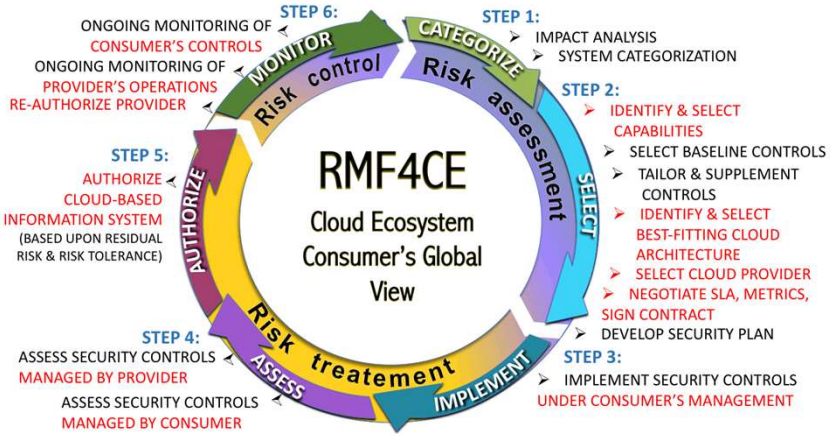
OSCAL

From schema to metaschema

Wendell Piez
National Institute of Standards and Technology
Information Technology Lab

Balisage 2019
Rockville, Maryland

Challenges of the domain

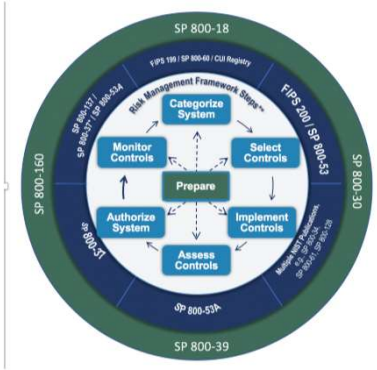


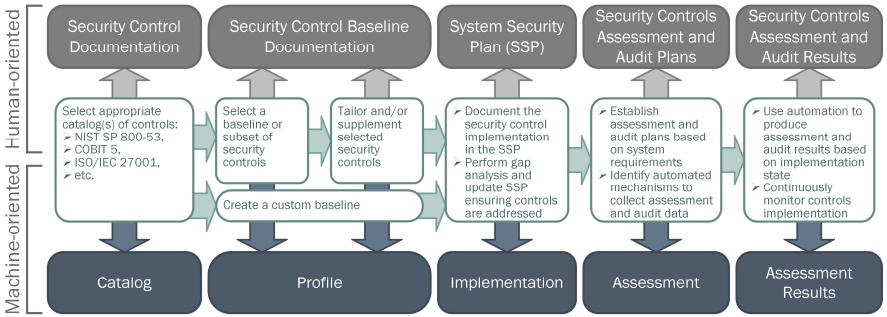
- RMF – Risk Management Framework
- CE – Cloud Ecosystem
- Things I have learned
  - This is one of dozens of depictions
  - "Cloud" is only one leading edge of it
  - We are actually talking about system (security) configuration, documentation, validation and assessment at many levels
  - Across many domains and industries
  - RMF is already a mature idea*

* And mandated for the US Federal Government by the Federal Information System Management Act (FISMA)

Outlines of an information topography

- OSCAL is the “Open Security Controls Assessment Language”
- Supporting a range of activities related to security: planning and coordination, implementation, assessment and more
- Information choreographies already exist, but they are largely manual and unsupported by software or automation
- We believe that a combination of standard reference data with validable models supporting interchange can help change this
- Building trust starts with demonstration and documentation working together



