

UNITE 2020 Annual Conference

Web Services Are Always Cloud Ready!

Michael Recant
MGS, Inc.

Session MCP4046
2:00pm – 3:00pm
Wednesday, November 18, 2020

MGS^{INC}
Computer Business Solutions

Overview

- **Web Services (generic) Goal**
 - Make network program-to-program exchanges as easy as browsing the Web



Overview

- The Web Services (generic) concept contains extremely powerful elements
- “Loose Coupling” between provider and consumer
- Abstracts out business functionality
- Leverage existing business functionality

The Web Services (generic) concept contains extremely powerful elements:

Simple, well-defined, standards-based interfaces

Technology independent implementation

“Loose Coupling” between provider and consumer

Anonymous client

Flexible data content

Asynchronous

Abstracts out business functionality

Creates machine (technology) independent functionality

Indirect reference to a business service

Leverage existing business functionality

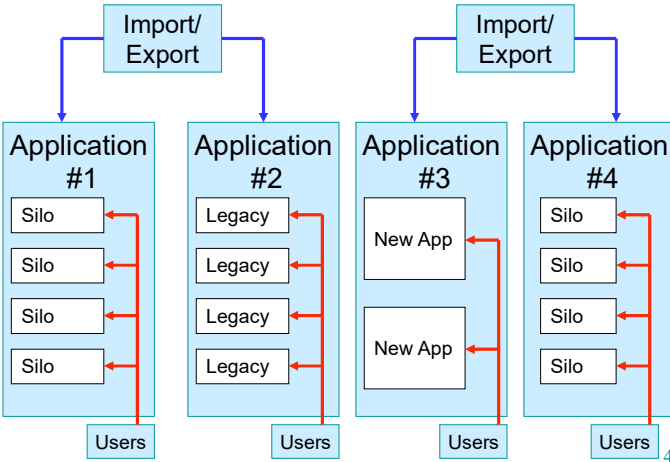
Rewrites/Redesigns are expensive

Placing a Web Services (generic) envelope around existing functionality is relatively inexpensive

Preserves investment in known, reliable business solutions

Overview

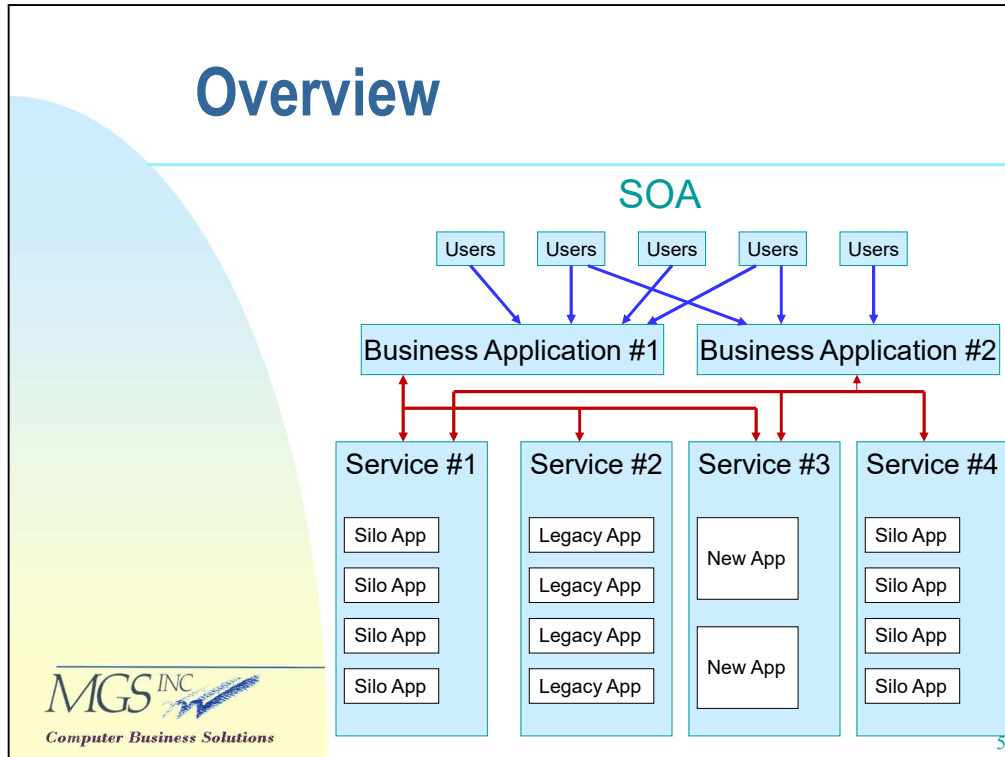
Traditional Architecture



MGS INC
Computer Business Solutions

Individual Application Silos

- Data tied to an individual application
- Batch transfer from Silo to Silo
- User tied to a specific application



