Leidraad VISI-systematiek versie 1.4

Bijlage 2
VISI-systematiek Deel 1; Raamwerken

Normatief

Documentversie: 1.1
Datum: 17 oktober 2014
Status: definitief

N.B.
De wijzigingen t.o.v. de vorige versie zijn ***geel*** gemarkeerd.

Inhoud

1 Elementtypen

1.1 AppendixType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3]
Referenties: complexElements [4.1]

```
ENTITY AppendixType;	--	Een AppendixType bevat de definitie van een bijlage. Welke gegevens bijgehouden worden bij een bijlage is te definiëren in het xml veld.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;

 complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
END_ENTITY;
```

Een AppendixType bevat de definitie van een bijlage. Welke gegevens bijgehouden worden bij een bijlage is te definiëren in de complexElements [4.3] veld.

Simpel voorbeeld:

```
<AppendixType id="StandardAppendixType">
  <description>Standaard appendix type</description>
  <startDate>2011-01-23T00:00:00Z</startDate>
  <endDate>2011-12-31T00:00:00Z</endDate>
  <state>active</state>
  <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
  <userLaMu>Peter Bonsma</userLaMu>
</AppendixType>
```

1.2 ComplexElementType

Attributen: id [2.1]
Elementen description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9]
Referenties: complexElements [4.1], simpleElements [4.11]

```
ENTITY ComplexElementType;	--	Een ComplexElementType is een verzameling van SimpleElementTypes, elk genoemd SimpleElementType komt precies het aantal keer voor dat hij genoemd wordt.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
```

```
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;

 complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
 simpleElements : OPTIONAL SET [0:?] OF SimpleElementType;
END_ENTITY;
```

Een ComplexElementType is een verzameling van SimpleElementTypes [1.12], elk genoemd SimpleElementType [1.12] komt precies het aantal keer voor dat hij genoemd wordt.

Simpel voorbeeld:

```
<ComplexElementType id="MenukaartItem">
 <description>Item op menukaart</description>
 <startDate>2011-01-23T00:00:00Z</startDate>
 <endDate>2011-12-31T00:00:00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
 <simpleElements>
  <SimpleElementTypeRef idref="Naam"/>
  <SimpleElementTypeRef idref="Prijs"/>
  <SimpleElementTypeRef idref="Omschrijving"/>
  <SimpleElementTypeRef idref="Calorieën"/>
 </simpleElements>
</ComplexElementType>
```

1.3 ElementCondition

Attributen: id [2.1]
Elementen: description [3.6], condition [3.4], helpInfo [3.9]
Referenties: complexElement [4.2], simpleElement [4.11], messageInTransaction [4.9]

```
ENTITY ElementCondition;   --          De conditie op een SimpleElementType gebruikt binnen een
specifiek MessageType.
 description : STRING;
 condition : STRING;
 helpInfo : OPTIONAL STRING;

 complexElement : OPTIONAL ComplexElementType;
 simpleElement : OPTIONAL SimpleElementType;
 messageInTransaction : OPTIONAL MessageInTransactionType;
END_ENTITY;
```

De conditie op een SimpleElementType [1.12] gebruikt binnen een specifiek MessageInTransactionType [1.5] of als onderdeel van een ComplexElementType [1.2] of altijd.

Simpel voorbeeld:

```
<ElementCondition id="Prijsrestrictie">
 <description>Minimale prijs van een menukaart item</description>
```

```
 <condition>FREE</condition>
 <simpleElement>
  <SimpleElementType>Prijs</SimpleElementType>
 </simpleElement>
 <messageInTransaction>
  <MessageInTransactionTypeRef idref="mitt006" />
 </messageInTransaction>
</ElementCondition>
```

1.4 GroupType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9]

```
ENTITY GroupType;          -- De definitie van de groep voor het opslaan van bijlagen verzonden
met een bericht binnen een transactie.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
END_ENTITY;
```

De definitie van de groep voor het opslaan van bijlagen verzonden met een bericht binnen een transactie. Op het moment wordt in de praktijk geen functionaliteit door leveranciers toegekend aan dit element. Een GroupType maakt echter wel onderdeel uit van de structuur van een raamwerk.

Simpel voorbeeld:

```
<GroupType id="StandardGroupType">
 <description>Standaard groep</description>
 <startDate>2011-12-20T00:00:00Z</startDate>
 <endDate>2012-12-31T00:00:00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-12-20T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
</GroupType>
```

