

Using RT to Keep Track of Bugs, Ideas, Users, and Your Life in General

Simon Myers

GBdirect • <http://www.gbdirect.co.uk/>

1 Introduction

Developers generally aren't fond of administration, yet all serious development projects involve admin to some degree. RT is a tool designed to ease this burden, keeping track of things so you don't have to. Its basic unit is a 'ticket' — a single issue which requires coding, discussing, documentation, declining, or in some other way resolving.

Much of the administration in a project falls into one of these categories:

- *Communication* between people working on the project, and with end-users or customers
- Keeping a *to-do list* of tasks to be undertaken, including bugs to be fixed
- Discovering the *status* of the project

This paper outlines some of the administration and communication problems frequently encountered, then examines how 'RT' can be used to address them.

2 Using E-Mail

E-mail is often the primary means of communication in projects. Since RT has its origins in e-mail helpdesks — 'RT' stands for 'Request Tracker' — the rôles of 'requester' and 'helper' will be used in these examples. However the situation is analogous to many other situations: a request could be a feature request or a bug report and the requester could be either another member of staff involved with the project or an end-user.

Typically a requester, let's call him Ron, will mail a helper, Harry, with the request. At some point later, hopefully Harry will respond to Ron. There are likely to be further mails between Ron and Harry, and the request should get dealt with. That sounds simple enough, but problems can arise:

- If Harry has many requests to deal with he's likely to be using his mailer to keep track of them, perhaps leaving mails still requiring attention in his inbox and using it as a to-do list. This is far from ideal since other mail gets mixed in with requests, and there isn't a sensible way of sorting tasks. The problem gets worse if Harry is involved in several projects.

- Harry may be part of a team who can deal with such requests, but mail sent straight to him isn't seen by anybody else.
- Harry's boss or other people he works with can't discover what requests Harry currently has to deal with other than by asking him.
- Another person brought in to assist with a particular request won't have access to the previous discussion, as the only record of the conversation is in Harry's mail folder. The existing mails will have to be forwarded to the new helper, then future mails on the issue should be between all three people now involved.
- If several months later Ron phones up with an urgent problem related to this request and Harry happens to be out of the office, it's hard for anybody else to discover what was done originally. Trawling through Harry's mail folders raises privacy issues and is unlikely to be fruitful.

Examining past correspondence can crop up in other circumstances — finding evidence to back up claims of work done for billing purposes, or because somebody else makes a similar request.

- If Harry leaves, the previous correspondence has to be passed on to his successor. Harry's e-mail address may have to live on after him, because requesters are used to mail it.

Some of the above issues can be addressed by having requests sent to a non-personal e-mail address especially for that purpose; such mails can then be forwarded automatically to a team of people. This isn't perfect though:

- There needs to be a policy for determining who deals with new incoming requests. Duplicate responses are fairly easy to spot, but it's hard to ensure that no requests are missed.
- Everybody still has to keep their own copy of everything, just in case they need to refer to it in the future.
- The only way of keeping track of requests is indiscriminately to receive every mail on them.
- Many people end up receiving many mails in which they have no interest. This creates a culture of people getting used to ignoring mails, and risks somebody deleting a mail they were supposed to act on.
- The system is reliant on everybody replying to mail to ensure that it gets sent to the appropriate people.

3 RT Intercepting Mail

RT helps with many of the above problems by intercepting all mail on requests. In its simplest form, Ron would mail the address set up for requests; this would be received by RT, which stores the mail in a database and forwards the mail to Harry. Similarly, Harry's response travels back through RT.

Harry and Ron could continue as before, largely oblivious to RT's existence. There are no worse off than with mailing each other directly, and having all the correspondence in a database makes it easier for others to find it, either to review the state of things now or to dig things up in future. RT provides a web interface for accessing its database, covered below.

If a second helper, Hermione, is taken on to help with requests in general, RT can be made to send mail to her as well; the requester does not need to do anything differently.

If a particular request involves somebody else helping, Hagrid say, then he can be added to those receiving correspondence about just that request.

RT distinguishes comments from correspondence. Comments are internal and do not get sent to the requester. This enables internal discussion, such as who is going to deal with a request, to be associated with the request but kept private from the requester.

