A Workflow Checking Approach for Inherent Privacy Awareness in Network Monitoring

Maria N. Koukovini

Eugenia I. Papagiannakopoulou

Georgios V. Lioudakis

Dimitra I. Kaklamani

Iakovos S. Venieris

The 6th International Workshop on Data Privacy Management

(DPM-2011)

Leuven, Belgium, September 15 – 16, 2011

Passive Network Monitoring

- Inspection of the actual network traffic using special software and/or hardware equipment
- Range of applications:
  - Operation and management of communication networks
  - Identification of performance bottlenecks
  - Network security (IDS, ADS, …)
  - Network planning
  - Accounting and billing of network services
  - Validation of SLAs
  - Observation and fine-tuning of QoS parameters
  - Internet research based on collected traffic traces
  - Law enforcement (data retention, lawful interception, …)

Passive Network Monitoring

- Serious drawback: privacy implications!
  - Relies natively on personal data collection and processing
  - Various documented privacy violation mishaps
- Passive Network Monitoring special characteristics:
  - Privacy-sensitive information exceeds payload and spans across various protocol headers and other communication metadata
  - Too much personal information can be inferred and extracted using advanced processing techniques (statistical analysis, fingerprinting, …)
  - Specific regulations govern the underlying services and data
  - Very high data rates and consequent performance requirements
  - Distributed and cooperative nature of operations and infrastructures
    - Intra-domain
    - Inter-domain

Privacy-Preserving Network Monitoring: Regulatory Requirements

- Lawfulness of data processing
- Purposes for which data are processed
- Necessity, adequacy and proportionality of the data processed
- Quality of the data processed
- Minimal use of personal identification data
- Storage of personal data
- Data retention
- Access limitation
- Information to and rights of the data subject
- Consent of the data subject
- Data security measures
- Special categories of data
- Coordination with competent data protection Authority
- Supervision and sanctions
- Communications confidentiality and lawful interception
- Flexibility and adaptability of legal compliance provisions

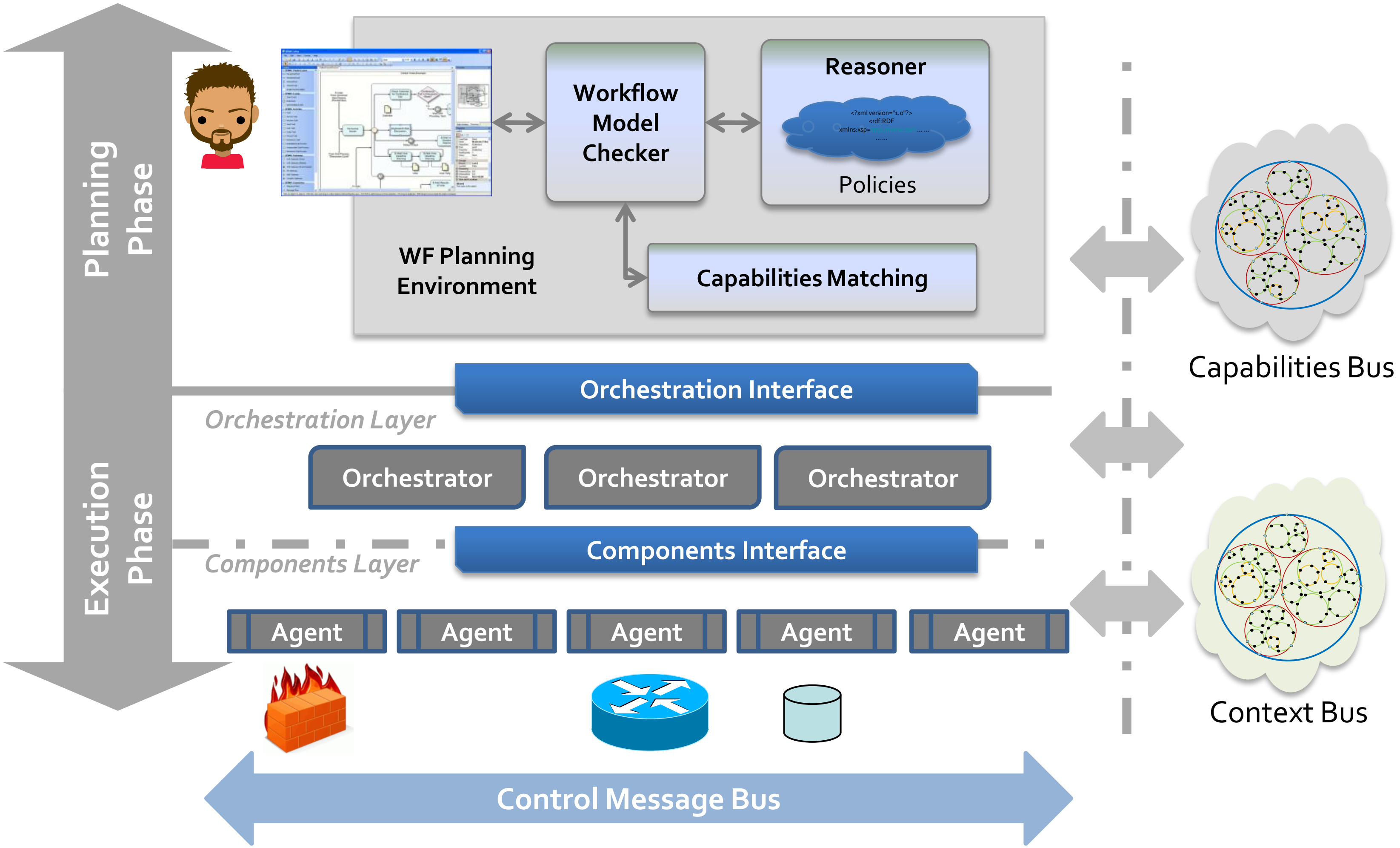
Fundamental Principles of the Approach

Realisation of *Privacy by Design*

⇨ Privacy-aware information flows

- Enforcement of privacy-aware access control across the flows
- Contextual behaviour of the system
- Automatic integration of protection means
  - Anonymisation, pseudonymisation, aggregation modules
  - Complementary actions
- Consideration of the semantics of various concepts, such as:
  - Data
  - Roles
  - Operational processes
  - Purposes for data collection and processing

Architecture Overview



Planning Phase

Execution Phase

Workflow Model Checker

Reasoner

<?xml version="1.0"?>
<rdf:RDF
xmlns:xsp=http://www.owl-... ...
... ...

