***Balisage:* The Markup Conference**

XForms User Interfaces for Small Arcane Nontrivial Datasets

Joshua Lubell

Computer Scientist
National Institute of Standards and Technology
<lubell@nist.gov>

Abstract

Small Arcane Nontrivial Datasets (SANDs) are frequently complex enough to warrant custom software for access and editing, yet too small or specialized to justify a full-blown server-based database application. Such data is typically presented in tabular form within documents or as editable spreadsheets. To test the alternative of using XForms as a user interface for SANDs, an application was built for browsing a conformance test suite for Product and Manufacturing Information, a formal specification of a product's functional and behavioral requirements as they apply to production. XForms proved a much better match than tabulations for the underlying data model. To further test the concept, XForms was evaluated for use with the NIST Special Publication 800-53 security control catalog, which is a comprehensive catalog of security controls for managing cyber-risk, many parts of which are already available in XML form. The model-view-controller (MVC) software pattern of XForms seems well-suited for creating specialized applications for tailoring and navigating this catalog.

Table of Contents

Introduction

XForms, an Extensible Markup Language (XML) application for specifying forms for the Web [1], adopts the model-view-controller (MVC) software pattern [2]. The model in an XForms XML document is not an explicit schema, but rather a collection of "instances." The MVC approach reduces coding effort, server-side processing, and dependence on scripting languages and browser platforms. Because XForms documents conform to a standard XML schema, forms authored in XForms will age more gracefully than forms dependent upon a particular software vendor or programming environment. Since XForms is “data-driven,” it is well-suited to use cases where the data is already available as XML, or in a format easily transformable to XML.

In this paper, I study the use of XForms for authoring user interfaces (UIs) for a specific class of structured datasets. I call a dataset belonging to this class a *SAND* (Small Arcane Nontrivial Dataset). A SAND is sufficiently complex to merit specialized software for access, yet too small and/or specialized to justify developing a full-blown server-based database application. SANDs are typically presented to users in a tabular format as part of a document, or perhaps as an editable spreadsheet. These presentation methods are cumbersome for any SAND requiring more than a few pages (or screens) to display in its entirety. Additionally, tabular formats do a poor job presenting data where the underlying data model contains cross-references, hierarchical structures, or inheritance relationships.

The rest of this paper focuses on two SANDs. The first is a conformance test suite for Product and Manufacturing Information (PMI) [3], for which I have authored a simple browser application using XForms. The second is the security control catalog specified in the National Institute of Standards and Technology (NIST) Special Publication 800-53, *Security and Privacy Controls for Federal Information Systems and Organizations*, Revision 4 [4]. No one to my knowledge has written a UI in XForms for this second SAND but, as I will discuss later, the security control catalog has several qualities making it a promising candidate for XForms.

Two SANDs as XForms Use Cases

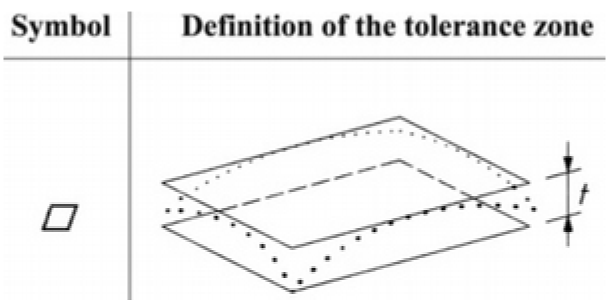
This section presents XForms use cases centered around each SAND. For each use case, I first provide an overview of the subject matter. Next I present a Unified Modeling Language (UML) [5] class diagram showing the SAND's conceptual data model. This conceptual model is my creation — it is not part of the SAND's accompanying documentation. I then show how the SAND is documented by its creator, which is how a user would see the SAND in absence of a specialized software implementation. Finally, I discuss how XForms can be used to define a dynamic UI to the SAND. For the PMI conformance test suite, I describe an actual implementation. For the security control catalog, I provide some implementation guidance.

PMI Conformance Test Suite

Product and Manufacturing Information (PMI) specifies, in a formal and precise language, a product’s functional and behavioral requirements as they apply to production. PMI communicates allowable product geometry variations (tolerances) in form, size, and orientation. PMI annotations include Geometric Dimensioning and Tolerancing, surface texture specifications, finish requirements, process notes, material specifications, and welding symbols [6]. The American Society of Mechanical Engineers (ASME) defines standards for PMI within the United States [7,8], and these standards are complex. If the software applications interpreting PMI do not conform to the standards, the same PMI data is likely to be interpreted and presented differently by different engineering and manufacturing applications. Incorrect presentation and misinterpretation of PMI can cause significant delays and costly errors. For example, an Aberdeen Group study showed that catching PMI anomalies up front provides substantial savings to manufacturers in both time to market and product development costs. One large aerospace supplier reported that, prior to modernizing and improving their engineering processes, more than 30% of their change orders were due to inaccurate PMI [9].

