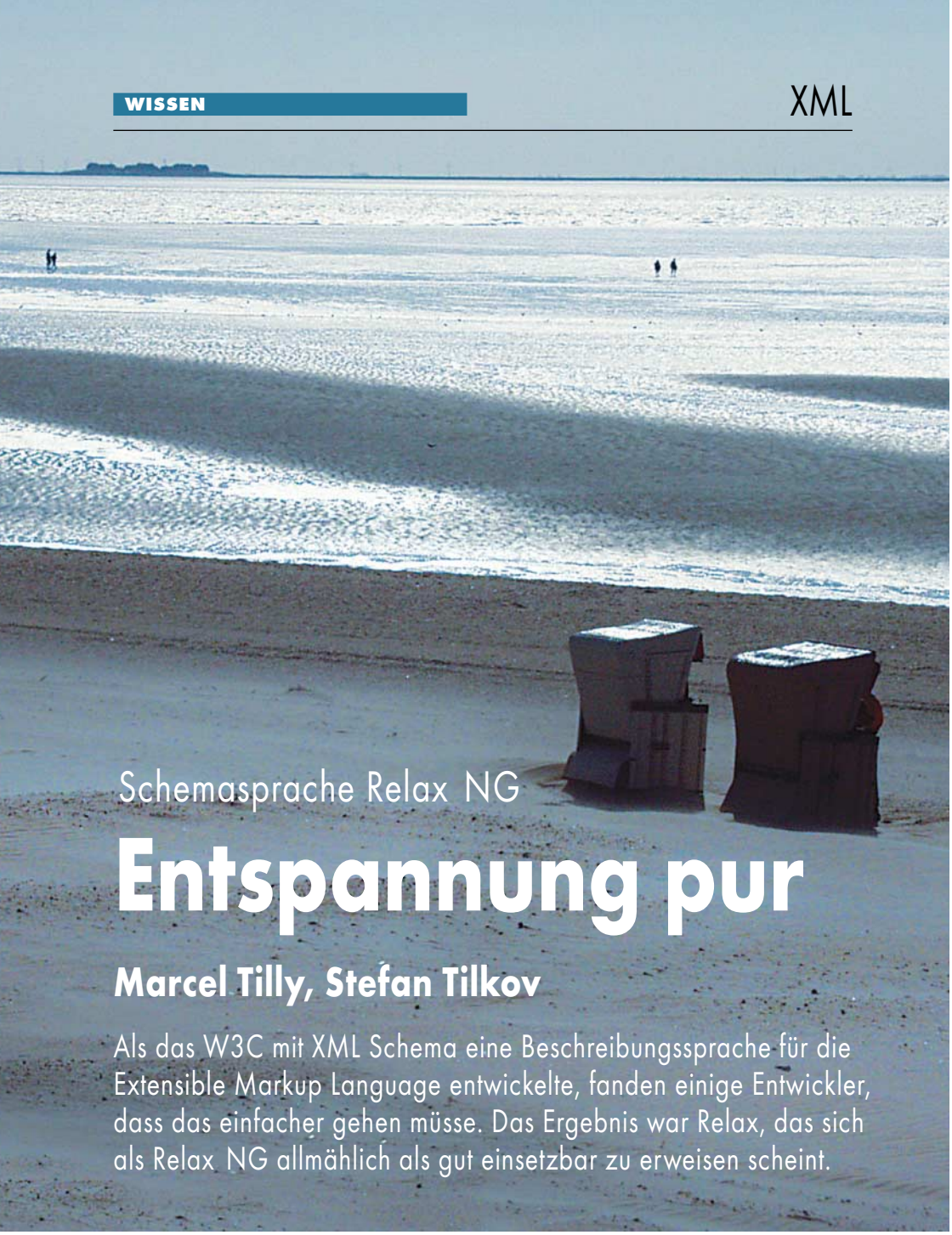

Schemasprache Relax NG

Entspannung pur

Marcel Tilly, Stefan Tilkov

Als das W3C mit XML Schema eine Beschreibungssprache für die Extensible Markup Language entwickelte, fanden einige Entwickler, dass das einfacher gehen müsse. Das Ergebnis war Relax, das sich als Relax NG allmählich als gut einsetzbar zu erweisen scheint.

