

# Modernize and Improve Performance of COBOL Applications on z17

Roland Koo

Program Director, Product Management and Strategy, Compilation Technologies, and Enterprise Products

[rkoo@ca.ibm.com](mailto:rkoo@ca.ibm.com)

# Agenda

Accelerating Application Modernization

COBOL 6.5 and Automatic Binary Optimizer (ABO) 2.3 on z17

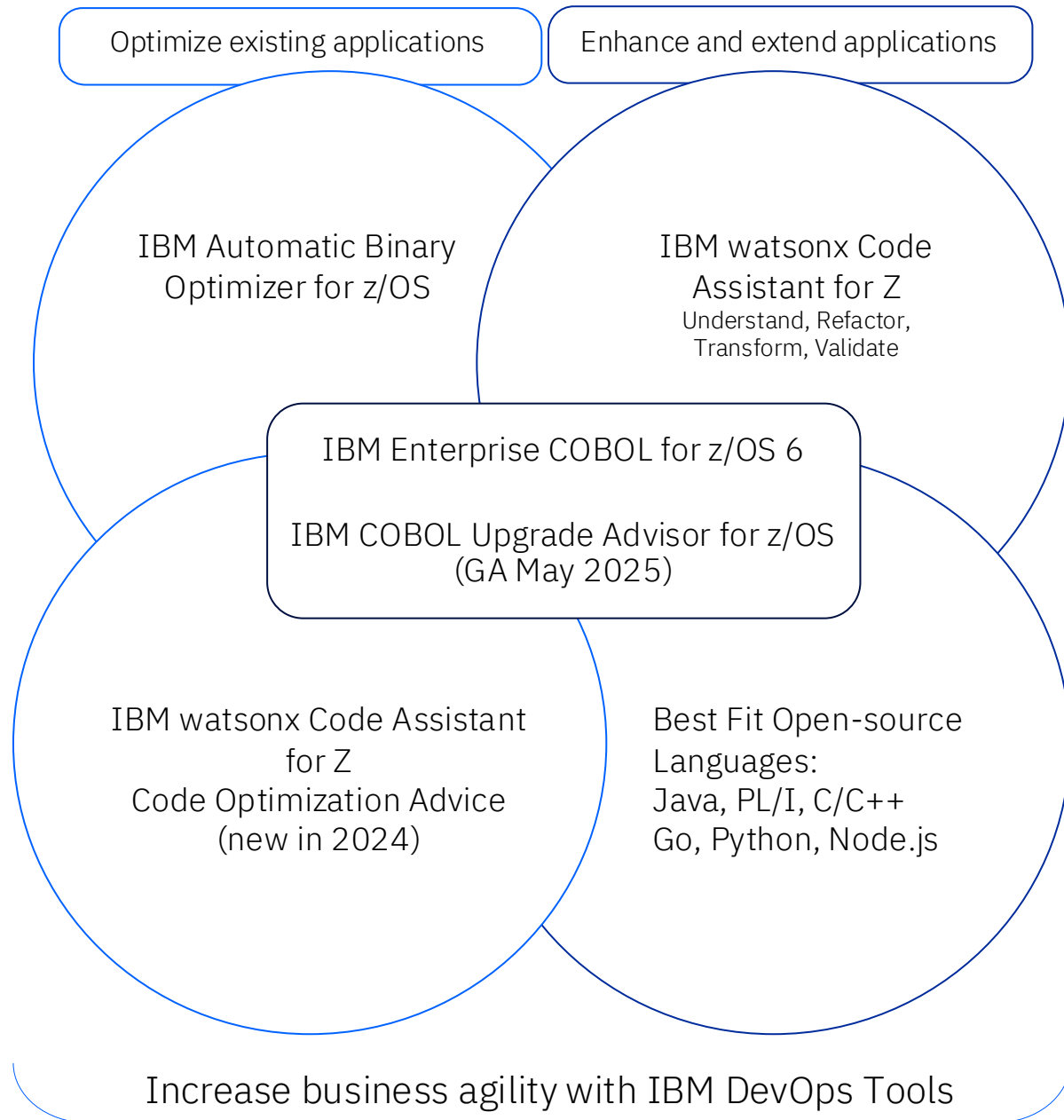
Compiler FixFinder Agent

Watson Code assistant for Z – Optimize

COBOL Upgrade Advisor

# Modernize COBOL Applications on z/OS

A unified approach to optimize, modernize, and extend COBOL workloads



# Application performance delivered by new IBM Z hardware features and compiler support



**IBM zEC12/zBC12**  
ARCH=10

Improve performance of zone decimal arithmetic

Enterprise COBOL 5.1  
ABO 1.1

**IBM z13/z13s**  
ARCH=11

Improve performance of packed decimal arithmetic and string operations

Enterprise COBOL 5.2  
Enterprise COBOL 6.1  
ABO 1.2

**IBM z14/z14 ZR1**  
ARCH=12

Key COBOL data types now first order hardware types

Enterprise COBOL 6.2  
ABO 1.3

**IBM z15/z15 T02**  
ARCH=13

Multi-language application acceleration  
Faster conversions & computations

Enterprise COBOL 6.3  
ABO 2.1

**IBM z16**  
ARCH=14

Exponentiation and float/packed performance  
Numeric-editing acceleration

Enterprise COBOL 6.4  
ABO 2.2

**IBM z17**  
ARCH=15

Hardware accelerated data validations for faster COBOL upgrades  
Faster binary computations used for inter-lang apps

Enterprise COBOL 6.5  
ABO 2.3

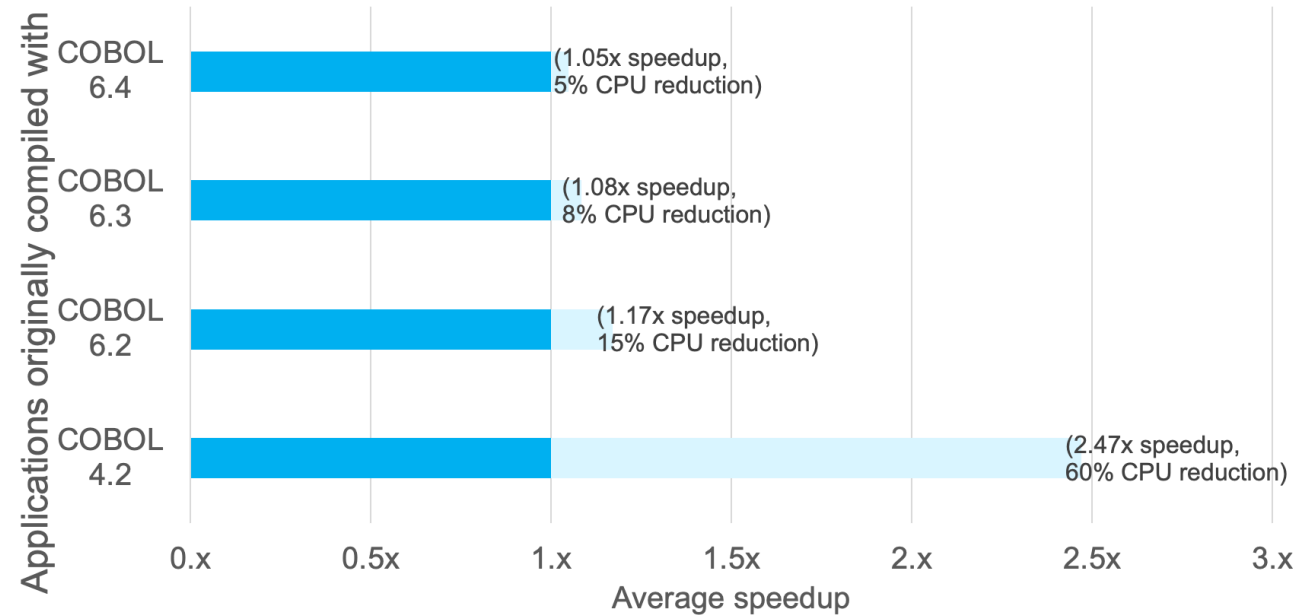
# COBOL 6.5 utilizes IBM z17 better than earlier COBOL compilers

Use new ARCH(15) compiler option to take advantage of the new z17 vector enhancement facility 3

Increases performance for COBOL programs

- Contains BINARY, COMP, COMP-5 or COMP-5 data items when the TRUNC(BIN) compiler option is set
- With NUMCHECK compiler option enabled in production

