IBM Software Group – TXSeries for Multiplatforms

TXSeries (CICS) Recommenations

Application Programming

Iain Boyle, Software Group Services

Iain_boyle@uk.ibm.com

Version 1
19-December-2005

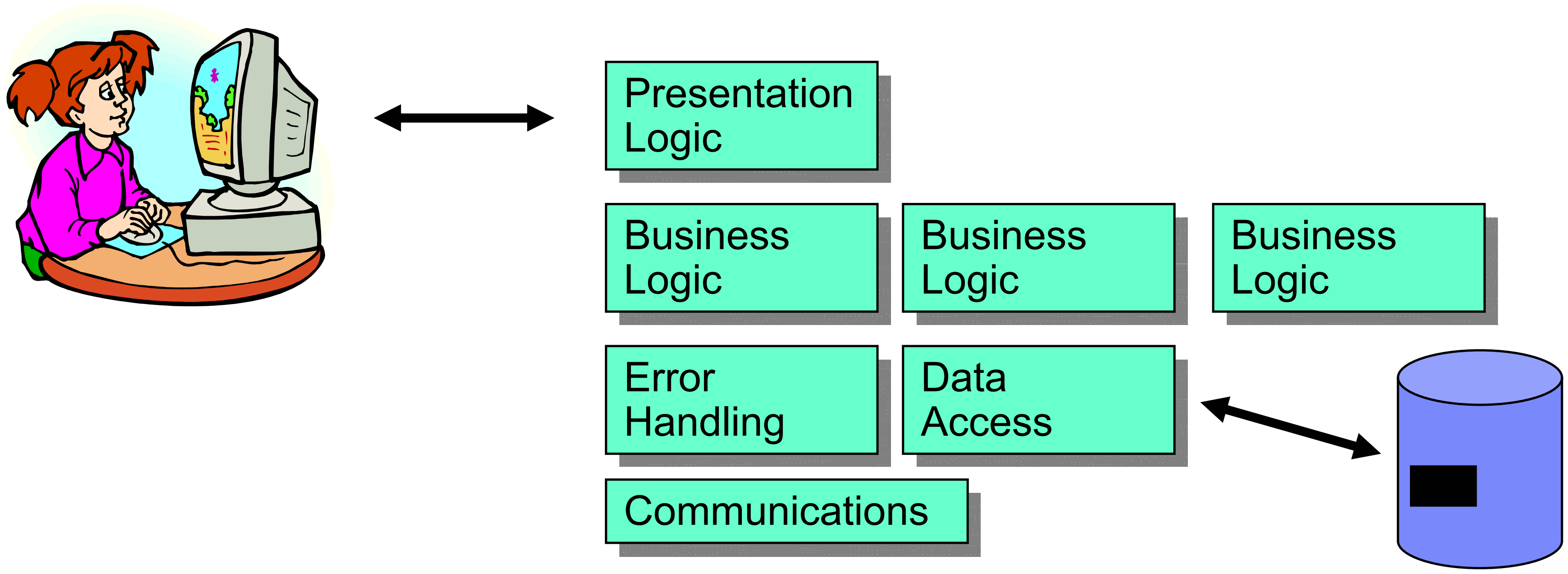
Introduction

- This presentation is a series of recommendations and guidelines from IBM on how to design and write TXSeries (CICS) based applications.

- As with all recommendations:
  - There will be exceptions
  - Good reasons to ignore the recommendation
  - Different alternatives available

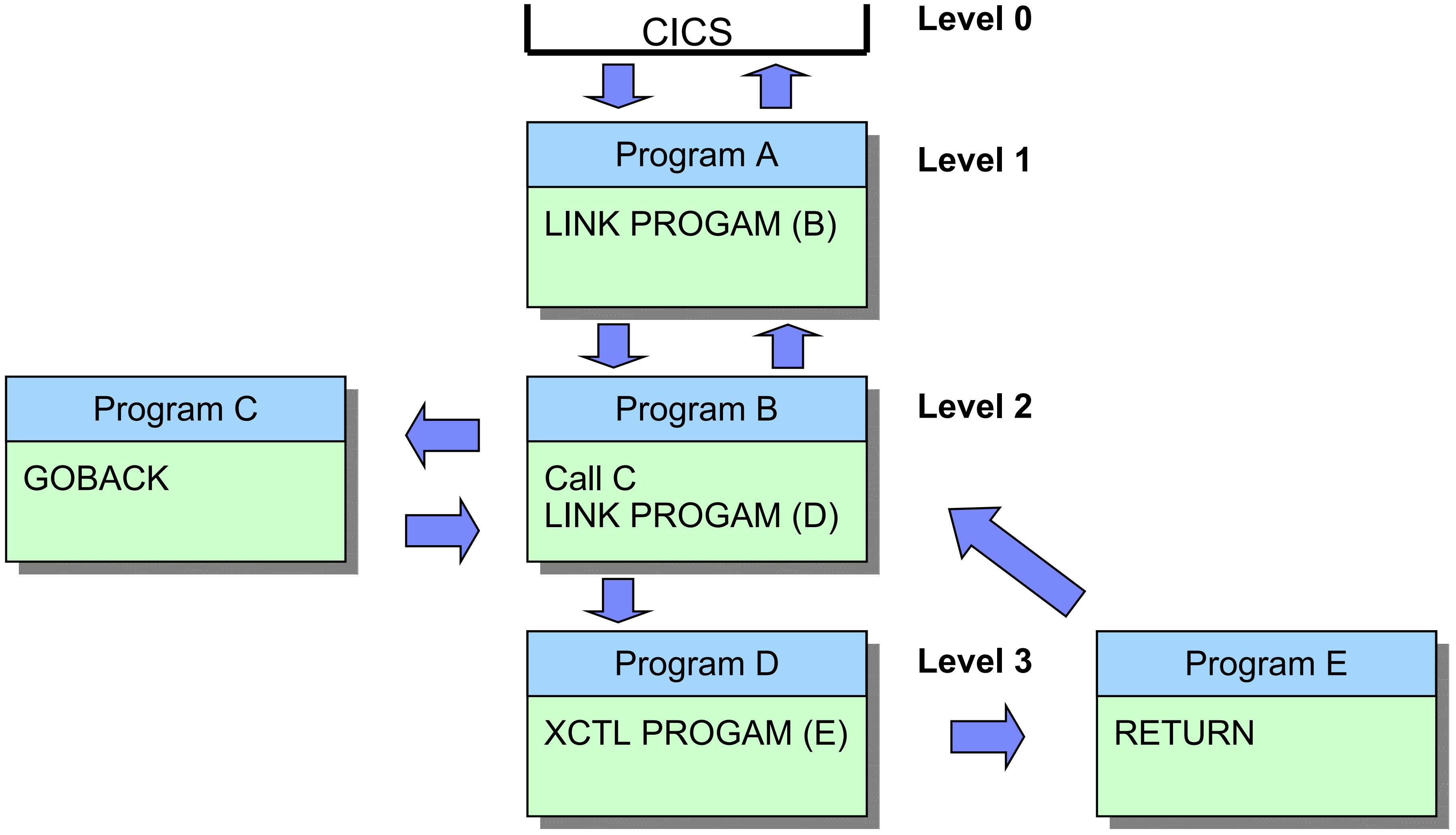
- What you should do:
  - Understand the advice
  - Consider your environment
  - Decide if the recommendation is suitable for you

