functional_constraint

This perspective is conform when the given functional constraints evaluate to true.

author:
`Michael Igler (michael.igler@uni-bayreuth.de)`
version:
`0.8`
date:
`2010/4/19`

compilation:
`static, context_switching_calls`

(no dependencies on other files)

Public interface

functional_constraint/4

Contains the POPM constraints of the functional perspective.

compilation:
`static`

mode - number of solutions:
`functional_constraint(+list(actions),+modelstate,-instance_identifier, +callable) - zero_or_more`

functional_constraint/5

Contains the POPM constraints of the functional perspective.

compilation:
`static`

mode - number of solutions:
`functional_constraint(+modal_prefix,+list(actions),+modelstate,-instance_identifier,+callable) - zero_or_more`

functional_constraints_conform/3

This predicate expects: instantiated Action(+var), instantiated ModelState(+list) and a free or instantiated Instance(?var).

compilation:
`static`

mode - number of solutions:
`functional_constraints_conform(?action,+modelstate,?instance_identifier) - zero_or_more`

counter_increasable/1

Verifies if the counter can be increased by 1. The current counter has to be smaller than supremum of the counter_interval.

compilation:
`static`

mode - number of solutions:
`counter_increasable(+var) - zero_or_one`

interval_supremum/1

Retrieves the interval supremum of a process.

compilation:

```
static
```

mode - number of solutions:

```
interval_supremum(?var) - zero_or_one
```

interval_infimum/1

Retrieves the interval infimum of a process.

compilation:

```
static
```

mode - number of solutions:

```
interval_infimum(?var) - zero_or_one
```

counter_interval/1

Retrieves the interval of a process.

compilation:

```
static
```

mode - number of solutions:

```
counter_interval(-interval) - one_or_more
```

correlated_to_other_processes/1

Checks if the counter_interval of the process is dependent/correlated to other processes.

compilation:

```
static
```

mode - number of solutions:

```
correlated_to_other_processes(+modelstate) - zero_or_one
```

exists_instance/3

Verifies if a certain instance has been executed in the given model state.

compilation:

```
static
```

mode - number of solutions:

```
exists_instance(+modelstate,?action,?instance_identifier) - zero_or_more
```

actions_applicable/3

Checks of the actions are applicable in the given model state.

compilation:

```
static
```

mode - number of solutions:

```
actions_applicable(+modelstate,?action,?instance_identifier) - zero_or_more
```

in_domain/1

Checks if process counter is in the domain interval in a given model state.

compilation:

```
static
```

mode - number of solutions:

`in_domain(+modelstate) - zero_or_one`

all_done/1

Checks if the counters of all processes build a solution in the given model state.

compilation:

`static`

mode - number of solutions:

`all_done(+modelstate) - zero_or_one`

instance_agent_data_tool/6

Returns which instances are started/finished/aborted in a given model state. Lists of agents, data and tools are also retrieved expressing more detailed information about the execution.

compilation:

`static`

mode - number of solutions:

```
instance_agent_data_tool(+modelstate,+action,?instance_identifier,?
list(agent_identifiers),?list(data_identifiers),?list(tool_identifiers)) -
zero_or_one
```

optional/0

Checks if a process is optional.

compilation:

`static`

mode - number of solutions:

`optional - zero_or_one`

highest_instance_id/2

Retrieves highest instance-ID of a process.

compilation:

`static`

mode - number of solutions:

`highest_instance_id(+list,-number) - zero_or_one`

Protected interface

(none)

Private predicates

(none)