UniMI/UniBG

Elvinia Riccobene, Chiara Braghin, Paolo Arcaini

UNIVERSITÀ DEGLI STUDI
DI MILANO

Dipartimento di Tecnologie dell'Informazione
Università degli Studi di Milano

Angelo Gargantini, Patrizia Scandurra

Dip. di Ing. dell'Informazione e Metodi matematici
Università di Bergamo

Obbiettivi per WPs

- **WP1 (Notazioni per la Modellazione)**
  - Sviluppo di un ambiente semantico per DSMLs di architetture SW adattabili
- **WP2 (Tecniche e Strumenti per Verifica e Simulazione)**
  - Strumenti per l'analisi formale di modelli per sistemi software adattabili
- **WP4 (Tecniche e Tools per il Testing)**
  - model-based testing per applicazioni dinamiche affidabili
    - UniBG

Agenda

- **Risultati** primo anno
- **Obbiettivi** secondo anno
  - WP1 (*Notazioni per la Modellazione*)
  - WP2 (*Tecniche e Strumenti per Verifica e Simulazione*)
  - WP4 (Tecniche e tool per il Testing)
    - → A.Gargantini

Risultati -> direzioni

WP1

1. Sviluppo di un **framework semantico** per linguaggi metamodel-based
   - Applicazione a diversi casi (DSL e non)

2. **Integrazione** tra MDE e Formal Methods per lo sviluppo e l'analisi di sistemi
   - Applicazione nel contesto di EMF ed ASM

Risultati -> direzioni

WP1

1. Sviluppo di un **framework semantico** per linguaggi metamodel-based
   - Applicazione a diversi casi (DSL e non)

2. **Integrazione** tra MDE e Formal Methods per lo sviluppo e l'analisi di sistemi
   - Applicazione nel contesto di EMF ed ASM

Motivazioni

- Framework MDE per lo sviluppo di DSMLs (Domain-Specific Languages)
  - supportano
    - formalismi (linguaggi per metamodelli, come MOF, Ecore, etc.) per la definizione sintattica (metamodello) di un DSL
      - basati su class diagrams
    - meccanismi di trasformazione di modelli,
  - ma non consentono astrazioni comportamentali a livello di linguaggi per metamodelli
    - semantica di metamodelli è data in linguaggio naturale
      - vedi i tanti lavori per formalizzare la semantica di UML

WP1: risultato 1

- Sviluppo di un **ASM-based semantic framework**
  - per definire una semantica formale ed eseguibile per linguaggi metamodel-based
  - A. GARGANTINI, RICCOBENE E., P. SCANDURRA (2009). A semantic framework for metamodel-based languages. AUTOMATED SOFTWARE ENGINEERING, vol. 16, n. 3-4, p. 415-454
  - Perché su ASMs?
    - astratte e formali, ma prive di "formal overkill"
    - eseguibili e capaci di catturare diversi MoCs
    - dotate di un metamodello
  - Diverse **tecniche translational/weaving** permettono di usare le ASMs in framework MDE

ASM-based SEMANTICS di MM

- Idea generale:
  - **A**: metamodello di un DSML
    - class model + OCL constraints
  - **A** ha una semantica ben-definita se
    - identificato un **semantic domain S** e
    - definito un **semantic mapping M_S : A → S**, tale che per ogni **m ω A** (read: m conforms to A) :
      $M_S(m)$ = modello semantico di m
  - Assumi $S = S_{AsmM}$
    - S_{AsmM} : semantic domain di ASM metamodel (AsmM)
  - Definiamo **$M_S = M_{S_{AsmM}}$ o M**
    - $M_{S_{AsmM}}$: AsmM → S_{AsmM} semantic mapping di AsmM
    - M: A → AsmM è la **building function** che associa una ASM ad un terminal model m ω A

