# CASE

## Cyber-investigation Analysis Standard Expression

# Workshop

Harm van Beek

CASE Technical Direcotor, Netherlands Forensics Institute

harm.van.beek@nfi.nl

24 April 2019

# Workshop Agenda

- 0930-1000: Introduction, Status Update, & Governance
- 1000-1030: Ontology & Adoption Overview
- 1030-1100: CETIC Demo
- 1100-1115: Coffee Break
- 1115-1200: Mapping & Integration Tutorial
- 1200-1230: Closed CASE Community Discussion

CASE

# Outline

- What is CASE?
- Project Status
- Community Status & Organization
- Works in Progress
- Interested/Involved Organizations
- Membership & Resources

# Cyber-investigation Analysis Standard Expression

CASE is a community-developed ontology to support:

- reporting of digital traces
- exchanging of digital traces
- tool validation (express ground truth)

in the context of:

- digital forensic science
- incident response
- counter-terrorism
- criminal justice
- forensic intelligence
- situational awareness

CASE

# Project Status

**Example Expressions**

- Bulk Extractor Forensic Path (info)
- Call Log
- Device
- Email
- EXIF Data
- Files (info)
- Forensic Lifecycle
- Location
- Message
- Multipart File (info)
- Oresteia (info)
- Raw Data
- Reconstructed File (info)
- SMS and Contacts

**Proof-of-Concepts**

- CETIC
- Plaso/log2timeline
- Volatility

**Reference Documents**

- Representing Mobile Devices and SIM Cards
- Representing File and File System information
- Representing Recoverability of Unallocated Files
- Representing Accounts

**Mappings**

- Autopsy/Sleuthkit
- Bulk Extractor
- Cellebrite
- DC3DD
- NSRL
- Plaso/log2timeline
- Volatility

**Framework Tools**

- RDFDiff
- Python API

# Community Status

2015-03 Initial ideas presented (DI-12-1, 102-110)

2017-07 CASE introduction paper (DI-22, 14-45)

2018-04 workshop → first roadmap

2018-08 community formalization started:

- 2018-11 bylaws
- 2019-01 governance committee elected
- 2019-01 code of conduct
- 2019-02 ontology committee (charter)
- 2019-04 caseontology.org

Biweekly virtual meetings, approx. 1 hour:

- Governance committee
- Ontology committee

# Community Organization

- Focus on running the organization so the community can thrive
- Directors elected for one-year terms
- Charter new committees as necessary

CASE Governance Committee

Presiding Director

- Secretary (appointed)
- Treasurer – Assigned to Director
- Non-voting Directors (appointed)
- UCO Community Representative

Commercial Organization Representative Director

Academia Organization Representative Director

Government Organization Representative Director

Non-Profit Organization Representative Director

Reports to

CASE Ontology Committee

Interact

CASE Mapping and Adoption Committees (coming soon)

