Developer Web Conference

14 May 2010

CPE™

cpe.mitre.org

MITRE

Opening Remarks

- CPE v2.3 on fast track to release for public comment by 11 June 2010
- Purpose of today’s conference is to review highlights of planned changes, provide opportunity for real-time discussion
- Not everything you will hear today is cast in concrete—comments/suggestions welcome
- Post-conference feedback to cpe-list is strongly encouraged

Work Schedule

22 Feb:	CPE Developer Day Workshop
16 Mar:	Core Team formed
22 Apr:	V2.3 roadmap posted
10 May:	Naming, Dictionary plans posted
14 May:	Developer web conference
11 Jun:	Draft specs released for public comment
16 Jun:	CPE Developer Day Workshop
9 Jul:	Public comment period closes
23 Jul:	Final v2.3 drafts submitted to NIST

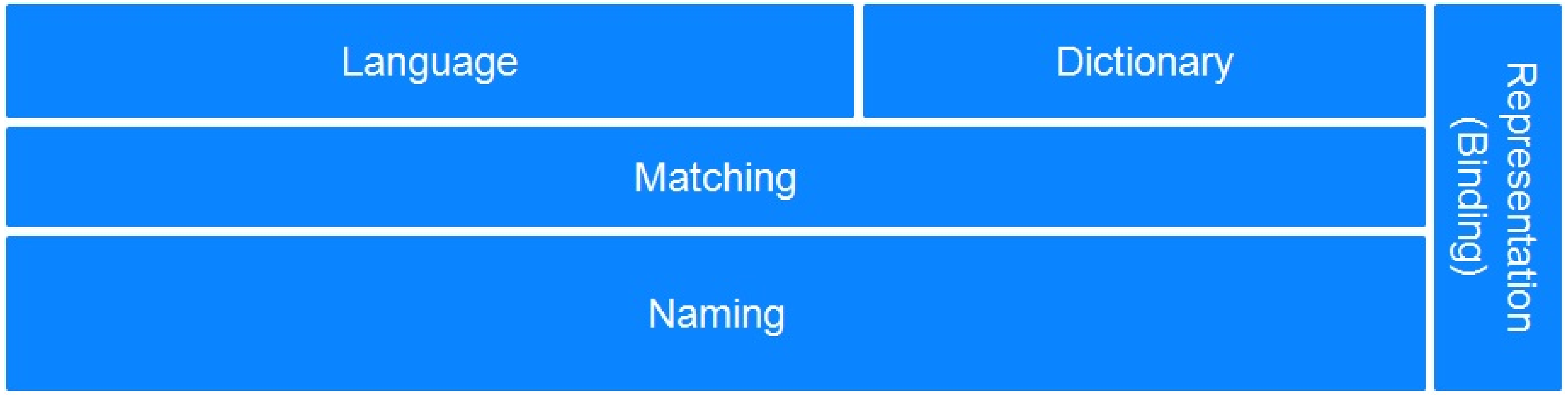
V2.3 Objectives

- Due to the short time frame, changes in CPE 2.3 intended to address immediate community concerns while minimizing risk to adopters
  - Limit potential disruption during release of SCAP 1.2
  - Provide basis for innovation of future capabilities to address larger community needs

Fear Not!

- CPE v2.2 will continue to be supported for several years after v2.3 is released
  - NIST SCAP lifecycle ensures ongoing support
  - V2.2-conformant names will remain valid
  - V2.2 dictionary content will continue to be maintained

Big Picture



- V2.3 implemented as a stack of specifications
- Minimalist Naming specification at the bottom
- Matching builds on Naming
- Dictionary and CPE Language on the top

Naming Specification

- Highlights of the Naming Specification

Key Improvements

- No prefix property
- V2.2 URI binding is retained
- A simple formatted string binding is introduced
  - Easily distinguished from v2.2 URIs by inspection
- Four edition-related attributes are broken out
  - sw_edition, target_sw, target_hw, other_edition
- Need for percent-encoding largely eliminated
- Foundation provided for embedded wildcard characters

Highlights: Naming (1/5)

