Architecture Options for API Enabling z Systems
z API Roadshow

Richard Gamblin
European Technical Leader | Digital Transformation
richard.gamblin@uk.ibm.com | @RichGx
Agenda

- 09:30 Registration & Coffee
- 10:00 Welcome & Trends in the API Economy: the Business Opportunity
- 10:30 Creating APIs from mainframe applications with z/OS Connect
- 11:30 BREAK
- 12:15 Creating and managing APIs with API Connect
- 13:15 LUNCH
- 14:00 Securing APIs from End-to-End
- 15:00 Architecture Options for API Enabling z Systems
- 15:45 IBM Engagement Offerings & Closing Comments
- 16:00 CLOSE
Discussion points

• There are many different ways to do REST with the mainframe. What option is best for me?

• How do I introduce APIs into my existing application integration architecture?
– V1 published December 2015

– V2 has arrived! Draft from Feb 2017

– Introduces new hybrid reference architecture

– Provides guidance on choosing between REST API enablement solutions

– Contains real world scenarios

Agenda

• Application integration architecture considerations
  • How do I introduce APIs into my existing application integration architecture?

• Reminder and positioning
  • z/OS Connect Enterprise Edition
  • API Connect
  • IBM DataPower Gateway
  • IBM Integration Bus

• REST-enabling z Systems of Record
  • CICS
  • IMS
  • DB2
Application integration architecture considerations

(How do I introduce APIs into my existing application integration architecture?)
Integration Reference Architecture for z Systems

Consumers
- Mobile
- Partners
- API Economy
- IoT
- Cloud Apps

Systems of Engagement
- Security + API Gateway
- Channel Applications
- Microservices Applications

Systems of Record
- Enterprise Application
- Enterprise Data
- Enterprise Application

Integration Layer
- Adapter

Access Layer
- Enterprise Application

Cloud Affinity
- XaaS
- On-Premise
Access Layer encapsulates the functional units of an application by providing an interface that is well-defined and implementation-independent.

Security Gateway acts as the policy enforcement point (PEP).

API Gateway acts as the policy enforcement point (PEP) and an API discovery point.

Integration Layer connects service requesters and service providers.

Adapter

Integration Reference Architecture for z Systems

Consumers

Mobile

Partners

API Economy

IoT

Cloud Apps

Systems of Engagement

Microservices Applications

API Gateway

Enterprise Application

Enterprise Application

Enterprise Application

Enterprise Data

z Systems

Security Gateway acts as the policy enforcement point (PEP).
Access Layer

• Does a service interface already exist?
  • You might already have SOAP web services running in the SOR that can be reused by an API Gateway
  • However exposing a REST interface directly might be more convenient and streamlined

• What service granularity is exposed by the SoR application?
  • It is possible that the granularity of the existing services will need to be adapted e.g to perform service aggregation

• Do services need to be discovered?
  • To facilitate fast and agile development of SOE applications (e.g mobile apps), you need a way to allow developers to determine quickly what services are available
Integration Layer

• Is an Integration Layer used in the existing application integration architecture?
  • An Integration Layer may already be used for data and protocol conversion, and may also have a governance role in applying policies such as authentication, audit, logging, service versioning etc.
  • Many Integration Layers have been enhanced in recent years to support REST/JSON

• How many types of service requesters and service providers are needed by the application integration architecture?
  • The value of an Integration Layer is partly determined by the range of different service requesters and providers that need to be integrated

• Does the solution need to support asynchronous requests?
  • An Integration Layer is able to provide correlation for asynchronous invocations through a messaging engine
Security Gateway

• Is a security gateway used in the existing application integration architecture?
  • A security gateway allows the decoupling of security policy from the underlying application
  • Extending the security gateway to control access to APIs makes sense

• Consider the following questions when deciding what role an API gateway should play in an application integration architecture:
  • Do you need to make your business services more consumable?
  • Do you want to reach new markets and customers?
  • Do you need more control over who uses your business services?
  • Do you need to charge consumers for accessing your business services?
IBM Integration technologies in context: (some!) examples
REST-enabling z Systems

(There are many different ways to do REST with the mainframe. What option is best for me?)
Several Ways to Bring REST into Mainframe

**Mid-Tier Function or Device**

REST-handling and data conversion handled in the mid-tier, and other connectivity mechanism (SOAP/HTTP, JCA, JMS, JDBC) used to connect to backend.

**Examples:**
- DataPower Gateway
- IBM Integration Bus
- API Connect
- Other

**Direct to z/OS Backend**

REST-handling and data conversion handled by each backend system (such as CICS or IMS).

**Examples:**
- CICS JSON web services

"Gateway" to the LPAR

REST-handling and data conversion handled by a function on the LPAR, with backend systems unchanged.

