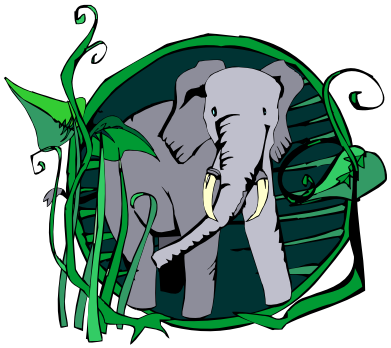
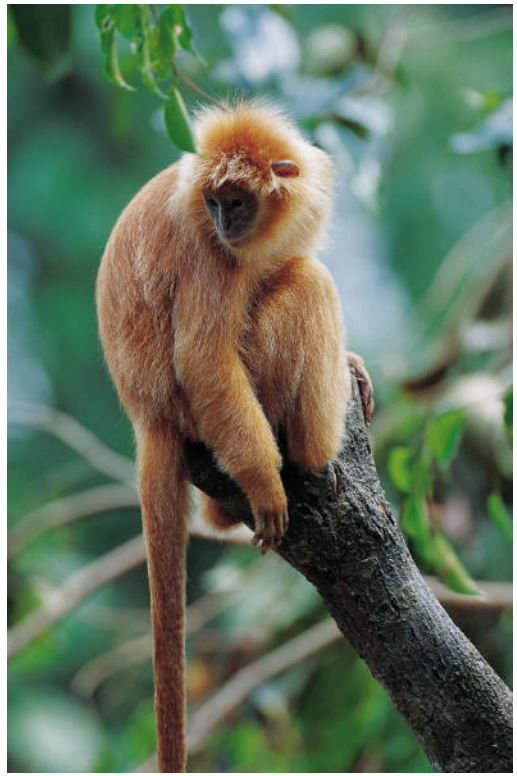
A Jungle Guide to TINE Naming:



S. Herb, MCS

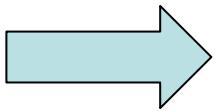
08.02.2008



Goals for TINE (&DOOCS) CS Naming

- **Expose accelerator control functionality** via human-readable object-oriented component hierarchy

- **Make everybody happy** (hardware groups, software developers, operations staff, computers).



Always a set of messy compromises;
lots of work is needed just to end up
with a useable system!

Summary of Presentation

III. Depiction of Accelerator Structure

Accelerator Structure vs. CS Structure

II. ‘Orthographic’ Rules and Conventions

EmbeddedBlanks, camel_case etc.

I. Tine Naming Primer

Address String Names / CS Structure Names

I. Tine Naming Primer

Address String **(‘external’ name for CS component functionality)**

/CONTEXT / DEVICESERVER / DEVICE / PROPERTY

CS ‘Internal’ Structure Names **(but needed for trouble-shooting!)**

hostname [points to a computer with an IP#]

FECNAME [points to a TINE process running on the computer]

equipment module TAGNAME [name for a device server within the ‘fec’]

‘export name’ [= ‘DEVICESERVER’ , points to an equipment module]

Subsystem [used for sorting into groups, but not part of name resolution]

TINE NAME SERVER Address Resolution (in TINE Nameserver)

‘/CONTEXT/DEVICESERVER’ => go to

Eqp. Module ‘TAGNAME’ in Process ‘FECNAME’ with PortOffset Y running on platform ‘hostname’ with IP# Z , listed under ‘CONTEXT’

‘fec’ vs. ‘equipment module’ [1]

Early Times: (FEC= ‘*front end computer*’, **MSDOS**, no multitasking)

FEC -- Eqp. Module1 e.g. beam position monitors => DeviceServer ‘**BPM**’
-- Eqp. Module2 e.g. beam loss monitors => DeviceServer ‘**BLM**’

Now: (WinXP, Linux …)

Host (= computer, with ≥1 FEC = ‘*front end controller*’ processes)

FEC-A – Eqp. Module1
-- Eqp. Module2

FEC-B – Eqp. Module3

⇒ Two different ways to support multiple “Device Servers” on a Host, within a single ‘fec’ process, or with with multiple processes

Recommendation: ?? It depends. Are the devices related?
Do they share a fieldbus ? ...

