Most Influential Paper Award presented at MODELS-2021 10 years after publication in SoSyM 2011

A dependability profile within MARTE

Software & Systems Modeling, 10(3): 313-336 (2011)

Simona Bernardi[1], José Merseguer[1], Dorina C. Petriu[2]
[1] University of Zaragoza, Spain
[2] Carleton University, Ottawa, Canada

Talk Outline

- Part I: Paper contributions
  - Context
  - Goals
  - Approach
  - Application to a case study
- Part II: Influence of the paper
  - Citations
  - Influence on our own work
    - Theses supervised
    - Publications (book, journal and conference papers)
    - Tool support
  - Future challenges
    - Integrating the analysis of multiple NFPs in the MDE process
    - Standardization

Dependability evaluation

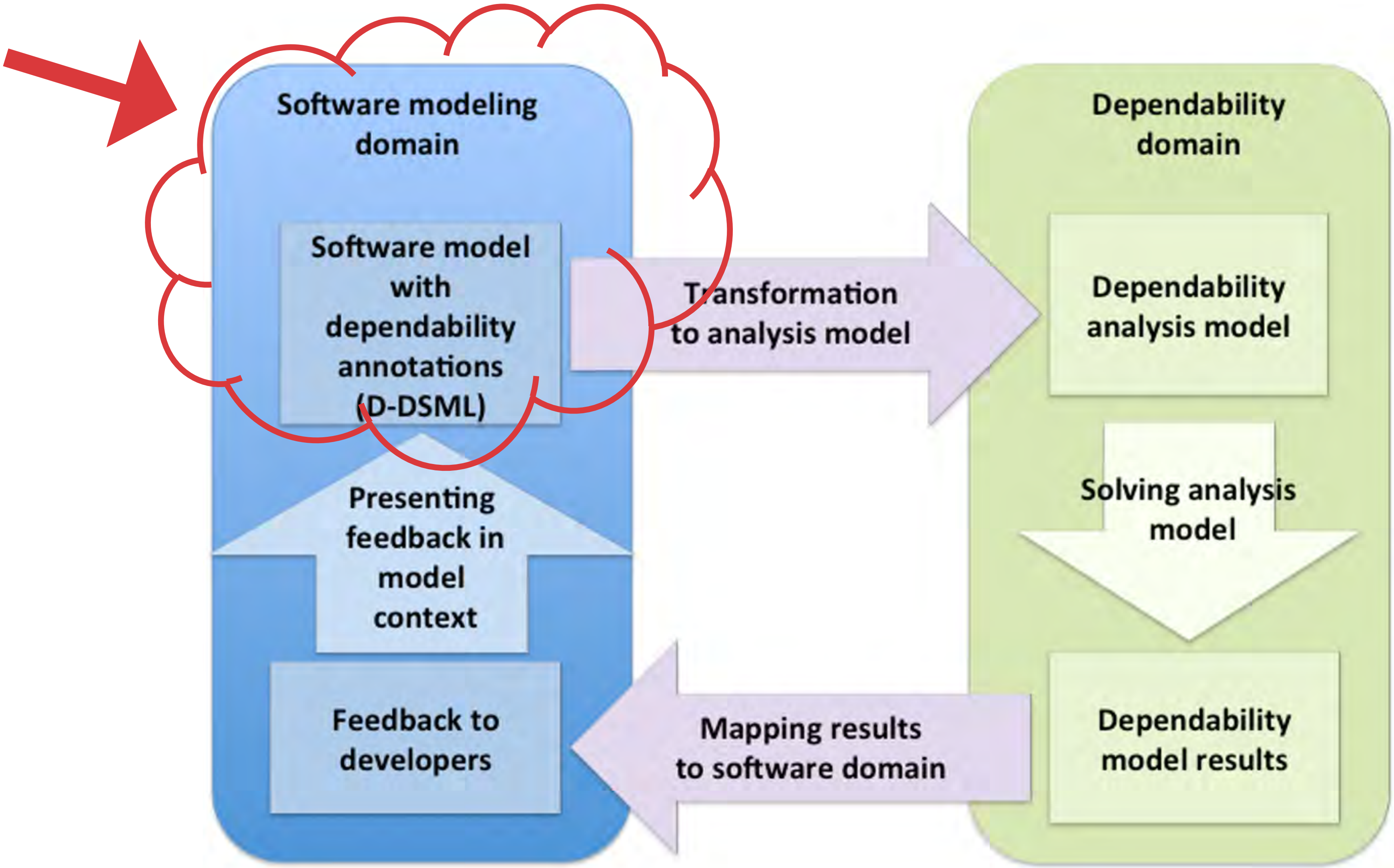
Fault Forecasting:

Means to estimate the present number, the future incidence, and the likely consequences of faults* [Avizienis et al. 2004]

- **Conducted by carrying out an evaluation of the system behavior w.r.t. fault occurrence**
  - **Qualitative: identify, classify and rank failure modes, causes-effect analysis**
  - **Quantitative: probabilistic/stochastic evaluation of dependability measures via modeling and testing**

*A. Avizienis, J.C. Laprie, B. Randell, C.E. Landwehr: Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Trans. Dependable Secur. Comput. 1(1): 11-33 (2004)

Integration of dependability modeling and analysis in Model-Driven Software Engineering



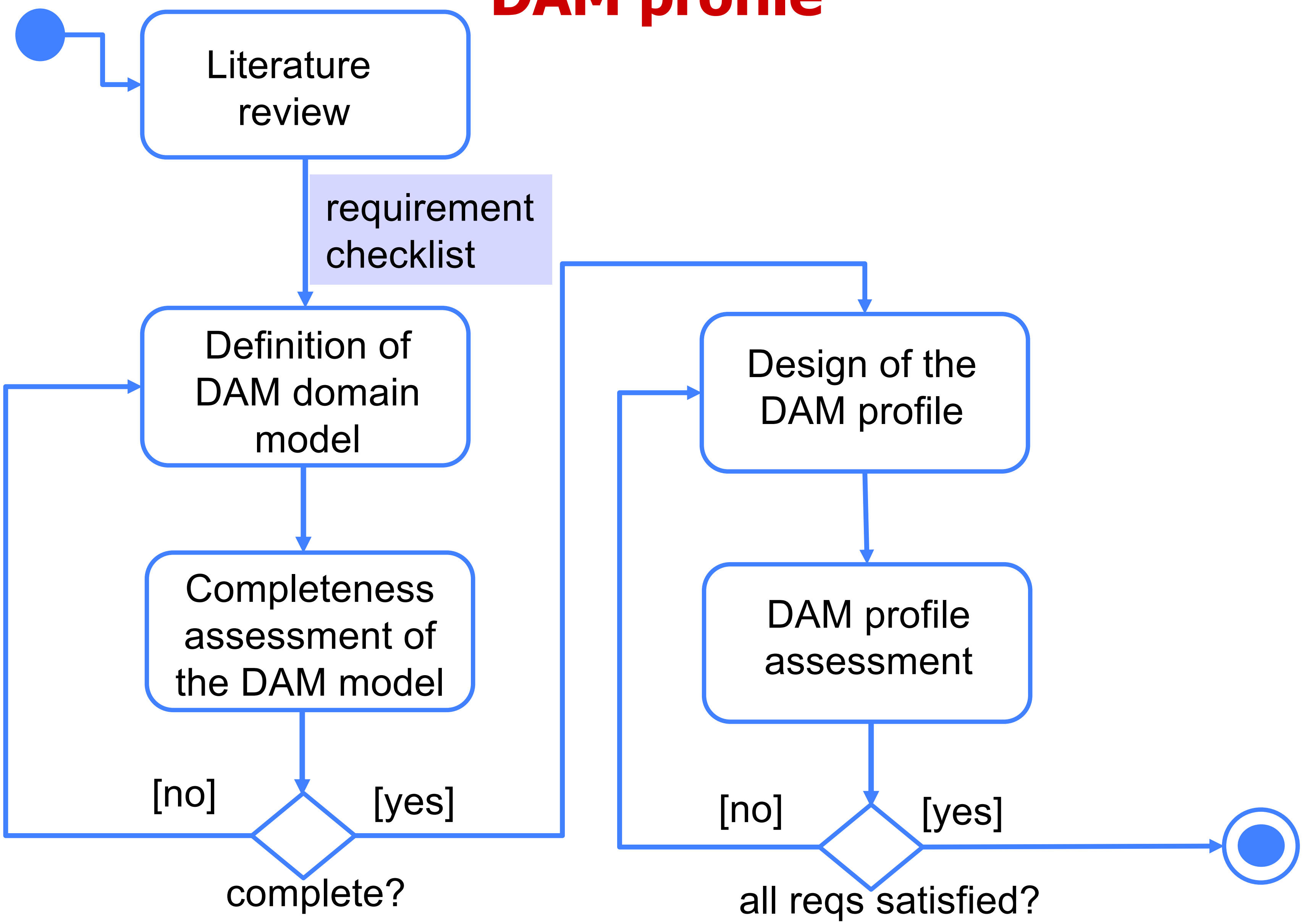
Use of UML profiling to define DSML for the assessment of Non-Functional Properties

- Construction of formal models for the assessment of NFP of software systems [from '99 to today]
  - Specified in UML
  - From the early phases of the life-cycle
- Such contributions lead to the definition of standard OMG UML profiles for modelling and analysis
  - Performance & schedulability (UML SPT [2005], UML MARTE[2011])
  - QoS characteristics (UML QoS&FT [2008]
  - Dependability for safety consumer devices (UML DAF [2016])

