

Specification and implementation methods for swarm robot scenarios

Martin Scheerer



Outline

- ▶ Part I:
 - ▶ Formal Notation
 - ▶ Scenario Bucket Brigade
 - ▶ Specification of Bucket Brigade
- ▶ Part II:
 - ▶ System Overview
 - ▶ Database
 - ▶ Evaluation Software
 - ▶ Results



Formal Notation – What & Why?

▶ What?

- ▶ scenario goals
- ▶ NOT scenario implementation

▶ Why?

- ▶ exact and unambiguous specification
- ▶ better comparability of implementations
- ▶ reuse of code for evaluation and implementation



Formal Notation

- ▶ virtual world, mathematical description
- ▶ vector space W , in general euclidean plane \mathbf{R}^2
- ▶ environment E
- ▶ objects O
- ▶ robots R subset of O
- ▶ properties for objects, ex.: $\text{pos}: O \rightarrow W$
- ▶ properties for environment, ex.: $f_{pher}: W \rightarrow [0,1]$

Scenario Bucket Brigade

- ▶ transportation of food
- ▶ 2 classes of robots: fast robots, slow robots
- ▶ transfer of food:
 - when a loaded slow robot meets a fast and empty robot
- ▶ locations of food source and nest are known
 - scenario goal:
 - ▶ maximize throughput
 - ▶ use transfer whenever possible



Specification Bucket Brigade

- ▶ throughput: counting events (simple)
- ▶ transfers: use *Meeting* from the notation:

- ▶ **precondition:**

general: $d(r_1, r_2) < \min \{ d_{\text{comm}}(r_1), d_{\text{comm}}(r_2) \}$

specific: $(f_{\text{size}}(r_1) > f_{\text{size}}(r_2)) \ \&\&$

$\text{NOT}(r_{\text{loaded}}(r_1)) \ \&\& \ r_{\text{loaded}}(r_2)$

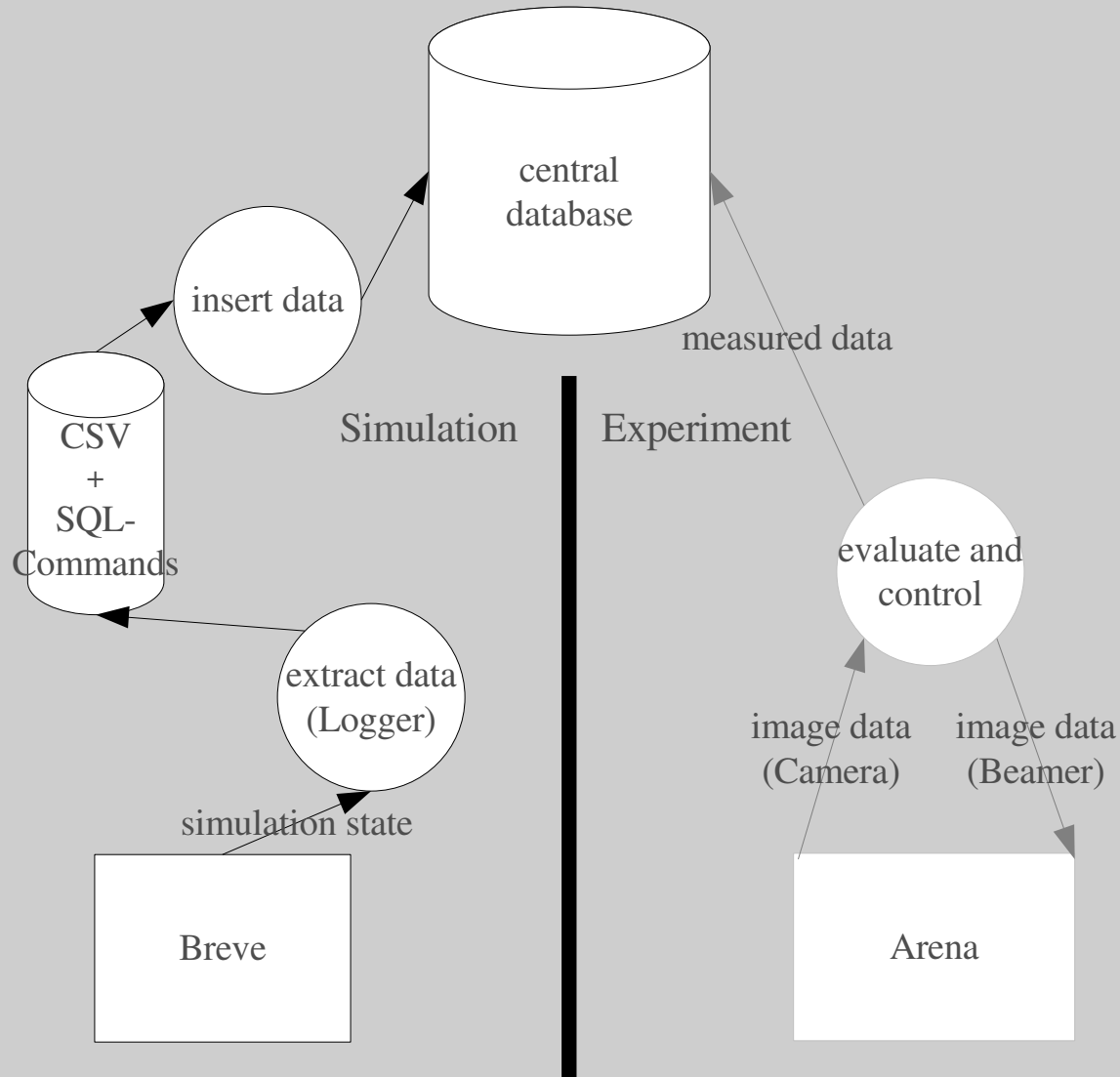
- ▶ **postcondition:**

- ▶ $r_{\text{loaded}}(r_1) \ \&\& \ \text{NOT}(r_{\text{loaded}}(r_2))$

