



Automatic Restart Manager User Experience

GSE, Large Systems Group
Lincoln, May 2000

Paul Arnerich
Tsd (UK) Ltd/Laird Ståhl Ltd
Paul_TSD@compuserve.com or Paul@tsdd.co.uk

Agenda



- **Overview**
 - What it is and how it works
 - Invocation
 - Terms
- **Implementation**
 - Policy Definition
 - Operator Commands
 - Exits
 - Supported Applications
 - Unsupported Applications
- **User Experience**



ARM Overview



The Problem

- In a highly available environment, we need fast and efficient restart in case of:
 - Critical application failure
 - System loss involving critical applications
- Automation and Scheduling software is a start
- But what if Automation fails ?
- What about cross system tasks ?
- How can Automation track system components ?



ARM - The Solution ?

- Automatic Restart Manager
- Delivered by base OS/390 software
- Requires Monoplex at a minimum
- Supports Basic and Parallel Sysplex
- Requires a Policy dataset and an active Policy
- Code executes in XCFAS
- Some new Operator commands



ARM Does.... (1)

- ...get notified when STC End of Memory Resource Manager (EOM) is invoked
- ...get notified when JOB End of Job Resource Manager (EOJ) is invoked
- ...get notified when XCF 'sysgone' is triggered
- ...therefore know when an applications has terminated unexpectedly
- ...trigger automatic restart of failed application:
 - on same system if Element Failure
 - on alternate system if sysgone event



ARM Does.... (2)

- ...query WLM to make informed decision for restart candidate system
- ...cater for Application priority for restart
- ...cater for Application grouping for restart on alternate system to honour affinities
- ...allow data centre control via Exit invocation



ARM Does Not....

-provide transaction or database recovery
 - That is the Applications responsibility
- ...initially start Applications
- ...need Operator intervention
- ...replace Automation
- ...magically produce needed Capacity (MIPS and CSA) for restart



Terms etc. (1)

■ IXCARM

→ authorised assembler service to request ARM services

■ Element

→ an ARM registered Application

■ Registration

→ The IXCARM macro option which requests ARM to monitor an elements health/existence

■ Restart_Group

→ logically related group of elements for restart purposes

→ e.g. 1 CICS TOR and 3 CICS AOR's



Terms etc. (2)

