Applies to D1 and later machines

Expired Operator Password:

[] Run !!GATOR!Local.Operator_Password, using cluster ID, and date to generate the authorization codes for Operator_Password. Be sure to use the day when the authorization code will be used. Default is today's date. The authorization code will only be in effect until midnight of the date specified.

[] When you/they login use the old operator password for password, and the Operator_Password authorization code when it asks for the override authorization code.

BE SURE TO USE UPPER CASE FOR THE LETTERS IN THE CODE!

example:

```
Enter user name: operator
Enter password: <old password>

Operator password has expired.  Call Rational for override authorization
code.
Enter authorization code: 1FF79C650C12E8B   -- UPPER CASE
```

[] Change the operator password using Op.Change_Password.

Forgotten Operator Password:

[] Run !!GATOR!Local.[redacted] using cluster ID, and date to generate the authorization codes for Change_Password. Be sure to use the day when the authorization code will be used. Default is today's date. The authorization code will only be in effect until midnight of the date specified.

[] Log into any account

[] If no accounts are known then

```
EEDB: e ed
      *** If, at this point, you get a message
            "A subsystem with name IMAGE (or
             something else) is already elaborated"
          see the bottom of this tip
ed: x Ac_Off  -- only need to do this if operator access is restricted
ed: x create_user
  Password: xyz
  User: xyz
```

[] Login at environment terminal

[] change_password

```
Op.Change_Password (User => ">>USER NAME<<",
                    Old_Password => "",  -- enter auth. code in UPPER CASE
                    New_Password => "",
                    Response => "<PROFILE>");
```

[] Don't forget to delete user if one was created.

```
ed: X Ac_On
ed: quit
EEDB: u ed
Subsystem:     -- just hit return
```

```
      Log onto operator account and delete user -
         Op.Delete_User (User => "XYZ", Response => "<PROFILE>");

[] Done!
```

*** Help promised above

If you are having trouble with the "e ed" command telling you something is already elaborated, you are the victim of a Rational messup. The contents of some of the configurations don't match those of the main configuration. You probably have one of the interim releases like 12_6_6 or 12_6_7.

Let's say you have 12_6_7. Here's what to do. Below, everything in front of a colon is a prompt, everything after is user supplied.

```
   EEDB: vd $
```

Look for the name of the subsystem which was mentioned in the "already elaborated" message. In this case, that's IMAGE. Note the version number immediately following its name. Write it down. Do the same for everything appearing above it in the list but below the subsystem that you were trying to elaborate. In this case, that would be ED_TESTS.

Next, look at the contents of the smaller configuration that you were just trying to elaborate. In this case, that's ed.

```
   EEDB: vd ed
```

Look for the same subsystems and their version numbers. Something doesn't match, right? That's the problem.

```
   EEDB: build_configuration
```

We're going to build a new configuration. We can either build a brand new one and use that or if we're brave, we can rebuild the existing one. We'll be cowards.

```
   New Configuration: ed2
```

We now want a model, so we supply the correct configuration, d_12_6_7.

```
   Existing Configuration: d_12_6_7
```

We want to specify the subsystem which is just below the one we are building. For ED_TESTS, that's OBJECT_EDITOR. I think this means that we will take everything up to and including OBJECT_EDITOR from d_12_6_7, which is what we want.

```
   Parent Subsystem: object_editor
```

Now, we want to specify the subsystem and version of that subsystem that we want at the top of this configuration. That would be ED_TESTS in this case.

```
   Subsystem.Version: ed_tests.10.0.0d
```

The system will keep asking for other subsystems. Just hit RETURN to exit.

```
   Subsystem.Version:
```

Now, when you look at the new configuration, you should see a perfect match between the parts from IMAGE to ED_TESTS.

```
   EEDB: vd ed2
   EEDB: vd d_12_6_7
```

*** Now go back to the original procedure above and retry the "e ed" making sure to use "ed2" if that's the name of the new configuration you just created.

:REFERENCES:
DEG, SMP 29-JUN-90, JGP 02-APR-90 GBD Csr2219
Thu, 4 Apr 91 18:10 PDT DEG Csr 4378

TCP/IP 2000 DTIA -- Rational DTIA Remote_Operations server

```
-- This file defines services accessible via Rational Networking.
-- Each non-empty line contains
--    * a Transport_Defs.Network_Name,
--    * a Transport_Defs.Socket_Id,
--    * a service name, and
--    * optionally a machine name;
-- in that order, separated by white space (blanks or tab characters).
-- Characters from "--" to the end of the line are a comment.  A
-- Network_Name, Socket_Id, service name or machine name must not
-- contain any white space or the character sequence "--".  The
-- Transport_Defs.Socket_Id is in decimal dotted notation; that is,
-- a sequence of non-negative decimal integers separated by periods.
-- Each integer represents one or more bytes, as minimally required to
-- represent that integer in binary form, most significant byte first.
-- For example, 258 and 1.2 represent the same Socket_Id, which is
-- expressed in Ada as (1,2).  A service name or a machine name should
-- be a valid Ada identifier.  The case (upper or lower) of a service
-- name or a machine name is not significant.

-- A line signifies that the named service is available via the given
-- network at the given socket in the given machine, specifically;
-- if no machine name is given, the line signifies that the named service
-- is available via the given network at the given socket in all machines
-- not specifically named in other lines.

-- Some (perhaps most) services are defined here for reference only:
-- their network and socket cannot be changed by modifying this file.
```

What is DTIA?

DTIA (Distributed Tools Integrated Architecture) is a Rational utility similar to the Unix remote shell (rsh) mechanisms. Via a network connection, the DTIA client on a Rational Environment passes strings that represent commands or programs to the DTIA server running on a workstation. The DTIA server simply executes the commands or programs and transfers the standard output and error messages generated by these programs back to the client on the Rational Environment.

As a prerequisite for RPI and RTI, you need to order and install DTIA as a separate product.

The RCI base product links in DTIA, so all RCI extensions do so. But only the RS/6000 AIX extension actually uses DTIA. Even then, the DTIA code is linked into the RCI's load proc so there is no need to order and install DTIA separately. There should be no conflict if RCI is installed on a Rational Environment that already has DTIA installed.

The default RCI extension uses FTP and Telnet for communication with the target. Since DTIA is linked in, it is also available but we strongly recommend using FTP/Telnet instead of DTIA. There was a time (in particular, Geoff Fitch's comment in a *Sheep* article, July 1991, pages 19-20) when DTIA was preferred, but our approach changed since then. Now DTIA should be used only when it is the only option.

Rational's distributed tools integration architecture (DTIA) is the foundation of the Publishing Interface. DTIA extends the reach of the Rational Publishing Interface by enabling the management of Interleaf information from the Ratonal Environment. The Rational CMVC interface can control Interleaf documents as well as information on the R1000 and from other workstation tools. DTIA automates information sharing between the workstations and R1000s on the network and ensures consistency of documents. -- from data sheet (?)

[Source: JAK, etc.]

:Reference: also see:
!DOCUMENTATION.RC_INFO.Software.Environment.LOGIN_PROBLEMS.
NO_ONE_CAN_LOGIN_WITH_SOME_OR_ALL_TERMINAL_TYPES

:Originated From:

:Revised By: RIP 920918

TOPIC: Promoting an ADA units using CG_DIR_TESTS.

Introduction:

If you can't login into a system and you need to promote an object such as !Machine.Initialize or the >>Terminal_Type<<_commands procedures you can use DT.

Instructions.

