Binder for Beginners

Things You Ought to Know,
Whether You Knew It or Not!

Session 8758

Barry.Lichtenstein@us.ibm.com

IBM Poughkeepsie, New York

Trademarks

Agenda

- What is the binder and why you need to know about it
- The most prominent users (Where you see it)
- The most frequently used options etc. (Let’s talk options)
- The less obvious users (*That* was the binder?)
- Deeper dive into binder processing
- Problem diagnosis
- More advanced stuff

What is the binder?

- Wikipedia® under linker (computing): “In IBM mainframe environments such as OS/360 this program is known as a linkage editor.”

- In z/OS it is the program management binder

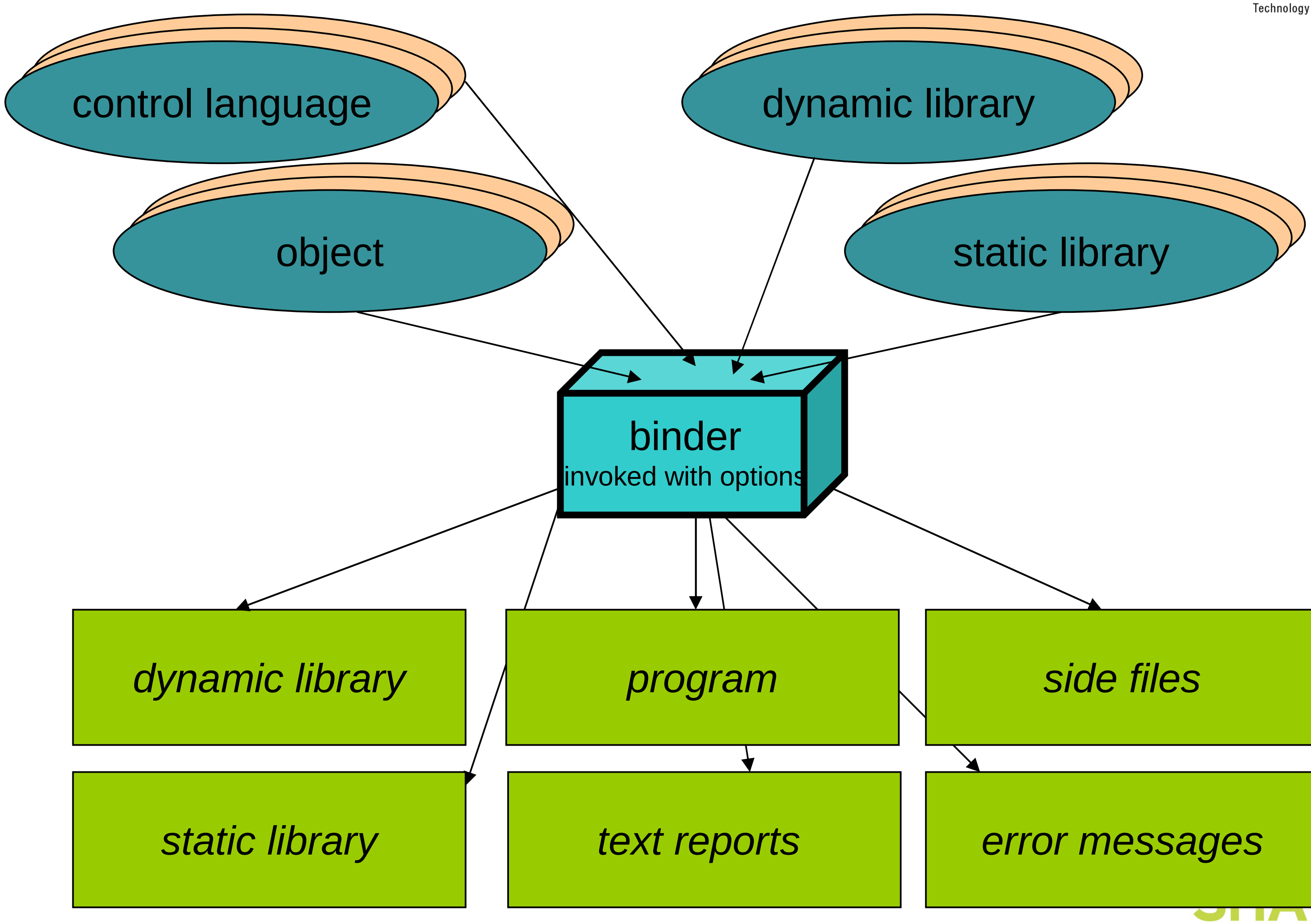
program management binder

- BCP exclusive base element
  - Wave 0
- z/OS system linker
- Related utilities
- Programming interfaces

program management binder …

- The binder converts the output of language translators and compilers into an executable program unit …
  - … that can either be read directly into virtual storage for execution or stored
- The binder processes object modules, load modules and program objects…
  - *link-editing* or *binding* multiple modules into a single load module or program object
  - … with contiguous virtual storage addresses

program management binder …



program management binder …

- Symbol resolution
  - all *external* symbol references which need to be satisfied
    - between all input parts
- Relocation
  - all modules combined, relocated relative to origin address
    - zero (or start of segment)
  - final relocation is done by the loader
    - based on information created by the binder

binder inputs

- SYSLIN ddname
  - object modules
    - OBJ, XOBJ, GOFF
  - program modules (executables)
    - load modules
    - program objects
  - control statements

binder outputs

- SYSPRINT ddname
  - text reports
  - error messages
- SYSLMOD ddname
  - program module / static or dynamic libraries
- SYSDEFSD ddname
  - side-file for dynamic libraries

