BEDIFFERENT

ACE 2012 INTERNATIONAL

ACE 2012
INTERNATIONAL

Aras Innovator Deployment Methodology

Strategies for Success

Agenda

- Methodology Overview
- Getting Organized
- Resources
- Getting Started
- Breaking down a project
- Some Tips
- How things get sideways
- Final thoughts

The Aras Methodology

- **Iterative design not a waterfall approach**
  - Based on principles from IBM's Rational Unified Process
  - Incorporates Agile development principles
- **Designed to use the "Small Win" approach**
- **Take a well defined problem(s) and implement with less risk and a higher degree of confidence**
- **Implement a series of Production releases that comprise the complete solution**
- **Each release provides value to the business**
- **Scope each release to be completed in 90 days or less**

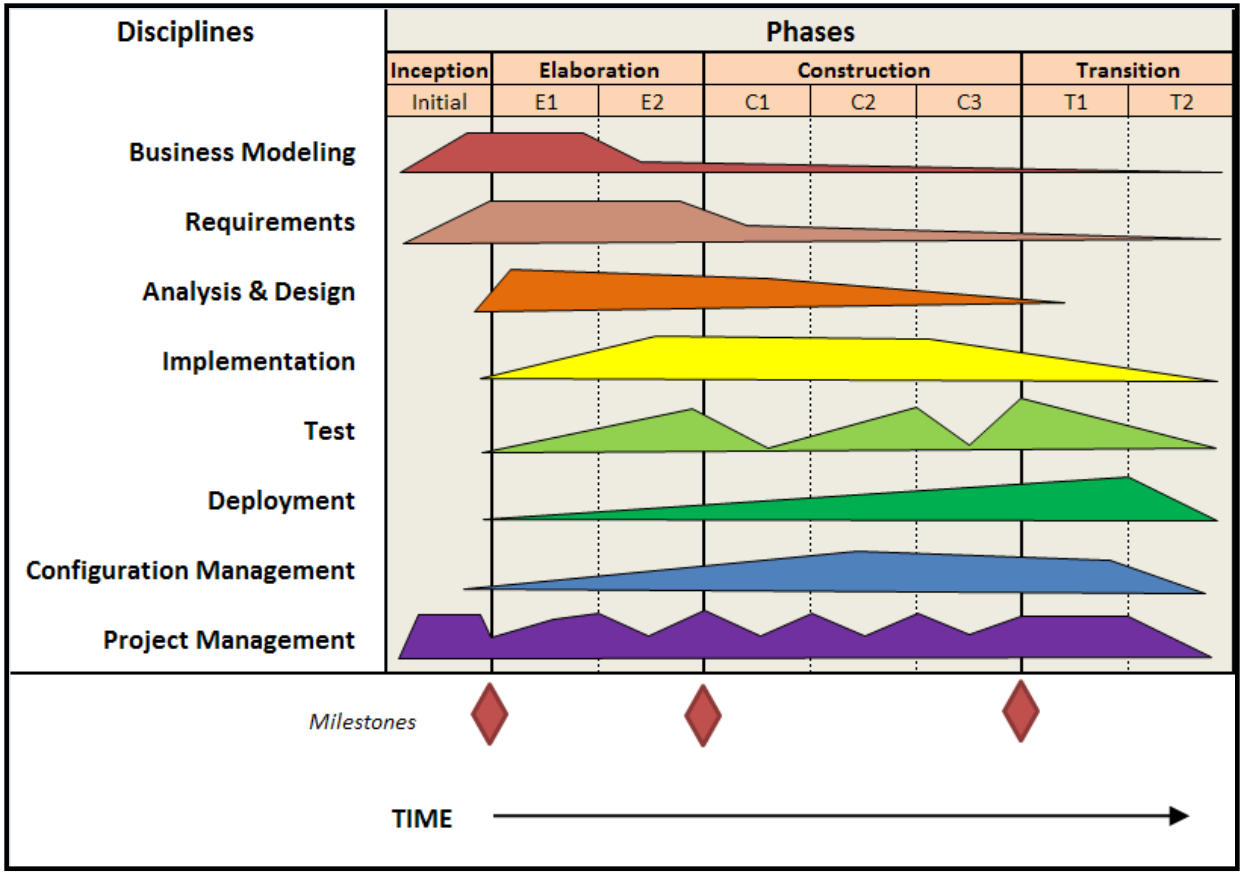
A little about the process

- **Designed to be an iterative approach to solution development**
- **Well defined but not a concrete prescriptive process**
- **Provides an adaptable implementation process framework**
  - Intended to be tailored by selecting the elements that meet your needs
- **Based on Best Practices of software development**
  - Develop Iteratively, risk is the primary iteration driver
  - Manage Requirements and scope
  - Continuously verify quality
  - Control Changes
  - Manage customization

