

S1000D Applicability Model

Myths, Tips and Tricks and Gory Details

Jeroen van Rotterdam

General Manager XML Solutions
EMC Corporation

Agenda

S1000D Applicability Model

- Use Cases, why applicability
- Myths
- Gory details
- Conclusions
- Q&A

Myths

Myth:

- S1000D Applicability Model is extremely difficult

Reality:

- S1000D Applicability Model is very simple but can be used for very advanced applicability models

Myth:

- S1000D Applicability model is only for advanced documentation requirements

Reality:

- It can be deployed in very simple environments

Myth:

- iSpec 2200 and DITA can do this as well

Reality:

- Both applicability models are limited compared to S1000D



Use Cases

Why applicability ?

1. Content such as manuals, instructions or procedures shown to the end-users might be conditioned for:

- Models
- Serial numbers
- Configurations/Modifications of individual equipment
- Preferences of end-users
- Weather conditions
- Experience level of end-users
- Fault codes produced by the equipment
- Geographical location
- Type of location; e.g. hangar vs. line vs. shop

Prime goal of applicability:

- Show only the right content to the end user given its parameters.
- Allow for automatic processing of applicability definitions

End result

Serial number=1B070701

Serial number 1B070701

Attributes

Serial number : 1B070701
Model : Mountain storm
Version : Mk1
Version rank : 1

Handlebar

WARNING
DO NOT RIDE WITH A CRACKED STEM

If Stem cracked

- Replace stem

Else if Stem is loose

- Tighten stem

If Handlebars twist in stem

- Tighten clamp bolt

Computer

Challenge	Response
Computer Display	Mountain storm Mk1 ALTITUDE 0 miles SPEED 0 mph DISTANCE 0 miles

Serial number=1B070644

Serial number 1B070644

Attributes

Serial number : 1B070644
Model : Brook trekker
Version : Mk9
Version rank : 1

Handlebar

WARNING
DO NOT RIDE WITH A CRACKED STEM

If Stem cracked

- Replace stem

Else if Stem is loose

- Tighten stem

If Handlebars twist in stem

- Tighten clamp bolt

Computer

Challenge	Response
Computer Display	Brook trekker Mk9 SPEED 0 mph DISTANCE 0 miles

Same Document, Different content

The mechanic only sees relevant content for the serial number he/she is working on

Use Cases

Examples:

- This procedure only applies for tail number 1234, 1237 and 1339
- This document is only valid for model 747-400 post Service Bulletin SB001
- Show this paragraph when working on serial number S12345 under icy conditions
- This maintenance step is not valid for combat situations
- Only show this procedure when the mechanic is inexperienced
- Only destroy your credit card when your wife intends to go shopping
- Hide beer on Saturday and Sunday
- Don't show this graphic to end-users in Russia

(Not a realistic) Alternative:

- Maintain copies for each permutation of conditions
 - Maintenance nightmare
- State the conditions in the text
 - Hard to read for technicians
 - Error prone

S1000D approach

S1000D defines applicability in 3 basic steps

1. Declare conditions or values that *are allowed* in applicability statements in the content
2. Define the actual conditions in the content (applicability annotations) that use the declarations.
3. Filter the content for applicability

Example declaration of a property:

- name: "model"
- Description: "Model of a bicycle"
- Values:
 - Brook Trekker
 - Mountain Storm

Why do we need declarations ?

- We don't !
- They are handy to validate applicability conditions, aid the author


Example

Simple example:

```
<dmodule>
  <identAndStatusSection>
    <dmstatus>

      <applic>
        <assert applicPropertyIdent="model" applicPropertyValues="Mountain storm" />
      </applic>

    </dmstatus>
  </identAndStatusSection>
  <content>...</content>
</dmodule>
```



The diagram shows two callout boxes pointing to the XML code. The first callout box, labeled "Property", points to the value "model" in the `applicPropertyIdent` attribute. The second callout box, labeled "Value", points to the value "Mountain storm" in the `applicPropertyValues` attribute. Both callout boxes contain the text "Note: can be anything".