Separate Program Components

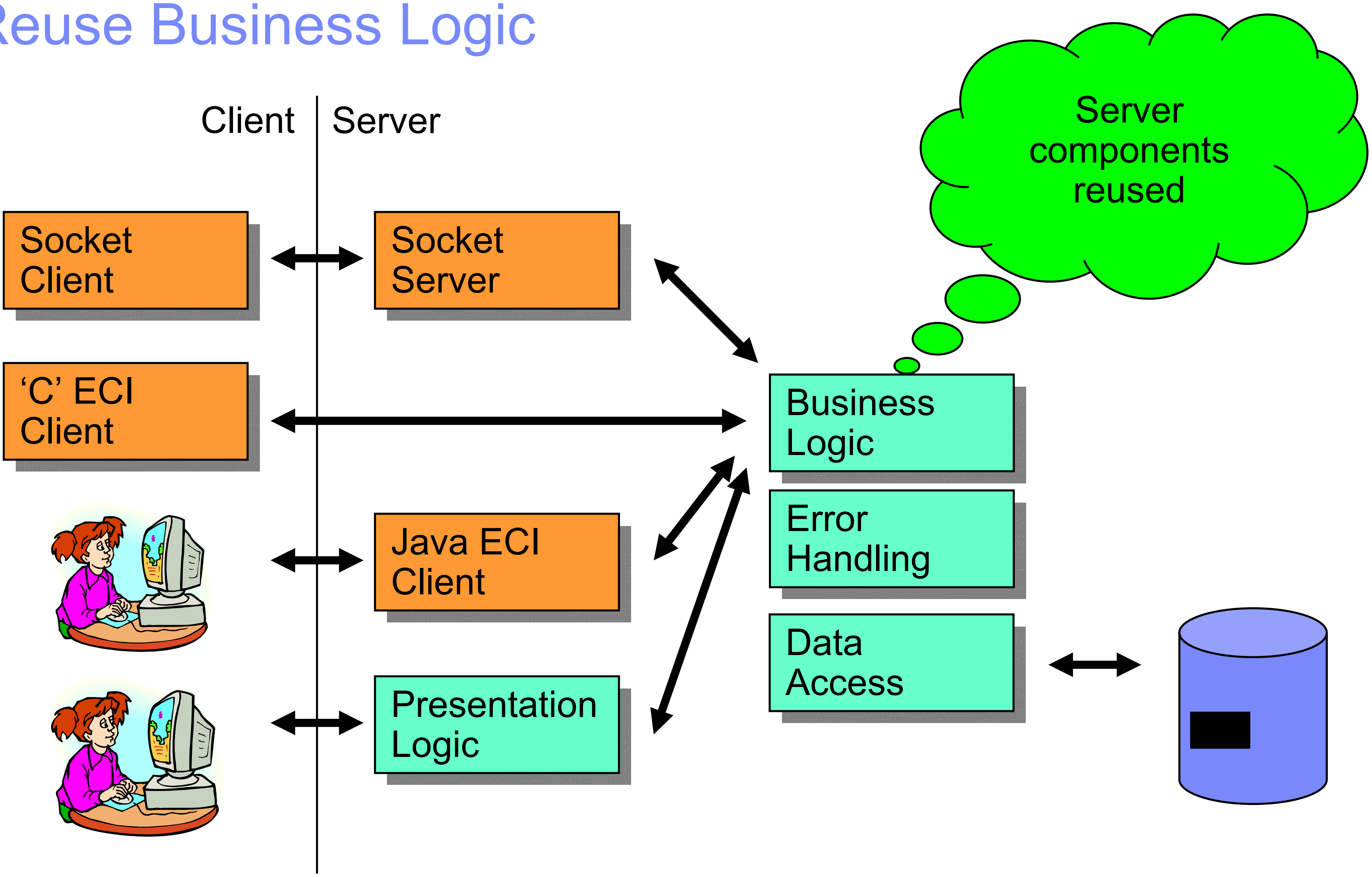


- Split program components into separate components or modules
- Sub-divide Business Logic into self contained services
- Use CICS facilities, LINK, XCTL or language calls to access modules

Separate Program Components - Logical Levels



Reuse Business Logic



Program Link Considerations

```
EXEC CICS RETURN
  TRANSID
  IMMEDIATE
```

- Use IMMEDIATE to prevent terminal ATI break ins

```
EXEC CICS LINK
  PROGRAM
  SYNCONRETURN
```

- Use SYNCONRETURN (if possible)to reduce size of LUW and additional syncpoint flow

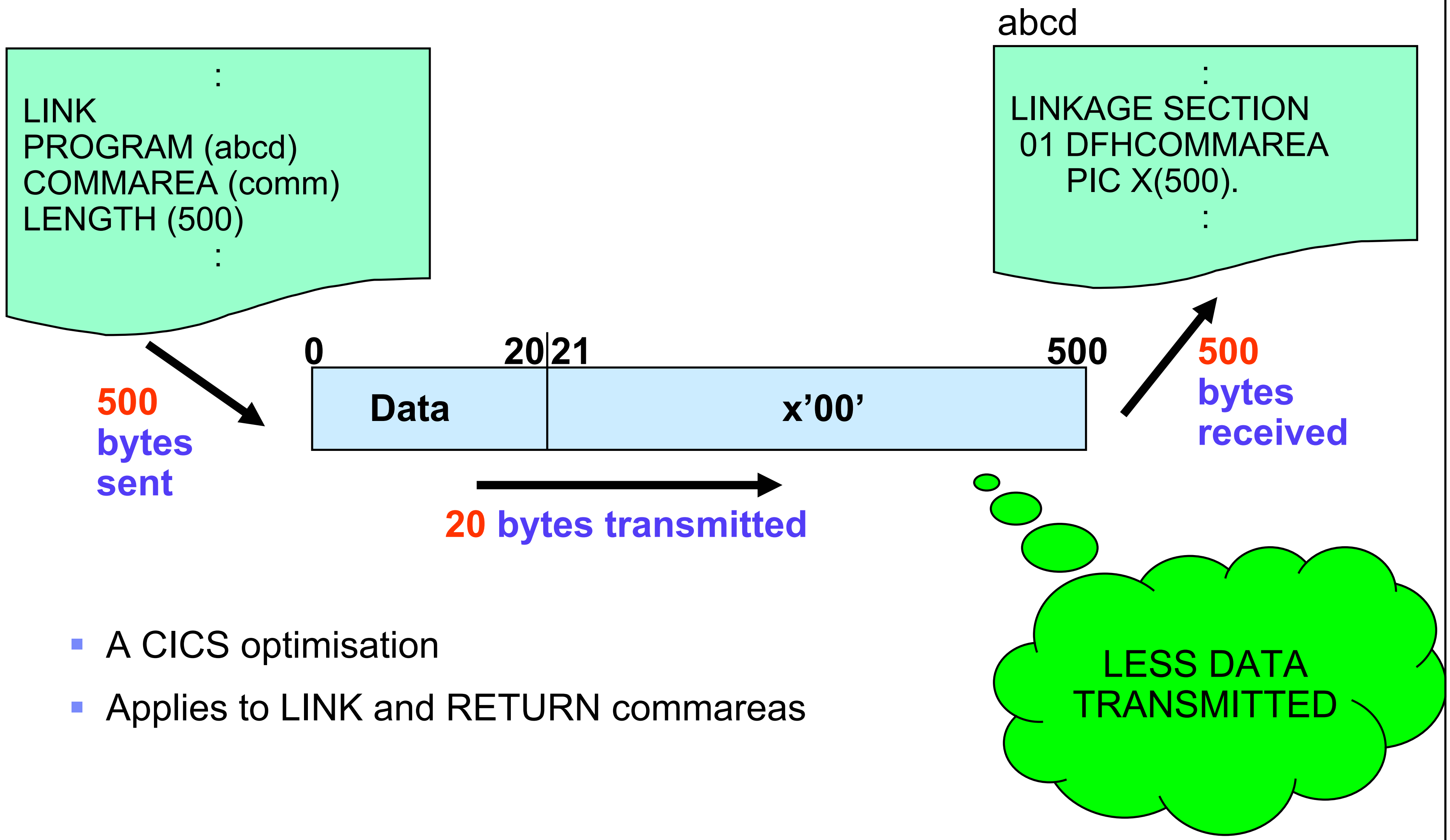
```
EXEC CICS LINK
  PROGRAM
  COMMAREA
  LENGTH
  DATALENGTH
```

