Eclipse Cyclone

"The Data is the Network"

..The source of **DDS** fuels Eclipse ..

eclipsecon Europe 2017

ADLINK
® Leading **EDGE COMPUTING**

Challenging Opportunity ..

.. What messaging protocol(s) do you use for your IoT solution?

Protocol	Percentage
HTTP	60,1%
MQTT	54,7%
CoAP	26,7%
In-house / proprietary	18,4%
HTTP/2	16,8%
AMQP	15,0%
XMPP	10,3%
Other	7,1%
Don't know	7,1%
Proprietary vendor protocol (specify below)	4,9%
DDS	4,0%
None	3,6%

IoT Developer Survey 2017 - Copyright Eclipse Foundation, Inc.
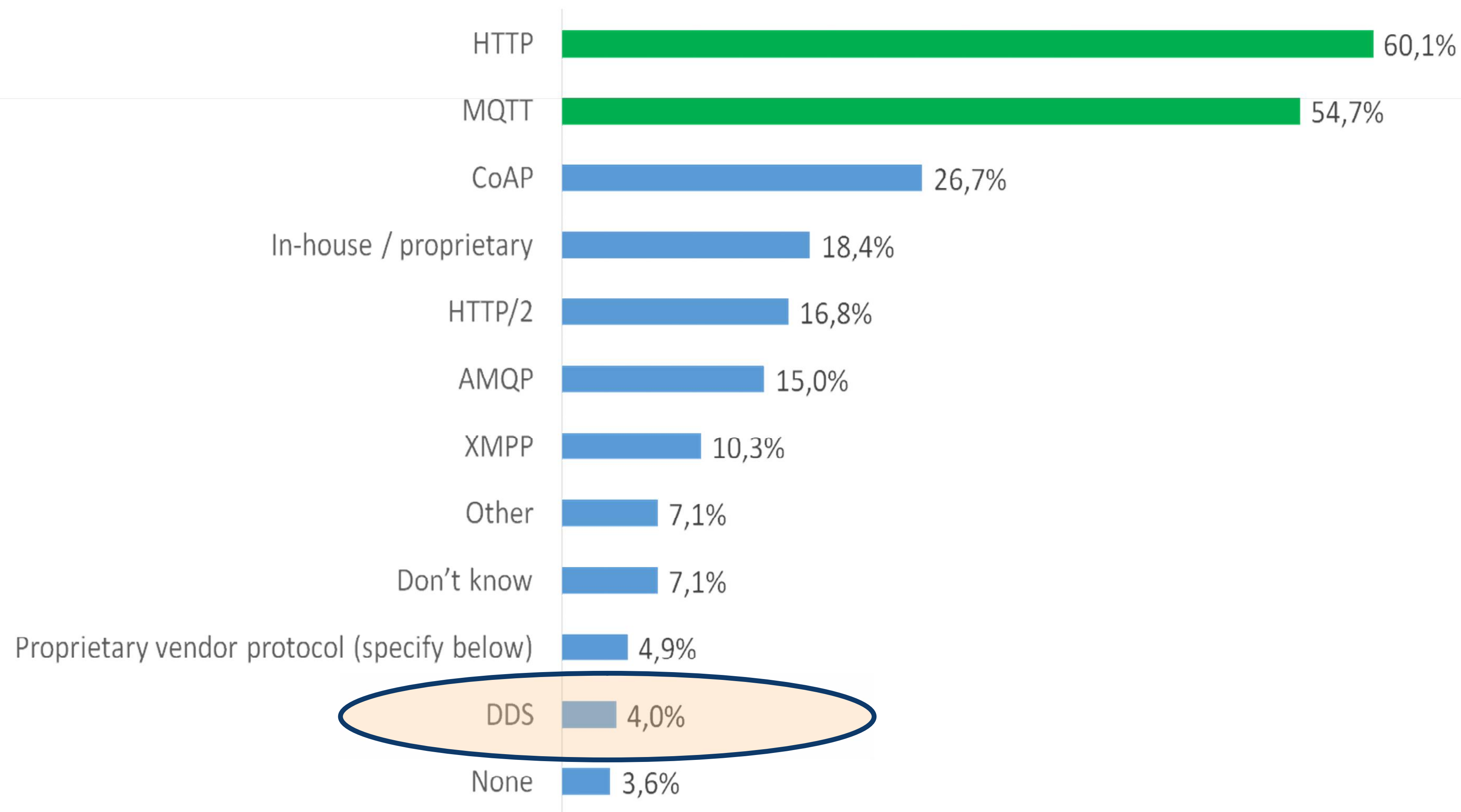
The new kids on the block ..

- We **ARE** the source of DDS
  (*we invented it, driven the OMG-DDS standard and witnessed its success in becoming
  a mandated standard in several mission-critical environments*)

- We will **OPENSOURCE** our complete DDS product line as "**Eclipse Cyclone**"
  (*both existing/proven codebase as well upcoming nimble/agile evolutions*)

- We will **FUEL** the Eclipse IoT community with something **cool**, new and different
  (*that has proven applicability in data-centric environments such as IIoT*)

- We **TRULY BELIEVE** that DDS deserves a thriving **ecosystem**
  (*where both adoption and participation will be its measure of true success*)

- we **CONTINUE** to provide long-term commercial **support**
  based on guaranteed quality of stable-versions of the **Eclipse Cyclone** codebase
  (*under our ADLINK "Vortex" family brand*)

ADLINK

**THE source of DDS
fuels Eclipse**

Leading **EDGE COMPUTING**

www.adlinktech.com

Eclipse Cyclone: "*The Data is the Network*"

- **What is it ?**
  - Full implementation of the **OMG** **DDS** specification (both DCPS/API and DDSI/wire-protocol)
  - Including a **proven/stable V1** and **nimble/agile V2** version with lots of cool innovation ideas & plans

- **What does it do ?**
  - Offers reliable/robust and real-time **data sharing** in (often mission/business-critical) distributed IoT systems
  - Reduces system **complexity** by promoting **autonomous** data-centric components (microservices)

- **Why is it different ?**
  - It is uniquely '*data-centric*' where data has both structure & context and populates a virtual '*global dataspace*'
  - Where fine-grained **QoS**-annotations of the data(-model) allow the middleware to manage **availability & distribution**

- **Why does it matter ?**
  - It allows applications that are decoupled in space, time & frequency to **fully concentrate** on their business logic
  - And where **non-functional properties** (reliability, robustness, timeliness, availability) are maintained by the middleware

So summarizing, it's an elegant/simple solution that allows to build hugely complex systems out of simple app's