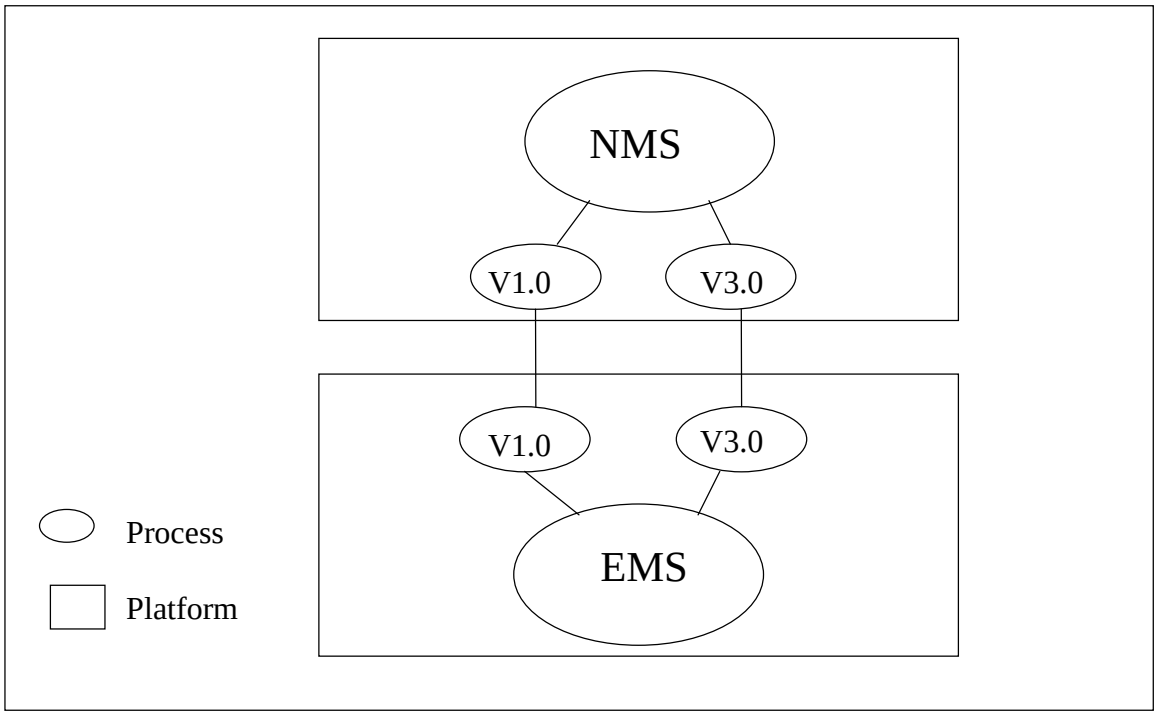
Programmatic Versioning

The programmatic versioning requires a process per IDL version to avoid name clashes. An example deployment is shown below. This can be expanded to many different combinations.



Naming Service

This describes the naming context conventions used for registration and resolution with the Naming Service for EMS entry point objects. Exchanging stringified IOR's via another mechanism is acceptable if agreed to by both implementing vendors. The programmatic solution takes advantage of the following requirement:
EMS Name (distinguishing, invariant identifier established by the EMS, unique across the network management domain using "CompanyName/EMSName" or “CompanyName:EMSName” (where the “/” and the “:” are semantically equivalent[1]); each company ensures that the EMSName is unique relative to the CompanyName and blank spaces, “/” and “:” are not allowed within the name).

The Naming Graph uses the unique EMS name “*CompanyName*/*EMSName*” or “*CompanyName*:*EMSName*” (e.g. VendorA:EMSA).

[1] The “/” character has a reserved usage in the OMG Naming Service. Using the “/’ requires usage of escape characters which that does not work well with many Naming Service implementations. As an alternative the “:” character may be used.

