Automated Negotiation

A Tutorial

Yasser Mohammad[1,2]

[1] NEC-AIST Collaborative Lab., AIST, Japan

[2] Assiut University, Egypt

October 29, 2019

Outline

Outline

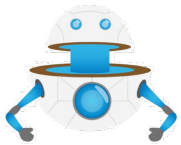
Outline

Outline

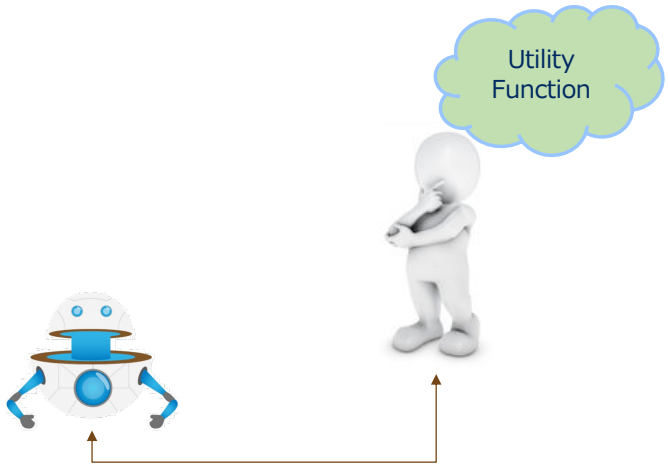
Outline

Outline

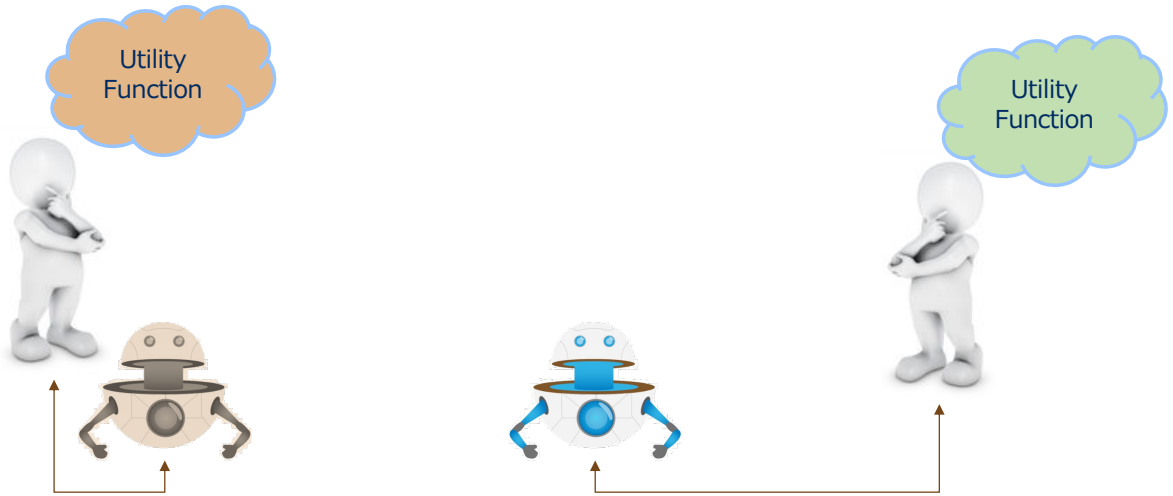
Outline



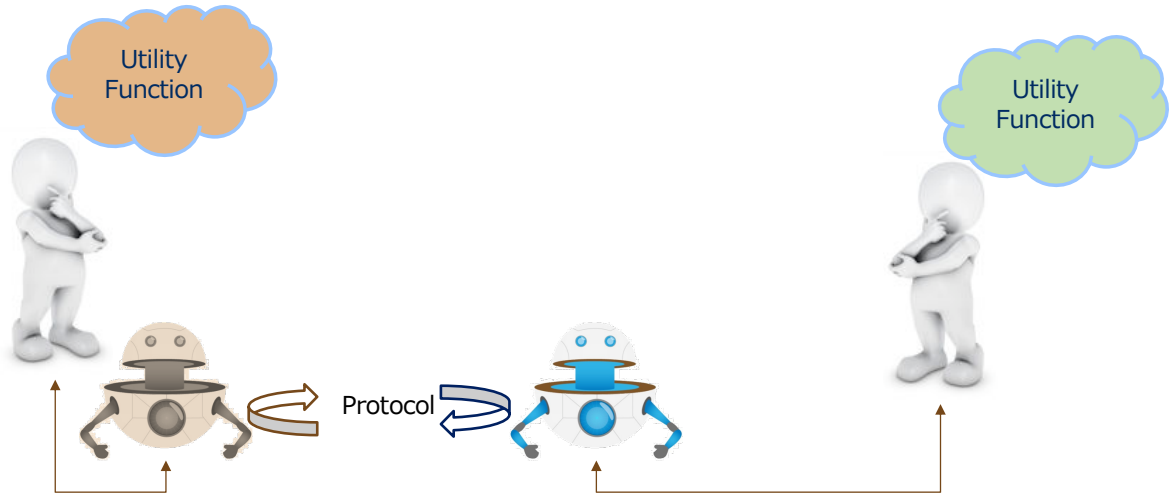
- A method to achieve agreement among self-interested actors.
- Negotiation is important $\rightarrow$ win-win agreements.
- Automatic Negotiation $\rightarrow$ \$\$\$
  - smart contracts, resource allocation, SCM, etc



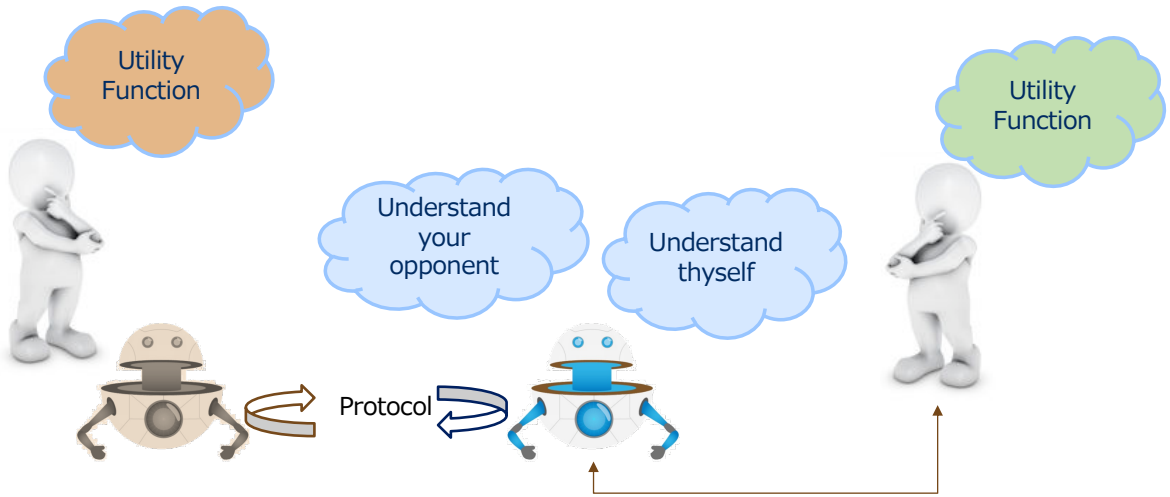
- A method to achieve agreement among self-interested actors.
- Negotiation is important $\rightarrow$ win-win agreements.
- Automatic Negotiation $\rightarrow$ \$\$\$
  - smart contracts, resource allocation, SCM, etc



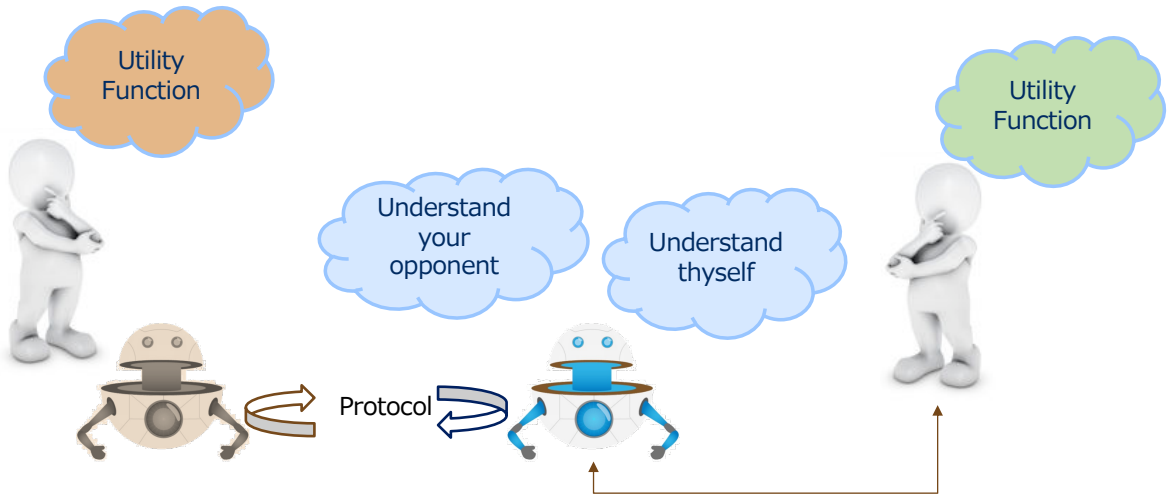
- A method to achieve agreement among self-interested actors.
- Negotiation is important $\rightarrow$ win-win agreements.
- Automatic Negotiation $\rightarrow$ \$\$\$
  - smart contracts, resource allocation, SCM, etc



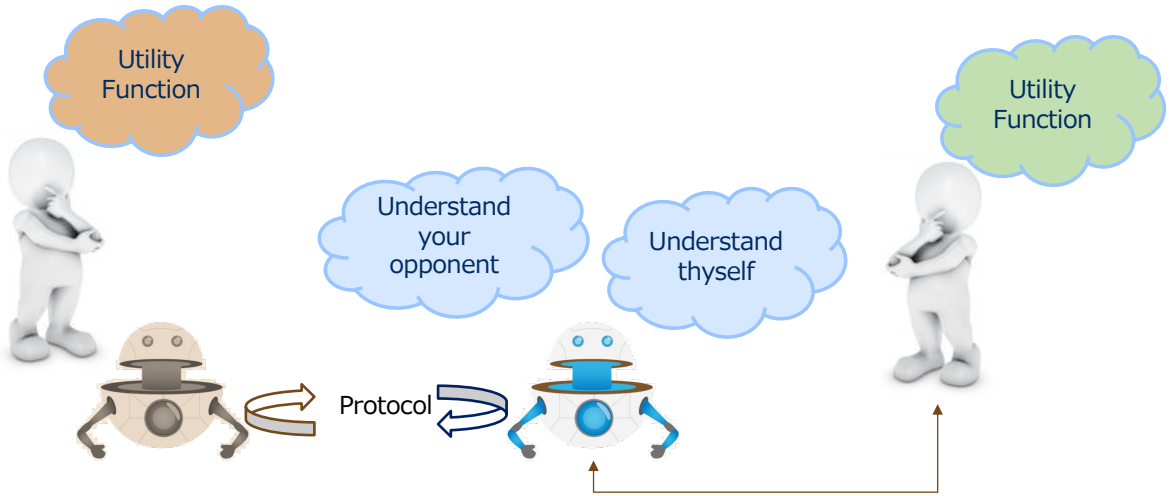
- A method to achieve agreement among self-interested actors.
- Negotiation is important $\rightarrow$ win-win agreements.
- Automatic Negotiation $\rightarrow$ \$\$\$
  - smart contracts, resource allocation, SCM, etc



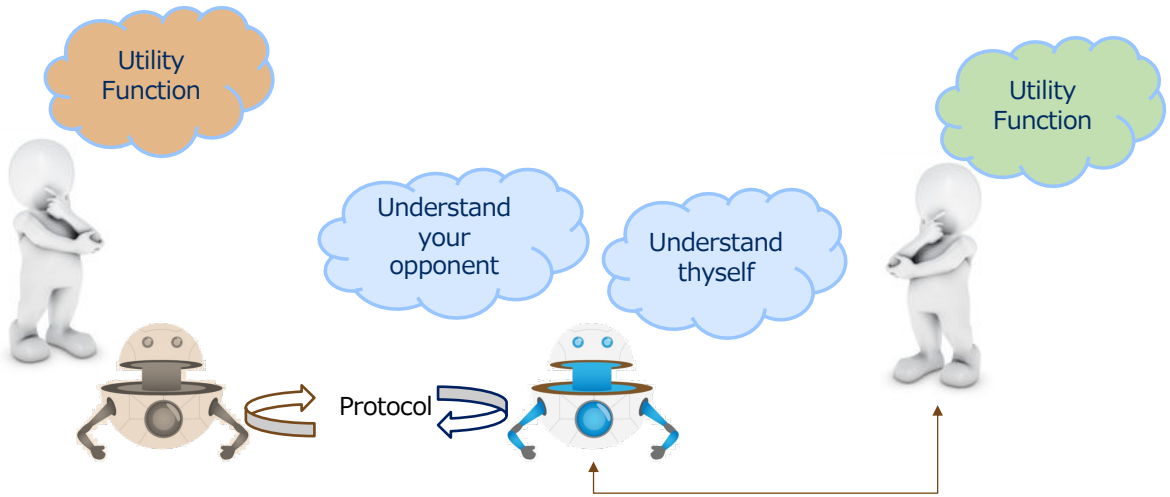
- A method to achieve agreement among self-interested actors.
- Negotiation is important $\rightarrow$ win-win agreements.
- Automatic Negotiation $\rightarrow$ \$\$\$
  - smart contracts, resource allocation, SCM, etc



- A method to achieve agreement among self-interested actors.
- Negotiation is important → win-win agreements.
- Automatic Negotiation → $$$
  - smart contracts, resource allocation, SCM, etc

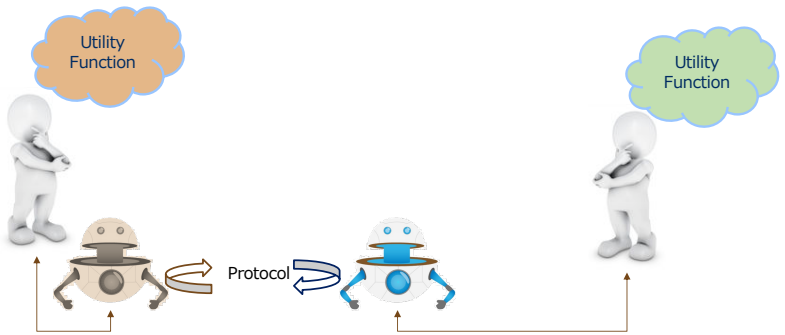


- A method to achieve agreement among self-interested actors.
- Negotiation is important → win-win agreements.
- Automatic Negotiation → $$$
  - smart contracts, resource allocation, SCM, etc



- A method to achieve agreement among self-interested actors.
- Negotiation is important $\rightarrow$ win-win agreements.
- Automatic Negotiation $\rightarrow$ \$\$\$
  - smart contracts, resource allocation, SCM, etc

Components of the Negotiation Problem



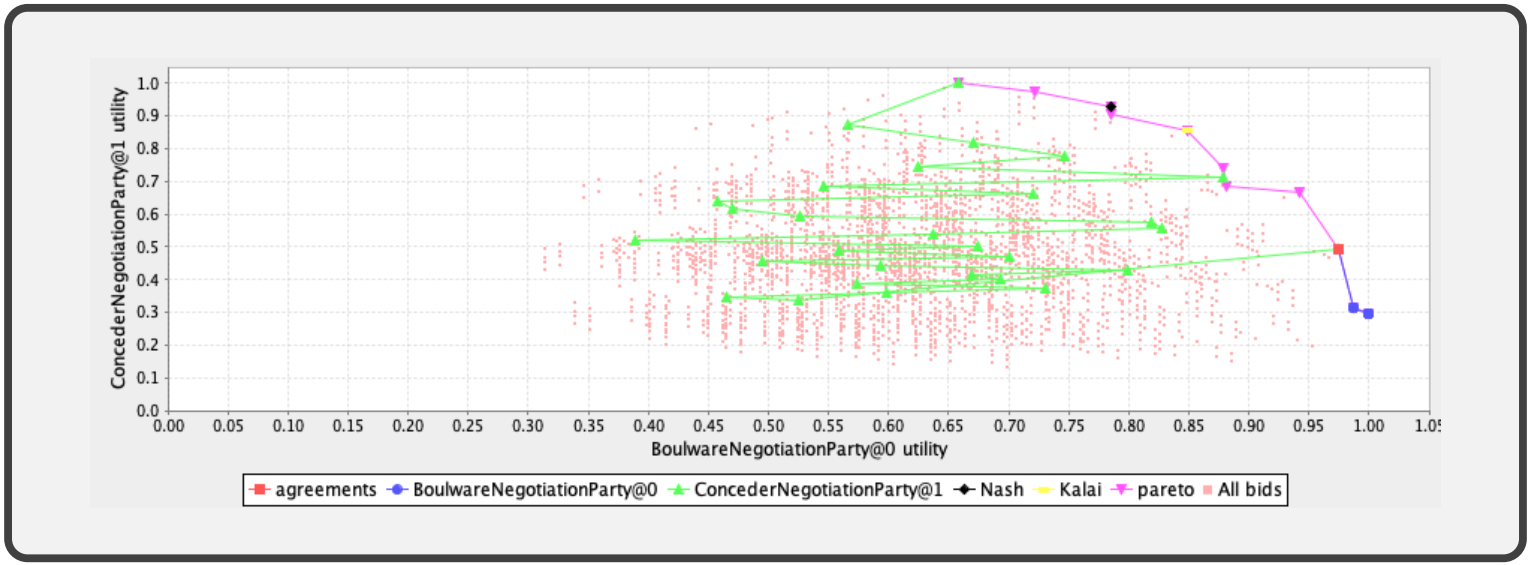
Negotiation Protocol Defines how negotiation is to be conducted [Mechanism Design Problem].

- Alternating Offers Protocol
- Single Text Protocol
- ...

Negotiation Strategy Defines how an agent behaves during the negotiation [Effective Negotiation Problem].

- Time-based strategies: Boulware, conceder, ...
- Tit-for-tat variations
- ...

Important Concepts

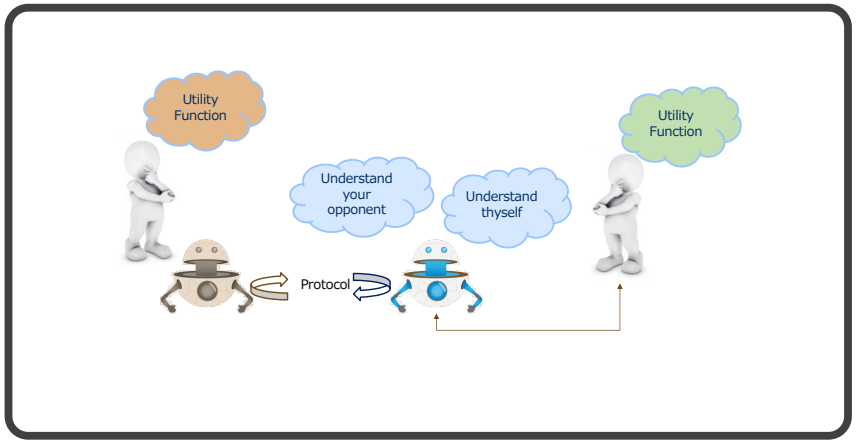


Pareto Frontier Outcomes that cannot be improved for one actor without making another worse off.

Welfare Total utility received by all actors.

Surplus utility Utility above disagreement utility.

Types of Automated Negotiation Problems

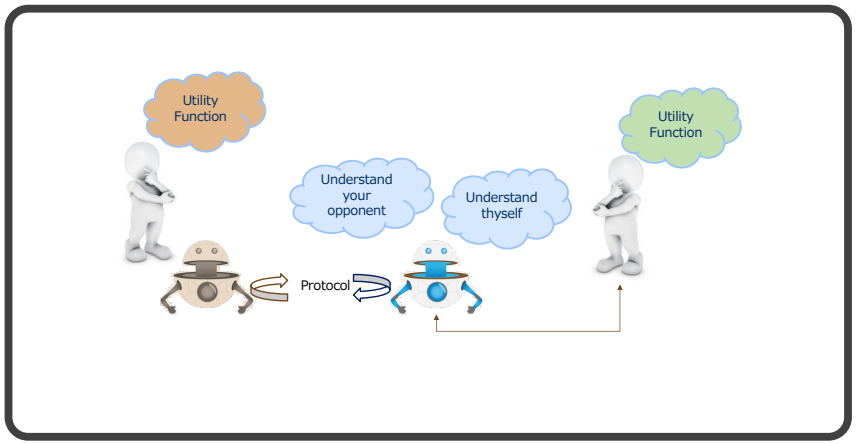


Types of Automated Negotiation Problems

NEC-AIST AI Cooperative Research Laboratory

Negotiator type

1. Agent-Agent negotiation
2. Agent-Human negotiation



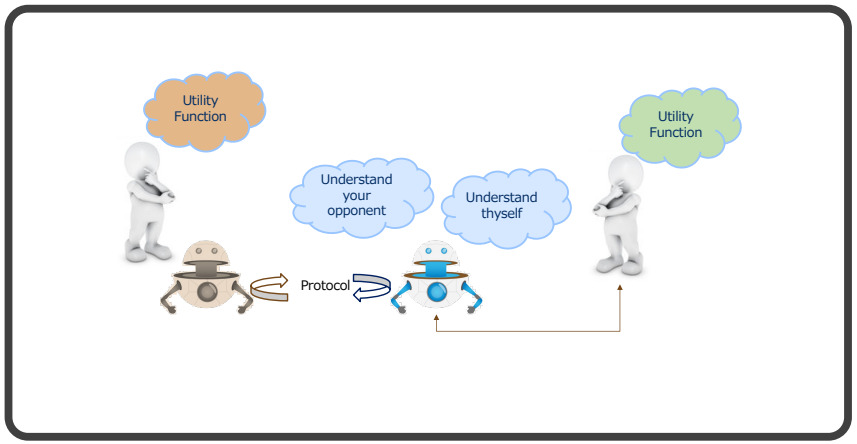
Types of Automated Negotiation Problems

Negotiator type

1. Agent-Agent negotiation
2. Agent-Human negotiation

Number of negotiators

1. Bilateral negotiation
2. Multilateral negotiation



Types of Automated Negotiation Problems

Negotiator type

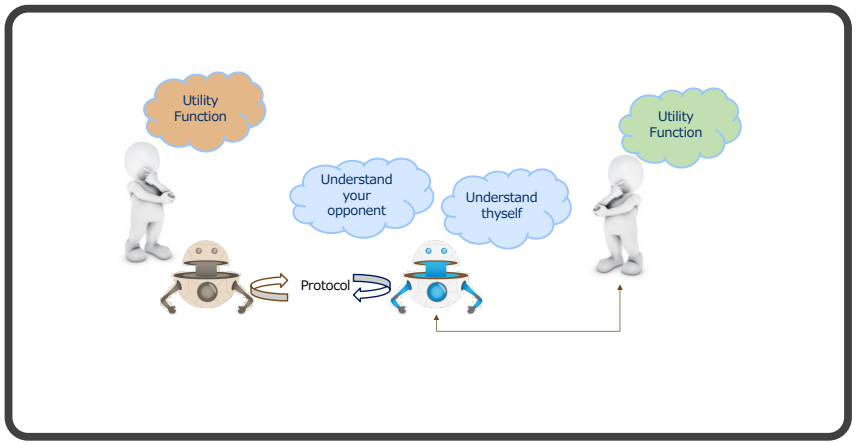
1. Agent-Agent negotiation
2. Agent-Human negotiation

Number of negotiators

1. Bilateral negotiation
2. Multilateral negotiation

Outcome Space

1. Single Issue: $\Omega = \{\omega_0, \omega_1, ...\}$
2. Multiple Issues: $\Omega = \prod_{i=1}^{n_i} I_i$



Types of Automated Negotiation Problems

NEC-AIST AI Cooperative Research Laboratory

Negotiator type

1. Agent-Agent negotiation
2. Agent-Human negotiation

Number of negotiators

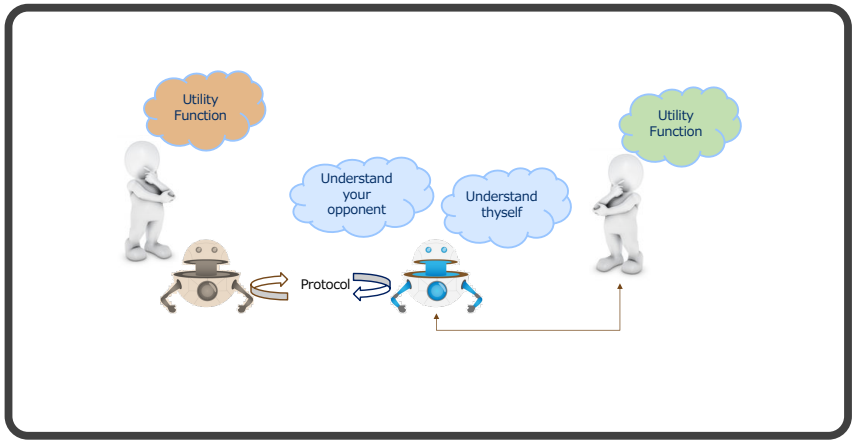
1. Bilateral negotiation
2. Multilateral negotiation

Outcome Space

1. Single Issue: $\Omega = \{\omega_0, \omega_1, ...\}$
2. Multiple Issues: $\Omega = \prod_{i=1}^{n_i} I_i$

Protocol Type

1. Mediated
2. Unmediated

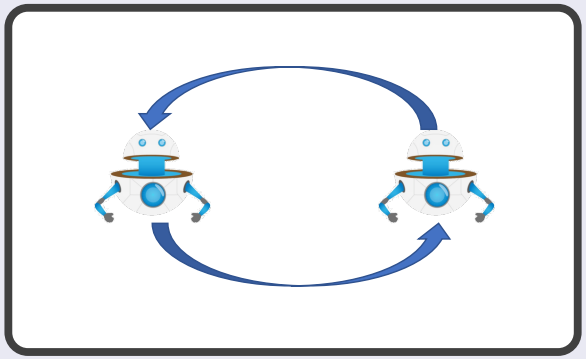


Outline

Unmediated Protocols

Main Features

- No central coordinator.
- Agents negotiate by exchanging *messages*.
- All proposals come from negotiators.

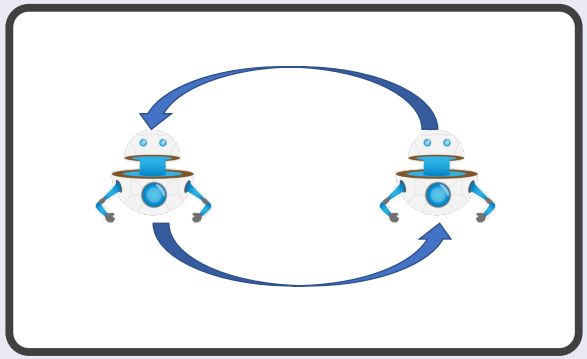


Unmediated Protocols

NEC-AIST AI Cooperative Research Laboratory

Main Features

- No central coordinator.
- Agents negotiate by exchanging *messages*.
- All proposals come from negotiators.



Examples

Nash Bargaining Game Single iteration, single issue, bilateral protocol with complete information.

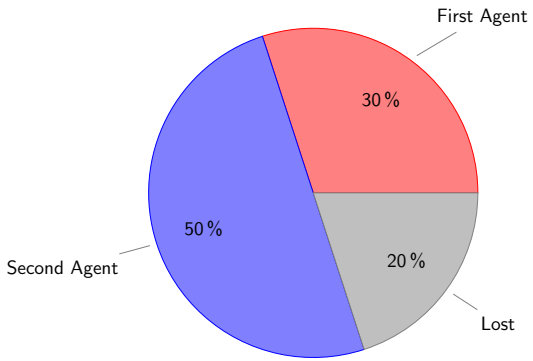
Rubinstein Bargaining Protocol Infinite horizon, single issue, bilateral protocol with complete information [Rubinstein, 1982].

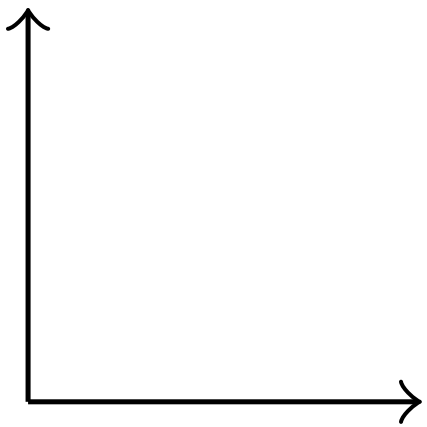
Stacked Alternating Offers Protocol Finite horizon, multi-issue, multilateral protocol with partial information [Aydoğan et al., 2017].

