threaded_call/1

Description

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

Prove `Goal` asynchronously using a new thread. The argument can be a message sending. This call always succeeds. The result (success, failure, or exception) is sent back to the thread of the object containing the call (*this*).

When the argument is a *conjunction* of goals, the call is equivalent to 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)
```

Examples

Prove `Goal` asynchronously in a new thread:

```
threaded_call(Goal)
```

Send an asynchronous message to *self*:

```
threaded_call(::Message)
```

Send an asynchronous message to an object:

```
threaded_call(Object::Message)
```


Last updated on: October 26, 2005