Methodologies in Information Systems Development

Modeling Computational Aspects of a System in OPM

Group No` 1:
More
Aviel
Arik
Yossi
Roman
Aharon
Motivation

- Some computational scenarios are not well defined in OPM.
- This causes a lack of consistency and makes it difficult to use OPM modeling for developing software.
- We created models, based on the correct OPM semantics according to “Operational Semantics for OPM” by Valeria Perelman.
- They could be used as templates in OPCAT.
While Loop – Our Solution
Why is Control-Flow important?

- The standard OPM model conceptually captures the system’s behavior (along with its structure).
- A concrete representation of behavior is not fully existent in the OPM model. For this matter, a parallel representation of the model is being developed, called OPM Matlab Layer (OPM\ML).
In order to correctly replace an OPD representation with a OPM\ML representation (and vice versa) it is necessary to define a single accurate representation for Control-Flow processes.

It is vital that the model representation is compatible with the true OPM semantics, in order to maintain the compatibility between the model and the OPM\ML, which is created by those semantics.
The Parallel Representation

```java
1 - while ParameterSet.Integer > 0 {
2 - ParameterSet.Integer = ParameterSet.Integer - 1;
3 - end
```
While Loop – Count to 3 example

The user needs to connect

Initialized to 3

Parameter Set

Integer: INT\^10

>0

>0

Build in

Condition Evaluating

Satisfying

Dissatisfying

Condition

True False
While Loop – Count to 3 example

Condition
True
False

Condition Checking

Continuing

Breaking

Action Performing

Loop Exiting
While Loop – Count to 3 example

Decrease integer by 1 in each iteration

How do we know when the effect takes place?
While Loop – Alternative 1

What is the order of the state changing?

When procedural links that originate from an entity are labeled, the one that must be followed is the one whose label is identical with the label of the procedural link that arrives at the entity.
In the process iteration, the context of the process already exists and is reused, thus all the local objects of the process instance preserve their state from the previous run.
While Loop – Alternative 3

Wait until means that the triggered process checks the preprocess object set.

- Requirement link instead of checking condition
- The process can get stuck because of the wait until mechanism
Skip / Wait until

If Pre-Process Conditions Fail
The process is **skipped**

If Pre-Process Conditions Fail
The process *waits until* they become true
Sub process outside the in-zooming ellipse isn’t Part of the time flow and needs to be invoked by the process’s internal events.
Other Points of Interest

Referencing Processes

• A sub process can be referenced using the "ref:path" role, with a path to the process definition.
• All the objects linked to the referenced process must match the types defined for the process.
• If there are several links of the same type, they should be distinguished by different tag "roles" (divisor and dividend)
• When running in an enclosed process, the ref process uses the parents’ local variables. If not, it uses global context of the system.
Example – Recursion
A Constraint process is re-activated in an infinite loop during the life time of the thing it’s bounded to by the characterization link.
Conclusions

- Even given elaborate semantics, there are many different ways of modeling systems.
- While it is important to allow flexibility in modeling systems, it is essential that fundamental aspects of software be consistent and build in within OPCAT.
- There needs to be a golden path in allowing both flexibility and conformity.
Thank You