CybOX™ Version 2.1.1 Part 6: UML Model

Working Draft 01

15 December 2015

Technical Committee:

[OASIS Cyber Threat Intelligence (CTI) TC](https://www.oasis-open.org/committees/cti/)

Chair:

Richard Struse ([Richard.Struse@HQ.DHS.GOV](mailto:Richard.Struse@HQ.DHS.GOV)), [DHS Office of Cybersecurity and Communications (CS&C)](http://www.dhs.gov/office-cybersecurity-and-communications)

Editors:

Desiree Beck ([dbeck@mitre.org](mailto:ikirillov@mitre.org)), [MITRE Corporation](http://www.mitre.org/)

Trey Darley ([trey@soltra.com](mailto:trey@soltra.com)), [Soltra](http://www.soltra.com/)

Ivan Kirillov ([ikirillov@mitre.org](mailto:ikirillov@mitre.org)), [MITRE Corporation](http://www.mitre.org/)

Rich Piazza ([rpiazza@mitre.org](mailto:ikirillov@mitre.org)), [MITRE Corporation](http://www.mitre.org/)

Additional artifacts:

This prose specification is one component of a Work Product which consists of:

* CybOXTM Version 2.1.1 Part 1: Overview. [URI]
* CybOXTM Version 2.1.1 Part 2: Common. [URI]
* CybOXTM Version 2.1.1 Part 3: Core. [URI]
* CybOXTM Version 2.1.1 Part 4: Default Extensions. [URI]
* CybOXTM Version 2.1.1 Part 5: Vocabularies. [URI]
* CybOXTM Version 2.1.1 Part 6: UML Model. (this document)
* CybOXTM Version 2.1.1 Part 7: API Object. [URI]
* CybOXTM Version 2.1.1 Part 8: ARP Cache Object. [URI]
* CybOXTM Version 2.1.1 Part 9: AS Object. [URI]
* CybOXTM Version 2.1.1 Part 10: Account Object. [URI]
* CybOXTM Version 2.1.1 Part 11: Address Object. [URI]
* CybOXTM Version 2.1.1 Part 12: Archive File Object. [URI]
* CybOXTM Version 2.1.1 Part 13: Artifact Object. [URI]
* CybOXTM Version 2.1.1 Part 14: Code Object. [URI]
* CybOXTM Version 2.1.1 Part 15: Custom Object. [URI]
* CybOXTM Version 2.1.1 Part 16: DNS Cache Object. [URI]
* CybOXTM Version 2.1.1 Part 17: DNS Query Object. [URI]
* CybOXTM Version 2.1.1 Part 18: DNS Record Object. [URI]
* CybOXTM Version 2.1.1 Part 19: Device Object. [URI]
* CybOXTM Version 2.1.1 Part 20: Disk Object. [URI]
* CybOXTM Version 2.1.1 Part 21: Disk Partition Object. [URI]
* CybOXTM Version 2.1.1 Part 22: Domain Name Object. [URI]
* CybOXTM Version 2.1.1 Part 23: Email Message Object. [URI]
* CybOXTM Version 2.1.1 Part 24: File Object. [URI]
* CybOXTM Version 2.1.1 Part 25: GUI Dialogbox Object. [URI]
* CybOXTM Version 2.1.1 Part 26: GUI Object. [URI]
* CybOXTM Version 2.1.1 Part 27: GUI Window Object. [URI]
* CybOXTM Version 2.1.1 Part 28: HTTP Session Object. [URI]
* CybOXTM Version 2.1.1 Part 29: Hostname Object. [URI]
* CybOXTM Version 2.1.1 Part 30: Image File Object. [URI]
* CybOXTM Version 2.1.1 Part 31: Library File Object. [URI]
* CybOXTM Version 2.1.1 Part 32: Link Object. [URI]
* CybOXTM Version 2.1.1 Part 33: Linux Package Object. [URI]
* CybOXTM Version 2.1.1 Part 34: Memory Object. [URI]
* CybOXTM Version 2.1.1 Part 35: Mutex Object. [URI]
* CybOXTM Version 2.1.1 Part 36: Network Connection Object. [URI]
* CybOXTM Version 2.1.1 Part 37: Network Flow Object. [URI]
* CybOXTM Version 2.1.1 Part 38: Network Packet Object. [URI]
* CybOXTM Version 2.1.1 Part 39: Network Route Entry Object. [URI]
* CybOXTM Version 2.1.1 Part 40: Network Route Object. [URI]
* CybOXTM Version 2.1.1 Part 41: Network Socket Object. [URI]