Policies

WF Planning Environment

Capabilities Matching

Capabilities Bus

Orchestration Interface

Orchestration Layer

Orchestrator

Orchestrator

Orchestrator

Components Interface

Components Layer

Agent

Agent

Agent

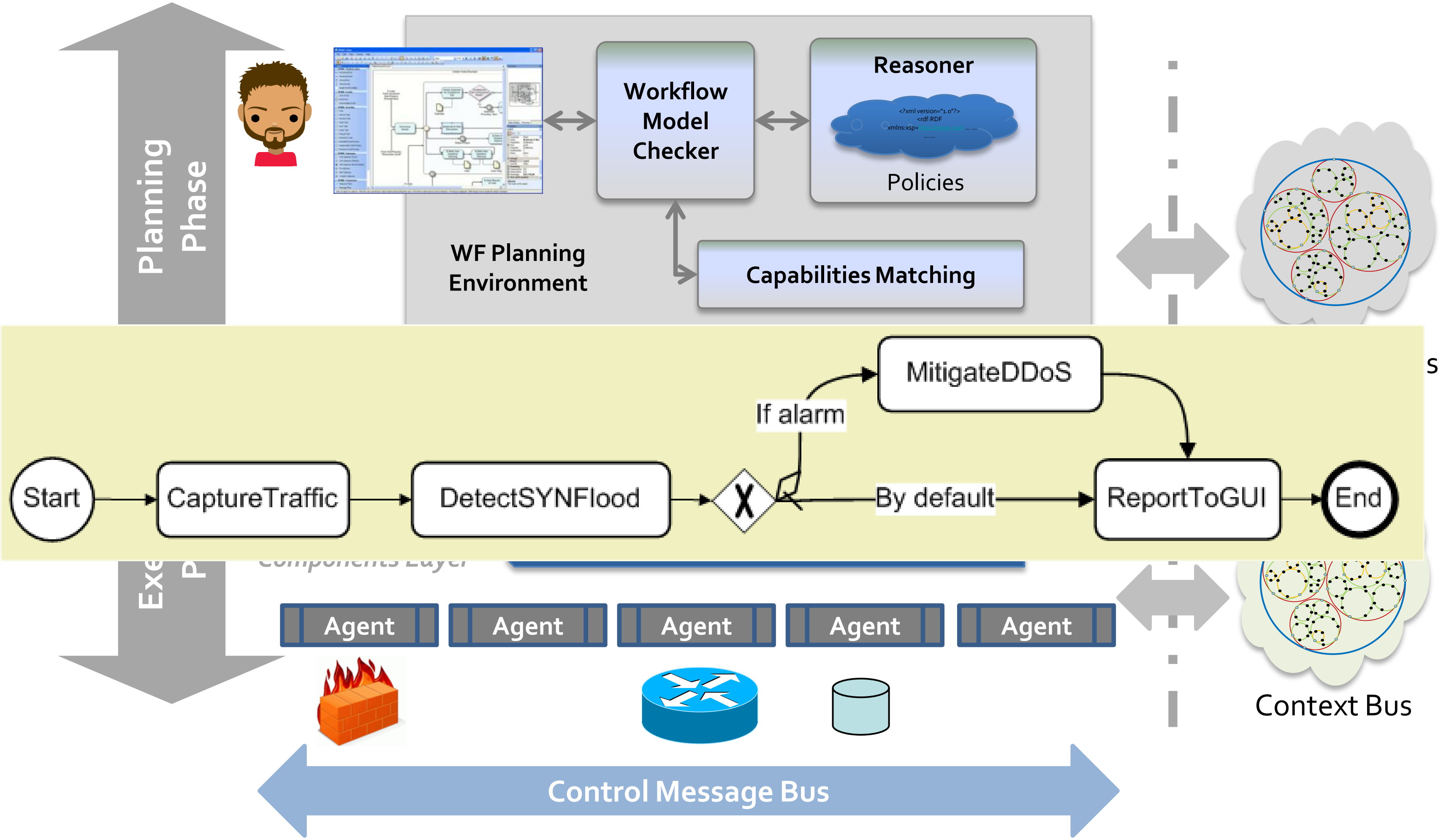
Agent

Agent

Context Bus

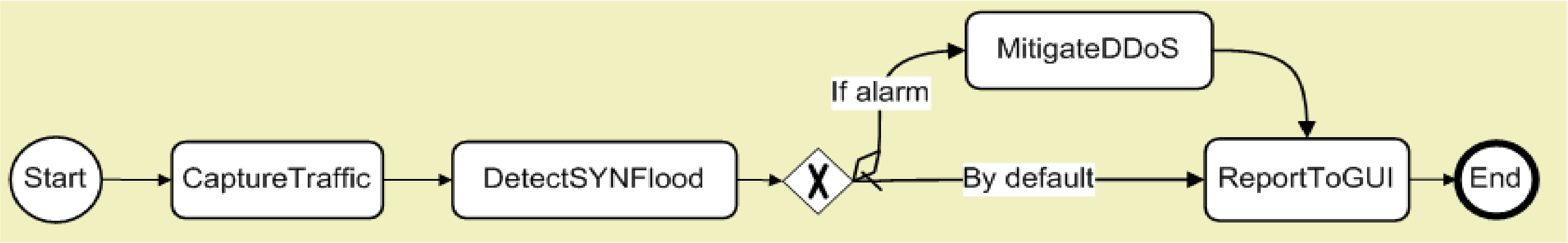
Control Message Bus

Architecture Overview



Workflows

- Workflows and other important parameters…
  - $w = \langle t_1, t_2, \ldots, t_n \rangle$, where $t_i = \langle a_i , op_i , res_i \rangle_w$
    - a_i : actor
    - op_i : operation
    - res_i : resource
  - + a declared purpose *pu*, e.g., **`NetworkSecurity`**
  - + User role(s) *r*, e.g., **`NetworkAdministrator`**
- Overall… $\langle w, \langle r \rangle^k, pu \rangle$
- or maybe…
  - $\langle w, \langle r \rangle^k, \langle pu \rangle^m \rangle$, stored workflow template



Workflow Verification Mechanism

- Ensures that the user-specified workflow is rendered privacy compliant before entering the execution phase
- A three steps procedure:
  1. Purpose Verification:
     Checks regarding purpose compliance (relevance, consistency, etc.)
  2. Skin Task Verification:
     User-specified tasks checked individually and in relation to each other
  3. Decomposition:
     Composite skin tasks’ refinement and evaluation, until the level of atomic tasks
- Relies on a policy-based access control model
- Core components: *Model Checker* and *Reasoner*