- Use DATALENGTH to reduce data transmission

```
Call PROG1 USING parm1
```

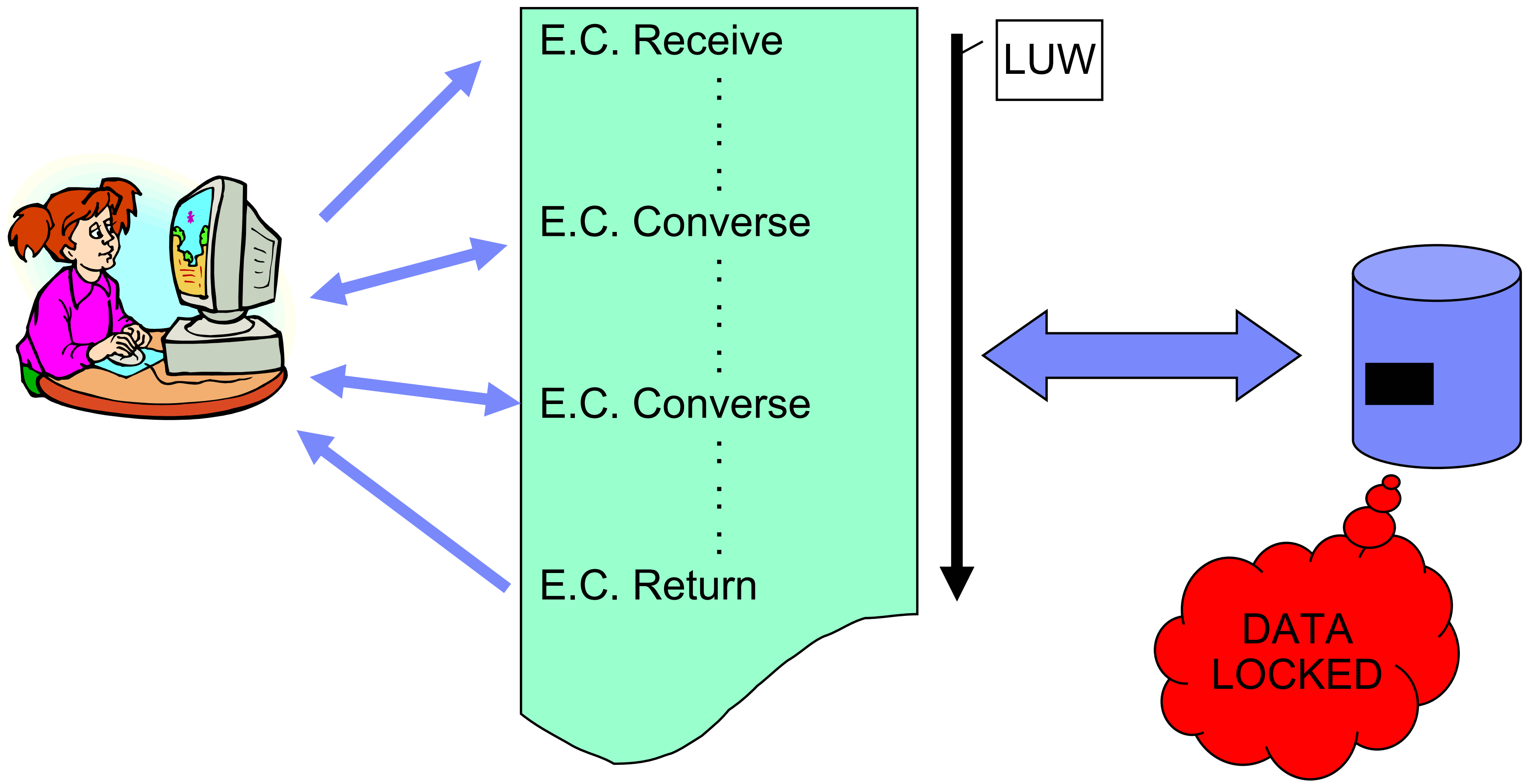
- Use Cobol CALL or C functions for modules with no CICS commands if speed is important.

