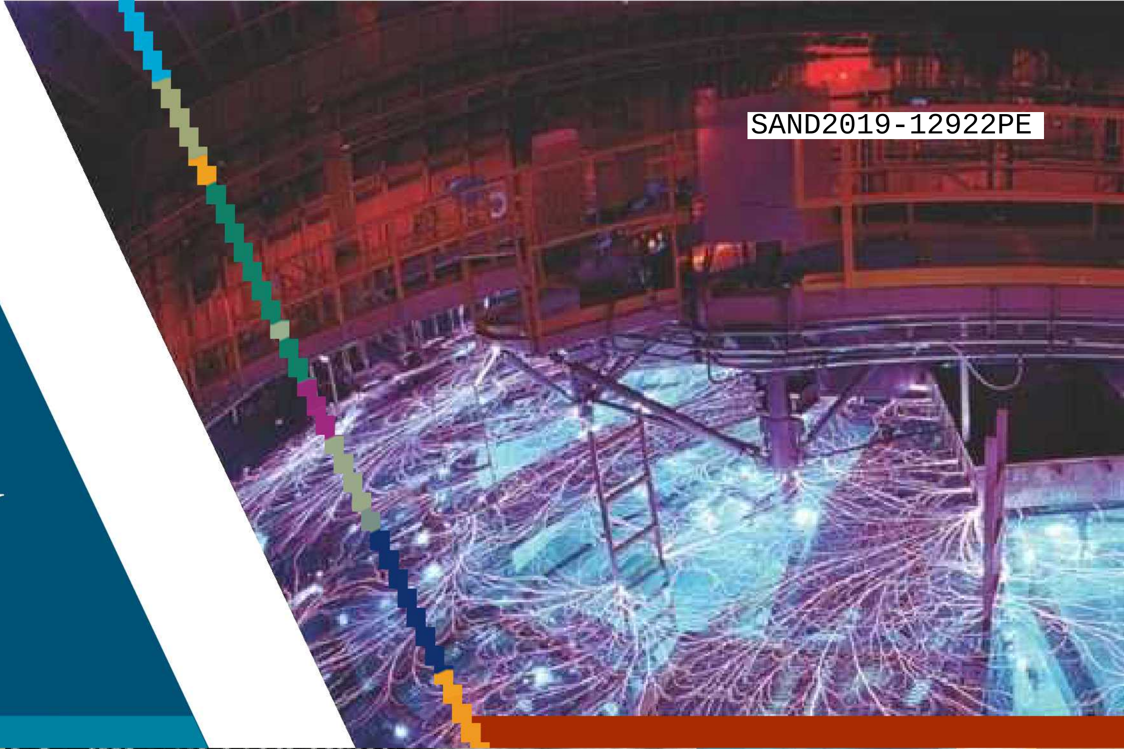
SAND2019-12922PE

Exceptional service in the national interest

Model Interoperability/Credibility Research Demonstration Results



SANDIA NATIONAL LABORATORIES

Ed Carroll (PI), John McCloud, Carlos Tafoya, Pete Chandler, Jonathan Compton, Cengiz Akinli(GT), Alicia Sudol(GT), Jason Jarosz, Sebastian Quimbay, Will McLendon

UPDATED FEBRUARY 2019

Model Interoperability / Credibility

The DOD has placed significant emphasis on digital engineering.
Why? Because digital models contain significantly more information than paper.
<u>However, a digital strategy assumes that models interoperate.</u>

The USN has engaged with Sandia National Laboratories
to develop digital engineering models

The USN has asked to receive copies of our models

When modeling tools do not interoperate, your tools cannot read our models

Translating the model to PDF or paper looses significant context/information

Does SSP/USN have a plan for model sharing/interoperability?

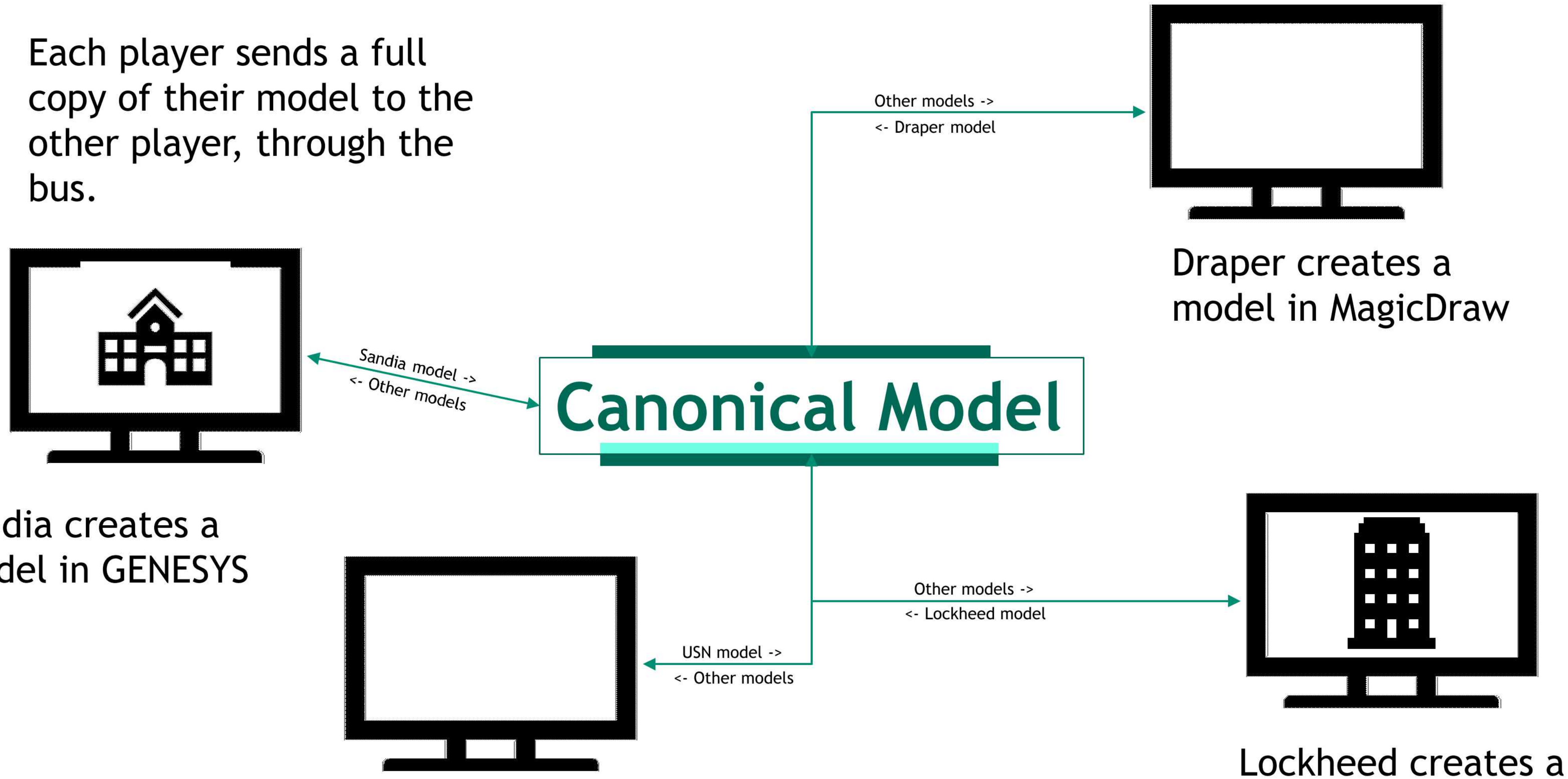
– across tools and layers of fidelity?

- We estimate that just at Sandia, our portion of a full weapon system will include a 1,000 models
- When we make models the source of truth, how will we ensure they are authoritative, trusted, credible?

Model Interoperability / Credibility

Example Use Case: Full Model Transfer/Transformation (USN+Contractors)

Each player sends a full copy of their model to the other player, through the bus.



