Galaxy Small-Scale-Admin Poll Results

Vlad Visan

Galaxy Small-Scale-Admin February 2024 Poll Results

Vlad VISAN, March 6th 2024

Table of Contents

I About the document

I.A How to read the document/how it was made

A) Overall, I didn't change the question names nor the data
B) But I did change a few values when I saw they were incoherent (ex : an answer of " 0 " that was meant to be " -1 " (no answer) judging by the comment which clearly said " I don't know "). (However I did not systematically do this, I assume the answers to be correct).
C) I also renamed and re-ordered a few questions for better clarity
D) I removed a line entirely (" Comments " associated to end-user support) because it had 0 answers
E) Next to each graph that had a " Comment " section, I put the (value/comments) combinations for only the values that had comments.
F) The report includes Hans' answer from Feb. 20
G) (Most of the graphs were generated by a R script I wrote)

I.B Personal info :

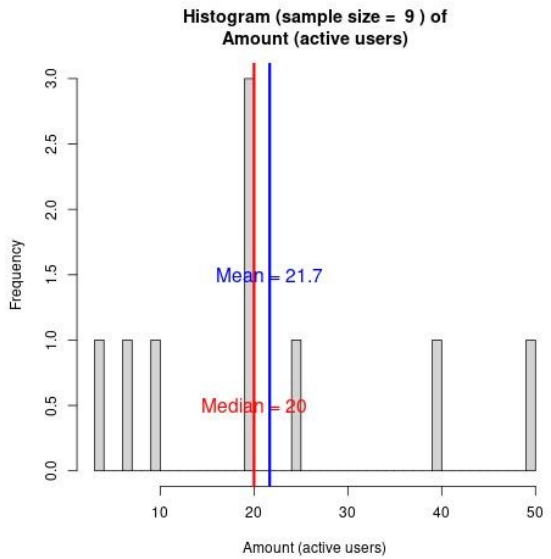
A) I removed the names/emails before analysing
B) I deleted the answers from the Framaforms website (after analysing them) since they contained names/emails.

I.C Limitations

A) " Computing " and " Object Store " should have been multiple-choice
B) The " end-user support " question and comment was vague
    - Accidentally called it end-user assistance
        - But the numerical "avg. nb hours of end-user support per month " still got seemingly valid answers
    - The associated" Comment " field was misleading (it said " Comment (tool packaging) " because of a copy/paste mistake)
        - It got no answers because of this
C) For the " Other recurrent admin. Tasks " question, I should have specified monthly/yearly.
    - As such, the apparent outliers might not actually be if divided by 12
    - I was intentionally vague since the question was meant to be open-ended, but I should've still been more precise.
D) Some arbitrariness in the time-length of some questions (week/month/year)
E) Some arbitrariness in some questions' location
    - " Do you use Ansible ? " could've been grouped together with " Do you group Gravity "
    - " Nb users " could have been in the " Users " section
F) No doubt some missing questions
G) Some potential overlap (e.g. "end-user support" and "other monthly admin.")
H) Overall feeling that, question-wise, some sections were too detailed while others not enough, but I don't have enough Galaxy expertise to really fine-tune this.

II Results starting from here : Galaxy instance characteristics

II.A Active users

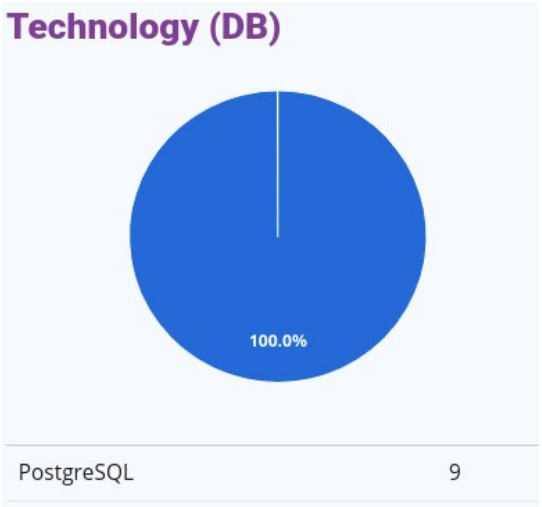


Amount (active users)	20	20	50	25	10
Comment (active users)	This number varies a lot, since there is a constant turnover at our institute...and not everybody who has attended the internal training sessions keeps on using Galaxy	Total 200	they're all students, generally they start/stop using the galaxy in batches of 20-30	There are about 5 times as many registered users as active users	total 84

Analysis :

- 3 categories : <10 // circa 20 // circa 50
- Active vs signed up: no problem having a lot more signed up than active users, no need to delete inactive users (in case they come back).

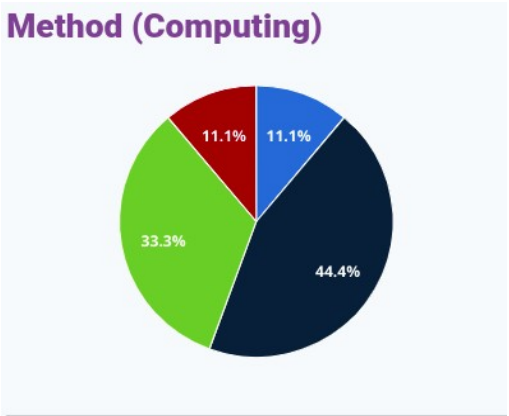
II.B DB



Technology (DB)	PostgreSQL
Comment (DB Technology)	Original, our server was run by a MySQL database, I have managed to do the transition (without loosing data) to PostgreSQL (see: https://galaxyproject.org/blog/2015-07-mysql-2-postgresql/)

Analysis : PostgreSQL necessarily

II.C Computing

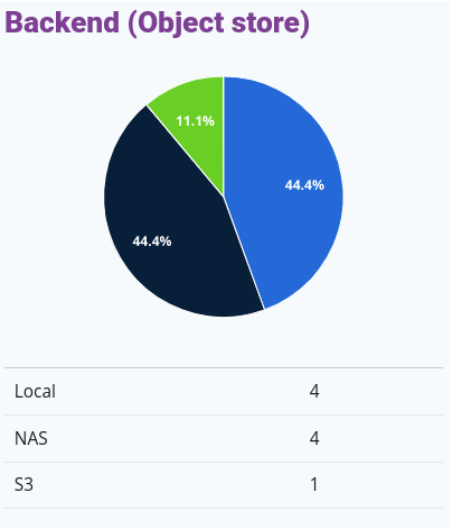


Pulsar	1
Batch scheduler	4
Local	3
Other	1

Analysis :
Multiple batch schedulers (usually HTCondor), sometimes with Pulsar, at least one on a separate machine from the batch scheduler, at least one on the same.

