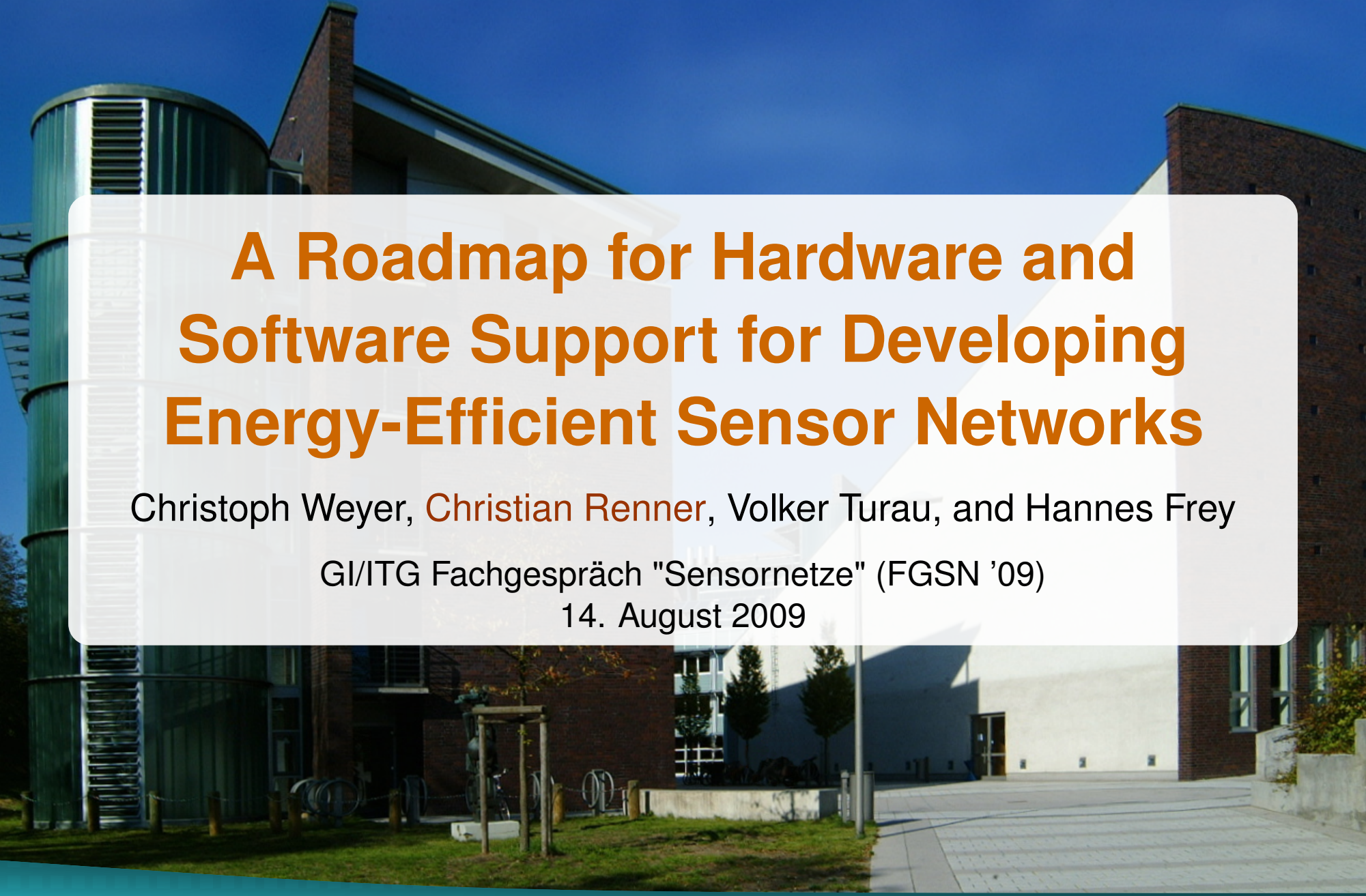


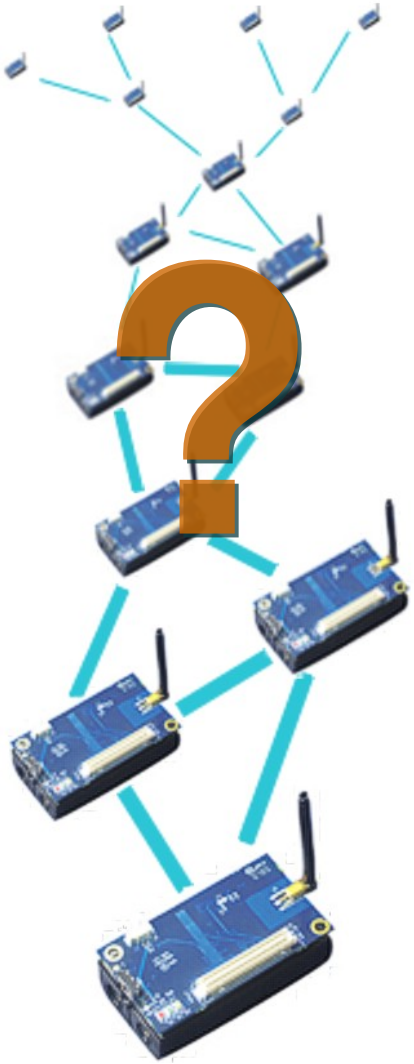
A Roadmap for Hardware and Software Support for Developing Energy-Efficient Sensor Networks