Null Fill Commareas Before Use

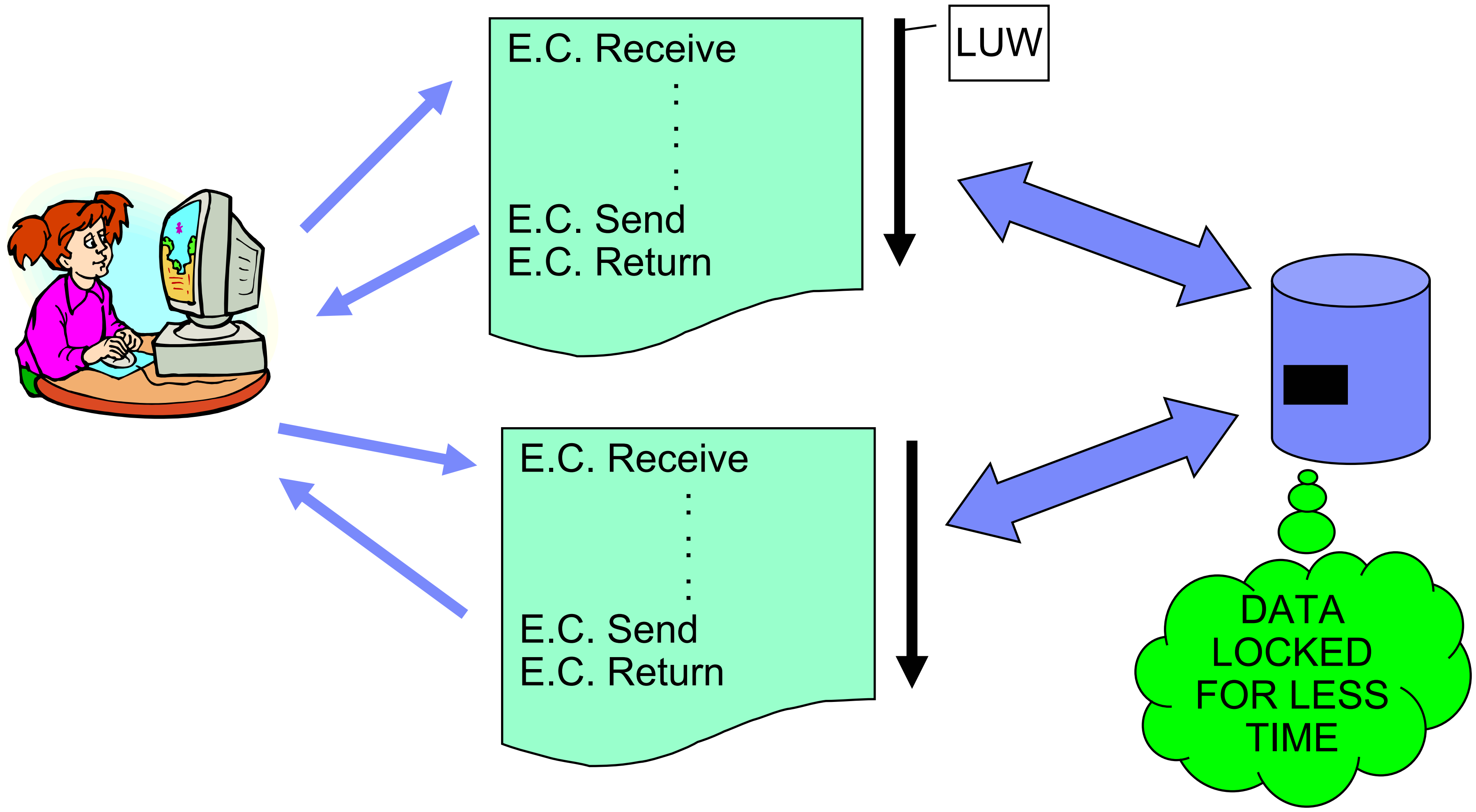


- A CICS optimisation
- Applies to LINK and RETURN commareas

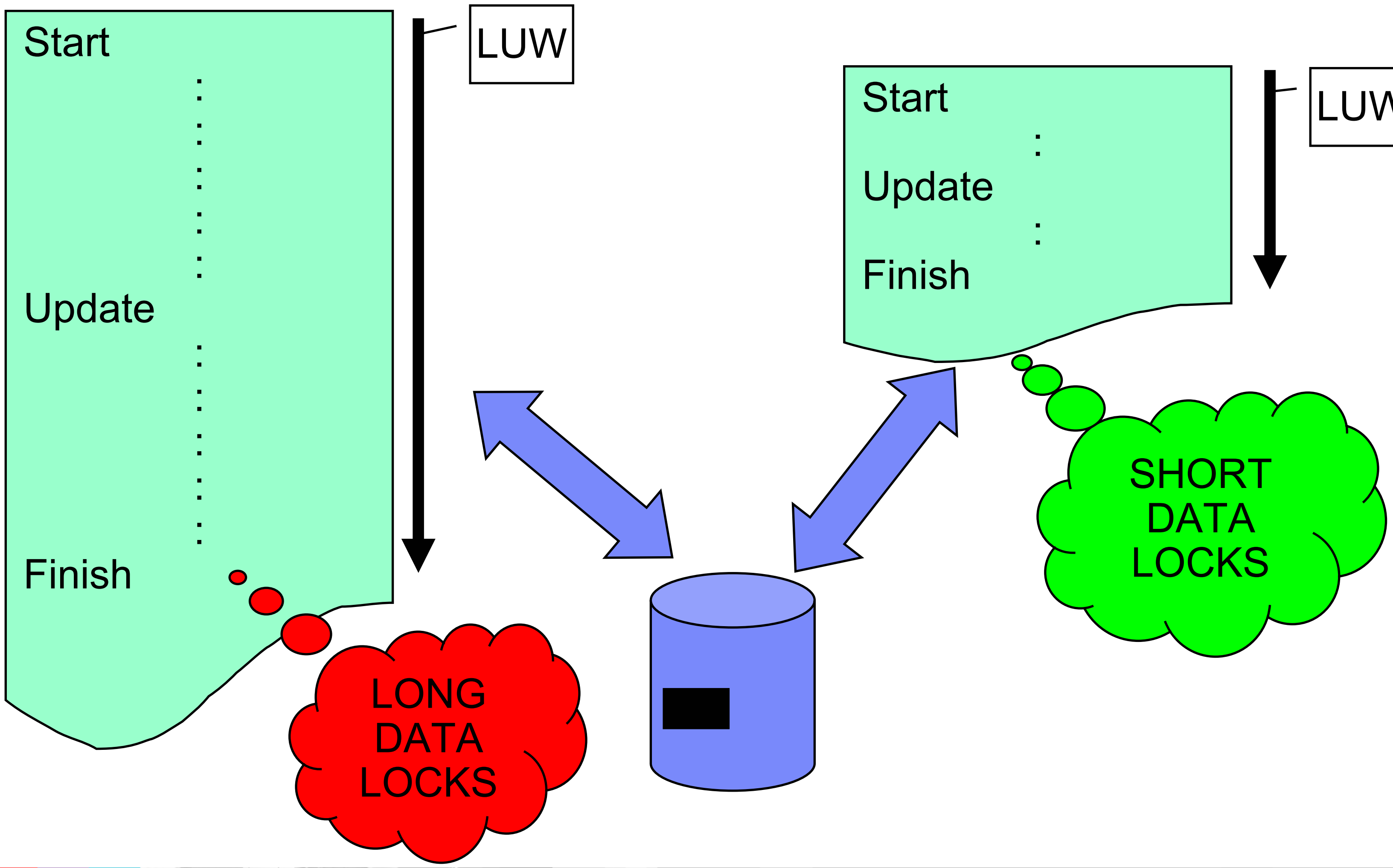
Avoid Conversational Programs



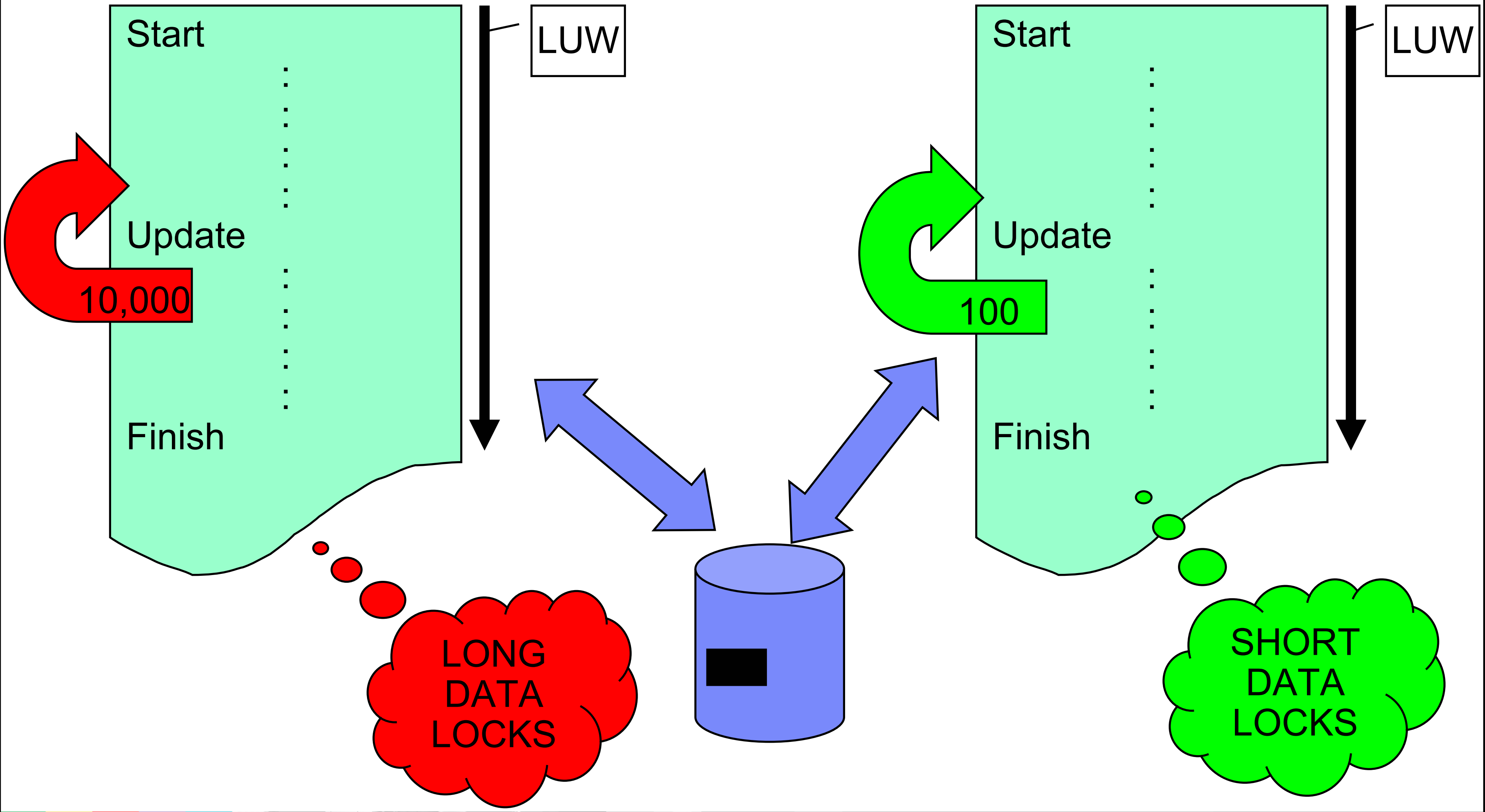
Avoid Conversational Programs: Use Pseudo or Non-Conversational Instead