1.5 MessageInTransactionType

Attributen: id [2.1]
Elementen: requiredNotify [3.16], dateLaMu [3.4], userLaMu [3.25], received [3.15], send [3.22], state [3.24], initiatorToExecutor [3.10], openSecondaryTransactionsAllowed [3.13], firstMessage [3.8]
Referenties: message [4.8], previous [4.9], transaction [4.16], transactionPhase [4.17], group [4.6], appendixTypes [4.1], conditions [4.4]

```
ENTITY MessageInTransactionType; -- De instantiatie van een MessageType binnen een
TransactionType behorende bij een bepaald groep type (GroupType).
 requiredNotify : OPTIONAL INTEGER;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 received : OPTIONAL BOOLEAN;
 send : OPTIONAL BOOLEAN;
 state : OPTIONAL STRING;
 initiatorToExecutor : BOOLEAN;
 openSecondaryTransactionsAllowed :
  OPTIONAL BOOLEAN;
 firstMessage : OPTIONAL BOOLEAN;

 message : MessageType;
 previous : OPTIONAL SET [0:?] OF MessageInTransactionType;
 transaction : TransactionType;
 transactionPhase : OPTIONAL TransactionPhaseType;
 group : OPTIONAL GroupType;
 appendixTypes : OPTIONAL SET [1:?] OF AppendixType;
 conditions : OPTIONAL SET[1:?] OF MessageInTransactionTypeCondition;
END_ENTITY;
```

Het gebruik van een MessageType [1.6] binnen een TransactionType [1.14] behorende bij een bepaald groeptype (GroupType [1.4]).

Simpel voorbeeld:

```
<MessageInTransactionType id="MiTT_002">
 <requiredNotify>0</requiredNotify>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
 <received>true</received>
 <send>true</send>
 <state>active</state>
 <initiatorToExecutor>false</initiatorToExecutor>
 <openSecondaryTransactionsAllowed>
  true</openSecondaryTransactionsAllowed>
 <firstMessage>false</firstMessage>
 <message>
  <MessageTypeRef idref="VerstrekkenVanMenukaartBericht"/>
 </message>
 <previous>
  <MessageInTransactionTypeRef idref="MiTT_001"/>
 </previous>
 <transaction>
  <TransactionTypeRef idref="MenukaartVerkrijgenTransactie"/>
 </transaction>
 <transactionPhase>
  <TransactionPhaseTypeRef idref="MenukaartGegeven">
 </transactionPhase>
 <group>
  <GroupTypeRef idref="StandardGroupType"/>
 </group>
 <appendixTypes>
  <AppendixTypeRef idref=”Menukaart” />
 </appendixTypes>
</MessageInTransactionType>
```

1.6 MessageInTransactionTypeCondition

Attributen: id [2.1]
Elementen: state [3.24], dateLaMu [3.5], userLaMu [3.25], helpInfo [3.9]
Referenties: sendAfter [4.11], sendBefore [4.12]

```
ENTITY MessageInTransactionTypeCondition;
 sendAfter : OPTIONAL SET [1:?] OF MessageInTransactionType;
 sendBefore : OPTIONAL SET [1:?] OF MessageInTransactionType;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
END_ENTITY;
```

Toegestane verwijzingen bij "sendAfter" en "sendBefore"zijn "MessageInTransactionType's" die ontvangen kunnen worden in de actuele transactie of van de aangesloten transacties, waarbij de persoon die het actuele bericht behandelt initiator of executor is. Met aangesloten transacties worden de transactie waaruit een transactie geïnitieerd is en de directe subtransacties bedoeld. Met directe subtransacties worden transacties bedoeld die vanuit de actuele transactie geïnitieerd zijn, dus niet subtransacties van subtransacties.

1.7 MessageType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3]
Referenties: complexElements [4.1], appendixTypes [4.1]

```
ENTITY MessageType; -- De definitie van een type bericht (MessageType), hierin is ook
gedefinieerd hoe dit bericht gestructureerd is en welke verzameling van SimpleElementType's (via
ComplexElementType's) hierbij horen.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;

 complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
 appendixTypes : OPTIONAL SET [1:?] OF AppendixType;
END_ENTITY;
```

De definitie van een type bericht (MessageType [1.6]), hierin is ook gedefinieerd hoe dit bericht gestructureerd is en welke verzameling van SimpleElementType's [1.12] (via ComplexElementType's [1.2]) hierbij horen.

Simpel voorbeeld:

```
<MessageType id="VerstrekkenVanMenukaartBericht">
  <description>Bericht welke de menukaart bevat.</description>
```

```
 <startDate>2011-01-23T00:00:00Z</startDate>
 <endDate>2011-12-31T00:00:00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
 <complexElements>
  <ComplexElementTypeRef idref="Menukaart"/>
 </complexElements>
 <appendixTypes>
  <AppendixTypeRef idref=”Menukaart” />
 </appendixTypes>
</MessageType>
```

1.8 OrganisationType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3]
Referenties: complexElements [4.1]

```
ENTITY OrganisationType;    -- Definitie van een bepaalde groep organisaties, in het algemeen 1 instance aanwezig in een raamwerk met als reden het definiëren van de structuur van elementen die door voor elke instance van dit tot object gepromote OrganisationType ingevuld moet worden.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;

 complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
END_ENTITY;
```

Definitie van een bepaalde groep organisaties, in het algemeen eenmalig aanwezig in een raamwerk met als reden het definiëren van de structuur van elementen die voor elke instantie van dit tot object gepromote OrganisationType ingevuld moet worden.

