A Generic Framework for implementing TTCN-3 Logging Modules

Anthony Baire, Radu Muresan, César Viho

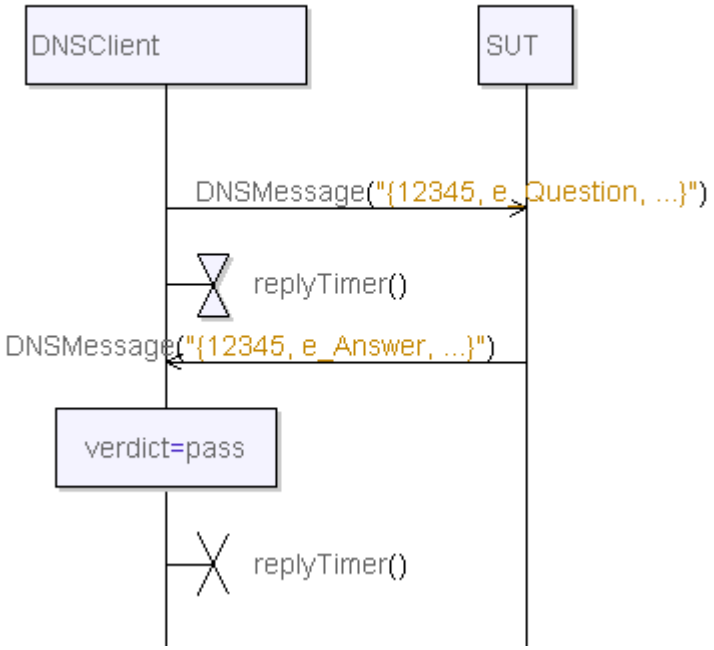
UMR IRISA

cnrs UNIVERSITÉ DE RENNES 1 INSA RENNES ENS CACHAN BRETAGNE

November 2009
TTCN-3 Asia User Conference

TTCN-3 and Test Logging

- The language is well- formalised
  - most of available tools provide a generic logging module
  - test developers do not have to implement logging



Why do we need specialised logging modules ?

- Some examples:
  - to describe the content of the messages
  - to use a specific presentation format
  - to reconstruct a session from different packets

Example 1: describing a message
Existing solution in LibSip

```
type record Request
{
  RequestLine   requestLine,
  MessageHeader msgHeader,
  MessageBody   messageBody optional,
  Payload       payload optional
}
```

Generic type definition for a SIP Request

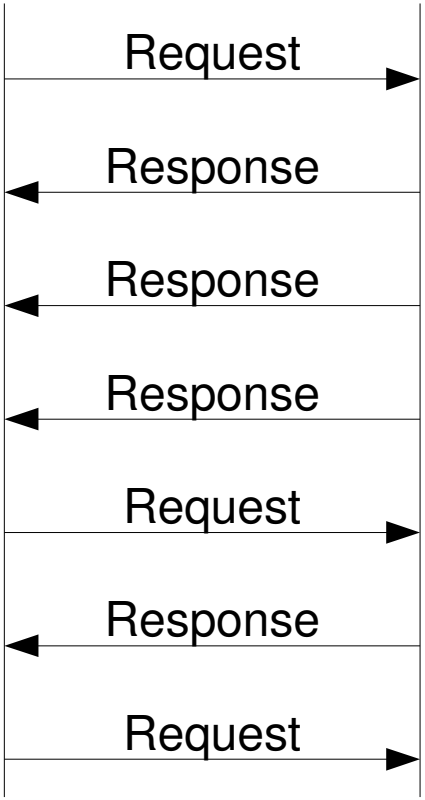
```
type record REGISTER_Request
{
  RequestLine   requestLine,
  MessageHeader msgHeader,
  MessageBody   messageBody optional,
  Payload       payload optional
}

type record INVITE_Request
{ ... }

type record OPTIONS_Request
{ ... }

...
```

