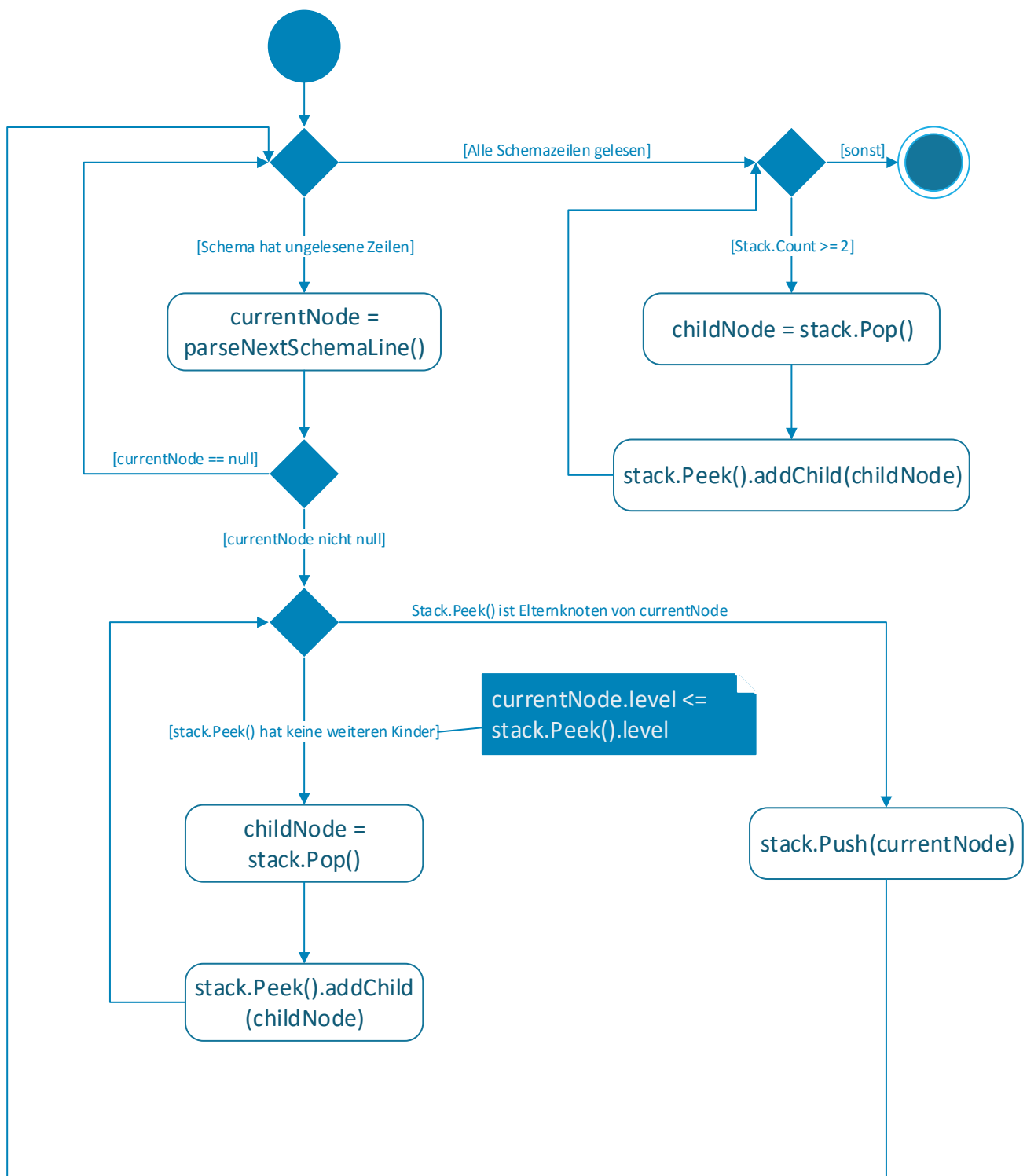
