Karlsruhe Institute of Technology

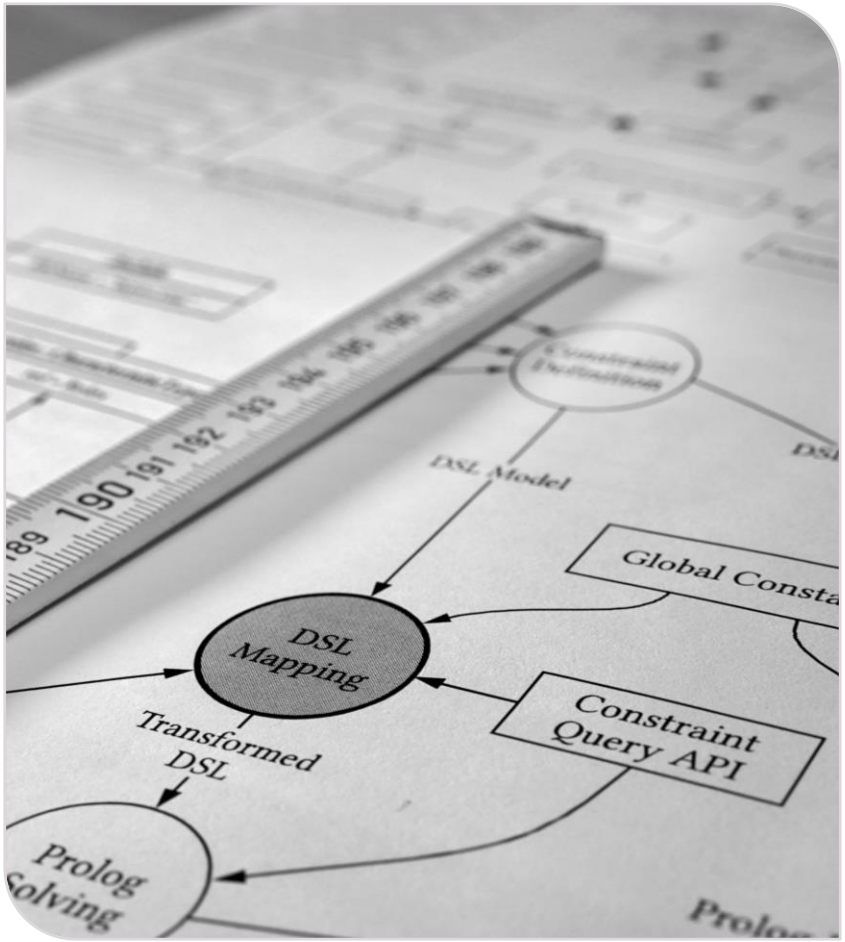
FluidTrust
Trust by fluid access control in Industry 4.0

Modeling Data Flow Constraints for Design-Time Confidentiality Analyses

@ 18TH IEEE INTERNATIONAL CONFERENCE ON SOFTWARE ARCHITECTURE, ICSA'21

Sebastian Hahner, Stephan Seifermann, Robert Heinrich, Maximilian Walter, Tomáš Bureš, Petr Hnětynka

