An Algebraic Framework for Modeling and Processing Hyperdocuments

KPS '09, Maria Taferl

Volker Mattick

Universität Dortmund, Fakultät für Informatik, Lehrstuhl Logik in der Informatik
volker.mattick@cs.uni-dortmund.de

12. Oktober 2009

Einleitung

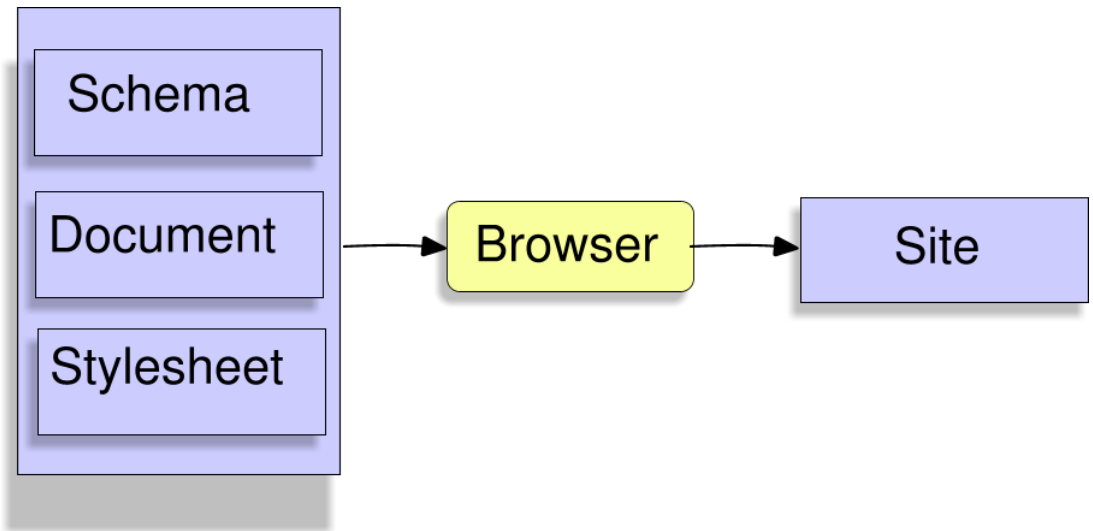
Motivation

- Modellierung von Hyperdokumenten (und Universal Design)
- Entwurf und Realisierung von Übersetzern mit algebraischen Methoden (Padawitz 2007 KPS, 2008 IFIP, 2009 in Vorbereitung)

