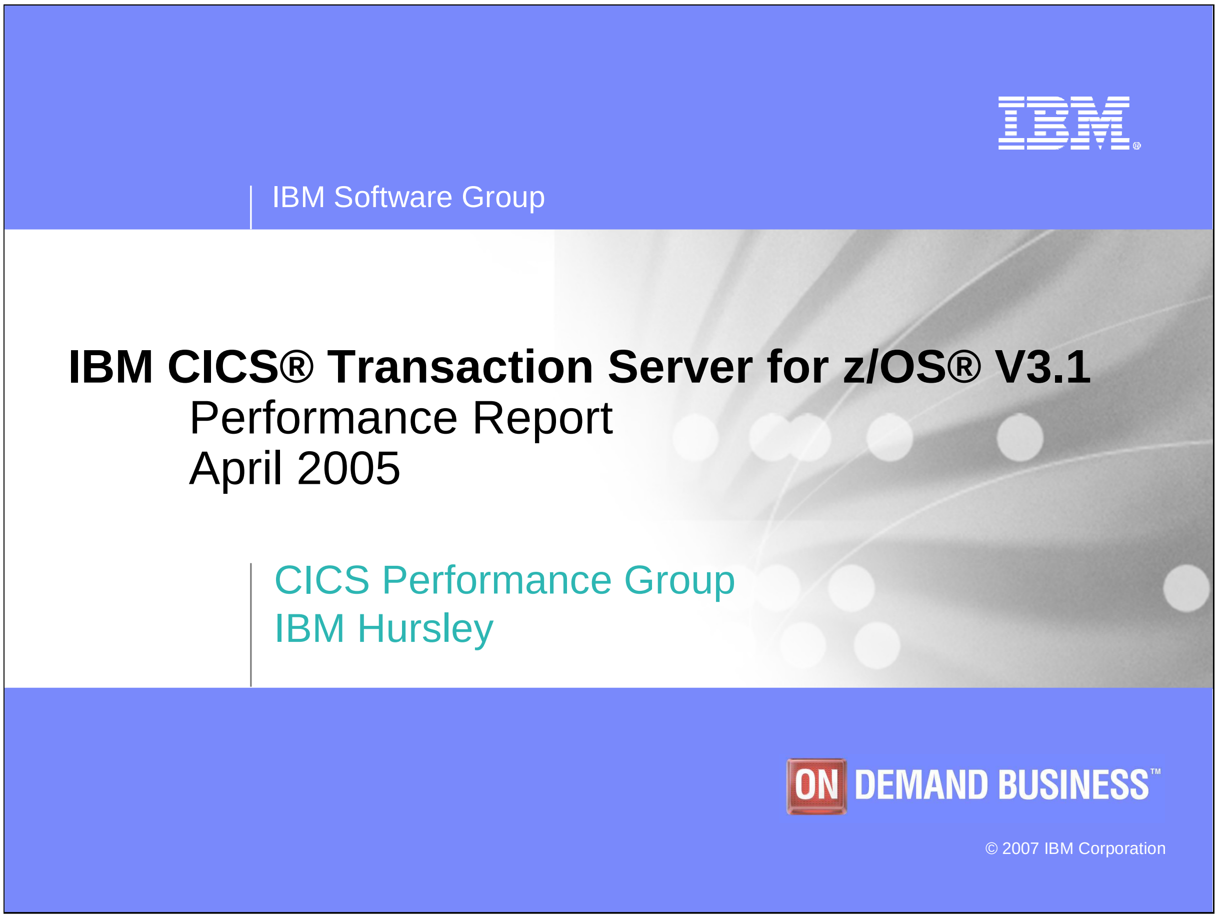
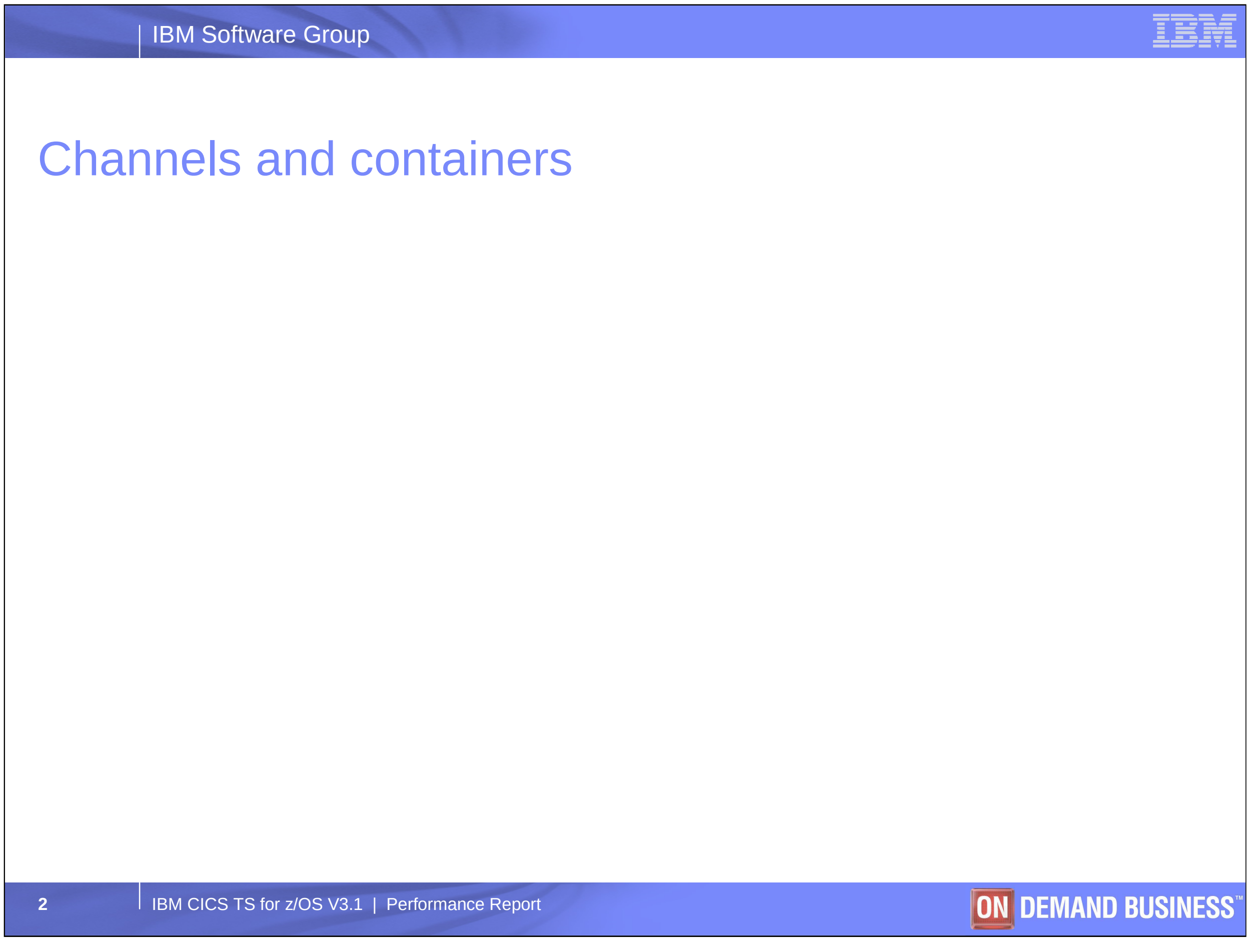


IBM Software Group

IBM CICS® Transaction Server for z/OS® V3.1
Performance Report
April 2005

CICS Performance Group
IBM Hursley

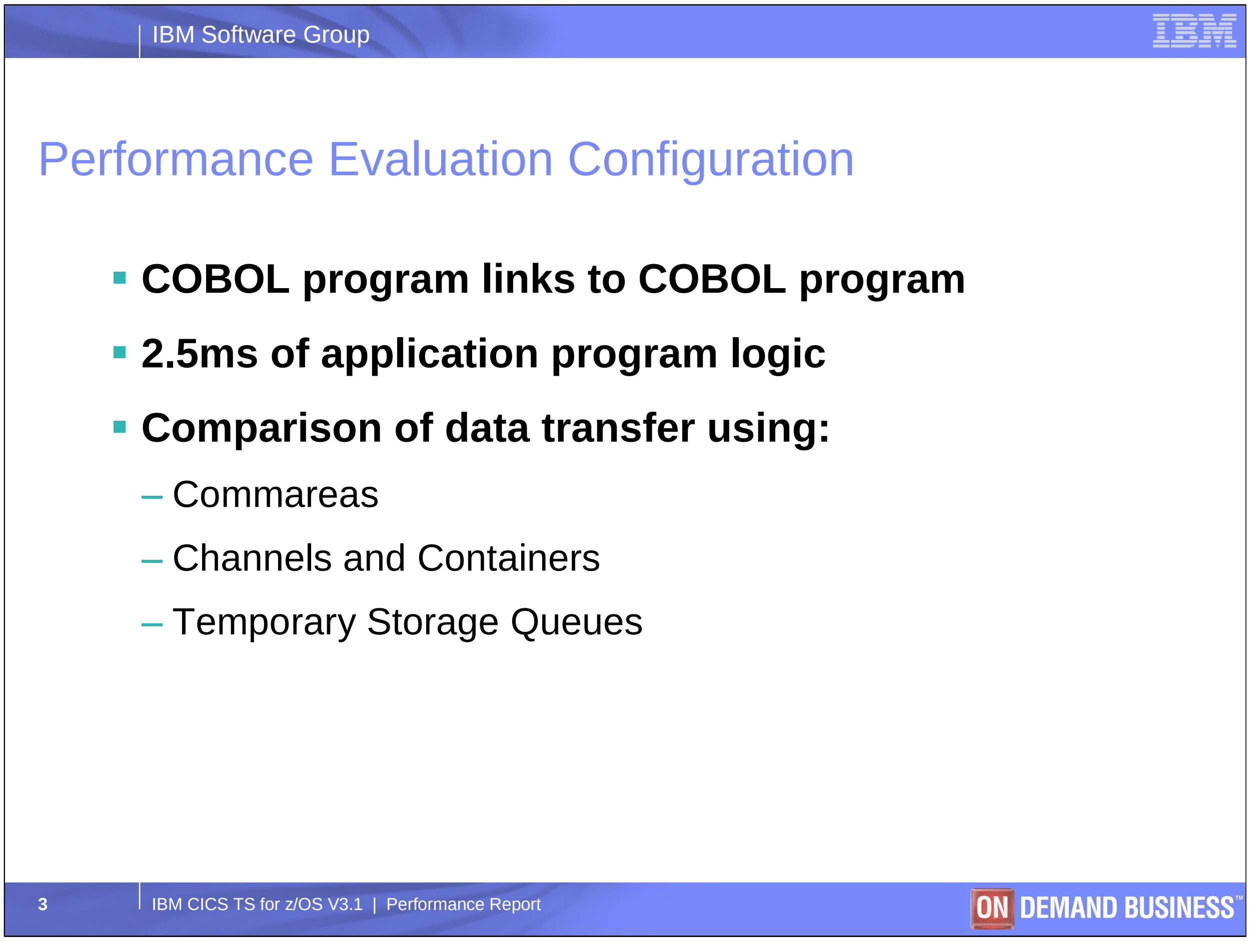
ON DEMAND BUSINESS™



Channels and containers

In CTS 3.1, data can be passed from program to program in Containers. Each Container is associated with a Channel

The main advantage of using Containers instead of Commareas for passing data, is that the data size can exceed 32K



IBM Software Group

Performance Evaluation Configuration

- **COBOL program links to COBOL program**
- **2.5ms of application program logic**
- **Comparison of data transfer using:**
  - Commareas
  - Channels and Containers
  - Temporary Storage Queues

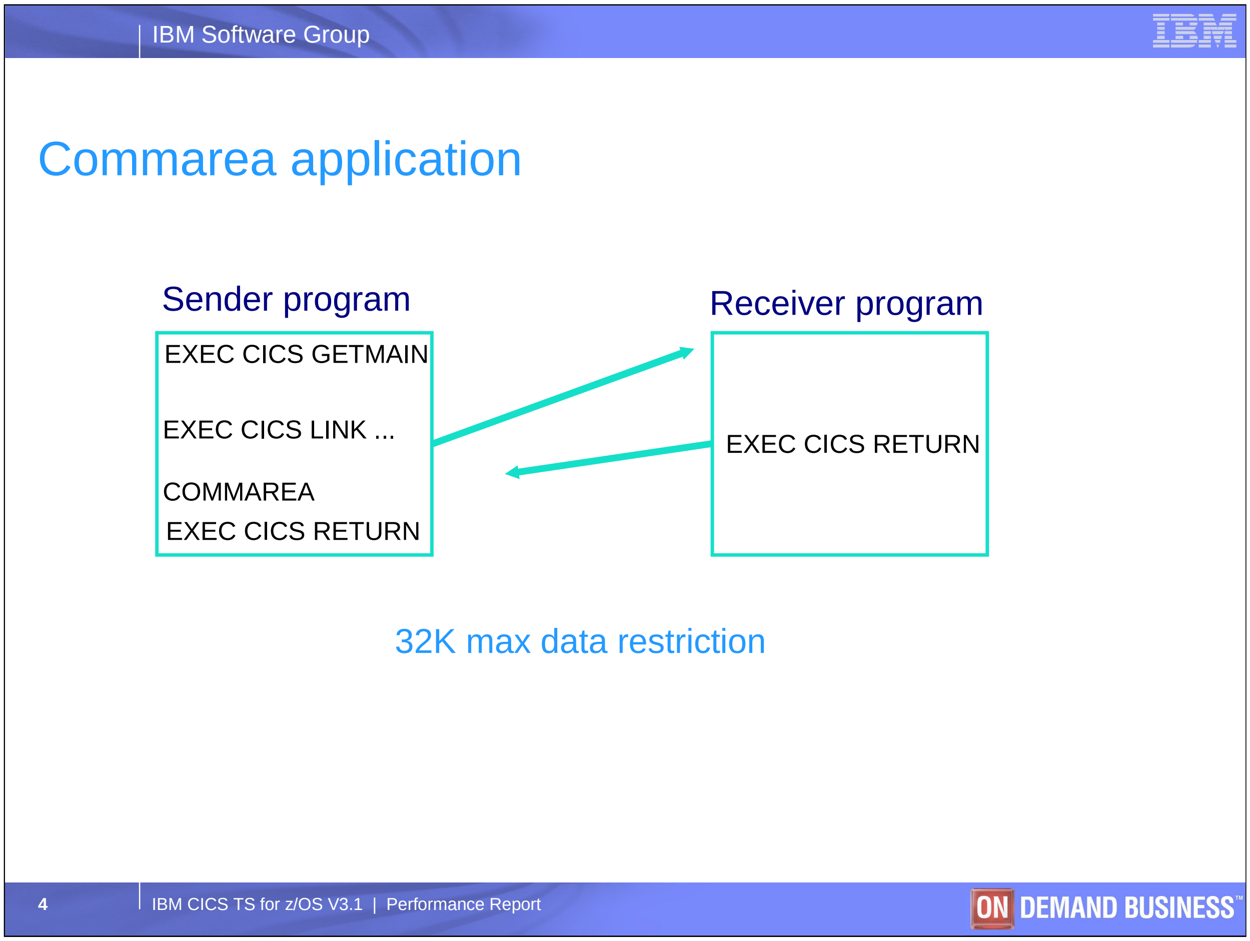
3 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

The perfromance evaluations involved passing data between 2 cobol programs running in the same region, and also in MRO-connected regions

Three methods for passing data were evaluated:

1. Commareas
2. Containers
3. Temporary Storage



Commarea application

Sender program

EXEC CICS GETMAIN

EXEC CICS LINK ...

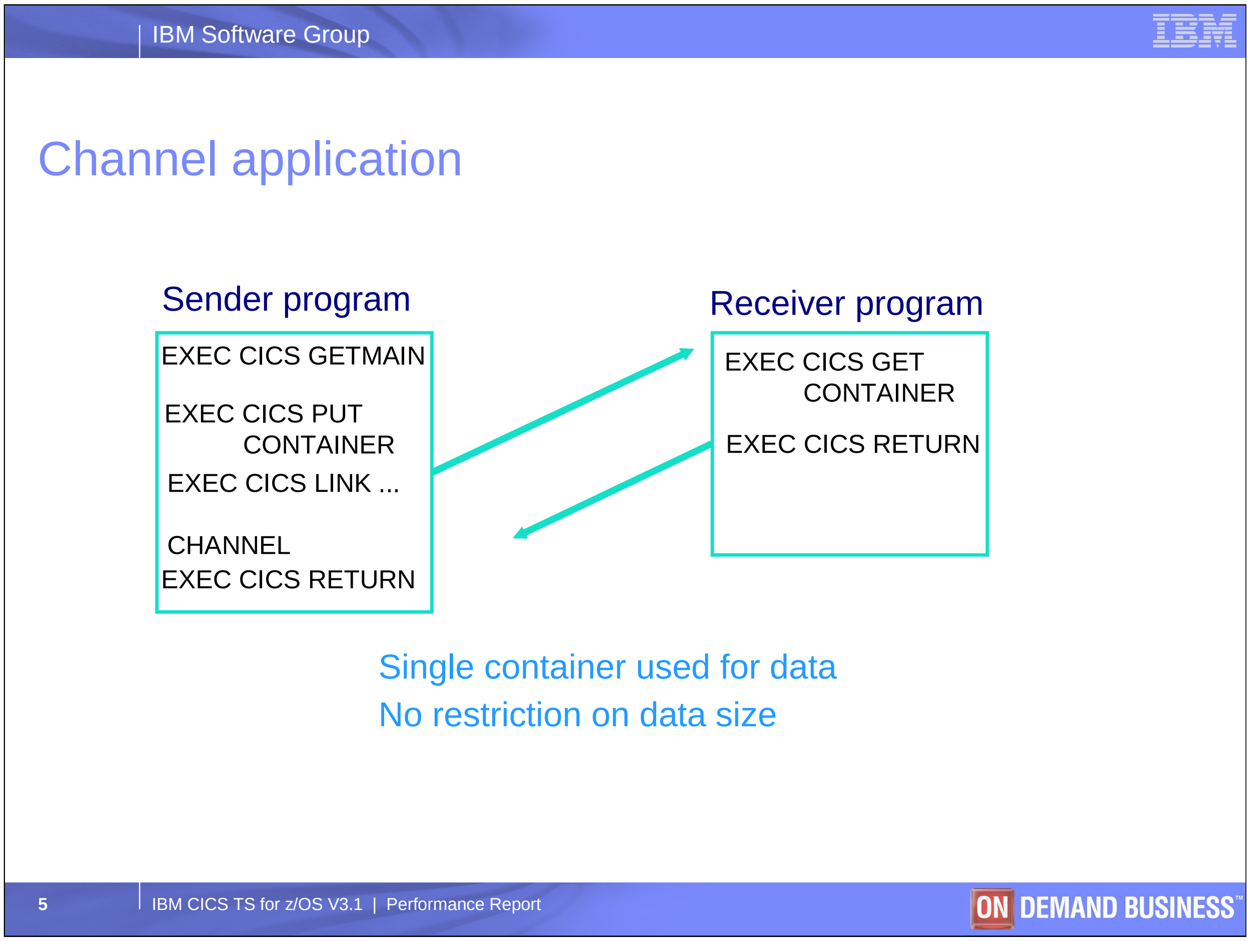
COMMAREA

EXEC CICS RETURN

Receiver program

EXEC CICS RETURN

32K max data restriction



IBM Software Group

Channel application

Sender program

```
EXEC CICS GETMAIN

EXEC CICS PUT
          CONTAINER
EXEC CICS LINK ...

CHANNEL
EXEC CICS RETURN
```