14 similar definitions for each variant of SIP Request

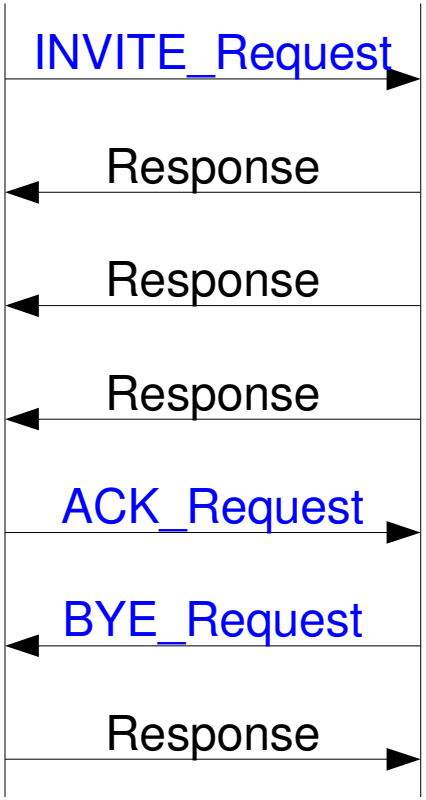
« the introduction of the specific types is to enable better means for logging »

Example 1: log traces

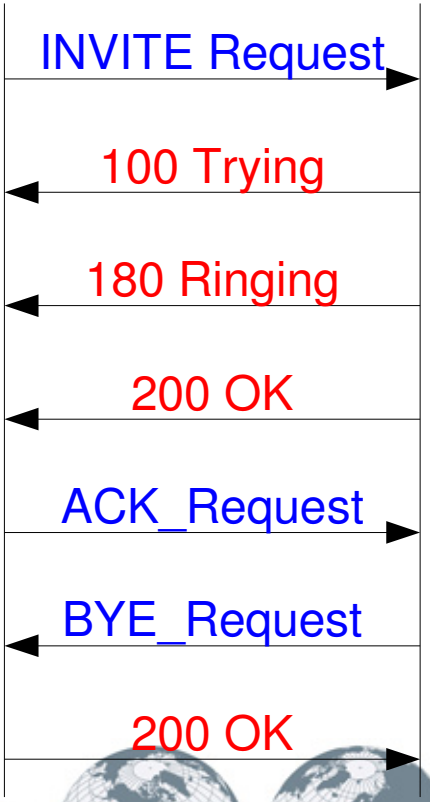
using a generic type



using multiple types (workaround)



our objective



Example 2: presentation format

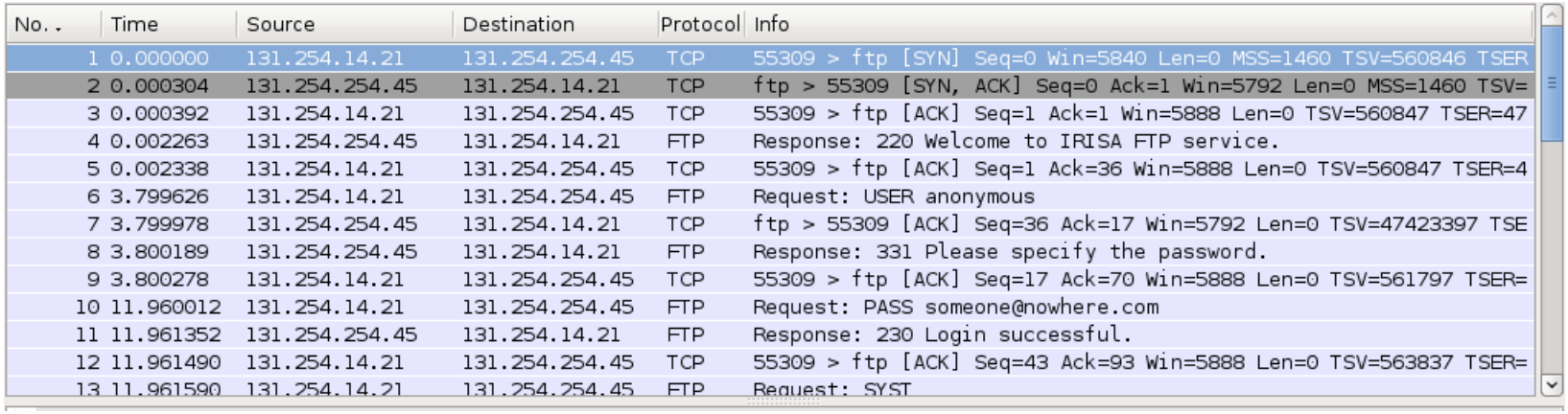
- How to represent an IP Address in TTCN-3 ?
  - octetstring → 'C0A8002A'O
  - bitstring → '1100000
    0000000
  - integer → 3232235562
  - charstring → "192.168.0.42"
  - record of → {192, 168
  - record → {a:=192,

