Introduction to the shifthelper program

D. Neise

Contents

Abstract **2**

Status of operation before **3**
Missing no alarm . . . 3
Flare Alerts . . . 3
Robotic operation . . . 4

Feature List **4**
1. Check the status of the system and the QLA results . . . 4
Security Limits . . . 4
2. Alert the shifter . . . 5
3. Be robust . . . 5
4. Be easy to install and use: . . . 6
Calling it . . . 6
First time ever . . . 6
Complete start up example . . . 6
The running shifthelper . . . 8
Quitting . . . 8

Future plans **9**
Redundancy . . . 9

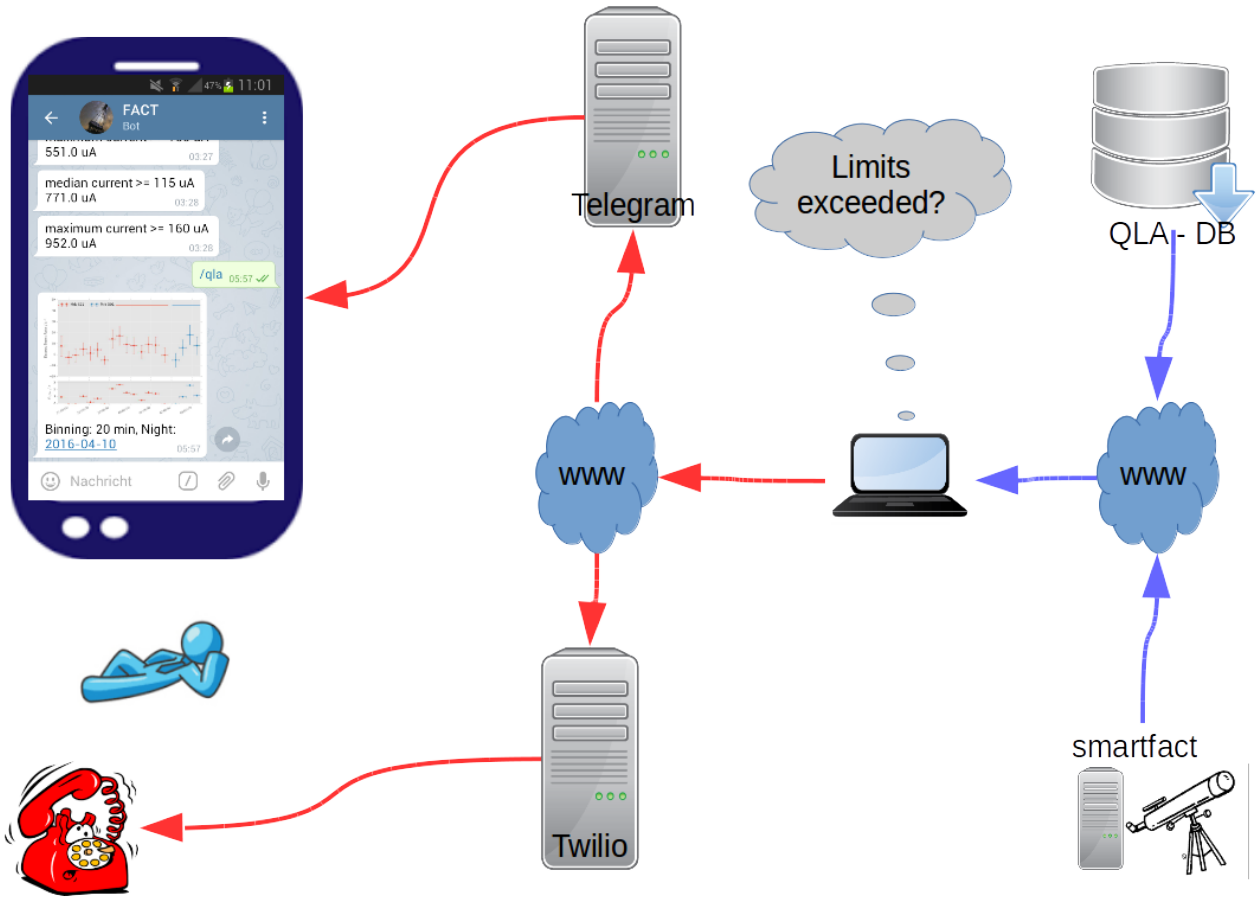
Spin offs **10**

Abstract

This document aims to serve as a rationale for the creation of the program `shifthelper`. It should also clarify certain design decisions.

