

# The Object Game

## IV&V Game Series Overview

- Focus on issues encountered implementing an agile, model driven approach as applied in the IV&V environment to understand the systems, projects, and programs that IV&V supports.
- Modeling approach is not focused on construction, but analysis
- Can be applied to IV&V VBS sections:
  - 2.0 Verify and Validate Concept Documents (2.1, 2.2, 2.3, 2.5)
  - 3.0 Verify and Validate Requirements (3.3, 3.4)
  - 4.0 Verify and Validate Test Documents (4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8)
  - 5.0 Verify and Validate Design (All sections)
- Combines best practices from:
  - Color Modeling
  - Object Oriented Design
  - Feature Driven Development (FDD)
  - Scrum and other Agile methods used to produce artifacts not limited to code

## The Object Game - Overview

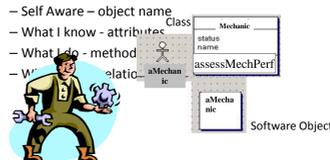
- Rationale:** In order to model modern systems, it is essential to understand the basic characteristics of any object oriented system.
- This includes:**
  - understanding what objects and classes represent,
  - how to show objects and classes using UML,
  - show how objects in a system can relate to each other.

## The Object Game - Goals

- Participants will be able to answer the following questions upon completion of the session;
  - What is an object?
  - What are the four characteristics of an object?
  - What is a class?
  - Identifying classes?
  - What are the three relationships between classes?
  - What are Archetypes?
  - What color is the class?
  - What is the domain neutral component (pattern)?

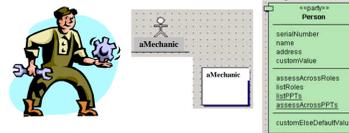
## Objects

- What is an object
- 4 characteristics of an object
  - Self Aware – object name
  - What I know - attributes
  - What I do - method
  - What I relate to - relationships



## Classes

- What is a class
- Difference between a class and an object



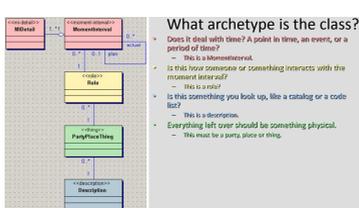
## Domains

- Technical equivalent of informal "subject area" concept
- Usually short for "problem domain" or "application domain"
  - There are also various solution domains in typical applications, such as data and communication services
  - We will only consider problem domains in this tutorial
- Characterized by high internal coherence, low external dependencies. Software oriented

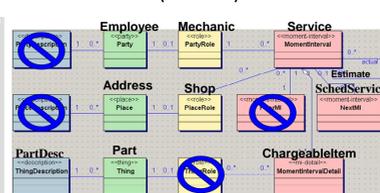
## Color and Archetypes

- What is an archetype
- Review the 4 major archetypes
  - Moment-Interval (Event or Activity)
  - Role
  - Party, Place, Thing
  - Catalog Description
- Apply Color to the archetypes

## Develop an Overall Model - Business Pattern



## Domain Neutral Component (Pattern)



## Summary - Object Game

- Identify objects and define classes,
- Create relationships between classes
- Categorize classes by Archetype using color
- Discover the Domain Neutral Component (pattern)

# The Model Game

## The Model Game - Overview

- Rationale:** Often development groups will start building solutions without understanding the system well, without traceability to requirements, and without clear direction about what is to be delivered.
- Building a domain model helps to understand and communicate how a system is supposed to behave. This is critical to successful, agile development.
- The domain model can be linked directly to requirements to provide traceability.

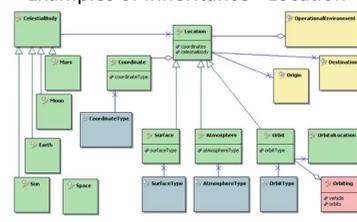
## Exercise

- Teams of 3 (from The Object Game)
- 5 minutes
- Review the model created in the previous session
  - Are all the events identified and colored?
  - Is there a role or roles associated with each event?
  - Is there an party, place, or thing attached to the role?
  - Are there any event details?

