ISTITUTO NAZIONALE DI FISICA NUCLEARE

Sezione di Bologna

INFN/TC–03/15
14 Novembre 2003

PROGETTO CONDOR: INSTALLAZIONE AUTOMATICA DEL SOFTWARE

Daniela Bortolotti, Paolo Mazzanti, Franco Semeria

INFN, Sezione di Bologna

Sommario

Viene descritta una semplice realizzazione di un tool di installazione, basato su web, del software *Condor* particolarmente adattato alle esigenze di base per il Condor Pool INFN su WAN. Tale procedura viene incontro a necessita' da tempo espresse dalla comunita' INFN associata ai problemi di calcolo. In sostanza si e' detto che una delle ragioni della staticita' del pool stesso per quanto riguarda il numero delle macchine che lo compongono, e' dovuta ad oggettive difficolta' dovute alla installazione di base, ossia quella fornita con il software. La nostra speranza e' che questo metodo di installazione permetta effettivamente di aggiungere macchine al Condor Pool Nazionale e cosi' servire meglio e meglio ottimizzare le risorse di calcolo. Per questi motivi e' stato volutamente mantenuto a livelli di estrema semplicita' di uso, ma, speriamo, di ragionevole efficacia per le necessita' dell'INFN. Puo' facilmente essere esteso, se la comunita' scientifica lo richiedera'.

Published by ***SIS–Pubblicazioni***
Laboratori Nazionali di Frascati

1 INTRODUZIONE AL CONDOR POOL INFN SU WAN

Il Condor Pool INFN su WAN e' stato piu` volte descritto e molte presentazioni sulle sue caratteristiche, funzionalita' ed utilizzo sono state fatte e disponibili in rete:

http://server11.infn.it/condor/

La sua configurazione e la sua distribuzione geografica ed in particolare il numero di macchine che lo costituiscono sono tuttavia statici da molto tempo.

Non si vuole qui cercare di descrivere o di commentare sulle ragioni di una tale situazione, l'unica cosa che si fa notare e' che il pool e' molto usato ed un incremento delle risorse disponibili potrebbe essere indubbiamente positivo.

La configurazione di una macchina in un Condor Pool da parte di un singolo utente o di chi abbia responsabilita' di gestione del calcolo nell'ambito di una Sezione o di un Laboratorio dell'Ente, puo' avere svariate motivazioni.

La motivazione piu' immediata e' che siano presenti esigenze di potenza di calcolo che non sono disponibili localmente.

Un'ulteriore e piu' sottile motivazione ha a che fare con qualche forma di convincimento personale che un migliore utilizzo delle risorse che si hanno a disposizione potrebbe essere ottenuto mediante la condivisione delle risorse stesse, senza naturalmente perderne il controllo. E, in effetti, *Condor* permette proprio questo: quanto viene messo a disposizione viene immediatamente rilasciato non appena il 'propritario' della risorsa ne richiede l'uso; *Condor* utilizza il tempo di 'wait while idle' .

Ci sono tuttavia difficolta' oggettive, in mancanza di contingenti maggiori necessita' di calcolo, che spesso impediscono l'aggiunta di macchine al pool nazionale.

L'installazione del sistema partendo dai principi primi implica una attenta lettura di parte del manuale, preliminare alla installazione stessa, che comunque richiede tempo.

E proprio questo manca molto spesso, in conseguenza delle molteplici attivita' che ognuno di noi e' tenuto a svolgere nell'ambito della propria sede di lavoro.

E' stato suggerito, e questa e' la motivazione principale del lavoro qui descritto, che un tool di installazione semplice permetterebbe di superare le difficolta' sopra descritte e conseguentemente ci si potrebbe aspettare un soprendente aumento delle risorse nell'INFN WAN Condor Pool.

Il tool proposto, disponibile per i S.O. LINUX e DUX, e' estremamente semplice, basato su RPM per Linux e su shar per DUX ed e' accessibile via web (fig. 1 in appendice B) al seguente URL:

http://www.bo.infn.it/calcolo/condor/infn-installation-tool.html

2 OBIETTIVI E REQUISITI DEL TOOL DI INSTALLAZIONE

Il principale obiettivo e' la semplicita': il tool deve essere semplice in senso generalizzato (accesso, utilizzo, ecc.). Un prezzo tuttavia si deve pagare: vengono fatte scelte sui file di configurazione, ossia su come si presenta la macchina nel pool, che riflettono l'uso fin ora fatto delle macchine presenti nel pool nazionale. Questo significa che non tutte le quasi illimitate (!) possibilita' di configurazione di una WS in un Condor Pool vengono usate.

Il tool e' stato preparato soltanto per le architetture INTEL Linux RedHat con glibc 2.1, glibc 2.2 e Digital UNIX 4.x, e questo riflette sia una vecchia decisione presa al riguardo in un meeting, sia l'effettivo utilizzo in ambito INFN.

Si e` anche deciso di predisporre, nelle installazioni di tipo "client" e "completa", la creazione di un utente condor locale, a discapito di eventuali configurazioni in NIS, etc., in quanto non era possibile prevedere ogni tipo di amministrazione dei sistemi.

Le informazioni richieste dai form (spiegate in dettaglio nel successivo capitolo) consentono di caratterizzare alcuni parametri del software, lasciando comunque inalterata la politica di utilizzo globale.

Il pool e` suddiviso in domini di checkpoint, ognuno dotato di un checkpoint server, a cui afferiscono le sedi geograficamente meglio connesse. In questo modo si snellisce l'utilizzo della rete e si ottimizzano i tempi di esecuzione. I job cercano le risorse disponibili nell'ambito del proprio dominio e solo in caso di mancanza di risorse fuoriescono da esso.

Attualmente sono predefiniti otto checkpoint server a copertura di tutto il territorio nazionale:

- condor.bo.infn.it (Bologna, CNAF, Pisa)
- Condor.pv.infn.it (Pavia)
- cerere.to.infn.it (Torino)
- ckptmi.mi.infn.it (Milano, Bicocca, Udine)
- condorckpt.pd.infn.it (Padova, Legnaro)
- ckpt.lngs.infn.it (Gran Sasso)
- lxmaster2.na.infn.it (Bari, Perugia, Napoli, Lecce, Roma1, Roma2)
- ckpt.ts.infn.it (Trieste)

E` inoltre possibile configurare dei sub-pools di macchine, tramite il parametro SUBMIT_GROUP_ID, in cui poter privilegiare l'esecuzione di job di gruppo. Se tale campo non viene compilato, il tool prevede comunque di avvantaggiare i job sottomessi dallo stesso dominio (bo.infn.it, oppure na.infn.it, etc).

3 TIPI DI INSTALLAZIONE

La installazione del software di *Condor* e` piuttosto semplice ma se il numero di macchine da gestire per sede e` alquanto elevato, diventa difficoltoso mantenerlo aggiornato. Abbiamo riscontrato che e` importante avere versioni uniformi nell'ambito dello stesso pool

per un migliore funzionamento generale e poiche` spesso manca questo allineamento causa la mancanza di "manpower" locale o conoscenza del prodotto, abbiamo realizzato uno strumento immediato, capace di distribuire *Condor* in formato di facile installazione, con incluse le caratteristiche per l'INFN. In questo modo diventa semplice l'aggiornamento o l'installazione del software e non e` indispensabile sapersi muovere tra i parametri di configurazione di *Condor*.

Sono state individuate tre modalita` di installazione, in grado di comprendere buona parte dei casi:

- installazione server
- installazione client
- installazione completa

3.1 Installazione server

Scegliendo questo tipo di installazione la distribuzione di librerie, binari e la configurazione generale e` presso un server centrale (AFS, NFS, etc) a cui possono accedere macchine della sezione. Questa strategia e` consigliata quando sono presenti diversi sistemi operativi e si vuole centralizzare la distribuzione del software di *Condor*, inoltre esistono gia` utenti locali di condor in ogni macchina.

Vengono richiesti parametri di configurazione del software (fig. 2 in appendice B) come:

- USE_CKPT_SERVER
- DEFAULT_DOMAIN_NAME
- HOSTALLOW_ADMINISTRATION
- CONDOR_ADMIN

altri di gestione come :

- RELEASE_DIR
- OPSYS

Il parametro di DEFAULT_DOMAIN_NAME indica il dominio internet della sezione o del laboratorio, e` fondamentale per individuare il dominio di checkpoint e per definire nella espressione di RANK il privilegio di esecuzione dei job.

Il parametro di HOSTALLOW_ADMINISTRATION rappresenta la macchina da cui inviare remotamente comandi di gestione (oltre al Central Manager naturalmente) verso il sottoinsieme; tale server puo` fermare, riconfigurare, riavviare *Condor* di altre macchine, pertanto e` bene che non disponga di utenza generica.

Il parametro di USE_CKPT_SERVER abilita l'utilizzo di un checkpoint server locale o "vicino", normalmente e` settato e cio` evita di salvare localmente i job in esecuzione, interrotti.

La interfaccia web distribuisce binari e librerie, per architettura, in formato tar e prepara uno shar file (script unito a dati) che al momento della esecuzione personalizza i file template di configurazione (vedi capitolo successivo), inserendo i parametri del form.Viene generato

un albero di software a partire da RELEASE_DIR, suddiviso per architettura e configurazione. In sintesi abbiamo la seguente struttura:

- $RELEASE_DIR/$OPSYS/bin
- $RELEASE_DIR/$OPSYS/sbin
- $RELEASE_DIR/$OPSYS/lib
- $RELEASE_DIR/etc

3.2Installazione client

Questa installazione e` associata alla presenza di un server per la distribuzione di librerie e di binari (vedi paragrafo precedente). Il lato "client" prevede la creazione di un utente unix condor locale, l'attivazione del demone al boot e la definizione delle priorita` locali. Vengono richiesti parametri di configurazione (fig. 3 in appendice B) come:

- DEFAULT_DOMAIN_NAME
- SUBMIT_GROUP_ID

altri di gestione come:

- RELEASE_DIR
- OPSYS
- CONDOR_HOME

I parametri di configurazione personalizzano il file condor_config.local, mentre quelli di gestione occorrono per creare la directory iniziale dell'utente unix e per individuare il file di configurazione generale condor_config (tramite link).

I parametri di DEFAULT_DOMAIN_NAME e di SUBMIT_GROUP_ID, presenti nel file condor_config.local, consentono nella espressione di RANK, di privilegiare l'esecuzione di job pertanto solo il condor-admin locale e` in grado di fare questa valutazione. L'espressione utilizzata (tratta dal template di condor_config.local) e` la seguente:

```
RANK : (SUBMIT_SITE_DOMAIN =?=  "SUBSTITUTEDOMAINNAME") + \
      (10*(SUBMIT_GROUP_ID  =?=  "SUBSTITUTEGROUPID"))
```

se il dominio e` "bo.infn.it" e il gruppo di ricerca "CMS-group", a fine installazione diventa:

```
RANK : (SUBMIT_SITE_DOMAIN =?= "bo.infn.it") + \
   (10 * (SUBMIT_GROUP_ID =?= "CMS-group"))
```

I job avviati da "bo.infn.it" hanno maggiore possibilita` di esecuzione (pari a uno) nelle macchine del dominio, rispetto a quelli schedulati da sezioni differenti. I job avviati da una macchina di "CMS-group" hanno maggiore possibilita` (pari a dieci) di elaborazione in macchine con target uguale. In questo modo i gruppi di ricerca possono privilegiare i lavori dei propri utenti nelle macchine di esperimento. I valori di default sono:

- SUBMIT_SITE_DOMAIN uguale aldominio locale (es. bo.infn.it)

- SUBMIT_GROUP_ID uguale alla prima parte del dominio seguito da stringa (es. bo_Group)

La interfaccia web prepara un pacchetto, in formato shar per l' architettura ALPHA o in formato rpm per la piattaforma INTEL, scaricabili da browser. In entrambi i casi, per poter installare il materiale e` necessario disporre dei privilegi di root. L'utente condor creato contiene la seguente struttura:

- ~condor/execute/
- ~condor/log/
- ~condor/spool/
- ~condor/condor_config
- ~condor/$(HOSTNAME)/condor_config.local

3.3 Installazione completa

Questa installazione comprende la distribuzione di librerie, binari e file di configurazione, la creazione di un utente unix condor locale, la attivazione del demone al boot e la definizione delle priorita` locali. Vengono richiesti i parametri di configurazione (fig. 4 in appendice B) come:

- USE_CKPT_SERVER
- DEFAULT_DOMAIN_NAME
- CONDOR_ADMIN
- SUBMIT_GROUP_ID
- HOSTALLOW_ADMINISTRATION

e quelli necessari alla gestione come:

- OPSYS
- CONDOR_HOME

A seconda del tipo di sistema operativo selezionato, sono preparati pacchetti di differente formato: tar file (con la distribuzione del software) e shar file (con comandi di personalizzazione) per architettura ALPHA e due pacchetti rpm (uno per la distribuzione e uno per le personalizzazioni) per architettura INTEL.

