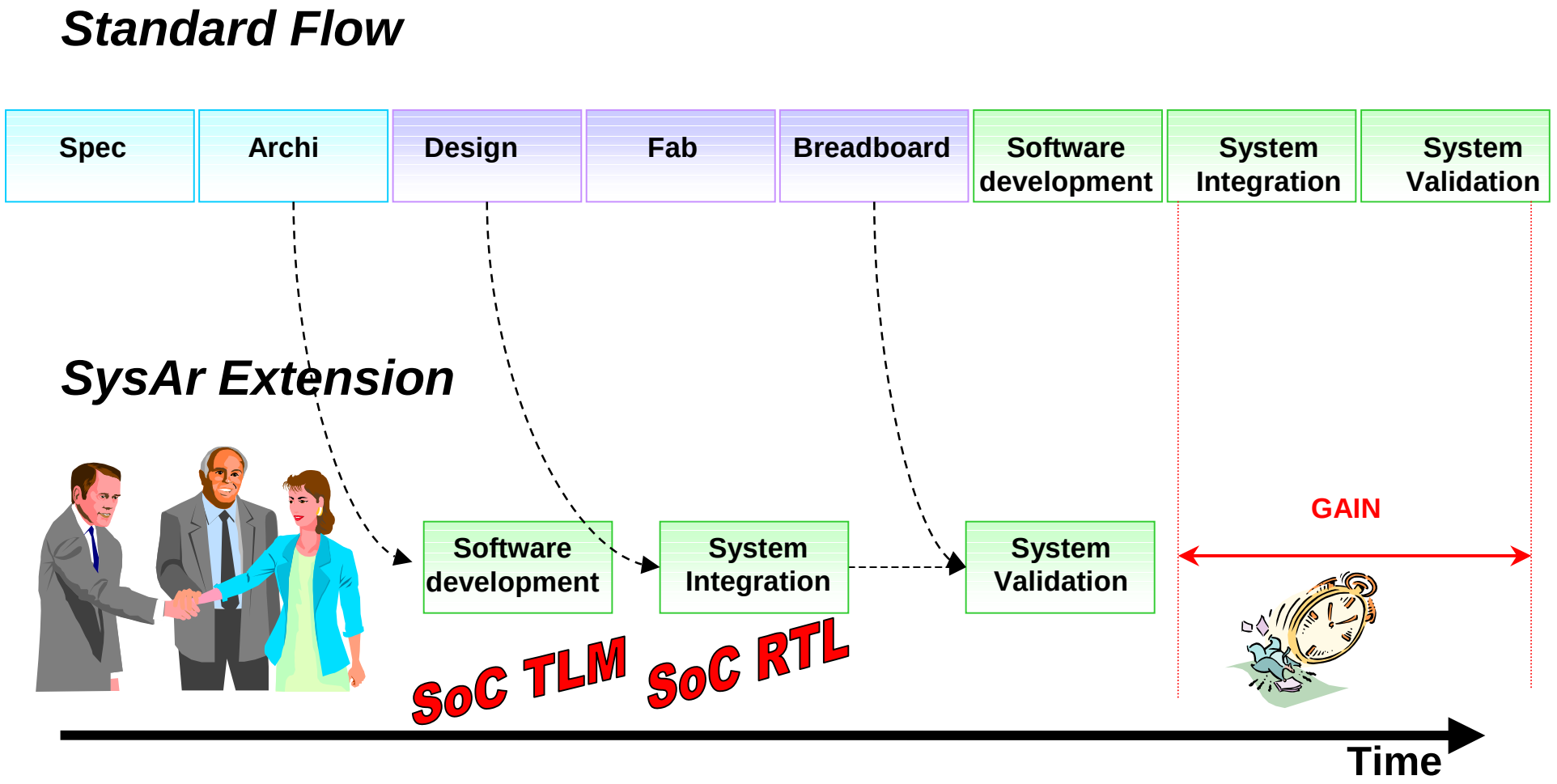
# Experiences and Challenges of Transaction-Level Modelling with SystemC 2.0