1) Elaborate DT at the EEDB. If it doesn't elaborate you will probably have to rebuild it.

```
EEDB: e dt
CG_DIR_TESTS.9.0.0D            5/29/87 16:52:21
CG_DIR:
```

2) Start the CDIR_TEST Program

```
CG_DIR: run cdir
CDIR_TESTER started

====>> DIR_TEST <<====
Type argument (@argument for indirect):
```

3) Turn off Access Control.

```
---- dirtest directory tester.
Type argument (@argument for indirect): /disable_access_control
Type argument (@argument for indirect):
```

4) Enter the Universe.

```
Type argument (@argument for indirect): /
COMP_UNIT>
```

5) Go to the unit that you wish to promote.
*** Type only the part of the command in CAPS. The rest is displayed as part of electric completion. The Portion in the () are displayed by the program.

```
COMP_UNIT> G(oto )D(ecl named: )!MACHINE.INITIALIZE'BODY
procedure INITIALIZE is separate;
```

6) Promote the unit.
*** Type only the part of the command in CAPS. The rest is displayed as part of electric completion.

```
SUBPROGRAM_BODY> M(ake )P(romotion)
goal state = C(oded )
    100  instructions for subprog INITIALIZE
```

```
    434  instructions for segment 147712
results: SUCCESSFUL

7) Exit the Universe

SUBPROGRAM_BODY> Q(uit)

8)  Turn on Access Control

Type argument (@argument for indirect): /enable_access_control

9) Exit

Type argument (@argument for indirect): /q
==== dirtest PASSED ====================
     Total passed =  1

=====>> Elaborator Database <<=====
CG_DIR: CDIR_TESTER finished

CG_DIR: quit

9) Unelaborate dt

EEDB: un dt
Subsystem: <RET>
Unelaborated CG_DIR_TESTS.9.0.0D
EEDB:
```

```
with Io;
with Job;
with Machine;
with Directory;
with Scheduler;
with Disk_Daemon;
with System_Utilities;
with String_Utilities;

procedure Job_Killer is
    I, O : Io.File_Type;
    Console : constant String := "!machine.devices.terminal_1";

    Prompt : constant String := "Job To Kill: ";

    type User_Job is new Natural range 6 .. 255;

    Job_Numbers : constant String := User_Job'Image (User_Job'First) & " .. " &
                                     User_Job'Image (User_Job'Last);

    procedure Do_Kill (Job_Id : Machine.Job_Id;
                       Session_Id : System_Utilities.Session_Id;
                       Job_Image : String;
                       Job_Name : String;
                       User_Name : String;
                       User_Session : String) is

        Kind : Scheduler.Job_Kind := Scheduler.Get_Job_Kind (Job_Id);

        The_Session_Object : Directory.Object :=
           System_Utilities.Session (Session_Id);
        The_Session : constant String :=
           Directory.Naming.Get_Full_Name (The_Session_Object);
    begin
        if String_Utilities.Equal (User_Name, "*system") then
            Io.Put_Line (O, "Can't kill a " & User_Name & " job");
        else
            case Kind is

                when Scheduler.Attached | Scheduler.Detached =>
                    Io.Put_Line (O, "Killing user job" & Job_Image &
                                    ", " & User_Name & ", " &
                                    The_Session & ", " & Job_Name);
                    Job.Kill (Job_Id, The_Session);
                    Job.Enable (Job_Id, The_Session);

                when others =>
                    Io.Put_Line (O,
                                 "Can't kill a " &
                                    Scheduler.Job_Kind'Image (Kind) & " job.");
            end case;
        end if;
    end Do_Kill;


    procedure Do_Killing (J : String) is
        Job_Number : User_Job;
        Job_Id : Machine.Job_Id;

        function Intent_Confirmed (J : String) return Boolean is
```

```
            Input : String (1 .. 128);
        begin
            Io.Put (O, "Do you really want to kill job " & J & " [N]?");
            declare
                Input : constant String := Io.Get_Line (I);
            begin
                if Input = "Y" or Input = "y" or
                   Input = "YES" or Input = "yes" then
                    return True;
                else
                    return False;
                end if;
            end;
        end Intent_Confirmed;
    begin
        if Intent_Confirmed (J) then

            Job_Number := User_Job'Value (J);
            Job_Id := Machine.Job_Id (Job_Number);

            if Job_Number not in User_Job then
                raise Constraint_Error;
            end if;

            Do_Kill (Job_Id, System_Utilities.Get_Session (Job_Id),
                     Machine.Job_Id'Image (Job_Id),
                     System_Utilities.Job_Name (Job_Id),
                     System_Utilities.User_Name
                        (System_Utilities.Get_Session (Job_Id)),
                     System_Utilities.Session_Name
                        (System_Utilities.Get_Session (Job_Id)));
        else
            Io.Put_Line (O, "Killing of job cancelled");
        end if;
    exception
        when Constraint_Error =>
            Io.Put_Line (O, J & " is not a job number in the range " &
                            Job_Numbers);
        when others =>
            Io.Put_Line (O, Machine.Job_Id'Image (Job_Id) &
                            " isn't an active job.");
    end Do_Killing;

begin
    Scheduler.Set_Job_Attribute (System_Utilities.Get_Job, "Kind", "Server");
    Disk_Daemon.Set_Prevent_Stop_By_Warning (True);

    Io.Open (O, Io.Out_File, Console);
    Io.Open (I, Io.In_File, Console);

    loop
        Io.Put (O, Prompt);

        Do_Killing (Io.Get_Line (I));
    end loop;

    Io.Close (I);
    Io.Close (O);
end Job_Killer;
```

```
with Activity;
with Io;
with Log;
with Operator;
with Scheduler;
with Debug_Tools;
with Library;
with Program;
with Time_Utilities;
with Error_Reporting;
with Duration_Until_Next;
with Start_Fth;
with Start_Ernie;
with Start_Physical;
with Profile;
with Daemon;
with Queue;
with Operator;
with Initialize_Daemons;
procedure Initialize_Site is

    Daily_Daemons_Start_Hour : constant Time_Utilities.Military_Hours := 1;

    Job_Profile_Image : constant String :=
       "WIDTH=79, ~:::, Activity=!MACHINE.RELEASE.CURRENT.ACTIVITY, <DEFAULT>";
    Job_Profile : Profile.Response_Profile := Profile.Value (Job_Profile_Image);

begin

--================================================================
-- Overwrite some default scheduler settings to allow better response
-- for editor jobs:
    -- GBD commented-out this stuff 22 Jun 90 and again 10 Apr 91
    -- experimentally.
    -- Scheduler.Set ("Min_CE_Wsl", 250);
    -- Scheduler.Set ("Max_CE_Wsl", 1000);
    -- Scheduler.Set ("Min_OE_Wsl", 150);
    -- Scheduler.Set ("Max_OE_Wsl", 1500);
    -- Scheduler.Set ("Min_Server_Wsl", 50);
--================================================================

    Operator.Set_Password_Policy (Minimum_Length => 6, Change_Warning => 360);

    Library.Delete ("!Machine.Sims.To_Rational.Internal_Spr.@",
                    Response => "<ERRORS>");
    Library.Compact_Library ("!Machine.Sims.To_Rational.Internal_Spr");
    Library.Delete ("!Machine.Sims.To_Rational.Internal_Processing.@",
                    Response => "<ERRORS>");
    Library.Compact_Library ("!Machine.Sims.To_Rational.Internal_Processing");
    Library.Compact_Library ("!Machine.Sims.Reports");


    Dbms_Server:
        begin
            Program.Run (S        =>
                            """!Projects.DBMS'spec_view.Units"".Start_DBMS (" &
                               """!Machine.Databases.@_DB"");",
                         Context  => "!Machine.Error_Logs",
                         Response => "<VERBOSE>");
        exception
```

```
        when others =>
            Error_Reporting.Report_Error
               (Caller      =>
                   "!Machine.Initialization.Local.Initialize_Site.DB_Server"
                Reason      =>
                   Error_Reporting.Create_Condition_Name
                      ("Unhandled_Exception", Error_Reporting.Problem),
                Explanation => Debug_Tools.Get_Exception_Name (True, True));
    end Dbms_Server;

-- Disabled 01/23/95 by RJG:
-- Cluster_Server:
--     begin
--         Program.Run_Job (S =>
--                             """!Users.operator.Servers"".Cluster_Db_Serve
--                          After => Time_Utilities.Duration_Until_Next (6,
--                          Context => "!Machine.Error_Logs",
--                          Options => "Name = (Cluster_DB_Server)",
--                          Response => "<VERBOSE>");
--     exception
--         when others =>
--             Error_Reporting.Report_Error
--                (Caller      =>
--                    "!Machine.Initialization.Local.Initialize_Site.Cluster
--                 Reason      =>
--                    Error_Reporting.Create_Condition_Name
--                       ("Unhandled_Exception", Error_Reporting.Problem),
--                 Explanation => Debug_Tools.Get_Exception_Name (True, True
--     end Cluster_Server;
--
--
Dbms_Compaction_Server:
    declare
        Day         : constant String := "Saturday";
        H           : constant Time_Utilities.Military_Hours := 18;
        M           : constant Time_Utilities.Minutes := 0;
        After_Delay : constant Duration := Duration_Until_Next (Day, H, M);
    begin
        Program.Run_Job
           (S => """!Users.operator.Servers"".DBMS_Compaction_Server(" &
                    '"' & Day & """, " &
                    Time_Utilities.Military_Hours'Image (H) & ',' &
                    Time_Utilities.Minutes'Image (M) & ");",
            Context => "!Machine.Error_Logs",
            Options => "Name = (DBMS Compaction Daemon)",
            After => After_Delay);
    exception
        when others =>
            Error_Reporting.Report_Error
               (Caller      =>
                   "!Machine.Initialization.Local.Initialize_Site.DBMS_Compa
                Reason      =>
                   Error_Reporting.Create_Condition_Name
                      ("Unhandled_Exception", Error_Reporting.Problem),
                Explanation => Debug_Tools.Get_Exception_Name (True, True));
    end Dbms_Compaction_Server;


-- Pm_Daemon:
--     declare
```

```
--         Day         : constant String := "Thursday";
--         H           : constant Time_Utilities.Military_Hours := 6;
--         M           : constant Time_Utilities.Minutes := 0;
--         After_Delay : constant Duration := Duration_Until_Next (Day, H, M
--     begin
--         Program.Run_Job
--            (S => """!Users.operator.Servers"".PM_Daemon(" & '"' & Day &
--                  """, " & Time_Utilities.Military_Hours'Image (H) &
--                  ',' & Time_Utilities.Minutes'Image (M) & ");",
--             Context => "!Machine.Error_Logs",
--             Options => "Name = (PM Daemon)",
--             After => After_Delay);
--     exception
--         when others =>
--             Error_Reporting.Report_Error
--                (Caller      =>
--                    "!Machine.Initialization.Local.Initialize_Site.PM_Daem
--                 Reason      =>
--                    Error_Reporting.Create_Condition_Name
--                       ("Unhandled_Exception", Error_Reporting.Problem),
--                 Explanation => Debug_Tools.Get_Exception_Name (True, True
--     end Pm_Daemon;
--
--
-- Raps_Prs_Server:
--     declare
--         H           : constant Time_Utilities.Military_Hours := 0;
--         M           : constant Time_Utilities.Minutes        := 1;
--         After_Delay : constant Duration                      :=
--            Time_Utilities.Duration_Until_Next (H, M);
--     begin
--         Program.Run_Job
--            (S       => """!users.operator.Servers"".RAPS_PRS_Server(" &
--                           Time_Utilities.Military_Hours'Image (H) &
--                           ',' & Time_Utilities.Minutes'Image (M) & ");",
--             Context => "!Machine.Error_Logs",
--             Options => "Name = (RAPS_PRS_Server)",
--             After   => After_Delay);
