Towards An Open Trace-based Mechanism
position paper

Authors:
Paul Leger and Éric Tanter

Department of Computer Science (DCC)
University of Chile
A Trace-based Mechanism (TM) in a nutshell

A TM observes the execution of the software and executes a piece of code when this TM matches a specified sequence of events.

Trace of execution:
- d → f → d → a → b

The piece of code of TM:
- a → b

TM needs to match:
- a → b
Matching a sequence

Is it a valid match?

It depends because different TMs have different semantics to define sequences.
Strategies for multiple matching

Is there another match?

It depends because different TMs have different strategies for multiple matching
The life cycle of sequences

Trace of execution

TM needs to match

Most TMs cannot control the life cycle of sequences
### Current TMs

- **Tracematch** [Allan+@OOPSLA05],
- **Alpha** [Herzeel+@ILC07],
- **Halo** [Ostermann+@ECOOP05],
- **PTQL** [Goldsmith+@OOPSLA05],
- ...

### Specific and non-configurable features

- What happens if developers have specific needs?

### Is it a solution to “code around”?
Towards an Open TM (OTM) model

We identified three points of openness in a TM:

- **Sequences definition.** For example:
  - Regular expressions
  - Context-free languages
  - ...

- **The multiple matching strategy.** For example:
  - To match several sequences at the same time
  - To match only one sequence at the same time
  - ...

- **The life cycle of the sequences.** For example:
  - To remove all sequences if some condition is satisfied (*except example*)
  - To match all sequences if some condition is satisfied
  - ...
OTM model

Trace of execution

TM needs to match

Sequence

Env

seq

a

b

Env'

b

MATCH

Env"
Nondeterminism support

• Some operators to define sequences, like OR, are not deterministic. For example:

  ![Diagram showing nondeterminism](image)

  • If $a$ happens, then two different histories of the matching of the same sequence are generated.
Example: The AnyOrder Operator

The *AnyOrder* operator matches several *sequences in any order*.

Trace of execution

![Trace Diagram]

TM needs to match

![TM Need Diagram]

10
Towards an Open TM (OTM) model

We identified three points of openness in a TM:

- **Sequences definition.** For example:
  - Regular expressions
  - Context-free languages
  - ...

- **The multiple matching strategy.** For example:
  - To match several sequences at the same time
  - To match only one sequence at the same time
  - ...

- **The life cycle of the sequences.** For example:
  - To remove all sequences if some condition is satisfied (except example)
  - To match all sequences if some condition is satisfied
  - ...

Extending OTM

- We add a *multiplexer* entity to define strategies for multiple matching.

- We add a *sequence controller* entity to control the life cycle of the sequences.
Conclusions

- We explored the points where a TM can be opened:
  - Sequences definition
  - Multiple matching strategies
  - Life cycle of sequences
  - Any others?

- We designed an open TM model, taking into account these points
Thank you!

Future work:

- Extend AspectScript[1] to support the OTM model.

The class diagram of our open TM

Developers can build strategies for multiple matching.

Developers can build strategies to control life cycle of sequences.
A Open TM

Sequence 1

Env
seq
sa
sb

Env1
sb

MATCH
Env1'

Sequence 2

Env2
sb

MATCH
Env2'

Sequence Manager

Multiplexer

Trace of Execution

TM needs to match

a → a → b

a → b