Einleitung

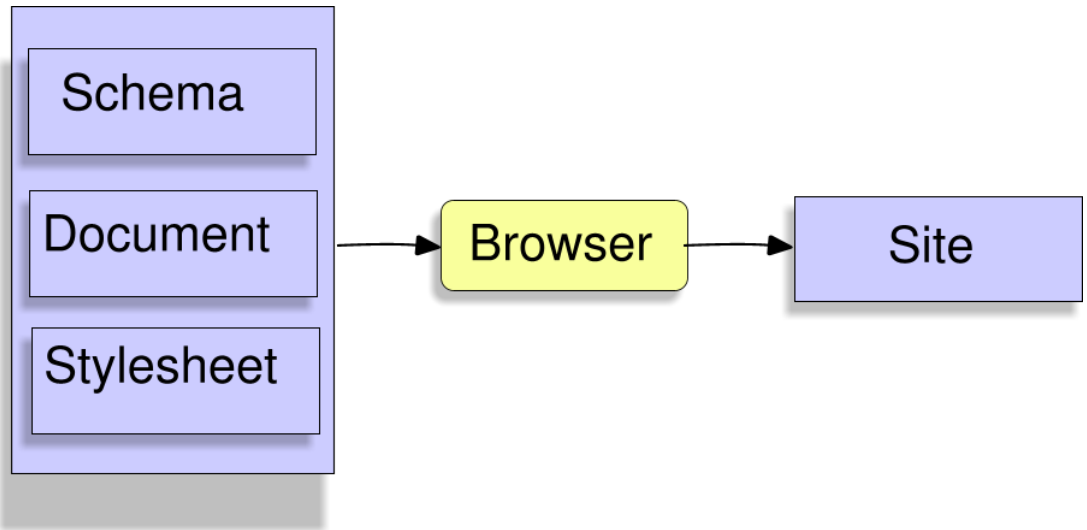
Motivation

- Modellierung von Hyperdokumenten (und Universal Design)
- Entwurf und Realisierung von Übersetzern mit algebraischen Methoden (Padawitz 2007 KPS, 2008 IFIP, 2009 in Vorbereitung)



Einleitung

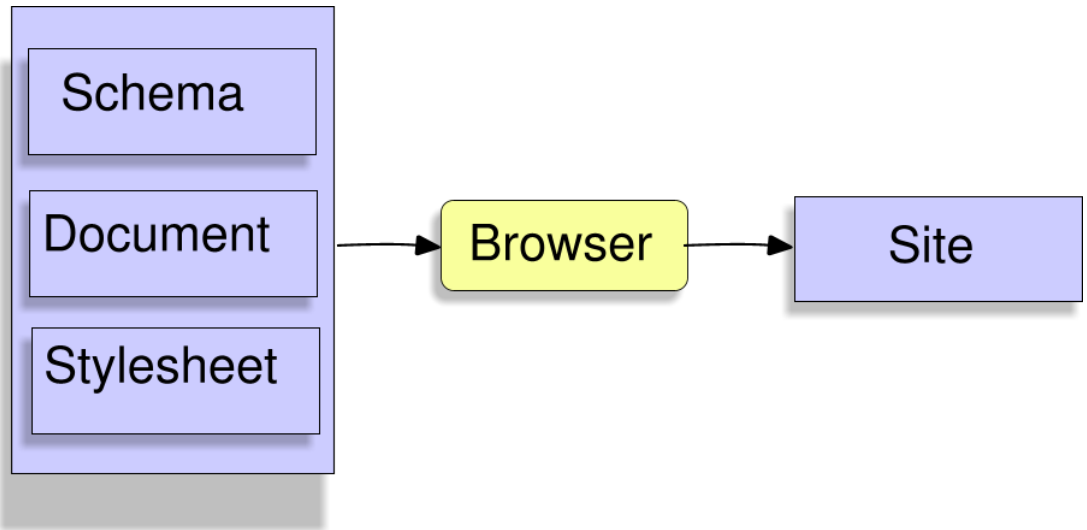
Motivation



- Typische Schemasprachen: DTD, XSD, RelaxNG

Einleitung

Motivation



- Typische Schemasprachen: DTD, XSD, RelaxNG
- Typische Dokumentsprachen: XHTML, SVG

Einleitung

Motivation

$G = (N, T, P, S)$ ist eine *reguläre Baumgrammatik*, wenn alle Regeln die Form $X \longrightarrow tr$ haben, wobei $X \in N$, $t \in T$ and $r \in N^*$. $L(G)$ heit dann *reguläre Baumsprache*.

Einleitung

Motivation

$G = (N, T, P, S)$ ist eine *reguläre Baumgrammatik*, wenn alle Regeln die Form $X \longrightarrow tr$ haben, wobei $X \in N$, $t \in T$ and $r \in N^*$. $L(G)$ heit dann *reguläre Baumsprache*.

Reguläre Baumsprachen $\subset$ Kontextfreien Sprachen

Einleitung

Motivation

$G = (N, T, P, S)$ ist eine *reguläre Baumgrammatik*, wenn alle Regeln die Form $X \longrightarrow tr$ haben, wobei $X \in N$, $t \in T$ and $r \in N^*$. $L(G)$ heit dann *reguläre Baumsprache*.