→ Document is only valid for **model** “**Mountain storm**”

- Single Assert
- Global applicability: applies to the entire Data Module; defined at the document level

Adding human readable explanation

Simple example:

```
<dmodule>
  <identAndStatusSection>
    <dmstatus>
      <applic>
        <displayText>
          <simplePara>For model Mountain storm only</simplePara>
        </displayText>
        <assert applicPropertyIdent="model" applicPropertyValues="Mountain storm"/>
      </applic>
    </dmstatus>
  </identAndStatusSection>
  <content>...</content>
</dmodule>
```

→ Enhanced applic statement with optional display text for human readability

Warning

Author is responsible for keeping the display text in sync with the assert statement

Using multiple values

Simple example:

```
<dmodule>
  <identAndStatusSection>
    <dmstatus>

      <applic>

        <displayText>
          <simplePara>For model Mountain storm or Brook trekker</simplePara>
        </displayText>

        <assert applicPropertyIdent="model" applicPropertyValues="Mountain storm | Brook trekker"/>
      </applic>

    </dmstatus>
  </identAndStatusSection>
  <content>...</content>
</dmodule>
```

- Enhanced assertion with multiple values
- Using an “OR” operator: “|”

Adding multiple asserts

Multiple values don't work if you use different type of assertions.

Example: “for all mountain bikes but also for all ‘Brook trekker’ models”

```
<dmodule>
  <identAndStatusSection>
    <dmstatus>

      <applic>
        <displayText>
          <simplePara>All mountain bikes and models Brook trekker</simplePara>
        </displayText>

        <evaluate andOr="or">
          <assert applicPropertyIdent="type" applicPropertyValues="Mountain bike"/>
          <assert applicPropertyIdent="model" applicPropertyValues="Brook trekker"/>
        </evaluate>

      </applic>

    </dmstatus>
  </identAndStatusSection>
  <content>...</content>
</dmodule>
```

Nested evaluations

Evaluations can be nested to construct more advanced applicability

```
( Type == 'Mountain bicycle' )  
    AND  
(  
    ( Model == 'Mountain storm' AND Version == 'MK1' )  
      OR  
    ( Model == 'Brook trekker' AND Version == 'MK9' )  
)
```

```
<evaluate andOr="and">  
  <assert applicPropertyIdent="type" applicPropertyValues="Mountain bicycle"/>  
  <evaluate andOr="or">  
    <evaluate andOr="and">  
      <assert applicPropertyIdent="model" applicPropertyValues="Mountain storm"/>  
      <assert applicPropertyIdent="version" applicPropertyValues="Mk1"/>  
    </evaluate>  
    <evaluate andOr="and">  
      <assert applicPropertyIdent="model" applicPropertyValues="Brook trekker"/>  
      <assert applicPropertyIdent="version" applicPropertyValues="Mk9"/>  
    </evaluate>  
  </evaluate>  
</evaluate>
```

Inline applicability

So far we only saw “GLOBAL” applicability

→ Applicability defines whether the entire document applies

In a lot of cases it is more subtle:

→ Applicability defines when a piece of the document is relevant

This can be done by defining “INLINE” applicability

Inline applicability

Inline applicability is *defined* in the `inlineApplicGroup` and referred to in the content using a `applicRefId` attribute.

```
<dmodule>
  <identAndStatusSection>
    <dmstatus>
      <referencedApplicGroup>
        <applic id="myFirstApplicDefinition">
          <assert applicPropertyIdent="model" applicPropertyValues="Mountain storm"/>
        </applic>
      </referencedApplicGroup>
    </dmstatus>
  </identAndStatusSection>
  <content>
    ...
    <para applicRefId="myFirstApplicDefinition">This para has applicability</para>
    <para>This para is always visible
    <para applicRefId="myFirstApplicDefinition">This para has applicability</para>
    ...
  </content>
</dmodule>
```

Definition(s)

Reference

Reference

Applicability *Declarations*

Challenge:

1. How do you make sure that applicability values are consistently used ?
2. How do you know which values are available ?

There is a need to

- Aid the author by giving lookup dialogs
- Validate applicability annotation

Example

Every body knows this is incorrect:



Typo

```
<dmodule>
  <identAndStatusSection>
    <dmstatus>

      <applic>
        <assert applicPropertyIdent="model" applicPropertyValues="Mountaain storm" />
      </applic>

    </dmstatus>
  </identAndStatusSection>
  <content>...</content>
</dmodule>
```

But Authoring Tools and IETP's can't detect this !