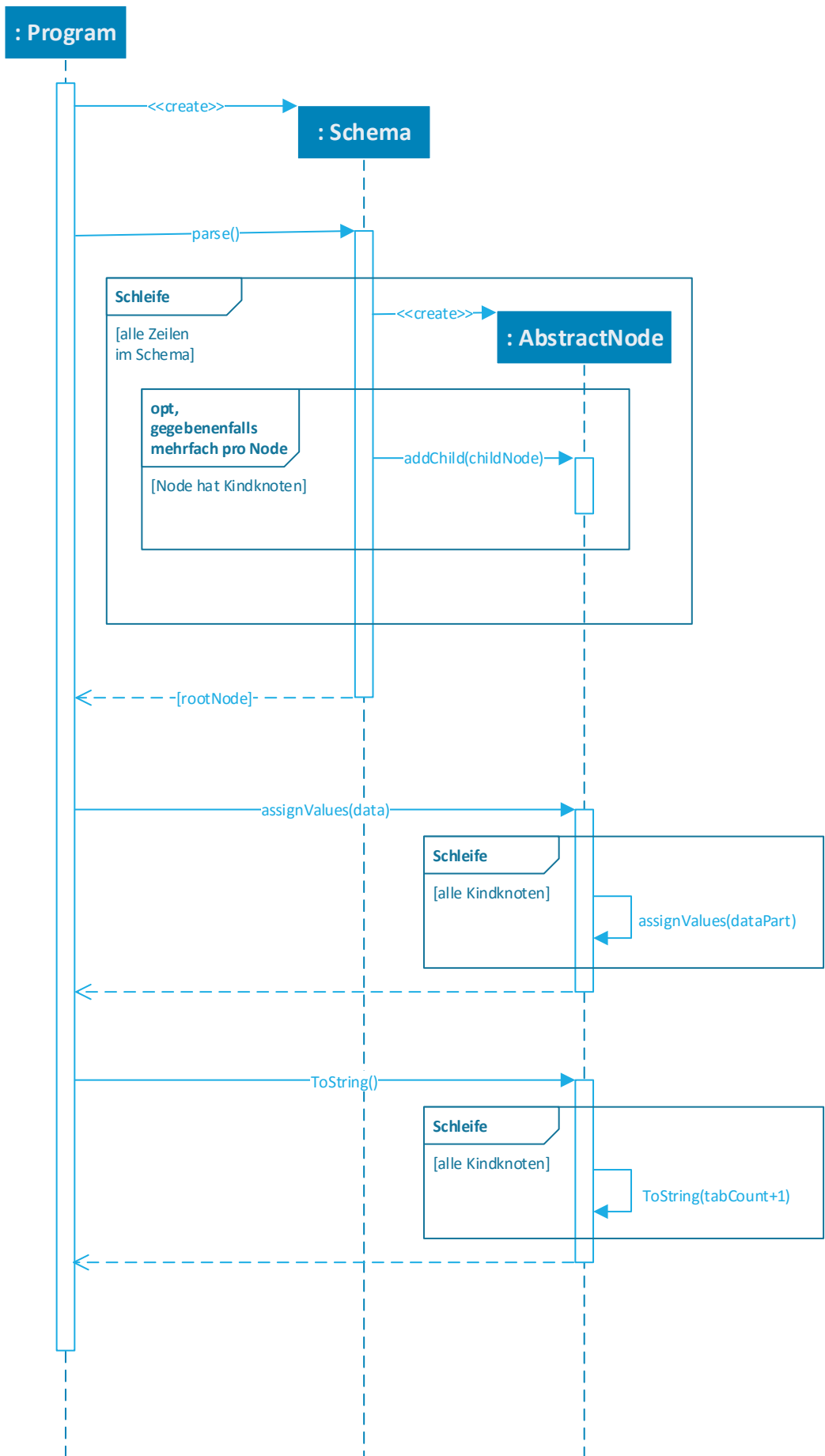


