Performability Modeling & Analysis in UML

March 2-3, 2010: PaCo second mid-term meeting (L'Aquila, Italy)

Luca Berardinelli | luca.berardinelli@univaq.it |
Dipartimento di Informatica
Università dell’Aquila (ITALY)
PACO: project macro objectives

1. Studyng of logic, models and languages for:
   - a) Modeling performability-aware systems;
   - b) Specification of performability metrics;

2. Definition of model transformation functions
   - a) From design to (multiple) analysis models (direct);
   - b) Among different analysis models (direct/inverse);
   - c) From analysis to design models (inverse);

L. Berardinelli, S. Bernardi, V. Cortellessa, and J. Merseguer,
"UML Profiles for Non-Functional Properties at Work: Analyzing Reliability, Availability and Performance",
Proc. of the 2nd International Workshop on Non-functional System Properties in Domain Specific Modeling Languages

L. Berardinelli, S. Bernardi, V. Cortellessa, and J. Merseguer,
Submitted to the Sixth International Conference on the Quality of Software Architectures (QoSA 2010)
UML Profiles for NFPs at Work

1. Studying of logic, models and languages for:
   a) Modeling performability-aware systems:
      - **Input**: UML Design Model. **Output**: Petri Net, Queuing Network, Fault Tree
      - Integration of different NFPs by I/O parameter sharing:
        Dependability (Reliability, Availability), Performance
   b) Specification of performability metrics: Not considered

2. Definition of model transformation functions
   a) From design to analysis models (direct):
      - **Unique Source** Design Model (UML), **Multiple Target** Analysis Model
      - Methodologies and Tools
   b) Among different analysis models (direct/inverse);
   c) From analysis to design models (inverse);

UML Profiles for NFPs at Work: the Framework

- **UML 2 Model**
  - Framework
  - Design Model
  - Analysis Models

**FRAMEWORK**
- UML Profiles for NFPs

**DESIGN MODEL**
- UML 2 Model
- MARTE (Model and Analysis of Real-Time and Embedded Systems)

**ANALYSIS MODELS**
- Reliability Model
- Availability Model
- Fault Tree (FT)
- Generalized Stochastic Petri Net (GSPN)
- Queuing Network (QN)

**METHODOLOGY**
- DAM (MARTE-based Dependability Analysis and Modeling)
UML Profiles for NFPs at Work: MARTE

- **UML lightweight extension (profile) for Modeling and Analysis of Real-Time Embedded systems**
- Allows the Design of Software and Hardware Resources
- Allows the specification of NFPs of Sw/Hw Resources using the Value Specification Language (VSL) for
  - Generic Quantitative Analysis (GQAM)
  - Schedulability Analysis (SAM)
  - **PERFORMANCE ANALYSIS (PAM)**
UML Profiles for NFPs at Work: DAM

- UML lightweight extension (profile) for Dependability Analysis Modeling.
- **Built upon MARTE**: it reuses MARTE Foundation and Generic Quantitative Analysis Model (GQAM)
- It support the specification of dependability properties and requirements such as **Reliability**, **Availability**, Maintainability and Safety
A doctor provided with his/her generic computing device (e.g. PDA, laptop), through **medical distributed services**, is able to:

**Make Prescription**: after visiting the patient, the doctor can make a prescription to be sent to the hospital where eventually the patient will take the medicines.

This service requires some particular **Availability**, **Reliability** and **Performance** requirements.

The eHealth System is equipped with Message Redundancy Systems:

**Redundancy Manager**:
(i) create a Message Replicator for each prescription sent to the Application Server

**Message Replicator**:
(i) replicates n-times the prescription message,
(ii) create and destroy Payloads
(iii) assign each replica to a different Payload
(iv) calculate the voting result.