Where you see it

- **PGM=IEWL** (in JCL)
  - True name
    - **IEWBLINK** (default Link-Edit Utility for **SMP/E**)
  - aliases ala linkage editor names
    - HEWL, HEWLH096
    - HEWLDRGO, HEWLOAD, HEWLOADR
  - aliases of the modern day for binder loader
    - IEWBLDGO, IEWBLODI, IEWBLOAD
    - LOADER
    - IEWLDRGO, IEWLOADI, IEWLOAD, IEWLOADR
  - binder aliases of the modern day
    - IEWL, LINKEDIT
  - alias for customized options
    - IEWBODEF
    - Caution! for sysprogs, rarely used

a moment of legacy…

- *Invocations of actual* <u>linkage editor</u> and <u>batch loader</u>
  - *HEWLD**
    - *Any remaining invocations of these are batch loader*
  - *IEWL* or HEWL**
    - *Any remaining invocations of these are linkage editor*
  - *If you have any of these, we'd like to know!!!*
- ***<u>NOTE:</u>*** *Program Management loader used for PGM=yourpgm*
  - *That is <u>not</u> the Binder!*
  - *It's what is <u>mostly</u> used for program invocation*

Let’s talk options!

- options for SYSPRINT
- to LET or not to LET
- options, options everywhere
- program changing options

options for SYSPRINT

- **LIST**, **MAP**, **XREF**
  - most common (more on these later…)
  - SMP/E Link-Editor Utility defaults:
    - LET, LIST, NCAL, XREF
    - NCAL once upon a time was unconditionally set
      - *now based on CALLIBS*
    - If you specify overrides, you must list the others too!
    - SMP/E is picky (it's *not really* JCL)

options for SYSPRINT

- **INFO** about service level of binder
- **MSGLEVEL** of lowest severity messages to write
  - Default is all (0)
  - Suppresses text, no change to return code!
- **LISTPRIV** for a listing of “private code” sections
  - and if so make it an error (YES)
  - or just informational (INFORM)
- **STRIPSEC/STRIPCL** to remove and list “unneeded” stuff
  - To see the “removed” report requires **MAP** option

to LET or not to LET

- **LET**=number
  - “LET this be an executable, even if the return code is equal to or less than number”
  - EXECUTABLE is an attribute in the program and in the case of datasets, in the directory
    - NX in ISPF member list means “Not Executable”
    - Nothing to do with the UNIX execute permission
  - “LET” in batch means LET=8
    - Unspecified or “NOLET” means LET=4

to LET or not to LET…
what was the question ??

- **STORENX**
  - STORENX controls whether the "Not Executable" program is saved
    - The default is NOREPLACE (same as NO)…
    - That means by default, a "Not Executable" program WILL BE SAVED if it does not already exist!
    - STORENX=NEVER
      - *introduced in z/OS V1R8 -- but not the default!*

to LET or not to LET…
what if I LET it STORENX ?

- Depends where and how invoked…
  - from batch

```
CSV016I REQUESTED MODULE STOREDNX IS NOT EXECUTABLE
CSV028I ABEND706-04  JOBNAME=BARRYLR   STEPNAME=GO
IEA995I SYMPTOM DUMP OUTPUT  467
SYSTEM COMPLETION CODE=706  REASON CODE=00000004
```

  - from UNIX… usually you will see…

```
BARRYL [478] /u/barryl/binder/SHARE/SHARE116 $  ./a.out
IEWPLMH: ./a.out 14: FSUM7351 not found
```

    - …shell semantics for a failed spawn, to treat as a shell script
    - as a DLL (a bug?)

```
CEE3512S An HFS load of module SNX.dll failed. The system return code was 0000000130; the reason code was 0BDF0000.
         From entry point main at compile unit offset +0000044A at entry offset +0000044A at address 21708DDA.
```

Options, options everywhere!

- **OPTIONS=***ddname*
  - primarily invented to overcome JCL limitations…
    - typically in-stream data set
  - but can be convenient for example to have files of options common to a set of JCL
    - *making it easy to update options without changing JCL etc.*

Options, options everywhere! …

- Options on control statements
  - MODE - AMODE, RMODE options
  - ENTRY - EP option
  - SETOPT – just about everything!
    - Nothing “environmental” please!

