Equivalence-based Analysis of Performability-Aware Systems

Introduction

Problem:

- analyzing both *quantitative* aspects (e.g. performance) and *dependability* aspects (such as reliability, safety, security, and availability)
- in a component-oriented fashion

Goal:

- guiding the system design towards a balanced trade-off among all these aspects

Solution:

- integrated view
- equivalence-based integrated analysis

Scenario:

- a single component may cope with a sole specific aspect in a one-to-one fashion, or else crosscutting aspects may be handled by several components.
- in any case, the components may interfere each other when pursuing the goal of satisfying the requirements of different aspects.

Examples:

- mechanisms for controlling power-consumption may interfere with service availability and, in particular, with performance indices like service response time.
- access control mechanisms dedicated to security aspects may interfere with QoS parameters like system throughput.

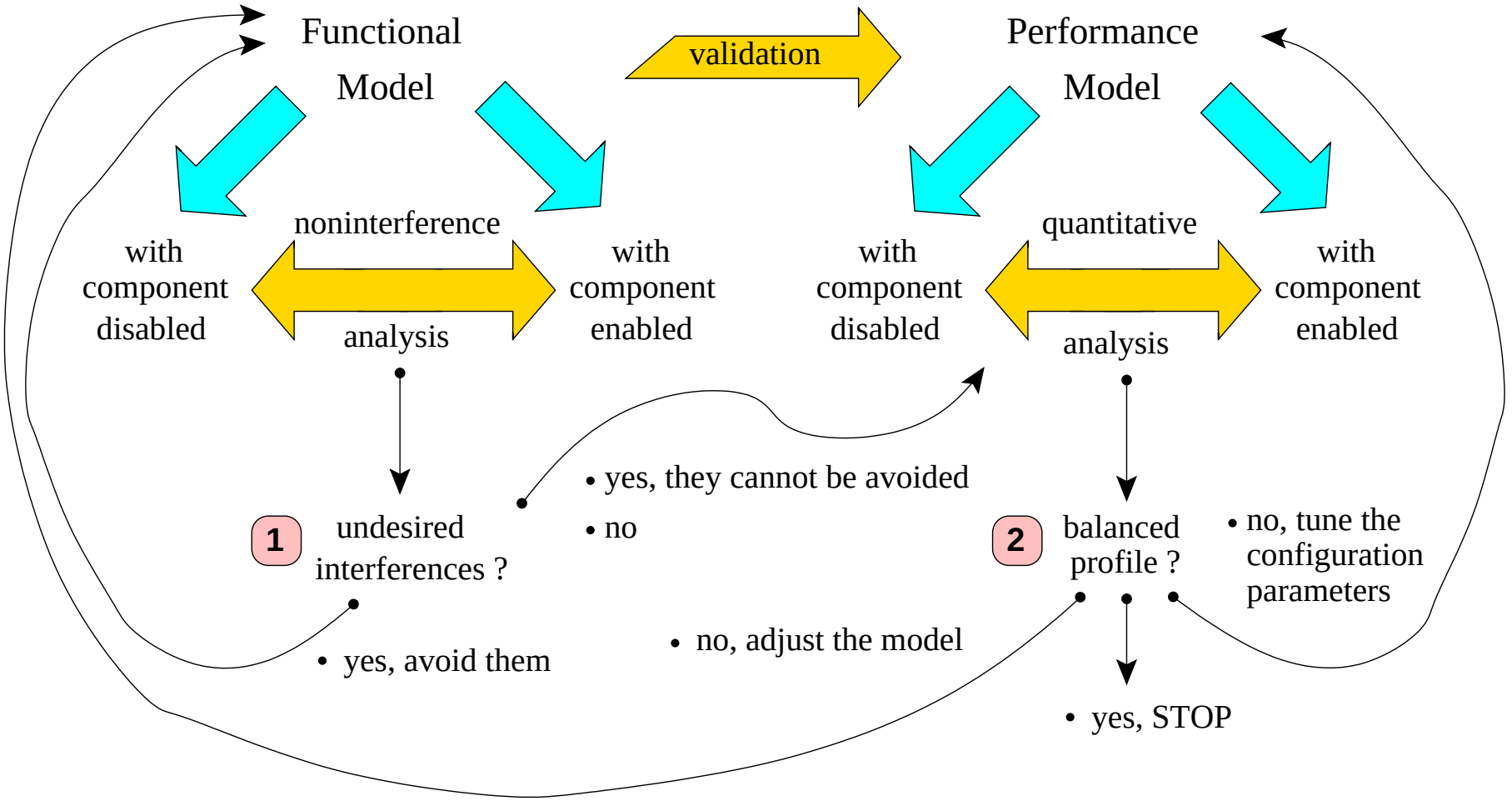
Integrated Methodology

Goal:

- evaluating the capability of a components of interfering with the behaviors (of other components) aiming at satisfying the requirements of specific aspects.
- the ultimate goal is to reach a balanced trade-off among all the functional and nonfunctional aspects.

