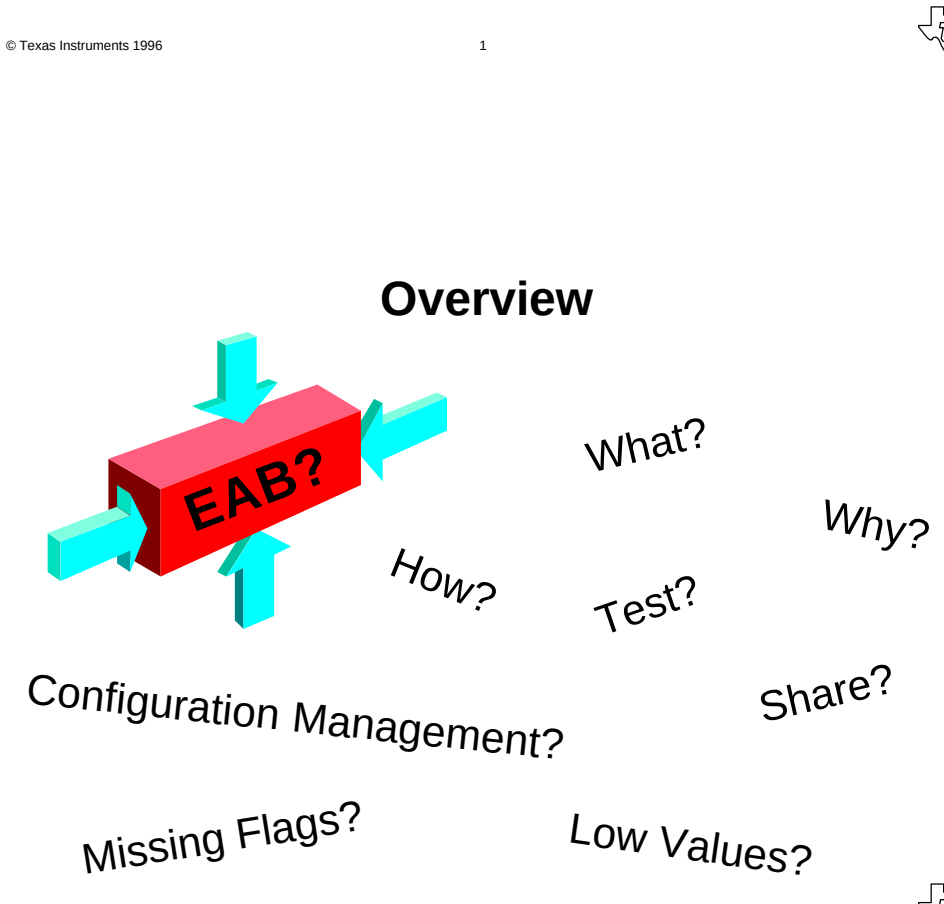
Building
External Action Blocks

Session 580

Mary Russell
Texas Instruments



Overview

What?

Why?

How?

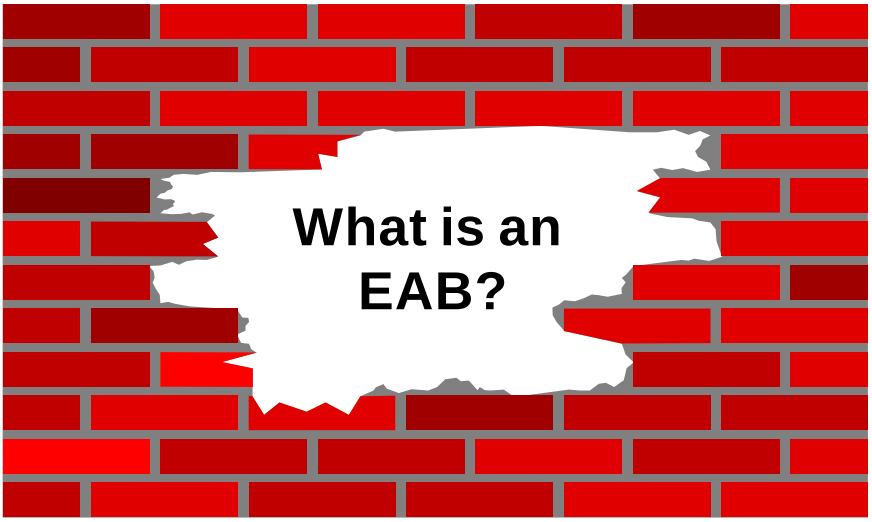
Test?

Configuration Management?

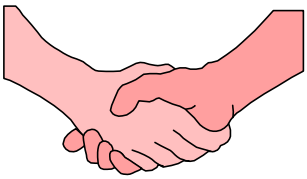
Share?

Missing Flags?

Low Values?

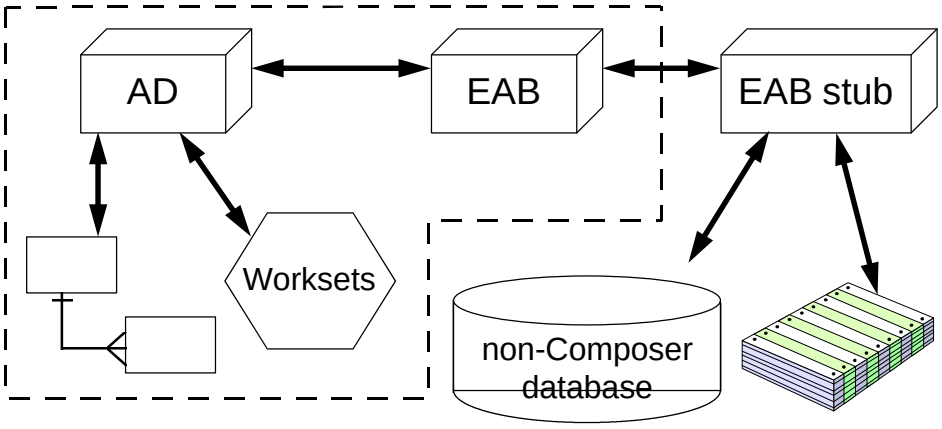


What is an External Action Block (EAB)?

