

Transforming Enterprises

*meetup*



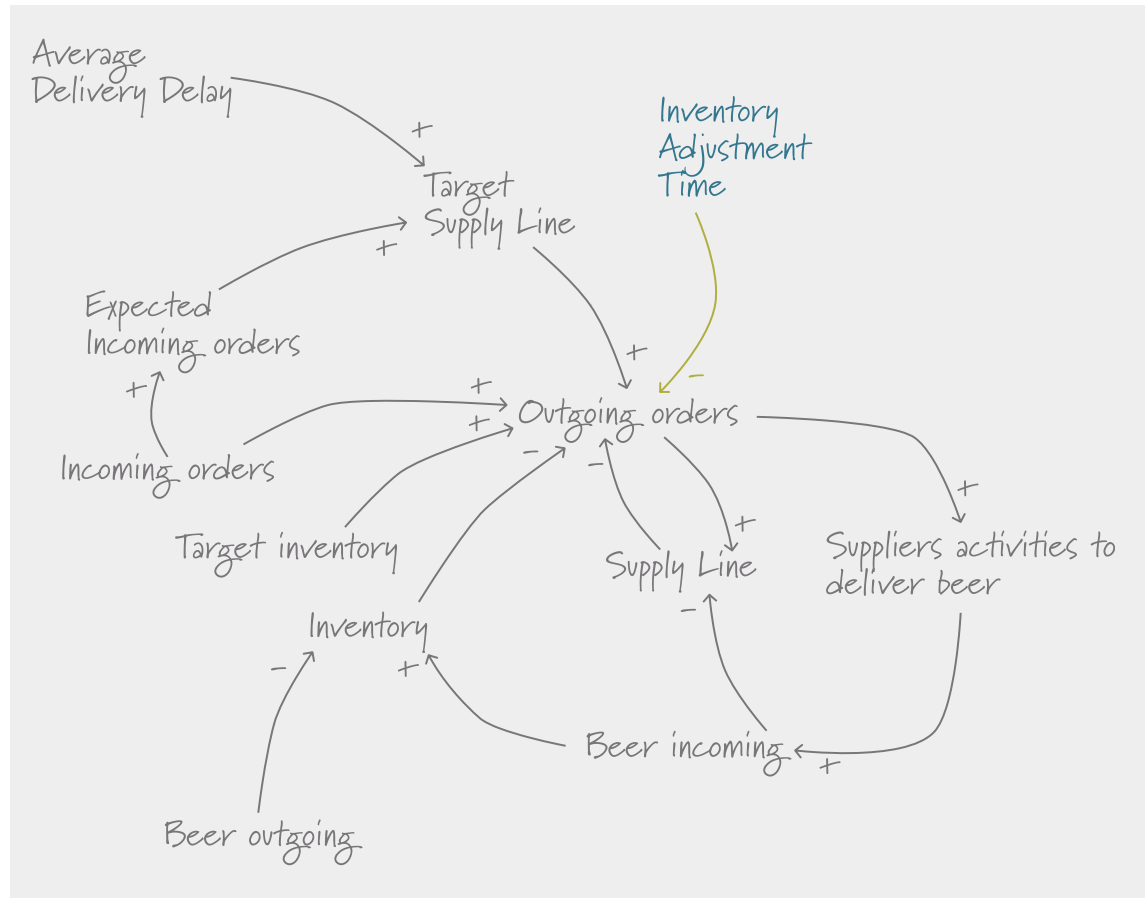
# Using Simulations And AI to Optimize Supply Chains

Illustrated using the Beer Distribution Game

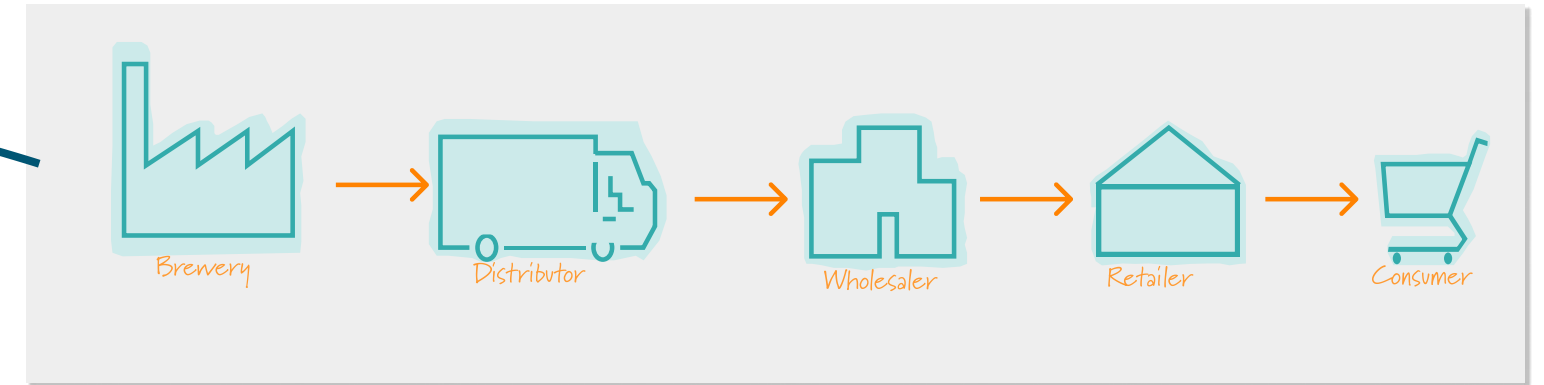
Berlin, 28.5.2020

# Topics Today

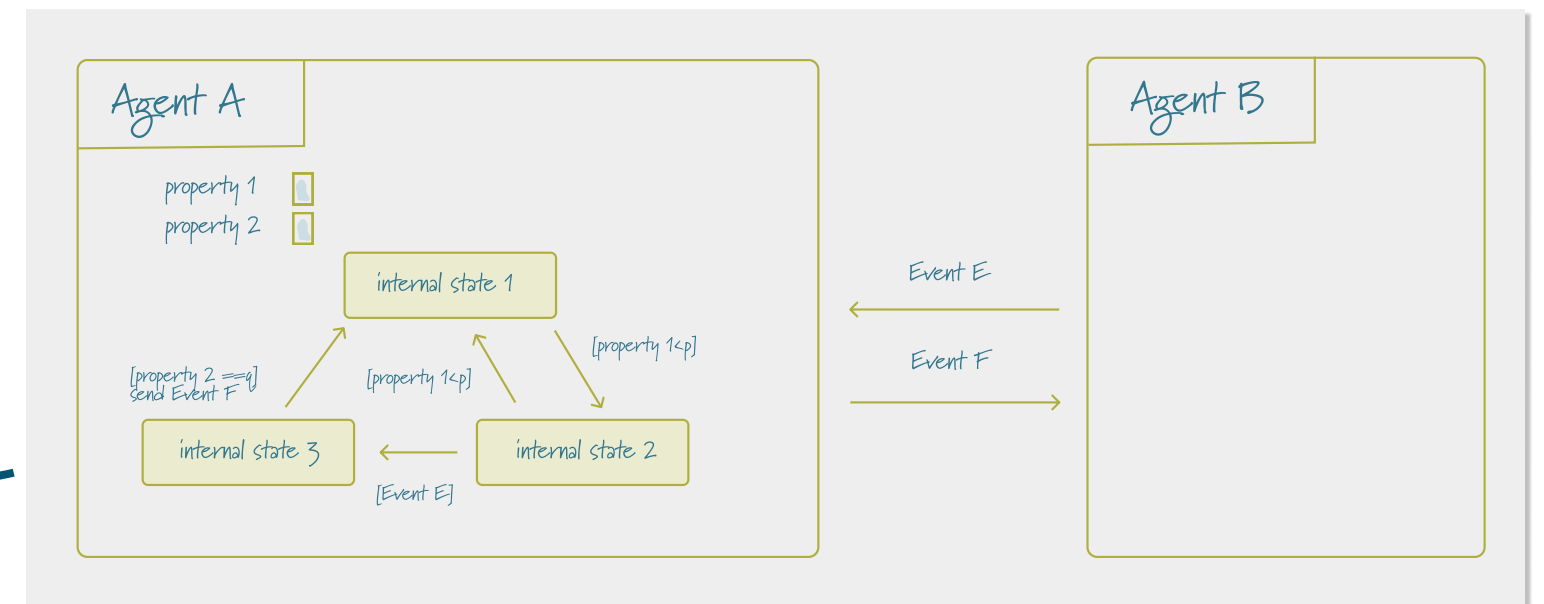
Introduction to the Beer Distribution Game, a supply chain simulation.



Train autonomous agents to play the game using a machine learning/reinforcement learning approach.



Use computational modeling to understand supply chain dynamics and find better playing strategies.





# Download The Code And Computational Notebooks

Download all the code and in-depth computational notebooks from GitHub.

The screenshot shows a JupyterLab environment. On the left is a file explorer with a tree view of the project structure, including folders like 'data', 'export', 'images', 'scenarios', 'simulation\_models', 'src', 'test', and 'venv', and files like 'beergame\_abm.ipynb', 'beergame\_export.ipynb', 'beergame\_ql.ipynb', 'beergame\_screenshot...', 'bptk\_py.log', 'readme.md', 'requirements.txt', 'training\_ai\_beergame.i...', and 'understanding\_the\_be...'. The main area displays a notebook titled 'understanding\_the\_beergame'. The notebook content includes:

The objective of the game is to ensure that the consumers demand for beer can be met directly or at least with as small a delay as possible, while keeping each players inventory as small as possible.

The sketch below illustrates the supply chain and the flow of information along it:

```
graph LR; Brewery -- "6. Demand for beer is fulfilled as soon as there is enough beer in stock." --> Distributor; Distributor -- "7. Demand for beer is fulfilled as soon as there is enough beer in stock." --> Wholesaler; Wholesaler -- "8. Demand for beer is fulfilled as soon as there is enough beer in stock." --> Retailer; Retailer -- "9. Demand for beer is fulfilled as soon as there is enough beer in stock." --> Consumer; Consumer -- "1. The consumer buys beer from the retailer" --> Retailer; Retailer -- "2. The retailer either has beer in stock or orders extra units of beer from the wholesaler." --> Wholesaler; Wholesaler -- "3. The wholesaler either has beer in stock or orders extra units of beer from the distributor." --> Distributor; Distributor -- "4. The distributor either has beer in stock or orders extra units of beer from the brewery." --> Brewery; Brewery -- "5. The brewery either has beer in stock or it increases its production." --> Distributor
```

Initially customer demand for beer is stable at 100 units per week and the entire supply chain is in a steady state. Each member of the supply chain has an inventory of 400 units.

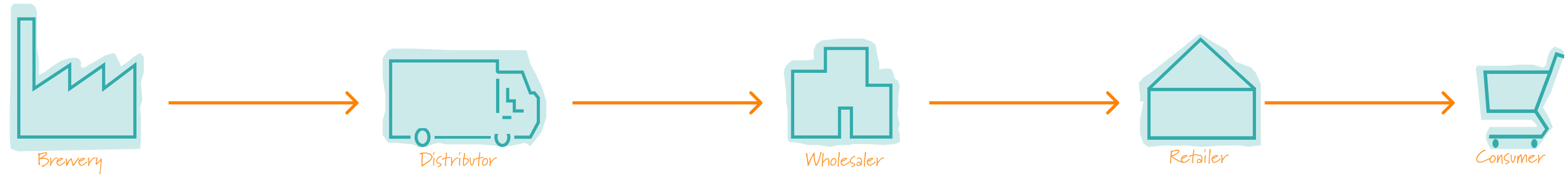
The rules of the game are simple – in every round each player performs the following four steps:

- **Check deliveries.** Check how many units of beer are being delivered to him from his supplier in the supply chain

All resources are available via <https://www.transentis.com>

# The Beer Distribution Game

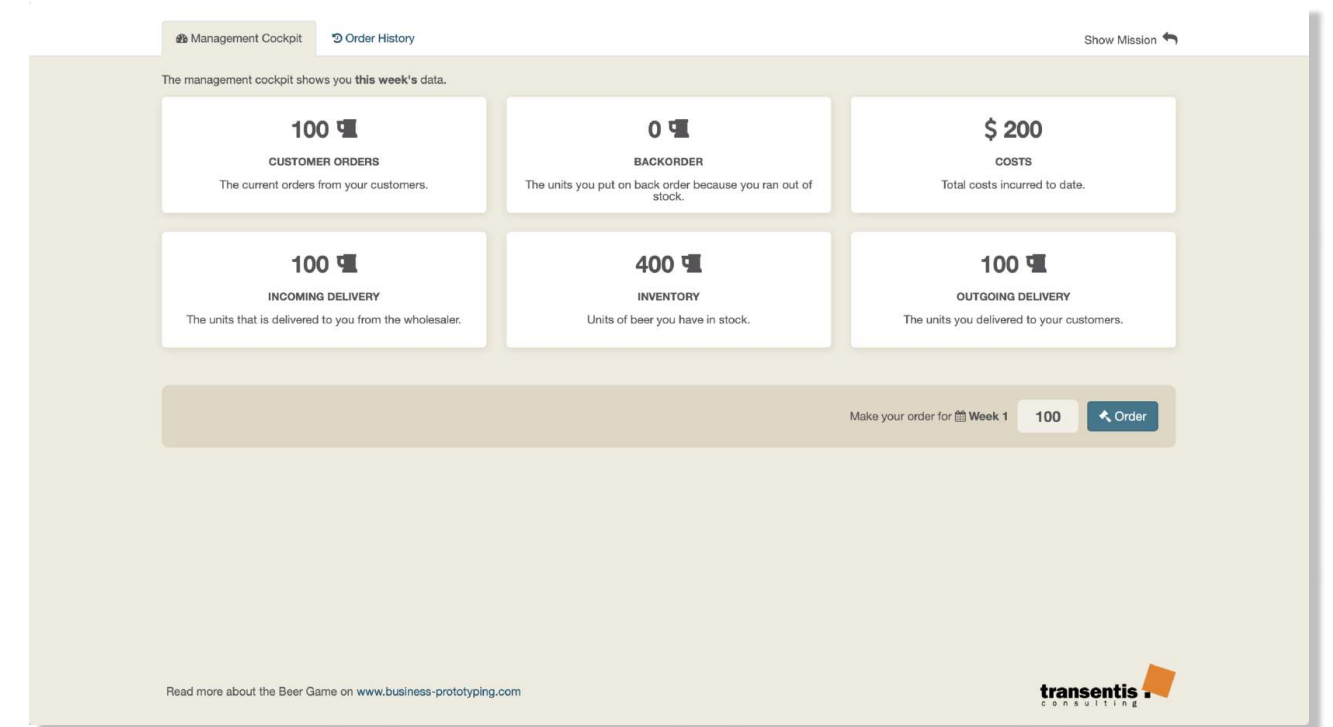
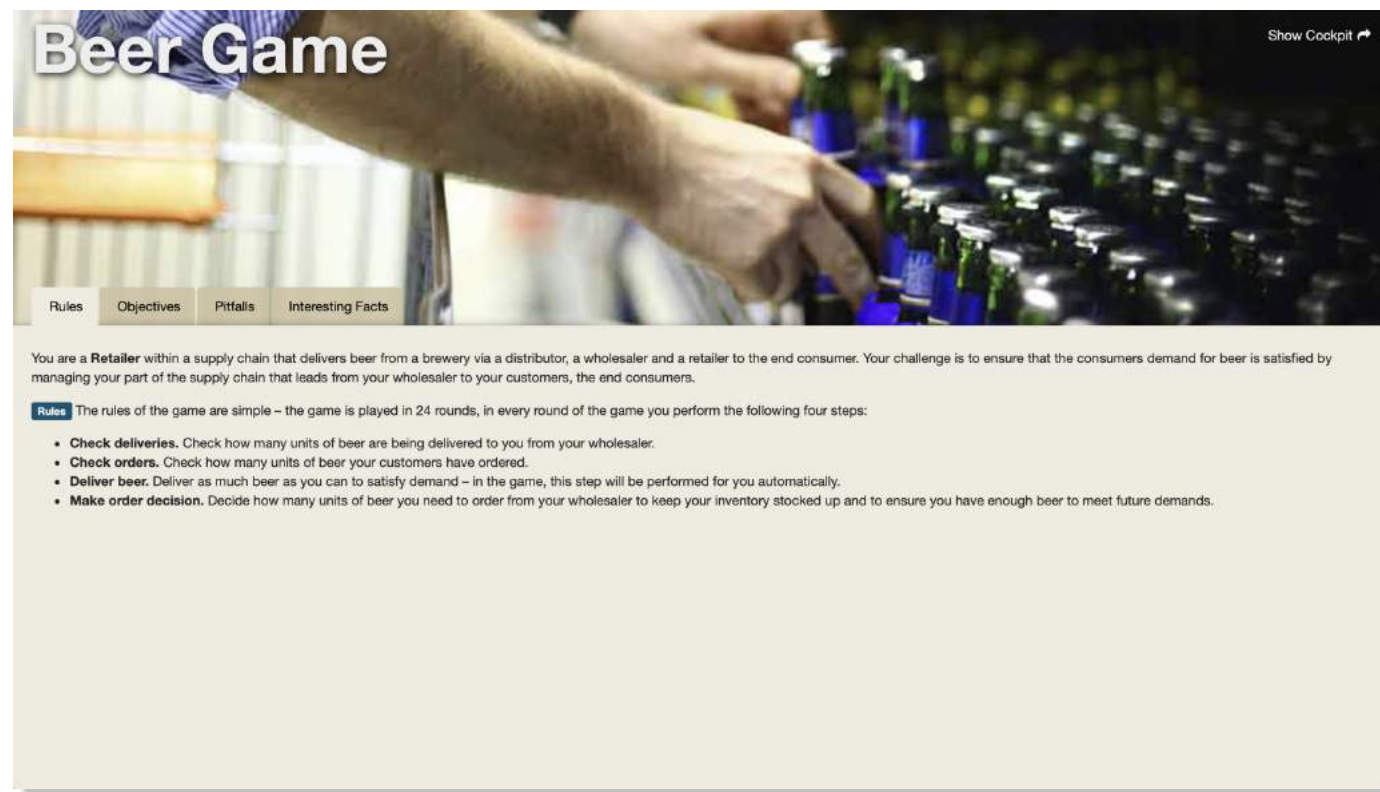
## An Introduction



- The Beer Game was developed in the 1960s at MIT to illustrate how difficult it is to manage dynamic systems – in this case, a supply chain that delivers beer from a brewery to the end consumer
- The game became well known in the 1990s after Peter Senge's description of it in his worldwide best selling book *The Fifth Discipline*
- The game is great because even though it is about a very simple system and despite very simple rules, the resulting behaviour is quite complex.