**Examples:**
- z/OS Connect EE
IBM z/OS Connect Enterprise Edition: REST APIs into z/OS

**Client Need**
- Discover and consume z/OS services as full REST APIs
- Complete abstraction from data and underlying transaction details
- z/OS Connect Enterprise Edition is IBM’s strategic REST API Gateway into z/OS subsystems: CICS, IMS, DB2, z/OS & WAS z/OS

**z/OS Connect EE Capabilities**
- Self-service discovery of z assets as a set of REST APIs
- Exploitation of industry-standard OpenAPI (Swagger 2.0)
- Convert in-bound REST calls into native calls for:
  - CICS, IMS, DB2, WAS and MQ
  - Handling protocol & data conversion
- Full exploitation of z/OS Security
- zCEE works hand in hand with API Connect to simplify the governance, security and ease of consumption of z/OS subsystems
IBM API Connect: Simplified & comprehensive API foundation

- Rapid model-driven API creation
- Datasource to API mapping automation
- Standards-based visual API spec creation in Swagger 2.0
- Local API creation and testing
- On-cloud & on-premises staging of APIs, Plans & Products

- Policy enforcement
- Enterprise security
- Quota management & rate limiting
- Content-based routing
- Response caching, load-balancing and offload processing
- Message format & transport protocol mediation

- Node.js & Java Microservice runtime
- Node.js & Java integrated runtime management
- Enterprise HA & scaling
- On-cloud & on-premises staging of Microservice applications

- API discovery
- API, Plan & Product policy creation
- API, Plan & Product lifecycle mgmt.
- Self-service, customizable, developer portal
- Advanced Analytics
- Subscription & community mgmt.
DataPower: the Gateway for the Multi-channel Enterprise

Secure
- Authentication, authorization, auditing
- Security token translation
- Threat protection
- Schema validation
- Message filtering
- Digital signatures
- Encryption/decryption

Integrate
- Any-to-any message xform
- Transport protocol bridging
- Message enrichment
- Database connectivity
- Mainframe connectivity
- B2B trading partner connectivity
- Hybrid cloud integration
- Amazon EC2/SoftlayerCCI

Control
- Quota enforcement, rate limiting
- Content-based routing
- Failure re-routing
- Integration with governance platforms
- B2B partner management

Optimize
- SSL/TLS offload
- Hardware accelerated crypto operations
- JSON, XML offload
- JavaScript, JSONiq, XSLT, Xquery acceleration
- Response Caching
- Intelligent load distribution

Before DataPower Gateway

After DataPower Gateway
IBM Integration Bus: Universal Connectivity in the Enterprise

Client Need
- Gateway for routing, transformation enrichment of all kinds of “messages”
- Excellent performance
- Scalable and High availability architecture

Universal Connectivity
- All major “message” types: MQ, JMS, HTTP, SOAP, REST, TCP/IP, flat files, Databases, FTP, MFT, C:D, MQTT, VSAM, QSAM, etc.
- All major “message” formats: XML, SOAP, JSON, fixed, variable, tagged, IDOC, SWIFT, etc.

Extensive functional support
- Major databases (DB2, Informix, Oracle, Informix, Microsoft SQL Server, etc.)
- Major EIP and EIS interfaces (CICS, IMS, SAP, PeopleSoft, Siebel, etc.)
- Interfaces to BPM, ODM, Analytics, etc.

Rich development & runtime
- Visual development
  - Full development: Java, eSQL, .Net
  - Transactional 2PC support
- Major platforms (z/OS, Linux, AIX, Windows, Cloud, etc.)
- Full DevOps support, admin + security
REST-enabling z Systems of Record

(There are many different ways to do REST with the mainframe. What option is best for CICS, IMS, DB2 and so on?)
REST-enabling CICS applications (+ don’t forget SOAP)

<table>
<thead>
<tr>
<th>Client type</th>
<th>Solution</th>
<th>When to use….</th>
</tr>
</thead>
</table>
| REST        | **z/OS Connect EE V2.0** REST APIs | RESTful interface (with API Editor)  
Seperately priced feature  
Discovery of z/OS assets leveraging Swagger 2.0  
Option for optimized native JSON parsing when run in CICS |
| REST        | Liberty **JAXRS** | Java solution supported with Liberty in CICS TS V5.1  
Inbound and outbound support |
| REST        | CICS **JSON web services** | Reuses same framework as CICS SOAP web services.  
Option for optimized native JSON parsing in CICS TS V5.3  
Inbound and outbound support |
| SOAP        | Liberty **JAXWS** | Java solution supported with Liberty in CICS TS V5.2  
JAXB used for XML  
Inbound and outbound support |
| SOAP        | CICS **SOAP web services** | Reuse investment in CICS web services  
Inbound and outbound support |
## REST-enabling IMS applications and data (+ SOAP)

<table>
<thead>
<tr>
<th>Client type</th>
<th>Solution</th>
<th>When to use….</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST</td>
<td>z/OS Connect EE V2.0</td>
<td>RESTful interface (with API Editor)</td>
</tr>
<tr>
<td></td>
<td>REST APIs</td>
<td>Separately priced feature (at V2.0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Discovery of z/OS assets leveraging Swagger 2.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exploits IMS Connect</td>
</tr>
<tr>
<td>SOAP</td>
<td>IMS SOAP Gateway</td>
<td>Reuse investment in IMS web services</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Inbound and outbound support</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Exploits IMS Connect</td>
</tr>
</tbody>
</table>
### REST-enabling DB2 z/OS data

