Too Many Buttons*

Why Not Domain-Specific Scientific Visualization?

Chad Zanocco †
School of Public Policy
Oregon State University
Corvallis OR 97331 USA
zanoccoc@oregonstate.edu

Denise Lach
School of Public Policy
Oregon State University
Corvallis OR 97331 USA
denise.lach@oregonstate.edu

Judith Cushing
Computer Science
The Evergreen State College
Olympia WA 98505 USA
judyc@evergreen.edu

Jonathan Halama
Environmental Science
Oregon State University
Corvallis OR 97331 USA
halamaj@oregonstate.edu

Nikolas Stevenson-Molnar
Software & Tools
Conservation Biology Institute
Corvallis OR 97333 USA
nik.molnar@consbio.org

ABSTRACT

This case study involves social scientists, environmental scientists, computer scientists and software developers who use knowledge co-production methods to develop scientific visualization software for the complex domain of environmental science. We observe that existing generalized visualization software is not usable by these scientists ("too many buttons"), but that special purpose software like ours is likely too resource intensive to develop and maintain for specific domains. We therefore propose that CHI researchers consider researching ways to engineer general-purpose visualization software so that domain specific products can be generated via specifications. A second lesson learned is that co-production methods, particularly if employed by interdisciplinary teams with CHI or social science expertise, are especially effective for specifying and developing software where the domain is complex and end users have little prior experience with the particular technology.

CCS CONCEPTS

• **Visualization application domains~Scientific visualization** • **Human computer interaction (HCI)~HCI design and evaluation methods** • Visualization systems and tools:~Visualization toolkits • Software creation and management~Designing software ~Requirements analysis

KEYWORDS

Knowledge co-production, domain-specific software, scientific visualization, environmental science, user-centered software development

FirstName Surname, FirstName Surname and FirstName Surname. 2018. Insert Your Title Here: Insert Subtitle Here. In *Proceedings of ACM Woodstock conference (WOODSTOCK'18). ACM, New York, NY, USA, 2 pages.* https://doi.org/10.1145/1234567890

1 Introduction: Case Study Context

This case study considers one of several software development phases of the VISTAS (Visualization of Terrestrial and Aquatic Systems) project, a collaborative effort funded by the U.S. National Science Foundation to co-produce visualization software for environmental scientists [4, 18, 19]. The primary research questions we explored are 1) how environmental scientists use software for visualizing models and data, and 2) how the process of co-producing domain-specific visualization software impacts the way these scientists conduct research.

While there are many approaches to collaborative and iterative software development (e.g., user-centered design, agile development, etc.), the VISTAS project uses a conceptualization of co-production from the social sciences frequently applied to technology and knowledge production activities involving such interdisciplinary fields as health services, climate science, and environmental management (e.g., [16]). Launched in 2011, the VISTAS project is a co-production effort that seeks to enable scientists to better understand and communicate information about complex environmental problems through visualization. The project brings together researchers and staff at educational institutions including The Evergreen State College and Oregon State University, non-profit research and development institutes, and government agencies. The VISTAS development team is comprised of a diverse group of experts including computer scientists and software engineers with expertise and prior experience in scientific databases, computer graphics, and the development of scientific visualization software. VISTAS users are primarily environmental scientists, ecologists, and members of their research groups interested in using established and emergent visualization techniques to better understand and communicate complex landscape-level data and models. VISTAS developers and users work together to co-produce visualization software, referred to as the same name as the project, VISTAS. During this

co-production process, a social science team is facilitating and studying the efficacy of VISTAS software development. VISTAS software co-production is an ongoing effort evolving across multiple development stages, with the most recent of these development phases forming the basis for the case study activities reported here.

