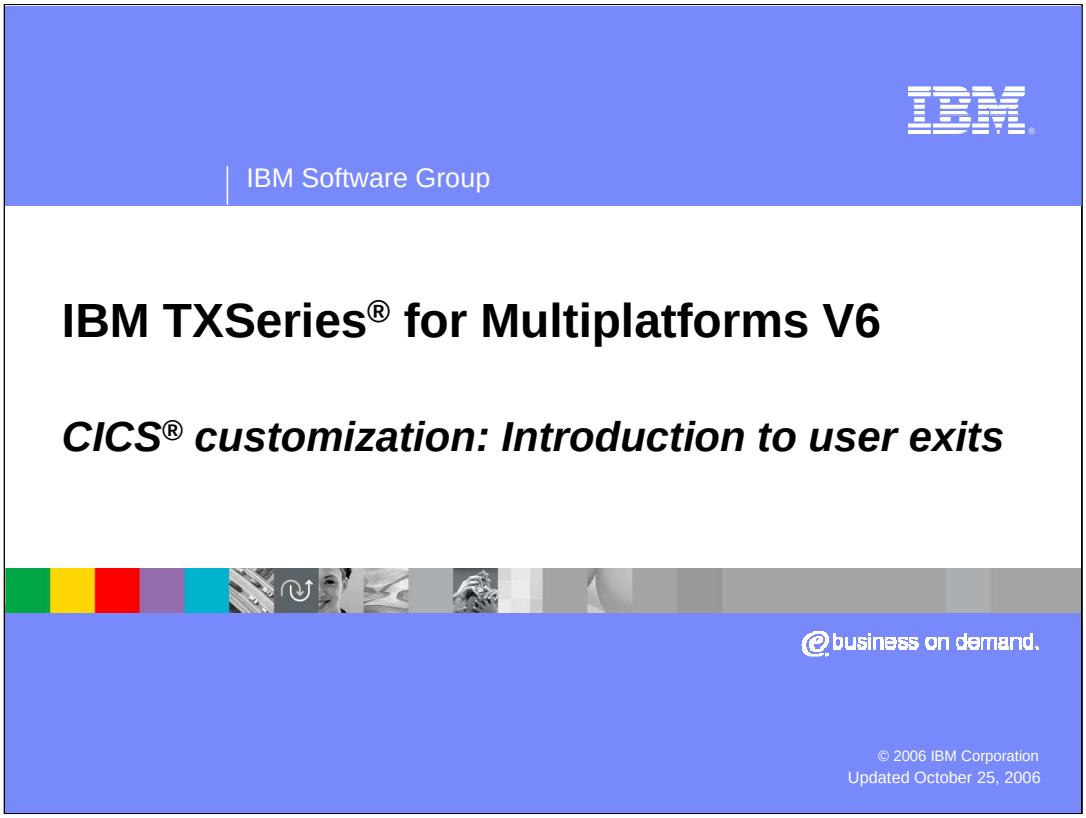


This presentation covers the CICS customization technique called User Exits.

