• I am Stephen LeTourneau from Sandia National Laboratories
• Sandia’s National Security Missions include:
  • Nuclear Weapons
  • Defense Systems & Assessments
  • Energy, Climate & Infrastructure Security
  • International, Homeland & Nuclear Security
• This presentation describes a method that combines practices from other industry methods
• Originally conceived as a way to “jumpstart” projects that are struggling to “get started”
• Helpful for agile projects to “find” their bearings and to “keep” their bearings
Organization is discipline-based, all software engineering disciplines
- IT consulting group for the labs
- Teams are formed using resources from all disciplines
  - Many are multi-disciplined
- Use a variety of development methodologies, including BUFD, RUP, Agile, ad hoc
• Begin with an analogy
• Have you ever been so focused on the step-by-step directions of a GPS device that you actually get lost?
• Consider this scenario
  • You are in a new town and trying to find your way around with only a GPS device as your guide.
  • Only know how to get from point A to point B with GPS?
    • Can’t recreate return trip (point B to point A) from memory?
  • Don’t know where you are because of “recalculating hell”?
    • Worse… don’t know how to get home?
  • You drive past your destination because the GPS device spoke too late
    • You were not paying attention to “outside the car”

• Sometimes you need just a Plain Old Map (POM)?
  • Map set the context to better understand the GPS-generated route
  • Can help you get you bearings
• What can happen if we fail to consider the *Big Picture* (i.e. *Start, Destination, Route in between*) in favor of just the *next step*?

• Mainstream news has many stories that show what can happen if we blindly take the *next step*. 
• What can happen if we fail to consider the Destination (i.e. Big Picture) in favor of just the next step?

• Mainstream news has many stories that show what can happen if we blindly take the next step.
• What can happen if we fail to consider the Destination (i.e. Big Picture) in favor of just the next step?

• Mainstream news has many stories that show what can happen if we blindly take the next step.
A GPS Analogy: “You Can’t Get There from Here”

- Too often we are overly-focused on just the next step of *The Journey*
  - “Turn Right Here”
  - “Turn Left in 1.2 miles”
  - “Make a legal U-Turn”

- Doing this, we can lose sight of *The Destination*

- A Plain Old Map can help set the context of your trip (start, route, destination, etc.)
- Helps set expectation of the entire trip
- Sets your bearings at the start of the trip and can keep your bearings during the trip

- I’m sure there are several reports of accidents caused by drivers trying to read a map while the vehicle is moving
  - Maps must be used wisely too
• With that analogy in mind, I want to tell you about a similar experience that I had on a project (before coming to Sandia)

• I was on an agile project and we were given our first user stories to implement
• During the launch, someone asked the project lead what the architecture looked like
  • They said something like “The architecture will emerge...”
• We implemented the stories and proudly demonstrated the functionality to the customers
• The customers gave their approval, provided feedback, and the team picked the next set of user stories to implement.
• Again, someone asked the project lead what the architecture was supposed to look like and they said something like “The architecture is still evolving...”.

Similar Story, Project Context:
“Turn Left in 1.2 Miles...”

- Customer demo
- They loved what they saw
- Got next set of user stories to implement
- Someone asked...
- What we were told: “The system is still evolving...”
• Repeat this pattern several iterations/sprints/cycles/etc.
• The project is well on its way when suddenly, it happened...

• We received new requirements that changed the fundamental design of the solution
  • E.g. Security, scalability, performance, etc.

• The project was forced to stop, and execute a couple “Refactoring Iterations”
• That’s when we realized that we had been overly-focused on the next leg of the journey
  • We had forgotten about the overall trip (start, route, destination)
• Sound Familiar?

• We needed a Map, to get our bearings!

• NOTE:
  • Refactoring the architecture may be a very valid thing to do
  • But, in order to refactor, one has to “consider the architecture” that the solution is being “refactoring to”
  • So why not “consider the architecture” up-front, while there are more options available, less re-work, etc.?
What is a Reasonable “Map” for an Agile Project?

- A map for an agile project provides a high-level perspective for one or more of the following:
  - Business Processes
  - Requirements
  - Architecture
  - Other

- In general, a map provides a high-level representation of some aspect of a celestial body
- Business processes tie together business roles, business activities, and various work products
  - Helps to describe the business domain
  - Shows how the product solution supports the business
- Requirements are high-level use cases (use case model survey)
  - Can also be an initial product backlog with high-level user stories and epics
  - Will focus on use cases for this presentation
- Candidate architecture(s) with tradeoffs that sets the context of how the product solution is to be developed
Product Analysis Jumpstart Method

- Draws from proven industry practices
  - Rational Unified Process – Vision, Use Case Model Survey
  - QA/ADD Workshop – Architecture Drivers, Candidate Architecture
  - Lean Six Sigma/Kaizen – Key Process Description

- Collaboration of disciplines
  - Requirements Engineer
  - Software Architect
  - Business Process Analyst

- Focuses on “Breadth Work”
- Executed as quickly as possible

- Uses practices from several different methods
  - Provides the “Map” (i.e. High-level description) of the product solution
  - Sets the context for the development iterations (e.g. the steps along the route)
  - Lets team know if they have (potentially) “Fallen in a ditch” along the way
  - Reminds all stakeholders of the “Final destination”
- Uses several disciplines to ensure a variety of perspectives
- Focuses on the “Breadth” of the product solution, not the details
- Initially meant to “Jumpstart” product development efforts that were having difficulty getting started
• Designed to be a series of working sessions (workshops) where all stakeholders participate
• Facilitated by practitioners in the various disciplines
  • Requirements Analyst
  • Business Process Analyst/Engineer
  • Software Architect
• A Product Analysis Jumpstart is a concentrated, workshop-focused effort:
  • Led by a team of experienced individuals
  • to produce project, requirements, and architecture breadth documentation
  • for the purpose of guiding and facilitating a project team in their efforts to develop an information system solution for a business problem or opportunity.
A Closer Look: The Business Process