- Naming specification introduces the concept of a well-formed name (WFN)
  - A conceptual data structure, not machine-readable
  - An unordered set of attribute-value pairs
  - Attributes selected from a specified vocabulary
  - Each attribute appears at most once in a WFN
  - Values of attributes are character strings
  - Some attributes have specified valid values, for most others the Naming specification recommends that values be chosen from valid-values lists

Highlights: Naming (2/5)

- Key ideas:
  - Separate the specification of a WFN from the specification of how a WFN is <u>bound</u> to a machine-readable representation
  - Support two distinct uses of WFNs:
    - Partial (potentially ambiguous) descriptions of products, for matching against a dictionary
    - Identifiers for individual products listed in a dictionary
  - A WFN need not match anything in a dictionary
    - Being “well formed” does not mean “correct”, “valid”, or referring to an actual product

Highlights: Naming (3/5)

- **No prefix property**
- Allowed attributes:
  - Imported from 2.2:
    - Part, vendor, product, version, update, edition, language
    - Edition is deprecated
  - New in 2.3:
    - Sw_edition, target_sw, target_hw, other_edition
- Legacy dictionary content will not be converted to take advantage of new attributes

Highlights: Naming (4/5)

- **Need for “percent encoding” largely eliminated**
- Most formerly-reserved characters now permitted to be embedded in value strings
  - Allows upper-stack specifications to attach special interpretations to particular characters, e.g., use ‘?’ and ‘*’ as wildcards
- Several characters handled specially:
  - Asterisk, Dollar-sign, Question-Mark, Hyphen

Highlights: Naming (5/5)

- To create names for machine interchange, WFNs are bound to machine-readable encodings
- Two bindings supported in v2.3
  - URI binding
    - For backward compatibility w/ v2.2
  - Formatted string binding
    - New in v2,3
- Either binding may be used
  - Mechanical conversion algorithms will be provided

URI Binding: Example 1

WFN: [part="a",vendor="adobe",product="acrobat++",version="9.2", edition="*"]

Straightforward binding to v2.2-conformant URI, respecting the component order defined in v2.2:

```
cpe:/a:adobe:acrobat%43%43:9.2:-::-
```

Notes:

- Reserved characters must be percent-encoded
- Unspecified attributes in WFN bind to single hyphen
- Asterisk and dollar-sign, when used alone, bind to blank
- Asterisk/dollar-sign <u>embedded</u> in a value are deleted

Formatted String Binding: Overview

Looks like this:

