Real Time Embedded Software Development Using Agile Technology

An Experience Report

Vincent Rivas

Joseph N Frisina

BAE SYSTEMS

Information and Electronic Systems Integration Inc

CNIR

Agile Development Concepts

- **Deliver customer valued and tested software early and continuously.**
  - Progress is measured by passing customer approved acceptance tests executed during demos
  - Customer participates in the planning effort
  - Automated continuous builds verify coded repository integrity
  - Test Driven Design is practiced
    - Emphasis on early requirements based test development
- **Evolving requirements are managed through continuous customer/developer collaboration.**
- **Design artifacts created through "automated" methods**
  - Design Documents
  - UML Models

Agile Development Concepts

- **Continuous attention to technical excellence and high quality design is crucial.**
  - Object Oriented Design Patterns and principles applied
  - Continuous refactoring
  - Fagan inspections applied at regular intervals
- **Development team monitors and adjusts behaviors at regular intervals to increase effectiveness.**
  - How are we doing according to our plan?
  - Team velocity estimation based on “Yesterday’s Weather”
- **Do the simplest thing that will work (pass tests) at any given time (KISS principle).**
  - Only include features/infrastructure when absolutely needed

Evolution of Agile Technology at BAE SYSTEMS

- **Maintainability issues arose on some programs**
  - Inadequate Unit Testing
  - "Big Bang" Integrations occurred introducing latent undetected bugs
  - Needed a better way to manage integration activities
- **Industry advances in Software Development were tapped into**
  - Design Patterns
  - OO Design Principles
  - Open Source Community offerings in tools and technology
  - Unit Testing/Regression Testing Frameworks
- **Agile Technology solution initiated through the Software Technology Insertion Review Board (STIRB) Initiative**
- **SW Developer Grass Roots movement created**
  - Received Management Level Support
  - Isolated Agile practices were adopted by a few developers with prior Agile experience
  - Those few developers provided a highly fertile ground for allowing other people to get involved and grow

Evolution of Agile Technology at BAE SYSTEMS

- **Agile Development Environment (ADE) Toolset developed to support a more comprehensive approach to Agile development**
- **ADE Toolset Technology Insertion Pilot completed and approved**
- **Pilot Program on a real time embedded software defined radio project Initiated and Completed**
- **ASCEND Methodology Developed based on Pilot Program Activities**
- **Deployed successfully to second real time embedded project**
  - Medium scale (6-8 engineers)
- **Large Scale Program Rollout in process**

ASCEND Methodology

- **A**gile **S**oftware **C**onfigurable **EN**gineering **D**evelopment environment
- Customized Development Methodology based on industry standard Agile practices
  - **Customizations needed to allow for classic waterfall artifacts and processes to be utilized while still remaining true to the Agile Manifesto**
- High level workflows created to guide development activities
- Tailoring/Specialization of some existing software processes is documented in the ASCEND Methodology document
  - **E.g. SCM, Software Planning, Code/Unit Test , Modeling**
- Customized training packages created to support the methodology
- Is a part of the existing overarching Software Engineering Process Handbook (SEPH) developed to support BAE Systems CMMI level 5 software organization.
- Served as the key process document used to roll out the ASCEND methodology beyond the pilot project.

Agile Development Environment (ADE) Tools

- **Key Toolset Attributes**
  - Overall toolset comprised of an integrated set of constituent tools
  - Exhibits a common web based interface
  - Customizable and Extensible
  - Supports ASCEND Workflow Processes

Agile Development Environment Tools

- **XPlanner - Agile Planning tool**
  - Supports the Agile “Planning Game”
    - Multiple levels of tracking , stories, tasks, projects
    - Tracks “perfect programming hours”
    - Provides for user friendly way of checking story progress to support iteration based replanning
- **Scarab - Issue Tracking tool**
  - No Special Requirements
- **CVS – Revision Management**
  - Support standard Branch Merge model
  - Integrate with repository usage monitor and repository integrity verifier tool

