



# A Middleware Model in Alloy

*for Supply Chain-Wide Agent Interactions*

R. Haesevoets - D. Weyns - M. Torres  
A. Helleboogh - T. Holvoet - W. Joosen

# Robrecht Haesevoets

[robrecht.haesevoets@cs.kuleuven.be](mailto:robrecht.haesevoets@cs.kuleuven.be)



**DistriNet**  
RESEARCH GROUP

# Research Project\*

Industrial partners

Managing collaborations with agents

Supply Chain Management domain

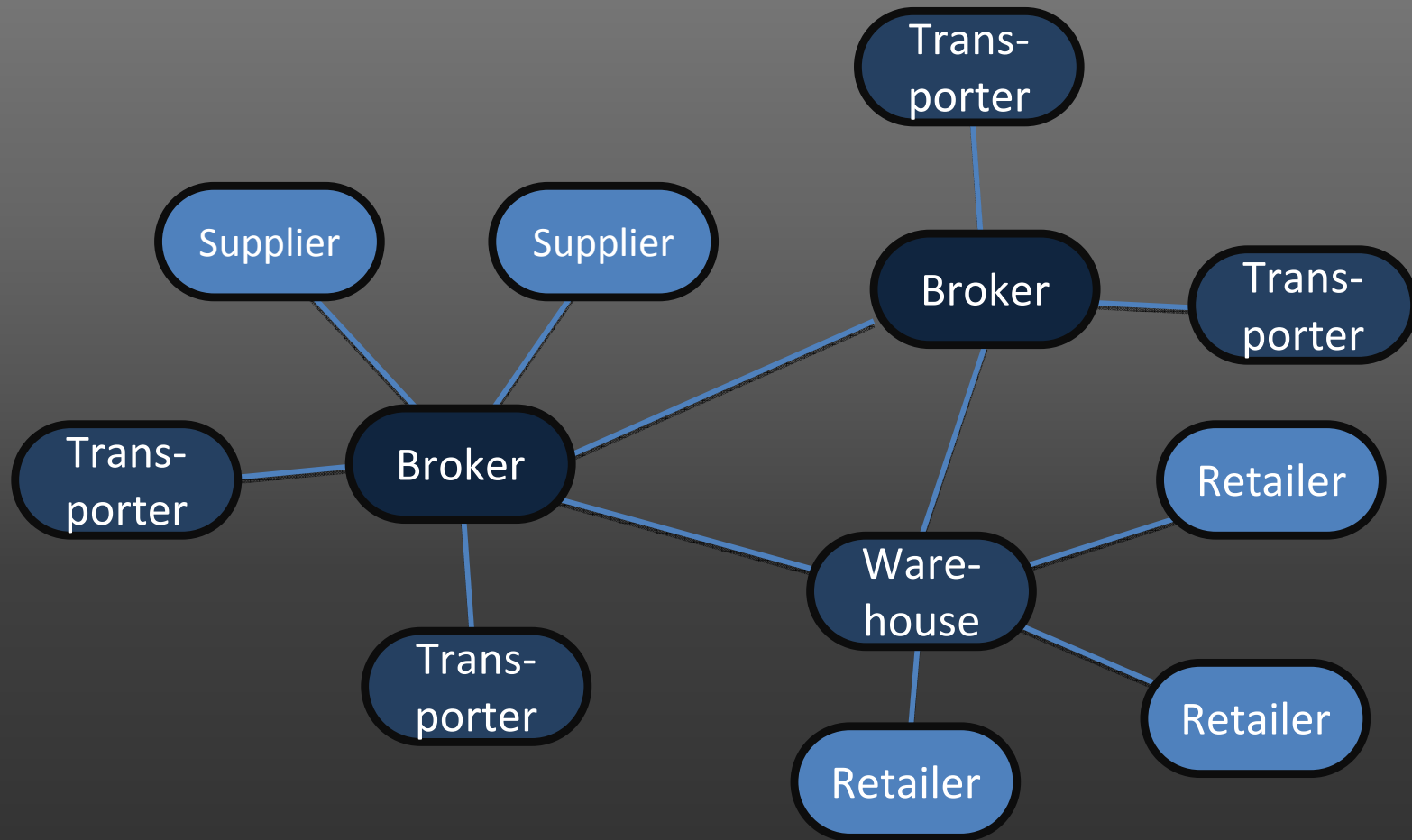
## 10<sub>min</sub> Presentation

Controlled supply chain-wide interactions

Organization middleware

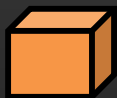
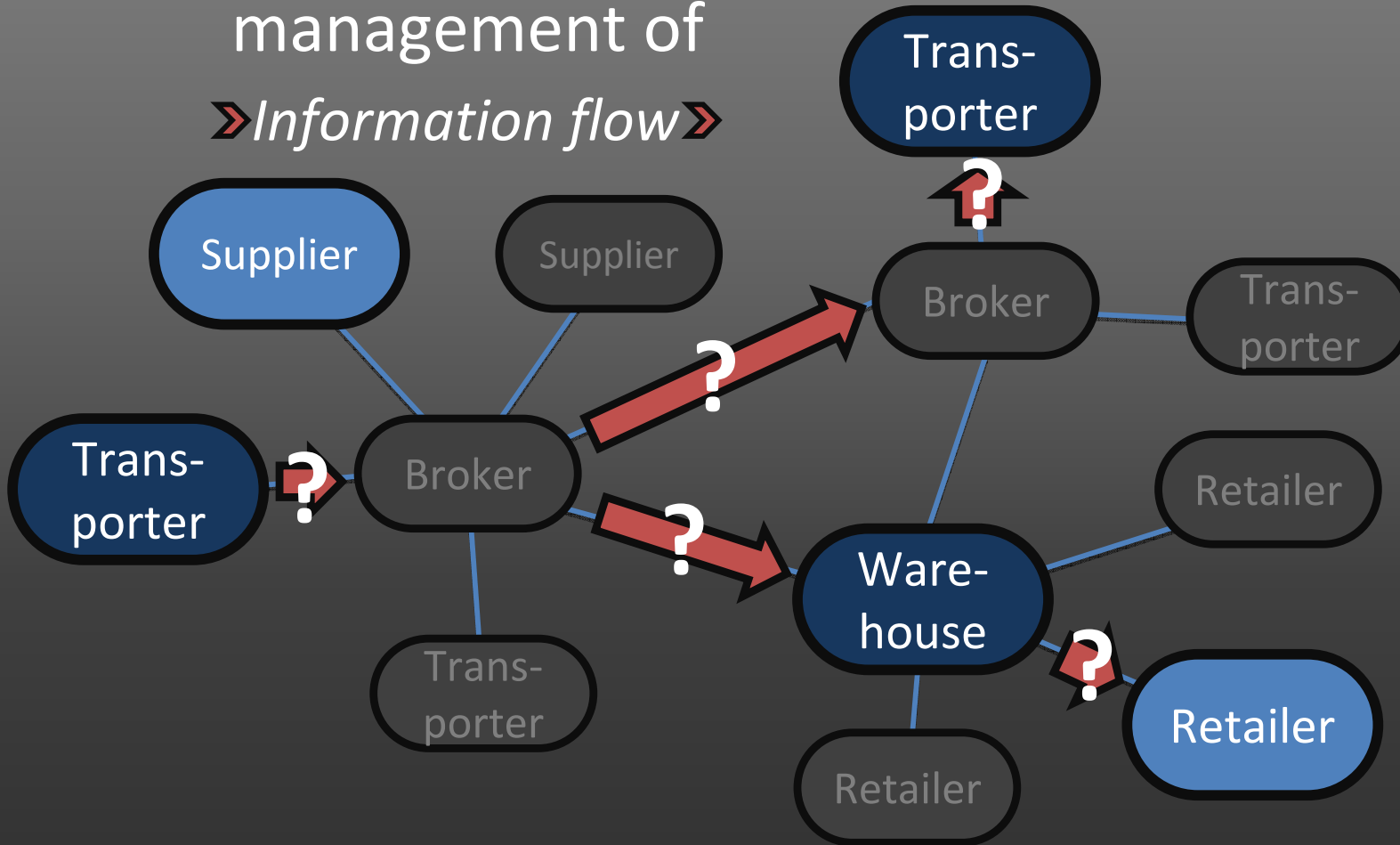
Simple model + Alloy

\* DiCoMas: Distributed Collaboration using Multi-agent System Architectures  
<http://distrinet.cs.kuleuven.be/projects/dicommas/index.html>