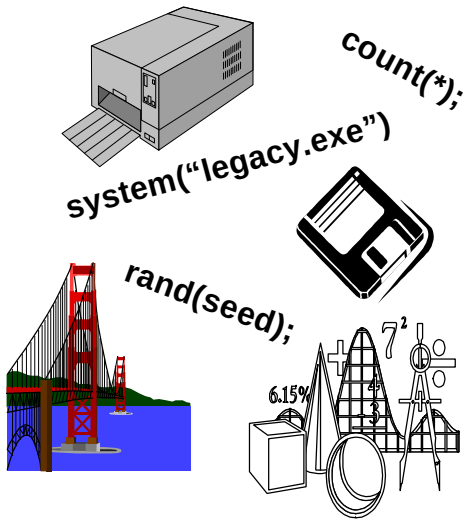


- Composer's *handshake* with the rest of the world
- EABs are Composer's communication mechanism with non-Composer applications
- EABs can also access non-Composer data stores

What is the EAB's Role?

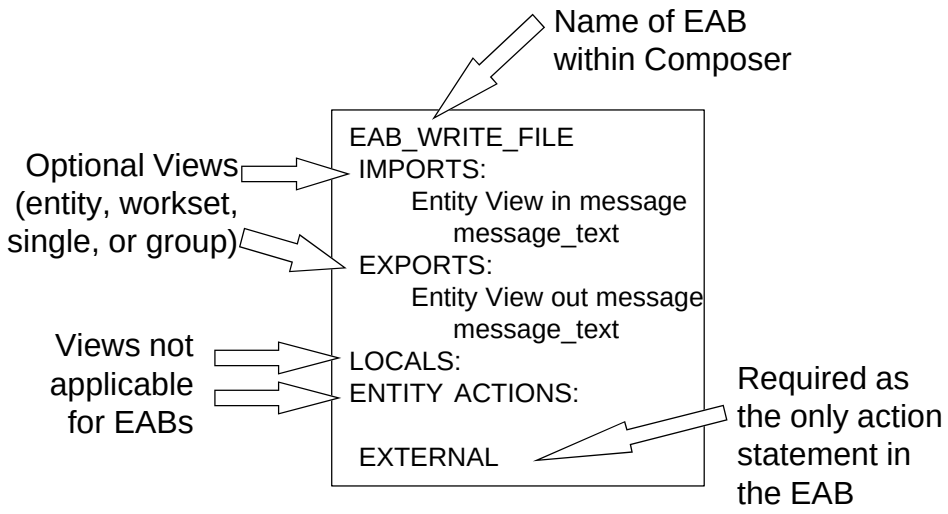


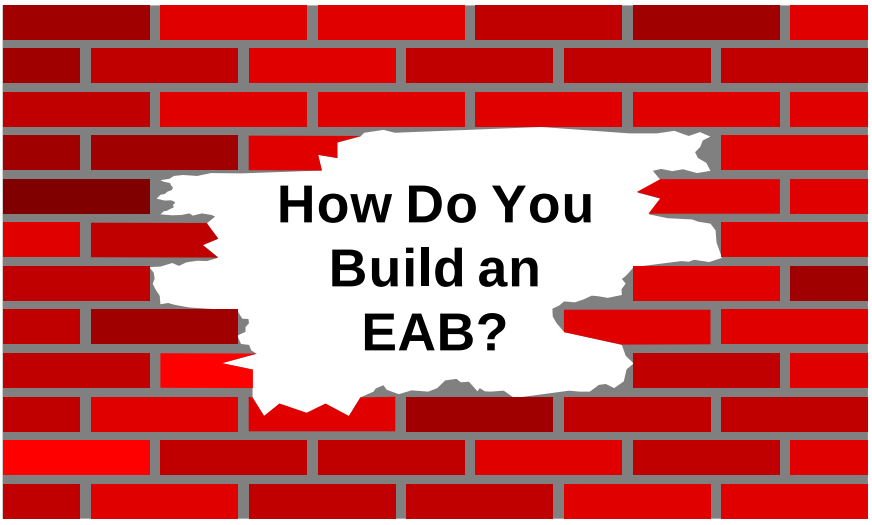
Why Would You Use an EAB?



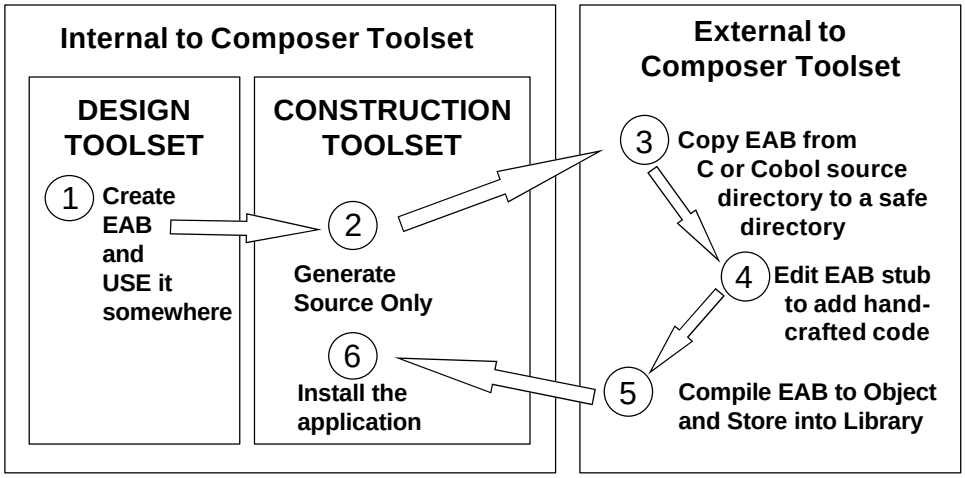
- Bridge to legacy systems
- Read and write files
- Construct and print reports
- Perform database functions
- Perform native language functions
- Call multi-media files
- Call non-Composer applications/desktop tools

What is the EAB Anatomy?