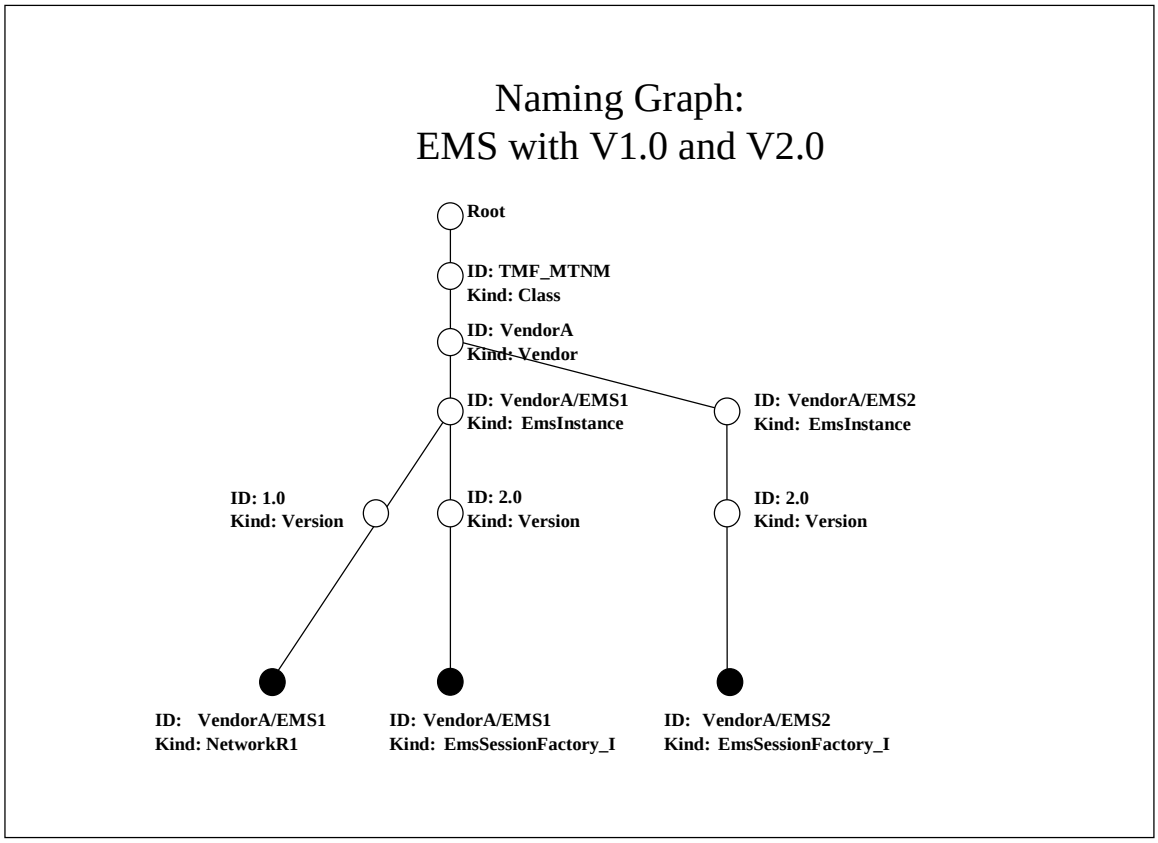
- Naming Containment – kind and id (i.e. a Name-Value pair with Name as KIND and Value as ID)
  - kind = NetworkR1 or EmsSessionFactory_I; Version, EmsInstance, Vendor, Class
  - id = value:
    - For Network_R1, the id is the name of the EMS as specified in emlnmlif::NetworkR1::name.
    - For EmsSessionFactory_I, the id is the name of the EMS as specified in the value field of the only NameAndStringValue_T element of the name field of the EMS_T structure.
    - For Version, the id is the value returned by the getVersion operation. The version may be reported as "n.m" or "n_m"[2] (E.g. "3.0" or the equivalent "3_0").
    - For EmsInstance, the id is the same as the Network_R1 and/or EmsSessionFactory_I id, as applicable.
    - For Vendor, the id is the first part of the EmsInstance id (i.e., the *CompanyName* part, which precedes the "/" or the ":").
    - For Class, the id is always "TMF_MTNM".