Options, options everywhere! …

Options precedence (low to high)

1. Installation options from IEWBODEF
2. Primary invocation options, from one of the following:
   1. The PARM field of the JCL EXEC statement
   2. The first parameter passed to IEWBLINK, IEWBLOAD, etc.
   3. The PARMS parameter of IEWBIND FUNC=STARTD
3. ***The IEWPARMS DD statement -- introduced in z/OS V1R11 !***
4. The OPTIONS parameter of IEWBIND FUNC=STARTD
5. IEWBIND_OPTIONS environment variables via the ENVARS parameter of IEWBIND FUNC=STARTD
6. Dynamic option changes from either:
   1. Options set from attributes by an INCLUDE -ATTR control statement or
   2. The SETOPT control statement, or
   3. The PARMS parameter, followed by the OPTION/OPTVAL parameter, of IEWBIND FUNC=SETO

Program changing options

- **COMPRESSion=YES**

  - Can significantly shrink size of <u>program object</u> on disk
  - <u>**No Change**</u> to size of in-storage program!
    - <u>No Change</u> to the program itself (loader / run-time data), only binder owned data
  - Distinguished in **Save Module Attributes** (**LIST** output):

```
    MODULE SIZE (HEX)   00002BFC
    DASD SIZE (HEX)     0000D000        (this had been 00015000)
```

  - Requires COMPAT(ZOSV1R7)

```
PROGRAM TYPE        PROGRAM OBJECT(FORMAT 4 OS COMPAT LEVEL z/OS V1R7 )
```

    - AUTOmatically happens, if beneficial, with this or later COMPAT level
      - *default is COMPAT(MIN)*
      - *will still execute back to ZOSV1R3*
        - *but no rebind, AMBLIST, ZAP, etc.*

- **EDIT=NO**

  - *Permanently deletes* the data that COMPRESS would have compressed
  - Thus *limited* rebind, AMBLIST, ZAP, etc. *anywhere*

```
    MODULE SIZE (HEX)   00002BFC
    DASD SIZE (HEX)     00005000
```

    - Limitation is binder based so:
      - *AMBLIST of LM works because it doesn't use binder*
      - *Binder supports limited processing of INTENT=ACCESS LM*

Program changing options …

- **DYNAM=DLL** – Dynamic Link Library
  - exported symbols to SYSDEFSD as IMPORT control statements
  - Control information (visible in **MAP** and AMBLIST output, macros in ‘SYS1.MACLIB’)
    - IEWBLIT section B_LIT class – Loader Information Table
    - IEWBCIE section B_IMPEXP class – Import/Export table
- Language Environment high-level languages and High Level Assembler (LE provides macro)
- Execution requires Language Environment run-time support
  - Function “descriptors” enable dynamic linking
- Exploits deferred load C_WSA[64] class
  - Writable / Static Area
  - LE controls unique instance for each “enclave” of execution
- Dynamic resolution follows all static resolution

Program changing options …

- **SIGN=YES –** Program Signing – introduced in z/OS V1R11 !
  - Digital signature is written into program object
    - Constructed based on program data
    - Becomes part of program
    - PDSEs supported only!
  - Requires SAF/RACF setup & services
    - Require keyring or PKCS #11 token to sign
    - Program must be identified as requiring digital signature for execution
      - *… loader verifies correct digital signature prior to execution*
  - Cannot use traditional (SMP/E) service methodology since only signer can bind
    - Could use EDIT=NO

That was the binder?

- Who was that masked binder?
  - The usual suspects:
    - Batch LINKEDIT, IEWL, etc.
  - Invoked as a program call:
    - SMP/E (it's *not really* JCL!)
    - TSO LINK, LOAD, LOADGO
    - ld command (UNIX)
  - Using the binder Application Programming Interfaces (APIs)
    - c89 (c++), cob2, pli, xlc (xlC)
    - IEBCOPY (sometimes!)
    - ZAP
    - AMBLIST

That was the binder? …

- Where did *that* come from??? (the wonderful world of UNIX)
  - makefiles
    - Watch out for environment variables which become make macros
      - *LDFLAGS*
  - c89 – YAEV (“yet another environment variable”)
    - _C89_OPTIONS
    - _C89_OPERANDS
  - ld- yikes, just like (guess why!)…
    - _LD_OPTIONS
    - _LD_OPERANDS

That was the binder? …

- You know it's the binder if…
    - **<u>All</u>** IEW2xxxx messages
        - *SYSPRINT and SYSTERM in batch*
        - *IEWDIAG also!*
- IEWBIND environment variables
    - IEWBIND_DIAG
        - to catch a message
    - IEWBIND_OPTIONS
        - from regular API only, no *known* users

Deeper dive into binder processing

- All about AUTOCALL
- A closer look at the logical structure

All about AUTOCALL

- SYSLIB ddname
- AUTOCALL control statements
- LIBRARY control statements

- PDSs and PDSEs
  - “C370LIB” Object Libraries
- UNIX archive files

- Traditionally
  - The unresolved symbol name is searched for as the member name
    - The expectation is that the member, if found, will contain the symbol

- Object Libraries and UNIX archives extend this
  - They have their own directories of defined symbol names

