Hands-On Modelling for Digital Twins

By Likhith Kanigolla,
Advisor: Dr. Karthik Vaidhyanathan
Software Engineering Research Centre, IIITH

What are Digital Twins?

Digital twins are virtual representations of physical objects, processes, or systems that serve as the real-time digital counterparts of physical entities. In the IoT context, digital twins enable:

- Real-time monitoring and visualization
- Predictive analytics and simulation
- Enhanced decision-making capabilities
- Optimized performance and maintenance

Problem

Existing infrastructure often lacks readily usable models and manual simulation setup is complex and slow.

What we need

Efficient simulation of complex, interconnected IoT systems (like in Smart Cities) for analysis & optimization.

Model-Driven Simulation Generation

High-level models defined in a Domain-Specific Modelling Language (DSML).

Automated Model-to-Text (M2T) transformation generates simulation code.

Executable Python code using the Python PDEVS simulation framework.

Model-Driven Simulation Generation

High-level models defined in a Domain-Specific Modelling Language (DSML).

Automated Model-to-Text (M2T) transformation generates simulation code.

Executable Python code using the Python PDEVS simulation framework.

Modeling and Simulating IoT Infrastructures

Philipp Zech[1][a], Karthik Vaidhyanathan[2][b], Likhith Kanigolla[2][c], Luca Rahm[1], and Ruth Breu[1][d]

[1]*Department of Computer Science, University of Innsbruck, Technikerstrasse 21a, Innsbruck, Austria*

[2]*Software Engineering Research Center, IIIT Hyderabad, Gachibowli, India*

{philipp.zech, luca.rahm, ruth.breu}@uibk.ac.at, karthik.vaidhyanathan@iiit.ac.in,likhith.kanigolla@research.iiit.ac.in

Discrete Event System Specification (DEVS)

Atomic DEVS

Behaviour of a single component.

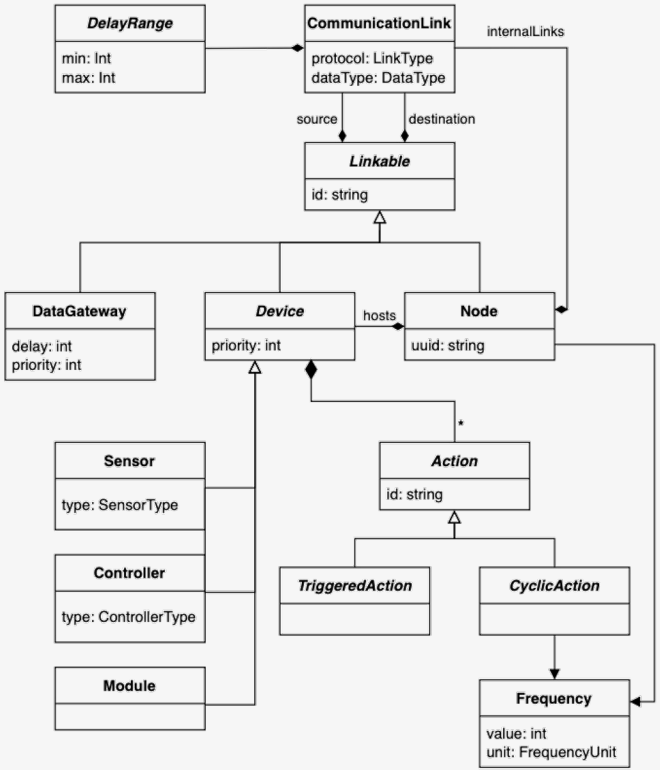
$$aDEVS = \langle S, ta, \delta_{int}, X, \delta_{ext}, Y, \lambda \rangle$$

Coupled DEVS

How multiple components connect & interact.

$$cDEVS = \langle X_{self}, Y_{self}, D, \{M_i\}, \{I_i\}, \{Z_{i,j}\}, select \rangle$$

DSML for Modeling Connected IoT Infrastructures



Algorithm

Input: Node `node` with links, frequency, and priority
Output: Atomic DEVS model

Initialize `nodeLinks` (list) and `distinctLinkTypes` (set) from `node.links`;
Compute `postFrequency` from `node.freqValue` and `node.freqUnit`;
Determine `knownController` based on `node.controller` type;

Generate Class `NodeState` definition:
    Initialize `data_aggregated` (empty) and set `next_internal_time` with randomness around `postFrequency`;

Generate Class `Node (AtomicDEVS)` **definition**:
    Initialize (`name, pinout`):
        Set `state` (NodeState), `timeLast` (0.0), `pins` (pinout);
        Create input ports for `distinctLinkTypes`;
        Define output port and set node priority;
    Generate Function `timeAdvance()`:
        Return time until next event if data exists, else `INFINITY`;
    Generate Function `extTransition(inputs)`:
        Update state with received inputs if matching links exist;
        Update `timeLast` to `next_internal_time`;
    Generate Function `intTransition()`:
        Increment `next_internal_time` and update `timeLast`;
    Generate Function `outputFunc()`:
        If data exists, package with timestamp, send via output, and clear state;

What About the Hardware?

CAPS Modeling Framework

A Model-Driven Engineering (MDE) framework designed for engineering Situational Aware Cyber-Physical Systems (SiA-CPS)

Three distinct views: Software Architecture, Hardware Configuration, and Physical Space

Built upon the Eclipse Modeling Framework (EMF)

CupCarbon

- A Smart City & IoT network simulator
- Models wireless sensor networks (WSN), mobile IoT nodes
- Allows realistic visualization on maps (Google Maps/OpenStreetMap)
- Open-source, used for teaching, research, and prototyping

Time for Hands-on!

Any Questions?