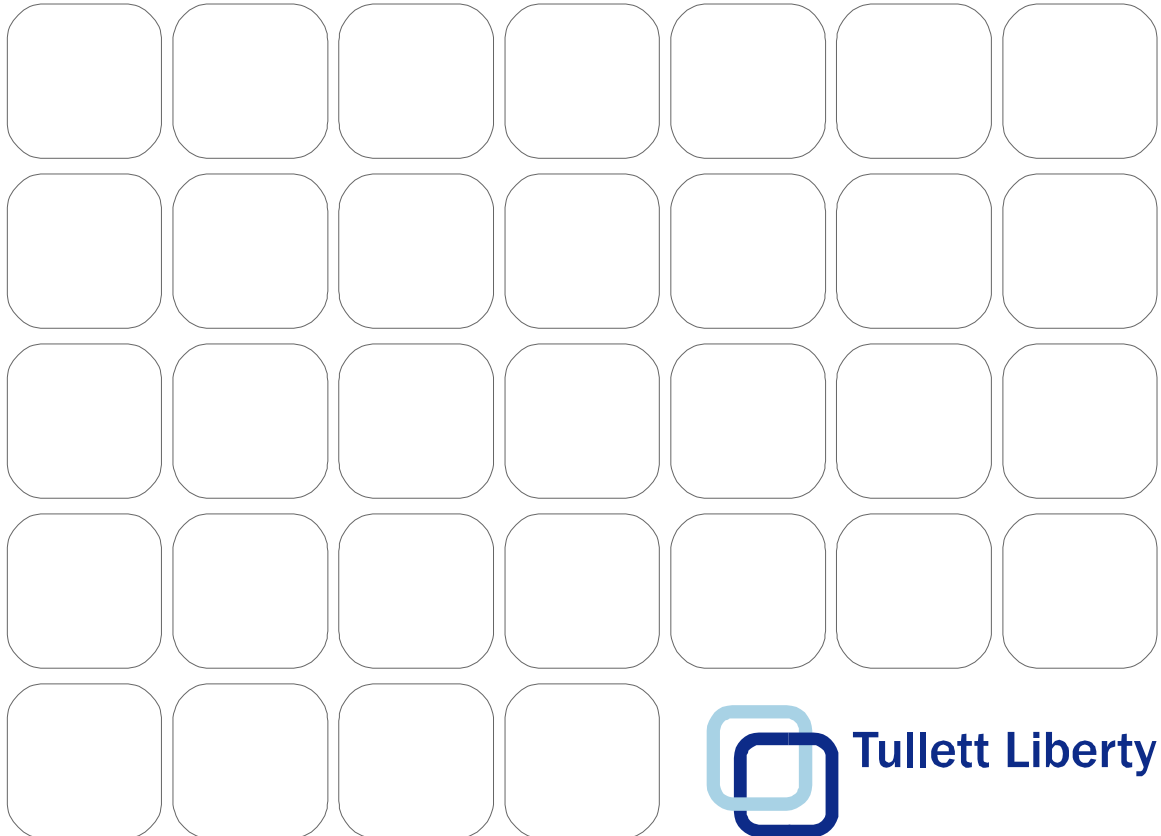


BrokerML

Business Component Technical Definitions



Tullett Liberty

Version History

Version: 2.3
Version date: 11th April 2003
Authors: Peter Meekings – Tullett & Tokyo Liberty PLC
Ashutosh Tripathi – Quantum Imagination Limited
With acknowledgements to FpML standards team

Synopsis

This document describes the Tullett standard vocabulary for exchanging deal, trade and product information– BrokerML. BML is largely based on FpML 2.0 and 3.0 standards with extensions required to support the broking business.

