Tazama Design Principles

Table of Contents

Background

One of our first tasks as a project in The Linux Foundation was to collaborate with our community and determine what our design principles were going to be to make sure we continue to build Tazama responsibly and deliberately, while keeping our eye on the mission for financial inclusion. Our community defined the following principles for the design and development of Tazama.

1. Open Source First

Imagine you're building a house. You have access to free, high-quality materials, but there's a catch: some of these materials might suddenly become unavailable, or worse, might now come with strings attached. This uncertainty can turn your dream project into a nightmare. This is the reality we face in the world of software development, and it's why we've embraced the "Open Source First" principle for our project, Tazama.

At first glance, the idea of prioritizing open source software might seem straightforward. After all, open source means free and open, right? Unfortunately, our experience has taught us that open source isn't always as open as it seems.

The Freedom of Apache 2.0

Tazama is available for free under the Apache 2.0 license, a "permissive" license that offers flexibility and freedom. This means you can use Tazama with minimal restrictions, and even build your own commercial products around it. But to maintain this freedom, we need to ensure that the software we integrate into Tazama is equally permissive.

Navigating Licensing Pitfalls

One of the biggest risks we face is relying on software from organizations that might change their licensing terms. Imagine building a key feature with a tool, only to find out later that you now need to pay for it. To avoid this, we steer clear of software from companies offering commercial versions of their open source products. Instead, we rely on software licensed under MIT or BSD, which tend to be safer bets.

Ensuring Upstream Transparency

When we incorporate open source software, we dig deep into its dependencies. We want to be sure that we're not exposing ourselves or our users to hidden compliance risks. This means verifying that all upstream dependencies are transparent and compliant with our standards.

Beyond the Code: Evaluating Open Source Products

When selecting open source software, we look beyond the code itself. Here are some key attributes we consider:

- **Community Support**: Is there an active and supportive community?
- **Stability**: Is the product well-established, and has the development team been around for a while?
- **Documentation**: How useful is the documentation in helping us implement the software?
- **Scalability**: Can the software meet our growing and dynamic performance needs?

Why It Matters

By adhering to the "Open Source First" principle, we ensure that Tazama remains a robust, flexible, and truly open product. This commitment not only benefits us as developers but also provides our users with the peace of mind that Tazama will remain free and open.

---

2. Do Not Reinvent a Serviceable Wheel

As software engineers, we love to build things; sometimes with so much enthusiasm that we don't always stop to check if our grand idea or solution is original!

At Tazama we quickly learned that in the world of open-source development, the principle "Do not reinvent a serviceable wheel" must be a guiding beacon. It underscores the importance of harnessing existing solutions, maintaining flexibility, and avoiding unnecessary custom development. We firmly believe in this principle, which drives our approach to building robust, scalable, and efficient software.

Harnessing Existing Open Source Solutions

One of our core beliefs is to reuse good work already performed in open source projects to solve common problems. The open-source community is filled with brilliant solutions developed by talented individuals and organizations. By integrating these solutions, we can focus on our unique challenges while benefiting from the collective wisdom and experience of the community. For example, we use NATS.io for our messaging needs, a tool that has proven its reliability, scalability and efficiency across numerous projects. This not only saves us development time but also ensures we are using a battle-tested solution.

Abstraction for Flexibility

We aim to abstract our use of, and integration into, third-party software to make it easier to change our minds if we find something better or if an existing solution is no longer suitable. This approach allows us to stay agile and responsive to new developments in technology. For instance, our modular architecture means we can easily switch from one authentication solution to another without significant rewrites. This flexibility is crucial for adapting to changing requirements and leveraging the best tools available.

Don't Fork!

Forking someone else's product to adapt it to our requirements can lead to significant maintenance burdens and divergence from the main project. Instead, we strive to use the officially published packages for our products without modification. For example, Tazama can be deployed with the versions of Redis, NATS, and ArangoDB published on Docker Hub to make our deployments simple and straightforward for our users.

Custom Solutions as a Last Resort

Building custom software to solve a known problem should always be a last resort. Custom development is time-consuming and resource-intensive, and it often introduces new risks and maintenance challenges. Whenever possible, we prefer to adopt existing tools and frameworks that have already been tested and proven. For instance, instead of developing a custom logging framework, we chose to integrate with Elasticsearch, Logstash, and Kibana (ELK Stack), which offers comprehensive logging and monitoring capabilities out of the box.

Why it Matters

The principle of "Do not reinvent a serviceable wheel" is about smart resource management and capitalizing on the strengths of the open-source community. By enforcing this principle in our design decisions, we can focus on building the really cool things that make Tazama unique and simultaneously also fill a niche that our users and other open-source projects can benefit from.

