OFFICIAL GUIDE

A way of working for AI-Native teams

v1.0

By Flink X · May 2026 · Buenos Aires

About this guide

FX emerges from the need to **systematize a way of working that is already happening**. To do so we brought together **engineers, product managers, agile practitioners, AI engineers, trainers** and other roles, with the goal of agreeing on what is happening today and proposing a set of best practices for AI-Native teams.

We do not expect to reach a definitive, static version. AI-native practice is evolving too fast to immortalize a final guide. We expect to keep receiving feedback from many teams and practitioners experimenting with and building knowledge on **AI-first ways of working** — and to keep the framework current together.

In a nutshell

- **What it is:** a framework for teams where **carbon** people (humans) and **silicon** people (AI agents) work side by side as members of the same squad.
- **Who it's for:** born with a focus on teams building **digital products** that already use or plan to use agents as a substantial part of their workflow.
- **Where it starts:** it takes traditional agile frameworks as the starting point and extends them where AI-native logic demands new workflows and decision-making.
- **What changes:** it introduces changes in elements such as **roles, ceremonies and artifacts** that until now were standard in many teams and organizations.
- **What it keeps:** many of the values and principles remain in force: **empiricism, iterative work, and the conviction that we need to build teams, not just work groups**, among others. FX changes the operating system, not the philosophy.

Contents

00 · Prologue: why FX Framework

A COGNITIVE REVOLUTION

- **Until now, frameworks were relatively stable; context was volatile.** FX starts from the premise that the framework is now at least as volatile as the context itself.
- **In the industrial revolution, humans stopped executing tasks and started auditing the machines.** Something similar is happening now, with one essential difference: for the first time we delegate **cognitive** actions, not only physical ones. We are facing a **cognitive revolution**.
- **The effort to produce code drops significantly.** In parallel, decision-making becomes easier: while refining, we can **prototype with almost the same effort**. The decision process has always been key; now it is amplified.
- **The importance of the quality process goes up.** For years we hoped quality would stop being a separate stage with proposals like TDD; now this process becomes a **central asset** of the team.
- **The human-auditor becomes a key task.** It is not a new role: it is a central feature of AI-native teams.

CONTEXT

The agile frameworks we know were designed under an implicit assumption: work is done by people. The whole machinery of roles, ceremonies and artifacts assumes that behind every task there is a human who understands, decides and executes.

That assumption no longer holds. **A significant share of the work is executed by agents.** Agents generate code, write tests, draft tickets, analyze logs, handle customers, and more.

This is not "rolling out a new tool". It is a **disruptive change in the system**. It changes **who** does the work, **how** it is coordinated, **what** it means for something to be done, and forces us to revisit assumptions such as **value and productivity**, among others.

01 · Foundational definitions

1.1 AI-Native team

An AI-Native team is one where agents are **declared members** — with identity, role and context — and not tools invoked ad hoc.

1.2 Carbon people and silicon people

FX uses an operational distinction, not a metaphysical one:

- **Carbon people:** the humans in the squad. They have judgment, implicit context, legal responsibility and the ability to say "this is not delegated". **Accountability for the squad's work is always and exclusively on carbon.**
- **Silicon people:** the agents the squad governs. They have a declared identity, an assigned role and curated context. Each agent has a **human owner** who is accountable for its performance.

Our goal is for **humans to settle into the role of auditor** without transferring accountability.

1.3 Prompt-first workflow

In FX, the user story remains the minimum unit of work. What changes is the flow around it — the **prompt-first workflow**: a workflow that declares which agents will execute the story, which prompts they will receive, what context they consume, where a human steps in to validate, and how every step is logged for later audit.

We call it **prompt-first** because the prompt becomes a **central artifact across the whole process**: versioned, reviewed, reusable, assigned to an agent with a declared identity. If code was the team's central asset for twenty years, in FX that role is now shared among **code, prompts, curated contexts and checkpoint definitions**.

A concrete example

Let's take this user story as an example: **"As a user I want to filter transactions by date range"**. Here is how it gets executed in a traditional team vs. an AI-native team. These are illustrative, not prescriptive examples.

