Relaxed—on the Way Towards True Validation of Compound Documents

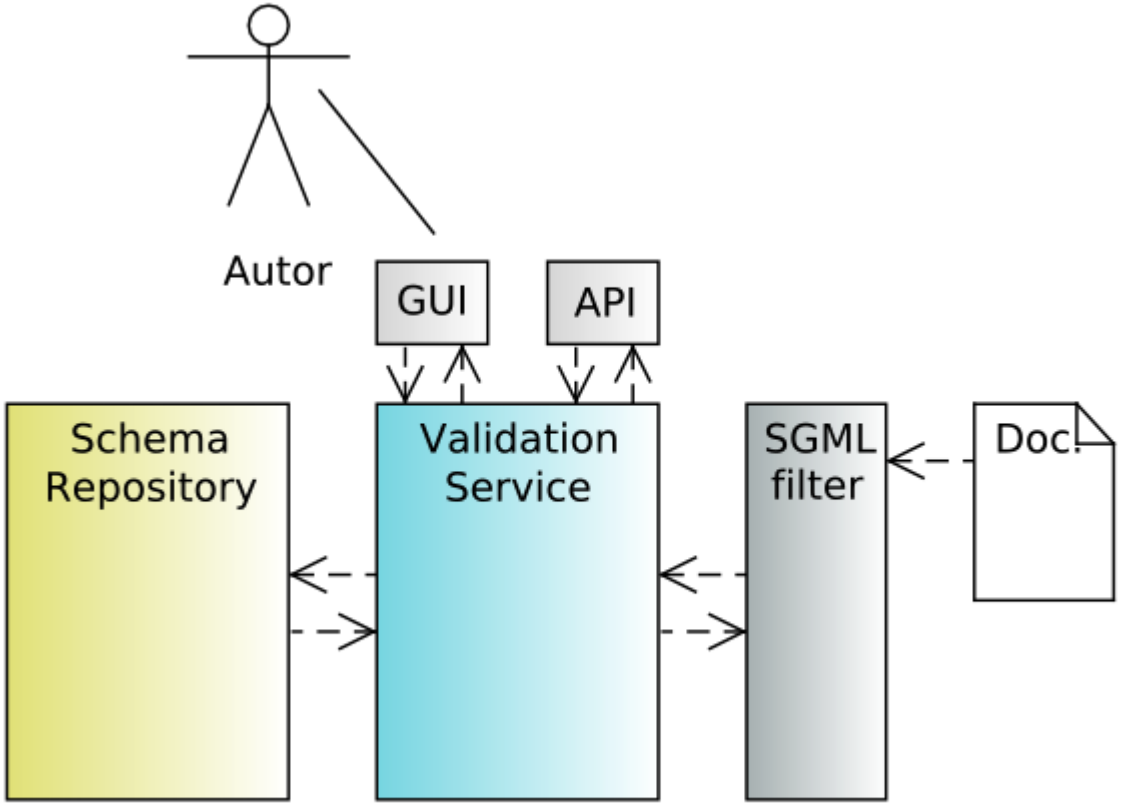
- Table of Contents
  - Introduction to Relaxed
  - Schema languages
  - HTML specification
  - Formalizing restrictions
  - Relaxed validation service
  - Compound documents
  - NVDL
  - Reference

Introduction to Relaxed

- Motivation
  - Make use of modern validation language's expressive power to validate more restrictions (Relax NG, Schematron)
  - Formalize specifications which cannot be formalized using common approaches (WCAG)
  - Create a validation service accessible through an GUI (for document authors) as well as through an API for automated processing
  - Allow straightforward compound documents validation