ASM-based SEMANTICS di MM

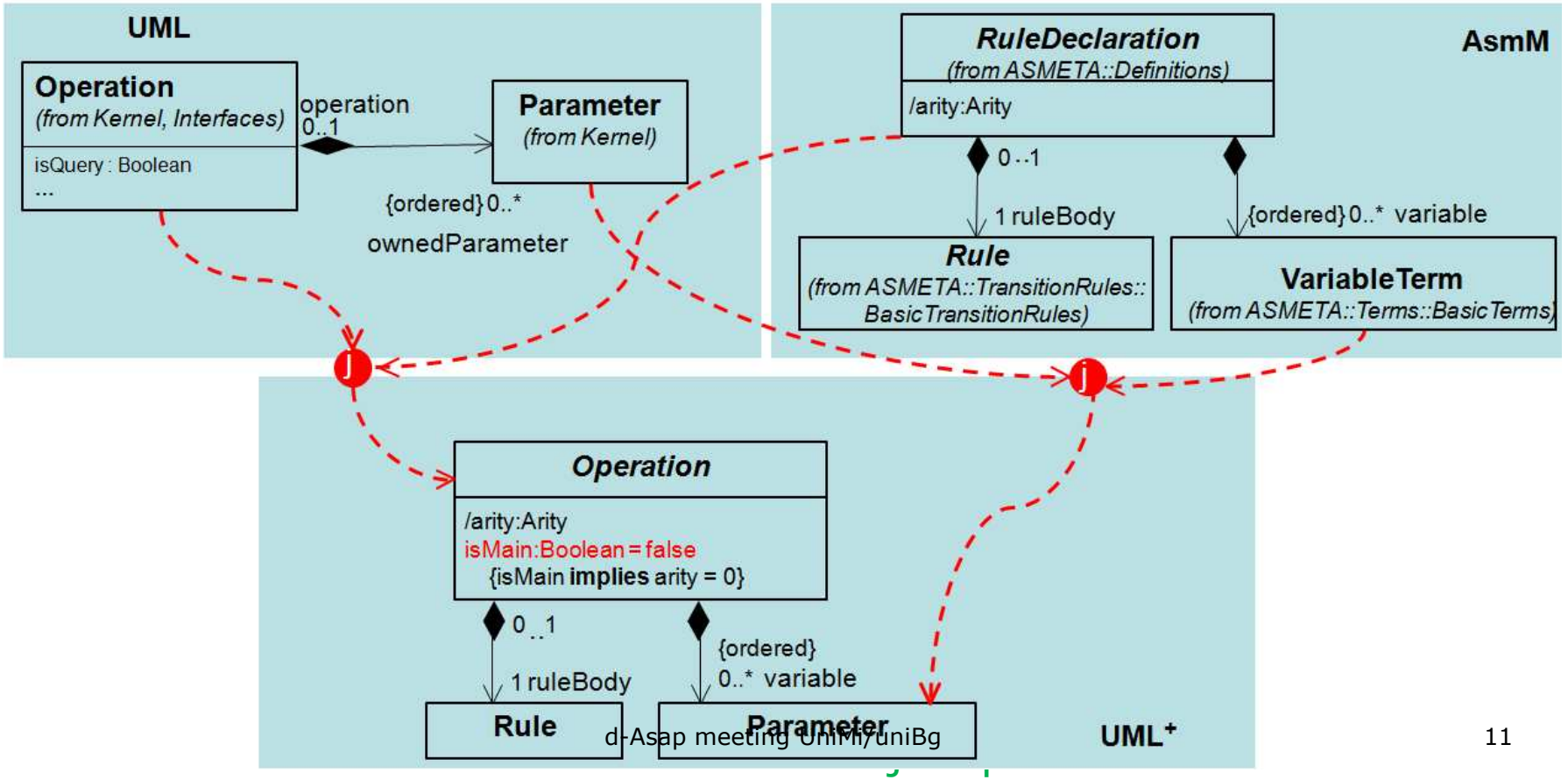
- Diverse tecniche per definire la building function M: A → AsmM
  - **Traslazionali**
    - mappano un modello in una ASM (modello formalmente ben definito)
  - **Weaving**
    - tesse i due metamodelli A e AsmM in uno nuovo

ASM-based SEMANTICS di MM

- Tecniche Traslazionali
  - in base al livello di astrazione dello stack di meta - modellazione
  - **mapping**: a livello **model**
    - Mappatura diretta da modello terminale ad ASM
  - **hooking**: a livello **metamodel**
    - Mappatura da metamodello + inizializzazione
      - $M(m) = i_A(\Gamma_A, m)$ con Γ_A: AsmM
  - **meta-hooking**: a livello **meta-metamodel**
    - **Mappatura da meta-linguaggio + regole + val. iniz.**
      - $M(m) = i_A(\omega(m))(\tau_A(\delta(\omega(m))), m)$
  - Risalendo lo stack di MM:
    - maggiore automazione nella trasformazione di modelli
    - maggior riuso
    - minore dipendenza della ASM risultante rispetto al modello terminale

ASM-based SEMANTICS di MM

- **Tecnica di Weaving**
  - tesse A con AsmL in A+



WP1: risultato 1/a

- Applicazioni del framework semantico:
  - **Semantica di Avalla** (Asmeta validation language)
    - Linguaggio domain-specific per costruire scenari di modelli ASM
    - Usa la tecnica di **mapping**
      - CARIONI A, GARGANTINI A, RICCOBENE E., SCANDURRA P (2008). A Scenario-Based Validation Language for ASMs. Proc. ABZ 2008, vol. LNCS 5238

WP1: risultato 1/b

- Applicazioni del framework semantico:
  - **Semantica di un profilo UML per SystemC**
    - Usa la tecnica **meta-hooking**
    - Definisce la semantica eseguibile di un profilo UML per SystemC (notazione – definita con la STM – per modelli di sistemi embedded)
    - Permette di mappare modelli grafici del profilo in modelli formali ASM allo scopo di poter applicare tecniche di analisi formale di modelli
      - RICCOBENE E., SCANDURRA P. (2010). An Executable Semantics of the SystemC UML Profile. Proc. ABZ 2010, vol. LNCS (to appear)

WP1: risultato 1/c

- Applicazioni del framework semantico:
  - **Eseguibilità di modelli a livello PIM** (Platform Independent Modeling)
    - Usa la tecnica di **weaving**
    - Si estende il meta-modello UML delle classi e della loro Action Semantics, ottenendo un formalismo che permette di esprimere sia aspetti statici che dinamici
    - Tale estensione è ottenuta "*tessendo*" la dichiarazione statica degli operatori di una meta-classe UML con gli aspetti comportamentali espressi come regole ASM
      - RICCOBENE E., SCANDURRA P (2009). Weaving executability into UML class models at PIM level. In Proc. BM-MDA'09, ACM Vol. 379

