CD.FOUNDATION

# Securing Fidelity Investments' Software Supply Chain

A Platform Approach to Enhancing Continuous Delivery Security Practices

## CHALLENGE

Fidelity Investments, a leader in the financial services industry, was looking for a solution to substantially improve developer experience, reduce risk to the firm, improve feature velocity, and drive greater end-to-end quality and efficiency.

To address this challenge Fidelity first conceded to the enhanced pressure faced by developers in today's world. Developer teams are faced with additional responsibilities pushed onto them including but not limited to:

- Cloud v Data Centre;
- Microservice v Monolith;
- Least Privilege & Separation of Duties;
- Pipeline as Code (Everything as Code);
- Compute Rehydration Policies;
- Open Source Security;

How can you improve feature velocity if you are increasing the amount of overhead to feature development?

How can you substantially improve developer experience if you are adding significant friction to the developer workflow?

How can you reduce risk to the firm or drive greater end-to-end quality and efficiency if you are adding significant new requirements onto development teams in domains where they have no previous experience? Not only do developers have to ensure their code and products are compliant now, they must also worry about their code and products remaining continuously compliant over time.

**CHALLENGES**
Developer Experience, Feature Velocity, Efficiency Scalability, Supply Chain Security

**INDUSTRY**
Financial Services

**LOCATION**
Global

**PUBLISHED**
January 2023

**AUTHOR**
Gerard McMahon, Head of ALM Tools and Platforms at Fidelity Investments

**PROJECTS**

Jenkins cd events

How can you possibly scale within a large enterprise where you have tens of thousands of technologists, managing tens of thousands of applications with hundreds of thousands of automation and Continuous Integration and Continuous Delivery (CI/CD) jobs? How can you manage security and risk, or the technical debt and operational overhead with the level of complexity when each team is responsible for managing all requirements of their own software delivery lifecycle processes?

## SOLUTION

For a large financial services enterprise to scale and meet the requirements of the various regulations, it is essential to automate security and compliance where the policies are built-in as controls across the entire software delivery lifecycle process.

This is why Fidelity is shifting to a platform strategy establishing a “Software Delivery Platform” for enabling application development while simultaneously reducing risk, accelerating velocity, and increasing quality. Additionally, Fidelity wanted to reduce the complexity and the total cost of ownership for delivering software into the hands of their customers. To achieve this goal they want their developers to focus their time building customer features, termed above the ”Business Value Line,” and less of their time on non-business value work, or below the ”Business Value Line,” which they can commoditize via the platform.

The developer experience of any platform is essential for adoption, as the “right way” simply has to be the “easy way.” When discussing the “Software Delivery Platform” concept with a DevOps industry leader, he quoted that this should be an “Invisible Platform” and if you achieve that level of experience, you ensure success in achieving your goals and outcomes.

In addition to its platform strategy, Fidelity is focused on a “Unified Developer Experience” initiative, the objective of which is to design a holistic work experience that enables developer effectiveness and job satisfaction. The initiative includes a series of stages for aligning, learning, analyzing, envisioning and testing including the following:

- Prioritization strategy with near/mid/long-term ideas
- Recommendations for alignment of ongoing and future initiatives
- Business value and technology feasibility assessment

With the shift to a platform strategy and a focus on the developer experience, Fidelity is poised for scalability while meeting customer needs for security and compliance.

The developer experience of any platform is essential for adoption, as the ‘right way’ simply has to be the easy way.

