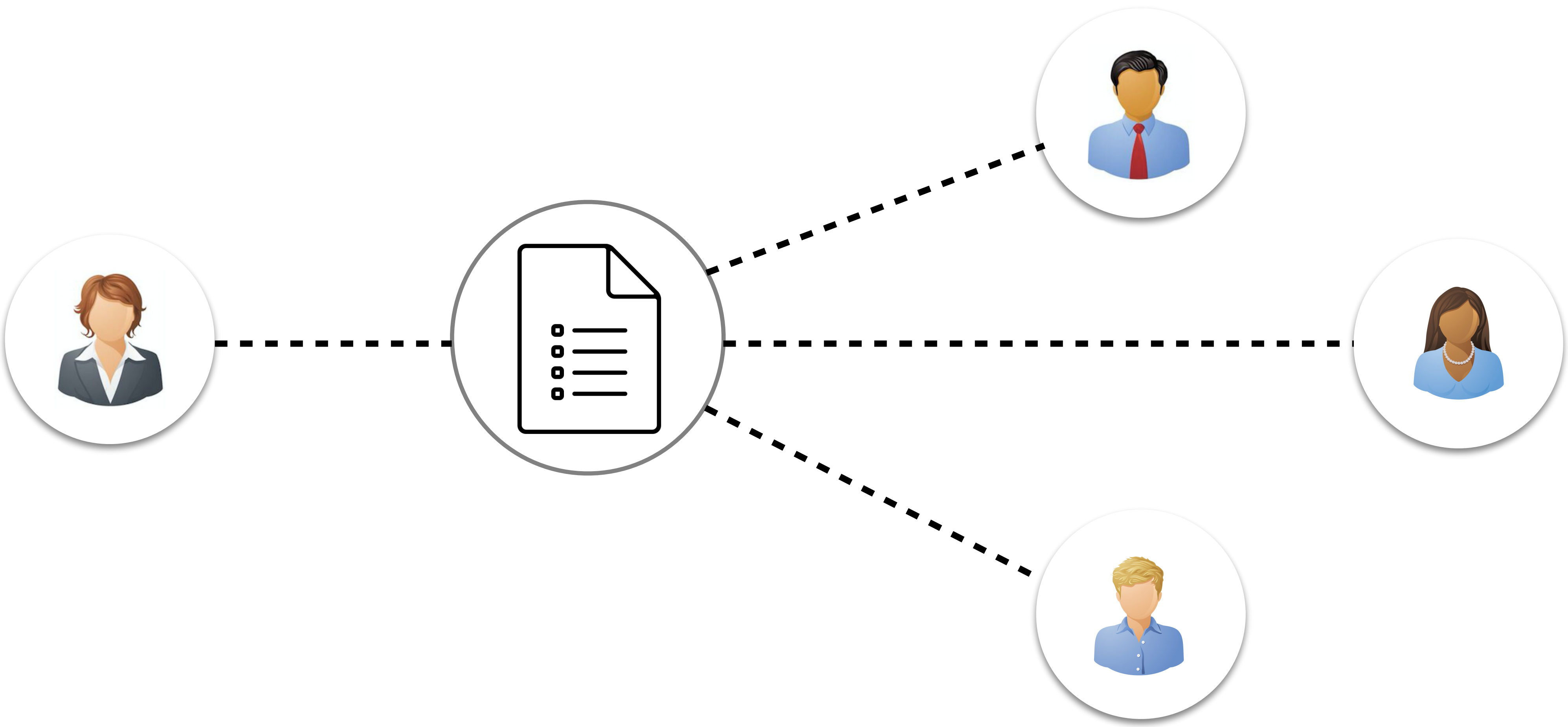
UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

Erhan Leblebici, Anthony Anjorin, Andy Schürr

ON COMBINING TRIPLE GRAPH GRAMMARS AND LINEAR OPTIMISATION TECHNIQUES

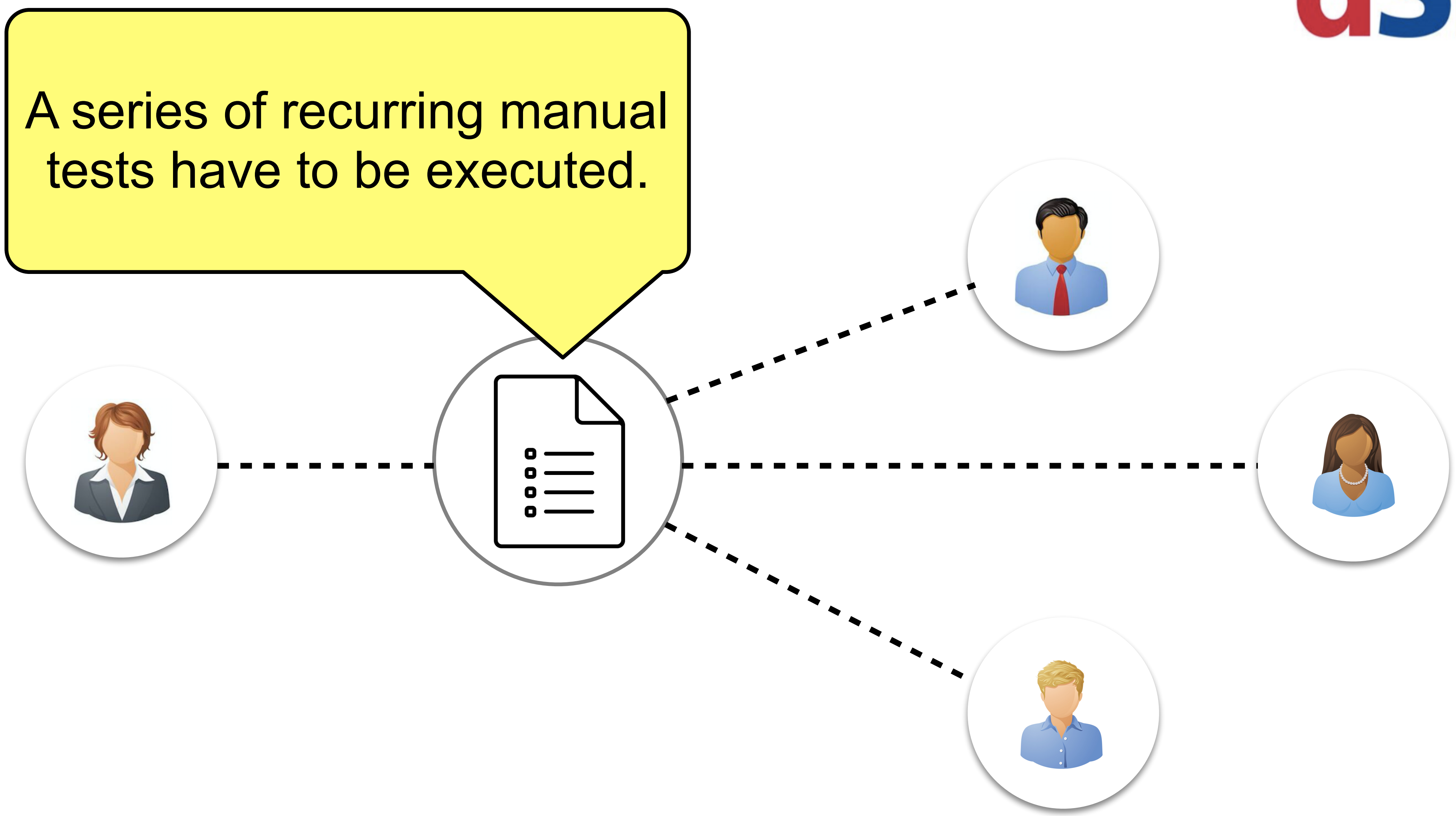
A Real-World Example: Overview

dSPACE



Robin Oppermann:
A Configurable, Model-Driven Approach to Optimal Scheduling using Triple Graph Grammars and Linear Progamming.
Ongoing Master's Thesis, Paderborn University in collaboration with dSpace

A Real-World Example: Overview

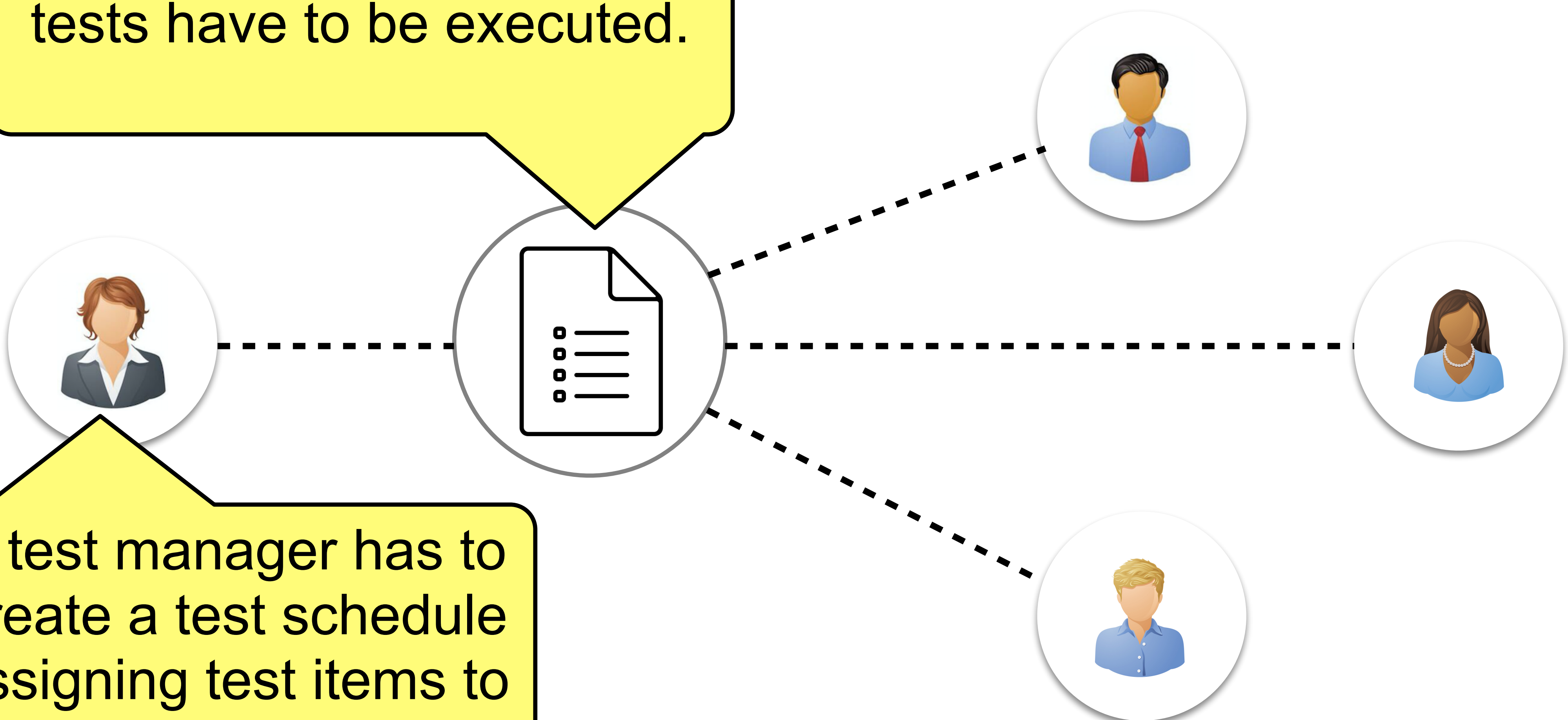


Robin Oppermann:
A Configurable, Model-Driven Approach to Optimal Scheduling using Triple Graph Grammars and Linear Progamming.
Ongoing Master's Thesis, Paderborn University in collaboration with dSpace

A Real-World Example: Overview

dSPACE

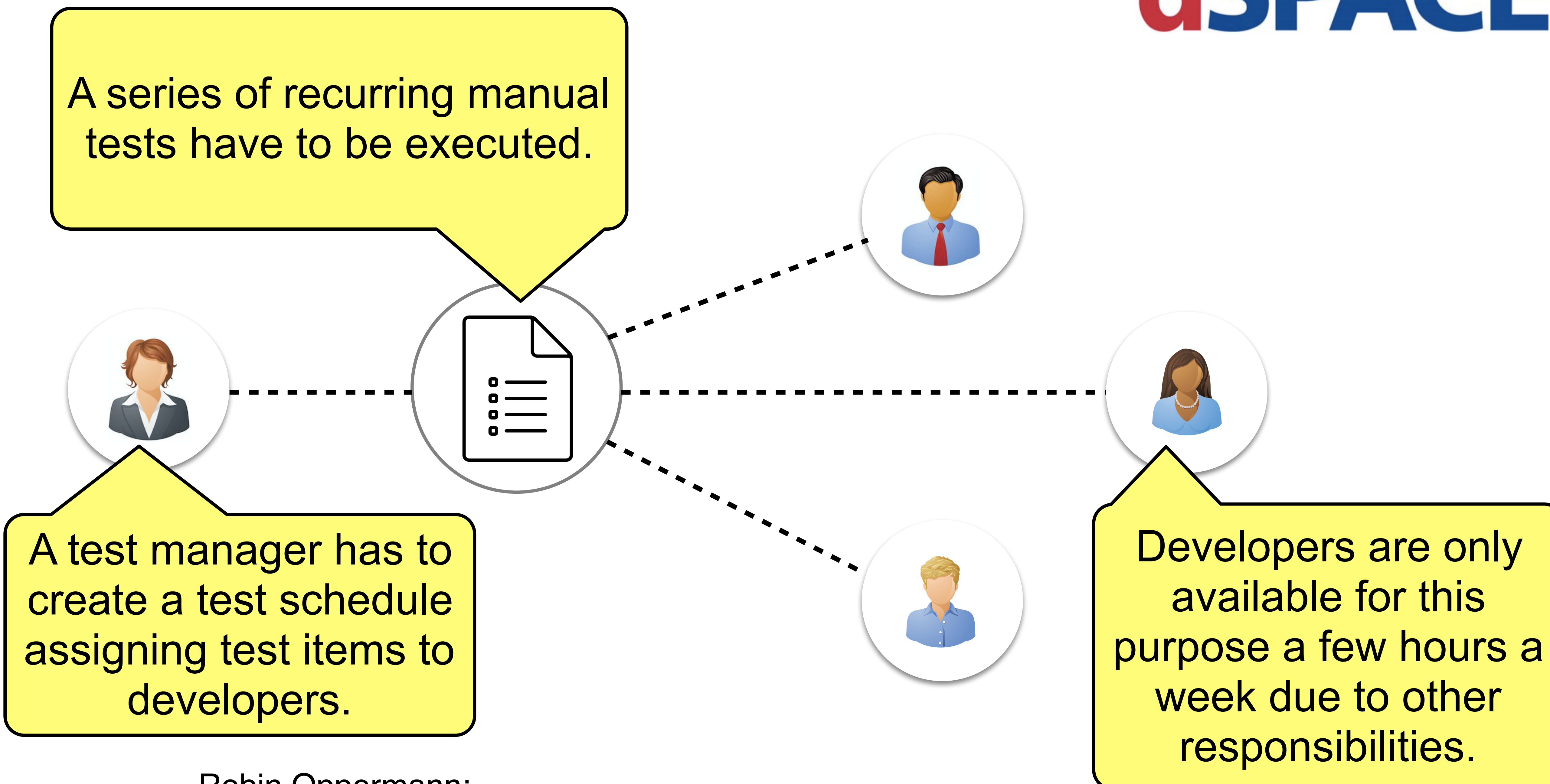
A series of recurring manual tests have to be executed.



A test manager has to create a test schedule assigning test items to developers.

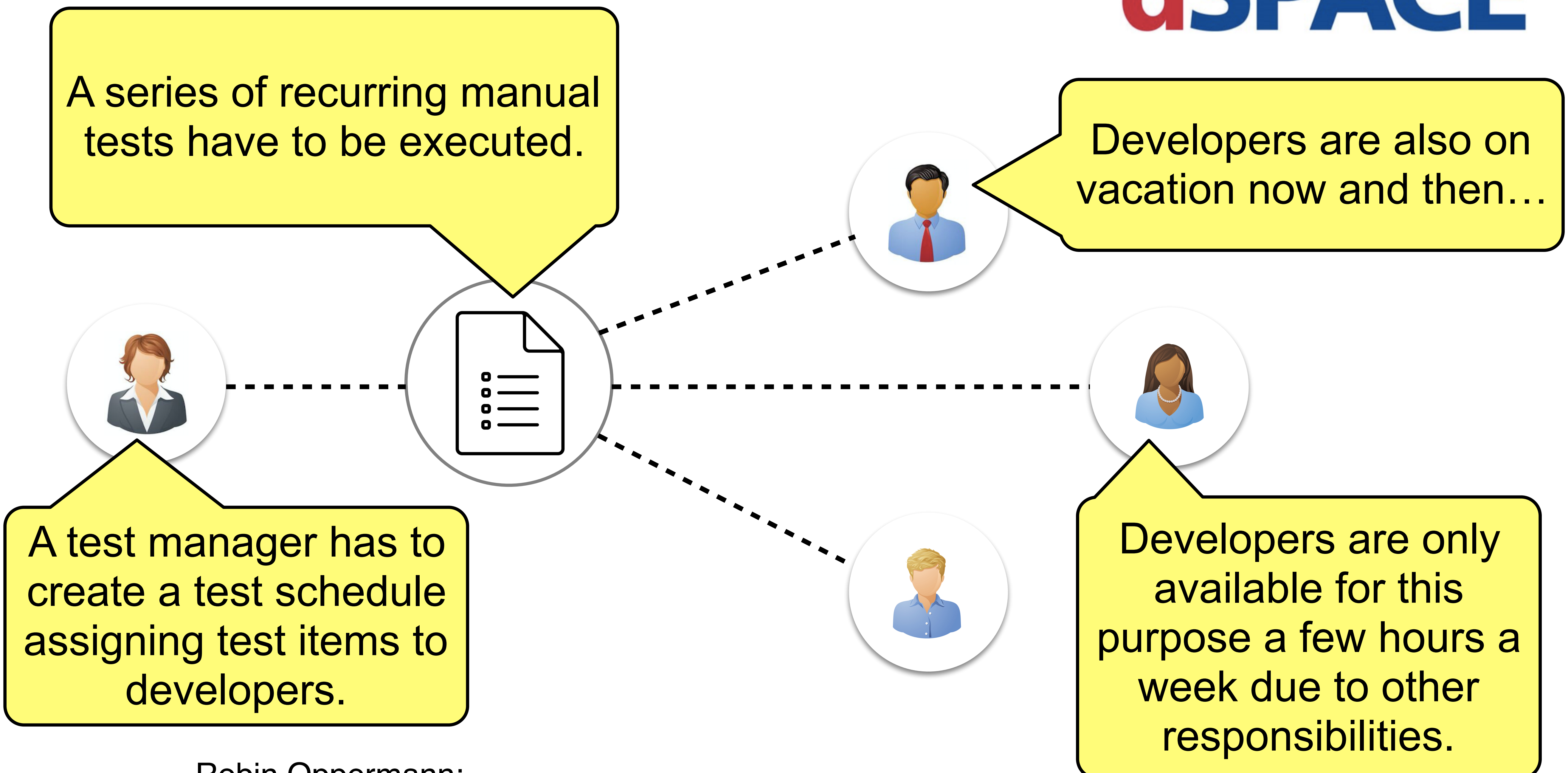
Robin Oppermann:
A Configurable, Model-Driven Approach to Optimal Scheduling using Triple Graph Grammars and Linear Progamming.
Ongoing Master's Thesis, Paderborn University in collaboration with dSpace

A Real-World Example: Overview



Robin Oppermann:
A Configurable, Model-Driven Approach to Optimal Scheduling using Triple Graph Grammars and Linear Progamming.
Ongoing Master's Thesis, Paderborn University in collaboration with dSpace

