Freie Universität Berlin

Running Real-World Software on Simulated Wireless Sensor Nodes

Georg Wittenburg, Freie Universität Berlin

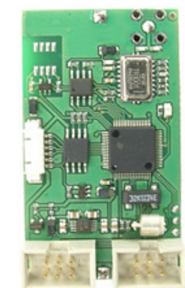
ACM Workshop on Real-World Wireless Sensor Networks (REALWSN'06), Uppsala, Sweden

Outline

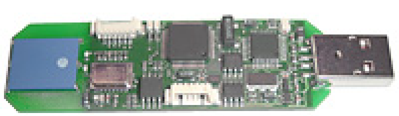
- Background / Motivation / Idea
- Conceptual Perspective / Comparison
- Implementation Details
- A Look into Scalability Issues
- Results / Conclusion

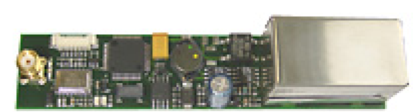
Background (1)

- ScatterWeb WSN Platform:
  - Developed by AG CST at FU Berlin.
  - Project started in 2002.
  - Components commercially available.



- Embedded Sensor Board:
  - TI MSP430 microcontroller
  - TR1001 radio transciever
    - 868 MHz, up to 19.2 kbit/s
  - 2 KB RAM, 8 KB EEPROM

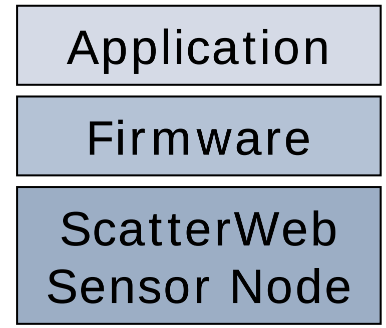




- Embedded Chip Radio
- eGate (USB/Web)

Background (2)

- ScatterWeb Applications:
  - Ad-hoc routing, DSDV
  - Directed Diffusion
  - SQL-like queries
  - RSA, MD2/4/5
  - Localization
  - Virtual Machine Abstraction
  - Rule-Oriented Middleware

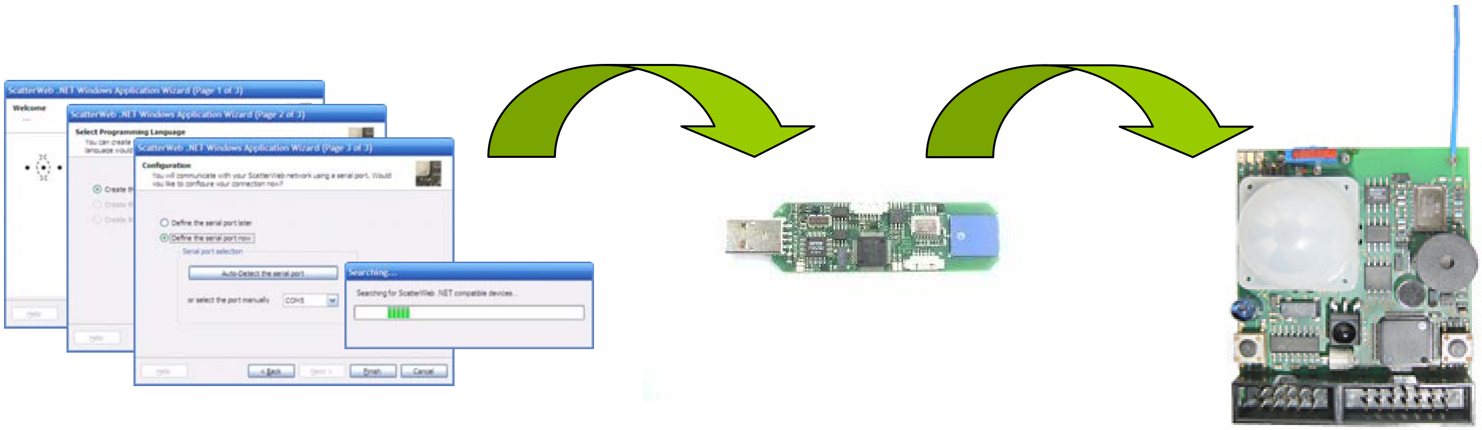


- Firmware provides hardware abstraction.
- Applications link against firmware API (111 C functions).

- More at http://scatterweb.mi.fu-berlin.de.

Motivation

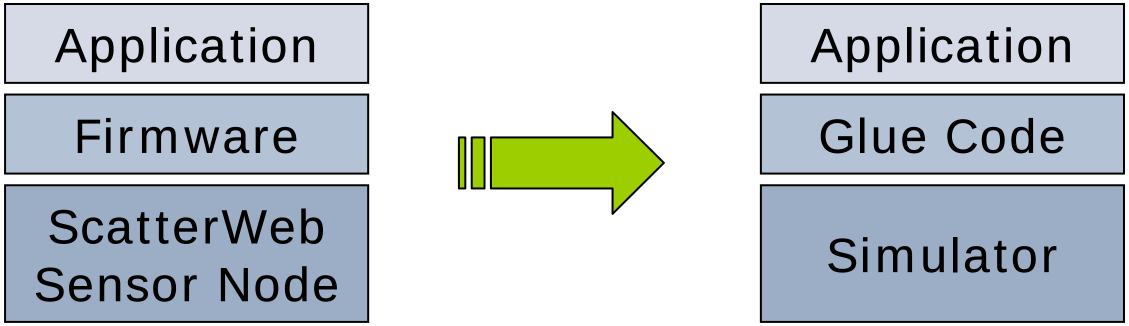
- Development Cycle:
  - Implement, compile, flash to sensor node, test, debug.



- Generally works, but has disadvantages:
  - Real ScatterWeb hardware must be available.
  - Flashing and debugging may be tedious.
  - Testing algorithms on large networks is not feasible.