In questo tipo di installazione non e` possibile modificare il parametro RELEASE_DIR, percio` il software e i file di configurazione vengono installati in /usr/condor/$OPSYS mentre i file di configurazione principali (condor_config, condor_config.policy, site.infn.it) in /usr/condor/etc/

4 FILE GENERALI DI CONFIGURAZIONE E LORO PARAMETRI

Al fine di ottimizzare le risorse e l'utilizzo del pool, nel corso degli anni sono stati modificati i file di configurazione originali secondo necessita` comuni.

Requisiti fondamentali, per un buon funzionamento generale, sono rappresentati da omogeneita` di software e politica di utilizzo delle risorse. Proprio per questo, tale tool oltre ad essere di aiuto per la sua facile applicazione, e` uno strumento di controllo.

L'attivazione del software ha essenzialmente bisogno di un paio di file di configurazione e di uno script di boot, ma abbiamo voluto suddividere i parametri in ulteriori file per snellire e rendere dinamica la configurazione principale. Pertanto sono stati predisposti file di configurazione per sistema operativo, per sezione, per politica di esecuzione e di priorita`. Rimane comunque, ovviamente, locale la scelta di come e dove implementare il software di *Condor* in quanto le esigenze e le opportunita` sono estremamente differenti. Abbiamo comunque individuato pochi metodi di gestione che ci sono sembrati funzionali e semplici e preparato opportune interfacce web di richiesta informazioni.

I dati forniti, tramite i form, integrano parametri organizzativi e di configurazione di *Condor*: il condor-admin locale sceglie se distribuire il software centralmente o installarlo singolarmente su ogni macchina, se attivare priorita` di esecuzione differenti nell'ambito del proprio sottoinsieme di macchine, etc. Per applicare tutto cio` il tool utilizza alcuni file template di configurazione che vengono personalizzati durante la sua elaborazione. Riportiamo l'elenco dei file template utilizzati e i loro parametri configurabili:

condor_config	RELEASE_DIR USE_CKPT_SERVER OPSYS
condor_config.ALPHA.OSF1	(parametri statici)
condor_config.INTEL.LINUX	(parametri statici)
condor_config.policy	(parametri statici)
site.infn.it	CONDOR_ADMIN DEFAULT_DOMAIN_NAME HOSTALLOW_ADMINISTRATION
condor_config.local	SUBMIT_GROUP_ID
condor.csh	RELEASE_DIR
condor.sh	RELEASE_DIR
condor.OSF1	RELEASE_DIR
condor.LINUX	RELEASE_DIR

i singoli file di configurazione vengono descritti di seguito.

- condor_config: configurazione globale, suddivisa per sezioni in cui vengono definiti central manager, directory locali di installazione, parametri di rete, autorizzazioni di accesso al pool, uso di checkpoint server, file di output e binari dei demoni, etc.
- condor_config.ALPHA.OSF1, condor_config.INTEL.LINUX: elenco dei demoni da attivare, mail, device di console, distinti per architettura.
- *site*.infn.it: file specifico per dominio (bo.infn.it, na.infn.it,…), include il dominio, il recapito email del condor-admin locale, il ckpt server, locale o piu` vicino, a disposizione del sottoinsieme di macchine.
- condor_config.local: file specifico della macchina, definisce le priorita` dei job nell'ambito locale; questa scelta e` data al condor-admin.
- condor.sh, condor.csh: script linux di login per la definizione del path con binari e librerie di *Condor*.
- condor.OSF1, condor.LINUX: script di boot distinto per architettura e i parametri configurabili tramite i form.
- RELEASE_DIR: indica la directory contenente binari, librerie e file di configurazione di *Condor*; puo` essere di tipo locale o remoto. Nel caso server, contiene distribuzioni differenziate per architettura.
- OPSYS: indica il sistema operativo della macchina in cui viene installato il software, le voci prese in considerazione sono OSF1, LINUX, LINUX-GLIBC.
- USE_CKPT_SERVER: setta l' uso o meno di un server per il deposito di checkpoint file. Nel pool INFN sono state configurate alcune macchine con queste funzioni, a copertura di tutto il territorio; le sezioni che non dispongono di un server nella propria LAN utilizzano quello piu` vicino. Il tool e` gia` stato istruito nell'individuare i ckpt server geografici.
- CONDOR_HOME: indica la directory dell'utente condor. Tale parametro viene utilizzato dagli script, generati dalle form, al momento della creazione dell'utente locale; nel file di configurazione generale, condor_config, equivale a $(TILDE).
- CONDOR_ADMIN: recapito del condor-admin locale a cui vengono inviate segnalazioni di errori o di diagnostica del sottoinsieme di macchine del dominio.
- HOSTALLOW_ADMINISTRATION: macchina, del sottoinsieme locale, da cui possono essere schedulati comandi *Condor* di amministrazione verso il sottoinsieme stesso. E` bene che tali macchine non posseggano utenza generica. Il CM, di default, puo` amministrare remotamente tutti i sistemi condor del pool.
- DEFAULT_DOMAIN_NAME: settaggio del dominio internet locale. Questo campo se non e` stato definito viene comunque settato utilizzando quello di provenienza del browser da cui e` stato eseguita la form.
- SUBMIT_GROUP_ID: definizione di un gruppo, con priorita` superiore di utilizzo delle risorse della macchina, tramite *Condo*r. In questo modo e` possibile favorire l' esecuzione di job su macchine di proprieta` di gruppi o esperimenti.

5 DESCRIZIONE DEL CGI SCRIPT

L'interfaccia grafica, o meglio il form, una volta compilata, richiama uno script perl che esegue le seguenti operazioni:

- se viene scelto di usare il checkpoint server, sceglie il nome della macchina che fa da checkpoint server in base al dominio, utlizzando un piccolo database interno
- copia i template dei file di configurazione, che sono contenuti in un repository sul web server, in una directory temporanea sempre del web server ed esegue le sostituzioni necessarie per customizzare l'installazione a seconda dei parametri contenuti nel form,
- crea i file rpm o shar, che a loro volta, quando eseguiti:
  - creano l'utente e il gruppo condor ,
  - creano le opportune directory con le necessarie protezioni e vi depositano i file, binari e/o di configurazione,
  - creano i file che permettono l'avvio automatico di *Condor* al boot della macchina (per esempio per Linux RedHat `/etc/rc.d/init.d/condor` e i corrispondenti link nei diversi run level, per esempio `/etc/rc.d/rc5.d/S99condor`),
  - creano i file che inseriscono automaticamente le directory dei binari nella variabile PATH dell'utente (es: `/etc/profile.d/condor.sh`)
- crea la pagina html finale che visualizza i dati forniti dall'utente, eventuali warning, i link con i file (rpm o shar) da scaricare e da installare e alcune semplici istruzioni per l'installazione.

La struttura delle directory dell'utente condor e' la seguente:

- ~condor/*nome-pc*/log/
- ~condor/*nome-pc*/execute/
- ~condor/*nome-pc*/spool/

dove "nome-pc" e' il nome della macchina su cui viene installato *Condor*. Dato che al momento della creazione del file rpm o shar non e' possibile conoscere il nome della macchina, e' lo script che viene eseguito all'interno di questi file che riconosce il nome della macchina e modifica "nome-pc" col nome effettivo al momento dell'installazione.

Nella home directory dell'utente condor vi e' inoltre il file `condor_config` che e' un link al file di configurazione globale di *Condor*, presente nella directory di installazione specificata dall'utente nel form, mentre nella directory "nome-pc" vi e' il file di configurazione locale per quella particolare macchina: `condor_config.local`.

I file rpm o shar vengono creati sul web server dall'utente 'apache' (o comunque dall' owner del processo httpd) e non da root, pertanto non possono essere costruiti con i file binari e di configurazione gia' nella directory specificata dall'utente nel form, non avendo i privilegi per creare una directory arbitraria (non si e' voluto dare privilegi particolari all'utente 'apache' per motivi di sicurezza). Viene quindi costruita sul web server una struttura di directory in /tmp, copiati i file necessari per l'installazione, e quindi creati con essi

i file rpm o shar. Al momento dell'installazione, quindi, sulla macchina dell'utente vengono prima creati i file nella locale /tmp, e poi e' compito degli script eseguiti dai file rpm e shar di ricreare la struttura di directory effettiva, e di spostarvi i file contenuti nelle directory provvisorie sotto /tmp, che vengono infine rimosse. Questo e' un processo che naturalmente si svolge in maniera del tutto trasparente all'utente.

6 RINGRAZIAMENTI

Si ringrazia tutto il *Condor Team*, del Dipartimento di Computer Science, Madison Wisconsin, che in questi anni ci ha assistito nella implementazione e configurazione del software.

7 RIFERIMENTI

In aggiunta ai riferimenti dati nel corso del documento si segnalano i seguenti link:

- http://www.cs.wisc.edu/condor/
- http://www.cs.wisc.edu/condor/manual/v6.4/

APPENDICE A

In questa sessione sono pubblicati i file di configurazione o startup, utilizzati dalla procedura, senza alcuna personalizzazione. Alcuni di questi template non necessitano di variazioni.

