



TINE Release 4.0 News

(Nov 12, 2010: That was the month that was !)

“What a long, strange trip it’s been”

[Release 4.1.10]

- Important Bug Fixes / Improvements:
 - **Latency** for initial call attaching to a *redirected* server dramatically **reduced!**
 - synchronous calls in **Java** ('100 ms' -> ~ clock tick)
 - asynch. calls with CM_WAIT in **C** (several clock ticks -> ~ 1 clock tick)
 - e.g. asynchronous *Listeners* do this.
 - **Protection** against absurdly large Link '*heartbeats*'
 - Heartbeat for TIMER link given by polling interval.
 - > 16383 (0x3fff) trips over a 'submit request' flag.
 - 600 second cap !
 - **Return codes** from equipment module or property handlers now '**validated**'
 - 'systematic' codes (e.g. *invalid_interval*, *illegal_protocol*, etc.) are rejected
 - return '*not_applicable*' instead

[Release 4.1.10]

- Important Bug Fixes / Improvements:
 - **watchdog** link table **mutex** added.
 - **CM_DATACHANGE** or **CM_EVENT** mode will initiate a link watchdog to the target server (if not already active).
 - Connection management
 - If such a link is **started** and then **closed** upon receipt of the initial callback there was a **concurrency** problem!
 - *Now repaired via this mutex !*

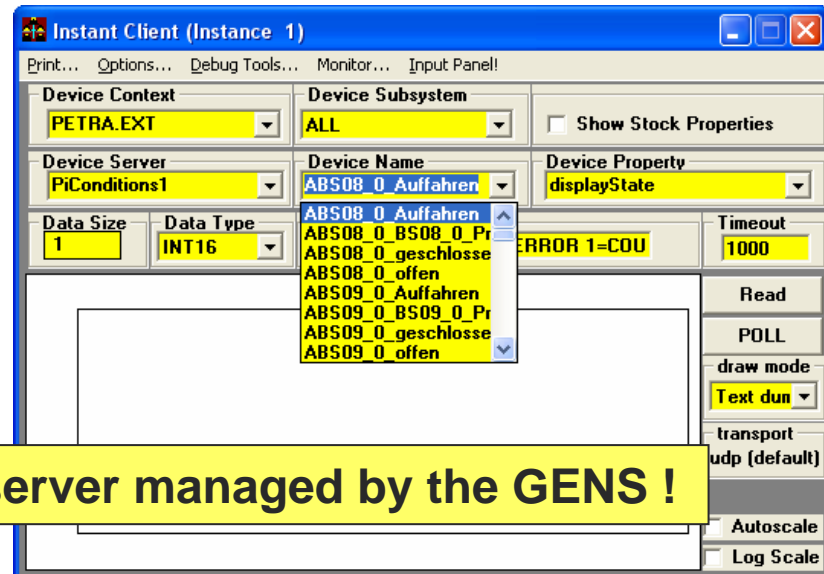
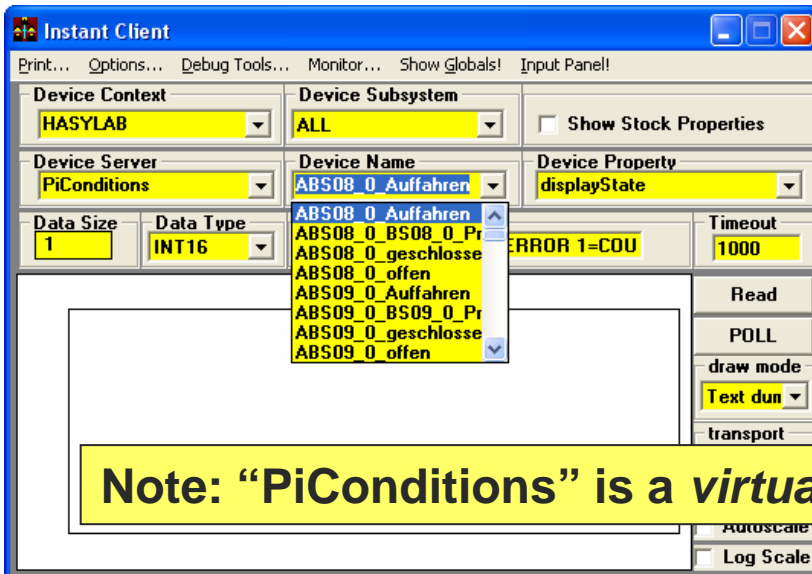
[Release 4.1.10]