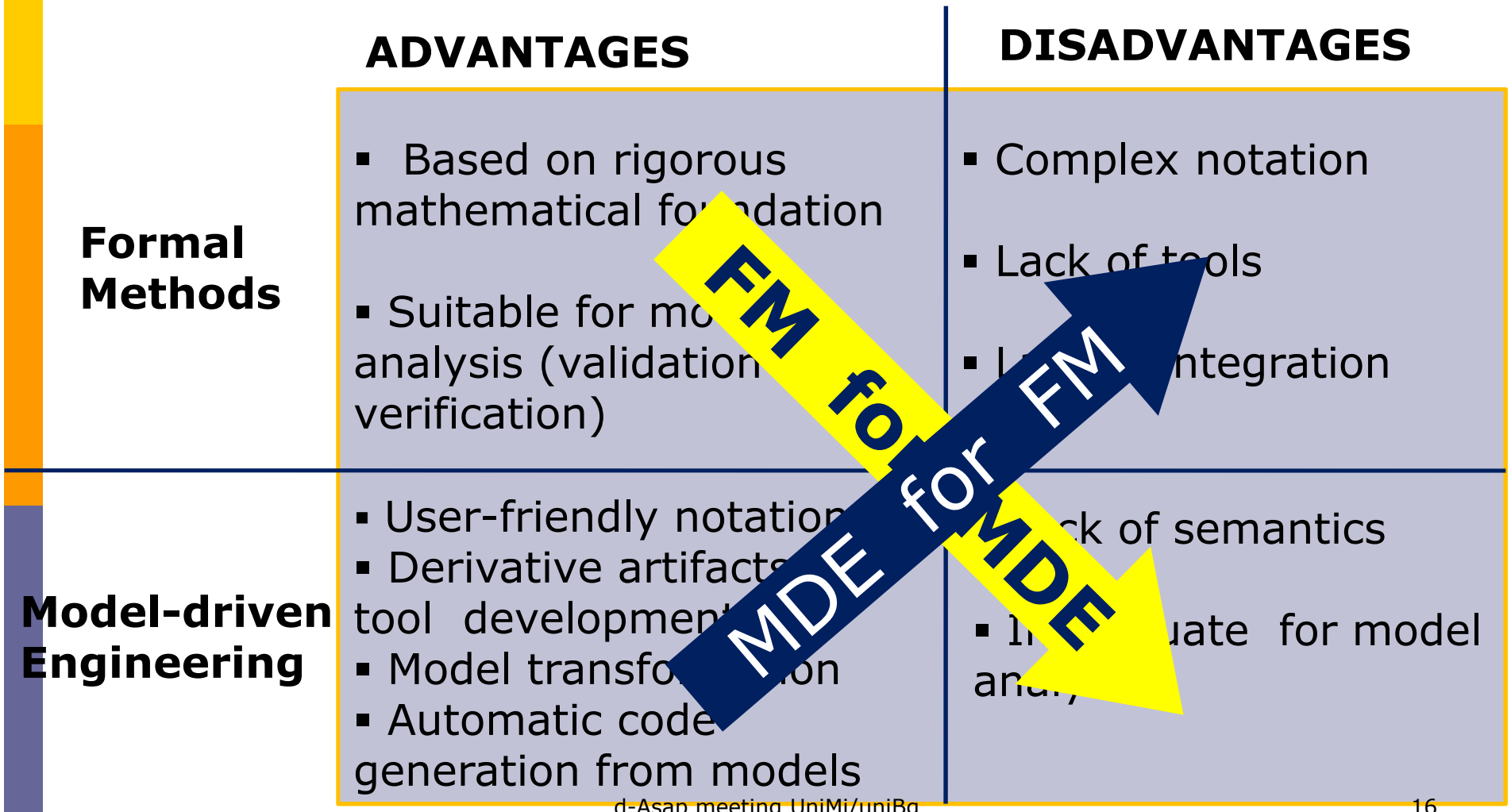
Risultati -> direzioni

WP1

1. Sviluppo di un **framework semantico** per linguaggi metamodel-based
   - Applicazione a diversi casi (DSL e non)

2. **Integrazione** tra MDE e Formal Methods per lo sviluppo e l'analisi di sistemi
   - Applicazione nel contesto di EMF ed ASM

Motivazioni



Motivazioni

MDE for FM

Formal Methods Improvements

- Intuitive **modeling notation**
  - possibly with graphical view of models
- Modeling **techniques**
  - to automatize system development
- Open and flexible **architecture**
  - for tools dev. & integration

FM for MDE

Model-driven Engineering Improvements

- Definition of the **behaviors** (semantics) of MDE models
- Techniques &methods for **formal analysis** of models
  - e.g. simulation, testing, property proving, model checking, etc.

WP1: risultato2

- **Integrazione tra FM e MDE framework**
  - Combina ASMs con EMF in un *in-the-loop-integration*

    - GARGANTINI A, RICCOBENE E., SCANDURRA P (2009). Integrating Formal Methods with Model-Driven Engineering. Proc. of Fourth International Conference on Software Engineering Advances (ICSEA'09), p. 86-92,2009 IEEE.
  - **Actività 1**:**Definisce un processo per la ingegnerizzazione di tool-set per FMs**
    - Applicato su ASM per costruire ASMETA
      - ARCAINI, CARIONI, GARGANTINI A, RICCOBENE E., SCANDURRA P (2010). A model-driven process for engineering a tool set for a formal method. Proc. WTBFM 2010, LNCS (to appear)
  - **Attività 2**: **uso del semantic framework per la definizione di un DSL (struttura + semantica)**

WP1: Obiettivi futuri

- Sulla base dei risultati raggiunti su:
  - Framework sematico
  - Integrazione tra ASM ed EMF
  - Sviluppo di tecniche per validazione e testing
- siamo in grado di passare a:
  - **Service-oriented component systems**
    - gli approcci component-oriented e service-oriented come paradigmi che supportano lo sviluppo di applicazioni adattabili dinamicamente
- per lo sviluppo di **notazioni** per modelli e **tecniche di analisi** di modelli

WP1: Obiettivi futuri

Service-oriented component systems

- **Stato dell'arte di notazioni**:
  - proliferazione di diversi linguaggi/formalismi per la modellazione di sistemi service-oriented
    - a livello business process & business services: BPMN, BPEL, UML4SOA, SOAML, Service Component Architecture (SCA), SRML (EU project SENSORIA), etc.

