ARTiSAN Software Tools

Systems Engineering and UML (SysML)
Manohar Rao
Greg Narog

www.artisansw.com
UML: Industry Standard Notation

- Notation only
- Defining the System
  - Use Cases
  - Scenarios
- Designing the System (Object Model)
  - Class
  - Collaboration
  - Dynamic
  - Package
The Benefits of UML

• Communication, Communication, Communication
  – Among team members
  – Among teams
  – Between contractors and sub-contractors
  – Between developers and project sponsors
• Standard language for the capture and reuse of IP
• Better estimate project effort
• Better control project budget
• Better prepared to deal with changing requirements
• Requirements traceability
The Basics of UML

Use Case Model

Scenario Model

Dynamic Model
The Good News - Real-time UML

- OMG formed a working party in 1999 - the Real-time Analysis and Design Group (RTAD)
  - Issued Request for Proposal (RfP) for UML extensions to support “Timeliness and Schedulability” - Adoption in mid-2001;
  - Two other RfP’s identified (not yet issued):
    • Large-scale systems;
    • Quality of Service e.g. reliability, robustness (other than timeliness);
- ARTiSAN is the current chair (Alan Moore) of the submission team to address the RfP.
Evolution of The UML and SysML Standards

1960’s 1970’s 1980’s 1990’s 2000’s

Structured Methods: SSA&D Entity Modeling Event Modeling

US D.O.D Bell Labs XEROX PARC etc. etc.....

OO Programming: ADA Eiffel Smalltalk Simula C++

Shlaer/Mellor Booch Wirfs-Brock Coad/Yourdon Rumbaugh: OMT Martin/Odell

UML 1.x

OMG INCOSE

UML 2.0 SysML

Jacobson: OOSE

Bell Labs XEROX PARC US D.O.D etc. etc.....
UML 2.0 - Main Diagrammatic Changes

• New Diagrams
  – Composite Structure
    • Component hierarchies
    • The Part/Port/Connector paradigm
  – Interaction Overview
    • high-level flow of control
    • variant of Activity Diagram
  – Timing
    • change in state over time (linear)

• Significant changes to other diagrams
  – Sequence Diagrams
    • More structure to interactions
    • Sequence diagram hierarchies
  – Activity Diagrams
    • Divorced from state diagrams
    • Additional notations
UML for Systems Engineers?

• UML is already de facto standard within software engineering community
  – simplify handover
  – minimize errors

• UML is mature and extensible, and can be adapted to support SE requirements
  – SysML profile

• UML tools and training are widely available

• OMG standardization process supports UML customization for specific domains (e.g., systems engineering)
Modeling Requirements for Systems Engineers

- Structure
  - e.g., system hierarchy, interconnection, decomposition

- Behavior
  - e.g., function-based behavior, state-based behavior, decomposition of behavior

- Properties
  - e.g., parametric models, time property

- Requirements
  - e.g., requirements hierarchy, traceability, relationships

- Verification/Validation
  - e.g., test cases, verification/validation results

- Other
  - e.g., trade-off studies, spatial relationships
Key SysML Extensions

• Additional Diagrams
  – Assembly Diagram
  – Requirements Diagram
  – Parametric Diagram

• Extensions to some UML diagrams
  – Activity Diagram
  – Structure Diagram
  – Class Diagram

• Other
  – Item Flows
  – Quantities Model
SysML Assembly Diagram

- **sensors**: Sensor
- **control**: VehicleController
- **b&c**: Body & Chassis
- **sensors**: Sensor
- **control**: VehicleController
- **b&c**: Body & Chassis
- **sensors**: Sensor
- **control**: VehicleController
- **b&c**: Body & Chassis
- **sensors**: Sensor
- **control**: VehicleController
- **b&c**: Body & Chassis

**Vehicle**

- **assembly**: Power Train
  - **clutch**: Driveshaft
  - **halfshaft**: Transaxle

- **assembly**: Engine
  - **clutch**: Driveshaft

- **assembly**: Transmission
  - **clutch**: Driveshaft

- **assembly**: Transaxle
  - **clutch**: Driveshaft

- **assembly**: Wheel
  - **clutch**: Driveshaft

- **Exhaust System**:
  - **socket**: Hub
  - **socket**: Rack

- **Steering**
  - **socket**: Hub
  - **socket**: Rack

- **Brakes**
  - **socket**: Hub
  - **socket**: Rack

**Wiring**

- **Socket**
- **Struts**
SysML Requirements Diagram

Vehicle System Specification

- **Requirement** (id# = 102)
  - **txt**: System shall ... (criticality = H)
  - **R102**

