Explicit Request Contexts

Request contexts provide a means of sending an unlimited number of parameters from client to server without having to mention these parameters in the signature of an operation. For example, consider the following definition:

Slice

```
struct Address {
    // ...
};

interface Person {
    string setAddress(Address a);
    // ...
};
```

Assuming that the client has a proxy to a `Person` object, it could do something along the following lines:

C++

```
PersonPrx p = ...;
Address a = ...;

Ice::Context ctx;
ctx["write policy"] = "immediate";

p->setAddress(a, ctx);
```

In Java, the same code would looks as follows:

Java

```
PersonPrx p = ...;
Address a = ...;

java.util.Map<String, String> ctx = new java.util.HashMap<String, String>();
ctx.put("write policy", "immediate");

p.setAddress(a, ctx);
```

In C#, the code is almost identical:

C#

```
using System.Collections.Generic;

PersonPrx p = ...;
Address a = ...;

Dictionary<string, string> ctx = new Dictionary<string, string>();
ctx["write policy"] = "immediate";

p.setAddress(a, ctx);
```

On the server side, we can extract the policy value set from the `Current` object to influence how the implementation of `setAddress` works. A C++ implementation might look like this:

C++

```
void
PersonI::setAddress(const Address& a, const Ice::Current& c)
{
    Ice::Context::const_iterator i = c.ctx.find("write policy");
    if (i != c.ctx.end() && i->second == "immediate") {

        // Update the address details and write through to the
        // data base immediately...

    } else {

        // Write policy was not set (or had a bad value), use
        // some other database write strategy.
    }
}
```

For this example, the server examines the value of the context with the key `"write policy"` and, if that value is `"immediate"`, writes the update sent by the client straight away; if the write policy is not set or contains a value that is not recognized, the server presumably applies a more lenient write policy (such as caching the update in memory and writing it later).

See Also

- Request Contexts
- The Current Object