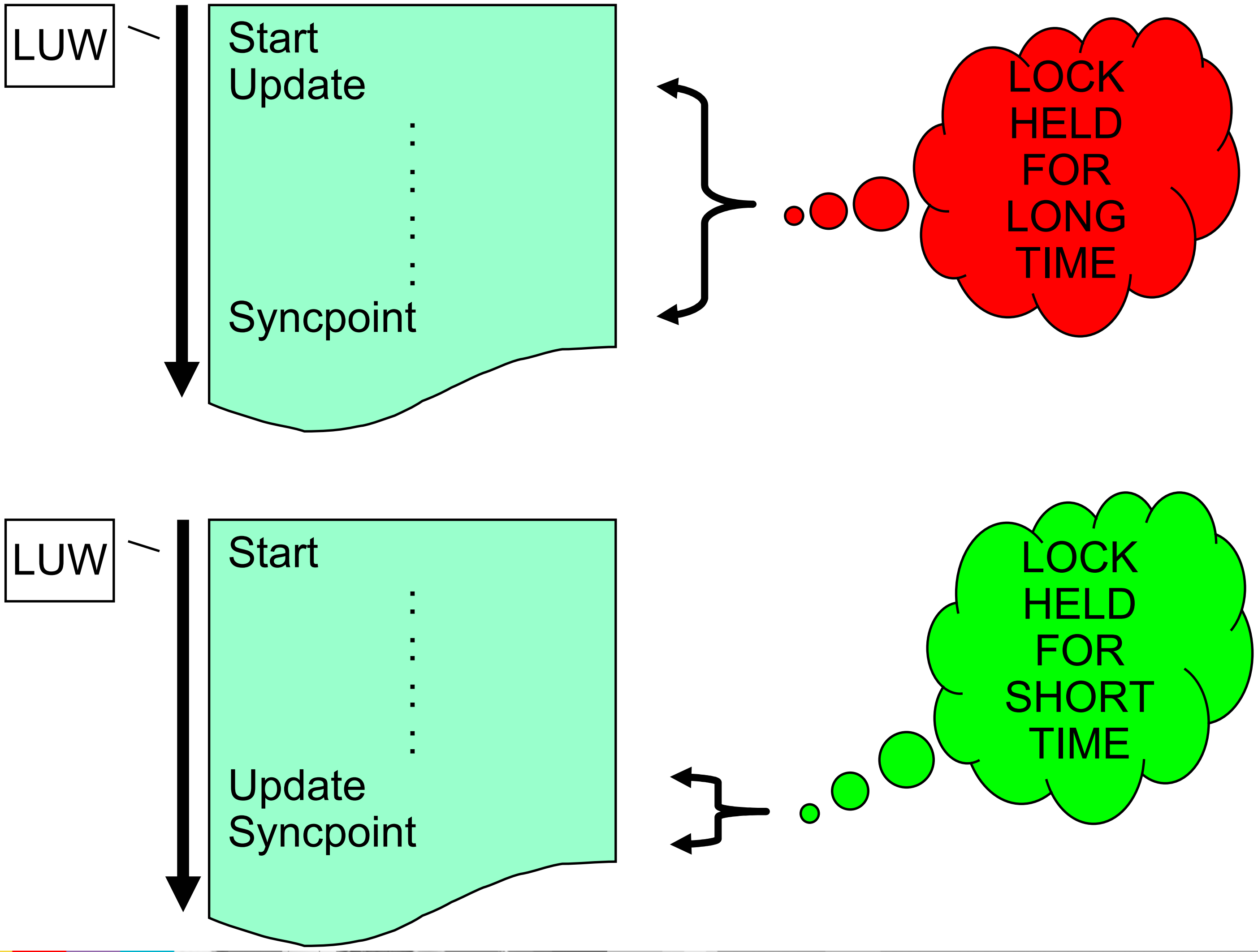
Keep Logical Units of Work **SHORT**



Keep Logical Units of Work **SMALL**



Update Close to Syncpoint



Minimise ENQ Times – Implicit ENQ

Resource	Command	ENQ Argument
VSAM File	READ UPDATE WRITE DELETE	Logical record
Intrapartition Transient Data Queue	WRITEQ READQ DELETEQ	Destination
Temporary Storage Queue	WRITEQ READQ* DELETEQ	Queue name

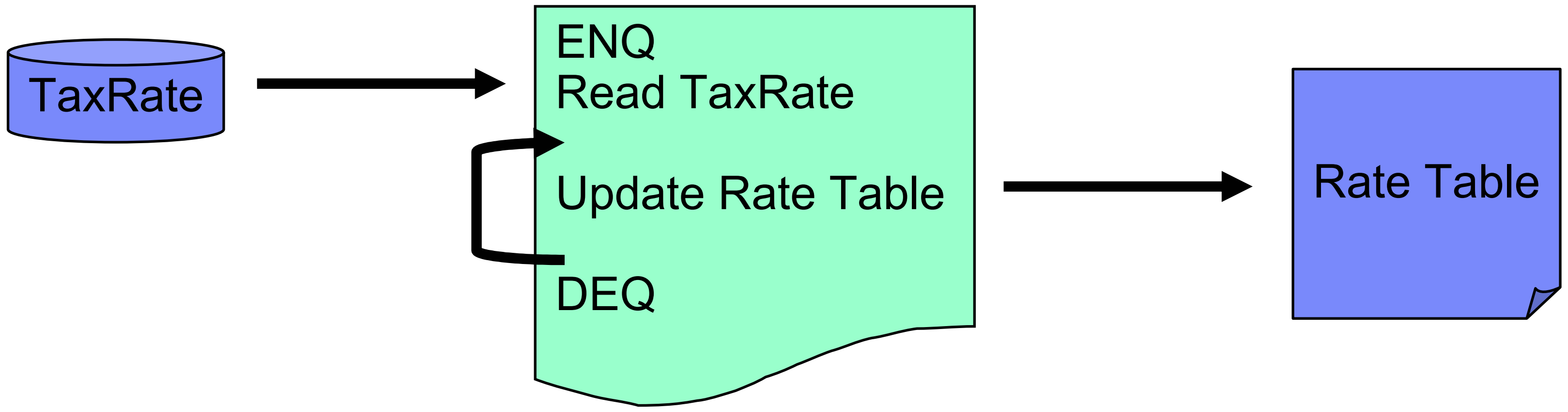
- Not locked if SFS used as a file manager, “dirty reads allowed”
- Implict DEQ occurs at end of LUW (hence update close to Syncpoint / Rollback)

Minimise ENQ Times – Explicit ENQ

Provides exclusive control when updating a shared resource

```
EXEC CICS ENQ
  RESOURCE (data-area)
  LENGTH (data-value)
  NOSUSPEND
```

```
EXEC CICS DEQ
  RESOURCE (data-area)
  LENGTH (data-value)
```



Minimise ENQ Times

- Used in following circumstances
  - Protect data with no implicit ENQ (Rate Table in example)
  - Prevent deadlocks (deadly embrace)
  - Protect TS queue from concurrent read and update (if using SFS)

- To be effective
  - Minimise ENQ time
  - All programs must use same rules for accessing resource

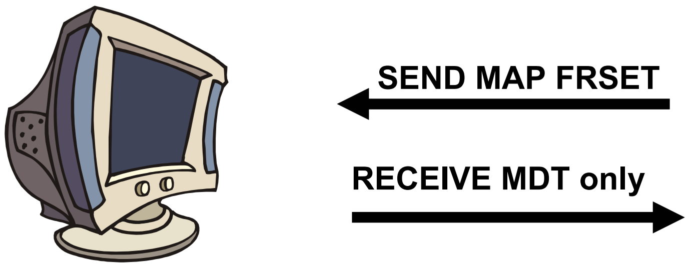
Minimise Terminal Traffic

```
EXEC CICS SEND MAP
  FROM MAPONLY
```

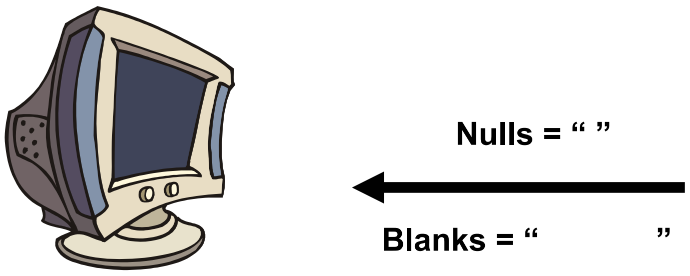
- Use MAPONLY for new screens

```
EXEC CICS SEND MAP
  FROM DATAONLY
```

- Send only changed fields to existing screen

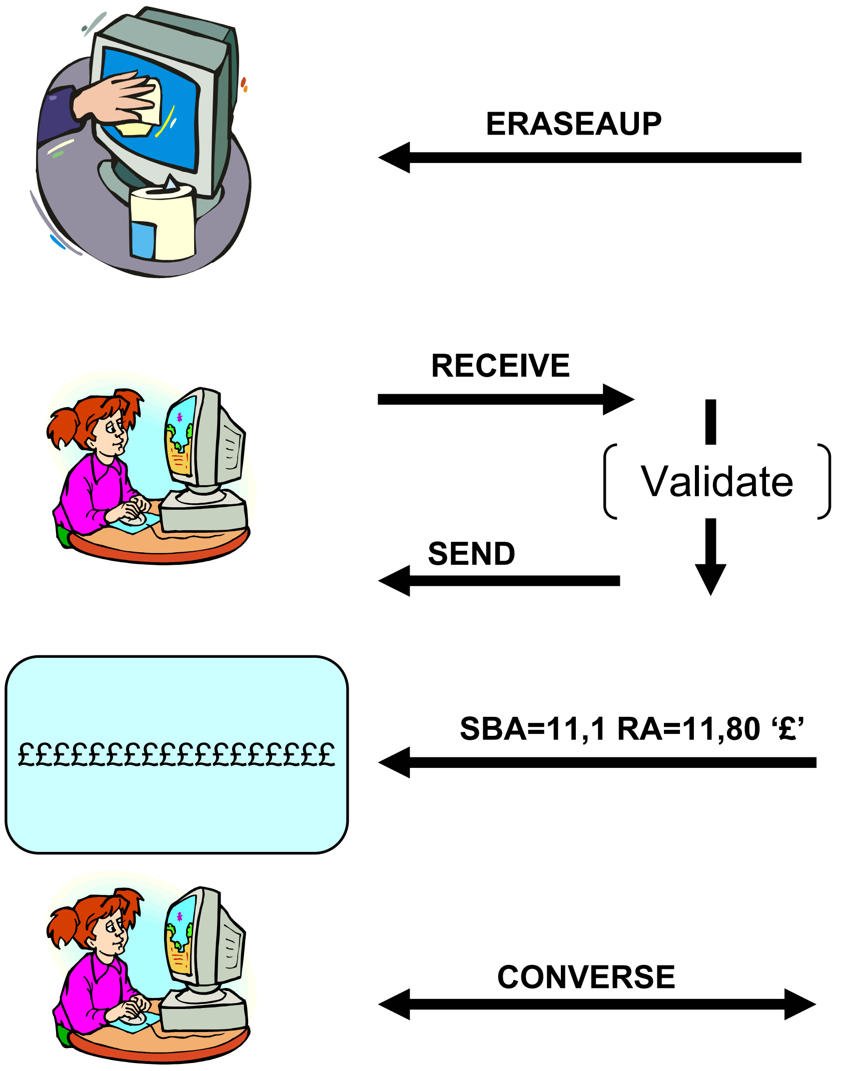


- Reset MDTs using FRSET



- Send Nulls (x’00’) not Blanks

Minimise Terminal Traffic



- Use ERASEAUP on SEND MAP or SEND CONTROL to clear **unprotected** fields
- Validate all entry-fields in 1 pass
- Use SBA and RA to send repeat data
- Use CONVERSE rather than SEND, RECEIVE if conversational