Simpel voorbeeld:

```
<OrganisationType id="StandardOrganisationType">
 <description>Standaard organisation type</description>
 <startDate>2011-01-23T00:00:00Z</startDate>
 <endDate>2011-12-31T00:00:00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
</OrganisationType>
```

1.9 PersonType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3]
Referenties: complexElements [4.1]

```
ENTITY PersonType;          -- Definitie van een bepaalde groep personen, in het algemeen 1
instance aanwezig in een raamwerk met als reden het definiëren van de structuur van elementen
die door voor elke instance van dit tot object gepromote PersonType ingevuld moet worden.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;

 complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
END_ENTITY;
```

Definitie van een bepaalde groep personen, in het algemeen eenmalig aanwezig in een raamwerk met als reden het definiëren van de structuur van elementen die voor elke instantie van dit tot object gepromote PersonType ingevuld moet worden.

Simpel voorbeeld:

```
<PersonType id="StandardPersonType">
  <description>Standaard persoons type</description>
  <startDate>2011-01-23T00:00:00Z</startDate>
  <endDate>2011-12-31T00:00:00Z</endDate>
  <state>active</state>
  <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
  <userLaMu>Peter Bonsma</userLaMu>
</PersonType>
```

1.10 ProjectType

Attributen: id [2.1]
Elementen: namespace [3.13], description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3]
Referenties: complexElements [4.1]

```
ENTITY ProjectType; -- Definitie van een bepaalde groep projecten, in het algemeen 1 instance
aanwezig in een raamwerk met als reden het definiëren van de structuur van elementen die door
voor elke instance van dit tot object gepromote ProjectType ingevuld moet worden.
 namespace : STRING;
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
```

```
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;
 complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
END_ENTITY;
```

Definitie van een bepaalde groep projecten, in het algemeen eenmalig aanwezig in een raamwerk met als reden het definiëren van de structuur van elementen die voor elke instantie van dit tot object gepromote ProjectType ingevuld moet worden.

Simpel voorbeeld:

```
<ProjectType id="StandardProjectType">
 <namespace>http://www.visi.nl/testproject</namespace>
 <description>Standaard project type</description>
 <startDate>2011-01-23T00:00:00Z</startDate>
 <endDate>2011-12-31T00:00:00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
</ProjectType>
```

1.11 RoleType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3], responsibilityScope [3.18] , responsibilityTask [3.20], responsibilitySupportTask [3.19], responsibilityFeedback [3.17]

```
ENTITY RoleType; -- De definitie van een bepaald rol type, belangrijk voor TransactionType.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;
 responsibilityScope : OPTIONAL STRING;
 responsibilityTask : OPTIONAL STRING;
 responsibilitySupportTask : OPTIONAL STRING;
 responsibilityFeedback : OPTIONAL STRING;
END_ENTITY;
```

De definitie van een bepaald roltype, belangrijk voor TransactionType [1.14].

Simpel voorbeeld:

```
<RoleType id="ober">
 <description>Verantwoordelijk voor het opnemen en uitzetten van bestellingen</description>
 <startDate>2011-05-04T00:00:00:00Z</startDate>
```

```
 <endDate>2011-05-04T00:00:00.00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-05-04T00:00:00.00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
 <responsibilityScope/>
 <responsibilityTask/>
 <responsibilitySupportTask/>
 <responsibilityFeedback/>
</RoleType>
```

1.12 SimpleElementType

Attributen: id [2.1]
Elementen: description [3.6], interfaceType [3.11], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], valueList [3.26]
Referenties: userDefinedType [4.18]

```
ENTITY SimpleElementType; -- Een specificatie van een simpel element type
(SimpleElementType). Dit ElementType beschrijft een eigenschap die binnen verschillende
objectstructuren zoals bijv. in MessageType kan voorkomen (zie ook AppendixType, ProjectType,
PersonType en OrganisationType), de relatie is dan altijd via ComplexElementType.
 description : STRING;
 interfaceType : OPTIONAL STRING;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 valueList : OPTIONAL STRING;
 userDefinedType : UserDefinedType;
END_ENTITY;
```

Een specificatie van een simpel elementtype (SimpleElementType [1.12]). Dit elementtype beschrijft een eigenschap die binnen verschillende objectstructuren zoals bijv. in MessageType [1.6] kan voorkomen (zie ook AppendixType [1.1], ProjectType [1.10], PersonType [1.9] en OrganisationType [1.8]), de relatie is dan altijd via ComplexElementType [1.2].

Simpel voorbeeld:

```
<SimpleElementType id="Naam">
 <description>Naam van het menu item</description>
 <interfaceType/>
 <state>active</state>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLaMu>Peter Bonsma</userLaMu>
 <userDefinedType>
  <UserDefinedTypeRef idref="String"/>
 </userDefinedType>
</SimpleElementType>
```

1.13 TransactionPhaseType

Attributen: id [2.1]

Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3]

```
ENTITY TransactionPhaseType; -- Het definiëren van de transactie fase types waarin een
transactie zich kan bevinden. Voorbeelden zijn 'opdracht geaccepteerd' en 'deelresultaat
ontvangen'.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;
END_ENTITY;
```

Het definiëren van de transactiefasetypes waarin een transactie zich kan bevinden. Over het algemeen worden in VISI raamwerken de volgende transactiefasetypes gebruikt: ‘Start’, ‘Verzocht’, ‘Beloofd/Executie’, ‘Wijziging/Hold’, ‘Melding Gereed’ en ‘Einde’. Leveranciers van VISI-compatible software hebben vooralsnog weinig tot geen functionaliteit aan de transactiefasetypes verbonden. TransactionPhase is optioneel.