Idea / Solution

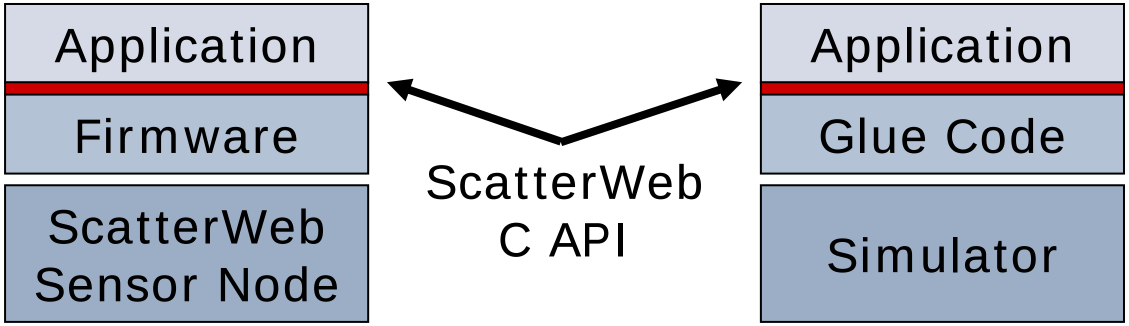
- Use a simulator to develop and test applications.



- Advantages:
  - Faster development cycle.
  - Developer can concentrate on algorithmic issues first and deal with hardware-specific issues later.
  - Algorithms can actually be tested in realistic scenarios under reproducible conditions.

Idea / Solution

- Use a simulator to develop and test applications.



- Preconditions:
  - ScatterWeb firmware has a stable C API.
  - A method exists to port this C API to a network simulator.

- But is there a way to port a C API to a network simulator that is both transparent *and* generic?

Outline

- Background / Motivation / Idea
- **Conceptual Perspective / Comparison**
- Implementation Details
- A Look into Scalability Issues
- Results / Conclusion

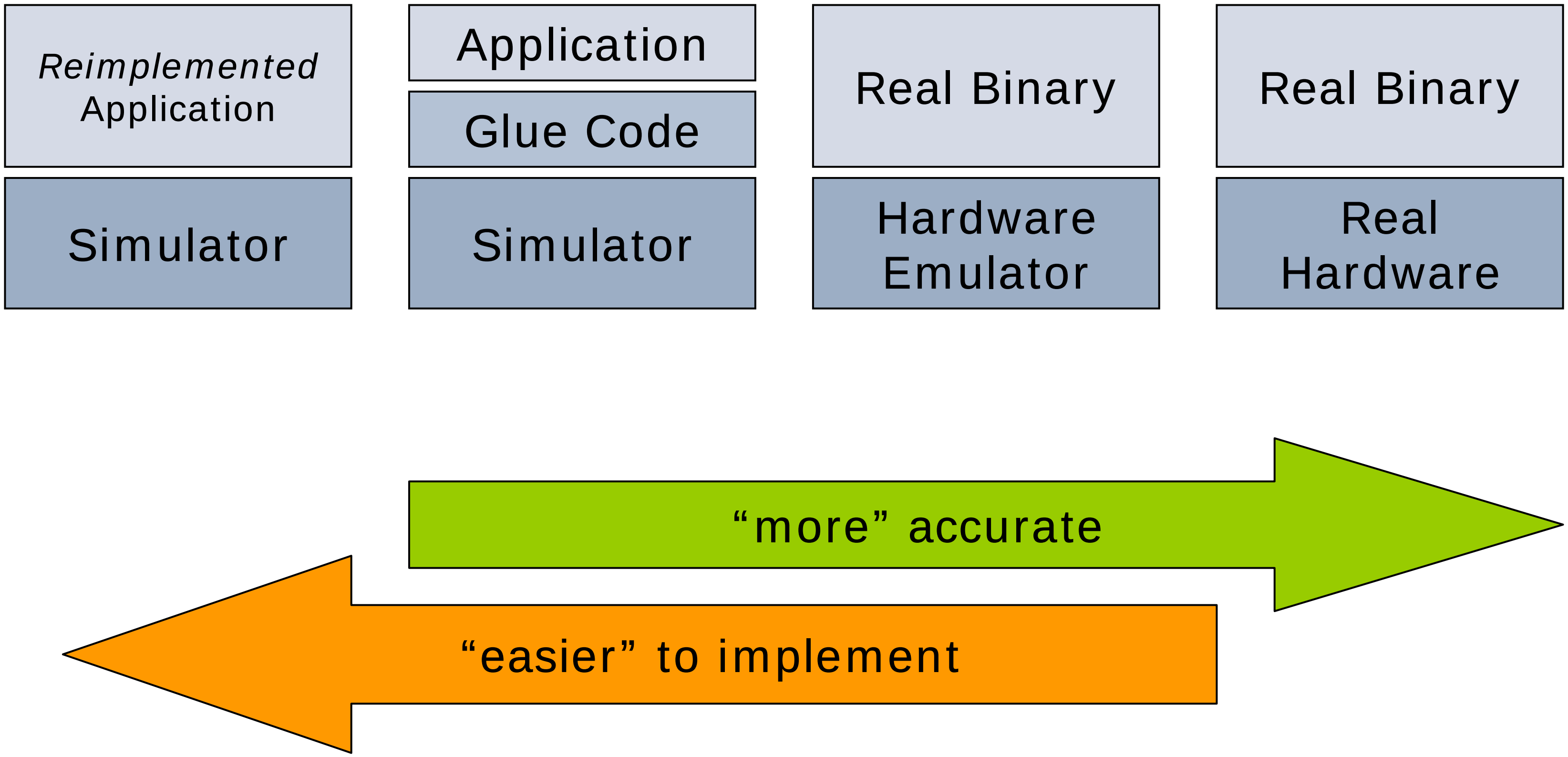
Conceptual Perspective

- Where would this leave us conceptually?

Reimplemented Application	Application	Real Binary	Real Binary
	Glue Code		
Simulator	Simulator	Hardware Emulator	Real Hardware
Plain simulation, e.g. classic ns-2	Simulation based on real code, e.g. TOSSIM [1]	Hardware emulation using real binaries, e.g. VMNet [2], Avrora [3]	Real platform, e.g. ScatterWeb