Use Right Data Storage Choice

USE THE RIGHT ONE

- Generally 2 types of storage available.
  - Task-private – for sole use of current transaction
  - Task-shared – shared by all transactions
- Using the wrong type affects performance and wastes storage

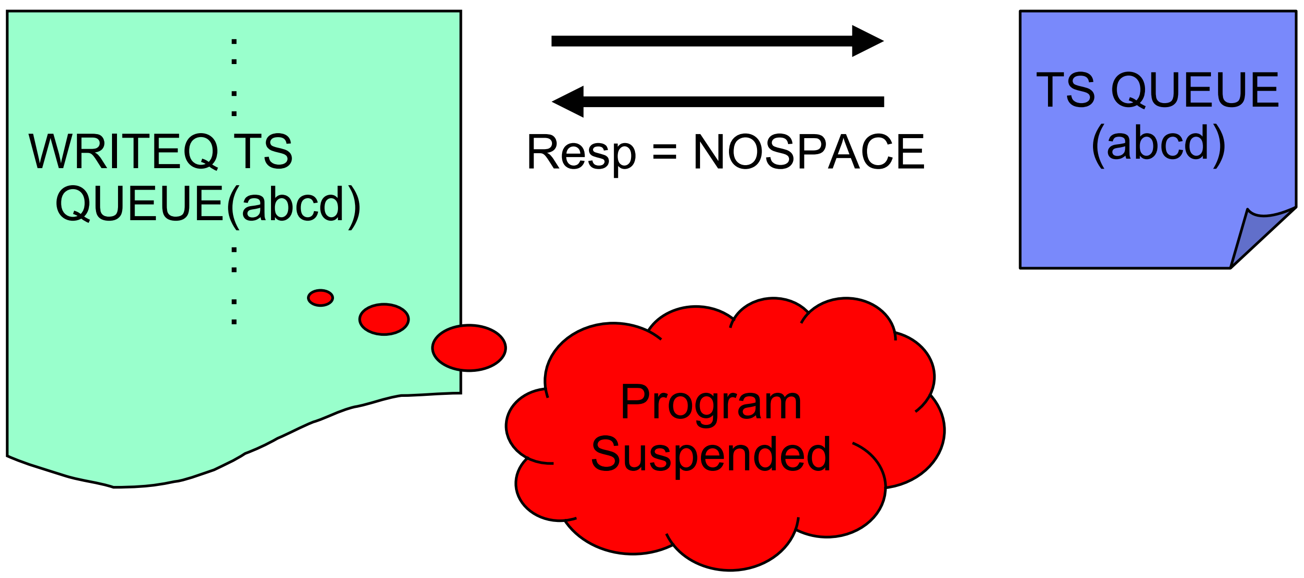
Use Right Data Storage Choice – TASK PRIVATE

- Transaction Work Area (TWA)
  - Set by TWASize in TD stanza
  - Last for duration of transaction
  - Available to all programs in transaction
- GETMAIN without SHARED
  - Useful for large, variable storage requirements
  - Released at task termination
- COMMAREA
  - Passed on LINK, XCTL and RETURN commands
- INPUTMSG
  - Similar to commarea but use RECEIVE to obtain
- Program storage
  - Either reused or a new copy depending on language and invocation

Use Right Data Storage Choice – TASK SHARED

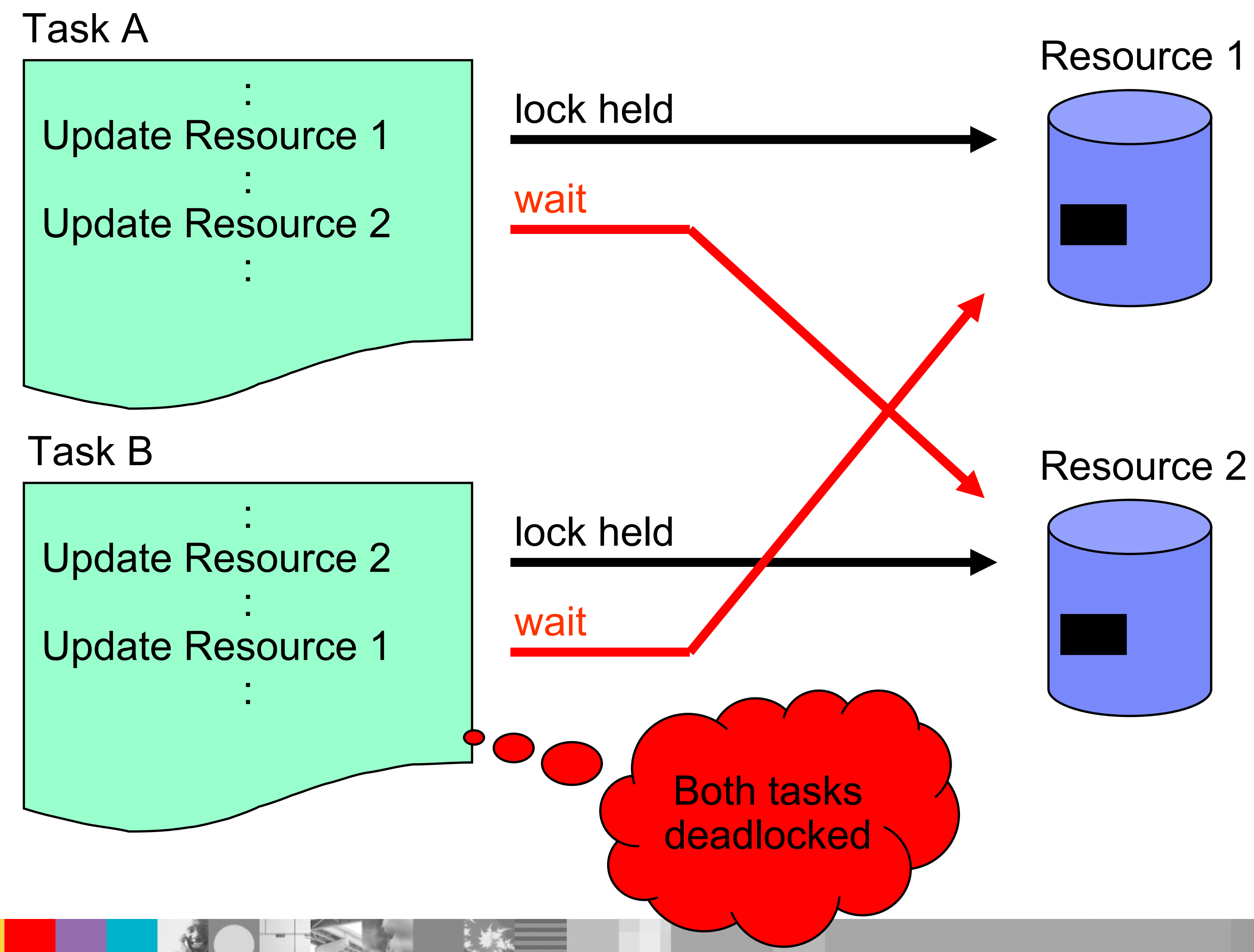
- Temporary Storage Queues (Main or Auxiliary)
- Transient Data Queues
- Common Work Area (CWA)
  - Set by CWASize in RD stanza
  - Available to all programs in the region
  - Application managed
- Terminal User Area (TUTUA)
  - Similar to CWA, but only available to a terminal user
- Display screen
  - Data hidden on screen for next transaction
- Operating System files
  - Flexible with large overhead
- GETMAIN with SHARED
  - Available to any program. Application managed

