WPM Künstliche Intelligenz

Projekt: “Schiffe-Versenken”

Victor Apostel, Stefan Bogdanski und Sibille Ritter

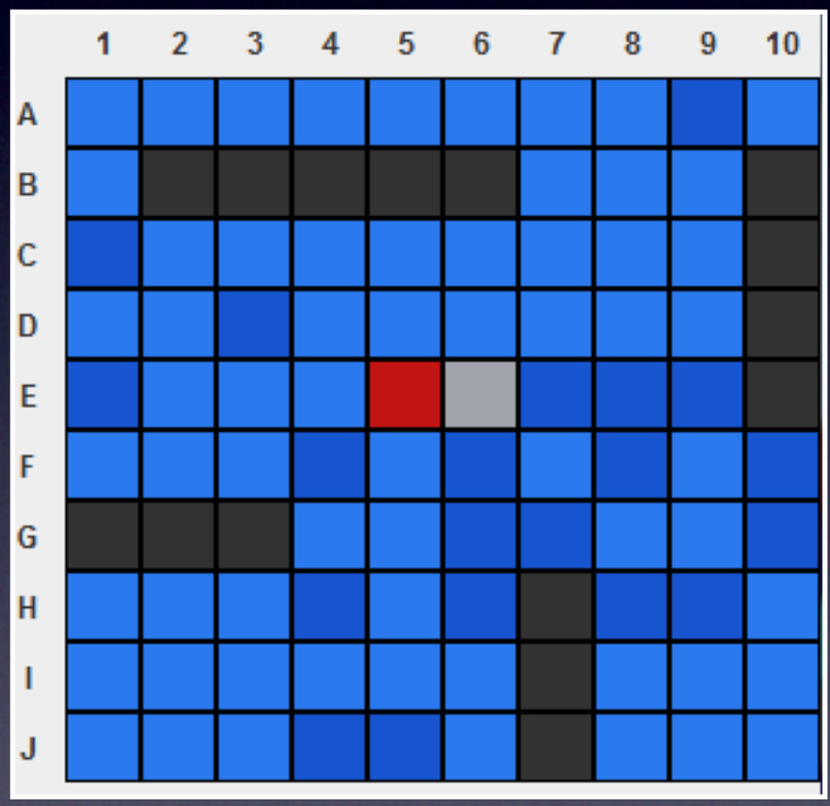
19.01.2011

Agenda

- Spielregeln
- Kommunikation
- Clientdesign
- Prolog Implementierung
- Demo: Spieler gg. KI
- Demo: KI gg. KI
- Ausblick

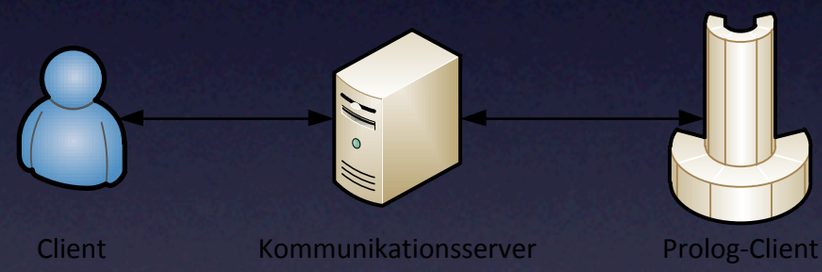
Spielregeln

- Spielfeld: 10x10
- Horizontale / Vertikale Schiffe
- 1x 5er, 1x 4er, 2x 3er, 1x 2er
- Schiffe dürfen nicht aneinander stoßen
- Abwechselndes ‘beschießen’ des Gegners



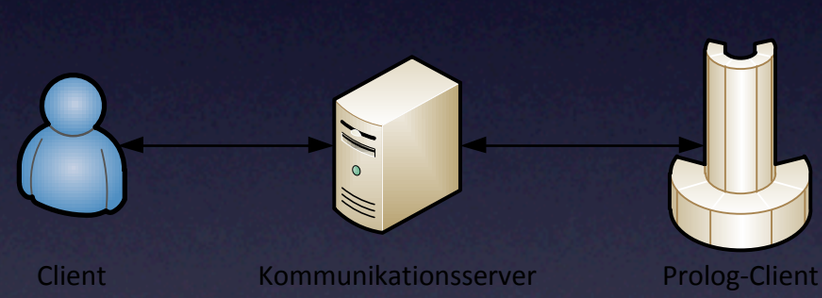
Kommunikation

- Client-Server Architektur
- Server:
  - koordiniert Partien
  - kann beliebig viele Spiele parallel ausführen
  - dient als Nachrichtenverteiler