All about AUTOCALL …

- CALL (default) or NCAL
  - CALL=YES or NOCALL or CALL=NO

- SYSLIB ddname
  - Concatenation of data sets
    - All kinds – object modules, load modules, program objects

  - Searched for only after reading all SYSLIN input

All about AUTOCALL …

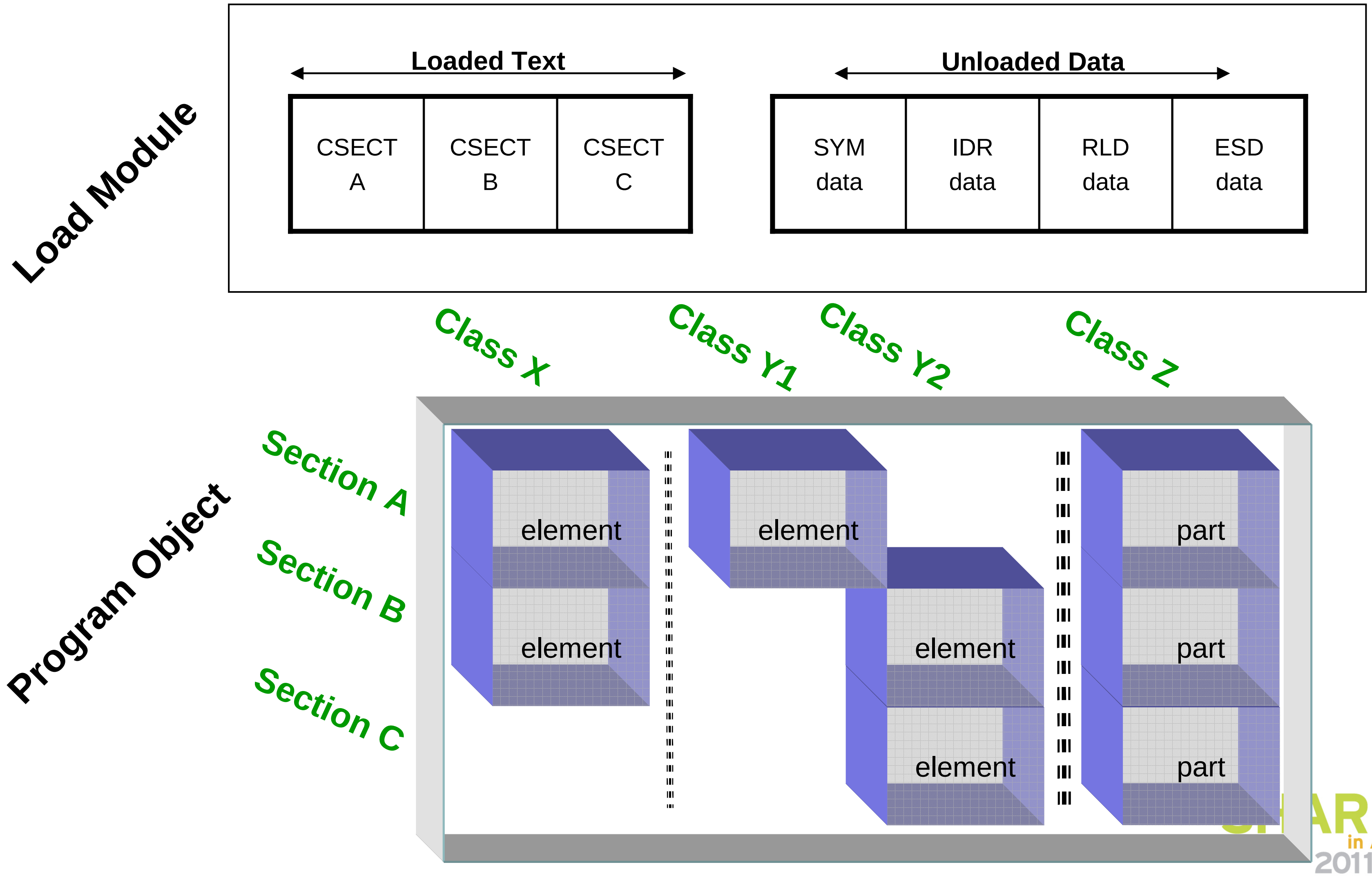
- AUTOCALL

  - UNIX “incremental”
  - Use this library right now
  - Then, forget about it!

