Dynamic Gate Product and Artifact Generation from System Models

Maddalena Jackson
Jet Propulsion Laboratory
California Institute of Technology
Overview

Document Generator ("Docgen"): Generate model content as a paper artifact ("document").

- **Stakeholder**: someone with a UML/SysML model
  - UML: Unified Modeling Language
  - SysML: Systems Modeling Language

- **Use-case**: stakeholder has to produce a document containing information in the model
  - Requirements documents, test procedures, n-squared matrices, architecture description documents, etc.
Docgen Context

• Current systems engineering paradigm: “Power Point Engineering”
  – Plethora of disjointed documents
  – Human overhead finding latest data
  – Documents obsolete immediately
  – Spaghetti diagrams
• Future SE paradigm: Model-Based Systems Engineering
  – Centralized “gold source” of information
  – Documents are views of the system
  – Diagrams conform to visual rules
• How do we transition?
Docgen Solution

• Reality of the transition to MBSE: legitimate need for artifacts
  – Human-friendly communication format
  – Engineers must provide and communicate evidence for claims about the system
  – Documents (of specific format) are often non-negotiable requirements

• Docgen provides an interface
  – A document is just a view into the model
  – Query models to produce documents
  – Store the view (document) specification with the model
What is a document?

• Information organized logically
  – Sequence
  – Depth

• Rules
  – Sections contain sections, text, figures, tables, etc.
  – Paragraphs do not contain sections, etc.

• Abstract this…
  – A document is a serialized view of a system model
What is Docgen?

- Developed for JPL AMMOS Ops Revitalization – model centric project
- Part 1: Semantics (UML profile)
  - Model logical structure of output (the view)
  - Link queries to system model
- Part 2: Execution
  - Traverse the logical structure
  - Execute queries and analysis
  - Serialize to DocBook XML
- Part 3: Provide user support
  - Model validation
  - Infrastructure support
Document Profile

Structural Elements:

- **Document**
  - `title`: String
  - `subtitle`: String
  - `index`: Boolean = true
  - `version`: String [1]
  - `debugMode`: Boolean

- **Section**
  - `FIRST`: Boolean
  - `NEXT`: [Section or Appendix]
  - `PREVIOUS`: [Sect, Para, Qry]
  - `Title`: String
  - `xref-ID`: String
  - `IGNORE`: Boolean

Other Properties:
- `Authors`

Inherited Properties:
- `FIRST`: Boolean
- `NEXT`: [Para, Qry, or Section]
- `PREVIOUS`: [Paragraph or Qry]
- `xref-ID`: String
- `IGNORE`: Boolean

Other Properties:
- `Body`: String

- **Paragraph**
  - `hasDocbook`: Boolean

Inherited Properties:
- `FIRST`: Boolean
- `NEXT`: [Para, Qry, or Section]
- `PREVIOUS`: [Paragraph or Qry]
- `xref-ID`: String
- `IGNORE`: Boolean

Other Properties:
- `Body`: String

- **Query**
  - `queryTargets`: NamedElement [0..*]
  - `queryTitles`: String [0..*]

Inherited Properties:
- `FIRST`: Boolean
- `NEXT`: [Para, Qry, or Section]
- `PREVIOUS`: [Paragraph or Qry]
- `xref-ID`: String
- `IGNORE`: Boolean

Other Properties:
- `Body`: String

Document “rules” enforced through profile relationships

Query is abstract; subclassed for specific analysis
Execution Implementation

• Behavior is logically tool-independent
  – Implemented in any framework with access to model and profile
  – We used MagicDraw UML’s Velocity Template Language and OpenAPI interface

• Serialize document model
  – Linked list
  – Hierarchical
Recursive section processing:

```
act [Activity] Process Section |
  «block» s : section
  Process Current Section
  subsection
  section
  content
  Process Query Element
  [contentContains]
  [content.type = query]
  next
  query
  Process Paragraph Element
  [content.type = paragraph]
  next
  paragraph
  current.hasNext
  return
```

3/29/2012
Using Docgen

• Docgen validation
  – Correct usage of first, next, previous properties
  – Maintenance of cross-referencing
  – Cosmetic section numbering
  – Pre-generation Diagnostics

• Docgen user convenience
  – Pop-up text editing

• Downstream Processing – producing a document:
Advantages and Applications

• Goal: interface between MBSE and document-centric paradigm
  – Leverage MBSE and document advantages
• Centralize models with document views; generic format
• Generate documents concurrent with system design
• Re-use of code; engineers focus on their expertise (not on coding or fighting with word processing software)

• Applications
  – JPL AMMOS Ops Revitalization: 2 Architecture Description Documents; successful model-based reviews;
  – JPL Integrated Model Centric Engineering Initiative; Ops Con
  – Mars Science Laboratory
  – Docgen user guide: Docgen test model produces the user manual
Conclusions and Future Work

• Conclusions:
  – Leverages power and extensibility of UML/SysML
  – Allows MBSE environment to produce documents satisfactory to current engineering paradigm
  – One-time labor investment produces perpetually current documentation
  – Docgen can help demonstrate feasibility and power of MBSE

• Future Work:
  – External generation (remotely from web)
  – Interface with executable models
  – Data exchange with external simulation tools
Backup (1)