Zur Beschreibung der Struktur eines XML-Dokuments dienen Schemasprachen, die Informationen für eine Validierung sowie für diverse andere, automatisierbare Schritte bei der Verarbeitung liefern können. Noch aus der guten alten SGML-Zeit stammen DTDs (Document Type Definitions); seit 2001 setzt sich die vom W3C veröffentlichte Sprache XML Schema durch. Abseits vom Mainstream existiert eine Alternative: das ebenso einfach zu lesende wie zu schreibende Relax NG (gesprochen wie das englische „relaxing"), das das W3C mittlerweile sogar innerhalb eigener Spezifikationen (RDF, XHTML 2) verwendet.

Einfachheit und leichte Erlernbarkeit waren die Hauptziele bei der Entwicklung von Relax NG – der Hauptunterschied zu XML Schema. So ist dessen Spezifikation mehr als zehnmal, ohne den (von Relax NG nicht abgedeckten) Teil der Datentypen immerhin noch mehr als sechsmal so umfangreich wie die Spezifikation von Relax NG. Trotzdem erheben dessen Autoren Murata Makoto und James Clark den Anspruch, mit ihrem Entwurf einen mindestens ebenbürtigen Konkurrenten ins Leben gerufen zu haben.

Während der Fokus von XML Schema stark auf der Definition der verwendeten Typen liegt und somit eher eine objektorientierte Dokumenttypdefinition [1] ist, setzt Relax NG darauf, die Struktur eines XML-Dokuments zu beschreiben. Relax NG bedeutet „Regular Language for XML – New Generation", seit Dezember 2001 mit Standardstatus bei OASIS (Organization for the Advancement of Structured Information Standards).

In Relax NG definieren so genannte Patterns ein Schema. Dabei sind drei Basis-Patterns *text*, *attribute* und *element* bekannt. Ein *element*-Pattern kann andere *element*-, *attribute*- und *text*-Pattern enthalten. Ein *attribute*- hingegen kann nur ein *text*-Pattern enthalten. Zusätzlich gibt es noch weitere Patterns, die Listen, sortierte oder unsortierte Gruppen und Wertebereiche definieren können. Selbstverständlich werden Namespaces unterstützt. Relax NG kennt allerdings nur zwei Built-in-Datentypen, *token* und *string*, und unterscheidet sich dadurch vom ausgeprägten Typsystem von XML Schema.

Wie leicht erlernbar Relax NG ist, zeigt Listing 1 anhand eines Musik- und Filmgeschäfts. Ein Relax-NG-Schema beginnt mit dem Wurzelelement *grammar*. Autoren dürfen zwar ein Schema ohne dieses Pattern schreiben, können es allerdings in diesem Fall nicht von anderen Schemata aus referenzieren beziehungsweise inkludieren. Verwendet man das Wurzelelement *grammar*, muss ein Subelement *start* vorhanden sein, das das Wurzelelement des Instanzdokumentes, in diesem Fall *musicstore*, definiert. Da *musicstore* aus Elementen bestehen soll, muss es seinerseits ebenfalls eins sein. Einen Namen erhält es durch das Attribut *name*. Hier zeigt sich ein weiterer Unterschied zu XML Schema: Während dort Attribute-Elemente erst am Ende einer *ComplexType*-Definition stehen, darf bei Relax NG das Attribut-Element an jeder Stelle innerhalb eines Elements stehen. Das erhöht die Flexibilität bei der Strukturdefinition eines XML-Dokumentes. Zusätzlich können Autoren weitere Struktur-Pattern beliebig schachteln.

Daten zum Musik- und Filmladen

Der „Music Store" soll CDs (mindestens eine) und DVDs (optional) im Angebot

haben. *interleave* legt fest, dass die Reihenfolge der Elemente im Instanzdokument ungeordnet ist. Fehlte es, müssten die Elemente in genau der Reihenfolge auftreten, wie im Schemadokument definiert. *zeroOrMore* bedeutet, dass 0 bis n DVD-Elemente auftreten dürfen, während *oneOrMore* dafür steht, dass mindestens ein CD-Element vorhanden sein muss. Für den Inhalt der beiden enthält Listing 1 bisher keine Definitionen. Hier steht nur ein Verweis auf den *content* über das *ref*-Pattern; das heißt, sie sind an anderer Stelle des Dokuments beschrieben. Man kann in Relax NG auf externe Referenzen verweisen – unter Verwendung des *external Ref*-Pattern (beispielsweise *<externalRef href="content.rng"/>*).