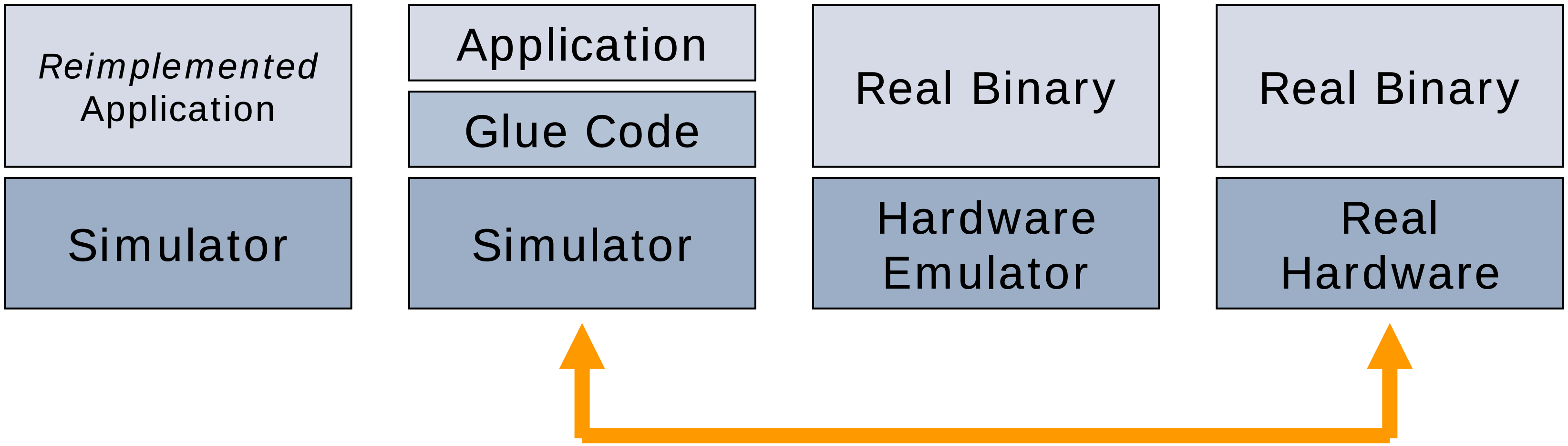
Conceptual Perspective

- Advantages / Disadvantages



Conceptual Perspective

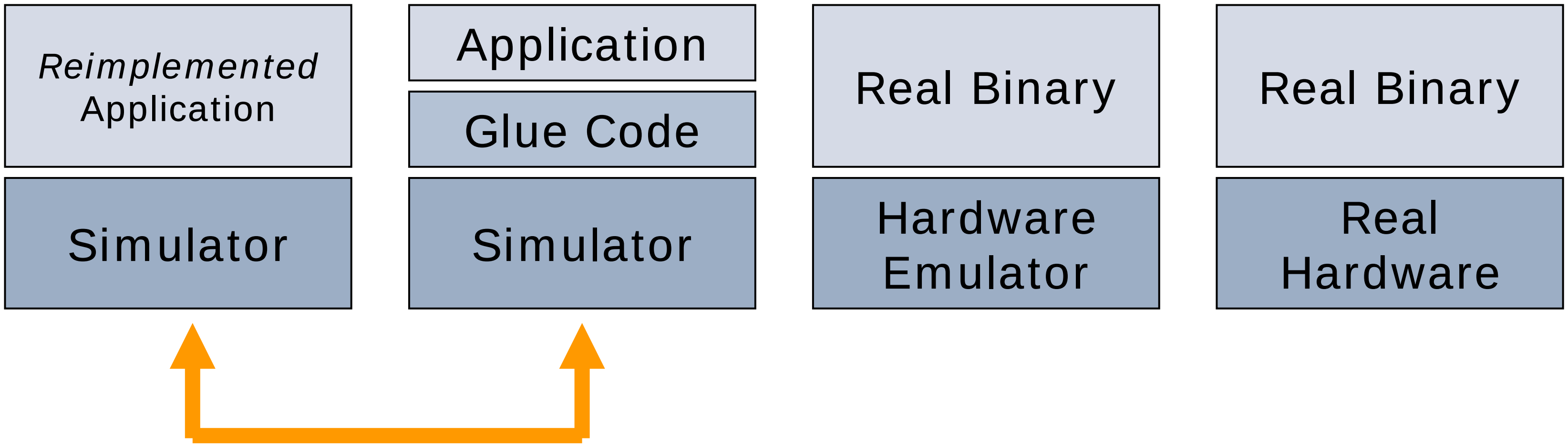
- Advantages / Disadvantages in Detail



- Pro:
  - Faster development cycle.
  - Algorithms first.
  - Testing under reproducible conditions.
- Contra:
  - Less accurate.

Conceptual Perspective

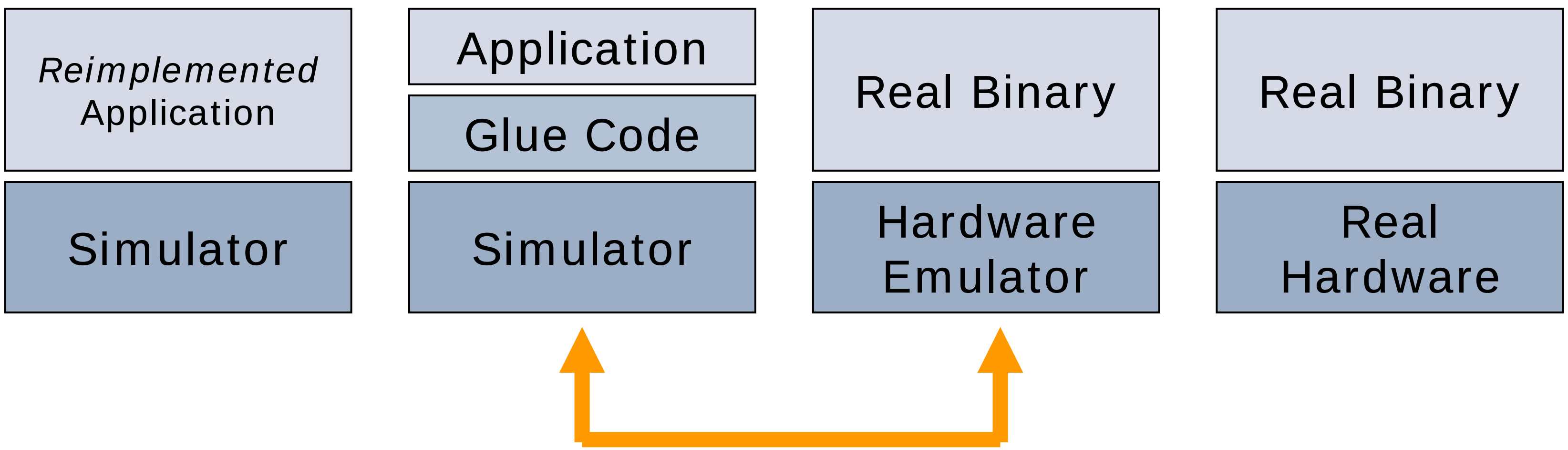
- Advantages / Disadvantages in Detail



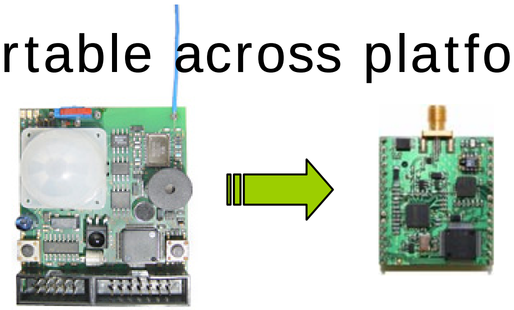
- Pro:
  - No reimplementation, no programming inaccuracies, more realistic simulation.
- Contra:
  - ?