Outline

Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:



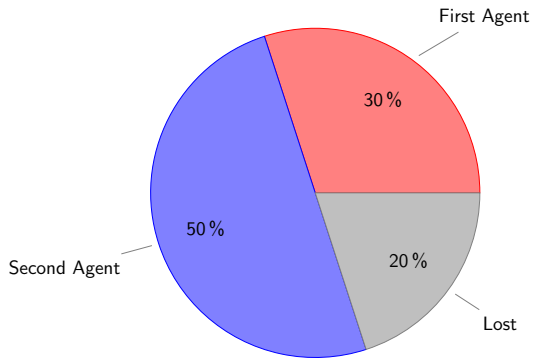


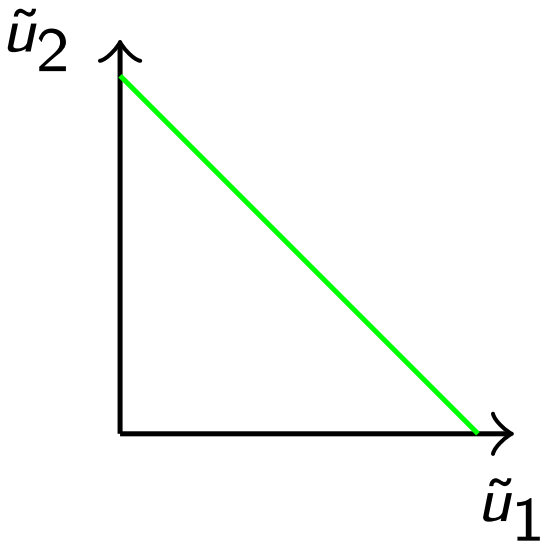
Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A (usually convex) feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$



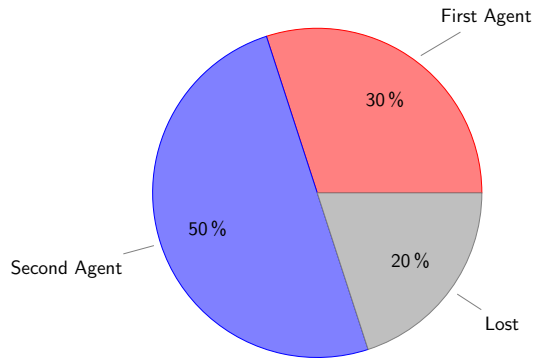


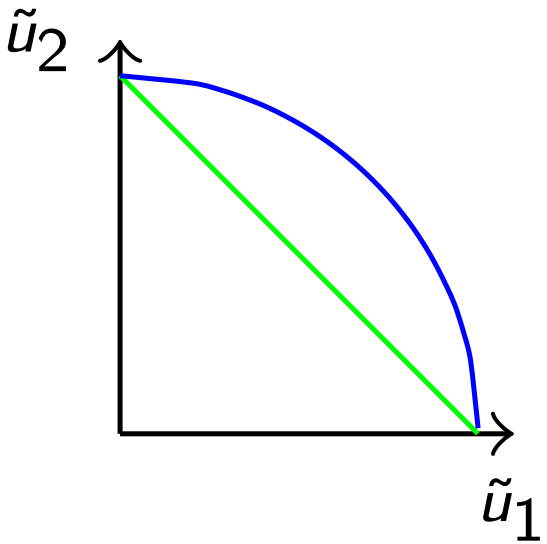
Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A (usually convex) feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$



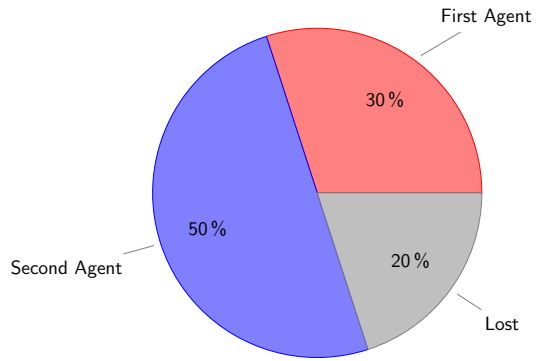


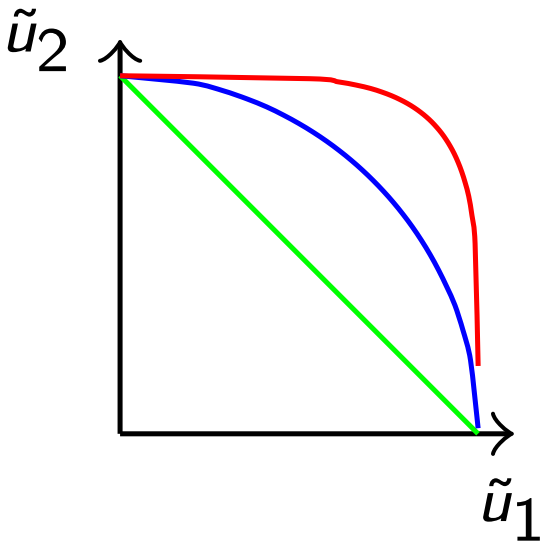
Nash Bargaining Game: Description

A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A (usually convex) feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$





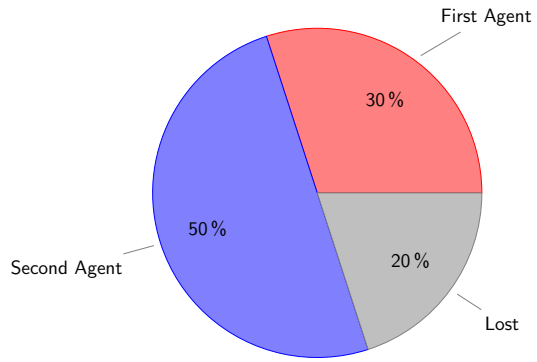
Nash Bargaining Game: Description

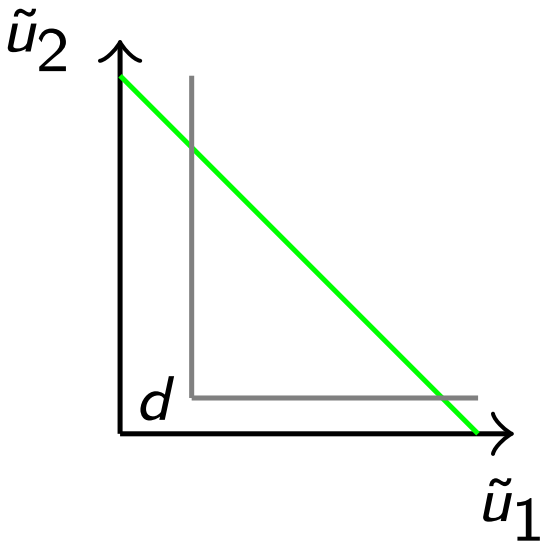
A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A (usually convex) feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$

- A disagreement point $d \equiv \tilde{u}_1(\phi) + \tilde{u}_2(\phi) \in \Re^2$ which is the utility value received by the two players in case of disagreement (reserved values).





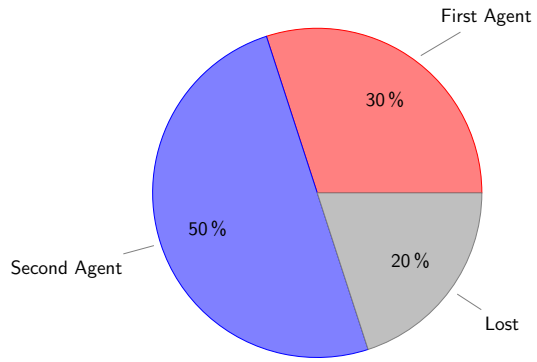
Nash Bargaining Game: Description

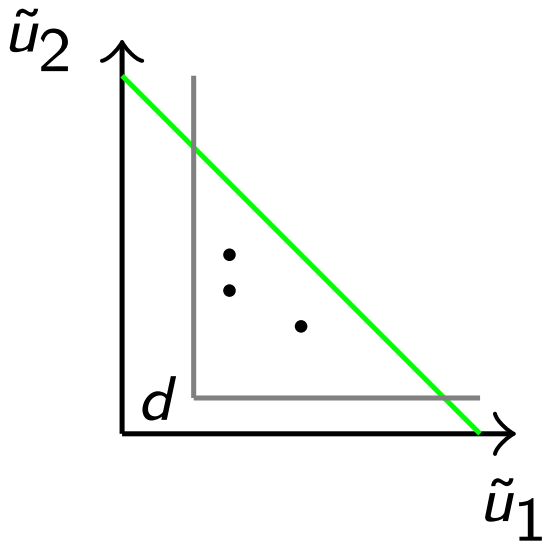
A single-step full-information bilateral negotiation with $\Omega = [0, 1]^2$ and two utility functions $(\tilde{u}_1, \tilde{u}_2)$ such that:

- A (usually convex) feasible set of agreements F. A common example is to define F as all the outcomes for which the total utility received by negotiators is less than or equal to one:

$$F = \{(\omega_1, \omega_2) | \tilde{u}_2(\omega_2) + \tilde{u}_1(\omega_1) \leq 1\}.$$

- A disagreement point $d \equiv \tilde{u}_1(\phi) + \tilde{u}_2(\phi) \in \Re^2$ which is the utility value received by the two players in case of disagreement (reserved values).





Outline

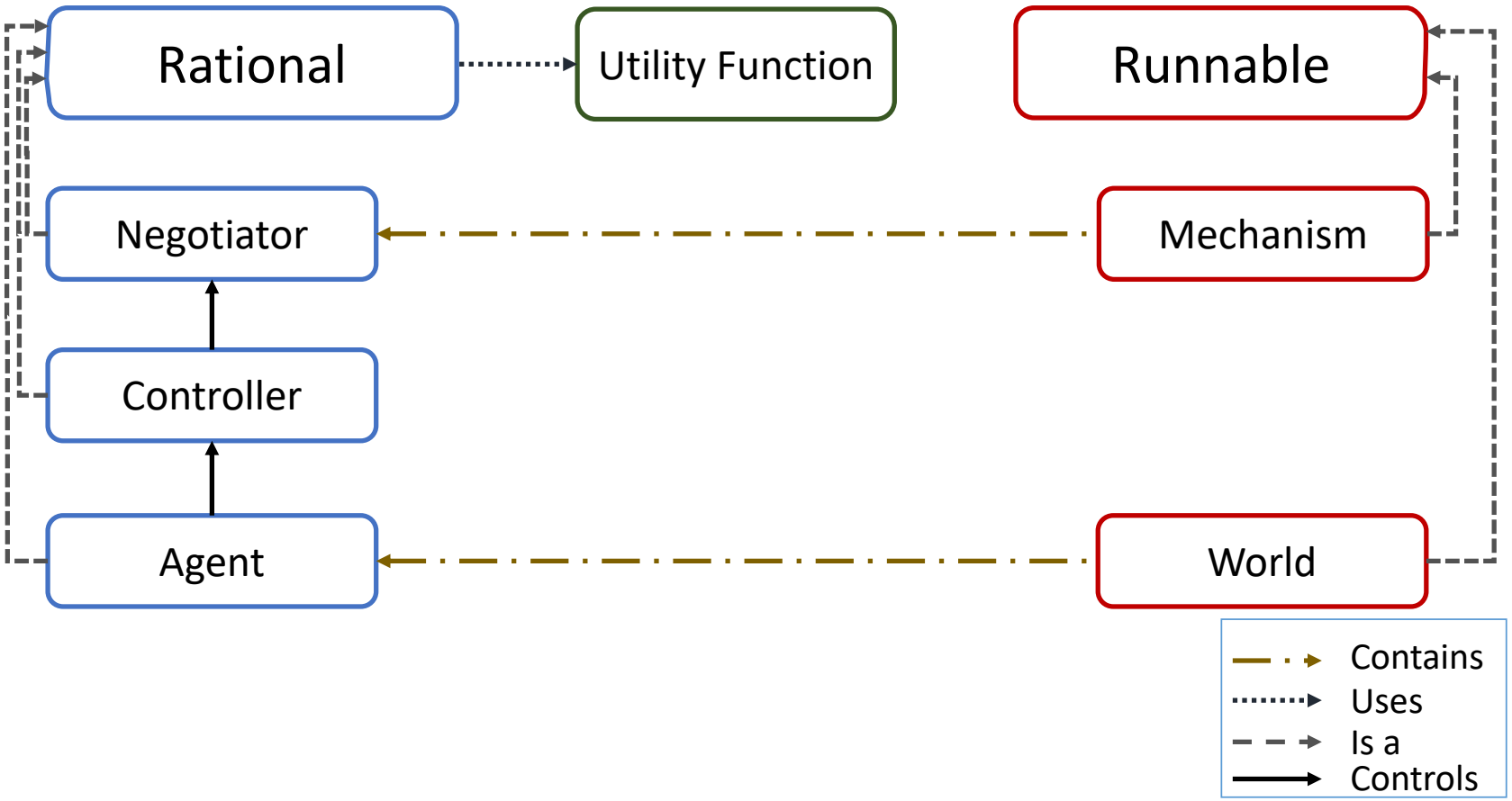
A brief history of ANAC

- The Automated Negotiating Agents Competition (ANAC)
- Started on 2010 and is currently in its 11th incarnation.
- Was conducted in conjunction with AAMAS but currently IJCAI.

Year	Challenge	Year	Challenge
2010	Domain Independence	2011	Linear Ufuns
2012	Reservation Value	2014	Learning and Adaptation
2015	Three-party negotiation	2016	Energy Grid Theme
2017	Repeated Negotiations, Diplomacy, HAN	2018	Repeated Negotiations, Diplomacy, HAN
2019	Elicitation, Diplomacy, HAN, Werewolf, **SCML**	2020	Uncertainty, HAN, Werewolf, **SCML**. HUMAINE

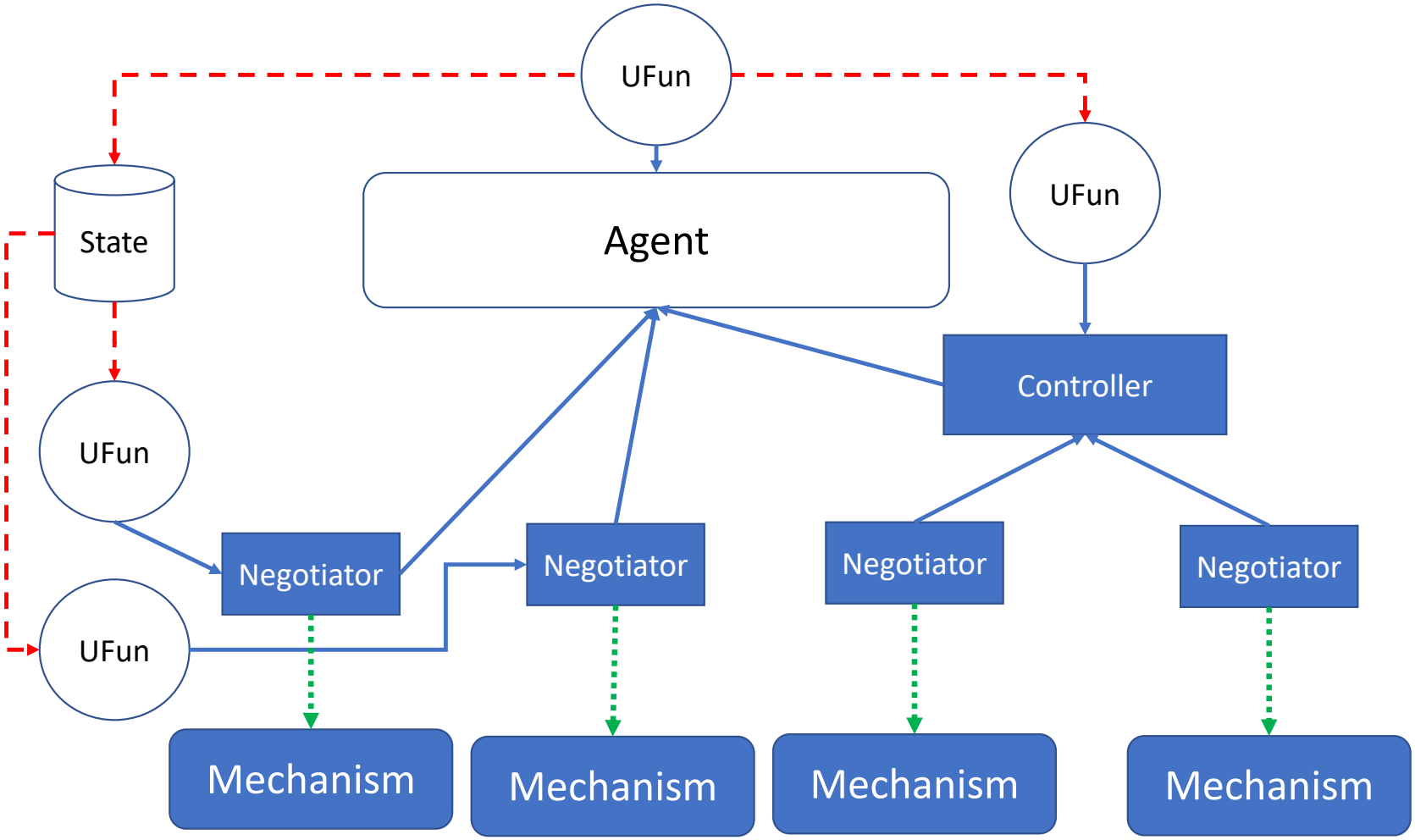
Outline

NegMAS[1] in two slides

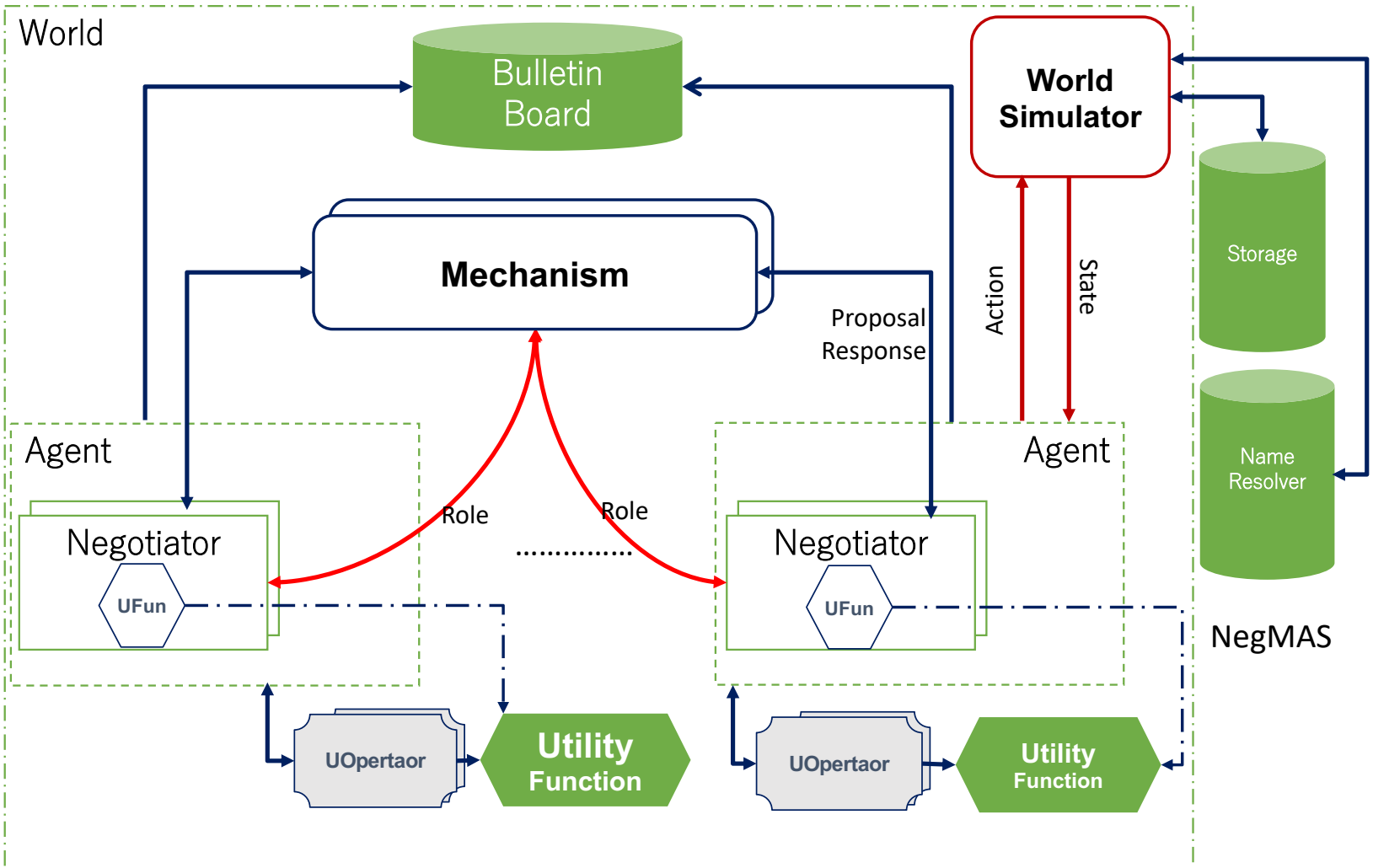


[1] https://www.github.com/yasserfarouk/negmas

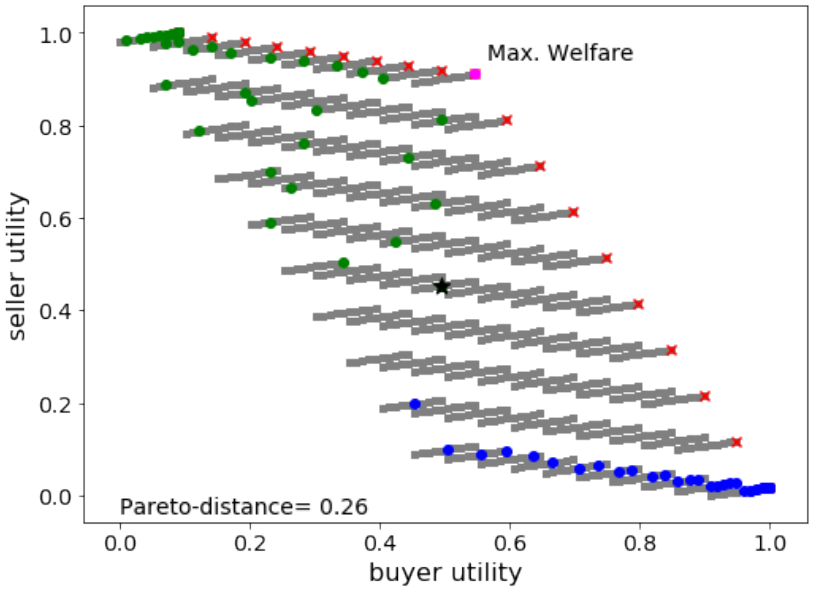
NegMAS in two slides

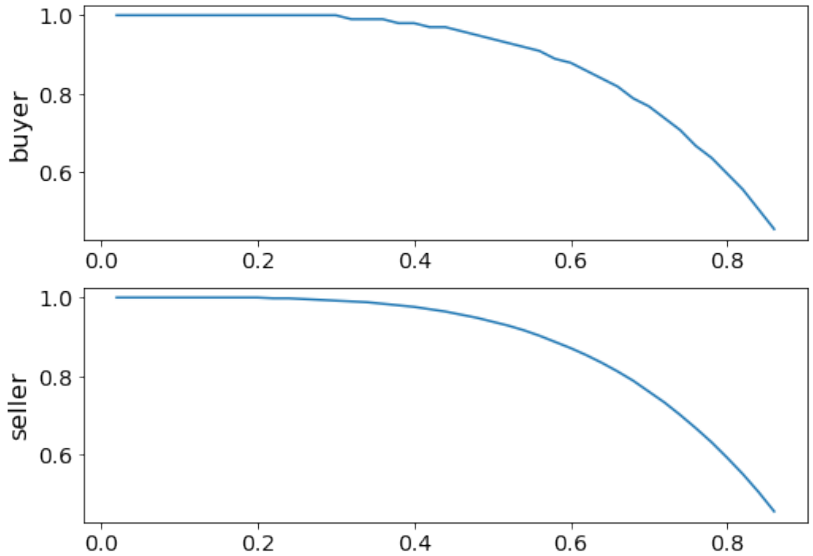


NegMAS in two slides (... OK 3)



NegMAS in two slides (... really!!!)





- An Example negotiation.
- Can you spot a problem?

Outline

Outline

Negotiation in SCM Business

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [2]

[2] KPMG report: https://bit.ly/3kDRy6I
[3] Forrester report: https://bit.ly/3nwXEaY
[4] UN/CEFACT Project website: https://bit.ly/38LOsLX

Negotiation in SCM Business

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [2]
- A recent study suggests that at least 15 companies are working in *contracting support systems* [3].

[2] KPMG report: https://bit.ly/3kDRy6I
[3] Forrester report: https://bit.ly/3nwXEaY
[4] UN/CEFACT Project website: https://bit.ly/38LOsLX

Negotiation in SCM Business

- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [2]
- A recent study suggests that at least 15 companies are working in *contracting support systems* [3].
- A recent UNECE UN/CEFACT proposal to standardize negotiation protocols for SCM and other applications [4]

[2] KPMG report: https://bit.ly/3kDRy6I
[3] Forrester report: https://bit.ly/3nwXEaY
[4] UN/CEFACT Project website: https://bit.ly/38LOsLX

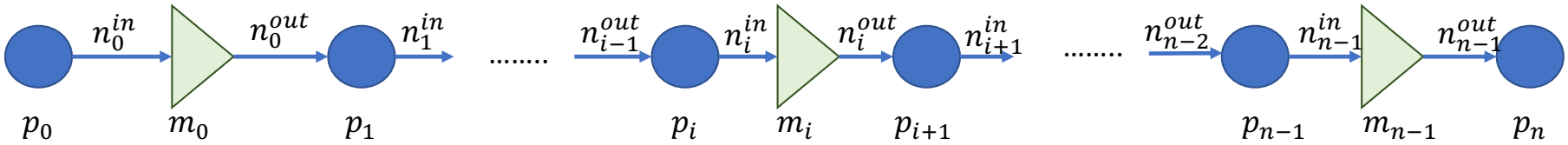
Negotiation in SCM Business

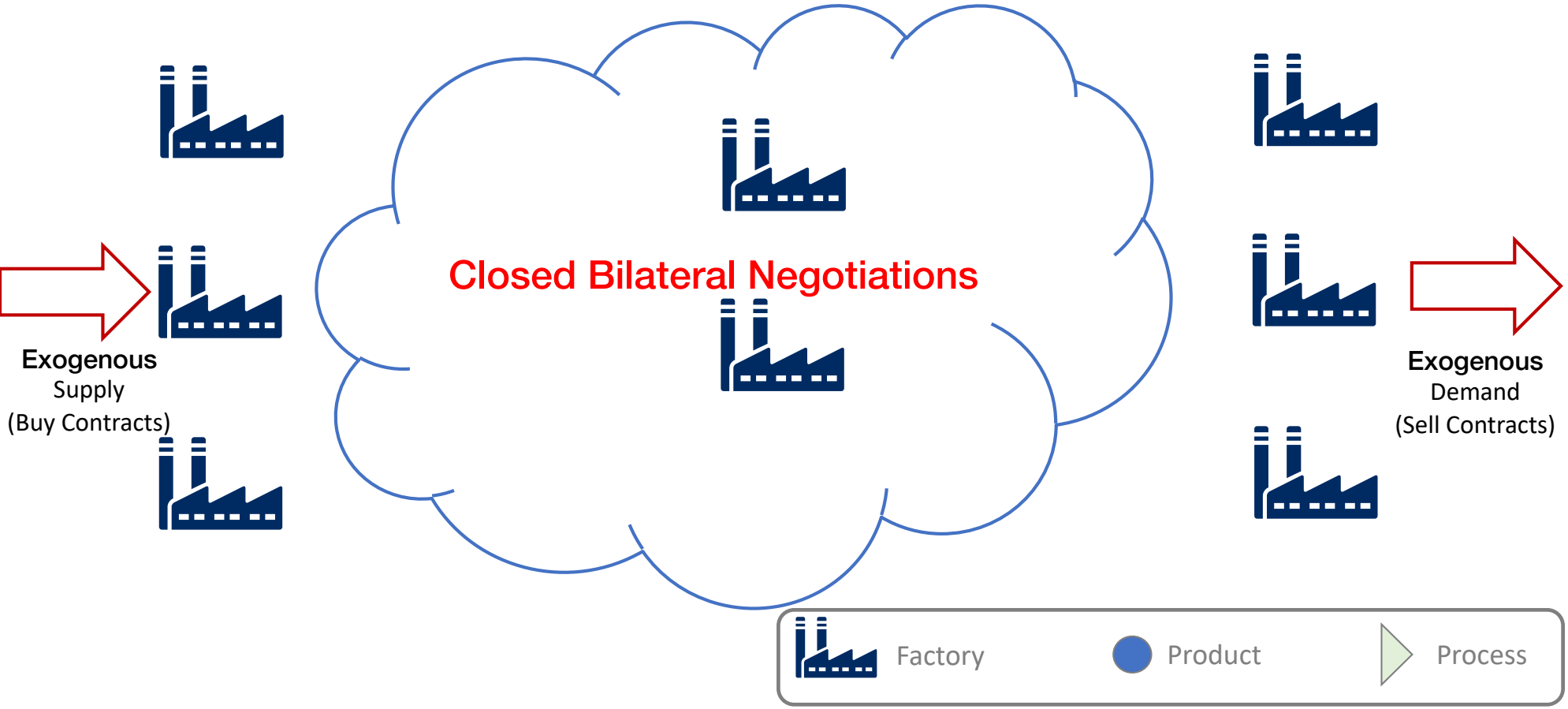
- Human negotiations lead to an estimated 17-40% *value leakage* in some estimates [2]
- A recent study suggests that at least 15 companies are working in *contracting support systems* [3].
- A recent UNECE UN/CEFACT proposal to standardize negotiation protocols for SCM and other applications [4]
- More to come [Mohammad et al., 2019].

[2] KPMG report: https://bit.ly/3kDRy6I
[3] Forrester report: https://bit.ly/3nwXEaY
[4] UN/CEFACT Project website: https://bit.ly/38LOsLX

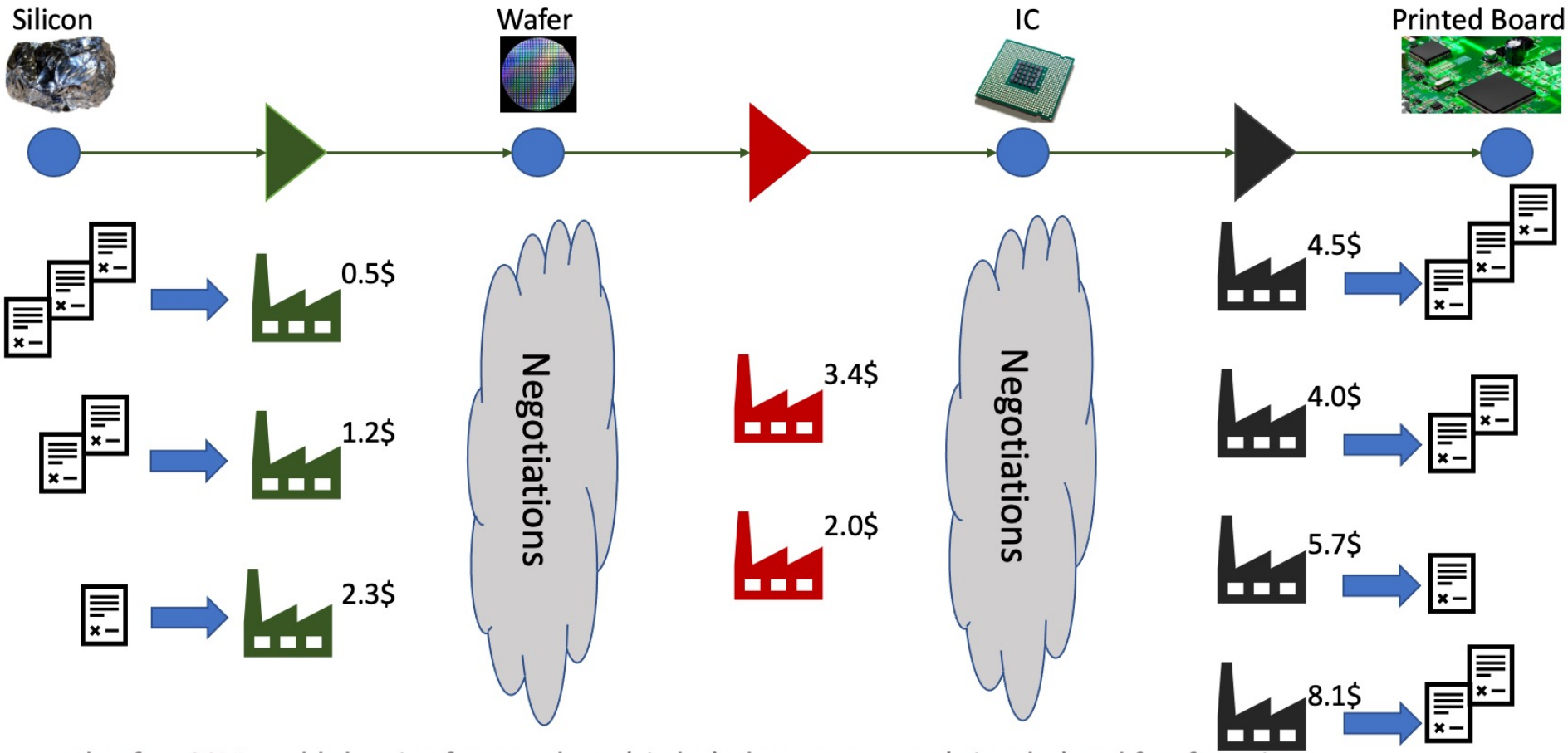
Outline

SCML Competition [Mohammad et al., 2019]





Example Configuration



An example of an SCM world showing four products (circles), three processes (triangles) and few factories.
Each process consumes one item of its input and generates one output of its output in one day.
Each factory requires a different cost to run its process (shown in its top right).
Factories in the first level have exogenous contracts to buy raw material (silicon) and factories at the last level have exogenous contracts to sell the final product (printed boards). These contracts drive the market.

