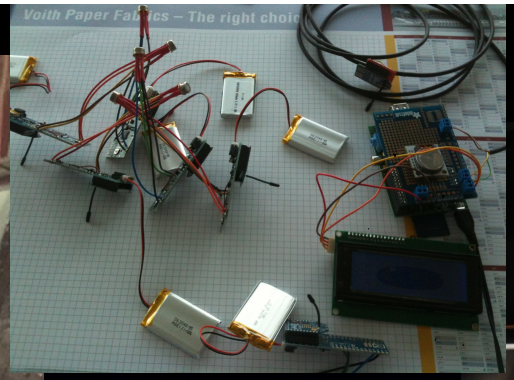
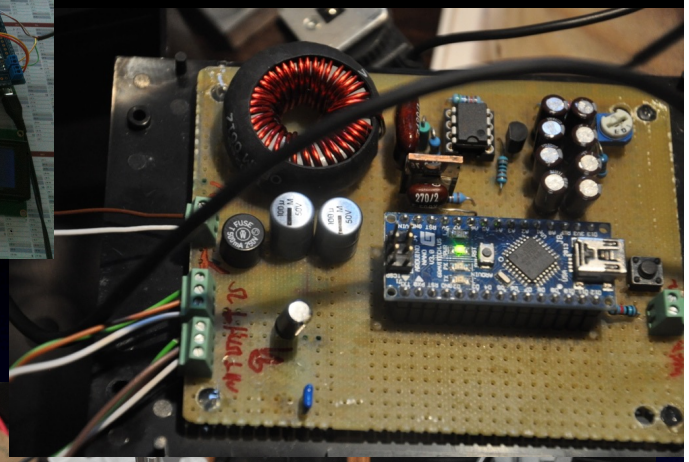
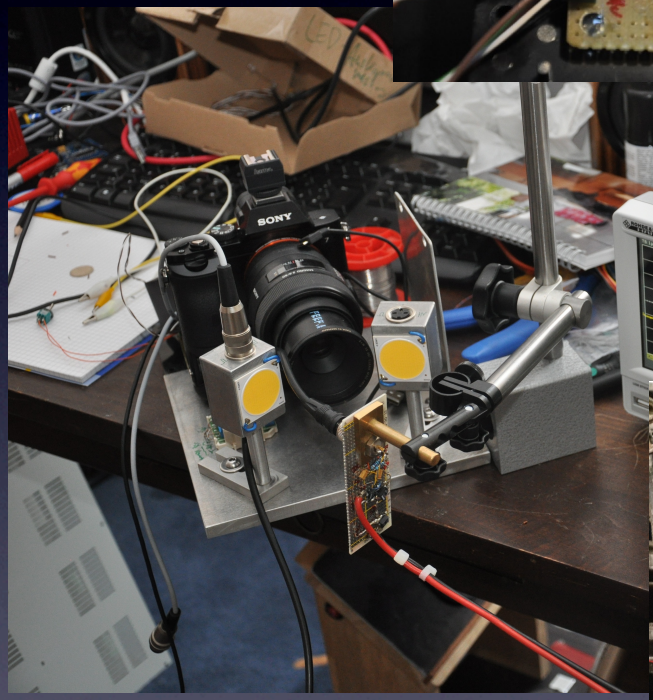
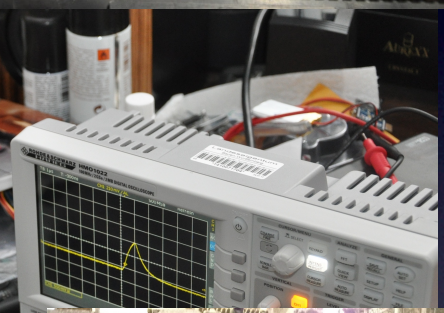
The Case for a Functional Internet of Things

