How to Stop Stalling

Will J. Roden, Jr.
Endicott, NY
March 31, 2000

Contents

Disclaimer

The information contained in this document has not been submitted to any formal IBM test and is distributed on an "As is" basis without any warranty either express or implied. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environment do so at their own risk

.

In this document, any references made to an IBM licensed program are not intended to state or imply that only IBM's licensed program may be used; any functionally equivalent program may be used instead

.

Any performance data contained in this document was determined in a controlled environment and, therefore, the results which may be obtained in other operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

It is possible that this material may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming or services in your country.

Agenda

- What is a Pipeline?
- Multistream Pipelines
- Keeping Records in Order
- Writing Your Own Stages in REXX
- How Stalls Happen
- How to Prevent Stalls
- CMS Pipelines HELP

What is a Pipeline?

Pipelines

- Definition
  - A Pipeline is a series of programs called stages through which data flows
  - The output of one stage automatically becomes the input to the next
  - It is a method of executing a Data Flow diagram

What is a Pipeline?

Pipelines

What is a Pipeline?..... Pipeline

- **Pipe Command:**

```
Pipe   < INPUT FILE A| drop 4| locate 5.1 /4/
|   sort 34-36 | specs 1.3 1 5.1 nw 22.16 nw
|   > OUTPUT FILE A
```

- **Pipe Command in REXX EXEC:**

```
/* TEST1 EXEC                                    */
'Pipe ( endchar ?)',
  '      < INPUT FILE A',        /*Read           */
  '|     drop 4',                 /* Select        */
  '|     locate 5.1 /4/',
  '|     sort 34-36',            /*Sort,Pick      */
  '|     specs 1.3 1 5.1 NextWord 22.16 nw',
  '|     > OUTPUT FILE A'        /*Write          */
```

What is a Pipeline?.. Pipeline

- **Input File:**

```
Nbr S Last      Ans          Next          Nx  Dpt
    E Act       Code         Action        Ac  Nxt
    V Done                   To Do         ID  Act
----+-------1----+--------2--------+-------3-----+--
001 3 RECEIVED               ANSWER        KL  A32
002 3 RECEIVED               ANSWER        BR  A32
003 3 RECEIVED               ANSWER        BI  A00
004 5 ANSWERED   COLD        INT1          VM  B29
005 3 ANSWERED   DUP         CLOSE         CO  B26
006 3 OPENED                 RECEIVE       BI  A00
007 4 REROUTED               RECEIVE       PE  B18
008 3 OPENED                 RECEIVE       BI  A00
009 4 RECEIVED               ANSWER        BE  B18
010 3 ANSWERED   COLD        SSV           DE  A31
011 4 CLOSED     CNEW        NONE
012 4 CLOSED     CNEW        NONE
013 3 OPENED                 RECEIVE       CP  A64
014 4 RECEIVED               ANSWER        NE  B18
```

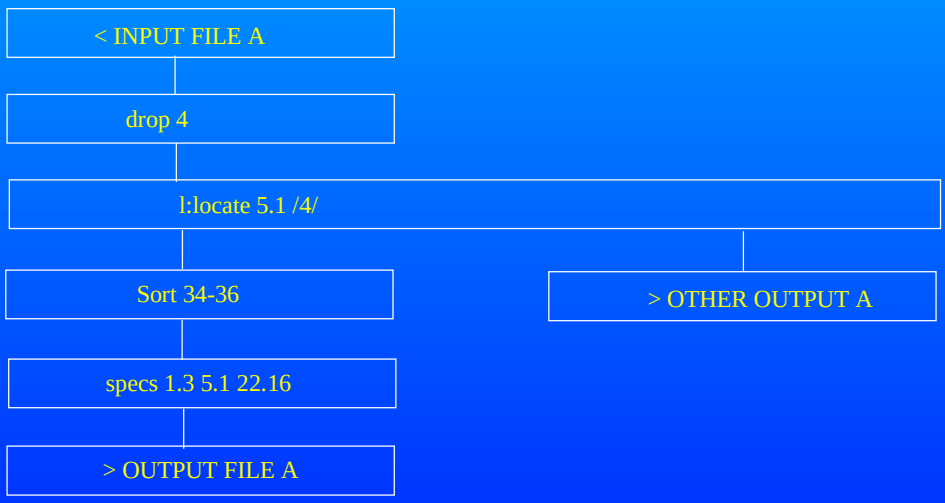
What is a Pipeline?..

