Model based Quantitative Resilience Assessment of Critical Information Infrastructures

Andrea Bondavalli

bondavalli@unifi.it

http://rcl.dsi.unifi.it

Università degli Studi di Firenze

Outline

- Short Intro
  - Basics
  - The research framework and the challenges
- Some contributions
  - A Hierarchical, Modular, Extensible modeling approach for the QoS analysis in dynamic, ubiquitous UMTS network scenarios in the automotive domain
  - A Decomposition-Based Modeling Framework for Complex Systems
  - A MDE Transformation Workflow for Dependability Analysis
- Directions for the future

Quantitative Analysis often has validation purposes, but what is Validation?

- Definition of **Valid** from Webster's Third New International Dictionary – "Able to effect or accomplish what is designed or intended"
- Two basic notions:
  1. Specification - A description of what a system is supposed to do.
  2. Realization - A description of what a system is and does.
- Definition (here):

Validation - the process of determining whether a realization meets its specification.

What is a system?

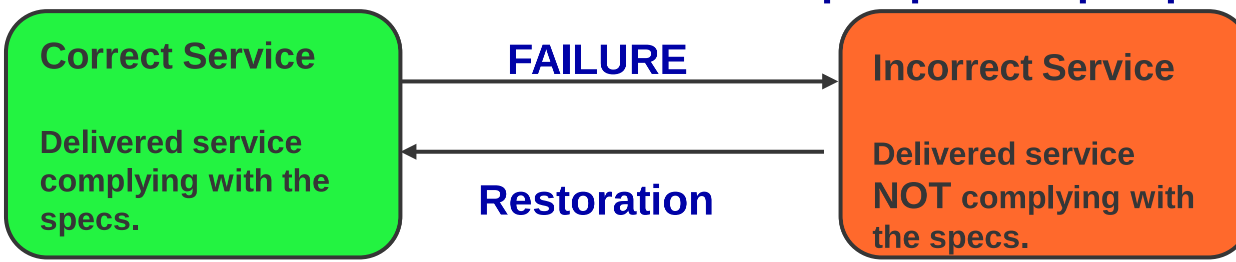
- Many things, here, a collection of
  - hardware
  - networks
  - operating systems, and
  - application software

  that is intended to be dependable, secure, survivable or have predictable performance.

- Before learning how to validate we must review basic performance and dependability concepts and measures

Dependability

- Dependability is the ability of a system to deliver a specified service.
- System service is classified as proper if it is delivered as specified; otherwise it is improper.
- System failure is a transition from proper to improper service.
- System restoration is a transition from improper to proper service.



The "properness" of service depends on the user's viewpoint!

Reference: J.C. Laprie "Dependability - its attributes, impairments and means," in "Predictably Dependable Computing Systems", B. Randell, J. C. Laprie, H. Kopetz and B. Littlewood Ed., Springer-Verlag, 1995, pp. 3-24.

Examples of Specifications of Proper Service

- *k* out of *N* components are functioning.
- every working processor can communicate with every other working processor.
- every message is delivered within *t* milliseconds from the time it is sent.
- all messages are delivered in the same order to all working processors.
- the system does not reach an unsafe state.
- 90% of all remote procedure calls return within *x* seconds with a correct result.
- 99.999% of all telephone calls are correctly routed.

➡ Notion of proper service provides a specification by which to evaluate a system's dependability.

Dependability Concepts

Measures - properties expected from a dependable system

- Availability
- Reliability
- Safety
- Confidentiality
- Integrity
- Maintainability
- Coverage

Impairments - causes of undependable operation

- Faults
- Errors
- Failures

Means - methods to achieve dependability

- Fault Avoidance
- Fault Tolerance
- Fault Removal
- Dependability Assessment

Critical Infrastructures

- Critical computing systems have evolved becoming more and more complex and their interconnection has resulted in Critical Information Infrastructures widely used in our society
- They are now pervading most of our life – sometimes in a way we are not even aware of.
- Their malfunctions, breaking or a disruption of their services is very costly and in many cases not acceptable.
- They need to be protected against accidental faults, environmental disasters and deliberate attacks.

Examples of Critical Infrastructures

CI interdependencies

Energy

Transport

A few specific aspects

- Much **bigger** and much **more complex** that any system we have been dealing with-
- In addition to that Critical Information Infrastructures are also INTERDEPENDENT
  - Not designed anew as space missions or many automotive embedded systems.
  - Not only Off the Shelf but a lot of Legacy components hw and sw. Sometimes even the source code does not exist anymore
  - Maintenance is extremely complex and costly .... and critical