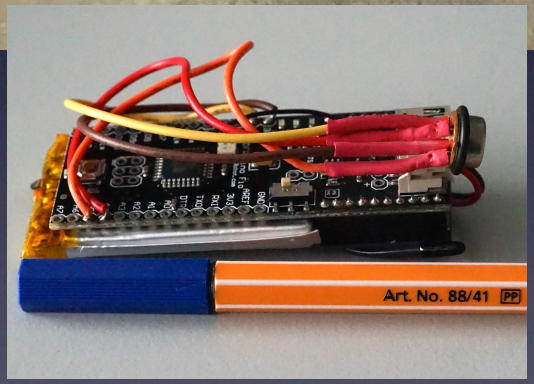
Till Hänisch, Baden Württemberg State University, www.tillh.de

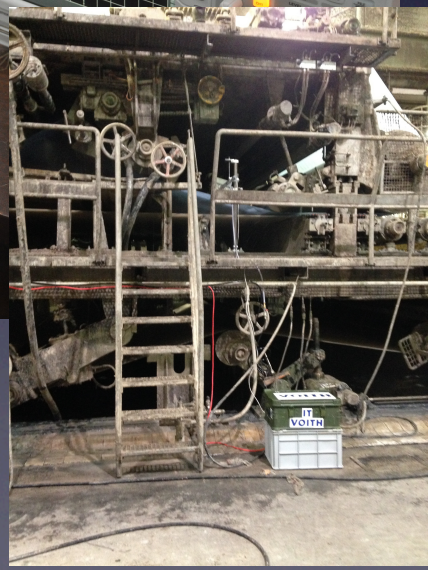








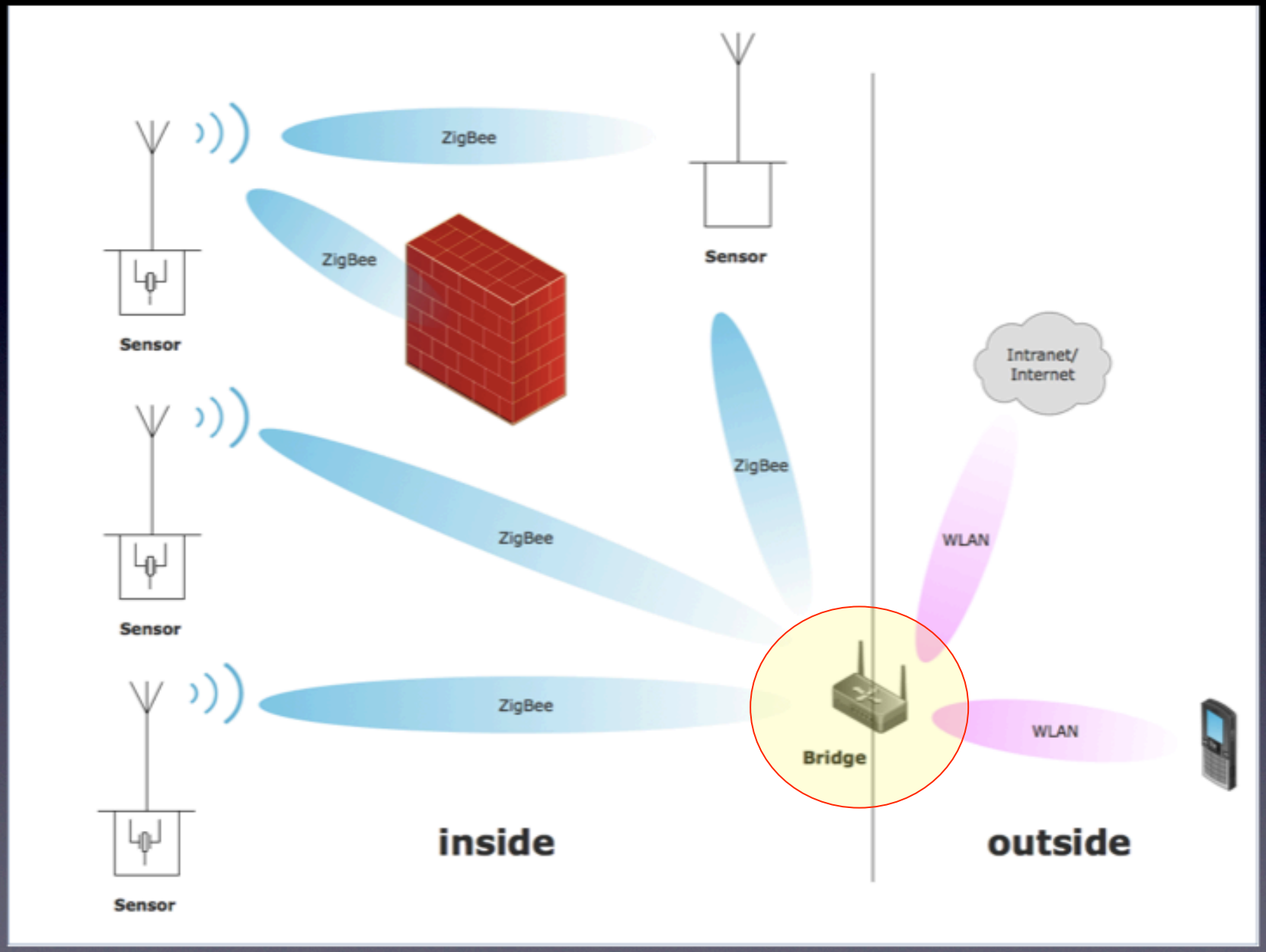




The Case for a Functional Internet of Things

Till Hänisch, Baden Württemberg State University, www.tillh.de

Overview of the system



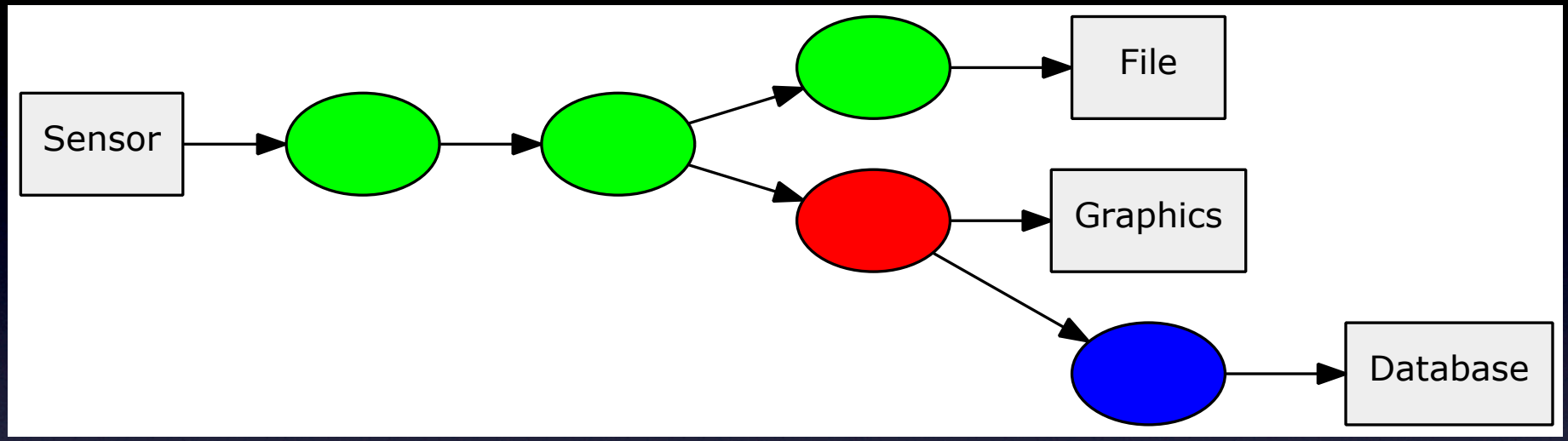
Messages from the sensor nodes

```
12 DATA 37855 15 332912 612740 730486 105 753 END
13 DATA 37856 15 332912 612920 730326 41 237 END
12 DATA 37857 15 332912 613006 730353 109 255 END
13 DATA 37858 15 332912 613200 730166 0 445 END
12 DATA 37859 15 332912 613220 729953 41 498 END
13 DATA 37860 15 332912 613333 729286 89 333 END
12 DATA 37861 15 332912 613553 728700 51 523 END
13 DATA 37862 15 332912 613680 728860 101 394 END
12 DATA 37863 15 332912 613840 728646 50 853 END
13 DATA 37864 15 332912 613960 727313 105 366 END
12 DATA 37865 15 332912 614106 726913 70 450 END
```

