3.1 Capability perspective: API management

Let's briefly consider how API management (or sometimes called *Full Lifecycle API management*) relates to the three aspects of agile integration:

- People and process - Decentralized ownership: API management's goal is to enable both API providers and API consumers to be as autonomous as possible. Ideally, it enables them to configure, expose, discover, and consume APIs without relying on the enterprise architecture community.
- Architecture and design - Delivery led architecture: API management is in itself an architectural pattern that better enables delivery by being *consumer-centric* at its core, which means that API management is primarily concerned with enabling APIs for consumers. As such, it provides discovery, self-subscription, traffic management, secure connectivity, metrics, and more to ensure that the API is seen as needed by consumers. The API is a product in its own right.
- Technology and infrastructure - Cloud-native infrastructure: There are two sides to this aspect. The API management capability itself must be able to run on a cloud-native infrastructure such as containers if required. However, there is a more subtle side to this aspect in that API-based communication is fundamental to cloud-native applications. So, we look at when and where it is appropriate to insert an API management gateway into, for example, a microservices application.

The act of introducing API management is fundamental to integration modernization. However, even though the concepts of API management are relatively mature, it does not mean every enterprise has implemented it. Many organizations are on a digital transformation journey but they are starting within a service-oriented architecture (SOA). Their challenge is knowing what it will look like to add API management their environment. So, we are going to begin with a rough history that is similar to the one Chapter 2, “Agile integration” on page 5, but with a particular focus on the evolution of API management. This history helps you see where and why enterprises might be where they are today and what the future might look like for them.

3.1.1 A brief history of API management

When synchronous web services were made available when SOA was common, it became clear that providing an interface for consumers to call was only part of the story. Consumers needed to find that interface, learn how to use it, and self-subscribe so that they could use it securely.

Service-oriented architecture

SOA initially addressed this challenge by adding a service registry (Figure 3-1), which was a separate component of the integration run time that was used to implement the web service. It was hard to keep the definitions in the registry synchronized with the implemented interfaces of the integration run time.