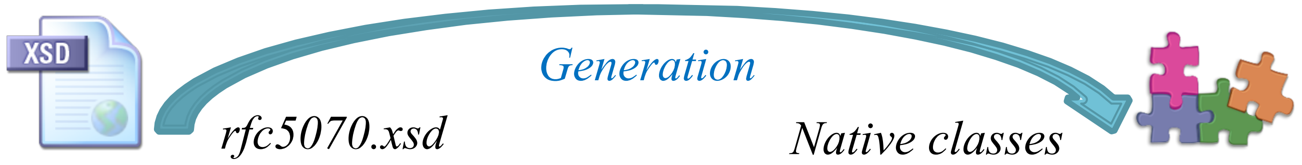
Knowledge obtained from the implementation experience of an IODEF-capable incident response management system

Daisuke Miyamoto, The University of Tokyo
Takeshi Takahashi, NICT
daisu-mi@nc.u-tokyo.ac.jp

Overview

- We share two types of implementation issues
  - We have been developing an IODEF-capable incident response management system
  - We encountered coding and usability issues
- Code-level issue
  - Writing XML document from scratch is cumbersome, and the use of class libraries will dramatically reduce programming costs



  - We thus want to generate class libraries for outputting IODEF document
  - However, some code generators could not work as expected
- Usability issue
  - There are IODEF fields where we couldn't find suitable values

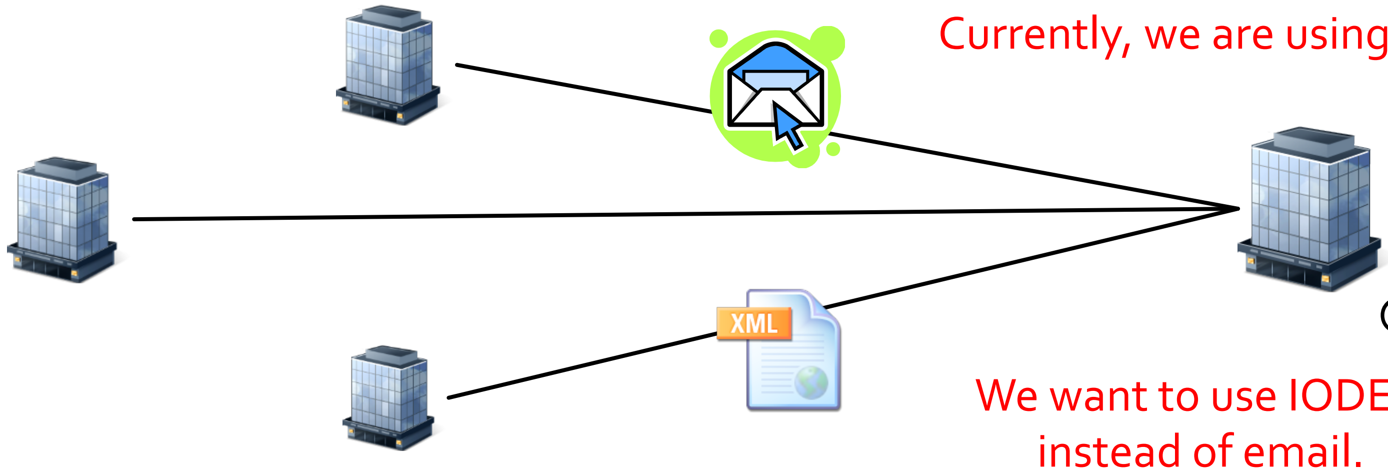
Background – our university's CERTs

DIVISIONAL CERT

- 30 or more CERT for each division
  - graduate school of economics, medicine, engineering ....
  - Having responsibility for incident containment

CENTRAL CERT

- One central CERT
  - Exchanging incident information from external CERT
  - Management of Incident Response



Currently, we are using email

Central CERT

We want to use IODEF, instead of email.