Receiver program

```
EXEC CICS GET
          CONTAINER

EXEC CICS RETURN
```

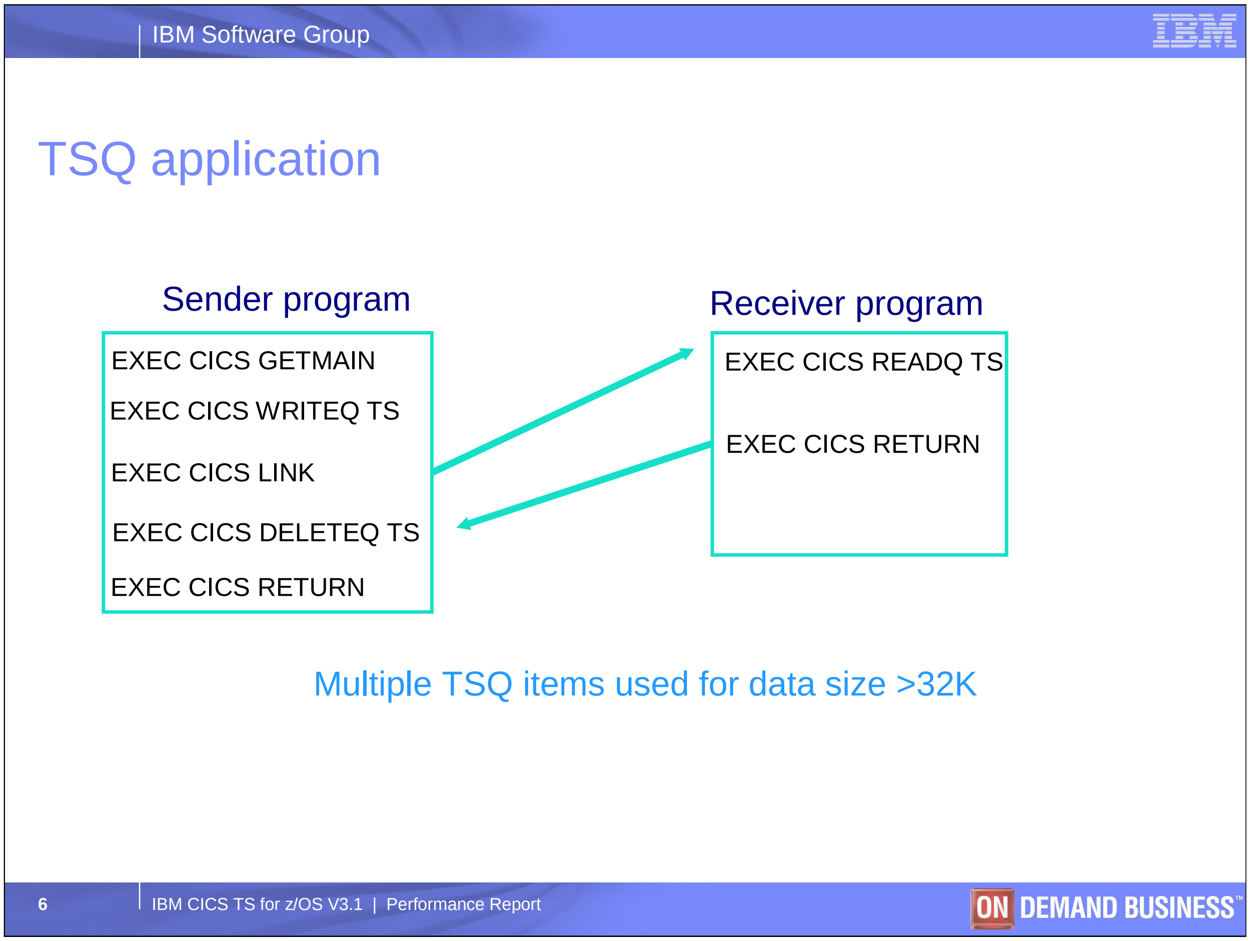
Single container used for data

No restriction on data size

5 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

A single container within the channel was used in these tests



TSQ application

Sender program

EXEC CICS GETMAIN

EXEC CICS WRITEQ TS

EXEC CICS LINK

EXEC CICS DELETEQ TS

EXEC CICS RETURN

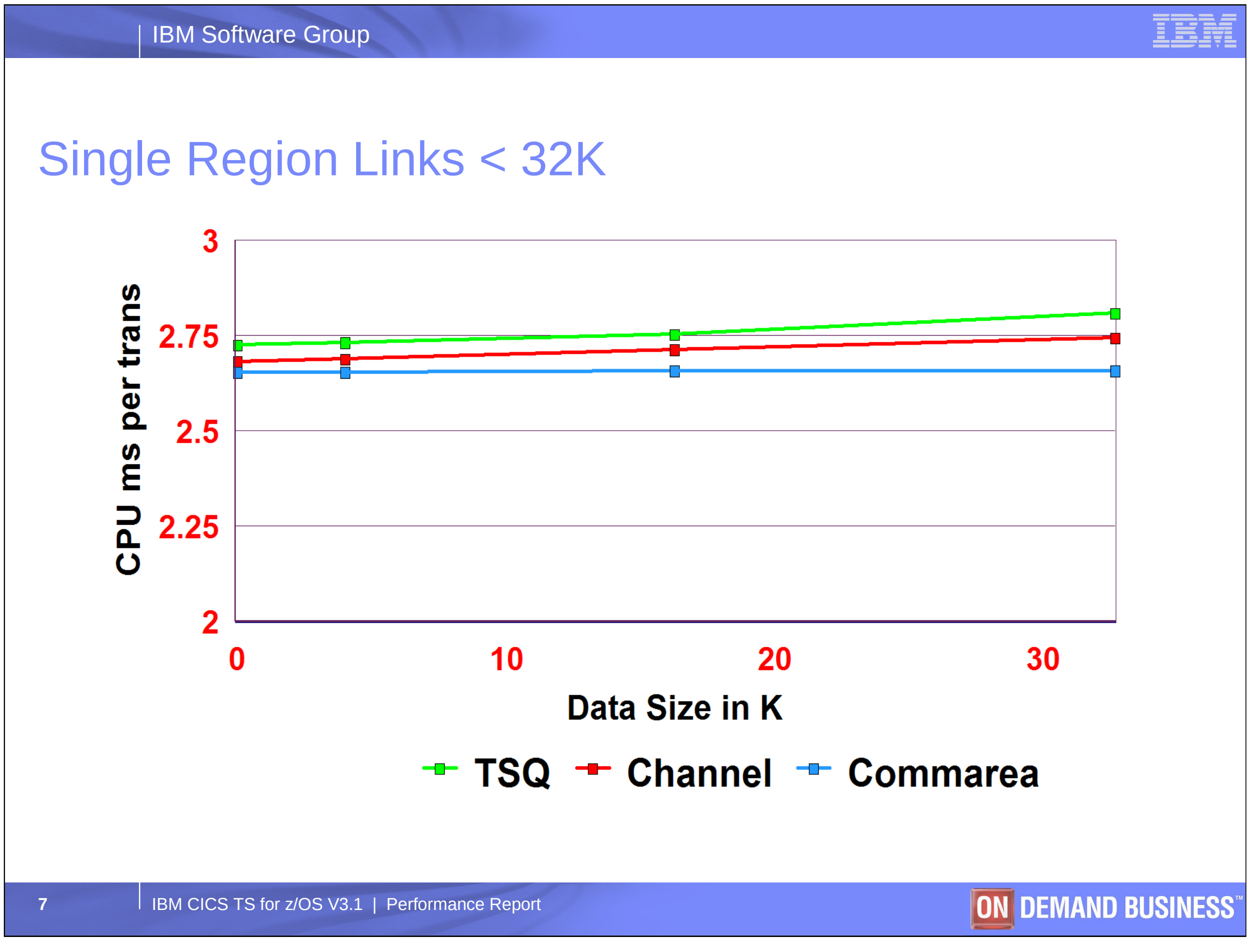
Receiver program

EXEC CICS READQ TS

EXEC CICS RETURN

Multiple TSQ items used for data size >32K

To send more than 32K, sending program issues multiple TSQ writes, and receiving program issues multiple TSQ reads

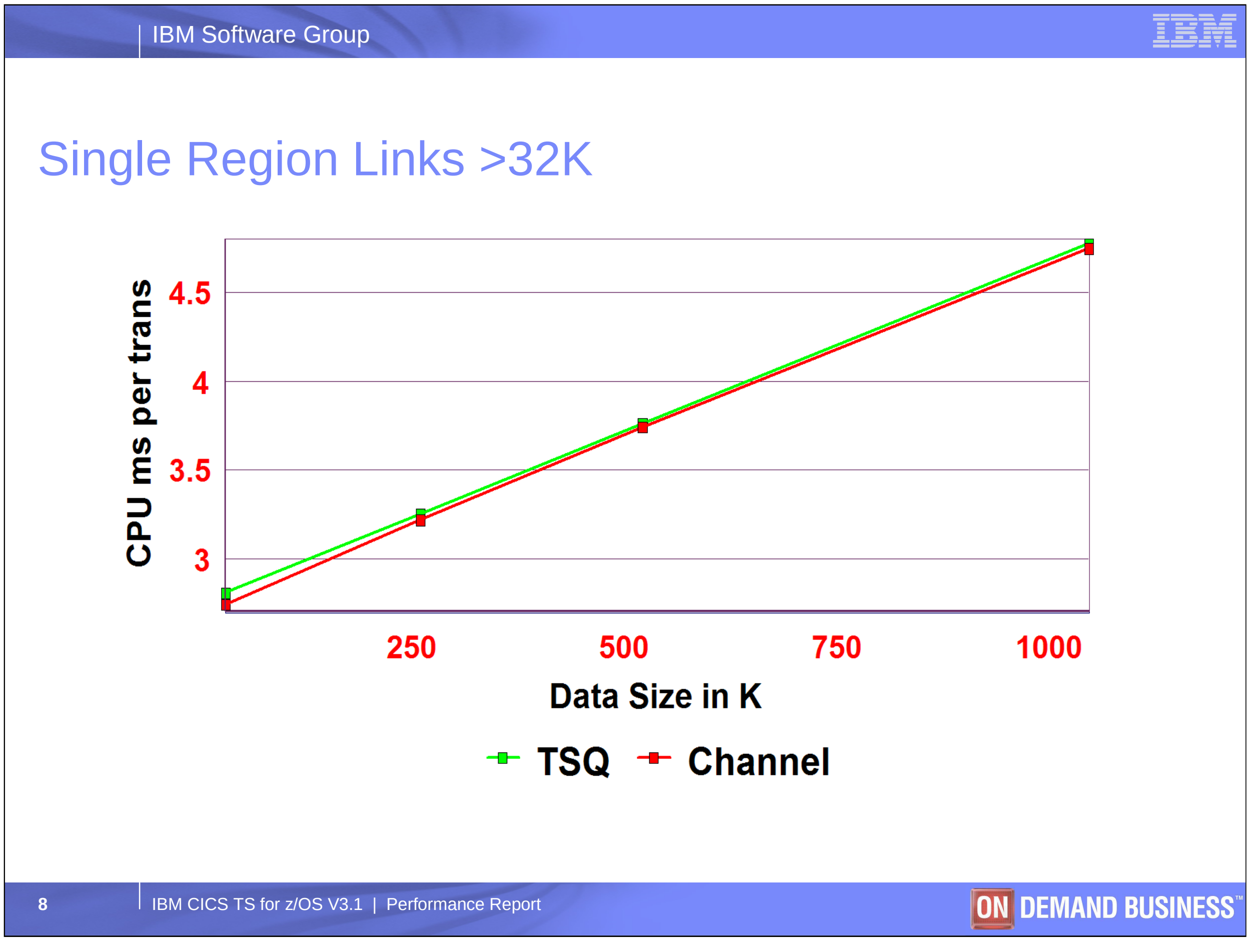


IBM Software Group

Single Region Links < 32K

7 IBM CICS TS for z/OS V3.1 | Performance Report

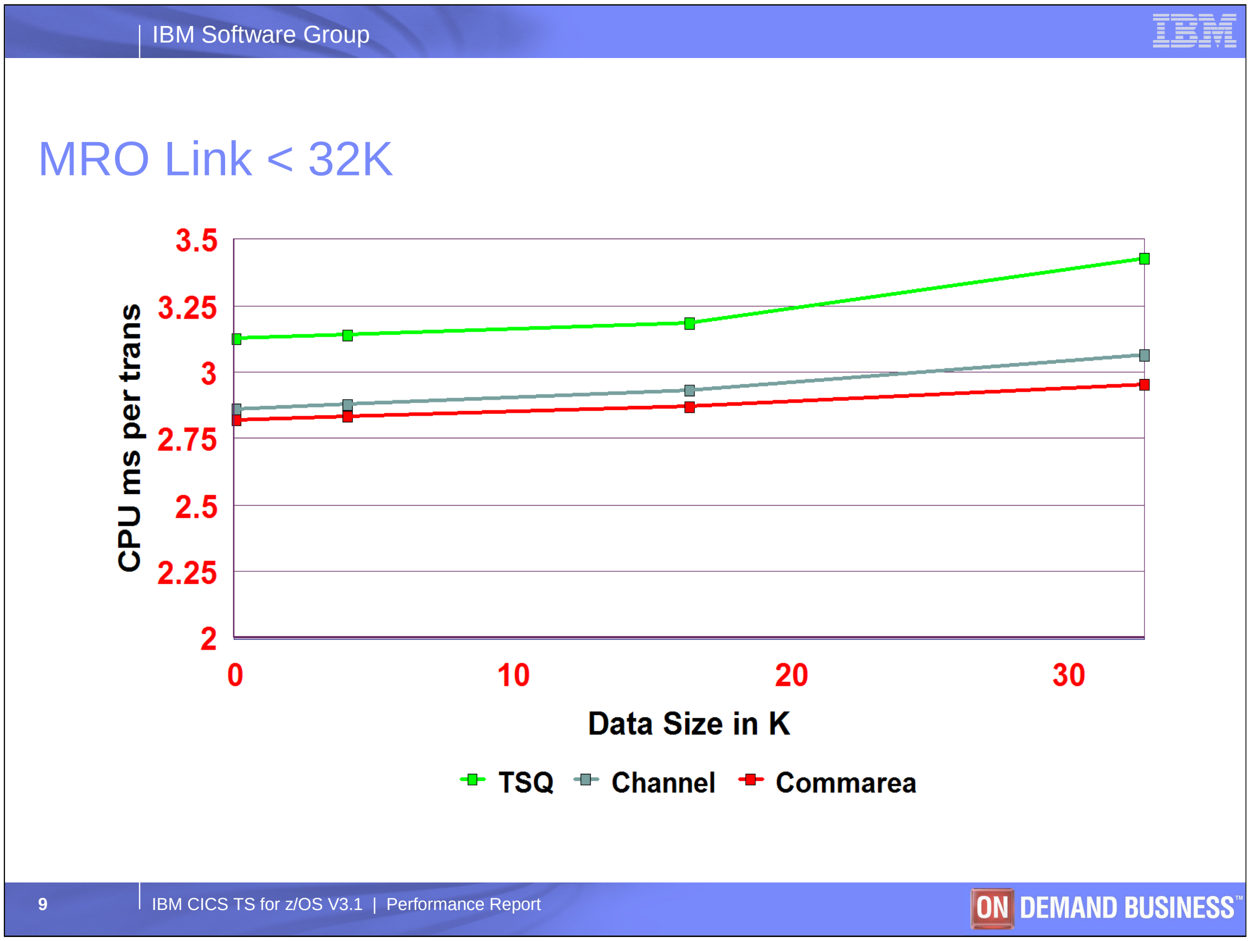
Application that is called has about 2.5 milli secs of application logic. Roughly in the region of 1000000 machine instructions.



Single Region Links >32K

Commareas cannot be used for data sizes >32K.

Due to the 32K restriction on TS item size, mutiple WRITEQ TS and READQ TS commands are issued.



IBM Software Group

MRO Link < 32K

9 IBM CICS TS for z/OS V3.1 | Performance Report

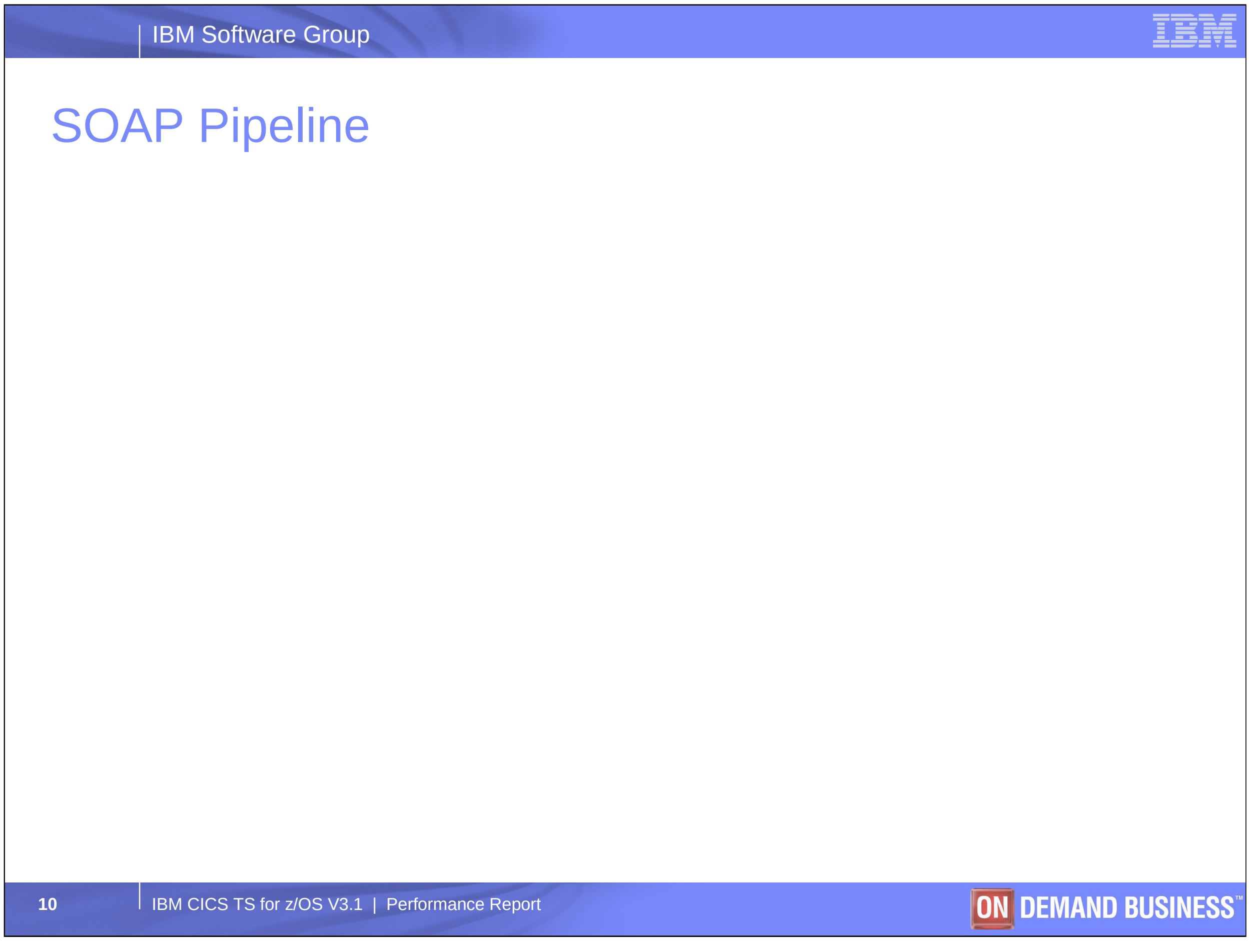
ON DEMAND BUSINESS™

For these tests, an MRO XM link was established between two CICS systems running in the same LPAR

TS queues on the local CICS system were defined as remote using TSMODEL definitions. READQ, WRITEQ and DELETEQ requests issued on the local system were handled by mirror transactions on the remote system. This is a significant overhead per transaction.

Commareas slightly out performed Channels for MRO. TS queues performed significantly worse.

When the size of data being transferred exceeds 32K, Commareas can not be used (due to the 32K restriction), and the performance of TS queues degrades. For Channels, the cpu increase remains linear.



