



TINE Release 4.0 News

(Feb 14, 2014: That was the month that was !)

“What a long, strange trip it’s been”

[Release 4.4.0]

■ Bug-fixes and Embellishments

○ *Save/Restore*

- now works with buffered servers (e.g. [LabView](#))
- Other issues concerning array properties

○ **SYS_MASK** issue with java server

- (see doocs2tine news)

○ Some refactoring:

- **CA_MCAST** synonym for **CA_NETWORK**
- error code '**not_properly**' -> '**no_data_in_range**'

[Release 4.4.0]

■ Bug-fixes and Embellishments

- **attachfec** console application (**unix/linux builds**) no longer crashes when fed faulty input !
- **data protection mutex** to avoid concurrency issues when examining bound data independent of incoming data set.

[Release 4.4.0]

■ Save/Restore

- Issue when restoring **array data** now resolved (thanks to **MSK**)
 - Normally: save/restore used with 'attribute' style properties
 - *i.e. save/restore '1' value per device*
 - What about an 'array of values' for each device?
 - Save-restore file references other (*binary*) files containing the array data.
 - There was a problem during initialization (the restore procedure) when one of these files was missing.

[Release 4.4.0]

- **Data protection mutex (C-Library)**
 - Possible concurrency problem with *bound data*.
 - `AttachLink(target,&dout,NULL,CA_READ,CM_TIMER,cb,1000);`
 - But 'dout' is bound to e.g.
`float buffer[1000];`
 - Callback 'cb' is fired directly after buffer is filled with 'new incoming data'
 - But some other polling activity looks at the contents of 'buffer' *independent of the callback!*
 - This is what happens with an *asynchronous listener* in **MatLab** when there is no callback event monitored!

[Release 4.4.0]

- **Data protection mutex (C-Library)**
 - New API call: **LinkDataMemcpy()**
 - Will guarantee the integrity of the bound data set during the 'memcpy' operation.
 - Plus: **LockLinkData()** and **UnlockLinkData()** can be used for more lengthy operations on the bound data.

[Release 4.4.0]

- New Stock Properties:
 - “ENSADDR”
 - “STRUCTURES”
 - “SRVBASELINE”

Release 4.4.0

■ “STRUCTURES”

The screenshot displays the Java Instant Client window with the following configuration and data:

- Context:** TEST
- Subsystem:** ALL
- Server:** SineServer
- Device:** ALL
- Property:** STRUCTURES
- Data Size:** 512
- Data Type:** NAME64
- Timeout:** 1000

The main display area shows the following output:

```
/TEST/SineServer/ALL STRUCTURES @ 21:08:45.822  
system stamp: 89111, user stamp: 0  
(0,0) SineInfo  
(0,1) RgnRule  
(0,2) WRACCTBL  
(0,3) CONTBLr4  
(0,4) CLOG  
(0,5) DSUMMARY  
(0,6) HRSr4  
(0,7) AWSr4  
(0,8) ADSr4  
(0,9) ADS  
(0,10) AMSr4  
(0,11) AMS  
(0,12) CLNQS
```

On the right side, the **Draw Mode** is set to **Textbox**. Other options include **Autoscale**, **Log Scale**, **History**, **Suggest Decorations** (checked), **Suggest Draw Mode** (checked), **Overlap** (checked), and **Input Pane** (unchecked).