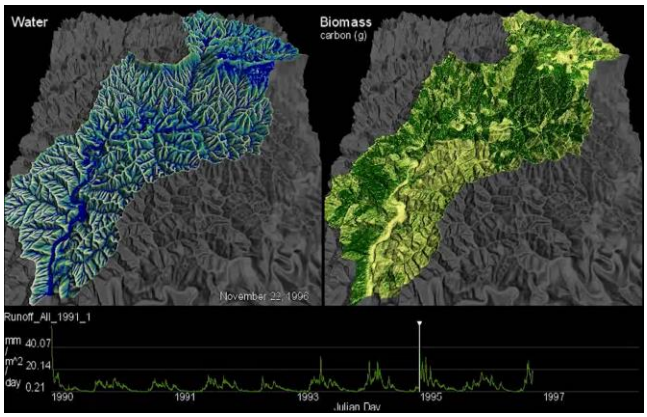

Figure 1: VISTAS Visualization of Modeled Precipitation (left panel), Biomass (right panel) and Hydrograph of Watershed Stream Flows (bottom panel)

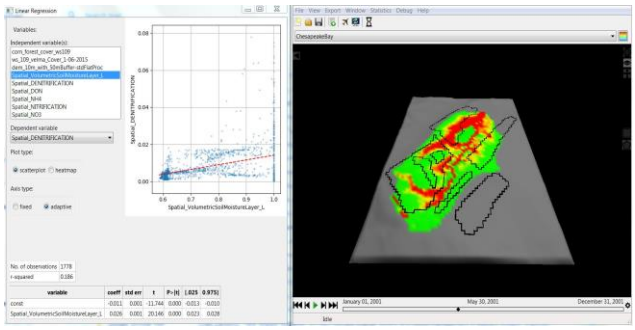

Figure 2: Example of VISTAS Data Analytics Feature (left panel; linear regression) alongside Visualized Landscape (right panel)

1.1 Description of VISTAS Software

The VISTAS visualization software is designed to display complex spatiotemporal data. Now written in Python (originally developed in C^{++}), VISTAS is capable of displaying three-dimensional topographic data that varies across time. Users can import data into VISTAS using common file formats that are vector-based (e.g., ESRI shapefile) or raster-based (e.g., Digital Elevation Model, ASCII). They then can control in-tool visualization properties such as data layers, select colors for viewing data attributes, and change perspectives of the displayed scene. Users can also manage multiple scenes viewed concurrently and label static images, as well as generate "fly-throughs" over either space or time. Images, including videos, can be exported for incorporation into presentations and reports. Recent additions to VISTAS include new analysis functionality—including linear regression and principal component analysis (PCA)—that are dynamically updated to match data displayed in the current scene. These analytics can be viewed concurrently with other visual display selections. See Figures 1 and 2 for examples of how environmental scientist users have applied VISTAS data display and data analytics. For more information about VISTAS, please see http://blogs.evergreen.edu/VISTAS.

2 Methods

For this case study, we recruited collaborators from the VISTAS project to co-produce visualization software in a development phase from Fall 2016 through Spring 2018. These collaborators include both user groups and members of the development team. To understand the effects of software co-production on domain expert users and developers, we used a multi-period data collection approach to elicit perspectives of project collaborators at different stages of the co-production process. In the case study research described below, this elicitation occurred in three stages within a pre/post-test case study design [13] with three stages:

1. In the first stage, the pre-period, a baseline is established where collaborator needs and perspectives are assessed.
2. In the second stage, the development period, collaborators interact regularly to provide input and guidance about software design.
3. In the third and final stage, the post-period, the newly developed software is assessed by those involved in the co-production effort, and the overall process of co-production is considered by project collaborators.

We formally applied this pre/post design in the assessment of the VISTAS project.

To understand the perspectives of project participants, we conducted 23 semi-structured interviews, ranging in length from 30 minutes to 2 hours, with 12 interviews conducted in the pre-period and 11 interviews conducted in the post-period. Of these interviews, five participants were interviewed in both the pre- and post-period. See Table 1 for a summary of interview timing in the context of the development phase. Open coding was used to analyze interview data and emergent themes were identified; see Table 2 for an abridged codebook.

