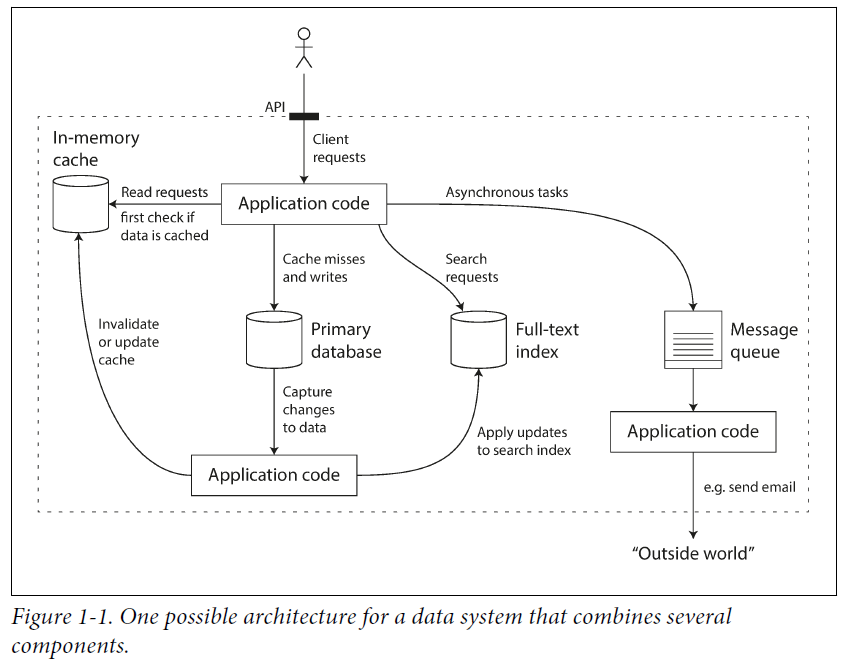
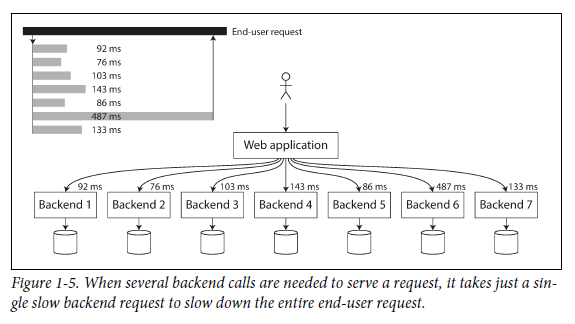
**Chapter 1**



**Goals**

* Scalable
* Maintainable
* Reliable



**Operability: Making Life Easy for Operations**

It has been suggested that “good operations can often work around the limitations of bad (or incomplete) software, but good software cannot run reliably with bad operations. While some aspects of operations can and should be automated, it is still up to humans to set up that automation in the first place and to make sure it’s working correctly.

Operations teams are vital to keeping a software system running smoothly. A good

operations team typically is responsible for the following, and more:

* Monitoring the health of the system and quickly restoring service if it goes into a bad state
* Tracking down the cause of problems, such as system failures or degraded performance
* Keeping software and platforms up to date, including security patches
* Keeping tabs on how different systems affect each other, so that a problematic change can be avoided before it causes damage
* Anticipating future problems and solving them before they occur (e.g., capacity
* planning)
* Establishing good practices and tools for deployment, configuration management, and more
* Performing complex maintenance tasks, such as moving an application from one platform to another
* Maintaining the security of the system as configuration changes are made
* Defining processes that make operations predictable and help keep the production environment stable
* Preserving the organization’s knowledge about the system, even as individual people come and go

Also:

* Providing visibility into the runtime behavior and internals of the system, with good monitoring
* Providing good support for automation and integration with standard tools
* Avoiding dependency on individual machines (allowing machines to be taken down for maintenance while the system as a whole continues running uninterrupted)
* Providing good documentation and an easy-to-understand operational model (“If I do X, Y will happen”)
* Providing good default behavior, but also giving administrators the freedom to override defaults when needed
* Self-healing where appropriate, but also giving administrators manual control over the system state when needed
* Exhibiting predictable behavior, minimizing surprises