

JDriven Tech Radar

Fall 2025

Our view on recent developments in our field

12th edition

**Commit to know.
Develop to grow.
Share to show.**





Commit. Develop. Share.

Opening Remarks

We proudly present the 12th edition of the JDriven Tech Radar.

As many will have noticed, two subjects have been dominating conferences and social media in our profession for the last year: AI and digital sovereignty in Europe. In such a situation, it's tempting to follow along with the sentiment and unquestionably copy or reject criticism. The way we bring the JDriven Tech Radar about protects us from this. First of all, we need to present convincing arguments to our colleagues about whether a subject and our opinion on it belong on the radar. Once that's been ascertained, we follow a style guide for writing the texts, where each blip always consists of an objective description followed by a subjective opinion. This distinction helps with taking various perspectives into consideration and separating fact and opinion. Once the texts have been written, they're treated the same way we treat any merge request: colleagues review each other's contributions. This way, we can follow trends and interpret them with nuance and substance.

I think we've succeeded at that once more.

Thank you to everyone who made this possible.



Jasper Bogers
Consultant JDriven



Introduction

Our experienced specialists participate every day in the development of many different software projects all over the Netherlands, and are involved in worldwide professional communities. Each half year our engineers gather to discuss the latest emerging trends and developments in software engineering. We seek to capture these trends in a technology radar. Each edition of the radar shows the changes in these trends, compared to the previous edition. A shift can indicate that we see a technology is becoming more, or less, relevant for certain use cases. If a trend does not show up in later editions, it signifies that there are no relevant developments or experiences that cause us to shift our assessment. With this document we will discuss the shifts we have observed over the past half year. This analysis guides us in deciding which technologies we use and recommend.

Tech Radar

The idea to create a Tech Radar was initiated by ThoughtWorks. They periodically showcase their view on new trends and development. In addition, [they advise everyone to define their own radar](https://www.thoughtworks.com/radar/byor) [https://www.thoughtworks.com/radar/byor].

At JDriven we fully support that view. Building a Tech Radar is an instructive and valuable experience, where we mutually share our knowledge and create a common awareness around technology. We believe that specialists should be able to create and maintain their own toolset to perform their job to their best ability. When composing a radar, you facilitate a broad discussion on technology, so organisations can strike the right balance between the risks and rewards of innovation. We can help you to kick-start these discussions in your organisation. Let your teams innovate and inspire each other. Together they can draw up a set of technologies and techniques that can accelerate your business.

Overview

The radar consists of quadrants and rings, containing blips to indicate technologies of interest.

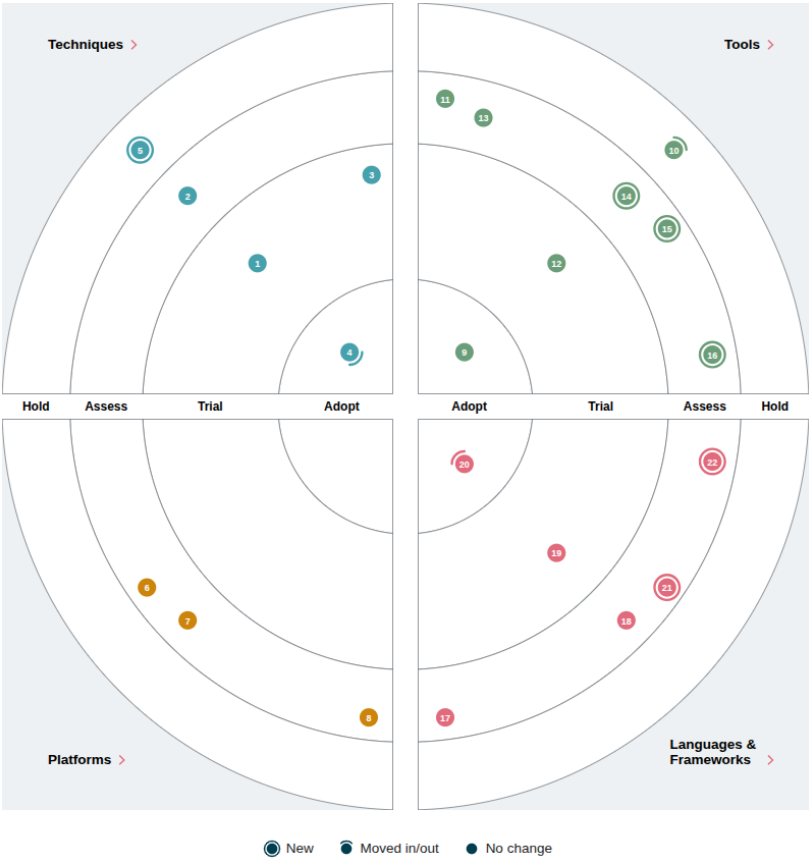
The quadrants subdivide the different subjects into categories:

- **Languages and frameworks** that support developers in their daily tasks.
- **Platforms** to deploy and execute programs.
- **Techniques** help developers create better software.
- **Tools** aid in the development and delivery process.