Goals

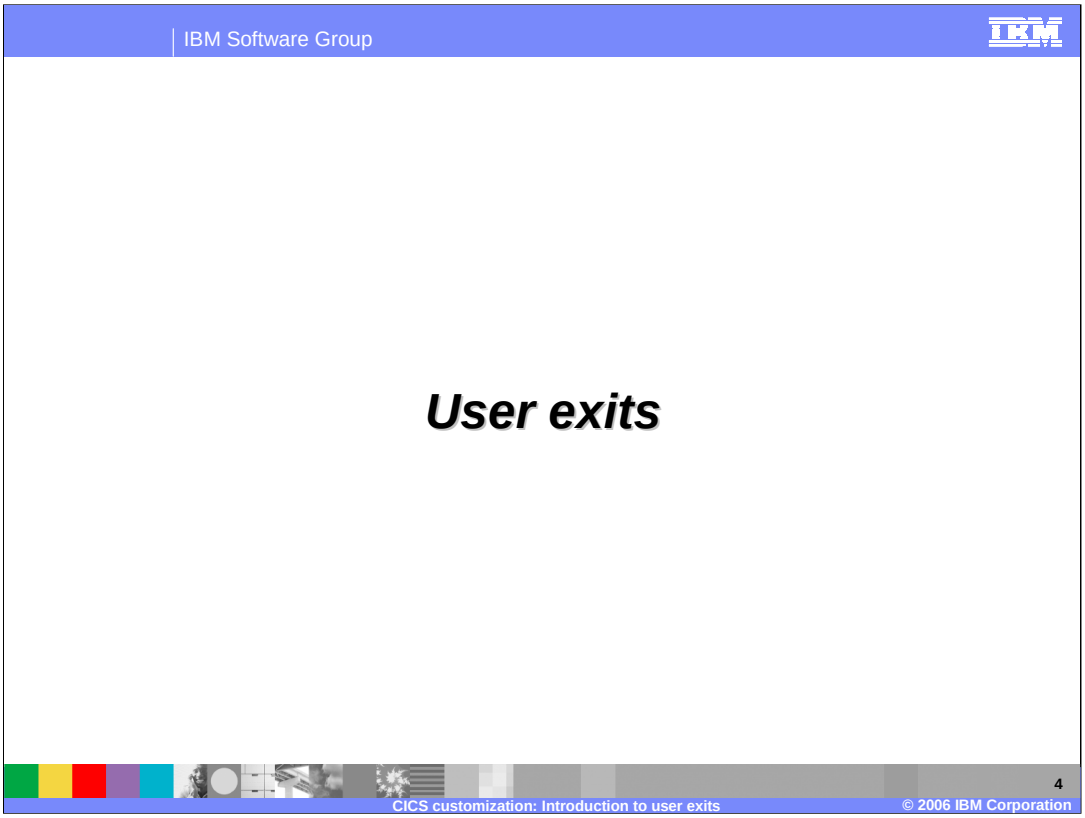
- Define user exits and their semantics
- Understand user exit points provided by TXSeries
- Guidance or rules to write user exit programs
- Typical advantages of user exit programs

The goal of this presentation is to provide an overview of user exits and their semantics, an understanding of user exit points provided by TXSeries, and an overview of the rules for writing user exit programs. You will also see some typical advantages of user exit programs.

Agenda

- User exits
  - What are user exits?
  - User exit semantics
- User exit points provided by TXSeries
- User exits principle
  - Guidance or rules for writing user exit program
  - Typical uses of user exits

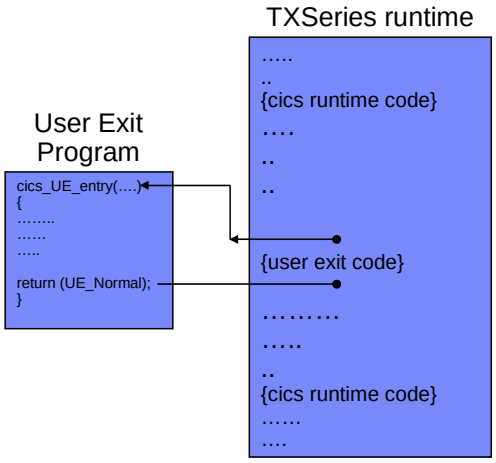
The agenda for this presentation is to first define user exits, then describe the user exits provided by TXSeries, and then describe the rules for using and typical uses of user exits.



