ANEXO A

USANDO LA HERRAMIENTA IMIBTOOL

Como mencionamos en el Capitulo IV sección 4.7 OSIMIS nos ofrece una herramienta para trasladar las MIBs definidas en formato SMI al formato GDMO para poder utilizar estas MIBs en OSIMIS a través del IQA, la forma en que trabaja esta herramienta imibtool y su forma de uso es lo que concierne a este anexo.

El proceso de traslación de las MIBs de SNMP a las MIBs de GDMO es ampliamente automatizado y envuelve la conversión del SMI (estructura de la información de gestión) de SNMP dentro de su código GDMO equivalente en el ambiente OSI. La herramienta con que contamos para cumplir esta conversión es llamada imibtool , la cual solo soporta actualmente MIBs de SNMPv1, y no desempeña una conversión al 100%, así que es necesario realizar algún post-procesamiento manual para obtener la MIB GDMO dentro de la forma correcta para que pueda ser usada por el compilador GDMO. El compilador GDMO toma la MIB GDMO post-procesada como entrada y produce un archivo .sif y un archivo oidtable como salida, tal como lo vimos en el Capitulo IV sección 4.7.

El archivo ".sif" producido es especifico para la estructura de la MIB SNMP que va a ser trasladada, si mas de una MIB SNMP va a ser trasladada, entonces para aquellas MIBs que son estructuradas idénticamente se necesita producir solo un archivo ".sif", pero cuando las MIBs SNMP difieren en su estructura, un archivo ".sif" separado es producido para cada MIB SNMP. La herramienta imibtool no nos permite tratar con la conversión de "traps" ya que estos no aparecen en la definición SMI SNMP, por lo tanto los "traps" deben ser tratados separadamente después de la etapa de compilación GDMO y no hacen parte del proceso de conversión.

A.1 PROCESO DE TRASLACIÓN DIRECTA.

Haciendo uso de la herramienta imibtool con que cuenta OSIMIS es posible realizar la traslación de las MIBs SNMP a MIBs GDMO siguiendo los pasos dados a continuación:

1. Establecer las variables de ambiente GDMODIR, OSIMISETCPATH y SMICINCL.
2. Establecer un archivo MIB_include_file el cual especifica el nombre de todas las MIBs SNMP que serán trasladadas al GDMO.
3. Adicionar la información del grupo MIB SNMPv1 al archivo smdump_gdmo.ic.
4. Llamar a imibtool para crear el código GDMO.
5. Desempeñar el post-procesamiento manual.
6. Llamar la compilador GDMO para crear el archivo con extensión ".sif" y el archivo oidtable.
7. Establecer la información de las comunicaciones IQA.
8. Establecer la información de la traslación trap/evento

En las secciones siguientes cada uno de los anteriores pasos será tratado con mas detalle.

Para una mejor ubicación del IQA dentro de OSIMIS, y saber la ubicación correcta de los archivos que utilizaremos durante el proceso de traslación, en la Figura #A1 presentamos la estructura de los directorios OSIMIS donde el software IQA es mantenido.

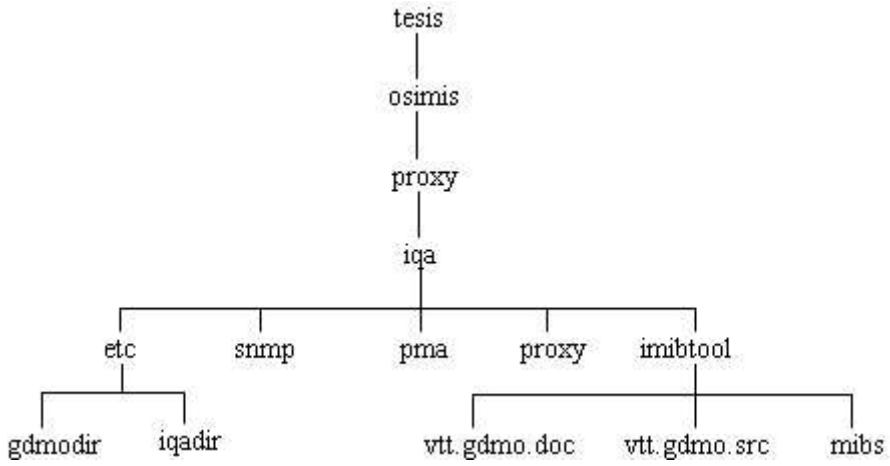

Figura #A1 Estructura del directorio conteniendo IQA.

