Trustworthy and Secure Future Internet

Reputation-based Composition of Social Web Services

Gianpiero Costantino

in collaboration with

Alessandro Celestini, Rocco De Nicola, Zakaria Maamar,
Fabio Martinelli, Marinella Petrocchi, and Francesco Tiezzi

IANA I 2014/05/15

e-mail : gianpiero.costantino@iit.cnr.it

Overview

- **Social Web Services**
  - Competition and Collaboration Social Network
  - Scenario
  - Klaim Language
  - Simulation
  - Model Checking
- **Conclusion**

Le ONLUS
BUILDING DONOR LOYALTY
The Volunteer Management Handbook
CONDUCTING A SUCCESSFUL FUNDRAISING PROGRAM
CAUSE RELATED MARKETING
BEYOND FUND RAISING

Social Web Service

- A Social Web Service (**SWS**) is an entity that provides a specified service.

- SWSs are similar to traditional Web Service, but they keep a network of collaborative social group.

- SWSs can be grouped to form a **Composition**.

Social Web Service (2)

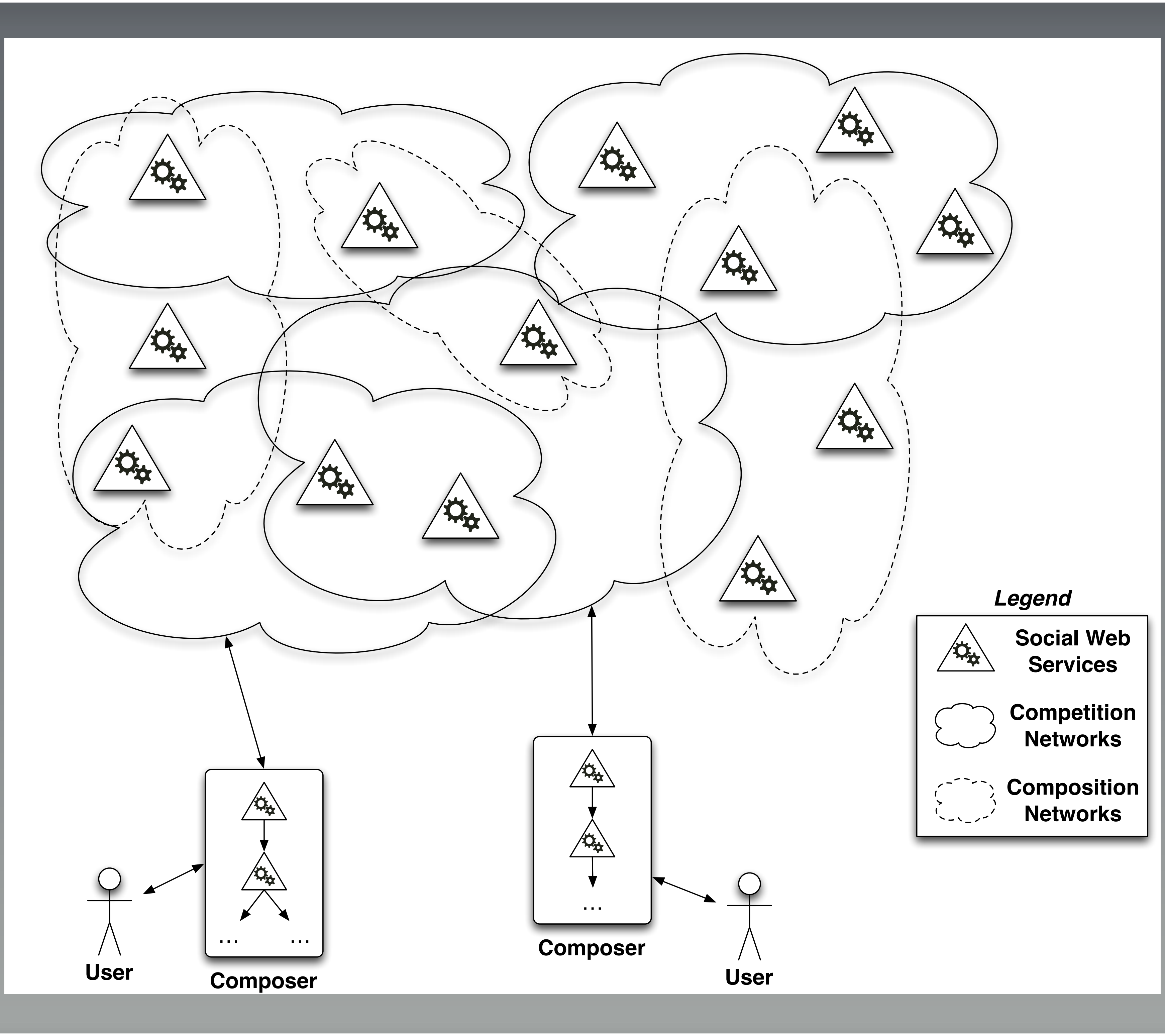
- A SWS can **sign up** to be part of a Composition of SWSs.