This document can be found in its most recent form here:

https://github.com/fact-project/shifthelper/blob/master/rationale.md



Status of operation before

We write the year 2015 and the typical FACT night shift goes smooth. Shifters still need to restart `Main.js` quite often. And the startup in the evening and the shutdown in the morning are a tiny bit more effort than today (2016). During the night, the shifter can basically study or watch movies, while keeping an eye on the DAQ and the weather.

We were training some new colleagues for the night shift and had to answer a lot of questions like:

- When the `Main.js` program throws an exception, and I find nothing that looks horribly wrong. Can I just try to restart it once?
- In the instructions it says: "At high winds for longer time, park the telescope" What exactly do you mean by high winds and longer time?
- Here it says: "If the currents are high for a longer time, and there is no other source available, park the telescope", What exactly do you mean by high currents? What exactly do you mean by longer time?

The instructions were quite clear but not giving precise numbers in all cases. Leaving the decision to people who had no prior experience with the FACT hardware would either mean, letting them choose the save way and loose a lot of data, or letting the choose the wrong way and maybe loose the camera. So we had no choice but taking the decision for them in advance. We needed to provide more detailed specifications based on our prior knowledge.

This reminded me of a book about software design I read once.

> [...] specifying requirements in such detail [...] is programming. Such a specification is code.

This was the time, when we first thought: This can all be scripted.

Missing no alarm

The browser based smartfact program was already in place and helping a lot. One still needed the shell on La Palma back then for startup and shutdown, but only then. During the night your ssh-connection could break and you wouldn't notice until the morning, when you tried to close the lid.

smartfact was also very helpful in not forgetting about the different security limits. Assume currents were rising but you happened to forget about the limit. smartfact helped you by sounding an alarm when the currents rose to high. The same for high wind and rain. Also when for some reason the DAQ boards had a problem and blocked. Then you got an alarm telling you the trigger rate just went to zero.

However by far the most common reason for manual interaction at that time was for the top level program, which controlled everything, to throw an exception and stop. This did not cause an alarm but just a faint 'bing'.

So it happened that people were walking around their house missing that 'bing' and something around 30 minutes of data were lost, before they realized.

Flare Alerts

At some point we decided to send around flare alerts from FACT to interested people in the Cherenkov community. Night shifters now needed to decide if a source was flaring or not based on the QLA results, and in case it flared call Daniela Dorner in order to compile the according message. Depending on the binning every 5 to 20 minutes a new point would be added to the QLA plot and while studying or working, I typically used an alarm clock to remind me another 20 minutes had passed, so I wouldn't forget to check the QLA plot.

While specific flare alert limits were missing at first, after a while clear specifications were given in terms of excess events and significance, about when a flare limit is reached.

You can guess it already. This needed to be scripted. Why would I look at a plot every 20 minutes to see no flare, if the plot can also alert me, if it contains a flare?

Robotic operation

Last but not least the FACT collaboration decided in its general meeting of 2015 to progressively work towards a `robotic operation` of the FACT telescope. Despite the lack of a clear definition of the term it was clear that a robotic operation would a least in the first step involve a night shifter, which would be alarmed by the telescope, when ever manual interaction is needed.

The shifthelper program, was intended to call the shifter on the phone, so she does not miss anything important. Thus is was a natural step to let the shifthelper be one first test program of the collaborative effort of making FACT robotic.

Feature List

1. Check the status of the system and the QLA results
   - Limits
2. Alert the shifter
3. Be robust
4. Be easy to install on shifters PCs.