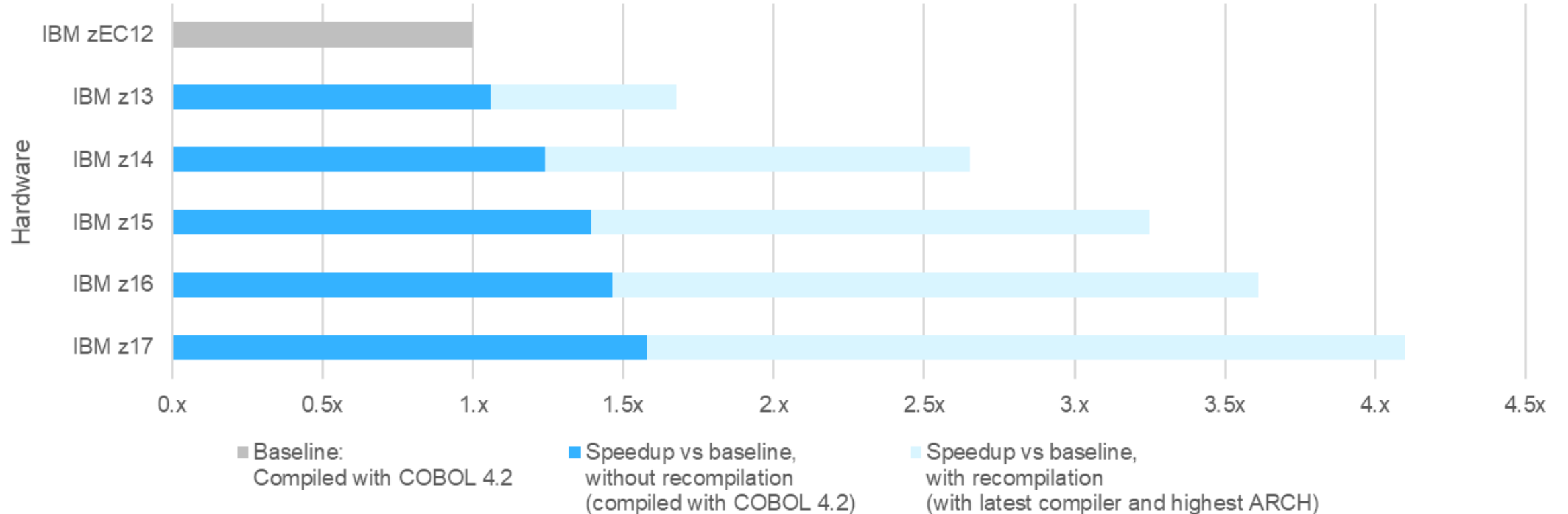
Performance of COBOL 4.2 through COBOL 6.4 applications running on IBM z17, after recompilation with COBOL 6.5 and highest ARCH



Performance results for customer applications will vary, depending on the source code, the compiler options specified, and other factors. Find the full disclaimer [here](#).

# Enterprise COBOL Performance for compute intensive applications

Performance of COBOL 4.2 applications running on IBM zEC12,  
with and without recompilation (with latest compiler and highest ARCH),  
as hardware is upgraded



# COBOL Enhancements (v5 – v6.4)

<p><b>COBOL 5.1 – 5.2</b></p> <p><b>New advanced optimization framework</b> (z10 – z13)</p> <p><b>New COBOL runtime</b></p> <p><b>New object and debugging format</b></p> <p>Co-processor for SQLIMS</p> <p><b>XML processing</b> and UTF 8 enhancements</p> <p>Language features/enhancements: Table Sort, Exit, COPY, REPLACE, VOLATILE clause, Floating comment (*&gt;)</p>	<p><b>COBOL 6.1</b></p> <p>GA: 2016</p> <p><b>Improved scalability with 64-bit optimization framework</b></p> <p>Performance improvements (z13)</p> <p>New language feature: ALLOCATE, FREE, INITIALIZE. <b>JSON GENERATE</b></p> <p>Improved reporting on invalid data and out of bounds for smoother upgrade from COBOL 4</p>	<p><b>COBOL 6.2</b></p> <p>GA: 2017</p> <p>Performance improvements (z14)</p> <p>New language features: <b>Conditional compilation support, JSON PARSE</b></p> <p>New TEST sub-options: SEPARATE/NOSEPARATE</p> <p>NOSTROPT option to suppress optimization of DATA DIVISION regardless of optimization levels</p> <p>INLINE control</p>	<p><b>COBOL 6.3</b></p> <p>GA: 2019</p> <p>Performance improvements (z15)</p> <p>New language features: <b>UTF8</b>, Dynamic-length elementary items, FUNCTION keyword in REPOSITORY paragraph</p> <p><b>64-bit (AMODE 64) Support</b></p>	<p><b>COBOL 6.4</b></p> <p>GA: 2022</p> <p>Performance improvements (z16)</p> <p><b>New language features: User Defined Functions</b></p> <p><b>New, simplified COBOL/Java interoperability Framework</b></p> <p><b>AMODE 31/AMODE 64 interoperability</b></p> <p><b>Integration with Automatic Binary Optimizer</b>; programs compiled with COBOL 6.4 can take full advantage of future Z hardware without recompilation</p>
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

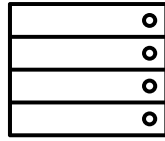
# New features in Enterprise COBOL 6.5



## Optimize performance on IBM z17

Support of the latest IBM z/Architecture to maximize your IBM z16 investment, reduce CPU usage, and reduce operating costs by improving performance of critical COBOL applications

Supports the new vector-packed-decimal-enhancement facility 3 and vector enhancement facility 3 in IBM z17



## NoSQL (VSAMDB) support

### VSAMDB

- Stores unstructured data on z/OS as JSON/BSON documents.
- Scalable and highly available

Support reading, writing, updating, and deleting of JSON documents in a VSAMDB file using COBOL's file processing statements.

Applications written in Python, Java, C, Assembler, and **COBOL** can read, write, update, and delete access and modify shared JSON documents which are stored in VSAMDB



## User-defined types

Support for user-defined types to save you time and minimize source code.

A type is a template that can be used to create multiple instances of a customized user data structure defined using the new TYPEDEF clause.

User-defined types help to guarantee consistency of data definitions across different modules and increase data integrity.

COBOL support for user-defined types is similar to user-defined types in other languages, such as 'C'.



## Integration with Automatic Binary Optimizer

Improved integration with IBM Automatic Binary Optimizer for z/OS to invest in your future so that modules you compile today with COBOL 6.5 take advantage of future IBM Z hardware enhancements, without having to be recompiled

**documents in VSAMDB.**

# VSAMDB: COBOL Program

1) Use new VSAMDB- prefix on DD name.

2) Use new LENGTH phrase on RECORD KEY clause to indicate a satellite item that contains byte length for the key.

3) Use DEPENDING ON phrase to indicate record length in bytes.

4) Define file record with PIC U BYTE-LENGTH or PIC U DYNAMIC.

5) Record key is **NOT** defined in file record. Key can be in W-S, L-S. Use PIC U BYTE-LENGTH or PIC U DYNAMIC.

```
IDENTIFICATION DIVISION.  
PROGRAM-ID.      EXAMPLE1.  
ENVIRONMENT DIVISION.  
INPUT-OUTPUT SECTION.  
FILE-CONTROL.  
    SELECT FILE1 ASSIGN TO VSAMDB-INDD0  
    ORGANIZATION IS INDEXED  
    ACCESS MODE IS RANDOM  
    RECORD KEY IS VSAMDB-PKEY  
                LENGTH IS VSAMDB-PKEY-LEN BYTES.  
DATA DIVISION.  
FILE SECTION.  
FD FILE1      RECORD IS VARYING IN SIZE  
    DEPENDING ON FILE1-REC-LEN.  
01 FILE1-REC-PAYLOAD PIC U BYTE-LENGTH 640.  
WORKING-STORAGE SECTION.  
01 FILE1-REC-LEN PIC 9(9) BINARY.  
01 VSAMDB-PKEY PIC U BYTE-LENGTH 240.  
01 VSAMDB-PKEY-LEN PIC 9(9) BINARY.  
01 USER-DATA.  
    05 FIRST-NAME PIC U(20).  
    05 LAST-NAME PIC U(20).  
    05 ADDRESS.  
        10 STREET PIC U(20).  
...
```

# Automatic Binary Optimizer (ABO)

## What is ABO ?

- IBM's unique, cutting-edge optimization technology for COBOL applications
- Targets VS COBOL II 1.3 to COBOL 6.5
- Benefits all IBM Z customers who run critical COBOL applications on z/OS

## What does ABO do ?

- Optimizes performance of COBOL applications without the need recompiling source
- Fully supports IBM Z hardware features designed to improve performance of COBOL workload
- Maintains strict functional compatibility with the original COBOL binaries

## What is ABO's value ?

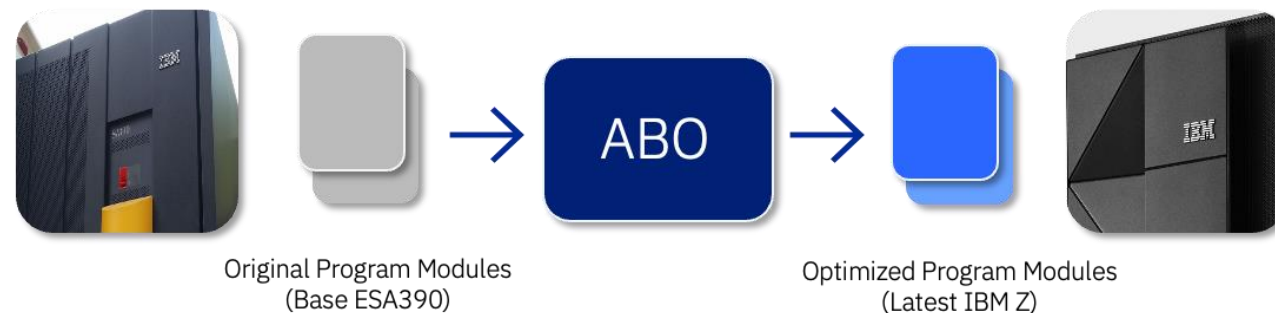
- Reduces CPU usage, processing time, and operating costs, for critical COBOL applications
- Quick time to value by significantly reducing build and testing efforts
- Reduces the scope of migration to COBOL 6

## Automatic Binary Optimizer (ABO) Assistant

- A suite of tools that eliminates the manual and time-consuming tasks for identifying top CPU consuming COBOL programs for optimization by ABO
- Simplifies ABO evaluation and accelerates ABO deployment for batch and online COBOL/CICS applications with enhanced workflow



[Scan QR to access the new ABO Essentials Badge](#)