In a traditional team: the PO drafts the story and the team refines it. A dev picks up the story at the start of the sprint, writes the backend filter, builds the UI component on the frontend, writes tests, opens a PR, waits for code review, adjusts, merges. Time measured in story points; quality measured by post-merge bug rate.

In an AI-native team, the same work is designed as a prompt-first workflow:

- **Step 0 — Product Engineer (with help from a Story Agent):** drafts the initial story from the business need. The agent suggests use cases, tentative acceptance criteria and visible dependencies. Output: a story written and validated by the PE.
- **Step 1 — Spec Agent (silicon):** takes the story, reads the transactions module context in the repo, proposes an API contract, edge cases and verifiable acceptance criteria. Output: spec in markdown.
- **Human checkpoint A — Architect:** reviews the spec in 5 minutes. Approves, adjusts or reassigns. If rejected, the spec returns to the backlog with feedback.
- **Step 2 — Test Agent (silicon):** converts the approved spec into a set of tests (unit and integration) covering edge cases. Tests fail (red) because the implementation does not exist yet.
- **Step 3 — Implementation Agent (silicon):** generates the code that passes the tests. Iterates until green.
- **Human checkpoint B — Engineer:** reviews code and tests. Validates that the code does not introduce regressions elsewhere in the module, that the spec is respected, and that the tests actually cover the acceptance criteria. Approves or requests changes.
- **Step 4 — UX Agent (silicon):** generates the filter component following the design system. Output: frontend PR.
- **Human checkpoint C — Designer / PM:** validates UX and approves the merge.

What changes vs. the traditional version: the dev does not write the feature's code, they write **the specification the agents consume**. Productivity stops being measured by typing speed and starts being measured by the quality of prompts and checkpoints. Auditing no longer happens at the commit level ("who wrote this line?") but at the flow level ("which agent, with what prompt, against what context, approved by whom?").

Anatomy of a prompt-first workflow

A well-designed prompt-first workflow defines:

- **Participating agents:** each one identified by its Agent Roster card (name, role, model, owner).
- **Prompts in use:** versioned references in the Prompt Library, not improvised chats.
- **Context each agent consumes:** which files, which documents, which portions of the repo. Curated context is the team's primary asset.
- **Human checkpoints:** where an accountable human reviews, what they decide at that point, and what happens if they reject.
- **Logging and audit:** what is preserved at every step to reconstruct the decision later (prompt, context, output, identity of the human reviewer, timestamp).

02 · Roles

2.1 Product Engineer (PE)

Keeps the responsibilities of a product manager. The PE is accountable for the product and the value it delivers to both users and the business. At the same time, the Product Engineer is **semi-technical**: can read specs, understand prompts, evaluate technical outputs and prototype. Not a senior engineer, but technical enough to participate in designing prompt-first workflows.

Expanded responsibilities in FX:

- Validate that prompt-first workflows deliver business value, not just technical efficiency.
- Decide, together with the squad, which parts of the flow are delegated to agents and which remain human.
- Actively participate in the design of prompt-first workflows, not only in final validation.
- Accept the increment only when it meets the expanded Definition of Done (§05).

2.2 FX Coach (FXC)

Keeps the responsibilities of a Scrum Master or Team Coach. At the same time, they support both the team and the organization in the **AI-first transformation**. Inwards into the squad —coaching, facilitating, sustaining the framework—, outwards —building the bridges with the organization so the squad can operate.

Responsibilities towards the squad:

- **Coach and facilitate.** Accompany the team in iterative work. Facilitate framework ceremonies. Remove impediments. Protect the squad from "AI anxiety": focus on outcomes, not on proving that AI is being used.
- **Sustain the coherence of the framework.** Keep the Trust Boundaries Matrix (§4.5) alive. Ensure traceability: versioned prompts, documented agent decisions, auditable context. Watch over the systemic health of the agent roster (without being its technical owner — that belongs to the AI Engineer).

Responsibilities towards the organization:

- **Navigate the organization.** Help the squad move between areas, committees, governance bodies and corporate guidelines. The squad builds; the FX Coach clears the path.