Simpel voorbeeld:

```
<TransactionPhaseType id="WachtenOpMenukaart">
  <description>Menukaart gevraagd maar nog niet gegeven</description>
  <startDate>2011-01-23T00:00:00Z</startDate>
  <endDate>2011-12-31T00:00:00Z</endDate>
  <state>active</state>
  <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
  <userLaMu>Peter Bonsma</userLaMu>
</TransactionPhaseType>
```

1.14 TransactionType

Attributen: id [2.1]
Elementen: description [3.6], startDate [3.23], endDate [3.7], state [3.24], dateLaMu [3.4], userLaMu [3.25], language [3.12], category [3.2], helpInfo [3.9], code [3.3], result [3.21], baseType [3.1]
Referenties: subTransactions [4.15], initiator [4.7], executor [4.4], appendixTypes [4.1]

```
ENTITY TransactionType; -- De definitie van een type transactie, een transactietype kan tevens zelf
weer naar transactietypes verwijzen. Een transactie wordt geïnitieerd door een persoon
behorend bij een organisatie in een bepaalde rol, op dit niveau geven we aan welk roltype deze
initiator moet bezitten (het gepromote schema zal dit vervolgens afdwingen) idem voor executor.
 description : STRING;
 startDate : OPTIONAL DATETIME;
 endDate : OPTIONAL DATETIME;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 language : OPTIONAL STRING;
 category : OPTIONAL STRING;
```

```
 helpInfo : OPTIONAL STRING;
 code : OPTIONAL STRING;
 result : OPTIONAL STRING;
 basePoint : OPTIONAL STRING;
 subTransactions : OPTIONAL SET [1:?] OF TransactionType;
 initiator : RoleType;
 executor : RoleType;
 appendixTypes : OPTIONAL SET [1:?] OF AppendixType;
END_ENTITY;
```

De definitie van een transactietype (TransactionType). Een transactietype kan tevens zelf weer naar transactietypes verwijzen. Een transactie wordt geïnitieerd door een persoon behorend bij een organisatie in een bepaalde rol. Op dit niveau geven we aan welk rol type deze initiator [4.7] moet bezitten (het gepromote schema zal dit vervolgens afdwingen) idem voor executor [4.4].

Simpel voorbeeld:

```
<TransactionType id="MenukaartVerkrijgenTransactie">
 <description>
  De transactie om te komen tot het verkrijgen van de juiste
  Menukaart
 </description>
 <startDate>2011-01-23T00:00:00Z</startDate>
 <endDate>2011-12-31T00:00:00Z</endDate>
 <state>active</state>
 <dateLaMu>2011-01-23T00:00:00Z</dateLaMu>
 <userLamu>Peter Bonsma</userLaMu>
 <initiator>
  <RoleTypeRef idref="Consument"/>
 </initiator>
 <executor>
  <RoleTypeRef idref="Werknemer"/>
 </executor>
</TransactionType>
```

1.15 UserDefinedType

Attributen: id [2.1]
Elementen: description [3.6], state [3.24], dateLaMu [3.4], userLaMu [3.25], baseType [3.1], xsdRestriction [3.27], language [3.12], helpInfo [3.9]

```
ENTITY UserDefinedType; -- Een specificatie van een data type (SimpleElementType). Dit datatype
geeft de vorm aan invulvelden in het uiteindelijk bericht, zoals bijvoorbeeld een postcode begint
altijd met 4 cijfers en dan volgen verplicht 2 letters.
 description : STRING;
 state : OPTIONAL STRING;
 dateLaMu : OPTIONAL DATETIME;
 userLaMu : OPTIONAL STRING;
 baseType : STRING;
 xsdRestriction : OPTIONAL STRING;
 language : OPTIONAL STRING;
 helpInfo : OPTIONAL STRING;
END_ENTITY;
```

Een specificatie van een datatype (SimpleElementType [1.12]). Dit datatype geeft de vorm aan invulvelden in het uiteindelijk bericht, zoals bijvoorbeeld een postcode begint altijd met 4 cijfers en dan volgen verplicht 2 letters.