- Asynchronous Listeners
 - Used in LabView, MatLab, tine2tango, tineRepeater.
 - Add a table Mutex to avoid an 'operation_busy' message reported by HASYLAB (via tine2tango).
 - Initial bug in initialization led to a windows handles leak in LabView applications.
 - Can now set the listener table capacity to a value other than the default 5000.

Release 4.1.10

tine repeater: acquire and re-register and export information from the specified TINE device server

Usage : `tineRepeater <context> <device server> [/c=<new context> /s=<new server name> /f=<fec name> /p=<port offset> /r=<polling interval> /m=<polling mode> /l=<listener table capacity> /x=TRUE /d=<debug level>]`



Note: "PiConditions" is a *virtual* server managed by the GENS !

[Release 4.1.10]

■ New Stock and Meta Properties

○ “**SRVSUBSYSTEM**”

- returns the registered subsystem

○ “**ADDALIAS**”

- Adds a *name/alias* pair to the local alias table.
- Updates ‘alias.csv’

○ “**.XEGU**”

- returns the x-axis egu
- If property is NOT a wave-form spectrum, then max and min are ‘0’ and units are “”

[Release 4.1.10]

■ Local History News

- **“/SAVED/”** subdirectory
 - scanned for additional local history files
 - not affected by cleanup operations.
 - **SetHistoryStaticFilesRepository()** can set the directory path (“./SAVED” is the default).
- **‘standard’** local history files vs **‘dynamic’** history files
 - Always available in release 4 but:
 - **‘dynamic’** is the default !
 - **‘standard’** files are **‘non-fragmented’** and created once for the **‘worst-case’** archive capacity scenario.
 - Assume no filtering at the given polling interval (allow buffer space).
 - Utility **‘mkhstfiles’** will create the standard file set from a **history.csv** file.

Release 4.1.10 (mkhstfiles)

Configures 'standard' local history random access files according to contents of history.csv

Usage : mkhstfiles [/m=<eqm name> /s=<server name> /p=<history home> /l=<legacy home> /b=<spillover buffer size>] /t=<monthly>

