systemPipeShiny: An Interactive Framework for Workflow Management and Visualization

Le Zhang, Daniela Cassol, Ponmathi Ramasamy, Jianhai Zhang, Gordon Mosher & Thomas Girke

Institute for Integrative Genome Biology, University of California, Riverside, California 92521

Introduction

systemPipeShiny (SPS) extends the widely used systemPipeR (SPR) workflow environment with a versatile graphical user interface provided by a Shiny App. This allows non-R users, such as experimentalists, to run many systemPipeR's workflow designs, control, and visualization functionalities interactively without requiring knowledge of R. Most importantly, SPS has been designed as a general purpose framework for interacting with other R packages in an intuitive manner. Like most Shiny Apps, SPS can be used on both local computers as well as centralized server-based deployments that can be accessed remotely as a public web service for using SPR's functionalities with community and/or private data. The framework can integrate many core packages from the R/Bioconductor ecosystem. Examples of SPS' current functionalities include: (a) interactive creation of experimental designs and metadata using an easy to use tabular editor or file uploader; (b) visualization of workflow topologies combined with auto-generation of R Markdown preview for interactively designed workflows; (d) access to a wide range of data processing routines; (e) and an extendable set of visualization functionalities. Complex visual results can be managed on a 'Canvas Workbench' allowing users to organize and to compare plots in an efficient manner combined with a session snapshot feature to continue work at a later time. The present suite of pre-configured visualization examples include different methods to plot a count table: PCA, t-SNE and plots of other clustering methods; volcano plots, data mining plots including heatmaps, and more. The modular design of SPR makes it easy to design custom functions without any knowledge of Shiny, as well as extending the environment in the future with contributions from the community.

SPS Structure

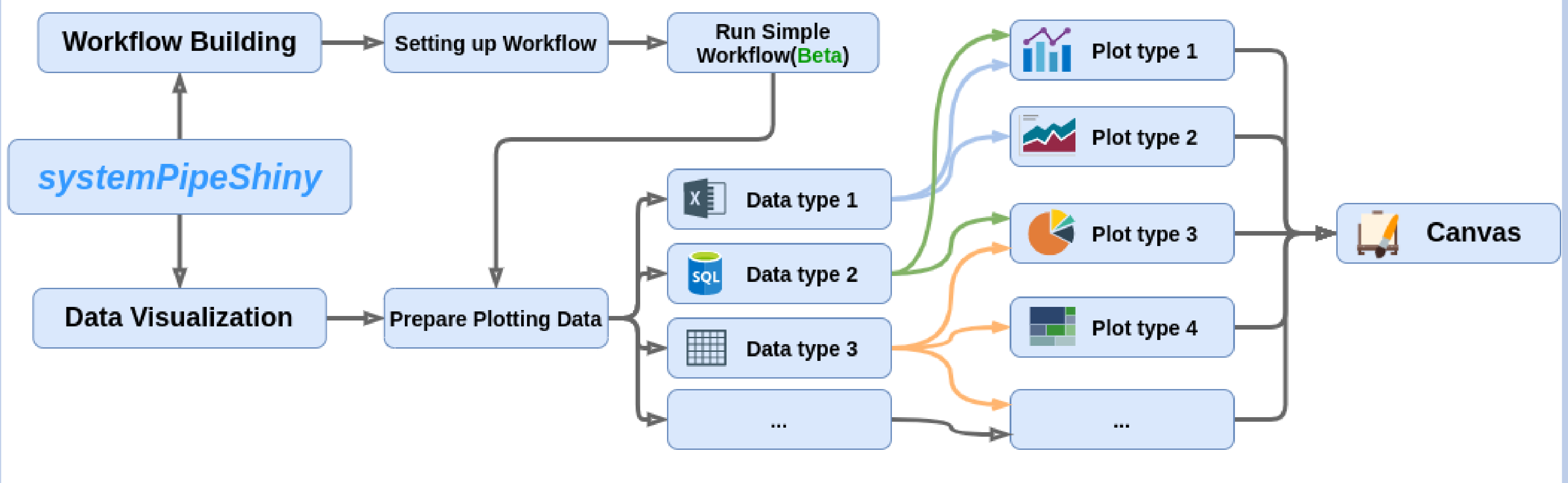


Figure 1: **Design of SPS.** The framework provides an interactive web interface for workflow management and data visualization.

