Towards the Ontological Foundations for the Software Executable DEMO Action and Fact Models

Marek Skotnica, Steven J. H. van Kervel, Robert Pergl

Agenda

- Motivation
- DEMO Machine
- Fact and Rule Ontology
- Volley Club Example

Motivation

- We want to support DEMO-driven enterprise with software systems
- Integration with existing IT systems
- Separation of concerns - separate facts and acts
- The challenge of DEMO models execution

DEMO Machine

- A theoretical computation model for simulation of DEMO models
- Based on state machines and deterministic evaluation
- Implementation independent

DEMO Machine Motivation

- **Integration** - DEMOSL concepts are either new (to be carried out) or existing concepts already present in the enterprise. DEMOSL does not deal with this separation.
- **Facts duplication** - The facts representation must be physically present in one place to assure the consistency of enterprise systems.
- **Lack of expressiveness** - At the execution level, there are many domains, where DEMOSL is not expressive enough to describe it, like scientific computations.
- **Modularity** - DEMOSL does not specify, how the solution is modularised, which is at the same time a crucial execution concern.
- **Lack of version transparency** - DEMOSL does not deal with evolvability of the models with the respect to the running instances.
- **Execution semantics of DEMOSL** - Currently, the execution semantics of DEMOSL is not fully specified.

Fact and Rule Ontology

- To be based on and compliant with the FI, TAO and PSI theories of EE
- Truthfulness and good appropriateness qualities and compliance with the three cardinality laws
- Maintaining the strict C4-ness criteria

Fact and Rule Ontology

- **Integration and Facts duplication** - Based on the Separation of Concerns Principle from the Normalised Systems Theory, the DEMO Machine should not supply the functionality of the already-existing enterprise systems, such as a database. Also, the DEMO Machine should not specify scales, dimensions, sorts, units such as time, money and others.
- **Lack of expressiveness** - For areas, where there are already established solutions (like programming language libraries), these should not be represented in a DEMO Machine.
- **Execution semantics of DEMOSL** - The execution semantics should be specified by the DEMO Machine. The FAR Ontology focuses on the subset of execution, namely the facts, agenda and rules concepts.
- **Modularity and Version transparency** - They are a subject for future work that should be based on the studies of Normalised Systems Theory mentioned above.

Facts

- A **Fact** is a proposition about something that exists in the real world and provides us with factual knowledge about the world.
- FAR ontology distinguishes three types of facts:
- **Internal** - a factual statement about a DEMO model instance. FAR provides a grammar and DEMO machine an evaluation.
  - E.g. “Are invoices paid?” -> *“T03<"Invoice">.all(t => t.state == accepted)”*
- **External** - a Fact about the world outside the DEMO Machine. DEMO Machine calls an outside world system to evaluate this fact.
  - It can be implemented e.g. programming language function or a web service.
- **Composed** - a fact composed from internal and external facts. Kleene and Priest three-valued logics is used for evaluation.
  - E.g. “F01 and F02”

Fact Value

- **Fact value** is a valuation function:
  - ***(Transaction Instance, Fact) -> {True, False, Undefined}***
  - The fact value depends on a transaction instance e.g. fact “Is initiator of T1 older than 18 years?”
  - Undefined means that the subjects of the proposition does not exist, yet, or we do not know the valuation, as a result of e.g. a technical failure or timeout.

FAR Agenda

- **FAR Coordination Act (cAct)** is a proposed or intended action for an actor.
  - *cAct := (Transaction, TransactionInstance, ActorInstance, Intention, SettlementType)*
  - Intention ∈ { Create(T, n), Promise, Decline, … }
  - SettlementType ∈ { Allow, Enforce, Restrict }
- **Agenda** is a function that calculates a set of actor's possible actions based on the current state of the model taking into account the composition axiom and the respective rules.
- DEMO Machine defines how an Agenda is calculated.

