A Sociotechnical Typology of Scientific Software

Drew Paine, Hannah Cohoon, Sarah Poon, Rajshree Deshmukh, Cody O'Donnell, Dan Gunter, Lavanya Ramakrishnan
strudel@lbl.gov

Abstract

User experience (UX) work is key to the development of usable, sustainable software. Within scientific software development the adoption of UX methods is increasing but not yet common enough. The **S**cientific sof**T**ware **R**esearch for **U**ser experience, **D**esign, **E**ngagement, and **L**earning (STRUDEL) project launched in 2022 to develop resources for scientific software development that are needed to improve user experience, software quality, and software sustainability. One key project objective was the identification and classification of key characteristics of scientific work and software into a typology. This report documents the STRUDEL Typology, including discussion of the motivation for this type of conceptual tool, and discusses known gaps for future work.

Keywords — scientific software development, user experience, sociological typology, sociotechnical, software sustainability, STRUDEL
Report Number: *LBNL-2001570, https://escholarship.org/uc/item/9g27k0h8*
Acknowledgments: This work is funded by gifts of the Sloan Foundation to Berkeley Lab, grants #10074, #10572, and #22557. The contents of this report represent the views of the authors and do not represent that of any other entity.

Executive Summary

The **S**cientific sof**T**ware **R**esearch for **U**ser experience, **D**esign, **E**ngagement, and **L**earning (STRUDEL) project began in 2022 to develop resources for scientific software development to improve user experience, software quality, and software sustainability. A key objective was the identification of key characteristics of scientific work and software; these characteristics were organized into a *typology* that facilitates planning and identification of knowledge gaps. This report defines the STRUDEL Typology which classifies scientific software development work.

Typologies are an approach to classification based on general types, enabling structured analysis and comparison. STRUDEL leverages this social science approach to systematically analyze and compare the work diverse scientific projects producing software under take. Developed based on the STRUDEL team's experience supporting scientific software teams and in light of established literature, the typology recognizes commonalities that impact user experiences and sustainability of the software being produced.

This first version of the STRUDEL Typology includes three key facets that include additional dimensions. The first facet, Project Composition, captures the overall organizational elements that shape how a scientific project is organized and executed. Domain, primary purpose, funding model, key products, and success metrics are all dimensions within Project Composition. The second facet, People and Teams, includes dimensions like team size, physical distribution, roles and composition, turnover and diversity of experience. The third and largest facet, Software Product, includes five subsections: meta, user groups, user interfaces, data products, and STRUDEL Task Flows. The last of these subsections, Task Flows, are sets of steps (represented by a series of screens) that help to accomplish a task and represent how a user progresses through a UI. A series of Task Flows may be composed to create a formal Workflow using the STRUDEL Design System, a complement to the STRUDEL Typology that is described on the STRUDEL website: strudel.science. To recognize knowledge gaps or guide planning, users of the STRUDEL Typology can answer questions regarding the various dimensions. The STRUDEL Typology supplies these questions and example answers.

Introduction and Background

Software is a ubiquitous and critical resource in scientific research. Scientific software development is a complex sociotechnical process essential to the production of knowledge. A significant ongoing challenge to developing and stewarding scientific software is the substantial work required to ensure that it is a reliable, usable, and sustainable product for user communities. Sustainable software continues to be available and meet user needs over time, despite changes in the surrounding software ecosystem [3]. User experience (UX) research and UX guided development are critical aspects of sustainability and are important for delivering reliable and useful software to user communities [4,6,7].

The Scientific sofTware Research for User experience, Design, Engagement, and Learning (STRUDEL[1]) project was created by the User Experience (UX[2]) team in Berkeley Lab's Scientific Data Division in 2022 to provide UX resources to scientific communities. STRUDEL is a tool for developing usable scientific interfaces and helping teams better manage outcomes around their software sustainably over time. The STRUDEL team set out to address two key questions.

- What are the characteristics of scientific work and software and how can they be classified into a typology to better improve user experience and software sustainability?
- What are the elements of a design framework for user experience of scientific software?

The remainder of this report addresses the first question by presenting the STRUDEL Typology. The second question is addressed through our Design System, available on our website[3]. We present the motivation for the STRUDEL Typology, its overarching structure, and detailed facets and questions an end user can consider when building scientific software.