# ABO on COBOL 6.4+ Modules

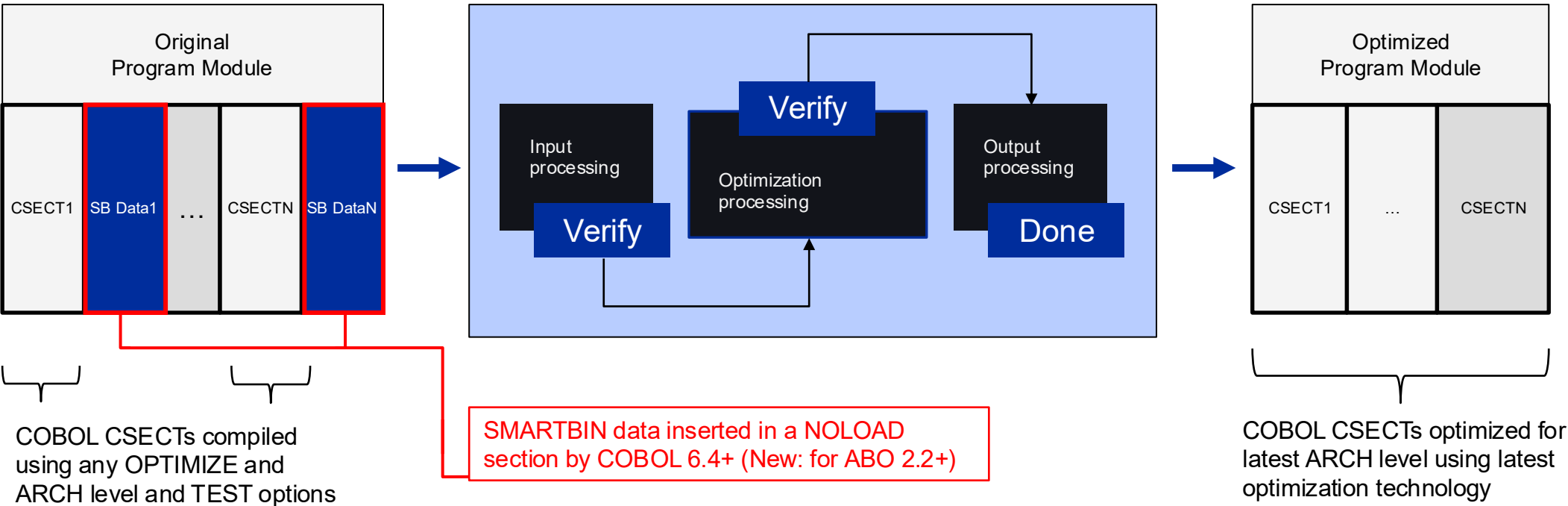
**Option Default** : The inclusion or exclusion of the extra binary metadata is controlled by a **COBOL compiler option**: SMARTBIN/NOSMARTBIN, *default*: SMARTBIN

- **Without SMARTBIN data, the COBOL 6.4+ compiled code will not be eligible for ABO optimization**
- Only downside to inclusion is on disk size increase of module
- The upside is all the benefits of ABO we have talked about

**Size** : The SMARTBIN data is in a NOLOAD section so there will only be an increase of the on-disk size of the compiled program

- no memory increase (data is not loaded) when running
- only loaded/used during later ABO optimization
- we will minimize SMARTBIN data size as much as possible, but it could be up to **2x→3x** the code size depending on the program

ABO Internal Processing for COBOL 6.4+ code  
(same process as for v4 and earlier modules)



# What is Automatic Binary Optimizer (ABO) Assistant

A suite of tools that eliminates the manual and time-consuming tasks for identifying top CPU consuming COBOL programs for optimization by ABO

Simplifies ABO evaluation and accelerates ABO deployment for batch and online COBOL/CICS applications with enhanced workflow

Generates concrete performance evaluation results from both high-level System Management Facility (SMF) data, and lower-level profiling data

Accelerates deployment of ABO optimized binaries



# Complementary Technologies

## Automatic Binary Optimizer for z/OS 2.3

Use to improve performance of binaries that are not regularly updated and not compiled for a long time

Reduce build resources compared to a recompile and gets comparable performance to Enterprise COBOL 6.5 (see graph on right)

Optimize binaries built with VS COBOL II 1.3 to Enterprise COBOL 6.5

## Enterprise COBOL for z/OS 6.5

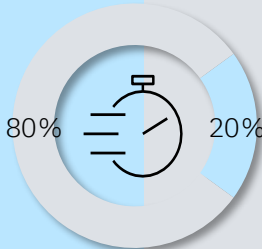
Use as production compiler for application development and maintenance

Use new COBOL language features to modernize critical applications

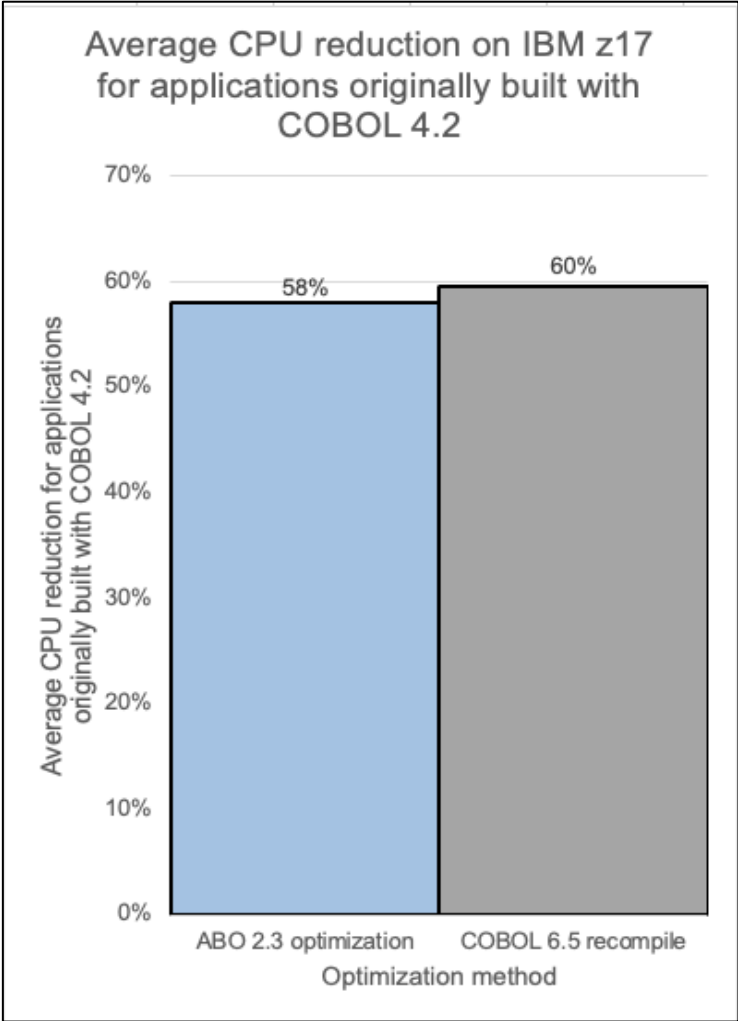
Optimize performance

Generate binary metadata for future optimization by ABO

Code not updated or compiled for a long time



Code under active development or maintenance



# UnipolSai Assicurazioni

Modernizing core  
business  
applications for the  
digital age

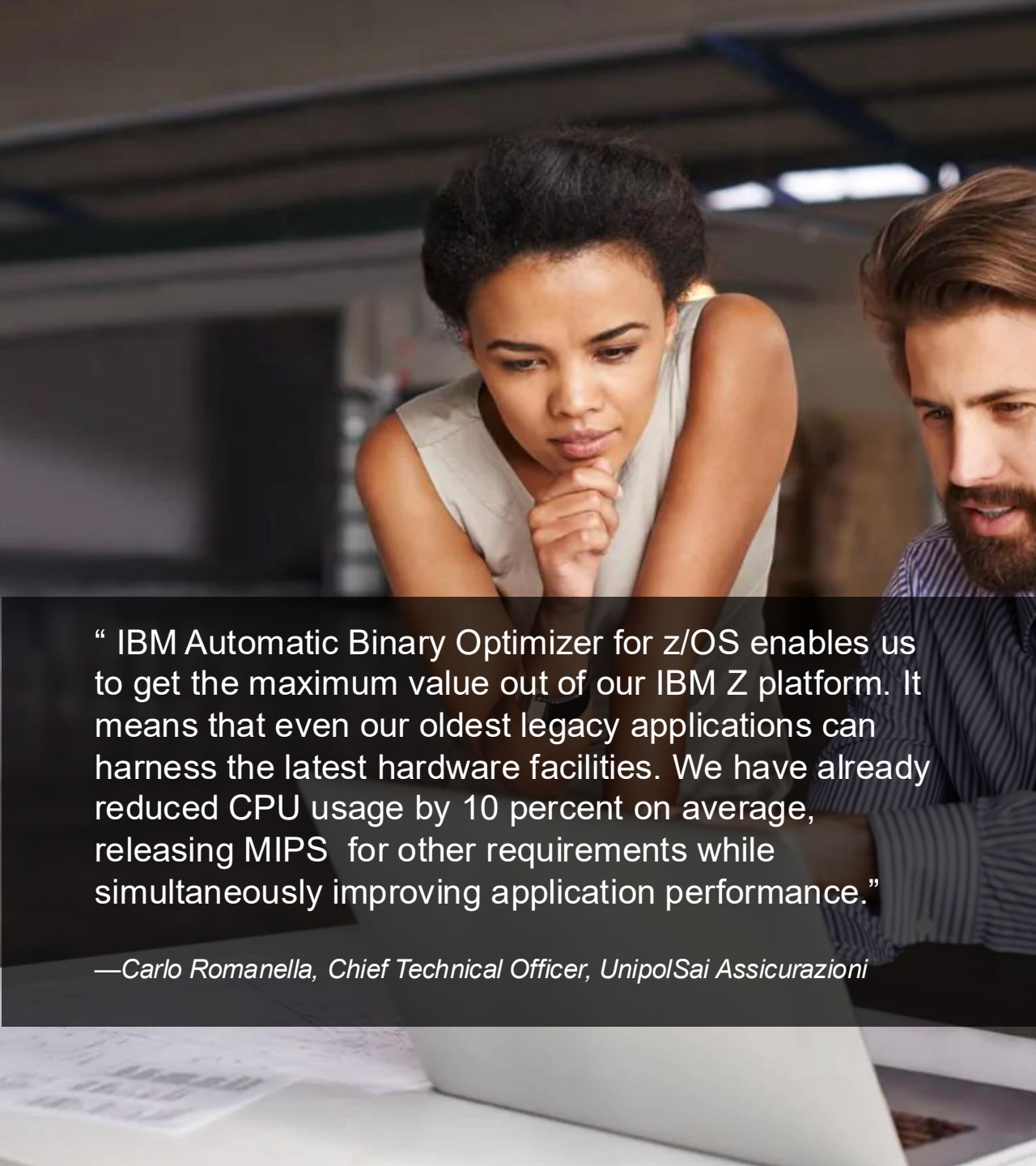
Solution: Systems software

Industry: Insurance

Determined to deliver on growing demand for digital services, Italian insurer UnipolSai Assicurazioni wanted to bring its legacy applications into the 21st century.

