Redbooks

ibm.com/redbooks

Building an Enterprise API Strategy

Mark Moore

z Systems

IBM

Point-of-View

Unlocking IBM z Systems assets for investment in the API economy

Highlights

There is untapped value in your IBM® z/OS® assets. APIs are the way to extract and grow that value:

- APIs provide access to systems and data on z/OS that otherwise require specialized knowledge and skills to reach.
- Interfaces based on REST and JSON are highly consumable, providing near universal access.
- The API economy enables further opportunity by providing a mechanism through which your data and services can be integrated with those of business partners in web and mobile applications.
- Using z/OS Connect to develop APIs for your z/OS assets is easy, cost-effective, and secure.

Explore how to build an enterprise API strategy and tap into your z/OS assets with this IBM Redbooks® Point-of-View publication.

A new business opportunity requires research to weigh risk and reward and to understand the time frame for return on that opportunity. When you, as the decision-maker, decide to act, liquidity is important. If capital is tied up in other investments, the ability to pursue a new one is lost.

Digital capital should be different. In theory, the value in digital assets can be invested repeatedly with no loss of liquidity. Each new point of contact with an asset—a mobile application that might use sensor data, credit evaluations, inventory, and others—creates new value. However, as the saying goes, “In theory, there is no difference between practice and theory. In practice, there is.” The difference in investing the value of a digital asset in a new opportunity is that you must create access to the asset for that opportunity.

You create access to digital assets with APIs, which are, in effect, the analog to investment liquidity in a digital market. Having an API enables you to engage in new opportunities. Just as with an investment, in creating that access, risk must be minimized while maximizing reward. Time-to-value must be as short as possible. As the markets change and the business adjusts it strategy, you must be agile and responsive, enabling the engagement in new opportunities and renewal of prior ones.

An organization can have its own internal economy where APIs are useful and produce value. APIs can remain private even if they are supporting public applications or a trusted partner. The potential rewards and risks are elevated with the exposure of APIs as public, inviting third-party development. One popular real estate site, for example, offers its data through a query API, encouraging the development of unique real estate web and mobile apps. In either case, the best opportunities are where the asset exposed through the API has unique value and a waiting audience. The balance of risk and reward can be made more favorable with partners that provide additive or complimentary value. To the extent existing skills, tools, and capabilities can be leveraged, costs are reduced.

The reason the API economy is building momentum and the reason to participate in it is that APIs enable the business to expose systems and data, which might have been developed at different times and with a variety of goals in mind, in a way that represents the best, current face of the business.

Of course, every platform has unique considerations and capabilities when participating in the API economy, and IBM z/OS® is no exception.

© Copyright IBM Corp. 2016.

Realizing an enterprise API strategy with IBM z/OS Connect Enterprise Edition

Business strategy necessarily drives technology choices. The role of any tool in business is to improve a result or bring about that result faster, or in the best case, both. APIs improve results by delivering information and services that are otherwise locked behind the company firewall to websites and applications that provide value to the customer or client. Depending upon the nature of the z/OS asset that is provided, there might be sufficient value to deliver their data or services in isolation. A bank, for example, that can deliver more timely and accurate decisions to credit applications by exposing a z/OS asset through an API to the bank's website has value.

However, there is strength in numbers. Perhaps the greatest strategic value of APIs is in their ability to provide integration between a business and its partners and vendors. If that same bank is able to deliver decisions on home loans through a real estate website, the value of that partnership can be multiplicative. The real estate agent is far more likely to sell a home if the buyers can be sure they will receive financing, and the bank is far more likely to win a mortgage bid if they are present, if only in a virtual sense, at the time the decision is made to buy a home. APIs make this possible by providing easy-to-use, web-accessible access to the data and services that are required to do business.