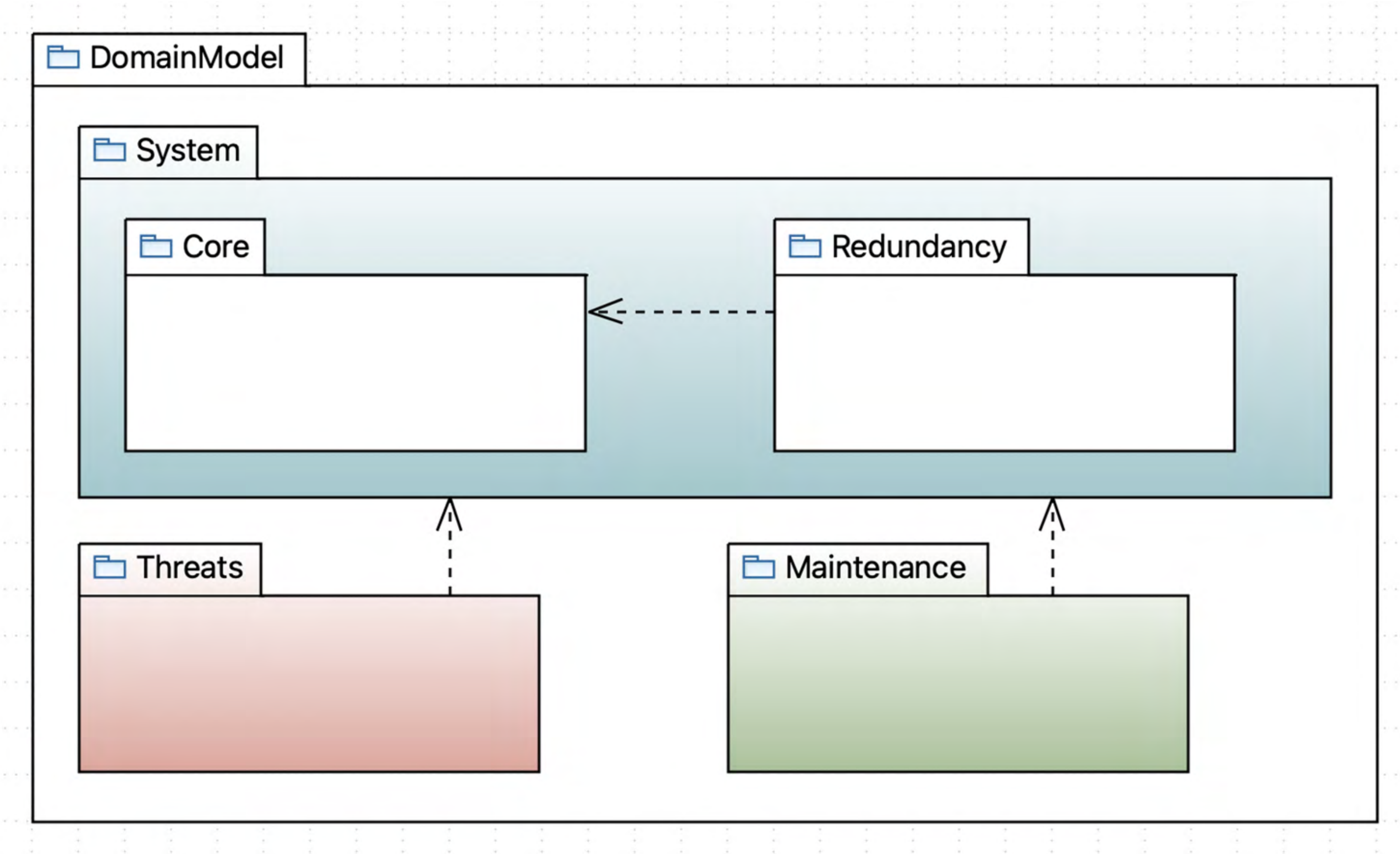
Aim of the Dependability Analysis and Modeling (DAM) profile

- **Define a UML Profile to support the dependability evaluation of software systems with focus on RAMS properties**
  - **Reliability: the continuity of correct service delivery**
  - **Availability: the promptness of correct service delivery**
  - **Maintainability: the capability to undergo modifications and repairs**
  - **Safety: the absence of catastrophic consequences on the users and environment**
- **Unify the terminology and concepts for different dependability aspects under a common dependability domain model**

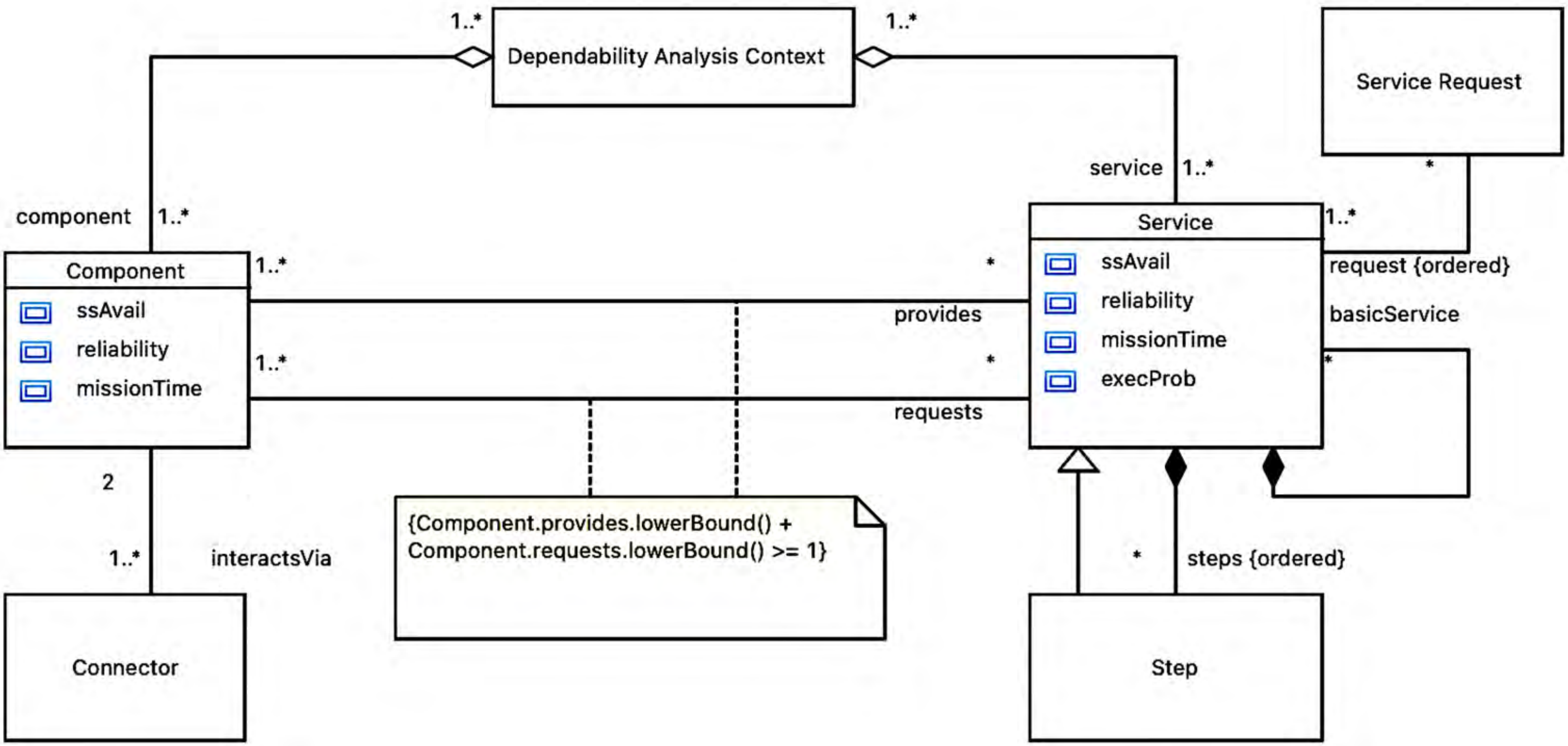
Methodological approach for the definition of the DAM profile



