

Practical SysML Applications: A Method to Describe the Problem Space

Ray Jorgensen

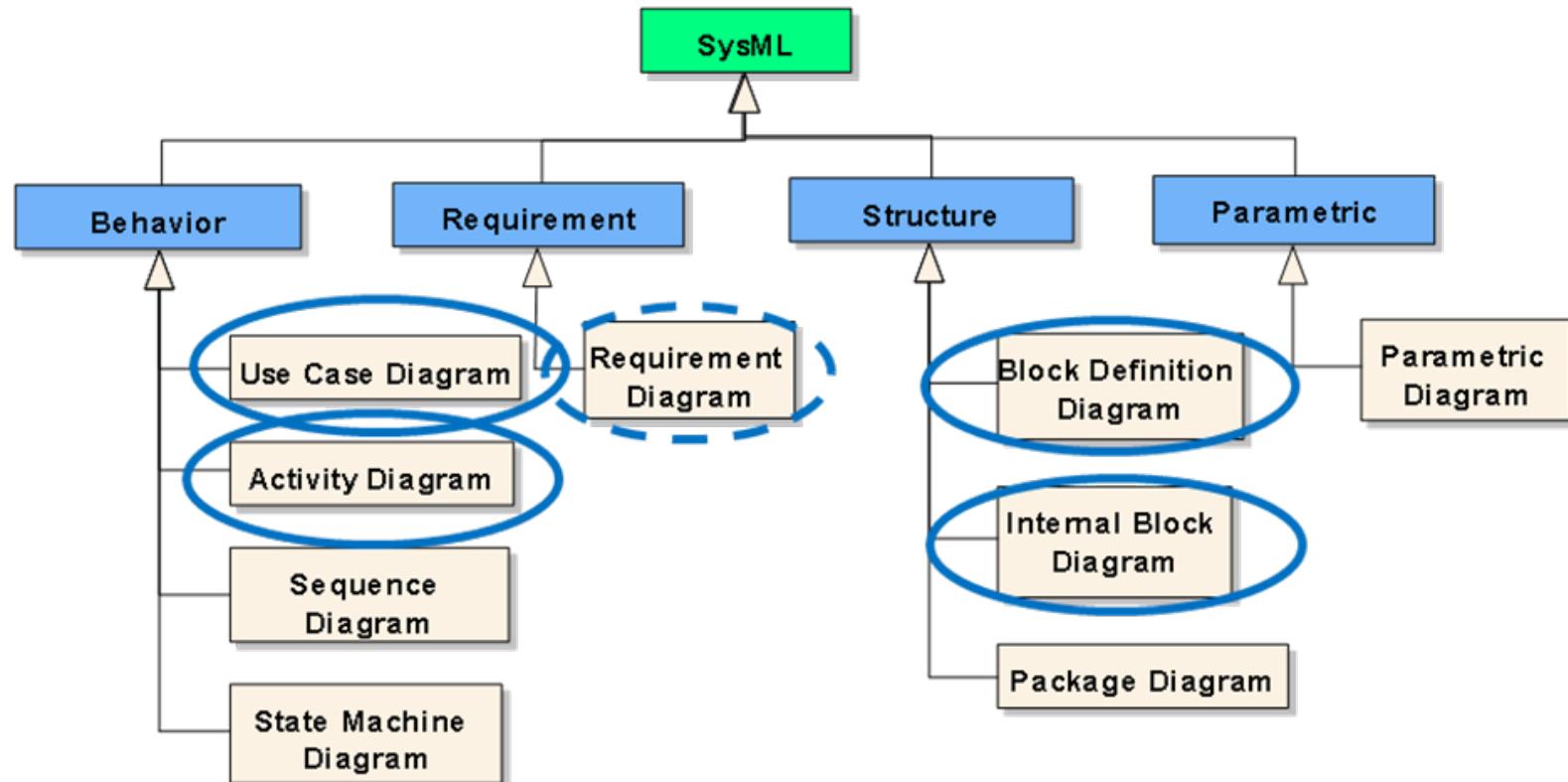
David Lempia



Problem Space

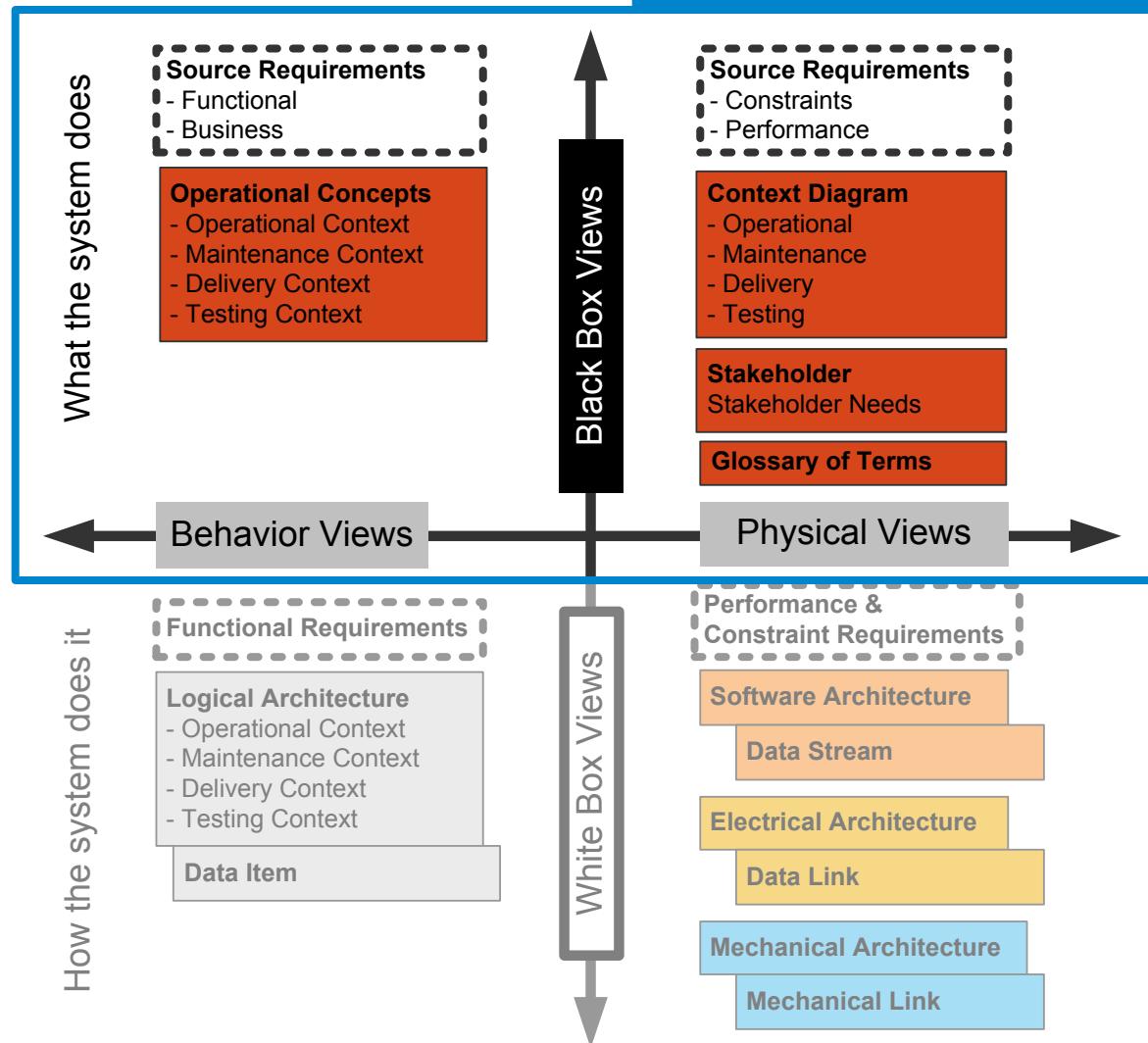
- Answers the question
 - What does the customer want and why?
- What is unique about this method?
 - This method
 - Identifies the customer (stakeholder)
 - Describe the problem facing the customer(s)
 - Keep the problem and the solution separate
 - (adding solution ideas early constrains the possible solutions)
 - Starts to clarify the handoffs between teams
 - (organizational hierarchy)