Draper creates a model in MagicDraw

Sandia creates a model in GENESYS

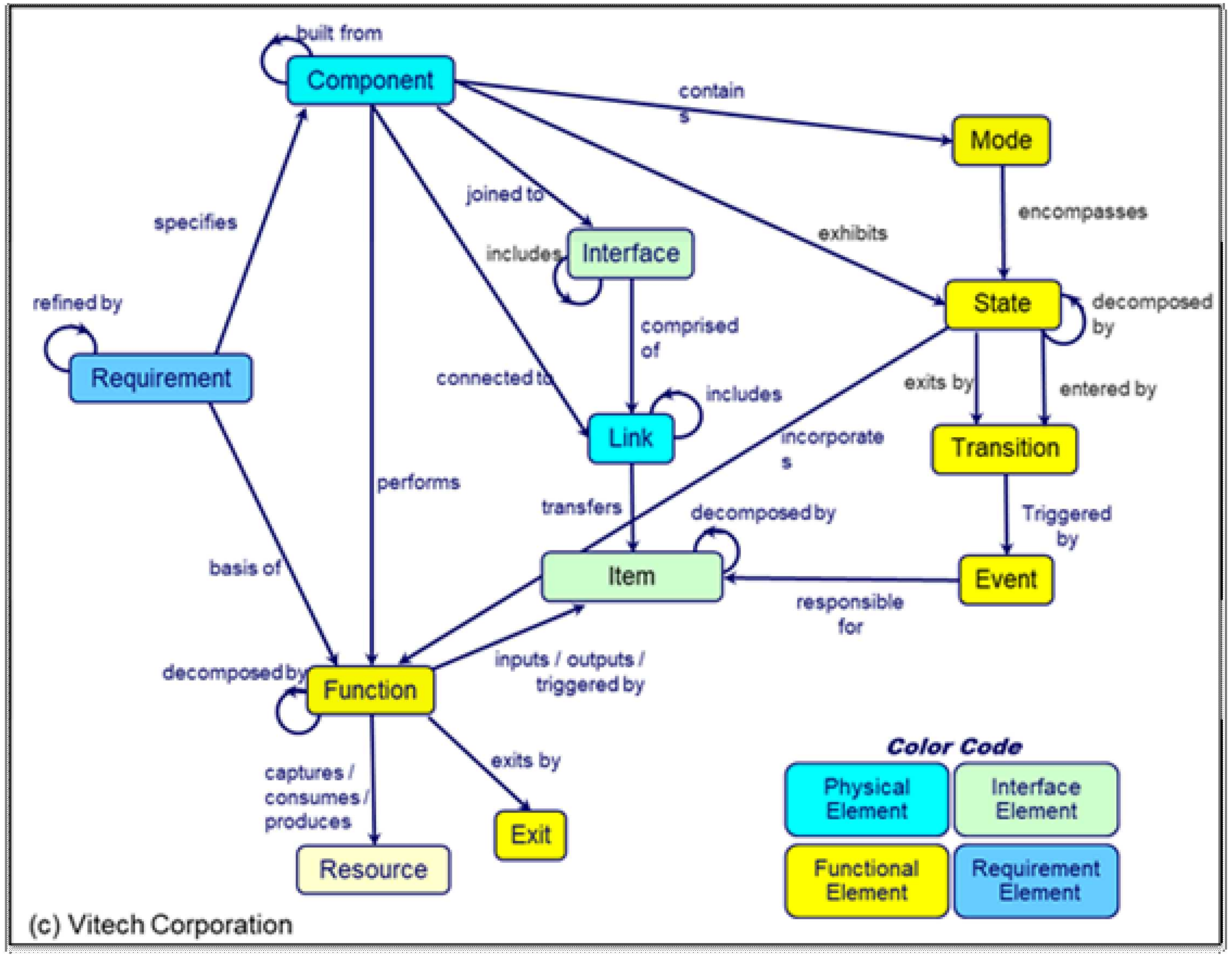
Lockheed creates a model in MagicDraw

The USN receives all models into separate models in MagicDraw

Model Interoperability / Credibility

Unlike Traditional Databases, Systems Engineering Models Have Structure

Traditional databases do not retain the integrity of an ontological structure, so they lose data (the model meaning)



Whereas, a standard ontologically well formed canonical reference model provides the foundational structure for object mapping and comparison

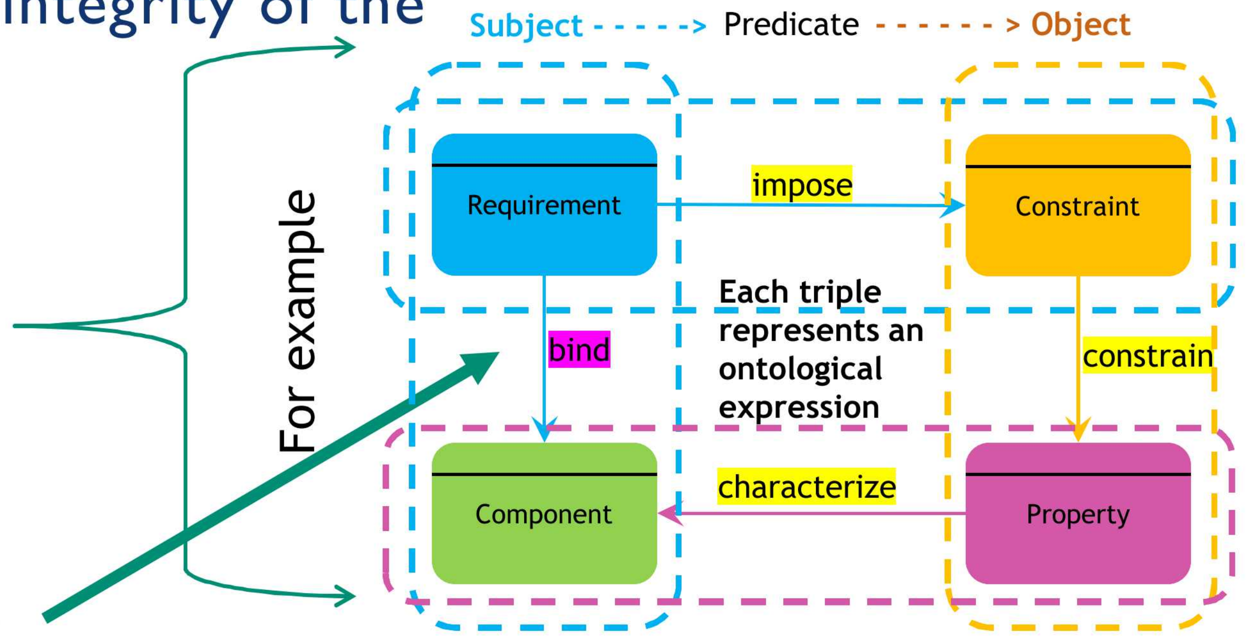
In a manner that preserves that model structure (a graph data structure)

Component – performs – Function
Function – ouputs – Item
Function – triggered_by – Item
Component - built_from – Component
Component – connected_to – Link
Component – documented_by – Document
Component – joined_to – Interface
Component – provided_by – Organization
Environment – experienced_during – Phase
Link – connects_to – Component
Product – triggers – Program Activity

Model Interoperability / Credibility

Our Approach: Can we enable (Systems) engineering models to interoperate if we retain the integrity of the underlying ontological-structure?

What is an ontology? Models derive meaning from their structure. For SE models, in the form of: facts, roles, relationships, and intent, the way a sentence does in natural language processing (NLP).


For example
Subject - - - - -> Predicate - - - - - - > Object
Requirement
impose
Constraint
bind
constrain
Each triple represents an ontological expression
Component
characterize
Property

Challenge Example: When we transfer data related to "**Requirements** **bind** **Components**", we must also capture the other three triples or the model meaning is lost and we cannot analyze accurately. This is called the compositionality of the model, where the meaning of a complex expression is systematically put together from the meanings of its parts. The challenge is that each object may have many unrelated relationships to any specific question, so we are researching if we can reason through the model navigation to only the triples that relate to the questions being asked in the analysis.

NLP reasons the meaning in a sentence.

In the model above are four triple expressions:

- **Requirements** impose **Constraints**;
- **Constraints** constrain **Properties**;
- **Property** characterizes **Components**;
- **Requirements** bind **Components**

Based on the first three triples, we can reason that Requirements bind Components. We need all four triples in this structure to understand <u>why</u> Requirements bind Components.

Model Interoperability / Credibility

Another Example

Make sure we use a Safety Harness to transport the rocket, and transport it on Low-Traffic Roads.

Semantic Disconnect

Same words; same data types.
Meaning different things.

**An Ontology recognizes this semantic disconnect.
An Ontology knows that these two separate meanings are not equivalent.**

What he Said!
What he Meant!

<u>Low-Traffic Roads</u> Any road with < 5,000 cars a day
<u>Safety Harness:</u> Any suspension system with belts having 5-points or more

What he Understood!

<u>Low-Traffic Road:</u> Any road that is not a highway
<u>Safety Harness:</u> Any harness with 3-points or more

Model Interoperability / Credibility

Translating Languages (One Language and back)

English

- It's always hard to explain puns to kleptomaniacs because they're always taking things literally.