By providing API access to z/OS assets, IBM z/OS Connect Enterprise Edition (EE) contributes strongly to the achievement of business objectives at all perspectives of the Balanced Scorecard[1]:

- **Financial**: z/OS Connect EE provides a cost-effective way to unlock the value of z/OS assets that, over time, provide increasingly greater leverage because the APIs can be reused in an unlimited number of contexts.
- **Customer**: Customers will be delighted not only by the new capabilities the business can deliver by unlocking z/OS assets, but by integrated offerings that APIs enable with business partners and other vendors.
- **Business process**: By providing a formalized access methodology in the form of APIs, the business avoids one-off solutions to delivering z/OS assets. Consistency of implementation is assured in all contexts in which the APIs are used.
- **Learning and growth**: Because z/OS Connect EE APIs employ ubiquitous web standards, the business can leverage skills that are plentiful in the marketplace to deliver high-value solutions in a rapid and cost-effective manner.

RESTful services enable the execution of your API strategy

Where once there was direct database access or calls to runtime libraries, tightly coupled and difficult to leverage outside the corporate firewall, now there is a loosely coupled, flexible, robust, and easily leveraged method of access—RESTful services that deliver data in JSON format.

The challenge in gaining access to z/OS assets is that many are mature and were implemented long before Representational State Transfer (REST) and JavaScript Object Notation (JSON) came into being. The challenge is worth addressing, though, because in many respects, is it the maturity of these assets that makes them valuable. With time, they have acquired depth and can be the source of significant insights and the foundation of sophisticated services.

By allowing the construction of RESTful interfaces to z/OS assets, IBM z/OS Connect EE enables you to meet your requirements and enter the API economy with confidence. z/OS Connect EE is built on the small-footprint application server Liberty Profile for z/OS. It exposes selected functions and data of z/OS assets as RESTful services and JSON, mapping JSON to the data format requirements of the asset. Customizable security interceptors enable role-based security and auditing. The tooling enables development by those familiar with web technologies without requiring that they also know CICS or IMS.

[1] https://en.wikipedia.org/wiki/Balanced_scorecard

IBM z/OS Connect EE addresses the key aspects of API development, deployment, and management:

- **Universal access**: The medium of exchange in the API economy is JSON delivered through RESTful services. Lightweight and easily consumed, virtually anything that can establish an HTTP connection can interact through REST and JSON, from browsers and mobile devices to applications on virtually any platform or in the cloud.
- **Comprehensive coverage**: The components of z/OS assets (IBM CICS®, IMS™, batch, and so on) employ a number of protocols and access methods. Although creating bespoke RESTful services for particular opportunities might be possible, a solution that provides platform-wide access is desirable.
- **Quality of interface**: Converting a protocol to REST and JSON directly is likely to result in something that is at best oddly idiomatic and at worst unusable. A broker that can provide a proper RESTful interface and resolve the underlying idiosyncrasies is required. These interfaces must be made discoverable and consumable with an API framework like Swagger[2].
- **Security and auditing**: Just as providing access to banking services does not mean opening the vault to the public, neither does providing an API mean granting all users access to all data. Securing access to the APIs with fine-grained control that can filter data based on access level and integrating an audit trail with the z/OS System Management Facility is imperative.
- **Price and performance**: These are necessarily linked. There is a danger that maintaining the current workload performance while adding support for APIs might come at the expense of adding extra computing cycles. To the extent that the additional cycles demanded by the API can be offloaded to zIIPs[3], both cost and performance are maintained at appropriate levels. This would have to compare favorably to alternatives, such as offloading data to servers dedicated to hosting the APIs.
- **Tooling**: To accomplish all of this, tools that support agile, iterative API development and ongoing lifecycle management are required.

Clearly, an enterprise API strategy requires far more than responding to an HTTP request with data wrapped in curly braces. But, with the proper approach, that strategy does not have to be complex.

Universal access through REST

RESTful services use the HTTP protocol. However, where a browser uses a URL to retrieve a web page, a client of a RESTful service uses a URI—Uniform Resource Identifier—to identify an object or objects on a server. For example `/employees/123` might refer to a company employee with the employee ID “123”. If you are telling a service to do something, that is the noun in your request. The verbs are the HTTP operations POST, GET, PUT, and DELETE. These HTTP commands are analogous to database operations, create, read, update, and delete (known colorfully as CRUD operations). Where you are creating or updating with POST or PUT, the body of the request is JSON. The same is true when data is being returned in response to a request. For example, in response to a GET for the URI `/employees`, one might receive the following JSON:

```
{"employees":[
    {"firstName":"John", "lastName":"Smith"},
    {"firstName":"Donna", "lastName":"Noble"},
    {"firstName":"Jane", "lastName":"Doe"}
]}
```

This highly consumable response is especially useful for JavaScript-based frameworks like a browser or Node.js.