SysML Diagrams for the Problem Space



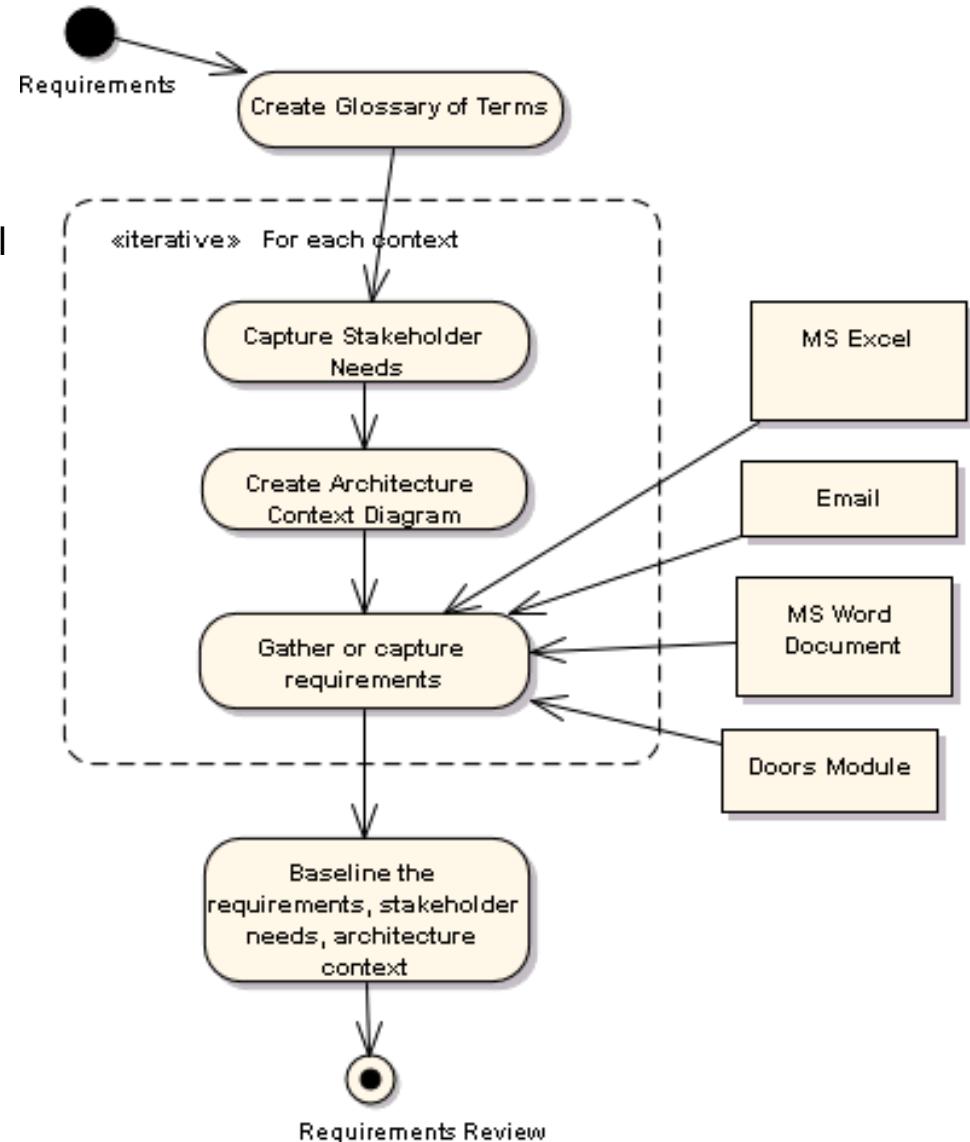
Abstraction Views

Problem Space



Originating Customer Requirements

- Answer the questions:
 - What is the problem?
 - What are the users doing?
 - What are the objects in the real world?
- Work from the user requirements inward
- Ends with requirements review
 - Vocabulary consistent between stakeholder needs, context diagram, & requirements



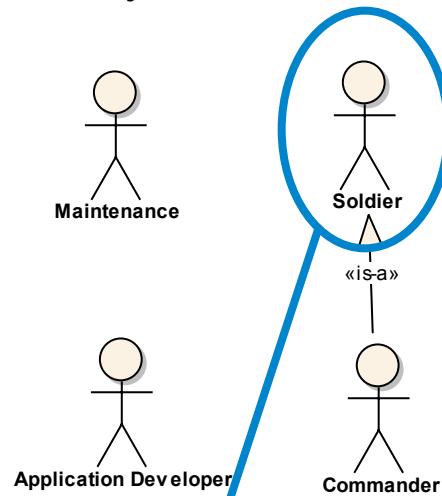


Glossary of terms

- Any noun or acronym used in the source requirements
- Stakeholder focused and not designer or implementer focused

Stakeholder needs

1) Identify stakeholders



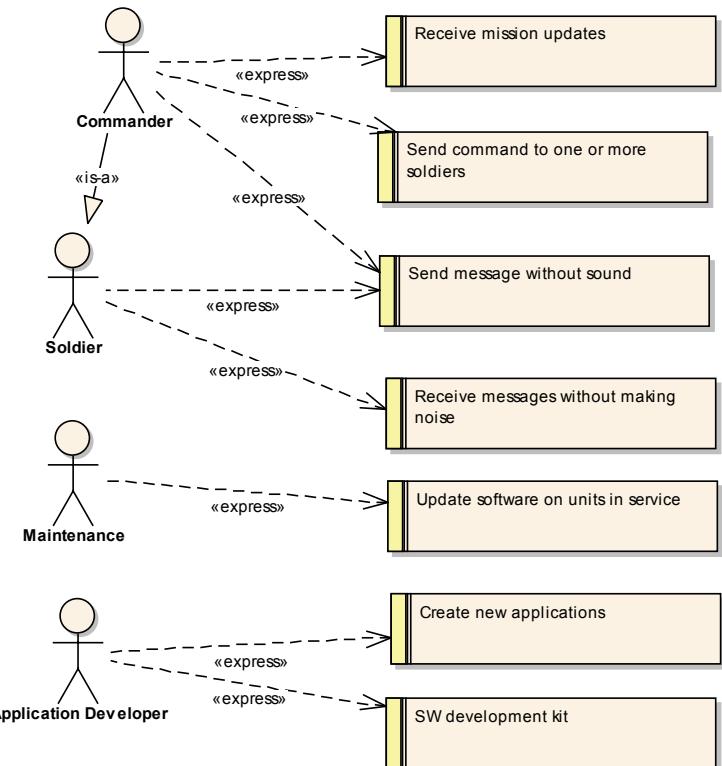
2) Describe stakeholders

Roles: Describe the role the stakeholder plays. There may be more than one role each stakeholder operates in.

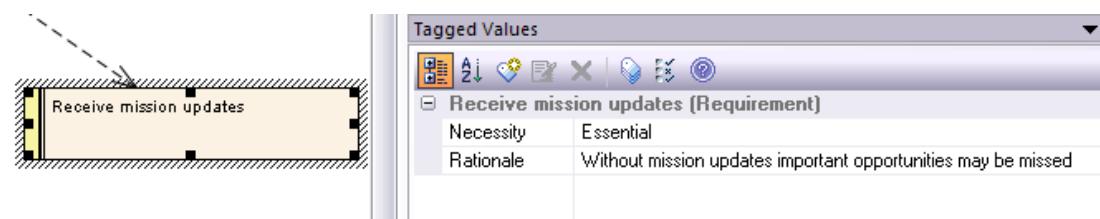
Authority: Describe the authority of the stakeholder in each of their roles.

Knowledge: Describe the level of knowledge the stakeholder has.

3) Identify Stakeholder Needs

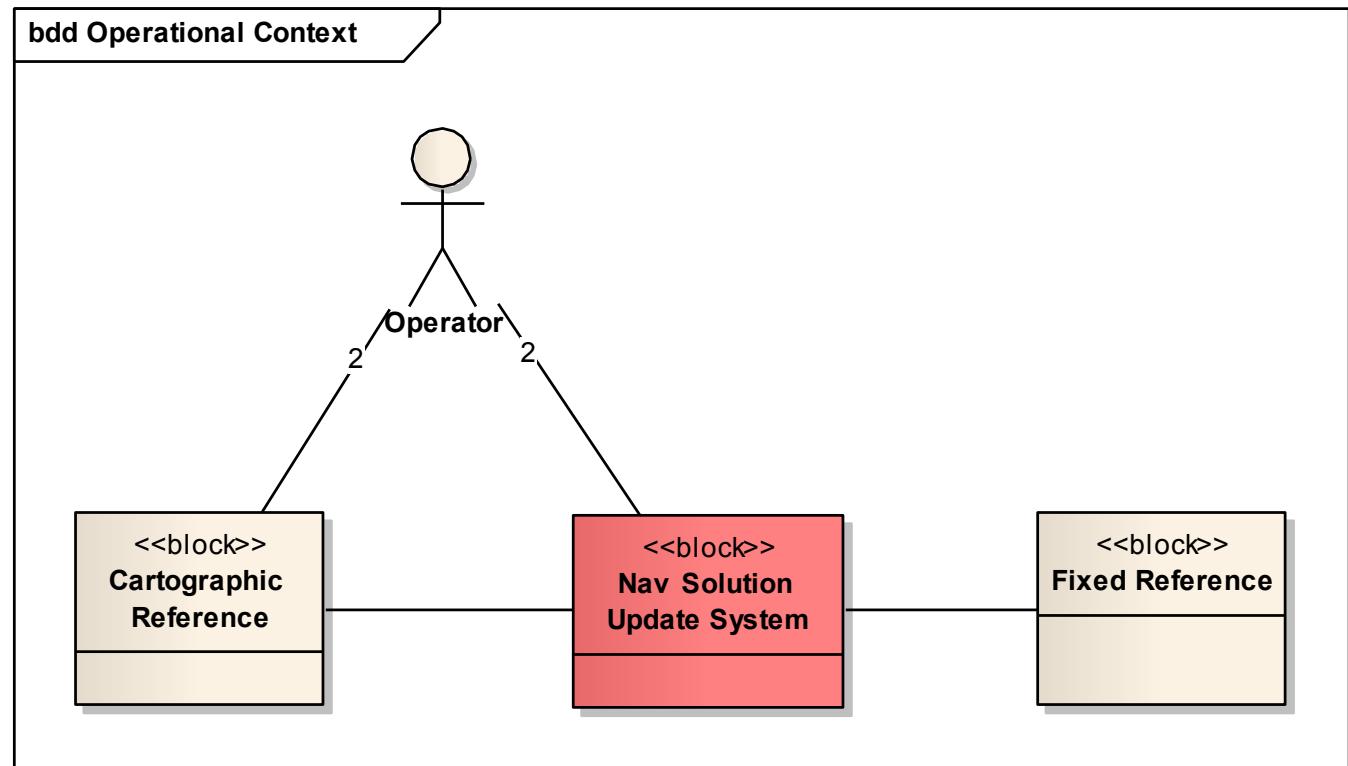


4) Add need necessity & rational



Capture Architectural Context

- Show actors and association with system of interest
- Purely “black box” perspectives