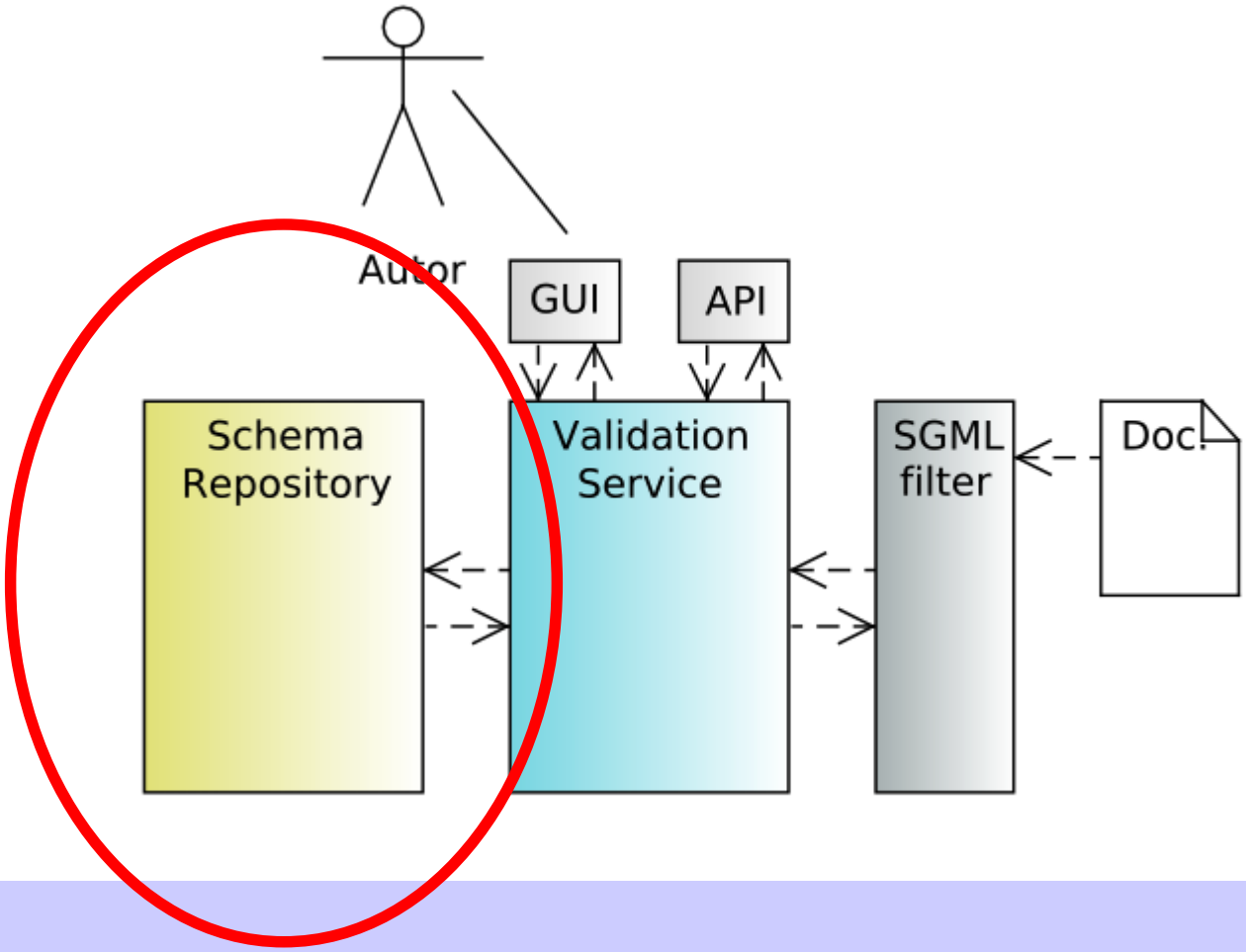
Introduction to Relaxed

- High-level architectural overview



Relaxed
the HTML validator
www.relaxed.cz

Schema languages



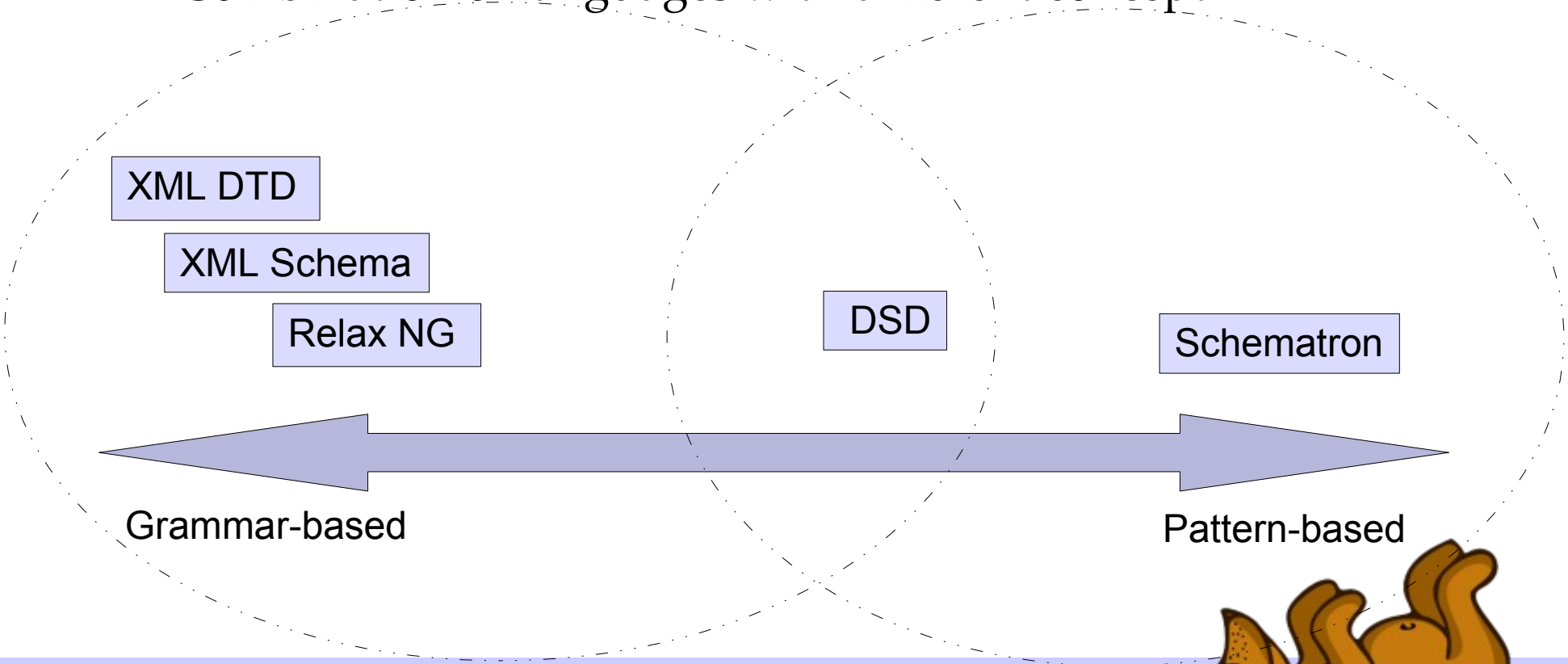
Relaxed
the HTML validator
www.relaxed.cz

Schema languages

- Searching for expressive power
  - XML DTD, XML Schema, XDR, DSD, Schematron, Relax NG
    - (XDR, Sox, Trax, ...)
- Criteria
  - Language concept
    - Grammar-based(XML Schema, Relax NG....) / pattern-based (Schematron, DSD)
  - Support for schema modularity
  - Interoperability
  - & other ... tool support, documentation...

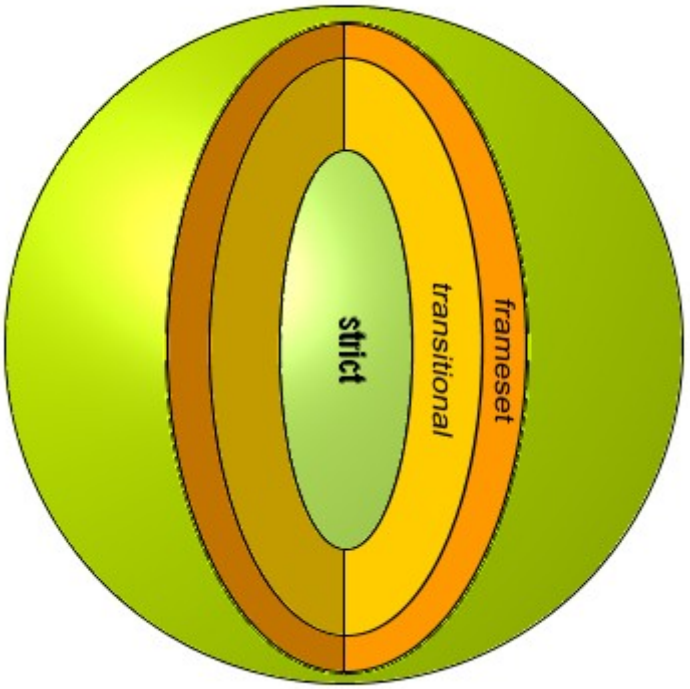
Schema languages

- Language concept
  - Combination of languages with different concept



Schema languages

- Modularity
  - Relax NG – combine, interleave
  - Schematron – simply add rules
    - Allowed unless restricted



HTML 4.01, XHTML 1.0
backward compatibility

Relaxed
the HTML validator
www.relaxed.cz

Schema languages

- Modularity – Relax NG
  - No need for driver schema

MODULE 1

```
<define name="br">
  <element name="br">
    <ref name="br.attlist"/>
    <empty/>
  </element>
</define>

<define name="br.attlist">
  <ref name="Core.attrib"/>
</define>
```

MODULE 2

```
<define name="br.attlist" combine="interleave">
 <optional>
   <attribute name="clear">
     <choice>
    <value>left</value>
    <value>all</value>
    <value>right</value>
    <value>none</value>
     </choice>
   </attribute>
 </optional>
</define>
```

MODULE 1 + MODULE 2

```
<include href="MODULE 1"/>
<include href="MODULE 2"/>
```

Schema languages

- Interoperability
  - Relax NG + Schematron = Good fellows

```
  <define name="select">

      <!-- If select element isn't marked as multiple, just one
 of his child option elements may be marked as selected -->

      <sch:pattern name="select.multiple.selected.options">
          <sch:rule context="html:select">
              <sch:report test="not(@multiple) and count(html:option[@selected]) > 1">
                   Select elements which aren't marked as multiple may not have more then
one selected option.
              </sch:report>
          </sch:rule>
      </sch:pattern>

    <element name="select">
      <ref name="select.attlist"/>
      <oneOrMore>
     <choice>
       <ref name="option"/>
       <ref name="optgroup"/>
     </choice>
      </oneOrMore>
    </element>
  </define>
```

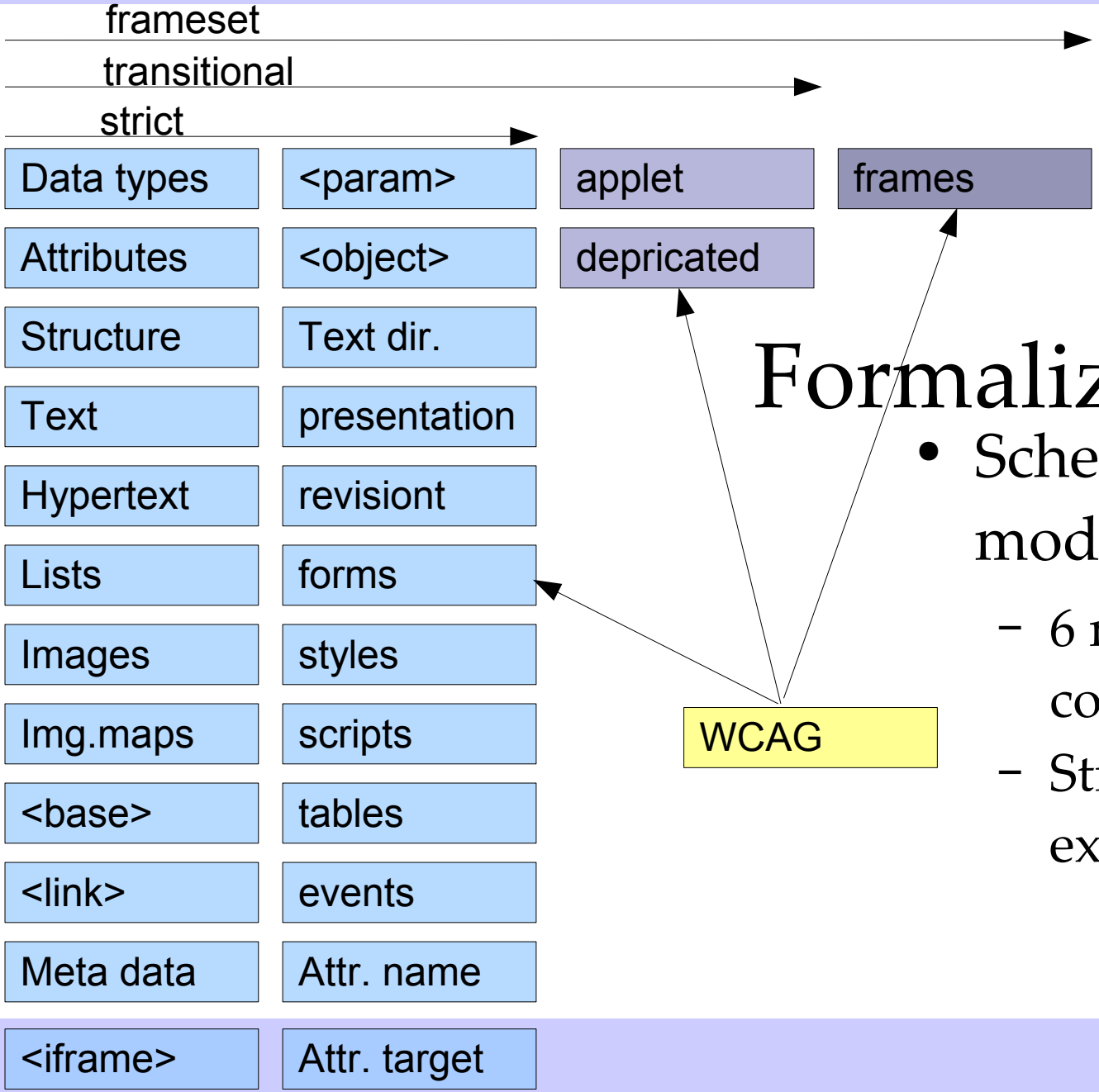
HTML specification

- Formalized
  - HTML 4.01, XHTML 1.0, WCAG 1.0
  - Formalize maximum restrictions – authors mostly rely on automated validation
- Formalization principles
  - Modularization (avoid duplicity, simplify maintenance)
    - Optional level of restriction
  - Testing