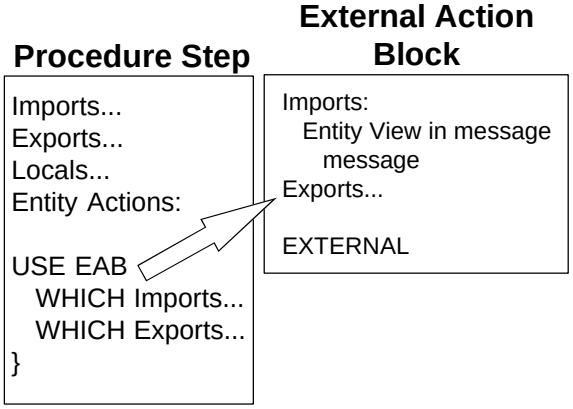




How Do You Build an EAB?



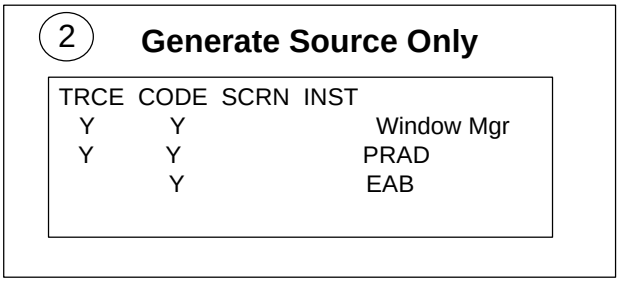
Where Do I Create My EAB?



1

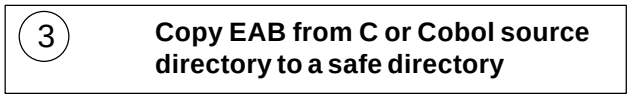
- Use the *Design* Toolset to create an EAB the same way a common action block is created
- Build the *views* as appropriate and insert the *EXTERNAL* action statement
- *USE* the EAB in an action diagram

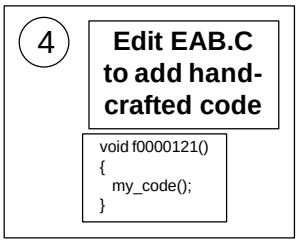
Why Generate Source Only?



- Must allow Composer to generate the source code for the EAB stub before it can be modified
- Can double-click on the EAB name here to see and optionally change the source name
- TRACE does not apply for EABs; has no affect when turned on

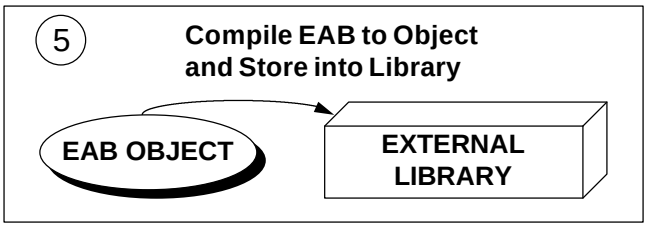
Why Should I Move the EAB?





- Locate EAB source code in appropriate directory (e.g., /c or /cobol)
- Move it to a safe directory before editing it; this prevents accidental overlays of user-written code if EAB stub is regenerated
- If embedding SQL, may change extension to ***.sqc*** before modifying stub

Why Compile and Store in Library?

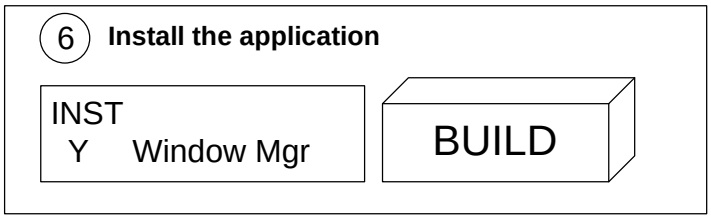


- Must compile EAB stub to an object manually
- Then must store it in a *Composer-recognized* library
- Storing object in library enables Composer to find it during the link process of the install

What About Samples of Syntax?

Step	OS/2	Win 3.1	UNIX
Precompile	sqlprep [pgm] [dbname] sqlprep eab.sqc iefdb	(varies for each database)	(varies for each database)
Compile	icc /c [pgm.c] icc /c eab.c	(use Microsoft Programmer's Workbench)	cc -c [pgm.c] cc -c eab.c
Library Storage	lib [libname] [ops] [obj] lib extrnc -+ eab	(use Microsoft Programmer's Workbench)	ar [opts] [lib] [obj] ar -rv extrnc.a eab.o

What Does the Install Do?