Settings: UDP, Timer | Suppress Query Properties, Property Query Precedence

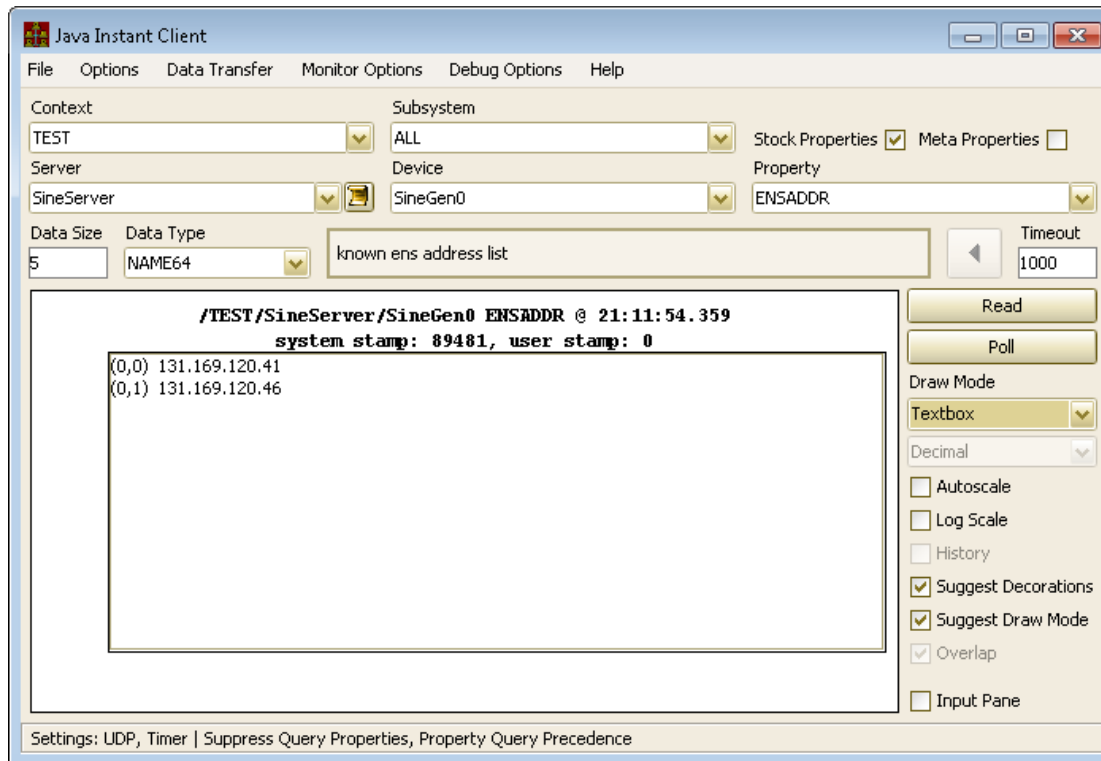
[Release 4.4.0]

- “SRVBASELINE”

- As ‘TEXT’ or ‘UTC’ Long value
- Server can ‘inform’ attached clients that ‘something has changed’ by calling `ResetServerBaseline()`.

[Release 4.4.0]

■ “ENSADDR”



[Release 4.4.0]

■ Filter Links

- Applies to *local histories*
 - Only commit to long term storage if filter condition is VALID!
- Applies to *alarm watch table* alarms
 - Only set alarm if filter condition is VALID!

Local archive Filters ...

'AppendHistoryInformation()' at initialization time. In any case, where property "<PROPERTY>" is keeping a local history, meta-properties such as "<PROPERTY>.HIST" (see [Meta Properties](#)) will be available to allow clients access to the local history data.

A 'history.csv' file (or a 'HISTORY' section within fec.xml) supports the following csv columns (or xml tags):