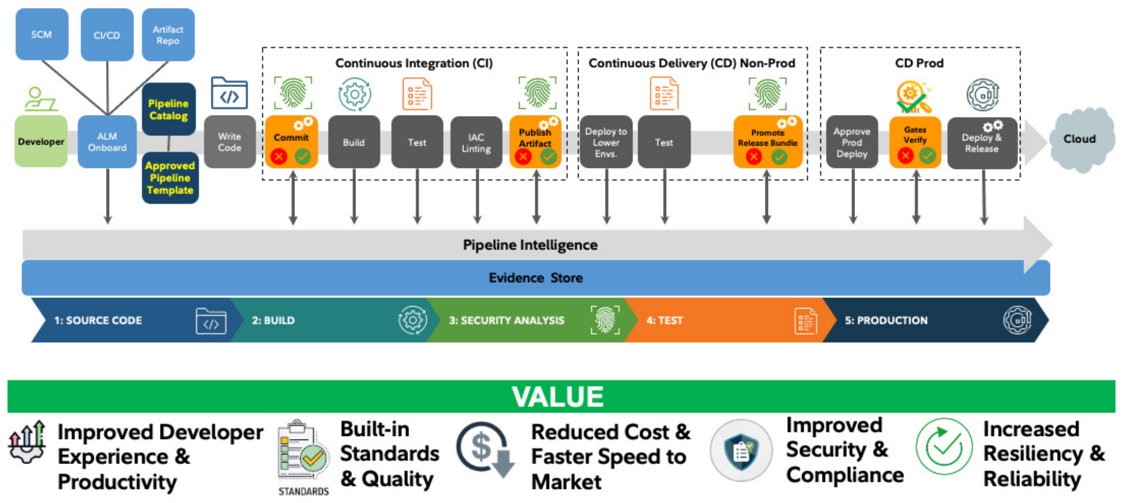
## PLATFORM VS PIPELINE CAPABILITIES

One of the biggest challenges within large organizations is the number of teams all doing their own thing.

This makes it impossible to achieve any level of consistency or to introduce standards due to the difficulty in monitoring or forcing. With every application being a "snowflake," an organization is unable to achieve any economy of scale from the perspective of time, cost, security, and audit. Even if you manage to standardize on enterprise pipelines, these pipelines become very burdensome on time to execute, flexibility, and maintenance along with compromising on the applications team's autonomy, feedback speed, technology adoption, ability to innovate and move fast.

Fidelity's Software Delivery Platform is all about meeting the developer where they are within their developer flow. Moving things to the platform level and removing them from application pipelines means controls can be triggered as part of normal developer actions ensuring the developer is getting immediate feedback from the platform, which is the sought optimal experience. This means things that could cause an application to be non-compliant and introduce a serious risk into the environment (like secret scanning, securing the supply chain, static application security testing, binary scanning, etc.) are blocked at the correct moment with the feedback provided to the developer. Platform-level capabilities have the hooks and integrations centrally provided and managed by the platform team. This helps ensure code and application resources are continuously scanned, which mitigates risk being introduced into the environment and achieves economies of scale.

With Jenkins, CI and CD pipelines can be really simplified and allowed to focus on capabilities that are technology-specific and to be executed on every pull request like application build, unit testing, static code analysis, IAC scanning, functional/performance testing, deployment and so on. Reduced and simplified CI and CD requirements allow for the creation of immutable pipeline templates that can be offered via a pipeline catalog. This allows enterprises to standardize on the Software Delivery Lifecycle (SDLC) process across the organization, embedding automated control gates to bring consistency, and ensure the quality, security, and compliance of changes deployed into production.



