How to compile and link an application using OpenView APIs on OpenVMS

You can compile and link an application program using the OpenView Agent Message and Monitor APIs. While using the APIs to compile and link an application program, the following factors must be considered:

- Programs calling APIs must include a call to the routine **`OVO$VMS_INITIALIZE`** before calling an API routine. The application program must check the return values to verify whether the **`OVO$VMS_INITIALIZE`** is successfully executed. If this check is not done and **`OVO$VMS_INITIALIZE`** fails; the application program may display an error indicating that some of the files in the directory structure cannot be found.

  For example:
  "`Internal error: Cannot open message catalog /opt/OV/lib/nls/C/opcagt.cat`"

- The compile command requires the qualifier **`/NAME=AS_IS`** if the calling program is written in C++. Without this qualifier, C++ compiler converts the opc API calls to uppercase, and as a result they cannot be found at link time. The compile command does not require the qualifier **`/NAMES=AS_IS`** if the calling program is written in C.
- The program must be linked against the shared image **`OVO$LIBOPC_R.EXE`**. You can do this by including the following line in an options file:

  ```
  SYS$SHARE:OVO$LIBOPC_R/SHARE
  ```

- The program must be run from the SYSTEM ACCOUNT.

Note:
Do **not** use **`OVO$VMS_SPI_INIT`** in application program as it is replaced by **`OVO$VMS_INITIALIZE`**.

API Reference

OVO$VMS_INITIALIZE()

```
Int OVO$VMS_INITIALIZE()
```

Parameters

None

Description

This routine does the initialization that is required on the managed node. It defines **SYS$POSIX_ROOT**. It must be called before calling any **opc** API.

Return Values

Return Value	**Description**
SS$_NORMAL	On normal successful completion.
SS$_NOLOGNAM	Openview agents are not installed on the node.
SS$_EXLNMQUOTA	Unable to create SYS$POSIX_ROOT logical due to insufficient quota.
SS$_INSFMEM	Unable to create SYS$POSIX_ROOT logical insufficient dynamic memory.

opcmsg()

```
#include opcapi.h
int opcmsg (
      const int severity, /* in */
      const char * application, /* in */
      const char * object, /* in */
      const char * msg_text, /* in */
      const char * msg_group, /* in */
      const char * nodename, /* in */
      );
```

Parameters

Parameter Name	Description
severity	Severity level of the new message.
application	Application which generates the message.
object	Object of the message source.
msg_text	Message text
msg_group	Message group
nodename	Name of the node which generates the message.

Description

This function is used to send a message created on the managed node to the management server.

Return Values

Return Value	Description
OPC_ERR_OK	On successful generation and sending of messages.
OPC_ERR_APPL_REQUIRED	The attribute OPCDATA_APPLICATION is not set.
OPC_ERR_OBJ_REQUIRED	The attribute OPCDATA_OBJECT is not set.
OPC_ERR_TEXT_REQUIRED	The attribute OPCDATA_MSGTEXT is not set.
OPC_ERR_INVAL_SEVERITY	The severity set is invalid.
OPC_ERR_MISC_NOT_ALLOWED	The message group “misc” is not allowed.
OPC_ERR_NO_MEMORY	Out of memory.

opcmon()

```
#include opcapi.h
int opcmon (
      const char *objname, /* in */
      const double monval /* in */
      );
```

Parameters

Parameter Name	Description
objname	Name of the monitored object.
monval	Actual value of the monitored object.

Description

This function is used to send a monitor value created on the managed node to the management server.

Return Values

Return Value	Description
OPC_ERR_OK	On successful sending of a monitor value.
OPC_ERR_OBJNAME_REQUIRED	objname is NULL.
OPC_ERR_NO_AGENT	The agent is not running.
OPC_ERR_NO_MEMORY	Out of memory.

Sample Program

This is a sample program TEST.C to send a message from managed node to the management server (15.146.239.127)

```
#include<stdio.h>
#include<opcapi.h>

extern Int OVO$VMS_INITIALIZE();

int main()
{
const int severity = 64;
const char * application="app";
const char * object="obj";/* in */
const char * msg_text="MYOPCMSG: Testing with sample program"; /*
in */
const char * msg_group="Test"; /* in */
const char * nodename="15.146.239.127"; /* in */

int status;

status = OVO$VMS_INITIALIZE();
   if ( ! (status & 1) )
      return (status);
status = opcmsg (severity, application, object, msg_text,
msg_group, nodename);

printf("opcmsg is called and the status is %d\n", status);
}
```

Step1: Compile TEST.C

```
CC TEST.C
```

Step 2: Use TEST.OPT for linking

```
$ type  TEST.OPT
$ SYS$LIBRARY:OVO$LIBOPC_R/SHARE
```

Step 3: Link TEST

```
$ link  TEST, TEST.OPT/OPT
```

Step 4: Execute TEST.EXE

```
$ run TEST.EXE
```