Simpel voorbeeld:

```
<UserDefinedType id="String">
  <description>Standaard string</description>
  <state>active</state>
  <dateLaMu>2010-12-20T00:00:00Z</dateLaMu>
  <userLaMu>Peter Bonsma</userLaMu>
  <baseType>STRING</baseType>
  <xsdRestriction/>
</UserDefinedType>
```

2 Attributen

2.1 id

Unieke 'korte' naam[1] van de objectinstantie, deze naam zal na de promotorstap een objectnaam zijn, voorbeeld:

Raamwerk

```
<OrganisationType id="Organisatie">
  <description>De attributen van een organisatie</description>
  <startDate>2010-12-12T00:00:00Z</startDate>
  <endDate>2099-12-31T00:00:00Z</endDate>
  <state>actief</state>
  <dateLaMu>2010-12-12T00:00:00Z</dateLaMu>
  <userLaMu>visitestbeheer</userLaMu>
  <language>NL</language>
  <category>S</category>
  <helpInfo>http://x/</helpInfo>
  <code>DEFAULT</code>
</OrganisationType>
```

Bericht

```
<Organisatie id="TNO">
  <name>TNO Building & Construction</name>
  <state>active</state>
  <dateLaMu>2010-12-02T00:00:00Z</dateLaMu>
  <userLaMu>Peter Bonsma</userLaMu>
</Organisatie>
```

[1] Attribuut ‘id’ is in XML een specifieke definitie en is daarom aan meer restricties verbonden dan ‘gewone’ attribuutwaarden. Zo kan een id-waarde geen spaties bevatten en veelal ook geen bijzondere karakters. Ook kan een id-waarde niet met een cijfer beginnen. Zie verder xml:id Version 1.0 [http://www.w3.org/TR/xml-id/].

3 Elementen

3.1 baseType

```
baseType : STRING;
```

Geeft het basistype van een SimpleElementType [1.12] aan.

Voorbeeld:

```
<SimpleElementType id="Hoogte">
 ...
 <userDefinedType>
  <UserDefinedType id="...">
   ...
   <baseType>INTEGER</baseType>
   ...
  </UserDefinedType>
 </userDefinedType>
</SimpleElementType>
```

hierbij is dus het SimpleElementType [1.12] *Hoogte* altijd een integer (eventueel met als restrictie xsdRestriction [3.27]).

3.2 category

```
category : OPTIONAL STRING;
```

Categorie waar deze objectinstantie toe behoort, dit is een optionele waarde.

3.3 code

```
code : OPTIONAL STRING;
```

Binnen een raamwerk af te spreken code voor deze objectinstantie.

Voorbeeld:

```
<... id="...">
 <code>EAN 33156</code>
</...>
```

3.4 condition

```
condition : STRING;
```

Conditie element voor een ElementCondition [1.3]. Voor gedefinieerde waarden:

- EMPTY in alle gevallen zal het SimpleElement leeg worden gemaakt

- FIXED in alle gevallen zal het SimpleElement gefixeerd zijn
- FREE in alle gevallen zal het SimpleElement vrij invulbaar zijn

3.5 dateLaMu

```
dateLaMu : OPTIONAL DATETIME;
```

Datum en tijd van laatste mutatie aan deze objectinstantie.

```
<... id="...">
 ...
 <dateLaMu>2010-12-02T00:00:00Z</dateLaMu>
 ...
</...>
```

3.6 description

```
description : STRING;
```

Omschrijving van het geïnstantieerde object.

Voorbeeld:

```
<... id="Deurblad">
 ...
  <description>Het blad van een vlakke deur.</description>
  ...
</...>
```

3.7 endDate

```
endDate : OPTIONAL DATETIME;
```

Eind datum en tijd van geldigheid van deze objectinstantie. In de praktijk wordt hier door leveranciers van VISI-compatible software nog geen functionaliteit aan toegekend.

Voorbeeld:

```
<... id="...">
 ...
  <endDate>2011-02-03T00:00:00Z</endDate>
  ...
</...>
```

3.8 firstMessage (zie TC022)

```
firstMessage : OPTIONAL BOOLEAN;
```

Aanduiding of een MessageInTransaction [1.5] de mogelijkheid biedt om als startbericht van een nieuwe transactie te fungeren.

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <firstMessage>true</firstMessage>
 ...
</MessageInTransactionType>
```

3.9 helpInfo

```
helpInfo : OPTIONAL STRING;
```

Een URL/URI naar meer informatie over deze objectinstantie.

Voorbeeld:

```
<... id="...">
 ...
 <helpInfo>http://www.visi.nl/helpInfo_object0001.html</helpInfo>
  ...
</...>
```

3.10 initiatorToExecutor

```
initiatorToExecutor : OPTIONAL BOOLEAN;
```

Een logische waarde die aangeeft in welke richting een bericht geacht wordt verstuurd te worden.

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <initiatorToExecutor>false</initiatorToExecutor>
 ...
 <message>
  <MessageType id="OfferteAcceptatie">
   ...
  </MessageType>
 </message>
 <transaction>
  <TransactionType id="OfferteTraject">
  ...
   <initiator>
    <RoleType id="Uitvoerende">
     ...
    </RoleType>
   </initiator>
   <executor>
    <RoleType id="Opdrachtgever">
     ...
    </RoleType>
   </executor>
    ...
```

```
  </TransactionType>
 </transaction>
  ...
 <MessageInTransactionType id="...">
```

Hier verwachten wordt het bericht *OfferteAcceptatie* van Opdrachtgever (executor [4.4]) naar Uitvoerende (initiator [4.7]) gestuurd.