**Client type** | **Solution** | **When to use…**
--- | --- | ---
REST | DB2 DDF native REST Service Interface | REST Services into DB2 z/OS
 | z/OS Connect EE V2.0 REST APIs | REST services
 |  | Discovery of z/OS assets via browser
 |  | z/OS Connect Enterprise V2 Support is in plan

1 per ENUS215-493 Statement of Direction
REST-enabling MQ z/OS

Considerations for using z/OS Connect EE with MQ:
• Reuse existing MQ infrastructure for driving back end applications (MQ two-way service)
• Enable asynchronous REST services (MQ one-way service)

**Note:** MQ Service Provider supports z/OS Connect services only (no support for API Editor).
Summary
<table>
<thead>
<tr>
<th>Integration architectures</th>
<th>Description</th>
<th>When to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST</td>
<td>Resource-oriented architecture based on HTTP URL and verbs, and JSON.</td>
<td>De facto standard for mobile, web, and cloud applications. JSON payloads are less brittle to application changes.</td>
</tr>
<tr>
<td>Web services</td>
<td>Service-oriented architecture based on standards like SOAP, XML, and Web services (WS.*) specifications.</td>
<td>Use when solution requires application integration between different service types and platforms, and XML is the predominant data payload.</td>
</tr>
<tr>
<td>APIs and API management</td>
<td>Architecture for creating, assembling, managing, securing, and socializing web APIs.</td>
<td>Use when business functions need to be discoverable and need a high degree of operational governance.</td>
</tr>
<tr>
<td>Messaging</td>
<td>Asynchronous transport mechanism.</td>
<td>Use when two applications are not necessarily available at the same time. Exploit for MQTT, for publish/subscribe applications, or when reusing existing messaging infrastructure. Use when assured delivery is required.</td>
</tr>
</tbody>
</table>
### Integration solutions

<table>
<thead>
<tr>
<th>Integration solution</th>
<th>Description</th>
<th>When to use</th>
</tr>
</thead>
<tbody>
<tr>
<td>z/OS Connect Enterprise Edition</td>
<td>IBM’s strategic REST gateway into mainframe subsystems</td>
<td>Use when discoverable REST/JSON APIs are required for access to CICS, IMS, WAS, DB2 on z/OS.</td>
</tr>
<tr>
<td>API Connect</td>
<td>IBM solution for creating, publishing, and managing web APIs.</td>
<td>Use to manage the full lifecycle of API management: creation and publication of APIs; governance and monetization; plus socialization/discovery and analytics.</td>
</tr>
<tr>
<td>IBM DataPower Gateway</td>
<td>IBM’s security and integration gateway.</td>
<td>Use for securing access and integrating with REST and web service-enabled systems. Use as the deployment runtime for API Connect.</td>
</tr>
<tr>
<td>IBM Integration Bus</td>
<td>IBM’s strategic Integration Layer with comprehensive support for any-to-any transformation.</td>
<td>Use for complex integration requirements, service aggregation, and industry-standard message formats.</td>
</tr>
<tr>
<td>IBM MQ</td>
<td>IBM’s asynchronous messaging solution.</td>
<td>Use for reliable once, and once only, assured delivery for asynchronous and pseudo-synchronous connectivity.</td>
</tr>
</tbody>
</table>
## Strategic REST API solutions

<table>
<thead>
<tr>
<th>Integration solution</th>
<th>Description</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>z/OS Connect EE V2.0</strong></td>
<td>Strategic solution for enabling REST APIs based on z/OS applications.</td>
<td>Use to enable unified REST interface for CICS, IMS and DB2.</td>
</tr>
<tr>
<td></td>
<td>Includes tooling for API creation and deployment.</td>
<td>Avoids multiple data transformations (use REST/JSON as message format from the client to the mainframe).</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use to enable discovery of APIs.</td>
</tr>
<tr>
<td><strong>IBM API Connect</strong></td>
<td>Comprehensive API lifecycle to Create, Run, Manage and Enforce APIs and Microservices.</td>
<td>Use to create APIs and microservices that consume z Systems APIs</td>
</tr>
<tr>
<td></td>
<td>DataPower and Node.js deployment options</td>
<td>Manage and secure z System APIs created by z/OS Connect</td>
</tr>
<tr>
<td><strong>IBM DataPower Gateway</strong></td>
<td>SOA and mobile security gateway.</td>
<td>Use for securing access to mainframe, and as runtime for API Gateway.</td>
</tr>
</tbody>
</table>
References

• z/OS Connect Enterprise Edition

• API Connect

• IBM DataPower Gateway

• Integration Bus

• IBM MQ
Thank you!

Any Questions?