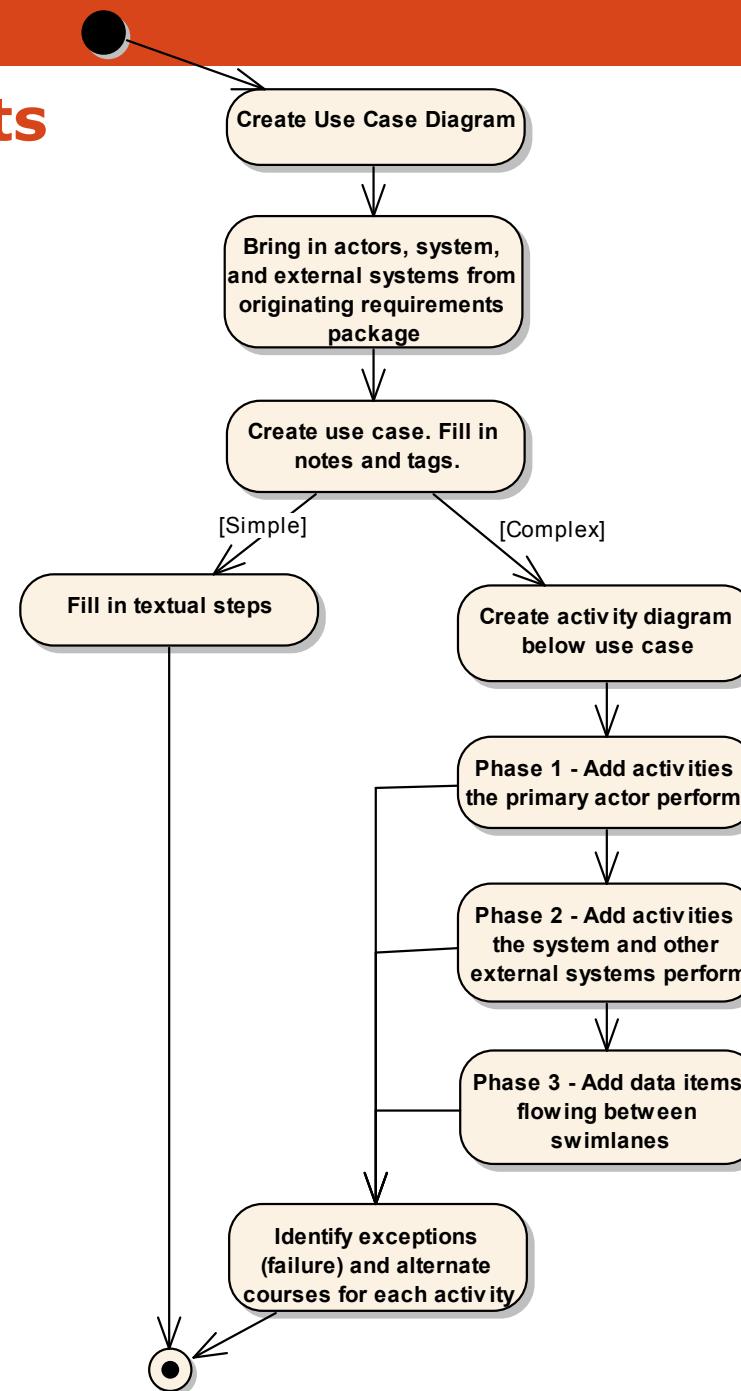


Source Requirements

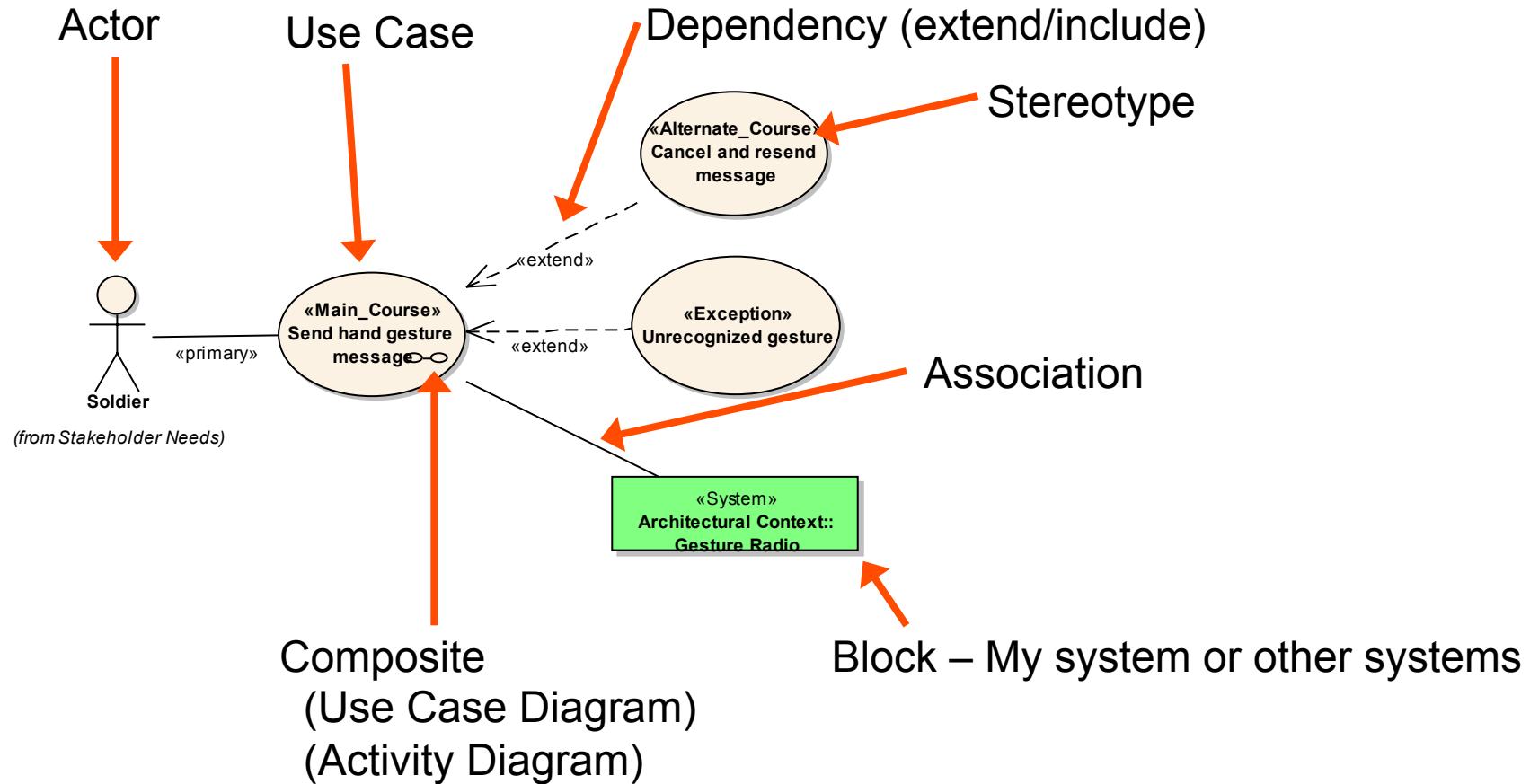
- Source requirements are requirements provided by external stakeholders, not by the designers
- This task involves gathering and organizing the requirements
- In an ideal world, the external stakeholders
 - Provide requirements at the black-box level (do not over-constrain the solution)
 - Elaborate the requirements with use cases

Operational Concepts

- Start with source requirements and stakeholder needs
- Re-use actors and external systems from the stakeholders and architecture context packages
 - If you discover a new actor or system, add it to the stakeholder/architecture context also

