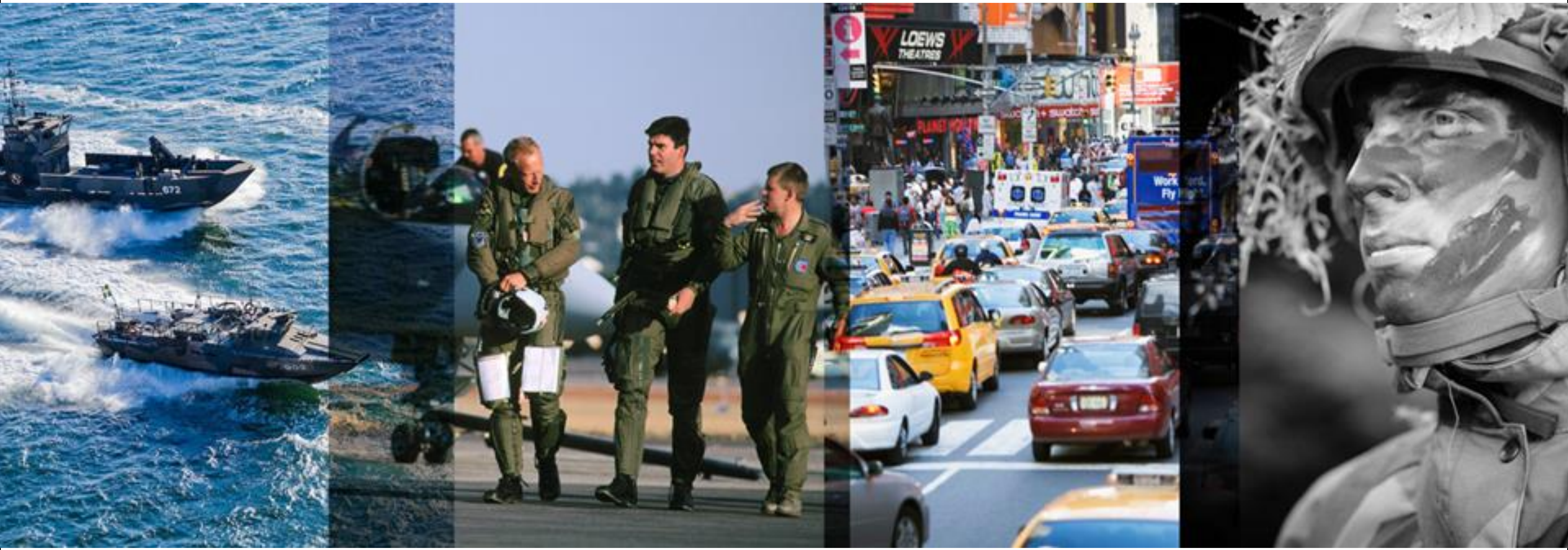
# BREX
# Maintaining Technical Publication Quality



Mårten Szymanowski

September 17 2013

S1000D User Forum 2013, Vienna

# AGENDA

## BACKGROUND

- New project → Develop IETP
- Legacy data:
  - Paper-oriented publications in SGML and PDF
- Problem:
  - IETP Neutral delivery not readily importable
- Need to detect all critical instances that prevents import and correct IETP-functionality, e.g.:
  - Hard-coded applicability
  - Hard-coded references
  - Unreferenced figures, tables
  - SGML to XML conversion issues (XML is case sensitivity, SGML not)
  - No titles in figures, tables

## BACKGROUND cont.

- Desire to increase automatic quality assurance
- Gradual streamlining of data in accordance with our publication standard
- Solution
  - BREX-based validation

## BUSINESS RULES EXCHANGE (BREX)

- What?
  - BREX is a construct that enables
    - the **communication** of Business Rules between projects,
    - facilitating **interchangeability** and **reuse**
  - BREX has its own Data Module Type
  - BREX can be used to **prevent incorrect data to enter into a data module**
    - use a validation-engine that can interpret and apply the rules contained in the BREX-file.