The power of REST and JSON is that the description above is about all there is to them. They are so straightforward that one can even interact with RESTful services by typing requests at a command line or in a web browser. z/OS Connect EE provides universal access by interacting in the API economy using REST and JSON, mapping the requests to the z/OS assets and unlocking their value.

[2] http://swagger.io/

[3] IBM z Integrated Information Processor (zIIP) is used under z/OS for designated workloads, which include IBM Java Virtual Machine (JVM), various XML System Services, and others.

Comprehensive coverage

z/OS Connect EE connects to all of the major systems of record, abstracting away the individual access methods and data formats (Figure 1). It can serve as a proxy to existing RESTful services that might be on z/OS or elsewhere in the enterprise. IBM WebSphere® Optimized Local Adapters (WOLA) are supported for connection to WOLA services. WOLA is used for in-memory communication to CICS COMMAREA or Channel programs, and also long-running batch programs. IMS access is through IMS Connect.

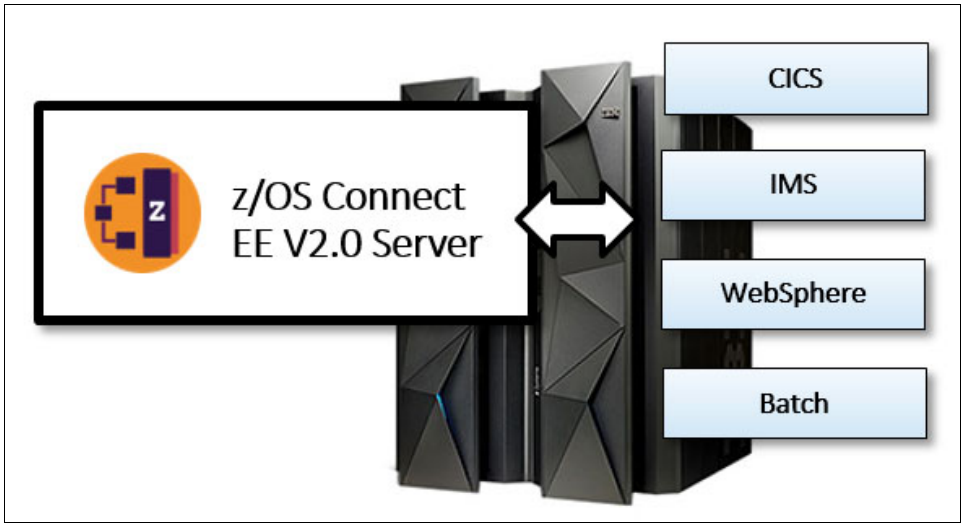

Figure 1 z/OS Connect provides access to systems of record

Quality of interface

z/OS Connect EE exposes the z/OS assets as services. Service descriptions identify the z/OS source, what should be exposed, and required data transformations. An API editor can then use these service definitions to create the RESTful services. The z/OS Connect EE API Editor enables the visual mapping of RESTful requests to those services, as Figure 2 shows. The context of the request can be considered in the mapping. The result can generate a Swagger description that makes the service easily consumable and facilitates client development. The editor is based on Eclipse, which through its plug-in architecture and extensive ecosphere can support all phases of software development from gathering requirements to testing.