Preferred format for **manipulating** IP addresses

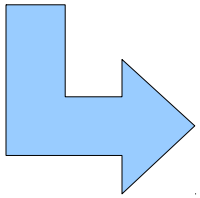
Preferred format for **presenting** IP addresses

Example 3: reconstruct a conversation

- A useful feature *wireshark* provides



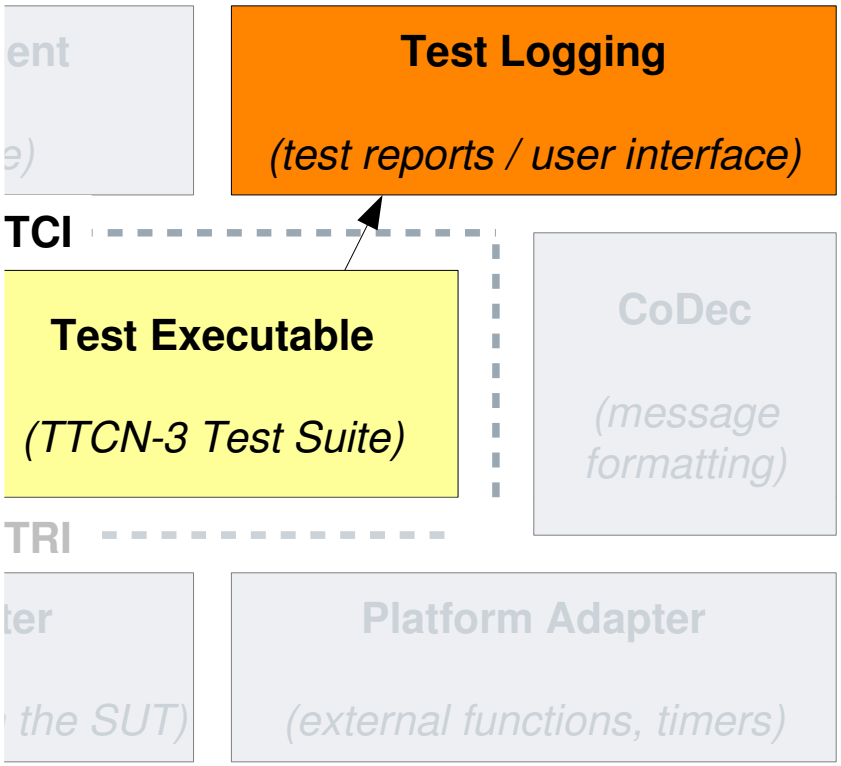
No.	Time	Source	Destination	Protocol	Info
1	0.000000	131.254.14.21	131.254.254.45	TCP	55309 > ftp [SYN] Seq=0 Win=5840 Len=0 MSS=1460 TSV=560846 TSER
2	0.000304	131.254.254.45	131.254.14.21	TCP	ftp > 55309 [SYN, ACK] Seq=0 Ack=1 Win=5792 Len=0 MSS=1460 TSV=
3	0.000392	131.254.14.21	131.254.254.45	TCP	55309 > ftp [ACK] Seq=1 Ack=1 Win=5888 Len=0 TSV=560847 TSER=47
4	0.002263	131.254.254.45	131.254.14.21	FTP	Response: 220 Welcome to IRISA FTP service.
5	0.002338	131.254.14.21	131.254.254.45	TCP	55309 > ftp [ACK] Seq=1 Ack=36 Win=5888 Len=0 TSV=560847 TSER=4
6	3.799626	131.254.14.21	131.254.254.45	FTP	Request: USER anonymous
7	3.799978	131.254.254.45	131.254.14.21	TCP	ftp > 55309 [ACK] Seq=36 Ack=17 Win=5792 Len=0 TSV=47423397 TSE
8	3.800189	131.254.254.45	131.254.14.21	FTP	Response: 331 Please specify the password.
9	3.800278	131.254.14.21	131.254.254.45	TCP	55309 > ftp [ACK] Seq=17 Ack=70 Win=5888 Len=0 TSV=561797 TSER=
10	11.960012	131.254.14.21	131.254.254.45	FTP	Request: PASS someone@nowhere.com
11	11.961352	131.254.254.45	131.254.14.21	FTP	Response: 230 Login successful.
12	11.961490	131.254.14.21	131.254.254.45	TCP	55309 > ftp [ACK] Seq=43 Ack=93 Win=5888 Len=0 TSV=563837 TSER=
13	11.961590	131.254.14.21	131.254.254.45	FTP	Request: SYST