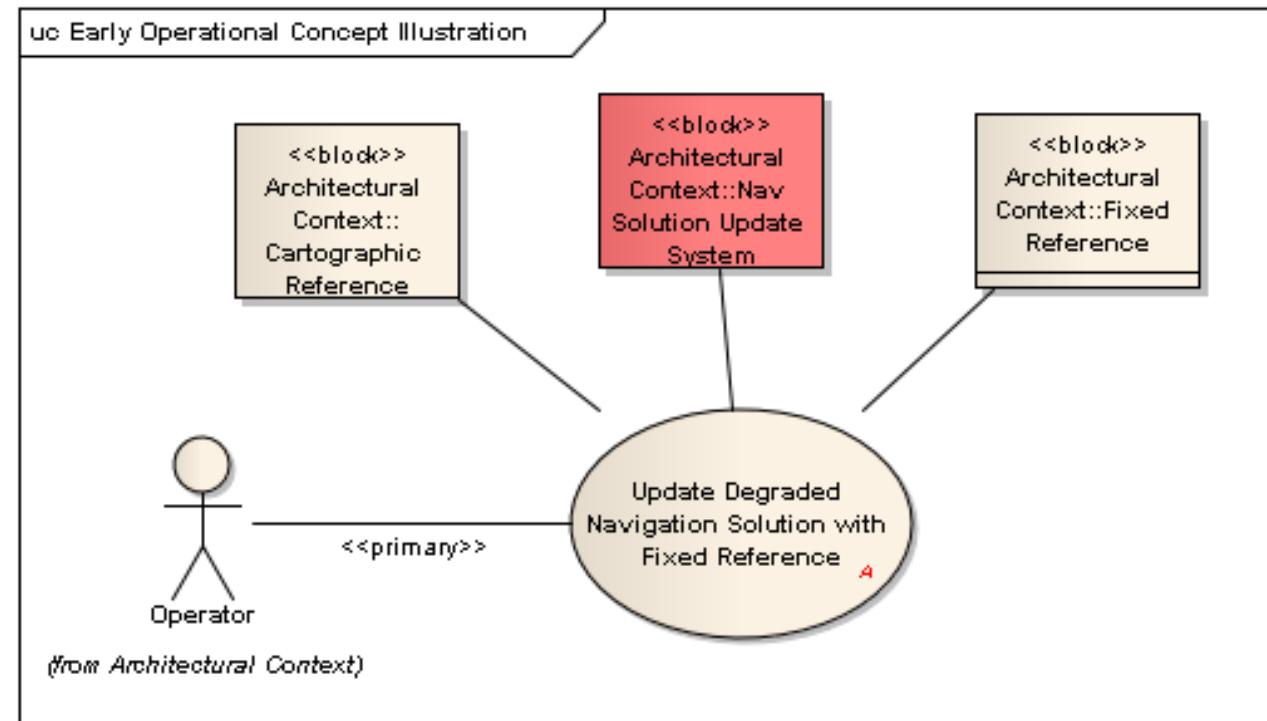


Use Case Diagram



Conduct Use Case Analysis

- Identify prospective Use Case: “verb-noun” convention
- Primary actor identification



Characterize Use Case

- Purpose, Goal, or Objective
- Trigger Stimulus
- Preconditions
- Post Conditions

UseCase : Update Degraded Navigation Solution with...

General Requirements Constraints Links Scenarios Files

Name: Update Degraded Navigation Solution with Fixed Reference

Stereotype: Abstract

Author: rwiorgen Status: Proposed

Scope: Public Complexity: Easy

Alias: Language: <none>

Keywords:

Phase: 1.0 Version: 1.0

Notes:

Purpose:
To update a navigation solution that is no longer receiving a reliable source of reference updates (i.e. GPS or [Nav](#) Radio fixes). A degraded navigation solution is prone to large drift over time, thus it requires a periodic update to reset the solution to a higher probability of certainty.

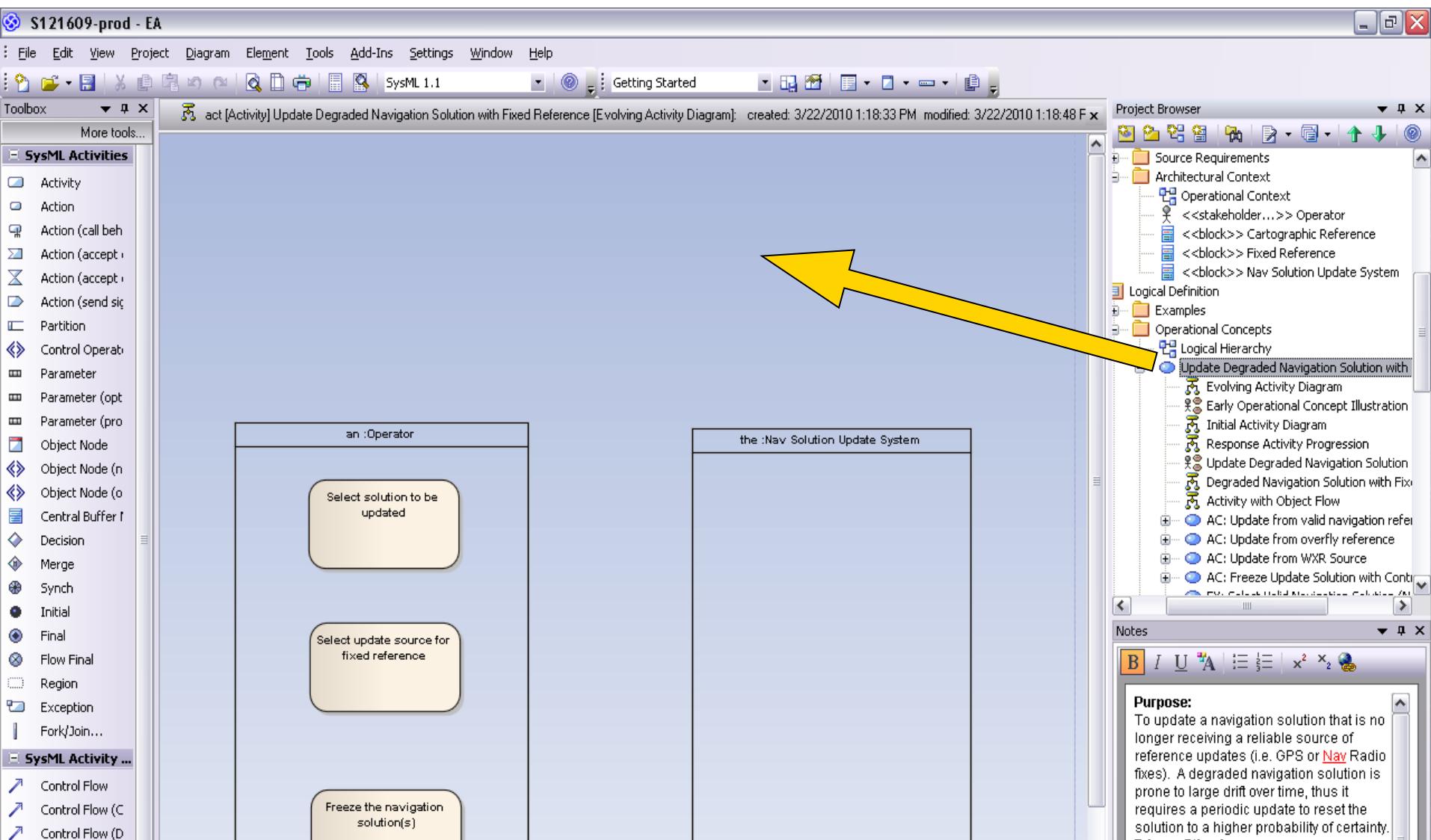
Trigger Stimulus:
Pilot determines that own aircraft present position estimates are significantly different than actual aircraft present position. Normally, the pilot will not take any corrective action unless: 1) the aircraft present position appears to be erroneous (greater than [8nm](#) from expectation), or 2) degraded navigation solution has not been updated within the last 60 minutes.

Preconditions:
The navigation solution has reached a degraded state with no automatic updates from fixed reference sources. The degraded navigation solution navigation errors are significantly different than actual aircraft present position (greater than 1 [nm](#) from true position).

Post Conditions:
The degraded navigation solution is now providing present position estimates with the update correction.