Working with IBM, UnipolSai Assicurazioni upgraded its COBOL code, improving performance, increasing efficiency and making it easier for developers to add exciting new features.

Public reference ([Link](#))



“ IBM Automatic Binary Optimizer for z/OS enables us to get the maximum value out of our IBM Z platform. It means that even our oldest legacy applications can harness the latest hardware facilities. We have already reduced CPU usage by 10 percent on average, releasing MIPS for other requirements while simultaneously improving application performance.”

—Carlo Romanella, Chief Technical Officer, UnipolSai Assicurazioni

## IBM Z Compilers Fix Finder Agent \*

Empowers system programmers to use natural language search for fixes for Enterprise COBOL for z/OS

\* Part of the WXA4Z capabilities

### User personas



Zach  
Senior z/OS Systems  
Programmer



Chris  
Junior System  
Programmer

### Challenge

- I have a misbehaving COBOL workloads on my system after a change
- I need to quickly check if there are fixes available that may address the behavior I am seeing

### Benefits

- Understands natural language descriptions of symptoms in combination with actual output
- Does an in-depth search of all available COBOL compiler fixes
- Compares the reported symptoms with the behavior corrected in fixes to recommend good candidates

### Use case

The new COBOL program has resulted in errors

- Use flexible, natural language description to describe the observed behavior
- Include error messages and symptoms to help refine the agent's search

- The agent cross references the error details with available COBOL compiler fixes
- Receive a tailored list of potentially related fixes that may be related to the observed behavior

\* [Entitlement Memo](#) available with download and configuration instruction

# Example

Finding fixes to resolve an SOC4 abend when generating a large XML table

I'm getting an SOC4 abend when generating a large XML table using the XML GENERATE statement in COI

Agents

IBM Z Compiler Fix Fi... x v

Chats

Today

I'm getting an SOC4 abend when genera

You 5:13 PM

I'm getting an SOC4 abend when generating a large XML table using the XML GENERATE statement in COBOL. Is this a known issue with the compiler, and are there any published limits on the size of the table that I should be aware of?

IBM Z Compiler Fix Finder Agent 5:13 PM | Show Reasoning v

The SOC4 abend when generating a large XML table using the XML GENERATE statement in COBOL is a known issue. The problem is caused by a limitation on the number of arrays for a group item in XML GENERATE, which can result in an SOC4 abend when a large XML table is being generated.

The issue is addressed in APAR PI48232, which provides a fix for the Enterprise COBOL V5 compiler. The fix involves changing the XML GENERATE run-time library routine to use a variable-size array to hold information about each OCCURS, rather than a fixed-size array.

To resolve the issue, you can apply the provided service, which includes the V5/V6 Runtime September 2015 PTF. This should allow you to generate large XML tables without encountering the SOC4 abend.

It's worth noting that there are no published limits on the size of the table that can be generated using the XML GENERATE statement, but the fix provided in APAR PI48232 should allow for larger tables to be generated without encountering the SOC4 abend.

👍 🗑️ 📄

watsonx Orchestrator

New chat

Agents

IBM Z Compiler Fix Fi... x v

Chats

Today

New chat

5:12 PM

Hello, welcome to watsonx Orchestrator

Accuracy of generated content may vary. Please double-check responses.

What can you do for me? Learn about watsonx Orchestrator AI Agent and Gen AI capabilities.

Formalize Message Craft more professional messages.

Summarize Meeting Notes Summarize key points from meeting notes.

do new agent

new agent

Type something...

# IBM watsonx Code Assistant for Z

Accelerated application lifecycle

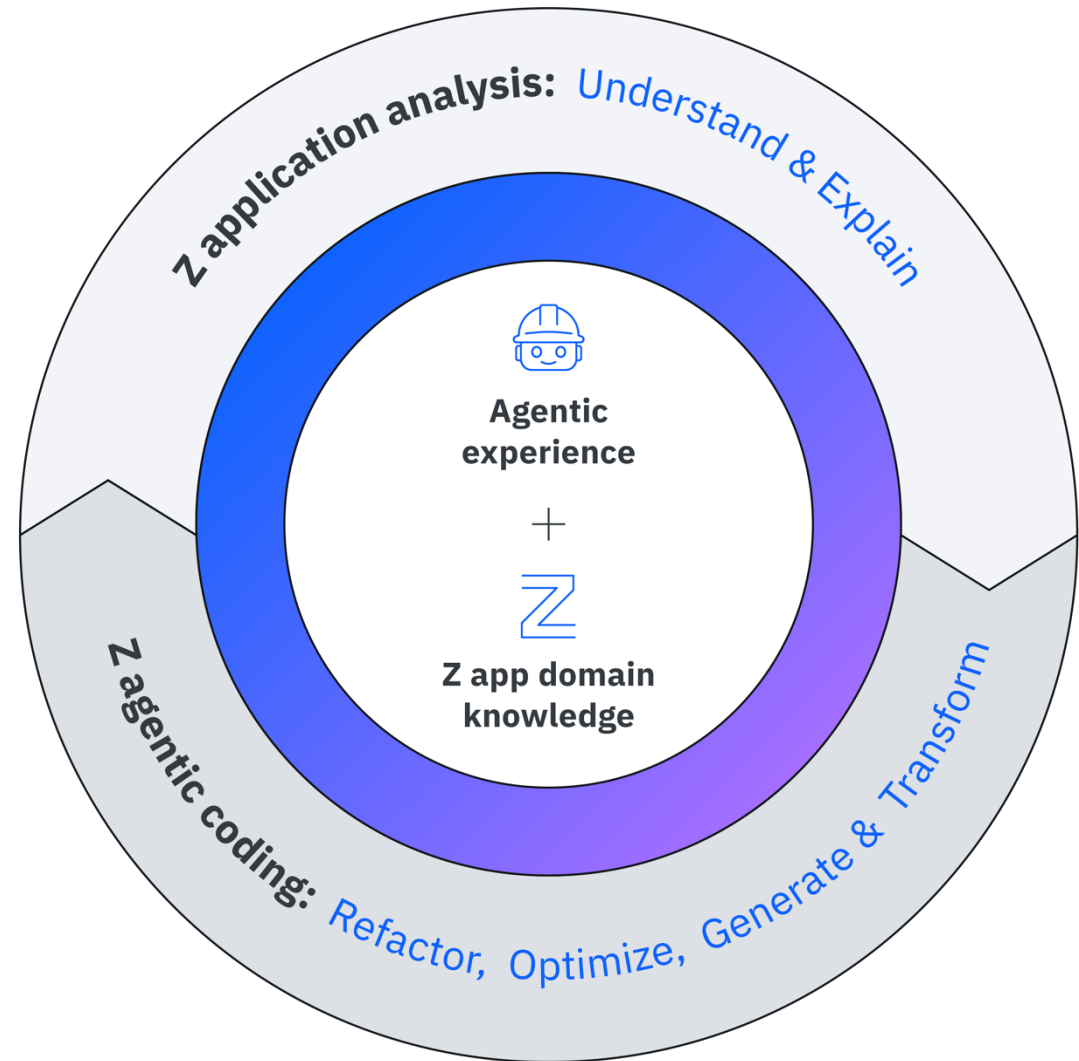
## Agentic experience

brings purpose-built AI and automated tooling for end-to-end modernization workflows, from application analysis to coding

## Z application domain knowledge

infuses tooling with Z specific language and middleware context

## AI-assisted mainframe application modernization



# Optimize

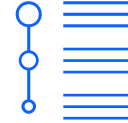
## Enhance COBOL code with prioritized performance insights

Supports batch COBOL 4 and 6, including measuring and reporting on the CPU time spent making Db2 calls in COBOL programs



### Provides performance insight

Conducts comprehensive analysis of COBOL modules using both static and dynamic analysis, enabling quick identification and resolution of performance issues.



### Provides prioritized and actionable recommendations

Ranks performance issues based on impact, enabling developers to focus on high-priority tasks for maximum efficiency.



### Identifies performance issues in source

Access source files and copybooks at the exact line of code to apply recommended fixes and enhancements



### Statement-level performance insights

Analyze CPU usage with statement-level insights to understand where CPU is being spent in programs, prioritizing and resolving lingering performance issues.