- **"INDEX"** provides a unique index which identifies 'this' local history record. This should be unique with respect to the entire FEC process (not just the equipment module).
- **"LOCAL_NAME"** gives the local equipment module name to which this history record refers.
- **"PROPERTY"** gives the calling property for which this local history is maintained.
- **"DEVICE"** gives the calling device for which this local history is maintained. Note that multi-channel arrays should be called as such, that is an array beginning with a specific device name or number. Calls to the local history server for a specific device (i.e. channel) name will resolve which element of the multi-channel array is required.
- **"DATA_LENGTH"** gives the data size of the local history call.
- **"FORMAT"** gives the data format of the local history call.
- **"HEARTBEAT"** provides an archive 'heartbeat' in seconds, which effectively specifies the maximum time 'gap' between archived records on the long-term storage disk. Thus a value of 900 will ensure that, regardless of tolerance conditions, a value will be stored on the disk at least every 15 minutes (900 seconds).
- **"POLLING_RATE"** gives the calling interval (in milliseconds) which the local history subsystem should use in accessing the property to be archived. Note that the property can also be 'scheduled' at a faster rate.
- **"ARCHIVE_RATE"** gives the minimum archive interval and hence the minimum time 'gap' (in milliseconds) for which data should be archived on the local disk in long-term storage. A value of 1000 for instance will ensure that there is no more than 1 data record per second stored on the disk, regardless of any scheduling or other criteria.
- **"TOLERANCE"** provides a tolerance for archiving data to disk in long-term storage. This can either be given in percent (a number followed by '%', e.g. '10%') or as an absolute value (a number).
- **"SHORT_DEPTH"** gives the depth (array size) of the short-term storage. This is essentially the size of a ring buffer which holds the results of the calling parameters. The short term data will of course disappear when the server is restarted.
- **"LONG_DEPTH"** gives the depth (in months) of the long-term storage on the local disk. A value of 0 or less will NOT write data to the local disk, in which case the short-term storage is the only source of local history data.
- **"FILTER"** gives a filter to be used in committing archive data to disk or not. If the filter criterion is correctly parsed and resolves to a valid control system address then the associated readback value is used to determine whether storage conditions are valid or not. In the event of any error, conditions are assumed to always be valid. In other words, in order NOT to reject long-term storage a filter must be correctly parsed and correctly supply a readback value which does not fulfill the filter conditions. The filter string is parsed according to / <context>/<server>/<device>[<property>]<comparator>

where <comparator> is one of "=", "!", ">", or "<".

An example of history.csv might be:

```
Index,Export Name,Local Name,Property,Device,Data Length,Format,Heartbeat,Polling Rate,Archive Rate,Tolerance,Short Depth,Long Depth,Filter
1,BPM,BPMEQM,ORBIT.X,WL197,300,float,18000,1000,10,10%,600,1,/PETRA/GLOBALS[BeamCurrent]>0.5
2,BPM,BPMEQM,ORBIT.Y,WL197,300,float,18000,1000,10,10%,600,1,/PETRA/GLOBALS[BeamCurrent]>0.5
```

The relevant section within a fec.xml file will be embedded within the associated <PROPERTY> section as shown in:

```
<PROPERTY>
  <NAME>Sine</NAME>
  <DEVICE_SET></DEVICE_SET>
  <EGU>V</EGU>
  <XEGU>F</XEGU>
  <MAX>1000</MAX>
  <MIN>0</MIN>
  <XMAX>8092</XMAX>
```

API: ApplyHistoryFilter(idx, "/PETRA/Idc[I]>0.5");

Alarm watch filters ...

The available csv columns within alwatch.csv or xml tag within the 'ALARM' tag are

- "LOCAL_NAME" gives the local equipment module name to which the call is to be made.
- "DEVICE_NAME" gives the device name to use in the monitoring call
- "PROPERTY" gives the property name to use in the monitoring call
- "SIZE" gives the data size to use in the monitoring call
- "FORMAT" gives the data format to use in the monitoring call
- "SEVERITY" gives a default severity to use in case of an alarm and in the absence of other information
- "SEVERITY_HIGH" gives the specific severity to use in case of a value_too_high alarm
- "SEVERITY_LOW" gives the specific severity to use in case of a value_too_low alarm
- "SEVERITY_HIGHWARN" gives the specific severity to use in case of a warn_too_high alarm
- "SEVERITY_LOWWARN" gives the specific severity to use in case of a warn_too_low alarm
- "ALARM_SYSTEM" gives a specific alarm system to associate with the alarm. If not given, then the registered alarm system at the CAS for the server will be used.
- "MASK" applies the given mask to the readback data before checking thresholds or patterns.
- "NORMAL" gives a pattern for a 'normal' readback value. If the readback value does not match the given pattern an 'invalid_data' alarm is issued. If the input value in this column is preceded by a '!' (NOT) then only if the readback value matches the given value is an invalid_data alarm issued.
- "COUNT_THRESHOLD" gives the number of times a threshold exceeded or pattern mismatch must occur (consecutively) before an alarm is issued.
- "HIGH" gives the high treshold for a readback value before issuing an alarm.
- "LOW" gives the low treshold for a readback value before issuing an alarm.
- "HIGHWARN" gives the high warn treshold for a readback value before issuing an alarm.
- "LOWWARN" gives the low warn treshold for a readback value before issuing an alarm.
- "ALARM_CODE" gives an alternative alarm code to assign to a threshold or pattern mismatch alarm Otherwise 'value_too_high', 'value_too_low', etc.
- "ALARM_CODE_HIGH" gives an alternative alarm code to assign to a high threshold
- "ALARM_CODE_LOW" gives an alternative alarm code to assign to a low threshold
- "FILTER" gives a filter to be used in applying an alarm or not. If the filter criterion is correctly parsed and resolves to a valid control system address then the associated readback value is used to determine whether alarm conditions are valid or not. In the event of any error, conditions are assumed to always be valid. In other words, in order NOT to apply a watch table alarm a filter must be correctly parsed and correctly supply a readback value which does not fulfill the filter conditions. The filter string is parsed according to /<context>/<server>/<device>[<property>]<comparator><value> where <comparator> is one of "=", "!=", ">", or "<".

```
LOCALNAME, DEVICENAME, PROPERTY, SIZE, FORMAT, SEVERITY, HIGH, LOW, HIGHWARN, LOWWARN, FILTER  
SINEQM, #0, SINE, 10, Float, 15, 500, 0, 400, 10, /DESY2/GLOBALS[ParticleType]=1
```