Table 1: Timeline of research activities

VISTAS co-production	Pre-period: Nov - Feb 2017	Development: Feb 2017 – May 2018	Post-period: May-Sept 2018
Software co-development		Vector flows, data analytics, Python integration	
Participant interviews	Prior to development (N=12)		After development (N=11)
Participant observations	Attend weekly development meetings, record interactions, collect documentation		

Table 2: Codebook for analyzing participant data

Code Group	Primary Code	Secondary Code
Visualization	Technical challenges	Challenges with data
		Challenges producing visualizations from data
	Exploration	Understanding data
		Validation and troubleshooting
		Analytics
	Communication	Scientist audiences
		Non-Scientist audiences
Co-production	Context	Institutional factors
		Logistical factors
	Inputs	Proficiency & expertise for knowledge co-production
		Legitimacy and trust
		Inclusivity
	Process	Setting-up
		Development and design
		Implementation
		Outputs
	Outcomes-impacts	Changes in practice
		Salience of knowledge co-production

3 Findings: Existing software products were not always usable

One of the main themes that emerged in analyzing interview data in the pre-period was a need for visualization software that is intuitive to use yet also fits the specific, and technically complex, user applications. This sentiment was described by our ecologist and environmental scientist collaborators who required visualization software capable of displaying topographic data across time in order for them to interpret and communicate results from computational models. With respect to this overall need, one scientist described the burden of learning to use other programs capable of displaying their data: "A lot of us just don't have time to learn Arc [ESRI ArcGIS]…to [be] able [to], within minutes, load a DEM [Digital Elevation Map] in and then drape output over that. [VISTAS has] really been a help to what we do and [to] communicating that work" [20].

While this researcher felt that this need was unique to his field, other researcher collaborators working with different data in a different domain also saw a need for visualizing data topographically to validate and interpret model results. While these two groups of scientists studying different topics felt that their needs were unique, they actually had common needs that were not easily satisfied using existing scientific visualization or GIS software, such as ArcGIS, ParaView, QGIS, Google Earth, etc. For these groups, the scientific questions they asked were influenced or determined by terrain and topographic features, and VISTAS software was designed to fit this specific niche. Software co-production led to unexpected uses, new features, and reprioritizing needs.

During the recent development phase, participants in the project indicated that VISTAS software was often used in unanticipated ways, for example not only (as expected) to communicate model results to stakeholders and to refine their scientific models, but also to integrate scientific models that run over hundreds or even thousands of time steps. In these integrated models, the output of one model from time t is used as the input to another at time $t+1$ and vice versa. Model integration is generally considered as complicated as developing new models since unexpected model conflicts arise, and scientists with different domain expertise need to collaborate to identify issues, and modify code or parameters for one or another of the models.

One collaborator from Environmental Team 1 had extensive experience using multiple versions of the VISTAS software, and served as a beta tester. One reason he was so active as a VISTAS user was that he was developing a sunlight irradiance model (Model B below) that he was coupling with an existing ecological model already in use within his research team (Model A below). Two of our collaborators used VISTAS first to refine their individual models and then to help integrate those two models so that biomass data produced by Model A [1, 5, 15] would be an input at each time step to Model B [7]. To use visualization in the validation of the integrated models, this scientist needed to view outputs of each model simultaneously, synchronize the views by time step, and control the speed of the visualization. At various points during the model integration process he modified Model B so that additional variables could be viewed; he wrote into Model B the ability to output biomass data from Model A and view in VISTAS how biomass at each time step affected leaf transmittance and tree height results (even though these were intermediate data) so he could visualize the data conversions in Model B and assess whether they were spatially correct.

For phenomena this researcher was exploring, terrain displayed in 3D was critical, as he later explained: "If you imagine that you have a hillside and a tree on top of a hillside, depending on your solar angle then that tree could cast a shadow all the way down that hillside to the base of what you are looking at" [20]. He noted that displaying model output in VISTAS became important for exploring phenomena such as "whether shadows are moving the right way, is the sun being modeled correctly, etc." [20]. This researcher used topographic features displayed through VISTAS to corroborate if newly integrated Model B was casting shadows appropriately, and whether this irradiance information was properly integrated in Model A.