Figure 1 shows the definition for the flatness tolerance symbol (a parallelogram), one of many PMI constructs defined in the ASME standards. The source of this figure is ISO 1101 [10], an International Standard aligned with the ASME PMI standards. When associated with a value `t`, specifying a flatness tolerance on a surface in a product model says that the actual surface of the manufactured part has to be contained within two parallel planes `t` units apart. Unless otherwise specified in the part model or drawing, all units are in millimeters.

Figure 1

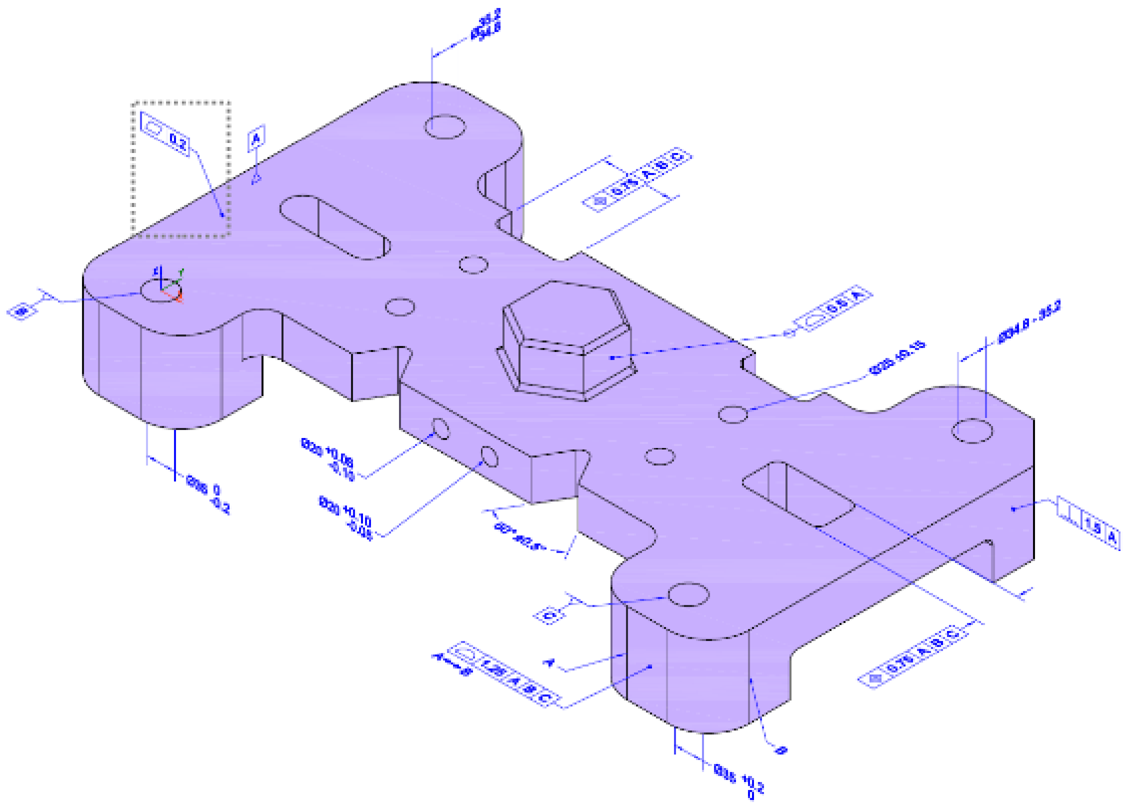


Flatness tolerance symbol as defined in ISO 1101.

Figure 2 shows a two-dimensional (2D) presentation of a part model with PMI. The PMI includes a flatness tolerance of 2 millimeters, shown inside a dashed box on the upper left superimposed on the model. The part model's PMI includes numerous other PMI specifications containing different tolerance symbols, as well as additional PMI symbols.

The PMI conformance test suite consists of a collection of test cases for determining whether computer-aided design (CAD) software correctly implements a representative set of PMI concepts as defined in the ASME Y14.5 *Dimensioning and Tolerancing* [7] and Y14.41 *Digital Product Data Definition Practices* [8] standards. The test cases are PMI-annotated 2D isometric drawings collectively representing the machined bar part shown in Figure 2 plus four other distinct parts. There are currently a total of fifty-five test cases: fifty of them atomic and five of them complex. An Atomic Test Case (ATC) highlights an individual PMI concept to be tested, called the measurand. The ATC is not a complete specification of the part's PMI, but rather contains only the PMI needed to specify enough context information to understand the measurand. The machined bar part in Figure 2 includes multiple ATCs. A Complex Test Case (CTC) is a test case whose PMI is a superset of the PMI of all ATCs associated with that part. Each CTC specifies one of the five distinct parts. Thus, Figure 2 is one of five CTCs.

