THE COMMON DEVICE INTERFACE 2.0

Philip Duval and Honggong Wu, DESY MCS, Hamburg, Germany

Jaka Bobnar, Cosylab, Ljubljana, Slovenia

Contents

- CDI and TINE (brief review)
- How CDI works (brief review)
- Features new to release 2.0
- CDI Editor
- CDI Deployer

A brief history of TINE

(Three-fold Integrated Networking Environment)

- CERN Isolde (1989) – DOS/Windows based
- Equipment Modules (early object-oriented)
- Continuous development @ DESY since 1992
- Unix/VMS ports ~ 1993/4
- VxWorks ~1995
- Produce-Consume; Publish-Subscribe ~1995/6
- Publish/Consume (multicast, video) ~ 1999
- Java ~ 1998/9
- …
- .NET, LabView, MatLab, Lazarus, Python, IDL, …

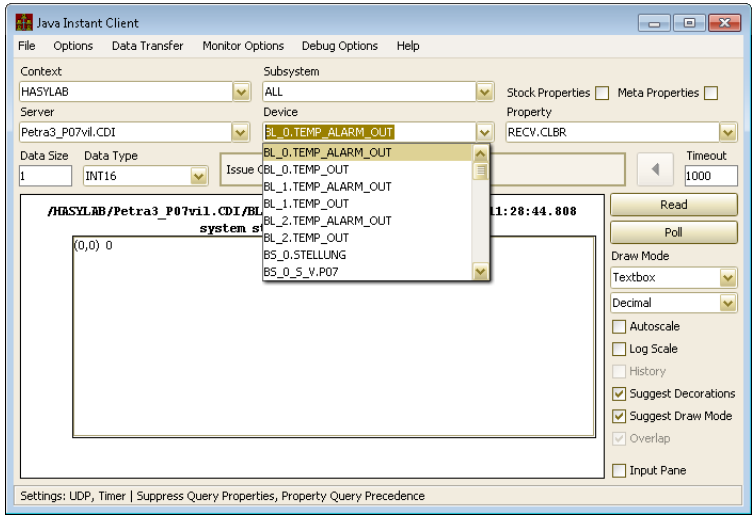
How CDI works

- *Hardware access* thru hardware-specific bus-plugs.
- CDI reads a *manifest* telling it which bus-plug libraries to load
  - 'SEDAC', 'CANPeak', 'EtherCat', 'RS232', etc.
- When a bus-plug library loads it …
  - Registers bus access routines
    - read, write, monitor, scan, …
- CDI reads an *address database* giving
  - Device name, bus plug, bus address, etc.
- CDI offers *common API* independent of hardware bus.

CDI and TINE

- CDI and TINE are loosely coupled
  - You can use CDI without TINE, but not without the TINE library.
  - TINE has no dependencies on CDI, but does have hooks for CDI.

- A TINE server can use embedded CDI services for hardware access !
- Generic CDI hardware server
  - Export CDI address database as a control system service.
  - Automatic TINE server !



CDI Hardware server

PCaPAC 2006 (JLAB)

- Vintage release 1.0:
  - *Passive* (ready and waiting for duty)
  - *No device control intelligence* beyond that configured in the database.
    - Masks, calibration, …
  - *Use as a device server* as is or …
  - *Write a device (middle layer) server* which targets the CDI hardware server.

CDI Hardware server

- Vintage Release 2.0
  - *Address Templates*
    - Extended properties
  - *Asynchronous Listeners*
    - Server no longer passive
  - *Scheduling*
    - Notifying clients upon data change
  - *Local histories*
  - *Remote database access*

But first: A short digression on property servers before we cover these …

CDI Server is a Property Server

- Brief Review of Control System Paradigms:
  - *Database model*: EPICS, VISTA, …
    - Control points are given names and regarded as elements in a database
  - *Device Server model*: DOOCS, TANGO, and TINE (and STARS, !CHAOS and others)
    - Control points are instances of (named) equipment and are accessed via properties (methods, attributes).
      - Locate a hardware instance and ask it for its properties.

CDI Server is a Property Server

- 3rd Paradigm:
  - *Property Server:* TINE
  - Services and information provided by some control system process are named properties
  - Properties can have keywords (possibly device names).
- *Same naming hierarchy as device server*
  - Locate a property and ask it for its keywords.

DOOCS and TANGO hate this !