- Mechanisms:
  - Xpath
    - Syntax for defining parts of an XML document, and
    - Using path expressions to navigate in the document
  - Regular expressions (RegExp)
    - A sequence of characters that forms a search pattern
    - Many programming languages provide RegExp capabilities

# A MINIMAL BREX-FILE

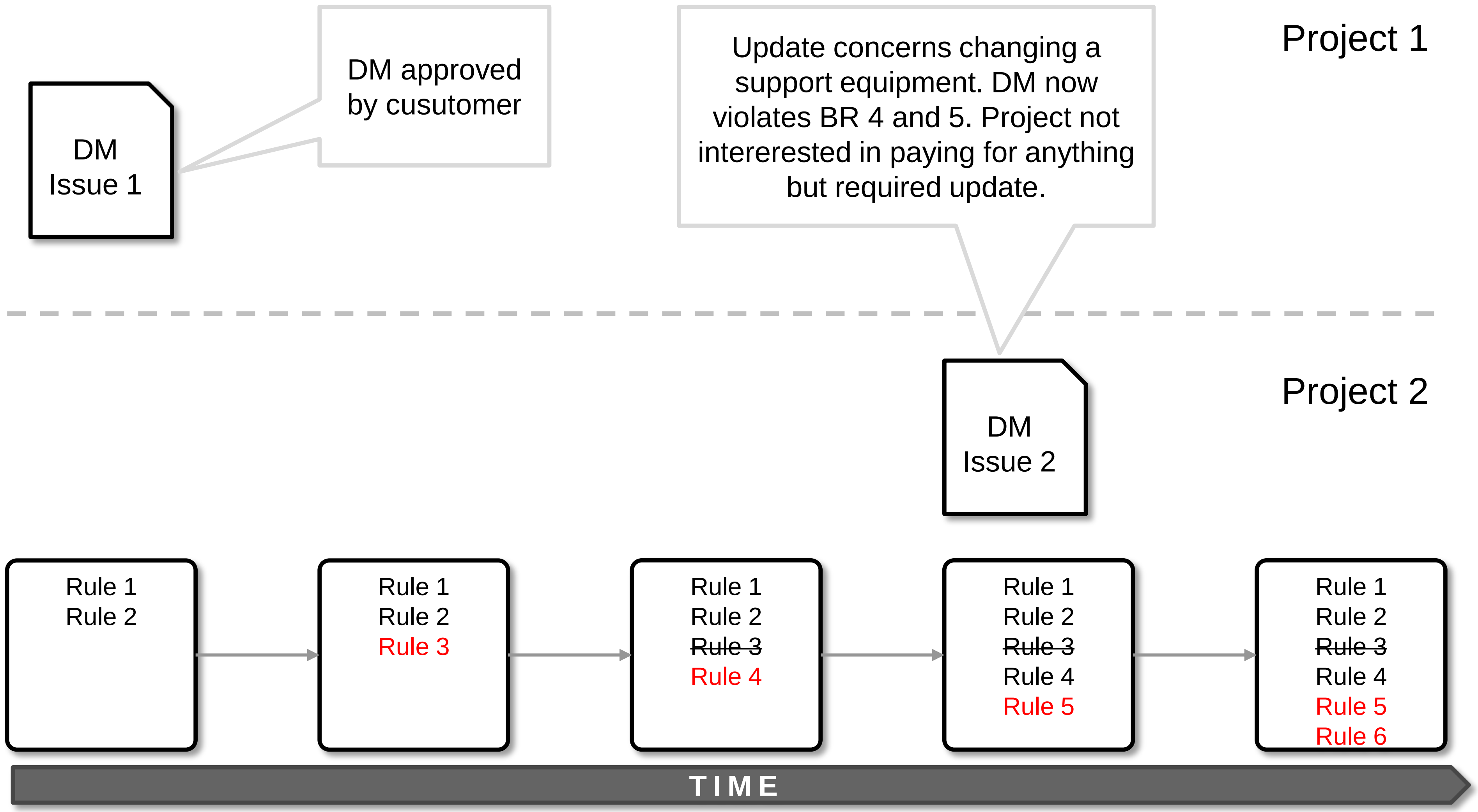
BREX S1000D 4.0.1

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE dmodule []>
<dmodule xsi:noNamespaceSchemaLocation="http://www.s1000d.org/S1000D_4-0-
1/xml_schema_flat/brex.xsd" xmlns:dc="http://www.purl.org/dc/elements/1.1/"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:xlink="http://www.w3.org/1999/xlink"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
<identAndStatusSection></identAndStatusSection>
  <content>
    <brex>
      <contextRules>
        <structureObjectRuleGroup>
          <structureObjectRule>