4 How RT's Mail Works

The collective term for requests is a *queue*. Each new cry for help gets sent to the queue's e-mail address, such as `conquering-evil@example.com`. RT assigns a *ticket number* to the request and stores it in the database. The request is then mailed to the specified helpers, known as *queue watchers*. RT puts the ticket number in the subject of the mail.

Mails sent by RT always have the queue e-mail address in the `From:` header, but retain the original sender's name.

For example, if the original request had these headers:

```
From: Ron <ron@example.org>
Subject: Lost Invisibility Cloak
To: Helpdesk <conquering-evil@example.com>
```

the mail sent by RT to the queue watchers would have headers along the lines of:

```
From: Ron <conquering-evil@example.com>
Subject: [example.com #174] Lost Invisibility Cloak
To: Ccs of ticket #174: ;
```

The body of mail sent to queue watchers typically includes at the start some extra information about the ticket:

```
queue:      Conquering Evil
ticket:     Lost Invisibility Cloak
```

```
status:    new
owner:     nobody
intranet:  http://badger/Ticket/Display?id=174
```

When somebody replies to a request, the reply gets mailed to RT. RT will notice the existing ticket number in the subject line and forward the mail to everybody else related to that ticket, again setting the `From:` header to ensure that any further replies come back through RT.

Comments use a different e-mail address. For example, the e-mail address `conquering-evil-comments@example.com` could be set up for this purpose. Any mail sent to that address with a ticket number in the subject line will be forwarded to everybody involved with that ticket except for the requester.

It's possible, indeed usual, to have several queues in the system, one per project or team. Each queue has its own e-mail addresses, and can have different watchers.

5 RT's Web Interface

RT's web interface shows tickets in a queue and details of individual tickets. When a ticket has been resolved it should be marked as such; normally only new and open tickets are listed. Each ticket can be set a priority, an owner, and a due date. As well as queue watchers, each individual ticket can have watchers associated with it.

You can also set up multiple-choice keywords. These can denote anything you want:

- The version of the software affected
- Whether the ticket relates to a bug, a desired feature, or a clarification
- Which aspect of the system the ticket relates to (user interface, back end, documentation, etc).
- Whether a purchase order has been received for the work
- Which language the request is conducted in

It's possible to search for tickets by most fields, including queue, requester, owner, priority, due date, and keywords.

Tickets can be merged (useful when a requester sends a new mail instead of replying to an existing one) and linked to each other in hierarchies.

The web interface presents a complete history of all correspondence and comments relating to a ticket, and it's also possible to send correspondence and comments from there. Sometimes this can be easier than finding messages in your mail folders. It's also possible to create new tickets from the web interface.

6 Permissions

RT has a flexible, albeit complex, set of permissions. You can place users in groups then assign group permissions for different capabilities. Most permissions can be either global or per-queue; they include:

- Manipulate all tickets in a particular queue
- View tickets in a queue but not change them
- Create new tickets
- Edit keywords

All permissions can be set using the web interface.

7 Command-Line Interface

There are command-line programs capable of performing similar things to the web interface. This is less-user friendly, but has the advantage of being callable from other scripts. In programs such as Mutt and Vim you can assign keystrokes to perform RT commands, such as to take ownership of the current ticket or to mark it resolved.

8 Customizing RT

RT is incredibly customizable. The web interface lets you edit the *scripts* which determine which conditions result in who being mailed, and the *templates* which dictate what they get mailed.

RT is written in Perl. The processing is performed in a number of library modules and the web pages are generated with `HTML::Mason` components. This separation facilitates customizing RT's interface.

Changes to RT's appearance are fairly straightforward: you can personalize your RT, for example in your company's livery, with very little hassle. Small usability modifications can be achieved without much understanding of what's going on: for example, making the titles of tickets (rather than their numbers) clickable links in search results, or replacing list boxes with drop-down boxes.

More complex customization requires putting some effort into getting your head what RT's doing: it takes a couple of days to get the hang of which code does what and how it does it. This is not a criticism of RT: any complex system of that size is going to need a while to become familiar with it. RT uses many small components, making it easy to isolate modifications.