Top

3. High Performance at Low Total Cost of Ownership

In a rural community in Malaysia, a community bank aims to provide essential financial services to its members, many of whom have limited access to traditional banking. The bank plans to introduce digital banking but faces challenges due to the high costs of implementing a robust transaction monitoring system. This situation highlights a critical issue in digital financial services: balancing performance and affordability. At Tazama, we believe that protecting financially vulnerable customers shouldn't come at an exaggerated cost to those same customers. This is the cornerstone of our design principle, "High Performance at Low Total Cost of Ownership."

Real-Time Interdiction

Real-time interdiction is vital for detecting and potentially blocking fraudulent transactions instantly. At Tazama, we emphasize real-time monitoring to ensure immediate responses to suspicious activities, without impacting the customer experience with undue delays.

Scalability in Response to Load

As digital banking grows and cycles, transaction volumes can ebb and flow. A system that scales effectively is crucial to avoid high base fees. Tazama's must be designed to idle at low cost and handle increased loads on demand without a significant rise in costs to ensure efficient, reliable, and cost-effective service regardless of transaction volume.

Low Cost to Operate (Total Cost of Ownership)

Total cost of ownership (TCO) includes setup, maintenance, and operational costs. Tazama prioritizes cost-effectiveness by using open-source components and building automated and configurable processes to reduce the need for specialized personnel. Costs saved by the operator translates to costs saved by the customer.

Efficient Use of Resources

Resource management is key to reducing costs and maintaining performance. Tazama must optimize computational and storage resources through efficient workflows and data management strategies. New features must minimize hardware requirements, making the system accessible even for institutions with limited resources.

Avoid Obvious Bottlenecks

Bottlenecks, such as data processing delays, can impair system performance. Avoiding obvious bottlenecks seems, well, obvious, but a deliberate and proactive approach will ensure that we don't do something obviously suboptimal just because we're not paying attention.

Benchmark New Features

Benchmarking is essential for maintaining high standards in new features. Tazama automatically conducts performance regression tests when GitHub Pull requests are submitted to ensure changes maintain (or improve!) performance against expectations.

Why It Matters

"High Performance at Low Total Cost of Ownership" is central to Tazama's mission. We focus on deliberate consideration of the requirements for real-time interdiction, scalability, cost-efficiency, and thorough benchmarking to provide a transaction monitoring solution that is performant and promotes accessible digital financial services. Robust digital banking protections should be available to all, regardless of their financial situation.

At Tazama, we are dedicated to creating a safer and more inclusive digital financial ecosystem. We invite you to join our community and contribute to this transformative journey, making high-quality financial services accessible to everyone.

---

4. Keep it Simple to Keep it Inclusive

In a rural region of Kenya, a small agricultural cooperative faces challenges as they try to integrate mobile money services with a secure transaction monitoring system. The cooperative's members, lacking technical expertise, struggle with complex setup processes, leaving their financial transactions vulnerable. This scenario is all too common in areas where technology is essential, but technical resources are limited.

At Tazama, we understand that true inclusivity means making technology accessible, regardless of technical background. This belief is the foundation of our design principle: "Keep it Simple to Keep it Inclusive."

Easy to Adopt

For technology to be inclusive, it must be easy to adopt, especially in regions with limited technical expertise. Tazama prioritizes user-friendly onboarding in its design. We aim to deliver clear, non-technical documentation and step-by-step guides to ensure that anyone can get started easily. We intend to lower barriers to entry, and make transaction monitoring accessible to all.

Easy Integration (Ingress and Egress)

Integration with other systems is sometimes the most complex and onerous technical task in any project. Our design decisions focus on interoperability and providing modular interface components that must minimize the nuts-and-bolts work of system integrations. Our APIs and connectors must be flexible, easy to use, and conform to well-known and universal standards. A user should be able to implement Tazama in diverse environments without overhauling existing systems.

Operational Configurability Over Development

In many regions, the expertise required for custom software development is scarce. Tazama addresses this by emphasizing operational configurability over the need for ongoing development. Users must be able to adjust rules, alerts, and reporting parameters through configuration, without needing expert coding skills. The focus on configurability over development will allow users to adapt the system to their needs, maintaining control over their transaction monitoring processes regardless of technical background.

Why It Matters

"Keep it Simple to Keep it Inclusive" is more than just a design principle—it's a commitment to making advanced financial technology truly accessible. We want to ensure that Tazama is easy to adopt, integrate, and configure, so that we can empower users in regions with limited technical resources to protect their financial transactions with confidence. This approach is essential for supporting financial inclusion, as it allows communities that are often left behind to participate fully in the digital economy.

---

5. Design for Failure

In the world of technology, failure isn't just a possibility—it's an inevitability. Systems crash, networks go down, and bugs find their way into even the most carefully crafted code. But what separates a robust platform from a fragile one is how it handles failure when it happens. At Tazama, we believe that designing for failure is so important that it's a fundamental design principle. We prioritize resilience, plan for failure conditions, and embrace eventual consistency, with the ultimate goal of responding to failure gracefully.

