draw.io

# UML diagrams in draw.io

A practical guide to UML diagramming in Confluence

# Contents


**p.3** Why UML diagrams deserve a better home

**p.4** UML and draw.io: A shared philosophy

**p.5** UML diagramming in draw.io

- **p.5** UML shape libraries
- **p.6** Generate UML diagrams with AI
- **p.6** UML templates
- **p.7** Real-time collaboration

**p.8** draw.io for Confluence

- **p.8** Versioning and exportability
- **p.9** Security

**p.10** UML diagram types: A quick reference

**p.11** Further resources

**p.11** What our users say

**p.12** Bring your UML diagrams to life with draw.io



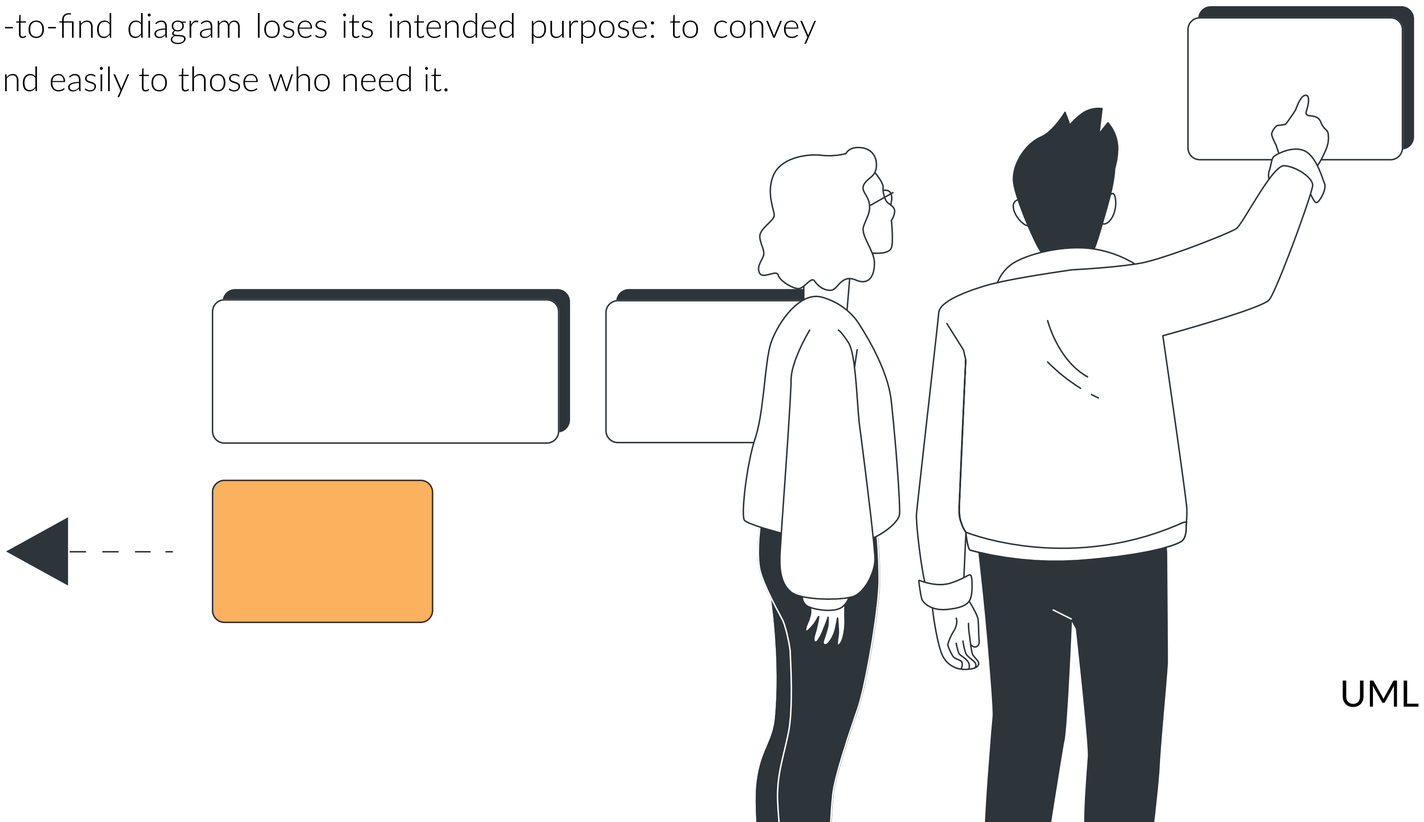

# Why UML diagrams deserve a better home

For over 25 years, the Unified Modeling Language, a standardized set of diagram types used to model the structure and behavior of software systems, has been a staple of software engineering. It gives teams a shared, universal way to describe systems, from high-level requirements down to the nuts and bolts of class relationships and deployment architectures. If you work in software, chances are UML is already part of your toolkit.

Creating the diagram itself is only part of the process. The crucial bit is, what happens afterward? Too often, UML diagrams end up scattered across tools that half the team doesn't have access to, saved as local files that never get shared, or exported as static images that are out of date by the time they're attached to a Confluence page. Despite the best intentions, a hard-to-find diagram loses its intended purpose: to convey meaning quickly and easily to those who need it.

This is very much a documentation and access problem, and one that gets worse as teams grow and systems get more complex; the distance between where the diagram and its related documentation are stored gets ever wider.

This whitepaper looks at how draw.io for Confluence has everything you need for UML diagramming: what's available in terms of shape libraries, templates, and features like AI-generated diagrams. We'll also look at why Confluence is the natural home for your UML diagrams, keeping them in context, easy to find, and ready for collaboration.