CDI Server is a Property Server

- Properties:
  - *Bus properties:* "RECV", "SEND", "RECV.CLBR", …
    - Keywords are CDI devices
  - *Info properties:* "TEMPLATE", "INSTANCES", …
    - Keywords are template names
  - *Database properties:* "CDIADDR.DB", …
    - Keywords are database operations
  - *Extended properties* …
    - Keywords are CDI template instances
  - …

What's an 'extended property' ?

CDI Address Database

- Example CDI address database

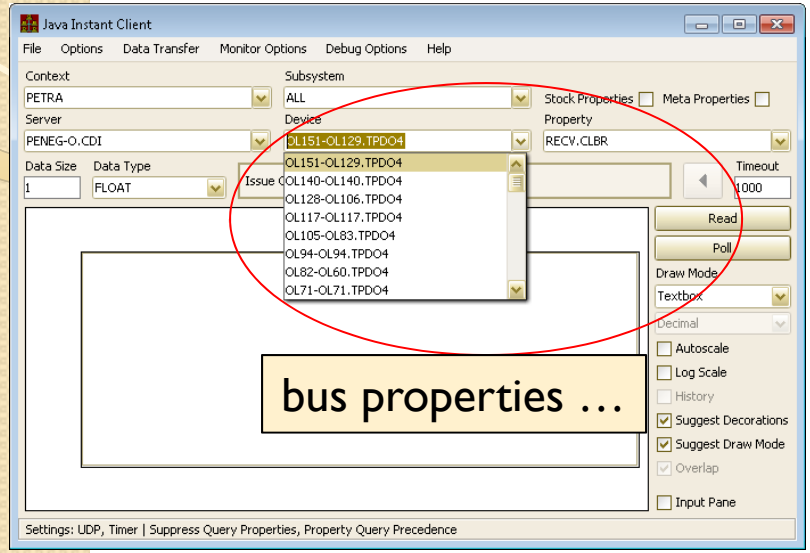
NAME	BUS	LINE	ADDRESS_BASE	ADDRESS_PARAMETERS	FORM/	ACCESS
SEDAC:BLMs	FIELDBUS	1	0	0	short	RD
BLM:hiWord	TEMPLATE	1	0	2	short	RD
BLM:loWord	TEMPLATE	1	0	1	short	RD
BLM:Mode	TEMPLATE	1	0	0:05	short	WR
BLM:Reset	TEMPLATE	1	0	0:03	short	WR
BLM:PowerClr	TEMPLATE	1	0	0:07	short	WR
BLM:Time	TEMPLATE	1	0	0:06	short	WR
BLM:PreScaler	TEMPLATE	1	0	0:04	short	WR
PU01I	SEDAC	1	2.8	<BLM>	short	
PU01O	SEDAC	1	2.8	<BLM>	short	
PU02I	SEDAC	1	2.8	<BLM>	short	
PU02I_I	SEDAC	1	1.8	<BLM>	short	
PU03O	SEDAC	1	2.8	<BLM>	short	
PU03O_I	SEDAC	1	1.8	<BLM>	short	

