**Opslog (Basic) Requirements Doc**

In this document I try to layout the basic requirements for the Opslog tool based on using it’s replacement and discussing the current tool.

1. Operator Usage

1.2 Shifts

There exists a schedule, 12 hour shifts, 7 AM – 7 PM, 7 PM – 7 AM. This has existed since the life of the GBT and does not change. We’ll call this the scheduled shift. All reporting is tied to these scheduled shifts.

1.2.1 Operator Logins

Operators log into a specific shift.

Operators can log in and out at anytime, though only one operator can be logged in to a certain shift at any time. Operator login and loguts are treated as an entry in the operator’s log.

Operators give their name when they log into/out of a shift. This name is then associated with any entries they create and/or edit.

Note how operators login to the DSS using a shared account (‘gbtops’). This will have design implications in the new tool. But Pete can’t think of a good reason why the operators should continue logging into the DSS w/ this account, that using their own personal DSS accounts would not be a big deal.

1.3 Operators create Entries

The most basic of requirements is that operators can make an entry into the logs. The details depend on what type of entry they make:

Types of log entries:

* Operator Login
* Operator Logout
* Information
* Weather
* Maintenance
* Comment
* Scan Information
* Receiver Change
* Project Change
* Failure
* Failure Restore

The first five are very basic and only involve the following information:

* Scheduled shift the entry is associated with
* A unique Entry ID
* A time for the event we are logging
* A time and operator associated with the Entry’s creation and subsequent edits
* The Entry’s type
* A comment (text)

The following associations could be made, some of which can just be inferred by the time of the entry.

* Associated operator?
* Associated project?
* Associated Rx?
* Associated Observer?

A Comment entry can be thought of as child of one of the other types of entries, containing additional info on the parent Entry. Information includes same attributes as the above entries, plus one more:

* A pointer to the parent Entry .

The remaining types not only share all the attributes of the first Entry types, and include additional information, but also may contain relationships between themselves and other entries.

Log entry relationships:

* All Entries are associated with a scheduled shift.
* All Comment Entries are associated with some ‘parent’ Entry.
* All Failures are associated with a specific Project Change.
* All Failure Resotres are associated with an original Failure entry

Also, as noted above, all Entries could be associated with a project, observer and rx change – this could just be inferred from times, or through FKs?

1.2.1 Some Log Entry Details

1.2.1.1 Info – seems to be for general purpose info

1.2.1.2 Weather – this seems to contain a formatted string of weather summary info (may be auto generated by software)

1.2.1.3 Maintenance – specifically for including information about activities during a Maintenance project

1.2.1.4 Scan Info – specifically for making a comment about a range of scan numbers. Additional fields:

* start scan number
* end scan number

1.2.1 Observer Responsible

These are entered when a project is **not** changing, but the current observer **is**.

These aren’t necessary when observer changes are lined up with project changes. Additional fields:

* Observer name
* Room number
* Phone number

1.2.1 Receiver Change

These are entered when a project is **not** changing, but the receiver used for observations **is**. These aren’t necessary when receiver changes are lined up with project changes. Additional fields:

* Receiver name
* Turret rotated (yes/no)
* PF boom deployed (yes/no)

1.2.1 Project Change

This appears to be quit straightforward: when the DSS schedule says it’s time for the next project, one of these entries are created. Sometimes, an operator might enter these ahead of time. Additional fields:

* Observation type
* Observer name
* Proposal name
* Project (Astrid directory) name
* Receiver name
* Turret rotated (yes/no)
* PF boom deployed (yes/no)

One should note that Project Change info contains parts of Receiver and Observer Changes. Perhaps a Project Change should **not** contain Observer and Receiver info, but instead be **associated** with that info?

1.2.2 Failures

These are the most complex type of Entries. Each Failure Entry not only stores a lot of additional information, but it also needs to be associated with a Project (Change Entry).

Failure fields (in addition to basic Entry fields):

* Item: what broke?
* Track lost time: yes/no
* Work Order: ?
* Failure Type: a list of types
* Closed/Restored datetime
* Lost time: hours, minutes? Also, this could be a derived field
* Closed: yes/no
* Scale: can scale back lost time with this.