Motivation — Why have a typology?

A typology enables researchers to systematically analyze and compare diverse phenomena. By applying a typology, you can recognize key characteristics of what you are studying and distinguish it from similar phenomena. The STRUDEL typology for scientific projects helps surface factors that impact user experience and software sustainability, enabling those projects to make better decisions and plan effectively. When software projects apply the typology, they can identify knowledge gaps that need to be addressed, consider how certain characteristics may affect their planning and development efforts, and they have the opportunity to surface tacit knowledge.

[1] https://strudel.science
[2] https://ux.lbl.gov
[3] https://strudel.science/design-system/overview/

Scientific software ranges in complexity from stand-alone analysis scripts to large-scale cyberinfrastructure for collecting, analyzing, and storing data from multi-site experiments. Scientific software project teams also vary in size and makeup. Much of scientific software is developed by a single author though large, distributed teams also produce important packages and libraries [1]. Relevant incentives and challenges may be influenced by a project's organizational makeup and setting [2]. For instance, doctoral students preparing for a faculty role may prioritize novel contributions and rapid development over best practices and thorough documentation, a sustainability challenge for software developed in university labs. In contrast, Other organizations developing scientific software may hire professional research software engineers, minimizing those concerns though introducing others like budget.

Our team developed the typology as a thinking aid so that we could offer strategic advice to scientific software projects, tailored to the projects' individual circumstances. However, the STRUDEL Typology can be used by anyone involved with planning, developing, or sustaining a scientific software product who needs to think through current or potential challenges in their work. Example stakeholders include individuals fulfilling a variety of roles such as:

- **Project Leadership.** Project leaders can identify opportunities for UX, Software Sustainability, Community Engagement, etc. and determine when and how much to invest over a project's life cycle.
- **Project Domain Experts.** Scientists on the project can identify & give feedback on common workflows to help specify requirements for the software products being developed.
- **Project Developers.** Project developers can use the typology to identify common Task Flows and examine which pieces of the Design System to start their work with.
- **Funding Agencies.** Staff can use the typology to evaluate proposed projects and request investment or planning for UX, software sustainability, and so on.

Example Use Case

Imagine you are a leading a scientific team that has acquired funding to build a web-based repository for environmental datasets...

Your project has research software engineers to help build the repository and domain science experts who will be engaging with the community on how to process and submit their data.

Your project kickoff is taking place in a week and the leadership team needs to prioritize effort. A variety of community needs emerged through the proposal process. A few key foundational technical concerns are driving the initial backend engineering work, but it's not clear how to approach the prioritization and design of the user interfaces or allocation of effort once core infrastructure is in place.

To help orient your thinking you turn to the STRUDEL typology. It helps you categorize more of the project work and determine key questions for the team to discuss during the kickoff meeting. For example, your answers might prompt you to ask:

- How will we reduce the negative effects of high contributor turnover on our code quality?
- How will we stand out against competitors?
- What workflows will our users expect to be supported?
- When should user testing be done?

Our long-term goal is to leverage the typology to design a Planning Framework. This Planning Framework would enable scientific software teams to improve their project planning and design processes, going beyond highlighting knowledge gaps and surfacing important questions and instead providing recommendations to users.

Typology Overview

The STRUDEL Typology is a conceptual tool for categorizing elements of scientific projects building and using software products. The STRUDEL Typology was iteratively developed through review of sociotechnical literature and analysis of interviews with five scientific project leaders. We crafted the typology to help identify and compare common activities in scientific software work that ought to be considered when focusing on requirements, user experience, sustainability, and other important considerations. The focus on scientific software production and use, especially its UX and sustainability, is a key distinction and bias of the STRUDEL Typology, rather than focusing on other scientific concerns like data life cycles [9] or more general topics like the organization of human infrastructure [5]. STRUDEL is constructed to help tease apart three interconnected aspects of work to help typology users focus on the relationships between different key elements of software (Figure 1).

