SasView 5 Year Roadmap

The purpose of building and operating large scattering facilities is to provide unique tools to answer new scientific questions with the final presentation of results (usually in the form of a paper) as the output. The biggest obstacle to that output is often the analysis of the acquired data. Data analysis software has been variously viewed as being in the domain of the scientist using the facility, a service to be provided by scattering facilities, or as the individual responsibility of the scientists running the facility beamlines. The result has been a proliferation of packages and libraries, many written and supported by one key person, often not as their primary responsibility [1].

Over the past decade several trends have contributed to exacerbate the analysis bottleneck: 1) As the techniques have matured the user pool has broadened. This combined with an apparent decrease in the overall level of programming taught to scientists, means that fewer users are capable of building their own analysis tools. 2) With the increasing maturity of the field, a large amount of basic modeling is well understood and developed. Even those capable of coding their own should not be wasting their time re-inventing the wheel but focus on new science and perhaps new analysis developments to enable that new science. 3) The quantity of data being produced by instruments and the complexity of the experiments being performed have increased. 4) Finally, as the general software landscape has moved towards increased quality of usability and support, users of scattering facilities, in particular new users, have similar expectations of the software they use to operate the instrument and process and analyse their data.

To enable the production and maintenance of software that meets users expectations, a greater level of resources needs to be applied to the problem. This has been recognised in the neutron scattering community through projects such as DANSE [2], Mantid [3], CCPSAS[4] and projects within the European Union Horizon 2020 programme such as SINE2020[5]. While each facility or scientist may not be able to commit the necessary resources, pooling of expertise between facilities and with the user community into a single project can amplify the effort and provide gains in quality, functionality, longevity and supportability with obvious benefits to both the facilities and the users.

The aim of the SasView project is to provide open source, *collaboratively developed software for the analysis of small angle scattering data*

The collaborative development model is designed to encourage and enable contribution from a range of experts in small angle scattering, computer science and software development. Users of the software are encouraged to participate through bug reports, submitting code and by integrating their own scattering models.

This is a regularly updated document, reviewed at each Code Camp. Document history is maintained and previous iterations can be found in the SasView GitHub repository[6]

Development Model

The SasView project is co-ordinated by a small management team who are also members of the broader development team. The development team currently consists of scientists and software engineers from scattering facilities around the world. Progress is monitored and project direction discussed at bi-weekly video conferences. The work is divided into various work packages that bundle together related tasks and issues. The project aims for a major release each year, with minor bug fix releases as required in between.

The source code for SasView is hosted on GitHub, allowing anyone to fork the code base, add functionality or fix bugs, and easily request that the code be merged into the project. The use of unit tests and code quality metrics is employed to help manage the contribution of code from a varied and dispersed developer base.

As of 2015, few if any of the developers have SasView as a major component of their job assignments. Given the nature of software development this currently means most development occurs during annual code camps. While this has proven to be a highly successful approach, it limits the rate of progress. The Roadmap presented below is intended to give guidance to the development team and to our stakeholders as to the direction we aim to take over the next 5 years. It is an optimistic one predicated on more resources becoming available, either through new developers joining the project or work on SasView becoming a greater part of the day-to-day assignment for the current developers. It also should be used as a source of ideas for projects that could be undertaken with short-term resources or to meet specific requirements of stakeholders if they have the necessary resources available.

Roadmap

Every release should include bug fixes and new models as requests come in. These are assumed with each release and not included specifically below. Likewise general robustness and ease of use issues will be addressed in each release cycle.

Late 2016 - Early 2017 (from code camp V - SNS) - Towards Release 4.1

The main aim for release 4.1 was to address the growing list of requests for smaller feature enhancements and improvements to the interface and workflow rather than any major structural changes. However, due to the availability of SINE2020 resources, code camp work minimised major GUI enhancements in favor of making progress on the new, Qt based, GUI interface which will be the foundation of the 5.0 release. Cleanup of projects left over from the 4.0 release such as finishing the model documentation standardization and review were also a focus of this code camp. While a complete overhaul of saving state is being built into the new GUI project, the ability to save constrained fit pages was targeted for incorporation in this release. Fitting of SESANS data was integrated into the GUI. Finally, with funding for a summer student at ISIS the old corfunc functionality from CCP13 was integrated as a new perspective.