Summary of the findings

- #1: Complexity of XSD and its evasion
- #2: hyphen symbols
- #3: “type” attribute @ Impact Class
- #4 : “category” attribute @ NodeRole Class
- #5 : “action” attribute @ Expectation Class
- #6 : Potential information leakage
- #7 : Configuration of Nodes

#1 : Complexity of XSD …

- Trial of code generators

```
#!/usr/bin/perl
use XML::Pastor;
my $file = shift;
my $pastor = XML::Pastor->new();
$pastor->generate(
   mode =>'offline',
   style => 'single',
   schema=>$file,
   class_prefix=>'IODEF::',
   destination=>'./',);
```

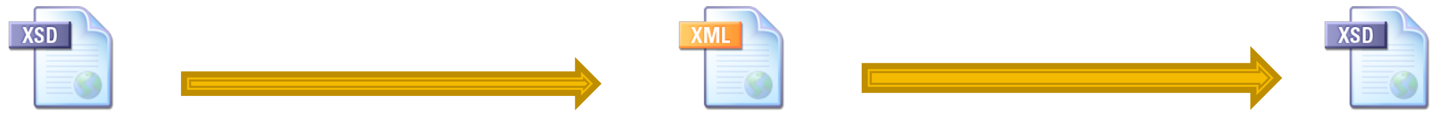
```
# perl xsd-generate.pl rfc5070.xsd
Pastor : Unexpected element 'any'
in schema!

<sequence >
  <any namespace="##any"
processContents="lax"
minOccurs="0"
maxOccurs="unbounded" />
</sequence>
```

- Result

Code generator	Result for RFC5070.xsd
XML::Pastor (perl)	Error
RXSD (ruby)	Error
PyXB (python)	OK
JAXB (Java)	Error
Codesynthesis XSD (c++)	OK
XSD.exe (c#)	OK

#1 : … and its evasions

- Evasions
  - Bad Know-How : XSD-to-XML-to-XSD (not recommend, but work)



Adding data to generate XML　　Converting from XML to XSD with XSD.exe

- Result of 2nd round

Code Generator	RFC5070.xsd	Converted XSD
XML::Pastor (perl)	Error	OK
RXSD (ruby)	Error	OK
PyXB (python)	OK	OK
JAXB (Java)	Error	OK
Codesynthesis XSD (c++)	OK	OK
XSD.exe (c#)	OK	OK

#2 : hyphen symbols

- Trying to generate class libraries
  - Problem: some programming languages prohibit the use of “hyphen” symbol for name of class libraries
  - Generated class libraries violate that
    - **`IODEF-Document`** class
      - the top level class of the IODEF data model
    - The **`vlan-name`** and **`vlan-num`** attribute
      - the name and number of Virtual LAN
      - the attributes for Address class
    - Extending enumerated values of attributes
      - has a prefix of "`ext-`“
        e.g., **`ext-value`**, **`ext-category`**, **`ext-type`**

#2 : hyphen symbols

- Evasion for XML::Pastor (perl) and RXSD (ruby)
  - Replacing hyphen with underscore
- Observation
  - During code generation, if hyphen symbols are …
    - remained; it does not work
    - replaced with _ (underscore), or removed ; it works
  - Some serialize functions support to restore hyphen symbols

Code Generator	RFC5070.xsd	Converted XSD	Hyphens
XML::Pastor (perl)	Error	OK	Remained
RXSD (ruby)	Error	OK	Remained
PyXB (python)	OK	OK	Replaced with _
JAXB (Java)	Error	OK	Removed
Codesynthesis XSD (c++)	OK	OK	Replaced with _
XSD.exe (c#)	OK	OK	Removed

