Data Pipelines: Using a Model-View-Controller Framework in R

Brandon Rose and Natalie Goulett

12/2024

Contents

Chapter 1

What is a Data Pipeline?

A data pipeline is an intentionally designed process that transforms data from its input(s) to its output(s). A proper data pipeline has complexity that matches its aims. For any data project it's critical to have a **conceptual framework** and to **use the right tools**. This book will use the general concept and language of a Model-View-Controller framework to contextualize examples of effective data pipelines. What tools are ultimately chosen (intentionally or not) are a combination of cost, dependencies, security, familiarity, reusability, maintainability, scalabiltiy, and more.

1.1 What is Model-View-Controller?

1.2 When is a Data Pipeline Needed?

A project will increasingly benefit from a data pipeline the more longitudinal, important, sensitive it is. For example, a one-time personal project does not need a data pipeline. A clinical trial may have several interconnected data pipelines.

1.3 Applications as a data pipeline

Most common software applications can be thought of as a *controlled* data pipeline. Think of your bank website, a social media website, and the Epic Electronic Medical Record (EMR) as examples. Even if they don't necessarily use the MVC framework, they have different users, secure data model(s) on a backend server, a designed view on the frontend browser, and a series of controllers (computer programming and logic) that dictate the interactions between the view and the data model.

One of the main highlights of this book is the versatility of REDCap, which can be thought of as both a model (data source) and an MVC application.

1.4 Authors

- Brandon Rose, MD, MPH
- Natalie Goulett

Chapter 2

Users

In any data pipeline you should take stock of various users you have. You can't do it all on your own and if you can you shouldn't! The best work is produced by diverse teams with shared goals.

2.1 Four Categories of Data Pipeline Users

The value of a team member has nothing to do with the categories below. However, in the context of data pipeline design you have four categories of users: End User, Basic User, Intermediate User, and Advanced User.

The End User

The end user may struggle to interact with the software directly. They may have limited technical proficiency or lack interest in learning the software at all. Despite this, they may be primary consumer of the outputs, such as visualizations, dashboards, reports, or data summaries. Their primary role is to use these outputs for decision-making or other purposes.

The Basic User

The basic user can install and run the software with minimal guidance. They may not fully understand the inner workings of the tool but can follow instructions, execute straightforward tasks, and produce simple outputs if given clear directions. Troubleshooting or resolving errors is often challenging, and they rely on others for deeper technical help.

The Intermediate User

The intermediate user is comfortable navigating the software independently. They can create scripts, automate simple workflows, and troubleshoot common errors. They've likely used the software to produce custom outputs (e.g., charts, reports, or transformations). They have some foundational understanding of how the software works but may not engage deeply with advanced topics like performance optimization, APIs, or integrations.

The Advanced User

The advanced user is highly proficient and can use the software to tackle complex problems. They understand how to customize, extend, and integrate it with other systems or tools. This includes creating reusable solutions like functions, modules, or pipelines. They may leverage APIs, manage databases, and optimize workflows, often serving as the go-to expert for solving challenging technical issues or innovating within their domain.

2.2 Types of R Users as an Example

It's very important to gauge where you fall on the spectrum of R users before and during a project. You want to recognize your limitations so that more experienced users can accelerate your learning. Regardless of your current ability, the best way forward is with active learning with real projects.

The Never (End) R User

The never R user may struggle to download R and/or R Studio. They may have difficulty navigating software and file systems in general. Generally speaking they have no interest in even knowing what R is. However, this may be the ***typical*** **end-user** and could be the primary audience of R output such as shiny apps, markdown reports, excel sheets etc.

The Basic R User

The basic R user is capable of installing R and RStudio on their computer. They may be very hesitant working through any errors or understanding "how it all works". They likely prefer point-and-click software that they are more familiar with. However, they know how to use software and can run some lines of code if the process is straight forward enough.

The Intermediate R User

An intermediate user knows how to create an R project and even has some scripts they have written themselves somewhere on their computer. The haven't gone down many rabbit holes but they know to make some ggplots and are capable of working through most common errors.

The Advanced R User

An advanced R user understands how R packages work. They can write reusable functions and do complex tasks that may include API calls, data transformations, working with SQL and more.

2.3 How does this affect your data pipeline?

You should tailor how you design and how you communicate your data pipeline based on the different users. If your team is all end users, then you should rely on commercial and free software to accomplish your goals. If you have many different types of users then you may have the opportunity to create more custom applications.

Chapter 3

Models

about models as a concept.

3.1 What makes a good model?

3.2 Love the data you have

Data is everywhere but you only have access to some of it. Getting access is often time consuming. Once you have access to data you should consistently assess its strengths, weaknesses, opportunities, and threats (SWOT). In the name of deliverables, you have to have endpoints. Your productivity will be judged by the consistent quality and quantity of work.

3.3 data sources

3.4 EMR data (Relational)

3.5 Transactional vs Analytical Data

OLTP vs OLAP

3.6 Data Structure

3.7 Metadata

3.7.1 Coded data

3.8 REDCap

3.9 Other Examples

3.9.1 Qualtrics

3.9.2 Google forms

Chapter 4

Views

4.1 plots

4.1.1 static plots

4.1.2 interactive plots

4.2 Why is R our choice?

Chapter 5

Controllers

5.1 Zero Access

Proprietary software with no interest in you knowing their logic.

5.2 Pseudo Controllers

You can get some reporting

5.3 The Coding Languages!

5.3.1 R vs Python

5.3.2 Why is R our choice?

5.4 Application Program Interfaces (APIs)

Chapter 6

Categorizing Software by MVC

6.1 REDCap

Category: MVC (End and Basic Users); Model (Intermediate and Advanced Users) Model: A- Security: A Log: A+ Structure: B Metadata: A+ Controller: A+ Customization: C API: A+ View: C+ Customization: B User Rights: A+ Public-Facing option: Yes Cost: Free for non-profit institutions who will host the server

6.2 SPSS

6.3 Tableu

6.4 SASS

6.5 STATA

6.6 Microsoft Excel

When to use: viewing and storing data, one time jobs such as designing your REDCap metadata Avoid for: data analysis

6.7 qualtrics

6.8 Power BI

6.9 chat gpt

Chapter 7

Example Pipelines

7.1 Pan-Sarcoma Database

7.2 Quienticential Chart Review Project

7.3 Multi-instititional

7.4 Report Back Letters

7.5 Admissions Data

7.6 Running Data

Chapter 8

Conclusion

8.1 choosing your tools

8.2 putting it all together

8.3 Resources for more learning

Chapter 9

About the Authors

9.1 Brandon Rose, MD, MPH

Dr. Rose earned his MD/MPH at the University of Miami Miller School of Medicine. He is currently a third-year internal medicine resident at Jackson Memorial Hospital and is a rising fellow in hematology and oncology at the University of Texas Southwestern Medical Center. He specializes in clinical data pipelines and R computer programming and is passionate about precision oncology, disparities research, and quality improvement. He has led numerous projects with sarcoma, thoracic oncology, neuroendocrine tumor, and the Firefighter Cancer Initiative.

9.2 Natalie Goulett, BS

Natalie is a healthcare simulation operations specialist and MPH student majoring in biostatistics at Florida International University. As a data science instructor, Natalie helps investigators conduct efficient and reproducible research using R. Her research interests include the impact of occupational hazards on first responders and the promotion of equity and quality improvement in research. Outside of her academic pursuits, Natalie enjoys live music and scuba diving.

9.3

Chapter 10

Glossary

Table here of important terms