# Fault Management Architecture Assessment

K. Barltrop (JPL/Caltech) and D. Garlan (CMU)

April 11, 2012

Copyright 2012, California Institute of Technology. Government sponsorship acknowledged.

# Topics

- Background
- Architecture Assessment
- NASA directive to develop Assessment Process
  - General Approach
  - Process Overview
  - FM Architectural Assessment Database

# Architecture Assessment

- (Introduce D. Garlan)

# Response to NASA Request

- Provide tools and methods for performing a technical assessment that can address three types of questions with respect to fault management:
  - 1. How well does a proposed solution fit a given mission and organization?
    <u>OR</u>
  - 2. How well do other existing solutions fit a given mission and organization?
    <u>OR</u>
  - 3. How well do individual features from existing solutions fit a given mission and organization?

# GENERAL APPROACH

# General Approach

1. Develop a process and structured data resource to help answer any of the three key fault management solution questions.
2. Implement an ongoing process and method to collect and maintain the data for past and future projects.
   - NASA FM Workshop serves as pilot for collecting data.
3. Implement a method to allow users to answer any of the three key fault management questions for their individual cases.
   - NASA FM Workshop serves as pilot for demonstrating the use of that method.

# General Approach

- Clarify what is included in fault management:
  - We define fault management as the aspects of a mission, such as practices, tools, staff, and on-board hardware and software features, that allow a mission to continue after faults or unexpected events.
  - We refer to a *fault management solution* as the chosen combination of practices, tools, and features.
  - To understand a particular fault management scenario, we consider:
    - Mission Characteristics
    - Heritage
    - Design Dimensions
    - Implementation Approach
    - Quality Attributes

# General Approach

- Role of the workshop:
  - As mentioned earlier, the Workshop will serve as a pilot program for collecting data and using it to make an assessment of a future mission concept in real time.
  - During the workshop, case study leads will lead discussions for several examples:
    - Present an overview of the mission or application.
    - Walk through the mission characteristics and the basis for the data entered.
    - Walk through heritage assessments and their bases.
    - Walk through design and implementation as well as notable quality outcomes and possible connections between those and the mission and heritage aspects.
  - During the workshop, the collected data will be used to perform an assessment of a hypothetical future manned mission to an asteroid.
  - An out-brief will summarize feedback from the participants about the activity.

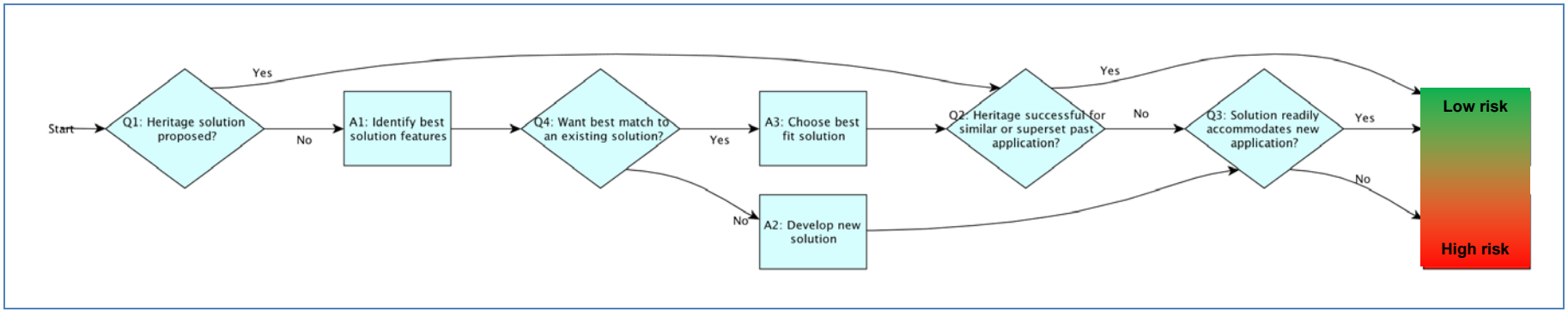
# ASSESSMENT PROCESS

# Assessment Process Overview

- The assessment process consists of two key elements:
  1. A top level process flow for examining the heritage risk story.
  2. An online database and reporting tool to ground the assessment in measureable data.

# Heritage Risk Assessment Process

- We begin with a heritage risk assessment covering at least these areas of a fault management solution:
  - Staff -- Engineering practices
  - Analyses & design tools -- Flight software
  - Flight hardware -- Mission design
- The figure below depicts the assessment flow.
  - Note that even a difficult-to-use solution, can be applied successfully to identical missions once it has been debugged sufficiently.
  - A project may also iterate this process across multiple aspects of the architecture and across multiple changes to the architectural approach.
  - Details for each box are now explained…



# Q1: Has a heritage solution been proposed for a new mission?

- Heritage should be considered in each of these areas:
  - Hardware
  - Staff
  - Practices
  - Software
  - Tools
  - Mission Design
- Heritage should be examined for multiple aspects of the entire fault management solution, such as high level software framework, system redundancy, local fault handling, etc…
- Breaking of heritage in even one area, such as by the introduction of new staff or tools, can introduce risk, especially if the consequences of change are not adequately identified and mitigated.