Agile Development Environment Tools

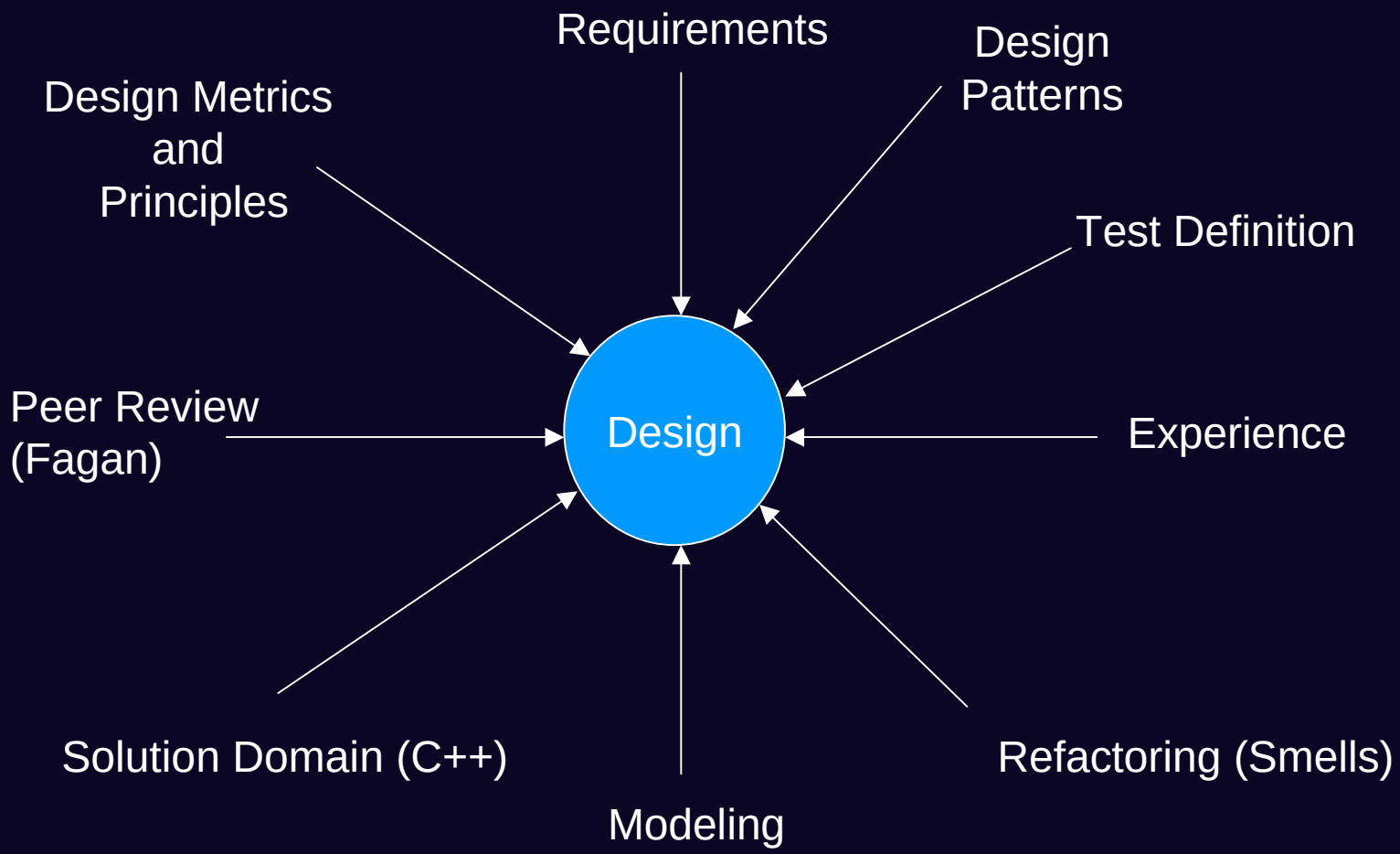
- **Tinderbox**
  - Support continuous monitoring of the code repository
    - Automated builds
    - 24/7 repository verification
      - ✓ Runs regression test suite
      - ✓ Verifies documentation is in place
      - ✓ Identifies “guilty” party (e.g. what code change broke the build or caused the regression test to fail?)
  - Repository status must be clearly visible to all developers
    - Color coded views of repository build status
    - Email failure notification reports

- **lcov – Code Coverage tool**
  - shows unexecuted lines of code
  - shows number hits per line