Reguläre Baumsprachen $\subset$ Kontextfreien Sprachen

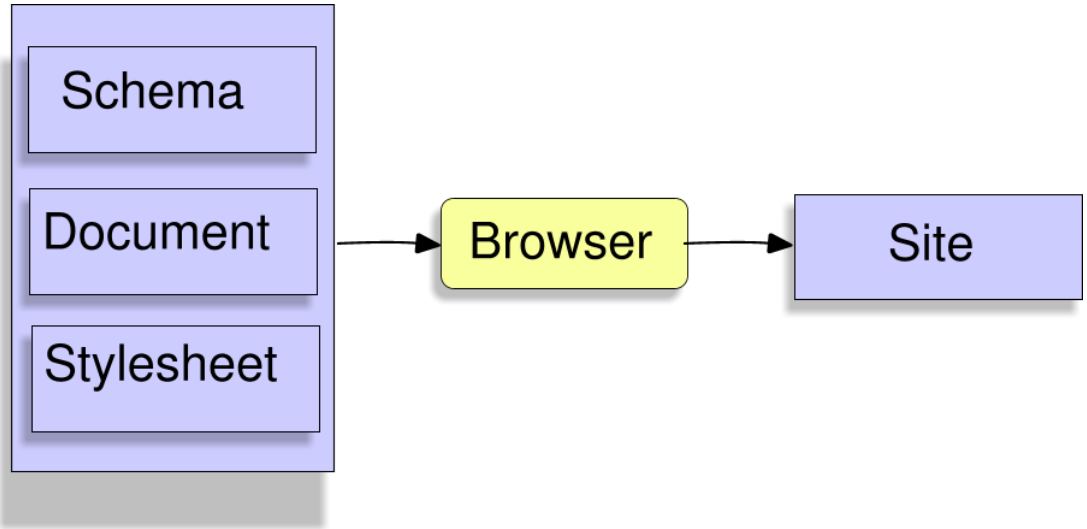
$G = (N, T, P, S)$ ist eine *balancierte Grammatik*, wenn

- $T = A \cup \overline{A}$, wobei $\overline{A}$ eine disjunkte Kopie von A ist, und
- alle Regeln die Form $X \longrightarrow t\ r\ \overline{t}$ with $r \in N^*$, $X \in N, t \in A, \overline{t} \in \overline{A}$ oder $X \longrightarrow \epsilon$ haben.

$L(G)$ heit dann *balancierte Sprache*.

Einleitung

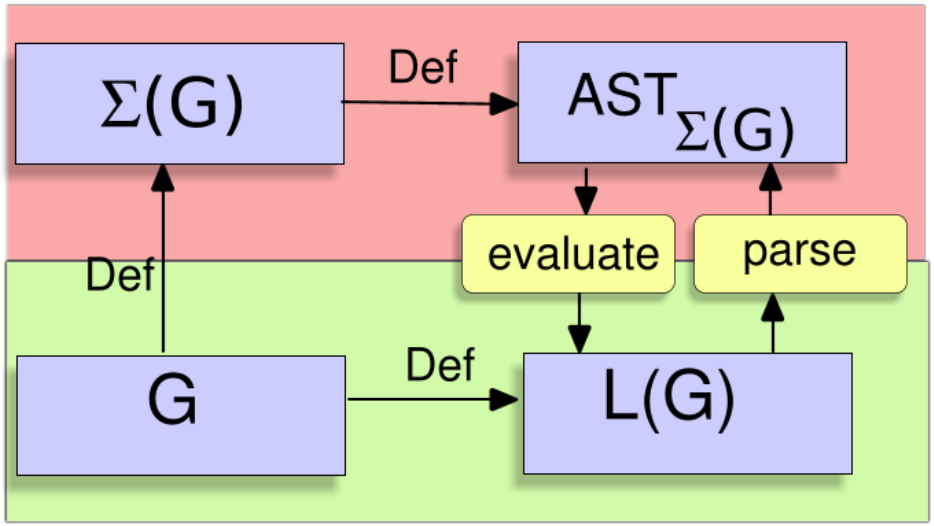
Motivation



- Typische Schemasprachen: DTD, XSD, RelaxNG
- Typische Dokumentsprachen: XHTML, SVG
- Typische Stylesheetsprachen: CSS, XSLT