* CybOXTM Version 2.1.1 Part 42: Network Subnet Object. [URI]
* CybOXTM Version 2.1.1 Part 43: PDF File Object. [URI]
* CybOXTM Version 2.1.1 Part 44: Pipe Object. [URI]
* CybOXTM Version 2.1.1 Part 45: Port Object. [URI]
* CybOXTM Version 2.1.1 Part 46: Process Object. [URI]
* CybOXTM Version 2.1.1 Part 47: Product Object. [URI]
* CybOXTM Version 2.1.1 Part 48: SMS Message Object. [URI]
* CybOXTM Version 2.1.1 Part 49: Semaphore Object. [URI]
* CybOXTM Version 2.1.1 Part 50: Socket Address Object. [URI]
* CybOXTM Version 2.1.1 Part 51: System Object. [URI]
* CybOXTM Version 2.1.1 Part 52: URI Object. [URI]
* CybOXTM Version 2.1.1 Part 53: URL History Object. [URI]
* CybOXTM Version 2.1.1 Part 54: Unix File Object. [URI]
* CybOXTM Version 2.1.1 Part 55: Unix Network Route Entry Object. [URI]
* CybOXTM Version 2.1.1 Part 56: Unix Pipe Object. [URI]
* CybOXTM Version 2.1.1 Part 57: Unix Process Object. [URI]
* CybOXTM Version 2.1.1 Part 58: Unix User Account Object. [URI]
* CybOXTM Version 2.1.1 Part 59: Unix Volume Object. [URI]
* CybOXTM Version 2.1.1 Part 60: User Account Object. [URI]
* CybOXTM Version 2.1.1 Part 61: User Session Object. [URI]
* CybOXTM Version 2.1.1 Part 62: Volume Object. [URI]
* CybOXTM Version 2.1.1 Part 63: Whois Object. [URI]
* CybOXTM Version 2.1.1 Part 64: Win Computer Account Object. [URI]
* CybOXTM Version 2.1.1 Part 65: Win Critical Section Object. [URI]
* CybOXTM Version 2.1.1 Part 66: Win Driver Object. [URI]
* CybOXTM Version 2.1.1 Part 67: Win Event Log Object. [URI]
* CybOXTM Version 2.1.1 Part 68: Win Event Object. [URI]
* CybOXTM Version 2.1.1 Part 69: Win Executable File Object. [URI]
* CybOXTM Version 2.1.1 Part 70: Win File Object. [URI]
* CybOXTM Version 2.1.1 Part 71: Win Filemapping Object. [URI]
* CybOXTM Version 2.1.1 Part 72: Win Handle Object. [URI]
* CybOXTM Version 2.1.1 Part 73: Win Hook Object. [URI]
* CybOXTM Version 2.1.1 Part 74: Win Kernel Hook Object. [URI]
* CybOXTM Version 2.1.1 Part 75: Win Kernel Object. [URI]
* CybOXTM Version 2.1.1 Part 76: Win Mailslot Object. [URI]
* CybOXTM Version 2.1.1 Part 77: Win Memory Page Region Object. [URI]
* CybOXTM Version 2.1.1 Part 78: Win Mutex Object. [URI]
* CybOXTM Version 2.1.1 Part 79: Win Network Route Entry Object. [URI]
* CybOXTM Version 2.1.1 Part 80: Win Network Share Object. [URI]
* CybOXTM Version 2.1.1 Part 81: Win Pipe Object. [URI]
* CybOXTM Version 2.1.1 Part 82: Win Prefetch Object. [URI]
* CybOXTM Version 2.1.1 Part 83: Win Process Object. [URI]
* CybOXTM Version 2.1.1 Part 84: Win Registry Key Object. [URI]
* CybOXTM Version 2.1.1 Part 85: Win Semaphore Object. [URI]
* CybOXTM Version 2.1.1 Part 86: Win Service Object. [URI]
* CybOXTM Version 2.1.1 Part 87: Win System Object. [URI]
* CybOXTM Version 2.1.1 Part 88: Win System Restore Object. [URI]
* CybOXTM Version 2.1.1 Part 89: Win Task Object. [URI]
* CybOXTM Version 2.1.1 Part 90: Win Thread Object. [URI]
* CybOXTM Version 2.1.1 Part 91: Win User Account Object. [URI]
* CybOXTM Version 2.1.1 Part 92: Win Volume Object. [URI]
* CybOXTM Version 2.1.1 Part 93: Win Waitable Timer Object. [URI]
* CybOXTM Version 2.1.1 Part 94: X509 Certificate Object. [URI]