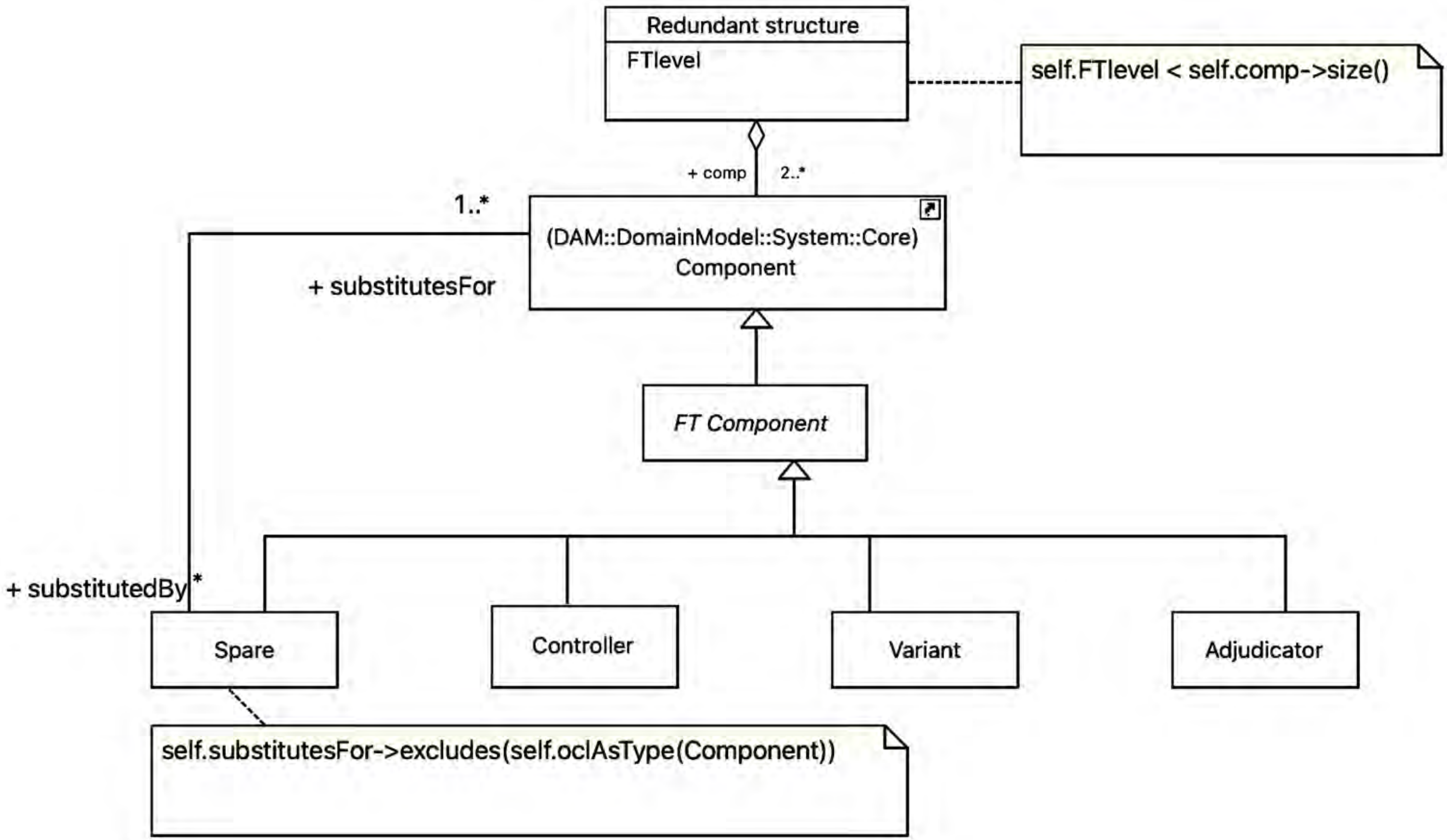
DAM domain model: Package overview



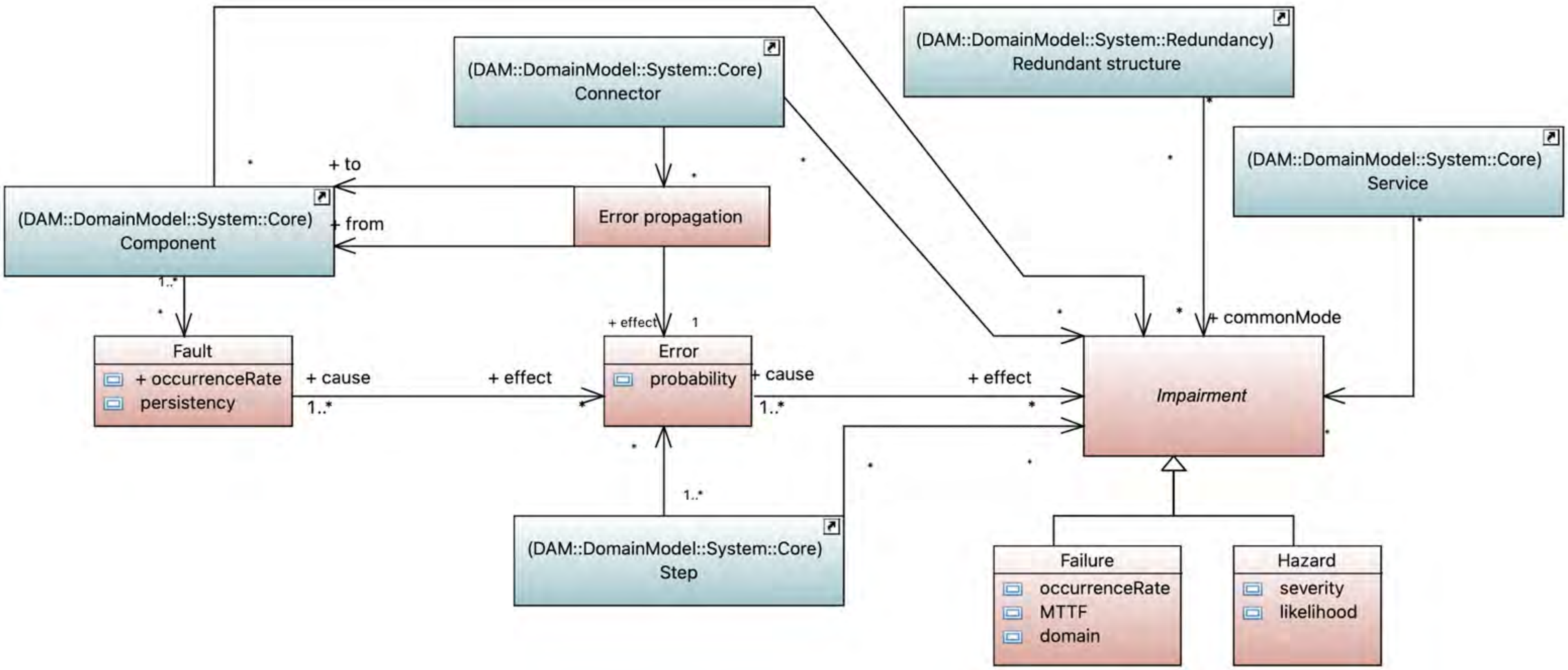
DAM domain model: Core



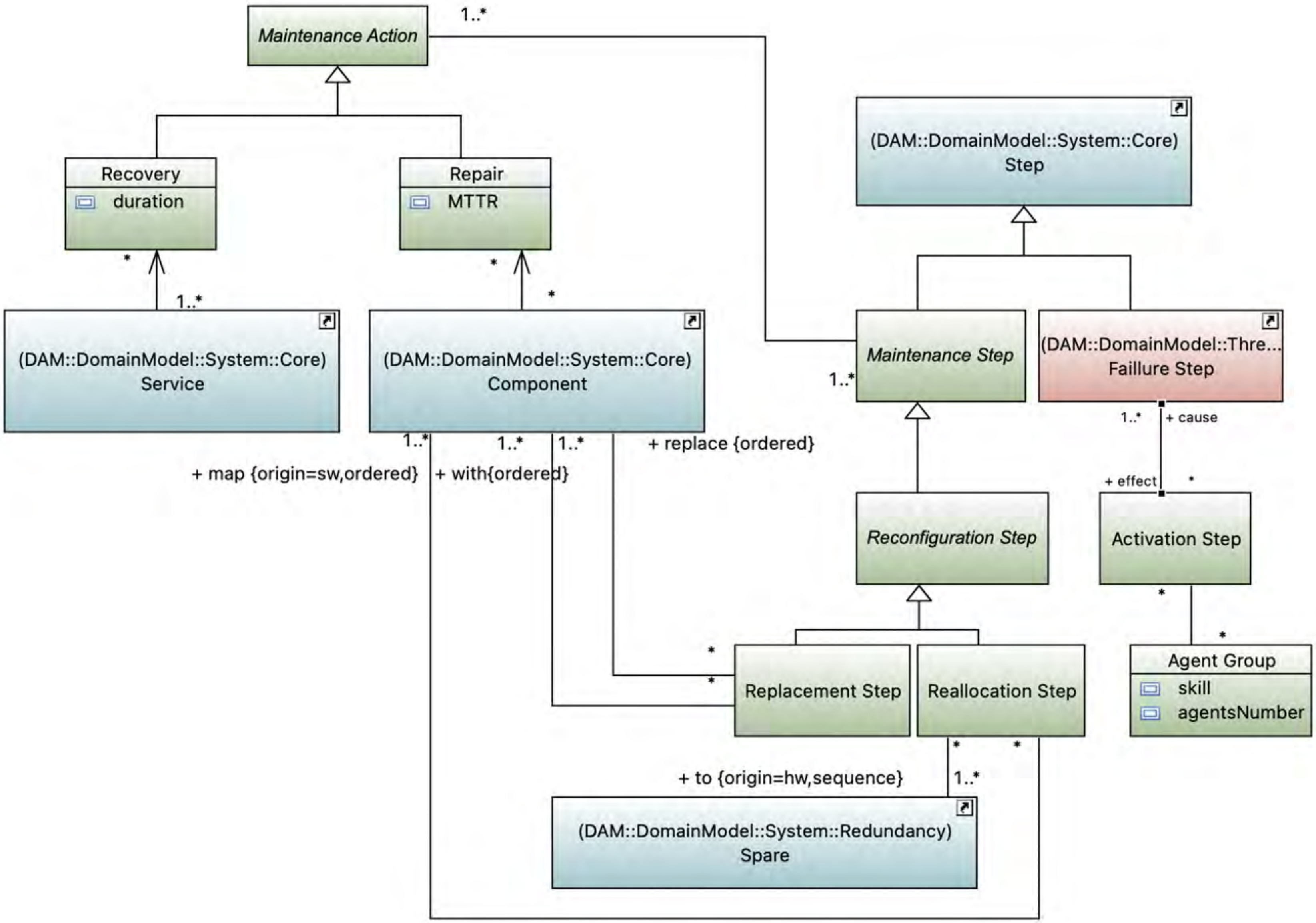
DAM domain model: Redundancy



DAM domain model: Threats

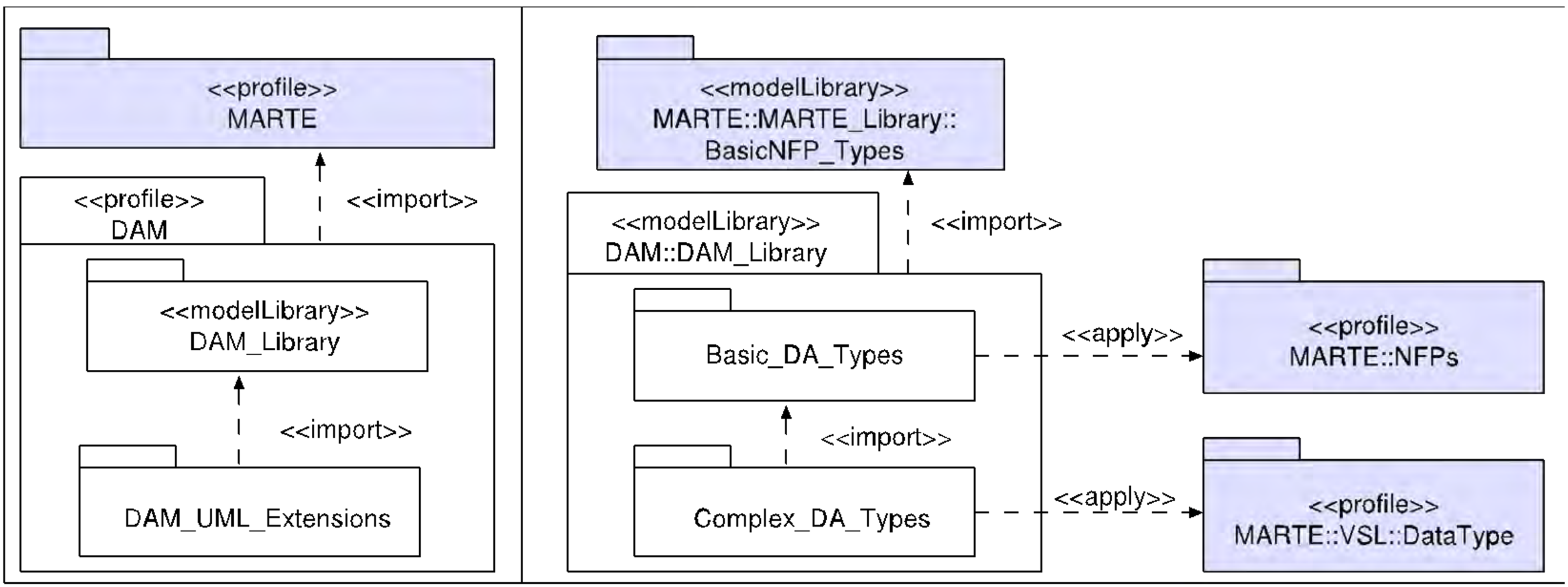


DAM domain model: Maintenance



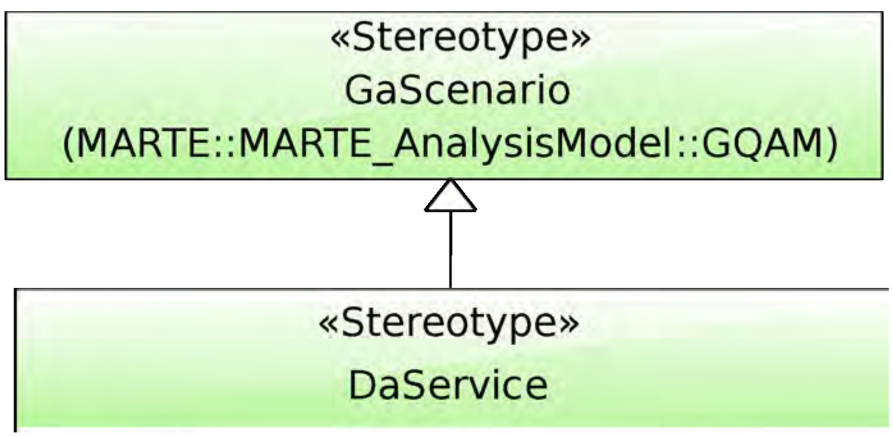
Mapping the domain concepts into a UML profile

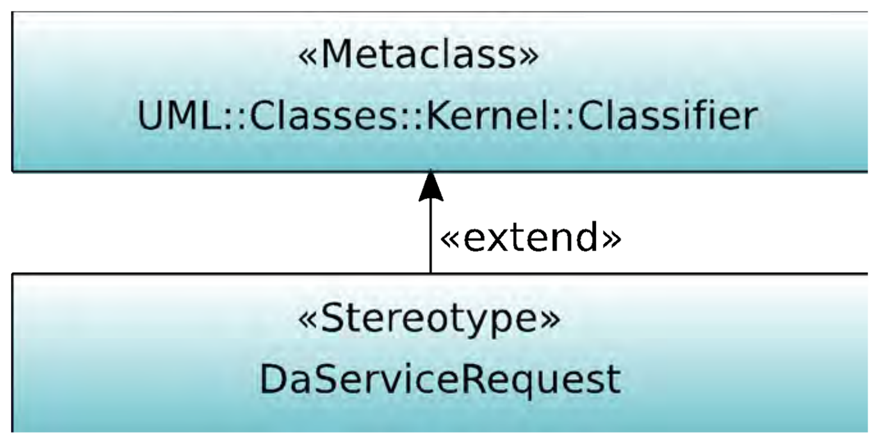