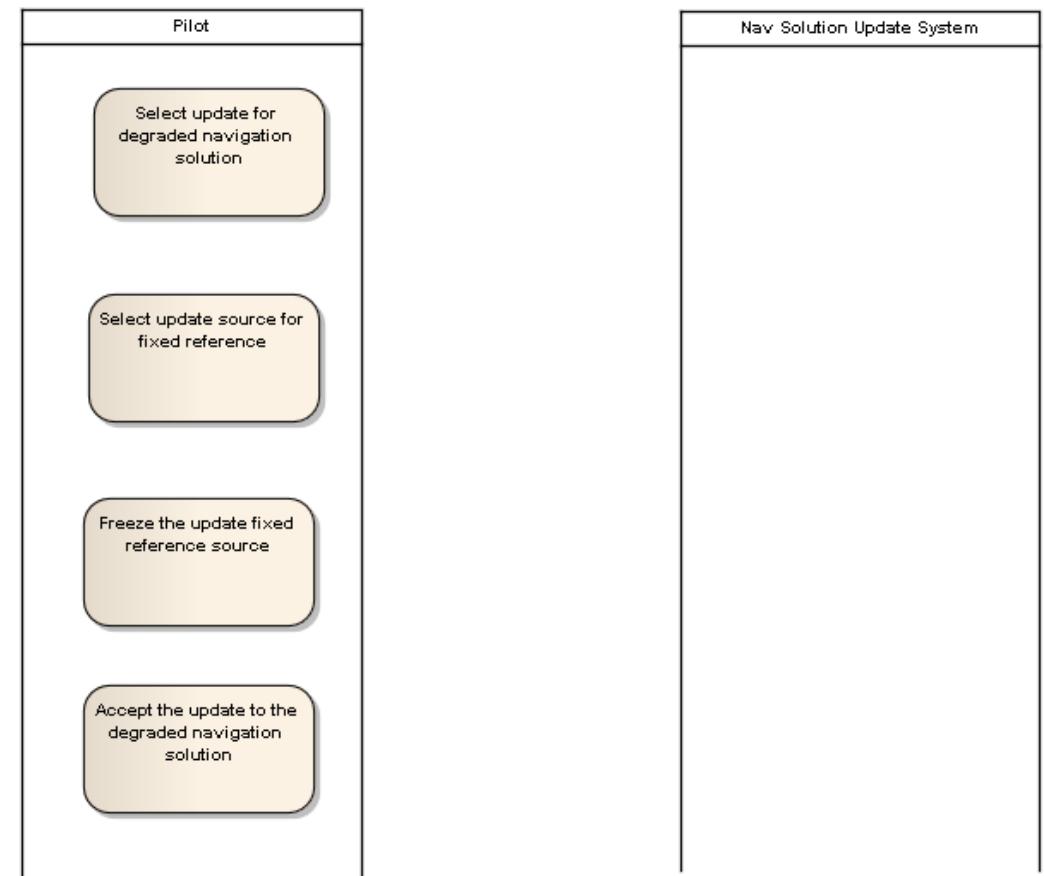
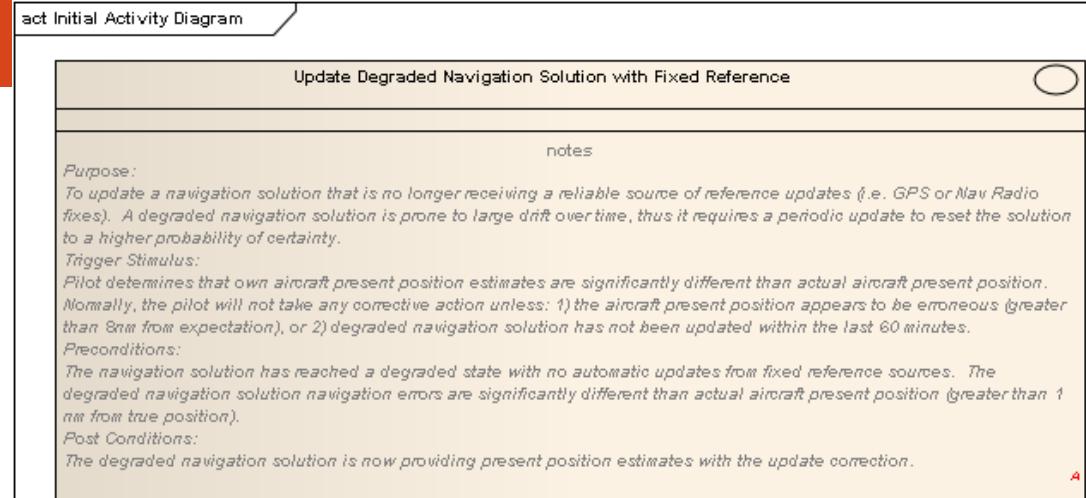
OK Cancel Apply Help

Build Activity Diagram (Scenario)



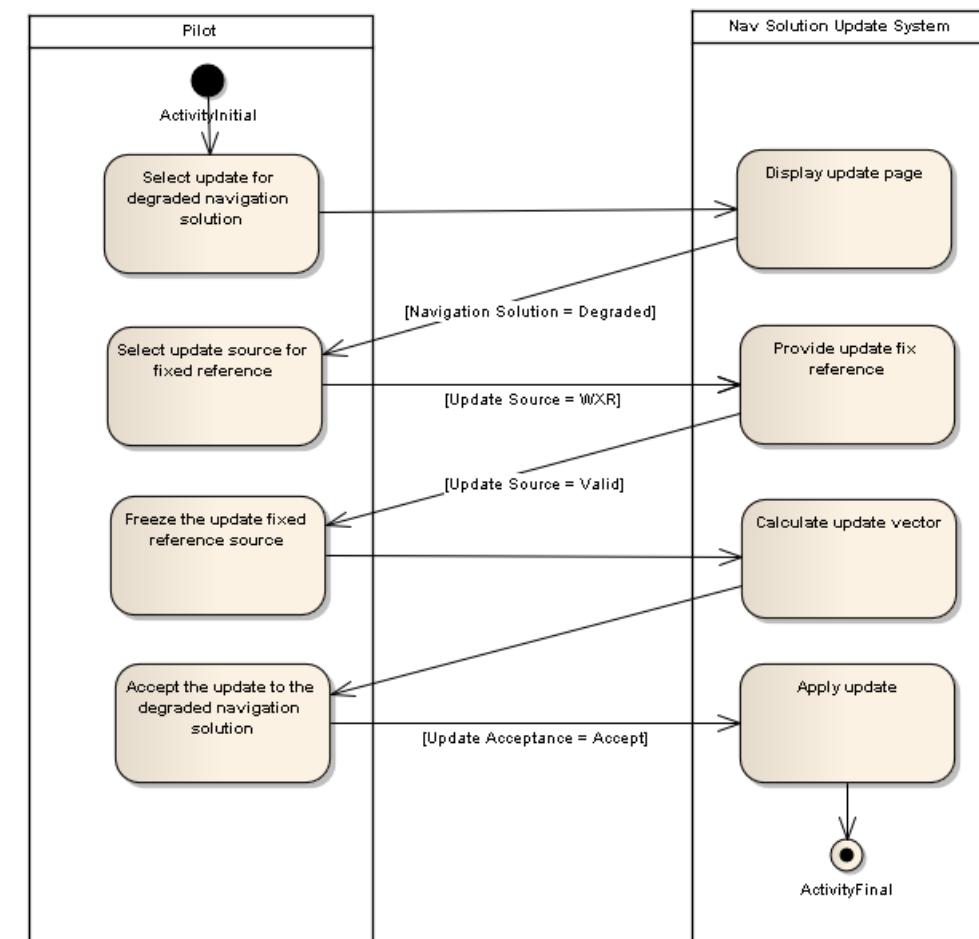
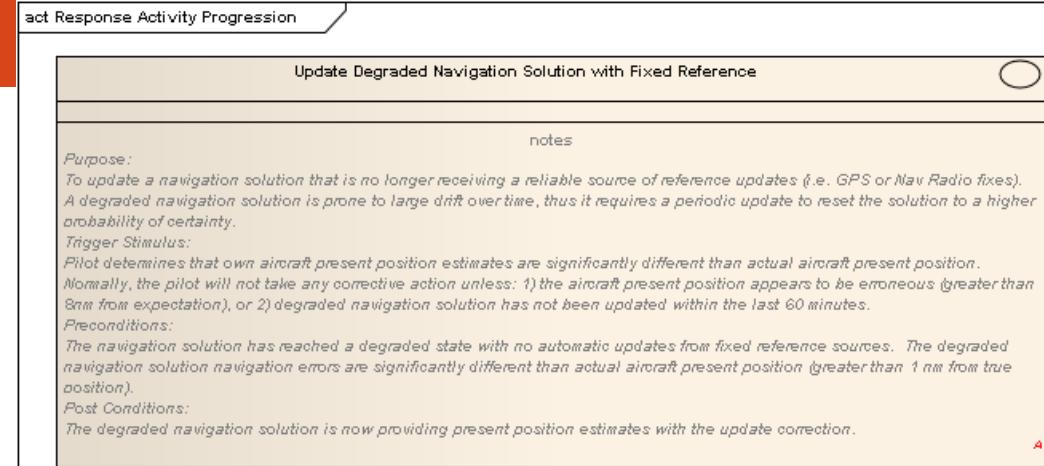
Define Scenario