SCML World

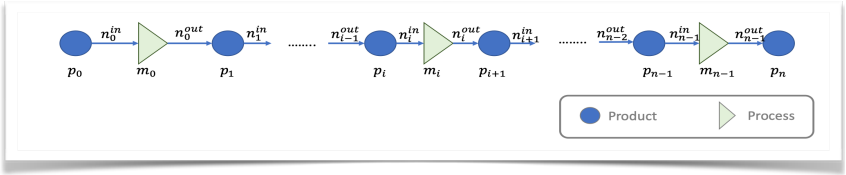
NEC-AIST AI Cooperative Research Laboratory

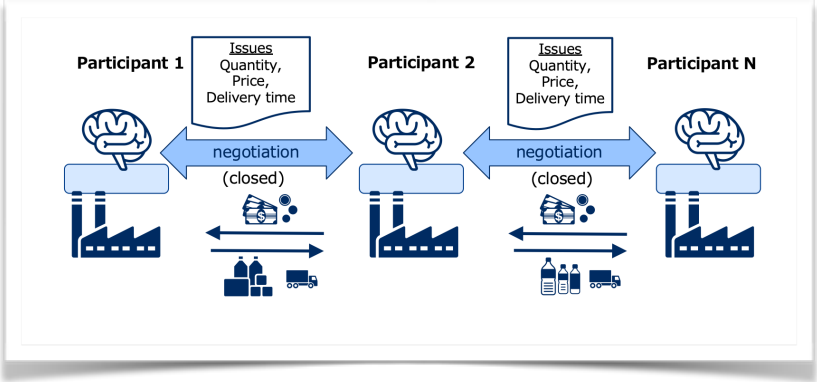
Challenge

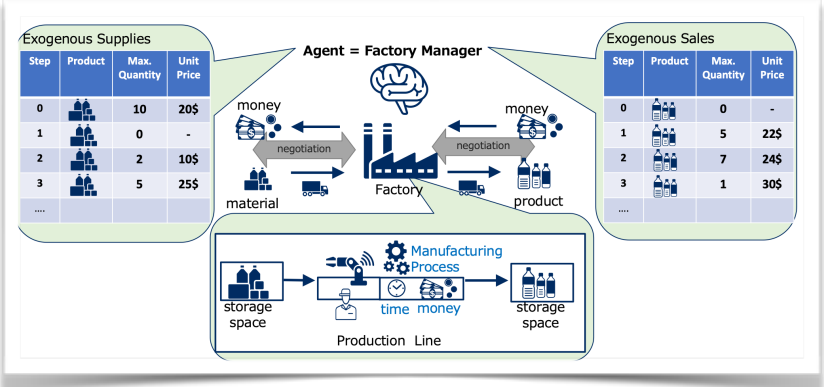
- Turn maximize profit into a ufun!!
- Dynamic interdependent ufuns.
- Sequential negotiations.
- Concurrent Negotiations.
- Negotiation under uncertainty.
- Adaptation and learning.
- Trust management.

Information

- **Website** https://scml.cs.brown.edu/
- **Code** https://www.github.com/yasserfarouk/scml
- **Youtube** https://www.youtube.com/playlist?list=PLqvs51K2Mb8IJe5Yz5jmYrRAwvIpGU2nF







SCML Competition

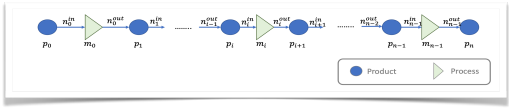
NEC-AIST AI Cooperative Research Laboratory

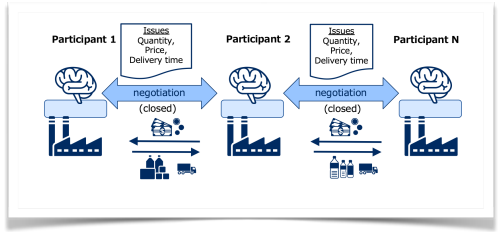
Competition Details

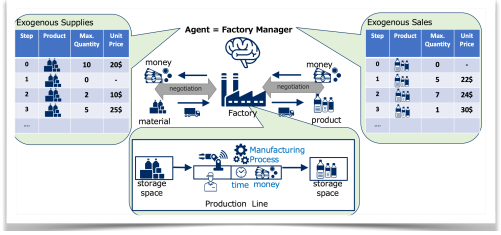
- Runs as part of ANAC IJCAI.
- You control one or more factories.
  - **Standard track** → one factory.
  - **Collusion track** → multiple factories (3).

Score Evaluation

- **Per instantiation**: Total profit counting inventory at half the trading price.
- **Total**: median of per-instantiation scores.

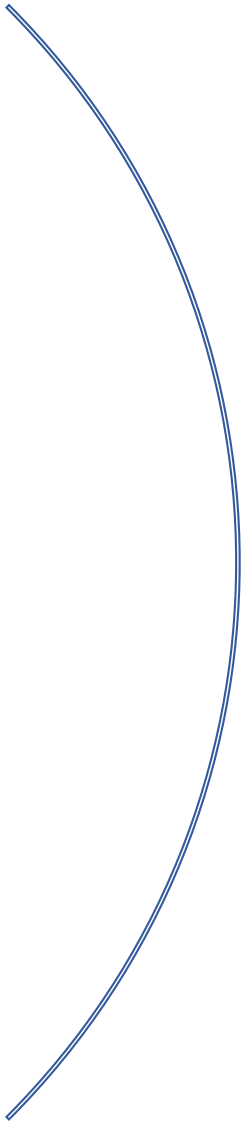






Simulation Steps

Simulation Steps



Simulation Steps

Simulation Steps

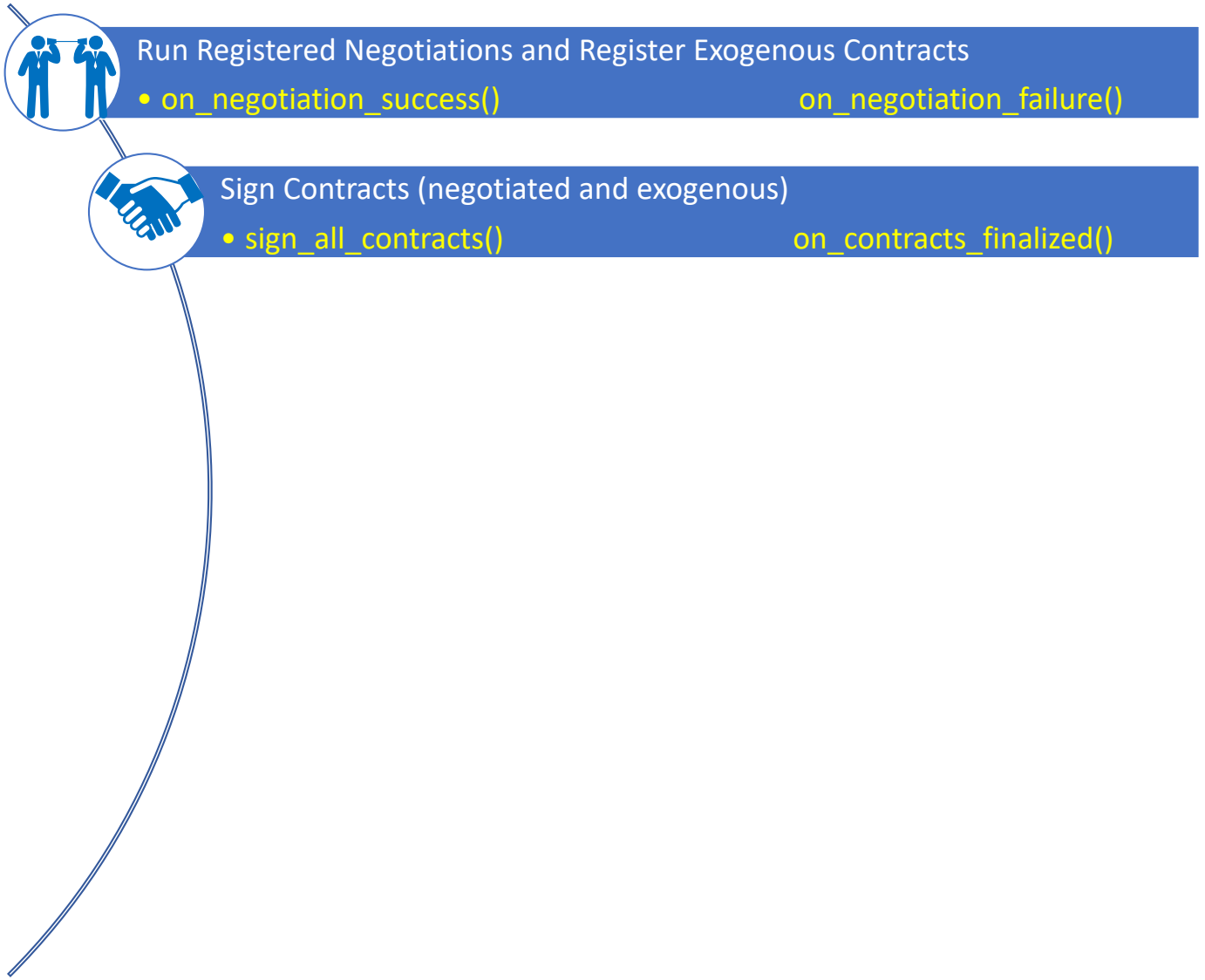
Run Registered Negotiations and Register Exogenous Contracts

- on_negotiation_success() on_negotiation_failure()

Simulation Steps

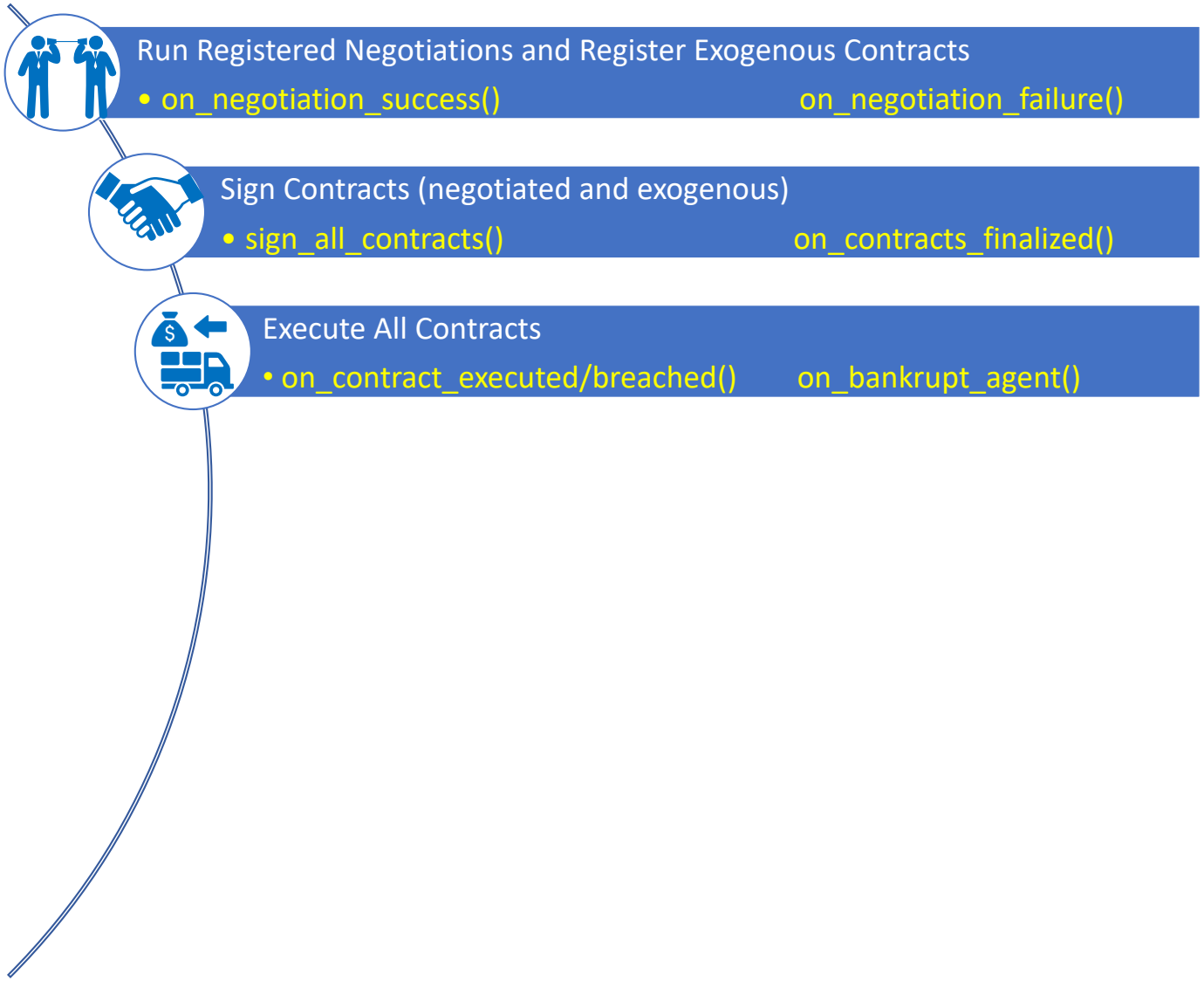
NEC-AIST AI Cooperative Research Laboratory

Simulation Steps



Simulation Steps

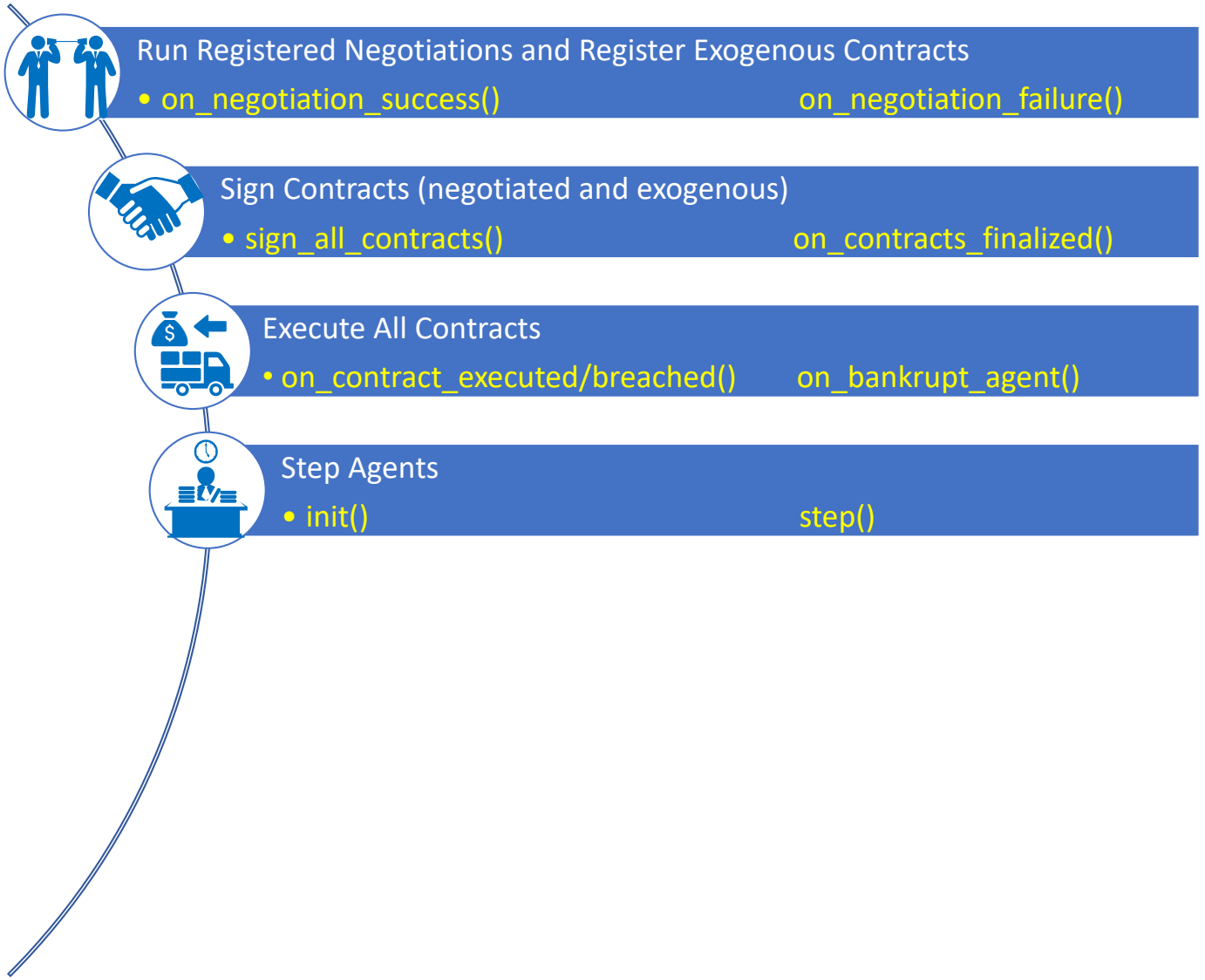
Simulation Steps



Simulation Steps

NEC-AIST AI Cooperative Research Laboratory

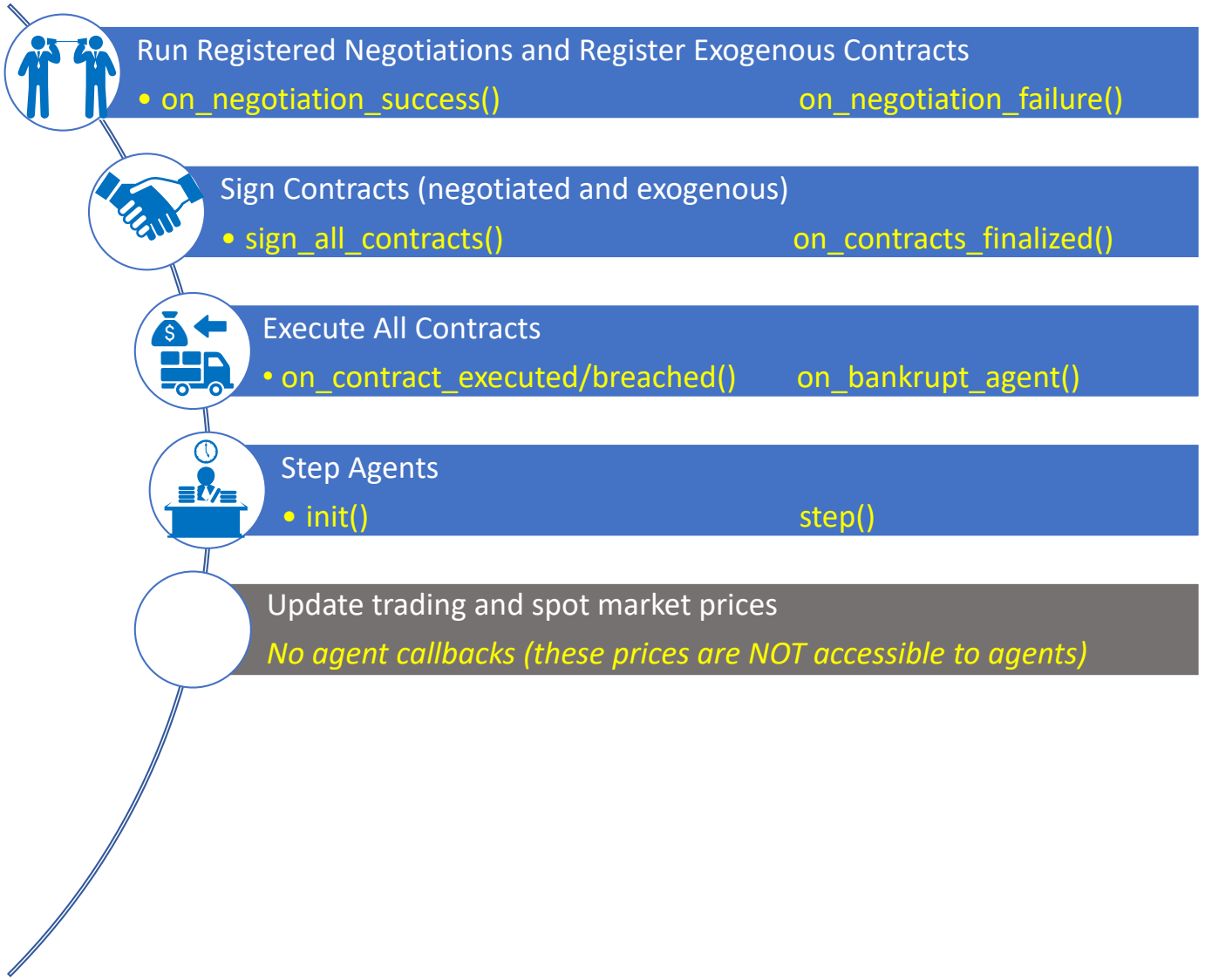
Simulation Steps



Simulation Steps

NEC-AIST AI Cooperative Research Laboratory

Simulation Steps



Simulation Steps

NEC-AIST AI Cooperative Research Laboratory

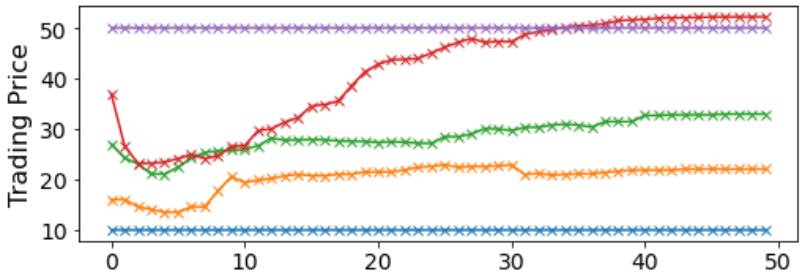
Simulation Steps

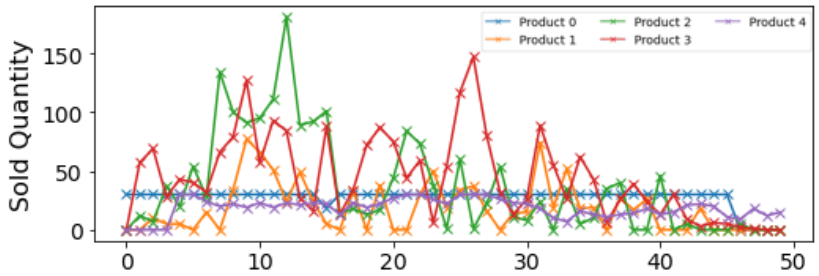
Simulation Steps

NEC-AIST AI Cooperative Research Laboratory

Simulation Steps

Trading prices





What is trading price and why is it calculated?

- A value calculated by the system for each product.
- Represents some estimate of the current price.
- Never revealed to agents.
- Usages:
  - Used at the end to value inventory.
  - Used when calculating spot-market price. during breach processing.

How does the system calculate it?

$$\text{tp}(p,s) = \frac{\beta^{s+1}\ Q_{-1}(p)\ \text{cat}(p) + \sum_{i=0}^{s} \beta^{s-i}\ Q_i(p)\ \mu_i(p)}{\beta^{s+1} + \sum_{i=0|Q_i(p)>0}^{s} \beta^{s-i}} \quad ,$$

Trading prices: The details

Quantities and prices

$$Q_i(p') = \sum_{\{c \in C^i | c.p = p'\}} c.\bar{q}$$

$$\mu_i(p') = \frac{\sum_{\{c \in C^i | c.p = p'\}} c.\bar{q} \times c.u}{Q_i(p')}$$

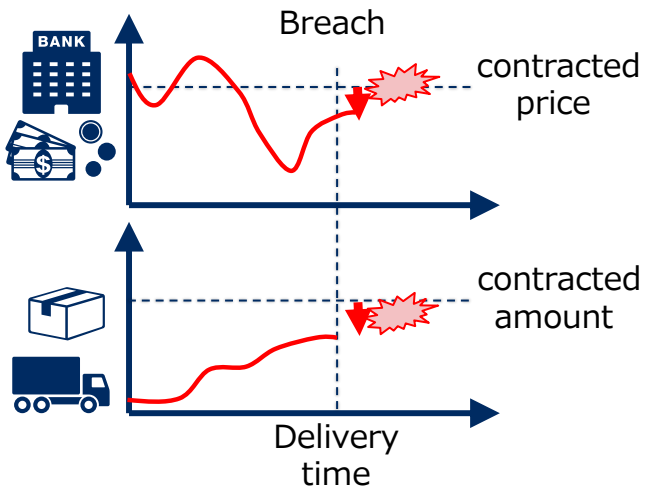
How does the system calculate it?

$$\text{tp}(p, s) = \frac{\beta^{s+1}\ Q_{-1}(p)\ \text{cat}(p) + \sum_{i=0}^{s} \beta^{s-i}\ Q_i(p)\ \mu_i(p)}{\beta^{s+1} + \sum_{i=0 | Q_i(p) > 0}^{s} \beta^{s-i}} \quad ,$$

When things go wrong

What is a breach

- Insufficient funds or insufficient inventory.



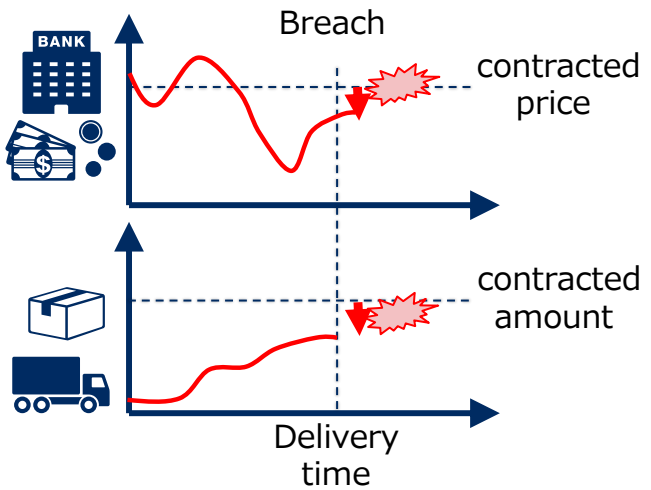
When things go wrong

What is a breach

- Insufficient funds or insufficient inventory.

Breach Processing

- A breach report is **always** published.
  - who and fraction.



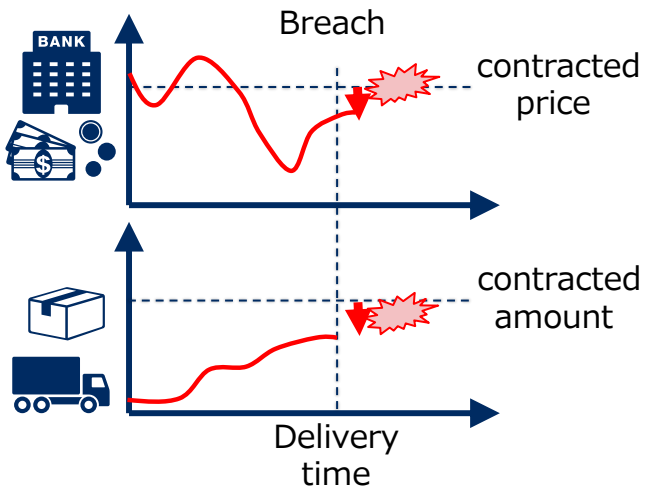
When things go wrong

What is a breach

- Insufficient funds or insufficient inventory.

Breach Processing

- A breach report is **always** published.
  - who and fraction.
- Insufficient products $\rightarrow$ **forced to** buy from the spot market.



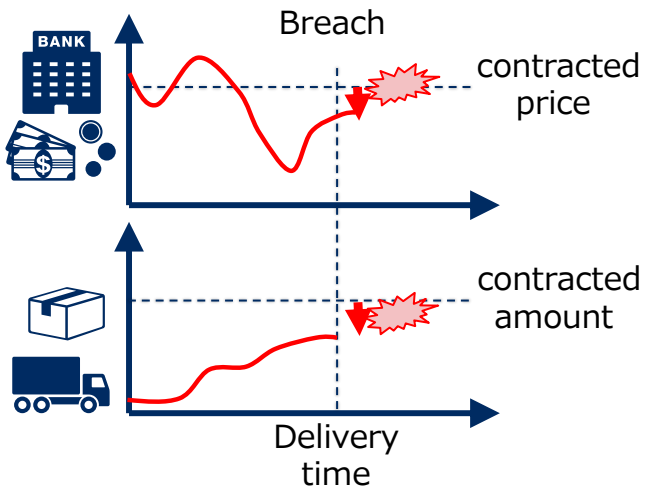
When things go wrong

What is a breach

- Insufficient funds or insufficient inventory.

Breach Processing

- A breach report is **always** published.
  - who and fraction.
- Insufficient products → **forced to** buy from the spot market.
- Insufficient funds → bankruptcy.



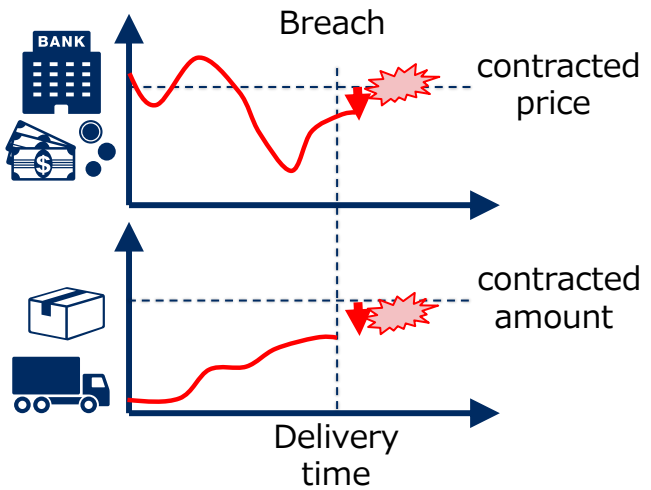
When things go wrong

What is a breach

- Insufficient funds or insufficient inventory.