WP1: Obiettivi futuri

Service-oriented component systems

- **Applicare il framework semantico a SRML**
  - **Sensoria Reference Modelling Language** (sviluppato nell'ambito del progetto EU SENSORIA)
  - linguaggio per descrivere e comporre servizi
    - in a technology agnostic way
  - dotato di un modello di computazione/coordinazione
  - esistono tecniche di analisi qualitative e quantitative
    - ad es. con il model checker UMC (vedi S. Gnesi et al.)
- **Definire un formalismo ASM-based per la modellazione/composizione di servizi**
  - rigoroso, operazionale, composizionale, tool-supported

WP1 : Obiettivi futuri

Service-oriented component systems

- **Applicare la tecnica di hooking per SRML**
  - Intra-component behavior
  - Inter-component behavior (protocolli)
    - *SOA predefined interaction patterns* già formalizzati in ASMs
      - Alistair P. Barros, Egon Börger: A Compositional Framework for Service Interaction Patterns and Interaction Flows. ICFEM 2005: 5-35
      - Egon Börger, Bernhard Thalheim: Modeling Workflows, Interaction Patterns, Web Services and Business Processes: The ASM-Based Approach. ABZ 2008: 24-38
      - Egon Börger, Ove Sörensen: Execution Semantics for BPMN's Core Modeling Concepts (draft version)

Obbiettivi futuri (WP2)

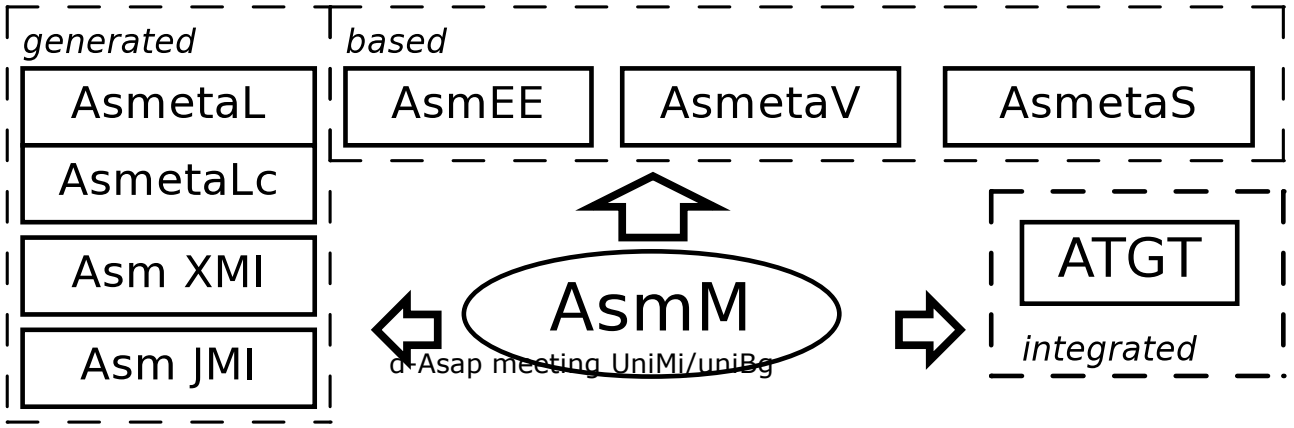
Service-oriented component systems

- **Applicare e potenziare ASMETA per analisi (validazione e verifica) di service oriented component systems**
  - Simulazione (interattiva, random, scenario-based)
  - Tecniche di composizione/decomposizione di proprietà
  - Verifica di proprietà (LTL/CTL) con tecniche di *compositional verification*
  - Run-time monitoring

WP2 : Tecniche e Strumenti per Verifica e Simulazione + WP4. Tecniche e Tools per il Testing

Base di partenza:

- ASMETA (ASM mETAmodelling) tool-set
  - http://asmeta.sf.net/
  - per la scrittura, controllo sintattico, simulazione, validazione e model-based testing di modelli ASM
    - Gargantini, Riccobene, Scandurra. A Metamodel-based Language and a Simulation Engine for Abstract State Machines, JUCS 14(12), 2008
    - CARIONI A, GARGANTINI A, RICCOBENE E., SCANDURRA P (2008). A Scenario-Based Validation Language for ASMs. Proc. ABZ 2008, LNCS 5238
    - Gargantini, Riccobene, Scandurra. Model-Driven Language Engineering: the ASMETA Case Study, ICSEA '08



