EduViz – Agile Approach

EduViz does not lend itself well to the 'Waterfall' approach, or traditional software development. With data visualization in particular, many design questions cannot be answered until data exploration and preliminary visualization has been conducted. In complex visualizations, the data informs the design, and iterations of design will give better understanding of the dataset. As questions arise from the dataset, and further exploration is needed, the design of future visualizations will depend on discoveries made – e.g., if an interesting trend is discovered regarding a specific loan (perhaps the Pell Grant), then the visualization can focus more extensively on that area.

Waterfall also seems to lack cohesive and adaptable documentation, an area that is absolutely critical to modern software design. This is particularly relevant for projects that combine multiple computing technologies – in this case Python, R, D3, JavaScript, and more. Moving "up" the waterfall is frowned upon, but modern software development features extensive version control (e.g. GitHub) and allows users to traverse past versions and split code into different branches.

Given the involved nature of development among statisticians and front-end design, collaboration and evolving documentation is essential. The *Agile Manifesto* states that we value "working software over comprehensive documentation" This is important, however, our goal with this visualization project is to create a completely transparent analysis: one that can be reproduced by anybody. With tools such as *iPython Notebook*, creating working documentation from existing code becomes simple.

Enter Scrum

Because this project is highly dynamic, consists of modular software components, and needs to adapt quickly, Scrum is a sensible choice. The use of short-lived sprints, perhaps 2 weeks for this project, allows the developers to better meet the needs of the client. Data visualization of a large scale is iterative in nature, and daily standups with proper sprint planning will allow the entire team to direct the visualization in a better direction.

Agile seems to promote collaboration and regular meetings between developers and businesspeople, which stands to benefit the project enormously. Because computer scientists, statisticians and developers have more analytic minds, they may interpret visualizations very differently. Something that is obvious to a programmer or mathematician may be completely lost on the average person.

The goal of a successful visualization is to provide information in a manner that is accessible to as many people as possible. In this sense, getting regular feedback from the product owner and businesspeople will help the technical staff to cater their product to the needs of the client and eventually, readers.

Key Changes with Scrum

If this project were implemented with Scrum, the main changes would be in the **data analysis**, **visualization design,** and **beta build** phases. The original project plan calls for these phases to be executed sequentially. However, Scrum puts focus on generating working code quickly – with the waterfall approach, functional code would not emerge until **3/5/16**, whereas with Scrum it should emerge after the first sprint (early February). Given that responding to changes in needs and requirements is a focal point of Scrum, it

makes sense to forego the antiquated waterfall approach. This stands to benefit both the understanding of the dataset and the resulting visualization.

Schedule

Scrum calls for a major change in schedule:

Original plan:

- Phase 2: Data analysis – 1/27 to 2/19
- Phase 3: Visualization design – 2/20 to 3/4
- Phase 4: Beta Build – 3/5 to 3/27

Note that data analysis and visualization design preceded the first release of development. Scrum aims to implement this phase of **43 workdays** (excludes 9 Saturdays and Sundays) with sprints. This will entail the implementation of **four sprints**, of 10-11 days each. See the ***Sprints*** section for specific dates.

Team

Two major additions are required:

- **Product Owner –** A ProPublica staff member with experience in education reporting, data visualization, and some familiarity with statistical analysis. This person will serve to ensure that development is integrated and collaborative, rather than being a hand-off from developers to ProPublica.
- **Scrum Master –** A project management expert with experience in software development and interface design. This team member will direct the planning, conducting, and review of sprints in such a way that encourages collaboration and iterative design. The Scrum Master is expected to hold others accountable to their

deliverables and ensure that each sprint utilizes work-hours effectively, with a focus on functional code that can be presented to the customer.

Sprints

The tentative sprint schedule is as follows:

#	Date	Focus
1	1/27 – 2/12	Data analysis and exploration. Statisticians focus on parsing data while developers focus on creating loosely-designed shells for front-end website
2	2/15 – 2/26	Greater collaboration between statisticians and developers. Focus is on creating a functional prototype of a visualization by the end of this sprint.
3	2/29 – 3/11	Begin distributing visualizations for feedback and criticism from external readers. Begin interfacing more with the customer to ensure that their needs are being met. If an area of interest is discovered in the dataset, further tailor visualizations to reflect these observations.
4	2/14 – 3/25	Polish, improve, add features and ensure that cross-platform development (e.g. responsive design) is working properly.

Sprint planning and **reviews** will be conducted during the first workday and last workday of each sprint, respectively. Planning should allow the team to see an actionable vision for that sprint, with a sensible division of workload among members. Planning should also give members the chance to ask questions about the dataset and gear development towards particular discoveries from the dataset. Reviews should serve to highlight interesting findings from data, examine successes and failures from the sprint, and give an idea of what kind of visualization can be expected from the next sprint.