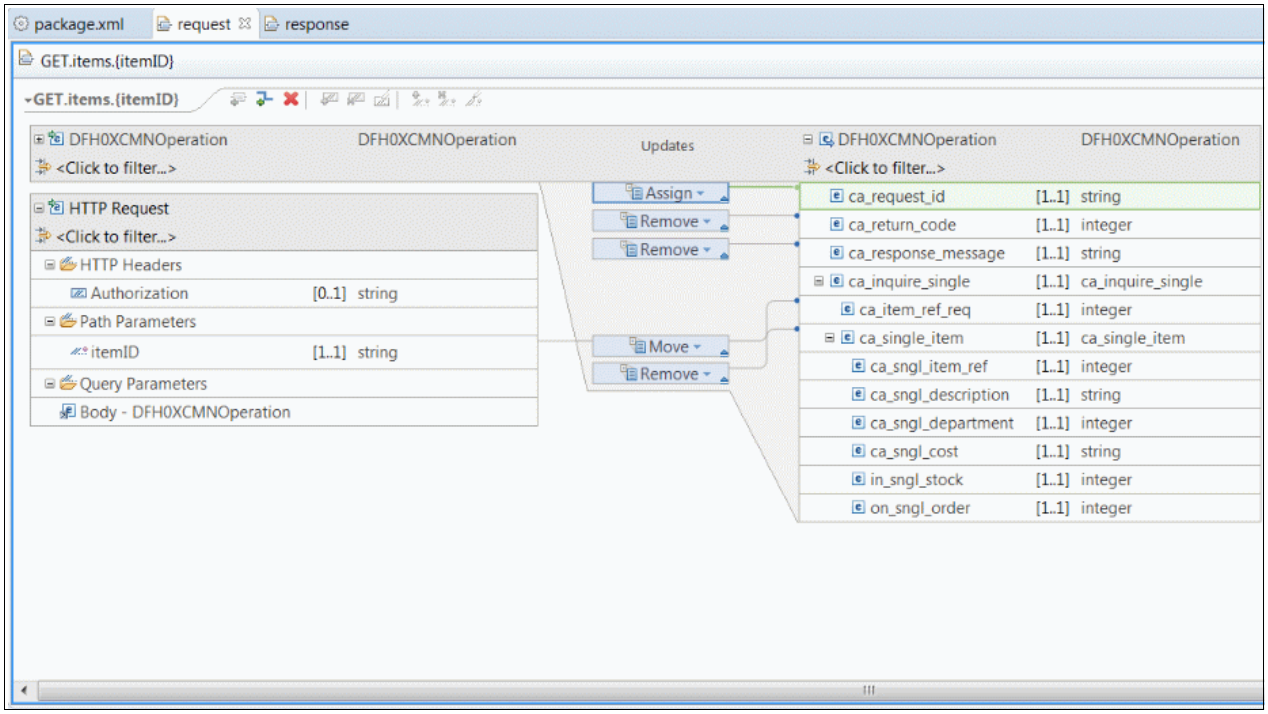

Figure 2 The z/OS Connect API Editor enables the visual construction of interfaces

Security and auditing

z/OS Connect EE integrates with the z/OS System Authorization Facility (SAF) to control access to requested services. It also integrates with System Management Facilities (SMF) to provide an audit trail. Interceptors that are fired before and after a service is invoked can provide additional security screening and logging.

Price and performance

To avoid impacting the performance of existing workloads, or perhaps in the pursuit of cost savings, trying to offload the required data to dedicated systems that host the APIs might be tempting. In that case, you would have to implement and maintain both the API and the ongoing extract, transform, and load (ETL) cycle. Data would become stale, even as it arrived at its destination.

However, z/OS Connect EE can be more than 99% zIIP-eligible. This means that z/OS Connect EE has little or no impact on software charges and it leverages spare zIIP capacity that many have. Leaving the data in place and using z/OS Connect EE is likely to be very cost-effective, whereas deploying hardware and performing the required ETL would be simultaneously more expensive and of poorer quality.

Tooling

The ability to produce new APIs visually and without coding empowers agile development that can respond rapidly to a changing business climate. Because z/OS Connect EE services conceal implementation details, users of the APIs can construct clients quickly and with no knowledge of CICS, IMS, or other underlying technology.

Additional considerations

Other considerations might come into play based on what sort of solution is being developed, how it is to be used, and so on.

High-quality, continuous delivery

Some modernization of the back-end lifecycle might be required to support the demands of the API strategy implementation. Providing developers with proper tooling and creating an environment of continuous delivery will be important.

