Introduction

A new generation of APIs will be created as a result of microservices that move away from integration and reuse, and toward organizational and technical scalability and resiliency.

Credit: Christina Cardoza

http://sdtimes.com/digging-into-microservices/7/

Componentization via services

The most common way microservices will break down its components of software is by splitting them up into individual services.

Organized around business capabilities

The individual services should focus on the organization's business capability, not the technology layer.

Products, not projects

Instead of using a project model where a piece of software is handed off to another group upon completion, development teams in a microservice own their service throughout its entire life cycle.

Smart endpoints and dumb pipes

"Applications built from microservices aim to be as decoupled and as cohesive as possible," wrote Fowler and Lewis in an article about microservices. "They own their own domain logic and act more as filters in the classical Unix sense: receiving a request, applying logic as appropriate and producing a response."

Decentralized governance

Microservices give developers the choice as to what languages and tools they want to use for each service.

Decentralized data management

Microservices are meant to be individual components of a software service. Therefore, in a microservice approach, each service manages its own database.
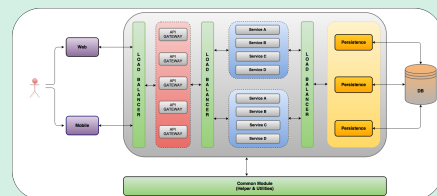
Infrastructure automation

Microservices should include infrastructure automation techniques such as Continuous Delivery and Continuous Integration.

Design for failure

Applications need to be able to endure failure.

Microservices

Evolutionary design

"Microservice practitioners usually have come from an evolutionary design background and see service decomposition as a further tool to enable application developers to control changes in their application without slowing down change," - Fowler and Lewis.