Common Templates and Organisation for Beam Instrumentation Front End Software Upgrade

Stephen Jackson

CERN SL / BI Software

16th October – PCaPAC 2002 Frascati, Italy

Who am I?

- Computer scientist from England
- First arrived at CERN in 1995
- Involved in many different projects for beam instrumentation
  - SPS (Orbit, Trajectory)
  - LEP (Bunch measurement :Cameras, Telescopes, etc)
  - Transfer Lines (Trajectory from PS->SPS->LEP)
- Developer of control & data extraction software for beam instrumentation

A bit of history…

- CERN relies heavily on software
- Many instrumentation systems created
  - By permanent & *temporary* personnel such as Fellows, PhD students etc
  - Using languages such as Fortran, Modula 2, C, C++ etc
- CERN is a research environment
  - Resulting in *state-of-the-art* systems
  - But…
  - When personnel leave, their expertise leaves too
  - … leaving the rest of the team to pick up the pieces!

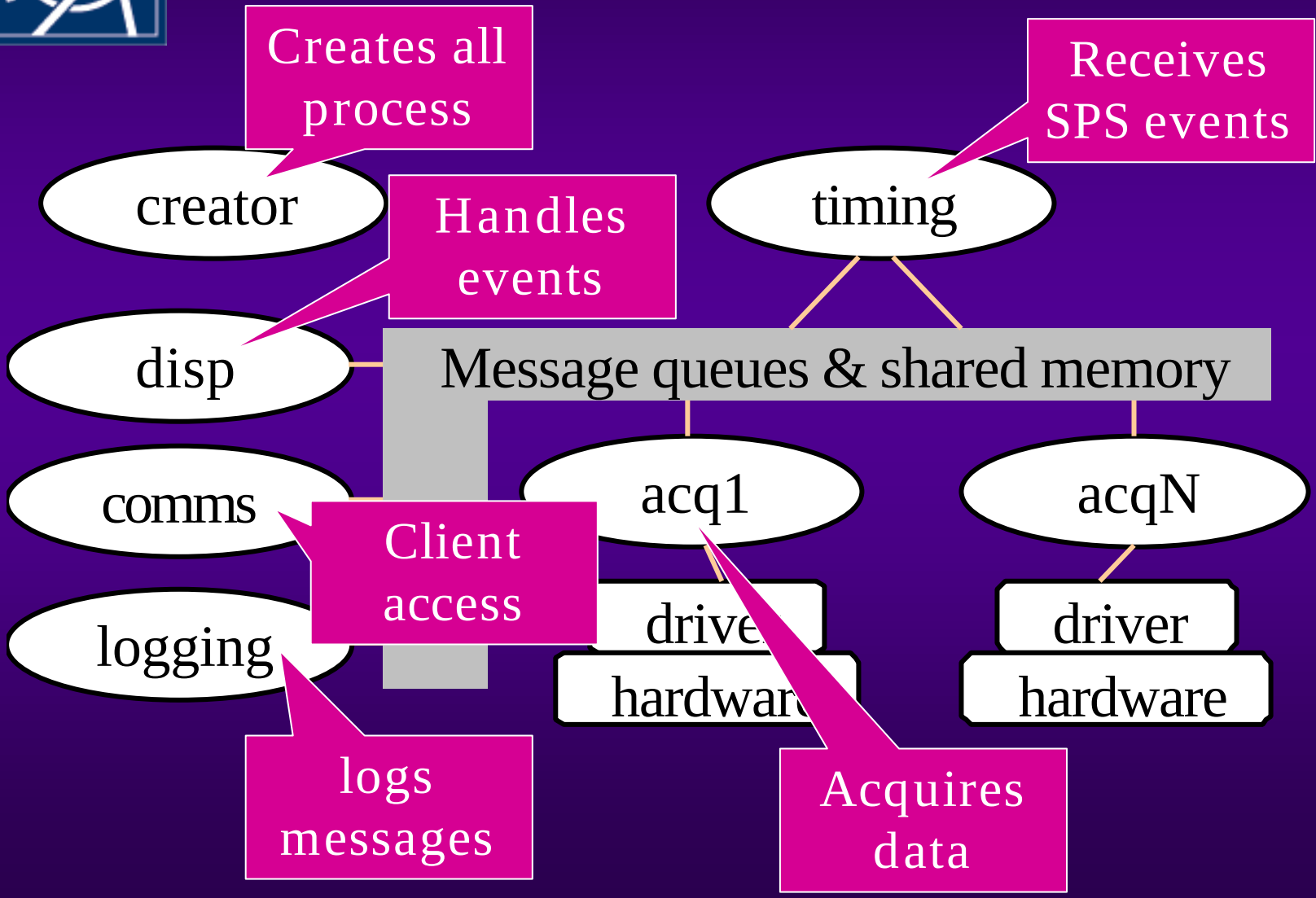
LHC on the horizon…

- With the advent of LHC, many systems need (re) development
- Perfect opportunity to standardise before the production of 50+ new systems
- The BISCoTO project was born
  - Beam Instrumentation Software Common Tools & Organisation