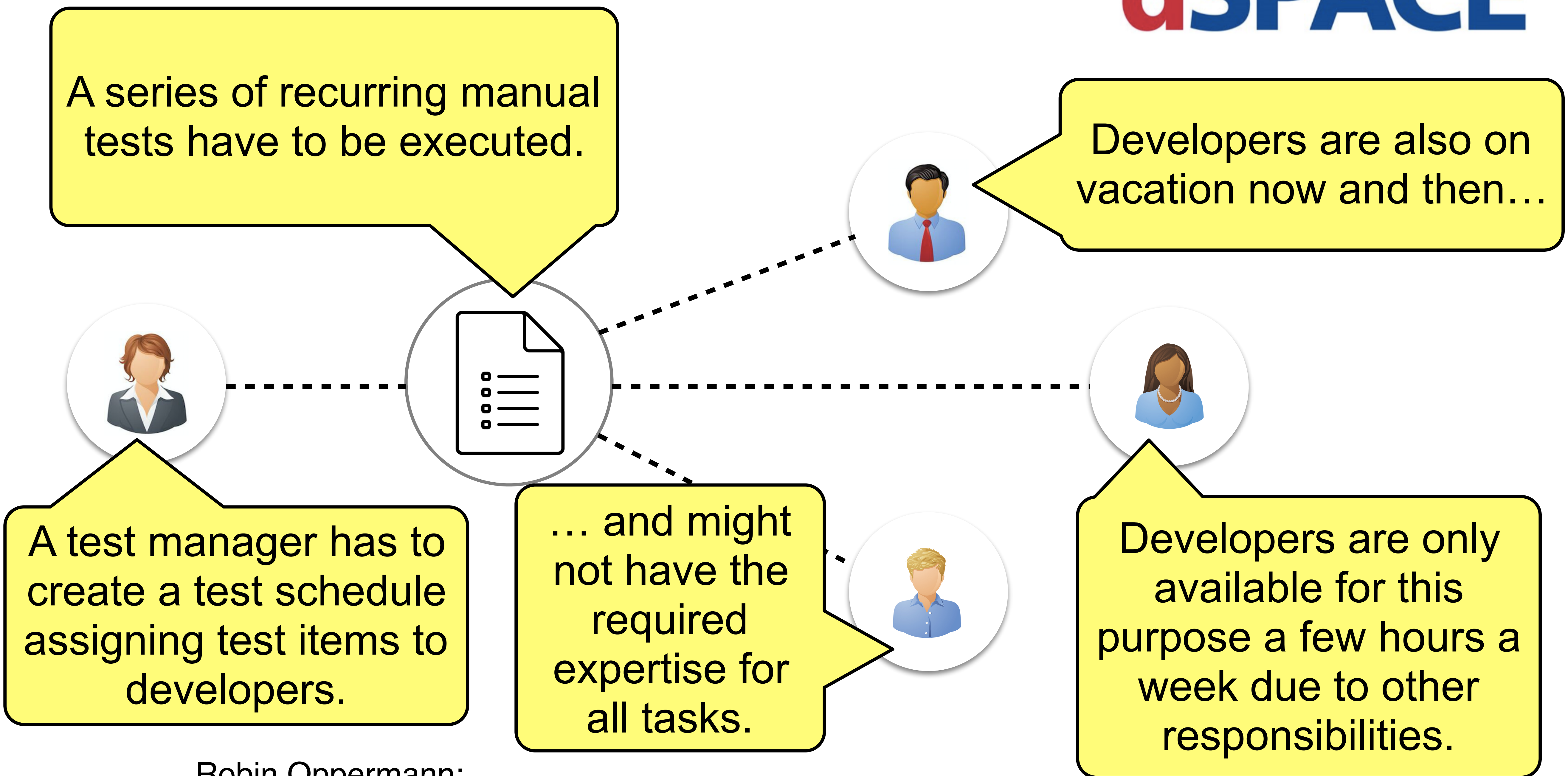
A Real-World Example: Overview



Robin Oppermann:
A Configurable, Model-Driven Approach to Optimal Scheduling using Triple Graph Grammars and Linear Progamming.
Ongoing Master’s Thesis, Paderborn University in collaboration with dSpace

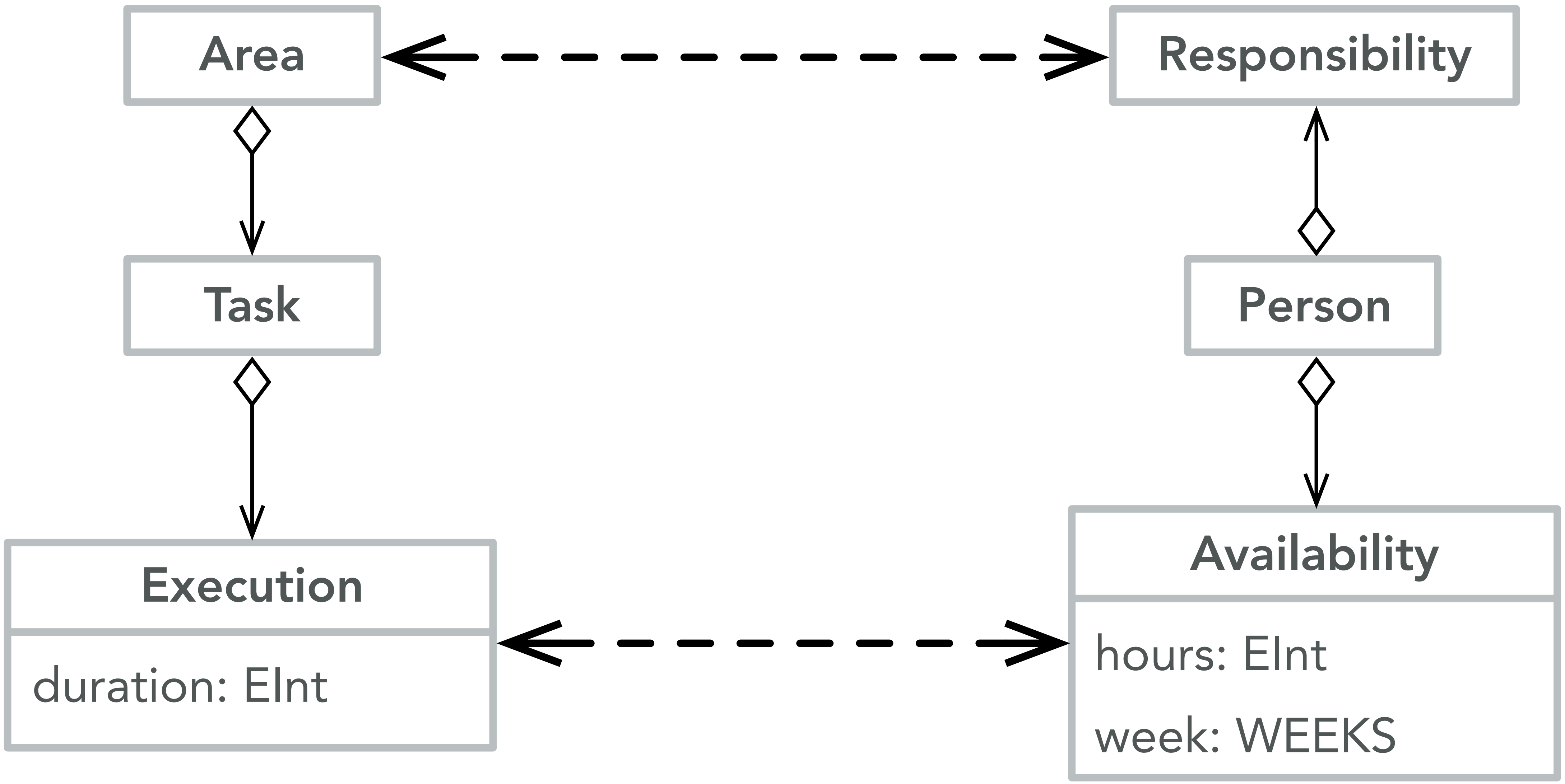
A Real-World Example: Overview

dSPACE

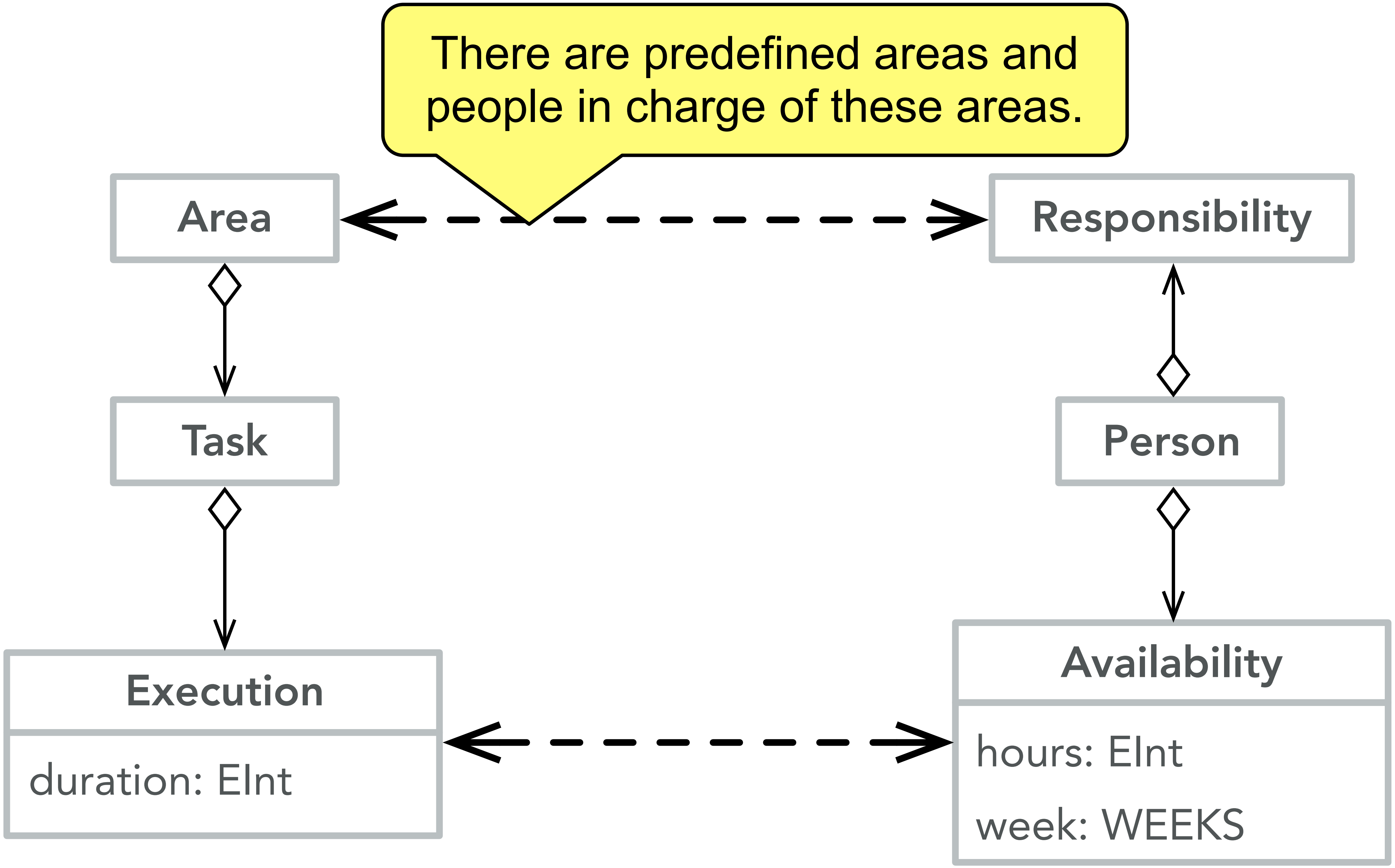


Robin Oppermann:
A Configurable, Model-Driven Approach to Optimal Scheduling using Triple Graph Grammars and Linear Progamming.
Ongoing Master's Thesis, Paderborn University in collaboration with dSpace

