Industrial Control

XML Interface

Tech Note – TN3005
December 1, 2020

1891 N. Gaffey St. Ste. E
San Pedro, CA 90731

p. 310.241.2973

support@ctekproducts.com
www.ctekproducts.com

Table of Contents

Introduction

Ctek's XML interface provides read/write access to internal parameters/settings found within the Ctek SkyRouter product line. It also supports the remote execution of programs resident on the SkyRouter. The interface is defined as a hierarchy of resource identifiers within which each level qualifies access to successively lower levels. The XML interface is useful to remotely access SkyRouter resources, or as a building block for applications and higher level protocols.

Data Model

As mentioned in the introduction the XML interface supports multiple levels. Since the SkyRouter is capable of simultaneously hosting multiple applications the Level 1 defines access to the base router and any installed application designed to utilize the XML interface. Levels beyond Level 1 further define and segment data and resources associated with entities defined at Level 1. See table 1 for detailed data model definition.

The XML interface is designed to support recursion for any Level 1 entity that requires this capability. For instance, when accessing the Level 1 router entity, reading at a specific level returns everything below that level. A read of router will return everything that is known about the router while a read of wireless will return all parameters defined under wireless. To read one specific wireless parameter the request should be for Router/Wireless/Specific Wireless Parameter.

Level 1	*Level 2*	*Level 3*	*Level 4*	*Notes*
router				
	model			Read only - See section 7 for definition
	hardware			Read only - See section 7 for definition
	firmware			Read only - See section 7 for definition
	firewall			0 = Disable; 1 = enable Example: <firewall> 1</ firewall >
	portforward			
		f1 – f12		Ports to forward
			addr	LAN IP address to forward to
			state	State of port – 0 = disabled; 1 = enabled
			srcport	Source (WAN side) port number to forward from
			dstport	Destination (LAN side) port number to forward to
			protocol	0 = TCP; 1 = UDP; 2 = both

defaults			Restore factory defaults - unvalued
reboot			Restart/Reboot - unvalued
wireless			
	rssi		Read only - See section 7 for definition
	networkid		Read only - See section 7 for definition
	servicetype		Read only - See section 7 for definition
	serviceclass		Read only - See section 7 for definition
	band		Read only - See section 7 for definition
	roam		Read only - See section 7 for definition
	servicestatus		Read only - See section 7 for definition
	connectstatus		Read only - See section 7 for definition
	ipaddress		Read only - See section 7 for definition
	deviceid		Read only - See section 7 for definition
	phonenumber		Read only - See section 7 for definition
	temperature		Read only - See section 7 for definition
	currenttemplog		Read only; See specifics in the Get Response section below
	lasttemplog		Read only; See specifics in the Get Response section below
	wwanstate		Wireless WAN setting R+W 0 = disabled; 1 = enabled
	netstability		CDMA Only - Read Only – 0 = Not ready for activation; 1 Ready for activation
	apn		HSPA Only APN – HSPA R+W
		name	APN Name
		authtype	0 = None; 1 = PAP; 2 = Chap; 3 = Both
		username	APN User name
		password	APN Password
	apn2		HSPA Only APN – HSPA R+W
		name	APN2 Name

