How APIs can help airlines

WHAT IS AN API?

An API is a mechanism for exchanging data and/or sharing functionality between business partners. API stands for Application Programming Interface and is the mechanism powering many of today's mobile and web applications. An API works by exposing small pieces of (data and/or business logic) software to developers. Developers use these building blocks to build their own software. For an airline, this could mean providing its schedule data as an API and developers or strategic partners could then leverage this API, combine it with other information and build their own solutions.

WHY IS THIS IMPORTANT?

Many firms are realizing that there is value in their internal systems and data. APIs enable firms to expose these internal assets. This in turn opens them up to new partners and developers, reduces time to market for new products and spurs innovation.

Innovation in this space lies in finding new ways to merge APIs and move beyond specific problem-solving, bringing several APIs together means the whole becomes greater than the sum of its parts –

ZOOM INTO THE TOPIC

Web-based APIs have become an integral part of enterprise IT architectures, evolving from platforms and concepts such as SOA (Service Application Integration), EAI (Enterprise Application Integration), ESB (Enterprise Service Bus) or even ETL (Extract Transform Load). The lines around these concepts have blurred, with integration platforms such as MuleSoft, Boomi or Informatica able to deliver a blend of the above concepts. These integration platforms typically provide message transformation, routing and API interoperability, while more non-functional requirements (some technical aspects of running APIs) can be better tackled via API governance.

API Governance

When implementations start to scale, API governance provides ways to manage access to API endpoints (e.g. authentication, key management), policies, performance management (setting usage limits on API calls, throttling connections and caching) and telemetry / monitoring / analytics.

Platforms such as TIBCO Mashery are well equipped for managing, front-ending APIs and handling the load of large-scale implementations, providing a consistent single point of entry for all API consumers, as well as addressing scalability and security considerations.

API Specifications

It is expected that APIs will be well documented – clear usage instructions are essential so that a developer is able to use an API without having to understand exactly how it has been constructed by the API provider.

Well-defined APIs allow developers to get started and to demonstrate a working API call within minutes.

This also ties in with organizations' willingness to embrace an "Open-API" philosophy, facilitating access to their APIs without forcing developers to read hundreds of pages of prerequisite documentation or commit financially for proof-of-concepts and/or small-scale implementations.

INDUSTRY STATE OF PLAY

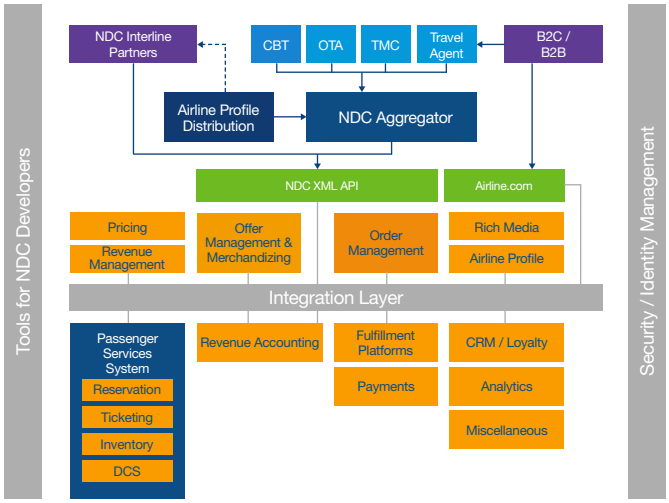
Low cost carriers have typically distributed by API, while network airlines still tend to rely on the legacy distribution channels. However, more and more airlines are now embracing NDC and have started creating APIs for shopping, flight searching, and booking. The NDC Reference Architecture is a good illustration of how a modular architecture is made possible through API integration.

NDC Reference Architecture

Leveraging integration middleware is all the more important for implementers who do not wish to rebuild or consolidate legacy systems. They can tap into their business functions and underlying data via inter-

nal APIs and integrate with the new NDC Offer/Order Management Systems. This way, existing systems can fulfil their business functions and live out their expected lifespan without having to be decommissioned earlier than expected. The NDC Reference Architecture illustrates how a modular architecture is made possible through API integration.

Different Flavors of APIs



Broadly speaking, there are two mainstream approaches to APIs - typically, low/medium-complexity B2C APIs will adopt a RESTful architecture (e.g. the types exposed by Google, Facebook, Twitter or GitHub). These are usually simpler to implement than their most common counterpart, SOAP (Simple Object Access Protocol) web services. However, when dealing with enterprise-class implementations of complex, large scale B2B systems, experience will lead towards the use of SOAP, as it is robust, well-established and provides a solid security stack.

Other emerging technologies, such as GraphQL, provide a new way of interacting with external services, notably giving developers control over the structure of the set of data being returned. IATA is currently looking into GraphQL's technical feasibility in the context of NDC.
An additional consideration is the format of message in the pipe. With SOAP, we are required to use XML.

Other data formats, such as JSON (JavaScript Object Notation), can also be used in RESTful API implementations of NDC. Even though NDC is defined as an XML messaging standard, it also provides support to JavaScript developers by providing JSON-to-XML-to-JSON transformation tools – these are intended to be deployed in front of existing NDC XML APIs to handle the transformation seamlessly without requiring changes to NDC APIs. They are openly available for download on IATA's AIR Tech Zone's GitHub repository.

Those adopting JSON as a means of data transport will still be required to adhere to the NDC XML-based standards (the seamless transformation to and from XML/JSON easily allow this validation).

For more information regarding NDC's support for REST architectures and JSON messaging, see airtechzone.iata.org.