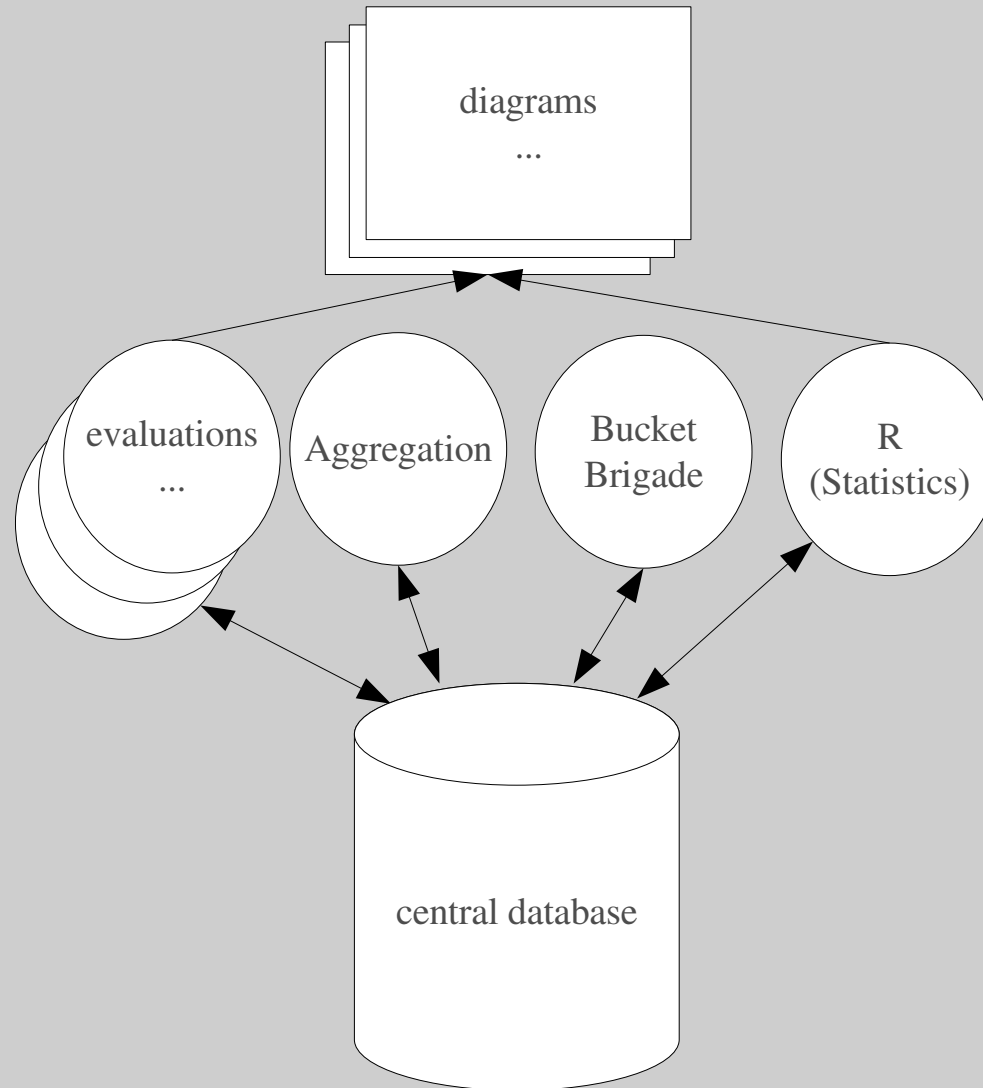
Part II: Implementation



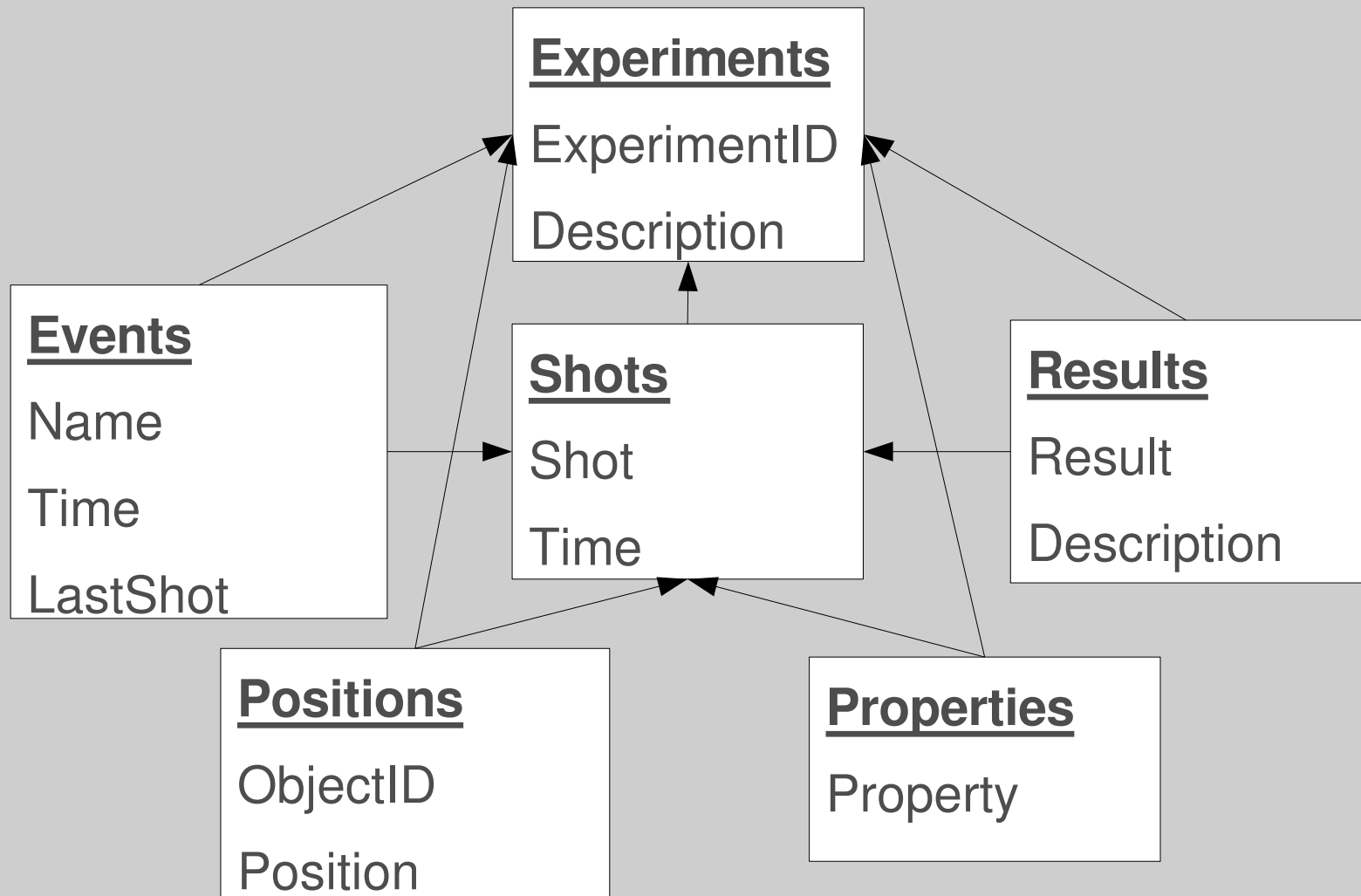
System Overview 1



System Overview 2



Database - Structure



- ▶ every table contains ID, changeDate and other fields

Evaluation: Bucket Brigade Simulations

- ▶ 2 Parts:
 - ▶ R with SQL: counting data by different criteria
 - ▶ objects transported
 - ▶ successful transfers
 - ▶ division of labor between slow and fast robots
 - ▶ Java: analyses *Meetings* by checking pre- and postcondition.
 - ▶ missed transfers
 - ▶ time of transfer



Java evaluation

- ▶ used technology:

- ▶ SQL Spatial Extension, ex.:

```
INSERT INTO geo
```

```
VALUES (GeomFromText ( 'POINT (1 1) ' ) )
```

- ▶ Java Topology Framework (JTS)