Also in this development phase, as VISTAS was refined to meet basic visualization needs, users started to restate and reprioritize desirable features. While data analysis had been a long-term

overall project goal, scientists started to ask that the next versions of the software integrate non-spatial displays of data analytics, specifically linear regression and principle component analysis (PCA). This shifted the original objective of the VISTAS software as primarily a tool for visually exploring and communicating results to a tool that provided statistical output displayed alongside visualized landscape(s). Scientists in post-period interviews stated that these new statistical features were not aimed at helping them communicate with their stakeholders, an original intent of the project, but rather to better understand data output of their computational models and to present analytically verifiable results to scientist-colleagues. The desire for these statistical features emerged through many developer-scientist interactions in all-group meetings and multiple development iterations. It was only after the VISTAS software had adequately refined topographic and temporal visualization display features were the specifications for new data analytics features articulated and given high priority. As one senior scientist commented in the post-period:

> [With VISTAS] we can play a 3D video and wave our hands, look at those nutrients go, and [note that] here is a riparian buffer that seems to be reducing the flow of pollutant to a water body…and that's a big deal. But with the statistics it will allow us to actually quantify that…Being able to test the validity of what our model is doing against measured data, and statistically show whether it is valid or invalid, is really important [20].

On the other hand, some project participants were less enthusiastic about adding statistical features to VISTAS. One saw this development of statistical analytics as detracting from VISTAS' main purpose, which to this scientist was developing fly-through videos of gridded data: "Statistical stuff, PCA, doesn't matter to me. Can you more simply, more quickly, develop movies for me and my ASCII data? That's enough for me" [20]. For this scientist, VISTAS software development that further refined existing visualization features should take priority to assure the longevity and usability of the software for different user-groups.

4 Discussion

For our scientist collaborators, generalized GIS applications or visualization software commonly used within their domain did not enable them to efficiently and effectively conduct their science; there were, as one collaborator put it, simply "too many buttons" and generating many images over time or space from large data sets was too slow [20]. In other words, usability and performance of the software products were not tuned to their use. Overcoming this deficiency was challenging due to the complexity of spatiotemporal data that our collaborating scientist groups produced and consumed. We addressed this deficiency through the creation of the VISTAS specialized visualization software with novel functionality that included display of terrain-level time series data. Through multiple development iterations, initial features were extended beyond data visualization to include data analytics. The downside of developing this specialized software is the high cost of software development and the problem of long-term software maintenance.

We are aware that certain general-use software products are capable of producing similar visualizations. However, our collaborators, while familiar with these software programs, did not gravitate toward their use even though those applications have some obvious strengths including a higher probability of longevity, don't require time and resources of scientists to interact with developers, and are in widespread use by other science teams (usually by specialist users of that software within those teams). At the time this study was conducted, it appeared that no existing general-use platform achieved the ease-of-use and performance required by our collaborators, though we do not have a definitive explanation for why this is the case, it is clear that to be used on a day-by-day basis in the research process, a tool must be used directly by scientists themselves for data exploration and presentation, rather than by some resident expert in the visualization software who runs the visualization software at their request.

While we believe that developing specialized software for our users was justified as a scientific research project, this approach is not viable for most other domains facing similar issues with existing generalized software. One of our collaborators, a government scientist, told us that his agency had already dedicated substantial time and resources in his agency to developing the computational model, and few resources were available for developing specific data visualizations or visualization software, and, as an extension, data analytics of those visualizations. Participation in this NSF-funded research allowed this team to participate in co-producing software to meet their specific needs without agency resources beyond their time commitment to the co-production development.

One important lesson learned from our research is that existing tools in widespread use (e.g., ARC-GIS, Paraview) might be more widely useful to scientists if they had interfaces to easily build streamlined software that met domain-specific needs of scientists. This would necessitate potentially less effort and long-term risk than writing specialized tools. These ideas are not new: consider domain-specific data structures devised to make programming tasks more efficient and maintainable, or where domain-specific languages are devised as separately compilable or interpreted "overlays" to general-purpose programming languages such as Python to enable end-user programming or more efficient software development by professional programmers [14]. Most recently, Heer and others have created domain-specific visualization schemes for information visualization [8, 9, 10, 11]. If it became cost effective to specialize existing software products, domain user groups could define needs in much the same way we did.

A significant research question remains, however: even if generalized scientific visualization packages (e.g., ParaView or ArcGIS) were repurposed as domain-specific application drivers, how could user goals and needs, specifications, and use cases be elicited where potential users have little experience with scientific

visualization? For this, we have learned valuable lessons about applying co-production in software development.