Step 1: Purpose Verification

- Based on two types of associations contained in / implied by the Policy Model:
  - *role-purpose*:
    not all roles can initiate a workflow serving a given purpose
    - **`NetworkAdministrator`** relevant to **`NetworkSecurity`**
    - **`Accountant`** not relevant to **`NetworkSecurity`**
  - *task-purpose*:
    not all tasks make sense to be used for serving a purpose
    - **`DetectSYNFlood`** is relevant with **`NetworkSecurity`**
    - **`InterceptCommunications`** has nothing to do with **`NetworkSecurity`**

Step 2: Skin Task Verification

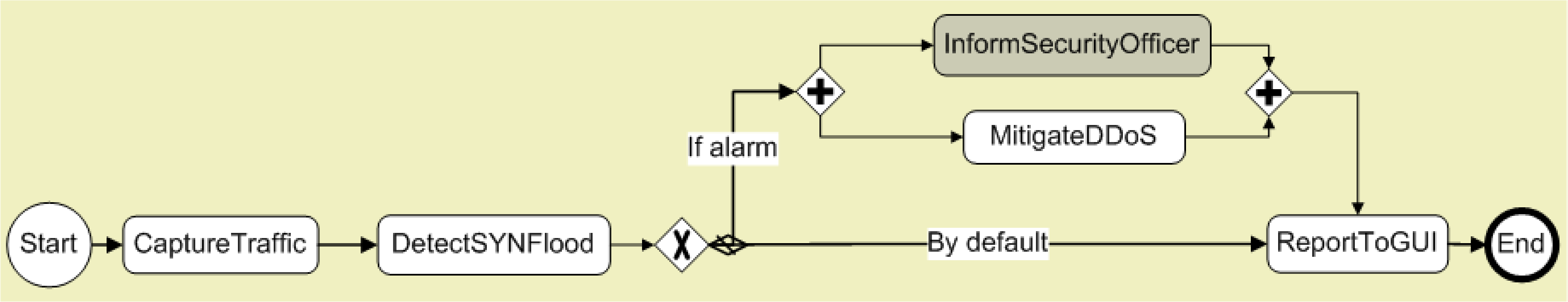
- Requirements checked:
  - The initiator must have the right to include the task in the workflow.
  - The task $\langle a_i, op_i, res_i \rangle_w$ must be valid, i.e., the actor a_i must have the right to perform the operation op_i on the resource res_i.
  - Each task must not conflict with precedent and subsequent tasks.
  - Potentially required complementary tasks must be present.
  - The system must be able "by definition" to offer the respective capability.
- Approach: for each skin task t_i of *w*, the Model Checker
  1. checks the task's availability by the system
  2. asks the Reasoner about task's acceptability

Step 2: Skin Task Verification

Possible results:

1. Unconditional acceptance, aka no changes are needed
2. Conditionally accept with task addition: ok, but some extra tasks are required
   Solution: required tasks addition

   e.g., **`MitigateDDoS`** requires **`InformSecurityOfficer`**



Step 2: Skin Task Verification

More possible results:

3. Conditionally accept provided some conflicts with other tasks are resolved
   Solution: task removal, substitution, task insertion
4. Conditionally accept, subject to contextual parameters
   Solution: conditional branching
   - Special case: ⟨actor, operation, resource⟩ inter-dependencies
   - Can be combined with all the above
5. Conditionally accept, subject to history-related conditions
   - Contextual constraints are *a priori* resolved by the flow itself, or
   - History creates additional contextual constraints

   Solution: conditional branching

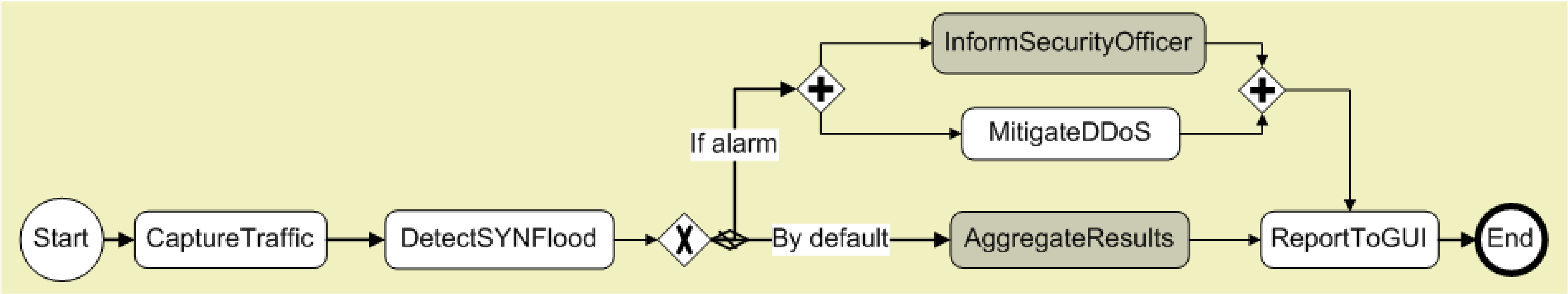
Step 2: Skin Task Verification

More possible results:

6. Task is not acceptable due to invalid $\langle a_i, op_i, res_i \rangle_w$ combination

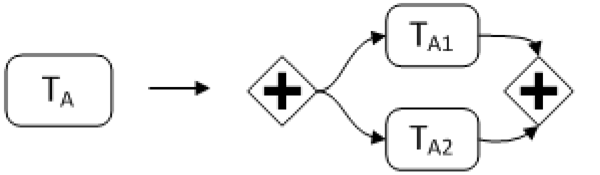
   Solution: task removal, substitution, task insertion

   - e.g., a role may require aggregated results, therefore, **`AggregateResults`** is inserted before **`ReportToGUI`**



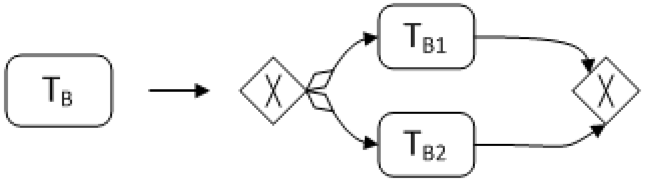
Step 3: Decomposition

- 3 types of decomposition
  - AND: all subtasks will be executed
    → all tasks must be acceptable