One of the best features of RT from a customization point of view is its local component tree. This is arranged such that any local component will automatically override a default component of the same name. This avoids ever having to fork the supplied RT code, which makes upgrading to newer RT release as painless as possible — your local changes don't get wiped out.

Heavy customization is not something to be undertaken lightly. But if RT does some of what you want, using it as a starting point is likely to involve less work in the long run than developing your own system entirely from scratch.

A publicly-available RT instance with a high degree of front-end tweaking can be seen at <http://rt.cpan.org/>. This service for Perl CPAN authors provides a separate RT queue for tracking bugs in each Perl module, and it has been made to look like CPAN rather than RT.

9 Deploying RT

RT's background is in helpdesks, but it now aims to be useful in many other areas; a new feature does not have to be useful in helpdesks to make it into RT.

As the CPAN RT system shows, RT can be used for tracking bugs in software. Typically there's one queue per major piece of software, with a ticket for each bug in need of fixing, enhancement to be implemented, or issue to be discussed.

Some of these could be originated by end users, but even if RT is just used internally it provides great benefits: all outstanding work is clearly listed on the intranet; developers no longer have crucial knowledge scribbled on the on the corners of their own mouse pads; when a bug is fixed, it's easy to mail everybody interested; when somebody's out of the office, you don't have to phone her/him at home because required information is only in her/his mail folders.

From a developer's point of view, RT can feel very liberating: not having to remember a dozen small bugs to work on in the future frees up your mind for concentrating on the catastrophe in need of fixing now.

RT can usefully be applied in a surprising number of situations — office politics, purchases, recruitment. When considering whether RT would be useful for a particular task, it's worth bearing these two points in mind:

- It isn't necessary to use all of RT in every deployment. For example, you can drive a personal to-do list entirely from the web interface without having any e-mail involved. Conversely, you could almost forget about the web interface, just using RT to intercept particular types of e-mail so that you have an archive on the intranet should it be needed.
- A ticket is merely a single 'thing' which can be resolved or completed in some way. RT can be used to track actions assigned at meetings, where performing the action predicates resolving its ticket.

Issues can also be tracked with RT: if a decision is needed on something, RT provides a convenient place to keep all discussion relating to the issue in one place (and somewhere to direct people who keep bringing up any old issues that have been discussed too much already). The ticket is then resolved when a decision is reached — actually undertaking the work decided upon may well require another ticket.

10 More Features

RT has several features and add-ons which this paper does not cover, but which may be of interest to you. These include:

- Hooks into CVS, so that committing changes automatically updates a ticket
- Fields for recording the time spent working on a ticket, and a running total of time so far.
- Graphing and statistics add-ons, for displaying data about pending tickets and the rate at which tickets are being resolved
- Priority escalation, so that tickets can automatically have their priority increased as their due date becomes closer.

11 Problems with RT

There are some irritations with RT. It's possible to merge two tickets into one, but not to split a single ticket into two. You can't send mail to yourself from the web interface. Values of keywords have to be selected from a list; free-text fields aren't possible. Sometimes it isn't possible to specify exactly the search criteria you want.

However, RT is actively being developed so some of these complaints are likely to go away in the near future.

A more fundamental problem with RT is the time that it takes to set up. In any situation a fair amount of configuring is required, and forcing an RT system on people does require them to go through a learning curve to use it correctly.

Situations where RT doesn't sound to do quite what you want by default are harder to deal with. It's likely that some customization will provide the desired functionality, but it's hard to determine this without putting considerable effort into getting acquainted with RT, and it's obviously a risk to devote resources to a system you haven't definitely decided to use.

Don't expect RT to be an instant solution to all of your admin problems. The system is still driven by humans, and they can still manage to be just as lazy, forgetful, and disorganized as they were prior to RT.

12 Finding Out More

RT is written in Perl and is completely free. It's available at <http://www.fsck.com/projects/rt/>. There are several mailing lists relating to RT. The RT users list is recommended if you start to use RT. Many people also share their customizations on the list, making them available for others to use.

13 Conclusion

RT can be of assistance in organizing your administration in a variety of situations, including managing software projects. If you're suffering from not being able to find information or having trouble keeping track of what needs doing, RT is worth considering. However, it requires a fair bit of effort setting up and you need to be prepared to delve in and customize it to get the most benefit for your organization.