TINE Release 4.0 News

(Feb 6, 2009: That was the week that was !)

“What a long, strange trip it’s been ….”

TINE Kernel 4.0.6

- Long break since last meeting
  - Many ‘small changes’ and ‘big changes’
  - Infrastructure:
    - Tunnel to EMBL finally works!
    - Loose ends with CYCLE number from LINAC2, PIA

TINE Kernel : Recent Bug Fixes

- Java:
  - Multiple, identical, simultaneous, asynchronous links now work properly
    - (For this use-case: thank K. Rehlich and jDDD)
  - Client can now send LARGE datasets
    - (Thank you J. Maass)
  - Server’s completion string is now correctly set
    - (Thank you J. Wilgen)
  - A Server’s property handler now sees the CA_FIRST access bit properly
    - (Thank you S. May)

TINE Kernel: Standard Lib

- Fail-safe scheduling of local history at high rates now possible
  - (Thank you K. Brede)
- VxWorks build now officially ‘multi-threaded’
  - TINE threads/Mutexes wrap VxWorks tasks/Semaphores
  - Inherit priorities and flags from spawning task
  - Caveat: Watch the FPU flag when you spawn!
- And now for some fun stuff …

TINE 4.0.6 New Features

- New Stock Properties:
  - “SRVIDLE” (or API call)
    - Sets the equipment module to an IDLE state.
    - In idle state:
      - Calls receive ‘server_idle’
      - Background tasks, history call, alarm calls, scheduling, etc. suspended.
  - "DEVDESCRIPTION“
    - Set via API call or devices.csv file column
    - Descriptive string of the device
      - (Thank you H. Wu)
- FEC-Specific properties
  - now respond even if no equipment module has been registered
  - e.g. “LOGFILE”, “SRVOS”, “STOCKPROPS”, etc.

TINE 4.0.6 New Features

- Link Watchdog(s) for DATACHANGE links
  - CM_DATACHANGE (i.e. CM_REFRESH)
  - Starts a polling link to the FEC @ given polling rate
    - Property “SRVSTARTTIME”
    - If server ‘disappears’ then ALL CM_DATACHANGE links to the FEC will receive an immediate callback with link_timeout
    - no waiting for a missing link heartbeat !

TINE 4.0.6 New Features

- STAND ALONE mode
  - Only set via environment variable
    - set **TINE_STANDALONE = TRUE**
  - Server
    - does not attempt to register with the ENS
    - Updates/creates its entry in the local client address cache using the loopback address.
  - Client
    - does not attempt to use the ENS
    - Finds the loopback address in the local address cache

TINE 4.0.6 New Features

- New Format : CF_STRING
  - Array of non-fixed length strings !
  - In C, C++:
    - DTYPE::data.strptr = &myStringArray;

```
int getStringArrayTest(void)
{
  DTYPE dout;
  char *testStr[10];
  int cc, i;

  dout.dFormat = CF_STRING;
  dout.dArrayLength = 10;
  dout.data.strptr = &testStr;
  dout.dTag[0] = 0;
  cc = ExecLinkEx("/TEST/WinSineServer/SineGen0","StringArray",&dout,NULL,CA_READ,1000);

  if (cc == 0)
  {
    for (i=0; i<10; i++) printf("%s\n",testStr[i]);
  }
  return strid;
}
```

TINE 4.0.6 New Features

- New Format : CF_STRING
  - Array of non-fixed length strings !
  - In Java:
    - new TDataType(new String[10]);

```
String[] strvals = new String[10];
TDataType wrtype = new TDataType(strvals);
TLink strlnk = new TLink("/TEST/WinSineServer/SineGen1","StringArray",
    new TDataType(strvals),wrtype,TAccess.CA_READ);

if ((cc=strlnk.execute()) == 0)
{
  for (int i=0; i<10; i++)
    System.out.println(strvals[i]);
}
```

TINE 4.0.6 New Features

- NAME16, NAME32, NAME64, etc.
  - Still more efficient with scanning, archive retrieval, queries, etc.
  - No need to ‘realloc’ string length capacities, etc.
  - Nothing fancy going on ‘under the hood’.
  - Remember:
    - a ‘String’ is not a primitive!
    - is an array of chars terminated with a ‘0’.
- But, opening the door for relaxing length restrictions on device names, property names, etc.
  - queries use CF_NAME64 at the moment.

TINE 4.0.6 New Features

- Property Access Lists:
  - Individual properties can be assigned their own ‘users’ access lists.
    - <property>-users.csv (a la users.csv)
    - Or API call.
  - Applied after all general access lists have been cleared!
  - Not yet in java (but soon).

TINE 4.0.6 New Features

- New Transfer Mode: CM_STREAM
  - Uses TCP/IP + blocking sockets
    - Note: CM_CONNECT uses TCP/IP and non-blocking sockets + stream parceling so as to apply TINE connection management.
  - Only available on multi-threaded builds.
  - Data flow from server to client is a stream
    - No timeout notification in asynchronous mode!
  - Not yet available on java server (but soon!).

TINE 4.0.6 New Features

- Property Signal Functions
  - Can register a signal handler to receive certain notifications concerning property access:
    - PS_NONE 0x00 /**< NULL signal */
    - PS_ACCESS 0x01 /**< is being accessed by a caller */
    - PS_RETRY 0x02 /**< property is being retried */
    - PS_LATE 0x04 /**< property is being returned late */
    - PS_PENDING 0x08 /**< is being called while last transmission still pending */
    - PS_SENT 0x10 /**< property has been sent to caller */
    - PS_ALL 0xff /**< signal on any event */

TINE 4.0.6 New Features

- More information updated/include in the TINE web site.
  - API call updated (alphabetically sorted)
  - error codes with (brief) explanation
  - More examples (e.g. SetAccessLock())
- More java doc!

Java API Caveat:

- Sending data from a client:

```
float[] a = new float[1];
float[] b = new float[1];
TDataType dout0 = new TDataType(a);
b[0] = (float)300.0;
TDataType din0 = new TDataType(b);
TLink link0 = new
   TLink("/TEST/LxSineServer/SineGen0","Amplitude",dout0,din0,TAccess.CA_READ|TA
cc = link0.execute();
```

Sends ‘300.0’ to the server

```
float[] a = new float[1];
float[] b = new float[1];
TDataType dout0 = new TDataType(a);
TDataType din0 = new TDataType(b);
b[0] = (float)300.0;
TLink link0 = new
   TLink("/TEST/LxSineServer/SineGen0","Amplitude",dout0,din0,TAccess.CA_READ|TAc
cc = link0.execute();
```

Sends ‘0’ to the server

Java API Caveat:

- Data references are ‘bound’ to the TDataObject at the time of creation
  - Internal byte stream, etc.
- Input Data is ‘Part of the Contract’ !!

```
float[] a = new float[1];
float[] b = new float[1];
TDataType dout0 = new TDataType(a);
TDataType din0 = new TDataType(b);
b[0] = (float)300.0;
din0.putData(b);
TLink link0 = new
   TLink("/TEST/LxSineServer/SineGen0","Amplitude",dout0,din0,TAccess.CA_READ|TA
cc = link0.execute();
```

Sends ‘300.0’ to the server