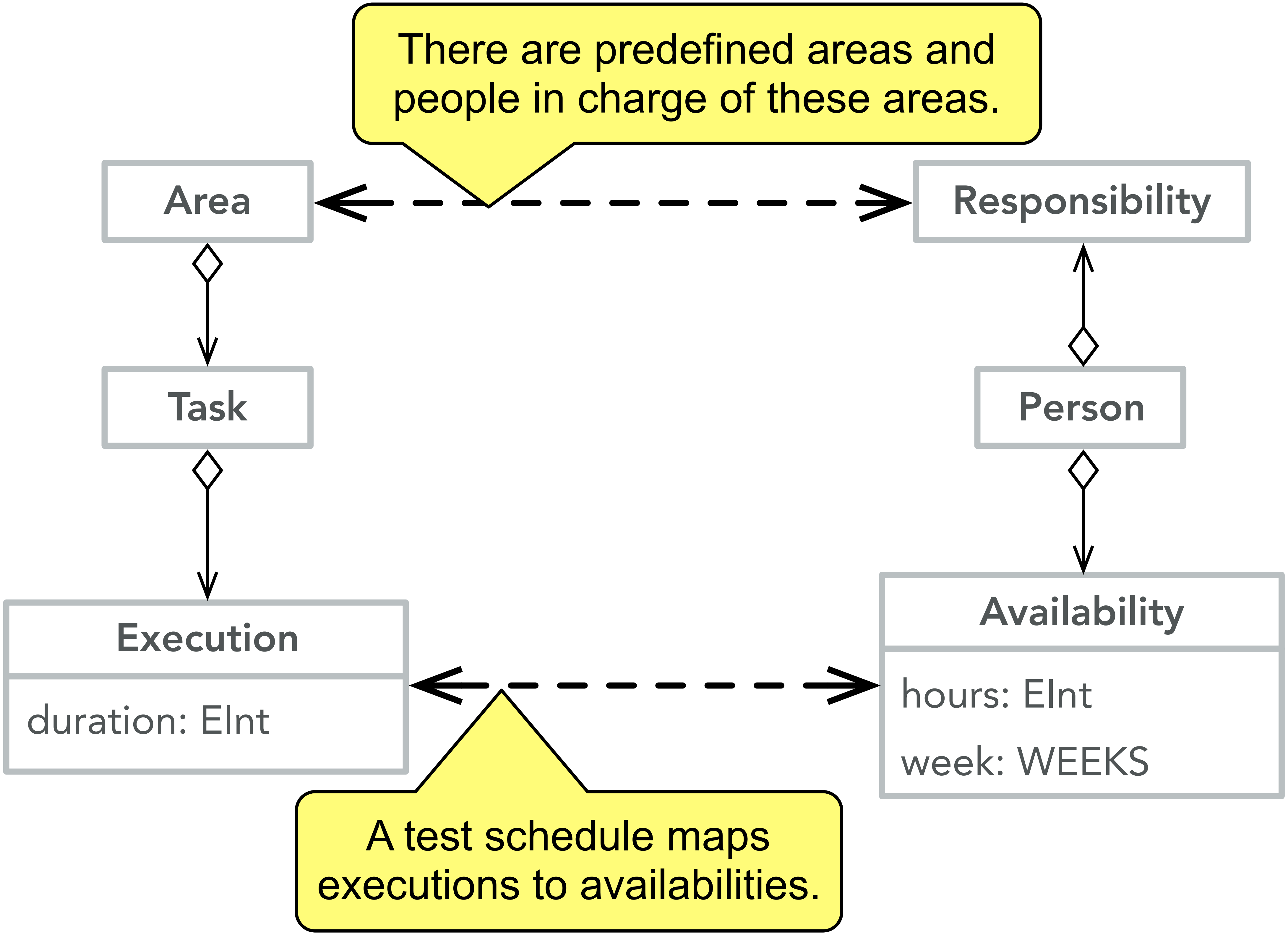
A Real-World Example: Metamodels



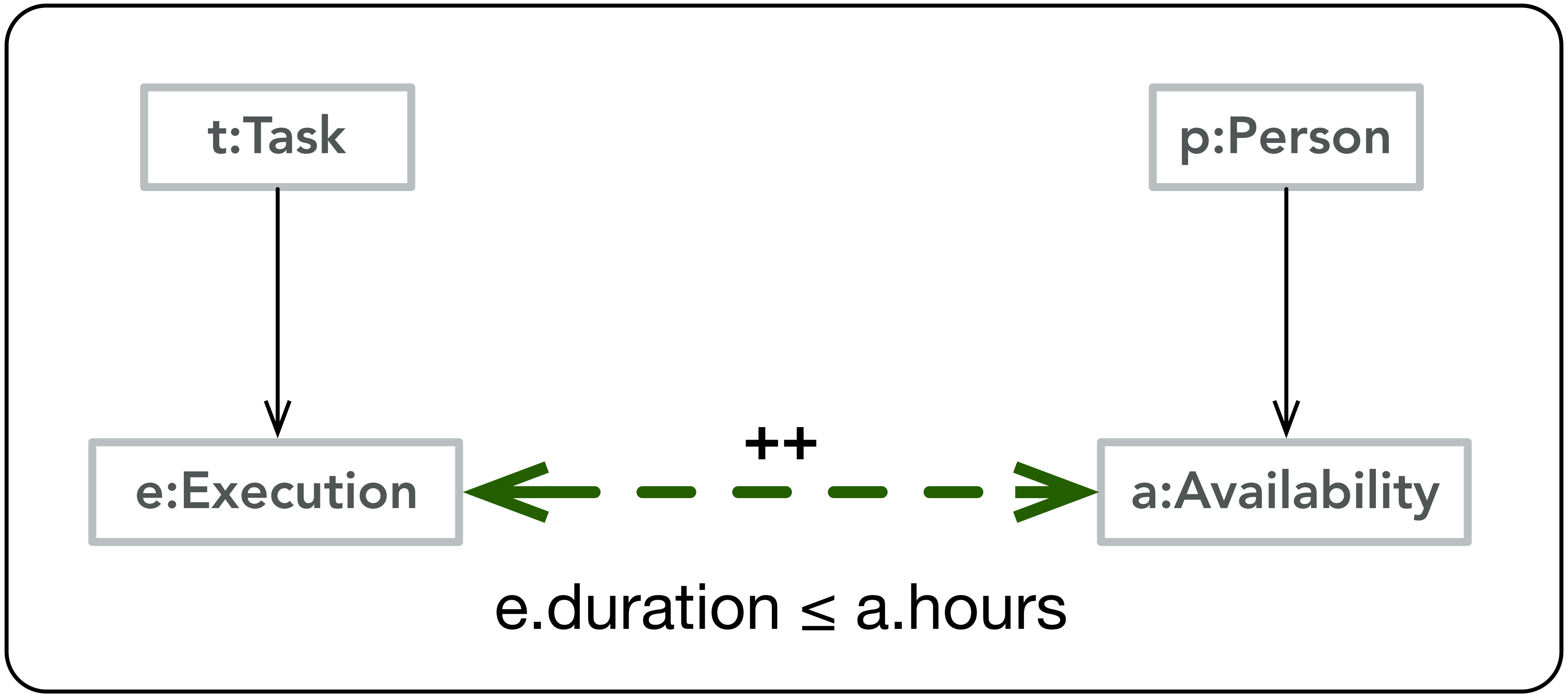
A Real-World Example: Metamodels



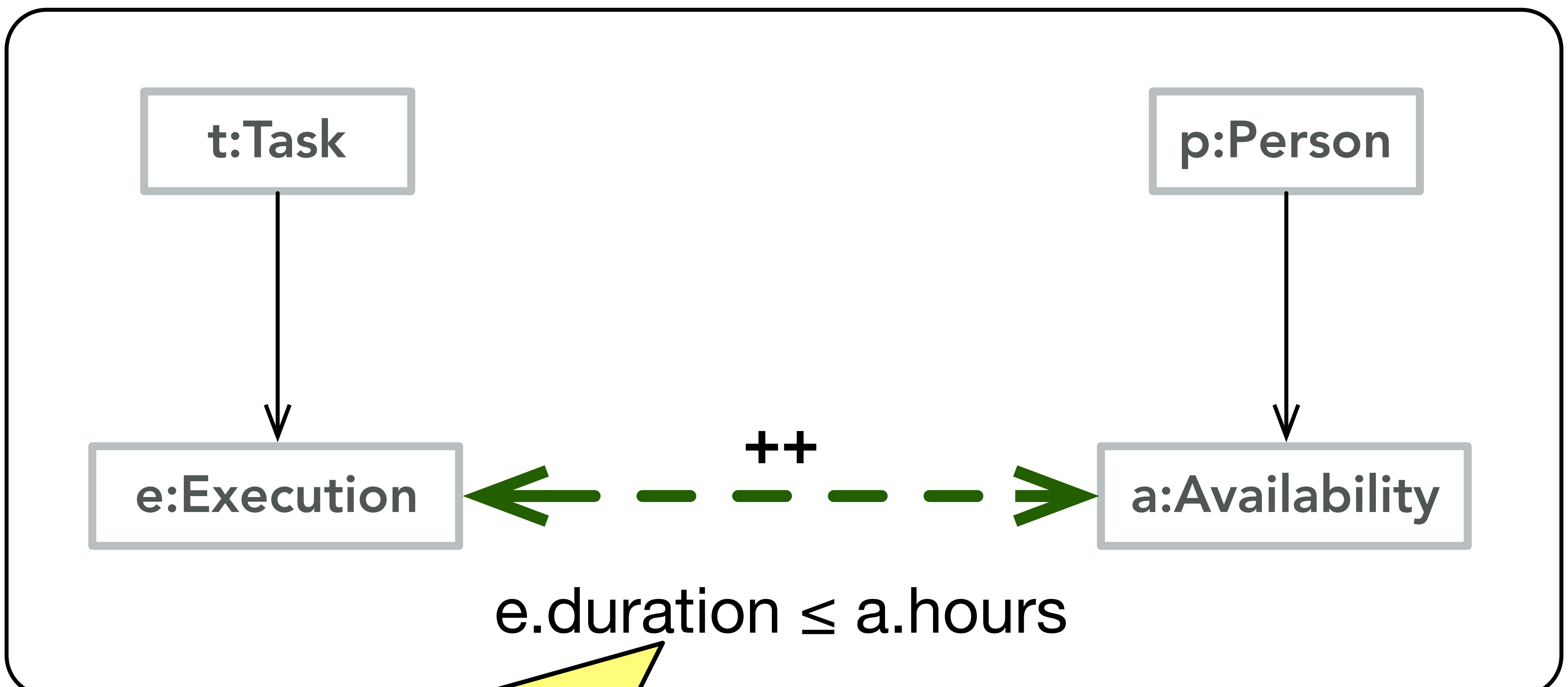
A Real-World Example: Metamodels



A Real-World Example: Allocation Rules

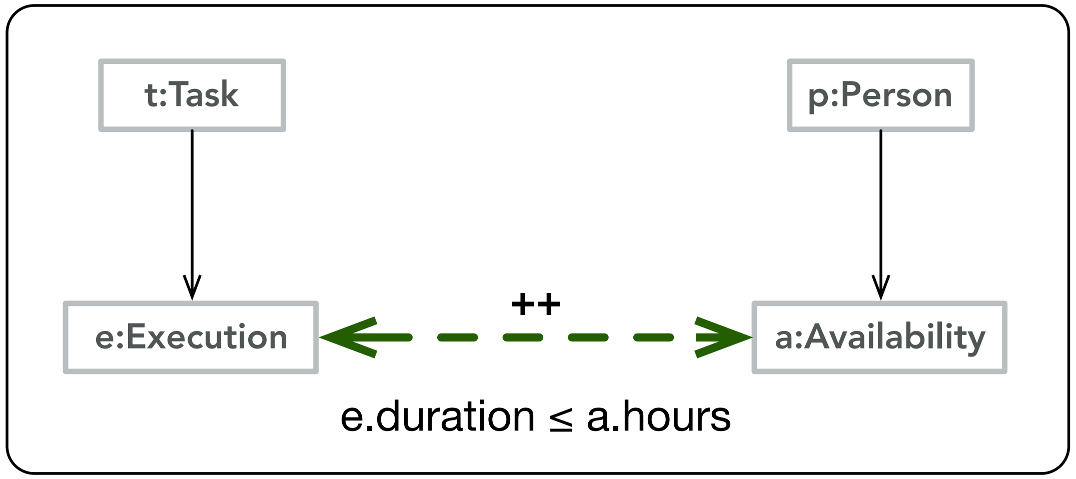


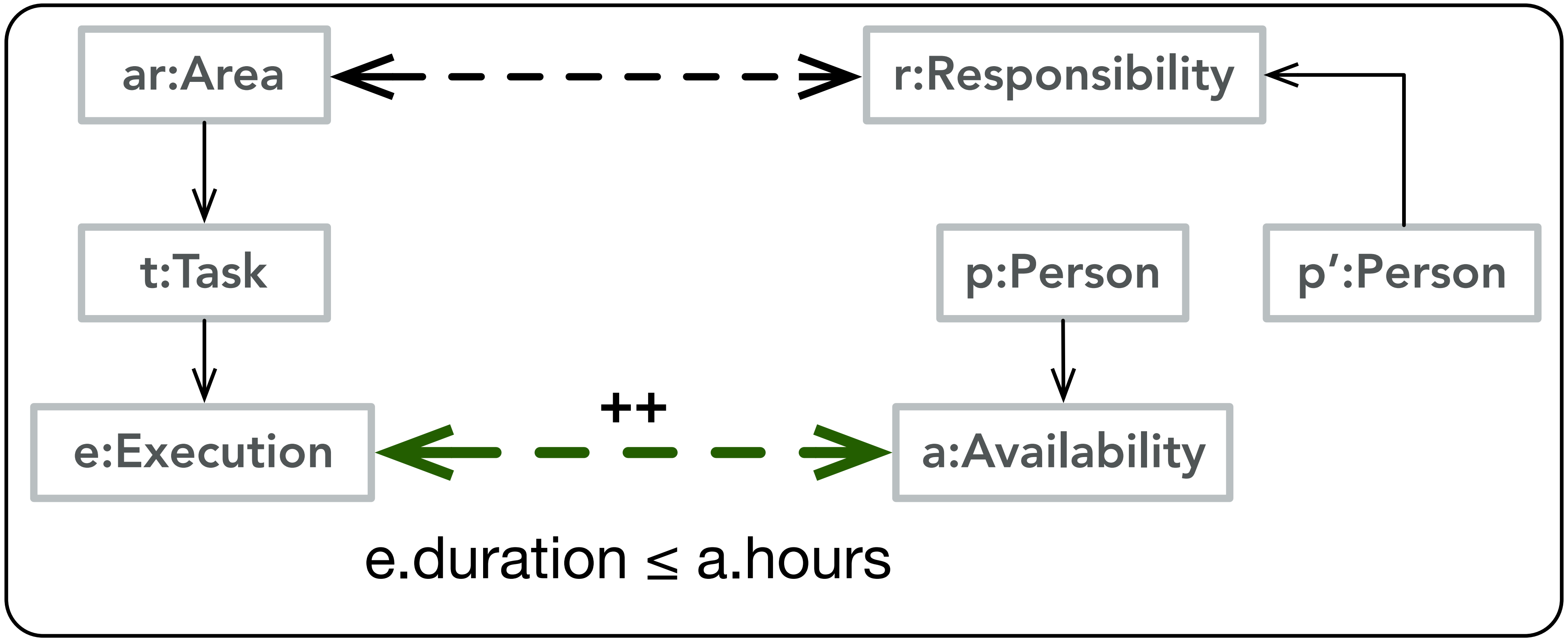
A Real-World Example: Allocation Rules



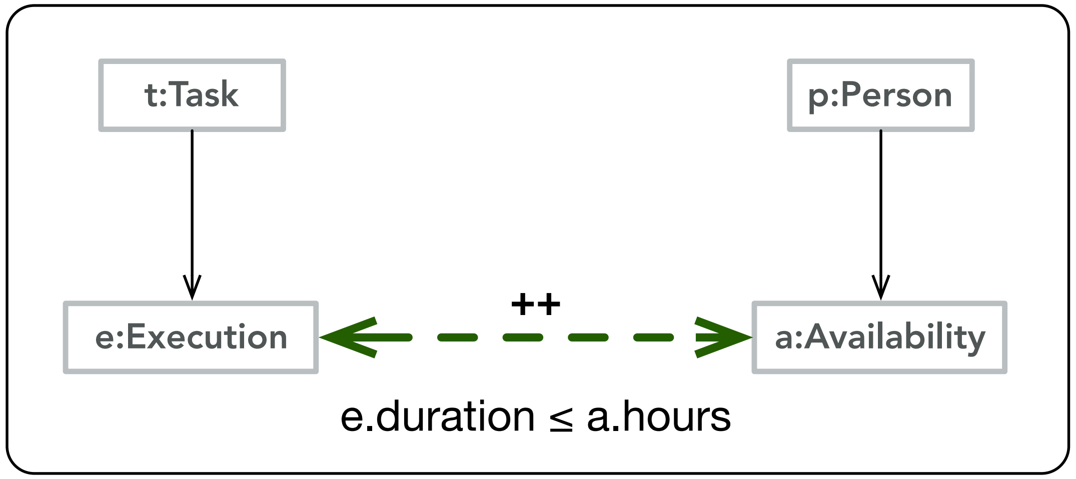
In general, anyone can do anything as long as they have the time for it…