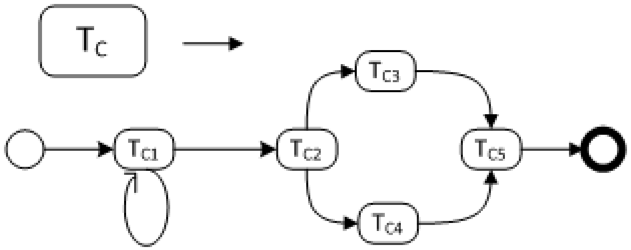


  - XOR: exactly one subtask will be executed, depending on:
    - Context
    - Capabilities availability
    - Prioritisation
    - Flow constraints

    → at least one task must be acceptable



  - Subworkflow: worklet implementation
    → all subtasks must be acceptable



Step 3: Decomposition

- Approach:
  For each skin task t_i of w, the Model Checker asks the Reasoner for a decomposition
  - Input: $\langle\langle a_i, op_i, res_i\rangle_w, r, pu\rangle$
  - Output: a decomposition that
    - is valid as a standalone structure, but
    - there may be constraints
- Possibly many levels of decomposition
  - Iterative procedure
  - Combined depth-first/ breadth-first verification
- If there is no valid decomposition (conflicts, other parameters), the parent task is rejected

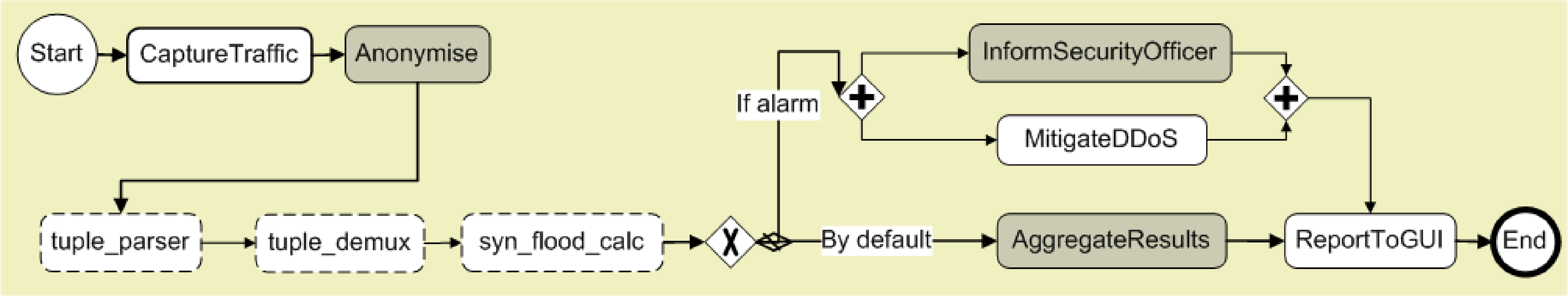
Decomposition Constraints

- Contextual constraints:
  - The aggregated contextual constraints of its subtasks
  - XOR: each subtask applicable under a different context
- Complementary required tasks:
  - The aggregated subtasks' requirements
  - XOR: each subtask requires different complementary tasks

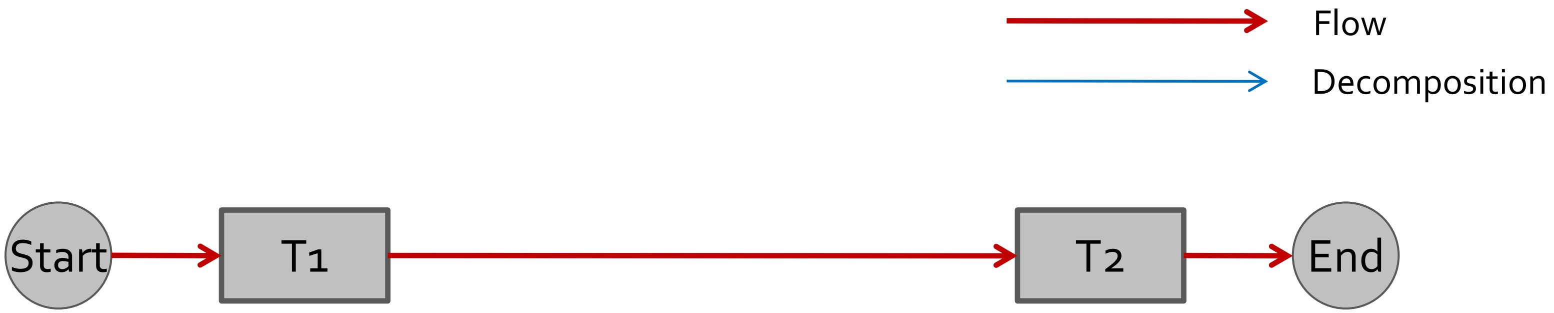
Decomposition Constraints

- Conflicts:
  - AND / Subworkflow: no subtask must conflict with other workflow tasks
  - XOR: at least one subtask must not conflict with other workflow tasks
  - Conflict resolution: removal, addition, substitution