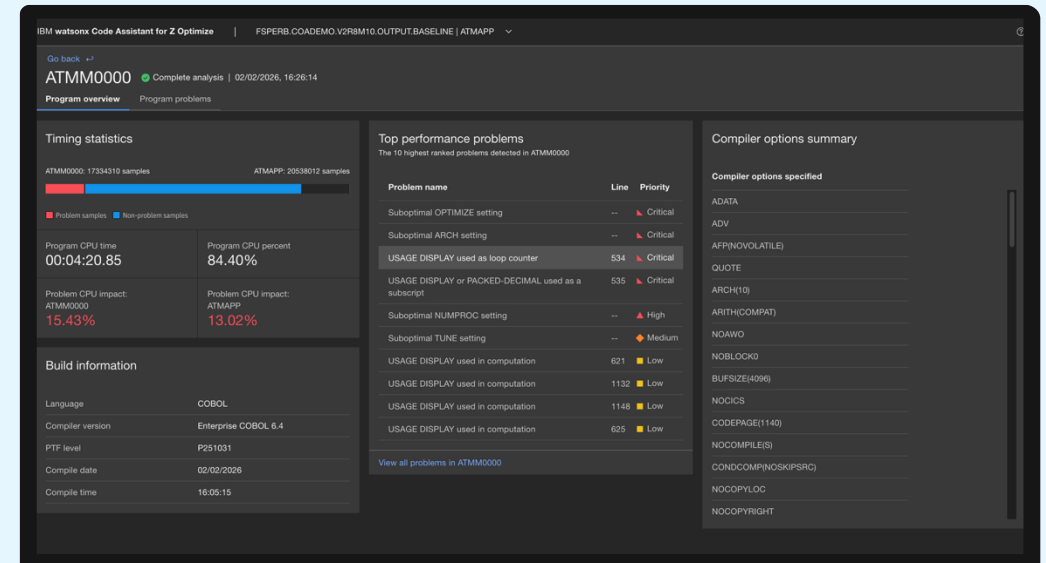
## Key Values

**Boost productivity** by enabling quick identification and resolution of performance issues

**Reduce skill gap** by allowing developers of all skill levels to independently resolve performance issues

**Quick resolution** of performance issues in COBOL source

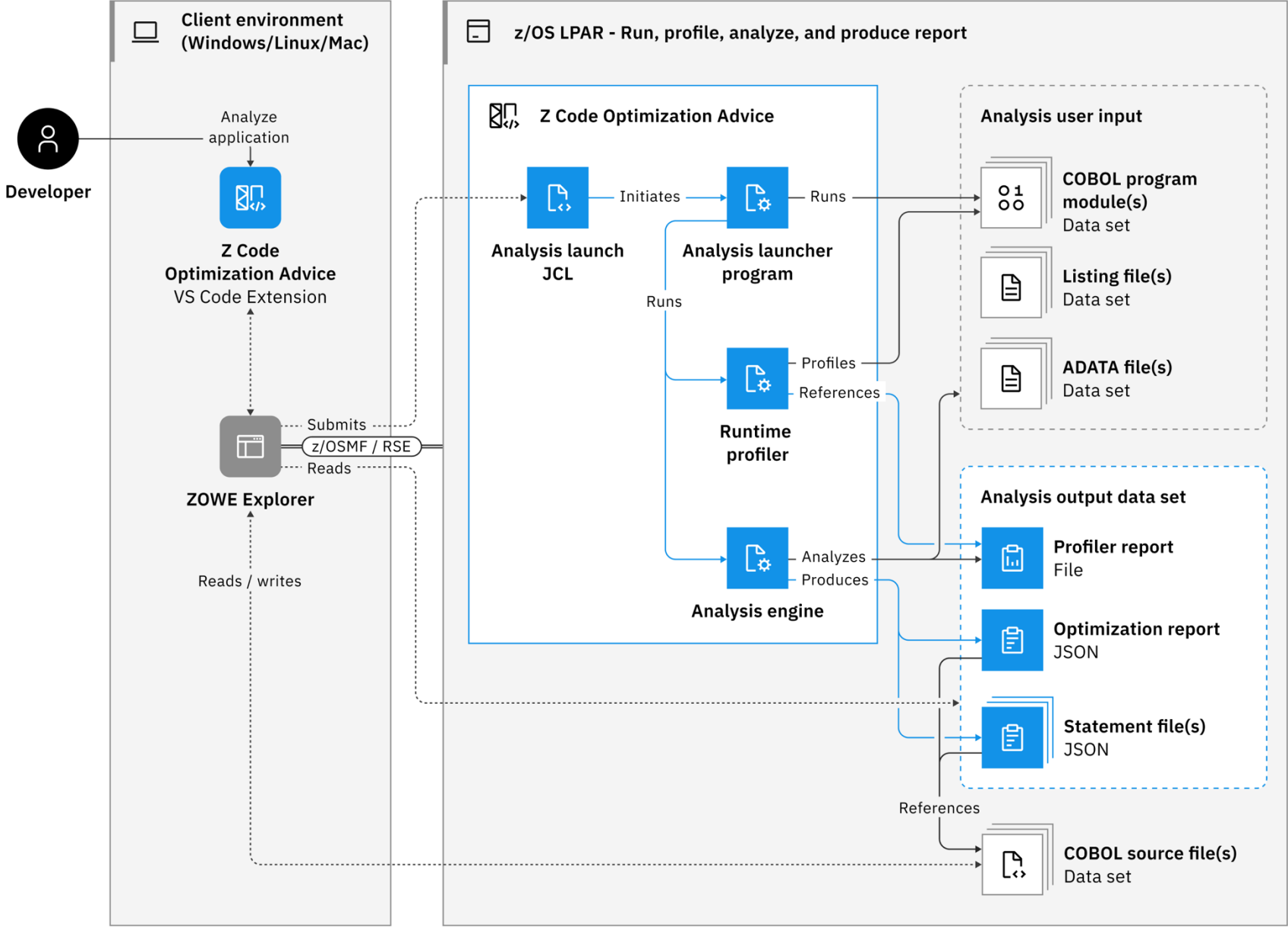
**Shift-left strategy** to proactively prevent performance problems before they reach production



# Optimize - Current architecture overview

There are two components:

1. The on-prem component profiles and analyzes the Batch programs running on the IBM Z Hardware to generate the report
2. The client-side VS Code extension displays the report data directly in the editor and links to the user's source code on host



— External call    — Inter-component call    - - - - API call    = Connection

# COBOL and Db2 support

Developers can now see how much of their application's CPU is spent executing Db2 related instructions.

This can help them quickly determine if they should focus on optimizing the COBOL source code or investigate Db2 performance issues using IBM's Db2 SQL performance analysis tools

*Db2 isn't the problem here. I'll focus on optimizing my COBOL!*

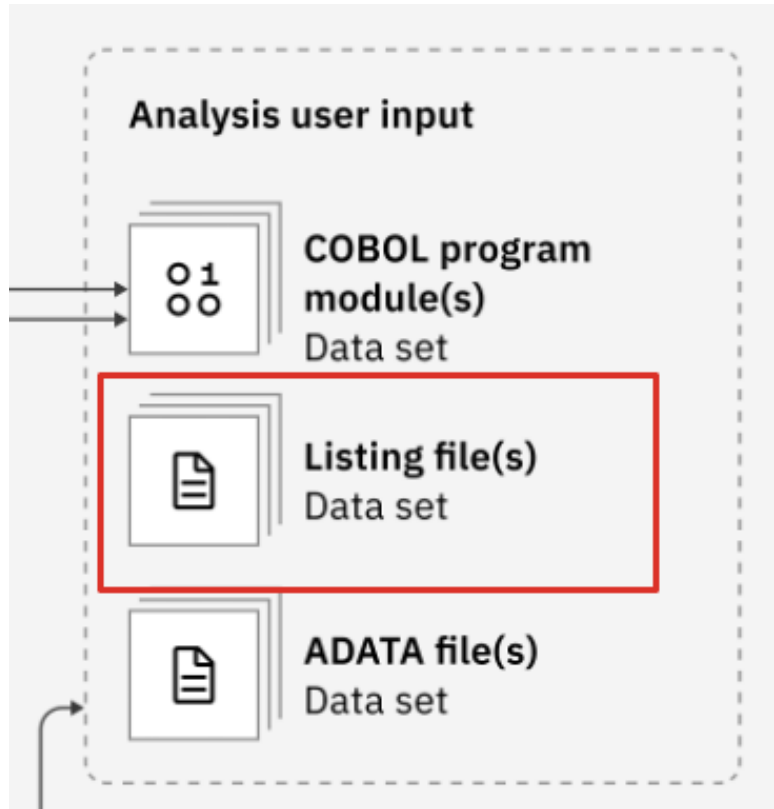


ATMCF000 CPU allocation

	Samples	CPU %
<b>Languages</b>	<b>780028</b>	<b>98</b>
COBOL	775128	97
COBOL Runtime	4900	1
<b>Subsystems</b>	<b>0</b>	<b>0</b>
Db2	6000	1
<b>System</b>	<b>0</b>	<b>0</b>
z/OS	0	0
z/OS Language Environment	0	0
<b>Others</b>	<b>12412</b>	<b>2</b>

# SMARTBIN Option

Enhancement for COBOL 6.4 and 6.5



**SMARTBIN** or **SMARTBIN(NONAMES)** – instructs the COBOL compiler to generate binary metadata, enabling the modules to be optimized by the IBM® Automatic Binary Optimizer (ABO) for z/OS® 2.2 or later. This is the default option