- It can be part of a **Competition**, **Collaboration** or Social Network:
  - Within a Competition Network all SWSs perform the same services.
  - Within a Collaboration Network all SWSs cooperate to provide a service.

Goal

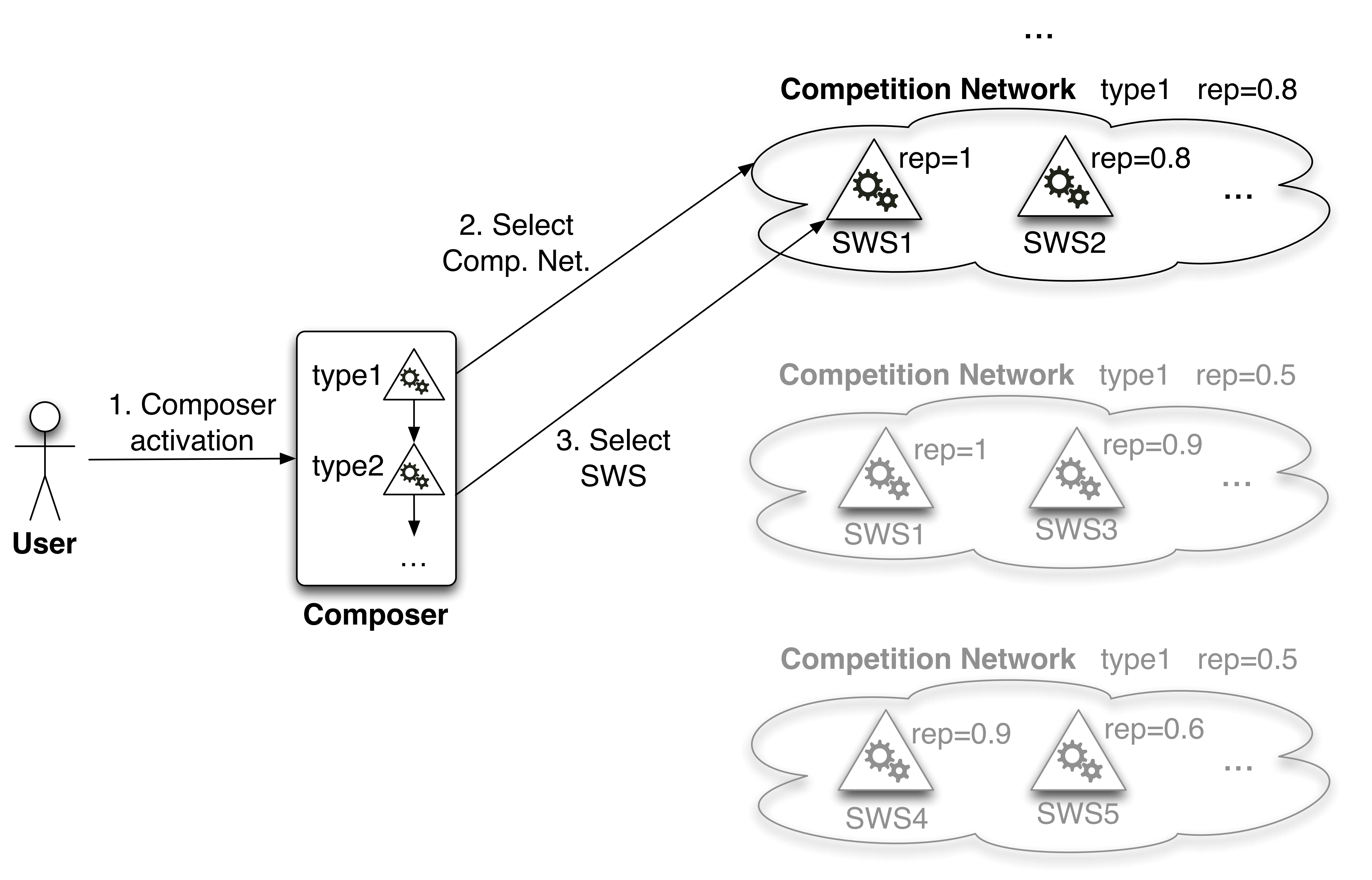
“In this work we aim at formally modelling and analysing a composition of SWSs in which SWSs selection is based on their reputation”