If a *fec.xml* configuration file is used, the relevant section will be embedded within the associated property and might look something like:

```
<PROPERTY>  
  <NAME>Sine</NAME>  
  <DEVICE_SET></DEVICE_SET>  
  <EGU>V</EGU>  
  <XEGU>r</XEGU>  
  <MAX>1000</MAX>
```

API: see ApplyAlarmWatchFilter()

[Release 4.4.0]

- Allow multicast mask/address 'lists'.
 - Previously: only a single entry defined the multicast mask
 - **Current systematics:**
 - sender has ip address: 131.169.151.58
 - Multicast group is: 238.1.151.58
 - '238' signals a multicast
 - '1' is a 'DESY HH' 'metric'.
 - Zeuthen is using '2' as a 'metric'
 - Use multicast mask '238.2.0.0'
 - Distribute file 'mcastmask.csv' or use ENV variable.
 - => override the systematics with metric '2' instead of '1'.

[Release 4.4.0]

- But what about the 192.168.x.x. addresses?
 - 'mcastmask.csv' can now take a list of addresses with an ip 'pattern'.

	A	B
L	ADDRESS	PATTERN
2	238.1.1.1	131.169.0.0
3	238.2.1.1	192.168.0.0
L		

- Are there 'better' systematics?

[Release 4.4.0]

- *Coming soon ...*

- Systematically use the last 3 bytes in the ip address
 - '131.169.151.58' -> '238.169.151.58'
 - '192.168.151.58' -> '238.168.151.58'
 - '141.34.200.67 -> '238.34.200.47'
 - Etc.
- Also: use '239' instead of '238'
 - Site local multicasts

[Release 4.4.0]

- *Coming soon ...*

- Involves a lot of router configurations
- Updating all the libraries to support the new systematics
- Pay attention to the '10' subnets ...
- It's a good thing we have a long shutdown ...

[Release 4.4.0]

- Console Line utilities

(or via AttachFec)

- get structures
- get mcastmask
- get filterlinks

Console Line utilities

```
ca Z:\Projects\Service\vc++\tine32R4\SineGen\vc___Win32_Debug_MTVfec.exe
>get filters
>debug text filter :
>debug negative text filter :
>get structures
>structure registry : user-defined structures
> (local) : Funky
> (local) : StCmp
> (local) : StBod
> (local) : StHdr
> (local) : SineInfo
>structure registry : system-defined structures
> (system) : RgnRule
> (system) : WRACCTBL
> (system) : CONTBLr4
> (system) : CLOG
> (system) : DSUMMARY
> (system) : HRSr4
> (system) : AWSr4
> (system) : ADSr4
> (system) : ADS
> (system) : AMSr4
> (system) : AMS
> (system) : CLNQS
> (system) : CTQSr4
> (system) : CONQS
> (system) : AQS
> (system) : PRPQSr4
> (system) : XPQS
> (system) : PQSx
> (system) : PQS
> (system) : FDSr4
> (system) : SrvSetQy
>get filterlinks
> Current Filters <local alarms/histories>
> /DESY2/GLOBALS [ParticleType] = 1 : valid
> /TEST/SineServer/SineGen0 [Amplitude] < 300 : not valid
>get mcastmask
>multicast address mappings
>default: 131.169.9.161 -> 238.1.9.161
>
```