Kommunikation

- Client:
  - implementiert eigentliche Spiellogik
- zwei Varianten
  - computergesteuert (Prolog)
  - HMI (Java)

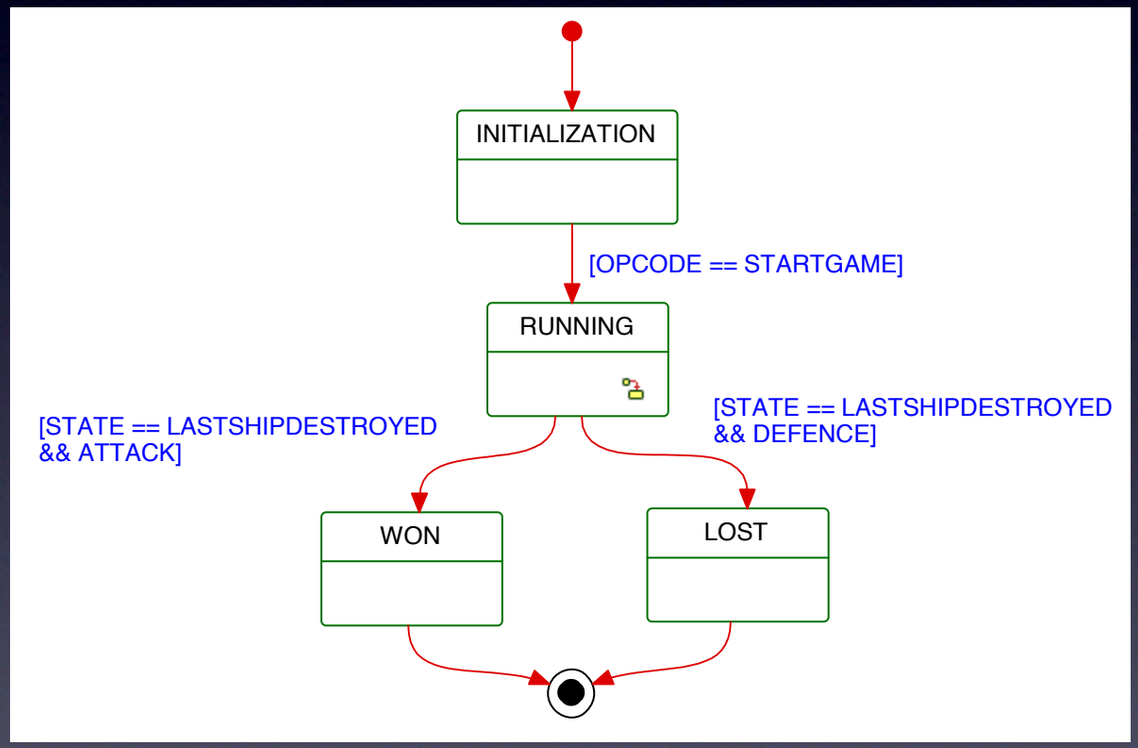


Kommunikation

- Nachrichtenaustausch auf Textbasis
- Format
  - (OPCODE, [Param 1, Param 2, Param 3]).
  - OPCODE
    - Angriff: (ATTACK, [X, Y]).
    - Antwort: (ATTACKRESPONSE, [X, Y, STATE]).
    - ...