- ▶ 2D spatial predicates and functions

- ▶ cli-based

- ▶ results are written back to central database



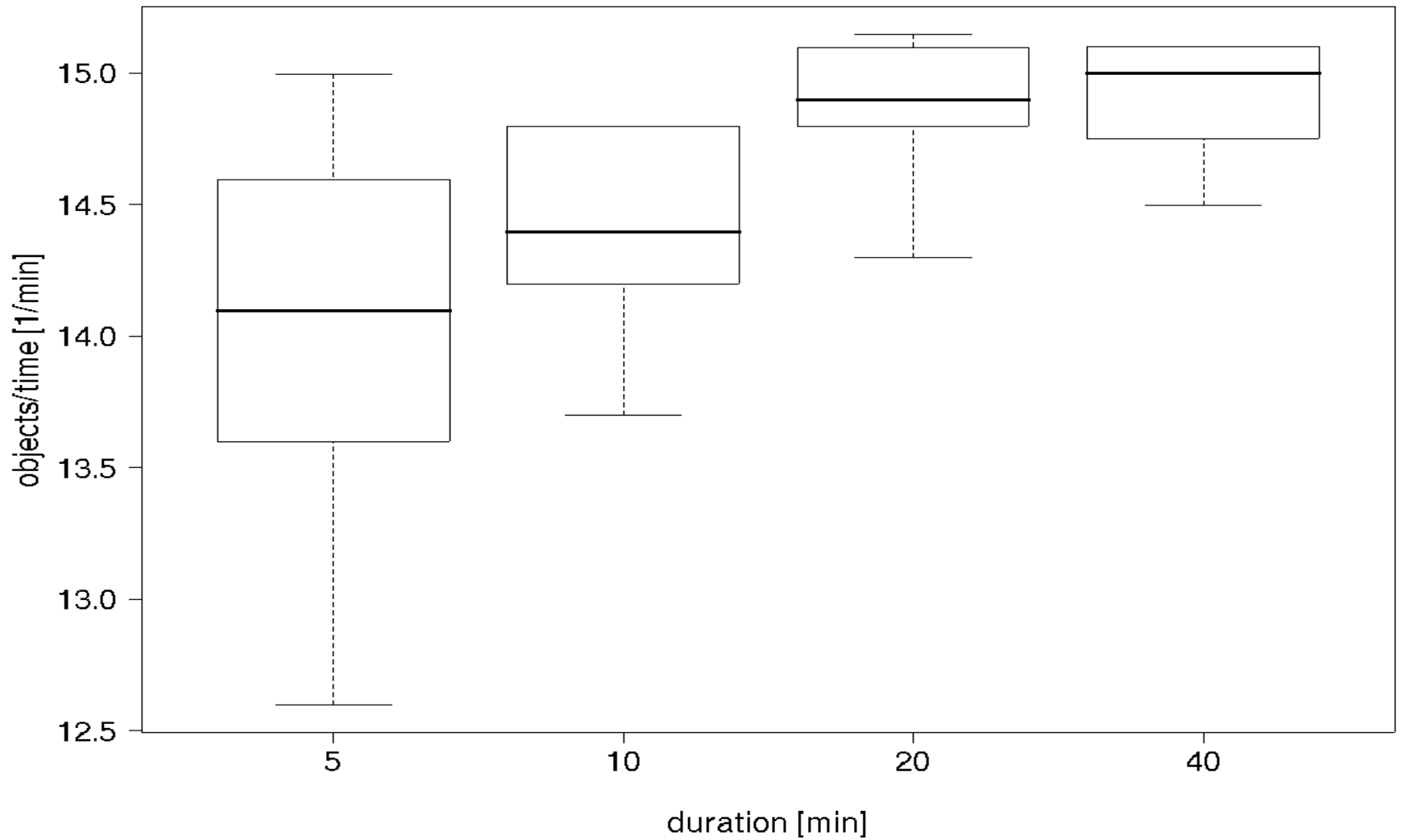
Results: Duration 1

- ▶ Motivation: random start positions
 - big variance in results
 - but small variance is needed to see small effects
- ▶ Solution:
 - ▶ longer duration and
 - ▶ average of different start positions

How long is long enough?

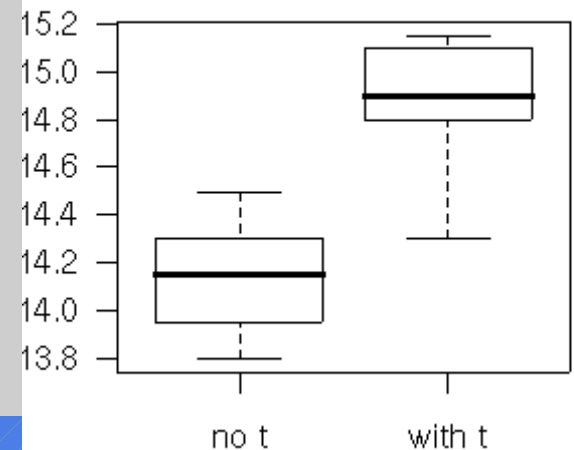
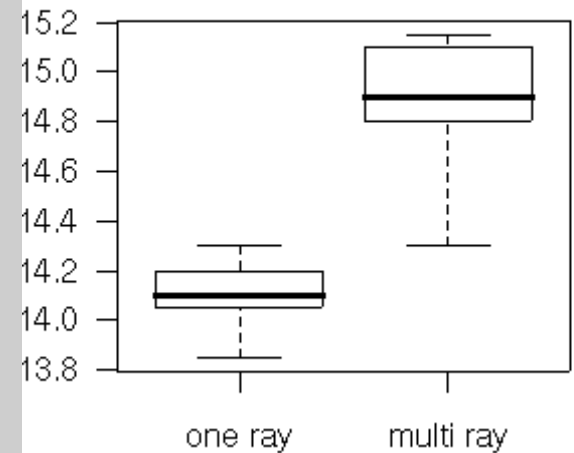
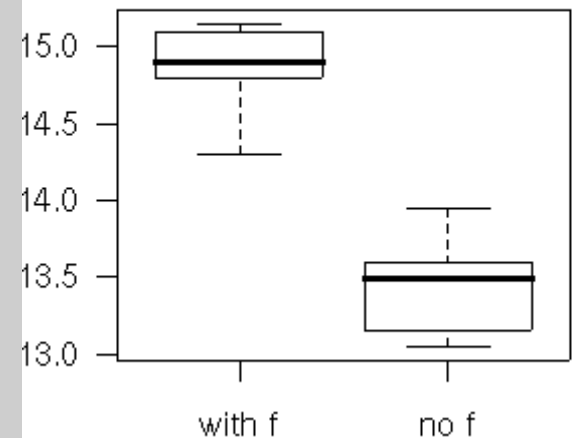