--     exception
--         when others =>
--             Error_Reporting.Report_Error
--                (Caller      =>
--                    "!Machine.Initialization.Local.Initialize_Site.RAPS_PR
--                 Reason      =>
--                    Error_Reporting.Create_Condition_Name
--                       ("Unhandled_Exception", Error_Reporting.Problem),
--                 Explanation => Debug_Tools.Get_Exception_Name (True, True
--     end Raps_Prs_Server;
--
--
-- Raps_Report_Server:
--     declare
--         H           : constant Time_Utilities.Military_Hours := 5;
--         M           : constant Time_Utilities.Minutes        := 0;
--         After_Delay : constant Duration                      :=
--            Time_Utilities.Duration_Until_Next (H, M);
--     begin
--         Program.Run_Job
--            (S       => """!users.operator.Servers"".RAPS_Report_Server("
--                           Time_Utilities.Military_Hours'Image (H) &
```

```
--                              ',' & Time_Utilities.Minutes'Image (M) & ");",
--             Context => "!Machine.Error_Logs",
--             Options => "Name = (RAPS_Report_Server)",
--             After   => After_Delay);
--     exception
--         when others =>
--             Error_Reporting.Report_Error
--                (Caller      =>
--                    "!Machine.Initialization.Local.Initialize_Site.RAPS_Re
--                 Reason      =>
--                    Error_Reporting.Create_Condition_Name
--                       ("Unhandled_Exception", Error_Reporting.Problem),
--                 Explanation => Debug_Tools.Get_Exception_Name (True, True
--     end Raps_Report_Server;


Reboot_Server:
    begin
        Program.Run_Job
           (S       => Program.Current
                          ("!Users.operator.Reboot_Server",
                           "Reboot_Server.Server", "",
                           Activity => "!Machine.Release.Current.Activity"),
            Debug   => False,
            Context => "!Machine.Error_Logs",
            After   => 15 * 60.0,
            Options =>
               "Output=!Machine.Error_Logs.Reboot_Server_Output" &
                  ", Error => !Machine.Error_Logs.Reboot_Server_Error" &
                  ", Name=(Reboot_Server)");
    end Reboot_Server;


Job_Killer:
    begin
        Program.Run_Job
           (S       => """!Users.operator.Servers"".Job_Killer",
            Debug   => False,
            Context => "!Machine.Error_Logs",
            After   => 15 * 60.0,
            Options =>
               "Output=!Machine.Error_Logs.Job_Killer_Output" &
                  ", Error => !Machine.Error_Logs.Job_Killer_Error" &
                  ", Name=(Job_Killer)");
    end Job_Killer;


Fth:
    begin
        Start_Physical;
        delay 10.0;
        Start_Fth;
        Start_Ernie;
    exception
        when others =>
            null;
    end Fth;

-- Rcf_Ibm_Rs6000:
--     begin
```

```
--          Program.Run_Job
--             (S       =>
--                """!Targets.Implementation.Rcf_release1_0_6"".Start_Rs6000
--              Context => "!Machine.Error_Logs",
--              Options =>
--                 "Output => !Machine.Error_Logs.Rcf_Ibm_Rs6000_Server_Log")
--      end Rcf_Ibm_Rs6000;


Print_Queue_Server:
    begin
        Program.Run_Job
           (Program.Current
               ("!Tools.Rpc_Servers", "Queue_Service.Start",
                Activity => "!Machine.Release.Current.Activity"),
            Context => "!Machine.Error_Logs",
            Options =>
               "Output => !Machine.Error_Logs.Queue_Service_Output," &
                  "Error => !Machine.Error_Logs.Queue_Service_Error," &
                  "User => Network_Public, Password => ()");

    exception
        when others =>
            Error_Reporting.Report_Error
               (Caller      =>
                   "!Machine.Initialization.Local.Initialize_Site.Print_Queu
                Reason      =>
                   Error_Reporting.Create_Condition_Name
                      ("Unhandled_Exception", Error_Reporting.Problem),
                Explanation => Debug_Tools.Get_Exception_Name (True, True));
    end Print_Queue_Server;


Bridge_Security_Server:
    begin
        -- This permits faster connections for login.
        -- In case of trouble, contact TDG.
        Program.Run_Job
           ("""!Local"".Udp_Unreachable_Server",
            Context => "!Machine.Error_Logs",
            Options =>
               "Output => !Machine.Error_Logs.Bridge_Security_Server_Log," &
                  "Name => (Bridge Security Server), " &
                  "User => Network_Public, Password = ()");

    exception
        when others =>
            Error_Reporting.Report_Error
               (Caller      =>
                   "!Machine.Initialization.Local.Initialize_Site.Bridge_Sec
                Reason      =>
                   Error_Reporting.Create_Condition_Name
                      ("Unhandled_Exception", Error_Reporting.Problem),
                Explanation => Debug_Tools.Get_Exception_Name (True, True));
    end Bridge_Security_Server;


-- Excelan_Boot_Server:
--     begin
--         -- This is for the CDF.  In case of trouble, contact GBD.
```

```
--          Program.Run_Job
--             (S        => "Excelan_Boot_Server",
--              Context  =>
--                 "!Targets.Implementation.Motorola_68k_Download'Spec_View.U
--              Options  =>
--                 "Output => !Machine.Error_Logs.Excelan_Boot_Log, Name => (
--              Response =>
--                 "Activity => !Machine.Release.Current.Activity, <DEFAULT>"
--      exception
--          when others =>
--              Log.Put_Line ("Excelan_Boot_Server: Unexpected Exception " &
--                            Debug_Tools.Get_Exception_Name);
--      end Excelan_Boot_Server;


--*********************************************************************
--
-- START THE SERVER TO PERIODICALLY LOGOFF "IDLE" USERS
--
--*********************************************************************

--*** this is an optional capability that can be relatively important for
--    sites that are "session-based"
--
-- Ported from !!Universe by GBD 12 Jun 91.
-- Note that it is idle LOGINS (i.e., SESSIONS), not USERS, that get
-- logged off after they have been in use but idle for the specified
-- amount of time (given by the first parameter in minutes).
-- BED (Universe_Mgr) says this daemon will not force off logins that have
-- active jobs other than the usual pair: the Command job and the Editor
-- job.
--
-- 19 Nov 92 GBD changed parameters from
-- Idle_Users.Logoff_Server (Maximum_Idle_Time_In_Minutes => 210,
--                           Polling_Interval_In_Minutes  => 15);
-- to
--
-- Idle_Users.Logoff_Server (Maximum_Idle_Time_In_Minutes => 4 * 60,
--                           Polling_Interval_In_Minutes  => 24 * 60);
--
-- Also changed After parameter (previously 5 minutes) so the daemon
-- now runs once daily after midnight, just before daily daemon
-- runs.
--
Logoff_Idle_Sessions:
    declare
        After_Delay : constant Duration :=
           Time_Utilities.Duration_Until_Next
              (H => Daily_Daemons_Start_Hour) - Duration'(15 * 60.0);
    begin
        Io.Set_Output ("!MACHINE.ERROR_LOGS.INITIALIZE_SITE");
        Io.Set_Error (Io.Current_Output);
        Activity.Set ("!MACHINE.RELEASE.CURRENT.ACTIVITY");
        Profile.Set (Job_Profile);
        -- Program.Run_Job
        --     (S       =>
        --         """!Machine.Logoff_Idle_Users_Server"".Idle_Users.Logoff_S
        --      Debug    => False,
        --      Context  => "!Machine.Error_Logs",
        --      After    => After_Delay,
```

```
        --      Options  => "Name => (Logoff Sessions Idle Overnight)", --***
        --      Response => Job_Profile_Image);
        Io.Reset_Output;
        Io.Reset_Error;

    exception
        when others =>
            Log.Put_Line
               (Message  =>
                   "Could not start the Logoff Idle Sessions daemon (Excepti
                      Debug_Tools.
                         Get_Exception_Name (Fully_Qualify => True,
                                             Machine_Info  => True) & ")",
                Kind     => Profile.Exception_Msg,
                Response => Job_Profile);
            Io.Reset_Output;
            Io.Reset_Error;
    end Logoff_Idle_Sessions;

Error_Log_Monitor:

    declare

        Program_Name    : constant String :=
           """!Commands.Internal"".Error_Log_Monitor";
        Program_Context : constant String := "!Machine.Error_Logs";
        H_Hour          : constant Time_Utilities.Military_Hours := 1;

    begin
        -- This program checks the error logs for problems and notifies
        -- the system manager.  Contact SHO in case of problems.
        Program.Run_Job
           (S       => Program_Name,
            Context => Program_Context,
            After   => Time_Utilities.Duration_Until_Next (H_Hour),
            Options => "name => (Error Log Monitor)");

    exception

        when others =>

            Error_Reporting.Report_Error
               (Caller      =>
                   "!Machine.Initialization.Local.Initialize_Site.Error_Log_
                Reason      =>
                   Error_Reporting.Create_Condition_Name
                      ("Unhandled_Exception", Error_Reporting.Problem),
                Explanation => Debug_Tools.Get_Exception_Name (True, True));

    end Error_Log_Monitor;

Backup_Server:

    declare

        Program_Name    : constant String :=
           """!Users.Operator"".Backup_Server.Server";
        Program_Context : constant String := "!Users.Operator";

    begin
```

```
            -- Contact GBD or SHO in case of problems.
            Program.Run_Job (S       => Program_Name,
                             Context => Program_Context,
                             After   => 30 * 60.0,
                             Options => "name => (Backup Server)");

        exception

            when others =>

                Error_Reporting.Report_Error
                   (Caller      =>
                       "!Machine.Initialization.Local.Initialize_Site.Backup_Ser
                    Reason      =>
                       Error_Reporting.Create_Condition_Name
                          ("Unhandled_Exception", Error_Reporting.Problem),
                    Explanation => Debug_Tools.Get_Exception_Name (True, True));

        end Backup_Server;

    -- Csr_Hold_To_Open_Server:
    --
    --     declare
    --
    --         Program_Name    : constant String :=
    --            """!Users.Operator"".CSR_Hold_to_Open";
    --         Program_Context : constant String := "!Users.Operator";
    --
    --     begin
    --         -- Starts server to awaken expired on-hold unassigned CSRs.
    --         -- Contact GBD in case of problems.
    --         Program.Run_Job
    --            (S       => Program_Name,
    --             Context => Program_Context,
    --             After   => Time_Utilities.Duration_Until_Next
    --                           (H => 5, M => 45), -- run at 05:45 AM daily
    --             Options => "name => (CSR Hold to Open Server)");
    --
    --     exception
    --
    --         when others =>
    --
    --             Error_Reporting.Report_Error
    --                (Caller      =>
    --                    "!Machine.Initialization.Local.Initialize_Site.CSR_Hol
    --                 Reason      =>
    --                    Error_Reporting.Create_Condition_Name
    --                       ("Unhandled_Exception", Error_Reporting.Problem),
    --                 Explanation => Debug_Tools.Get_Exception_Name (True, True
    --
    --     end Csr_Hold_To_Open_Server;

    Queue.Default ("RC_LP");
    Initialize_Daemons (Daily_Start_Hour => Integer (Daily_Daemons_Start_Hour));

end Initialize_Site;
```