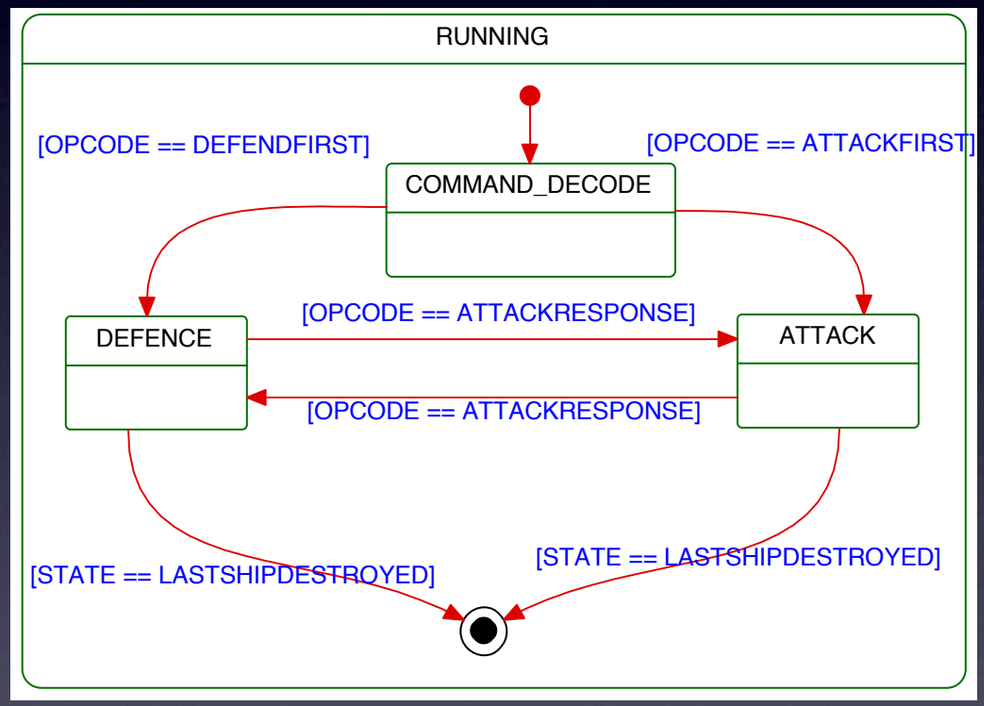
Clientdesign

- Client setzt primär Zustandsdiagramm um
- Zustandsänderungen durch Nachrichten
- Innerhalb von RUNNING Sub-Zustandsdiagramm