The rings in each quadrant indicate which phase of the adoption cycle we believe a technology is currently at:

- **Adopt:** We recommend to use this technology, wherever it fits the requirements.
- **Trial:** We advise to gain experience with this technology, wherever a project allows for a certain degree of risk.
- **Assess:** Interesting topic to learn more about and assess its future impact; yet too early to use in production.
- **Hold:** Do not deploy in a project not already using this technology.

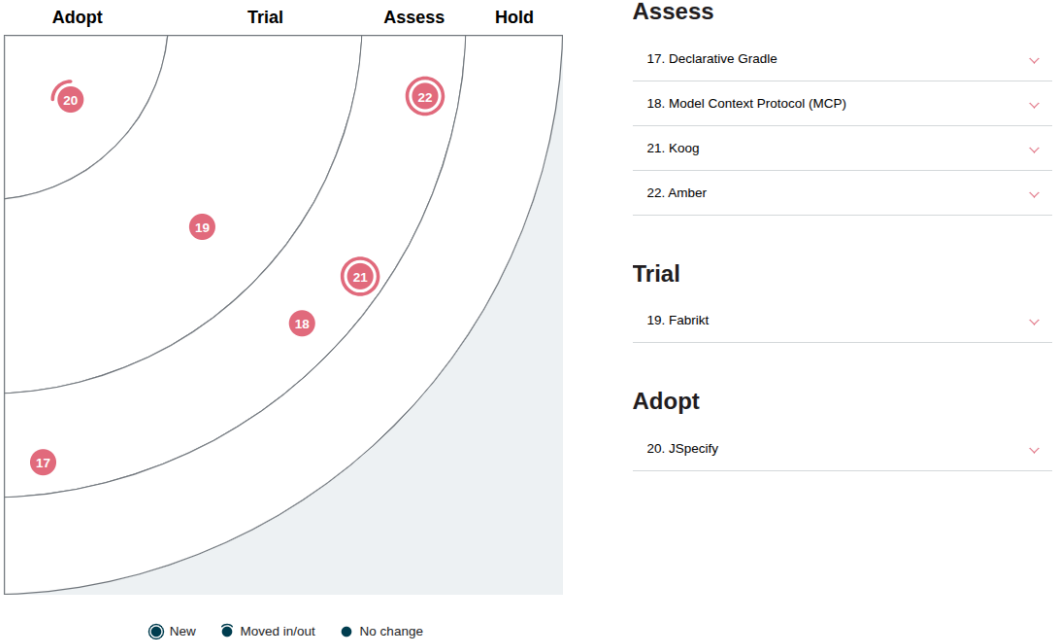
In the subsequent sections we will delve into our views on recent developments in the field of software engineering more closely.



Tools	Languages & frameworks
ADOPT IntelliJ HTTP client	ADOPT JSpecify
TRIAL Nushell	TRIAL Fabrikt
ASSESS Anubis BlueDolphin Cursor IDE DeepSeek Maven 4	ASSESS Amber Declarative Gradle Koog Model Context Protocol (MCP)
HOLD JetBrains AI	HOLD -
Platforms	Techniques
ADOPT -	ADOPT Event Modeling
TRIAL -	TRIAL Embrace Shadow IT Passkeys
ASSESS AWS European Sovereign Cloud EU Cloud alternatives Proxmox	ASSESS AI Agents
HOLD -	HOLD Vibe coding



Languages & frameworks



JSpecify

ADOPT

JSpecify [<https://jspecify.dev/>] is an annotation framework designed to improve nullability handling in Java by providing explicit type information, inspired by Kotlin’s nullability system. It helps prevent `NullPointerException`s by standardizing nullability contracts at compile-time. The project has been in development for over five years and is backed by major players such as Google, JetBrains, Meta, Microsoft, Oracle, and Uber. With the release of version 1.0, the specification has stabilized, providing a solid foundation for adoption. Currently, IntelliJ IDEA, Project Lombok, NullAway, and the Checker Framework already provide support for JSpecify.

Migration guides are available to transition from JSR-305 or Checker Framework annotations to their JSpecify variants. With Spring Framework 7 embracing JSpecify and comprehensive toolchain support now available, it has become the standard choice for improving Java’s null safety. JSpecify has proven itself as a stable, production-ready solution with excellent tooling support and wide acceptance in the Java ecosystem. We strongly recommend using JSpecify for new projects and considering it for existing codebases. Therefore, we place it in the Adopt ring.