Algebraic Framework

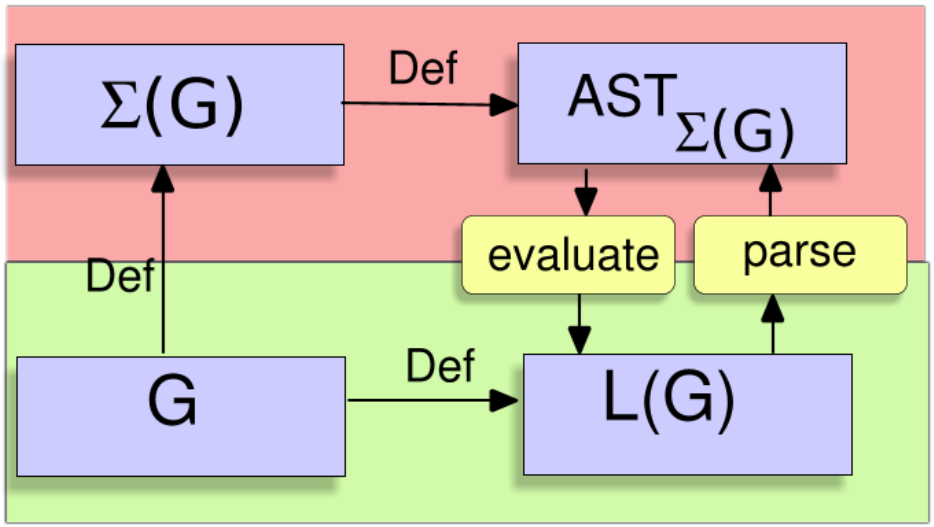
Konkrete und abstrakte Syntax



$G = (N, T, P, S)$ ist eine kontextfreie Grammatik.

Algebraic Framework

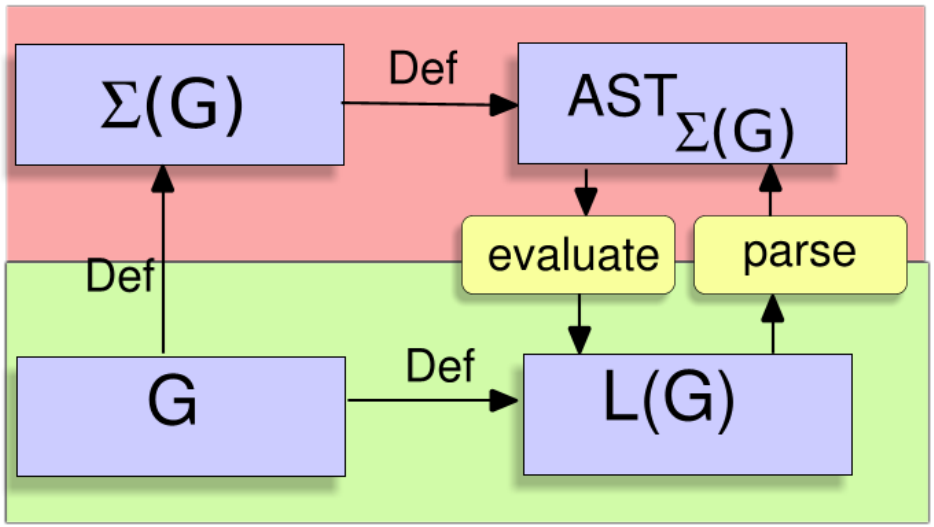
Konkrete und abstrakte Syntax



$G = (N, T, P, S)$ ist eine kontextfreie Grammatik.
$\Sigma(G) = (N, CO)$ ist die Signatur zu G mit:
$CO = \{c_p : X_1 \times \ldots \times X_n \to X \mid \exists : p = (X \to w_1 X_1 w_2 \ldots w_n X_n w_{n+1}) \in P,$
$w_i \in T^*, 1 \leq i \leq n+1, X_j \in N, 1 \leq j \leq n\}.$