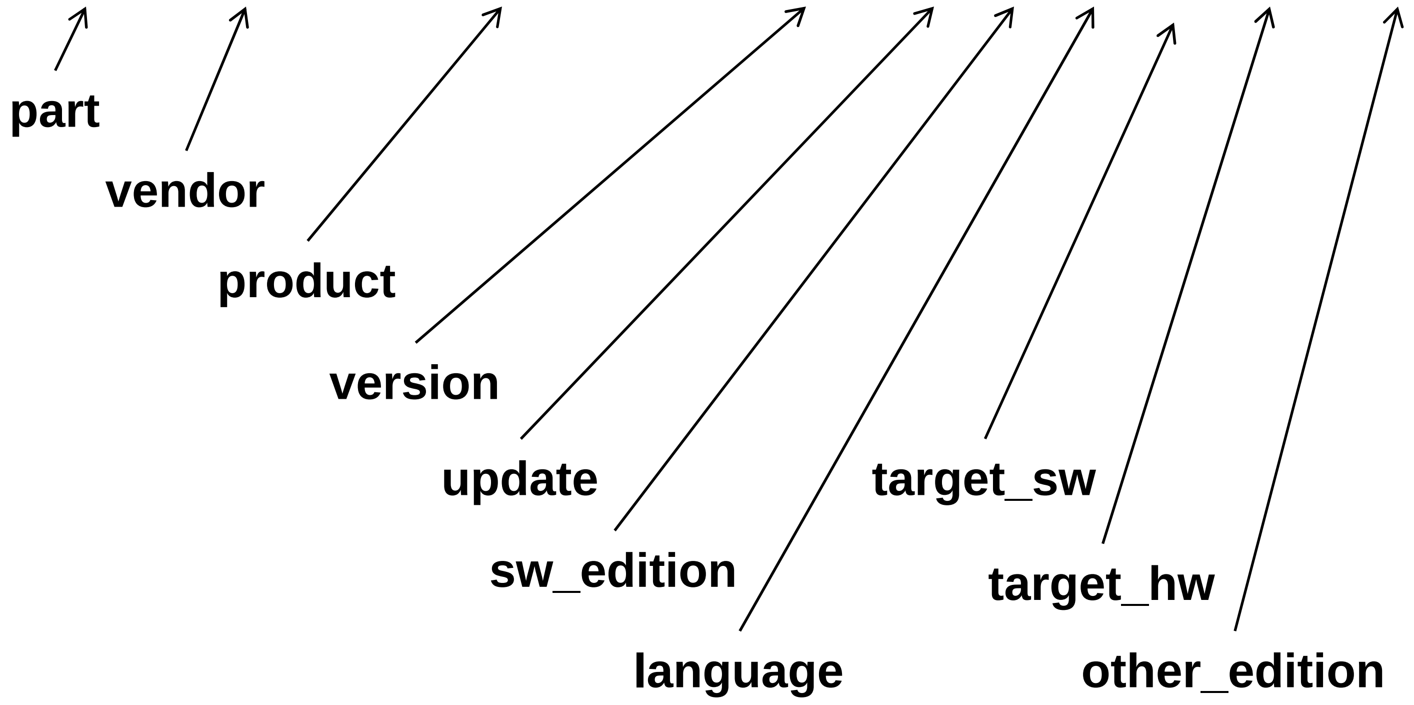
```
cpe-2.3:/<part>:<vendor>:<product>:
  <version>:<update>:<sw_edition>:
  <language>:
  <target_sw:>:<target_hw>:<other_edition>
```

Notes:

- Distinct URI-like scheme name
- Using a “URI-like” binding to minimize differences from 2.2

Formatted String Binding

```
cpe-2.3:/a:adobe:acrobat++:9.2:-:*:-:-:x64:-
```



Formatted String Bindings: Example 1

WFN: [part=“a”,vendor=“adobe”,product=“acrobat++”,version=“9.2”, sw_edition=“*”,target_hw=“x64”]

Binds to

```
cpe-2.3:/a:adobe:acrobat++:9.2:-:*:-:-:x64:-
```

Notes:

- Reserved characters are not percent-encoded
- Unspecified attributes in WFN bind to single hyphen
- No special handling of $, *, etc.—may be used and embedded as wildcards

URI Binding: Example 2

WFN: [part=“a”,vendor=“adobe”,product=“acrobat++”,version=“9.2”, sw_edition=“*”,target_hw=“x64”]

Binds to

```
cpe-2.3:/a:adobe:acrobat%42%42:9.2:-:~~-~x64~-:-
```

Notes:

- `“~~-~x64~-”` is a “packed” encoding of the four extended edition attributes introduced in v2.3

Packing

- “Packing” algorithm used to consolidate the four extended edition attributes into a single component value in the 2.2 URI binding

```
…:~<sw_ed>~<t_sw>~<t_hw>~<o_ed>:…
```

- Tilde (~) used to sub-delimit the four fields
- Not currently used in 2.2 dictionary
- Leading tilde serves as flag

Key Improvements Redux

- No prefix property
- V2.2 URI binding is retained
- A simple formatted string binding is introduced
  - Easily distinguished from v2.2 URIs by inspection
- Four edition-related attributes are broken out
  - sw_edition, target_sw, target_hw, other_edition
- Need for percent-encoding largely eliminated
- Foundation provided for embedded wildcard characters

Matching Specification

- Highlights of the Matching Specification

Highlights: Matching (1/3)

- The CPE 2.3 Matching Specification will define two forms of matching:
  1. The current CPE 2.2 Name Matching algorithm
  2. An extended CPE 2.3 Name Matching algorithm that adds functionality for matching the additional CPE components and special characters.
- CPE Language matching will be defined in the CPE Language Specification as an extension to the Name Matching definition.