- LIBRARY

  - Qualified with member/symbol for:
    - additional CALL (like INCLUDE but only if unresolved)
    - restricted NO-CALL
    - NEVER-CALL
  - Unqualified support added for UNIX final autocall
    - couldn’t do it with SYSLIB concatenation
  - Searched in order just before SYSLIB

load module vs. program object



load module vs. program object …

- Load modules
  - Single-dimensional
  - Documented format
  - Format never to be (substantially) be changed

- Program objects
  - Multi-dimensional
    - Class vs. section
  - Format never to be documented
    - Changes regularly – COMPAT(PMnn) levels and zOSVnRn sublevels
      - *Currently 8 levels and sublevels*

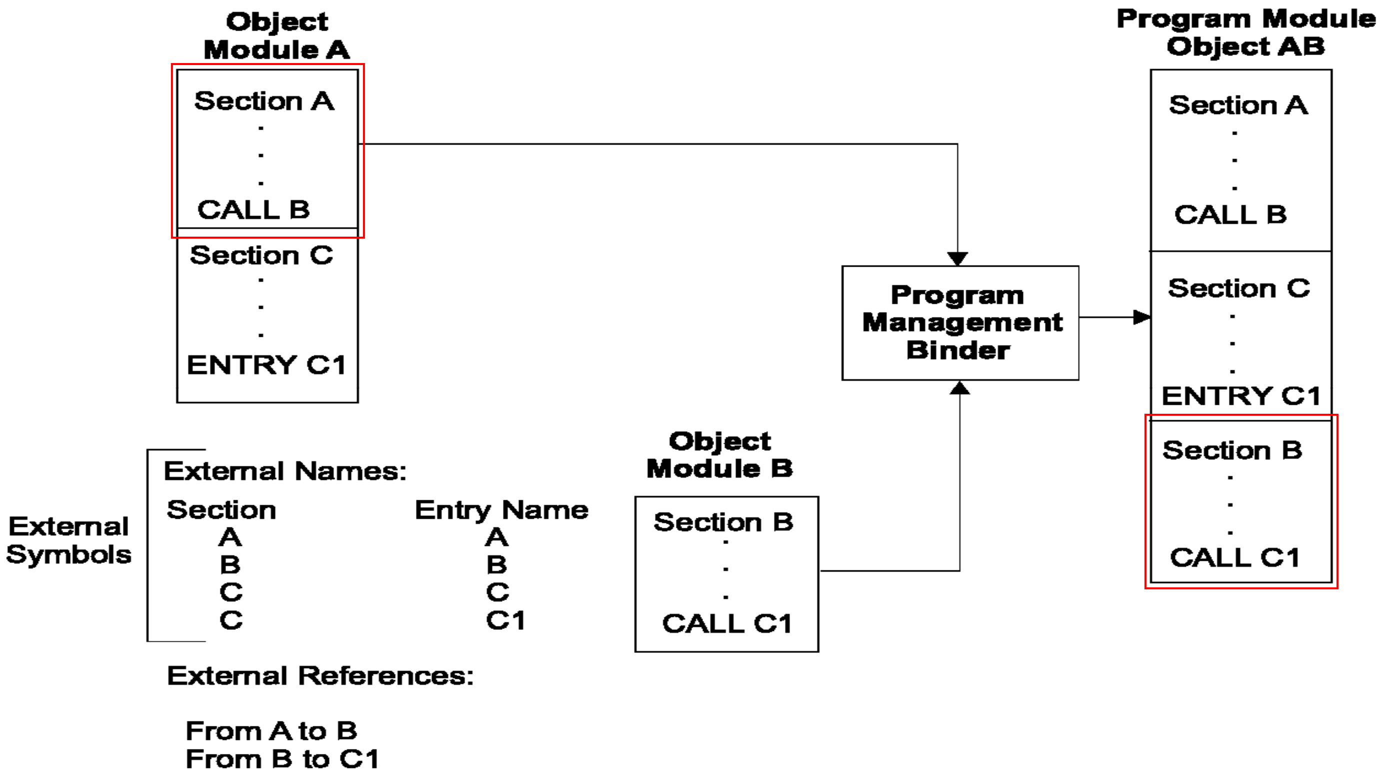
Problem diagnosis

- A little more of what goes on inside
- Understanding the outputs
  - For when that program won't bind
  - For when the program needs to be debugged
- Common problems and helpful tips