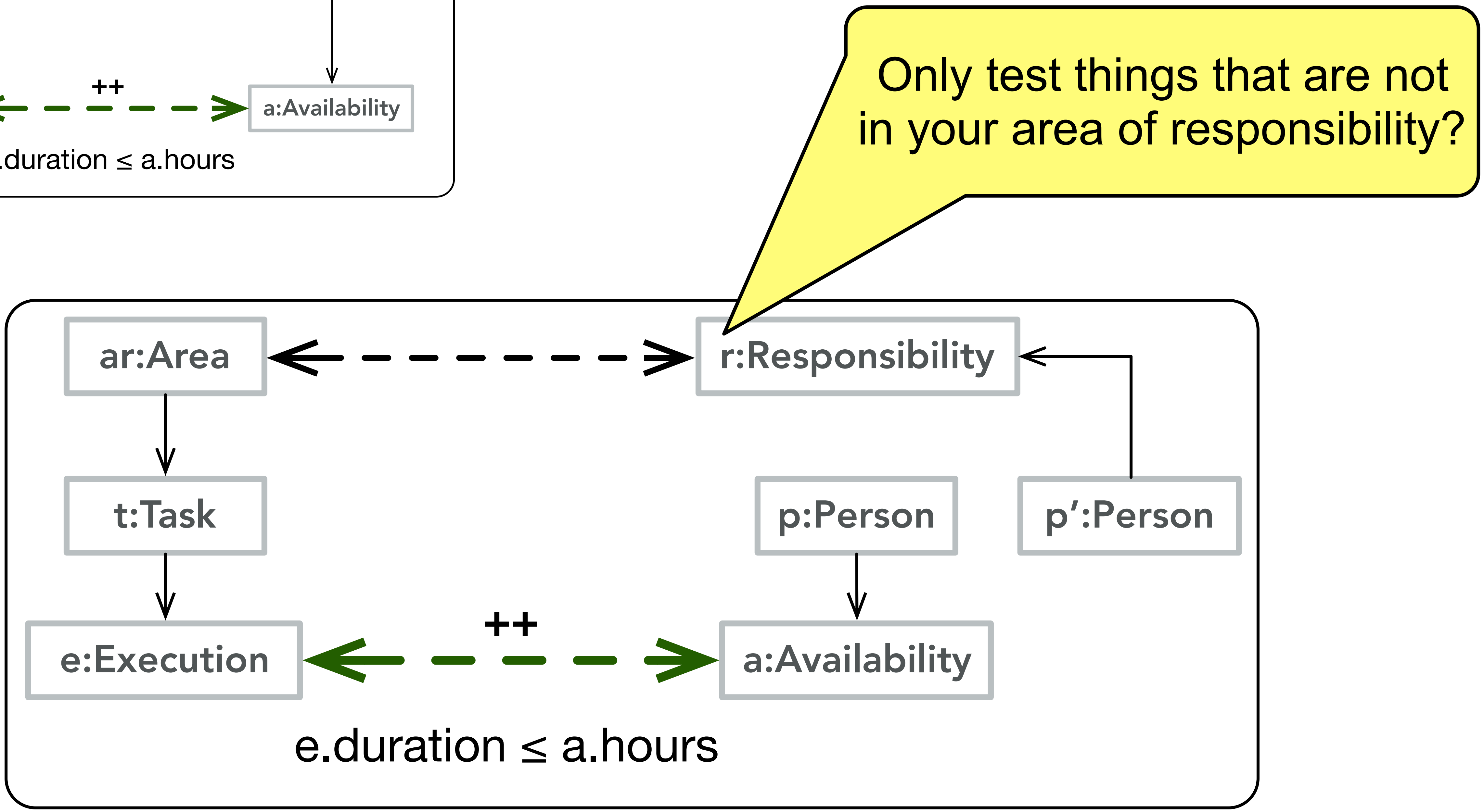
A Real-World Example: Allocation Rules



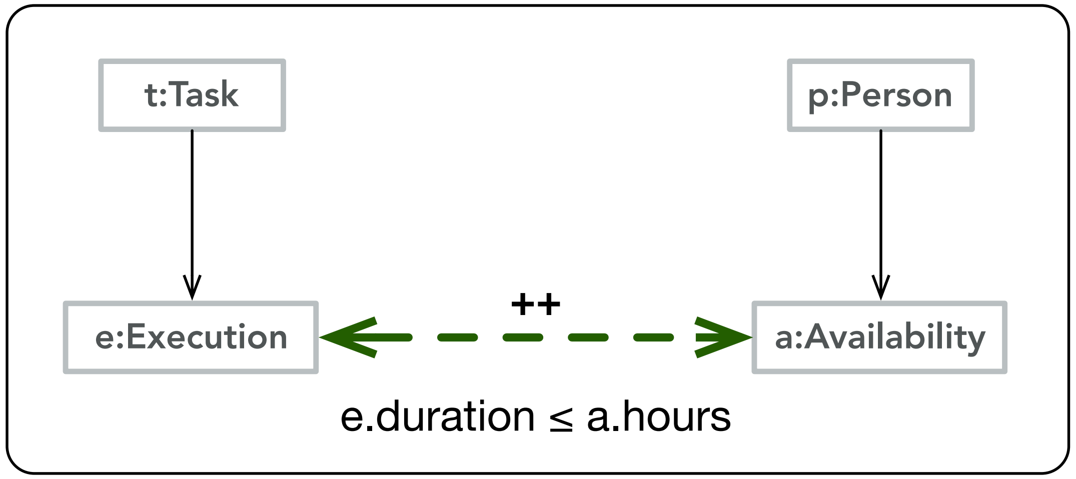


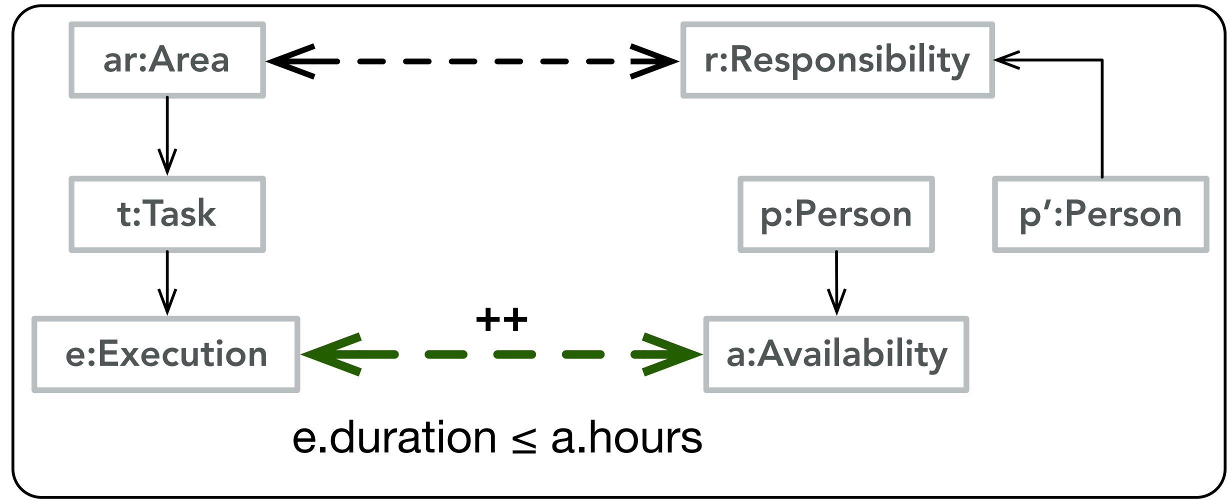
A Real-World Example: Allocation Rules

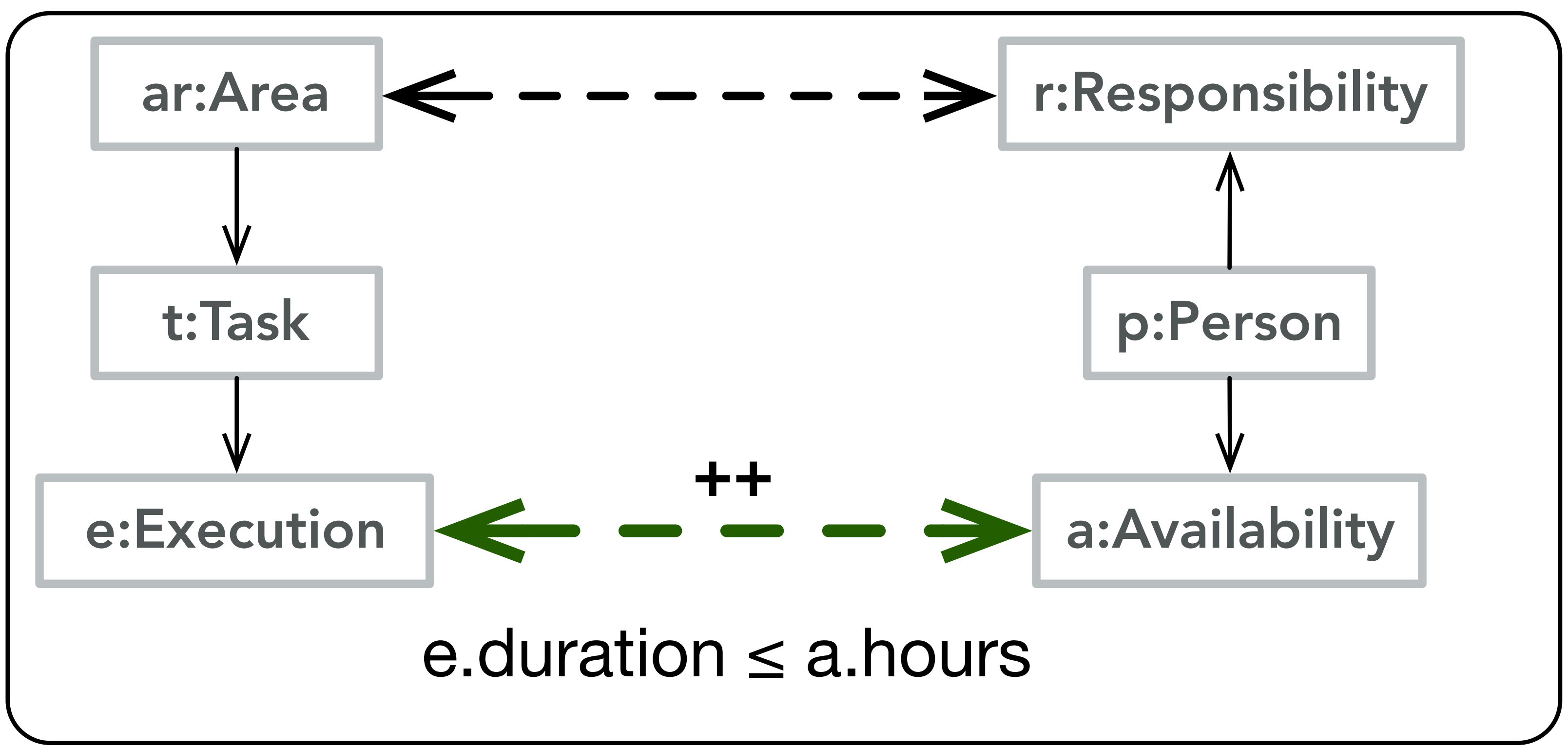




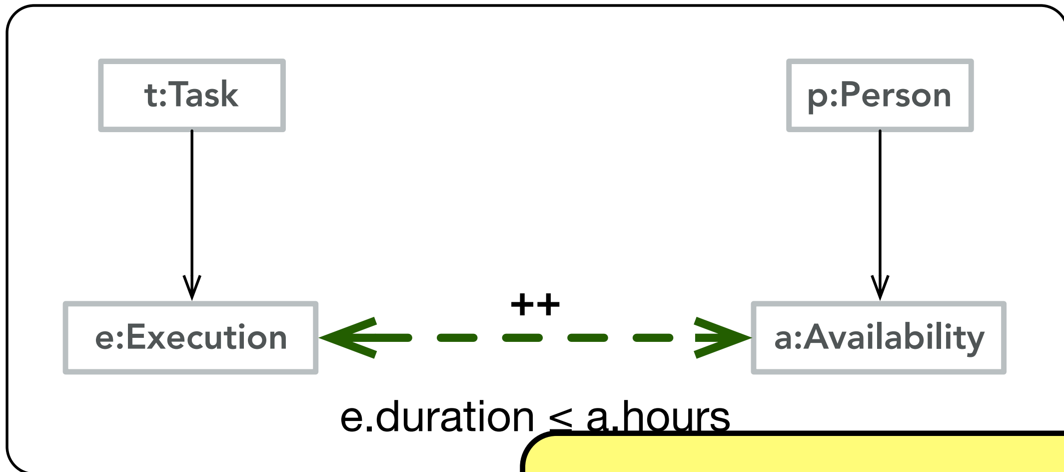
A Real-World Example: Allocation Rules

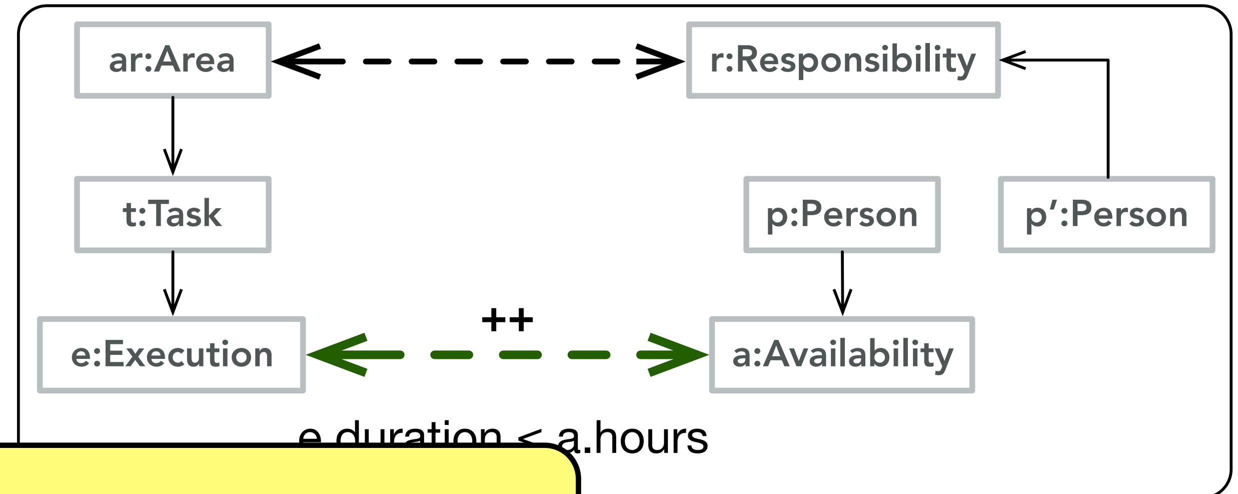




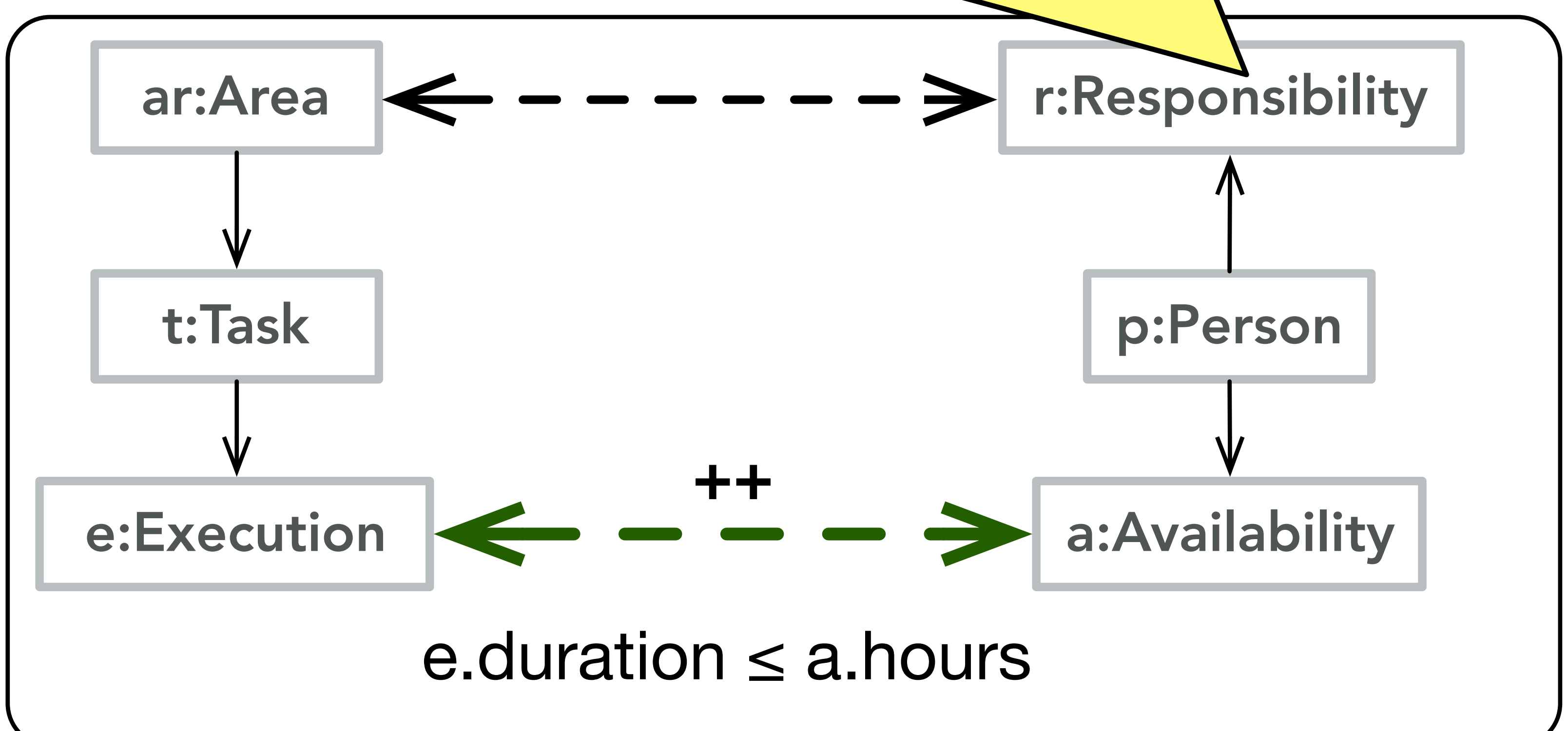


A Real-World Example: Allocation Rules

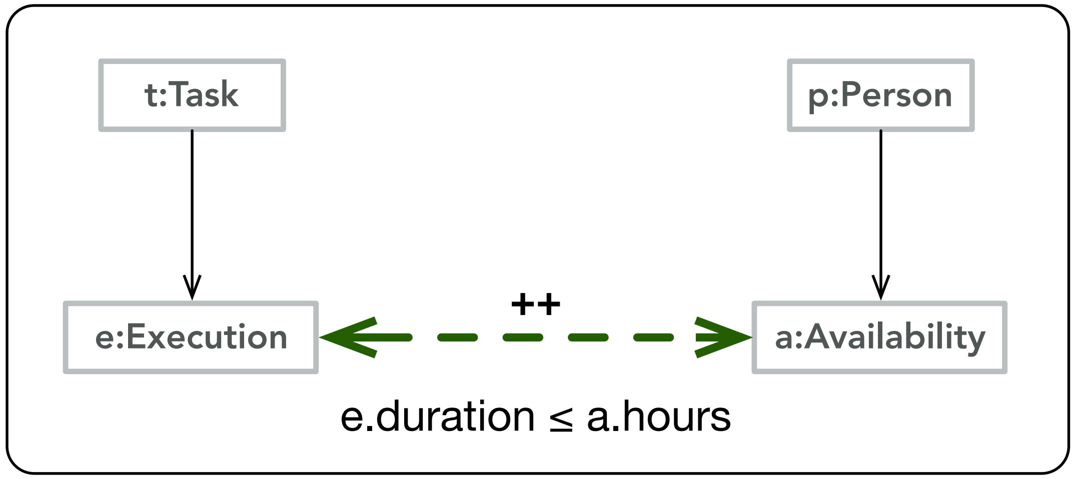


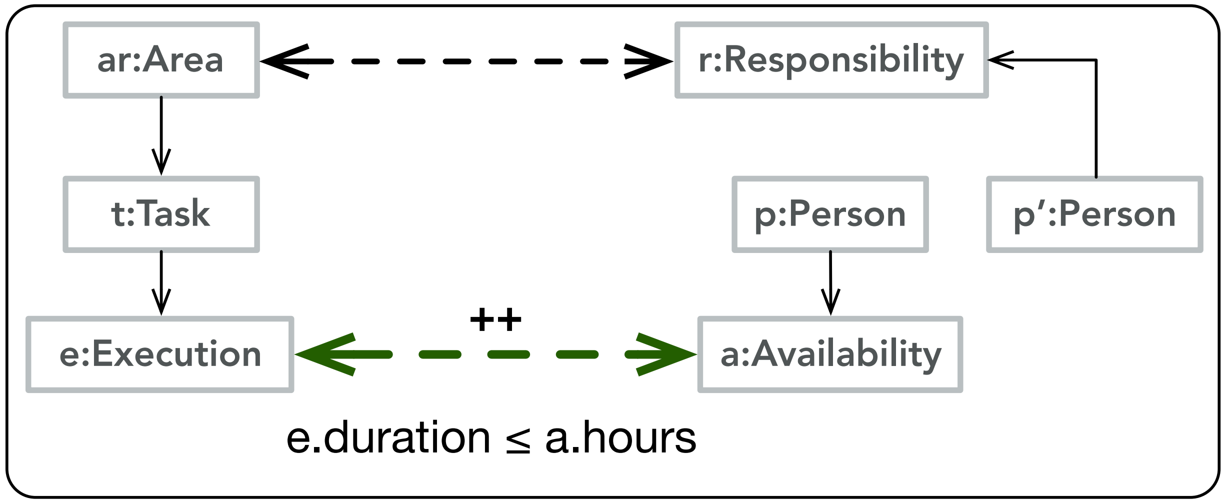


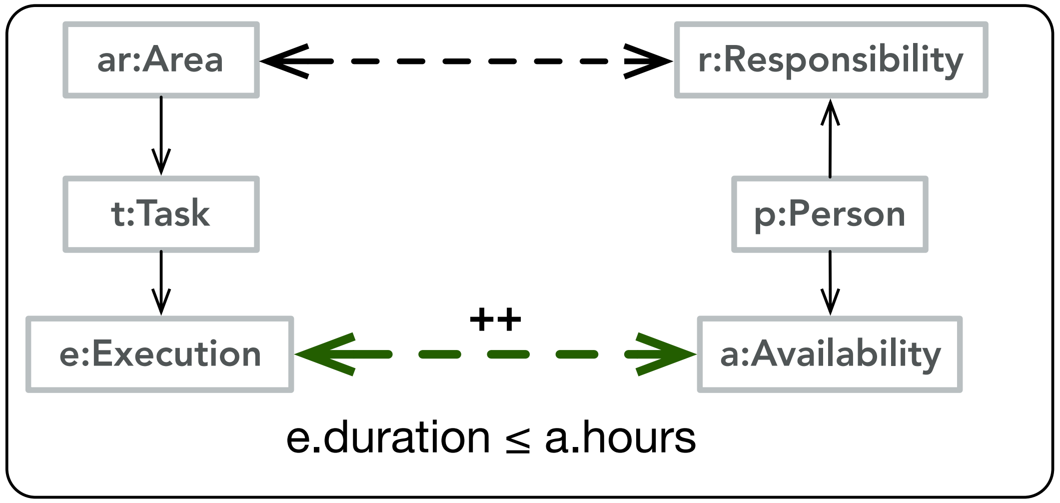
Or perhaps being an expert makes you the best tester possible?