Figure 2

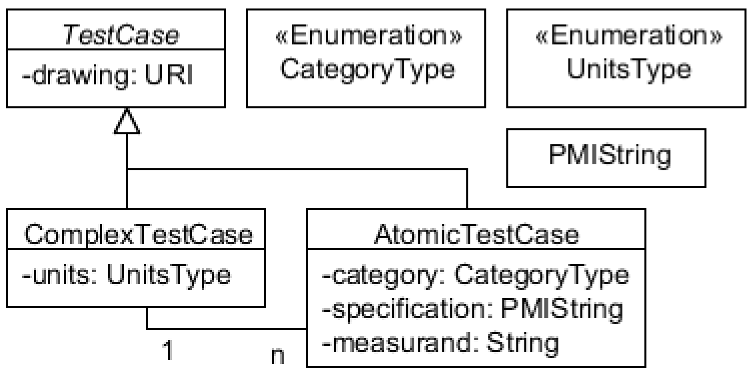


2D presentation of a machined bar part annotated with PMI. Inside the dashed box is a flatness tolerance with `t = 2`. A leader line associates the tolerance with the surface labeled with datum feature symbol `A`.

Figure 3 presents a UML conceptual model of the PMI conformance test suite. A `TestCase` is an abstract (non-instantiable) class whose `drawing` attribute points to the location of a 2D PMI-annotated part model. A `ComplexTestCase` is a `TestCase` with an additional attribute `units` specifying whether dimensions are in metric or English units. An `AtomicTestCase` is a `TestCase` with additional attributes identifying its PMI `category`, the PMI `specification` being tested (with PMI symbols represented as Unicode characters), and a prose description of the `measurand`. Every `ComplexTestCase` is associated with many `AtomicTestCases`. Every `AtomicTestCase` is associated with a single `ComplexTestCase`*.

The PMI subject matter expert who created the test cases provided two spreadsheets to document test suite metadata, examples of which include the association links and attribute values shown in Figure 3. The first spreadsheet provided metadata for each CTC as shown in Table I. Each row corresponds to one of the five CTCs. The first row provides metadata for the machined bar CTC whose drawing was shown in Figure 2. The second spreadsheet provides metadata for each of the fifty ATCs. Table II shows an example row from this spreadsheet, which contains the metadata for the ATC corresponding to the flatness tolerance from the machined bar CTC.

Figure 3



Test case browser conceptual model.

Table I

Metadata for CTCs

ID	Description	Units	Comments	Atomic Test Cases
1	Bar with Simple Features	Metric	Updated model: replaced hex hole with hex boss for manufacturability	1 2 3 4 7 8 17 21 33 48
2	Cast Part	Metric	Most surfaces have draft. Various angles.	26 28 29 31 34 35 41 43 47 50
3	Sheet Metal Part 2	Inch	Created new sheet metal model for this test case	6 13 14 20 27 32 36 39 45 46
4	Machined Part 1 - Simplistic	Metric	Two views created	5 9 10 12 15 16 22 30 40 49
5	Machined Part 2 - Round	Metric	Two views created	11 18 19 23 24 25 37 38 42 44

The test case browser UI I authored with XForms provides access to the 2D presentations of all of the test cases. The browser has three tabs (patterned after the tabbed browsing example in Steven Pemberton's XForms tutorial [11]), each providing a different way to browse the test cases and view their corresponding images. One tab enables users to browse by CTC, drilling down to the ATCs associated with the CTC. Another tab lets users select from a list of all ATCs. The third tab allows users to browse by PMI category. Figure 4 shows a screen shot of the UI. To get to the state shown in the screen shot, the user first selected the machined bar part (“Bar with Simple Features”) CTC using the Complex Test Cases tab. This interaction caused the UI to generate a list of radio buttons representing each of this CTC's ATCs. Next, the user selected the “Feature Control Frame Directed to Surface - Flatness” ATC from the auto-generated list of ATCs, resulting in this ATC's PMI category, specification, and measurand appearing at the bottom of the screen. The user then clicked on the “View Image” button, causing the ATC's 2D image to appear.

Table II

Metadata for the ATC testing for conformance to the ASME definition of flatness tolerance applied to a surface.