**Payload**:
(i) scan & decipher message replicas and
(ii) votes for replica integrity.
UML Profiles for NFPs at Work: UML Model

UML DESIGN MODEL (without annotation)

Use Case Diagram

State Machine

Sequence Diagram

Component Diagram

Deployment Diagram
UML Profiles for NFPs at Work: Reliability

1. Annotate $\rightarrow$ 2. Transform $\rightarrow$ 3. Solve

Model-Driven Analysis Methodology

RELIABILITY ON DEMAND:

The probability of a system working within specifications
for a certain number of invocations without system-level repair.

Vittorio Cortellessa, Harshinder Singh, and Bojan Cukic.

Early reliability assessment of UML based software models.

Probability of System Failure

$$\theta_S = 1 - \sum_{k=1}^{K} \prod_{i=1}^{N} (1 - \theta_i)^{bp_{ik}} \cdot \prod_{(i,j)} (1 - \psi_{ij})^{\text{interact}(i,j,k)}$$

NFPs (i.e. UML annotations)

- $\theta_i$ = failure prob of component $i$
- $bp_{ik}$ = busy periods (i.e. activation) of component $i$ in scenario $k$
- $\psi_{ij}$ = failure prob of hardware connector among remote hosting nodes of components $i$ and $j$
- interact$_{i,j,k}$ = interactions (# of invocations) between components $i$ and $j$ in scenario $k$
- $p_k$ = execution probability of scenario $k$

It can be represented using a FAULT TREE.
UML Profiles for NFPs at Work: Reliability

1. Annotate (MARTE+DAM) → 2. Transform → 3. Solve

UML DESIGN MODEL (with Reliability Annotation)

**Component Diagram**
\[ \theta_i = \text{failure prob of component } i \]

**Deployment Diagram**
\[ \psi_{i,j} = \text{failure prob of hardware connector} \]

**Sequence Diagram**
\[ \text{interact}_{i,j,k} = \text{message exchanged between components } i \text{ and } j \text{ in scenario } k \]

**DAM Profile**
- DaComponent
  - Client
  - {failure = "occurrenceProb = $intfClient"}

**DAM Profile**
- DaConnector
  - WAN
  - {fault = "(occurrenceProb=(value=$ft_probWAN))"}

**Use Case Diagram**
\[ p_k = \text{execution probability of scenario } k \]

**DAM Profile**
- MakePrescription
  - {execProb = "0.18"}

**DAM Profile**
- \( \theta_i \)
  - \( \text{failure prob of component } i \)

**DAM Profile**
- \( \psi_{i,j} \)
  - \( \text{failure prob of hardware connector} \)

**DAM Profile**
- \( \text{interact}_{i,j,k} \)
  - \( \text{message exchanged between components } i \text{ and } j \text{ in scenario } k \)

**DAM Profile**
- \( \text{bp}_{i,k} \)
  - \( \text{# of busy periods (i.e. activation) of component } i \text{ in scenario } k \)
UML Profiles for NFPs at Work: Reliability

Sensitivity w.r.t. $\psi_{\text{WAN}}$ = failure prob of WAN connector

1. Annotate $\rightarrow$ 2. Transform $\rightarrow$ 3. Solve

UML Design Model

Reliability Target Formalism (Mathematical Expression)

$$ (1 - \theta_{\text{Client}})^2 \cdot (1 - \theta_{\text{MsgReplicator}})^4 \cdot (1 - \theta_{\text{RedundancyMgr}})^f(N) \cdot (1 - \psi_{\text{Client-MsgReplicator}})^2 $$

<table>
<thead>
<tr>
<th>FIXED INPUT PARAMETER</th>
<th>$ft_probWAN$</th>
<th>Make Prescription Reliability on Demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>$ft_probPDA$ = 0.001</td>
<td>0.01</td>
<td>97.80%</td>
</tr>
<tr>
<td>$ft_prob\text{AppHost}$ = 0.000001</td>
<td>0.1</td>
<td>81.00%</td>
</tr>
<tr>
<td>$N$ = 3 (i.e. multiplicity of Payload Comps)</td>
<td>1</td>
<td>0.00%</td>
</tr>
</tbody>
</table>
UML Profiles for NFPs at Work: **Availability**

**Availability**:
The **probability** that the system is up and running to deliver its service to users when they request them.