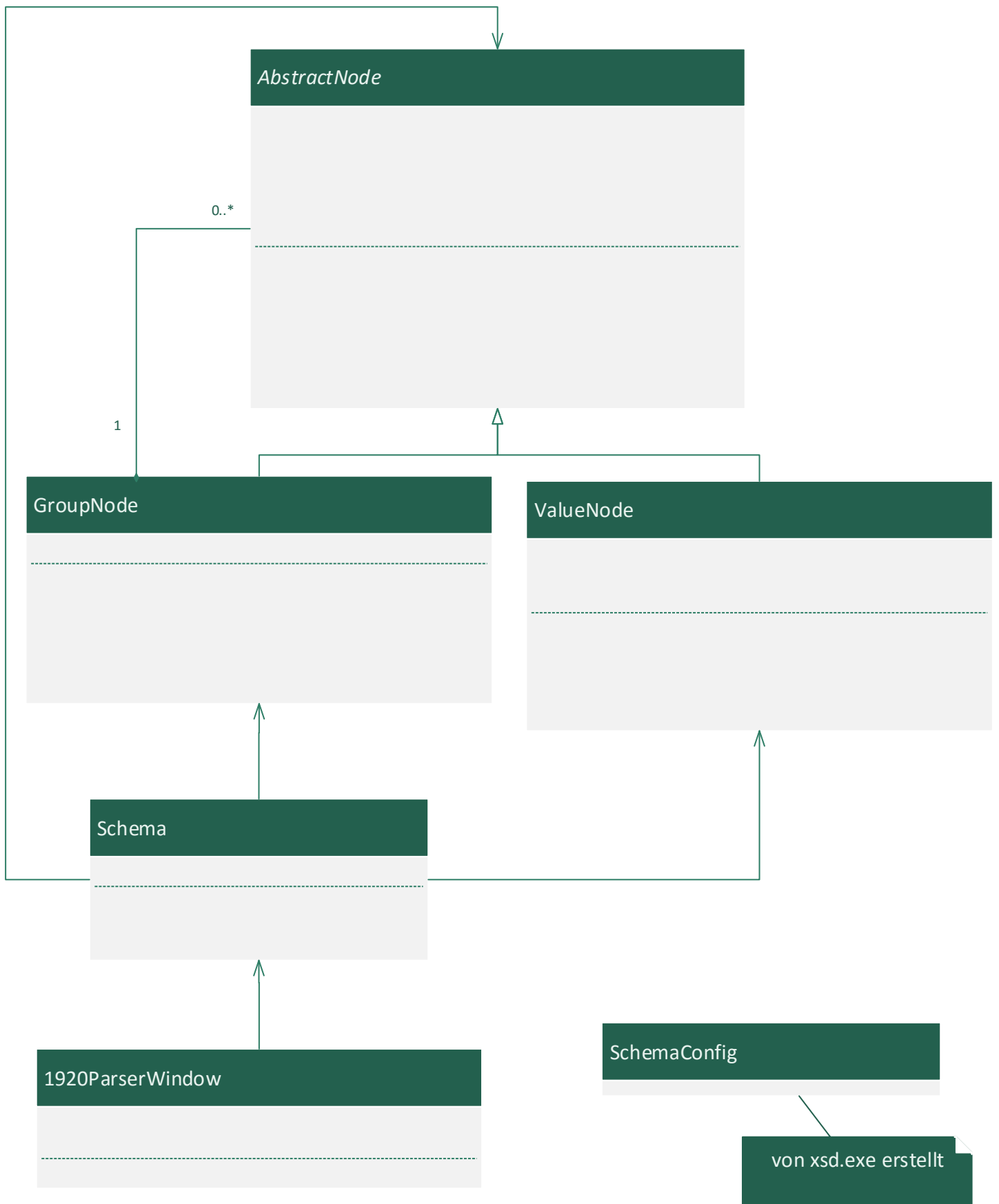
```
.fo off
.pa
+-----+-----+
I          I          I          I
I          I          I NACHFUELL I
I P H A R M O S I          Copy-Book-Beschreibung I          I
I          I Schnittstelle : Host --> PC I          I
I          I          I          IOVK91 I
+-----+-----+
I P H A R M O S I Froh zu sein bedarf es I Ausg.: 1 Wink I
I 8510 Fuerth 2 I          P H A R M O S I Vom :13.12.99I
I 0911-9300-695 I          I Wink :29:10.02I
+-----+-----+
Dateiname: PH326591 OPUS

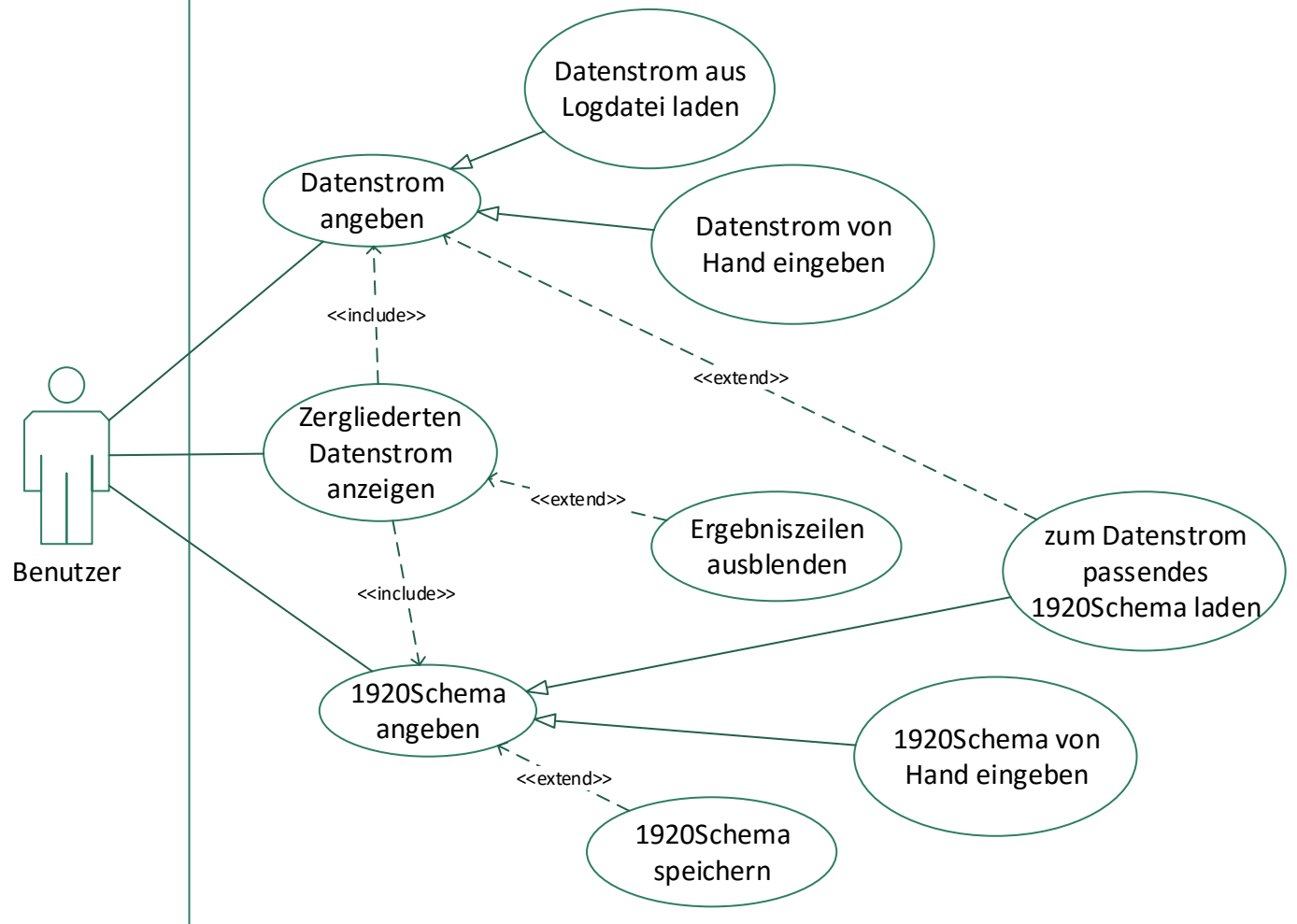
Satzlaenge:          Bytes
:h6.IOVK91 Schnittstelle Host->PC
.sp 2
.fo off
.tr * 40
.cm FELD-BESCHREIBUNG IOVK91
.pi /Bereich/IOVK91
.bx 2 9          27 31 38 42          &$11
    Stufe  Feldname          Typ Laenge Tab  Kommentar
.bx
****
03  IO91-AREA          Schnittstellenbereich
05  IO91-TRANID        C 4          Host --> PC
05  IO91-FUNKID        C 2          Transaktionscode (VK91)
05  IO91-EBENE         N 1          Funktionscode
****
05  IO91-DATEN-AREA          Struktur fuer Wannendaten
****
10  IO91-WANNE          4 Wannendaten
15  IO91-SA            C 1          SATZART 1 = VORAB, 2 = LA
15  IO91-FIL           N 2          Filiale
15  IO91-L1RRN         N 6          Re. Satznr. L1-Satz = LA-Nr
15  IO91-STPL          N 3          Platznummer auf Datamobil
15  IO91-KZ-LAST       C 1          KZ letzte Wanne
15  IO91-VZEIT         N 4          Vorlaufzeit in SEK
15  IO91-UZEIT         N 6          Uebergabezeit an LR
15  IO91-IPUNKT-WNR    N 7          Wannennr vom I-Punkt
