TINE Release 4.1
Responding to the User's Needs

Emphasis on *Control System Evolution*

Control System Evolution

- Some forces driving ‘evolution’ :
  - Things are changing around you …
    - 64-bit OSes are now common
    - Gigabit ethernet is now common
    - New hardware available, Old hardware is obsolete, etc.
    - New technologies appear (and disappear) all the time.
  - Users are requesting change …
    - ‘Fix this bug …’
    - ‘Add this feature …’
    - ‘I need an interface to Software X …’
  - You want to improve things anyway !

The Red Queen Syndrome

‘The most curious part of the thing was that the trees and the other things round them never changed their places at all: however fast they went, they never seemed to pass anything.’

…

`Now, *here*, you see, it takes all the running *you* can do, to keep in the same place. If you want to get somewhere else, you must run at least twice as fast as that!'

- Lewis Carroll, “Through the Looking Glass”

See also: “The Red Queen” by Matt Ridley

The '*Race Condition*'

- Predators and Prey
  - Predators devise new ways to trap prey
  - Prey devise new ways to escape predators
- Viruses and the Immune System
  - (ditto)

"*The natural enemy of the control system developer is the User*!"

Who are the users?

- Application developers
  - API problems

    “why didn’t that work?”, “I need a method to …”
- Hardware engineers
  - behavioral problems

    “why don’t I have an archive of …”
- Machine physicists/operators
  - systematic problems

    “why do I see a fatal alarm with beam in the machine?”
- Other control systems

***problem* = bug or feature request**

Application Developers and TINE 4.1

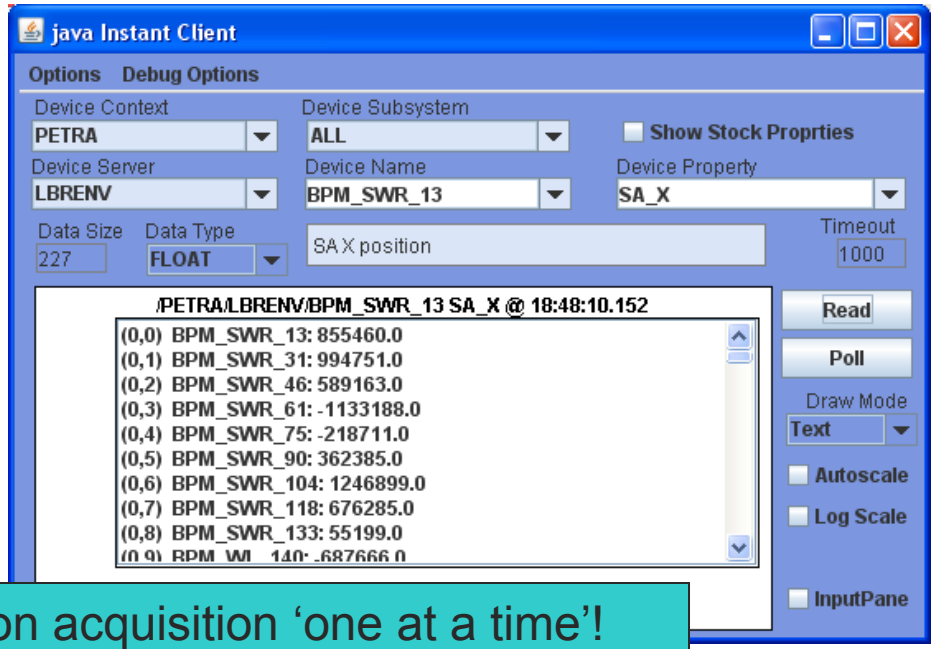
- Some new TINE 4.1 API Features
  - Security :
    - *Property* specific access lists
    - Access locks
      - easier to use
      - offer exclusive read
  - Optional dispatch routines :
    - Property access signals
      - accessed, retried, pending, sent, etc.
    - Cycle Trigger functions
      - Optional dispatch routine
  - Data objects 'stamped' with the cycle number
  - Scheduling can be eager or lazy
  - etc.

/Context/server/device [*property*]

Application Developers and TINE 4.1

- Example of new ‘use case’ and response :
  - Multi-Channel Arrays (**MCA**s)

MCAs are an efficient way to atomically deliver a collection of devices (all 300 vacuum pressures, power supply currents, BPMs, temperature sensors, etc.)



