Verification and Validation

Each breakout session had a group that discussed this topic. There was some overlap in the two groups. The group members' names appear at the end of this document. Across the two discussion groups, four common themes emerged.

1. Practices

- Practices vary widely. Some systems adhere to high standards of coding and coverage measurement; others are stretched threadbare. Convincing CSE folks of the importance of testing is difficult, especially if they believe in "science first" - not "modularization first"
- Even if we can't develop common tools for SW, how about common tools for domains, or across all domains (e.g. numerical accuracy estimators?)

2. Testing and Debugging difficulty

- Difficulty of writing assertions, don't know the requirements often (they are being discovered)
- Bugs are elusive and often requires expertise (domain or platform) to debug. However expert-time is scarce.
- The inability to reproduce results is a real worry (need to then find out what went wrong). Could be due to the associativity of numerical operations changing, or some component/library changing. To locate these changes is difficult. Seeking bit-precise results is extreme - can there be more liberal criteria that are applied?
- Daily regressions are important, but when regression fails, it is very difficult to root-cause. It is perhaps worth running more unit tests.
- Some explorations are in small teams; effort to do validation runs is high, and often unaffordable. UQ too unaffordable time-wise
- Some of the codes used in simulation are inherently complex (e.g. telescopes, interferometers). These complex systems with ~50 components and only 5 or so developers - faces inherent challenges.
- Nuggets of really important ideas have been discovered and incorporated into existing projects, but these results/ideas have not been written up as papers (the experts involved being really busy).

3. Newness of hardware/libraries

- For performance reasons, one must often run on modern parallel hardware; however, lack of incisive tools in that area (e.g. GPUs) is an impediment.
- Software is a utility. Want to reuse what worked in gaming. Unfortunately, most people in the lab have non-CS focus.

4. Difficulty of system infrastructure

- Complex dependencies in the software build; so end-to-end testing is difficult. At some point, wholesale code rewrite is essential - but many groups can't afford. Clean modular design methodologies can be evolved, but wide applicability is yet to be determined

Group Members (2 groups)

1. Robin Betz
2. Diego Caminha Barbosa de Oliveira
3. Upulee Kanewala
4. Laura Brattain
5. Karla Morris
6. Ganesh Gopalakrishnan

1. Hanna Remmel
2. Atsuhiro Ishiawa
3. Diego Caminha Barbosa de Oliveira
4. Peter Teuben
5. Anshu Dubey
6. Bryan Marker
7. Ganesh Gopalakrishnan