**Model-Driven Analysis Methodology**


**AVAILABILITY**:

A compositional semantics for UML State Machines aimed at performance evaluation.


NFPs (i.e. UML annotations)

- Service availability requirement;
- Software Failure (no fault masking mechanism) and Recovery rates;
- Workload specification for service behaviors;
- Hardware Fault (propagates to sw failure) and Recovery rates of hardware nodes.
UML Profiles for NFPs at Work: Availability

1. Annotate (MARTE+DAM) \(\rightarrow\) 2. Transform \(\rightarrow\) 3. Solve

DESIGN MODEL (with Availability Annotation)

**Component Diagram**

**Message Replicator Component**

- **WaitForMessage**
  - **Fail Event**
    - `failure = "(Fcause=(occuranceRate=(value=\$ft_rateNode, unit=ft/s))), kind = failure)"
  - **Restart**
    - `kind = reallocation, recovery = "rate=(value=(\$re_rate1, unit=rec/s)), map=(MR\text{;}onto=(AppHost_node1)))"`

**Deployment Diagram**

- **HW Fault and Recovery rates of execution and comm. hosts**

- **<DaConnector>**
  - **WAN**
    - `fault = "(persistency=transient; occurrenceProb=(value=\$ft_probWAN); duration=(value=\$ft_durWAN, unit=s))"

**Use Case Diagram**

- **Service availability requirement**
  - **MakePrescription**
    - `{ssAvail = "(value=99\%, statQ=min, source=req)"}`

**State Machine**

- **Software Failure and Recovery rates of components**

**Sequence Diagram**

- **Workload specification of system services**

- **MakePrescription()**
  - `{pattern = "closed (population=\$pop, extDelay=(3.3))"}

- **SendPrescription2MR(-,-)**

**$ft_probWAN - INPUT Parameter sharing with Reliability Analysis**
UML Profiles for NFPs at Work: **Availability**

1. Annotate $\rightarrow$ **2. Transform** $\rightarrow$ 3. Solve

**ANALYSIS MODEL (gspn)**

- Execution Host Failures (subnet) cause failure of deployed Component (subnet)

---

**Legend (parameter specification)**

<table>
<thead>
<tr>
<th>Transition</th>
<th>Type</th>
<th>Rate/Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftPDA</td>
<td>EXP</td>
<td>ft_ratePDA</td>
</tr>
<tr>
<td>ft_node</td>
<td>EXP</td>
<td>ft_rateNode</td>
</tr>
<tr>
<td>ftDuration1, ftDuration2</td>
<td>EXP</td>
<td>1/ft_durWAN</td>
</tr>
<tr>
<td>reNode</td>
<td>EXP</td>
<td>re_rate1</td>
</tr>
<tr>
<td>rePDA</td>
<td>EXP</td>
<td>re_rate2</td>
</tr>
<tr>
<td>think</td>
<td>EXP</td>
<td>1/(3s.)</td>
</tr>
<tr>
<td>result</td>
<td>EXP</td>
<td>1/0.5834s</td>
</tr>
<tr>
<td>trKO1, trKO2</td>
<td>IMM</td>
<td>ft_probWAN</td>
</tr>
<tr>
<td>trOK1, trOK2</td>
<td>IMM</td>
<td>1-ft_probWAN</td>
</tr>
</tbody>
</table>
UML Profiles for NFPs at Work: **Availability**

Sensitivity analysis w.r.t. fault rate probability of **WAN connector**
(no masking mechanism then hw fault $\rightarrow$ hw failure)

The **Service Availability** is defined as the
**PROBABILITY THAT ALL DOWN PLACES**
MODELING THE DOWN STATES ARE **ALL EMPTY**

**FIXED INPUT PARAMETER**

<table>
<thead>
<tr>
<th>$ft_probWAN$</th>
<th>AVAIL</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>99.98%</td>
</tr>
<tr>
<td>0.1</td>
<td>99.88%</td>
</tr>
<tr>
<td>1</td>
<td>99.38%</td>
</tr>
</tbody>
</table>

- **$ft\_ratePDA$** = $ft\_rateNode$ = 30 CRASH FAULT/ YEAR FOR **HW COMPS**
- **$rec\_PDA$** = $rec\_Node$ = 60 SEC RECOVERY MEAN DURATION FOR PDA
- **$ft\_durWAN$** = 1 HOUR OF WAN COMMUNICATION MEAN DOWN TIME
PERFORMANCE:
The probability that the system is up and running to deliver its service to users when they request them

Antinisca Di Marco and Paola Inverardi.