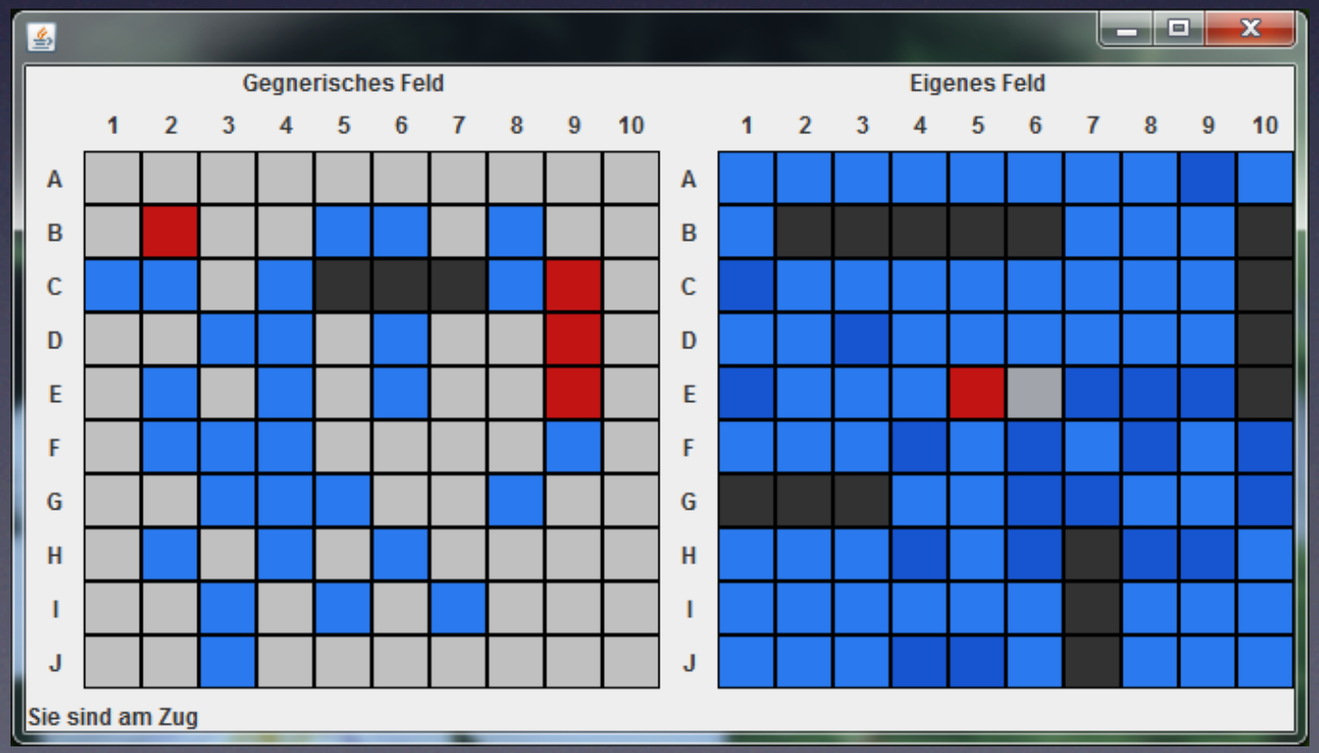
Clientdesign

- Primäre Zustände
  - ATTACK
  - DEFENCE
- Reihenfolge der Nachrichten
- Alternierend und zum Gegenspieler entgegengesetzt



Clientdesign

- Java-Client benutzt Entwurfsmuster
  - Model-View-Controller
  - Oberserver-Pattern



Prolog Implementierung
Spielregeln

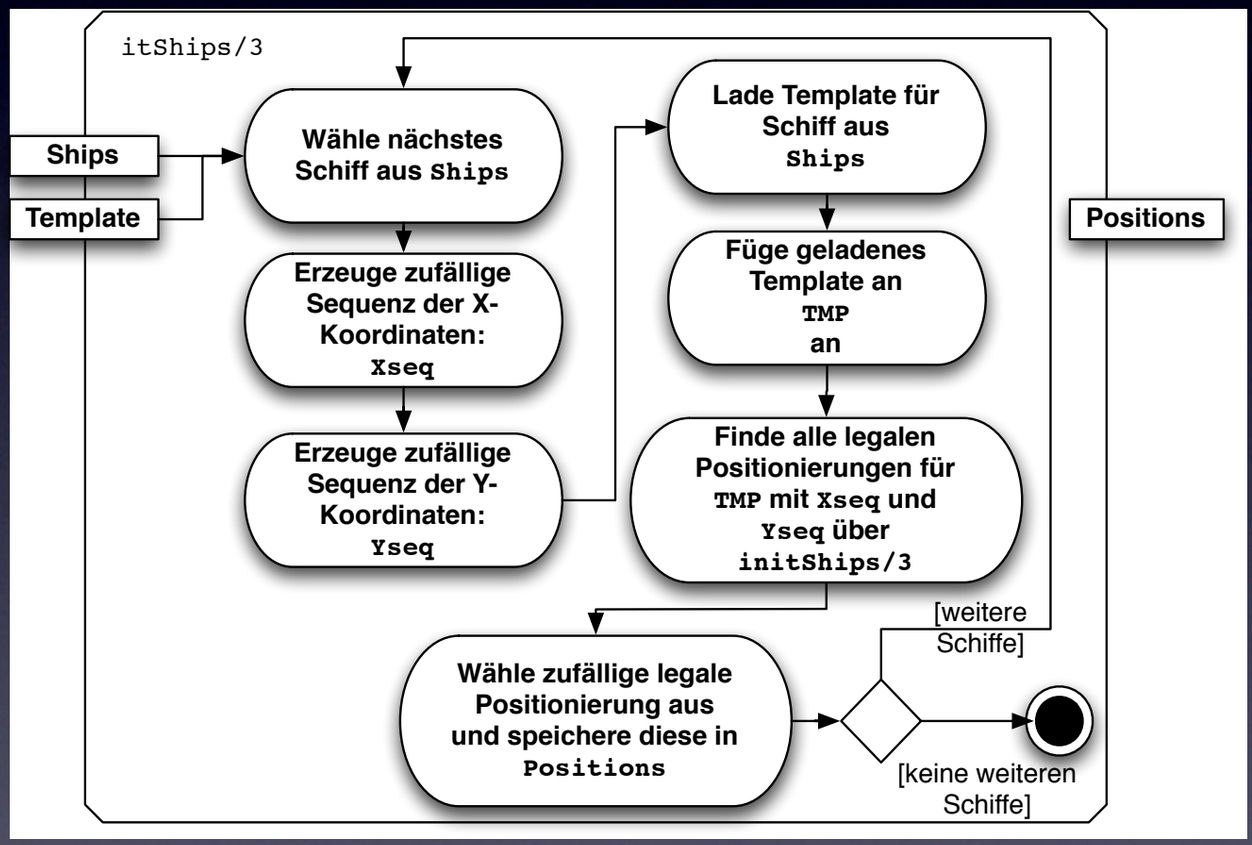
- “Schiffe müssen Horizontal (oder Vertikal) stehen”
- Schiffe dürfen einander nicht berühren, aber diagonal versetzt stehen.

```
connectedHV
(S1,P1,X1,Y1,S2,P2,X2,Y2):-
    S1 == S2,
    P1 \== P2,
    X1-X2 =:= P1-P2,
    Y1 == Y2,
    !
.

distance(S1,S2,X1,Y1,X2,Y2):-
    S1 \== S2,
    ((X1 == X2,
    abs(Y1-Y2)>1);
    (X1 \== X2,
    abs(Y1-Y2)>=1)),
    !
.
```