■ Element State

→ one of five states an element can exist in

■ Restart_Method

→ The command issued when restart is required

→ Usually same as original element start

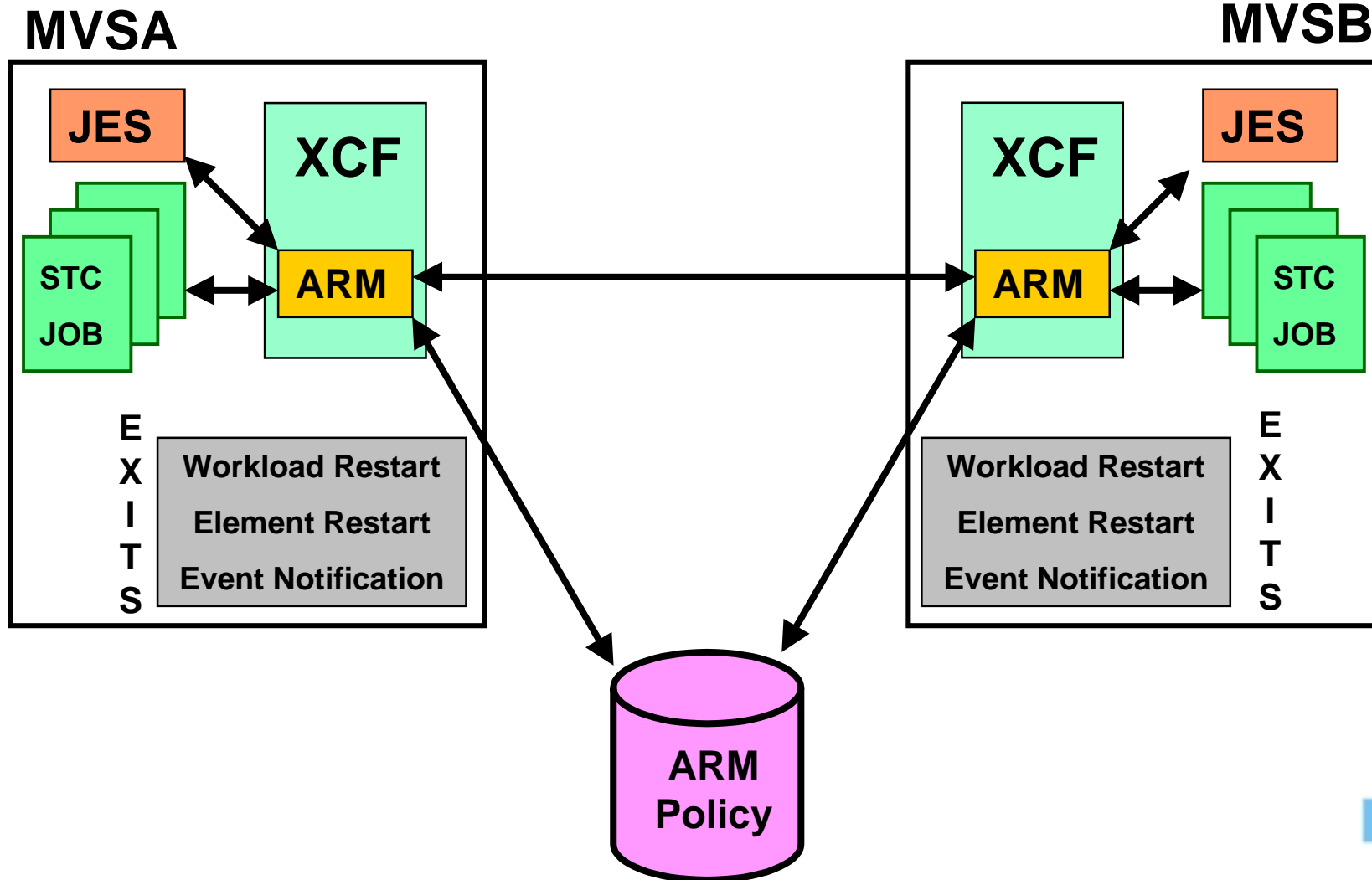
→ Can be completely different

→ PROC to Start or JCL to submit

→ Can be different for 'sysgone' or Element Failure



How it works (1)



How it works (2)

- An ARM enabled element 'registers' with ARM
- ARM invoked by EOJ/EOM Resource Managers and XCF 'sysgone' condition
- If Element Failure :
 - consults policy for this element
 - restarts according to policy RESTART_METHOD
 - Yes or No
 - Which JCL/PROC to use
 - restarts failed element on same system



How it works (3)

■ If 'sysgone':

- consults policy for all registered elements on failed system
- restarts according to policy RESTART_METHOD
 - Yes or No
 - Which JCL/PROC to use
- consults WLM for most appropriate system
- honours RESTART_GROUPS
- honours RESTART_ORDER
- restarts failed element on desired system

■ Then ARM watches for re-registration



IXCARM Services

- Communication to ARM is via IXCARM Macro
- Authorised Assembler Service
- MVS Programming: Sysplex Services Guide
 - GC28-1771 Chapter 3
- MVS Programming: Sysplex Services Reference
 - GC28-1772 ref. IXCARM
- Caller must be authorised and have no locks
- Element communicates with ARM (XCFAS) via cross memory services
- ARM is an XCF group and signalling user
- Macro has five basic invocations
- Macro parms can override Policy



IXCARM MACRO

- **REQUEST(REGISTER)**
 - request ARM monitoring and restart in case of failure
- **REQUEST(READY)**
 - Element is ready to accept new work
- **REQUEST(WAITPRED)**
 - Element is waiting for another element to be READY
- **REQUEST(ASSOCIATE)**
 - Element is a backup for another task
 - When other task fails, if this element is READY, no restart will be performed
- **REQUEST(DEREGISTER)**
 - Usually for normal termination



Element States

■ STARTING

- From initial registration until element indicates READY
- To get 'READY' element must issue IXCARM again

■ AVAILABLE

- From element READY until Deregister or termination

■ FAILED

- From ARM detected termination until ARM issues restart

■ RESTARTING

- From ARM restart until element re-REGISTERS

■ RECOVERING

- From re-REGISTER after ARM restart, until element issues READY



Restart Process (1)