# A Brief Walk Through The Beer Game

The game is usually played with four players, but the single player version is also fun.

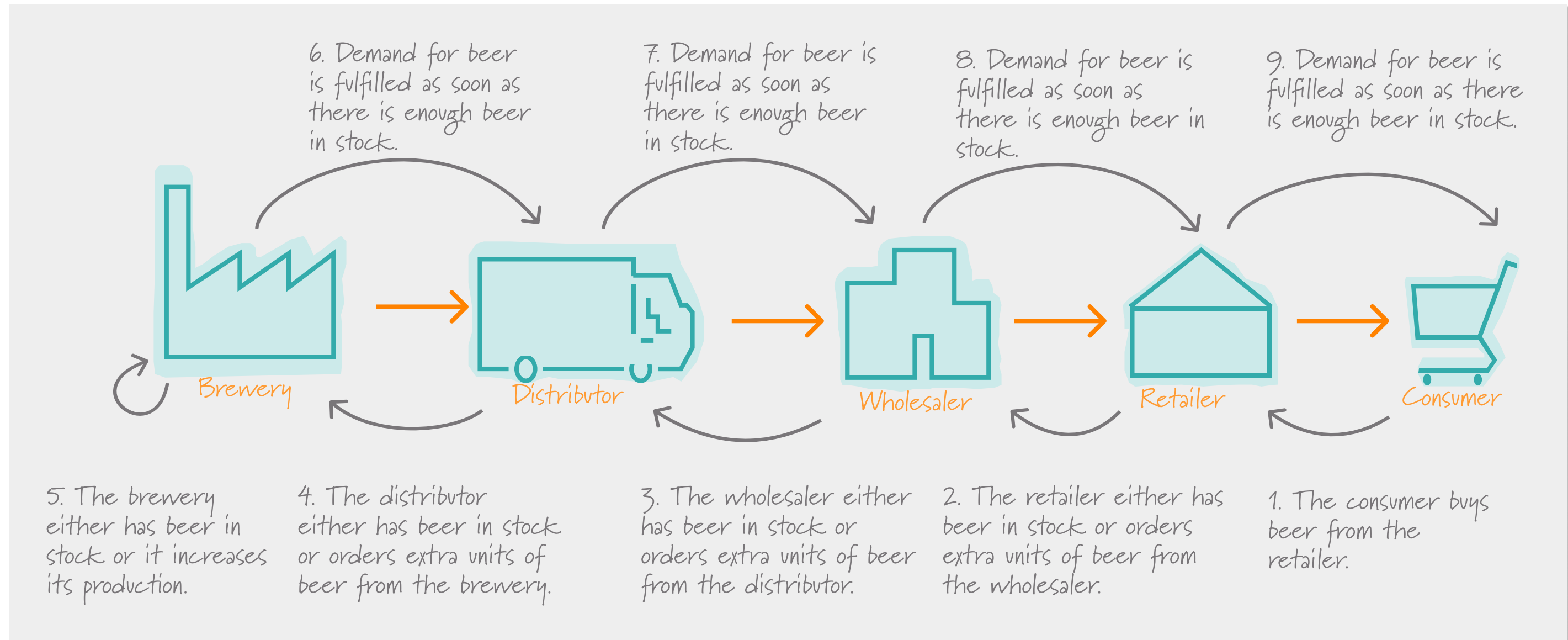


Try it yourself at [beergame.transentis.com](https://beergame.transentis.com)

But let's look at the rules and some potential pitfalls first!

# The Situation

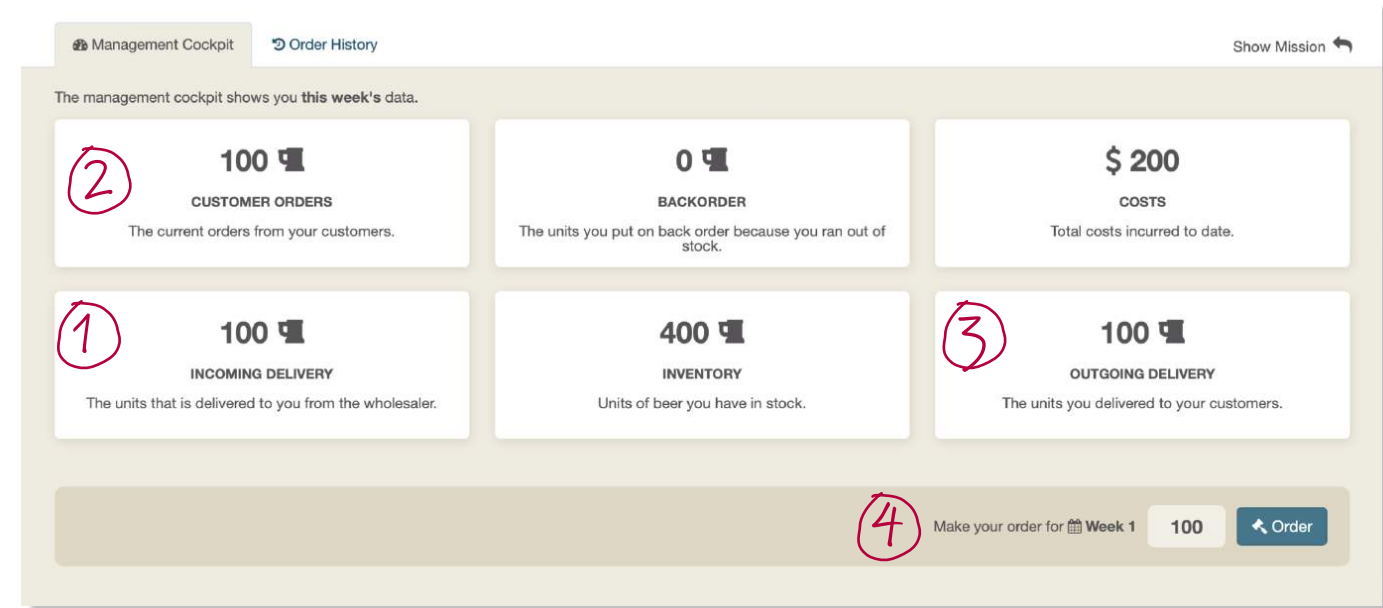
- You are part of a supply chain that delivers beer from a brewery to the end consumer
- Try to meet the demand of your respective customer at all times, while keeping inventory low.



# The Rules

The rules of the game are simple – in every round, you perform the following steps:

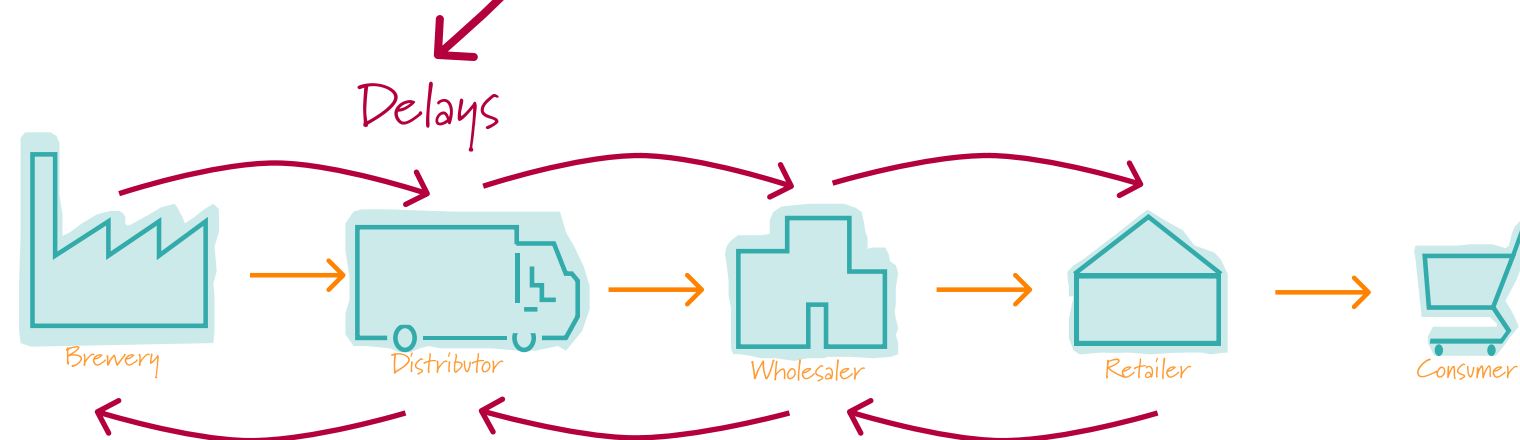
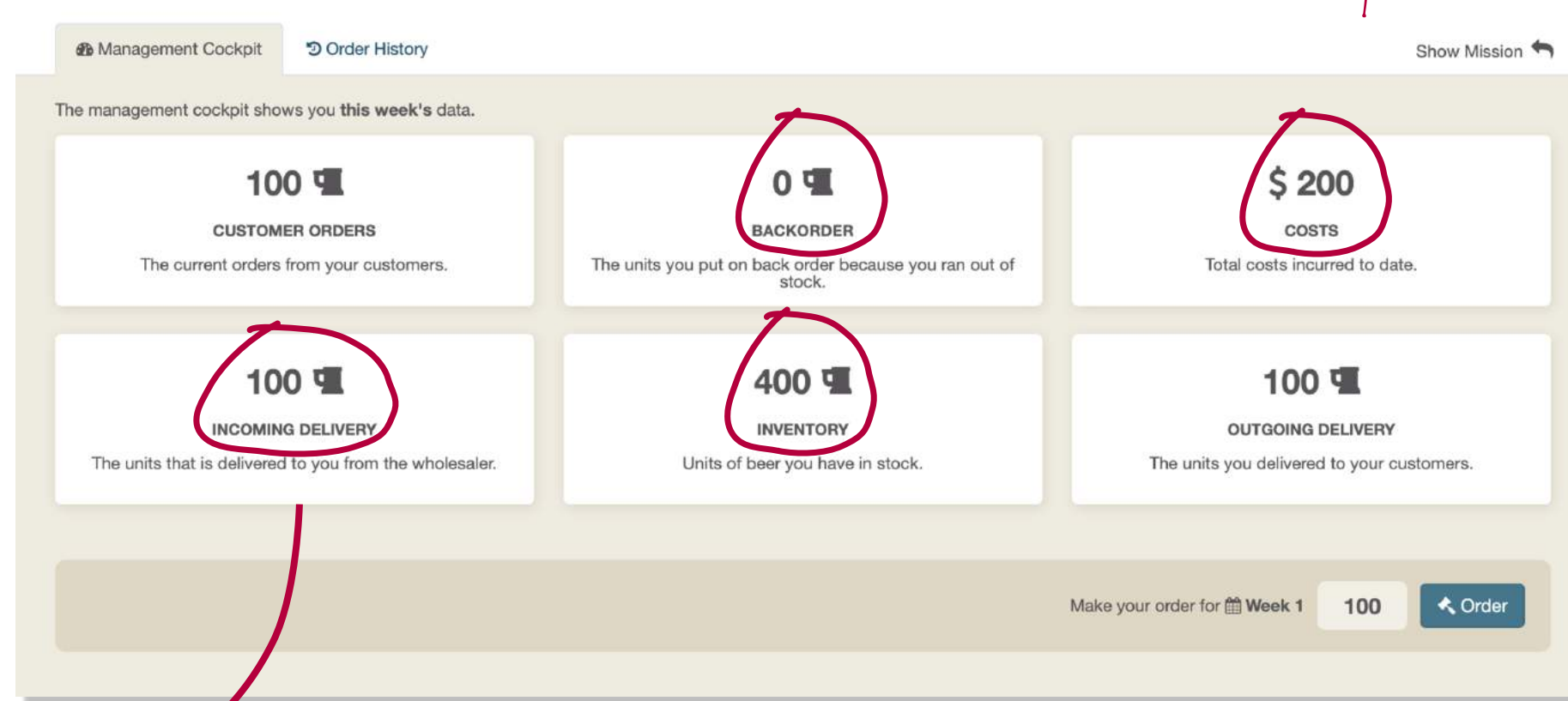
1. **Check deliveries.** Check how many units of beer are being delivered to you from your supplier in the supply chain.
2. **Check orders.** Check how many units of beer your client in the supply chain has ordered.
3. **Deliver beer.** Deliver as much beer as you can to satisfy demand (the game does this for you).
4. **Make an order decision.** Decide how many units of beer you need from your supplier to keep your inventory stocked up.