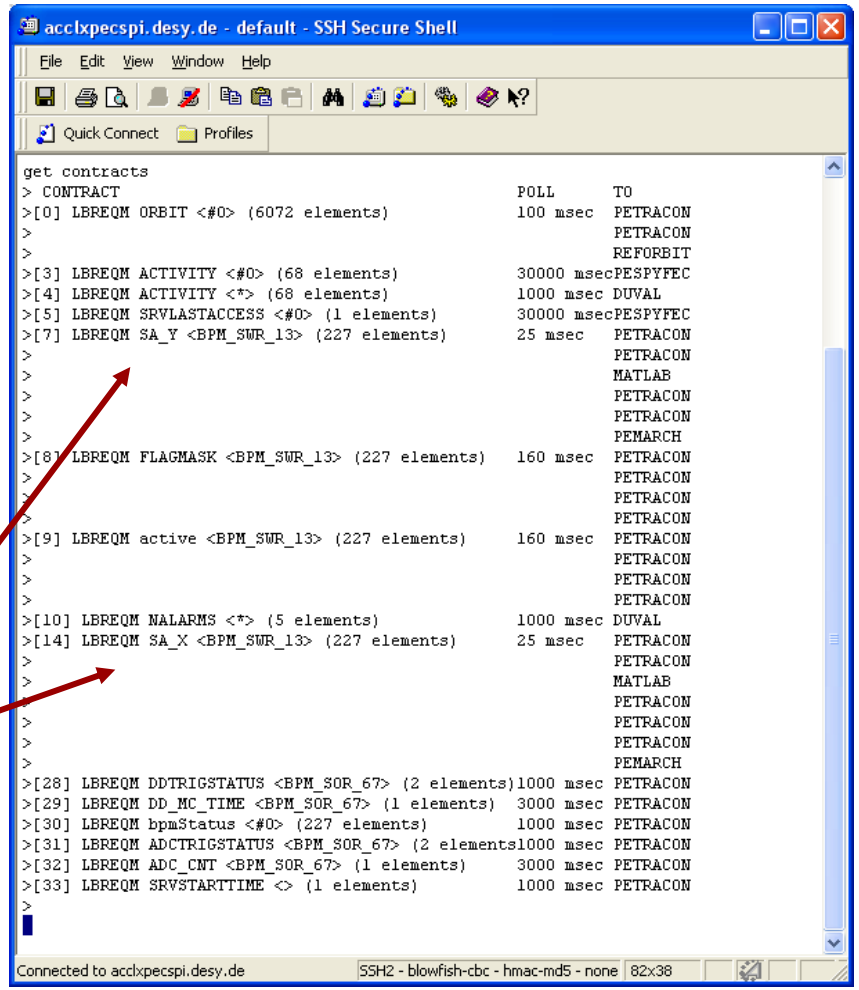
Some applications insist on acquisition ‘one at a time’!
-> dispatch gets 300 interrupts/sec instead of 1/sec.

Application Developers and TINE 4.1

- Solution in TINE 4.1:

MCA acquisition enforced!
- Handshaking returns requested array index and length of array.
- All happens beneath the API !

e.g. BPM Server has several clients getting all 227 horz. and vert. orbit positions



