O'REILLY®

Streaming Systems

流式系统（影印版）

Tyler Akidau, Slava Chernyak,
Reuven Lax 著

東南大學出版社

流式系统（影印版）

Streaming Systems

Tyler Akidau, Slava Chernyak,
Reuven Lax 著

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

O'Reilly Media, Inc.授权东南大学出版社出版

南京 东南大学出版社

图书在版编目(CIP)数据

流式系统:英文/(美)泰勒·阿克道(Tyler Akidau),(美)斯拉瓦·切尔尼亚克(Slava Chernyak),(美)鲁文·拉克斯(Reuven Lax)著. —影印本. —南京:东南大学出版社,2019.6

书名原文:Streaming Systems

ISBN 978-7-5641-8367-7

Ⅰ.①流… Ⅱ.①泰… ②斯… ③鲁… Ⅲ.①数据处理系统—英文 Ⅳ.①TP274

中国版本图书馆 CIP 数据核字(2019)第 074538 号

图字:10-2019-064 号

流式系统(影印版)

出版发行:东南大学出版社
地　　址:南京四牌楼 2 号　　邮编:210096
出 版 人:江建中
网　　址:http://www.seupress.com
电子邮件:press@seupress.com
印　　刷:江苏扬中印刷有限公司
开　　本:787 毫米×980 毫米　　16 开本
印　　张:22
字　　数:431 千字
版　　次:2019 年 6 月第 1 版
印　　次:2019 年 6 月第 1 次印刷
书　　号:ISBN 978-7-5641-8367-7
定　　价:128.00 元

本社图书若有印装质量问题,请直接与营销部联系。电话(传真):025-83791830

Preface Or: What Are You Getting Yourself Into Here?

Hello adventurous reader, welcome to our book! At this point, I assume that you're either interested in learning more about the wonders of stream processing or hoping to spend a few hours reading about the glory of the majestic brown trout. Either way, I salute you! That said, those of you in the latter bucket who don't also have an advanced understanding of computer science should consider how prepared you are to deal with disappointment before forging ahead; *caveat piscator*, and all that.

To set the tone for this book from the get go, I wanted to give you a heads up about a couple of things. First, this book is a little strange in that we have multiple authors, but we're not pretending that we somehow all speak and write in the same voice like we're weird identical triplets who happened to be born to different sets of parents. Because as interesting as that sounds, the end result would actually be less enjoyable to read. Instead, we've opted to each write in our own voices, and we've granted the book just enough self-awareness to be able to make reference to each of us where appropriate, but not so much self-awareness that it resents us for making it only into a book and not something cooler like a robot dinosaur with a Scottish accent.[1]

As far as voices go, there are three you'll come across:

Tyler

> That would be me. If you haven't explicitly been told someone else is speaking, you can assume that it's me, because we added the other authors somewhat late in the game, and I was basically like, "hells no" when I thought about going back and updating everything I'd already written. I'm the technical lead for the Data

1 Which incidentally is what we requested our animal book cover be, but O'Reilly felt it wouldn't translate well into line art. I respectfully disagree, but a brown trout is a fair compromise.

Processing Languages ands Systems[2] group at Google, responsible for Google Cloud Dataflow, Google's Apache Beam efforts, as well as Google-internal data processing systems such as Flume, MillWheel, and MapReduce. I'm also a founding Apache Beam PMC member.

Slava

Slava was a long-time member of the MillWheel team at Google, and later an original member of the Windmill team that built MillWheel's successor, the heretofore unnamed system that powers the Streaming Engine in Google Cloud Dataflow. Slava is the foremost expert on watermarks and time semantics in stream processing systems the world over, period. You might find it unsurprising then that he's the author of Chapter 3, *Watermarks*.

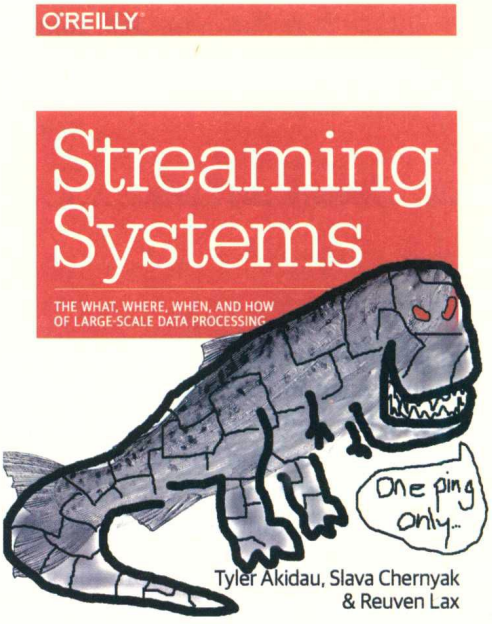

Figure P-1. The cover that could have been...