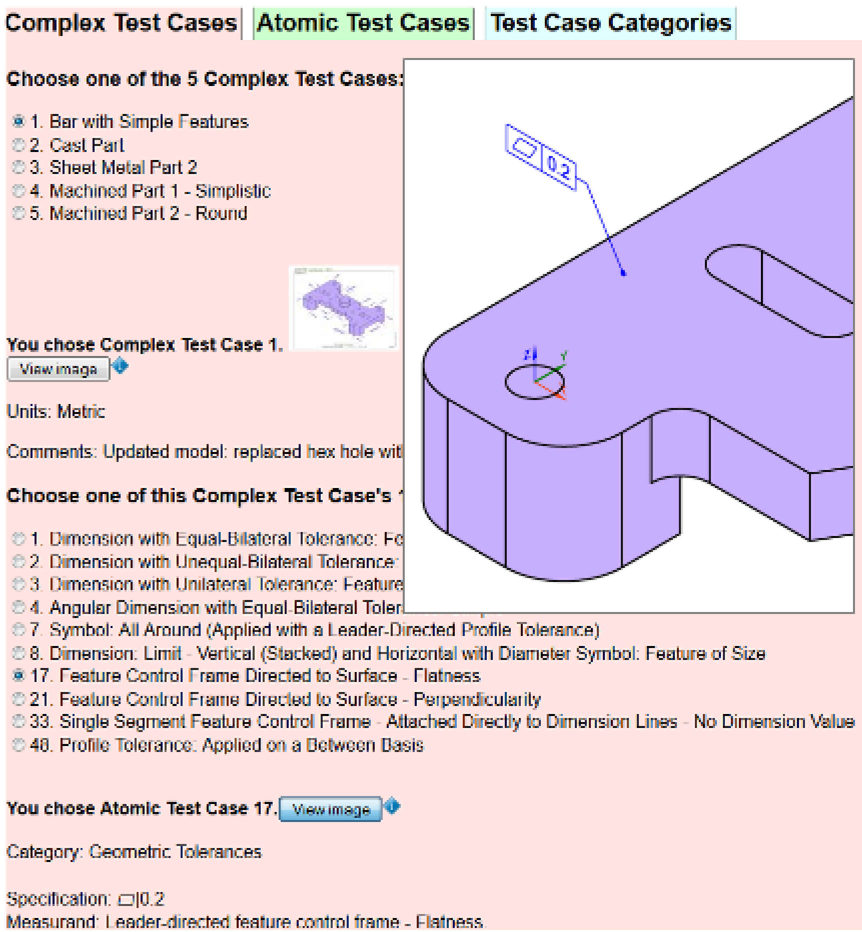
ID	Description	Complex Test Case	Specification	Measurand
17	Feature Control Frame Directed to Surface - Flatness	1	|0.2	Leader-directed feature control frame - Flatness.

The test case browser UI's XForms processor is XSLTForms† [12], which is implemented as an Extensible Stylesheet Language Transformation (XSLT) [13] that runs natively in common Internet browser clients without the need for plugins. I chose XSLTForms both to ensure cross-platform support and to eliminate the need to install any specialized software on the server. Anyone with a typical online desktop computing environment can access the UI. All that is needed is for the browser client to be able to retrieve the files, either from a server or from the local file system.

The source document defining the test case browser UI is an XML document I authored using Leigh Klotz's XHTML+XForms schema [14]. The source document's XForms `model` element contains static instances extracted from the spreadsheet data shown in Table I and Table II. The source document is approximately 500 lines long, excluding the static instance data, which resides in separate files. The `model` element also specifies dynamic instances corresponding to each tab in the UI. The dynamic instance data changes as the user interacts with the form. For example, the instance data associated with the Complex Test Cases tab after the user selected the machined bar CTC and the ATC for flatness tolerance as shown in Figure 4 is as follows:

```
<data>
  <ctcNumber>1</ctcNumber>
  <ctcURL>TestCases/CTC/1.pdf</ctcURL>
  <ctcThumbnailURL>TestCases/CTC/thumbnails/1.jpg</ctcThumbnailURL>
  <atcNumber>17</atcNumber>
  <atcFileName>nist_atc_017_asme1_rc</atcFileName>
  <atcURL>TestCases/ATC/PNG/nist_atc_017_asme1_rc.png</atcURL>
  <ctcCount>5</ctcCount>
  <atcCount>10</atcCount>
</data>
```

Figure 4



Test case browser screen shot.

NIST SP 800-53 Security Control Catalog User Interface

NIST Special Publication 800-53 Revision 4 is a widely-used standard that provides a comprehensive catalog of tailorable security controls for organizations to manage cyber-risk. The controls are organized into eighteen families shown in Table III. SP 800-53 specifies three security control baselines (low, moderate, and high-impact), as well as guidance for customizing the appropriate baseline to meet an organization's specific requirements. In addition to customizing a baseline, an organization or a group of organizations sharing common concerns can create an *overlay* customizing a set of controls with additional enhancements and supplemental guidance. One overlay recently developed is for Industrial Control Systems (ICS), which are prevalent in the utility, transportation, chemical, pharmaceutical, process, and durable goods manufacturing industries. ICS are increasingly adopting the characteristics of traditional information systems such as Internet connectivity and use of standard communication protocols. As a result, ICS are vulnerable to many of the same security threats that affect traditional information systems, yet ICS have unique needs requiring additional guidance beyond that offered by NIST SP 800-53 [15].

Table III

NIST SP 800-53 security control identifiers and family names.