1. Check the status of the system and the QLA results

In order to sense the need to alert a shifter, the shift helper needs access to the relevant status information of the system. By comparison of the current state with a set of limits, the shifthelper decides about alerting the shifter.

All relevant information about the status of the FACT telescope is accessible via the DIM protocol. The first approach was to implement the shifthelper as a DIM-client, which would listen to the relevant servers in order to understand the current status of the system. Two reasons disqualified this approach.

First The DIM protocols implementation contains errors, which can lead to dead locks, which completely stall data acquisition.

Second in order to connect to the DIM protocol one needs to be connected to FACTs internal network. This can be either accomplished by running on a local machine or establishing a connection via VPN. It was not clear, how the shifthelper DIM client would affect the DIM network in case the VPN connection breaks.

Also, the shifthelper should monitor the availability of the network connection to the island. In case the island is not accessible anymore the shifter should be called in order to try and reach the MAGIC shift crew via telefone.

The second possibility to learn about the current status of the FACT telescope system is via a custom DIM to HTTP bridge named smartfact By requesting up to date information about the telescope via this bridge, any negative side effect of the shifthelper on the data acquisition was ruled out. Also it was irrelevant on which machine the shifthelper was running.

A custom module to request this information from the DIM-HTTPS-bridge was written by Sebastian Müller under the name smartFACTcrawler.

Security Limits

A collection of conditions under which a manual interaction is needed, was available in the FACT by the name DataTakingLimits. As one can see, not all conditions in that list require an action, some require the shifter to "monitor the temperatures carefully". These conditions were not converted into alerts, since monitoring carefully is exactly what is done by the shifthelper. Other conditions, like the Zenith angle of the observed source are predictable and thus do not need to be converted into an alert.

Not mentioned in the list of DataTakingLimits, are the flare alerts. By studying the flare alerts post in the FACT logbook, the shifter can decide under which conditions a flare alert might be sent to interested members of the cherenkov astronomy community.

The following list of alerts has been subsequently derived from the available information. It can of course be easily changed or enhanced.

Check every minute:

- Is Main.js still runnning?
- Is wind speed below 50 km/h?
- Are wind gusts below 50 km/h?
- Is outside humidity below 98%?
- Is median current per SiPM below 115 uA?
- Is maximum current per SiPM below 160 uA?
- Is the relative camera temperature below 10°C?

Check every 5 minutes:

- Is the excess event rate of each observed source below its own limit
  - Is it significant? (>= 3 sigma)

It might be interesting to add

- Is the Drive Ctrl-Deviation small enough?

At the moment, the maximum current per SiPM does ignore the bias patches containing crazy pixels. One might consider to also ignore channels containing a star, in order to not stop observation in case a bright start is in the FoV.

2. Alert the shifter

When ever one of the above conditions is not fulfilled anymore or any exception is thrown by a part of the running code, the program will react by alerting the shifter. This is done via two largely independent ways.

The first way to alert the shifter is via telephone, the shifthelper uses the service of Twilio in order to generate the call. At the moment no information as to why the shifter was called is transmitted through the call itself, i.e. there is no machine generated message played. Also the shifthelper does at the moment not care if the call was rejected, taken by the shifter, or taken by an answering machine. After placing the call, the shifthelper is designed to keep on going, i.e. in case the condition for a call is continuously fulfilled the shifter will recieve a call every minute.

The second way to alert the shifter is via the instant messaging app Telegram, the shifthelper sends Telegram messages to the shifter, which can depending on the shifters settings result in a sound being played or not. In any case the shifter recieves a text message conveying why an alert was sent, in case of a flare alert an image is generated for the shifter to crosscheck the status immediately. In case an alert was sent due to a software exception, the complete stack trace is appended to the message.

3. Be robust

We were concerned what might happen if the shifthelper program looses conection to the internet for a certain time. Will it throw an uncaught exception, which results in a program crash? Therefore the independent parts of the program were implemented as independent threads, which keep running even if e.g. the Telegram interface completely stalls and hangs due to a program error, the telephone call would still be dispatched and vice versa. And in case the DIM-HTTP-bridge cannot be reached, resulting in a fatal error due to a programming error, the resulting exception would still be caught and transformed into an alert for the shifter.