A Failure Entry and a Restore Failure Entry should always come in pairs, unless, of course, a failure is still open (has not been restored).

It appears that operators only enter the times of the Failure and it’s matching Restore, and whether to track lost time or not. These times in turn are used to calculate the number of hours lost, which may be the supervisor’s job.

Use Cases:

(from talks with Dave R.)

1.2.2.1 No Lost Time Failure in one Project

Here observations continue, although a problem comes up, which is resolved before the project is done.

* Something stops working, but isn’t affecting observing
* A failure entry is created: no lost time
* Before the project is finished observing, things start working again
* A restore failure entry is entered

1.2.2.2 Lost Time Failure in one Project

Here observations can’t continue as planned due to a problem, which is then resolved before the project is done. This project will claim some lost time.

* Something stops working, and **is** affecting observing
* A failure entry is created: **track** lost time
* Before the project is finished observing, things start working again
* A restore failure entry is entered

1.2.2.3 Lost Time Failure across multiple projects.

Here observations for more then one project cant’ continue as plan due to a problem. The problem is finally solved, but multiple projects will claim lost time.

* Something big stops working (say the Antenna), and **is** affecting observing
* A failure entry is created: **track** lost time
* The current project’s time is up, and a new project is started
* Before this new project’s time is finished, things start working again
* A restore failure entry is entered

1.2.2.4 Lost Time and No Lost Time Failure across multiple projects.

Here observations for the first project cant continue as planned due to a problem. But the next scheduled project is able to start and is not affected by the problem. Before this project’s observations are done, the original problem is solved. Only the first project will be claiming lost time.

* Something stops working (say Vegas), and **is** affecting observing
* A failure entry is created: **track** lost time
* The current project’s time is up, and a new project is started that **is not** affected by the current failure.
* The original failure, which is tracking lost time is closed by entering a restore failure entry.
* A new failure, associated with the new project, is opened, identical to the previous failure, but **without** tracking lost time.
* Before this new project’s time is finished, things start working again
* A restore failure entry for the second failure entry is entered

1.2.2.5 Lost Time in one project, across multiple shifts

Here observations are affected by a problem. Before the project is finished or the failure is restored, the current operator logs out, and a new operator logs in, starting a new scheduled shift. Shortly after, the failure is resolved before the project is done. This project will claim the same lost time regardless of the operator shift change.

* Something stops working, and **is** affecting observing
* A failure entry is created: **track** lost time
* A scheduled shift change occurs: old operator logs out, new operator logs in.
* Before the project is finished observing, things start working again
* A restore failure entry is entered

1.2.2.6 Multiple, concurrent failures in one project

Here observations are affected by one problem, though another, seemingly unconnected problem must also be reported, even though it is not affecting observations. The project will claim lost time only from the first of these failures.

* Something stops working, but is not affecting observing
* A failure entry (A) is created: **no** lost time
* Something stops working, and **is** affecting observing
* Another failure entry (B) is created: **track** lost time
* Before the project is finished observing, the second problem is cleared up.
* A restore failure entry for B is entered
* Before the project is finished observing, the original problem is cleared up
* A restore failure entry for A is entered.

1.3 Operators Edit Log Entries

Operators can create, read, and update Entries. However, they can only edit entries for shifts they own.

However, they can’t delete them. Ever.

1.4 Entry Views

Here we try to describe how they can view entries, w/ out getting into the details of the UI.

1.4.1 Entry Lists

The opslog needs 4 types of lists of entries

* Entries only for the current shift
* Entries, all of them! But readonly for operators!
* All failures that have lost time for the current shift – includes project & rx info. Readonly!
* All failures that are still open for all shifts

There is also a read only visualization of the schedule for the current shift: project, rx changes, and shift description (shift number, time range, operator name). This is always up.

1.4.2 Entry Creation/Edit

During an operator’s shift, they commonly use the view of the entries for their current shift to create and edit entries. For these actions, a basic form is used to enter entry details.

For basic entries, once the type is choosen, only these fields are edited, all others are readonly (but displayed):

* Datetime for event
* Comment

For other more complicated entries, the needed extra fields are also there in the form for editing.