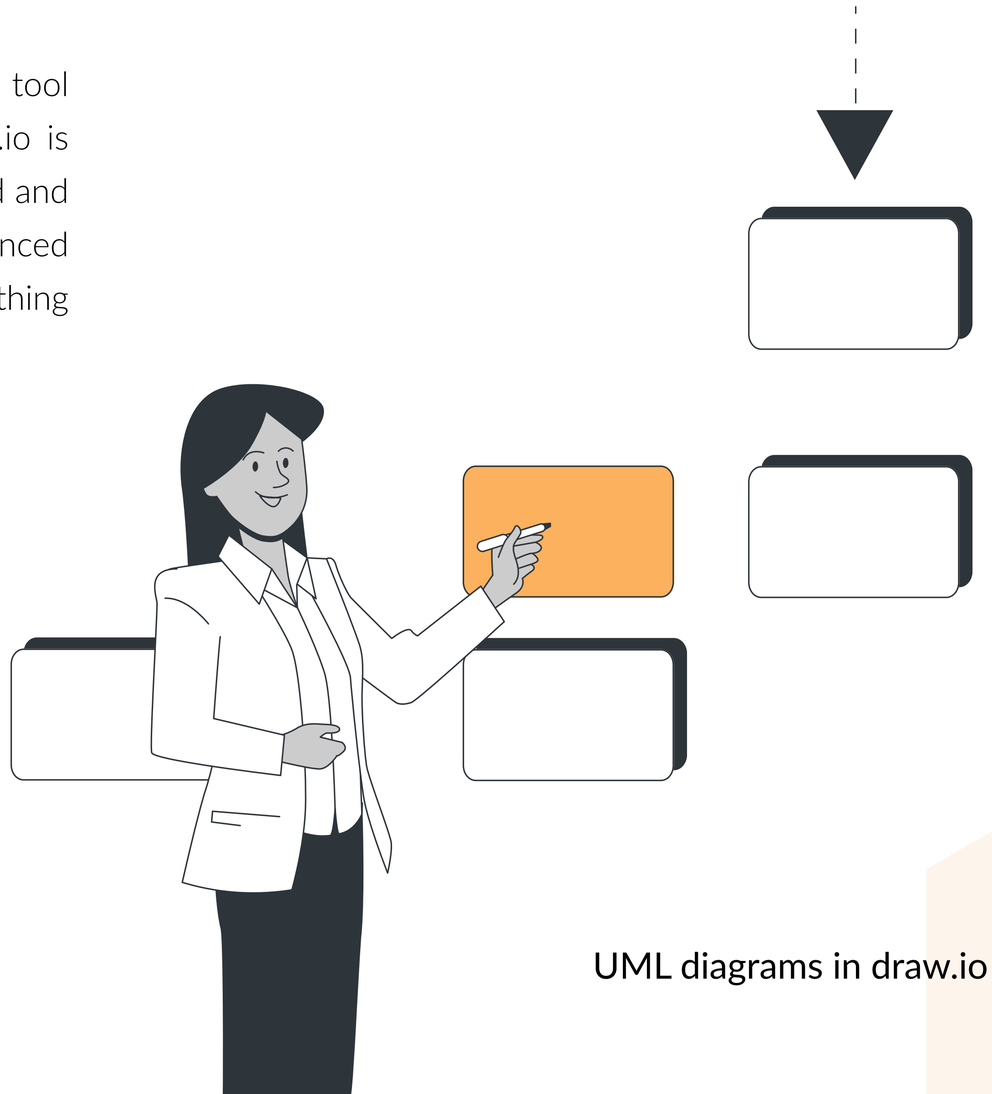


# UML and draw.io: A shared philosophy

UML has been an **ISO standard** since 2005 and an **OMG specification** since 1997, and remains one of the most widely used modeling notations in software development. It provides a common language that teams rely on, giving everyone a shared understanding without having to dig into the underlying code. At its core, UML prioritizes clarity and function. A class diagram doesn't need to look flashy; it needs to be precise and readable. The notation is there so you can focus on the system, not the drawing.

draw.io is built on a similar principle: like UML, your diagramming tool should never be the obstacle — it should be the facilitator. draw.io is intuitive by design: insert a new diagram on a Confluence page, build and customize your diagram, and then share it with your team. Advanced features, like **multipage diagramming** and **layers**, keep everything organized and uncluttered.

There are dedicated UML tools out there, and some of them are excellent. But if your team already uses Confluence as its single source of truth, there is a real advantage to having your UML diagrams live right alongside your design documents, technical specs, and project pages, rather than in a separate application that some teammates may not have access to.



