Java and C# interoperability

Asked 16 years, 4 months ago Modified 7 years, 10 months ago

Viewed 24k times

33

I have two programs. One is in C# and another one in Java. Those programs will, most probably, always run on the same machine.

What would be the best way to let them talk to each other?

So, to clarify the problem:

This is a personal project (so professional/costly libraries are a no go). The message volume is low, there will be about 1 to 2 messages per second. The messages are small, a few primitive types should do the trick. I would like to keep the complexity low. The java application is deployed as a single jar as a plugin for another application. So the less external libraries I have to merge, the better. I have total control over the C# application. As said earlier, both application have to run on the same computer. Right now, my solution would be to use sockets with some sort of csv-like format.

c# java interop

Share

edited Feb 21, 2017 at 9:04

Improve this question

Follow

Kirill Kobelev

10.5k ● 6 ● 32 ● 52

asked Aug 19, 2008 at 18:31

pbreault

There is a related discussion about CLR/JVM interoperability. – Anderson Green May 1, 2016 at 17:25

8 Answers

Sorted by: Highest score (default)

9

Kyle has the right approach in asking about the interaction. There is no "correct" answer without knowing what the usage patterns are likely to be.

Any architectural decision -- especially at this level -- is a trade-off.

You must ask yourself:

- What kind of messages need to be passed between the systems?
- What types of data need to be shared?
- Is there an important requirement to support complex model objects or will primitives + arrays do?
- what is the volume of the data?
- How frequently will the interactions occur?
- What is the acceptable communication latency?

Until you have an understanding of the answers, or potential answers, to those questions, it will be difficult to choose an implementation architecture. Once we know which factors are important, it will be far easier to choose the more suitable implementation candidates that reflect the requirements of the running system.

Share Improve this answer

Follow

answered Aug 19, 2008 at 20:16

Cheekysoft

35.6k ● 20 ● 74 ● 86

Although Cheekysoft is absolutely right, my guess is you are best off with XML over sockets and object-xml binding in Java (JAXB) and C# (XML Schema Definition Tool). See: stackoverflow.com/questions/765422/jaxb-equivalent-in-c – Carsten Nov 16, 2009 at 5:30

I've heard good things about IKVM, the JVM that's made with .NET.

7

Share Improve this answer

Follow

answered Aug 19, 2008 at 18:41

Mark Cidade

99.8k ● 33 ● 229 ● 237

> we recently used IKVM to communicate between a Java server and C# UI. The communication protocol was RMI which IKVM handled very nicely. – Matt Sep 22, 2008 at 21:00

4

Ice from ZeroC is a really high performance "enterprisey" interop layer that supports Java and .net amongst others. I think of it as an updated Corba - it even has its own object oriented interface definition language called Slice (like Corba's IDL, but actually quite readable).

The feature set is extensive, with far more on offer than web services, but clearly it isn't an open standard, so not a decision to make lightly. The generated code it spits out is somewhat ugly too...

Share Improve this answer

Follow

answered Aug 19, 2008 at 19:32

serg10

32.6k ● 16 ● 75 ● 94

4

I realize you're talking about programs on the same machine, but I've always liked the idea of passing messages in XML over HTTP.

Your server could be a web server that's ready to accept an XML payload. Your client can send HTTP messages with XML in the body, and receive an HTTP response with XML in it.

One reason I like this is that HTTP is such a widely used protocol that it's easy to accept or create HTTP POST or GET requests in any language (in the event that you decide to change either the client or server language in the future). HTTP and XML have been around for a while, so I think they're here to stay.

Another reason I like it is that your server could be used by other clients, too, as long as they know HTTP and XML.

Share Improve this answer Follow

edited Nov 24, 2009 at 0:45

answered Aug 20, 2008 at 2:19

Josh Brown

53k ● 11 ● 56 ● 81

1 I would be really interested in hearing about throughput in this approach. My initial thought on reading this was, that this would be feasible only for small payloads simply due to the overhead of assembling and disassembling the XML and

putting it in a http message. – Thorbjørn Ravn Andersen Nov 16, 2009 at 3:30

3

I used JNBridge (http://www.jnbridge.com/jnbpro.htm) on a relatively simple project where we had a .NET client app using a relatively significant jar file full of business object logic that we didn't want to port. It worked quite nicely, but I wouldn't say we fully exercised the capabilities of JNBridge.

Share Improve this answer

Follow

answered Aug 19, 2008 at 18:35

Chris Farmer

25.4k • 36 • 123 • 168

1

I am a big fan of Thrift an interoperability stack from Facebook. You said they code will probably run on the same machine so it could be overkill but you can still use it.

Share Improve this answer

Follow

answered Aug 19, 2008 at 18:34

John Downey

14.1k • 5 • 38 • 33

0

If they are separate programs and running as independent applications,you may use sockets. I know it's bit complex to define communication protocol but it'll be quite straight-forward.

However if you have just two separate programs but want to run them as single application, then I guess IKVM is a better approach as suggested by marxidad.

Share Improve this answer

Follow

answered Aug 26, 2008 at 8:01

jatanp

4,092 ● 4 ● 42 ● 46

0

It appears a very similar question has been asked before here on stack overflow (I was searching Google for java windows shared memory):

[Efficient data transfer from Java to C++ on windows]

From the answer I would suggest you to investigate:

> "Your fastest solution will be memory mapping a shared segment of memory, and them implementing a ring-buffer or other message passing mechanism. In C++ this is straight forward, and in Java you have the FileChannel.map method which makes it possible."

Share Improve this answer

Follow

edited May 23, 2017 at 10:29

Community Bot

1 ● 1

answered Nov 16, 2009 at 3:34

Thorbjørn Ravn Andersen