15  IO91-WANREST       C 6          Filler
15  IO91-POSTAB        40 Tabelle der Positionen
20  IO91-KANAL         N 8          Kanalnummer
R20 IO91-KANAL-C       C 8          LAGERORT
20  IO91-MENGE         N 2          Menge
20  IO91-S-REST        C 1          Filler
10  IO91-REST          C 9          REST
****
.bx
R05  IO91-FEHLER-AREA          Struktur fuer Fehlermeldung
10  IO91-FEHLMELD        C 80         Meldungstext
R10  IO91-FEHLERMELDR          Struktur fuer Fehlermeldung
15  IO91-FEHLMELD2       C 20         Meldungstext
15  IO91-FEHLNR         N 5          L1-Nummer usw.
15  filler              C 1
15  IO91-FEHLNR2        N 5          L1-Nummer usw.
15  filler              C 49
10  IO91-REST2          C 1831        REST
****
.bx
R05  IO91-einzel          Struktur fuer Fehlermeldung
10  IO91-E-L1RRN         N 5          Taktnummer
10  IO91-E-stpl         N 2          Platznummer im takt
10  IO91-REST3          C 1906        REST
.bx
R05  IO91-STATION-AREA          STRUKTUR FUER FEHLERMELDUNG
10  IO91-STTAB          40 TABELLE DER STATIONEN
15  IO91-VSTATION       N 2          STATION VOR AUTOMAT
****
.bx
.bx off
.cm ENDE-FELDER
Copy-Book der Anwenderfelder in Commarea des GMO-Online-Rahmens
.sp
.tr * *
Typ:
C = Alpha-Feld  P = num. Feld gepackt      N = num. Feld ungepackt
X = Hex-Feld    H = Binaerfeld (Halbwort) V = Binaerfeld (Vollwort)
Laenge:
Die Laengenparameter bedeuten: Gesamtlaenge, davon Dezimalstellen
.fo on
.kp off
.cm Angelegt am 13.12.99 15:13:49 von 0SPHWINK
```