Services Oriented Architecture (SOA)

Componentize Enterprise business functions

Service is a “black box” for consumers

Encapsulate existing business functions for easier access

IT Functionality now available as a set of objects that can be mixed and matched as needed

Application development done by architecting service consumers

Overview

- **Web Service (generic) Characteristics**
 - Client-Server Architecture
 - Stateless Transactions
 - Layered System
 - ❖ Client can be directly connected or have communication routed
 - ❖ Client is unaware of layers
 - Loose Coupling
 - ❖ Client tech not tied to server tech
 - ❖ Both side can evolve independently
 - Built on proven Internet communications standards

Built on proven Internet communications standards

Communication Transports:

TCP/IP – Transmission Control Protocol /
Internet Protocol

TLS – Transport Layer Security
(formerly SSL)

Communication Protocols:

HTTP - HyperText Transfer Protocol

SOAP - Simple Object Access Protocol

REST - Representational State Transfer

Payload Data Representations:

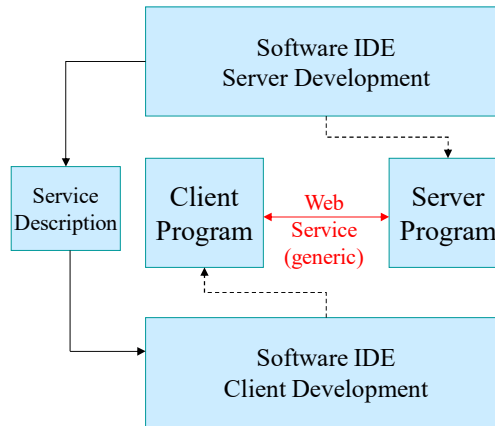
Text - plain text

XML - eXtensible Markup Language

JSON - JavaScript Object Notation

Technology

APIs Supported by software IDEs



MGS INC
Computer Business Solutions

7

APIs Supported by software IDEs

- Automatic creation of Web Services objects
- Web Services Server object support
- Web Services Client object support

Included as part of the application framework

- Microsoft .NET
- Oracle/Sun JAVA
- Unisys Agile Business Suite
- Unisys ePortal
- Unisys MCP WEBAPPSUPPORT
- MGS-Web

Technology

Description File (WSDL)

```
<message name="WSTEST_SCRN01">
  <part name="Trancode" type="xsd:string" />
  <part name="Input_data" type="xsd:string" />
</message>
<message name="WSTEST_SCRN01Response">
  <part name="Trancode" type="xsd:string" />
  <part name="Input_data" type="xsd:string" />
  <part name="statusLine" type="xsd:string" />
</message>

<service name="COMSWebServices">
  <documentation>Access COMS applications via Web Services
  </documentation>
  <port name="WSTEST" binding="wsdl:WSTESTHttpBinding">
    <soap:address location=
      "http://laptop1mcp/COMSWebServices/" />
  </port>
</service>
```



8

Service Description

User Defined

- HTTP payload is typically user defined
- GET (payload part of the URI)
- POST (payload in the HTTP request body)

WSDL

Web Services Description Language

Specifies:

- The protocol (typically HTTP)
- The operation
- An XML encoded SOAP request
- An XML encoded SOAP response

RSDL, Swagger, etc

RESTful API Description Languages

JSON or XML encoding

Technology – HTTP vs XML vs JSON Encoding

□ HTTP Query

- `http://xyz.com?Name=Mike&Email=mike@xyz.com&Country=USA`

□ XML Encoding

```
<Person>
  <ID>1</ID>
  <Name>M Vaqqas</Name>
  <Email>m.vaqqas@gmail.com</Email>
  <Country>India</Country>
</Person>
```

□ JSON Encoding

```
{
  "ID": "1",
  "Name": "M Vaqqas",
  "Email": "m.vaqqas@gmail.com",
  "Country": "India"
}
```