Expmon User Macros

```
XC  experiment [parameters]  --runs experiment on IOC board
XF  experiment [parameters]  --runs experiment on FIU board
XI  experiment [parameters]  --runs experiment on IOA board
XJ  experiment [parameters]  --runs experiment on MEM3 board
XK  experiment [parameters]  --runs experiment on MEM2 board
XL  experiment [parameters]  --runs experiment on MEM1 board
XM  experiment [parameters]  --runs experiment on MEM0 board
XQ  experiment [parameters]  --runs experiment on SEQ board
XS  experiment [parameters]  --runs experiment on SYS board
XT  experiment [parameters]  --runs experiment on TYP board
XV  experiment [parameters]  --runs experiment on VAL board

PREP  uaddr      --Prepare to run at specified microaddress; follow with RM
CFH              --Continue From Halt; follow with RM
RM               --Run Machine; must preceed with PREP or CFH
RMN              --Run Machine with No parity checking enabled
RD               --Run Diagnostic
RDN              --Run Diagnostic with No parity checking enabled
SM               --Stop Machine

READ_FIU_WCS  uaddr     --displays a FIU WCS word
READ_IOC_WCS  uaddr     --displays a IOC WCS word
READ_SEQ_WCS  uaddr     --displays a SEQ WCS word
READ_SYS_WCS  uaddr     --displays a SYS WCS word
READ_TYP_WCS  uaddr     --displays a TYP WCS word
READ_VAL_WCS  uaddr     --displays a VAL WCS word
CHANGE_FIU_WCS  uaddr   --allows you to change fields of a FIU WCS word
CHANGE_SEQ_WCS  uaddr   --allows you to change fields of a SEQ WCS word
CHANGE_SYS_WCS  uaddr   --allows you to change fields of a SYS WCS word
CHANGE_TYP_WCS  uaddr   --allows you to change fields of a TYP WCS word
CHANGE_VAL_WCS  uaddr   --allows you to change fields of a VAL WCS word
AH  uaddr               --Add a Halt microorder to the specified microaddress
RH  uaddr               --Remove Halt microorder from the specified microaddress
AH_ECC                  --Add a Halt microorder to the ECC handler at 18F
RH_ECC                  --Remove Halt microorder from the ECC handler at 18F

TT          --display Control Top register
MARS        --displays the MAR (Memory Address Register) on FIU & all Memories
WDRS        --displays the WDR (Write Data Register) on TYP, VAL & all Memories
 TWDR       --displays the TYP board WDR
 VWDR       --displays the VAL board WDR
XTWDR       --sets the TYP board WDR
XVWDR       --sets the VAL board WDR

MDR         --displays FIU board MDR (Merge Data Register)
READ_TAR    --displays FIU board TAR (Typ Assembly Register)
READ_VAR    --displays FIU board VAR (Val Assembly Register)
XTAR        --sets the FIU board TAR (Typ Assembly Register)
XVAR        --sets the FIU board VAR (Val Assembly Register)

MBHITS      --shows which memory set hit
MBRDR       --displays the RDR (Read Data Register) of the hitting set
MBTVR       --displays the TVR (Tag Value Register) of the hitting set
RDRS        --displays the RDR of all memories
TVRS        --displays the TVR of all memories

LMR  space  name  bit_offs  [number_of_words]   --Logical Memory Read
LMW  space  name  bit_offs  typ_data  val_data  --Logical Memory Write
LTR  space  name  bit_offs                      --Logical Tag Read
PMR  set  line  word                            --Physical Memory Read
```

```
PMW  set  line  word  typ_data  val_data        --Physical Memory Write
PTR  set  line                                  --Physical Tag Read
PTW  set  line  tag_data                        --Physical Tag Write
NTAGS  line                              --displays all tags on specified line
TAGS   line                              --obsolete version of NTAGS

  GP  reg                    --displays specified GP register on both TYP & VAL
 TGP  TGPA  TGPB
 VGP  VGPA  VGPB
XTGP
XVGP
  FREG  frame  reg          --displays specified Frame REGister on both TYP & VAL
 TFREG  TFREGA  TFREGB  TFREGAPAR  TFREGBPAR
 VFREG  VFREGA  VFREGB  VFREGBPAR  VFREGAPAR
XTFREG
XVFREG
  TOP   TOPM[1..8]   TOPP1    BOT   BOTM1  --display TYP & VAL CSA regs
 TTOP  TTOPM[1..8]  TTOPP1   TBOT  TBOTM1  --display only  TYP CSA regs
 VTOP  VTOPM[1..8]  VTOPP1   VBOT  VBOTM1  --display only  VAL CSA regs
XTTOP XTTOPM[1..8] XTTOPP1  XTBOT XTBOTM1  --set TYP CSA regs
XVTOP XVTOPM[1..8] XVTOPP1  XVBOT XVBOTM1  --set VAL CSA regs

FGP     --???
FGPS
FGP[0..9]
```