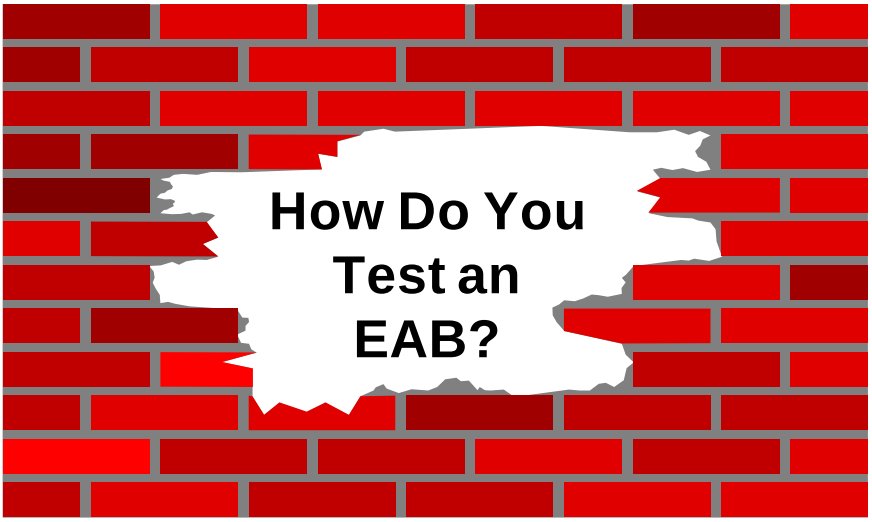


- The Composer install process can be run in the Toolset or the Build Tool
- The Install process creates a makefile that compiles all Composer source code into objects, links all Composer and external objects into executables, and optionally binds to the specified database
- The install process will resolve all externals (EABs) in the default libraries, or other libraries if specified

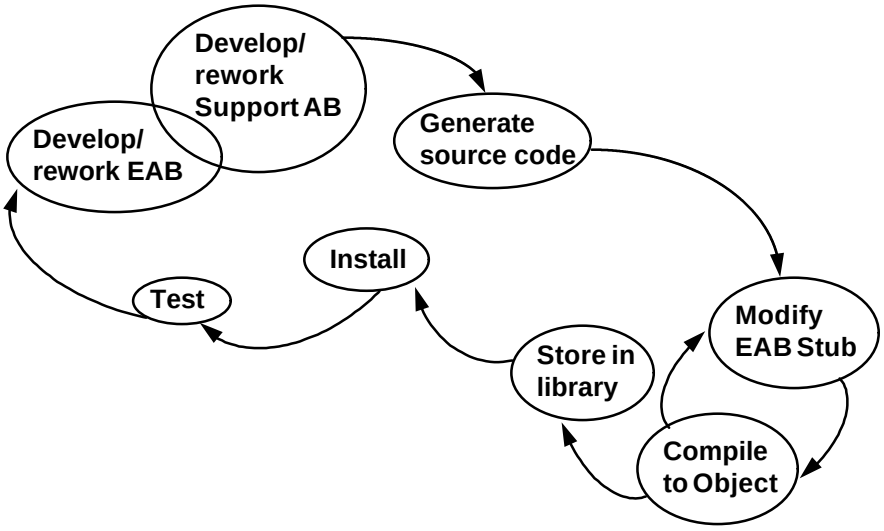
What Do EAB Stubs Look Like?

```
/*************************************************************
 * *              Source Code Generated by
 *                  Composer (tm)
 *           Texas Instruments Incorporated
 *   Copyright (c) Texas Instruments Incorporated 1996
 *
 *   Name: EAB_WRITE                Date: 96/02/03
 *   Target OS:  OS2                Time: 18:06:00
 *   Target DBMS: DB2/2             User: OS2USER
 *
 *   Generation options:
 *************************************************************/
/* * * * * * * * * * * * * * * * * * * * */
/* START OF EXPORT VIEWS */
/* * * * * * * * * * * * * * * * * * * * */
struct w_ioa_000
{
/* Entity View: OUT */
/*       Type: IEF_SUPPLIED */
struct
{ char command_001[81] /* 80 + 1 */;
} ev001;
};
static struct w_ioa_000 *a_0006815910_ioa;
static void f_65557(void);
static char func_0000065557_esc_flag;
```

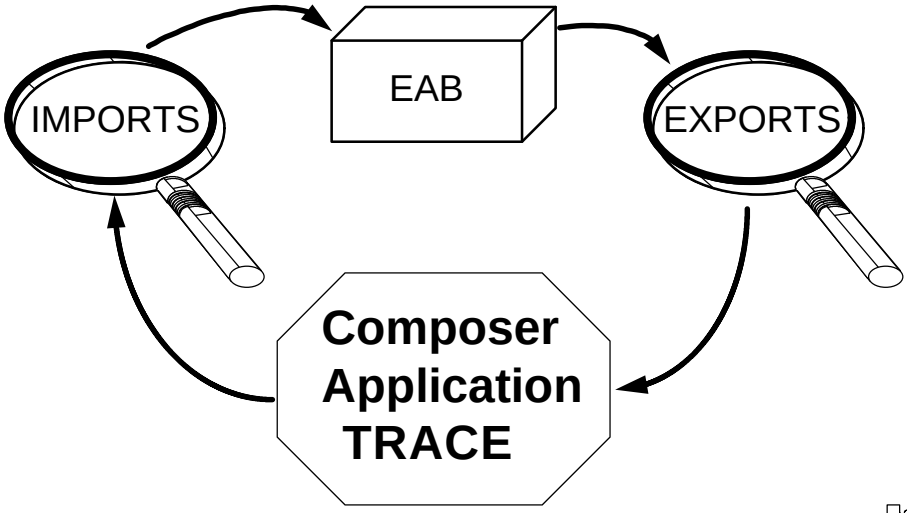
```
/*  +-> EAB1                          02/03/96  18:06      */
/*  |   EXPORTS:                                           */
/*  |    Work View out ief_supplied  (Tra... */
/*  |        command                                       */
/*  |   EXTERNAL ACTION BLOCK                              */
/*  +---                                                   */

void EAB1(in_runtime_parm1,in_runtime_parm2,in_globdata,
ex_v_000)
char *in_runtime_parm1; char *in_runtime_parm2;
struct ief_globdata *in_globdata; struct w_ioa_000 *ex_v_000;
{
ief_runtime_parm1 = in_runtime_parm1;
ief_runtime_parm2 = in_runtime_parm2; globdata = in_globdata;
a_0006815910_ioa = ex_v_000;
f_65557();
return;
}

static void f_65557(void)
{ func_0000065557_esc_flag = '\0';
/* User-written code should be inserted here */
}
```



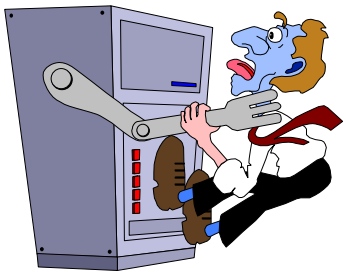
What is the EAB's Testing Cycle?