Christoph Weyer, Christian Renner, Volker Turau, and Hannes Frey

GI/ITG Fachgespräch "Sensornetze" (FGSN '09)
14. August 2009

Institute of Telematics
Hamburg University of Technology
TUHH

Development Support for WSNs



Problem:

- Debugging new implementations
  - → Logging of internal information
- Comparing existing implementations
  - → Extracting comparable metrics

Looking for: Development support tools

Solution: Automated approach

- Instrumentation & Evaluation
- Real Hardware & Testbed

Automated Development Support

Goals:

- Extract information without any manual interventions
- Gather essential information about network state
  - State of nodes
  - Message flows within the network
- Small memory footprint and low runtime overhead

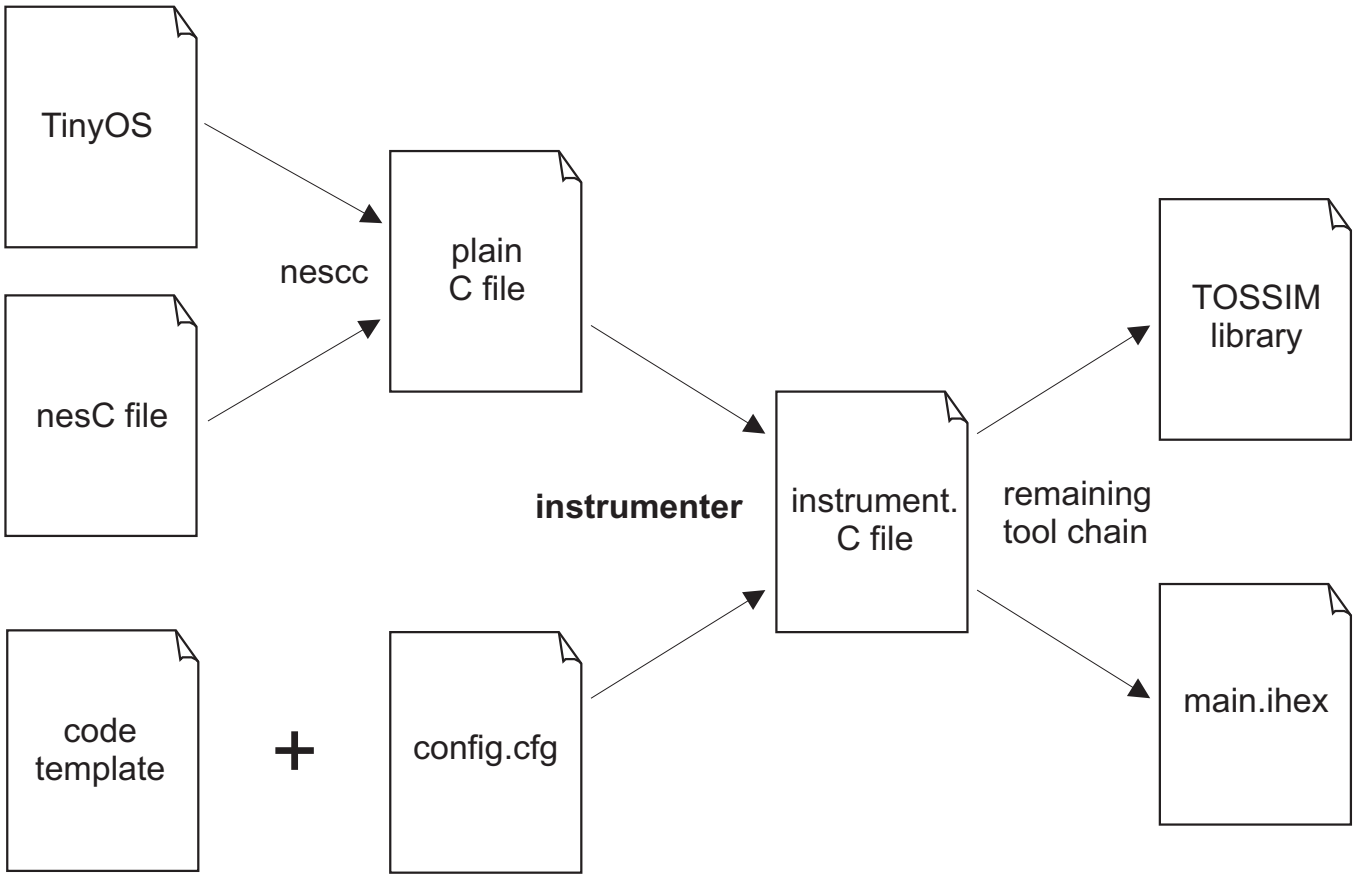
Preconditions:

- Coding conventions are required
- TinyOS provides some kind of conventions

Information Gathered by TinyAID

- Call-chain logging
  - Occurrence of events reflects change of node state
    e.g., TinyOS events `Timer.fired`
  - Currently executed component
    Part of the C function name that is called
  - Monitoring what components are turned on/off
    Tracing specific events, e.g., `Radio.startDone`
- Message logging
  - Sending and receiving a packet
  - Tracking of packets over multiple hops
- Adding timestamps to logged data

Instrumentation



Configuration of Call-Chain Logging

```
-d /opt/tinyos-2.x/.*   # exclude everything in /opt/tinyos-2.x
+f Test.nc              # include everything in file Test.nc
+h fired                # include all fired event handler
+h booted               # include all booted event handler
```

- Instrumentation code is inserted based on configuration
- Configuration includes (+) or exclude (-)
  - Directories (d)
  - Files (f)
  - Commands or Events (h)
- First match decides action
- If no match is found no instrumentation is inserted

Message Logging

- Multi-hop packet tracing
- Adding additional fields to each packet
  - Originating node
  - Unique sequence number for each node
- Information is inserted by calling `Packet.clear()`
- Trace: creating, sending, and receiving of packets

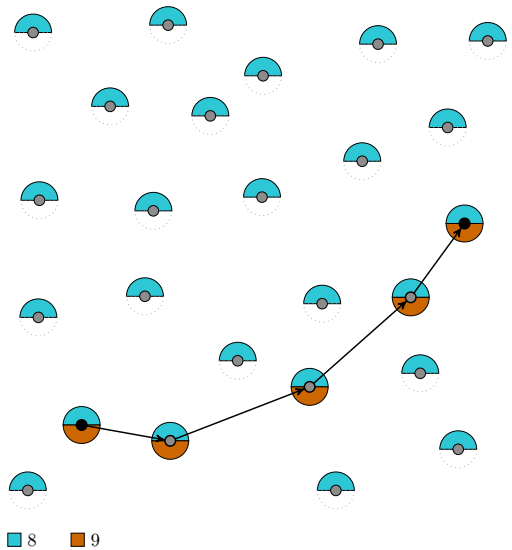
node	time [ms]	action	type	src	dest	origin	seqno
3	3520	c				3	42
3	3521	s	17	3	12	3	42
5	3524	c				5	14
5	3525	s	34	5	65535	5	14
12	3535	r	17	3	12	3	42
3	3520	c				3	43
...	...	...	...	...	...	...	...

Message Logging

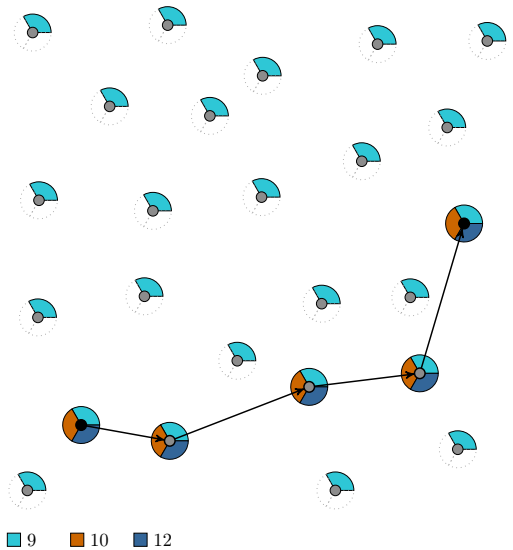
- Multi-hop packet tracing
- Adding additional fields to each packet
  - Originating node
  - Unique sequence number for each node
- Information is inserted by calling `Packet.clear()`
- Trace: creating, sending, and receiving of packets

node	time [ms]	action	type	src	dest	origin	seqno
3	3520	c				3	42
3	3521	s	17	3	12	3	42
5	3524	c				5	14
5	3525	s	34	5	65535	5	14
12	3535	r	17	3	12	3	42
3	3520	c				3	43
...	...	...	...	...	...	...	...

