Protocol and Encoding Versions

On this page:

Version Flexibility

As we saw in the preceding sections, both the Ice protocol and encoding have separate major and minor version numbers. Separate versioning of protocol and encoding has the advantage that neither depends on the other: any version of the Ice protocol can be used with any version of the encoding, so they can evolve independently. (For example, Ice protocol version 1.1 could use encoding version 2.3, and vice versa.)

The Ice versioning mechanism provides the maximum possible amount of interoperability between clients and servers that use different versions of the Ice run time. In particular, older deployed clients can communicate with newer deployed servers and vice versa, provided that the message contents use types that are understandable to both sides.

For an example, assume that a later version of Ice were to introduce a new Slice keyword and data type, such as `complex`, for complex numbers. This would require a new minor version number for the encoding; let us assume that version 1.1 of the encoding is identical to the 1.0 encoding but, in addition, supports the `complex` type. We now have four possible combinations of client and server encoding versions:

Client Version	Server Version	Operation with `complex` Parameter	Operation without `complex` Parameter
1.0	1.0	N/A	Yes
1.1	1.0	N/A	Yes
1.0	1.1	N/A	Yes
1.1	1.1	Yes	Yes

Interoperability for different versions.

As you can see, interoperability is provided to the maximum extent possible. If both client and server are at version 1.1, they can obviously exchange messages and will use encoding version 1.1. For version 1.0 clients and servers, only operations that do not involve `complex` parameters can be invoked (because at least one of client and server do not know about the new `complex` type) and messages are exchanged using encoding version 1.0.

Version Ground Rules

For versioning of the protocol and encoding to be possible, all versions (present and future) of the Ice run time adhere to a few ground rules:

1. Encapsulations always have a six-byte header; the first four bytes are the size of the encapsulation (including the size of the header), followed by two bytes that indicate the major and minor version. How to interpret the remainder of the encapsulation depends on the major and minor version.
2. The first eight bytes of a message header always contain the magic number 'I', 'c', 'e', 'P', followed by four bytes of version information (two bytes for the protocol major and minor number, and two bytes for the encoding major and minor number). How to interpret the remainder of the header and the message body depends on the major and minor version.

These ground rules ensure that all current and future versions of the Ice run time can at least identify the version and size of an encapsulation and a message. This is particularly important for message switches such as IceStorm; by keeping the version and size information in a fixed format, it is possible to forward messages that are, for example, at version 2.0, even though the message switch itself may still be at version 1.0.

See Also

- Basic Data Encoding
- Protocol Messages
- IceStorm
- IceGrid