ID	FAMILY	ID	FAMILY
AC	Access Control	MP	Media Protection
AT	Awareness and Training	PE	Physical and Environmental Protection
AU	Audit and Accountability	PL	Planning
CA	Security Assessment and Authorization	PS	Personnel Security
CM	Configuration Management	RA	Risk Assessment
CP	Contingency Planning	SA	System and Services Acquisition
IA	Identification and Authentication	SC	System and Communications Protection
IR	Incident Response	SI	System and Information Integrity
MA	Maintenance	PM	Program Management

Multiple overlays can be applied simultaneously. For example, a consortium of automobile manufacturers might want to develop their own overlay addressing security concerns specific to their industry. They would then apply both the ICS overlay and the automotive-specific overlay to the NIST SP 800-53 security control baselines.

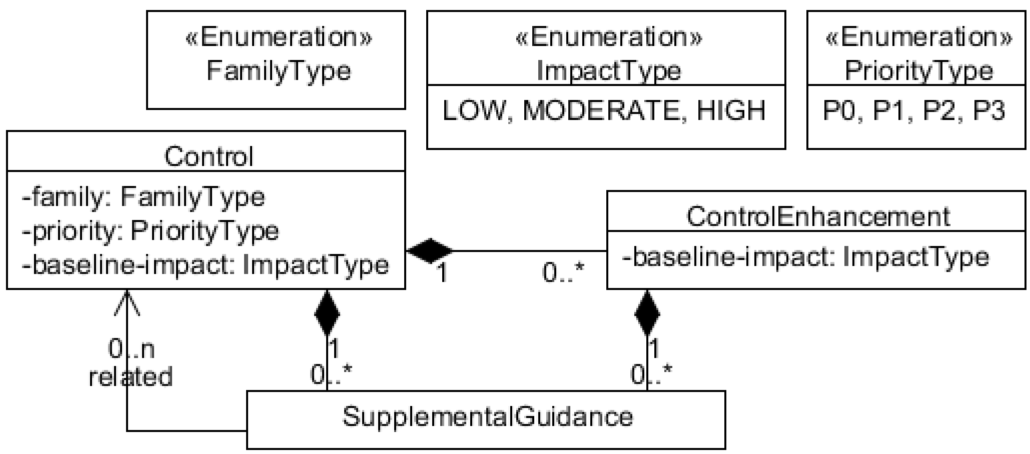
Figure 5 shows a UML conceptual model of the NIST SP 800-53 security control catalog. A `Control` has the following attributes:

- `family` – The ID of the family to which the control belongs.
- `priority` – A designation recommending the order in which the control should be implemented relative to other controls in a baseline. Controls with priority `P1` should be implemented first, followed by `P2` controls, and finally `P3` controls. `P0` controls are not selected in any baseline.
- `baseline-impact` – Specifies whether the control is included in the baseline for low-impact, moderate-impact, or high-impact systems. A low-impact system is a system where the adverse effects from loss of information confidentiality, integrity, or availability would be minimal. For a moderate-impact, system, the consequences

> would be moderate. For a high-impact system, the consequences would be severe. Consequently, the moderate-impact baseline is a superset of the low-impact baseline, and the high-impact baseline is a superset of the moderate-impact baseline.

A `Control` also contains zero or more `ControlEnhancements` and `SupplementalGuidances`. Each of the `Control`'s `ControlEnhancements` has a `baseline-impact` attribute specifying whether the enhancement applies to low, moderate, or high-impact systems. Each of the `Control`'s `SupplementalGuidances` includes zero or more associations to other controls.

Figure 5



Security control conceptual model.

NIST SP 800-53 provides the security control catalog, baselines, and impacts as document appendices. This information totals over 280 pages of text and tables, more than half the page count of the publication as a whole. Figure 6 shows the SP 800-53 definition for the CONTENT OF AUDIT RECORDS control. This control is a member of the Audit and Accountability (AU) family, has a unique ID of `AU-3`, a priority of `P1` (i.e., implementation of this control should be a top priority), a baseline impact of `LOW` (i.e., this control is included in all three SP 800-53 baselines), and has supplemental guidance specifying associations with three other controls in the AU family and one control in the System and Information Integrity (SI) family.

The CONTENT OF AUDIT RECORDS control definition includes two enhancements: (1) ADDITIONAL AUDIT INFORMATION and (2) CENTRALIZED MANAGEMENT OF PLANNED AUDIT RECORD CONTENT. Enhancement 1 has a baseline impact of `MODERATE` (i.e., additional information is required for audit records of moderate and high-impact systems). Enhancement 2 has a baseline impact of `HIGH` (i.e., generation of audit records must be centrally managed for high-impact systems), and is associated two other controls in the AU family.

Figure 6

AU-3 CONTENT OF AUDIT RECORDS

Control: The information system generates audit records containing information that establishes what type of event occurred, when the event occurred, where the event occurred, the source of the event, the outcome of the event, and the identity of any individuals or subjects associated with the event.