BISCoTO's aim…

- Combine the skills of the development team to create a standard real-time template
  - Capable of satisfying all the needs of the team
  - Avoiding instrument specifics – no redundancy
  - Data driven so minimal changes required for a system
- Create generic tools for use by ALL BISCoTO developers / users
  - Only possible if data driven

BISCoTO's real-time



Data driven

- All processes are data driven to avoid divergence from the template
  - Instrument specific code ‘include’d at defined places
- Definition name-space
  - Class.Process.Definition
  - Many definitions for many processes for 1 Class of instrument
- Data name-space
  - Class.System.Qualifier.Process.Definition
  - A class of instrument has many systems, ordered by qualifiers

The Config tool

Class.Process.Definition

Defintition entry

CLASS TTPOS PROCESS Shared DEFINITION AcqConf MAX_INSTANCES 256 DESCRIPTION ? VERSION 6

Field Idx	Field Comment	Field Type	Field ID	Size/Value	Viewable by	Editable by	New group
1	Bisc: Name of the Reload Data...	CHAR_A			SWExp		
2	Bisc: Reload Conf....	INT			SWExp		
3	...	INT	acqConfIndex	1	SWExp		
4	Bisc: Counter incremented by u...	INT	refreshCou	1	SWExp		
5	Bisc: Last 'critica ange time (...	CHAR_ARRAY	refreshTime ng	80	SWExp		ALL
6	Bisc: Event Sour (0 -> UsrCm...	INT	eventSourc	1	SWExp	WExp	ALL
7	Bisc: Cycle Detailed Description.	CHAR_ARRAY	cycleDesc	80	SWExp	SWExp	ALL
8	Bisc: Human Cycle Name (Coul...	CHAR_A RAY	cycleName	32		SWExp	ALL
9	Bisc: Human Sequence Name (...	CH RAY	mtgSequenceName	32			LL
10	Bisc: MTG cycle		mtgCycleId	1			LL
11	Bisc: Cycle Type		cycleType	1			LL
12	Bisc: Cycle Num		cycleNumber	1			LL
13	Bisc: Will allow to have a set of ...	INT	usrCmdId	1	SWExp	SWExp	ALL
14	Bisc: Will allow to have a set of ...	INT	bstCmdId	1	SWExp	SWExp	ALL
15	Bisc: Automaticaly set by the ac...	INT	acqMemOffset	1	SWExp	SWExp	ALL
16	Bisc: Depth requested for this c...	INT	acqMemSizeReq	1	SWExp	SWExp	ALL
17	Bisc: Depth obtained.	INT	acqMemSize	1	SWExp	SWExp	ALL
18	Bisc: State of the Cycle/Cmd (1 -...	INT	cycleState	1	SWExp	SWExp	ALL
19	Bisc: Requested state for this cy...	INT	cycleReqState	1	SWExp	SWExp	ALL