Risultati -> direzioni

WP2

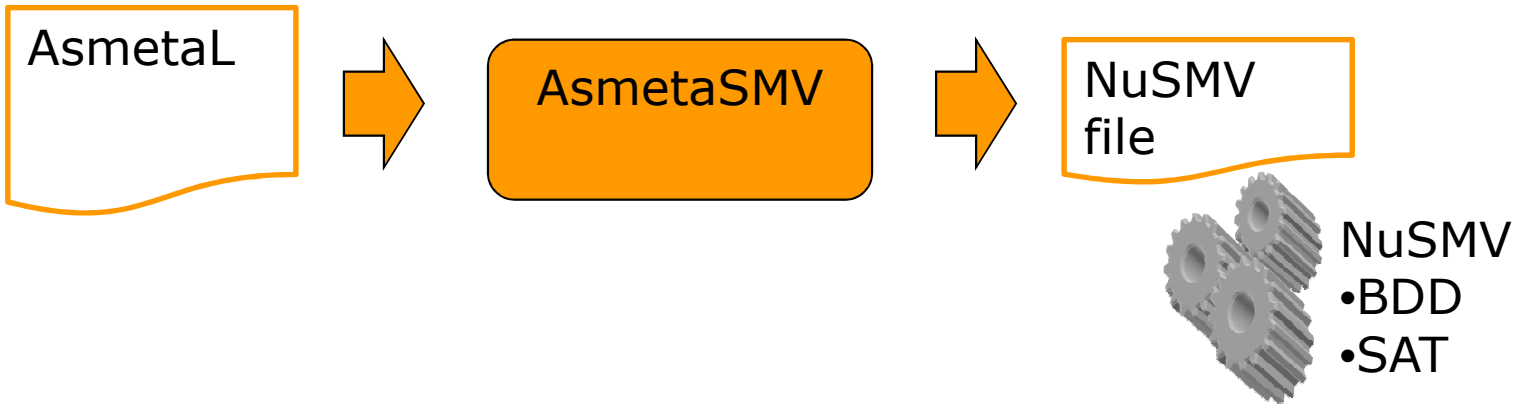
1. Potenziamento del tool-set
   - Verificatore basato su SMV
2. Applicazione degli strumenti per il model review
   - Tecnica per l'analisi statica di modelli ASM

WP4 (riuso di strumenti di WP2)

3. Generazione di casi di test
   - Mediante strumenti normalmente usati per la verifica come model checking o SMT/SAT solver

WP2: risultato 1

- Sviluppata una componente per la verifica di proprietà temporali (CTL ed LTL)
  - AsmetaSMV mappa ASMs in modelli NuSMV
    - P. ARCAINI, A. GARGANTINI, RICCOBENE E., (2010) AsmetaSMV: a Way to Link High-Level ASM Models to Low-Level NuSMV Specifications. Proc. ABZ 2010, vol. LNCS (to appear)



WP2: risultato 2

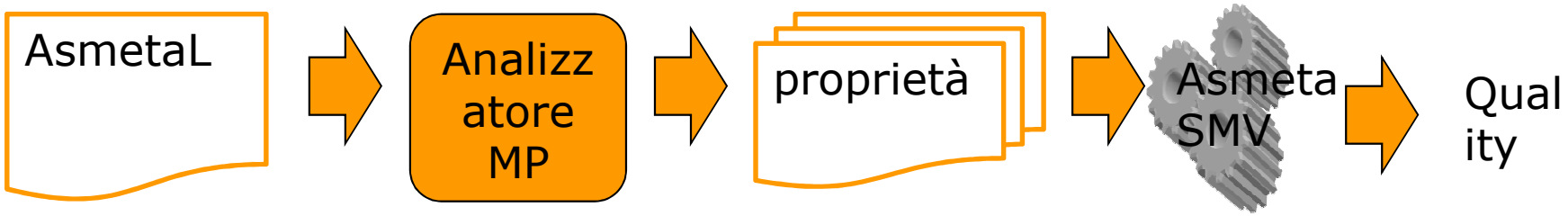
- Strumenti automatici per il model review
- Motivazione:
  - Verifica di proprietà potrebbe non essere fattibile (non riesco a provare) e neanche significativa (non sono sicuro che la specifica catturi i requisiti)
  - Tecniche più leggere rispetto la verifica formale orientate all'analisi statica
    - Intermedio tra simulazione e verifica formale

WP2: risultato 2

- tecnica di model review per ASMs basata sulla verifica di meta-proprietà che si possono assumere come misura di model quality assurance
  - ogni meta-proprietà si riferisce a un dato attributo che si vuole garantito del modello
  - P. ARCAINI, A. GARGANTINI, RICCOBENE E., (2010) Automatic review of Abstract State Machines by Meta Property Verification (submitted)
- Basata sull'osservazione di Parnas: "reviewers spent too much of their time and energy checking for simple, application-independent properties" which distracted them from the more difficult, safety-relevant issues."

Meta-proprietà

- Esempio di queste proprietà:
  - Nessun update inconsistente può mai avvenire, proprietà temporali vacuamente vere, trivial updates, …
- Individuazione di una famiglia di difetti tipici nelle ASM e loro modellazione mediante meta-proprietà.