Method (Computing)	Local	Batch scheduler	Local	Batch scheduler	Pulsar	Batch scheduler	Other
Comment (Computing)	everything runs on a 28cores (double threaded) box	SLURM	sbatch scheduler on the same machine	Local HTCondor Deployment	HTCondor again, in addition to pulsar	Jobs run on a compute cluster provided by the university's central research IT team using a local custom job runner based around shared folders, to separate Galaxy from the cluster. Also, a script on the cluster head node submits the job to the cluster and writes files back to the directory indicating the job status. Essentially it's a file-based communication mechanism	mixture of custom job runner deploying to external resources and local

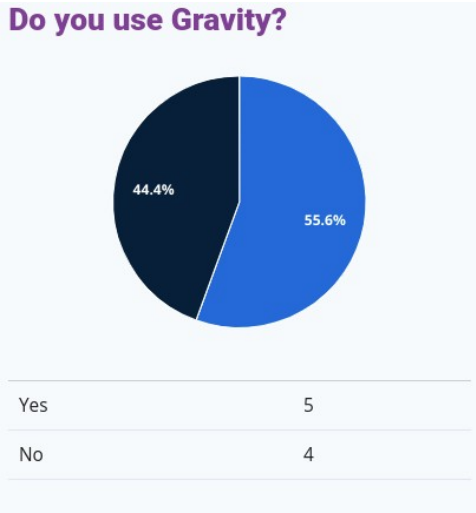
II.D Object store



Backend (Object store)	S3	NAS
Comment (Object store backend)	I mounted a s3 bucket locally that is accessible through the data libraries.	Wish this was a multi-select, NAS + local files :)

Analysis : Usually local &/or NAS, a S3.

II.E Gravity



Do you use Gravity?	Yes	Yes	No
Details/ Comment (Gravity)	Yes but not directly, via Ansible	indirectly via Ansible	Not yet upgraded to a Galaxy version that requires use of Gravity

Analysis :

- Have recent versions made it mandatory /included?
- Used implicitly through Ansible

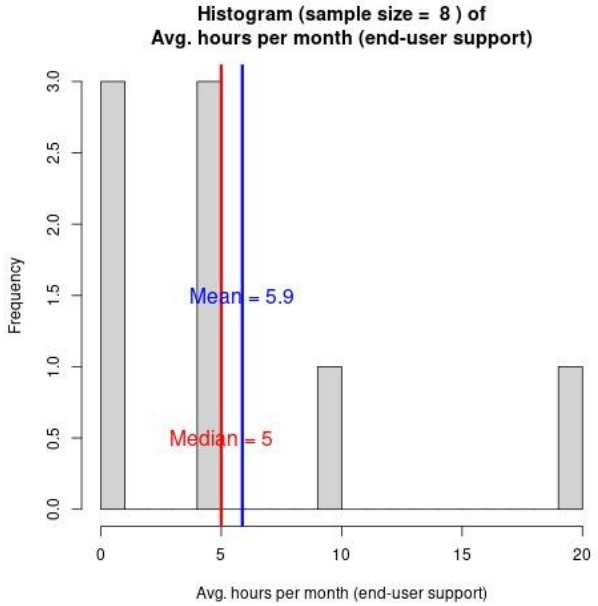
II.F Other comments

Other comments (cluster characteristics)	Hardware: 16 cores (32 CPUs), 384GB RAM, 16TB storage	Mostly single-node machines with some shared storage.	it was 3 nodes, one test, one prod, and one spare that ran other services + shared computational load

Analysis : Small clusters with, sometimes, a shared storage

III Users - various

III.A End-user support

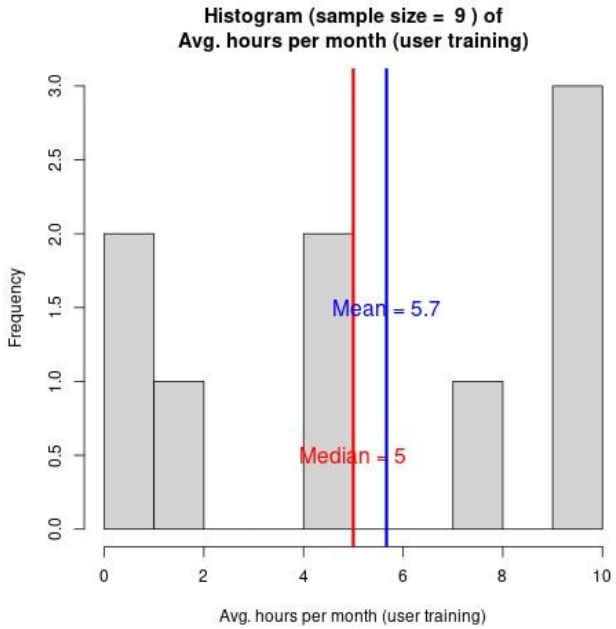


Analysis :

- 1 hour per week average end-user support
- Lots of training (though that's part of the next question)
- Lots of developing &/or debugging users' tools/workflows for them.

On average, hours per month (end-user support/assistance)	20	3	4
Comment (end-user support/assistance)	This is where I spend *most* of my time, users need help learning how to use Galaxy (we can point them at tutorials but sometimes they want personal, live instruction), or need help developing workflows (they don't have the skills and they are experiencing time pressures), or just aren't as familiar with what tools are available or how to work with them yet.	Usually issues with Galaxy server not being available, tool failures, run out of space, jobs not running, requests to install new tools from tool shed.	Mostly fixing tools after tool error reports. Sometimes wrapping of new tools after requests.

