Systems Engineering
Foundations of Software Systems Integration

Peter Denno,
Allison Barnard Feeney
Manufacturing Engineering Laboratory
National Institute of Standards and Technology
Gaithersburg, MD
USA
Automating Integration : The Problem

• Integration - Enabling (actors, agents, components) to work jointly to achieve some task.

• Enabling the information flows that permits joint action.

• The Questions:
  – For Science & Technology generally:
    • What of integration can be automated?
    • What abilities do we lack?
  – For Systems Engineering:
    • What are the relevant viewpoints?
    • How do viewpoints compose and communicate?
    • What are the problems?
    • How are problems solved?
      – What is the systems engineering process?
What are the Relevant Viewpoints?

• A viewpoint on viewpoints, RM-ODP says:

  – **Enterprise**: purpose, scope, and policies of the system

  – **Information**: nature of information, constraints on its use and interpretation

  – **Computation**: functional decomposition into components, interaction and interfaces

  – **Engineering**: mechanisms and functions required to support integration

  – **Technology**: particular choice of technology
RM-ODP

- Assuming these viewpoints are sufficient (coverage seems right), what next?
  - What modeling notations populate these viewpoints?
    • ...and need they cleanly fit those categories?

  - Are other classifications of viewpoint also useful?
    • e.g. conceptual / technical

  - How is viewpoint communication implemented?
Challenges of being comprehensive

– **Consistency**: Do the concerns expressed in the various notations contradict each other?

– **Overlap**: Do the viewpoints tend to emphasize equivalent concerns?

– **Traceability**: How do I know whether a concern is being addressed amid the diversity of notations?

– **Composition**: Do the various notations complement each other?
Advantages of being comprehensive

• Using a common modeling core means:
  – More concise language definition
  – Inter-model coherence
    • Because viewpoint have a common vocabulary to ‘communicate,’ constraints can be written across models.
Common Core & Meta-Models

• Purpose of the meta-model:
  – Identify the fundamental modeling concepts that underlie the various notations (a *common core*)
  
  – So that we can reason about the models
    • Identify equivalence

  – May be a different usage of MOF!
Meta-models (2)

There are at least two kinds…

1: Axiomatic meta-model
   - formal definitions of fundamental modeling concepts based on a small set of informally defined terms (primitives).

2: Hierarchical / Structural meta-models:
   - modeling concepts defined as a population of the primitive terms.
   - Interpretation of primitives ultimately relies on another approach (axiomatic, probably).
   - OMG MOF, ISO IRDS
Which kind of Meta-Model?

- Aspects of both are needed:
  - The axiomatic approach helps one define constraints on usage and interpretation:
    - What do we really mean by “class”? 
  - The hierarchical / structural approach provides management of instances
    - And the models we use are composed of instances
      - e.g. in files encoded in XMI.
It becomes more difficult...

- The enterprise viewpoint concerns matters of purpose and policy
  - This may entail ontologies
    - These may be expressed in their own axiomatic language
    - And have no home in the hierarchical / structural framework
Part 2 : Automating Integration Tasks

• What stands in the way?
  – Technical Conflicts
  – Semantic Conflicts
  – Functional Conflicts
  – Quality of Service Conflicts
  – Logistical Conflicts

• A common core of modeling notions should enable reasoning about these!
  – If you can’t identify an equivalence in one of these dimensions, you can’t solve a problem in that dimension
Technical Conflicts

• Mismatch in the detail of behavior and information representation at the interface
  – Differing communication mechanism
  – Syntactic differences in data exchanged
  – Control conflicts

• Solutions?
  – Problem solvers with knowledge of interface technology
  – Mapping languages
  – Planning algorithms to synthesize coordination models
Semantic Conflicts

• Mismatch on concepts and terms leading to unexpected behavior

• *Not* about “preserving semantics” across information flows
  – The purpose of communication is to influence behavior

• Agreement on *sense*: models of objects and behaviors in the context of the role of the resource in the system

• Agreement on *reference*: the mapping of names to objects
Semantic Conflicts (2)

• Solutions?
  – Many research projects, past and present
  – Most problems require human intelligence
  – The Standard Upper Ontology may help
Functional Conflicts

• Behavior of a resource in joint action fails to provide expected function, or even interferes with achieving a function
  – mismatch in purpose
  – overlap in purpose
  – unexpected side-effect

• Solutions?
  – Self-description (behavioral, functional) of resources
    • (but not easy?)
Quality of Service Conflict

• A resource fails to support a non-functional requirement
  – performance
  – reliability
  – security
  – cost

• Solutions?
  – Diverse bodies of knowledge and an ability to appraise the non-functional abilities of a proposed design.
    • Difficult
Logistical Conflicts

• A matter of system validation:
  – Does the system *really* serve the requirements?
  – Have we provided for graceful evolution of the system?

• Solutions?
  – ?
Original Requirements

Model Repository (meta-object facility)

SE Executor
- System Concept
- System Solution
- ATMS

Enterprise Ontology

Conceptual Problem Solver
- View translation

Technical Problem Solver
- View translation

refinements and design commitments
Conclusions

• Finding the correct meta-model and identifying what modeling notions to distinguish is at the heart of an approach to MDA.

• A model-driven architecture that adequately addresses all 5 integration concerns may be a long way off.
  – But some very useful work seems possible now
Similar to IEEE 1471

• Viewpoint - a reusable template from which to build views; it defines well-formedness conditions on the views, and identifies an audience.

• View - a representation of the whole system from the perspective of a related set of concerns

• Model - the expression of a view conforming to a viewpoint’s well-formedness conditions.
Systems Engineering

- Any methodical approach to the synthesis of an entire system that:
  - defines views of that system that help elaborate requirements
  - manages the relationship of requirements to
    - performance measures,
    - constraints,
    - components
    - discipline-specific system views.
References

• NIST has a project to explore what can be automated and experiment with approaches:
  
  – *Concepts for Automating Systems Integration*  
    (Barkmeyer, et al.) Long report, to be published soon.
  
  
  – [Peter.Denno@nist.gov](mailto:Peter.Denno@nist.gov)