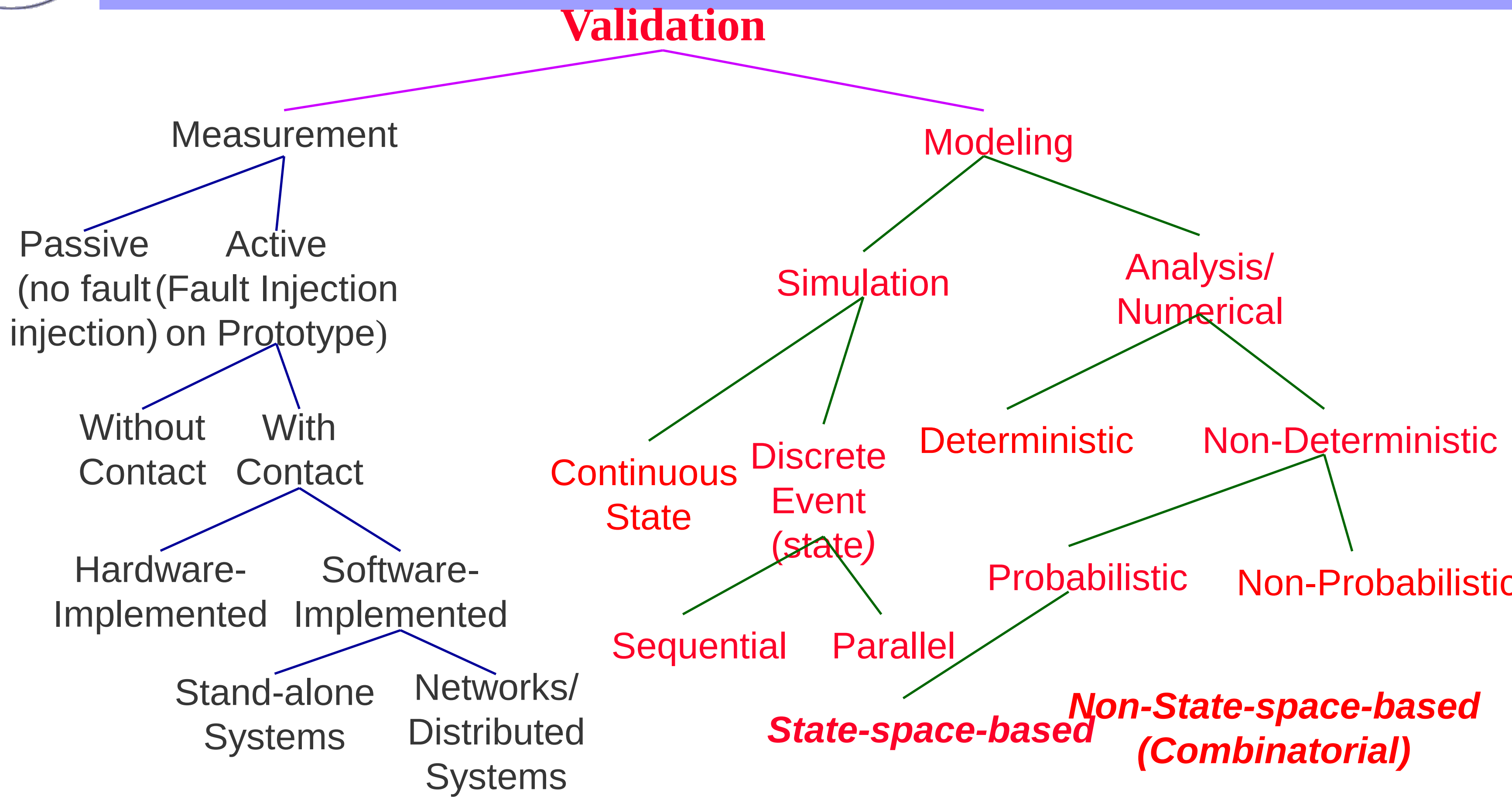
Assessment and Evaluation

- Properties such as:
  - Safety,
  - Security,
  - Availability,
  - and in general Quality of service (QoS),

Have to be guaranteed (supported as far as possible) and quantitatively assessed to understand if risks are acceptable.

Not only **BEFORE** but also **WHILE** using such systems → links with monitoring and dynamic reaction

Quantitative Validation



Choosing Validation Techniques

- There are several choices (each with advantages and disadvantages)
  - Combinatorial modeling
  - Analytic/numerical modeling
  - Simulation (including fault injection on a simulated system)
  - Measurement (including performance benchmarking and fault injection on a prototype)
- Choice of a validation method depends on
  - Stage of design (is it a proposed or existing system?)
  - Time (how long until results are required)
  - Tools available
  - Accuracy
  - Ability to compare alternatives
  - Cost
  - Scalability

Choosing Validation Techniques cont.

Criterion	Combinatorial	State-based space	Simulation	Measurement
Stage	Any	Any	Any	Post-prototype
Time	Small	Medium	Medium	Varies
Tools	Formulae, spreadsheets	Languages & Tools	Languages & Tools	Instrumentation
Accuracy	Low	Moderate	Moderate	High
Comparison	Easy	Moderate	Moderate	Difficult
Cost	Low	Low/Medium	Medium	High
Scalability	High	Low/Medium	Medium	Low

Quantitative evaluation - Options

Experimental (Measurement-based) approach

- The required measures are estimated from data measured from a real system or from a prototype using statistical inference techniques.
- The system or prototype can be exercised in specific conditions including erroneous ones (fault/attack)) injection

expensive, it requires to exercise a real system, take the measurements and analyze the data.

- typically applied to components or subsystems
- Very impractical for end to end evaluation of large systems
- Would require more rigor in taking measurements
- Impossible to inject faults in existing running infrastructures….

Quantitative evaluation through models

(more) Theoretical (Model-based) approach:

- the required measures are obtained through the solution of a (stochastic) model, that is an abstraction of the system.
- The solution can be analytical or by simulation

- Working on a model allows to consider any kind of faults and attacks that can be modeled.
- Analytical solution (when it exists) is relatively inexpensive and easier to perform.
- Simulation may become very long and expensive (in some cases though is the only option)

Grand Challenge: complexity

- Complexity depends on several factors
  - Dependability/Security measures
  - detail level of the models
  - stochastic dependencies and inter-dependencies
  - systems and environment characteristics such as:
    - **dynamicity and heterogeneity of the network conditions**
    - **mobility and nature of the actors (including attakers)**
    - **large number of components and scenarios**
- Consequence:
  - Very complex models ...to build and ... to solve...

How to model complex systems? -1

- **Care in model construction**
  - Modular composition of simple sub-models + composition rules
- **and solution techniques**
  - Largeness avoidance techniques
    - Creating smaller, equivalent representations; Increased levels of abstraction
  - Largeness tolerance techniques
    - Facilitating the creation of large models; Solving larger representations; Speeding up the solution time

How to model complex systems? -2

- (Automatic) Derivation of dependability models from engineering ones (in Model Driven Engineering Frameworks)

- Hybrid approaches
  - **Combination of different modelling formalisms and evaluation methods** (including experimental ones), exploiting their complementarities and synergies.
  - appears **the** viable option for running information infrastructures

We explore now some directions…

- A Hiererchical, Modular, Extensible modeling approach for the QoS analysis in dynamic, ubiquitous UMTS network scenarios in the automotive domain
  - Key elements: Modular Composition + Hybrid Approach
- A MDE Transformation Workflow for Dependability Analysis
  - Key elements: UML2 profiling for dependability + automatic transformations