# Q2: Has this heritage been successfully used for past similar or enveloping applications?

- Points to consider:
  - Did past application use the same hardware, tools, people, software, mission features?
  - Did past application avoid cost and schedule overruns?
  - Did past application avoid near-miss situations related to design flaws?
  - Is it possible that the past applications got lucky in avoiding certain pitfalls?

# Q3: Does the proposed solution readily accommodate new applications?

- Points to consider:
  - How well has solution been adapted for new applications in the past?
  - Was the solution deliberately developed to support easy and reliable adaptation?

# Q3, Cont'd

- **Observation**: The solution that ultimately works is the one that provides "sufficient" correspondence between the things being managed and the solution's representation of those things.
  - You know you have sufficient correspondence when you have a system that works correctly.
- So how easily does a given solution allow one to achieve that "sufficient" correspondence?
  - The big challenge comes from determining what aspects (states, constraints, objectives, relationships) of the system and the world to represent and with what degree of fidelity.
  - As a matter of practice, we make choices about that correspondence by any of several methods such as trial and error from testing, by rules of thumb, by organizing states and modes for the system, and/or by modeling the physics of the system.

# Q3, cont'd

- How well does the solution allow the operator to implement a design in terms of the specific concepts of fault management?
  - Does the development environment provide useful references tied to fault management, such as the notion of errors, faults, and responses?

# Q4: Do we need a proposed best matching heritage solution for a new mission?

- Points to consider:
  - Which heritage solution has done well for similar missions?
  - Given the mission attributes, which solution best fulfills the quality priorities of the new mission?
  - We can filter and rank data from past missions to illustrate that matching.

# Q5: Are we looking for a new fault management solution?

- Points to consider:
  - What solution techniques and features (practices, system design, and tools) best fulfill the quality priorities for the mission?
  - What architectural solution provides that set of techniques and features?