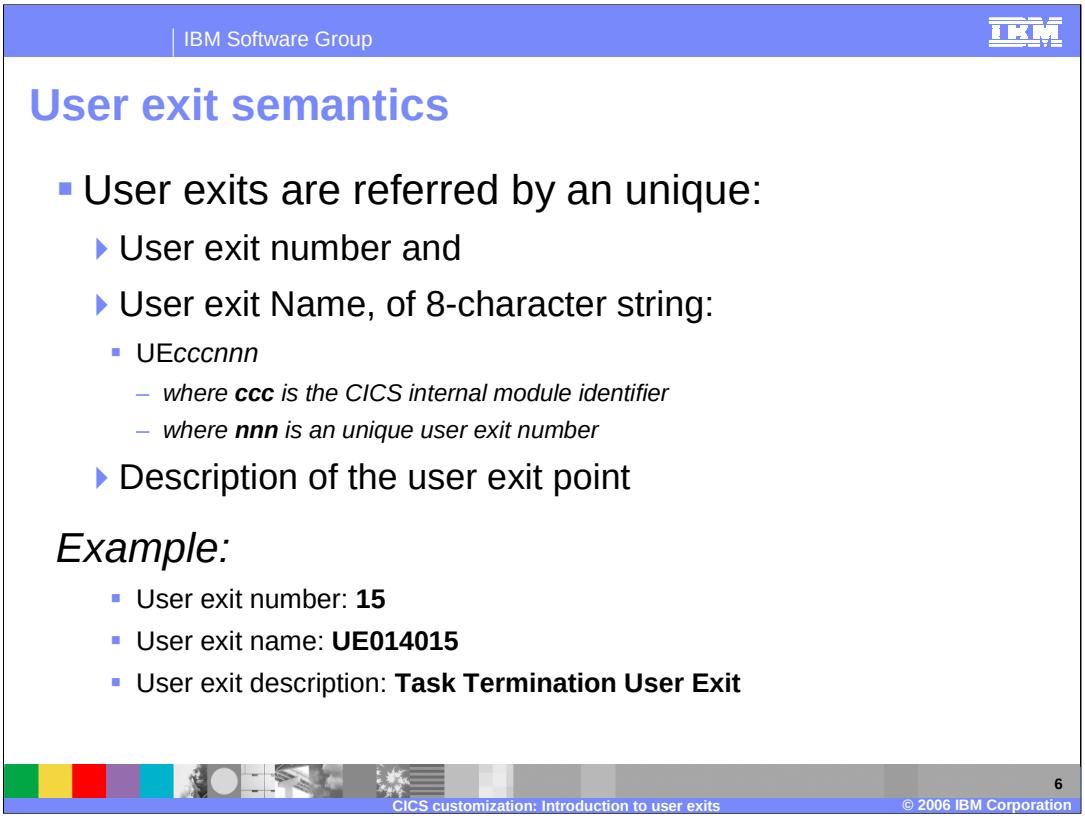
This section covers the concept of user exits, semantics and how to define user exit programs to the CICS region.

What are user exits ?

- A user exit (also referred as an user exit point) is a place in a CICS runtime at which CICS transfers control to a user program, and resumes control when your exit program has finished its work.
  - Used to extend and customize the function of your CICS system in accordance with your own environment



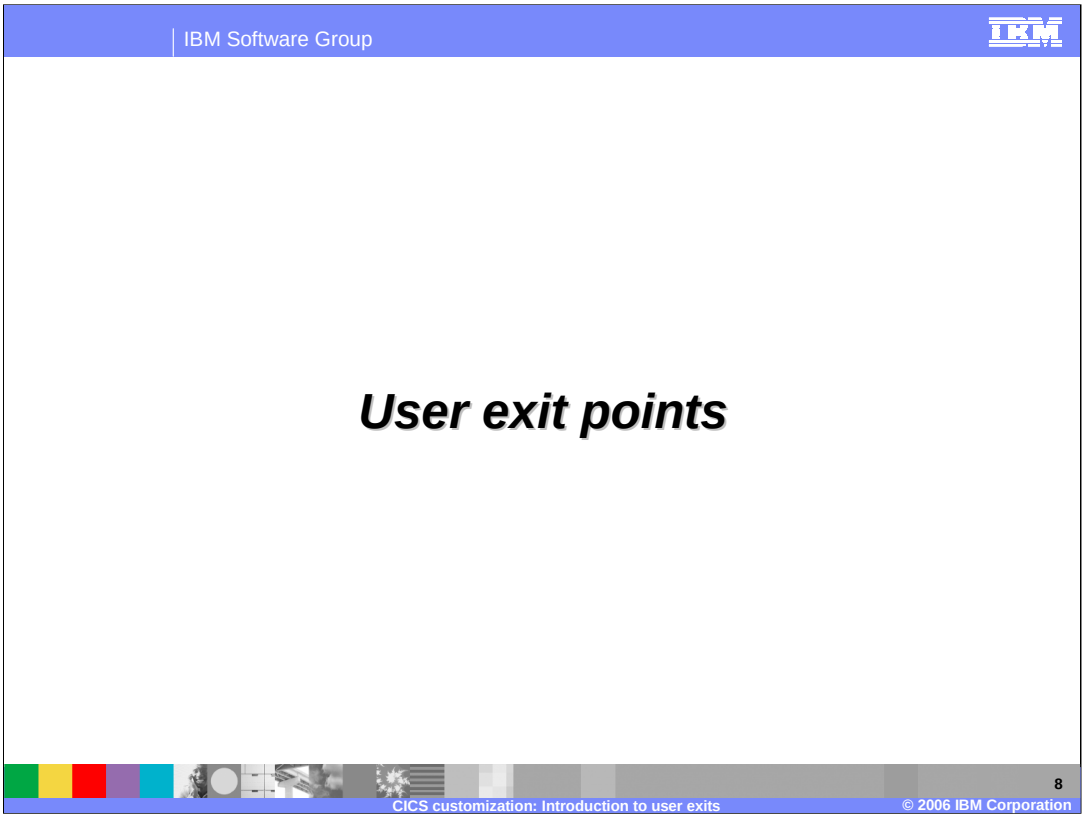
A user exit, also referred to as an user exit point, is the point in a CICS program at which CICS can transfer control to a program that you have written (a user exit program), and can then resume control when your program has finished its work. You do not have to use any of the user exits, but they are useful if you want to extend and customize the function of your CICS system to meet your specific requirements.



