



December 3-6, 2007, Santa Clara Marriott, Santa Clara, CA

CIM V3 Status and Outlook

Jeff Piazza
Hewlett-Packard
Architecture WG

Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change. The Standard Specifications remain the normative reference for all information.
- For additional information, see the Distributed Management Task Force (DMTF) Web site.





Current Status, CIM Specification

- DSP0004, CIM Infrastructure, V2.4, released as Preliminary Specification
- Version numbers of CIM Infrastructure and CIM schema long ago parted company (CIM schema 2.16 is current)
- Current V3 “working document” identifies 30-40 issues



DSP0004 V2.4 highlights

- Clarify STATIC qualifier (ARCHCR00082)
- Clarify EXPERIMENTAL and DEPRECATED are incompatible (90)
- New PUNIT qualifier to replace UNITS qualifier (72)
- New qualifiers for OCL constraints (MethodConstraint, ClassConstraint, PropertyConstraint) (57)
- Clarify arrays (81)
- Clarifications on datetime data type:
 - special values (56)
 - arithmetic and comparison (39)
 - ISO standard calendar (50)



DSP0004 Next Releases

- DSP0004 V2.4 Preliminary released (or about to be)
- DSP0004 V2.4 Final will differ only to correct *errors introduced in V2.4 Preliminary*, as well as to incorporate feedback from implementation. Final occurs when two companies can claim implementation experience.
- DSP0004 V2.5 will be an ISO-ized version of V2.4. ISO conversion to begin immediately based on V2.4 Preliminary.
- Expect to publish V2.5 preliminary in March, 2008. (V2.4 may not be final by then.)
- CRs for “new business” must be addressed to DSP0004 V2.6



Compatibility Requirements

- DMTF process requires no incompatible changes for a minor revision (e.g. 2.x) of a specification or schema
 - Except errata, of course
- A minor revision of the CIM Infrastructure specification cannot require changes to parsers, browsers, or other tools that consume CIM MOF (per DSP4004, section 4.5).
- Ergo, even small, slightly-incompatible changes get deferred to the next major revision
- Even small incompatibilities are presumed to incur very large costs to implementations
 - No mechanism for estimating these costs or evaluating whether a change is worth the cost

Degrees of Incompatibility

More disruptive/
Less compatible

- Abandon CIM, and jump completely to UML
- Replace LogicalIdentity with multiple inheritance

Opinions vary!

- Convert to InstanceID everywhere

- Introduce new data types (enumerations, etc.)

Less disruptive/
More
compatible

- Remove TRANSLATABLE flavor
- Remove WEAK and PROPAGATED



Dimensions of Incompatibility

- Changes that affect CIMOMs
- Changes that affect providers
- Changes that affect MOF files
- Changes that affect clients
 - Changes that affect client libraries
 - Changes that affect how the client operates



Two Views of Scope for CIM V3

- The “Refresh” View
 - Let’s fix the sharp edges
 - Small incompatibilities aren’t really that expensive
 - “Major” revisions should occur at a faster tempo, and then needn’t be so major
- The “Full” View
 - The *number* of incompatible versions, rather than the degree of change, is what governs cost
 - The world can’t afford multiple small incompatible changes, and will reject CIM if it changes too often
 - Let’s re-architect deeply, so we won’t need CIM V4 for a long, long time



Questions to Consider

- Is there a “middle ground” for changes that are technically incompatible, but that few would notice in practice?
- Can new features be introduced with a compatible fallback, to lessen their impact?
 - E.g., could a new embedded instances data type be presented in the current kludgy string form to clients that expect that?
- How will incompatibilities between CIM V2 and V3 be managed in practice?



