPS?
  – Collection of loosely connected cyber-physical (sub)-systems, from possibly different physical domains, that provide an open platform to host dynamic applications
    • E.g.: Smart city applications such as smart emergency response system, cluster of fractionated spacecraft providing cluster-as-a-service in space
  – Shared resources (Computation, communication, sensors, actuators)
  – Application lifecycle independent of underlying system
The Lifecycle of Extensible CPS

**Create**
- **Domain Specific Model**

**Generate**
- **Software Generator**
- **Generated Artifacts**

**Design-time**
- **Reconfiguration Engine**
- **Managed System**
- **Management Infrastructure**

**Runtime**
- **Monitor** (Failure detection + diagnosis)
- **Fault**
- **Manage**
Properties and Challenges

### Properties

<table>
<thead>
<tr>
<th>Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multi-tenant</td>
</tr>
<tr>
<td>Dynamic</td>
</tr>
<tr>
<td>Remotely deployed</td>
</tr>
<tr>
<td>Heterogeneous</td>
</tr>
<tr>
<td>Resilient (Desired property)</td>
</tr>
</tbody>
</table>

### Design-time Challenges

- **Challenge 1**: Modeling heterogeneous (i.e., middleware agnostic) applications to achieve maximum interoperability
- **Challenge 2**: Modeling resilience criteria/logic and patterns at design time

### Runtime Challenges
Extensible CPS

CHARIOT DSL Concepts

Role assignment
- App Developer
- SDK Developer
- System Architect

Systems

Compositions

Components

Nodes

Functionalities

Platform Interactions

Data Types

Node Categories

 Depends on

Uses (run-time dependency)
Middleware Agnostic Applications

- Follow generic component model that isn’t tightly coupled with any middleware solution
- Clean separation-of-concerns between communication and computation logic
- Communications modeled using common interaction patterns via different ports
  - Client/Server
  - Buffered, Sampling receiver
  - Sender
- Event-driven computation logic
  - Timers, Lifecycle events
  - External (message) events
    - Triggers based on state of common ports and not middleware specific callbacks
Extensible CPS

Runtime Mapping

Node A
Middleware
- Transport A
- Transport Proxy A
- Component A
  - Port A
  - Computation Logic

Node B
Middleware
- Transport B
- Transport Proxy B
- Component B
  - Port B
  - Computation Logic

App A
Component A
Transport A
Transport Proxy A
Port A
Computation Logic

App B
Component B
Transport B
Transport Proxy B
Port B
Computation Logic
• A system’s goal requires one or more objectives to be satisfied

• An objective requires existence of one or more functionalities, where functionalities are provided by components
  – A component provides exactly one functionality

• Dynamic representation of a system, which doesn’t require systems to be described as a collection of concrete components
  – Doesn’t matter which component provides a certain functionality as long as it is provided
  – Live system comprising one or more instances of different components

• Redundancy patterns can be applied to functionalities for resilience
  – Voter pattern, Consensus pattern, Simple cluster pattern
Summary

• Extensible CPS are the future of ubiquitous computing
  – Complex devices capable of interacting with their surroundings
  – E.g.: Smart city applications such as smart emergency response, smart parking, smart transportation

• Properties of extensible systems and CPS results in interesting challenges both at design-time and run-time

• Addressing heterogeneity at design-time by using generic component model that supports cleanly separated computation logic and communication logic
  – Middleware agnostic solution

• Modeling goal-based systems at design-time by encoding dynamic representation of system descriptions
  – Using well known redundancy patterns with goal-based system design