Every user exit point in TXSeries has a unique name and a number associated with it. The unique names and the numbers for user exit points are defined by TXSeries and cannot be modified. User Exits are generally named based on the CICS internal module identifier of the CICS component that invokes the user exit program, along with a unique user exit number.

As an example, TXSeries defines a user exit point called Task Termination User Exit which has a User Exit Number of 15 and User Exit Name of UE014015.

A CICS region invokes a particular user exit if you define the user exit in the region. The user exit programs can be defined by adding a PD entry and specifying the UserExitNumber. CICS would automatically recognize the PD entry as a user exit entry, and would invoke the program as mentioned in the PathName during the user exit point.



This section provides a list of user exit points provided by TXSeries, and describes some of the advantages offered by these user exit points.

User exit points provided by TXSeries

User exit number	User exit name	Description
13	UE015013	Program autoinstall
15	UE014015	Task termination
17	UE052017	Dump request
25	UE014025	Dynamic transaction routing
31	UEI09031	FEPI data conversion
33	UE046033	Dynamic resource definition
50	UE015050	Dynamic distributed program link
51	UE016051	Syncpoint
52	UE115052	IIOP security exit
53	UE115053	IIOP certificate exit

At a glance, the table in the slide shows the list of user exit points provided by TXSeries. Every user exit point has a unique name and number associated with it.

User exit points – In brief

- Program autoinstall user exit (UE015013)
  - Allows the program definition to be installed dynamically when a program or map that is being run does not have a PD entry.
- Task termination user exit
  - Invoked at normal and abnormal task termination.
- Dump request user exit
  - Invoked immediately before a system or transaction dump is taken.

The Program autoinstall user exit allows the program definition to be installed dynamically when a program or map that is being run does not have a PD entry. This user exit is invoked when any of the following conditions exists: LINK, LOAD, XCTL, SEND MAP and RECEIVE MAP. This user exit is called also for maps and programs.