A.1 Template del file condor_config

```
#
# D. Bortolotti (16-Aug-99)
# La definizione di alcune variabili e` stata spostata in:
# $(SITES), $(POLICY), $(PLATFORM)
#
# SITES : ogni dominio deve disporre di un file personalizzato
#         Bologna (bo.infn.it)      ---> bo.infn.it.local
#         Milano  (mi.infn.it)      ---> mi.infn.it.local
#
# POLICY : politica generale e comune di run, stop e checkpoint dei jobs
# PLATFORM : variabili in funzione dell'architettura e sistema
#
# RELEASE_DIR ed ETC rappresentano:
#
# i directory LOCALI (di sezione) in cui vengono depositati binari e
# file di configurazione
# (nell'esempio sotto sono dichiarati i dir AFS di divulgazione release)
#
# La variabile LOCAL_DIR e` per tutti $(TILDE)/$(HOSTNAME), in questo
# modo si comprendono entrambi i casi di:
#
# - un account di condor per macchina
# - un solo account di condor per N macchine
##########################################################################
##
##  condor_config
##
##  This is the global configuration file for condor.
##
##  The file is divided into four main parts:
##  Part 1:  Settings you MUST customize
##  Part 2:  Settings you may want to customize
##  Part 3:  Settings that control the policy of when condor will
##           start and stop jobs on your machines
```

```
##  Part 4:  Settings you should probably leave alone (unless you
##  know what you're doing)
##
##  Please read the INSTALL file (or the Install chapter in the
##  Condor Administrator's Manual) for detailed explanations of the
##  various settings in here and possible ways to configure your
##  pool.
##
##  If you are installing Condor as root and then handing over the
##  administration of this file to a person you do not trust with
##  root access, please read the Installation chapter paying careful
##  note to the condor_config.root entries.
##
##  Unless otherwise specified, settings that are commented out show
##  the defaults that are used if you don't define a value.  Settings
##  that are defined here MUST BE DEFINED since they have no default
##  value.
##
##  Unless otherwise indicated, all settings which specify a time are
##  defined in seconds.
##
##  Part 1:  Settings you must customize:
##
##  What machine is your central manager?
CONDOR_HOST                = cmcondor.bo.infn.it
CONDOR_VIEW_HOST       = cmcondor.bo.infn.it
##  Pathnames:
##  Where have you installed the bin, sbin and lib condor directories?
RELEASE_DIR          =
##  Where is the local condor directory for each host?
LOCAL_DIR                  = $(TILDE)/$(HOSTNAME)
##  Where is the machine-specific local config file for each host?
#
ETC_DIR                    =
#SITES                     = $(ETC_DIR)/sites/site.infn.it
SITES                      =
#++
POLICY                     = $(ETC_DIR)/condor_config.policy
PLATFORM                   = $(ETC_DIR)/condor_config.$(ARCH).$(OPSYS)
LOCAL                      = $(LOCAL_DIR)/condor_config.local
LOCAL_CONFIG_FILE    = $(SITES), $(POLICY), $(PLATFORM), $(LOCAL)
# D. Bortolotti 21/11/2002
```

```
LOWPORT                 = 32000
HIGHPORT                = 32500
##  Mail parameters:
##
##  When something goes wrong with condor at your site, who should get
##  the email?
#CONDOR_ADMIN          = condor-admin@your.domain
##  Full path to a mail delivery program that understands that "-s"
##  means you want to specify a subject:
#MAIL                   = /usr/bin/mail
##--------------------------------------------------------------------
##  Network domain parameters:
##--------------------------------------------------------------------
##  Internet domain of machines sharing a common UID space.  If your
##  machines don't share a common UID space, use the second entry
##  which specifies that each machine has its own UID space.
UID_DOMAIN              = $(FULL_HOSTNAME)
##  Internet domain of machines sharing a common file system.
##  If your machines don't use a network file system, use the second
##  entry which specifies that each machine has its own file system.
FILESYSTEM_DOMAIN   = $(FULL_HOSTNAME)
##
##  Part 2:  Settings you may want to customize:
##  (it is generally safe to leave these untouched)
##--------------------------------------------------------------------
##  Flocking: Submitting jobs to more than one pool
##--------------------------------------------------------------------
##  Flocking allows you to run your jobs in other pools, or lets
##  others run jobs in your pool.

##  pools they will run in.)
FLOCK_TO =
##  An example of this is:
#FLOCK_TO = central_manager.friendly.domain, condor.cs.wisc.edu
##
##  To let others flock to you, define FLOCK_FROM.
##
##  To flock to others, define FLOCK_TO.
##
##  (and if they are running a condor_view server, then you can also
##  define FLOCK_VIEW_SERVERS).
```

```
##  FLOCK_FROM defines the machines where you would like to grant
##  people access to your pool via flocking. (i.e. you are granting
##  access to these machines to join your pool).
FLOCK_FROM =
##  An example of this is:
#FLOCK_FROM = somehost.friendly.domain, anotherhost.friendly.domain
##  FLOCK_TO defines the central managers of the pools that you want
##  to flock to. (i.e. you are specifying the machines that you
##  want your jobs to be negotiated at -- thereby specifying the
##  FLOCK_COLLECTOR_HOSTS should almost always be the same as
##  FLOCK_NEGOTIATOR_HOSTS (as shown below).  The only reason it would be
##  different is if the collector and negotiator in the pool that you are
##  flocking too are running on different machines (not recommended).
##  The collectors must be specified in the same corresponding order as
##  the FLOCK_NEGOTIATOR_HOSTS list.
FLOCK_NEGOTIATOR_HOSTS = $(FLOCK_TO)
FLOCK_COLLECTOR_HOSTS = $(FLOCK_TO)
## An example of having the negotiator and the collector on different
## machines is:
#FLOCK_NEGOTIATOR_HOSTS = condor.cs.wisc.edu, …
#FLOCK_COLLECTOR_HOSTS =  condor.cs.wisc.edu, …
##  (optional) FLOCK_VIEW_SERVERS is used if you would like to report
##  your usage in a remote pool to a remote condor-view server.  If so,
##  FLOCK_VIEW_SERVERS should contain a list of hostnames where the
##  condor-view server is running in the pools to which you want your
##  jobs to flock.  The order of this list must correspond to the
##  order of the FLOCK_COLLECTOR_HOSTS and FLOCK_NEGOTIATOR_HOSTS
##  lists.  List items may be empty for pools which don't use a
##  separate condor-view server.  FLOCK_VIEW_SERVER may be left
##  undefined if no remote pools use separate condor-view servers.
##  Note: It is required that the same hostname does not appear twice
##  in the FLOCK_VIEW_SERVERS list and that the CONDOR_VIEW_HOST does
##  not appear in the FLOCK_VIEW_SERVERS list.
#FLOCK_VIEW_SERVERS = condor-view.cs.wisc.edu, condor-view.friendly.domain
##
##  Host/IP access levels
##
##  Please see the administrator's manual for details on these
##  settings, what they're for, and how to use them.
```

```
##  What machines have administrative rights for your pool?  This
##  defaults to your central manager.  You should set it to the
##  machine(s) where whoever is the condor administrator(s) works
##  (assuming you trust all the users who log into that/those
##  machine(s), since this is machine-wide access you're granting).
HOSTALLOW_ADMINISTRATOR = $(CONDOR_HOST)
##  If there are no machines that should have administrative access
##  to your pool (for example, there's no machine where only trusted
##  users have accounts), you can uncomment this setting.
##  Unfortunately, this will mean that administering your pool will
##  be more difficult.
#HOSTDENY_ADMINISTRATOR = *
##  What machines should have "owner" access to your machines, meaning
##  they can issue commands that a machine owner should be able to
##  issue to their own machine (like condor_vacate).  This defaults to
##  machines with administrator access, and the local machine.  This
##  is probably what you want.
HOSTALLOW_OWNER = \
$(FULL_HOSTNAME),$(HOSTALLOW_ADMINISTRATOR)
##  Read access.  Machines listed as allow (and/or not listed as deny)
##  can view the status of your pool, but cannot join your pool
##  or run jobs.
##  NOTE: By default, without these entries customized, you
##  are granting read access to the whole world.  You may want to
##  restrict that to hosts in your domain.  If possible, please also
##  grant read access to "*.cs.wisc.edu", so the Condor developers
##  will be able to view the status of your pool and more easily help
##  you install, configure or debug your Condor installation.
##  It is important to have this defined.
HOSTALLOW_READ = *.infn.it, *.cs.wisc.edu
#HOSTALLOW_READ = *.your.domain, *.cs.wisc.edu
#HOSTDENY_READ = *.bad.subnet, bad-machine.your.domain, 144.77.88.*
##  Write access.  Machines listed here can join your pool, submit
##  jobs, etc.  Note: Any machine which has WRITE access must
##  also be granted READ access.  Granting WRITE access below does
##  not also automatically grant READ access; you must change
##  HOSTALLOW_READ above as well.
##  If you leave it as it is, it will be unspecified, and effectively
##  it will be allowing anyone to write to your pool.
HOSTALLOW_WRITE = *.infn.it, *.cs.wisc.edu
```

```
#HOSTALLOW_WRITE = *.your.domain, your-friend's-machine.other.domain
#HOSTDENY_WRITE = bad-machine.your.domain
##  Negotiator access.  Machines listed here are trusted central
##  managers.  You should normally not have to change this.
HOSTALLOW_NEGOTIATOR = $(NEGOTIATOR_HOST)
##  Now, with flocking we need to let the SCHEDD trust the other
##  negotiators we are flocking with as well.  You should normally
##  not have to change this either.
HOSTALLOW_NEGOTIATOR_SCHEDD = $(NEGOTIATOR_HOST),\
$(FLOCK_NEGOTIATOR_HOSTS)
##  Config access.  Machines listed here can use the condor_config_val
##  tool to modify all daemon configurations except those specified in
##  the condor_config.root file.  This level of host-wide access
##  should only be granted with extreme caution.  By default, config
##  access is denied from all hosts.
#HOSTALLOW_CONFIG = trusted-host.your.domain
##  Flocking Configs.  These are the real things that Condor looks at,
##  but we set them from the FLOCK_FROM/TO macros above.  It is safe
##  to leave these unchanged.
HOSTALLOW_WRITE_COLLECTOR = $(HOSTALLOW_WRITE), \
$(FLOCK_FROM)
HOSTALLOW_WRITE_STARTD    = $(HOSTALLOW_WRITE), $(FLOCK_FROM)
HOSTALLOW_READ_COLLECTOR  = $(HOSTALLOW_READ), $(FLOCK_FROM)
HOSTALLOW_READ_STARTD     = $(HOSTALLOW_READ), $(FLOCK_FROM)
##--------------------------------------------------------------------
##  Security parameters for setting configuration values remotely:
##--------------------------------------------------------------------
##  These parameters define the list of attributes that can be set
##  remotely with condor_config_val for the security access levels
##  defined above (for example, WRITE, ADMINISTRATOR, CONFIG, etc).
##  Please see the administrator's manual for futher details on these
##  settings, what they're for, and how to use them.  There are no
##  default values for any of these settings.  If they are not
##  defined, no attributes can be set with condor_config_val.
##  Attributes that can be set by hosts with "CONFIG" permission (as
##  defined with HOSTALLOW_CONFIG and HOSTDENY_CONFIG above).
##  The commented-out value here was the default behavior of Condor
##  prior to version 6.3.3.  If you don't need this behavior, you
##  should leave this commented out.
#SETTABLE_ATTRS_CONFIG = *
```

```
##  Attributes that can be set by hosts with "ADMINISTRATOR"
##  permission (as defined above)
#SETTABLE_ATTRS_ADMINISTRATOR = *_DEBUG, MAX_*_LOG
##  Attributes that can be set by hosts with "OWNER" permission (as
##  defined above) NOTE: any Condor job running on a given host will
##  have OWNER permission on that host by default.  If you grant this
##  kind of access, Condor jobs will be able to modify any attributes
##  you list below on the machine where they are running.  This has
##  obvious security implications, so only grant this kind of
##  permission for custom attributes that you define for your own use
##  at your pool (custom attributes about your machines that are
##  published with the STARTD_EXPRS setting, for example).
#SETTABLE_ATTRS_OWNER = your_custom_attribute, another_custom_attr
##  You can also define daemon-specific versions of each of these
##  settings.  For example, to define settings that can only be
##  changed in the condor_startd's configuration by hosts with OWNER
##  permission, you would use:
#STARTD_SETTABLE_ATTRS_OWNER = your_custom_attribute_name
##--------------------------------------------------------------------
##  Network filesystem parameters:
##--------------------------------------------------------------------
##  Do you want to use NFS for file access instead of remote system
##  calls?
#USE_NFS              = False
##  Do your machines run AFS?
#HAS_AFS              = False
##  Do you want to use AFS for file access instead of remote system
##  calls?
#USE_AFS              = False
##--------------------------------------------------------------------
##  Checkpoint server:
##--------------------------------------------------------------------
##  Do you want to use a checkpoint server if one is available?  If a
##  checkpoint server isn't available or USE_CKPT_SERVER is set to
##  False, checkpoints will be written to the local SPOOL directory on
##  the submission machine.
USE_CKPT_SERVER        =
##  What's the hostname of this machine's nearest checkpoint server?
#CKPT_SERVER_HOST    = checkpoint-server-hostname.your.domain
##  Do you want the starter on the execute machine to choose the
```

```
##  checkpoint server?  If False, the CKPT_SERVER_HOST set on
##  the submit machine is used.  Otherwise, the CKPT_SERVER_HOST set
##  on the execute machine is used.  The default is true.
STARTER_CHOOSES_CKPT_SERVER = True
##--------------------------------------------------------------------
##  Miscellaneous:
##--------------------------------------------------------------------
##  Try to save this much swap space by not starting new shadows.
##  Specified in megabytes.
#RESERVED_SWAP                  = 5
##  What's the maximum number of jobs you want a single submit machine
##  to spawn shadows for?
#MAX_JOBS_RUNNING  = 200
##  Condor needs to create a few lock files to synchronize access to
##  various log files.  Because of problems we've had with network
##  filesystems and file locking over the years, we HIGHLY recommend
##  that you put these lock files on a local partition on each
##  machine.  If you don't have your LOCAL_DIR on a local partition,
##  be sure to change this entry.  Whatever user (or group) condor is
##  running as needs to have write access to this directory.  If
##  you're not running as root, this is whatever user you started up
##  the condor_master as.  If you are running as root, and there's a
##  condor account, it's probably condor.  Otherwise, it's whatever
##  you've set in the CONDOR_IDS environment variable.  See the Admin
##  manual for details on this.
LOCK                    = $(LOCAL_DIR)/log
##  If you don't use a fully qualified name in your /etc/hosts file
##  (or NIS, etc.) for either your official hostname or as an alias,
##  Condor wouldn't normally be able to use fully qualified names in
##  places that it'd like to.  You can set this parameter to the
##  domain you'd like appended to your hostname, if changing your host
##  information isn't a good option.  This parameter must be set in
##  the global config file (not the LOCAL_CONFIG_FILE from above).
#DEFAULT_DOMAIN_NAME = your.domain.name
##  Condor can be told whether or not you want the Condor daemons to
##  create a core file if something really bad happens.  This just
##  sets the resource limit for the size of a core file.  By default,
##  we don't do anything, and leave in place whatever limit was in
##  effect when you started the Condor daemons.  If this parameter is
##  set and "True", we increase the limit to as large as it gets.  If
```

```
##  it's set to "False", we set the limit at 0 (which means that no
##  core files are even created).  Core files greatly help the Condor
##  developers debug any problems you might be having.
#CREATE_CORE_FILES    = True
##--------------------------------------------------------------------
##  Settings that control the daemon's debugging output:
##--------------------------------------------------------------------
##
## The flags given in ALL_DEBUG are shared between all daemons.
##
ALL_DEBUG                       =
MAX_COLLECTOR_LOG               = 64000
COLLECTOR_DEBUG                 =
MAX_KBDD_LOG                    = 64000
KBDD_DEBUG                      =
MAX_NEGOTIATOR_LOG              = 64000
NEGOTIATOR_DEBUG                =
MAX_SCHEDD_LOG                  = 64000
SCHEDD_DEBUG                    = D_COMMAND
MAX_SHADOW_LOG                  = 64000
SHADOW_DEBUG                    =
MAX_STARTD_LOG                  = 64000
STARTD_DEBUG                    = D_COMMAND
MAX_STARTER_LOG                 = 64000
STARTER_DEBUG                   = D_NODATE
MAX_MASTER_LOG                  = 64000
MASTER_DEBUG                    = D_COMMAND
##  When the master starts up, should it truncate it's log file?
#TRUNC_MASTER_LOG_ON_OPEN       = False
##
##  Part 3:  Settings control the policy for running, stopping, and
##  periodically checkpointing condor jobs:
##
##  This section contains macros are here to help write legible
##  expressions:
#
#HOUR                   = (60 * $(MINUTE))
#StateTimer             = (CurrentTime - EnteredCurrentState)
#ActivityTimer          = (CurrentTime - EnteredCurrentActivity)
#ActivationTimer        = (CurrentTime - JobStart)
```

```
#LastCkpt                       = (CurrentTime - LastPeriodicCheckpoint)
##  The JobUniverse attribute is just an int.  These macros can be
##  used to specify the universe in a human-readable way:
#STANDARD                       = 1
#PVM                            = 4
#VANILLA                        = 5
#IsPVM                          = (JobUniverse == $(PVM))
#IsVanilla                      =JobUniverse == $(VANILLA))
#IsStandard                     =JobUniverse == $(STANDARD))
#NonCondorLoadAvg               = (LoadAvg - CondorLoadAvg)
#BackgroundLoad                 = 0.3
#HighLoad                       = 0.5
#StartIdleTime                  = 15 * $(MINUTE)
#ContinueIdleTime=  5 * $(MINUTE)
#MaxSuspendTime                 = 10 * $(MINUTE)
#MaxVacateTime                  = 10 * $(MINUTE)
#KeyboardBusy                   = (KeyboardIdle < $(MINUTE))
#ConsoleBusy                    = (ConsoleIdle  < $(MINUTE))
#CPU_Idle                       = ($(NonCondorLoadAvg) <= $(BackgroundLoad))
#CPU_Busy                       = ($(NonCondorLoadAvg) >= $(HighLoad))
#BigJob                         =(ImageSize >= (50 * 1024))
#MediumJob                      = (ImageSize >= (15 * 1024) && ImageSize < (50 * 1024))
#SmallJob                       = (ImageSize <  (15 * 1024))
#JustCPU                        = ($(CPU_Busy) && ($(KeyboardBusy) == False))
#MachineBusy                    = ($(CPU_Busy) || $(KeyboardBusy))
##  The RANK expression controls which jobs this machine prefers to
##  run over others.  Some examples from the manual include:
##    RANK = ImageSize
##    RANK = (Owner == "coltrane") + (Owner == "tyner") \
##                 + ((Owner == "garrison") * 10) + (Owner == "jones")
##  By default, RANK is always 0, meaning that all jobs have an equal
##  ranking.
#RANK                           = 0
##
##  This where you choose the configuration that you would like to
##  use.  It has no defaults so it must be defined.  We start this
##  file off with the UWCS_* policy.
##
##  Also here is what is referred to as the TESTINGMODE_*, which is
##  a quick hardwired way to test Condor.
```

```
##  Replace UWCS_* with TESTINGMODE_* if you wish to do testing mode.
##  For example:
##  WANT_SUSPEND            = $(UWCS_WANT_SUSPEND)
##  becomes
##  WANT_SUSPEND            = $(TESTINGMODE_WANT_SUSPEND)
#WANT_SUSPEND            = $(UWCS_WANT_SUSPEND)
#WANT_VACATE             = $(UWCS_WANT_VACATE)
#START                   = $(UWCS_START)
#SUSPEND                 = $(UWCS_SUSPEND)
#CONTINUE                = $(UWCS_CONTINUE)
#PREEMPT                 = $(UWCS_PREEMPT)
#KILL                    = $(UWCS_KILL)
#PERIODIC_CHECKPOINT     = $(UWCS_PERIODIC_CHECKPOINT)
#PREEMPTION_REQUIREMENTS = $(UWCS_PREEMPTION_REQUIREMENTS)
#PREEMPTION_RANK         = $(UWCS_PREEMPTION_RANK)
##
## This is the UWisc - CS Department Configuration.
##
#UWCS_WANT_SUSPEND       = ( $(SmallJob) || $(JustCpu) || $(IsPVM) \ ||
$(IsVanilla) )
#UWCS_WANT_VACATE        = $(ActivationTimer) > 10 * $(MINUTE) \
                          || $(IsPVM) || $(IsVanilla)
##  The following START expression is the exact start expression that
##  we use here at UWCS.  Here at UWCS we require that people declare
##  how much RAM they are going to use (as we have a number of
##  machines that have a low amount of RAM).  To do so, they put the
##  following into their command file (what they give to condor_submit):
##  +MemoryRequirements = 12
##  (Where the first ## is not there, and 12 is the number of megabytes
##  of RAM they are going to use)
##  If they do NOT specify this (which is often), we'll assume that they'll
##  need 113 megabytes of RAM (128 - 15 megabytes that we typically save for the
##  operating system.)
##  So, the expression below is, if the CPU & Keyboard are both idle,
##  and the MemoryRequirements fit (or are not defined and this machine
##  has 128 or greater megabytes of RAM), then the job can start here.
##
##  If you decide to use this, please be aware that if you do not
##  have any (or many) machines with 128 megabytes (or more) of RAM, your
##  jobs will *NOT* start unless you specify MemoryRequirements in
```

```
##  your command file!
#UWCS_START             = ( ($(CPU_Idle) && KeyboardIdle > $(StartIdleTime)) \
#                           && (TARGET.ImageSize <= ((Memory - 15)*1024)) \
#                           && ( (MemoryRequirements < (Memory - 15)) || \
#                           (MemoryRequirements =?= UNDEFINED && \
#                           (RemoteUserCpu > 0.0 || Memory > 127)) ) )
##  An easier-to-use start expression follows.  It is only dependent on
##  if the CPU is idle, and if the keyboard has been idle for a while.
##  (The memory requirements are implied to be that the amount of RAM
##  the incoming job requires will be less than the amount of memory on
##  this machine).
#UWCS_START             = $(CPU_Idle) && KeyboardIdle > $(StartIdleTime)
#UWCS_SUSPEND           = $(MachineBusy)
#UWCS_CONTINUE          = $(CPU_Idle) && KeyboardIdle > $(ContinueIdleTime)
#UWCS_PREEMPT           = ( ($(ActivityTimer) > $(MaxSuspendTime)) && \
                           (Activity == "Suspended") ) || \
                           ( SUSPEND && (WANT_SUSPEND == False) )
#UWCS_KILL              = $(ActivityTimer) > $(MaxVacateTime)
##  Only define vanilla versions of these if you want to make them
##  different from the above settings.
#SUSPEND_VANILLA        = $(MachineBusy)
#CONTINUE_VANILLA       = $(CPU_Idle) && KeyboardIdle > $(ContinueIdleTime)
#PREEMPT_VANILLA        = ( ($(ActivityTimer) > $(MaxSuspendTime)) && \
                           (Activity == "Suspended") ) || \
                          ( SUSPEND_VANILLA && (Activity != "Suspended") )
#KILL_VANILLA           = $(ActivityTimer) > $(MaxVacateTime)
##  We use a simple Periodic checkpointing mechanism, but then
##  again we have a very fast network.
#UWCS_PERIODIC_CHECKPOINT        = $(LastCkpt) > (3 * $(HOUR))
##  You might want to checkpoint a little less often.  A good
##  example of this is below.  For jobs smaller than 60 megabytes, we
##  periodic checkpoint every 6 hours.  For larger jobs, we only
##  checkpoint every 12 hours.
#UWCS_PERIODIC_CHECKPOINT        = ( (ImageSize < 60000) && \
#                                    ($(LastCkpt) > (6 * $(HOUR))) ) || \
#                                  ( $(LastCkpt) > (12 * $(HOUR)) )
##  The negotiator will not preempt a job running on a given machine
##  unless the PREEMPTION_REQUIREMENTS expression evaluates to true
##  and the owner of the idle job has a better priority than the owner
##  of the running job.  This expression defaults to true.
```

```
#UWCS_PREEMPTION_REQUIREMENTS = $(StateTimer) > (1 * $(HOUR)) &&
RemoteUserPrio > SubmittorPrio * 1.2
##  The PREEMPTION_RANK expression is used to rank machines which the
##  job ranks the same.  For example, if the job has no preference, it
##  is usually preferable to preempt a job with a small ImageSize
##  instead of a job with a large ImageSize.  The default is to rank
##  all preemptable matches the same.  However, the negotiator will
##  always prefer to match the job with an idle machine over a
##  preemptable machine, if the job has no preference between them.
#UWCS_PREEMPTION_RANK = (RemoteUserPrio * 1000000) - ImageSize
##########################################################################
##  This is a Configuration that will cause your Condor jobs to
##  always run.  This is intended for testing only.
##########################################################################
##  This mode will cause your jobs to start on a machine an will let
##  them run to completion.  Condor will ignore all of what is going
##  on in the machine (load average, keyboard activity, etc.)
#TESTINGMODE_WANT_SUSPEND        = False
#TESTINGMODE_WANT_VACATE         = False
#TESTINGMODE_START               = True
#TESTINGMODE_SUSPEND             = False
#TESTINGMODE_CONTINUE            = True
#TESTINGMODE_PREEMPT             = False
#TESTINGMODE_KILL                = False
#TESTINGMODE_PERIODIC_CHECKPOINT         = False
#TESTINGMODE_PREEMPTION_REQUIREMENTS = False
#TESTINGMODE_PREEMPTION_RANK = 0
##
##  Part 4:  Settings you should probably leave alone:
##  (unless you know what you're doing)
##  Daemon-wide settings:
##########################################################################
##  Pathnames
LOG             = $(LOCAL_DIR)/log
SPOOL           = $(LOCAL_DIR)/spool
EXECUTE         = $(LOCAL_DIR)/execute
BIN             = $(RELEASE_DIR)/bin
LIB             = $(RELEASE_DIR)/lib
SBIN            = $(RELEASE_DIR)/sbin
HISTORY         = $(SPOOL)/history
```

```
## Log files
COLLECTOR_LOG= $(LOG)/CollectorLog
KBDD_LOG        = $(LOG)/KbdLog
MASTER_LOG    = $(LOG)/MasterLog
NEGOTIATOR_LOG= $(LOG)/NegotiatorLog
SCHEDD_LOG    = $(LOG)/SchedLog
SHADOW_LOG  = $(LOG)/ShadowLog
STARTD_LOG    = $(LOG)/StartLog
STARTER_LOG   = $(LOG)/StarterLog
## Lock files
SHADOW_LOCK             = $(LOCK)/ShadowLock