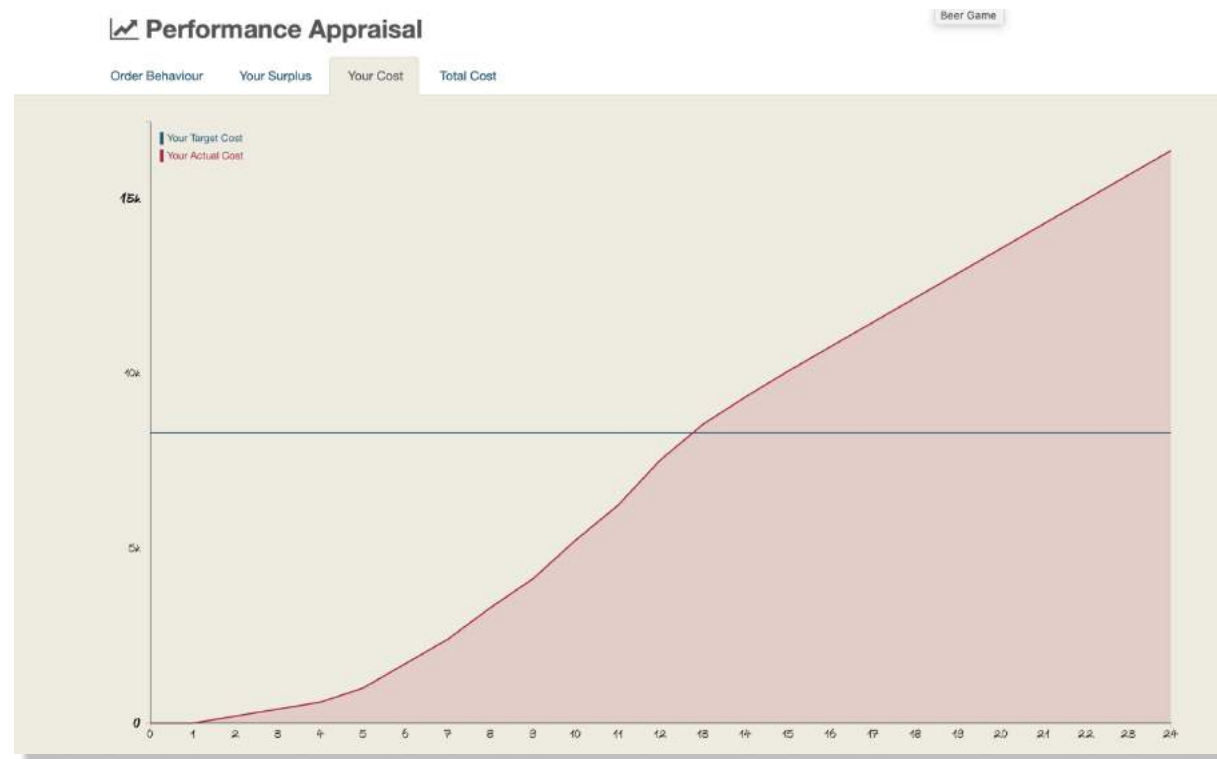
# Some Pitfalls to be Aware of

Backorder and inventory costs



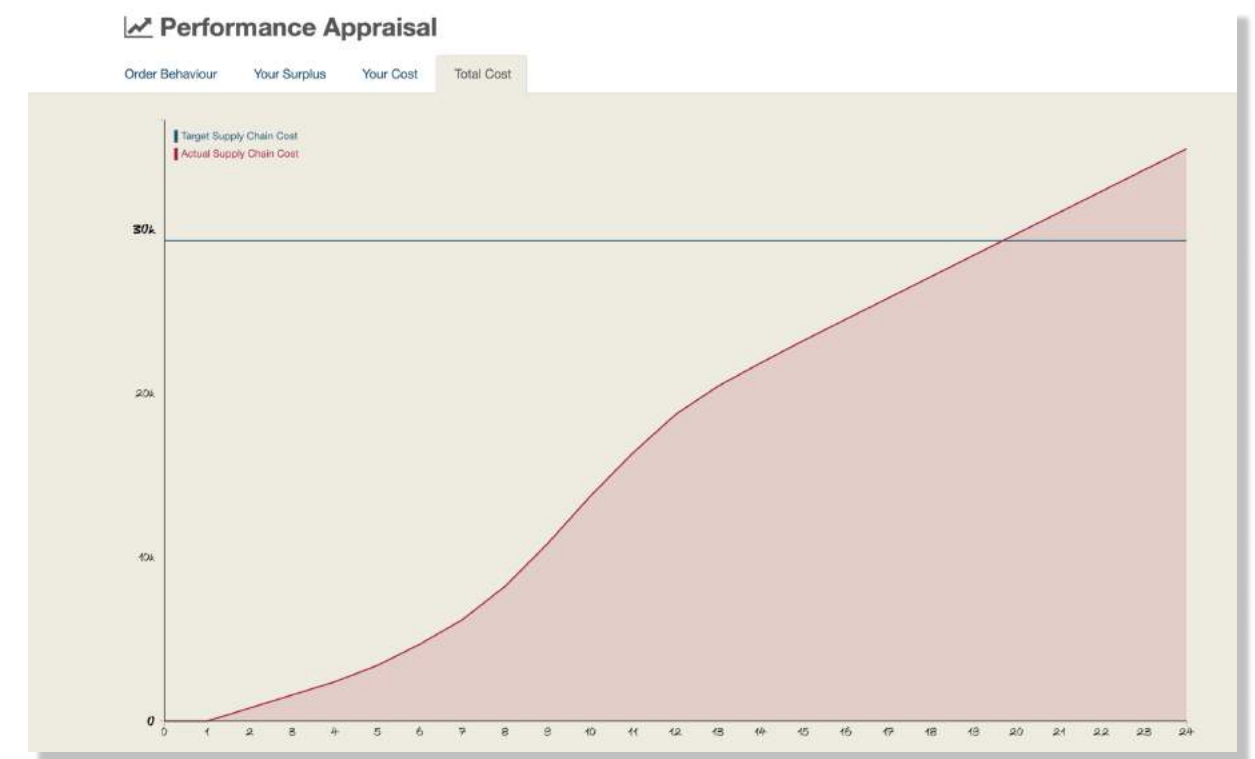


# How Performance is Appraised



## Individual Supply Chain Cost.

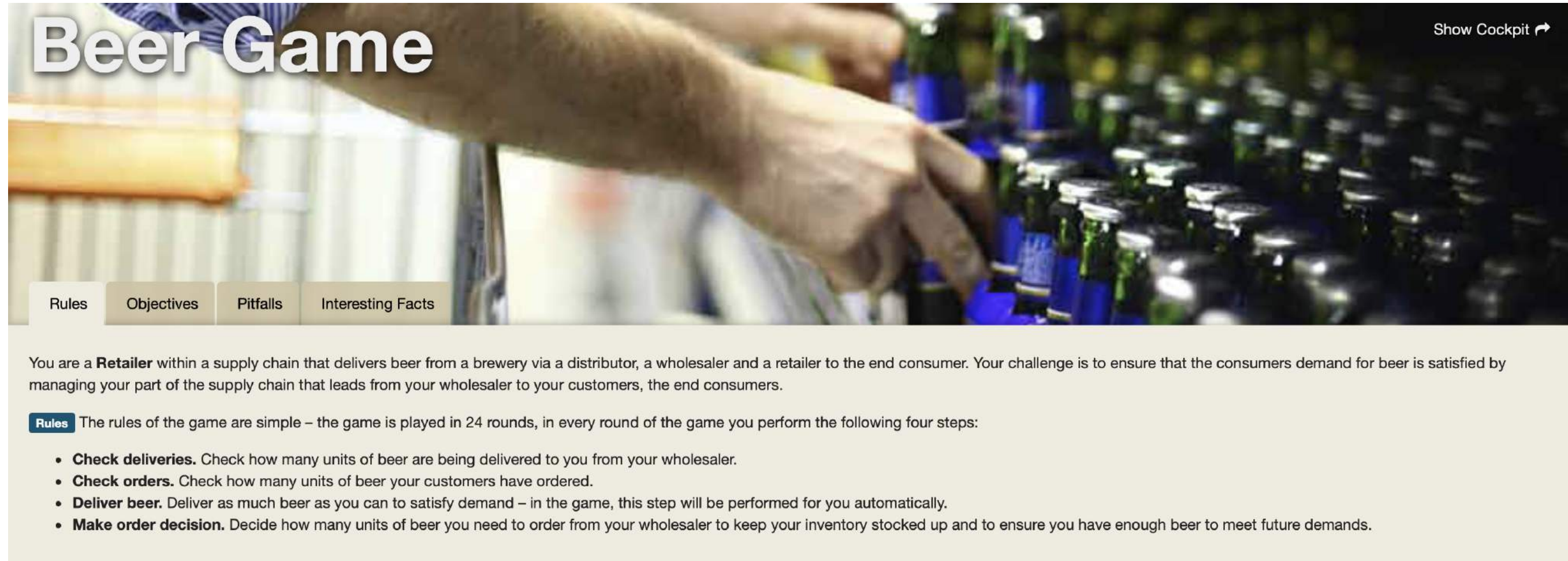
Your accumulated cost should remain below \$8.300.



## Overall Supply Chain Costs.

Total supply chain costs should remain below \$29.300.

# Let's Give it a go

A screenshot of the 'Beer Game' web application. The background image shows hands packaging beer bottles. The title 'Beer Game' is in large white font. A 'Show Cockpit' link with an external icon is in the top right. Below the title are four tabs: 'Rules' (active), 'Objectives', 'Pitfalls', and 'Interesting Facts'. The main text describes the role of a Retailer in a supply chain. The 'Rules' section lists four steps: Check deliveries, Check orders, Deliver beer, and Make order decision.

## Beer Game

[Show Cockpit ↗](#)

[Rules](#) [Objectives](#) [Pitfalls](#) [Interesting Facts](#)

You are a **Retailer** within a supply chain that delivers beer from a brewery via a distributor, a wholesaler and a retailer to the end consumer. Your challenge is to ensure that the consumers demand for beer is satisfied by managing your part of the supply chain that leads from your wholesaler to your customers, the end consumers.

**Rules** The rules of the game are simple – the game is played in 24 rounds, in every round of the game you perform the following four steps:

- **Check deliveries.** Check how many units of beer are being delivered to you from your wholesaler.
- **Check orders.** Check how many units of beer your customers have ordered.
- **Deliver beer.** Deliver as much beer as you can to satisfy demand – in the game, this step will be performed for you automatically.
- **Make order decision.** Decide how many units of beer you need to order from your wholesaler to keep your inventory stocked up and to ensure you have enough beer to meet future demands.

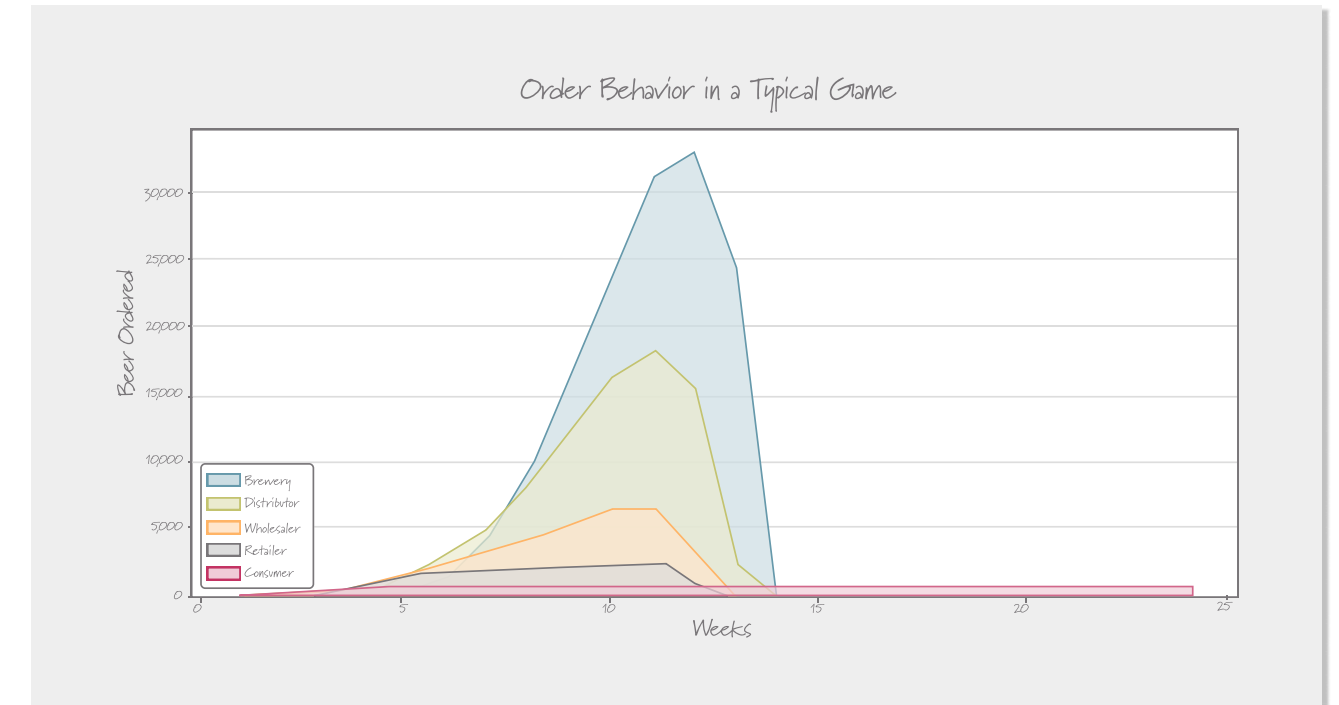
<https://beergame.transentis.com>

# Order Behavior in a Typical Game



## What the consumer orders

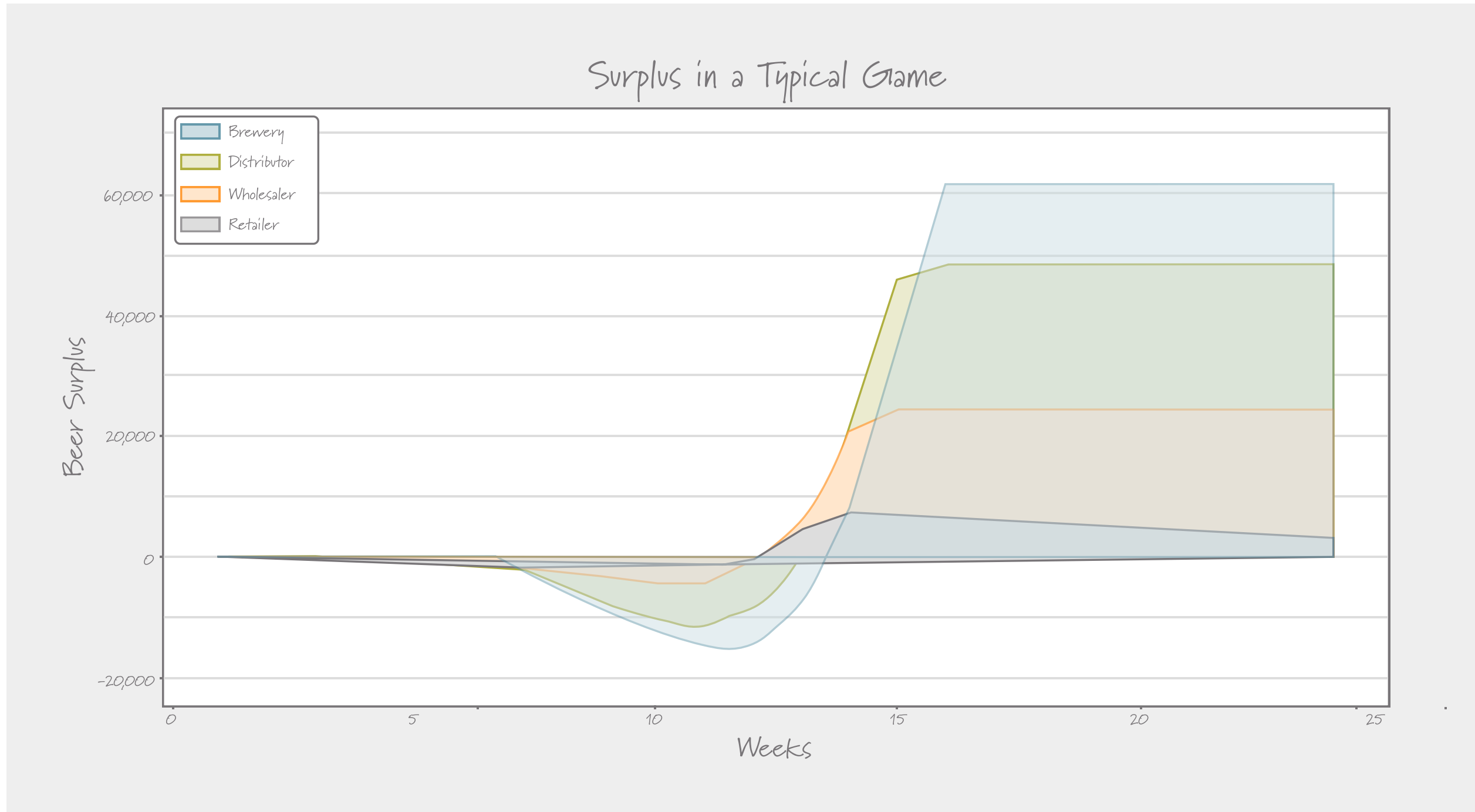
Change in order behavior from 100 units to 400 units



## How the supply chain reacts

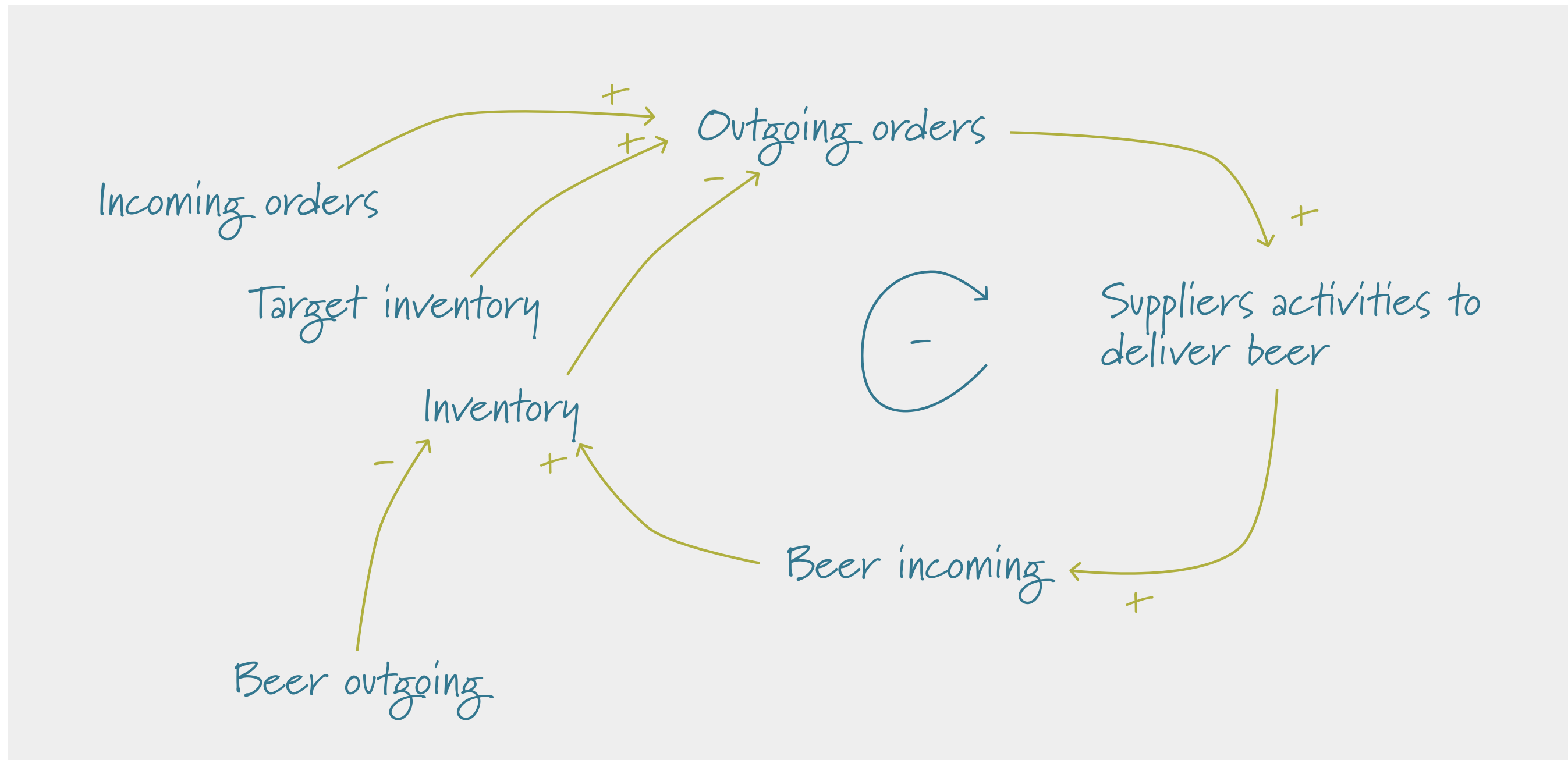
Peak order of over 30,000 units!