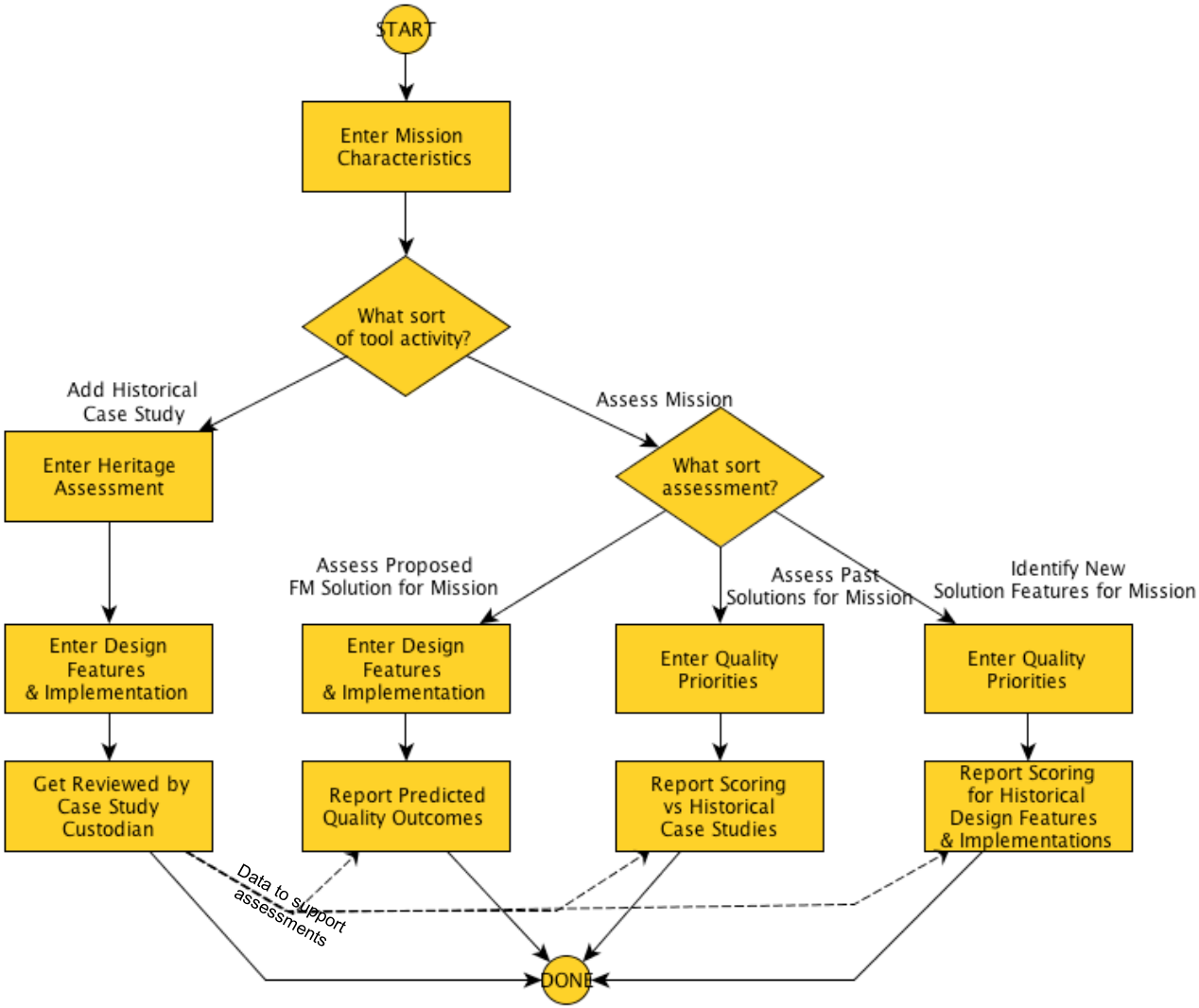
# FM ARCHITECTURE ASSESSMENT DATABASE

# A1: Identify best solution features

- Use the FM Architectural Assessment Database to help collect and apply FM arch information.

- Information captured is:
  - Mission Characteristics
  - Heritage
  - Design Dimensions
  - Implementation Approach
  - Quality Outcomes

# Flow for using the Database



# Mission Characteristics

- Mission features significantly affect the how well certain designs, tools, and practices will work.

| | APPLICATION CATEGORY | |
|---|---|---|
| COMPLEXITY | Diverse Activities | Major System Configurations |
| | | Operating Modes |
| | Dynamic Environment | Mission Phase Environments |
| | | Environmental Variation |
| | Configuration Complexity | System Interactions |
| | | Cross Strapping and Redundancy |
| | Critical Timing Windows | Performance Windows |
| | | Health and Safety Windows |
| | Large Comm Lag | Line of Sight Propagation Delay |
| | | Outage Delays |
| | | Network Propagation Delays |
| CRITICALITY | Risk Tolerance | Flight Crew Safety |
| | | Ground Bystander Safety |
| | | System Safety |
| | Investment | Stand Alone Investment |
| | | Infrastructure Investment |
| | Unique Opportunities | Science Opportunities |
| | | Prestige Opportunities |

# Design Dimensions

- Fault management largely evolved out of ad-hoc solutions to the question: *What should we do when something goes wrong?*
- An examination of fault management across domains and implementation approaches reveals recurring dimensions of designs.
- Often we're unaware of these because they're not explicitly called out in the design.

| DESIGN | |
|---|---|
| NAME | DIMENSION |
| **Knowledge** | |
| | Representing<br>Estimation |
| **Assessment** | |
| | Desired States<br>Discrepancies<br>Discrepancy Tolerance |
| **Response** | |
| | Strategy<br>Constraint Checking<br>Coordination<br>Influence<br>Mitigation<br>Character Control<br>Synchronization Granularit<br>Synchronization Control<br>Priority Accommodation |
| **Operations** | |
| | Visibility<br>Modification |

# Implementation Approaches

- Organizations introduce numerous implementation constraints that are often driven by cost phasing, trustworthiness, and history.
- This often requires an organization "pick its poison" when choosing an approach.

| MEANS | DEPLOYMENT | ORGANIZATION | ORGANIZATION DRIVER | PARTITIONING | THREADS | STANDARDIZATION | DESIGN SPECIFICATION |
|---|---|---|---|---|---|---|---|

# Quality Attributes

- Organizations have begun looking beyond the immediate requirements of a project, to consider other attributes that greatly affect the outcome of a project.
- We're can get stuck with unpleasant results from heritage solutions where these attributes were not considered.

| |
|---|
| *Analyzability* |
| *Appropriateness for Organization* |
| *Avoid Unnecessary Interruptions* |
| *Conceptual Applicability* |
| *Conceptual Integrity* |
| *Correctness* |
| *Cost For Development* |
| *Cost for Development Environment/Tools* |
| *Cost for Development Time and Testing* |
| *Cost for Operations* |
| *Cost For Repeated Work-Arounds* |
| *Cost for Training* |
| *Degrade Gracefully* |
| *Doesn't cause mission loss* |
| *Familiarity* |
| *Fault Coverage* |
| *Integrability* |
| *Interoperability* |
| *Modifiability during Development* |
| *Modifiability during Operations* |
| *Modifiability Mission-to-Mission* |
| *Modularity* |
| *Perceived Cost/Benefit* |
| *Preserve Resources and Opportunities* |
| *Reduce Recovery Time* |
| *Reliability* |
| *Reusability* |
| *Safety* |
| *Scalability* |
| *Testability* |
| *Thrustworthiness* |
| *Tolerate Modeling Errors* |
| *Usability/Operability* |