CASE Technical Director

Appoints

Chair and Assistant Chair

Appoints

For-Profit Organization Contributor x2+

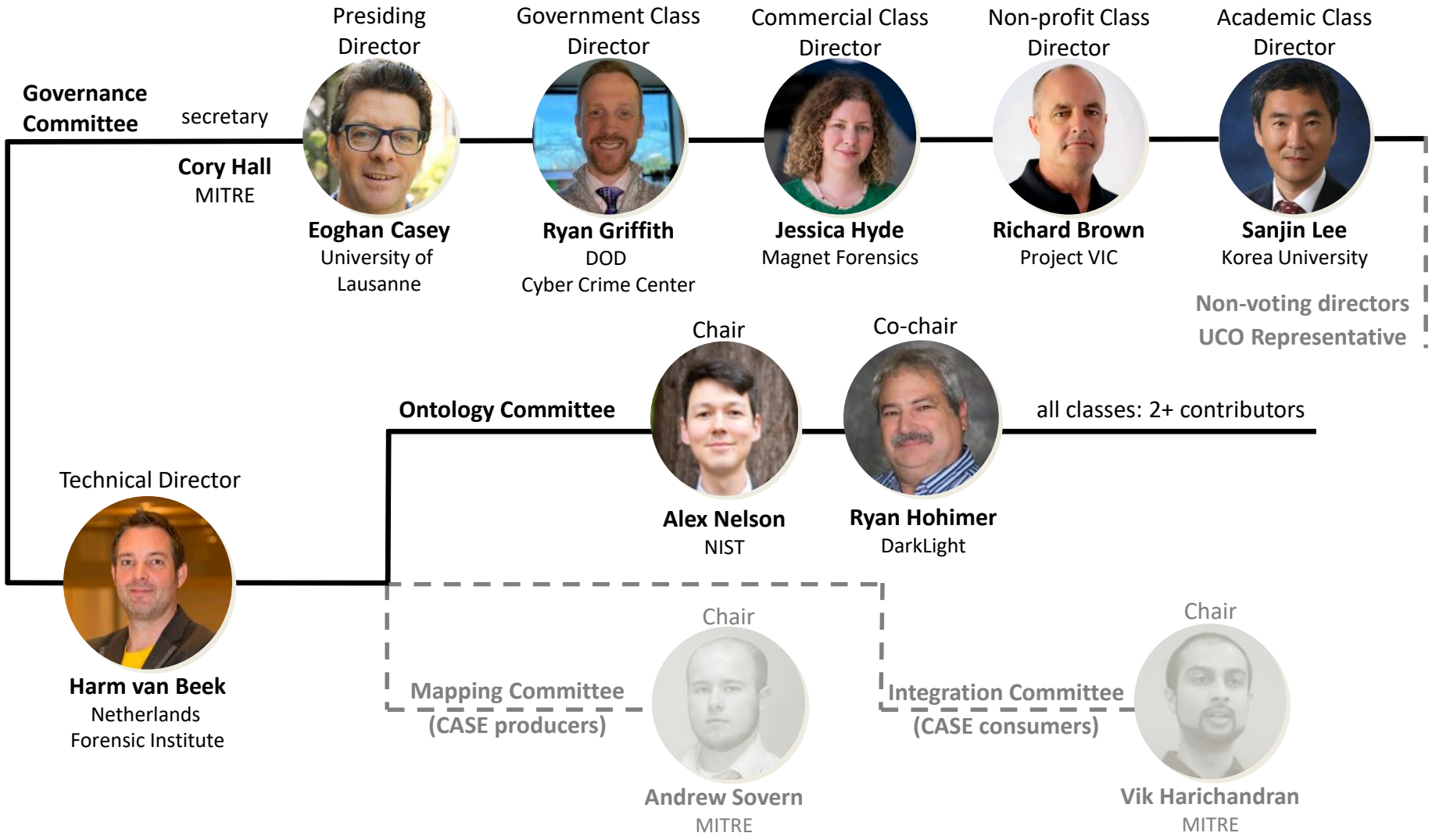
Academia Organization Contributor x2+

Government Organization Contributor x2+

Non-Governmental Organization Contributor x2+

- Sets technical direction for CASE
- Involves technical representatives across stakeholder communities
- No more than two appointees from any employer with voting rights

# Community Organization



# Class Representation is Key to Success

Elected representatives appoint Advisory Committees:

**For-profit**

- Tool Vendor
- Practitioner
- Government Contractor

**Non-profit**

- Separate Non-Profit

**Academia**

- Academic Organizations
- Independent R&D Institutes

**Government**

- International
- National
- Sub-national
- Law Enforcement

CASE

# Works in Progress

**Organization**

Mapping committee (charter)

Integration committee (charter)

Privacy statement

Application form

Operations guidelines

- Trello
- Github
- …

**Ontology**

Roadmap

Documentation

License

- Apache 2

**Workshops**

NIST Ontology Workshop
June 2019

DFRWS USA
July 2019

CASE

# Interested/Involved Organizations





























# Community Membership

## Online application via the CASE Community Website

- Active Members assigned to committee
  - Ontology
  - Mapping (coming soon)
  - Integration (coming soon)
- Observer Member
  - Receive updates from the community
  - Membership for organization leaders and administrative staff
- Organization Member
  - For organizations that want to join the CASE Community (coming soon)
- Membership fee structure is in the works

CASE

# Resources

**Community Website**
`www.caseontology.org`
- organization
  - bylaws
  - code of conduct
  - meeting notes
- documentation
  - roadmap
  - publications
  - use cases
- online membership application

**Organization***
`trello.com/caseworks`
- work in progress
  - draft documentation
- meeting agendas

**CASE Ontology**
`github.com/casework/CASE`
- RDF
  - natural language glossary
  - open issues
- documentation
  - guides

**Development Forum***
`groups.google.com/d/forum/case-dev`

* Requires community membership

# Ontology Overview


## Deborah L. Nichols

**CASE Ontology Committee / CASE Project Team, MITRE**
**DLNichols@mitre.org**



**Approved for public release under PRE 18-4297.**

# Outline

- Purpose: A Standard Ontology for Cyber-investigation
- Use Cases: Capabilities Supported by CASE
- Initial Version of CASE Ontology (CASE/UCO)
- New CASE Ontology Engineering Work in Progress
- CASE Ontology Committee
- How to Get Involved
- CASE Ontology Resources

CASE

MITRE

# Purpose: A Standard Ontology for Cyber-investigation

- **Improve the efficiency and effectiveness of cyber-investigations**
- **Enable trusted, accurate, machine-understandable sharing of cyber-investigation information**
  - Tool-to-tool data exchange
  - Cross-unit and cross-organizational exchange of information
- **Support integrated management of investigative sources and analysis**
- **Long-term objective: Unified analysis and interactions that enable multiple organizations to combine investigations**

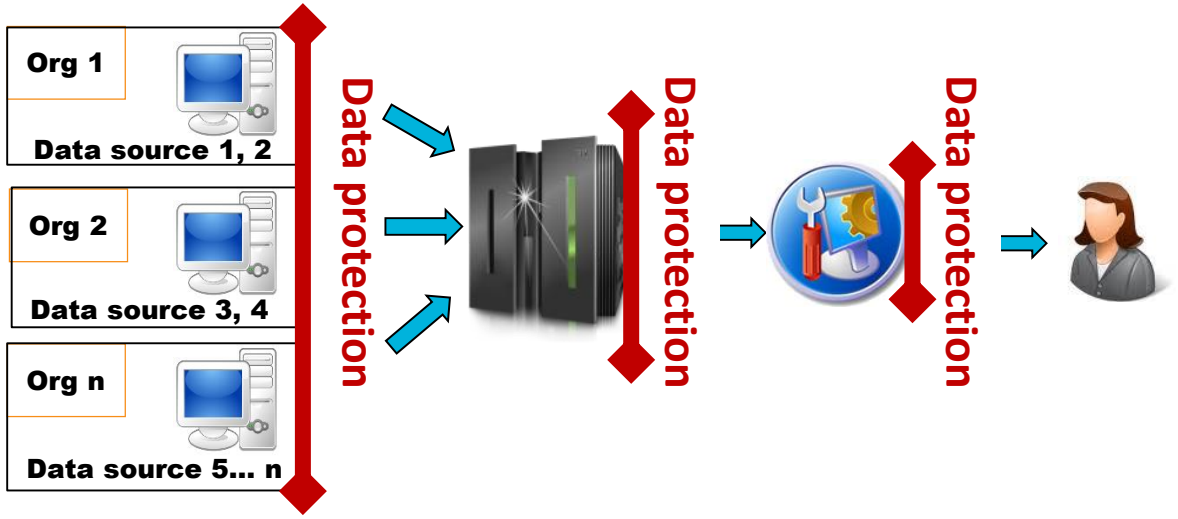
*Cyber-investigation:* *Any investigation (including criminal, civil, corporate, and defense) that has a digital dimension, often involving information from multiple digital data sources, organizations, and jurisdictions.*