            <objectPath allowedObjectFlag="0">
              //xref[(not(attribute::xrefid = //*/@id))]  <!-- RULE -->
            </objectPath>
            <objectUse>
              Reference to undefined object.              <!-- DESCRIPTION -->
            </objectUse>
          </structureObjectRule>
        </structureObjectRuleGroup>
      </contextRules>
    </brex>
  </content>
</dmodule>
```

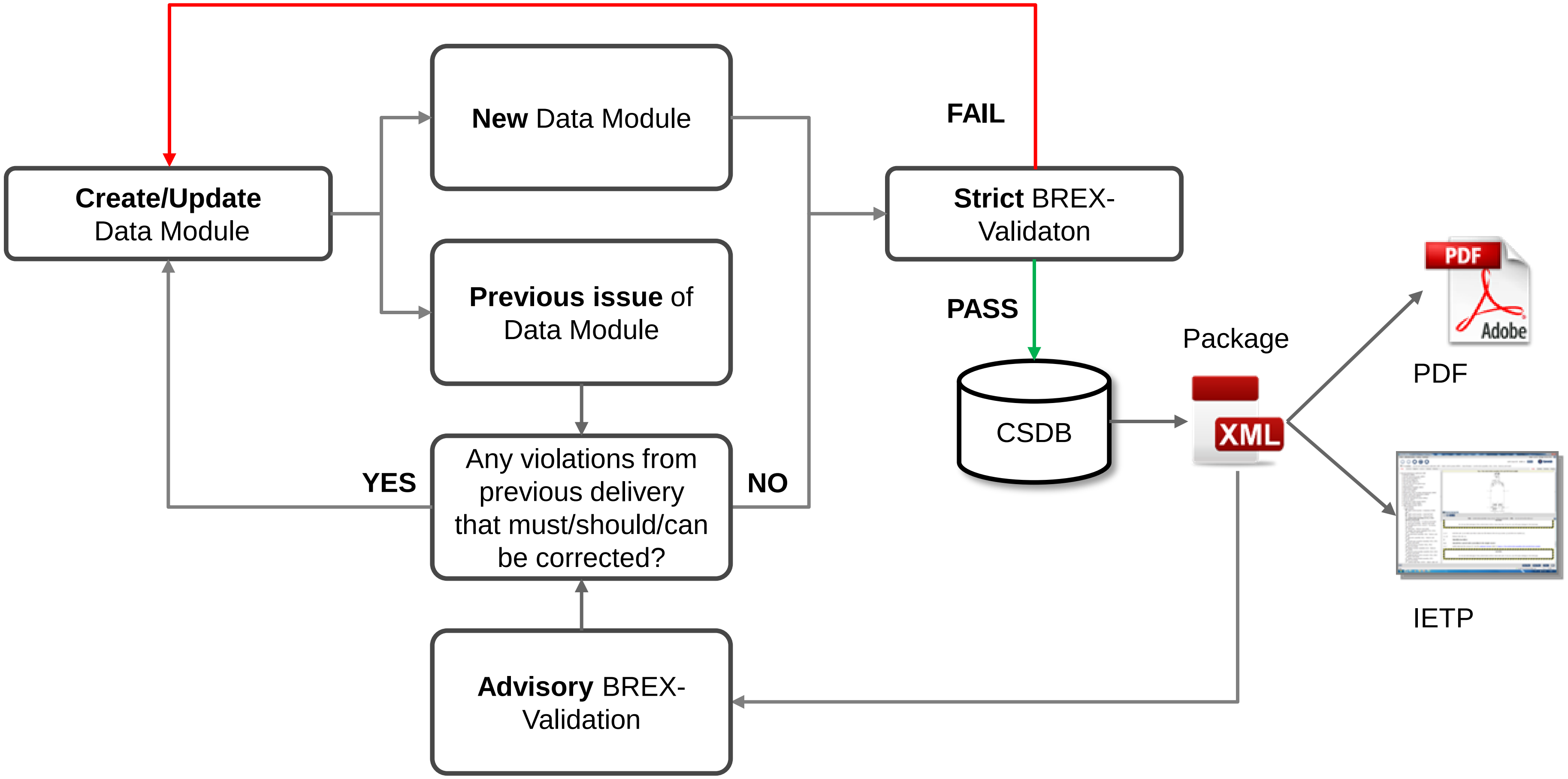
"0" = Not allowed
"1" = Allowed

For each `xrefid` attribute there must exist an `id` with the same value

# LIFE-CYCLE OF A DM


Project 1
DM Issue 1
DM approved by cusutomer
Update concerns changing a support equipment. DM now violates BR 4 and 5. Project not intererested in paying for anything but required update.
Project 2
DM Issue 2
Rule 1 Rule 2
Rule 1 Rule 2 Rule 3
Rule 1 Rule 2 ~~Rule 3~~ Rule 4
Rule 1 Rule 2 ~~Rule 3~~ Rule 4 Rule 5
Rule 1 Rule 2 ~~Rule 3~~ Rule 4 Rule 5 Rule 6