Chinese (Traditional)

- 總是很難向雙關語者解釋雙關語，因為他們總是從字面上理解事物。

Back to English

- It is always difficult to explain puns to puns because they always understand things literally.

Model Interoperability / Credibility

Translating Languages (Round trip with intermediary language)

English
- It’s always hard to explain puns to kleptomaniacs because they’re always taking things literally.

Chinese (Traditional)
- 總是很難向雙關語者解釋雙關語，因為他們總是從字面上理解事物。

French
- Il est toujours difficile d’expliquer les jeux de mots aux joueurs car ils comprennent toujours les choses au sens littéral.

Back to Chinese(Traditional)
- 向玩家解釋文字遊戲總是很困難，因為他們總是從字面意義上理解事物。

Back to English
- It is always difficult to explain word games to players because they always understand things literally.

Model Interoperability / Credibility

Our Model Translation Problem

Genesys -> RDF

RDF -> Magic Draw

Magic Draw -> RDF

RDF -> Genesys

What is RDF?

- Resource Description Framework - a file format that conforms to the ontology structure
  - As an alternative to a traditional database structure
  - So, by translating the model into RDF format, we retain the integrity of the ontological structure of the model.

Why RDF?

- We can use natural language reasoning rules to assess the model for consistency (does it follow good ontological rules?)
  - Including translation issues related to alignment between the target model's ontology and that of the reference model
- We can also use natural language reasoning rules to assess for programmatic requirements (is the data, data format, context correct>)

Model Interoperability / Credibility

Genesys to RDF mapping.

Entity Definitions => Classes

- The subclass predicate allows us to maintain inheritance and polymorphism.
- This doesn't account for the Entity Definition's attributes

Relationship Definitions => Classes

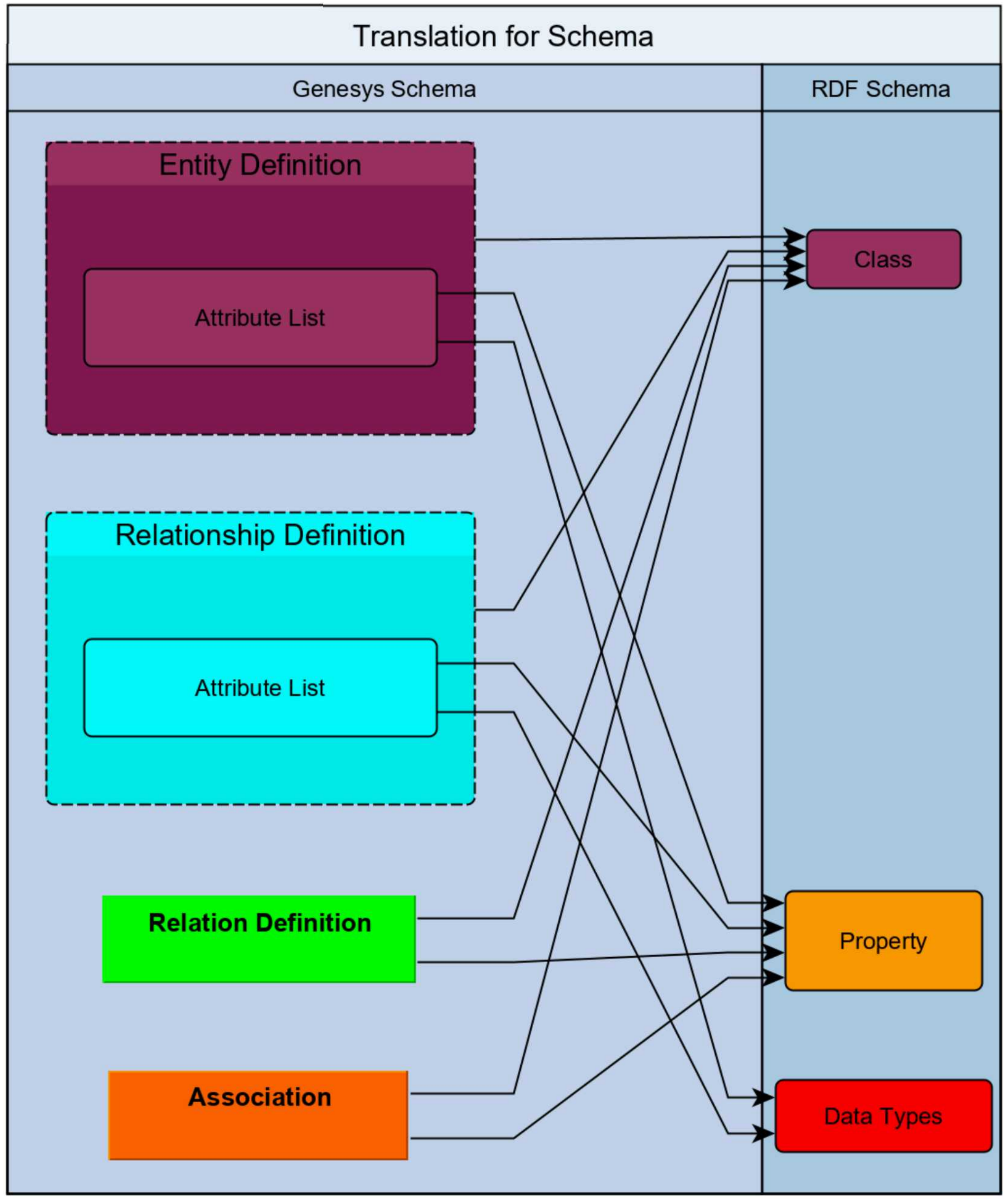
- This doesn't account for the Relationship Definition's attributes

Attributes => Properties and Data Types

- We also require the use of the Domain and Range predicates

Relation Definitions and Associations => Classes and Properties

- These two must be translated with each other in mind. Together they let Entities and Relationships be constrained.

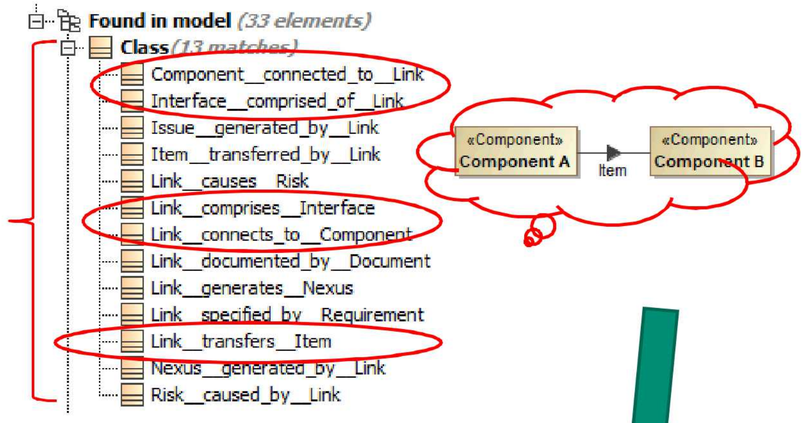


Model Interoperability / Credibility

Then we translated the GENESYS/RDF model into MagicDraw

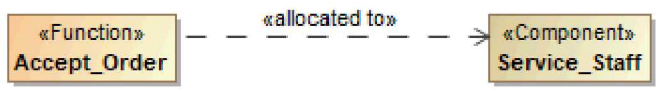
GENESYS/RDF Entity definitions are imported as custom stereotypes.

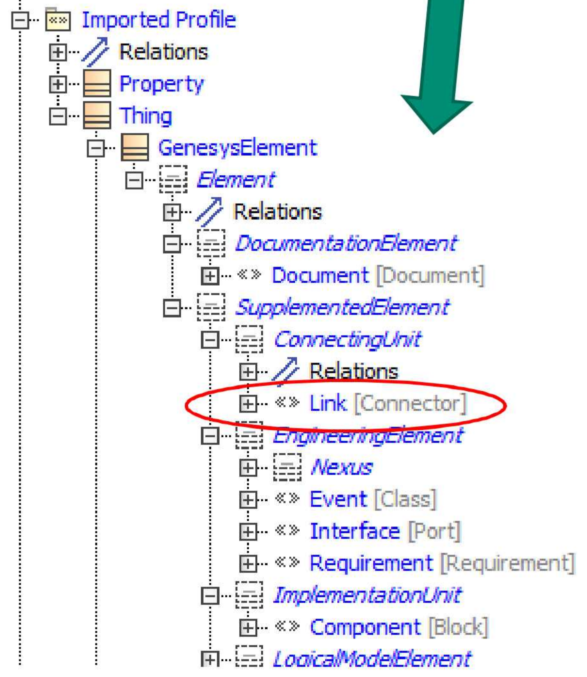
- The GENESYS/RDF schema (particularly, the relationships allowed between objects) was subjectively compared to SysML and UML object definitions and model semantics to determine equivalency. Custom stereotypes preserve the semantic meaning of the system model.