CASE

MITRE

# Use Cases: Capabilities Supported by CASE

- Interoperability between systems and tools
- Maintaining provenance at all phases of the cyber-investigation lifecycle
- Enhanced tool testing and validation of results
- Controlled access to data
- Capturing unsupported data structures
- Support for intelligent analysis




CASE
MITRE

# Applicable to Sub-domains of Cyber-investigation

- Digital forensic science
- Incident response
- Counter-terrorism
- Criminal justice
- Forensic intelligence
- Situational awareness

CASE
MITRE

# Overview of Concepts Used in Cyber-investigations

- **People / Roles** (technical, legal, offender, victim, *etc*.)
- **Objects & Relationships** (links, behaviors, *etc.*)
- **Actions** (performed by people, *e.g.*, seizure, running tools, concealing, *etc*.)
- **Legal authorization**
- **Process / Lifecycle of Investigation** (acquisition, analysis, *etc*.)
- **Chain of custody** (who did what, when, and where)
- **Chain of evidence** (maintaining the link from data source to final result)

*Cyber-investigations require concepts of the cybersecurity world (e.g., assets, behaviors, observations) as well as concepts specific to digital investigations.*

From Eoghan Casey, *CASE/UCO Ontology*, Briefing 20181214

CASE

MITRE

# CASE/UCO Prototype (CASE v0.1.0)

- **Described in *Digital Investigations* 22 (2017) by E. Casey *et al*.**
- **Included concepts (classes and properties) from two ontologies**
  - UCO = Unified Cyber Ontology
    - Represents the common objects of the cybersecurity domain
  - CASE = Cyber-investigation Analysis Standard Expression
    - Specifies concepts for representing investigations (*e.g.*, evidence, provenance)
    - Applicable for digital forensics, incident response, terrorism, and criminal justice
    - Satisfies the needs of many use cases (via duck-typing)
- **Single namespace: http://case.example.org/core#**
- **Encoded using Turtle (.ttl) for the ontology specification**
  - Instance implementation in JSON-LD
  - Existing API and mappings conform to the prototype

CASE

MITRE

# New CASE Ontology Engineering Work in Progress

- **Planned work by CASE Community Ontology Committee**
  - New group created Jan. 2019 *(more on this later)*
- **Establishing the official CASE namespace**
- **Separation of CASE and UCO concepts into own namespaces**
- **Processing accumulated change requests for both ontologies**
  - Collaboration between CASE and UCO communities
  - Change requests submitted to CASE are referred to UCO as needed
- **Examining improved support for automated reasoning**
- **CASE v1.0 planned for release in 2020**
- **CASE API and mappings will be migrated to CASE v1.0**

CASE

MITRE

# UCO: Unified Cyber Ontology

- **baseURI: http://unifiedcyberontology.org**
  - Namespace for uco-core: http://unifiedcyberontology.org/core#
- **Domain: Types of entities applicable across all cybersecurity domains**
- **Managed by the UCO Community**
  - Presiding Director: Sean Barnum (FireEye)
  - Technical Director: Ryan Hohimer (DarkLight)
- **Web site (GitHub): https://github.com/ucoProject/UCO**

- **The UCO and CASE Communities overlap in membership and coordinate their ontology-development processes**

CASE

MITRE

# CASE Ontology

- **baseURI: http://caseontology.org**
- **Domain: Concepts and terminology specific to cyber-investigation**
- **Managed by the CASE Community**
  - Presiding Director: Eoghan Casey (University of Lausanne)
  - Technical Director: Harm van Beek (Netherlands Forensic Institute)
  - Ontology Committee Chair: Alex Nelson ((U.S.) National Institute of Standards (NIST))
- **Liaison representatives are appointed between the CASE and UCO communities, including between their ontology committees**
- **Web site: https://caseontology.org**