Rules

- Rules in the FAR ontology are used to express the action model (dynamic transitions and restrictions in the state execution space).
- Rules are specifications of either a prescriptive execution of a coordination act, or a conditional prohibition of a coordination act for an actor, depending on the evaluation of a fact.
- A rule restrict the available freedom of an actor to issue coordination acts at the execution time.
- The evaluation, if the rule or dependency applies, takes place at runtime, depending on the state of that model instance.
- Two types of rules are needed - **Conditional** and **Causal**

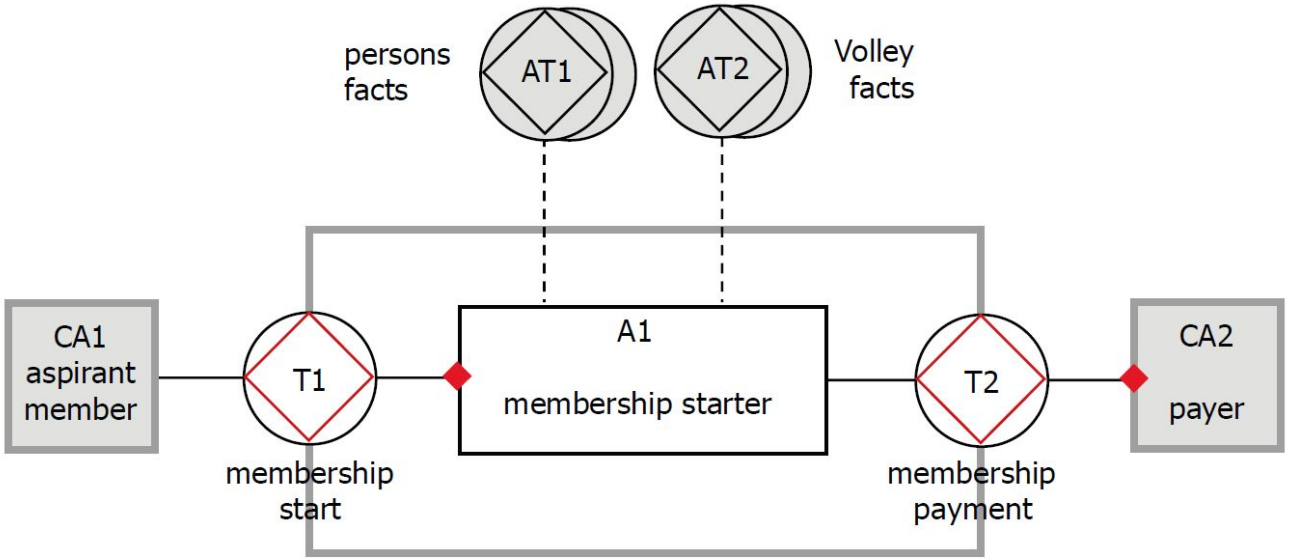
Conditional Rule

- The application of a rule that results in a restriction of an agendum, in such a way that one of the allowed coordination acts is prohibited while the rule applies.
  - *ConditionalRule = (Transaction, Fact, cActToRestrict)*
- Simply put - transaction instance state can be only reached if the fact associated with rule is evaluated as True.
- E.g. - “T1.State can be only performed when the requested pizza is baked and paid.”