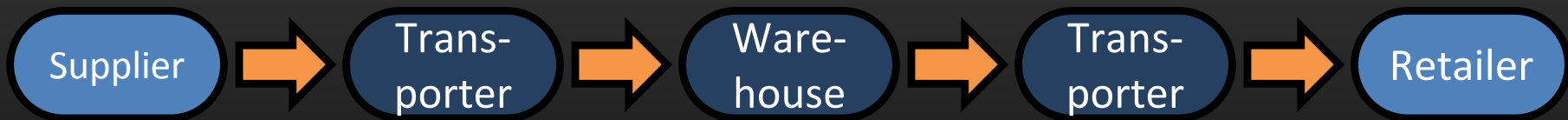


No overall view or management of

» *Information flow* »



**DELAY!**

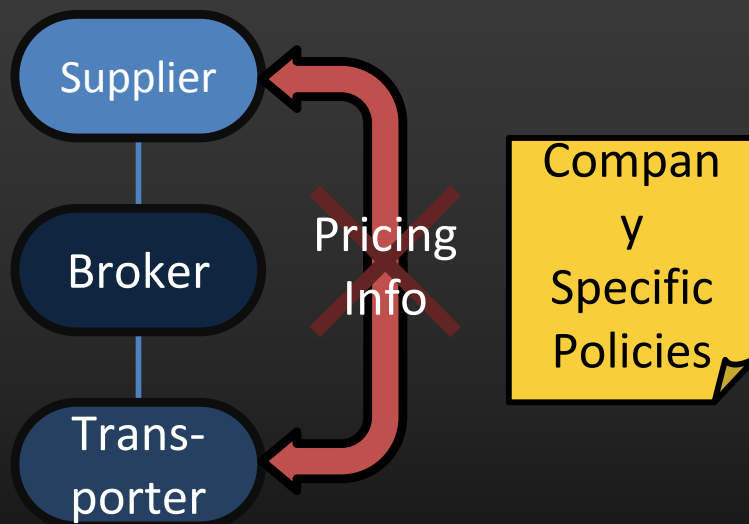


» *Product Flow* »



