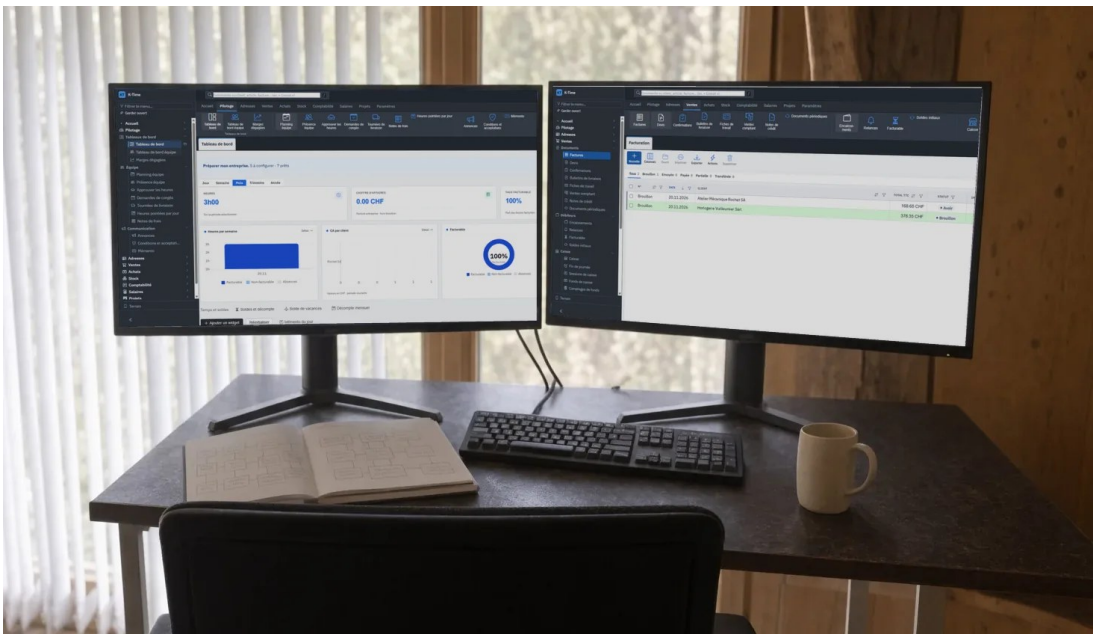
CHANTIER EN COURS

# Un ERP et sa fabrique

*Nous développons K-Time, un logiciel de gestion pour les PME suisses, avec une équipe d'agents d'IA. Le plus dur n'est pas de leur faire écrire du code, mais de garder la main et de prouver que ça marche.*

Publié le 26 novembre 2026 — Imprimé le 27 novembre 2026



We are developing K-Time, business management software for Swiss SMEs, with a team of artificial intelligence agents. The hardest part is not getting them to write code. It is keeping control of what they produce, and proving that it works.

## Why an ERP

Accounting, payroll, purchasing, sales, stock, projects, interventions, invoicing: an ERP makes trades coexist that each have their own rules. Touching stock can change the value of goods, therefore the invoice, therefore the accounts. The difficulty is not each module taken on its own, it is what happens between them.

That is why we chose it as a testing ground. If a way of working holds up on an ERP, it will hold up on many other kinds of software. Our practice with management software goes back to 1999; K-Time lets us both have the tool we want for our clients and put these methods to the test, at scale.

The software itself remains conventional: PHP, SQL and the web. Artificial intelligence comes in mainly in its manufacture.

## The factory

Giving a task to an agent is simple. Getting several to work at the same time, on the same software, for weeks, without losing the thread, is another trade. So we built what we call the Factory.

- The work is split into **batches**: a scope, a contract, acceptance criteria.
- Each agent works on its own branch, in an isolated environment, on one of several machines and with the most suitable model.
- A change passes automated checks, then integration checks and, if necessary, an independent evaluation.
- A single merge queue brings batches into the product.
- Each defect found becomes a new batch. The loop runs continuously.

## What the figures say

As of 26 November 2026, the Factory's register holds **3,254 batches** tracked, of which **2,862 merged** and **224 rejected**. The median time between a batch being launched and its arrival at the merge gate is **55 minutes**; for the 710 batches merged since 21 November, this time has come down to **40 minutes**.

A batch is often a small thing: a median of 26 lines of code and 232 lines of tests. The ratio between the two says the essentials of our method: we write far more to check than to do.

These figures measure neither the quality of the software nor an equivalent in developer hours. They show a pace and a granularity.

## The one who codes is not the one who judges

An agent readily announces "finished" when the work is only half done. In an ERP, a page that displays or a test that passes does not prove that it can be used.

We have therefore introduced **judges**, who play a business role: accountant, salesperson, buyer, administrator. They start if possible from a fresh installation, with known data, and without having read what the developer thinks of their work. The salesperson must draw up a quote, print it, find it again, duplicate it and turn it into an invoice. The accountant must post an entry, post an invoice and produce a VAT return.

What the judge finds becomes a correction batch; after merging, it starts again. The work has changed in nature: producing the code is quick, demonstrating that it meets the need takes all the attention.

## A factory that repairs itself

The Factory has become software in its own right, with its own failures. Many are conventional: files opened by two processes, locks, network outages, Git conflicts, queues, recovery after an incident. Others come from the agents: contracts that are too long, conclusions not checked enough, fixes done twice, a task given to the wrong model.

Each incident is used to correct the process. The next steps: states managed transactionally, less waiting, the real cost of each batch, and an allocation of tasks according to what each model can really do. Local models have their place in it: not everything needs a cutting-edge model.

## Experience makes the difference

None of this replaces skills that are older than AI: architecture, databases, continuous integration, system administration, knowledge of the trades, quality assurance. It is thanks to them that the right constraints are set and that a bad solution is recognised before it enters the product. Agents multiply the capacity to execute; they do not remove the need to understand what is being built.

K-Time is at version 0.9.1 RC73, in the qualification phase. Migration of existing data, accounting consistency, complete validated journeys: these steps remain indispensable before going into production. The Factory serves precisely to make this path measurable and reproducible, and it could serve other products than K-Time tomorrow.

K-Time screen by screen · The factory: a year of development with agents