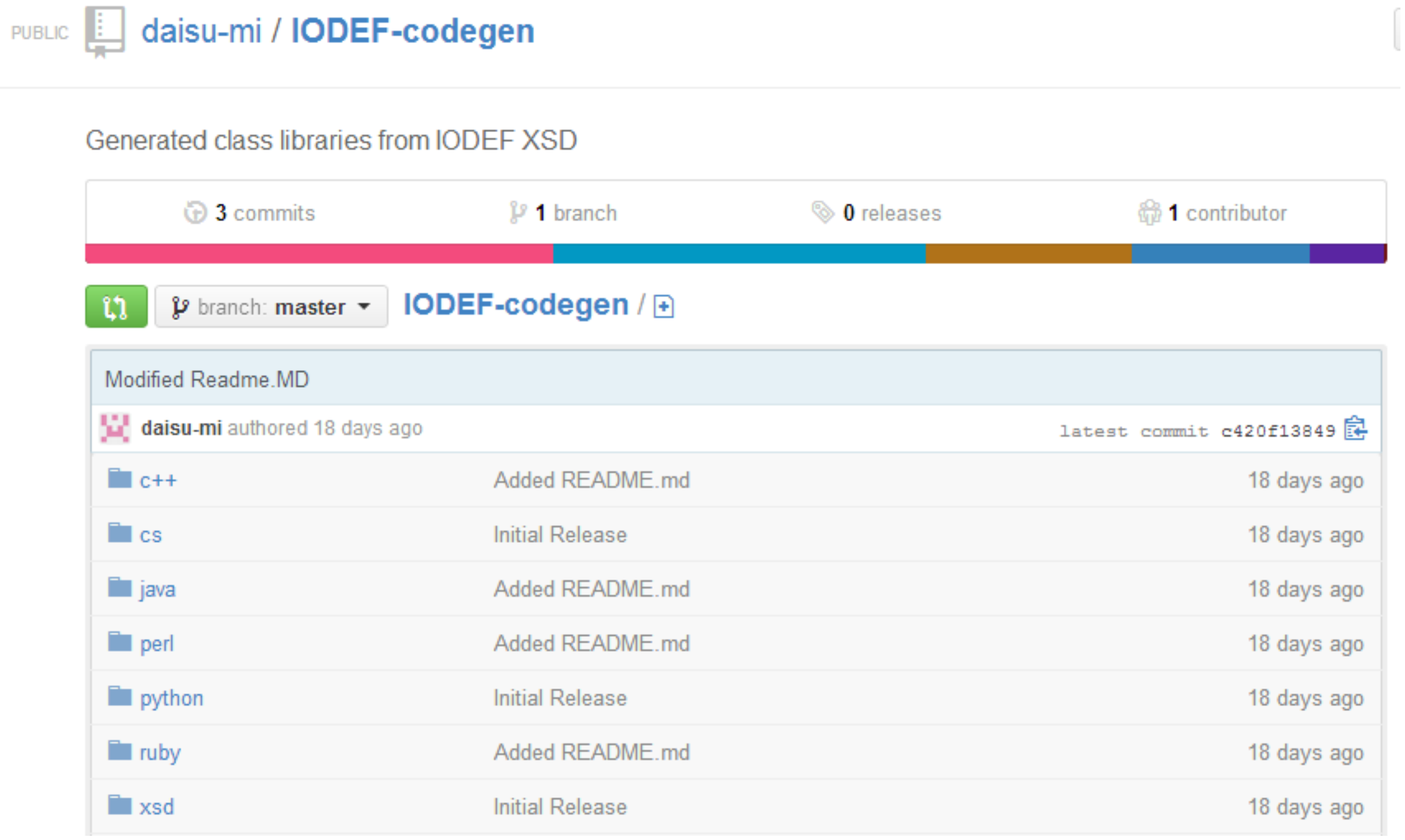
Use of generated libraries

- Case of XML::Pastor (perl) with replacing “-” with “_”

```
#!/usr/bin/perl
use XML::Pastor;
use IODEF.pm

my $iodef = IODEF::IODEF_Document->new;

  my $iodef_incident = IODEF::Type::IODEF_Document_Incident->new;

  $iodef_incident->purpose("reporting");

  $iodef_incident->IncidentID($incident_id);

(snip)

$iodef->Incident($iodef_incident);

print $iodef->to_xml_string();
(don’t forget replace with s/_/-/g)
```