- ARM does not clean up elements environment
- Clean up may need to be performed by an Exit
- At restart, ARM loses track of element, so waits for re-registration
- If RESTART_TIMEOUT is exceeded:
 - Issue ENF
 - Issue Error Message
 - Clean up ARM environment
 - De-register element



Restart Process (2)

- ARM waits for READY state based on TIMEOUT_INTERVAL
 - If exceeded, ARM issues a Warning Message only
- RESTART_ATTEMPTS value protects against multiple failures
 - e.g. JCL errors, Parameter errors, Capacity problems



Element Failure Flow

- Application termination
- RTM is invoked
- RTM invokes EOM/EOJ Resource Manager
- EOM/EOJ RM notifies ARM
- ARM then:
 - Invokes Exits if any
 - Determines if Element is restartable
 - Executes RESTART_METHOD on same system
 - Waits for re-REGISTER

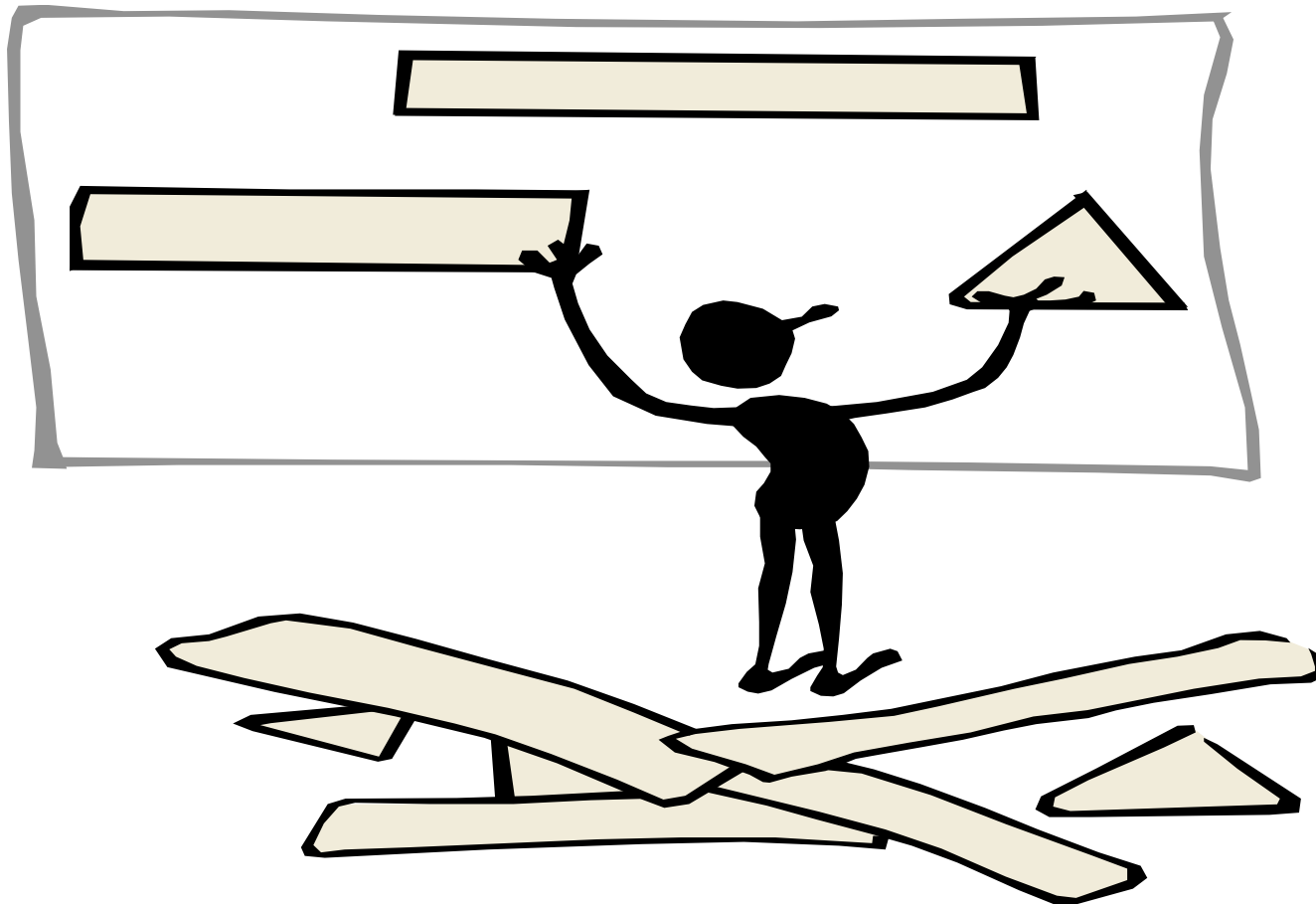


'sysgone' Failure Flow

- System Terminates
- An XCF in plex detects SSUM condition
- Failing system is removed from the plex
- XCF notifies ARM on detecting MVS of failure
- ARM on that MVS supervisors restart
- ARM then:
 - determines if any registered elements are involved
 - reads Policy to determine eligibility for restart
 - determines which elements are 'grouped'
 - queries WLM to identify candidates for group restarts
 - honours Policy for target system eligibility
 - restarts groups as appropriate
 - NB: all elements are 'grouped' even single elements



Implementation



Implementation Steps

- Create Policy dataset
- Define Policy
- Restart Applications in order to register
 - there are some smart applications
- Test and Document
- Inform Operations Staff



Policy Dataset



- Create with IXCL1DSU
- Program resides in SYS1.MIGLIB
- Not particularly performance critical
- High availability requirement
- Practical to implement an alternative (x2)
- To size, do the maths on numbers of elements, then double it



Policy Definitions (1)

■ ELEMENT

- name coded on IXCARM macro for registration
- decided by programmer
- A to I and SYS reserved for IBM

■ ELEMENT_NAME

- actual task name (wildcard '?' and '*' are supported)
- helps to identify each task

■ TARGET_SYSTEM & FREE_CSA

- List candidate systems/CSA required for restart

■ RESTART_PACING

- delay in seconds between each element restart

■ TERMTYPE