Formalizing restriction

- Formalization development cycle

Relaxed
the HTML validator
www.relaxed.cz



Formalizing restriction

- Schema repository modules' structure
  - 6 main validation configurations
  - Straightforward expansion
    - XHTML Basic

Relaxed
the HTML validator
www.relaxed.cz

Formalizing restriction

- Formalization examples (WCAG 1.0)

WCAG 1.0 Checkpoint 4.2 (Priority 3) Specify the expansion of each abbreviation or acronym in a document where it first occurs.

```
<sch:rule context="html:abbr">
  <sch:report test="not(@title) and not(preceding::html:abbr[. = string(current())][@title])">
      ...
  </sch:report>
</sch:rule>
```

WCAG 1.0 Checkpoint 3.4 (Priority 2) Consider using relative lengths instead of absolute in frameset.

```
<sch:rule context="html:frameset">
  <sch:report test="@rows and (not(contains(@rows, '%')) and not(contains(@rows, '*')))">
      ...
  </sch:report>
      ...
</sch:rule>
```


Formalizing restriction

WCAG 1.0

Checkpoint 5.1 (Priority 1) Identify row and column headers in table.

Checkpoint 5.1 (Priority 1) A table should have a caption

Checkpoint 5.5 (Priority 3) A table should have a summary

Checkpoint 5.6 (Priority 3) A table header should give abbreviation

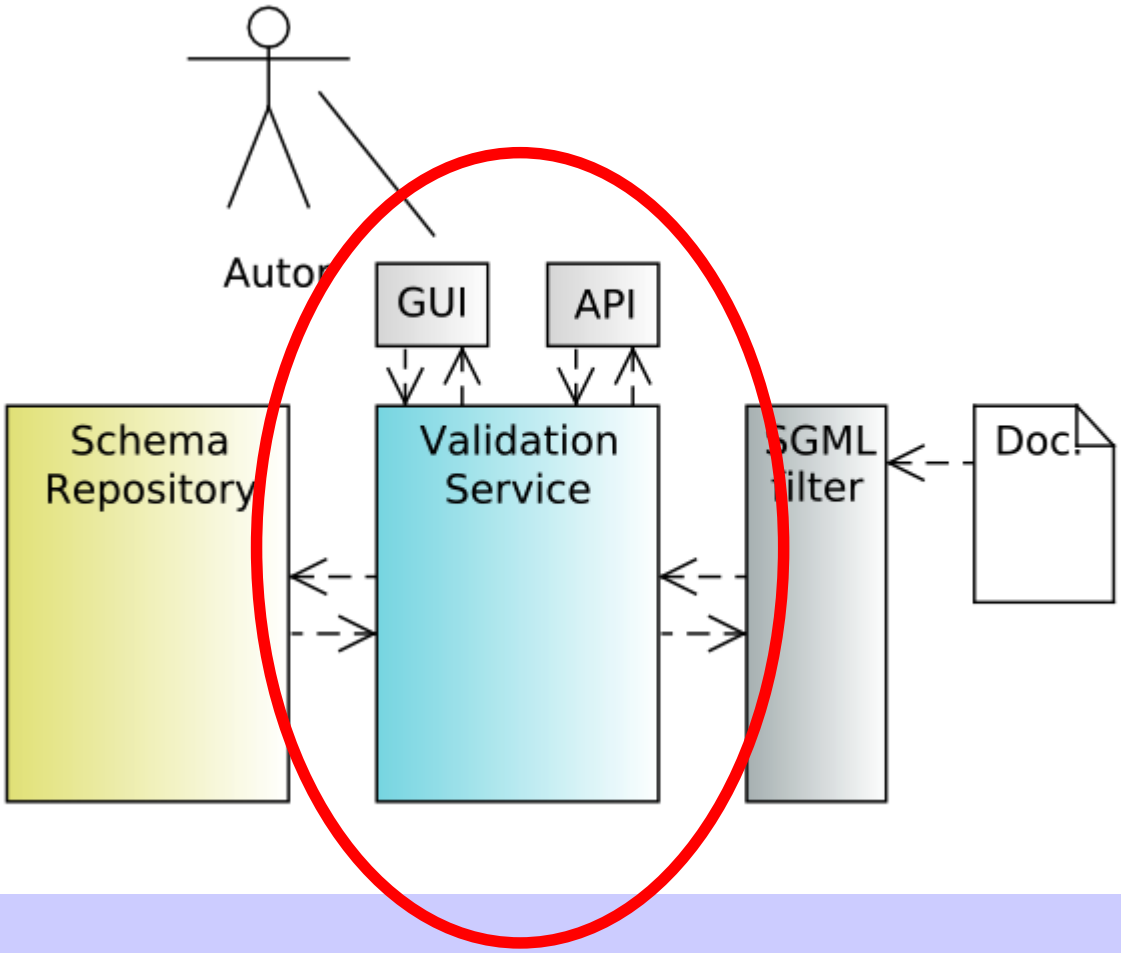
```
<sch:rule context="html:table">
    <sch:assert test="count(descendant::html:td) > 0 and count(descendant::html:th) > 0">
      WCAG 1.0 Checkpoint 5.1 (Priority 1) Identify row and column headers in table.
    </sch:assert>
    <sch:assert test="html:caption">
      WCAG 1.0 Checkpoint 5.1 (Priority 1) A table should have a caption
    </sch:assert>
    <sch:assert test="@summary">
      WCAG 1.0 Checkpoint 5.5 (Priority 3) A table should have a summary attribute
    </sch:assert>
  </sch:rule>
  <sch:rule context="html:th">
    <sch:assert test="@abbr">
      WCAG 1.0 Checkpoint 5.6 (Priority 3) A table header should have an abbr attribute to give
abbreviation
 </sch:assert>
</sch:rule>
```


Formalizing restriction

WCAG 1.0 Use header elements to convey document structure and use them according to specification. [Priority 2]. For example, in HTML, use H2 to indicate a subsection of H1.

```
    <sch:rule context="html:h1">
      <sch:assert test="following::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6][1][self::html:h1 or self::html:h2]
      or not(following::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6])">
        WCAG 1.0 Checkpoint 3.5 (Priority 2) Use header elements to convey document structure.
        Do order heading elements properly. Element <sch:value-of select="name(following::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or
self::html:h6][1])"/>
        may not directly follow h1. Do not skip heading levels.
      </sch:assert>
    </sch:rule>

    <sch:rule context="html:h2">
      <sch:report test="not(preceding::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6])">
          WCAG 1.0 Checkpoint 3.5 (Priority 2) Use header elements to convey document structure.
          The first heading element in the document should be h1 but not <sch:value-of select="name()"/>
      </sch:report>
      <sch:assert test="following::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6][1][self::html:h1 or self::html:h2 or self::html:h3]
      or not(following::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6])">
        WCAG 1.0 Checkpoint 3.5 (Priority 2) Use header elements to convey document structure.
        Do order heading elements properly. Element <sch:value-of select="name(following::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or
self::html:h6][1])"/>
        may not directly follow h2. Do not skip heading levels.
      </sch:assert>
    </sch:rule>

  .....

    <!-- There can be any heading after H5 -->
    <sch:rule context="html:h5">
      <sch:report test="not(preceding::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6])">
          WCAG 1.0 Checkpoint 3.5 (Priority 2) Use header elements to convey document structure.
          The first heading element in the document should be h1 but not <sch:value-of select="name()"/>
      </sch:report>
    </sch:rule>
    <sch:rule context="html:h6">
      <sch:report test="not(preceding::html:*[self::html:h1 or self::html:h2 or self::html:h3 or self::html:h4 or self::html:h5 or self::html:h6])">
          WCAG 1.0 Checkpoint 3.5 (Priority 2) Use header elements to convey document structure.
          The first heading element in the document should be h1 but not <sch:value-of select="name()"/>
      </sch:report>
    </sch:rule>

  </sch:pattern>
```