How Does Trace Treat an EAB?

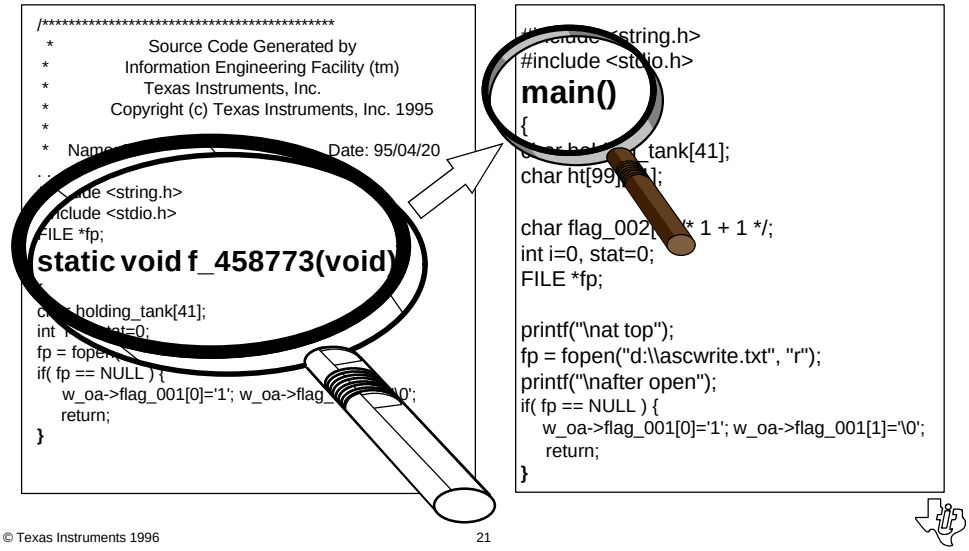


How Do You Debug an EAB?



- Cannot use Composer Trace Facility
- Can embed debugging commands such as *printf* or *DISPLAY*
- Can use compiler-supplied test facilities such as *PWB* or *Code View*
- Difficult to test a complex EAB unless you make it standalone
- *Common Practice:* use EAB as a stub that calls hand-crafted code

How Can I Make it Standalone?



What Happens When You Add More Attributes or Views?

IMPORTS:
Entity View in message
message
name
ssn

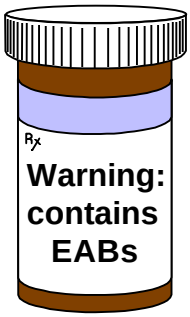
```
struct
{
char message_001[26];
char name_002[21];
}ev001;
```

IMPORTS:
Entity View in message
employee_code
message
name
ssn

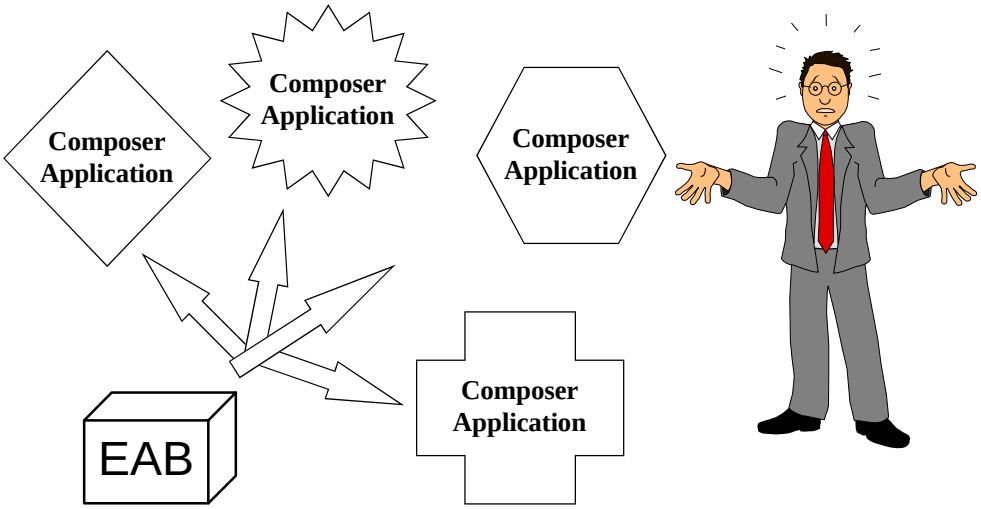
```
struct
{
int employee_code_001;
char message_002[26];
char name_003[21];
}ev001;
```

✷ Add new attributes or views at bottom of existing views

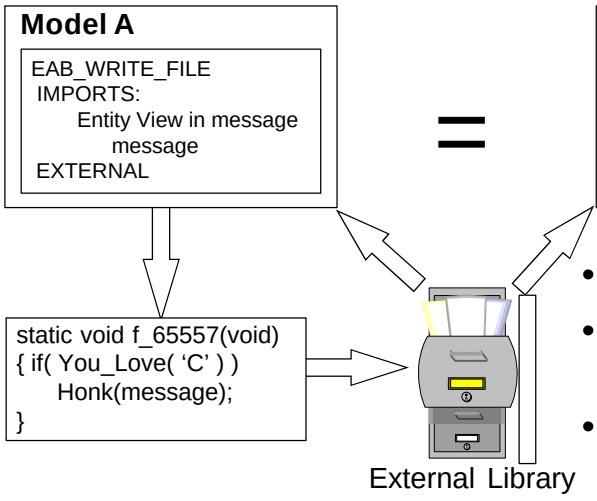
Who Should Build the EABs?