All this leads to improved quality and better predictability

Iterative Development

The Aras Approach



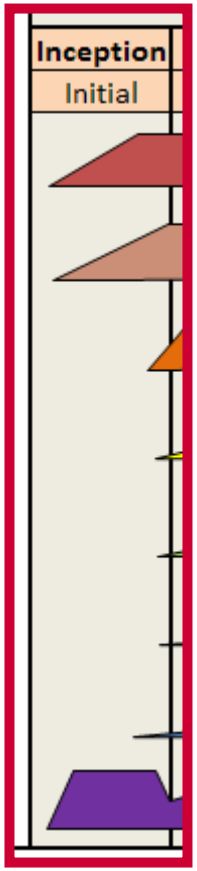
Business value is delivered incrementally in time-boxed cross-discipline iterations

BREAKING DOWN YOUR PROJECT

Manageable Phases

Inception Phase

Decide what to do

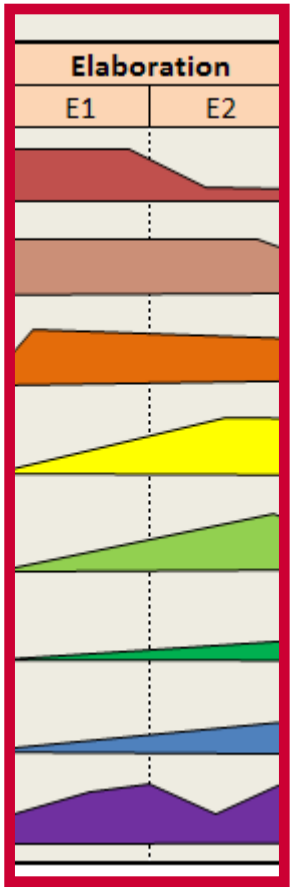


- **Primary Objectives:**
  - Scope the system adequately
  - Validate initial costing and budget estimates
- **Activities**
  - Establish Business Case and high level Requirements
  - High level Use Cases
  - Project Plan and Initial Risk Assessment
- **Milestones:**
  - Stakeholder concurrence on scope, cost, and risk
  - Initial requirements defined (not a lot of detail)
  - Achieve Project Plan concurrence
  - Plan is realistic
  - Business case makes sense

Elaboration Phase

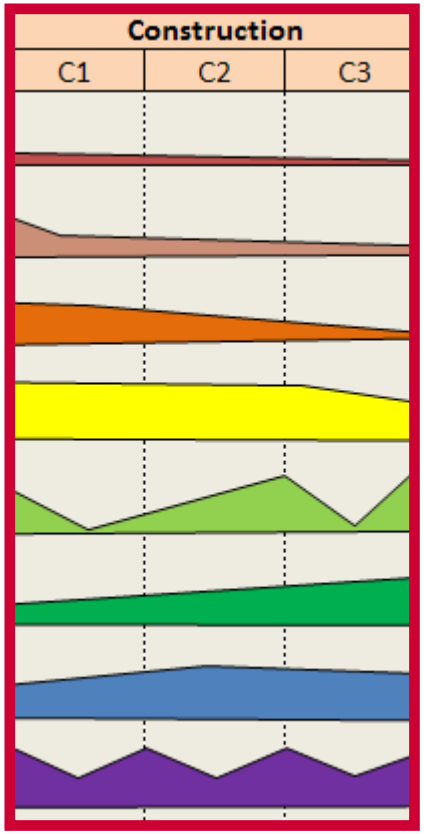
Plan the details

- **Primary Objectives:**
  - Improve requirements and validate
  - End to End skeleton
- **Activities**
  - Requirements elaboration
  - Use Cases
  - Design Workshops
  - User validation
- **Milestones:**
  - Use Cases identified and 80% complete
  - Requirements are understood & documented
  - Project Plan refined, cost & risk are managed
  - Detailed plans for iterations are in place



Construction Phase

Build it

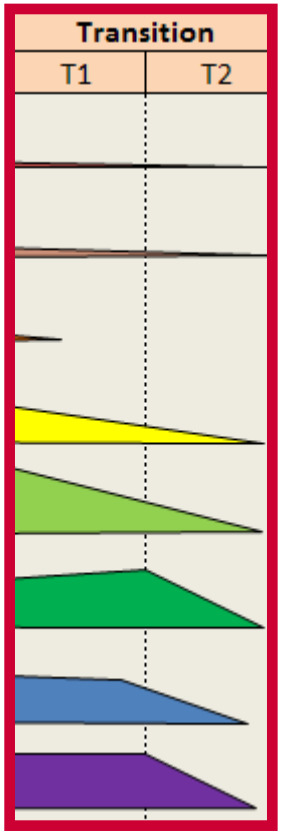


- **Primary Objectives:**
  - Build the system end to end
- **Activities**
  - Visual prototypes
  - Behavioral Prototypes
  - Data Migration
  - Unit test
- **Milestones:**
  - Solution is acceptable to deploy
  - Project Plan refined, cost & risk are managed
  - Additional Iterations are planned