Algebraic Framework

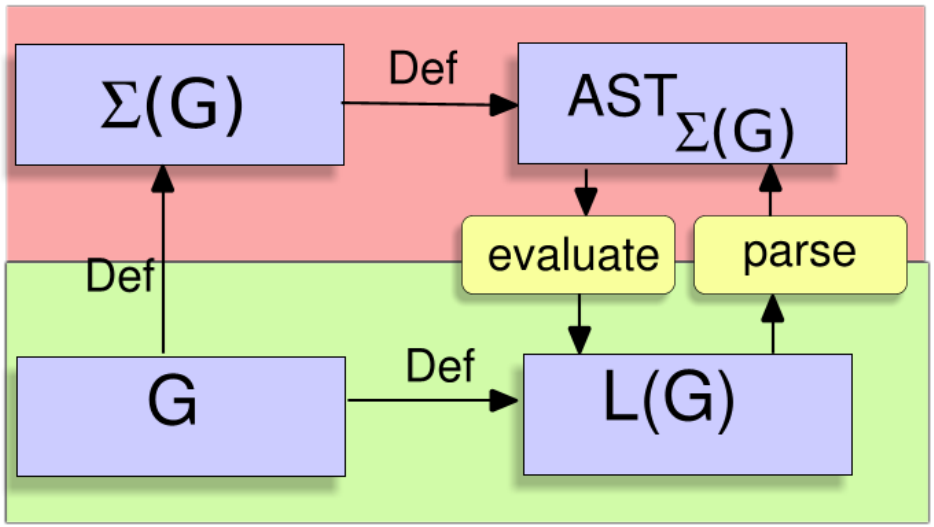
Konkrete und abstrakte Syntax



$G = (N, T, P, S)$ ist eine kontextfreie Grammatik.
$\Sigma(G) = (N, CO)$ ist die Signatur zu G mit:
$CO = \{c_p : X_1 \times \ldots \times X_n \to X \mid \exists : p = (X \to w_1 X_1 w_2 \ldots w_n X_n w_{n+1}) \in P,$
$w_i \in T^*, 1 \leq i \leq n+1, X_j \in N, 1 \leq j \leq n\}$.
$AST_{\Sigma(G)}$ ist die Menge der abstrakten Syntaxbäume.

Algebraic Framework

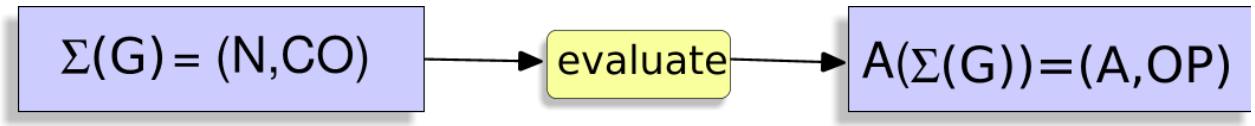
Konkrete und abstrakte Syntax



$G = (N, T, P, S)$ ist eine kontextfreie Grammatik.
$\Sigma(G) = (N, CO)$ ist die Signatur zu G mit:
$CO = \{c_p : X_1 \times \ldots \times X_n \to X \mid \exists : p = (X \to w_1 X_1 w_2 \ldots w_n X_n w_{n+1}) \in P,$
$w_i \in T^*, 1 \leq i \leq n+1, X_j \in N, 1 \leq j \leq n\}$.
$AST_{\Sigma(G)}$ ist die Menge der abstrakten Syntaxbäume.
Ein Wort aus $L(G)$ wird repräsentiert durch einen abstrakten Syntaxbaum.