III.B User training



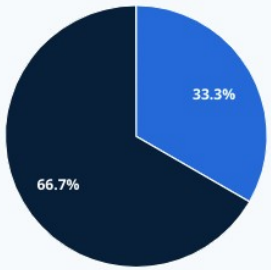
Nb. users	20	20	50	25	10
Avg. hours per month (user training)	5	10	10	0	5
Avg. hours training/user/ month	.25	.5	.2	0	.5
Comment (user training)	I give 2-3 courses (half day) a year using the GTN material.	This is the majority of my time. They do a tutorial, but then they need to do something different or special and I am the only one who can help debug workflow issues they encounter	So much student training required and support questions	No user training offered	A rarely do training classes. Preferred method is direct user assistance whenever help is required.

Analysis :

- 1 hour per week average user training
- Usually makes use of existing Galaxy tutorials, sometimes in-person
- The tutorials aren't always specific enough, requiring some custom Q&A

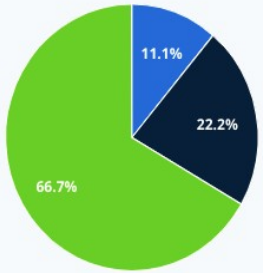
III.C User tool development

Are there any users developing tools themselves locally?



Yes	3
No	6

If so, do they use Planemo?



Yes	1
No	2
No answer	6

Analysis :

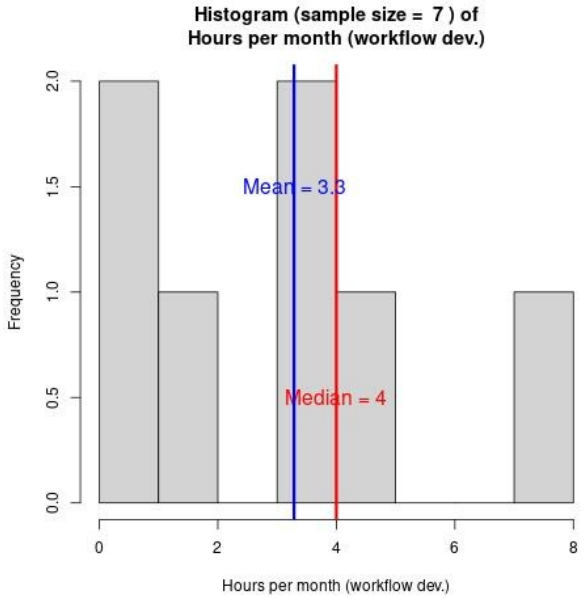
Users developing their own tools is rare, and if they do, they rarely use Planemo, which I found surprising since AFAIK it's supposed to simplify tool dev.

IV Admin – development

IV.A Why this section exists

- I wanted to see how time-consuming and easy/difficulty developing tools/workflows was.
- I couldn't reach end-users directly, so I assumed some admins are also workflow/tool devs themselves
- Furthermore, I was curious how often tailor-made (or rather admin-made) tools/workflows were necessary for universities

IV.B Admin workflow development

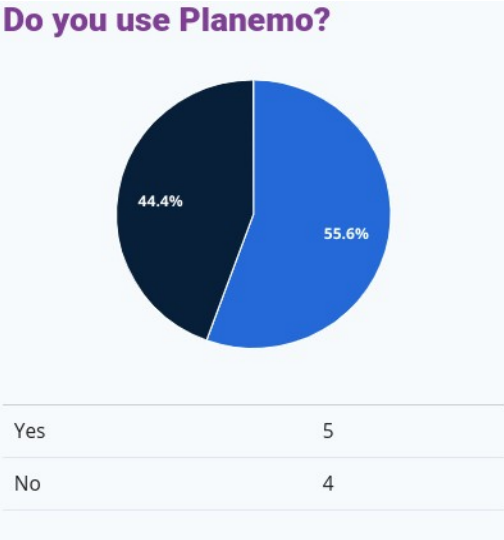


Hours per month (workflow dev.)	2
Comment (workflow dev.)	Very variable it can be 0 for months and then days...

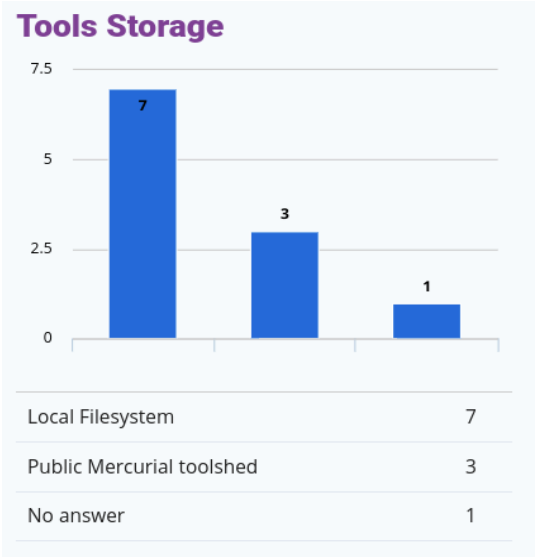
Analysis :

- Average several hours a month, variable

IV.C Admin tool development

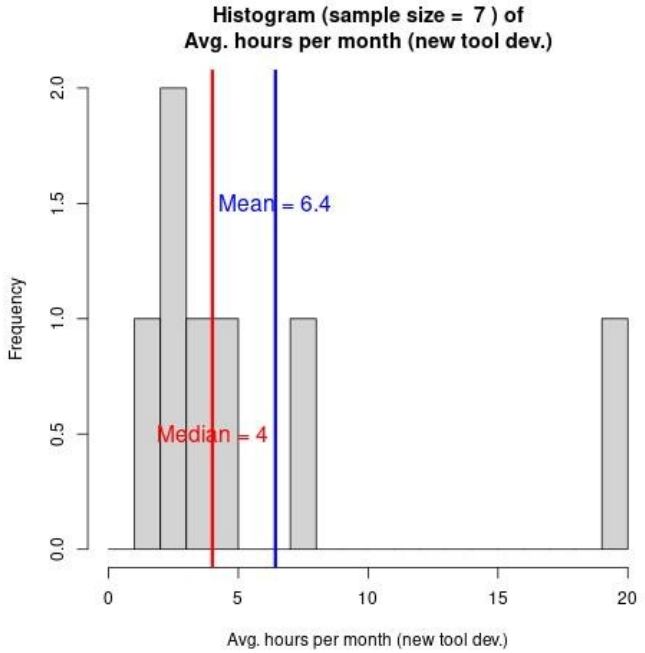


Analysis :
Amongst tool-developing admins, 5/7 use Planemo which means it's a good tool