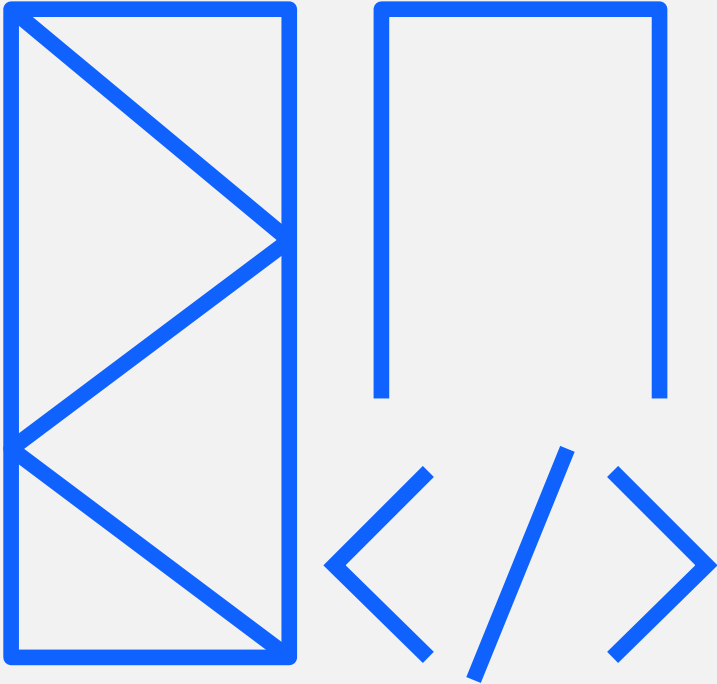
**SMARTBIN(NAMES)** – instructs the COBOL compiler to generate additional metadata for *IBM watsonx Code Assistant for Z Code Optimization Advice*. This replaces the need for Listing Files

Metadata generated by SMARTBIN option is stored in the NOLOAD section of the program object, making it easier to manage

<https://www.ibm.com/docs/en/cobol-zos/6.4.0?topic=options-smartbin>

<https://www.ibm.com/docs/en/cobol-zos/6.5.0?topic=options-smartbin>

# Demo



```
zosmf > ROBK.WCA4ZCOA.SCQIJCL > CQIINITJ
100  /**          variables. A setting of N defaults          *
101  /**          APPNAME equal to the provided PGMNAME.      *
102  /**                                                    *
103  /**          - for dataset allocation sample see CQIALC.   *
104  /**                                                    *
105  /*******
106  /** EXPORT SYMLIST=(JCLNAME,PGMNAME,COA,EXE,PROCNAME,LISTING,
107  /** ADATA,CONFIG,PROF,JSON,ENTRY,APPNAME)
108  /**
109  /** SET JCLNAME=ROBK.ATM06.SAMP.JCL(RCPI64Y) name
110  /** SET PGMNAME=ATMCF000          <- put COBOL program name
111  /** SET COA=ROBK.WCA4ZCOA.SCQIMOD1 - put COA loadlib location
112  /** SET EXE=ROBK.WCA4ZCOA.SCQIJCL <- put REXX library name
113  /** SET PROCNAME=N                <- put your.lib(procname)
114  /** SET LISTING=ROBK.WCA4ZCOA.LISTING(ATMCF000) listing[(member)]
115  /** SET ADATA=ROBK.WCA4ZCOA.ADATA(ATMCF000) adata dataset[(member)]
116  /** SET CONFIG=N                  <- put your.config.file here
117  /** SET PROF=ROBK.WCA4ZCOA.SYSPROFD(ATMCF000) ling dataset[(member)]
118  /** SET JSON=ROBK.CPI.JSON.ATMCF000(ATMCF000) dataset[(member)]
119  /** SET ENTRY=N                    <- put entry point [name]
120  /** SET APPNAME=N                  <- put application name
121  /**
122  /**IKJ EXEC PGM=IKJEFT01,REGION=0M,DYNAMNBR=25
123  /**SYSEXEC DD DSN=&EXE.,DISP=SHR
124  /**SYSTSPRT DD SYSOUT=*,DCB=(LRECL=132,RECFM=FBA,BLKSIZE=1320)
125  /**SYSPRINT DD SYSOUT=*
126  /**SYSTSIN DD *,SYMBOLS=JCLONLY
127  CQIINIT -
128  &JCLNAME -
129  &PGMNAME -
130  &COA -
131  &EXE -
132  &PROCNAME -
133  &LISTING -
134  &ADATA -
135  &CONFIG -
136  &PROF -
137  &JSON -
138  &ENTRY -
139  &APPNAME
140  /*
141
```

WELCOME TO COBOL UPGRADE ADVISOR | ATMCFF000 | ATMCFF000 report | GROUPA Report | APP01 Report | A Report | Start COBOL Inventory Scan

IBM watsonx Code Assistant for Z Code Optimization Advice

Application overview | Programs | Problems

### ATMCFF000 Timing statistics

CPU time  
**00:03:04.97**

Elapsed time: 00:03:06.60

Application samples: 1918173

EXCPs: 7286

Application run start time: 20:57:54 02/24/2025

Application run end time: 21:01:00 02/24/2025

### Top CPU-consuming programs

The 10 highest CPU-consuming programs called by ATMCFF000

Program	Language	CPU %	Problems
✓ ATMM0000 →	COBOL	84	20
✓ ATMSTACO →	COBOL	15	9
✓ ATMCFF00 →	COBOL	0	9
✓ ATMOFF00 →	COBOL	0	9
✓ ATMR TALG →	COBOL	0	9
✓ ATMR TBLG →	COBOL	0	9
✓ ATMR TCLG →	COBOL	0	9
✓ ATMSTBRO →	COBOL	0	9
✓ ATMSTCU0 →	COBOL	0	10
✓ ATMURND0 →	COBOL	0	38

[View all programs](#)

### ATMCFF000 CPU allocation

	Samples	CPU %
<b>Languages</b>	<b>1910646</b>	<b>100</b>
COBOL	1907417	99
COBOL Runtime	3229	0
<b>Subsystems</b>	<b>0</b>	<b>0</b>
Db2	0	0
<b>System</b>	<b>0</b>	<b>0</b>
z/OS	0	0
z/OS Language Environment	0	0
<b>Others</b>	<b>7527</b>	<b>0</b>

### Runtime options summary

**Runtime options specified**

- ABTERMENC(ABEND)
- AIXBLD(OFF)
- ALL31(ON)
- ANYHEAP(0000016384,0000008192,ANYWHERE,FREE)
- NOAUTOTASK
- BELOWHEAP(0000008192,0000004096,FREE)
- CBLOPTS(ON)
- CBLPSHPOP(ON)
- CBLQDA(OFF)
- CEEDUMP(0000000060,SYSDUMP=\*,FREE=END,SPIN=U,NALLOC)
- CHECK(ON)
- COUNTRY(US)
- DEBUG(OFF)
- DEPTHCONDLMT(0000000010)

**DATA SETS**

- zozmf
  - ROBK.WCA4ZCOA.SCQIJCL
    - CQIIFL
    - CQICONFG
    - CQIDDDDEF
    - CQIINIT
    - CQIINITJ
    - CQIIVPE
    - CQIIVPJ
    - CQIIVPS1
    - CQIIVPS2
    - CQIRECV
    - CQISMPE
  - ROBK.WCA4ZCOA.SCQIMOD1
  - ROBK.WCA4ZCOA.SYSPROFD
  - ROBK.WCA4ZCOA.V2R4M10.AFTER.JS...
    - ATMCF000
    - ATMG0000
    - ATMROFFL
  - ROBK.WCA4ZCOA.V2R4M10.BEFORE.J...
    - ATMCF000 ★
    - ATMG0000
    - ATMROFFL

**UNIX SYSTEM SERVICES (USS)**

- Favorites
- zozmf

**JOBS**

- Favorites
- zozmf
  - Use the search button to display jobs

IBM watsonx Code Assistant for Z Code Optimization Advice

[Go back](#) | **ATMM0000** | Complete analysis | 02/24/2025, 21:01:07

**Program overview** | Program problems

### Build information

Language	COBOL
Version	Enterprise COBOL 6.4
PTF level	P241211
Date compiled	02/24/2025, 15:24:15

### Timing statistics

ATMM0000 CPU time: 00:02:34.57 | ATMCFO00 CPU time: 00:03:04.97

CPU time	00:02:34.57
Program samples	1610623
Program CPU	84%

### Top performance problems

The 10 highest ranked problems detected in ATMM0000

Problem name	Line	Priority
Suboptimal OPTIMIZE setting	--	Critical
Suboptimal ARCH setting	--	Critical
USAGE DISPLAY used as loop counter	534	Critical
USAGE DISPLAY or PACKED-DECIMAL used as a subscript	535	Critical
Suboptimal NUMPROC setting	--	High
Suboptimal TUNE setting	--	Medium
USAGE DISPLAY used in computation	621	Low
USAGE DISPLAY used in computation	1132	Low
USAGE DISPLAY used in computation	1143	Low
USAGE DISPLAY used in computation	625	Low

[View all problems in ATMM0000](#)

### Compiler options summary

#### Compiler options specified

- ADATA
- ADV
- AFP(NOVOLATILE)
- QUOTE
- ARCH(10)
- ARITH(COMPAT)
- NOAWO
- NOBLOCK0
- BUFSIZE(4096)
- NOCICS
- CODEPAGE(1140)
- NOCOMPILE(S)
- CONDCOMP(NOSKIPSRC)
- NOCOPYLOC
- NOCOPYRIGHT

ZOWE EXPLORER

- DATA SETS
  - zosmf
    - ROBK.WCA4ZCOA.SCQIJCL
      - CQICNF
      - CQICONFG
      - CQIDDEF
      - CQIINIT
      - CQIINITJ
      - CQIIVPE
      - CQIIVPJ
      - CQIIVPS1
      - CQIIVPS2
      - CQIRECV
      - CQISMPE
    - ROBK.WCA4ZCOA.SCQIMOD1
    - ROBK.WCA4ZCOA.SYSPROFD
    - ROBK.WCA4ZCOA.V2R4M10.AFTER.JS...
    - ROBK.WCA4ZCOA.V2R4M10.BEFORE.J...
      - ATMCF000 ★
      - ATMG0000
      - ATMROFFL
    - ROBK.WCA4ZCOA.V2R4M10.UI99906.S...
    - rse
    - zftp
  - UNIX SYSTEM SERVICES (USS)
    - Favorites
    - zosmf
  - JOBS
    - Favorites
    - zosmf
      - Use the search button to display jobs