- The EMS is only required to register with the Naming Service
- The Naming Service is persistent
- A Naming Service is required only on the NMS
- A Naming Service on the EMS is optional
- A vendor independent solution is recommended

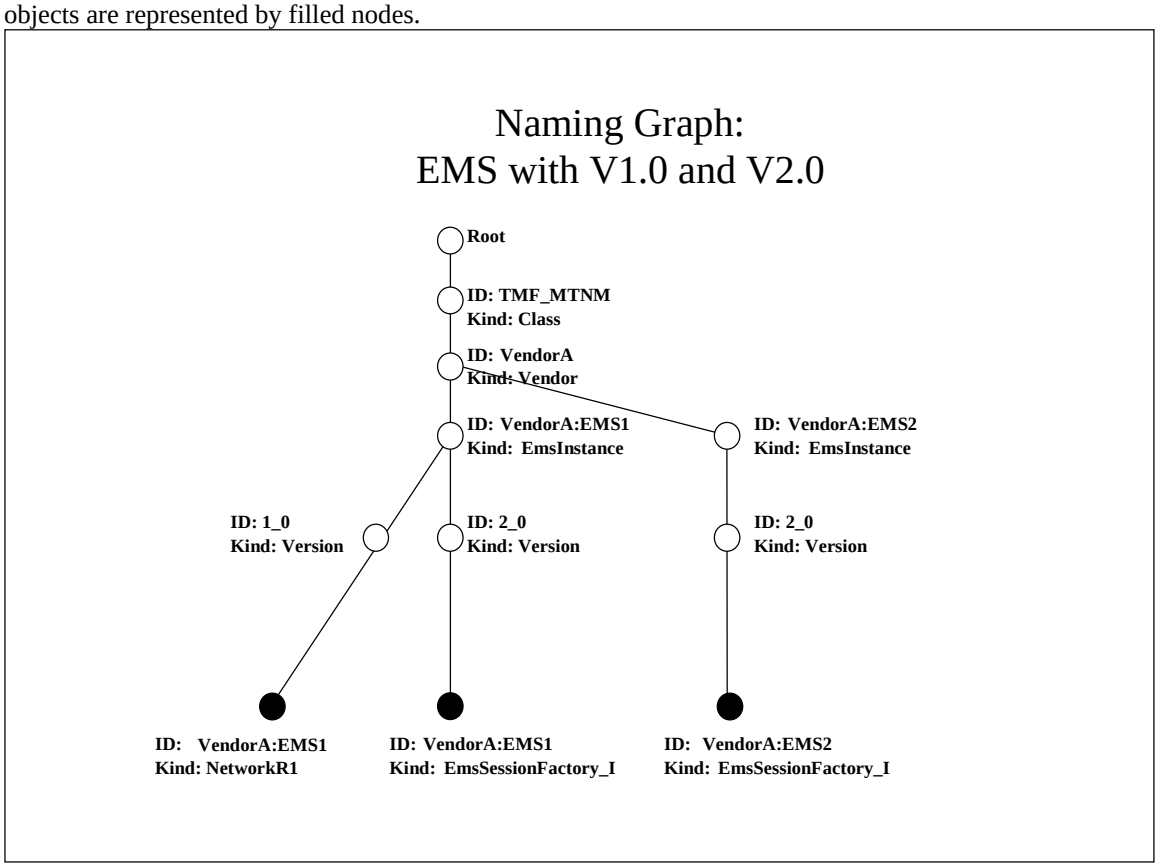
The EMS only registers with the Naming Service for the version that is supported by the EMS.

[2] The "." character has a reserved usage in the OMG Naming Service. Using the '.' requires usage of escape characters which that does not work well with many Naming Service implementations. As an alternative the "_" character may be used.

The diagram below is an example EMS Naming Context using a single Naming Service. The naming contexts are represented by hollow nodes and the EMS entry point objects are represented by filled nodes.
Here is an example of an EMS that supports 2 IDL versions.