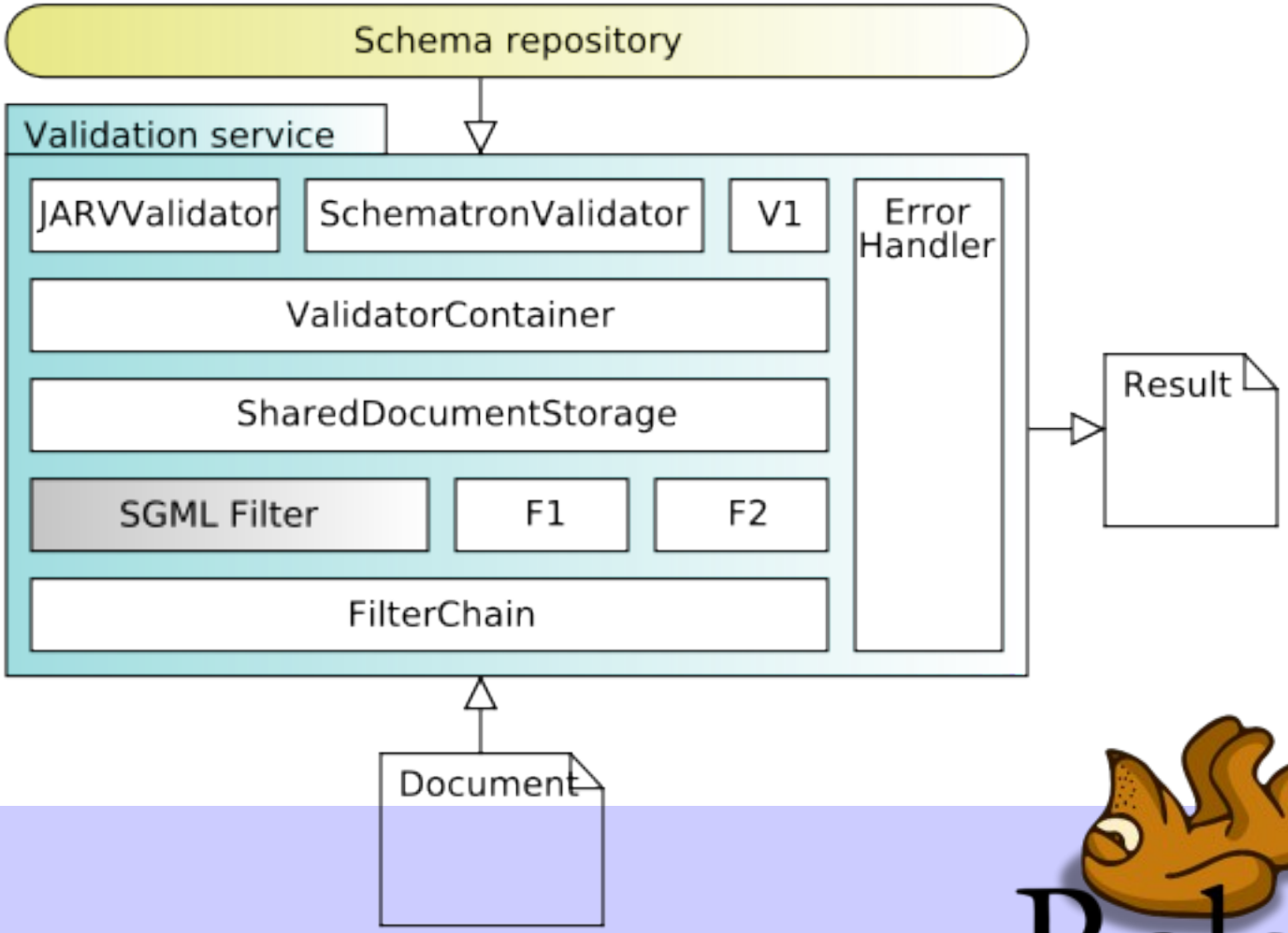

Relaxed validation service



Relaxed
the HTML validator
www.relaxed.cz

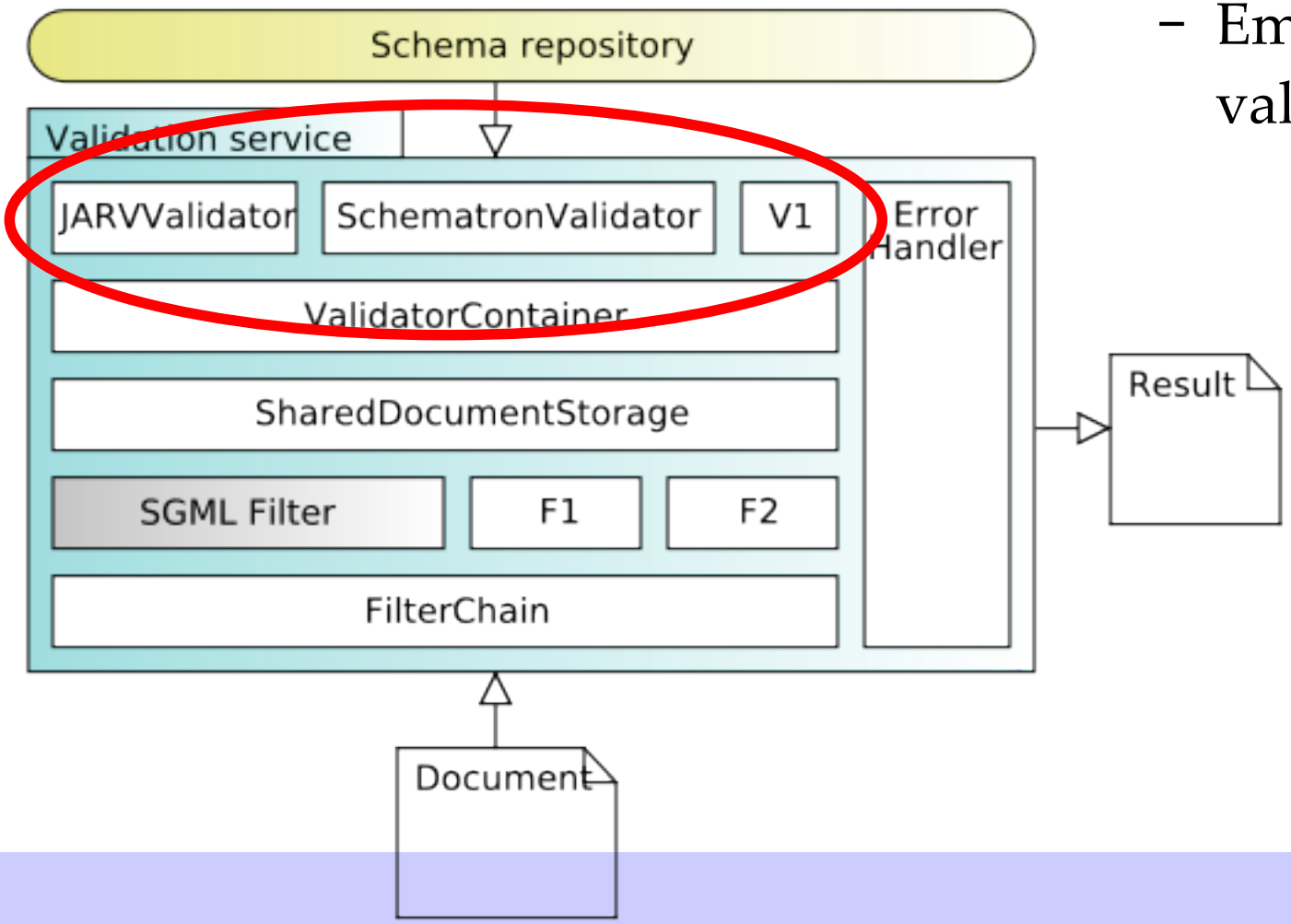
Relaxed validation service

- Detailed architectural overview



Relaxed
the HTML validator
www.relaxed.cz

Relaxed validation service

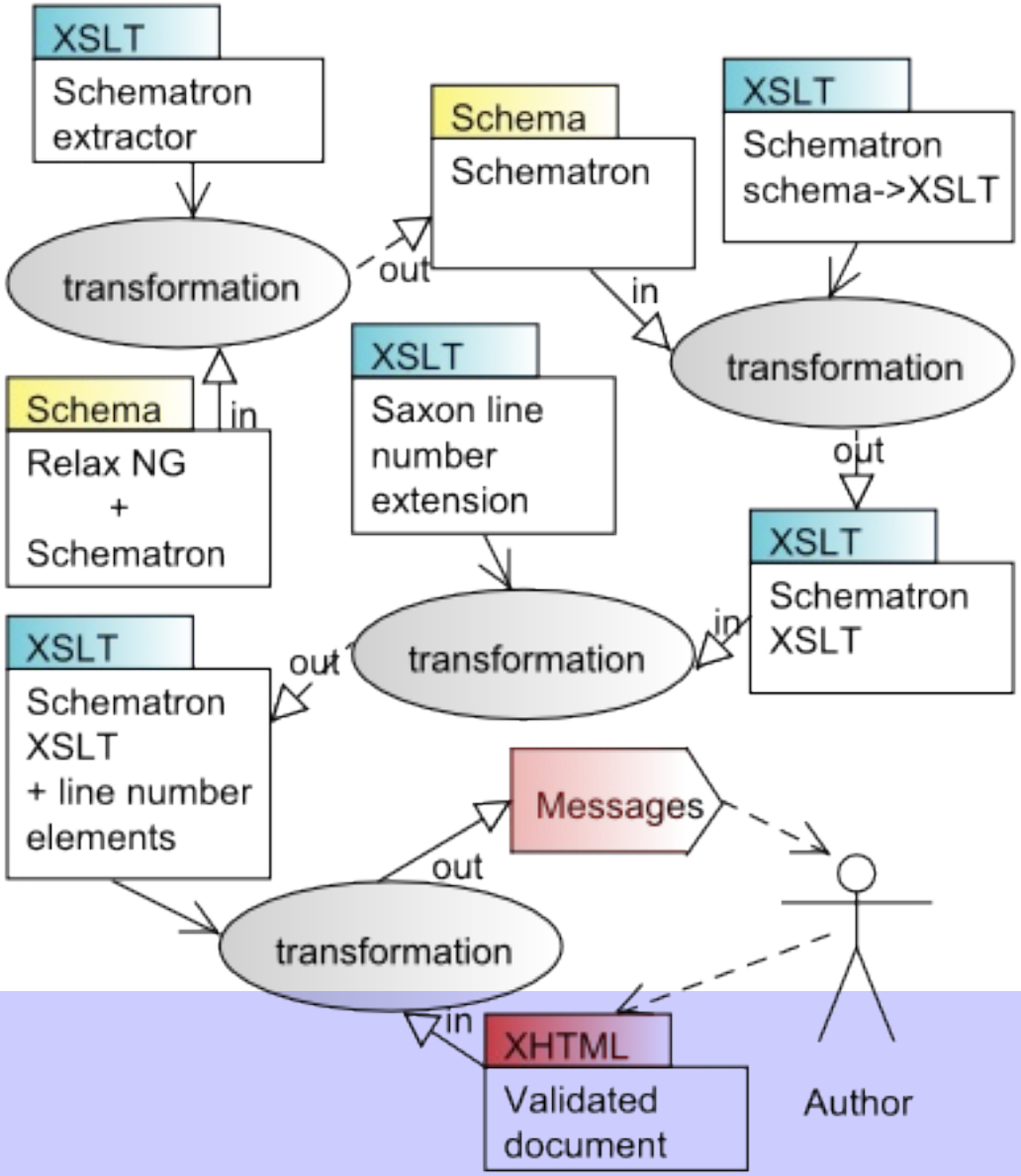


- Validation components
  - Embedded Schematron validator
    - JARV validator (vendor-neutral, implementation-independent and schema language independet interface for validators)
      - Relax NG (currently uses Sun MSV)