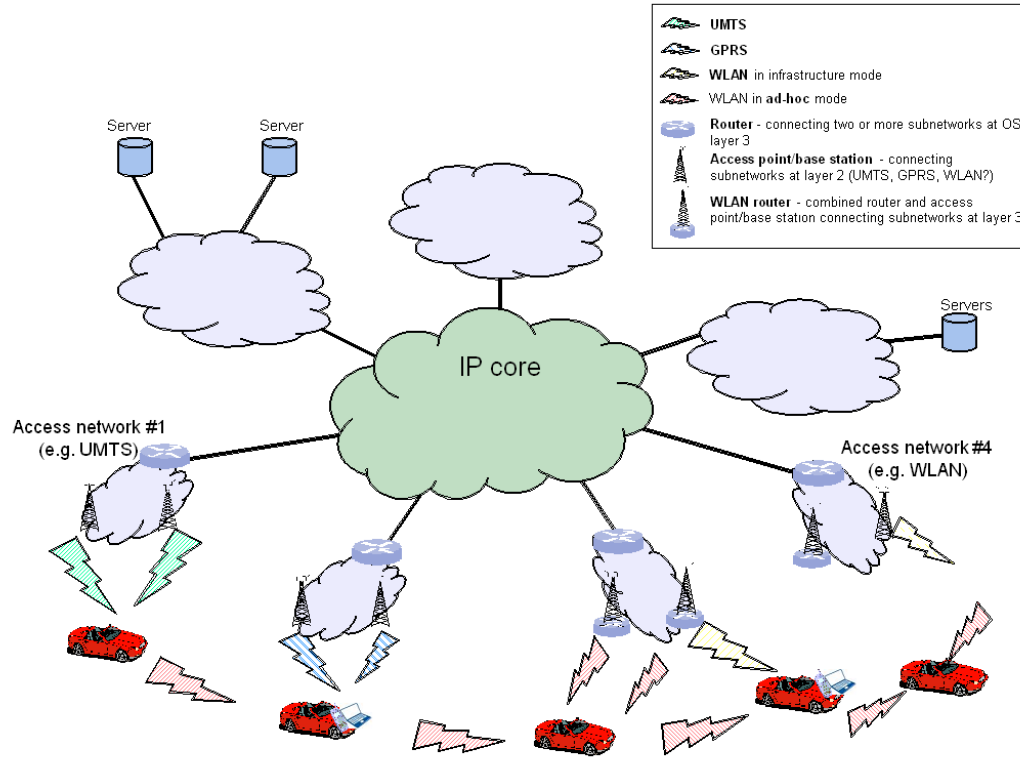
Domain specific modelling

– A Hiererchical, Modular, Extensible modeling approach for the QoS analysis in dynamic, ubiquitous UMTS network scenarios in the automotive domain

- **Key elements: Modular Composition + Hybrid Approach**

Dynamic and Ubiquitous Systems in the Automotive Domain

www.HIDENETS.aau.dk



HIDENETS
HIghly DEpendable IP-based NETworks and Services

Università degli Studi di Firenze

Motivations

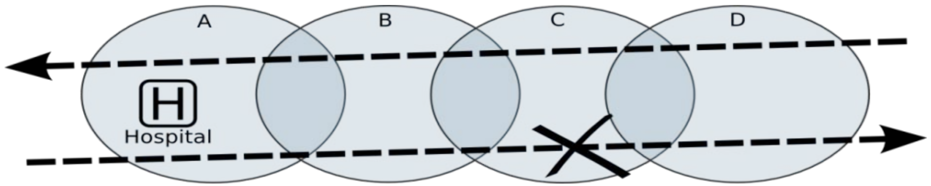
- **GOAL**: QoS analysis in dynamic, ubiquitous network scenarios, accounting for:
  - heterogeneous users, applications and QoS requirements
  - outage events affecting the availability of the network resources
  - mobility of users (possibly at highway speeds) and its effects on link quality
- **NEED** of a methodology to manage the system's complexity and facilitate the modeling process. Useful properties:
  - Modularity
  - Hierarchical composability
  - Adaptability/extensibility

Context and system description

- Context
  - Car-to-car and car-to-infrastructure communications
  - Different applications, different networks domains, different actors…
- The “Car-accident” use case to show the modeling process
  - Accident involving cars and other road users including an upcoming ambulance
  - The ambulance needs to use the network to communicate with the hospital both at the accident site and heading back to the hospital
  - Before the site gets cleared, approaching vehicles are in a traffic jam, and start using the network for calling, or for entertainment applications
- UMTS the network technology
  - Faults may occur, reducing the available radio resources of UMTS base stations

A sample scenario

- Composed by a set of overlapping UMTS cells, covering a highway
  - Four basestations with partially overlapping coverage areas (A, B, C, D)
  - Users are moving in the highway in two different lanes, with opposite directions



- Four different phases
  - Nominal behavior
  - Emergency behavior (accident occurred – ambulance approaching, traffic jam developing)
  - Ambulance at the crash site
  - Ambulance heads back to the hospital and traffic flow is restored
- 5 different network services
  - Telephony, Browsing, FileTranfer for “normal” users
  - EmergencyStreaming and EmergencyVideoConference for the ambulance, (together “access to medical expertise” application)

Measures of interest

- The measure of interest concern the QoS levels both from a users' perspective and from a mobile operator's point of view
- User oriented
  – Probability of service interruption
  – Probability to maintain the "access to medical expertise" connection until the ambulance arrives at the hospital
  – Probability that a service request is blocked or dropped
- Infrastructure oriented
  – Throughput
  – Base stations' load
  – Number of allocated channels (i.e., served users)

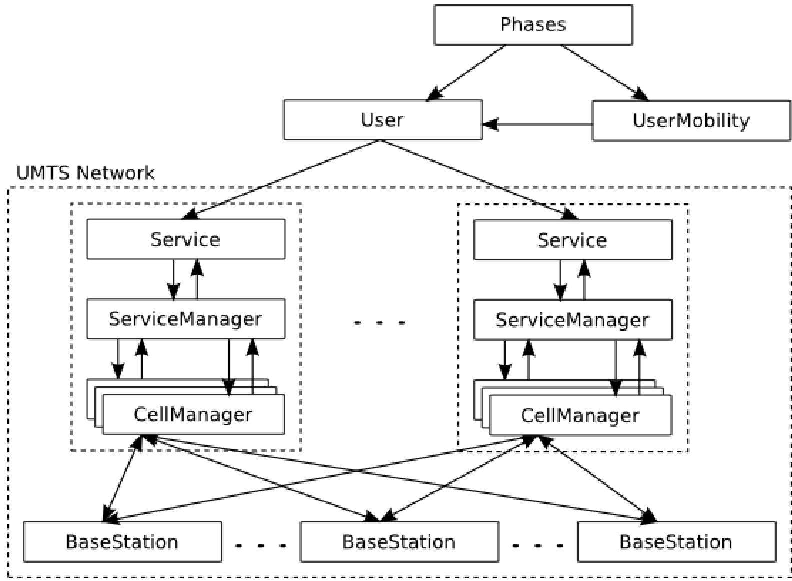
Modeling process: overview

- Identify the main UMTS features relevant for the QoS:
  - RACH procedure – procedure to initiate services, subject to collisions
  - Admission Control – decides whether a new service request can be accepted, based on the available network "capacity".
  - Soft Handover – UMTS devices can have two or more simultaneous connections with different cells (improves support to mobility)
- Identify the main "components" of the scenario
  - E.g., base stations, users… Further details in next slides
- Use of Stochastic Activity Networks (SAN)
  - An extension of the Stochastic Petri Nets formalism
  - Has useful features that can be exploited to improve usability and modularity of the model

Identify basic modeling "bricks"