Fabrikt

TRIAL

OpenAPI, formerly known as Swagger, is a popular interface specification for defining and documenting APIs. The best-known tool in use for generating code based on an OpenAPI specification is probably OpenAPITools' openapi-generator. Fabrikt is an alternative, that focuses purely on generating Kotlin code from an OpenAPI v3 specification. It supports Spring, Micronaut Ktor, plus it offers a Gradle plugin.

Contrary to openapi-generator, Fabrikt deals well with the more complex structures of OpenAPI v3, with *anyOf* being a well known example. Besides that, the smaller technical focus helps the team behind Fabrikt manage to keep up with open issues and merge requests, where openapi-generator lags behind and has built up a substantial backlog. These are reasons why we advise using Fabrikt for OpenAPI v3 based Kotlin code. Because Fabrikt can handle more complex specifications and can generate code different from openapi-generator, we recommend starting small and evaluating the impact of a migration.

Amber

ASSESS

Amber [<https://amber-lang.com/>] is a modern programming language that compiles to Bash shell scripts. It addresses the challenges of error-prone and hard-to-maintain shell scripts by adding compile-time type checking, modern ECMAScript-like syntax, and a comprehensive standard library. This makes shell scripts more reliable and prevents runtime errors. In 2024, the project won the award for "The Best Engineering Project in Technology" at Wrocław University. The language supports five data types (Text, Num, Bool, Null, and `Array`s), provides modern programming constructs such as functions, loops, and conditions, and integrates seamlessly with existing Bash scripts.

Amber is currently at version 0.4.0-alpha and still under active development. While the concept is promising for improving shell scripting, there are still limitations, such as the lack of nested array support due to Bash constraints, and the risk of breaking changes between versions. The language is worth watching for teams that rely heavily on shell scripts, but is not yet mature enough for production use. Therefore, we place it in the Assess ring.

Declarative Gradle

ASSESS

When using Gradle, there are many different ways to structure your build scripts. This can cause a lot of confusion, especially when working with multiple projects that have different approaches. Gradle is aware of this, and at the end of 2023, they announced that they had started an experiment to make Gradle declarative. This project is called **Declarative Gradle** [<https://declarative.gradle.org>] and is still in an experimental phase. The goal of this project is to provide a fixed structure for Gradle build scripts, making them easier to read and maintain. This should also improve consistency between projects.

We place this new standard in the Assess ring, because we find it worthwhile to follow developments in this area and see what impact it will have on Gradle build scripts.



Koog

ASSESS

With **Koog** [<https://docs.koog.ai/>], JetBrains has launched an open source framework that helps developers overcome various challenges regarding the creation of AI agents. Our definition of an AI agent here is: An independent application that accepts loosely defined input, sends that to an LLM for interpretation and then returns output that is principally non-deterministic. It's hard to build an application that makes one or more LLM calls in a way that is predictable, reusable and generally reliable. Koog offers various mechanisms for dealing with that, like creating workflows, memorizing conversations, compressing history, and consequently building up a knowledge base for the AI-agent to build on, rather than having to query the LLM every time. It also allows switching to a different LLM during a conversation, for example to only call a slower or more expensive LLM when a faster cheaper one yields insufficient answers. Koog is Kotlin-oriented and supports OpenTelemetry, Model Context Protocol (MCP), Spring Boot and Ktor.

JDriven sees in Koog a framework that might help mitigate the big risks of unpredictable outcomes and possible high costs of using an LLM. It does this up to a point where one may wonder if an application, that goes to such lengths to put safeguards around an AI, wouldn't be better built without AI entirely. But it's there, where developers are reluctant to use AI in their applications and are in search for a solution architecture where non-deterministic components might shine, that such safety is necessary. We think Koog can play a part there, although it will have to compete with LangChain4j and Spring AI.

Model Context Protocol (MCP)

ASSESS

Model Context Protocol (MCP) [<https://modelcontextprotocol.io/introduction>] is an open standard that enables applications to provide context to large language models (LLMs) in a consistent way. MCP has integrations with databases (PostgreSQL, SQLite), platforms, filesystems and many other systems. By reducing reliance on proprietary solutions, MCP improves interoperability with multiple LLM models. It also has the potential to help non-technical users query and understand their data more easily by allowing AI models to access relevant context dynamically.

However, adoption depends on industry support, and challenges remain in standardizing context delivery across diverse systems while addressing security and performance concerns. We are placing MCP in Assess as it shows a lot of potential, but still needs to be proven in practice.