Algebraic Framework

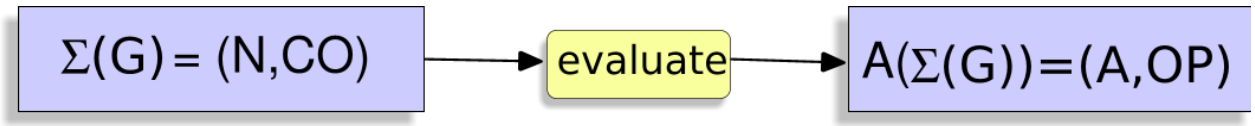
Konkrete und abstrakte Syntax



$A(\Sigma(G)) = (A, OP)$ ist eine $\Sigma(G)$-Struktur, wobei
$\forall : s \in N$ ex. genau eine Menge $A_s \in A$,
$\forall : (co : s) \in CO$ ex. genau eine Funktion $co^A \in A_s$ und
$\forall : (co : e \to s) \in CO$ ex. genau eine Funktion $co^A : A_e \to A_s \in OP$.

Algebraic Framework

Konkrete und abstrakte Syntax

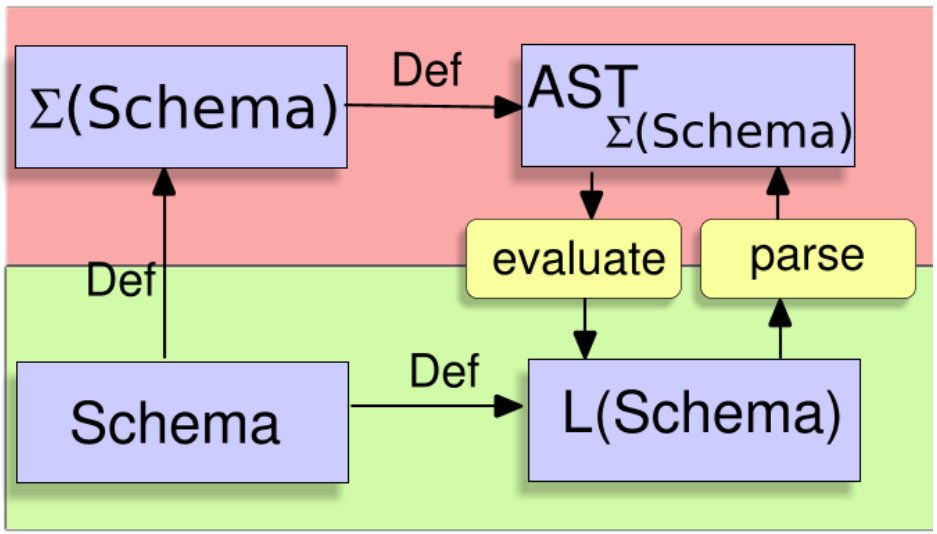


$A(\Sigma(G)) = (A, OP)$ ist eine $\Sigma(G)$-Struktur, wobei
$\forall : s \in N$ ex. genau eine Menge $A_s \in A$,
$\forall : (co : s) \in CO$ ex. genau eine Funktion $co^A \in A_s$ und
$\forall : (co : e \to s) \in CO$ ex. genau eine Funktion $co^A : A_e \to A_s \in OP$.

Diese Struktur ist eine Algebra.

Algebraic Framework

Konkrete und abstrakte Syntax



Algebraic Framework

Schema

```
<xsd:element name="gpx" type="gpxType"/>
<xsd:complexType name="gpxType">
<xsd:sequence>
 <xsd:element name="metadata" type="metadataType" minOccurs="0"/>
 <xsd:element name="wpt" type="wptType"
              minOccurs="0" maxOccurs="unbounded"/>
 <xsd:element name="rte" type="rteType"
              minOccurs="0" maxOccurs="unbounded"/>
 <xsd:element name="trk" type="trkType"
              minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="version" type="xsd:string"
               use="required" fixed="MiniGPX"/>
<xsd:attribute name="creator" type="xsd:string" use="required"/>
</xsd:complexType>
```