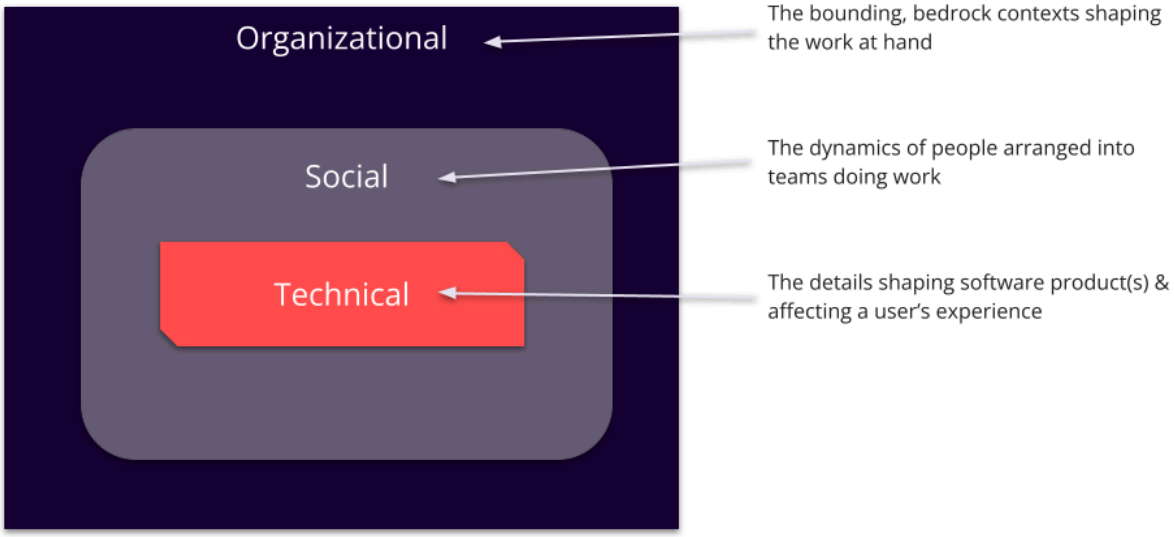

Figure 1. Illustration of the three high level interconnected aspects of scientific work shaping software development and use. At the highest level Organizational concerns shape the context of work, Social concerns capture and highlight the dynamics of people doing work, and Technical concerns foreground details shaping software products.

The three overarching aspects are categorized by Organizational, Social, and Technical concerns. The Organizational and Social concerns shape the entire context of a project and its ability to deliver software to a user community. Technical concerns foreground particular threads in a project's context that are bound up with Organizational and Social aspects to influence the design and use of scientific software for research work. We unpack the dimensions and questions for every facet below.

Typology Facets

The following sections lay out the dimensions of the STRUDEL Typology's three high level facets: Project Composition, People and Teams, and Software Products. For each high level facet we present dimensions capturing a category or concept. Each dimension has one or more key questions that a scientific software project team may want to consider the answer(s) to in their decision making processes. Example answers are provided.

Project Composition

Project Composition captures the overall organizational elements that shape how a scientific project is organized and executed. This category of information is important to helping stakeholders characterize and reflect upon the structure of a project. Such understanding contextualizes the software this entity is creating, using, or supporting for a focal user community.

Recognizing and reflecting on the categories of questions below is helpful to Principal Investigators and technical leaders on a project when evaluating how the team will be able to support development and stewardship of particular software(s).

Dimension	Key Questions	Example Answers
Science Domain	How many domain(s) does the project and its products serve?	● One domain ● A handful of domains ● General purpose for science
	What domain(s) does the project and its products serve?	
Primary Purpose	What is the foundational purpose of the project?	● Develop Software application or library ● Produce theoretical results ● Produce experimental results ● Create a physical artifact (instrument) ● Develop data repository or service
Funding Model(s)	Who is funding the project?	● NSF ● DOE ● NASA
	What is being funded?	

Dimension	Key Questions	Example Answers
	Are software products being directly funded? If so, for how long?	
Key Product(s)	What are the key products being created and/or focused on?	● Scientific Papers ● Dataset(s) ● Instrument(s) ● Analyses ● Software ● Documentation
Success Metrics of Product(s)	What are key metrics that the project must collect & share with a funding agency about its product(s)?	● Number of citations to dataset(s) ● Type(s) of citations to dataset(s) ● Number of users: visitors and/or downloads ● Science impact

People and Teams

The People and Teams facet captures the overall social elements shaping how a scientific project conducts work. This facet captures information about the individuals responsible for developing and supporting the project's software products and how they're organized to accomplish this work. These dimensions are represented to help capture the diversity of contributors to the work while making visible the relationships which can shape and affect a project's success delivering software over time.