TIME

# BREX-VALIDATION WORKFLOW



Create/Update Data Module
New Data Module
Previous issue of Data Module
Any violations from previous delivery that must/should/can be corrected?
YES
NO
Advisory BREX-Validation
FAIL
Strict BREX-Validaton
PASS
CSDB
Package
XML
PDF
Adobe
PDF
IETP

# BREX-VALIDATION TOOL

- Continous Quality Improvement
  - Summary of **all violations** in delivery package
  - Comparison of amount and severity level of violations **between** delivery packages
- Technical authors can check the ”backlog” of a Data Module
  - List all (or a filtered subset) of the files that violate **any rule**
- Developers can check the impact of each rule and tweak it if neccesary
  - List all (or a filtered subset) of the files that violate a **specific rule**
- Pinpoint where in the Data Module the violation occured
  - Present the **context** in which the violation occured

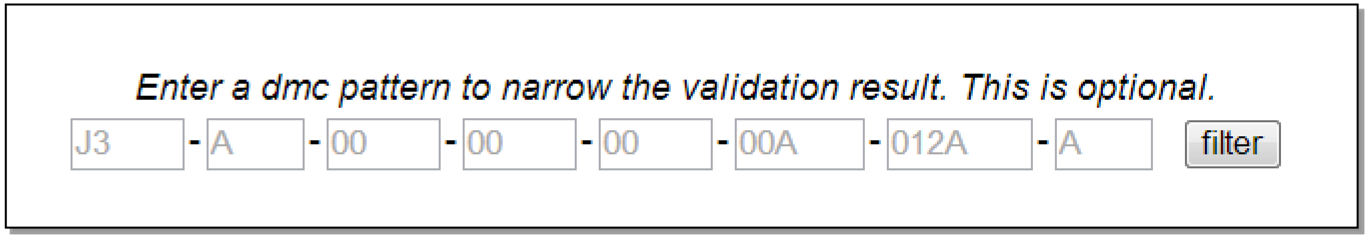
# COMPARISONS BETWEEN DELIVERIES

- Summary of all violations in a delivery package
- Mainly used for continous quality improvement

| CRITICALS | WARNINGS | NOTICES | BREX FILE | PACKAGE FILE | DATE |
|---|---|---|---|---|---|
| 871 | 929 | 3008 | BREX-201202261644.XML | IETP_Neutr_SZ002_01-Mar-2013.zip | 2013-03-03 15:48 |
| 319 | 675 | 2178 | BREX-201202261644.XML | IETP_Neutr_SZ002_02-Sep-2013.zip | 2013-09-06 15:33 |

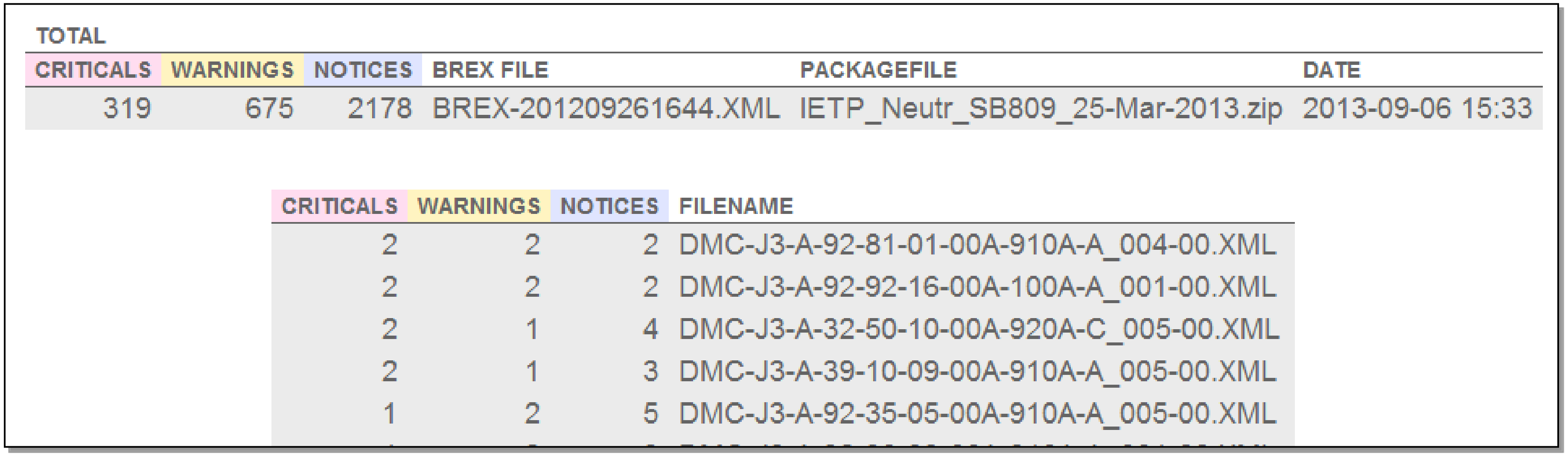
# VIOLATIONS PER DATA MODULE

- List all (or a filtered subset) of the files that violate **any rule**
- Mainly used by technical authors



*Enter a dmc pattern to narrow the validation result. This is optional.*

J3 - A - 00 - 00 - 00 - 00A - 012A - A  filter



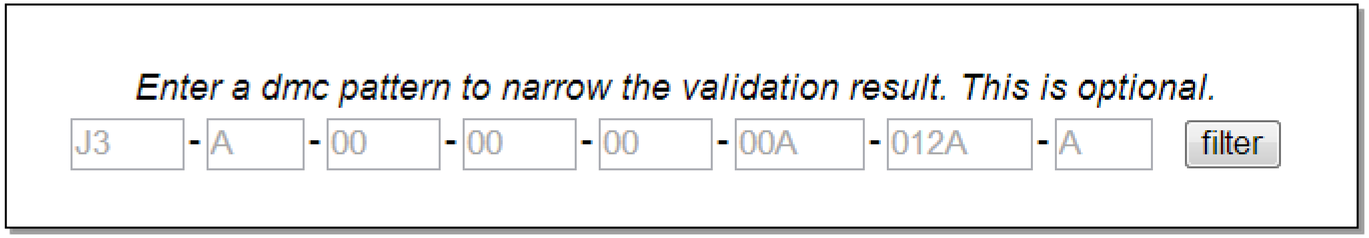
TOTAL

| CRITICALS | WARNINGS | NOTICES | BREX FILE | PACKAGEFILE | DATE |
|---|---|---|---|---|---|