Prolog Implementierung
Platzierung

- Rekursiv
- **`Template`** der Schiffe
- Liste **`Ships`** mit zufälliger Reihenfolge
- Liste **`Positions`** mit konkreter Positionierung

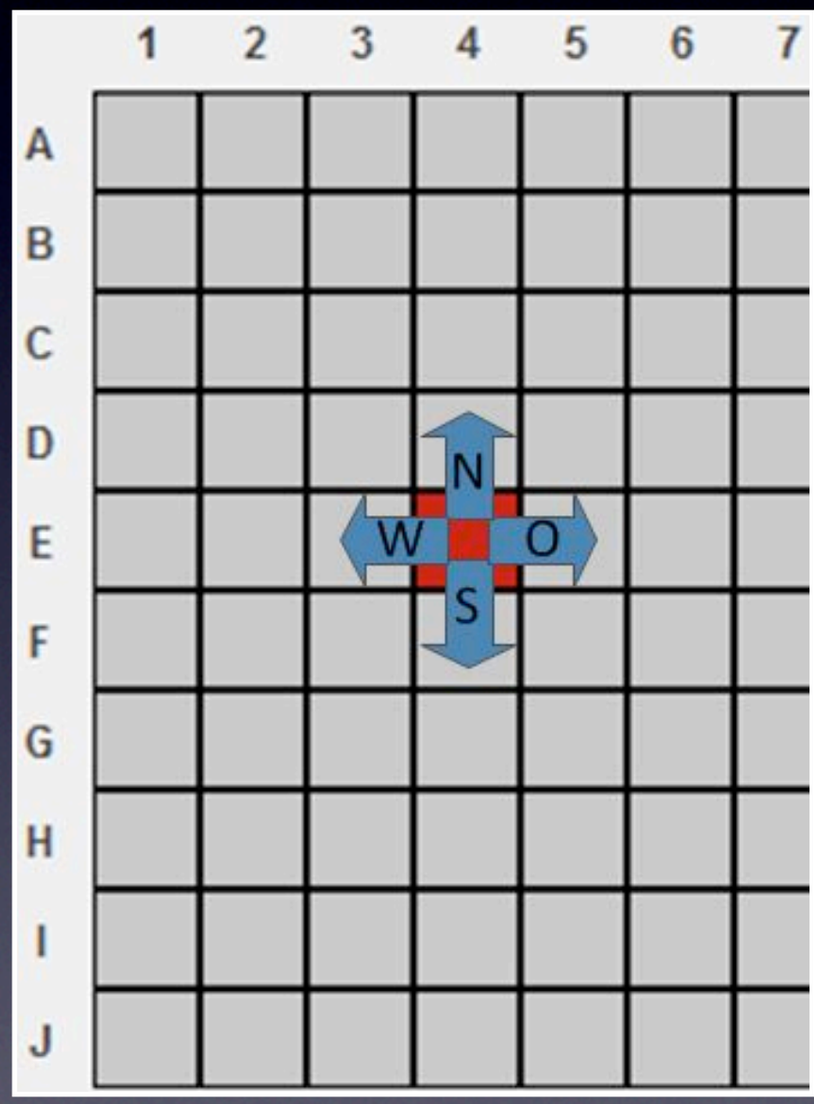


Prolog Implementierung