[Release 4.4.0]

- Contract persistence ...
 - New API: `GetContractDataReference()`
 - Allows persistence over multiple calls of a given contract.
 - Together with other referenced API calls:
 - Seamless C++ API: future presentation by Karol

[Release 4.4.0]

- ENS address resolution
 - cshosts.csv (location given by TINE_HOME)
 - *multiple addresses*
 - ENV: TINE_ENS
 - *multiple addresses*
 - Multicast query
 - *single address* (todo: catch multiple responses)
 - DNS:
 - tineens (or tineens1)
 - tineens2
 - tineens3

[Release 4.4.0]

- New Meta Properties:

- *IF* a property maintains a local history
 - **.RBAVE** (ring-buffer average over interval)
 - **.RBMAX** (ring-buffer maximum over interval)
 - **.RBMIN** (ring-buffer minimum over interval)
 - **.RBSUMMARY** (ring-buffer ave, max, min, deviation over interval)
- e.g. stored in ring buffer @ 10 Hz
 - Can obtain and archive summary @ 1 Hz

[Release 4.4.0]

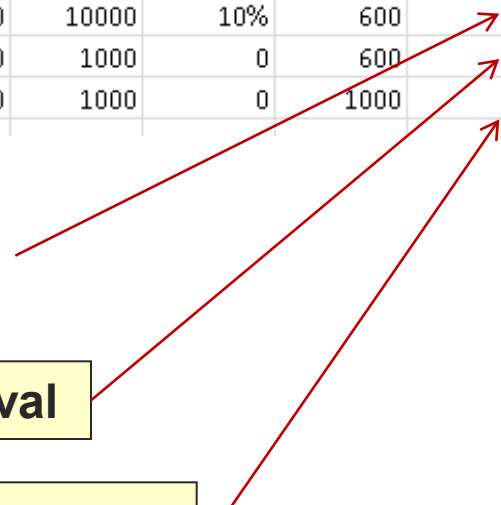
■ Archiving summary info locally:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Index	Local Nam	Property	Device	Data Leng	Format	Heartbeat	Polling Ra	Archive R	Tolerance	Short Dep	Long Depth	
2	1	SJNEQM	Sine	#0	1024	float	1800	250	10000	10%	600	1	
3	2	SJNEQM	Amplitude	#0	10	float	20	100	1000	0	600	0	
4	3	SJNEQM	Frequency	#0	10	float	20	100	1000	0	1000	-1	

Normal long-term archive: no summary

Keep 1 month summary @ 1 sec. sample interval

No long-term archive, no summary



[Release 4.4.0]

- Storing ring buffer summary info:
 - *Central archive:*
 - Link to e.g. <P>.RBAVE and save as reasonably named keyword
 - *Local archive:*
 - Retrieve as e.g. <P>.RBAVE.HIST

[Release 4.4.0]

- doocs2tine matters ...
 - Multi-dimensional array data type implemented and mapped !
 - Format code: **CF_MDA**
 - *carried data type specified by 'dTag'*
 - Format type: **DMDA**
 - header info
 - comment (?)
 - number dimensions
 - carried data type
 - axis information (maximum 6 dimensions)
 - the data