- Use a light-weight Value-Stream/Kaizen event to capture and document the existing business processes
- Use RUP workflows to:
  - Define the scope of the production solution
  - Identify the Needs and Features
    - Identify any Risks
- Is the Business Process well known?
  - No: Develop and document it
  - Yes: Is the business process documented?
    - Yes: Present it and get agreement among all stakeholders
    - No: Document it now
- Define the “problem space”
  - What are the Needs & Features of the product solution?
  - What is in-scope/out-of-scope?
- Capture this in a “Vision” of the production solution
- Optionally, capture any Risks that were identified
• Tailor your QAW so that it is “just enough” ceremony to identify the Architecture Drivers
• What are the stakeholder gripes/concerns/goals that you hear them articulate?
  • Can you turn that into a Quality Attribute Scenario?
• Prioritize the QA Scenarios
  • What are the primary drivers of the proposed product solution?
• Capture this in a (RUP) “Supplementary Specification” of the product solution
• Map the Architecture Drivers to the Needs and Features list
• Optionally, capture any Risks that were identified
Tailor your Requirements Workshop so that it is focused on the “Breadth” of the requirements
  • Identify as many Use Cases as possible
  • What are the stakeholder gripes/concerns/etc. that you hear?
    • Can you turn that into a Use Case Scenario?
  • What are the stakeholder “goals”?
    • Can you turn that into a Use Case Scenario?
  • Prioritize the Requirements
    • What are the “must have” requirements of the proposed product solution?
    • What are the “primary use cases”?
  • Capture this all in a (RUP) “Use Case Model Survey” of the production solution
  • Map the Requirements to the Needs and Features list in the traceability matrix
  • Optionally, capture any Risks that were identified
  • Optionally, capture any domain-specific terms in an initial Glossary
A Closer Look: Close-Out

- Prioritize Risks
- Communicate back everything that was captured and get validation from stakeholders
- Develop Project Proposal or Plan
  - Cost Estimates
  - Develop initial mitigation for top risks
  - Resource Profile

- Prioritize any risks that were identified
- Present what was captured/developed in the workshop
- Optionally, develop a project proposal or project plan based on breadth work
  - Resource profile needed
  - Cost & Schedule
  - Develop risk mitigations for top risks
What are Some Benefits of this Approach?

- Agreement on *what* is being developed
- Common understanding on *why* it is being developed
- Identification of major components of the system

- Development teams need to know what is being developed and how it will be used in the context of the customer’s workflows
- Customers need to know what is being developed and what it will look like
- Agreement on a Candidate Architecture
  - Major architecture components are identified
  - Sets “project bearings”
  - Reduces need for “Refactoring Sprints/Iterations”
What are Some Challenges?

- Keeping the scope/scale of this effort to *just enough*
  - Don’t revert back to *big up-front design*
- Staying focused on the outcomes
- Getting customer participation in each workshop

- BUFD Baggage
- Avoid “deep dives” into implementation details
- Losing focus on desired outcomes
  - Common Vision
  - Candidate Architecture(s)
  - How the product solution fits into the Business Processes
- Agile purists may reject these activities as unnecessary or to academic
  - Just keep telling yourself: “*Recalculating...*”
What does it Look like in Practice?

- Depends on several factors
  - The size of the project
  - The diversity of the stakeholders
  - The availability of big-picture thinkers from the business domain
- Remember, this is an up-front activity conducted as fast as possible
  - Consider a “Sprint/Iteration 0”
  - Requires intense focus and fast pace to make best use of project resources

- Factors
  - The size of the project will impact the amount of time/effort spent on creating the various maps
  - Each stakeholder will bring their “Need” to the table
    - All may have very different “Goals”
  - Customers must see themselves in the vision and agree on how the product solution helps them achieve their goals
- Remember, this is an up-front activity conducted as fast as possible
- Stay focused on the “breadth” of the product solution
- Collaboration!
  - Co-located
  - JAD-style
  - Team Room
- Sprint/Iteration 0
  - Iterations are typically short durations so it should not be a huge “hit” to the project
  - “Demo” the various “Maps” to the stakeholders as part of the Retrospective
- Really have to “lay some rubber”
  - Keep from diving into too much detail
• The Disciplined Agile Delivery method spends energy “up-front” ensuring all stakeholders are in agreement before launching into iterative development cycles
• Use “some” approach to consider the big picture before launching in
I love agile projects
  • Need to make sure we know where we are headed and why
  • *Still need to consider the Big Picture*
  • Maps can help address the problems that cause the “Refactoring Iteration”
    • “What” you use may vary
  • If you are thinking of just sprinting into development without considering the big picture, then remember one thing...
    • “Recalculating...”

Conclusions

- Agile development is great
  - Want to make sure we build the right system
- Want to be careful not to *throw out the baby with the bath water*
- Maps can keep you on-course or get you back on-course if you have gone off
- Still want to *sprint* right into development? Remember...

“Recalculating...”
Thank you!

- Questions?

- Thank you for your participation

  Stephen T. LeTourneau
  Sandia National Laboratories
  stletou@sandia.gov