# Surplus in a Typical Game

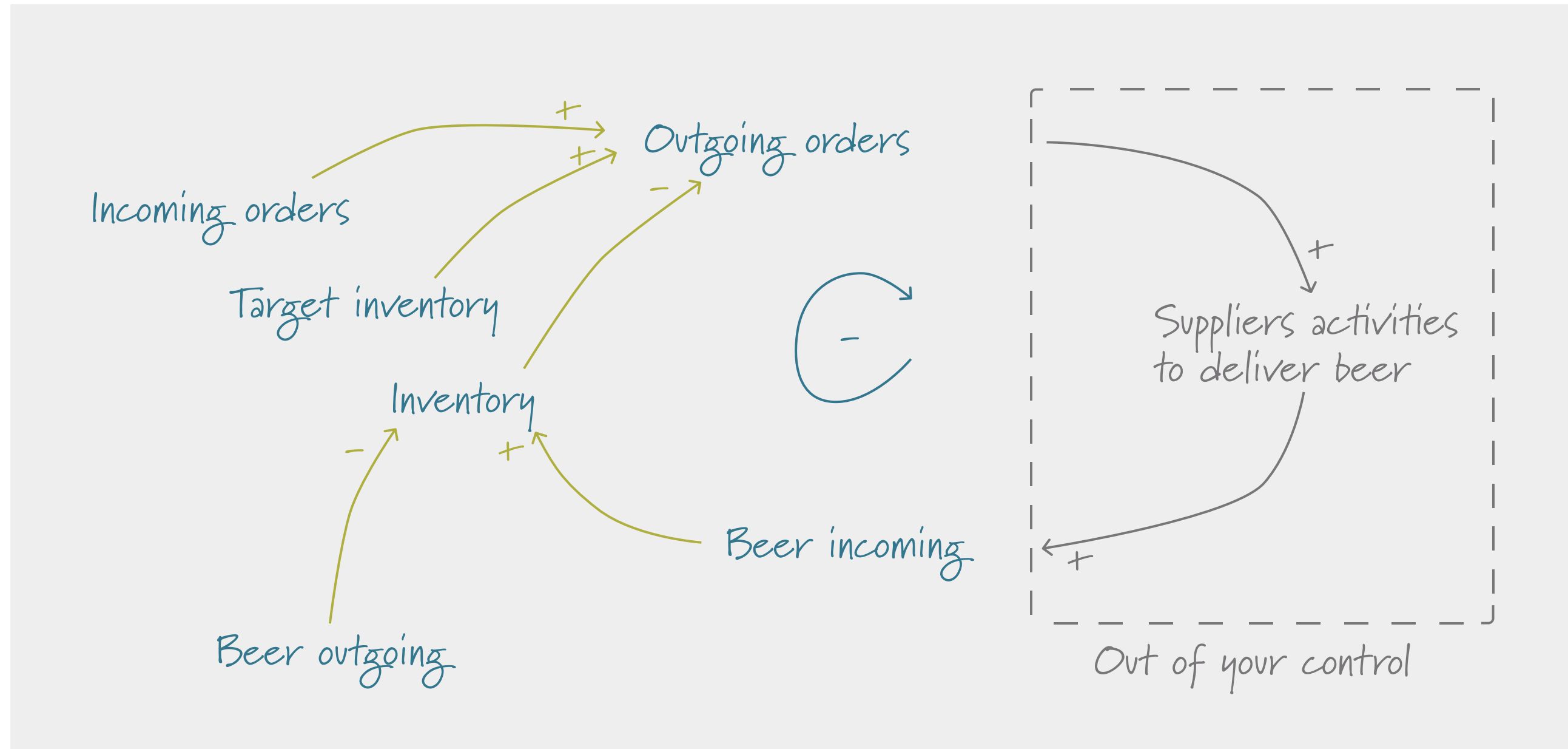




# The Feedback Loop Governing The Supply Chain

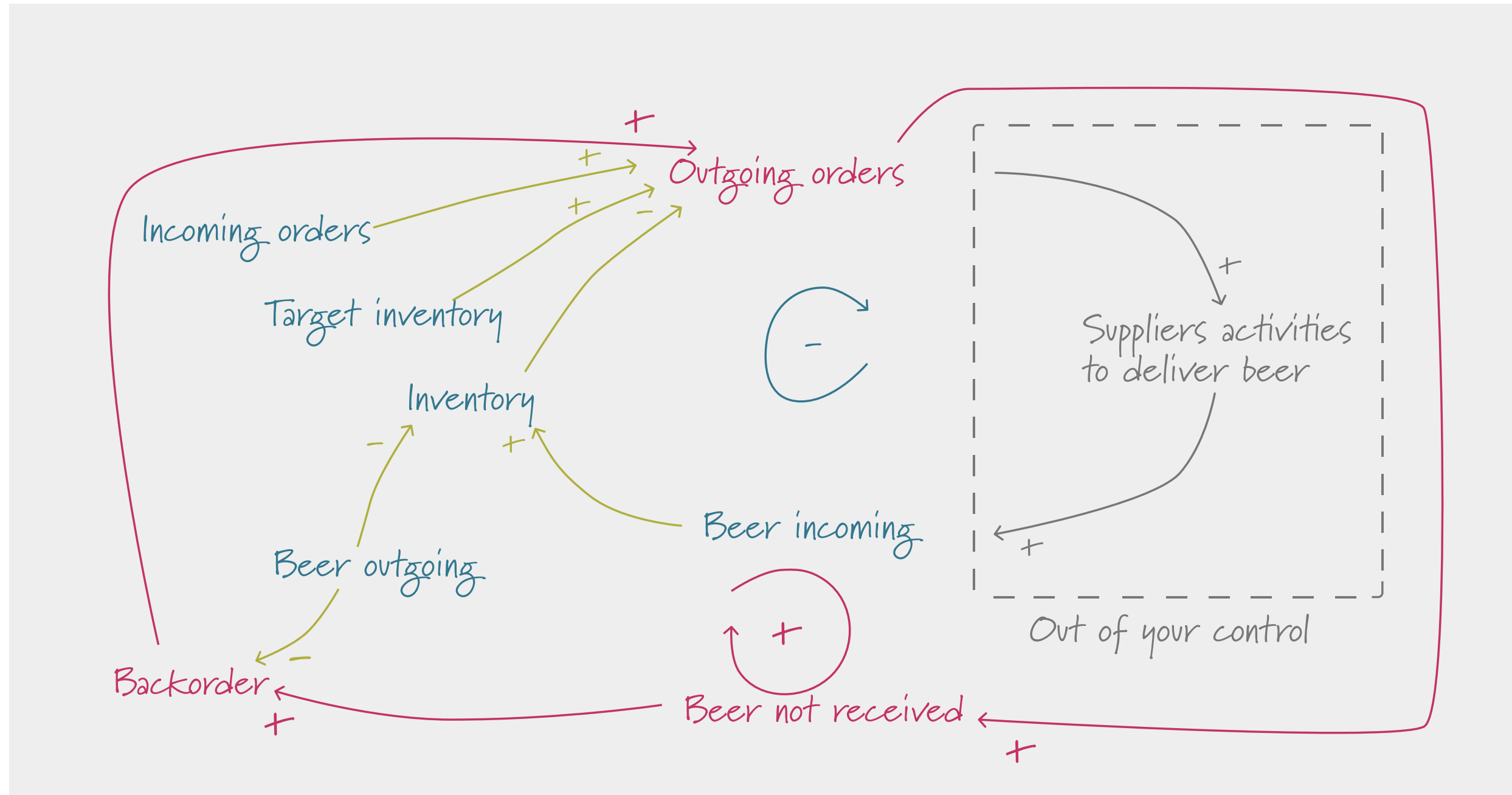


# The Feedback Loop Governing The Supply Chain

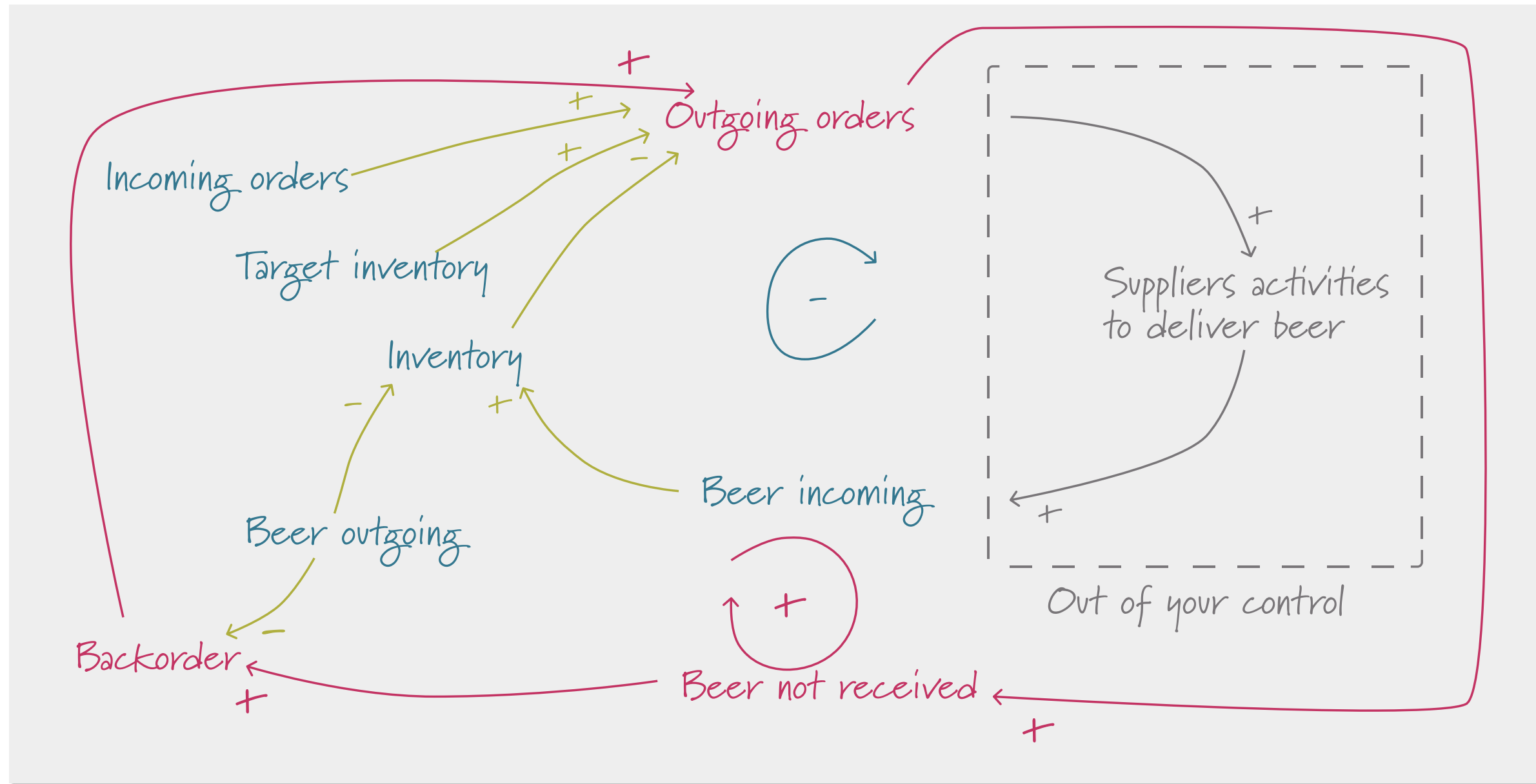


The problem: Each player can only control part of the control loop

# The Error Most People Make: Including The Back Order



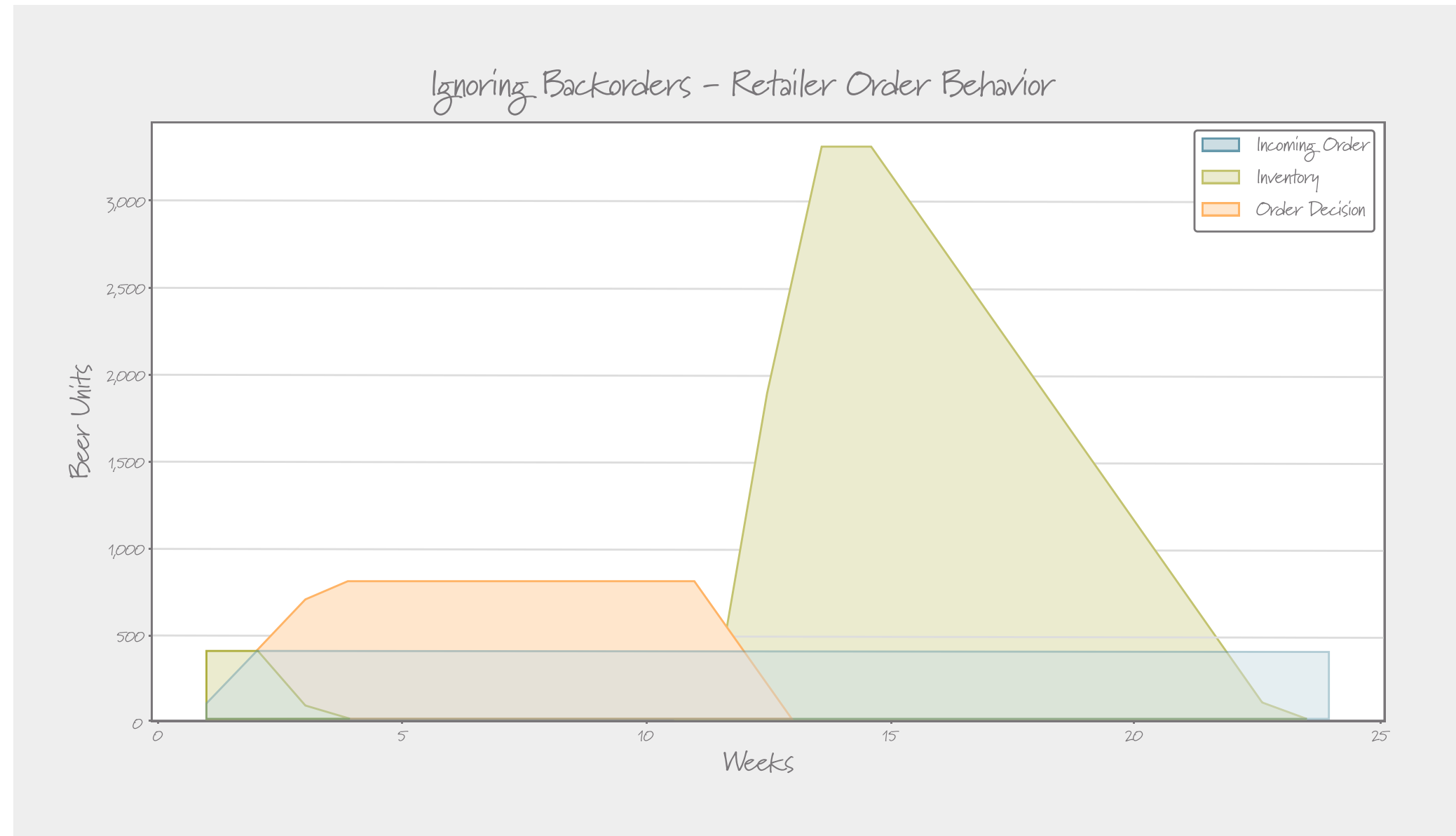
# Improvement Strategy 1: Ignore Back Orders