Alain CLOUARD
STMicroelectronics
Central R&D – Crolles (Grenoble, France)

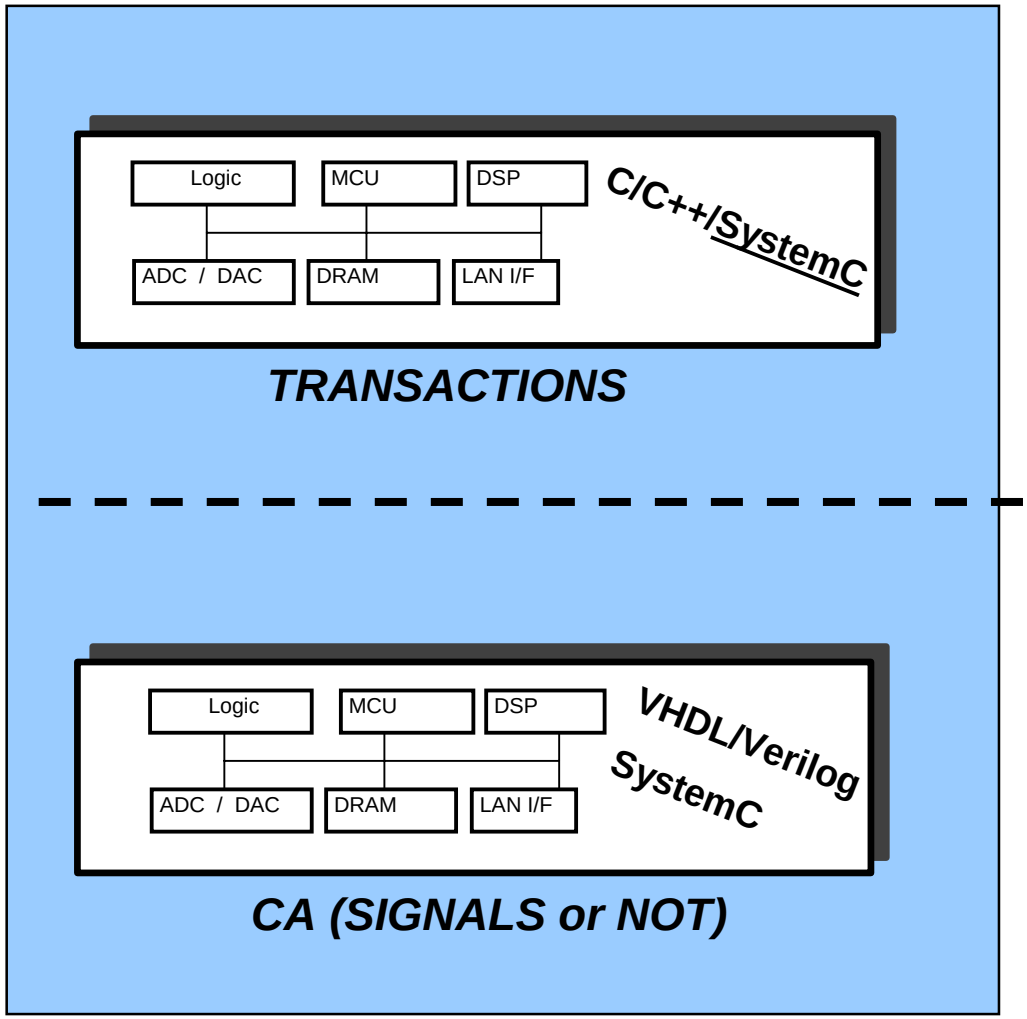
# TLM is useful – SoC HW/SW design flow



# Users of SoC TLM Platforms

- Target users
  - SW developers (drivers/firmware/appli)
  - SoC Architects
  - Technical marketing
- Enabled activities
  - Functional software development
  - Early architecture evaluation
  - (Reuse of test data)