- General guideline to extend UML meta-model
- Best practice of UML MARTE for traceability
- Reuse of UML MARTE (import/application of sub-profiles)



Mapping process: classes

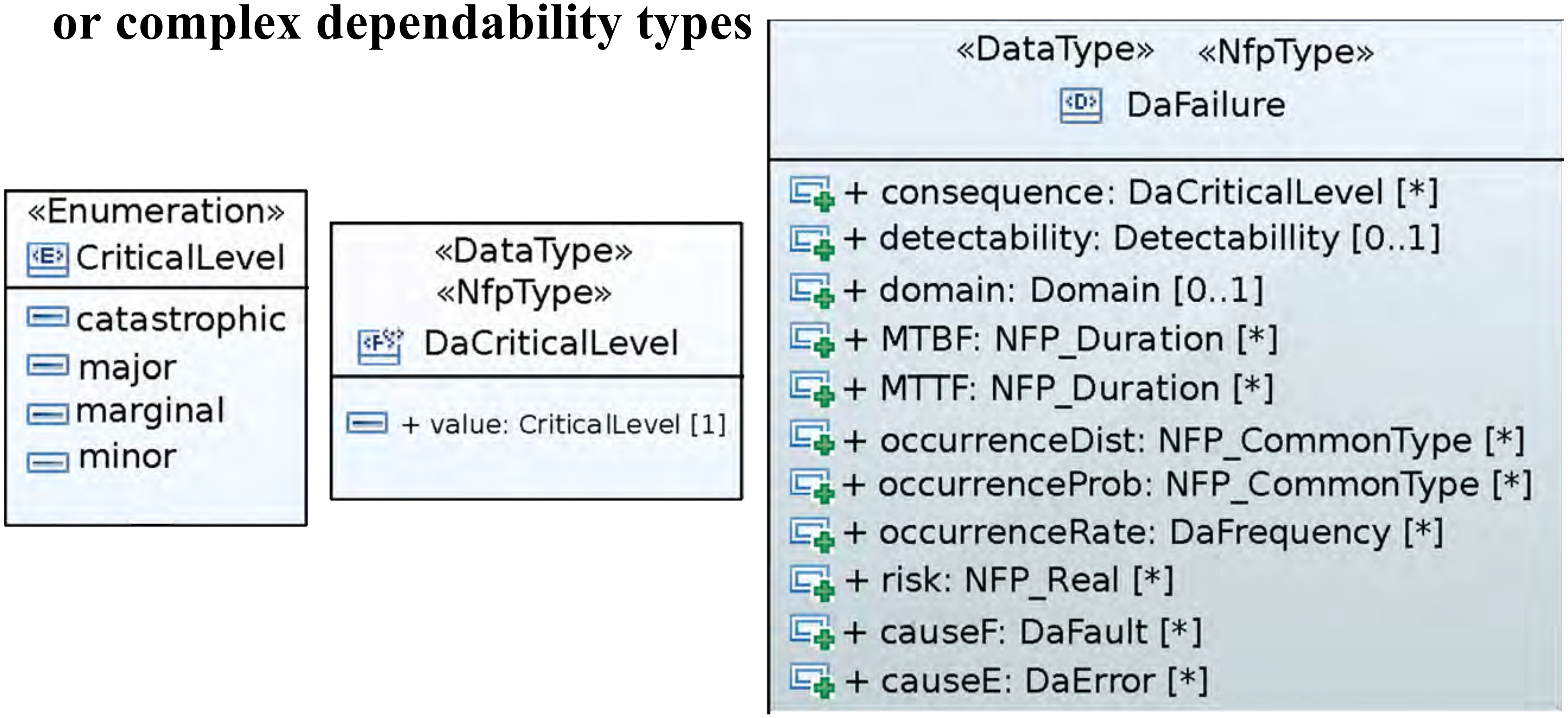
- Eventually only a subset of classes have been mapped to stereotypes
  - Abstract classes: not considered
  - Threat/Maintenance classes: complex dependability types (DAM library)
  - Specialization of MARTE stereotypes vs/ Extension