Algebraic Framework

Abstrakte Syntax und Implementierung

$$
\begin{aligned}
CO = \{ & \texttt{root\_} :: \texttt{gpxType\_avList} \times \texttt{gpxType} \rightarrow \texttt{gpx} \\
& \texttt{gpxType\_} :: \texttt{metadataAlt} \times \texttt{wptList} \times \texttt{rteList} \\
& \qquad \times \texttt{trkList} \rightarrow \texttt{gpxType} \\
& \texttt{gpx\_MiniGPX} :: \rightarrow \texttt{gpxType\_av} \\
& \texttt{gpx\_creator} :: \texttt{xsd:string} \rightarrow \texttt{gpxType\_av} \\
& \texttt{\_ : \_} :: \texttt{gpxType\_av} \times \texttt{gpxType\_avList} \rightarrow \texttt{gpxType\_avList} \\
& [] :: \rightarrow \texttt{gpxType\_avList} \\
& \ldots \}
\end{aligned}
$$

Algebraic Framework

Abstrakte Syntax und Implementierung

$$
\begin{aligned}
CO = \{&\mathsf{root\_} :: \mathtt{gpxType\_avList} \times \mathtt{gpxType} \to \mathtt{gpx} \\
&\mathsf{gpxType\_} :: \mathtt{metadataAlt} \times \mathtt{wptList} \times \mathtt{rteList} \\
&\qquad\qquad \times \mathtt{trkList} \to \mathtt{gpxType} \\
&\mathsf{gpx\_MiniGPX} :: \to \mathtt{gpxType\_av} \\
&\mathsf{gpx\_creator} :: \mathtt{xsd{:}string} \to \mathtt{gpxType\_av} \\
&\_ : \_ :: \mathtt{gpxType\_av} \times \mathtt{gpxType\_avList} \to \mathtt{gpxType\_avList} \\
&[] :: \to \mathtt{gpxType\_avList} \\
&\ldots\}
\end{aligned}
$$

```
data GpxSIG gpx gpxType gpxType_av ... =
     GpxSIG {root_mt :: [gpxType_av] → gpx,
             root_ :: [gpxType_av] → gpxType → gpx,
             gpxType_ :: (Maybe metadata) → [wpt] → [rte]
                                          → [trk] → gpxType,
             gpx_MiniGPX :: gpxType_av,
             gpx_creator :: XSD_string → gpxType_av, ... }
```

Algebraic Framework

Struktur

$$
\begin{aligned}
OP = \{ & \text{root\_} :: \text{avList} \times \text{String} \to \text{String} \\
& \text{gpxType\_} :: (\text{Maybe String}) \times [\text{String}] \times [\text{String}] \\
& \qquad \times [\text{String}] \to \text{String} \\
& \text{gpx\_MiniGPX} :: \to \text{av} \\
& \text{gpx\_creator} :: \text{String} \to \text{av} \\
& \text{\_ : \_} :: \text{String} \times [\text{String}] \to \text{avList} \\
& [] :: \to \text{avList} \\
& \ldots \}
\end{aligned}
$$