Related work:

This specification is related to:

* *STIXTM Version 1.2.1 (placeholder)*

Abstract:

The Cyber Observable Expression (CybOX™) is a standardized language for encoding and communicating high-fidelity information about cyber observables, whether dynamic events or stateful measures that are observable in the operational cyber domain. By specifying a common structured schematic mechanism for these cyber observables, the intent is to enable the potential for detailed automatable sharing, mapping, detection and analysis heuristics. This document describes the use of UML to create a data model for CybOX.Status:

This [Working Draft](https://www.oasis-open.org/policies-guidelines/tc-process#dWorkingDraft) (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or [approved](https://www.oasis-open.org/policies-guidelines/tc-process#committeeDraft) as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document [Approval Process](https://www.oasis-open.org/policies-guidelines/tc-process#standApprovProcess) begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

URI patterns:

Initial publication URI:  
http://docs.oasis-open.org/cti/stix/v1.2.1/csd01/part9-coa/stix-v1.2.1-csd01-part9-coa.docx

Permanent “Latest version” URI:  
http://docs.oasis-open.org/cti/stix/v1.2.1/stix-v1.2.1-part9-coa.docx

(Managed by OASIS TC Administration; please don’t modify.)

Copyright © OASIS Open 2015. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](https://www.oasis-open.org/policies-guidelines/ipr) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Portions copyright © United States Government 2012-2015. All Rights Reserved.  
  
STIX™, TAXII™, AND CybOX™ (STANDARD OR STANDARDS) AND THEIR COMPONENT PARTS ARE PROVIDED “AS IS” WITHOUT ANY WARRANTY OF ANY KIND, EITHER EXPRESSED, IMPLIED, OR STATUTORY, INCLUDING, BUT NOT LIMITED TO, ANY WARRANTY THAT THESE STANDARDS OR ANY OF THEIR COMPONENT PARTS WILL CONFORM TO SPECIFICATIONS, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR FREEDOM FROM INFRINGEMENT, ANY WARRANTY THAT THE STANDARDS OR THEIR COMPONENT PARTS WILL BE ERROR FREE, OR ANY WARRANTY THAT THE DOCUMENTATION, IF PROVIDED, WILL CONFORM TO THE STANDARDS OR THEIR COMPONENT PARTS. IN NO EVENT SHALL THE UNITED STATES GOVERNMENT OR ITS CONTRACTORS OR SUBCONTRACTORS BE LIABLE FOR ANY DAMAGES, INCLUDING, BUT NOT LIMITED TO, DIRECT, INDIRECT, SPECIAL OR CONSEQUENTIAL DAMAGES, ARISING OUT OF, RESULTING FROM, OR IN ANY WAY CONNECTED WITH THESE STANDARDS OR THEIR COMPONENT PARTS OR ANY PROVIDED DOCUMENTATION, WHETHER OR NOT BASED UPON WARRANTY, CONTRACT, TORT, OR OTHERWISE, WHETHER OR NOT INJURY WAS SUSTAINED BY PERSONS OR PROPERTY OR OTHERWISE, AND WHETHER OR NOT LOSS WAS SUSTAINED FROM, OR AROSE OUT OF THE RESULTS OF, OR USE OF, THE STANDARDS, THEIR COMPONENT PARTS, AND ANY PROVIDED DOCUMENTATION. THE UNITED STATES GOVERNMENT DISCLAIMS ALL WARRANTIES AND LIABILITIES REGARDING THE STANDARDS OR THEIR COMPONENT PARTS ATTRIBUTABLE TO ANY THIRD PARTY, IF PRESENT IN THE STANDARDS OR THEIR COMPONENT PARTS AND DISTRIBUTES IT OR THEM “AS IS.”

Table of Contents

[1 Introduction 6](#_Toc440957173)

[1.1 CybOX™ Specification Documents 6](#_Toc440957174)

[1.2 Document Conventions 6](#_Toc440957175)

[1.2.1 Fonts 6](#_Toc440957176)

[1.3 Terminology 7](#_Toc440957177)

[1.4 Normative References 7](#_Toc440957178)

[1.5 Non-Normative References 7](#_Toc440957179)

[2 UML Model Artifact 8](#_Toc440957180)

[3 Data Model Conventions 9](#_Toc440957181)

[3.1 UML Packages 9](#_Toc440957182)

[3.2 Naming Conventions 10](#_Toc440957183)

[3.3 UML Diagrams 10](#_Toc440957184)

[3.3.1 Class Properties 11](#_Toc440957185)

[3.3.2 Diagram Icons and Arrow Types 11](#_Toc440957186)

[4 Conformance 13](#_Toc440957187)

[Acknowledgments 14](#_Toc440957188)

[Revision History 15](#_Toc440957189)

# Introduction

[All text is normative unless otherwise labeled.]

