LoanPro
Modern Credit Platform

Flexible rule-based lending automation using Clojure

Automation to optimize operations

LoanPro provides flexibility and automation to optimize underwriting, servicing, and collections efficiency for almost any type of loan or credit product. This flexibility enables lenders to take products, hardship programs, servicing solutions, and more to market quicker, an advantage in the competitive marketplace, while offering the agility to provide a better customer experience.

LoanPro automations make it easier to keep loans up to date, process payments, send custom communications, and complete lending processes. While the automated actions themselves are extremely important, maximum flexibility requires a robust method of creating the rules that define when automations will occur. LoanPro chose Clojure as the flexible, powerful, secure way to write these rules.

Why Clojure?

With so many options available for automation and system rules, you might ask why LoanPro chose Clojure. LoanPro could have created their own rules language or chosen a more mainstream language like Python. At LoanPro, we considered many options, and ultimately chose to use Clojure for these key reasons:

- Clojure is consistent because of its predictable structure and data immutability
- Clojure is performant
- Clojure provides all the functionality needed to create both simple and complex rules
- Clojure has a simple core for ease of learning and use
- Clojure rules can be executed securely within the LoanPro platform

Consistency

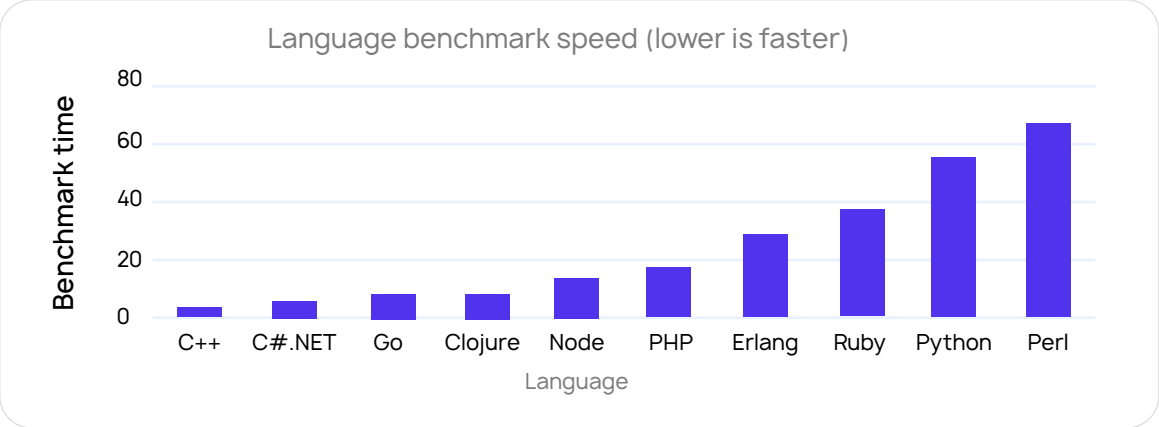
Clojure is a standout choice for consistency and predictability, thanks to its functional paradigm and immutability at the core. Clojure syntax is kept at a minimum, so it's very easy to explain the core set of rules needed to write Clojure code. Because there is immutability at the core, there are no cases when an instruction inadvertently mutates a collection, preventing the creation of subtle, difficult to detect bugs.

Clojure performance

When choosing the technology for automation rules, it was important that encoded rule instructions execute quickly. To correctly determine when automations should run, LoanPro evaluates tens of millions of rules daily. The wrong technology choice would have significantly increased the time these evaluations required, hindering customer use of the LoanPro platform. Clojure provides the required speed so rule evaluations and related automations occur in a timely way, ensuring that loans, leases, and lines of credit are always up to date.

While Clojure isn't the fastest programming language, it strikes a good balance between performance, ease of use, consistency, and security. For example, rules written in Go might be faster, but would be more difficult to serialize and evaluate securely.

The chart below shows Clojure speed, based on benchmarks, relative to other popular web languages (lower scores are faster).



Complex rules

With over 600 customers, rules are required to do some complex things, like only run an automation on the last Friday of every month, or only on loans that are more than six months old that have had at least two payments. LoanPro is able to accommodate these and other complex rules without the need to put lenders on hold in order to develop new rule capabilities. And, thanks to Clojure's functional programming paradigm coupled with immutability at the core, complex rules can be run with consistent outputs.

Using Clojure to create complex rules has empowered LoanPro lenders to send the right communications and make loan updates for the right audiences at the right times, increasing collections success and lowering servicing time and collections costs. Employing powerful, configurable automations, LoanPro lenders have increased their loan-to-agent ratio by as much as 300%.

Core simplicity

Clojure provides simple, core functionality. Core functions of Clojure don't often change, with a remarkable track record of code stability and backwards compatibility. This stability translates to stable LoanPro rules that keep working as intended, regardless of system updates. As a standalone language, Clojure may already be familiar, potentially saving significant training time. Lots of documentation already exists for Clojure at the depth required to teach anyone how to create simple or complex rules.

Clojure is REPL-driven, so many training and testing tools are available in addition to simple documentation, which provide a more interactive experience, rules testing, and a community of users to answer questions and offer expertise. These available tools will lower the time and cost of training and help lenders create their own subject matter experts.

Security

Security is a top priority at LoanPro. We maintain SOC 2 Type II, PCI-DSS level one, and ISO-27001 security certifications. When evaluating rules technology, it was extremely important not to introduce vulnerabilities, especially when evaluating untrusted code. Clojure was chosen because custom Clojure code can be executed by LoanPro with a significantly lower risk. Clojure code is stored as text within the LoanPro platform, and sent to a secure interpreter that runs outside of LoanPro's primary servers for execution.

Conclusion

Configurable automations provide a large advantage to LoanPro lenders, by allowing them to quickly go to market with new products and programs, while minimizing training and maximizing efficiency. To make automations as effective as possible, rules that trigger automations must execute quickly, be configurable to meet precise needs, and execute securely. When choosing the rule-creation technology, LoanPro considered other options, including building our own domain-specific language. Clojure was chosen, because it met the needs for security, performance, ease of use, and consistency.