# TLM – a concept, not a language




**TLM - SoC Architecture**

***UNTIMED***

***& TIMED***

**SoC Micro-Architecture**

***CYCLE-ACCURATE***

# TLM Positioning

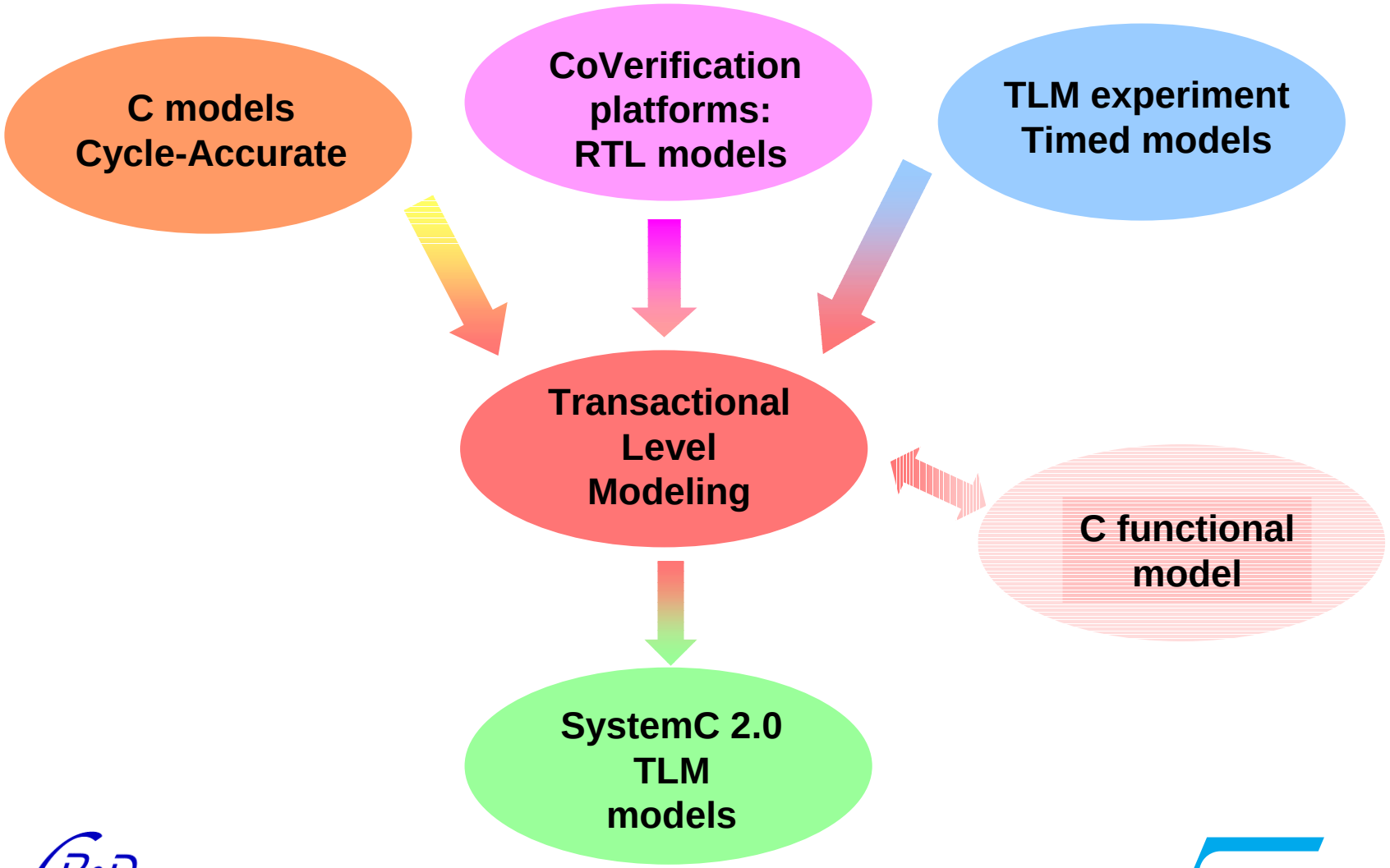
| | RTL | SysC CA | TLM (incl SysC) |
|---|---|---|---|
| Speed (Hz) | 1-10 | 10k | 50k – 500k |
| Effort ratio | 1 | /3 | /10 – /50 |