			authtype	0 = None; 1 = PAP; 2 = Chap; 3 = Both
			username	APN2 User name
			password	APN2 Password
		apn3		HSPA Only APN – HSPA R+W
			name	APN3 Name
			authtype	0 = None; 1 = PAP; 2 = Chap; 3 = Both
			username	APN3 User name
			password	APN3 Password
		autoapn		
			mode	APN selection – 0 = APN, 1 = APN2, 2 = APN3, 3 = AUTO Auto attempts to select valid APN from a possible list of APN, APN2, APN3
		activate		CDMA Only Activation R+W put 0 = deactivate; 1 = activate (deactivate impacts internal status only) get – 0 = Not Activated; 1 = Activated; 2 = In Progress; 3 = Saving data; 4 = Failed
	relayout			
		[null]		Relay Output *get* format - 0 = off, 1 = on
		0		Relay Output *put* format - 0 = off
		1		Relay Output *put* format - 1 = on
plc				
	Output*			
		o1 - o9999 [pin number]		Digital <o12> x</o12> Output *put* format (x = 0 = off; x = 1 = on) Numeric <o12> nnn</o12> Output *put* format (nnn = whole or decimal number)
	Output*			
		o1 - o9999 [pin number]		<037> </o37> Output *get* format
	Input*			
		i1 - i9999 [pin number]		<i18> </i18> Input *get* format for all input types
	Input*			
		i1 - i9999 [pin number]		Digital <i18>x</i18> Input *put* format for all input types (x = 0 = off; x = 1 = on)

		Numeric <i18>nnn</i18> Input *put* format for all input types (nnn = whole or decimal number)
rpc		Level 1 for all remote procedure operations
	p1 - p10	Example: <p3> - slot 3 in the 10 slot RPC table.
gps		
	status	Read only – GPS Status Warning or OK Warning means do not use
	lat	Read only - Latitude
	lon	Read only - Longitude
	course	Read only - Course
	speed	Read only - Speed
	units	Read only – Units = Mph or Kph
	date	Read Only Date (UTC)
	time	Read only – Time (UTC)
	fixtype	Read only – Fix Type - 2D, 3D and None
	sats	Read only – Satellites – SatID1, SatID2 … SatIDn

Table 1

* **See Input/Output explanation in Section 4.**

Message Formats

Basic Message Format

Messages for Ctek's XML Interface are comprised of a series of paired tags. A tag begins with a < (less than) symbol and ends with a > (greater than) symbol. In the closing tag of a pair the tag identifier is preceded by a / (back-slash) character. An example of a very basic tag pair would be **<msg> some message </msg>**.

There are five basic commands used in the XML interface; **Get** (read), **Put** (write), **Status** (RPC), **Start** (RCP) and **Stop** (RPC). The message's command is identified between the tag pair <cmd></cmd>.

Note – Case Sensitivity:

> Ctek's XML interface accepts all supported XML tags in upper case, lower case, or mixed case.
> Internally the tags are all converted to lower case for processing. Replies are always with all tags in lower case and no attempt is made to modify the case of any parameter that is not within the XML <> brackets.

Get Command

The format of a message for Get Request is as follows:

<msg><seq>xxxx</seq><cmd>get</cmd> <L1><L2>><Ln></Ln></L2></L1></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<cmd> is the command requested.
<reply> replaces cmd in response messages
<L1> is the first (highest) level. L1 is mandatory
<L2> is the next level required. L2 and lower are optional
<Ln> represents any subsequent levels that may be requested

Get Response

The format of a message for a Get response is as follows:

<msg><seq>xxxx</seq><reply>get</reply> <L1><L2>><Ln></Ln></L2></L1></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<reply> replaces cmd in response messages
<L1> is the first (highest) level. L1 is mandatory
<L2> is the next level being addressed.
<Ln> represents and subsequent levels that may be specified

Put Command

The format of a message for a Put request is as follows:

<msg><seq>xxxx</seq><cmd>put</cmd><L1><L2>><Ln></Ln></L2></L1></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<cmd> is the command requested.
<reply> replaces cmd in response messages
<L1> is the first (highest) level. L1 is mandatory
<L2> is the next level being addressed.
<Ln> represents and subsequent levels that may be specified

Put Response

The format of a message for a Put response is as follows:

<msg><seq>xxxx</seq><reply>put</reply><status>0</status><statusdesc>Ok</statusdesc></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<reply> replaces cmd in response messages
<status> the tag that brackets the status resulting from the put, start, or stop operation
Status Value *0 – Ok* or *1 - Error*
0 – Ok = Request performed
1 – Error = Bad parameter
<statusdesc>
Status Description – Textual description of the return status
Ok
Error