- **Generate agreements.** Between the squad and stakeholders, and with areas such as legal, compliance, risk or infrastructure. This is the role that unblocks when the team hits organizational policy.
- **Sustain the transformation lens.** Keep the AI-first adoption conversation alive beyond the squad. Support other squads and areas that come after. Customize and evolve how the framework is applied to each context, and bring learnings back to the global framework.

2.3 AI-Native Squad

Delivers product increments that meet the expanded Definition of Done, **governing an agent roster** that executes substantial parts of the work.

The squad — human composition

- **Engineers:** orchestrate agents instead of coding directly. They design workflows, review outputs, integrate systems. The core skill is no longer "knowing how to write code" but **knowing how to implement AI in processes**.
- **Quality Engineers (QEs):** review agent outputs with judgment. This can be a dedicated person or rotate within the squad.
- **Product Designer:** same mandate as in traditional agile frameworks, but participates in designing prompt-first workflows.

The agent roster — agents under governance

Agents are **not full members of the squad**. They are silicon people that the squad governs. Each active agent has an **identity card** in the **Agent Roster (§4.5)**:

- Name and declared role ("Reviewer-01", "TestGen-Backend", etc.).
- Context it consumes: datasets, documents, systems it can access.
- Underlying model and version.
- Human owner: the person accountable for its performance and signing off on its outputs.
- Current performance metrics.

Additional roles

Depending on the size of the organization and the moment of the team, these roles may participate in a squad part-time or full-time.

- **AI Engineer.** Technical owner of the silicon system. Designs prompt-first workflows, maintains the **Prompt Library (§4.4)** and is **owner of the Agent Roster (§4.3)**: updates identity cards, audits the consistency of agents against the Prompt Library, and proposes additions, removals and changes to the squad.
- **Software Architect.** Oversees the architecture of the code and systems the squad builds, aligned with **business strategy and roadmap**. In a team where most code is written by agents, this role is critical: agents optimize locally without seeing the big picture, and someone has to safeguard the architectural coherence of the product. Validates delegation decisions with structural impact. Many developers are migrating into this role — in AI-native teams, product architecture becomes one of the core skills.

03 · Events

All events from iterative frameworks remain in force (Sprint, Sprint Review, etc.). FX does not replace them. We only describe here the events where FX makes substantial changes or adds something new.

The trend is **smaller teams moving faster**. In that context, today's two-week iteration standard is too long. FX proposes **one-week iterations as the standard**. Event timeboxes below are calibrated for that cycle.

3.1 Sprint Planning + Agent Briefing

Timebox: half of the traditional timebox — **up to 1 hour for a 1-week sprint**. With prompt-first workflows declared in advance, part of the definition work shifts to Refinement (§3.4) and planning concentrates on confirming and starting. A dedicated block is added: the **Agent Briefing**.

Suggested structure:

1. Block 1 — Sprint Goal and item selection (as in traditional planning).
2. Block 2 — Human work plan (as in traditional planning).
3. Block 3 — Agent Briefing: for each item, which agents participate? Which prompt templates are used? Where are the human checkpoints? Which prompts need to be adapted?

3.2 Daily Sync

Timebox: 15 minutes. The daily **continues practically unchanged**. A short update is added about responses received from agents and whether they need adjustments to execute during the day.

3.3 Retrospective

Timebox: up to 1 hour weekly — can be bi-weekly in small teams. The retro keeps its focus on the **team and its interactions, relationships, communication and working agreements**. It is a human space: communication, alignment, what supported us, what slowed us down.

Concrete actions come out of the retro, with owner and deadline.

3.4 Refinement

In FX, refinement is a **formal mid-week checkpoint** — typically post-daily. It has two goals:

4. Review the upcoming Sprint Goal and the candidate items to enter.
5. Validate the agents required for those items: do we have what they need (context, prompts, owners)?

The output of refinement is the list of items **ready for Sprint Planning**, with their prompt-first workflows sketched out.

04 · Artifacts

Same case as events. We only describe here substantial changes or additions.

4.1 Sprint Backlog (extended)

With an extension on top of traditional use: each Sprint Backlog item includes, in addition to its description and acceptance criteria, its **associated workflow** — which agents participate, which prompts are used, where the human checkpoints are.

4.2 Expanded Definition of Done

Covered in detail in §05. The principle: in FX, the DoD adds criteria for agent outputs, versioned prompts, auditable decisions and quality metrics observed by the squad.

