Internet of Things
- The Quest for Dependability -

Lothar Thiele et al (see last slide)
ETH Zurich, Computer Engineering and Network
Objects collect data autonomously.

Objects act.
Smart World

VIRTUAL WORLD

PHYSICAL WORLD

Computation

reasoning
deciding

Communication

observing

influencing

physical/biological/social processes
Smart World

VIRTUAL WORLD

PHYSICAL WORLD

physical/biological/social processes

end-to-end guarantees
predicability

observing

influencing

reasoning

deciding

Computation

Communication
Society will **increasingly depend** on the Internet of Things. *Dependability and trust* are key for societal acceptance.
Predictability and Dependability

natural hazards

surveillance

desaster management

health

care

factory

automation

mobility
What are reasons for low dependability
(or tremendous engineering effort)?
Reason 1: Resource Constraints

The closer the system works at its resource limits, the less reliable it operates.

Interference is one of the prominent sources.
Reason 2: Adaptivity

- **Adaptivity** and feedback due to
  - changes in environment
  - user interaction
  - switching between (power saving) modes, mixed criticality operation, failures, ...

- Typical for **applications** in
  - surveillance
  - early warning
  - personalized medical and health
  - industrial control, ...

© Mark Dixon
Reason 2: Adaptivity

Example: Acoustic sensing of mountain phenomena

1. acoustic events are detected
2. host decides on relevance
3. switching on other sensor modalities
4. high rate data communication
Challenges

- Predictable
- Adaptive
- Low energy
- Sleep well
- Operate efficiently
- Avoid interference
- Bound interference
- Wake-up quickly
- Switch gears quickly
System View

wireless network

C

A

S

... physical process

S

A

physical process
System View

wireless network

physical process

physical process
Multi-Hop Communication

© Siemens Building Technologies
Wireless Mesh Networks
Wireless Mesh Networks
Wireless Mesh Networks
Can we expect that this construction works reliably and efficiently?
Can we expect that this construction works reliably and efficiently?

No!

Complex and non-deterministic interference & distributed state: How could we provide tight bounds?
Synchronous Transmissions

Transmissions are overlapping in time and space.

Nevertheless, the receiver decodes useful information.

Implicit redundancy in time and space.

Do not avoid interference but use it to your advantage!
Broadcasting with Synchronous Transmissions

- no explicit routing
- no network state
The reliability stems from redundancy

**Sender diversity:**
Synchronous transmissions over different channels

**Spatio-temporal diversity:**
Receive from different sets of nodes at different times
The reliability comes from diverse redundancy

Sender diversity:
Synchronous transmissions over different channels

Spatio-temporal diversity:
Receive from different sets of nodes at different times
Low Power Wireless Bus: Time-triggered communication

Low Power Wireless Bus: Time-triggered communication

- Rounds
- Slots
  - Current schedule
  - Data
  - Requests
  - Compute schedule
  - New schedule

Low Power Wireless Bus: Time-triggered communication

- **Rounds**
- **Slots**
  - current schedule
  - data
  - requests
  - compute schedule
  - new schedule
- **Flooding**

Low Power Wireless Bus: Time-triggered communication

- Rounds
- Slots
  - current schedule
  - data
  - requests
  - compute schedule
  - new schedule
- Floods

Based on synchronous transmission
LWB: Interference Management

Make use of interference

Low Power Wireless Bus

- energy (resource) efficient
- adaptive
- robust and predictable
- provable guarantees

Partitioning in time

[Rounds]

[Slots]
[Current schedule] data [...]

[Ferrari et al: Glossy & Low Power Wireless Bus]
Provide **hard real-time guarantees** via a flexible and adaptive bandwidth (slot) assignment algorithms.

Build **higher level protocols** that guarantee global properties and achieve consensus.

Build a formal model that allows to estimate the **reliability** and **energy consumption** with a high accuracy.
Providing guarantees: three sub-problems

- **Determine when next round starts**
  Defer as much as possible to save energy without causing deadline misses

- **Allocate messages to slots in a round**
  As many as possible, prioritized according to earliest deadline first (EDF) principle

- **Perform admission control**
  - Ensure there is enough bandwidth to serve the highest possible demand
Providing guarantees: three sub-problems

- **Determine when next round starts**
  Defer as much as possible to save energy without causing deadline misses
Providing guarantees: three sub-problems

• **Determine when next round starts**
  Defer as much as possible to save energy without causing deadline misses

\[ \text{#streams} < \text{start time of first packet } S_i, \text{ period } P_i, \text{ relative deadline } D_i > \]
Providing guarantees: three sub-problems

- Determine when next round starts
  Defer as much as possible to save energy without causing deadline misses
Real-time scheduler:

\[
< \text{start time of first packet } S_i, \text{ period } P_i, \text{ relative deadline } D_i >
\]
Real-time scheduler:

Start → Stream request? → Yes → Compute synchronous busy period → Admission control → Compute start time of next round → Slot allocation → End

No → < start time of first packet $S_i$, period $P_i$, relative deadline $D_i$ >

Rounds, allocation of packets and data slots

What is the latest starting time for a new round?