## In most cases, your condor_collector and condor_negotiator are
## going to run on the same machine.  If for some reason, this isn't
## the case, here's where you'd change it:
COLLECTOR_HOST        = $(CONDOR_HOST)
NEGOTIATOR_HOST       = $(CONDOR_HOST)
## How long are you willing to let daemons try their graceful
## shutdown methods before they do a hard shutdown? (30 minutes)
#SHUTDOWN_GRACEFUL_TIMEOUT       = 1800
## How much disk space would you like reserved from Condor?  In
## places where Condor is computing the free disk space on various
## partitions, it subtracts the amount it really finds by this
## many megabytes.  (If undefined, defaults to 0).
RESERVED_DISK                   = 5
## If your machine is running AFS and the AFS cache lives on the same
## partition as the other Condor directories, and you want Condor to
## reserve the space that your AFS cache is configured to use, set
## this to true.
#RESERVE_AFS_CACHE  = False
## By default, if a user does not specify "notify_user" in the submit
## description file, any email Condor sends about that job will go to
## "username@UID_DOMAIN".  If your machines all share a common UID
## domain (so that you would set UID_DOMAIN to be the same across all
## machines in your pool), *BUT* email to user@UID_DOMAIN is *NOT*
## the right place for Condor to send email for your site, you can
## define the default domain to use for email.  A common example
## would be to set EMAIL_DOMAIN to the fully qualified hostname of
## each machine in your pool, so users submitting jobs from a
## specific machine would get email sent to user@machine.your.domain,
## instead of user@your.domain.  In general, you should leave this
```

```
##  setting commented out unless two things are true: 1) UID_DOMAIN is
##  set to your domain, not $(FULL_HOSTNAME), and 2) email to
##  user@UID_DOMAIN won't work.
#EMAIL_DOMAIN = $(FULL_HOSTNAME)
##
##  Daemon-specific settings:
##--------------------------------------------------------------------
##  condor_master
##--------------------------------------------------------------------
##  Daemons you want the master to keep running for you:
DAEMON_LIST                     = MASTER, STARTD, SCHEDD
##  Which daemons use the Condor DaemonCore library (i.e., not the
##  checkpoint server or custom user daemons)?
#DC_DAEMON_LIST = MASTER, STARTD, SCHEDD, KBDD, COLLECTOR,
NEGOTIATOR, EVENTD
##  Where are the binaries for these daemons?
MASTER                          = $(SBIN)/condor_master
STARTD                          = $(SBIN)/condor_startd
SCHEDD                          = $(SBIN)/condor_schedd
KBDD                            = $(SBIN)/condor_kbdd
##  When the master starts up, it can place it's address (IP and port)
##  into a file.  This way, tools running on the local machine don't
##  need to query the central manager to find the master.  This
##  feature can be turned off by commenting out this setting.
MASTER_ADDRESS_FILE = $(LOG)/.master_address
##  Where should the master find the condor_preen binary? If you don't
##  want preen to run at all, just comment out this setting.
PREEN                           = $(SBIN)/condor_preen
##  How do you want preen to behave?  The "-m" means you want email
##  about files preen finds that it thinks it should remove.  The "-r"
##  means you want preen to actually remove these files.  If you don't
##  want either of those things to happen, just remove the appropriate
##  one from this setting.
PREEN_ARGS                      = -m -r
##  How often should the master start up condor_preen? (once a day)
#PREEN_INTERVAL                 = 86400
##  If a daemon dies an unnatural death, do you want email about it?
#PUBLISH_OBITUARIES             = True
##  If you're getting obituaries, how many lines of the end of that
##  daemon's log file do you want included in the obituary?
```

```
#OBITUARY_LOG_LENGTH        = 20
##  Should the master run?
#START_MASTER               = True
##  Should the master start up the daemons you want it to?
#START_DAEMONS              = True
##  How often do you want the master to send an update to the central
##  manager?
#MASTER_UPDATE_INTERVAL             = 300
##  How often do you want the master to check the timestamps of the
##  daemons it's running?  If any daemons have been modified, the
##  master restarts them.
#MASTER_CHECK_NEW_EXEC_INTERVAL     = 300
##  Once you notice new binaries, how long should you wait before you
##  try to execute them?
#MASTER_NEW_BINARY_DELAY= 120
##  What's the maximum amount of time you're willing to give the
##  daemons to quickly shutdown before you just kill them outright?
#SHUTDOWN_FAST_TIMEOUT              = 120
######
##  Exponential backoff settings:
######
##  When a daemon keeps crashing, we use "exponential backoff" so we
##  wait longer and longer before restarting it.  This is the base of
##  the exponent used to determine how long to wait before starting
##  the daemon again:
#MASTER_BACKOFF_FACTOR              = 2.0
##  What's the maximum amount of time you want the master to wait
##  between attempts to start a given daemon?  (With 2.0 as the
##  MASTER_BACKOFF_FACTOR, you'd hit 1 hour in 12 restarts...)
#MASTER_BACKOFF_CEILING             = 3600
##  How long should a daemon run without crashing before we consider
##  it "recovered".  Once a daemon has recovered, we reset the number
##  of restarts so the exponential backoff stuff goes back to normal.
#MASTER_RECOVER_FACTOR              = 300
##--------------------------------------------------------------------
##  condor_startd
##--------------------------------------------------------------------
##  Where are the various condor_starter binaries installed?
STARTER_LIST = STARTER, STARTER_PVM, STARTER_STANDARD
STARTER             = $(SBIN)/condor_starter
```

```
STARTER_PVM             = $(SBIN)/condor_starter.pvm
STARTER_STANDARD        = $(SBIN)/condor_starter.std
##  When the startd starts up, it can place it's address (IP and port)
##  into a file.  This way, tools running on the local machine don't
##  need to query the central manager to find the startd.  This
##  feature can be turned off by commenting out this setting.
STARTD_ADDRESS_FILE= $(LOG)/.startd_address
##  When a machine is claimed, how often should we poll the state of
##  the machine to see if we need to evict/suspend the job, etc?
#POLLING_INTERVAL        = 5
##  How often should the startd send updates to the central manager?
#UPDATE_INTERVAL         = 300
##  How long is the startd willing to stay in the "matched" state?
#MATCH_TIMEOUT           = 300
##  How long is the startd willing to stay in the preempting/killing
##  state before it just kills the starter directly?
#KILLING_TIMEOUT         = 30
##  When a machine unclaimed, when should it run benchmarks?
##  LastBenchmark is initialized to 0, so this expression says as soon
##  as we're unclaimed, run the benchmarks.  Thereafter, if we're
##  unclaimed and it's been at least 4 hours since we ran the last
##  benchmarks, run them again.  The startd keeps a weighted average
##  of the benchmark results to provide more accurate values.
##  Note, if you don't want any benchmarks run at all, either comment
##  RunBenchmarks out, or set it to "False".
BenchmarkTimer = (CurrentTime - LastBenchmark)
RunBenchmarks : (LastBenchmark == 0 ) || ($(BenchmarkTimer) >= (4 * $(HOUR)))
#RunBenchmarks : False
##  Normally, when the startd is computing the idle time of all the
##  users of the machine (both local and remote), it checks the utmp
##  file to find all the currently active ttys, and only checks access
##  time of the devices associated with active logins.  Unfortunately,
##  on some systems, utmp is unreliable, and the startd might miss
##  keyboard activity by doing this.  So, if your utmp is unreliable,
##  set this setting to True and the startd will check the access time
##  on all tty and pty devices.
#STARTD_HAS_BAD_UTMP = False
##  This entry allows the startd to monitor console (keyboard and
##  mouse) activity by checking the access times on special files in
##  /dev.  Activity on these files shows up as "ConsoleIdle" time in
```

```
##  the startd's ClassAd.  Just give a comma-separated list of the
##  names of devices you want considered the console, without the
##  "/dev/" portion of the pathname.
#CONSOLE_DEVICES      = mouse, console
##  The STARTD_EXPRS entry allows you to have the startd advertise
##  arbitrary expressions from the config file in its ClassAd.  Give
##  the comma-separated list of entries from the config file you want
##  in the startd ClassAd.
##  Note: because of the different syntax of the config file and
##  ClassAds, you might have to do a little extra work to get a given
##  entry into the ClassAd.  In particular, ClassAds require "'s
##  around your strings.  Numeric values can go in directly, as can
##  boolean expressions.  For example, if you wanted the startd to
##  advertise its list of console devices, when it's configured to run
##  benchmarks, and how often it sends updates to the central manager,
##  you'd have to define the following helper macro:
#MY_CONSOLE_DEVICES = "$(CONSOLE_DEVICES)"
##  Note: this must come before you define STARTD_EXPRS because macros
##  must be defined before you use them in other macros or
##  expressions.
##  Then, you'd set the STARTD_EXPRS setting to this:
#STARTD_EXPRS = MY_CONSOLE_DEVICES, ….
##  When the startd is claimed by a remote user, it can also advertise
##  arbitrary attributes from the ClassAd of the job its working on.
##  Just list the attribute names you want advertised.
##  Note: since this is already a ClassAd, you don't have to do
##  anything funny with strings, etc.  This feature can be turned off
##  by commenting out this setting (there is no default).
STARTD_JOB_EXPRS = ImageSize, ExecutableSize, JobUniverse, NiceUser
##  If you want to "lie" to Condor about how many CPUs your machine
##  has, you can use this setting to override Condor's automatic
##  computation.  If you modify this, you must restart the startd for
##  the change to take effect (a simple condor_reconfig will not do).
##  Please read the section on "condor_startd Configuration File
##  Macros" in the Condor Administrators Manual for a further
##  discussion of this setting.  Its use is not recommended.  This
##  must be an integer ("N" isn't a valid setting, that's just used to
##  represent the default).
#NUM_CPUS = N
##  Normally, Condor will automatically detect the amount of physical
##  memory available on your machine.  Define MEMORY to tell Condor
```

```
##  how much physical memory (in MB) your machine has, overriding the
##  value Condor computes automatically.  For example:
#MEMORY = 128
##  How much memory would you like reserved from Condor?  By default,
##  Condor considers all the physical memory of your machine as
##  available to be used by Condor jobs.  If RESERVED_MEMORY is
##  defined, Condor subtracts it from the amount of memory it
##  advertises as available.
#RESERVED_MEMORY = 0
######
##  SMP startd settings
##
##  By default, Condor will evenly divide the resources in an SMP
##  machine (such as RAM, swap space and disk space) among all the
##  CPUs, and advertise each CPU as its own "virtual machine" with an
##  even share of the system resources.  If you want something other
##  than this, there are a few options available to you.  Please read
##  the section on "Configuring The Startd for SMP Machines" in the
##  Condor Administrator's Manual for full details.  The various
##  settings are only briefly listed and described here.
######
##  The maximum number of different virtual machine types.
#MAX_VIRTUAL_MACHINE_TYPES = 10
##  Use this setting to define your own virtual machine types.  This
##  allows you to divide system resources unevenly among your CPUs.
##  You must use a different setting for each different type you
##  define.  The "" in the name of the macro listed below must be
##  an integer from 1 to MAX_VIRTUAL_MACHINE_TYPES (defined above),
##  and you use this number to refer to your type.  There are many
##  different formats these settings can take, so be sure to refer to
##  the section on "Configuring The Startd for SMP Machines" in the
##  Condor Administrator's Manual for full details.  In particular,
##  read the section titled "Defining Virtual Machine Types" to help
##  understand this setting.  If you modify any of these settings, you
##  must restart the condor_start for the change to take effect.
#VIRTUAL_MACHINE_TYPE_ = 1/4
#VIRTUAL_MACHINE_TYPE_ = cpus=1, ram=25%, swap=1/4, disk=1/4
#  For example:
#VIRTUAL_MACHINE_TYPE_1 = 1/8
#VIRTUAL_MACHINE_TYPE_2 = 1/4
##  If you define your own virtual machine types, you must specify how
##  many virtual machines of each type you wish to advertise.  You do
##  this with the setting below, replacing the "" with the
##  corresponding integer you used to define the type above.  You can
##  change the number of a given type being advertised at run-time,
##  with a simple condor_reconfig.
```

```
#NUM_VIRTUAL_MACHINES_TYPE_ = M
#  For example:
#NUM_VIRTUAL_MACHINES_TYPE_1 = 6
#NUM_VIRTUAL_MACHINES_TYPE_2 = 1
##  The number of evenly-divided virtual machines you want Condor to
##  report to your pool (if less than the total number of CPUs).  This
##  setting is only considered if the "type" settings described above
##  are not in use.  By default, all CPUs are reported.  This setting
##  must be an integer ("N" isn't a valid setting, that's just used to
##  represent the default).
#NUM_VIRTUAL_MACHINES = N
##  How many of the virtual machines the startd is representing should
##  be "connected" to the console (in other words, notice when there's
##  console activity)?  This defaults to all virtual machines (N in a
##  machine with N CPUs).  This must be an integer ("N" isn't a valid
##  setting, that's just used to represent the default).
VIRTUAL_MACHINES_CONNECTED_TO_CONSOLE = 1
##  How many of the virtual machines the startd is representing should
##  be "connected" to the keyboard (for remote tty activity, as well
##  as console activity).  Defaults to 1.
VIRTUAL_MACHINES_CONNECTED_TO_KEYBOARD = 1
##  If there are virtual machines that aren't connected to the
##  keyboard or the console (see the above two settings), the
##  corresponding idle time reported will be the time since the startd
##  was spawned, plus the value of this parameter.  It defaults to 20
##  minutes.  We do this because, if the virtual machine is configured
##  not to care about keyboard activity, we want it to be available to
##  Condor jobs as soon as the startd starts up, instead of having to
##  wait for 15 minutes or more (which is the default time a machine
##  must be idle before Condor will start a job).  If you don't want
##  this boost, just set the value to 0.  If you change your START
##  expression to require more than 15 minutes before a job starts,
##  but you still want jobs to start right away on some of your SMP
##  nodes, just increase this parameter.
#DISCONNECTED_KEYBOARD_IDLE_BOOST = 1200
######
##  Settings for computing optional resource availability statistics:
######
##  If STARTD_COMPUTE_AVAIL_STATS = True, the startd will compute
##  statistics about resource availability to be included in the
##  classad(s) sent to the collector describing the resource(s) the
##  startd manages.  The following attributes will always be included
##  in the resource classad(s) if STARTD_COMPUTE_AVAIL_STATS = True:
##    AvailTime = What proportion of the time (between 0.0 and 1.0)
##      has this resource been in a state other than "Owner"?
##    LastAvailInterval = What was the duration (in seconds) of the
##      last period between "Owner" states?
##  The following attributes will also be included if the resource is
##  not in the "Owner" state:
##    AvailSince = At what time did the resource last leave the
##      "Owner" state?  Measured in the number of seconds since the
##      epoch (00:00:00 UTC, Jan 1, 1970).
```

```
##    AvailTimeEstimate = Based on past history, this is an estimate
##      of how long the current period between "Owner" states will
##      last.
#STARTD_COMPUTE_AVAIL_STATS = False
##  If STARTD_COMPUTE_AVAIL_STATS = True, \
STARTD_AVAIL_CONFIDENCE sets
##  the confidence level of the AvailTimeEstimate.  By default, the
##  estimate is based on the 80th percentile of past values.
#STARTD_AVAIL_CONFIDENCE = 0.8
##  STARTD_MAX_AVAIL_PERIOD_SAMPLES limits the number of samples of
##  past available intervals stored by the startd to limit memory and
##  disk consumption.  Each sample requires 4 bytes of memory and
##  approximately 10 bytes of disk space.
#STARTD_MAX_AVAIL_PERIOD_SAMPLES = 100
##--------------------------------------------------------------------
##  condor_schedd
##--------------------------------------------------------------------
##  Where are the various shadow binaries installed?
SHADOW_LIST = SHADOW, SHADOW_PVM, SHADOW_STANDARD
SHADOW                  = $(SBIN)/condor_shadow
SHADOW_PVM              = $(SBIN)/condor_shadow.pvm
SHADOW_STANDARD         = $(SBIN)/condor_shadow.std
##  When the schedd starts up, it can place it's address (IP and port)
##  into a file.  This way, tools running on the local machine don't
##  need to query the central manager to find the schedd.  This
##  feature can be turned off by commenting out this setting.
SCHEDD_ADDRESS_FILE             = $(LOG)/.schedd_address
##  How often should the schedd send an update to the central manager?
#SCHEDD_INTERVAL      = 300
##  How long should the schedd wait between spawning each shadow?
#JOB_START_DELAY       = 2
##  How often should the schedd send a keep alive message to any
##  startds it has claimed?  (5 minutes)
#ALIVE_INTERVAL                  = 300
##  This setting controls the maximum number of times that a
##  condor_shadow processes can have a fatal error (exception) before
##  the condor_schedd will simply relinquish the match associated with
##  the dying shadow.
#MAX_SHADOW_EXCEPTIONS   = 5
##  Estimated virtual memory size of each condor_shadow process.
```

```
##  Specified in kilobytes.
SHADOW_SIZE_ESTIMATE          = 1800
##  The condor_schedd can renice the condor_shadow processes on your
##  submit machines.  How how "nice" do you want the shadows? (1-19).
##  The higher the number, the lower priority the shadows have.
##  This feature can be disabled entirely by commenting it out.
SHADOW_RENICE_INCREMENT = 10
##  By default, when the schedd fails to start an idle job, it will
##  not try to start any other idle jobs in the same cluster during
##  that negotiation cycle.  This makes negotiation much more
##  efficient for large job clusters.  However, in some cases other
##  jobs in the cluster can be started even though an earlier job
##  can't.  For example, the jobs' requirements may differ, because of
##  different disk space, memory, or operating system requirements.
##  Or, machines may be willing to run only some jobs in the cluster,
##  because their requirements reference the jobs' virtual memory size
##  or other attribute.  Setting NEGOTIATE_ALL_JOBS_IN_CLUSTER to True
##  will force the schedd to try to start all idle jobs in each
##  negotiation cycle.  This will make negotiation cycles last longer,
##  but it will ensure that all jobs that can be started will be
##  started.
#NEGOTIATE_ALL_JOBS_IN_CLUSTER = False
######
## Queue management settings:
######
##  How often should the schedd truncate it's job queue transaction
##  log?  (Specified in seconds, once a day is the default.)
#QUEUE_CLEAN_INTERVAL         = 86400
##  How often should the schedd commit "wall clock" run time for jobs
##  to the queue, so run time statistics remain accurate when the
##  schedd crashes?  (Specified in seconds, once per hour is the
##  default.  Set to 0 to disable.)
#WALL_CLOCK_CKPT_INTERVAL = 3600
##  Do you want to allow remote machines to be able to submit jobs to
##  this queue as user "nobody"?
#ALLOW_REMOTE_SUBMIT          = False
##  What users do you want to grant super user access to this job
##  queue?  (These users will be able to remove other user's jobs).
##  By default, this only includes root.
QUEUE_SUPER_USERS   = root, condor
```

```
##--------------------------------------------------------------------
##  condor_shadow
##--------------------------------------------------------------------
##  If the shadow is unable to read a checkpoint file from the
##  checkpoint server, it keeps trying only if the job has accumulated
##  more than MAX_DISCARDED_RUN_TIME seconds of CPU usage.  Otherwise,
##  the job is started from scratch.  Defaults to 1 hour.  This
##  setting is only used if USE_CKPT_SERVER (from above) is True.
#MAX_DISCARDED_RUN_TIME = 3600
##  Should periodic checkpoints be compressed?
#COMPRESS_PERIODIC_CKPT = False
##  Should vacate checkpoints be compressed?
#COMPRESS_VACATE_CKPT = False
##  Should we commit the application's dirty memory pages to swap
##  space during a periodic checkpoint?
#PERIODIC_MEMORY_SYNC = False
##  Should we write vacate checkpoints slowly?  If nonzero, this
##  parameter specifies the speed at which vacate checkpoints should
##  be written, in kilobytes per second.
#SLOW_CKPT_SPEED = 0
##--------------------------------------------------------------------
##  condor_shadow.pvm
##--------------------------------------------------------------------
##  Where is the condor pvm daemon installed?
PVMD                    = $(SBIN)/condor_pvmd
##  Where is the condor pvm group server daemon installed?
PVMGS                   = $(SBIN)/condor_pvmgs
##--------------------------------------------------------------------
##  condor_starter
##--------------------------------------------------------------------
##  The condor_starter can renice the processes from remote Condor
##  jobs on your execute machines.  If you want this, uncomment the
##  following entry and set it to how "nice" do you want the user
##  jobs. (1-19)  The larger the number, the lower priority the
##  process gets on your machines.
#JOB_RENICE_INCREMENT    = 10
##  Should the starter do local logging to its own log file, or send
##  debug information back to the condor_shadow where it will end up
##  in the ShadowLog?
#STARTER_LOCAL_LOGGING   = TRUE
```

```
##  If the UID_DOMAIN settings match on both the execute and submit
##  machines, but the UID of the user who submitted the job isn't in
##  the passwd file of the execute machine, the starter will normally
##  exit with an error.  Do you want the starter to just start up the
##  job with the specified UID, even if it's not in the passwd file?
SOFT_UID_DOMAIN       = TRUE
##--------------------------------------------------------------------
##  condor_submit
##--------------------------------------------------------------------
##  If you want condor_submit to automatically append an expression to
##  the Requirements expression or Rank expression of jobs at your
##  site, uncomment these entries.
#APPEND_REQ_VANILLA          = (expression to append to vanilla job requirements)
#APPEND_REQ_STANDARD         = (expression to append to standard job
requirements)
APPEND_RANK_STANDARD        = ((LastCkptServer == TARGET.CkptServer) ||\
(LastCkptServer =?= UNDEFINED))
#APPEND_RANK_VANILLA         = (expression to append to standard job rank)
##  These are default values for the rank expression that are used if
##  none are specified in the submit file.  Comment these out to
##  disable the feature.
DEFAULT_RANK_VANILLA        = ((Memory*1024) > ImageSize)
DEFAULT_RANK_STANDARD       = ((Memory*1024) > ImageSize)
##  If you want condor_submit to automatically append expressions to
##  the job ClassAds it creates, you can uncomment and define the
##  SUBMIT_EXPRS setting.  It works just like the STARTD_EXPRS
##  described above with respect to ClassAd vs. config file syntax,
##  strings, etc.  One common use would be to have the full hostname
##  of the machine where a job was submitted placed in the job
##  ClassAd.  You would do this by uncommenting the following lines:
#MACHINE = "$(FULL_HOSTNAME)"
#SUBMIT_EXPRS = MACHINE
## Condor keeps a buffer of recently-used data for each file an
## application opens.  This macro specifies the default maximum number
## of bytes to be buffered for each open file at the executing
## machine.
#DEFAULT_IO_BUFFER_SIZE = 524288
## Condor will attempt to consolidate small read and write operations
## into large blocks.  This macro specifies the default block size
## Condor will use.
```

```
#DEFAULT_IO_BUFFER_BLOCK_SIZE = 32768
##--------------------------------------------------------------------
##  condor_preen
##--------------------------------------------------------------------
##  Who should condor_preen send email to?
#PREEN_ADMIN            = $(CONDOR_ADMIN)
##  What files should condor_preen leave in the spool directory?
VALID_SPOOL_FILES      = job_queue.log, job_queue.log.tmp, history, \
                  Accountant.log, Accountantnew.log
##  What files should condor_preen remove from the log directory?
INVALID_LOG_FILES      = core
##--------------------------------------------------------------------
##  Java parameters:
##--------------------------------------------------------------------
##  If you would like this machine to be able to run Java jobs,
##  then set JAVA to the path of your JVM binary.  If you are not
##  interested in Java, there is no harm in leaving this entry
##  empty or incorrect.
JAVA    = /usr/local/bin/java
##  Some JVMs need to be told the maximum amount of heap memory
##  to offer to the process.  If your JVM supports this, give
##  the argument here, and Condor will fill in the memory amount.
##  If left blank, your JVM will choose some default value,
##  typically 64 MB.  The default (-Xmx) works with the Sun JVM.
JAVA_MAXHEAP_ARGUMENT   =
## JAVA_CLASSPATH_DEFAULT gives the default set of paths in which
## Java classes are to be found.  Each path is separated by spaces.
## If your JVM needs to be informed of additional directories, add
## them here.  However, do not remove the existing entries, as Condor
## needs them.
JAVA_CLASSPATH_DEFAULT = $(LIB) $(LIB)/scimark2lib.jar .
##  JAVA_CLASSPATH_ARGUMENT describes the command-line parameter
##  used to introduce a new classpath:
JAVA_CLASSPATH_ARGUMENT = -classpath
##  JAVA_CLASSPATH_SEPARATOR describes the character used to mark
##  one path element from another:
JAVA_CLASSPATH_SEPARATOR = :
##  JAVA_BENCHMARK_TIME describes the number of seconds for which
##  to run Java benchmarks.  A longer time yields a more accurate
##  benchmark, but consumes more otherwise useful CPU time.
```

```
##  If this time is zero or undefined, no Java benchmarks will be run.
JAVA_BENCHMARK_TIME = 2
##  If your JVM requires any special arguments not mentioned in
##  the options above, then give them here.
JAVA_EXTRA_ARGUMENTS =
```