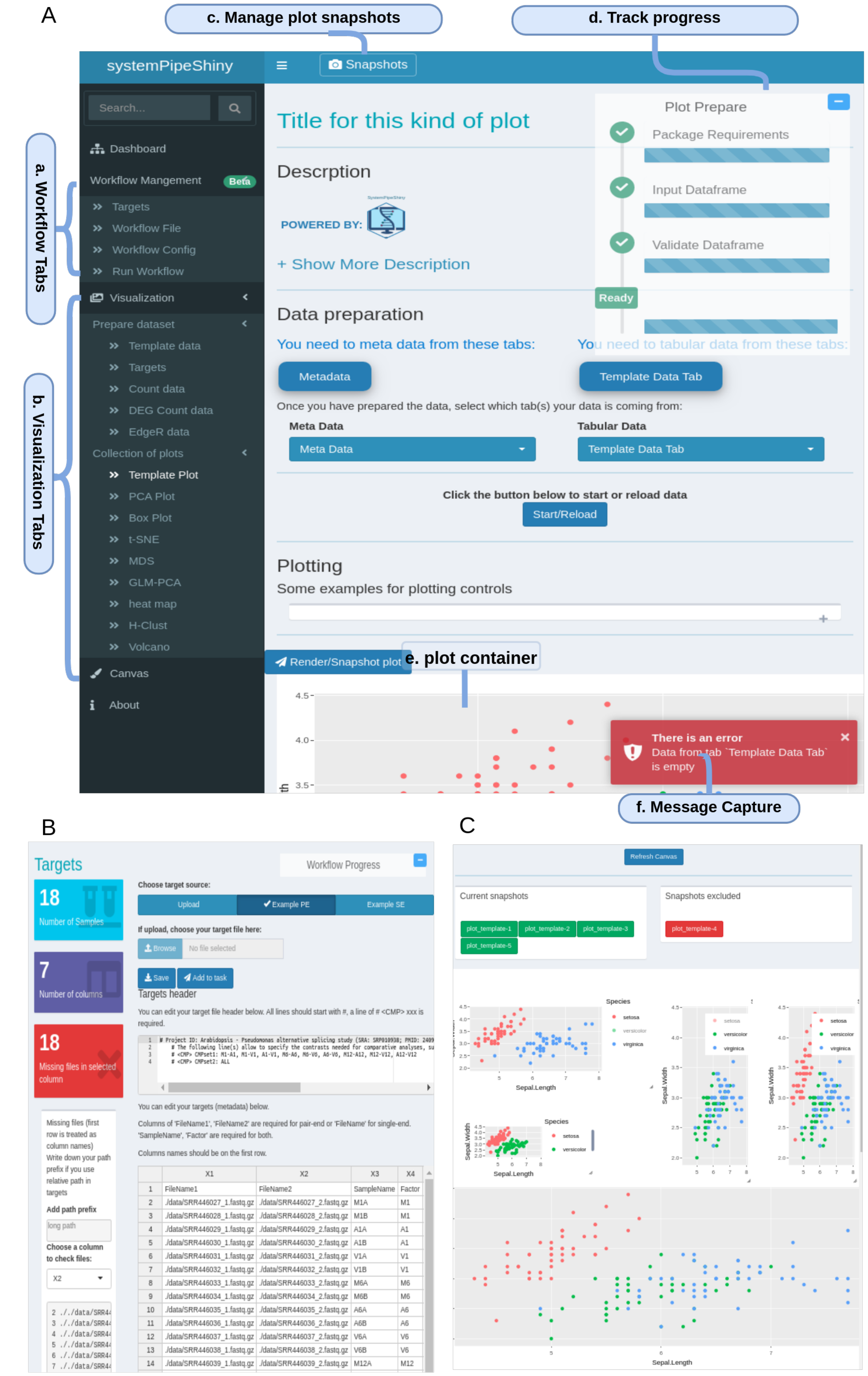
When SPS launches, one can switch between the workflow and visualization utilities (Fig 1). Starting with the workflow module, one can design *systemPipeR* (SPR) workflows that are compatible with other CWL-based workflow environments. In the upcoming release, SPR workflows can be run directly from SPS. Current workflow functionalities allow users to: **(a)** interactively define experimental designs and provide associated metadata using an easy to use tabular editor and/or file uploader; **(b)** visualize workflow topologies combined with auto-generation of R Markdown previews for interactively designed workflows; and **(c)** manipulate yaml style configuration files.