IBM Software Group

SOAP Pipeline

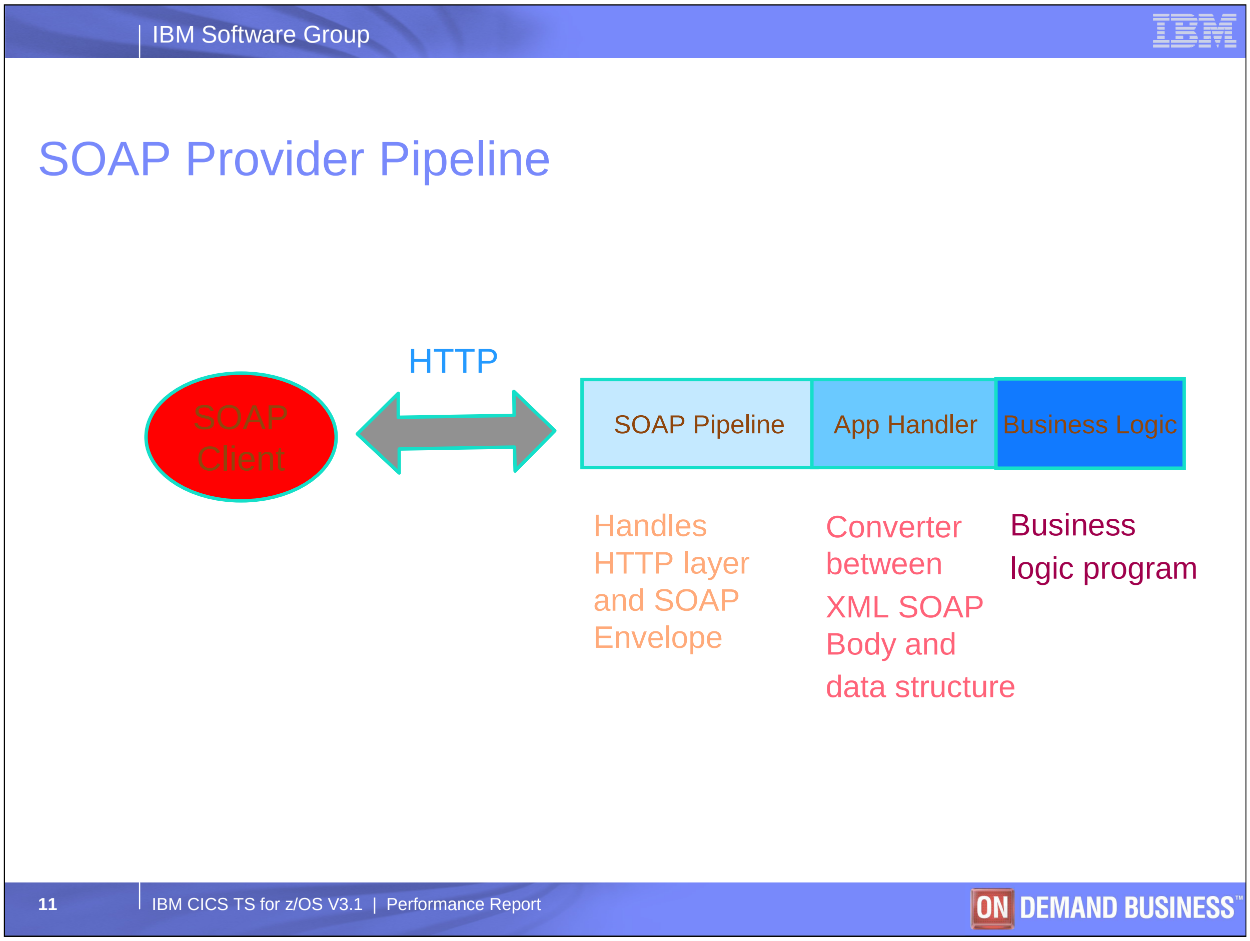
10 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

SOAP support in CICS is implemented as a pipeline, and supports both requester and provider business logic applications

These applications pass data in language data structures and do not get involved in XML SOAP structures

A SOAP message consists of a SOAP Body and optionally SOAP Headers contained within a SOAP Envelope



IBM Software Group

SOAP Provider Pipeline

SOAP Client

HTTP

SOAP Pipeline | App Handler | Business Logic

Handles HTTP layer and SOAP Envelope

Converter between XML SOAP Body and data structure

Business logic program

11 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

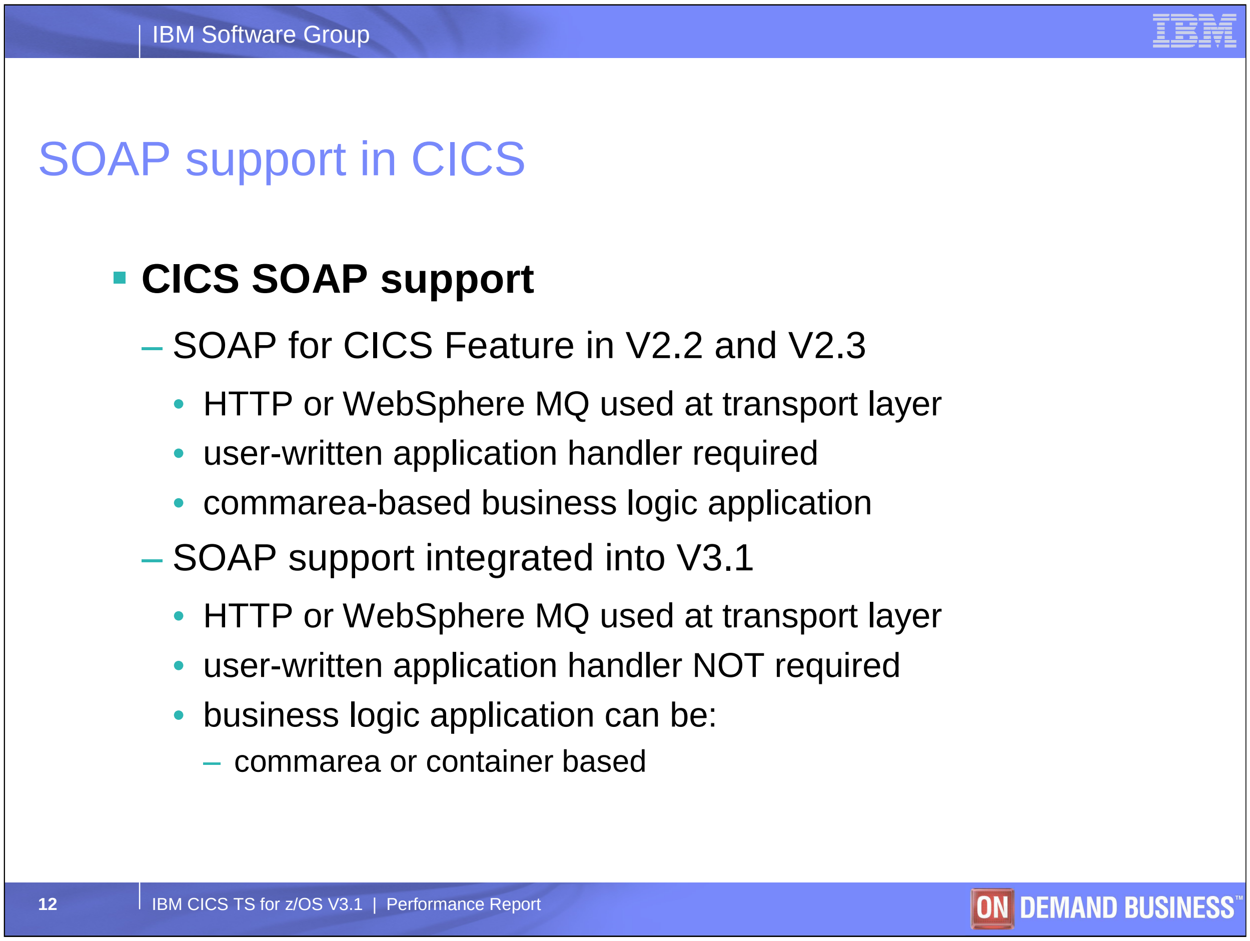
SOAP Client - TPNS and WebSphere® Studio Workload Simulator were used

HTTP or WMQ (WebSphere MQ) supported at the transport layer

The SOAP Pipeline in V3.1 is new, and replaces the SOAP for CICS Feature in V2.2 and V2.3

App Handler - The application handler converts the SOAP Body to a data structure on input, and converts the data structure to a SOAP Body on output

Business logic program - commarea based, but in V3.1 can alternatively be container based. Null business logic program used in tests



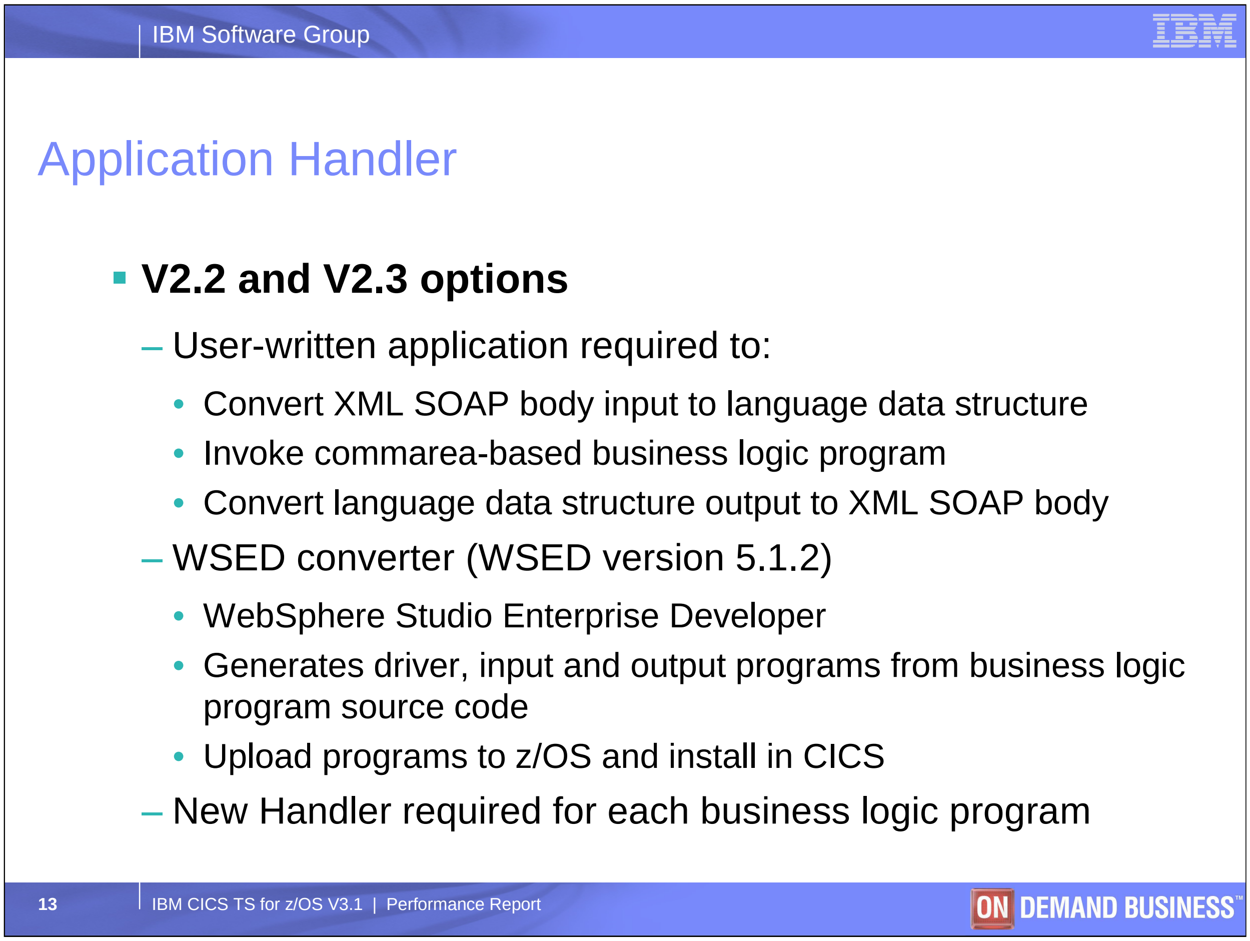
SOAP support in CICS

- **CICS SOAP support**
  - SOAP for CICS Feature in V2.2 and V2.3
    - HTTP or WebSphere MQ used at transport layer
    - user-written application handler required
    - commarea-based business logic application
  - SOAP support integrated into V3.1
    - HTTP or WebSphere MQ used at transport layer
    - user-written application handler NOT required
    - business logic application can be:
      - commarea or container based

User-written application handler converts XML SOAP body to language data structure on input and vice versa on output

CICS Web Services Assistant replaces need for user-written application handler. Batch program generates a wsbind file that is used at runtime to convert between the SOAP body and language structures

CICS as a SOAP requester must use the container interface in V31

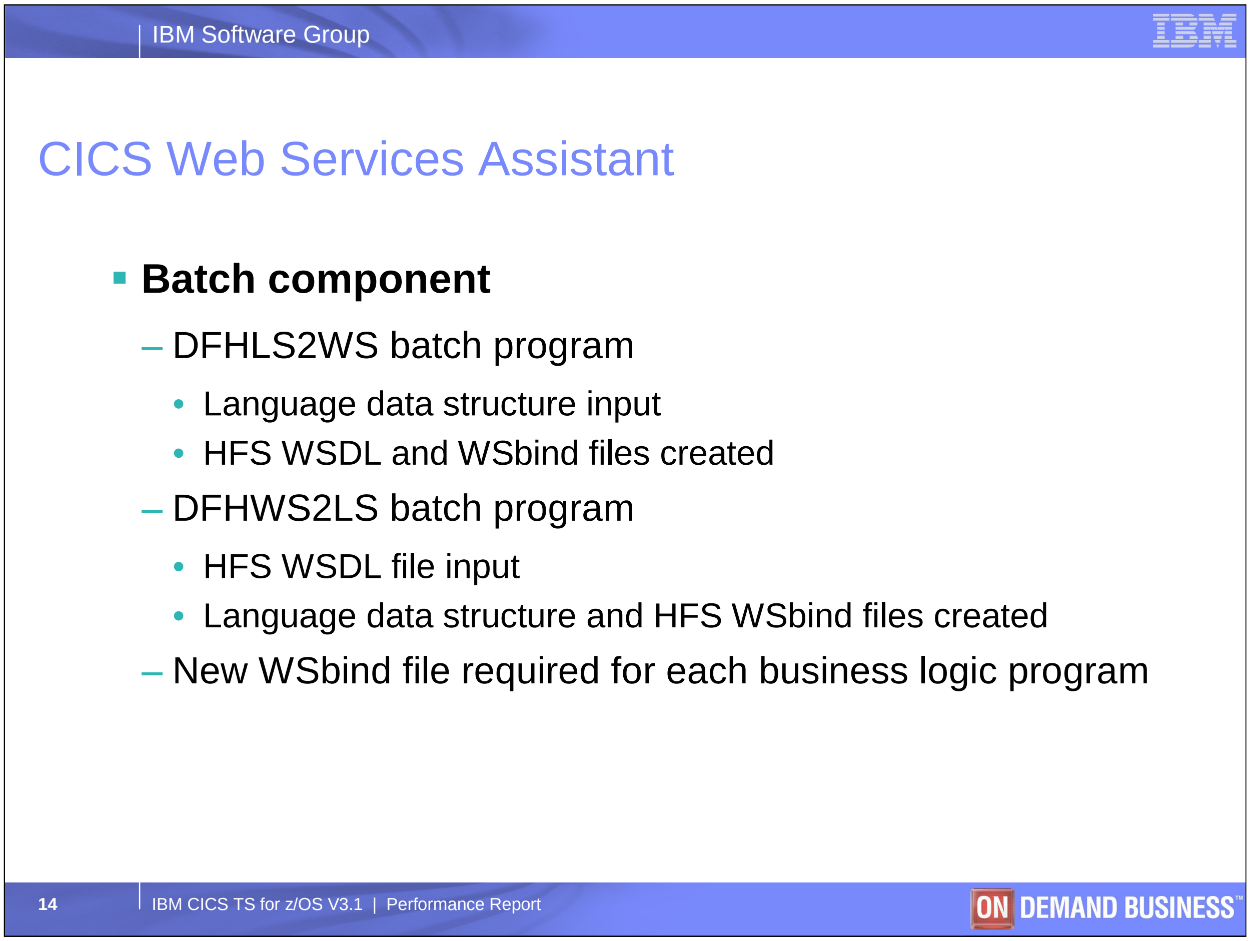


Application Handler

- **V2.2 and V2.3 options**
  - User-written application required to:
    - Convert XML SOAP body input to language data structure
    - Invoke commarea-based business logic program
    - Convert language data structure output to XML SOAP body
  - WSED converter (WSED version 5.1.2)
    - WebSphere Studio Enterprise Developer
    - Generates driver, input and output programs from business logic program source code
    - Upload programs to z/OS and install in CICS
  - New Handler required for each business logic program

WSED runs on Windows

WSED generates source code, so needs compiling on z/OS



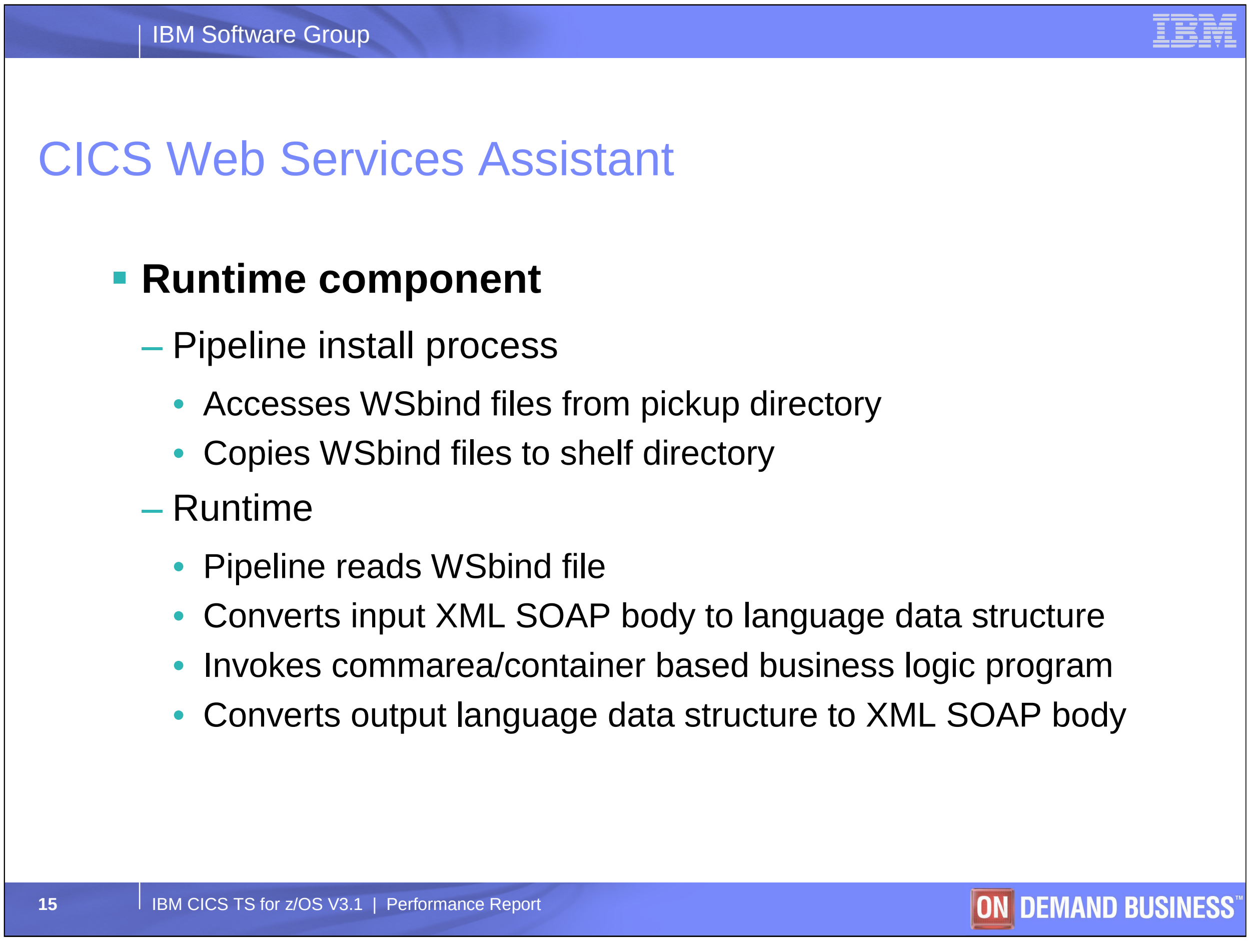
CICS Web Services Assistant

- **Batch component**
  - DFHLS2WS batch program
    - Language data structure input
    - HFS WSDL and WSbind files created
  - DFHWS2LS batch program
    - HFS WSDL file input
    - Language data structure and HFS WSbind files created
  - New WSbind file required for each business logic program