Scenario


Legend
Social Web Services
Competition Networks
Composition Networks
...
...
...
User
Composer
Composer
User

Scenario

Competition Network

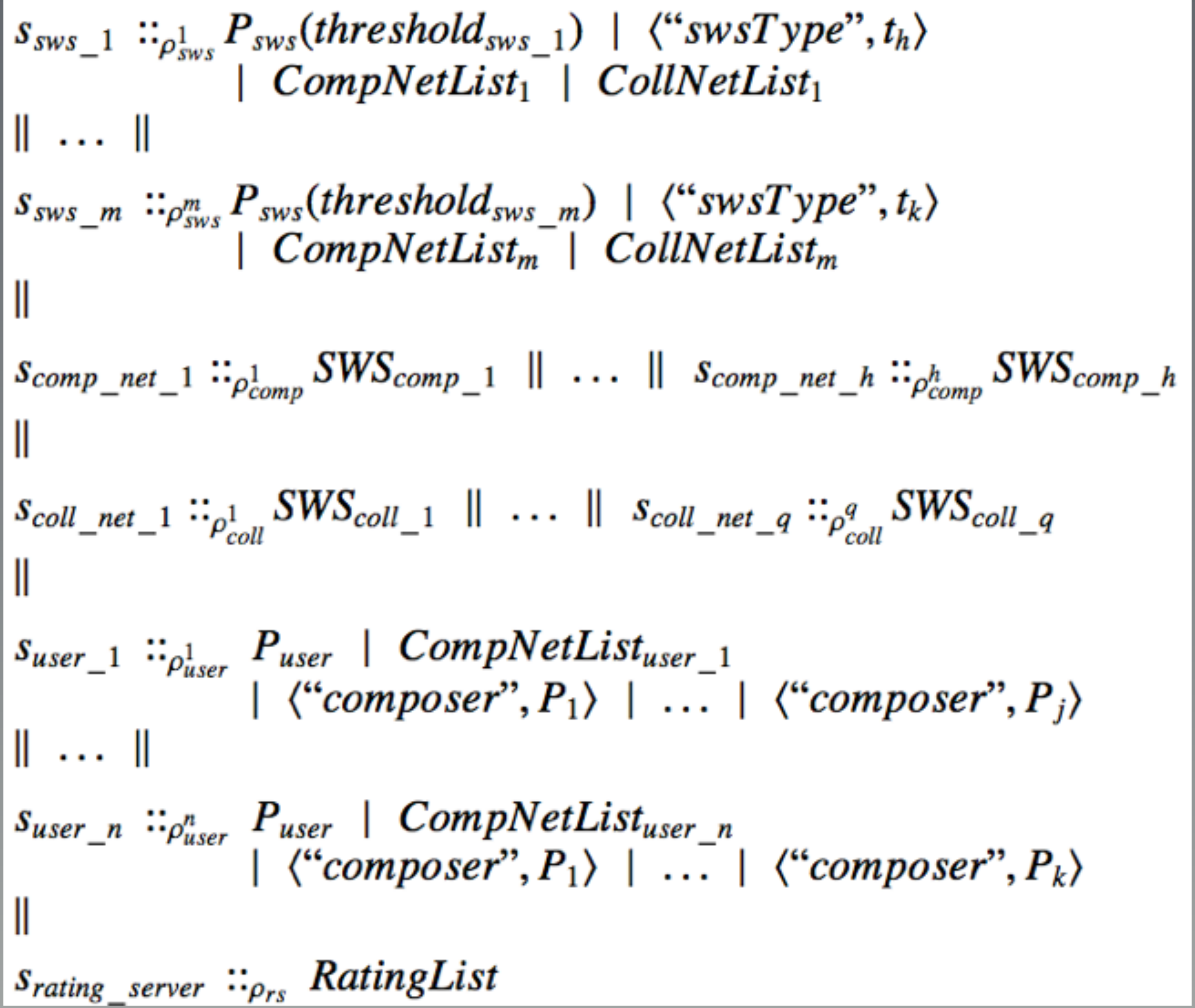


Klaim