Consciously Consider Conditions for Failure

Failure is often unpredictable, but that doesn't mean we can't plan for it. With every new feature we design, we should consciously consider various conditions under which failure might occur, whether it's a sudden spike in transaction volume, invalid, incorrect or incomplete data, an infrastructure outage, or a software bug. We aim to identify and prioritize failure conditions and then build safeguards into the system to mitigate their impact. It is vital to do this work during the design phase, where we can proactively address them and minimize the cost of rework and fixes and impact on our users later.

Prioritize Resilience Across the Platform

Resilience is the cornerstone of a system designed to handle failure. Our goal here is to ensure that critical services maintain functionality even when less critical components fail. This means identifying the most vital aspects of our system and implementing robust failover mechanisms to keep them running. For example, historical data persistence is guaranteed so that even if a down-stream evaluation processor fails, the system can recover the evaluation from that point onwards and still deliver a result when the processor is restored. Blessing each component with the same priority for resilience is not only extremely expensive, but also ultimately results in no priority at all, but if we prioritize resilience according to the importance of each service, we create a system that can withstand disruptions without compromising its core, essential functions.

Fail Gracefully, and Recover

Tazama is a real-time transaction monitoring system - a crashed processor is as bad as an overturned truck on a busy highway during peak travel hours. When failure does occur, it's crucial that the system fails gracefully. This means that instead of crashing entirely, the platform should degrade in a controlled manner, allowing users to continue operating at a reduced level of functionality while the issue is resolved. Failures must be isolated, preventing them from cascading across the system. For example, if a particular service becomes unavailable, dependent services must still be able to function, albeit in a reduced capacity, to minimize the impact on users. The spice must flow. And just as important as failing gracefully is the ability to recover quickly. Tazama must incorporate automated recovery mechanisms that restore normal operations as soon as possible, ensuring that disruptions are short-lived.

Eventual Consistency Over Immediate Consistency

In a distributed system like Tazama, striving for immediate consistency across all components can lead to significant delays and increased susceptibility to failure. Instead, we embrace the principle of eventual consistency. This means that while the system may not be instantly synchronized across all nodes, it will eventually reach a consistent state. Eventual consistency allows Tazama to remain responsive and functional even when parts of the system are experiencing issues. For instance, during high traffic periods, data updates may be temporarily delayed, but the system continues to operate effectively, catching up as conditions stabilize - our pub/sub design reduces the pressure on the system and helps maintain its overall resilience.

Why It Matters

"Design for Failure" is not just about avoiding the inevitable—it's about embracing it and ensuring that when things go wrong, they don't go catastrophically wrong. Tazama intends to be a system that remains reliable and effective, even under challenging circumstances, so that the operational burden on the operator is reduced, and the system can continue to fulfill its ultimate purpose: building trust with new and vulnerable users in the emerging digital financial economy.

---

6. Appropriate Best Practices

Let's preface this design principle by acknowledging just how loaded a term such as "best practices" is. It is, as it should be, a somewhat subjective term and greatly influenced by the unique context and capabilities of a particular project. "Best" practice in Tazama design is also a function of our maturity, and we balance the burden of policies and standards with the pursuit of innovation expressed (quite often, still) in a "minimum viable product" and "iterative progress". That said, there are a few things that we generally strive for in our product design because by now we've worked out that these are "good" things, even if they're not yet the "best" of everything, and considering them early saves a lot of rework later.

Seamless Component Replacement

We need to design for easy replacements through established practices such as:

- **Bounded Contexts**: We define clear and firm boundaries within which components operate.
- **Abstraction**: We abstract component interaction into services and libraries.
- **Encapsulation**: We ensure that components function independently.

Sound Dependency Management

Managing dependencies effectively is key to a stable platform. At Tazama, we minimize risks by carefully cataloging, selecting and managing external libraries and services, regularly reviewing and updating them to maintain security and compatibility.

Separation of Concerns

Separation of concerns is fundamental to Tazama's design, especially considering the distributed processing nature of the system and its components. Different processors have specific and discrete jobs, and are supported by common functions that are centralized into fit-for-purposes libraries. We drive modularity very hard to make the system more flexible and easier to maintain.

Code Maintainability and Readability

We maintain clear, well-documented code and enforce code quality through automated tools. An accessible Tazama is not just for end-users and implementers, but also for our contributors.

Why It Matters

"Appropriate Best Practices" isn't just a checklist for us - it's a philosophy that ensures Tazama remains easy to use, easy to implement, and easy to contribute to. These practices enable us to deliver a high-quality, maintainable platform that can easily outlive any of its individual parts. Tazama must be enduring, and accessible to everyone to be useful for everyone.

Top