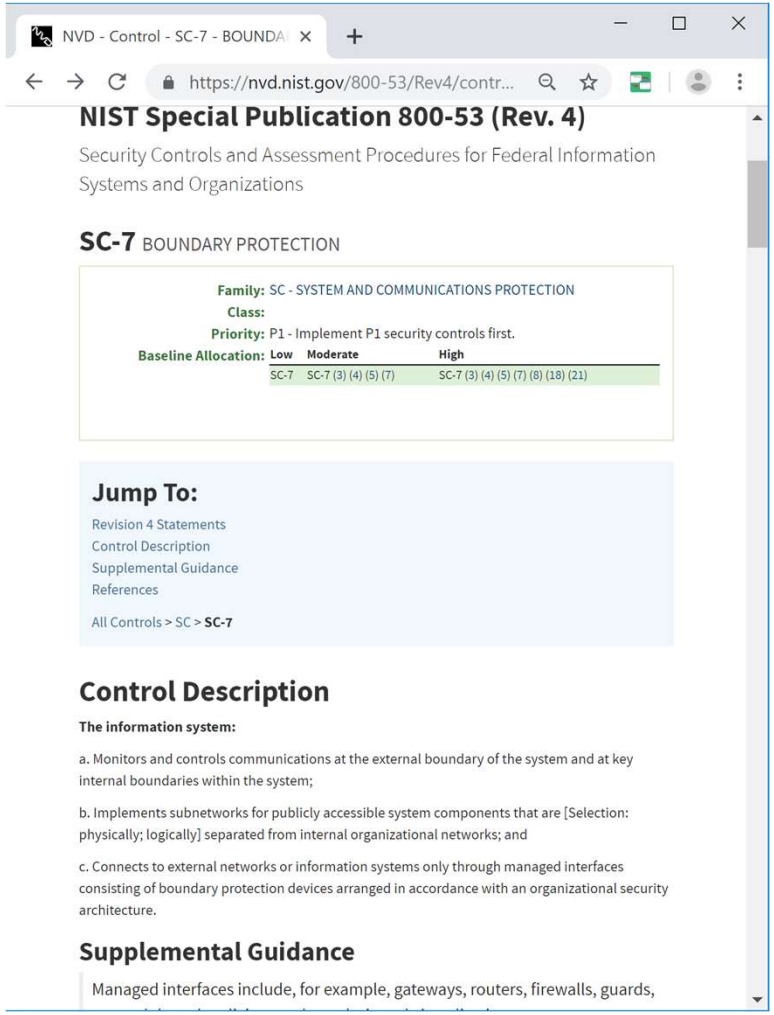


Problem

- Not simply (un)availability of the data
  - Who shares which data with whom
- Not simply all the protocols
- Questions keep coming back to – *scope of governance*
  - Who makes the rules for whom
  - At/for which step(s) of the process(es)
- The domain is large
  - size (number of documents)
  - complexity (functional requirements)

Solution

- Abstract and generic vocabulary
  - very small very reusable tag set
- Provision for organizations to extend by restriction
- *Layered validation* in support of this
  - Base layers generic and permissive
  - Higher layers can be customized by and for user communities and organizations
  - This can be built using tools we know (XSD, Schematron)



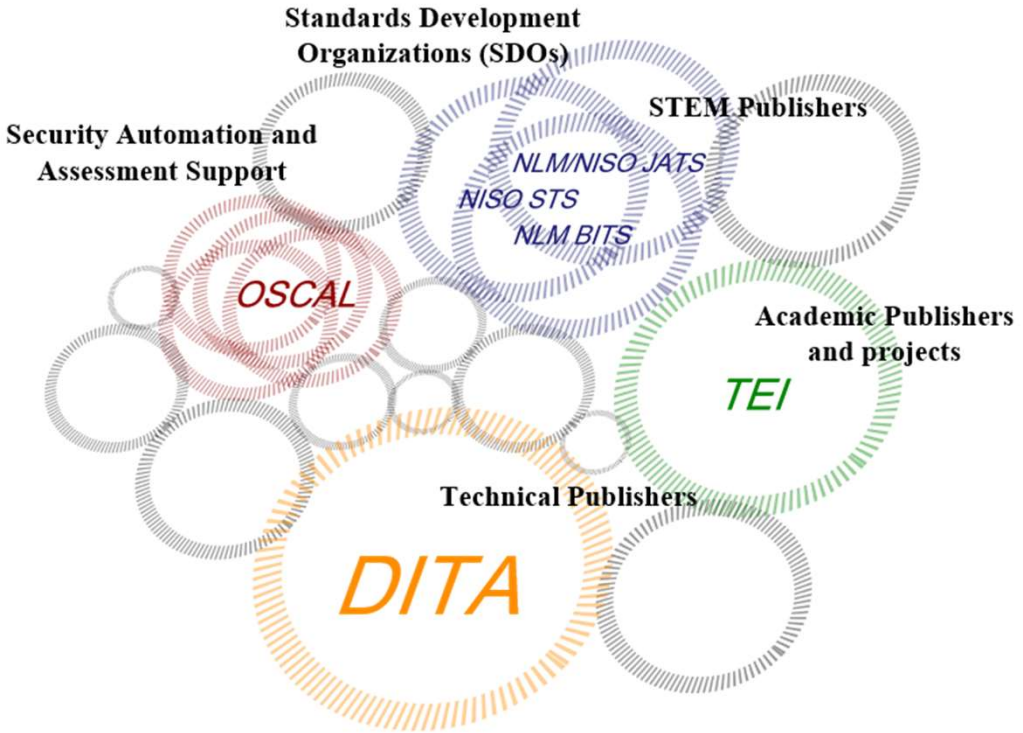
A very simple model

```
<control class="SP800-53" id="sc-7">
  <title>Boundary Protection</title>
  <param id="sc-7_prm_1">
    <select>
      <choice>physically</choice>
      <choice>logically</choice>
    </select>
  </param>
  <prop name="label">SC-7</prop>
  <link href="#ref015" rel="reference">FIPS Publication 199</link>
  <link href="#ref072" rel="reference">NIST Special Publication 800-41</link>
  <link href="#ref093" rel="reference">NIST Special Publication 800-77</link>
  <part id="sc-7_smt" name="statement">
    <p>The information system:</p>
    <part id="sc-7_smt.a" name="item">
      <prop name="label">a.</prop>
      <p>Monitors and controls communications at the external boundary of the
        system and at key internal boundaries within the system;</p>
    </part>
    <part id="sc-7_smt.b" name="item">
      <prop name="label">b.</prop>
      <p>Implements subnetworks for publicly accessible system components that
        are <insert param-id="sc-7_prm_1"/> separated from internal
        organizational networks; and</p>
    </part>
    <part id="sc-7_smt.c" name="item">
      <prop name="label">c.</prop>
      <p>Connects to external networks or information systems only through
        managed interfaces consisting of boundary protection devices arranged in
        accordance with an organizational security architecture.</p>
    </part>
  </part>

. . . </control>
```

