
Subject: Writing an IDL application using a C++ data model.

Posted by [Robbie](#) on Mon, 12 Nov 2007 04:14:22 GMT

[View Forum Message](#) <> [Reply to Message](#)

Writing an IDL application using a C++ data model.

I was wondering if anyone has written an IDL application where most of the data and meta data is stored and handled in C++?

Until now, I have taken the philosophy that I should call as little C++ as possible. In fact, all of my C++ is totally experimental is not in production yet.

There are several reasons why I would prefer a C++ data model:

- 1) C++ is strongly typed, so the data model would be well defined.
- 2) C++ is much faster and flexible for organising meta data. IDL structures are slow and tend towards disorganisation.
- 3) I could call external C++ code directly from the data model itself.

Here is an example class with a single property

```
class my_object: public idl_object {
public:
    float height;
    my_object() {
        height = 0;
    }
    float getHeight() {return(height);};
    void setHeight(float h) {height = h;};
};
```

The class could be registered in C++ as being accessible from IDL. I would use C++ templates to assist in resolving type errors at compile time.

```
registerClass<my_object>("MY_OBJECT");
registerProperty<float,
my_object>("HEIGHT",&my_object::getHeight,&my_object::setHeight);
```

The object could be instantiated and accessed from IDL like this:

```
obj = cpp_obj_new("MY_OBJECT")
obj -> GetProperty, HEIGHT=height
help, height
```

I have an existing implementation of casting variables from IDL to boost::multi_array so the interface should support most IDL types.