4.3 Agent Roster

A living list of the squad's active agents, with each one's identity card (§2.3). It is a living artifact: it is updated in Refinement (§3.4) or whenever the AI Engineer detects that an agent needs adjustment.

Every active agent must have a declared human owner and curated context.

An agent without an owner is an unowned agent. An agent without curated context is a blind automation. If at any point the list contains an agent that does not meet this, the FX Coach removes or reassigns it.

4.4 Prompt Library

A versioned repository of the squad's prompts and prompt-first workflows. Not an abandoned Notion: it lives inside the product repository, with a changelog and review process.

No production prompt lives only in a single person's head.

The Prompt Library is what makes the squad's work replicable and transferable. It is also the asset preserved when there is turnover.

4.5 Trust Boundaries Matrix

A matrix that defines, for each category of task, three levels of delegation:

- **Autonomous:** the agent decides and executes without per-instance supervision (aggregate auditing only).
- **Assisted:** the agent proposes, a human validates before executing.
- **Human-only:** the decision is never delegated to an agent.

No critical decision is executed without an accountable human.

The matrix is reviewed at least once a quarter. What is "assisted" today can move to "autonomous" when data supports it, and vice versa.

Responsibility is always human

Some decisions are **—for now— 100% human**, no matter how much models improve. The reason is simple: these decisions can have legal, contractual or strategic consequences that require an accountable human. Some examples:

- AI does not validate the final business domain.
- AI does not approve critical architecture.
- AI does not accept increments as "done".
- AI does not authorize production deployments.
- AI does not sign commitments with external clients.

05 · Expanded Definition of Done

The DoD is the shared contract between the squad and the business about what "done" means. In traditional agile frameworks it includes criteria for technical quality (tests, code review, deploy). In FX, the DoD expands with criteria specific to human-agent work.

What follows is an example and a suggestion, not something fixed. Each squad builds its own DoD collectively and **evolves it when needed**.

5.1 Suggested standard criteria

- **Agent outputs reviewed by a human in the squad.** Every output that reaches production has a human who approved it or validated it by sampling per the Trust Boundaries Matrix.
- **Prompts used are versioned in the Prompt Library.** If the item's output was generated with a new or modified prompt, that prompt is in the library with its changelog.
- **Agent decisions are auditable.** For every decision an agent made, one can reconstruct: prompt used, context consumed, model, output, owner.
- **Quality metrics observed.** Hallucination rate, human acceptance rate, cost per outcome — the squad inspects them sprint by sprint and sets its own thresholds.

06 · Glossary

Agent Roster — FX artifact. Living list of the squad's active agents with their identity card.

Carbon (person of) — Human member of the squad. Holds the squad's legal and judgment accountability.

Expanded DoD — Definition of Done with additional FX criteria: reviewed outputs, versioned prompts, auditable decisions, quality metrics observed.

Prompt-first workflow — Workflow declared around a story, where agents execute steps and humans validate at defined checkpoints. Replaces the informal flow around the story.

Human owner — The human accountable for an agent's performance. Curates its context and reviews its outputs.

Prompt Library — FX artifact. Versioned repository of the squad's prompts and workflows.

Silicon (person of) — Agent with declared identity, assigned role and curated context, governed by the human squad. Not a full member: it executes, it does not sign.

Trust Boundaries Matrix — FX artifact. Defines which decisions are autonomous, assisted or human-only.

07 · Closing and next iterations

STATUS OF THIS VERSION

This is **version v1.0** of the FX Framework. It covers the minimum structure: roles, events, artifacts, expanded DoD and glossary.

CURATED, NOT INVENTED

FX is part of a broader conversation. This guide draws from and extends ideas from: **Scrum**, **spec-driven development**, **human-agent relationship models**, the **literature on multi-agent systems** and **empirical research on agent identity**.

HOW TO CONTRIBUTE

FX Framework is an evolving system, shaped by conversations with practitioners around the world. If you are applying FX in your organization, your learnings feed the next version. Write to us at **info@flink-x.com**.

FX FRAMEWORK

A way of working for AI-Native teams

flink-x.com · info@flink-x.com