- Phases
  – The different phases of the scenario
- User
  – The behavior of the user, in terms of service requests
- UserMobility
  – The user's mobility patterns
- BaseStation
  – Models a UMTS base station, including its possible failures
- CellManager
  – Handles the connection of one user to a given UMTS station, including channels allocation and deallocation
- ServiceManager
  – Provides the resources to execute a service. It also implement the soft handover mechanism
- Service
  – the interface between the user and the network

...which are used multiple times and composed to obtain the overall model for the given scenario

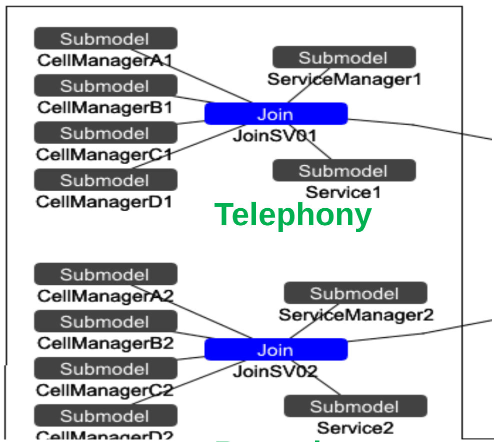


Atomic models and "parameterization"

- As in an object-oriented philosophy, basic "template" SAN atomic models have been defined, to be instantiated with specific parameters
  - SANs 'Extended' places allow for non-integer parameters (e.g., required bandwidth for the networks service, load factor of the base station)
- The overall model then is obtained by composition of some "instances" of such models.
  - Avoids duplicating the code and structure of the models, - a time consuming and error-prone process.
  - The resulting model is more flexible and can be easier adapted to a different scenario.
- The model for the scenario described before consists of 40 atomic model instances from only 10 different templates using parameterization.

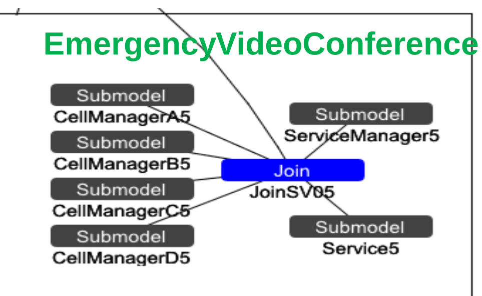
The overall model

- The model is built in a bottom-up fashion, through composition of the atomic models, using the join and replicate operators.

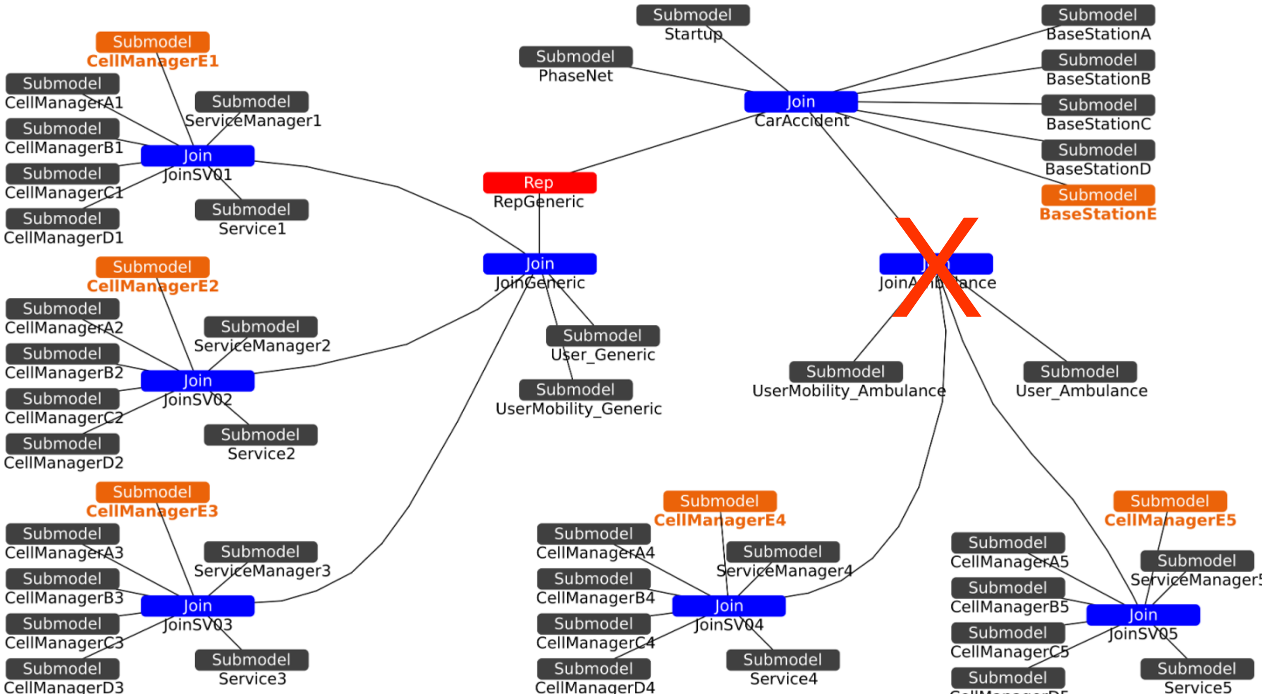


- At the bottom level we have five joins, one for each network service
- These joins are then composed with the respective user model (1-3 with User_Generic and 4-5 with User_Ambulance) and the corresponding mobility models

- The generic user is then replicated as needed, and
- added to the top level join, together with the ambulance join and the BaseStation models (top right in the figure)



Other possible scenarios



➔ **A scenario without emergency vehicles can be obtained deleting the “JoinAmbulance” composed model**

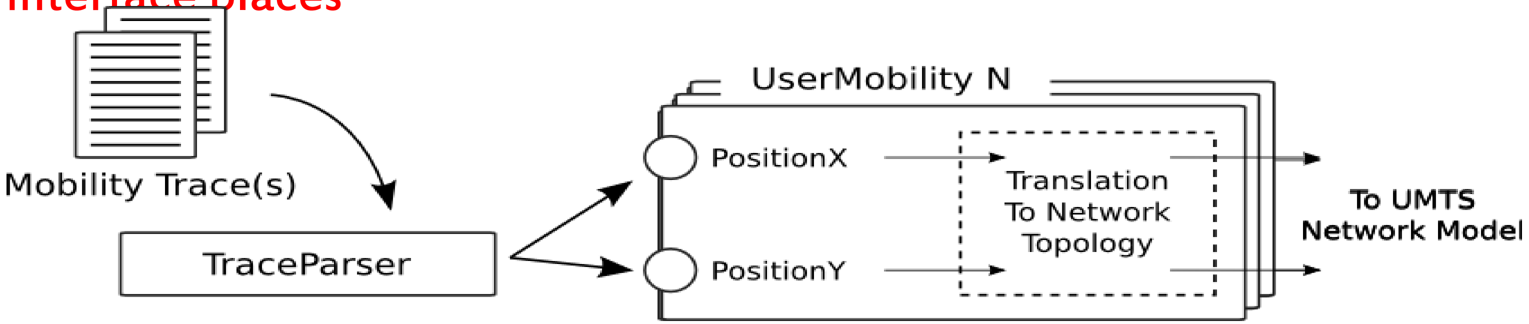
➔ **Adding another base station (thus obtaining a different network topology) would simply consist in adding another “CellManager” atomic model to each “JoinSV” composed model, and another “BaseStation” atomic model (“BaseStationE”) to the “CarAccident” composed model.**