ACT_LINK
ADD_ACK_REFRESH
ADD_ADA_DEFINED
ADD_LOAD_WDR
ADD_MILD
ADD_SEVERE
ADD_UNIQUE
ALW
ARG
ASSERT_OF_KIND
AUX_ALLOC
AUX_STATE
BACKALINE
BAD_BITS
BENCH
BID_TEST
BUFFER_TEST
BUFF_PARITY
BUFF_WORDS
CASE_BS
CASE_DAY
CASE_HUNT
CASE_MAP
CASE_MONTH
CASE_NODE
CASE_PARITY
CHANGE_BUFSTAT_CONTROL
CHANGE_PROMPT
CHECK_BUFFER
CHECK_ECC
CHECK_PAK
CHECK_PARITY
CHECK_PARITY_T
CHECK_WCS
CINS
CLEAR_BREAK_MASK
CLEAR_EVENTS
CLEAR_HERR
CLEAR_HITS
CLEAR_RESYNC
CLEAR_TAGSTORES
CLEX
CONDITIONAL_WRITE
COND_MEM_WIRE_OK
CONTINUE
COUNTER_OK
COUNT_LEADING_ZEROS
CP
CRASH_INFO
CSAS
CSA_OK
CTB
CTC
CTWCS
CVWCS
CW
DATA_UNIQUE
DECIMAL
DECODE_TEST

DEC
DEC_MEM_START
DEC_SEQ
DEFINE_MEM_VARS
DELAY
DISABLE_IOA_RCV
DISPLAY_FIU_MAR
DISPLAY_FIU_UIR
DISPLAY_FRAME_INFO
DISPLAY_MEM_WORD
DISPLAY_NOVRAM_INFO
DISPLAY_RDR
DISPLAY_SEQ_STATE
DISPLAY_SYS_UIR
DISPLAY_TAG
DISPLAY_TAG_MULTIPLE
DISPLAY_TAG_WORD
DISPLAY_TYP_WCS
DISPLAY_VAL_WCS
DISP_ADDR_SRC
DISP_ALU
DISP_A_ADDRESS
DISP_BKPT
DISP_BRANCH_ADDR
DISP_BRANCH_KIND
DISP_BRANCH_TIME
DISP_BS
DISP_BUFFER_CONTROL
DISP_BUF_STAT_SEL
DISP_B_ADDRESS
DISP_CALENDAR
DISP_CLASS_LIT
DISP_COND_BOARD
DISP_COND_FIRST_HALF
DISP_COND_KIND
DISP_COND_SECOND_HALF
DISP_CSA_CNTL
DISP_C_ADDRESS
DISP_C_SOURCE
DISP_DIAG_BUFFER
DISP
DISP_FILL_MODE_LITERAL
DISP_FILL_MODE_SOURCE
DISP_FIU_SRC
DISP_FRAME
DISP_GENERAL_CONTROL
DISP_GEN_CTL
DISP_HUNTERS
DISP_IOA_BUFFER
DISP_IOA_ERRORS
DISP_IOA_HEADER
DISP_IOA_IPC_STATUS
DISP_LATCH_CNTRL
DISP_LENGTH_LITERAL
DISP_LENGTH_SOURCE
DISP_LEX_ADDR
DISP_LFREG_CNTL
DISP_MAP
DISP_MAR_CNTL
DISP_MDR_CNTL

DISP_MEM_START
DISP_MERGE_INPUT_SRC
DISP_MERGE_VMUX_SELECT
DISP_MICRO_CNTRL
DISP_MULT_BS
DISP_MULT_IN
DISP_OFFSET_SOURCE
DISP_OFFS_REG_CNTL
DISP_OFFS_REG_SRC
DISP_OP_SELECT
DISP_PRIVACY
DISP_SEQ
DISP_SEQ_PARITY
DISP_SEQ_RANDOM
DISP_SEQ_READ_SRC
DISP_STATUS_CONTROL
DISP_TAR_CNTL
DISP_TI_VI_SOURCE
DISP_TV_SRC
DISP_TYPE_SRC
DISP_TYP_MUX
DISP_TYP_PARITY
DISP_TYP_RANDOM
DISP_VAL_MUX
DISP_VAL_PARITY
DISP_VAL_RANDOM
DISP_VAL_SRC
DISP_VARS
DISP_VAR_CNTL
DISP_VECTORS
DISP_WDR_CONTROL
DISP_XERR_CODE
DOWN_FRAME
DPC
DQ_HEAD
DRAM_EXIST
DRPC
DSPA
DSP
DUMP_CSA
DW_HEAD
ECC_DISPLAY_BITS
ECC_DISPLAY_PLANE
ECC
ECC_ERROR
ECC_INFO
ECC_LOG
ECC_LOG_ENTRY
ECC_OF
ECC_OF_VERY_SLOW
ELOG
ENABLE_IOA_IPC
ENABLE_IOA_RECV
ENABLE_IOA_XMIT
ENABLE_IO_SYSTEM
EP1
ERRORS
ERSATZ_TEST
EVALUATE_TAG
EXT