To use the visualization module, users need to select a "data tab" and prepare the input data according to the provided instructions. Next, one can activate different "plotting tabs" depending on a chosen data pre-processing method. On the plotting tab, users can optimize plotting parameters. The present suite of pre-configured visualization tools include methods to interpret read count tables using PCA, MDS, t-SNE or related methods; as well as volcano plots, heatmaps, and other commonly used visualization methods for clustering results. Finally, users can take snapshots of plots and view them on the "Canvas Workbench". This functionality enables users to compare the same plot with different plotting options or combine plots from different tabs to complex figures. In addition, users can save their results and continue to work at a later time.

Important Features

- **User friendly**: SPS is easy to use for scientists from different backgrounds. The GUI operations require no programming experience. Detailed design templates are provided for advanced users allowing to add new features to SPS.
- **Dual-end logging**: Messages, warnings and errors from R functions are automatically captured and logged on both the server and client ends. The corresponding information can be displayed in a message box.
- **Progress tracking**: A progress tracking panel is provided on each tab to notify users about run status.
- **App options**: a robust exception handling system has been implemented (similar to Shiny options), that provides error solutions to to users, e.g. invalid parameter settings.
- **Modular isolation**: SPS is built on Shiny modules, which provides local scope isolation between each tab. Objects on one tab do not conflict with other tabs. To enable cross-tab communication, SPS also supports global scope interactions.
- **Canvas**: Under this workbench users can take snapshots of different plots, and combine or resize them. This feature is useful for generating complex scientific summary graphics.

SPS User Interface



You need to meta data from these tabs:

You need to tabular data from these tabs:

Once you have prepared the data, select which tab(s) your data is coming from:

Click the button below to start or reload data

Some examples for plotting controls

There is an error
Data from tab `Template Data Tab` is empty

Choose target source:

If upload, choose your target file here:

You can edit your target file header below. All lines should start with #, a line of # <CMP> xxx is required.

```
# Project ID: Arabidopsis - Pseudomonas alternative splicing study (SRA: SRP010938; PMID: 2409
    # The following line(s) allow to specify the contrasts needed for comparative analyses, su
    # <CMP> CMPset1: M1-A1, M1-V1, A1-V1, M6-A6, M6-V6, A6-V6, M12-A12, M12-V12, A12-V12
    # <CMP> CMPset2: ALL
```

You can edit your targets (metadata) below.

Columns of 'FileName1', 'FileName2' are required for pair-end or 'FileName' for single-end. 'SampleName', 'Factor' are required for both.

Columns names should be on the first row.

Missing files (first row is treated as column names) Write down your path prefix if you use relative path in targets

	X1	X2	X3	X4
1	FileName1	FileName2	SampleName	Factor
2	./data/SRR446027_1.fastq.gz	./data/SRR446027_2.fastq.gz	M1A	M1
3	./data/SRR446028_1.fastq.gz	./data/SRR446028_2.fastq.gz	M1B	M1
4	./data/SRR446029_1.fastq.gz	./data/SRR446029_2.fastq.gz	A1A	A1
5	./data/SRR446030_1.fastq.gz	./data/SRR446030_2.fastq.gz	A1B	A1
6	./data/SRR446031_1.fastq.gz	./data/SRR446031_2.fastq.gz	V1A	V1
7	./data/SRR446032_1.fastq.gz	./data/SRR446032_2.fastq.gz	V1B	V1
8	./data/SRR446033_1.fastq.gz	./data/SRR446033_2.fastq.gz	M6A	M6
9	./data/SRR446034_1.fastq.gz	./data/SRR446034_2.fastq.gz	M6B	M6
10	./data/SRR446035_1.fastq.gz	./data/SRR446035_2.fastq.gz	A6A	A6
11	./data/SRR446036_1.fastq.gz	./data/SRR446036_2.fastq.gz	A6B	A6
12	./data/SRR446037_1.fastq.gz	./data/SRR446037_2.fastq.gz	V6A	V6
13	./data/SRR446038_1.fastq.gz	./data/SRR446038_2.fastq.gz	V6B	V6
14	./data/SRR446039_1.fastq.gz	./data/SRR446039_2.fastq.gz	M12A	M12

```
2 ././data/SRR44
3 ././data/SRR44
4 ././data/SRR44
5 ././data/SRR44
6 ././data/SRR44
7 ././data/SRR44
```

Figure 2: **GUI snapshot of SPS. A. (a-f)** Dashboard view of SPS with plotting tab shown as an example. **B.** Sample instance of workflow tabs, where users can interactively prepare metadata. **C.** Canvas Workbench with drag, resize and reposition options for comparing plots.

