Brandenburg
University of Technology
Cottbus

RNKS

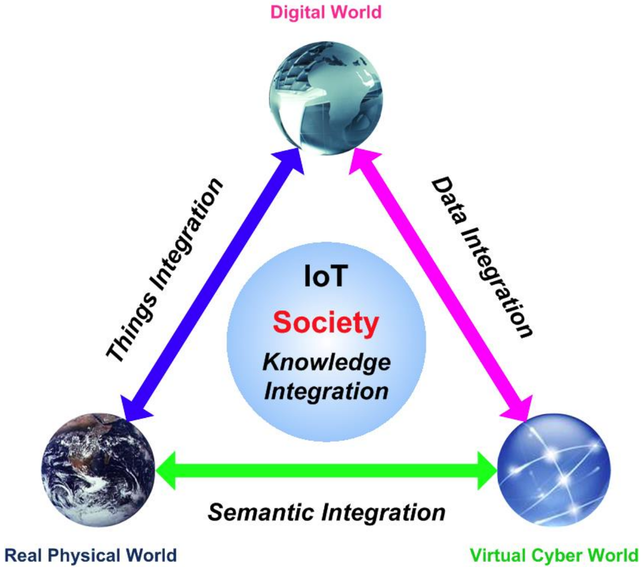
6th OMNeT++ Workshop 2013

A 6LoWPAN Simulation Model for OMNeT++

Michael Kirsche
Computer Networks Communication Systems Group
Brandenburg University of Technology, Germany

6LoWPAN and the IoT

- IoT → interconnecting the physical world with the digital / virtual world
  - All devices end-to-end connected by IPv6
- IPv6 over Low Power Wireless Personal Area Networks (6LoWPAN)
  - *“Glue between embedded and desktop world”*
  - Missing simulation support



© http://www.internet-of-things-research.eu

“Traditional“ TCP/IP Protocol Stack				6LoWPAN Protocol Stack		
HTTP	RTP		*Application Layer*	Applications		
TCP	UDP	ICMP	*Transport Layer*	TCP	UDP	ICMP
IP			*Network Layer*	IPv6	6LoWPAN	
Ethernet MAC and PHY			*Data Link and Physical Layer*	IEEE 802.15.4 MAC and PHY		

Preliminary Considerations

- What is available / used?
  - Real world: IPv6 and 6LoWPAN-capable OS / stacks
  - OMNeT++ world: IPv6 support (via INET)
  - Other simulators: Cooja (simulate Contiki entities)
    NS-3 (has code??)
- What is required?
  - Full functionality (to analyse interconnectivity & other advanced features)
  - Compatibility and validation
- What to do?
  1. Model from scratch
     - Probably a bad choice due to abstraction and reduced functionality
  2. Integrate an existing implementation
     - First step of “integrating” (simulating) an embedded OS (Contiki) in OMNeT++

Using Contiki Code with OMNeT++

- For this work → integrate 6LoWPAN implementation in OMNeT++
- Several approaches (refer to publication)

  ↳ *Create a new platform for Contiki*

- How to integrate and use 6LoWPAN in OMNeT++:
  (simplified summary)
  1. Define new platform for Contiki `(omnetpp)`
  2. Compile Contiki as a static library
  3. Create a “6loWPAN wrapper” for OMNeT++ / INET
  4. Redirect Contiki’s 6LoWPAN input / output functions at linking time (linker option `--wrap`) to appropriate OMNeT++ / INET functions
  5. Integrate the 6LoWPAN wrapper in IPv6-capable INET host
  6. Simulate with an IEEE 802.15.4-capable network
     - IEEE 802.15.4 provided by INETMANET or MiXiM / mixnet

The Modules' Modus Operandi (1)

1. Initialize the 6LoWPAN module and according buffers

2. Set function pointers from Contiki's input / output interfaces to according OMNeT++ functions

3. Initialize memory management to simulate multiple Contiki instances
   - Instances are identified via OMNeT++ gate ID
   - Check interfaces for IEEE 802.15.4 capabilities

4. Transform incoming higher layer packets to Contiki's format
   - Identify memory slot / instance and switch to last context
   - Data from OMNeT++ message is written directly in `UIP_BUFFER`

The Modules’ Modus Operandi (2)

5. Call `tcp_output` function when transformation is completed
   - 6LoWPAN handles compression and fragmentation
   - Link-local address of next hop is provided by INET IPv6-ND

6. Finished packets are sent (from Contiki’s code, function call is redirected to OMNeT++ via link-time wrapping)
   - Generated bytestream is captured and written in a queue

7. Incoming packets from lower layers are treated in a similar way
   - Input function of 6LoWPAN code is called instead
   - Memory management used to switch contexts and handle packet fragmentation / reassembly
   - Only reassembled packets are sent to OMNeT++, fragments are treated by Contiki’s 6LoWPAN code

In conclusion…

- We provide a 6LoWPAN simulation model that supports:
  - TCP, UDP, ICMP
  - Integration into INET
  - Use with INETMANET / MiXiM
  - IPv6, HC1, HC06 compression
  - Fragmentation
  - Limited neighbour discovery (without context distribution)
  - *… a first step for an integration of Contiki into OMNeT++ …*

- Current problems:
  - No extension headers
  - Code still pretty buggy
  - Integration via static library

- Future plans:
  - Integration via dynamic library
  - RFC 6775 extensions
  - Add more protocols from Contiki
  - Simulate more IoT scenarios with OMNeT++

See you at my poster for additional information