FAIL_MESSAGE
FAQ_HEAD
FAR
FHB_1
FHB
FILR
FIRST
FIU1399
FIU_BUS_TEST_OK
FIU_DIAG2_KERNEL
FIU_DIAG_KERNEL
FIU_FRU
FIU_MM_CSA10_TESTS
FIU_MM_CSA20_TESTS
FIU_MM_CSA2_TESTS
FIU_MM_CSA3_TESTS
FIU_MM_CSA3_TESTS_OLD
FIU_MM_CSAA_TESTS
FIU_MM_CSA_TESTS
FIU_MRG_ROTATR2_TESTS
FIU_MRG_ROTATR_TESTS
FIU_PARAM_BUS_TESTS
FIU_PARITY_TESTS
FIU_RESET_TEST
FIU_TILE5_INIT
FIU_WCS10_TESTS
FIU_WCS20_TESTS
FIU_WCS2_TESTS
FIU_WCS30_TESTS
FIU_WCS3_TESTS
FIU_WCS4_TESTS
FIU_WCSA_TESTS
FIU_WCS_TESTS
FIX_LOOP_BACK
FKEY
FLUSH_CSA
FMS0
FMS1
FOR_ALL_MEM
FPFMAR
FRAME
FREE_SET_LIM
FUSTACK
GET_ARG
GET_A_ADDRESS
GET_B_ADDRESS
GET_CLASS_LITERAL
GET_CPU_CONFIG
GET_C_ADDRESS
GET_FILL_MODE_LITERAL
GET_FILL_REG
GET_FIU_ERRORS
GET_FIU_LFREG
GET_FIU_UIR
GET_IOC_ERRORS
GET_LENGTH_LITERAL
GET_LINE
GET_MEM_CONFIG
GET_MEM_DATA
GET_MEM_ERRORS

GET_MEM_STATE
GET_MULT_IN
GET_OFFSET_LITERAL
GET_OFFSET_SOURCE
GET_RF_FRAME
GET_SEQ_ERRORS
GET_SEQ_UIR
GET_SYS_ERRORS
GET_SYS_UIR
GET_TAG_DATA
GET_TYP_ERRORS
GET_TYP_UIR
GET_VAL_ERRORS
GET_VAL_UIR
GREEN_LIGHT
HASH
HEADER
HEX
HSH
IBUFF
IMC4
IM
INFO
INIT_CHECK_WCS
INIT
INIT_FIU_MRU
INIT_IOA_BIDPRI
INIT_IOA
INIT_IPL
INIT_MEM_STATE
INIT_MRU
INIT_NOVRAM
INIT_VARS
INMSK
IOA_ALL_IPC
IOA_FRU
IOA_INIT
IOA_IPCY_TESTS
IOA_IPC_TESTS
IOA_RAM_TESTS
IOC_DIPROC_TEST
IOC_ECC_TEST
IOC_ENABLE_TEST
IOC_EVENTS_TEST
IOC_FRU
IOC_MAIN_MEMORY_TEST
IOC_SCAN_CHAIN_TEST
IOC_TIMER_TEST
IOC_TRACE_TEST
IOC_WCS_TEST
IPC
IPC_INIT
IPSEA0
IPSEA1
I_MEM1
I_MEM
I_TAGS
KILL
LCS
LENR

LEX_VALID
LINE_NUM
LINK
LL
LMP
LOAD_CALENDAR
LOAD_CLOCK_TIMER
LOAD_GP_TIMER
LOAD_NEW_HASH_RAMS
LOAD_SLICE_TIMER
LOAD_SYS_CODE
LOOKQ
LOOP
LOOP_LRU
LOOP_TAGSTORE
LOOP_WRITE_MEM
LOU
MEM0_EXISTS
MEM1_EXISTS
MEM2_EXISTS
MEM3_EXISTS
MEM_STATUS_TEST
MEM_TEST
MERGER_OK
MISC_CUR_ADDR
MISC_ERRORS
MISC_JUNK
MISC_MEVENTS
MISC_SEQ
MISC_UEVENTS
MODEL
MPCC
MPC
MST
MULT_PROMPT
MVMUX_OK
MW_HEAD
NAME_MESSAGE
NEGATIVE_LOGIC
NEW_ADD_UNIQUE
NEW_DISPLAY_TAG_WORD
NEXTD
NEXT
NFPFMAR
NINS
NMSK
NORMAL
NORMAL_MACS
NOT_FLAG
NO_P2VAL
OFFR
OLD_DISPLAY_TAG_WORD
OLD_ECC_LOG
OLD_PROMPT
OUR_TYPE
PAK_INIT
PARAM_REGS_OK
PARITY_OF_SLOW
PASS_MESSAGE
PATTERN_1

PATTERN_2
PC
PFMAR
PINIT
POFF
POLL_ALL
POLL_FOR_MC
POSITIVE_LOGIC
PROMPT
PROMPT_FILL_MODE_LITERAL
PROMPT_LIT
PROMPT_OFFSET_SOURCE
PUT_MEM_DATA
QSTEP
QSUCC
QUAD_DENSITY
QUIT
RBP
RCV
READ_BUFF_WORDS
READ_CLOCK_TIMER
READ_DEC
READ_FIU_UIR
READ_GP_TIMER
READ_IOP
READ_LINE
READ_MEM_WORDS
READ_MISC
READ_NOVRAM
READ_PHYSICAL_WORDS
READ_SDR
READ_SEQ_UIR
READ_SLICE_TIMER
READ_SYS_BUFFER
READ_SYS_UIR
READ_TAG_MULTIPLE
READ_TYP_UIR
READ_UIR
READ_VAL_UIR
REMOVE_ADA_DEFINED
REMOVE_MILD
REMOVE_SEVERE
RESET_ALL
RESET_CPU
RESOLVE_AND_READ
RESOLVE
RESTORE_MEM_STATE
RESTORE_SEQ_STATE
REST
RESULT
RF16
RF8
RFM
RF
RHB_1
RHB
RMAR
RNO
ROFF
ROOT

ROTATOR_OK
RPCC
RPC
RST
RTWCS
RVWCS
SAVE_MEM_STATE
SAVE_SEQ_STATE
SBUT
SBUV
SCAN_CHAINS_OK
SDR_SEQ
SD
SEQ_1570
SEQ_ADR_STARTS_TESTS
SEQ_CONTRL_REG_TESTS
SEQ_DECODE_RAM_TESTS
SEQ_DIAG2_KERNEL
SEQ_DIAG_KERNEL
SEQ_DISPATCH_TEST
SEQ_ERLY_CONDITL_SEQ
SEQ_FOO
SEQ_FRU
SEQ_IBUFF_INST_TESTS
SEQ_LATCH_TEST
SEQ_MACRO_LOG_TESTS
SEQ_MACRO_RPC_TESTS
SEQ_RARELY2_SEQUENCIN
SEQ_RARELY_SEQUENCIN
SEQ_STACK_TEST
SEQ_UNCONDITIONL_SEQ
SEQ_USUALLY2_SEQUNCIN
SEQ_USUALLY_SEQUNCIN
SEQ_WCS_TESTS
SET_CONTROL_PRED
SET_CONTROL_TOP
SET_CURRENT_LEX_LEVEL
SET_CURRENT_NAME
SET_HOME
SET_IBUFF
SET_MACRO_PC
SET_RETURN_PC
SEXT
SHORT_UADDR_PARITY
SLEX
SM_COUNT
SM_HIST
SOR
SO
SRF
SRO
SST
SS
STEP
STEP_FIU_OK
STIMULATE_VAL_BUS
ST
SYNDROME_CASE_0
SYNDROME_CASE_1
SYNDROME_CASE_2