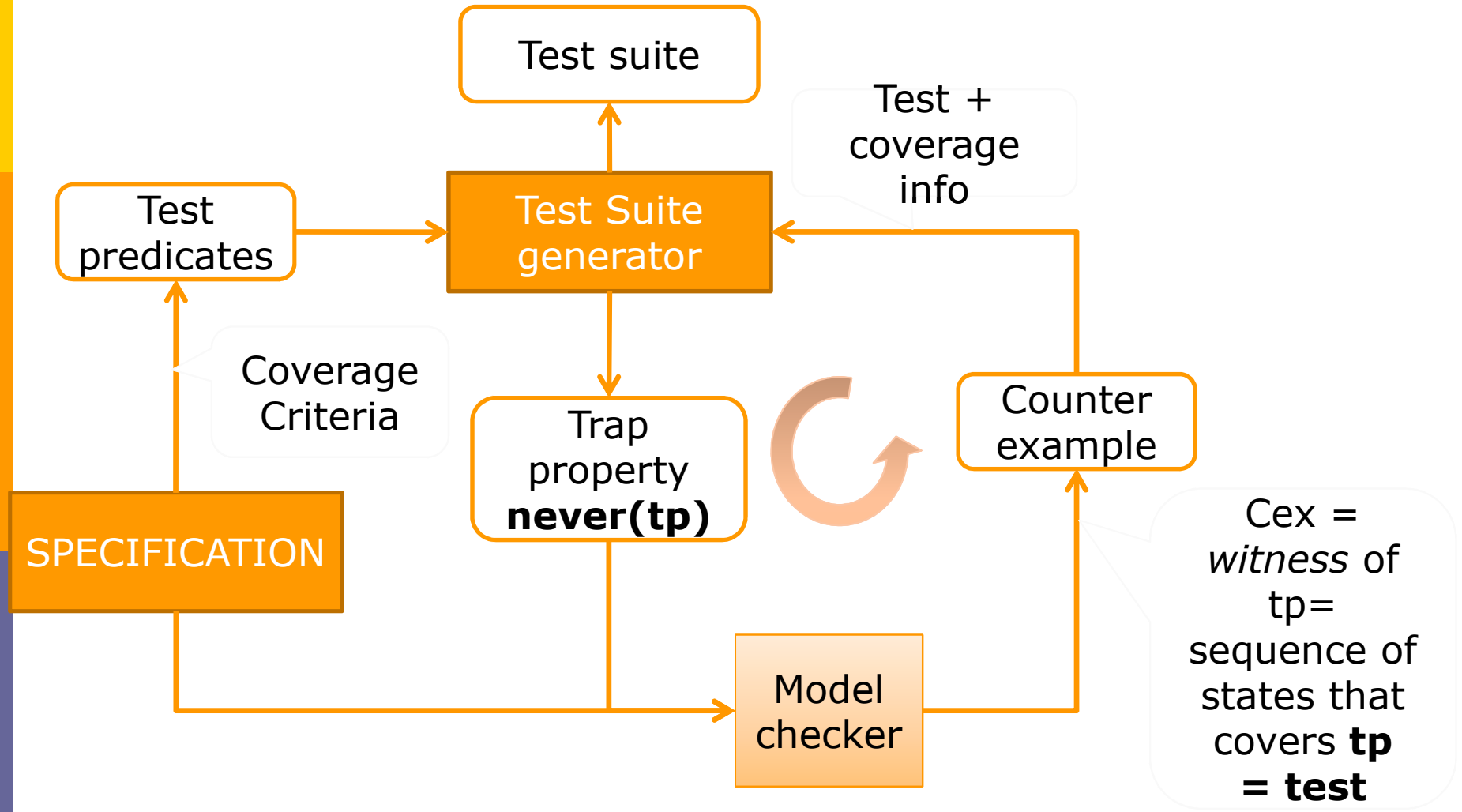


- Questo tipo di analisi da fare prima della verifica di proprietà mediante MC

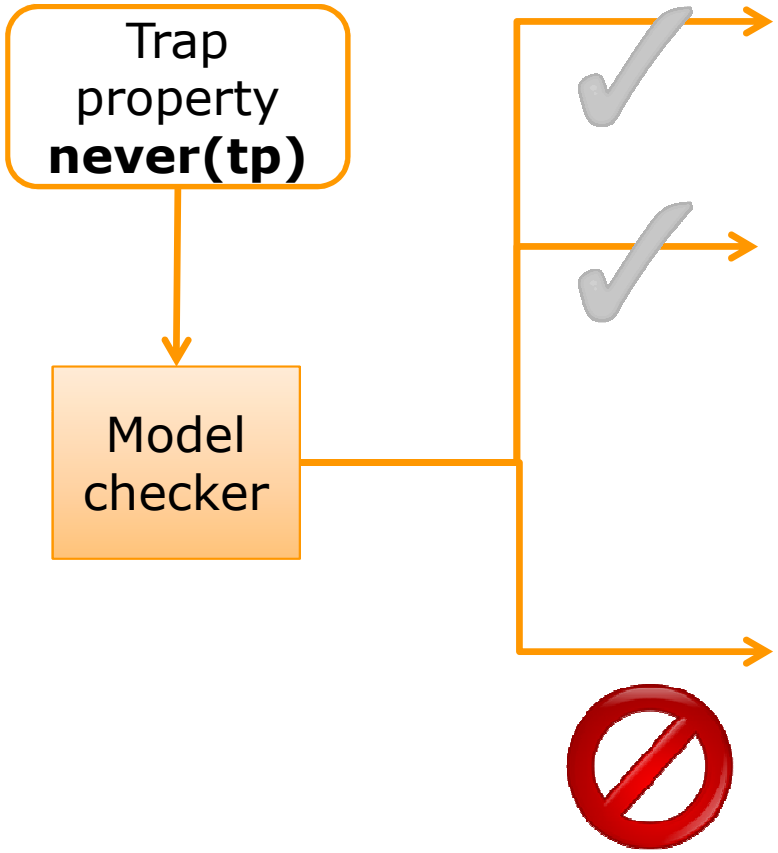
WP4: risultato 3

- Generazione di casi di test mediante strumenti usati normalmente per la verifica
- Motivazione
  - Conformance testing delle implementazioni
  - Generazione automatica dei casi di test con oracoli
  - Integrazione “tests and proofs”
    - in particolare quelli sviluppati in WP2
    - model checking/ SMT/SAT solvers
  - Alcuni problemi aperti (mix risultati e lavoro futuro)