Supplemental Guidance: Audit record content that may be necessary to satisfy the requirement of this control includes, for example, time stamps, source and destination addresses, user/process identifiers, event descriptions, success/fail indications, filenames involved, and access control or flow control rules invoked. Event outcomes can include indicators of event success or failure and event-specific results (e.g., the security state of the information system after the event occurred). Related controls: AU-2, AU-8, AU-12, SI-11.

Control Enhancements:

(1) *CONTENT OF AUDIT RECORDS | ADDITIONAL AUDIT INFORMATION*

The information system generates audit records containing the following additional information: [*Assignment: organization-defined additional, more detailed information*].

Supplemental Guidance: Detailed information that organizations may consider in audit records includes, for example, full-text recording of privileged commands or the individual identities of group account users. Organizations consider limiting the additional audit information to only that information explicitly needed for specific audit requirements. This facilitates the use of audit trails and audit logs by not including information that could potentially be misleading or could make it more difficult to locate information of interest.

(2) *CONTENT OF AUDIT RECORDS | CENTRALIZED MANAGEMENT OF PLANNED AUDIT RECORD CONTENT*

The information system provides centralized management and configuration of the content to be captured in audit records generated by [*Assignment: organization-defined information system components*].

Supplemental Guidance: This control enhancement requires that the content to be captured in audit records be configured from a central location (necessitating automation). Organizations coordinate the selection of required audit content to support the centralized management and configuration capability provided by the information system. Related controls: AU-6, AU-7.

References: None.

Priority and Baseline Allocation:

P1	**LOW** AU-3	**MOD** AU-3 (1)	**HIGH** AU-3 (1) (2)

A control as presented in the NIST SP 800-53 document.

As an aid to implementers of NIST SP 800-53, the United States government's National Vulnerability Database (NVD) [16] provides the security control catalog information in an XML format. The following shows a simplified version of the NVD data representing the CONTENT OF AUDIT RECORDS control shown in Figure 6 (with prose text suppressed to shorten the listing).

```
<control>
  <family>AUDIT AND ACCOUNTABILITY</family><number>AU-3</number>
  <title>CONTENT OF AUDIT RECORDS</title>
  <priority>P1</priority><baseline-impact>LOW</baseline-impact>
   <supplemental-guidance>
    <related>AU-2</related><related>AU-8</related>
    <related>AU-12</related><related>SI-11</related>
  </supplemental-guidance>
  <control-enhancement>
    <number>AU-3 (1)</number><title>ADDITIONAL AUDIT INFORMATION</title>
    <baseline-impact>MODERATE</baseline-impact>
    <supplemental-guidance/>
  </control-enhancement>
  <control-enhancement>
    <number>AU-3 (2)</number>
    <title>CENTRALIZED MANAGEMENT OF PLANNED AUDIT RECORD CONTENT</title>
    <baseline-impact>HIGH</baseline-impact>
    <supplemental-guidance>
      <related>AU-6</related><related>AU-7</related>
    </supplemental-guidance>
  </control-enhancement>
</control>
```

User interfaces for navigating the NIST SP 800-53 security controls already exist. These implementations [17,18] contain hyperlinked web pages generated from the SP 800-53 XML data. However, software support for creating and browsing SP 800-53 overlays does not yet exist. XForms is well-suited for authoring overlays for the following reasons:

- NIST SP 800-53 catalog data is already available in XML, so an XForms document can readily use this data as a model instance.
- The conceptual model in Figure 5 is a good fit for XML and XForms. XML naturally represents the composition relationships (connectors with solid diamonds) as element containment. An XForms document can use XPath to retrieve sets of controls based on family, impact, priority, or supplemental guidance.
- The forms in an XForms document are instance data-driven, allowing the author to specify a complex UI as declarative mark-up without the need for a scripting language [19]. This reduces the effort needed to create and maintain SP 800-53 overlay software.

A UI for browsing the security controls catalog and creating overlays can be implemented in a manner similar to that of the PMI test case browser, with static model instances corresponding to XML data from the NVD, the information in Table III, the data sorted by priority, and the data sorted by impact. The latter three instances can easily be generated from the first instance using XSLT.

Although a standardized XML vocabulary for representing overlays does not exist, XML's namespace mechanism provides a handy way to augment the NVD's NIST SP 800-53 XML data with overlay information. As an example, let us revisit the CONTENT OF AUDIT RECORDS control (`AU-3`) shown in Figure 6. The ICS overlay in the draft second revision to NIST SP 800-82 [15] adds additional guidance to this control stating that, if a particular ICS information system lacks the ability to generate and maintain audit records, a separate information system could provide the required auditing capability instead. Now suppose a subset of the ICS community were to create its own sector-specific overlay to be used in addition to the NIST SP 800-82 ICS overlay, and that this sector-specific overlay changes the baseline impact of Enhancement 1 of `AU-3` from `MODERATE` to `LOW`.

