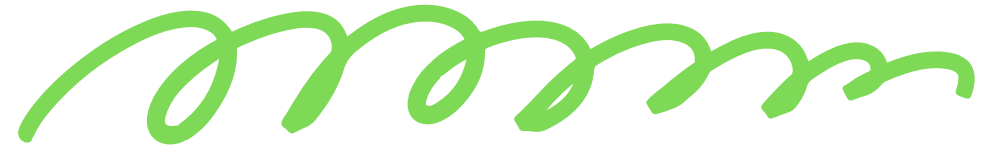


Game Theory for Products: Trust, Signals, and Escalations



Prepared by Kristina Egorova

Players



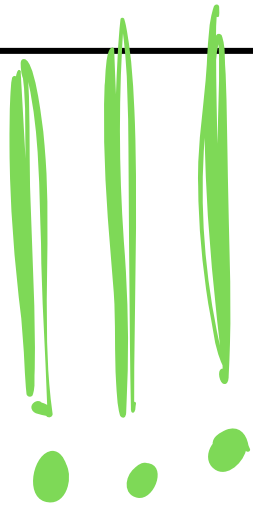
- **Product Owner (PO)** – Delivery accountability, balance scope, avoid escalation.
- **Lead QA** – Ensure quality, prefers thorough testing over speed.
- **Lead Dev** – Responsible for implementation, values interesting technical work.
- **Manager (M)** – Oversees multiple teams, prefers self-organization, avoids intervention.



Player Types (Incomplete Information)

Each player has a type:
it is hidden from others, but affects decision-making.

Player	Compliant (C)	Stubborn (S)
Product Owner	Willing to negotiate and reduce scope	Prefers to escalate and defend full scope
QA Lead	Open to time-bound compromises	Strict quality gatekeeper
Dev Lead	Follows sprint goals	Pushes for tech priorities




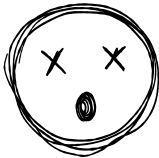




Assume:

- $Probability(Type = C) = 0.7$
- $Probability(Type = S) = 0.3$

These are common priors among players.

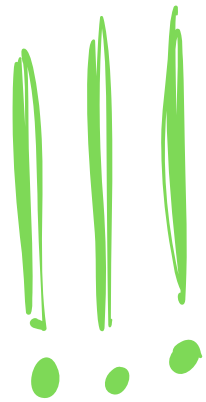
Possible Strategies

Each player can choose between 2 strategies.

Player	Strategy 1	Strategy 2
Product Owner	Negotiate (N) 	Escalate (E) 
QA Lead	Time-bound testing (T) 	Quality-first (Q) 
Dev Lead	Sprint Goals (D) 	Technical tasks (T) 

- Strategy is the **actual choice** a player makes during the game.
- Others **observe the strategy**, and **infer the type** over time.

Strategy vs Type



Product Owner (PO)

If Type **Compliant (C)** → Prefers collaboration
then likely strategy: **Negotiate (N)**

Else Type **Stubborn (S)** → Prioritizes delivery over harmony
then likely strategy: **Escalate (E)**

Lead Dev

If Type **Compliant (C)** → Delivers planned work
then likely strategy: **Deliver sprint goals (D)**

Else Type **Stubborn (S)** → Prioritizes tech over scope
then likely strategy: **Tinker / push refactoring (T)**






How does Game mechanics work?

1. PO observes that Dev deprioritized sprint work for a refactor (strategy = T).
2. Based on this, PO now updates their belief that Dev is type = Stubborn (S).
3. Next time, PO may prepare to escalate or reduce reliance on Dev's commitment.

Payoff Matrix (Example when PO = Compliant)

PO	QA	Dev	PO	QA	Dev	M	Trust
N	T	D	8	7	7	9	✔ High
N	Q	D	6	9	6	9	✔ High
N	T	T	5	6	9	8	⚠ Medium
N	Q	T	4	9	8	6	⚠ Medium

NQT: What happens?

- PO tries to negotiate 
- QA insists on full testing 
- Dev prioritizes architecture 

Likely Outcome:

- PO: frustrated — no agreement, scope at risk → score ≈ 4
- QA: satisfied — held the line → score ≈ 9
- Dev: happy — pursued interesting tech → score ≈ 8
- Manager: sees lack of delivery or alignment → score ≈ 5

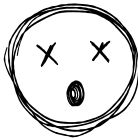
Trust: medium or low, depending on tone → ⚠ or 

Payoff Matrix (Example when PO = Stubborn)

PO	QA	Dev	PO	QA	Dev	M	Trust
E	Q	D	4	9	5	5	Low
E	Q	T	2	9	6	4	Very low
E	T	T	3	6	8	5	Low
E	T	D	3	6	8	5	Low

EQT: What happens?

- PO escalates to enforce full scope delivery.
- QA refuses to compromise and blocks unfinished features.
- Dev prioritizes tech debt over sprint goals.



Likely Outcome:

- PO: forced to escalate, fails to align the team → score ≈ 2
- QA: satisfied they maintained quality → score ≈ 9
- Dev: happy they got to work on interesting tech → score ≈ 6
- Manager: sees misalignment and tension → score ≈ 4

Trust: collapsed → ~~Very low~~

⚠️ Your Actions Are Signals

Every decision — especially escalation — reveals your type. Your team responds to what you signal, not what you say.

✅ Observe Behavior to Update Beliefs

QA and Dev may appear resistant or passive — but it's not always intentional. Like in a Bayesian game, their "type" is hidden. Watch. Learn. Adjust.

❌ Escalation Is a Trust Event

Even if you get delivery, escalation lowers trust. Over time, this changes how others act — and how your manager sees your leadership.



✅ Trust Multiplies Delivery

In high-trust teams, even partial alignment leads to progress. In low-trust teams, even clear goals cause friction. Trust is part of your product velocity.



✅ Incentives Create Types

You're not just managing personalities — you're navigating incentives. Change those, and the "types" change too.