CASE

MITRE

# CASE Ontology Committee

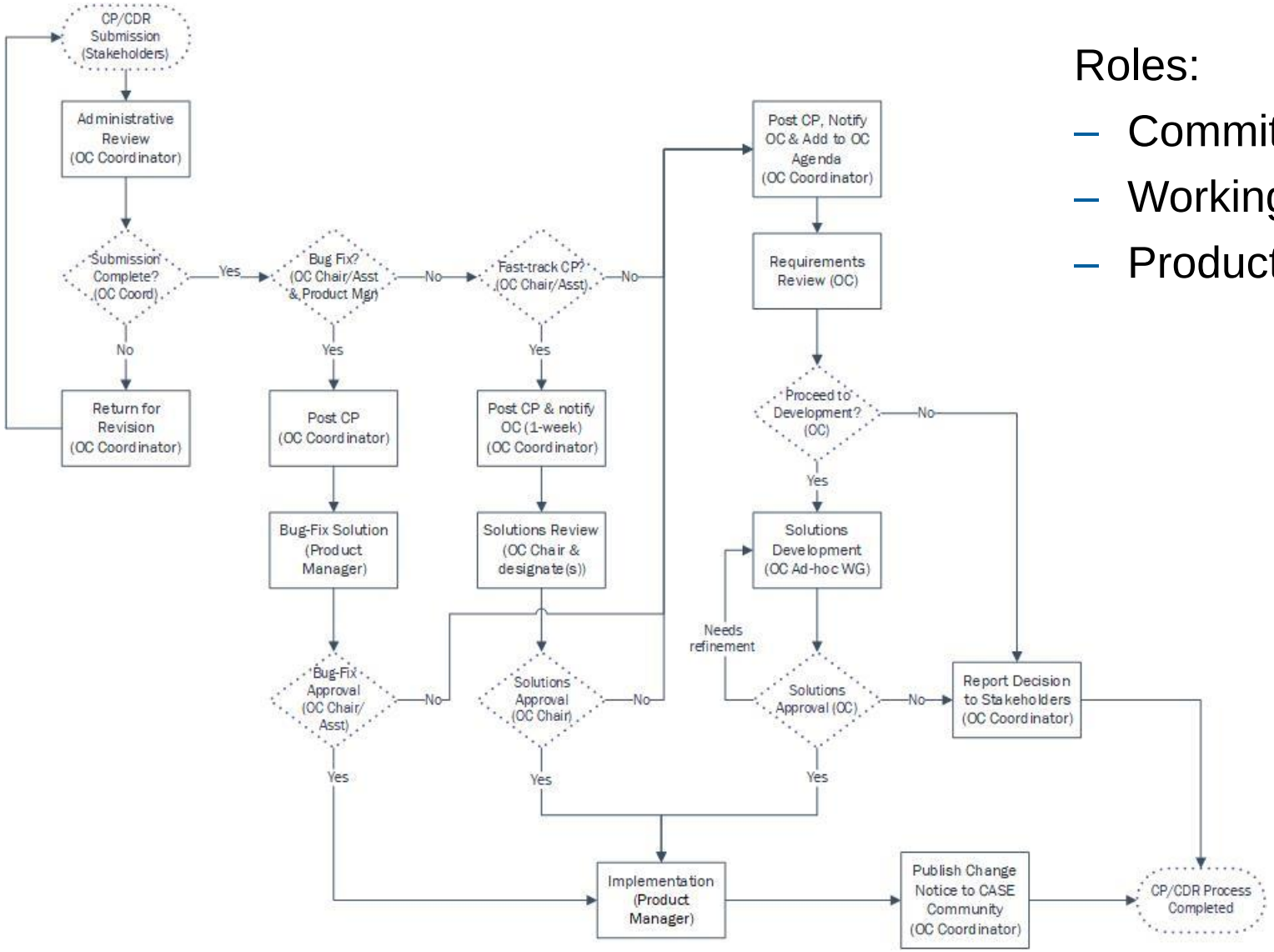
- **Standing Committee responsible for the CASE Ontology, serves as:**
  - *Working group* for management and publication of the CASE Ontology
  - *Coordination body* for CASE Ontology change requests
  - *Advisory group* to the CASE Technical Director
- **Meets regularly (monthly and as-needed)**
  - Conducts CASE requirements reviews and develops technical solutions
  - Coordinates with Unified Cyber Ontology (UCO) Community
- **Members have expertise in ontology and/or data modeling in one or more of the cyber-investigation subdomains**
- **Interacts with CASE Mapping and Integration Committees to support CASE adopters**

CASE

MITRE

# CASE Ontology Development Process

- **Change Proposals / Change Development Requests**
- **Requirements Review**
- **Technical Solutions**
- **Documentation**
- **Version Control**
- **Change Notifications**




Roles:

- Committee Chair
- Working Groups
- Product Manager

CASE

MITRE

# How to Get Involved with the CASE Ontology

- **Who: CASE Ontology Committee**
- **What: Ontology development processes (*e.g.*, change requests, requirements review, and solutions development)**
- **Where: Meeting online and occasionally in-person**
- **When: Monthly Ontology Committee meetings and other activities**
  - June: CASE Workshop (Rockville, Maryland, U.S.)
  - July: DFRWS US 2019 (Portland, Oregon, U.S.)
- **Why: Promote a standard ontology and tools to support interoperability for cyber-investigations**
  - Align your tools and/or domain representations with CASE
  - Coordinate with Integration and Mapping Committees activities
- **How: Apply at [https://caseontology.org/community/membership.htm](https://caseontology.org/community/membership.htm)**

CASE

MITRE

# CASE Ontology Resources

- **CASE Community web site: https://caseontology.org**
  - Community: *Members, Committees, Meetings*
  - Resources: *Bylaws, Publications*
  - Mailing Lists enrolment: https://caseontology.org/contribute.html

- **CASE Ontology GitHub: https://github.com/casework/CASE**

- **CASE Community process web sites (membership required)**
  - Community Trello (workflow)
  - CASE Development Forum (Google groups)

CASE
MITRE

# Implementations Using CASE

- **Implementations**
  - EVIDENCE2eCODEX: https://evidence2e-codex.eu/
  - Autopsy
  - U.S. government tools

- **Upcoming sections of this workshop:**
  - Adoption Overview
  - CETIC Presentation & Demo
  - Mapping & Integration Tutorial

CASE
MITRE

# Questions?

CASE

MITRE

# Adoption Overview


## Vik Harichandran

**CASE Integration Committee Chair / CASE Project Team, MITRE**
**vharichandran@mitre.org**



**Approved for public release under PRE 18-4297.**

# Outline

- **Concepts vs. Representation vs. Instantiation**

- **Project Layers**
  - Meta
  - Top
  - Middle
  - Bottom

CASE

MITRE

# Concepts vs. Representation vs. Instantiation

- **Idea:**
  - "The quick brown fox jumps over the lazy dog."
- **Representation**
  - Sight (our only option when distanced from each other, minus phone calls which is not efficient)
    - Writing
    - Picture/diagram
    - Sign language
- **Instantiation**
  - An actual brown fox jumping over a lazy dog – not just an idea.