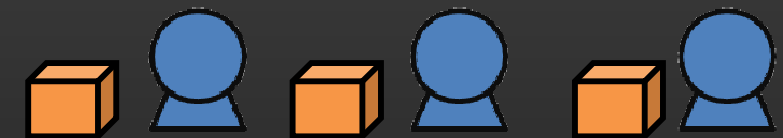
Distributed Interaction Platform

Need for controlled interactions



Finding correct interaction partners

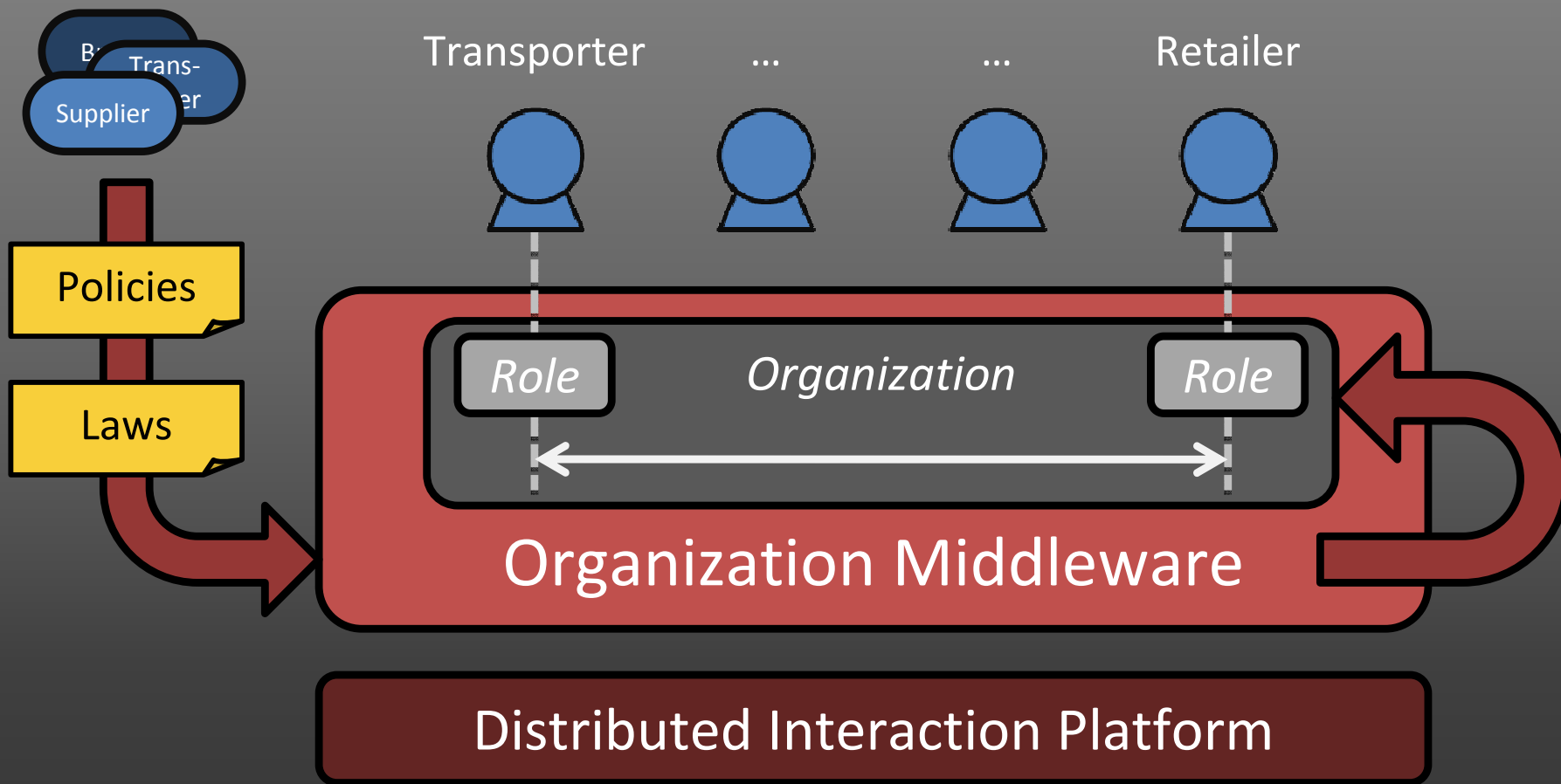
*Dynamic context*



Define desired interaction scope

*Declarative*

Laws



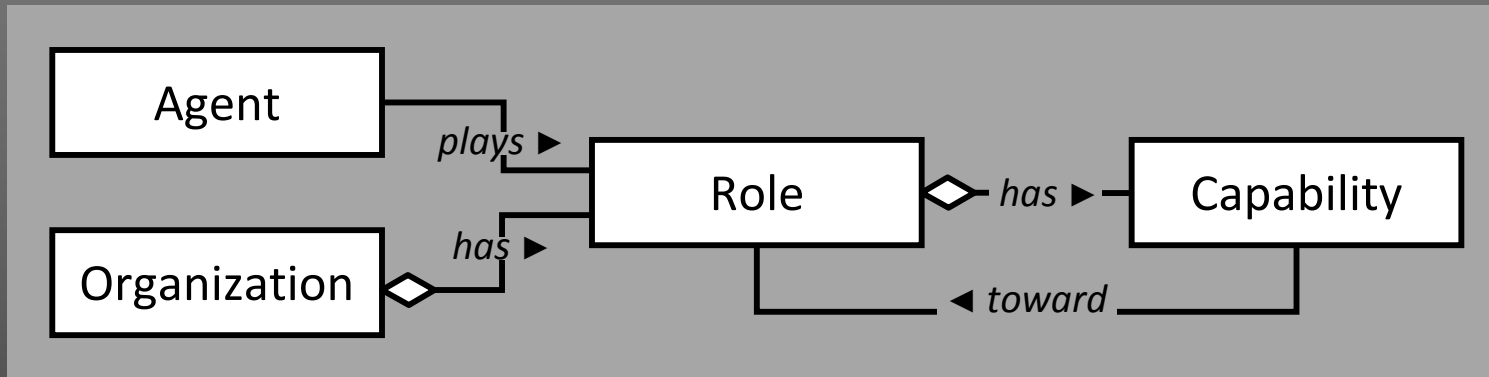
## Additional layer

### Organization and roles

Structure, control and limit interactions

Managed by middleware, not by agents

# Simple organization model



## + Alloy\*

*Formal modeling language - based on first-order logic*

## + Alloy Analyzer\*\*

*Simulate models - Check models in Limited Scope*

---

**=** Explore concepts – guarantees  
*for controlled interactions*

## Specify Universe

sig Agent{...}

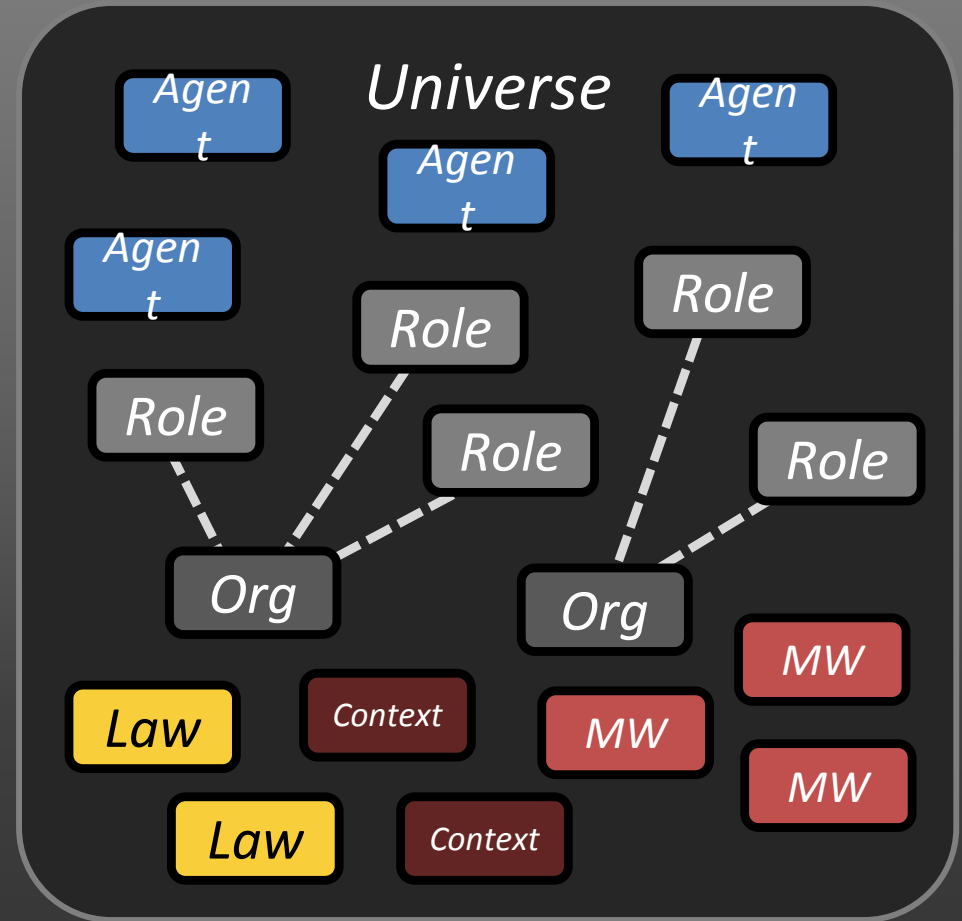
sig Role{...}

```
sig Organization{
  roles:set Role
}
```

sig Context{...}

sig Law{...}

sig MiddlewareModel{...}



## Search space for Analyzer

*To find instances and counter-examples*

## Abstract Model of Middleware

*Check assertions – Check Laws and Policies*

## Define assertions

```
assert Property2{
  all mw:MiddlewareModel, disj a1,a2,a3:Agent |
    !indepContractPath[a1,a2,a3,mw.context] and
    (all c1,c2:(client+provider).a3 |
      no Flow->c1->c2 & mw.context.flowPolicies) implies
      no r1,r2:mw.orgs.roles | some r2->Flow & r1.capabilities
      and r1.agent = a1 and r2.agent = a2
}
```

check Property2 for 10  *Limited scope*

 *Check universe for counter-examples*

If no counter-examples found

 *Assertion holds in scope*

# Summary

## Reusable organizational abstractions

- For controlled supply chain-wide interactions

## Middleware Model + Alloy

- Check assertions, specific laws and policies

## Even with simple concepts

- Control interactions and provide some guarantees

## Future work

- Execution semantics of laws
- Dynamically group agents according to context

# Discussion Topics

## Formal methods in AOSE

- Place in engineering process
- Model checking at run-time?

## Organization middleware

- Agent-managed versus Middleware-managed
- Autonomy of Agents
- Norms versus Laws

# References

- Preist et al. *Automated business-to-business integration of a logistics supply chain using semantic web services technology*. LNCS, 2005
- H. Stadtler. *Supply chain management and advanced planning: basics, overview and challenges*. European Journal of Operational Research, 2005
- V. Dignum. *Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. Information Science Reference, 2009
- Weyns et al. *The MACODO Middleware for Context-Driven Dynamic Agent Organizations*. ACM Trans-action on Autonomous and Adaptive Systems, 2010.
- \* D. Jackson. *Software Abstractions: logic, language, and analysis*. The MIT Press, 2006
- \*\* <http://alloy.mit.edu/community/>