- **Output:**

```
011 4 NONE
012 4 NONE
007 4 RECEIVE  PE B18
009 4 ANSWER   BE B18
014 4 ANSWER   NE B18
```

What is a Multistream Pipeline?

- **A Multistream Pipeline Set that contains multistream stages.**



- **This 2D picture needs to be represented in a 1D format**

What is a Multistream Pipeline?.... Pipelines

```
/* TEST1 EXEC                    */
'Pipe ( endchar ?)',
  '    < INPUT FILE A',          /*Read                 */
  '|  drop 4',                   /* Select              */
  '|l:locate 5.1 /4/',
  '|  sort 34-36',               /*Sort,Pick            */
  '|  specs 1.3 1 5.1 NextWord 22.16 nw',
  '|  > OUTPUT FILE A',         /*Write                */
  '?l:',
  '|  > OTHER OUTPUT A'  /*Write                */
Exit
```

- Definitions
  - Pipeline Set
  - Pipeline
  - Stage
  - Stage Separator
  - End Character
  - Device Drive
  - Filter
  - Stream
  - Label

What is a Multistream Pipeline?... Pipelines

- Multistream Stages
  - Selection Stages
  - Fanout - Copy the input record to all output streams
  - Fanin - Combines multiple input streams into one output stream in specified
  - Faninany - Combines multiple input streams into one output stream in arrival order
  - Lookup, Merge, and Collate - Combines streams
  - Overlay - Overlays streams
  - Predselect - For destructive tests
  - Synchronize - Balance two streams
  - Gate - Stop selected streams
  - Hole - One EOF from multiple streams

What is a Multistream Pipeline?... Pipelines

What is Multistream Pipeline?... Pipelines

- **RPTDG EXEC**

```
/* RPTHDG EXEC    4/24/92                */
'Pipe ( endchar ?)',
  '   < INPUT FILE A',      /* Read      */
  '|f:fanout',
  '|  drop 4',             /* Select    */
  '|  locate 5.1 /4/',
  '|  sort 34-36',         /* Sort,Pick */
  '|i:faninany',
  '|  spec 1.3 1 5.1 NextWord 22.16 nw',
  '|  > OUTPUT FILE A ',   /* Write     */
  '?f:',
  '|  take 4',             /* Keep Hdg  */
  '|i:'
```

- **Output data for RPTHDG EXEC - OUTPUT FILE A**

```
Nbr S Next     Nx Dpt
   E Action    Ac Nxt
   V To Do     ID Act
--- + ---+----3----+-
011 4 NONE
012 4 NONE
007 4 RECEIVE  PE B18
009 4 ANSWER   BE B18
014 4 ANSWER   NE B18
```

Keeping Records in Order Pipelines

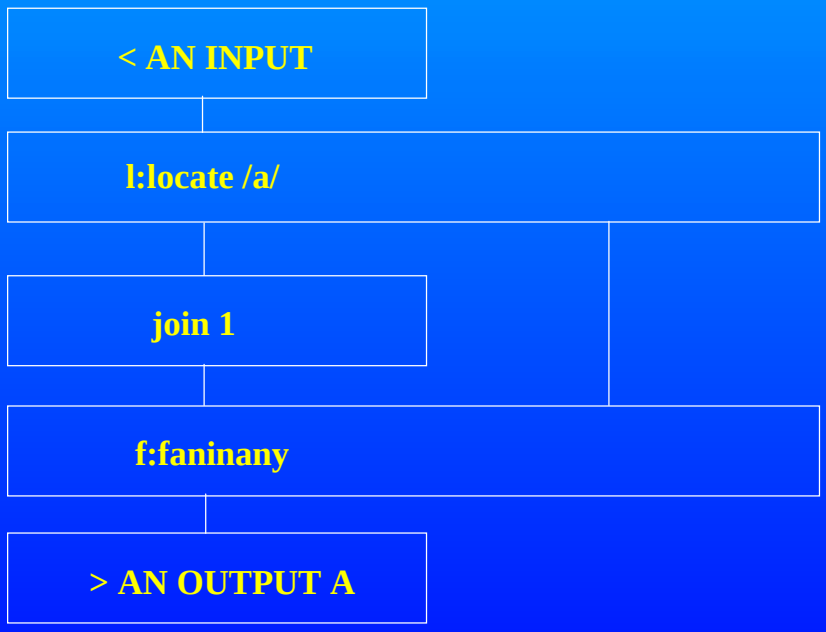
- This documentation allows the relative order of records in a set of multistream pipelines to be kept in order
- Documentation
  - The VM/ESA CMS Pipeline User's Guide documents the concepts needed to write pipelines that keep records in order
  - the first usage note for each stage indicates whether the stage delays the records or not (CMS Pipelines Reference Manual)
- Types of Stages
  - Stages that KEEP records in order
    - Don't delay the record
    - Examples - XLATE, LOCATE, and CMS
  - Stages that DO NOT KEEP records in order
    - Do delay the records
    - Examples - BUFFER and SORT
  - Stages that CAN DO EITHER
    - Depends on options
    - Examples - JOIN 1, SPECS..READ, TAKE LAST
  - Stages where this does not apply
    - Stages that have to be the first stage in a pipeline
    - Example - <.

