Subject: Re: Is it possible to use Core/Rpc in non blocking mode?
Posted by steffen on Wed, 24 Apr 2013 08:56:16 GMT
View Forum Message <> Reply to Message

Hi nlneilson,

Thank you for you input. I appreciate the feedback.

It is not possible with a server in both ends because all GPRS connections are automaticly put behind NAT routers.
The server cannot connect to the clients unless I implement some peer to peer technology, and since it is a standard webserver that is not an option.

I will go on making my own JsonRpc implementation, where the transport layer is independent.
Then it will be much easier to use it across different socket types and web protocols.
It will also be easier to implement batch requests.

Here comes a bit of clarification on what I'm facing, just for information if you are interested:

JsonRpc is a very simple format which we have used for several years now, but I have first recently started to move our clients from Delphi running on WinXP Embedded to U++ on Linux.
Having completed the user interface design for our touch screens and all the serial communication with the trucks in place, it is now time to bring them all online again.
Now I have the chance to get rid of the few caveats I know about in our current design.

We use JsonRPC for all communications, and normal HTTP for file transfers, like updates, bug reports and so on.

99% of my clients are placed in trucks and they sometimes move into zones without GSM/GPRS coverage. Causing all kinds of weird connection problems. And our error and reconnect handling is a very big part of the current implementation. Sometimes we even have to kill the power to the modems to bring them back online.

We currently have three threads for communications, Client->Server, Server->Client and a Server->Client download thread for updates.
The last one is only active when needed, but it takes 7-10 minutes to download an update, so it gets its own thread and runs in the background.
Bug reports never exceeds 100kB so they are just send through our normal outgoing queue.
The queue holds both jsonrpc requests as well as direct http requests. It is even possible to overwrite items in the queue, so stuff like fuel level and current GPS coordinates only reside once in the queue. There is no need to send outdated data to the server
When a connection dropout occurs it influences all connections and we sometimes have to abort all connections, repowering the modem, redial and restart all the threads.
Very complicated and the reason I would like to try a different approach.
The dial part is actually done using the Windows RAS api, and occasionally it simply refuses to dial until we reboot the entire system. This is one of the motivations for moving everything to U++ on Linux.
Another is the wish to move to an ARM based platform we currently have at the first prototype

stage.

Regards,
Steffen