3.11 interfaceType

```
interfaceType : OPTIONAL STRING;
```

Type interface c.q. view op dit SimpleElementType [1.12] voor dit specifieke bericht. Bijvoorbeeld als het gegevenselement bedoeld is als invoer (inputText) of slechts een vaste inhoud bevat en niet aangepast mag worden (label). Op het moment wordt hier in de praktijk geen functionaliteit aan verbonden.

3.12 language

```
language : OPTIONAL STRING;
```

Taal die gebruikt wordt voor deze tot object te promoten instantie, bijvoorbeeld:

```
<... id="...">
 ...
  <language>NL</language>
  ...
</...>
```

3.13 namespace

```
namespace : STRING;
```

Namespace target naam ter identificatie van berichten die bij dit raamwerk horen, bijvoorbeeld:

```
<ProjectType id="...">
  <namespace>http://www.visi.nl/testraamwerk</namespace>
  ...
</ProjectType>
```

3.14 openSecondaryTransactionsAllowed

```
openSecondaryTransactionsAllowed : OPTIONAL BOOLEAN;
```

Optionele logische waarde die de mogelijkheid aangeeft of secundaire transacties nog niet afgerond hoeven te zijn voordat met de primaire transactie kan worden verder gegaan.

De interpretatie voor ‘TRUE’ is dat niet alle instanties van secundaire transacties hoeven te zijn afgerond voordat met de primaire transactie kan worden verder gegaan. Als de waarde ‘FALS’" is dienen alle instanties van secundaire transacties te worden afgerond voordat de primaire transactie hervat kan worden. Indien openSecondaryTransactionsAllowed niet is gedefinieerd wordt dit geïnterpreteerd als ‘TRUE’.

3.15 received

```
received : OPTIONAL BOOLEAN;
```

Logische waarde die aangeeft of het vorige bericht ontvangen zou moeten zijn. In de praktijk wordt dit element niet gebruikt.

3.16 requiredNotify

```
requiredNotify : OPTIONAL INTEGER;
```

Op het moment wordt aan het element requiredNotify geen betekenis toegekend.

3.17 responsibilityFeedback

```
responsibilityFeedback : OPTIONAL STRING;
```

Terugkoppeling die vanuit de verantwoordelijkheid van de rol wordt verwacht richting andere rollen.

3.18 responsibilityScope

```
responsibilityScope : OPTIONAL STRING;
```

Scope/kader waarbinnen de verantwoordelijkheden behorende bij de betreffende rol zijn gedefinieerd.

3.19 responsibilitySupportTask

```
responsibilitySupportTask : OPTIONAL STRING;
```

Taken die worden uitgevoerd om andere rollen te ondersteunen. Denk hierbij bijvoorbeeld aan gedelegeerde verantwoordelijkheden.

3.20 responsibilityTask

```
responsibilityTask : OPTIONAL STRING;
```

Taken die voortkomen uit de verantwoordelijkheden van de betreffende rol.

3.21 result

```
result : OPTIONAL STRING;
```

Resultaat.

3.22 send

```
send : OPTIONAL BOOLEAN;
```

Logische waarde die aangeeft of het huidige bericht inmiddels verstuurd zou moeten zijn. In de praktijk wordt dit element niet gebruikt.

3.23 startDate

```
startDate : OPTIONAL DATETIME;
```

Startdatum en tijd van geldigheid van deze objectinstantie. In de praktijk wordt hier door leveranciers van VISI-compatible software nog geen functionaliteit aan toegekend.

Voorbeeld:

```
<... id="...">
 ...
  <startDate>2010-02-03T00:00:00Z</startDate>
 ...
</...>
```

3.24 state

```
state : OPTIONAL STRING;
```

Status van deze objectinstantie, op dit moment mogelijke stadia:

```
<... id="...">
 ...
  <state>active</state>
   ...
</...>
```

en

```
<... id="...">
 ...
  <state>passive</state>
   ...
</...>
```

In de praktijk wordt door enkele leveranciers van VISI-compatible software de functionaliteit aan dit element verbonden om een element type zichtbaar of niet zichtbaar te maken in de software.

3.25 userLaMu

```
userLaMu : OPTIONAL STRING;
```

Gebruiker die de laatste mutatie aan deze objectinstantie heeft uitgevoerd (gewoon een string met de naam).

```
<... id="...">
 ...
  <userLaMu>Peter Bonsma</userLaMu>
  ...
</...>
```

3.26 valueList

```
valueList : OPTIONAL STRING;
```

Door puntkomma's gescheiden lijst van waarden die een instantie op berichtniveau uiteindelijk aan mag nemen. Oorspronkelijk was dit element bedoeld als enumeratie. In de huidige praktijk wordt dit opgelost met het element type UserDefinedType [1.15] en het element xsdRestriction [3.27]. In de xsdRestriction worden de enumeration values aangegeven. Aan het element valueList wordt in de huidige praktijk geen betekenis toegekend.