| 319 | 675 | 2178 | BREX-201209261644.XML | IETP_Neutr_SB809_25-Mar-2013.zip | 2013-09-06 15:33 |

| CRITICALS | WARNINGS | NOTICES | FILENAME |
|---|---|---|---|
| 2 | 2 | 2 | DMC-J3-A-92-81-01-00A-910A-A_004-00.XML |
| 2 | 2 | 2 | DMC-J3-A-92-92-16-00A-100A-A_001-00.XML |
| 2 | 1 | 4 | DMC-J3-A-32-50-10-00A-920A-C_005-00.XML |
| 2 | 1 | 3 | DMC-J3-A-39-10-09-00A-910A-A_005-00.XML |
| 1 | 2 | 5 | DMC-J3-A-92-35-05-00A-910A-A_005-00.XML |

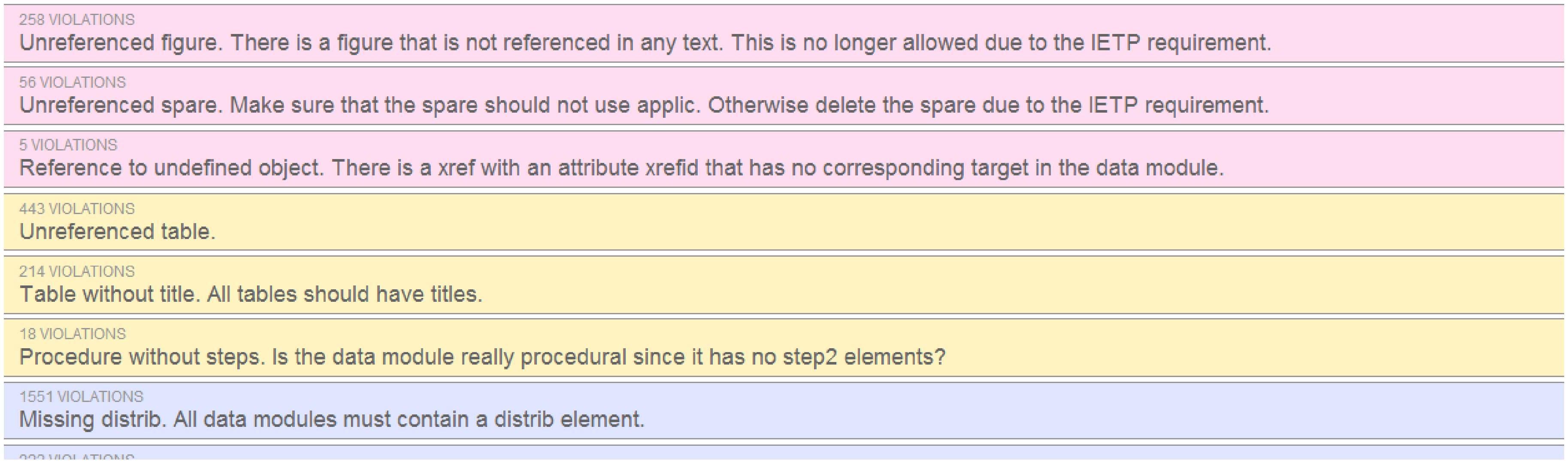
# VIOLATIONS PER RULE

- List all (or a filtered subset) of the files that violate a **specific rule**
- Mainly used when tweaking/developing the rules



*Enter a dmc pattern to narrow the validation result. This is optional.*

J3 - A - 00 - 00 - 00 - 00A - 012A - A filter



258 VIOLATIONS
Unreferenced figure. There is a figure that is not referenced in any text. This is no longer allowed due to the IETP requirement.

