A Comparison between MSL and The Essence of XML

Erik Knoop

January 13th 2004

1 Introduction

In this article we will match a XML document against XML Schema. That is, we will give the representation of such a document according to the standards described in [**?**]. After that we will give represent the same document according to the standards described in [**?**]. Finally, we will conclude their differences.

But first we will introduce, in the syntax of [**?**], the running example.

2 Running example

The example XML Schema is defined below.

```
define element configuration of type configurationType
define type configurationType {
 element shuttle of type shuttleType,
 element laser of type laserType
}
define type shuttleType {
 element height of type miles
}
define type laserType {
 element height of type miles
}
define type miles restricts xs:integer
```

The example document is defined below.

```
element configuration {
 element shuttle {
  element height { 120 }
 },
 element laser {
  element height { 10023 }
 }
}
```

3 Matching like The Essence of XML

In this section we show a derivation in the style of [**?**]. We start with the main judgement in figure 1, and from there, other figures are referenced.

```
 configuration
yields                                                  (figure 2)
 configuration of type configurationType

 configurationType
resolves to                                             (figure 2)
 element shuttle of type shuttleType,
 element laser of type laserType

 configuration
substitutes for                                         (figure 2)
 configuration

 configurationType
derives from                                            (figure 2)
 configurationType

 element shuttle of type shuttleType {
  element height of type miles { 120 }
 },
 element laser of type laserType {
  element height of type miles { 10023 }
 }
matches                                                 (figure 2)
 element shuttle of type shuttleType,
 element laser of type laserType
```

```
 element configuration of type configurationType {
  element shuttle of type shuttleType {
   element height of type miles { 120 }
  },
  element laser of type laserType {
   element height of type miles { 10023 }
  }
 }
matches
  element configuration
```

Figure 1: Matching (part 1)

```
 define element configuration of type configurationType
```

```
 configuration
yields
configuration of type configurationType


 define type configurationType {
  element shuttle of type shuttleType,
  element laser of type laserType
 }
```

```
 configurationType
resolves to
 element shuttle of type shuttleType,
 element laser of type laserType
```

```
 configuration
substitutes for
 configuration
```

```
 configurationType
derives from
 configurationType


 element shuttle of type shuttleType {
  element height of type miles { 120 }
 }
matches                                                (figure 3)
 element shuttle of type shuttleType

 element laser of type laserType {
  element height of type miles { 10023 }
 }
matches                                                (figure 6)
 element laser of type laserType
```

```
 element shuttle of type shuttleType {
  element height of type miles { 120 }
 },
 element laser of type laserType {
  element height of type miles { 10023 }
 }
matches
 element shuttle of type shuttleType,
 element laser of type laserType
```

Figure 2: Matching (part 2)

element shuttle of type shuttleType
yields (figure 4)
element shuttle of type shuttleType

shuttleType
resolves to (figure 4)
element of height of type miles

shuttle
substitutes for (figure 4)
shuttle

shuttleType
derives from (figure 4)
shuttleType

element height of type miles { 120 }
matches (figure 4)
element height of type miles

---

element shuttle of type shuttleType {
element height of type miles { 120 }
}
matches
element shuttle of type shuttleType

Figure 3: Matching (part 3)

element shuttle of type shuttleType
yields
element shuttle of type shuttleType

define type shuttleType {
element height of type miles
}

---

shuttleType
resolves to
element height of type miles

---

shuttle
substitutes for
shuttle

---

shuttleType derives from shuttleType

element height of type miles
yields (figure 5)
element height of type miles

miles
resolves to (figure 5)
xs:integer

height
substitutes for (figure 5)
height

miles
derives from (figure 5)
miles

120
matches (figure 5)
xs:integer

---

element height of type miles { 120 }
matches
element height of type miles

Figure 4: Matching (part 4)

element height of type miles
yields
element height of type miles

define type miles restricts xs:integer

---

miles
resolves to
xs:integer

---

height
substitutes for
height

---

miles
derives from
miles

xs:integer
derives from
xs:integer

120
matches
xs:integer

---

xs:integer
derives from
xs:integer

Figure 5: Matching (part 5)

element laser of type laserType
yields (figure 20)
element laser of type laserType

laserType
resolves to (figure 7)
element of height of type miles

laser
substitutes for (figure 7)
laser

laserType
derives from (figure 7)
laserType

element height of type miles { 10023 }
matches (figure 7)
element height of type miles

---

element laser of type laserType {
element height of type miles { 10023 }
}
matches
element laser of type laserType