Results: Duration 2



Other Results

- ▶ different communication types (filter on/off)
- ▶ different sensors (one ray, multi ray)
- ▶ effect of object transfer (transfer on/off)

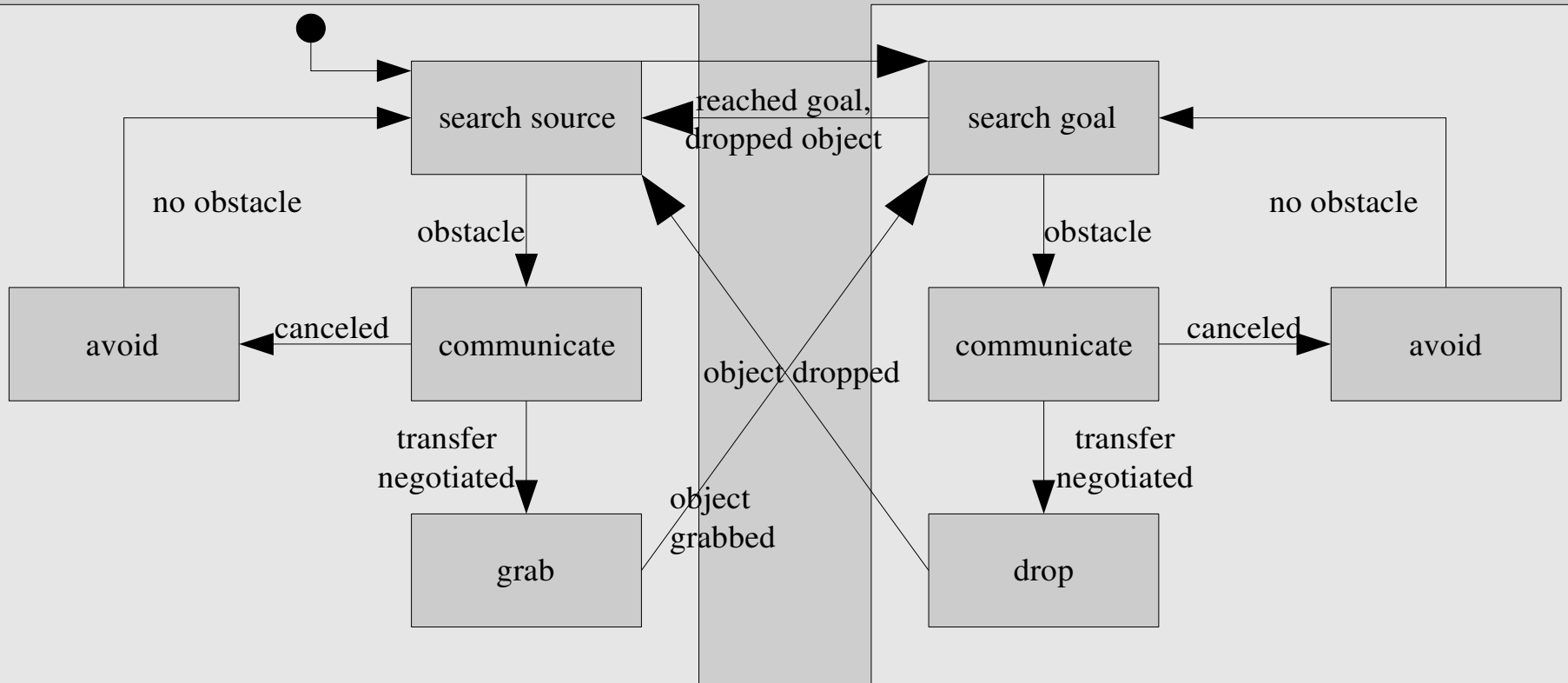


Conclusions & Future Work

- ▶ formal notation:
 - ▶ aggregation and meeting based scenarios
 - ▶ not yet supported: distribution, formations, ...
- ▶ framework for automatic evaluation process:
 - ▶ scripts and cli-based components
 - ▶ offline
- ▶ scenarios:
 - ▶ optimization
 - ▶ experiments in real arena



Implementation of Bucket Brigade



empty

loaded