

Re: C++ in embedded systems

Source: <http://linux.derkeiler.com/Newsgroups/comp.os.linux.embedded/2003-08/0148.html>

From: Ken Lee (postmaster_at_noname.com)

Date: 08/15/03

Date: Thu, 14 Aug 2003 23:19:01 GMT

On Thu, 14 Aug 2003 14:20:04 +0100, Ian Bell <ianbell> wrote:

>Mad@Spammers wrote:

>> Hi.

>>

>> I have followed some discussions about C++ deployment in embedded
>> systems and I learned there are several criticism on C++ itself and
>> its deployment in such systems. Someone mentioned he designs using OOD
>> but implements in C and Assembly, which would be my approach as well
>> (maybe because I started as hardware designer and tend to consider
>> hardware limitations when designing software :-). I work with people
>> that have software engineering formation though and they tend to use
>> every resource of C++ without concerns with performance in the
>> embedded realm, leaving eventual optimizations to the end of the
>> design which we know it's much more difficult and neither practical
>> nore echonomicaly wise. I don't think these folks would go back to C
>> even using OOD techniques so I'd like to get the best possible from
>> C++ advantages and avoid its drawbacks. I would appreciate if you
>> could comment on this and give hints on what to avoid when programming
>> for embedded systems in C++. Suggestions of books and links will be
>> very much appreciated as well.

>>

>

>My personal (and possibly controversial) view is that OOD is a complete
>waste of time in general and absolutely useless for embedded systems.
>there a many reasons why I hold this view but some of them are:

Yes it is controversial. Whether you accept it or not, the OO approach is the current vogue. Graduating Software Engineers are conversant with OOA & OOD. A common language, UML, has been developed so that analysis & design can be better expressed. There are a plethora of OO tools available now.

Also why is software developed for embedded system any different in quality requirements from that of other fields? It isn't. The same process that is adopted in the OO methodology is applicable for embedded development. The name of the game is to produce "quality" software. OO in itself doesn't guarantee this, but it's the associated

process & approach (in using modelling) that empowers the Software Engineer with the ability to achieve this goal.

>1. *Objects bear no relationship to actual objects*

Don't quite understand this comment.

>2. *Information hiding and the other OO so called 'attributes' are
>counterproductive to embedded development where the intimate connection
>between the hardware and the software must never be diluted.*

I disagree. Sure for embedded systems there is an association between the hardware & software, but like any software system the hardware can be mapped to drivers. I can imagine that one can write software for an embedded system where hardware access is spaghetti-ed throughout the code. If you do, then one is not writing their software in a fashion that would make it easily testable. Sure one has In-Circuit Emulators & alike, but I regard them as belated options in testing.

Where I work, we write the software such that it can be readily ported to other platforms which may employ different micros and/or different hardware and/or different operating systems. To remotely achieve this, one has to delineate the hardware from the software application. Also I'm not talking about monster applications but embedded systems based on micros like Hitachi's SH-1, Tiny H8 & Motorola's 6805.

In a current project with the Tiny H8, we are achieving about 98% structural (branches & conditional) unit test coverage of the Software Application, with a test harness that runs in a console window on a PC and about 95% structural unit test coverage of drivers that run on the target system. That's the current status, but our goal is to achieve 100% coverage — which we expect to do. The integrated system does fit & run on a Tiny H8, but unfortunately, there isn't enough ROM space to accomodate a full test harness on the target system & so it has to be broken up.

Ken.

>Ian
>

+=====+

I hate junk email. Please direct any
genuine email to: kenlee at hotpop.com