

Next generation Cobol debugging

Petr Vacula – May 31 – 3.17

Prague Technology Days

May 30 - June 1, 2018

For Informational Purposes Only

This presentation was based on current information and resource allocations as of **May 2018** and is subject to change or withdrawal by CA at any time without notice. Notwithstanding anything in this presentation to the contrary, this presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. The development, release and timing of any features or functionality described in this presentation remain at CA's sole discretion. Notwithstanding anything in this presentation to the contrary, upon the general availability of any future CA product release referenced in this presentation, CA will make such release available (i) for sale to new licensees of such product; and (ii) to existing licensees of such product on a when and if-available basis as part of CA maintenance and support, and in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to CA maintenance and support on a when and if-available basis. In the event of a conflict between the terms of this paragraph and any other information contained in this presentation, the terms of this paragraph shall govern.

Certain information in this presentation may outline CA's general product direction. All information in this presentation is for your informational purposes only and may not be incorporated into any contract. CA assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, CA provides this presentation "as is" without warranty of any kind, including without limitation, any implied warranties or merchantability, fitness for a particular purpose, or non-infringement. In no event will CA be liable for any loss or damage, direct or indirect, from the use of this document, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if CA is expressly advised in advance of the possibility of such damages. CA confidential and proprietary. No unauthorized copying or distribution permitted.

Abstract

CA InterTest is more visual, easier to use, requires fewer steps, and less maintenance!

Ensure your critical mainframe applications are maintained and enhanced efficiently and effectively by NexGen developers who don't have the benefit of the collective insights gained by decades of experience. This session will describe how CA InterTest's Visual Debugger helps to expedite both problem resolution and general application understanding by leading developers down the same path the code takes using images easily translated to program paths. It will also show how CA InterTest will allow developers to use the TEST(SOURCE) NOLOAD compile option and then leverage the DWARF method of obtaining symbolic information for debug purposes, eliminating the requirement to match a load module with a listing and maintain a symbolic file.

Visual Debugger

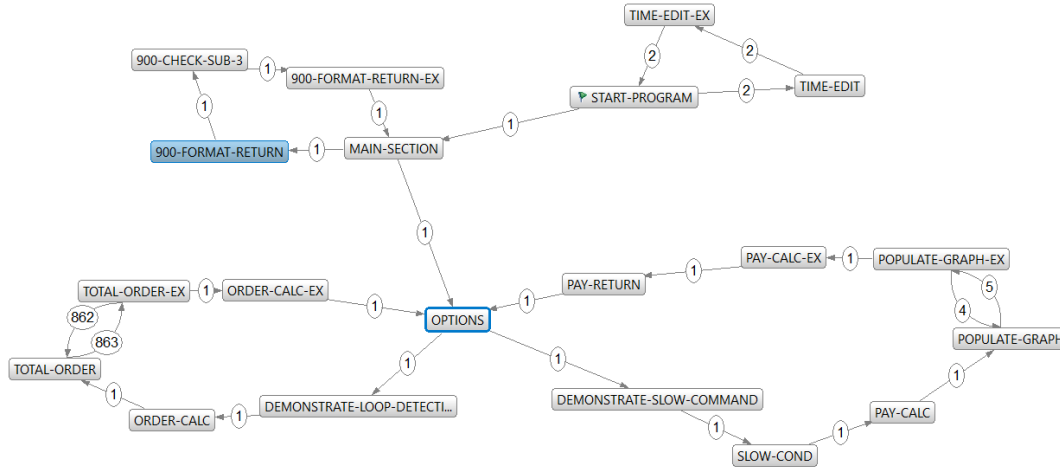
Legacy COBOL applications

Problems

- Experience workforce shortage
- Maintaining legacy applications requires understanding
- Enhancing legacy application almost impossible
- Learning curve for new mainframers is steep
- NextGen developers don't find green screen cool enough

InterTest Visual Debugger

- Visual representation of a running COBOL (CICS/Batch)
 - “A picture is worth a thousands words”