Task Summary:

- Selection of high priority bug tickets were addressed
- Added a new corfunc perspective
- Added file converter to support multifile data
- Integrated SESANS into the SasView GUI

- Work progressed on Save Project when constrained fits are used
- Continued work on GUI refactoring (and the clean separation of UI from computational code.
- Worked on increasing model coverage looking at non-overlap with SASfit models.
- Worked on adding batch functionality to operations on ROI such as box sum and slices.
- Continued model documentation review and formatting
- Added missing documentation and documentation of new functionality
- Worked on improving infrastructure (build systems, 64 bit/Anaconda on all platforms of build machines, trac, licensing, etc)

SasView 4.1 was released in March 2017 incorporating the key pieces of work done at the previous code camp and cleaned up/bug fixed in the interim period.

- https://github.com/SasView/sasview/releases/tag/v4.1.0

Early-Mid 2017 (from code camp VI - Grenoble) - Towards Release 4.2

Subject to the availability of sufficient resources, release 4.2 will again focus on feature enhancements and bug fixes while continuing to work on the GUI refactoring in preparation for a release 5.0. As part of that effort plotting requirements and design work should begin. Met with SASfit team at the code camp to discuss integration of SASFitr models and functionality; improved code optimization in particular adding support for multiple GPUs; began work to streamline the API to make scripting and pipelining more straightforward. Also discussed the need to allow users to chose the integration method as well as discussed building the beta approximation into sasmodels. On the documentation front, besides the continual updating, work began on a tutorial series. Finally, an effort to reach to full unit test coverage began as will as verifying code usage and weeding out redundant code.

Task Summary:

- Continued work on new GUI and code separation
- Finalized approach for handling orientational distributions.
- Enhancing of plot functionality - Collected requirements and began design.
    - Identifed requirements and improvements desired from current design such as tighten space, better fonts, provide both graphical and text entry of controls, put residuals on same panel, provide option to turn auto- plotting of residuals on or off etc)
    - Discussed options for technology to use matplotlib, qtplot, pyqtgraph - currently use matplotlib which is most used but slow at times. ESRF presented their Silex framework.
- Worked on ensuring all computational code has proper unit tests and that they are all being run
- Worked on verifying code redundancy and weed out old/unused/obsolete code
- Began work on tutorial series

- Updated documentation
- Worked on SASfit model integration including discussion with SASFit team
- SasView 4.1.1 and SasView 4.1.2 were released in Mid 2017
    - https://github.com/SasView/sasview/releases/tag/v4.1.1
    - https://github.com/SasView/sasview/releases/tag/v4.1.2
- A hands-on tutorial was developed for use at a Nordic Summer School.

Late 2017 - Mid 2018 (from code camp VII - ESS-DMSC) - Towards releases 4.2 and 5.0

Code Camp VII focussed on exposing developers to the new GUI framework and working on a 4.2 release candidate. Managing parallel development of the 4.x and 5.x code bases began. Release 5.0 will provide a clean API so that future GUI efforts such as web interfaces etc will be simpler to introduce while significantly improving the user experience by providing better integration between the various parts of the GUI that have evolved and grown organically over time in response to requests. This will hopefully also address the mac vs PC usability, save state, reporting, and preferences setting which have been raised as high priority useability issues. Documentation review and creation of new documentation continued.

Task Summary:

- A significant number of bug and small enhancement tickets were addressed
- Continued refactoring new Qt based GUI toward 5.0 release and beginning to bring non SINE2020 coders to bear on these tasks
- Work on J. Appl. Cryst. Paper begun
- Orientational distribution work was completed
- Released sasmodels 0.97 and loaded to PyPI
- Development of beta approximation started
- SasView 4.2 beta released

Late 2018 to mid 2019 (from code camp VIII - ESS) - Release 4.2, Release 5.0 The focus in this period will be on development and release of version 5.0 of SasView. In parallel version 4.2 and possibly 4.3 will be released providing a maintained, stable, release for current users of SasView. This managed transition from the 4.x series to the 5.x series will allow for extensive user testing of the 5.0 version prior to release. We expect to continue maintenance of the final 4.x release beyond the release of 5.0, with an eventual end-of-life for 4.x occurring with the 5.2 release. Full integration of the beta approximation work into 5.0 will be completed, with some limited beta approximation functionality being made available in 4.x. The first SasView community meeting will be held at the SAS 2018 meeting in October 2018 providing SasView users and contributors with an introduction to the new functionality being made available in 5.0 and training on how to get involved in contributing to the SasView project. Building on this meeting a plan for expanding community interactions will be developed. Release 4.2 and 5.0 will support separate plotting of the P(Q) and S(Q) components in a P*S analysis. Work will begin on integration of McSAS into SasView, primarily planning and design work. The

SasModels Marketplace will be updated to better support user needs and fix bugs in the deployment, including bringing the backend up to recent versions.