Document Control

Status: Issue
Authorised By:
Distribution: As required

Document History

Version:	Date:	Description:
2.2	11 th April 2003	Document created.
2.2	16 th April 2003	Released for initial review.
2.3	1 st May 2003	Styling and Look and Feel changes

References

Version:	Date:	Document:	Location
2.2	27 th March 2003	BrokerML - Overview v2.2	???
2.2	28 th March 2003	BrokerML – Coding Schemes v2.2	???
2.2	10 th April 2003	BrokerML – Business Component Framework v2.2	???
1.1	26 th February 2002	BrokerML - Design Guidelines v1.1	???
1.0	14 th May 2001	FpML 1.0	http://www.fpml.org/spec/2001/rec-fpml-1-0-2001-05-14
2.0	10 th February 2003	FpML 2.0	http://www.fpml.org/spec/2003/rec-fpml-2-0-2003-02-10
1.0	16 th March 2001	FpML Architecture 1.0	http://www.fpml.org/spec/2001/rec-fpml-arch-1-0-2001-03-16

Contents

1	Introduction	5
2	The <i>notification</i> Component.....	6
2.1	The <i>notificationHeader</i> Component	7
	The <i>notificationSTPStatuses/notificationSTPStatus</i> Component	8
3	The <i>parties/party</i> Component	9
3.1	The <i>partyContacts/partyContact</i> Component	9
	The <i>communicationMethods/communicationMethod</i> Component	10
	The <i>deal</i> Component.....	10
4	The <i>deal</i> Component.....	12
4.1	The <i>dealHeader</i> Component.....	13
	The <i>partyDealIdentifiers/partyDealIdentifier</i> Component	14
	The <i>dealDuration</i> Component.....	15
	The <i>dealSTPStatuses/dealSTPStatus</i> Component.....	16
4.2	The <i>trades/trade</i> Component	16
4.3	The <i>options/option</i> Component	17
4.4	The <i>additionalPayments/additionalPayment</i> Component	17
	The <i>paymentDate</i> Component	18
4.5	The <i>instructions/instruction</i> Component	19
5	The <i>trades/trade</i> Component	20
5.1	The <i>tradeHeader</i> component.....	21
	The <i>partyTradeIdentifiers/partyTradeIdentifier</i> Component	22
	The <i>tradeSTPStatuses/tradeSTPStatus</i> Component	23
	The <i>tradeDuration</i> Component	23
5.2	The <i>options/option</i> Component	24
5.3	The <i>additionalPayments/additionalPayment</i> Component	24
5.4	The <i>instructions/instruction</i> Component	24
6	The <i>options/option</i> Component	26
6.1	The <i>buyerOrgReferences</i> and <i>sellerOrgReferences</i> Components.....	26
6.2	The <i>premium</i> Component	27
6.3	The <i>optionStartDate</i> Component.....	27
6.4	The <i>optionExerciseDate</i> Component.....	28
6.5	The <i>optionExpiryDate</i> Component.....	28
7	The <i>product</i> Component	29

Figures

Figure 1 - Graphical representation of a deal notification	6
Figure 2 - Graphical representation of a notification header	7
Figure 3 - Graphical representation of notification STP status.....	8
Figure 4 - Graphical representation of a party in a transaction.....	9
Figure 5 - Graphical representation of party contact details.....	9
Figure 6 - Graphical representation of communication information	10
Figure 7 - Graphical representation of deal information.....	12
Figure 8 - Graphical representation of a deal header.....	13
Figure 9 - Graphical representation of deal identifiers for parties	14

Figure 10 - Graphical representation of transaction references for parties	15
Figure 11 - Graphical representation of the duration of a deal	15
Figure 12 - Graphical representation of the STP status of a deal.....	16
Figure 13 - Graphical representation of a set of trades	16
Figure 14 - Graphical representation of a set of options on the terms of a transaction.....	17
Figure 15 - Graphical representation of the additional payments for a transaction	17
Figure 16 - Graphical representation of the payment date for a transaction	18
Figure 17 - Graphical representation of the date adjustments for a transaction.....	19
Figure 18 - Graphical representation of textual instructions for a transaction.....	19
Figure 19 - Graphical representation of a trade	20
Figure 20 - Graphical representation of a trade header.....	21
Figure 21 - Graphical representation of a trade header.....	22
Figure 22 - Graphical representation of the STP status of a trade	23
Figure 23 - Graphical representation of the duration of a deal	23
Figure 24 - Graphical representation of a set of options on the terms of a transaction.....	24
Figure 25 - Graphical representation of the additional payments.....	24
Figure 26 - Graphical representation of textual instructions for a transaction.....	24
Figure 27 - Graphical representation of option information.....	26
Figure 28 - Graphical representation of organisation references.....	27
Figure 29 - Graphical representation of option premium	27
Figure 30 - Graphical representation of option start date	27
Figure 31 - Graphical representation of option exercise date	28
Figure 32 - Graphical representation of option expiry date	28
Figure 33 - Graphical representation of a product	29

Copyright © 2003 by companies in the Tullett Liberty Group.