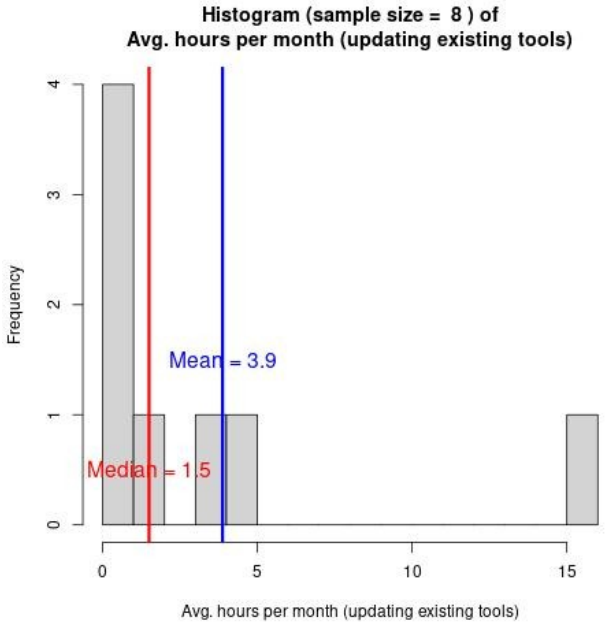
Analysis :

- Most tools are files on the same machine as Galaxy
- Some tools are used from public toolsheds
- No tools come from private toolsheds/repos
- For versioning, some admins load multiple versions locally e.g. tool_v1.1 and tool_v1.2 etc.
- Currently, admins write the tool, test it with Planemo, then publish it locally.
- Galaxy uses Mercurial internally, but one should forget this when developing tools
- All admins use git for their tools (maybe with git hooks to auto-copy a new version to a Galaxy tool folder?)



On average, hours per month (new tool dev.)	20	3
Comment (new tool dev.)	I do not understand the question on Tools Storage? I only use tools via the public toolshed.	Very variable it can be 0 for months and then days...

Analysis : Small (1heur per week avg.) amount of tool dev per month.



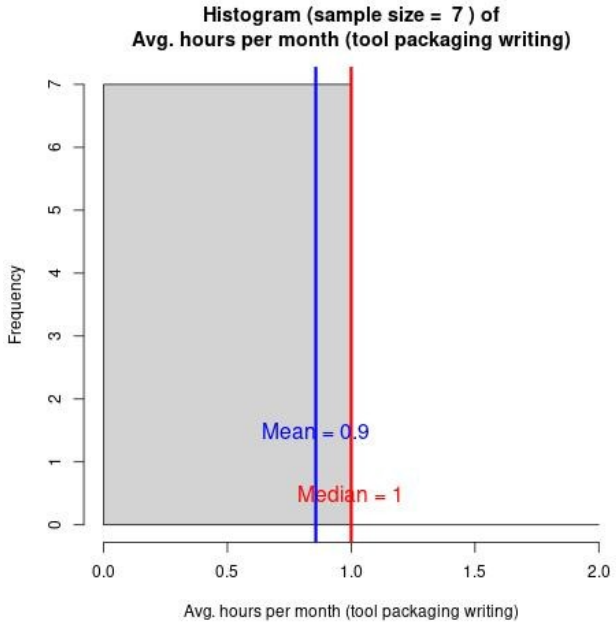
On average, hours per month (updating existing tools)	1	1
Cause/Comment (tool updates)	Could be 0 if I would automate it.	It's more likely to be 12 hours once a year than 1 hour a month every month for a year

Analysis :
I wondered if tools became invalid because of non-backwards-compatible format changes, but that does not seem to be the case or someone would have mentioned it taking a lot of time (excluding the " 16 " value, which was anonymous so I cannot follow-up).

IV.D Tool packaging

Tool Packaging Method	Conda + Container
Comment (tool packaging method)	Conda only historic. I try to get rid of it.

Analysis :
Lots of Conda, a few containers.



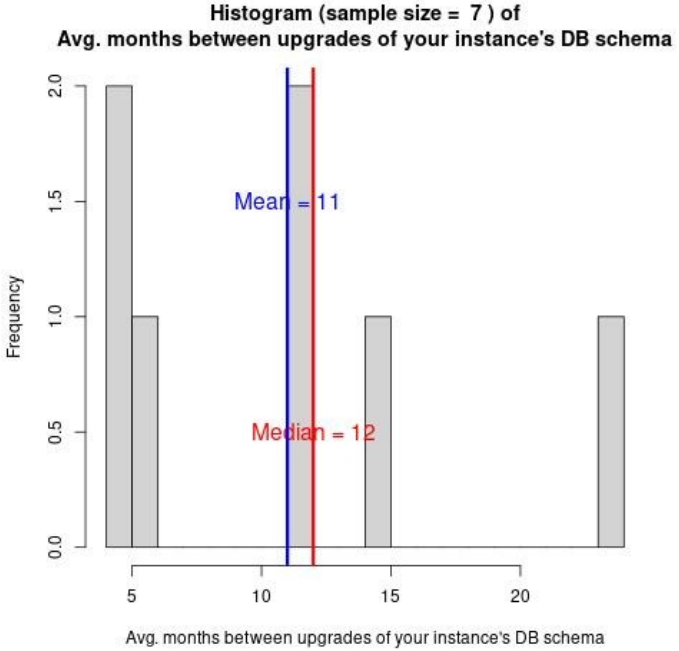
Avg. hours per month (tool packaging writing (Conda recipes/container config/...))	1	1
Comment (tool packaging cost)	We mostly used already packaged software	Should actually be "<1"

Analysis :

Managing Conda envs for tools within Galaxy, doesn't take extra time compared to outside of Galaxy – I initially thought it might.