Breach Processing

- A breach report is **always** published.
  - who and fraction.
- Insufficient products → **forced to** buy from the spot market.
- Insufficient funds → bankruptcy.
  - bankruptcy → liquidation.



Spot Market

Spot market

- **No-choice:** Can be used **only** for insufficient inventory breaches.

Calculation

Spot Market

Spot market

- **No-choice:** Can be used **only** for insufficient inventory breaches.
- **Penalizing:** Entails higher price than the trading price

Calculation

Spot Market

Spot market

- **No-choice:** Can be used **only** for insufficient inventory breaches.
- **Penalizing:** Entails higher price than the trading price
- **Personalized:** More breaches $\rightarrow$ higher spot price **for you**

Calculation

Spot Market

Spot market

- **No-choice:** Can be used **only** for insufficient inventory breaches.
- **Penalizing:** Entails higher price than the trading price
- **Personalized:** More breaches $\rightarrow$ higher spot price **for you**

Calculation

- Agent's spot price penalty: $\text{ip}_a(p, s) = \lambda \sum_{i=0}^{s} \alpha^{s-i} q_a(p, i)$,

Spot Market

NEC-AIST AI Cooperative Research Laboratory

Spot market

- **No-choice:** Can be used **only** for insufficient inventory breaches.
- **Penalizing:** Entails higher price than the trading price
- **Personalized:** More breaches $\rightarrow$ higher spot price **for you**

Calculation

- Agent's spot price penalty: $\text{ip}_a(p, s) = \lambda \sum_{i=0}^{s} \alpha^{s-i} q_a(p, i)$,
- Breaching agents buy **expensive**:

$$\text{tp}(p, s) \times (1 + \text{gp}) \times (1 + \text{ip}_a(p, s))$$

Spot Market

Spot market

- **No-choice:** Can be used **only** for insufficient inventory breaches.
- **Penalizing:** Entails higher price than the trading price
- **Personalized:** More breaches $\rightarrow$ higher spot price **for you**

Calculation

- Agent's spot price penalty: $\text{ip}_a(p, s) = \lambda \sum_{i=0}^{s} \alpha^{s-i} q_a(p, i)$,
- Breaching agents buy **expensive**:

$$\text{tp}(p, s) \times (1 + \text{gp}) \times (1 + \text{ip}_a(p, s))$$

- Bankrupt agents are liquidated **cheap**:

$$\text{tp}(p, s)/((1 + \text{gp}) \times (1 + \text{ip}_a(p, s))$$

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.

What exactly happens?

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.
- Insufficient money to buy from the spot market for as sell contract.

What exactly happens?

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.
- Insufficient money to buy from the spot market for as sell contract.

What exactly happens?

1. The agent is stopped from every buying or selling.

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.
- Insufficient money to buy from the spot market for as sell contract.

What exactly happens?

1. The agent is stopped from every buying or selling.
2. Its inventory is sold on the spot market.

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.
- Insufficient money to buy from the spot market for as sell contract.

What exactly happens?

1. The agent is stopped from every buying or selling.
2. Its inventory is sold on the spot market.
3. All agents are informed.

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.
- Insufficient money to buy from the spot market for as sell contract.

What exactly happens?

1. The agent is stopped from every buying or selling.
2. Its inventory is sold on the spot market.
3. All agents are informed.
4. Agents with future contracts with it are informed about the expected level of breach.

Bankruptcy Processing

Bankruptcy conditions

- Insufficient money for a buy contract.
- Insufficient money to buy from the spot market for as sell contract.

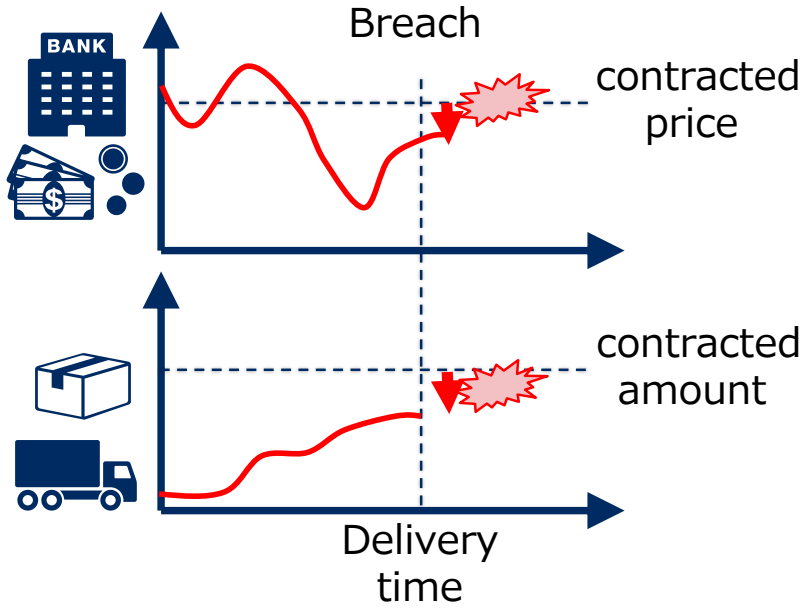
What exactly happens?

1. The agent is stopped from every buying or selling.
2. Its inventory is sold on the spot market.
3. All agents are informed.
4. Agents with future contracts with it are informed about the expected level of breach.
5. The agent's score is set to -1.

When things go wrong: Summary

Summary

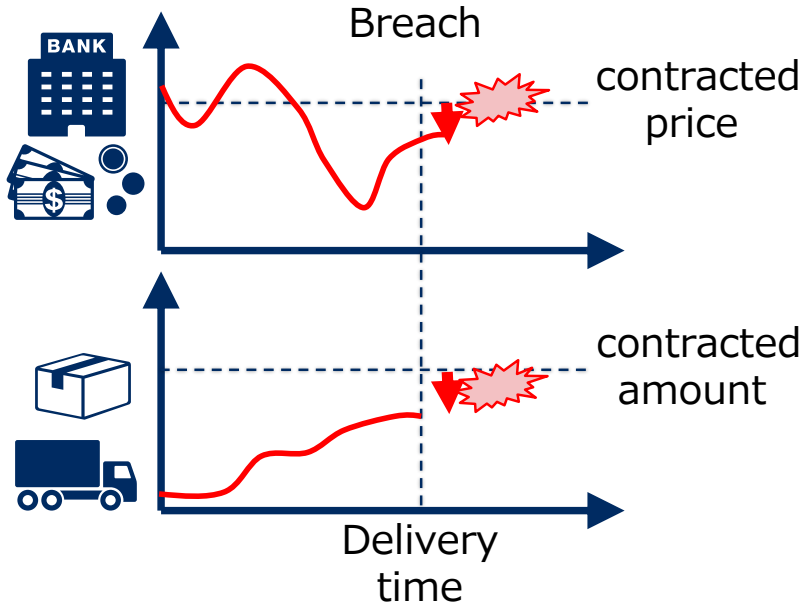
- An unfulfilled contract is reported to the **breach-list** (who and fraction).



When things go wrong: Summary

Summary

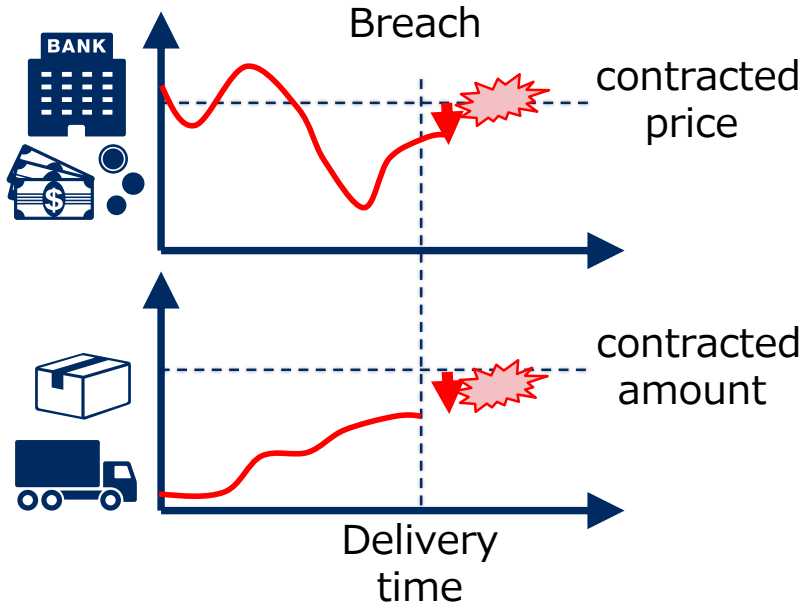
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds $\rightarrow$ bankrupt.



When things go wrong: Summary

Summary

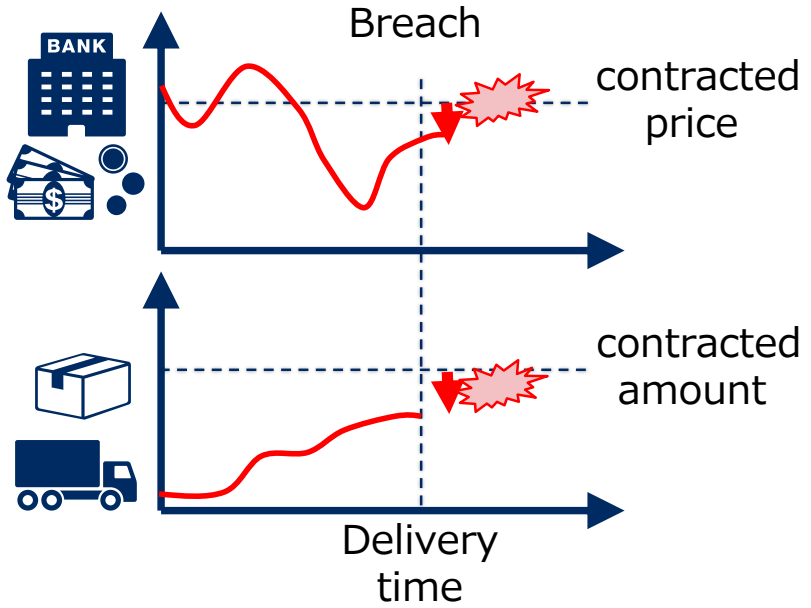
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.



When things go wrong: Summary

Summary

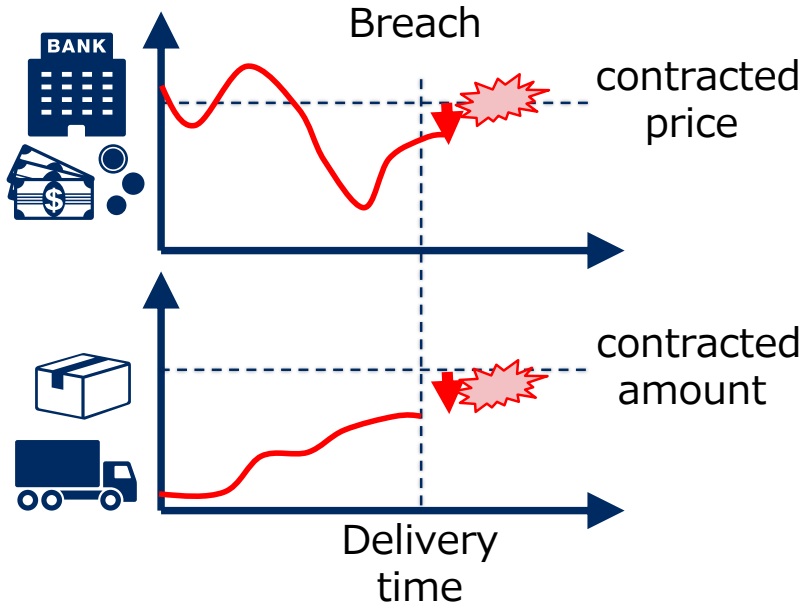
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.
  - Cannot buy → bankrupt.



When things go wrong: Summary

Summary

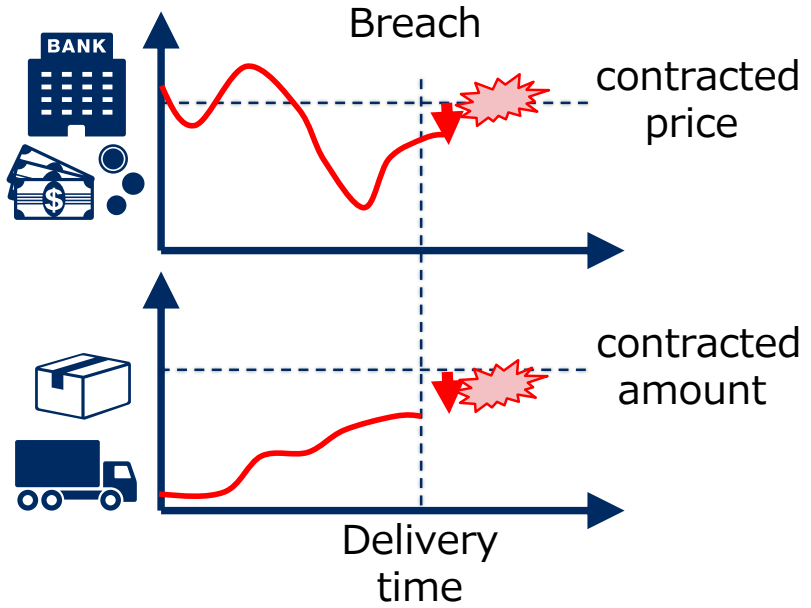
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.
  - Cannot buy → bankrupt.
- Bankrupt → **really really bad**



When things go wrong: Summary

Summary

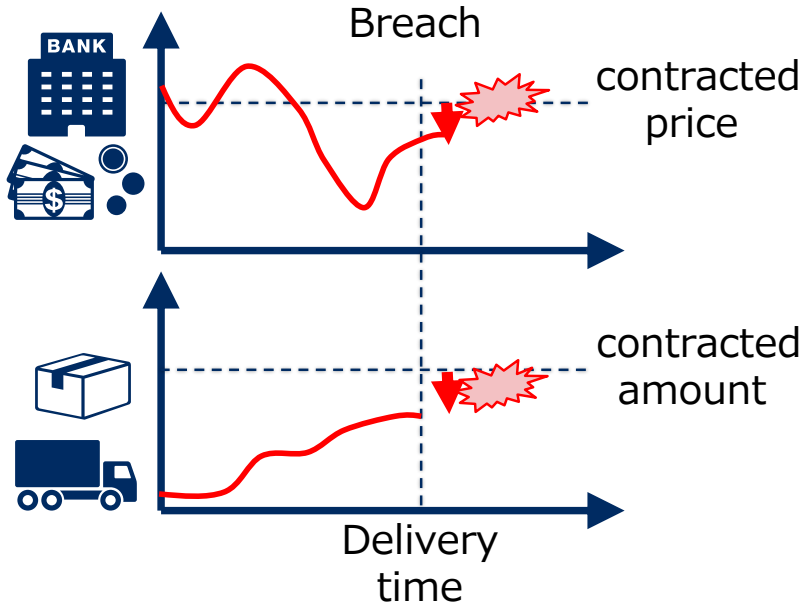
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.
  - Cannot buy → bankrupt.
- Bankrupt → **really really bad**
  - No more trade.



When things go wrong: Summary

Summary

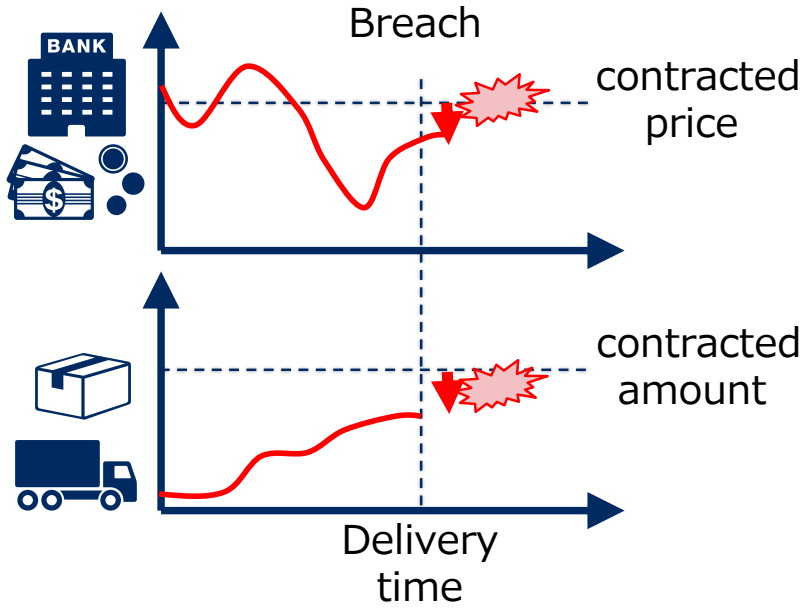
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.
  - Cannot buy → bankrupt.
- Bankrupt → **really really bad**
  - No more trade.
  - Very low score.



When things go wrong: Summary

Summary

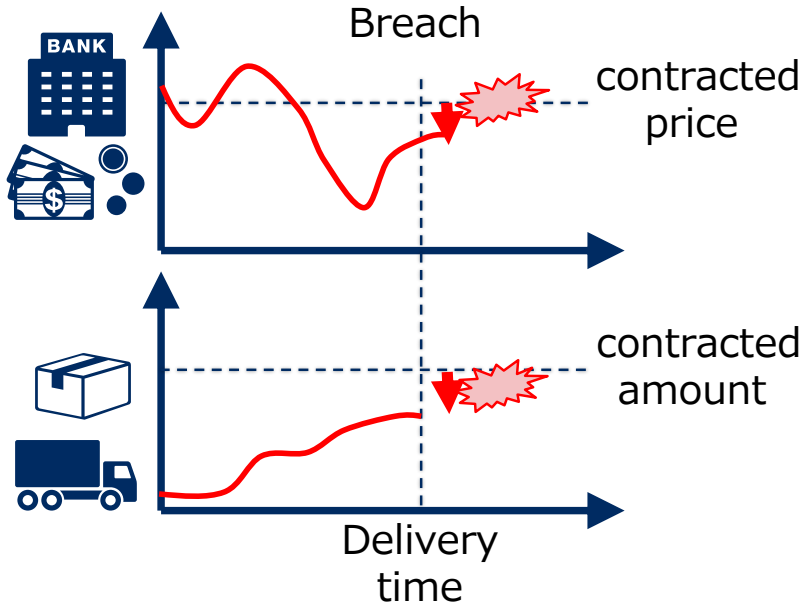
- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.
  - Cannot buy → bankrupt.
- Bankrupt → **really really bad**
  - No more trade.
  - Very low score.
  - All inventory is liquidated.



When things go wrong: Summary

Summary

- An unfulfilled contract is reported to the **breach-list** (who and fraction).
- Insufficient funds → bankrupt.
- Insufficient product → buy at high cost.
  - Cannot buy → bankrupt.
- Bankrupt → **really really bad**
  - No more trade.
  - Very low score.
  - All inventory is liquidated.
  - May hurt other agents.



Outline

Agent Knowledge

About itself

- **Its capabilities:** lines and production cost.
- **Its location:** input and output products.
- **Its partners:** suppliers and consumers (and competitors).
- **Its state:** inventory, wallet, contracts, and negotiations.

Agent Knowledge

About itself

- **Its capabilities:** lines and production cost.
- **Its location:** input and output products.
- **Its partners:** suppliers and consumers (and competitors).
- **Its state:** inventory, wallet, contracts, and negotiations.

About the market

- The production graph and factories at each level.
- Time and simulation length.

Agent Knowledge

About itself

- **Its capabilities:** lines and production cost.
- **Its location:** input and output products.
- **Its partners:** suppliers and consumers (and competitors).
- **Its state:** inventory, wallet, contracts, and negotiations.

About the market

- The production graph and factories at each level.
- Time and simulation length.

About others

- **Financial Reports:** balance, assets, breach fraction/probability.
- **Past Interactions:** negotiations and contracts between itself and that agent.

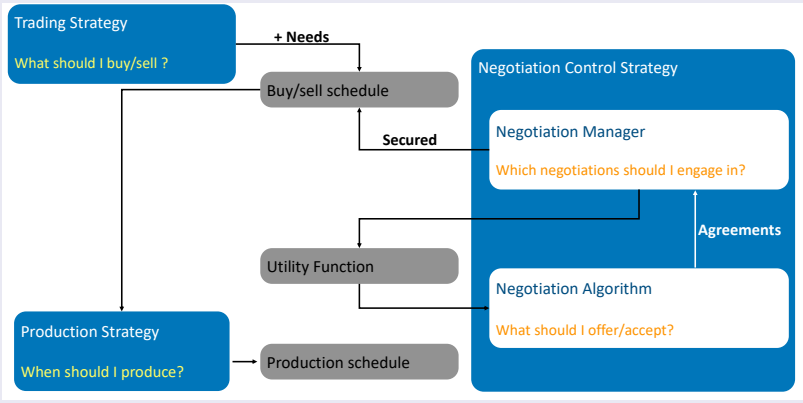
Development Approaches

Monolithic Agent

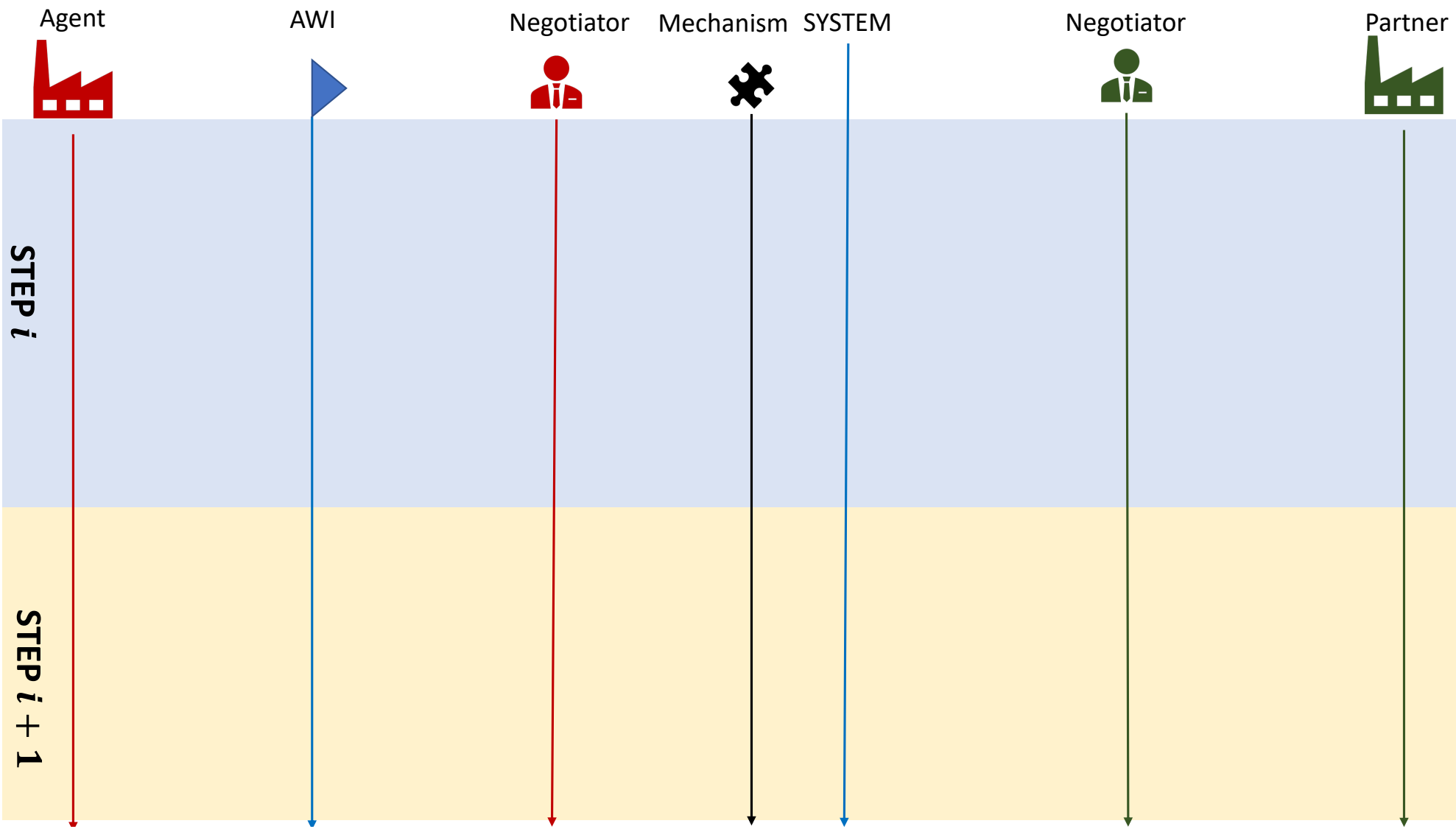
- Respond to callbacks in the Agent class.
- Functionality is distributed **among callbacks**.
- Everything is in one place (the agent class).
- Harder to reuse.

Component-Based Agent

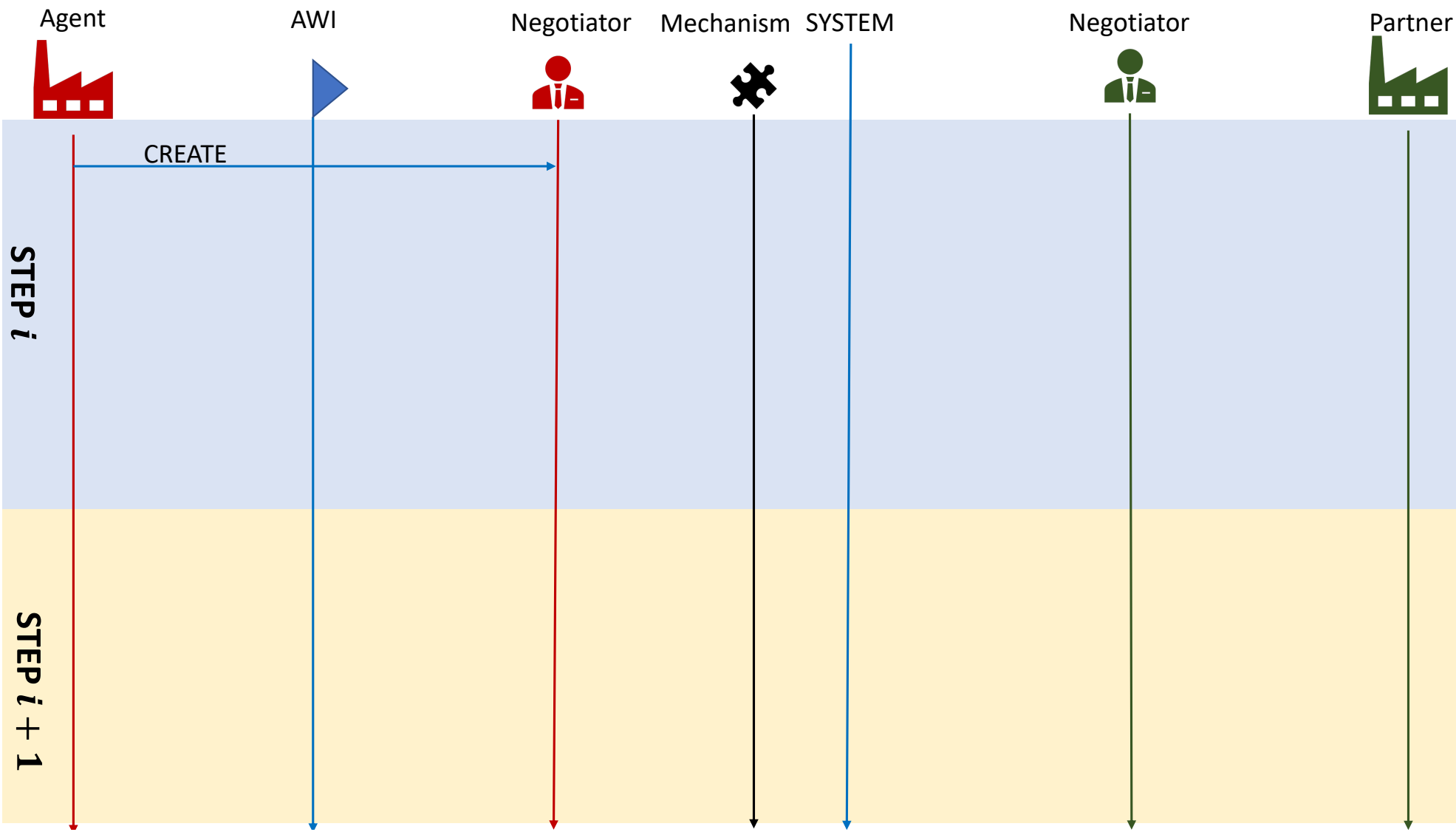
- Divides the agent into semi-independent components.
- Functionality is distributed between **components**.
- Easier to reuse.