Status Command

The format of a message for a Status request is as follows:

<msg><seq>xxxx</seq><cmd>status</cmd><rpc><rpc_id>></rpc_id></rpc></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<cmd> is the command requested.
<rpc> is the first (highest) level and is appropriate for all RPC commands
<rpc_id> is the alpha-numeric ID of the RPC slot being started (p1 - p10)

Status Response

The format of a message for a Status response is as follows:

<msg><seq>xxxx</seq><reply>status</reply><rpc><rpc_id>Status Response</rpc_id></rpc></msg>

Where:

<msg> is a mandatory start of message tag

<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<reply> replaces cmd in response messages
<rpc> is the first (highest) level and is appropriate for all RPC commands
<rpc_id> is the alpha-numeric ID of the RPC slot being started (p1 - p10)
Status Response

0 - No Task
1 - Starting
2 - Running
3 - Stopping
4 - Killing
5 - Terminated

Start Command

The format of a message for a Start request is as follows:

<msg><seq>xxxx</seq><cmd>start</cmd><rpc><rpc_id>RPC_Name Arg1 Arg2 Argn</rpc_id></rpc></msg>

Where:
<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<cmd> is the command requested.
<rpc> is the first (highest) level and is appropriate for all RPC commands
<rpc_id> is the alpha-numeric ID of the RPC slot being started (p1 - p10)
RPC_Name is the file name of the process (RPC) to be started.
Arg[1-10] is a space separated list (10 maximum) of parameters that are passed to the RPC upon startup.

Start Response

The format of a message for a Start response is as follows:

<msg><seq>xxxx</seq><reply>start</reply><status>0</status><statusdesc>Ok></statusdesc></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<reply> replaces cmd in response messages
<status> the tag that brackets the status resulting from the put, start, or stop operation

Status Value *0 – Ok* or *1 - Error*

0 – Ok = Request performed

1 – Error = Bad parameter

<statusdesc>

Status Description – Textual description of the return status

Ok

Error

Stop Command

The format of a message for a Stop request is as follows:

<msg><seq>xxxx</seq><cmd>stop</cmd><rpc>< rpc_id ></ rpc_id ></rpc></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<cmd> is the command requested.
<rpc> is the first (highest) level and is appropriate for all RPC commands
<rpc_id> is the alpha-numeric ID of the RPC slot being started (p1 - p10)

Stop Response

The format of a message for a Stop response is as follows:

<msg><seq>xxxx</seq><reply>stop</reply><status>Status Value</status><statusdesc>Status Description</statusdesc></msg>

Where:

<msg> is a mandatory start of message tag
<seq> is a mandatory start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
<reply> replaces cmd in response messages
<status> the tag that brackets the status resulting from the put, start, or stop operation

Status Value *0 – Ok* or *1 - Error*

0 – Ok = Request performed

1 – Error = Bad parameter

<statusdesc>

Status Description – Textual description of the return status

Ok

Error

Note - Messages with a bad or unrecognizable format may also receive a NAK (negative acknowledgment or not acknowledged)

Message Delivery Confirmation

If NAK is enabled on a SkyRouter and a format error is detected the NAK response will be sent. It takes the form of:

<nak><seq>xxxx</seq></nak>

Where:
<nak> indicates message type
<seq> is the start of sequence message tag
xxxx is an ASCII representation of a sequence (variable length - 4 digit maximum) number identifying a specific message
</seq> is the sequence terminator
</nak> indicates end of negative acknowledgement message

The application will send NAK if anything after the <msg> tag is malformed. This means that in some cases, the application will send a NAK without having received a valid sequence number. In those cases, the NAK will contain a sequence number of 0000. Therefore, it is recommended that user applications not use a 0000 sequence number when sending messages to this application.

Details of Specific Message Types

PLC Messages

For plc ***get*** messages the level three tag is a number referencing a specific I/O pin. The level three area of this message type is can be multi-valued, that is more than one pin can be specified.