Zum Inhalt der Instanzen: *content* ist als Named-Pattern definiert und steht in der Struktur neben dem *start*-Element. Named-Pattern darf man nur bei vorhandenem *grammar*-Element verwenden (ein weiterer Grund, weshalb das hier geschehen ist).

```
<define name="content">
 <attribute name="name"/>
 <choice>
  <ref name="music"/>
  <ref name="movie"/>
 </choice>
</define>
```

content umfasst ein Attribut *name* und entweder eine Referenz auf das Element *music* oder auf *movie*, wobei *<ref...* wiederum bedeutet, dass die Definition sich an anderer Stelle desselben Dokuments befindet. Das *choice*-Pattern gibt an, dass im Instanzdokument eines der aufgeführten Elemente verwendet werden muss. Die beiden sind ebenfalls als Named-Pattern definiert (siehe Listing 2: *<define name="music">...*).

Für das Attribut *type* vergibt Listing 2 einen feststehenden Wert (*<value>*). Damit ist für das Attribut im Instanzdokument an dieser Stelle nur der Wert *music* gültig. Über das *value*-Element in Kombination mit dem *choice*-Pattern lassen sich außerdem Enumerationen abbilden. So kann die Kategorie, in der die CD oder DVD einsortiert ist, nur aus der Liste der Elemente Pop, Rock, R&B, Classic oder Folk kommen. Eine Stelle, an der sich die Einfachheit von Relax NG zeigt: Viele Elemente lassen sich kombinieren. Zum einen kann ein *choice*-Pattern kennzeichnen, dass die enthaltenen Elemente oder Attribute alternativ auftreten können, zum anderen kann es innerhalb eines *attribute*-Pattern zur Abbildung einer Enumeration dienen.

Des Weiteren kann das *music*-Element entweder von einem einzelnen Künstler (*artist*) oder von einer Gruppe (*group*) sein und aus einem oder mehreren Stücken (*title*) bestehen. Die Stücke haben wiederum einen Namen und ein Länge.

Unterschiedliche Namensräume

Namespaces bieten einen Mechanismus, auf der Basis von URIs XML-Vokabulare auseinander zu halten. Relax NG bietet, so wie in XML-Dokumenten üblich, ebenfalls die Verwendung von Namespaces an. Autoren können Default-Namespaces über *xmlns* und Präfixe wie *xmlns:x1* verwenden. Üblicherweise sind Namespaces als URI definiert, wie *xmlns="http://www.innoq.com/relaxng/musicstore"*. Es handelt sich allerdings nur um einen Namen; die Adresse muss kein Dokument enthalten. Präfixe helfen, Elemente aus verschiedenen Namespaces zu mixen, ohne ständig den gesamten Namen aufführen zu müssen. Soll ein Namespace für das Instanzdokument festgelegt werden, so geschieht dieses über das Setzen des Attribute *ns*:

```
<grammar
 xmlns="http://relaxng.org/ns/
 structure/1.0"
 ns="http://innoq.com/rng/
 musicstore">
```

In diesem Fall muss man im Instanzdokument den Namensraum des *musicstore*-Elements über das Attribut *xmlns* auf *http://innoq.com/rng/musicstore* setzen (siehe den Anfang von Listing 4).

movie soll ein Element aus einem anderen Namensraum enthalten. Der Kommentar „Movie Definition“ soll als XML-Element aus dem XHTML-Namensraum vorkommen. Hierzu kann man im *define*-Element ein Namensraum-Präfix (*xmlns:x*) definieren. In diesem Fall dürfen innerhalb des *define* Elemente dieses Namensraums auftauchen, wie es durch die Überschrift *h1* der Fall ist.

```
<define name="movie"

xmlns:x="http://www.w3.org/1999/
xhtml">
 <x:h1>Movie Definition</x:h1>
 <attribute name="type">
  <value>movie</value>
 </attribute>
```

Die Struktur von *movie* ist vergleichbar mit der von *music*. Das Attribut *type* kann bei Filmen verständlicherweise nur den Wert *movie* annehmen. Und es gibt eine Liste von Schauspielern (*actor*) anstelle der Stücke-Liste sowie eine abweichende Liste für die Kategorienauswahl, wie Listing 2 verdeutlicht.