Compositional Generation of Software Architecture Performance QN Models.

NFPs (i.e. UML annotations)

- Scheduling Policies of Software Components and Resource Demand (Time) for their Operations,
- Workload specification of service behaviors;
UML Profiles for NFPs at Work: Performance

1. Annotate (MARTE+DAM) → 2. Transform → 3. Solve

DESIGN MODEL (with Availability Annotation)

Component Diagram

Use Case Diagram

Deployment Diagram

GaWorkloadEvent - INPUT Parameter sharing with Availability. Analysis
UML Profiles for NFPs at Work: **Performance**

1. Annotate $\rightarrow$ 2. Transform $\rightarrow$ 3. Solve

**ANALYSIS MODEL (QN)**

- Component $\rightarrow$ Multiqueue Service Center; Component Operation $\rightarrow$ Queue
UML Profiles for NFPs at Work: Performance

Sensitivity analysis w.r.t. workload characteristics (e.g. population)

INPUT/OUTPUT Parameter sharing with Availability Analysis

Simulation (MVA Analysis not possible due to Fork&Join)

<table>
<thead>
<tr>
<th>$\text{pop}$</th>
<th>$\text{out$RT$}$</th>
<th>$\text{pop}$</th>
<th>$\text{out$RT$}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.58333333</td>
<td>6</td>
<td>0.74803449</td>
</tr>
<tr>
<td>2</td>
<td>0.61200189</td>
<td>7</td>
<td>0.78952187</td>
</tr>
<tr>
<td>3</td>
<td>0.64261899</td>
<td>8</td>
<td>0.83276521</td>
</tr>
<tr>
<td>4</td>
<td>0.67536272</td>
<td>9</td>
<td>0.87926688</td>
</tr>
<tr>
<td>5</td>
<td>0.71042945</td>
<td>10</td>
<td>0.93027834</td>
</tr>
</tbody>
</table>
UML Profiles for NFPs at Work: conclusion

1. Studyng of logic, models and languages for:
   a) Modeling performability-aware systems:
      - **Input:** UML Design Model+Profiles. **Output:** GSPN, QN, FT
      - Integration of different NFPs by I/O parameter sharing:
        Dependability (Reliability, Availability), Performance
   b) Specification of performability metrics: Not considered

2. Definition of model transformation functions
   a) From design to analysis models (direct):
      - **Unique Source** Design Model (UML), **Multiple Target** Analysis Model
      - **Methodologies** and **Tools**
   b) Among different analysis models (direct/inverse);
   c) From analysis to design models (inverse);

L. Berardinelli, S. Bernardi, V. Cortellessa, and J. Merseguer,
"UML Profiles for Non-Functional Properties at Work: Analyzing Reliability, Availability and Performance",
Proc. of the 2nd International Workshop on Non-functional System Properties in Domain Specific Modeling Languages

04/03/2010
Performability Modeling and Analysis in UML 20
The FEF Chain: M&A Performability in UML SA

1. Studyng of logic, models and languages for:
   a) Modeling performability-aware systems:
      - **Input**: UML Design Model+Profiles+FEF PAttern. **Output**: GSPN, QN, FT
      - Integration of different NFPs by I/O paramter sharing:
        Dependability (Reliability, Availability), Performance
   b) Specification of performability metrics: **Mean Service Time** of System subject
to working/repairing/working cycles