WSDL is the Web Services Definition Language, and descibes in XML format how to invoke the Web Service, in terms of transport layers supported, the URI required, operations (methods) supported and descriptions of the data elements

DFHLS2WS typically used when running a service provider application. Generated WSDL then used by service requester to create SOAP message

DFHWS2LS typically used when running a service requester application. WSDL supplied by service provider

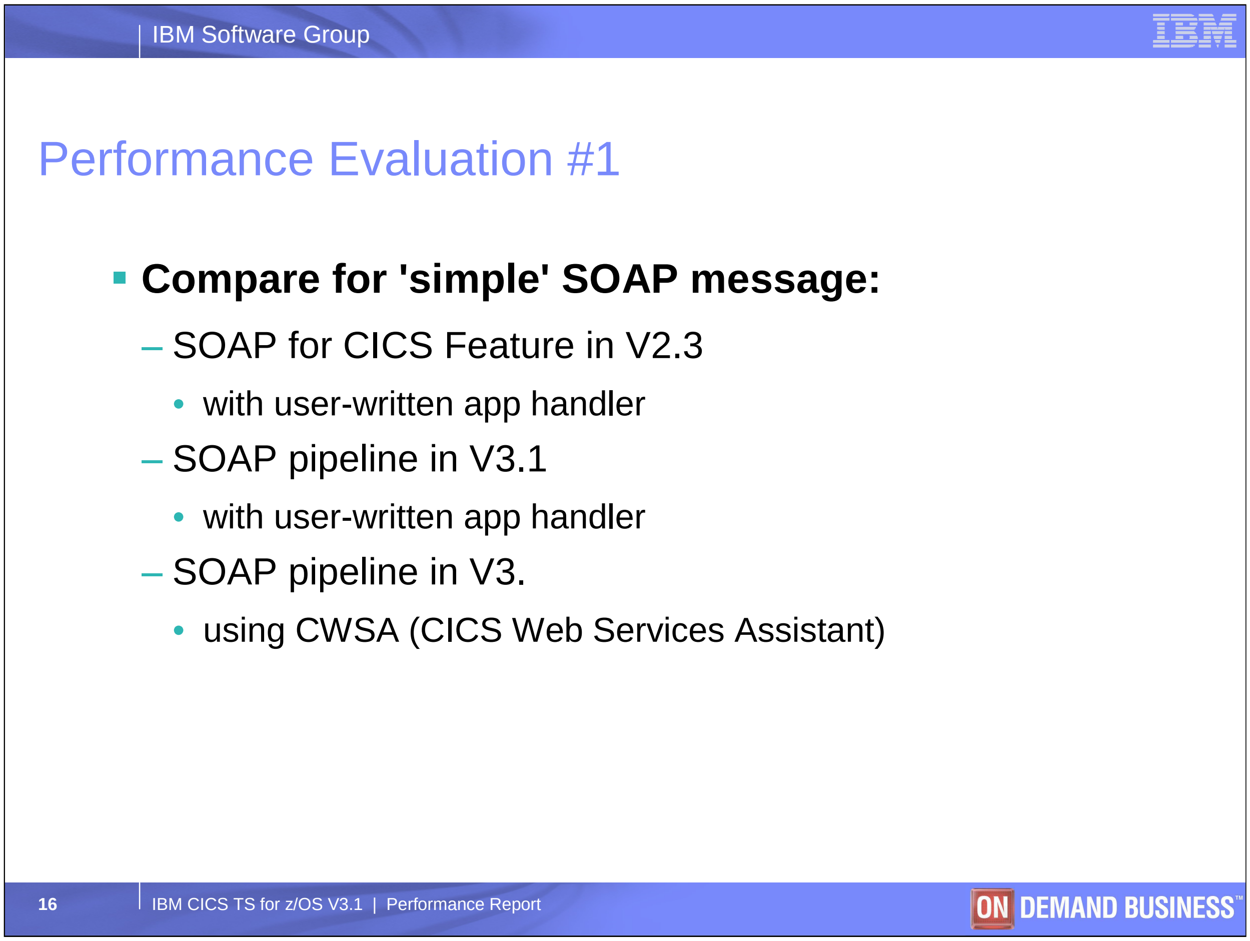


CICS Web Services Assistant

- **Runtime component**
  - Pipeline install process
    - Accesses WSbind files from pickup directory
    - Copies WSbind files to shelf directory
  - Runtime
    - Pipeline reads WSbind file
    - Converts input XML SOAP body to language data structure
    - Invokes commarea/container based business logic program
    - Converts output language data structure to XML SOAP body

The CEDA pipeline install will install all applicable Web Services, by reading the WSbind files from the Pickup directory and copying them to the Shelf directory

Application handler DFHPITP invokes the CICS Web Services Assistant



Performance Evaluation #1

- **Compare for 'simple' SOAP message:**
  - SOAP for CICS Feature in V2.3
    - with user-written app handler
  - SOAP pipeline in V3.1
    - with user-written app handler
  - SOAP pipeline in V3.
    - using CWSA (CICS Web Services Assistant)

The 'simple' SOAP message consists of one element in, one element out

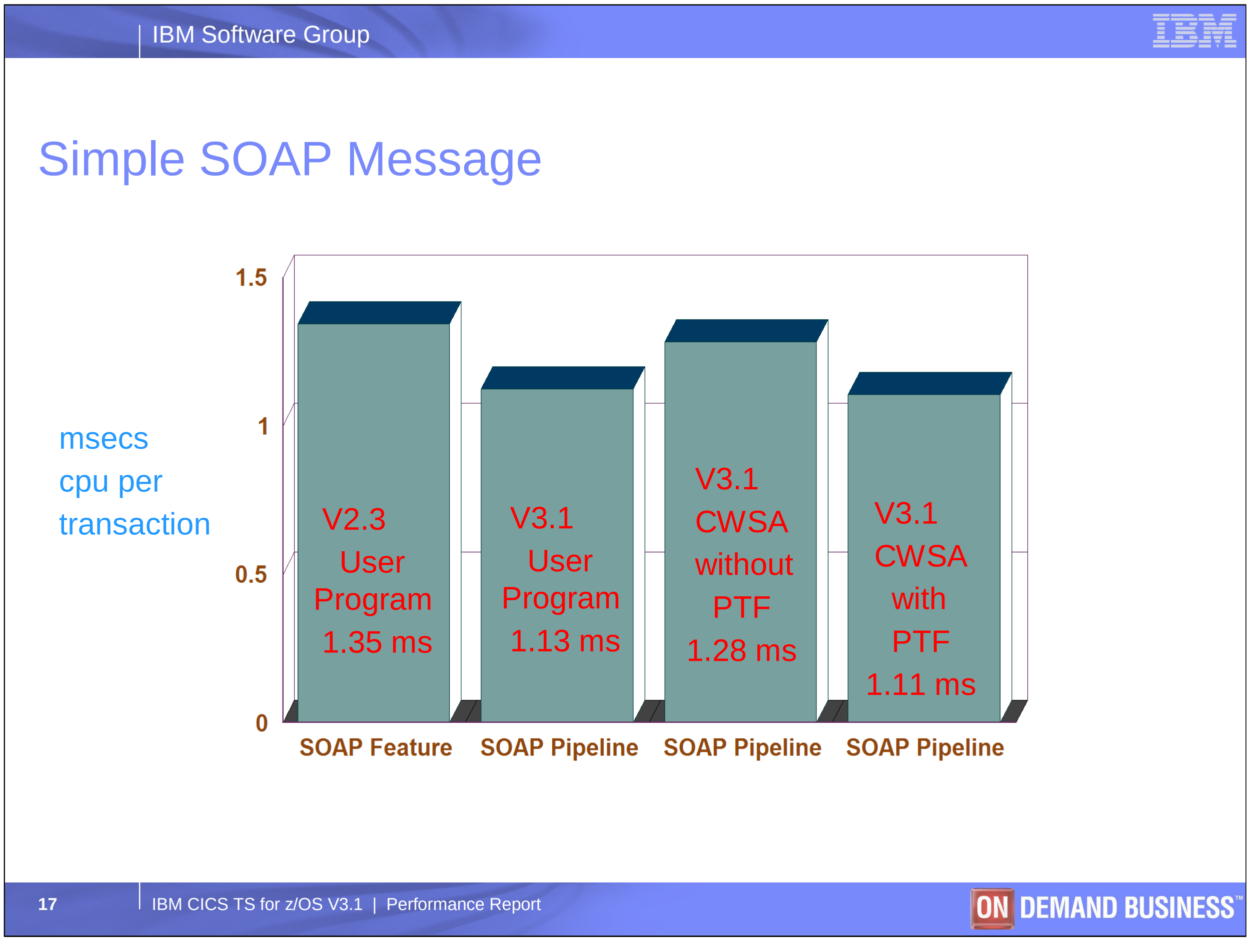
Same user-written app handler and null business logic program

User-written app handler uses XML PARSE for input and DOCUMENT API for output

CICS Web Services Assistant can take a language data structure used by a provider application and generate the WSDL and wsbind file. The wsbind file is used is used at run time to convert SOAP body to language data structure, and vice versa

CICS Web Services Assistant can also take WSDL and generate language data structure and wsbind file

The acronym 'CWSA' is used in this presentation to represent use of the CICS Web Services Assistant



IBM Software Group

Simple SOAP Message

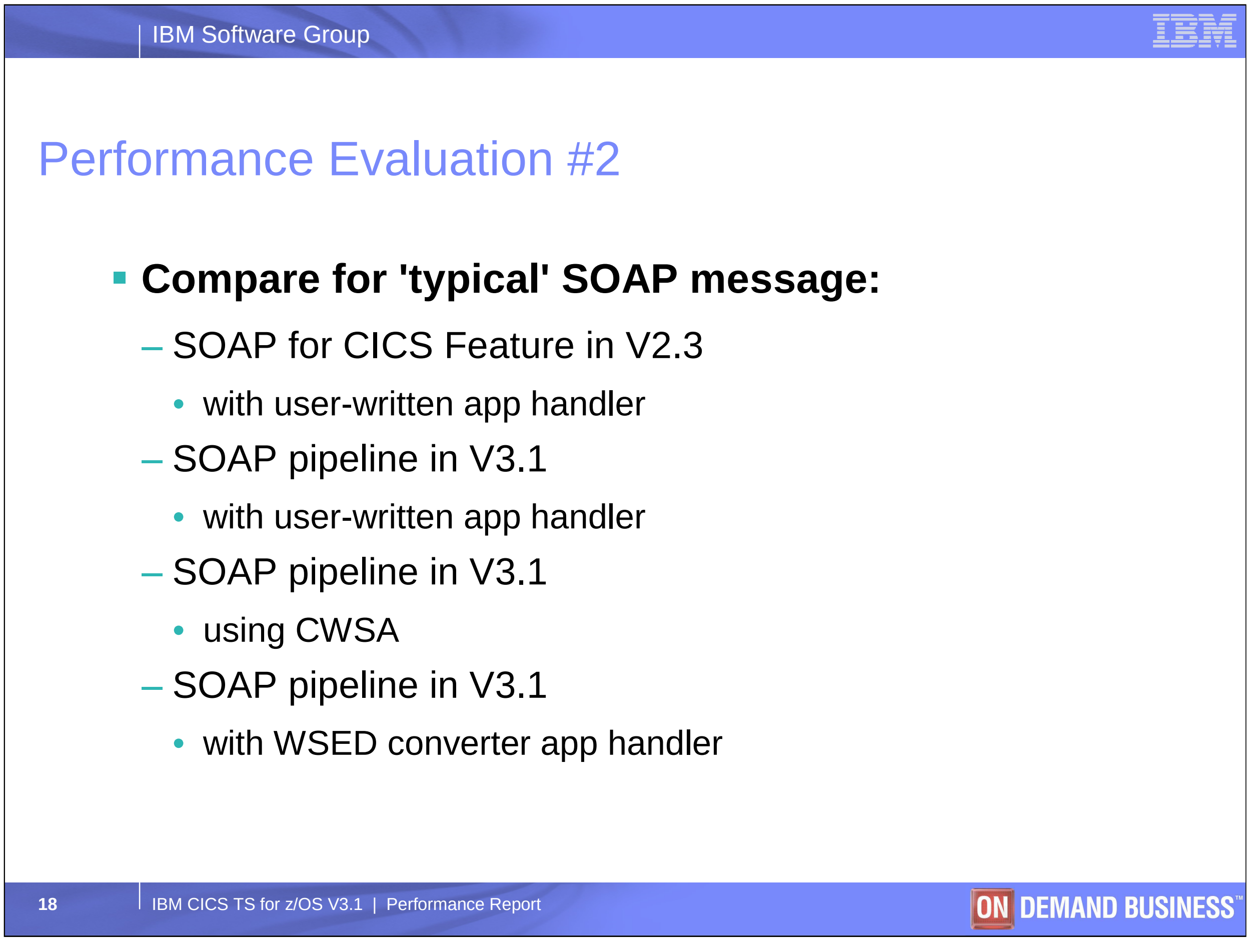
17 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

PTF peformance fix - apar PK03397

CPU per tran includes CWXN+CWBA for feature and CWXN+CPIH for V3.1 plus the CSOL TCP/IP listener task

CWXN handles initial HTTP input and attaches CPIH which processes SOAP pipeline



Performance Evaluation #2

- **Compare for 'typical' SOAP message:**
  - SOAP for CICS Feature in V2.3
    - with user-written app handler
  - SOAP pipeline in V3.1
    - with user-written app handler
  - SOAP pipeline in V3.1
    - using CWSA
  - SOAP pipeline in V3.1
    - with WSED converter app handler

The 'typical' SOAP message consists of 123 elements in, 123 elements out

Cobol display, binary and packed-decimal fields used

Cobol OCCURS 9 TIMES clause

App handler uses XML PARSE for input

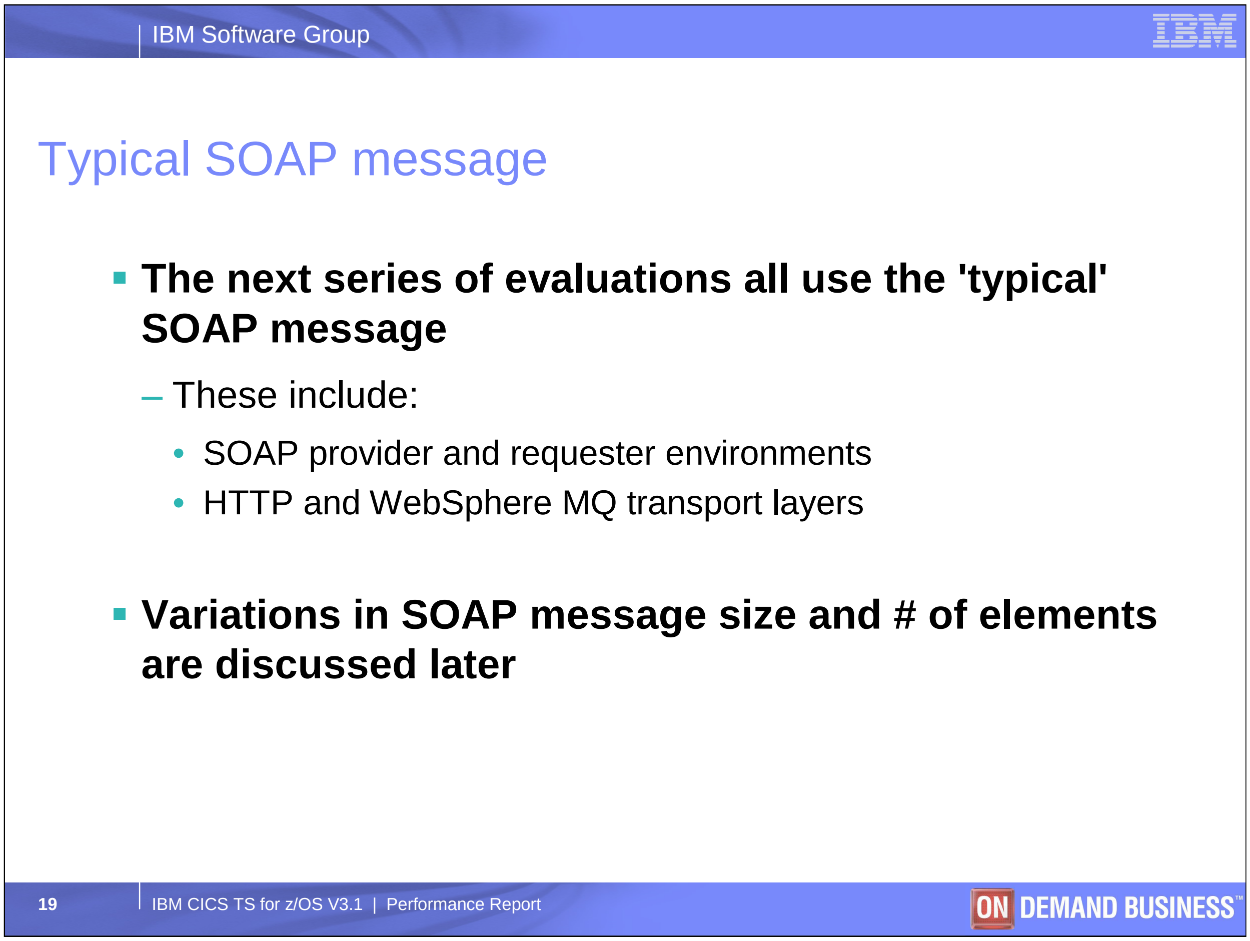
App handler uses DOCUMENT API for output

CWSA app handler uses wsbind file

WSED converter program generated to handle this specific case

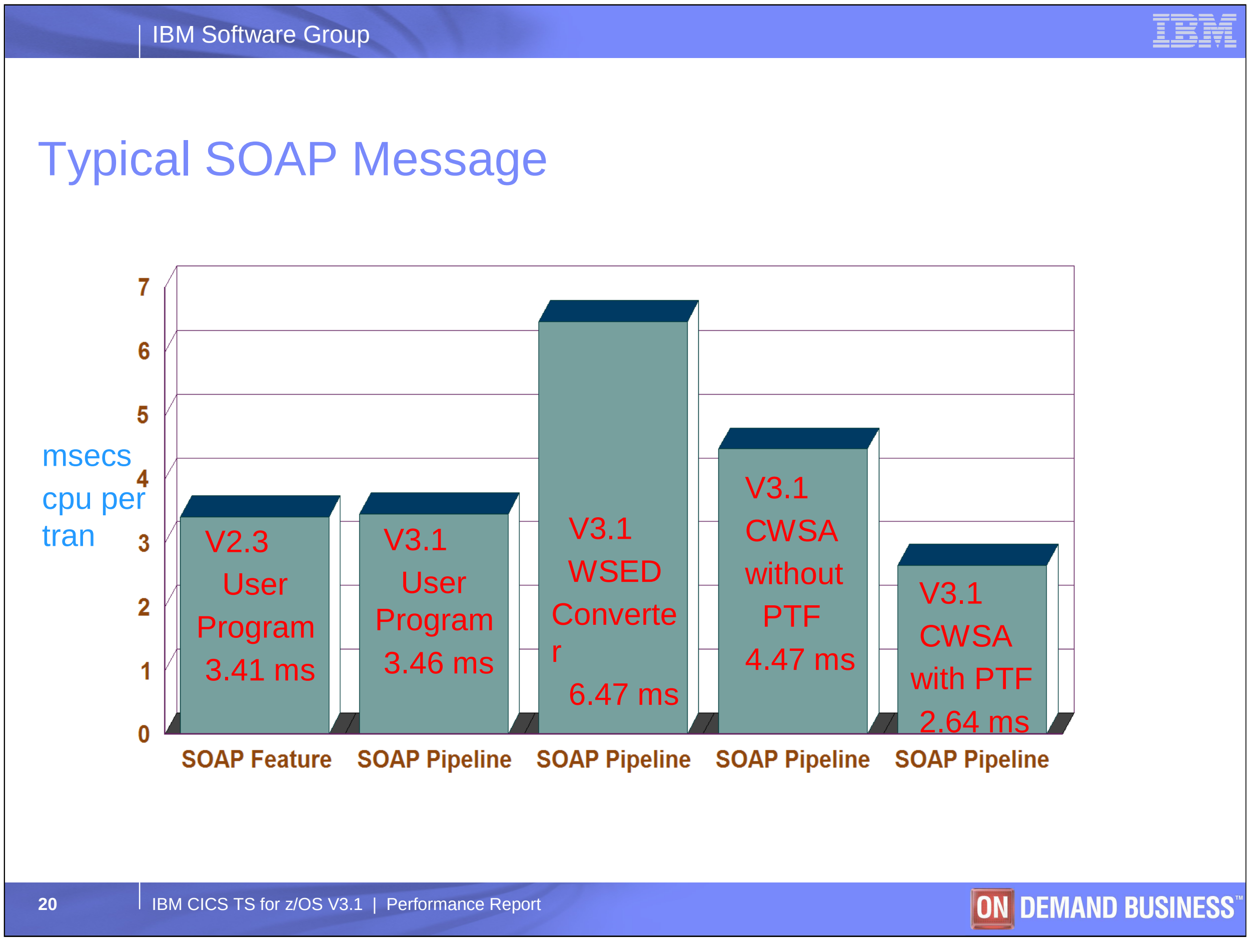
Multiple SOAP messages require multiple wsbind files with a single app handler program for CWSA

For WSED multiple application handler programs are required, one per Data Structure type.