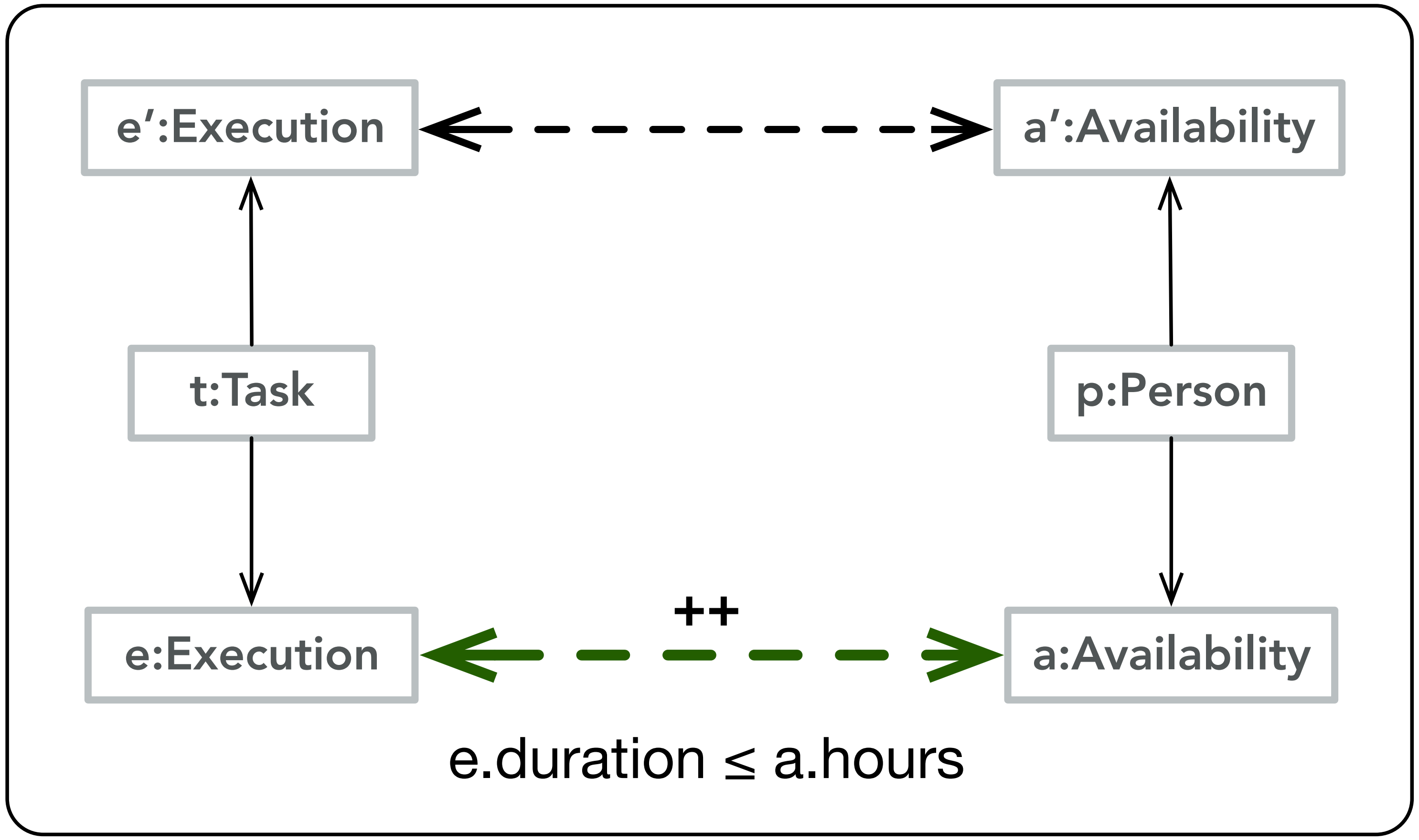


A Real-World Example: Allocation Rules

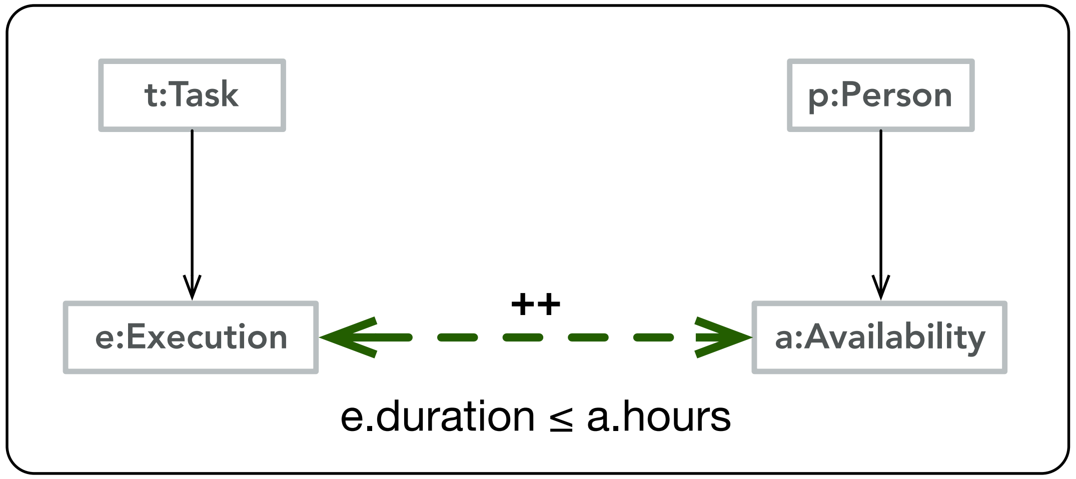


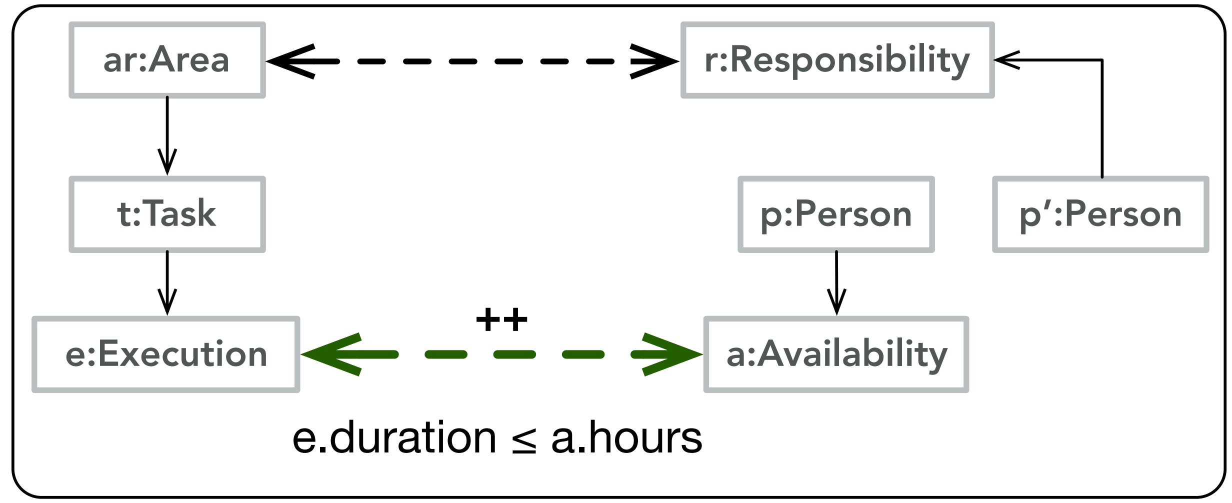


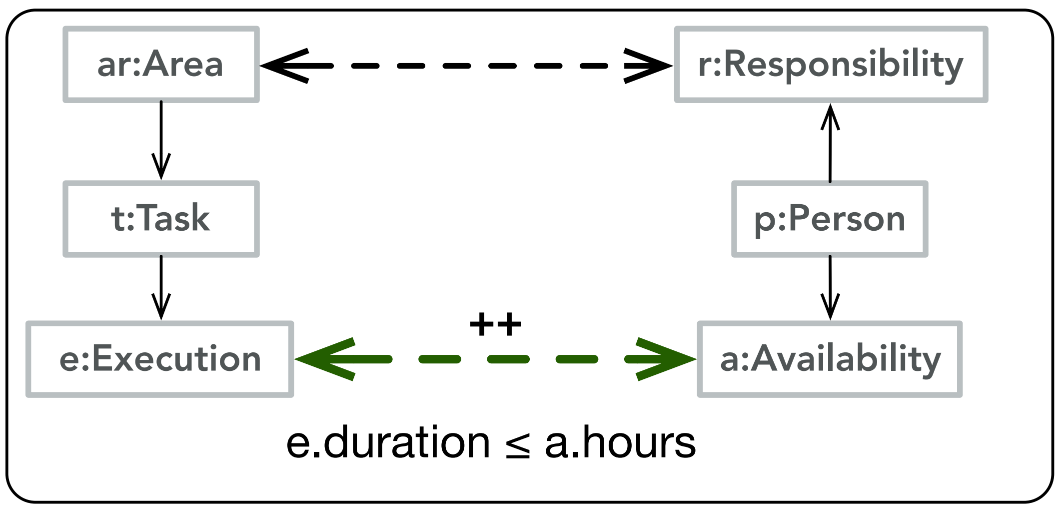


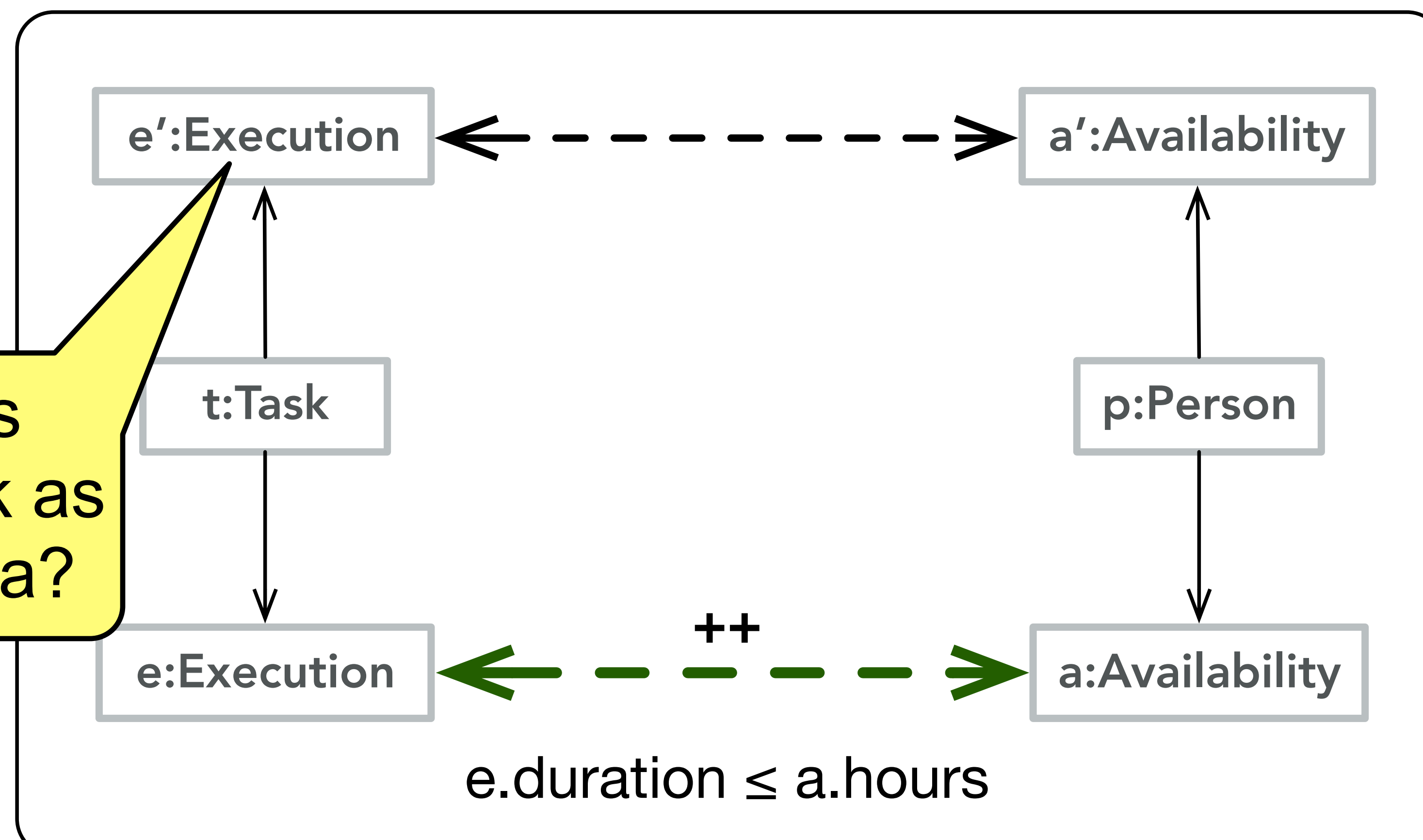


A Real-World Example: Allocation Rules



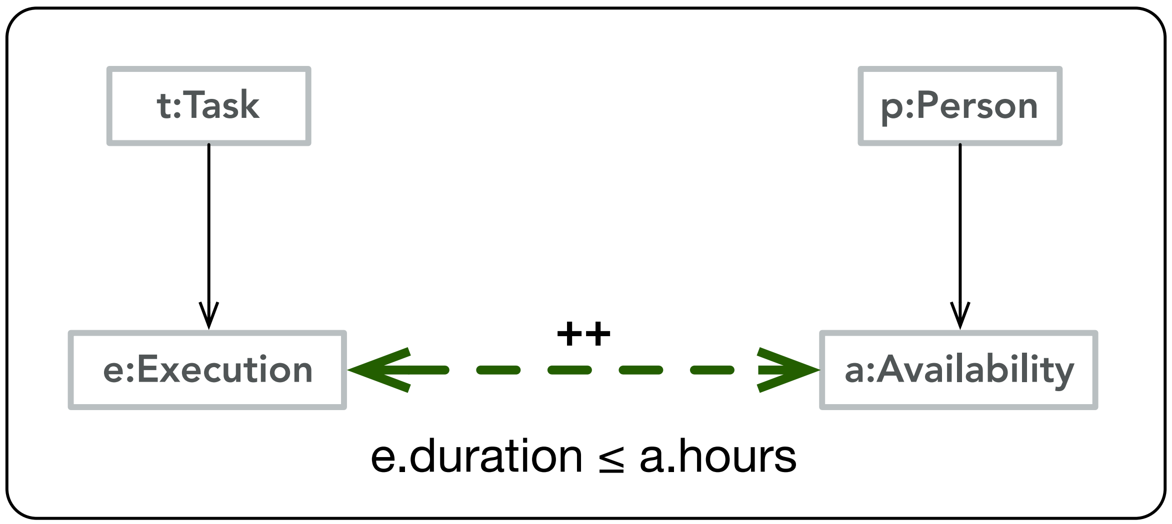


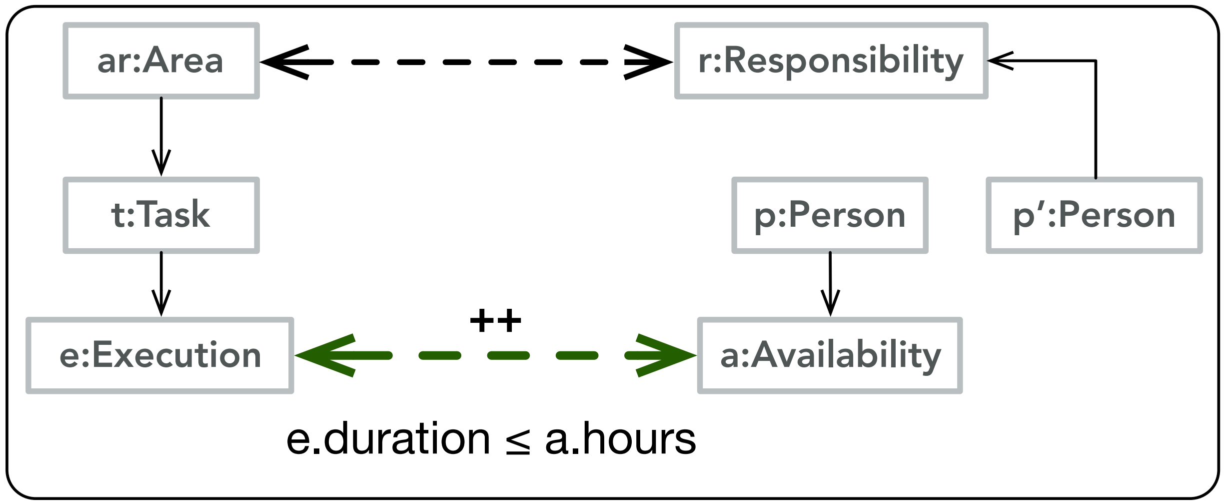


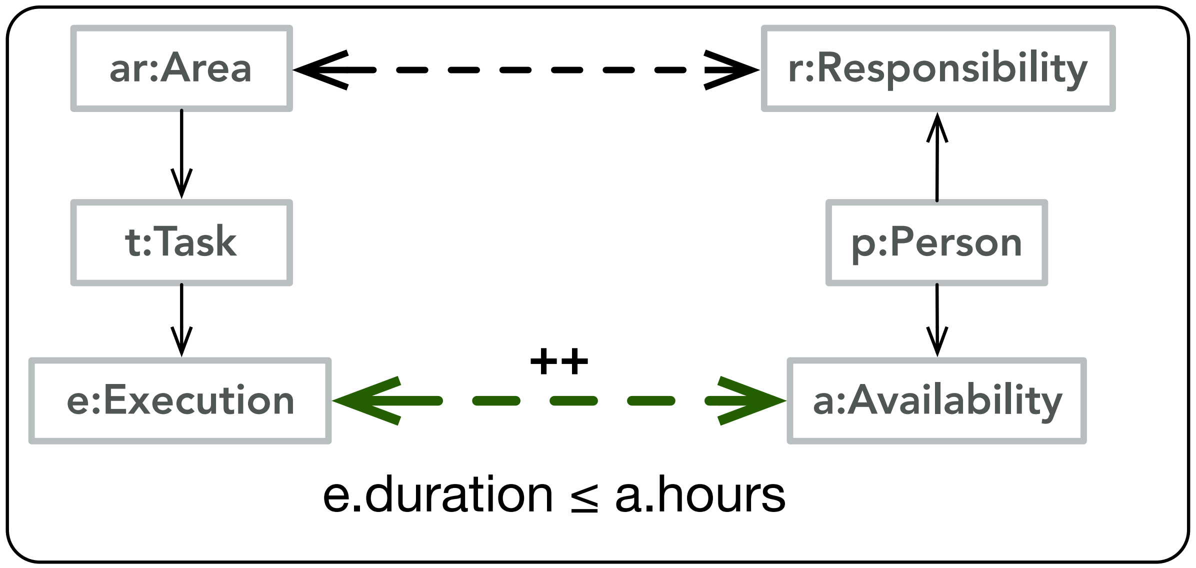


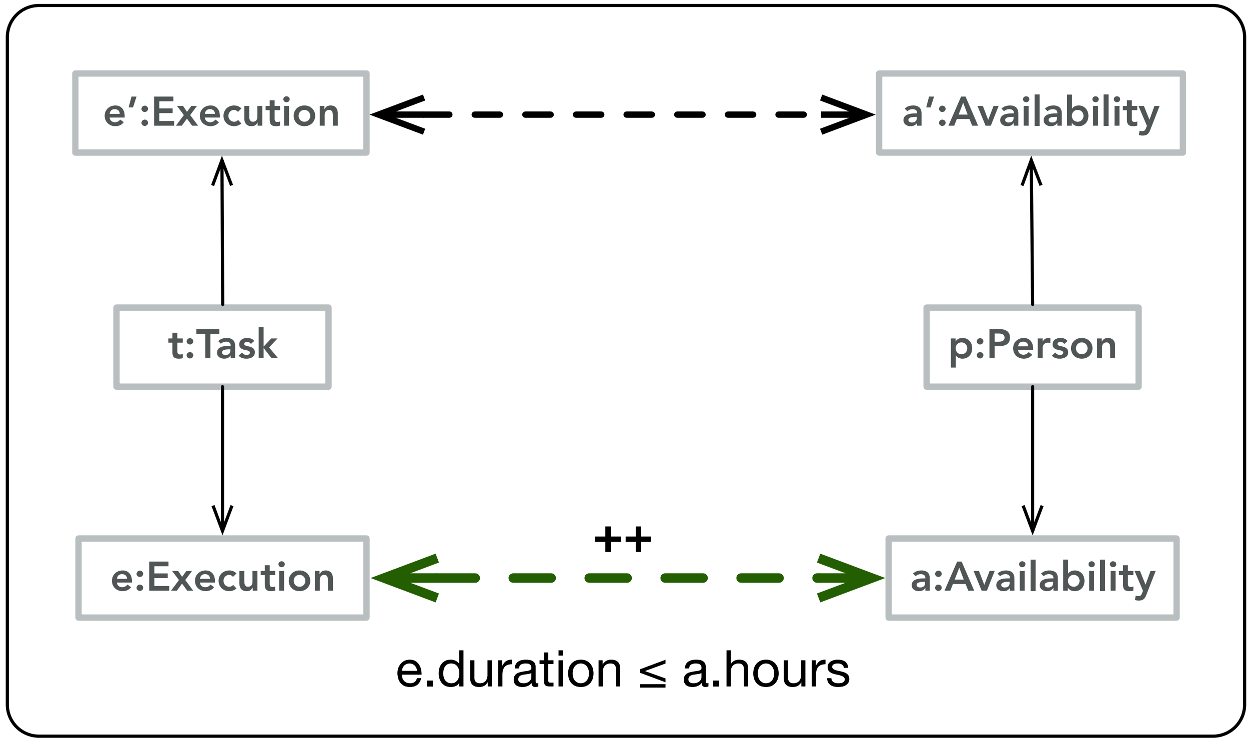
Should the same person test as many executions of the same task as possible? Or is this a terrible idea?

A Real-World Example: Allocation Rules









Similar Application Domains

Similar Application Domains

1. **Allocation Engineering:**
    - Tasks to resources
    - Programs to ECUs
    - Functions to nodes in a network
    - …

Similar Application Domains

1. **Allocation Engineering:**
    - Tasks to resources
    - Programs to ECUs
    - Functions to nodes in a network
    - …

2. **Traceability Maintenance:**
    - Suggest traceability links
    - Check manually created traceability links
    - Flag “suspect links” after changes

Similar Application Domains

1. **Allocation Engineering:**
   - Tasks to resources
   - Programs to ECUs
   - Functions to nodes in a network
   - …

2. **Traceability Maintenance:**
   - Suggest traceability links
   - Check manually created traceability links
   - Flag “suspect links” after changes

3. **Model Synchronisation:**
   - Start with existing, independently created models
   - Identify inconsistencies

Our Approach

2016: Erhan Leblebici:
Towards a Graph Grammar-Based Approach to Inter-Model Consistency Checks with Traceability Support.
Bx@ETAPS 2016: 35-39

2017: Erhan Leblebici, Anthony Anjorin, Andy Schürr:
Inter-model Consistency Checking Using Triple Graph Grammars and Linear Optimization Techniques.
FASE 2017: 191-207

2018: Erhan Leblebici:
Inter-Model Consistency Checking and Restoration with Triple Graph Grammars.
PhD Thesis, Darmstadt University of Technology, Germany 2018

2018: Nils Weidmann:
Consistency Management via a Combination of Triple Graph Grammars and Integer Linear Programming.
Master's Thesis, Paderborn University, Germany 2018

Our Approach

2016: Erhan Leblebici:
Towards a Graph Grammar-Based Approach to Inter-Model Consistency Checks with Traceability Support.
Bx@ETAPS 2016: 35-39

2017: Erhan Leblebici, Anthony Anjorin, Andy Schürr:
Inter-model Consistency Checking Using Triple Graph Grammars and Linear Optimization Techniques.
FASE 2017: 191-207

2018: Erhan Leblebici:
Inter-Model Consistency Checking and Restoration with Triple Graph Grammars.
PhD Thesis, Darmstadt University of Technology, Germany 2018

2018: Nils Weidmann:
Consistency Management via a Combination of Triple Graph Grammars and Integer Linear Programming.
Master's Thesis, Paderborn University, Germany 2018

Our Approach

Basic idea of how to perform consistency checking with TGGs

2016: Erhan Leblebici:
Towards a Graph Grammar-Based Approach to Inter-Model Consistency Checks with Traceability Support.
Bx@ETAPS 2016: 35-39

2017: Erhan Leblebici, Anthony Anjorin, Andy Schürr:
Inter-model Consistency Checking Using Triple Graph Grammars and Linear Optimization Techniques.
FASE 2017: 191-207

2018: Erhan Leblebici:
Inter-Model Consistency Checking and Restoration with Triple Graph Grammars.
PhD Thesis, Darmstadt University of Technology, Germany 2018

2018: Nils Weidmann:
Consistency Management via a Combination of Triple Graph Grammars and Integer Linear Programming.
Master’s Thesis, Paderborn University, Germany 2018

Our Approach

2016: Erhan Leblebici:
Towards a Graph Grammar-Ba
Consistency Checks with Trac
Bx@ETAPS 2016: 35-39

Full details, implementation, and evaluation in eMoflon

2017: Erhan Leblebici, Anthony Anjorin, Andy Schürr:
Inter-model Consistency Checking Using Triple Graph Grammars and Linear Optimization Techniques.
FASE 2017: 191-207

2018: Erhan Leblebici:
Inter-Model Consistency Checking and Restoration with Triple Graph Grammars.
PhD Thesis, Darmstadt University of Technology, Germany 2018

2018: Nils Weidmann:
Consistency Management via a Combination of Triple Graph Grammars and Integer Linear Programming.
Master's Thesis, Paderborn University, Germany 2018

Our Approach

2016: Erhan Leblebici:
Towards a Graph Grammar-Based Approach to Inter-Model Consistency Checks with Traceability Support.
Bx@ETAPS 2016: 35-39

2017: Erhan Leblebici, Anthony Anjorin, Andy Schürr:
Inter-model Consistency Check
Grammars and Linear Optimiza
FASE 2017: 191-207

Remaining formal proofs, industrial case with Siemens

2018: Erhan Leblebici:
Inter-Model Consistency Checking and Restoration with Triple Graph Grammars.
PhD Thesis, Darmstadt University of Technology, Germany 2018

2018: Nils Weidmann:
Consistency Management via a Combination of Triple Graph Grammars and Integer Linear Programming.
Master’s Thesis, Paderborn University, Germany 2018

Our Approach

2016: Erhan Leblebici:
Towards a Graph Grammar-Based Approach to Inter-Model Consistency Checks with Traceability Support.
Bx@ETAPS 2016: 35-39

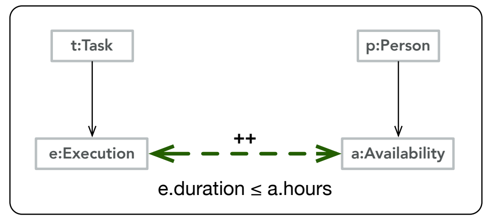
2017: Erhan Leblebici, Anthony Anjorin, Andy Schürr:
Inter-model Consistency Checking Using Triple Graph Grammars and Linear Optimization Techniques.
FASE 2017: 191-207

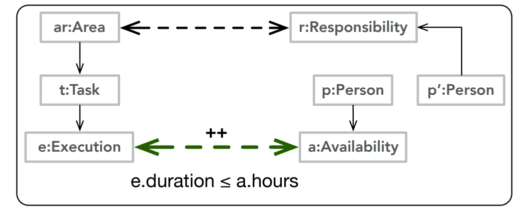
2018: Erhan Leblebici:
Inter-Model Consistency Checki[ng] ... Triple Graph Grammars.
PhD Thesis, Darmstadt University ...