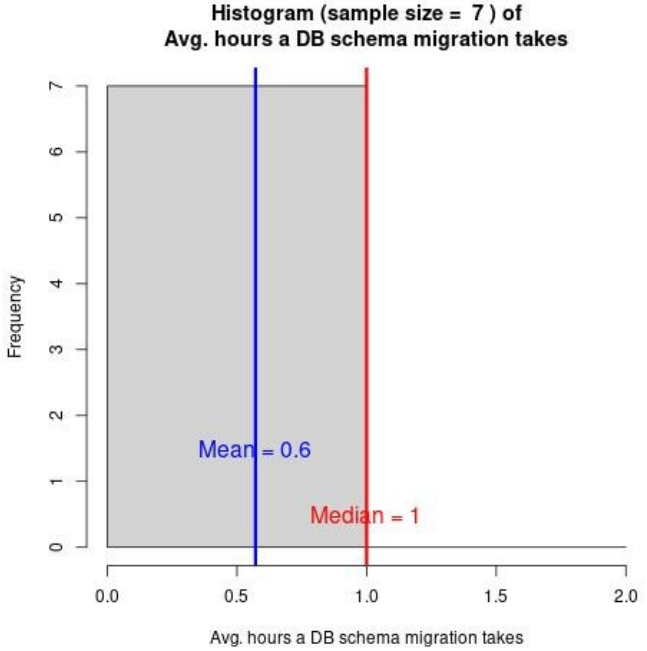
V Admin – Galaxy upgrades

V.A DB schema migration



Analysis

Seem to happens about once a year

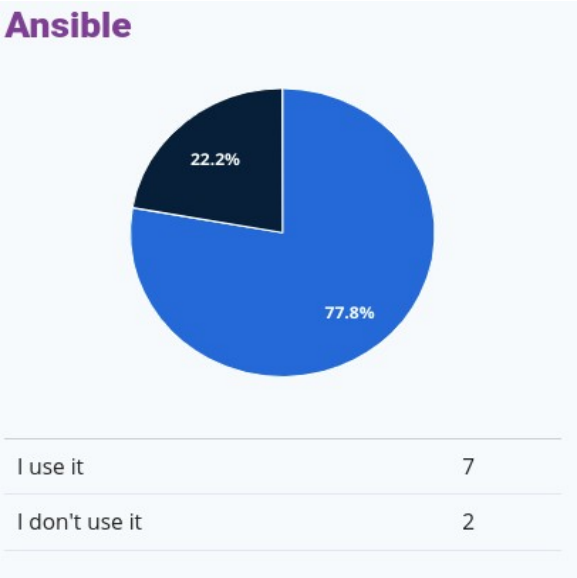


Analysis

Is very quick and transparent (assuming Ansible usage)

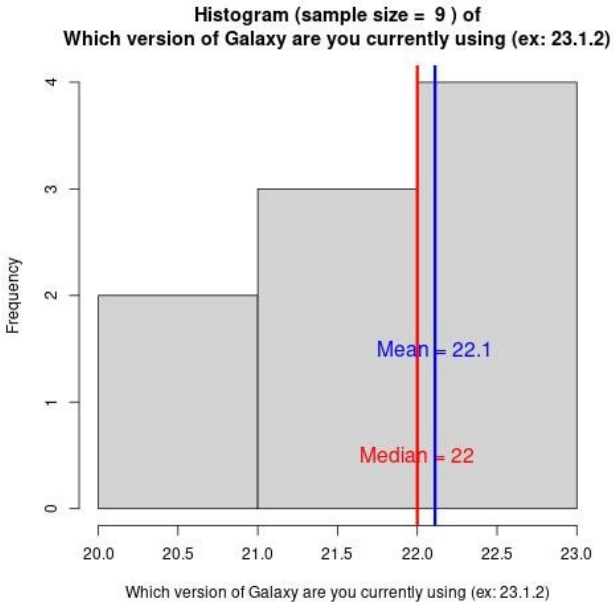
Avg. months between upgrades of your instance's DB schema	15	6	-1	4	4	24	-1
Avg. hours DB schema migration takes	0	0	-1	1	1	1	-1
Comment (DB Migration)	upgrading the database schema has not been a problem (except very long time ago) and happens within minutes (or even seconds)	I usually update to the previous version when the next release is published. DB migration is completely automatized .. don't think that admins need to worry about this.	The database migration is usually linked to the upgrade, I cannot dissociate.	It's very fast thanks to Ansible. Upgrades of schemas only happen during galaxy releases which are like, kinda sorta 3x per year.	it's fine, thanks Ansible	Actually <1 (done as part of Galaxy version upgrades)	can't tell about migration time, sorry. This job is done by a sys admin

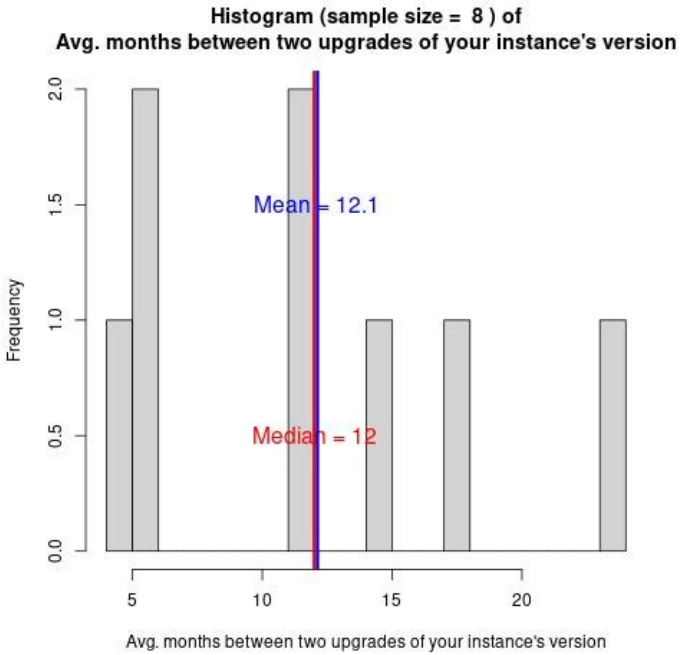
V.B Galaxy version upgrades



Analysis :