SystemC: free public domain simulator
Numbers provided as rule of thumb…

# Defining a TLM approach

- Need to formalise abstraction level
  - Semantics
  - Ex. System scheduling occurs at transaction boundaries or by explicit synchronisation
- Then define communication / synchro scheme
  - Set of APIs with defined behaviour
  - Avoid current proliferation of TLM dialects
- *Then* develop productivity tools

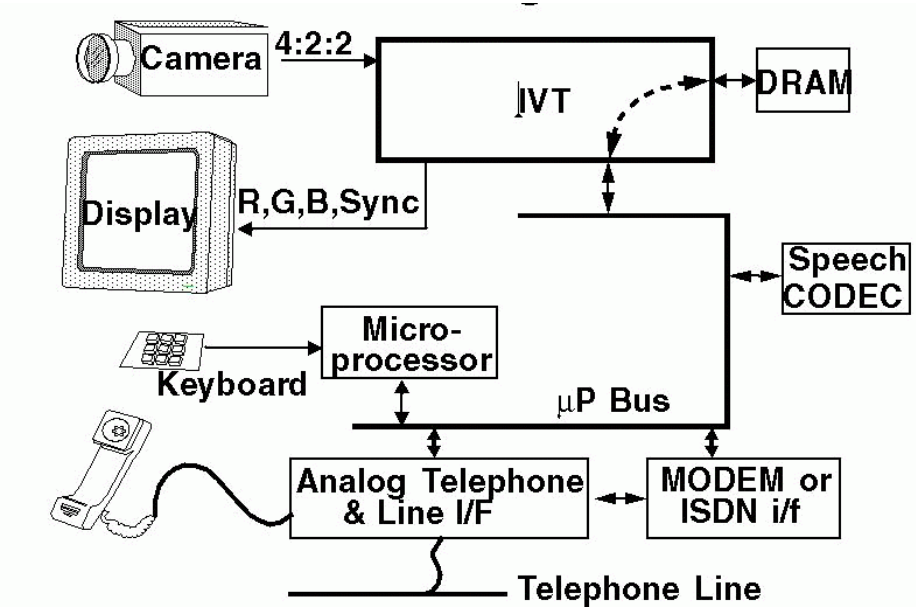
# Background


C models
Cycle-Accurate
CoVerification
platforms:
RTL models
TLM experiment
Timed models
Transactional
Level
Modeling
C functional
model
SystemC 2.0
TLM
models

# TLM Platform – Our case

- SoC Model
  - Developed by HW provider
  - No need to model cycles and bus protocols
  - Bit-true functional model of peripherals – reg. etc
  - Transactions to model communications
    - Data exchange between 2 system synchronizations
- Reuse of existing C models from architects
- Becomes SoC golden model for RTL

# MPEG4 SoC *Transactional* Model




## Objective

Enable application software development *concurrently* with HW design

## Users

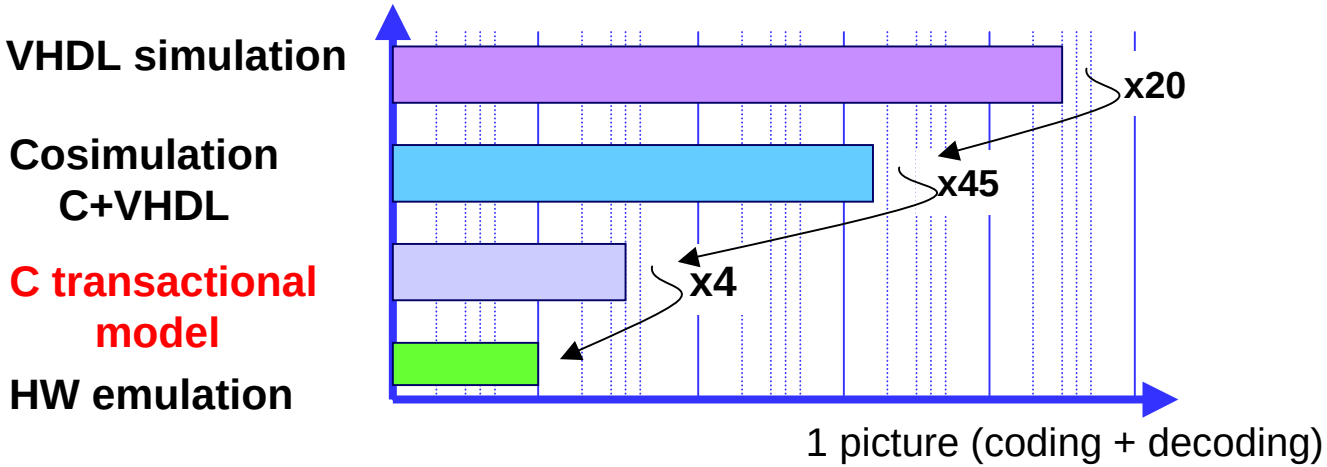
**Used by MPEG4 IVT team for software development 6 months before RTL top netlist ready**