Comment

Field ID

Field exposure

Field Type

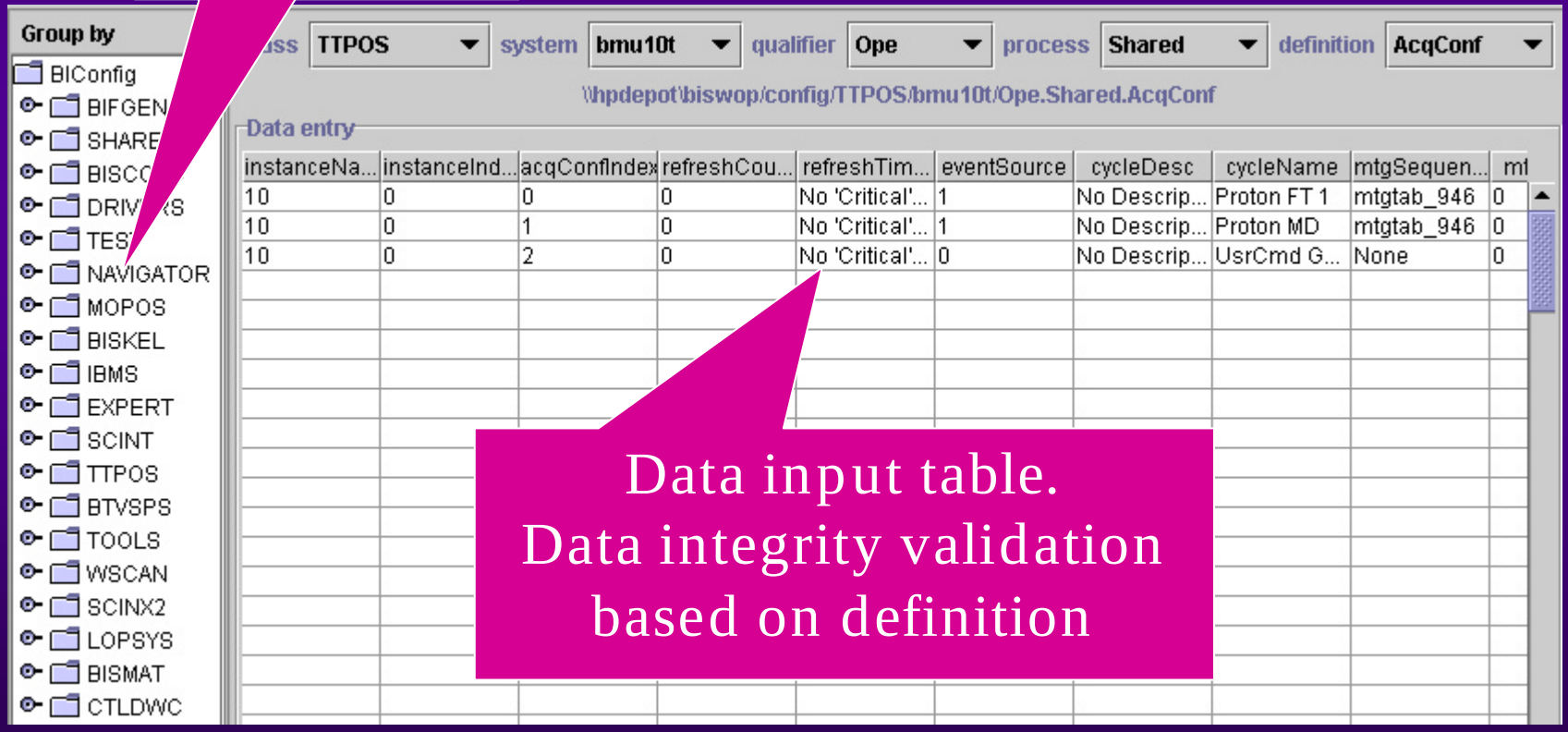
Optional Size / Value

Mapping C to Java

- **Server written in C, Clients written in Java**
  - **Different data types, so BISCoTO data types limited**
    - No unsigned types for example
  - **C structs don't map to Java classes under current middleware**
    - May change with Corba's data marshalling… in 1-2 years
- **Config tool automatically creates C code & Java Classes for handling data**
  - **Generated Java class translates C struct into Java bean**
  - **Generated C code allows anonymous manipulation of server data in files and memory with run-time 'where clauses'**
  - **All code regenerated automatically when a definition changes**