Models refinement

- Goal: a more accurate modeling of the mobility of users
- The modularity of the approach allows to easily achieve it:
  1. define a new “UserMobility” template model
  2. Then Combine the SAN model with the traces produced as output by an ad-hoc mobility simulator (VanetMobiSim)
- This change allows to refine also the UMTS network behavior
  - Enables a more precise estimation of the load factor
  - Taking into account also for the path loss caused by the distance
- Opens other interesting perspectives:
  - Use real-world data (e.g., traces taken from GPS of real vehicles)

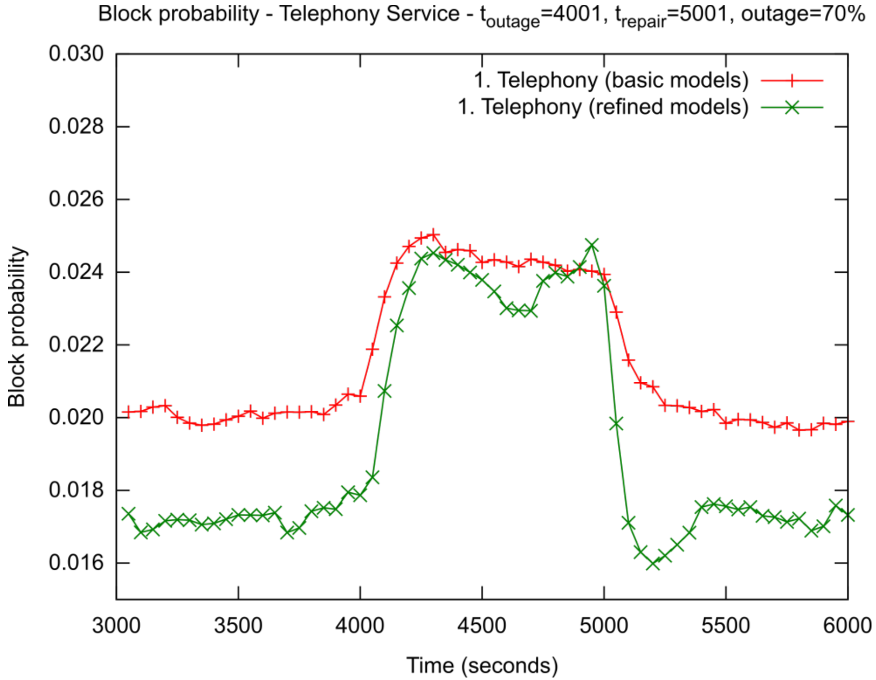
Integration with VanetMobiSim (1/2)

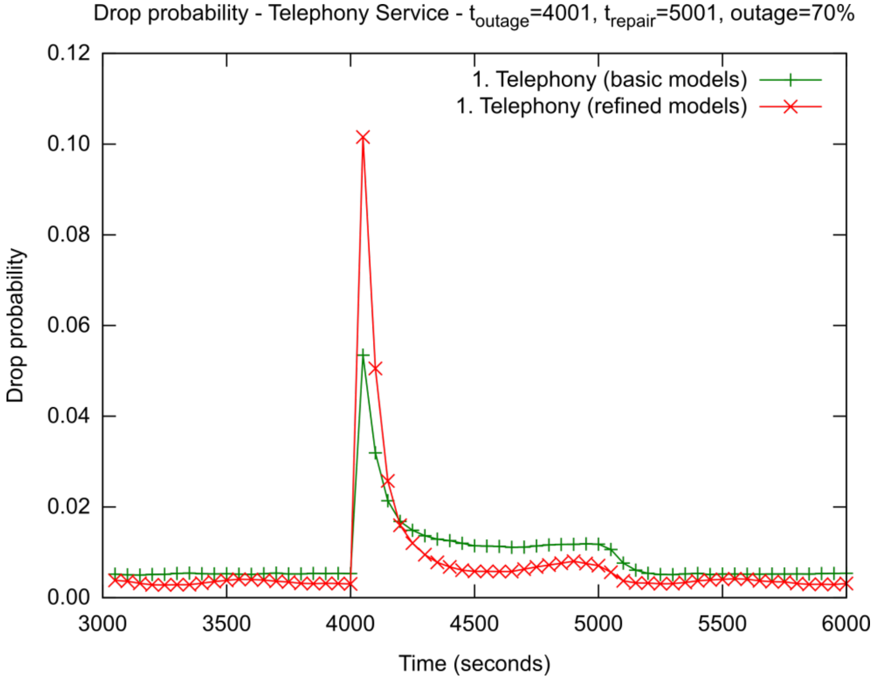
- The "UserMobility" atomic model (in general):
  1. Implements the user mobility patterns and updates the position
  2. Translates the user position to a network-related position (e.g., "user is in the coverage area of base station A")
- Few modifications required
  - The "mobility pattern" part is replaced by some interface places which hold the current user position
  - A new "TraceParser" atomic model is intoduced; it parses the trace(s) and updates the interface places



Example results

- Drop and block probability of "Telephony" service (phone calls)
  - the peak is when the accident occurs
- Green: basic model – Red: refined model combined with the simulator





General modelling approach

– A MDE Transformation Workflow for Dependability Analysis

- Key elements: UML profiling for dependability + automatic transformations

Model-Driven Engineering

- A system development methodology that relies on models as primary artifacts
- Basic concepts
  - (Meta-)Modeling
  - Model Transformation
- The system model is built using an high-level engineering language (e.g., UML, AADL, SysML…)
- Automatic model transformations are used to:
  - Provide an implementation (code generation)
  - Translate the model in an alternative representation
  - **Build analysis models**

Dependability Analysis in MDE

- MDE methodologies extended to include dependability analysis
  - Dependability models are automatically built from modeling languages like UML
- Considerable effort has been spent in trying **to integrate dependability analysis within development process models**
- Still building a comprehensive framework for automated dependability analysis is a very open and challenging goal:
  - Different domains (e.g., automotive, railways, aerospace…)
  - Different analysis methods
  - Different kind of systems
  - Different measures of interest….

