

CoreAPI

The core API defines how applications, (which reside on places) interact with each other and modify their attributes. See AskemosDVM for definition of the virtual machine. TODO: provide a WSDL₂ description of the core API here.

The Reply Element

In the simple case of a read type request, the reply to be returned from an actors invocation is just the output to the requestor.

Things are more complicated for write type requests.

A write type request returns an element `reply` from the XML namespace <http://www.askemos.org/2000/CoreAPI#>. The children of this element describe the effect of the process step, along the axis

TODO: link from here to detailed information about the list items.

- Hierarchical structured data values, the `become` element.
- interconnection: subscription, naming and growth; (networks, sozietäres Prinzip)
- rights, relations
- entry points: secrets etc.
- sugar; (relations [tables], forwarding, email (TrustedCode etc.)

TODO describe the API from the code. CAUTION small changes ahead.

Data Values

One mandatory child is the `become` element. It contains the new state of the place. In other words it's the new data value stored conveyed. for example: if the place stores constant data, this data is returned as content of replies given to read requests. Low level slot: `mind-body` and corresponding `xml-parse'd` representation `body/parsed-xml`.

remark The `become` element is usually the 1st, since it was historically the only positional return parameter. Furthermore an alternate name "`continue`" is supported, which stems from the Scheme inheritance of the project. However reading <http://www-sop.inria.fr/oasis/caromel/TDO/ASP-DistributedCalculi.pdf> page 4 (top) was convincing to find "become" the better name.)

Optionally any number of other elements can follow. These effect the other (all but the first) axis.

An element `output` can be used to specify more details about the output, i.e, narrow it or generate answers with a content type other than XML.

A good application design puts everything, which is efficient described by the hierarchical principle (english?) at that axis.

interconnection

TODO subscription (not yet implemented)

read

To resolve the name of a link originating from a place, call the accessor (which is passed as the first argument see [ActionDocument](#)) like this: (place *name*)

To resolve a public name see: [public-context](#)

To *read* data from another place see [fetch](#).

write

- [link](#)
- [new](#)
- [send](#)

The interconnection axis is used to convey information on societal relations.

Rights

protection and capabilities see [AskemosProtection](#). grant rights to other users.

- [grant and revoke](#)

entry points

The `secret` element is used to control credentials for user authorisation.

[TODO](#) document `EntryPoint?` to create new entry points.

Sugar

`forward` is equivalent to an unchanged data value and a message to send to the next step of the path as described by the `destination` slot of the message. The `location` in the resulting message extended by an element reflecting this next step.

[TODO](#) document `let`, `letseq` etc.

See also [xsql](#).

Last modification: Fri, 05 May 2006 12:13:45 +0200

Author(s): jfw,

Document number A849640f672ed0df0958abc0712110f3c page CoreAPI delivered to public at Wed, 08 Sep 2010 15:40:43 +0200