‘fec’ vs. ‘equipment module’ [2]

Historically (HERA) a lot of confusion was generated by mixing up the address string and CS structure names, in particular:

Hostname / Fecname / Tagname / ‘ExportName’ = Device Server /

Two possible approaches are:

1. **Make them all the same**: this works in simple cases but not in general, so end result is a mix where it holds for many cases but not for all.
2. **Make them different:** use naming / orthographic conventions to help distinguish which namespace we are dealing with.

Recommendation: **2.** (some details below)

II. Orthographic Rules and Conventions

(+ see http://adweb.desy.de/mcs/tine/TineNamingConvention.html)

- Special Characters:

Forbidden: **“/”** **“\”** **“,”** **“*”** tab newline leading & trailing blanks

Separators: **“-”** **“.”** **“_”** are encouraged (but “_” seems to be controversial)

“ “ (embedded blank(s)) strongly discouraged

Others: strongly discouraged (is “+” a borderline case ?)

Recommendation: strictly avoid all except ‘encouraged’ special characters. They may work ‘now’ but make trouble later (especially when trying to create bridges between different control systems!)

- Upper/Lower/Camel Case:

TINE names are case insensitive

Recommendation: develop guidelines and enforce them strictly to enhance readability (see below) [**Note:** **DOOCS** UPPERCASE but case-insensitive]

Orthographic Rules and Conventions [2]

- ‘Encoded Sequences’ *vs*. ‘Words’ for Naming

“acclxhebpm” = linux host for hera beam position monitors

“**/HERA/HEPhakoWr”** = device server for Hera (Protons?) + ???

vs.

/HERA/SCRAPERS

/PETRA/VAC.ION_PUMP

Recommendation:

Use **Words** or **Standard abbreviations** for the **Address string** names

Use **Encoded Sequences** for the ‘**CS Internal** Names’

This helps to

- Make the Address strings comprehensible to non-Insiders
- Keep the namespaces distinct

Conventions, (some) Examples

hostname : dns hostname or alias, lower case [accxpl2seki, mskvxw01]
may refer to group, facility, OS, functionality

fecname: encoded, UPPER case [L2SEKI , PECOOL]
includes facility, indication of functionality

tagname: (traditionally e.g. XXXMOD, BPMMOD, CURMOD)

Device Server Name: **word(s)**, UPPER case **(?)**, does **not** include Facility
old 'HPCUR' => 'CURRENT'

Device Instance Name: Camel case (but NL130, not Nl130)
words / sequence mix depends on system!
often includes device 'location' *(more later)*

Device Property: **word(s),** Camel case
don't use 'setXyz' / 'getXyz' ('Xyz' is RD, WR, or RD|WR)

III. Depiction of Accelerator Structure

Accelerator *vs*. Control System Structure

/CONTEXT / DEVICESERVER / DEVICE / PROPERTY

CONTEXT = “FACILITY” , HERA, PETRA, FLASH, etc , ok

(“FACILITY.EXTENSION” mostly for CS ‘special features’)

DEVICESERVER is intrinsically a Control System Structure => ??

Sometimes mapping is simple:

DEVICETYPE/DeviceInstance (BPM / MX.WL030 …)

DEVICE(TYPE)/Location (**DOOCS** terminology)

But there are all sorts of situations for which this doesn’t fit so easily

Accelerator Structure [2]

- Single Device per Device Server
- Many different device types per fec or host (e.g. 1 of each)
- DeviceType devices spread over many servers
- Group of similar devices with non-identical properties

A more general problem is 'Assemblies' , i.e. how do we depict a multiple level hierarchy involving components on multiple servers when only a single level within our Address string is available, ie Context/DeviceType/DeviceInstance

(In some sense sorting the system by groups of 'DeviceType' corresponds to a procedural rather than an OO view of the system)

Assemblies [1]

Example from FLASH (**DOOCS**, see **jDTool** (web start address on DOOCS site))