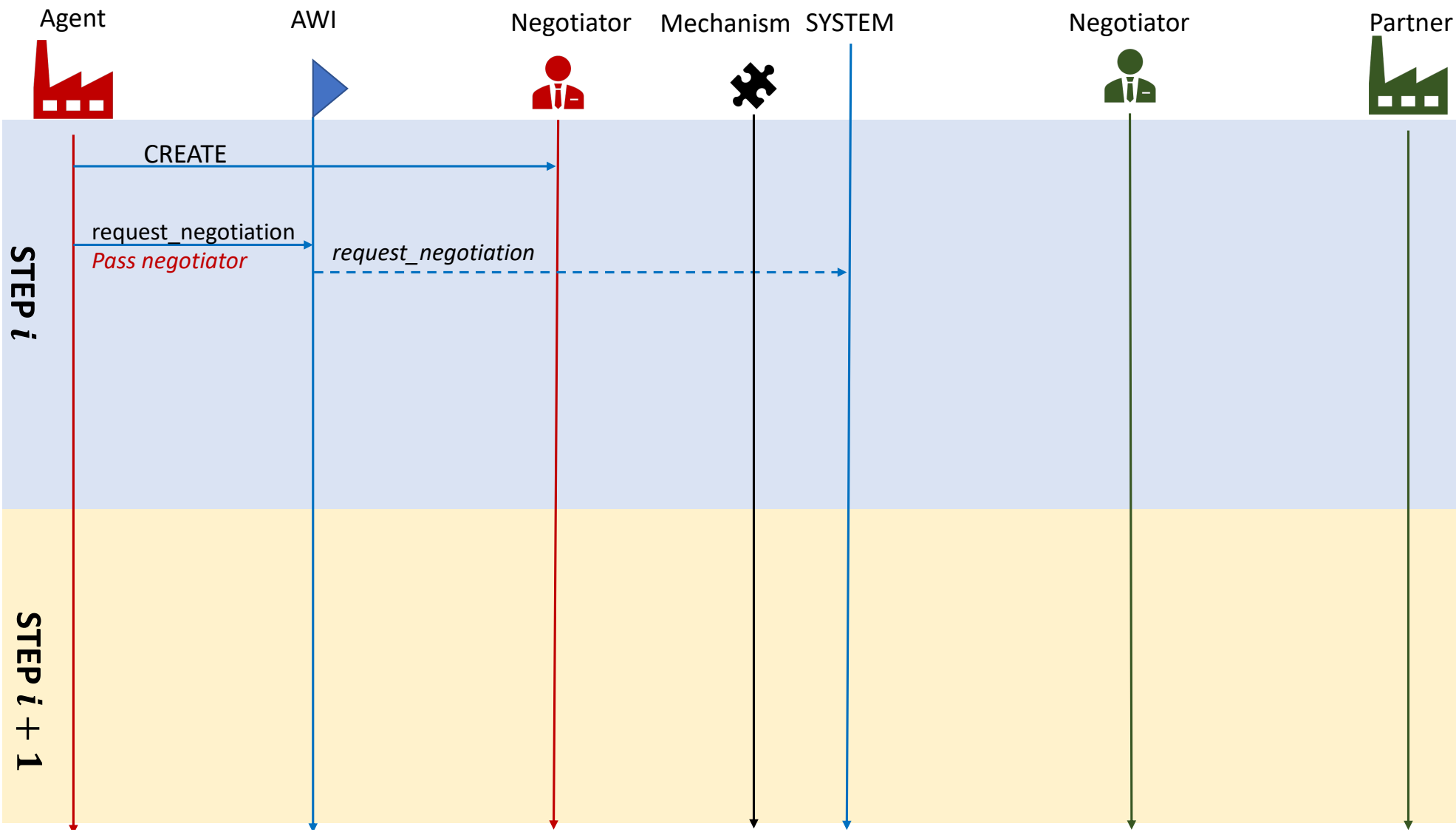
Callbacks and Timing



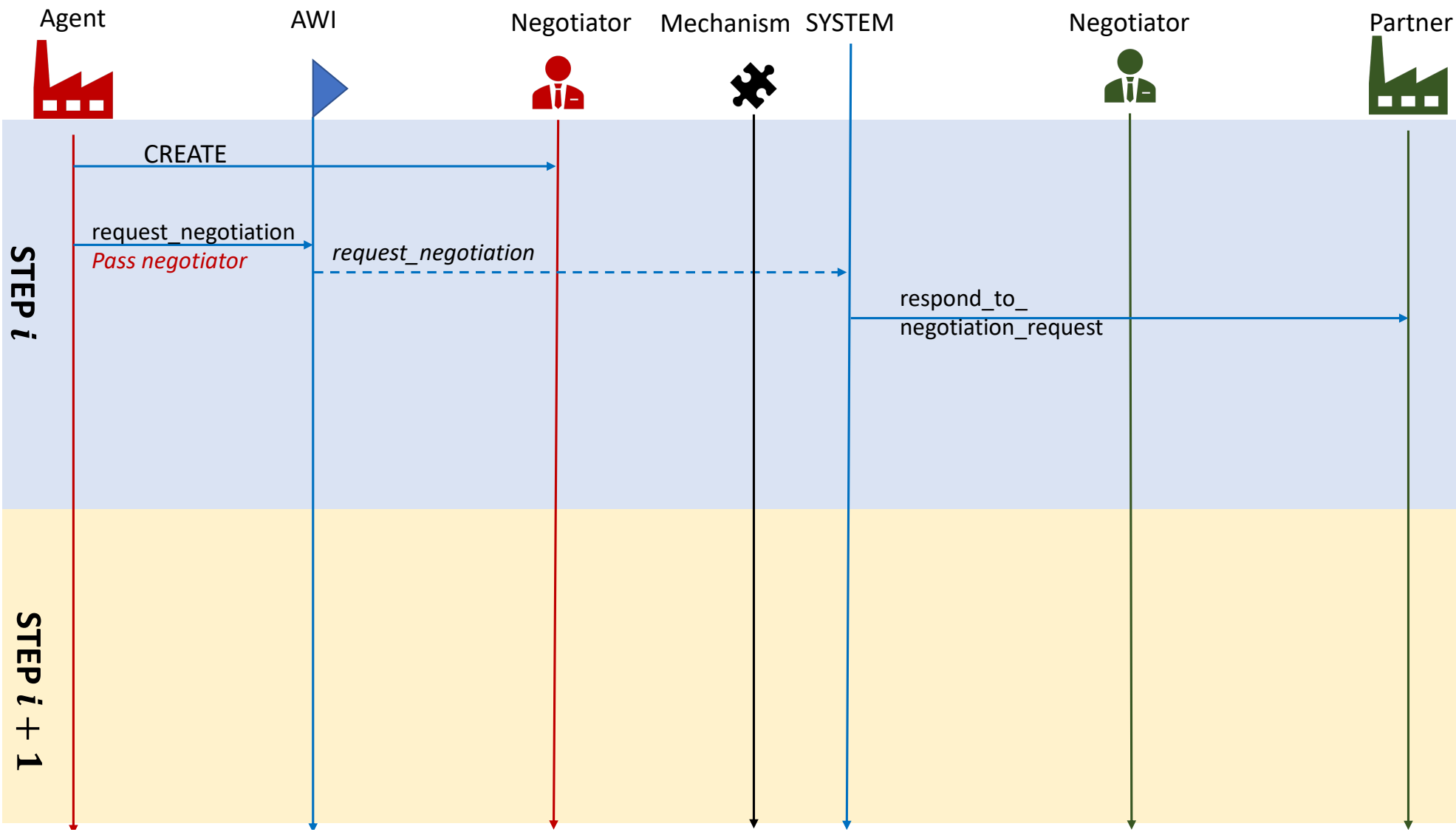
Callbacks and Timing



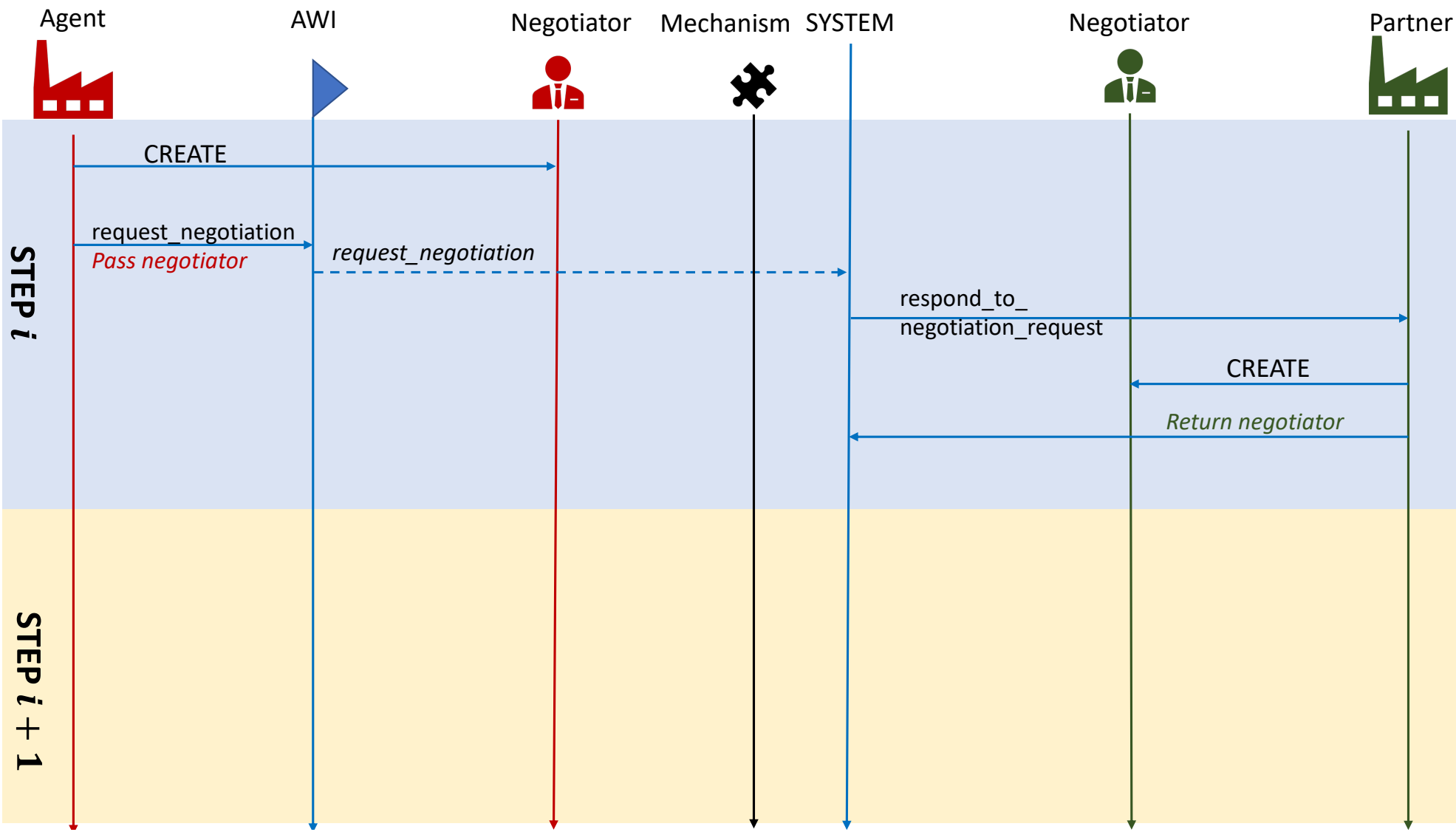
Callbacks and Timing



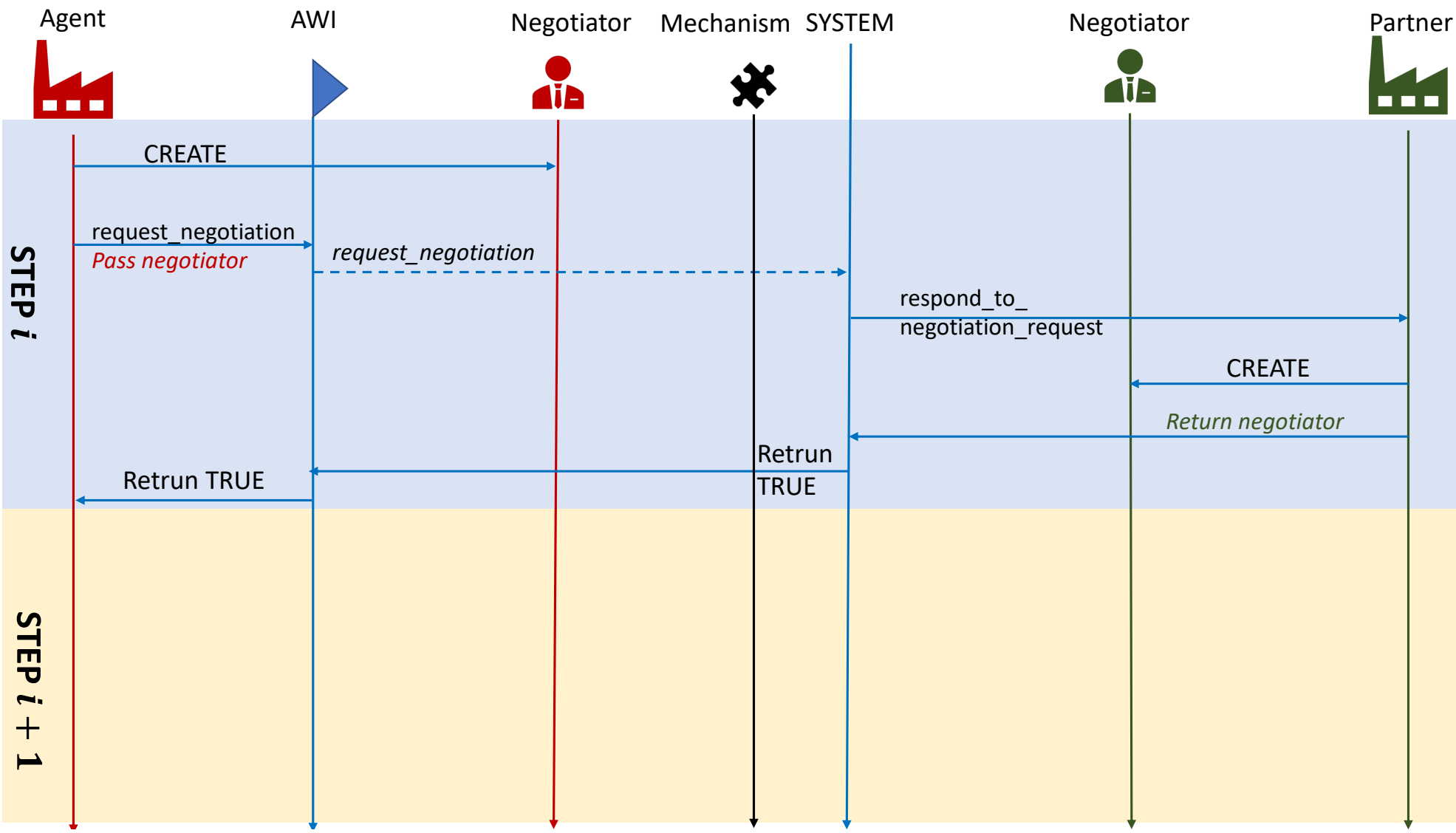
Callbacks and Timing



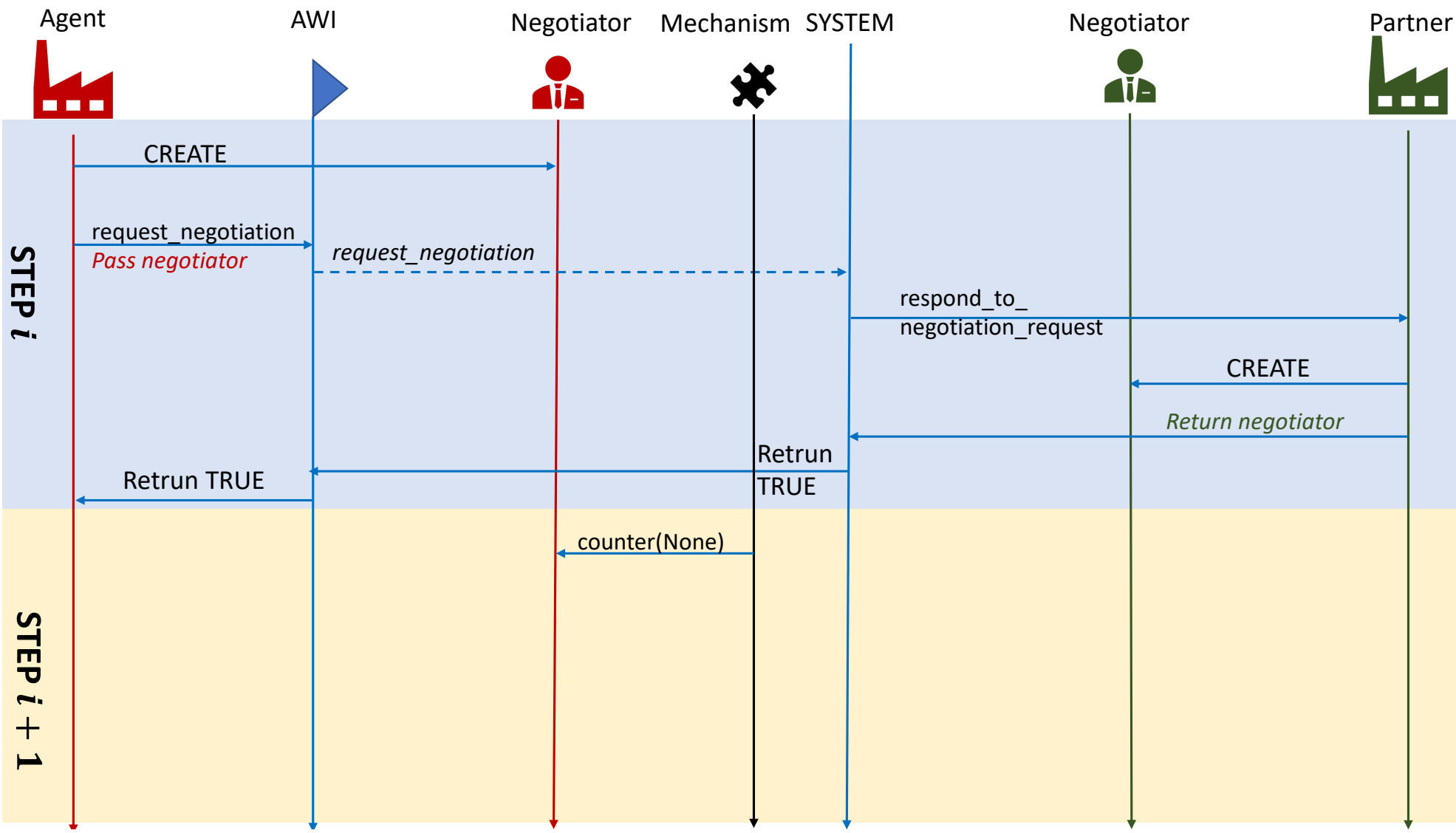
Callbacks and Timing



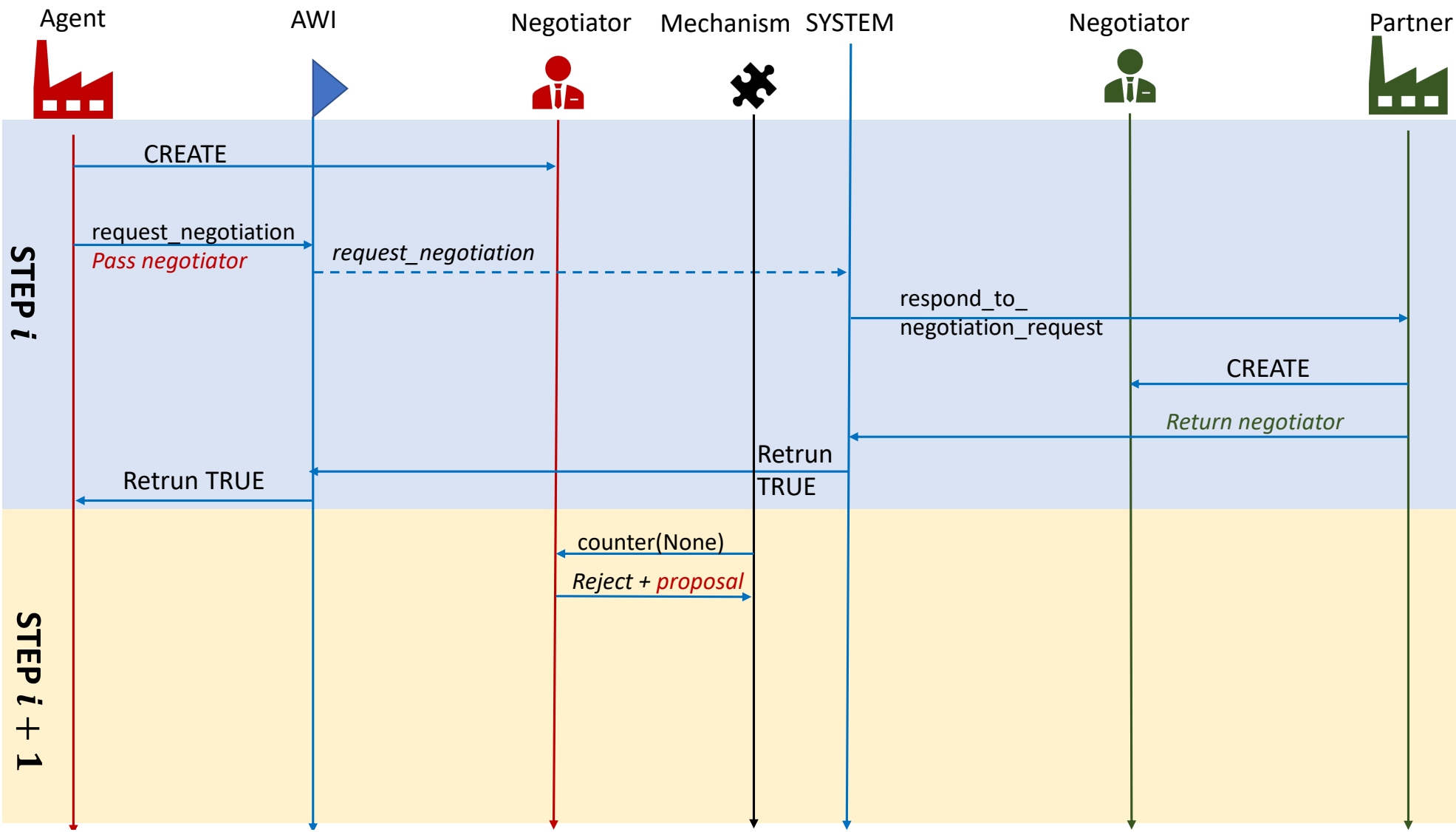
Callbacks and Timing



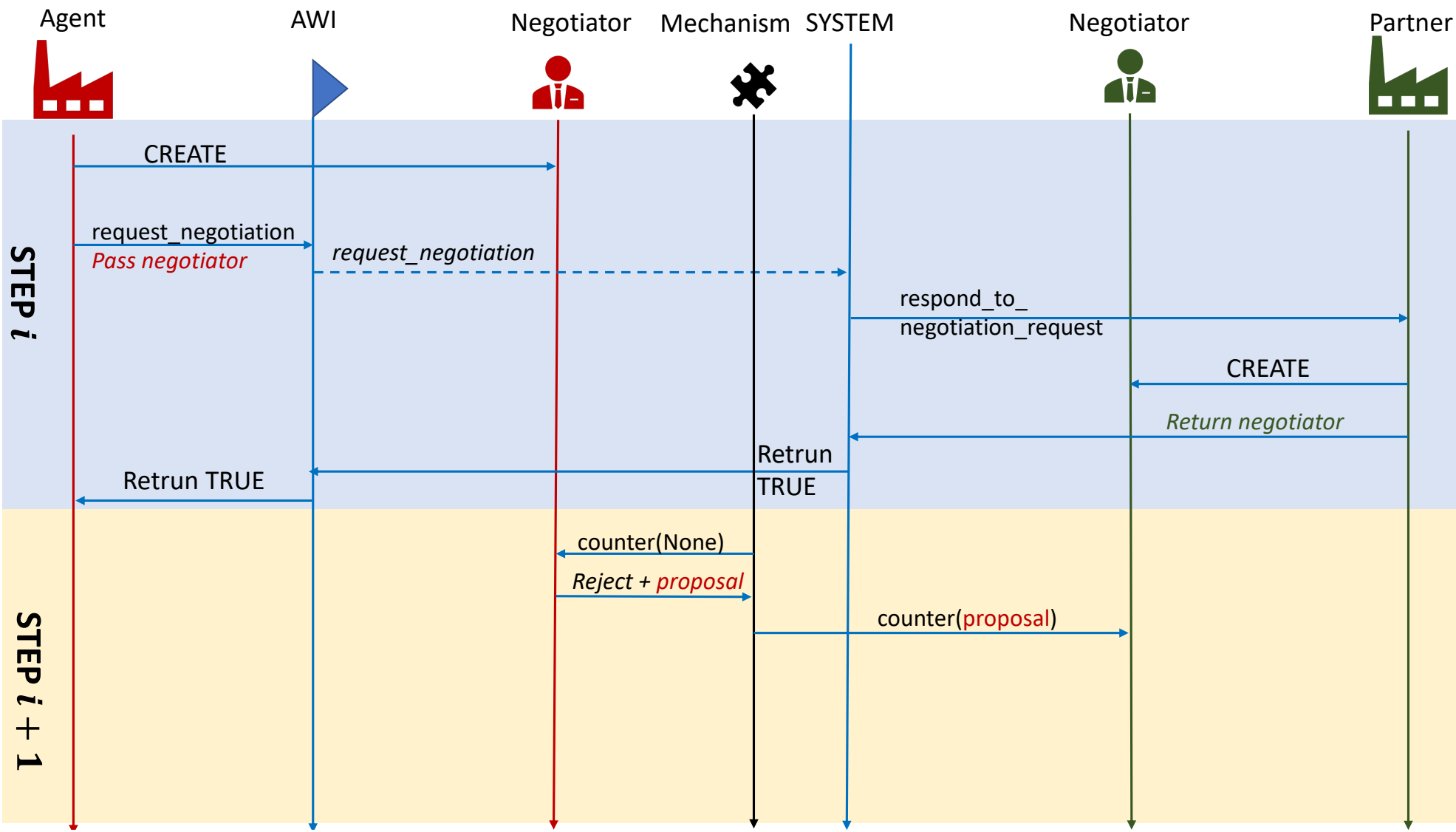
Callbacks and Timing



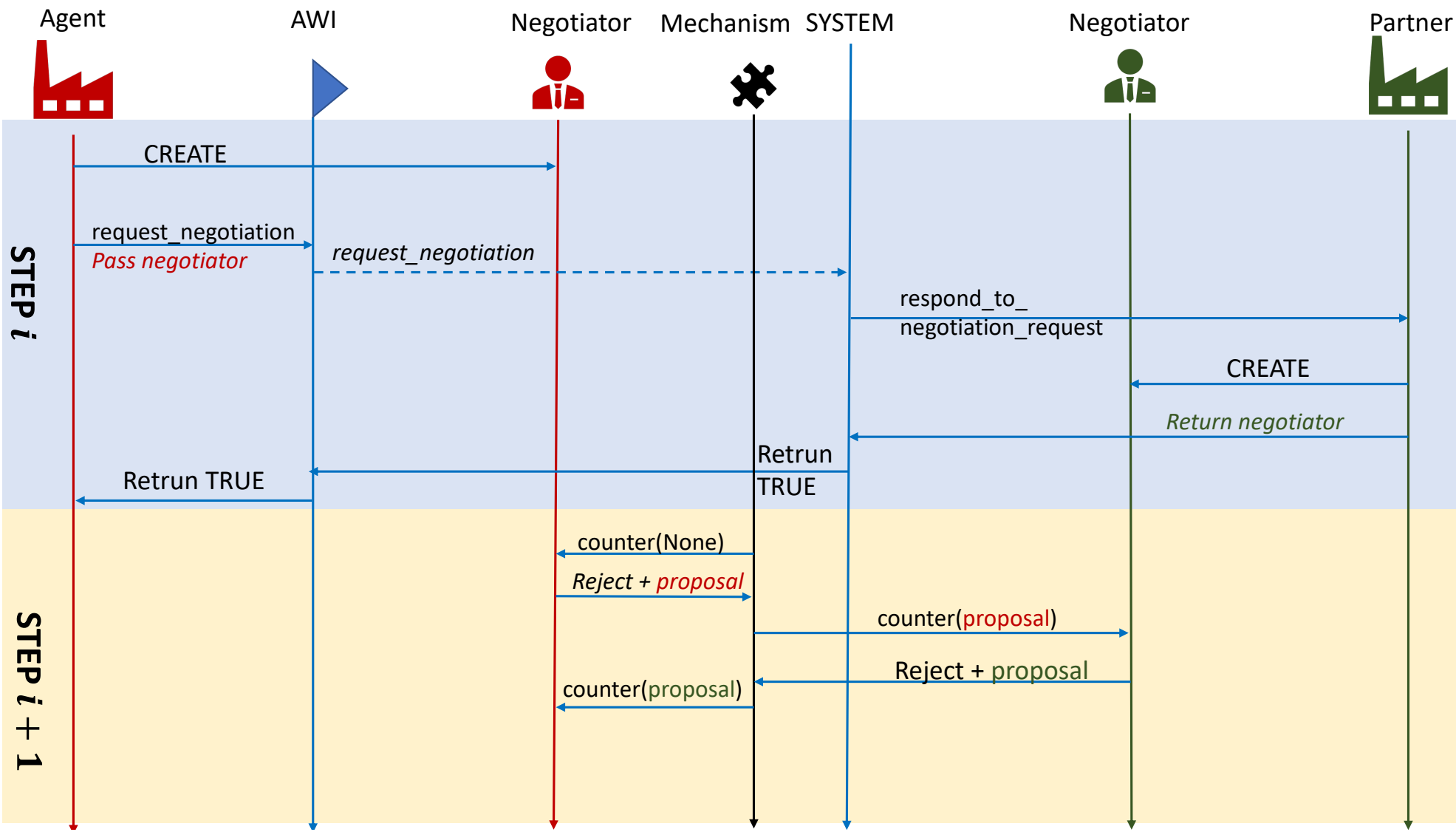
Callbacks and Timing



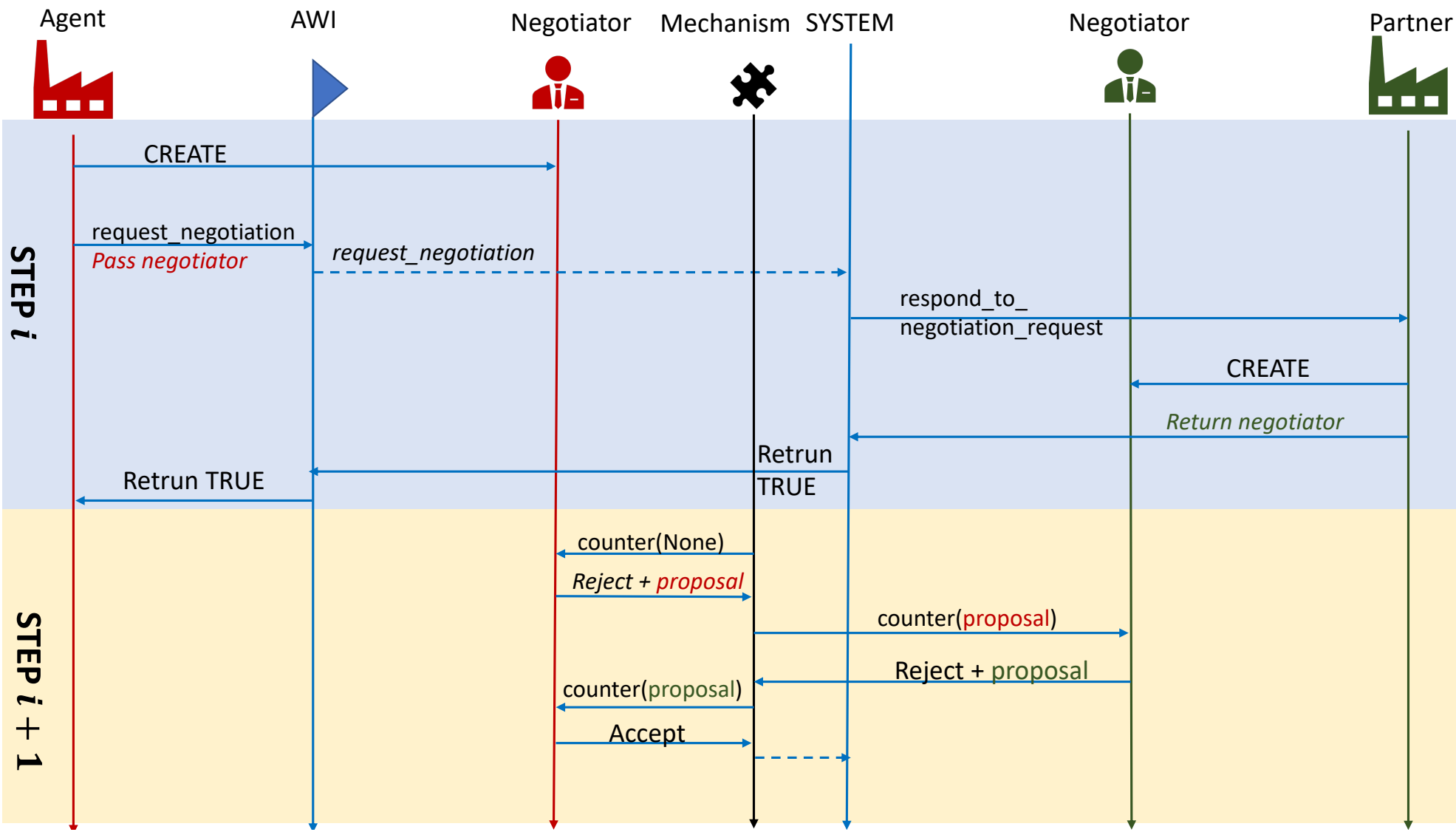
Callbacks and Timing



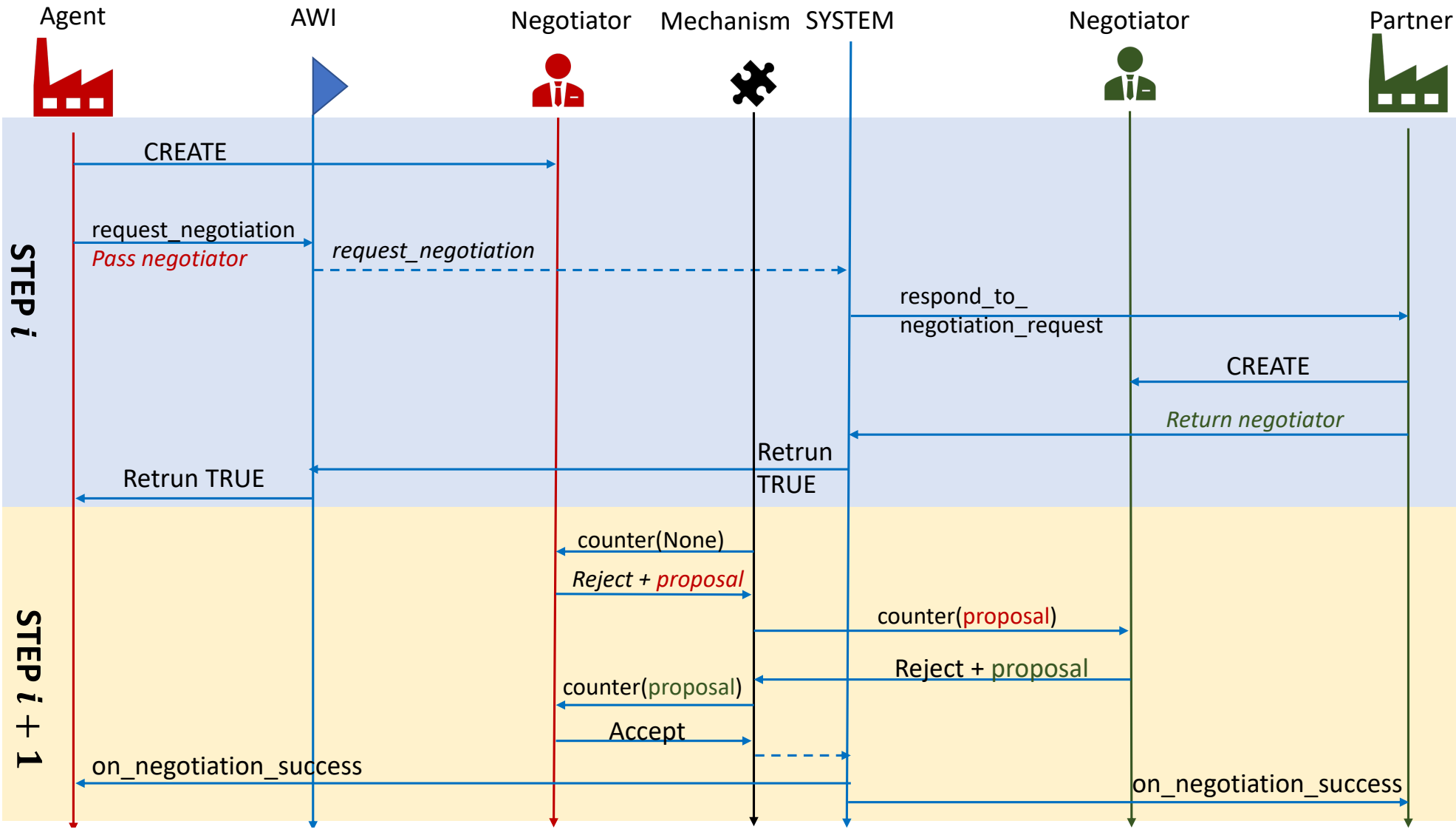
Callbacks and Timing



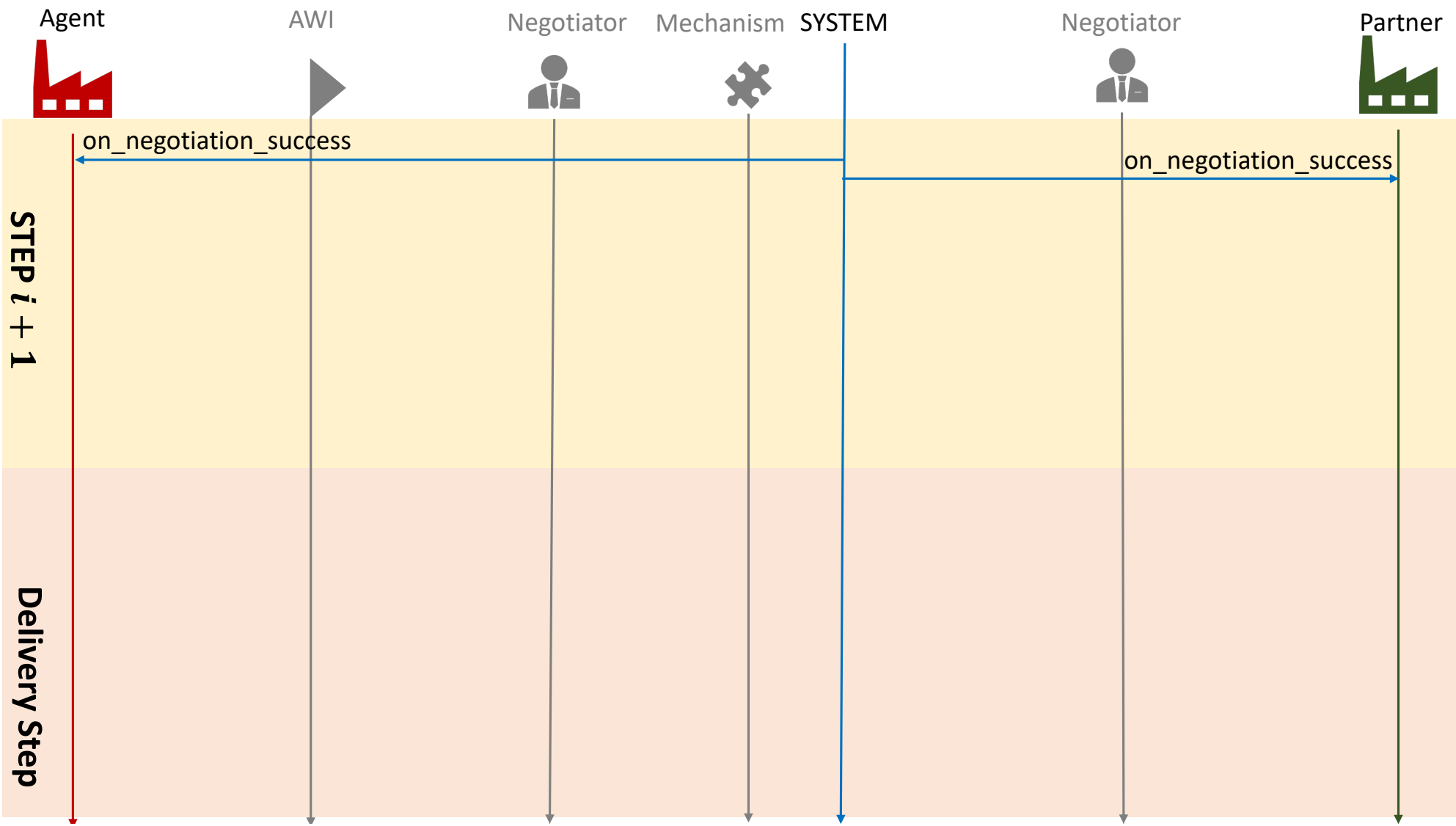