Too many bugs !

- Why ?
  - Mostly special case/error handling
  - Many special cases
  - Complex code
  - Hard to test
- Solution
  - ???
  - Make it easier
  - But how

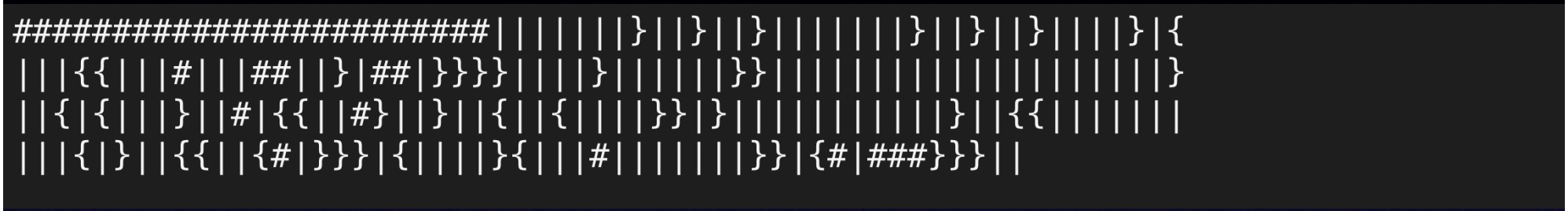
Stream of messages



Processing data from sensor nodes =
transforming, duplicating, storing messages =
applying functions to messages =
functional programming

Shorter, simpler

ruby version



elixir version

Analogue to: Ward Cunningham, Signature Survey: A Method for Browsing Unfamiliar Code