Schließlich soll optional eine Beschreibung im Instanzdokument zum Film erlaubt sein, die XHTML-Tags beinhalten kann. Es gibt in der XHTML2-Spezifikation eine Abbildung auf Relax-NG-Module. Für eine Beschreibung reichen die Abschnitte „Attribute Collections“, „Inline Text“ und „Datatypes“. Liegen diese RNG Schema in den Dateien xhtml-attribs-2.rng, xhtml-inltext-2.rng und xhtml-datatypes-2.rng vor, so können Autoren sie im Music-Store-Schema inkludieren. Das *movie*-Element kann in diesem Fall das Element *Inline.model* (befindet sich in xhtml-inltext-2.rng) referenzieren (in den unterschiedlichen Entwürfen mag dies voneinander abweichen).

```
<optional>
 <element name="description">
  <ref name="Inline.model"/>
 </element>
</optional>
</define>
<include
 href="xhtml-attribs-2.rng"
 ns="http://www.w3.org/2002/06/
 xhtml2"/>
<include
 href="xhtml-inltext-2.rng"
 ns="http://www.w3.org/2002/06/
 xhtml2"/>
<include
 href="xhtml-datatypes-2.rng"
 ns="http://www.w3.org/2002/06/
 xhtml2"/>
```

Listing 1: Kurz-Schema

```
<grammar
 xmlns="http://relaxng.org/ns/structure/1.0">
 <start>
  <element name="musicstore">
   <attribute name="name"/>
   <interleave>
    <zeroOrMore>
     <element name="dvd">
      <ref name="content"/>
     </element>
    </zeroOrMore>
    <oneOrMore>
     <element name="cd">
      <ref name="content"/>
     </element>
    </oneOrMore>
   </interleave>
  </element>
 </start>
</grammar>
```

iX-TRACT

- Relax NG, das in der Version 2.0 vorliegt, ist für Entwickler von XML-Schemasprachen einfacher als die des W3C.
- Im Gegensatz zu XML Schema, das eher eine objektorientierte Dokumenttypdefinition ist, setzt Relax NG darauf, die Struktur eines XML-Dokuments zu beschreiben.
- Die Unterstützung von Namensräumen sowie die Wahl, eine ausführliche oder eine knappe Notation zu verwenden, erleichtern die Arbeit.

