Software engineering and integration issues

Discussion Lead: Richard Brower

Scribe:

Participants:

- Babak Behzad
- George Biros
- Anshu Dubey
- Cynthia Gibas
- Eric Heien
- David Hudak
- Kevin Jorrisen
- George Karypis
- Yan Liu
- Frank Loffler
- David McQueen
- Marlon Pierce
- Ivan Rodero
- Jay Roloff
- Todd Tannenbaum
- Eric Van Wyk
- Julie Wernert

Questions:

Software Engineering

- How many projects benefit from engineering requirements flowing down from a larger organization (such as eclipse.org, apache.org)?
- What are some best practices in build, test, verification, validation of our software?
- What tooling is effective in promoting good software engineering practices?

Integration

- What are the integration models used in our SI2 projects? What are their pros and cons?
- What are attractive/effective integration/docking/affiliation points amongst the SI2 projects?
- How do we foster integration and affiliation with other software efforts
- What are good mechanisms to provide integration points for proprietary software?'
- What can we learn on how to push our integration to better levels? Are there high-level interfaces we should be considering? Service interfaces?

Notes:

- Benefits from engineering requirements of larger organizations:
    - Established large open source foundations provide key benefits:
        - Provide tools for aiding in the development, maintenance, build, and test process.
        - Benefit from their distribution channels.
        - Get exposed to developers and a larger community.
        - For new projects, these organizations provide some mentoring.
            - Incubator type of projects.
        - Having access to the resources of the large group (hardware, testing, experimental data for validation, etc.).
        - Benefit from the process for resolving conflicts (social engineering).
        -
    - However the benefits in term of getting access to software developers are somewhat limited when advanced technical domain knowledge is required in order to do the software development.
        - There is a need to have foster and organize open source communities that are vertical within each discipline area.

- Build:
    - Build as often as you can. Don't leave the build broken for a long period of time.
    - Provide incentives to developers for correct builds after commits.
    - Should build everything and not partial. It should go all the way to the end (packaging).
        - For packaging you should use tools to make makefiles to ensure that system dependent info.
    - Use a clean machine as possible.
        - Do not make assumptions about all the external packages that you may need.
        - Clean room build.
        - Cannot depend on third party repositories and have self-contained installation.
    - XSeed should provide more build and test facilities.
- Test
    - Static and dynamic test analysis. Static analysis for covering cases in which regression tests are not covered.
    - It should be easy to add new tests. Nice user interface.
    - Testing and V&V in the case of unreliable hardware.
- Verification & Validation
    - Model-checkers. Temporal state analysis.
        - Some of those tools do not even apply in scientific codes, no floats.
    - Different code bases are often used for verification?
    - Perform static analysis using tools that does find bugs. Static analysis is more testing than verification.
    - Validation is often compared against experimental results.

    - Hard to validate simulation type codes for which you do not have experimental results.
    - Validate against analytical solutions.
    - Compare the results against other tools.
- Integration
    - Standardization and defining interfaces.
    - Understanding the conceptual interfaces.
    - The hard work is the design of the abstract interface.
    - Need to educate and police people to adhere to a given standard.
    - Trial and error.

Best practice: security firedrill. How do you deal when a security whole is found. This applies to software that need to have root access. How do you disclose? There are many issues?