Note - Recursion allows for a single ***get*** against inputs or output status. This allows the status of all active (enabled) inputs or outputs to be retrieved with a single ***get*** message.

In the case of a ***put*** messages the level three tag is also a number referencing a specific I/O pin. Again, for ***put*** commands the level three area of the message can be multi-valued, valued, meaning that multiple output pins can be set in a single message.

The level 2&3 portion of a ***get*** message to input i7 would take the form <input><i7></i7></input>, and a possible response to this message would be <input><i7>3.14</i7></input> where 3.14 is the value read on pin i7.

The level 2&3 portion of a ***put*** message to turn on output o23 and turn off output o37 would take the form <output><o23>1</o23><o37>0</o37></output>. Put messages for outputs use a 0 (zero) to turn an output off and a 1 (one) to turn an output on.

The level 2&3 portion of a ***put*** message to set virtual numeric output 19 to 3.14 and turn off physical output o7 would take the form <output><o19>3.14</o19><o7>0</o7></output>.

The level 2&3 portion of a ***get*** message for all inputs would take the form <input></input>. A possible response to this message for a unit having three inputs, i1 (digital), i2 (analog), and i3 (pulse) might be <input><i1>0</i1><i2>3.14</i2><i3>119</i3></input>

PLC Inputs and Outputs

Get messages can read digital, analog, pulse, and numeric inputs as well as the status of digital and numeric outputs. The return value is an ASCII text representation of the value or state of the pin(s) being interrogated and is found inside the level 3 pin identifying block(s). The response to a get message is formatted appropriately for the type of input pin being read.

Beginning with SkyRouter release 4.1.6 the Automation Control application (PLC) provides both physical and virtual inputs and outputs. The following table defines the read/write options and data types provided.

Pin Type	**R**ead/**W**rite	Data Type
Physical Input - Digital	R	Digital – 0 or 1
Physical Input - Analog	R	Floating point – Whole or decimal
Physical Input - Pulse	R	Floating point – Whole or decimal
Virtual Input - Digital	R/W	Digital – 0 or 1
Virtual Input - Numeric	R/W	Floating point – Whole or decimal
Physical Output - Digital	R/W	Digital – 0 or 1
Virtual Output Digital	R/W	Digital – 0 or 1
Virtual Output Numeric	R/W	Floating point – Whole or decimal

Get response from pin o14 (digital output) - A get command directed at a digital output will return the current state of the digital output. Example: A response containing <output><o14>0</o14></output> indicates that digital output pin o14 is in an off state.

Get response from pin i11 (digital input) - A get command directed at a digital input will return the current state of the digital input. Example: A response containing <input><i11>1</i11></input> indicates that digital input pin i11 is in an on state.

Get response from pin i2 (analog input) - A get command directed at an analog input will return a signed decimal number indicating the current value of the analog input. Example: A response containing <input><i2>-1.749</i2></input> indicates that analog input pin i2 is sensing a value of -1.749.

Get response from pin i5 (pulse input) - A get command directed at an pulse input will return an integer value between 0 and 32,767 indicating the number of pulses received since the pin was last read. Example: A response containing <input><i5>273</i5></input> indicates that pulse input pin i5 has counted 273 pulses since last read.

Disabled and Nonexistent Pins In PLC Messages

As I/O modules are added to the controller the pins (both in and out) are assigned sequentially to a pool of I/O pins. The pins assigned to this pool are considered to be within the scope of the configuration. As an example, adding a single 16-port I/O module to a controller creates 8 input pins and 8 output pins. The output pins will always be in a 0 (zero) or 1 (one) state. Output pins that have not been configured are set to zero. Input pins can be configured as analog, digital, pulse, or disabled. Input pins that have not been configured are in a disabled state.

