# Experimenting different software architectures performance techniques: a case study


**Simonetta Balsamo**[1] balsamo@dsi.unive.it

**Antinisca Di Marco**[2] adimarco@di.univaq.it

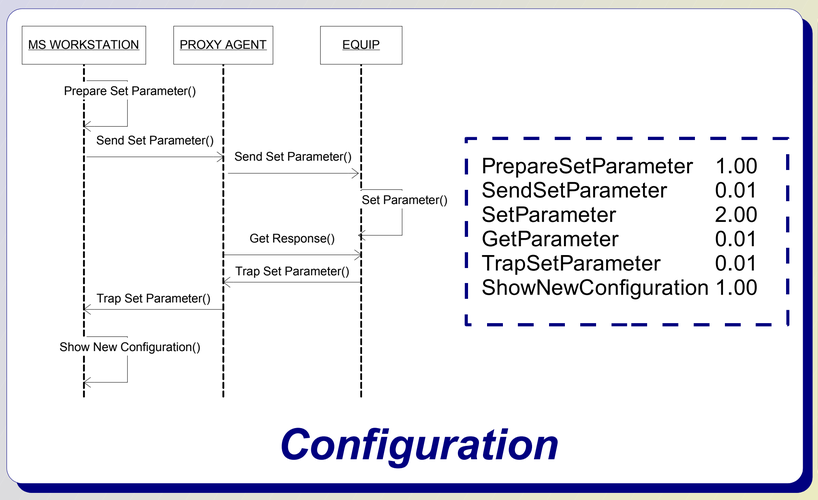
**Paola Inverardi**[2] inverard@di.univaq.it

**Moreno Marzolla**[1] marzolla@dsi.unive.it

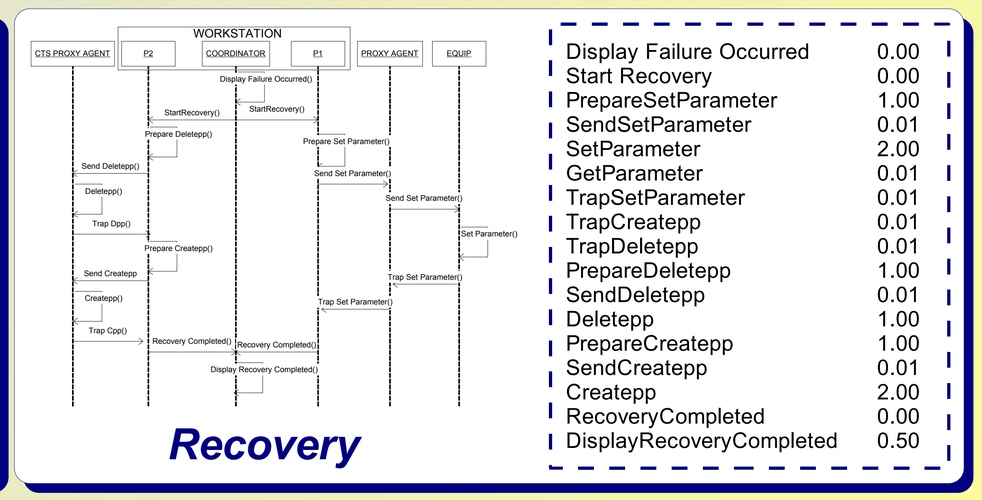
[1] Dipartimento di Informatica, Università Ca' Foscari di Venezia, via Torino 133, 30153 Mestre, ITALY
[2] Dipartimento di Informatica, Università dell'Aquila, via Vetoio 1, 67010 Coppito, L'Aquila, ITALY


## The NICE Case Study

The *Naval Integrated Communication Environment (NICE)* provides voice, data and video delivery in a naval communication environment.
We want to analyze two scenarios involving configuration of new equipments and recovery of a failed device. Sequence diagrams for the two scenarios are provided, as well as mean execution time of actions.




***Configuration***



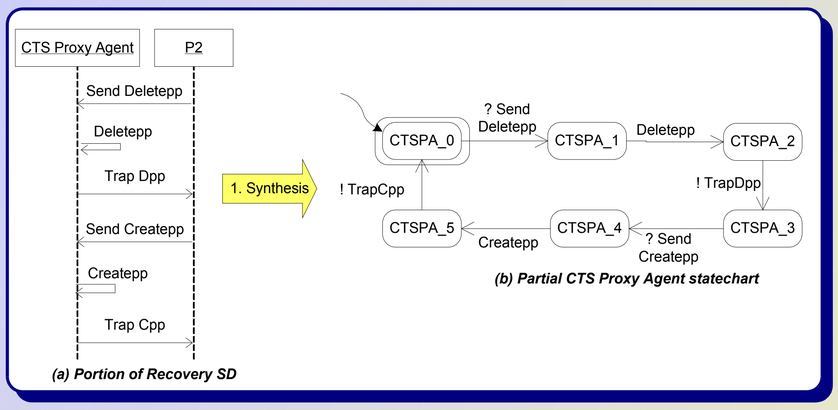

***Recovery***

The following requirements must be checked:

- ***The mean execution time of the equipment configuration has to be lower than 5 seconds (with a variance of at most 1 second)***
- ***The mean execution time of the recovery has to be lower than 5 seconds (with a variance of at most 1 second), when a failure occurs***

## AEmilia Modeling

***Synthesis*** of a partial statechart for each component of the scenario.

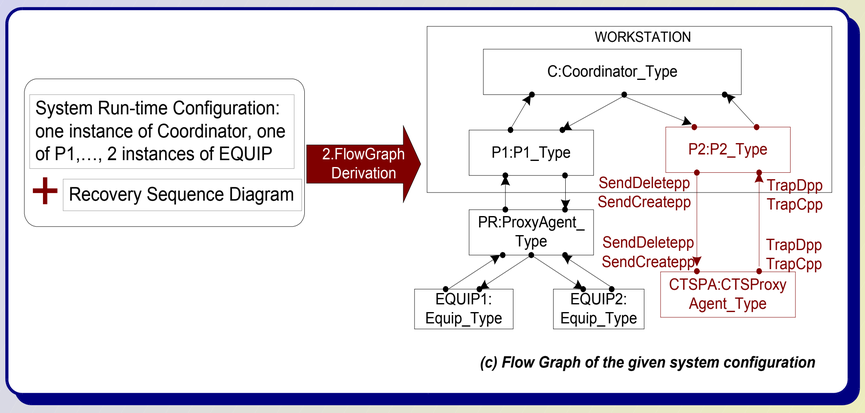



*(a) Portion of Recovery SD*

*(b) Partial CTS Proxy Agent statechart*

*Initial state*: the component waits for taking place in the scenario execution.
*State transition*: the component lifeline is involved by an arrow.
*Final state*: the component finishes its work in the scenario.

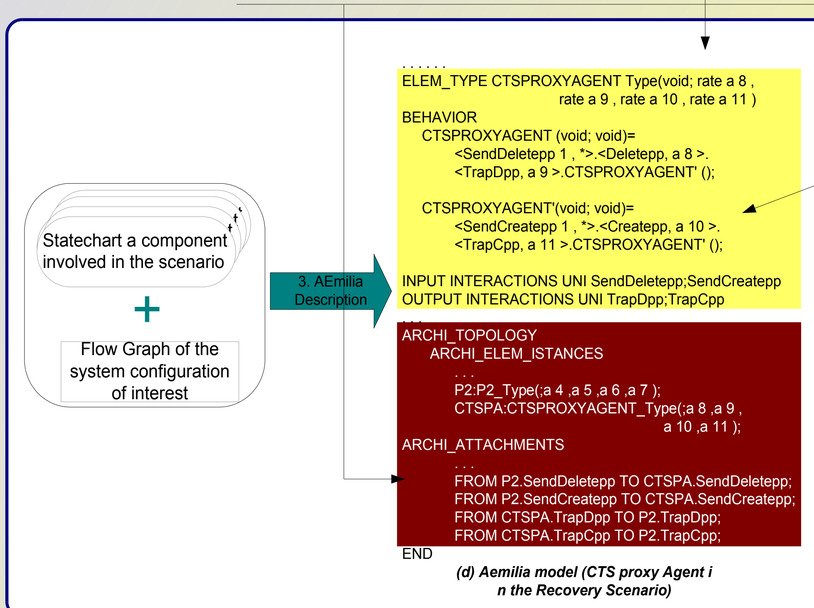
***Flow graph derivation*** from the seq. diagram and the SA configuration.




*(c) Flow Graph of the given system configuration*

*Box*: component instance. *Black circles on boxes*: input and output interactions local to the instance behavior. *Arrows:* interactions between instances represented in the scenario.

***AEmilia Textual Description***: it is derived from the statecharts, which represent the behavior evolution of components, and from the flow graph, which contains information on the interactions between instances.




*(d) Aemilia model (CTS proxy Agent in the Recovery Scenario)*

Performance information (such as execution time) are included into the component actions.

Performance indices are expressed through the reward theory.

The model is evaluated by TwoTowers tool.