One experienced software engineering expert on the project development team observed significant differences between agile methodology and the co-production process. Agile technology bases software development on user stories, which in turn depend on at least a first cut articulation of system goals and objectives, functional specifications, and use cases. Furthermore, Agile promotes a clear separation of responsibilities between defining the "what" and "how" of software to be developed. In situations such as ours, the "what" of the system (in Agile the province of the customer) is not easily articulated, primarily because the customer has little experience with the technology to be employed. In complex domains where users have little or no experience with the new technology and developers have little understanding of the domain, we believe that co-production processes could be employed effectively by an interdisciplinary team to develop usable proof of concept software.

Our users simply told us how they wanted to look at their data, not why they wanted to look at it in that way or for what purpose. In fact, it became clear from pre- and post- interviews that the scientists could not have stated the "why" or "what" of the system prior to development. Our scientist collaborators had only an intuition that visualization might be useful, and it was only after they had used the visualization tool in a variety of situations that they could articulate how the software would be used, or even how they would prioritize future development. Figure 3 shows a few of the typical and frequent interactions among the project team.

Figure 3: Knowledge co-production in action.

One can understand knowledge co-production within the context of a research system referred to as Mode 2 knowledge. Mode 2 knowledge is research that is produced "in the context of application" [12, pg. 740] through the efforts of collaborations across multiple disciplines [6]. This is in contrast to Mode 1 knowledge, produced in a single discipline context. The classic separation in software development between defining the "why" and "how" of software should be viewed as Mode 1 knowledge. We assert that 1) Mode 2 knowledge, produced in the context of application, is needed in situations such as we describe in this case study and 2) knowledge co-production is an effective way of developing Mode 2 knowledge.

Agile has been studied and improved by CHI researchers using the social science method grounded theory [2, 3]. We propose that software development involving new technologies for complex domains, as in this case could be effectively addressed by CHI researchers incorporating knowledge co-production methods into their skill set.

4 Conclusion

In this case study, we worked with a small group of ecologists and environmental scientists with complex models they used to study topographically complex terrain. Our collaborators found existing software over-complicated and difficult to learn, so agreed to work with us on developing domain-specific visualization software. We employed a co-production approach, ensuring that needs and expectations of domain experts were articulated and addressed at each step. Users conducted software testing, and developers integrated changing needs into the software as the scientist collaborators used resulting visualizations in their research and communication with stakeholders.

We learned that many domain experts were unwilling or unable to learn complicated generalized visualization software but were quickly able to apply and use specialized software not only to communicate their results, but also to learn more about the systems they were studying. The co-production approach allowed us to add features as users were able to articulate exactly what they needed. In this way, the VISTAS software remained flexible, quick, and reliable in producing visualizations used for multiple purposes by our collaborators. Using this approach for co-producing solutions may provide software developers a way to ensure that system goals and objectives are articulated correctly and that there are just enough bells and whistles visible to users as they begin exploring the use of a visualization tool. As users become more comfortable with a tool, they are likely to engage with more sophisticated features; but, if they're scared off by the time needed to learn how to apply a generalized software tool to their own work, they may miss out on the capacity of visualizations to explore complicated data in complex terrain.

We look forward to further research and development of domain-specific software for scientific visualization and the use of knowledge co-production methods for HCI studies in systems specification where potential users have little or no prior experience in the target technology.

ACKNOWLEDGMENTS

We gratefully acknowledge VISTAS funding from the U.S. National Science Foundation CISE-1637320, DBI-1062572, CISE-0917708, and thank other members of the VISTAS team, in particular environmental science collaborators Bob McKane, Allen Brookes, and John Bolte, and software developers Taylor Mutch and Ken Ferschweiler, as well as other collaborators Kirsten Winters, Mike Bailey and Susan Stafford.

REFERENCES

[1] Alex Abdelnour, Marc Stieglitz, Feifei Pan, and Robert McKane. 2011. Catchment hydrological responses to forest harvest amount and spatial pattern. *Water Resources Research*. 47, 9. DOI:https://doi.org/10.1029/2010WR010165.