The Cyber Observable Expression (CybOXTM) Language provides a common structure for representing cyber observables across and among the operational areas of enterprise cyber security. CybOX improves the consistency, efficiency, and interoperability of deployed tools and processes, and it increases overall situational awareness by enabling the potential for detailed automatable sharing, mapping, detection, and analysis heuristics.

This specification document provides brief summary information on the form and use of the CybOX Language UML model. In addition to this textual specification document, [*CybOX Version 2.2.1 Part 6: UML Model*](#Additional_Artifacts) consists of an actual digital serialization of the UML model and a set of relevant UML diagrams extracted from the UML model and used throughout the CybOX Language specification.

In Section **1.1** we discuss the additional specification documents, in Section **1.2** we provide document conventions, and in Section **1.3** we provide terminology. References are given in Sections **1.4** and **1.5**. In Section **2**, we give summary information on the form of the digitally serialized UML model artifact, and in Section **3** we provide general information and conventions for how the UML model is used to define the individual data models. Conformance information is provided in Section **4**.

## CybOX™ Specification Documents

Specification documents have been written for each of the individual data models that compose the full CybOX UML model.

CybOX has a modular design comprising two fundamental data models and a collection of Object data models. The fundamental data models – CybOX Core and CybOX Common – provide essential CybOX structure and functionality. The CybOX Objects, defined in individual data models, are precise characterizations of particular types of observable cyber entities (e.g., HTTP session, Windows registry key, DNS query).

Use of the CybOX Core and Common data models is required; however, use of the CybOX Object data models is purely optional: users select and use only those Objects and corresponding data models that are needed. Importing the entire [CybOX suite of data models](#AdditionalArtifacts) is not necessary.

The [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts) document provides a comprehensive overview of the full set of CybOX data models, which in addition to the Core, Common, and the eighty-eight Object data models, includes a set of default controlled vocabularies. [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts) also summarizes the relationship of CybOX to other externally defined data models, and outlines general CybOX data model conventions.

## Document Conventions

The following conventions are used in this document.

### Fonts

The following font and font style conventions are used in the document:

* Capitalization is used for CybOX high level concepts, which are defined in [*CybOX Version 2.1.1 Part 1: Overview*](#AdditionalArtifacts).

Examples: Action, Object, Event, Property

* The Courier New font is used for writing UML objects.

Examples: ActionType, cyboxCommon:BaseObjectPropertyType

Note that all high level concepts have a corresponding UML object. For example, the Action high level concept is associated with a UML class named, ActionType.

* The ‘*italic’* font (withsingle quotes) is used for noting actual, explicit values for CybOX Language properties. The *italic* font (without quotes) is used for noting example values.

Example:  *‘HashNameVocab-1.0,’ high, medium, low*

## Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

## Normative References

[RFC2119] Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels”, BCP 14, RFC 2119, March 1997. <http://www.ietf.org/rfc/rfc2119.txt>.

## Non-Normative References

**[****GitHub-IO]** CybOX – Cyber Observable eXpression | CybOX Project Documentation. (n.d.). The MITRE Corporation. [Online]. Available: <http://cyboxproject.github.io/>. Accessed Dec 15, 2015.

**[****UML-2.4.1**] Documents associated with Unified Modeling Language (UML), V2.4.1. (Aug. 2011). The Object Management Group (OMG). [Online]. Available: <http://www.omg.org/spec/UML/2.4.1/>.

**[XMI**] Documents associated with XMI Version 2.1. (September 2005). The Object Management Group (OMG). [Online]. Available: <http://www.omg.org/spec/XMI/2.1/>.

**[PNG]** Portable Network Graphics (PNG) Specification (November 2003). The World Wide Web Consortium (W3C). [Online]. Available: <http://www.w3.org/TR/PNG/>.

# UML Model Artifact

The CybOX UML model is formally represented in the form of a digital serialization using the XML Metadata Interchange (XMI) language. The XMI language is intended to be an open standardized form supporting the expression of UML models in a non-proprietary manner. In reality, many UML modeling tools tend to include some proprietary elements in their XMI output. The CybOX UML model was produced using Rational Software Architect (RSA) version 9.1, a product of the IBM Corporation. Effort has been made to minimize the level of proprietary content (from the RSA tool) in the XMI serialization, but it should be noted that some portion may still remain.

