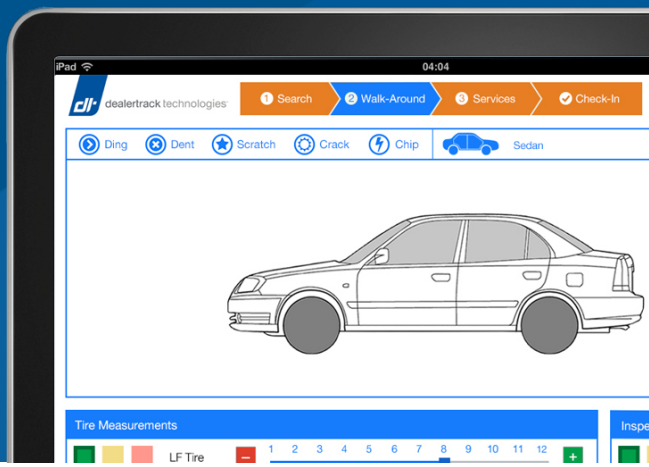
Customer story / media

How Dealertrack Keeps an Error Free Car Registration Process

OverOps helps Dealertrack's focus on advancing the product roadmap instead of debugging production issues



Dealertrack Registration and Title Solutions, a division of Cox Automotive, offers vehicle registration and title services for dealers and the overall automotive ecosystem worldwide. The division has over 30k users across 12 states that process more than 10k deals per day, handled by 5 Agile Scrum teams across 3 states.

60K	22K+	$5B	200
Employees	U.S auto dealers	Revenue	Locations

Production Monitoring Ecosystem: OverOps splunk> Nagios dynatrace

Highlights

- With OverOps, Dealertrack's engineers can focus on advancing the product roadmap deadlines instead of debugging
- OverOps helps Dealertrack handle application exceptions in a 10+ year old legacy codebase
- Thanks to OverOps, the company increased its application reliability and delivered an improved user experience
- OverOps answers "why" code breaks, allowing the Dealertrack team to identify critical errors and fix them quickly

Key challenges and pain points

Our primary pain point is the number of edge cases our users face – especially when a significant number of these edge cases happen either within our user interface or with data we send to various DMV offices. Since every state DMV has completely different vehicle registration rules and requirements, our system needs to change and adapt accordingly. Users face these edge cases when DMVs release changes to their system, resulting in exceptions thrown by our own code.
These could include, for example, differences in data required to be communicated to DMV systems.

Tel: 1-800-324-3053 www.overops.com
797 Bryant Street, San Francisco, CA 94107

This could, in a worst-case scenario, result in a transaction not being completed – and a vehicle not being registered – in a timely manner.

"OverOps helped our engineers save time and effort when searching, identifying and trying to understand the root cause for each issue."

Additionally, we handle a significant amount of code change – with our 5 Agile Scrum teams actively managing changes to 6 different products at any given time, all with feature overlap, and all occurring over several releases each month. Once these products reach production, our IT and Ops teams monitor the log files and detect critical issues. This leads to an almost endless number of logs. Only after an issue has occurred in production, our engineering team jumps in, but they too have to go through the log files to try and reproduce the issue at hand. Wading through the log was time-consuming, and there's never enough detail to reproduce the exact situation that failed. This means our team spends most of their time solving bugs, instead of working on new features.

"OverOps gave us immediate value as soon as we installed it, showing us exceptions thrown from within our code that we weren't aware of."

Example problem that OverOps helped resolve:

Before OverOps, our debugging process consisted of asking users about the time and day when the processing error was encountered. Also, we had no way to know if an error was caused by a new or existing version, and tracing errors back to the root cause would mean digging through source code and commits trying to understand when and how our code changed.

OverOps gave us immediate value, and presented us with exceptions we weren't aware of. It helps us identify errors and exceptions as soon as they occur, allowing us to track back a specific error and fix it within minutes. It also alerts us if this error happened in new or existing code, and provides the context in which it happens.

We can immediately reproduce it, fix it, and deploy a new version - saving valuable time, effort and money. The biggest value we see from OverOps is staff efficiency, helping our developers be more productive and spend more time on building instead of debugging.

Additionally, OverOps helps us handle application exceptions better with our 10+ year old legacy codebase. The ability to show exceptions - even if they are uncaught, or don't appear in the logs - is very valuable.

"The biggest value we see from OverOps is staff efficiency, helping our developers be more productive and spend their time on building instead of debugging."

How are you integrating OverOps with your daily workflow?

We use the OverOps Slack integration to receive reports about newly-encountered application issues. We've had several cases in which our engineers weren't aware of some of the exceptions since they either failed silently, or failures were hidden in the logs. Now with OverOps, we can stay on top of every new error that's introduced into our environment.

Tel: 1-800-324-3053 www.overops.com
797 Bryant Street, San Francisco, CA 94107

Full code and variable state to immediately reproduce any error.

No need to manually reproduce issues by searching for information in logs. Reduce MTTI by 90%+

OverOps operates between the JVM and processor level enabling it to run in staging and production.

Proactive detection of all new and Critical errors

New issues are detected and routed to the right developer vs. discovered by users. Each error receives a unique code fingerprint unique it across the app.

No change to code or build

New issues are detected and routed to the right developer vs. discovered by users. Each error receives a unique code fingerprint unique it across the app.

Supported Platforms:

JDK 1.6 and above | HotSpot, OpenJDK, IBM JVM | Java, Scala, Clojure, Groovy | Linux, OS X, Windows | Docker, Chef, Puppet, Ansible | Coming soon: .NET

Integrations:

SLF4J, Log4j, Logback, Apache Commons Logging, Java Logger | Splunk, ELK, SumoLogic, and any other log management tool | AppDynamics, New Relic, Dynatrace | Workflow automation: Slack, HipChat, JIRA, Pagerduty | Webhooks | StatsD

Learn how OverOps can help you automate your deployments -
Schedule a demo with an OverOps monitoring engineer

Tel: 1-800-324-3053 www.overops.com
797 Bryant Street, San Francisco, CA 94107