DeployHub

Migrating to Microservice using DeployHub - Step 1: Define Your Domains

Steve Taylor, CTO DeployHub, Inc.

Getting Started

Organizing your reusable components and services is key to a successful implementation of a service-based architecture. Because it is so key, it should be the first step you take in your modern architecture journey. This is often called *Domain Driven Design (DDD)*. The good news is that you can start the process by looking at your current monolithic application in terms of components and *Domains*. From that perspective you can slowly and successfully move to microservices.

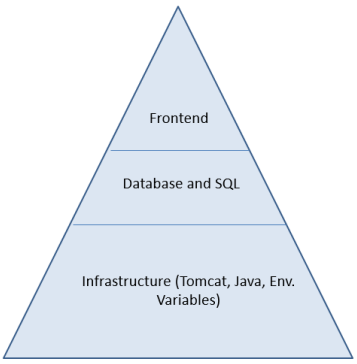
DeployHub supports both monolithic and microservice releases. It facilitates the move to microservices by leveraging the use of Domains. Domains are critical when you begin to decompose functions into independently deployable services.

To understand this first step, lets break down a monolithic into components. If we take a simple web store application, you may have a `.jar` file, the database, the infrastructure components and environment variable settings. While the `.jar` file may be your primary focus, you need to start seeing the database, infrastructure and environment variables as independent components of your overall application. You can build a strategy for managing each of these layers as individual services. DeployHub does this using *Domains*. Domains track, catalog and expose reusable components allowing them to be easily shared, a critical step in the shift to microservices.

Your monolithic Domains may include:

- Infrastructure Components (java runtime, Tomcat, Environment. Variables)
- Database and SQL (.sql)
- The Website (.jar)

To visualize this, we can use a triangle that shows the lowest to highest level of dependencies. This is a simple organizational method that can be useful for our more complex service-based architectures.



Monolithic Domains

With DeployHub, these Domains are controlled by different *User Groups* and can contain lower level *Sub-Domains*. For example, Operation Teams may control the Infrastructure Domain, DBAs the middle tier and developers control the website frontend. As you break down your application into services, you will expand on these Domains. Starting at this monolithic level helps you begin thinking 'functions' and prepares you for the next step.

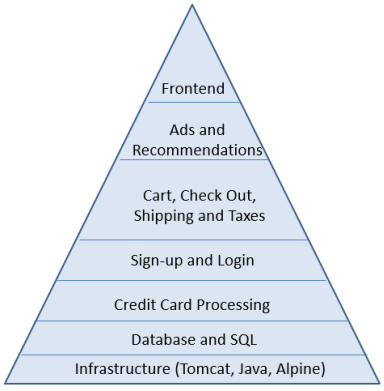
Working with Domains and Services

One of the **goals for Domain Driven Design is taking an application and breaking it down into its smaller parts.** We then organize those parts in a fashion that will allow reuse.

If we think about the potential microservices for a simple store website, we'll have:

- the website frontend,
- a cart,
- credit card processing,
- a check out system,
- advertisements and recommendations,
- sign-up,
- login,
- shipping,
- taxes.

We start by expanding the triangle. Think about what's most common. Like our monolithic, the lowest level is infrastructure and database. In our service-based architecture, these are followed by credit card processing and the login/sign up. Above that, we may have the cart and on top of that are the more specific pieces such as the ads and recommendations. At the peak of the triangle is the frontend.

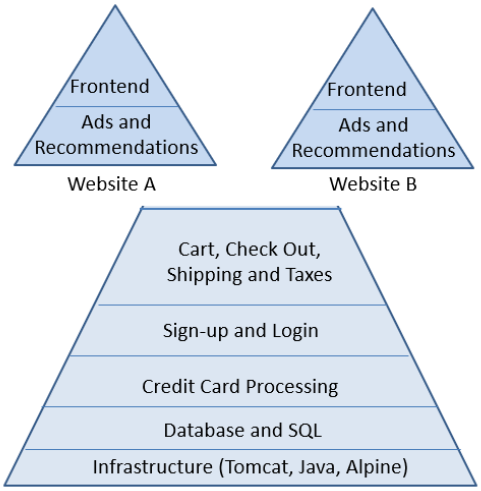


Website A

This triangle shows how we build up from the most common pieces and represents how we would define Domains. Each section of our triangle becomes a Domain.

Domains and Sharing

Say we have another team that's writing a store website, Website B. With DeployHub they find common pieces based on the Domains. Once you define your Domains, your developers can begin publishing and deploying microservices under the Domain structure making it possible for other teams to easily find and then reuse the services. When our next team creates a new store, they begin the process of re-use.



Website A and B

Between Website A and B we have accomplished a level of reuse for all the shareable layers from the infrastructure to the cart. This allows us to write the login, sign up, the cart, and the credit card processing once. DeployHub facilitates this level of design by supporting a highly customizable Domain structure allowing you to define what your triangle looks like and then Domains make it easy to find and share the microservices.

The OOPS in Object-Oriented Programming – Let's Not Repeat that Mistake

Object Oriented Programming (OOPS) attempted to achieve a similar level of reuse. Part of the problem of OOPS was the inability for teams to accurately run builds where linking was carefully managed from a central 'common' location. To solve this problem, developers would add the 'common' object, like our login service, to their local repository so the build could use it. Their build would always reference that version of the library. Updates to the library would be private. There would be little, if any, contribution of updates to the 'common' version. This mistake could be easily repeated in microservices. That would be unfortunate and represent a failure in the implementation of a service-based architecture.

The question always is 'how backward compatible is the new versions?' Are we breaking any signatures on the methods? If we're not breaking any of the interfaces, then can we consume those latest versions? And which application is consuming what version of a service?

For example, website B is going to use v2 of the login and website A will use v3. We need to track which Website is using which version. If you want to deprecate v2, it will require Website B move to v3. What is needed is a method for impact analysis to accurately make these types of decisions. This tracking ability is key to a successful microservice implementation and part of DeployHub's Domain dependency management features. DeployHub provides this level of tracking, versioning and reporting to make every microservice deployment successful and visible for high frequency updates.

Get Started with DeployHub

To begin managing your reusable components and versioning your configurations, get started by signing up to use DeployHub Team for Free. DeployHub Team is based on the open source Ortelius.io project and is provided as a hosted offering to high performing development teams who want to start managing Services in the new modern architecture.

More information

DeployHub Team Sign-up at DeployHub.com
The hosted team version can be used to deploy to unlimited endpoints by unlimited users.

Video of this Presentation
See a video of Steve presenting the concepts of Domains.

Versioning Microservices
This Whitepaper furthers the discussion on Domains, Applications, Versions and microservices.

Get Involved in the OS Project. Help us create the best, open source microservice sharing platform available at ortelius.io.

About DeployHub

We empower development teams with hosted next generation software deployments, specifically designed for the new Agile and microservices environment.

Find us at:
DeployHub.com