A.2 Template del file condor_config.policy

```
#
# D. Bortolotti (aprile 2001)
##
##  Settings control the policy for running, stopping, and
##  periodically checkpointing condor jobs:
##
##  These macros are here to help write legible expressions:
MINUTE              = 60
HOUR                = (60 * $(MINUTE))
StateTimer          = (CurrentTime - EnteredCurrentState)
ActivityTimer       = (CurrentTime - EnteredCurrentActivity)
ActivationTimer     = (CurrentTime - JobStart)
LastCkpt            = (CurrentTime - LastPeriodicCheckpoint)
##  The JobUniverse attribute is just an int.  These macros can be
##  used to specify the universe in a human-readable way:
STANDARD            = 1
PVM                 = 4
VANILLA             = 5
IsStandard          = (JobUniverse == $(STANDARD))
IsPVM               = (JobUniverse == $(PVM))
IsVanilla           = (JobUniverse == $(VANILLA))
#
NonCondorLoadAvg    = (LoadAvg - CondorLoadAvg)
BackgroundLoad      = 0.3
HighLoad            = 0.5
StartIdleTime       = 15 * $(MINUTE)
ContinueIdleTime    =  5 * $(MINUTE)
MaxSuspendTime      = 15 * $(MINUTE)
MaxVacateTime       = 20 * $(MINUTE)
#
KeyboardBusy        = (KeyboardIdle < $(MINUTE))
ConsoleBusy         = (ConsoleIdle  < $(MINUTE))
```

```
CPU_Idle            = ($(NonCondorLoadAvg) <= $(BackgroundLoad))
CPU_Busy            = ($(NonCondorLoadAvg) >= $(HighLoad))
#
BigJob              = (ImageSize >= (50 * 1024))
MediumJob           = (ImageSize >= (15 * 1024) &&  ImageSize < (50 * 1024))
SmallJob            = (ImageSize <  (15 * 1024))
#
JustCPU              = ($(CPU_Busy) && ($(KeyboardBusy) == False))
MachineBusy         = ($(CPU_Busy) || $(KeyboardBusy))
#
WANT_VACATE    : $(ActivationTimer) > 10 * $(MINUTE) ||  $(IsPVM) || $(IsVanilla)
WANT_SUSPEND                    : (ImageSize < (Memory*256) )
WANT_SUSPEND_VANILLA    : True
#
START                : $(CPU_Idle) && KeyboardIdle > $(StartIdleTime)
SUSPEND              : $(MachineBusy)
CONTINUE             : $(CPU_Idle) && KeyboardIdle > $(ContinueIdleTime)
PREEMPT              : ( ($(ActivityTimer) > $(MaxSuspendTime)) && \
                         (Activity == "Suspended") ) || \
                          ( $(MachineBusy) && (WANT_SUSPEND == False) )
KILL                 : $(ActivityTimer) > $(MaxVacateTime)
PERIODIC_CHECKPOINT     : $(LastCkpt) > (6 * $(HOUR))

# Modifica 18/4/2001
UWCS_PREEMPTION_REQUIREMENTS = $(StateTimer) > (1 * $(HOUR)) && \
                    RemoteUserPrio > SubmittorPrio * 1.2
PREEMPTION_REQUIREMENTS = $(UWCS_PREEMPTION_REQUIREMENTS)
```