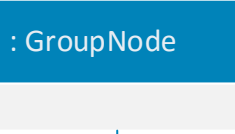


Schema in Baum
parsen









Schema

01	Daten
03	Personendaten
05	Vorname C 5
05	Nachname C 4
R03	Gesamter-Name C 9
03	Bestellungen 2
05	Artikelnr N 3

Datenstrom

VN~~~NN~~123456

children[0]

: GroupNode

children[0]

: GroupNode

children[0]

: ValueNode

children[1]

: ValueNode

children[1]

: ValueNode

children[2]

: GroupNode

children[0]

: ValueNode

children[3]

: GroupNode

children[0]

: ValueNode

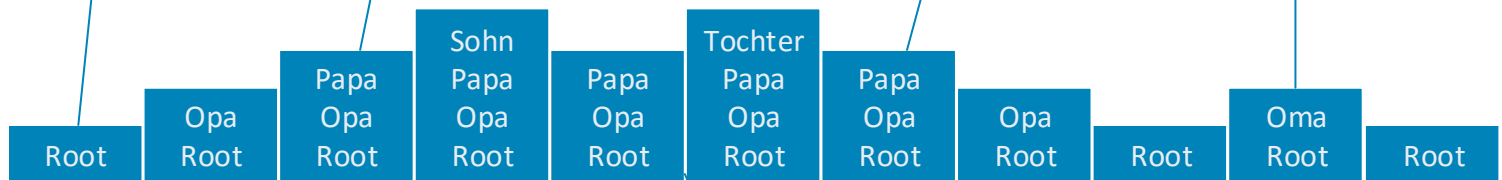
01 Opa
03 Papa
05 Sohn
05 Tochter
01 Oma

künstlicher
Wurzelknoten mit
Stufennummer 0

Weil ein Element nur auf
den Stack gepusht wird,
wenn es direkter
Nachfolger von StackTop
ist, enthält der Stack zu
allen Zeiten ausschließlich
die direkten Vorfahren des
obersten Elements.

Zeile Oma wurde aus dem Schema gelesen.
Oma ist kein Kind von Tochter. Das oberste
Stackelement wird vom Stack gepoppt und
dem jetzt obersten Stackelement als Kind
hinzugefügt. Das wird solange wiederholt,
bis $\text{Oma.Level} > \text{stack.Peek().Level}$

Wenn alle Schemazeilen
gelesen wurden, werden
die restlichen Elemente
solange gepoppt und
hinzugefügt, bis auf dem
Stack nur noch Root liegt.



Stufennummern vergleichen:
 $1 > 0$, Opa ist also ein Kindknoten von Root.
Der Baum unterhalb von Opa ist noch nicht
bearbeitet, daher kommt Opa auf den Stack.

Stufennummern von Tochter und Sohn
vergleichen, $5 \leq 5$, Tochter ist kein Kind von
Sohn. Sohn hat daher keine weiteren Kinder,
wird vom Stack gepoppt und zu Papa
hinzugefügt. Der Vorgang wird wiederholt,
bis der richtige Elternknoten für Tochter
gefunden wurde ($\text{Tochter.Level} > \text{stack.Peek().Level}$).