Subject: Re: Compiling file with many functions: huge performance difference between IDL and IDLDE
Posted by Sidney Cadot on Thu, 18 Mar 2004 22:48:09 GMT
View Forum Message <> Reply to Message

Ben Tupper wrote:

```
> Sidney Cadot wrote:
>
>> Mirko Vukovic wrote:
>>
>>>> P.S. the reason we're doing this is that we need to implement a
>>>> string-based map with optiomal performance, like this:
>>>>
>>>> FUNCTION f_tom
>>>>   RETURN, 123
>>>> END
>>>>
>>>> FUNCTION f_dick
>>>>   RETURN, 456
>>>> END
>>>>
>>>> FUNCTION f_harry
>>>>   RETURN, 789
>>>> END
>>>>
>>>> FUNCTION f, name
>>>>   CATCH, error_status
>>>>   IF error_status EQ 0 THEN RETURN, -1
>>>>   RETURN, call_function("f_" + name)
>>>> END
>>>
>>>
>>>
>>>
>>> Out of curiosity, would a structure work here:
>>> a={f_tom:123,f_dick:456,f_harry:789...} ?
>>>
>>> It could be created using create_struct.
>>>
>>> Retrieve info using
>>> a=str.f_dick
>>>
>>> Curious minds want to know :-)
>>
>>
>>
>> Your idea is sound, but I am not aware of a way to retrieve the index
```

>> of a tag-name based on its name.
>>
>> You assume that "f_dick" is available at compile time, whereas I need
>> to resolve the string at runtime. Something like this would work:
>>
>> i = TAG_INDEX("f_dick", str)
>> value = str.(i)
>>
>> ... But only if functionality to get a tag index can be retrieved from
>> a struct (anyone knows how to do this?) and if its fast, i.e. if IDL
>> implements it via a hash table or similar.
>>
>
> How about this ? [[code snipped]]

Thanks for the effort, but this sort of defeats the purpose of the whole
exercise, which is to have a fast mapping function. Your solution is
linear search (the TAG_NAMES and WHERE functions), which is too slow for
out application.

Best regards,

Sidney