Transition Phase

Deploy it

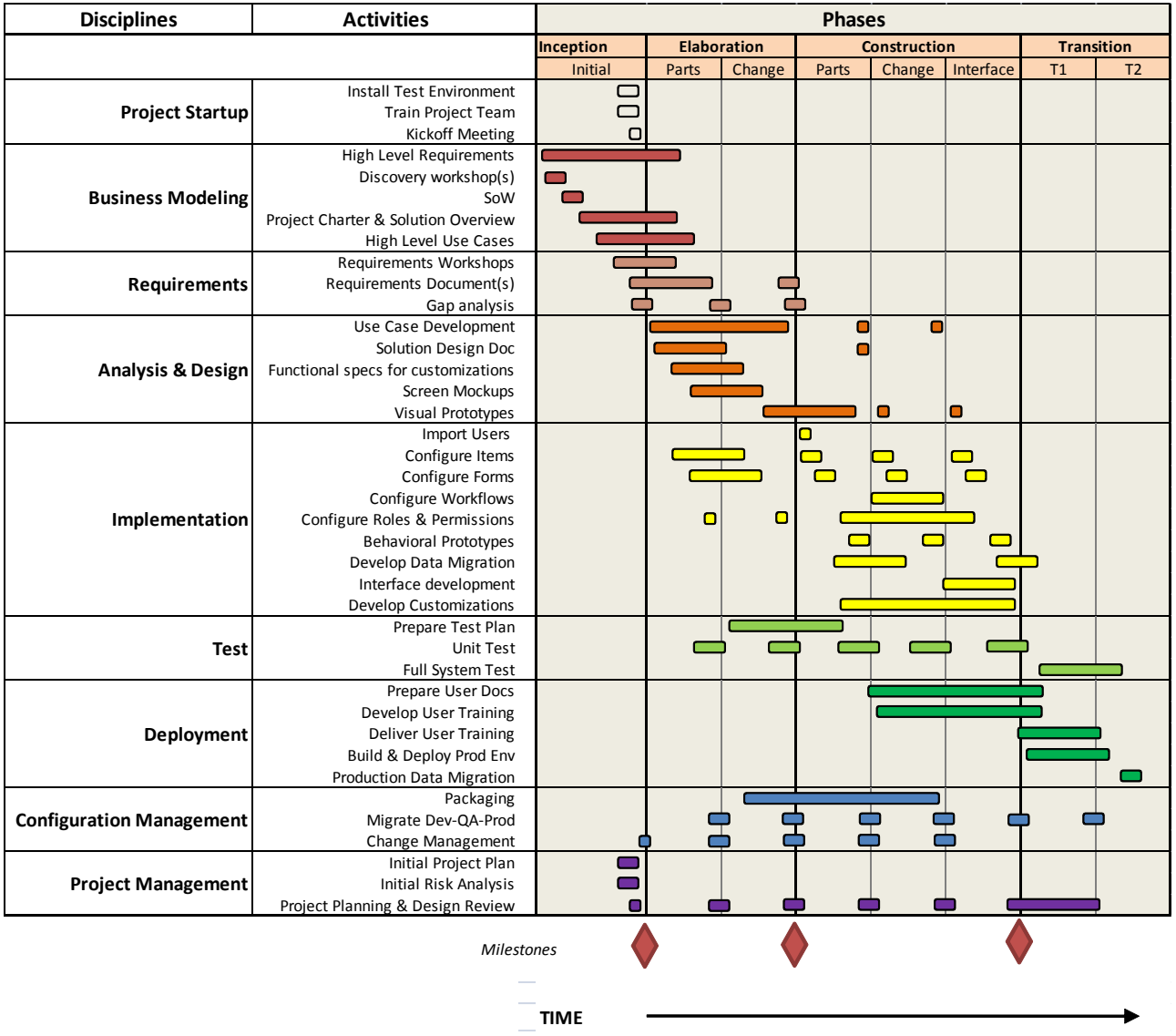
- **Primary Objectives:**
  - Move the system to production
  - Training & fine tuning
- **Activities**
  - End user training
  - Documentation
  - Full system test (production environment)
- **Milestones:**
  - Solution & documentation is ready to deploy
  - Stakeholders are near ready to deploy
  - System is functional in production



OK, GREAT !