Stream Content

```
220 Welcome to IRISA FTP service.
USER anonymous
331 Please specify the password.
PASS someone@nowhere.com
230 Login successful.
SYST
215 UNIX Type: L8
CWD /pub/OpenBSD
250 Directory successfully changed.
PASV
227 Entering Passive Mode (131,254,254,45,76,191)
LIST
150 Here comes the directory listing.
226 Directory send OK.
TYPE I
200 Switching to Binary mode.
PASV
227 Entering Passive Mode (131,254,254,45,74,144)
RETR README
```

How to implement a logging module for a TTCN-3 test suite?



- Use the TCI-TL *(TTCN-3 Control Interface)*
  - a standard interface
  - it reports all events in the test execution *(eg. message sent)*
  - usable in C/Java/XML

Implement the TCI-TL interface... at what cost?

- 100+ functions to be provided by the TL module
- ~9 parameters in each function
- Flat design: one event → one TL function
- TCI-TL is still evolving

→ a huge task... and this just to support the TL interface

The TL Dilemma

- Most of the tasks of a TL module are generic
  - display the events, draw sequence charts
  - load/save the logs on the disk
  - provide a GUI

- a tiny part in the TL is specific to the actual test suite
  - describe the content of the messages
  - present a type in a specific format

→ choosing between a generic logger or a home-made logger is an **all-or-nothing** situation

Objectives for our TL Framework

- provide a generic representation for TL events
  - easy to define
  - easy to update (futures changes in the standard)
  - easy to browse (without knowing the their structure)
- provide means for storing the logs
- allow to implement new back-ends independently