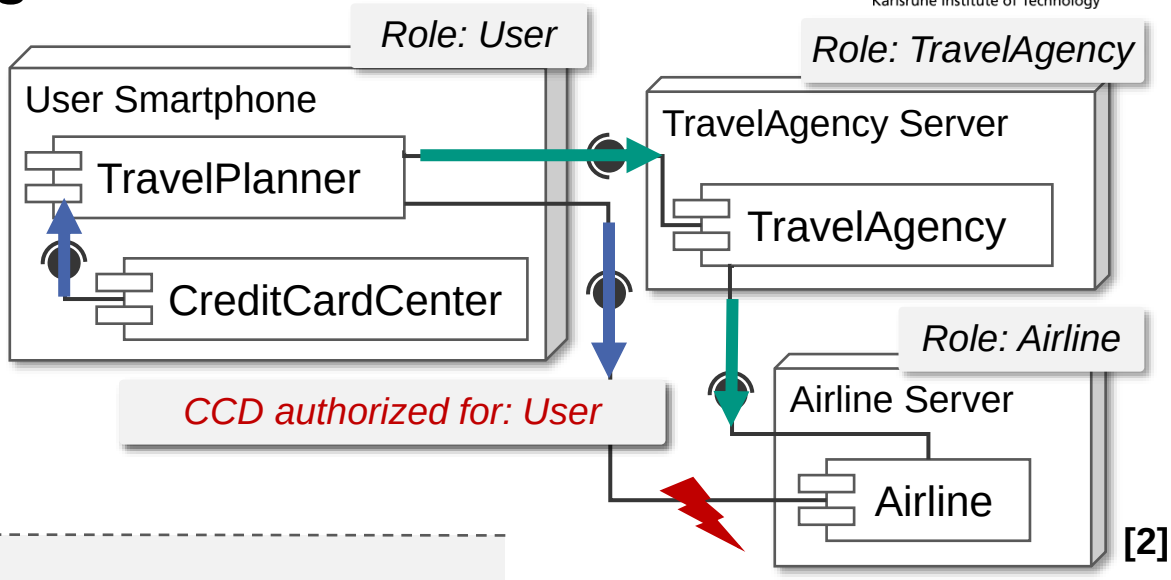
Short version



Modeling Data Flow Constraints for Design-Time Confidentiality Analyses

Karlsruhe Institute of Technology

Data flow-based design-time analyses identify *confidentiality violations* in architectural models **[1]**



[2]

***Constraint**: „Data is only allowed to flow to entities with authorized roles“*

```
constraint EnforceRBAC {
 data.attribute.accessRights.$rights{} NEVER FLOWS
 node.property.roles.$roles{} WHERE
 isEmpty(intersection(rights,roles))}
```

Contributions: DSL, mapping to the analysis formalism, mapping of analysis results

[1] Seifermann et al. “Data-Driven Software Architecture for Analyzing Confidentiality,” in *2019 IEEE International Conference on Software Architecture (ICSA)*, 2019
[2] Katkalov et al. “Model-Driven Development of Information Flow-Secure Systems with IFlow,” in *2013 International Conference on Social Computing*, 2013

Karlsruhe Institute of Technology

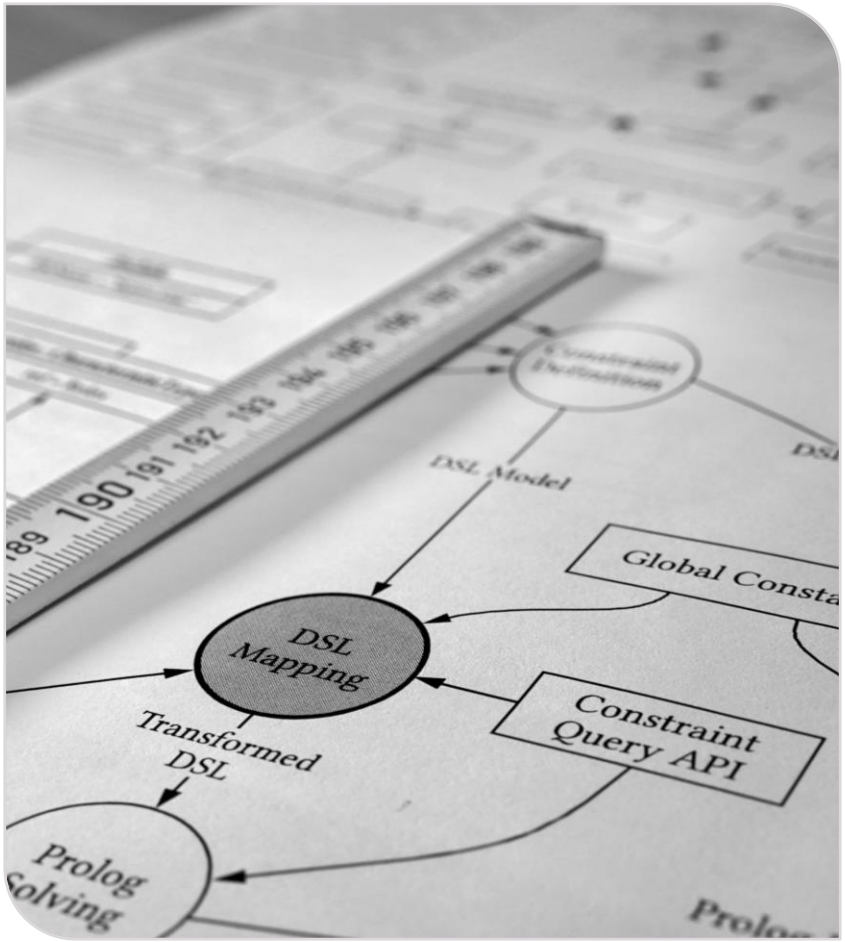
FluidTrust
Trust by fluid access control in Industry 4.0