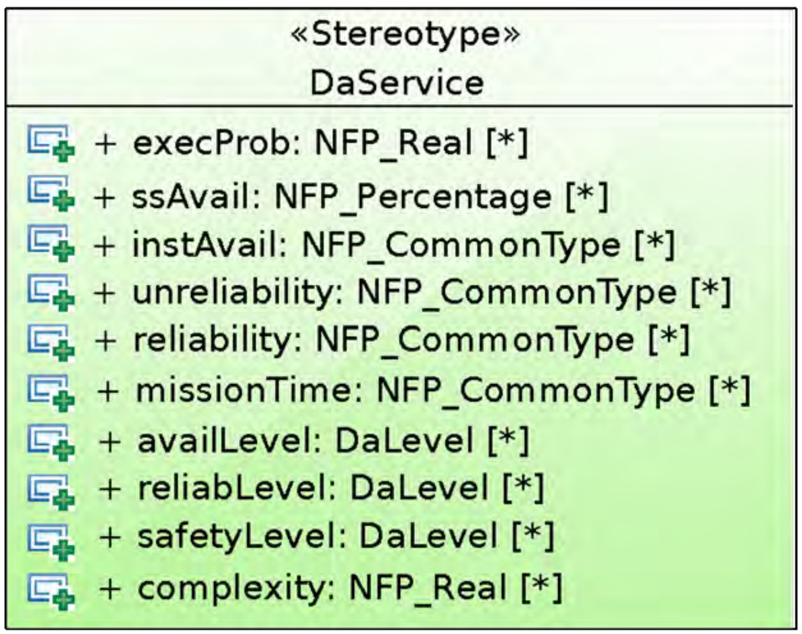
Mapping process: dependability types

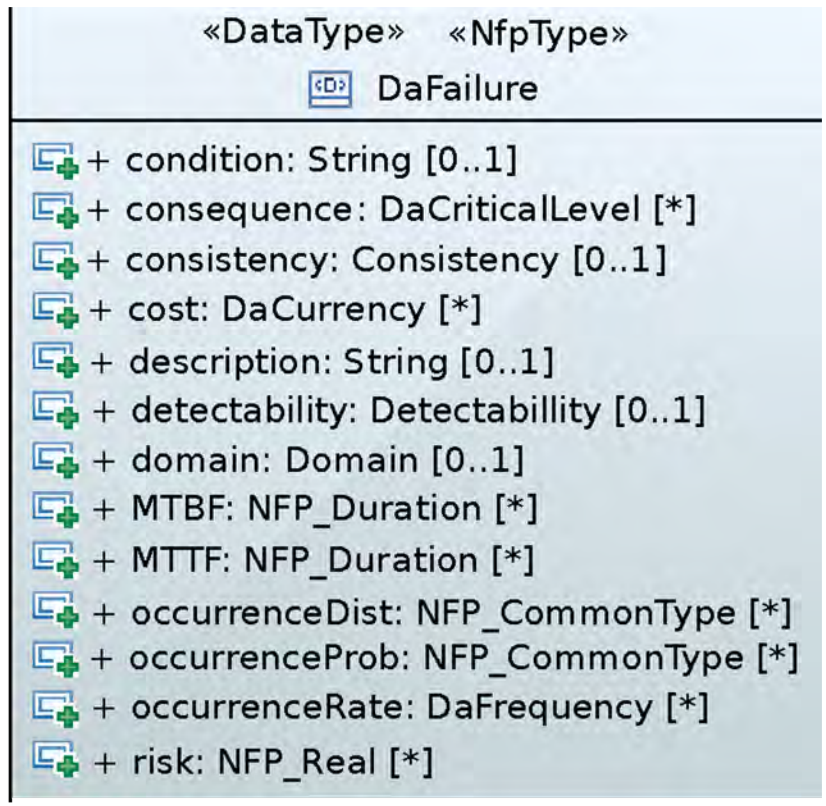
- Basic dependability types
  - Enumeration
  - Datatypes that inherit from MARTE basic NFP types
- Complex dependability types
  - Datatypes with attributes of basic NFP types, basic dependability types or complex dependability types



Mapping process: attributes

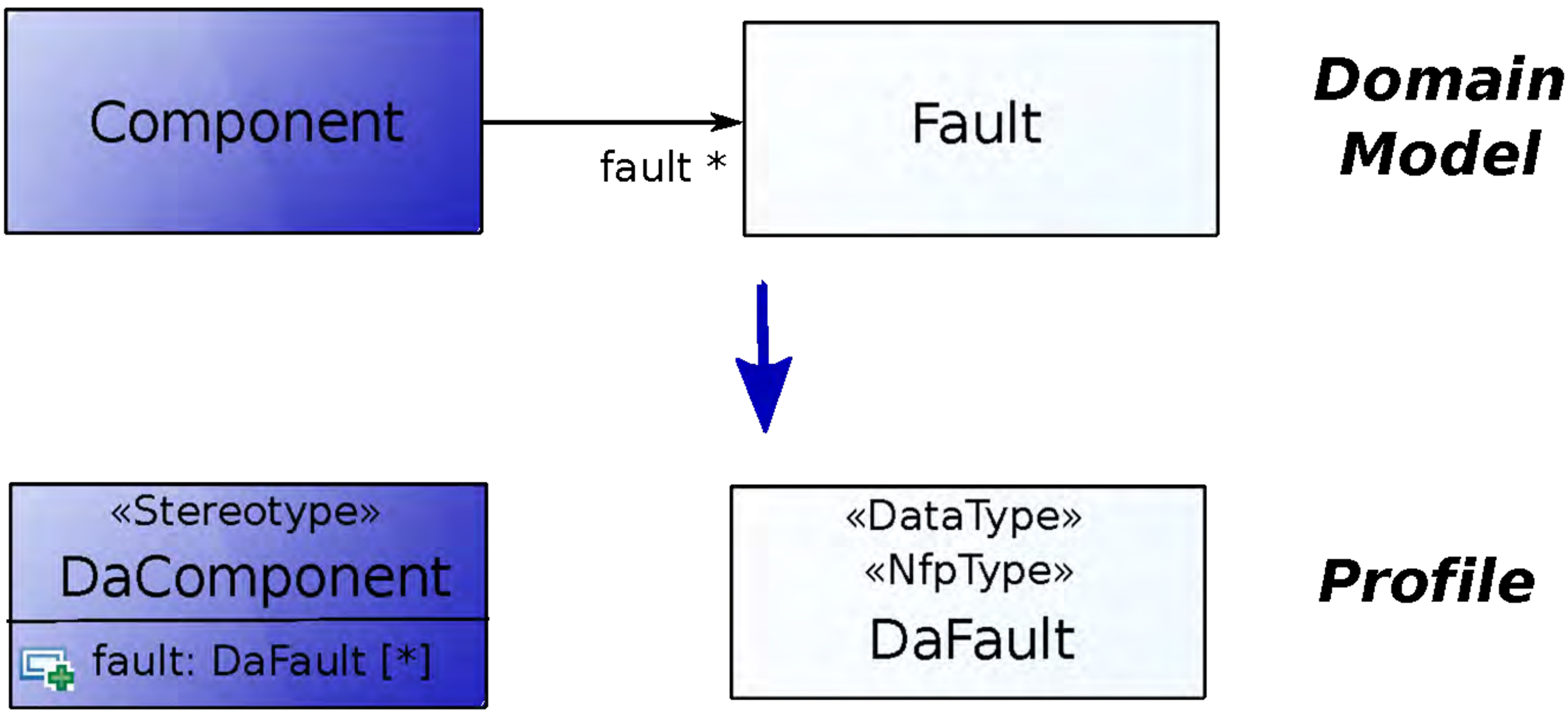
- Domain attributes have been mapped to attributes of
  - stereotypes
  - complex dependability types
- For each attribute
  - a type is associated
  - a multiplicity is defined





Mapping process: associations

- Application of the "reference association" pattern

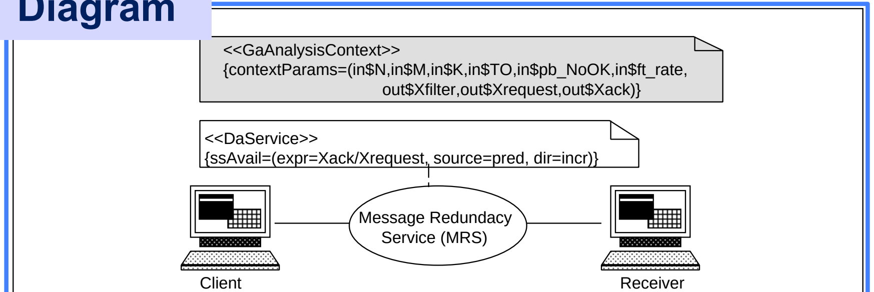


Application to a case study

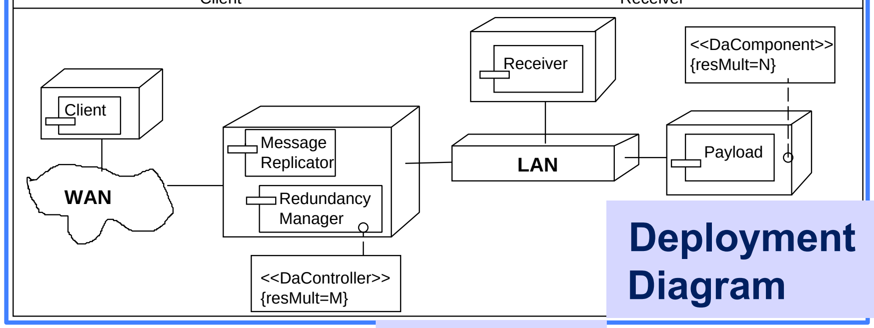
- Message redundancy service (MRS)
  - Aimed at delivering of only uncorrupted messages to the target system
- Modelling of fault (intrusion) tolerance mechanisms
  - Hw/sw redundancy
  - Voting & time-out
- Analysis of different NFPs
  - Availability
  - Response time
  - Robustness (application specific metric)
    - Rate of filtered messages
- Sensitivity analysis
  - Availability and response time metrics vs/ fault occurrence rate and time-out duration
  - Rate of filtered messages vs/ fault occurrence rate and probability of incorrect messages

