vFunction

vFunction 3.6.0 Release Notes

Key Additions

- Distributed Applications

  The Distributed Applications introduces a new feature for visualizing and understanding the architecture of distributed applications composed of multiple microservices. At its core is the Service Map Graph, an interactive visualization that represents each service as a node, with edges depicting dependencies, API calls, and shared resources. This graph provides comprehensive details about individual services, including name, type, tech stack, and hosting information.

  Architects can create and configure distributed application entities, set learning periods for data gathering, detect drifts with automatically generated TODOs, trace specific service flows, and seamlessly connect to monolithic application views. Additionally, users can optimize their architectures through manual TODOs for breaking dependencies, merging services, and introducing new interactions. With enhanced filtering, search, and visual customization options, this release empowers users to gain deep insights into the intricate structures and interactions within their distributed applications. Learn More.

Architecture Observability Improvements

- Distributed Applications

  Distributed Application Creation and Configuration

  Users can now create and configure new distributed application entities within the platform. This process includes specifying details such as the application's name, the APM provider, authentication keys, and criteria for including or excluding specific services.

  Service Map Graph Visualization - Distributed applications introduces a new visualization tool for distributed applications, providing a comprehensive and intuitive representation of the services comprising the application, their interactions, and dependencies. The Service Map Graph presents a node-based diagram, with each node representing a service and edges illustrating the connections and resource sharing between them.

  Learning - Similar to monolithic applications, users can define learning periods for their distributed applications, allowing the system to gather necessary data and trace information. These learning periods can be scheduled or initiated immediately, resulting in loaded measurements that capture the application's current state.

  Drift TODOs - vFunction monitors drift between loaded measurements for distributed applications, automatically generating relevant TODOs to notify users of changes or

deviations. These TODOs cover a range of scenarios, including new services, service dependencies, resource exclusivity changes, circular traces, and multi-hop traces.

Manual TODOs - In addition to automatically generated TODOs, users can create manual TODOs to optimize their distributed application's architecture. These include the ability to break service dependencies, merge services, and introduce new service dependencies, providing greater control over the application's structure.

Flow Tracing - The new flow tracing feature enables users to visualize and analyze specific flows through the services within their distributed application, allowing for a deeper understanding of service interactions and potential performance bottlenecks.

Analysis Improvements

- Compile-Time Dependencies for Common Classes

  In this release, we have addressed the need for increased visibility and control when removing classes from the common library. Previously, users encountered messages indicating dependencies with common packages, but lacked specific details about the compile-time dependencies between the class and the common package. This enhancement provides users with more granular information, enabling informed decisions and appropriate actions.

- Highlight Pure Static JARs

  The JARs Graph is a crucial tool for visualizing and managing JARs across different applications. While runtime classes are important during execution, they are not present at compile time, making them dynamic in nature. The ability to identify JARs that do not contain these runtime (dynamic) classes is essential for separating infrastructure-related JARs from others, helping the management process.

- Indication When Static Analysis Has No Classes

  When vFunction is unable to detect any classes during the static analysis process, an indication is now displayed on the analysis page. This message communicates that no classes were detected, providing users with a clear understanding of the underlying issue, and providing guidance on the necessary steps to resolve the issue.

- Export Analysis Data

  This feature allows users to export analysis data from vFunction. This enhancement addresses the need for collaboration and sharing detailed analysis information with others who may not have direct access to the application.

  Export All Domains - Users now have the ability to download a file containing detailed analysis data for all Domains within their application. This file will be a zipped YAML format.

vFunction

Export Specific Domain - In addition to exporting all Domains, users can export the analysis data for a specific Domain. This feature provides flexibility and allows users to share details about a particular Domain of interest.

The exported files will contain the following information: Domain (Entry Points, Dynamic Classes, Static Classes, Dead Code), Common classes (Dynamic Classes, Static Classes) and Application-level information (Dead Code).

- Enhanced Stored Procedure Tracking for Microsoft SQL Server
  Whenever a stored procedure is called, also for MS SQL Server all relevant DB tables will be marked as accessed, and not only the stored procedure itself. Enhanced stored procedures support is now available both for Java and for .NET applications.

Usability Improvements

- Deep Linking in All Screen States

  In this release, we have introduced a new feature that enables deep linking for all screen states within our application. Deep linking allows users to generate unique URLs that represent specific combinations of filters, tabs, and views applied on any screen. These URLs can be bookmarked, shared, or accessed directly, instantly restoring the application to the exact state represented by the URL. By implementing deep linking for all screen states, we aim to improve the user experience, collaboration and information sharing, and provide a more seamless and intuitive navigation experience within our application.

- Class Method Categorization for Refactoring

  Introduced a new feature that provides enhanced visibility and categorization of class methods, specifically designed to help users in the refactoring process. By offering a clear distinction between exclusive and non-exclusive methods, along with additional contextual information, users can now make informed decisions and efficiently refactor classes to improve code organization and maintainability.

.Net Improvements

- Support Stored Procedures in .Net

  Introduced a visibility of stored procedures and their associated database tables in .NET applications. This functionality enables users to gain deeper insights into the dependencies

and relationships between stored procedures and the tables they interact with, facilitating better understanding and maintainability of their applications.

 www.vfunction.com To learn more about vFunction Inc. click here.