- **Klaim** is a tuple-based language specifically designed for modelling distributed applications.

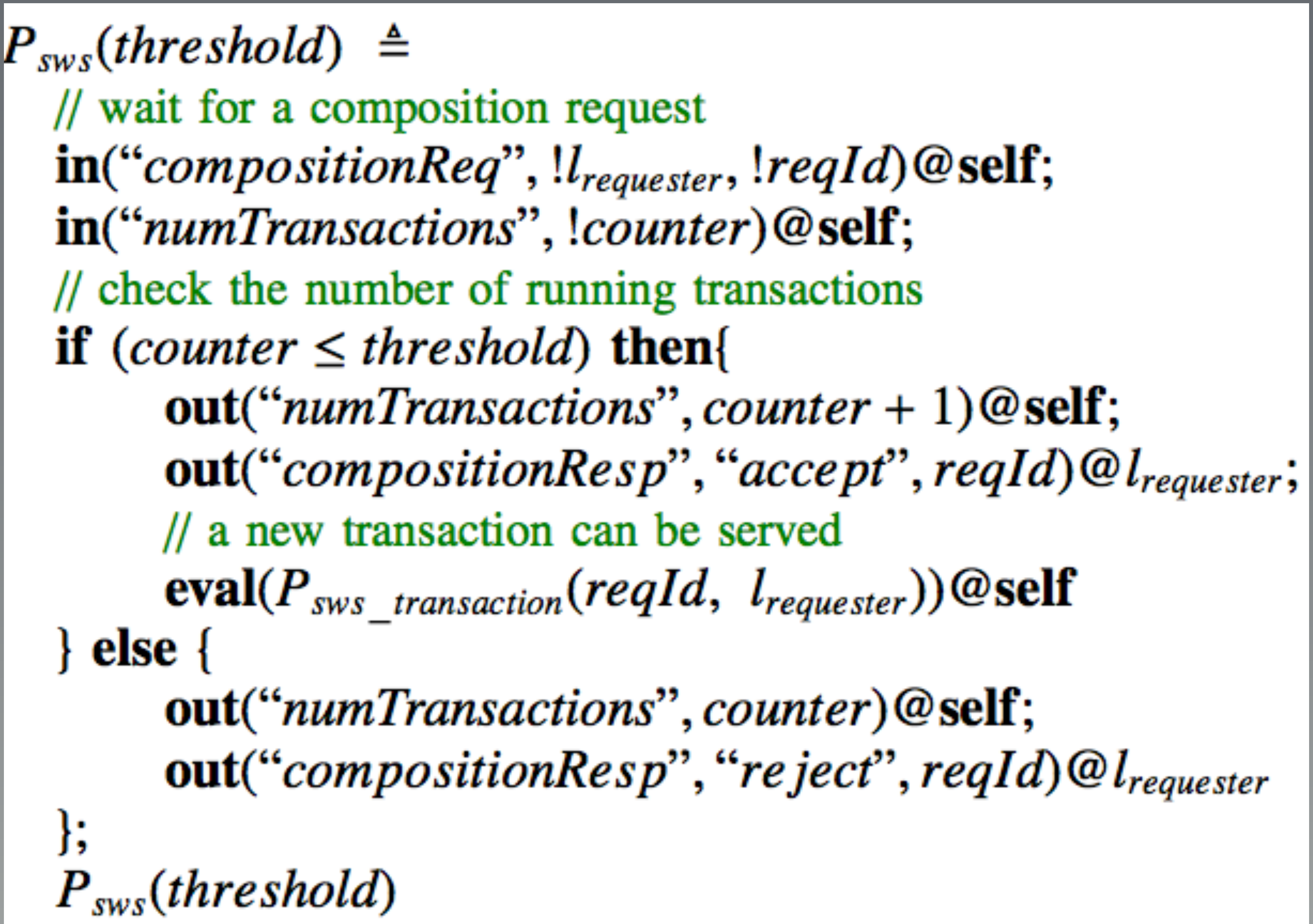
- In Klaim *processes* and *actions* can be defined:
    - *Read(T)@l*
    - *Out(T)@l*
    - *Eval(P)@l*
    - *…*