Strategie

- Es werden zufällige Koordinaten attackiert, bis…

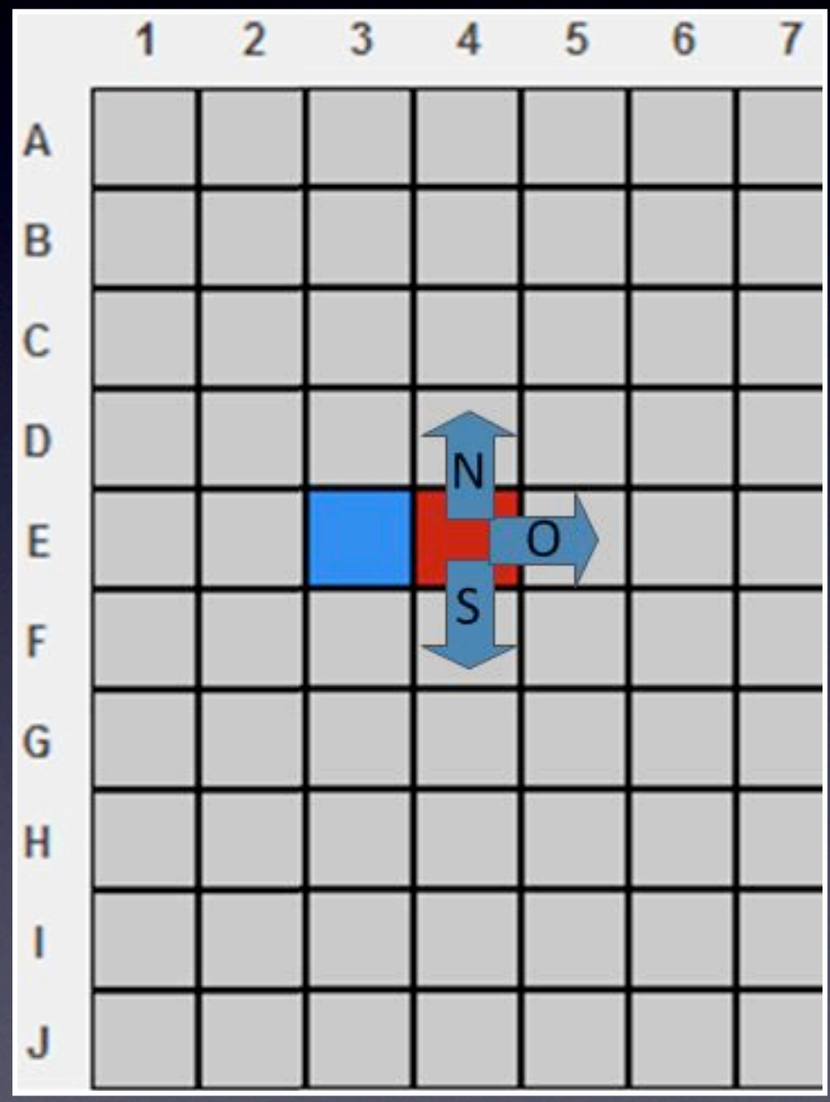
- … ein Treffer gelandet wird

➡ Hinzufügen der Nachbarn in die Openlist



Prolog Implementierung
Strategie