A.3 Template del file condor_config .local

```
#
# D. Bortolotti (16-Aug-99)
SUBMIT_GROUP_ID    =
SUBMIT_SITE_DOMAIN =
SUBMIT_EXPRS       = SUBMIT_GROUP_ID, SUBMIT_SITE_DOMAIN
RANK : (SUBMIT_SITE_DOMAIN =?= "SUBSTITUTEDOMAINNAME") + \
          (10 * (SUBMIT_GROUP_ID =?= "SUBSTITUTEGROUPID"))
```

A.4 Template del file *site*.infn.it

```
#
# D. Bortolotti
##  When something goes wrong with condor at your site, who chould get
##  the email?
CONDOR_ADMIN           =
##
##  Network domain parameters:
##
##  What machines have administrative rights for your pool?  This
##  defaults to your central manager.  You should set it to the
##  machine(s) where whoever is the condor administrator(s) works
##  (assuming you trust all the users who log into that/those
##  machine(s), since this is machine-wide access you're granting).
HOSTALLOW_ADMINISTRATOR = $(CONDOR_HOST)
##  CKPT-hostname?
CKPT_SERVER_HOST     =
##  If you don't use a fully qualified name in your /etc/hosts file
##  (or NIS, etc.) for either your official hostname or as an alias,
##  Condor wouldn't normally be able to use fully qualified names in
##  places that it'd like to.  You can set this parameter to the
##  domain you'd like appended to your hostname, if changing your host
##  information isn't a good option.  This parameter must be set in
##  the global config file (not the LOCAL_CONFIG_FILE from above).
DEFAULT_DOMAIN_NAME    =
CkptServer           = "$(CKPT_SERVER_HOST)"
DefaultDomain      = "$(DEFAULT_DOMAIN_NAME)"
STARTD_EXPRS = START, CkptServer, DefaultDomain
```