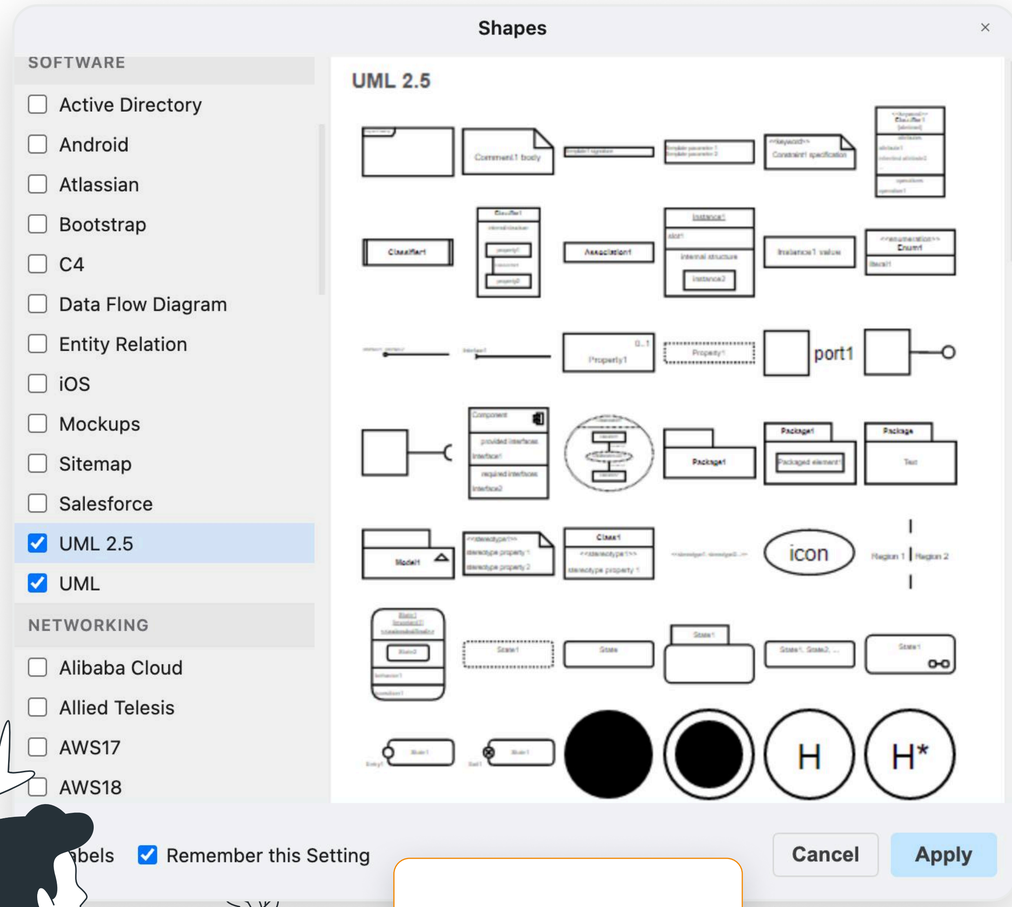
# UML diagramming in draw.io

## UML shape libraries

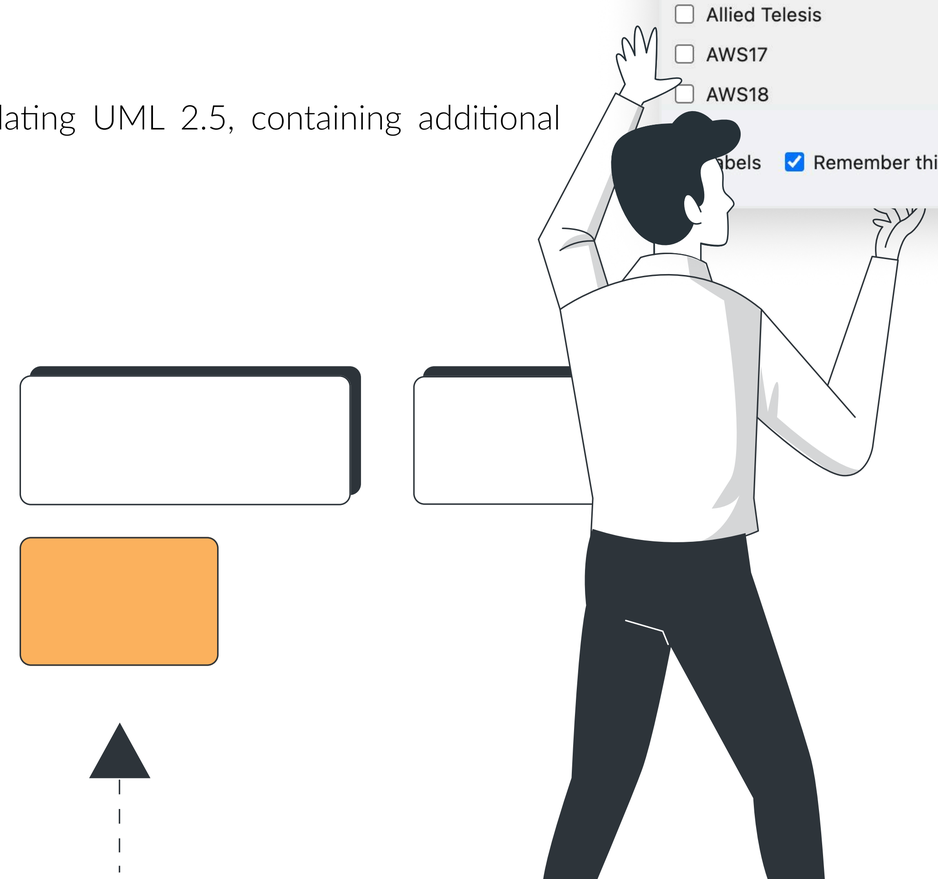
draw.io includes dedicated UML shape libraries, including UML 2.5 notation. You'll find the standard structural and behavioral elements — classes, interfaces, packages, actors, lifelines, swimlanes, states, activities, nodes, artifacts, etc.