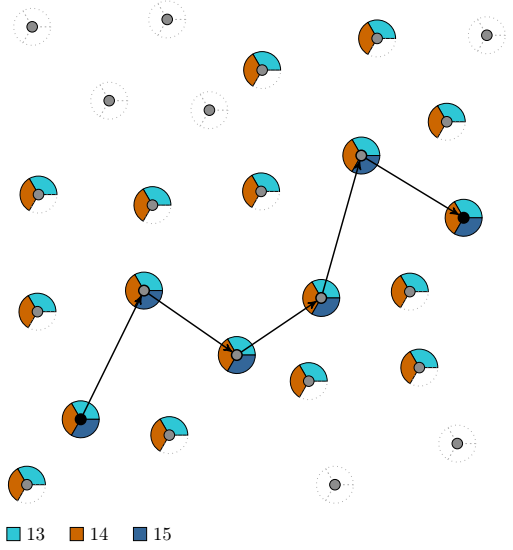
Concept Evaluation: Packet Types



TYMO



DSR



Greedy

- Visualization of network activities based on packet types
- Identification of protocol execution

The Need for a Hardware Adapter

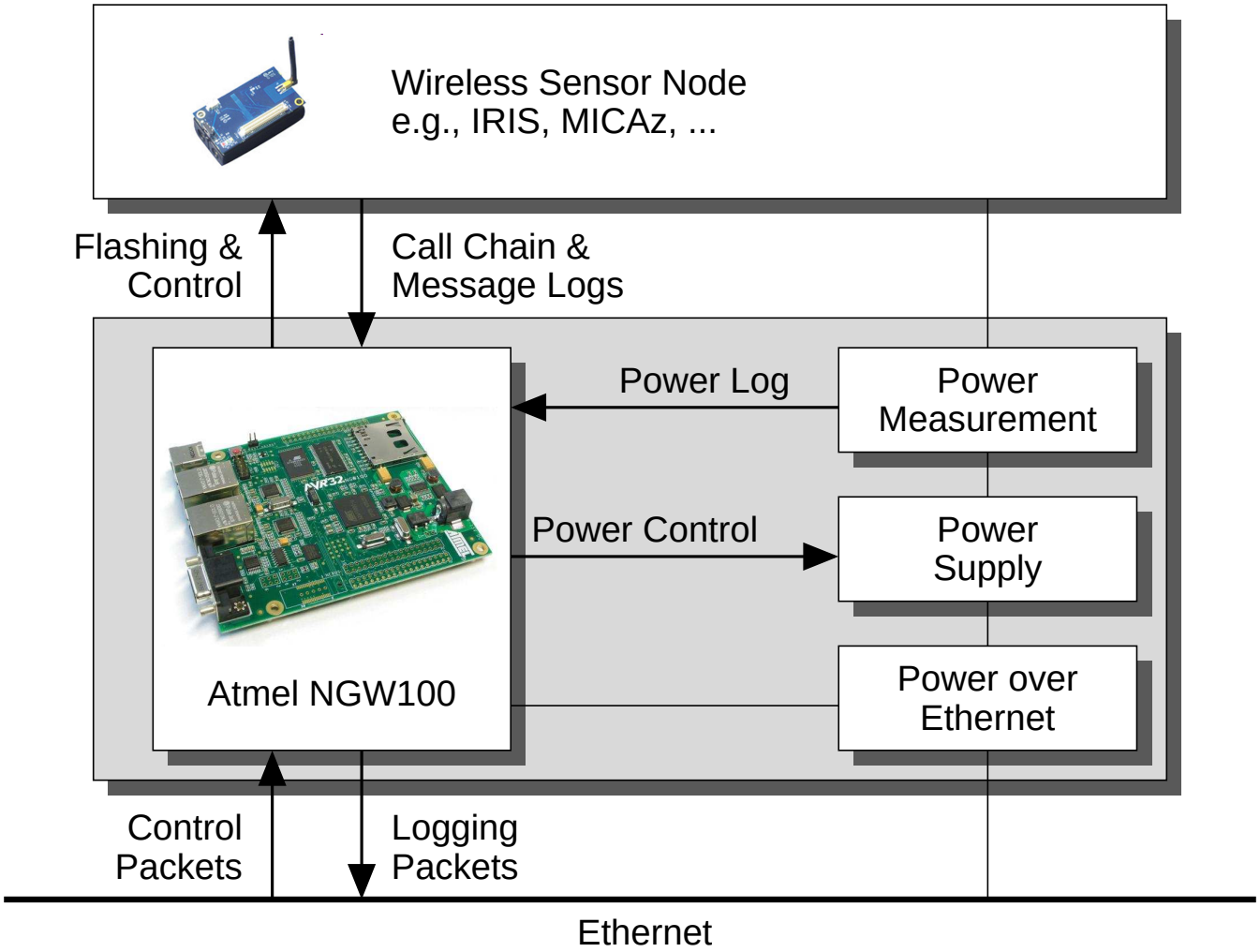
Goals:

- Run applications on real hardware
- Gather information provided by TinyAID
  $\longrightarrow$ node *and* network state
- Low overhead
- Energy supply control

Solution:

- Design and build a hardware adapter
- Combine many adapters to form a testbed

Hardware Adapter Building Blocks



Measuring Current and Dynamic Power Supply

Requirements:

- Fine-grained, periodic sampling of current drawn by sensor node
- Precise measuring over several orders of magnitudes
  - Sleeping node: a few μA (10^{-6} A)
  - Radio active: some mA (10^{-3} A)
  - High load plus active sensors: up to 100 mA (10^{-1} A)
- Dynamic power supply for energy-aware applications