Found in model (33 elements)
- Class (13 matches)
  - Component__connected_to__Link
  - Interface__comprised_of__Link
  - Issue__generated_by__Link
  - Item__transferred_by__Link
  - Link__causes__Risk
  - Link__comprises__Interface
  - Link__connects_to__Component
  - Link__documented_by__Document
  - Link__generates__Nexus
  - Link__specified_by__Requirement
  - Link__transfers__Item
  - Nexus__generated_by__Link
  - Risk__caused_by__Link

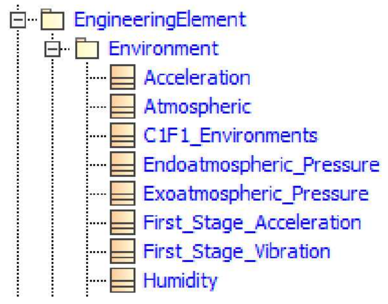
Genesys relationships are mapped to equivalent SysML/UML relationships.





- Imported Profile
  - Relations
  - Property
  - Thing
    - GenesysElement
      - *Element*
        - Relations
        - *DocumentationElement*
          - «» Document [Document]
        - *SupplementedElement*
          - *ConnectingUnit*
            - Relations
            - «» Link [Connector]
          - *EngineeringElement*
            - *Nexus*
            - «» Event [Class]
            - «» Interface [Port]
            - «» Requirement [Requirement]
          - *ImplementationUnit*
            - «» Component [Block]
          - *LogicalModelElement*

Usage of custom stereotypes permits the preservation of attributes defined for GENESYS/RDF entities and relationships.



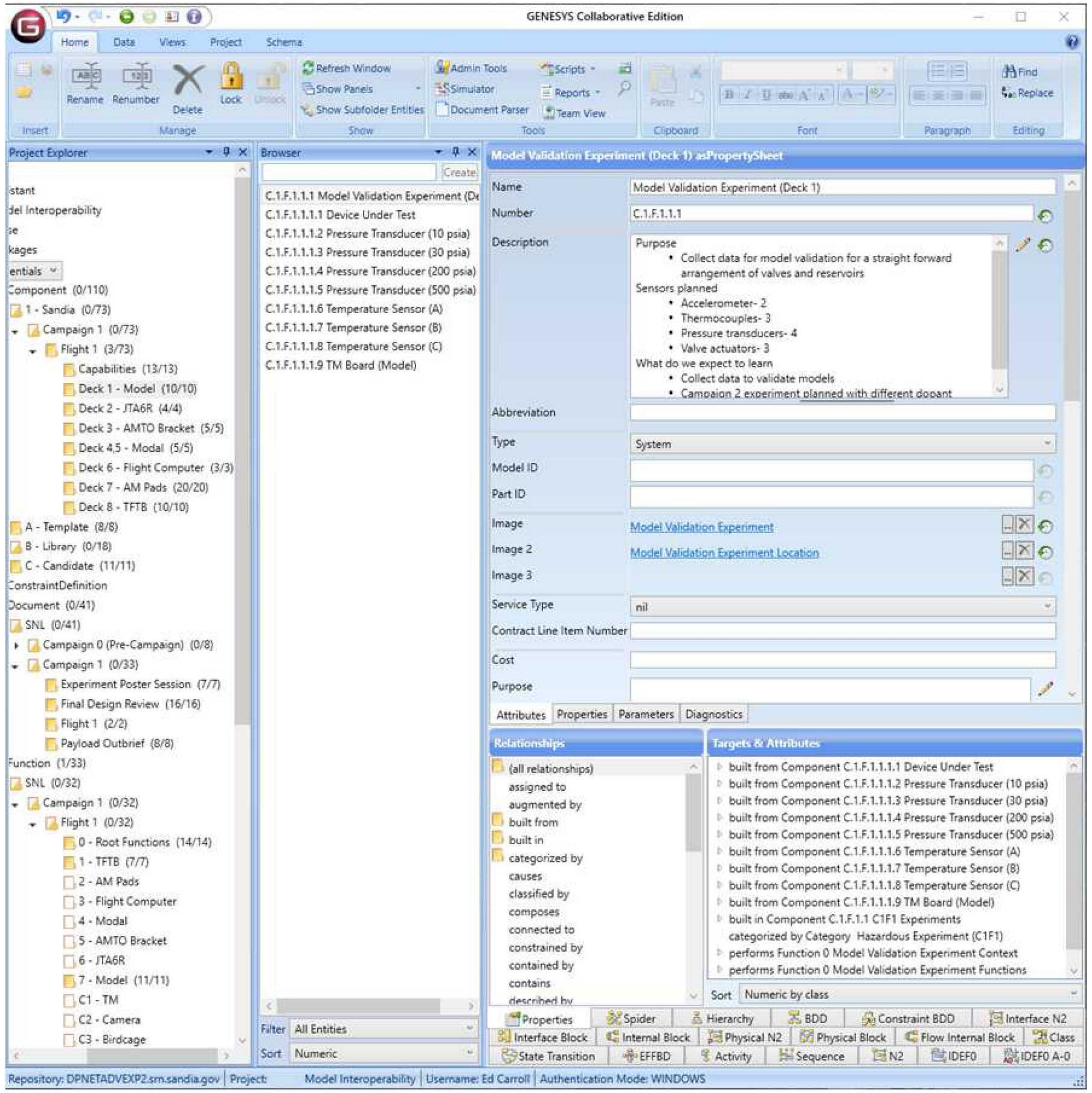
- EngineeringElement
  - Environment
    - Acceleration
    - Atmospheric
    - C1F1_Environments
    - Endoatmospheric_Pressure
    - Exoatmospheric_Pressure
    - First_Stage_Acceleration
    - First_Stage_Vibration
    - Humidity

100% of RDF individuals (GENESYS entities) are imported and placed within the model.

Model Interoperability / Credibility

Our Demo: We started with 2 GENESYS models (to test with)

A model from HotShot



Project Explorer:

- stant
- del Interoperability
- e
- kages
- entials
- Component (0/110)
  - 1 - Sandia (0/73)
    - Campaign 1 (0/73)
      - Flight 1 (3/73)
        - Capabilities (13/13)
        - Deck 1 - Model (10/10)
        - Deck 2 - JTA6R (4/4)
        - Deck 3 - AMTO Bracket (5/5)
        - Deck 4,5 - Modal (5/5)
        - Deck 6 - Flight Computer (3/3)
        - Deck 7 - AM Pads (20/20)
        - Deck 8 - TFTB (10/10)
  - A - Template (8/8)
  - B - Library (0/18)
  - C - Candidate (11/11)
- ConstraintDefinition
- Document (0/41)
  - SNL (0/41)
    - Campaign 0 (Pre-Campaign) (0/8)
    - Campaign 1 (0/33)
      - Experiment Poster Session (7/7)
      - Final Design Review (16/16)
      - Flight 1 (2/2)
      - Payload Outbrief (8/8)
- Function (1/33)
  - SNL (0/32)
    - Campaign 1 (0/32)
      - Flight 1 (0/32)
        - 0 - Root Functions (14/14)
        - 1 - TFTB (7/7)
        - 2 - AM Pads
        - 3 - Flight Computer
        - 4 - Modal
        - 5 - AMTO Bracket
        - 6 - JTA6R
        - 7 - Model (11/11)
        - C1 - TM
        - C2 - Camera
        - C3 - Birdcage

Browser:

- C.1.F.1.1.1 Model Validation Experiment (De
- C.1.F.1.1.1.1 Device Under Test
- C.1.F.1.1.1.2 Pressure Transducer (10 psia)
- C.1.F.1.1.1.3 Pressure Transducer (30 psia)
- C.1.F.1.1.1.4 Pressure Transducer (200 psia)
- C.1.F.1.1.1.5 Pressure Transducer (500 psia)
- C.1.F.1.1.1.6 Temperature Sensor (A)
- C.1.F.1.1.1.7 Temperature Sensor (B)
- C.1.F.1.1.1.8 Temperature Sensor (C)
- C.1.F.1.1.1.9 TM Board (Model)