→ Solution: Declare applicability properties

→ Can't be part of the schema since the values are too dynamic

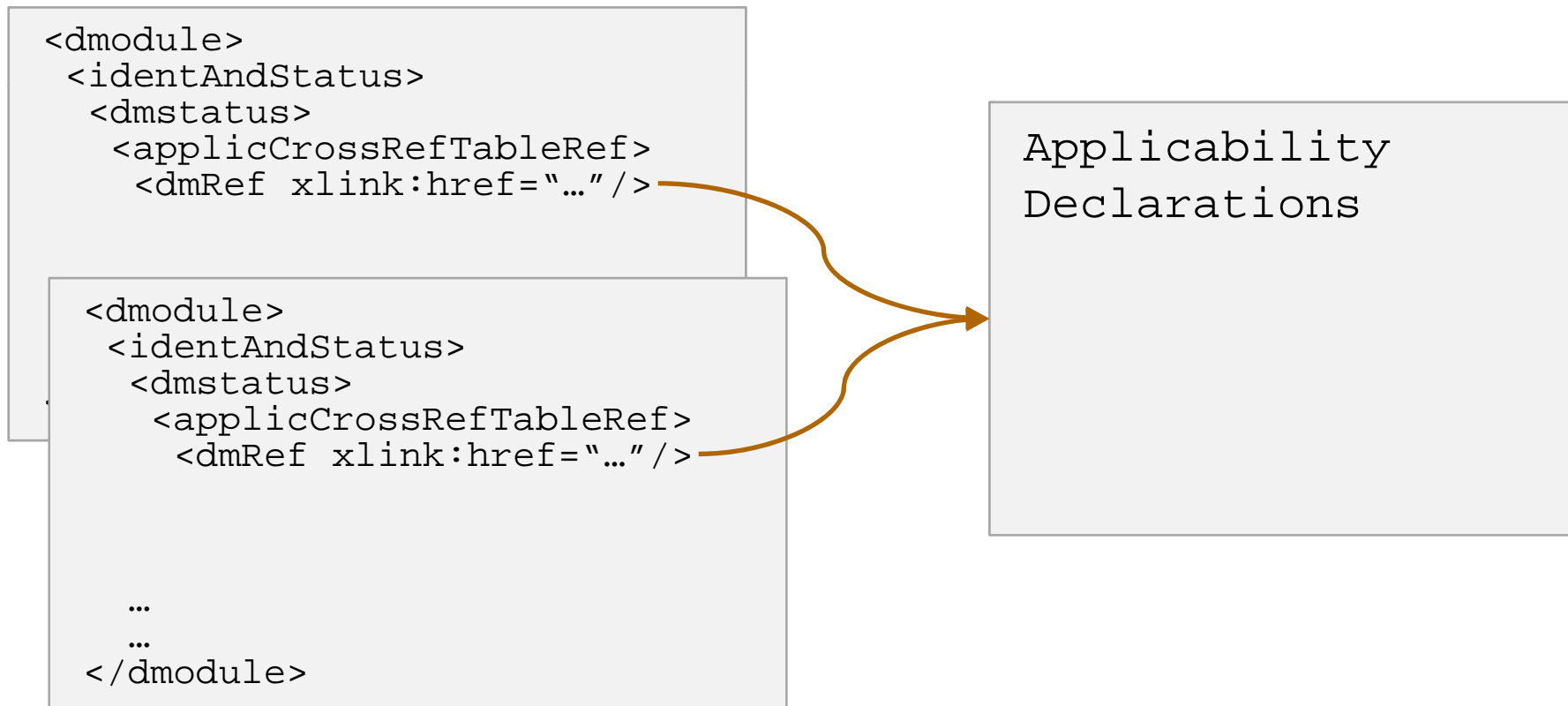
It's not just typo's, it is also discovery:

- Which serial numbers do we have
- Which Service Bulletins are there
- Which weather conditions can be used