Figure 6: Matching (part 6)

element laser of type laserType
yields
element laser of type laserType

define type laserType {
element height of type miles
}

---

laserType
resolves to
element height of type miles

---

laser
substitutes for
laser

---

laserType derives from laserType

element height of type miles
yields (figure 8)
element height of type miles

miles
resolves to (figure 8)
xs:integer

height
substitutes for (figure 8)
height

miles
derives from (figure 8)
miles

10023
matches (figure 8)
xs:integer

---

element height of type miles { 10023 }
matches
element height of type miles

Figure 7: Matching (part 7)

element height of type miles
yields
element height of type miles

define type miles restricts xs:integer

---

miles
resolves to
xs:integer

---

height
substitutes for
height

---

miles
derives from
miles

xs:integer
derives from
xs:integer

10023
matches
xs:integer

---

xs:integer
derives from
xs:integer

Figure 8: Matching (part 8)

4 Running Example (revised)

In this section we give the same example as in section 2, but now in the notation of [**?**].

```
component(
 sort = element
 name = configuration
 base = xs:UrElement
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = configuration[configurationType]
)

component(
 sort = complexType
 name = configurationType
 base = xs:UrType
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = shuttle, laser
)

component(
 sort = element
 name = shuttle
 base = xs:UrElement
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = shuttle[shuttleType]
)

component(
 sort = complexType
 name = shuttleType
 base = xs:UrType
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = height
)
```

```
component(
 sort = element
 name = laser
 base = xs:UrElement
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = laser[laserType]
)

component(
 sort = complexType
 name = laserType
 base = xs:UrType
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = height
)

component(
 sort = element
 name = height
 base = xs:UrElement
 derivation = restriction
 refinement = {restriction,extension}
 abstract = false
 content = height[miles]
)

component(
 sort = simpleType
 name = miles
 base = xs:UrSimpleType
 derivation = restriction
 refinement = {restriction}
 abstract = false
 content = xs:integer{0, ∞}
)
```

And the example document:

```
configuration[
 shuttle[
  height[120]
 ],
 laser[
  height[10023]
 ]
]
```

5 Normalisation and Validation in MSL

$\vdash$ configuration $\Rightarrow$ configuration

xs:type $\notin$
shuttle[
 height[120]
],
laser[
 height[10023]
]

deref(configuration).content = configuration[configurationType]

configurationType $\vdash$
shuttle[
 height[120]
],
laser[
 height[10023]
] $\Rightarrow$
shuttle[
 shuttleType
 $\ni$
 height[miles $\ni$ 120]
],
laser[
 laserType
 $\ni$
 height[miles $\ni$ 10023]
]

$\vdash$ configuration[
 shuttle[
  height[120]
 ],
 laser[
  height[10023]
 ]
] $\Rightarrow$
configuration[
 configurationType
 $\ni$
 shuttle[
  shuttleType
  $\ni$
  height[miles $\ni$ 120]
 ],
 laser[
  laserType
  $\ni$
  height[miles $\ni$ 10023]
 ]

Figure 9: Normalisation in MSL (part 1)

configurationType
$\vdash$
shuttle[
 height[120]
]
$\Rightarrow$
shuttle[
 shuttleType
 $\ni$
 height[miles $\ni$ 120]
]

configurationType
$\vdash$
laser[
 height[10023]
]
$\Rightarrow$
laser[
 laserType
 $\ni$
 height[miles $\ni$ 10023]
]

---

configurationType
$\vdash$
shuttle[
 height[120]
],
laser[
 height[10023]
]
$\Rightarrow$
shuttle[
 shuttleType
 $\ni$
 height[miles $\ni$ 120]
],
laser[
 laserType
 $\ni$
 height[miles $\ni$ 10023]
]

Figure 10: Normalisation in MSL (part 2)

$$\frac{\begin{array}{l}\text{configurationType} \vdash \text{shuttle} \Rightarrow \text{shuttle} \\ \\ \text{xs:type} \notin \text{height}[120] \\ \\ \text{deref(shuttle).content} = \text{shuttle[shuttleType]} \\ \\ \text{shuttleType} \vdash \text{height}[120] \Rightarrow \text{height[miles} \ni 120]\end{array}}{\begin{array}{l}\text{configurationType} \\ \vdash \\ \text{shuttle[} \\ \ \text{height}[120] \\ ] \\ \Rightarrow \\ \text{shuttle[} \\ \ \text{shuttleType} \\ \ \ni \\ \ \text{height[miles} \ni 120] \\ ]\end{array}}$$

$$\frac{\begin{array}{l}\text{shuttleType} \vdash \text{height} \Rightarrow \text{height} \\ \\ \text{xs:type} \notin 120 \\ \\ \text{deref(height).content} = \text{height[miles]} \\ \\ \text{miles} \vdash 120 \Rightarrow 120\end{array}}{\text{shuttleType} \vdash \text{height}[120] \Rightarrow \text{height[miles} \ni 120]}$$

$$\frac{}{\text{miles} \vdash 120 \Rightarrow 120}$$

Figure 11: Normalisation in MSL (part 3)

