[Lecture 11, Timestamp 27:15](https://lsp-lectures.s3.us-east-1.amazonaws.com/spring-2025/2025-04-10/lecture11.mp4?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIA44GOPIMOZKLHXN4K%2F20250418%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250418T022950Z&X-Amz-Expires=604800&X-Amz-SignedHeaders=host&X-Amz-Signature=2ab221d587d4e26649946e817e4ccee58afe5e3d706501d8a730db367df5d6e0)

Despite design patterns being useful in most cases, the potential problem you run into by using one is finding that it's too complex for the problem that you’re trying to solve. Design patterns are best used when code needs to increase in scale over time, be edited regularly, especially by multiple editors, or have high flexibility. If the problem has a simple solution, strictly adhering to a design pattern can cause your solution to be more difficult to maintain, take longer to reuse, and more difficult to understand. Naturally, that runs counter to the purpose of using a design pattern in the first place. So, the best course of action is to identify what design pattern works best for the problem or if a unique direct solution is best.