UML models

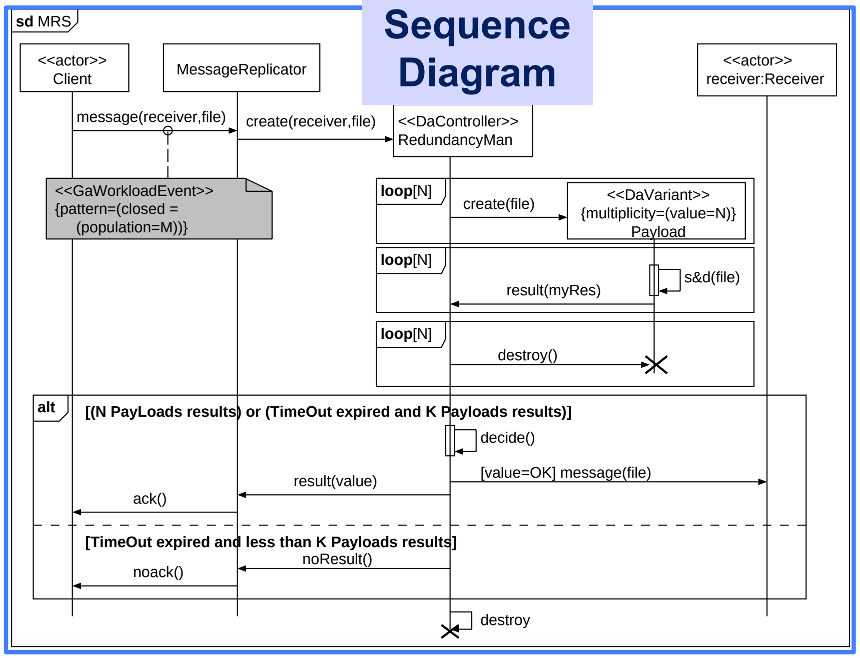
Use case Diagram



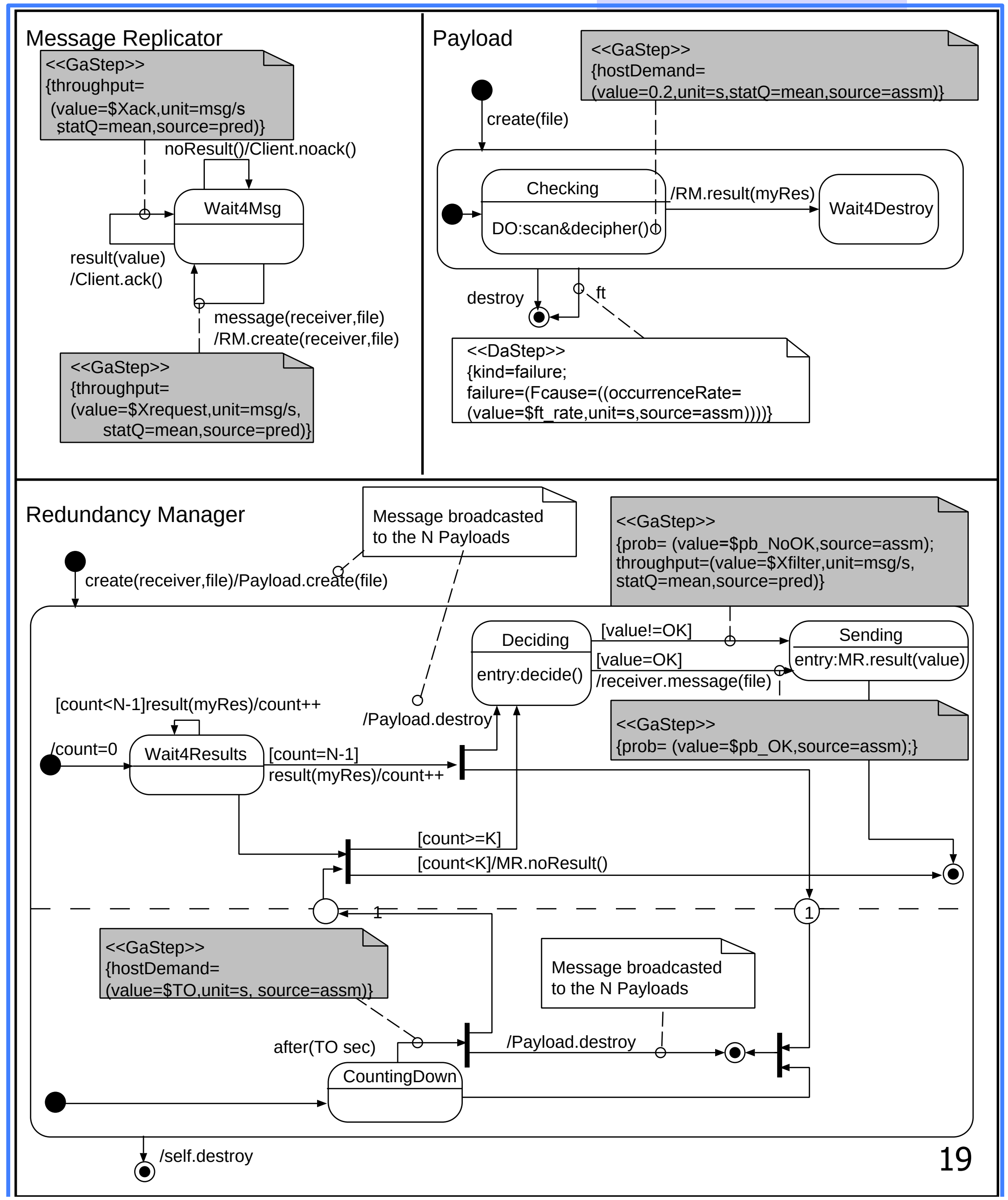
Deployment Diagram



Sequence Diagram



State Machines



DAM annotation

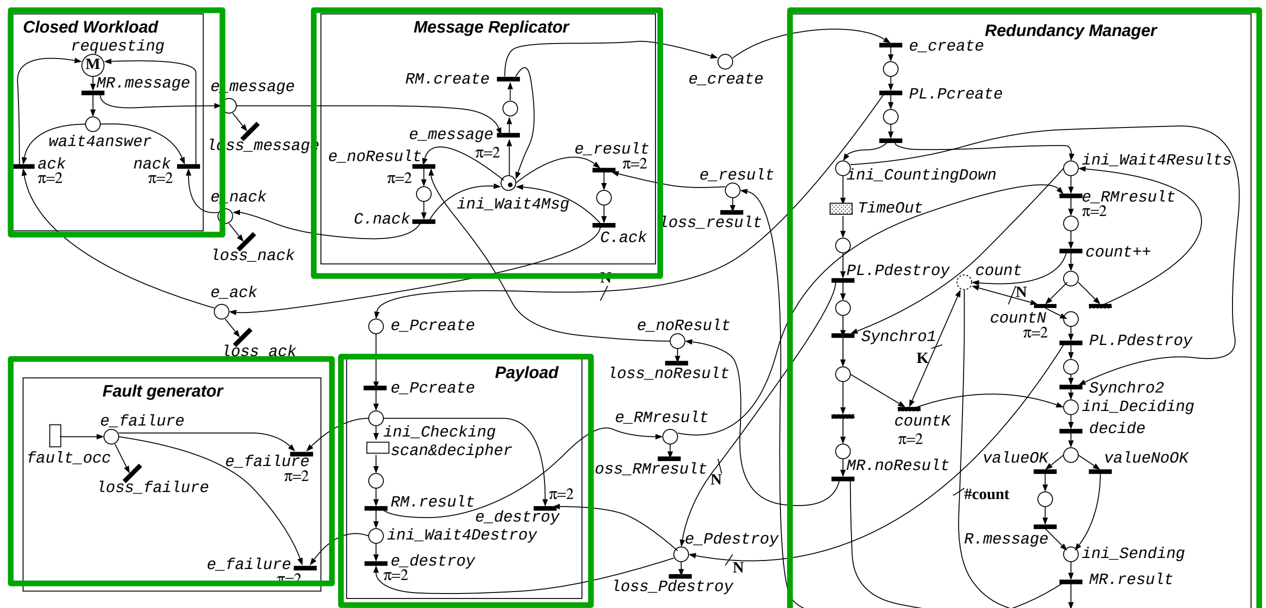
Diagram	Stereotype	Annotations
Use Case Diagram		
	DaService	availability metric
Deployment Diagram		
	DaController,DaVariant, DaComponent	redundancy structure (hw redundancy level, fault occurrence rate)
Sequence Diagram		
	DaController, DaVariant	use of redundant sw component (redundancy level)
State Machine		
	DaStep	failure occurrence (caused by fault annotated in DD)

MARTE annotation

Diagram	Stereotype	Annotation
Use Case Diagram		
	GaAnalysisContext	parameters declaration
Sequence Diagram		
	GaWorkloadEvent	closed arrival pattern
	GaService	response time metric
State Machine		
	GaStep	throughput metric, probability and duration

Analysis with MARTE-DAM

- Deterministic and Stochastic Petri Net (DSPN)
- Customization of a model transformation approach based on model composition*



Legend (parameter specification)

Place	Initial marking
requesting	M

Transition	Type	Delay/Rate/Prob
scan&decipher	EXP	0.2 (sec.)
fault_occ	EXP	ft_rate
TimeOut	DET	TO (sec.)
valueOK	IMM	pb_OK
valueNoOK	IMM	pb_NoOK

Arc	Multipl.
PL.Pcreate -> e_Pcreate	N
PL.Pdestroy -> e_Pdestroy	N
count <-> countN	N
count <-> countK	K
count -> count0	#count

* Merseguer, Bernardi, Campos, Donatelli, “A compositional semantics for UML State Machines aimed at performance evaluation”. WODES02

Part II
Our paper's influence

Citations

Our paper's citations

- *Google Scholar*: **195 citations**
- *Microsoft Academic*: **202 citations**
- *Scopus*: **99 citations**
- *Semantic Scholar*: **151 citations**
  - **Highly influential citations: 8**
  - **Background citations: 41**
  - **Method citations: 58**
- *Springer Link* **metrics:**
  - *Web of Science*: **67 citations**
  - *CrossRef*: **96 citations**
  - **Accesses to the paper in Springer Link: 483**

Google Scholar citations: how cited are they in turn?