But why (not) … ?

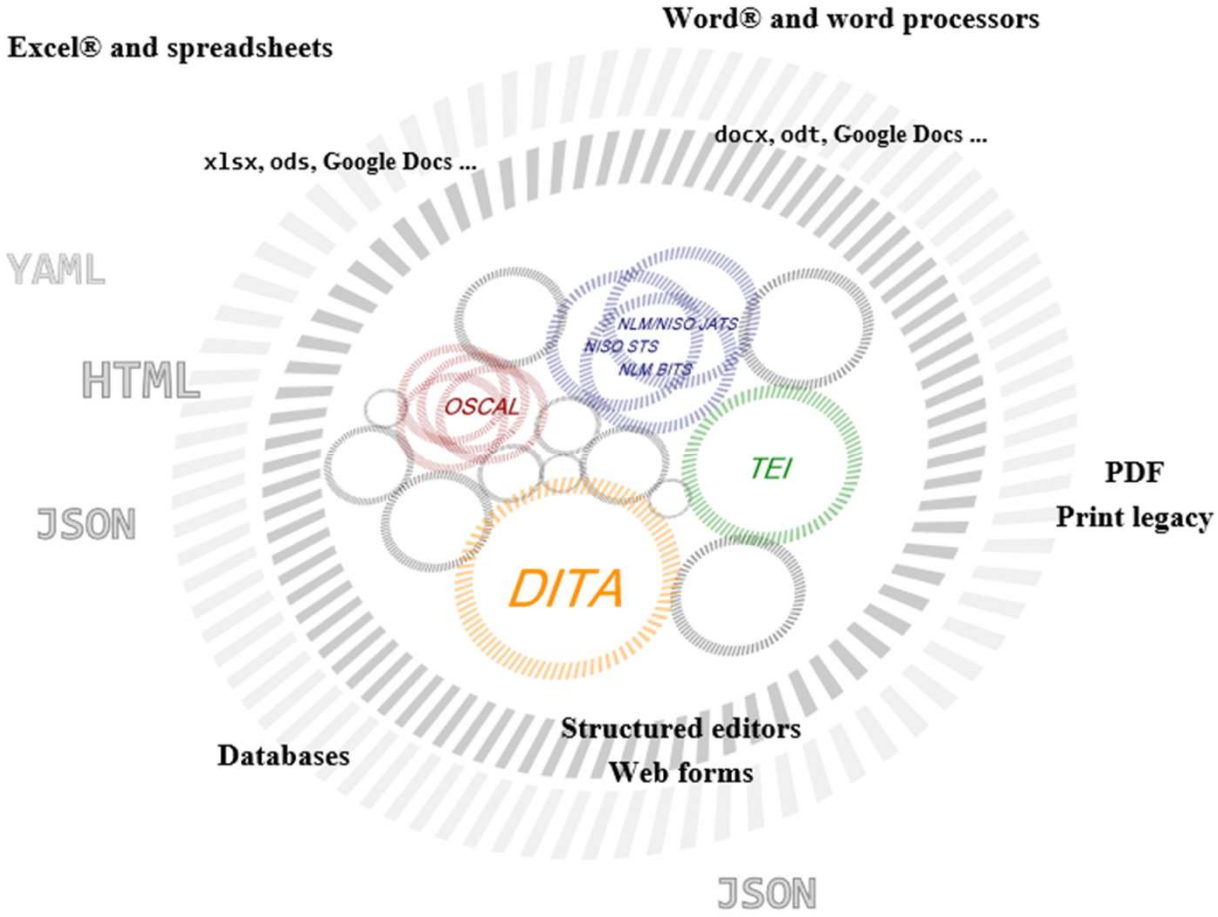
- Would we really be proposing a new XML vocabulary now?
- What about current documentary standards already in wider use?