Filter: All Entities
Sort: Numeric

Model Validation Experiment (Deck 1) asPropertySheet

Name	Model Validation Experiment (Deck 1)
Number	C.1.F.1.1.1
Description	Purpose • Collect data for model validation for a straight forward arrangement of valves and reservoirs Sensors planned • Accelerometer- 2 • Thermocouples- 3 • Pressure transducers- 4 • Valve actuators- 3 What do we expect to learn • Collect data to validate models • Campaign 2 experiment planned with different dopant
Abbreviation	
Type	System
Model ID	
Part ID	
Image	Model Validation Experiment
Image 2	Model Validation Experiment Location
Image 3	
Service Type	nil
Contract Line Item Number	
Cost	
Purpose	

Relationships: (all relationships), assigned to, augmented by, built from, built in, categorized by, causes, classified by, composes, connected to, constrained by, contained by, contains, described by

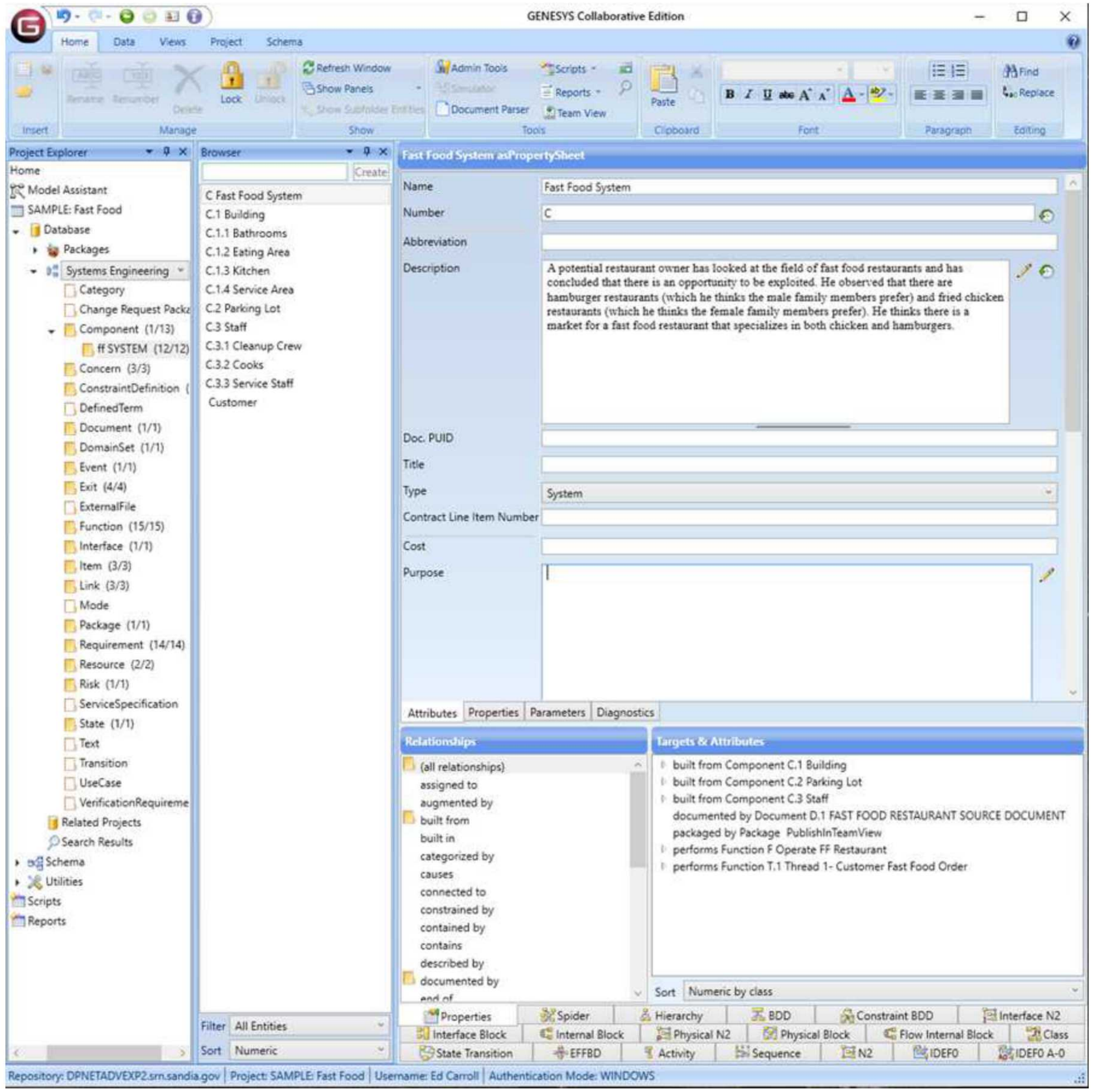
Targets & Attributes:

- built from Component C.1.F.1.1.1.1 Device Under Test
- built from Component C.1.F.1.1.1.2 Pressure Transducer (10 psia)
- built from Component C.1.F.1.1.1.3 Pressure Transducer (30 psia)
- built from Component C.1.F.1.1.1.4 Pressure Transducer (200 psia)
- built from Component C.1.F.1.1.1.5 Pressure Transducer (500 psia)
- built from Component C.1.F.1.1.1.6 Temperature Sensor (A)
- built from Component C.1.F.1.1.1.7 Temperature Sensor (B)
- built from Component C.1.F.1.1.1.8 Temperature Sensor (C)
- built from Component C.1.F.1.1.1.9 TM Board (Model)
- built in Component C.1.F.1.1 C1F1 Experiments
- categorized by Category Hazardous Experiment (C1F1)
- performs Function 0 Model Validation Experiment Context
- performs Function 0 Model Validation Experiment Functions

Sort: Numeric by class

Repository: DPNETADVEXP2.srn.sandia.gov | Project: Model Interoperability | Username: Ed Carroll | Authentication Mode: WINDOWS

And a Fast Food Restaurant model



Project Explorer:

- Home
- Model Assistant
- SAMPLE: Fast Food
- Database
  - Packages
  - Systems Engineering
    - Category
    - Change Request Packa
    - Component (1/13)
      - ff SYSTEM (12/12)
    - Concern (3/3)
    - ConstraintDefinition (
    - DefinedTerm
    - Document (1/1)
    - DomainSet (1/1)
    - Event (1/1)
    - Exit (4/4)
    - ExternalFile
    - Function (15/15)
    - Interface (1/1)
    - Item (3/3)
    - Link (3/3)
    - Mode
    - Package (1/1)
    - Requirement (14/14)
    - Resource (2/2)
    - Risk (1/1)
    - ServiceSpecification
    - State (1/1)
    - Text
    - Transition
    - UseCase
    - VerificationRequireme
  - Related Projects
  - Search Results
- Schema
- Utilities
- Scripts
- Reports

Browser:

- C Fast Food System
- C.1 Building
- C.1.1 Bathrooms
- C.1.2 Eating Area
- C.1.3 Kitchen
- C.1.4 Service Area
- C.2 Parking Lot
- C.3 Staff
- C.3.1 Cleanup Crew
- C.3.2 Cooks
- C.3.3 Service Staff
- Customer

Filter: All Entities
Sort: Numeric

Fast Food System asPropertySheet

Name	Fast Food System
Number	C
Abbreviation	
Description	A potential restaurant owner has looked at the field of fast food restaurants and has concluded that there is an opportunity to be exploited. He observed that there are hamburger restaurants (which he thinks the male family members prefer) and fried chicken restaurants (which he thinks the female family members prefer). He thinks there is a market for a fast food restaurant that specializes in both chicken and hamburgers.
Doc. PUID	
Title	
Type	System
Contract Line Item Number	
Cost	
Purpose	

Relationships: (all relationships), assigned to, augmented by, built from, built in, categorized by, causes, connected to, constrained by, contained by, contains, described by, documented by, end of

Targets & Attributes:

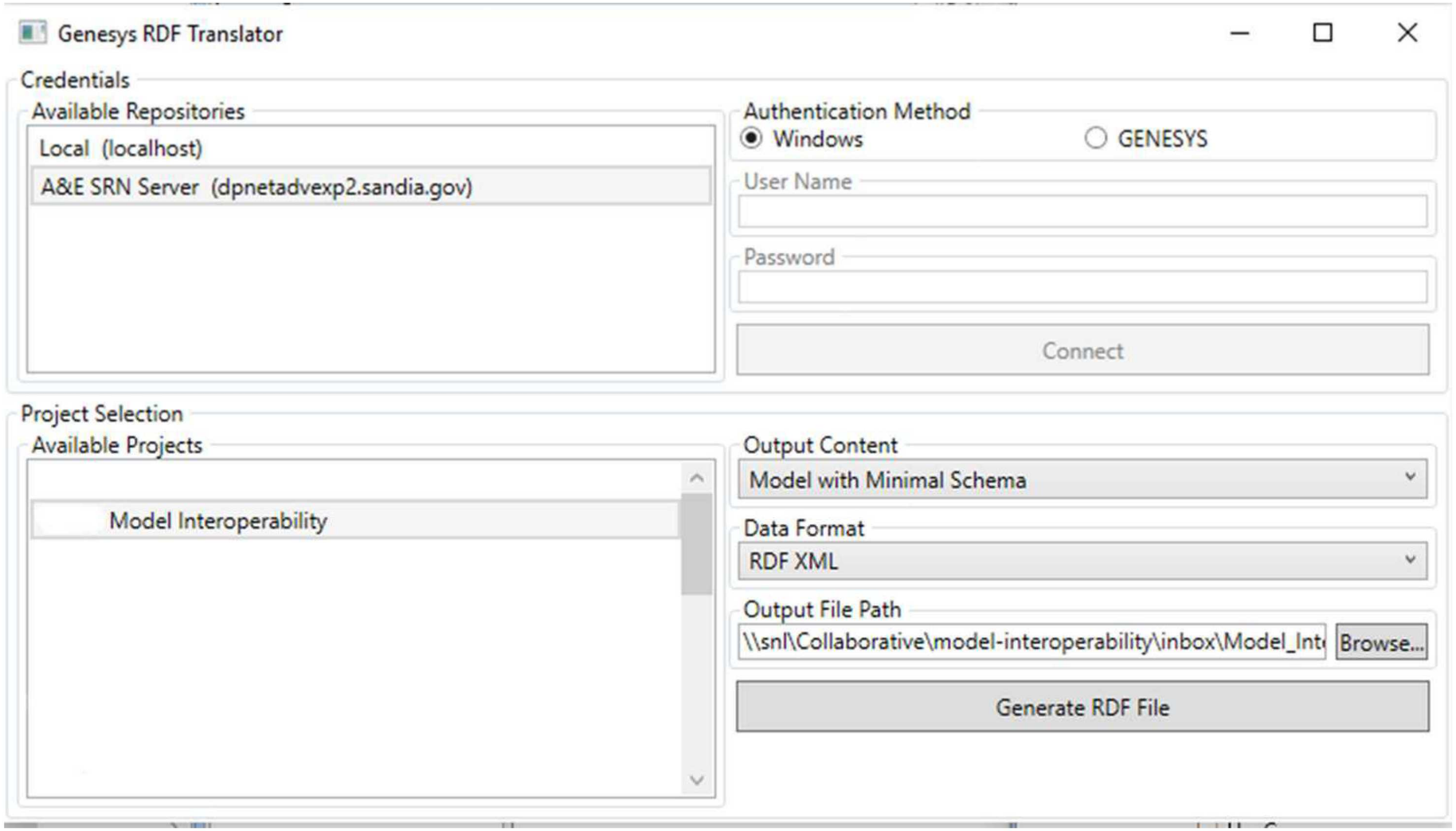
- built from Component C.1 Building
- built from Component C.2 Parking Lot
- built from Component C.3 Staff
- documented by Document D.1 FAST FOOD RESTAURANT SOURCE DOCUMENT
- packaged by Package PublishInTeamView
- performs Function F Operate FF Restaurant
- performs Function T.1 Thread 1- Customer Fast Food Order

Sort: Numeric by class

Repository: DPNETADVEXP2.srn.sandia.gov | Project: SAMPLE: Fast Food | Username: Ed Carroll | Authentication Mode: WINDOWS

Model Interoperability / Credibility

Our Demo: We developed an Application Program Interface (API) that extracted the entire models from GENESYS and transformed them into RDF formatted files



Genesys RDF Translator

Credentials

Available Repositories
- Local (localhost)
- A&E SRN Server (dpnetadvexp2.sandia.gov)

Authentication Method: [✓ Windows] GENESYS

User Name:

Password:

Connect

Project Selection

Available Projects
- Model Interoperability

Output Content: Model with Minimal Schema

Data Format: RDF XML

Output File Path: \\snl\Collaborative\model-interoperability\inbox\Model_Int

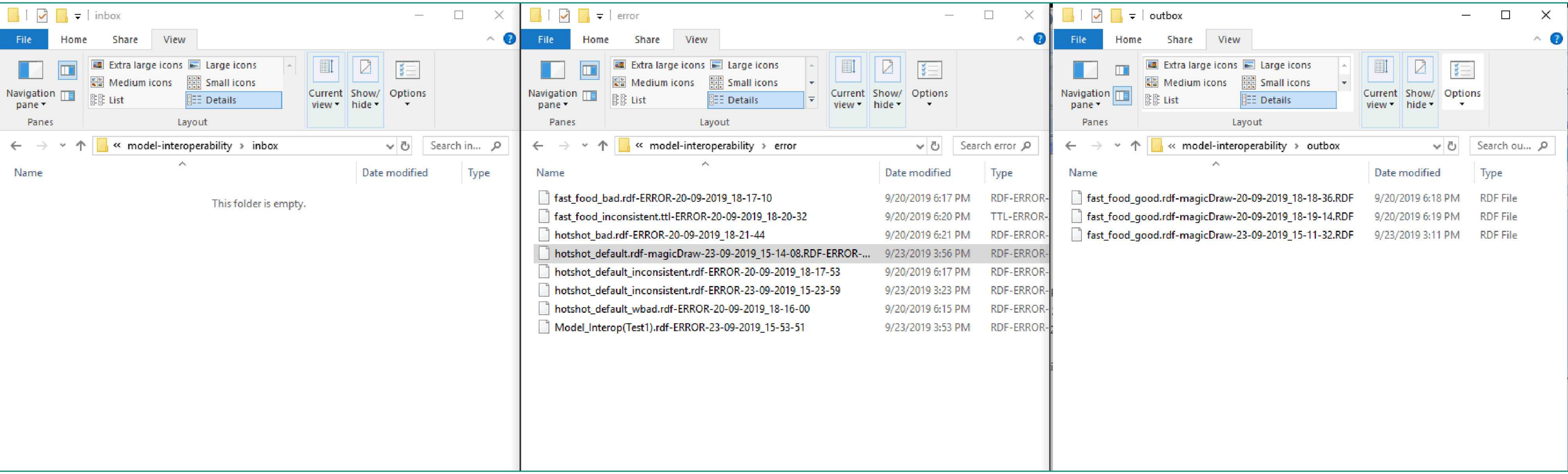
Browse...

Generate RDF File

Model Interoperability / Credibility

Our Demo: We also created a Service Bus to process the model transfer and apply our reasoning rules against the model – in transit

1. The Bus picks up the RDF file from the INBOX and passes it to the Reasoner.
2. If it fails the reasoning rules the Bus moves the RDF file to the ERROR box.
3. If it passes the reasoning rules, the RDF file is moved to the OUTBOX



inbox — « model-interoperability › inbox

Name	Date modified	Type

This folder is empty.

error — « model-interoperability › error

Name	Date modified	Type
fast_food_bad.rdf-ERROR-20-09-2019_18-17-10	9/20/2019 6:17 PM	RDF-ERROR-
fast_food_inconsistent.ttl-ERROR-20-09-2019_18-20-32	9/20/2019 6:20 PM	TTL-ERROR-
hotshot_bad.rdf-ERROR-20-09-2019_18-21-44	9/20/2019 6:21 PM	RDF-ERROR-
hotshot_default.rdf-magicDraw-23-09-2019_15-14-08.RDF-ERROR-...	9/23/2019 3:56 PM	RDF-ERROR-
hotshot_default_inconsistent.rdf-ERROR-20-09-2019_18-17-53	9/20/2019 6:17 PM	RDF-ERROR-
hotshot_default_inconsistent.rdf-ERROR-23-09-2019_15-23-59	9/23/2019 3:23 PM	RDF-ERROR-
hotshot_default_wbad.rdf-ERROR-20-09-2019_18-16-00	9/20/2019 6:15 PM	RDF-ERROR-
Model_Interop(Test1).rdf-ERROR-23-09-2019_15-53-51	9/23/2019 3:53 PM	RDF-ERROR-