Conceptual Perspective

- Advantages / Disadvantages in Detail



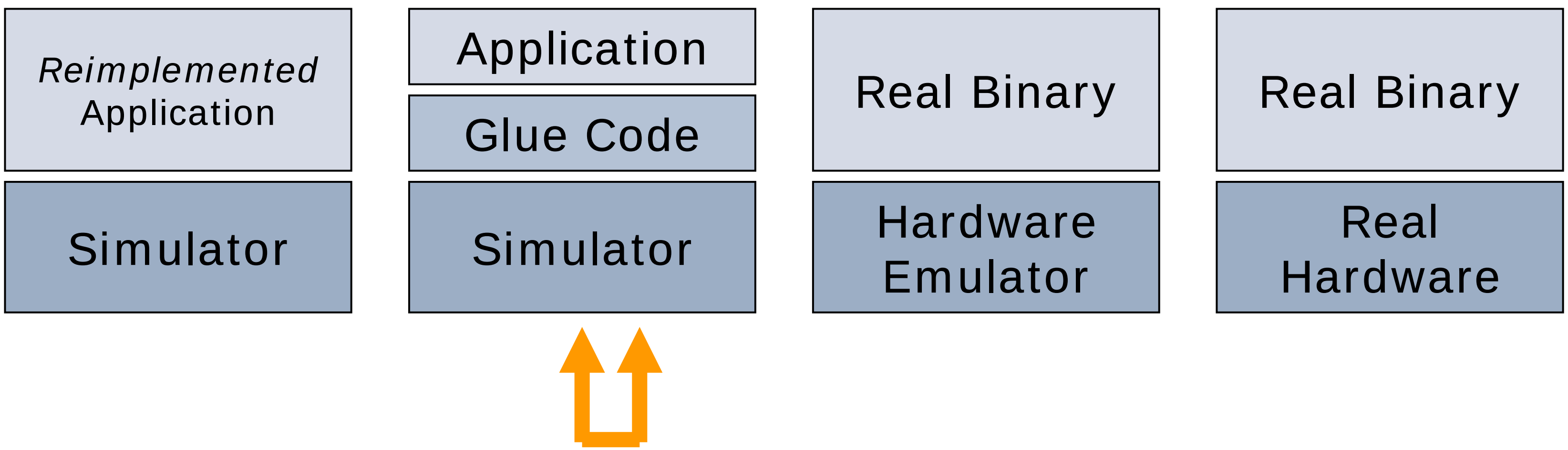
- Pro:
  - Portable across platforms.



- Contra:
  - Possibly less accurate.

Conceptual Perspective

- Advantages / Disadvantages in Detail



- TOSSIM integrates simulator implemented from scratch.

- Pro:
  - Our approach reuses existing simulation technology.
- Contra:
  - ?

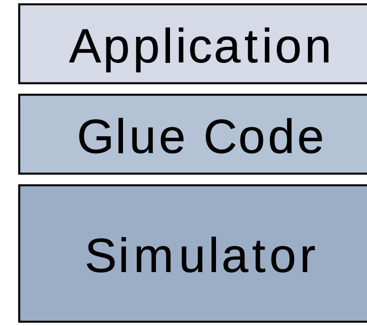
Outline

Freie Universität Berlin

- Background / Motivation / Idea
- Conceptual Perspective / Comparison
- **Implementation Details**
- A Look into Scalability Issues
- Results / Conclusion

Implementation Issues

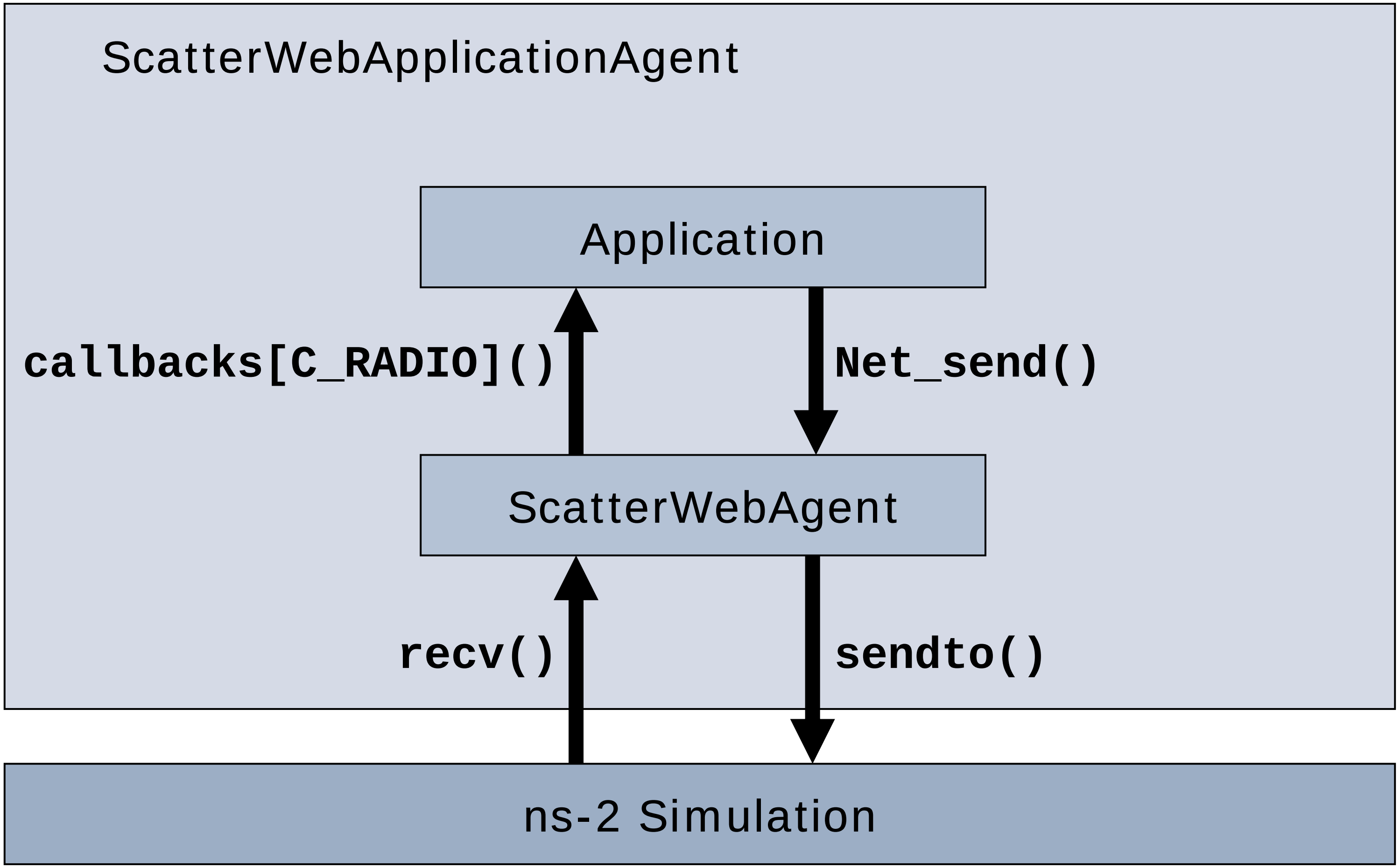
- Why use the ns-2 network simulator [4]?
  - Written in C++, supports linking C code.
  - Open source, compatible license.
  - Large user base.
  - But need to consider scalability issues.