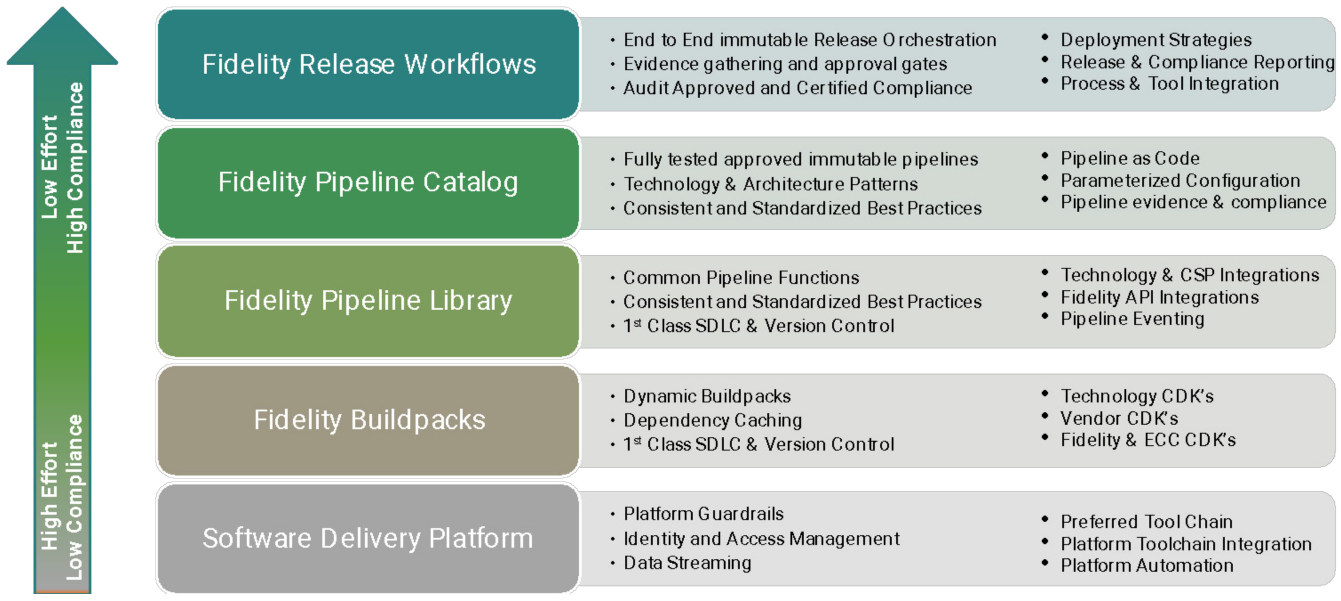
*Enterprise Software Delivery Platform*

## PLATFORM PIPELINE TEMPLATES VS APPLICATION PIPELINES

As mentioned above, moving to immutable pipeline templates at the platform level versus every application having its own individual pipelines, allows Fidelity to standardize on the SDLC process for application build and deployments with all the enterprise controls.

If you want a frictionless experience in any large organization, you can't sacrifice flexibility, technology choice, innovation or speed when standardizing. Additionally, there are many non-standard patterns out there, so there isn't a one-size-fits-all approach, so any solution must be flexible. Fidelity created a layered-cake model for platform pipeline capabilities where teams are free to adopt whichever layer they want. The benefit is that adopting at the highest layers means developers get more out-of-the-box capabilities from the platform with lower effort and higher compliance.

A governed inner-source model at all layers ensures that the developer community is part of creating Fidelity's platform ecosystem, which for all intents and purposes is their working environment. Developers collaborating on how to ensure the highest quality of engineering craftsmanship is key for creating that buy-in, which is critical for ensuring the Software Delivery Platform is successful in enabling developers and providing a safe and secure way to get value into the hands of Fidelity's customers.



Low Effort
High Compliance
High Effort
Low Compliance
Fidelity Release Workflows
• End to End immutable Release Orchestration
• Evidence gathering and approval gates
• Audit Approved and Certified Compliance
• Deployment Strategies
• Release & Compliance Reporting
• Process & Tool Integration
Fidelity Pipeline Catalog
• Fully tested approved immutable pipelines
• Technology & Architecture Patterns
• Consistent and Standardized Best Practices
• Pipeline as Code
• Parameterized Configuration
• Pipeline evidence & compliance
Fidelity Pipeline Library
• Common Pipeline Functions
• Consistent and Standardized Best Practices
• 1st Class SDLC & Version Control
• Technology & CSP Integrations
• Fidelity API Integrations
• Pipeline Eventing
Fidelity Buildpacks
• Dynamic Buildpacks
• Dependency Caching
• 1st Class SDLC & Version Control
• Technology CDK's
• Vendor CDK's
• Fidelity & ECC CDK's
Software Delivery Platform
• Platform Guardrails
• Identity and Access Management
• Data Streaming
• Preferred Tool Chain
• Platform Toolchain Integration
• Platform Automation

*Enterprise Software Delivery Automation Capabilities, Pipelines and Workflow*

## SECURE THE SUPPLY CHAIN WITH DEFENCE IN DEPTH & BREADTH

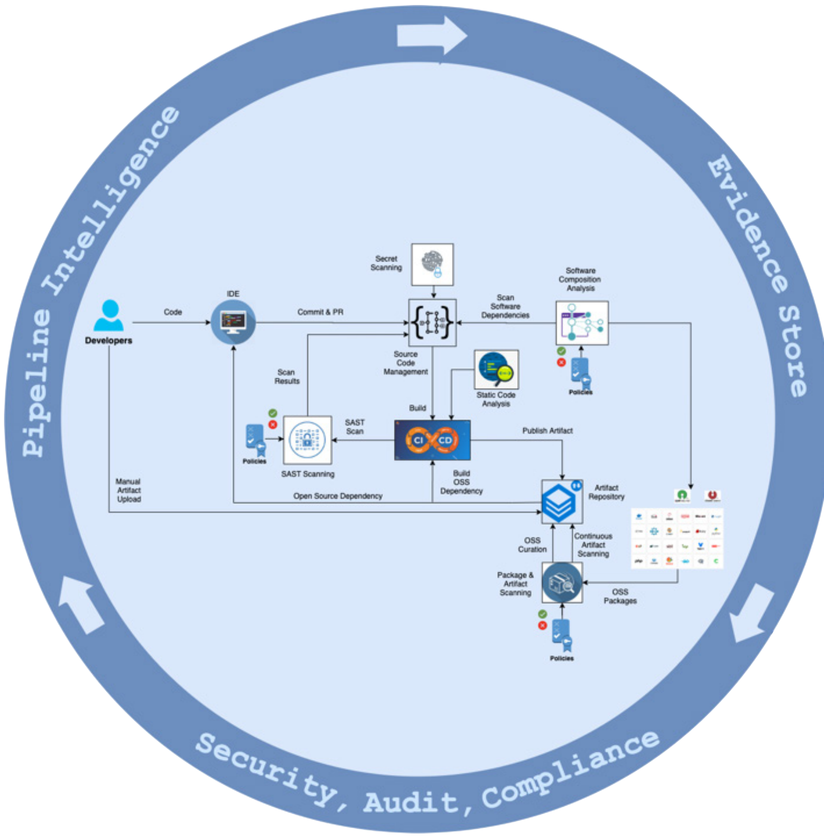
In today's world of increasing software supply chain attacks corporations need security everywhere, requiring both in-depth and in-breadth defences.