2. Definition of model transformation functions
   a) From design to analysis models (direct):
      - **Unique Source** Design Model (UML), **Unique Target** Analysis Model
      - **Methodologies** and **Tools**
   b) Among different analysis models (direct/inverse);
   c) From analysis to design models (inverse);

L. Berardinelli, S. Bernardi, V. Cortellessa, and J. Merseguer,
The FEF Chain: M&A Performability in UML SA

General Modeling & Analysis Process for quantitative analysis of SA:

1. **Annotate** a software model with appropriate performability parameters (**UML Model + stereotypes**)

2. **Transform** the software model in a performability model (**UML2GSPN**)

3. **Solve** the performability model (**GSPN**) and get the result

0. Define a Failure Model to be applied (reliability)
PERFORMABILITY: The **probability** that the system is up and running to deliver its service to users when they request them

L. Berardinelli, S. Bernardi, V. Cortellessa, and J. Merseguer,
Submitted to the Sixth International Conference on the Quality of Software Architectures (QoSA 2010)

**NFPs (i.e. UML annotations)**

- Failure Model
- Workload specification of service behaviors;
- Software Fault, Error, Failure probabilities and Error Detection/Isolation duration
The FEF Chain: M&A **Performability** in UML SA

0. Failure Model → 1. Annotate → 2. Transform → 3. Solve

- **Type of fault**: Single Failure Mode (info content failure)
- **Modeling System Behavior**: generic FEF chain

---

**Execution fragments**

**Predicates on sw c. status**

**Control flow**

**Events**

\[
\begin{align*}
\text{No Fault:} & \quad \forall c \in C_i \mid State_c = \text{Correct} \\
\text{Fault in } C_i: & \quad \forall c \in C_i \mid State_c = \text{Correct} \\
\text{Normal Scenario:} & \quad \text{PRE} \\
\text{Faulty } C_i \text{ Scenario:} & \quad \text{PRE} \\
\text{Error in } C_i: & \quad \exists c_i \in C_i \mid State_{c_i} = \text{Error} \\
\text{No Error in } C_i: & \quad \forall c_i \in C_i \mid State_{c_i} = \text{Correct} \\
\text{Detection in } C_i: & \quad \exists c_i \in C_i \mid State_{c_i} = \text{FaultMasked} \\
\text{No Detection in } C_i: & \quad \forall c_i \in C_i \mid State_{c_i} = \text{Correct} \\
\text{Error Detection & Isolation Mechanism:} & \quad \exists c_i \in C_i \mid State_{c_i} = \text{Error} \\
\text{No Propagation from } C_i: & \quad \forall c_i \in C_i \mid State_{c_i} = \text{Correct} \\
\text{Propagation from } C_i: & \quad \exists c_i, c_j \in C_i \mid State_{c_i} = \text{Error} \\
\text{Normal Scenario:} & \quad \text{POST} \\
\text{Failure:} & \quad \exists c \in C_{\text{output}} \mid State_c = \text{Error} \\
\text{Success:} & \quad \exists c \in C_{\text{output}} \mid State_c = \text{Error} \\
\end{align*}
\]
The FEF Chain: M&A Performability in UML SA

0. Failure Model → 1. Annotate → 2. Transform → 3. Solve

- Correct path
- Error detected and isolated
- Failure
- Fault M asked
- Error NOT detected BUT NOT Propagated

**generic FEF chain**

- No Fault
- Fault in $c_i$

**Normal Scenario**
- $\forall c \in C, \text{State}_c = \text{Correct}$

**Faulty $c_i$ Scenario**
- $\exists c \in C_i, \text{State}_c = \text{Error}$
- $\forall c_{\text{rest}} \in C, \text{State}_{c_{\text{rest}}} = \text{Correct}$

**Detection in $c_i$**
- No Detection in $c_i$
- Detection in $c_i$

**Error Detection & Isolation Mechanism**
- No Propagation from $c_i$
- Propagation from $c_i$

**Success**
- $\exists c \in C_{\text{output}}, \text{State}_c = \text{Error}$

**Failure**
- $\exists c \in C_{\text{input}}, \text{State}_c = \text{Error}$
The FEF Chain: M&A Performability in UML SA

A doctor provided with his/her generic computing device (e.g. PDA, laptop), through medical distributed services, is able to:

**Make Prescription**: after visiting the patient, the doctor can make a prescription to be sent to the hospital where eventually the patient will take the medicines.