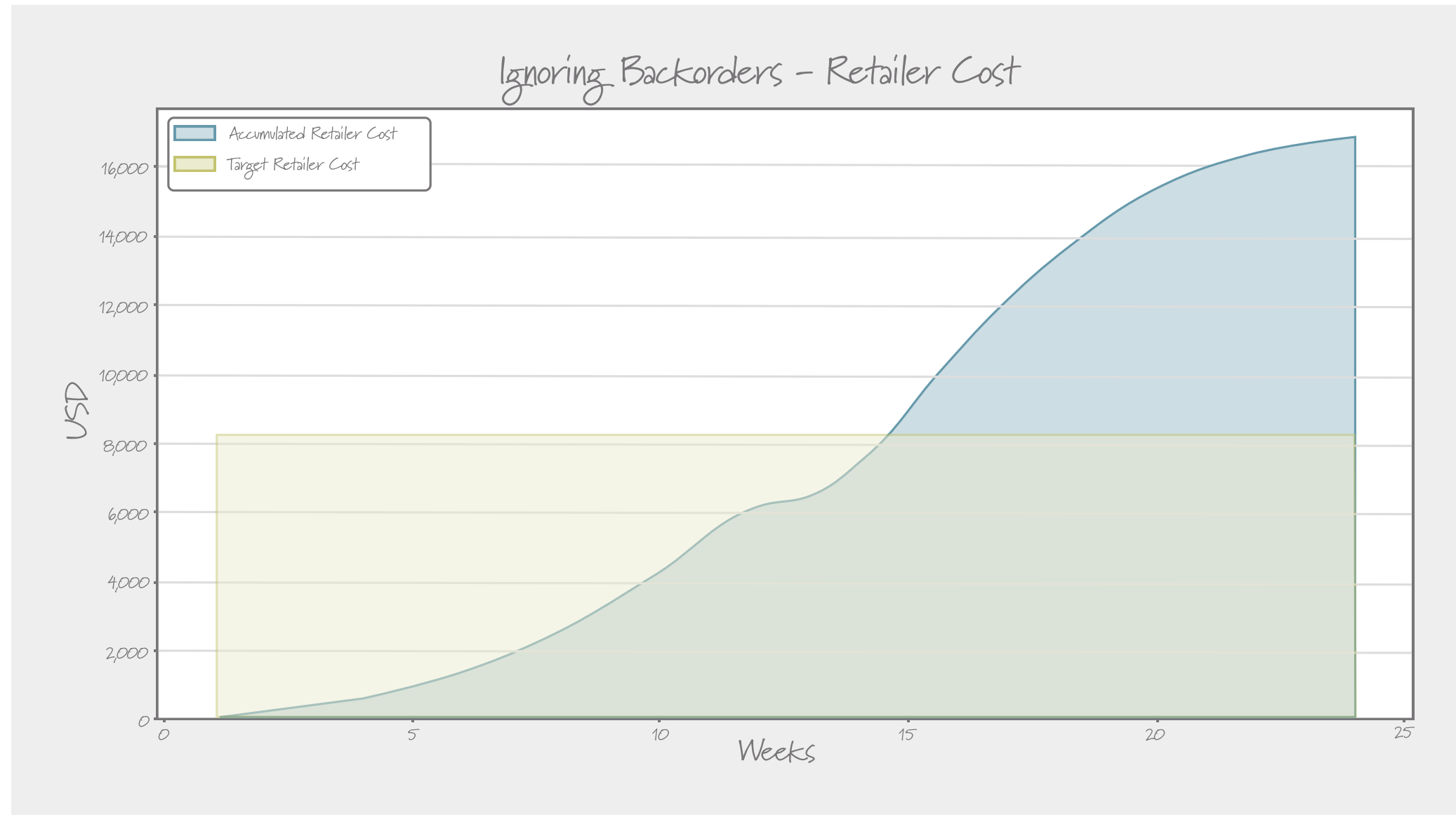
$$\text{Outgoing Orders} = \text{Incoming Orders} + \text{Target Inventory} - \text{Inventory}$$



# Improvement Strategy 1: Ignore Back Orders

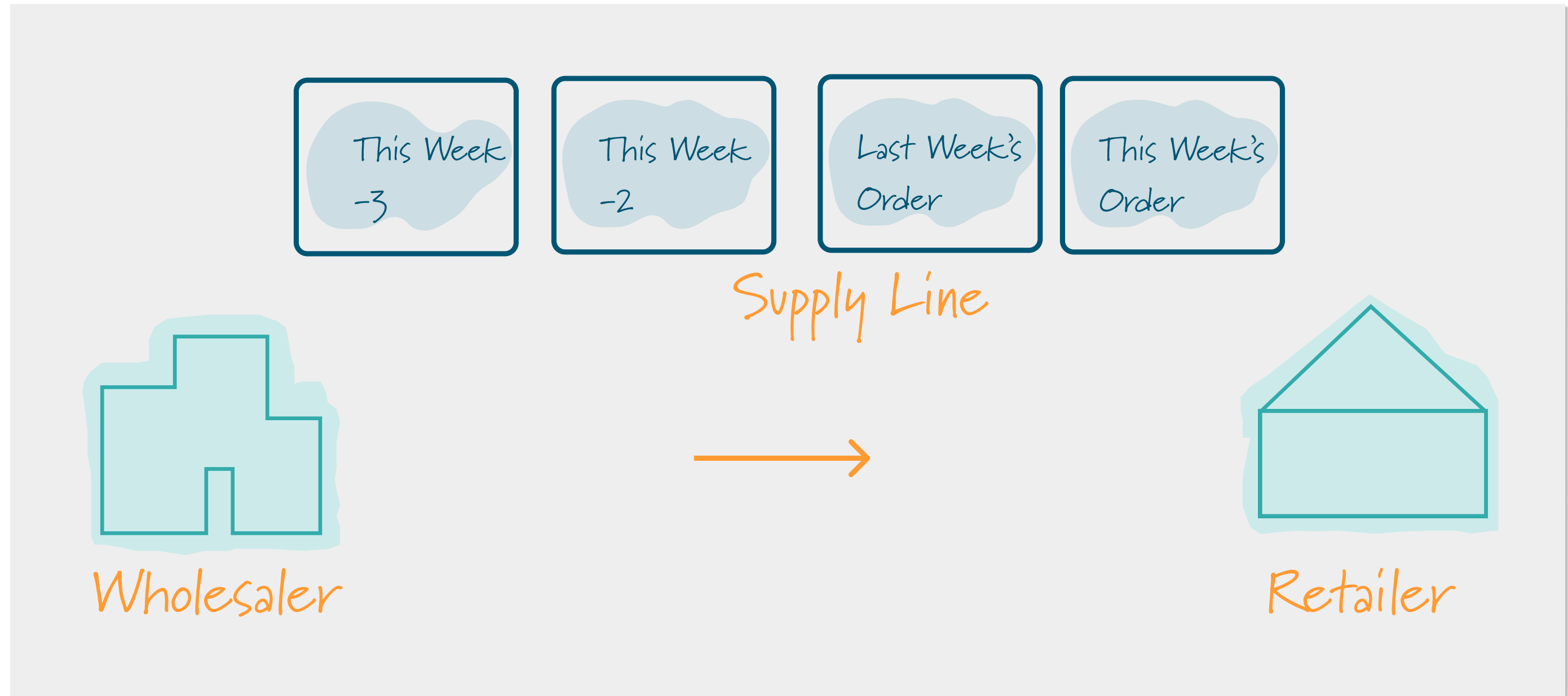


# Improvement Strategy 1: Ignore Back Orders



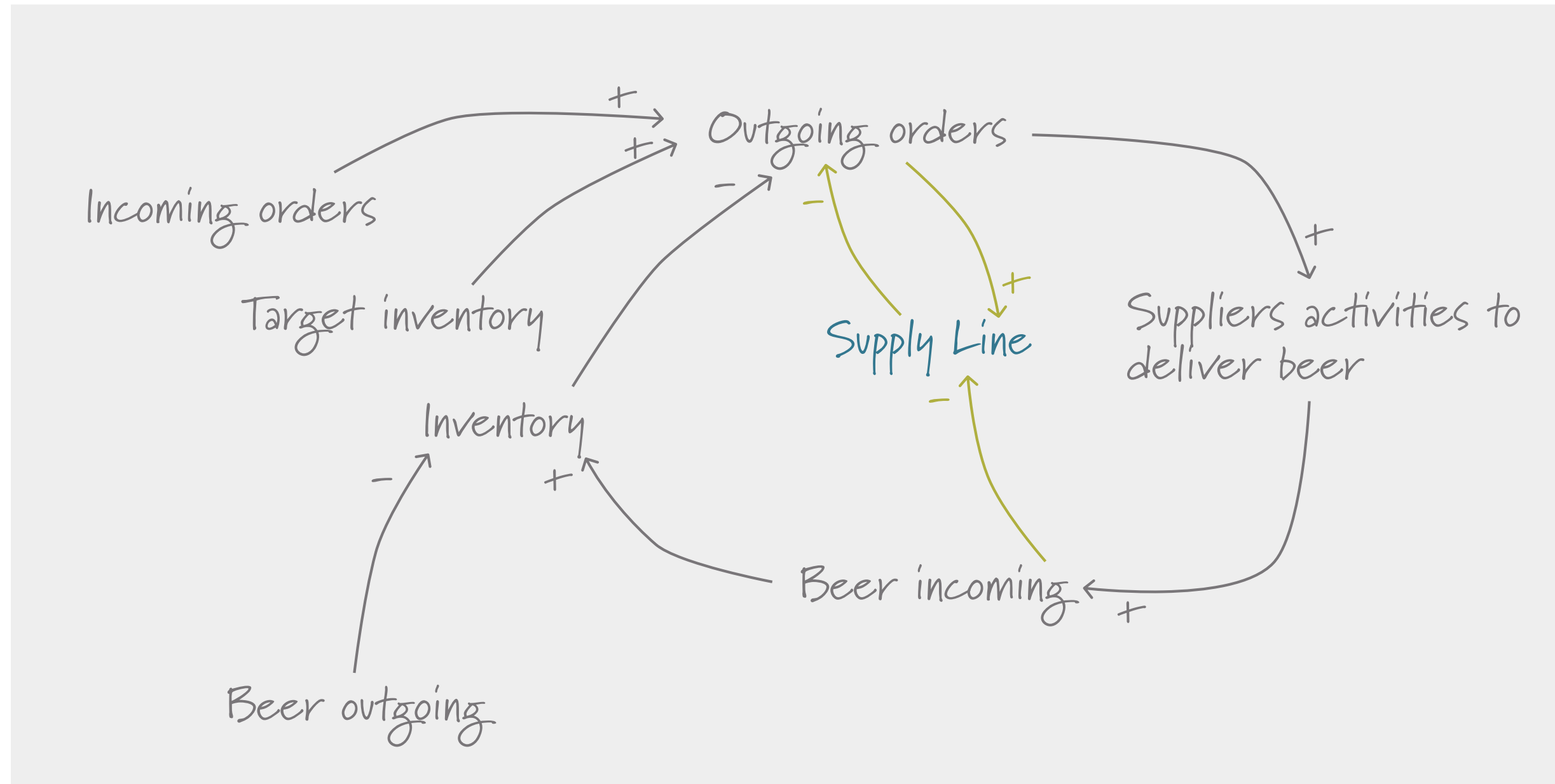
# Improvement Strategy 2: Remember Open Orders

Remember the orders that are in the supply line



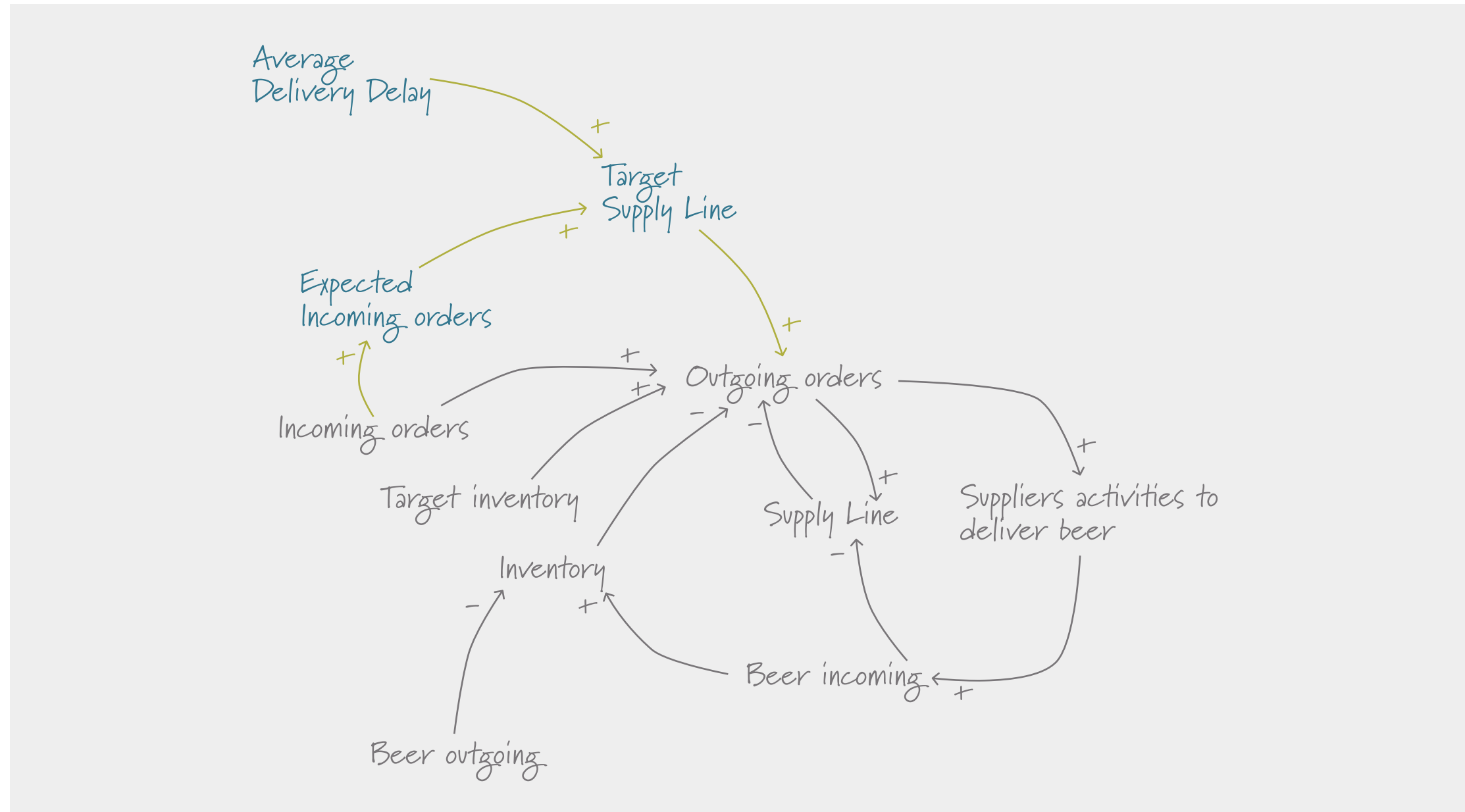
# Improvement Strategy 2: Remember Open Orders

Remember the orders that are in the supply line





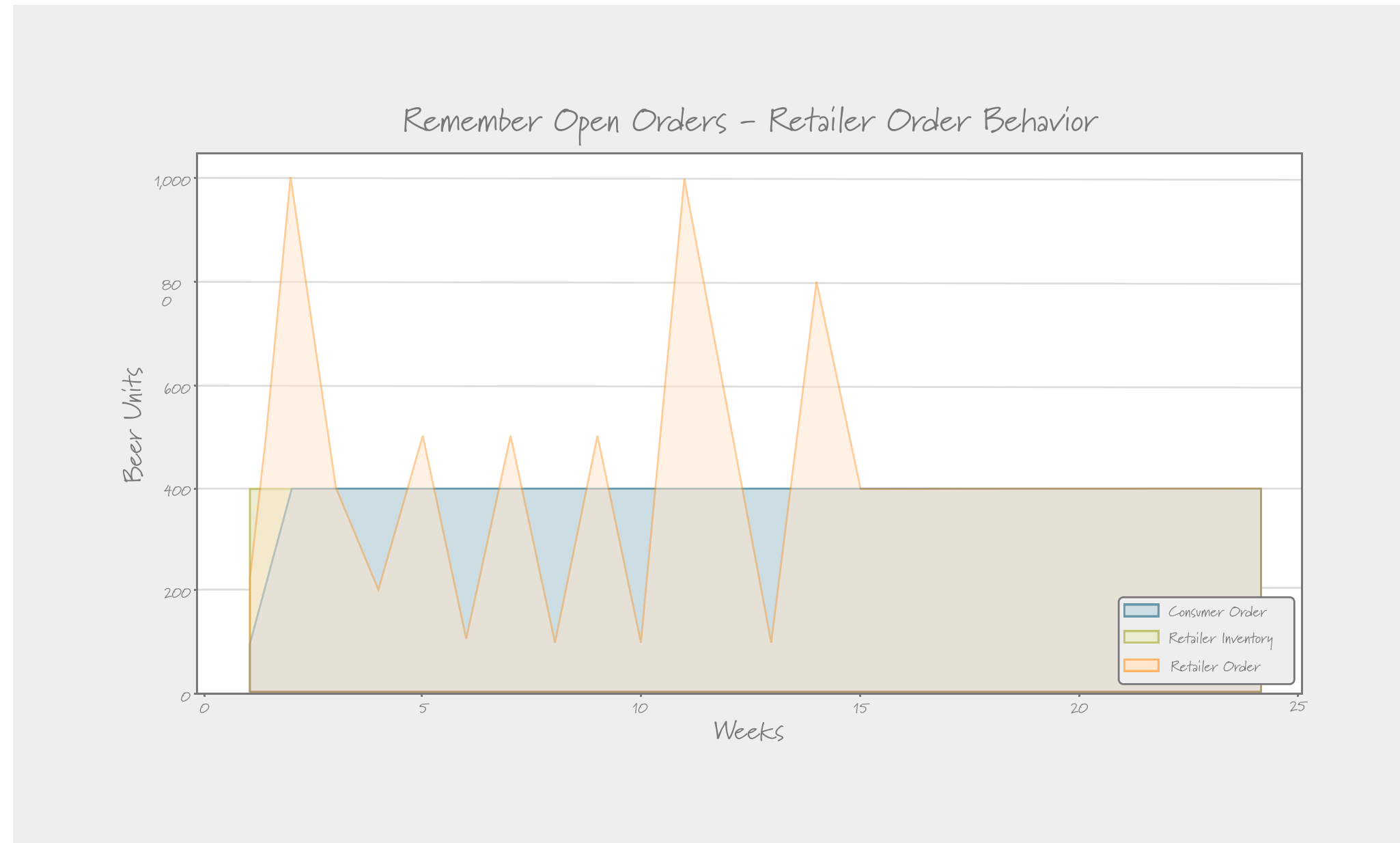
# The Target Supply Line Depends on The Delivery Delay