Open Issues

- Most of the current approaches using model transformation for dependability analysis
  - focus on a specific analysis method,
  - are bound to a particular application domain,
  - address specific aspects of the system
- There is no common understanding of what are the non-functional properties that should be included in a high-level modeling language
- There is no completely satisfactory language yet that allows to include dependability and security related non-functional properties in an engineering model

The CHESS Framework (1/2)

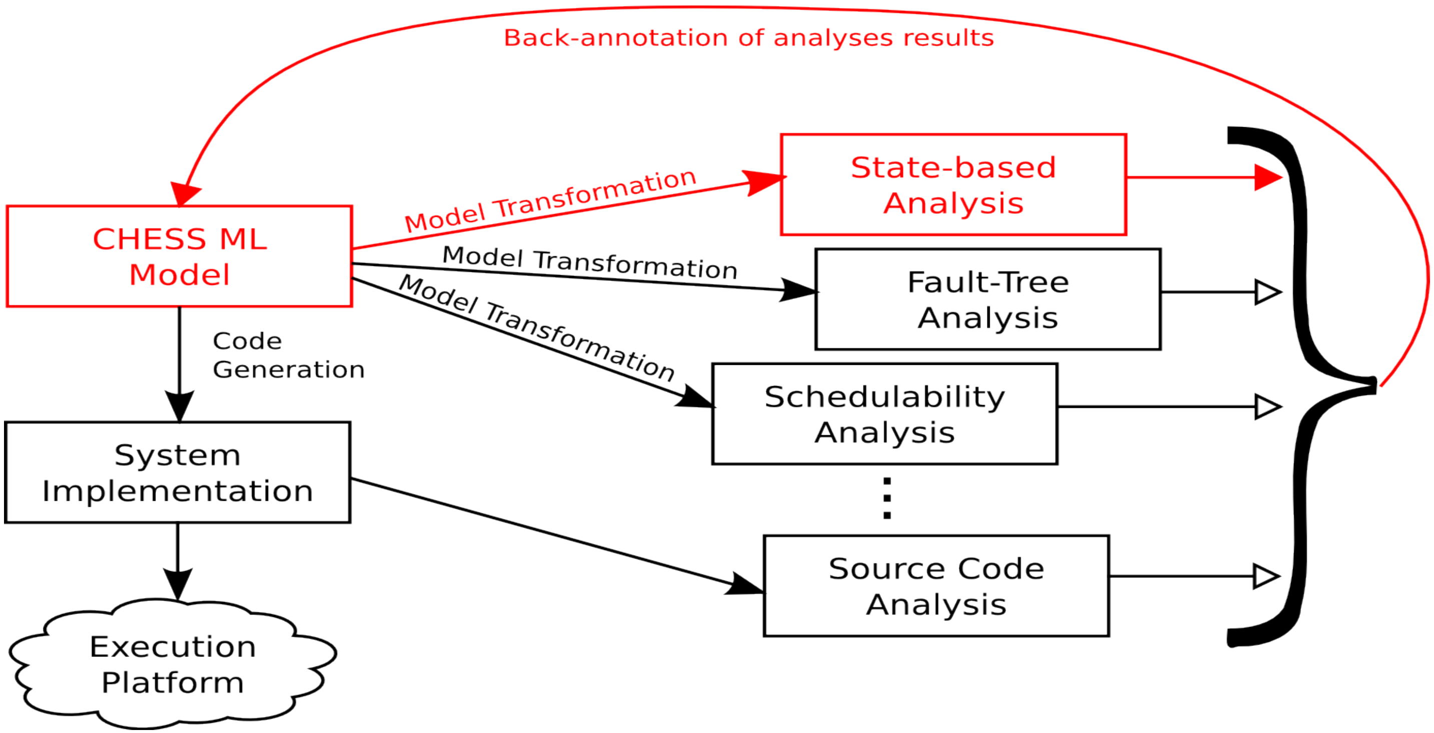
- Composition with guarantees for High-integrity Embedded Software components aSsembly (CHESS) - ARTEMIS-JU project
- Objective: develop an innovative MDE framework for component-based system development supporting
  - specification, analysis, and verification of extra-functional properties (mainly dependability and predictability)
- CHESS Framework
  - CHESS Modeling Language (CHESS ML), based on UML, SysML, and MARTE
  - CHESS Editor, based on MDT/Papyrus and Eclipse
  - CHESS Plugins, implementing model transformations to support different kind of analysis and code generation

The CHESS Framework (2/2)

- The CHESS methodology supports different analysis techniques:
  - Fault Tree Analysis (FTA)
  - Failure Modes, Effects, and Criticality Analysis (FMECA)
  - Failure Propagation and Transformation Calculus (FTPC)
  - State-based analysis (e.g., using Stochastic Petri Nets)
- Each analysis technique requires a set of information related to dependability, and some of them are shared
- The dependability concepts are then instantiated into the **CHESS Dependability Profile,** enriching a CHESS model with dependability related information
- The analysis models are automatically derived from the same high-level language

Overall CHESS workflow

– Automated transformation to an extension of Generalized Stochastic Petri Nets

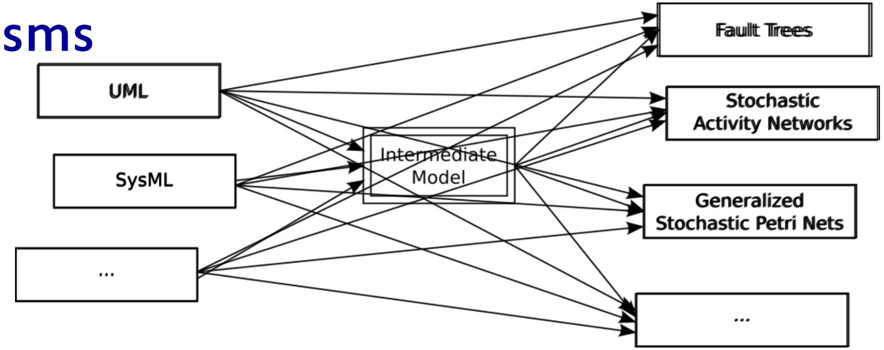


Main dependability concepts captured

- Within each (hw/sw) component
  - The fault-error-failure chain
- Between (hw/sw) components
  - Failure propagation
- Fault tolerant structures
- Maintenance activities (both preventive and corrective)
- Error detection activities
- Different types of analyses (transient, interval of time)
- Many Metrics of interest (reliability, availability)