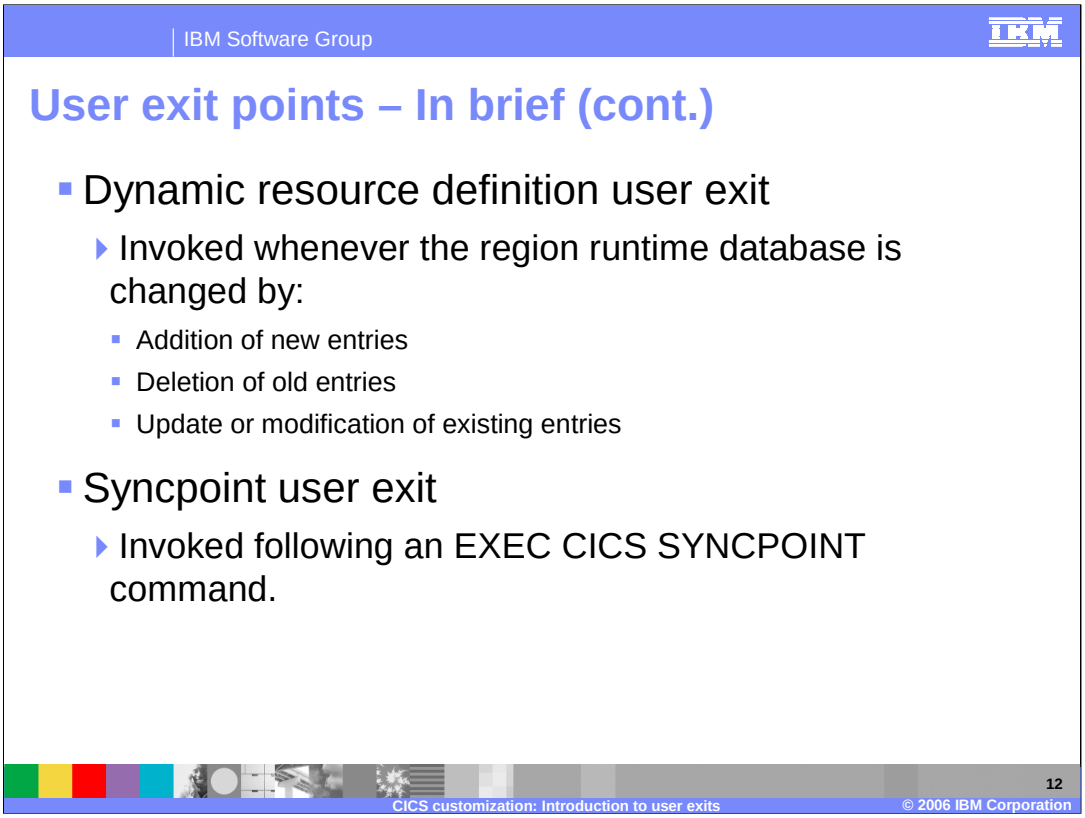
The Task termination user exit is invoked at normal and abnormal task termination (after any syncpoint has been taken). Ensure that no recoverable work is done in the user exit program.

The Dump request user exit is invoked immediately before a system or transaction dump is taken.

User exit points – In brief (cont.)

- Dynamic transaction routing exit
  - Invoked for the following four different cases:
    1. Immediately before a transaction that is defined as dynamic in its Transaction Definition is run either locally or remotely.
    2. After a failed attempt to route a dynamic transaction to a remote system.
    3. After a successful invocation of a dynamic transaction (remotely or locally) but transaction failure is detected.
    4. After successful execution of a dynamic transaction.

The dynamic transaction routing user exit is invoked in the four different cases listed in the slide.



The Dynamic resource definition user exit is invoked whenever the region runtime database is changed. The changes can be addition, deletion or updates to the existing resource entries.

The Syncpoint user exit is invoked following an EXEC CICS SYNCPOINT command. The result of the EXEC CICS SYNCPOINT command is made available in this user exit program. It is not possible to change the outcome of the EXEC CICS SYNCPOINT command in this user exit program.

User exit points – In brief (cont.)