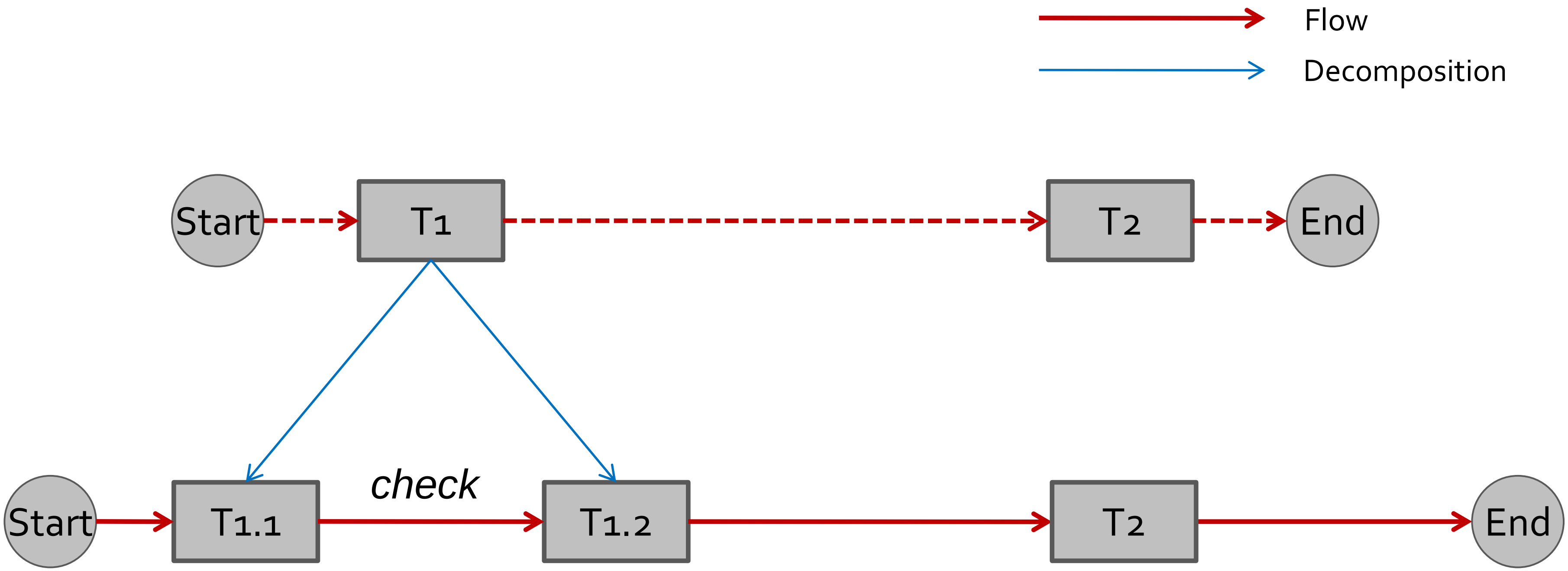
e.g., **`CaptureTraffic`** conflicts with **`tuple_parser`**
→ **`Anonymise`** task is inserted for conflict resolution