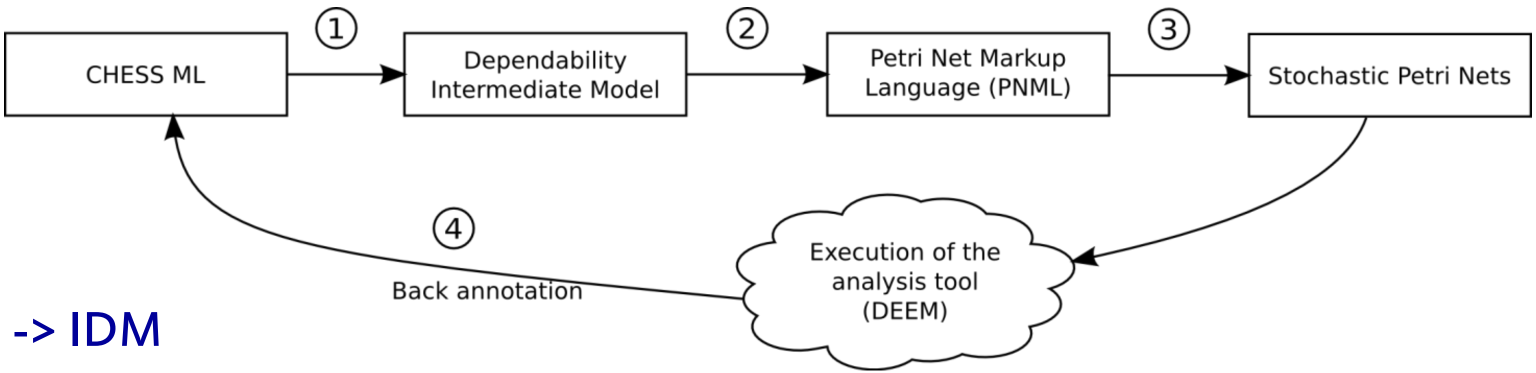
Intermediate Dependability Model

- The key element of the transformation process for state-based analysis is the definition of an Intermediate Dependability Model (IDM)
  - Intermediate representation of the system where only information that is useful for the analysis is retained
- Introduces an additional level of abstraction
  - Abstracts from the high-level engineering description
  - Abstracts from the low-level implementation in the selected analysis formalism
- Easier implementation of transformations
- Easier to switch to other high-level modeling languages and/or to other analysis formalisms



Workflow for state-based analysis

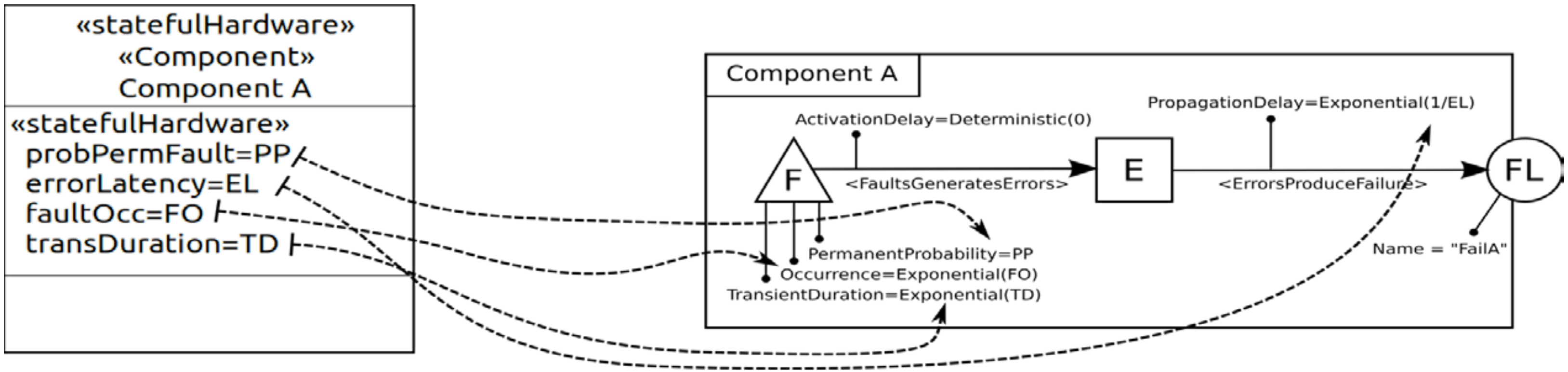
- composed of four major steps



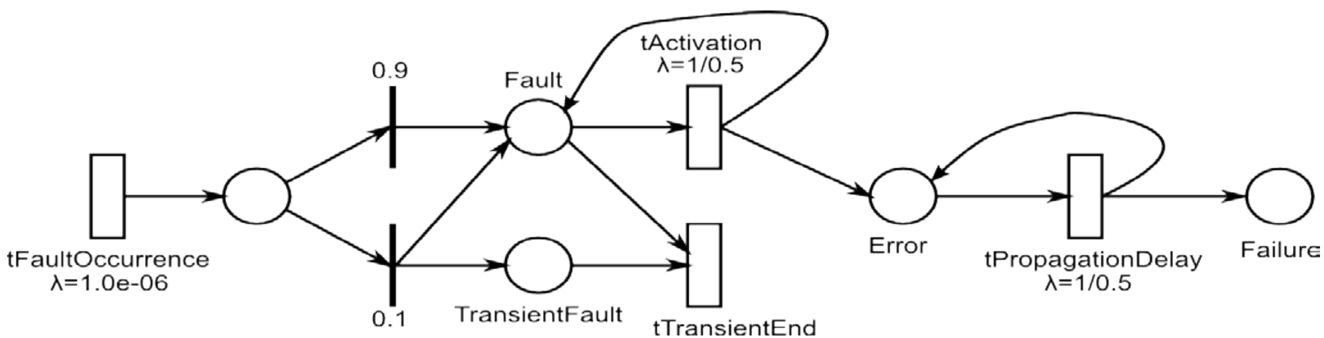
1. CHESS ML -> IDM
2. IDM -> PNML
   - The information contained in the IDM model is "implemented" in a Stochastic Petri Net model
   - PNML: an XML-based interchange language for Petri Nets, currently under standardization
3. PNML -> Analysis model (DEEM input model)
4. Backannotation
   - The results of the analysis are used to enrich the starting CHESS ML model (backannotation)
   - This new values could be used to perform subsequent analyses (possibly with other techniques)

Transformation example: Stateful Hardware elements

From CHESS ML to IDM...