For the broadest possible interoperability between UML tools the model is provided as an XMI serialization using UML2.2/XMI2.1 [[XMI]](#XMI) containing only the model and not the diagrams. A set of relevant UML diagrams, extracted from the UML model and leveraged throughout the CybOX Language specification documents, is also provided in a rastered (portable network graphics [[PNG]](#PNG)) form.

In addition, for those with tools that can import the more complete RSA tool native .EMX format, the model with embedded diagrams is also provided in this form.

# Data Model Conventions

The following general information and conventions are used to define the individual data models in UML.

## UML Packages

Each CybOX data model is captured in a different UML package (e.g., Core package, FileObj package, etc.). To refer to a particular class of a specific package, we use the format package\_prefix:class, where package\_prefix corresponds to the appropriate UML package. **Table 3‑1** lists some of the key packages used throughout the CybOX data model specification documents, along with the prefix notation and an example. Each of the eighty-eight CybOX Objects are defined within their own UML package, to support modularity. They are too numerous to mentioned here, but are described in each of the separate specifications documents, [parts 7 through 94](#AdditionalArtifacts).

Table 3‑1. Package prefixes used by the CybOX Language

|  |  |
| --- | --- |
| Package | CybOX Core |
| **Prefix** | **cybox** |
| Description | The CybOX Core data model defines the main classes of the CybOX data model, such as ActionType, EventType, ObservableType, and ObjectType. |
| Example | cybox:ObservableType |
|  | |
| Package | CybOX Common |
| **Prefix** | **cyboxCommon** |
| Description | The CybOX Common data model defines classes that are shared across the various CybOX data models. |
| Example | cyboxCommon:ConfidenceType |
|  |  |
| Package | CybOX Default Vocabularies |
| **Prefix** | **cyboxVocabs** |
| Description | The CybOX default vocabularies define the classes for default controlled vocabularies used within CybOX. |
| Example | cyboxVocabs:ActionTypeVocab |
|  | |
| Package | CybOX Basic Data Types |
| **Prefix** | **basicDataTypes** |
| Description | The CybOX Basic Data Types data model defines the types used within CybOX. |
| Example | basicDataTypes:URI |
|  |  |

## Naming Conventions

The UML classes, enumerations, and properties defined in CybOX follow the particular naming conventions outlined in **Table 3‑2**.

Table 3‑2. Naming formats of different object types

|  |  |  |
| --- | --- | --- |
| **Object Type** | **Format** | **Example** |
| Class | CamelCase ending with “Type” | ActionType |
| Property (simple) | Lowercase with underscores between words | scale |
| Property (complex) | Capitalized with underscores between words | Discovery\_Method |
| Enumeration | CamelCase ending with “Enum” or “Type | DateTimePrecisionEnum; EffectTypeEnum |
| Enumeration value | *varies* | Flash drive; Public Disclosure; Externally-Located |
| Data type | CamelCase, or if the words are acronyms, all capitalized with underscores between words | PositiveInteger; URI |

## UML Diagrams

This document indicates how UML diagrams are used to visually depict relationships between CybOX Language constructs in the rest of the specification. Note that the example diagrams have been extracted directly from the full UML model for CybOX; they have not been constructed purely for inclusion in this or the other specification documents. Typically, diagrams are included where the visualization of their relationships between classes is useful for illustration purposes. This implies that there will be very few diagrams for classes whose only properties are either a data type or a class from the CybOX Common data model. All component data models include a top-level diagram (see **Figure 3‑1**).



Figure 3‑1. Top-level package diagram (ObjectType data model)

In UML diagrams, classes are often presented with their attributes elided, to avoid clutter. The fully described class can usually be found in a related diagram. A class presented with an empty section at the bottom of the icon indicates that there are no attributes other than those that are visualized using associations (see **Figure 3‑2**).

