Question Space System (QSS)

Viktor Jevdokimov, Vilnius, Lithuania

Table of contents

1. INTRODUCTION & QUICK START

1.1 Index

1.2 010 how to use this system

2. THEORY

2.1 Vision, Principles, and Beliefs

> **QSS (Question Space System)** is a way to *design the environment in which questions live*, not just the questions themselves.
> It treats inquiry as architecture: conditions first, then functions.

2.1.1 Purpose of QSS

The Question Space System exists to answer a simple but demanding need:

> **How can we consistently create conversations and reflections that lead to real clarity, alignment, and better consequences for humans – not just more information?**

QSS is:

- A **mental model** for what a "good" question space is.
- A **design discipline** for building such spaces in different domains.
- A **shared language** for talking about questions as structures, not isolated lines.

It does **not** prescribe one "correct" way to ask questions.
Instead, it offers a way to shape:

- **Orientation** – why we're asking.
- **Topology** – which dimensions of inquiry we activate.
- **Flow** – how we move through them.
- **Recursion** – how the space corrects and evolves itself.

2.1.2 Vision

Core Vision

> **Question spaces are aesthetic, friction-reducing environments for thinking together.**

The vision of QSS:

1. **Conversations and reflections feel clear, spacious, and safe**, even when topics are complex or emotionally loaded.
2. **Misalignment and hidden consequences become visible early**, before they turn into conflict, waste, or regret.
3. **Different perspectives can coexist and interact**, without collapsing into "who is right" battles.
4. **Inquiry becomes reusable**: once a solid question space exists for a type of situation, it can be adapted and replayed in future contexts.

In other words:

> QSS aims to make *high-quality thinking* and *low-friction collaboration* more repeatable, in any domain where humans need to understand and decide together.

2.1.3 Principles

These principles define how question spaces should be conceived, designed, and used.

Intention Before Inquiry

- Every question space starts with **clear intent**:
- What are we trying to understand or change?

- For whom does this matter?
- What kind of consequences are we actually trying to influence?
- The system prioritizes **purposeful curiosity** over aimless questioning.
- If intention is fuzzy, the first task is to clarify it, not to “collect more data.”

Implication: before designing questions, design the *why*.

Conditions Before Functions

- Borrowing from systemic thinking: **conditions enable functions**.
- Good outcomes (alignment, insight, better decisions) cannot be forced; they emerge when:
- People feel safe enough to speak.
- The problem is framed clearly enough to think about.
- The space is structured enough to avoid chaos, but loose enough to allow discovery.
- Question spaces therefore focus on **setting the right conditions**, not scripting specific answers.

Implication: design the space so that the right functions *have room* to appear.

Spaces, Not Lists

- A **question list** is linear. A **question space** is structured and multi-dimensional.
- QSS treats questions as elements in a **topology**:
- Clarification
- Assumptions
- Boundaries
- Value & Impact
- Contrast & Alternatives
- Causality & Consequence
- Layering / Abstraction
- Temporal
- Meta / Reflexive
- Different situations require different mixes of these dimensions.

Implication: we design **landscapes of inquiry**, not scripts.

Friction → Curiosity, Not Conflict

- Many problems in teams, partnerships, and personal choices are not “technical failures” but **communication and meaning failures**.
- Question spaces are built to:
- Transform **blame** into shared understanding.
- Transform **defensiveness** into safe exploration.
- Transform **anxiety** into clearer options.
- The preferred move is always:

> From “Who is wrong?” → to “What are we each seeing, and where do the views diverge?”

Implication: if a question space increases interpersonal friction, it is mis-designed.

Consequence-Aware Questioning

- Not all questions are equal. Some:
- Surface crucial risks.
- Reveal misaligned expectations.
- Expose unintended harms.
- QSS gives priority to questions that:
- Make **consequences visible** (for people, systems, and time).
- Illuminate **trade-offs** instead of hiding them behind abstractions.
- "Interesting but inconsequential" inquiry is treated as optional, not core.

Implication: question spaces are evaluated by their impact on *real-world outcomes*, not by intellectual elegance alone.