56 VIOLATIONS
Unreferenced spare. Make sure that the spare should not use applic. Otherwise delete the spare due to the IETP requirement.

5 VIOLATIONS
Reference to undefined object. There is a xref with an attribute xrefid that has no corresponding target in the data module.

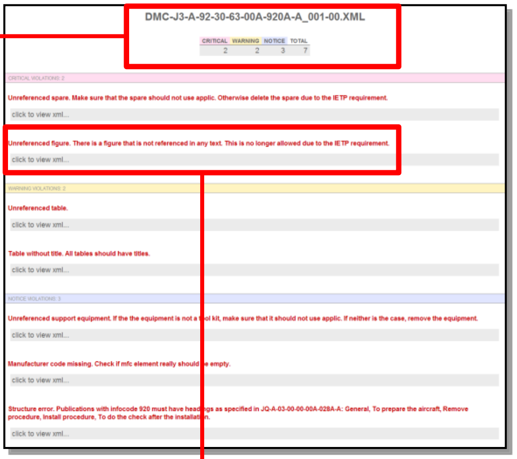
443 VIOLATIONS
Unreferenced table.

214 VIOLATIONS
Table without title. All tables should have titles.

18 VIOLATIONS
Procedure without steps. Is the data module really procedural since it has no step2 elements?

1551 VIOLATIONS
Missing distrib. All data modules must contain a distrib element.

# EXPLORE EACH DATA MODULE



DMC-J3-A-92-30-63-00A-920A-A_001-00.XML

| CRITICAL | WARNING | NOTICE | TOTAL |
|---|---|---|---|
| 2 | 2 | 3 | 7 |

CRITICAL VIOLATIONS: 2

Unreferenced spare. Make sure that the spare should not use applic. Otherwise delete the spare due to the IETP requirement.

click to view xml...

Unreferenced figure. There is a figure that is not referenced in any text. This is no longer allowed due to the IETP requirement.

click to view xml...

WARNING VIOLATIONS: 2

Unreferenced table.

click to view xml...

Table without title. All tables should have titles.

click to view xml...

NOTICE VIOLATIONS: 3

Unreferenced support equipment. If the the equipment is not a tool kit, make sure that it should not use applic. If neither is the case, remove the equipment.

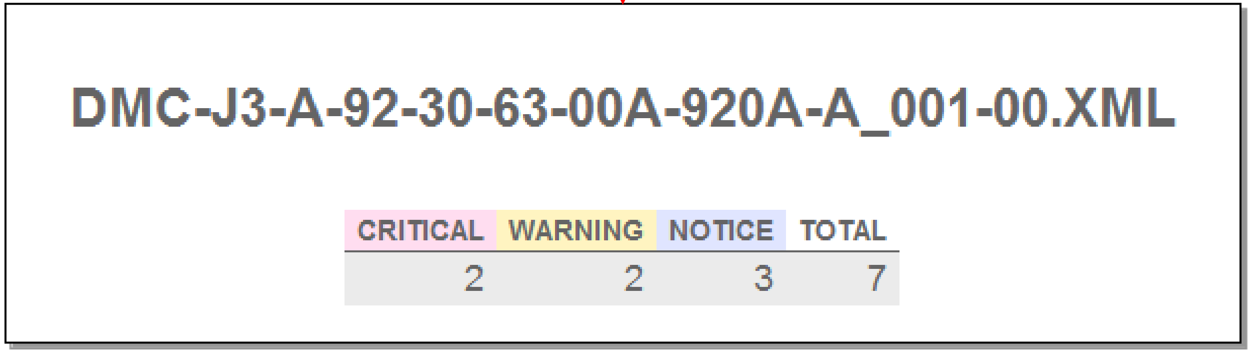
click to view xml...