Applicability Declarations

S1000D declares applicability properties in a separate data module

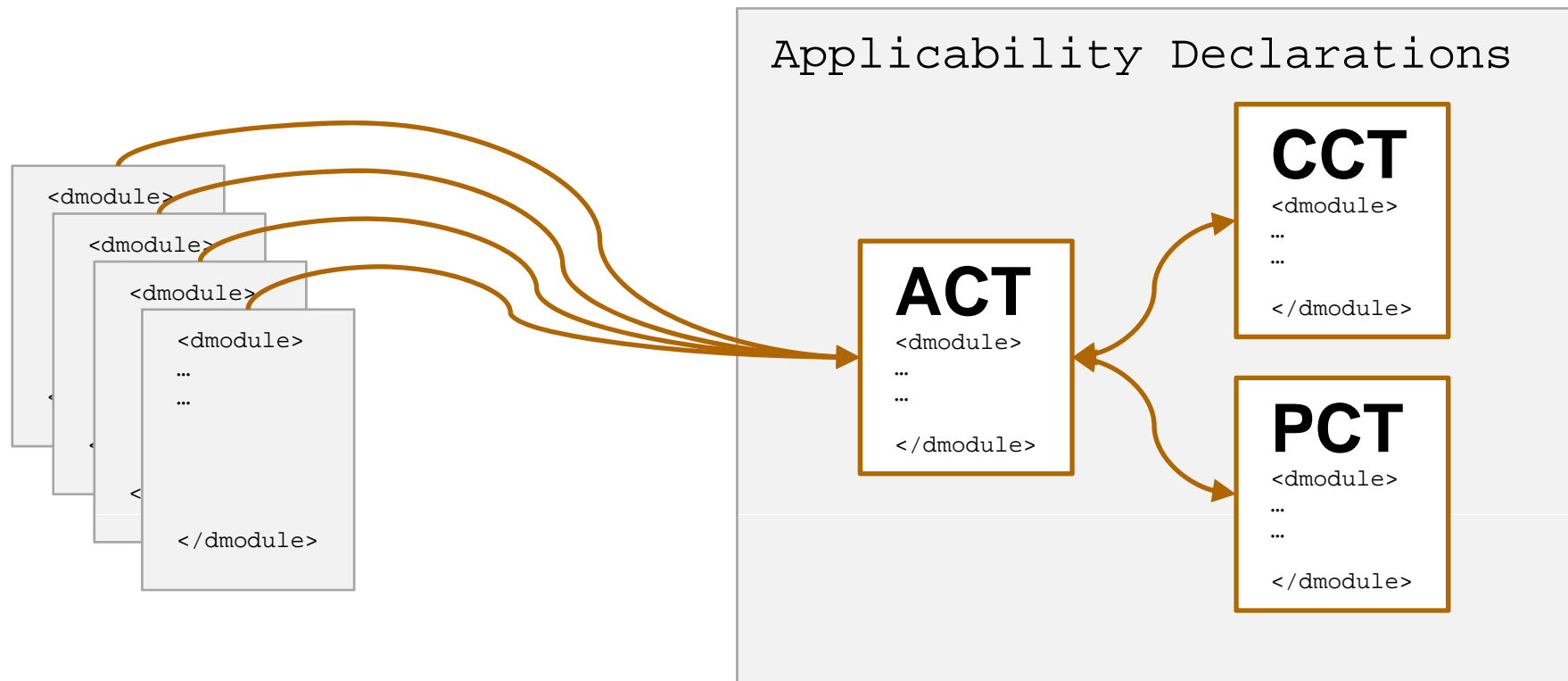
→ Multiple data modules can share applicability declarations



Applicability Declarations

Applicability is defined in 3 data modules:

- ACT: *Applicability* Cross-reference Table
- CCT: *Conditions* Cross-reference Table
- PCT: *Product* Cross-reference Table



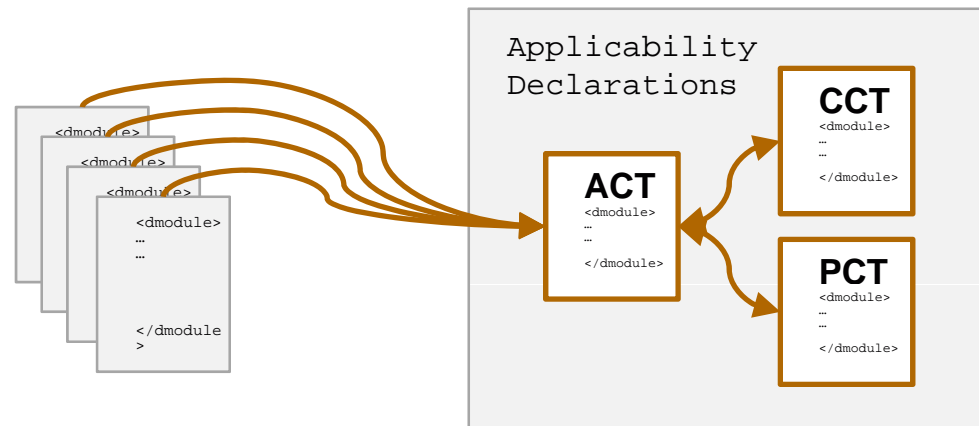
Applicability Declarations

Why in 3 Data Modules ??