Identifying and reflecting on these questions is key to all members of a project when evaluating how they're collectively able to support the development and stewardship of particular software(s). Understanding the disciplinary backgrounds and skill sets directly influences the trajectory of the development work depending on whether the appropriate individuals are available. Identifying and tracking turnover within a project and its teams alongside changes in the physical distribution of members may also be key to ensuring work is sustained reliably over time based on the availability of key individuals or skill sets.

RSE = Research Software Engineer

Dimension	Key Questions	Example Answers
Size	What is the scale of the project team?	● Xtra Small: 2-5 people ● Small: 6-15 ● Medium: 16 - 50

Dimension	Key Questions	Example Answers
		• Large: 51 - 100 • Xtra Large: 100+
Physical Distribution	How are the members of the project team distributed geographically?	• 1 site • 2-3 sites • 4+ sites
	How many people are co-located at each site and which roles do they hold?	
Roles & Composition	What roles individuals hold and how many people from each role are on the team?	• Domain Scientists: X • RSE - Facility Staff: X • RSE - Full Stack: X • RSE - AI/ML: X • RSE - UX/Front End: X • Community Engagement: X • Administrative: X • Other: X
	How many roles do individuals typically hold?	
	How many Domain Scientists write code for the project's software products? (As opposed to just their own analysis code)	
	How many RSEs, from which categories, came to that role from a Domain Scientist background?	
Turnover	How frequently do individuals join and leave teams?	
	What roles do these individually commonly fulfill?	
	What backgrounds do these individuals come from?	

Dimension	Key Questions	Example Answers
Diversity of Experience	What is the distribution of people across levels of seniority?	• Student: X • Postdoc: X • Early Career: X • Senior: X
	What types of educational and/or professional backgrounds do team members have?	

Software Product(s)

The Software Product(s) facet is the most extensive and is crafted to encompass a range of technical considerations. This facet has five key sub-elements which each have multiple dimensions and questions organized to highlight key aspects relevant to the production of the software product(s) a project is developing and sustaining over time. The five key sub-aspects characterized so far are:

- Meta: contextual details about the purpose of the software
- User Groups: details about intended users
- User Interfaces: elements for user interaction
- Data Products: details on data key to the software
- STRUDEL Task Flows: how users complete activities

In addition, future work may call out a dimension specifically on documentation which is a key element of any software's use and sustainability over time.

Software Products — Meta

The Meta sub-dimension surfaces general elements characterizing a scientific software product's purpose and how it is developed. Much of this information may be needed to understand the history and context of the particular software product in question. Meta dimensions capture broad technological aspects (computing paradigms, tech stacks) along with social/organizational issues (contribution and support models) highlight details key to getting a product to be successfully used by a scientific community.

Dimension	Key Questions	Example Answers
Required Types	What types of software products does the project require?	• Framework/Library • Web app

Dimension	Key Questions	Example Answers
		• API • Other (list)
	When are the individual types of software needed by the project?	• Immediately • During instrument construction • Before experiment begins • Other
Computing Paradigms	What types of systems is the software developed to run on without modifications?	• Personal • Cluster • HPC • Cloud
Product Competition	What products already exist from outside of this project?	
	What would be in competition with this project's product outputs?	
Technology Stack(s)	What technologies are being used or developed?	Programming languages or frameworks, etc.
	Who is primarily developing this stack?	• The project team • Scientific community • Open Source organization • Commercial entity • Hybrid
Distribution Model(s)	How is the software product distributed to end users? (possibly multiple ways)	• Direct desktop download • Package manager • Service • Web GUI • Mobile GUI • API only
Contribution Model(s)	Who is able to contribute to the software's development?	• Solely the project team • Fully Open Source

Dimension	Key Questions	Example Answers
		• Other?
Support Model(s)	How is (or does) the project support the software product(s) being developed?	
	How is this tied to the project's funding model?	
	How long does the project anticipate supporting a particular software product?	

Software Products — User Groups

User Groups are key sub-dimensions of Software Product(s). These dimensions emphasize characteristics about the intended user community of a software product whose user experience is being shaped.