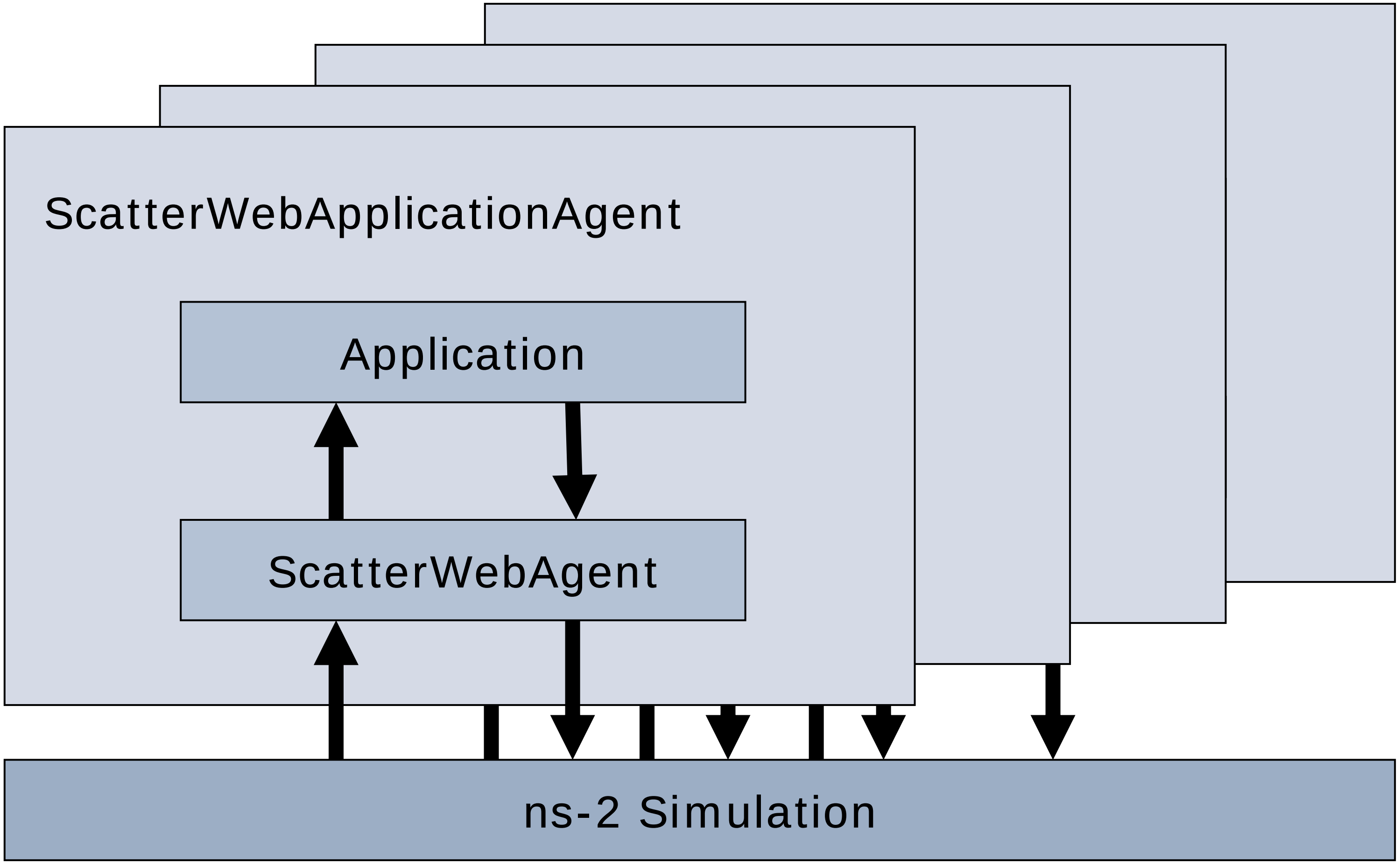
- Key problems when linking application C code into a C++ simulator:
  - Our C code is static, but we need to have dynamic sensor node objects in the simulation.
  - From the point of view of the application, these object must appear as normal sensor nodes.

- The goal is to run ScatterWeb applications within the simulation with as little changes to their code as possible.