When an output pin is read with a get message the value returned will be a one or a zero so long as the address of the pin (pin number) is within the pool of pins assigned by configuring modules. A request to get all output pins <output></output> will return either a 0 or a 1 value for each output pin in the configuration. A get message for a pin number that does not exist within the configuration will return an error.

When an input pin that is within the configuration is read the value returned can be a properly formatted analog return, a properly formatted pulse return, a digital return vale (1 or 0), or null. The null or empty return value indicates that the input pin is disabled. As an example a null return for pin i5 would look like <i5></i5>. A request to get all input pins <input></input> will return either a formatted value for each input pin in the configuration These return values could be a mixture of analog, digital, pulse, and null returns. As with outputs, a get message for a pin number that does not exist within the configuration will return an error.

Temperature Logs

Level 3 specifics for get currenttemplog and get lasttemplog.

The currenttemplog and lasttemplog conform to the basic data model described in this document. Both return values have the additional characteristics of being multi-valued and temporal. Both logs record the daily high and low radio temp observed on a SkyRouter. The values are recalculated every 15 seconds and at midnight of each day, the extreme values are written to a log file (currenttemplog). At the end of the calendar month, the log file becomes lasttemplog and a new currenttemplog is created. This feature is separate from the xml client but the xml client is used to read the current and last logs.

The <currenttemplog></currenttemplog> and <lasttemplog></lasttemplog> tags have been added to the xml client and are appropriate for a GET request. If the logs are not there, the xml will return the following text between the tags:

No Data

If the logs are there, the xml will return the following between the tags:

mm/dd,ll,hh(cr)mm/dd,ll,hh(cr).....mm/dd,ll,hh(cr)

Where mm is month 0-12, dd is day 1-31, ll is low temp and hh is high temp and (cr) is carriage return. Note that ll and hh could appear with a minus sign in front of them under the right conditions.

Relay Out (Put and Get)

<**relayout> [Put or Get]**

0 - turn relay off

1 - turn relay on

Turn relay on example:

A put request of:

<msg><seq>0001</seq><cmd>put</cmd><router><relayout>1</relayout></router></msg>

If successful returns:

<msg><seq>0001</seq><reply>put</reply><status>0</status><statusdesc>Ok></statusdesc></msg>

If a bad parameter is used returns:

<msg><seq>0001</seq><reply>put</reply><status>1</status><statusdesc>Error></statusdesc></msg>

Relay status example:

A get request of:

<msg><seq>0001</seq><cmd>get</cmd><router><relayout></relayout></router></ms>

If the relay is off returns:

<msg><seq>0001</seq><reply>get</reply><router><relayout>0</relayout></router></msg>

Message examples

Get sequences

Request to get router model:
<msg><seq>5678</seq><cmd>get</cmd><router><model></model></router></msg>

Reply to get router model:
<msg><seq>5678</seq><reply>get</reply><router><model>4200S</model></router></msg>

Request to get router IP address:
<msg><seq>1234</seq><cmd>get</cmd><router><wireless><ipaddress></ipaddress></wireless></router></msg>

Reply to get router IP address:
<msg><seq>1234</seq><reply>get</reply><router><wireless><ipaddress>72.236.100.185</ipaddress></wireless></router></msg>

Request to get plc value of input i20 (digital input):
<msg><seq>1234</seq><cmd>get</cmd><plc><input> <i20></i20></input></plc></msg>

Reply to get plc value of input 20 (digital input):
<msg><seq>1234</seq><reply>get</reply><plc><input><i20>1</i 20></input></plc></msg>

Request to get plc value of outputs 100, 101, and 102 (digital outputs):
<msg><seq>1234</seq><cmd>get</cmd><plc><output><o100></o100><o101></ o101><o102></ o102></output></plc></msg>

Reply to get plc value of outputs 100, 101, and 102 (digital outputs):
<msg><seq>1234</seq><reply>get</reply><plc><output><o100>0</ o100><o101>0</o101><o102>1</o102> </output></plc></msg>