Retrieving TinyAID-Generated Log Data

- Log as less as possible to avoid affecting the node's function
- Use I/O pins of μC
  - 8 bits for call-chain logging
    - 1 bit to indicate entering/leaving of functions
    - 7 bits to identify function
    - all zeros indicating that no data is available
  - Message logging requires one byte for each origin, sequence number, type, sender, (intended) receiver
- Additional data retrieved by hardware adapter e.g., time, local node, or energy consumption

Communicating Data

- Central management unit
- Exchange of data via Ethernet
  e.g., firmware images, configurations, log data
- I/O pin as switch to turn on/off logging

Hardware Prototype – The Trouble Child

Output Voltage Regulator:

- Considered two different, suitable voltage regulators
- Tested ability to reproduce sine and square wave with sampling rates between 50 Hz and 200 kHz
- Only one of the regulators is generally suitable, but it cannot drive a node under full load

Measuring Current:

- Considered various instrumentation amplifiers feeding a logarithmic amplifier
- No precise measuring possible over 5 orders of magnitude at 1 kHz

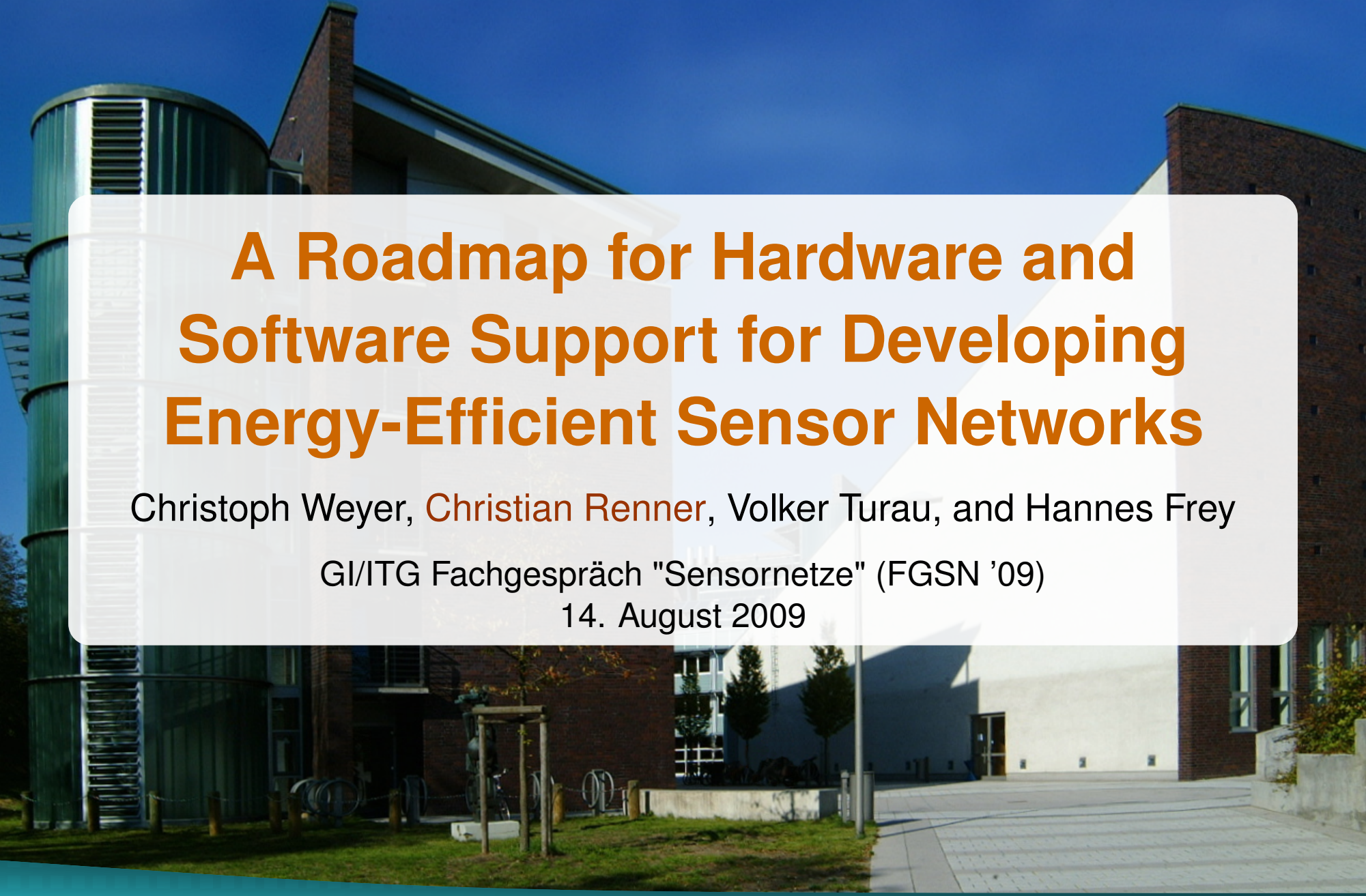
Conclusion

Where we are:

- TinyAID: valuable support for TinyOS programming
- Simple practicability on legacy source code
- Instant information about internal sequences
- Automated packet tracing
- Hardware support requires additional effort

And where we'll go:

- Design and build better suited hardware adapter
- Apply TinyAID to real hardware



A Roadmap for Hardware and Software Support for Developing Energy-Efficient Sensor Networks

Christoph Weyer, Christian Renner, Volker Turau, and Hannes Frey

GI/ITG Fachgespräch "Sensornetze" (FGSN '09)
14. August 2009

Institute of Telematics
Hamburg University of Technology
TUHH

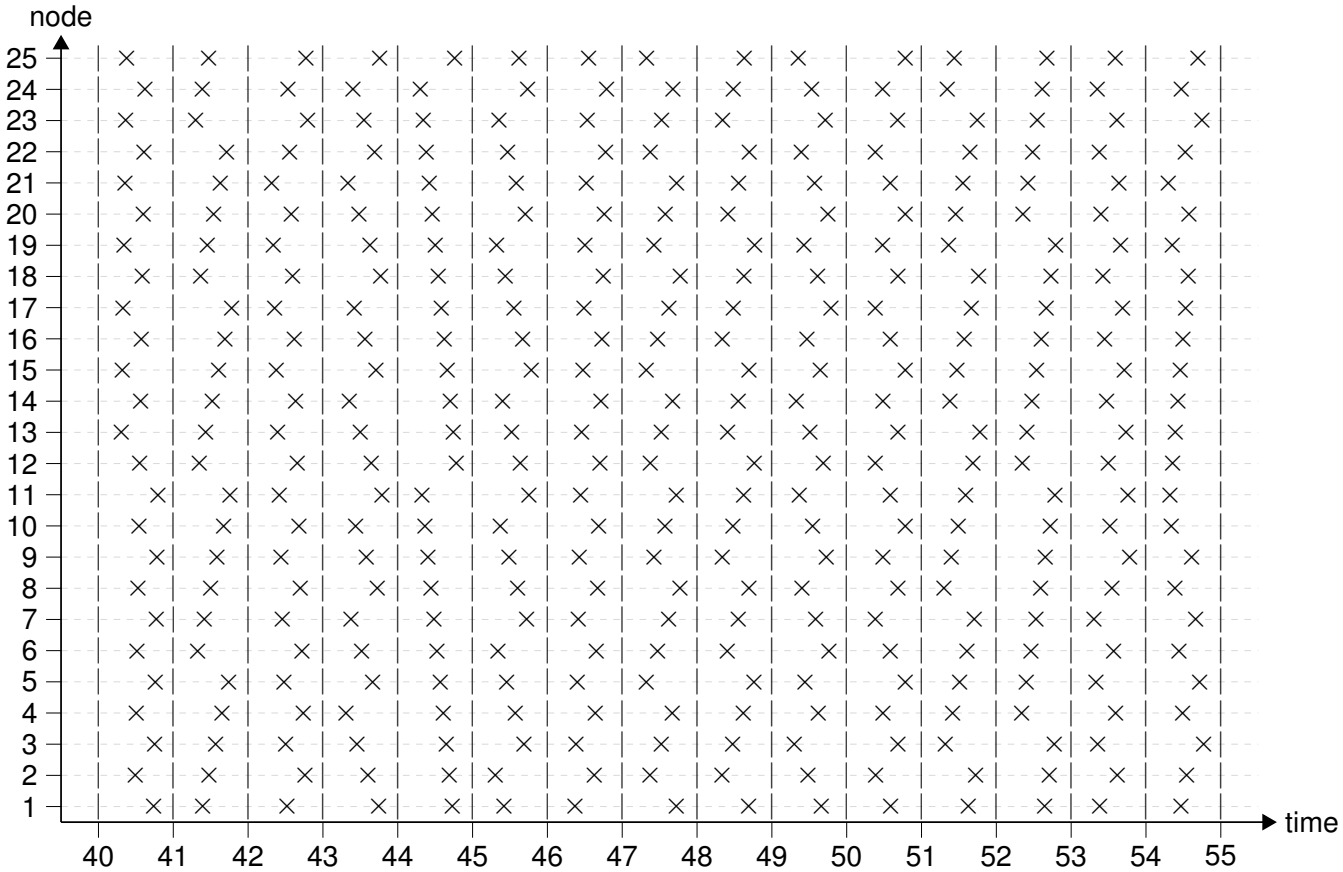
Call-Chain Logging

Node ID	Time [ms]	Direction	Handler ID
5	1320	>	42
5	1322	>	36
5	1323	>	12
5	1324	<	12
5	1328	<	36
5	1333	<	42
3	1648	>	20
3	1649	<	20
7	1930	>	42
7	1931	<	42
...	...	...	...