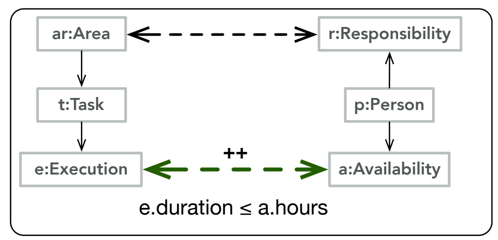
Generalisation of the approach to other consistency management tasks (work in progress)

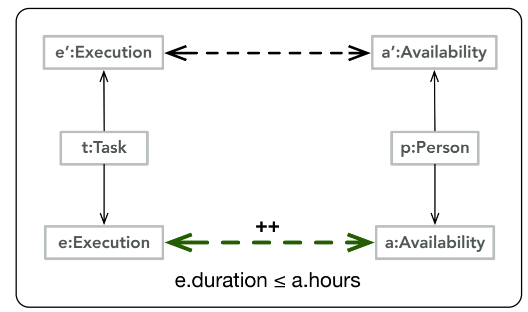
2018: Nils Weidmann:
Consistency Management via a Combination of Triple Graph Grammars and Integer Linear Programming.
Master's Thesis, Paderborn University, Germany 2018

Step 1: Collect all Candidates



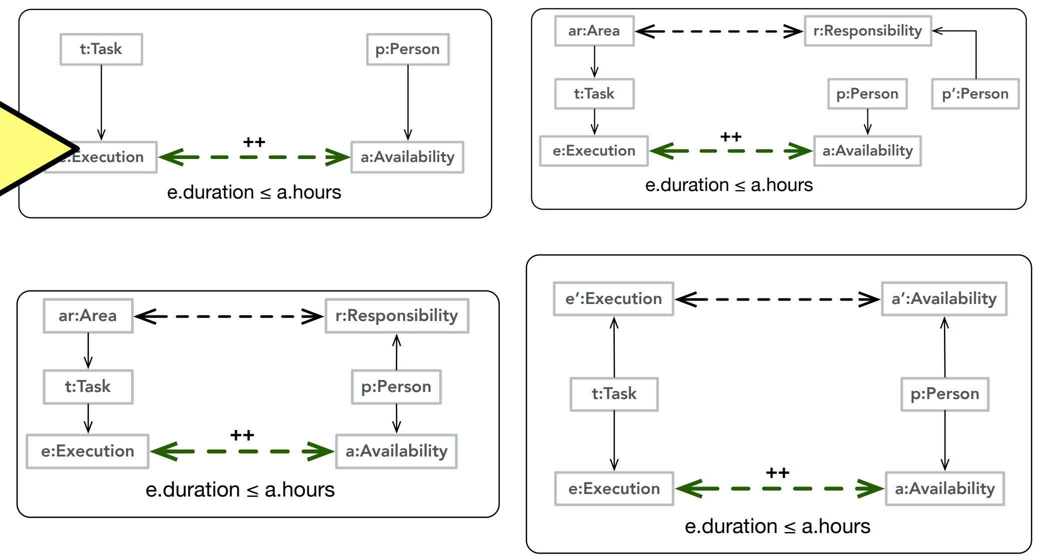






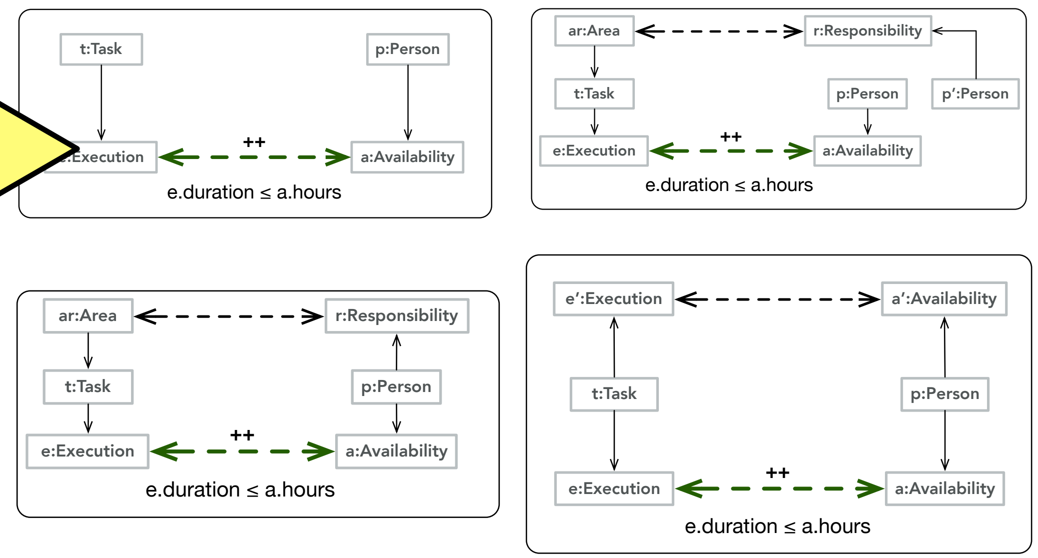
Step 1: Collect all Candidates

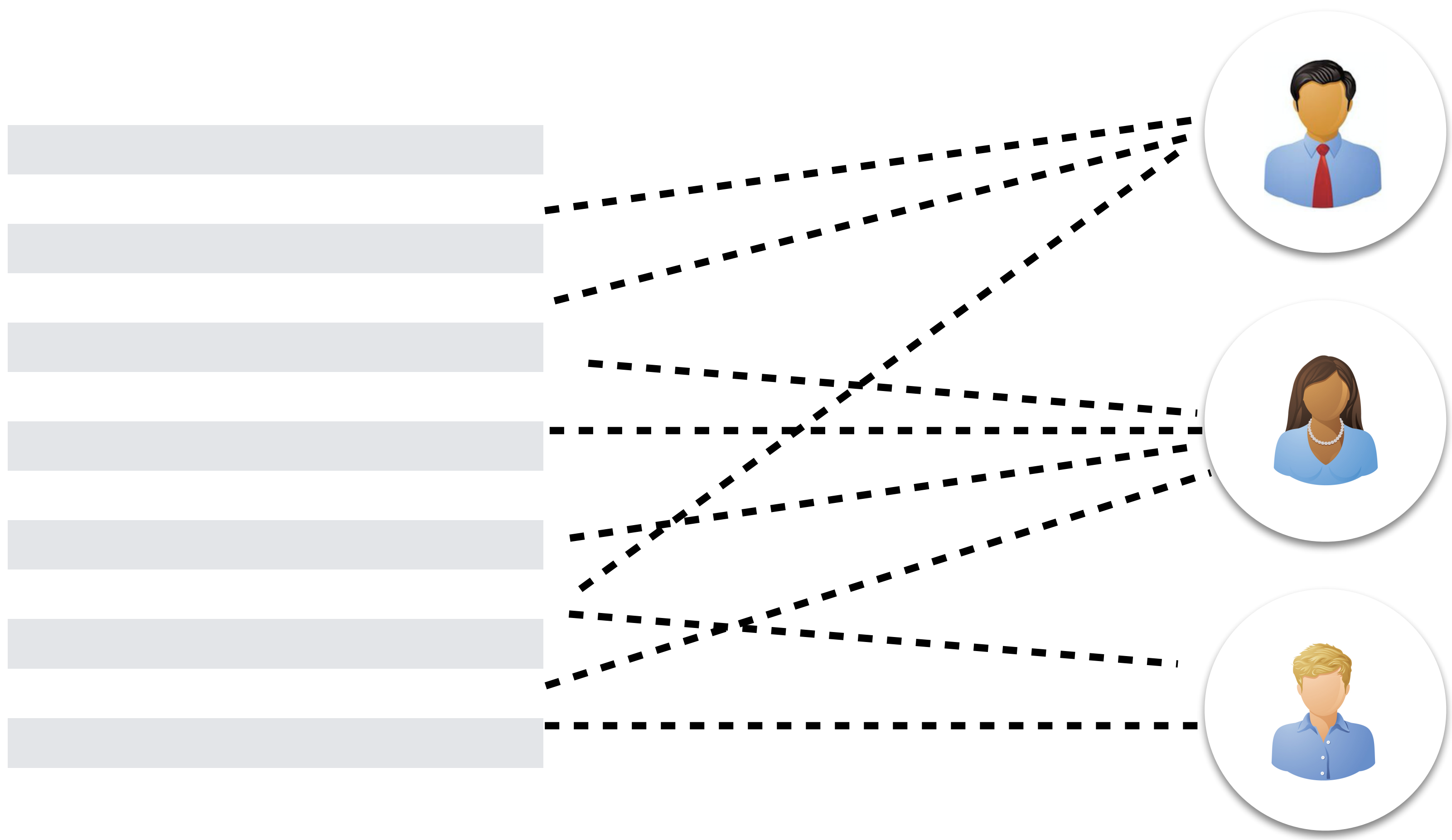
Use allocation rules (derived from a TGG) to create *all possible* correspondence links.



Step 1: Collect all Candidates

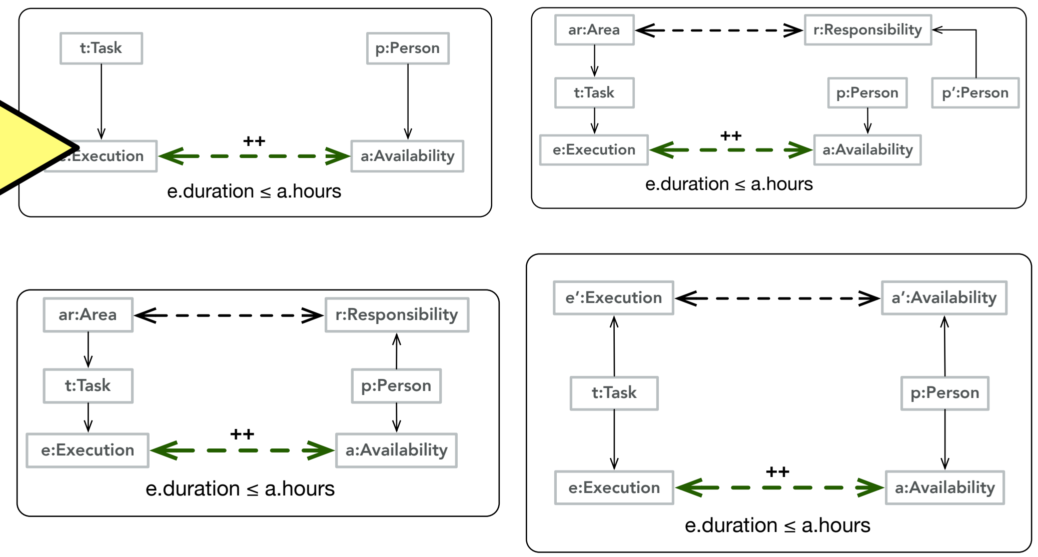
Use allocation rules (derived from a TGG) to create *all possible* correspondence links.



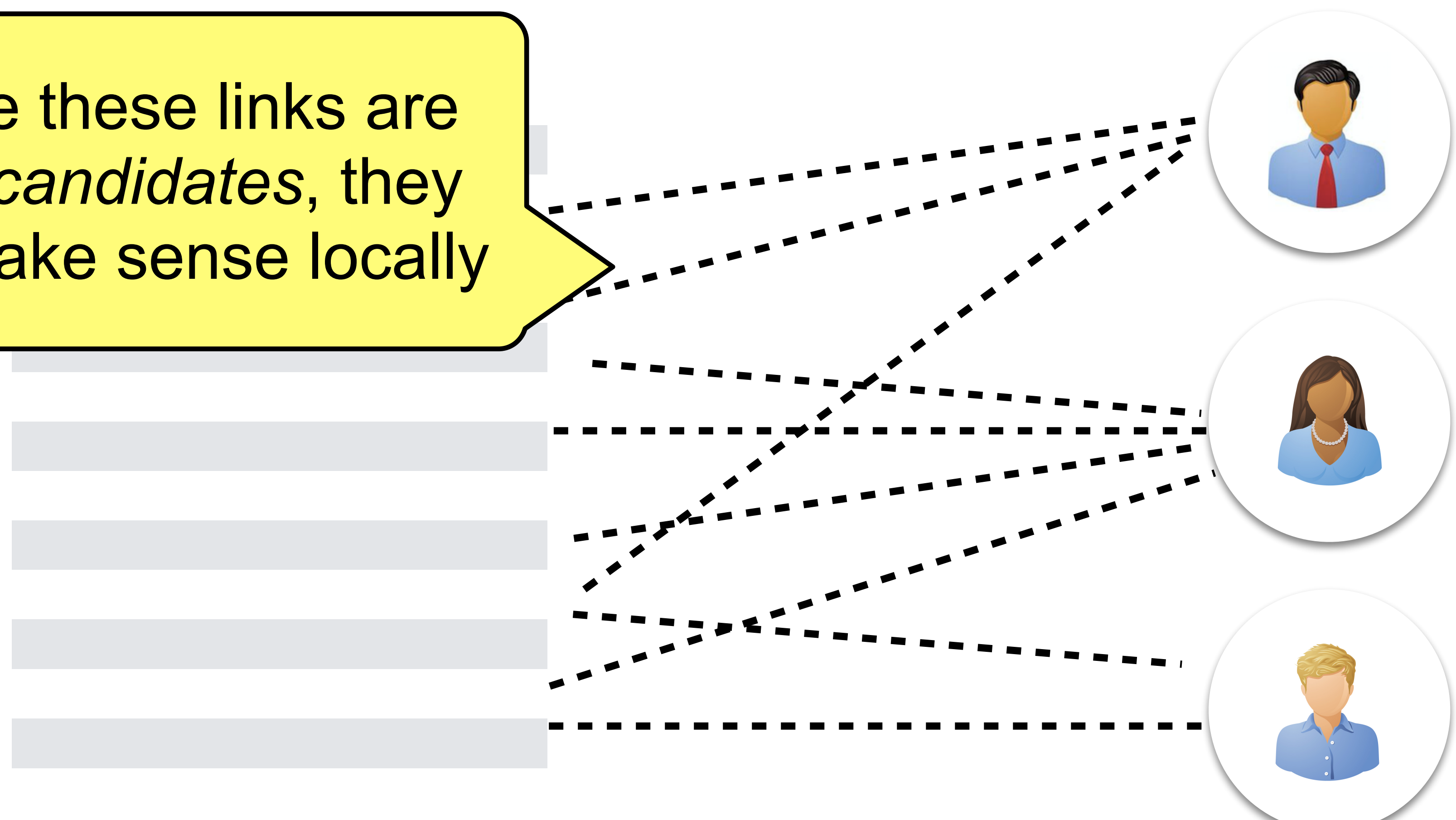


Step 1: Collect all Candidates

Use allocation rules (derived from a TGG) to create *all possible* correspondence links.

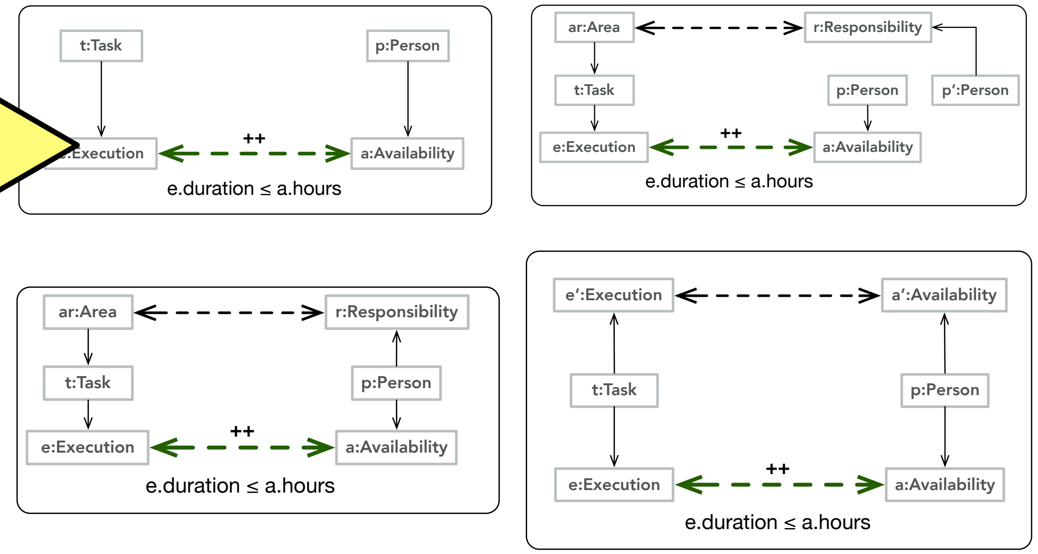


While these links are only *candidates*, they still make sense locally

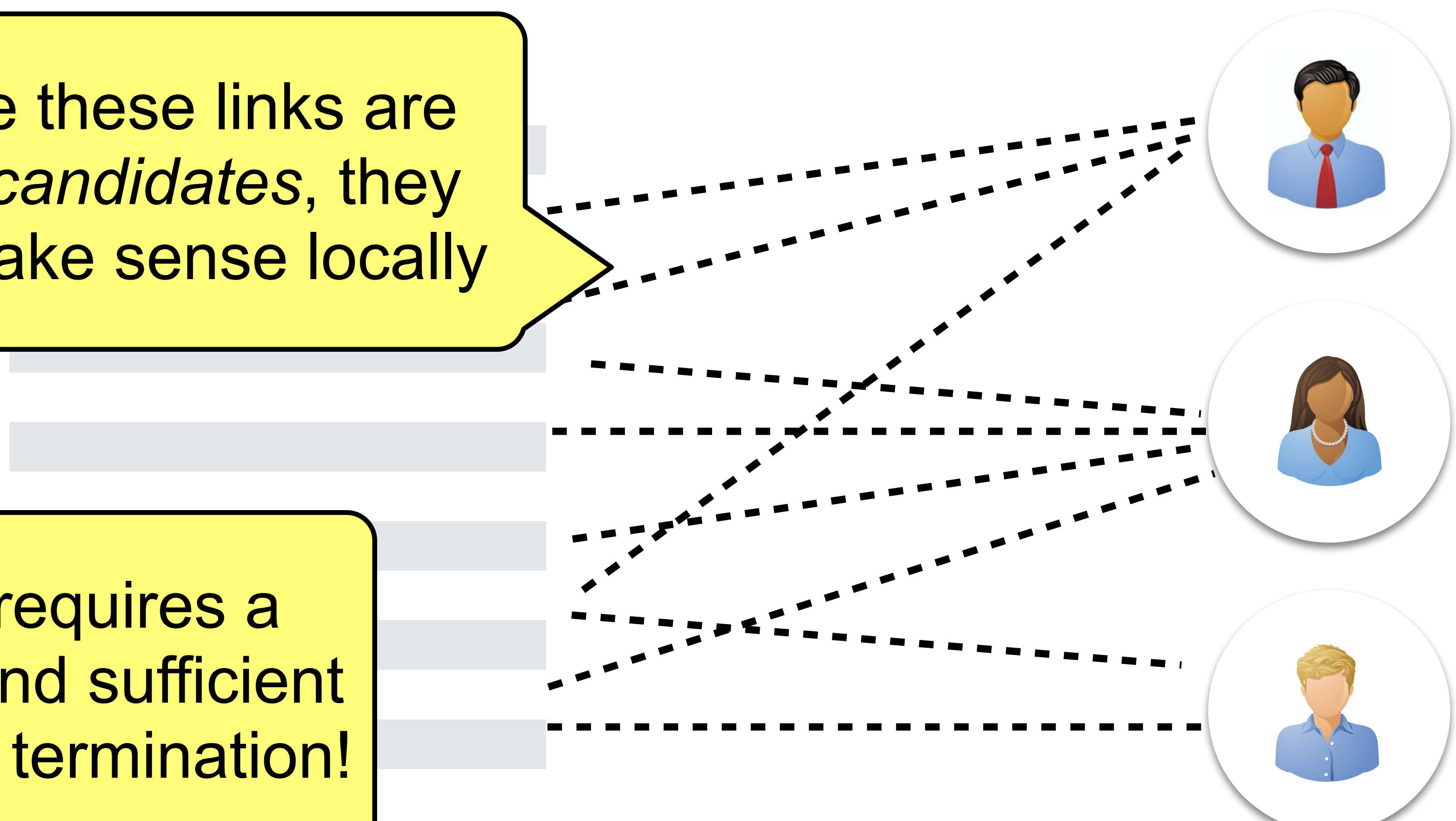


Step 1: Collect all Candidates

Use allocation rules (derived from a TGG) to create *all possible* correspondence links.