e.g. mkhstfiles /m=BPMEQM /l=x:\histories\BPM

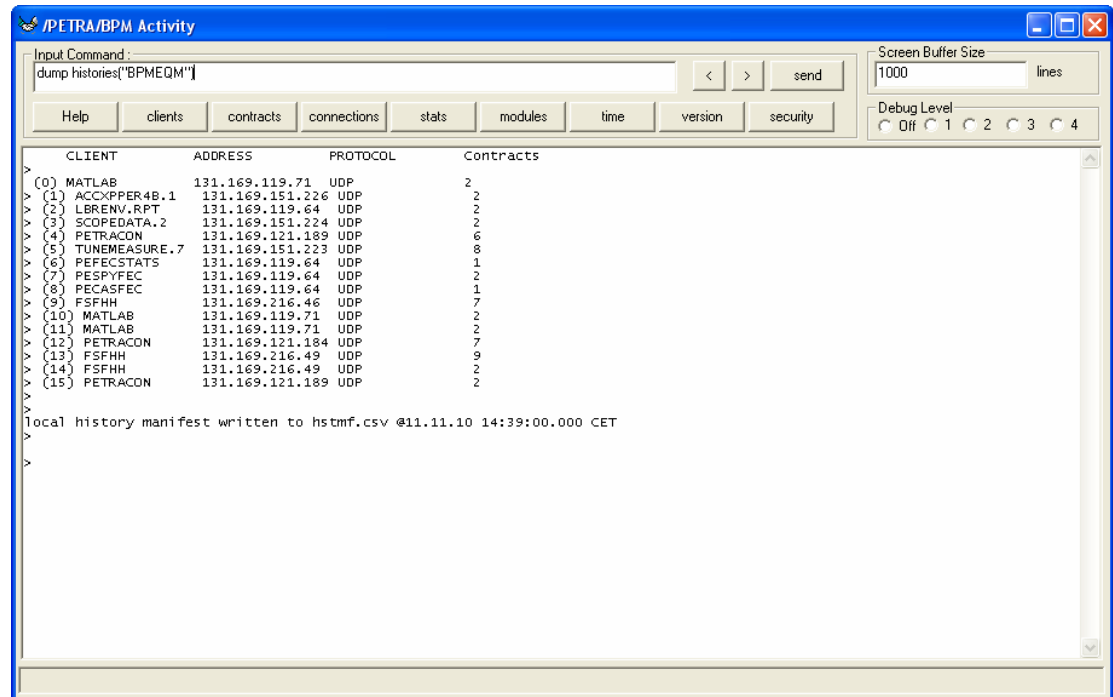
- will look for history.csv in the BPMEQM subdirectory of the FEC_HOME area
- will assume that HISTORY_HOME is given by an environment variable or is a HISTORY directory adjacent to FEC_HOME
- will search for legacy history files in x:\histories\BPM and convert the contents of any files found

e.g. mkhstfiles /s=BPM

- will look for history.csv in the FEC_HOME area and attempt to match the contents of the EXPORT_NAME column with the server 'BPM'
- will assume that HISTORY_HOME is given by an environment variable or is a HISTORY directory adjacent to FEC_HOME
- will assume that any legacy files are located in the same directory or in a 'SAVED' subdirectory of HISTORY_HOME

Release 4.1.10

- Local History News
 - Don't have a history.csv ?
 - Call 'dump histories' at the console command prompt !



The screenshot shows a terminal window titled "/PETRA/BPM Activity". The "Input Command" field contains "dump histories('BPMEQM')". The output is a table with columns "CLIENT", "ADDRESS", "PROTOCOL", and "Contracts". Below the table, a message states "local history manifest written to hstmf.csv @11.11.10 14:39:00.000 CET".

CLIENT	ADDRESS	PROTOCOL	Contracts
(0) MATLAB	131.169.119.71	UDP	2
> (1) ACCPPER4B.1	131.169.151.226	UDP	2
>> (2) LBRENV.RPT	131.169.119.64	UDP	2
>>> (3) SCOPEDATA.2	131.169.151.224	UDP	2
>>>> (4) PETRACON	131.169.121.189	UDP	6
>>>>> (5) TUNEMEASURE.7	131.169.151.223	UDP	8
>>>>>> (6) PEFECSSTATS	131.169.119.64	UDP	1
>>>>>>> (7) PESPYFEC	131.169.119.64	UDP	2
>>>>>>>> (8) PECASFEC	131.169.119.64	UDP	1
>>>>>>>>> (9) FSFHH	131.169.216.46	UDP	7
>>>>>>>>>> (10) MATLAB	131.169.119.71	UDP	2
>>>>>>>>>>> (11) MATLAB	131.169.119.71	UDP	2
>>>>>>>>>>>> (12) PETRACON	131.169.121.184	UDP	7
>>>>>>>>>>>>> (13) FSFHH	131.169.216.49	UDP	9
>>>>>>>>>>>>>> (14) FSFHH	131.169.216.49	UDP	2
>>>>>>>>>>>>>>> (15) PETRACON	131.169.121.189	UDP	2

local history manifest written to hstmf.csv @11.11.10 14:39:00.000 CET

[Release 4.1.10]

- Thread Priorities (C only)
 - SetKernelPriority() calls
 - SetServerThreadPriority()
 - SetClientThreadPriority()
 - SetBackgroundThreadPriority()

Release 4.1.10

Thread Priorities

int SetServerThreadPriority (int *priority*)

Determines the priority of the server cycle thread as well as other associated server-side threads.