$$\frac{\begin{array}{l}\text{configurationType} \vdash \text{laser} \Rightarrow \text{laser} \\ \\ \text{xs:type} \notin \text{height}[10023] \\ \\ \text{deref(laser).content} = \text{laser}[\text{laserType}] \\ \\ \text{laserType} \vdash \text{height}[10023] \Rightarrow \text{height}[\text{miles} \ni 10023]\end{array}}{\begin{array}{l}\text{configurationType} \\ \vdash \\ \text{laser}[\\ \ \text{height}[10023] \\ ] \\ \Rightarrow \\ \text{laser}[\\ \ \text{laserType} \\ \ \ni \\ \ \text{height}[\text{miles} \ni 10023] \\ ]\end{array}}$$

$$\frac{\begin{array}{l}\text{laserType} \vdash \text{height} \Rightarrow \text{height} \\ \\ \text{xs:type} \notin 10023 \\ \\ \text{deref(height).content} = \text{height}[\text{miles}] \\ \\ \text{miles} \vdash 10023 \Rightarrow 10023\end{array}}{\text{laserType} \vdash \text{height}[10023] \Rightarrow \text{height}[\text{miles} \ni 10023]}$$

$$\frac{}{\text{miles} \vdash 10023 \Rightarrow 10023}$$

Figure 12: Normalisation in MSL (part 4)

```
shuttle[
 shuttleType
 ∋
 height[miles ∋ 120]
],
laser[
 laserType
 ∋
 height[miles ∋ 10023]
]
∈
configurationType

configurationType <: configurationType
──────────────────────────────────────
configuration[
 configurationType
 ∋
 shuttle[
  shuttleType
  ∋
  height[miles ∋ 120]
 ],
 laser[
  laserType
  ∋
  height[miles ∋ 10023]
 ]
]
∈
configuration[configurationType]
```

Figure 13: Validation in MSL (part 1)

shuttle[
 shuttleType
 ∋
 height[miles ∋ 120]
],
laser[
 laserType
 ∋
 height[miles ∋ 10023]
]
∈
shuttle, laser

shuttle, laser = deref(configurationType).content

---

shuttle[
 shuttleType
 ∋
 height[miles ∋ 120]
],
laser[
 laserType
 ∋
 height[miles ∋ 10023]
]
∈
configurationType

Figure 14: Validation in MSL (part 2)

shuttle[
 shuttleType
 ∋
 height[miles ∋ 120]
]
∈
shuttle

laser[
 laserType
 ∋
 height[miles ∋ 10023]
]
∈
laser

---

shuttle[
 shuttleType
 ∋
 height[miles ∋ 120]
],
laser[
 laserType
 ∋
 height[miles ∋ 10023]
]
∈
shuttle, laser

height[miles ∋ 120]
∈
shuttleType

shuttleType <: shuttleType

---

shuttle[
 shuttleType
 ∋
 height[miles ∋ 120]
]
∈
shuttle[shuttleType]

height[miles ∋ 120]
∈
height

height = deref(shuttleType).content

---

height[miles ∋ 120]
∈
shuttleType

Figure 15: Validation in MSL (part 3)

$$\frac{\begin{array}{l}120 \in \text{miles} \\ \\ \text{miles} <: \text{miles}\end{array}}{\begin{array}{l}\text{height}[\text{miles} \ni 120] \\ \in \\ \text{height}[\text{miles}]\end{array}}$$

$$\frac{120 \in_p \text{miles}}{120 \in \text{miles}}$$

$$\begin{array}{l}\text{height}[\text{miles} \ni 10023] \\ \in \\ \text{laserType}\end{array}$$

$$\frac{\text{laserType} <: \text{laserType}}{\begin{array}{l}\text{laser}[\\ \ \text{laserType} \\ \ \ni \\ \ \text{height}[\text{miles} \ni 10023] \\ ] \\ \in \\ \text{laser}[\text{laserType}]\end{array}}$$

$$\begin{array}{l}\text{height}[\text{miles} \ni 10023] \\ \in \\ \text{height}\end{array}$$

$$\frac{\text{height} = \text{deref}(\text{laserType}).\text{content}}{\begin{array}{l}\text{height}[\text{miles} \ni 10023] \\ \in \\ \text{laserType}\end{array}}$$

$$\begin{array}{l}10023 \in \text{miles} \\ \\ \text{miles} <: \text{miles}\end{array}$$

$$\frac{\begin{array}{l}\text{height}[\text{miles} \ni 10023] \\ \in \\ \text{height}[\text{miles}]\end{array}}{\begin{array}{l}10023 \in_p \text{miles} \\ \\ 10023 \in \text{miles}\end{array}}$$

Figure 16: Validation in MSL (part 4)