vFunction
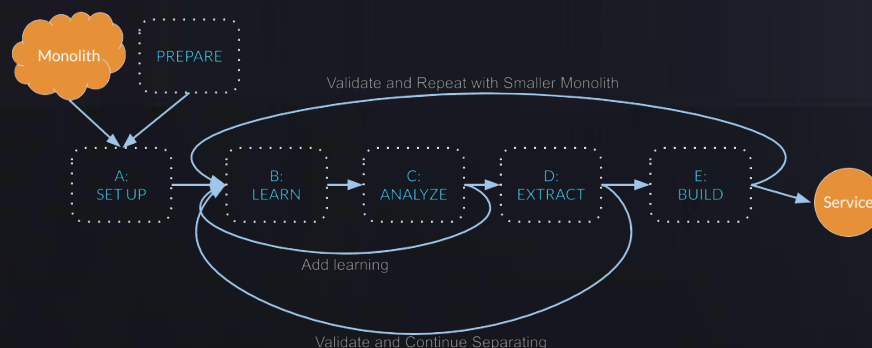
Installation Questionnaire / Checklist

November 2021

Introduction

One of the biggest challenges facing IT organizations is what to do with their legacy enterprise applications. These applications have evolved over time, changed ownership through generations of developers, and often end up as a "ball-of-mud" monolith. The difficulty in maintaining these applications and supporting the business need for scale, agility, and innovation is driving the journey to modernize legacy application assets. Modern architectures employ a pattern of microservices or functions that enable much higher development velocity with separate teams working independently on separate services, as well as much better utilization of infrastructure by using containers vs. virtual machines.

vFunction is the first and only solution explicitly built to automatically, efficiently, and rapidly transform monolithic apps into microservices. Unlike the previous generation of older, slower generic tools and processes, vFunction is a Cloud Native Modernization platform that is purpose-built for today's enterprise challenges with intelligent, fast microservice extraction and creation. vFunction allows architects and developers to see interdependencies amongst services to not only facilitate decomposition but also automatically package the resulting new services by exposing required Spring context, EJBs, and other dependencies, downloading directories with templates as a basis for the code structure of the extracted service, and integrating with source repos, and build systems, to actually create the project of the extracted services. The result is container-ready code, which is built as separate, API-connected services. vFunction is easy to install, configure, and deploy and starts returning rich results in an interactive, navigable console interface within minutes, all the way to extracted, automatically-created, and compiled microservices within days.

The Journey at a Glance

The following diagram provides a top-level view of the modernization journey you take with vFunction. The Journey begins with identifying and **preparing** an existing monolithic application and a target deployment environment in which to run the resulting services.

vFunction

Application Questionnaire

Please review the following checklist and answer where there is an information requirement
In order to provide you with the right vFunction agent and server please fill out the below:

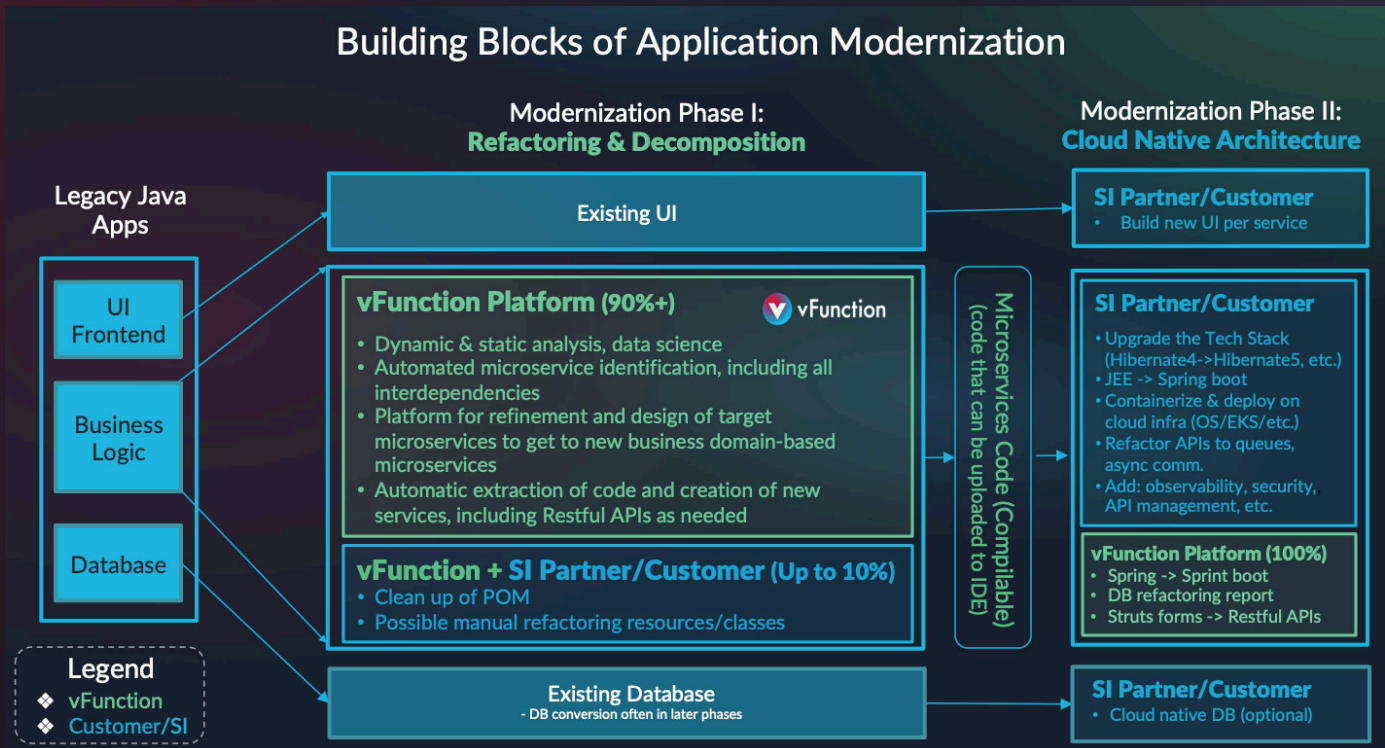|	Item	Yes/No/Information
1.	Identify the monolithic application you want to analyze and from which you want to start extracting services	No [ ] Yes [ ] Name of the Application:
	Name the key business functions of the application:	Information:
	Have you tried to modernize this application in the past?	No [ ] Yes [ ] Provide details:
	Do you have an architectural document for the app?	No [ ] Yes [ ] Can it be shared?
	Do you have an estimation of time and resources needed to modernize this application without vFunction? How many lines of code (LOC) or classes does this application contain?	
2.	What JVM and which version are used?	Information:
	Where do you want to get to after the modernization (versions)?	Information:
3.	What application server and which version are used today?	Information:
	Where do you want to get to after the modernization (versions)?	Information:
3.	What database is being used?	Information:
4.	On what operating system is the JVM running on? Do you have superuser/root access on that machine?	Information:
5.	Is the application using Spring? If so, which version and what features are used in the application? How is Spring bootstrapped to the application?	Information:
	Are you looking to migrate to a different Spring version? To which one?	Information:
6.	Is the application using EJBs?	
	Are you looking to use EJBs post refactoring?	
7.	What are the main frameworks being used by the application?	

	Which persistence framework is in use (Hibernate, EclipseLink, MyBatis, etc.)? Which versions?	
	Are you looking to migrate to a differnet version after the modernization? To which one?	
8.	<u>For non-production installations</u>: Make sure you have the application installed in a pre-production environment (highly recommended 4GB of memory, and 10GB of free disk space) ?	Yes [ ] No [ ]
9.	<u>For non-production installations</u>: Make sure you have automatic testing ready for the application or manual QA resources that can test the application flows (more coverage the better)	Yes [ ] No [ ]
10.	What is your build system? What is your dependency management system? (Gradle, Maven, Ant/Ivy, etc.)	Information:
11.	What is your target microservices/functions environment? (Openshift/K8s, Knative, other) which versions?	Information:
12.	How many services are you looking to extract?	Information:

Once we receive the information for the above items we will be able to send you the correct vFunction agent and server artifacts as well as the licensing key.

POV Scope

vFunction POV covers the installation and initial assessment of the application, with vFunction personnel to assist in this process. The culmination of the POV occurs once services are identified and a refactoring plan is generated. Subsequent phases, post-POV, include the extraction of services, upgrading, re-platforming, etc (see chart below):

Building Blocks of Application Modernization

Modernization Phase I:
Refactoring & Decomposition

Modernization Phase II:
Cloud Native Architecture

Legacy Java Apps

- UI Frontend
- Business Logic
- Database

Existing UI

SI Partner/Customer
- Build new UI per service

vFunction Platform (90%+) vFunction
- Dynamic & static analysis, data science
- Automated microservice identification, including all interdependencies
- Platform for refinement and design of target microservices to get to new business domain-based microservices
- Automatic extraction of code and creation of new services, including Restful APIs as needed

vFunction + SI Partner/Customer (Up to 10%)
- Clean up of POM
- Possible manual refactoring resources/classes

Microservices Code (Compilable)
(code that can be uploaded to IDE)

SI Partner/Customer
- Upgrade the Tech Stack (Hibernate4->Hibernate5, etc.)
- JEE -> Spring boot
- Containerize & deploy on cloud infra (OS/EKS/etc.)
- Refactor APIs to queues, async comm.
- Add: observability, security, API management, etc.

vFunction Platform (100%)
- Spring -> Sprint boot
- DB refactoring report
- Struts forms -> Restful APIs

Legend
- ❖ vFunction
- ❖ Customer/SI

Existing Database
- DB conversion often in later phases

SI Partner/Customer
- Cloud native DB (optional)

Name:	
Company:	
Partner: (if applicable)	
Role:	
Email:	
Date:	