Use NOSUSPEND



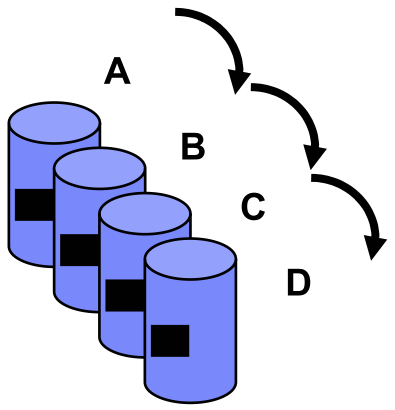
- Use NOSUSPEND to prevent program suspension
- Suspension is default for ENQBUSY, NOJBUFSP, NOSPACE, QBUSY, and SYSBUSY conditions
- Available for ALLOCATE, CONNECT PROCESS, ENQ, JOURNAL, READQ TD, and WRITEQ TS commands

Deadlocks – How it happens

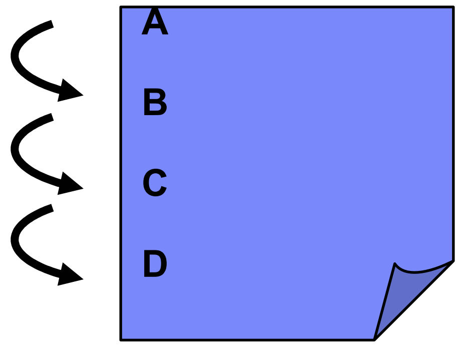


Deadlocks – How to Avoid

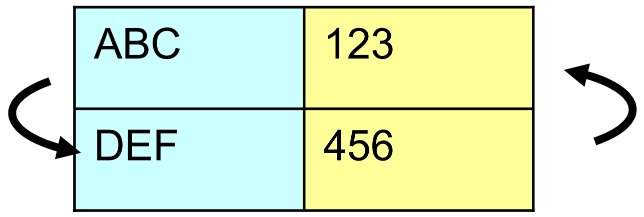
- Access data in same order. Includes
  - Multiple resources (files, queues)



  - Multiple records in same resource
    - Example, ascending
    - Sort data before accessing



  - Beware of updating files via Base and Alternate indexes



ABC	123
DEF	456

Deadlocks – How to Avoid

- Design ENQ development standards
  - Use same strings
  - Use strings in same order

- After a READ UPDATE use REWRITE, DELETE without RIDFLD or UNLOCK to release record.

- Use ENDBR to release file lock before using READ UPDATE, WRITE or DELETE with RIDFLD on same file.

Test for Errors

Cobol

```
                      :
EXEC CICS
  SEND TEXT FROM (data)
  RESP (xxx) END-EXEC

IF xxx=DFHRESP(INVREQ) THEN
                      :
```

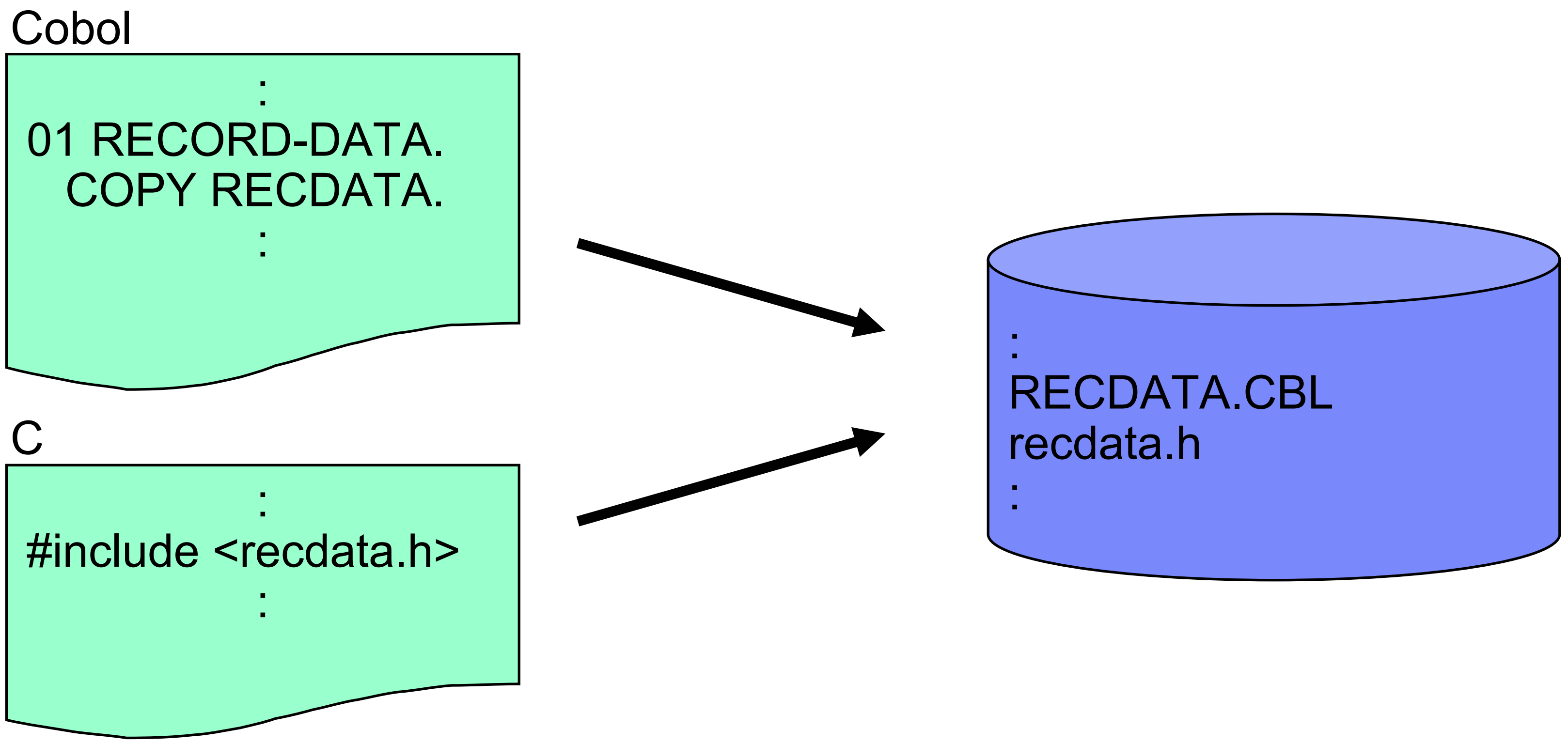
C

```
                      :
EXEC CICS
  SEND TEXT FROM (data)
  RESP (xxx);

if ( xxx = DFHRESP(INVREQ)) {
                      :
```

- Always code RESP and if needed RESP2
- Use DFHRESP function to test RESP and RESP2 values
- Use NOHANDLE to ignore any conditions
  - Assumed in C programs by default. Be careful
- Avoid HANDLE CONDITION, IGNORE CONDITION, HANDLE ABEND, POP/PUSH HANDLE commands (Cobol and PL/I)