En la estructura anterior el directorio etc contiene los archivos isoentities, isotailor, osimistailor, mib.init.iqa, oidtable.at y oidtable.gen, y dentro de este el subdirectorio iqadir contiene los archivos iqastartupagents.icm e iqamibconfig.icm y es la misma ubicación donde el archivo .sif se encuentra. El subdirectorio vttt.gdmo.src de imibtool contiene el ejecutable imibtool junto con el archivo smdump_gdmo.ic. También ubicado en el directorio imibtool dentro del subdirectorio mibs existen documentadas un numero de MIBs que han sido pre-limpiadas. El directorio pma contiene el ejecutable iqa y el archivo Pma.h donde la información de los eventos es ingresada. Existen como en todo OSIMIS un numero de Makefiles contenidos en esta estructura de directorios que son los encargados de facilitar la construcción y reconstrucción de los ejecutables, en este caso del IQA.
Es importante aclarar también que varios de estos archivos que hemos mencionado son movidos a otros directorios al realizar la instalación de OSIMIS, como lo son el ejecutable iqa que queda en la ruta "/osimis/bin" y los archivos contenidos bajo el directorio etc que son ubicados en la ruta del etc principal de OSIMIS "/osimis/etc"; como nuestra tarea ahora es adicionar MIBs SNMP al IQA, los archivos aun no han sido movidos a estas rutas ya que el IQA debe ser luego compilado e instalado.

A.1.1 VARIABLES DE AMBIENTE GDMODIR, OSIMISETCPATH y SMICINCL.

Como hasta el momento necesitamos trabajar sobre la estructura de directorios por defecto que trae OSIMIS, las variables de ambiente que necesitamos, las podemos fijar como se muestra en la Figura #A2.

```
[root@osimis root]# export GDMODIR=/tesis/osimis/proxy/iqa/etc/gdmodir
[root@osimis root]# export OSIMISETCPATH=/tesis/osimis/proxy/iqa/etc
[root@osimis root]# export SMICINCL=/tesis/osimis/proxy/iqa/imibtool/mibs
[root@osimis root]#
```

Figura #A2 Establecer Variables de Ambiente.

La variable de ambiente llamada SMICINCL es usada para que imibtool pueda localizar los archivos MIB que va a usar, y debe ser fijada con una lista de los nombres de los directorios en los cuales se van a buscar los archivos MIB, de ser varios y estar en distintas ubicaciones los archivos a utilizar.

A.1.2 El ARCHIVO MIB_include_file.

El próximo paso es construir un archivo llamado MIB_include_file usando un editor de textos, este archivo utiliza las directivas "include" para poder nombrar todas las MIBs a ser usadas por imibtool, las inclusiones deben estar ordenadas de forma tal que todos los items IMPORTS que sean usados en la MIB estén definidos en una MIB incluida previamente.

Veamos como ejemplo el archivo “mibii” mostrado en la Figura #A3 para la MIB-II de SNMP ubicado en “/tesis/osimis/proxy/iqa/imibtool/mibs”.

```
-- file: MIBII - MIB II (with SNMP generic traps)

-- include base mibs
#include "rfc1155.smi"    -- SMI items
#include "rfc1212.smi"    -- OBJECT-TYPE macro
#include "rfc1215.smi"    -- TRAP-TYPE macro
#include "rfc1213.mib"    -- MIB-II
#include "rfc1215.trp"    -- SNMP generic traps
```

Figura #A3 Archivo mibii, un Archivo MIB_include_file.

Una MIB puede ser contenida en un documento tal como un RFC, si este es el caso, el texto contenido antes y después del modulo MIB y la señalización usada para separar las paginas dentro del modulo MIB deben ser limpiados en el documento para que puedan ser usados correctamente por imibtool. El proceso de limpieza puede ser hecho manualmente en un editor de texto y/o con la ayuda de un programa de limpieza de la MIB llamado un “stripper”. El programa de limpieza y todas las MIB estándar son proveídas junto con el paquete imibtool, este programa para limpiar un RFC y revelar un modulo MIB es llamado “mstrip” y se encuentra en la ruta “/tesis/osimis/proxy/iqa/imibtool/vtt.gdmo.src”, la ejecución de “mstrip” es muy simple, ya que este tiene como único argumento el nombre del documento a ser limpiado, el resultado del uso de “mstrip” es colocado sobre la salida estándar, por lo que es conveniente redirigir esta información a un archivo, a continuación la Figura #A4 nos muestra como correr “mstrip”.

```
[root@osimis vtt.gdmo.src]#  mstrip DocumentoRFC  > Nombre-Archivo-de-Salida
[root@osimis vtt.gdmo.src]#
```

Figura # A4 Ejecución del Comando mstrip.