Relaxed
the HTML validator
www.relaxed.cz

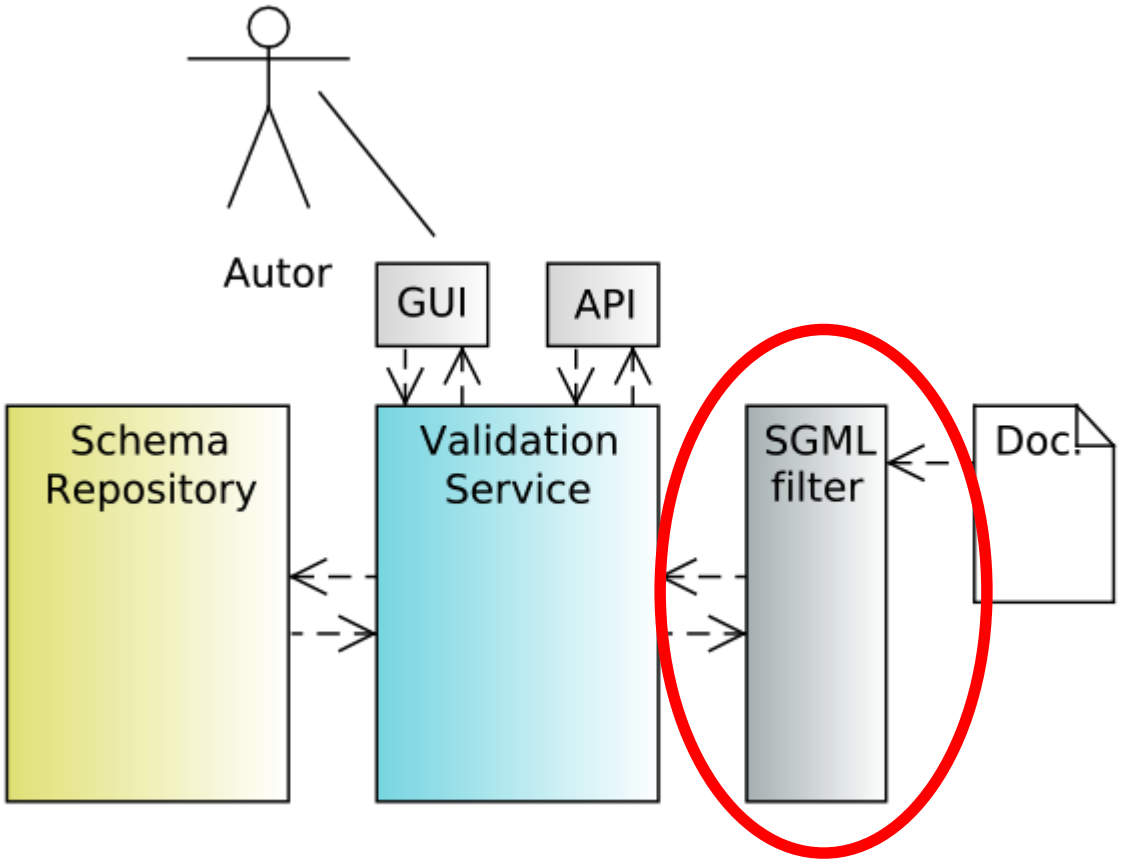
Relaxed validation service

- Embedded Schematron validator



Relaxed
the HTML validator
www.relaxed.cz

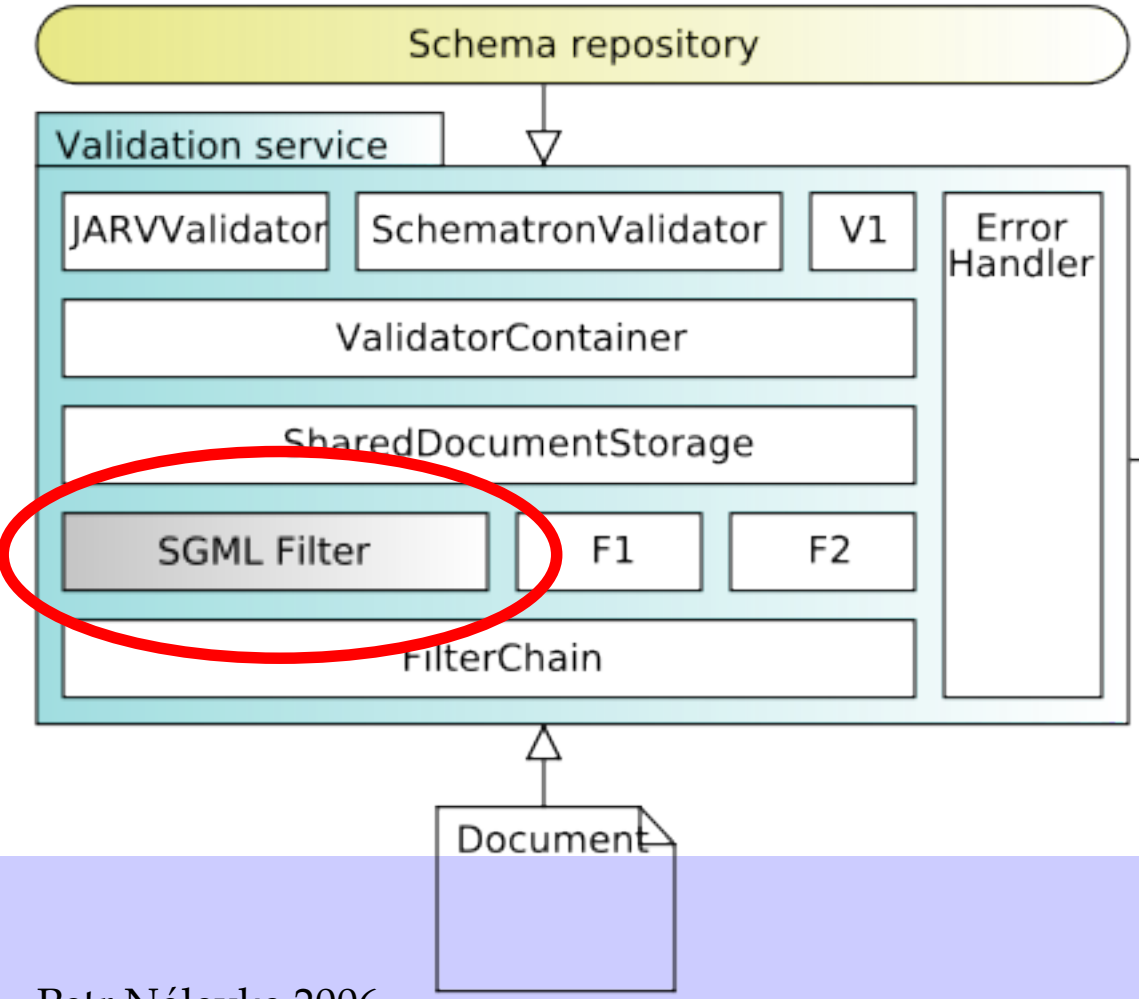
Relaxed validation service



Relaxed
the HTML validator
www.relaxed.cz

Relaxed validation service

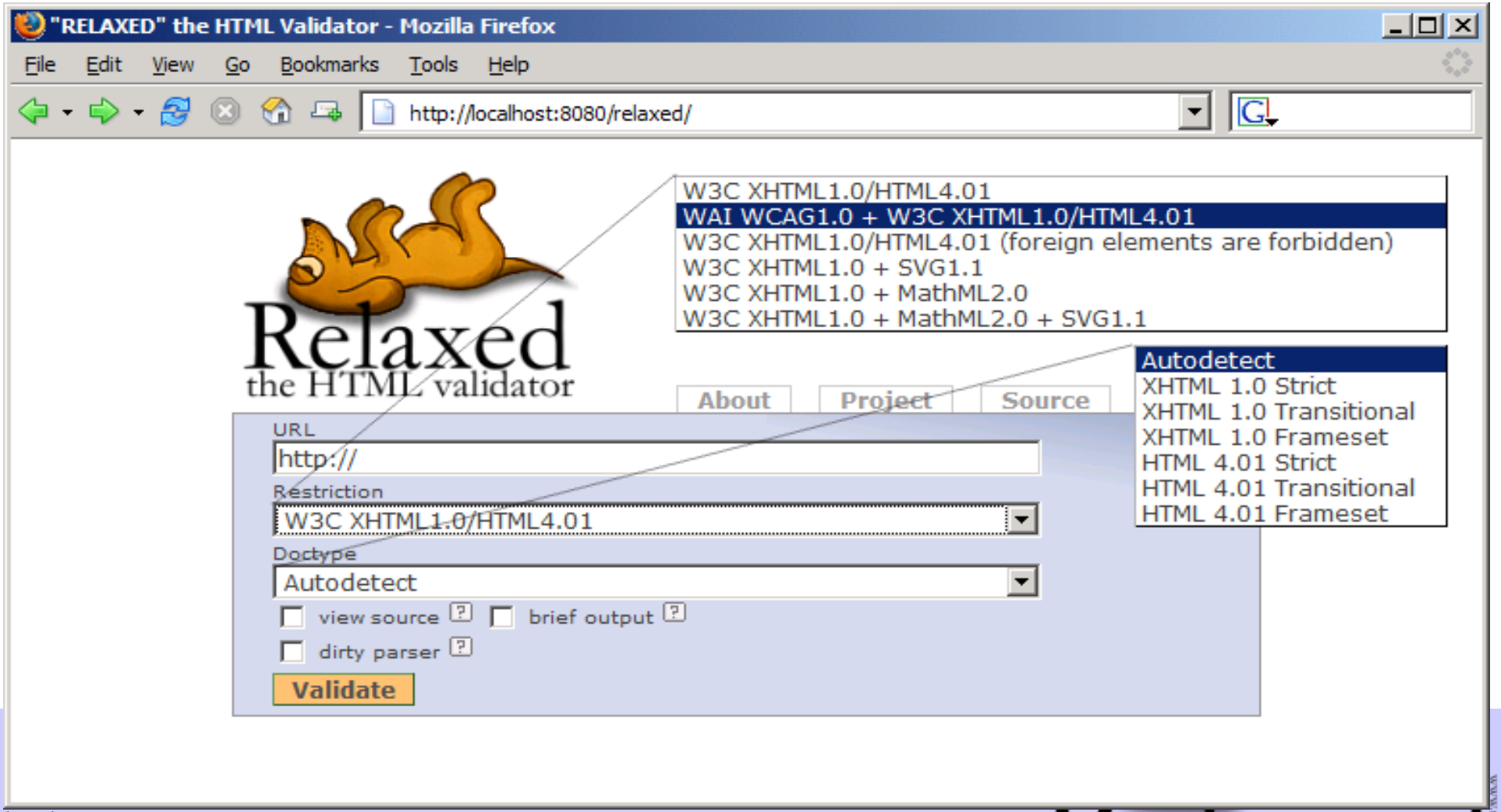
- Plenty of HTML 4.01 (SGML) documents on the Web
  - Important for real life operation
    - Based on TagSoupe library



Relaxed
the HTML validator
www.relaxed.cz

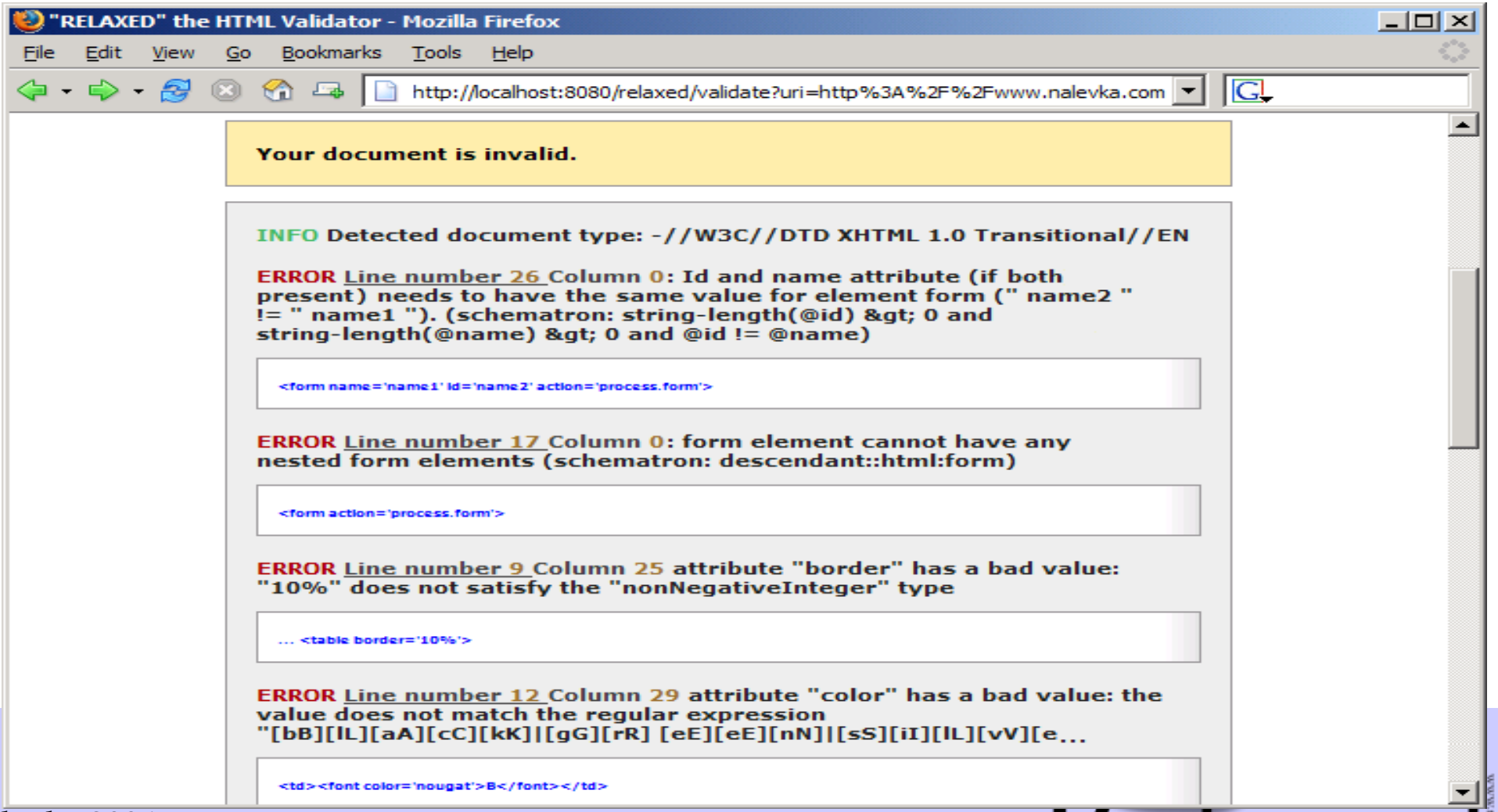
Relaxed validation service

- Web user interface



Relaxed validation service

- Web user interface



Relaxed validation service

- Other interfaces
  - Command line validation interface
  - Web-service interface (simple REST API)

```
<relaxed>
  <source url="http://nalevka.com/resources/
    relaxed/poc.html"/>
  <output result="Your document is invalid.">
    <message severity="INFO">
      <text>Forced document type: -//W3C//DTD XHTML 1.0 Strict//EN</text>
    </message>
    <message severity="ERROR">
      <locator line="9" column="25" />
      <text>attribute "border" has a bad value: "10%" does not satisfy the "nonNegativeInteger"
type</text>
      <source>...&lt;table border='10%'&gt;</source>
   </message>
   ... more messages ...
   </output>
</relaxed>
```