Techniques

Trial

1. Passkeys

3. Embrace Shadow IT

Assess

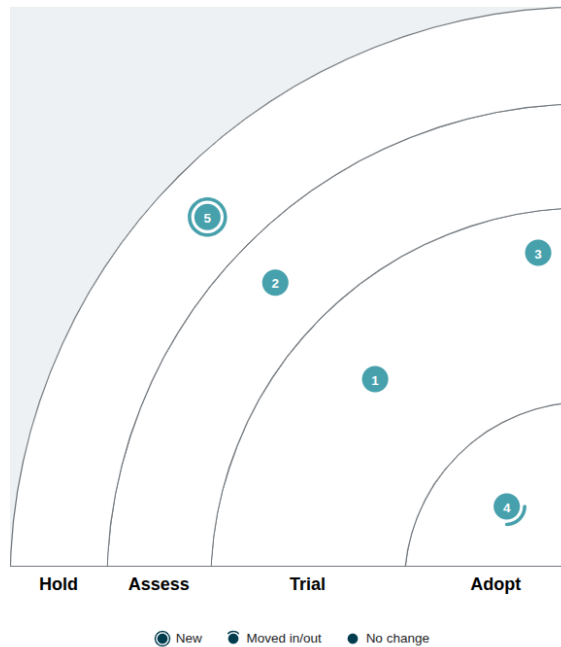
2. AI Agents

Adopt

4. Event Modeling

Hold

5. Vibe coding



Embrace Shadow IT

TRIAL

Shadow IT is a phenomenon where software developers within an organization build up IT facilities outside official supervision and regulation by the IT department responsible. This typically occurs when IT support fails to facilitate everything that developers need to deliver their software solutions. It may concern unauthorized use of network, storage, and compute resources for running applications, or for CI/CD tooling for the delivery of those applications. Sometimes it's self-built tooling that escapes audit trails, or third party software packages that are accessed via public internet. Shadow IT can lead to a scattered software landscape with blind spots. Required maintenance, security and compliance of crucial parts can remain unknown and unaddressed, which is why management generally regards shadow IT as a risk.

We see a trend where platform teams, born from earlier DevOps insights, are gradually being moved back to separate departments, governed by goals of predictable and stable infrastructure and CI/CD tooling, and a corresponding increase of shadow IT among software developers. JDriven encourages organizations to not excessively restrain shadow IT - where possible, which is why it's on *Trial* on the Tech radar - and to seek to bridge the gap. Developers need insight into the reasons why IT support puts limits on what it facilitates, and IT support should know the developer needs for the growing software landscape and the demands for CI/CD tooling to continue building software. DevOps was the answer to the friction in the software landscape between software developers' agile software delivery and operations' drive to keep that landscape stable and controllable. Similarly, the DevOps mentality should be to embrace shadow IT and let it lead to controlled innovation and sustained effectiveness of IT support.



Passkeys

TRIAL

While we still have some reservations whether passkeys are generally accepted by the public, we decided to put this technique in *Trial*. We consider passkeys as a technique ready to be offered as an option together with the traditional authentication techniques such as passwords. The idea behind passkeys is that they are secure through the "something you have" principle of the "authentication factors" (https://en.wikipedia.org/wiki/Authentication#Authentication_factors), because the private key doesn't leave the users' device. Compared to passwords, this prevents some vulnerabilities. Because of the human factor, passwords are susceptible to things such as low password strength, reuse, data breaches, social engineering and carelessness. Large IT players, such as Apple and Google have broad adoption for passkeys on their devices, increasing ease of use for the average user. This has made passkeys a realistic option as primary authentication.

If you are considering adoption of passkeys as a developer, take a look at Hanko.io's website that demonstrates the benefits of passkeys: <https://www.passkeys.io/>.

Of course, be diligent in defining the account recovery process if you want to adopt passkeys. Give users enough opportunity to alter their authentication through familiar processes.

AI Agents

ASSESS

LLMs provide the capability to interact with a computer in a language that is natural to humans. Being able to pose a question or define a task via a prompt written in natural language might be the biggest game changer since the internet. Yet, they don't interact with the real world but take an input and produce some output. Modern **AI Agents** [<https://www.ibm.com/think/topics/ai-agents>] are programs that use LLMs to autonomously reason which sequence of actions is necessary to complete its goal. They are not restricted to a particular script and can dynamically utilize the results of previous actions to select the subsequent action.

What would this look like in practice? Imagine a digital assistant for an e-commerce platform. With this agent, instead of having to fill in a form, the user can ask, "What is the status of my order?". The assistant can reason that it should retrieve the status by querying the order database. Moreover, the agent can also infer that it must first request the order id from the customer.

At the same time, AI agents are still in their early stages. While the underlying models are powerful, they can be prone to errors. Reliable deployment requires careful attention to robustness, security, and ethics. Still, the potential for practical applications is already significant enough to warrant active experimentation — which is why we're placing AI agents in the Assess ring.

Event Modeling

ADOPT

Event Modeling [<https://eventmodeling.org/>] is a method of describing systems using an example of how information has changed within them over time.

An event-centric approach to a business process and describing it in a clear manner helps business, architects, and developers to have a deep understanding about the problem to be solved. Besides an event-centric approach, Event Modeling also takes the user experience into account. This combination results in a complete picture of user interaction with the system and how information flows through the system.