A.1.3 GRUPOS SNMPv1.

Desafortunadamente SNMPv1 no utiliza una palabra clave para identificar los grupos, por consiguiente debemos incorporar una tabla para poder identificar los grupos SNMPv1, el archivo conteniendo esta tabla es llamado smdump_gdmo.ic y esta ubicado en la ruta “/tesis/osimis/proxy/iqa/imibtool/vtt.gdmo.src”.

Como es necesario escribir una tabla para cada MIB que vamos a usar, siguiendo con el ejemplo para la MIB-II debemos escribir una tabla pszSnmpV1mib_2BuiltinGroups, y adicionar a la tabla GRPINFO del archivo smdump_gdmo.ic una entrada adecuada para esta MIB, pero antes de hacer esto, debemos conocer que grupos están en nuestra MIB, y estos grupos pueden ser identificados por

inspección manual de la SMI SNMP (en este caso rfc1213.mib) mirando las entradas que incluyan OBJECT IDENTIFIER.

Las Figuras #A5 y #A6 muestran las tablas pszSnmpV1mib_2BuiltinGroups y GRPINFO adecuadas para la MIB-II.

```
...⇒⇒⇒⇒⇒ Línea #1501
static PSZ pszSnmpV1mib_2BuiltinGroups[] = {
 /* groups in MIB-II (RFC1213) */
   "system",          /* RFC1213 OID ::= { mib-2 1 } */
   "interfaces",      /* RFC1213 OID ::= { mib-2 2 } */
   "at",              /* RFC1213 OID ::= { mib-2 3 } */
   "ip",              /* RFC1213 OID ::= { mib-2 4 } */
   "icmp",            /* RFC1213 OID ::= { mib-2 5 } */
   "tcp",             /* RFC1213 OID ::= { mib-2 6 } */
   "udp",             /* RFC1213 OID ::= { mib-2 7 } */
   "egp",             /* RFC1213 OID ::= { mib-2 8 } */
#if 0
   "cmot",            /* RFC1213 OID ::= { mib-2 9 } */
   "transmission",    /* RFC1213 OID ::= { mib-2 10 } */
#endif
   "snmp",            /* RFC1213 OID ::= { mib-2 11 } */

 /* end of table */
   NULL
}; /* pszSnmpV1mib_2BuiltinGroups[] */
...
```

Figura #A5 Tabla pszSnmpV1mib_2BuiltinGroups.

```
...⇒⇒⇒⇒⇒ Línea #1670
static GRPINFO grpinfoPretranlationTable[] = {
  /* filling in this table is a pretranslation activity */

  { "mib-2",          pszSnmpV1mib_2BuiltinGroups } ,    /* RFC1213 */
  { "clns",           pszSnmpV1clnsBuiltinGroups } ,          /* RFC1238 */
...
```

Figura #A6 Tabla GRPINFO.

Una vez que la información del grupo a sido adicionada al archivo smdump_gdmo.ic, la herramienta imibtool necesita ser recompilada usando el Makefile que existe en el mismo directorio.

A.1.4 LA HERRAMIENTA imibtool.

Imibtool es usada para generar automáticamente una descripción de MIB GDMO OSI desde las descripciones de la MIB SNMPv1, imibtool esta basado sobre las reglas de traslación de SNMP a GDMO del NMF, esta herramienta no desempeña una conversión completa al 100% pero en la practica si llega a una conversión del 80% al 95%, por consiguiente es necesario desempeñar una cantidad de post-procesamiento manual para lograr completar la conversión.
Imibtool esta basado en una herramienta anterior llamada SMIC (“*SNMP MIB compiler”*) [41] el cual fue escrito por David T. Perkins de SynOptics Communication, y la herramienta imibtool que vamos a usar fue producida por Jim Reilly de Nokia Finland [42].