program management binder …

symbol resolution

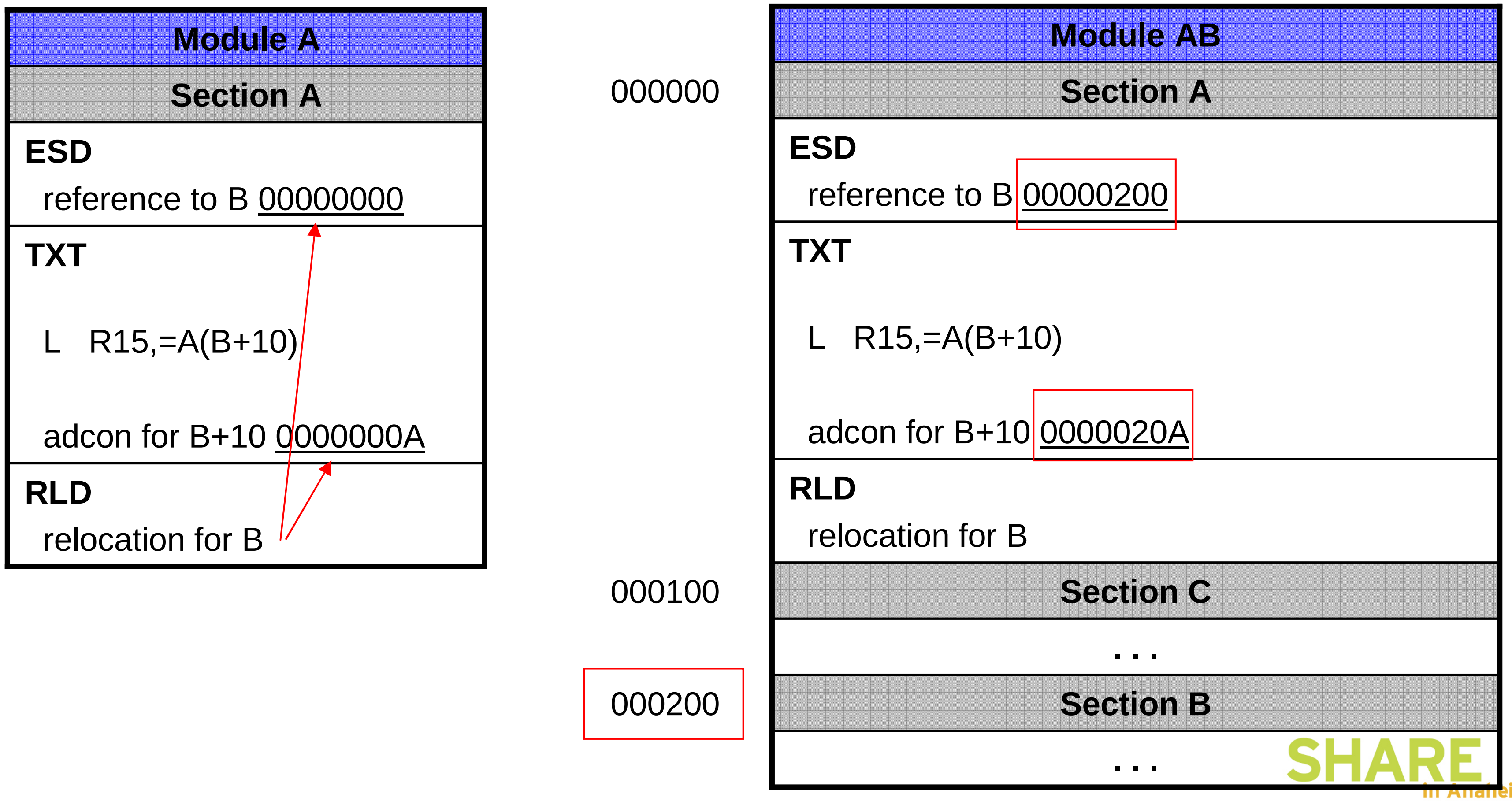
- In Section A there is a call (reference) to B which will be statically linked to A
- Location of B relative to the call in Section A is determined at bind time
- Final relocation of entire executable program module determined at load time



program management binder …

relocation

- There is an External Symbol Dictionary (ESD) entry for the location of B
- There is an Relocation Dictionary (RLD) entry for the location in A to write the location of B
- What if B is unresolved ?



SYSPRINT details

- SYSPRINT
  - Messages (IEW2nnnns) also *SYSTERM*
  - DDname cross-reference
  - Message Summary

  - **LIST**ing of processing information
  - Module **MAP**
    - Includes Data Set Summary
  - Cross(**X**) **REF**erence between symbol definitions and references
    - includes DLL IMPORT/EXPORT table

SYSPRINT details …

- SYSPRINT extras; requires **MAP** or **XREF**
  - **Renamed symbol cross-reference**
    - Usually only for special predefined list of C symbol names
    - Also RENAME control statement
  - **Long symbol abbreviation table**
  - **Short Mangled Name report**
  - **Symbol References Not Associated with any AdCon**
    - “Dangling” External References
    - Also produced with **LIST**
    - Heading may be there even if no symbols
    - Due to external reference ESD entry from object module

MAP

```
*** M O D U L E   M A P ***

 ---------------
 CLASS  C_CODE             LENGTH =      160  ATTRIBUTES = CAT,   LOAD, RMODE=ANY
                           OFFSET =        0 IN SEGMENT 001       ALIGN = DBLWORD
 ---------------

  SECTION    CLASS                                      ------- SOURCE --------
   OFFSET   OFFSET  NAME                TYPE    LENGTH  DDNAME   SEQ  MEMBER

                 0  CEESTART            CSECT       7C  /0000001  01
         0       0     CEESTART            LABEL

                80  this_is_a-g_name    CSECT       E0  /0000001  01
         0      80     this_is_a-g_name    LABEL
        28      A8     main                LABEL
```