The libraries are found in the left side panel under ***+More shapes > Software***, and are organized as follows:

- **UML 2.5:** The current specification, covering all standard diagram elements
- **UML:** The original library predating UML 2.5, containing additional and legacy shapes




*UML shape libraries*

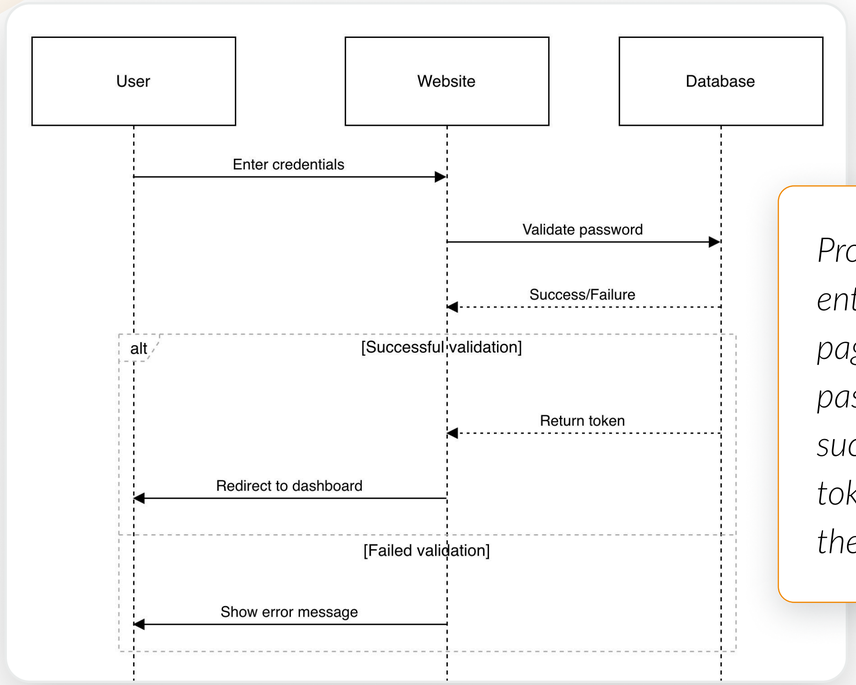


## Generate UML diagrams with AI

In addition to creating UML diagrams from scratch, you can also use our AI feature to produce some great results.

draw.io's **Generate tool** (a.k.a. Sparkle button✨) lets you create UML diagrams using AI. Describe the system or process you want to model, and the Generate tool will produce a diagram you can then refine and build on.

This is helpful when you want to scaffold a first draft quickly, explore different ways to represent a system, or get a head start when documenting something new. The Generate tool handles the initial structure; you bring the domain knowledge to shape it exactly how you want.

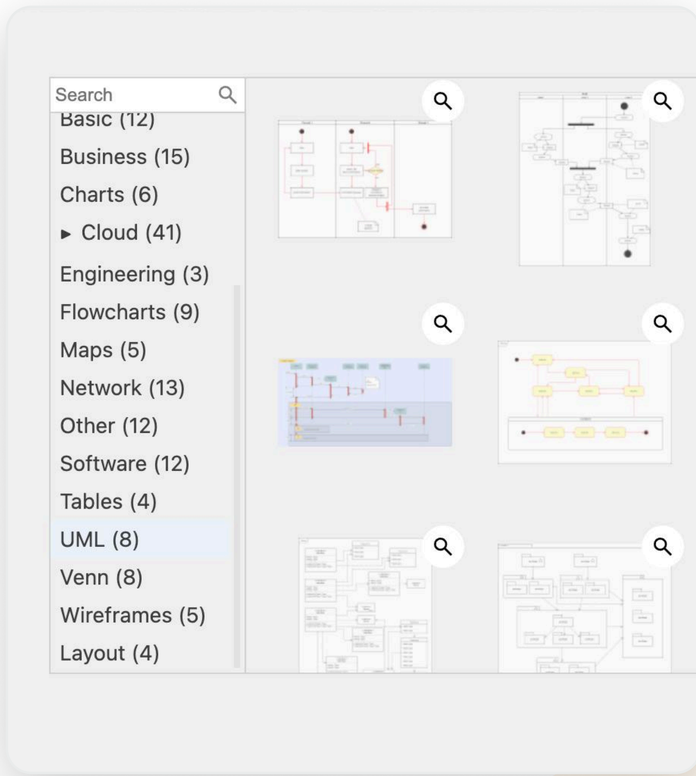



*Prompt: "sequence diagram - a user enters their credentials on a login page. The website validates the password with the database. If successful, the database returns a token and the user is redirected to the dashboard."*

## UML templates

draw.io includes ready-made templates for the following UML diagram types:

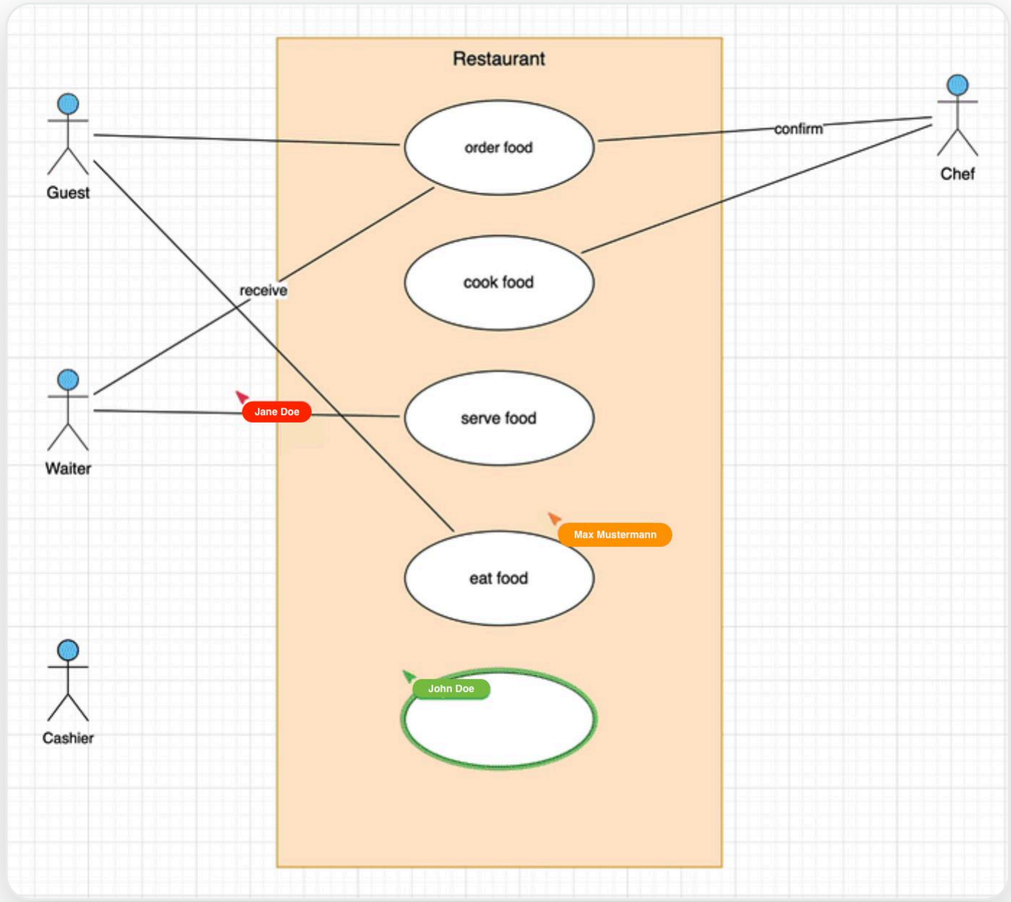
- Activity diagrams
- Class diagrams
- Package diagrams
- Sequence diagrams
- State machine diagrams
- SysML

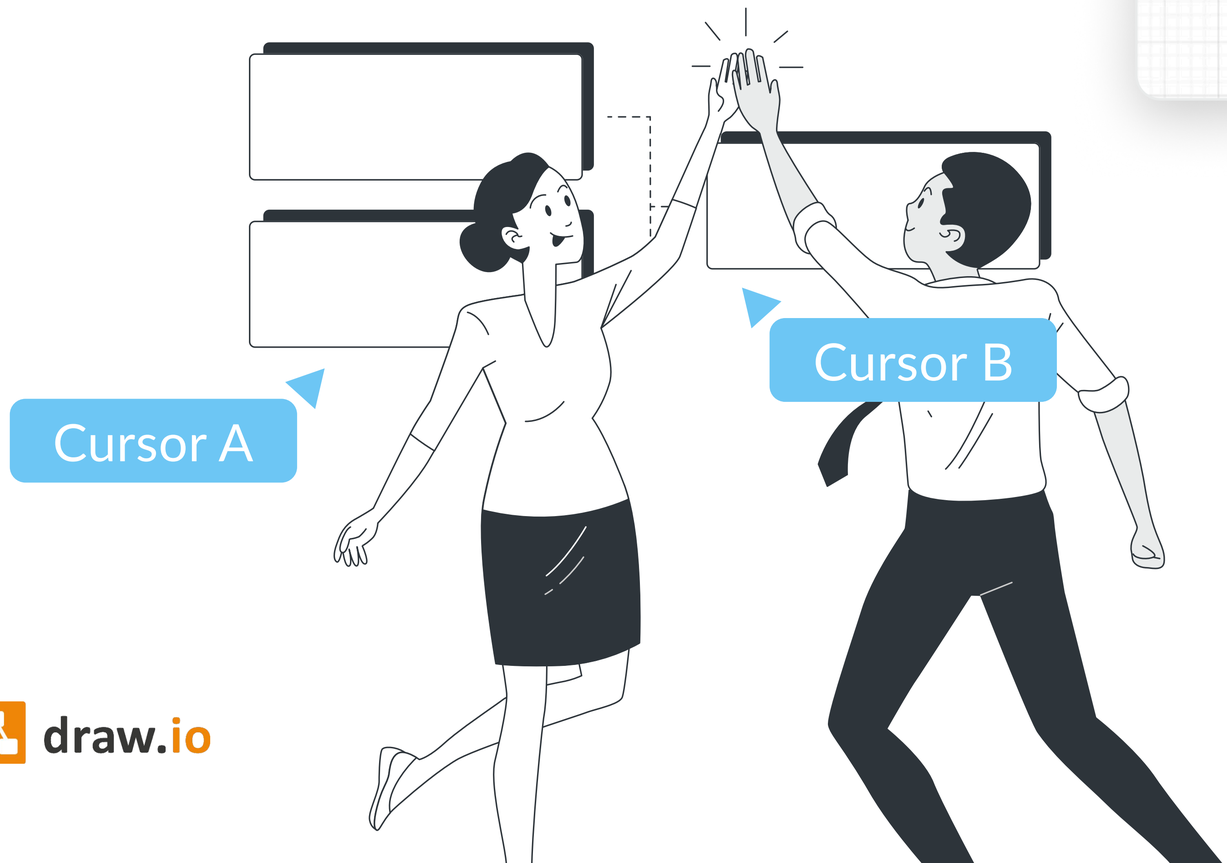


## Real-time collaboration

System design is rarely a solo activity. The architect drafts the component diagram, a developer refines the class structure, etc.

draw.io supports real-time collaboration with shared mouse cursors, so your team can work on the same UML diagram simultaneously without leaving Confluence. In Agile environments where diagrams evolve alongside the system they describe, being able to update a UML diagram together in real time saves a lot of back-and-forth. This is ideal for resolving conflicting design decisions in the same diagram session, or updating a diagram during a sprint review.