- Add activities of primary actor



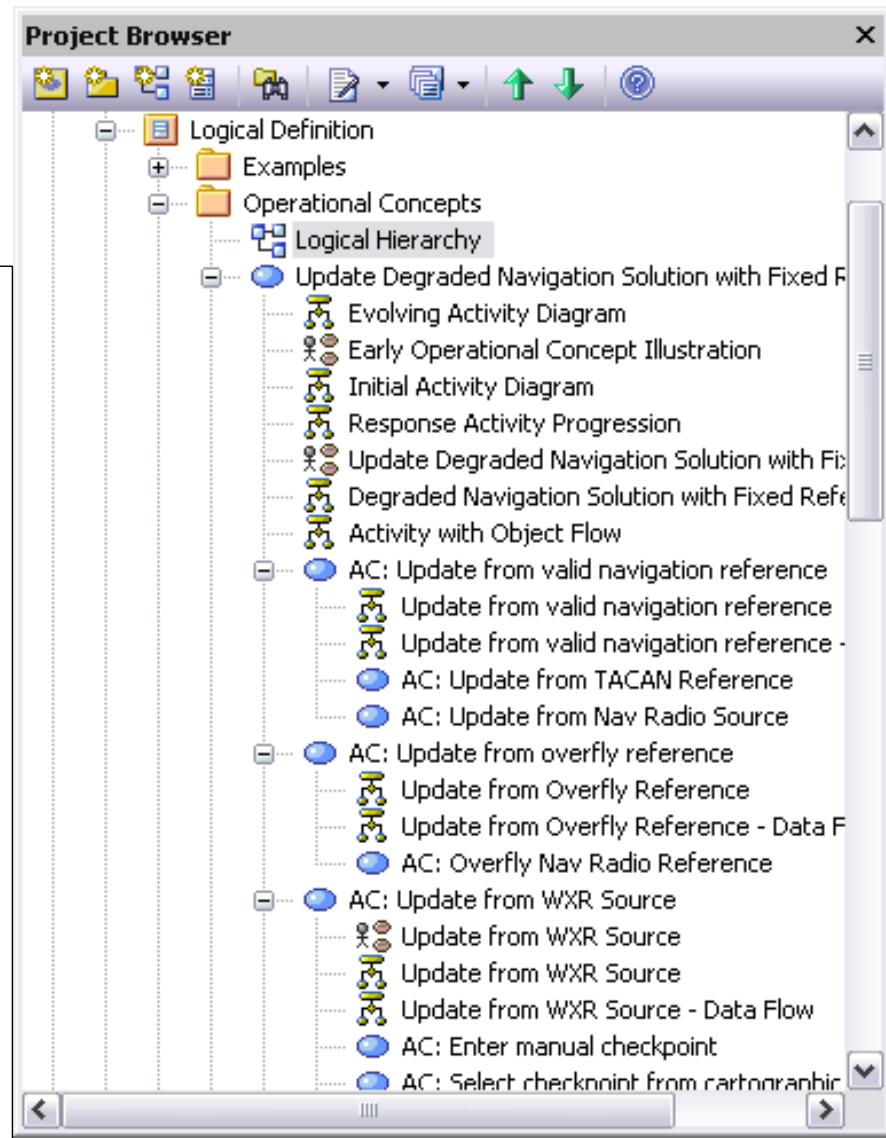
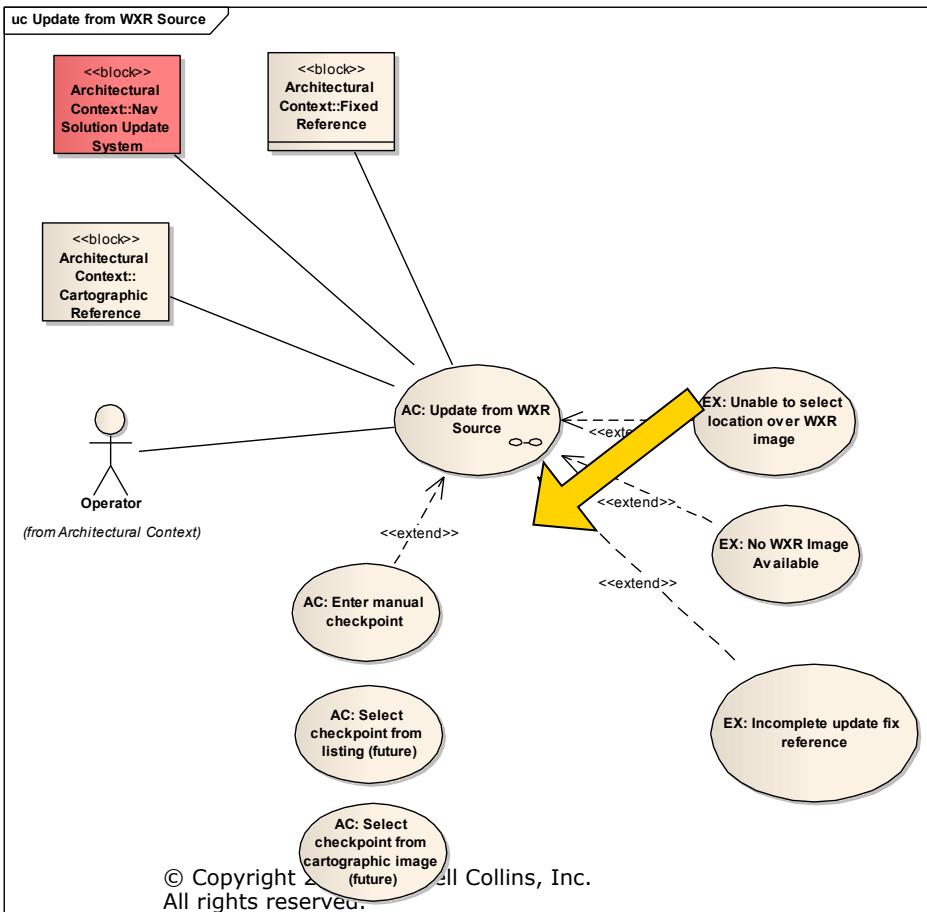
Define Scenario

- Start with functional requirements
- Add activities/actions to satisfy the requirement
- Each action is traced to one or more requirements
- Derive new requirements as needed
- Maintain purely “black box” perspective



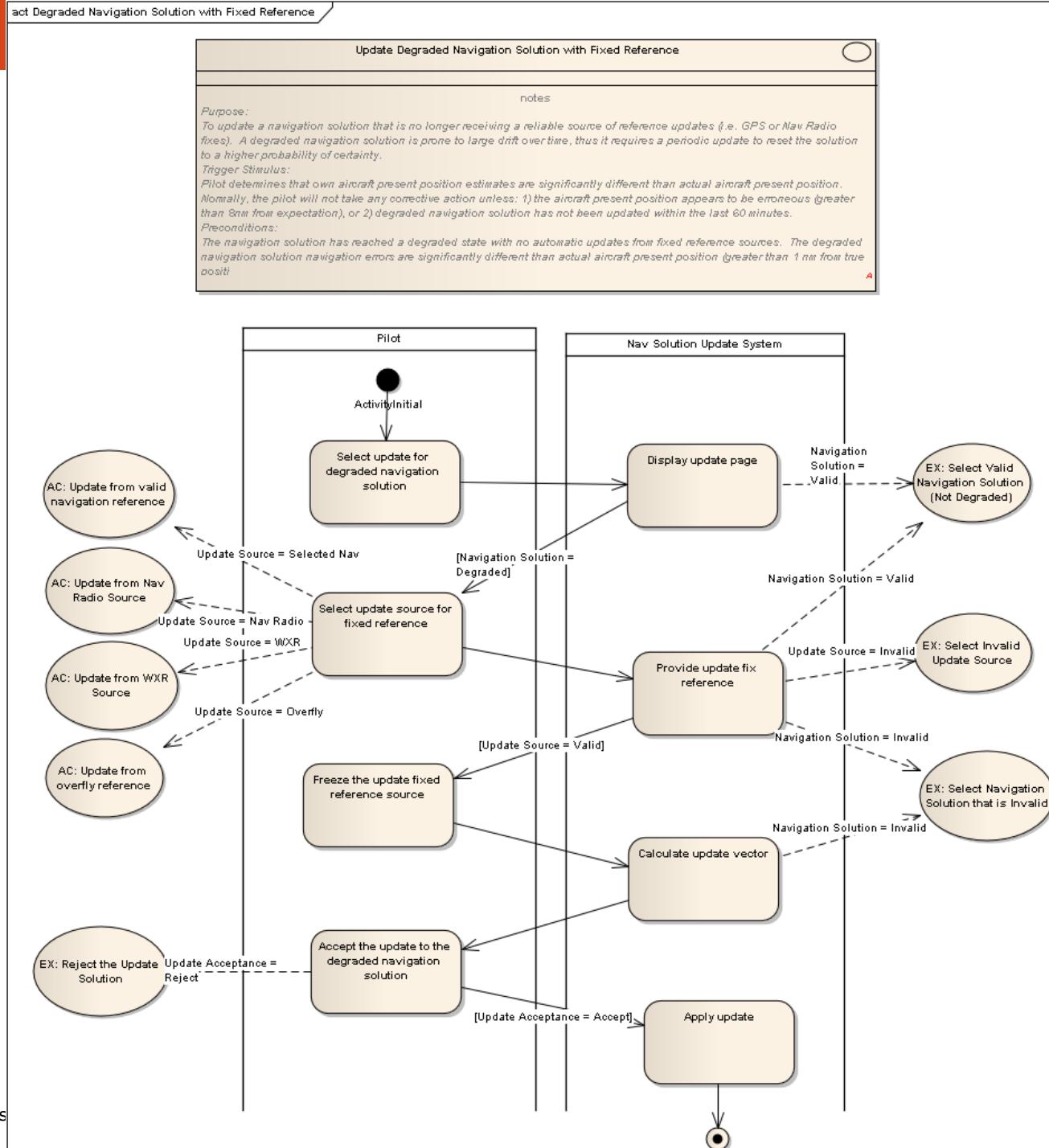
Organize Model Elements

- Activity Diagrams (Scenarios) under Use Case elements
- Make Use Case “composite” – double click opens scenario