This service requires some particular **Availability, Reliability** and **Performance** requirements.

The eHealth System is equipped with Message Redundancy Systems:

**Redundancy Manager**:  
(i) create a Message Replicator for each prescription sent to the Application Server

**Message Replicator**:  
(i) replicates n-times the prescription message,  
(ii) create and destroy Payloads  
(iii) assign each replica to a different Payload  
(iv) calculate the voting result.

**Payload**:  
(i) scan & decipher message replicas and  
(ii) votes for replica integrity.
The FEF Chain: M&A Performability in UML SA

0. Failure Model → 1. Annotate → 2. Transform → 3. Solve

UML DESIGN MODEL

Sequence Diagram

(FEF pattern)
The FEF Chain: M&A Performability in UML SA

0. Failure Model ➔ 1. Annotate ➔ 2. Transform ➔ 3. Solve
The FEF Chain: M&A Performability in UML SA

0. Failure Model \rightarrow 1. Annotate \rightarrow 2. Transform \rightarrow 3. Solve

Workload specification of service behaviors;
Software Fault, Error, Failure probabilities and Error Detection/Isolation duration
The FEF Chain: M&A Performability in UML SA

0. Failure Model → 1. Annotate → 2. Transform → 3. Solve

Fig. 7. GSPN model: main scenario (A), post-error propagation scenario (B)
The FEF Chain: M&A Performability in UML SA


Model-Driven Analysis Methodology

Performability Target Formalism
(Generalized Stochastic Petri-Net)

Mean time to process correctly the makeprescription request (MST) under payload fault assumption (single/content failure model)

- pop = population (i.e. # of doctor that concurrently requires prescriptions)
- X_reinit = Throughput of transition re_init (termination of main scenario, see GSPN)
- X_S_fail = Throughput of transition S_fail (error propagation from RM to MR, see ErrorProp SD)
- think = external delay

\[ MST = \frac{pop}{(X_{\text{reinit}} - X_{\text{S\_fail}}) - \text{think}} \]

Parameter Name | UML Notation | Dia | Var | Value/[Range] | Type
--- | --- | --- | --- | --- | ---
fault probability | DAM::DaStep.fault | SD | Sft | [0.1-90] | %
error probability | DAM::DaStep.error | SD | Serr | [0.1-90] | %
error prop. P RM | DAM::DaConnector.errorProp | SD | Serrprop | [0.1-90] | %
error prop. RM-MR (binomial) | DAM::DaStep.failure | SD | Sftail | [3E-18; 8.19E-1] | %
The FEF Chain: M&A Performability in UML SA

Model-Driven Analysis Methodology

UML Design Model → 1. Annotate → 2. Transform → Solve → Performability Target Formalism (Generalized Stochastic Petri-Net)

0. Failure Model → 1. Annotate → 2. Transform → 3. Solve

MST = 1 sec

MST = 7 sec

MST = 1,005 sec

MST = 1,8 sec

Fig. 8. Mean service time under different fault assumptions.

pop = 1
think = 3s

GreatSPN (Markovian solver)
**The FEF Chain: M&A Performability in UML SA**

**Model-Driven Analysis Methodology**

UML Design Model → 0. Failure Model → 1. Annotate → 2. Transform → 3. Solve → Performability Target Formalism

- Generalized Stochastic Petri-Net

**Fig. 9.** Mean service time under different fault and workload assumptions.

- MST = 2.19 sec
- MST = 10.19 sec

**GreatSPN**
( Simulator: accuracy = 15%, confidence interval 99%)

pop = [1,50]
think = 3s
Future works

(i) Experiment the approach on different mechanisms for error masking that take very different time to be executed.

(ii) Extend the Failure Model to multiple failures and/or different type of failures. Consequently the UML pattern should change.