JavaIOC: Overview

Marty Kraimer
EPICS Collaboration Meeting
DESY April 23-27 2007

Implemented

- PV (Process Variable) Database
- XML Parsers: DBD and Record Instance
- Record Processing
- Record Scanning: Periodic and Event
- Record Monitoring
- Channel Access – Local only
- Support: Beginning of standard set
- SWTSHELL: GUI iocshell

Not Implemented

- Remote Channel Access
  - JavaIOC client <-> JavaIOC server
  - CSS <-> JavaIOC server
  - ? JavaIOC client <-> V3 server
  - ? V3 client <-> JavaIOC server
- More complete set of standard support
- JavaIOC version of ASYN.
- Lots to do!!

Other Desirable Components

- VDCT: Visual Database Configuration Tool
- Access security
- Error logging
- Put logging
- CSS <-> JavaIOC Server
- ? Other clients: Display Manager, Archiver, etc.
- ???
- Lots and lots to do!!

PV (Process Variable) Database

- Field types
  - Primitive + String : all Java primitives except char
    - boolean, byte, short, int, long, float, double, string
  - enum: set of choices + index to select choice
  - menu: enum with immutable choices
  - Link: no data but configuration structure
  - array: elementType can be any type
  - structure: fields can be any type

PV Introspection

- Introspection Interfaces, i.e. no data
  - Field: primitive + string
  - Enum: extends Field
  - Menu: extends Enum
  - Array: extends Field
  - Structure: extends Field
- FieldFactory: Sufficient for all needs?
- Field can have properties
  - Related fields, e.g. status, severity, timeStamp

PV Data

- PVField: Base for data interfaces
  - PVBoolean,...,PVStructure
  - PVArray: Base for array data interfaces
    - PVBooleanArray,...,PVArrayArray
- PVDataFactory
  - Default implementation – used by DB and CD.
  - Any field can be replaced. Often useful.
  - Base classes are provided.

XML Parsers

- Include and Macro Substitution
- DBD - Database Definitions
  - menu
  - structure
  - recordType
  - support
  - linkSupport
- DB - Record Instance

Include – Macro Substitution

```
<include addPath = "path" removePath = "path"
    href = "filename" />

<substitute from = "fromString" to = "toString"
   fromTo = "from=to,from=to,..."/>
```

Macro Substitution

- Macro substitution can be performed on:
  - xml attribute value in any element definition
    - <record name = "ai${recordExtension}Record" ...
  - content of any xml element definition.
    - <pvname>${pvname}</pvname>

menu definition

```
<menu name = "menuAlarmSevr">
        <choice>none</choice>
        <choice>minor</choice>
        <choice>major</choice>
        <choice>invalid</choice>
</menu>
```

structure definition

```
<structure name = "doubleLimit">
        <field name = "low" type = "double" />
        <field name = "high" type = "double" />
</structure>
```

recordType definition

```
<recordType name = "example”
        supportName = “exampleRecord"/>
   <include href = “common.xml” />
   <field name = "value" type = "double">
   <field name = "units" type = "string" />
   <field name = “displayLimit”
        type = “structure” structureName = “doubleLimit”
/>
</recordType>
```

support definition

```
<support name = "linkArray"
   factoryName = "org.epics.ioc.support.LinkArrayFactory" />

<linkSupport name = "processLink"
   configurationStructureName = "processLink"
   factoryName = "org.epics.ioc.support.CALinkFactory" />

<support name = "calcRecord"
  factoryName = "org.epics.ioc.recordSupport.CalcFactory" />
```

record instance definition

```
<record name = "exampleInstance" type = "example" >
   <value>10.0</value>
   <units>volts</units>
   <displayLimit>
      <low>0.0</low>
      <high>10.0</high>
   </displayLimit>
   <scan>
      <scan>periodic</scan>
      <rate>1.0</rate>
   </scan>
</record>
```

