# IoT Trends and Technologies

iPLON GmbH 2016

iPLON® The Infranet Company

KFW DEG

# What is IoT, what is it good for?

There is a lot of fuzz around “IoT”: “Internet of Things”
wikipedia definition:

> “The **Internet of Things** (**IoT**) is the network of physical objects, devices, vehicles, buildings and other items which are embedded with electronics, software, sensors, and network connectivity, which enables these objects to collect and exchange data”

Lots of **funny things** like smart watches, networked cars, door locks, even “smart” light bulbs (“Hue”).
Applications in the **energy sector**: Smart Meters, Room Controllers, Home Battery Systems.
Also in **PV plants** pervasive computing down to panel scale has its own challenges → IoT technologies
The biggest challenge is the sheer **number** of connected, but independently acting devices.
Despite decentralization of the logic,
monitoring/supervision and control still needs to end up at a **central point**.
The bigger the controlled domains, the higher the expectations to the mon/con **system quality**.

**See some numbers on next slide...**

# Some Numbers from PV Monitoring

10 GW → 50 M panels → 1 lakh SMUs → 10,000 inverters → big data !!!

Monitoring System's purpose is to catch these data for evaluation.

→ Never loose them!   → Data is/are sacred.

- buffered data transport
- redundant data storage

Which up-to-date off-the shelf technologies can be used?

**More numbers on next slide ...**

iPLON®
The Infranet Company

KFW DEG

# Renewable Energy Plant Monitoring is Big Data!

e.g. PV Plant, 50 MW:
average ~ 1 monitored Datapoint / kW
Loginterval typ. 1 minute
20 plants → 20 x 50000 x 1440 x 30
→ > 40 billion (1E9) samples *per month*

Some attributes of the samples:

- **timestamp**
- **value**
- **quality/validity**
- *customer*
- *plant*
- *block/room/station*
- *device*
- *field*
- *unit*
- *sample interval .....*

→ stuff into specialized timeseries database !!

additional requirements:

- scalability → distr. storage, cloud
- high availability → clustered solution
- realtime downsampling
- disk space efficiency

**„Big Data“ is challenging,**
**so we need to dig deeper .....**

# From Scada to Cloud based Distributed Datacenters

- starts small: locally redundant scada backend systems
- grows quickly: synchronize part of plant datapoints with enterprise plant management system
- aims high: enterprise- / regional- / state- / global- level data integration (clusters!)

The **CAP theorem** (*Brewer 1998 / 2012*), is about the conflict of:
*Consistency* (all nodes see the same data at the same time)
*Availability* (a guarantee that every request receives a response about whether it succeeded or failed)
*Partition tolerance* (the system continues to operate despite arbitrary partitioning due to network failures)

- keep data **c**onsistent over geographically separate locations
- keep backends **a**vailable 24/7
- tolerate temporarily **p**artitioned systems, i.e. resynchronize consistently after communication losses

only 2 out of three can be 100% fulfilled, and even that is **hard to implement**

**→ We need help from the community …….**

iPLON® The Infranet Company

KFW DEG

# Successful Technologies go the Open Source way

OS helps for **SW quality improvement:**

The Power of “Crowd” development is that **1000 eyes are looking**.

“Standing on the **Shoulders of Giants**”,
possible/affordable only by OS licensing protecting your I.P.

Benefit from investments of big/global players like Google, IBM, ...

International Scale **Teamwork** through github, sourceforge, etc.

Realize maximum code + architecture **transparency**

**e.g. Docker success story as example on next page ……**

iPLON® The Infranet Company

KFW DEG

# Short Digression on the DevOps challenge

Application Configuration / Deployment should be treated like Development, i.e. Version Control, Workflow, QualityControl, ...
Development and Operation of Highly Complex Datacenter SW can/should not be completely separated;

Need for Technologies to

- provide each appication component with its own specially suited containerized runtime environment and
- make the testing of the installation process easier, quicker and repeatable
- lock applications from depending on  each other's runtime environment

Solution: "Dockerization":

- user space virtualization, containers use kernel functions, kernel locks them from each other
- incremental/overlayed filesystems with step-wise "commit" (and replay) of the individual installation steps

Challenge:
Find a good compromise on the way between "monolith" (all apps in 1 container) and "microservices" (1 appl.=1 container)

Big winner is Docker (some helper apps around LXC=LinuxContainers), the biggest IT success story ever,
its success forced Microsoft to implement  a Linux Kernel inside Microsoft Azure Cloud Solution :-)

**But now back to the IoT challenges, let's find a timeseries database to save our stuff in ......**

# In search of a IoT timeseries database

Some selected Requirements:

- Easily Distributable/Clustered
- Effective Disk Space usage
- Failure tolerant
- Query Speed independent of Database Size

Our candidate this young OS project:
“influxdb”,
they went through hard times last year,
now is stable again,
popularity now great:

Not yet reached V. 1.0, but highly promising!