Now how does this apply to Aras Innovator

A Complete Look



Implementation Best Practices

- **Develop iteratively**
  - Its always best to know all the requirements up front but this is not reality
  - Don’t spend an inordinate amount of time upfront on requirements and design w/o user validation
  - Model a little, test a little & repeat
  - This is a big strong point or Aras Innovator
- **Manage Requirements**
  - Always keep in mind the requirements set by the users
- **Model visually**
  - Use diagrams and mock ups
  - Its what Aras Innovator does really well

Implementation Best Practices

- **Use Components**
  - Break down large projects into manageable pieces
  - Item reuse is a big plus
  - Again, this is what Aras Innovator does really well
- **Verify Quality**
  - Always make user testing a major part of the project
- **Control Change**
  - Ensure changes are synchronized and verified constantly
  - Use a managed process for implementing iterations

Making Iterations work

Things to think about

- **Work does not proceed entirely serial**
  - This is not a waterfall approach
- **Work is done in a serial fashion but you don't need to finalize a discipline before moving on to the next**
  - Address some requirements, analyze, develop and go back
- **Iterations should be planned according to risk**
  - Higher priority risks first
- **Near term iterations are planned in more detail**
  - Longer term items may change in scope, etc
  - Doesn't mean that longer term items are not planned

GETTING STARTED

Who, What & How

Resources

Who do you need

Role	Responsibility
Project Manager	• Direct Implementation resources • Manage project schedules • Track Status • Resolve conflicts and issues
Business Process Owner(s)	• Provide project priorities and objectives • Direct participation of resources • Resolve business process issues
Subject Matter Experts	• Communicate current process • Provide information details • Support user community during rollout
I.T. System Support	• Support site infrastructure • Extract legacy data • Provide technical expertise
Technical Resources	• Configure application • Develop customizations • Provide technical expertise

How to approach it

Divide and Conquer

- **Divide implementation into phases**
  - Preferably phases that provide business values and can be deployed independently
  - Build a plan for each phase
  - Choose goals for each phase and make sure you address items with importance or high risk sooner
- **Divide phases by related business processes**
  - Build high level use cases for these business processes
  - Its OK if use cases overlap
- **Break down high level use cases**
  - Detail each use case from a user perspective
  - These will likely be developed iteratively

Getting Started

- **Develop a Project Charter if you need funding**
  - Lets management know what you will do and will not do
  - Defines a goal and an endpoint to measure success
  - There are plenty of examples on the web
- **Develop a Project Plan**
  - Lets resources know what is expected of them
  - Sets the schedule for the project
- **Get Trained**
  - The team needs to understand Aras Innovator
- **Review the standard Aras Innovator Solutions**
  - Required for effective gap analysis

Getting It Done

Elaboration - The Key Points

- **Make sure you achieve concurrence on requirements**
  - This provides mutual understanding and prevents individual interpretations of the project
- **Document use cases and requirements**
  - These will be used to provide context to the team
  - Will be leveraged later in the project for other activities
- **Validate this with Subject Matter Experts & End Users**
  - Do NOT underestimate the IMPORTANCE of this

Getting It Done

Construction - The Key Points

- **Visual Prototypes**
  - This is just building things in Aras Innovator
  - Used to solicit user feedback
  - May cause you to revisit use cases and specifications
  - Should not include automation
  - Don’t worry about getting it 100% right
- **Behavioral Prototype**
  - Adds automation to the Visual Prototype
  - Will likely cause you to revisit use cases and specifications
  - Will introduce changes to the Visual prototype
  - Includes building interfaces and integrations

Getting It Done

Transition – The Key Points

- **Test Plans**
  - These are based on use cases
- **Testing and Validation**
  - Unit testing is important for managing iterations
  - Conference Room Pilots are good for user validation
  - Full system test
- **Training**
  - Don’t under estimate the time it takes to develop materials or train end users
- **Production Cutover**
  - Plan, Plan, Plan, - There WILL be external influences

How Things Get Sideways

- Lack of understanding of requirements or agreement on requirements with end users
- Lack of understanding of the standard out-of-the-box Aras Innovator solutions
- Lack of Training
- Understanding the impact of change
- Biting off too much at one time!

Our Recommendations

- **Training is a MUST**
- **Engage Aras or a partner for jump start activities**
  - Project phasing & planning
  - Requirements review
  - Architectural guidance
  - Leverage experience to recommend best practices or how approached in similar scenarios
  - Use case development
  - Initial architectural design review
  - Periodic reviews and questions
- **Solve a real problem and move on to the next**

Final Thoughts

Dos and Don’ts

- **DO**
  - Create visual prototypes and get user validation before developing any method code
  - Develop accurate Use Cases and keep them up to date
    - They will save you time down the road !!
  - Look for “Small Wins” that provide business value
- **DON’T**
  - Spend a significant amount of time developing specs w/o prototyping the solution
  - Worry about not getting 100% of the detailed requirements up front… because you can’t!

THANK YOU

Questions?