6.3 改良现有架构

赋予现有架构演进能力取决于三个因素：组件耦合度、工程实践成熟度，以及开发人员构建适应度函数的难易程度。

6.3.1 适当的耦合和内聚

组件间的耦合很大程度上决定了技术架构的演进能力。然而，如果数据模式死板、僵化，即便最具演进性的技术架构也注定会失败。清晰解耦的系统易于演进，充满耦合的系统则会妨碍演进。想构建出真正可演进的系统，架构师必须考虑架构中所有受影响的维度。

除了技术层面的耦合，架构师还必须考虑和保护系统中组件的功能内聚。当从一种架构迁移到另一种架构时，功能内聚性决定了组件重构后的最终粒度。这并不意味架构师可以随心所欲地分解组件，而是说基于特定的问题上下文，组件的大小应该是适当的。例如，有些业务问题相较其他问题耦合度更高，比如有着大量事务的系统。试图构建极其解耦的架构与这类问题相左，因此是徒劳的。

选择架构前，需要理解面临的业务问题。

虽然这个建议看似多余，但是仍然有团队选择最炫目的新架构模式，而不是最合适的架构。在一定程度上，选择架构要基于对业务问题和物理架构的综合理解。

6.3.2 工程实践

工程实践对定义架构的可演进性至关重要。虽然持续交付实践无法保证架构能实现演进，但它依然不可或缺。

为了追求效率，很多团队着手改进工程实践。一旦这些实践增强了，它们便能为更高级的能力（例如演进式架构）提供基础。因此，构建演进式架构能够促进效率提升。

很多公司处于新、旧实践的过渡阶段。它们可能已经实现了一些相对容易的实践，例如持续集成，但是仍然需要大量手工测试。虽然这些手工测试会延长生产周期，但在部署流水线中包含一些手动阶段很重要。第一，这样会将应用构建的每个阶段都置入部署流水线中。第二，随着团队逐步将更多部署工作自动化，手动阶段也会实现自动化，不再中断部署过程。第三，阐明每个阶段有助于我们更好地理解构建的各个手工部分，创造更好的反馈环并推动改进。