Example of stream requests:

- 3(0,5,4)
- 4(2,7,5)
- 5(1,15,12)

Future demand function:

- \( h(t) = \sum_{j=1}^{n} \left\lfloor \frac{(t - d_j)}{P_j} \right\rfloor + 1, \) if \( d_j \leq t \)
- \( 0, \) otherwise

End of last round:
What is the latest starting time for a new round?

example of stream requests

future demand function

future demand (# slots)

end of last round

$h_t(t) = \sum_{j=1}^{n} \left\lfloor \frac{(t - d_j)}{P_j} \right\rfloor + 1, \quad \text{if } d_j \leq t
\quad 0, \quad \text{otherwise}$
What is the latest starting time for a new round?

example of stream requests

future demand function

future demand (# slots)

end of last round

where do we stop?
What is the latest starting time for a new round?

Example of stream requests:

1<‐2, 2, 2>
1<‐3, 3, 3>
4<‐25, 25, 25>

Future demand function (B=2):

\[ \omega^0 = 0 \]
\[ \omega^{m+1} = \frac{1}{B} \sum_{i=1}^{n} \left( \left\lfloor \frac{\omega^m}{P_i} \right\rfloor + 1 \right) \]
\[ \omega^{m+1} = \omega^m \Rightarrow T_b = \lfloor \omega^m \rfloor \]
What is the latest starting time for a new round?

Let $T_b$ be the synchronous busy period of the stream set $\Lambda$, $T_{\text{max}}$ the largest time by which the next round can be delayed after the previous one, and $B$ the number of data slots available in a round. Using $\text{LS}$, the start time of each round $t_i$ for all $i = 0, 1, \ldots$ is given by

$$ t_{i+1} = \min(t_i + T_{\text{max}}, T_i), $$

where $t_0 = -1$ and $T_i$ is given by

$$ T_i = \min_{t \in D_i} \left( t - \left\lfloor \frac{h_i(t)}{B} \right\rfloor \right). $$

Within a round of $\text{LS}$, packets are sent according to $\text{EDF}$.

set of unsent deadlines in $[t_i + 1, t_i + T_{\text{max}} + T_b + 1]$
What is the latest starting time for a new round?

Let $T_b$ be the synchronous busy period of the stream set $\Lambda$, $T_{max}$ the largest time by which the next round can be delayed after the previous one, and $B$ the number of data slots available in a round. Using LS, the start time of each round $t_i$ for all $i = 0, 1, \ldots$ is given by

$$t_{i+1} = \min(t_i + T_{max}, T_i),$$

where $t_0 = -1$ and $T_i$ is given by

$$T_i = \min_{t \in D_i} \left( t - \left\lfloor \frac{h_i(t)}{B} \right\rfloor \right).$$

Within a round of LS, packets are sent according to EDF.

The LS policy is real-time optimal and minimizes the communication energy consumption within the limits of the underlying LWB communication support.
How can we perform admission control?

Critical instant: $S_i = 0$

All deadlines in $[0, T_b]$
Various considerations

How do we best determine busy periods, demands and EDF scheduling on a resource constrained platform?

\[ h_i(t) = \sum_{j=1}^{n} \begin{cases} \left\lfloor \frac{(t - d_j)}{P_j} \right\rfloor + 1, & \text{if } d_j \leq t \\ 0, & \text{otherwise} \end{cases} \]

\[ \omega^{m+1} = \frac{1}{B} \sum_{i=1}^{n} \left( \left\lfloor \frac{\omega^m}{P_i} \right\rfloor + 1 \right) \]

- Use a special class of a one-level bucket queue with constant time Insert, Delete, UpdateKey and linear time (in \( P \)) First and Next. [Zimmerling et. al.: ACM TECS 2017]

How do we determine the worst-case runtime that depends on \( n, P, \) bandwidth demand \( u \) and \( T_b \)?

