ENRISE

AGENTIC AGILE MANIFESTO

A NEW WAY OF WORKING

Agentic development without losing

quality

durability

maintainability



Rick van der Staaij
Lead software engineer, Enrise

October 2026

PROLOGUE

Building software changed forever

Building software changed forever with AI agents. Code that used to take weeks is written, tested and shipped in days. That speed is only worth something if you remain the one who steers: **you** decide where your product goes, the agents do the driving.

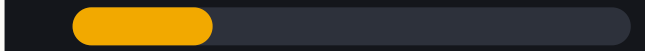
So we kept the agile way of working and gave it a new engine. We still build towards the desired product step by step, in small increments that are reviewed, tested and released one at a time. Every step is visible, and every step is yours to accept or send back.

The result surprised even us: time to market dropped by over **75%**, without giving up the quality, durability and maintainability we are known for. In this manifesto we tell you how we do it.

75%

shorter time to market

Time to market dropped by over 75% with this new workflow, while quality stayed guaranteed by automated tests and human gates.




Rick van der Staaij
Lead software engineer

Enrise
The digital partner you need

INDEX

What you will find in this manifesto

01 Prologue page 2
Why software development changed, and what we gained

02 Building blocks page 4
Automation, continuous deployment, human gates

03 The agentic base page 5
AGENTS.md, agent skills and MCP connections

04 Continuous deployment page 6
Review applications, staging and production

05 Workflow page 7
From backlog to done, step by step on the board

06 Full test coverage page 8
Unit, quality, functional and user tests on every environment

07 Project setup page 9
One mono repository that runs locally

08 Concluding page 10
The right steer and pedals

Dark pages open and close the manifesto. Light pages describe the way of working. Amber marks the Enrise accents.


Rick van der Staaij
Lead software engineer

Enrise
The digital partner you need

BUILDING BLOCKS

Three principles carry the whole workflow

01

Automation where possible

Every repeatable step is handled by agents and pipelines: writing stories, building features, running tests, deploying environments. Humans spend their time on decisions, not on chores.

Agents do the work. Pipelines verify the work.

02

Continuous deployment

A release per feature. Each merged feature flows through staging to production on its own, so there is no big-bang release, no release day and no queue of unshipped work.

Small releases are safe releases.

03

Human gates

People decide what is built, and when it is done. Priorities are set on the backlog, and every feature is checked by a developer before it reaches the primary codebase.

Steer at the start, approve at the end.

What agents know, can do, and can reach

KNOWLEDGE

AGENTS.md

One file holds all information about the project. It is the briefing every agent reads before touching code.

```
# AGENTS.md
## Project
All information about the
project
## Composing features
How new features should be
composed
## Quality
How quality is guaranteed
## Testing
How and what should be tested
```

SKILLS

Agent skills

Automated flows an agent can run on request.

Writing ticket assist
Turn a ticket title into a complete story

Discovering features
Derive features straight from the design

Picking up tickets
Build the story and open a merge request

Processing feature feedback
Apply review remarks to the feature

CONNECTIONS

MCP connections

What agents can talk to, through the Model Context Protocol.

The design
See what should be made

Project management tooling
Tickets, priorities, board columns

Code repository
Features, branches, merge requests

Kubernetes / server infra
See what is live and how it runs

CONTINUOUS DEPLOYMENT

Every feature travels the same road to production

STAGE 1

Review applications

`feature-123.review.example`

- Every feature gets its own test environment
- Spun up automatically with the merge request
- Functional assessment happens here, on the real thing

STAGE 2

Staging

`staging.example`

- Everything that is merged gets deployed to staging
- An environment where humans can play around with growing features
- Automated tests run against staging too

STAGE 3

Production

`www.example`

- All features automatically go live when the staging tests pass
- Guaranteed working and desired features
- Changes are made in no time with the regular workflow

Gate: human review
A developer approves the merge request after checking the review application.

Gate: automated tests
Unit, quality, functional and user tests must pass on staging.