Manufacturer code missing. Check if mfc element really should be empty.

click to view xml...

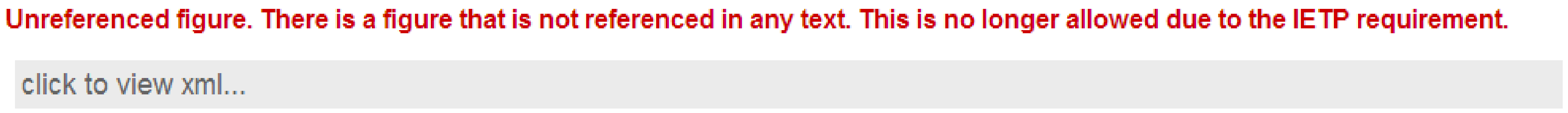
Structure error. Publications with infocode 920 must have headings as specified in JQ-A-03-00-00-00A-028A-A: General, To prepare the aircraft, Remove procedure, Install procedure, To do the check after the installation.

click to view xml...



DMC-J3-A-92-30-63-00A-920A-A_001-00.XML

| CRITICAL | WARNING | NOTICE | TOTAL |
|---|---|---|---|
| 2 | 2 | 3 | 7 |



Unreferenced figure. There is a figure that is not referenced in any text. This is no longer allowed due to the IETP requirement.

click to view xml...

## IMPLEMENTATION DETAILS

- Upload
  - Delivery packages (XML)
  - BREX-files (XML)
- Validate
  - Any BREX-file against any package
- Store validation results
  - BREX File 1 + Package A
  - BREX File 2 + Package A
  - BREX File 1 + Package B
  - Etc.
- Differentiate between different severity levels:
  - 1: CRITICAL, 2: WARNING, 3: NOTICE
  - Maintaining S1000D 4.0.1 schema, (mis-)using attribute `caveat`
- Explore violations per:
  - Package
  - Data Module
  - Rule

# IMPLEMENTATION DETAILS cont.

- Severity levels
  - cv51 = 1 - Critical
  - cv52 = 2 - Warning
  - cv53 = 3 - Notice

```
</structureObjectRule caveat="cv51">
  <objectPath allowedObjectFlag="0">
    //xref[(not(attribute::xrefid = //*/@id))] <!-- RULE -->
  </objectPath>
  <objectUse>
    Reference to undefined object.           <!-- DESCRIPTION -->
  </objectUse>
</structureObjectRule>
```

NOTE: This is **not** the *intended* usage of the attribute `caveat`

# AN EXTERNAL ADVISORY BREX: WHY?

- Not all rules are binary
  - Sometimes we just want a technical writer to make sure that their case is not the exception
    - E.g. support equipment should normally always be referenced in the body of the data module. However tool sets are an exception. They are only included under required conditions.
- Readily implementable
  - A rudimentary system can be developed with minimal knowledge in XML, S1000D and programming (trust us on this one)
  - Most languages support XML parsing, XPath and regular expressions using standard libraries
- Data agnostic
  - Can validate any desired XML within the same framework (i.e. not S1000D specific)
- Easily extendable
  - Batch processing can easily be added
  - Graphing functionality is a few lines of code away
  - Free to ”misuse” S1000D for needed functionality (e.g. severity levels)
- System independable
  - With monolithic approaches to a technical publication production system (does-it-all-platform) you are entirely in the hands of your system developer of choice

# KEYPOINTS

- All business rules decisions are **not** made before production of data modules starts
  - BREX rules evolve over time
- Projects are rarely interested in financing complete rewriting of previously approved data modules on the sole basis that they do not comply with new business rules decisions
  - S1000D experts have little room to navigate in the face of financial concerns
- All business rules are not equal when it comes to importance and criticality
- BREX with Severity Levels
  - Provide a potentially very useful QA metrics

SAAB

SAABGROUP.COM