Use Libraries for Common Structures

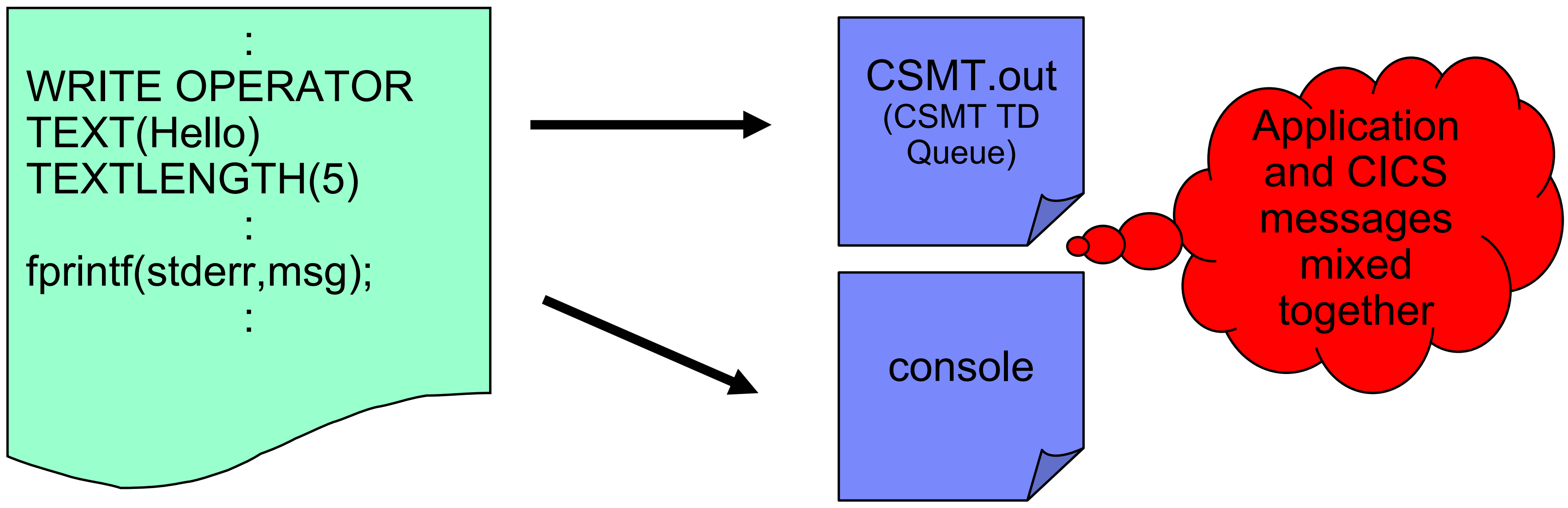


- Use libraries for structures used in several programs
- Applies to
  - Record layouts, file formats, BMS structures, commareas…
  - … in fact, anything that is common between two programs

Avoid CICS Reserved Letters and Names

- Abend codes starting with letter **A**
- Resources (transactions, TD, TS Queues) starting with letter **C**
- Maps, tables or programs starting with letters:-
  - **DFH**
  - **ERZ**
  - **FAA**
- User variable names:-
  - **EXEC**
  - **CICS**
  - **END-EXEC**

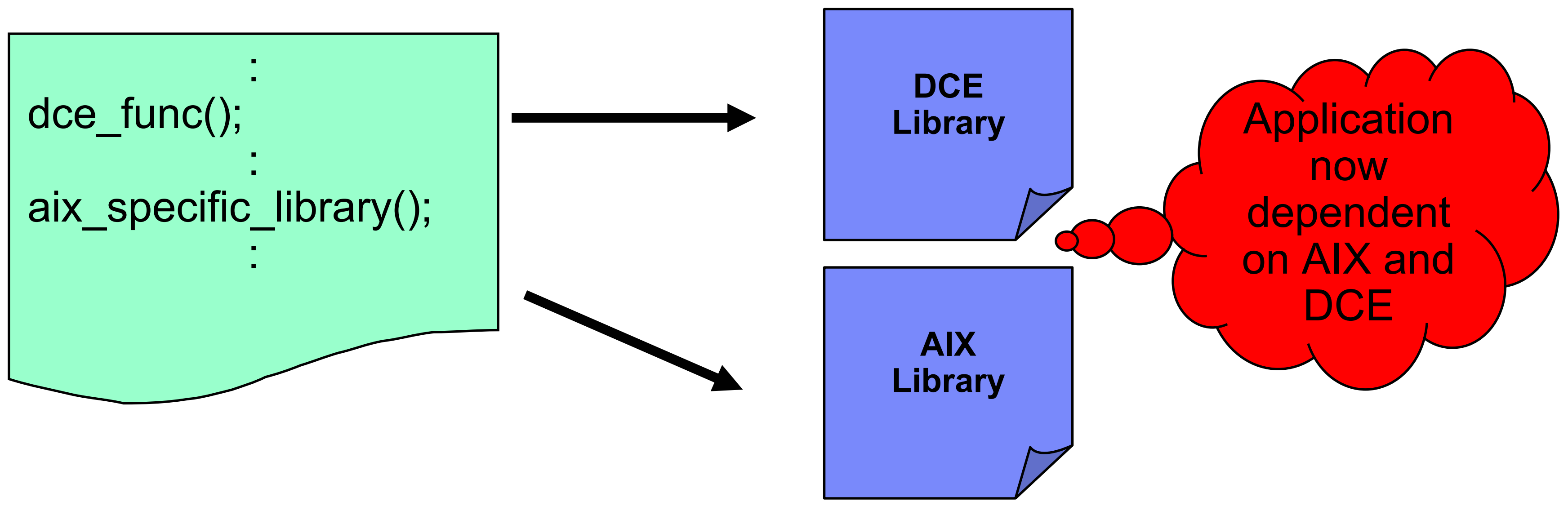
Avoid use of WRITE OPERATOR



```
:
WRITE OPERATOR
TEXT(Hello)
TEXTLENGTH(5)
:
fprintf(stderr,msg);
:
```

- Application and CICS messages stored in CSMT and CICS console
- Write application messages to own extra-partition TD queue

Avoid DCE and Operating System Libraries



- Application is now dependent on TXSeries and specific Operating System
  - Migration more dificult
- Libraries may affect behaviour of CICS program
  - For example, library may not be thread-safe

Thread Safety

- Always use thread-safe libraries
  - Example: **ctime** keeps static data between calls.
- Thread-safe versions of most libraries provided – USE THEM
  - Example: **libc_r** rather than **libc**
  - **ctime_r** rather than **ctime**
- Only make EXEC CICS calls and do XA work from MAIN thread

Some functions are thread-safe but not CICS-safe. Do not use the following

- Any function that is not thread safe
- exec
- setlocale
- Shared memory functions (do not attach memory at the address specified with the Region Definitions (RD) RegionPoolBase attribute. CICS uses this address for region pool shared memory).
- CICS internal functions
- exit, abort or _exit
- fork
- stdin, stdout, stderr
- kill (do not send signals to any CICS process)
- raise
- raise
- assert
- abort
- sigprocmask
- signals
- signal masks
- cin, cout, cerr
- DCE asynchronous cancellation
- DCE threads
- Encina TRAN
- Encina Transactional C
- Encina threadTid
- catch(...) (in C++ programs –any exceptions not generated by the application must be rethrown (using throw with no argument).

Default CICS Options in C Programs

- Always specify LENGTH option, except for
  - SEND MAP, RECEIVE, READ, READNEXT, READPREV, REWRITE and WRITE
- Can omit FROM on SEND MAP and INTO on RECEIVE MAP provided MAP option is a literal.

Commands	Option	Defaults to
EXEC CICS SEND MAP('MAP1')	FROM	&map1.map1o
EXEC CICS RECEIVE MAP('MAP1')	INTO	&map1.map1i

EIB Data Declarations Needed in C and C++

The following data types for EIB are used:

C Type	CICS Data Type
8-bit unsigned binary	cics_ubyte_t
16-bit unsigned binary integers	cics_ushort_t
16-bit signed binary integers	cics_sshort_t
32-bit unsigned binary	cics_ulong_t
32-bit signed binary	cics_slong_t
Single character	cics_char_t
Character strings	cics_char_t arrays
Boolean	cics_bool_t

General C and C++ Considerations

- When C or C++ programs are cached, variables in static storage are not reinitialized when the program is re-invoked.
  - Fresh copy of static data only on first instance of program
- BMS maps may be stored as static, which allows previous data to be shown on the screen. If you do not want this because of the possiblity of security exposures, either change the code so that the variables in static storage are not used, or do not cache the program.
- CICS does not support null-terminated string handling
  - All Strings returned by CICS are space padded
  - Example: TS queues must be 8 characters
- Do not use iostream in C++ programs
- Do not place CICS statements in:
  - Header files as part of inline function definition
  - Static object consructors and destructors

EIB Data Declarations Needed in Cobol

The following data types for EIB are used:

COBOL Type	CICS Data Type
16-bit binary integers	PIC S9(4) COMP
32-bit binary integers	PIC S9(8) COMP
Character strings	PIC X(n) where n is number of bytes

General Cobol Considerations

- Cobol programs are given a fresh copy of working storage when first used.
  - Use /DATA-CONTEXT on compile to give a fresh copy every time (Micro Focus only)
- Generally should avoid recursion
  - Acucobol sets RECURSION=true by default
- Mixed languages supported via CALL provided called program does not contain any CICS commands
- Declare data passed between Cobol and C in a commarea as COMP-5 to preserve byte ordering
- Using STOP RUN causes application server to terminate
- Micro Focus programs are never unloaded from memory