Keeping Records in Order... Pipelines

- Concept - Any stage will execute until held up by a dispatcher call. For example a stage may be held up:
  - Waiting for a record to be available to it
  - Waiting for a record to be consumed from its OUTPUT subcommand
- Theory - To maintain the relative order of records in a set of multistream pipelines, the pipelines must:
  - Start at one common stage
  - Be split into multiple pipelines using only stages that do not delay the record
  - Contain only stages that do not delay the records
  - Be combined into a single stream using a stage that combines multiple streams as records arrive (for example, FANINANY)

Keeping Records in Order... Pipelines

- Example of How Records Get Out of Order



Keeping Records in Order... Pipelines

```
Pipe ( endchar ?)
   |  < AN INPUT
   |l:locate /a/
   |  join 1
   |f:faninany
   |  > AN OUTPUT A
   ?l:
   |f:

 AN INPUT
   a1
   b2
   a3
   b4
 AN OUTPUT
   b2
   a1a3
   b4
```

- Notice the JOIN 1 stage cause the problem because:"JOIN delays records. The last input record in each set of concatenated records is not delayed." (this is from usage note 1).
- If JOIN is replaced with XLATE the records will be kept in order since "XLATE does not delay the record."

Stages Written in REXX — Pipelines

- Obtaining Arguments
  - Arg ....(normal REXX)
- Pipes within Pipes
  - Callpipe - Subroutine pipelines
  - Addpipe - Parallel pipelines
- Reading Pipeline Records
  - Readto - Consumes the record
  - Peekto - Doesn't consume the record
  - *.input.0: - Consumes the record
- Writing Pipeline Records
  - Output - Write to pipeline
  - *.output.0: - Write to pipeline

Stages Written in REXX — Pipelines

- Stream Management
  - Select - Pick a stream
  - Sever - Get rid of a stream
  - Short - Attach two streams
  - MaxStream - How many streams?
  - StageNum- Where is my stage?
  - StreamState - What is my stream doing?
  - StreamNum - What stream am I using now?
- Other
  - Message - Issue a message
  - Rexx - invokes a REXX program
  - SetRC - sets return RC

Stages Written in REXX Pipelines

- **Msgevery REXX**

```
 /* To demonstrate MSGEVERY REXX filter */
'Pipe',
  '   < INPUT FILE A',
  ' | msgevery 3 This is an Example',
  ' | > OUTPUT FILE A'
Exit
```

MSGEVERY Console Log

```
THIS IS AN EXAMPLE        Record Number: 3
THIS IS AN EXAMPLE        Record Number: 6
THIS IS AN EXAMPLE        Record Number: 9
THIS IS AN EXAMPLE        Record Number: 12
THIS IS AN EXAMPLE        Record Number: 15
THIS IS AN EXAMPLE        Record Number: 18
THIS IS AN EXAMPLE        Record Number: 20
```