Implementation Overview (1)



Implementation Overview (2)



Implementation Details (1)

- How does this work?

```
[...]
#define Net_send ScatterWebAgent::instance->Net_send
[...]
```

Excerpt from ScatterWebFirmwareWrapper.h

```
[...]

#include "ScatterWebFirmwareWrapper.h"

[...]

#include "ScatterWeb.Event.c"
#include "ScatterWeb.Process.c"
```

Excerpt from ScatterWebApplication.cc

Implementation Details (2)

- What changes are necessary in the application code?

```
[...]
#ifndef SCATTERWEB_ON_NS2
int global_variable;
#endif
[...]
```

Excerpt from ScatterWeb.Event.c

```
class ScatterWebApplicationAgent {
      public:
            [...]
            int global_variable;
};

#define global_variable ((ScatterWebApplicationAgent*)
      ScatterWebAgent::instance)->global_variable
```

Excerpt from ScatterWebApplication.cc

Outline

- Background / Motivation / Idea
- Conceptual Perspective / Comparison
- Implementation Details
- **A Look into Scalability Issues**
- Results / Conclusion

A Look into Scalability Issues

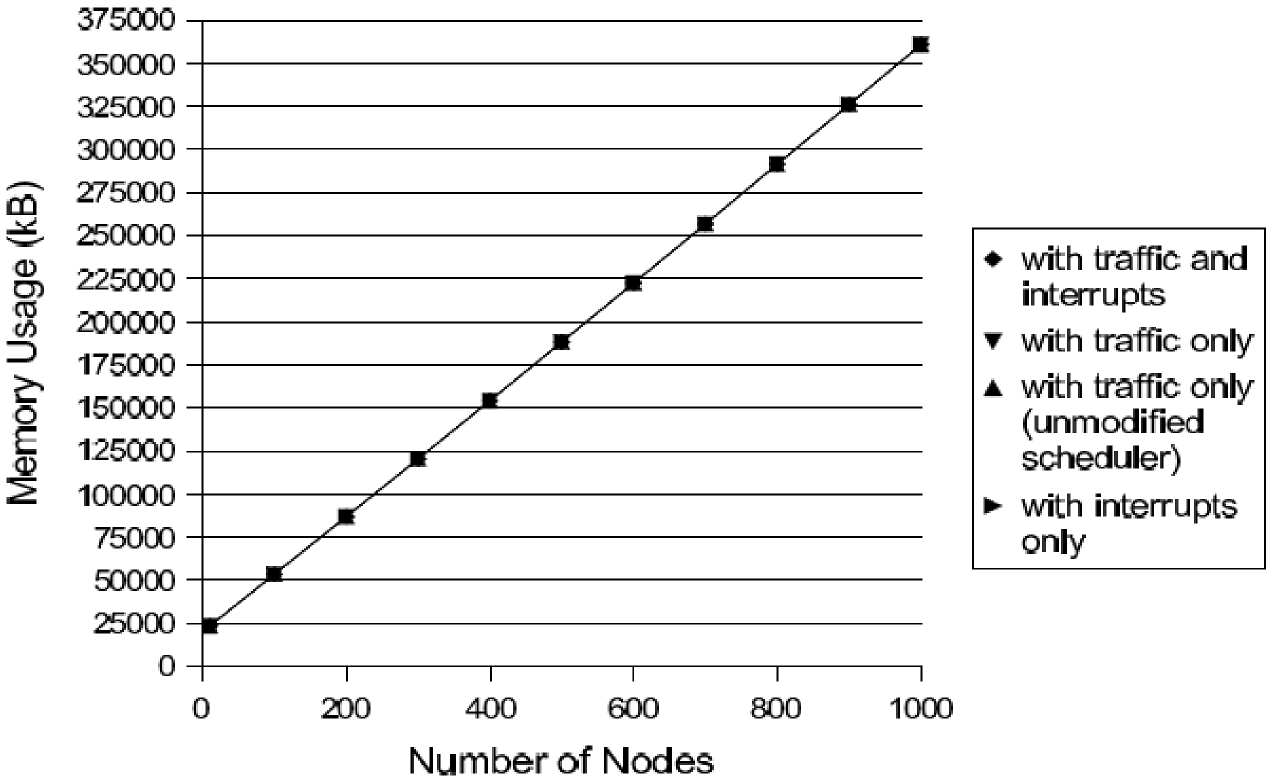
- ns-2 is not exactly famous for its scalability.
  - Addressing this is out of scope.
  - However, make sure not to add *additional* overhead.

- Consider both memory consumption and simulation time.

- Specifically, look at:
  - Different network sizes.
  - Different network densities / average node degrees.
  - Different interrupt loads on simulated sensor nodes.
  - Different traffic loads as incurred by different network densities

Scalability Issues (1)

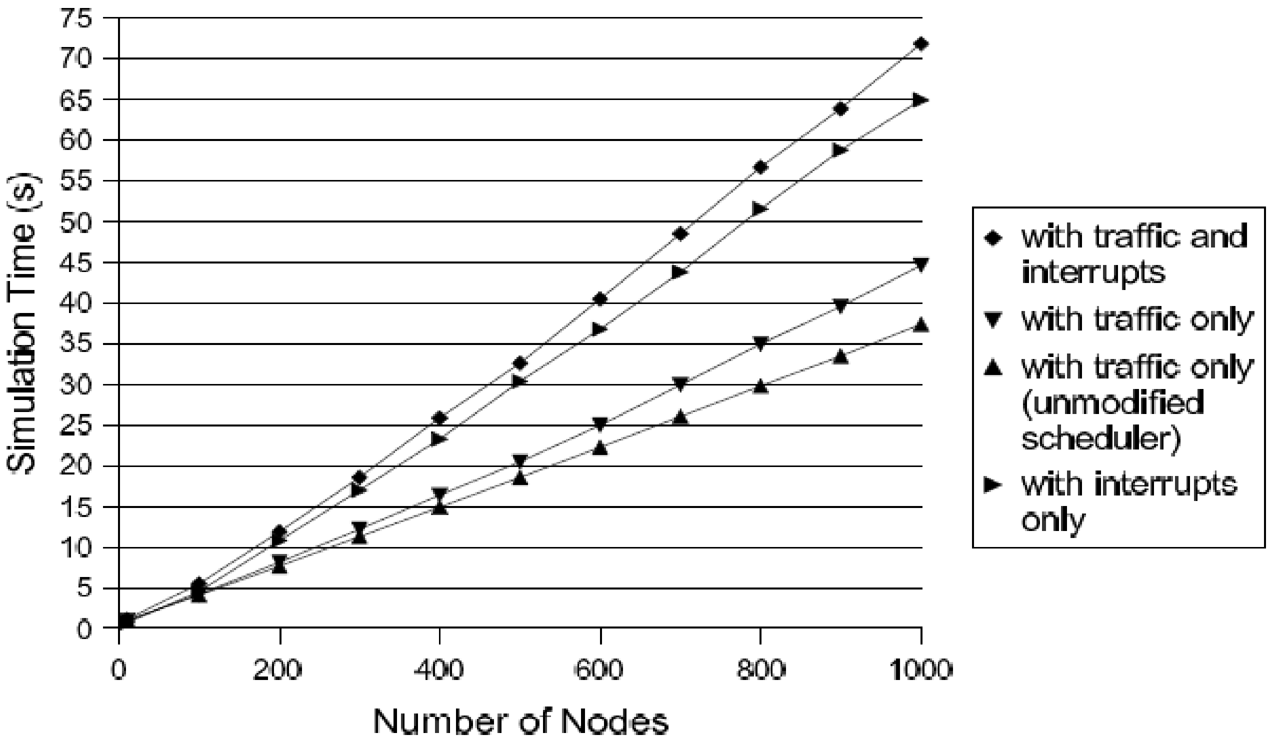
- Number of Nodes vs. Memory Usage:



- Memory usage increases proportionally at about 345 kB per simulated ScatterWeb sensor node.