**And what's behind the scenes? …..**

## DB-Engines Ranking - Trend of Time Series DBMS Popularity

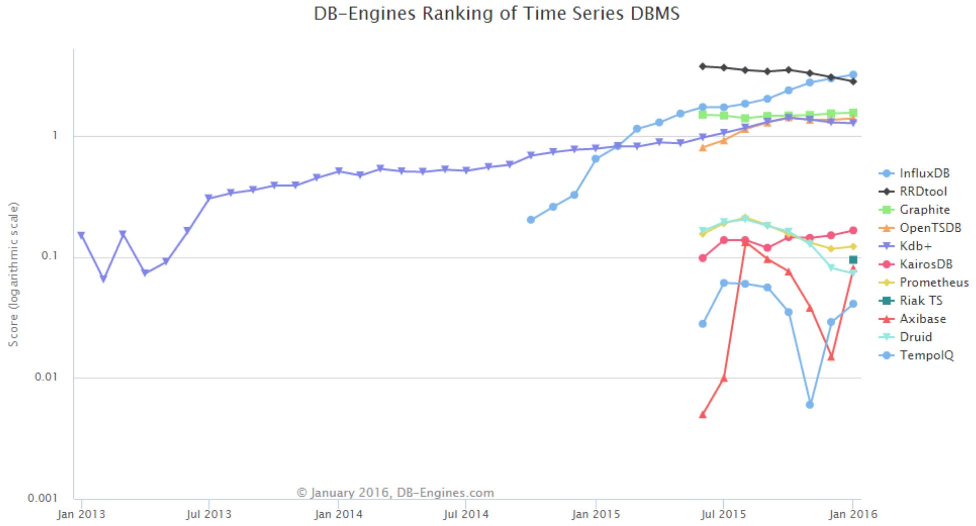
The DB-Engines Ranking ranks database management systems according to their popularity.

This is a partial trend diagram of the complete ranking showing only time Series DBMS.

Read more about the method of calculating the scores.

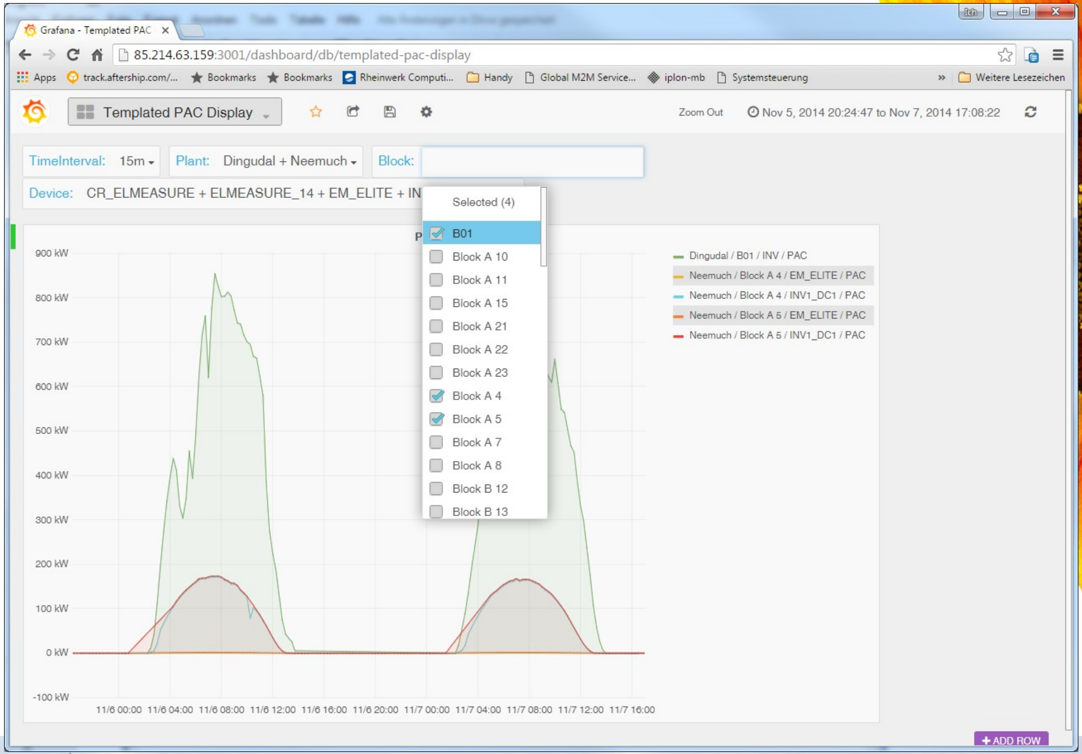
| Rank | Trend | System | Score | Change |
|---|---|---|---|---|
| 1 | | Oracle | 1560 | + 27 |
| 2 | ↑ | MySQL | 1342 | + 47 |
| 3 | ↓ | SQL Server | 1278 | - 40 |
| 4 | | PostgreSQL | 174 | - 3 |
| 5 | | MS Access | 161 | - 8 |
| 6 | | DB2 | 155 | - 4 |

