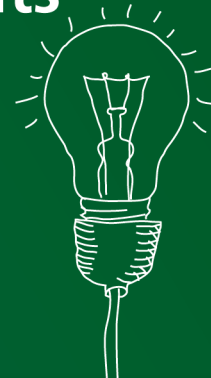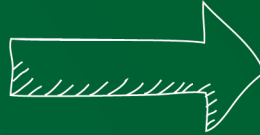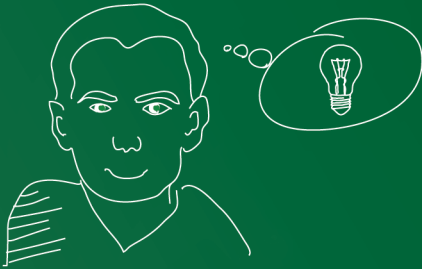# PragmaDev Tracer

Graphical Requirements and Traces

FREE TECHNOLOGY

PragmaDev Tracer is a brand new set of tools that aims at verifying a system matches its expected dynamic.

BENEFITS

- Express the requirements of your system,
- Trace your system behaviour on-line or off-line,
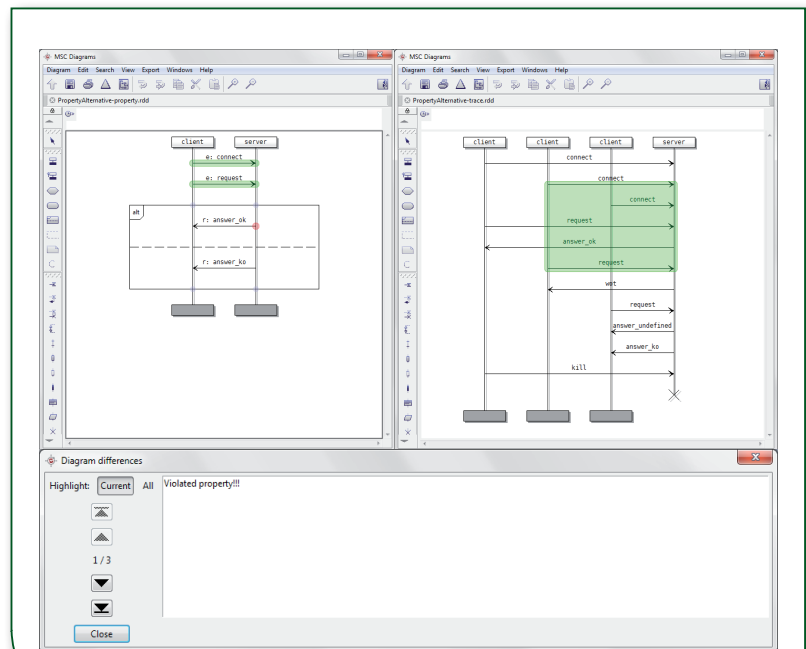- Verify the trace matches the requirements.

FEATURES

- Easy edition of the expected behaviour,
- Define the functional and non-functional properties of your system,
- Trace on-line via a socket or off-line in a file,
- Match the traces against the requirements and/or the properties.

The execution trace on the top right does not match the property on the top left

GRAPHICAL REPRESENTATION OF MAJOR EVENTS

- Tasks,
- States,
- Messages,
- Semaphores,
- Timers,
- Objects,
- Alternatives,
- Loops,
- Time constraints,
- Chain constraints.

THREE LEVELS OF VERIFICATION WITH FILTERING CAPABILITIES

- Diagram comparison
- Match requirements on traces with the support of relative time constraints, and inline expressions such as alternatives and loops.
- Match properties on traces using the Property Specification Chart notation.

GET MORE …

PragmaDev Tracer is an integrated part of PragmaDev Real Time Developer Studio. The whole studio includes formal modeling and testing technologies that can be simulated or compiled down to target.