Callbacks and Timing



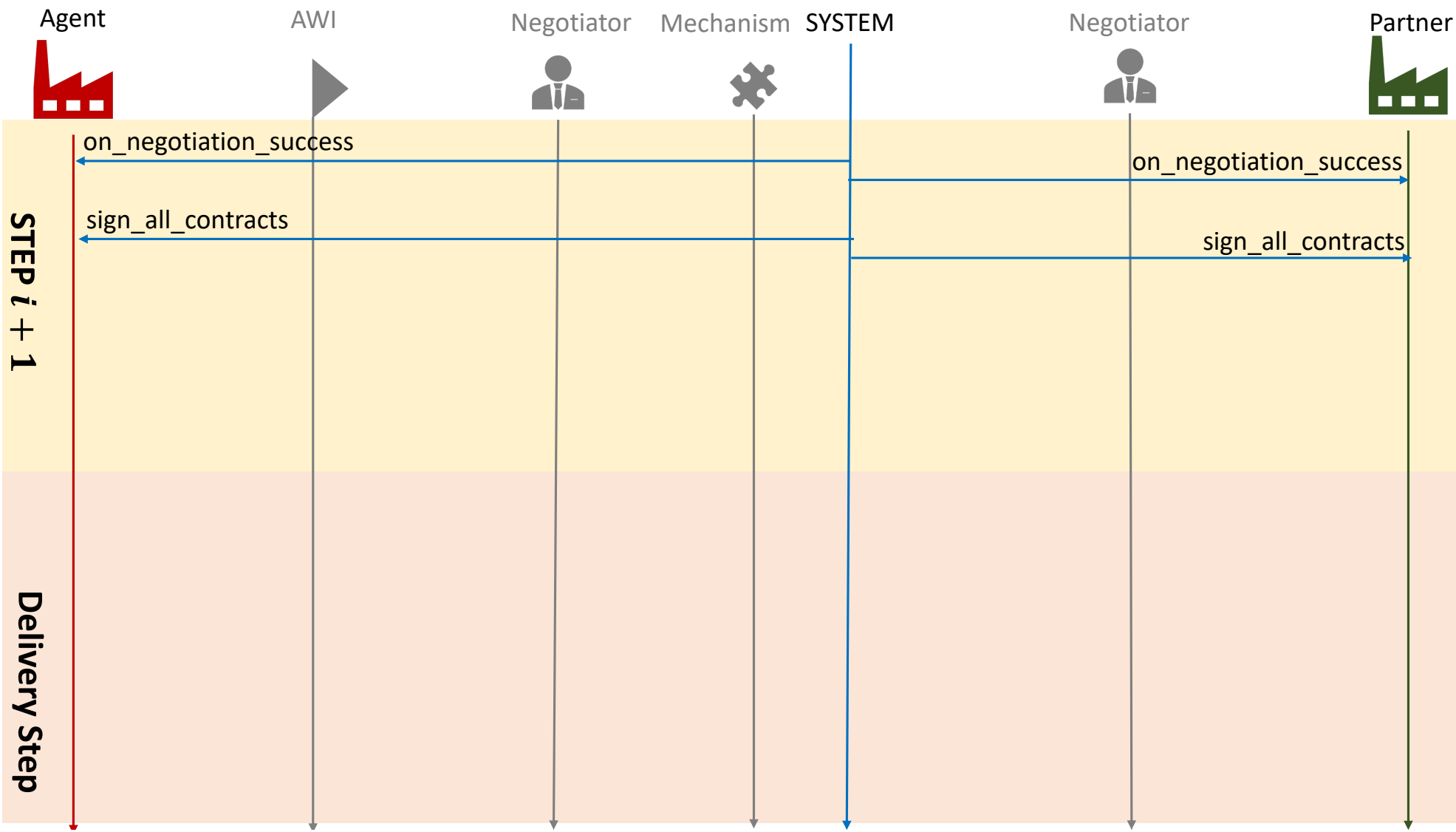
Callbacks and Timing



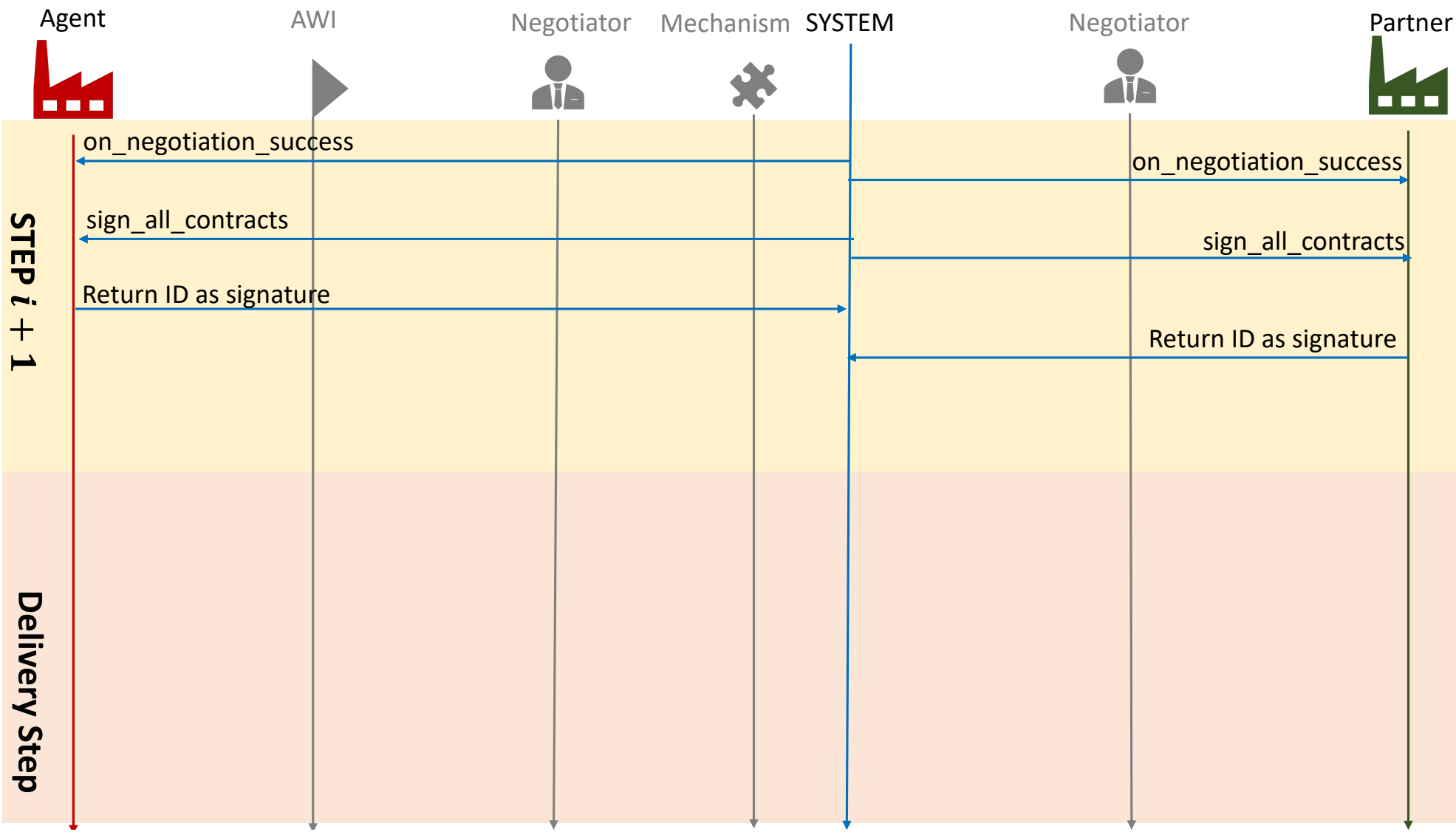
Callbacks and Timing



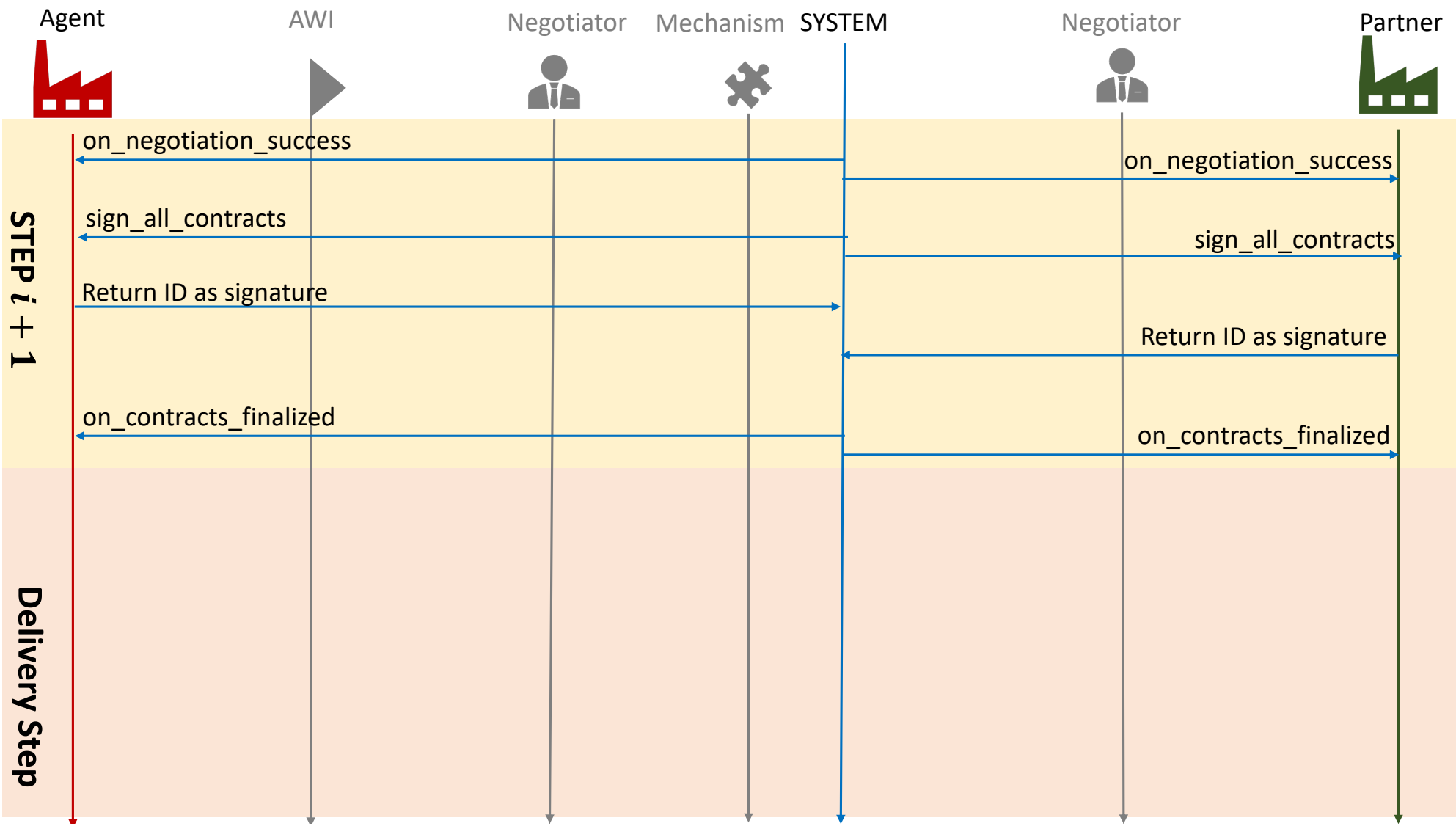
Callbacks and Timing



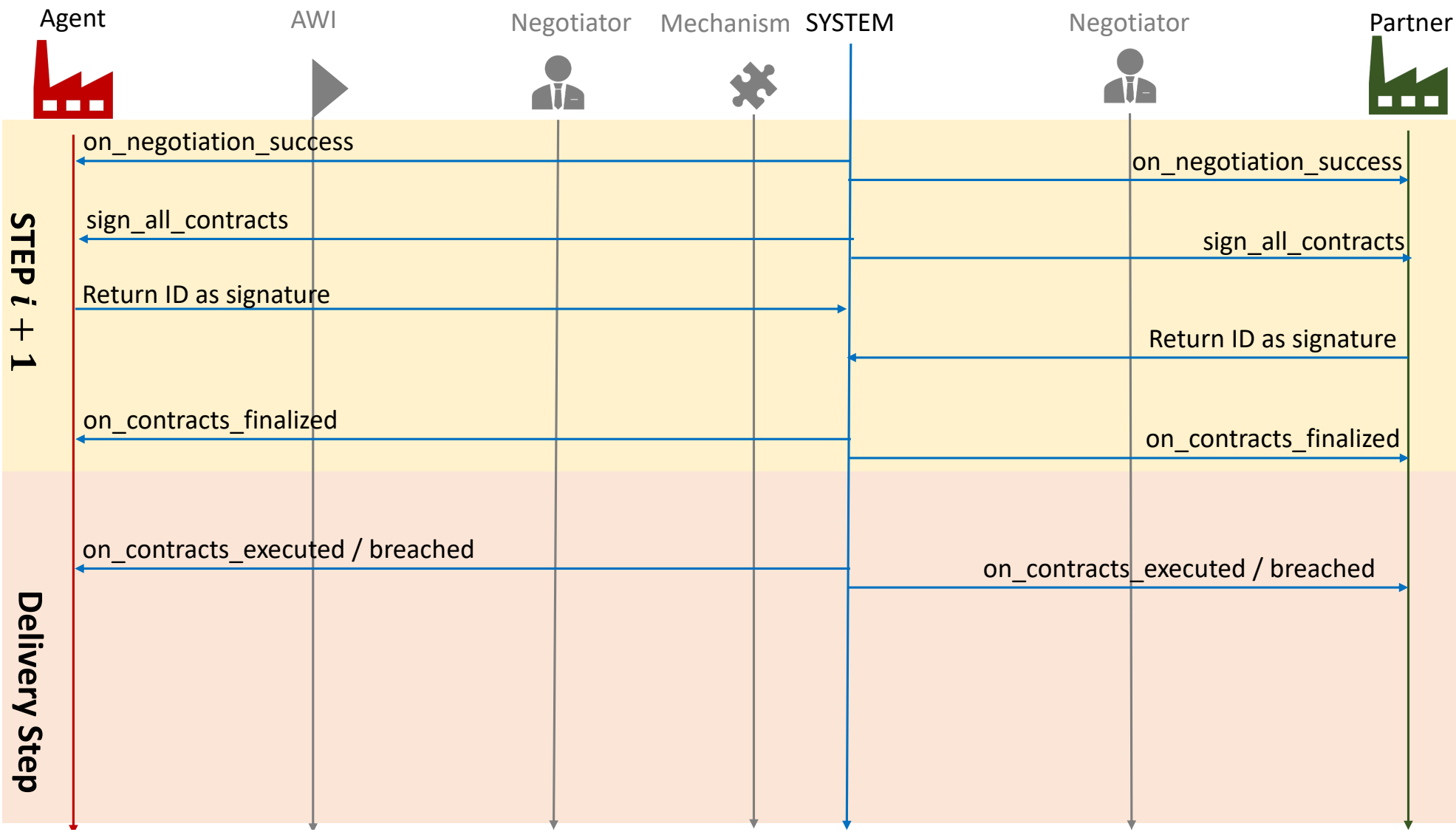
Callbacks and Timing



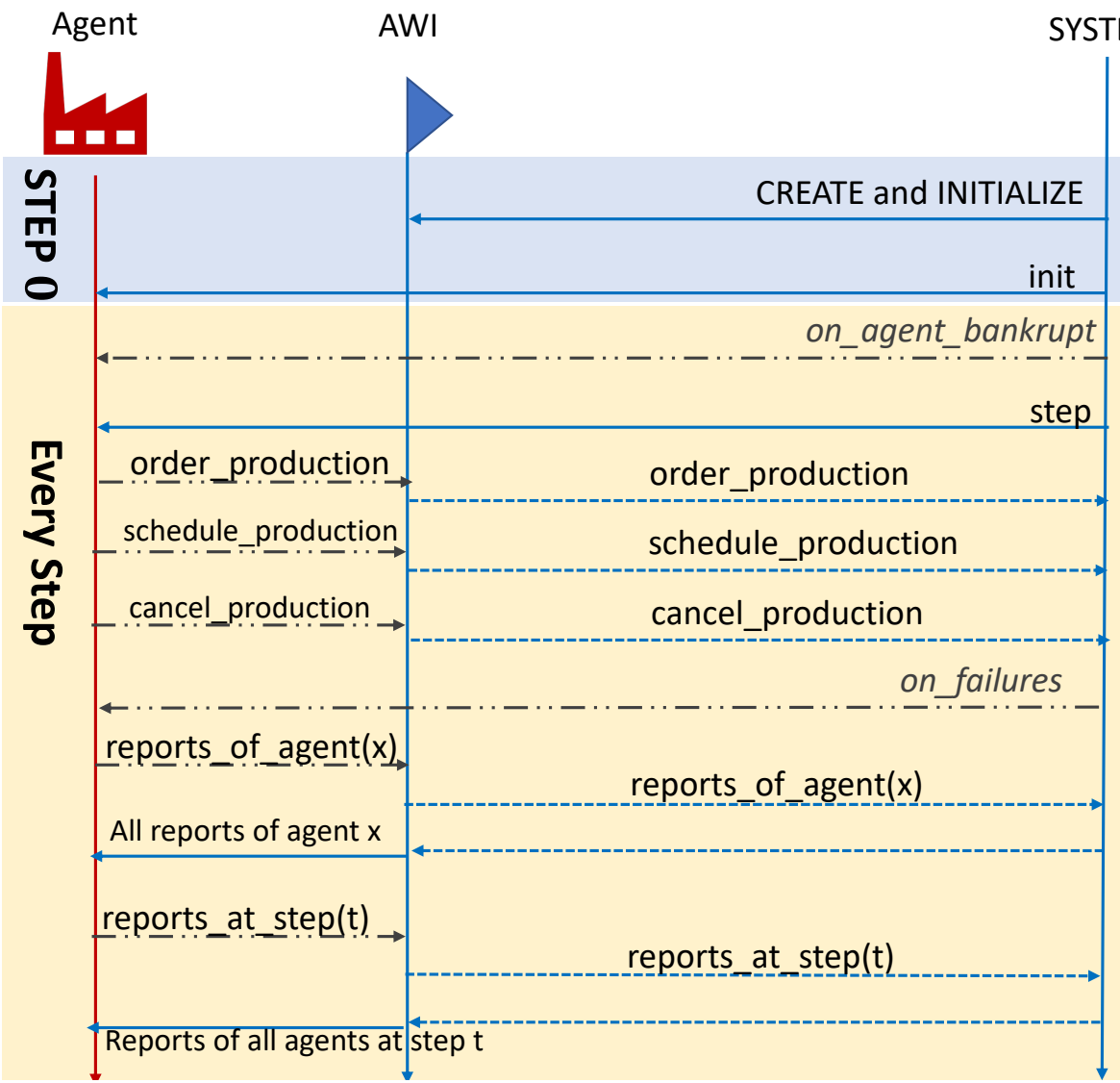
Callbacks and Timing



Callbacks and Timing



Callbacks and Timing



OTHER CALLBACKS

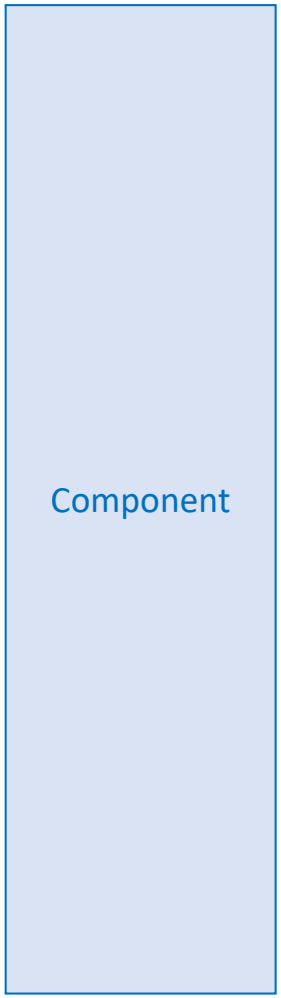
Guaranteed

Optional

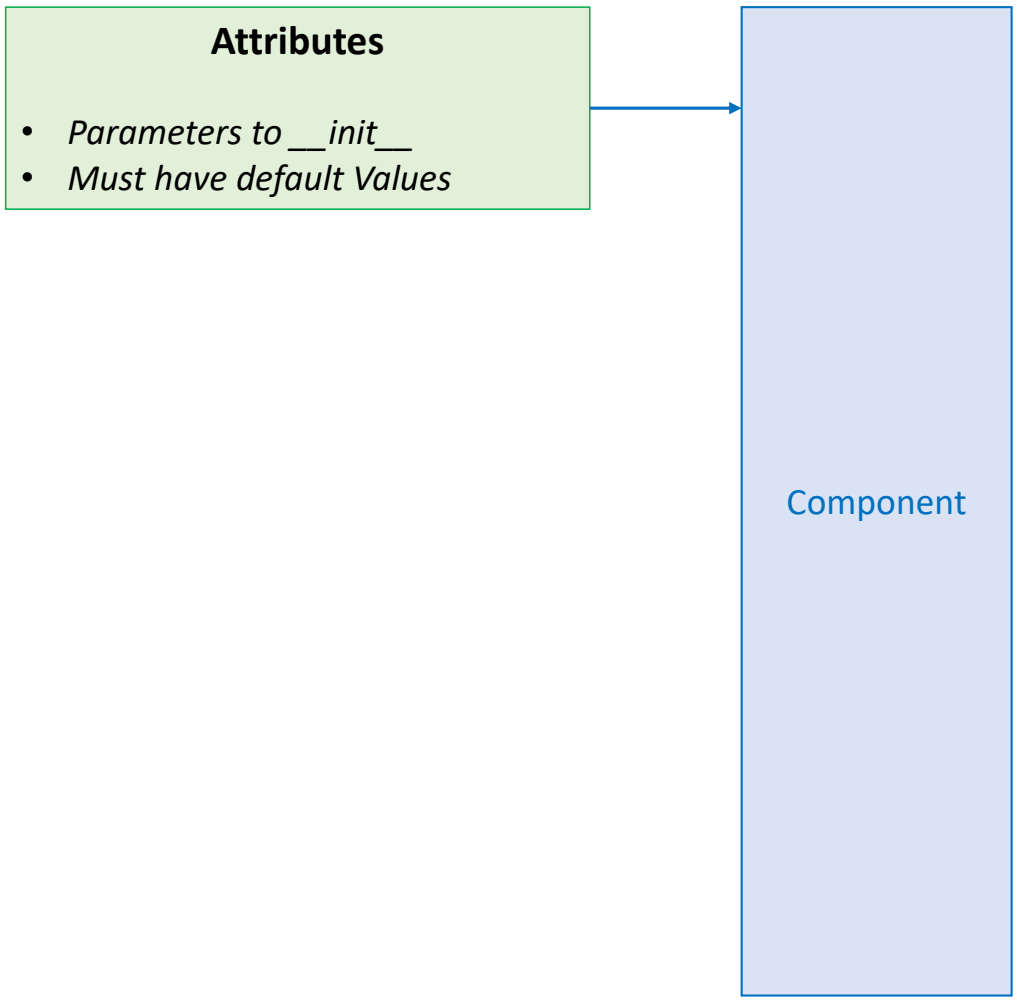
Bulletin Board Sections

breaches = Breach List
reports_agent = Financial Reports
reports_time = Financial Reports
settings = Global Settings