Approach:

1. performing a noninterference check to the nondeterministic view of the system architecture in order to assess the interference of every component on the dependability aspects.
2. applying quantitative analysis techniques to the performance view of the system architecture in order to assess the impact of the interfering components on the performability aspects.


Functional
Model
validation
Performance
Model
noninterference
analysis
with
component
disabled
with
component
enabled
quantitative
analysis
with
component
disabled
with
component
enabled
1
undesired
interferences ?
• yes, they cannot be avoided
• no
• yes, avoid them
2
balanced
profile ?
• no, tune the
configuration
parameters
• no, adjust the model
• yes, STOP

Phase 1: Noninterference Analysis

Steps:

a. provide a functional description of the system

b. choose the dependability property of interest and single out the components dedicated to this aspect

c. determine the components of which the interference has to be evaluated

d. perform the noninterference check

e. decide whether the revealed interference is tolerable or not

Phase 1: Noninterference Analysis

Examples:

- determine the influence of faults triggered by a component upon the behavior of system components performing safety-critical applications.
- determine the influence of events triggered by non-trusted components upon the behavior of system components performing security-critical applications.
- determine the impact of mechanisms for controlling power consumption on aspects like service availability and reliability.

Noninterference Theory

Original idea:

- a group of high-security level users, employing confidential operations only, is not interfering with a group of low-security level users, observing public operations only, if what the first group of users can do with the confidential operations has no effect on what the second group of users can see.

- in the security setting, noninterference analysis can reveal direct and indirect information flows, called *covert channels*, that violate the access policies based on the different access clearances assigned to different user groups.

Example: a Noninterference Property

- action names are divided into two disjoint sets:
  - *High*, representing system activities at high-security level
  - *Low*, representing system activities at low-security level
- a system model Q has no covert channels if the system view where all the high-level activities are **hidden** to low-level observers, is indistinguishable with respect to the system view where these activities are **prevented** from execution:

$$Q/High \approx_B Q \backslash High$$

Noninterference Theory

General idea:

- a system execution can be viewed as an information flow.
- a group of system components (high components), described by a certain set of behaviors, is not interfering with another group of system components (low components) if the behaviors of the first group of components have no effect on what the second group of components can see.
- in this more general setting, noninterference analysis can reveal covert channels indicating the existence of undesired information flows among component behaviors that are responsible for compromising several different dependability aspects.

Example: a Noninterference Property

- let $\mathcal{A}$ be an architectural description with AEIs $K, C_1, \ldots, C_n, B_1, \ldots, B_m$.
- suppose to be interested in evaluating the impact of K on the functional behavior of $C_1, \ldots, C_n$ that is related to a specific system aspect.
- action names are divided into two disjoint sets:
  - $High \subseteq Name$, representing the system activities performed by K of which we intend to evaluate the impact.
  - $Low \subseteq Name$, representing the system activities performed by $C_1, \ldots, C_n$ and related to the behavior we intend to monitor.
- all the remaining activities carried out by the system are simply disregarded and can be hidden.
- at the architectural level, the different system views can be defined as behavioral modifications by employing the static operators for hiding and restriction.

- K does not interfere with $C_1, \ldots, C_n$ if the *low* behavior of $C_1, \ldots, C_n$ with the *high* activities of K being made unobservable is equivalent to the *low* behavior of $C_1, \ldots, C_n$ with the same activities being prevented from taking place:

$$\boxed{\begin{array}{c} [\![C_1, \ldots, C_n, K, B_1, \ldots, B_m]\!]_{\mathcal{A}} / (Name - Low^*) \\ \approx_B \\ [\![C_1, \ldots, C_n, K, B_1, \ldots, B_m]\!]_{\mathcal{A}} \backslash High^* / (Name - Low^*) \end{array}}$$

  where:

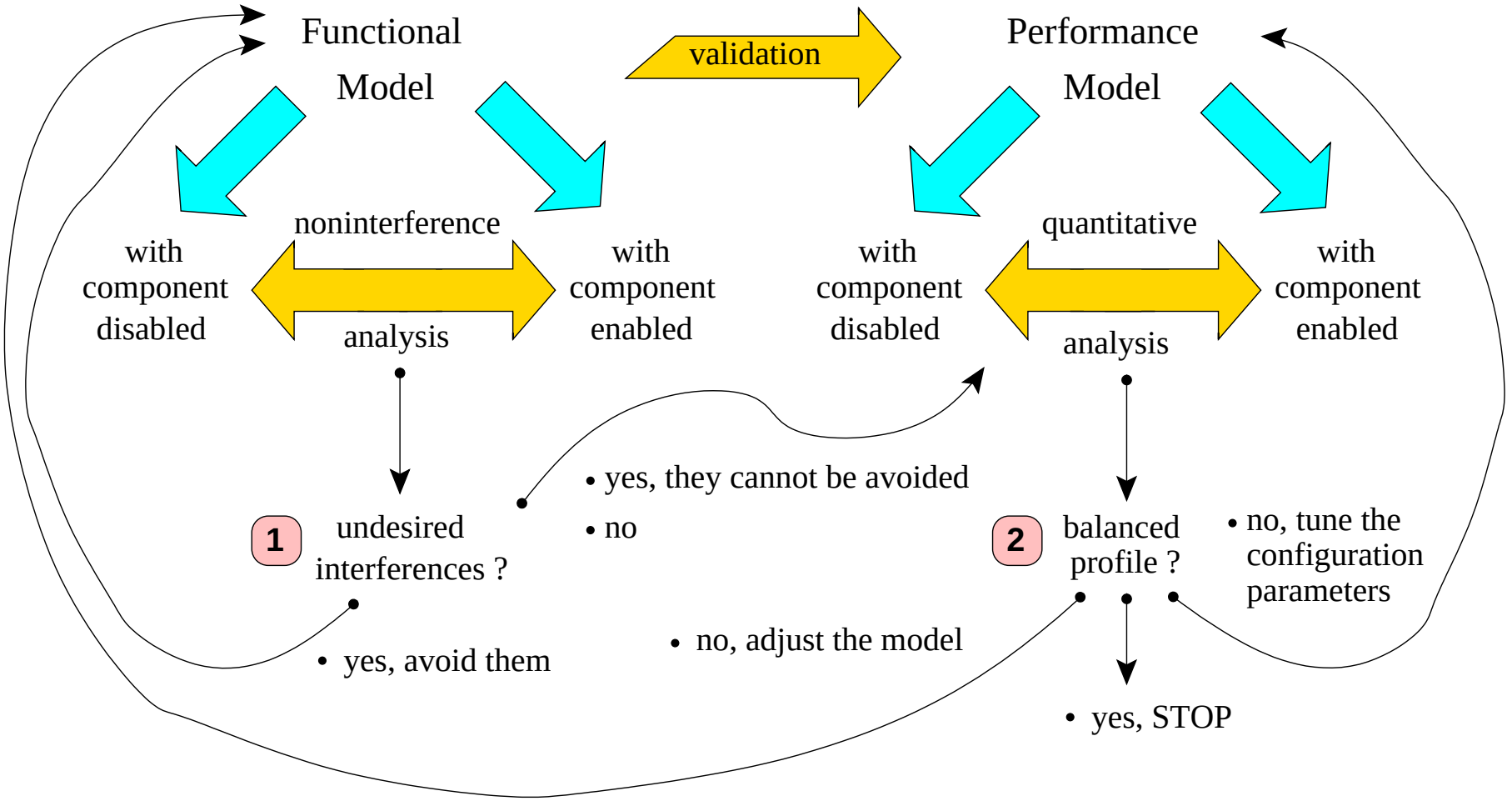
  - $High^* = \mathcal{S}^{High}(K; \mathcal{A})$, where $\mathcal{S}^{High}(K; \mathcal{A}) \subseteq \mathcal{S}(K; \mathcal{A})$ is such that $\varphi_{K;\mathcal{A}}(a) \in \mathcal{S}^{High}(K; \mathcal{A})$ if and only if $a \in High$.
  - $Low^* = \bigcup_{i=1}^{n} \mathcal{S}^{Low}(C_i; \mathcal{A})$ where $\mathcal{S}^{Low}(C; \mathcal{A}) \subseteq \mathcal{S}(C; \mathcal{A})$ is such that $\varphi_{C;\mathcal{A}}(a) \in \mathcal{S}^{Low}(C; \mathcal{A})$ if and only if $a \in Low$.