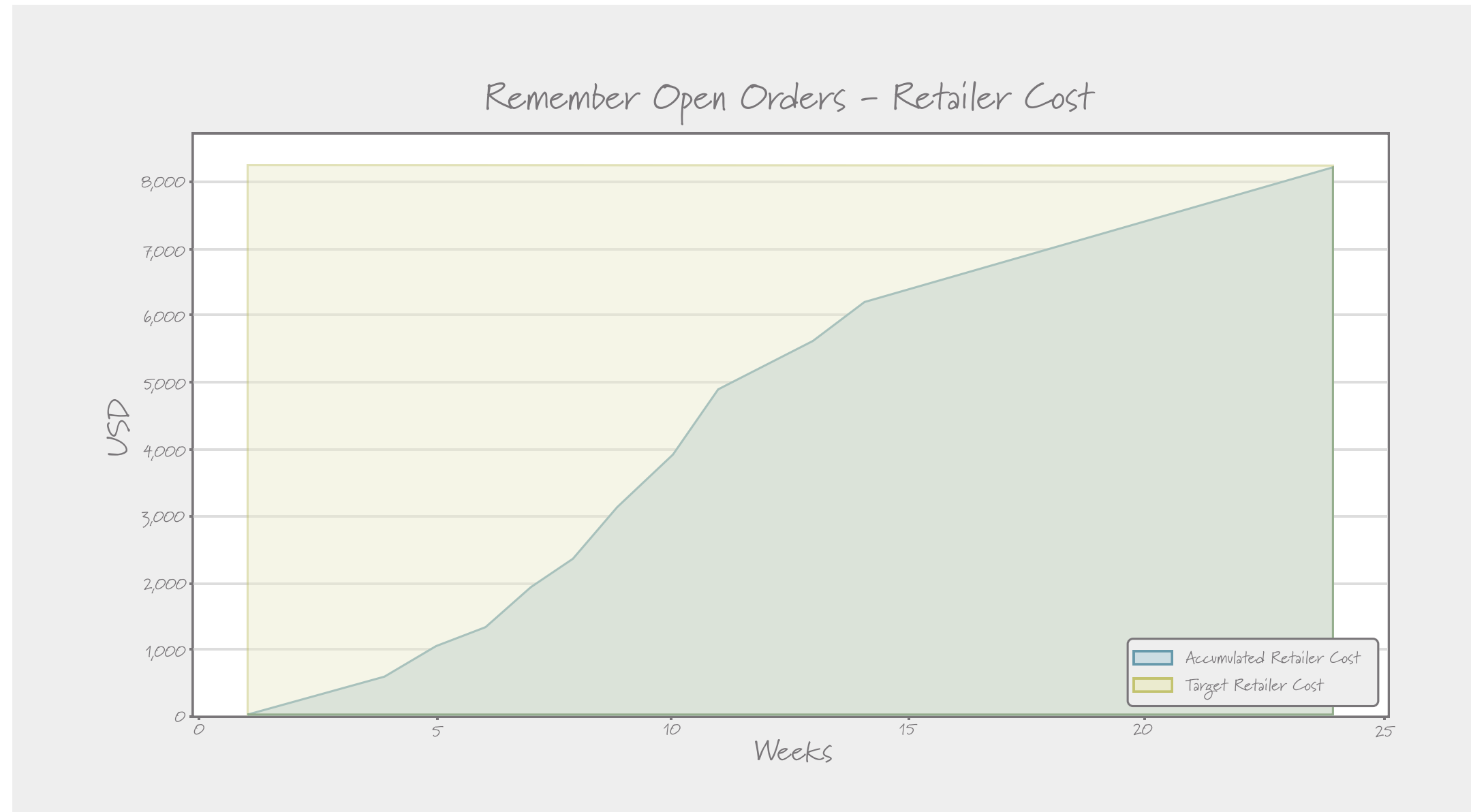
Target Supply Line = Delivery Delay \* Incoming Orders

Orders = Incoming Orders + Target Inventory - Inventory + Target Supply Line - Supply Line

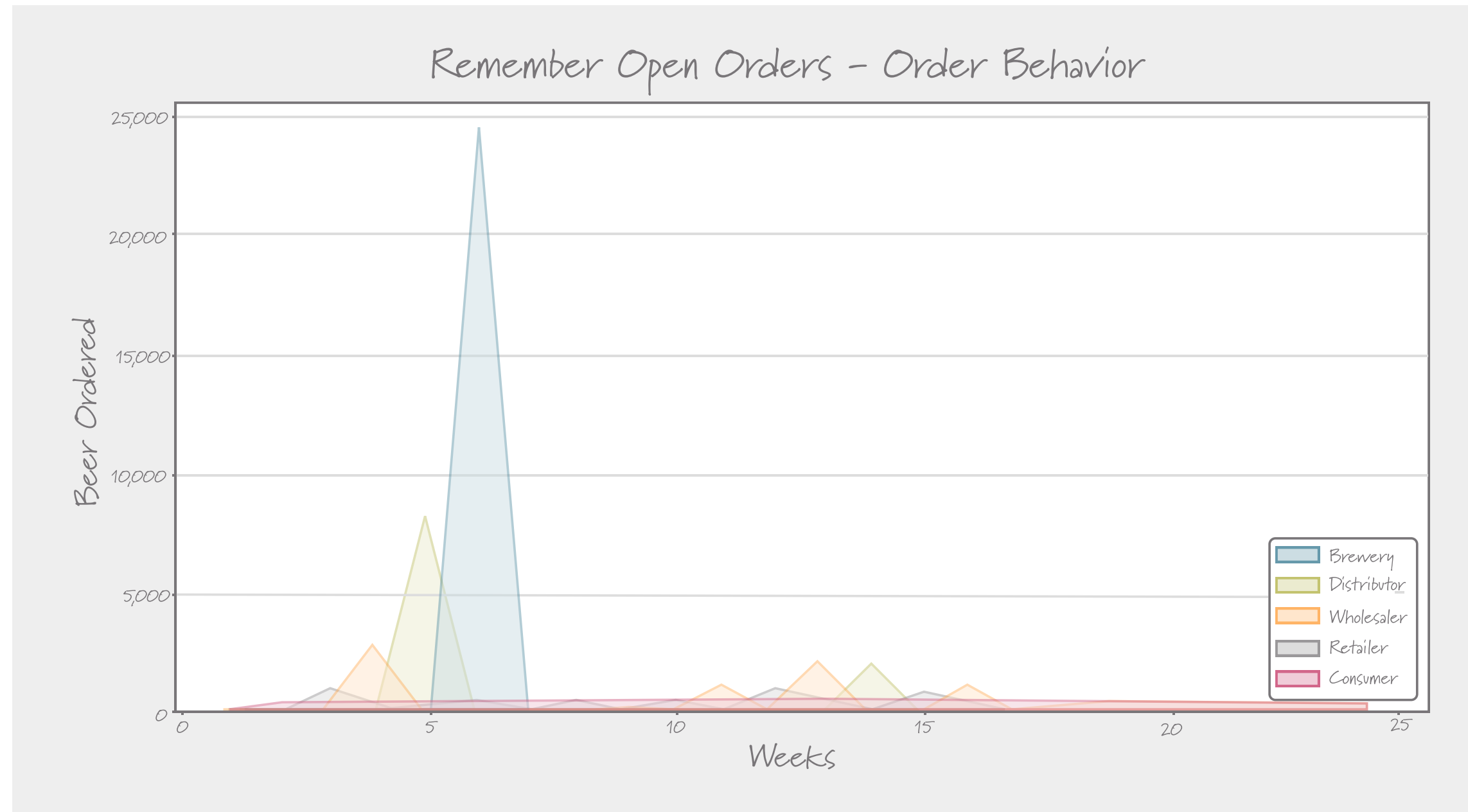
# Order Behavior With The New Ordering Policy



# The Individual Cost Target is Now Reached

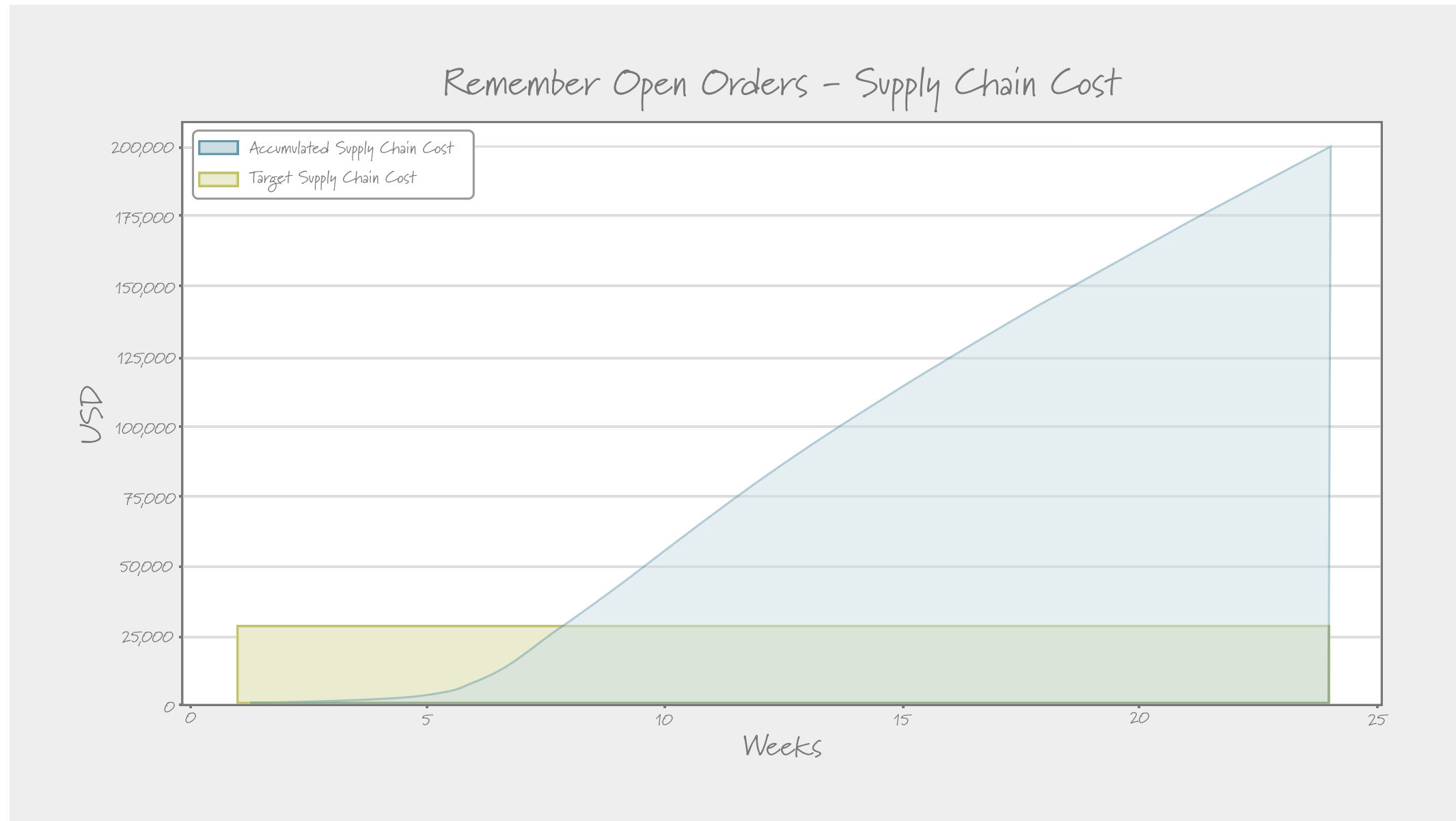


# The Whiplash Effect



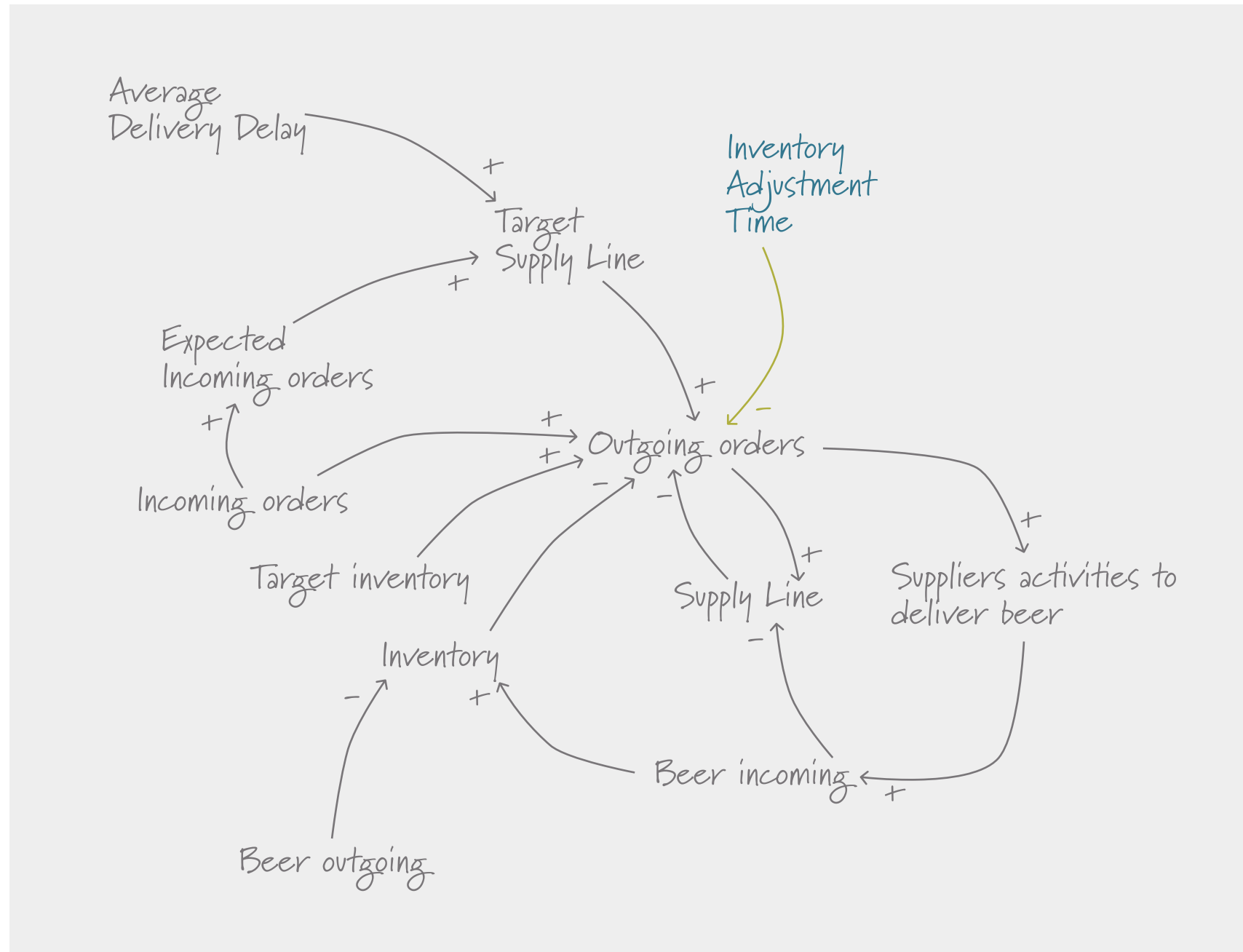
Even if every player behaves rationally, there will still be a "whiplash" effect – because the orders become successively larger along the supply chain.