- ELEMTERM - restart only if Element failure
- ALLTERM - restart for both Element failure and 'sysgone'



Policy Definitions (2)

■ RESTART_METHOD

- Coded on IXCARM call or in Policy
- Can do nothing
- Can trigger same JCL/PROC to be used as original element
- Can have different JCL/PROC from original element
- Can have different JCL/PROC for Element Failure than for 'sysgone' NB: This is crucial
- Symbolic substitution resolved as for system where element first registered

■ e.g. RESTART_METHOD

- RESTART_METHOD(BOTH,PERSIST)
- RESTART_METHOD(ELEMTERM,PERSIST)
- RESTART_METHOD(SYSTEM,STC,'S PANIC,P1=OS HIT')



Policy Definitions (3)

■ RESTART_GROUP

- All elements must belong to a group, even if it is just one element
- You choose the group name
- All elements in a group are restarted on the same system

■ Default processing is crucial

- To override the defaults, code `RESTART_GROUP(*)`
- all subsequent parms until next GROUP override the default for everything.
- That is, if you do not specify an operand on a GROUP, it will still be overridden
- There is also a default GROUP, which is any element not specifically identified in Policy
- This may also have separate values overridden



ARM Exits

■ Event Exit

- Specified on IXCARM Register
- Module must be LPA or LNKLST resident and Authorised
- Invoked prior to restart and during registration

■ Workload Restart Exit

- dynamic exit support
- invoked on systems where workload is to be restarted
- invoked prior to restart
- use to cancel unimportant work to free resources ?

■ Element Restart

- dynamic exit support
- invoked prior to element restart
- may be provided by application supplier to do clean up



Operator Commands

- D XCF,ARMSTATUS | ARMS
- D XCF,ARMS,RESTARTGRP | RG
- D XCF,ARMS,ELEMENT | EL=eIname
- D XCF,ARMS,JOBNAME | JOB=jobname
- D XCF,ARMS,STATE=
START | AVAIL | FAILED | RESTART | RECOVER
- D XCF,ARMS,DETAIL
- CANCEL jobname,ARMRESTART
- FORCE jobname,ARM,ARMRESTART



ARM Enabled Applications

- List is growing with each OS/390 release
- CICS Transaction Server 1.1 +
- CICS/ESA 4.1
- CP/SM 1.1.1 +
- DB2 4.1 +
- IMS TM 5.1 +
- IMS/DBCTL 5.1 +
- VTAM 4.3 +
- TME 10 Netview for OS/390 (SAFO ?)
- IRLM (with APARs PQ06465 and OW28526)
- MQ Serious 2.3 +
- TCPIP w/ OS/390 r8 ?