1. It could have been done in one !
2. S1000D distinguishes 3 different type of declarations which are organized in different data modules

Declarations are found by references

- Uni-directional reference from a Data Module to an ACT
- Bi-directional reference between the ACT and the CCT
- Bi-directional reference between the ACT and the PCT



Applicability Declarations

ACT

Defines Product Attributes

- Typically set at manufacturing time

Examples:

- Serial Number
- Aircraft Tail Number
- License Plate
- Manufacturers Serial Number
- Windows XP Product Key
- Model
- Steering wheel position

CCT

Defines Conditions

- Technical
- Operational
- Environmental

Examples:

- Icy conditions yes or no
- Pre or post Service Bulletin SB1
- Day of the week: mo - su
- Combat situation yes or no
- Maintenance Hub
- Gate

PCT

Defines Product Instances

- More a product database
- Defines *attributes* and *conditions* for an actual instance

Examples:

Car 1

- License plate PR-XG-63
- Model: V40
- Year: 1996
- Manufacturer: Volvo

Car 2

- License plate AL-32-22
- Model: Daffodil
- Year: 1964
- Manufacturer: DAF



ACT: Defines Product Attributes

```
<dmodule>
  <content>
    <applicCrossRefTable>
      <productAttributeList>

        <productAttribute id="mfg">
          <name>Manufacturer</name>
          <descr>The name of the manufacturer</descr>
          <enumeration applicPropertyValues = "AIRBUS | BOEING | DOUGLAS | PRATT | GE | ROLLS | GOODRICH" />
        </productAttribute>

        <productAttribute id="tail">
          <name>Tail Number</name>
          <descr>Tail Number for an aircraft.</descr>
          <enumeration applicPropertyValues =
            "3101~3102 | 3114~3178 | 3201 | 3203~3205 | 3209~3245 |
            3247~3280 | 3301~3321 | 3351~3361 | 6622~6628 | 6631~6635" />
        </productAttribute>

      </productAttributeList>
    </applicCrossRefTable>
  </content>
</dmodule>
```

Identifies the product attribute
Referred to from the applicability annotation (in the content)

Series of ranges

ACT: Example with Global Applicability

Content

```

<dmodule>
  <identAndStatusSection>
    <dmstatus>

    <applic>
      <assert
        applicPropertyIdent="tail"
        applicPropertyType="prodattr"
        applicPropertyValues="3210" />
      </assert>
    </applic>

    </dmstatus>
  </identAndStatusSection>
  <content>...</content>
</dmodule>

```

Global Applicability

applicPropertyType:

- "prodattr" → look in the ACT
- "condition" → look in the CCT

ACT

```

<dmodule>
  <content>
    <applicCrossRefTable>
      <productAttributeList>
        <productAttribute id="mfg">
          <name>Manufacturer</name>
          <descr>The name of the manufacturer</descr>
          <enumeration applicPropertyValues =
            "AIRBUS|BOEING|DOUGLAS|
            PRATT|GE|ROLLS|GOODRICH" />
        </productAttribute>
        <productAttribute id="tail">
          <name>Tail Number</name>
          <descr>Tail Number for an aircraft.</descr>
          <enumeration applicPropertyValues =
            "3101~3102|3114~3178|3201|3203~3205|
            3209~3245|3247~3280|3301~3321|
            3351~3361|6622~6628|6631~6635" />
        </productAttribute>
      </productAttributeList>
    </applicCrossRefTable>
  </content>
</dmodule>