Dimension	Key Questions	Example Answers
User Type(s)	What types of users are anticipated?	
	What backgrounds do they come from?	
	Are project team members also members of the end user community? If so, which roles do they fulfill?	
Size of User Base	How many people actively use the product?	• Small (<100 people) • Medium (100 - 1,000 people) • Large (>1,000 people)
Usage Frequency	For each type of user how frequently are they using the product?	• Daily • Weekly • Monthly • Few times a year • Yearly • Once

Dimension	Key Questions	Example Answers
Usage Motivation	For each type of user why are they using this product?	

Software Products — User Interfaces

The User Interfaces (UIs) sub-dimension focuses on key details about the user interfaces a project's software needs to provide to support its target user base. Capturing and reflecting on the information these dimensions capture is likely necessary for teams to understand what interfaces they are building (APIs, web sites, etc.) and how they plan to accomplish this essential work.

Dimension	Key Questions	Example Answers
UI Type(s)	What types of UIs does the software product need to provide?	• CLI • API • GUI - Mobile • GUI - Web • GUI - Desktop • GUI - Notebook
Usage by UI Type(s)	Which types of UIs are the most commonly used? By which user roles?	
UI Framework(s)	Has the product team chosen any UI frameworks?	
	What UI framework(s) is the project using?	• React • Bootstrap
	What motivated the selection of a particular UI framework?	
UI Development Stage(s)	When are each type of UI being developed by the project?	• As part of first release/MVP • Secondary release • Uncertain

Dimension	Key Questions	Example Answers
UI Patterns	What types of UI patterns are currently being used or anticipated to be implemented?	• Information modules: entity summaries, entity detail views, dashboards, timelines, history • Data views: catalogs, data tables • Data inputs: multi step wizards, very long forms • Data visualizations: basic charts and graphs, maps, structural diagrams, tabular diagrams

Software Products — Data Products

Data Products is the next sub-dimension of Software Product(s) and foregrounds a couple of key concerns about scientific data. We scoped this dimension narrowly since there are myriads of characterizations of scientific data (e.g., FAIR principles [10] and data life cycle models [8]) that project teams may reference.

Dimension	Key Questions	Example Answers
Data Types	What types of data are being produced or used through this software product?	• Experimental • Observational • Computational or Theoretical • Other
Openness	How is data obtained or used in the project in conjunction with this software product?	• User contributed • Project gathered • Community repository • Proprietary commercial • Other

Software Products — STRUDEL Task Flows

STRUDEL Task Flows are the final sub-dimension of the Software Product(s) aspect. Task Flows are a primary element of the STRUDEL Design System which are sets of steps (represented by a series of screens) that help to accomplish a task and represent how a user progresses through a UI[4]. A series of Task Flows may be composed to create a formal

[4] https://strudel.science/design-system/task-flows/overview/

Workflow that can be specified using a graph representation. Workflows enable scientific software users to accomplish their work and we are identifying key Task Flows that end up being combined by projects and users to accomplish their work.

Dimension	Key Questions	Example Answers
Type(s) in Use	What types of Task Flows are required and in use in the software product?	• Compare Data • Contribute Data
Purpose of Flows	What are the reasons for using the flow?	• Enabling user to upload data to a repository • Enabling setup of a computational job for execution on an HPC system
Complexity of Flows	How complex is the task flow to enable a user's work?	• # of steps to complete • # of points of human interaction • Amount of time • Density of data/information

Known Gaps for Future Work

The first version of this typology captures an extensive state of scientific software production. It has been drafted based on the experience and expertise of the UX team to reflect the dimensions that were most relevant to consultation and design efforts.

The sociotechnical landscape related to scientific software development is broad and further additions to the STRUDEL Typology are needed. Through preliminary socialization and examination of additional projects we know that there are a range of gaps worth further exploring and resolving in future iterations. For example, documentation and mechanisms for providing code contributions are both largely overlooked in our initial STRUDEL Typology.

As the STRUDEL Typology is expanded and revised, some dimensions may be reorganized and renamed. The key facets and dimensions of the present STRUDEL Typology were chosen to strategically direct attention. For example, the Data Products dimension in the Software Products facets was named as such to distinguish it from other products like papers or instruments and maintain a narrower scope. As the Typology becomes more comprehensive, it may prove more appropriate to include these other possible products, prompting a new dimension title and/or position within the Typology.