### IBM watsonx Code Assistant for Z Code Optimization Advice

[Go back](#)

# ATMM0000

Complete analysis | 02/24/2025, 21:01:07

Program overview | **Program problems**

## Performance problems

All problems detected in ATMM0000

Search problems

Problem name	Line	Data Items / Option
Suboptimal OPTIMIZE setting	--	OPTIMIZE(0)
Suboptimal ARCH setting	--	ARCH(10)
USAGE DISPLAY used as loop counter	534	COUNTER
USAGE DISPLAY or PACKED-DECIMAL us...	535	COUNTER
Suboptimal NUMPROC setting	--	NUMPROC(NOPFD)
Suboptimal TUNE setting	--	TUNE(10)
USAGE DISPLAY used in computation	621	WS-ATM-TXNFL-AMT-SIG
USAGE DISPLAY used in computation	1132	WS-ATM-TXNFL-AMT-SIG
USAGE DISPLAY used in computation	1143	WS-ATM-TXNFL-AMT-SIG
USAGE DISPLAY used in computation	625	WS-ATM-TXNFL-AMT-SIG
USAGE DISPLAY used in computation	1148	WS-ATM-TXNFL-AMT-SIG
100 or more bytes of padding being moved	521	RECEIVER, SENDER
Suboptimal FASTSRT setting	--	NOFASTSRT
Suboptimal AWO setting	--	NOAWO
Suboptimal BLOCK0 setting	--	NOBLOCK0
USAGE DISPLAY used in computation	975	WS-SEED

### USAGE DISPLAY or PACKED-DECIMAL used as a subscript

Problem class: Procedure Division

Program: ATMM0000

Line: Listing: 535  
Source: 271

Data Items: COUNTER

Priority: Critical

#### Problem description

USAGE DISPLAY or PACKED-DECIMAL data item COUNTER is used as a subscript to access a table. This requires additional overhead to convert COUNTER to BINARY.

#### Recommended actions

Use a BINARY data item as a subscript.

#### Justification

This removes the need to convert other numeric data types to BINARY before being used as a subscript, saving overhead.

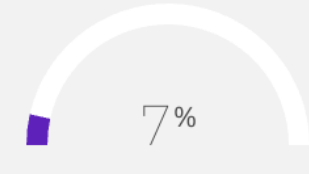
#### Considerations

No considerations at this time.

#### Performance impact

Problem samples: **105961**

Program samples: **1610623**





## DATA SETS

## zosmf

## ROBK.WCA4ZCOA.SCQIJCL

- QIIRFFL
- CQICONFG
- CQIDDDDEF
- CQIINIT
- CQIINITJ
- CQIIVPE
- CQIIVPJ
- CQIIVPS1
- CQIIVPS2
- CQIRECV
- CQISMPE

## ROBK.WCA4ZCOA.SCQIMOD1

## ROBK.WCA4ZCOA.SYSPROFD

## ROBK.WCA4ZCOA.V2R4M10.AFTER.JS...

- ATMCF000 ★
- ATMG0000
- ATMROFFL

## ROBK.WCA4ZCOA.V2R4M10.BEFORE.J...

- ATMCF000
- ATMG0000
- ATMROFFL

## UNIX SYSTEM SERVICES (USS)

## Favorites

## zosmf

## JOBS

## Favorites

## zosmf

Use the search button to display jobs

## IBM watsonx Code Assistant for Z Code Optimization Advice

## ATMCF000 performance report 11 | 02/25/2025, 00:59:56

[Environment details](#)

## Application overview

[Programs](#)
[Problems](#)

## ATMCF000 Timing statistics

## CPU time

00:00:45.90

Elapsed time 00:00:47.56

Application samples 474883

EXCPs 7227

Application run start time 00:58:58 02/25/2025

Application run end time 00:59:46 02/25/2025

## ATMCF000 CPU allocation

	Samples	CPU %
<b>Languages</b>	<b>467740</b>	<b>98</b>
COBOL	466119	98
COBOL Runtime	1621	0
<b>Subsystems</b>	<b>0</b>	<b>0</b>
Db2	0	0
<b>System</b>	<b>0</b>	<b>0</b>
z/OS	0	0
z/OS Language Environment	0	0
<b>Others</b>	<b>7143</b>	<b>2</b>

## Top CPU-consuming programs

The 10 highest CPU-consuming programs called by ATMCF000

Program	Language	CPU %	Problems
✓ ATM0000 →	COBOL	62	8
✓ ATMSTACO →	COBOL	34	5
✓ ATMSTCU0 →	COBOL	1	6
✓ ATMCFEEO →	COBOL	0	5
✓ ATMOFEE0 →	COBOL	0	5
✓ ATMRTALG →	COBOL	0	5
✓ ATMRTCLG →	COBOL	0	5
✓ ATMSTBR0 →	COBOL	0	5
✓ ATMURNDO →	COBOL	0	34
✓ ATMVITFO →	COBOL	0	11

[View all programs](#)

## Runtime options summary

## Runtime options specified

ABPERC(NONE)

ABTERMENC(ABEND)

AIXBLD(OFF)

ALL31(ON)

ANYHEAP(0000016384,0000008192,ANYWHERE,FRE E)

NOAUTOTASK

BELOWHEAP(0000008192,0000004096,FREE)

CBLOPTS(ON)

CBLPShPOP(ON)

CBLQDA(OFF)

CEEDUMP(0000000060,SYSOOT=\*,FREE=END,SPIN=U NALLOC)

CHECK(ON)

COUNTRY(US)

DEBUG(OFF)

**DATA SETS**

- zozmf
  - ROBK.WCA4ZCOA.SCQIJCL
    - CQIIFL
    - CQICONFG
    - CQIDDEF
    - CQIINIT
    - CQIINITJ
    - CQIIVPE
    - CQIIVPJ
    - CQIIVPS1
    - CQIIVPS2
    - CQIRECV
    - CQISMPE
  - ROBK.WCA4ZCOA.SCQIMOD1
  - ROBK.WCA4ZCOA.SYSPROFD
  - ROBK.WCA4ZCOA.V2R4M10.AFTER.JS...
    - ATMCF000 ★
    - ATMG0000
    - ATMROFFL
  - ROBK.WCA4ZCOA.V2R4M10.BEFORE.J...
    - ATMCF000
    - ATMG0000
    - ATMROFFL

**UNIX SYSTEM SERVICES (USS)**

- Favorites
- zozmf

**JOBS**

- Favorites
- zozmf
  - Use the search button to display jobs

# IBM watsonx Code Assistant for Z Code Optimization Advice

Go back → **ATMM0000** Complete analysis | 02/25/2025, 00:59:56

Program overview | Program problems

## Build information

Language	COBOL
Version	Enterprise COBOL 6.4
PTF level	P241211
Date compiled	02/25/2025, 00:15:15

## Timing statistics

ATMM0000 CPU time: 00:00:28.16    ATMCF000 CPU time: 00:00:45.90

CPU time	00:00:28.16
Program samples	296629
Program CPU	62%

## Top performance problems

The 10 highest ranked problems detected in ATMM0000

Problem name	Line	Priority
Suboptimal FASTSRT setting	--	Unmeasured
Suboptimal AWO setting	--	Unmeasured
Suboptimal BLOCK0 setting	--	Unmeasured
USAGE DISPLAY used in computation	975	Unmeasured
100 or more bytes of padding being moved	521	Unmeasured
Even-length PACKED-DECIMAL	99	Unmeasured
Suboptimal HGPR setting	--	Unmeasured
Suboptimal STGOPT setting	--	Unmeasured

[View all problems in ATMM0000](#)

## Compiler options summary

### Compiler options specified

- ADATA
- ADV
- AFP(NOVOLATILE)
- QUOTE
- ARCH(13)
- ARITH(COMPAT)
- NOAWO
- NOBLOCK0
- BUFSIZE(4096)
- NOCICS
- CODEPAGE(1140)
- NOCOMPILE(S)
- CONDCOMP(NOSKIPSRC)
- NOCOPYLOC
- NOCOPYRIGHT

# COBOL and Db2 support

Developers can now see how much of their application's CPU is spent executing Db2 related instructions.

This can help them quickly determine if they should focus on optimizing the COBOL source code or investigate Db2 performance issues using IBM's Db2 SQL performance analysis tools

*Db2 isn't the problem here. I'll focus on optimizing my COBOL!*



ATMCF000 CPU allocation

	Samples	CPU %
<b>Languages</b>	<b>780028</b>	<b>98</b>
COBOL	775128	97
COBOL Runtime	4900	1
<b>Subsystems</b>	<b>0</b>	<b>0</b>
Db2	6000	1
<b>System</b>	<b>0</b>	<b>0</b>
z/OS	0	0
z/OS Language Environment	0	0
<b>Others</b>	<b>12412</b>	<b>2</b>

# Statement Performance Table

Developers can view CPU usage of their program code statement by statement in a data table within their VS Code editor.

When a COBOL source file is open in the editor the performance data for that file automatically loads into the table and is available to view.

This performance insight helps developers to quickly identify hotspots in their programs and where to focus their optimization effort where it matters most.

The screenshot shows the VS Code editor with a COBOL source file open. The 'OPTIMIZE' panel on the right displays a 'Statement Performance Table' for the program 'ATMM0000'. The table lists source lines, program CPU percentages, and sample counts for various statements. A 'Timing Statistics' section is also visible below the table.