Component-Oriented Noninterference Check

- for specific system topologies, the noninterference check turns out to be an architectural modification of the compatibility and interoperability checks.
- the interference of K on the behavior of $C_1, \ldots, C_n$ can be inferred in a more efficient way whenever:
  1. K is the central AEI of a star topology that includes $C_1, \ldots, C_n$.
  2. K is an AEI of a cycle that includes $C_1, \ldots, C_n$.

Results of the Noninterference Check

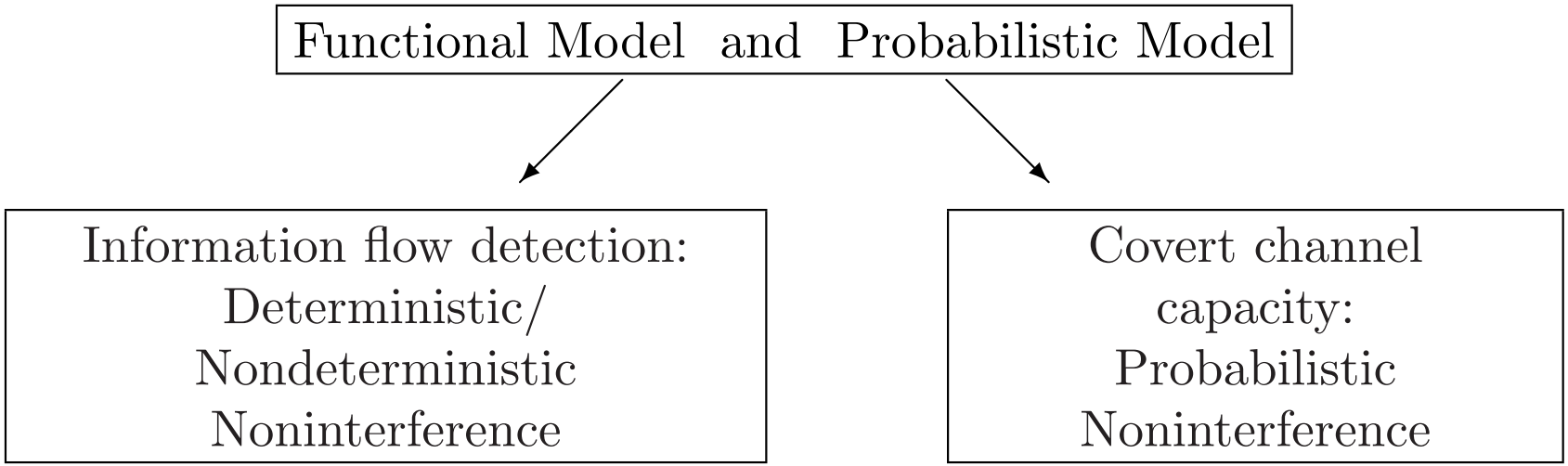
- **no information flow is revealed**: we have the guarantee that K is functionally transparent with respect to the activities of $C_1, \ldots, C_n$ modeling the dependability aspect under consideration.
- **an undesired, direct or indirect, information flow is revealed**: diagnostic information provided by the noninterference check can be employed to determine the causes of the interference.
- If the interference can be eliminated with a minor impact on the functionalities of the system behavior, the diagnostic information can be employed to suitably modify the functional model.
- In contrast, for all information flows that are either unavoidable or tolerated as they would require a significant revision of the functional model, it is necessary to estimate the interference.


Functional
Model
validation
Performance
Model
noninterference
analysis
with
component
disabled
with
component
enabled
quantitative
analysis
with
component
disabled
with
component
enabled
1
undesired
interferences ?
• yes, they cannot be avoided
• no
• yes, avoid them
2
balanced
profile ?
• no, tune the
configuration
parameters
• no, adjust the model
• yes, STOP

Phase 2: Quantitative Analysis

- what do we mean by *quantitative behavior*?
- different extensions: probabilities vs. time.
- the functional model must be refined into a fine-grained model.
- different approaches:
  - (probabilistic vs. Markovian) noninterference
  - performance comparison

Quantitative behavior expressed by probabilities



- exact analysis
- 0/1 result

- approximate analysis

  Probabilistic noninterference

  (Gray, Dipierro et al., Aldini et al., Smith, Desharnais et al., ...)

