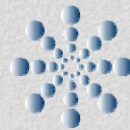


Open Health Tools/Eclipse Development Process

Dr. Brian M. Barry
Chief Executive Officer
Bedarra Research Labs





Overview

▪ Dual roots

- **Agile software development as practiced at OTI**
Evolved over 1985-2000
Creators of Smalltalk, VisualAge, IBM JVM, Eclipse
- **Open Source Development as practiced at Eclipse**
Commercial friendly open source
Openness, transparency, meritocracy

▪ Chinese wall enforces separation of concerns

- **Ecosystem**
Board of Stewards rules
OHT operations
Marketing & community building
Establish requirements
Charter projects, appoint leads
- **Open Source Development**
Meritocracy and peer review rules
Process is completely open
Issues resolved by voting
Developers decide plans and commitments
Board's only direct lever is power to create, terminate & appoint



OHT Technical Organization

- **Technical activities led by Chief Technology Officer (CTO)**
- **Management input will be provided by the Requirements, Architecture, and Planning Councils**
- **Management of Development Projects**
 - **Divided into Projects**
 - **Operate by open source rules: open, meritocracy, transparency, and peer review**
 - **Follows Eclipse-like open source development model**
 - Charter, Project Management Committee, contributor categories, etc.
 - **Three kinds of projects:**
 - Core – essential code for mission, hands on management
 - Sponsored – complementary to Core, led by sponsor, staff supports
 - Organic – community funded and led, minim support
 - **Strict IP and quality controls for Core and Sponsored**
 - **Organic projects on eHealthForge**



Development Resource Model

- **Four kinds of resources:**

- **Open Health IT technical staff**

- Small number of domain and IT experts

- **Open source contributors**

- In-kind resources provided by members

- Other volunteer contributors

- The largest source of resources

- **Funded open source projects**

- Small number of projects directly funded by Open Health IT

- Projects selected based on Open Health IT priorities plus technical and financial merit

- Funding provided and managed at the project level

- **Top Level Project resources**

- Resources provided by members and Open Health IT

- Architecture

- Development

- Project management

- Executive and operational sponsors



Clinical Council

- **Led by Chief Clinical Officer (CCO)**
- **Membership**
 - CCO
 - Peer selected members
 - CEO appointments (domain experts)
- **Responsibilities:**
 - Ensuring that the design, development and deployment of the Open Health Tools technology meets the needs of health professionals
 - Composed of health care and health informatics professionals
 - Liaison with Healthcare Community
- **Deliverables**
 - Quarterly Reports to the Community
 - Project Reviews
- **No Eclipse equivalent**



Requirements Council

- **Co-Led by CCO & CTO**
- **Membership**
 - **CCO & CTO**
 - **Peer selected members representing**
 - Consumers ??
 - Institutions & Standards Bodies ??
 - **CEO appointments (domain experts)**
- **Responsibilities:**
 - **Harvesting and analyzing requirements from existing public, private, and commercial health and standard organizations**
 - **Harvesting and analyzing requirements from academic, research, publications and data bases**
 - **Creation and management of requirements**
 - **Liaison with Healthcare Community,**
- **Deliverables**
 - **Statements of Requirement**
 - Use cases
 - Applicable standards
 - Acceptance criteria
 - Priorities
 - **Quarterly Reports to the Community**
- **Use Eclipse as a model for other details**



Architecture Council

- **Chaired by CTO**
- **Membership**
 - **CTO & CCO**
 - **Peer selected members representing**
Implementers ??
Institutions & Standards Bodies ??
 - **CEO appointments (domain experts)**
- **Responsibilities:**
 - **Creation and management of OHT Architecture**
 - **Creation and management of Road Map (jointly with Planning Council)**
 - **Identify and reconcile applicable standards**
- **Deliverables**
 - **Road Map, High Level Architecture**
 - **Quarterly Reports to the Community**
- **Re-use where possible, invent if necessary**
- **Use Eclipse as a model for other details**