Citing work	Cited by
(Selic & Gerard, 2013)	118
(Bernardi et al., 2012)	111
(Biggs et al., 2016)	72
(Bernardi et al., 2010)	61
(Bernardi et al., 2013)	56
(Yang et al., 2012)	51
(de la Vara et al., 2013)	48
(Bernardi et al., 2013)	47
(Bernardi et al., 2011)	43
(Saadatmand et al., 2011)	42
(Brosch et al., 2011)	41

Citing work	Cited by
(Yang et al., 2010)	34
(Giese & Schäfer, 2010)	31
(Marrone et al., 2015)	31
(Leitner & Leue, 2011)	31
(Pham et al., 2011)	30
(Montecchi et al., 2011)	27
(Rodriguez et al., 2010)	26
(Merseguer & Bernardi, 2013)	26
(Marrone et al., 2014)	26
(Aziz & Rashid, 2016)	25
... 174 more	under 25

Influence on our own work: Supervised theses

Student	Thesis Title	Contributions
Zhao Zhao, M.A.Sc., 2014	Automatic derivation of Fault Trees from UML models	Develops an ATL transformation from UML (Use case, Composite Structure Dgr and Sequence Dgr) extended with MARTE+ DAM) into Fault Tree Models.
Naif Alzahrani, Ph.D., 2015	Automatic Derivation of Dependability and Fault Tolerance Analysis Models from Software Architecture	Develops a QVT-O transformation chain from UML software models to Stochastic Reward Net (SRN) reliability models. Proposes also an aspect-based Fault Tolerance analysis approach.
Paul Devi Deji, M.A.Sc., 2016	Derivation of Failure Mode and Effects Analysis (FMEA) from UML Software Model	Develops a transformation in Epsilon that takes as input an UML model annotated with failure mode info and produces an FMEA model.
Bashar AlShboul, Ph.D., 2019	Safety Analysis of SysML Models in the Context of Model-Driven Engineering	Develops a pattern-based transformation in Epsilon which transforms SysML+ MARTE+DAM into Fault Tree models and feeds back the results.

Influence on our own work: Publications

Authors	Title	Contributions
	Book: 1	
S. Bernardi, J. Merseguer, D.C. Petriu	Model-driven dependability assessment of software systems, *Springer Verlag*, 2013	Our book detailing DAM and two model-driven approaches with two case studies
	Journal papers: 10	
J. Merseguer, S. Bernardi	Dependability analysis of DES based on MARTE & UML State Machines, *Discrete Event Dynamic Systems*, 2012	Use of DAM and new model-driven approach to derive Det. Stochastic Petri Nets (formalizing the translation)
S. Bernardi, J. Merseguer, D.C. Petriu	Dependability modeling and assessm. in UML-based software development, *The Scientific World Journal*, 2012	Use of DAM and proposal of a model-driven approach to derive Deterministic Stochastic Petri Nets.
S. Bernardi, J. Merseguer, D.C. Petriu	Dependability modeling and analysis of software systems specified with UML, *ACM Computing Surveys*, 2013	DAM profile compared with other dependability modeling approaches in the context of the survey.

Influence on our own work: Journal publications

Authors	Title	Contributions
S. Bernardi, F. Flammini, et al.	Enabling the usage of UML in the verification of railway systems, *Reliability Eng & System Safety,* 2013	Extend DAM for maintenance and service degradation. Customize DAM to the railway domain. Two transformations: a) from MARTE+ DAM to Repairable Fault Trees; b) from DAM-Rail to Bayesian Nets.
R. Rodriguez, J. Merseguer, et al.	Modelling Security of Critical Infra-structures: A Survivability Assess-ment, *The Computer Journal*, 2015	SecAM profile to support security properties. Analysis performed in GSPN.
S. Bernardi, L. Dranca, J. Merseguer	A model-driven approach to survivability requirements assessment for critical systems", *Journal of Risk and Reliability*, 2016	Extending DAM to support survivability requirement assessment. Case study of a command and control system.
B. AlShboul, D.C. Petriu	Automatic Derivation of Fault Tree Models from SysML Models for Safety Analysis, *JSEA*, 2018.	Model transformation from SysML+ MARTE+DAM to Fault Tree models. Implemented in Epsilon over Eclipse

Influence on our own work: Journal publications

Authors	Title	Contributions
J.I. Requeno, J. Merseguer, et al.	Quantitative Analysis of Apache Storm Applications: NewsAsset Case Study, *Inf Syst Frontiers,* 2019	Defines a DSML for performance and dependability analysis of Apache Storm as a UML profile.
S. Bernardi, S. Marrone, et al.	Towards a Model-Driven Eng. approach for the assessment of NFPs using multiformalism, *SoSyM,* 2019.	Use of MARTE+PAM+DAM to derive performance, reliability and performability models
S. Bernardi, U. Gentile, et al.	Security modelling and formal verif. of survivability properties: Appl. to cyber-physical systems , *JSS*, 2021	Survivability extensions are modelled directly as extensions of Misuse cases (supported by tool Surreal).
	Conference papers: 19 (not listed)	

Tools supporting DAM

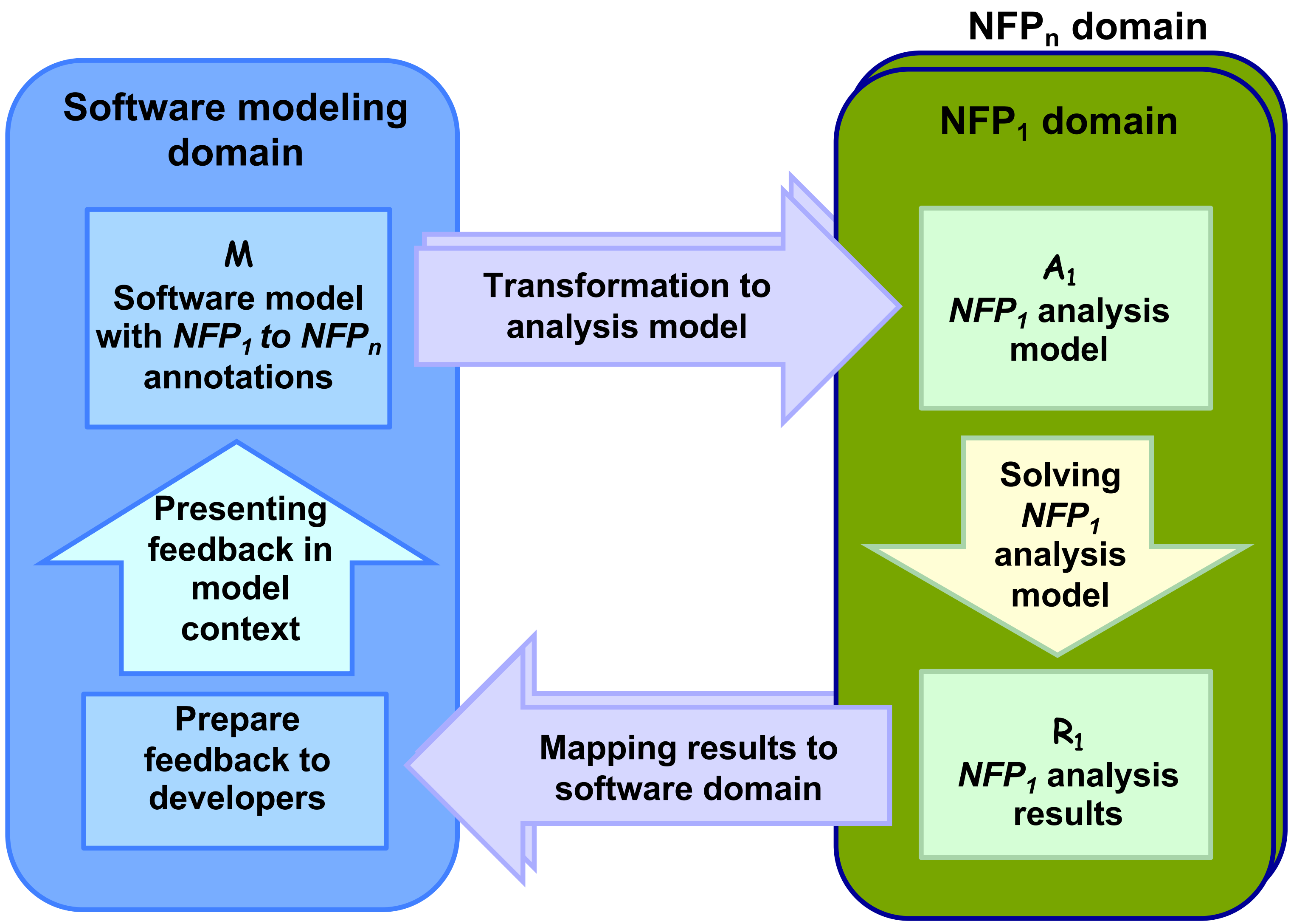
- **MASDES: https://bitbucket.org/masdesgroup/masdes/wiki/Home**
  - **First version of DAM implemented as Eclipse plugin**
- **DICE Simulation https://github.com/dice-project/DICE-Simulation**
  - **Developed within the EU project "Developing Data-Intensive Cloud Applications with Iterative Quality Enhancements" (DICE).**
  - **Tool Functionalities:**
    - **Annotation of UML diagrams with system quantitative properties for performance and reliability, based on UML.MARTE and DAM profiles.**
    - **Determines *performance metrics* (response time, throughput and resource utilization) and *reliability metrics* (MTTF, availability, reliability, prob. of failure) on a UML scenario that represents a system execution.**
    - **Specializes DAM for platform-dependent data-intensive applications: Apache Hadoop, Spark, Storm and Tez.**
    - **Performs *what-if* or *sensitivity* analysis for performance and reliability metrics. The user can see multiple output results (e.g., as plot charts).**
    - **Transforms a UML scenario into stochastic Petri nets.**