A final important avenue for further development is the Planning Framework. Going beyond highlighting knowledge gaps and surfacing important questions as the Typology does now, this tool is envisaged as a way for developers to apply the Typology and receive tailored advice. Our expectation is that software teams will leverage the Planning Framework to apply appropriate resources in a timely manner and anticipate and mitigate UX challenges.

Conclusion

The STRUDEL Typology is a tool for assessing the characteristics of scientific software projects. By responding to key questions about project facets and dimensions, software developers and other stakeholders can identify knowledge gaps, initiate important planning discussions, and surface tacit knowledge. The STRUDEL Typology is informed by the experience of the UX team in Berkeley Lab's Scientific Data Division and will continue to be revised and expanded. Feedback and contributions are welcome on the project's Github (https://github.com/strudel-science/).

References

1. V. R. Basili, J. C. Carver, D. Cruzes, L. M. Hochstein, J. K. Hollingsworth, F. Shull, and M. V. Zelkowitz. 2008. Understanding the High-Performance-Computing Community: A Software Engineer's Perspective. *Software, IEEE* 25, 4: 29–36.
2. Johanna Cohoon, Caifan Du, and James Howison. 2025. Tales of Transitions: Seeking Scientific Software Sustainability. *Proceedings of the ACM on human-computer interaction* 9, 1. https://doi.org/10.1145/3701208
3. Asif Imran and Tevfik Kosar. 2019. Software sustainability: A systematic literature review and comprehensive analysis. *arXiv [cs.SE]*. Retrieved from http://arxiv.org/abs/1910.06109
4. Drew Paine, Devarshi Ghoshal, and Lavanya Ramakrishnan. 2020. *Investigating Scientific Data Change with User Research Methods*. Lawrence Berkeley National Laboratory, Berkeley, CA. Retrieved from https://escholarship.org/uc/item/87b7h27d
5. Drew Paine and Charlotte P. Lee. 2020. Coordinative Entities: Forms of Organizing in Data Intensive Science. *Computer supported cooperative work: CSCW: an international journal* 29, 3: 335–380. https://doi.org/10.1007/s10606-020-09372-2
6. Lavanya Ramakrishnan and Daniel Gunter. 2017. *Ten Principles for Creating Usable Software for Science*. https://doi.org/10.1109/eScience.2017.34
7. Lavanya Ramakrishnan, Sarah Poon, Valerie Hendrix, Daniel Gunter, Gilberto Z. Pastorello, and Deborah Agarwal. 2014. *Experiences with User-Centered Design for the Tigres Workflow API*. https://doi.org/10.1109/eScience.2014.56
8. UK Data Archive,. 2013. Research Data Lifecycle. Retrieved 2013 from http://data-archive.ac.uk/create-manage/life-cycle
9. Jillian C. Wallis, Alberto Pepe, Matthew S. Mayernik, and Christine L. Borgman. 2008. *An Exploration of the Life Cycle of eScience Collaboratory Data*. Retrieved from http://hdl.handle.net/2142/15122
10. Mark D. Wilkinson, Michel Dumontier, Ijsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, Jildau Bouwman, Anthony J. Brookes, Tim Clark, Mercè Crosas, Ingrid Dillo, Olivier Dumon, Scott Edmunds, Chris T. Evelo, Richard Finkers, Alejandra Gonzalez-Beltran, Alasdair J. G. Gray, Paul Groth, Carole Goble, Jeffrey S. Grethe, Jaap Heringa, Peter A. C. 't Hoen, Rob Hooft, Tobias Kuhn, Ruben Kok, Joost Kok, Scott J. Lusher, Maryann E. Martone, Albert Mons, Abel L. Packer, Bengt Persson, Philippe Rocca-Serra, Marco Roos, Rene van Schaik, Susanna-Assunta Sansone, Erik Schultes, Thierry Sengstag, Ted Slater, George Strawn, Morris A. Swertz, Mark Thompson, Johan van der Lei, Erik van Mulligen, Jan Velterop, Andra Waagmeester, Peter Wittenburg, Katherine Wolstencroft, Jun Zhao, and Barend Mons. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data* 3: 160018. https://doi.org/10.1038/sdata.2016.18