Request to get plc value of input 5 (analog input):
<msg><seq>1234</seq><cmd>get</cmd><plc><input> <i5></i5></input></plc></msg>

Reply to get plc value of input 5 (analog input):
<msg><seq>1234</seq><reply>get</reply><plc><input><i5>-33.98</i5></input></plc></msg>

Request to get plc value of input 7 (pulse input):
<msg><seq>1234</seq><cmd>get</cmd><plc><input><i7></i7></input></plc></msg>

Reply to get plc value of input 7 (pulse input):
<msg><seq>1234</seq><reply>get</reply><plc><input><i7>53</i7></input></plc></msg>

Request to get entire port forwarding entry 7:
<msg><seq>5678</seq><cmd>get</cmd><router><portforward><f7></f7></portforward></router></msg>

Reply to get entire port forwarding entry 7:
<msg><seq>5678</seq><reply>get</reply><router><portforward><f7><addr>192.168.1.48</addr><state>1</state><srcport>5900</srcport><dstport>1783</dstport><protocol>2</protocol></f7></portforward></router></msg>

Port forwarding entry 7 is forwarding wireless network's port 5900 to port 1783 of LAN side address 192.168.1.48. Port forwarding port 7 is enabled for both TCP and UDP traffic

Request to get GPS Latitude, Longitude, Date and Time

<msg><seq>0000</seq><cmd>get</cmd><gps><lat></lat><lon></lon><date></date><time></time></gps></msg>

Response to get GPS Latitude, Longitude, Date and Time

<msg><seq>0000</seq><reply>get</reply><gps><lat>N 33-45.698148</lat><lon>W 118-17.688750</lon><date>051012</date><time>223220</time></gps></msg>

Request to get APN (all values)

<msg><seq>0000</seq><reply>get</reply><apn></apn></msg>

Response to get APN (all values)

<msg><seq>0000</seq><reply>get</reply><apn><name>Gold</name><authtype>3</authtype><username>Acme</username><password>my_secret</password></apn></msg>

Put sequences

Request to set router relay output state:

Relay on:

<msg><seq>0001</seq><cmd>put</cmd><router><relayout>1</relayout></router></msg>

Relay off:

<msg><seq>0001</seq><cmd>put</cmd><router><relayout>0</relayout></router></msg>

Reply to set router relay output state:

<msg><seq>0001</seq><reply>put</reply><status>0</status><statusdesc>Ok></statusdesc></msg>

<msg><seq>0001</seq><reply>put</reply><status>1</status><statusdesc>Error></statusdesc></msg> (bad parameter)

<nak><seq>0001</seq></nak> (bad format if NAK enabled)

Request to set plc outputs:

Output i1 and i20 on:

<msg><seq>0001</seq><cmd>put</cmd><plc><output><i1>1</i1><i20>1</i20></output></plc></msg>

Output i1 and i20 off:

<msg><seq>0001</seq><cmd>put</cmd><plc><output><i1>0</i1><i20>0</i20></output></plc></msg>

Reply to set plc outputs:

<msg><seq>0001</seq><reply>put</reply><status>0</status><statusdesc>Ok></statusdesc></msg> (success)

<msg><seq>0001</seq><reply>put</reply><status>1</status><statusdesc>Error></statusdesc></msg> (bad parameter)

<nak><seq>0001</seq></nak> (bad format if NAK enabled)

Request to set port forwarding entry 7 IP Address to 192.168.1.188 and Source Port to 5800:

<msg><seq>0003</seq><cmd>put<cmd><router><portforward><f7><addr>192.168.1.188</addr><srcport>5800</srcport></f7></portforward></router></msg>

Reply to set port forwarding entry 7 IP Address to 192.168.1.188 and Source Port to 5800:

<msg><seq>0003</seq><reply>put</reply><status>0</status><statusdesc>Ok</statusdesc></msg>