## Simulation Modeling

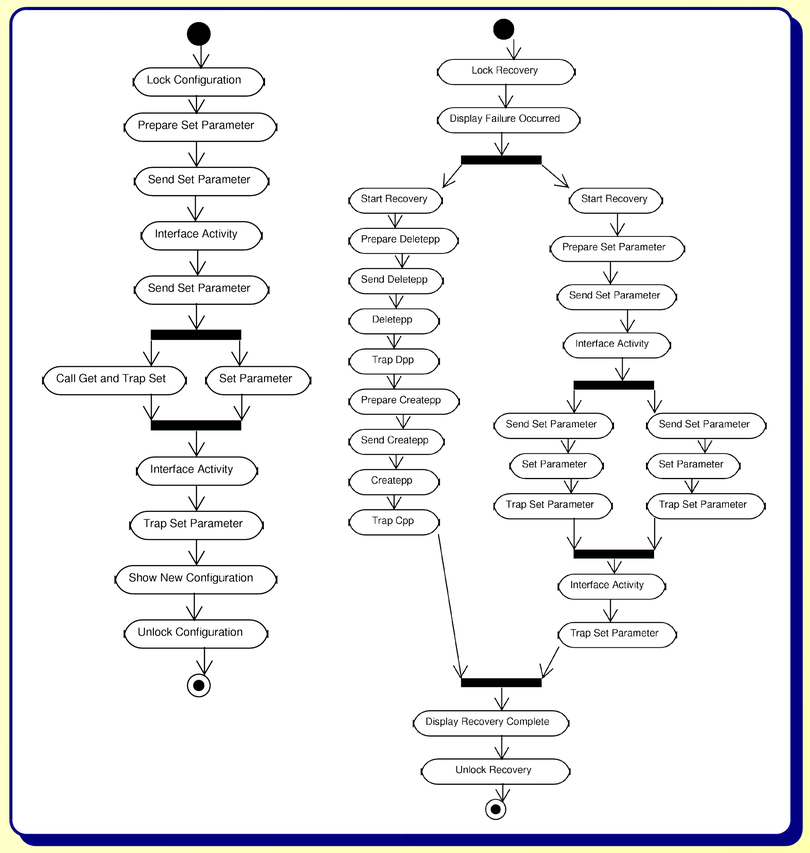
The system is described in term of Use Case, Activity and Deployment diagrams. Use Cases represent Workloads. Deployment diagrams describe physical resources (processors), and Activity diagrams represent computations performed on the processors.

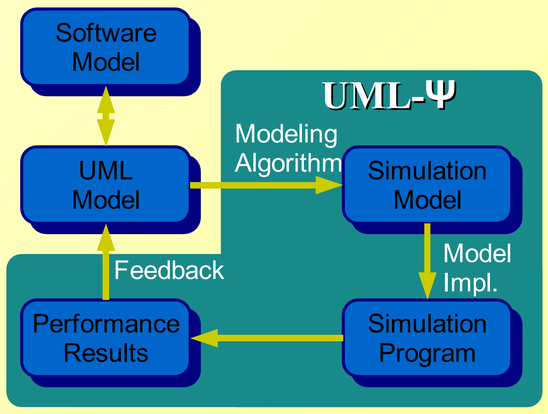
UML Diagrams are annotated according to (a subset of) the UML Profile for Schedulability, Performance and Time specification. Annotations specify the (mean) duration of each action. Parameters are set according to the mean durations specified by the system designers.

The annotated UML model is drawn with ArgoUML and exported in XMI format.

The UML-PSI tool builds a simulation program from the annotated UML model. Simulation results (mean execution time of actions and whole activity diagrams) are reported.

According to the results, the requirement on the configuration scenario is verified with less than 3 equipments. The requirement on the recovery scenario is never verified.




| *Num. Equip.* | *Configuration Scenario* | *Requirement Verified?* | *Recovery Scenario* | *Requirement Verified?* |
|---|---|---|---|---|
| 1 | 4.02 | yes | 6.61 | no |
| 2 | 4.68 | yes | 7.64 | no |
| 3 | 5.50 | yes | 10.26 | no |
| 4 | 6.87 | no | 13.70 | no |
| 5 | 8.95 | no | 17.66 | no |
| 6 | 10.94 | no | 23.97 | no |

## Conclusions

| | Simulation/UML-Ψ | Æmilia/TwoTowers |
|---|---|---|
| ***Model Derivation*** | Easy. There is an almost direct mapping between UML elements and simulation processes. | Easy due to the ADL AEmilia notation which allows performance models which tightly reflect the software specification. |
| ***Annotations*** | Use of stereotypes and tagged values to specify performance-oriented parameters following the UML Performance Profile | Parameters should be instantiated by the modeler when the performance model is executed. |
| ***Generality*** | No constraint on the software model. | Modeling some particular aspects of software systems, even if possible, can be difficult. |
| ***Computed Perf. Indices*** | Only approximate values are computed given as confidence intervals. UML-Ψ implements a set of statistical functions automatically applied to output data analysis in order to compute sound results without any user guidance. | Results are exact numerical values computed analytically. |
| ***Scalability*** | The size of the performance model increases linearly with the size of the UML model. | The size of the perf. model grows exponentially with the SA model size leading to the state space explosion even for small models. |
| ***Feedback*** | Performance results are inserted into the original UML model, as tagged values associated to the relevant model elements. | It is not easy to associate the computed performance measures to the software components to which they refer. |
| ***Integration*** | Only performance modeling is possible. | Both functional and non-functional analysis can be performed on the same AEmilia representation of the software system. |

*Table - Summary of the comparison between the simulation-based and the analytical approach.*


WOSP 04, Redwood City, CA, January 14-16 2004