\# = comment

| = non blank line

{/} = open/close block

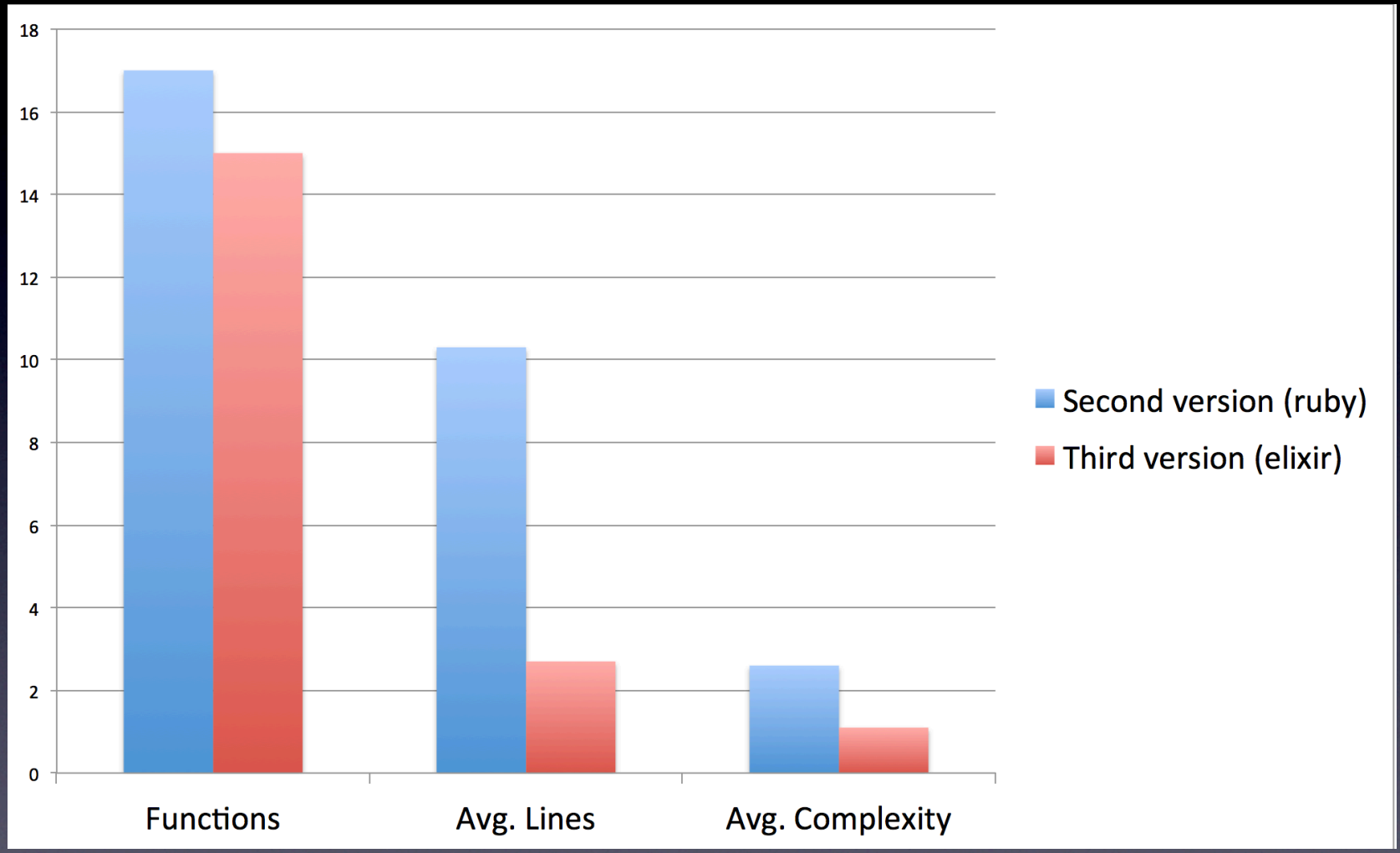
Typical functions

```
    def text_form(value,age) when age > 5, do: "---/---"

    def text_form(value,age) do
      format_display(value)
    end

    def show_data(values) do
      Enum.map(values,fn(x) -> IO.puts(text_form(x,age(x))) end)
    end
```

Most complex function

```
def process_file(input_file,current_values) do
  task = Task.async(fn -> IO.read(input_file, :line) end)
  try do
    row = Task.await(task,5000)
    if (row != :eof) do
      new_values = process_line(String.split(String.rstrip(row), " "),current_values)
    process_file(input_file,new_values)
      end
  catch
    :exit, _ -> IO.puts "timeout"
    process_file(input_file,current_values)
  end
end
```

Elixir (functional) version compared to ruby (OO)



Elixir version is better in any metric, even size (50%)

No bugs found till today

Key findings ….

- Shorter, easier code by using a functional language
  - lower complexity
  - fewer bugs
- Elixir is production ready
- Functional programming might be a way to develop reliable IoT applications

Why is (low) complexity important

- Complexity leads to bugs
- Bugs reduce security
  - Availability
  - Integrity/Confidentiality
- Internet of Things is
  - embedded, hard to patch
  - open, easy to access

Zero defect software is needed for the IoT
No general solution ! PROBLEM !