The following XML represents the original `AU-3` data augmented with the modifications provided in both the ICS and sector-specific overlays. Each overlay has its own namespace, and the overlay modifications are shown in boldface for readability. XML data not in an overlay namespace is identical to the NVD data for `AU-3` shown previously. The `ics:supplemental-guidance` element contains the additional guidance for `AU-3` as specified in the ICS overlay. The `s:baseline-impact` element contains the change to Enhancement 1's baseline impact as specified in the sector-specific overlay. The `s:rationale` element contains the rationale (prose text not shown) for increasing the scope of the low-impact baseline to include Enhancement 1.

```
<control xmlns:ics="http://www.nist.gov/ics-overlay"
      xmlns:s="http://www.example.com/sector-specific-overlay">
  <family>AUDIT AND ACCOUNTABILITY</family>
  <number>AU-3</number>
  <title>CONTENT OF AUDIT RECORDS</title>
  <priority>P1</priority>
  <baseline-impact>LOW</baseline-impact>
   <supplemental-guidance>...</supplemental-guidance>
  <ics:supplemental-guidance>Example compensating controls include
providing an auditing capability on a separate information
system.</ics:supplemental-guidance>
    <control-enhancement>
      <number>AU-3 (1)</number>
      <title>ADDITIONAL AUDIT INFORMATION</title>
      <baseline-impact>MODERATE</baseline-impact>
      <s:baseline-impact>LOW</s:baseline-impact>
      <s:rationale>...</s.rationale>
      <supplemental-guidance/>
    </control-enhancement>
    <control-enhancement>...</control-enhancement>
</control>
```

Conclusions and Final Thoughts

At the beginning of this paper, I introduced the term *SAND*, an acronym for “Small Arcane Nontrivial Dataset.” Positing a dearth of good user interfaces for accessing SANDs, I then examined the suitability of XForms for creating user interfaces for two SANDs: a PMI test suite and the NIST SP 800-53 catalog of security controls. For both SANDs, I demonstrated a mismatch between (my understanding of) the SAND's underlying conceptual model and the spreadsheets and text documents users of the SAND typically peruse to view the data. For the PMI test suite SAND, I presented a test case browser application I built using XForms. For the security controls SAND, I observed that the dataset is already available in XML, and that its underlying model is highly compatible with XML. I concluded that XForms, with its MVC paradigm, is well-suited for creating specialized applications for tailoring and navigating the catalog. I also demonstrated how a security control's XML data can be augmented to include information from one or more overlays.

SANDs exist under the radar, yet are important because they provide infrastructure essential for deploying larger, more visible data standards used for systems integration. To better understand the role of SANDs in data exchange and interoperability, consider McGilvray's taxonomy of data categories [20]. McGilvray defines *reference data* as “sets of values or classification schemas that are referred to by systems, applications, data stores, processes, and reports, as well as by transactional and master records.” The NIST SP 800-53 security controls are reference data in that they define and classify a set of security procedures and are referred to by security professionals and software applications. Reference data is distinct from the other, more widely-understood *master data* — data describing tangible objects — and *transactional data* — data associated with an event or business process — categories. The PMI test suite falls into a category distinct from reference data, but with some similarities. According to Kindrick [21], a conformance test suite is a “carefully constructed set of tests designed to maximize coverage of the most significant inputs,” where each test case “specifies purpose, operating conditions, inputs, and expected outputs.”

Both the NIST SP 800-53 security controls catalog reference data and the PMI test suite play important roles in systems integration and interoperability. As such, both datasets are part of the often-hidden infrastructure relied upon by systems that read and write master or transactional data. The Security Content Automation Protocol (SCAP), a collection of XML specifications standardizing the exchange of software flaw and security configuration information, includes mappings between Windows 7 system settings and the NIST SP 800-53 security controls. SCAP-conforming tools use these mappings to monitor and verify a system's compliance with an organization's security policies [22].

Data exchange standards for PMI define a machine-readable PMI syntax, but they do not provide a machine-readable representation of PMI semantics. PMI semantics are defined using natural language text and pictures such as the flatness tolerance definition shown in Figure 1. However, reliable and high-fidelity CAD data exchange requires not only that CAD software applications interpret PMI syntax, but also that their algorithms correctly implement PMI semantics [23]. Each test case in the PMI conformance test suite illustrates correct usage of a PMI concept as specified in the ASME standards. Therefore, by reproducing a test case using a CAD software application's authoring environment and comparing the result to the original, a conformance testing system can measure how well the software captures the syntax and semantics of that test case's PMI concepts.

Acknowledgments

I am grateful to Tom Hedberg, Harold Booth, and KC Morris for their meticulous and helpful reviews of earlier drafts of this paper. I also wish to thank the Balisage peer reviewers for their astute comments. Any remaining mistakes are my sole responsibility. Finally, I acknowledge my “PMI Validation and Conformance Testing”[3] colleagues —

particularly Bryan Fischer (the “PMI subject matter expert” mentioned in the section “PMI Conformance Test Suite”) — for many insightful discussions about PMI and PMI test case metadata requirements. These discussions were the genesis of my idea of using XForms to implement user interfaces for SANDs.