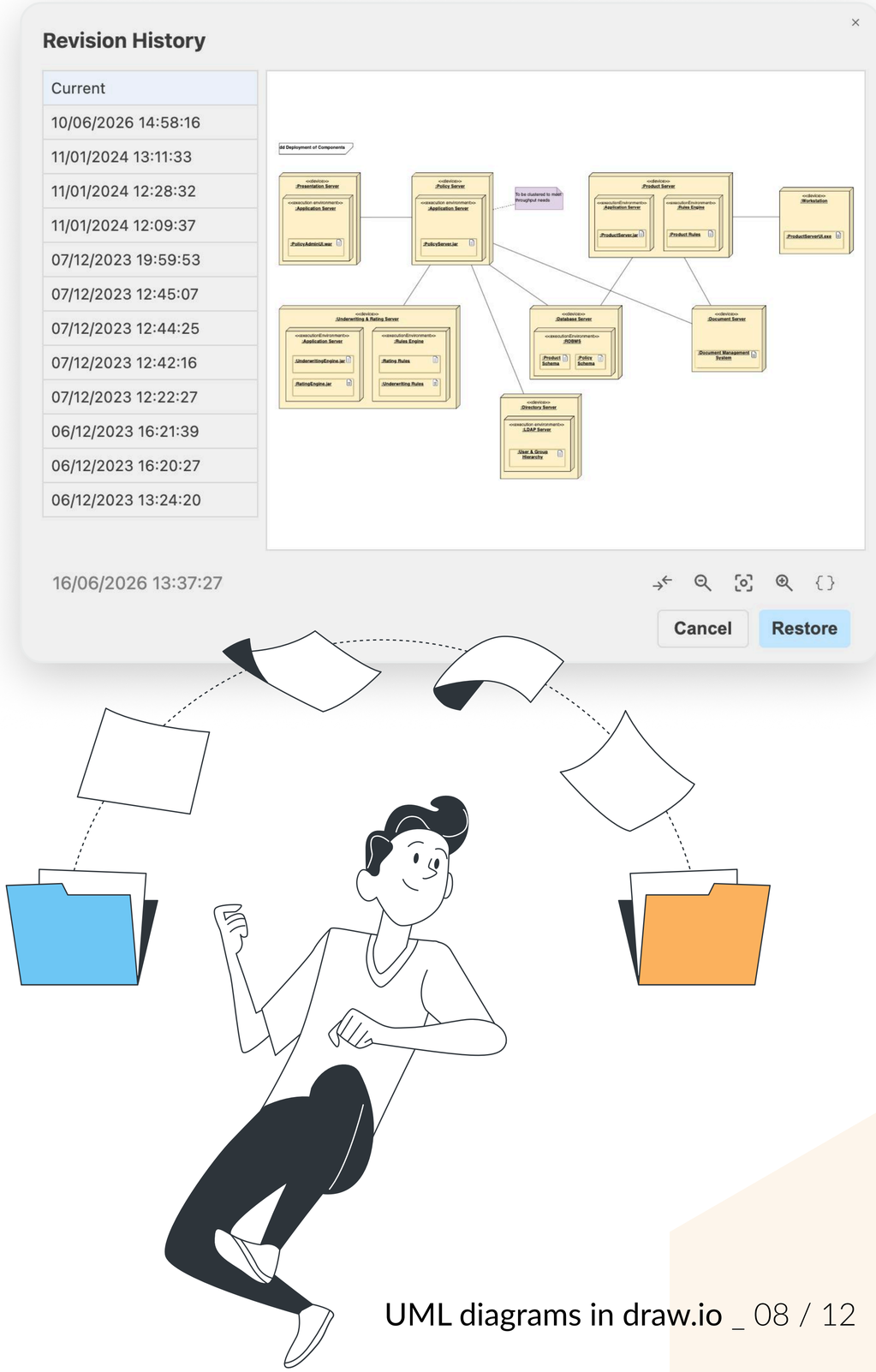
# draw.io for Confluence

draw.io is a native Confluence app, meaning your diagrams are embedded directly in Confluence pages right alongside the documentation they relate to. They're not in a separate tool that requires another login, another license, or another bookmark. Anyone with access to the Confluence page can view the diagram. You control who sees what, and who has view vs. editing rights, using Confluence's own permission settings.

## Versioning and exportability

Every change to a diagram is tracked through version history. You can compare revisions and revert to a previous version, either at the diagram level in draw.io or the page level in Confluence. When system designs are changing frequently, this kind of traceability matters.

When you need a diagram outside of Confluence, for a presentation, a PDF handout, or an architecture review, you can export to PNG, SVG, PDF, and other formats.