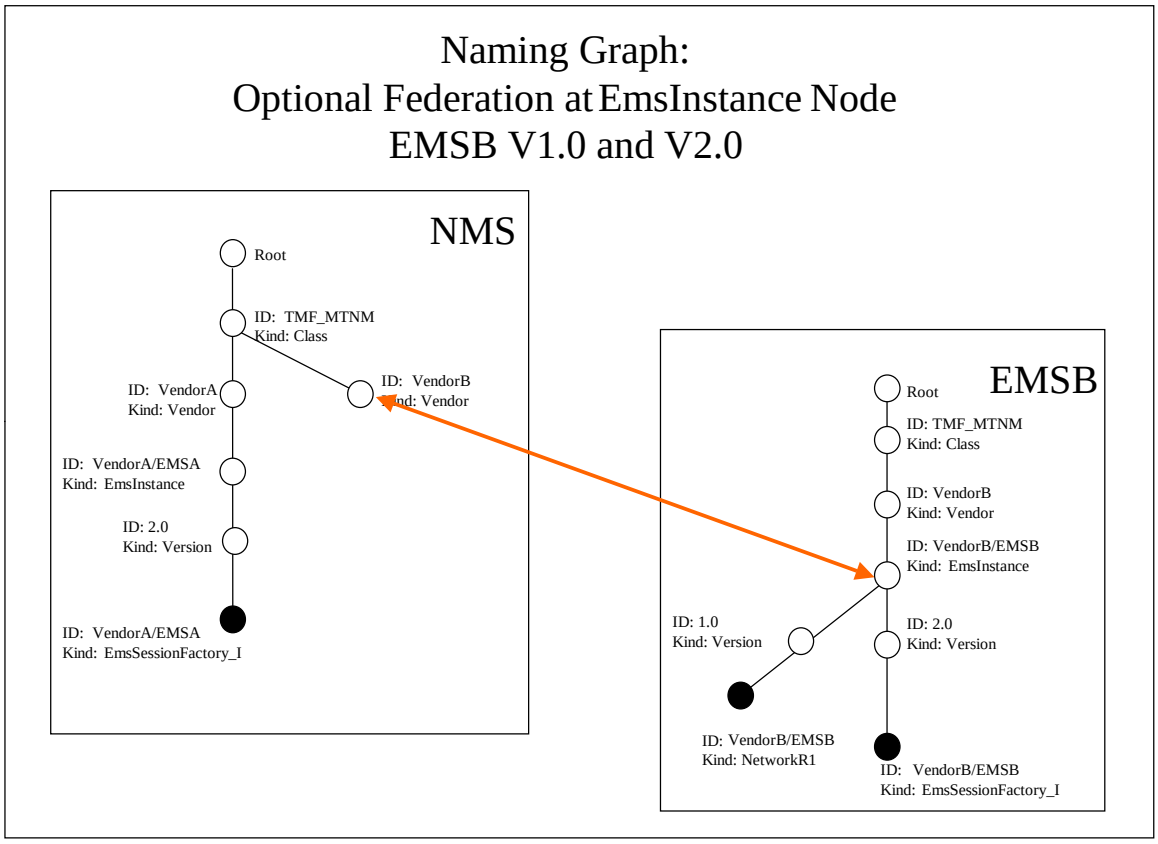


The diagram below shows an example EMS Naming Context using a single Naming Service and the alternative characters ":" rather than "/" and "_" rather than ".". The naming contexts are represented by hollow nodes and the EMS entry point objects are represented by filled nodes.