Generic data input tool

Data file selector

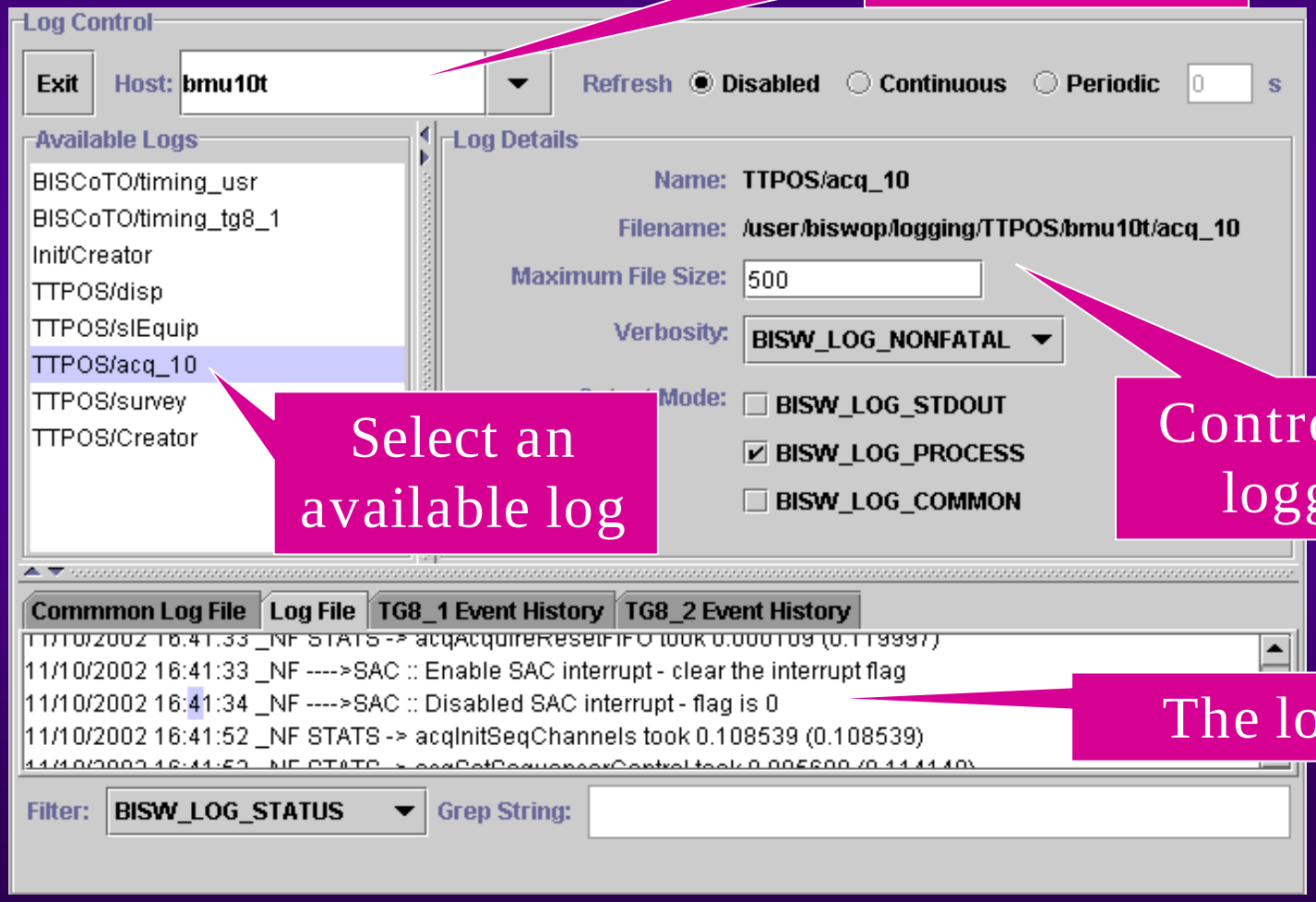


Data input table.
Data integrity validation based on definition

Generic Debugging

- Logging information is vital when a system has problems
- BISCoTO uses a separate logging process to handle this task
  - Performs the time-consuming task of file writing
  - The real-time tasks simply request a log
  - Logging is fast (<1ms) and reliable
- Logs viewed through another generic GUI
  - Allowing viewing and remote control of logs…

Logging GUI


Select host to interrogate
Log Control
Exit
Host: bmu10t
Refresh
Disabled
Continuous
Periodic
0
s
Available Logs
BISCoTO/timing_usr
BISCoTO/timing_tg8_1
Init/Creator
TTPOS/disp
TTPOS/slEquip
TTPOS/acq_10
TTPOS/survey
TTPOS/Creator
Log Details
Name: TTPOS/acq_10
Filename: /user/biswop/logging/TTPOS/bmu10t/acq_10
Maximum File Size: 500
Verbosity: BISW_LOG_NONFATAL
Mode:
BISW_LOG_STDOUT
BISW_LOG_PROCESS
BISW_LOG_COMMON
Select an available log
Control the logger
Commmon Log File
Log File
TG8_1 Event History
TG8_2 Event History
11/10/2002 16:41:33 _NF ---->SAC :: Enable SAC interrupt - clear the interrupt flag
11/10/2002 16:41:34 _NF ---->SAC :: Disabled SAC interrupt - flag is 0
11/10/2002 16:41:52 _NF STATS -> acqInitSeqChannels took 0.108539 (0.108539)
The log
Filter: BISW_LOG_STATUS
Grep String:

Server exploration

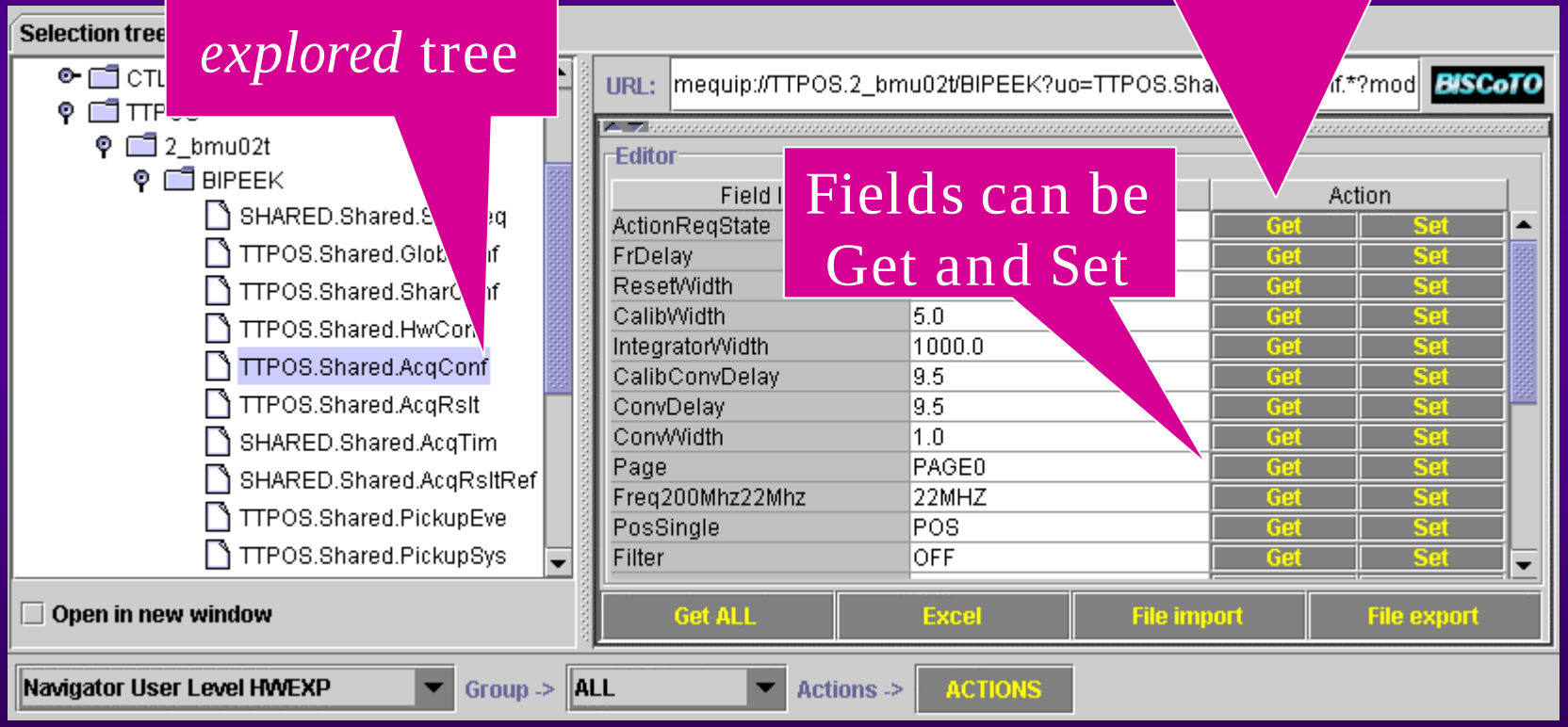
- Developers and users need to interrogate a BISCoTO server's properties
  - Server template is pre-programmed to allow 'definition' exploration
- The Navigator can *explore* definitions within a BISCoTO server given its location
  - All definitions made in the configuration GUI available
  - Uses the automatically generated Java beans
  - Involves NO effort from the developer – free!