Scenario in Klaim



$$
\begin{array}{l}
s_{sws\_1} \ ::_{\rho^1_{sws}} P_{sws}(threshold_{sws\_1}) \mid \langle \text{“}swsType\text{”}, t_h \rangle \\
\qquad\qquad \mid CompNetList_1 \mid CollNetList_1 \\
\parallel \ \ldots \ \parallel \\
s_{sws\_m} \ ::_{\rho^m_{sws}} P_{sws}(threshold_{sws\_m}) \mid \langle \text{“}swsType\text{”}, t_k \rangle \\
\qquad\qquad \mid CompNetList_m \mid CollNetList_m \\
\parallel \\
s_{comp\_net\_1} ::_{\rho^1_{comp}} SWS_{comp\_1} \parallel \ldots \parallel s_{comp\_net\_h} ::_{\rho^h_{comp}} SWS_{comp\_h} \\
\parallel \\
s_{coll\_net\_1} ::_{\rho^1_{coll}} SWS_{coll\_1} \parallel \ldots \parallel s_{coll\_net\_q} ::_{\rho^q_{coll}} SWS_{coll\_q} \\
\parallel \\
s_{user\_1} \ ::_{\rho^1_{user}} \ P_{user} \mid CompNetList_{user\_1} \\
\qquad\qquad \mid \langle \text{“}composer\text{”}, P_1 \rangle \mid \ldots \mid \langle \text{“}composer\text{”}, P_j \rangle \\
\parallel \ \ldots \ \parallel \\
s_{user\_n} \ ::_{\rho^n_{user}} \ P_{user} \mid CompNetList_{user\_n} \\
\qquad\qquad \mid \langle \text{“}composer\text{”}, P_1 \rangle \mid \ldots \mid \langle \text{“}composer\text{”}, P_k \rangle \\
\parallel \\
s_{rating\_server} \ ::_{\rho_{rs}} \ RatingList
\end{array}
$$