Extendable SPS Tabs

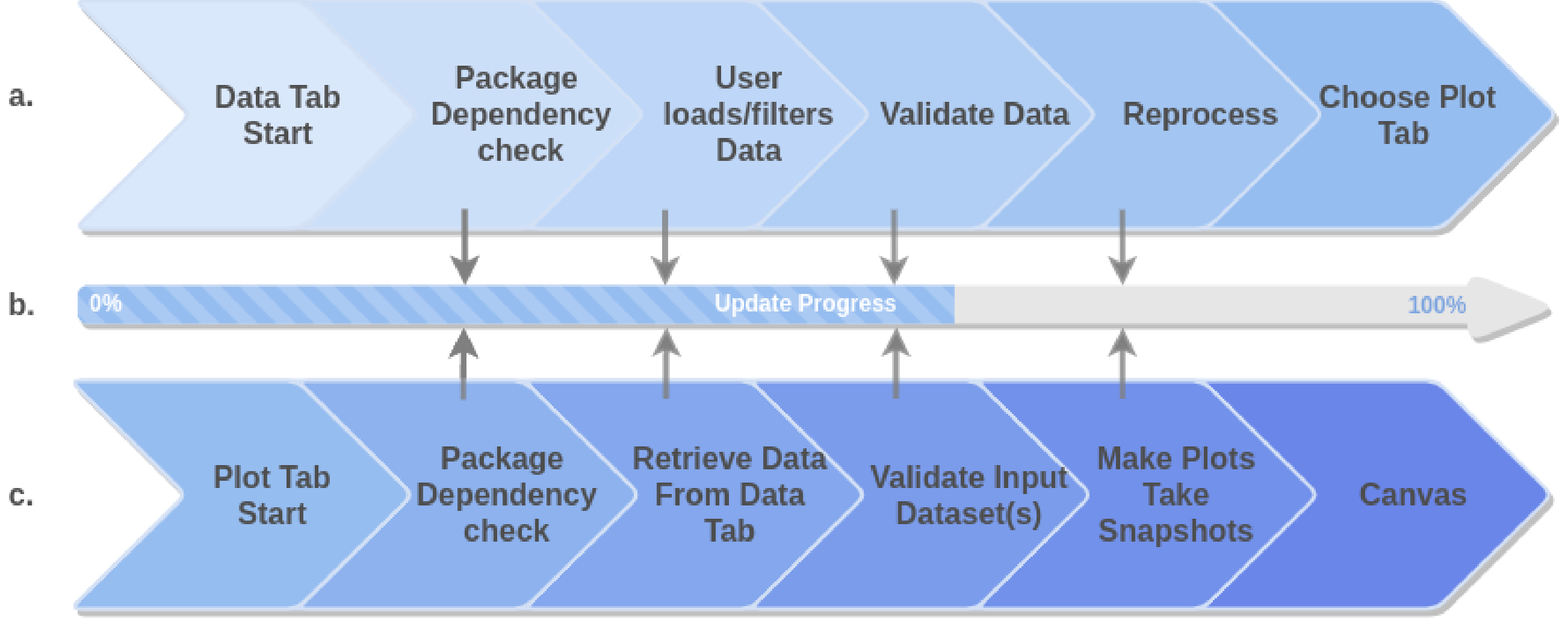


Figure 3: **Server logic of visualization tabs. a.** Modular design for preparing data on a "data tab". **c.** Similarly, data can be validated prior to plotting. **b.** Certain key steps in data/plot tabs will trigger a progress update. This is reflected on the progress tracker panel.

Templates are available to generate new visualization tabs. The design concepts of these tabs is illustrated in Figure 3. SPS has built-in functions to simplify the creation of data/plot tabs and register them in the framework. The open design supports modifications to the general tab structure.

Availability

- systemPipeShiny package is freely available on GitHub.
- A life server-based instance of SPS can be found here.

Acknowledgement

Funding

This work was supported by grants from the National Science Foundation (PGRP-1546879, ABI-1661152), the National Institutes of Health (U24AG051129, U19AG023122) and a fellowship from the National Aeronautics and Space Administration.

References

Backman TWH, Girke T (2015) systemPipeR: NGS workflow and report generation environment. *BMC Bioinformatics*, 17(1)

RStudio, Inc (2020) Shiny: Easy web applications in R

le.zhang001@email.ucr.edu