Planning Council

- **Led by CTO**
- **Membership**
 - **CTO**
 - **Peer selected members representing**
Implementers ??
Project Management Committees
 - **CTO appointments (domain experts)**
- **Responsibilities**
 - **Creation and management of Development Process**
 - **Creation and management of Road Map (jointly with Architecture Council)**
 - **Review and monitor development projects**
- **Deliverables**
 - **Development Process**
 - **Road Map**
 - **Proposal and Project reviews**
 - **Quarterly Reports to the Community**
- **Use Eclipse as a model for other details**



Project Management

- **Charter sets out Mission and Scope**
 - **Approved by Board**
- **Projects are led by Project Lead and Project Management Committee (PMC)**
 - **Initial PMC is appointed**
 - **Thereafter members are elected unanimously**
 - **All must be Committers**
 - **Large projects can have subprojects**
- **Terminology**
 - **Contributor is someone who contributes code or other IP**
 - **Committer is Contributor with write access to the repository**
 - A position of trust that must be earned
 - Elected by their peers
 - Committers are expected to be very active developers
- **Technical issues decided by voting**
 - **Process is to cast -1, +1, 0 (no, yes, abstain)**
 - **At least 3 +1s and no -1s to pass**



Project Planning

- **Each (sub)project Lead must produce a plan for the release cycle**
 - the plan must be approved by the Committers
 - the plan is reviewed by the PMC
 - PMC provides feedback and advice
 - final approval rests with the Committers.
- **Plan must be published on the web site**
 - Plan must identify requirements and prioritizations
 - Plan must show date and content of the next release, including any major milestones
 - Plan must always be up to date
- **Master software repository must be accessible to all developers and Committers**
 - Off-line development is discouraged
 - Only Committers can check in code
- **All technical discussions are conducted using the development mailing lists**



Review Process

- **Three formal reviews**
 - **Creation Reviews to evaluate new proposals**
 - **Checkpoint Reviews to monitor project status**
 - **Release Reviews before every major or minor release**
- **Advisory Board formed for each review by CTO**
 - **Projects Pass or Fail**
 - **Notification within 24 hours**
 - **Written review required for Fail vote**
- **What does Fail mean?**
 - **Creation Reviews = no project**
 - **Checkpoint Reviews = termination recommendation**
 - **Release Reviews = may not release**
- **The Review Process provides the point of leverage to allow the CTO and technical staff to influence projects and ensure they conform to the Road Map, Architecture, etc.**



Eclipse Project Planning

- **PMC sets release themes to establish big picture**
 - community input
 - requirements council new source of input
- **Subproject teams define subproject plans**
 - Approximately 12-15 subproject teams
 - Plus Release Engineering team
 - Spread over half a dozen locations
- **PMC collates initial project plan draft**
 - Consists of senior technical leads
 - tradeoff: requirements vs. available resources
 - Features are classified: Committed, Proposed , Deferred
 - The final plan is the consensus view
 - The plan is a living document

Eclipse Project Management



- **PMC meets at least once a week**

- **All subproject leads and the PMC meet for a weekly planning call**
 - **status**
 - **planning**
 - **identification of cross-component issues**
 - **meeting notes posted to the developer mailing lists**

- **Dynamic teams are established for solving cross-component issues**
 - **one cross-component issue per dynamic team**
 - **members are key developers from all effected components**
 - **find, implement, and roll-out solution of the assigned cross component issue**
 - **represented in the weekly planning calls**

Basic Principles



- **Deliver on time, every time**
 - **Decisions in this release impact what we can do next release**
 - **Must preserve architectural integrity**

- **Deliver quality**
 - **innovation with continuity**
 - **need to have a solid foundation**
 - scalable
 - performant
 - stable



Continuous Integration

- **fully automated build process**
- **build quality verified by automatic unit tests**
- **staged builds**
 - **nightly builds**
discover integration problems between components
 - **(weekly) integration builds**
all automatic unit tests must be successful
good enough for our own use
 - **Milestone builds**
good enough for the community to use
- **reality: build failures occur**
 - **but protecting the integrity of the build is a critical success factor**
 - **rebuild to create acceptable integration, milestone builds**



Eclipse: where the time goes

- **release cycle 12-16 months**
 - **milestones - 9 months**
 - **endgame - 1-2 months**
 - **decompression - 1 month**