The generic Navigator

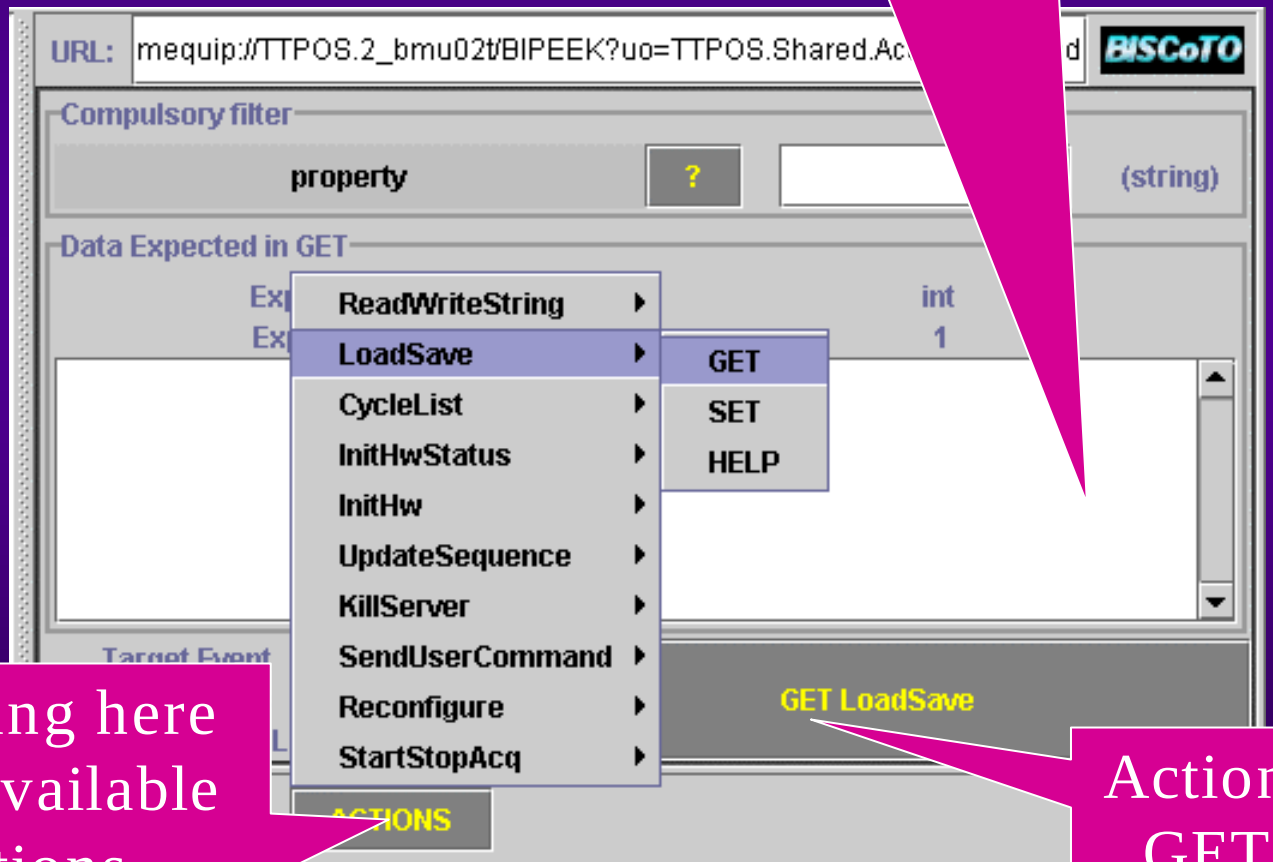
Select the definition in *explored* tree

Double clicking a definition loads the Java Bean's editor

Fields can be Get and Set



Navigator also triggers actions…



Other advantages

- The tools are generic so…
  - Enhancements made to tools available to all systems
  - Heavy use of the tools, means bugs eradicated quickly
- Automatic documentation of systems
  - API for all BISCoTO systems is automatically documented in a web page using comment, type, exposure etc.
- A lot of data is shared across all systems
  - So some configuration can be centrally maintained

A BISCoTO system for instrument XYZ

Pre-built Real-Time

Developer Input

Generated Code

Template modified with XYZ as Class

Included User Code

User Code For Drivers

Code for h/w access based on definition

Generic BISCoTO processes

BISCoTO Definitions

XYZ C lib

XYZ Beans

Free Tools

Definition Editor

Config Editor

LogViewer

Navigator

Conclusion

- BISCoTO aimed to create a standardised template, maintainable by all…
- By making the template data-driven, the integrity of the template is maintained.
- A data-driven system can benefit from generic tools
  - For configuration
  - For log control and management
  - For testing and basic interrogation

… Conclusion

- Things such as middleware and data management are commonly maintained
- Anybody in the team can diagnose another's system
- The systems are automatically documented by their definitions
- A fully operational server can be built with minimal code VERY QUICKLY!
- It works!
  - 6 systems already successfully use BISCoTO