Hardware Engineers and TINE 4.1

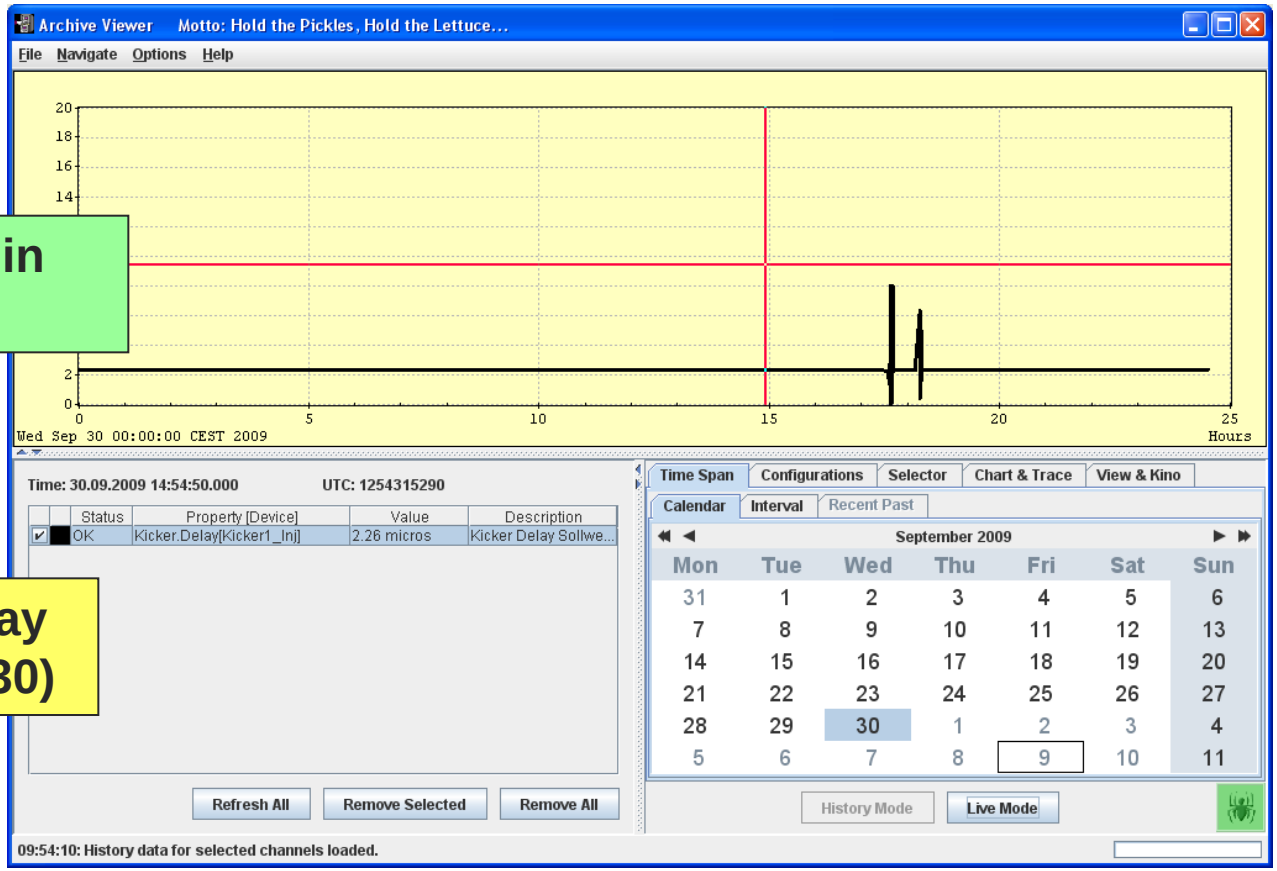
- Example of behavioral expectations:
  - TINE Archive is designed for speed !
    - Lookups
      - single channel over a time-range ~ 100µsecs/channel
      - MCA lookup at a single time ~ µsec/channel
    - Viewers
      - Use *optical zoom* with maximum 5000 data points over a time range
      - Each zoom re-acquires archive data
      - Very fast browsing !
    - BUT:
      - With this viewing strategy there's a raster!
      - Will I miss 'glitches' ?

TUP034

Hardware Engineers and TINE 4.1

Spikes appear in the afternoon

e.g. Kicker Delay Setting (Sept. 30)