Multi-Perspective by Design

- Any meaningful situation (project, relationship, creative work) has multiple valid perspectives.
- Question spaces should:
- Make it easy to **name the perspectives** in the system (e.g., client/vendor/user/team/individual).
- Avoid collapsing everything into a single narrative too early.
- Hold tensions between perspectives long enough for useful integration.
- QSS encourages patterns like:
- "How does this look from X's perspective?"
- "What would Y describe as success or failure here?"

Implication: a good question space **respects plural viewpoints** and uses questions to map them, not erase them.

Evolving, Not Static

- Question spaces are **living artifacts**:
- They change as more is learned.
- They can be versioned, retired, or refactored.
- QSS assumes:
- No initial design is final.
- Recurring use reveals missing dimensions and unnecessary complexity.
- Recursion (meta-questions) is a first-class principle:
- "What did this question space miss?"
- "Which dimension did we overuse or neglect?"
- "What became clearer / more tangled after going through it?"

Implication: the system bakes in *self-correction* as a normal part of practice.

Minimal Sufficient Structure

- Over-structured spaces suffocate discovery. Under-structured spaces dissolve into noise.
- QSS aims for **minimal sufficient scaffolding**:
- Enough shape to avoid getting lost.
- Enough openness to allow unexpected insights.
- Preference is always for:
- Clear, simple flows.

- Small, composable building blocks that can be reused.

Implication: if a question space feels heavy or bureaucratic, it should be simplified.

Humane Pace and Depth

- People have limited cognitive and emotional bandwidth.
- Question spaces should:
- Move at a **humane pace**.
- Offer “shallow entry, deep continuation”: quick value first, depth available if needed.
- The system respects that:
- Not every situation requires full exploration.
- Sometimes the best next step is “enough clarity to act,” not exhaustive mapping.

Implication: the design should make it easy to stop at “good enough” without guilt.

2.1.4 Beliefs

These beliefs are not enforced as “truths,” but as **explicit assumptions** behind QSS.
They explain *why* the system is built the way it is.

1. **Most failures are failures of shared meaning, not raw intelligence.**
   Smart people routinely misalign because they never shared the same problem framing or consequence map.
2. **Human impact matters more than conceptual elegance.**
   A beautiful model that does not reduce friction, harm, or confusion is a decorative artifact, not a useful one.
3. **Questions are interventions.**
   Questions do not merely extract information; they change how people see themselves, each other, and the situation.
4. **Clarity is a social resource.**
   When clarity increases in a system, people coordinate better, trust more, and waste less.
5. **Good question spaces are teachable and reusable.**
   While intuition is valuable, relying solely on “talent for asking good questions” is fragile. Structures help others reach similar quality more reliably.
6. **No framework is universal.**
   QSS is meant to be adapted and integrated with other systems (e.g., collaboration frameworks, delivery frameworks, personal reflection practices), not worshipped as a single source of truth.
7. **Exploration is a renewable source of energy.**
   When friction drops and consequences are better understood, curiosity returns. That curiosity is a key driver for growth, creativity, and better decisions.

2.1.5 How to Use This Document

Use this file as:

- A **north star** for evaluating any question space you design:
- Does it reflect these principles?
- Which beliefs is it implicitly assuming?
- A **reference** when:
- Extending QSS into new domains.
- Creating new templates or GPT instructions.
- Explaining the system to collaborators.

Subsequent chapters (Core Architecture, Dimensions, Practical Construction) turn these Vision, Principles, and Beliefs into concrete tools and methods.

2.2 110 what is a question space

2.3 120 core architecture orientation topology flow recursion

2.4 130 dimensions of inquiry

2.5 140 modes and lifecycle

2.6 150 patterns and anti patterns

3. PRACTICE

3.1 200 practical overview

3.2 210 quickstart builder

3.3 220 context intake and orientation

3.4 230 designing the topology dimensions

3.5 240 designing flow and pathways

3.6 250 recursion and iteration loops

3.7 260 domain playbooks software delivery consulting

3.8 261 domain playbook personal reflection

3.9 262 domain playbook music production

4. REFERENCE

4.1 300 glossary

4.2 310 question pattern cheatsheets

4.3 320 templates and checklists

4.4 330 example question spaces case studies

4.5 Version and Licensing

This section documents the current version, license, and attribution principles for the **Question Space System (QSS)**.

