**Discussion Topics: Low-Risk Releases**

Select one of the following and find at least one article on the topic.

1. Explain Strangler Application Pattern. Are there any disadvantages to incorporating this technique?
2. What is meant by the phrase Monoliths vs. Microservices? When might each be used?
3. Explain the difference between tightly-coupled and loosely-coupled architecture, and provide examples of each. When might each be used?

***Before you submit your thread, put your name in the subject line.***

For this discussion, I will be focusing on the Strangler Application Pattern. In my research, there were many articles online about the topic. The two I decided to focus on are by Jose Sousa from the Medium website and an article from the GeeksforGeeks website.

The name of the strangler pattern can be linked back to the strangler fig plant since it would strangle and replace a host tree (Sousa, 2023). The point of the Strangler Pattern in software development is to update and modernize legacy systems (Sousa, 2023). When a part of an application is aging, updating it to become more compatible with new technological elements and advancements is the best move (Sousa, 2023). Sticking with legacy systems poses additional risk and complexity, so slowly phasing these out using the Strangler Pattern is less risky and overwhelming (Sousa, 2023). In most cases, shutting down an entire system to work on bugs is ineffective for users and can lead to financial loss or other disruptions (Sousa, 2023). More resources, such as time and money, are also needed to fully rewrite a system (Sousa, 2023). There are a few steps to follow when using the Strangler Pattern: transform, co-exist, and eliminate (GeeksforGeeks, 2023).

The Strangler Pattern has its own advantages and disadvantages. Some advantages include incremental migration to make changes slowly, flexibility for updates, and coexistence (GeeksforGeeks, 2023). A few notable disadvantages are the complexity of introducing a migration process, data consistency when synchronizing it, increased network calls harming performance, and dependency magnetism between components (GeeksforGeeks, 2023).

**References**

GeeksforGeeks. (2023, June). *Strangler Pattern in Microservices | System Design*. GeeksforGeeks. https://www.geeksforgeeks.org/system-design/strangler-pattern-in-micro-services-system-design/

Sousa, J. (2023, August). *The Strangler Pattern: Kill Legacy Like a Boss - José Sousa - Medium*. Medium. https://medium.com/@josesousa8/the-strangler-pattern-kill-legacy-like-a-boss-db3db41564ed

**Assignment Requirements and Grading:**

1. An initial post of approximately 250 words is due by **Thursday, 11:59 p.m., CST**.
2. For the initial post to be considered substantive, it should be at least 250 words in length and fully cover the topics being presented. Single-sentence definitions or responses will not be awarded points.
3. Submit your post by clicking on the **Assignment Link** above, then **Create Thread**. You must create a thread in order to view your peers' posts. Tip: Create your post in a Word document and then copy and paste your work into the thread.
4. A minimum of three (3) responses, **to the original threads of other students**, of 100-200 words each are due by **Sunday, 11:59 p.m., CST**.
5. To view the rubric grading criteria, click on the following link: [Discussion Board Grading Rubric](https://content.bellevue.edu/cst/csd/rubricdbv3.pdf).

**(50 points)**

Hey, Brett! I really enjoyed reading your post for this module. I think you did a fantastic job covering monolithic structures, compared to microservices. I really enjoyed how you included a graphic for monolithic architectures and microservices and how the two compare. It helps to visualize the progress and provides an even deeper understanding of the concepts. If you were leading a development team and had to choose between the two, which would you go with? Even if the project and scale are smaller, if there is a large team, the resources, and money to go with microservices, I would choose that!

Hi, Colton! After reading your post for this module, I think you did a great job of covering how both monoliths and microservices approaches differ from one another. I agree with your assessment that the monolithic architectural approach is best suited for smaller teams and projects. I also agree that microservices are best for large-scale projects and companies. Once the scale of a project increases, sticking with or choosing a monolith system leads to problems in the future. Legacy code runs through a monolithic application, so as better and more efficient ways to code are released, a monolithic system instantly ages.

Hey, Nima! I think you did an excellent job describing the ins and outs of the Strangler Application Pattern. I also chose to cover the same topic for my post, using the same sources from the GeeksforGeeks and Medium websites. I found it particularly interesting how the strangler fig vine led to the name of this application pattern, but it is very fitting! When a company or developer chooses the monolith or legacy system route, slowly transitioning out of that is extremely helpful and improves the functionality of code without the frustration, cost, and time of doing it all at once.