Are EABs Shareable?

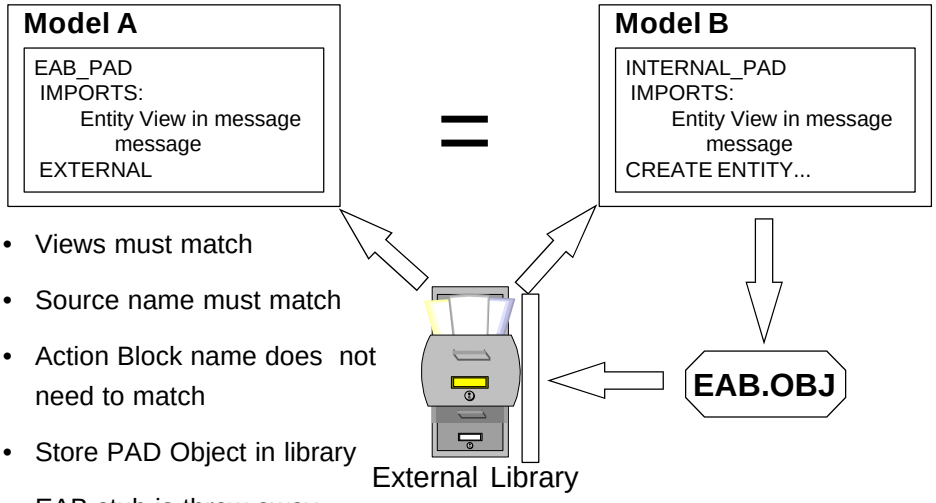


How Do You Share an EAB?

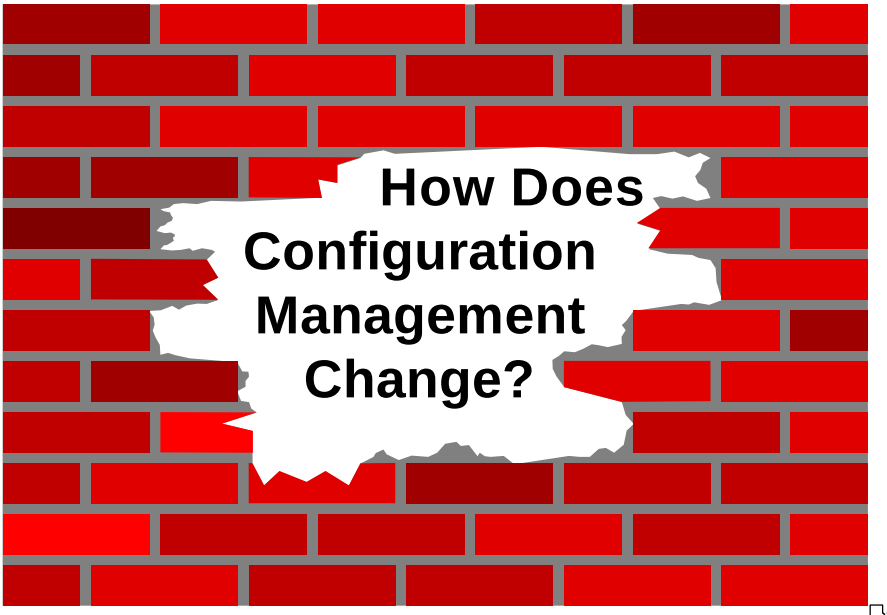


- Views must match
- Source name must match
- Action Block name does not need to match

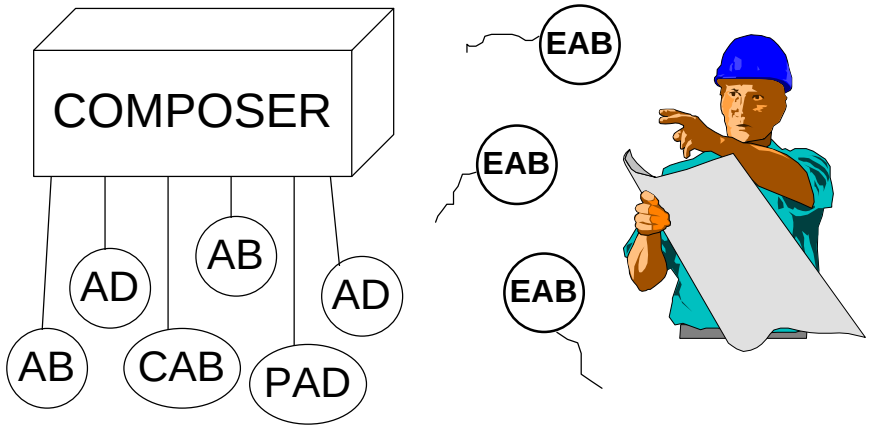
How Do You Share Action Blocks?



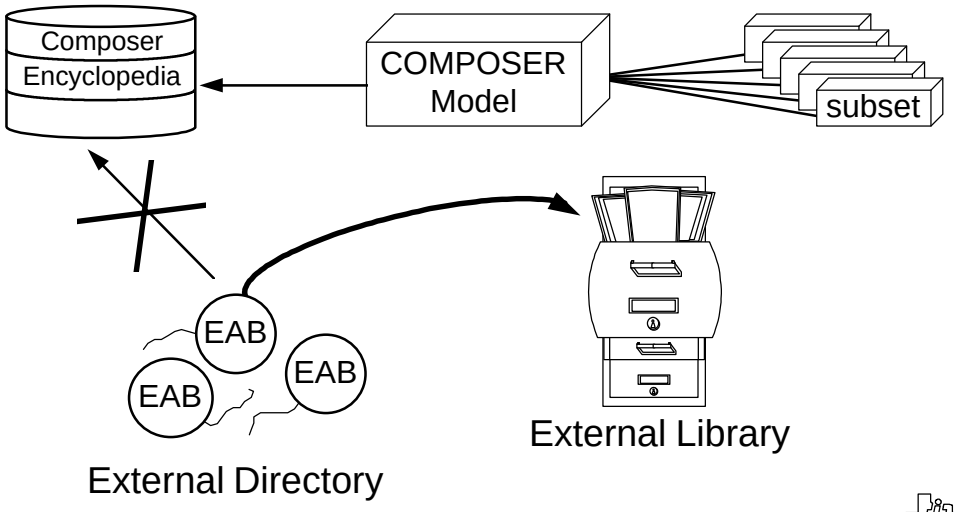
- Views must match
- Source name must match
- Action Block name does not need to match
- Store PAD Object in library
- EAB stub is throw away



