In fact, the data acquisition software developed by C language. In order to protect the shared data and configuration between multi threads mutex lock was used in the software.

If I have to develop data acquisition software using Java, I believe that thread synchronization of method or object in Java can resolve also this problem because thread synchronization can keep consistency of shared data and board configurations shared by several threads or processes.

Hi Changseung,

This was an interesting read. Great post. Since you are working in a nuclear plant, I am assuming that your software development project would be considered mission critical. Do you think that implementing thread synchronization using Java would be just as good as C? Would safety considerations demand that older, more proven technology be given precedence over newer technology?

Regards,

Emanuel

Hi James,

Thanks for sharing your work with us. I think the char data type is just weird. You do not need to do any type casting as long as you declare all the relevant variables as char. In the for loop of the ArrayWriter class, you could have:

  for ( char i = startValue;  i <  startValue + 3; i++)

  SharedArray.Add(i);

So, if you passed the character 'a' as the startValue, the next character that will be generated when startValue is incremented by 1 is 'b'. This is made possible by the fact that the next logical value in the unicode character set after 'a' is 'b', followed by 'c' and so on. Behind the hood, th increment is actually done on the numerial equivalents of these characters and not on the characters themselves. what do you think?

Best regards,

Ifeanyi.

Hi Ifeanyi,

I'll have to try this myself later, but what you suggest sounds more streamlined than my solution if it works, as I can rely on the numerical incrementation taking place 'under the hood'. I thought for a minute that the explicit casting might help explain to newcomers what's happening, but now I think its just obfuscating the code.

Jim

I actually did exactly what Ifeanyi mentioned above in my solution, and it worked. Though I must admit that I did not want to initially type cast because I wanted to first try out the solution that required the least amount of work. ☺

Dear Emmanuel

Thank you for interesting in my post.

I have several mission critical software development experiences in avionic field not nuclear field. Anyhow I am so familiar with software development standards (DO-178C, IEC61508) in these two fields.

In order to development mission critical software, all APIs are used for the development shall be verified. It means that if you are developing a mission critical application using Java, I think you must have test result and source code of JVM, JDK and OS system.

I think this is hard to get or purchase these kinds of stuff.

I saw lots of applications were developed in C, assembly language but I never saw and heard that mission critical system developed in Java.

The importance thing in these fields is safety, not a using new technology.

Nevertheless there are some challenges using Java for mission critical system.

I don’t know well real time Java, but if you interest in please read the article from below link.

<http://rtcmagazine.com/articles/view/100068>

Best regards

Changseung Kim

Hi Changseung,

I think this confirms what we have learned in 2 of our previous modules, which is that tight-coupling/low cohesion is more appropriate for mission / safety critical software. While Java programs (at least the well-written ones), tend to have the characteristics of low coupling/high-cohesion.

The article you cited is dated 2004, so perhaps Java has made more inroads into mission-critical applications, but for situations such as yours, I probably would just stick with the proven technology

Regards,

Emanuel

Hi Emanuel,

I looked at your solution after both you and Ifeanyi mentioned that it was unnecessary to cast between int and char values before incrementation. The run method in your ArrayWriter class does look easier to read than mine and is very concise.

The only other minor observations I have is that '%c' could have been used (instead of '%s') on line 36 of the SimpleArray class, and the toString method could have used the Arrays.toString method rather than the loop (Deitel and Deitel made this change for the 9th edition of their book).

Jim

Hi Jim,

Thanks for looking at my work. The thought of using %c didn’t occur to me. Even though using %s instead would probably not cause a significant problem in overhead, given how powerful hardware is nowadays, I still think try to use only the computing resources I need. Thanks for the tip.

Emanuel

Hello Emmanuel and Changseung,

Please allow me to join your interesting discussion. Already back in 2003, Sharp et al. have investigated Java’s chances in hard real-time environments. One of the interesting characteristics in the light of this week’s discussion is indeed thread-synchronization: To comply with the requirements for mission-critical applications, the latency for entering a synchronized method vs. entering its unsynchronized counterpart needs to be less than 5 microseconds. The resulst presented by Sharp et al. (2003) revealed that Java “only” requires roughly a 2 microsecond overhead added to the average speed of entering an unsynchronized method of 1.3 micro seconds. So Java can definitely compete here.

Since then obviously a lot “happened” in the industry and several real-time VM based on Java Standard edition have emerged such as Aonix Perc (atego 2013a) that is indeed used for mission-critical applications such as the Aegis Weapon System (atego 2013b).  
Although various “modifications” such as deterministic garbage collection and real-time threading makes Perc different from the everyday JVM, the “beauty” of Java especially in terms of productivity and platform independency is still present (atego 2013b).

I am certainly not an expert in “mission-critical” application design but from my perspective, Java is not the problem, the problem is based on the modular concept that Java introduces and that such an approach exposes a high work load in terms of testing to make sure that all components play together in a way that ensures mission critical operation.

What do you think?

Best regards,  
Daniel

References:  
atego (2013a) ‘Aonix Perc’. Available from <http://www.atego.com/products/aonix-perc/> [Accessed August 20, 2013]  
atego (2013a) ‘Lockheed Martin Selects Aonix PERC Virtual Machine for Aegis Weapon System’. Available from http://www.atego.com/pressreleases/pressitem/lockheed-martin-selects-aonix-perc-virtual-machine-for-aegis-weapon-system [Accessed August 20, 2013]  
Sharp et al. (2003) ‘Evaluating Real-Time Java for Mission-Critical Large-Scale Embedded Systems’.

Hi Daniel,

Thank you for joining the discussion It appears that Java has made leaps and bounds, especially if it is meeting flight safety standards. . I guess that it was only a matter of time with all the hardware advances, and it seems to have outdistanced the software. Though in my opinion, I think almost any language can be used for real-time systems, as long as its environment is compliant with the RT standards.

Regards,

Emanuel

Hi Changseung,

Nice work. The only thing that stuck out to me was what Jim pointed out in my code. And that was you could use %c instead of %s, since you’re only writing one character.

Regards,

Emanuel

Dear NuwnChamaraP

private final char array[];

Would it be possible to declare a generic array which can receive any object type? We started with

Private final int array[];

How do we create a memory structure that will be generic and capable of holding any given object type? Currently we have to keep modifying the array type. How do we manage cases where we cannot tell the type of data we will receive prior to runtime and need to cater for any given type?

Bibliography

Deitel, P. &, Deitel, H (2012) ‘Java How to Program’. Boston, USA: Prentice Hall.

Best Regards

RC

Would it be possible to declare a generic array which can receive any object type?

Hi Remigius,

Interesting that you brought this up. Since Java is a strongly typed language, I’m not sure this is possible. I think a possible solution is to create a new class with the different types:

Class NewClass {

int num1;

String myString;

float myFloat;

}

NewClass[] newClassArray = new NewClass[5];

Regards,

Emanuel

Hi John,

Interesting read. The traffic light serves as a good example, as well as a good analogy. Though I’m not sure that no 2 threads can never run concurrently. For example, I think that while a red light thread was running, I think that the green light thread for the other direction could also be running. Unless of course that one thread could be a green light thread, which controls green lights for all directions of the intersection.

Regards,

Emanuel