CTO Highlights: Patterns for Infrastructure Modernization.

Prepared by LunchBadger

Introduction

When planning an approach to modernizing legacy applications, there are alternatives to a full re-write and cut over. Each approach has it's own implications for the business and developer resources.

A rewrite means higher risk and longer time to value. This can tax your development resources and slow down engineers who could spend that time delivering value to customers. Less agile, slower velocity. It's just not always a practical approach.

A more compelling approach is to leverage the “strangler” pattern. Despite it's bizarre name (which we can totally relate to at LunchBadger), this modernization pattern for infrastructure development was introduced by Martin Fowler in 2004. Fowler was interested in how developers can give value steadily through frequent releases where engineers could monitor progress more carefully.

The Strangler Pattern is an application design pattern that transforms your monolithic application into microservices at an incremental level.

This pattern replaces a functionality with a new service and works as follows:

- New functionality is ready.
- Old, out dated component is "strangled."
- New service is put into production.
- Old component is decommissioned altogether.

We've put together this article to discuss important aspects of this modernization pattern and how you can put new technology to work achieving the core principles behind Martin Fowler's "strangler" pattern.

When it comes to modernization, some of the most prominent concerns are cost and application design. How can a business reduce costs by becoming more efficient? Additionally, what are the implications on core application design and microservices? Let's talk!

Concerns on Modernization Cost

> "Many people still don't consider a strangler since they think it will cost more - I'm not convinced about that. Since you can use shorter release cycles with a strangler you can avoid a lot of the unnecessary features that cut over rewrites often generate." - Martin Fowler

In Fowler's detailed explanation, he reasons that because you can use shorter release cycles with a strangler pattern, you can avoid unnecessary features. These features are just the result of a cut over rewrite. Shortening the release cycle also helps to keep your agility and velocity while developing applications.

Application Design & Microservices

> "Reducing the complexity of an application enables you to deliver business features faster. It also allows you to scale your application based on increasing load. Having an automated CI/CD pipeline makes it a lot easier to deploy the microservices and can make the transition from monolith to microservices much smoother." - Samir Behara

Applying the strangler pattern involves writing new pieces of functionality surrounding the monolithic application. Using a gateway, calls can be routed to the new pieces rather than the legacy code, over time, strangling the legacy with newly developed pieces. Microservices provide an ideal development paradigm for implementing the strangler pattern. Microservices help with breaking down application to micro, autonomous parts as well as promoting the separation of concerns principle to encapsulate functions of the application. Combined with an automated CI/CD pipeline, a microservices approach provides a high velocity capability and value delivery environment versus a monolithic application rewrite approach.

How does LunchBadger's Express Serverless Platform (ESP) deliver these capabilities and support the use of modernization patterns like Martin Fowler's "strangler" pattern? Stay tuned for an active case study or, you can try Express Serverless Platform free for 14 days.