```

CCT:Conditions Example

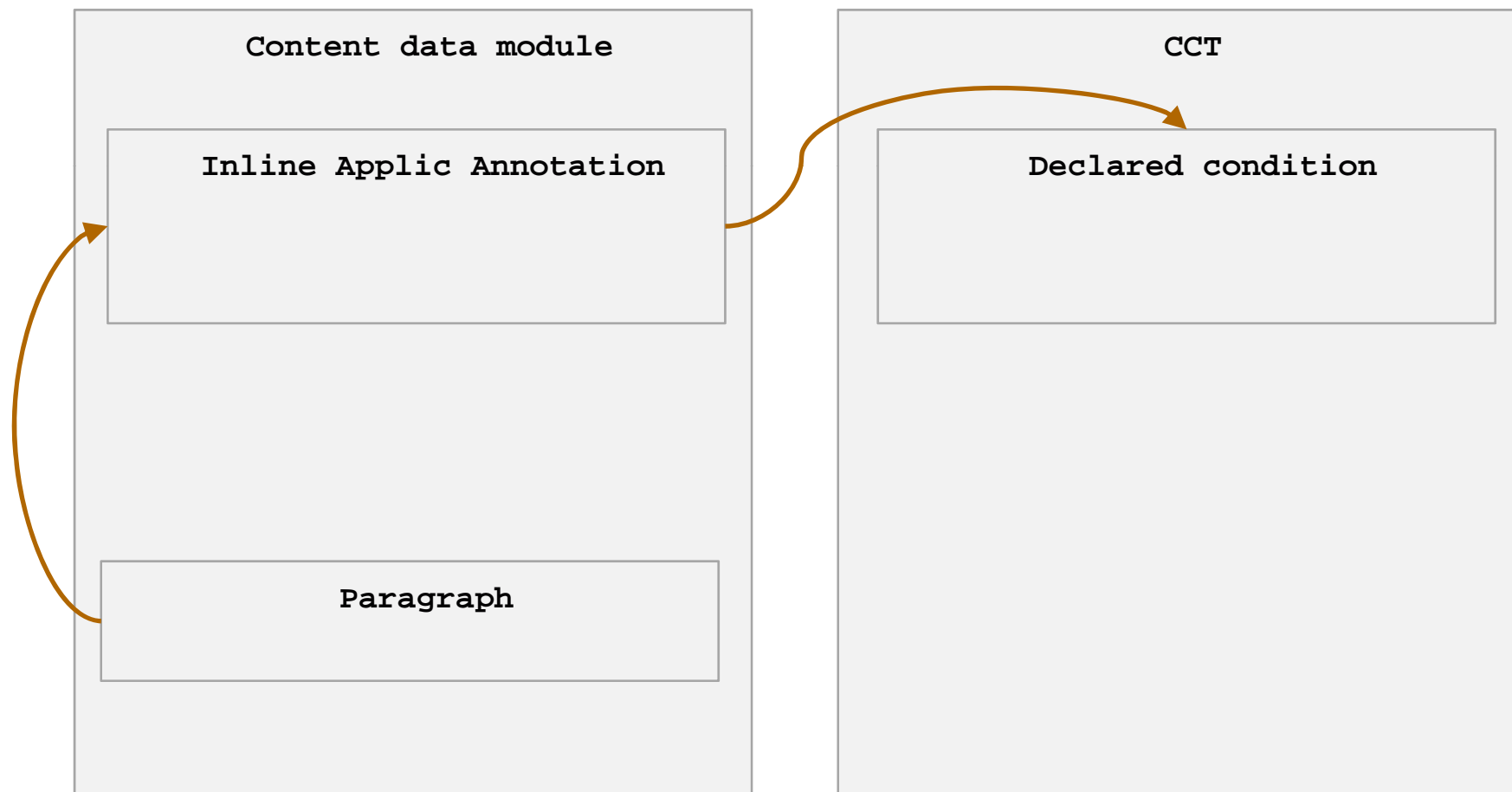
Lets assume we want to show the following paragraph

“De-ice the aircraft”

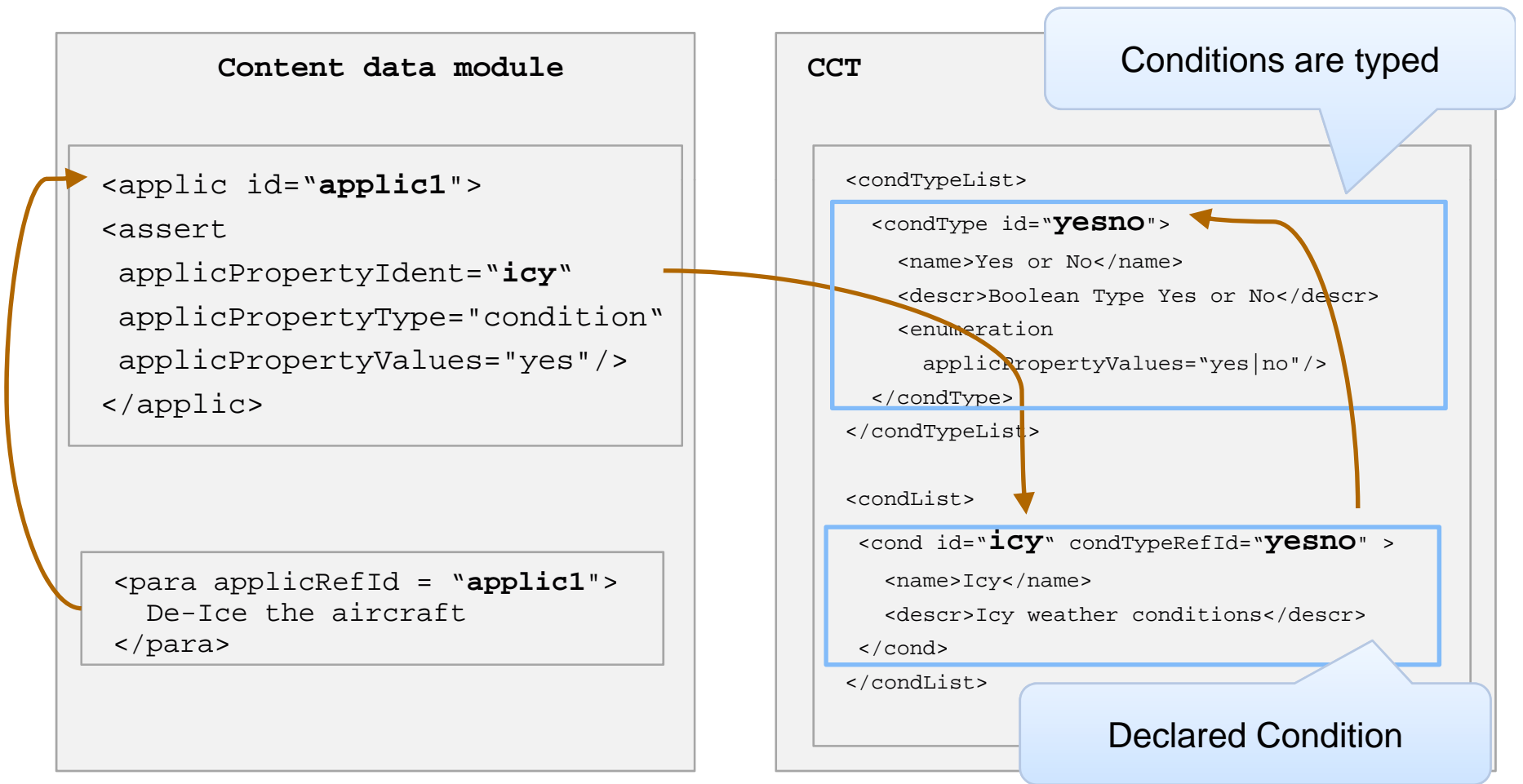
In the take-off procedure if the weather condition is “icy”