Reuven

Reuven is at the bottom of this list because he has more experience with stream processing than both Slava and me combined and would thus crush us if he were placed any higher. Reuven has created or led the creation of nearly all of the interesting systems-level magic in Google's general-purpose stream processing engines, including applying an untold amount of attention to detail in providing high-throughput, low-latency, exactly-once semantics in a system that nevertheless utilizes fine-grained checkpointing. You might find it unsurprising that he's the author of Chapter 5, *Exactly-Once and Side Effects*. He also happens to be an Apache Beam PMC member.

Navigating This Book

Now that you know who you'll be hearing from, the next logical step would be to find out what you'll be hearing about, which brings us to the second thing I wanted to

2 Or DataPLS, pronounced Datapals—get it?

mention. There are conceptually two major parts to this book, each with four chapters, and each followed up by a chapter that stands relatively independently on its own.

The fun begins with Part I, *The Beam Model* (Chapters 1–4), which focuses on the high-level batch plus streaming data processing model originally developed for Google Cloud Dataflow, later donated to the Apache Software Foundation as Apache Beam, and also now seen in whole or in part across most other systems in the industry. It's composed of four chapters:

- Chapter 1, *Streaming 101*, which covers the basics of stream processing, establishing some terminology, discussing the capabilities of streaming systems, distinguishing between two important domains of time (processing time and event time), and finally looking at some common data processing patterns.
- Chapter 2, *The What, Where, When, and How of Data Processing*, which covers in detail the core concepts of robust stream processing over out-of-order data, each analyzed within the context of a concrete running example and with animated diagrams to highlight the dimension of time.
- Chapter 3, *Watermarks* (written by Slava), which provides a deep survey of temporal progress metrics, how they are created, and how they propagate through pipelines. It ends by examining the details of two real-world watermark implementations.
- Chapter 4, *Advanced Windowing*, which picks up where Chapter 2 left off, diving into some advanced windowing and triggering concepts like processing-time windows, sessions, and continuation triggers.

Between Parts I and II, providing an interlude as timely as the details contained therein are important, stands Chapter 5, *Exactly-Once and Side Effects* (written by Reuven). In it, he enumerates the challenges of providing end-to-end exactly-once (or effectively-once) processing semantics and walks through the implementation details of three different approaches to exactly-once processing: Apache Flink, Apache Spark, and Google Cloud Dataflow.

Next begins Part II, *Streams and Tables* (Chapters 6–9), which dives deeper into the conceptual and investigates the lower-level "streams and tables" way of thinking about stream processing, recently popularized by some upstanding citizens in the Apache Kafka community but, of course, invented decades ago by folks in the database community, because wasn't everything? It too is composed of four chapters:

- Chapter 6, *Streams and Tables*, which introduces the basic idea of streams and tables, analyzes the classic MapReduce approach through a streams-and-tables lens, and then constructs a theory of streams and tables sufficiently general to encompass the full breadth of the Beam Model (and beyond).

- Chapter 7, *The Practicalities of Persistent State*, which considers the motivations for persistent state in streaming pipelines, looks at two common types of implicit state, and then analyzes a practical use case (advertising attribution) to inform the necessary characteristics of a general state management mechanism.
- Chapter 8, *Streaming SQL*, which investigates the meaning of streaming within the context of relational algebra and SQL, contrasts the inherent stream and table biases within the Beam Model and classic SQL as they exist today, and proposes a set of possible paths forward toward incorporating robust streaming semantics in SQL.
- Chapter 9, *Streaming Joins*, which surveys a variety of different join types, analyzes their behavior within the context of streaming, and finally looks in detail at a useful but ill-supported streaming join use case: temporal validity windows.

Finally, closing out the book is Chapter 10, *The Evolution of Large-Scale Data Processing*, which strolls through a focused history of the MapReduce lineage of data processing systems, examining some of the important contributions that have evolved streaming systems into what they are today.