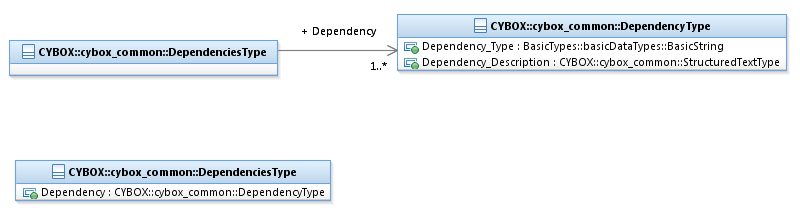


Figure 3‑2. Different presentations of class attributes

### Class Properties

Generally, a class property can be shown in a UML diagram as either an attribute or an association (i.e., the distinction between attributes and associations is somewhat subjective). In order to make the size of UML diagrams in the specifications manageable, we have chosen to capture most properties as attributes and to capture only higher level properties as associations, especially in the main top-level component diagrams. In particular, we will always capture properties of UML data types as attributes. For example, properties of a class that are identifiers, titles, and timestamps will be represented as attributes.

### Diagram Icons and Arrow Types

Diagram icons are used in a UML diagram to indicate whether a shape is a class, enumeration, or data type, and decorative icons are used to indicate whether an element is an attribute of a class or an enumeration literal. In addition, two different arrow styles indicate either a directed association relationship (regular arrowhead) or a generalization relationship (triangle-shaped arrowhead). The icons and arrow styles we use are shown and described in **Table 3‑3**.

Table 3‑3. UML diagram icons

|  |  |
| --- | --- |
| **Icon** | **Description** |
|  | This diagram icon indicates a class. If the name is in italics, it is an abstract class. |
|  | This diagram icon indicates an enumeration. |
|  | This diagram icon indicates a data type. |
|  | This decorator icon indicates an attribute of a class. The green circle means its visibility is public. If the circle is red or yellow, it means its visibility is private or protected. |
|  | This decorator icon indicates an enumeration literal. |
|  | This arrow type indicates a directed association relationship. |
|  | This arrow type indicates a generalization relationship. |

# Conformance

Implementations have discretion over which parts (components, properties, extensions, controlled vocabularies, etc.) of CybOX they implement (e.g., Observable/Object).

[1] Conformant implementations must conform to all normative structural specifications of the UML model or additional normative statements within this document that apply to the portions of CybOX they implement (e.g., implementers of the entire Observable class must conform to all normative structural specifications of the UML model regarding the Observable class and to additional normative statements contained in the document that describes the Observable class).

[2] Conformant implementations are free to ignore normative structural specifications of the UML model or additional normative statements within this document that do not apply to the portions of CybOX they implement (e.g., non-implementers of any particular properties of the Observable class are free to ignore all normative structural specifications of the UML model regarding those properties of the Observable class and any additional normative statements contained in the document that describes the Observable class).

The conformance section of this document is intentionally broad and attempts to reiterate what already exists in this document.

Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged.

Participants:

Dean Thompson, Australia and New Zealand Banking Group (ANZ Bank)

Bret Jordan, Blue Coat Systems, Inc.

Adnan Baykal, Center for Internet Security (CIS)

Liron Schiff, Comilion (mobile) Ltd.

Jane Ginn, Cyber Threat Intelligence Network, Inc. (CTIN)

Richard Struse, DHS Office of Cybersecurity and Communications (CS&C)

Ryusuke Masuoka, Fujitsu Limited

Eric Burger, Georgetown University

Jason Keirstead, IBM

Paul Martini, iboss, Inc.

Jerome Athias, Individual

Sanjiv Kalkar, Individual

Terry MacDonald, Individual

Alex Pinto, Individual

Patrick Maroney, Integrated Networking Technologies, Inc.

Wouter Bolsterlee, Intelworks BV

Joep Gommers, Intelworks BV

Sergey Polzunov, Intelworks BV

Rutger Prins, Intelworks BV

Andrei Sîrghi, Intelworks BV

Jonathan Baker, MITRE Corporation

Sean Barnum, MITRE Corporation

Mark Davidson, MITRE Corporation

John Wunder, MITRE Corporation

Mike Boyle, National Security Agency

Jessica Fitzgerald-McKay, National Security Agency

Takahiro Kakumaru, NEC Corporation

John-Mark Gurney, New Context Services, Inc.

Christian Hunt, New Context Services, Inc.

Andrew Storms, New Context Services, Inc.

Igor Baikalov, Securonix

Bernd Grobauer, Siemens AG

John Anderson, Soltra

Trey Darley, Soltra

Paul Dion, Soltra

Brandon Hanes, Soltra

Ali Khan, Soltra

The authors would also like to thank the larger CybOX Community for its input and help in reviewing this document.

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Editor** | **Changes Made** |
| wd01 | 15 December 2015 | Desiree Beck Trey Darley Ivan Kirillov Rich Piazza | Initial transfer to OASIS template |