While these links are only *candidates*, they still make sense locally

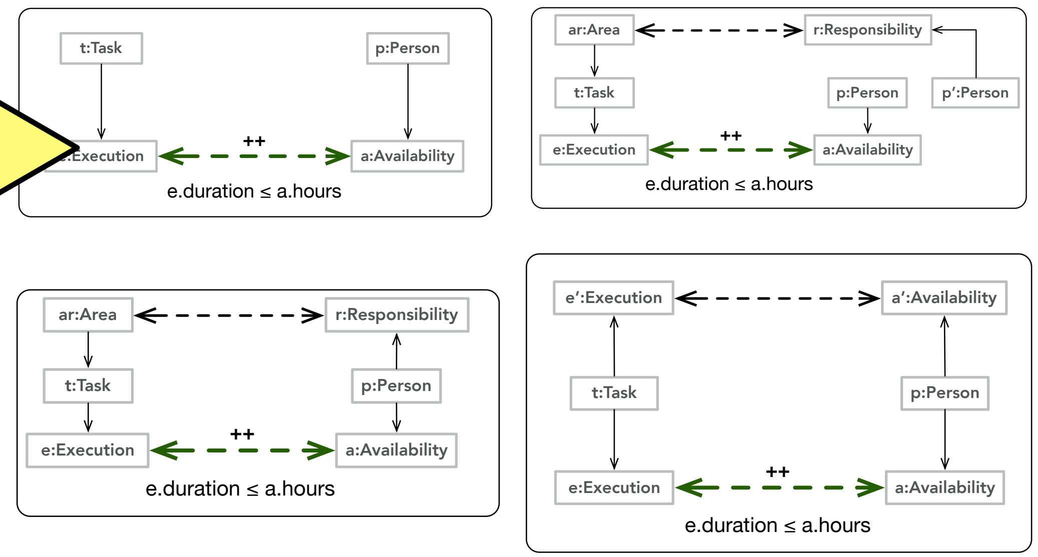


This step requires a necessary and sufficient condition for termination!

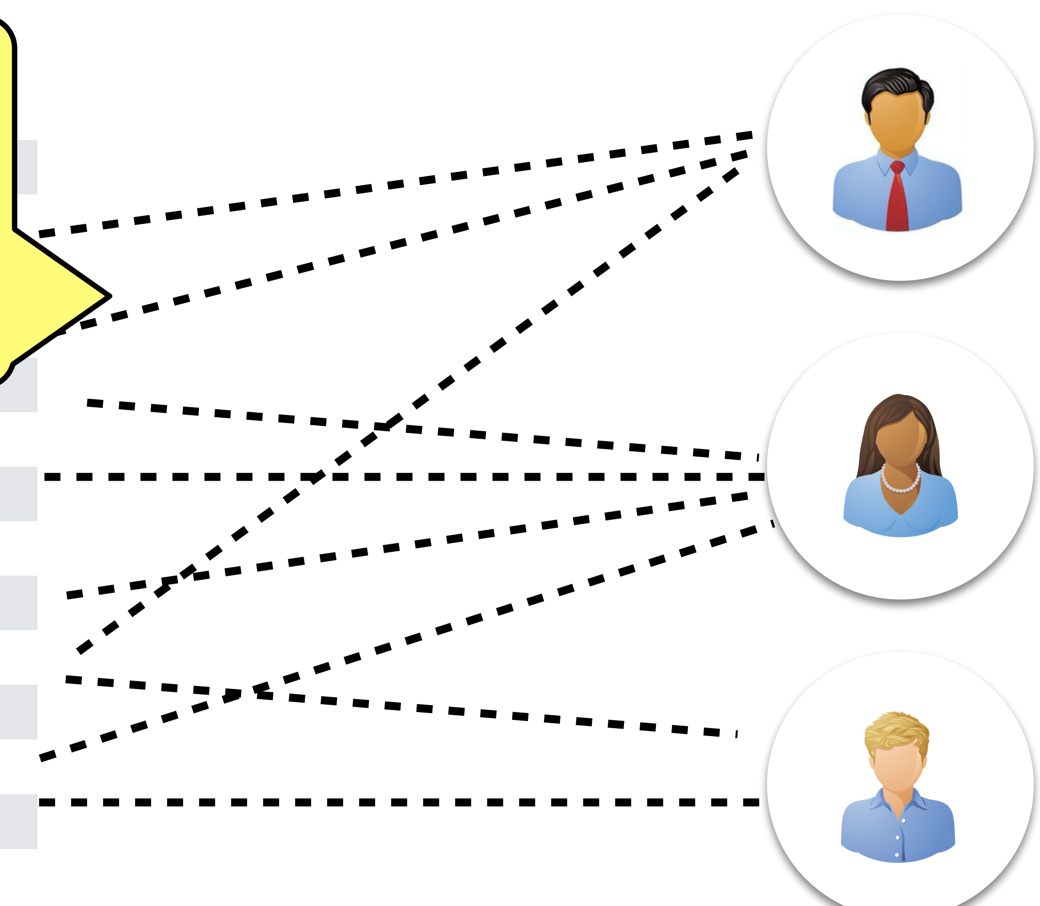
Step 1: Collect all Candidates

This step exploits an incremental graph pattern matcher

Use allocation rules (derived from a TGG) to create *all possible* correspondence links.

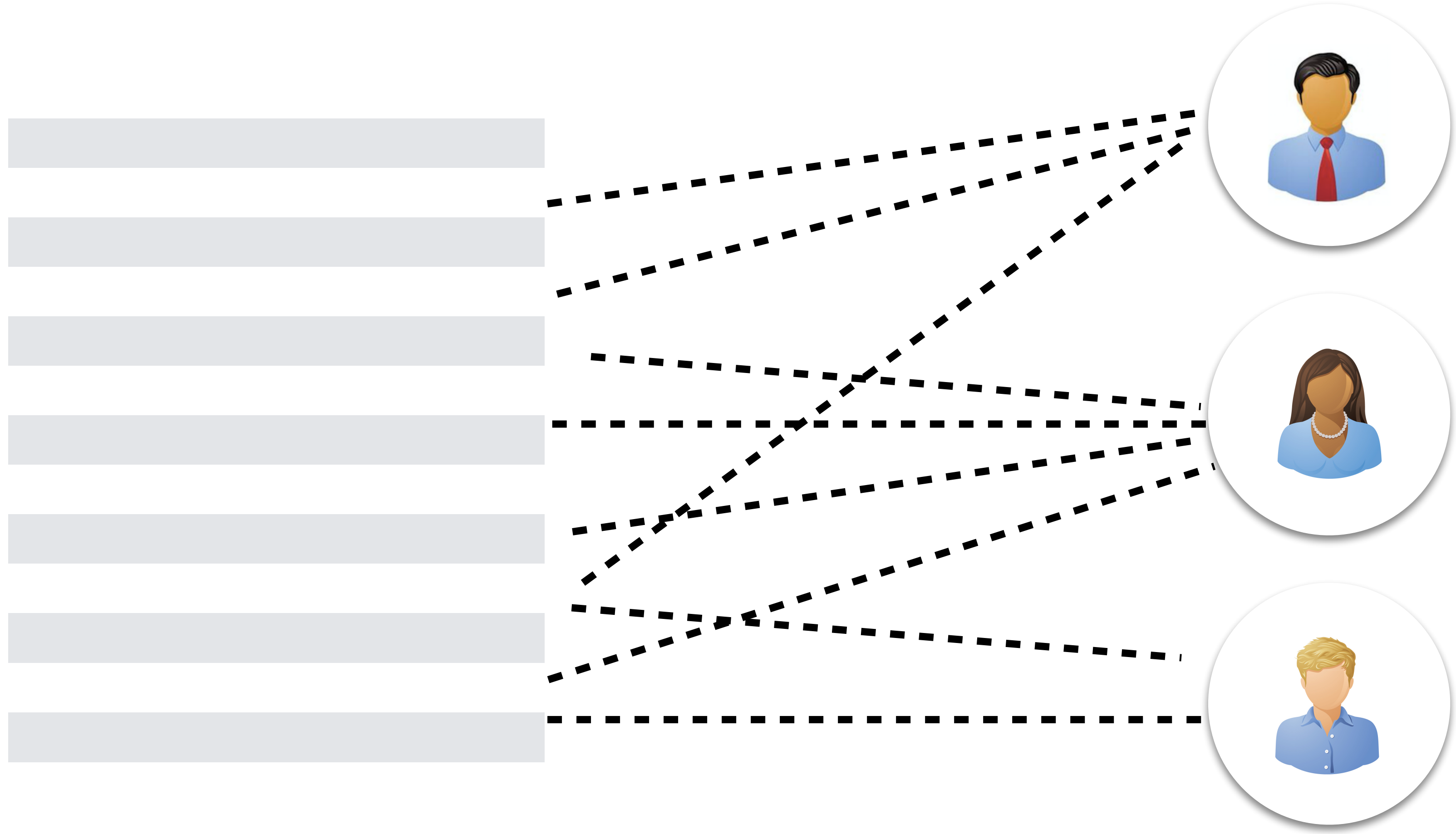


While these links are only *candidates*, they still make sense locally

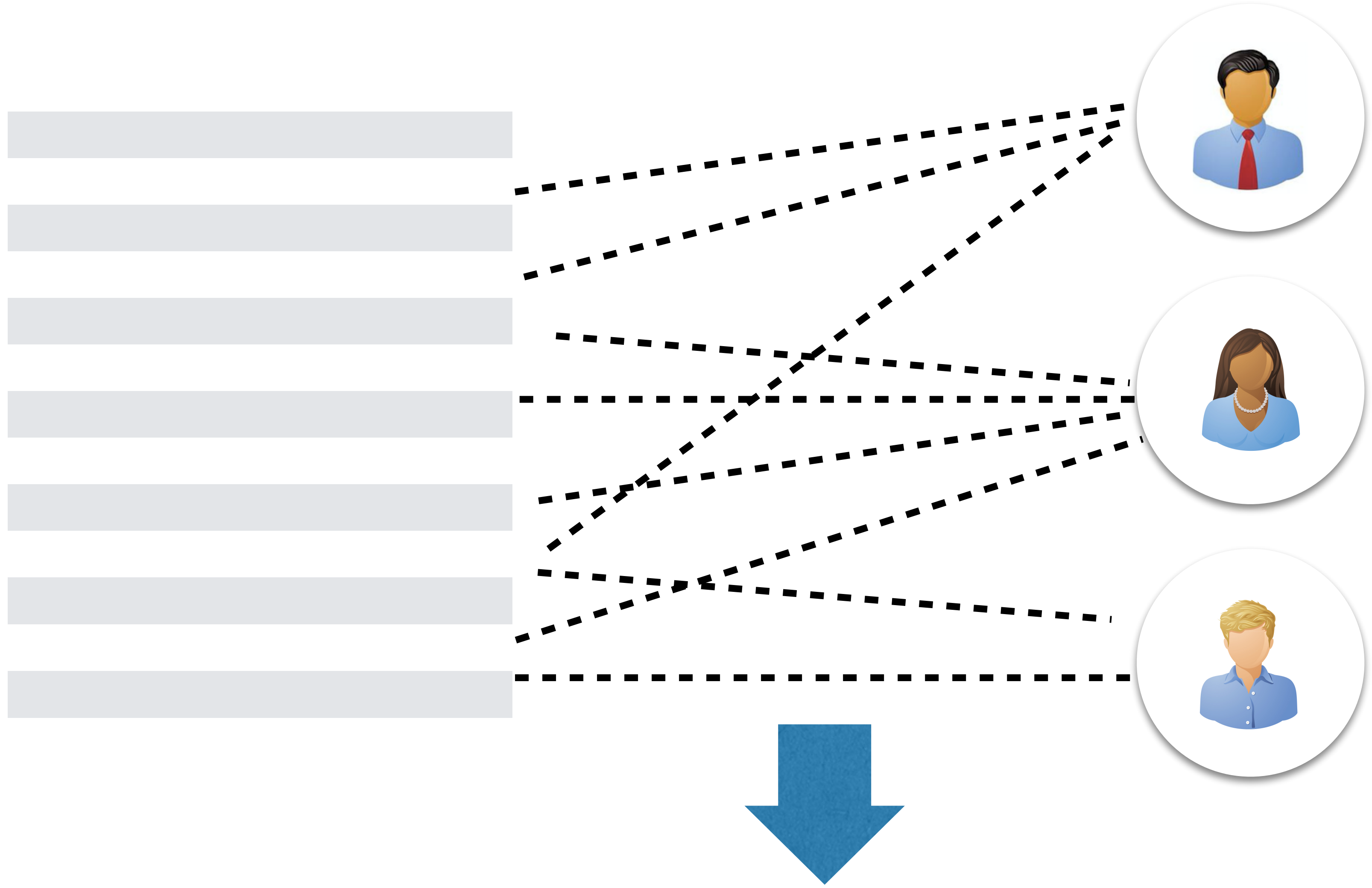


This step requires a necessary and sufficient condition for termination!

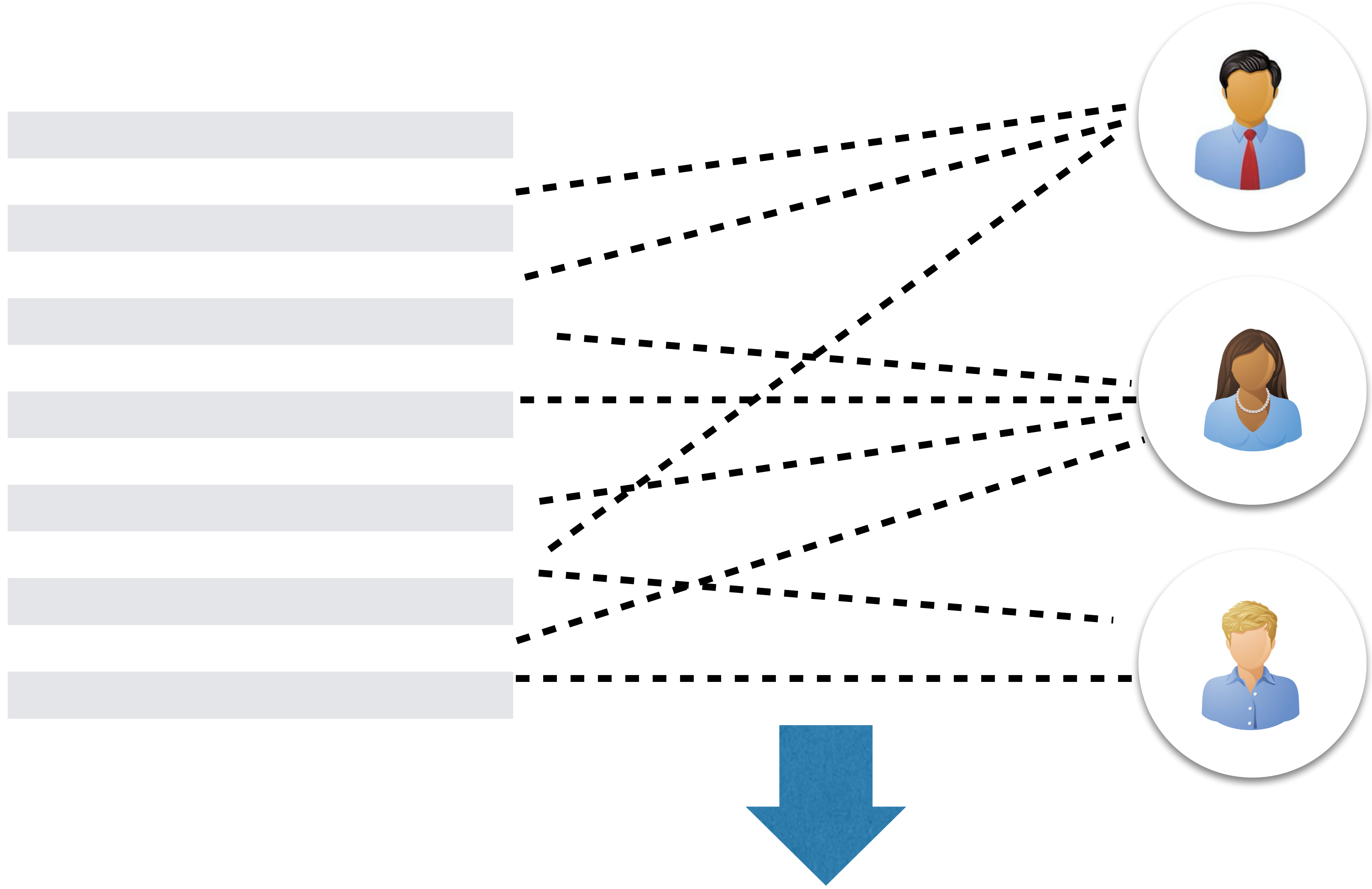
Step 2: Derive ILP



Step 2: Derive ILP



Step 2: Derive ILP



$$\max \quad \vec{c} \cdot \vec{x}$$

$$A\vec{x} \leq \vec{b}$$

Step 2: Derive ILP

$\vec{x} \in \mathbb{Z}_2^n$

Which candidates will be part of the solution?

$$\max \quad \vec{c} \cdot \vec{x}$$

$$A\vec{x} \leq \vec{b}$$

Step 2: Derive ILP

$\overrightarrow{c} \in \mathbb{R}^n$

Domain-specific weights for each candidate

$$\max \overrightarrow{c} \cdot \overrightarrow{x}$$

$$A\overrightarrow{x} \leq \overrightarrow{b}$$

Step 2: Derive ILP

E.g., prefer assigning multiple executions of the same task to the same person

$\overrightarrow{c} \in \mathbb{R}^n$

Domain-specific weights for each candidate

$$\max \overrightarrow{c} \cdot \overrightarrow{x}$$

$$A\overrightarrow{x} \leq \overrightarrow{b}$$

Step 2: Derive ILP

$$\max \quad \vec{c} \cdot \vec{x}$$

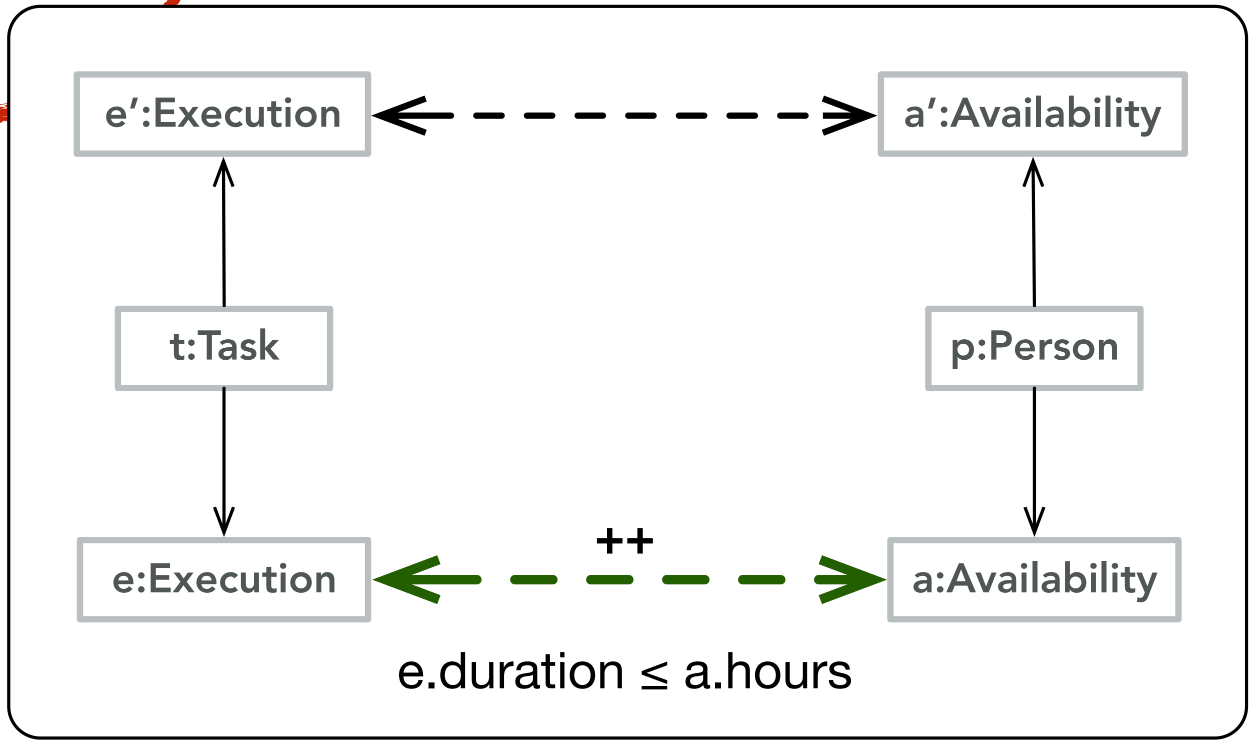
$$A\vec{x} \leq \vec{b}$$

Constraints to ensure that the chosen solution is in the language of the TGG.