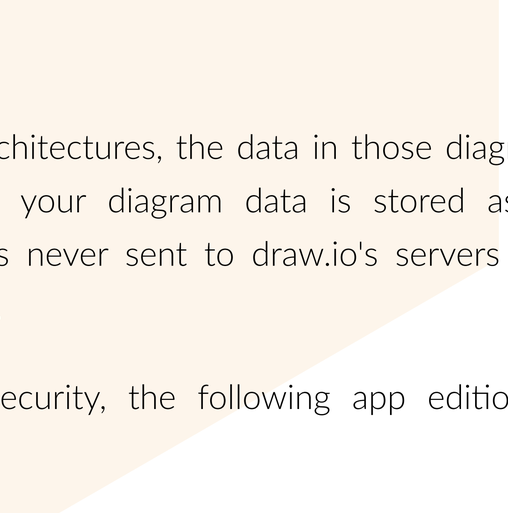


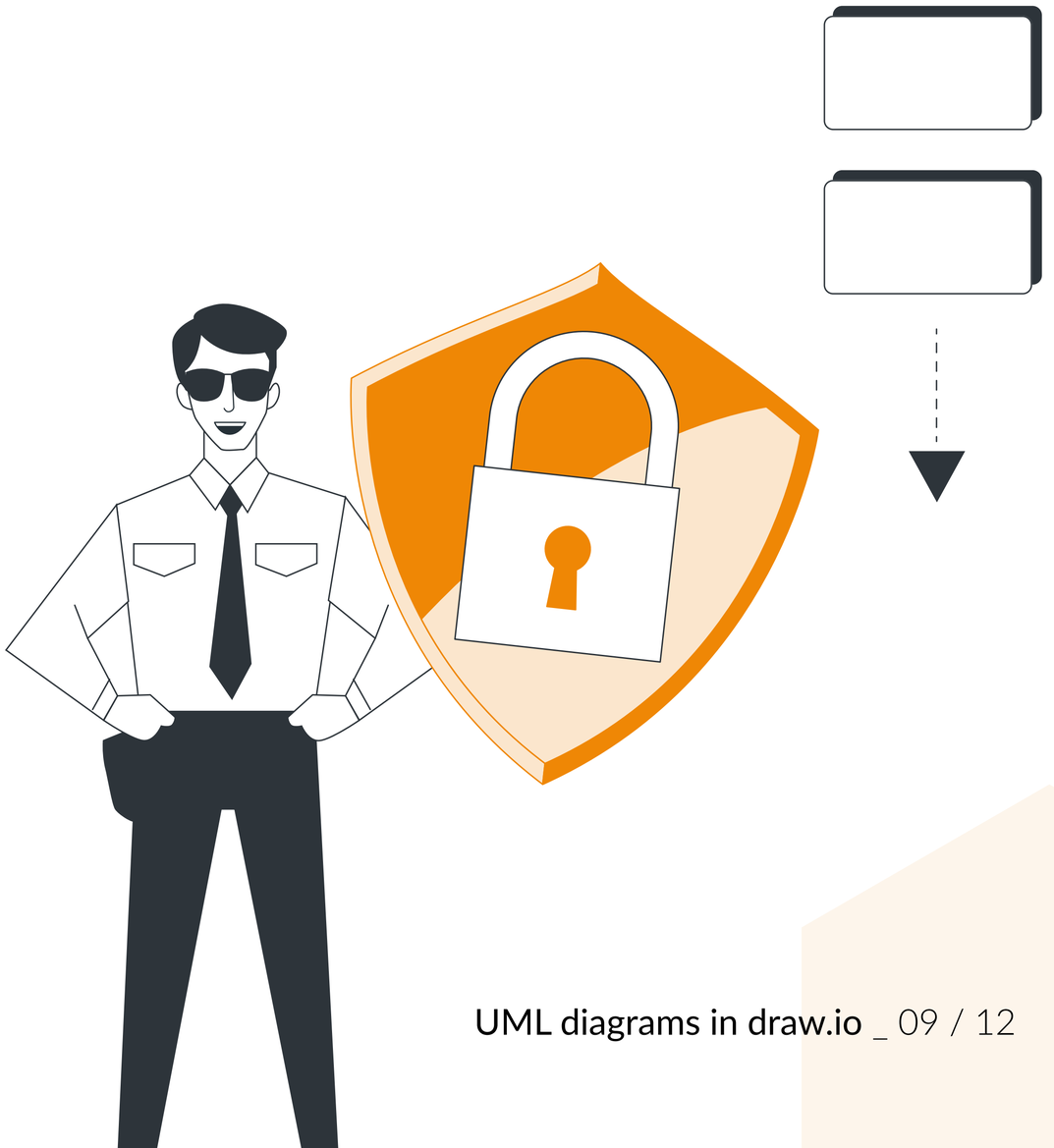
## Security

If you are visualizing system architectures, the data in those diagrams is often sensitive. With draw.io, your diagram data is stored as page attachments in Confluence. It's never sent to draw.io's servers during save or load, not even in transit.

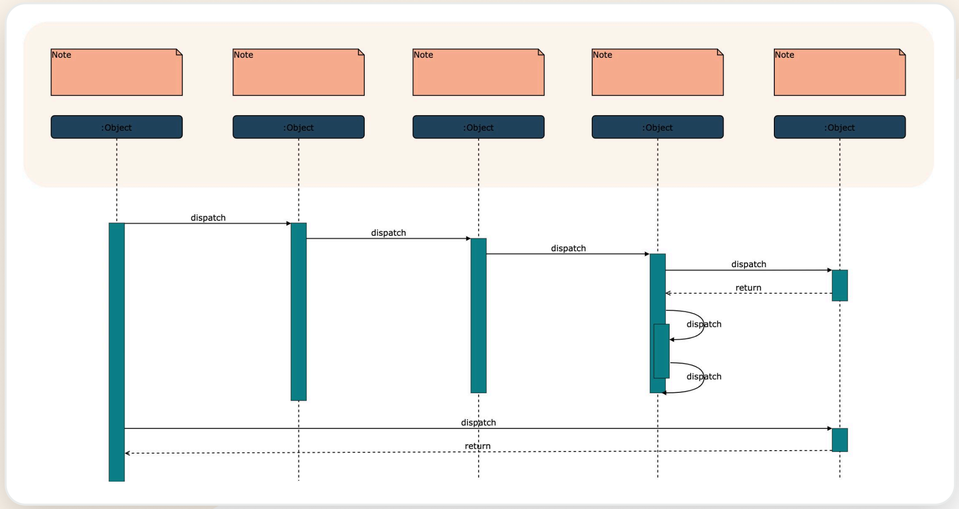
If you require even tighter security, the following app editions are available:

- **draw.io Advanced app edition:** Contains a zero-egress (i.e. “lockdown”) switch that can be enabled or disabled by your Confluence administrator. This restricts data transmission to only between the user's browser and the Confluence Cloud instance, eliminating cross-domain traffic except for Atlassian API calls.
- **draw.io Zero Egress app:** Built on Atlassian’s Forge platform, this removes cross-domain data transmission altogether. This is particularly relevant if you're working toward **ISO 27001, SOC 2,** or **GDPR compliance**, where controlling how and where data flows is a core requirement.