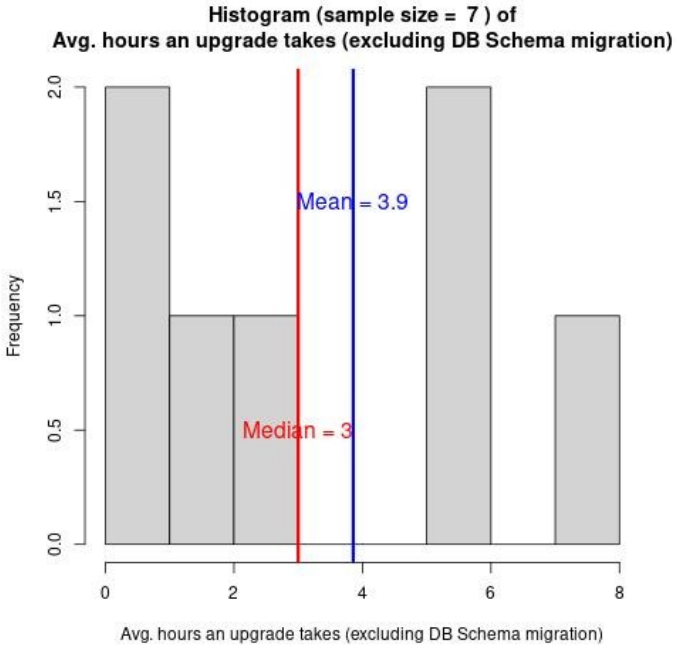
Seems to be a highly-used tool that, according to other parts, greatly simplifies admin.





Analysis

- Half the respondents are able to continue using a version that is 2-3 years old, which means yearly updating isn't mandatory.
- But most update about once a year



Nb. users	20	20	40	50	25	20	3
Avg. hours an upgrade takes	1	1	2	3	6	6	8

Analysis

- Avg 3 hours, big variance, which doesn't seem proportional to the amount of users, but rather other factors
- Apparently some updates are very simple (a few minutes) using Ansible, specifying a new " galaxy_commit_id " in the Ansible playbook. Conversely, those that don't use Ansible incur an extra cost.
- But other updates are long, for example 22.05, going from uWSGI to Gunicorn https://docs.galaxyproject.org/en/master/admin/migrating_to_gunicorn.html
  - Big infra/REST updates seem to happen about every 4 years based on historical-note , but maybe it's stabilised now ? The transition to ASGI only happens once, after all.
- Some choose to update during summer holidays
- Some have multiples instances, potentially unsynchronised too
- Some have test deployments, which adds stability but take more time per update
- Some deploy on VMs, which induces some extra admin time (Vagrant/docker/Kubernetes,...)
- Custom plugins come with the risk of a yearly non-backwards-compatible change
  - This change is (usually, not always) easy to fix, but time-consuming to pinpoint.
  - Idea for easier pinpointing: (ex. for job_runner plugin, v22 to v23, assuming meld installed)
    - *git difftool -t meld v22.05 v23.2.1 --dir-diff lib/galaxy/jobs/runners #specifically, __init.py__ BaseJobRunner*
  - Should be less frequent over time as the plugin APIs mature

Avg. hours upgrade takes	6	1	NA	1	3	6	NA	2	8
What are your motivations for an upgrade ?	- new features - trying to be up-to-date	Get the newest features and bug fixes (minor). Some tools will only run with recent versions. The effort to upgrade is usually really small when not skipping	I would like to have all new features / new datatypes into my instance. This is my motivation to upgrade. From 20.09 to 22.01 everything went smoothly I could upgrade in few minutes, simply by modifying the galaxy_commit_id in the playbook. But as 22.05 has a lot of changes (the change from uWSGI to Gunicorn) I need to test the upgrade on a VM before running it for real. This requires	Bugfixes (minor) mostly, occasionally new features.	we try not to change it during the school year, so need to get new features + bugfixes during the summer	Re motivations: keeping Galaxy current (easier to get help with newer version), some tool versions not available for older Galaxy versions), get security and bug fixes, get new features for users that are available on public instances Re average time for upgrade: this is the time taken to perform an upgrade and includes notifying users etc. Time to prepare for upgrade (e.g. testing on Vagrant & test instance) can be several days or longer depending on changes introduced in the	can't tell about upgrade time as well. This job is done by a sys admin	Be up to date	better to do it voluntarily than be forced at some point as the latter will be more painful; access to new features

		releases..	a lot of time that I could not dedicate yet.		break.	target Galaxy release*** **(continued next page)**			

***(continued) : Causes for this specific Galaxy admin, indicated below

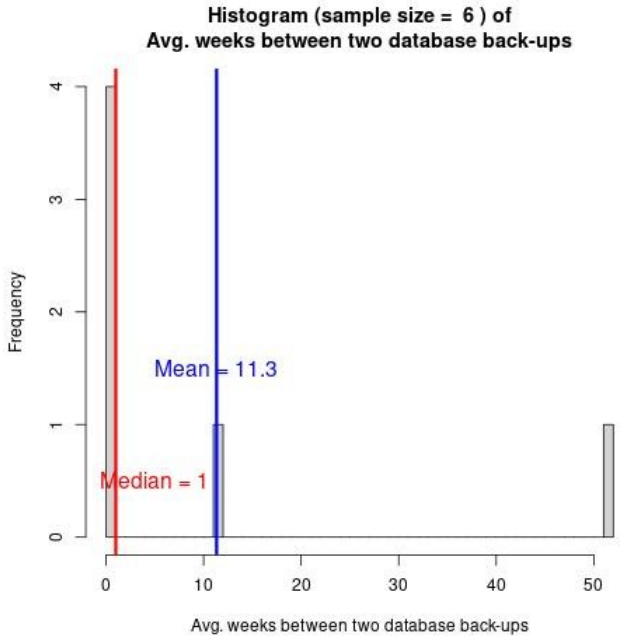
Major	Minor
• Custom job-runner plugin, with Galaxy upgrades (job-runner/job-wrapper base-class method body changes) inducing non-backwards-compatible changes in plugin behaviour, which can take a long time to pinpoint • Not using the official Ansible • uWSGI to Gunicorn (22.05) • Separate test infrastructure/instance, as a precaution before every upgrade	• Multiple instances • Some non-Galaxy-related VM maintenance • Not upgrading often => potentially bigger updates when the time comes • Part time Galaxy admin => less time to dedicate