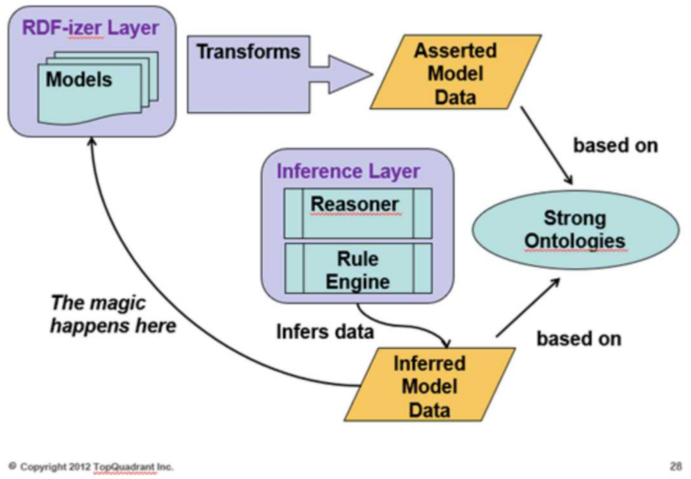
outbox — « model-interoperability › outbox

Name	Date modified	Type
fast_food_good.rdf-magicDraw-20-09-2019_18-18-36.RDF	9/20/2019 6:18 PM	RDF File
fast_food_good.rdf-magicDraw-20-09-2019_18-19-14.RDF	9/20/2019 6:19 PM	RDF File
fast_food_good.rdf-magicDraw-23-09-2019_15-11-32.RDF	9/23/2019 3:11 PM	RDF File

Not your typical science project. This was a proof-of-concept that combined 4 well known sciences into a new approach. When it is time to transition from research to the development team, we are ready with a standard approach, patterns, and tooling architecture.

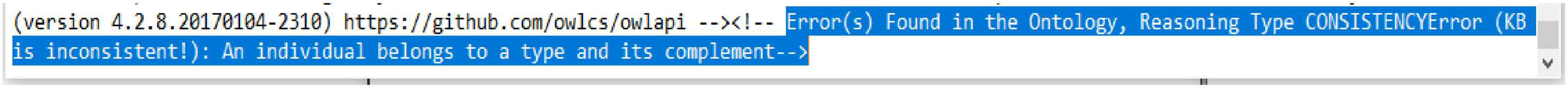
Model Interoperability / Credibility

Our Demo: We reason against the model's ontology Demonstrating that we can assess a model for credibility



We chose the Pellet reasoner in conjunction with Jena:

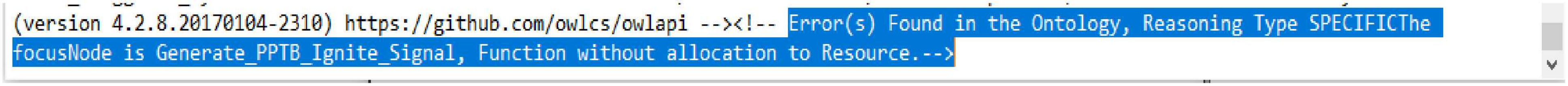
- Jena is an inference subsystem designed to allow a range of reasoners as plugins.
- Pellet is an open-source Java OWL 2 reasoner – an inference engine used to derive RDF assertions



```
(version 4.2.8.20170104-2310) https://github.com/owlcs/owlapi --><!-- Error(s) Found in the Ontology, Reasoning Type CONSISTENCYError (KB
is inconsistent!): An individual belongs to a type and its complement-->
```

We chose SHACL (shapes constraint language) reasoning language:

- To validate RDF graphs against a set of conditions (i.e., programmatic Systems Engineering model requirements)

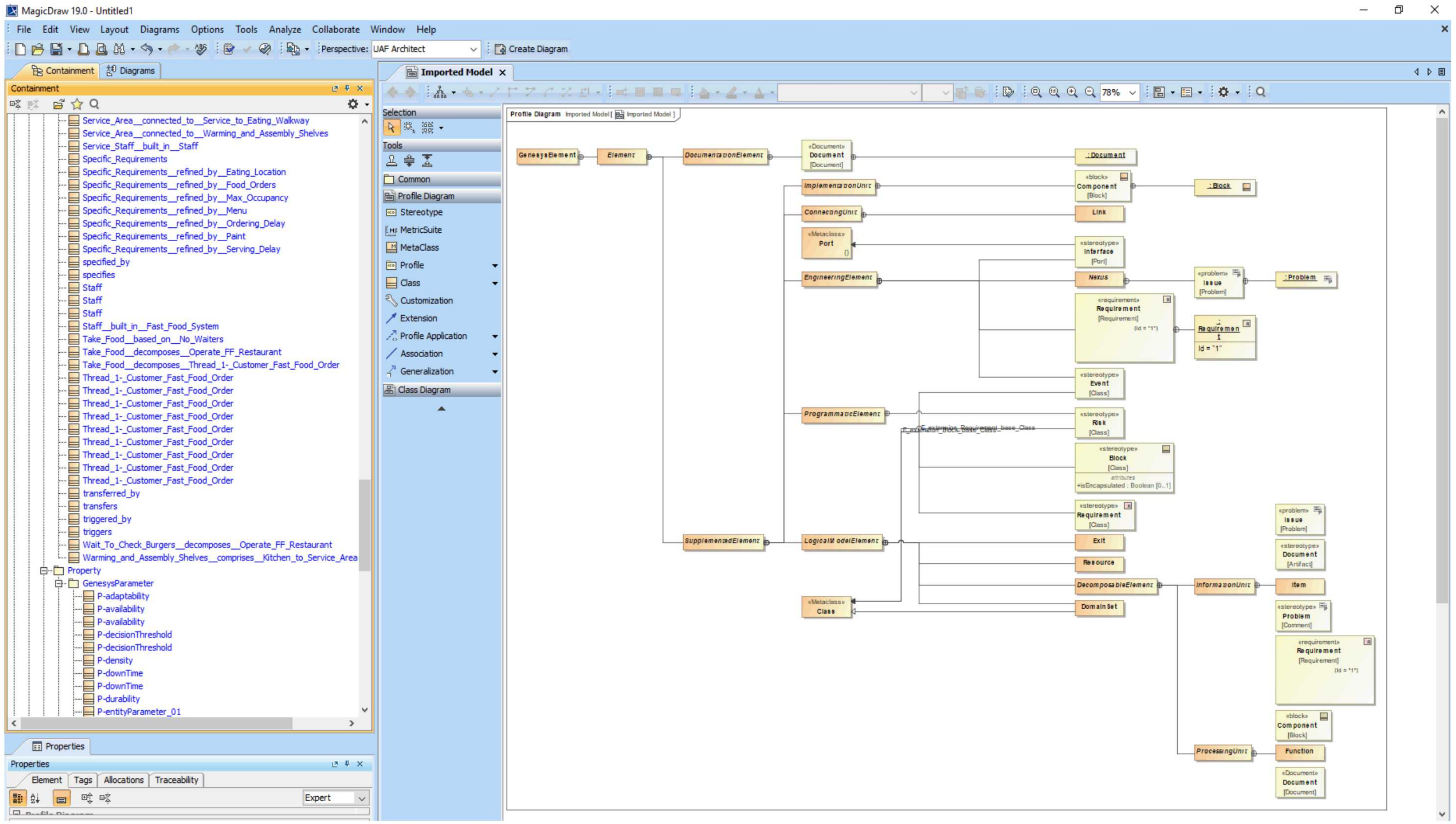


```
(version 4.2.8.20170104-2310) https://github.com/owlcs/owlapi --><!-- Error(s) Found in the Ontology, Reasoning Type SPECIFICThe
focusNode is Generate_PPTB_Ignite_Signal, Function without allocation to Resource.-->
```

Model Interoperability / Credibility

Our Demo: The (passed) RDF file is picked up from the OUTBOX and ingested into MagicDraw