[2] Steve Adolph, Wendy Hall, and Philippe Kruchten. 2011. Using grounded theory to study the experience of software development. Empirical Software Engineering 16, 4: 487–513.

[3] Steve Adolph, Philippe Kruchten, and Wendy Hall. 2012. Reconciling perspectives: A grounded theory of how people manage the process of software development. *Journal of Systems and Software* 85, 6: 1269–1286.

[4] Judith Cushing, Kirsten Winters, and Denise Lach. 2015. Software for scientists facing wicked problems lessons from the VISTAS project. *Proceedings of the 16th Annual International Conference on Digital Government Research*, 61–70.

[5] Environmental Protection Agency. 2016. VELMA Eco-hydrological Model, Version 2.0. ***https://www.epa.gov/sites/production/files/2016-09/documents/508_velma_fact_sheet_revised_6-21-16.pdf***. Accessed: 2018-06-09.

[6] Michael Gibbons. 1994. The New Production of Knowledge: The Dynamics of Science and Research in Contemporary Societies. SAGE.

[7] Jonathan J. Halama. 2015. Penumbra: A Spatiotemporal Shade-Irradiance Analysis Tool with External Model Integration for Landscape Assessment, Habitat Enhancement, and Water Quality Improvement. Ph.D. Dissertation. Oregon State University, Corvallis, Oregon. *https://ir.library.oregonstate.edu/concern/graduate_thesis_or_dissertations/kk91fn884*

[8] Jeffrey Heer, Joseph Hellerstein, and Sean Kandel. 2015. Predictive Interaction for Data Transformation. *CIDR* (2015).

[9] Jeffrey Heer. 2016. ACM Grace Murray Hopper Award. *https://awards.acm.org/award_winners/heer_1520709*. Accessed: 2018-06-29.

[10] Jeffrey Heer. 2017. Predictive Interaction. CMU HCII Seminar, March 2017. *https://hcii.cmu.edu/news/seminar/event/2017/03/predictive-interaction*. Accessed: 2018-06-29.

[11] Jeffrey Heer. 2018. Predictive Interaction. NWDS Annual Meeting, January 2018. *http://db.cs.washington.edu/events/database_day/2018/slides/heer.pdf*. Accessed: 2018-06-29.

[12] Laurens K. Hessels and Harro van Lente. 2008. Re-thinking new knowledge production: A literature review and a research agenda. Research Policy 37, 4: 740–760.

[13] Jason L. Jensen and Robert Rodgers. 2001. Cumulating the Intellectual Gold of Case Study Research. Public Administration Review 61, 2: 235–246.

[14] Richard B. Kieburtz, Laura McKinney, Jeffrey M. Bell, et al. 1996. A Software Engineering Experiment in Software Component Generation. *Proceedings of the 18th International Conference on Software Engineering*, IEEE Computer Society, 542–552.

[15] Bob McKane, Bradley Barnhart, Paul Pettus, et al. 2018. An integrated environmental and human systems modeling framework for Puget Sound restoration planning. Oregon Department Fish and Wildlife, Salem OR, April, 26, 2018.

[16] Alison M. Meadow, Daniel B. Ferguson, Zack Guido, Alexandra Horangic, Gigi Owen, and Tamara Wall. 2015. Moving toward the Deliberate Coproduction of Climate Science Knowledge. Weather, Climate, and Society 7, 2: 179–191.

[17] Kirsten M. Winters. 2015. Visualization in environmental science. Ph.D. Dissertation. Oregon State University, Corvallis, Oregon. *https://ir.library.oregonstate.edu/concern/graduate_thesis_or_dissertations/47429c578*

[18] Kirsten Winters, Judith Cushing, and Denise Lach. 2016. Designing visualization software for super-wicked problems. *Information Polity*. 21, 4: 399–409.

[19] Kirsten Winters, Denise Lach. and Judith Cushing, J.B. 2016. A conceptual model for characterizing the problem domain. *Information Visualization*. 15, 4: 301–311.

[20] Chad M. Zanocco. 2018. Co-producing visualization software for environmental science applications. Ph.D. Dissertation draft, October 2018 (unpublished). Oregon State University, Corvallis, Oregon.