XML Message Payload Example

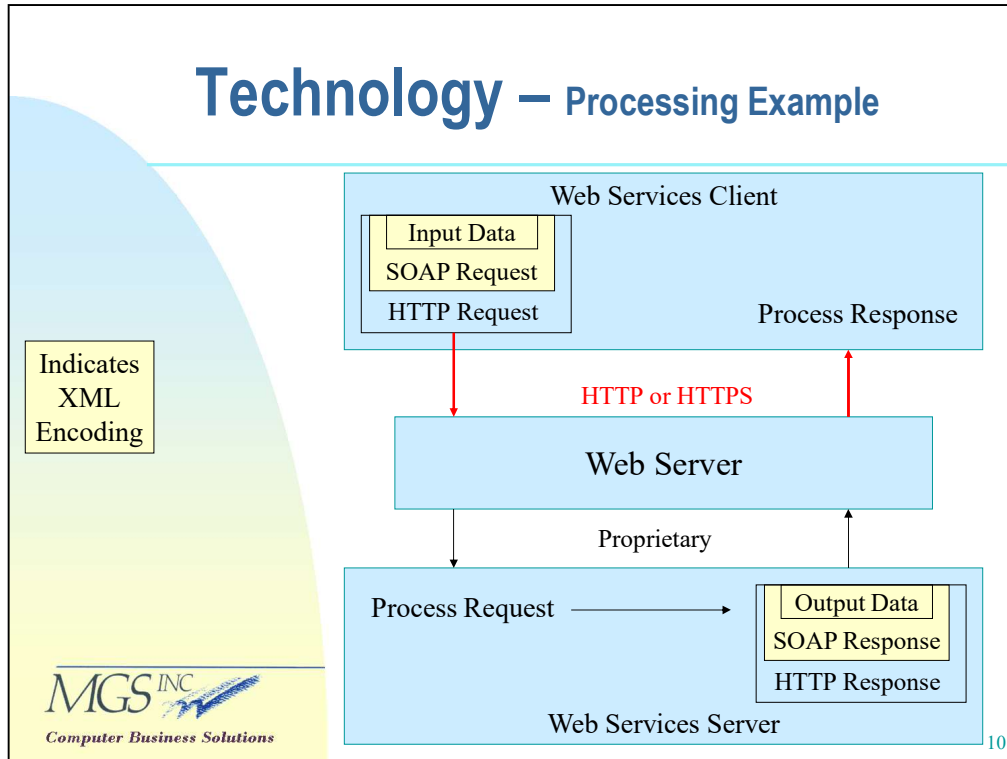
SOAP Request:

```
<soap:Envelope>
  <soap:Body>
    <tns:WSTEST_SCRN01>
      <Trancode>SCRN01</Trancode>
      <InputData>lower case letters</InputData>
    </tns:WSTEST_SCRN01>
  </soap:Body>
</soap:Envelope>
```

SOAP Response:

```
<soap:Envelope>
  <soap:Body>
    <tns:WSTEST_SCRN01Response>
      <Trancode>SCRN01</Trancode>
      <InputData>LOWER CASE LETTERS</InputData>
      <statusLine />
    </tns:WSTEST_SCRN01Response>
  </soap:Body>
</soap:Envelope>
```

Technology – Processing Example



Technology – HTTP Transaction

□ HTTP based Web Service

- Hypertext Transfer Protocol
- Core of Web Browser communications
- Can also be used for “web service” transaction processing
 - ◇ GET method
GET /test/form.php?name1=value1&name2=value2
 - ◇ POST method
POST /test/form.php HTTP/1.1
Host: w3schools.com
name1=value1&name2=value2
- **Payload** format is user application defined
- Can be HTTP form elements, XML, JSON, etc
- Response format is user application defined

Technology – SOAP Transaction

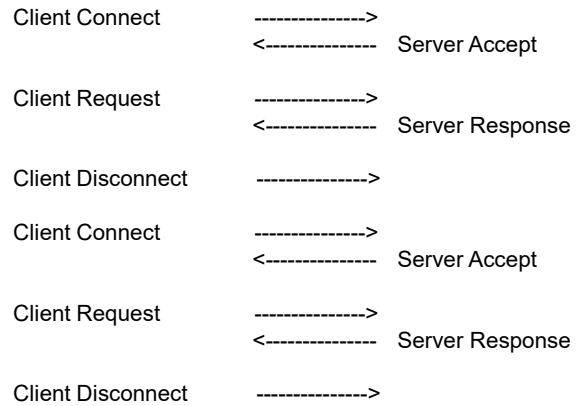
- SOAP based Web Service
 - HTTP used to encapsulate SOAP request message (POST)
 - ❖ SOAP operation
 - ❖ SOAP headers
 - ❖ SOAP body
 - HTTP response encapsulates the SOAP response
 - SOAP request and response are XML encoded
 - WSDL defines the specific format of the SOAP request/response for a given operation
 - Minimal use of HTTP control fields