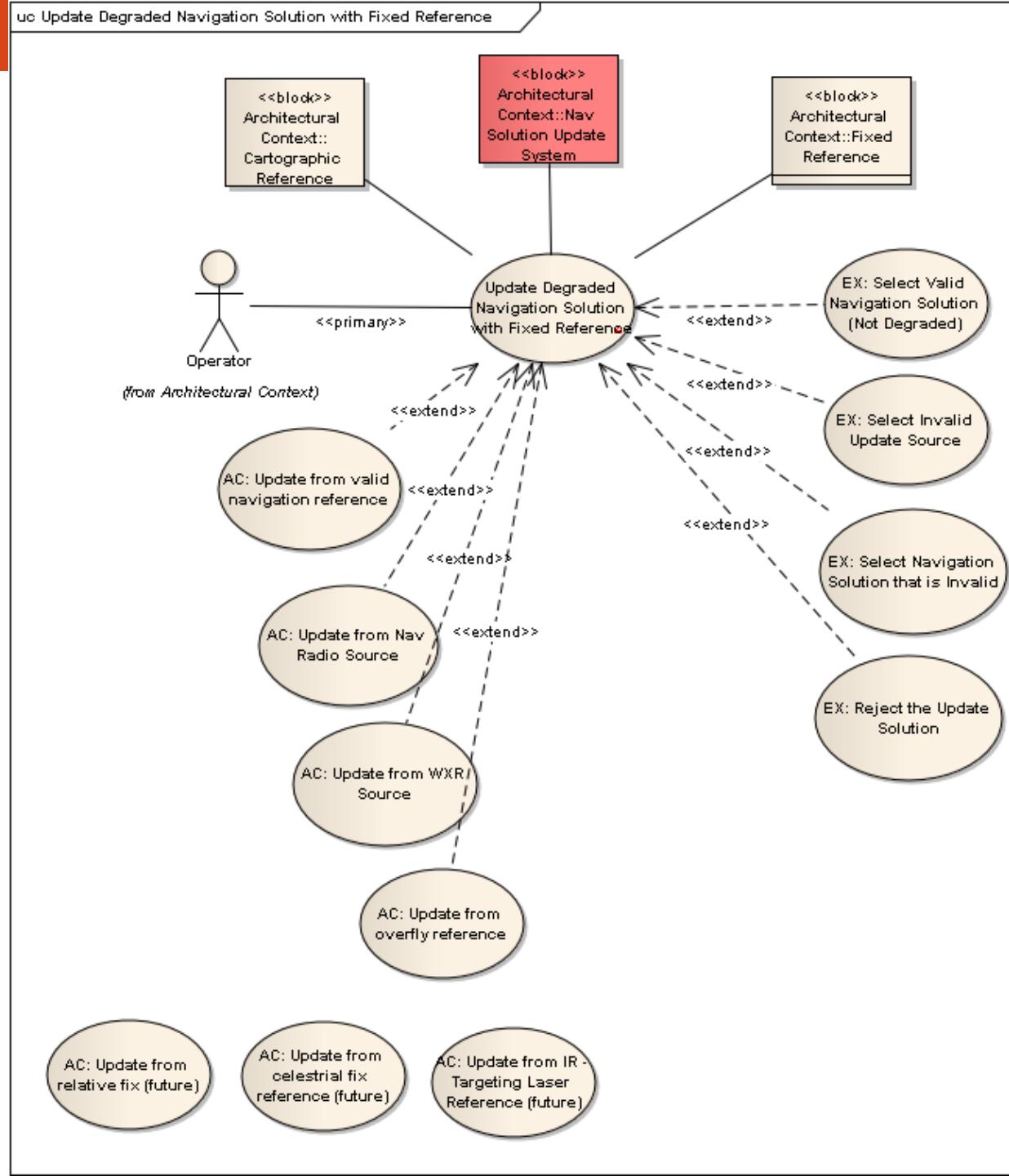
Add Exceptions and Alternate Courses

- Examine each “happy day” step (activity)
- What can go wrong?
- What else might the actor do?
- Does the step itself require further elaboration (extension)?
- Simple dependency relationship from activity to use case



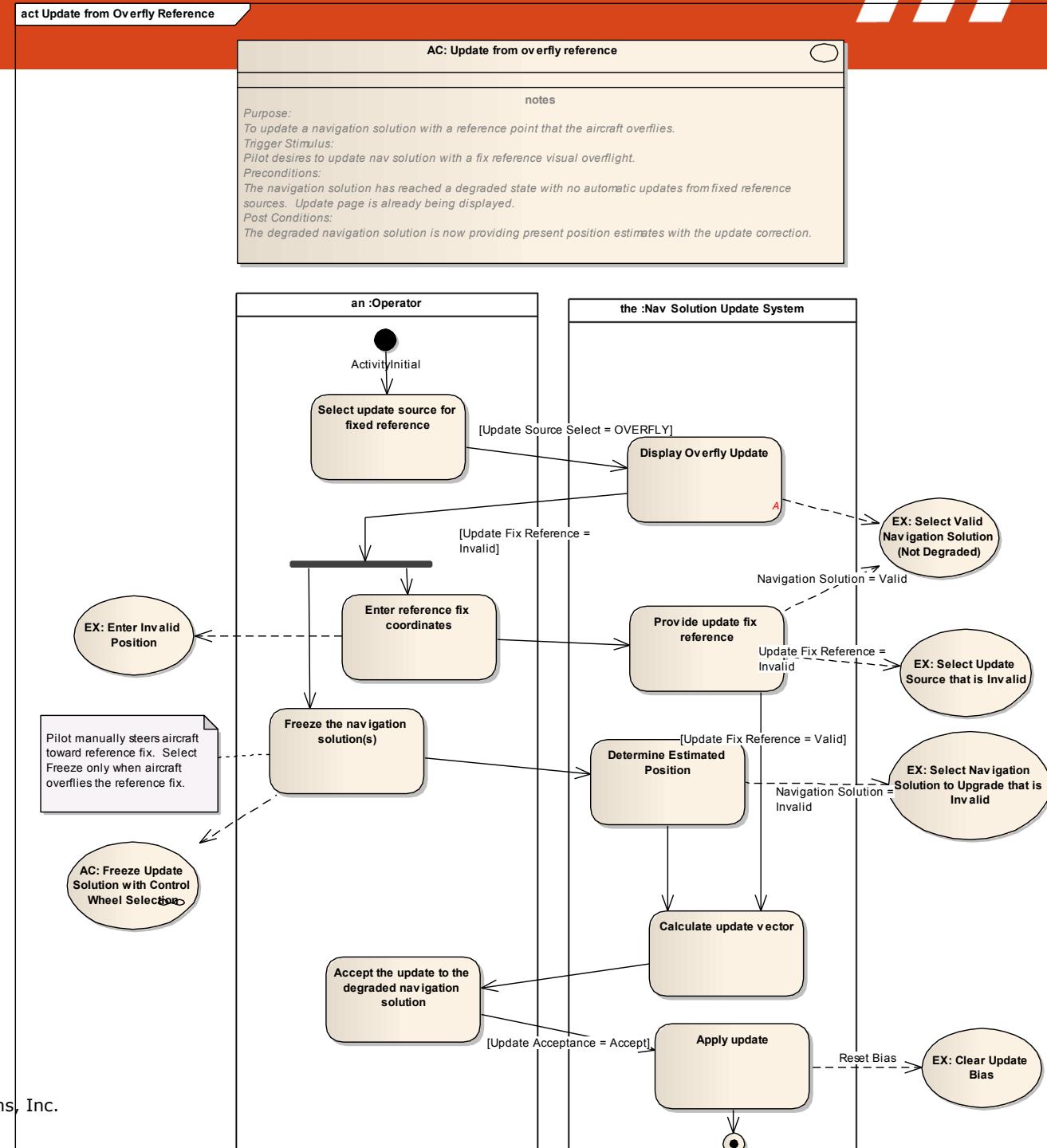
Use Case Extensions

- Add:
 - Alternate Courses
 - Exceptions Cases
 - Extensions
- Maintain singular focus
 - One use case is primary focus of Use Case Diagram



Elaborate Each Extension

- Activity diagram (scenario) for each AC or EX
- May be variant of original use case scenario



Scenario Diagrams

- Generally, use Activity Diagrams to express unique scenarios
- However, if alternate course or exception is “simple”, consider “Notes” narrative
- Use Case Analysis generates future opportunities – capture them!

UseCase : EX: Incomplete update fix reference

General Requirements Constraints Links Scenarios Files

Name: EX: Incomplete update fix reference
Stereotype: Abstract
Author: rwjorgen Status: Proposed
Scope: Public Complexity: Easy
Alias: Language: <none>
Keywords:
Phase: 1.0 Version: 1.0 Advanced
Notes:
B I U A | ⌂ ⌂ | x² x₂ ⚙
The selected fix reference as selected on the display is dependent on that display having a valid bearing/ range from the aircraft to the selected point on the display.
If the display does not have a valid bearing/ range to the selected fix, the selected fix will remain invalid.