El ejecutable imibtool esta ubicado en la ruta "/tesis/osimis/proxy/iqa/imibtool/vtt.gdmo.src" y en "/osimis/bin/" luego de la instalación de OSIMIS.

OSIMIS provee un manual sobre el uso de SMIC (el cual también cubre a imibtool) llamado smicug.txt, y este esta ubicado en la ruta "/tesis/osimis/proxy/iqa/imibtool/vtt.gdmo.doc". Existen otros documentos localizados en esta misma ruta que hacen referencia a imibtool y SMIC.
Imibtool tiene una multitud de opciones que pueden ser seleccionadas cuando este es ejecutado, y correr imibtool sin argumentos puede causar que una sinopsis del uso del programa y una lista de las opciones disponibles sean desplegadas, estas opciones de la línea de comandos son cubiertas en detalle en smicug.txt.

Imibtool tiene la capacidad de chequear MIBs a varios niveles de rigor, el nivel de chequeo puede ser controlado por medio de la línea de comandos y por medio de las directivas del compilador. La salida de imibtool es especificada por medio de la línea de comandos, y la generación de la información MIB en el formato de salida no es proveída directamente por imibtool y es lograda escribiendo un programa de respaldo adecuado o escribiendo un programa filtro para una de las salidas de imibtool.

Cuando convertimos una nueva MIB SNMP, varias pasadas pueden ser necesarias para corregir todos los errores encontrados en la MIB por imibtool, cuando todos los errores han sido eliminados, ya podemos utilizar la salida de imibtool.

Construir nuestro archivo GDMO usando imibtool desde la MIBII es bastante sencillo y el comando necesario es mostrado en la Figura #A7.

```
[root@osimis mibs]# imibtool -Y1 "RFC1213-MIB" mibii > vtt.mibii.gdmo
[root@osimis mibs]#
```

Figura #A7 Usando imibtool para Trasladar la MIBII.

En imibtool el switch -Y1 es usado para especificar cual MIB SNMP en el archivo MIB_include_file (mibii) debe ser convertida a GDMO, un archivo MIB_include_file puede especificar diferentes MIBs, pero podemos buscar solo convertir alguna de las MIB referenciadas en MIB_include_file, y no todas a través del uso del switch -Y1. Imibtool genera como salidas un archivo llamado "gdmo-asn1.tmp" el cual es simplemente producido como una ayuda para el post-procesamiento y no es usado de forma alguna en el proceso de construcción, y la otra salida que entrega imibtool es el GDMO convertido, y es colocado por defecto en la salida estándar, por tal razón redireccionamos la salida a un archivo que llamamos vtt.mibii.gdmo.

El Makefile del directorio vtt.gdmo.src contiene un símbolo de compilación "DGENERIC_QAF_OUTPUT=1" el cual puede ser usado para reducir el post-procesamiento requerido si la salida de imibtool es trasladada por el compilador GDMO a fin de producir entradas adecuadas para el IQA de OSIMIS.

A.1.5 POST-PROCESAMIENTO MANUAL.

Una vez que imibtool a sido ejecutado, podemos observar el archivo vtt.mibii.gdmo y veremos al inicio de éste el texto de post-procesamiento que es insertado automáticamente encima del gdmo producido, este texto es mostrado en la Figura #A8.

```
...⇒⇒⇒⇒⇒ Línea #1
-- GDMO produced by *modified* SMIC version 1.0.9, July 23, 1992.
```

```
--


--
-- Notes:
--  1. Notifications are handled by internetSystem
--     NOTIFICATION internet alarm.
--  2. Remeber to define ASN.1 modules IimcAssignedOids
--     and iimcCommonDef as in Sect 4.2 of [IMIBTRANS]
--  3. \-\-\* *\-\- comments are hints to reader.
--
-- PostProcessing should be done for:
--  1. "MATCHES FOR" matching rules
--     Only a best guess is genertated.
--  2. Parsable BEHAVIOUR clauses.
--  3. Comments with %ERR<txt>% indicate problems.
--  4. Comments with %PP% are for post processing
--     E.g. where CREATE/DELETE or DELETE<ATT/VAL>
--     statements may be needed in NAME BINDINGS.
--  5. Check the generated ASN.1 support file.
--
-- References:
--  ISO/CCITT and Internet Management Coexistence (IIMC):
--  Translation of Internet MIBs to ISO/CCITT GDMO MIBs
--  (IIMCIMIBTRANS), Internet Draft 4 (OMNIPOINT-1 ?),
--  Lee LaBarre (Editor)
--
--  ISO/CCITT and Internet Management Coexistence (IIMC):
--  Translation of Internet MIB-II (RFC1213)
--  (IIMCMIB-II), Internet Draft 4 (OMNIPOINT-1 ?)
--  Lee LaBarre (Editor)
--

-- ############################################################
--     Converted from SNMP MIB/SMI Module : RFC1155-SMI
--     Converted from SNMP MIB/SMI Module : RFC-1212
--     Converted from SNMP MIB/SMI Module : RFC-1215
--     Converted from SNMP MIB/SMI Module : RFC1213-MIB
--     Converted from SNMP MIB/SMI Module : RFC1215-TRAP
-- ############################################################
...
```