Actually …

Working with XML is the least of our problems …

- Much of our data is already available in (nominally) machine-readable form
- The problems are
  - Variety of formats
  - Lack of granularity and addressability (“soup”)
  - Lack of transparency / resistance to analysis
  - Need to address needs of disparate consumers
- How do we scale out the capability to map between formats?



We are ambitious

- We seek not only data validation but also
  - To document our schemas as we develop them
  - To build in alignment with JSON and YAML models
  - To produce tools for our formats via automation (machine tooling)
- We can do this with a schema back end: a *metaschema*
- Permits:
  - Iterative, more agile schema development
  - Preemptive solutions to many modeling problems
    - Sparing our users pain and complexity

Requirements for OSCAL Metaschema

- Single source for models *and their documentation*
- The format must be lightweight and easy to learn and use
- Transformation pipelines can produce useful artifacts from a Metaschema:
  - XSD (XML Schema) describing an XML tag set
  - JSON Schema (v7) describing an analogous JSON object
  - Linked and indexed documentation
  - Additional tools
- Alignment with JSON to be guaranteed
  - We treat JSON Schema v7 as a peer of XSD
  - Metaschema also provides a mapping for two-way data conversion
  - Keeping eyes on YAML validation as well

Benefits of the design

- Cost of schema development and deployment is reduced
- Design tradeoffs can be resolved “in the field”
- We can target functionalities we need
  - E.g., lexical checking for datatypes
  - Single control point for any downstream validations
    - Including both XSD and Schematron, or JSON analogs
    - JSON validation with XML technology? (It works!)
- Other tooling produces utilities
  - Simple CSS or HTML-producing XSLT
  - Whitespace normalization utilities
  - Mappings to UI components

What is working so far

Milestone 1 Release June 15 2019 : Milestone 2 Release planned for September

- XSD and JSON Schema v7 are produced for any Metaschema
  - OSCAL Metaschema has its own XSD, Schematron, CSS to support authoring and maintenance
  - May be useful for supporting other bridges between XML and JSON
- We have two metaschemas deployed for two formats, with more coming
  - Catalog: a collection of statements of requirements to be met, as **controls**
  - Profile: a traceable selection and configuration of controls from catalogs for a system or family of systems
- *Additionally*, we produce –
  - Documentation (tag libraries) for both XML- and JSON-based forms of any metaschema
  - **XSLTs that convert from schema-valid XML into equivalent JSON, and back the other way**
- OSCAL demo data
  - Catalogs and profiles for NIST SP800-53 *Security and Privacy Controls for Federal Information Systems and Organizations* and SP800-53A (assessment objectives) *Assessing Security and Privacy Controls in Federal Information Systems and Organizations*
  - Profiles representing FedRAMP baselines of security controls for cloud services (for the Federal Risk and Authorization Management Program of the US General Services Administration)
- Welcoming public interest and engagement *PLEASE BROWSE*
  - Main site: https://pages.nist.gov/OSCAL/
  - Code repository: https://github.com/usnistgov/OSCAL
  - Discussion list: https://email.nist.gov/mailman/listinfo/oscal-dev

Acknowledgements

OSCAL is the Open Security Controls Assessment Language, developed at the Information Technology Laboratory (ITL) of the US National Institute of Standards and Technology (NIST), Gaithersburg Maryland.

OSCAL Team:

- Project lead: Michaela Iorga (NIST)
- Project technical lead and architect: David Waltermire
- Current developers: Wendell Piez, Brian Ruf, Andrew Weiss
- Contributors: John Jedini (GSA), Peter Crayton (GSA), Travis Howerton, Gabe Alford

With thanks to colleagues and supporters who have helped to bring us this far

Thank you!

Questions?

Contact information: oscal@nist.gov

Disclaimer: The opinions expressed in this presentation are the author's own and do not necessarily represent the views of the National Institute of Standards and Technology.