References

[1] *XForms 1.1*, W3C Recommendation, 20-Oct-2009. http://www.w3.org/TR/xforms.

[2] G. E. Krasner and S. T. Pope, *A description of the model-view-controller user interface paradigm in the smalltalk-80 system*, Journal of object-oriented programming, vol. 1.3, 1988.

[3] MBE PMI Validation and Conformance Testing, National Institute of Standards and Technology. [Online]. Available: http://www.nist.gov/el/msid/infotest/mbe-pmi-validation.cfm.

[4] Joint Task Force Transformation Initiative, *Security and Privacy Controls for Federal Information Systems and Organizations*, NIST Special Publication 800-53, Revision 4, April 2013, `10.6028/NIST.SP.800-53r4`.

[5] *OMG Unified Modeling Language (OMG UML)*, Version 2.4.1, 2011.

[6] S.P. Frechette, A.T. Jones, B.R. Fischer, *Strategy for Testing Conformance to Geometric Dimensioning & Tolerancing Standards*, Procedia CIRP, Volume 10, 2013, Pages 211-215, ISSN 2212-8271, `10.1016/j.procir.2013.08.033`.

[7] ASME Y14.5-2009 *Dimensioning and Tolerancing*, American Society of Mechanical Engineers.

[8] ASME Y14.41-2012, *Digital Product Definition Data Practices*, American Society of Mechanical Engineers.

[9] *The Transition from 2D Drafting to 3D Modeling Benchmark Report*, Aberdeen Group, 2006. Available: http://images.autodesk.com/adsk/files/transition_from_2d_to_3d_modeling_report.pdf.

[10] ISO 1101:2012, *Geometrical product specifications (GPS) — Geometrical tolerancing — Tolerances of form, orientation, location and run-out*, International Organization for Standardization.

[11] S. Pemberton, *XForms for HTML Authors, Part 2*, 27-Aug-2010. [Online]. Available: http://www.w3.org/MarkUp/Forms/2010/xforms11-for-html-authors/part2.html.

[12] XSLTForms - agenceXML. [Online]. Available: http://www.agencexml.com/xsltforms.

[13] *XSL Transformations (XSLT) Version 1.0*, W3C Recommendation, 16 November 1999. http://www.w3.org/TR/xslt.

[14] XForms 1.1 and XHTML+XForms1.1 RelaxNG Schemas from Klotz, Leigh on 2010-01-15 (public-forms@w3.org from January 2010). [Online]. Available: http://lists.w3.org/Archives/Public/public-forms/2010Jan/0019.html.

[15] Keith Stouffer, Suzanne Lightman, Victoria Pillitteri, Marshall Abrams, Adam Hahn,

Guide to Industrial Control Systems (ICS) Security, NIST Special Publication 800-82, Revision 2 Initial Public Draft, May 2014. http://csrc.nist.gov/publications/PubsDrafts.html#SP-800-82-Rev.2.

[16] National Vulnerability Database. [Online]. Available: http://nvd.nist.gov.

[17] NIST Special Publication 800-53. [Online]. Available: http://web.nvd.nist.gov/view/800-53/home.

[18] Cryptsoft: SP 800-53. [Online]. Available: http://www.cryptsoft.com/sp800-53.

[19] M. Pohja, *Comparison of common XML-based web user interface languages*, J. Web Eng., vol. 9, no. 2, pp. 95–115, 2010.

[20] D. McGilvray, *Executing data quality projects: Ten steps to quality data and trusted information*, Morgan Kaufmann, 2010.

[21] J. D. Kindrick, J. A. Sauter, and R. S. Matthews, *Improving conformance and interoperability testing*, StandardView, vol. 4, pp. 61–68, 1996, `10.1145/230871.230883`.

[22] Stephen Quinn, Karen Scarfone, David Waltermire, *Guide to Adopting and Using the Security Content Automation Protocol (SCAP) Version 1.2 (Draft)*, NIST Special Publication 800-117, Revision 1 (Draft), January 2012. http://csrc.nist.gov/publications/PubsDrafts.html#SP-800-117-Rev.%201.

[23] A. Barnard Feeney, S. P. Frechette, and V. Srinivasan, *A portrait of an ISO STEP tolerancing standard as an enabler of smart manufacturing systems*, 13th CIRP Conference on Computer Aided Tolerancing, Hangzhou, China, May 2014.

---

[*] An ATC's PMI specification corresponds to a particular annotation from the ATC's CTC. Therefore, the PMI specification's dimensions use the same units as those specified by the CTC's `units` attribute.

[†] Mention of third-party or commercial products or services in this paper does not imply approval or endorsement by the National Institute of Standards and Technology, nor does it imply that such products or services are necessarily the best available for the purpose.