At JDriven we have seen that Event Modeling is used alongside or instead of Big Picture Event Storming, to develop a common understanding of a business domain. It results in a more concrete picture for software implementations.

Vibe coding

HOLD

Vibe coding is a term that has recently gained traction in software development. It refers to a development methodology where programmers primarily rely on intuition and feeling when iteratively using AI tools, particularly large language models (LLMs). Instead of structured development processes, it revolves around repeatedly adjusting prompts and experimenting with AI-generated code until the result "feels right" or appears to function.

The practice is characterized by a cycle of prompt engineering, where developers refine instructions based on output, without necessarily having deep understanding of the underlying code, language, or frameworks. Proponents claim this enables faster prototyping, while critics point to risks around code quality, maintainability, and understanding of the software.

JDriven sides with the critics and places vibe coding on *Hold*. It involves a trial-and-error approach that goes against the practices of professional software development, where systematic work, clean code and quality assurance are central. This lack of systematicity leads to technical debt, security issues and poorly maintainable codebases. Fact is that software development involves a lot more than typing in code, to get to a solid result, and vibe coding offers no solution for that.

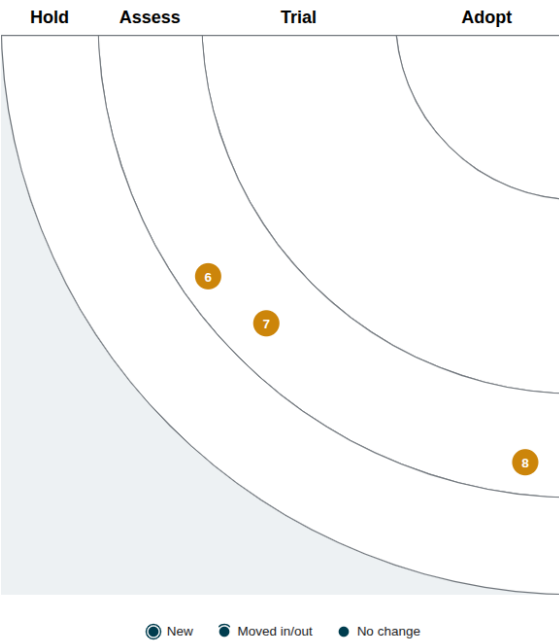
On the other hand, deliberate use of LLMs in a professional development environment *can* add value. As part of a structured development process, AI tools can effectively support developers with code review, documentation, generating boilerplate code, exploring design patterns and offering assistance with explaining technical error messages. The key difference lies in accountable, methodical integration versus the ad hoc experimentation that characterizes vibe coding.



Platforms

Assess

6. EU Cloud alternatives	▼
7. AWS European Sovereign Cloud	▼
8. Proxmox	▼



Dependency on US services

Via the General Data Protection Regulation (GDPR) the EU regulates how companies collect and process data related to EU citizens. However, the American FISA702 law and Executive Order 12.333 allow the US government to force any American company, or even an international company with an office in the US, to ignore the GDPR and allow it access to that data, even if the servers are not on US soil. An additional agreement called TADPF was put in place between the EU and the US to help safeguard data privacy, guarded in the US by the Privacy and Civil Liberties Oversight Board (PCLOB). The recent capricious American decisions surrounding the PCLOB are cause for concern among European companies that fear, that these safeguards for their data and services on American service providers may prove to be insufficient.

AWS European Sovereign Cloud

ASSESS

To address the concerns of European companies with regard to PCLOB and GDPR, AWS has announced [AWS European Sovereign Cloud](https://aws.amazon.com/compliance/europe-digital-sovereignty/) [https://aws.amazon.com/compliance/europe-digital-sovereignty/], which is an AWS Cloud located entirely in the EU, operated solely by European employees. The first location of the AWS European Sovereign Cloud is scheduled to be in Germany by the end of 2025. AWS offers an FAQ and whitepapers on its accompanying website to explain how they mean to comply with GDPR, and about the impact of FISA702, stressing that any lawful government request for data insight would be held to the highest scrutiny.

For any company seeking a way to remain compliant with GDPR without having to write off investments in AWS knowledge and implementation, JDriven recommends assessing whether the AWS European Sovereign Cloud is a safe alternative. It's important to note that the AWS European Sovereign Cloud service offering is considerably smaller than that of the regular AWS Cloud. Besides that, in their terms of use AWS mentions that they are bound by law to comply with legal requests for data insight, which raises the question if it makes any difference that AWS European Sovereign Cloud has European locations and employees, while the parent company is American, bound by American law. These are things that warrant a status of Assess on the radar.

