IETF API Considerations

Dave Thaler
dthaler@microsoft.com
Concrete vs Abstract APIs

- Like pseudocode vs actual code
- Concrete API
  - Specific programming language
  - Exact method names specified
  - Exact data types/structure specified
- Abstract API
  - Programming language agnostic
  - Exact method names may vary by concrete API
  - Exact data types/structure may vary
Abstract example (RFC 2743)

2.1.2: GSS_Release_creds call

Input:
• cred_handle CREDENTIAL HANDLE -- if GSS_C_NO_CREDENTIAL is specified, the call will complete successfully, but will have no effect; no credential elements will be released.

Outputs:
• major_status INTEGER,
• minor_status INTEGER
Concrete example (RFC 2744)

typedef gss_uint32 OM_uint32;

OM_uint32 gss_release_cred (  
    OM_uint32 *minor_status,  
    gss_cred_id_t *cred_handle)
What SDO Owns Concrete APIs?

- Sometimes a bit fuzzy
- JavaScript (ECMAScript): W3C
- POSIX C: The Austin Group (ISO/IEC + IEEE + The Open Group), published as IEEE docs
- Others: ???
The IETF

• Rarely specifies concrete APIs
• Has sometimes normatively specified C APIs that never became part of the POSIX standard
• Has sometimes informatively documented C APIs that are part of the POSIX standard
• Case Study: RFC 3678 (multicast source filter API)
  – Reviewed in parallel by IETF and Austin Group
  – AG feedback: setsockopt/ioctl not type-safe, use new methods instead
  – Published informational RFC from IETF
  – Published normative POSIX spec from IEEE
Advice from an author of RFC 3678

• SDO that defines semantics should specify abstract API
  – Allows multiple languages to specify concrete APIs

• Language-specific SDO should specify concrete API (syntax)

• Coordination between them is needed