Like to chat?

July 14th, 2008

This post marks the end of one era, and the beginning of another. What is ending is our newsletter *Connections*. After some navel gazing, we decided that the newsletter format was too inflexible as an effective communication tool. Besides creating the content for the newsletter, quite a bit of work goes into the typesetting for each issue that, apart from making for pretty PDF pages, does not really contribute anything. Moreover, because we publish *Connections* every two months, it does not provide us with a way to deal with "little" topics that are too small for an FAQ or an article. And, occasionally, we would like to discuss a news item or a topic that has popped up on someone else's blog but, by the time we publish the next issue of the newsletter, the topic is yesterday's news…

What we really need is something that gives us more flexibility and allows us to throw in short and sharp snippets about middleware without having to go to the trouble of publishing a newsletter, and blogs are an ideal way to do that. So, here we go—you are reading the first of our blog entries, which marks the start of the new era.

In this new era, are you going to have to make do without all the awesome information we used to publish in *Connections*? (Big pat on our collective shoulders here…) The answer is "Of course not!"

- We will continue to publish FAQs. (We have just added two new ones, about distributed garbage collection and MQ Series.)
- We will continue to publish articles that discuss various topics in depth. (See our article page for our latest literary masterpiece.)
- We will continue to publish opinion pieces in the style of the *Connections* editorials.

So, everything you have come to love (yes—I know I'm in a presumptuous mood today) will still be there, just in a different format. In addition, you'll get more topical and off-the-cuff contributions from us that, up to now, we did not really have an outlet for. And, of course, blogs allow you to comment and discuss things whereas communication with PDF documents is very much like Ice oneway invocations. I hope that you will take advantage of the comment feature and praise us for our latest brilliant ideas. (For example, you'll be at liberty to tell me straight away that, today, I'm being *too* presumptuous and put me back in my place, instead of having to send me an email that I will dutifully ignore.) In other words, we can have a good old chat about all things Ice, middleware, ZeroC's plans for world domination (just kidding…), and whatever else happens to pop up.

Which brings me to another way of chatting… If you look back through past issues of *Connections*, you will notice that most articles have concentrated on a specific feature of Ice, or explored a single design or implementation technique. All these designs and techniques are useful, apply in the real-world, and a typical Ice application will likely use several of these designs and techniques. And that is exactly the point: a typical Ice application will use several of these designs and

techniques, not just a single one. As always in software development, different designs do not stand in isolation and, when combined, can interact with each other and present design challenges that, otherwise, would not arise. Ice is no different, so we thought it would be useful to present a more complex Ice application that combines several features and services.

This show-case application is a chat application that allows people to communicate with each other in real time over the internet, much like MSN Messenger or Yahoo! Messenger. We have no ambitions to supplant these existing services, which do a fine job. Rather, we chose a chat application because it presents design and implementation challenges that are common to many distributed applications. For example, it must be possible to authenticate clients that want to use the application, eavesdroppers must not be able to monitor other people's conversations, the application must co-exist with existing client- and server-side firewalls, and it should be robust in the face of misbehaved or malicious clients. In other words, the idea is to show how to design and implement an industrial-strength Ice application that meets real-world requirements and is more than just a toy demonstration.

Obviously, you would expect Ice to be suitable for this (otherwise we would not dare to present the application in the first place). But you may not expect how remarkably easy it is to develop such an application. Despite the realistic requirements, code complexity rises (almost) linearly with the number of requirements and not as some higher-order function. This is because Ice provides features and services that are well designed, address orthogonal concerns, and, as a result, are easy to combine and adapt to specific application requirements. This is no accident; our design philosophy is that features do not make it into Ice because we think someone may find them useful. Instead, features make it into Ice when we encounter a real shortcoming that cannot reasonably be overcome without new functionality in the platform. This means that *everything* in the platform exists because it *really* was needed and that its design and implementation were driven by real (instead of imaginary) requirements. In the few cases where requirements interact with each other, we show you how to deal with them without suffering a blowout in complexity and development cost.

So, check out our new chat demo pages, where you can read about how the application works and get an overview of its design and implementation. If you want more meat, you can read our in-depth article that gives you the complete run-down, and, for even more meat, you can download the source code for the application. Of course, you can also point your browser at our chat server and chat via HTTPS, or use one of the clients to chat via the Ice protocol straight from your machine into our server (securely, of course).

If you do like (or don't like) what you see, you can let us know: by posting in our forum, by adding comments to this blog entry, or in real time, by chatting in our chat room.

At ZeroC, there are many ways to chat!

Cheers,

Michi.