SWS Process

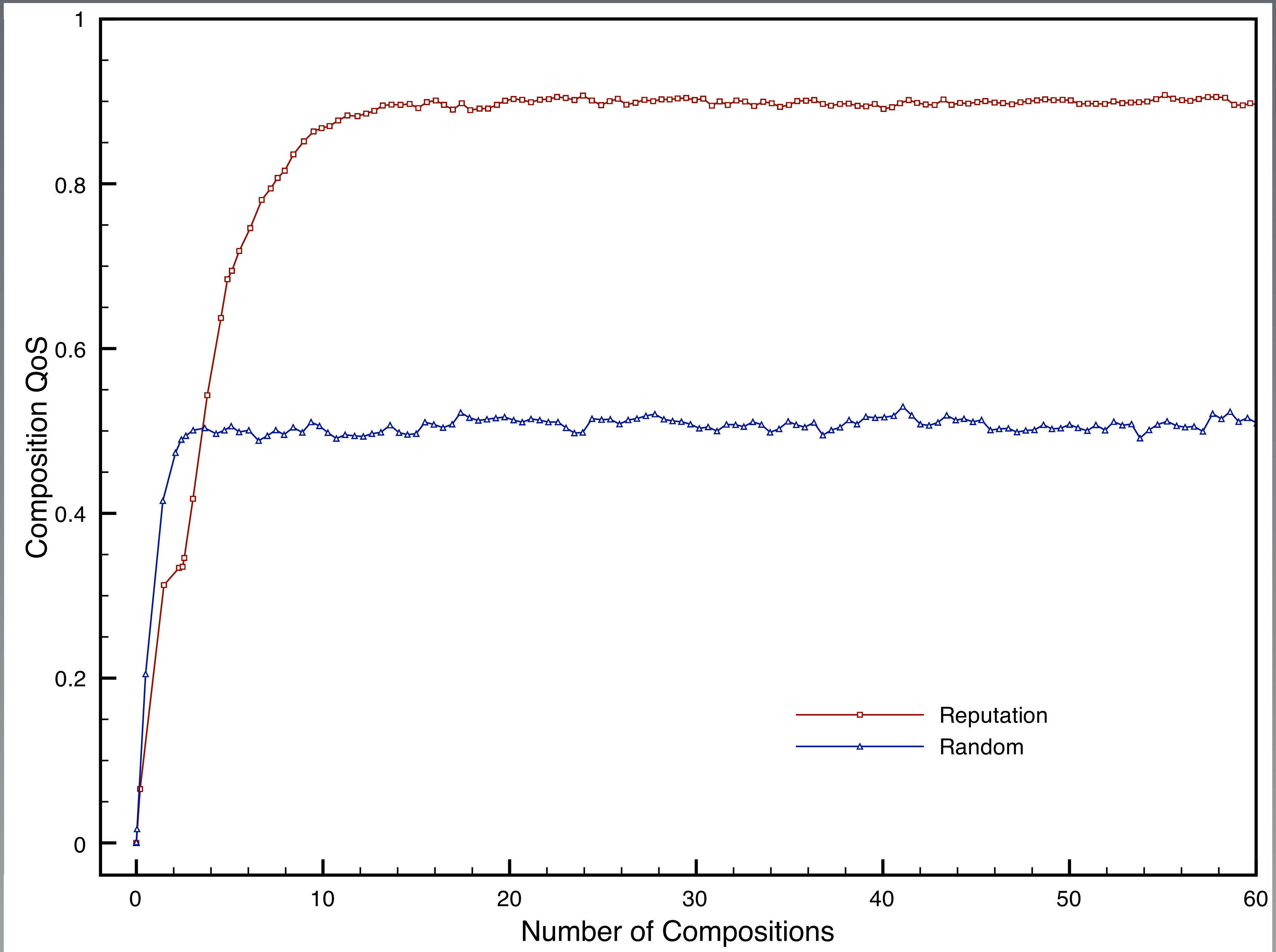


```
$P_{sws}(threshold) \triangleq$
    // wait for a composition request
    in("compositionReq", !$l_{requester}$, !$reqId$)@self;
    in("numTransactions", !$counter$)@self;
    // check the number of running transactions
    if ($counter \leq threshold$) then{
            out("numTransactions", $counter + 1$)@self;
            out("compositionResp", "accept", $reqId$)@$l_{requester}$;
            // a new transaction can be served
            eval($P_{sws\_transaction}(reqId,\ l_{requester})$)@self
    } else {
            out("numTransactions", $counter$)@self;
            out("compositionResp", "reject", $reqId$)@$l_{requester}$
    };
    $P_{sws}(threshold)$
```

Simulations

- **Goal**: Comparing two strategies for selecting SWSs. ***Reputation*** *Vs* ***Random***
- ***4*** *Competition* and ***2*** *Collaboration* Networks
- **12** SWSs with probabilistic behaviour
- **Composition Quality** is calculated as the mean of all SWSs reputations belonging to the composition.