PV Naming

- EPICS V3 pvname is <record>.field
- JavaIOC is <record>.name.name. ...
  - name.name. ... because of type structure
- Why “name” instead of “field” ?
  - Could be field, i.e. Traverse structure hierarchy
  - Could be property of a field

Field Property

```
<field name = "value" type = "int" >
    <property name = "status"
        associatedField = "status" />
    <property name = "severity"
        associatedField = "severity" />
    <property name = “timeStamp”
        associatedField = “/timeStamp” />
 </field>
```

Record Processing

- A record instance is basic process unit
  - It is locked during any processing or I/O
  - A record has at most one process requester
    - self scanned records allow multiple requesters
  - RecordProcess is master
  - Support modules implement semantics
  - ANY field can optionally have support
  - Record instance must have support
    - Can be noop

RecordProcess

- RecordProcess: many methods including
  - process(RecordProcessRequestor ...
  - requestProcessCallback(ProcessCallbackRequester ...
    - Call requester with record unlocked
  - ProcessContinue(ProcessContinueRequester ...
    - Call requester with record locked
- RecordProcessRequester
  - RecordProcessResult - Called with record locked
  - RecordProcessComplete - Called with record unlocked

Record Scanning

- Scan
  - Types: passive, periodic, event
  - Priority Uses Java priorities
  - Threads created as required
- Periodic
  - Arbitrary rate: minPeriod,deltaPeriod
- Event
  - Based on eventName
  - Replaces V3 event and I/O Inter

Support

- Support – many methods
  - Life cycle – initialize, start,stop,uninitialize
  - process(SupportProcessRequester requester)
- SupportProcessRequester
  - supportProcessDone(RequestResult result)
- AbstractSupport
  - Implements all methods
  - Is base for all support

Synchronous Support

- Support.process
  - Do what ever is needed and when done call supportProcessRequester.supportProcessDone

Asynchronous Support

- Support.process
  - recordProcess.requestProcessCallback(this);
- Support.processCallback
  - Called with record unlocked
  - Do what ever is needed and when done
  - recordProcess.processContinue(this);
- Support.processContinue
  - Called with record locked
  - Final processing, e.g. alarms
  - supportProcessRequester.supportProcessDone

Record Monitoring

- ANY field of a record instance
- If a structure is monitored
  - Notification of each subfield modification
- Can monitor a record instance
  - Notification of every subfield modification

Channel Access

- Only local is implemented
- CD (ChannelData) provides storage
- Flavors
  - Process
  - Get, CDGet
  - Put, CDPut
  - PutGet
  - Monitor – always uses CD

Channel Access Cont.

- CALinkFactory - Database links
- CDFactory – Channel Data Factory.
- Get/Put CDGet/CDPut
  - have option to process
- Get/Put
  - allow an array to be read/written in chunks
- CDGet/CDPut
  - Use CD for intermediate storage

Standard Support

- Field Support available now
  - scan
  - alarm - severity,message
  - doubleAlarm, intAlarm, booleanAlarm
  - calculation
  - linear conversion (in and out)
  - noop (support and linkSupport)
  - CA Links

Standard Record Support

- doubleRecord
  - Can be record or structure support
  - Supports recordTypes: double, aiDouble, aoDouble
  - Can be embeded in other records, e.g. PowerSupply
- intRecord – like double
- eventRecord – process announces event
- calcRecord – will be V3 calc + sub
- powerSupplyRecord - device example

SWTSHELL

- Currently implements
  - When started it initializes IOC
  - Probe: get/put/process via CA
  - LoadDatabase: on-line add
  - IntrospectDatabase: DBD and record instances
  - Monitor
- Started via
  - XMLToDataBase \
    -dbd xxxDBDfile.xml -db xxxDBfile.xml -swtshell

SWTSHELL Future

- Gui should be remote
  - Use remote CA for communication?
- More diagnostic info
  - Record state, etc, etc.
- CSS not SWT
- Not my main interest
  - Complaints imply volunteering to help :-)