API Ecosystem Thought Paper

Introduction

An API (Application Programming Interface) Ecosystem exists when an organization provides an API that can be easily used by an autonomous entity (group, agency or business) in a manner that adds value for the API provider, the API consumer or both.

Application Programming Interfaces (API) specifies how software components can interact with each other. Distributed computing environments such as Microsoft Windows and UNIX provide APIs that allow programmers to access core system services so that they do not have to program the same functionality within their solutions. Standardized interfaces provide access to functionality which can be integrated into new software systems and applications. The ubiquity of the internet provides for standardized protocols (HTTP, REST, WS*) for exchanging machine readable information. This has led to the adoption of both the Web and Mobile Computing by citizens seeking access to information-based public services.

APIs are categorized into two areas: Private APIs which is not available for external integration while Public APIs which are accessible both inside and outside of the providing organization through different business models. /1/

API Business Models utilize APIs where the API can be 1) the product, 2) provides an integration point to a third party product, 3) where it can provide data that drives interest in a product or 4) where it powers a product by exchanging content. These models are often summarized as one of three types:

- **APIs as services:** Often referred to as 'microservices' that expose an interface to a business process. *Example: AWS, Skype*
- **APIs as interactions**: How a physical device communicates through software. *Example: Mobile Phone Applications, Internet of Things (sensors, actuators)*
- **APIs as products**: Digital services exposed for integration into other products. *Example: Google Maps, Twitter*

API ecosystems are supported by a collection of components including API gateways, API registries, API developer portals often provided through API management platform products.

Definitions

- **API Management Platform** - A supporting environment that manages the published API by providing capabilities such analytics and usage reporting, API key and authorization management, documentation, developer community management and billing and payment services.
- **API Gateway** - is a type of API Management Platform that acts as a single point of entry to an API, providing capabilities such as authentication, security policy enforcement, load balancing, cache management and where required service level management. /2/
- **API Registry** - is a database of published APIs
- **API Portal** - is a central point where API providers can develop their API, publish supporting documentation including code examples. API consumers can locate information on how to integrate an API into their solutions including information relating to security requirements /3/ /4/
- **API Key** - is a code that is exchanged by computer programs invoking an API to identify the calling program, its developer, or its user (aka consumer). /5/
- **REST/SOAP** - are two popular means for implementing web services. REST (Representational State Transfer) has advantages for lightweight clients such as Mobile Phone Apps while SOAP (simple object access protocol) is primarily a set

of rules for XML based message exchange commonly used in machine to machine message exchange. Both use HTTP as the primary transport protocol. /6/

Government Use Cases

DataBC APIs - DataBC is a BC Government initiative to provide the management and sharing of data and information across government and to the public. In partnership with the BC Developers Exchange, DataBC provides an API management solution including a Developers Portal, a managed API Gateway and an API Registry. Published APIs can be discovered by accessing the DataBC online catalogue : https://catalogue.data.gov.bc.ca/dataset?tags=API

BCLaws API- BC Laws is an electronic library providing free public access to the laws of British Columbia. BC Laws is hosted by the Queen's Printer of British Columbia and published in partnership with the Ministry of Justice and the Law Clerk of the Legislative Assembly. This direct access to raw data is intended to enable third parties to build or add their own custom applications based on the structure of the data. https://catalogue.data.gov.bc.ca/dataset/bc-laws-api

Standards

While there is no one definitive API standard that have been adopted by the international or domestic standards bodies such as the ISO, IEEE or NIST the BC Government has adopted REST as its development standard for public facing APIs.

BC Government REST API Development Standard
https://www2.gov.bc.ca/assets/gov/government/services-for-government-and-broader-public-sector/information-technology-services/standards-files/rest_api_development_standard.pdf

Architecture Standards and Planning Branch of the OCIO is also tracking the openAPI and xAPI initiatives:

OpenAPI 3.0
https://github.com/OAI/OpenAPI-Specification/blob/master/versions/3.0.0.md

xAPI
http://standards.ieee.org/develop/wg/xAPI.html

Wikipedia provides comprehensive collection of API initiatives:
https://en.wikipedia.org/w/index.php?title=Special:Search&search=api+specification&fulltext=1&profile=default&searchToken=e6c8szwfnxem2qnw8tcmwapp0

Information on first generation XML based Web Services standards are also available via Wikipedia
https://en.wikipedia.org/wiki/List_of_web_service_specifications

Security and Privacy Considerations

Poorly designed APIs can provide unintended information for potential attackers. As with other interfaces to electronic services, developers are required to fully understand and take mitigation steps to prevent possible service exposure including information leakage. A detailed understanding of HTTP based protocols and how ill-designed message formats can provide

undesired information for hackers to exploit. Understand when to use transport level security (SSL) and how to provide authorization/authentication security based on evolving standards such as OAuth, SAML, OpenID Connect and WS_Security.

Take measures to find and mitigate the API attack surface by analyzing API metadata that can be used in brute force and local recording proxy type attacks. Developers need to prevent invalid and /or malicious input attacks such as injection vulnerabilities. Utilize tokenization to obfuscate important and private information. Consider what data security measures are used to protect both data in transit and at rest in order to maintain security and privacy integrity. Utilize regression testing tools to ensure APIs are created to be *secure by design*.

Recommendations

- Develop an API Lifecycle Strategy that considers the service design aspects (consumer management, service levels), service transition (development, test, release management, change management), service operations (access management, consumer support, incident management, API management) and Continuous Service Improvement based on API usage analytics and instrumentation.
- APIs should be "secure by design" – understand the potential risks and attack surfaces
- Publish your API in the DataBC API Registry
- Use Swagger to describe your API
- Public facing APIs must conform with the BC Government REST Developer API Standard
- Develop API services for scalability

Summary

Application Programming Interfaces are a foundational element of Digital Government. Through an understanding of the API ecosystem developers are able to provide access to government data and services that will foster a cycle of innovation to aggregate different sources of information to produce new high quality digital services. Business Analysts need to understand the opportunities for new API based business models through which to promote *Digital Government as a Service.*

References/Endnotes

/1/ Public vs Private APIs: https://www.apiacademy.co/lessons/2015/04/api-strategy-lesson-201-private-apis-vs-open-apis
/2/ API Gateway: https://auth0.com/blog/an-introduction-to-microservices-part-2-API-gateway//
/3/ API Portal example: https://anypoint.mulesoft.com/exchange/portals/addison-lee/85606ae9-0e84-443d-a324-871f5e571a98/account-api/
/4/ API Developer Portal: https://apifriends.com/2017/07/10/tips-to-choose-the-right-api-developer-portal/
/5/ API Keys: https://cloud.google.com/endpoints/docs/openapi/when-why-api-key
/6/ REST v SOAP: https://searchmicroservices.techtarget.com/tip/REST-vs-SOAP-Choosing-the-best-web-service