Impact Assessments

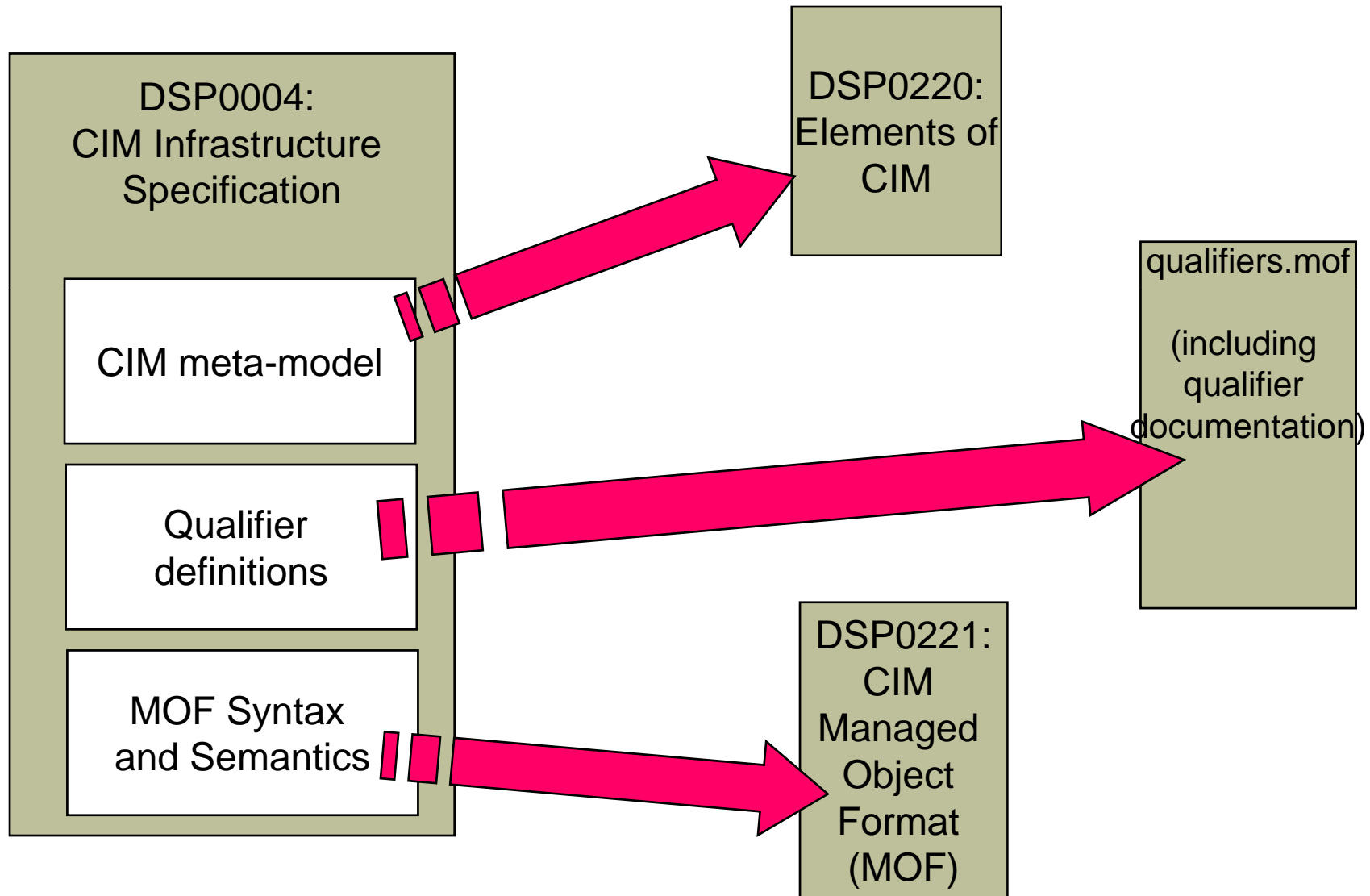
- Items assessed on a scale of none, slight, moderate, or severe disruption (to existing code).
- Most issues are assessed at ***slight*** to ***moderate*** disruption.
- Degree of disruption depends on perspective: schema, server, provider, client library, client application, tools.
- In some cases, disruption is significantly greater for entities that wish to be “version agnostic,” interoperating with both v2.x and v3.x components.
- In the following slides, most significant impact assessment shown [in square brackets]



Done or Nearly So

- Remove WEAK and PROPAGATED qualifiers [slight]
- Establish mechanism for handling property name collisions [slight]
- Split up DSP0004 into separate documents (meta-model, qualifier descriptions, MOF syntax) [none]
- Change default value for IN qualifier [slight]
- Adjust scope definitions (class, association, property, reference) [slight]
- Remove flavors on qualifiers (restrict to qualifier definitions) [slight]
- Limited method overloading (add optional parameters) [moderate to cimoms]
- Remove TRANSLATABLE qualifier flavor [slight]
- OctetString data type [slight]

CIM V3 Specification Plan





Working Issues

These issues require more work to develop complete proposals, but these can confidently be achieved.

- More standards-compliant Datetime data type [slight to moderate]
- Embedded instances [moderate]
- Enumerations [moderate]
- Translatable strings [moderate]
- Add void return type for methods [slight]



Requirements for Enumeration Data Type

- Enumeration is a subtype of some type (e.g., enumeration of uint16).
- Underlying value is stored in the original data type.
- Enumeration symbols are NOT for display to humans (internationalizable), they're restricted to follow rules for property names. (New restriction.)
- Not limited to numeric values.
- Enumerations no longer have ranges.
- Introduce a value description for each value.
- Enumeration defines semantics for values, almost more than it defines symbolic names for values.
- Nothing really new about where translation between symbol and value occurs – leverage existing values/valuemap model. Symbolic names supported only within clients (above the communication library) and the provider. Protocols could be extended to include (or not) symbols along with values.
- Specialization of symbols? Not bothering with this.

Under Discussion

- Aspect-Oriented CIM — Proposal from Andy Maier [moderate]
- Structs and unions [moderate]
- Transient instances [slight]
- Forward references in MOF [slight]



Topics for Other Working Groups

These topics relate to some commonly-occurring patterns of modeling, rather than to specific features of the meta-model. Addressing them would be on a case-by-case basis in individual working groups.

- Use of InstanceId property everywhere [moderate to individual providers]
- Consistency of class naming [slight]
- Remove “excessive” subclassing [slight]
- Consistency of enumerations (values for Unknown, Other) [slight]

Treated as a V2.x Issue

These topics have been addressed by work in CIM 2.x, rather than CIM V3.

- UML alignment — although there will be additional work to account for V3 changes in the UML mapping. [none]
- Add Object Constraint Language (OCL) annotations [none]
- Correlatable names [none]
- Allow arrays of references as method arguments [none]
- Meta-model missing parameter elements [none]



No Agreed Solution/Direction

Not every proposal enjoys universal support, and some topics are just problems in search of a solution.

- Single “top” class for CIM schema [slight]
- Namespaces overhaul [?]
- Clarify cross-namespace associations [?]
- “Future-Proofing” [?]
- “Virtual” instances [?]



Your Feedback Matters!

- What form CIM V3 ultimately takes depends largely on feedback from users and developers
- What are your pain points using CIM?
- What definition of CIM V3 would make it worth crossing the compatibility hurdle for you?
- Join Architecture WG