Voorbeeld:

```
<SimpleElementType id="...">
  ...
  <valueList>Groen;Rood;Oker Geel</valueList>
  ...
</SimpleElementType>
```

3.27 xsdRestriction

```
xsdRestriction : OPTIONAL STRING;
```

Dit is de restrictie die op berichtniveau door het berichtenschema zal worden uitgevoerd op de SimpleElementType's [1.12] van een UserDefinedType [1.15].

4 Referenties

4.1 appendixTypes

```
appendixTypes : OPTIONAL SET [1:?] OF AppendixType;
```

Een verwijzing naar een verzameling AppendixTypes [1.1] die in aanmerking komen voor een TransactionType [1.14] of een MessageType [1.6] of een MessageInTransactionType [1.5].

4.2 complexElement

```
complexElement : OPTIONAL ComplexElementType;
```

ComplexElementType [1.2] dat het SimpleElementType [1.12] bevat waarop de ElementCondition [1.3] op van toepassing is.

4.3 complexElements

```
complexElements : OPTIONAL SET [0:?] OF ComplexElementType;
```

Een verwijzing naar een verzameling SimpleElementType's [1.12] (verzameld in ComplexElementType [1.2]).

Voorbeeld:

```
<... id="Abc">
 ...
  <complexElements>
   <ComplexElementType id="ElementenSet1">
    ...
     <simpleElements>
      <SimpleElementType id="Element_A">
       ...
      </SimpleElementType>
      <SimpleElementType id="Element_B">
       ...
      </SimpleElementType>
     </simpleElements>
   </ComplexElementType>
   <ComplexElementType id="ElementenSet2">
    ...
   </ComplexElementType>
   <ComplexElementType id="ElementenSet3">
    ...
   </ComplexElementType>
  </complexElements>
</...>
```

Op berichtniveau ziet dat er dan als volgt uit:

```
<Abc id="...">
```

```
...
 <elementenSet1>
  <ElementenSet1>
   <Element_A>...</Element_A>
   <Element_B>...</Element_B>
  </ElementenSet1>
  ...
  <ElementenSet1>
   <Element_A>...</Element_A>
   <Element_B>...</Element_B>
  </ElementenSet1>
 </elementenSet1>
 <elementenSet2>
  <ElementenSet2>
   ...
  </ElementenSet2>
   ...
  <ElementenSet2>
   ...
  </ElementenSet2>
 </elementenSet2>
 <elementenSet3>
  <ElementenSet3>
   ...
  </ElementenSet3>
   ...
  <ElementenSet3>
   ...
  </ElementenSet3>
 </elementenSet3>
</...>
```

4.4 conditions

conditions : OPTIONAL SET[1:?] OF MessageInTransactionTypeCondition;

4.5 executor

executor : RoleType;

De 'uitvoerende' rol (RoleType [1.11]) die behoort tot een bepaalde transactie.

Voorbeeld:

```
<TransactionType id="OfferteTraject">
 ...
 <executor>
  <RoleType id="Opdrachtgever">
   ...
  </RoleType>
 </executor>
 ...
</TransactionType>
```

4.6 group

```
group : GroupType;
```

De GroupType [1.4] waar een bericht binnen een specifieke transactie toe behoort.

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <message>
  <MessageType id="M">
   ...
  </MessageType>
 </message>
 <group>
  <GroupType id="G">
   ...
  </GroupType>
 </group>
 <transaction>
  <TransactionType id="T">
   ...
  </TransactionType>
 </transaction>
 ...
</MessageInTransactionType>
```

We zien hier dat bericht *M* binnen transactie *T* behoort tot group [4.6] *G* (er kunnen binnen deze transactie *T* meer berichten *M* zijn die tot dezelfde of een andere group [4.6] behoren).

4.7 initiator

```
initiator : RoleType;
```

De 'initiërende' rol (RoleType [1.11]) die behoort tot een bepaalde transactie.

Voorbeeld:

```
<TransactionType id="OfferteTraject">
 ...
 <initiator>
  <RoleType id="Uitvoerende">
   ...
  </RoleType>
 </initiator>
 ...
</TransactionType>
```

4.8 message

```
message : MessageType;
```

Het berichttype dat gekoppeld is aan een MessageInTransactionType [1.5] instantie.

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <message>
  <MessageType id="...">
   ...
  </MessageType>
 </message>
 ...
</MessageInTransactionType>
```

4.9 messageInTransaction

```
messageInTransaction : OPTIONAL MessageInTransactionType;
```

MessageInTransactionType [1.5] waartoe de reikwijdte van de ElementCondition [1.3] is beperkt.

4.10 previous

```
previous : OPTIONAL SET [0:?] OF MessageInTransactionType;
```

Een MessageInTransactionType [1.5] kan afdwingen dat een eerder bericht binnen een specifieke transactie uitgevoerd moet zijn (dit voorgaande MessageInTransactionType [1.5] hoeft niet per definitie tot hetzelfde TransactionType [1.14] te behoren).