Modeling Data Flow Constraints for Design-Time Confidentiality Analyses

@ 18TH IEEE INTERNATIONAL CONFERENCE ON SOFTWARE ARCHITECTURE, ICSA'21

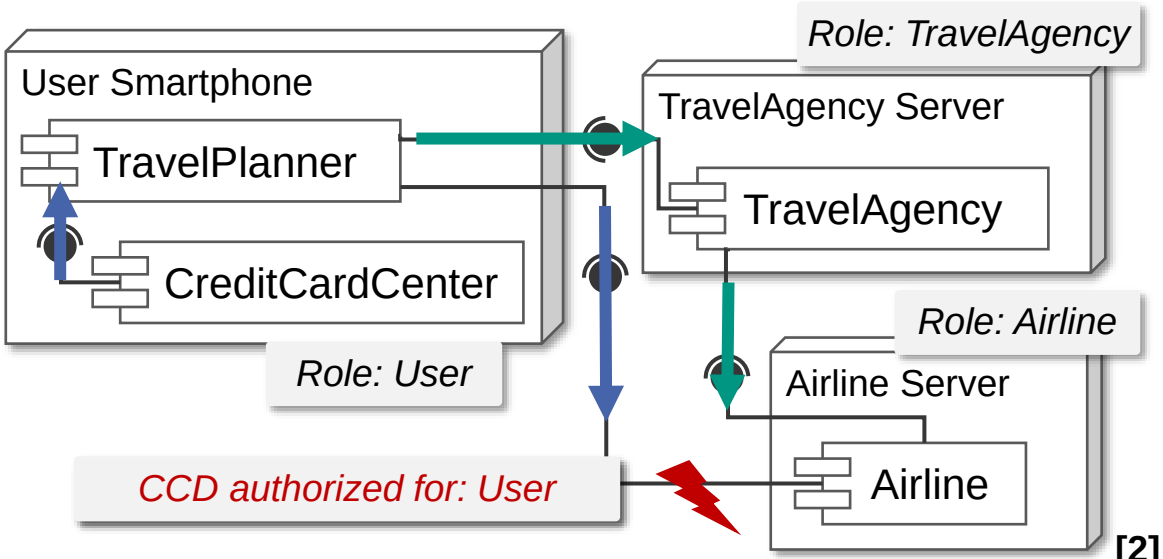
Sebastian Hahner, Stephan Seifermann, Robert Heinrich, Maximilian Walter, Tomáš Bureš, Petr Hnětynka

Long version



Data Flow-Based Confidentiality Analysis

Data flow-based design-time analyses identify *confidentiality violations* in architectural models **[1]**



[2]

***Constraint**: „Data is only allowed to flow to entities with authorized roles“*

[1] Seifermann et al. “Data-Driven Software Architecture for Analyzing Confidentiality,” in *2019 IEEE International Conference on Software Architecture (ICSA)*, 2019
[2] Katkalov et al. “Model-Driven Development of Information Flow-Secure Systems with IFlow,” in *2013 International Conference on Social Computing*, 2013