**Used by SoC team for ARM software development**

# MPEG4 SoC *Transactional* Model

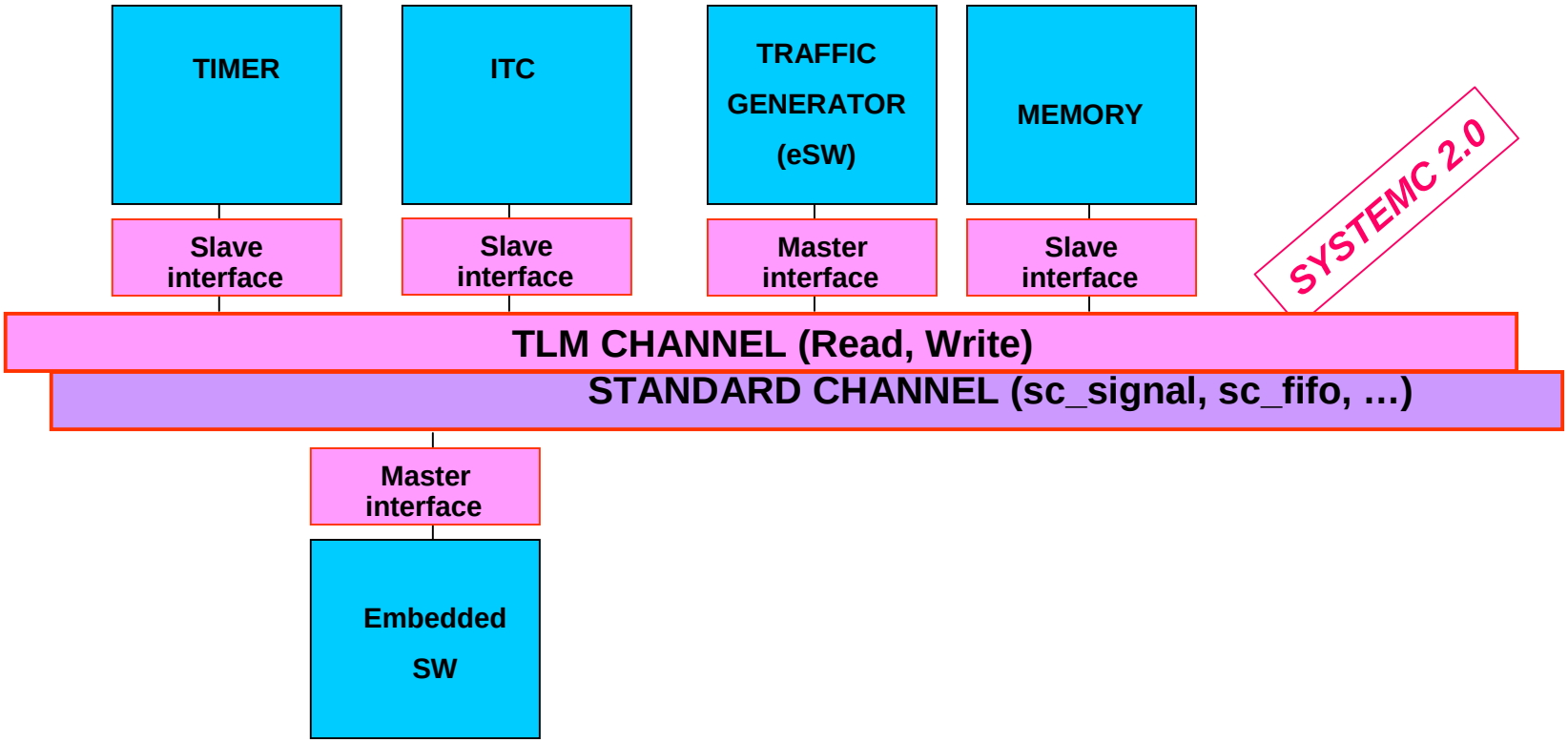
## Key benefits

- Close to emulation speed
- Debug facilities
- Fast development (days to few mm)




Simulation time for 1 image (coding + decoding)