Analysis

- New functionalities
- Minor bugfixes
- Easier to get assistance if on the most recent version
- Do it sooner rather than wait for a big cumulative update which may contain several issues.
  - Although some issues could " cancel each-other out " (e.g. if a part goes from tech A to B to C, then waiting could actually mean you only change from tech A to C directly).
- Some tools are only compatible with recent versions
  - Although, you can always just use a set older version of a tool ?
- Being compatible with Python versions
  - Although for Galaxy's python there are workarounds https://docs.galaxyproject.org/en/latest/admin/python.html
  - And for tools' python versions, there is always Conda as a workaround

VI Admin – other recurrent tasks

VI.A DB back-up

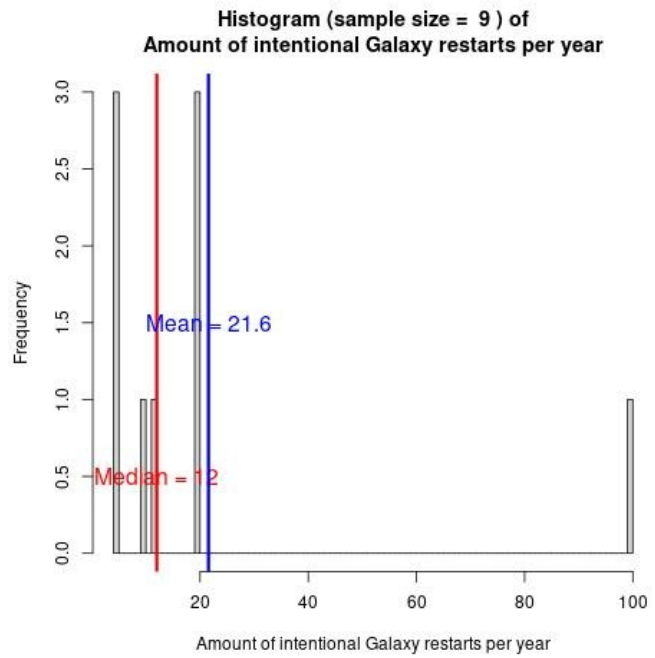


Analysis

- Although not necessarily Galaxy related, I was curious about the best practices
- Avg. is 1 time per week, and instantaneous
- Apparently you can do this through an Ansible playbook (but not the one included with Galaxy, a separate one?)

On average, weeks between two database back-ups	1	-1	1	1	1	12	-1	52
Method/Comment (DB back-ups)	I do daily backups (run as cronjob) using 'pg_dump'	Not sure. Told my IT dept to do backups and never checked :)	The DB back-ups are automatically in the Ansible playbook I am running. The cron job is set to run every week.	Ansible automatically installed database backups are what we use.	Ansible default	Dump SQL to flat file and gzip. Generally only done prior to upgrades	I can't tell exactly but I would expect that it is routinously as the other datasets on file system on daily basis.	rsync (yes, I know I need to do it more often)

VI.B Intentional Galaxy process restarts (for maintenance purposes)

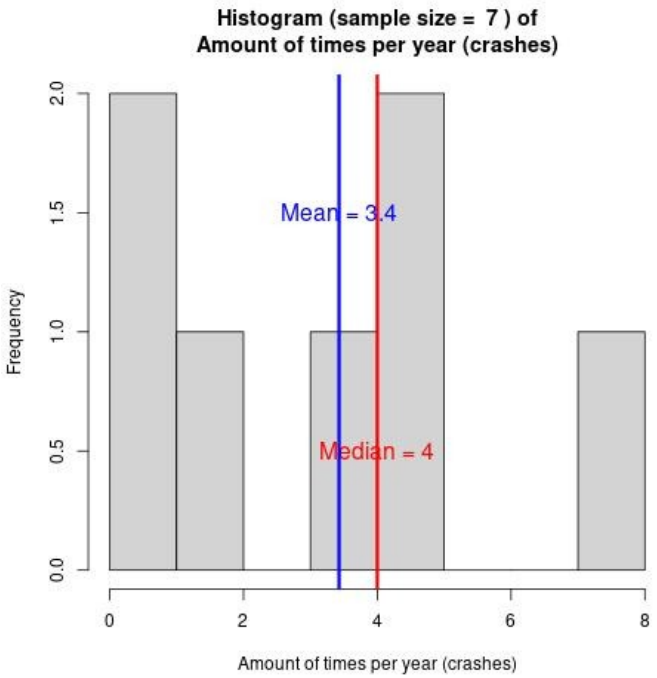


Analysis

- About once a month, although one admin manages to space it out to every 3 months
- The " 100 times per year" answer : the respondent was anonymous, I could not follow-up on it, and since it's different from the others, maybe it can be, for now, ignored.
- " Change in tool version " But I was able to update (or add new) tools without restarting Galaxy.
- "Re-ordering of tools" I was able to do this without restarting Galaxy though ?
- For applying system (Ubuntu, …) updates
- "adding new reference data" I think this refers to having to restart Galaxy to add a new File Source, which I believe is the case.
- " tool that was not picked up " rare, hard-to-reproduce bug
- " config changes ": I noticed some config files could be modified and taken into account, without restarting Galaxy. But others could not. I don't know if there is a clear list of these 2 categories of config files somewhere.
- For the TUS crashes details, see the next section (Crashes)
- Restart procedures:
  - see one of the answers in the next section (Crashes)
  - https://gravity.readthedocs.io/en/latest/subcommands.html#graceful
  - https://gravity.readthedocs.io/en/latest/advanced_usage.html#zero-downtime-restarts

Amount of intentional Galaxy restarts per year	12	4	4	10	20	20	4
Reason/Comment (restarts)	- change in tool version (not coming from toolshed) - adjustment to Galaxy code (e.g.: tool filtering) - re-ordering of tools - adding new reference data (not covered by Data Manager)	We have maintenance windows for our HPC every 3 month which I use for upgrades. I sometimes restart also unplanned which is no problem with a few dozen users (I just announce on short notice).	Server reboot for updates .	mostly for config tweaks or tools that didn't get picked up (rare & hard to reproduce)	config changes, testing out new features, etc.	Generally done in response to addressing issues, don't normally restart otherwise	actually we have far more restarts (e.g. due to constant issues with the TUS service for upload)