Technology – RESTful Transaction

- REST based web service
 - HTTP used to encapsulate REST request message (GET or POST)
 - ❖ REST operation
 - ❖ REST headers (really HTTP headers)
 - ❖ REST body
 - HTTP response encapsulates the REST response
 - REST request and response are either JSON or XML encoded
 - REST request/response format may be defined by a Service Description (may be built into application)
 - Often relies on HTTP control fields

Technology

□ Non-Persistent Connection



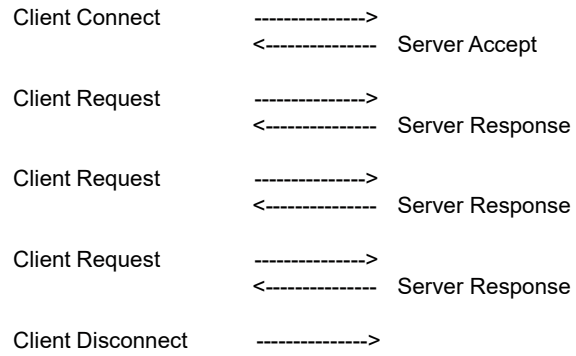
Non-Persistent Connections

MCP TCP/IP connect/disconnect slows throughput

On close, TCP/IP port must “time wait” for $TTL * 2$ seconds on most systems

Technology

□ Persistent Connection



Persistent Connections

Most servers default to persistent HTTP connections

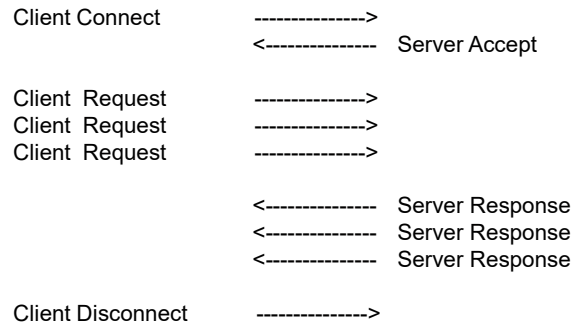
- Connection:Keepalive setting

Client then controls persistence

Persistence provides a 4 to 50 times throughput increase

Technology

□ Piplining Connection



Pipelining

Requires a persistent connection

Multiple requests are sent without waiting for responses

Responses return in send-order

Most web servers (including ATLAS) support this

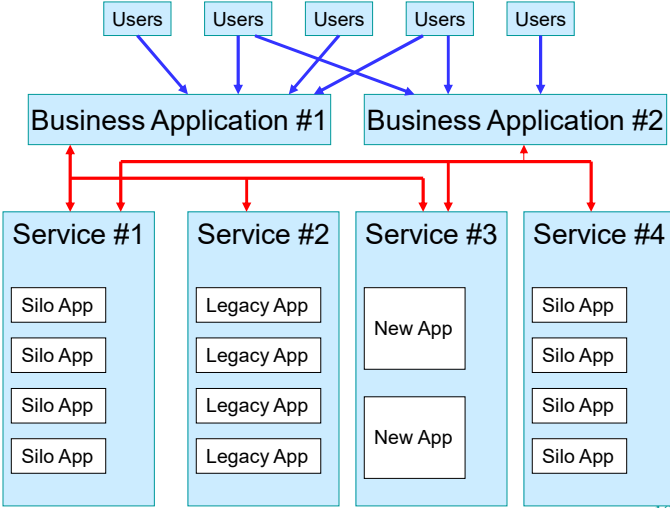
Use controlled by whether Client application is coded to take advantage of this

Security – Overview

- Digital security has three parts:
 - Identification
 - ❖ Authentication
 - ❖ Digital signature
 - Encryption
 - Journaling
- Web Services (generic) moves security considerations to a different place
 - May not be at the user's interface point
 - Often a machine-to-machine exchange
 - The "other" machine may not be trusted
- SOA changes the security landscape