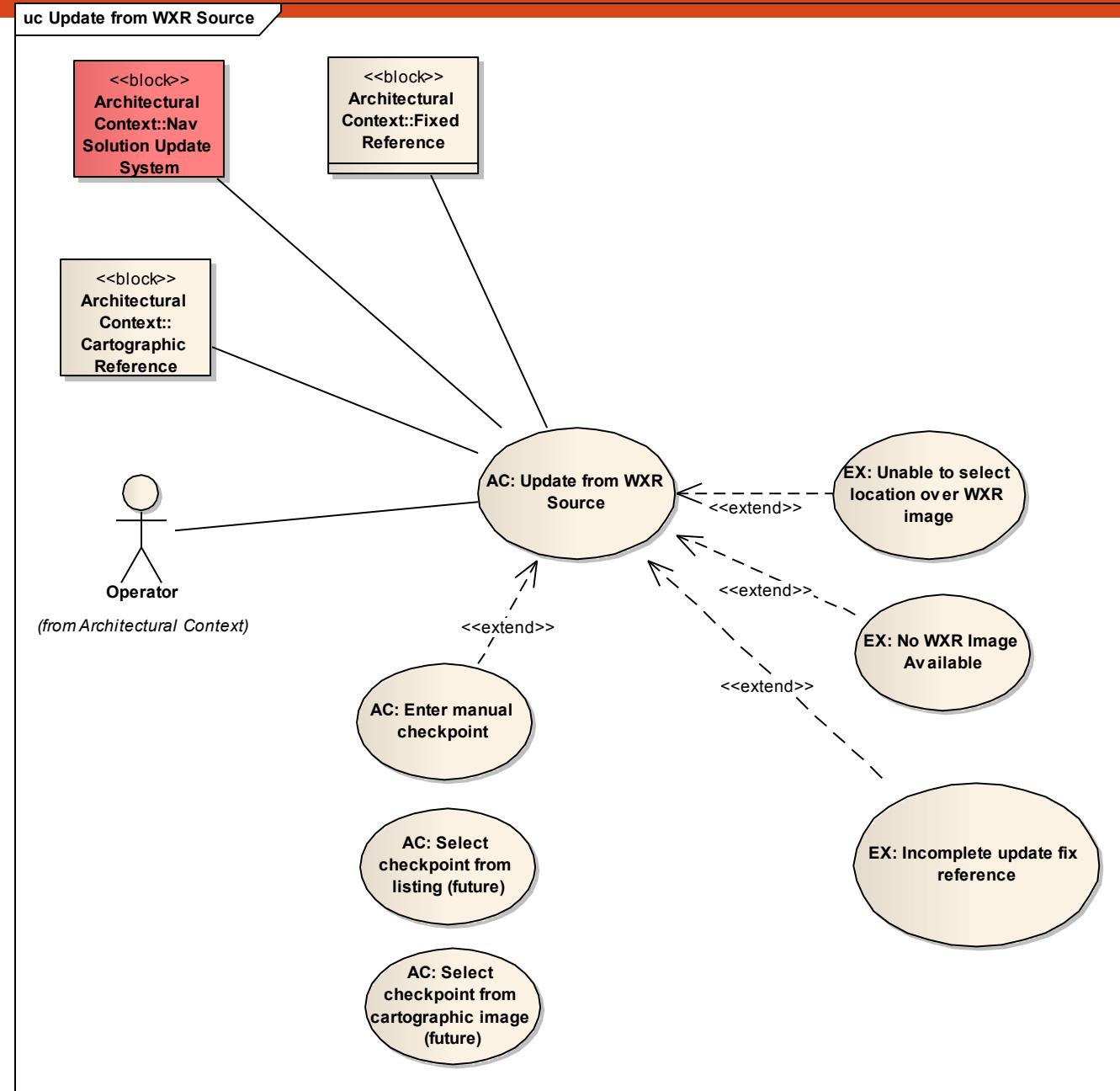
UseCase : AC: Select checkpoint from cartographic i...

General Requirements Constraints Links Scenarios Files

Name: AC: Select checkpoint from cartographic image (future)
Stereotype: Abstract
Author: rwjorgen Status: Proposed
Scope: Public Complexity: Easy
Alias: Language: <none>
Keywords:
Phase: 1.0 Version: 1.0 Advanced
Notes:
B I U A | ⌂ ⌂ | x² x₂ ⚙
Future opportunity: provide ability to select the checkpoint reference from a a digital map

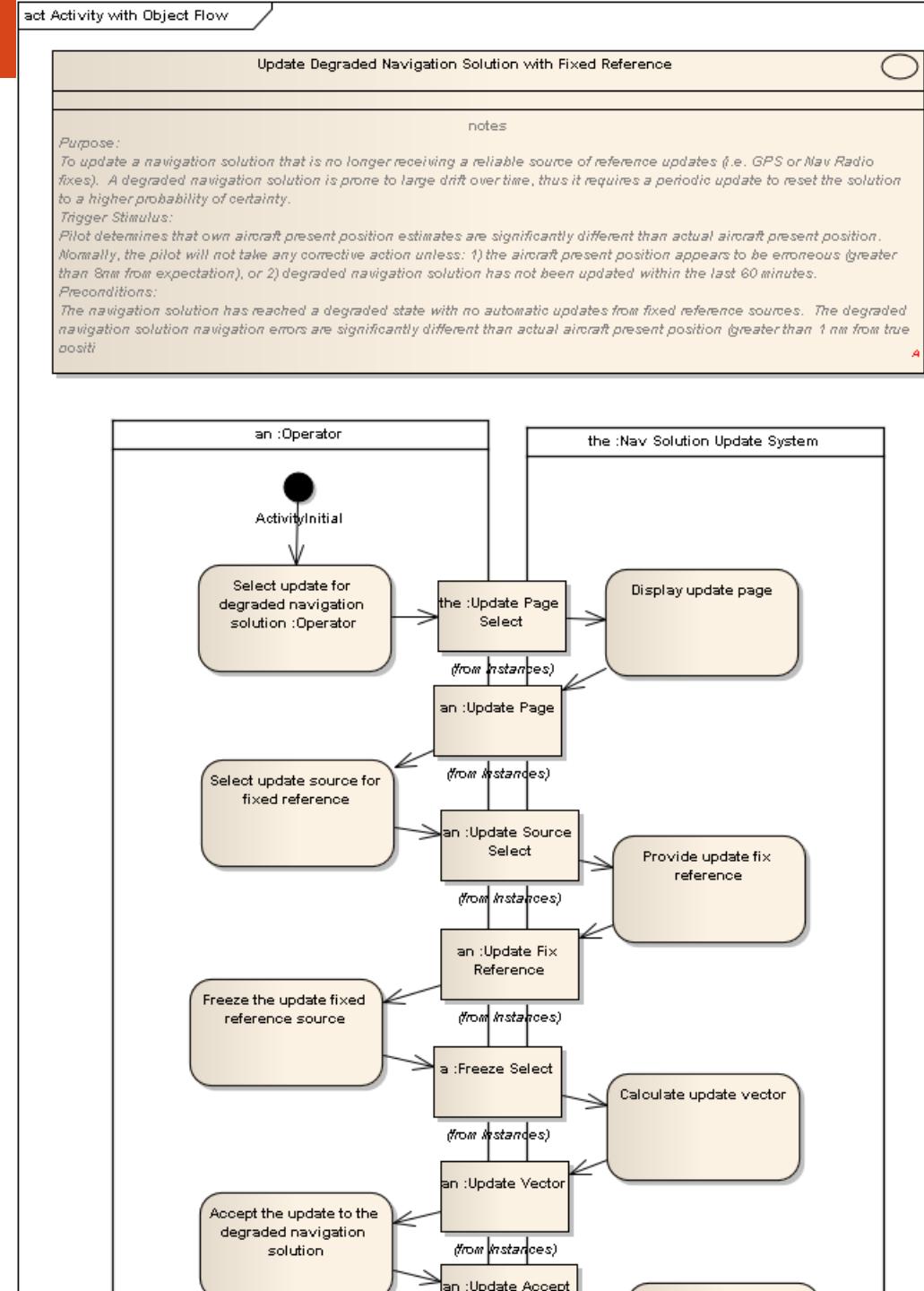
Repeat Use Case Process

- Use Case identification
- Scenario definition – primary actor
- Scenario definition – system of interest, secondary actors
- Consider alternate courses/ exceptions/ extensions



Add Object Flows

- Add object flow for each interaction
- Two activity diagrams:
 - Sequence flow of scenario
 - Object flow of scenario
- However:
 - Consider transaction “visibility” between actors (swimlanes)
 - Consider activity hierarchy (functional decomposition)
- Refine the object flow definition
 - Classify to a block
 - Stereotypes
 - User Interface
 - Data Item
 - Tagged values
 - Engineering Units
 - Max/Min Values



Conclusion

- Answers the question
 - What does the customer want and why?
- Look at from a black box perspective
 - Operational Concepts
 - Stakeholders & stakeholder needs
 - Source Requirements