A Semantic Reference Model for Capturing System Development and Evaluation

Thursday, 1/27/2022

Abha Moitra	David Archer	Daniel Russell
Paul Cuddihy	Eric Mertens	
Kit Siu	Kevin Quick	
Baoluo Meng	Valentin Robert	
John Interrante		
GE Research	**Galois**	**GE Aviation Systems**

Overview and Outline

- Motivation
- Our approach
- Core ontology
- Summary & conclusions & future work

Motivation

- Certification of software in military airborne systems
  - system architecture, requirements, hazards, testing
  - processes and standards to be followed
  - assemble an assurance case from evidence
- Challenges:
  - complex, diverse data, often embedded in documents
  - time-consuming, tedious, does not scale
  - from executive summary of Defense Innovation Board's Software Acquisition and Practices:
    - "The current approach to software development is broken and is a leading source of risk to DoD: it takes too long, is too expensive, and exposes warfighters to unacceptable risk"

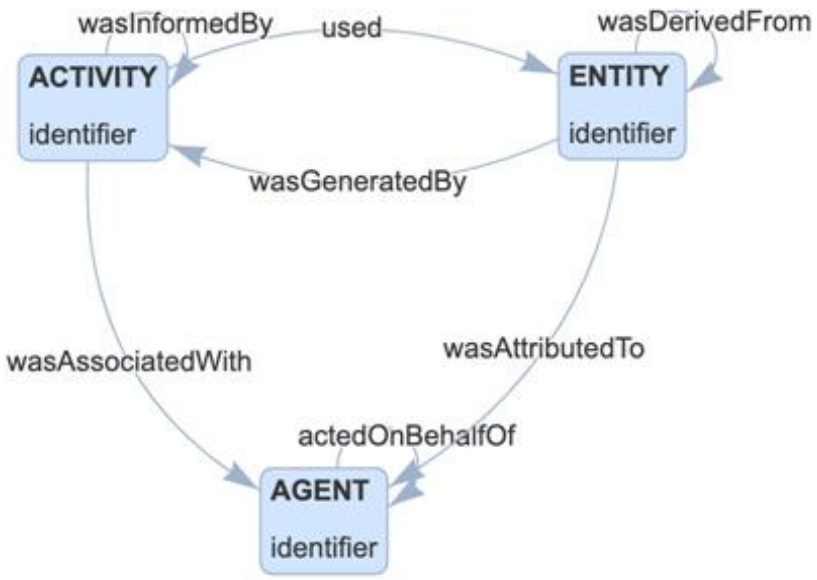
Our Approach

- DARPA's Automated Rapid Certification Of Software (ARCOS)
- Developed Rapid Assurance Curation Kit (RACK)
  - a semantic reference model (core ontology) that can be instantiated for different systems
  - APIs for ingesting, querying and visualizing evidence
  - https://github.com/ge-high-assurance/RACK

Core Ontology: Starting Point

Built on 3 classes from W3C PROV-W3C

- ENTITY: objects for which meta-evidence can be provided
- ACTIVITY: represent execution of defined processes that give rise to entities
- AGENT: represent humans, organizations, or software that cause activities to occur



Core Ontology: Representation Choice

Represented in Semantic Application Design Language (SADL)

- Controlled English that is automatically translated to OWL
- Eclipse IDE with linking, contextual help, etc.
- https://github.com/SemanticApplicationDesignLanguage/sadl

Core Ontology

```
// All classes are subclass of THING and inherit its properties.
THING is a class
    described by identifier with values of type string
    described by title with values of type string
    described by description with values of type string
    described by dataInsertedBy with values of type ACTIVITY.
ENTITY is a type of THING
    described by entityURL with values of type string
    described by wasDerivedFrom with values of type ENTITY
    described by wasRevisionOf with values of type ENTITY
    described by wasImpactedBy with values of type ENTITY
    described by wasGeneratedBy with values of type ACTIVITY
    described by wasAttributedTo with values of type AGENT
    described by generatedAtTime with values of type dateTime
    described by invalidatedAtTime with values of type dateTime.
AGENT is a type of THING
    described by actedOnBehalfOf with values of type AGENT.
ACTIVITY is a type of THING
    described by wasAssociatedWith with values of type AGENT
    described by wasInformedBy with values of type ACTIVITY
    described by startedAtTime with values of type dateTime
    described by endedAtTime with values of type dateTime
    described by goal with values of type ENTITY
    described by used with values of type ENTITY.
```

Airborne Systems

```
// Artifacts needed and produced in development process.
{FILE, FUNCTION, HWCOMPONENT, INTERFACE, SWCOMPONENT, SYSTEM} are types of
ENTITY.

// Development process
{ANALYSIS_ANNOTATION, ANALYSIS_OUTPUT, HAZARD, REQUIREMENT, TEST, TEST_RESULT}
are types of ENTITY.

// Objectives for certification process.
OBJECTIVE (note "An OBJECTIVE identifies tasks from a process for which evidence
must be provided to show that the task has been completed.") is a type of ENTITY.

// Several of the entities introduced above are defined via some activity.
{ANALYSIS, FILE_CREATION, HAZARD_IDENTIFICATION, REQUIREMENT_DEVELOPMENT, REVIEW,
SYSTEM_DEVELOPMENT, TEST_DEVELOPMENT, TEST_EXECUTION} are types of ACTIVITY.
{BUILD, CODE_DEVELOPMENT, CODE_GEN, COMPILE, PACKAGE} are types of FILE_CREATION.

// Activities are performed by agents.
{PERSON, ORGANIZATION, TOOL} are types of AGENT.
```

Illustrative Properties

```
// Properties related to SYSTEM.
partOf describes SYSTEM with values of type SYSTEM.

// Properties for HAZARD and HAZARD_IDENTIFICATION.
source describes HAZARD with values of type ENTITY.
source is a type of wasImpactedBy. // refines provenance ontology property

// Properties for REQUIREMENT and REQUIREMENT_DEVELOPMENT.
governs describes REQUIREMENT with values of type ENTITY.
mitigates describes REQUIREMENT with values of type ENTITY.
satisfies describes REQUIREMENT with values of type ENTITY.
{governs, mitigates, satisfies} are types of wasImpactedBy.
referenced describes REQUIREMENT_DEVELOPMENT with values of type ENTITY.
referenced is a type of used.

// Link objectives (from a standard) to activities.
satisfiedBy describes OBJECTIVE with values of type ACTIVITY.
```

Summary & Conclusion & Future Work

- Presented aspects of core ontology in RACK (for DARPA ARCOS program)
- There are ontology overlays for other performers on DARPA ARCOS program
- There is mechanism to ingest, query and visualize data in a triple store
- Currently in Phase 2 of DARPA ARCOS which is focused on security aspects related to certification
- All details: https://github.com/ge-high-assurance/RACK