Figura #A8 Texto de Post-procesamiento.

Aclarando un poco mas el texto mostrado en la Figura #A8, los comentarios que aparezcan en vtt.mibii.gdmo comenzando con la expresión %ERR<txt>% son automáticamente producidos en el GDMO para indicar los sitios donde se pueden presentar problemas y es aconsejable revisar manualmente, estos errores pueden ser de tres tipos y son presentados en la Tabla #A1.

Tipo de Error %ERR<txt>%	Descripción
#ERRENM#	La numeración con enteros necesita ser convertida dentro de tipos y módulos ASN.1 separados
#ERRSYNtc#	Las convenciones de tipos necesitan ser chequeadas a mano para seleccionar el nombre apropiado <ASN1Module>.typeName de los módulos ASN.1 involucrados.

#ERR-POSTTRANSLATION#	Indica las secciones del comportamiento que pueden necesitar ser hechas a mano.

Tabla #A1 Tipo de Error %ERR<txt>%.

Además el documento IIMC[A3] define ciertas palabras (CREATE/DELETE) para descripciones GDMO, las cuales especifican como los atributos de las MIBs pueden ser creados o borrados, desafortunadamente no es posible generar automáticamente las palabras dentro de GDMO al realizar la traslación con imibtool, por lo tanto la expresión %PP% es insertada cuando hay necesidad de realizar correcciones al GDMO para nombres enlazados en términos de CREATEs y DELETEs.

Para las expresiones %PP%s, se debe escribir un script sed, que nos va a permitir el post-procesamiento del archivo vtt.mibii.gdmo obtenido. Un ejemplo de un script sed para realizar el post-procesamiento del archivo vtt.mibii.gdmo es mostrado en la Figura #A9.

```
...⇒⇒⇒⇒⇒ Línea #1
# DEPRECATED atEntry Post Processing
#
#s/--%PP% DELETEATT atEntry--/DELETEATT atPhysAddress ;/
#s/--%PP% DELETEVALUE atEntry--/DELETEVALUE "h ;/
#s/--%PP% CREATE atEntry--/CREATE  WITH-AUTOMATIC-INSTANCE-NAMING, WITH-REFERENCE-OBJECT ;/
#s/--%PP% DELETE atEntry--/DELETE  DELETES-CONTAINED-OBJECTS ;/

#
# ipRouteEntry Post Processing
#
s/--%PP% DELETEATT ipRouteEntry--/DELETEATT ipRouteType ;/
s/--%PP% DELETEVALUE ipRouteEntry--/DELETEVALUE 2 ;/
s/--%PP% CREATE ipRouteEntry--/CREATE  WITH-AUTOMATIC-INSTANCE-NAMING, WITH-REFERENCE-OBJECT ;/
s/--%PP% DELETE ipRouteEntry--/DELETE  DELETES-CONTAINED-OBJECTS ;/

#
# ipNetToMediaEntry Post Processing
#
s/--%PP% DELETEATT ipNetToMediaEntry--/DELETEATT ipNetToMediaType ;/
s/--%PP% DELETEVALUE ipNetToMediaEntry--/DELETEVALUE 2 ;/
s/--%PP% CREATE ipNetToMediaEntry--/CREATE  WITH-AUTOMATIC-INSTANCE-NAMING, WITH-REFERENCE-OBJECT ;/
s/--%PP% DELETE ipNetToMediaEntry--/DELETE  DELETES-CONTAINED-OBJECTS ;/

#
# ipNetToMediaEntripForwardEntry Post Processing
#
s/--%PP% DELETEATT ipForwardEntry--/DELETEATT ipForwardType ;/
s/--%PP% DELETEVALUE ipForwardEntry--/DELETEVALUE 2 ;/
s/--%PP% CREATE ipForwardEntry--/CREATE  WITH-AUTOMATIC-INSTANCE-NAMING, WITH-REFERENCE-OBJECT ;/
s/--%PP% DELETE ipForwardEntry--/DELETE  DELETES-CONTAINED-OBJECTS ;/

#
#wipe out any remaining ones
#
```

```
s/--%PP%.*--/--\* \*--/
```