# Supply Chain Costs Are Still Way Off Target



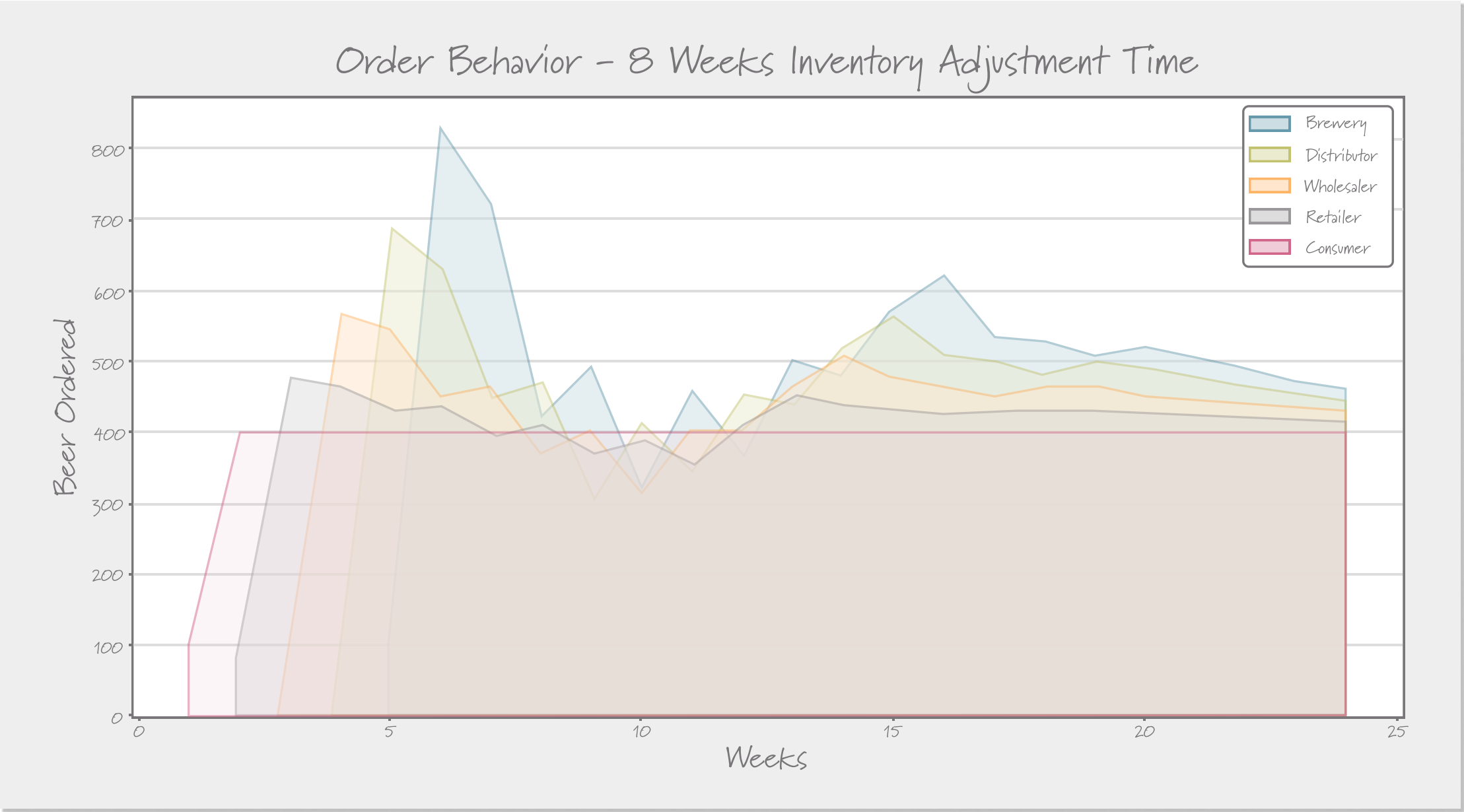
# Improvement Strategy 3: Increase Inventory Adjustment Time

The solution to dealing with the whiplash effect is to adjust inventory slowly!

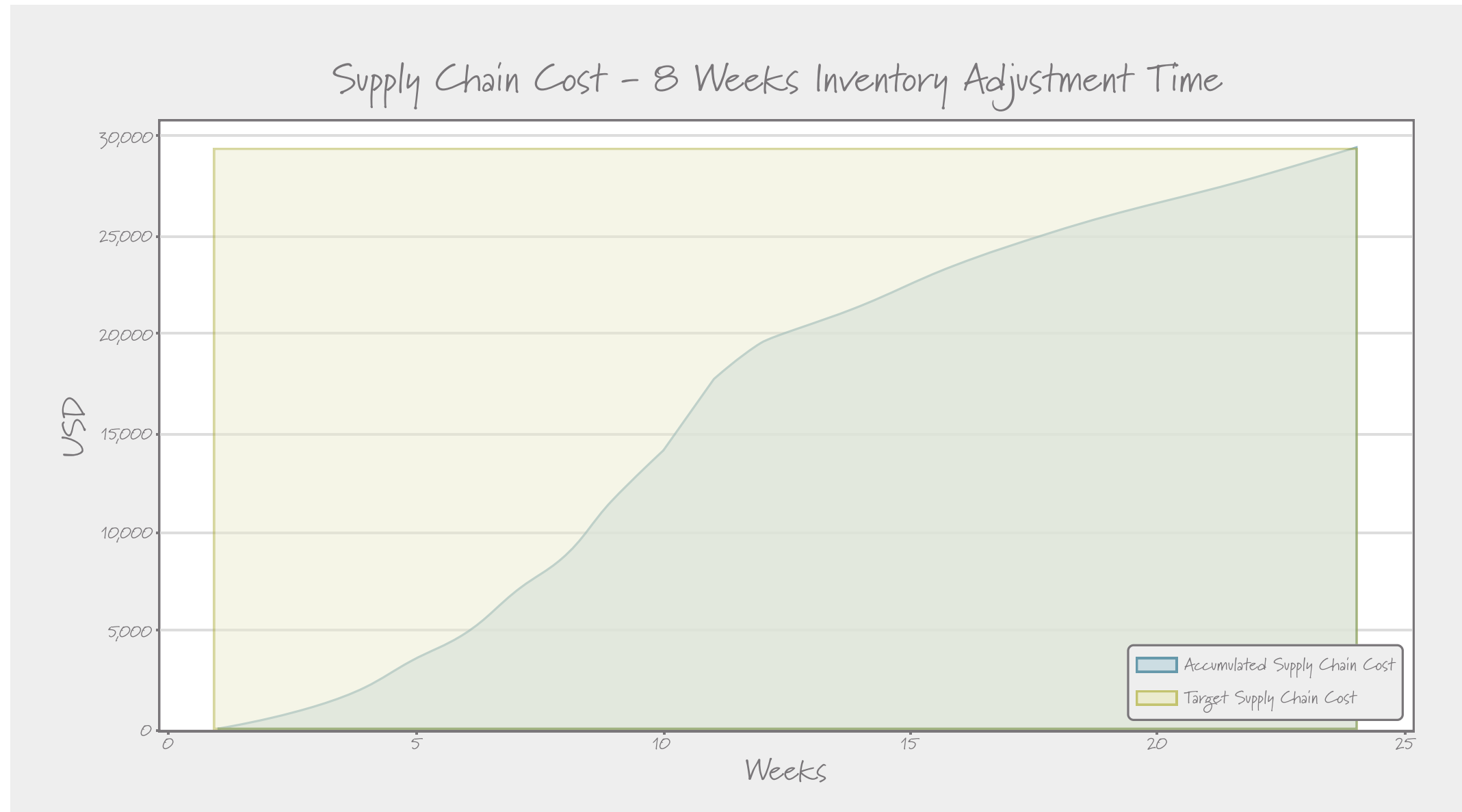




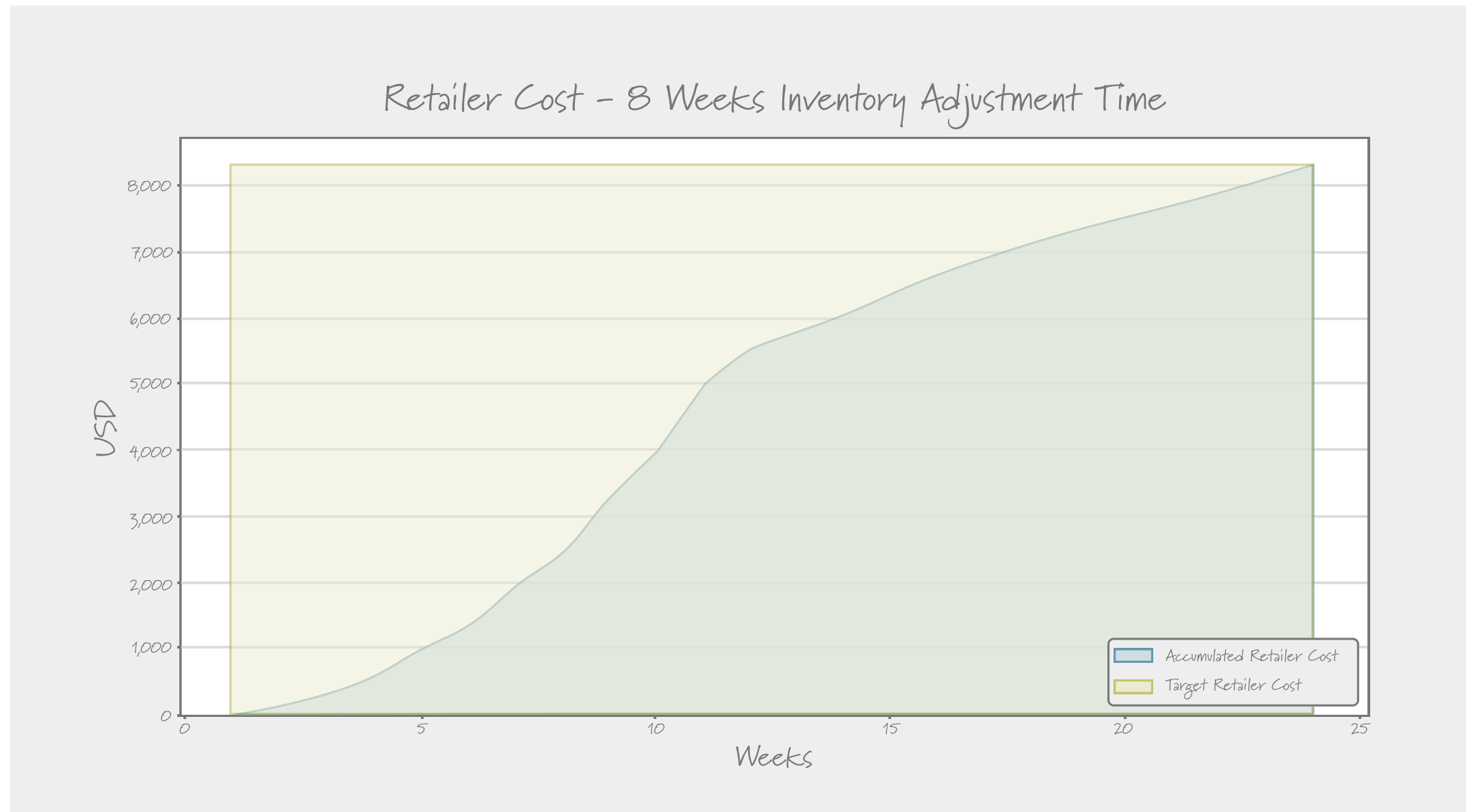
# Improved Order Behavior



# Target Supply Chain Costs Are Met



# Target Retailer Costs Are Also Met



# Summary of The Computational Modeling Approach

Build a computational model to capture your understanding of the system  
Use simulations to test the effect of different policies  
Find policies that help you reach your targets

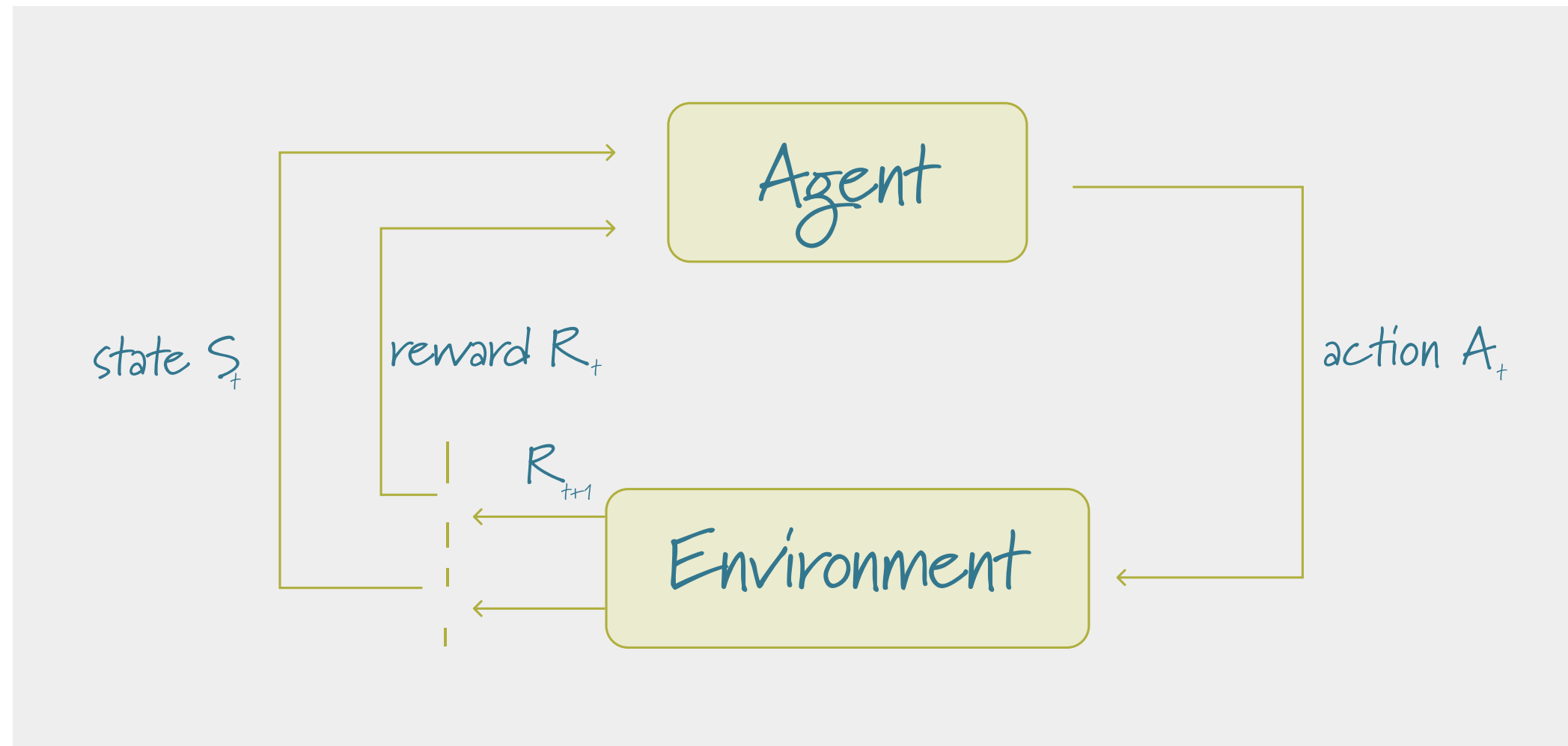
## Concrete learnings from the Beer Game

Controlling complex systems is hard and even harder when you only have partial control  
We often don't behave rationally when overwhelmed with information and under pressure  
Sometimes it pays to "have faith in the system"  
Small changes can have large effects  
Even if everybody optimises locally, this doesn't necessarily lead to a global optimum

# Training AI to Play The Beergame

## An approach using reinforcement learning

The idea: use autonomous agents to play the game and train them using reinforcement learning, a machine learning technique.



Agents have information about their environment.

They perform actions and receive rewards (or punishments) for them.

They learn through trial and error.