Compound documents

- The future of the Web
  - Demanding users... web moves from marked documents to a rich clients
  - XUL (Mozilla), Ajax (Google), Flash...
  - XML is well prepared
    - Support for grammar combination, namespaces
    - Many specific languages which can be combined and rendered by the browser
      - SVG, MathML, XForms, SMILE, RDF, RSS...
      - Need for compound document validation
        - New dimension, context of combined grammars

Compound documents

- Compound document validation in Relaxed
  - Using exclusively expressive power of Relaxed NG
    - Relax NG namespace and modularity support makes thinks easy
  - Allow foreign namespaces (W3C validator, doesn't allow those)
    - Primary precondition for compound document validation
  - XHTML + SVG, XHTML + MathML, XHTML + SVG + MathML

Compound documents

- Allow foreign namespaces

```
ALLOW ANY NAMESPACE MODULE
<define name="otherNamespaceElement">
  <element>
    <anyName>
      <except>
        <nsName ns="http://www.w3.org/1999/xhtml"/>
      </except>
    </anyName>
    <zeroOrMore>
      <choice>
        <attribute>
          <anyName>
            <except>
              <nsName ns="http://www.w3.org/1999/xhtml"></nsName>
            </except>
          </anyName>
        </attribute>
        <text/>
        <ref name="otherNamespaceElement"/>
      </choice>
    </zeroOrMore>
  </element>
</define>

<!-- CONTEXT -->
<define name="head.content" combine="interleave">
  <zeroOrMore>
    <ref name="otherNamespaceElement"/>
  </zeroOrMore>
</define>

<define name="Block.class" combine="choice">
  <ref name="otherNamespaceElement"/>
</define>

<define name="Inline.class" combine="choice">
  <ref name="otherNamespaceElement"/>
</define>
```

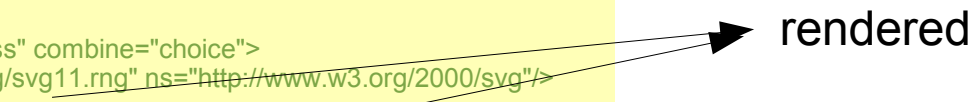
meta data

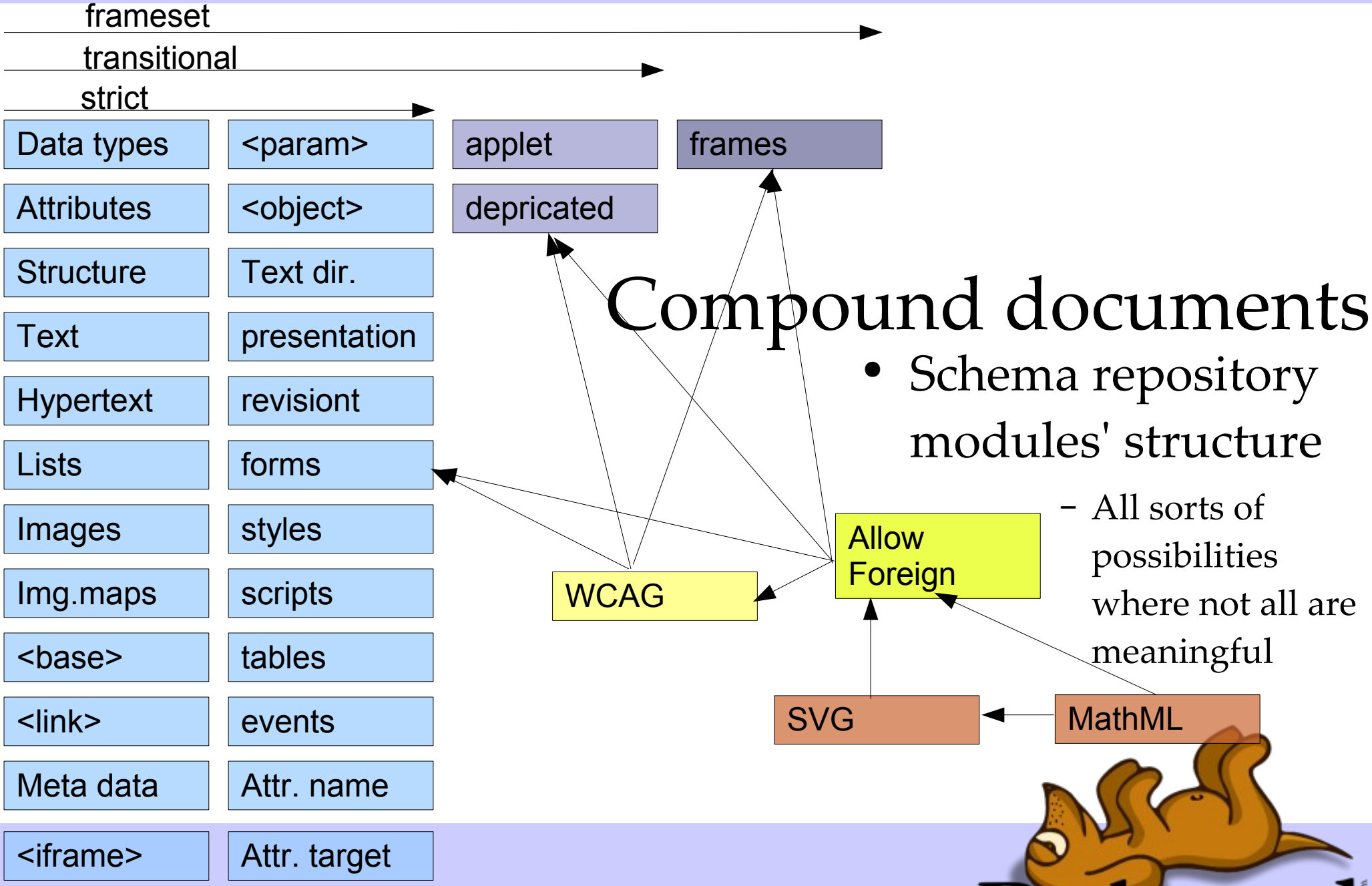
rendered

Compound documents

- XHTML + SVG

```
ALLOW ANY NAMESPACE MODULE
<define name="otherNamespaceElement">
  <element>
    <anyName>
      <except>
        <nsName ns="http://www.w3.org/1999/xhtml"/>
       <nsName ns="http://www.w3.org/2000/svg"/>
      </except>
    </anyName>
    <zeroOrMore>

SVG MODULE
<!-- CONTEXT -->
<define name="Block.class" combine="choice">
  <externalRef href="../svg/svg11.rng" ns="http://www.w3.org/2000/svg"/>
</define>

<define name="Inline.class" combine="choice">
  <externalRef href="../svg/svg11.rng" ns="http://www.w3.org/2000/svg"/>
</define>
```




frameset
transitional
strict
Data types
<param>
applet
frames
Attributes
<object>
depricated
Structure
Text dir.
Text
presentation
Hypertext
revisiont
Lists
forms
Images
styles
Img.maps
scripts
<base>
tables
<link>
events
Meta data
Attr. name
<iframe>
Attr. target
Compound documents
Schema repository modules' structure
All sorts of possibilities where not all are meaningful
Allow Foreign
WCAG
SVG
MathML
Relaxed
the HTML validator
www.relaxed.cz

NVDL (Namespace-based Validation Dispatching Language)