Use for generated code

- See github.com/daisu-mi/IODEF-codegen



Generated class libraries from IODEF XSD

Modified Readme.MD

daisu-mi authored 18 days ago latest commit c420f13849

c++	Added README.md	18 days ago
cs	Initial Release	18 days ago
java	Added README.md	18 days ago
perl	Added README.md	18 days ago
python	Initial Release	18 days ago
ruby	Added README.md	18 days ago
xsd	Initial Release	18 days ago

#3 : "type" attribute @ Impact Class

- Motivation
  - JP-CERT/CC (CERT for Japan) defines their own category of incidents, so our university CERT (is in Japan) wants to categorize along with the JP-CERT's category.
- Problem
  - Comply with IODEF and JP-CERT's category

Categories used in JPCERT	"type" attributes @ Impact Class
Phishing site	social-engineering
Web page hijack	file ?
Malware propagation	file ? admin ?
Scan	recon
DoS/DDoS	dos
Control systems	ext- type?

#4 : “category” attribute @ NodeRole Class

- Problem
  - What is the suitable NodeRole for following nodes
    - Proxy Server : `<NodeRole category=“www?"/>`
    - Web Mailer: `<NodeRole category=“www?mail?"/>`
  - Available options

```
    client, server-internal, server-public, www, mail,
    messaging, streaming, voice, file, ftp, p2p, name,
    directory, credential, print, application, database,
    infra, log, ext-value
```

#5 : “action” attribute @ Expectation Class

- Motivation
  - Central CERT informs incident with expected action
    - `<Expectation action="`**`investigation`**`"/>`
  - Divisional CERT replies with incident reports
- Problem
  - What is expected action from divisional CERT to central
    - `<Expectation action=“`**`nothing ?`**`"/>`
    - This is just confirmation, rather than problem
  - Available options

```
nothing, contact-source-site, contact-target-site,
contact-sender, investigate, block-host, block-
network, block-port, rate-limit-host, rate-limit-
network, remediate-other, status-triage, status-
new-info, other, ext-value
```

#6 : Potential information leakage

- Problem
  - Given following conditions, number of incidents time (incident per second) might be disclosed.

```
    <Incident ID="1">
      <DetectTime>2013-09-11T04-57-00+09:00</DetectTime>

    <Incident ID="2">
      <DetectTime>2013-09-11T05-02-34+09:00</DetectTime>

    <Incident ID="3">
      <DetectTime>2013-09-11T05-09-12+09:00</DetectTime>
```

- Countermeasure
  - Assignment of random number for Incident ID
  - Use Alternative ID instead of incident ID
    - When exchanging IODEF with other society, remove the alternative ID.

#7 : Configuration of Nodes

- Problem
  - How to define software ID (swid) and configuration ID (configid)

```
<Node>
  <NodeRole category=“client”>
  <Service>
    <Application swid=“?” configid=“?”
     vendor=“Adobe” family=“Flash” name=“Adobe Flash”
     version=“11.8.800.168” patch=“” />
  </Service>
  <OperatingSystem swid=“” configid=“”
   vendor=“Microsoft”, family=“Windows” name=“Windows 8”
   version=“”, patch=“SP1” />
</Node>
```

Sumary

- Code-level Issues
  - Complexity of XSD
    - Evasion is XSD-to-XML-to-XSD (not recommend, but work)
  - Use of Hyphens
    - Replacing with other symbol for particular code generators

- Usability Issues
  - Local problems (in the case of JP-CERT/CC)
  - Value-assignment issues (need some use case document)
  - Potential information leakage (considering about ID)

Thank you for your attention