Modeling Data Flow Constraints

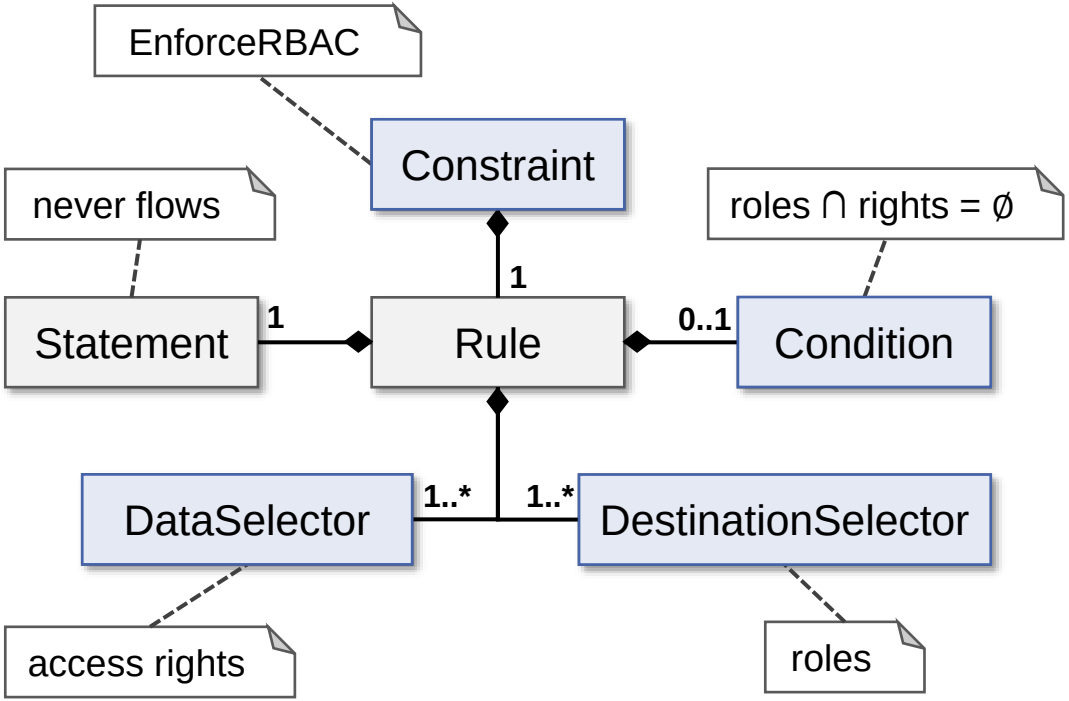
- *Gap*: Constraints are described directly by using the analysis formalism
- *Contribution*: A domain-specific language on architectural abstraction level, mapping to the analysis formalism, and mapping of analysis results
- *Benefit*: Use of known terminology, easier to use and higher productivity

***Constraint**: „Data is only allowed to flow to entities with authorized roles“*

DSL

```
constraint EnforceRBAC {
 data.attribute.accessRights.$rights{} NEVER FLOWS
 node.property.roles.$roles{} WHERE
 isEmpty(intersection(rights,roles))}
```