In the future, we plan to convert programming errors into calls to the shifthelper core developers, so they are punished for their programming errors and not the shifter. In the first place however, the shifters had to suffer.

4. Be easy to install and use:

Some time was also spend for streamlining the installation process of the shifthelper program on each shifters PC. The program was developed in Python. For details about the installation process please have a look at:

https://github.com/fact-project/shifthelper#install

Calling it

After installation, a binary called `shift_helper` is available in the users path. The program can be executed as easy as:

```
shifthelper +4177123456
```

A little help page is available of course.

```
shifthelper --help
```

First time ever

In order to access FACTs database for checking the QLA status, login credentials are needed. Also for calling the shifter using a Twilio and/or Plivo account needs certain login credentials.

On the first program start the shifthelper needs to download the necessary config file from fact-project.org to the users machine. The first program start might look like this:

```
ddneise@lair:~$ shift_helper +4177123456

================================================================================
                          Welcome to the shift_helper!
================================================================================

Please enter the current FACT password
Twilio Phone Setup
You entered:  +4177123456
Is that number correct? (y/n): y
```

The shifthelper will prompt the user for a password if necessary. The `scp` will fail in case the user has a different username on `fact-project.org` and the current machine. In that case the user needs to manually copy the file from La Palma to his own machine.

Complete start up example

The program starts by checking if the user has entered the correct number. In case the user confirms, the given number is called once, to check the connection. Consequently the user is asked whether she likes to be informed also via telegram. In case of a positive response, the shifthelper expects the user to identify herself by sending a message to "@factShiftHelperBot". Upon reception of the message, it returns the name of the telegram user having sent the message, so the shifter can crosscheck. An example startup of the shifthelper program looks like this:

```
dneise@lair:~$ shift_helper +4177123456

================================================================================
                          Welcome to the shift_helper!
================================================================================

Twilio Phone Setup
You entered:  +4177123456
Is that number correct? (y/n): y
```

```
I will try to call you now
Did your phone ring? (y/n): y

Telegram Setup
Do you want to use Telegram to receive notifications? (y/n): y

Please visit www.telegram.me/factShiftHelperBot
click on "SEND MESSAGE", the Telegram App will open.
Send the message "/start" to the shiftHelperBot.

Did you send the /start command? (y/n): y
Got a message from "Dominik Neise"
Is that you? (y/n): y
I will send you a message now.
Did you receive it? (y/n): y
```

The user can actually send any message to the @factShiftHelperBot, so identify himself.

The running shifthelper

After successfully starting the application, the screen will clear and one sees something like this:

```
2016-02-19 11:40:24

System Status                          Maximum Source Activity
bias current median       -5 uA
bias current max           5 uA
wind gusts               0.0 km/h
wind speed               0.0 km/h
Main.js                 Idle
rel. camera temp.         4.8 K
humidity                100.0 %




11:40:19 - INFO - shift_helper | Entering main loop.
11:40:19 - INFO - shift_helper | All checkers are running.
11:40:19 - INFO - shift_helper | Using Telegram
11:39:53 - INFO - shift_helper | Using phone_number: +41774528842
11:38:58 - INFO - shift_helper | version: 0.3.4
11:38:58 - INFO - shift_helper | shift helper started
```

The topmost line shows the last time the screen was updated.

The upper part of the screen shows how the system status looks for the shifthelper program. So this may be used as a poor mans check to see if the DIM-HTTP-bridge is working at the moment. On the upper right part, a list of sources would show up, together with the maximum excess rate measured for each source during the current night. In the lower part of the user interface one can always see the last few lines of the logfile, with the most recent entry up.

There is no way for the user to interact with the running shifthelper, no way to change the times between checks or the like. The only use of this command line interface is to show the user the current status of the program.

