Add group member names for anyone who wants attribution in the blog post:

1. Ulrike Yang, Lawrence Livermore National Laboratory,
2. David Moulton, Los Alamos National Laboratory, (moulton@lanl.gov, jd-moulton)
3. Han Yong Wunrow, Sandia National Laboratory, (@hwunrow on github)
4. Benjamin Sims, Los Alamos National Laboratory
5. Jay Lofstead, Sandia National Laboratory (@gflofst on github)

Add a paragraph describing each person's software team background and experience. Emphasize experiences that inform your opinions about challenges, and how to improve software teams. Can add paragraphs in parallel :)

- David Moulton, lead IDEAS-Watersheds, design/developer of Amanzi-ATS C++, building team of teams approaches to software development
- Ulrike Yang, lead xSDK project, member hypre-project, computational mathematician, developer of mathematical libraries, common build of independently developed libraries, sustainability, testing of combined build, portability across various architectures
- Han Yong Wunrow, summer intern at SNL on the PSIP team, prior to that I was a fellow at the Institute of Health Metrics and Evaluation (IHME) working with research software engineers on global health research.
- Benjamin Sims, sociologist, background in studying scientific teams and infrastructures, recent work with scientific software development teams through the ECP project
- Jay Lofstead works on large scale data management, workflows, reproducibility, and many related areas

Discuss as a group the most promising cultural approaches you see as opportunities for scientific software teams. Summarize discussion in outline form.

- Ben: What should not be handled with a cultural approach?
    - Culture of Safety, culture for safety - but can be an excuse for inaction
    - As a sociologist
    - Have to have infrastructure and processes, and those have to work, because the best culture and intentions can't get the work done.
- Culture builds on and provides all the tools and processes needed to do their work.
- Culture is something that emerges without thinking, so without conscious efforts good and bad behaviors and attributes can arise.
- Jay: Scientific side sees software as a means to an end.
    - could we change the culture to see this as a valuable product
    - reusable, reproducible, move to software ecosystems is helping
    - Part of it is talking about the value of software and best software practices within the scientific community
- Dumping code in open source as opposed to a reusable, documented valuable component.
    - Include more with journal submission
    - DOIs for software, more JOSS (or similar entries)

    - Documentation - still a bit of a weak link as codes are not funded. Now a recommended policy in xSDK community policies. Best to be integrated into code.
    - Policies need to be in place, accepted and supported by the community
- Policies do tend to come top-down, and aspects of culture do come down from the top.
    - how people treat other, norms etc. from the top
    - bottom tends to be lower-level
        - can be more resistance when it comes from the top
        - is grass-roots better, Ulrike: depends on approach - with empathy and understanding or flavor of the day.
    - Grass-roots efforts to improve scientific software culture are starting to have some impact on upper management levels
- Make code contributions and impact visible.
    - Finding ways to give developers credit for contributions outside of publications
    - When reporting to managers, emphasis not just publications, but also other metrics of code development, use, value
- Emphasize respect within the team
    - Listening and hearing diverse views is important
    - Trust is an important element of respect; if team members trust each other they can work together and coordinate more effectively
- Cultural Inertia
    - Jay: example from Uber, so when top change the underlying folks at the bottom took a long to change
        - in RSE, project leads hold on to the “way they do things”.
        - e.g., someone by processing processes for release to code.
        - Someone has to lead by example and take the time to do it right. To make it stick, it still needs management support (top-down).
- Research Software Engineering department impacts on culture.
    - Opportunity to bring in cultural changes with little impact on existing people by supplementing labor to demonstrate impact
    - Offer experience-based seminars using peer groups as examples to show benefits of cultural changes and how they can be achieved and at what cost.
    - Contribute research publications based on studying process incorporation into various maturity groups.
    - Having dedicated RSE groups can make people in that role more visible and provide a good career path
    - In national labs, there are different cultures around titles and roles, some have started to emphasize RSE as a distinct thing, others allow people to do different roles but don’t emphasize distinct titles or organizations for RSE work. There are pluses and minuses to both approaches.
- Broader challenges with organizational culture, not specific to software - sexism, racism, abusive working environments, etc.
    - This is still a big issue in many industries, including software (e.g. Uber, game development)
    - Scientific computing seems much better compared to some fields/industries, but there

are still problems in some organizations
    - We shouldn't assume that these kinds of cultural problems are completely solved in the scientific software community

About 20 prior to the end of the session, around 1:40 pm CDT, try to reach consensus on 3 - 5 high-level cultural approaches your team identified

Approach 1: Organization culture from the lab director (or equivalent plays a strong role in the evolution of the organizational culture over time, but it takes bottom-up effort and a willingness to change (or be removed) in the middle to affect change. We need to have buy-in at both ends to hope to achieve lasting cultural change. As a community, we may overlook things that are endemic elsewhere, not making the progress we could since our situation is far better than other places. (examples: Uber, Activision/Blizzard)

Approach 2: Culture of diversity and respect. Listen to each other. Benefit from different views and backgrounds. Team building activities that involve informal discussions, and include active listening, and empathy development (e.g., small-team design challenges and games that have nothing to do with their regular jobs). These activities really help by adjusting people's perspectives and open their mind to relate to the other person's needs and perspective.

Approach 3: Increase cultural acceptance and support for Research Software Engineers (and those serving those roles) through a combination of top-down and bottom-up. Top-down examples include creation of departments and positions (with appropriate titles), Bottom-up includes acceptance/roles on teams, teams value of reproducibility and sustainability in their code base, advancing processes for recognition (DOIs, awards), etc.

Approach 4: Documentation is still a weak link as codes are not directly funded. Although, it is becoming a recommended policy in more research software teams. In order to increase usage of documentation, teams need to first be provided the resources (e.g., weekly protected time for documentation) so that they feel supported in incorporating documentation practices. A general approach that is applicable to other cultural approaches is to lead by example. If the team lead displays the benefits and quick wins that documentation can lead to, others may be more open to incorporate this practice into their daily schedule. Balancing publication and milestones versus documentation continues to be a challenge. Public recognition (e.g., awards, badges) of good documentation could be a potential solution.