Task Summary:

- Move focus of all GUI efforts to the new Qt GUI
- Parallel development and release tracks
- Complete beta approximation work
- New, more flexible interaction volumes/radii
- Community meeting at SAS 2018
- Complete SasView paper
- Consolidate and extend training material - both written tutorials and hands-on training material.
- Update model marketplace
- Create plan for developing community interactions.
- Fixes to custom model editor to support polydispersity
- Incorporation of models from SASFit[7] and Scatter[8] (Förster - crystalline materials models primarily)
- Project infrastructure cleanup - ticket review/cull given 5.0 release and possible move to GitHub issues.
- Release 5.0 alpha (late 2018), 5.0 beta (early 2019), 5.0 (mid 2019)
- Release 4.2

Late 2019 to mid 2020 (from code camp VIII - ESS) - Release 4.3, Release 5.1

Subject to the availability of sufficient resources, release 5.1. Work will start on refactoring fitting to allow, for example custom re-parameterization of models (e.g. replace SLD with fraction of solvent in layer), using an input array for P or S in a P*S model, fitting oriented model to 1D cut etc. Work will begin on refactoring the simultaneous/constrained fitting workflow interface and on custom workflows identified as highest priority and having a well developed design. User documentation/tutorials will be reviewed, an advanced "how to fit my data" tutorial will be started, and an architecture manual begun. McSAS will be integrated into SasView, giving users an approach to obtaining particle size distributions. Work on providing the generic O-Z solver tool that is present in SASFit will begin. Work on support for multi-GPU and multi-CPU computation, which may involve refactoring away from OpenCL as support for this standard is waning. Incorporation of the PRISM[9] (polymer reference interaction site model) code into SasView.

Task Summary (Subject to the availability of sufficient resources):

- Begin model fitting refactoring work to allow custom re-parameterization of models, allow reading in an array representing either PQ or SQ for P*S fits, fitting oriented model to 1D cuts including revisiting orientation definitions etc.
- Complete architecture manual
- Begin work on refactoring constrained/simultaneous fits.
- Begin work on adding custom workflows identified as highest priority
- Work to update tutorials to support 5.x
- Begin work on advanced model fitting tutorial

- Usual bug fixes and other minor improvements as time and interest permit
- Integration of McSAS
- Begin work on generic O-Z solver
- Inclusion of PRISM[9] functionality
- Begin work to refactor/improve generic scattering calculator
- Improvements to custom model editors including features from compare.py
- Support for multi-GPU, multi-CPU and CPU/GPU computation

Late 2020 - Mid 2021 (from Code Camp IX)- Release 5.2

Subject to the availability of sufficient resources, release 5.2 will provide new fitting functionality such as custom re-parameterization of models, allow reading in an array representing either PQ or SQ for P*S, fitting oriented model to 1D cut etc. The refactored workflow interfaces for constrained/simultaneous fits and batch fitting and plotting module will be deployed in this release. Work will continue on an advanced data fitting with SasView tutorial. Work on new workflow/interfaces for contrast variation for example and new magnetic scattering workflows will begin. These workflows are not expected to be in the release however. Generic O-Z solver will be available in this release.

Task Summary (Subject to the availability of sufficient resources):

- Finish fitting refactoring work to allow custom re-parameterization of models, allow reading in an array representing either PQ or SQ for P*S fits, fitting oriented model to 1D cut etc.
- Refactor simultaneous/constrained workflow interface
- Continue development of advanced fitting tutorial
- Start new workflow/interfaces
- Usual bug fixes and other minor improvements as time and interest permit
- Finish work on O-Z solver.
- Complete work on refactor/improve generic scattering calculator.