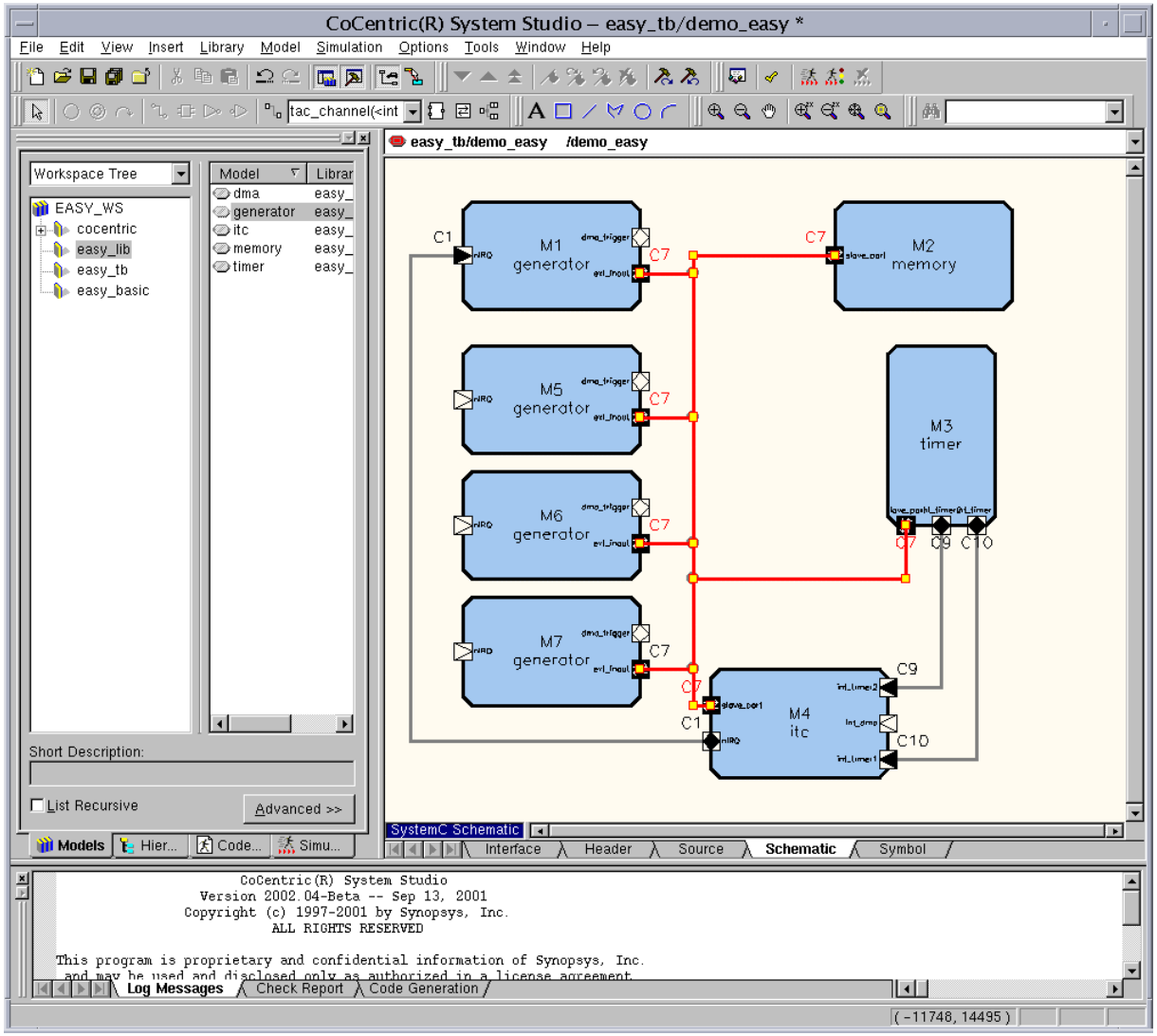
# Simple example of TLM platform



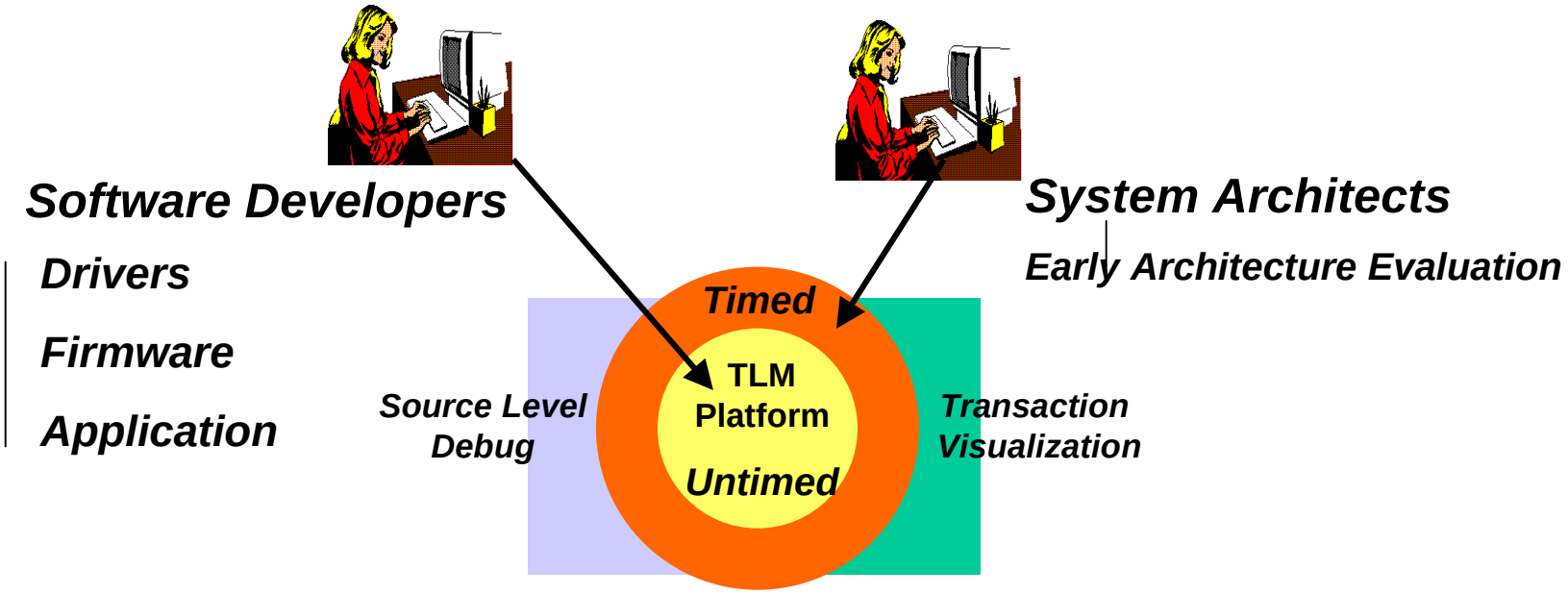
# Example: timer.h

```
#include "systemc.h"
#include "common.h"
#include "tac_prim_slave.h“

TAC_MODULE_SLAVE(timer) {
  // Timer interrupt signals
  sc_out<int> int_timer1;
  sc_out<int> int_timer2;
 // Timer internal registers
  DATA_TYPE timer1_value;
  DATA_TYPE timer1_load;
  …
// Registers init.
  void init_register();

// Module read access overload ( specific timer interface behavior )
  DATA_TYPE ReadAccess(const ADDRESS_TYPE addr_in);
// Module write access overload ( specific timer interface behavior )
  void WriteAccess(const ADDRESS_TYPE addr_in,const DATA_TYPE data);
  // Timer main function
  void Compute();

// Constructor
 TAC_CTOR_SLAVE(timer) {
  // Slave minimum service ( channel interface )
  TAC_MINIMUM_SLAVE_SERVICE;

  // Timer main function
  SC_METHOD(Compute);
  sensitive << dummy_event;

  // out port init ( interrrupt )
  int_timer1.initialize(0);
  int_timer2.initialize(0);

  // Registers init.
  init_register();
 }
 // Destructor
 ~timer() {
  }
};
```