EU Cloud alternatives

ASSESS

Over a decade ago, several American companies like Amazon, Google, and Microsoft decided to expand their server farms and rent them out to other companies. Infrastructure-as-a-service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service: "the cloud" was born. As the years progressed, cloud providers expanded and added "cloud native" services: for example, you no longer install a database on a rented server, but you rent a database service, which is maintained by the cloud provider. This has helped organizations with cost-efficient scalable infrastructure, and has also allowed them to focus on their unique core competitive capabilities. Software consultants however have been warning for vendor lock-in: cloud providers want to lure customers with their unique service offering, but using those unique services complicates efforts to depart from that cloud provider later.

In recent months it has become apparent that there's an additional aspect to vendor lock-in to consider. American cloud providers need to comply with a government that has been capricious or even hostile in their foreign relations, and has entered into a trade war with European countries. European organizations should therefore wonder, how vulnerable their software and data are when hosted at American cloud providers. Cloud-agnostic platforms like Kubernetes can help decouple the software architecture and underlying infrastructure, mitigating the risk to some extent. But JDriven thinks it is time organizations look at the various European cloud providers and consider to what extent the service offerings matches their software and infrastructure needs. Examples of European cloud providers are Cyso Cloud, OVH Cloud, Exoscale, Hetzner, Scaleway, T-Systems, and Proact. There is also the European Gaia-X initiative, which several cloud providers have joined. It offers interoperability and decentralized data security while using multiple clouds. An example of a European SaaS initiative is the German state Schleswig-Holstein, which is pursuing digital sovereignty via a migration to open source IT-infrastructure based on Nextcloud Hub.

Proxmox

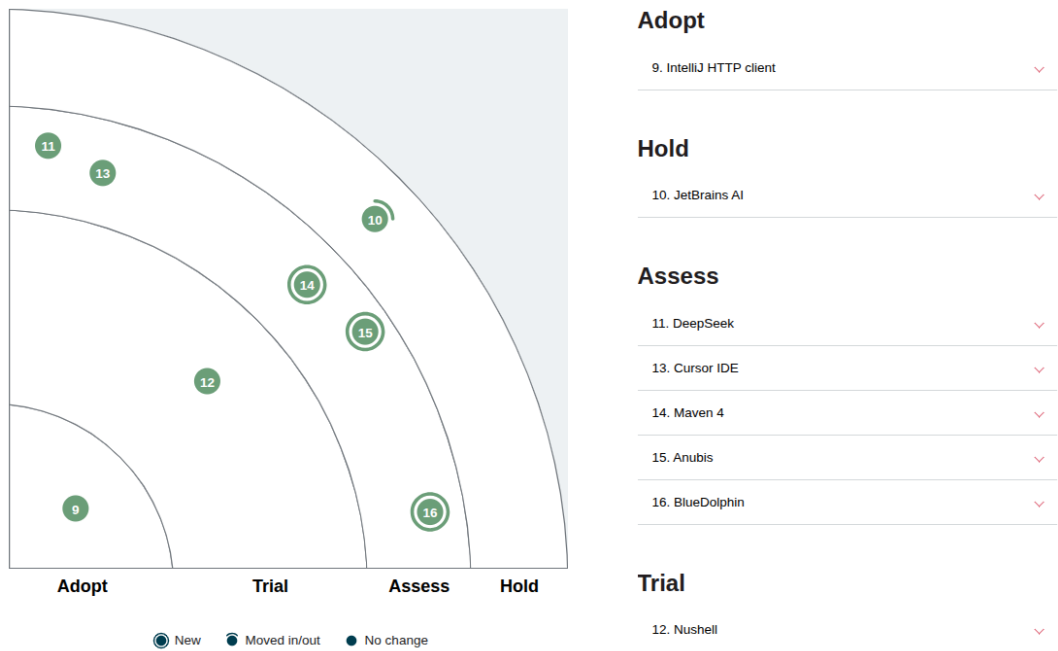
ASSESS

Proxmox [<https://www.proxmox.com/en/>] offers an alternative to the commonly-used VMWare platform. The Proxmox Virtual Environment (PVE) is an open source virtualisation platform for running containers and/or virtual machines with varying operating systems. The PVE supports high-availability over multiple servers, and includes resource-efficient backup functionality.

Although this technology is not new, in the context of ever-growing uncertainty surrounding international cloud services it may be worth considering hosting and operating IT services on self-maintained servers once more. We think Proxmox is a more interesting option than VMWare, as the latter has seen increasing licence costs since its acquisition by Broadcom. We have placed Proxmox in Assess because the applicability is entirely dependent on the situation. The gains in lower (licence) costs and increased control must be considered against the loss of scalability and available support.



Tools



IntelliJ HTTP client

ADOPT

The IntelliJ HTTP Client is an integrated feature within IntelliJ IDEA that enables developers to create and execute HTTP requests directly from the IDE. It uses `.http` or `.rest` files in which HTTP requests can be defined in a straightforward manner. This allows for fast and efficient API testing without relying on external tools like Postman.