4.5.1 Version Information

Attribute	Description
System Name	Question Space System (HCS)
Version	**V1.0**
Status	WIP — work in progress.
Release Date	November 2025
Maintained by	3in3.dev
Repository	GitHub – vitar/qss

Version 1.0 Summary

Version 1.0 consolidates the **foundational architecture** of the QSS.

4.5.2 Licensing

The **Question Space System** and all related documentation are licensed under the:

Creative Commons Attribution 4.0 International (CC BY 4.0)

You are free to:

- **Share** — copy and redistribute the material in any medium or format.
- **Adapt** — remix, transform, and build upon the material for any purpose, including commercial use.

Under the following terms:

- **Attribution** — You must give appropriate credit, provide a link to the license, and indicate if changes were made.
- **No additional restrictions** — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

License Reference

Creative Commons Attribution 4.0 International (CC BY 4.0)

To attribute this work, please use the following reference:

> *Question Space System (HCS) by 3in3.dev — licensed under CC BY 4.0 via GitHub repository vitar/hcs.*

4.5.3 Versioning Policy

- **Major versions (V2, V3, …)** introduce new theoretical constructs or expanded diagnostic models.
- **Minor revisions (e.g., V2.1)** include refinements, clarifications, or terminology alignment with derivative frameworks.
- All published versions will remain **permanently available** for reference and citation.
- Future releases will aim to maintain **backward compatibility** with the foundational definitions, rules, and models of HCS.

4.5.4 Attribution Guidelines

If reusing or adapting QSS content:

1. Include a visible credit line referencing *3in3.dev* and the license type.
2. Retain section numbering and core definitions where possible to preserve structural consistency.
3. When combining QSS content with other frameworks or methods, clearly separate attribution and derived materials.
4. For translations or derivative works, add a note identifying the adaptation (e.g., “Adapted from the original Human Cooperation System V1.0 documentation licensed under CC BY 4.0”).

4.6 About the Author

Viktor Jevdokimov, Vilnius, Lithuania — Creator of 3in3.dev, HCS, and 3SF

Viktor Jevdokimov is a software engineering leader, systems thinker, and framework designer with over 30 years of experience in software product delivery, modernization, and team alignment.

He is the creator of the **Human Cooperation System (HCS)** and the **3-in-3 SDLC Framework (3SF)**, and founder of the **3in3.dev** initiative — an independent platform dedicated to advancing collaboration and alignment between **Client**, **Vendor**, and **Product** ecosystems.

Professional Background

- Began career supporting distributed banking software on DOS and Windows, developing a deep appreciation for troubleshooting and system design.
- Progressed through roles of **developer**, **architect**, **delivery lead**, and **practice lead**, working with international clients on modernization and cloud migration initiatives.
- Specializes in **Client–Vendor relationship design**, **project leadership**, and **delivery system diagnostics**.
- Advocates for *"Context before Method"* and *"Trust before Control"* as guiding principles of effective collaboration.

Creative and Personal Work

Beyond software, Viktor is an **active musician and live sound engineer**, performing and mixing with the *Great Things* cover band.
He approaches both sound and systems with the same mindset: striving for **clarity, balance, and authenticity**.

About 3in3.dev

3in3.dev is an independent research and publishing initiative founded by Viktor Jevdokimov.
It consolidates his experience and experimentation into open frameworks that help organizations improve how they **engage, deliver, and measure value** across collaborative ecosystems.

3in3.dev publishes:

- The **Human Cooperation System (HCS)** — theoretical foundation for cooperative system design.
- The **3-in-3 SDLC Framework (3SF)** — practical application of HCS principles in software delivery.
- Supporting tools, templates, and learning materials under an open license.

> "These systems aren't about control — they're about clarity, trust, and the shared intent that makes collaboration work."
> — Viktor J., Creator of 3in3.dev