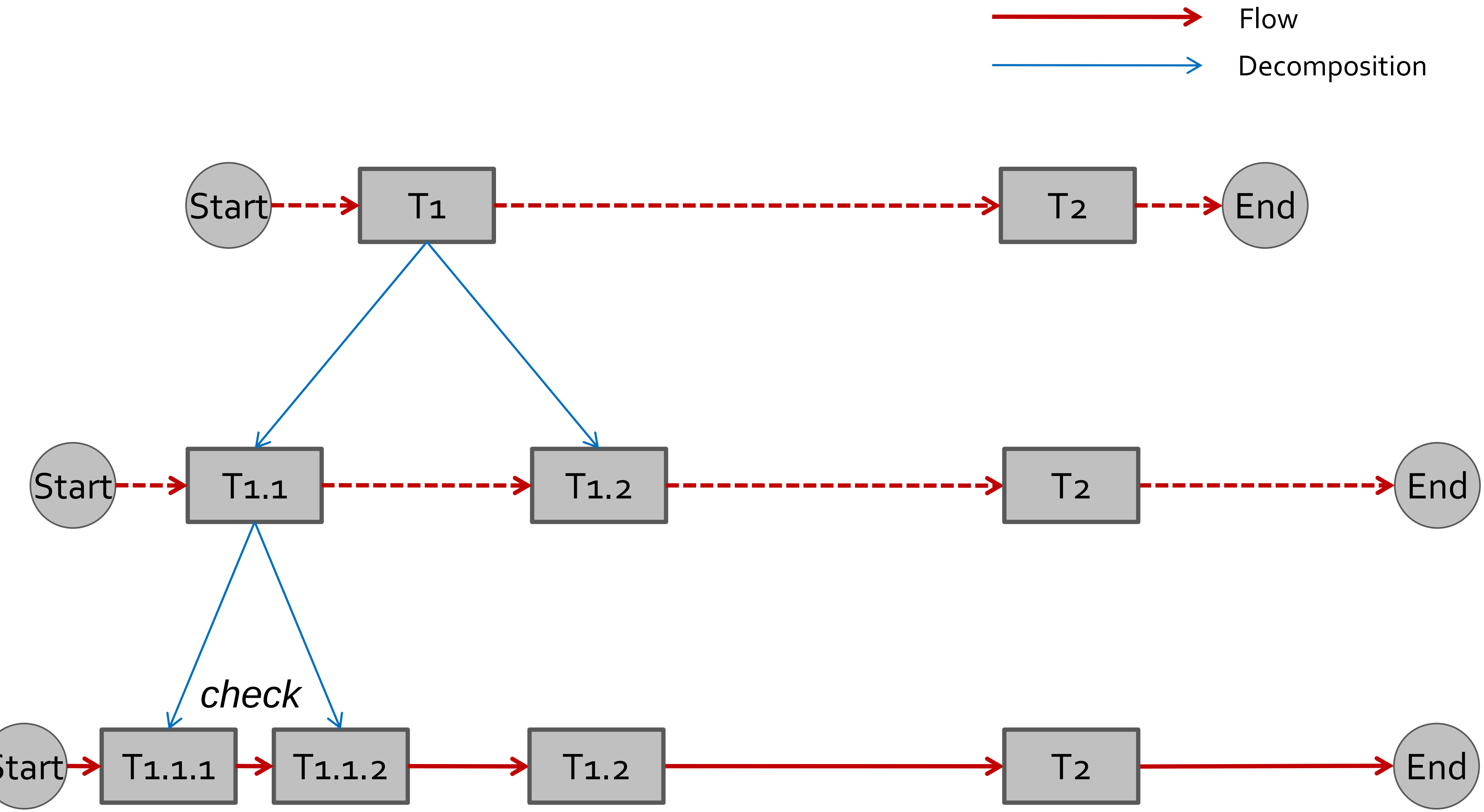
Decomposition Procedure Example



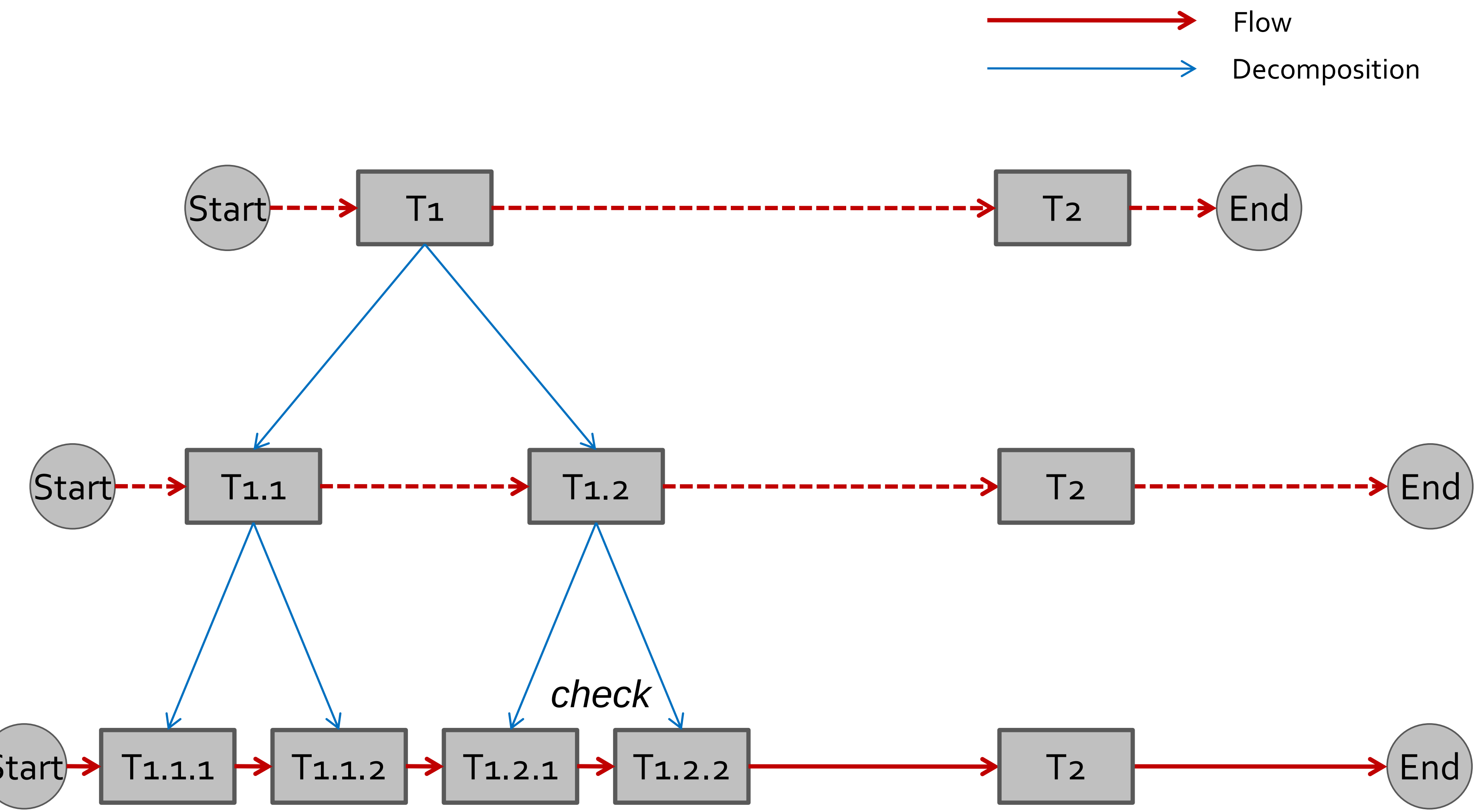
Decomposition Procedure Example



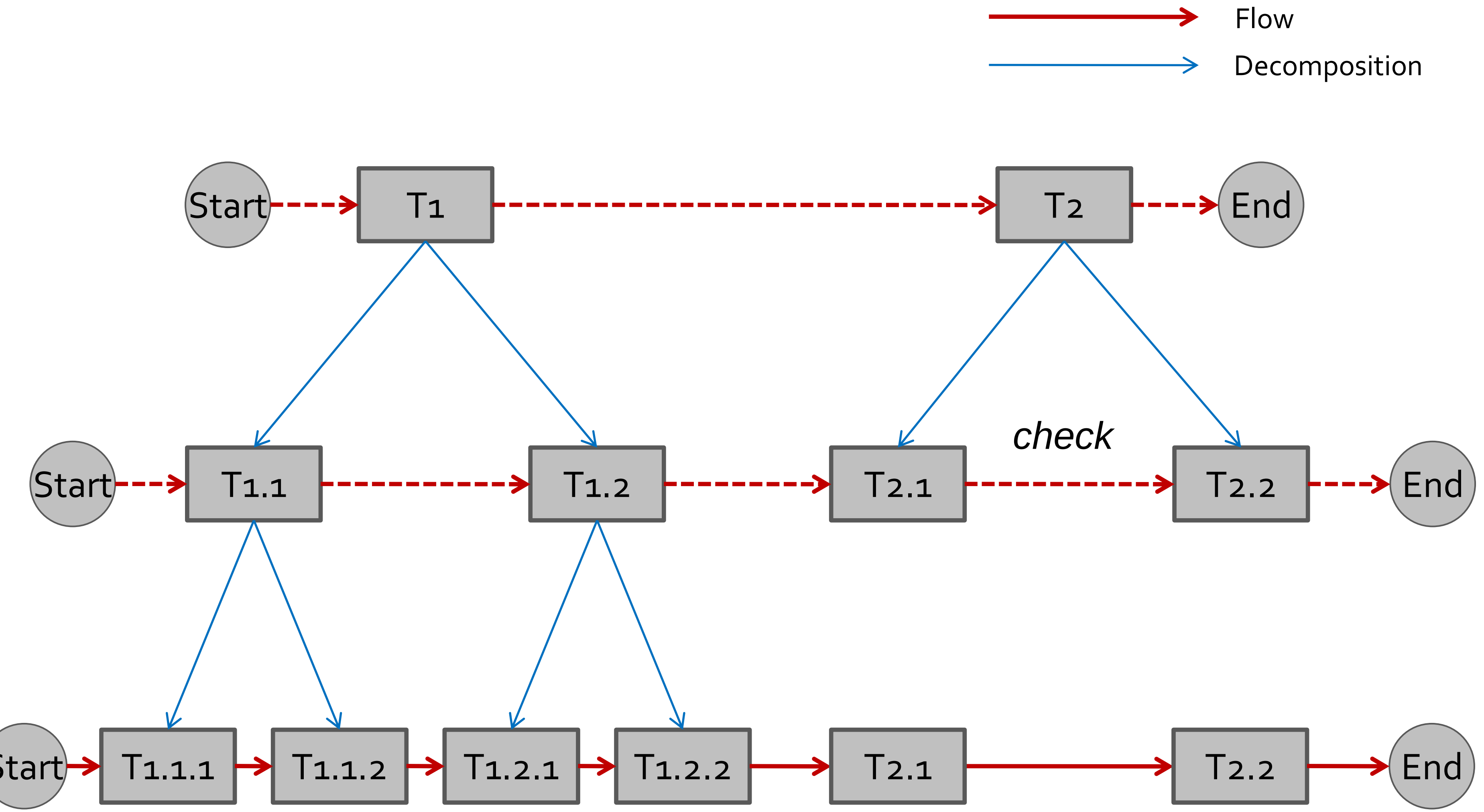
Decomposition Procedure Example



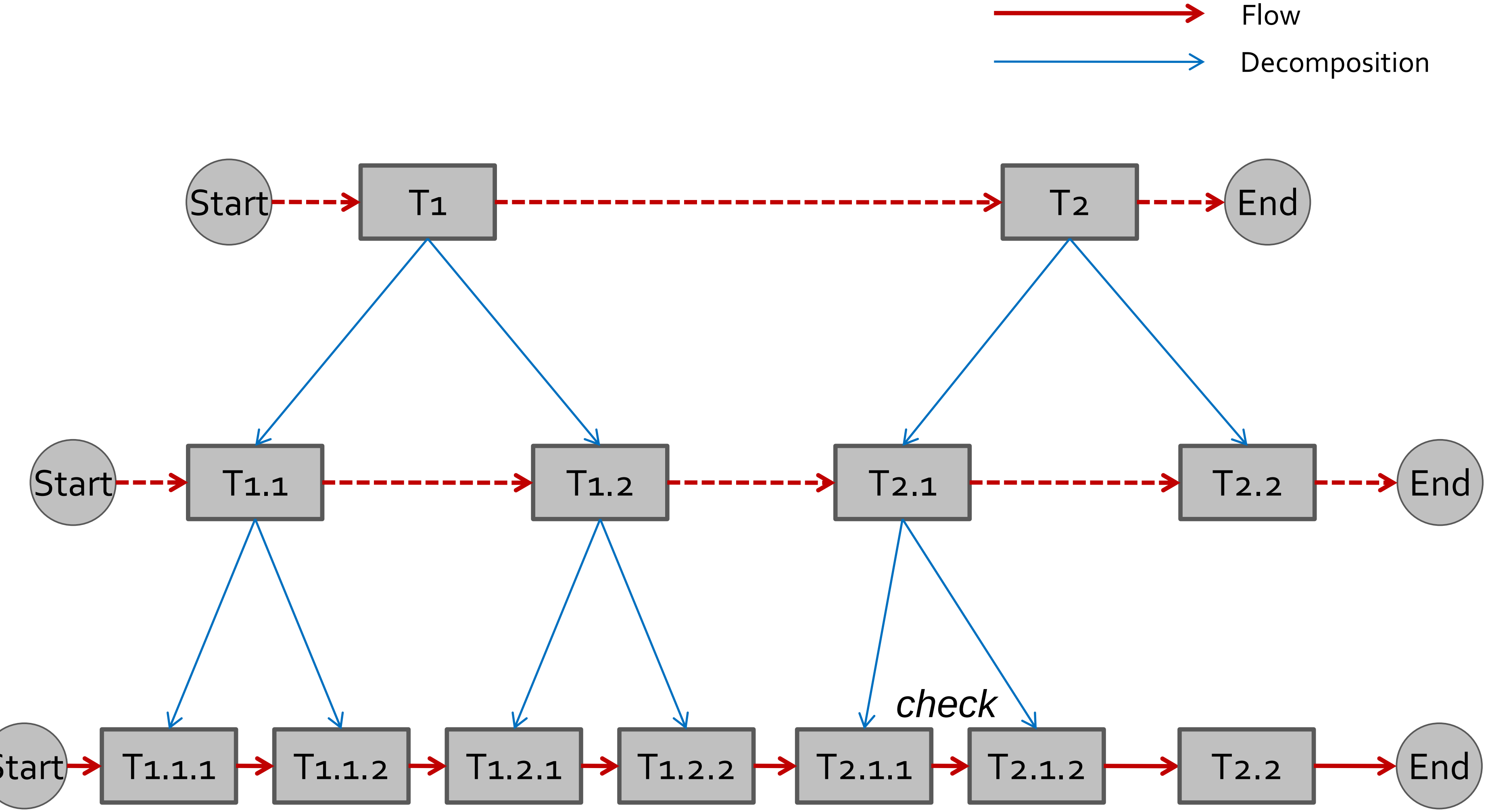
Decomposition Procedure Example



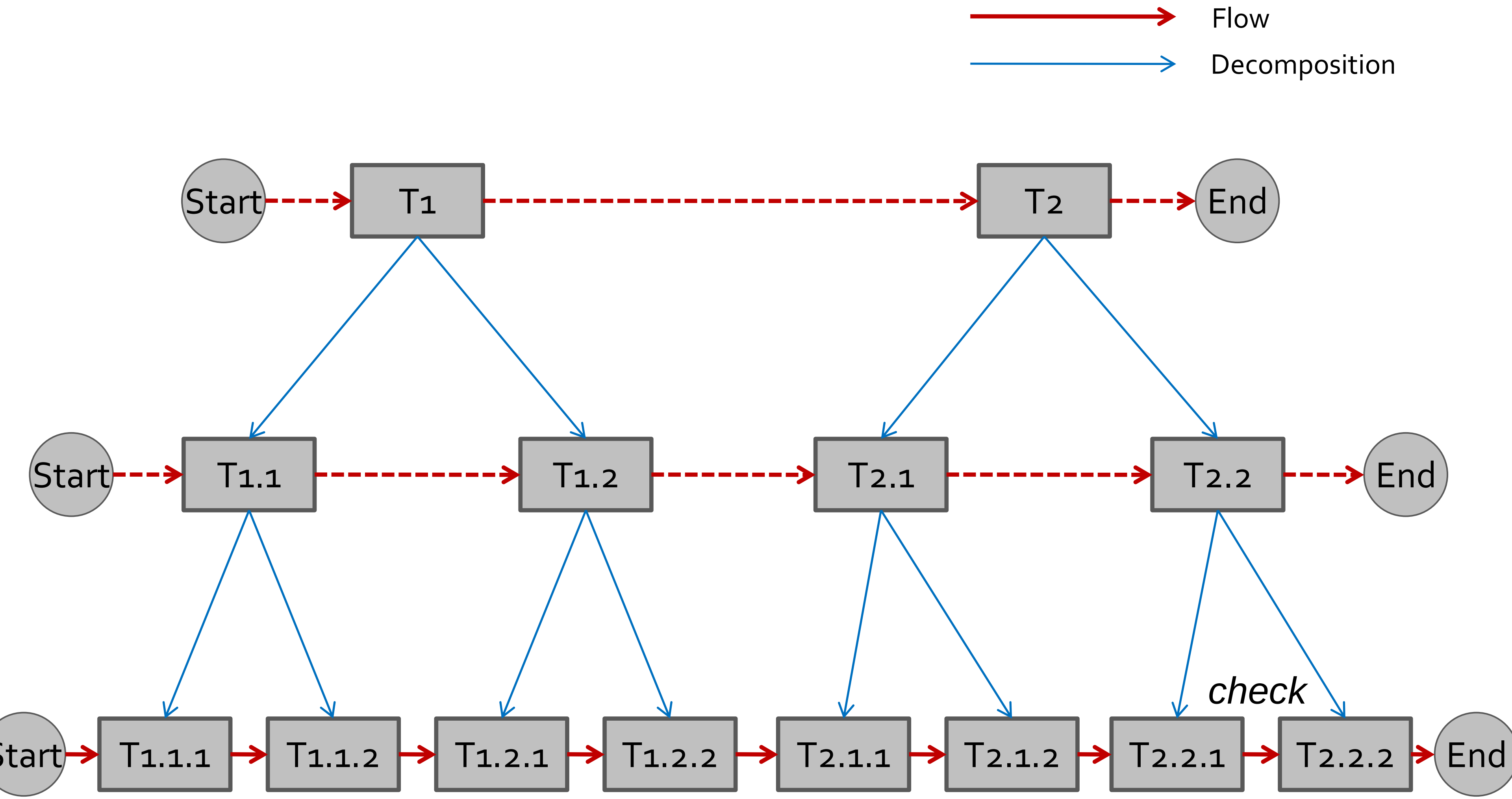
Decomposition Procedure Example



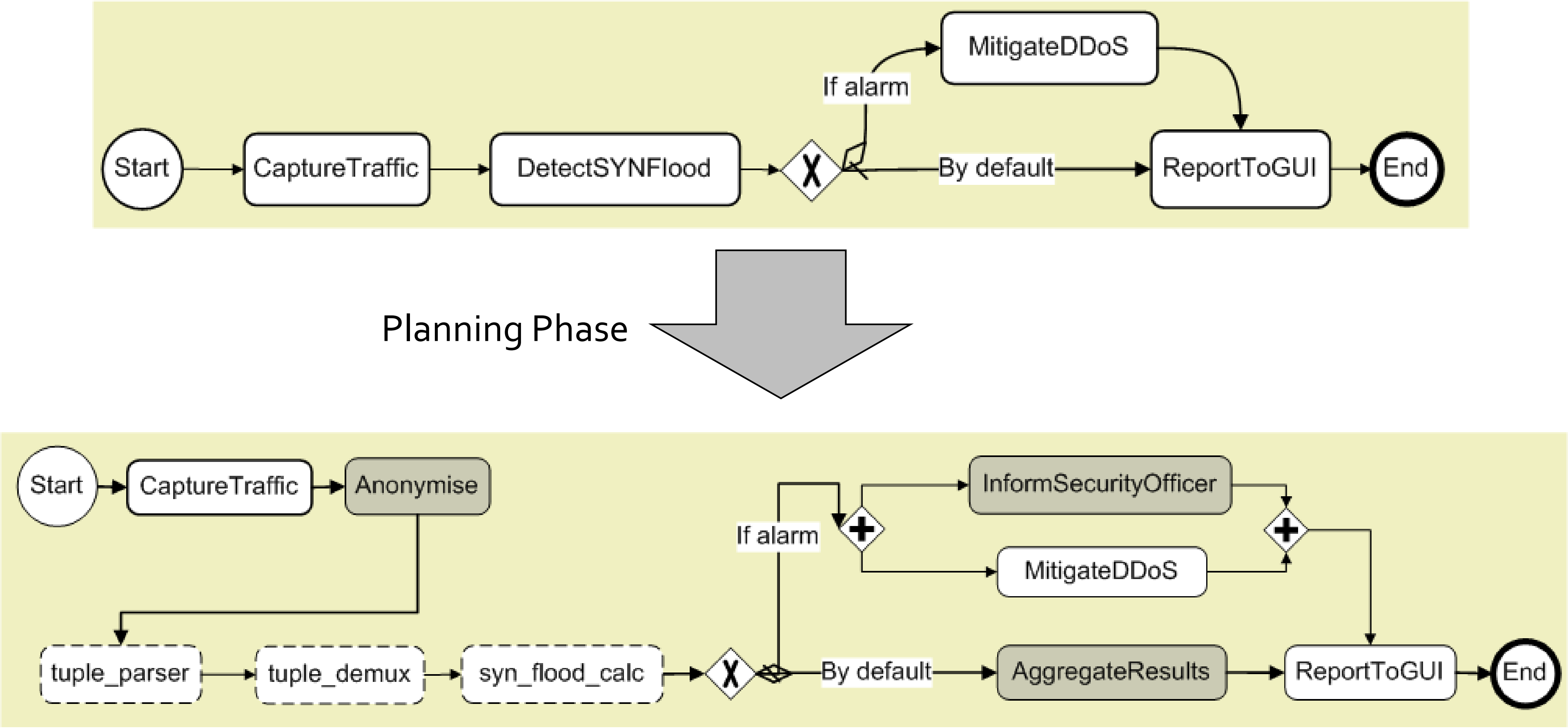
Decomposition Procedure Example



Decomposition Procedure Example



Planning Phase Summary



- Ready for the Execution Phase…

Current and Future Work

- Finalisation of prototype development
- Sophisticated approach for purpose verification
  - Fuzzy relations between purpose – role – operation
- Functionality vs. practicality trade-offs management
  - Evaluations’ complexity may result in impractical system
  - Certain aspects can be addressed offline
- Additional concepts under definition
  - Workflow “skeletons”
  - Workflow “paths”
  - Transformation and execution patterns
- Dynamic workflow adjustment based on real-time constraints
  - Availability of capabilities
  - Unexpected contextual changes
- Delegation of execution – actor “mobility”
- Inter-domain issues: negotiation of policies, semantic interoperation

- For more information:

  mariza@icbnet.ntua.gr

  http://www.fp7-demons.eu/

Thank you for your attention!

Any questions?