Takeaways

As a final bit of guidance, if you were to ask me to describe the things I most want readers to take away from this book, I would say this:

- The single most important thing you can learn from this book is the theory of streams and tables and how they relate to one another. Everything else builds on top of that. No, we won't get to this topic until Chapter 6. That's okay; it's worth the wait, and you'll be better prepared to appreciate its awesomeness by then.
- Time-varying relations are a revelation. They are stream processing incarnate: an embodiment of everything streaming systems are built to achieve and a powerful connection to the familiar tools we all know and love from the world of batch. We won't learn about them until Chapter 8, but again, the journey there will help you appreciate them all the more.
- A well-written distributed streaming engine is a magical thing. This arguably goes for distributed systems in general, but as you learn more about how these systems are built to provide the semantics they do (in particular, the case studies from Chapters 3 and 5), it becomes all the more apparent just how much heavy lifting they're doing for you.
- LaTeX/Tikz is an amazing tool for making diagrams, animated or otherwise. A horrible, crusty tool with sharp edges and tetanus, but an incredible tool nonetheless. I hope the clarity the animated diagrams in this book bring to the complex topics we discuss will inspire more people to give LaTeX/Tikz a try (in

"Figures" on page xii, we provide for a link to the full source for the animations from this book).

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
: Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context.

This element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

Online Resources

There are a handful of associated online resources to aid in your enjoyment of this book.

Figures

All the of the figures in this book are available in digital form on the book's website. This is particularly useful for the animated figures, only a few frames of which appear (comic-book style) in the non-Safari formats of the book:

- Online index: *http://www.streamingbook.net/figures*
- Specific figures may be referenced at URLs of the form:

  http://www.streamingbook.net/fig/<FIGURE-NUMBER>

  For example, for Figure 2-5: *http://www.streamingbook.net/fig/2-5*

The animated figures themselves are LaTeX/Tikz drawings, rendered first to PDF, then converted to animated GIFs via ImageMagick. For the more intrepid among you, full source code and instructions for rendering the animations (from this book, the "Streaming 101" (*http://oreil.ly/1p1AKux*) and "Streaming 102" (*http://oreil.ly/1TV7YGU*) blog posts, and the original Dataflow Model paper (*http://bit.ly/2sXgVJ3*)) are available on GitHub at *http://github.com/takidau/animations*. Be warned that this is roughly 14,000 lines of LaTeX/Tikz code that grew very organically, with no intent of ever being read and used by others. In other words, it's a messy, intertwined web of archaic incantations; turn back now or abandon all hope ye who enter here, for there be dragons.

Code Snippets

Although this book is largely conceptual, there are are number of code and psuedo-code snippets used throughout to help illustrate points. Code for the more functional core Beam Model concepts from Chapters 2 and 4, as well as the more imperative state and timers concepts in Chapter 7, is available online at *http://github.com/takidau/streamingbook*. Since understanding semantics is the main goal, the code is provided primarily as Beam `PTransform`/`DoFn` implementations and accompanying unit tests. There is also a single standalone pipeline implementation to illustrate the delta between a unit test and a real pipeline. The code layout is as follows:

src/main/java/net/streamingbook/BeamModel.java
: Beam `PTransform` implementations of Examples 2-1 through 2-9 and Example 4-3, each with an additional method returning the expected output when executed over the example datasets from those chapters.

src/test/java/net/streamingbook/BeamModelTest.java
: Unit tests verifying the example `PTransforms` in *BeamModel.java* via generated datasets matching those in the book.

src/main/java/net/streamingbook/Example2_1.java
: Standalone version of the Example 2-1 pipeline that can be run locally or using a distributed Beam runner.

src/main/java/net/streamingbook/inputs.csv
: Sample input file for *Example2_1.java* containing the dataset from the book.

src/main/java/net/streamingbook/StateAndTimers.java
: Beam code implementing the conversion attribution example from Chapter 7 using Beam's state and timers primitives.

src/test/java/net/streamingbook/StateAndTimersTest.java
: Unit test verifying the conversion attribution `DoFn`s from *StateAndTimers.java*.

src/main/java/net/streamingbook/ValidityWindows.java
: Temporal validity windows implementation.

src/main/java/net/streamingbook/Utils.java
: Shared utility methods.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing a CD-ROM of examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Streaming Systems* by Tyler Akidau, Slava Chernyak, and Reuven Lax (O'Reilly). Copyright 2018 O'Reilly Media, Inc., 978-1-491-98387-4."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

O'Reilly Safari

Safari (formerly Safari Books Online) is a membership-based training and reference platform for enterprise, government, educators, and individuals.

Members have access to thousands of books, training videos, Learning Paths, interactive tutorials, and curated playlists from over 250 publishers, including O'Reilly Media, Harvard Business Review, Prentice Hall Professional, Addison-Wesley Professional, Microsoft Press, Sams, Que, Peachpit Press, Adobe, Focal Press, Cisco Press, John Wiley & Sons, Syngress, Morgan Kaufmann, IBM Redbooks, Packt, Adobe Press, FT Press, Apress, Manning, New Riders, McGraw-Hill, Jones & Bartlett, and Course Technology, among others.