Class name and attributes

CLASS binding attribute

CLASS loading behavior

SEGMENT containing CLASS

Offset of LABEL **main** within section (CSECT) **this_is_a-g_name**

Offset of LABEL **main** within CLASS **C_CODE**

MAP …

```
*** M O D U L E  M A P ***

 ---------------
 CLASS  C_WSA       LENGTH =        24  ATTRIBUTES = MRG, DEFER , RMODE=ANY
                    OFFSET =         0 IN SEGMENT 002       ALIGN = DBLWORD
 ---------------

            CLASS
           OFFSET  NAME                TYPE     LENGTH   SECTION
                0  $PRIV000011         PART         10
               10  hw#S                PART         10   this_is_a-g_name
               20  world               PART          4   world
```

Class name and attributes

binder made-up name for "private" symbol

CLASS binding attribute

CLASS loading behavior

SEGMENT containing CLASS

Offset of PART **world** within CLASS **C_WSA**

XREF

CROSS-REFERENCE TABLE

All address constants in section CEESTART in CLASS C_CODE

Location to which adcons in section CEESTART have resolved

```
TEXT CLASS = C_CODE

---------------  R E F E R E N C E  --------------------------  T A R G E T  -------------------------------------------
  CLASS                              ELEMENT        |                                                 ELEMENT              |
  OFFSET SECT/PART(ABBREV)            OFFSET   TYPE | SYMBOL(ABBREV)   SECTION (ABBREV)                OFFSET CLASS NAME   |
                                                    |                                                                      |
     2C  CEESTART                         2C V-CON  | CEEMAIN          CEEMAIN                             0 C_DATA        |
     68  CEESTART                         68 V-CON  | CEEFMAIN         $UNRESOLVED                                         |
     6C  CEESTART                         6C V-CON  | CEEBLLST         CEEBLLST                            0 B_TEXT        |
     74  CEESTART                         74 V-CON  | CEEBETBL         CEEBETBL                            0 B_TEXT        |
     78  CEESTART                         78 V-CON  | CEEROOTD         CEEROOTA                            0 B_TEXT        |
    14C  this_is_a-g_name                 CC A-CON  | CEESTART         CEESTART                            0 C_CODE        |
```

We can see that section CEESTART begins CLASS C_CODE

XREF …

Symbol world is a part… we know from the Module MAP…

Adcon at X'1C' in section hw#S refers to IMPORTED symbol printf. Location of printf not known until run-time.

```
                                  C R O S S - R E F E R E N C E   T A B L E

TEXT CLASS = C_WSA

---------------  R E F E R E N C E  --------------------------  T A R G E T  ----------------------------------------
  CLASS                               ELEMENT       |                                                ELEMENT          |
  OFFSET  SECT/PART(ABBREV)            OFFSET   TYPE | SYMBOL(ABBREV)   SECTION (ABBREV)               OFFSET CLASS NAME |
                                                     |                                                                  |
      10  hw#S                             10 A-CON | world            $PRIV000003                       20 C_WSA       |
      18  hw#S                             18 R-CON | printf                                                             |
      1C  hw#S                             1C V-CON | printf           $IMPORTED                                         |
      20  world                            20 A-CON | this_is_a-g_name this_is_a-g_name                   0 C_CODE      |
      18  hw#S                             18 A-CON |                                                      B_IMPEXP    |
      1C  hw#S                             1C V-CON | CEETHLOC         CEETLOCE                            8 B_TEXT      |
```

common problems & helpful tips

- Mixed-case input

IEW2456E 9207 SYMBOL myfunc UNRESOLVED. MEMBER COULD NOT BE INCLUDED FROM THE DESIGNATED CALL LIBRARY.

  - Traditional names (from OBJ) are uppercase
    - Compatibility dictates the default CASE=UPPER
  - Affects options values and control statement symbols
    - Option names and control statement keywords are case insensitive
      - *INCLUDE, include, Include*
  - Most often an issue for IMPORT control statements (DLLs)
  - Recommendations
    - CASE=MIXED
      - *Import Code,a.dll,myFunc*
    - ‘quote_name’
      - *INCLUDE ‘/u/barryl/C/hello.o’*
      - *include PDSELIB(‘hello’)*

common problems & helpful tips …

- Long symbol names
  - Member names (at least in PDSs) are 8 characters
  - Problem introduced when building the object modules…
    - C/C++ LONGNAME option required
  - … or when creating the (object) libraries
    - UNIX archive libraries manage their own internal directory
    - data set based (“C370LIB”) Object Libraries have a special directory member
      - *@@DC370$, @@DC390$*

common problems & helpful tips …

- Long symbol names …

  IEW2459W 9206 INCLUDED **MEMBER s1** FAILED TO RESOLVE REFERENCE.