- **Requirement** (id# = 111)
  - **txt**: System shall ... (criticality = L)
  - **R111**

Power Subsystem Specification

- **Requirement** (id# = 337)
  - **txt**: The subsystem ... (criticality = H)
  - **R337**

- **Requirement** (id# = 340)
  - **txt**: The subsystem ... (criticality = M)
  - **R340**

- **Requirement** (id# = 337.1)
- **Requirement** (id# = 337.2)

Vehicle System Design

- **Document** Market Needs
- **Rationale** Ref: Statement of Work
- **Diagram Description**
  - version = 0.1
  - description = "Depicts flowdown of acceleration reqt"
  - reference = "Requirements Management Database"
  - completeness = "partial"

- **Use Case Model**

- **System** Vehicle
  - **Assembly** Power Train
  - **Assembly** Brakes

- **TestCase** Engine Horsepower Test

Rationale:
- Ref: Statement of Work

Use Case Model:
- Vehicle System Design
  - Power Subsystem Design (Alternative = V6)

Diagram Description:
- version = 0.1
- description = "Depicts flowdown of acceleration reqt"
- reference = "Requirements Management Database"
- completeness = "partial"
SysML Parametrics Diagram

- `Road.incline` → `Vehicle.weight`
  - «paramConstraint» \( F = \text{weight} \times \sin(\theta) \) : GravitationalForce

- `Vehicle.powertrainForce` → `Vehicle.dragForce`
  - «paramConstraint»: TotalForce

- `Engine.displacement` → `Vehicle.dragForce`
  - «paramConstraint»: Power Train Equations

- `Vehicle.weight` → `Vehicle.dragCoef`
  - «paramConstraint» \( \{ F = m \times a \} \) : Newton's Law

- `Vehicle.dragForce` → `Vehicle.speed`
  - «paramConstraint» \( \{ \text{Force} = 0.5 \times \text{dragCoef} \times \rho \times \text{speed}^2 \} \) : DragForce

- `Vehicle.acceleration` → `Vehicle.speed`
  - d\(v\)/dt
  - «paramConstraint»: Integration

- `Vehicle.powertrainForce` → `Vehicle.acceleration`

- `Vehicle.performance`
The ARTiSAN Approach: System Definition

- **Constraints**: Performance, Reliability
  - Performance: Max. Flow Rate, Flow Sample resolution, Display Volume Resolution
  - Reliability: MTBF, MTTR

- **Modes**: System Initialisation, System Operational, Updating Fuel Prices
  - System Initialisation: initialisation completed, power down
  - System Operational: start price update sequence, price verification completed
  - Updating Fuel Prices: power down

- **Usage**
  - Customer uses EPOS
  - Filling Station Manager uses Dispenser Display
  - Kiosk Operator uses EPOS

- **Scope**
  - Filling Station System
    - Dispenser
      - Fast Flow Valve
      - Motor
      - Flow meter
      - Holster Switch
      - Dispenser Display
    - System Software
      - Keyboard Unit
      - Kiosk Display
  - Transaction details
    - Customer: replace nozzle, remove nozzle
    - Filling Station Manager: display check sequence
  - Transaction details
    - Configure Dispensers
    - Start Dispensing
    - Suspend Dispensing
    - Stop Dispensing
    - Set Dispenser Fuel Prices
    - Power-on

- **Performance and Reliability**
  - Reliability
    - MTBF, MTTR
  - Performance
    - Max. Flow Rate, Flow Sample resolution, Display Volume Resolution

- **System Initialization**
  - Initialisation completed, power down

- **System Operational**
  - Start price update sequence, price verification completed

- **Updating Fuel Prices**
  - Power down
The ARTiSAN Approach: System Design

Application

Infrastructure

Physical

Multi-tasking

persistence

architecture
Putting it All Together: UML for the Whole System

Modes Diagram

Requirements

Non-Functional Constraints

Use Case Model

Context Diagram

Dynamic Model

Scenario Model

Class Model

Concurrency Model

Hardware Architecture

Slide 17
Questions, Comments, Discussion