# UML diagram types: A quick reference




You probably have your own sense for when a UML diagram is the right call. You're typically ready to start diagramming when you can answer:

1. **What's the scope?** Define the system or subsystem you're modeling.
2. **Who's involved?** Identify the users, roles, and external systems that interact with it.
3. **What are the building blocks?** Whether these will be classes, services, nodes, or states.
4. **How is your team working?** UML fits both waterfall and Agile approaches, but the way you use diagrams differs. In Agile contexts, treat them as living documents that evolve with each sprint.

From there, the diagram type depends on what you're trying to visualize:

| If you need to capture... | Consider... |
|---|---|
| System requirements and user interactions | Use case diagrams |
| Object structure and relationships | Class diagrams |
| Message flow between objects over time | Sequence diagrams |
| Process workflows and decision logic | Activity diagrams |
| Object lifecycle and transitions | State machine diagrams |
| System architecture and infrastructure | Deployment diagrams |
| High-level subsystem structure | Component diagrams |

# Further resources

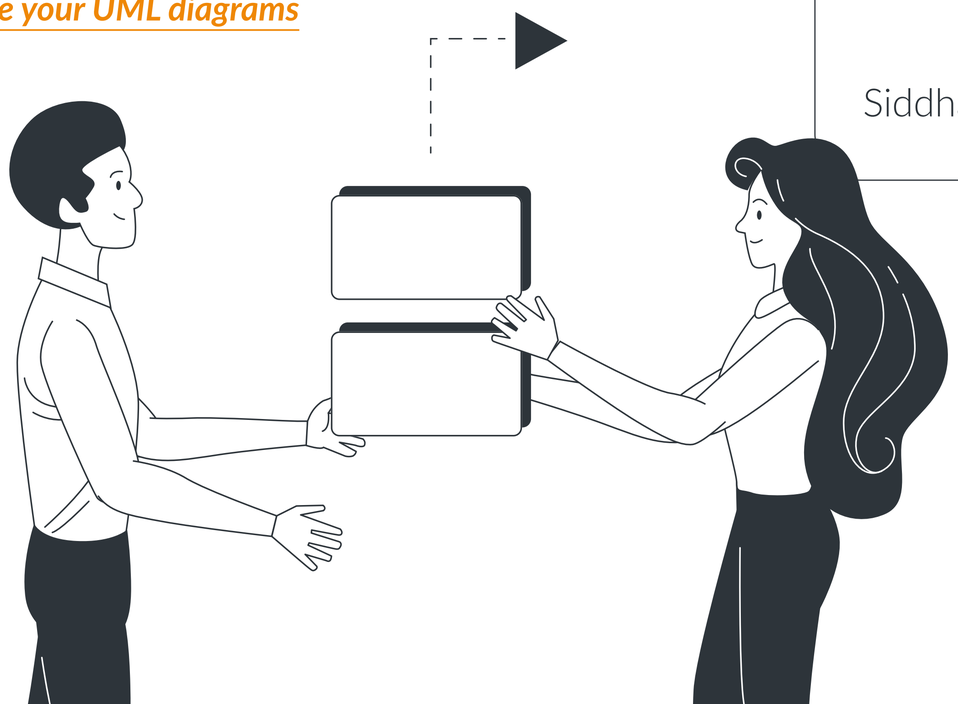
The blogpost links in the table above are a good place to start if you want to dive deeper into a specific diagram type. For more tips and tricks on creating UML diagrams in draw.io, see:

**Youtube videos:**

- *Demystifying UML: A beginner's guide to UML diagrams*
- *3 tips to optimize your draw.io diagrams*
- *3 facts about UML diagrams you need to know!*
- *Enhance your UML diagramming with live-collaboration with draw.io, in Confluence*

**Blogposts:**

- *UML diagrams: which diagram to use and why*
- *Collaborate on UML diagrams in draw.io*
- *Three ways to improve your UML diagrams*



# What our users say

"Love this tool... makes Use Case, UML or BPMN diagrams no effort at all with a perfect integration into Confluence and Jira available."

Dominico Klawitter



"draw.io needs no review, whoever has used it already knows that this is the best online tool for creating all sorts of design/ architecture/UML diagrams including complex flow charts."

Siddharth Kumar



# Bring your UML diagrams to life with draw.io

UML has lasted nearly three decades for a reason. It gives teams a shared language that cuts through ambiguity and provides clarity through a standardized notation and clear visualization. Creating UML diagrams is only part of the journey; they need to be somewhere central and accessible for team members to truly benefit from them.

draw.io goes beyond providing extensive UML shape libraries. It gives you ready-made templates, AI-assisted diagram creation with the Generate tool, real-time collaboration, and a security model that keeps your architectural data exactly where it belongs.

draw.io for Confluence makes your UML diagrams part of the living documentation your team already works in every day. All your diagrams are versioned alongside and mirror the same permission settings as the Confluence page they're embedded in. In short, your diagrams stay current, visible, and useful.

If your team is already in Confluence, there's no simpler way to make UML diagrams a working part of your documentation. Try it out with a **free 30-day trial**, or **book a demo** with our customer success team to see how it works in practice.