Listing 2: Relax-NG-Schema: Musikladen

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar
  xmlns="http://relaxng.org/ns/structure/1.0"
  ns="http://innoq.com/rng/musicstore">
    <start>
        <element name="musicstore">
            <attribute name="name"/>
            <interleave>
                <zeroOrMore>
                    <element name="dvd">
                        <ref name="content"/>
                    </element>
                </zeroOrMore>
                <oneOrMore>
                    <element name="cd">
                        <ref name="content"/>
                    </element>
                </oneOrMore>
            </interleave>
        </element>
    </start>
    <define name="content">
        <attribute name="name"/>
        <choice>
            <ref name="music"/>
            <ref name="movie"/>
        </choice>
    </define>
    <define name="music">
        <attribute name="type">
            <value>music</value>
        </attribute>
        <choice>
            <element name="artist">
                <text/>
            </element>
            <element name="group">
                <text/>
            </element>
        </choice>
        <oneOrMore>
            <element name="title">
                <attribute name="name"/>
                <attribute name="length"/>
            </element>
        </oneOrMore>
        <element name="publisher">
            <text/>
        </element>
        <attribute name="category">
            <choice>
                <value>Pop</value>
                <value>Rock</value>
                <value>R&amp;B</value>
                <value>Classic</value>
                <value>Folk</value>
            </choice>
        </attribute>
        <element name="publishingDate">
            <text/>
        </element>
    </define>
    <define name="movie"
      xmlns:x="http://www.w3.org/1999/xhtml">
        <x:h1>Movie Definition</x:h1>
        <attribute name="type">
            <value>movie</value>
        </attribute>
        <element name="directedBy">
            <text/>
        </element>
        <oneOrMore>
            <element name="actor">
                <attribute name="name"/>
            </element>
        </oneOrMore>
        <element name="publisher">
            <text/>
        </element>
        <element name="fsk">
            <text/>
        </element>
        <attribute name="category">
            <choice>
                <value>Horror</value>
                <value>Thriller</value>
                <value>Science Fiction</value>
                <value>Drama</value>
            </choice>
        </attribute>
        <element name="publishingDate">
            <text/>
        </element>
        <optional>
            <element name="description">
                <ref name="Inline.model"/>
            </element>
        </optional>
    </define>
    <include
      href="xhtml-attribs-2.rng"
      ns="http://www.w3.org/2002/06/xhtml2"/>
    <include
      href="xhtml-inltext-2.rng"
      ns="http://www.w3.org/2002/06/xhtml2"/>
    <include
      href="xhtml-datatypes-2.rng"
      ns="http://www.w3.org/2002/06/xhtml2"/>
</grammar>
```

Wie eine Beschreibung in XHTML im Instanzdokument aussieht, zeigt Listing 4. Damit ist die Schemadefinition für den Musikladen abgeschlossen. Das komplette Schema enthält Listing 2.

Kompakte Syntax gegen Wortschwall

Trotz der Einfachheit besteht bei Relax NG wie bei allen XML-Dokumenten die Gefahr, dass man vor lauter Tags den Inhalt nicht sieht. Recht früh, nach dem Einreichen der Relax-NG-Spezifikation bei OASIS, entstand Anfang 2003 eine weitere Spezifikation für eine kompaktere Syntax. Die Schreibweise erinnert ein wenig an DTDs oder an Java-Klassendefinitionen. Die Grundstruktur des Musikladens sieht in kompakter Schreibweise so aus:

```
element musicstore {
  attribute name { text },
  (element dvd { content }* &
    element cd { content }*)
}
```

Wie bei regulären Ausdrücken bedeutet der Stern, dass es sich um 0 bis n Elemente handeln kann. Ein Pluszeichen steht für 1 bis n Elemente und das Ampersand anstelle des Kommas für das *interleave*-Pattern (unsortierte Elemente). Runde Klammern bezeichnen eine Gruppierung von Elementen, und die Pipe definiert eine Auswahl:

```
(element artist { text } | element
group { text })
```

Wer die kompakte Syntax verwendet, tauscht den Vorteil, die Schemadefinition in XML zu haben, gegen eine einfachere Schreibweise. Glücklicherweise gibt es aber Tools, die ein Relax-NG-Schema in kompakter Schreibweise (kurz: RNC) in ein Relax-NG-Schema in XML-Struktur (kurz: RNG) umwandeln. Sie können üblicherweise außerdem RNG in RNC transferieren. James Clarks Open-Source-Tool *trang* (siehe „Online-Ressourcen“) bietet eine Vielzahl von Konvertierungsmöglichkeiten. Der Aufruf, um das RNG-Schema in Listing 2 in ein RNC-Schema zu konvertieren:

```
java -jar trang.jar store.rng store.rnc
```

trang kann auf die gleiche Weise ein XML-Schema nach der W3C-Spezifikation erzeugen:

```
java -jar trang.jar store.rng store.xsd
```

Listing 3: Musikladen in kompakter Syntax