Highlights: Matching (2/3)

- In order to maintain backward compatibility with the 2.2 matching algorithm the CPE Matching specification will extend the CPE Naming Specification to:
  - Add a specified component position constraint to a WFN;
  - Reserve the use of 2.2 component-level special characters, “empty” and “-”.
- We will also revise the verbiage in the 2.2 specification to clarify the functionality and scope of the name matching algorithm

Highlights: Matching (3/3)

- In order to expand matching capabilities in CPE 2.3 the CPE 2.3 Matching Specification will:
  - Define special characters to be used in WFN for name matching purposes.
  - * = a multi-character wild card
  - ? = a single character wild card
- Define a 2.3 matching algorithm that
  - Makes use of the new characters
  - Matches CPE ID to CPE ID
  - Matches CPE ID to a WFN

Dictionary Specification

- Highlights of the Dictionary Specification

Highlights: Dictionary (1/7)

- Dictionary Specification will define the concept of a dictionary and high-level rules for dictionary creators.
  - Defines how organizations instantiate dictionaries
  - Defines accompanying documents dictionary maintainers must create and maintain
  - Defines high-level, global rules for CPE name acceptance criteria
  - Define data model for capturing provenance information relating to CPE names
  - Does not single out any specific dictionaries as official

Highlights: Dictionary (2/7)

- **Dictionary is a repository of product identifiers**
  - A CPE Name serving as an identifiers is different than an abstract CPE name representing a set of products
- Only fully-qualified names permitted within the dictionary.
  - Fully-qualified means all CPE attributes must be populated with data (no blanks, ‘*’ or ‘?’ permitted).
  - Part, vendor, product, version attributes of CPE must be populated with known data.

Highlights: Dictionary (3/7)

- Dollar Sign ($) special character will be introduced for use in identifiers
  - ‘$’ is a full-component wildcard within a CPE name that represents data which is unknown, or which is not valued by a particular community
  - Dollar Sign ($) not permitted within part, vendor, product, or version component
    - These components must contain known data

Highlights: Dictionary (4/7)

- Use of Dollar Sign supports matching a more specific name against a less specific dictionary name
  - For example, if scanner finds product "cpe:/o:microsoft:windows_xp:6.0:gold:sp1:en_US" it will match against the dictionary entry "cpe:/o:microsoft:windows_xp:6.0:$:$:$"
- This use case is not supported in CPE 2.2
  - If a scanner finds the product "cpe:/o:microsoft:windows_xp:6.0:gold:sp1:en_US" it would not match against the dictionary entry "cpe:/o:microsoft:windows_xp:6.0"

Highlights: Dictionary (5/7)

- Dollar Sign is distinct from ‘*’
  - At matching level these characters mean similar things, but the semantics change higher in the stack.
  - ‘$’ represents “unknown” data vs ‘*’ which means “any” data
  - Allows explicit distinction between identifiers and names used in searching/applicability statements
- Different rules associated with ‘$’ and ‘*’
  - ‘$’ is full component wildcards only.
  - Conversion rules are different between a ‘$’ and ‘*’.
    - Different conversion rules result from different meaning

Highlights: Dictionary (6/7)

- **Metadata repositories will handle abstract CPE names**
  - Abstract names do not identify unique products and therefore do not belong in dictionary.
  - Metadata repositories can be stood up to capture metadata relating to abstract names.
- Metadata repository will not be formally defined in specification
  - Metadata repository Spec can be written outside of CPE 2.3 as a separate portion of the stack.

Highlights: Dictionary (7/7)

- Dictionary specification will require dictionary maintainers to produce accompanying documents.
- Dictionary Content Management/Decisions Document
  - Will define content management rules associated with dictionary content
  - Capture community decisions relating to how to populate component values (e.g. API calls, file locations)
- Dictionary Process Management Document
  - Will capture any dictionary specific process information (e.g. CPE name acceptance criteria)

CPE Language Specification

- NO SIGNIFICANT CHANGES ENVISAGED
- UPDATED TO BE CONSISTENT WITH LOWER-LEVEL STACK SPECIFICATIONS

THE END

- Please engage on the discussion list
  - What do you like about what you see in 2.3?
  - What don’t you like?