  IEW2497W 9229 THE **SYMBOL s1** WAS EXPECTED TO BE RESOLVED BY INCLUDING **MEMBER SUB3** FROM THE LIBRARY DEFINED BY **DDNAME C8961**

  - Worst case scenario!
    - Replacement object module incorrectly built (perhaps NOLONGNAME)
    - Directory member was previous built and not updated
  - IEW2497W is new for R12
  - Module already included, may resolve other symbols!

common problems & helpful tips …

- Unresolved but it's there?
  - DYNAM=DLL may be required!
    - If "definition" is on IMPORT statement
      - *Otherwise binder processes IMPORTs but silently ignores them*

common problems & helpful tips …

- Where did *that* thing come from?
  - Modules brought in by autocall
  - Turning on LIST=ALL
  - Introduced in z/OS V1R12 !

```
IEW2340I 1036 MEMBER NAME CEEROOTD IN THE LIBRARY DEFINED BY
DDNAME SYSLIB IS BEING INCLUDED TO RESOLVE REFERENCE TO CEEROOTD

IEW2308I 1112 SECTION CEEROOTA HAS BEEN MERGED.
```

  - Especially for archives & C370LIBs

common problems & helpful tips …

- for situations where options cannot otherwise be passed
  - particularly API based programs
    - IEWPARMS
      - *like OPTIONS*
    - IEWDIAG
      - *like SYSTERM with LIST=ALL, MSGLEVEL=0*
        - *useful if you are unable to pass those options*

common problems & helpful tips …

- AMBLIST
  - LISTOBJ – all object modules
  - LISTIDR – all identification records; user IDENTIFY, language, binder, zap (EDIT=YES required)

common problems & helpful tips …

- AMBLIST …
  - LISTLOAD – all program modules (EDIT=YES required!)
    - Like binder MAP and XREF and more!
    - PMAR (partially) decoded and (fully) dumped
    - MODLIST
      - *Section / Class information …*
      - *… including TEXT*
        - *Merge class part initializers decoded*
        - *IEWBCIE / B_IMPEXP decoded*
    - MAP
      - *SEGMENT map*
      - *Numerical MAP*
    - XREF
      - *SEGMENT map*
      - *Numerical MAP and XREF*
      - *Alphabetical MAP and XREF*
    - AMBLIST LISTLOAD ebcdic translation for load modules -- Introduced in z/OS V1R12 !

More advanced stuff

- It’s truly not the linkage editor !
- Diagnostic DD’s
- EXITs
- APIs

It’s truly not the linkage editor !

- Really not the linkage editor!
  - Application programming interface
  - DLLs, XPLINK
  - Classes (INIT load, NO load and DEFER load)
  - …

- PDSE, UNIX
  - program object format – PO (COMPAT(PMx))
  - exclusively binder
  - loaded by program management loader

- PDS
  - load module format
  - just like the linkage editor used to do
    - HEWLKED anybody?
  - loaded by program management loader (program fetch)

Diagnostic DD's

- IEWTRACE ddname – TRACE option
  - binder internal trace table
  - shows function entry / exit and other key processing points
  - shows ECODEs (part of which is 4 character code after message number)
  - can filter entries with TRACE=(start,end) or selectively TRACE='c[c…]'
- IEWDUMP – DUMP option
  - if allocated, automatically written upon terminal binder error or program check or abend
  - can be forced with DUMP option specifying ecode
    - binder continues processing for non-terminating condition
  - binder takes SNAP of binder storage and then formats key internal structures
- <u>note:</u> these diagnostics are normally used only for IBM problem determination
- limited information provided in program management documentation

Exits

- User exits – EXIT option
  - provide module exit name
  - MESSAGE
    - filter all messages of specified severity or higher
    - prevent or allow the message to print
    - no effect on final return code of binder
  - SAVE
    - notification of each primary (member) name and alias name to be saved
    - request retry for certain failures
  - INTFVAL (Interface Validation)
    - after all input processing, including autocall
    - examine all references (resolved and unresolved) for each section
    - can allow unresolved, can change resolution to another symbol or glue
    - default processing can result in error if target & reference disagree in
      1. *ESD signature fields*
      2. *XPLINK attributes*
      3. *AMODE(64) mismatch*
      4. *Namespaces (like code (instructions) vs. data)*
      5. *Certain class attributes (like catenate vs. merge)*

APIs

- Application Programming Interfaces (APIs)
  - data is input or output via buffers unique to each type of data
    - for example, ESDs
    - IEWBUFF macro can simplify creating buffers
      - *allocate, initialize, map and delete buffers*
      - *not required*
  - regular binder APIs
    - IEWBIND macro
      - *not required*
  - fast data access
    - for program objects only
      - *faster due to direct access, bypass workmod conversion*
    - request code interface
      - *obsoleted IEWBFDA macro "unitary" interface*
  - C APIs
    - NOXPLINK and XPLINK -- introduced in z/OS V1R12 !
    - buffers in a header, C language oriented structures
    - simplifies access by automatically managing buffers for you
    - both regular API and fast data access functions provided

program management documentation

- SA22-7643 - z/OS MVS Program Management: User's Guide and Reference
  for options & control statements
- SA22-7644 - z/OS MVS Program Management: Advanced Facilities
  for binder **APIs**
- GA22-7589 - z/OS MVS Diagnosis: Tools and Service Aids
  for **AMBLIST** and **SPZAP**
- SA22-7782 - z/OS TSO/E Command Reference
  for **LINK** and **LOADGO**
- SA22-7802 - z/OS UNIX System Services Command Reference
  for **c89** and **ld**