Security – Overview

SOA



Security – Overview

- ▣ Web Services (generic) built on HTTP
 - Hence security considerations are similar for all three:
 - ✦ HTTP
 - ✦ SOAP
 - ✦ REST
 - Need for connection encryption
 - Need for message or session authentication
 - HTTP authentication limited so additional authentication strategies may be needed

Security – Overview

- The challenge is controlling security at different levels and different ways
 - TLS/SSL (encryption)
 - HTTP Logon (authentication)
 - SOAP/REST Headers (authentication)
 - Actual call to logon (authentication)
 - SOAP WS-Security (authentication, signature, encryption)

Security – Transport

- ▣ Transport Layer Security (TLS)
 - TLS similar but more robust than SSL
 - TLS – Authenticates server
 - ❖ Get certificate from Server
 - ❖ Validate certificate from a trusted Certificate Authority
 - Two way TLS
 - ❖ Client Authenticates server
 - ❖ Server Authenticates client
 - Encryption, provided by the certificate keys, is transparent to application
 - Application must get/supply authentication info through an external interface

Security – HTTP Logon

□ HTTP Logon

- Logon required for a specific virtual directory
- Uses HTTP AUTHORIZATION header
- BASIC uses a Base64 exchange so SSL/TLS is required for secure communications
- DIGEST uses MD5 encrypted exchange
- NTLM provides username/pw encryption and is non re-playable
- No data encryption
- Application must get/supply authentication info through an external interface

Security – SOAP/REST Headers

□ SOAP Header Example

```
<Envelope>
  <Header>
    <ABECHHeader xmlns="service.abec.com">
      <MessageData>
        <MessageID>568425287</MessageID>
      </MessageData>
      <UserAuthorization>
        <UserName>MS0281331</UserName>
        <UserPassword>x@32!aX49#$$&</UserPassword>
      </UserAuthorization>
    </ABECHHeader>
  </Header>
  <Body>
    ..... SOAP body .....
  </Body>
</Envelope>
```

SOAP/REST Headers

One must pre-acquire authentication information before the service request call

The SOAP/REST message contains both a HEADER section as well as a body

Authentication information can be provided in SOAP/REST HEADER fields


TLS is still needed to encrypt HEADERS

Application must supply authentication info using special code by setting the header fields

Security – Logon Transaction

▣ **Call to Logon**

Client Connect	----->	
	<-----	Server Accept
Logon Request	----->	
	<-----	Logon Response (token)
Client Request (token)	----->	
	<-----	Server Response
Client Request (token)	----->	
	<-----	Server Response
Client Request (token)	----->	
	<-----	Server Response
Client Disconnect	----->	


24

Call to Logon a session

One must pre-acquire authentication information (usercode/password)

An initial web services (generic) call is made for authentication

The response contains a token to be placed in the body of all subsequent web services (generic) calls

Application must be “token” aware

TLS is still needed to encrypt dialogs

Security – WS-Security

- ▣ Specific to SOAP Web Services
- ▣ WS-Security (WSS)
 - Originally developed by IBM, Microsoft, VeriSign and Forum Systems
 - Attach signature and encryption headers to SOAP messages
 - Provides end-to-end integrity for each message
 - Protocol uses SAML, Kerberos and x.509 certificates
 - Requires application awareness

Security – Summary

□ Security Solutions

- Will be dependent on the technology used and the connection type
- Use a front-end http pass-thru processor to do encryption (TLS), authentication (back-end systems are trusted) and journaling
- Note, the front-end processing may be on the same system as the Web Service (generic)
- Use TLS to obfuscate user/password authentication and Web Service contents
- Have username/password aware applications

Cloud – Overview of Cloud Services

- ▣ Available “Cloud” Services
 - Infrastructure as a Service (IaaS)
 - Platform as a Service (PaaS)
 - Software as a Service (SaaS)
 - Mobile Backend as a Service (MBaaS)
 - Serverless Computing (transaction)
- ▣ Characteristics
 - Flexibility
 - Reduced Overhead
 - Reliability/Security (?)
 - Lower Maintenance & Low Cost

MGS INC
Computer Business Solutions

Characteristics

Agility (self service)

Location Independence (network access)

Scalability (pooled resources)

Reliability (security?)