Release 4.4.0

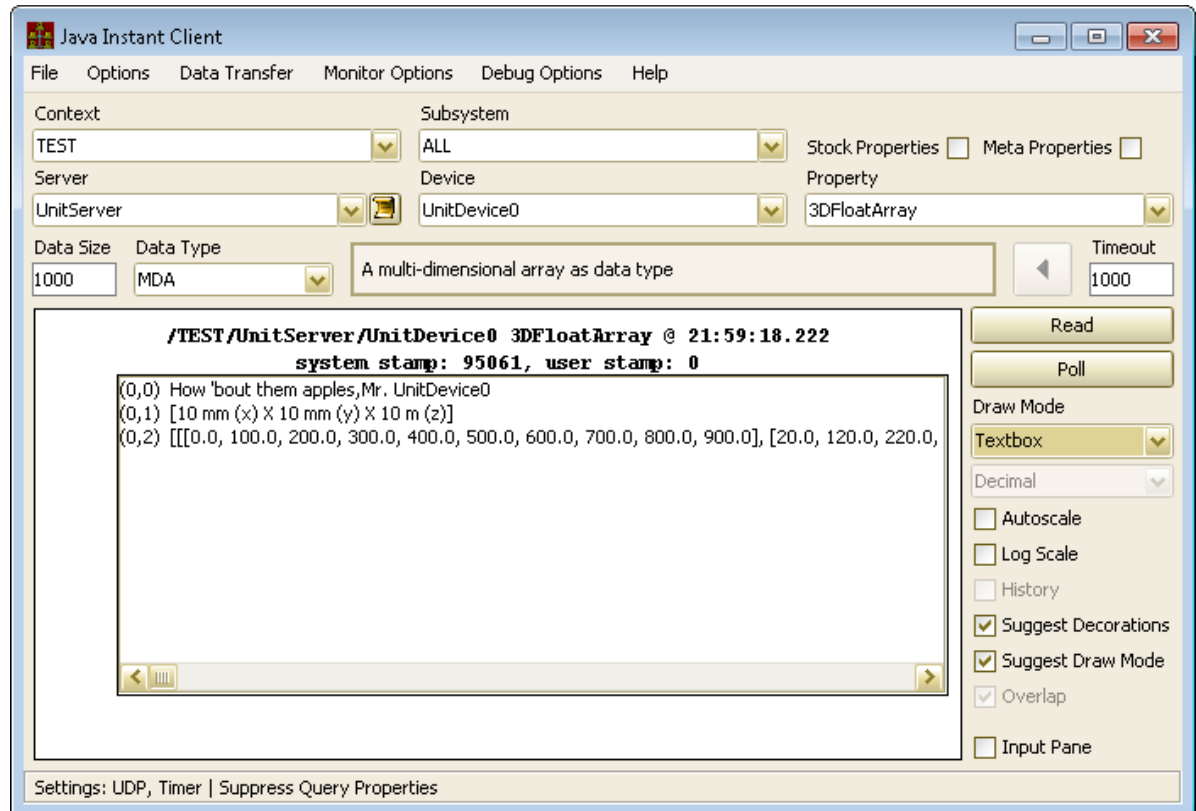
- MDA helper routines (C lib):

```
/**
 * \struct MDX
 * \brief Multi-Dimension array aXis
 *
 */
typedef struct
{
    int len;           /**< the number of elements in this dimension */
    int reserved;     /**< reserved field */
    char label[16];   /**< the axis label */
    char units[16];   /**< the axis units */
    float min;        /**< the axis value of the initial element */
    float max;        /**< the axis value of the last element */
} MDX;
#define MDX_SIZE 48

/**
 * \struct DMDA
 * \brief Datatype: Multi-Dimensional Array
 *
 */
typedef struct
{
    char cmt[80];     /**< descriptive comment */
    int ndim;        /**< number of dimensions */
    int fmt;         /**< data type format of the data */
    MDX axis[MAX_MDA_DIMS]; /**< axis information array */
    UNION data;      /**< accompanying data */
} DMDA;
#define MDA_FMT_OFFSET 84
TINE_EXPORT int MakeMDA(DMDA *mda, char *cmt, int fmt, void *data, int ndim, ...);
TINE_EXPORT void *GetMDAElemRef(DMDA *mda, ...);
int MDAtoDTYPE(DTYPE *d, DMDA *mda, int bufferlength);
```

Release 4.4.0

- Usage in java or C, C# depends on how multi-dimensional arrays are dealt with on those platforms



[Release 4.4.0]

- doocs2tine matters ...
 - Event number multicast (from Vladimir) used as 'CycleNumber' tag.
 - IF Context server 'CYCLER' is a doocs server!