The HTTP Client supports the use of variables, scripting, and various authentication methods, making it suitable for both simple and advanced use cases. Responses are clearly displayed and saved alongside the request files, which simplifies analysis and debugging.

A key advantage is that request files can be stored within the project structure, alongside the source code. This enhances reusability, version control, and collaboration within teams, and contributes to better-documented and reproducible API development. Since IntelliJ is a widely used IDE, it is a logical choice for developers already familiar with the environment who prefer not to use a separate tool for API testing.

While the client has been available for some time, recent updates have introduced robust new features, including support for environment files, secure handling of authentication, OAuth2 flows, and variable usage. For developers already working in IntelliJ, it offers all the necessary functionality to develop and test APIs, without ever leaving the IDE.



Nushell

TRIAL

Nushell [<https://www.nushell.sh>] is a new type of shell. The goal of Nushell is to take the Unix philosophy of shells, where pipes connect simple commands together, and combine it with a rich programming language. Making Nushell both a programming language and shell in one package, that runs on Linux, macOS, and Windows. Nushell connects both by bringing a rich programming language and a full-featured shell together into one package. Code written in Nushell can be run on Linux, macOS, and Windows. Nushell uses structured data everywhere, for example output of `ls` command, or REST API calls. Common commands to query, sort, filter and transform data are available and can be combined with each other. For example to find files larger than 10 MB and sort them by last modified: `ls | where size >= 10mb | sort-by modified`.

Nushell has been around for a couple of years, but it is still in the 0.x release stage. Therefore, it is very interesting to play around with it and see what the advantages and disadvantages are compared to other shells and tools.

Anubis

ASSESS

Anubis is a web AI firewall utility written in Javascript that presents proof-of-work challenges to the browser in order to protect upstream resources from scraper bots. Modern browsers are able to perform this proof-of-work challenge, which functions as an alternative to CAPTCHAs without requiring real human interaction. A default web scraper bot on the other hand won't be equipped to fulfill the proof-of-work challenge and will be unable to effectively scrape and extract the wanted data. Anubis thus acts as a 'tar pit' to slow down web scrapers, commonly used to gather data for training AI models, to avoid smaller web pages from being over-encumbered by these scraper bots.

Anubis is a young project, and it remains to be seen whether its approach will stay effective once scrapers adjust. But in adapting so that they can perform the proof-of-work challenge, scrapers would slow down considerably, leading to fewer requests. Because implementing Anubis is a relatively small investment, we think it's worth giving it a try. It's important to configure Anubis to let friendly bots like search indexers through, if so desired. The default image served by Anubis, that visitors will briefly see, won't fit with every website theme, but it can be adjusted to one's needs.



BlueDolphin

ASSESS

BlueDolphin [<https://www.valueblue.com/bluedolphin>] is a platform for enterprise architecture and process modeling. It offers a comprehensive set of features for visualizing organizational structures, application landscapes, and information flows. Its strength lies in supporting collaboration between various roles within an organization, enabling teams to shape both the current and the desired architecture in a visual and structured way. This enhances understanding of the impact of changes and helps to substantiate architectural decisions.

However, the abundance of features also comes with a downside. Due to the large number of options and configuration possibilities, it is easy for new users, and sometimes even experienced ones, to lose their way. The learning curve is steep, and without clear guidelines or governance, the use of BlueDolphin can quickly become unstructured or inconsistent.

While BlueDolphin has the potential to help organizations structure and communicate their architecture, it requires a clear vision and guidance to be used effectively. It is therefore important to carefully consider the purpose, scope and training needs related to the use within the organization.

For these reasons, BlueDolphin is placed in the Assess quadrant of this edition of our Tech Radar.

Cursor IDE

ASSESS

Cursor IDE [<https://www.cursor.com/>] is an AI code editor which is based on Visual Studio Code (VS Code). It offers integrated AI, so that AI support is embedded in the development process. It has different features like code completion, code generation, code review, etc.

We've been seeing speakers at various recent conferences use Cursor, and it's an interesting alternative for VS Code. Instead of having to rely on AI-plugins, Cursor distinguishes itself as an IDE with integrated AI support. JDriven sees the benefits for developer workflow and productivity by having an AI help with peripheral matters. There is a learning curve before being able to use it efficiently, however. And we notice that so-called *vibe coding* in general can cause a developer to have to spend more time reviewing and correcting AI-generated code, than on creating and understanding solutions. This is a risk, as developers remain responsible for the code that ends up in production. Requests to the LLMs involve credits and waiting times, which can be circumvented by pay-per-use. Either way, it's important to note that requests are sent to Cursor servers, which has privacy and security implications. All things considered we're placing Cursor IDE on Assess.