Late 2021 - Mid 2022 (from Code Camp X) - Release 5.3

Subject to the availability of sufficient resources, release 5.3 will again try place an emphasis on addressing requests for smaller feature enhancements and improvements to the interface and workflow. It will also include some new workflow interfaces. Use cases and design development will commence on a web interface (possibly including smartphone app capabilities). Advanced fitting tutorial and other unfinished documentation projects will be completed. Review all documentation and prioritize needs for next release. Work on integration of SasView into realtime analysis workflows at beamlines will begin.

Task Summary (Subject to the availability of sufficient resources):

- Finish advanced model fitting tutorial
- Include more workflow/interfaces
- Begin use case and design on Web interface (with possible smartphone app feature) * initial version can have minimal features but would be useful for demos?

- Finish outstanding documentation projects
- Prioritize new documentation tasks
- Usual bug fixes and other minor improvements as time and interest permit
- Headless operation/realtime analysis workflows started

Late 2022 - Mid 2023 (from code camp XI) - Release 5.4 / Release 6.0 alpha/beta

Subject to the availability of sufficient resources, release 5.4 will start providing intelligent feedback on unreasonable choices. Transition will start to the 6.x release series as support for ASAXS is added and other SAXS specific tools/workflows are added as needed. Web UI work will continue but is not expected to be ready for this release. Finally work will begin to allow computational code to run on a cluster and an intelligent launcher/scheduler design started for the GUI frontend which will make the use of the a cluster backend transparent to the user. Include documentation tasks prioritized in previous round. Integration of SasView into a realtime analysis workflow on beamlines will be completed.

Task Summary (Subject to the availability of sufficient resources):

- Start including intelligent limits/help (possibly include switch between enforcement and warning only) and explore the use of wizards in some cases
- Continue work on web UI and smartphone app
- ASAXS support added
- Add extra SAXS specific needs as appropriate
- Enable computational code to run on clusters and refactor GUI to add an intelligent launcher/scheduler that makes the use of a cluster back end transparent to the user
- Documentation tasks as determined during previous code camp or fortnightly meetings
- Headless operation and realtime analysis workflows completed.
- Usual bug fixes and other minor improvements as time and interest permit

Late 2023 - Mid 2024 (after code camp XII) - Release 6.0

Subject to the availability of sufficient resources, release 6.0 will allow running compute intensive portions of SasView computation on a cluster back end with a transparent access from the user GUI. It will also allow deployment as a webservice with a web based front end which will have limited functionality in this first instance. ASAXS and other SAXS specific workflows will be included. Work on a smartphone app interface to the webservice will continue but lilkely will not be ready for this release. The use of wizards and intelligent user guidance will be expanded and new workflows/interfaces may be added as appropriate. Include documentation tasks prioritized in previous round.

Task Summary (Subject to the availability of sufficient resources):

- UI refactoring complete
- Deploy computational code on clusters
- GUI includes intelligent launcher/scheduler that makes the use of a cluster back end transparent to the user

- Deploy Web application
- Expand use of wizards and intelligent user guidance
- Add new workflow interfaces as appropriate
- Continue work on smartphone UI
- Documentation tasks as determined during previous code camp or fortnightly meetings
- Usual bug fixes and other minor improvements as time and interest permit

Late 2024 - Mid 2025 (after code camp XIII - TBA) - Release 6.x

Subject to the availability of sufficient resources, release 6.x will again try place an emphasis on addressing requests for smaller feature enhancements and improvements to the interface and workflow. It will also continue to expand on intelligent guidance and include more functionality on web app and see the deployment of a smartphone app. Include documentation tasks prioritized in previous round.

- Continue to expand use of wizards and intelligent user guidance
- Deploy smartphone app
- Expanded functionality of web app
- Documentation tasks as determined during previous code camp or fortnightly meetings
- Usual bug fixes and other minor improvements as time and interest permit

Revision History

- 2015–11–24 : First release
- 2016–10–11 : Updated after Code Camp V discussions
- 2018–09–07 : Updated after Code Camp VI & VII discussions

---

1. http://smallangle.org/content/Software ↩
2. http://www.danse.us/ ↩
3. http://www.mantidproject.org ↩
4. http://www.ccpsas.org ↩
5. http://sine2020.eu ↩
6. http://github.com/SasView/documents/Roadmap ↩
7. https://github.com/SASfit/SASfit ↩
8. http://www.esrf.eu/UsersAndScience/Experiments/CRG/BM26/SaxsWaxs/DataAnalysis/Scatter ↩
9. https://github.com/usnistgov/pyPRISM ↩