1.4.2.1 Failure creation/editing

Of course, what behaves differently in this context are the Failure entries: when a new failure is entered, additional fields can be entered but if one chooses not to have it be ‘open’, the ‘closed/restored date time’ is readonly and blank, and ‘lost time’ total is readonly but calculated using the failure time and the end of shift time.

One can close a selected open failure, and simply enter the datetime and a comment, which creates a Failure Restore. But now the original Failure entry shows the closed time as the time of the Failure Restore entry, with ‘lost time’ calculated using that time.

One can do the above from really any view.

1.4.3 Log Details

This is an interesting view in the Opslog tool. For all types of entries is displays the same view, which means lots of times many fields are blank, but also some fields never get to be shown. Before we list these fields, we should also note that this view shows the link between an entry and any of these types of entry changes: project, observer, receiver.

* Entry Datetime
* Entry type
* Entry comment
* Entry created datetime/author
* Entry recent modification datetime/author
* Operator
* Failure Type
* Failure Item
* Failure Track lost time
* Failure Work Order
* Failure Restore Datetime
* Failure Restore author
* Failure Lost Time
* Observeration Type
* Proposal
* Project
* Observer
* Room
* Phone
* Receiver
* PF moved?
* Turret Moved?
* Log Key
* Shift Key
* Log Parent Key

For example, for an Info Entry, all the Failure fields will be blank, and parent key will be Null. But for a comment entry, the Failure fields will be blank, but the parent key will point to another log key.

As a further example, an entry right after a Project change will display the project, receiver and observer info related to that project change entry. If then, a receiver entry is entered, and a subsequent entry is made, that new entrie’s receiver field in this view will reflect that new receiver. When a new observer entry is made a similar thing happens.

1.y Reports

Operators can only do shift summary reports (see below)

1.x Multiple Telescopes

Here we dive into new territory (stuff the old opslog doesn’t do). Write now they are covering non-GBT entries just as type Info.

Dave R. says choosing a telescope first and then entering stuff is the way to go, but there will be cases where we’d want to enter things for **ALL** telescopes. Like if there’s a power outage on site …

Ideas:

* All entries will need to have a telescope type field
* A global filter at the top for telescope: everything shown is dependent on this.
* One of the choices would be All, which would do what exactly? You could enter in an entry that would get duplicated for all telescopes? NO, bad idea. How about the ALL entries show up no matter what telescope you filter for.

2. Supervisor

2.1 Entry management

Supervisor’s can do everything an operator can **and**:

* Delete entries
* Fine tune lost time calculations (see below)

2.2 Reports

2.2.1 Shift Summary

This basically lists all the entries associated with a given shift, and tying in information about the project schedule. Summaries of observations and lost time are also printed.

Shift Summary:

* Shift ID
* Operator name
* Shift time range
* Project ongoing at start of shift

Log Entries:

* Datetime
* Entry Type Code
* Description – for simple entries, this is just the entry comment. For more complex entries, like failures, this is a summary of info.
* Log Key
* Type of Project – single letter code for the type of project this entry is associated with

Interspersed with these lines are project changes.

Lost Time Summary:

* Proposal
* Receiver
* Failure (type + item)
* Log Key
* Lost time (hours)

Observing Summary:

* Observation type
* Proposal
* Receiver
* Duration

Proposals are split up by type, with a total time for each group.

2.2.2 Monthly Summary – must be supervisor!

For a given month, for each type of observing (Astronomy, Mainteance, Testing) the following tables are produced:

* Proposal
* Receiver
* Hours
* Lost Hours

Each combination of proposal and receiver gets it’s own line. And totals for time and lost time are produced.

It should be noted that this is then checked against the DSS monthly summary pages. The above format does not make this easy, as subtotals for a given project are done by hand, etc.

2.3 Failure Management

As noted above, operators only report on parts of a failure and when (if) a failure is restored. The fine tuning of how much time is considered lost is the supervisors job.

It appears that the supervisor can change the time range of the failure (simply changing the times of the entries), and use a scale factor to claim only part of that as lost time (defaults to 1.0).

3. GBT Staff

Opslog summary web page: almost identical to the Shift Summary Report! Only done in s a simple HTML web page format.