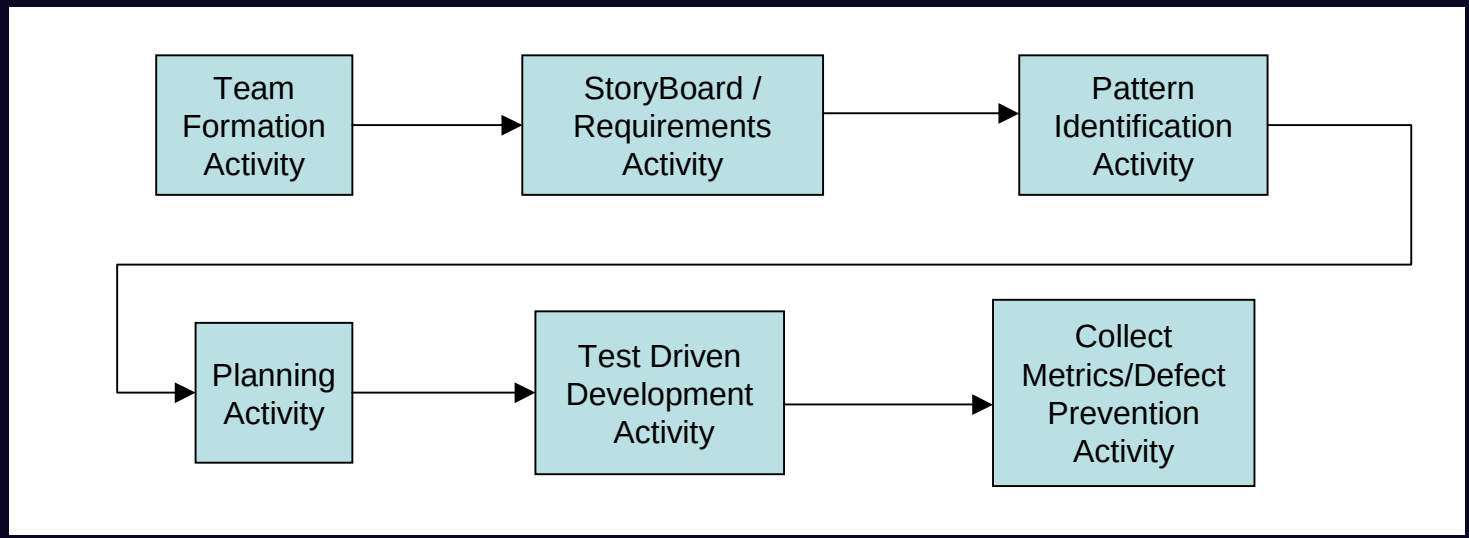
Agile Development Environment Tools

- **CPPUNIT – Unit Testing Framework**
  - Supports test suites and test case structure definition
  - Helper Macros available to check test results via assertions
  - Automatically aggregates test cases to create regression test suites
  - Test Cases coexist with operational code in the same repository
- **Doxygen**
  - Generates UML inheritance diagrams
  - Extracts comments from code
  - Has warning capability for missing comments
  - Is configurable
  - Provides output suitable for Mil Std documentation formats