Call-Chain Logging

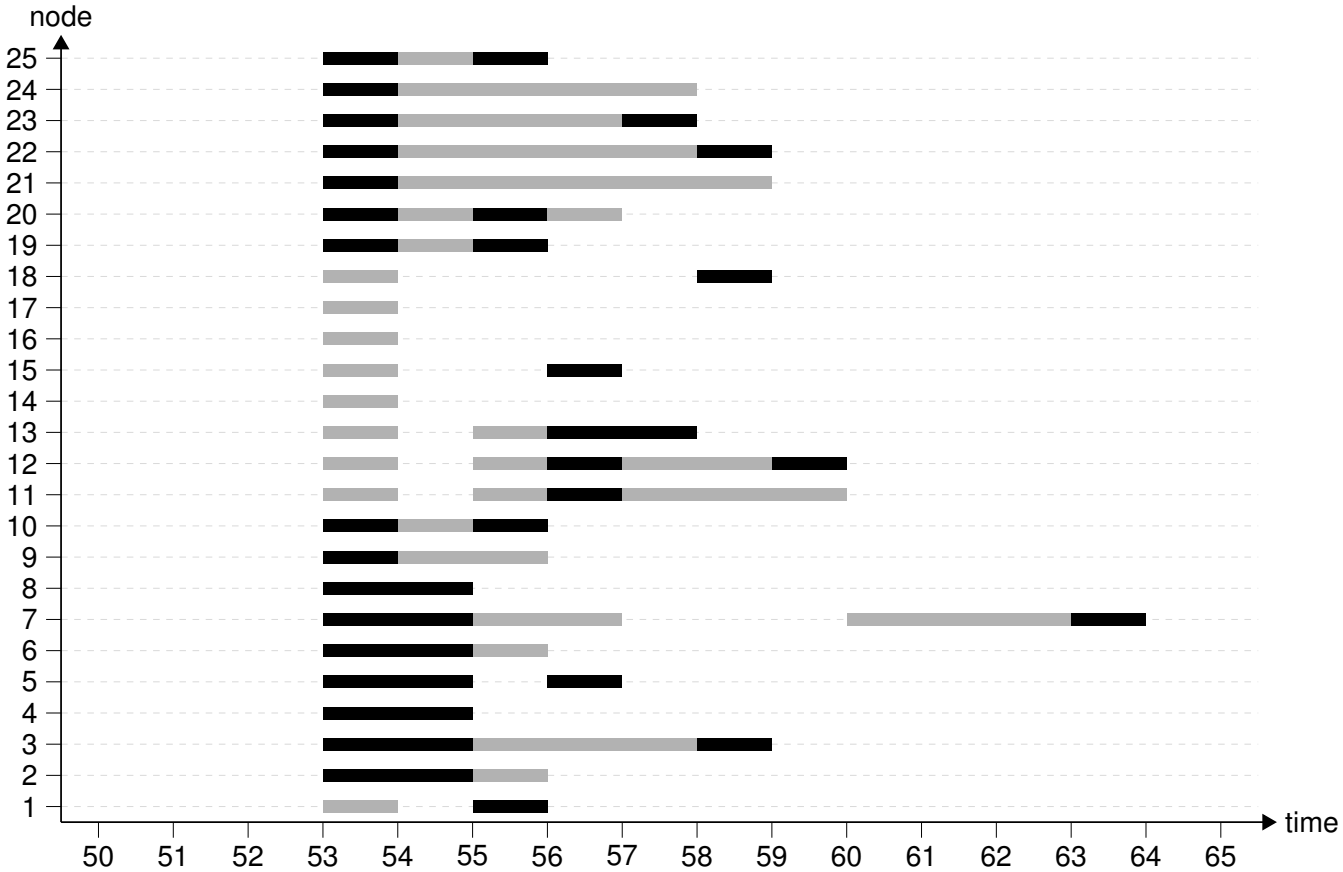
Node ID	Time [ms]	Direction	Handler ID
5	1320	>	42
5	1322	>	36
5	1323	>	12
5	1324	<	12
5	1328	<	36
5	1333	<	42
3	1648	>	20
3	1649	<	20
7	1930	>	42
7	1931	<	42
...	...	...	...