- $[0; 1]$ result

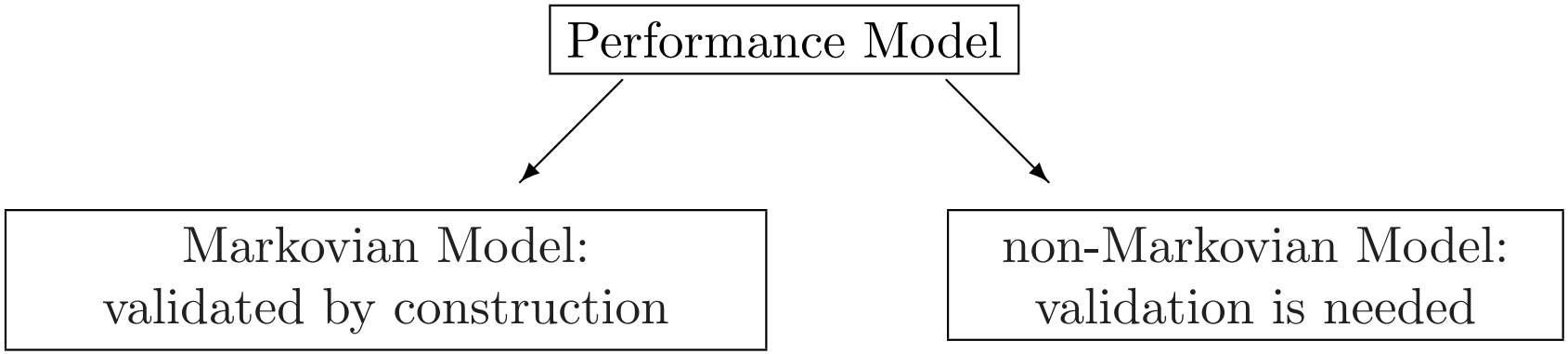
  $\uparrow$

  best-case/average/worst-case ?

(Approximate) Probabilistic Noninterference

- **Model**: generative and reactive probabilistic systems
- **Equivalence**: weak probabilistic bisimulation $\approx_{\mathrm{PB}}$ (two states belong to the same class if their weak transition probabilities are equal).
- **Noninterference Check**: if passed, see the nondeterministic case, otherwise an interference is revealed. *Can we measure the covert channel capacity?*
- **Approximation**: two states s and s' may belong to the same class if their weak transition probabilities differ and the difference is negligible.
- **Problem**: find the relation R approximating $\approx_{\mathrm{PB}}$ that minimizes these differences (*work in progress*). The maximum distance between any pair of states of the same class of R represents the covert channel capacity.
- **Approach**: why searching for R instead of, e.g., a pseudometric? We prefer an approach supporting diagnostic information and minimization.

Quantitative behavior expressed by time



- Markovian Noninterference
- Performance comparison through reward-based numerical analysis (MSL)

- Performance comparison through simulation

Markovian Noninterference

- **Model**: Markovian Model (CTMCs).
- **Equivalence**: weak Markovian bisimulation $\approx_{\mathrm{MB}}$ (two states belong to the same class if their weak exit rates are equal).
- **Noninterference Check**: if passed, see the nondeterministic case, otherwise an interference is revealed. *Can we measure the covert channel capacity?*
- **Approximation**: *open problem.*

Performance Comparison

- Define the performance metrics that are directly related to the bandwidth of the revealed covert channel (follow the guidelines provided by the diagnostic information).
- Alternatively, in the case no information flow has been revealed, define the QoS-related metrics of interest.
- In any case, the resulting performance figures reveal whether a balanced tradeoff between dependability related aspects - in terms of bandwidth of each covert channel - and QoS - in terms of performance measures like system throughput and response time - is met or not.

Performance Comparison

- Depending on the obtained results, the performance figures are then used as a feedback to decide whether the system is to be tuned by changing the configuration parameters that affect the metrics of interest, or else it is necessary to adjust the functional model and restart the integrated analysis.
- The choice is mainly guided by the design requirements, e.g. strict vs. relaxed dependability needs and loose vs. tight QoS.

Performance Comparison

Examples:

- determine the bandwidth of covert channels revealing security violations in order to estimate, e.g., the amount of sensitive information leaked per unit of time, or the overhead due to strategies minimizing these covert channels, or else the tradeoff between security requirements and quality of service parameters.
- estimate the impact of mechanisms for controlling power consumption on the service availability by measuring the service response time.

Conclusion

Further open problems:

- other weak equivalences (bisimulation may distinguish too much)
- approximation of Markovian equivalences
- extensions of the methodology (?)