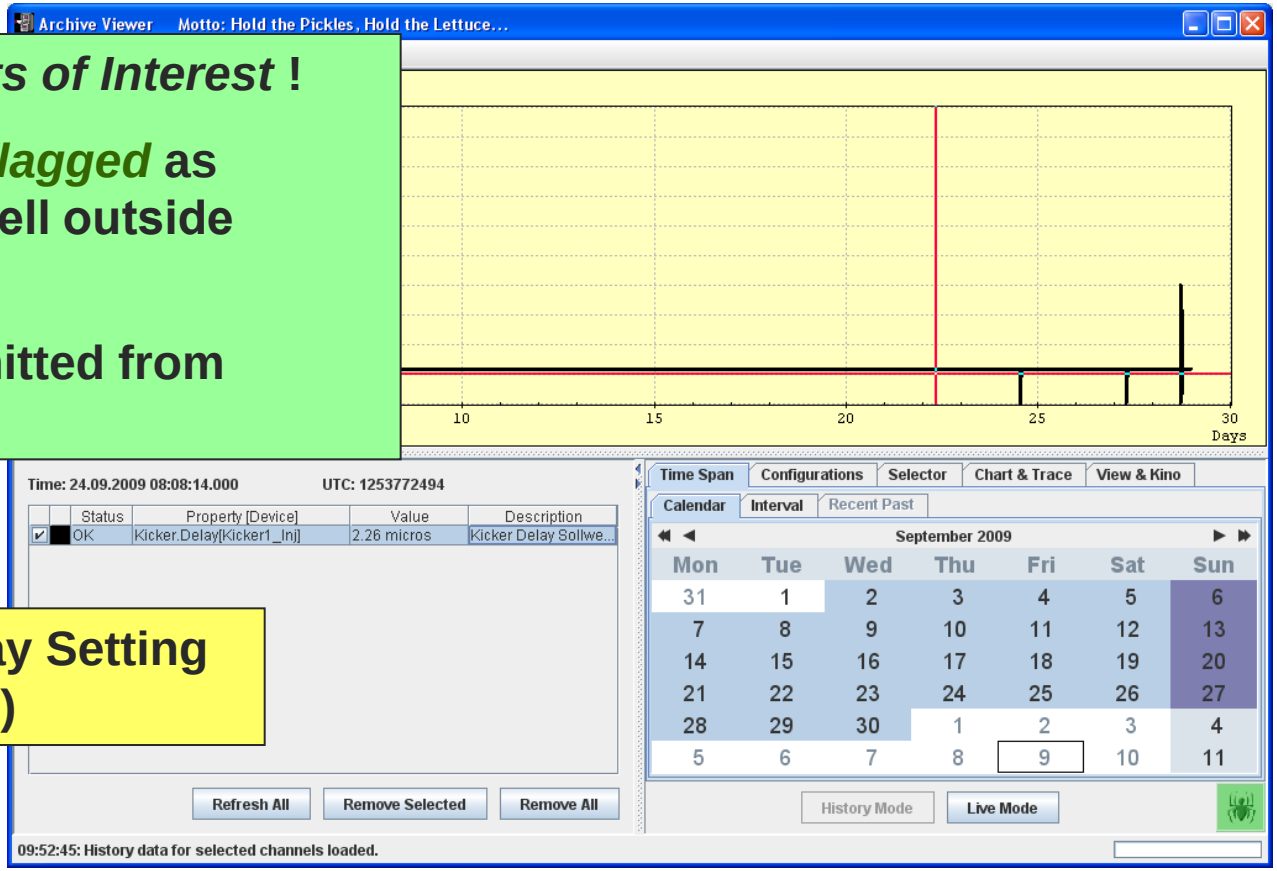
Hardware Engineers and TINE 4.1

Introduce: *Points of Interest* !

-Archived data *flagged* as '*interesting*' if well outside tolerance!

-Will NOT be omitted from archive call!

e.g. Kicker Delay Setting (month of Sept.)



Machine Physicists/Operators and TINE 4.1

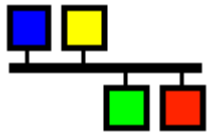
- Led to improvements in TINE Alarm System
- Expectations led to improvements in performance
  - e.g. real-time video over Gigabit ethernet should allow lossless video at 100 Mbytes/sec, right?

Interoperability and TINE 4.1

- DOOCS
  - TINE is embedded !
  - Many *impedance mismatches* found and fixed.
  - Make sure name-space, format space, etc. remain synchronized.
  - Alarms and Archives must map properly
  - Turing Test: “Is it DOOCS or is it TINE?”

Interoperability and TINE 4.1

- EPICS



  - Need fully functional mapping between TINE and EPICS
    - epics2tine runs embedded on the IOC
    - javaIOC also has a TINE interface (cosylab)
  - Understand difference between pvData and device server property access.
    - Database view vs. Property calls to a device instance.
    - Mapping is mostly straightforward
  - javaIOC
    - requirement of structures with mutable strings
    - TINE 4.1: allow TINE structures to contain variable length data types (STRING, IMAGE, SPECTRUM)

Interoperability and TINE 4.1

- TANGO
  - Generally a good fit!
  - tango2tine:
    - TANGO has no name length restrictions; TINE does. (does one worry about this? Device Names can contain up to 1024 chars)
    - TANGO classes can either map to TINE device servers or a TINE device group
  - tine2tango:
    - TINE structures aren't mapped at the moment
    - Servers with 'property-query precedence' do not map well.

Interoperability and TINE 4.1

- **STARS/COACK**
  - STARS bridge to TINE maps well
  - BUT:
    - STARS has no hierarchy limitations
      - Hierarchy beyond /context/server/device (i.e. sub-device, etc.) gets assigned to ‘device’
      - Device names such as “device/sub-device/sub-sub-device/etc” are in general not a problem for TINE.

Other Factors …

- Keeping pace with LabView
- Keeping pace with MatLab
- Keeping pace with .NET
- Keeping pace with java
- Keeping pace with Operating Systems (64bit or otherwise)
  - Subtle behavioral changes with Winsock starting with Vista!
- Etc.

The Future …

- Keep running in place toward TINE 4.2 !

Thanks for your attention …

see http://tine.desy.de *for details …*