Result: release per feature
No release day. Production moves forward one verified feature at a time.

WORKFLOW

From backlog to done on one board

BACKLOG 1

IMPLEMENTATION NOTE
Collection of ticket titles

IMPLEMENTATION NOTE
Create ticket titles by mass, based on a design

IMPLEMENTATION NOTE
Control priority: humans decide what comes first

REFINEMENT 2

IMPLEMENTATION NOTE
Agentic flows take a ticket from title to full story

IMPLEMENTATION NOTE
Analyses the code

IMPLEMENTATION NOTE
Analyses the design

IMPLEMENTATION NOTE
Analyses the infrastructure

IMPLEMENTATION NOTE
Writes a proper story including

- why
- acceptance criteria
- technical notes

READY FOR DEV 3

IMPLEMENTATION NOTE
Agents build the story

IMPLEMENTATION NOTE
Create a merge request with the code changes

IMPLEMENTATION NOTE
The merge request deploys a review application

READY FOR REVIEW 4

IMPLEMENTATION NOTE
Developer checks the code architecture

IMPLEMENTATION NOTE
Functional assessment of the new feature on the review application

DONE 5

IMPLEMENTATION NOTE
Feature is merged and added to the primary codebase

IMPLEMENTATION NOTE
Automatically deployed to staging and tested

IMPLEMENTATION NOTE
Automatically deployed to production if staging is OK

FULL TEST COVERAGE

Four layers of tests guard every change

Unit tests

Every function and class is verified in isolation. Agents write them alongside the code they produce.

Code quality checks

Static analysis, linting and architecture rules keep the codebase durable and maintainable.

Functional tests

Cover the business rules. The behaviour the product promises is asserted, not assumed.

User tests

Application happy flows and regression, clicked through like a real user would.

RUNS AGAINST

Three environments, same suite

Local environment

While building, agents run the tests on their own setup

Review applications

Every merge request is tested on its own environment

Staging environment

The final check before production goes live automatically

PROJECT SETUP

One mono repository

Everything that makes up the product lives together. An agent sees the whole picture in one checkout.

- **backend-api** the API every client talks to
- **frontend** the web application
- **mobile** iOS and Android apps
- **database** schema, migrations, seeds
- **external** links to the outside world
  - `cms` content management system
  - `e-commerce` shop integration
  - `crm` customer management system
  - `...` and whatever else the product needs
- **tests** unit, functional and user tests
- **docker-compose.yml** the whole stack, locally
- **AGENTS.md** the briefing for every agent

LOCALLY RUNNABLE

Build and test on your own machine

The whole stack starts with one command. Agents build and test against this local setup before anything is pushed.

```
$ docker compose up
… starting backend-api, frontend, database, cms-mock
✓ all services healthy
$ make test
✓ unit · quality · functional · user
```

- **Docker Compose** describes every service, so local equals review equals staging.
- **Agents build and test** with the local setup, catching problems before a merge request exists.
- **One checkout, one command.** Agents and humans work on the identical setup, so what passes locally passes in the pipeline.

CONCLUDING

Faster, without giving anything up

75%

The new way of working speeds up the process by about 75%

Without losing quality, durability or maintainability. The tests and the human gates make sure of that.

Speed with certainty

You need the right steer and pedals

Agents bring the speed. Priorities, acceptance criteria and reviews are the steering wheel and the brakes. Both are needed to arrive where you want to be.

Humans decide, agents deliver

Enrise has the workflow for the perfect end result

For everyone who wants the speed of agentic development and the certainty of a product that keeps working. This is how we build.

The digital partner you need

ENRISE

ABOUT ENRISE

The digital partner you need

A self-steering group of professionals who build digital products that last. We combine agentic development with the craft and judgement of experienced engineers.

enrise.com



AUTHOR

Rick van der Staaij

Lead web developer at Enrise

Rick leads the engineering of this workflow at Enrise: the agentic base, the pipeline and the gates that keep the quality in.

rick.nu