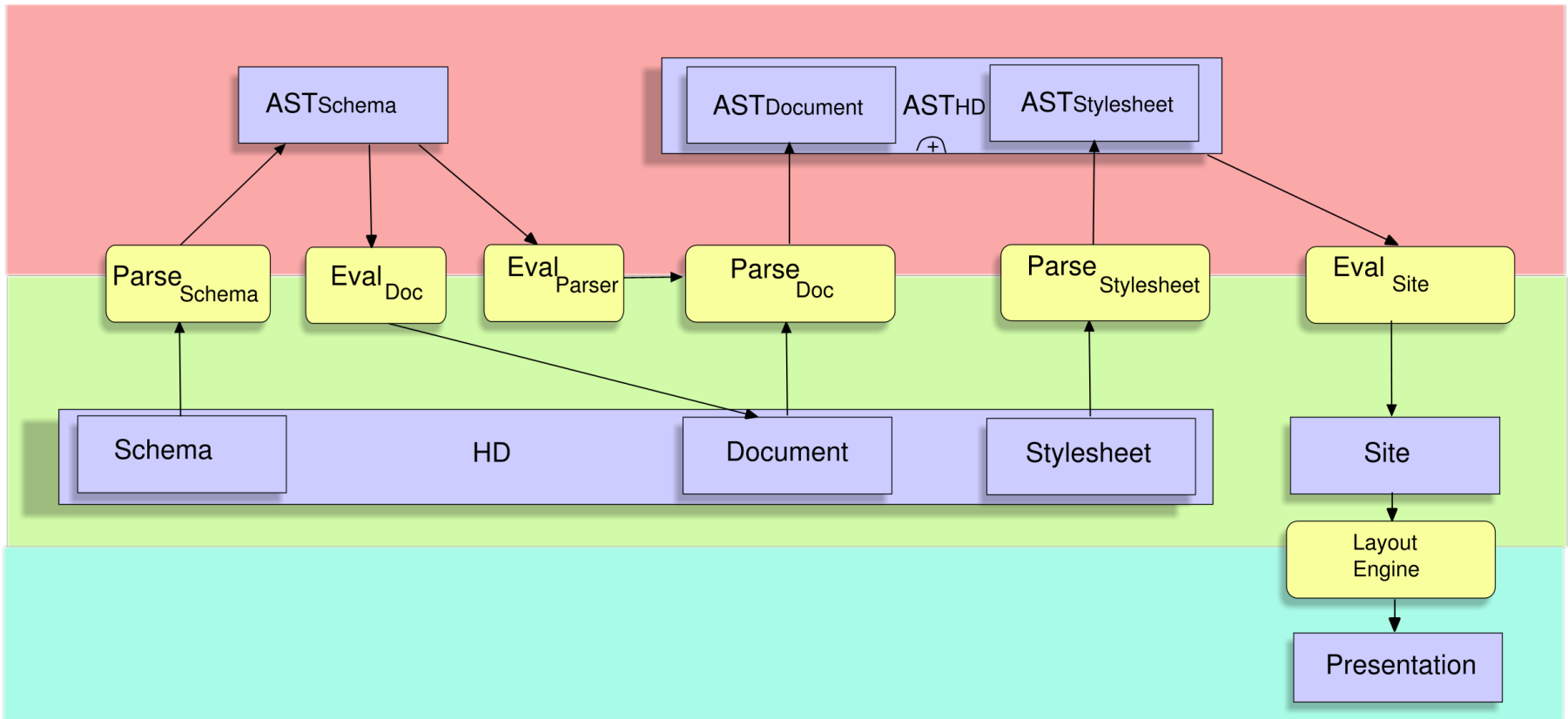
Algebraic Framework

Struktur

$$
\begin{aligned}
OP = \{&\texttt{root\_} :: \text{avList} \times \text{String} \to \text{String} \\
&\texttt{gpxType\_} :: (\text{Maybe String}) \times [\text{String}] \times [\text{String}] \\
&\qquad\qquad \times [\text{String}] \to \text{String} \\
&\texttt{gpx\_MiniGPX} :: \to \text{av} \\
&\texttt{gpx\_creator} :: \text{String} \to \text{av} \\
&\texttt{\_ : \_} :: \text{String} \times [\text{String}] \to \text{avList} \\
&[] :: \to \text{avList} \\
&\ldots\}
\end{aligned}
$$

```
svgALG :: GpxSIG String, String ...
svgALG = GpxSIG root_mt root_ gpxType_ gpx_MiniGPX gpx_creator...
  where root_ avlist gpxType = "<svg
           xmlns=http://www.w3.org/2000/svg"
         ++ gpxType ++ "</svg>"
       gpxType_ metadata wpt rte trk = (conc wpt) ++
              (conc rte) ++ (conc trk)
       gpx_MiniGPX = "MiniGPX"
       gpx_creator c = c
       ...
```

Algebraic Framework



Ende

Vielen Dank!

Fragen? Anregungen? Kritik?

Algebraic Framework

Editor