CASE

MITRE

# Project Layers (Meta)

- **Knowledge Representation Languages:**
  - Animal Representation Language ***
    - We will allow for animal objects to be represented.
      - Animal objects can have adjective properties (color, speed, energy level).
      - Animal objects can perform actions (jump over).
    - Animal objects are represented via this syntax: <first letter> - <last letter> (fox idea = f-x; dog idea = d-g)
  - OWL2 (Web Ontology Language v2) – *created by a W3C working group*
    - Specification uses functional syntax to represent ontological ideas used to create ontologies.
    - RDF/XML is the required serialization for defining the ontology itself (Turtle is optional).

*** *This is a fake standard.*

CASE

MITRE

# Project Layers (Top)

- **Designs:**
  - Fox-Dog Specification
    - Add an additional restriction, such as our ontology only represents mammals (“eww insects”). In other words, we’re restricting things further than the Animal Representation Language standard.
  - CASE Specification – *this document has yet to be written*
    - Since compliance with the OWL2 standard *requires* RDF/XML be the serialization used, CASE will likely adopt this.
    - Non-OWL2 requirements will also be added:
      - E.g. when does a newly proposed OWL2 class get put into *core* vs. *propbundle*?
      - E.g. all CASE exports must have at least one *Person/Role* object?
      - E.g. how is versioning handled?

CASE

MITRE

# Project Layers (Middle)

- **Ontologies:**
  - Fox-dog Ontology
    - Fox
      - (r) colors: list of str (at least 1)
      - (r) tail color: list of str (at most 2)
      - (o) energy: bool
      - (o) speed: int (km/h)
    - Dog
      - (r) colors: list of str (at least 1)
      - (o) tail color: list of str (at most 2)
      - (o) energy: bool
      - (o) speed: int (km/h)
  - CASE Ontology – *RDF graph* ***
    - Action
      - (r) startTime: timestamp
      - (r) endTime: timestamp
      - (o) subactionRefs: list of Action (any number)
    - ActionLifecycle (a kind of Action)
      - (r) startTime: timestamp
      - (r) endTime: timestamp
    - Identity
      - (r) name: str (only 1)

*** *Resource Description Framework (RDF) does not restrict what can be linked – hence ontologies!*

CASE

MITRE

# Project Layers (Bottom)

- **Mappings:**
  - Fox-Dog Ontology < - > Genus Ontology < - > Pokémon Data Model
    - Fox < - > Vulpes < - > Vulpix
    - Dog < - > Canis < - > Arcanine

- **Integrations:**
  - Only occurs with software. Pokémon aren't software so we can't integrate. Digimon are though!
    - Fox < - > Renamon
    - Dog < - > Dobermon

CASE
MITRE

# Questions?

CASE

MITRE

# CETIC Demo

CASE

MITRE

# Coffee Break
# (15 minutes)

CASE

MITRE

# Mapping & Integration Tutorial


## Vik Harichandran

**CASE Integration Committee Chair / CASE Project Team, MITRE**
**vharichandran@mitre.org**



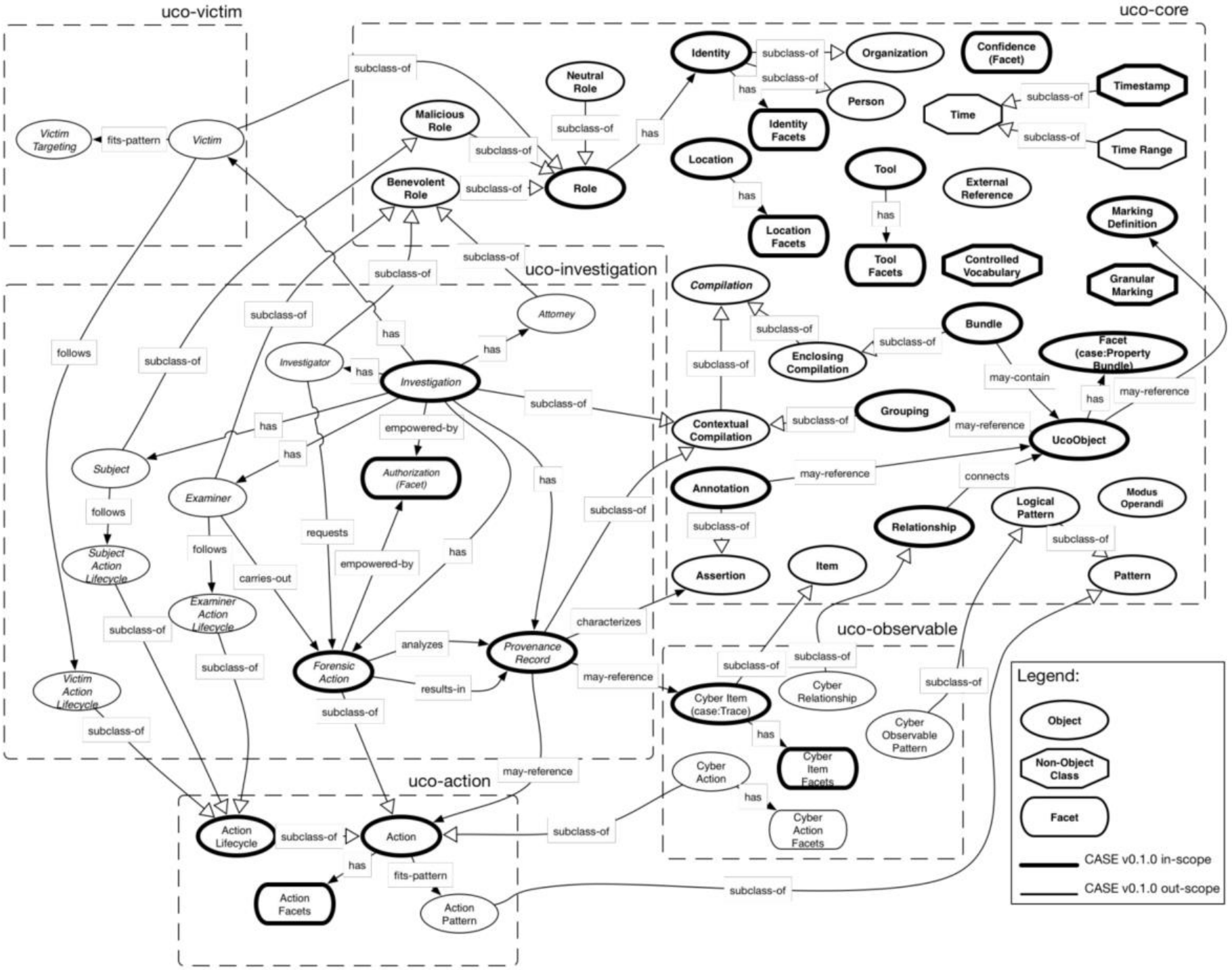
**Approved for public release under PRE 18-4297.**