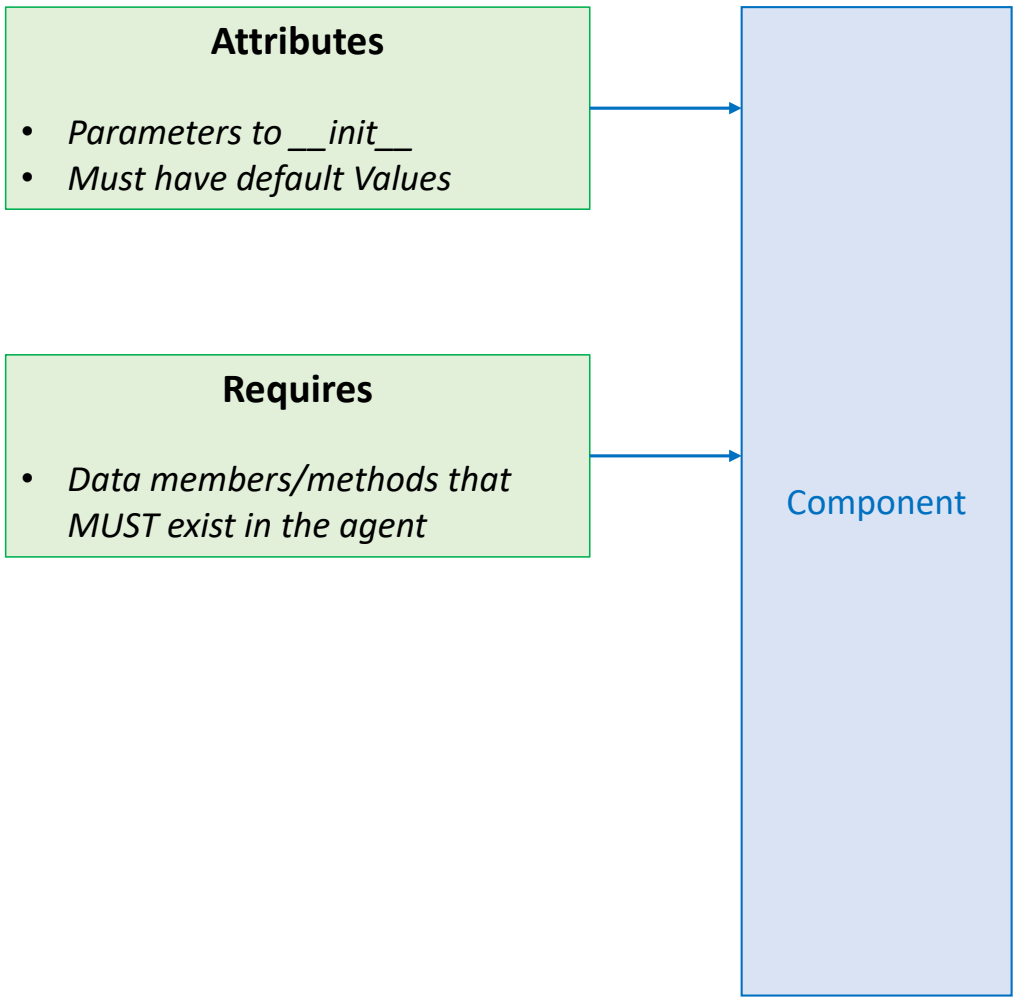
What is a component?



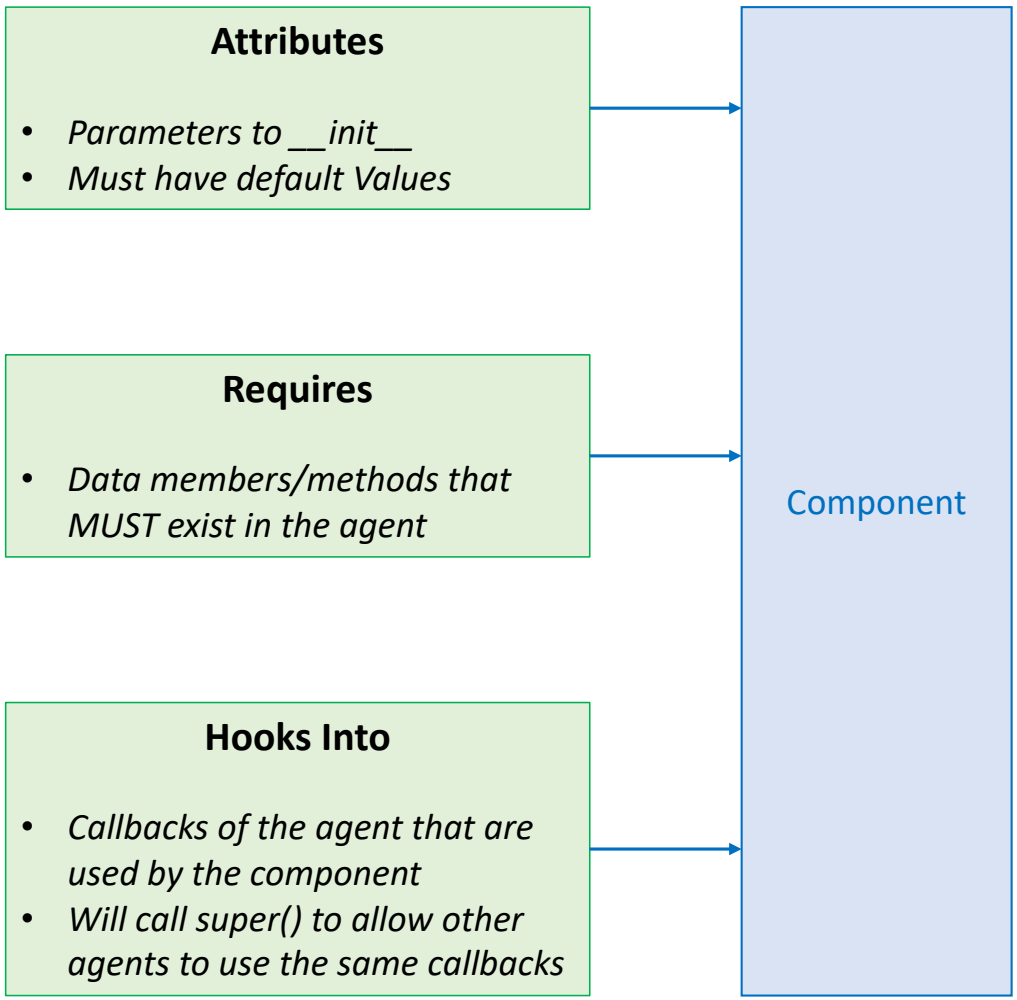
What is a component?



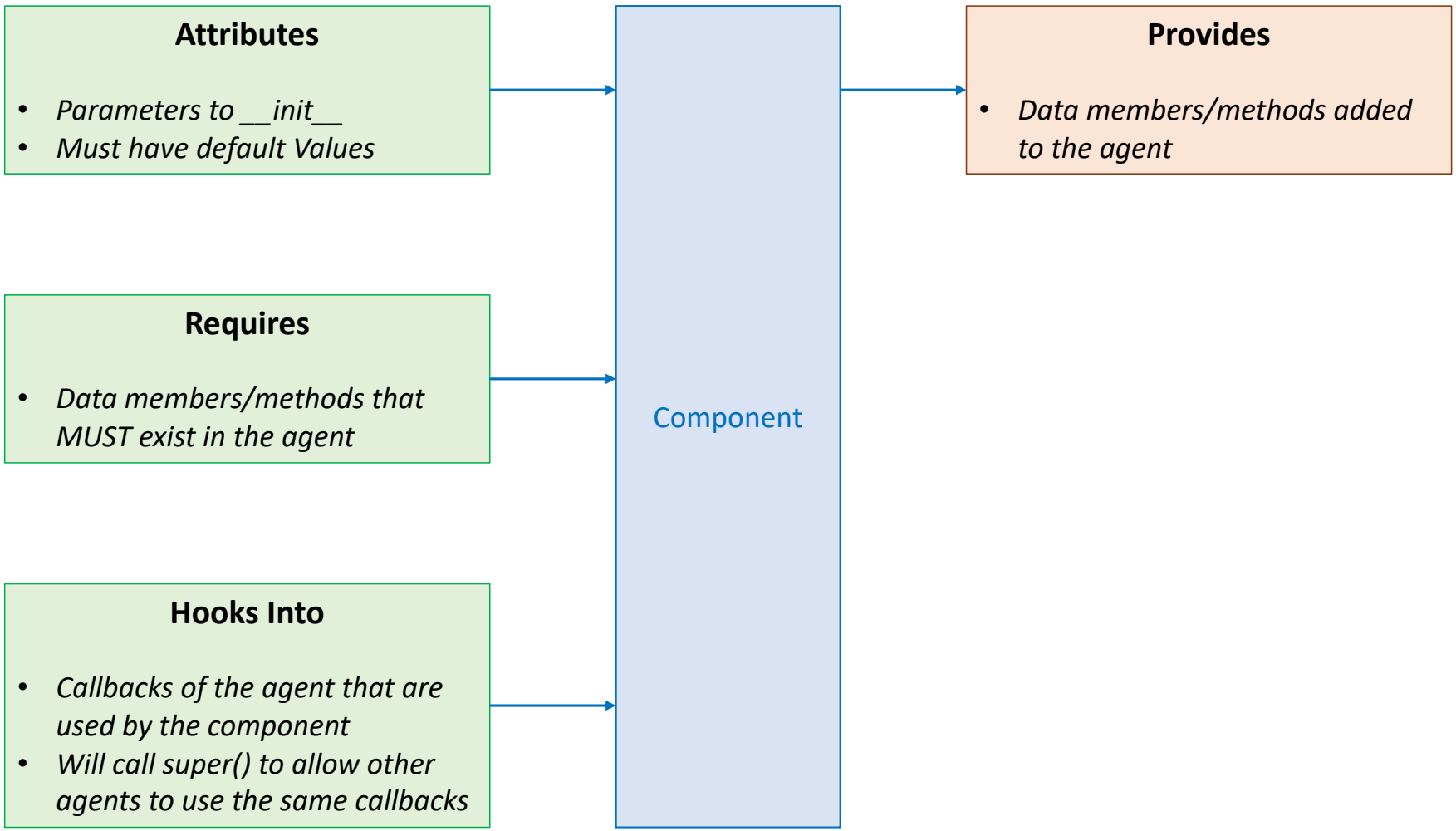
What is a component?



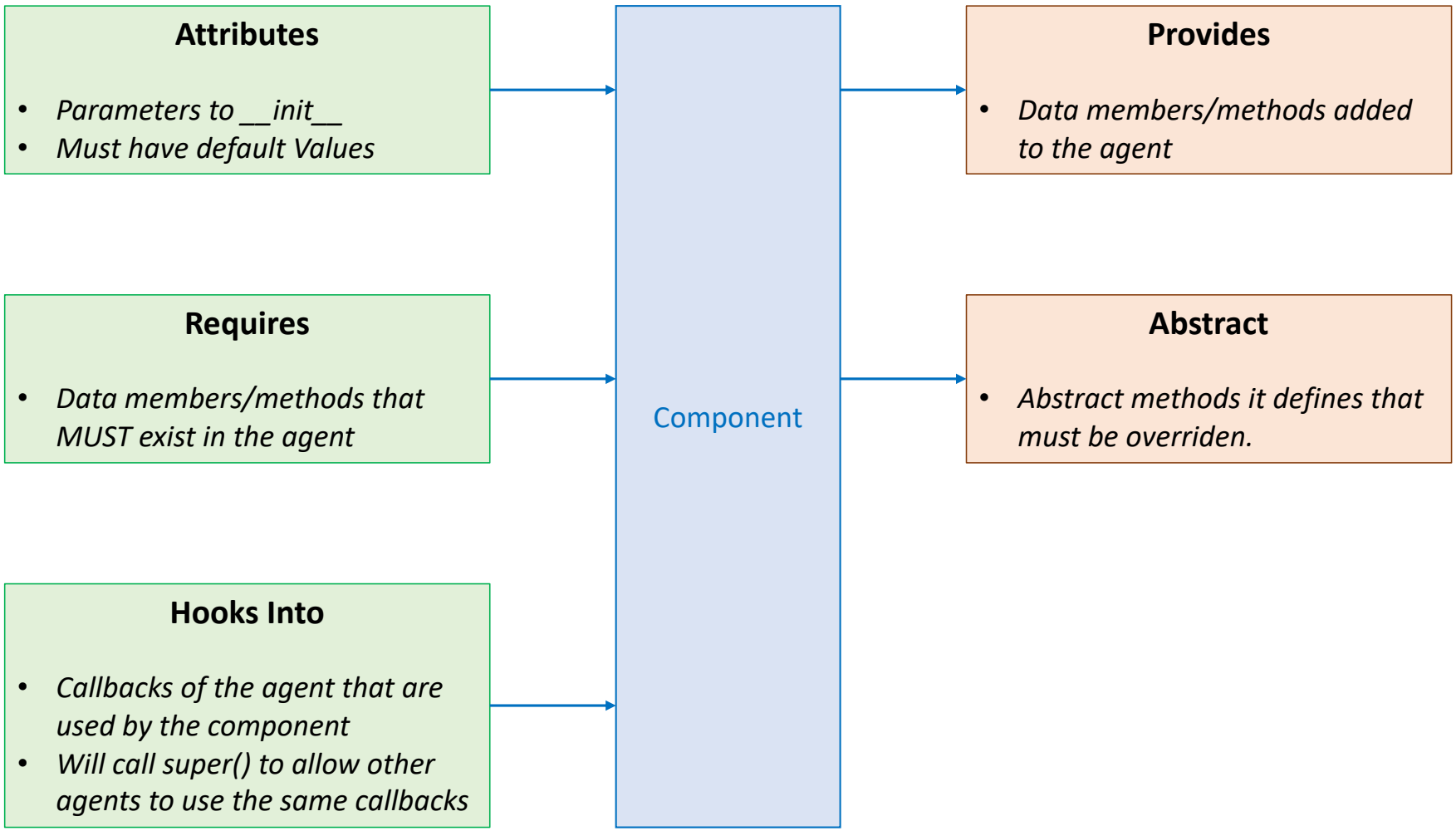
What is a component?



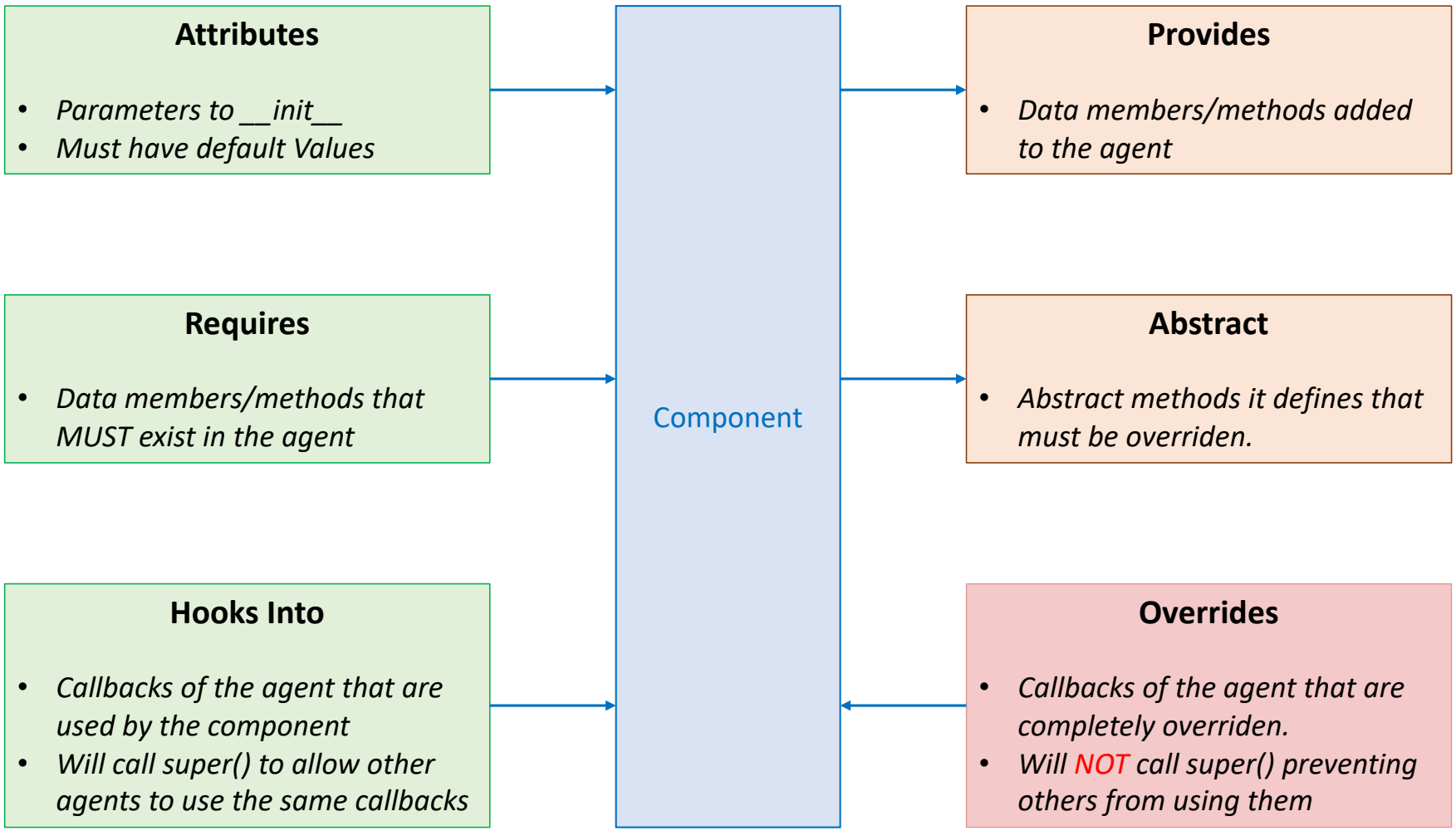
What is a component?



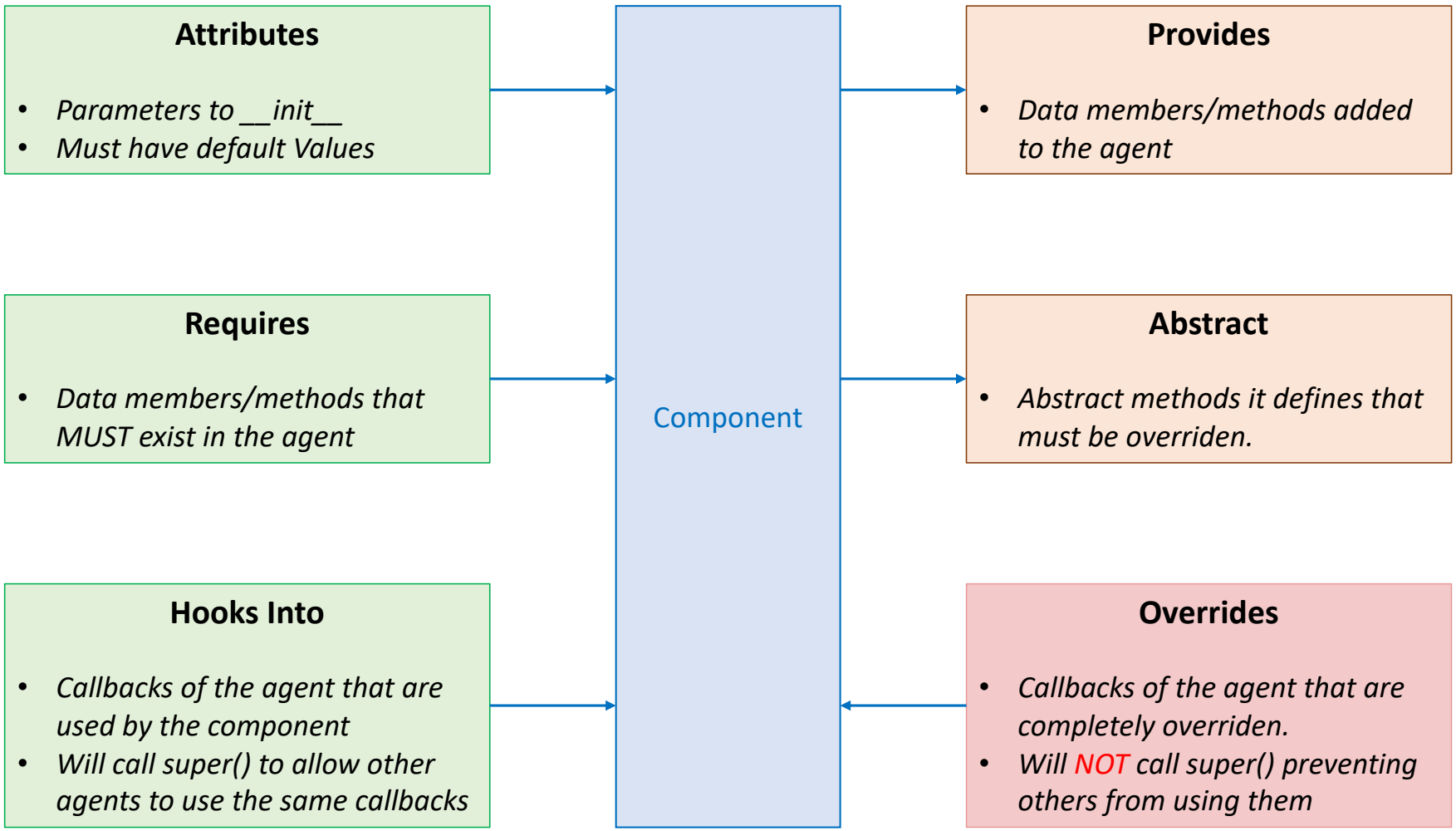
What is a component?



What is a component?

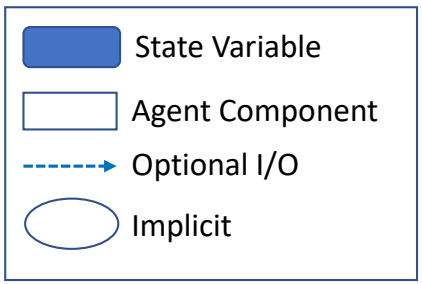


What is a component?