Request to set Reboot/Restart the SkyRouter

<msg><seq>0003</seq><cmd>put<cmd><router><reboot></reboot></router></msg>

Reply to set Reboot/Restart the SkyRouter

<msg><seq>0003</seq><reply>put</reply><status>0</status><statusdesc>Ok</statusdesc></msg>

RPC sequences (RPC operations only)

Start RPC Hello_world in slot p3 with startup parameters 17 and 409
<msg><seq>1234</seq><cmd>start</cmd><rpc><p3>Hello_world 17 409</p3></rpc></msg>

Response
<msg><seq>1234</seq><reply>start</reply><status>0</status><statusdesc>Ok></statusdesc></msg> (success) Or …
<msg><seq>1234</seq><reply>start</reply><status>1</status><statusdesc>Error></statusdesc> </msg> (bad parameter)

Stop RPC running in slot p3
<msg><seq>1234</seq><cmd>stop</cmd><rpc><p3></p3></rpc></msg>

Response
<msg><seq>1234</seq><reply>stop</reply><status>0</status><statusdesc>Ok></statusdesc></msg>

Status of RPC running in slot p3
<msg><seq>1234</seq><reply>status</reply><rpc><p3>2 - Running</p3></rpc></msg>
<msg><seq>1234</seq><reply>status</reply><rpc><p3>5 - Terminated</p3></rpc></msg>

Configuring the XML Application

Application Settings

The XML Interface application is configured using the SkyRouter's screen editor to change configuration file /etc/conf.d/opt.xmlclient. The configuration file contains the following three tags which are used to control the application.

TCP_PORT="5060"

MSG_ACK="Y"

MSG_NAK="Y"

TCP_PORT can be set to any valid TCP/IP port

MSG_ACK can be set to Y (yes, send acknowledgements) or N (no, don not send acknowledgements)

MSG_NAK can be set to Y (yes, send NAKs) or N (no, don not send NAKs)

Note - There is a 5 minute inactivity timeout on the TCP socket connection for both the XML and SMS clients. If a listening process exits or the connection fails, the client will disconnect, free up the port and go back to listening for a new connection after 5 minutes of no activity.

Remote Procedure Calls (RPC)

An RPC can be any executable program or script. The actual programs reside on the SkyRouter in the directory /etc/rpc. An RPC is started by its name and can have up to 10 arguments (parameters) passed to it. When an RPC is started it is placed in a slot (1 - 10) of the rpc table that is specified by the start command. status and stop commands are directed at the slot number of the desired process.

Each slot in the rpc table can have a unique process resident or multiple slots could have the same process started with different arguments. When an rpc is stopped its slot becomes available for another rpc.

Message Attributes For <Router>

<router> - The root of the message hierarchy

<model> [Get Only] SkyRouter model number Ex: 4100E, 4300S, etc

<hardware> [Get Only] installed radio Ex: Sierra 5727, Cinterion MC75i, etc.

<firmware> [Get Only] SkyRouter firmware release level

<wireless> [Get Only] Groups parameters below which represent the wireless status web page

<rssi> Signal strength in -dBm

<networkid> Network upon which the unit is operating

<servicetype> Connection being offered by network Ex: EVDO Rev A, GPRS, etc.

<serviceclass> Same as service type

<band> Frequency that the unit is operating at EX cellular, PCS, etc.

<roam> Roaming indicator (yes or no)

<servicestatus> Network service status – In service, not in service

<connectstatus> Call status – connected, not connected, dormant, etc.

<ipaddress> IP address assigned by wireless network

<deviceid> ESN for CDMA units, IMEI for GSM/HSPA/UMTS

<phonenumber> The MIN/MDN (CDMA) or MSISDN of this router

<**temperature**> Temperature of the radio module at this time
<**currenttemplog**> Daily high and low temperature history of the radio module during the current calendar month.
<**lasttemplog**> Daily high and low temperature history of the radio module during the preceding calendar month.