Result



Composition QoS
1
0.8
0.6
0.4
0.2
0
0
10
20
30
40
50
60
Number of Compositions
Reputation
Random

Model Checking

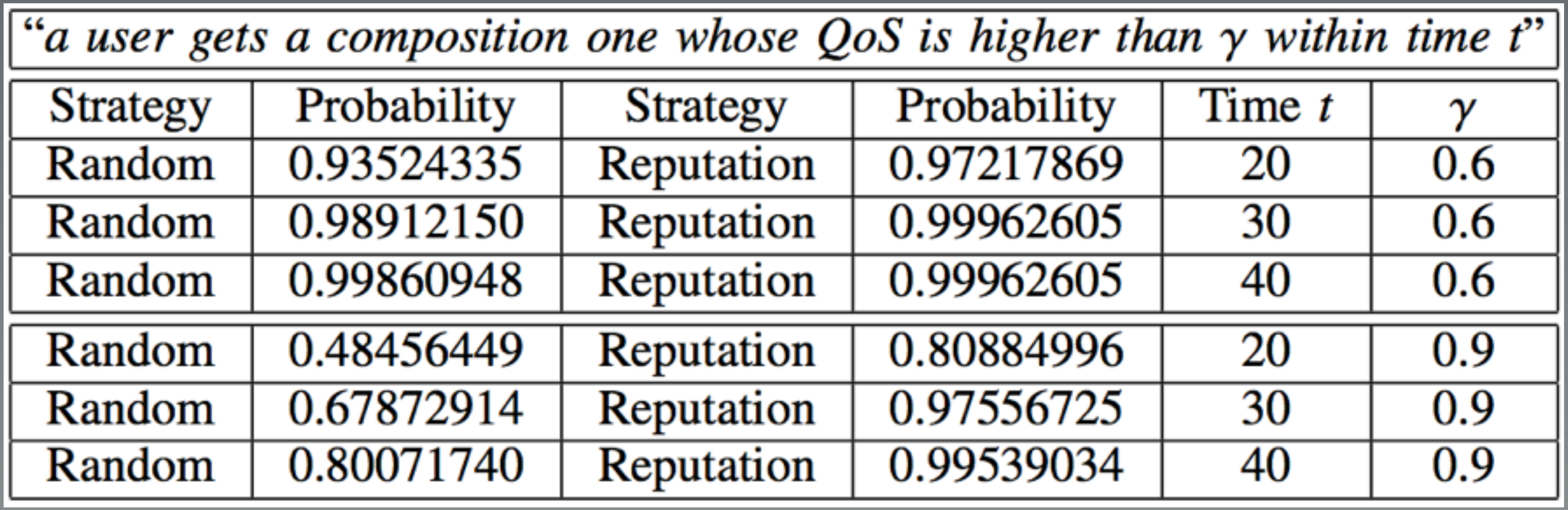
- We formalise a property in **MoSL** and verify it against the **StoKlaim** specification by means of **SAM** tool.
- **StoKlaim** is Klaim with stochastic aspects
- **MoSL** is logic language to formalise properties
- **SAM** is a tool to verify MoSL formulae

The Property

"A user gets a composition whose QoS is higher than a given threshold"

- Two different thresholds at **0.6** and **0.9**
- Composition time-frame observation at: **20**, **30** and **40** instants

Result



“a user gets a composition one whose QoS is higher than γ within time t”

Strategy	Probability	Strategy	Probability	Time t	γ
Random	0.93524335	Reputation	0.97217869	20	0.6
Random	0.98912150	Reputation	0.99962605	30	0.6
Random	0.99860948	Reputation	0.99962605	40	0.6
Random	0.48456449	Reputation	0.80884996	20	0.9
Random	0.67872914	Reputation	0.97556725	30	0.9
Random	0.80071740	Reputation	0.99539034	40	0.9

Conclusion

- In this work we adopted Klaim to formally specify the composition step of Social Web Services
- Composition have been tested between Reputation and Random SWSs selections
- In the future work we think extend SWSs rating not only to the QoS.