■ INTEGRATION IN YOUR TESTING OR DEVELOPMENT ENVIRONMENT

The tracer can be started in text mode and in batch mode and control commands can be sent through the socket.
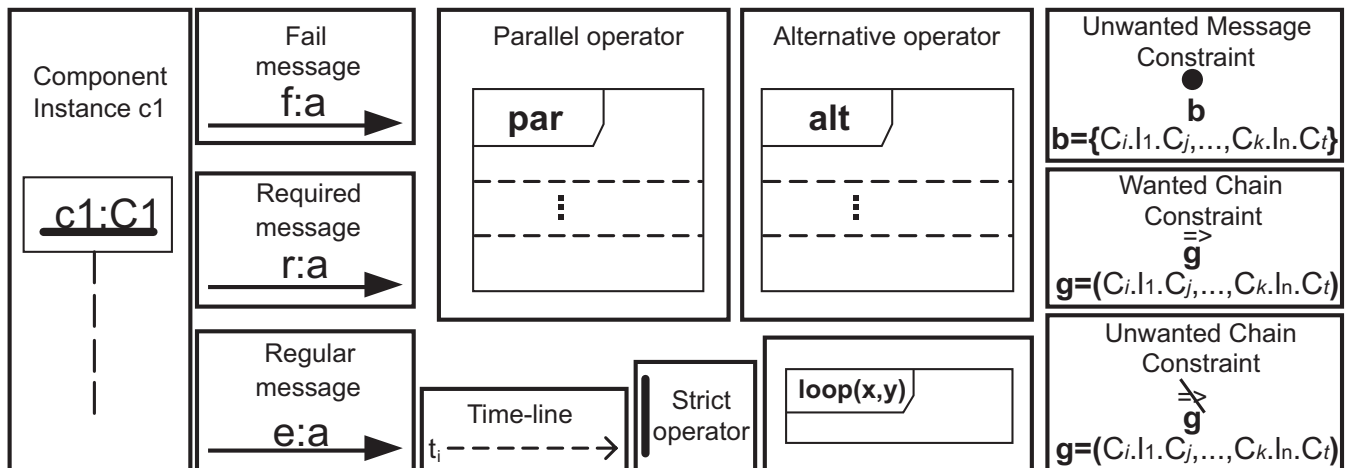
■ EASY CONNECTION

Connection to the tracer is done through a socket with a possibility of acknowledgement of each information received.

■ EASY DOCUMENTATION

• Generates PNG, PostScript and HTML or copy/paste to word processors for easy documentation.
• C macros are also available to easily generate traces in a file or in memory.
• Direct access from the trace event back to the source code or model

■ PROPERTY DEFINITION

Within the PSC language, a property is seen as a relation on a set of exchanged system messages, with zero or more constraints. PSC may be used to describe both positive scenarios (i.e., the "desired" ones) and negative scenarios (i.e., the "unwanted" ones) for specifying interactions among the components of a system. PSC has both formal notation and operational semantics.
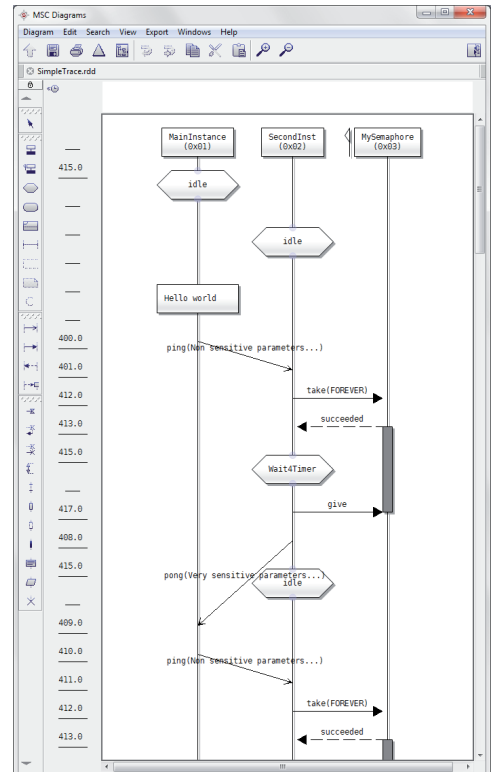
Trace the dynamic behaviour of your system down to the desired level of detail. Filter out events and collapse instances for further analysis or documentation.

Component Instance c1	Fail message $f:a$	Parallel operator **par**	Alternative operator **alt**	Unwanted Message Constraint ● **b** $b=\{C_i.l_1.C_j,...,C_k.l_n.C_t\}$
c1:C1	Required message $r:a$	⋮	⋮	Wanted Chain Constraint => **g** $g=(C_i.l_1.C_j,...,C_k.l_n.C_t)$
	Regular message $e:a$ — Time-line t_i - - - - → — Strict operator	**loop(x,y)**		Unwanted Chain Constraint ⇏ **g** $g=(C_i.l_1.C_j,...,C_k.l_n.C_t)$

■ TECHNOLOGIES

• MSC: ITU-T Z.120 international standard Message Sequence Chart,
• Sequence Diagram : UML 2.0 OMG diagram,
• PSC : University of L'Aquila Property Sequence Chart.
• PragmaDev Tracer has been developed in the framework of an FP7 ARTEMIS joint undertaking project.

■ FILE FORMAT

• Native XML,
• MSC PR,
• Trace commands,
• OTF.

GET IT FOR FREE

PragmaDev Tracer can be downloaded free of charge from our web site.