- Weitere Angriffe auf die Koordinaten in der Openlist

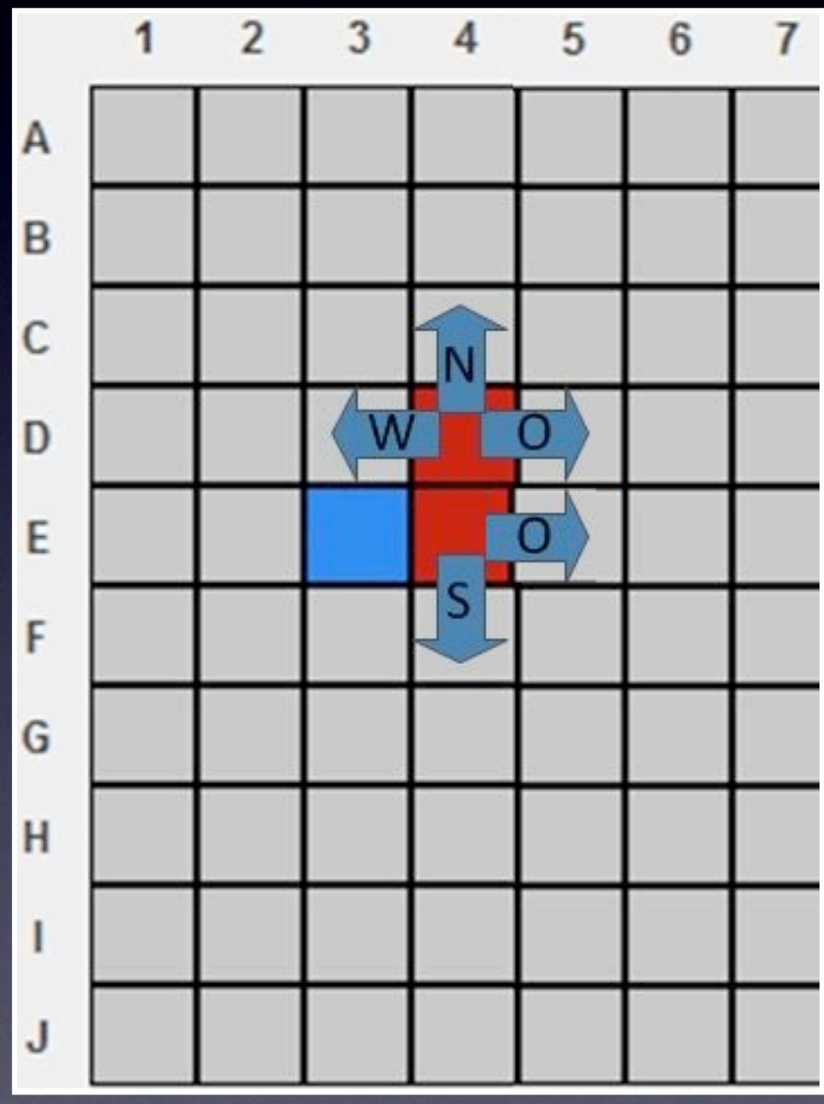


Prolog Implementierung
Strategie

- Wird ein weiterer Treffer gelandet:

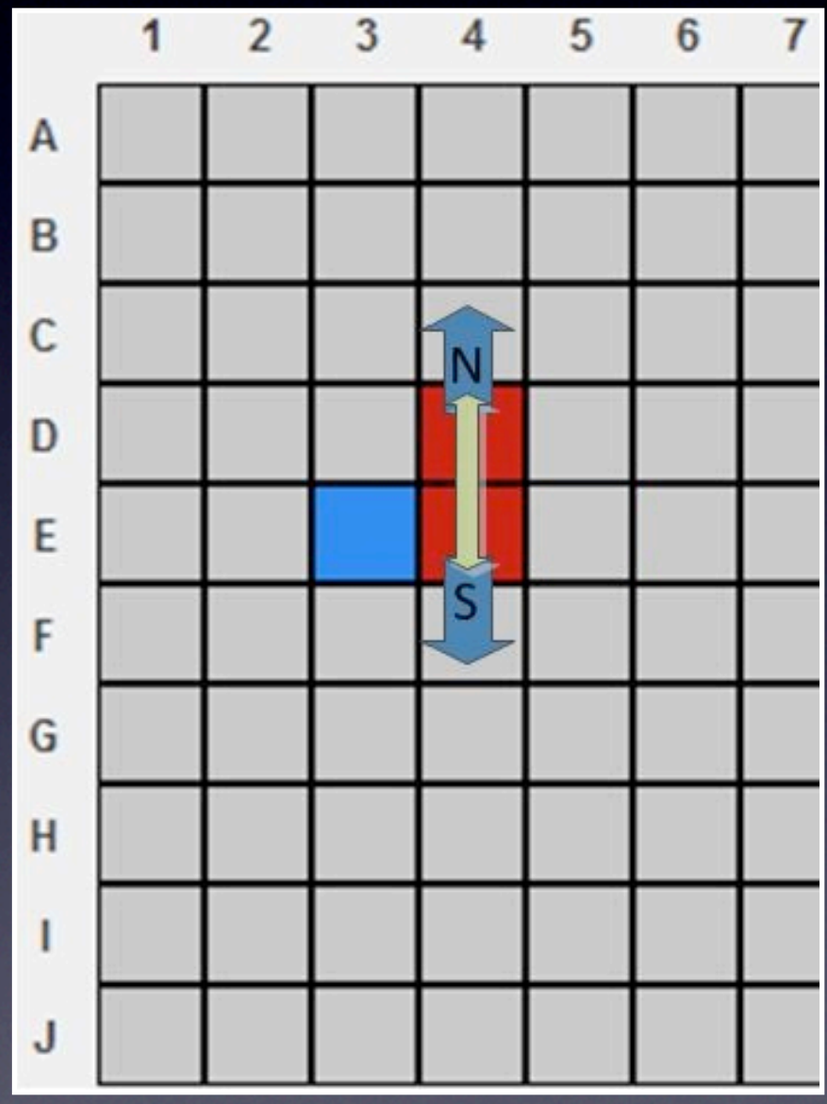
➡Kann die Orientierung des Schiffes bestimmt werden

