## threaded\_cal1/2

## **Description**

```
threaded_call(Goal, Options)
threaded_call((Goal1, Goal2, ...), Options)
threaded_call((Goal1; Goal2; ...), Options)
```

Prove Goal using a new thread. By default, the proof is done asynchronously. The argument can be a message sending. The result (success, failure, or exception) is sent back to the thread of the object containing the call (*this*). The option noreply may be used when no reply with the proof results is necessary. The option atomic may be used if the goal has side-effects such as dynamic database updates or input-output operations; this option is only safe if the side-effects are confined to the object where the goal is proved (or to the object receiving the asynchronous message).

When the argument is a *conjunction* of goals, the call is equivalent the conjunction of calls of the individual goals. However, when the argument is a *disjunction* of goals, the call is equivalent to the *competing* calls of the individual goals: when one of the goals complete, the other ones are aborted (i.e. their threads are terminated). In this case, the corresponding threaded\_exit/1-2 goal **must** match all the goals in the disjunction. This is useful when you have a set of different methods to solve a problem without knowing a priori which one will lead to the fastest result.

## Template and modes

```
threaded_call(+callable, +list)
```

## **Examples**

Copyright © Paulo Moura — Logtalk.org XHTML + CSS Last updated on: May 26, 2006