# Outline

- FY18 Work
- Mapping
- Supporting Tools
- Python API

CASE

MITRE


uco-victim
uco-core
uco-investigation
uco-observable
uco-action
Victim Targeting
fits-pattern
Victim
Malicious Role
Neutral Role
Benevolent Role
Role
Identity
Organization
Person
Identity Facets
Confidence (Facet)
Time
Timestamp
Time Range
Location
Location Facets
Tool
Tool Facets
External Reference
Marking Definition
Granular Marking
Controlled Vocabulary
Compilation
Enclosing Compilation
Contextual Compilation
Bundle
Facet (case:Property Bundle)
Grouping
UcoObject
Annotation
Assertion
Item
Relationship
Logical Pattern
Modus Operandi
Pattern
Attorney
Investigator
Investigation
Subject
Examiner
Authorization (Facet)
Subject Action Lifecycle
Examiner Action Lifecycle
Victim Action Lifecycle
Forensic Action
Provenance Record
Cyber Item (case:Trace)
Cyber Relationship
Cyber Observable Pattern
Cyber Item Facets
Cyber Action
Cyber Action Facets
Action Lifecycle
Action
Action Facets
Action Pattern
subclass-of
has
follows
empowered-by
requests
carries-out
analyzes
results-in
characterizes
may-reference
may-contain
connects
Legend:
Object
Non-Object Class
Facet
CASE v0.1.0 in-scope
CASE v0.1.0 out-scope

CASE
MITRE

# * Excerpt from CASE_Categories_v01.0.xlsx

| LEGEND: |
|---|
| Orange = Python class (category of CASE types); all verification function names begin with the prefix for the class they fall under. |
| Green = One-to-one relationship to NLG type and therefore a possible gap, outdated, or unnecessary. |
| Blue = A 'sub_' class is a level down from (derivative of) its parent. Example: The NLG type 'ForensicAction' (function name core_sub_ForensicAction) is a type of 'Action' (function name core_Action). |

| Prefix/Class | Object | Parent(s) | Child(ren) |
|---|---|---|---|
| core_ | Action | | ForensicAction, ActionLifecycle |
| class CoreObject | Assertion | | Annotation |
| | Bundle | EnclosingCompilation | |
| | ControlledVocabulary | | |
| | Identity | | |
| | Location | | |
| | MarkingDefinition | | |
| | Relationship | | |
| | Role | | |
| | Tool | NeutralRole | |
| | Trace | Item/Observable | |
| propbundle_ | Account | | |
| class PropertyBundle | AccountAuthentication | | |
| | ActionReferences | | |
| | Application | | |
| | ApplicationAccount | | |
| | ArchiveFile | | |
| | Attachment | | |
| | Audio | | |
| | Authorization | | |
| | AutonomousSystem | | |
| | BrowserBookmark | | |
| | BrowserCookie | | |
| | Build | | |
| | Calendar | | |
| | CalendarEntry | | |
| | CompressedStream | | |
| | ComputerSpecification | | |
| | Confidence | | |
| | Contact | | |

| | | | |
|---|---|---|---|
| context_ | Grouping | ContextualCompilation | |
| class Context | Investigation | ContextualCompilation | |
| | ProvenanceRecord | ContextualCompilation | |
| duck_ | AlternateDataStream | | |
| class Duck | ArrayOfHash | | |
| | ArrayOfObject | | ArrayOfAction |
| | ArrayOfString | | |
| | BuildConfigurationType | | |
| | BuildInformationType | | |
| | BuildUtilityType | | |
| | CompilerType | | |
| | ConfigurationSettingType | | |
| | ControlledDictionary | | |
| | ControlledDictionaryEntry | | |
| | DataRange | | |
| | DependencyType | | |
| | Dictionary | | |
| | DictionaryEntry | | |
| | GlobalFlagType | | |
| | GranularMarking | | |
| | Hash | | |
| | IComHandlerActionType | | |
| | LibraryType | | |
| | MarkingModel | | |
| | MimePartType | | |
| | TaskActionType | | |
| | TriggerType | | |
| | WhoIsContactType | | |
| | WhoIsRegistrarInfoType | | |
| | WindowsPEFileHeader | | |

| | | |
|---|---|---|
| Identity (propbundle) | | Address, Affiliation, BirthInformation, CountriesOfResidence, Events, Identifier, Languages, Nationality, Occupation, OrganizationDetails, PersonalDetails, PhysicalInfo, Qualifiacation, Relationship, |

# CASE Categories

**Action** — CORE

Model-Generated Definition:

A kind of UcoObject. A valid occurrence may have the following properties:

- *actionStatus* at most one occurrence of ControlledVocabulary.
- *startTime* at most one value of *Timestamp*.
- *endTime* at most one value of *Timestamp*.
- *error* any number of values of any type.
- *actionCount* at most one value of *NonNegativeInteger*.
- *subactionRefs* any number of occurrences of Action.