For more information, please visit *http://www.oreilly.com/safari*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-998-9938 (in the United States or Canada)
707-829-0515 (international or local)
707-829-0104 (fax)

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at *http://bit.ly/streaming-systems*.

To comment or ask technical questions about this book, send email to *bookquestions@oreilly.com*.

For more information about our books, courses, conferences, and news, see our website at *http://www.oreilly.com*.

Find us on Facebook: *http://facebook.com/oreilly*

Follow us on Twitter: *http://twitter.com/oreillymedia*

Watch us on YouTube: *http://www.youtube.com/oreillymedia*

Acknowledgments

Last, but certainly not least: many people are awesome, and we would like to acknowledge a specific subset of them here for their help in creating this tome.

The content in this book distills the work of an untold number of extremely smart individuals across Google, the industry, and academia at large. We owe them all a sincere expression of gratitude and regret that we could not possibly list them all here, even if we tried, which we will not.

Among our colleagues at Google, much credit goes to everyone in the DataPLS team (and its various ancestor teams: Flume, MillWheel, MapReduce, et al.), who've helped bring so many of these ideas to life over the years. In particular, we'd like to thank:

- Paul Nordstrom and the rest of the MillWheel team from the Golden Age of MillWheel: Alex Amato, Alex Balikov, Kaya Bekiroğlu, Josh Haberman, Tim Hollingsworth, Ilya Maykov, Sam McVeety, Daniel Mills, and Sam Whittle for envisioning and building such a comprehensive, robust, and scalable set of low-level primitives on top of which we were later able to construct the higher-level models discussed in this book. Without their vision and skill, the world of massive-scale stream processing would look very different.
- Craig Chambers, Frances Perry, Robert Bradshaw, Ashish Raniwala, and the rest of the Flume team of yore for envisioning and creating the expressive and powerful data processing foundation that we were later able to unify with the world of streaming.
- Sam McVeety for lead authoring the original MillWheel paper, which put our amazing little project on the map for the very first time.
- Grzegorz Czajkowski for repeatedly supporting our evangelization efforts, even as competing deadlines and priorities loomed.

Looking more broadly, a huge amount of credit is due to everyone in the Apache Beam, Calcite, Kafka, Flink, Spark, and Storm communities. Each and every one of these projects has contributed materially to advancing the state of the art in stream processing for the world at large over the past decade. Thank you.

To shower gratitude a bit more specifically, we would also like to thank:

- Martin Kleppmann, for leading the charge in advocating for the streams-and-tables way of thinking, and also for investing a huge amount of time providing piles of insightful technical and editorial input on the drafts of every chapter in this book. All this in addition to being an inspiration and all-around great guy.
- Julian Hyde, for his insightful vision and infectious passion for streaming SQL.
- Jay Kreps, for fighting the good fight against Lambda Architecture tyranny; it was your original "Questioning the Lambda Architecture" (*https://www.oreilly.com/ideas/questioning-the-lambda-architecture*) post that got Tyler pumped enough to go out and join the fray, as well.
- Stephan Ewen, Kostas Tzoumas, Fabian Hueske, Aljoscha Krettek, Robert Metzger, Kostas Kloudas, Jamie Grier, Max Michels, and the rest of the data Artisans extended family, past and present, for always pushing the envelope of what's possible in stream processing, and doing so in a consistently open and collaborative way. The world of streaming is a much better place thanks to all of you.

- Jesse Anderson, for his diligent reviews and for all the hugs. If you see Jesse, give him a big hug for me.
- Danny Yuan, Sid Anand, Wes Reisz, and the amazing QCon developer conference, for giving us our first opportunity to talk publicly within the industry about our work, at QCon San Francisco 2014.
- Ben Lorica at O'Reilly and the iconic Strata Data Conference, for being repeatedly supportive of our efforts to evangelize stream processing, be it online, in print, or in person.
- The entire Apache Beam community, and in particular our fellow committers, for helping push forward the Beam vision: Ahmet Altay, Amit Sela, Aviem Zur, Ben Chambers, Griselda Cuevas, Chamikara Jayalath, Davor Bonaci, Dan Halperin, Etienne Chauchot, Frances Perry, Ismaël Mejía, Jason Kuster, Jean-Baptiste Onofré, Jesse Anderson, Eugene Kirpichov, Josh Wills, Kenneth Knowles, Luke Cwik, Jingsong Lee, Manu Zhang, Melissa Pashniak, Mingmin Xu, Max Michels, Pablo Estrada, Pei He, Robert Bradshaw, Stephan Ewen, Stas Levin, Thomas Groh, Thomas Weise, and James Xu.