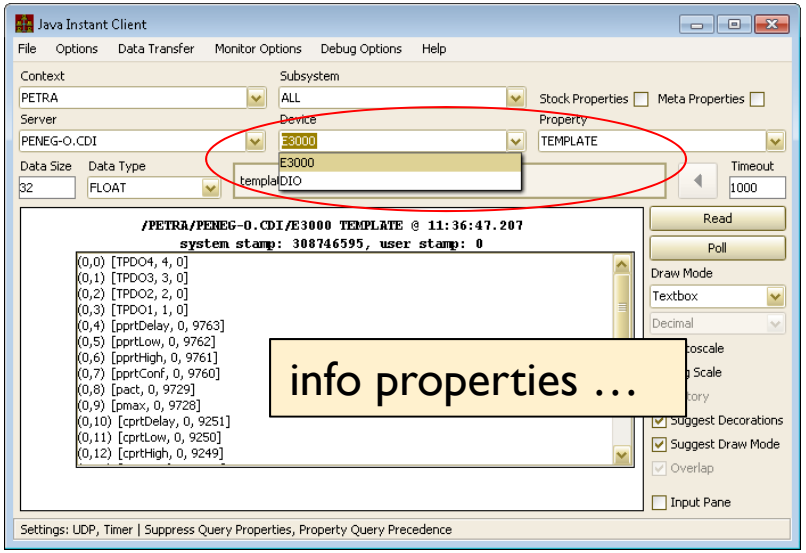
.CSV Spreadsheet

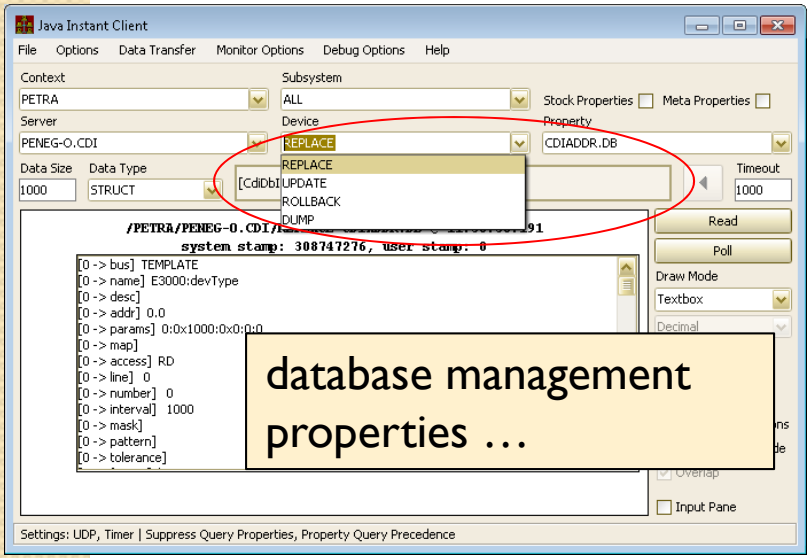
PU01I gives <BLM> as address parameter => expands into multiple devices (one for each template field): e.g. PU01I.PreScaler, PU01I.Time, etc.

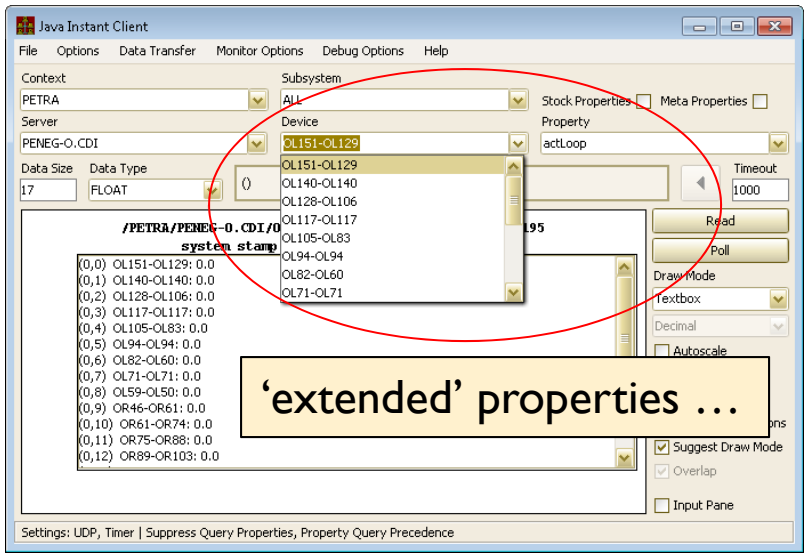
Template fields => 'Extended Properties' !
Property Mode is a TINE multi-channel array with PU01I, PU1O, etc. as keyword array elements !

CDI Server is a Property Server









CDI Asynchronous Listeners

- *IF* a CDI server is used as the primary device server *THEN*
  - It could have *many clients* !
  - Need to *efficiently access* hardware *data*.
  - Request to *monitor data* will start a *listener* (if not already started) at the desired polling interval.
    - Can specify this interval in the database
    - Can specify automatic listening in the database
      - i.e. CDI server starts listening at startup !
  - *Subsequent reads* come from the listener's *readback buffer* !

CDI Scheduling

- CDI can flag readback data to be *scheduled* upon data change (outside specified tolerance).
  - *readback* normally given by specified monitoring interval, but …
- CDI bus plugs can pipe any events all the way up to listening clients.
- Minimal latency !

CDI Remote Database Access

- CDI uses a .csv spreadsheet database
  - Those skilled in Excel might be semi-comfortable with this *but* …
    - *Still tedious and prone to errors*
  - Database changes require …
    - Distribution to local file system and
    - Server restart
  - *What about a real device server talking to multiple CDI servers ?*
    - Identical hardware distributed on many host CPUs?