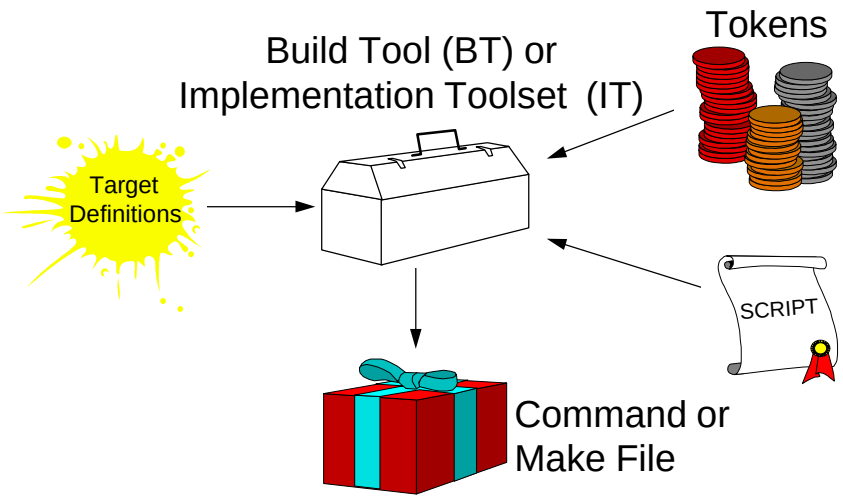
What are External Objects?



How Do They Affect Configuration Management?



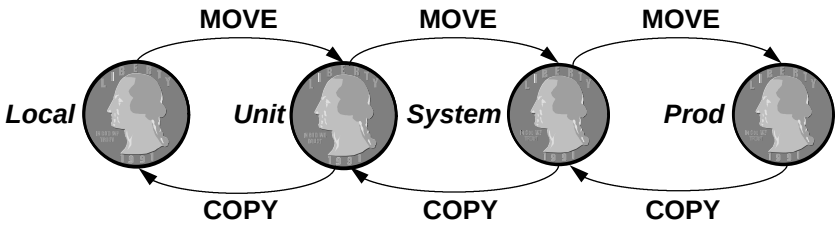
What is the Build Process?



What are Tokens?

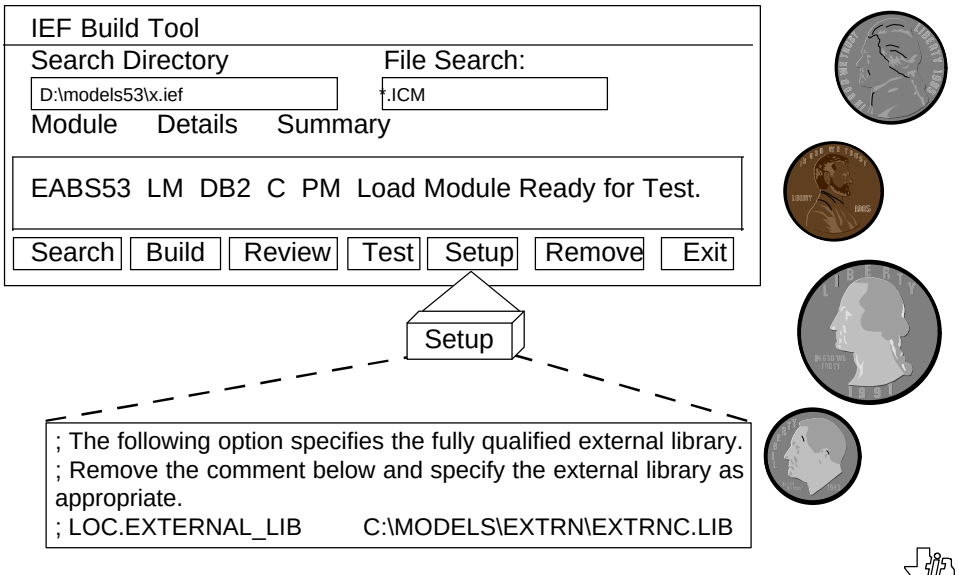
- Special variables used in Composer build process
- Assigned specific names and serve specific needs
- Appear in Composer build scripts
- Value is substituted during build process (detokenization)
- External library tokens allow variable locations and names

How Do I Apply Tokens?



- Tokens are how Composer implements external configuration management (CM)
- Refine your CM strategy to include EABs
- Tokens are specific to each platform and build mechanism (e.g., Build Tool or Implementation Toolset)

Where are the Build Tool Tokens?



Where are the Implementation Toolset Tokens?

```
IEFC1     ----- IEF Installation Tool ---
                    Main Menu
COMMAND ===>

  Select one of the options below,
then press enter.

   4  1.  Process Implementation P...
      2.  Install Load Module
      3.  Maintain Configuration Inf...
      4.  Access Utilities Menu

F01=HELP
```

```
IEFU5     ----- IEF Installation Tool -----
                  Decomposition Report
 COMMAND ===>

 Target Name: A846244_TARGET___
 Model Name:_____________________
 BSys Name:______________________
 Type  Token Name / Value Field   Seq#
 ----  -------------------------------- ----
 LOC  CODE_EXE                     001
       /users/jxn/target/codeexe/

 F01=HELP F03=EXIT F04=LIST...
```


What are Missing Flags?