IBM Rational® Developer for z Systems™ will provide developers with a toolset that enables more rapid and higher quality development of z Systems assets in PL/I, COBOL, and others. IBM Rational Team Concert will enable an agile development environment that enables traceability from business requirements all the way down to individual tasks, code releases, and bug fixes. DevOps will create an integrated environment of infrastructure planning, development, testing, and operations. The better the foundation from which one builds an API strategy, the better the result.

API management and microservices

As explained, fully RESTful services perform straightforward CRUD operations on objects that, in this case, are abstracted representations of the underlying z/OS assets. A useful approach might be to include a layer of business logic between the persistence layer represented by the RESTful service and consuming application. Microservices can combine calls to RESTful services and other microservices with a bit of logic that can provide building blocks of an application. Certainly, robust applications can be built with only RESTful service access to back-end data, but several applications may duplicate logic that would be better maintained in a single place.

For example, telephony, at its simplest, is two cans and a piece of string. The real product in telephony is the bill and the rules that produce it: “1000 SMS messages for a flat rate and $0.25/message after that.” That rule can change as required by market pressures. If that logic is implemented by a microservice, consumers of that service, like websites and mobile apps, can implement new product or service offerings quickly and without changing the underlying API or the clients unnecessarily.

In support of this, IBM offers IBM API Connect. It provides full API management to create, run, manage, and secure APIs and microservices. One can develop microservices in Node.js or Java. It includes a self-service developer portal. It supports service composition and debugging, discovery, policy-based access control and governance, and also monitoring and analytics.

A secure gateway

To truly participate in the API Economy, the APIs and microservices in question must be delivered through the enterprise firewall. Doing so requires consideration of the secure delivery of the data, impact on network traffic, and threat protection.

IBM DataPower® Gateway (Figure 3 on page 7) is an appliance, either physical or virtual, that can be dropped into the enterprise DMZ to secure access to APIs and microservices and also other resources that are to be delivered through the firewall. As a virtual appliance, it also can be run on cloud platforms such as IBM SoftLayer® or Amazon's AWS. The IBM DataPower Gateway provides threat protection, security enforcement, service level agreement enforcement, rate limiting, intelligent load balancing, and response caching for your APIs and microservices.