SYNDROME_CASE_3
SYNDROME
SYS_BID_TEST
SYS_BUFFER2_TEST
SYS_BUFFER_TEST
SYS_DECODE_TEST
SYS_DECODE_TEST_REV1
SYS_DIPROC_TEST
SYS_ERR_CODE
SYS_ERR_NODE
SYS_ERSATZ_TEST
SYS_ERSATZ_TEST_REV1
SYS_FRU
SYS_HEADER_NUDGE
SYS_LOOP4
SYS_LOOP5
SYS_LOOPER
SYS_LOOP_BACK
SYS_LOOP_INIT
SYS_MACROS_TEST
SYS_MISC_SCAN_TEST
SYS_MISC_SCAN_TEST_REV1
SYS_REGISTER_TESTS
SYS_STATUS_TEST
SYS_TILE11_INIT
SYS_TILE12_INIT
SYS_TILE7_INIT
SYS_TILE9_INIT
SYS_TIMER
SYS_TIMER_TEST
SYS_TRANSFER_TEST
SYS_UCODE
SYS_UIR_TEST
SYS_WCS_TEST
SYS_XMIT_ERRORS
TAG_QUERY --used by LTR
TAR_VAR_OK
TAR_XMIT_ERRORS
TCSAS
TESTY
TESTY_I_BOARD
TEST_2MEG_MEM_BOARD
TEST_2MEG_MEM_TILE
TEST_8MEG_MEM_BOARD
TEST_8MEG_MEM_TILE
TEST_ALU
TEST_AND
TEST_C_BOARD
TEST_DRAM32
TEST_DRAM
TEST_FIU_BOARD
TEST_FIU
TEST_F_BOARD
TEST_IMC2
TEST_IMC3
TEST_INCOMPLETE_MCYC
TEST_IOA_BOARD
TEST_IOA
TEST_IOC_BOARD
TEST_IOC

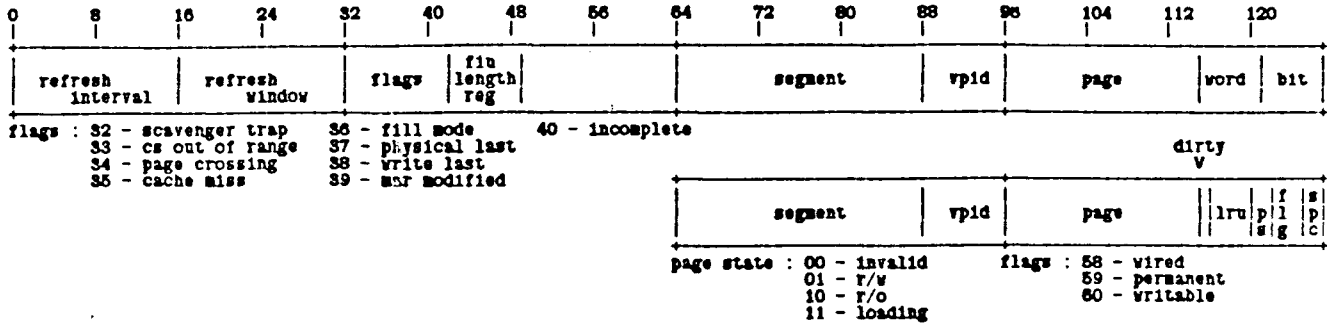
TEST_I_BOARD
TEST_LRU
TEST_MEM32
TEST_MEMORY_BOARD
TEST_MEM_BOARD
TEST_MEM
TEST_MEM_HASH
TEST_M_BOARD
TEST_PAREG
TEST_Q_BOARD
TEST_READ
TEST_RUN
TEST_SEQUENCER_BOARD
TEST_SEQ
TEST_SYSBUS_BOARD
TEST_SYS
TEST_SYS_REV1
TEST_SYS_UIR
TEST_S_BOARD
TEST_TAGSTORE_PARITY
TEST_TILE
TEST_TYP
TEST_T_BOARD
TEST_VAL
TEST_V_BOARD
TEST_WCS_VAL
TILE_MEM32_DATA_STORE
TILE_MEM32_DIAG_KERNEL
TILE_MEM32_SCAN_CHAINS
TILE_MEM32_TAGSTORE
TILE_MEM_DATA_STORE2
TILE_MEM_DATA_STORE
TILE_MEM_DIAGNOSTIC_KERNEL
TILE_MEM_SCANNABLE_REGSTERS
TILE_MEM_TAGSTORE2
TILE_MEM_TAGSTORE3
TILE_MEM_TAGSTORE
TIMERS
TLC
TLHB
TLR
TOGGLE
TRACE
TRANSPARENT
TRF16
TRF8
TRFA16
TRFA8
TRFA
TRFB16
TRFB8
TRFB
TRF
TYPE_ALL_BS
TYPE_ALL_MAPS
TYPE_ALU2_TESTS
TYPE_ALU_TESTS
TYPE_BUS_N_LOOP_CNTR
TYPE_CALENDAR
TYPE_CSA_TESTS

TYPE_DIAG_BUFFER
TYPE_DIAG_KERNEL
TYPE_IOA_BUFFER
TYPE_IOA_BUF_HALFWORD
TYPE_IOA_ERRORS
TYPE_IOA_HEADER
TYPE_IOA_IPC_STATUS
TYPE_IOA_STATUS
TYPE_MAPS
TYPE_RESET_TEST
TYPE_RF10_TESTS
TYPE_RF20_TESTS
TYPE_RF2_TESTS
TYPE_RF3_TESTS
TYPE_RFA_TESTS
TYPE_RF_TESTS
TYPE_TIGHT
TYPE_WCS_TESTS
TYP_FRU
UP_FRAME
USTACKS
USTATE1
USTATE2
USTATE
USTK
VAL103
VALID
VAL_9
VAL_ALU2_TESTS
VAL_ALU_TESTS
VAL_BUS_N_LOOP_CNTR
VAL_DIAG_KERNEL
VAL_FRU
VAL_R10_TESTS
VAL_RESET_TEST
VAL_RF10_TESTS
VAL_RF20_TESTS
VAL_RF2_TESTS
VAL_RF3_TESTS
VAL_RF4_TESTS
VAL_RF5_TESTS
VAL_RF6_TESTS
VAL_RFA_TESTS
VAL_RF_TESTS
VAL_WCS_TESTS
VAL_ZERO_CNTR_TESTS
VCSAS
VERIFY_FIELD_ERROR
VLC
VLR
VRF16
VRF8
VRFA16
VRFA8
VRFA
VRFB16
VRFB8
VRFB
VRF
WAIT_FOR_SEQ_MC

WCS_OK
WRITE_CSA_ENTRY
WRITE_FIU_WCS
WRITE_MEM_MAR
WRITE_SEQ_WCS
WRITE_SYS_WCS
WRITE_TYP_WCS
WRITE_VAL_WCS
WTWCS
WVWCS
XCMP
XFILRP
XFILR
XIBUFF
XLENRP
XLENR
XLL
XMAR
XMDR
XMIT
XMQ
XOFFR
XPC
XSQ
XTLC
XTLR
XTRF
XVLC
XVLR
XVRF

CW_ACCEPT_LINK
CW_ACCEPT_SUB
CW_ACCESS_VAL
CW_ACTIVATION_LINK
CW_ACTIVATION_STATE
CW_ALL
CW_ARRAY_VAR
CW_AUXILIARY_MARK
CW_AUXILIARY_STATE
CW_AUX_ALLOCATION
CW_BOOLEAN
CW_CONTROL_ALLOCATION
CW_CONTROL_STATE
CW_DEPENDENCE_LINK
CW_DISCRETE_VAR
CW_DISPLAY
CW_ENTRY_VAR
CW_EXCEPTION_VAR
CW_FAMILY_VAR
CW_GROUP_0
CW_GROUP_1
CW_GROUP_2
CW_GROUP_3
CW_INTEGER
CW_KIND
CW_MODULE
CW_REF
CW_SCHED_ALLOCATION
CW_SELECT_VAR

