Code, Code Management and Online Interactions

General Comments

- Many of the "R&D" activities in the generic tools area are surveys & technology tracking
  - What are others doing? Can we re-use it?
  - What are best practices (inside/outside HEP)?
  - Need to validate claims of greatness
  - Small prototypes with candidate systems
- Policies and Recommendations
  - Start tight, open up if necessary, establish culture upfront
  - From our experience, tightening later is very difficult and generates resentment

Code Quality, Fault Tolerance, Online

- Determine requirements and potential solutions for code sharing between Online & Offline
  - Minimum coding standards, policies and enforcement “solutions”
  - Investigate approaches to error handling & fault tolerance engineering. Determine appropriate levels, cost & tradeoffs
  - “Grouping” of code and packages / packaging
    - Tools, dependency analysis, etc.
  - Different frameworks (e.g. Online/Offline)
    - How to make the code work with different frameworks?
    - How to make frameworks work together?

QA, code quality, standards

- Research and propose a QA strategy & implementation for SuperB
  - What is QA?
    - QA code? QA results? QA documentation? - All of them!
  - Tools (code metrics, automated analyzers, unit testing, global testing)?
  - What are others doing?
  - Code structuring to simplify QA?

Supporting Multiple Platforms?

- Determine Benefits and Costs of supporting multiple OSes/Compilers/Platforms
  - Why?
    - Added value, added cost?
    - What do we want? What do we have to do?
    - How many and which additional platform(s)?
  - Virtualization as a tool to reduce the number of platforms?

Code & Release Management

- Determine code and release management structure, policies and workflows
  - What are others doing?
  - Automated tools for code review
  - Policies (who can do what, training, tracking)
  - Workflow, investigate tools to implement
  - packages, grouping, dependencies
  - Alternative to users==developers?

Deployment & Installation

- Determine which tools to use for deploying and installing SuperB software
- Requirements:
  - Platform independent (mostly)
  - Tools supported over lifetime of SuperB ?
  - "Simple and usable", grid-compatible, grid-portable
  - Manage dependencies (including external deps)
  - Multiple versions on same machine at same time
    - Relocatable
  - No admin privs required
  - Scale from laptop (over Wifi) to 1000s of nodes

Documentation

- Determine recommendations for documentation
  - What are others doing?
  - Tools, policies & enforcement, recommendations, best practices?

Programming Languages

- Produce a recommendation for supported / permitted programming languages in SuperB
  - Investigate solutions to "the Fortran problem" (in concert with LHC & theory?)
  - Scripting languages - review - but we probably already know the answer

Adapting BaBar Code

- Propose approach to adapting BaBar (and other) code to SuperB standards
  - Refactoring vs. re-implementation
  - Best practices & tools
  - Organizational structure
    - central team vs. farming out to experts?
  - Collaborate with optimization effort!

Help!

- Survey work can start now
  - It should now since a lot of it affects basic infrastructure
  - Try to get it right from the beginning
- Subject-matter experts are welcome to help starting now!