The 'B' List

- What about applications that are not ARM enabled ?
- Obtain the source and insert ARM Macros as appropriate

or

- Apply pressure to Vendor

or

- Use the 'Wrapper' !



The Wrapper

- Module ARMWRAP, IBM supplied Usermod
- Available via MKTTOOLS, see your rep
- Documented in SA presentation at:
 - www9.s390.ibm.com/products/sa/saparsys/sapsc.htm
- SMP/E installed - unsure about support
- Insert the ARMWRAP program as 'top and tail' in JCL/PROC for critical applications
- First invocation will Register (Parm driven)
- Later invocation Deregisters (Parm driven)
- 20 minutes to install and test
- Now supports unauthorised invocation
- It does what it says on the box !



User Experience



User Experience

- 2 way Parallel Sysplex
- CTS but no CP/SM
- DB2 Data Sharing
- MQ Serious
- CTL-O for automation
- All connectivity through TCP/IP
- Not truly parallel due to core application design



Operational Infrastructure

- Greenfield site
- UNISYS to OS/390 conversion
- One Operator per shift
- Only day shift supported
- Minimal automation in place
- Minimal automation skills
- Need highly automated failover capability
- ARM provided the necessary benefits easily



Policy (1)

```
DATA TYPE(ARM) REPORT(YES)
  DEFINE POLICY NAME(POL1) REPLACE(YES)
    /* CONTROLO */
    RESTART_GROUP(CNTO)
      ELEMENT(TSG_CTLO)
        RESTART_ATTEMPTS(2,120)
        RESTART_TIMEOUT(10)
        READY_TIMEOUT(180)
        TERMTYPE(ALLTERM)
        RESTART_METHOD(BOTH,PERSIST)
    /* JUST VTAM */
    RESTART_GROUP(VTAM)
      ELEMENT(NET@*)
        RESTART_ATTEMPTS(2,120)
        RESTART_TIMEOUT(30)
        READY_TIMEOUT(180)
        TERMTYPE(ALLTERM)
        RESTART_METHOD(ELEMTERM,PERSIST)
        RESTART_METHOD(SYSTEM,STC,'S MQRESTART,M=MQCONN')
```



Policy (2)

```
/* DB2          */
  RESTART_GROUP(DB2)
  ELEMENT(DSNDBP*)
  RESTART_ATTEMPTS(2,300)
  RESTART_TIMEOUT(60)
  READY_TIMEOUT(180)
  TERMTYPE(ALLTERM)
  RESTART_METHOD(BOTH,PERSIST)
/* CICS P1     */
  RESTART_GROUP(CICSP1)
  ELEMENT(SYSCICS_CICSP1*)
  RESTART_ATTEMPTS(2,120)
  RESTART_TIMEOUT(30)
  READY_TIMEOUT(180)
  TERMTYPE(ALLTERM)
  RESTART_METHOD(BOTH,PERSIST)
```



Issues

- MQ 2.2 not enabled
- Tried ARMWRAP but MQ startup failed SC03
- CTL-O can monitor Element failure, but not 'sysgone'
- MQ running in standby mode other system
- Standby MQ needs CICS Connection Open
-hang on ! How about VTAM ?
- VTAM needs Element restart for Element failure
- But VTAM has no restart if 'sysgone'



Solution

- Why not use VTAM 'sysgone' to indicate MQ 'sysgone' ?
- RESTART_METHOD for VTAM ELEMTERM is persistent restart
- But for SYSTEM is a STC driven CICS Connection Open for MQ
- ARMWRAP used for CTL-O



Additional Reading/Topics

- **Setting up a Sysplex**
 - GC28-1779
 - Chapter 7.5 and Appendix B.2.1 for definitions
- **Automation for S/390 Parallel Sysplex**
 - SG24-4549
 - Chapter 3.5
- **Predecessor Processing**
- **Restart Grouping**
- **Restart Pacing**
- **In flight transaction recovery**



Questions