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <previous>
  <MessageInTransactionType id="...">
   ...
   <message>
    <MessageType id="Offerte">
     ...
    </MessageType>
   </message>
   ...
  </MessageInTransactionType>
 </previous>
 ...
 <message>
  <MessageType id="OfferteAcceptatie">
   ...
  </MessageType>
 </message>
 ...
</MessageInTransactionType>
```

4.11 sendAfter

```
sendAfter : OPTIONAL SET [1:?] OF MessageInTransactionType;
```

4.12 sendBefore

```
sendBefore : OPTIONAL SET [1:?] OF MessageInTransactionType;
```

4.13 simpleElement

```
simpleElement : OPTIONAL SimpleElementType;
```

De conditie op een SimpleElementType [1.12] welke binnen een MessageType [1.6] gebruikt wordt.

Voorbeeld:

```
<ElementCondition id="...">
 ...
 <condition>FREE</condition>
 ...
 <simpleElement>
  <SimpleElementTypeRef idref="Deurhoogte">
 </simpleElement>
 ...
</ElementCondition>
```

4.14 simpleElements

```
simpleElements : OPTIONAL SET [0:?] OF SimpleElementType;
```

Set van SimpleElementType's [1.12] welke binnen een ComplexElementType [1.2] aanwezig zijn.

Voorbeeld:

```
<ComplexElementType id="Deur">
 ...
 <simpleElements>
  <SimpleElementType id="Deurblad">
   ...
  </SimpleElementType>
  <SimpleElementType id="HangEnSluitwerk">
   ...
  </SimpleElementType>
  <SimpleElementType id="Bovenraam">
   ...
  </SimpleElementType>
 </simpleElements>
</ComplexElementType>
```

Op berichtniveau ziet dat er dan als volgt uit:

```
<... id="...">
 ...
 <deur>
  <Deur>
   <Deurblad>...</Deurblad>
   <HangEnSluitwerk>...</HangEnSluitwerk>
   <Bovenraam>...</Bovenraam>
  </Deur>
  ...
  <Deur>
   <Deurblad>...</Deurblad>
   <HangEnSluitwerk>...</HangEnSluitwerk>
   <Bovenraam>...</Bovenraam>
  </Deur>
 </deur>
</...>
```

We zijn hierbij verplicht alle elementen precies één maal te noemen, deze deuren hebben dus altijd een bovenraam. Het is zoals we zien wel mogelijk een onbeperkt aantal deuren aan te geven.

4.15 subTransactions

```
subTransactions : OPTIONAL SET [1:?] OF TransactionType;
```

Transacties die binnen deze transactie vallen.

Voorbeeld:

```
<TransactionType id="KomenTotWerk">
 ...
 <subTransactions>
  <TransactionType id="AcquisitieTraject">
   ...
  </TransactionType>
  <TransactionType id="OfferteTraject">
   ...
  </TransactionType>
 </subTransactions>
</TransactionType>
```

Hier zien we dat TransactionType [1.14] *KomenTotWerk* o.a. bestaat uit de TransactionType's [1.14] *AcquisitieTraject* en *OfferteTraject*.

4.16 transaction

```
transaction : TransactionType;
```

De TransactionType [1.14] binnen een MessageInTransactionType [1.5] instantie.

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <transaction>
  <TransactionType id="...">
   ...
  </TransactionType>
 </transaction>
 ...
</MessageInTransactionType>
```

4.17 transactionPhase

```
transactionPhase : OPTIONAL TransactionPhaseType;
```

De TransactionPhaseType [1.13] waarin een specifiek MessageType [1.6] binnen een specifiek TransactionType [1.14].

Voorbeeld:

```
<MessageInTransactionType id="...">
 ...
 <message>
  <MessageType id="M">
   ...
  </MessageType>
 </message>
 <transaction>
  <TransactionType id="T">
   ...
  </TransactionType>
 </transaction>
 ...
 <transactionPhase>
  <TransactionPhaseType id="TP">
   ...
  </TransactionPhaseType>
 </transactionPhase>
 ...
</MessageInTransactionType>
```

Hier vinden we dus een MessageType [1.6] *M* binnen een specifieke TransactionType [1.14] *T* die TransactionPhaseType [1.13] *TP* bepaalt.

4.18 userDefinedType

```
userDefinedType : UserDefinedType;
```

Referentie naar UserDefinedType [1.15], geeft de te gebruiken vorm van het SimpleElementType [1.12] aan.

Voorbeeld:

```
<SimpleElementType id="Hoogte">
 ...
 <userDefinedType>
  <UserDefinedType id="...">
   ...
   <baseType>INTEGER</baseType>
   ...
  </UserDefinedType>
 </userDefinedType>
</SimpleElementType>
```

Hierbij geeft userDefinedType dus voor het SimpleElementType [1.12] *Hoogte* aan dat dit altijd een integer moet zijn (eventueel met als restrictie xsdRestriction [3.27]).

< einde Bijlage 2 >