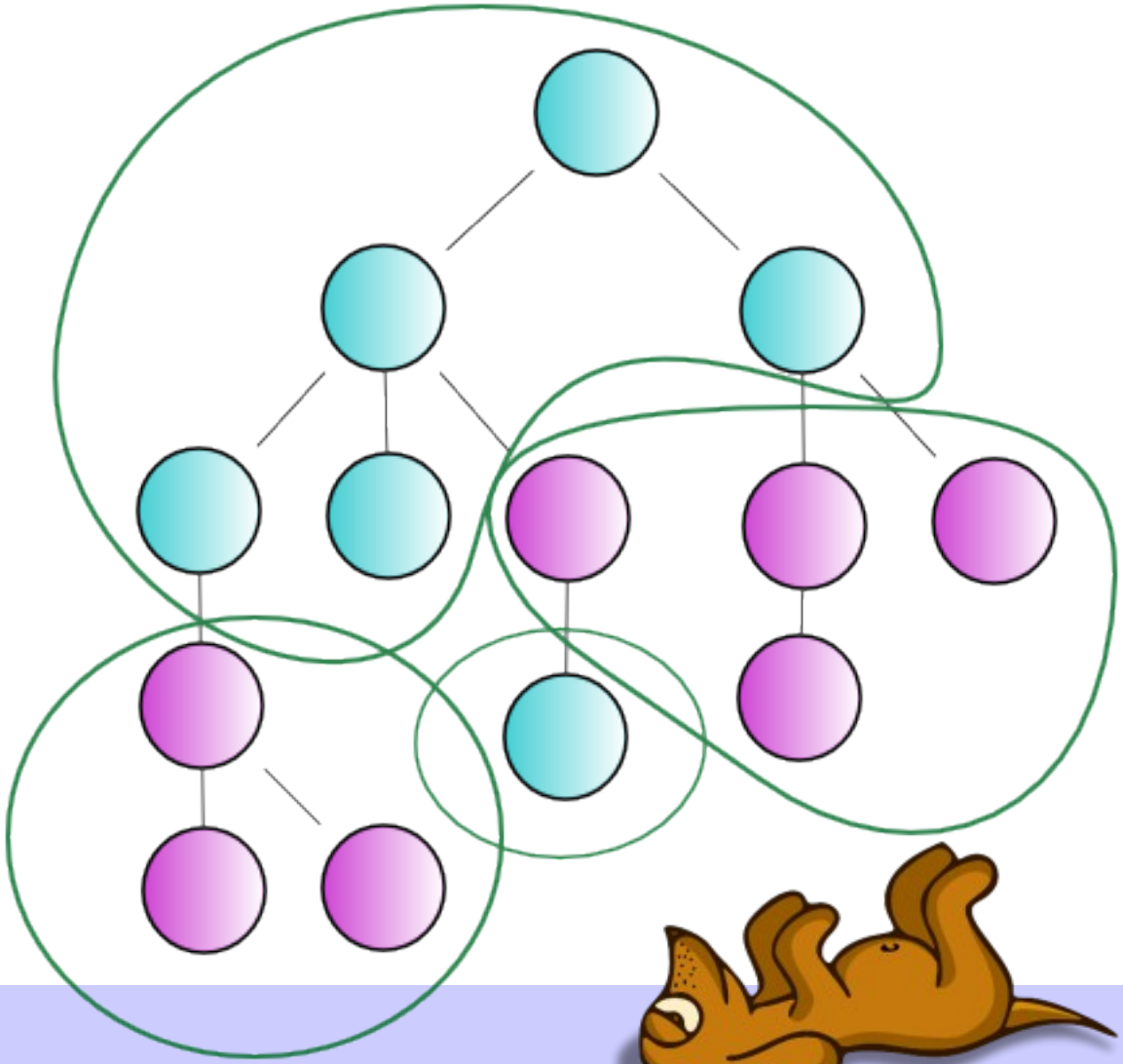
- ISO/IEC 19757-4 NVDL
  - History
    - RELAX Namespace(2001)
    - DSDL Part 4 Committee Draft (2002)
    - Modular Namespaces by James Clark (2003)
    - Namespace Switchboard by Rick Jelliffe (2003)
    - Namespace Routing Language by James Clark (2003)

NVDL

- Advantages
  - Validator transparent
    - NVDL engine just distributes validation candidates to appropriate validator
  - Schema languages transparent
    - We can combination of different schema languages (XML Schema, Relax NG)... in real life schemas are writen in different languages
  - Standardized and flexible way to express different grammars context

NVDL

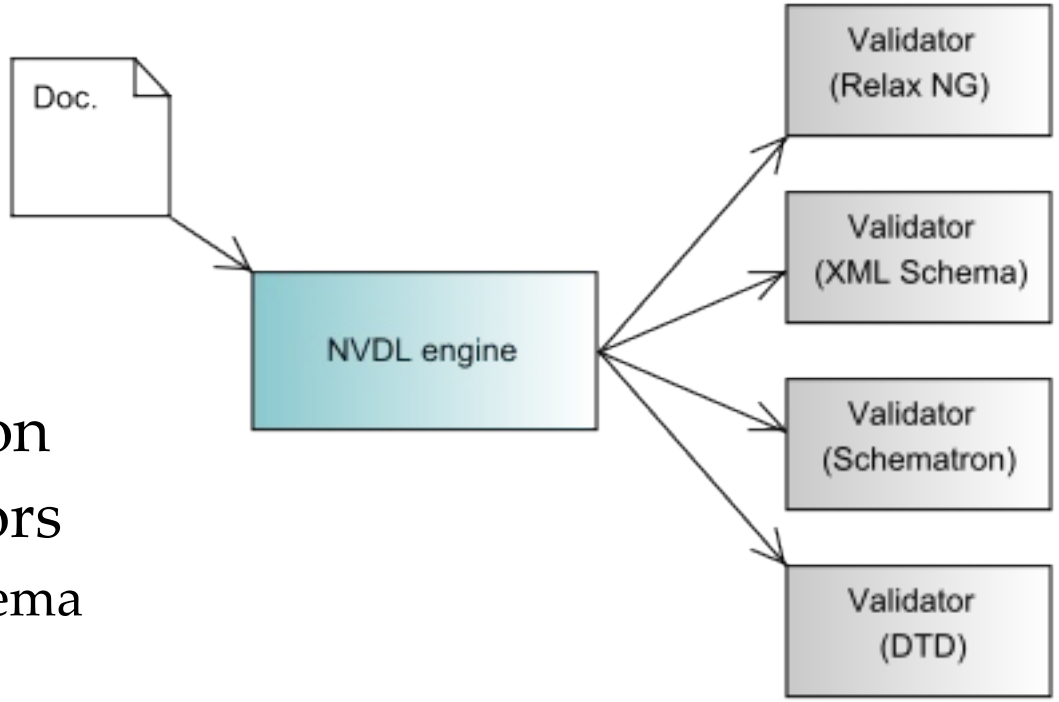
- Process
  - Dividing compound documents into fragments, according to defined rules and their namespace
  - Fragments are turned into validation candidates



Relaxed
the HTML validator
www.relaxed.cz

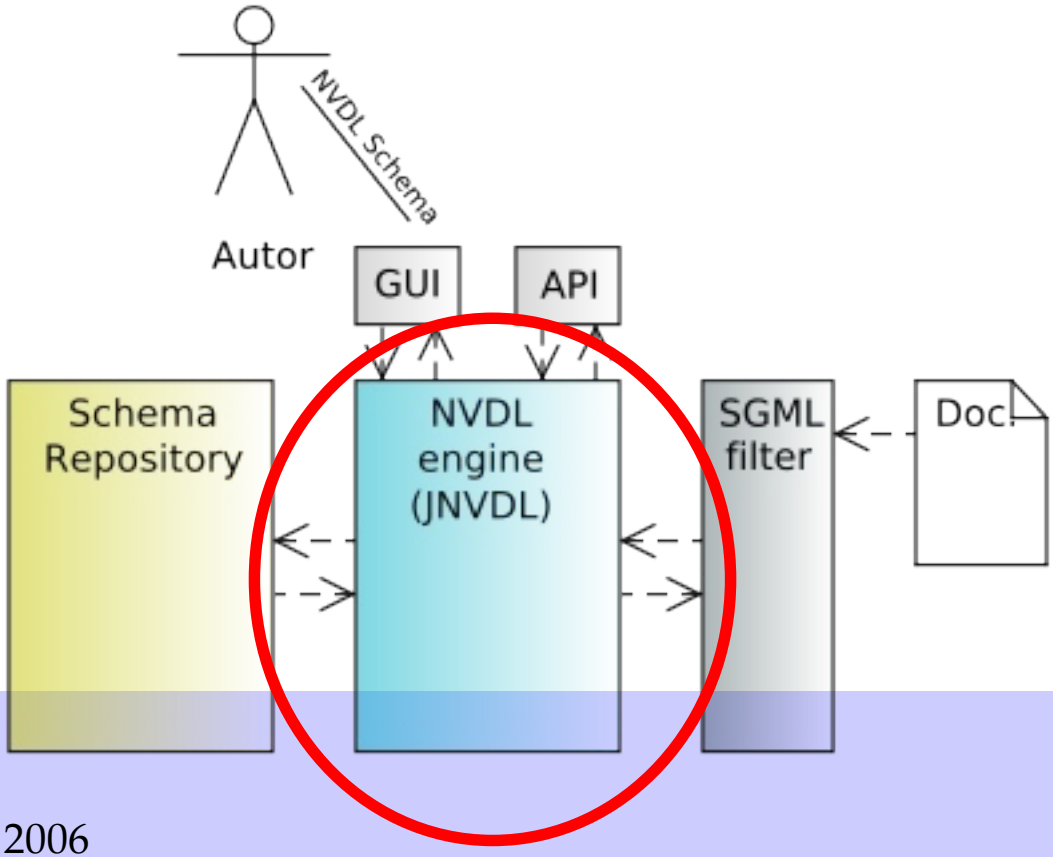
NVDL

- Process
  - Dispatching validation fragments to validators
    - According to schema namespace



NVDL

- Future plans
  - Replacement of current core validation service with JNVDL (NVDL implementation)



Reference

- HTML document authors using the online service
  - Accessible through http://www.relaxed.cz
    - Better outputs, compound document support
- EIAO (European Internet Accessibility Observatory)
  - "The EIAO project will establish the technical basis for a European Internet Accessibility Observatory. Frequently updated assessment data will be available online from a data warehouse providing a basis for benchmarking, policymaking, research and actions to develop accessibility to Internet."

Thank you for your attention