The Strangler Application Pattern is a software modernization technique used to gradually replace a legacy system by building a new application around it piece by piece. The term was inspired by the "strangler fig" tree, which grows around an existing tree and slowly replaces it. In this pattern, new functionality is developed in a modern architecture, while parts of the old system continue to operate until they are replaced. Over time, the legacy system is "strangled" and eventually phased out completely.

This approach minimizes risk because it avoids the “big bang” rewrite [1], where an entire legacy system is rebuilt all at once a process that can be costly, time-consuming, and prone to failure. By isolating and replacing components incrementally, teams can deliver value faster, test in real-time, and adjust based on feedback. Routing mechanisms, like API gateways, are often used to direct traffic between the legacy and new components.

However, there are some disadvantages to consider. Maintaining two systems simultaneously can introduce operational complexity. Integration between old and new components may require temporary workarounds, increasing technical debt. There's also a potential performance hit due to routing logic or data synchronization between systems [2]. Additionally, if the legacy system is poorly documented or highly entangled, identifying clean separation points for strangling can be difficult.

Despite these challenges, the Strangler Pattern is a practical, lower-risk path to modernization, especially for large, mission-critical systems that cannot afford significant downtime. When planned carefully, it enables smoother transitions and more manageable upgrades.

Sources:

1. <https://www.thoughtworks.com/en-us/insights/articles/embracing-strangler-fig-pattern-legacy-modernization-part-one>
2. https://www.openlegacy.com/blog/strangler-pattern/