**Discussion Topics: Integration and Releases**

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

1. **Explain Trunk-Based development, include how it compares to feature branches. Are there any disadvantages to incorporating Trunk-Based development?**
2. What is meant by the phrase Decouple Deployments from Releases? Is it even possible to do do? Any reason not to do so?
3. Select one case study from either Chapter 11 or Chapter 12 and provide a summary of what happened and why. Include your observations; was the change warranted? Costly? Timely?

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

For this module discussion post, I will cover Trunk-Based development, how it compares to feature branches, and any disadvantages. When digging deeper into Trunk-Based development, I found many articles, but I decided to focus on one from Sabri Mutluçağ on the Medium website.

Trunk-based development involves an all-hands-on-deck approach, tackling the same codebase called the trunk (Zettler, n.d.). When a change is made within the trunk, all developers can access it, increasing the frequency of code changes with smaller code being deployed (Zettler, n.d.). Trunk-based development stands out compared to other practices because the code changes are more minor yet more often (Zettler, n.d.). Trunk-based development is a highly collaborative environment for developers since everyone can see changes as soon as they are created (Zettler, n.d.). Feature branching involves isolated development with less collaboration (Zettler, n.d.). With feature branching, code commits are less frequent and typically bigger in size and length (Zettler, n.d.).

Trunk-based development comes with some challenges. A significant challenge is the extensive testing process to ensure that any deployed code will not affect the system functionality (Zettler, n.d.). Keeping up with code reviews is challenging since so many are constantly being implemented, so it can be hard to keep up (Zettler, n.d.). When developers focus solely on a single trunk, the codebase becomes more prone to conflicts and other errors (Zettler, n.d.). Since trunk-based development is highly collaborative, if communication is weak, it makes the entire process less effective (Zettler, n.d.). A few other challenges are that continuous integration and deployment are required, automation should be implemented, and reception plans need to exist (Zettler, n.d.).

**Reference**

Zettler, K. (n.d.). *Trunk-based Development*. Atlassian. Retrieved June 23, 2025, from https://www.atlassian.com/continuous-delivery/continuous-integration/trunk-based-development

**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, Samir! I think you did an excellent job on your discussion post for this module. I also chose to write about TBD. You clearly defined how Trunk-Based development works in a development environment. How do you feel about working with Trunk-based development? I would be interested to try it in an actual working environment, but I feel like it would lead to more complications since changes are constantly being made. I would prefer something like feature branches since they are separated. However, I do see how the smaller and more frequent code reviews may be beneficial in some instances.

Hey, Arely! I really enjoyed reading your post for this module. You clearly defined how decoupling deployments work and their advantages and disadvantages. Based on what I have discovered about decoupling deployments, I see why larger companies such as AWS, Microsoft, Atlassian, and LaunchDarkly incorporate it into their development process. I also discovered that non-tech companies like Coca-Cola and Zalando also use this same process. When a company is smaller in size, I do see how a monolithic structure may make more sense. But as a company begins to grow, transforming the IT infrastructure seems to be the best route to take.

Hello there, Jacob! I think your post for this discussion is very insightful and thorough. I like how you were able to link this concept back to the case study in our textbook. Connecting concepts in our books to those in our discussions and other assignments provides better clarity. It has some drawbacks, such as the cost, as you mentioned. Other disadvantages I discovered are that it can lead to technical debt, leaving developers behind where they should be in the development process, and to promised features. Even though it does have these negative aspects, overall, it seems like a great process to implement.