Synchronizing with the system stamp ...

The screenshot shows the 'Archive Viewer: DESY2' application window. The main plot area displays a time-series graph with a y-axis ranging from -1000 to 1000 and an x-axis from 6e5 to 1.3e6. A red vertical line is positioned at approximately 7e5. A context menu is open over the plot, listing options such as 'Use Optical Zoom', 'Use System Stamp', and 'System Stamp Offset'. Below the plot, a status bar indicates the current time: 'Wed Jan 22 00:00:00 CET 2014 (offset = 1.75E8) 23 Hours'.

Below the plot, the 'Live' section displays system parameters:

Status	Property [Device]	Value	Description	Log
OK	Energy	6.30 GeV	Energy	<input type="checkbox"/>
OK	Chop.Position [BUL90]	-3.20E-03 m	Chopper-Blenden Is...	<input type="checkbox"/>
OK	DIP.AC.Ist	329.62 A	Ist Values for D2 P5	<input type="checkbox"/>
OK	DIP.AC.Soll	329.61 A	Soll Values for D2 P5	<input type="checkbox"/>
OK	Freq-RFist	499.66 MHz	R/O in MHz	<input type="checkbox"/>
OK	Orbit.X.LWeg [M20]	-2.81 mm	L-Weg Orbit X	<input type="checkbox"/>

At the bottom of the interface, there are buttons for 'Refresh All', 'Remove Selected', and 'Remove All'. A 'Calendar' widget shows the date 'January 22, 2014' highlighted. A status bar at the very bottom reads: '22:07:13: History data for selected channels loaded.'

[Release 4.4.0]

- doocs2tine matters ...
 - Stock property alias 'SYS_MASK' for 'DEV_MASK' added to java server code.
 - And a bugfix:
 - wild-card calls were passing the sys_mask input along.

[Release 4.4.0]

- Doocs2Tine news (repeated from last time)
 - DOOCS watchdog integration in FEC Remote Panel
 - Systematics requires:
 - watchdog server has name
 - <host>.WATCH in same context as server
 - => are there servers on same host in different contexts?
 - subsystem should be 'WATCH'
 - decorated context <context>.WATCH
 - Watched server should have **location name**
 - SVR.<server name>
 - and NOT SVR.<binary name> !

Release 4.4.0


Alarm Regions: almost there ...

Alarm Viewer: PETRA

File View Options Navigate Help

Context: PETRA

Fatal	Error	Warning
41	7	171

Alarm Display  Live Archive

Fri Feb 14 08:00:57 Warning Severity >= 0 Selected/Total No. of Alarms: 219/219 Active Alarms Only (9 Disabled)

System	Device Name	Message	Sev	Alarm Descriptor	Alarm Time	Duration
Magnete	Main-EW1	> N PS ALARMS	13	New	08:00:54.549 - Feb 14 CET	2 sec
Magnete	Main-NO2	> N PS ALARMS	13	New	08:00:54.128 - Feb 14 CET	3 sec
Magnete	Main-NL	> N PS ALARMS	13	New	08:00:53.529 - Feb 14 CET	4 sec
Magnete	Main-O	> N PS ALARMS	13	New	08:00:52.847 - Feb 14 CET	4 sec
Magnete	Main-EW2	> N PS ALARMS	13	New	08:00:51.841 - Feb 14 CET	5 sec
Magnete	Main-EXR	> N PS ALARMS	13	New	08:00:51.393 - Feb 14 CET	6 sec