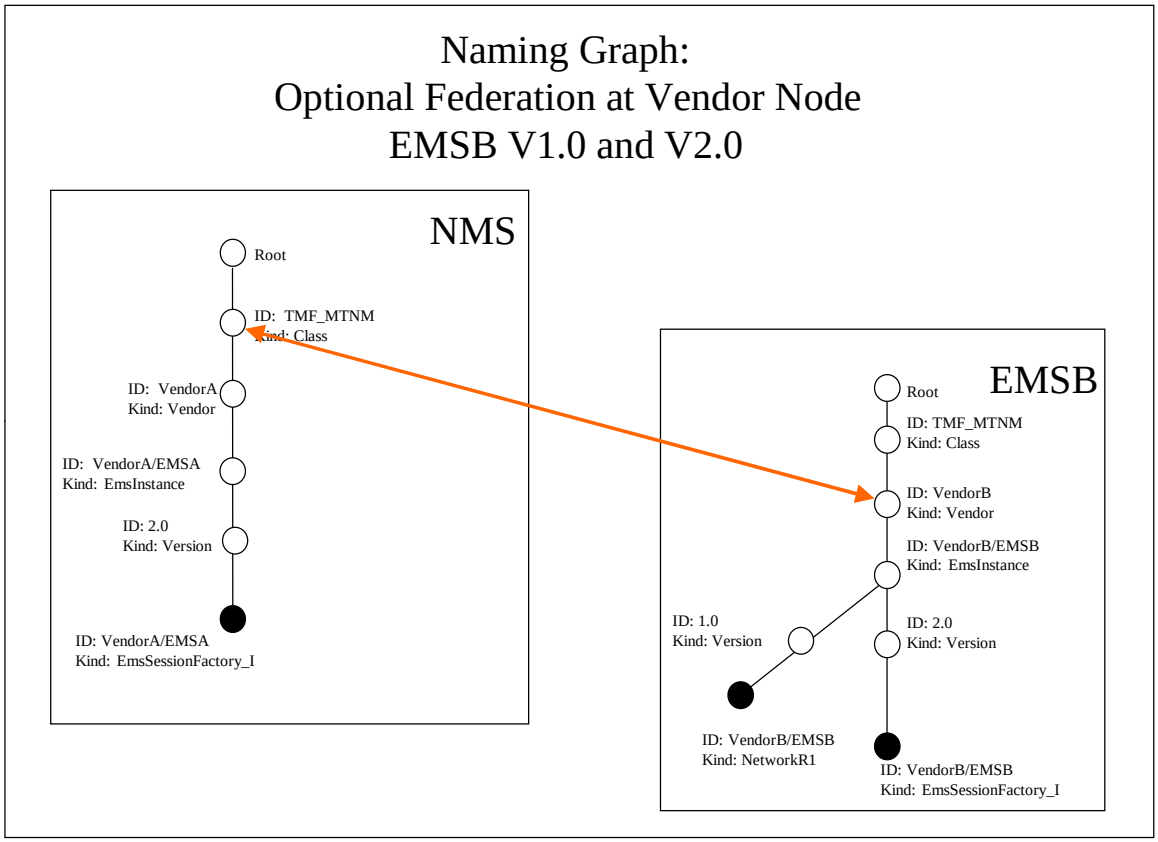


The following diagrams show 2 examples for NMS/EMS Naming Context using a federated Naming Service.

The first example shows federation at the EMS Name or EmsInstance node.



This second example shows federation at the Vendor Name or Vendor node.



Special Case

This section addresses the special case when both implementing vendors agree to exchange IORs via another mechanism rather than using the naming service and recommends, that in this case, that EMS create a persistent IOR (Interoperable Object Reference) for exchanging IORs between NMS and EMS.
This section also shows an implementation example of how this can be accomplished.

A persistent IOR denotes the same CORBA object even if the server shuts down and is restarted (whereas a transient IOR changes each time the IOR is created). So the EMS needs to create a persistent IOR and then stringify it. This string can be passed between EMS and NMS using a file. This ensures that NMS can use EMS generated EMS entry point IOR string over and over again to bind the EMS entry point object reference, even after EMS server has restarted.

Using IIOP, a CORBA server must listen for client connection requests on a fixed TCP/IP port. By default, the port number for each server is assigned by ORB on the server at start-up time and a transient IOR is generated. This IOR changes every time when the server is restarted. To create a persistent IOR, the port number assigned to a server needs to be fixed, for every time the server is started.

Here is a procedure for OrbixWeb based:

* disable port number auto assignment:
```
//Java
import IE.Iona.OrbixWeb._CORBA;
...
_CORBA.Orbix.setConfigItem("IT_IIOP_USE_LOCATOR", "" + false);
//This setting must be applied before any IORs are created in the server.
```

* During registration of a server in the Implementation Repository, a port number can be specified using "putit -port" switch. For example: putit EmlNmlServer -java -port 12014 ....

  You can also choose to specify the port number in your server property file which can be read from Orbix daemon.

* On the server side, you can obtain the stringified IOR by calling org.omg.CORBA.ORB.object_to_string() with required object. For example
  if you want to obtain the IOR for EmsSessionFactory_I, you do:

```
org.omg.CORBA.ORB orb;
String ior =  orb.object_to_string(EmsSessionFactory_I);
```

* Write the IOR string to a file and transfer the file to client machine.

* On the client side, read the IOR string from the file and create a proxy for that object by calling string_to_object() with the IOR string.
  For example:

```
org.omg.CORBA.ORB orb;
Object objRef = orb.string_to_object(ior_string);
```

Revision History

Version	Date	Description of Change
3.0	November 2006	Conversion of versioning into new template

Acknowledgements

<FirstName> <LastName> <Company>

How to comment on the document

Comments and requests for information must be in written form and addressed to the contact identified below:

Keith Dorking Ciena

Phone: +1 678 867 5007

Fax: +1 678 867 5010

e-mail: kdorking@ciena.com

Please be specific, since your comments will be dealt with by the team evaluating numerous inputs and trying to produce a single text. Thus we appreciate significant specific input. We are looking for more input than wordsmith" items, however editing and structural help are greatly appreciated where better clarity is the result.