Concept Evaluation: Event Tracing



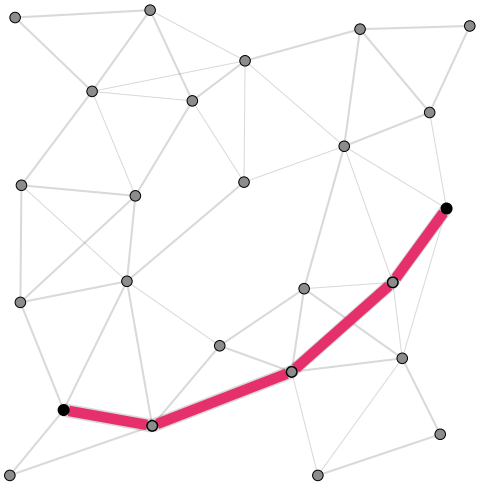
Shortcomings of the random number generator in TOSSIM

Concept Evaluation: State Tracing

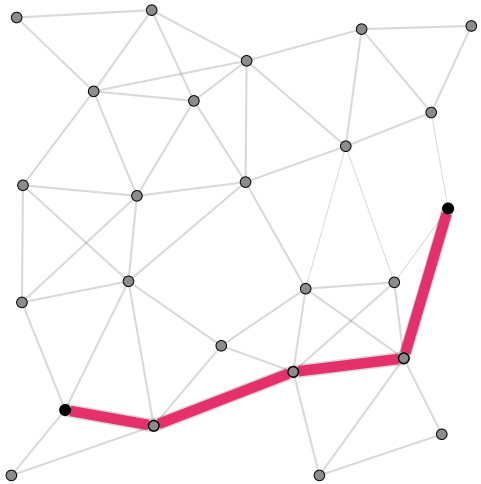


Visualization of program states over time

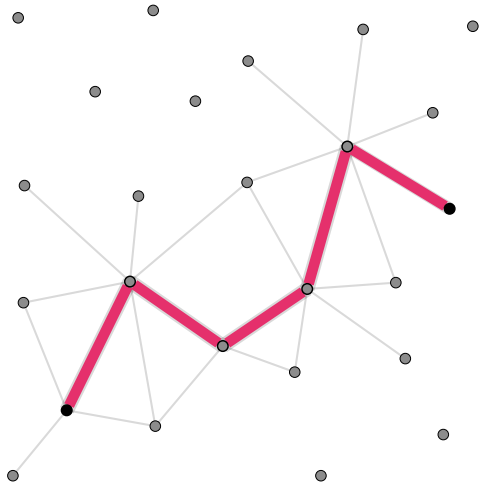
Concept Evaluation: Packet Flow



TYMO



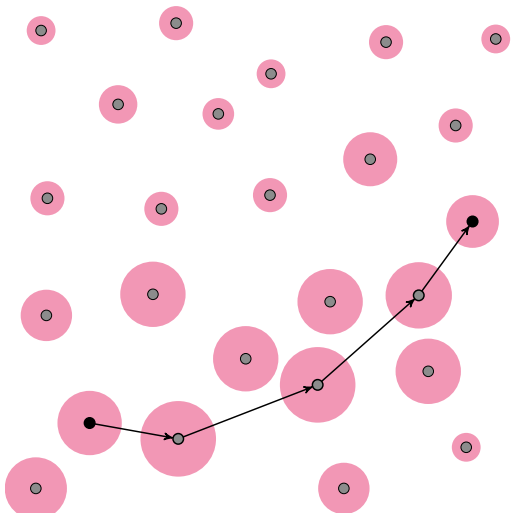
DSR



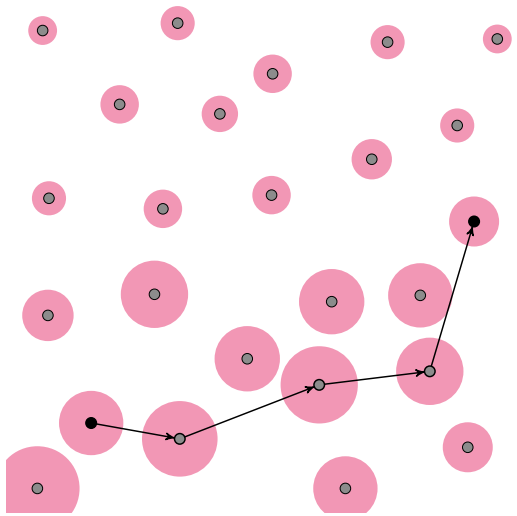
Greedy

- Visualization of number of sent packets over a link
- Identifying routing path decisions

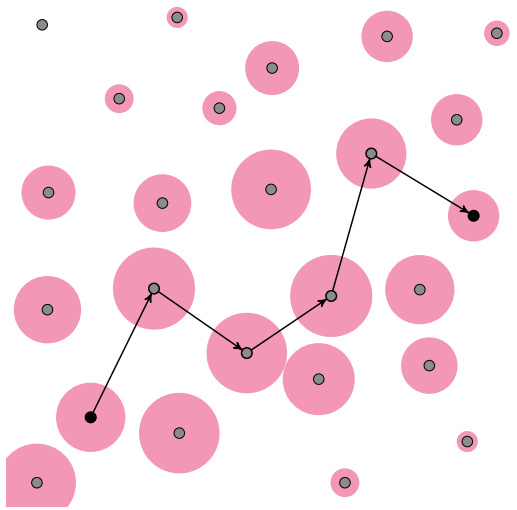
Concept Evaluation: Energy Consumption



TYMO



DSR



Greedy

- Energy consumption based on communication efforts
- Identification of hot-spots