Fatal	Error	Warning	Count
6	0	0	47
10	0	0	50
0	0	0	3
0	0	0	0
2	0	0	0
0	3	0	19
0	1	49	0
0	0	0	2

central Inj. east south west north

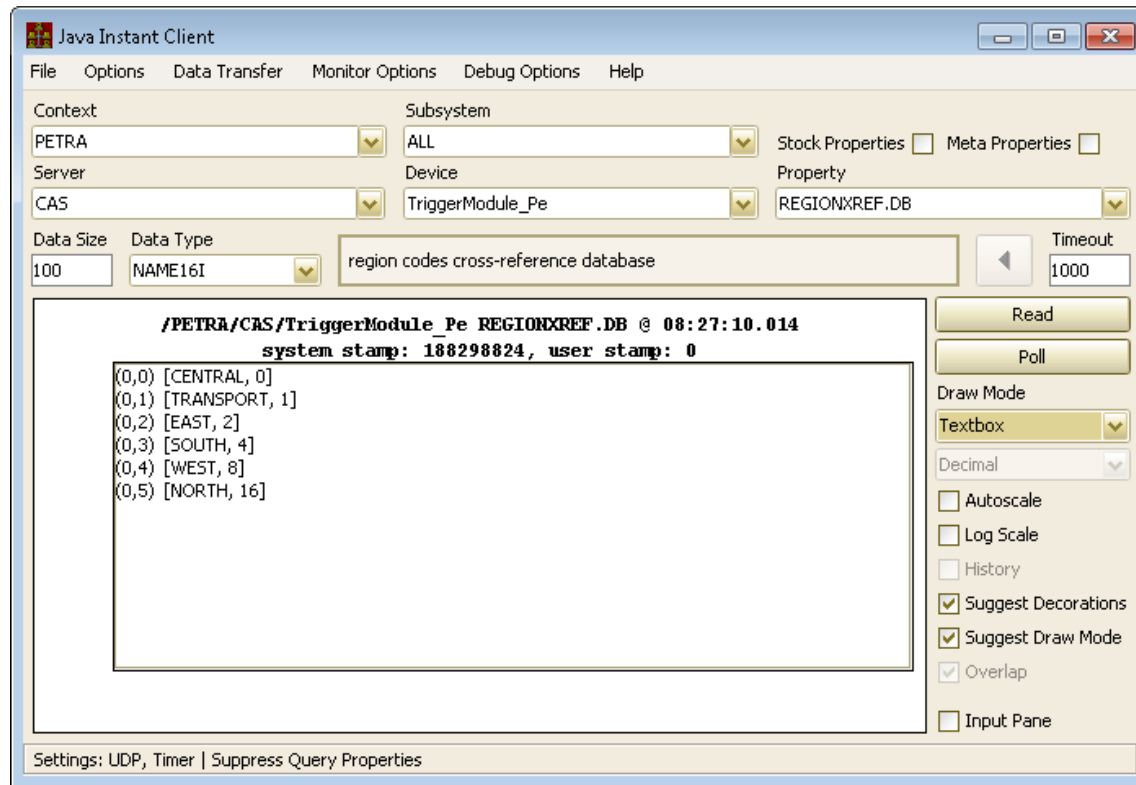
08:00:20: Alarms loaded.

[Release 4.4.0: Alarm Regions]

- server **configures** or *learns* region information for its devices
 - API, devices.csv, or fec.xml
 - Ask CAS at startup
- **device alarm** can then **carry region information**.

Release 4.4.0: Alarm Regions

- CAS manages region information



Release 4.4.0: Alarm Regions

- CAS manages region information

The screenshot shows the Java Instant Client interface. The 'Device' dropdown menu is highlighted with a red circle and contains the value 'Idc.OR08'. The main display area shows the following output:

```
/PETRA/CAS/Idc.OR08 REGIONRULES @ 08:31:43.180  
system stamp: 188300531, user stamp: 0  
[0 -> pattern] *  
[0 -> region] EAST  
[0 -> code] 2  
[1 -> pattern] *OR*  
[1 -> region] EAST  
[1 -> code] 2  
[2 -> pattern] *OL*  
[2 -> region] EAST  
[2 -> code] 2  
[3 -> pattern] *SR*  
[3 -> region] SOUTH  
[3 -> code] 4  
[4 -> pattern] *SL*  
[4 -> region] SOUTH  
[4 -> code] 4  
[5 -> pattern] *WR*
```

Settings: UDP, Timer | Suppress Query Properties

[Release 4.4.0: Alarm Regions]

- Either:
 - 1) the server programmer/responsible person *configures the region information*
 - 2) a *naming convention* is adhered to !