Below is the Fast Food Restaurant model


MagicDraw 19.0 - Untitled1
File Edit View Layout Diagrams Options Tools Analyze Collaborate Window Help
Perspective: UAF Architect
Create Diagram
Containment
Diagrams
Imported Model
Service_Area__connected_to__Service_to_Eating_Walkway
Service_Area__connected_to__Warming_and_Assembly_Shelves
Service_Staff__built_in__Staff
Specific_Requirements
Specific_Requirements__refined_by__Eating_Location
Specific_Requirements__refined_by__Food_Orders
Specific_Requirements__refined_by__Max_Occupancy
Specific_Requirements__refined_by__Menu
Specific_Requirements__refined_by__Ordering_Delay
Specific_Requirements__refined_by__Paint
Specific_Requirements__refined_by__Serving_Delay
specified_by
specifies
Staff
Staff
Staff
Staff__built_in__Fast_Food_System
Take_Food__based_on__No_Waiters
Take_Food__decomposes__Operate_FF_Restaurant
Take_Food__decomposes__Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
Thread_1-_Customer_Fast_Food_Order
transferred_by
transfers
triggered_by
triggers
Wait_To_Check_Burgers__decomposes__Operate_FF_Restaurant
Warming_and_Assembly_Shelves__comprises__Kitchen_to_Service_Area
Property
GenesysParameter
P-adaptability
P-availability
P-availability
P-decisionThreshold
P-decisionThreshold
P-density
P-downTime
P-downTime
P-durability
P-entityParameter_01
Properties
Element Tags Allocations Traceability
Expert
Selection
Tools
Common
Profile Diagram
Stereotype
MetricSuite
MetaClass
Profile
Class
Customization
Extension
Profile Application
Association
Generalization
Class Diagram
Profile Diagram Imported Model
GenesysElement
Element
DocumentationElement
ImplementationUnit
ConnectingUnit
Port
EngineeringElement
ProgrammaticElement
SupplementedElement
LogicalModelElement
Class
Document
Component
Link
Interface
Nexus
Requirement
Event
Risk
Block
Exit
Resource
DecomposableElement
DomainSet
InformationUnit
ProcessingUnit
Issue
Problem
Item
Function

The results are not an identical match (because this was a proof-of-concept), but 100% of the data has been verified.

If you are a MagicDraw user, you should see that we have mapped the GENESYS objects into MagicDraw Stereotypes

Model Interoperability / Credibility

Research direction in FY20 – From Model Interoperability to Model Credibility

In FY19 we:

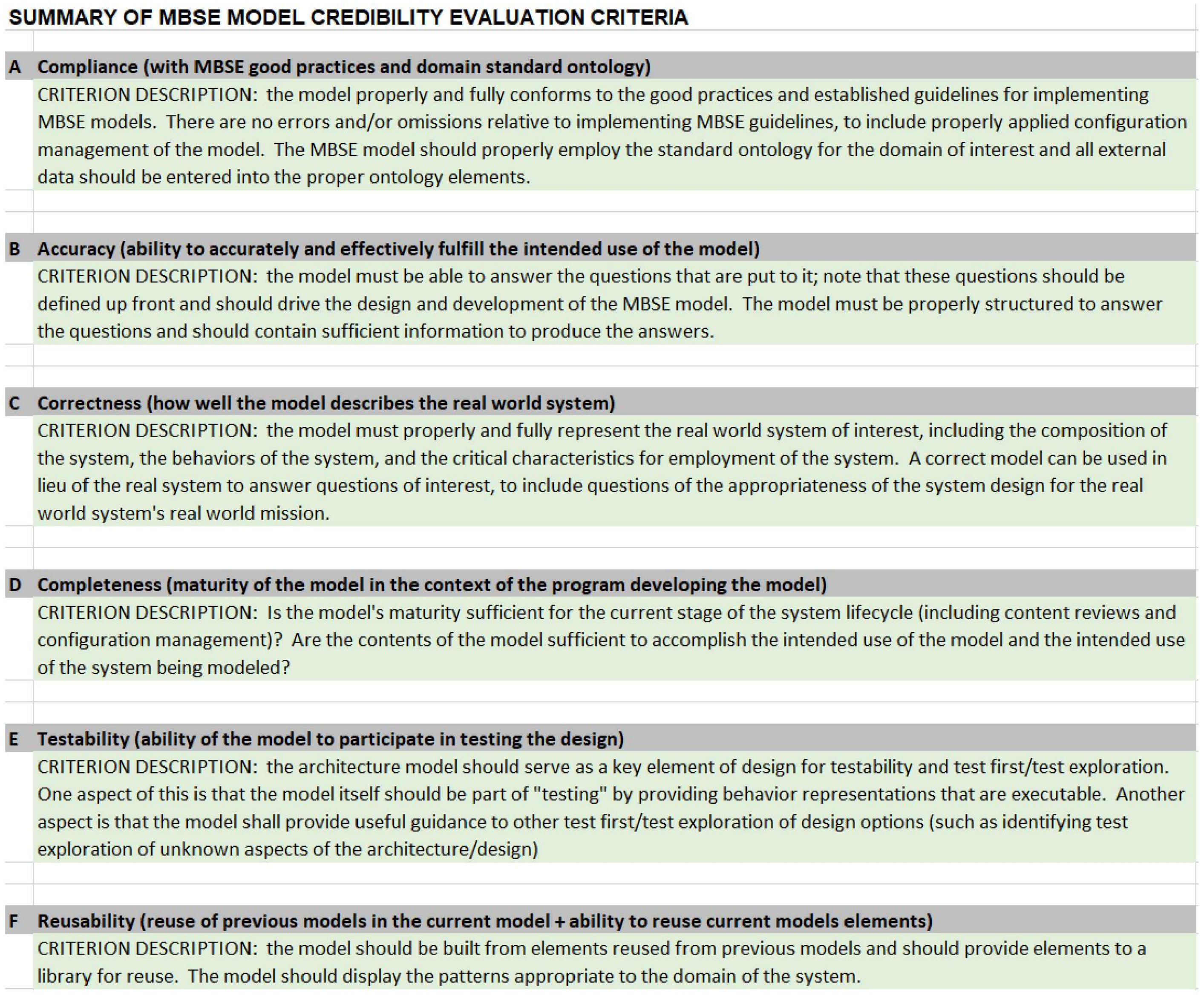
- Conducted preliminary research on the use of graph theory against an ontology to expand our reasoning:
  - Is the model well formed (OWL 2.0 compliant)?
  - Does the model conform to the reference ontology?
  - Are certain required model objects present?

In a tangential FY19 (Corporate SE Ontology) study:

- Assembled a list of questions that a Systems Engineer should ask to assess whether a model is credible.
- Compared ontology objects from JPL, NASA and DoDAF
- Analyzed ISO 15288 to identify the objects and relationships of tasks and activities.

In FY20+ we hope to leverage our work to:

- Expand interoperability successes
  - More complex models
  - Physics/Simulation models, MCAD/ECAD models
  - Explore ontology discovery, matching, and alignment reasoning
- Demonstrate reasoning rules to assess model credibility
  - Transform the questions into reasoning rules (combining ontology theory, pattern analysis theory, and graph theory)
  - Build a library of reasoning rules - does the model:
    - Follow good practice (Compliance),
    - Fulfill intended use (Accuracy),
    - Describe the real world (Correctness),
    - …
    - Incorporate VVUQ concepts (Testability)



SUMMARY OF MBSE MODEL CREDIBILITY EVALUATION CRITERIA

A	**Compliance (with MBSE good practices and domain standard ontology)**
	CRITERION DESCRIPTION: the model properly and fully conforms to the good practices and established guidelines for implementing MBSE models. There are no errors and/or omissions relative to implementing MBSE guidelines, to include properly applied configuration management of the model. The MBSE model should properly employ the standard ontology for the domain of interest and all external data should be entered into the proper ontology elements.
B	**Accuracy (ability to accurately and effectively fulfill the intended use of the model)**
	CRITERION DESCRIPTION: the model must be able to answer the questions that are put to it; note that these questions should be defined up front and should drive the design and development of the MBSE model. The model must be properly structured to answer the questions and should contain sufficient information to produce the answers.
C	**Correctness (how well the model describes the real world system)**
	CRITERION DESCRIPTION: the model must properly and fully represent the real world system of interest, including the composition of the system, the behaviors of the system, and the critical characteristics for employment of the system. A correct model can be used in lieu of the real system to answer questions of interest, to include questions of the appropriateness of the system design for the real world system's real world mission.
D	**Completeness (maturity of the model in the context of the program developing the model)**
	CRITERION DESCRIPTION: Is the model's maturity sufficient for the current stage of the system lifecycle (including content reviews and configuration management)? Are the contents of the model sufficient to accomplish the intended use of the model and the intended use of the system being modeled?
E	**Testability (ability of the model to participate in testing the design)**
	CRITERION DESCRIPTION: the architecture model should serve as a key element of design for testability and test first/test exploration. One aspect of this is that the model itself should be part of "testing" by providing behavior representations that are executable. Another aspect is that the model shall provide useful guidance to other test first/test exploration of design options (such as identifying test exploration of unknown aspects of the architecture/design)
F	**Reusability (reuse of previous models in the current model + ability to reuse current models elements)**
	CRITERION DESCRIPTION: the model should be built from elements reused from previous models and should provide elements to a library for reuse. The model should display the patterns appropriate to the domain of the system.