1. This applies to a paragraph only
→ we need inline applicability
2. This is a “condition”, we need to declare the weather condition in the CCT
3. We need to refer from the para to the inline applicability
4. We need to refer from the inline applicability to the CCT

CCT:Conditions Example



CCT:Conditions Example



PCT: Defines Products

Identifies a product

```
<dmodule>  
<content>  
<productCrossRefTable>
```

“prodattr”
→ Find the type in the ACT

```
<product>  
<assign applicPropertyIdent = "mfg"                applicPropertyType = "prodattr"  applicPropertyValue = "AIRBUS" />  
<assign applicPropertyIdent = "modelseries"        applicPropertyType = "prodattr"  applicPropertyValue = "A319/A320" />  
<assign applicPropertyIdent = "tail"               applicPropertyType = "prodattr"  applicPropertyValue = "3101" />  
<assign applicPropertyIdent = "serialno"           applicPropertyType = "prodattr"  applicPropertyValue = "1058" />  
<assign applicPropertyIdent = "registry"           applicPropertyType = "prodattr"  applicPropertyValue = "N301NB" />  
<assign applicPropertyIdent = "ETOPS-AC"           applicPropertyType = "condition" applicPropertyValue = "NON-ETOPS" />  
</product>
```

```
<product>  
<assign applicPropertyIdent = "mfg"                applicPropertyType = "prodattr"  applicPropertyValue = "BOEING" />  
<assign applicPropertyIdent = "modelseries"        applicPropertyType = "prodattr"  applicPropertyValue = "B757200" />  
<assign applicPropertyIdent = "tail"               applicPropertyType = "prodattr"  applicPropertyValue = "5517" />  
<assign applicPropertyIdent = "custeff"            applicPropertyType = "prodattr"  applicPropertyValue = "517" />  
<assign applicPropertyIdent = "ETOPS-AC"           applicPropertyType = "condition" applicPropertyValue = "NON-ETOPS" />  
</product>
```

```
</productCrossRefTable>  
</content>  
</dmodule>
```

Condition identifier

“condition”
→ Find the type in the CCT

Using the PCT

The PCT is the driver to filter content for the end-user in an IETP

1. The end-user selects a product from the PCT he/she is working on
2. The end-user might set environmental Conditions retrieved from the associated CCT
3. The content is now filtered for the Product Attributes (ACT) and the Conditions (CCT) defined in the product.

Warning

If the product doesn't define a Product Attribute or Condition then the content is not filtered.