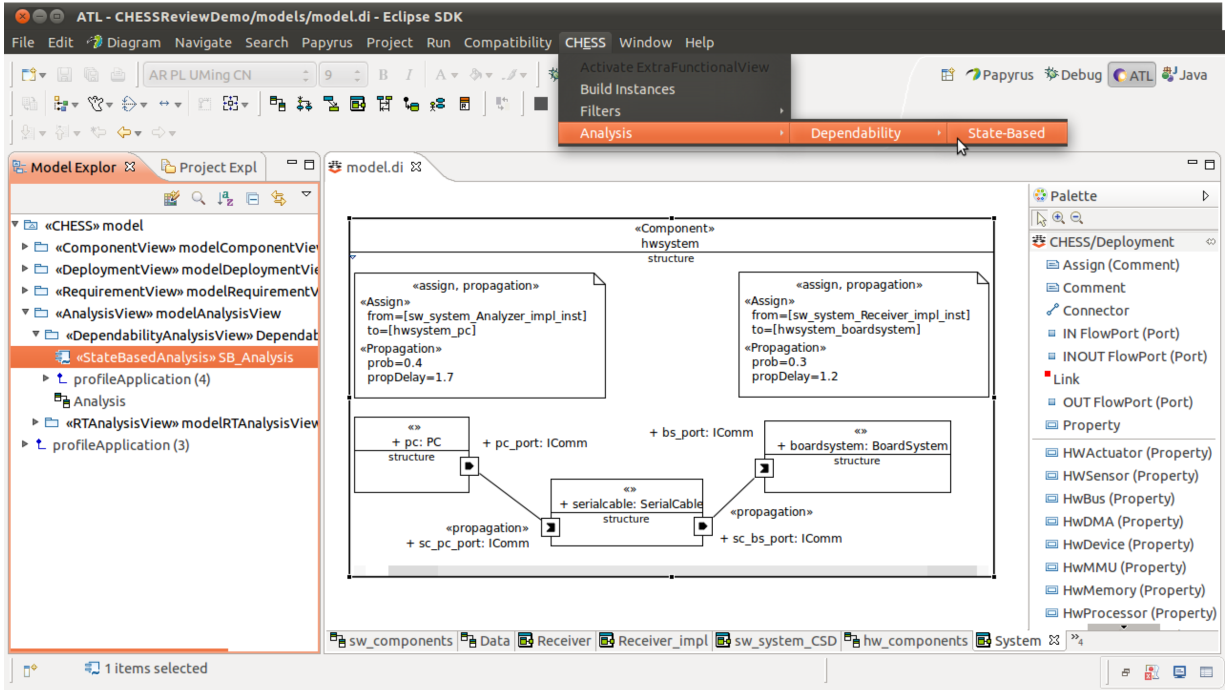


...and the resulting Stochastic Petri Net (sub)model



The CHESS Environment - Screenshot

- Latest technology
  - Eclipse 3.7 “Indigo”
  - MDT/Papyrus 0.8.1
- Cross-platform
  - Runs on Windows, Linux



Directions for the future

- **Some open research challenges in model-based resilience assessment, based on the lesson learned.**

Open research challenges (1)

- Addressing the role of modelling and quantitative evaluation in a more comprehensive assessment process
  - Need of a composite and trustable assessment framework including **complementary evaluation techniques**, (e.g. modelling and experimental measurements).
  - Mechanisms to ensure the cooperation and the integration of these techniques, in order to provide realistic assessments of architectural solutions and of systems in their operational environments.
  - Assessment of the **approximations** introduced in the modelling and solution process to manage complexity, as well as their impact on the final results.

Open research challenges (2)

- Need for a comprehensive modelling framework that can be used to assess the impact of **accidental faults** as well as **malicious threats** in an **integrated way**.

Open research challenges (3)

- Usage of quantitative (model-based) evaluation methods to support the effective use of adaptation mechanisms in dynamic and adaptable systems.
  - Efficient on-line mechanisms are needed to monitor the environment conditions of the system and to dynamically adapt to their changes.
  - Dynamic model construction and efficient model solution for providing the results online thus supporting dynamic adaptation

Open research challenges in the ‘Future Internet’ scenarios -1

- Adaptation of the validation methods (including model based ones) to cope with the Future Internet.
  - Traditional V&V approaches (applied to embedded critical systems) seem not adequate for the current and forthcoming large-scale and dynamic **service oriented systems**.
- Principal Challenging systems’ features:
  - Predominance of agility in the software development methodologies;
  - **Incremental software release development style**;
  - Unavailability of benchmarking and assessment standards.

Open research challenges in the ‘Future Internet’ scenarios - 2

- Need for online dynamic validation methods, tools and techniques capable of assuring the quality of
  open, large-scale, dynamic service systems
  without fixed system boundaries,

- addressing the complete service and software life cycle.

AMBER research roadmap

- The research challenges just described have been included and merged with others in a comprehensive and structured research roadmap defined within the FP7 CA AMBER:

Assessing, Measuring, and BEnchmarking Resilience

- The structured research roadmap consists a list of research directions worth pursuing, with associated priorities:
  - Scientific and technological foundations
  - Measurement and assessment
  - Benchmarking
  - Education, training, standardization and take up
- For each area of investigation, the roadmap specifies:
  - Needs and challenging issues
  - Objectives and Actions for their achievement

Final statements

- Target is moving!!

  .... And is changing!!

- Security is very important but **NOT** the only concern
  - accidental faults and deliberate attacks have to be considered alltogether

- Quantitative assessment needs to be integrated in development processes and in lifecycles (whatever they are..)

Some credits

Paolo Lollini

HIDENETS
Highly DEpendable IP-based NETworks and Services

amber

CHESS

ResilTech
Technologies for Resilience

Some references

1. A. Bondavalli (ed), L'Analisi Quantitativa dei Sistemi Critici. Esculapio. 2011.
2. FP7 - 216295 AMBER - Assessing, Measuring, and Benchmarking Resilience. Deliverable D3.2: Final Research Roadmap, Dec., 2009.
3. Paolo Lollini, Andrea Bondavalli and Felicita Di Giandomenico. A decomposition-based modeling framework for complex systems. In IEEE Transactions on Reliability, Volume 58, Issue 1, pp. 20-33, 2009.
4. Andrea Bondavalli, Ossama Hamouda, Mohamed Kaâniche, Paolo Lollini, Istvan Majzik, and Hans-Peter Schwefel. The HIDENETS Holistic Approach for the Analysis of Large Critical Mobile Systems. In *IEEE Transactions on Mobile Computing*, Volume 10, Issue 6, pp. 783 – 796, June, 2011.
5. Leonardo Montecchi, Paolo Lollini and Andrea Bondavalli. Towards a MDE Transformation Workflow for Dependability Analysis. To appear in Proc. of the *16th IEEE International Conference on Engineering of Complex Computer Systems* (ICECCS 2011), pp.157-166, Las Vegas, USA, 27-29 April, 2011.

Thank you!

Questions?