No acknowledgments section would be complete without a nod to the otherwise faceless cohort of tireless reviewers whose insightful comments helped turn garbage into awesomeness: Jesse Anderson, Grzegorz Czajkowski, Marián Dvorský, Stephan Ewen, Rafael J. Fernández-Moctezuma, Martin Kleppmann, Kenneth Knowles, Sam McVeety, Mosha Pasumansky, Frances Perry, Jelena Pjesivac-Grbovic, Jeff Shute, and William Vambenepe. You are the Mr. Fusion to our DeLorean Time Machine. That had a nicer ring to it in my head—see, this is what I'm talking about.

And of course, a big thanks to our authoring and production support team:

- Marie Beaugureau, our original editor, for all of her help and support in getting this project off the ground and her everlasting patience with my persistent desire to subvert editorial norms. We miss you!
- Jeff Bleiel, our editor 2.0, for taking over the reins and helping us land this monster of a project and his everlasting patience with our inability to meet even the most modest of deadlines. We made it!
- Bob Russell, our copy editor, for reading our book more closely than anyone should ever have to. I tip my hat to your masterful command of grammar, punctuation, vocabulary, and Adobe Acrobat annotations.
- Nick Adams, our intrepid production editor, for helping tame a mess of totally sketchy HTMLBook code into a print-worthy thing of beauty and for not getting mad at me when I asked him to manually ignore Bob's many, many individual suggestions to switch our usage of the term "data" from plural to singular. You've managed to make this book look even better than I'd hoped for, thank you.

- Ellen Troutman-Zaig, our indexer, for somehow weaving a tangled web of offhand references into a useful and comprehensive index. I stand in awe at your attention to detail.
- Rebecca Panzer, our illustrator, for beautifying our static diagrams and for assuring Nick that I didn't need to spend more weekends figuring out how to refactor my animated LaTeX diagrams to have larger fonts. Phew x2!
- Kim Cofer, our proofreader, for pointing out how sloppy and inconsistent we were so others wouldn't have to.

Tyler would like to thank:

- My coauthors, Reuven Lax and Slava Chernyak, for bringing their ideas and chapters to life in ways I never could have.
- George Bradford Emerson II, for the Sean Connery inspiration. That's my favorite joke in the book and we haven't even gotten to the first chapter yet. It's all downhill from here, folks.
- Rob Schlender, for the amazing bottle of scotch he's going to buy me shortly before robots take over the world. Here's to going down in style!
- My uncle, Randy Bowen, for making sure I discovered just how much I love computers and, in particular, that homemade POV-Ray 2.x floppy disk that opened up a whole new world for me.
- My parents, David and Marty Dauwalder, without whose dedication and unbelievable perseverance none of this would have ever been possible. You're the best parents ever, for reals!
- Dr. David L. Vlasuk, without whom I simply wouldn't be here today. Thanks for everything, Dr. V.
- My wonderful family, Shaina, Romi, and Ione Akidau for their unwavering support in completing this levianthantine effort, despite the many nights and weekends we spent apart as a result. I love you always.
- My faithful writing partner, Kiyoshi: even though you only slept and barked at postal carriers the entire time we worked on the book together, you did so flawlessly and seemingly without effort. You are a credit to your species.

Slava would like to thank:

- Josh Haberman, Sam Whittle, and Daniel Mills for being codesigners and cocreators of watermarks in MillWheel and subsequently Streaming Dataflow as well as many other parts of these systems. Systems as complex as these are never designed in a vacuum, and without all of the thoughts and hard work that each of you put in, we would not be here today.

- Stephan Ewen of data Artisans for helping shape my thoughts and understanding of the watermark implementation in Apache Flink.

Reuven would like to thank:

- Paul Nordstrom for his vision, Sam Whittle, Sam McVeety, Slava Chernyak, Josh Haberman, Daniel Mills, Kaya Bekiroğlu, Alex Balikov, Tim Hollingsworth, Alex Amato, and Ilya Maykov for all their efforts in building the original MillWheel system and writing the subsequent paper.
- Stephan Ewen of data Artisans for his help reviewing the chapter on exactly-once semantics, and valuable feedback on the inner workings of Apache Flink.

Lastly, we would all like to thank *you*, glorious reader, for being willing to spend real money on this book to hear us prattle on about the cool stuff we get to build and play with. It's been a joy writing it all down, and we've done our best to make sure you'll get your money's worth. If for some reason you don't like it...well hopefully you bought the print edition so you can at least throw it across the room in disgust before you sell it at a used bookstore. Watch out for the cat.[3]

3 Or don't. I actually don't like cats.

Table of Contents