A Simulation-Based Approach to Software Performance Modeling

Simonetta Balsamo
balsamo@dsi.unive.it
http://www.dsi.unive.it/~balsamo

Moreno Marzolla
marzolla@dsi.unive.it
http://www.dsi.unive.it/~marzolla

Dipartimento di Informatica
Università Ca' Foscari di Venezia
via Torino, 155
30172 Mestre (Venezia) ITALY

The Challenge
Evaluate the performances of software systems during the early design phase. Possibly without requiring the software architect to learn a new specialized notation to do that.

Motivation
Changing the design costs. The cost is higher if the change is done late during the software development cycle. The “fix it later” approach can not be applied here.

Approach
We derive a Simulation Model from a UML specification. The UML specification is annotated according to (a subset of) the UML Performance Profile. Simulation allows unconstrained representation of general time distributions, arbitrary scheduling policies, fork/join systems...

UML-ψ (UML Performance Simulator)

1. The starting point is a set of annotated UML diagrams. Currently, Use Case, Activity and Deployment diagrams are used. The diagrams are annotated with a (subset of) the annotations defined in the “UML profile for Schedulability, Performance and Time Specification”. Annotating the diagrams provides information on the service demand of each processing step, number of times a step has to be repeated, intervals between repetitions, scheduling policies and speed of the processors.

2. The UML model is translated into a simulation model. There is an almost direct mapping between UML elements and simulation processes. UML Actions represent workloads applied to the system. Use Cases represent possible execution scenarios, and the details of each scenario are represented by the associated Activity diagram which is translated into a sequence of processing steps. Node Instances in the Deployment diagram represent resources in the system. Currently only processing resources (CPUs) are considered.

3. The simulation model is implemented into a simulation program, according to the parameters given in the tagged values. We developed a custom C++ simulation library implementing basic facilities, such as SIMULA-like pseudoparallel process execution, and basic statistics collection. The simulation model is implemented using this library.

4. The simulation computes the following steady-state performance measures:
   - Execution time of each Activity
   - Overall execution time of each Use Case
   - Utilization of the resources

Results are inserted into the UML model as tagged values associated to the appropriate UML elements. In this way, the modeler obtains easily a feedback. The UML model can be changed and the evaluation process started many times until satisfactory performances are obtained.

A Simple Example

1. Drawing the UML model with ArgoUML or Poseidon.
2. The simulation model is generated by ArgoUML/Poseidon to generate the performance model.
3. Tagged Values are evaluated with a Perl interpreter.
4. Results are inserted into the UML model as tagged values.

Sketch of the Modeling Algorithm

for all Actor a in U do
   if a is tagged as OpenWorkload then
      // Make new OpenWorkload object
   else if a is tagged as ClosedWorkload then
      // Make new ClosedWorkload object
   for all Use Case u associated with a do
      Ac:=Make new ClosedWorkload object
      for all Activity s in A do
         Sc:=Make new PScenario object
         Ac:=Make new OpenWorkload object
      for all Node n in D do
         if n is an atomic step then
            p':=New PRhost_FIFO
         else if n is a fork step then
            p':=New PRhost_join object
         else if n is a join step then
            p':=New PRhost_join object
         p':=New PRhost_FIFO
         if n is tagged as ClosedWorkload then
            p':=New PRhost_FIFO
         else if n is LIFO then
            p':=New PRhost_FIFO
         else if n is PS then
            p':=New PRhost_FIFO
         Link p with p'

Note that ArgoUML and Poseidon do not store tagged values inside comments. We use here a comment-based notation for display purposes only.

ESEC/FSE 2003, Helsinki, FI, September 3-5 2003