DeepSeek

ASSESS

DeepSeek [<https://tweakers.net/nieuws/231566/nederlandse-ambtenaren-mogen-deepseek-niet-gebruiken-van-kabinet.html>] is an alternative LLM to ChatGPT that can be used for various tasks, including code assistance. Its reasoning model and training allow it to perform accurately even with fewer resources, making it part of a new generation of efficient open-source LLMs that can run locally while still delivering strong performance.

Despite its potential, DeepSeek is not without controversy. The legitimacy of its data acquisition raises concerns, similar to ChatGPT, and there are also worries about potential Chinese censorship. We have seen that many of our clients prefer not to use cloud-based LLMs due to the risk of their data being shared. A locally running LLM like DeepSeek addresses this concern by providing assistance without sending information externally.

When combined with tools like Devovx Genie, DeepSeek can offer in-IDE suggestions for code. However, its capabilities extend beyond coding, making it a versatile AI tool.

We place DeepSeek in Assess, as it can be used locally and presents an interesting alternative, but it still requires further investigation.

Maven 4

ASSESS

Maven 4 [<https://maven.apache.org/whatsnewinmaven4.html>] is currently in Release Candidate stage and brings significant improvements for Java projects. The main gains are in performance: parallel dependency resolution and improved caching can reduce build times significantly, especially noticeable in large multi-module projects. Additionally, Maven 4 introduces the separation between build and consumer POMs, allowing developers to use more flexible configurations without affecting dependent projects.

Improved error messages and automatic parent version inference make working with Maven more intuitive. For teams working with modern CI/CD pipelines, the better integration capabilities with container-based builds and cloud native environments are a welcome addition. Maven 4 introduces a new dedicated bom packaging type with support for exclusions and classifiers, simplifying dependency management in multi-module projects. The long-awaited built-in support for CI-friendly versioning with `${revision}`, `sha1`, and `changelist` properties now works without the Flatten Maven Plugin, along with automatic parent and subproject versioning.

The lifecycle has evolved from a graph to a tree structure with new phase types like `before:all` and `after:all` for more granular control. Profile management has become more flexible with optional profiles using a `?` prefix, preventing builds from breaking when a profile doesn't exist. These improvements make Maven 4 significantly more user-friendly for modern development workflows. There are some breaking changes, particularly around plugin compatibility and stricter POM validation, requiring careful migration. Note that Java 17 is the minimum requirement for Maven 4.

We expect that when Maven 4 reaches GA (General Availability), it will be a solid choice for projects, given the significant performance improvements and the enhanced developer experience. Therefore, it's important to already look at what's involved in a migration. Given the significant performance improvements and enhanced developer experience, we place Maven 4 in the Assess ring.



JetBrains AI

HOLD

JetBrains [<https://www.jetbrains.com/ai/>] has introduced two AI technologies: **Mellum** (October 2024) and **Junie** (January 2025). Mellum is a specialized LLM optimized for code completion with support for Java, Kotlin, Python and Go. It has a 3x faster response time and a 40% acceptance rate of suggestions. Junie is an AI coding agent within the JetBrains Pro and Ultimate line that can independently solve 53.6% (according to SWEBench) of development tasks. While Mellum functions as an intelligent code completer, Junie is a semi-autonomous assistant that can complete more complex tasks such as generating code, running inspections, and writing tests. Both tools are designed with privacy as a priority – Mellum is trained on publicly available, permissively licensed code – and give developers control over proposed changes. JetBrains' vision with these tools is to make software development more productive and enjoyable by combining immediate assistance (Mellum) with an AI partner that can fundamentally change the development workflow (Junie).

For a developer, designing and crafting solutions is often the enjoyable part of our work. We already spend quite some time reading and understanding code written by others (including AI), and not being actively involved in the solution process may make this part of the job more difficult. It is important to evaluate whether these technologies actually address the right problems, such as reducing boilerplate code and providing an efficient alternative to Stack Overflow. JetBrains has significantly improved support for running Mellum locally in the AI Assistant (using Ollama or LM Studio), which allows code traffic to the JetBrains servers to be prevented. This addresses a potential dealbreaker for organizations with strict privacy requirements. Additionally, it's good to remember that JetBrains AI encompasses more than just these two technologies. Despite Junie's advanced functionality as an autonomous agent and the strong privacy focus through local LLM support, we see practically no adoption of JetBrains' proprietary AI tools within our organization. Developer focus is much more on better-known, platform-agnostic alternatives that seem to better align with the daily workflow, such as GitHub Copilot, Cody, and Claude Code. Since adoption rate is an important factor for placement within a ring, we have placed Mellum and Junie in the *Hold* ring, and encourage evaluating multiple tools yourself to determine which one offers the most value.





Commit.
Develop.
Share.