A.5 Template del file condor_config.ALPHA.OSF1

```
#
# D. Bortolotti (16-Aug-99)
##  Full path to a mail delivery program that understands that "-s"
##  means you want to specify a subject:
MAIL                        = /usr/ucb/mail
##  Daemons you want the master to keep running for you:
DAEMON_LIST           = MASTER, STARTD, SCHEDD, KBDD
##  This entry allows the startd to monitor console (keyboard and
##  mouse) activity by checking the access times on special files in
```

```
##  /dev.  Activity on these files shows up as "ConsoleIdle" time in
##  the startd's ClassAd.  Just give a comma-seprated list of the
##  names of devices you want considered the console, without the
##  "/dev/" portion of the pathname.
CONSOLE_DEVICES = mouse, console
```

A.6 Template del file condor.OSF1

```
#!/bin/sh
#
# condor script for SysV-style init boot scripts.
# Daniela Bortolotti, Sezione INFN di Bologna, Agosto 2002
#
MASTER=SUBSTITUTERELEASE_DIR/sbin/condor_master
VACATE=SUBSTITUTERELEASE_DIR/sbin/condor_vacate
PS="/bin/ps -ef"
case $1 in
'start')
        echo -n "Starting condor:"
        echo
        if test -r /var/run/condor
        then
            echo "FOUND condor lock-file: /var/run/condor. Not started."
        else
             $MASTER
            touch /var/run/condor
            echo
        fi
        ;;
'stop')
        echo -n "Stopping condor:"
        echo
        if test -r /var/run/condor
        then
            # start a process vacate
            echo "Vacate jobs, wait a moment ... please. ;)"
            $VACATE
            sleep 15
            echo
            pid=`$PS | grep condor_m | grep -v grep | awk '{print $2}'`
```

```
            kill -QUIT $pid
            rm /var/run/condor
        else
            echo "NOT FOUND condor lock-file: /var/run/condor. Not stopped!"
        fi
        ;;
*)
    echo "Usage: condor {start|stop}"
    ;;
esac
exit
```

A.7 Template del file condor.INTEL.LINUX

```
#
# D. Bortolotti (16-Aug-99)
##  Full path to a mail delivery program that understands that "-s"
##  means you want to specify a subject:
MAIL                = /bin/mail
##  Daemons you want the master to keep running for you:
DAEMON_LIST         = MASTER, STARTD, SCHEDD
##  This entry allows the startd to monitor console (keyboard and
##  mouse) activity by checking the access times on special files in
##  /dev.  Activity on these files shows up as "ConsoleIdle" time in
##  the startd's ClassAd.  Just give a comma-seprated list of the
##  names of devices you want considered the console, without the
##  "/dev/" portion of the pathname.
CONSOLE_DEVICES = mouse, console
```

A.8 Template del file condor.LINUX

```
#!/bin/sh
#
# chkconfig: 345 99 01
# description: condor script for SysV-style init boot scripts.
# Daniela Bortolotti, Sezione INFN di Bologna, Agosto 2002
# Source function library.
. /etc/rc.d/init.d/functions
MASTER=SUBSTITUTERELEASE_DIR/sbin/condor_master
VACATE=SUBSTITUTERELEASE_DIR/sbin/condor_vacate
```

```
PS="/bin/ps -e"
# See how we were called.
case "$1" in
  start)
            echo -n "Starting condor:"
        echo
            if test -r /var/lock/subsys/condor
            then
           echo "FOUND condor lock-file: /var/lock/subsys/condor. Not started."
        else
            $MASTER
           touch /var/lock/subsys/condor
           echo
        fi
             ;;
  stop)
            echo -n "Stopping condor:"
        echo
        if test -r /var/lock/subsys/condor
        then
           # start a process vacate
           echo "Vacate jobs, wait a moment ... please. ;)"
           $VACATE
           sleep 15
           echo
           pid=`$PS | grep condor_m | grep -v grep | awk '{print $1}'`
           kill -QUIT $pid
           rm /var/lock/subsys/condor
        else
           echo "NOT FOUND condor lock-file: /var/lock/subsys/condor. Not stopped!"
        fi
             ;;
  *)
            echo "Usage: $0 {start|stop}"
            exit 1
esac
exit 0
```