Mc for test generation



Some limits



1. Trap property proved false
   - Cex found
2. Trap property proved true
   - Test predicate infeasible
3. MC does not complete the proof
   - Out of memory (state explosion problem)
   - Out of time (timeout reached)
   - Wrong abstractions used by MC

Open issues

- Quale model checking è il migliore?
  - Fraser e Gargantini, *Evaluation of Model Checkers for Test Case Generation* in IEEE International Conference on Software Testing Verification and Validation (ICST) (2009)
- C'è un ordine migliore in cui prendere i test predicates?
  - Fraser, Gargantini e Wotawa, *On the Order of Test Goals in Specification-Based Testing,* in Journal of Logic and Algebraic Programming, vol. 78, n. 6 (2009)
- La lunghezza dei casi di test ottenuti?
  - Fraser e Gargantini, *Experiments on the Test Case Length in Specification Based Test Case Generation* in Fourth International Workshop on the Automation of Software Test (AST09) - ICSE (2009)

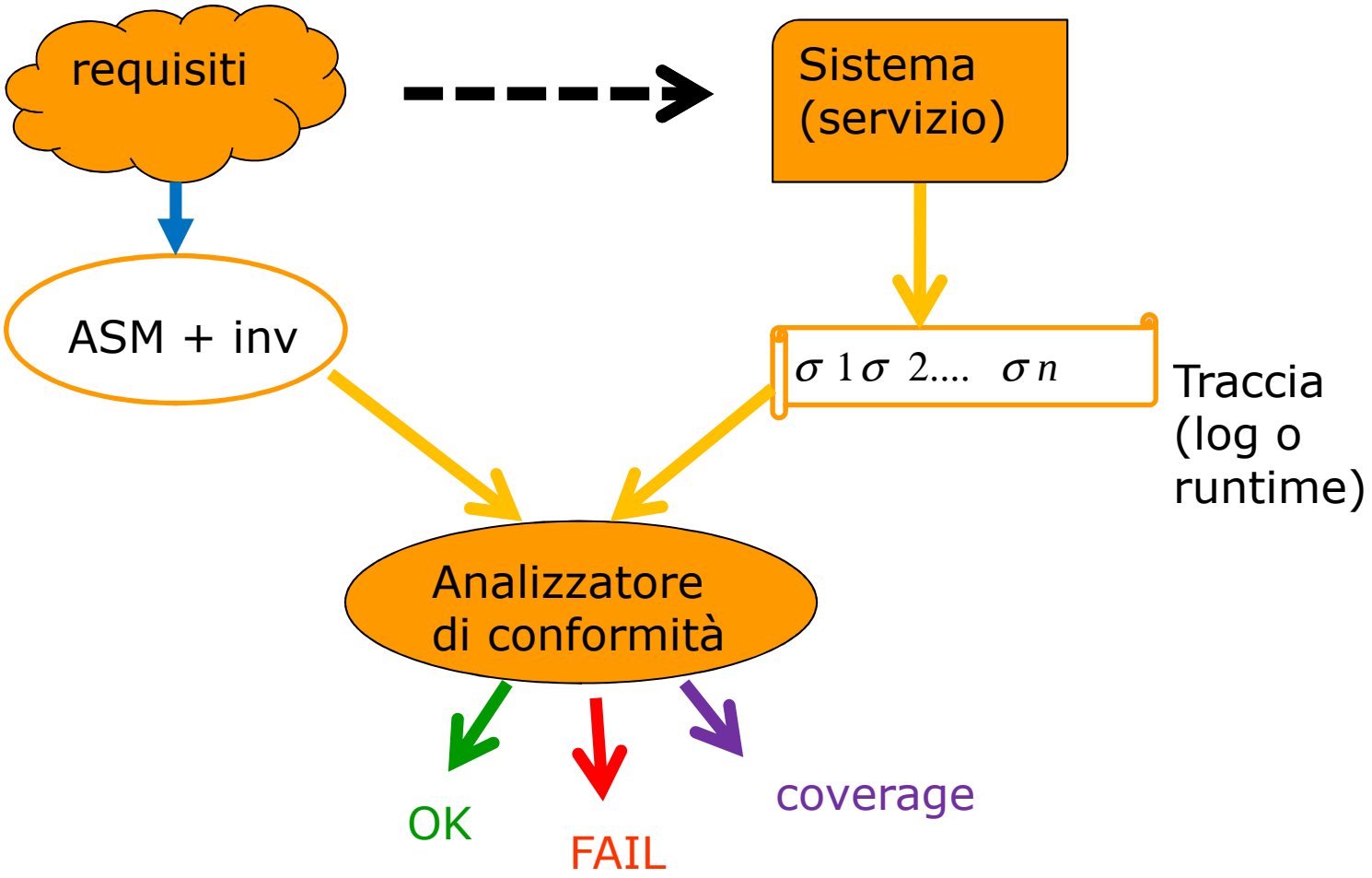
Open issuses (2)