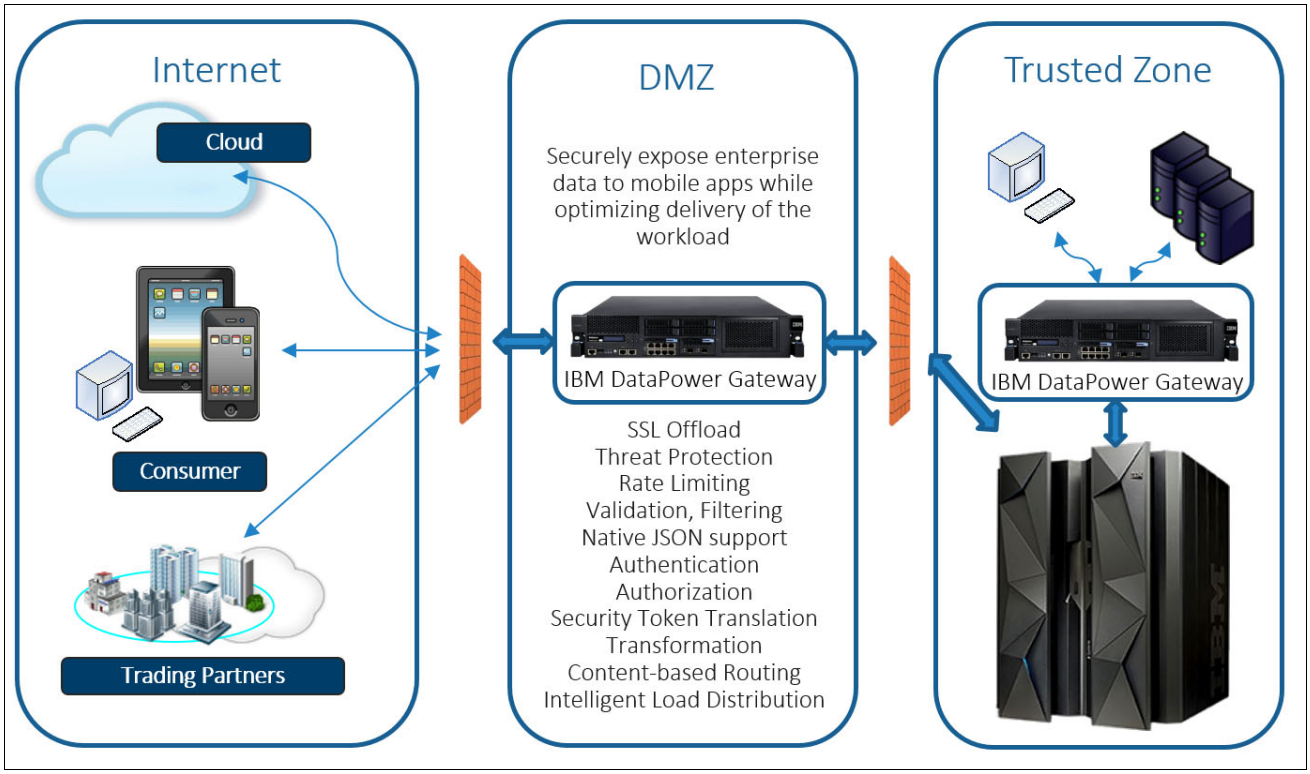

Figure 3 IBM DataPower Gateway

Application development

With APIs and microservices in place and a secure connection to the outside world, the next step is the development of client applications. Partner and third-party services, such as payment services, can be combined with your own to provide your customers with a robust, engaging experience. A natural venue for this is in the cloud with a Software as a Service (SaaS) platform such as IBM Bluemix®. You can even support third-party development using your APIs and microservices on Bluemix by providing them as custom services there.

What's next: How IBM can help

Notable disruptors of industry have questioned fundamental assumptions in their spaces. For example, one company questioned whether an actual store is needed in order to operate a store, or a data center to operate a data center. Another company, likewise, operates a taxi company without taxis.

What is described here is how questioning even small assumptions can provide great value. One might assume data to be "stuck on the mainframe" or that IBM z Systems assets cannot participate directly in the API economy. Those casual assumptions are not only incorrect, they can lead to costly disruption.

In questioning those assumptions, unlocking the value of z/OS assets, and making some investments in the API economy can provide insulation from disruption. Developing an API strategy that includes innovative delivery of the nascent value from those assets, integrating with business partners, and more, can enable the business to become a disruptor. Keep these key points in mind:

- Leave your z Systems data where it is. Moving it is expensive, unnecessary, and the data is stale the minute it is transferred.
- Providing access to the data where it is can be easy and cost-effective with z/OS Connect EE.
- z/OS Connect provides that access by using ubiquitous standards, REST and JSON.
- Security is paramount, especially when creating an open API for third-party consumption. Security with z/OS Connect EE is built in at multiple levels, and well integrated with the platform. The IBM DataPower Gateway secures external exposure.
- Consider the foundation of your strategy—the existing back end—and whether modernization will be required.

Developing an API strategy and investing in the API economy are important first steps in becoming a disruptor rather than one of the disrupted.

Resources for more information

For more information about the concepts highlighted in the paper, see the following resources:

- IBM z/OS Connect Enterprise Edition

  https://ibm.biz/Bd4gtr
- IBM DataPower Gateway

  https://ibm.biz/Bd4gts
- IBM WebSphere Optimized Local Adapters (WOLA)

  https://ibm.biz/Bd4gti
- REST

  https://ibm.biz/Bd4gtj
- JSON

  http://www.json.org/
- IBM Bluemix

  https://ibm.biz/Bd4AUL

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided “AS IS”, without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

© Copyright IBM Corp. 2016. All rights reserved.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

Bluemix®	IMS™	WebSphere®
CICS®	Rational®	z/OS®
DataPower®	Redbooks®	z Systems™
IBM®	Redbooks (logo)  ®	

The following terms are trademarks of other companies:

SoftLayer, and SoftLayer device are trademarks or registered trademarks of SoftLayer, Inc., an IBM Company.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Other company, product, or service names may be trademarks or service marks of others.

REDP-5376-00

ISBN 0738455326

Printed in U.S.A.

Get connected