

Protocol Compiler - Feature #11474

Allow fields to be individually marshalled

01/22/2016 10:04 AM - Richard Neswold

Status: Closed	Start date: 01/21/2016
Priority: Normal	Due date:
Assignee:	% Done: 0%
Category: Parser	Estimated time: 0.00 hour
Target version:	
Description	
<h3>The "Problem"</h3>	
<p>DPM routes data from front-ends to clients and, as the data is picked apart and sent, it is possibly scaled. This works well and, since the new DPMs aren't heavily loaded (yet), we haven't seen performance problems. There is a concern, however, that we won't scale up as much as we would like. Charlie King and Rich Neswold, during a discussion about DPM, realized that data is routed as follows:</p>	
<ol style="list-style-type: none">1. An Erlang process receives data from a front-end as a binary.2. It sends pieces of the binary to processes communicating with ACNET clients.3. If the client asks for raw data, the binary can be directly placed in a protocol message and marshaled. If the data needs to be scaled, the process takes the contents of the binary, scales it, puts the floating point value in the protocol message and the marshaller turns it into a binary.	
<p>Scaled data for a single scalar isn't too bad, but scaling an array of data is pretty wasteful because the process loops through the binary and generates a list of doubles which the marshaller turns back into a binary. It would be nice to avoid the intermediate list from being generated.</p>	
<h3>A Solution</h3>	
<p>The protocol source file could include an annotation that says we want to marshal a field ourselves. The generator would then create a function that marshals all fields of the message except the marked one. The function would take a parameter that would provide the marshaled field.</p>	
<p>However:</p>	
<ul style="list-style-type: none">• Do we only allow one field per message to be annotated this way?• This needs to be done in a type-safe way (i.e. we can't let the programmer provide an arbitrary binary as the marshaled value; it needs to be properly formed.)	
Related issues:	
Related to Protocol Compiler - Feature #11463: Add constraints to the grammar	New 01/21/2016

History

#1 - 01/22/2016 10:06 AM - Richard Neswold

- Related to Feature #11463: Add constraints to the grammar added

#2 - 11/05/2019 03:23 PM - Richard Neswold

- Status changed from New to Closed

- Category set to Parser

This needs to be done in a type-safe way (i.e. we can't let the programmer provide an arbitrary binary as the marshaled value; it needs to be properly formed.)

The fact that the data needs to be validated removes any massive speed-up we would get copying a field from one message to another (because the marshaling function can't know the data came from a validated, received message.)

Plus, annotating a field in the protocol to support an optimization in a service implementation doesn't feel right.