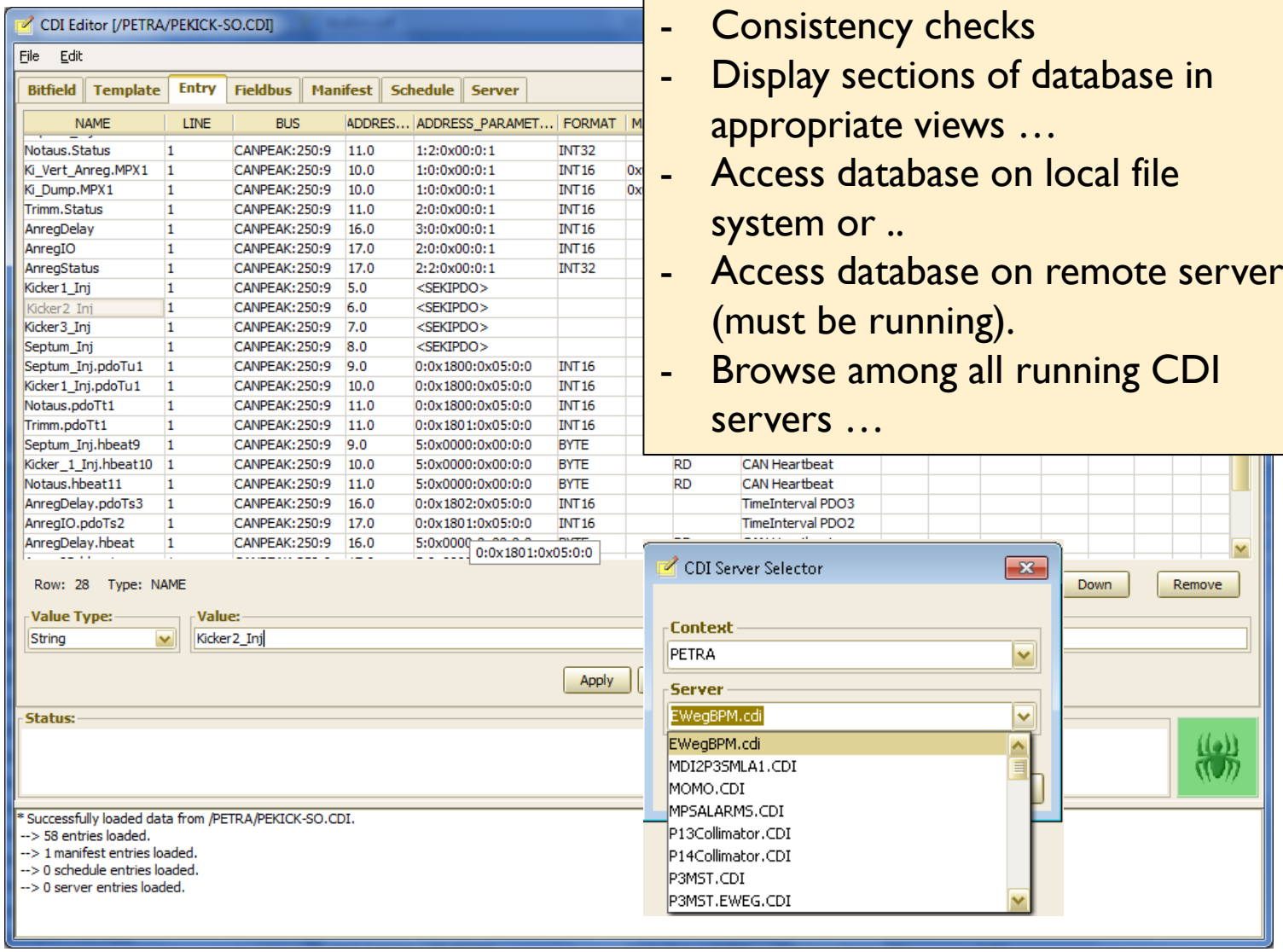
CDI Remote Database Access

- A CDI server now exposes its database as properties !
  - *Everyone* can read the database.
  - Modifying the database requires traversing TINE security.
  - If configured: can send remote 'RESET' to re-read and re-initialize !