A.9 Template del file condor.csh

```
#
set newpath="SUBSTITUTERELEASE_DIR"
echo $PATH | /bin/grep -q ${newpath}
if ( $status != 0 ) then
  set condor_dir="$newpath"
  setenv PATH "${PATH}:${condor_dir}/bin:${condor_dir}/sbin:${condor_dir}/lib"
  unset condor_dir
endif
```

APPENDICE B

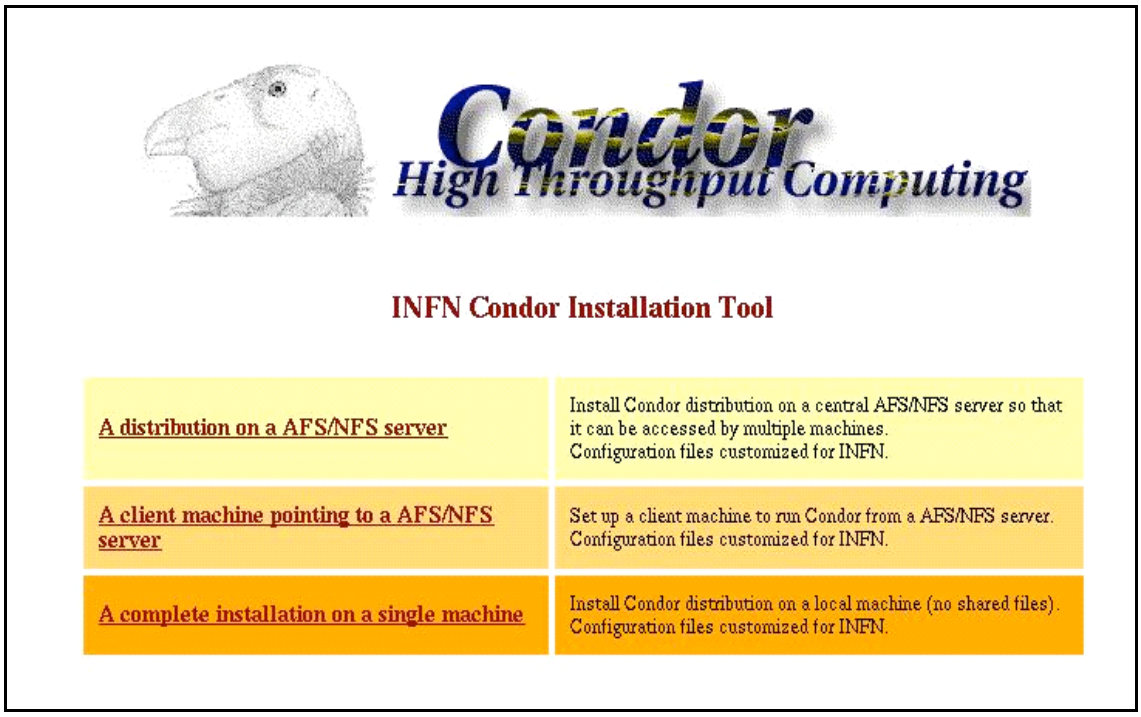

Fig. 1: Introduzione al tool

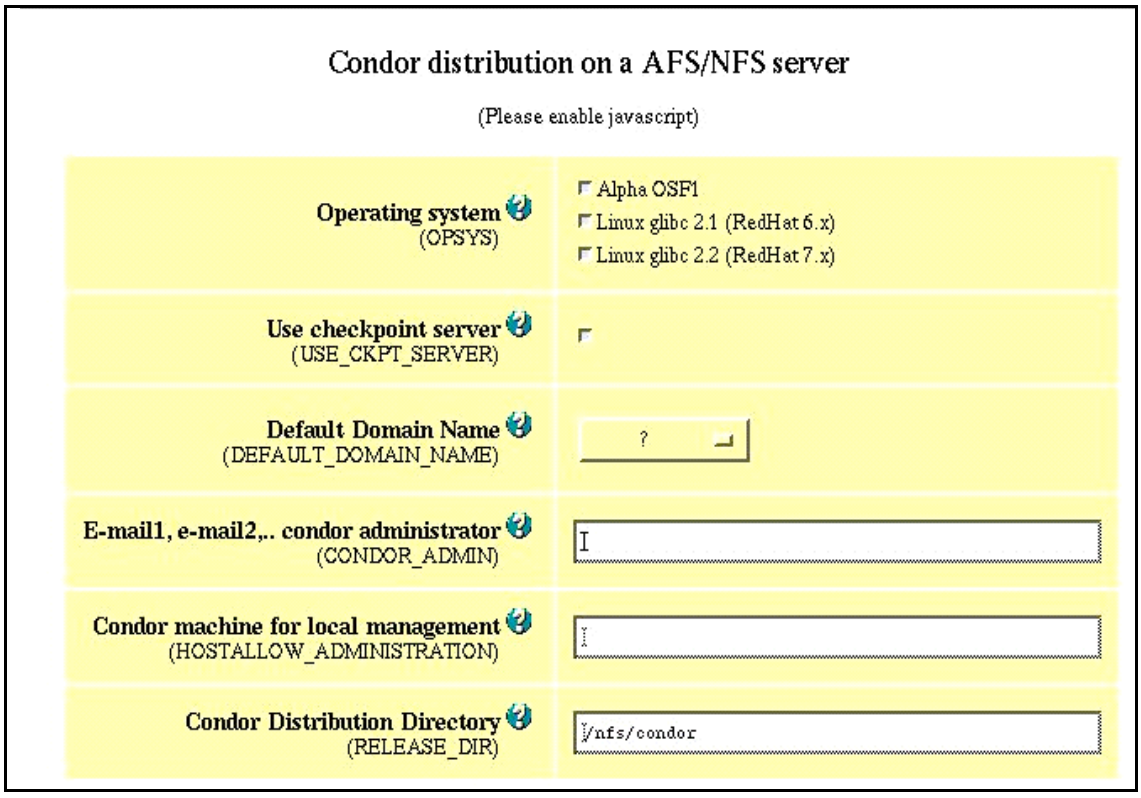

Fig. 2: Configurazione dei parametri per l'installazione server

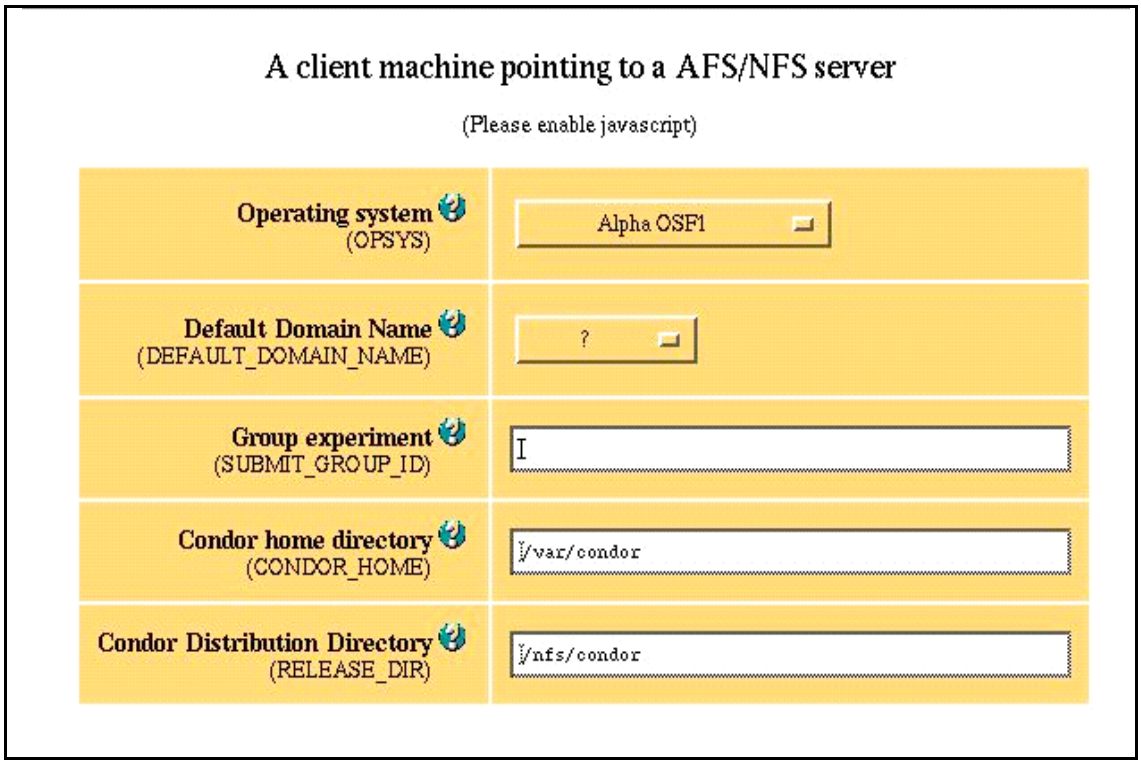

Fig. 3: Configurazione dei parametri per l'installazione client

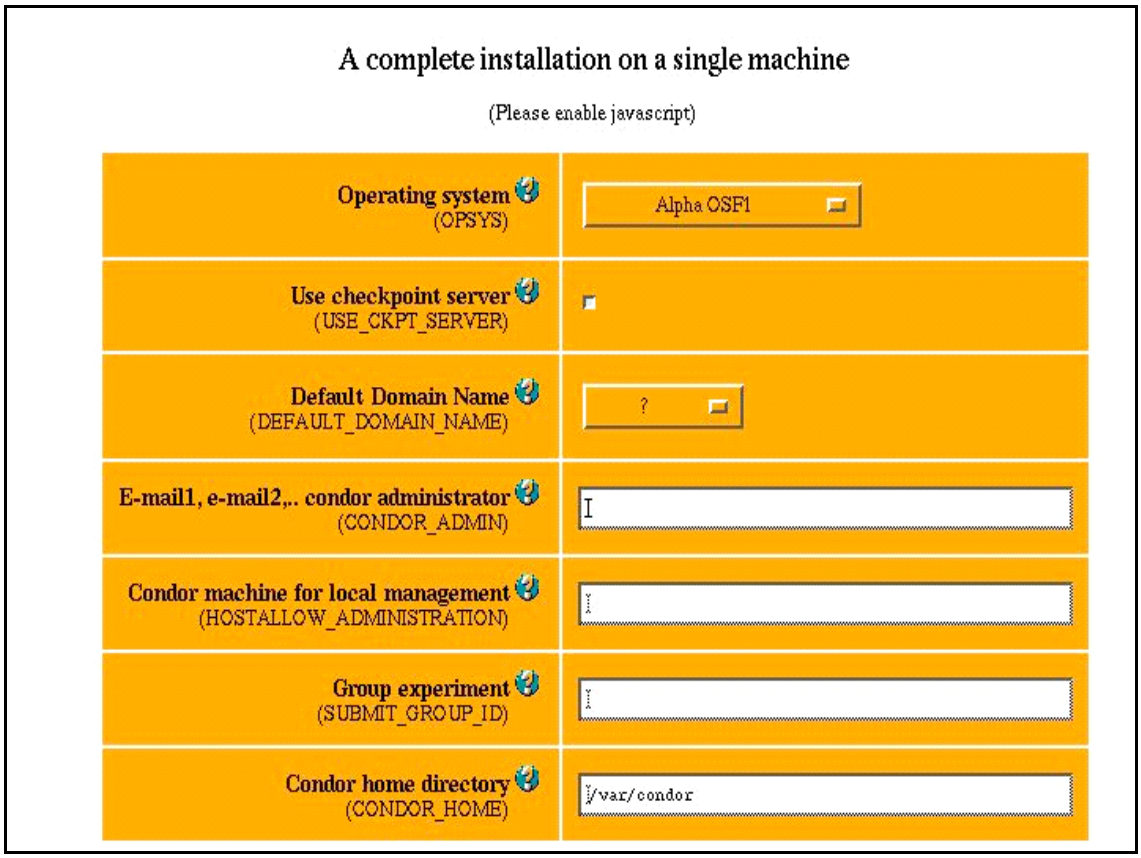

Fig. 4: Configurazione dei parametri per l'installazione completa