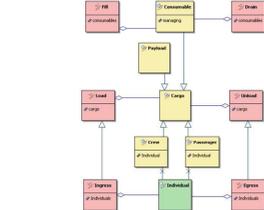
## Inheritance vs. Composition

- Inheritance
  - "Is a" relationship
  - Pro's
  - Con's
- Composition
  - Parts is Parts
  - Pro's
  - Con's

## Examples of Inheritance - Location



## Examples of Composition - Cargo / Consumables



## Exercise

- Teams – 6 minutes
- Create an inheritance structure in your model
- Create an example of composition, in the model

## The Model Game - Goals

- Identify classes in a problem domain.
- Show relationships between classes.
- Identify key attributes and place them in the correct class.
- Identify key operations and place them in the correct class.
- Understand when and why to use inheritance vs. composition.
- Use color and archetypes to clarify the model.
- Use the domain neutral component (pattern).

## Summary - Model Game

- Create an overall model
- Add relationships to the model
- Apply the Domain Neutral Component to the model
- Elaborate the model with key attributes and operations
- Create examples of inheritance and composition

# The Feature Game

## The Feature Game - Overview

- Rationale:** What's in a name? Something as simple as a consistent naming convention can convey a lot of valuable information at a glance.
- Naming features in a prescribed way can bring great value to a group practicing agile development.
- The Feature Syntax naming convention also aids in properly sizing deliverables to ensure that the organization can deliver results in an iterative and incremental manner.
- The Feature Syntax also tends to place focus on the deliverable, rather than the work or task. This aids the agile practitioners in delivering tangible results, frequently.

## The Feature Game - Goals

- Identify features for a domain.
- Build a feature list for a system.
- Know the syntax for naming features.
- Know the syntax for naming feature sets (activities or use cases)
- Know the syntax for naming capabilities (functional areas or topics)
- Group features into related sets.
- Group features into domains.
- Understand the size constraints of features.

## Exercise 1 – Identify system activities

- Five Minutes
- Use Subject Areas from first session
- Working in teams
- Pick three classes from the model
- List two to four actions from each class

## Anatomy of a Feature

- A very small block of client-valued, client visible functionality that can be delivered within the project's iteration size.
- Follows a specific format:
  - <Action> on <Result> on <Object>
  - Apply the power to the Crew Launch Vehicle
  - Transmit the load to the Flight Computer
  - Ignite the Solid Rocket Booster for the Crew Launch Vehicle
  - Assemble the Solid Rocket Booster for the Crew Launch Vehicle

## Exercise 3

- 3 minutes
- Working in Teams
- Using the Feature List previously constructed
- Break up the Feature into related Groups

## Feature Set Syntax

- A grouping of related features, often a business activity.
- <action>-ing a(n) <object>
  - Assembling a Mission Launch Vehicle
  - Loading the Propellants
  - Safing the Crew Launch Vehicle

## Exercise 4

- 3 Minutes
- Working in Teams
- Using the feature groupings from Exercise 3
- Name the Feature Sets using the Feature Set Syntax

## Major Feature Set Syntax

- A major feature set is often associated with the management of a business unit or major subject area within the domain.
- <object> management
  - Vehicle Assembly management
  - Launch Prep management
  - First Stage Separation management

## Exercise 5 (Optional)

- 5 minutes
- Working in Teams
- Group the Feature Sets into related domain areas
- Name the domain areas using the Major Feature Set Syntax

## Feature Milestones

- Has traceable weighted milestones
  - Domain Walkthrough 15%
  - Design 30%
  - Design Inspection 5%
  - Develop 40%
  - Final Inspection 5%
  - Promote to Build 5%
- NASA SRMV Backlog Item Milestones
  - Submitted 5%
  - Accepted 3%
  - Active 52%
  - In Peer Review 30%
  - Done 5%
  - Promoted 5%

## The Feature Game - Summary

- Explain the importance of fine grained features
- Identify and name candidate features
- Use the feature syntax to name deliverable functionality
- Group related features into sets (activities) and domains (capabilities)

Mac Felsing,  
John.M.Felsing@ivv.nasa.gov, Tasc, Inc.

NASA POC: Lisa Downs Sadie.E.Downs@nasa.gov

NASA Independent  
Verification and Validation  
Facility  
Fairmont, West Virginia