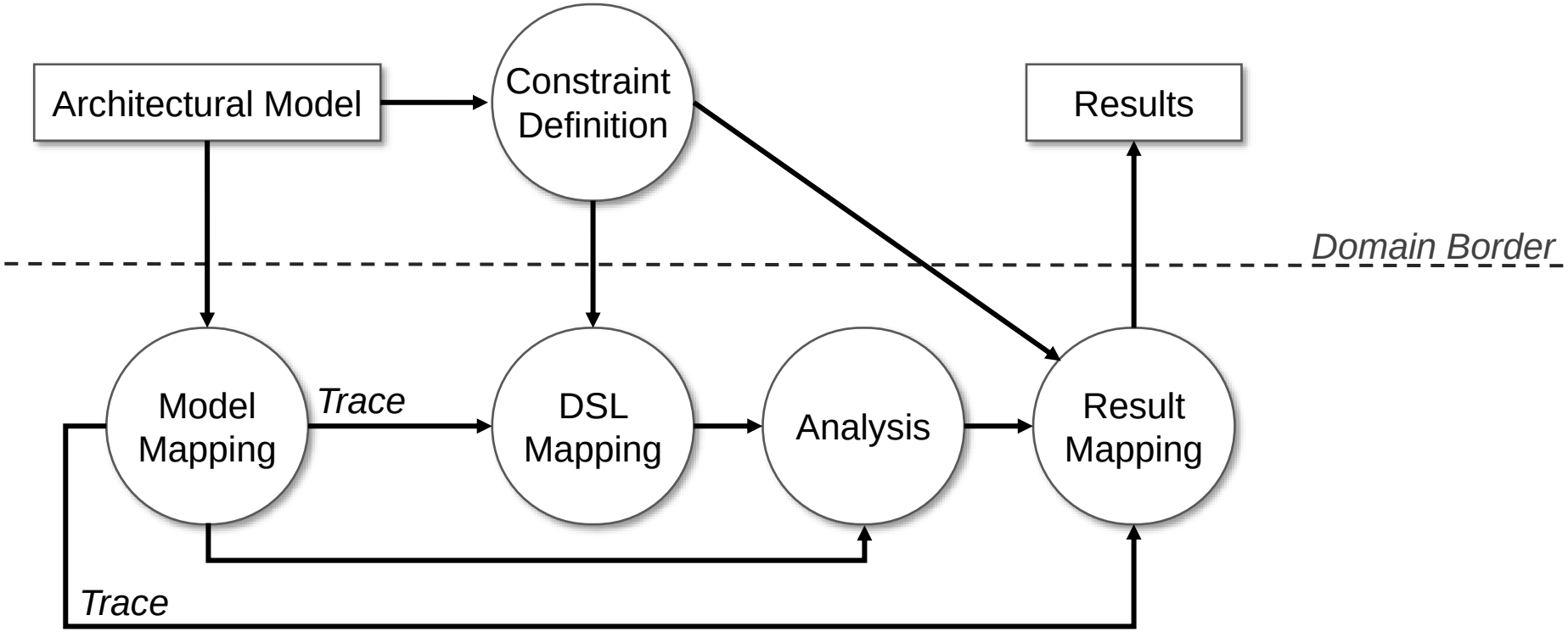
DSL for Data Flow Constraints



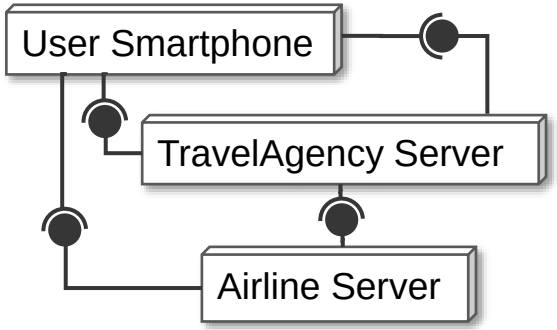
***Constraint**: „Data is only allowed to flow to entities with authorized roles“*

```
constraint EnforceRBAC {
 data.attribute.accessRights.$rights{}
   NEVER FLOWS
 node.property.roles.$roles{} WHERE
 isEmpty(intersection(rights,roles))}
```