Example:

Paragraph : Don't remove beer from fridge
Condition : Day of the week == 'Sunday ~ Friday'
Product : Husband

If you don't specify that the product "Husband" has a condition "Day of the week" then the beer will stay in the fridge even on Saturday because:

you will get the instruction: "Don't remove the beer from the fridge"

IETP example

AMDS-DS - S1000D - Set applicability - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:8181/amds-browser/viewDataModules.do

AMDS Delivery System
Currently logged in as Administrator in project DeliveryLive (logout)

Visits Workcards S1000D System Admin Preferences

Set applicability | Search Search results Browse Fault navigation Part ordering Bookmarks History Reset

Primary product identifier

Serial number *

Environmental conditions

Weather Conditions Snow

Location Hangar

Set

AMDS Delivery System 1.4.1 - X-Hive Corporation © 2004-2008 - report an error

Done

Single ACT in the project
no selection in required

Product Selector: PCT

Mandatory Conditions from the CCT

IETP Example

AMDS-DS - S1000D - Set applicability - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Product Attributes from the PCT

Product lookup

Serial number	Model	Version	Version rank	Customer effectivity
1B070643	Brook trekker	Mk9	2	1001
1B070644	Brook trekker	Mk9	1	1002
1B070701	Mountain storm	Mk1	1	001

Selecting a product from the PCT

Primary product id

Serial number *

Environmental conditions

Weather Conditions

Location

Select Cancel

Done

004-2008 - report an error

IETP Example

AMDS-DS - S1000D - Set applicability - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:8181/amds-browser/viewDataModules.do

AMDS Delivery System
Currently logged in as Administrator in project DeliveryLive (logout)

Visits Workcards S1000D System Admin Preferences

Set applicability | Search Search results Browse Fault navigation Part ordering Bookmarks History Reset

Set applicability

Primary product identifier

Serial number *

Environmental conditions

Weather Conditions

- Snow
- Snow
- Rain
- Sunny

Location

Set

AMDS Delivery System 1.4.1 - X-Hive Corporation © 2004-2008 - [report an error](#)

Done

Change my conditions defined in the CCT type definition

IETP Example

The screenshot displays the AMDS Delivery System web interface. The browser title is "AMDS-DS - S1000D - Browse S1000DBIKE-AAA-DA0: 8 - Mozilla Firefox". The address bar shows the URL: `http://localhost:8181/amds-browser/viewDataModules.do?dmc=S1000DBIKE_AAA_DA0&mode=html`. The page header includes "AMDS Delivery System" and "Currently logged in as Administrator in project DeliveryLive (logout)".

Key features and annotations:

- Filtered Search:** A yellow callout points to the search filters: "Serial number=1B070644 | Location=Hangar | Weather Conditions=Snow".
- Selected Product: PCT:** A yellow callout points to the breadcrumb "Browse S1000DBIKE-AAA-DA0: 8".
- Conditions Set:** A yellow callout points to the search filter "Weather Conditions=Snow".
- Filtered Browse:** A yellow callout points to the left-hand navigation tree, where the "DA0" folder is selected.
- Filtered Rendering:** A yellow callout points to the table of results, which shows only items relevant to the selected product and search conditions.

ATA	Title	Type	Reference number
<input type="checkbox"/> DA0-00	Wheel	Description of how it is made	S1000DBIKE-AAA-DA0-00-00-00AA-041A-A
<input type="checkbox"/> DA0-10	Inner tube	Remove and install a new item	S1000DBIKE-AAA-DA0-10-10-00AA-921A-A
<input type="checkbox"/> DA0-10	Tire	Fill with air	S1000DBIKE-AAA-DA0-10-20-00AA-215A-A
<input type="checkbox"/> DA0-10	Tire	Check pressure	S1000DBIKE-AAA-DA0-10-20-00AA-362B-A
<input type="checkbox"/> DA0-10	Front wheel	Fault reports and isolation procedures	S1000DBIKE-AAA-DA0-10-20-00AA-400A-A
<input type="checkbox"/> DA0-10	Tire	Remove and install a new item	S1000DBIKE-AAA-DA0-10-20-00AA-921A-A
<input type="checkbox"/> DA0-20	Rear wheel	Detected fault	S1000DBIKE-AAA-DA0-20-00-00AA-412A-A
<input type="checkbox"/> DA0-20	Rear wheel	Remove procedures	S1000DBIKE-AAA-DA0-20-00-00AA-520A-A

AMDS Delivery System 1.4.1 - X-Hive Corporation © 2004-2008 - report an error

Conclusions

S1000D applicability model defines:

1. Applicability declarations
 - Product attributes: ACT Data Module
 - Conditions: CCT Data Module
 - Products: PCT Data Module
2. Applicability Annotations
 1. Conditional content that refer to declarations
 2. From simple Asserts to Boolean expressions using Evaluate
3. The S1000D applicability model allows for content filtering based on product configurations as well as dynamic conditions.
4. Content filtering is transparent for the end-user. End-users will only see content that applies



Questions and Answers

Jeroen van Rotterdam
VanRotterdam_Jeroen@emc.com

www.emc.com
www.x-hive.com