Usage Reporting (measured service)

Low Maintenance & Low Cost

Cloud – Software as a Service (SaaS)

- ▣ Solutions provided over the Internet
- ▣ Solution can be utilized by customer without server or application involvement (data only)
- ▣ Examples
 - DropBox (Doc Control)
 - Salesforce.com (CRM)
 - Quickbooks (Financials)
 - Microsoft Dynamics 365

Cloud – Application Integration

- ▣ Avalara Sales Tax Service
 - Cloud based SaaS for:
 - ❖ Tax rate calculation
 - ❖ Exempt certificate management
 - ❖ Filing and returns
 - Validates the address of where the transaction is occurring
 - Calculates taxes on a document such as a sales order, sales invoice, purchase order, purchase invoice, or credit memo.

Cloud – Application Integration

- ▣ OpenText Facsimile Service
 - Cloud based SaaS for:
 - ❖ Facsimile transmission
 - ❖ Distribution lists
 - ❖ Supports a variety of formats
 - ❖ Provides result reports
 - API to Send Facsimile
 - API to check status

Cloud – Application Integration

- ▣ SaaS APIs are generally “standards” based
- ▣ Mainframe Applications require middleware Support:
 - Communications
 - XML/JSON construction and parsing
 - Conversion of flat COBOL data structures to/from XML/JSON

Cloud – Application Integration

- ▣ No “native” Mainframe Solution
- ▣ Middleware Required (example)
 - Hardware/Software (ePortal, AB Suite)
 - Software Only (MCPJava, MGSWeb)
 - Custom app development using WEBAPPSUPPORT features
 - ❖ HTTPCLIENT
 - ❖ XMLPARSER

Cloud – Application Integration

□ Example application code to call the SaaS Web Service (generic)

```
COPY "WEBSERVICES/USERFILES/WEBOUT/SAAS-GETTOKEN".

MOVE SPACES TO REQ-RECORD.
MOVE SPACES TO RESP-RECORD.
MOVE "GETTOKEN" TO TRANCODE.
MOVE "SaaS Request Info" TO INPUTDATA.

CALL "INVOKE OF WEBSERVICES/LIBRARY"
  USING SAASFUNCT-PARAM,
        REQ-RECORD,
        RESP-RECORD
        RESULT-STRING
  GIVING RESULT.

IF RESULT NOT EQUAL ZERO
  DISPLAY "Error calling Web Service: " RESULT-STRING
ELSE
  DISPLAY "Output:" OUTPUTDATA,
  DISPLAY "Status:" STATUSLINE.
```

Cloud – Application Integration

- ▣ SaaS can be accessed by mainframe applications
- ▣ Application invokes middleware
- ▣ Middleware Processing
 - Remap COBOL 01 Request to XML
 - Format SOAP/JSON Request
 - Add HTTP envelope
 - Connect via SSL to SaaS server
 - Send request, receive response
 - Extract SOAP/JSON Response from HTTP
 - Remap XML to COBOL 01 Response

Cloud – Application Integration

- Many Software as a Service Cloud Services are available today
- Robust middleware exists to allow mainframe applications to directly access these services
- Substantial advantages in using these services (eg. Sales tax calculation by sale location)

MGS INC
Computer Business Solutions

Example Services:

Office Time (time tracking)

Sage One (accounting)

SalesForce

DocuSign

Microsoft Office 365

Intuit QuickBooks

Intuit Online Payroll

Abukai Expenses (reporting/mgt)

Google Drive, SkyDrive

Microsoft Dynamics 365

Cloud – Application Integration

▣ Caveats

- Cloud Services are only as reliable as their provider
- Once functionality leaves the data center
 - ❖ Costs may be reduced
 - ❖ Control is lost
- Must be evaluated on an application-by-application basis
- Security different for each service

Additional Questions?

Michael S. Recant
VP Software Development

MGS, Inc.
11506 Allecingie Pkwy, Suite 2B
Richmond, VA 23235

Voice: (804)379-0230
Fax: (804)379-1299
Email: Mike.Recant@mgsinc.com
Web: www.mgsinc.com


Computer Business Solutions

UNITE 2020 Conference

**Web Services Are Always
Cloud Ready!**

MGS INC
Computer Business Solutions

This presentation is available on our WEB site
<http://www.mgsinc.com/download.html>

38