Use of DAM in influenced work

- Use/extend DAM to annotate different dependability NFPs:
  - reliability
  - availability
  - safety
  - fault tolerance
  - maintainability
  - survivability
  - security
  - privacy (incipient work)
- Different software and dependability models in influenced work:
  - Software/system models:
    - UML (class, composite structure, use case, deployment, state machine, seq., activity)
    - SysML (block definition diagram BDD, internal BD, state machine)
  - Dependability analysis models:
    - Different kind of Stochastic Petri Nets: DSPN, GSPN.
    - Stochastic Reward Networks (SRN)
    - Fault trees, Component Fault trees, Stochastic Fault trees
    - FMEA models

Future Challenges

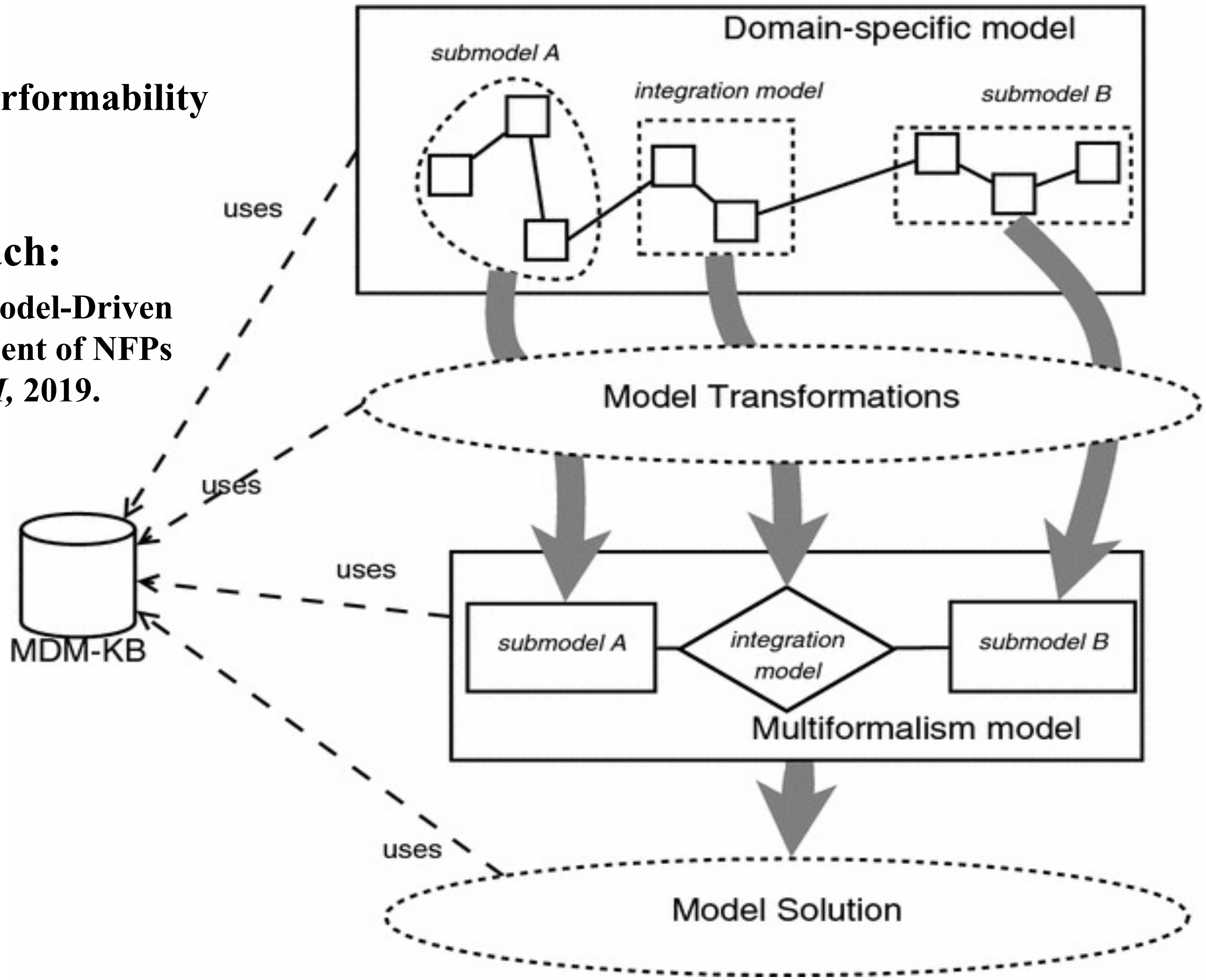
Integrating the analysis of multiple NFPs in the MDE process



Analysis models can be integrated

Example:

A: performance
B: reliability
A+B: performability



Work using this approach:

Bernardi et al., "Towards a Model-Driven Eng. approach for the assessment of NFPs using multiformalism", *SoSyM,* 2019.

Analysis models cannot be integrated

- **Using different formalisms for the NFP analysis models**
- **Example of NFPs to be analyzed: (performance, reliability, safety, cost)**

  QN DSPN FaultTree $

  - **The NFP models cannot be integrated in a single analysis model**
- **Modeling artifacts:**
  - M : Software system model **with** NFP_1 **to** NFP_n **annotations**
  - $D_i \in M$: subset of diagrams **in** M **with annotations for** NFP_i
  - A_i : Analysis model **for** NFP_i **(obtained from the subset** D_i**)**
  - T_i : Transformation **from the source model** D_i **to the target model** A_i
  - S_i : Solver **for** A_i
  - $R_i(X_{ij})$: Results **produced by** $S_i(X_{ij})$ **for the set of input parameters** X_{ij}**, which corresponds to the** j^{th} **data point for which model** A_i **is solved.**
    - **Challenge: select the data points for which to solve each model**
    - **Visualize the results in a user-friendly way (e.g., as charts).**

Design space exploration

Research challenge: how to use multiple NFP analysis models to find good design solutions (preferably optimal)?

1. The problem of balancing multiple NFPs lends itself to multi-criteria optimization
   - Complexity and huge design state space -> intractable solution
   - Traditional optimization methods used mostly when a single NFP analysis model is required
   - If multiple NFP are required, use metaheuristics search techniques
     - genetic algorithms, simulated annealing
2. Another approach for balancing different NFPs is using decision support systems for reasoning under uncertainty
   - decision support systems based on Bayesian Belief Network models
3. Diagnostic and design-change rule-based techniques are used to find good design solutions when a single NFP is considered at a time.

Standardization

- DAM is an extension of the MARTE profile standardized by OMG
  - Purpose: to add dependability annotations to UML and/or SysML software/system models
  - Compliance with OMG standards allows for using standard UML tools.
- MARTE versions:
  - Version 1.0: November 2009
  - Version 1.1: June 2011
  - Version 1.2: April 2019
  - Major revision in progress:
    - MARTE 2.0 RFI: issued September 2008
    - RFI Response: August 2019
- Future work: to align DAM with the new version MARTE 2 when it will be made public.

Conclusions (1)

Experience in conducting model-driven NFP analysis in the context of MDE shows that there still are several challenges:

- ***Human qualifications***
  - **Software developers are not trained in all formalisms used for NFP analysis -> this leads to idea of hiding analysis details from developers.**
  - **However, the software models must be annotated with extra info for each NFP, and the analysis results must be interpreted in order to improve the designs.**
  - **Challenge: find a better balance between what to hide and to expose.**
- ***Consistent model evolution***
  - **The analysis of different NFPs may require source models at different levels of abstraction/detail.**
  - **Changing diagrams $D_i \in M$ for NFP_i may affect other NFP models.**
  - **Challenge: to keep all the models/views consistent while making design changes to improve the NFPs.**

Conclusions (2)

- *Tool interoperability*
  - **Difficult to interface and to integrate seamlessly different MDE tools and NFP analysis tools, which were created at different times with different purposes and maybe running on different platforms.**
- *Software process*
  - **Integrating the analysis of different NFP raises process issues.**
  - **For each NFP should explore the state space for different design alternatives, configurations, workload parameters**
  - **How to compare different solution alternatives that may improve some NFPs and deteriorate others, and how to decide on trade-offs.**
- *Propagation of changes through the model chain*
  - **Currently, every time the software design changes, a new analysis model is derived from scratch to redo the analysis.**
  - **Challenge: to develop incremental transformation methods and to keep different models synchronized.**