```
default namespace = "http://innoq.com/rng/musicstore"
namespace ns1 = "http://www.w3.org/2002/06/xhtml2"
namespace x = "http://www.w3.org/1999/xhtml"

start =
    element musicstore {
        attribute name { xsd:token },
        (element dvd { content }*
         & element cd { content }*)
    }
content =
    attribute name { xsd:token },
    (music | movie)
music =
    attribute type { "music" },
    (element artist { xsd:token }
     | element group { xsd:token }),
    element title {
        attribute name { xsd:token },
        attribute length { xsd:duration }
    }+,
    element publisher { xsd:token },
    attribute category { "Pop" | "Rock" |
                         "R&B" | "Classic" | "Folk" },
    element publishingDate { xsd:date {
            pattern = "[0-9]{4}-[0-9]{2}-[0-9]{2}"
        }
    }
[ x:h1 [ "Movie Definition" ] ]
movie =
    attribute type { "movie" },
    element directedBy { xsd:token },
    element actor {
        attribute name { xsd:token }
    }+,
    element publisher { xsd:token },
    element fsk { xsd:unsignedByte },
    attribute category { "Horror" | "Thriller" |
                         "Science Fiction" | "Drama" },
    element publishingDate { xsd:date }
    element description { Inline.model }?
include "xhtml-attribs-2rnc" inherit = ns1
include "xhtml-inltext-2rnc" inherit = ns1
include "xhtml-datatypes-2rnc" inherit = ns1
```

Nur zwei Basis-Datentypen

Grundsätzlich unterscheidet Relax NG zwischen der Validierung der Struktur und der Validierung des Inhalts eines XML-Dokuments. Während beim Design von Relax NG der Schwerpunkt auf der strukturellen Validierung lag, überlässt es die inhaltliche Validierung ande-

Listing 4: Ein Instanzdokument des Musikladens

```
<?xml version="1.0" encoding="UTF-8"?>
<musicstore name="MyTunes"
  xmlns="http://innoq.com/rng/musicstore"
  xmlns:x="http://www.w3.org/2002/06/xhtml2">
    <cd name="So-Called Chaos" category="Pop" type="music">
        <artist>Alanis Morissette</artist>
        <title name="Eight Easy Steps" length="PT2M50S"/>
        <title name="Out is Through" length="PT3M52S"/>
        <title name="Excuses" length="PT3M31S"/>
        <publisher>Maverick</publisher>
        <publishingDate>2005-05-18</publishingDate>
    </cd>
    <dvd name="Last Samurai" category="Drama" type="movie">
        <directedBy>Edward Zwick</directedBy>
        <actor name="Tom Cruise"/>
        <actor name="Timothy Spall"/>
        <publisher>Warner Home Video</publisher>
        <fsk>16</fsk>
        <publishingDate>2005-05-18</publishingDate>
        <description>Traumatisierter Amerikaner lernt
          <x:em>japanischen Schwertkampf</x:em>.
        </description>
    </dvd>
    <cd name="Guns N'Roses: Greatest Hits" category="Rock" type="music">
        <group>Guns N'Roses</group>
        <title name="Paradise City" length="PT6M46S"/>
        <title name="November Rain" length="PT8M56S"/>
        <publisher>Geffen</publisher>
        <publishingDate>2004-03-15</publishingDate>
    </cd>
</musicstore>
```

ren. Zwar gibt es, wie erwähnt, zwei native Datentypen (*token* und *string*), die aber für eine inhaltliche Validierung in vielen Fällen nicht ausreichen. In diesen Fällen kann man auf externe Typsysteme zurückgreifen.

Relax NG sieht vor, generische, Domain- oder programmiersprachenspezifische Typsysteme zu verwenden. Implementierungen müssen nur die nativen Datentypen unterstützen, alle weiteren sind optional. Allerdings kennen viele Implementierungen DTD- und XML-Schema-Typen, auf die man in der Praxis zurückgreift. Um das Listing um ein externes Typsystem zu erweitern, kann ein Autor schon im *start*-Pattern über das *datatypeLibrary*-Attribut das zu verwendende Typsystem angeben – hier das von XML Schema. Auf diese Weise können sämtliche Elemente diese Typen verwenden. Zusätzlich kann jedes Element eine eigene Datentypbibliothek referenzieren.

```
<start
datatypeLibrary="http://www.w3.
org/2001/XMLSchema-datatypes">
  <element name="musicstore">
    <attribute name="name">
      <data type="NMTOKEN"/>
    </attribute>
    ...
  </element>
</start>
```

Für die Berücksichtigung solcher Datentypen bietet es sich an, sie in der kompakten Schreibweise zu notieren. Hierbei kann man auf die aus der XML-Schemadefinition zugreifen, da das in kompakter Schreibweise implizit referenziert wird – über das Präfix *xsd*.