Definition: *Something that may be done or performed.*

**ActionLifecycle** — SUB_CORE

Model-Generated Definition:

A kind of Action. A valid occurrence satisfies the following necessary condition:

- *phaseRefs* exactly one occurrence of ArrayOfAction.

A valid occurrence may also have the following properties:

- *actionStatus* exactly zero occurrences of ControlledVocabulary.
- *startTime* exactly zero values of *Timestamp*.
- *endTime* exactly zero values of *Timestamp*.
- *(Unnamed Class)* exactly zero values of any type.
- *actionCount* exactly zero values of *NonNegativeInteger*.

Definition: *An action pattern consisting of an ordered set of multiple actions or sub action-lifecycles.*

**ArrayOfAction** — SUB_DUCK

Model-Generated Definition:

A kind of ArrayOfObject.

Definition: *An ordered list of action object references.*

**ArrayOfObject** — DUCK

Model-Generated Definition:

A valid occurrence satisfies the following necessary condition:

- *object* at least one occurrence of UcoObject.

Definition: *An ordered list of object references.*

* The categories were derived from NLG v0.1.0

CASE

MITRE

# Mapping Workflow

- **Review CASE ontology & associated resources**
  - Ontology visualization tools
  - CASE implementation examples
  - Community developed resources
- **Export tool variables**
  - Examine tool report output & group into digestible chunks
- **Compare with existing examples**
  - Leverage existing similar mappings
- **Develop custom mappings**
  - Utilize community resources & tools
  - Ask questions

CASE

MITRE

# Ontology Exploration Tools Demo (Ontospy & Protégé)

CASE

MITRE

# RDFDiff

- **Verifies glossary terms by diff-ing two (custom) ontologies:**
  - For comparing custom ontologies to the public CASE Natural Language Glossary (NLG).

- **Why this is useful:**
  - Before mapping use this to identify high-level gaps and coverage.
  - Ontologies encompass a broad spectrum of data. You're focused on a subset of said data. As a developer, You should not have to learn an entire Ontology to implement your focus area.

- **Input can be ttl, n3, xml, and JSON-LD formats.**

CASE

MITRE

# Plaso Mapping Example – Android Calls

- **Relevant info stored in:**
  - Event_data
  - Event
- **call_type**
  - incoming/outgoing/missed

```
PhoneCall
    from
    participant
    to
    callType
    createdTime
    duration
    endTime
    startTime

PhoneAccount
    phoneNumber
```

```
Contact
    contactIdentifier
    contactName
    contactType
    firstName
    lastName
    middleName
    phoneNumber
```

```
call_type = self._GetRowValue(query_hash, row, 'type')
call_type = self.CALL_TYPE.get(call_type, 'UNKNOWN')
duration = self._GetRowValue(query_hash, row, 'duration')
timestamp = self._GetRowValue(query_hash, row, 'date')

event_data = AndroidCallEventData()
event_data.call_type = call_type
event_data.duration = self._GetRowValue(query_hash, row, 'duration')
event_data.name = self._GetRowValue(query_hash, row, 'name')
event_data.number = self._GetRowValue(query_hash, row, 'number')
event_data.offset = self._GetRowValue(query_hash, row, 'id')
event_data.query = query

date_time = dfdatetime_java_time.JavaTime(timestamp=timestamp)
event = time_events.DateTimeValuesEvent(date_time, 'Call Started')
parser_mediator.ProduceEventWithEventData(event, event_data)

date_time = dfdatetime_java_time.JavaTime(timestamp=timestamp)
event = time_events.DateTimeValuesEvent(date_time, 'Call Ended')
parser_mediator.ProduceEventWithEventData(event, event_data)
```

# Namespaces

- **To satisfy diverse use cases three different types of namespaces will exist:**
  - Private custom/proprietary (lowest priority)
  - Public community/in-review
  - Public official CASE (highest priority)

- **The highest priority possible should be used!**

- **Mappings from FY18:**
  - Autopsy/Sleuthkit
  - BulkExtractor
  - Cellebrite
  - DC3DD
  - Plaso
  - NSRL
  - Volatility

CASE

MITRE

# Python API (*case.py*)

- **CoreObject = core NLG types**
  **(derived from UCO; only class that can encapsulate a PropertyBundle)**
- **DuckObject = duck-typed**
  **(type not derived from the above classes; bundles define object)**
- **SubObject = a derivative of a type that fits into one of the above top-level categories**

```
class CoreObject(Node):

    RDF_TYPE = CASE.CoreObject

    def __init__(self, graph, rdf_type=None, **kwargs):
        self.type = rdf_type
        super(CoreObject, self).__init__(graph, rdf_type=rdf_type, **kwargs)
        self.add('CoreObjectCreationTime', datetime.datetime.utcnow())
        self.pb = ""

    def create_PropertyBundle(self, prop_type=None, **kwargs):
        self.pb = PropertyBundle(self._graph, rdf_type=prop_type, **kwargs)
        self.add(CASE.propertyBundle, self.pb)

        return self.pb

class PropertyBundle(Node):

    RDF_TYPE = CASE.PropertyBundle

    def __init__(self, graph, rdf_type=None, **kwargs):
        self.type = rdf_type
        self.propObj = kwargs
        super(PropertyBundle, self).__init__(
                graph, bnode=True, rdf_type=rdf_type, **kwargs)
```

CASE
MITRE

# Duck-typing

```
class DuckObject(Node):

    RDF_TYPE = CASE.DuckObject

    def __init__(self, graph, rdf_type=None, **kwargs):
        self.type = rdf_type
        super(DuckObject, self).__init__(graph, rdf_type=rdf_type, **kwargs)
        self.add('DuckObjectCreationTime', datetime.datetime.utcnow())


class SubObject(Node):

    RDF_TYPE = CASE.SubObject

    def __init__(self, graph, rdf_type=None, **kwargs):
        self.type = rdf_type
        super(SubObject, self).__init__(graph, rdf_type=rdf_type, **kwargs)
        self.add('SubObjectCreationTime', datetime.datetime.utcnow())
```

Type gets stored here. This is a hard-coded string from *NLG.py*.