Step 2: Derive ILP

$$\max \quad \overrightarrow{c} \cdot \overrightarrow{x}$$

$$A\overrightarrow{x} \leq \overrightarrow{b}$$



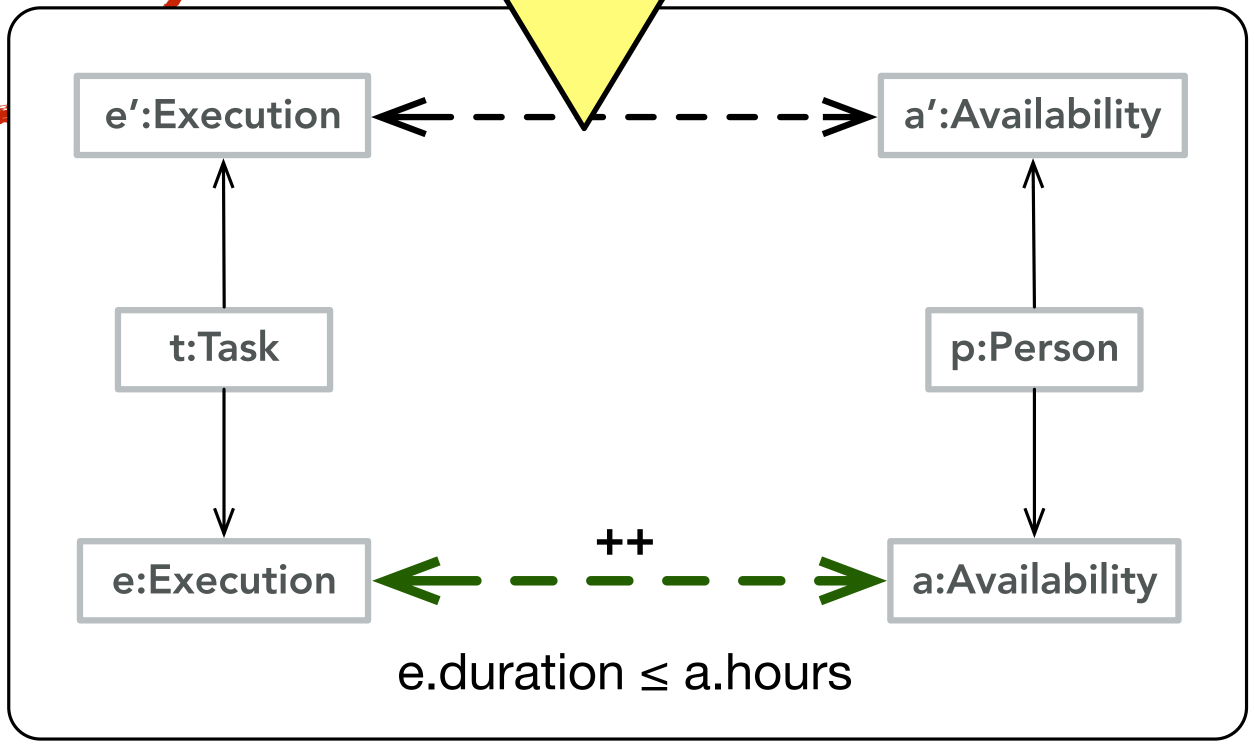
Step 2: Derive ILP

$$\max \quad \vec{c} \cdot \vec{x}$$

$$A\vec{x} \leq \vec{b}$$

E.g, let's assume all candidates for creating this link are:

$$x_1, x_2, x_3$$



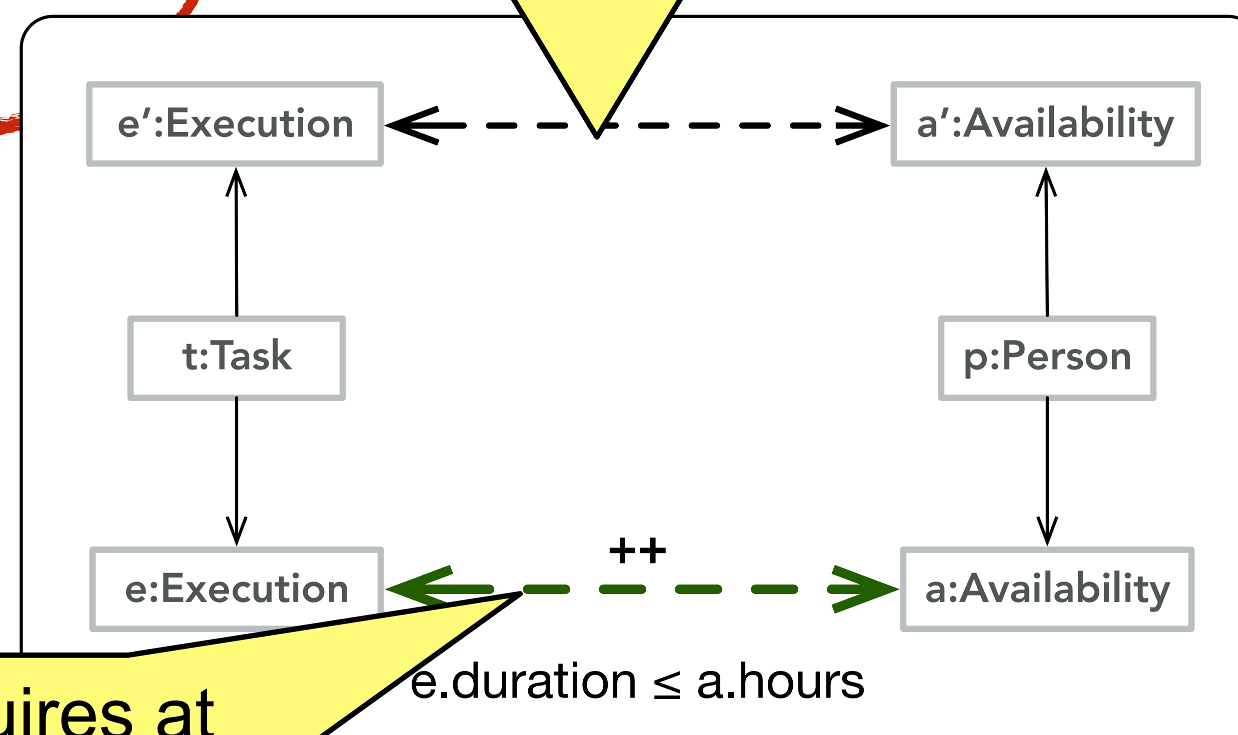
Step 2: Derive ILP

$$\max \ \vec{c} \cdot \vec{x}$$

$$A\vec{x} \leq \vec{b}$$

E.g, let's assume all candidates for creating this link are:

$$x_1, x_2, x_3$$



This candidate requires at least one of them to exist:

$$x_4 \Rightarrow x_1 \vee x_2 \vee x_3$$

$$x_4 \leq x_1 + x_2 + x_3$$

Step 2: Derive ILP

$$\max \quad \vec{c} \cdot \vec{x}$$

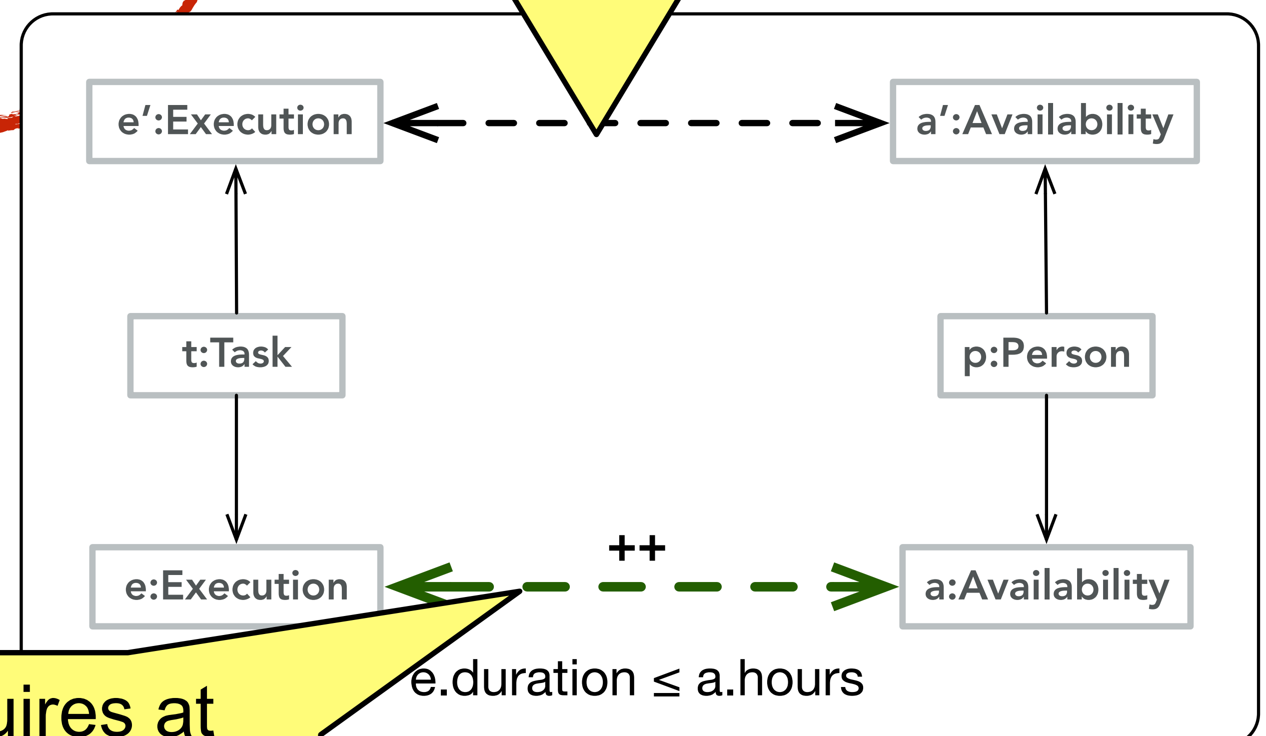
$$A\vec{x} \leq \vec{b}$$

All such inequalities are collected to form:

$$A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^{n}$$

E.g, let's assume all candidates for creating this link are:

$$x_1, x_2, x_3$$



This candidate requires at least one of them to exist:

$$x_4 \Rightarrow x_1 \vee x_2 \vee x_3$$

$$x_4 \leq x_1 + x_2 + x_3$$

Step 3: Solve (Optimise) ILP and Interpret Solution

$$\max \quad \vec{c} \cdot \vec{x}$$

$$A\vec{x} \leq \vec{b}$$

GUROBI
OPTIMIZATION

$$\vec{x}^*$$

Step 3: Solve (Optimise) ILP and Interpret Solution

This step exploits mature ILP solvers

$$\max \quad \vec{c} \cdot \vec{x}$$

$$A\vec{x} \leq \vec{b}$$

GUROBI
OPTIMIZATION

$$\vec{x}^{*}$$

Step 3: Solve (Optimise) ILP and Interpret Solution

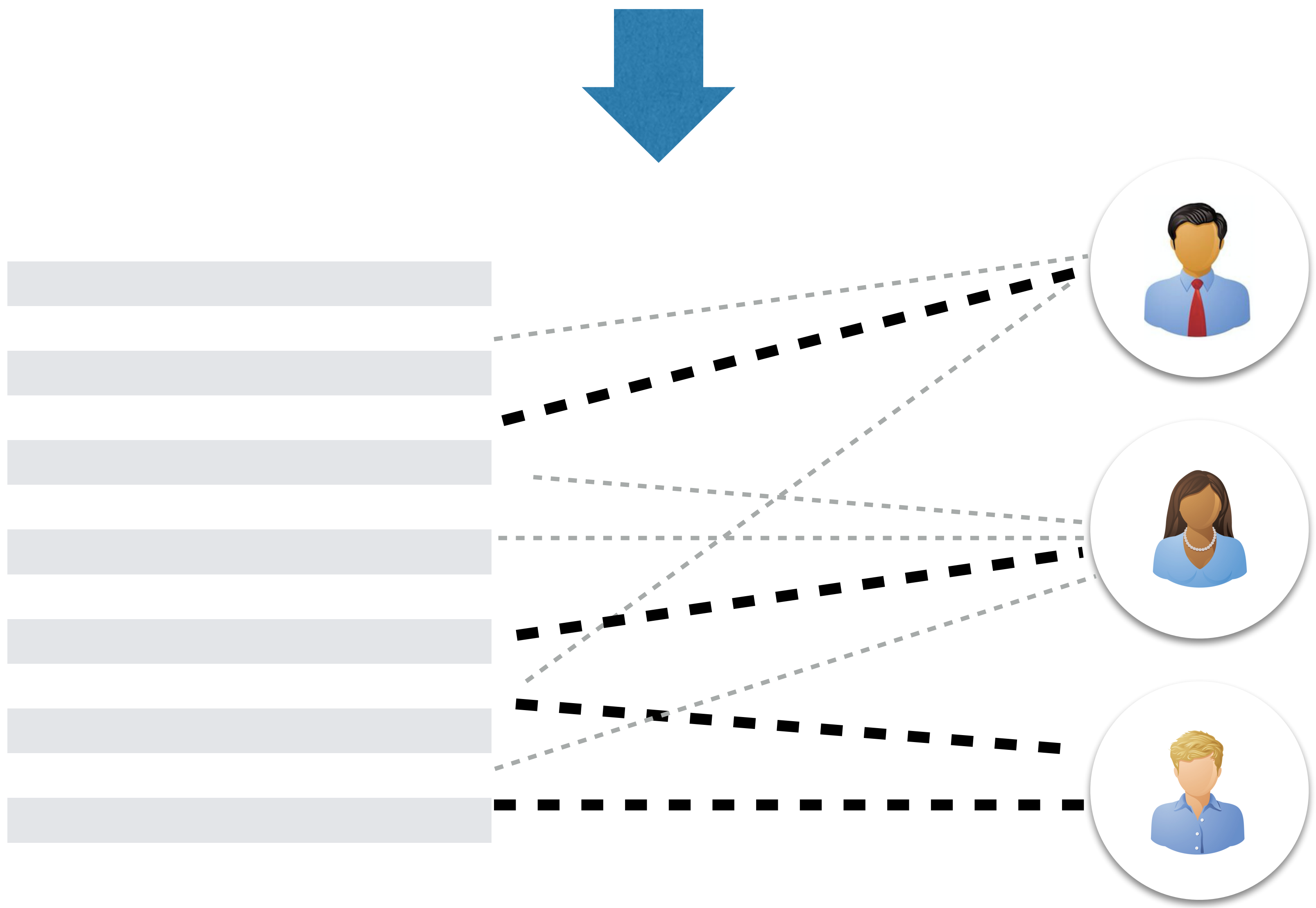
$$\overrightarrow{x}^*$$

Step 3: Solve (Optimise) ILP and Interpret Solution

$$\vec{x}^*$$

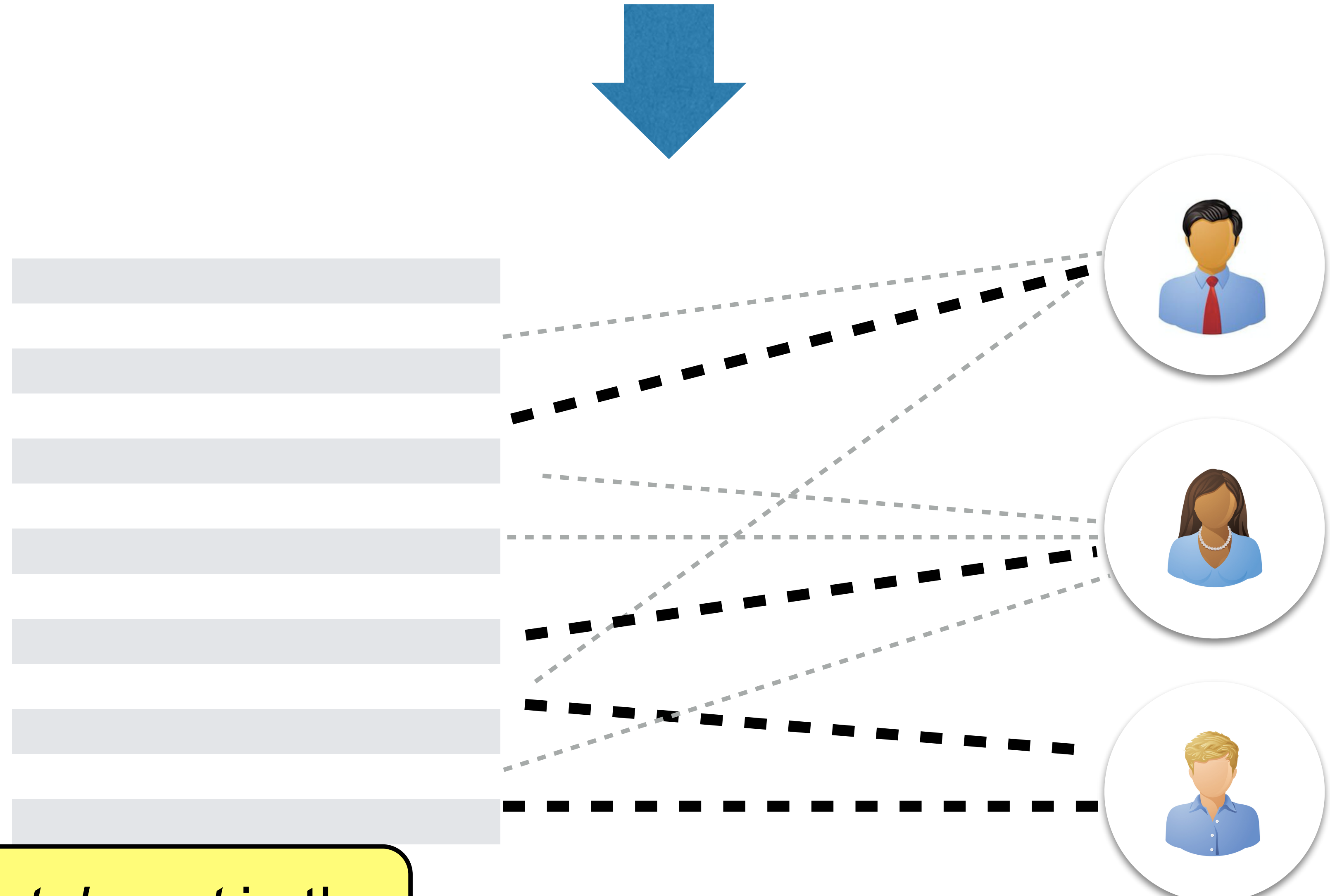
Step 3: Solve (Optimise) ILP and Interpret Solution

$$\overrightarrow{x}^*$$



Step 3: Solve (Optimise) ILP and Interpret Solution

$$\overrightarrow{x}^*$$



Our approach is *tolerant* in the sense that we can determine partial solutions (all variables are set to 0 in the worst case)

Ongoing and Future Work

Operation	Source	Corr	Target
CC	mark	create	mark
CO	mark	mark	mark
FWD_OPT	mark	create	create
BWD_OPT	create	create	mark

Ongoing and Future Work

Operation	Source	Corr	Target
CC	mark	create	mark
CO	mark	mark	mark
FWD_OPT	mark	create	create
BWD_OPT	create	create	mark

Our initial focus (**C**onsistency **C**heck via correspondence link creation)

Ongoing and Future Work

Operation	Source	Corr	Target
CC	mark	create	mark
CO	mark	mark	mark
FWD_OPT	mark	create	create
BWD_OPT	create	create	mark

Check **O**nly:
Check existing triple for consistency

Ongoing and Future Work

Operation	Source	Corr	Target
CC	mark	create	mark
CO	mark	mark	mark
FWD_OPT	mark	create	create
BWD_OPT	create	create	mark

Normal initial (batch) fwd and bwd transformations; but now complete, tolerant, and optimal wrt. to an objective function

Ongoing and Future Work

Operation	Source	Corr	Target
CC	mark	create	mark
CO	mark	mark	mark
FWD_OPT	mark	create	create
BWD_OPT	create	create	mark

All definitions, proofs, and most parts of the implementation can be formulated generically and configured for each case using the entries in this table!

Try it out! www.emoflon.org

eMoflon