Mapping of Architecture and Constraints

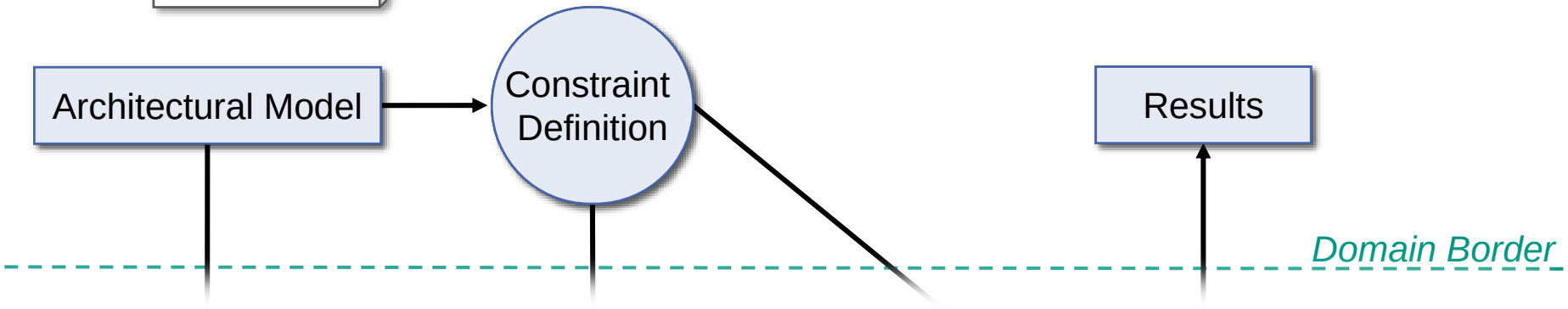


Mapping of Architecture and Constraints




```
constraint EnforceRBAC {
 data.attribute.accessRights.$rights{}
   NEVER FLOWS
 node.property.roles.$roles{} WHERE
 isEmpty(intersection(rights,roles))}
```

```
CONSTRAINT VIOLATIONS
1. Parameter 'CCD' not allowed
- Call Stack: 'getDetails','bookFlight'
- Variables:
   - 'rights' set to 'User'
   - 'roles' set to 'Airline'
```



Evaluation

- Expressiveness
  - Do the available DSL concepts allow versatile data flow constraint definitions?
  - Scenario and constraint evaluation based on case studies used in related work
  - ⇨ The expressiveness is sufficient for most of the evaluated constraints
- Usability
  - Does the DSL provide abstraction from the analysis formalism? Does the DSL hide complexity?
  - Effort measurement of change scenarios, domain border evaluation
  - ⇨ The DSL requires less effort and hides accidental complexity by requiring no knowledge about the underlying analysis
- Correctness
  - Is the mapping correct? Are the analysis results equivalent?
  - Formal correctness proof, Comparison of analysis results
  - ⇨ The mapping is correct
  - ⇨ The analysis reaches a recall of 100% while maintaining a precision of 92%

Related Work

- *Direct analyses* use architectural models directly without transformation **[1], [2], [3]**
  - ⇨ These approaches do not consider the gap between architecture and analysis
- *Indirect analyses* transform the architecture into an analysis formalism **[4], [5], [6]**
  - ⇨ These approaches do not provide a generic solution for bridging the abstraction gap
  - ⇨ Usually do not report on the mapping of analysis results

[1] Almorsy et al. “Automated software architecture security risk analysis using formalized signatures”, in *35th International Conference on Software Engineering (ICSE)*, 2013
[2] Tuma et al. “Automating the Early Detection of Security Design Flaws”, *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems,* 2020
[3] Yu et al. “Formal Software Architecture Design of Secure Distributed Systems”, *SEKE*, 2003
[4] Katkalov et al. “Model-Driven Development of Information Flow-Secure Systems with IFlow,” in *2013 International Conference on Social Computing*, 2013
[5] Gerking et al. “Model Checking the Information Flow Security of Real-Time Systems”, in *Engineering Secure Software and Systems*, 2018
[6] Jürjens et al. “UMLsec: Extending UML for Secure Systems Development”, in *UML 2002 — The Unified Modeling Language*, 2002

Conclusion

- Domain-specific language to define data flow constraints in the architectural domain which bridges the gap between definition and analysis
- Mapping to the analysis formalism and mapping of analysis results which are compliant to the architecture transformation
- Shall enhance the formulation capabilities and productivity of software architects
- In future work, we aim to increase the DSL's expressiveness