ranking table
January 2016



# influxdb concepts

- **Clustering** is very simple, implements “RAFT” consensus algorithm to meet CAP req
- can be used to increase
  - disk space
  - availability
  - query speed.
- with the new “tsm” storage engine it is **highly disc space effective**:
  2 TB mysql (~30 GiB compressed CSV files) boiled down to 150 GiB influxdb
  currently running with
  ~ 1 Million time series x 200000 entries →
  200 Billion samples !!
- separate **compression** of
  - time stamp differences
  - field value differences / XORs, etc.
  metadata are kept and searched separately
  unlimited number of fields per row
- **another OS example on next slide ...**

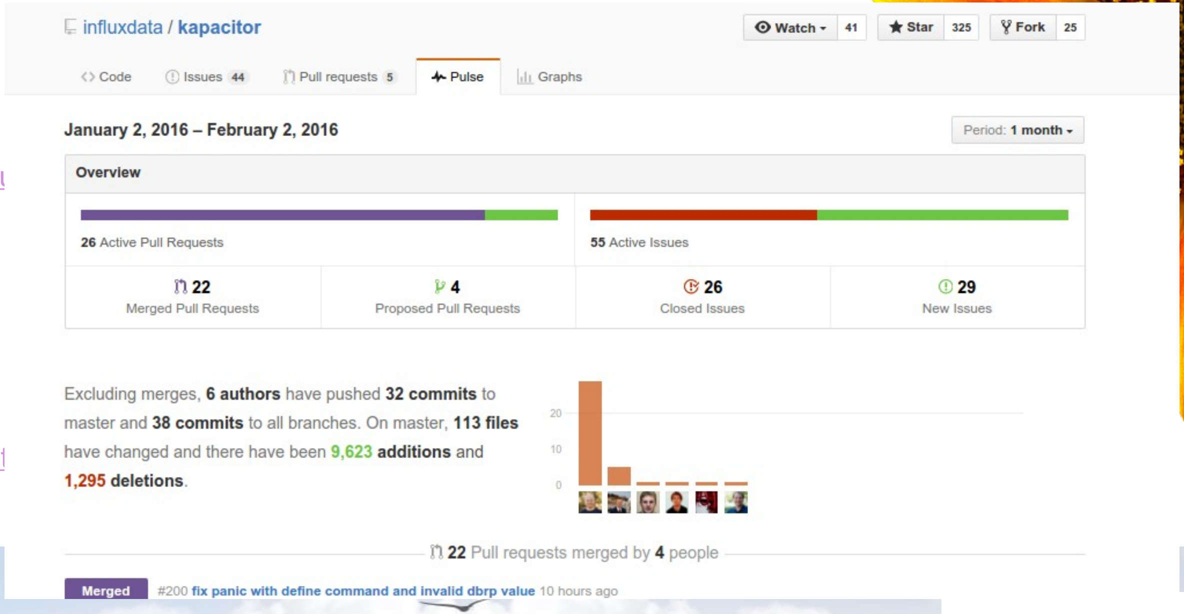



iPLON® The Infranet Company

KFW DEG

# OS development process, e.g. “kapacitor”

- “kapacitor”: complex event processor for
  - alerting based on complex and/or dynamic criteria
  - data processing
  - replay of situations
  - anomaly detection
- ongoing development by “influxdata”,
  - happens completely in public
  - ideas/proposal, bugs, changes are discussed on https://github.com/influ code changes are linked to the discussions, so motivation for code becomes transparent/traceable
  - active watchers: 5 from influxdata, another 40 from other companies;
  - 25 forks, results are fed back and discussed through “pull requests” https://github.com/influxdata/kapacit




**Conclusion follows ….**

# Conclusion / Summary

The **Open Source** approach allows us
* to borrow from and
* to take part in
the “cloud” infrastructure world which is advancing at high pace.

(Only?) using these opportunities enables **SME**s to accept the tremendous challenge of **IoT**.

**We are prepared !**

Thank you for your attention. .... Questions?