External threats come in various forms and through many vectors with malicious code attacks and Common Vulnerabilities and Exposures (CVE). These vulnerabilities aren't just confined to the code but also the developers' laptops, build environments, and the production environment. With so many open source components being used within applications, those dependencies and their transitive dependencies are getting pulled into an application and its infrastructure at build and runtime which can leave an organization open to an attack. Developers aren't security experts. To continuously protect your organization, security experts need to ensure your SDLC process is secure. Fidelity addresses this issue by inserting its security experts at the platform level to add in policies and watches so developers can focus on delivering secure features to their customers.

With the "Software Delivery Platform" Fidelity has incorporated the Secure Software Delivery Lifecycle (SSDLC), where the security capabilities are provided as part of the platform and integrated into the tooling. This means that when a developer requests a third-party or open source dependency, the platform ensures that each dependency is scanned prior to being made available. When a developer is executing normal developer actions like "git push" or "create/merge pull request", the platform can execute security scanning pre and post command executions to ensure compliance and provide feedback in the developer flow when not.

Integrating Software Composition Analysis (SCA) into your Source Code Management (SCM) allows the platform to continuously scan the application code for open source risk management. When a developer builds the application artifact and publishes to the artifact management repository, the platform can scan that artifact before storing it. Along with these capabilities being triggered and executed within the developer flow to ensure no vulnerability enters the environment, those same capabilities can continuously scan the code and artifacts to ensure they remain secure and compliant over time. This workflow can help counter attacks like Log4j, where a once secure and compliant piece of open source software, suddenly wasn't.

By enabling security at the platform level, the security experts can create the security policies and controls within the Software Delivery Platform tooling ecosystem and every developer and team then inherits that security expertise for ensuring their applications are secure and compliant.



## DATA AND CONTINUOUS COMPLIANCE AT SCALE

We discussed the many challenges developers face as part of managing an application in the cloud.

Adding the requirement to be continuously compliant with security, engineering, quality, internal audit, and regulatory policies adds to that challenge. Without an automated platform, that exhaustion is extended across the enterprise into other practices like security, risk, audit, etc.

How can an organization scale compliance, if compliance is based only on the trust of people doing things the right way in a compliant manner?

There is very limited "verify" and an overreliance on "trust" within organizations when it comes to compliance. A reliance on periodic, manual audits to spot-check applications and share any findings back for other teams to then self-check against. This process doesn't scale, as humans don't scale and it creates friction within the system. For those teams involved in the audits, it is invasive and time-consuming. Also, any findings identified can be very disruptive to the teams, due to the delay in the feedback provided and the requirement to immediately remediate.

There is a huge amount of valuable data created during the SSDLC process and within the tooling, that is largely siloed, inaccessible, and untapped. As part of the Fidelity Software Delivery Platform, they have created a data strategy around creating and harnessing data for piping into a home-built software delivery intelligence system and storing it in an evidence store. This strategy allows Fidelity to capture all data related to development, CI/CD, security, artifacts, etc. along with identity and access data of who, did what and when. The platform utilizes the data in real-time in many contexts during the SSDLC process. Automated control gates are used to govern compliance with policy and provide feedback. The data can additionally be used in a "trust but verify" model for gating at specific points within the software delivery process to reduce/eliminate human and digital waste.