The priority passed should be regarded as an offset to the 'normal' thread priority for the operating system in use. That is, the TINE definition `STD_THREAD_PRIORITY` will be 'correct' for the target operating system (where a multi-threaded build of the library is in use). It is up to the developer to take the appropriate steps to ensure that this call in fact works as planned. For instance, using real-time priorities on Windows will require a call to the Windows function `SetPriorityClass()` using `REALTIME_PRIORITY_CLASS` prior to server initialization (otherwise the resulting thread priorities will not be as desired). Using any non-standard priorities on Unix will likely require running the application as root! VxWorks will take the initial task priority as a reference for the 'standard thread priority'.

Parameters:

priority is the desired thread priority (default `STD_THREAD_PRIORITY`).

Returns:

the assigned server thread priority.

See also:

[GetServerThreadPriority\(\)](#).

Caveats:

-Windows -> apps need to call 'SetPriorityClass()' if e.g. REALTIME priority is desired.

-Unix (posix) -> apps will need to run as root

[Release 4.1.10]

- New server console diagnostics
 - Calculation of 'Average busy time' now refers exclusively to the serverCycle() (e.g. the server cycle thread)
 - Roughly follows CPU time for the process
 - 'get stats' now also displays
 - current server and client IDLE state
 - current system idle time
 - current contract data stale count
 - current delivery stale count
 - 'set serverIdle', 'set clientIdle' now available
 - 'flush contracts' now available

[Release 4.1.10]

- Software failover news:
 - A failover **SLAVE** which has become the '**MASTER**' will now:
 - Note when the real MASTER is back on line.
 - Flush its contract lists and remain idle for 2 minutes.
 - This forces any attached clients back to the '*official*' MASTER.

Database Managers

Event Archive

The image displays two software windows. The top window is the 'Event Archive Database Manager' with a blue title bar. It features a left sidebar with 'Options' and 'Edit Viewer Configurations'. The main area shows a 'Trigger Action List' with a list of commands such as '/WAIT 20 seconds' and various '/PETRA/MHFTrcTranslator/#0-#15[SL_Cy1] -> /PETRA/...' entries. Below this is the 'Action Items' section with fields for 'Device Context', 'Device Server', 'Device Name', 'Device Property', 'Array Size', and 'Format'. The bottom section contains 'Data Operations' with fields for 'offset', 'scale', 'repeat', 'interval', 'delay', 'wait', 'target', and 'timeout', along with 'Input Data' options like 'has data', 'Write Access', and 'input data' type selection (BYTE, INT16, INT32, FLOAT, DOUBLE).

The bottom window is the 'PETRA TRC Viewer Configuration Editor' with a blue title bar. It has a 'Configuration Navigation' section with dropdowns for 'PETRA' and 'mhf_sl1cav_trc', and a 'Selected:' field showing 'SL: 1Cav'. The 'Configuration Settings' section contains a text area with paths like '/PETRA/PE_SL_Cy1/Ch1.Amplitude[SAMPLE]' and '/PETRA/PE_SR_Cy1/Ch1.Amplitude[SAMPLE]'. At the bottom, there are dropdowns for 'stored server' (PE_SL_Cy1), 'stored device' (Ch1.Amplitude), and 'stored' (SAMPLE), along with an 'Add Item' button.

[Notes:]

- The database managers are for designated 'administrators' and **NOT** for the general public
 - '*user-friendliness*' is 'best effort'
 - Admins are advised to learn and deal with the '*quirks*'.
- Once again:
 - **WRITE** access does NOT necessarily have anything to do with **Input** Data !