Stages Written in REXX Pipelines

- **Msgevery REXX**

```
 /*  To do a SAY for every x lines      */

arg count message
signal on error
'PEEKTO in_data'
do rcd_nbr  = 1 while rc = 0
   'OUTPUT' in_data
   if rcd_nbr // count = 0
    then say message 'Record Number:' rcd_nbr
   'READTO'
   'PEEKTO in_data'
   end /* do while rc = 0 */

error:
   say message 'Record Number:' rcd_nbr
   Exit RC*(RC<>12)
```

How Stalls Happen — Pipelines

- The FANIN way


```
Pipe ( endchar ?)
   |    < AN INPUT
   |  f:fanout
   |  i:fanin
   |    > AN OUTPUT A
   ? f:
   |  i:
```

How Stalls Happen — Pipelines

- Stall Messages

```
DMSDSO2674E     Pipelines stalled
DMSPMG2653I     ...     Issued by stage number 2 of pipeline number 1
DMSPMG2651I     ...     Running stage:      fanout
DMSDSP2675I Stage is in state wait out
DMSPMG26531     ...     Issued by stage number 1 of pipeline number 1
DMSPMG2651I     ...     Running stage:  < profile exec
DMSDSP2675I Stage is in state wait out
DMSPMG2653I     ...     Issued by stage number 2 of pipeline number 1
DMSPMG2651I     ...     Running stage:  fanout
DMSDSP2675I Stage is in state wait loc
DMSPMG265eI     ...     Issued by stage number 3 of pipeline number 1
DMSPMG2651I     ...     Running stage:      fanin
DMSDSP2675I Stage is in state wait loc
DMSPMG2653I     ...     Issued by stage number 4 of pipeline number 1
DMSPMG2651I     ...     Running stage:      > AN OUTPUT A
    2 ** 'Pipe  (endchar  ?)|      < profile exec
||f:fanout  |      i:fanin     |      > AN OUTPUT A
?f:          |      i:'
        +++ RC(-4095) +++
```

How Stalls Happen — Pipelines

- The SPECS way


```
Pipe ( endchar ?)
   |    < AN INPUT
   |  f:fanout
   |  s:specs 1-* 1 select 1 1-*  nw
   |    > AN OUTPUT A
   ? f:
   |  s:
```

How Stalls Happen — Pipelines

- Stall Messages

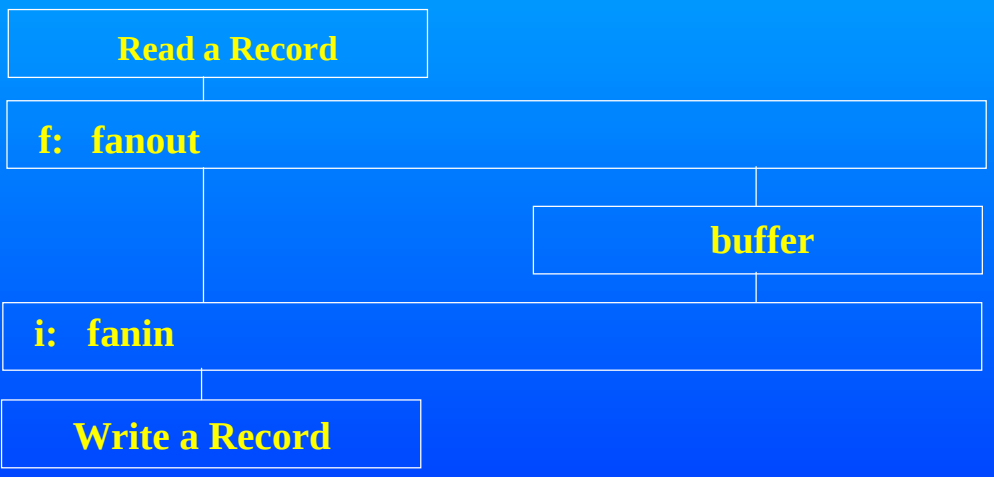
```
DMSDSO2674E      Pipelines stalled
DMSPMG2653I      ...     Issued by stage number 4 of ...
DMSPMG2651I      ...     Running stage:> AN OUTPUT A
DMSDSP2675I Stage is in state wait out
DMSPMG26531      ...     Issued by stage number 1 of ...
DMSPMG2651I      ...     Running stage:  < AN INPUT
DMSDSP2675I Stage is in state wait out
DMSPMG2653I      ...     Issued by stage number 2 ...
DMSPMG2651I      ...     Running stage:  fanout
DMSDSP2675I Stage is in state wait loc
DMSPMG265eI      ...     Issued by stage number 3 ...
DMSPMG2651I      ...     Running stage:specs 1-* 1 select 1 1-* nw
DMSDSP2675I Stage is in state wait loc
DMSPMG2653I      ...     Issued by stage number 4 of ...
DMSPMG2651I      ...     Running stage:> AN OUTPUT A
        2 ** 'Pipe  (endchar  ?)
|      < AN INPUT                          |f:fanout
|s:specs 1-* 1 select 1 1-* nw  |      > AN OUTPUT A
?f:                                       |s:'
            +++ RC(-4095) +++
```

