OpenSplice DDS

DDS for SCADA

Erik Boasson
Senior Engineer
PrismTech
erik.boasson@prismtech.com

The mismatch

What is DDS?

- Primarily, the DDS 1.2 standard
  - a programming model
  - an interface specification
- The standard operates at the level of an implementation
  - consequently, its applicability is a subset of that of the programming model

What is DDS?

- “DDS is not a good fit”
  - refers to the implementation-level specification
- “as it stands today”
  - standards can be extended and amended

What is SCADA?

- Supervisory Control and Data Acquisition
- In practice covers such things as
  - system monitoring
  - closed-loop control systems
  - operator interface to a system

OpenSplice DDS

What is SCADA?

- Supervisory Control and Data Acquisition
- In practice covers such things as
  - system monitoring
  - **closed-loop control systems**
  - operator interface to a system

What is SCADA?

- Feedback loop
  - thousands to millions of sensors and actuators
  - multi-layered control system
- Other aspects we ignore here
  - operator interfaces
  - off-line optimisation
  - post-mortem analysis
  - …

What is SCADA?

- Control blocks
  - control blocks often a given
  - "only" need to parametrize them
- Interconnections
  - it matters which specific sensor you use
  - fairly static

OpenSplice DDS

DDS

- □ Typically viewed as publish-subscribe
- □ From the OMG DDS Portal:
  - □ DDS is the first open international middleware standard directly addressing publish-subscribe communications for real-time and embedded systems.
    DDS introduces a virtual Global Data Space where applications can share information by simply reading and writing data-objects addressed by means of an application-defined Topic and a key.

DDS

- It really is the other way around:
  - DDS introduces a Global Data Space
  - pub-sub is a possible implementation

DDS and SCADA

- System state as a shared data space
  - containing measurement and control values
- Subscribe to individual measurements, &c.
  - topic per measurement, &c.
- Problem solved

DDS and SCADA

- System state as a shared data space
  - containing measurement and control values
- Subscribe to individual measurements, &c.
  - topic per measurement, &c.
- Problem solved — well, not quite!

Why not?

- □ DDS doesn't scale nicely to millions of topics
  - □ or readers and writers for that matter

  - □ resource consumption
  - □ discovery times
  - □ traffic overhead

Alternative mappings

- No *requirement* to have that many topics
- Must avoid fitting problems to solutions

If not this, then what?

A step back

- What can we throw out profitably?
  - multitude of QoS settings
  - detailed metadata
- Cost incurred by these
  - complexity in discovery
  - increased footprint
  - slower data handling
  - higher network load
  - …

A step back

- Assume
  - processing equidistantly sampled signals
  - control loop is hard real-time
  - network is highly reliable
  - procedure for dealing with lost samples
- Then
  - only latest values need to be kept around

A step back

- Data space characteristics
  - millions of "topics"
  - one (or a handful of) data type(s)
  - a small selection of QoSs
- Control block naming
  - GUIDs will do in practice
- Operations
  - read & write

OpenSplice DDS

A step back

- Domain-specific DDS variant
- Self-evident that you can implement this
  - with a small footprint
  - including dynamic discovery
- Obviously not covering all aspects

Desiderata

- □ Integrated with rest of DDS
- □ Leverage DDS features
- □ Simple interface

Approach

- □ Transient data for subscriptions
- □ Dynamically mapping data to partitions
- □ One topic for data

OpenSplice DDS

Approach

- Partitions in a small system
  - one partition per node
  - a common partition
- Subscriptions in two partitions
  - in its own & the common partition
- Publishing partition chosen dynamically
  - one subscribing node: that node's partition
  - multiple subscribing nodes: common partition

Approach

- Experiments show very load overhead
  - negligible CPU load
  - low memory overhead
  - ~10% network overhead
- Special care taken to minimise cost of updating values for which no subscriber exists
  - this is, after all, one of the real promises of DDS
- Potential for integrating into DDS proper

OpenSplice DDS

Conclusion

Conclusion

- DDS can be used as a foundation for domain-specific data spaces
- The large feature set of DDS can be a problem rather than a solution
- It is important to distinguish between the programming model & the implementation