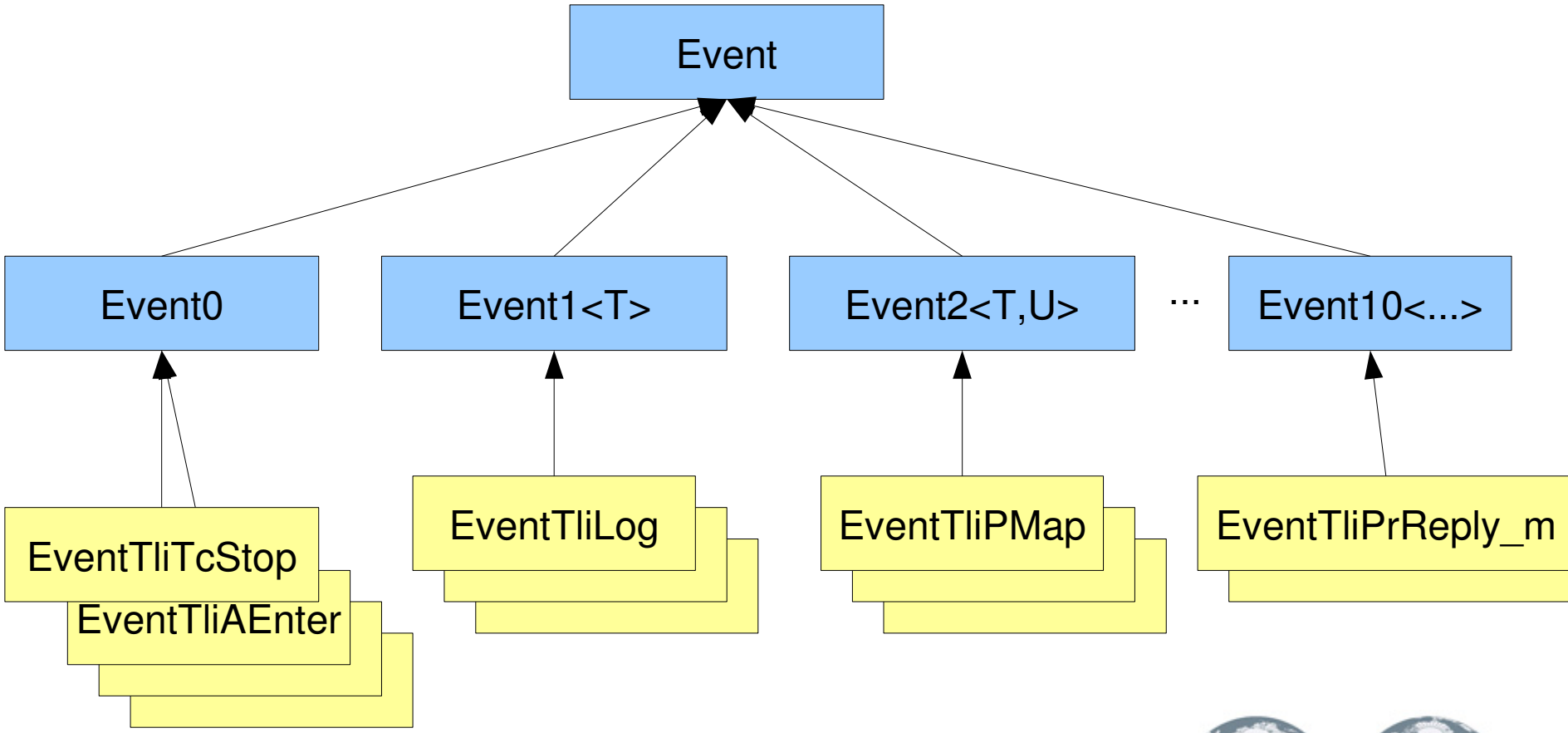
Example of TCI-TL functions

```
void tliTcStop  (String am, long int ts, String src,
                 long int line, TriComponentId c)

void tliLog     (String am, long int ts, String src,
                 long int line, TriComponentId c,
                 String log)

void tliPMap    (String am, long int ts, String src,
                 long int line, TriComponentId c,
                 TriPortId port1, TriPortId port2)
```