- Quali testing criteria usare?
  - TAICPART 09
- Si può applicare al combinatorial testing? Come gestire i **constraints** sugli input?
  - Andrea Calvagna and Angelo Gargantini
    Combining Satisfiability Solving and Heuristics to Constrained Combinatorial Interaction Testing
    in Third International Conference on Tests And Proofs (TAP) Springer-Verlag (2009): 27–42
- Si può applicare al fault based testing per specifiche Booleane?
  - A-MOST

WP2/4: obiettivi futuri

- Runtime monitoring mediante le ASM
  - monitoring di servizi
    - Bianculli, Ghezzi, Guinea, Baresi, …
  - Models @ runtime
- Stato dell'arte
  - Uso di logiche temporali (TL) per modellare le proprietà da monitorare
    - Ad esempio LTL
  - **Idea:** sostituire la logica temporale con le ASM
    - Il designer scrive la specifica (parziale) ASM M del sistema da monitorare e un monitor controlla la conformità del sistema reale rispetto a M
    - L'ASM contiene INVARIANTI

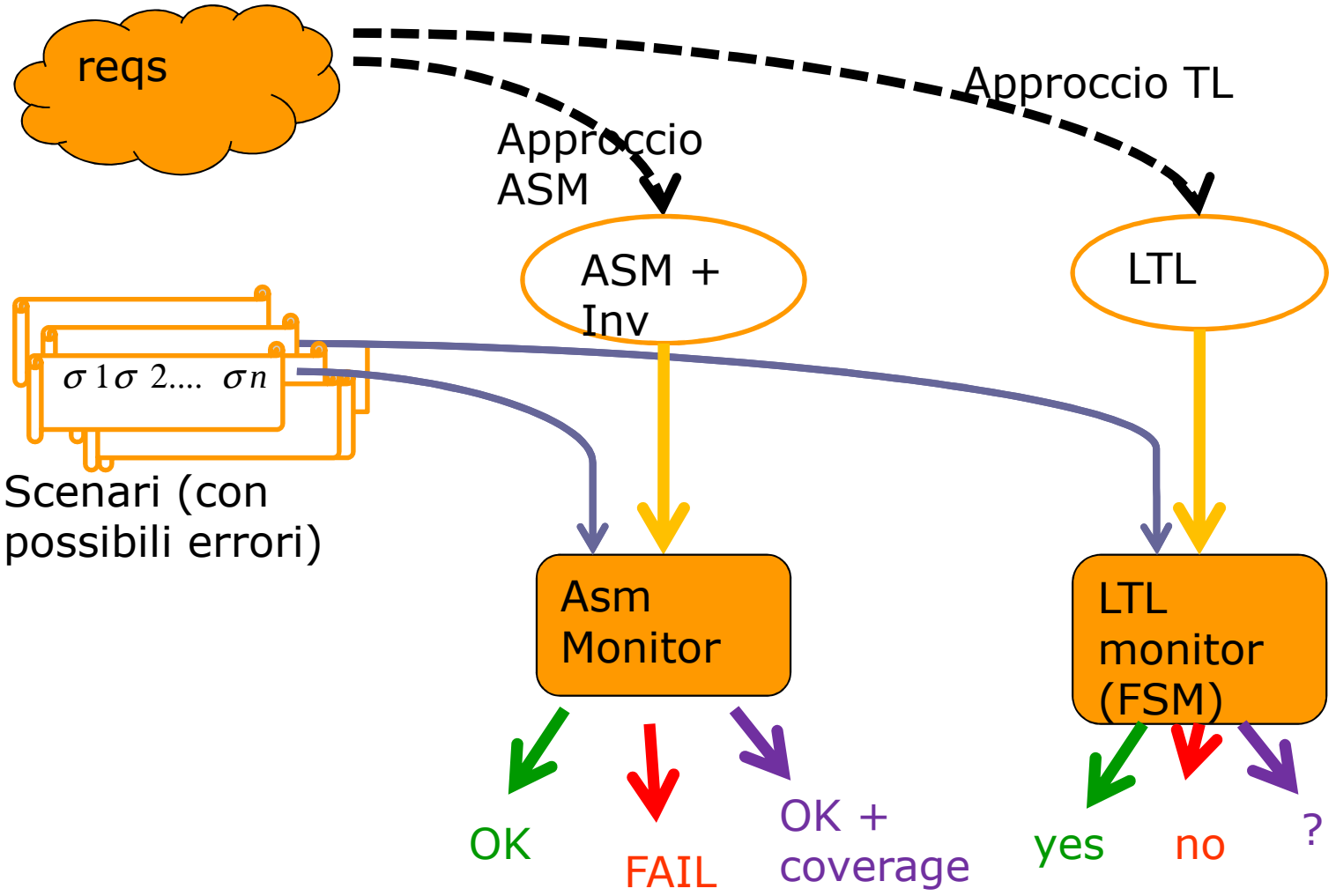
ASM for monitoring



Questioni aperte rispetto TL

- Fattibilità
- **Usabilità** E' più facile scrivere una ASM o una formula TL?
  - Operazionale vs dichiarativo
    - mix: ASM + invarianti
  - Lo stato delle ASM può essere utile per scrivere formule più semplici?
- **Espressività** Tutto quello che si può scrivere con TL si può anche con ASM?
  - Esempio: *eventually(a)*
- **Efficacia** E' ugualmente efficace?
  - Riesce a catturare gli stessi comportamenti scorretti?

Valutazione efficacia DOE



Stesso risultato?

Fine