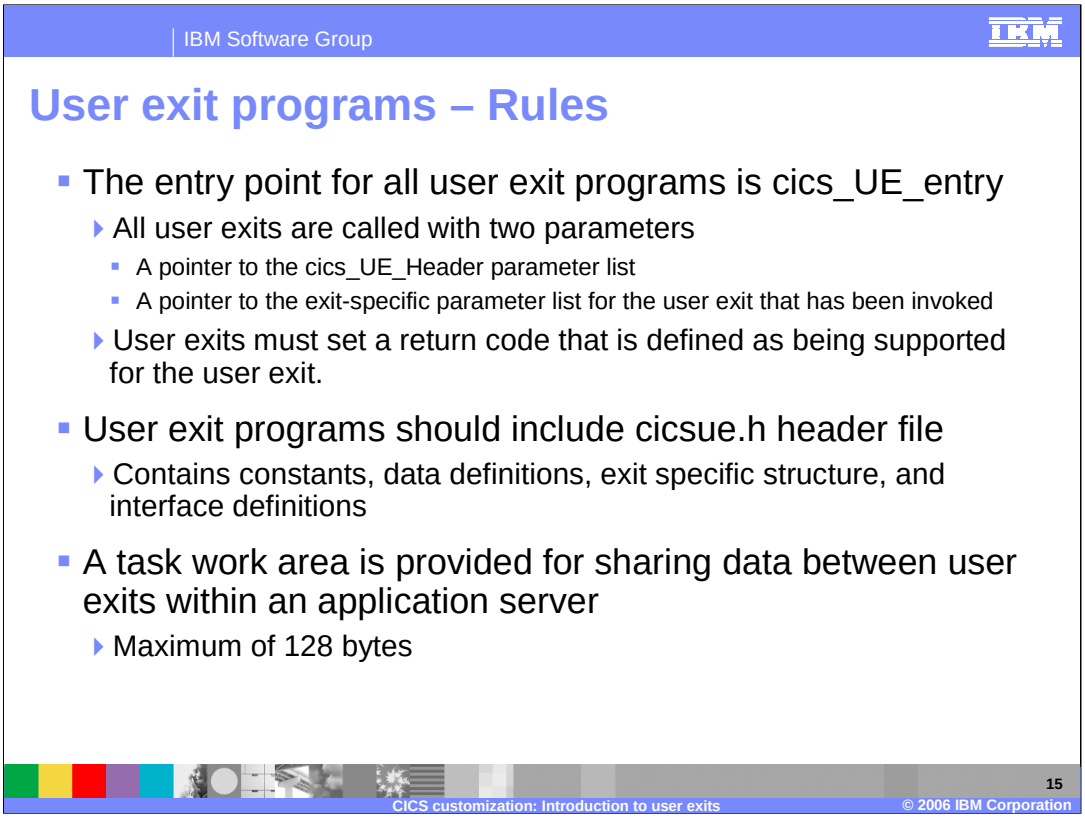
- Dynamic distributed program link user exit
  - Invoked for the following four different cases:
    1. When CICS is about to link to a program.
    2. When CICS is about to link to a program that does not have a Program Definition (PD).
    3. After a successful attempt to link to a program and the initial user exit program invocation requests reinvocation.
    4. After a failed attempt to link to a program, if the initial user exit program invocation requests reinvocation.

The distributed program link user exit is invoked in the four different cases listed in the slide.

Section

Rules for writing user exit programs

This section covers certain guidance or rules to be followed when writing user exit programs.



Writing User exit programs is different from writing a CICS server application. User exit programs should not be run as normal CICS transactions.

Every program running under TXSeries must have an entry point. Thus for user exit programs the entry point is defined as cics_UE_entry. CICS supplies parameters to the entry point – a header parameter list and a user exit specific parameter list, with which user exit programs can know which user exit point has been invoked and also to get relevant user exit specific information. As with any program, user exit programs must return a return code specifying certain action to be taken by CICS. The return code value depends on the user exit point.