# Agents For The Beer Game

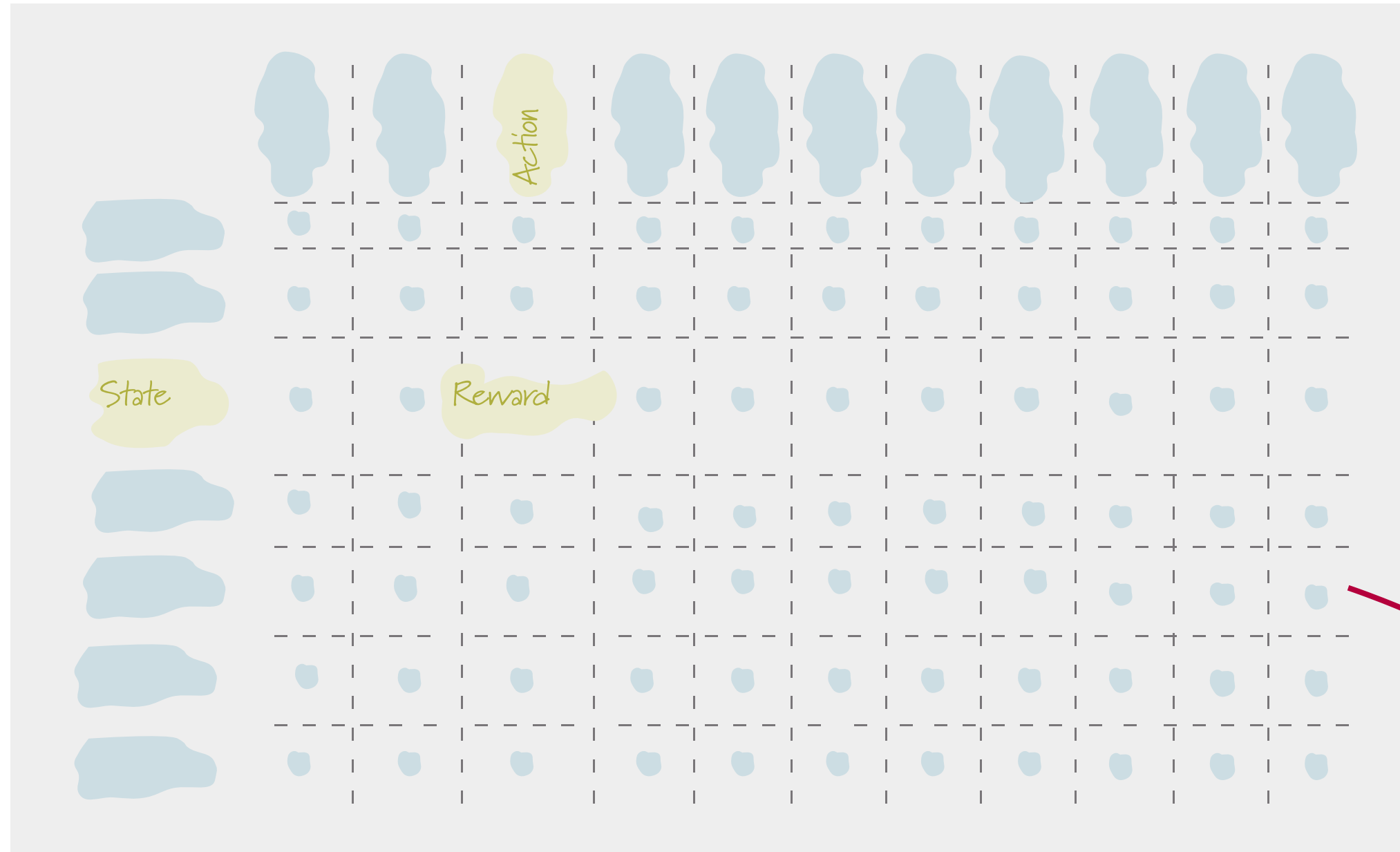
The agents for the Beer Game are very simple





# Q-Learning

A reinforcement learning technique



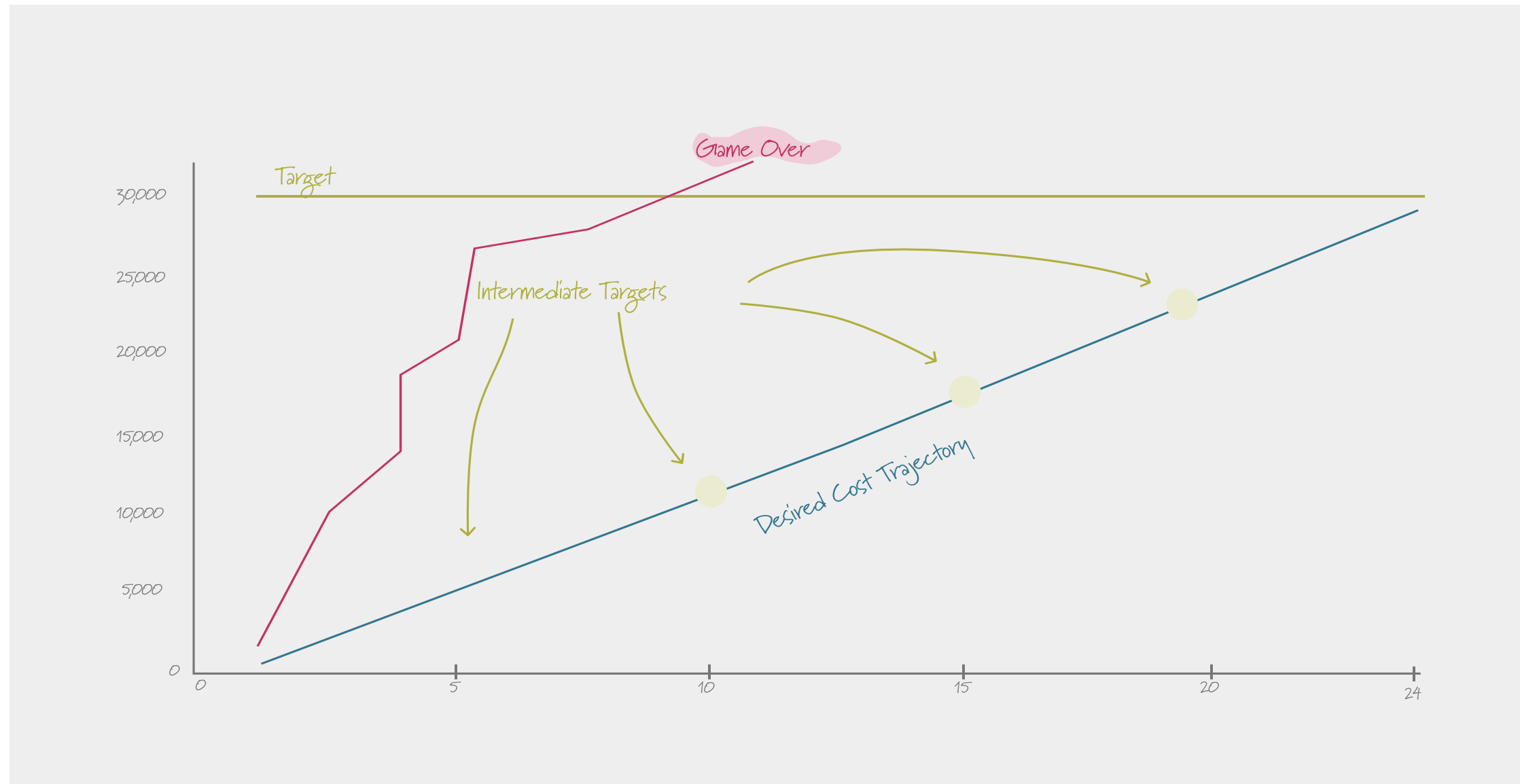
Each agent has a "Q-Table", which defines the expected total reward for each state and each possible action.

Agents start with an "empty" q-table and then fill it by learning through trial and error.

*Always choose the action with the highest reward!*

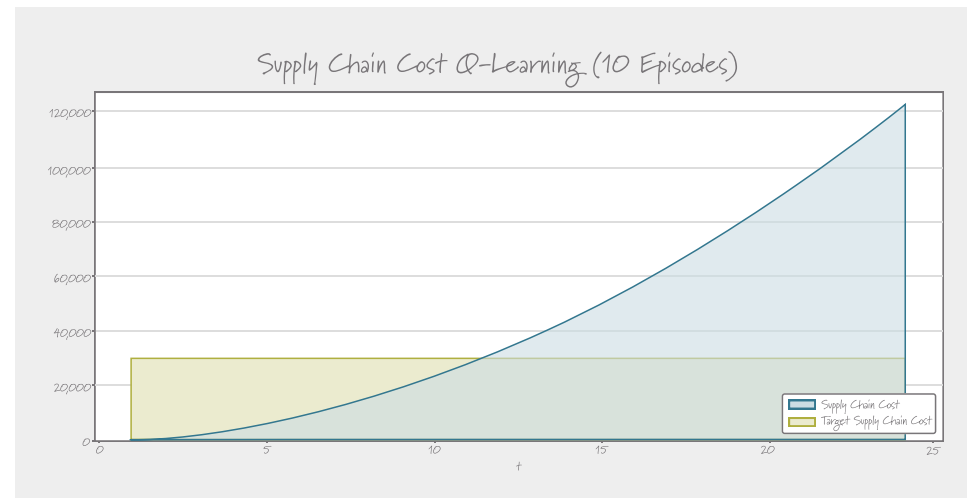
# The Key: Setting The Right Rewards

Much like with us human beings, we need to set the right rewards

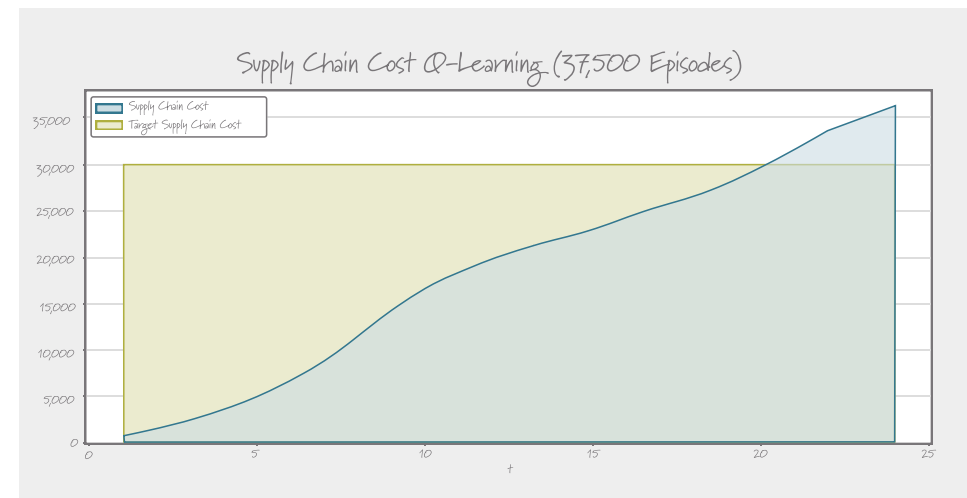


A reward for reaching the cost targets (and some milestones along the way)  
"Game over" as soon as cost targets are missed (to avoid wasting time and memory)

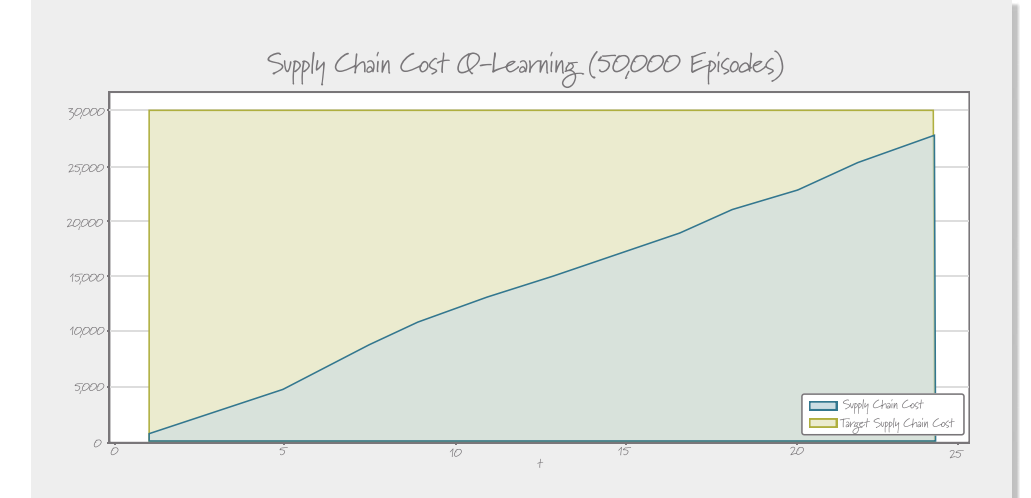
# The Results



Ten Episodes



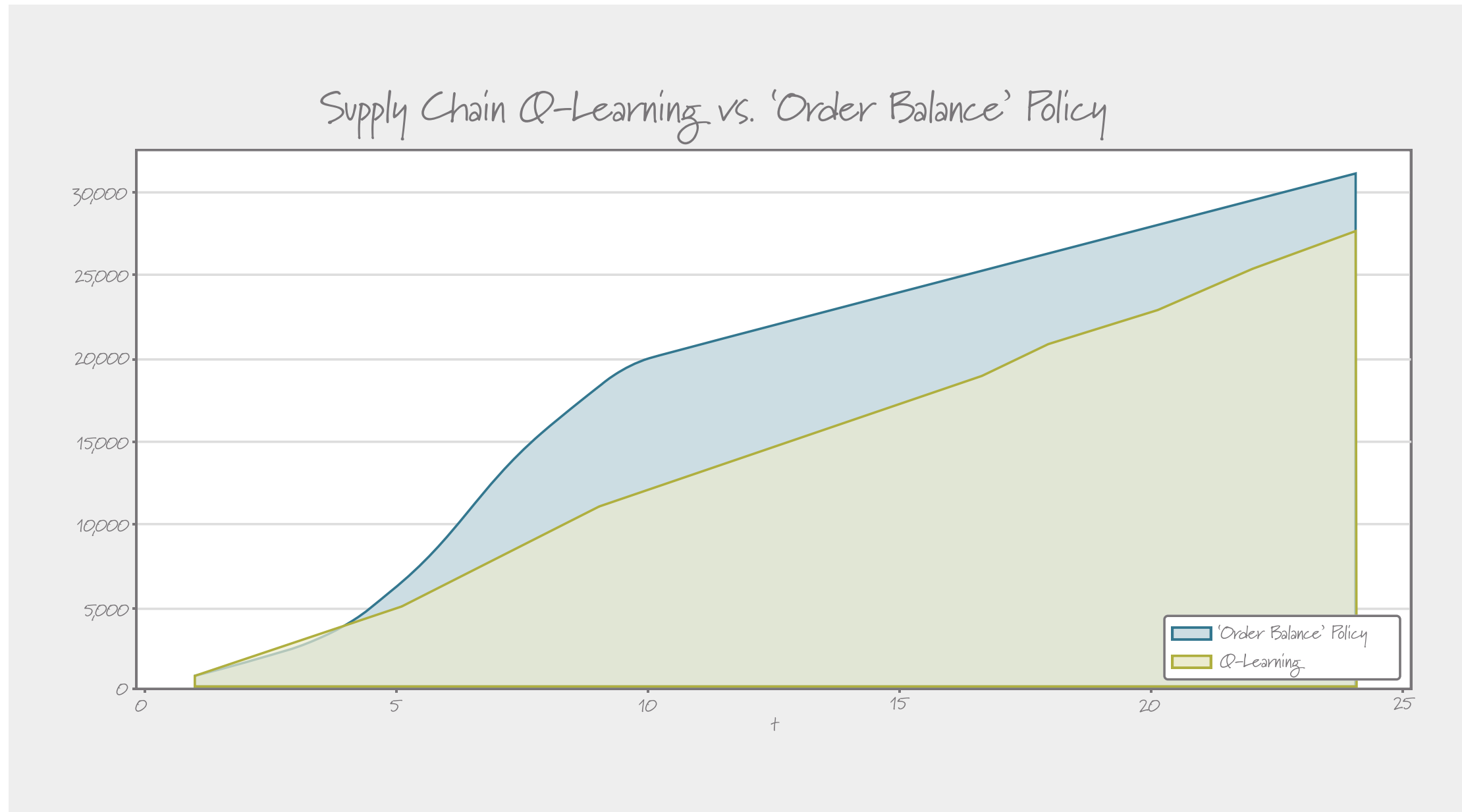
37,500 Episodes  
(almost there)



50,000 Episodes  
The AI agents can now  
play the game

# The Winner is...

The agents outperform our initial ordering strategy!



But: the agents are optimized towards the concrete game situation - they would fail in a more dynamic setting. The ordering strategy we developed initially is **robust in all ordering situations**.

# Summary

Reinforcement learning algorithms are quite easy to implement.

Finding the right reward policies is difficult: it is hard to avoid setting rewards too narrow (much like in real life).

BUT: If you can define clear objectives and rewards, using reinforcement learning in combination with simulations can be very useful for automating control systems.

# Oliver Grasl

Oliver Grasl co-founded transentis in 1997 and has been managing partner ever since.

After reading mathematics and theoretical physics at Cambridge University and the University of Innsbruck, and a brief foray into software development, he later specialized in business engineering at the University of St. Gallen (Executive MBA, Dr. oec.).

Oliver's personal mission is to help enterprises transform: explore their current situation, re-design their business model, organisation and IT-landscape and master the complexity of the actual transformation.



[oliver.grasl@transentis.com](mailto:oliver.grasl@transentis.com)

+49 173 6546727

+49 30 800937050





**We transform enterprises.**

Intelligently and systematically.

Since 1997.

We help our clients to explore, re-design and transform entire business ecosystems, using our expertise in transformation management, enterprise architecture and enterprise analytics.

**transentis consulting**  
**GmbH & Co. KG**  
Geisbergerstraße 9  
D-10177 Berlin  
[www.transentis.com](http://www.transentis.com)  
+49 30 9203833320