CASE

MITRE

# Python API (*NLG.py*)

- **What it verifies (while returning RDF nodes for the user):**
  - Parent-child relationships according to ontology.
  - Required vs. optional parameters ("exactly" or "at least one of" = required; "at most" or "any number of" = optional).
  - Types (format/native type of values in fields).

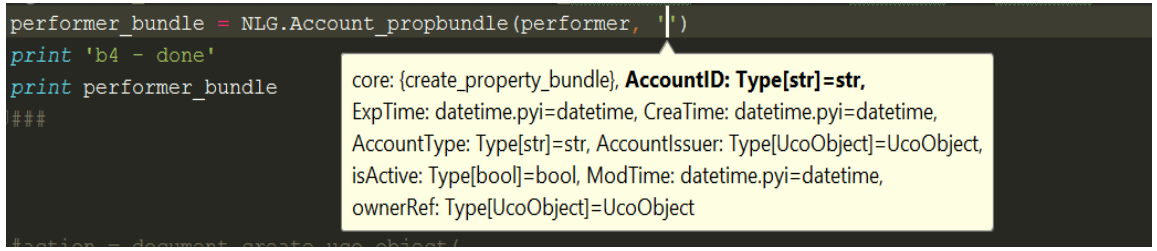
```
context_example          = context_Grouping(doc,
    context_strings      = ['the','teh','hte','het','eth','eht'])
print "Obj4: ", context_example

#=====================================================

core_example_3           = core_Action(doc,
    start_time           = datetime.datetime.utcnow())
print "Obj5: ", core_example_3

sub_example_1            = core_sub_ForensicAction(doc, core_example_3)
print "Obj6: ", sub_example_1

#=====================================================

propbundle_example_2     = propbundle_Identity(core_example_3)
print "Obj7: ", propbundle_example_2

sub_example_2            = propbundle_sub_SimpleName(doc, propbundle_example_2,
                                                     honorific_prefix = 'Mr.',
                                                     given_name       = 'Bond')
print "Obj8: ", sub_example_2

#=====================================================

duck_example             = duck_MarkingModel(doc)
print "Obj9: ", duck_example
```

CASE

MITRE

# Volatility POC

- **Performing runtime type checking ensures output is ontology-compliant.**




IntelliSense Auto Completion

```
Traceback (most recent call last):
  File "sandbox.py", line 11, in <module>
    nlgObj = propbundle_HTTPConnection(uco, http_message_body_data_ref=cObj)
  File "C:\Users\jestroud\PycharmProjects\CASE-API\parameter_approach\NLG.py", line 1807, in propbundle_HTTPConnection
    "[propbundle_HTTPConnection] request_method is required."
AssertionError: [propbundle_HTTPConnection] request_method is required.
```

CASE

MITRE

# Example JSON-LD Output

```
vol.py --plugins='volplugs/src/' -f volatility/memory_images/xp.img caseprocess
```


```
{
  "@id": "_:ac3b9bcc-9709-4bea-bb3b-4b9095256b08",
  "@type": "Process",
  "CreateTime": "2005-07-04 18:17:31 UTC+0000",
  "ProcessName": "svchost.exe",
  "ProcoessID": "680",
  "instrument": {
    "@id": "/usr/local/bin/vol.py --plugins=volplugs/src/ -f volatility/memory_images/xp.img caseprocess"
  },
  "performer": {
    "@id": "test"
  }
},
{
  "@id": "_:c4d768fd-df26-4f67-aa0d-ee60288e24e7",
  "@type": "Process",
  "CreateTime": "2005-07-04 18:20:58 UTC+0000",
  "ProcessName": "cmd.exe",
  "ProcoessID": "3256",
  "instrument": {
    "@id": "/usr/local/bin/vol.py --plugins=volplugs/src/ -f volatility/memory_images/xp.img caseprocess"
  },
  "performer": {
    "@id": "test"
  }
```


CASE

MITRE

# Python API Demo

CASE

MITRE

# Questions?

CASE

MITRE

# CASE Community Social Hour (Place & Time TBD)

CASE

MITRE

# Closed CASE Community Discussion


| | |
|---|---|
| Harm van Beek | Technical Director |
| Deborah L. Nichols | Ontology Committee Member |
| Vik Harichandran | Integration Committee Chair |


# Agenda

- **Questions from the previous sections?**
- **Add miscellaneous items via such questions to a list to get attention at a later date.**
- **Discuss top priority action items.**

CASE

MITRE

# Resources

- **Community Website: www.caseontology.org**
- **Github: www.github.com/casework**
- **Personal Consultation: cyberinvestigationexpress@gmail.com**
- **Email us:**
  - Harm van Beek harm.van.beek@nfi.nl
  - Deborah L. Nichols DLNichols@mitre.org
  - Vik Harichandran vharichandran@mitre.org

CASE

MITRE

MITRE

MITRE is a not-for-profit organization whose sole focus is to operate federally funded research and development centers, or FFRDCs. Independent and objective, we take on some of our nation's—and the world's—most critical challenges and provide innovative, practical solutions.

Learn and share more about MITRE, FFRDCs, and our unique value at www.mitre.org

CASE

MITRE