Figura #A9 script vtt.mibii.sed.

Para ayudar a entender este script vtt.mibii.sed aclaremos la siguiente información relacionada a los CREATEs y DELETEs.

- CREATE : Está presente si es permitido crear nuevas instancias de la clase de objeto gestionado referenciado por la clase de objeto subordinado.
    - CREATE WITH-REFERENCE-OBJECT: Si está presente, una referencia al objeto gestionado puede ser especificada en la creación como una fuente de valores por defecto, y puede también especificar la selección de paquetes condicionales.
    - CREATE WITH-AUTOMATIC-INSTANCE-NAMING: Si está presente, la petición CREATE puede omitir especificar el nombre de la instancia del nuevo objeto gestionado.

- DELETE: Está presente si es permitido borrar instancias de la clase de objeto gestionado referenciada por la clase de objeto subordinado
    - DELETE ONLY-IF-NO-CONTAINED-OBJECTs: Si es especificado, cualquier objeto gestionado contenido será explícitamente borrado por operaciones de gestión antes de borrar el objeto gestionado contenido, por ejemplo una petición delete puede causar un error si allí existen objetos gestionados.
    - DELETES-CONTAINED-OBJECTS: Si una petición delete es aplicada a un objeto gestionado para el cual el modificador DELETES-CONTAINED-OBJECT es especificado, la petición delete fallara si cualquier objeto gestionado contenido directa o indirectamente tiene el modificador ONLY-IF-NO-CONTAINED-OBJECTS especificado y también tiene un objeto gestionado contenido, en otro caso, una petición delete exitosa también tiene el efecto de borrar objetos gestionados contenidos. Dado que el modificador DELETES-CONTAINED-OBJECTS permite el borrado de objetos gestionados a pesar de si éste contiene otros objetos gestionados, es conveniente usar el modificador ONLY-IF-NO-CONTAINED-OBJECTS si existe duda de cual modificador es el apropiado.

El comando mostrado en la Figura #A10 puede ser usado para desempeñar el post-procesamiento %PP% con ayuda del script vtt.mibii.sed.

```
[root@osimis mibs]# sed -f vtt.mibii.sed < vtt.mibii.gdmo > vtt.mibii-p.gdmo
[root@osimis mibs]#
```

Figura #A10 Comando para Usar el script sed.

Donde vtt.mibii.gdmo es el archivo gdmo para ser procesado manualmente y vtt.mibii-p.gdmo contiene el resultado de correr el script sobre vtt.mibii.gdmo.

A.1.6 COMPILACIÓN GDMO.

El compilador GDMO necesita ser ejecutado dos veces, primero para producir un archivo ".sif" y segundo para producir el archivo oidtable. Un archivo ".sif" es asociado con una estructura MIB SNMP particular, en otras palabras, se deben producir diferentes archivos ".sif" para diferentes estructuras MIB SNMP.

El comando usado para producir un archivo ".sif" es mostrado en la Figura #A11.

```
[root@osimis iqadir]# gdmo-cmpl -x iimcgdmo.gen vtt.mibii-p.gdmo
[root@osimis iqadir]#
```

Figura # A11 Generación del Archivo ".sif"

Donde iimcgdmo.gen es un archivo de soporte que trae OSIMIS por defecto para ayudar en la construcción del archivo .sif y que se encuentra en la ruta "/tesis/osimis/proxy/iqa/etc/gdmodir".
El uso de este comando produce un archivo que es llamado gdmo.log por defecto y el cual debe ser renombrado y ubicado en la ruta "/tesis/osimis/proxy/iqa/etc/iqadir"., una manera de hacerlo es mostrada en la Figura #A12.

```
[root@osimis iqadir]# mv gdmo.log /tesis/osimis/proxy/iqa/etc/iqadir/rfc1213.sif
[root@osimis iqadir]#
```

Figura #A12 Renombrado del Archivo .sif

Ahora usamos el compilador GDMO por segunda vez para construir un archivo oidtable, y el comando que se puede usar es mostrado en la Figura #A13.

```
[root@osimis iqadir]# gdmo-cmpl -x oidtable.gen vtt.mibii-p.gdmo
[root@osimis iqadir]#
```

Figura #A13 Generación del Archivo oidtable.