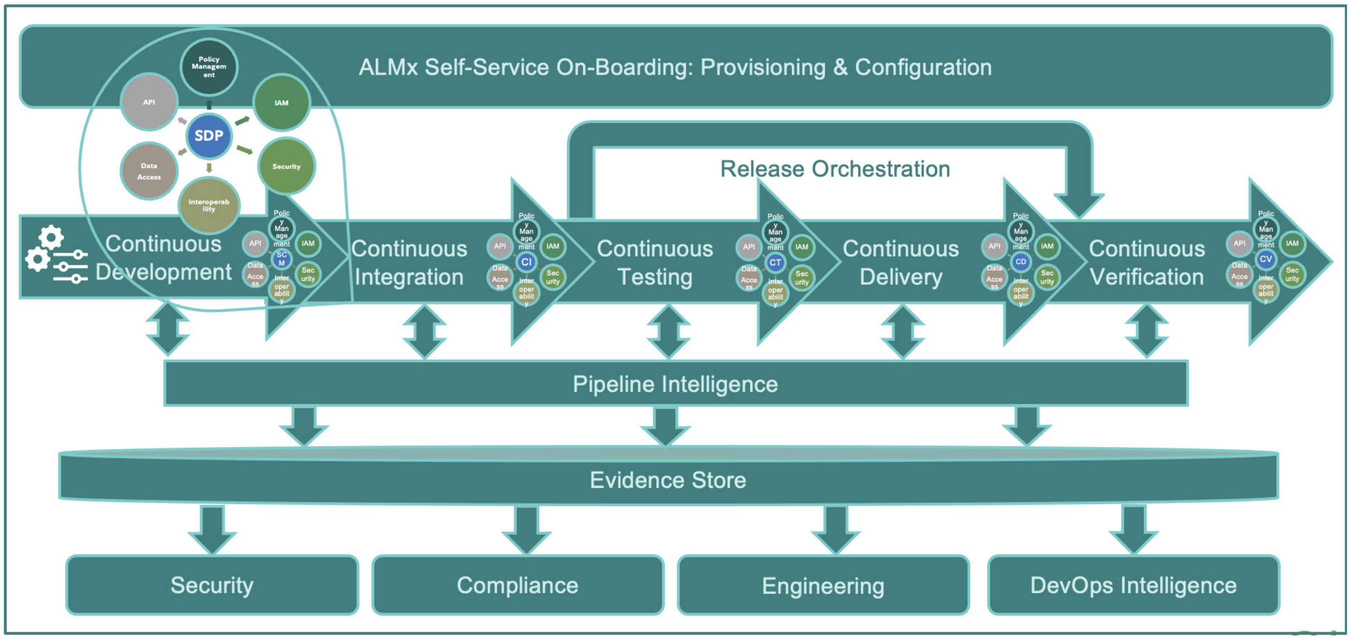
As part of the strategy, Fidelity is going to leverage and contribute to the CD Foundation's CDEvents project, a common specification for Continuous Delivery events. Fidelity will contribute a CDEvents Plugin for Jenkins to enable interoperability of Jenkins with other tools in the Software Delivery ecosystem, making it easier for developers to include Jenkins and their CI/CD pipelines. This will allow Jenkins to send and receive CDEvents and chain Jenkins with other tools for end-to-end Software Delivery, opening up additional opportunities for organizations to scale their CI/CD systems.

Thanks to this process, Fidelity will have an inventory of application technology along with all the changes ever made to an application for use in continuous compliance and data analytics for risk exposure when a threat on an existing piece of software is identified. The data enables compliance dashboards to be created, giving compliance and risk views of applications at an individual level or rolled up at different organization levels.

This strategy allows Fidelity to capture all data related to development, CI/CD, security, artifacts, etc. along with identity and access data of who, did what and when.

Fidelity can scale audits and risk by having automated continuous compliance across the entire firm, significantly reducing the effort and time involved, thus reducing the challenges on the developer and experience.

On top of all that, there is the added bonus of the data strategy. As mentioned, the data can be used in many different contexts within the SSDLC process. It can also be used in other aligned/complimentary contexts such as DevOps insights or engineering excellence and allow them to look at intelligent automation in the future.



*Software Delivery Platform Overview*

Fidelity can scale audits and risk by having automated continuous compliance across the entire firm, significantly reducing the effort and time involved, thus reducing the challenges on the developer and experience.

## SUMMARY

Fidelity wanted to solve an issue that affects many enterprise organizations: how to improve developer experience, reduce risk, improve feature velocity and drive greater end-to-end quality and efficiency at scale. The solution - Fidelity Software Delivery Platform.

The Fidelity "Software Delivery Platform" will provide developers and teams with the ability to focus on their Business Value "above the line" and allow the platform to bring that business value code all the way to production with minimum friction. The platform can then take care of the software delivery manufacturing process and non-business value "below the line" ensuring all the relevant build, quality, security and compliance capabilities are executed. Additionally, the "Software Delivery Platform" will ensure the relevant control gates are executed to ensure continuous compliance along the SDLC process and to also provide fast feedback to the developer within their flow.

CD.FOUNDATION