How Stalls Happen — Pipelines

- Delay records to prevent stalls
  - ► BUFFER (with no arguments)
    - – Buffers all records until EOF received
  - ► COPY
    - – Delays one record
  - ► ELASTIC
    - – Delays as many records as required to prevent a stall

How to Prevent Stalls — Pipelines

- The FANIN solution

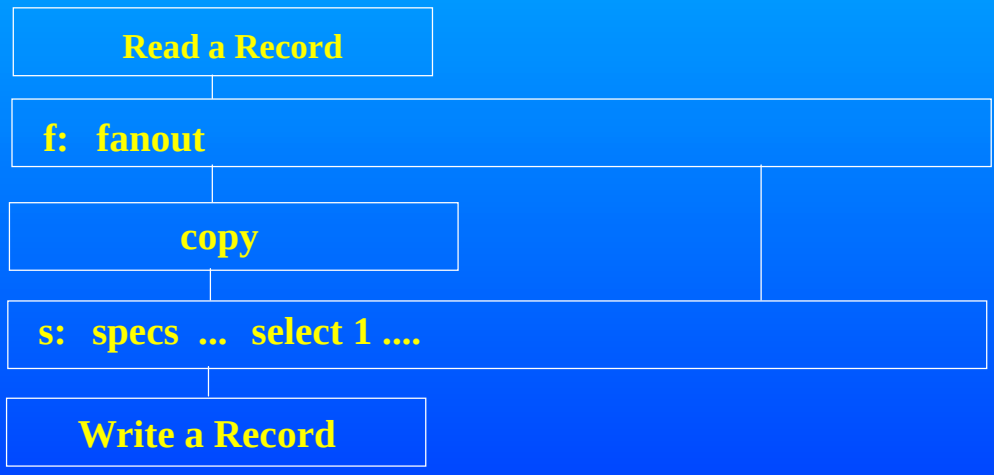



```
Pipe ( endchar ?)
   |    < AN INPUT
   |  f:fanout
   |  i:fanin
   |    > AN OUTPUT A
   ? f:
   |    buffer
   |  i:
```

- Or use ELASTIC

How to Prevent Stalls — Pipelines

- The SPECS solution




```
Pipe ( endchar ?)
   |  < AN INPUT
   |f:fanout
   |  copy
   |s:specs 1-* 1 select 1 1-* nw
   |  > AN OUTPUT A
   ? f:
   |  s:
```

- Or use ELASTIC

REXX Interface — Pipelines

- VAR
  - ► Gets/Sets a variable
- STEM
  - ► Gets/Sets a stemmed array
- REXXVARS
  - ► Gets all variables
- VARLOAD
  - ► Sets all variables

REXX Interface — Pipelines

- PRODUCER option
  - Obtains or sets REXX variables from the stage preceding it in the pipeline
  - Consider: pipe ONE | TWO | console
    - TWO can obtain or set variables in ONE
  - The following must exist:
    - Stage ONE must be blocked in output
    - stage TWO must issue the command from a CALLPIPE

For More Information — Pipelines

- CMS Pipelines User's Guide - SC24-5777
- CMS Pipelines Reference - SC24-5778
- CMS Pipelines Author's Edition - SL26-0018
- Quick Reference - SX24-5290
- Development Contacts:
  - Will J. Roden, Jr.
    - Internet: RODEN@US..IBM.COM
    - Phone: (607) 752-5065
    - Web: http://www.vm.ibm.com/devpages/roden
  - US Mail
    - IBM Department G37G
    - 1701 North Street
    - Endicott, NY 13760