CDI Editor

Avoid using a spreadsheet editor:

- Consistency checks
- Display sections of database in appropriate views …
- Access database on local file system or ..
- Access database on remote server (must be running).
- Browse among all running CDI servers …



CDI Deployer

- *Suppose*: Device middle layer server accesses *identical hardware devices* distributed over several host CDI Servers
  - Templates and other information needs to be repeated in **N** different databases.
    - *A simple change to a template can become an ordeal* !
- *Examples*:
  - XFEL vacuum hardware front-ends (CAN on µTCA)
  - PETRA NEG pump hardware front-ends (CAN on PC104)
  - Mass Spectrometer front-ends (OPC on Windows)
- *Solution*:
  - Maintain one master CDI address database and use the CDI deployer !

CDI Deployer

- Example of master CDI database:

TARGET	NAME	BUS	LINE	ADDRESS_	ADDRESS_PARAMETER	ACCESS	FORMAT	LIMIT	INTERVAL
	QMG:SemHvIstGbl	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|LSN	Float	1	2000
	QMG:SemHvSollGbl	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|WR|L	Float	1:01	2000
	QMG:SemCmd	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|WR	Byte	1:1	2000
	QMG:SemHvStatus	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|LSN	Long	1	2000
	QMG:DetTypeGbl	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|WR	Byte	1:1	2000
	QMG:FilActive	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|WR	Byte	1:01	2000
	QMG:FilCmd	TEMPLATE	1	0	1:0:1:0:1:0:2	RD|WR	Byte	1:01	2000
/PETRA.TEST/MassSpect.CDI.ms03	ms03						Short		2000
/PETRA.TEST/MassSpect.CDI.ms04	ms04	QMGOPC:85:131:16	1	131.169.1(	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms06	ms06	QMGOPC:85:131:16	2	131.169.1(	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms07	ms07	QMGOPC:85:131:16	2	131.169.1(	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms08	ms08	QMGOPC:85:131:16	3	131.169.1(	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms10	ms10	QMGOPC:85:131:16	3	131.169.1(	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms75	ms75	QMGOPC:85:192:16	3	192.168.9:	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms80	ms80	QMGOPC:85:192:16	3	192.168.9:	0:0:0:0:0:0:0:0:<QMG>		Short		2000
/PETRA.TEST/MassSpect.CDI.ms81	ms81	QMGOPC:85:192:16	3	192.168.9:	0:0:0:0:0:0:0:0:<QMG>		Short		2000

Supply the column header: TARGET

Specify which device goes to which target !

CDI Deployer

A command line utility which can perform miracles …

```
cdideploy /?

reads existing master cdi address db and deploys relevant entries to given targets

n.b. the master db file should have a column 'TARGET', which gives the context and server name
     of CDI server which should receive the address information (TEMPLATES, BITFIELDS, BUSNAMES
     are always sent to all targets)

usage: cdideploy [/x /f /t /v /h /n=filename /d=debug level]
     /h or /? => show this usage information
     /n=filename => use master cdi db file name (default: cdiaddr.csv)
     /x => send exit command to remote CDI server (process restarted via watchdog)
     /f => write local db files in 'target' subdirectories
     /t => just test (don't update remote CDI servers)
     /v => verbose (show log file output at the console)
     /d=debug level sets the TINE debug level (useful if updating remote CDI servers)

e.g. cdideploy
     deploys information from the master db file 'cdiaddr.csv', does not make local db files, no extra output
     at the console

e.g. cdideploy /f
     deploys information from the master db 'cdiaddr.csv' file AND makes local db files

e.g. cdideploy /n=mycdiaddr.csv
     deploys information from the master db 'mycdiaddr.csv' file
```

cdideploy /x will :

- scan master database ‘cdiaddr.csv’
- deploy relevant segments to target servers
- exit and restart target servers

CDI in operation

- Bus Plugs
  - SEDAC, CAN, Beckoff, OPC, PLCs, RS232, …
- DESY (PETRA3 complex, REGAE, FLASH, XFEL)
  - ~ 100 CDI servers
  - Motors, Screens, Vacuum pumps, valves, Temperatures, Kickers, RF stations, …
- HASYLAB
  - ~ 50 CDI servers
  - Vacuum interlock
- EMBL
  - ~ 10 CDI servers
  - Motors

Conclusions

- CDI is a major workhorse for TINE systems.
- CDI *remote database access* allows dynamic, on-the-fly editing.
- CDI *Editor* makes database configuration easier than ever.
- CDI *Deployer* allows easy updates to distributed databases.

- Web: http://tine.desy.de
- Email: tine@desy.de