CW_STATIC_CONNECTION
CW_SUB_REF
CW_SUB_VAR
CW_TYPE_LINK
CW_TYPE_VAL
CW_VAL
CW_VARIANT_REC
CW_VAR_REF

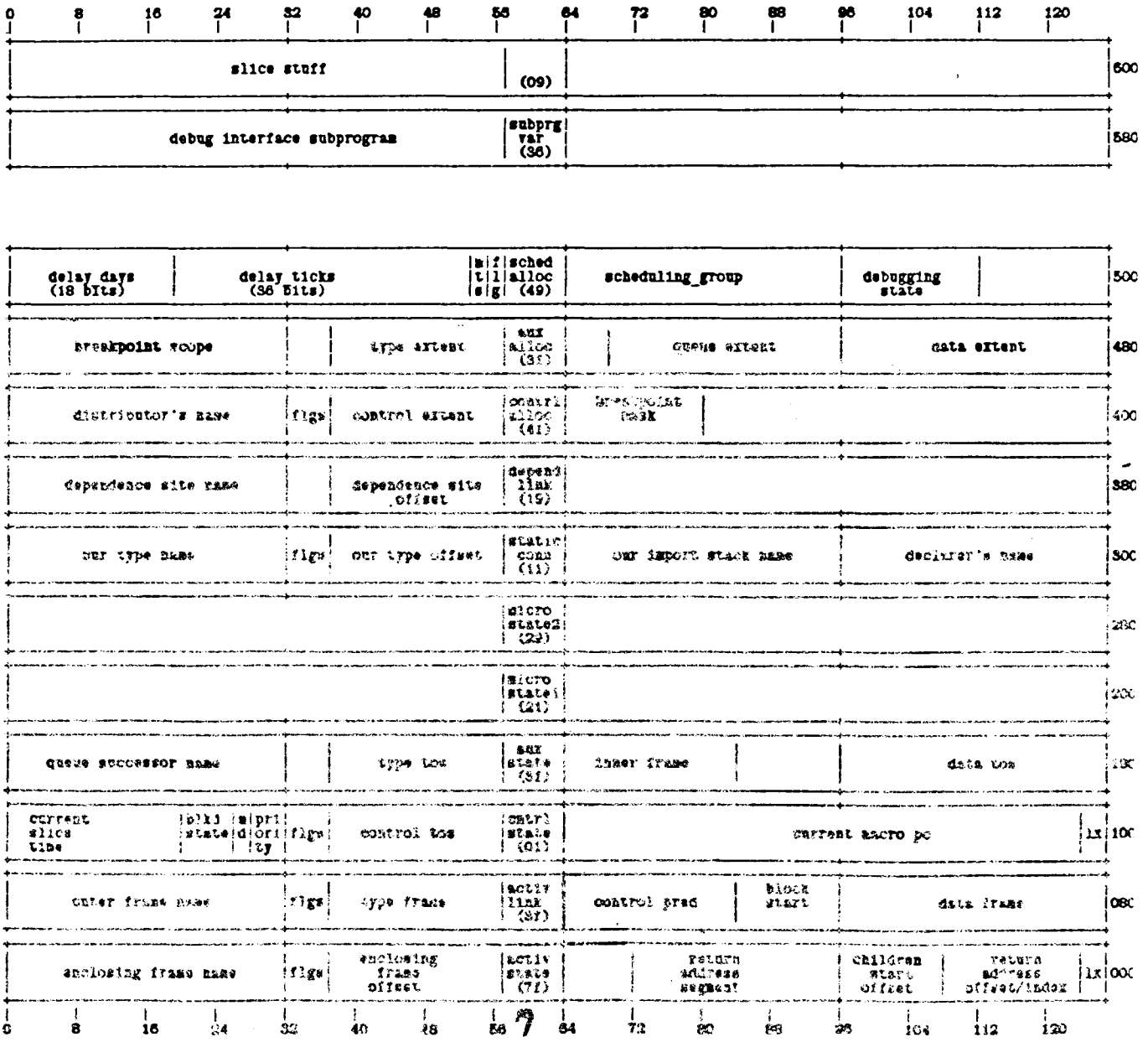


SOME USEFUL TAGS

Discrete : 00 (80)	Record : 44 (C4)	Subprogram : 06 (86)	Null Subprogram : 76	Seg Heap : 38 (B8)
Access : 10 (90)	Variant Record : 4C (CC)	(Elaborated) : 16 (96)	Utility " : 66 (E6)	
Task : 18 (98)	Vector : 6C (EC)	(Visible) : 26 (A6)	Accept " : 46 (C6)	
Package : 58 (D8)	Matrix : 74 (F4)	(Visible &	Interface " : 56 (D6)	
Float : 08 (88)	Array : 7C (FC)	Elaborated) : 36 (B6)	Exception Var : 7E (FE)	

BLOCKED STATES

Unblocked : 00	Terminable At End : 07	In FS Rendezvous : 0E	Blocking On Accept : 18
Declaring Module : 01	Blocking On Entry : 08	In Wait Svc : 0F	Blocking On Select : 19
Awaiting Activation : 02	Delaying On Entry : 09	Delay In Wait Svc : 10	Delaying On Select : 1A
Activating Module : 03	Attempting Entry : 0A	Blocking On Abort : 11	Await Children Select : 1B
Activating Tasks : 04	Delaying : 0B	Deleted : 12	Terminable In Select : 1C
Awaiting Task Activ : 05	Aborting Module : 0C	Aborted While In MTS : 13	
Awaiting Children : 06	Terminated : 0D	In_MTS_RendezVous : 14	



Vælg land

Support & Download
Søgeresultat
Tilbagemelding

Relaterede link
IT-produktuddannelse
IBM Redbooks
Salgsmanual
Annonceringsbreve
Softwaresupport-håndbog
Udviklere
IBM Business Partnere

DELTA: How to give a backup daemon a higher job priority to complete successfully overnight during the usual garbage collection

Technote

Problem
User jobs may use a lot of CPU time at night. The R1000s400 may run into garbage collection mode before the backup actually finishes. Difficulties getting a backup to complete can be circumvented giving the backup daemon a "higher priority".

Cause
Past a Foreground_Time_Limit seconds, any user job and the backup job will end up as background jobs.The R100s400 System resources are divided over background jobs and a
foreground job such as an OE job. The Percent_For_Background is relevant for the CPU time given to any runnable background jobs (including server jobs). The rest is available for the foreground job such as an OE job

Solution

The procedure Scheduler.Set_Job_Attribute should be used on the backup job to set the job kind to OE.

REFERENCE:
R1000 s400 System Management Utilities

Dokumentoplysninger

Produktkategorier:
Software

Software Development

Traditional Languages & Debug Tools

Rational Apex

Styresystem(er):
All Unix Platforms

Softwareversion:
Version independent

Referencenr.:
1161834

IBM-gruppe:
Software Group

Ændret den:
2004-02-27

DELTA: How to allocate a job's garbage to a particular volume?

Technote

Problem
A job's garbage may be composed of the job heap, segment space, or both. There are two load functions on the R1000s400 that can be used to get information about a job's garbage allocation on a particular volume. We provide the user a code sample "as is" to reallocate a job's garbage on another volume using a property of the delta environment.

Solution

Given a job id, the delta environment provides the user with a load function that returns the volume where the job heap is located (Job_Heap_Volume). Another load function can then be used to locate the segment space of the job id (Segment_Space_Volume).

Note that the user cannot allocate the garbage to a particular volume, but if it is on a volume which is saturated, the circumspection of the problem consists in restarting a new version of the job, then kill the first version. If you simply kill the job and restart it, the volume which was just released will be allocated to the job again.

This is usually coming up with large jobs (e.g so called RDF jobs). The garbage created by that kinds of jobs, which run for 24 hours or more, can consume up to 50% of a volume. Hence the need to allocate the garbage to a particular volume. The system will allocate the garbage to the Segment_Space.

The reader will find here a sample code of a procedure to allocate a job to a new volume:

```
with Job;
with Program;
with Machine;
with Segment_Space_Volume;

procedure Allocate_Job_To_Volume (This_Job : in String; Volume : in Integer) is
Temp_Vol : Integer;
Job_Id : Machine.Job_Id;
Status : Program.Condition;
begin
Program.Create_Job (This_Job, Job_Id, Status);
Temp_Vol := Segment_Space_Volume (Job_Id);
if Temp_Vol /= Volume
then
Program.Run_Job ("Allocate_Job_To_Volume " &
"(" &""" & This_Job & """ &
"," & Integer'Image (Volume) & ")"
);
delay 600.0;
Job.Kill (Job_Id);
else
Program.Wait_For (Job_Id);
end if;
end Allocate_Job_To_Volume;
```

Dokumentoplysninger

Produktkategorier:
Software

Software Development

Traditional Languages & Debug Tools

Rational Apex

Styresystem(er):
All Platforms

Softwareversion:
Version independent

Referencenr.:
1161864

IBM-gruppe:
Software Group

Ændret den:
2004-02-27