demo

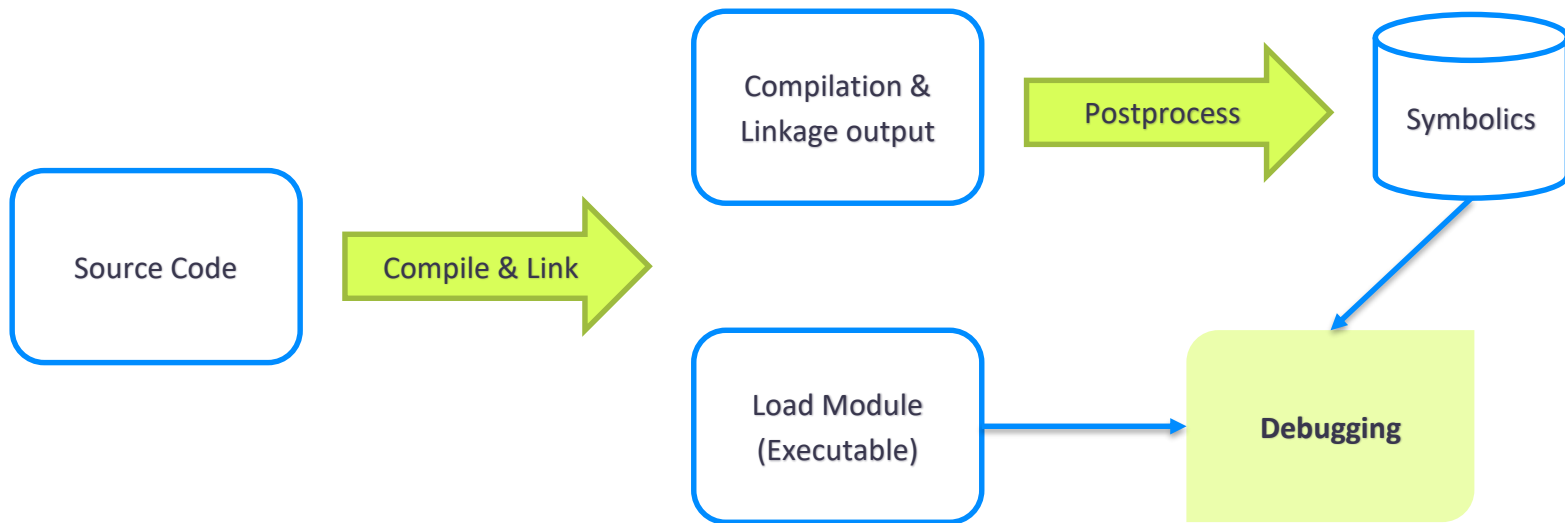


DWARF Symbolic support

Symbolic Support definition

- Without Symbolic Support, debugging experience is reduced to stepping through processor instructions.
- Correlation between executable and source code is essential for meaningful debugging experience.
- Symbolic Support creates mapping between executable and source, making the debugging experience viable.

Debugging using Symbolics repository



Symbolic Support repository

Difficulties

- Need to maintain repositories
- Synchronize between LPARs
- Not matching symbolic

DWARF definition

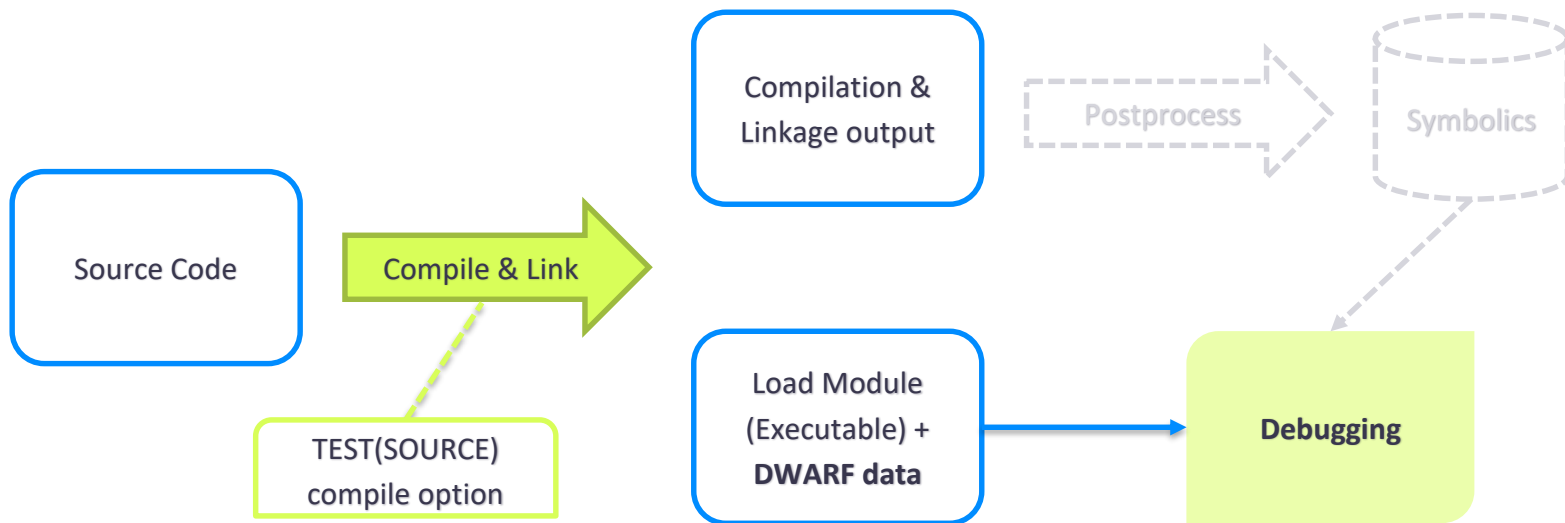
“DWARF is a widely used, standardized debugging data format.”

DWARF represents so called *extended source* that is included in a load module.

<https://en.wikipedia.org/wiki/DWARF>

Debugging using DWARF symbolics

Need to maintain repositories
Synchronize between LPARs
Not matching symbolic



Use case comparison

Symbolic Support repository

1. Compile & Link
2. Postprocess listings into repository
3. Distribute new repository
4. Distribute new load module
5. **Debug**

DWARF Symbolic Support

1. Compile with **TEST(SOURCE)** & Link
2. Distribute new load module
3. **Debug**

InterTest for CICS

Interactive debugger of CICS applications with Eclipse based GUI.

DWARF Symbolic Support currently under development

Initial drop will be available soon through validate.ca.com



Thank You.



Petr Vacula

Principal Product Owner

Petr.Vacula@ca.com



@cainc



slideshare.net/CAinc



linkedin.com/company/ca-technologies

ca.com

Call for Speakers Now Open

ca World[®] '18

November 12-16, 2018
Miami Beach Convention Center

Share your experience. Be the teacher. Enhance your resume.

Register your session today!

Learn more: ca.com/caworld