Donde de igual manera oidtable.gen es un archivo de soporte que trae OSIMIS por defecto para ayudar en la construcción del archivo oidtable y que se encuentra en la ruta /tesis/osimis/proxy/iqa/etc/gdmodir". De nuevo el archivo producido es llamado gdmo.log por defecto y debemos renombrarlo y ubicarlo en la misma ruta "/tesis/osimis/proxy/iqa/etc/iqadir" como se muestra en la Figura #A14.

```
[root@osimis iqadir]# mv gdmo.log /tesis/osimis/proxy/iqa/etc/iqadir/rfc1213.oidtable
[root@osimis iqadir]#
```

Figura #A14 Renombrado del Archivo .oidtable

El archivo resultante rfc1213.oidtable posee muy bien definidas y en forma clara partes que necesitan ser copiadas a los archivos "/tesis/osimis/proxy/iqa/etc/oidtable.at" y "/tesis/osimis/proxy/iqa/etc/oidtable.gen".

Si el compilador GDMO produce un archivo .sif vacío y lo que nos esta entregando en la salida son una cantidad de mensajes relacionados a errores, entonces podemos fijar la bandera DGENERIC_QAF_OUTPUT=1 dentro del Makefile que se encuentra en el directorio "/tesis/osimis/proxy/iqa/imibtool/vtt.gdmo.src" lo cual nos ayudara dramáticamente a reducir el numero de mensajes de error.

A.1.7 CONFIGURACION DE LAS COMUNICACIONES IQA.

Para realizar esta configuración lo que debemos hacer es editar el archivo "/tesis/osimis/etc/iqadir/iqastartupagents.icm" siguiendo los mismos pasos que ya fueron dados y

explicados en la sección 3.4.6.4 “Ajustando el archivo iqastartupagents.icm” del Capitulo III de esta monografía, por tal razón hacemos aquí la referencia a esta sección y no vemos la necesidad de explicar nuevamente los pasos a seguir.

A.1.8 ACTIVACION DE LA TRASLACION “trap/evento” DEL IQA.

El NMF026[A4] define una notificación genérica “internetAlarm” y una clase de objeto gestionado “internetSystem” sobre el cual todos los traps de Internet pueden ser mapeados. Actualmente imibtool no genera ninguna definición GDMO para la cláusula NOTIFICATIONS (excepto un corte para generar la clase de objeto gestionado “internetSystem”), por lo tanto los traps deben ser tratados fuera del GDMO adicionando para cada trap especifico una entrada a los archivos “/tesis/osimis/proxy/iqa/pma/pma.h” y “/tesis/osimis/proxy/iqa/etc/oidtable.at”, así como también en el archivo “/tesis/osimis/etc/iqadir/iqastartupagents.icm“.

En el archivo pma.h, las entradas son hechas en la tabla de notificación, si para la MIB que estamos adicionando se necesitan efectuar cambios en el archivo Pma.h, entonces el IQA necesita ser reconstruido usando el Makefile existente en el mismo directorio.
Un ejemplo de las entradas que se pueden realizar a la tabla en el archivo Pma.h es mostrado en la Figura #A15.

```
...⇒⇒⇒⇒⇒ Línea #154
NotificationMapping notificationMappingTable[] =
{
#if defined(MONMET)
{"qualityofServiceAlarm",                    "eventLogRecord"},
#endif

{"objectCreation",                    "eventLogRecord"},
{"objectDeletion",                "eventLogRecord"},
{"attributeValueChange",                    "eventLogRecord"},
{"stateChange",                "eventLogRecord"},

{"internetAlarmColdStart",                "eventLogRecord"},
{"internetAlarmWarmStart",                "eventLogRecord"},
{"internetAlarmLinkDown",                "eventLogRecord"},
{"internetAlarmLinkUp",                "eventLogRecord"},
{"internetAlarmAuthenticationFailure",            "eventLogRecord"},
{"internetAlarmEgpNeighbourLoss",        "eventLogRecord"},

#Entradas que se pueden adicionar
{"internetAlarmNode_authentication_failure", "eventLogRecord"},
{"internetAlarmUp/Download_success",        "eventLogRecord"},
{"internetAlarmUp/Download_unknown_files",   "eventLogRecord"},
...
```

Figura #A15 Archivo pma.h.