- Solve two connected ILPs.
Some experiments (30 nodes - MSP 430 CPU)

- Bootstrapping
- Stable operation
- $<0, 6, 3>$
- $<0, 6, 6>$
- Deadline change
Some experiments (94 nodes, w-iLab.t)

Setup:
- 90 source nodes, 3 destination nodes, 1 host node
- $B=51$ slots per round, packet size 10 Byte, base period 1s (and also 100ms)
- 180 streams, bandwidth demand between 2.9% and 19.4%

Some results:
- Deadline success ratio 99.97% (< 100% due to packet loss)
- Much less energy in comparison to greedy scheduling
But is the underlying concept competitive?
Early Evaluation of Synchronous Transmission

- Four testbeds

<table>
<thead>
<tr>
<th>Testbed</th>
<th>TWIST</th>
<th>KANSEI</th>
<th>CONETIT</th>
<th>FLOCKLAB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Location</td>
<td>TU Berlin</td>
<td>Ohio State Univ.</td>
<td>Univ. of Seville</td>
<td>ETH Zurich</td>
</tr>
<tr>
<td>Nodes</td>
<td>90</td>
<td>260</td>
<td>26 (5 mobile)</td>
<td>55</td>
</tr>
<tr>
<td>Diameter</td>
<td>3 hops</td>
<td>4 hops</td>
<td>3 hops</td>
<td>5 hops</td>
</tr>
</tbody>
</table>

- Seven combinations of routing+MAC protocols

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Many-to-one</td>
<td>CTP+{CSMA, LPL, A-MAC}, Dozer</td>
</tr>
<tr>
<td>Many-to-many</td>
<td>Muster+{CSMA, LPL}</td>
</tr>
<tr>
<td>Mobile sink/sources</td>
<td>BCP+CSMA, CTP+CSMA</td>
</tr>
</tbody>
</table>
Many To One: Light Traffic

One sink, low data rate, static nodes
- Twist: 85 nodes, 1 packet every minute

Data yield
- Avg: 99.97%
- Min: 99.45%

Radio duty cycle
- Avg: 1.69%
- Max: 1.90%
Many To One: Fluctuating Traffic

Flocklab (5 hops):
1 sink, 54 sources
14 with varying generation period

• LWB promptly adapts to varying traffic load
  • Round period $T$

• Additional complexity to make Dozer and LPL adaptable
Static vs. Mobile Sink

One sink, high data rate, mobile nodes

- DSN + FlockLab: 43 nodes (6 mobile), 1 packet every second

Per-node data yield

- Mobility does not affect data yield (average: 99.74 %)
- Same implementation and parameters as in static scenarios
DEPENDABILITY COMPETITION

Carlo Alberto Boano
Graz University of Technology, Austria
Evaluation Scenario

- Crowded RF spectrum
- RF interference generated using JamLab in the 2.4 GHz band
The best protocols use the design principles just presented.

2016-2019
Can this be implemented on standard hardware?
   Yes!

Is it trivial?
   No!
System View

wireless network

Hardware Platform

physical process

physical process

Wireless Node Platform

Classic Architecture

Sensor Interface  Network Protocol

Single Processor

memory interference  time-interference  power-interference
Wireless Node Platform

Classic Architecture

- Sensor Interface
- Network Protocol
- Single Processor

Event-Based Architecture

- Sensor Interface
- Network Protocol
- APP
- BOLT
- COM

firewall

memory interference  time-interference  power-interference
Bolt, in short
Bolt, in short

- Application Processor (AP)
  - BOLT API
  - Receive Buffer
  - flush
  - read
  - write

- Communication Processor (CP)
  - BOLT API
  - write
  - flush
  - read
  - Receive Buffer

Predictability
Reliability
Performance
Tight bounds on the API execution time
Bolt, in short

Application Processor (AP)

BOLT API
flush
read
write

Receive Buffer

Communication Processor (CP)

BOLT API
write
flush
read
Receive Buffer

Predictability
Reliability
Performance
Non-volatile memory
Bolt, in short

- Predictability
- Reliability
- Performance

Fast
- Low-power $\mu W$ to $m W$
- Sleep power $1.3 \mu W (BOLT)$
- Total $8.1 \mu W$
End-to-End Guarantees

wireless network

physical process

physical process
Challenges

- **Timeliness**: All distributed applications meet their end-to-end deadlines.
- **Adaptability**: The system adapts to dynamic changes at runtime.
- **Efficiency**: The system supports short end-to-end latency (in the ms range), and optimizes its energy consumption and bandwidth utilization.