Source line	Program CPU %	Samples	Statement
256	0.0000	1	move 2 to sender
257	0.0002	6	move sender to receiver
258	0.0004	10	PERFORM I000-INITIALIZATION
259	0.0017	47	IF WS-ATMM0000-RETURN-CODE-OK AND WS-MAIN-PROCESS-TXN
260	0.0003	7	PERFORM C000-PROCESS-MAIN-LOGIC
262	0.0012	33	IF WS-ATMM0000-RETURN-CODE-OK AND WS-MAIN-PROCESS-TXN AND WS...
264	0.0000	0	SET WS-XFR-TXN-DEP-OFFSET TO TRUE
265	0.0007	18	PERFORM I100-INIT-FOR-PROCESS
266	0.0000	1	PERFORM C000-PROCESS-MAIN-LOGIC
268	0.0009	25	PERFORM W000-WRAP-UP
270	13.6634	368752	PERFORM VARYING COUNTER FROM 1 BY 1 UNTIL COUNTER > 10000

Timing Statistics			
	STATEMENT	ATMCF000	ATMM0000
CPU time	00:00:35.52	00:05:08.28	00:04:20.22
Samples	368752	3197083	2698826

# IBM watsonx Code Assistant for Z *Optimize*

## Complementary Products and Use Cases

### Watsonx Code Assistant for Z *Optimize*

**User:** Developers

**Use Case:**

Enables developers to quickly improve the performance of their COBOL applications by providing prioritized insights and recommendations linked directly to high-impact performance issues in their source code.

**Value:**

- Find performance issues faster within the source code compile/runtime options
- Produces a prioritized list of recommendations/fixes, based on programs actual runtime performance

### IBM Automatic Binary Optimizer (ABO)

**User:** Operations and developers

**Use Case:**

Developers can optimize already compiled COBOL code. There is no source code or options tuning required.

**Value:**

- Achieves comparable performance to the latest COBOL compiler but needs no source code or compiler migration.
- Optimizes from already compiled modules

### IBM Enterprise COBOL Compiler

**User:** Developers

**Use Case:**

Compiles and optimizes source to binaries.

**Value:**

- Improve program performance and adds IBM Z hardware specific tuning
- *Optimize's* performance recommendations for source and options are targeted for IBM Enterprise COBOL compiler users.
- Additional lower-level compiler optimizations and IBM hardware usage are added by the compiler for more performance.

### IBM Application Performance Analyzer

**User:** System admin / operations

**Use Case:**

Focuses on broader system wide performance. Analyzes areas such as sub-system performance, I/O performance, and wait times

**Value:**

- Provides a source program mapping capability
- Provides production analysis, enabling the user to evaluate trends in resources, data volume, and application behavior

### Static analysis tools

**User:** Developers

**Use Case:**

Static analysis used to scan source code to identify quality issues throughout development cycle

**Value:**

- Integrated into CI/CD pipeline to enforce the company's code quality standard
- Provides overview of health and quality of source code. The code analyzers detect bugs, logic errors, and resource leaks

# COBOL Upgrade Advisor for z/OS 1.1 (CUAZ)

CUAZ **accelerates** and **simplifies** upgrade to Enterprise COBOL 6

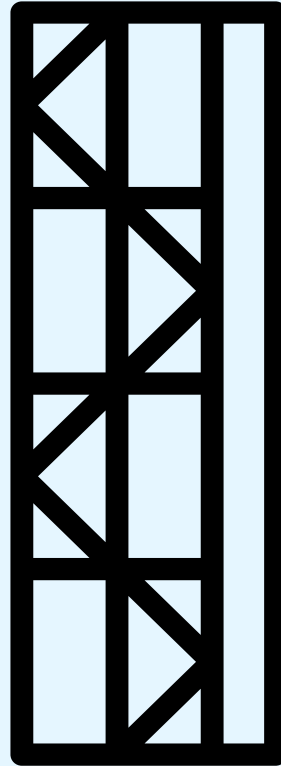
Initial GA – May 2025  
Next release expected in 1Q26\*

Product has 2 licensed parts:

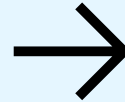
- Virtual Server
- Authorized User licenses

\*IBM reserves the right to modify this date without prior notice

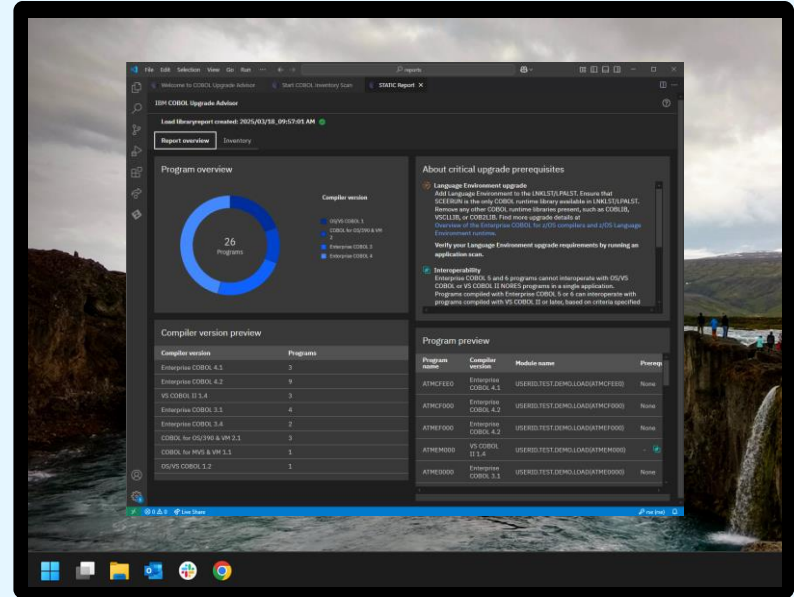
Covered in COBOL Upgrade Advisor session  
Feb 26, 2026 10:30 – 11:30



CUAZ analysis engine running on IBM Z



Zowe connects the mainframe to the workstation



CUAZ output displayed on workstation via VS Code

# Key Features



## Planning insights

- Visual summary of COBOL programs in a VS Code interface
- View compiler version, date of last compilation, compiler options and more
- Judge the size and complexity of your upgrade project
- Export COBOL inventory as CSV



## Resolving Prerequisites

- List of actionable pre-upgrade tasks per program
- Confirm LE is enabled
  - Identify potential interoperability issues
  - Use CUAZ [AI assisted Smart Code Converter](#) to automatically identify outdated COBOL syntax and provide recommended fixes

Ensures relevant upgrade tasks are not missed



## Recompiling Assistance

- Identifies deprecated options with removal/replacement guidance
- Identifies new compiler options available in COBOL 6, including ones to detect invalid data problems



## Run tests and fix invalid data problems

- Identifies invalid data and code problems based on runtime messages, with direct links to the exact source line, making it easy to jump in and fix issues fast
- [AI assisted](#) recommendations for fixing invalid data

## IBM Sponsor User program

### Shape the future of IBM

#### What we do

At IBM, we put our users at the **center** of our work to create amazing user experiences.

By **incorporating feedback** from our clients at every step of the design and development process, we build products that solve real user problems.

Together, we'll create **exceptional experiences** for you and the marketplace!

#### Benefits for You

**Connect directly** with our Product and Design teams who are eager to hear about your needs and perspectives.

**Insider access** to early prototypes and confidential information about future product plans.

**Influence** on our product roadmap and future releases.

#### How to Join

Let your IBM contact (preferably your design researcher) know you'd like to become a Sponsor User or drop an email to [vikash-kumar@ibm.com](mailto:vikash-kumar@ibm.com) and they will onboard you.

Or Fill out this form by clicking this [link](#) or scanning the below QR code to express your interest:





# Resources

- [IBM COBOL Upgrade Advisor \(CUAZ\) resources](#)
- Announcement letter: [Link](#)
- Product page: [Link](#)
- Blogs: [Link](#)
- New Feature Ideas: [Link](#)
- IBM Documentation – [Link](#)
- **Demos:**
  - [Interactive demo](#) – [Link](#)
- **Demo/POC resources:**
- zVA self-serve trial using preconfigured test data. Work with your sales rep.
- 90-day trial. Work with your sales rep.

## COBOL upgrade resources

COBOL Upgrade Portal: [Link](#)

Includes customer success stories, FAQs, upgrade videos and more

## COBOL upgrade services

IBM Expert Labs is ready to help with CUAZ with 2 upcoming offerings to help you with implementation and assistance.

IBM Consulting is ready to help with CUAZ. IBM Consulting has many years of experience with COBOL upgrade services.

# Resources

## Introduction to COBOL Learning Resources

[Open Mainframe COBOL Programming Course](#)

[Introduction to COBOL with VSCode](#)

Explore the newly launched COBOL-Java interoperability portal: [Link](#)

IBM Semeru Runtime: [Link](#)

IBM COBOL for AIX: [Link](#)

IBM COBOL for Linux on x86: [Link](#)



IBM Enterprise COBOL

Product page: [Link](#)

Blogs: [Link](#)

Resources: [Link](#)

New Requirement Ideas: [Link](#)



IBM Automatic Binary Optimizer for z/OS

Product page: [Link](#)

Blogs: [Link](#)

Resources: [Link](#)

New Requirement Ideas: [Link](#)



- [IBM Automatic Binary Optimizer Assistant Demo](#)

- [Increase performance of COBOL applications with Automatic Binary Optimizer](#)



[COBOL upgrade services](#) available from IBM

IBM TechXchange CUAZ [session replay](#)

Innovate and co-create with IBM Client Engineering teams: [Link](#)



See the [COBOL Migration Portal](#) for migration resources

[Why You Should Migrate](#)

[How to Migrate](#)



Watsonx Code Assistant for Z

Product page: [Link](#)

# Your feedback is important!

## Submit a session evaluation for each session you attend:

[www.share.org/evaluation](http://www.share.org/evaluation)