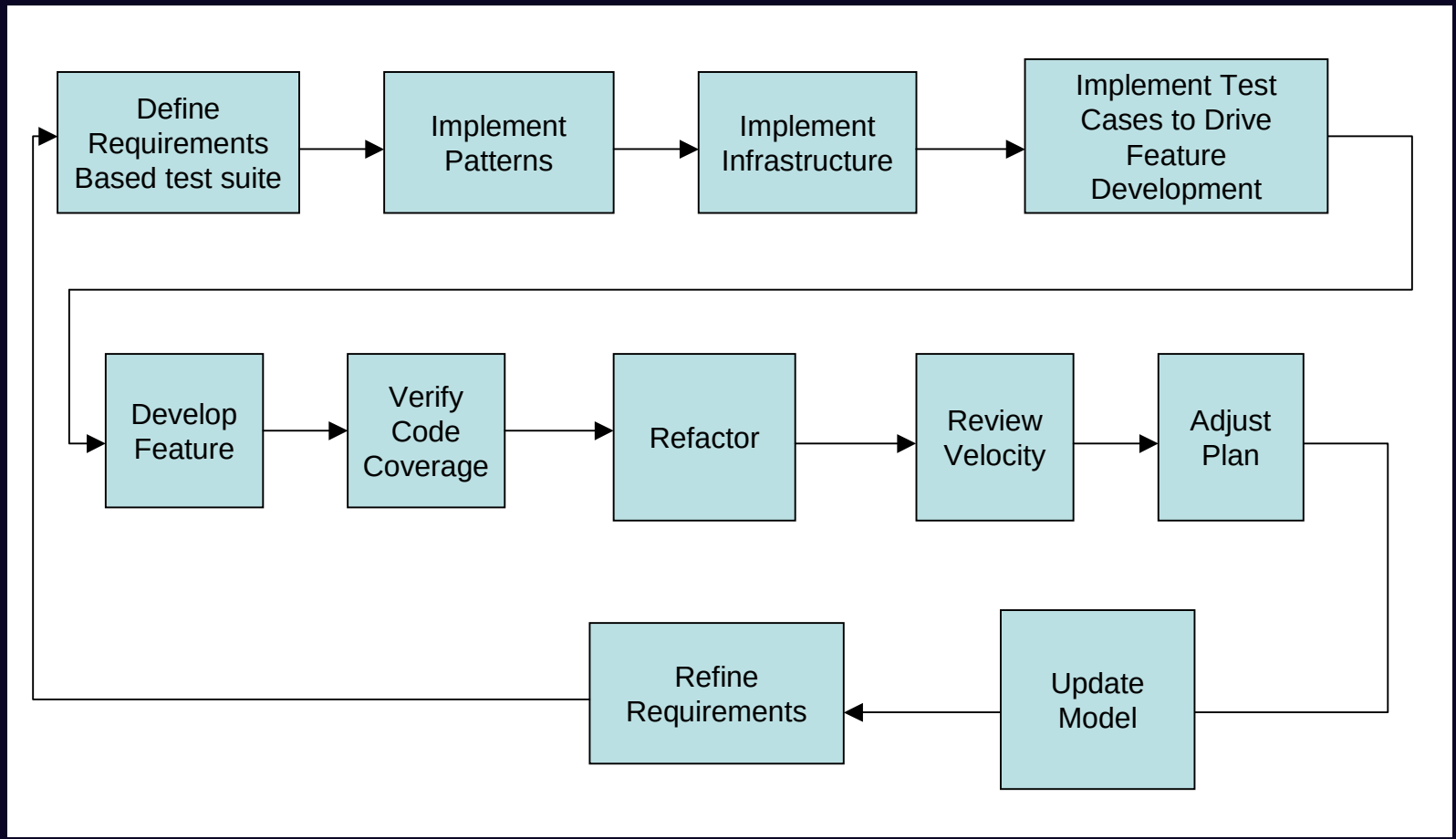
Major Contributors to Design



ASCEND High Level Workflow



ASCEND TDD Workflow



Design Patterns

- **Gang of Four**
  - Identified recurring object structure and interactions in industry.
  - Favor object composition over inheritance
- **Pattern based software development involves reusing standard or canonical forms of widely accepted and cataloged design patterns**
  - Patterns can be utilized in isolation or in conjunction with other patterns
  - Results in highly flexible and maintainable software
- **Domain specific Patterns**
  - BAE has establish sets of enterprise patterns
    - e.g. Link 16, Cryptos, etc.
  - SCA Developer's Kit
    - Encapsulates SCA patterns
    - e.g. extension object used in port definition
- **Waveforms designed using Pattern-Oriented SW Architecture**
  - Commonality/Variability Analysis used to find what varies and what is common in the design and encapsulate it
  - Heavy use of patterns/principles found in:
    - GoF (Gamma, et al.) , POSA I/II (Schmidt)

Test Driven Development (TDD)

- **Requirements based test cases developed very early in the development cycle.**
  - Involves writing test code for code that does not yet exist
  - Write unit tests against design
  - Tests exert force on design
    - Specify concept of class operation, interface, pre-post conditions.
  - Designing from a client's (class user's) perspective results in a more comprehensive class interface design.
- **Investigation of High Risk areas are addressed first using "spike solutions"**
- **Abstract Factory based spoofing approach used.**
  - Allows the design of code to be tested in the presence of a non-invasive test harness.

Test Driven Development (TDD)

- **Tests are continuously updated and run on both new and existing code/design**
  - Tests become part of the system's formal regression test suite
- **When tests are run against code, code coverage metrics are utilized.**
  - Drive unit test code coverage to 100%
  - Insures all requirements are satisfied and no untested design/code exists

Experience To Date

- **Pilot Program showed 18% improvement over highest coding rate on any BAE Systems CNIR Division project**
- **Defect Rate was unchanged**
  - As measured by Fagan Code Reviews
- **Running Tested Feature Metric collected**
  - The “mother of all” Agile metrics
  - Shows feature completion dates
- **Scalability to larger projects has not been proven yet**

Lessons Learned

- **Common misconceptions regarding Agile Technology were quickly dispelled by Agile pilot project "Hot Start"**
  - Progress of toolset development and early success of pilot program received management's attention and acceptance of Agile Methods
- **Formal customer presentations must be worded very carefully so as to not fuel any misconceptions about Agile.**
  - "I just reverse engineer the code to get the design ! "
- **Complete auto generation of documentation isn't there yet.**
  - High level design pattern UML diagrams created by hand
- **One Customer may not be enough.**
  - May need orthogonal views of the system
    - much like Fagan inspection approach

Lessons Learned

- **Iteration based target board demos provide quick, continuous, highly visible feedback to project stakeholders**

- **Agile software efforts can be tracked according to typical plan driven earned value method**
  - Basic approach is to define Agile “Releases” that correspond to the typical spiral stages (Requirements, Preliminary Design, Detailed Design, Code/Unit Test etc…)
  - Define classic Agile iterations within these “Releases”.
  - Can track according to the Agile Planning method
  - “Releases” are tracked using typical earned value methods

- **In house training crucial to technology insertion**
  - C++ Study Groups
  - Design Pattern Study Groups
  - C++ Standard Template Library (STL) courses
  - Fagan Inspection training
  - UML Training

Contacts

Name: Vincent Rivas

Tel: 973-305-2588

email: vincent.rivas@baesystems.com

Name: Joseph N Frisina

Tel: 973-305-2240

email: joseph.frisina@baesystems.com