Typical SOAP message

- **The next series of evaluations all use the 'typical' SOAP message**
  - These include:
    - SOAP provider and requester environments
    - HTTP and WebSphere MQ transport layers
- **Variations in SOAP message size and # of elements are discussed later**



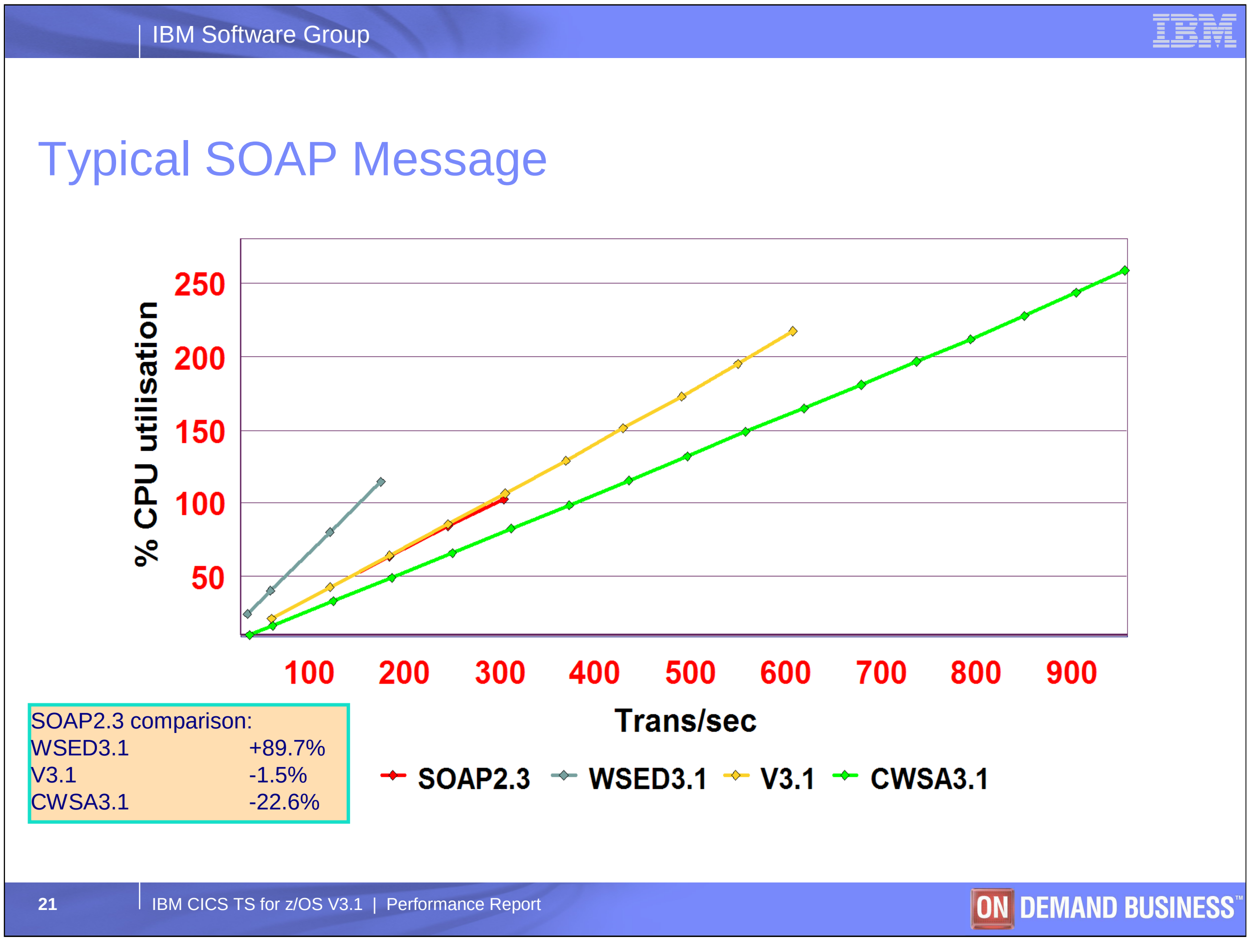
IBM Software Group

Typical SOAP Message

20 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

OPENAPI for V3.1 user program but not for WSED converter



IBM Software Group

Typical SOAP Message

SOAP2.3 comparison:	
WSED3.1	+89.7%
V3.1	-1.5%
CWSA3.1	-22.6%

21 IBM CICS TS for z/OS V3.1 | Performance Report

All subsequent performance tests are with the performance PTF

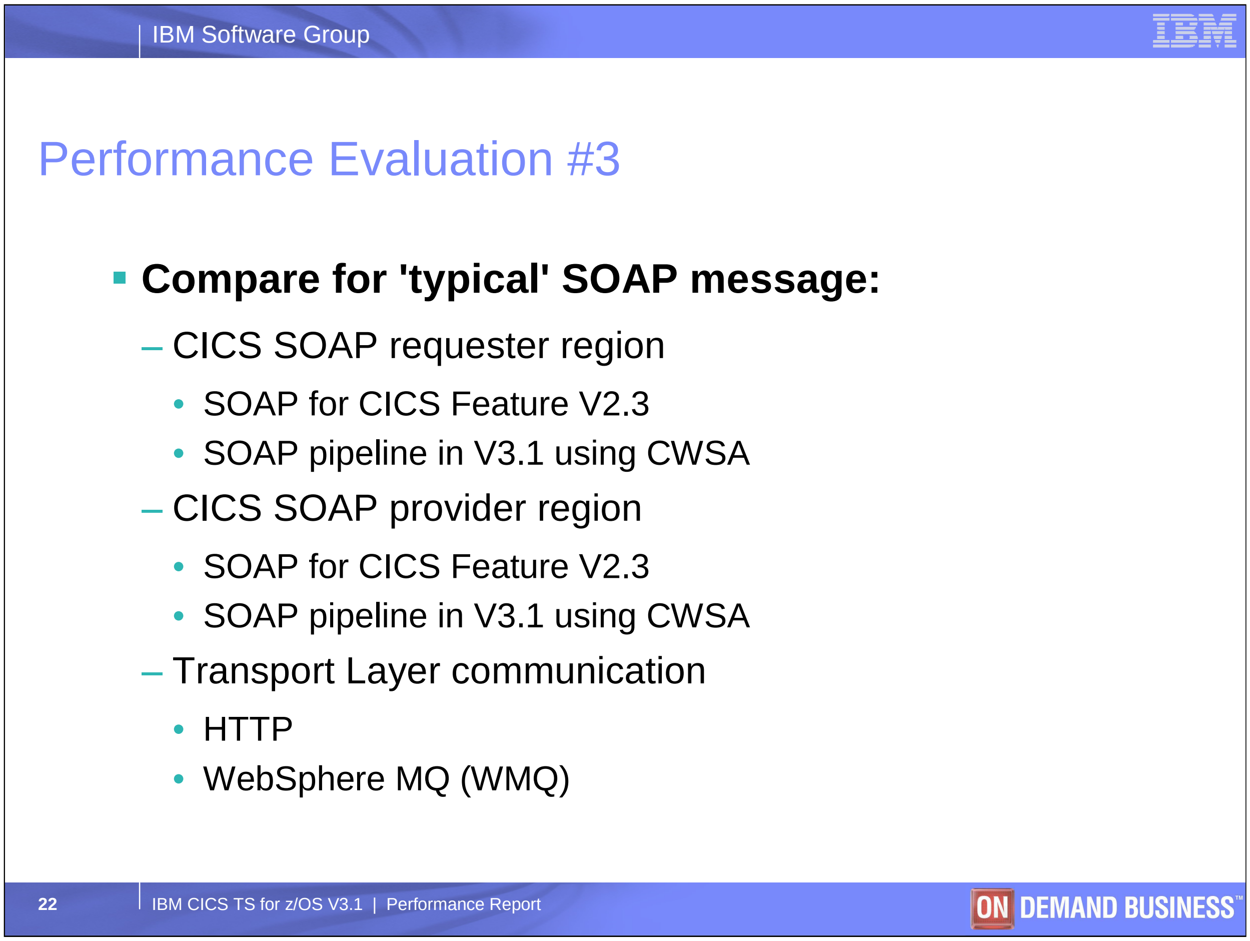
WebSphere Studio Workload Simulator used as input

A transaction consists of a CWXN + a CPIH task

The maximum throughput for our environment was 958 SOAP msgs/sec. Maximum achievable throughput will depend on the network bandwidth, size and complexity of the SOAP message, the business logic program complexity, as well as the speed and number of processors

The throughput of the SOAP pipeline has been improved by exploiting the use of OPENAPI, to make use of multiple L8 TCBs

For compaison, SOAP 2.3 is used as the base. All other % values are based on the change in cpu per transaction



Performance Evaluation #3

- **Compare for 'typical' SOAP message:**
  - CICS SOAP requester region
    - SOAP for CICS Feature V2.3
    - SOAP pipeline in V3.1 using CWSA
  - CICS SOAP provider region
    - SOAP for CICS Feature V2.3
    - SOAP pipeline in V3.1 using CWSA
  - Transport Layer communication
    - HTTP
    - WebSphere MQ (WMQ)

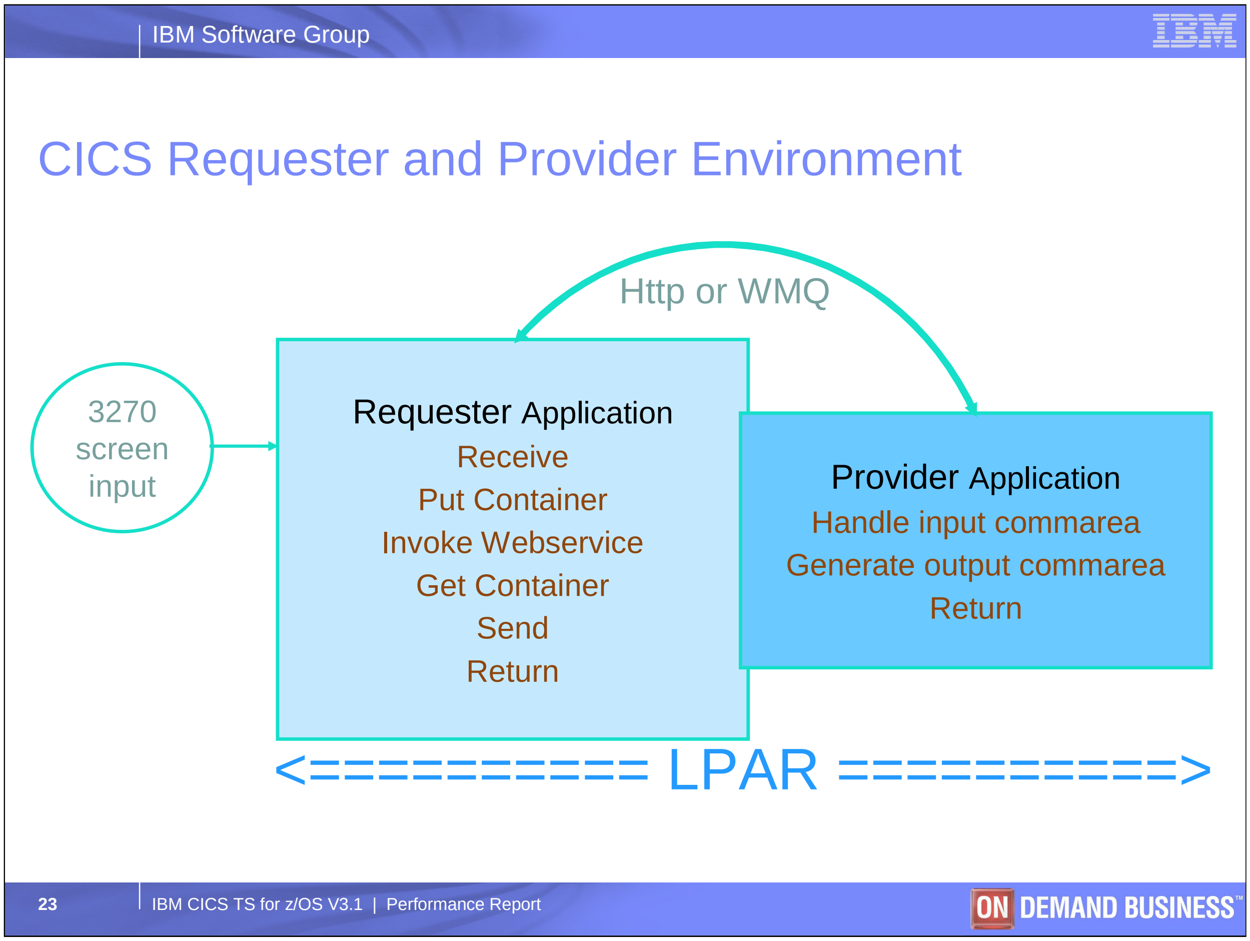
The 'typical' SOAP message again consists of 123 elements in, 123 elements out

This performance evaluation is used to measure the performance of the CICS SOAP Pipeline where Requester and Provider applications run in seperate CICS regions

The CICS Web Services Assistant is used by both the Requester and Provider applications in CTS 3.1

Performance evaluations were run using CWSA HTTP, CWSA WMQ (WebSphere MQ) and also using the SOAP for CICS Feature with HTTP on a CICS 2.3 system with user-written application handlers

Both CICS regions ran on the same LPAR



IBM Software Group

CICS Requester and Provider Environment

3270 screen input

Http or WMQ

Requester Application

Receive

Put Container

Invoke Webservice

Get Container

Send

Return

Provider Application

Handle input commarea

Generate output commarea

Return

<=========== LPAR ===========>

23 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

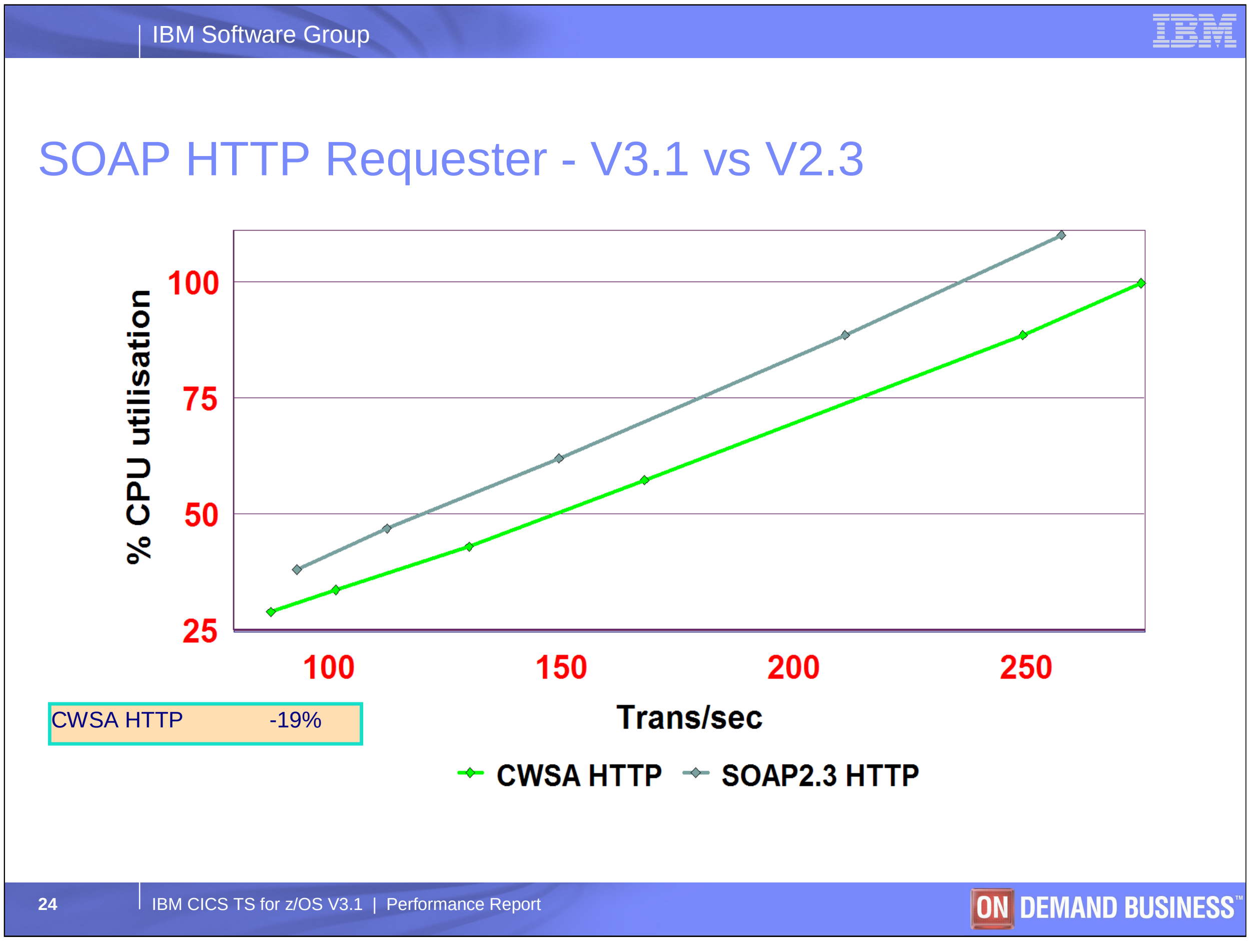
The application in the CICS Requester region is initiated from a 3270 screen.