IMPORTS:
Entity View in message (Transient, Optional, Import only)
readable_message
...

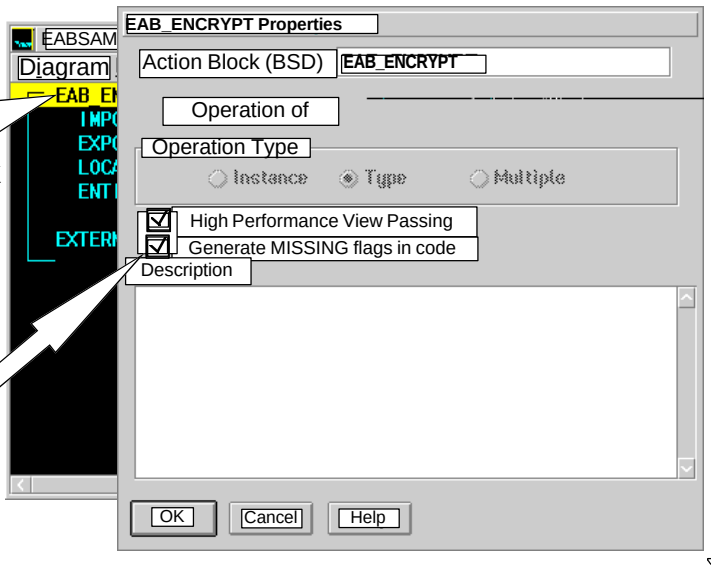
```
struct a_0007340113_ia
{
char readable_message_001as;
char readable_message_001[26];
};
```

- Additional single byte character per attribute
- Composer generates these flags to keep track of whether an attribute contains data
- Not used in EABs
- Removal of missing flags allows group moves and synchronized record layouts

How Do I Strip Missing Flags?

Double-Click on the EAB Name

Toggle Missing Flags Check Box off



What Does High Performance View Passing Mean?

☐ High Performance View Passing

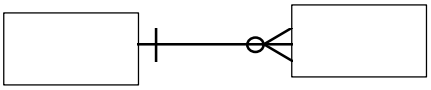
```
Struct a_00734_ia
{
char msg1_001[21];
char msg2_002[21];
}
static struct a_00734_ia *w_ia;
```

- Allows individual views to be passed
- Reference syntax changes

☑ High Performance View Passing

```
Struct w_ioa_000
{
   struct
   {
      char msg1_001[21];
   } ev001;
struct
   {
      char msg2_002[21];
   } ev002;
}
static struct w_ioa_000 *a_00734_ia;
```


What are Composer Low Values?



Only optional foreign keys are set to null by Composer

TEXT = (spaces)

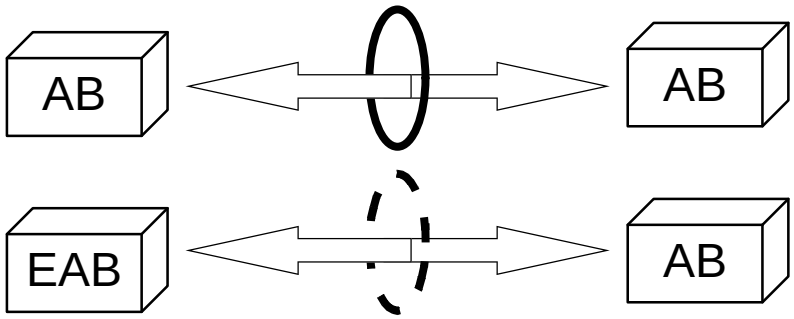
NUMBER = 0 or 0.0

TIME = 000000

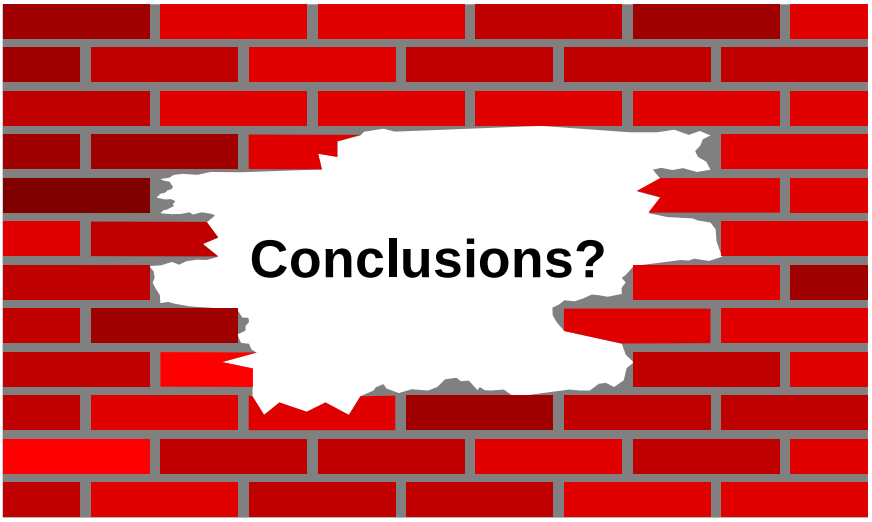
DATE = 00010101

TIMESTAMP = 00010101000000

How are Views Initialized?



- Composer protects itself against incompatible low values
- EABs views passed back to Composer must be carefully managed to avoid incompatible low values



The Facts...

- 90% of all Composer projects need to use EABs
- EABs should be built by developers experienced in the native language

- Don’t Leave EABs out of Configuration Management Strategy

Building
External Action Blocks

Session 580

Mary Russell
Texas Instruments