➡Daher können Koordinaten aus der Openlist ausgeschlossen werden

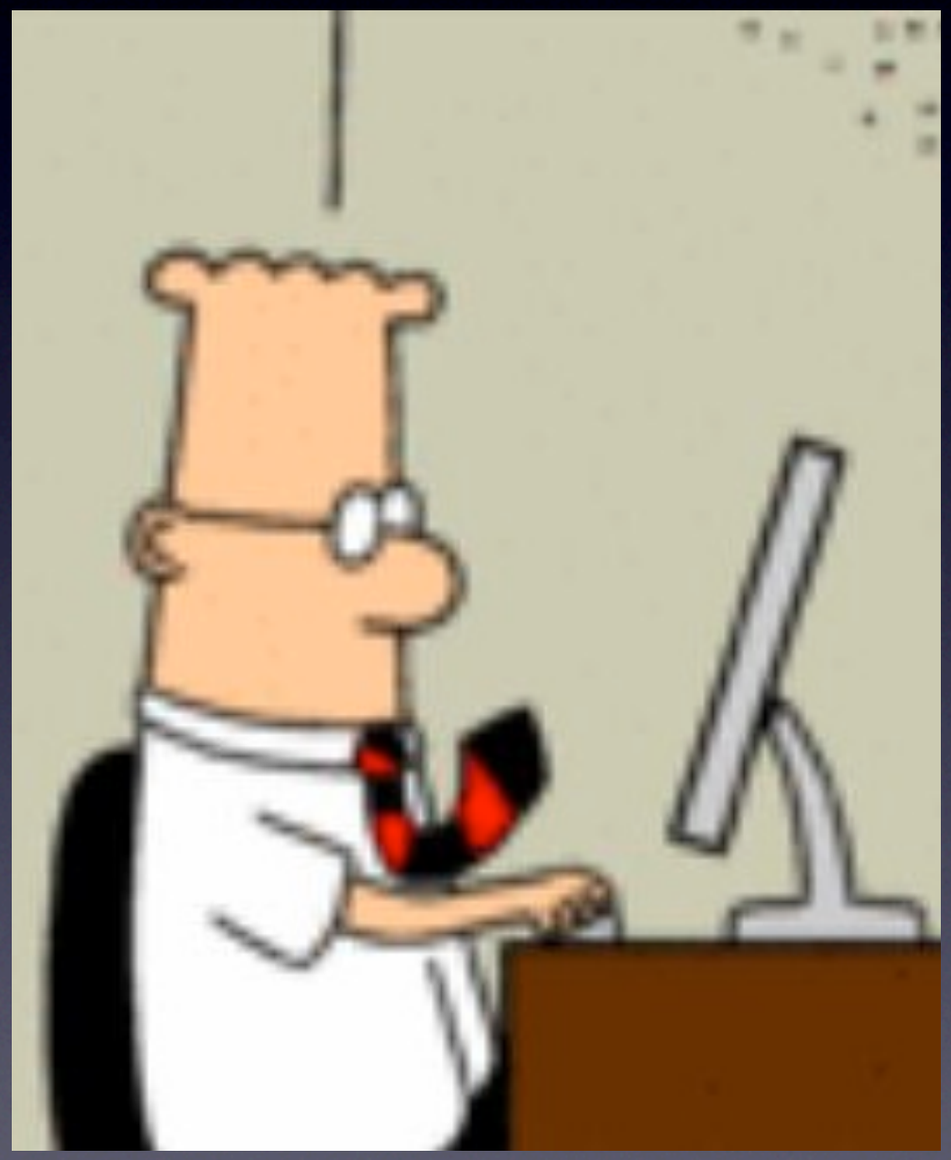


Prolog Implementierung
Strategie

- Weitere Angriffe entlang der Orientierung, solange bis „Schiff versenkt“



Demo
Spieler gg. KI



Demo
KI gg. KI

Ausblick

- Zufälliges Angreifen durch strukturiertes Vorgehen ersetzen
- Implementierung eines lernenden Spielers (Analyse der gegnerischen Angriffe und Schiffspositionen)

Vielen Dank,

für Ihre Aufmerksamkeit