```
attribute name { xsd:token },
```

In der Schemadefinition (siehe Listing 3) kommt häufig der Typ *xsd:token* vor. Dieser entspricht einem String mit Leerzeichen, aber ohne Zeilenumbrüche. Für das *publishingDate*-Element kommt der Typ *xsd:date* zum Einsatz, den eine so genannte *pattern*-Facette einschränkt. Die erlaubt die Verwendung beliebiger regulärer Ausdrücke. Das hier gültige Datum soll das Format YYYY-MM-DD haben, wie das Ende von Listing 3 zeigt.

Für die Validierung von XML-Dokumenten (ob sie einem Schema entsprechen) existiert eine ganze Reihe von Tools. Auf der Website von Relax NG befindet sich eine Liste mit Validierern (siehe „Online-Ressourcen“). So lässt sich etwa mit dem von James Clark entwickelten *jing* ein XML-Dokument (siehe Listing 4) gegen ein Schema wie Listing 2 oder 3 validieren:

```
java -jar jing.jar store.rng MyTunes.xml
```

Soll das Schema in kompakter Schreibweise verwendet werden, muss der Aufruf die Option *-c* (für compact) enthalten:

```
java -jar jing.jar -c store.rnc MyTunes.xml
```

Beide Schemata sollten das Instanzdokument (siehe Listing 4) validieren. Darüber hinaus lässt sich ein RNG-Schema ebenfalls validieren. Das RNG-Schema für Relax NG befindet sich in der Spezifikation. In einer Datei *rng.rng* gespeichert, muss man das Kommando lediglich leicht umformulieren: *java -jar jing.jar rng.rng store.rng*.

Fazit

Im Moment spielt Relax NG die Rolle des begabten Außenseiters; dass sich dies in naher Zukunft ändern könnte, ist keineswegs sicher. Relax NG genießt zwar sogar beim W3C einen so guten Ruf, dass das Konsortium es beispielsweise bei der noch in Arbeit befindlichen WSDL-Spezifikation 2.0 oder in der XHTML2-Spezifikation als eine alternative Schema-Sprache verwendet. Die Aufnahme in eine Spezifikation garantiert allerdings nicht, dass auf dem Markt verfügbare Werkzeuge diese zusätzliche Option unterstützen.

Die Vorteile des einfacheren, klaren Modells und der Compact Syntax, die das Erstellen von Schema-Definitionen „von Hand“ perfekt unterstützt, sind jedoch so groß, dass man Relax NG definitiv in Erwägung ziehen sollte, wenn die Rahmenbedingungen es ermöglichen. Und schließlich kann man immer noch eine RNC-Syntax-Definition automatisch in ein XML-Schema übersetzen – dass man so verfahren ist, muss man ja niemandem erzählen. (hb)

MARCEL TILLY

ist Senior Consultant bei der innoQ Deutschland GmbH.

STEFAN TILKOV

ist Geschäftsführer der innoQ Deutschland GmbH.

Literatur

[1] Nik Klever; XML; Elementarteilchen; XML-Schema: objektorientierte Dokumenttypdefinitionen; *iX* 6/2001, S. 62

ONLINE-RESSOURCEN

Relax NG Homepage	www.relaxng.org
Relax NG Spezifikation	www.relaxng.org/spec-20011203.html
XHTML Relax NG Module	www.w3.org/TR/2003/WD-xhtml2-20030506/relax_module_defs.html#a_relaxng_module_defs
Validierer	
James Clarks Jing (Java)	www.thaiopensource.com/relaxng/jing.html
Suns Multi-Schema XML Validator (MSV)	wwws.sun.com/software/xml/developers/multischema/
Konvertierer	
RNG2RNC; XSLT-Stylesheet konvertiert von RNG nach RNC	www.pantor.com/download.html
RNC2RNG; Konverter von RNC nach RNG	www.gnosis.cx/download/relax/
James Clarks Trang; Multi-format-Schema-Konverter	thaiopensource.com/relaxng/trang.html