Hook into step, init, and on_* and override anything else

SCML Agent Components

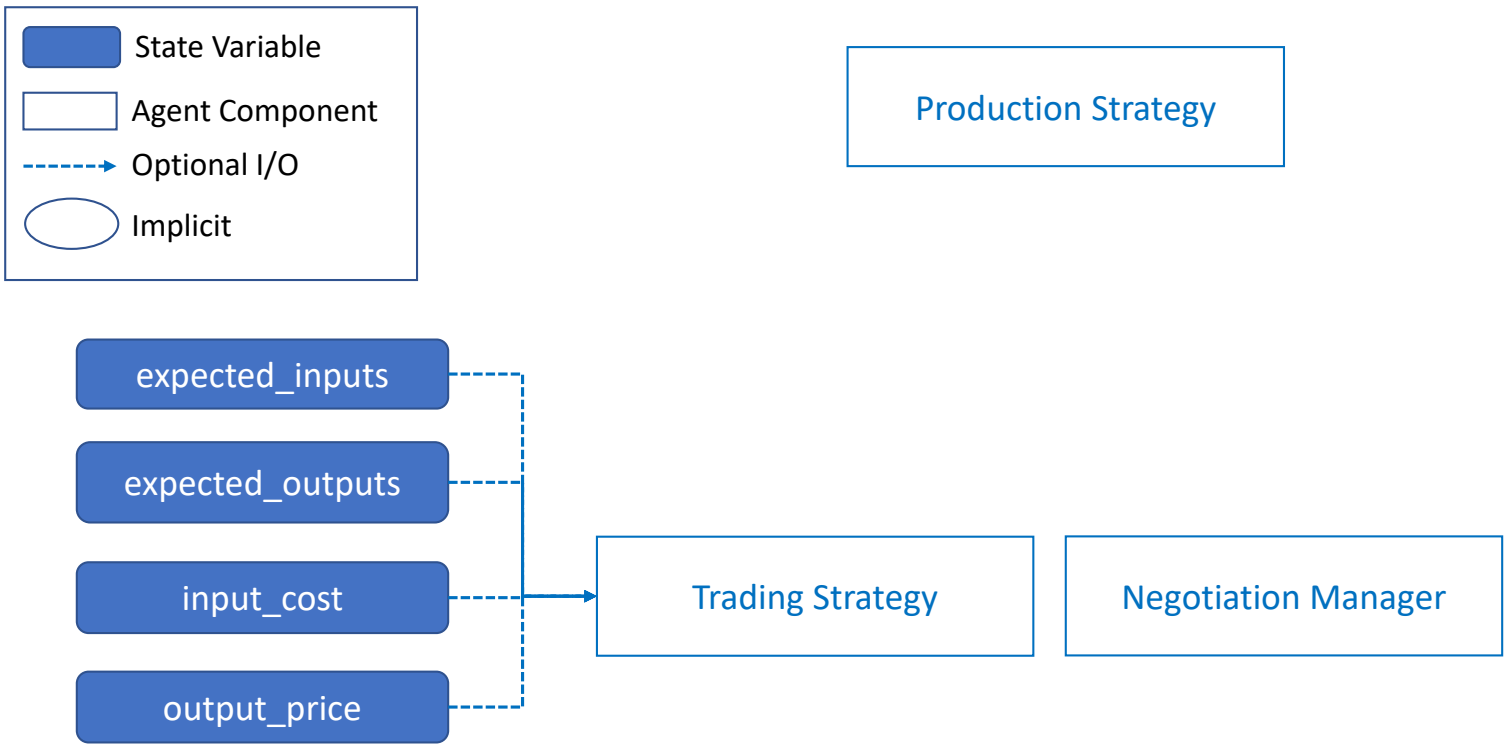


Production Strategy

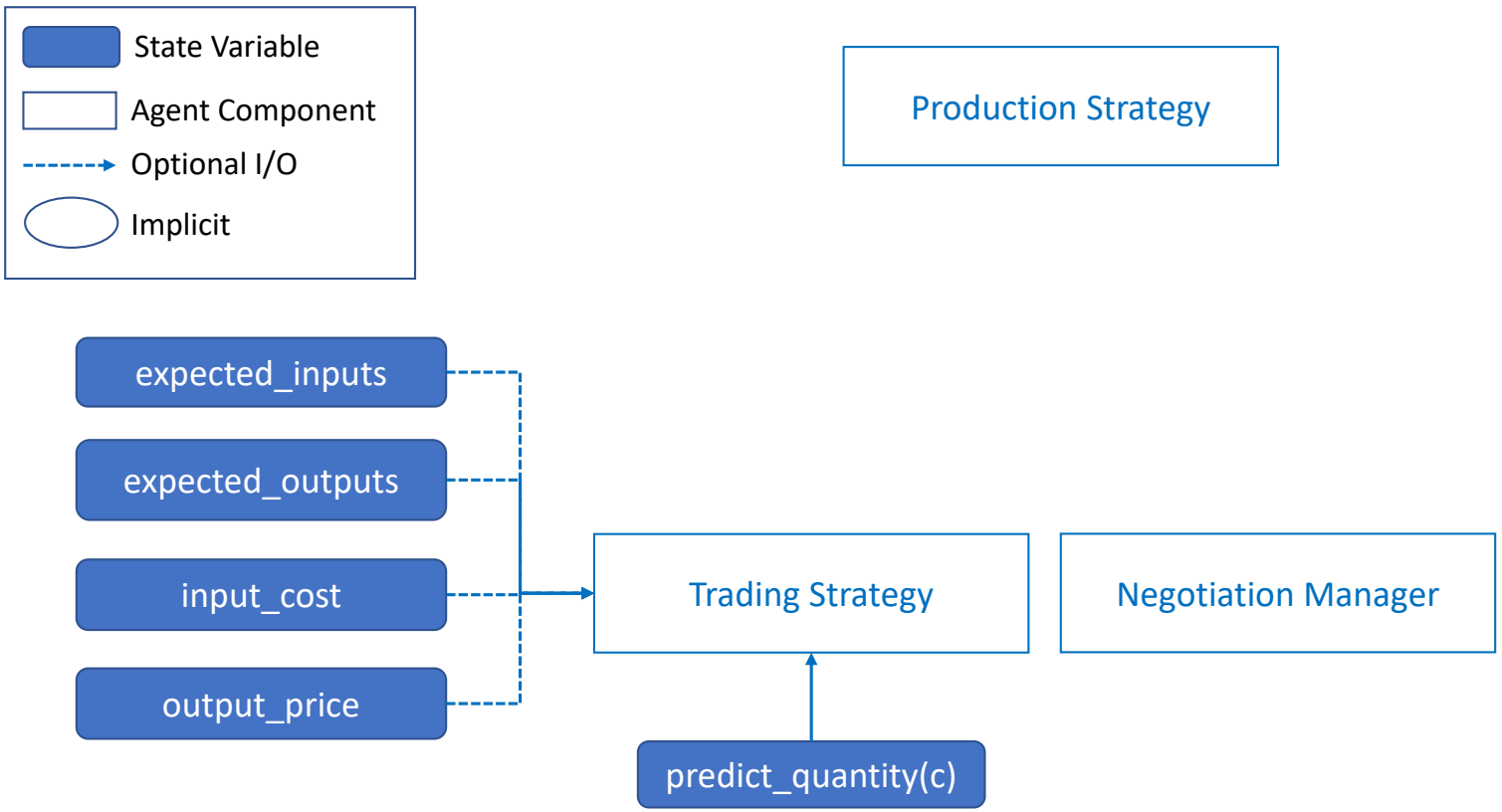
Trading Strategy

Negotiation Manager

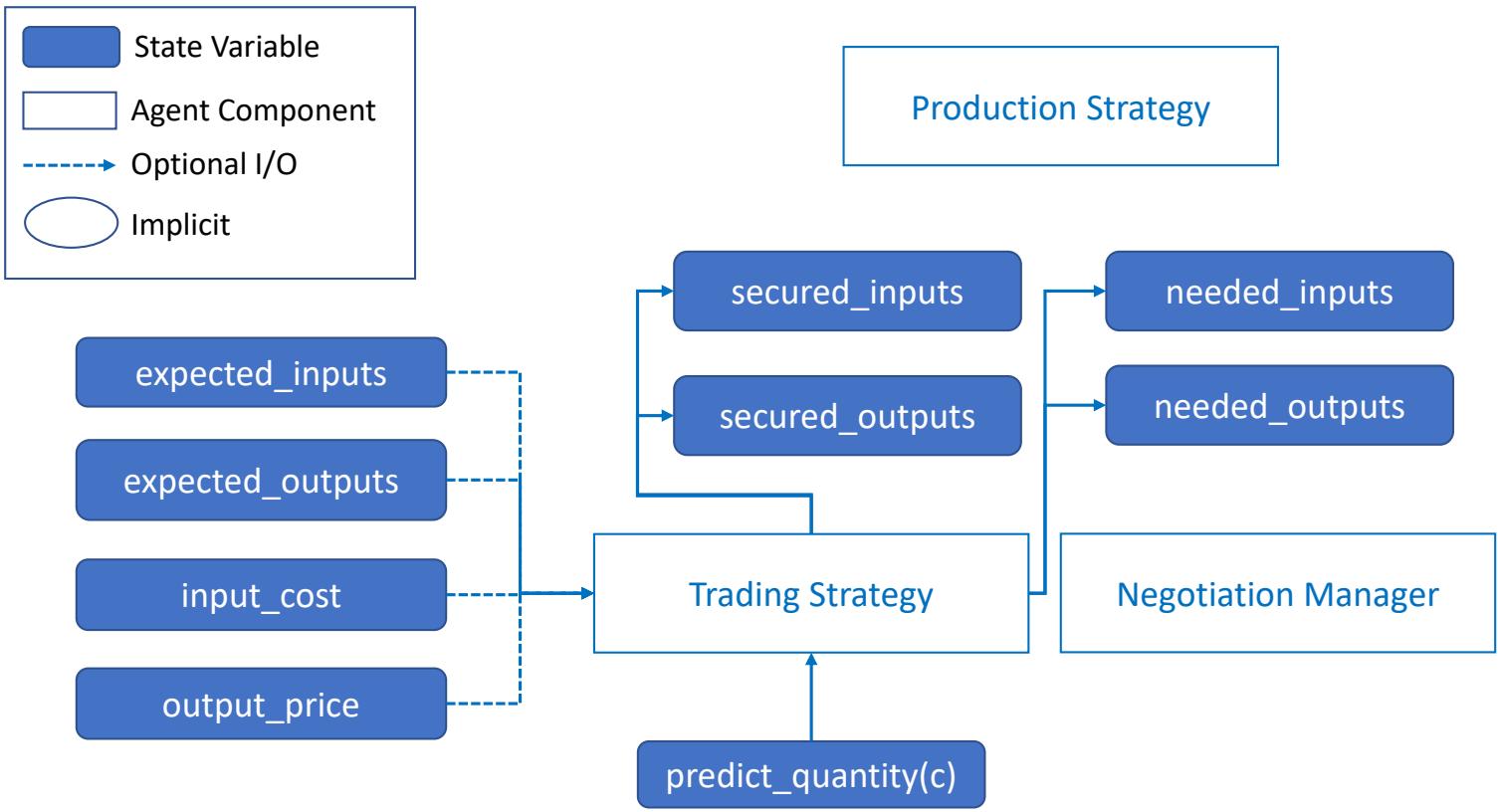
SCML Agent Components



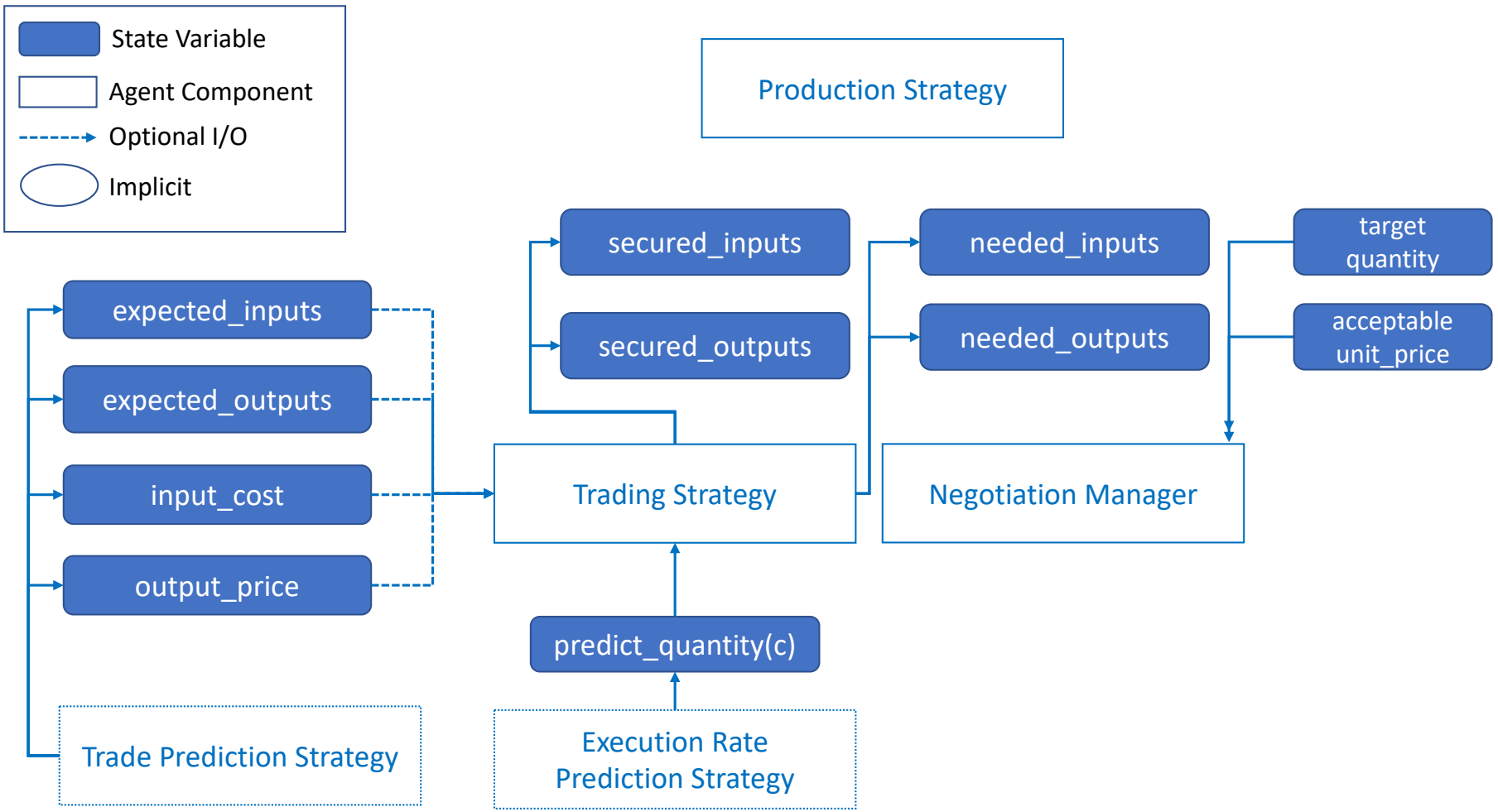
SCML Agent Components



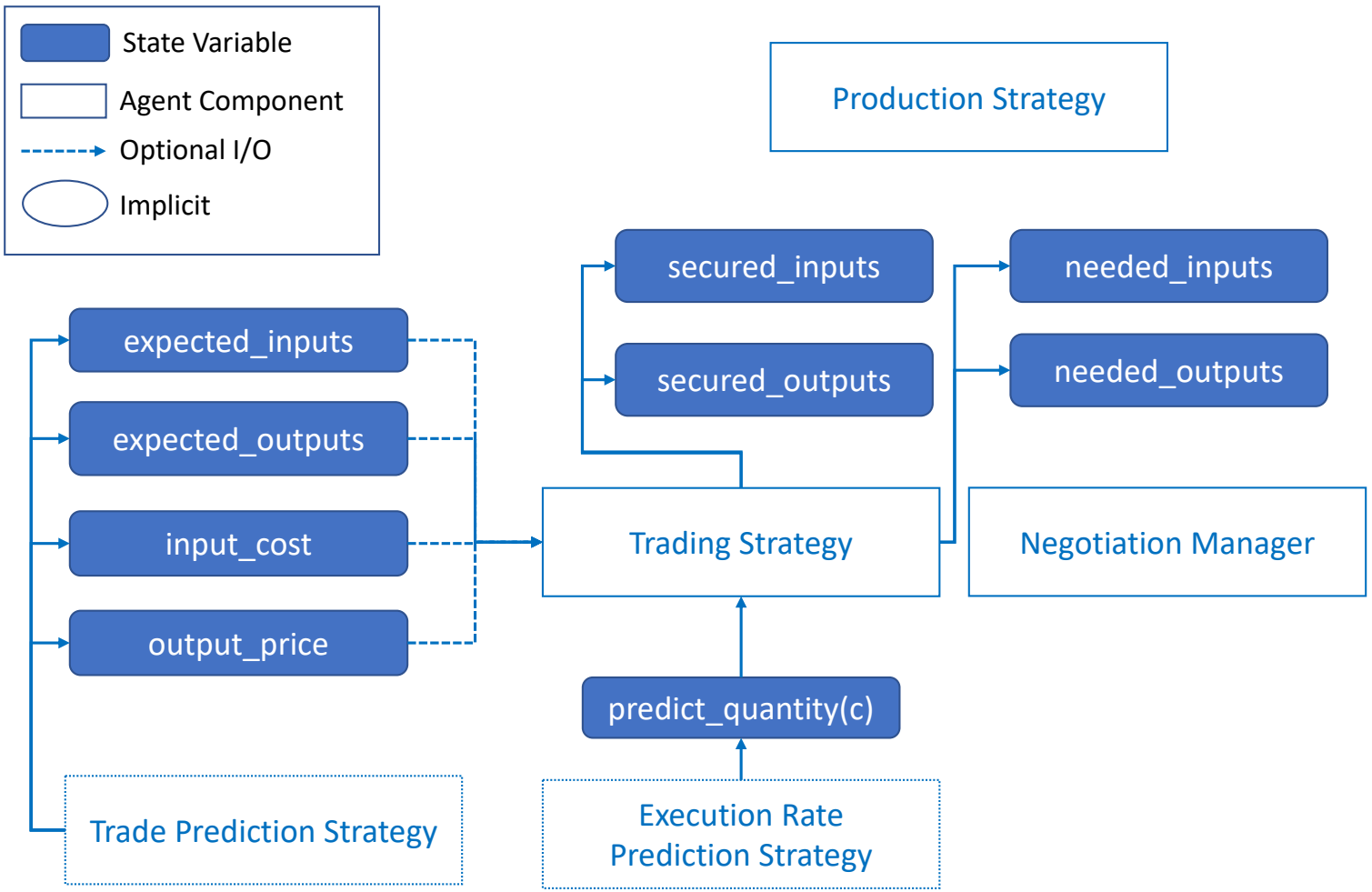
SCML Agent Components



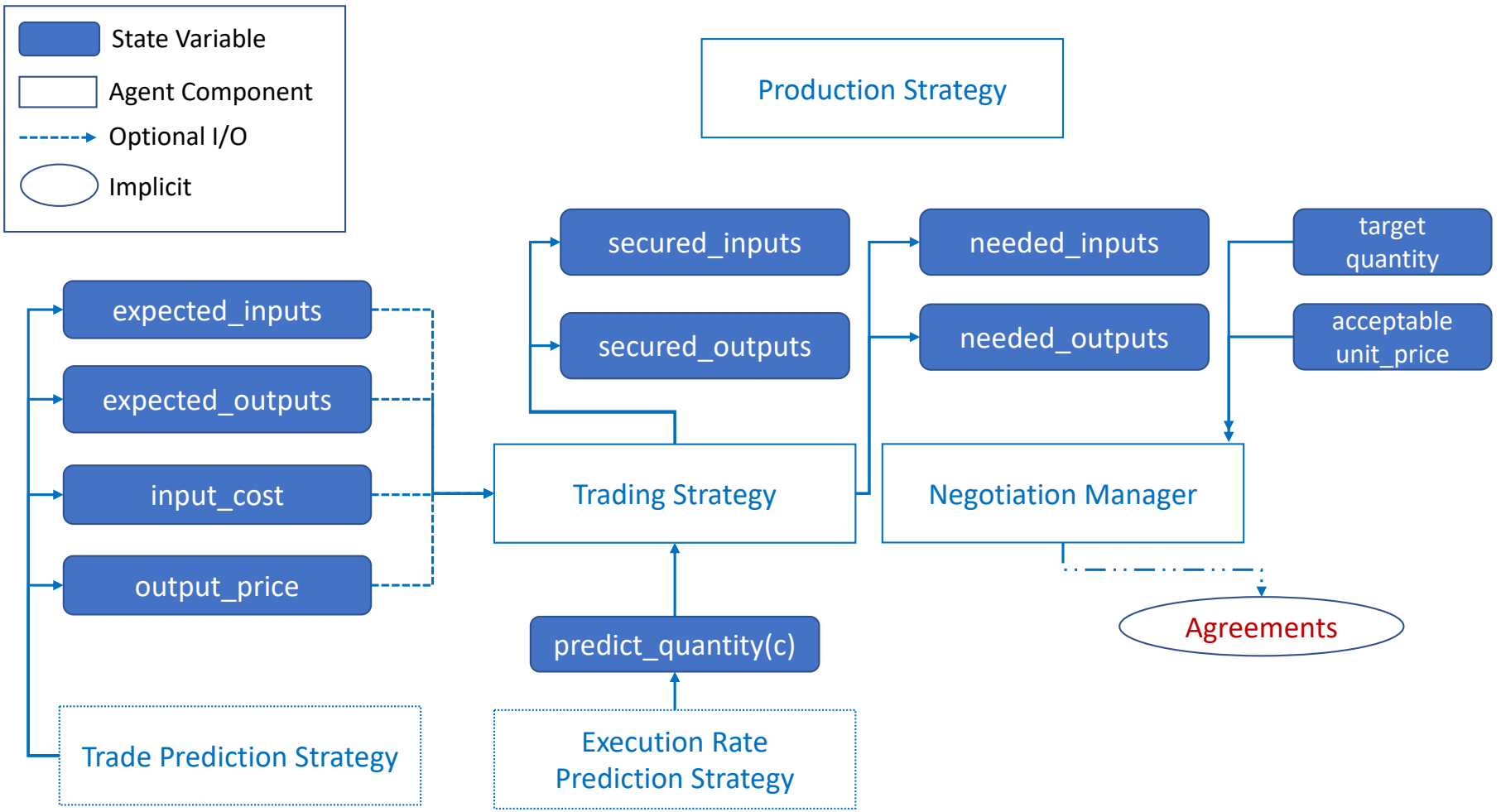
SCML Agent Components



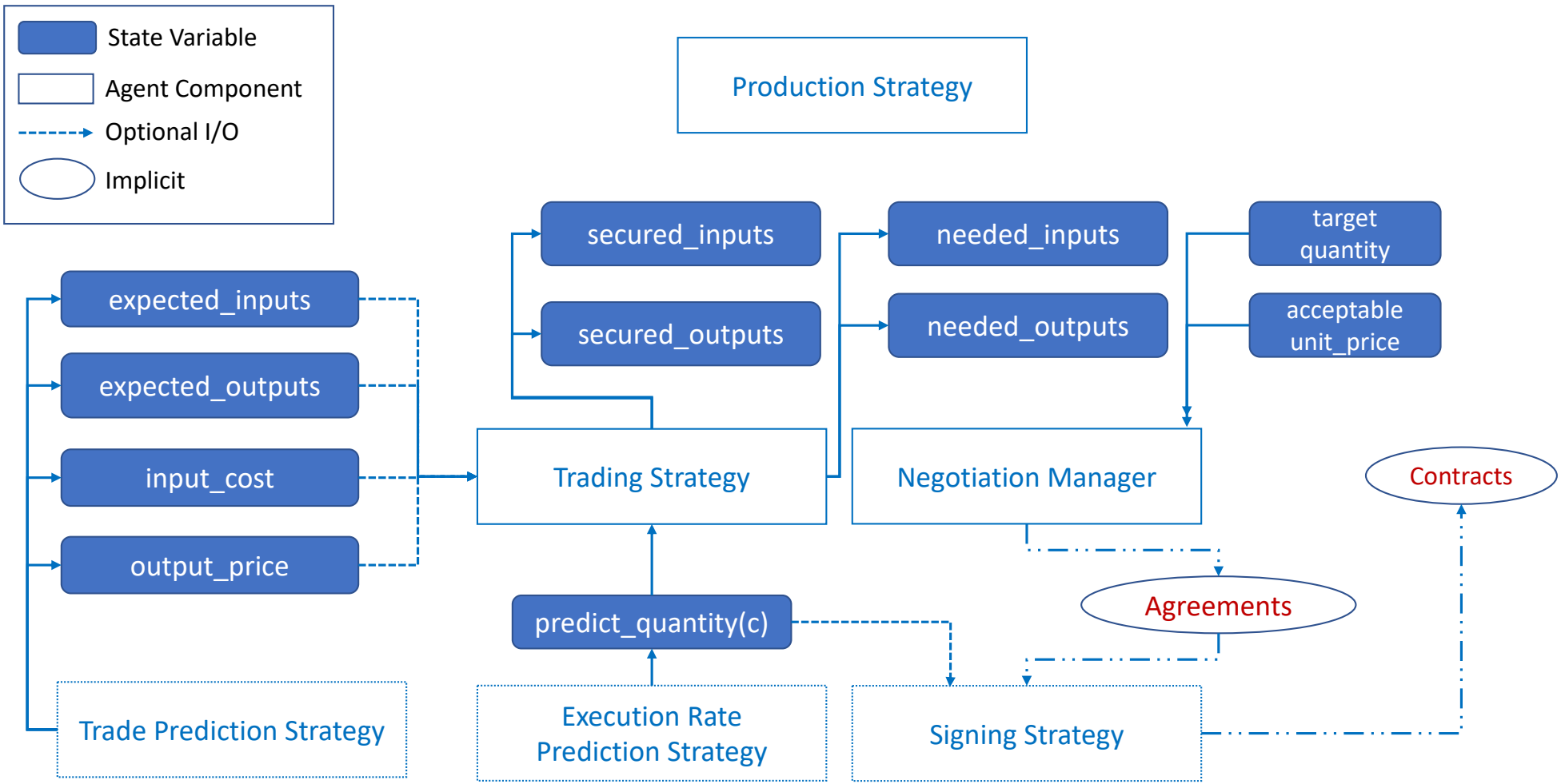
SCML Agent Components



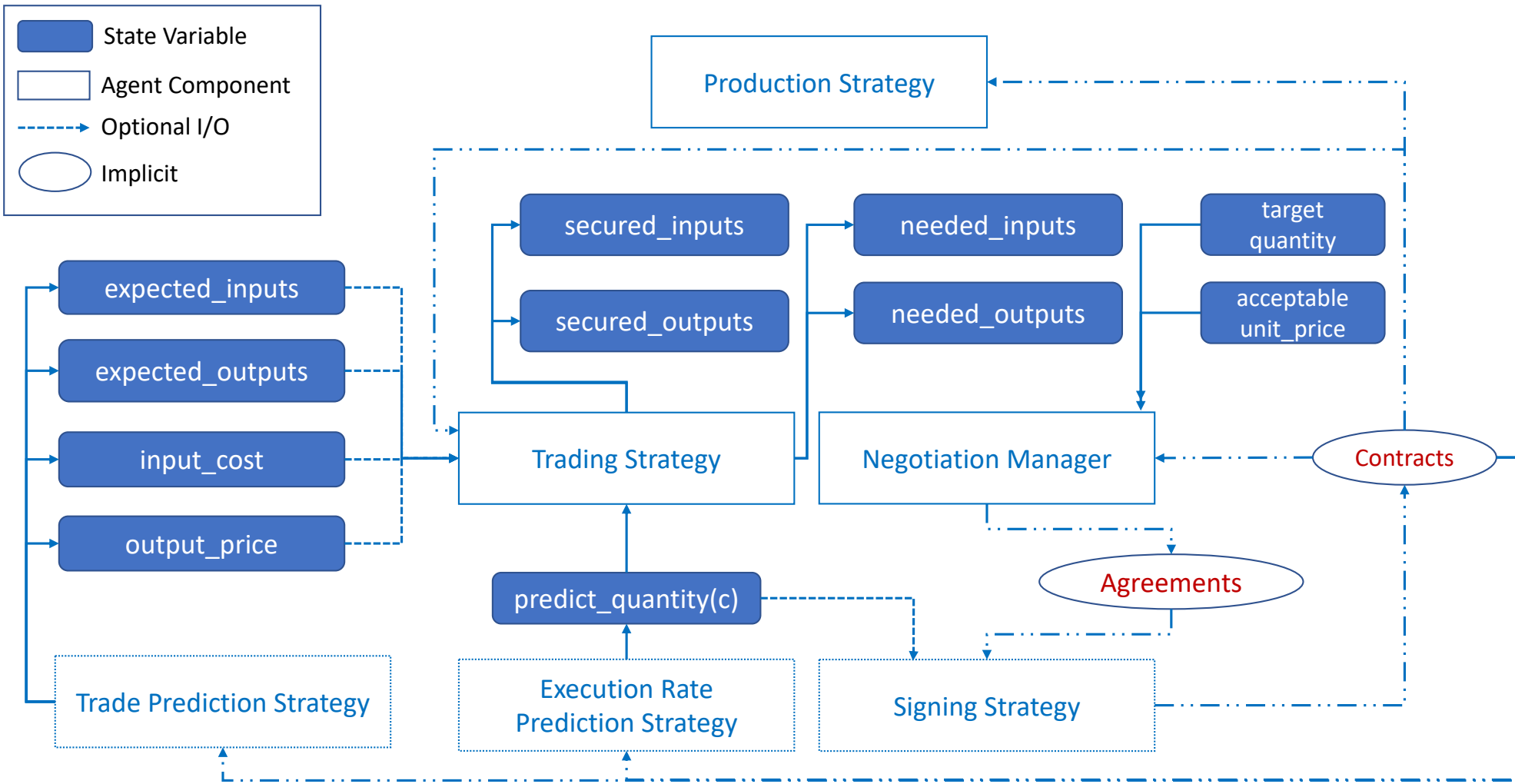
SCML Agent Components



SCML Agent Components



SCML Agent Components



Trading Strategy

NEC-AIST AI Cooperative Research Laboratory

Role

- Decides an overall **business plan**.
- Keeps track of buy/sell **needs**.

Built-in Options

No Strategy yes, just do nothing.

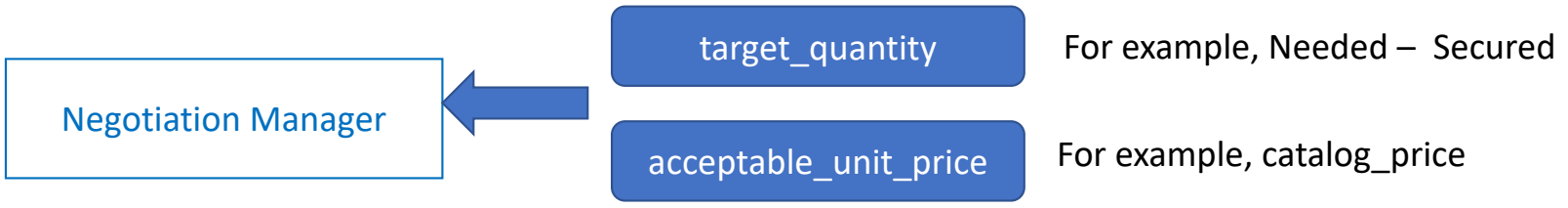
Reactive Zero needs.

Prediction Based trade and execution prediction.

- Needs come from trade predictions.
- Secured inputs/outputs come from execution prediction.

Negotiation Manager

Role



Controls negotiation.

- Sets negotiation agendas → **Proactive** .
- Accepts/rejects negotiation requests → **Reactive** .
- Defines utility functions.
- Goal: Achieve the target put by the trading strategy.

Built-in Options

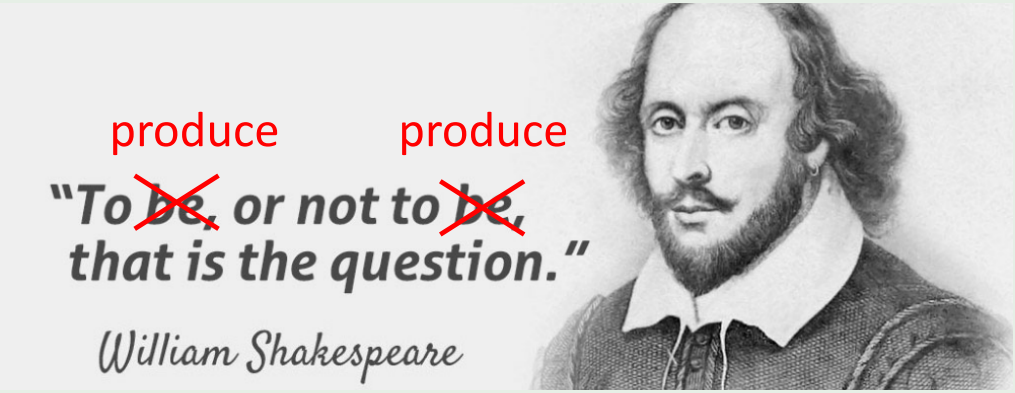
Independent Negotiations buy cheap ASAP, sell expensive ALAP.

Moving Range Creates one controller for selling and another for buying.

Step Manager Creates one controller per simulation step.

Production Strategy

Role



Built-in Options

Supply Driven Produce based on **buy** contracts.

- Inventory is always valued at the end.

Demand Driven Produce based on **sell** contracts.

- ... but inventory is valued at half the *trading* price!

Outline

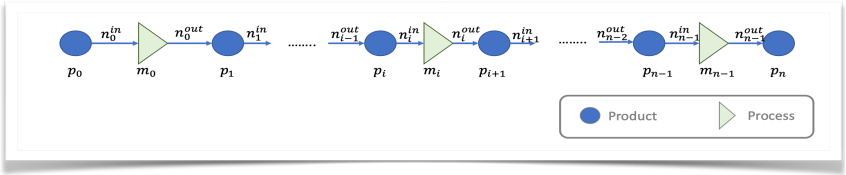
Summary

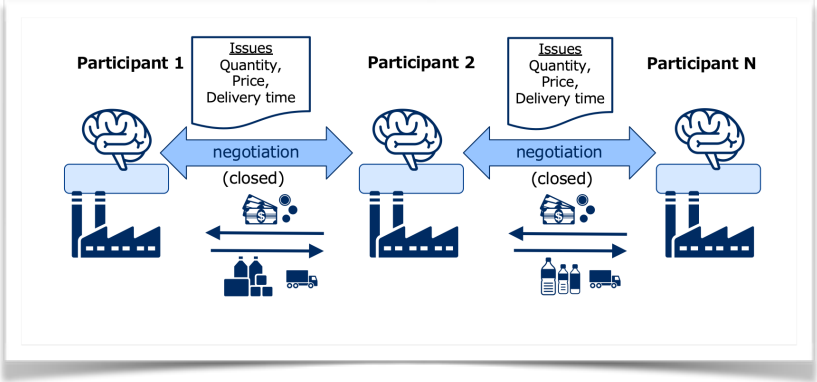
Challenge

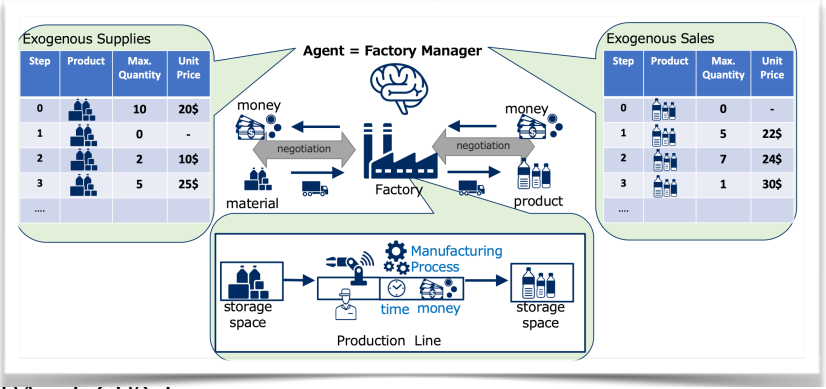
- Turn maximize profit into a ufun!!
- Dynamic interdependent ufuns.
- Sequential negotiations.
- Concurrent Negotiations.
- Negotiation under uncertainty.
- Adaptation and learning.
- Trust management.

Information

- **Website** https://scml.cs.brown.edu/
- **Code** https://www.github.com/yasserfarouk/scml
- **Youtube** https://www.youtube.com/playlist?list=PLqvs51K2Mb8IJe5Yz5jmYrRAwvIpGU2nF







References I

Aydoğan, R., Festen, D., Hindriks, K. V., and Jonker, C. M. (2017). Alternating offers protocols for multilateral negotiation. In *Modern Approaches to Agent-based Complex Automated Negotiation*, pages 153–167. Springer.

Mohammad, Y., Viqueira, E. A., Ayerza, N. A., Greenwald, A., Nakadai, S., and Morinaga, S. (2019). Supply chain management world. In *International Conference on Principles and Practice of Multi-Agent Systems*, pages 153–169. Springer.

Rubinstein, A. (1982). Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, pages 97–109.