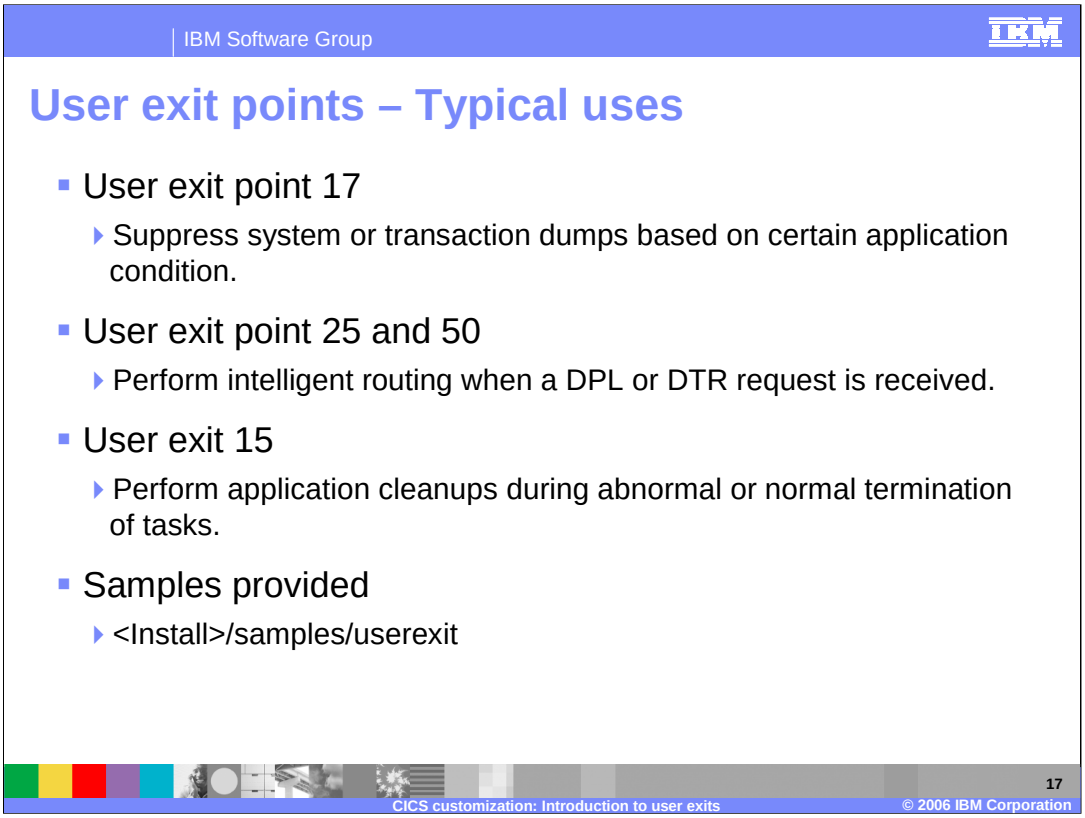
Every user exit program must include TXSeries provided cicsue.h header file, which consists of constants, data definition, exit specific structures, return codes and interface definitions.

If you want to share user data across user exits within the same application server process, you can use the task work area provided with every exit point. The task work area is a common buffer passed to all the user exit programs and can be shared among user exits. A maximum of 128 bytes can be stored in the task work area.

User exit programs – Rules (cont.)

1. Must be written in C language only
2. Must be reentrant
3. Must not switch threads
4. Must return control to CICS
5. Must not damage integrity of CICS
6. Must not use CICS API commands
7. Cannot access CICS resources
8. Signal handlers must not be included in user exit programs

Listed here are some of the important guidelines for writing User exit programs.



Some of the advantages offered by the user exit programs in a practical scenario are listed here.

User exit point 17 can be used to suppress system or transaction dumps to possibly avoid numerous dumps in the file system, especially when you are developing new applications and expecting failures.

User exit point 25 and 50 can be used to intelligently route when a Distributed Program Link or Distributed Transaction Request is received. The user exit program can modify the SYSID which on return will be used by CICS to route the requests. Before modifying the SYSID the user exit program can devise logic to identify which system is most appropriate for the requests to be routed to.

User exit point 15 (Task Termination User Exit Point) can be used to perform cleanups such as closing file descriptors, logging messages to application specific log files during abnormal or normal termination of tasks.

A sample user exit program is supplied with the product and can be found in the product samples/userexit directory.

Section

Summary

This section will provide a summary of the topics covered in this presentation.

Summary

- User exits definition and semantics
- User exit points provided by TXSeries
- Rules for writing a user exit program
- Typical advantages of user exit programs

In summary, this presentation has covered user exits, including the exit points provided by TXSeries, rules for using user exit programs, and advantages provided by them.

Template Revision: 04/25/2006 11:09 AM

Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a registered trademark of Linus Torvalds.

Other company, product and service names may be trademarks or service marks of others.

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.