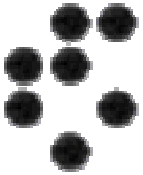


Jozef Stefan Institute

cosylab
CONTROL SYSTEM LABORATORY

Design considerations for framework integration

Aleš Pucelj (ales.pucelj@cosylab.com) , G. Tkacik, R. Šabjan

PCaPAC 2002

Framework Performance

- What is the added overhead of a framework ?

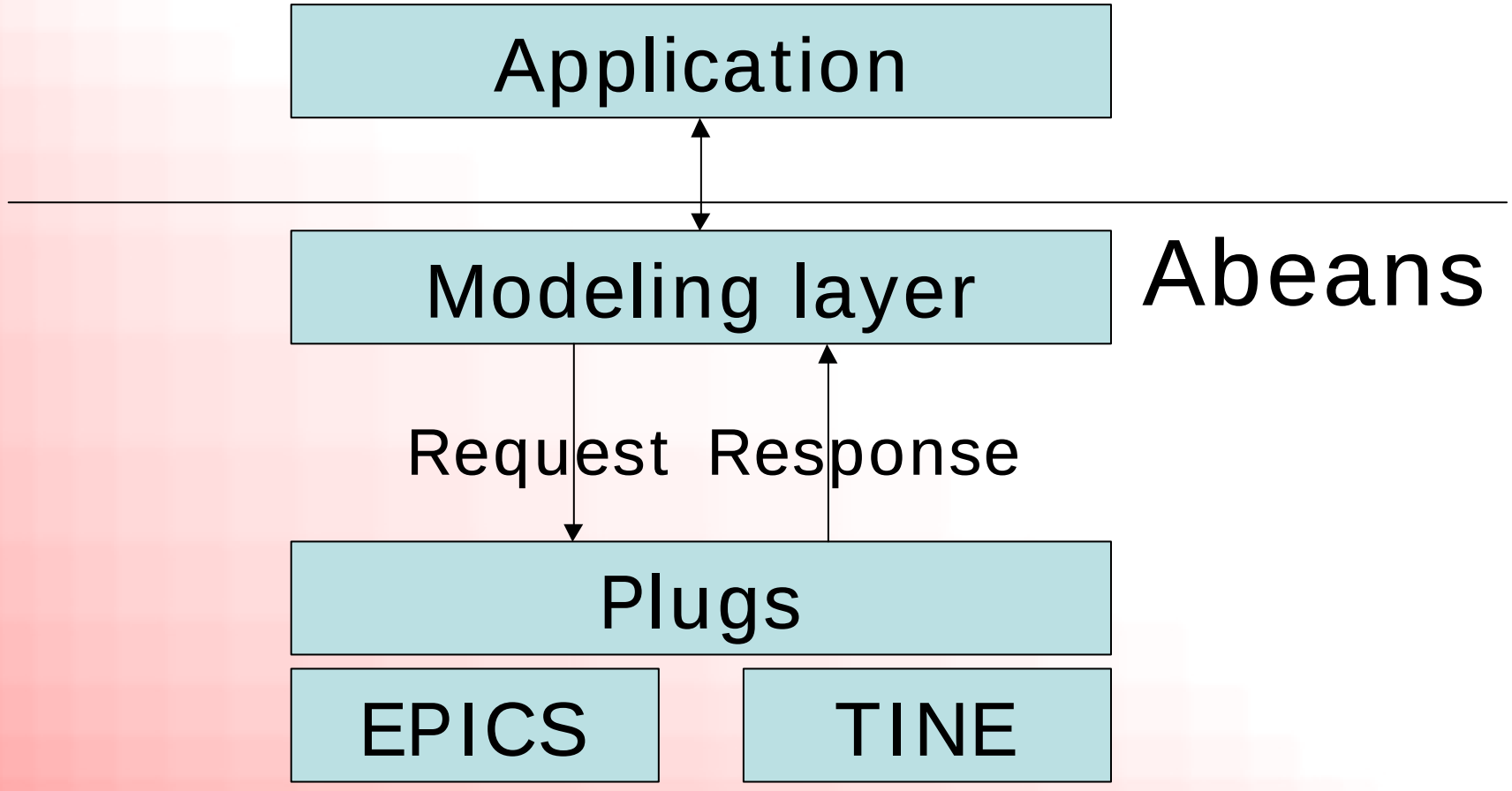
Control System Framework Design

- Closed architecture
- Open architecture

Requirements

- Simple to use
- Extensible
- Portable
- Minimal overhead

Abeans Framework Overview



Abeans Framework Communication

- Request
  - Resource identification
  - Data type
  - Operation
- Response(s)
  - Value
  - Error code
  - Timestamp

Abeans Namespace
Sample naming

- General structure

  [scheme:][//authority][path][?query][#fragment]

- EPICS

  abeans-EPICS:///PBEND_M_01_current?get

- TINE

  abeans-TINE://ns.desy.de/DESY/BENDS/PBEND_M_01/current?get

Abeans Namespace URI Definition

- Open standard (RFC 2396)
- Implementations already provided
- Is sufficiently rich for our purpose

Abeans Namespace URI handling

- All resources are identified by URIs
- URI creation hidden from application developer
- Customizable for specific implementation

Plugs

- Only link between Abeans and remote data
- Most critical for performance
- Reliable

Plug Implementation EPICS

- Realized using JCA (Boucher)
  - One process variable (PV) per field
  - All records accessed by fields
  - Different datatypes
  - CA designed for single-threaded access
  - No namespace
  - Not all Abeans features supported
  - Native C++ library

Plug implementation
EPICS data types

- Different in types
  - Abeans: Double, Long, String, Object
  - EPICS: Double, Float, Int, Byte, Short, Enum, ...
- Conversion made during each call
- All types can be accessed from Abeans
- Native types used for CA

Plug Implementation Details

- One connection - one PV
- Connections are cached
- Automatic establishing and closing of connections
- Hash table implementation still efficient for large number of PVs

Plug Implementation TINE

- Realized using Java implementation of TINE protocol
- No need for type conversions
- Persistent connections not directly available
- Some data access not directly supported

Performance Issues
Memory overhead

- Most overhead is imposed by Java and OOP
- Java implementation (TINE) preferred over native (EPICS)
- Mainly from type conversion

Performance Issues Communication

- Most overhead from Abeans request handling
- Negligible compared to network latency
- Slow-time control

Conclusions

- Framework overhead is less than expected
- Existing protocol implementations are sufficient
- Open architecture works