Causal Rule

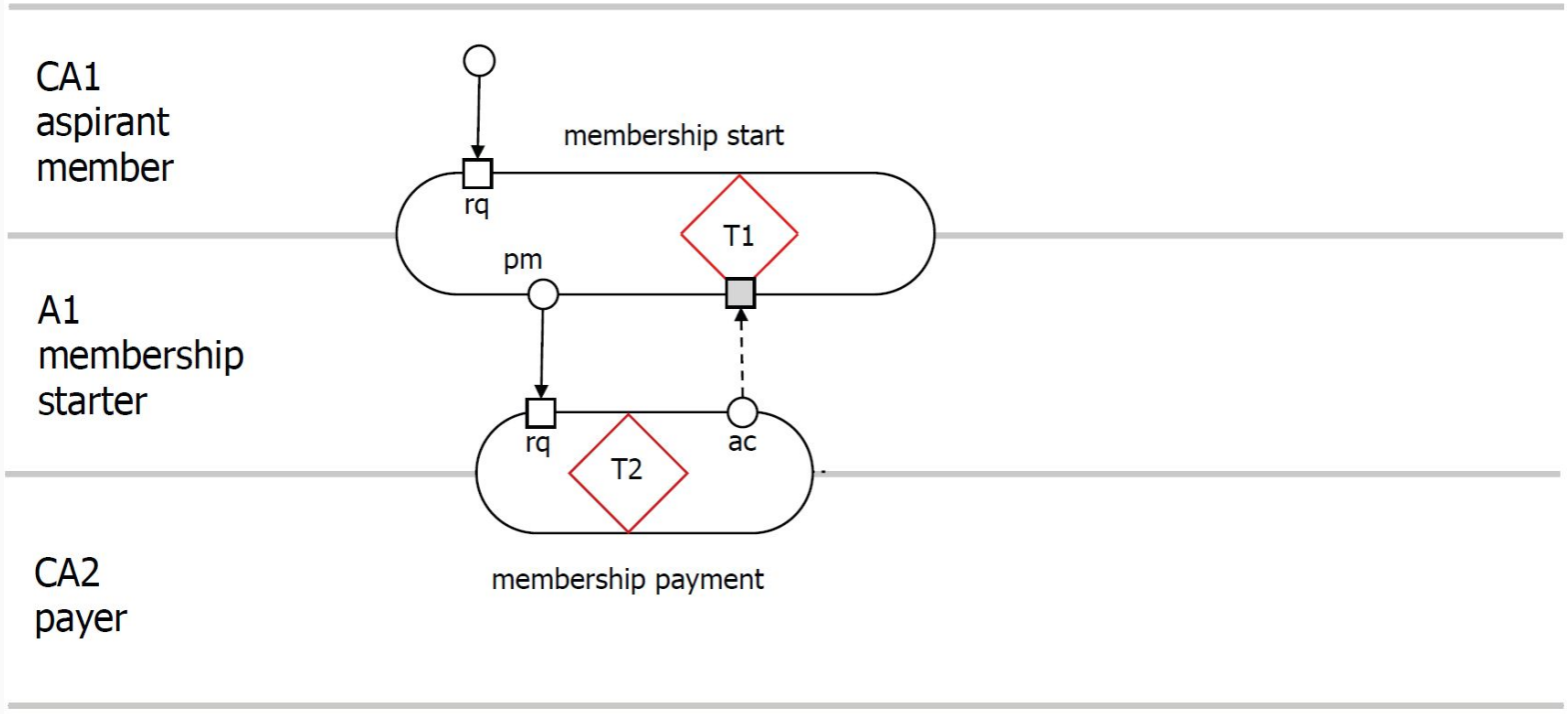
- The application of a rule that results in a transaction state change.
  - *CausalRule = (Transaction, TransactionState, Fact, cActTrue, cActFalse)*
- E.g. “If a person is eligible for a membership than promise the membership”.

Volley Club Example - OCD



transaction kind	product kind
T1 membership start T2 membership payment	P1 Membership **is** started P2 **the** first fee **of** Membership **is** paid

Volley Club Example - PSD



Volley Club Example - AM1

- When the membership start (T1) is requested, if the person who is requesting is eligible then it is automatically promised, otherwise declined.

when	membership start **for** new Membership is requested		(T1/rq)
	with	**the** member **of** Membership **is some** PERSON	
		the starting day **of** Membership **is some** DAY	
assess	*justice:*	**the** performer **of the** request **is the** member **of** Membership	
	sincerity:	< no specific condition >	
	truth:	**the** starting day **of** Membership **is the** first day **of some** MONTH;	
		the age **of the** member **of** Membership **on the** starting day **of** Membership **is equal to or greater than the** minimal age **in the** year **of the** starting day **of** Membership;	
		the number of members **on the** starting day **of** Membership	
		is less than the max members **in the** year **of the** starting day **of** Membership	
if	*complying with request is considered justifiable*		
then	promise	membership start **for** Membership	[T1/pm]
else	decline	membership start **for** Membership	[T1/dc]

Volley Club Example - AM1

- AM Rule 1 is in FAR represented by an external fact and a causal rule
- External Fact EF1
  - Id = isMemberElegibleFact
  - Label = "Is member eligible for application?"
  - LogicalProposition = "Person.Age >= Minimal_Required_Age"
  - At runtime, an external system implementation needs to be assigned to calculate EF1
- Causal Rule CR1
  - TransactionType = transaction1
  - ConditionState = TransactionState.Requested
  - Fact = isMemberElegibleFact
  - CActTrue = CActType.Promise
  - CActFalse = CActType.Decline
- Full example with simulation in Marek’s diploma thesis

Summary

- Proposed theoretical concepts to express the execution of DEMO models
- Proof of concept implementation was made in ForMetis
  - We plan public DEMO API to encourage development of DEMO-based applications
  - Please contact us if you are interested
- Future research topics proposed