# Platform code - read by tool



# Transaction Level Modeling Platform




**Key benefits**

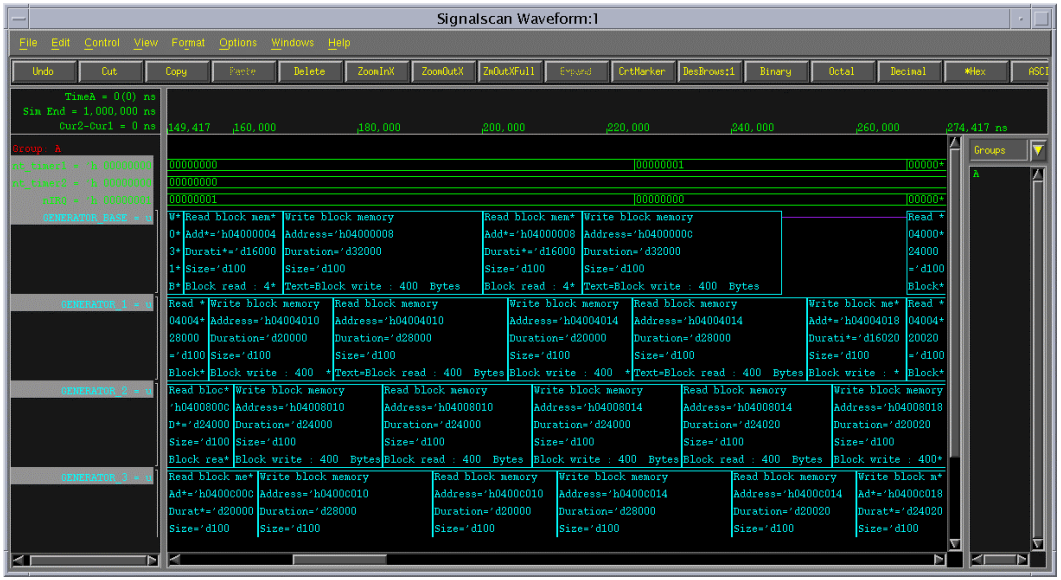
- Enable early functional software development
- Easy to implement (C / SystemC 2.0 functional models)
- Efficient simulation speed
- Source level debug
- Transaction visualization

# Annotating models with time – Ex.

- Transaction duration - Slave model
  - Additional, optional parameter
    in serveRead / serveWrite
  - Fixed value or complex, parameterised law
- Master duration between 2 transactions
  - Insert wait() statements

# SoC TLM debug and analysis

- Generate VCD files for signal tracing
- Generate database to record, visualise, analyse
- Untimed or timed TLM



# The SystemC TLM challenge

- Avoid current proliferation of TLM dialects
- Differentiate only where required
- Adopt common core set of modelling guidelines – promote interoperability

# Thank you !

- E-mail alain.clouard@st.com