TTF2.MAGNETS / DeviceType / DeviceInstance ('location')

QUAD	**/ Q2UBC2**
QUAD.PILO	**/ Q2UBC2**
QUAD.MOVER	**/ Q2UBC2**
QUAD.DB	**/ Q2UBC2**

⇒ QUAD devices are composed of systems sitting on multiple servers

- Requires coordination of naming on different server types (GOOD!!)
- In this case, QUAD (but not QUAD.PILO) is a virtual device server created by redirection of servers on 11 hosts, each of which supports a mixture of quads, dipoles, etc..
- "TTF2.MAGNETS" = "**FACILITY.SUBSYSTEM**" is **DOOCS** notation, non-supported usage for TINE.
- A possibility for **TINE** is "FACILITY/SUBSYSTEM.DEVICETYPE", for example as above, "**PETRA/VAC.ION_PUMP/..**" . *[**No mcs1 decision on this yet?**]*

Assemblies [2]

Another example from FLASH (DOOCS)

TTF2.RF/ ‘DeviceType’ / DeviceInstance (‘location’)

KLY.DIO	**/ KLY2**
KLY.ADC	**/ KLY2**
KLY.INTERLOCK	**/ KLY2**
KLY.CONTROL	**/ KLY2**
KLY.PLC	**/ KLY2**

- This is (I think) implemented very differently from the magnet case, namely each klystron has its own process (~fec), which supports a dio, adc, plc, etc, and there are multiple instances of the process. I.e many different devices in one process.
- But the same notation is used to show the Assembly structure

Recommendation: Use **“.”** separator to create ‘Assemblies’ or show hierarchy within DeviceType (and Properties !!)

Recommendation: Coordinate Device Type and Instance names across servers to support Assembly views of components.

[Corollary] : All Device Instance names should use the same system for describing position within the accelerator!! (*Petra3: in progress*)

‘Parallel’ Servers

Example from HERA: Transient Recorders, SPS for **QUENCH** system in the Hera Halls
/HERA/ WESTTR, NORDTR, OSTTR, SUEDTR , ie same Device Types per server

How might we do this differently now?

/HERA/**QUENCH.TRANSREC-W**/
/HERA/**QUENCH.TRANSREC-N**/
/HERA/**QUENCH.SPS-W**/ [SPS-W]
/HERA/**QUENCH.SPS-N**/ [SPS-N]

TINE Redirection (optional) could then result in virtual servers
QUENCH.TRANSREC/ TREC-W, TREC-N …
QUENCH.SPS/ SPS-W, SPS-N …

Recommendation: use hypenated extension to differentiate parallel servers

Recommendation: use Redirection to consolidate parallel servers IF
it results in a significant simplication of the visible namespace
(there is also a price in complexity and possible confusion)

Conclusions

- Using TINE (or DOOCS) Servers does not in itself give any guarantee that the resulting CS name structure is coherent or human readable.

- TINE (and DOOCS) naming does **not** naturally mirror the accelerator structure. A mixture of naming conventions and 'tricks' is necessary to produce a useable result. The 'tricks' make the internal structure of the system more complicated, so there are trade-offs in how far they are used.

- Coordination of conventions and name usage both within and between groups is essential. This of course requires published guidelines, many examples, AND extensive consultation between the groups.

- **What's Needed !?**

-- Published guidelines and examples from MCS1

-- Decisions, e.g. "Include Subsystem, as VAC.ION_PUMP : yes, no maybe?"

-- Consultation / Review mechanisms?

Addendum: a wee bit of advice

- DOOCS/FLASH people have devoted significant effort to developing naming which reflects a view of the accelerator.

- The jDTool shows tree diagrams of the system hierarchies. Use it to look at their solutions for some of these issues

  http://ttfinfo.desy.de/ttf_apps/jClients/jDTool.jnlp

- If there is a FLASH solution parallel to your needs, and it is not a worse fit than other alternatives, consider adopting it!

- FLASH and PETRA will be operated from the same control room by the same operations staff. It is foolish not to exploit commonalities