Design Principles

- **Timeliness**: Time-Triggered Co-Scheduling of Computation and Communication.
- **Efficiency**: Quasi-Static Schedules and Runtime Control
- **Adaptivity**: Multiple Operation Modes
Overview of TTW

Application Model

offline
Scheduling and Optimization

Platform Data

online
Scheduling Tables

Runtime

Mode Change Request

- network parameters
- w.c. execution times
- new mode
- change id
Application Model

Set of applications:

Mode change graph:
Scheduling and Optimization

- Casting all requirements into a Mixed Integer Linear Program (MILP), maximizing message deadlines.
- Standard, besides using network and real-time calculus for composing rounds:

\[
\begin{align*}
\text{next hyperperiod:} & \quad \text{af}_i : t \mapsto \left\lceil \frac{t - m_{i.o}}{m_{i.p}} \right\rceil + 1 \\
\text{af}_i : t \mapsto \left\lceil \frac{t - m_{i.o} - m_{i.d}}{m_{i.p}} \right\rceil \\
\text{next hyperperiod:} & \quad \text{df}_i(t) \leq \text{sf}_i(t) \leq \text{af}_i(t) \\
\text{sf}_i(r_j.t + T_r) \leq \text{af}_i(r_j.t) \\
\text{sf}_i(r_j.t) & \geq \text{df}_i(r_j.t + T_r)
\end{align*}
\]
Mode-Change Protocol

• Example of a mode graph:

• Challenge: Do not violate deadlines of persistent applications.

• Solution: Keep scheduling of persistent applications constant:

Conflict in mode $M_4$

Reservation of the schedule of $A_1$ when scheduling $A_5$ in mode $M_3$
TTW: Can it be Implemented in Practice?

- Application Model
- Scheduling and Optimization
- Scheduling Tables
- Runtime
- Platform Data
- Mode Change Request

Offline:
- Network parameters
- W.C. execution times

Online:
- New mode
- Change id
Example: Remote Control – Combining Everything


Example: CPSWEEK 2019

controller

2 inverted pendulums
Resulting key properties

- Long small end-to-end delay
- Significant negligible jitter
- Correlated i.i.d packet losses
Experiments reveal a **negligible jitter** of $\pm 23 \, \mu s$

For update intervals up to 100 ms
Cyber-Physical Testbed

3-hop low-power wireless network
Carts move in concert
Two inverted pendulums

One remote controller
Reality or Fiction?
Monitoring in the Extreme - Permasense
Why?
Development of early warning systems

New scientific knowledge about geophysical processes

- When is rockfall happening?
- What are its causes?
- What are the effects of climate change?
- How does this influence our environment and habitat in the Alps?
Our Field Sites
Real-time Experimentation at Valley-Scale
Camera

MEM relay sensor

AE-trigger Node

Rock surface

sensors

pre-processing

communication

data cleaning & processing

geophysical processes

society & early warning
5 field sites in surveillance
> 10 years autonomous operation
> 1 Billion data points


If Anything Can Go Wrong

IT WILL

Murphy's Law
time synchronization
packet loss
communication disconnect
buffer overflow
failing interconnects
energy harvesting
sensor fault
data analysis
computing faults

If Anything Can Go Wrong, It Will
Murphy's Law
Society will **increasingly depend** on the Internet of Things. **Dependability and trust** are key for societal acceptance.

Are there **new threats** on the horizon?
Zero-power systems (energy harvesting)

Node/host failures  
Transient systems

Sensor data from low-cost sensors

Decision making on the edge

Security and privacy
Complex Sensing: Acoustic Emissions and Micro-Seismics
Integration of Microseismic Sensors

Sensor Measurements  Wireless Network  Data Management

Example: Acoustic Monitoring in the Alps

All elements built in:
• BOLT Interface
• BLINK Real-time Protocol
• Event-driven architecture
Complex Signals Reveal the Formation of Cracks

Unprecedented Levels of Detail
We are not the only ones making noises up there...
Classification and Cleaning of Data

Challenge:

• Cleaning data from external influences and classification

• Low-power operation
Intelligent Triggered Sensors

- Moving the decision into the sensor
- Local **co-detection** over many sensors
- Less data, less power
Example: Acoustic Monitoring in the Alps

We use **machine learning** to classify acoustic signals

- mountaineers, helicopters, wind, hail
- relevant acoustic events from geophysical processes

---

Zero-power systems (energy harvesting)

Node/host failures

Sensor data from low-cost sensors

Decision making on the edge

Transient systems

Security and privacy
Acknowledgements

ETH Zurich
- Computer Engineering and Networks Lab
- Geodesy and Geodynamics Lab
- Micro and Nanosystems

Federal Office for the Environment

University of Zurich
- Department of Geography

EMPA
- Natural Resources & Pollutants

Current and Previous Contributors:
- Jan Beutel, Olga Saukh, Zimu Zhou
- Reto da Forno, Andres Gomez, Tonio Gsell, Romain Jacob, Balz Maag, Matthias Meyer, Lukas Sigrist, Mina Soliman, Felix Sutton, Roman Lim, Marco Zimmerling, Bernhard Buchli, David Hasenfratz, Federico Ferrari, Matthias Keller, Matthias Woehrle, Andreas Meier, Clemens Moser, Lennart Meier, Philipp Blum