Quitting

In the morning after shutdown the shifthelper can be stopped by simply killing it with `Ctrl-C`. The screen will be cleared again and one is greeted with this message:

```
Thank you for using shift_helper tonight.
-----------------------------------------
We hope it was a pleasant experience.
Please consider sending your log-file:
/home/dneise/.shifthelper/shifthelper_2016-02-18.log
to: neised@phys.ethz.ch for future improvements
```

As one can see, during the current test phase, there was unfortunately no functionality implemented into the program to automatically send the logfile to the development team.

Future plans

The shifthelper software repository has an issue tracker, which is full of ideas and plans for improvements. So in case you are missing a feature or found a bug, please head there to

https://github.com/fact-project/shifthelper/issues

and report it. Its free.

Redundancy

Under the following conditions the shifthelper program can not do its work properly:

- The shifthelper program has no Internet connection, i.e. can not place a call.
- The shifter has no telephone connection, i.e. cannot receive the call.

Redundancy is our proposed strategy against these circumstances. We propose to run two or more shifthelper instances on independent machines, without any interaction between each other.

The code base should stay as simple as possible.

In addition we propose to have two shifters on call. The second shifter being called in case the first shifter does not bring the system into a safe state in a certain time.

We propose not to care for the reason for the delayed reaction of the first shifter. Again to keep the code base extremely simple.

This approach needs a modification of the checks done by the shifthelper. The current situation is:

Check	Period	Reaction
Main.js not running?	60 sec	Immediate call
Wind speed > 50 km/h	60 sec	Immediate call
Wind gusts > 50 km/h	60 sec	Immediate call
Humidity > 98%	60 sec	Immediate call
Median Current > 90 uA	60 sec	Immediate call
Max Current > 110 uA	60 sec	Immediate call
Rel. Cam Temp > 10°C	60 sec	Immediate call
Excess Rate > Limit	300 sec	Immediate call
Significance > 3	300 sec	Immediate call

This check list can in the redundant setup without inter shifthelper communication only be valid for one (the master?) shifthelper.

All other (slave?) shifthelpers must check, if the first shifter solved the problem in a certain time. If not the must assume either the first shifter is unable to react, or the master shift helper is unable to call. So they call the second shifter.

For some situations however, there is no way to solve the problem. The first shifter cannot slow down the wind or stop the rain. So we must define proper reactions, which can be checked by the secondary shift helpers.

For a stopped Main.js there is a defined solution. Get it running again. For high currents there are defined solutions. Ramp down the voltage. For wind, we say, the telescope must be parked. So in case the drive gets damaged, it does not matter, since it is already parked.

High humidity, is not a real danger for the telescope. We want to avoid it, but it does not really damage the telescope. So the secondary shifters do not check for this.

If the SiPMs get too hot, the bias must be switched off. The rel. temperature might stay high for some time, so the checkable reaction is to switch off the bias voltage.

Flare alerts are important, but having a flare just when the main shifter is not reachable by phone or the first

shifthelper does not have access to the Internet is not very likely. So we propose to take the risk to miss the call and not let the secondary shifterhelpers check for flares.

Check	and	after	Period	Reaction
Main.js not running?		5 minutes	60 sec	Immediate call
Wind speed > 50 km/h	not Parked	5 minutes	60 sec	Immediate call
Wind gusts > 50 km/h	not Parked	5 minutes	60 sec	Immediate call
Humidity > 98%	False		60 sec	Immediate call
Median Current > 90 uA		3 minutes	60 sec	Immediate call
Max Current > 110 uA		3 minutes	60 sec	Immediate call
Rel. Cam Temp > 10°C	Bias On	3 minutes	60 sec	Immediate call
Excess Rate > Limit	False		300 sec	Immediate call
Significance > 3	False		300 sec	Immediate call

Spin offs

Parts of the codebase of shifthelper, namely the monitoring of the telescope status via the HTTP-DIM-bridge, can be used to quickly implement checkers, which inform the entire collaboration via email, in case certain conditions are met, like e.g. the camera humidity has reached 40% or the container temperature during the night was higher than 29°C.