VI.C Crashes



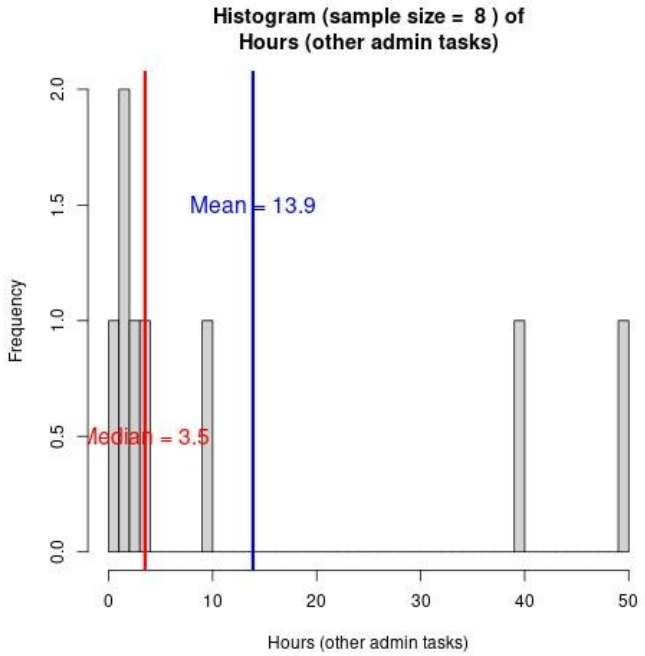
Analysis

- Avg. once every 3-4 months
- Should one have a back-up Galaxy instance just in case ?
- Causes
  - Storage space full, or access rights, or having a probe notify an admin if storage is nearly full
  - This causes secondary problems, such as Postgre getting full, and Galaxy sometimes restarting over and over and filling the remaining space with logs
- Solutions : https://github.com/usegalaxy-au/history-mailer , https://galaxyproject.org/admin/config/performance/purge-histories-and-datasets/
- TUS crashes specifically:
  - Unpredictability and unreproducibility are problems
  - But it's still a rare error (once every 4.5months avg., for 1/9 respondents explicitly)
  - Could automatically upload a file hourly, to (if a problem is detected) alert the admin /auto-restart Galaxy
  - Restarting Galaxy, using that admin's recommended procedure
    - can impact users slightly (unable to submit jobs while last jobs finish running)
    - but isn't excessively inconvenient or long
    - this downtime could be further reduced in the case of asynchronous, non-local job runners

Amount of times per year (crashes)	8	2	4	5
Identified cause/Comment (crashes)	Usually caused by problems (file system) of the HPC	It was always storage issues. Then postgres fails to write and locks. Then Galaxy gets in a restart loop and generates logs filling whatever space was left.	Usually either disks full up or NFS problems	TUS upload service failing, storage issues, users overloading the resources (BLAST search with 500 MB sequence as query) Summarised follow-up of the TUS service failing: • After 22.05 update, did not work at all (solution: Apache → NGINX, & external LDAP → Galaxy-configured auth) • Even after fixing, stops working for all users, at seemingly random times every 3-6 months • File size does not seem to be a factor • The rest of Galaxy still works • Requires restarting Galaxy • Restart procedure: Stop jobs from dispatching first and inform Galaxy users per mail about unplanned maintenance -> check the job queue -> if empty, perform restart; in most cases are still jobs to run -> wait until all jobs have been finished before system restart

				• Restart procedure: no data loss till today (except for the broken uploads)

VI.D Other recurrent admin tasks (not including tool dev, user assistance, upgrades, DB-backups, …)



Analysis

- Some ambiguity monthly/yearly (I should've specified), but seems to be 3h per month avg. (assuming 10 & 40 are annual ;50 is specified as being such ; and the others monthly)
- Cleaning " paused " jobs
  - I assume this refers to errored jobs that are infinitely " paused "
  - Could this be automated ?
- Adjusting user quotas & notifying users, see the " Solutions " sub-section of the " Crashes " section
- Networking with other admins
- Troubleshooting tools (although already included in a previous section)
- Testing new Galaxy versions
- Custom welcome-pages
- Working on workflows (although already included in a previous section)
- " Overview of (non-standard) resources " I assume this refers to adding custom File-sources
  - If so, could the users be allowed to do this themselves ?

Hours (other admin tasks)	10	3	2	4	50	2

Description/Comment (other admin tasks)	-interacting with our sysadmins who are responsible for storage and system software of our server - following up on tool failure - cleaning up 'paused' Galaxy jobs - testing new galaxy versions on our test and development servers	Working on automatization .. networking with other admins	automations	automation again (e.g. spent trying to set up automated "your account is too full" emails., similar to https://github.com/usegalaxy-au/history-mailer)	I think over the year this might be a conservative estimate, tasks include: - Managing user quotas - Adding new users - Adding/updating tools from toolshed - Investigating issues reported by users (tool failures, running out of space)	Not sure whether this is included in user assistance already: setting up/updating Galaxy's institutes custom welcome page with tutorials, overview of plant reference genome resources (all non-standard), setting up specialised workflows for users.

VII Total time, non-development admin tasks

Statistic (median) / **Tasks**	**End-user support (partially counting user training twice)**	**User training**	**DB migration**	**Galaxy upgrade**	**DB back-up**	**Intentional restarts**	**Crashes**	**Other recurrent admin tasks**	**Total**
Frequency (nb. months)	.25[1]	1	12	12	<1	1	3.5	1	NA
Hours taken	1.25[2]	5	<1	3[3]	0	1[4]	2[5]	3	NA
Hours/month (divided by frequency if initially non-monthly)	5	5	0	.25	0	1	.57	3	14.82

[1] Estimated

[2] Calculated proportionally

[3] Lots of variance, for some admins much higher, also depends a lot on which update

[4] Estimated

[5] Estimated