For CTS 3.1, the data structure to be sent by the requester application is placed in a container and sent to the provider application using an INVOKE WEBSERVICE command. The data structure response is then retrieved from the same container by the requester

For CTS 2.3, the data structure to be sent by the requester application is inserted into a SOAP Body template using the DOCUMENT API, which is then put in a BTS container and sent to the provider application using a LINK ACQPROCESS command. The SOAP Body response is then retrieved from a BTS container by the requester, and the data structure extracted using an XML parse

For both CTS 3.1 and CTS 2.3, the CICS Provider application is a commarea-based application that receives the incoming data structure in a commarea, and returns the outgoing data structure in the same commarea

The 3270 requester screen input was simulated using TPNS



IBM Software Group

SOAP HTTP Requester - V3.1 vs V2.3

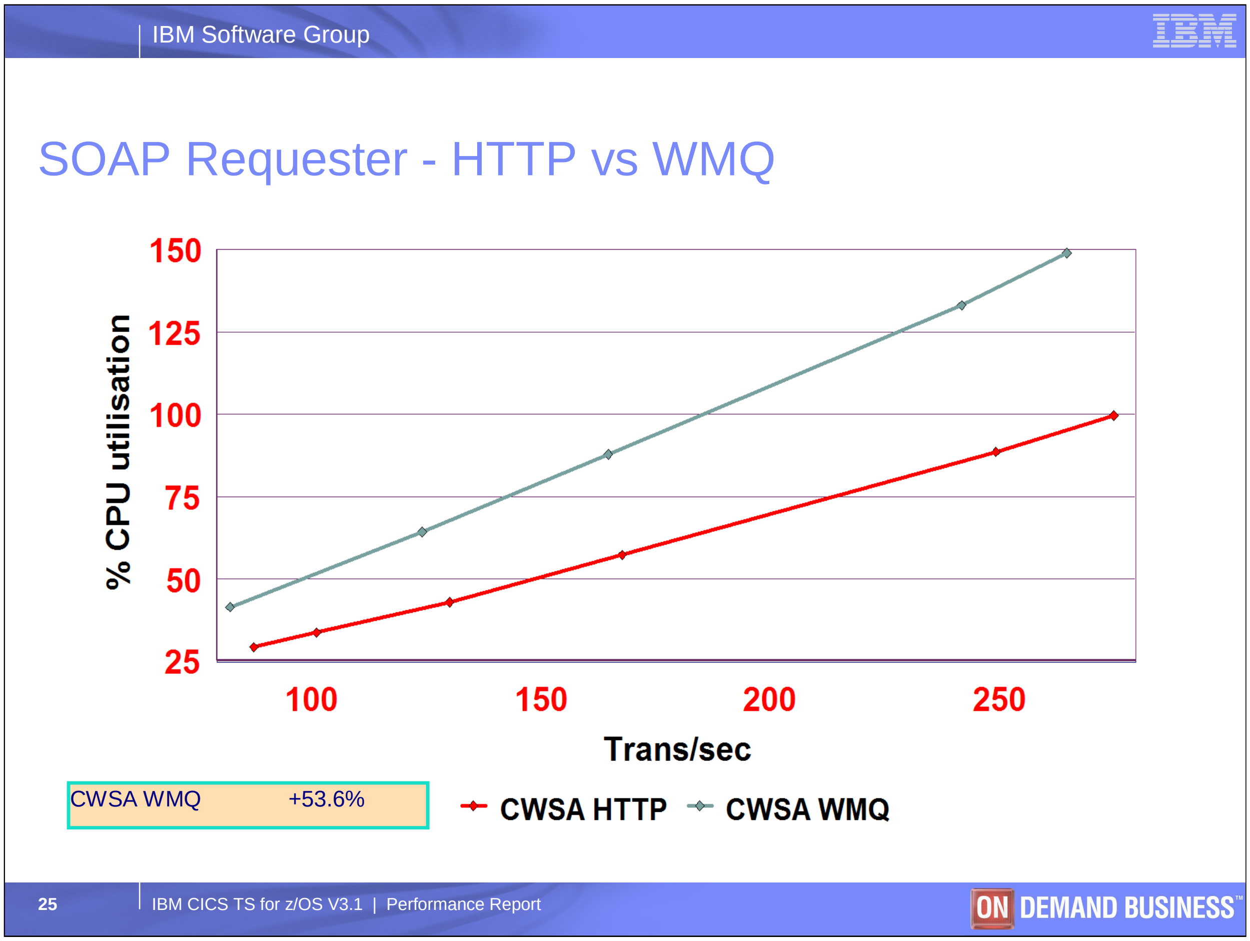
CWSA HTTP -19%

ON DEMAND BUSINESS™

Compares the use of CWSA HTTP and SOAP V2.3 HTTP in the requester region

As there is only a single SOAP message per task, the connection is broken each time - i.e. no keep-alive

Note that when using CWSA with HTTP from a requester region, an HTTP OPTIONS * method is sent by CICS. This is used to request information about the communications options available from the server in general



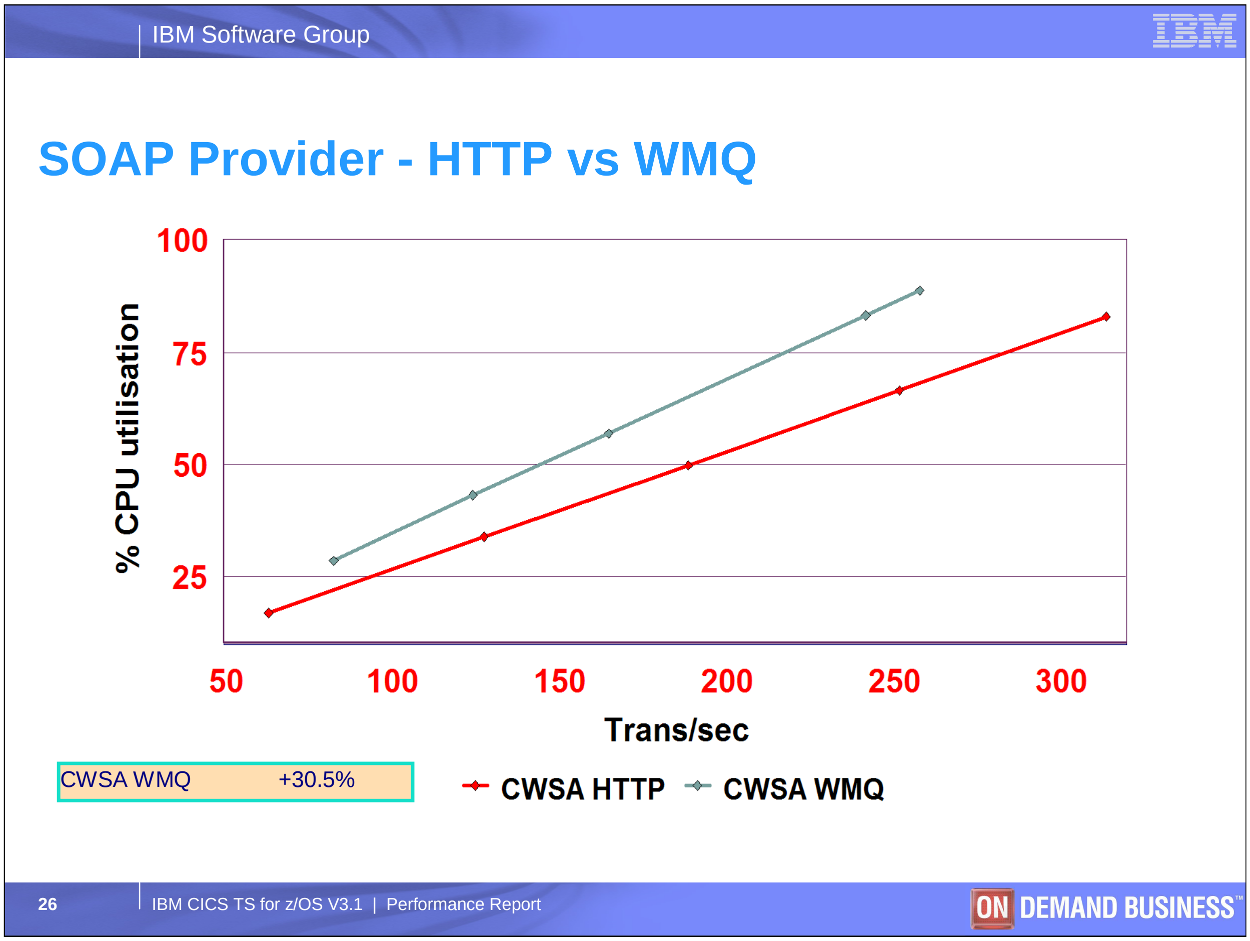
IBM Software Group

SOAP Requester - HTTP vs WMQ

CWSA WMQ	+53.6%

ON DEMAND BUSINESS™

Compares the use of HTTP and WMQ transport layers in the requester region



IBM Software Group

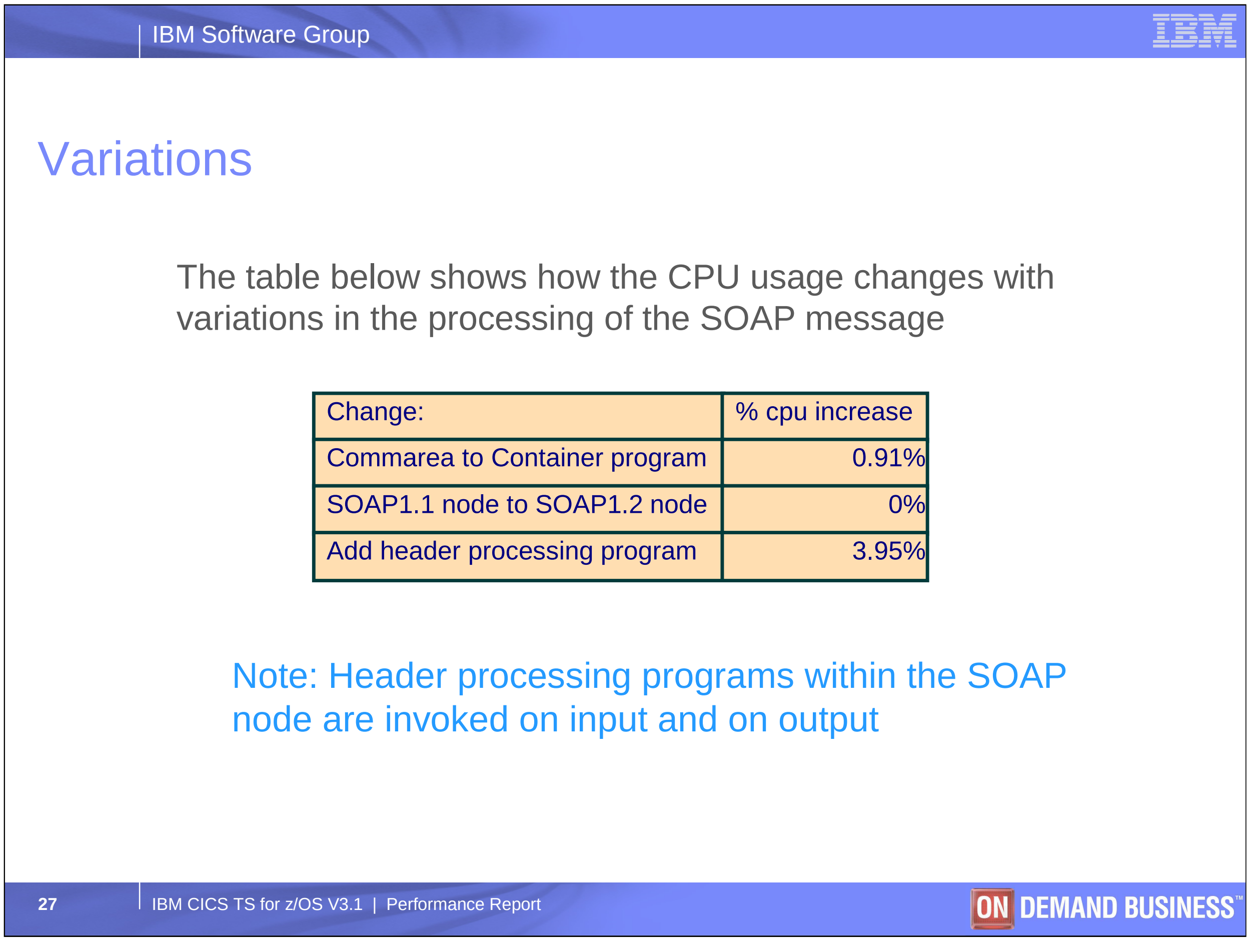
SOAP Provider - HTTP vs WMQ

CWSA WMQ	+30.5%

26 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

Compares the use of HTTP and WMQ transport layers in the provider region



Variations

The table below shows how the CPU usage changes with variations in the processing of the SOAP message

Change:	% cpu increase
Commarea to Container program	0.91%
SOAP1.1 node to SOAP1.2 node	0%
Add header processing program	3.95%

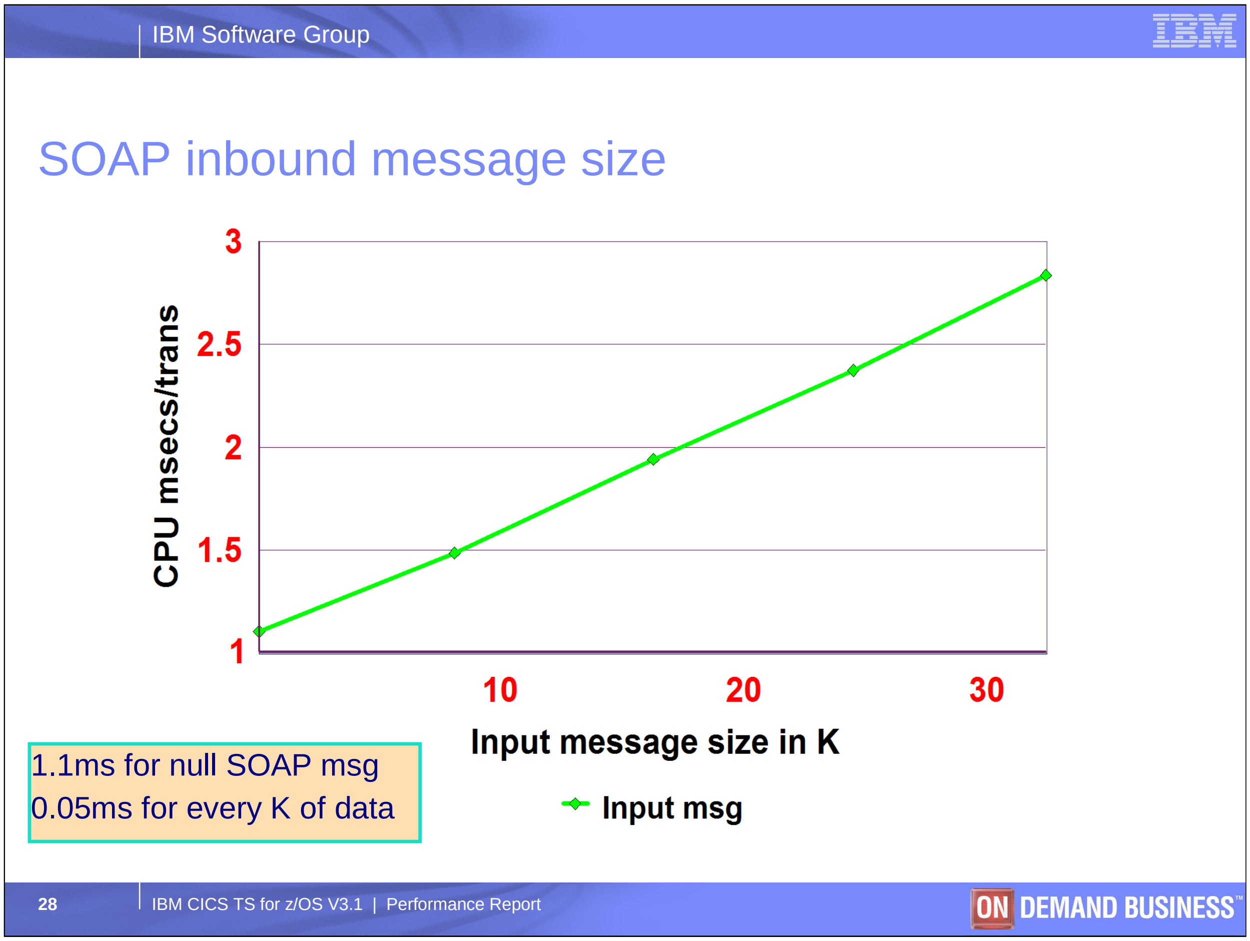
Note: Header processing programs within the SOAP node are invoked on input and on output

Shows variations on the standard SOAP message used in previous evaluations

1. Application provider changed to be a container-based program

2. SOAP 1.2 node in provider pipeline

3. Single dummy SOAP header processing program invoked in input and output pipeline, when a SOAP header is present between SOAP envelope and body. Note it is possible to configure the SOAP pipeline so that a header processing program is invoked regardless of whether the header is present in the input SOAP message

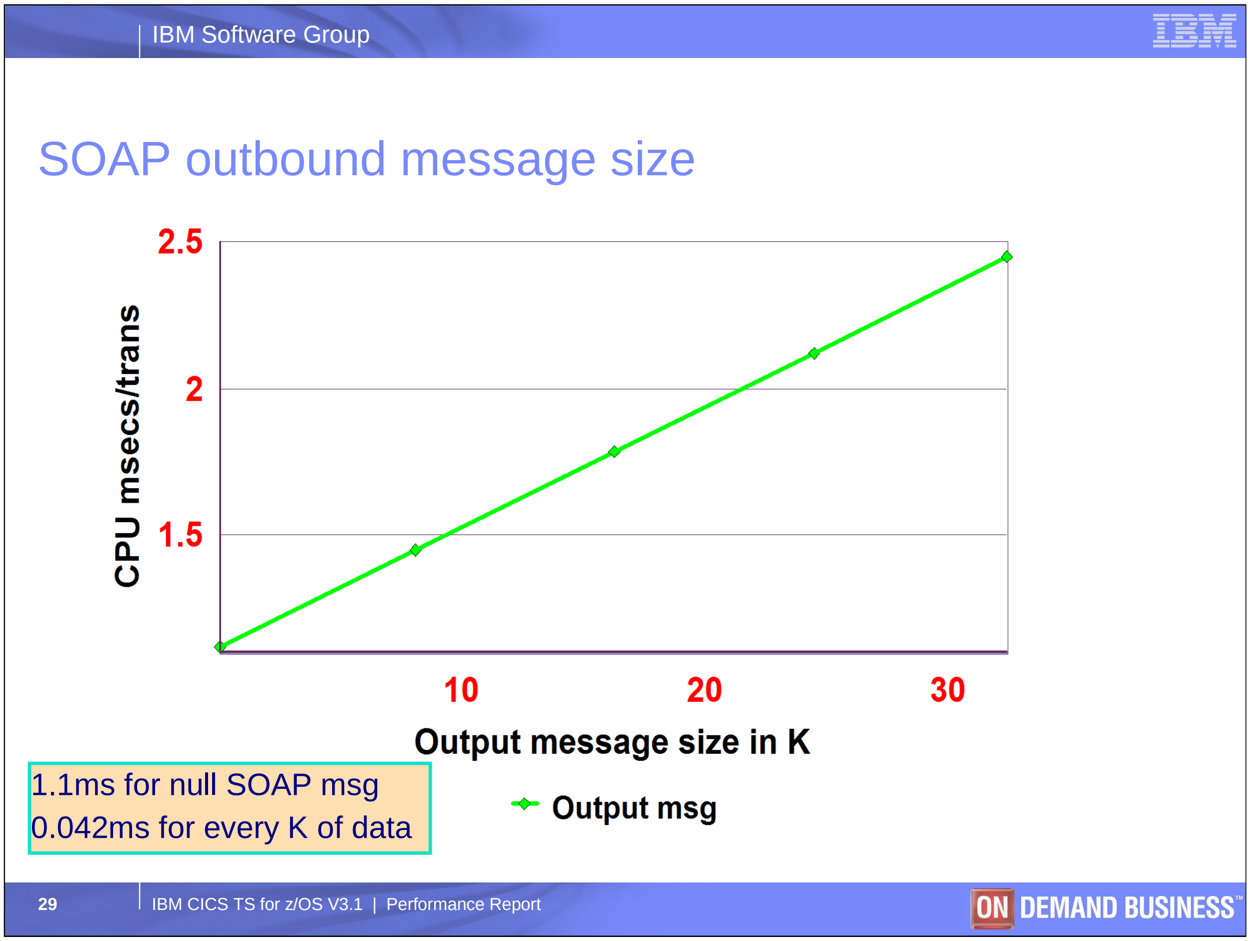


SOAP inbound message size

1.1ms for null SOAP msg
0.05ms for every K of data

Shows how the cpu per transaction is affected by an increase in the input SOAP message size.