Cada etiqueta sobre el lado izquierdo por ejemplo “internetAlarmUp/Download_unknown_files” es una etiqueta de nuestra selección la cual es transmitida con el evento desde el IQA, cuando este recibe un trap SNMP.

Similarmente, para cada trap se debe adicionar una entrada en el archivo oidtable.at. Un ejemplo de las entradas en este archivo son mostradas en la Figura #A16.

```
...⇒⇒⇒⇒⇒ Línea #756
internetAlarmNode_authentication_failure:   1.3.6.1.4.1.144.2.0.131089      :InternetAlarmInfo
internetAlarmUp/Download_success:           1.3.6.1.4.1.144.2.0.393233      :InternetAlarmInfo
internetAlarmUp/Download_unknown_files:  1.3.6.1.4.1.144.2.0.393234      :InternetAlarmInfo
...
```

Figura #A16 Archivo oidtable.at.

Para cada entrada, la secuencia de dígitos consiste de < el numero especifico de la compañía>.0.<el numero especifico del trap>. En algunos casos, el numero especifico de la empresa puede ser 1.3.6.1.4.1.144.2. como en el primer ejemplo (internetAlarmNode_authentication_failure) donde 131089 es el numero del trap especifico, esto hace que un enlace sea hecho entre un trap SNMP entrante y un evento saliente.

Finalmente una entrada debe ser hecha al archivo iqastartupagents.icm. donde como habíamos visto en la sección 3.4.6.4 del Capitulo III los parámetros 9 y 10 especifican el puerto que el IQA debe escuchar para traps y la cadena de la comunidad trap, en la Figura #A17 recordamos los valores del archivo iqastartupagents.icm que usamos.

```
...⇒⇒⇒⇒⇒ Línea #1
# IQASTARTUPAGENTS.ICM

"osimis", {snmpUDPDomain:0xc8:15:53:8e:00:a1}, snmpV1, {IIMCrfc1213}, 3, 2000, "public", 0, 3163, "traps"

"tao",      {snmpUDPDomain:0xc8:15:53:b9:00:a1}, snmpV1, {IIMCrfc1213}, 3, 2000, "public", 0, 3163, "traps"

#end
```

Figura #A17 Archivo iqastartupagents.icm.

A.2. CORRIENDO EL IQA.

La Figura #A18 muestra el ejecutable IQA en su modo de ejecución con sus dependencias sobre los archivos .sif, oidtable y archivos soportados.

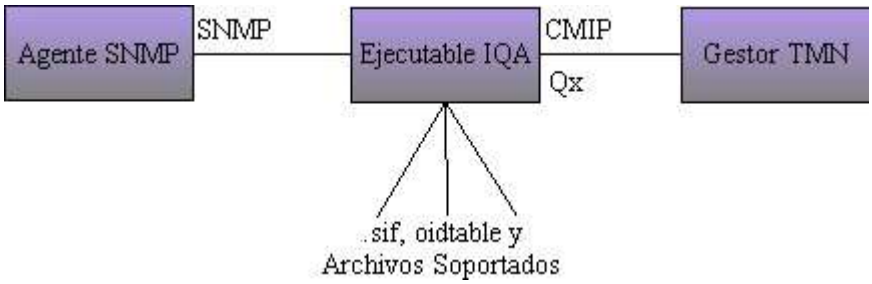

Figura #A18 Ejecutable IQA.

Tipeando iqa , el agente IQA arrancara, recordemos que el IQA necesita ser corrido como root para que este tenga permisos de escuchar todos los puertos y consecuentemente reciba traps SNMP. Una vez el IQA esta corriendo, este crea los OIDs necesarios y conecta al agente SNMP que esta corriendo, quedando así listo para aceptar los comandos CMIP que implementa OSIMIS.