Scalability Issues (2)

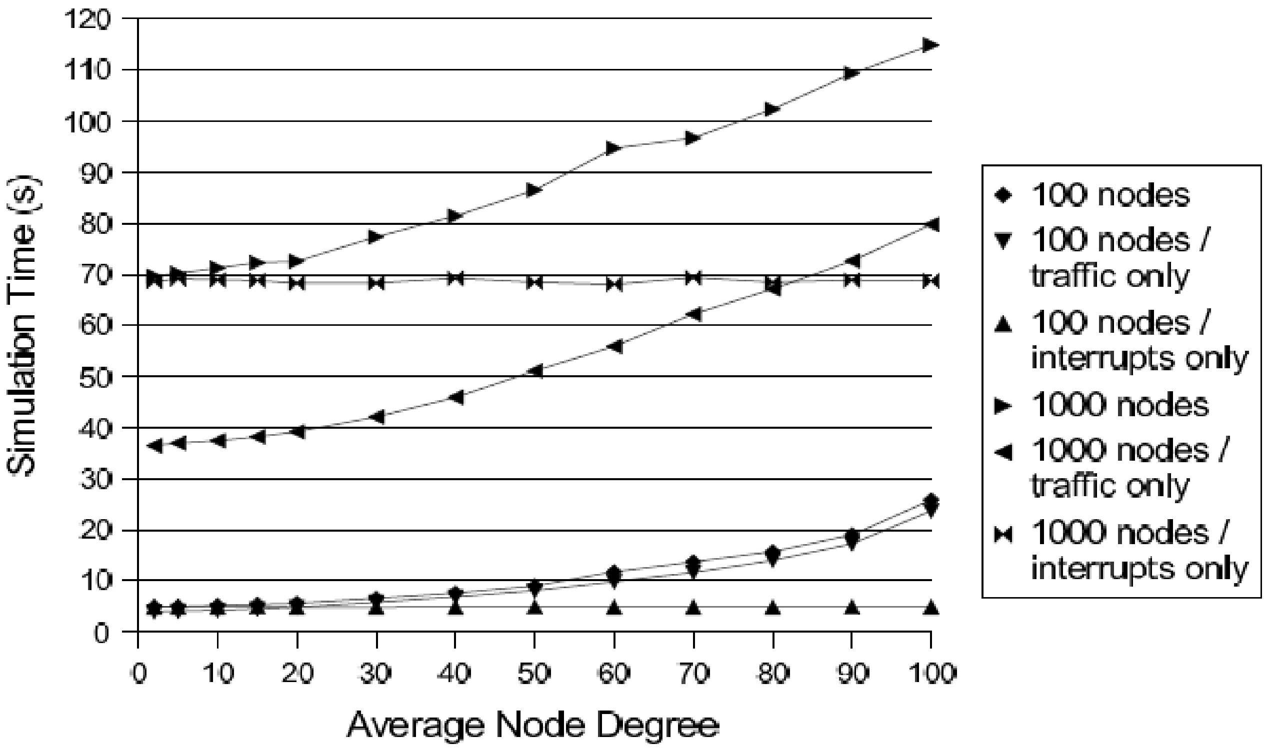
- Number of Nodes vs. Simulation Time:



- Time to complete a simulation run increases proportionally with the number of sensor nodes.

Scalability Issues (3)

- Average Node Degree vs. Simulation Time:

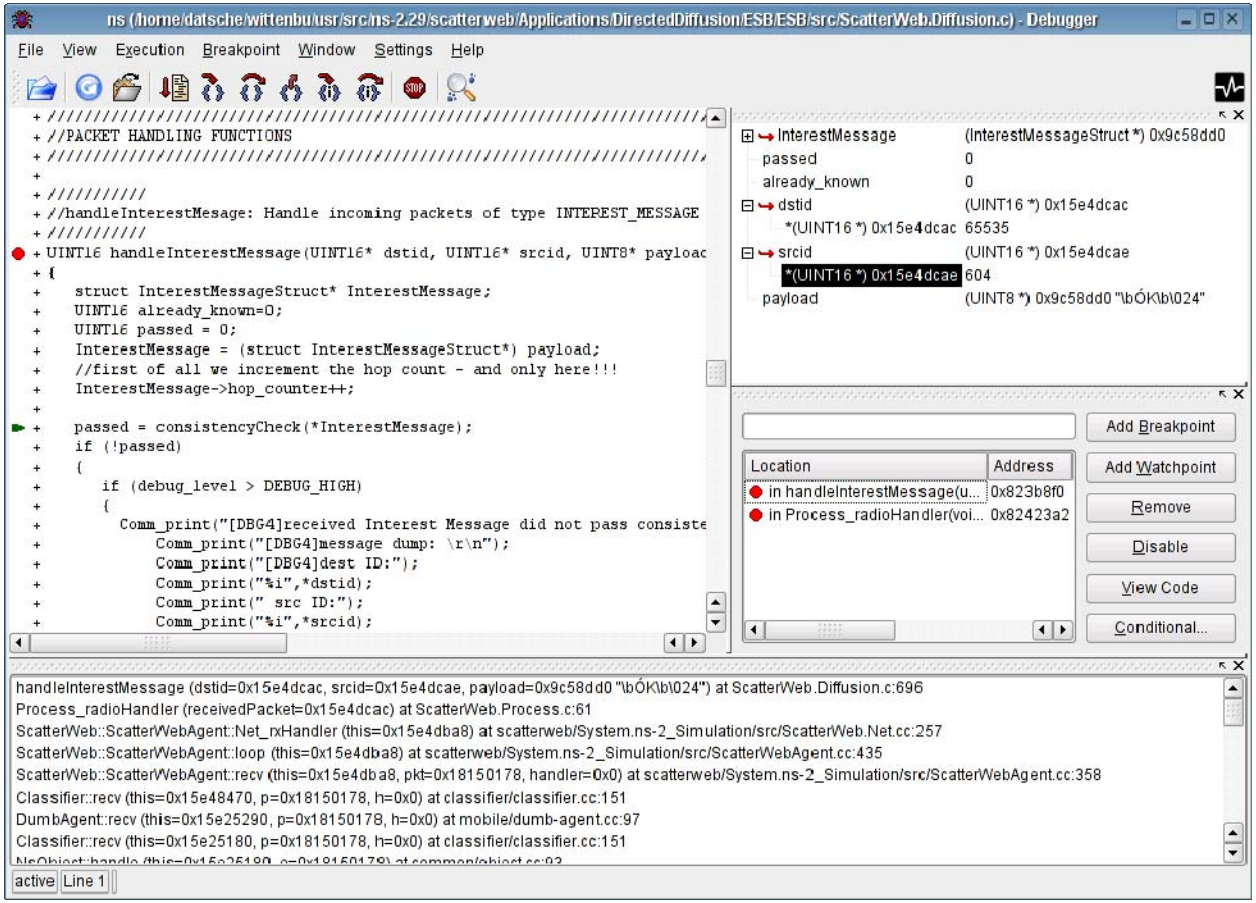


- For test cases with network traffic, simulation time increases polynomially with the average node degree.

Outline

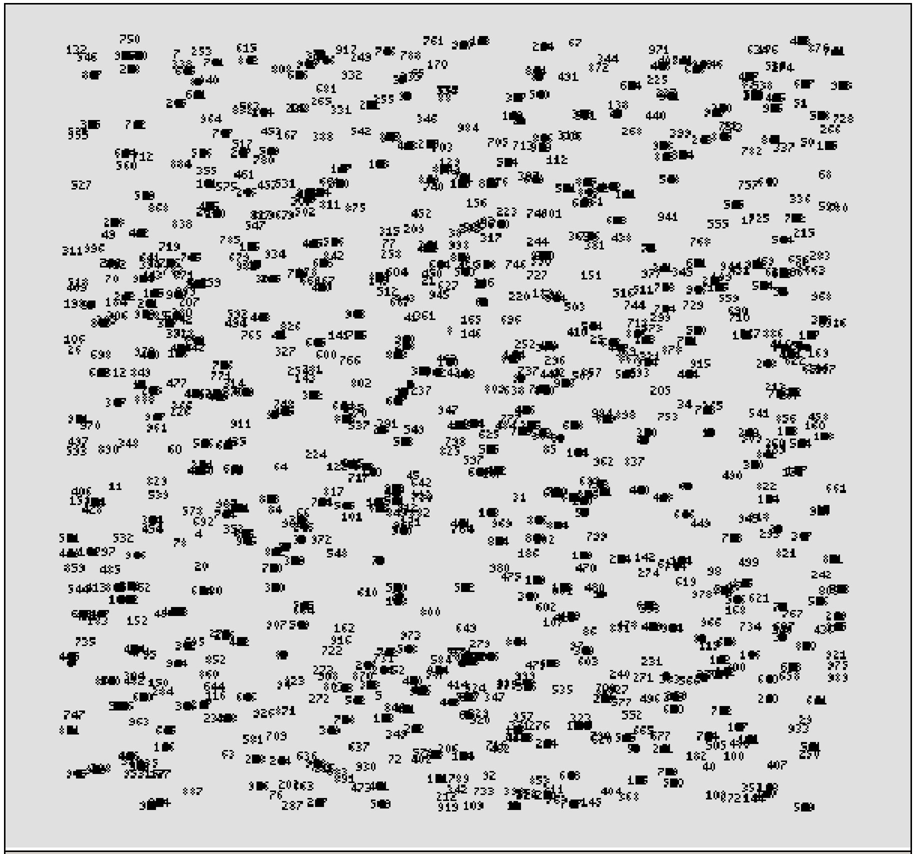
- Background / Motivation / Idea
- Conceptual Perspective / Comparison
- Implementation Details
- A Look into Scalability Issues
- **Results / Conclusion**

Results (1)



- The simulation can be used to implement and debug real ScatterWeb applications.

Results (2)



scatterweb_on_ns-2_running_directed_diffusion.avi

- Existing algorithms can be visualized for the first time.

Conclusion / Future Work

- Porting the API of an existing WSN platform to an existing simulator is possible.
- This approach has several advantages:
  - Faster development cycle with focus on algorithms.
  - Reuse of existing simulation technology.
  - Preserves abstraction / portability of API.
  - Applicable to other projects that use C for hardware abstraction.
- No *additional* run-time overhead is incurred.
- Next step is to quantify the accuracy of our simulation and compare it with the other approaches.

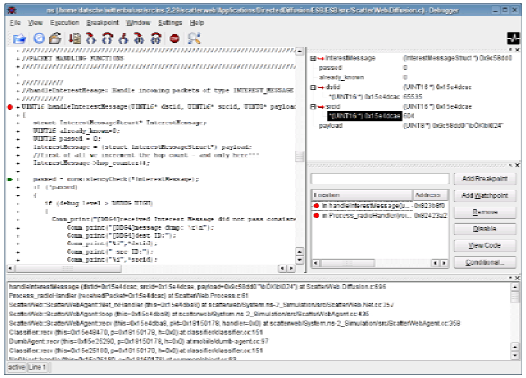
Thank you for your time!

Any questions?

Application

Glue Code

Simulator



More at:
http://www.inf.fu-berlin.de/inst/ag-tech/scatterweb_net/software/ns2.html

References

1. P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003), 2003.
2. H. Wu, Q. Luo, P. Zheng, B. He, and L. M. Ni. Accurate Emulation of Wireless Sensor Networks. In Proceedings of Network and Parallel Computing, IFIP International Conference, NPC 2004, pages 576–583, Wuhan, China, Oct. 2004.
3. B. Titzer, D. Lee, and J. Palsberg. Avrora: Scalable Sensor Network Simulation with Precise Timing. In Proceedings of the Fourth International Conference on Information Processing in Sensor Networks (IPSN'05), pages 477–482, 2005.
4. The Network Simulator – ns-2. http://www.isi.edu/nsnam/ns/.