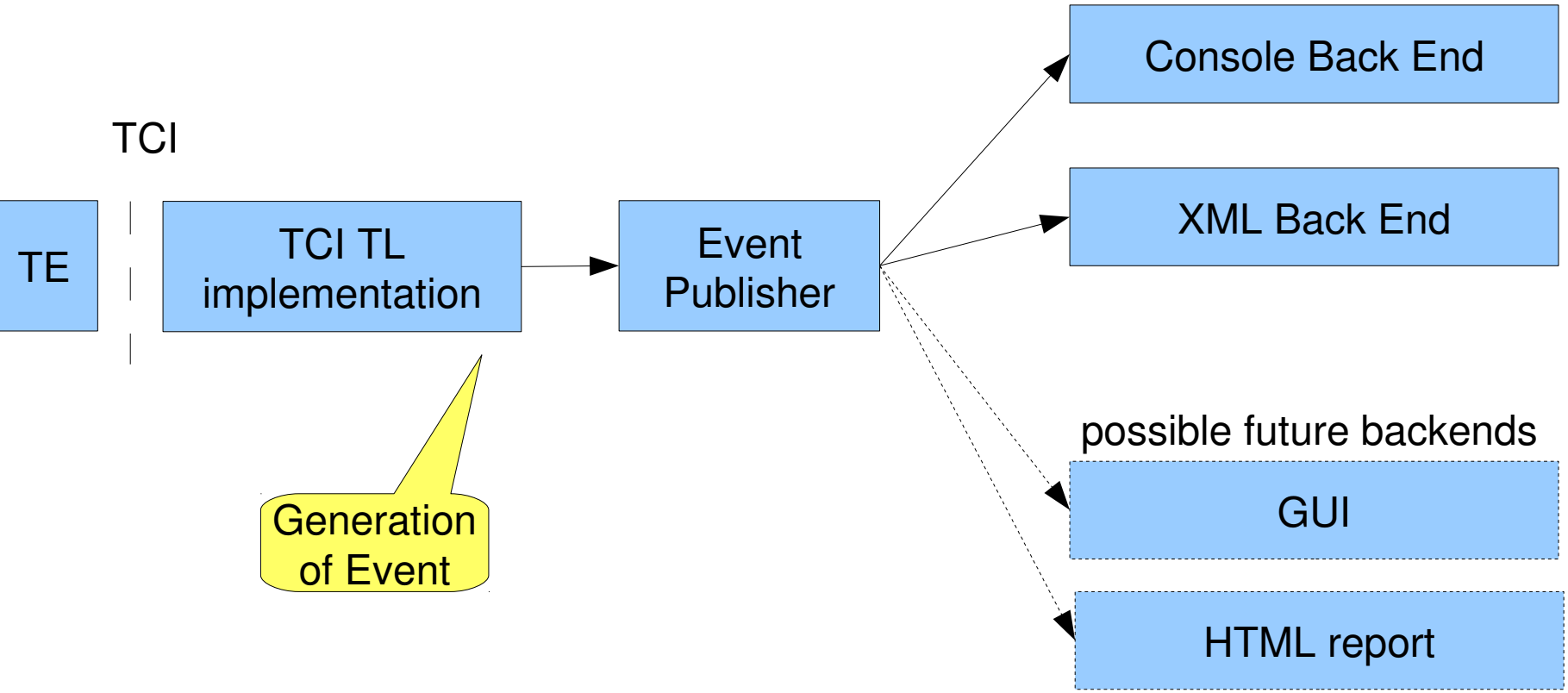
common parameters

event-specific parameters

Representation of Events



TL Module layout



Advantages of the framework

- A generic *Event* format
  - Backend implementation is easier
    - no need to know about all possible events
    - insensitive to future changes in the TCI standard
- XML format for storing event
  - standard format → good interoperability
  - ability to convert old log files to future TL formats using XSL transformations → good durability

Future tasks

- Define interfaces to allow :
  - implementing specific presentation formats
  - analysing the messages and generating a description
- Implement a GUI

Conclusion

- Specialised Test Loggers are useful
- TCI-TL does not facilitate the development of modular & reusable TL modules
- The proposed framework aims to solve this issue
  - Modular design
  - Insensitive to future API changes
  - Open source

Questions ?

Contacts: abaire@irisa.fr / viho@irisa.fr

The logging framework will be distributed in future releases of T3DevKit

→http://t3devkit.gforge.inria.fr/