The SOAP message consists of a single tag and user data

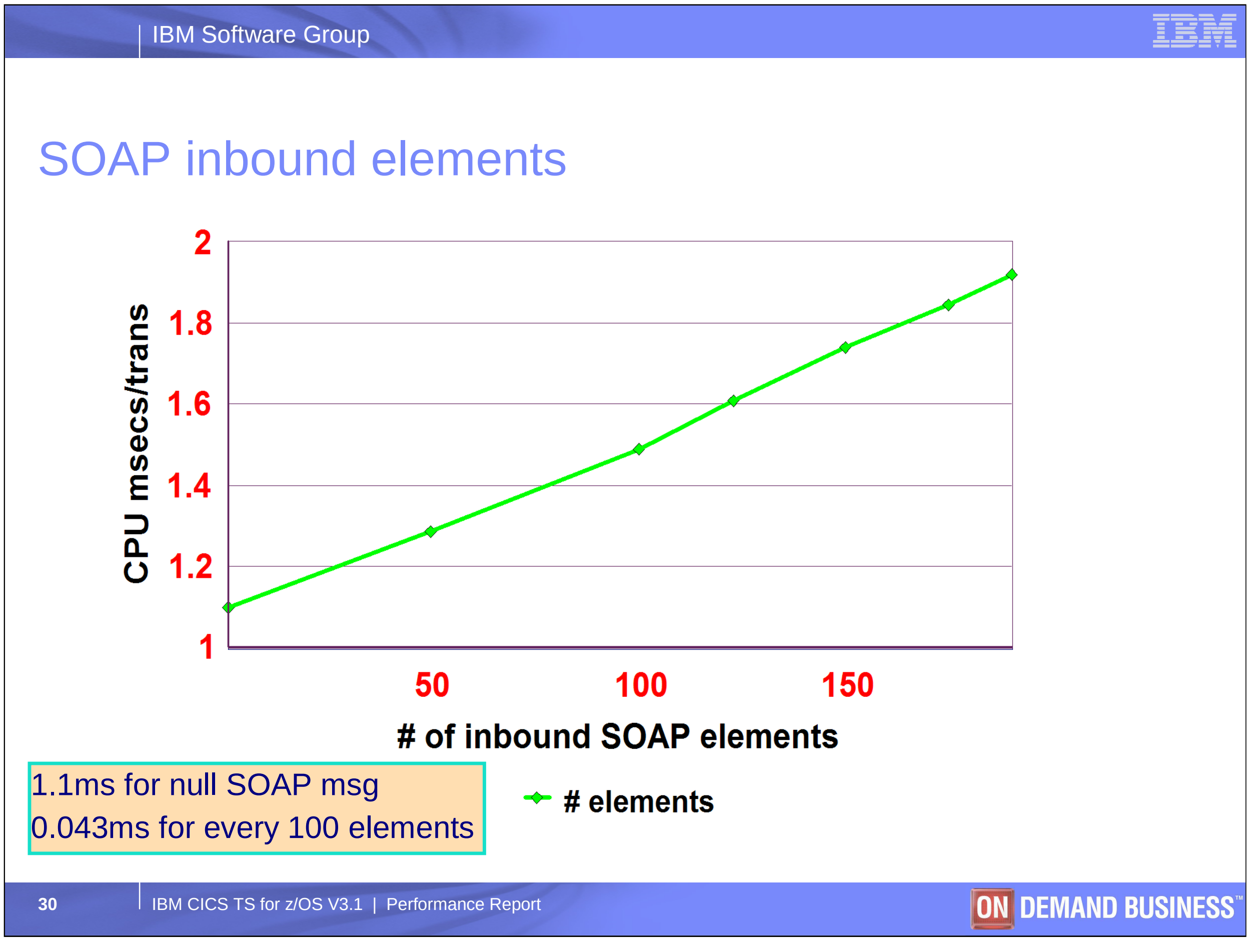


SOAP outbound message size

1.1ms for null SOAP msg
0.042ms for every K of data

Shows how the cpu per transaction is affected by an increase in the output SOAP message size

The SOAP message consists of a single tag and user data



IBM Software Group

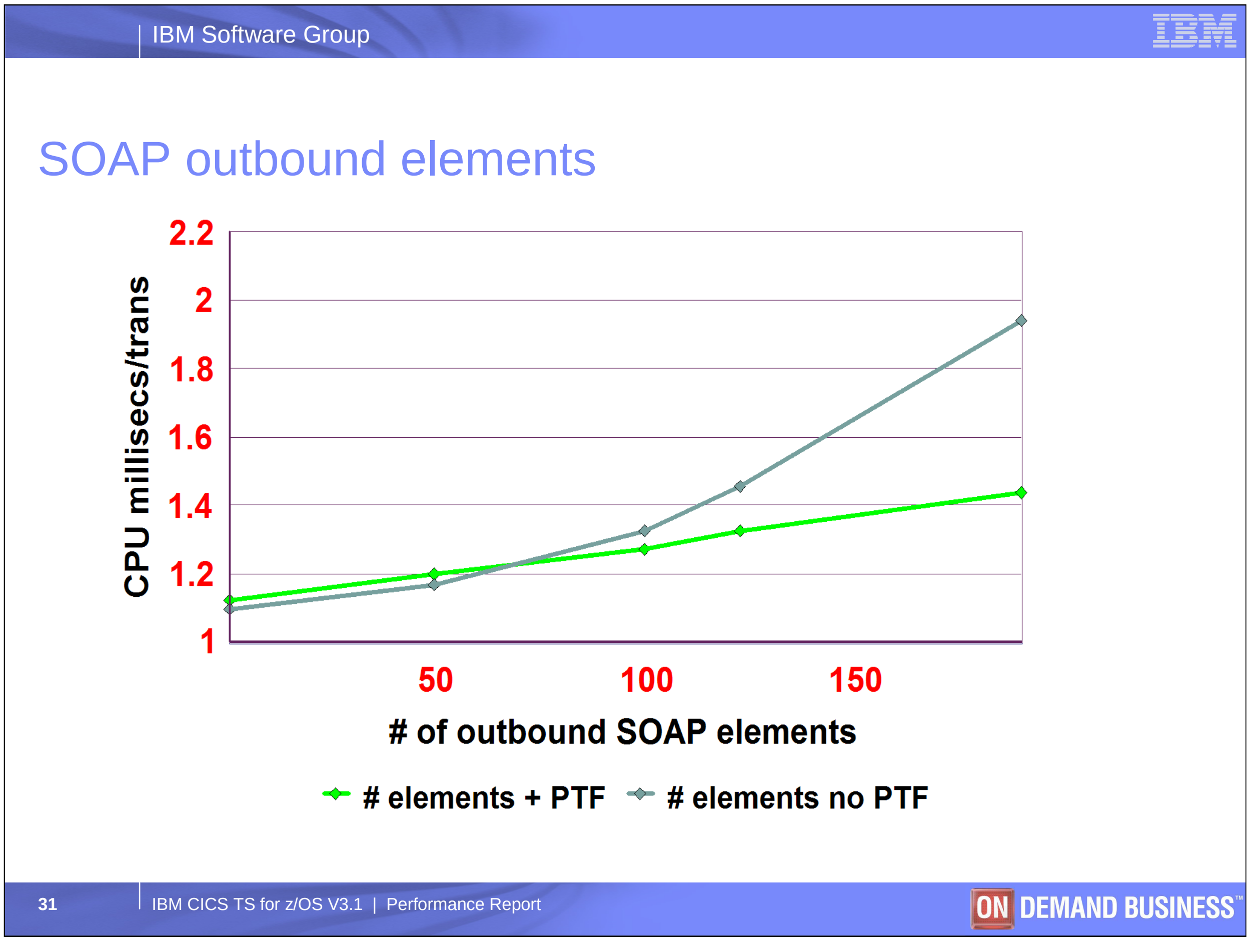
SOAP inbound elements

1.1ms for null SOAP msg
0.043ms for every 100 elements

Shows how the cpu per transaction is affected by an increase in the number of input SOAP elements

The data size of each element is 1 byte

Tag names vary from 5 to 8 bytes



IBM Software Group

SOAP outbound elements

31 IBM CICS TS for z/OS V3.1 | Performance Report

ON DEMAND BUSINESS™

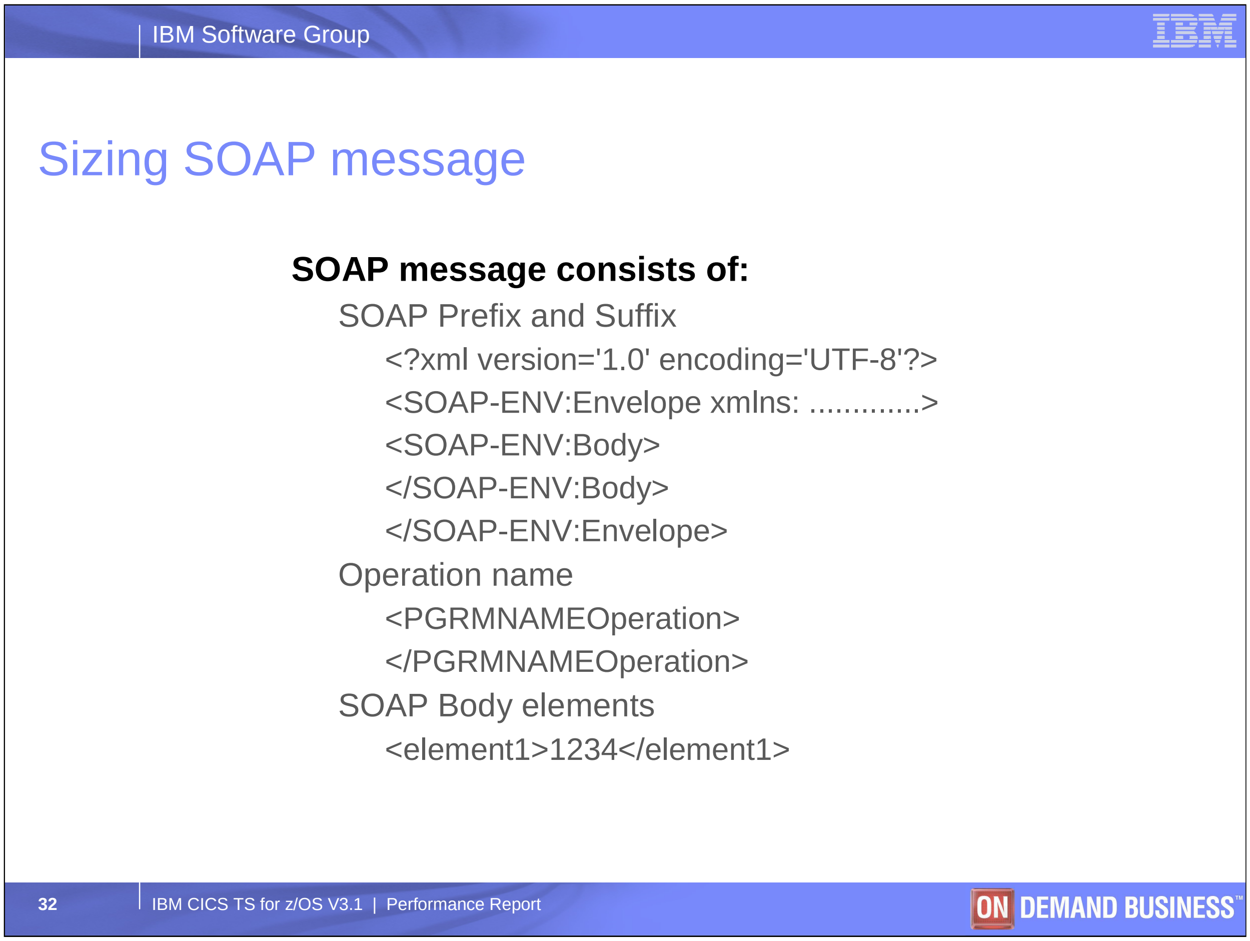
Shows how the cpu per transaction is affected by the number of output SOAP elements. A performance PTFproduces a linear increase in cpu per transaction, as the number of outbound SOAP elements increase.

The data size for each element is 1 byte

Tag names vary from 5 to 8 bytes

CPU increase is not linear for an increase in the number of output elements

Looking to change the method used for processing output elements



Sizing SOAP message

SOAP message consists of:

- SOAP Prefix and Suffix
  - <?xml version='1.0' encoding='UTF-8'?>
  - <SOAP-ENV:Envelope xmlns: ....>
  - <SOAP-ENV:Body>
  - </SOAP-ENV:Body>
  - </SOAP-ENV:Envelope>
- Operation name
  - <PGRMNAMEOperation>
  - </PGRMNAMEOperation>
- SOAP Body elements
  - <element1>1234</element1>

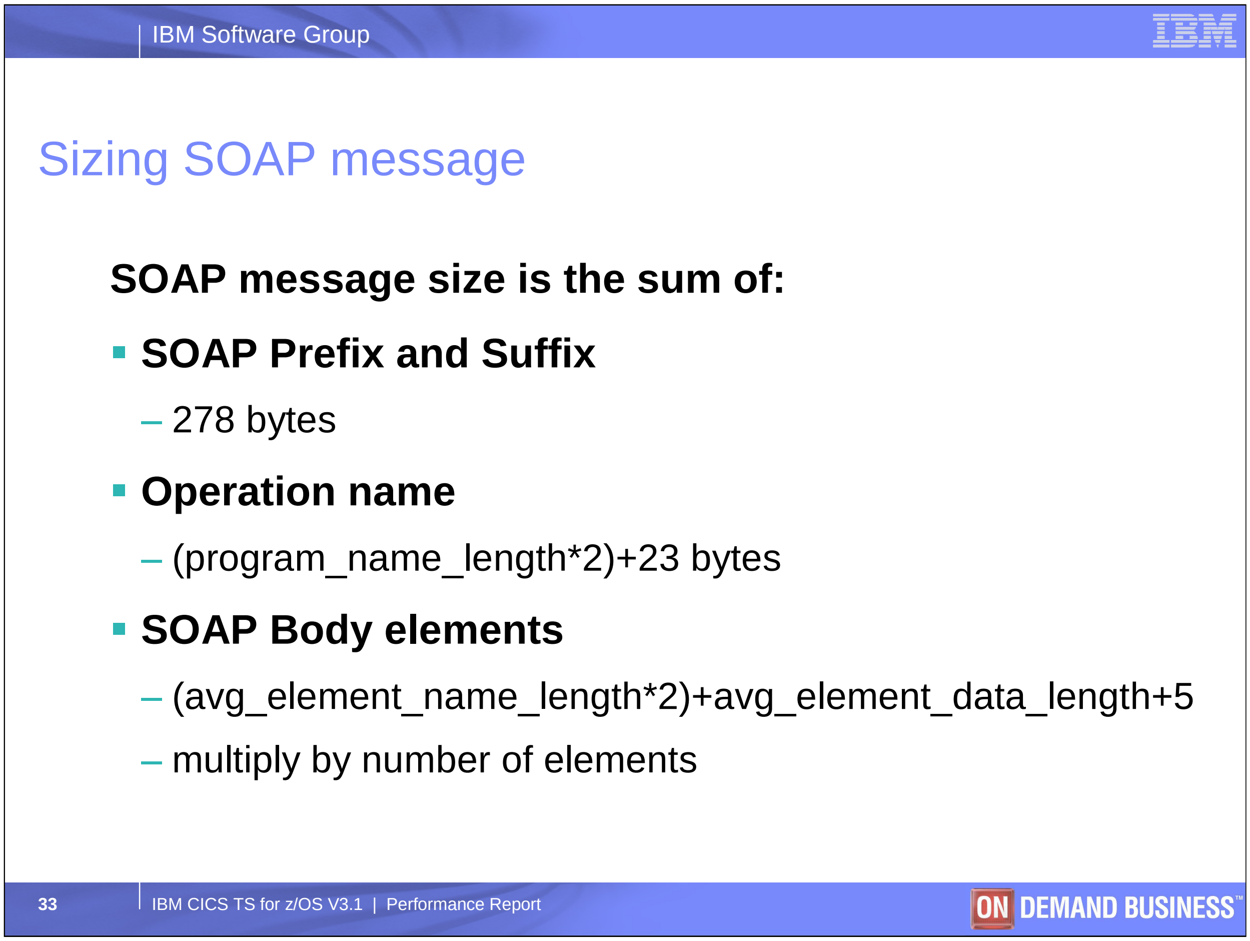
SOAP message consists of:

XML version and encoding scheme

SOAP envelope consisting of namespace URIs

SOAP body which includes operation name (method name in Java), and data elements enclosed in XML tags

'SOAP Suffix' are ending tags for SOAP body and envelope



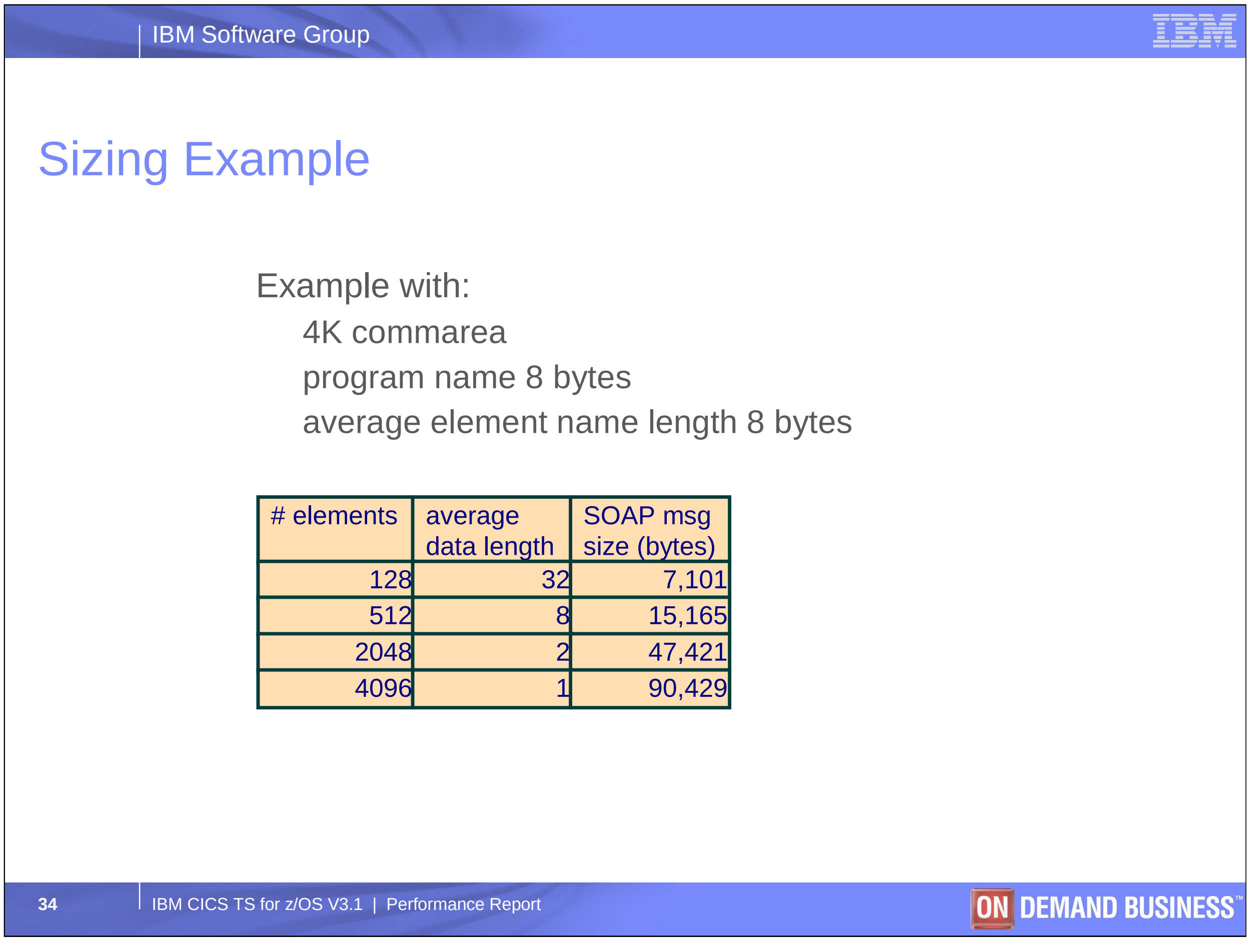
Sizing SOAP message

SOAP message size is the sum of:

- **SOAP Prefix and Suffix**
  - 278 bytes
- **Operation name**
  - (program_name_length*2)+23 bytes
- **SOAP Body elements**
  - (avg_element_name_length*2)+avg_element_data_length+5
  - multiply by number of elements

278 bytes includes namespace URIs which may vary

Operation name is generated by taking the program name and adding 'Operation' to the end of the name



Sizing Example

Example with:

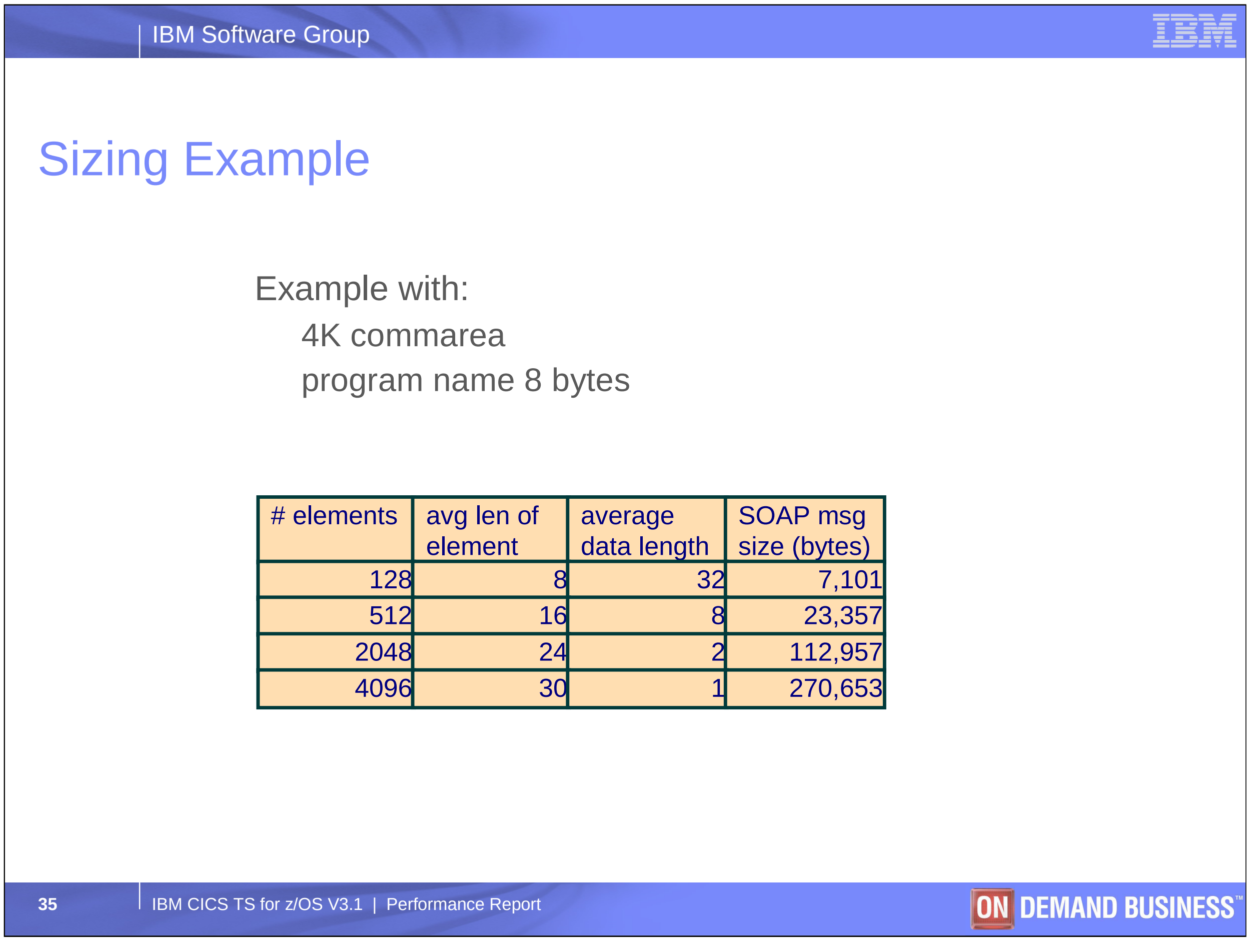
- 4K commarea
- program name 8 bytes
- average element name length 8 bytes

# elements	average data length	SOAP msg size (bytes)
128	32	7,101
512	8	15,165
2048	2	47,421
4096	1	90,429

Assumes the CICS Web Services Assistant has been used to generate the WSDL and wsbind file

Shows how the size of the SOAP message generated for a 4K commarea can vary depending on the number of elements of data within the 4K commarea

The more data elements the bigger the SOAP message



Sizing Example

Example with:

- 4K commarea
- program name 8 bytes

# elements	avg len of element	average data length	SOAP msg size (bytes)
128	8	32	7,101
512	16	8	23,357
2048	24	2	112,957
4096	30	1	270,653

Assumes the CICS Web Services Assistant has been used to generate the WSDL and wsbind file

Shows how the size of the SOAP message generated for a 4K commarea can vary depending not just on the number of elements of data, but also the data element name length defined in the data structure

Data element name lengths can have a major effect on the size of the SOAP message

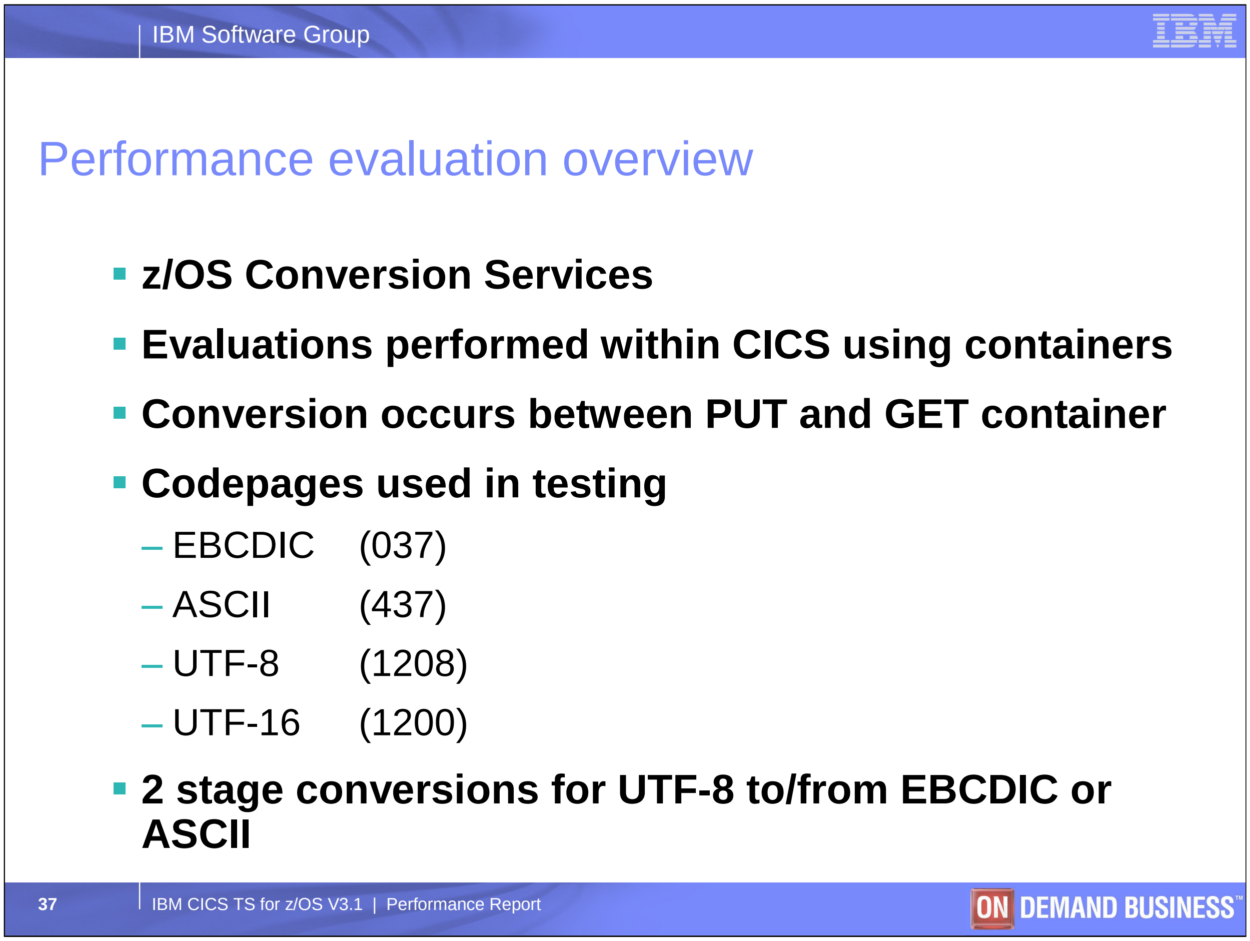
IBM Software Group

Codepage conversion

Codepage conversion

ON DEMAND BUSINESS™

This performance evaluation covers the use of z/OS Conversion Services within CICS to handle codepage conversions



Performance evaluation overview

- **z/OS Conversion Services**
- **Evaluations performed within CICS using containers**
- **Conversion occurs between PUT and GET container**
- **Codepages used in testing**

– EBCDIC	(037)
– ASCII	(437)
– UTF-8	(1208)
– UTF-16	(1200)

- **2 stage conversions for UTF-8 to/from EBCDIC or ASCII**

ON DEMAND BUSINESS™

The evaluations consisted of running a cobol program under CICS that issued:

PUT CONTAINER FROMCCSID

GET CONTAINER INTOCCSID

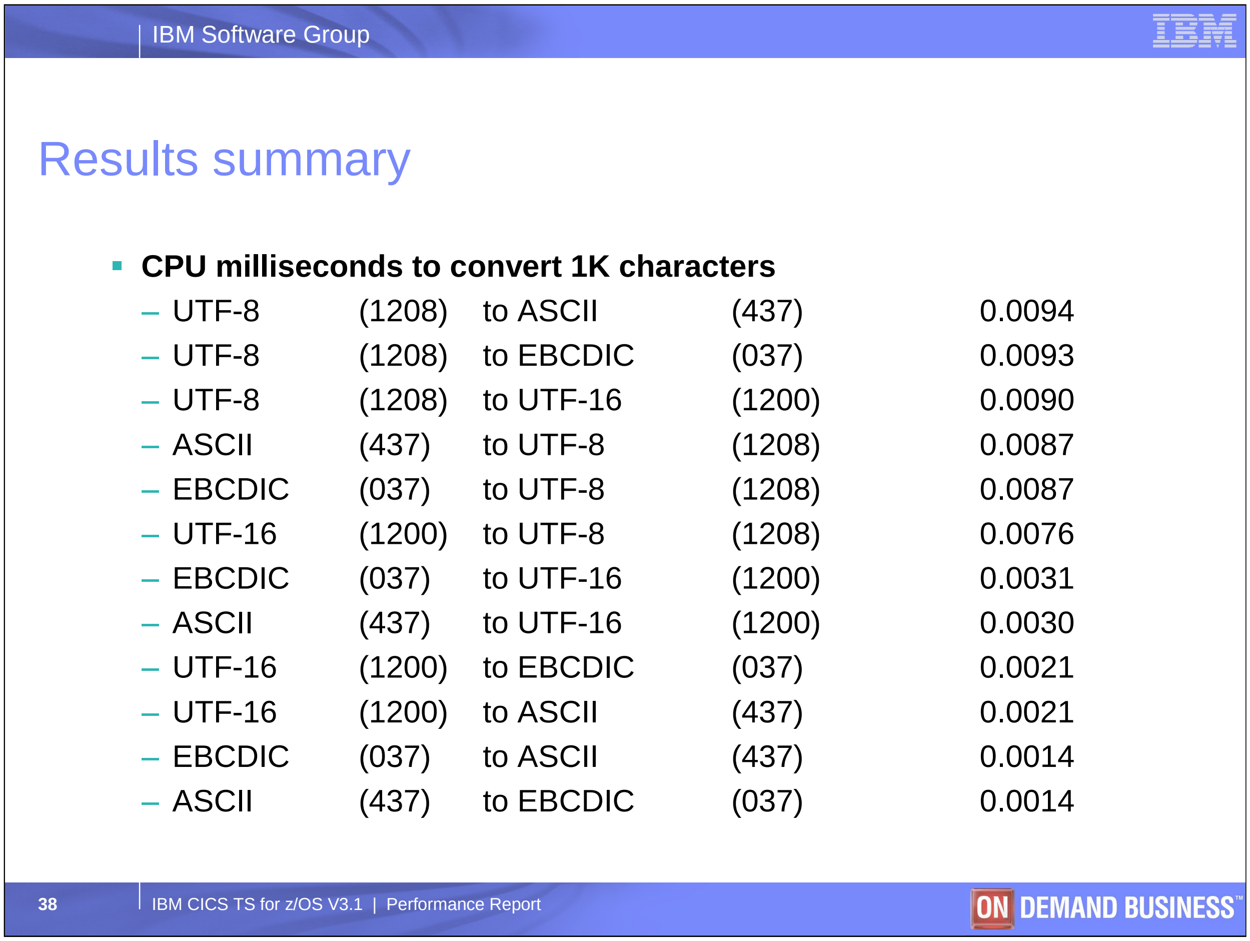
This process caused the z/OS conversion services to be run to convert the data from and to the required CCSIDs

The number of bytes converted was varied from 1K to 1M in the performance tests

With UTF-8, between 1 and 4 bytes are used to represent a character

With UTF-16, 2 bytes are used to represent a character

Note that 2 stage conversions convert to UTF-16 as the first stage in the process



Results summary

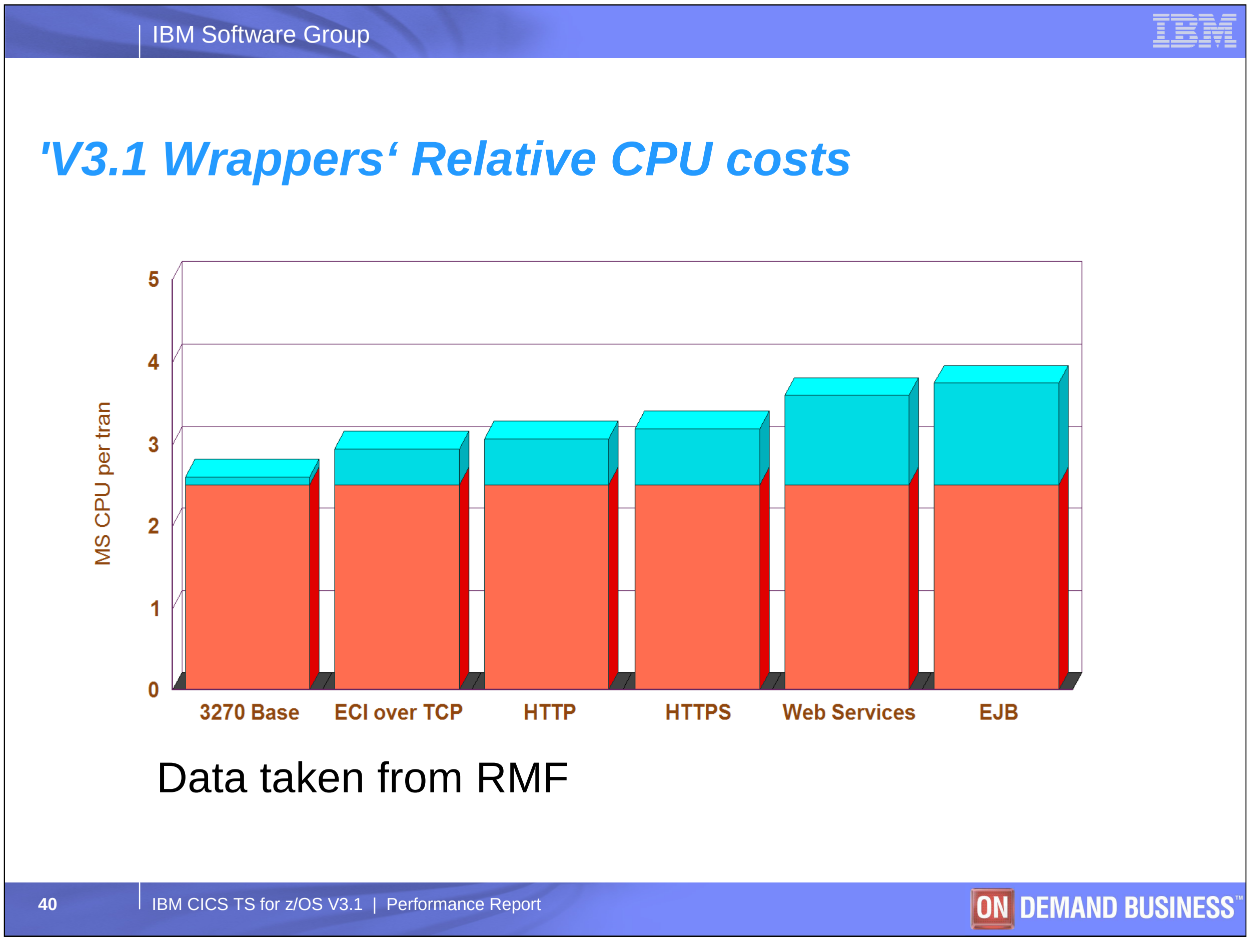
- **CPU milliseconds to convert 1K characters**

– UTF-8	(1208)	to ASCII	(437)	0.0094
– UTF-8	(1208)	to EBCDIC	(037)	0.0093
– UTF-8	(1208)	to UTF-16	(1200)	0.0090
– ASCII	(437)	to UTF-8	(1208)	0.0087
– EBCDIC	(037)	to UTF-8	(1208)	0.0087
– UTF-16	(1200)	to UTF-8	(1208)	0.0076
– EBCDIC	(037)	to UTF-16	(1200)	0.0031
– ASCII	(437)	to UTF-16	(1200)	0.0030
– UTF-16	(1200)	to EBCDIC	(037)	0.0021
– UTF-16	(1200)	to ASCII	(437)	0.0021
– EBCDIC	(037)	to ASCII	(437)	0.0014
– ASCII	(437)	to EBCDIC	(037)	0.0014

Translations involving UTF-8 consumed the most cpu per 1K of characters

Wrappers

'WRAPPERS'



'V3.1 Wrappers' Relative CPU costs

Data taken from RMF

This foil shows the relative costs of 'front ending' or 'wrapping' the same 2.5 CPU milli sec application with various protocols.

The measurements were taken on a steady state system with persistent connections. Approximately 1K of data flowed in each case.

The EJB case used the Continuous JVM

The red represents the application cost, the blue, the infrastructure costs.

These were all simple environments with no Security etc turned on.

IBM Software Group

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

CICS IBM RMF WebSphere z/OS

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. 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 in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.