This document contains information proprietary to companies in the Tullett Liberty Group, and may not be reproduced, disclosed or used in whole or part without the express written permission of the Tullett Liberty Group.

1 Introduction

BrokerML has been defined in the form of a business component framework that include business entities that constitute a wholesale financial transaction. These are described in the [BrokerML - Business Component Framework v2.2](#).

All the components in BrokerML are represented in XML based markup designed following the [BrokerML - Design Guidelines v1.1](#).

This document contains detailed technical XML markup definitions for the BrokerML business component framework.

The definitions contain references to the various BrokerML coding schemes, which are described in detail in [BrokerML Coding Schemes v2.2](#).

2 The *notification* Component

The *notification* is the top-level component for notification of a transaction in BrokerML. This notification has details about a deal that has been done between two or more parties.

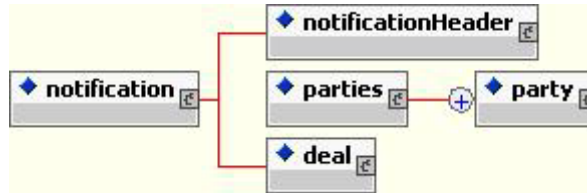


Figure 1 - Graphical representation of a deal notification

The XML definitions for *notification* entities and elements are as follows

```

...
<!ENTITY % BML Notification "notificationHeader , parties , deal">
...
<!ELEMENT notification (%BML_Notification;)>
<!ATTLIST notification type NMTOKEN #FIXED 'notification'
                        version NMTOKEN #FIXED '2.0'
                        notificationIdSchemeDefault CDATA #IMPLIED
                        notificationTypeSchemeDefault CDATA #IMPLIED
                        currencySchemeDefault CDATA #IMPLIED
                        partyIdSchemeDefault CDATA #IMPLIED
                        costCentreIdSchemeDefault CDATA #IMPLIED
                        dealTypeSchemeDefault CDATA #IMPLIED
                        tradingBookIdSchemeDefault CDATA #IMPLIED
                        transactionStatusSchemeDefault CDATA #IMPLIED
                        transactionIdSchemeDefault CDATA #IMPLIED
                        transactionVersionSchemeDefault CDATA #IMPLIED
                        businessCenterSchemeDefault CDATA #IMPLIED
                        businessDayConventionSchemeDefault CDATA #IMPLIED
                        dateRelativeToSchemeDefault CDATA #IMPLIED
                        dayCountFractionSchemeDefault CDATA #IMPLIED
                        dayTypeSchemeDefault CDATA #IMPLIED
                        periodSchemeDefault CDATA #IMPLIED
                        paymentTypeSchemeDefault CDATA #IMPLIED
                        rollConventionSchemeDefault CDATA #IMPLIED
                        payRelativeToSchemeDefault CDATA #IMPLIED
                        resetRelativeToSchemeDefault CDATA #IMPLIED
                        weeklyRollConventionSchemeDefault CDATA #IMPLIED
                        stepRelativeToSchemeDefault CDATA #IMPLIED
                        floatingRateIndexSchemeDefault CDATA #IMPLIED
                        rateTreatmentSchemeDefault CDATA #IMPLIED
                        negativeInterestRateTreatmentSchemeDefault CDATA #IMPLIED
                        payerReceiverSchemeDefault CDATA #IMPLIED
                        roundingDirectionSchemeDefault CDATA #IMPLIED
                        averagingMethodSchemeDefault CDATA #IMPLIED
                        discountingTypeSchemeDefault CDATA #IMPLIED
                        compoundingMethodSchemeDefault CDATA #IMPLIED
                        referenceBankIdSchemeDefault CDATA #IMPLIED
                        quotationRateTypeSchemeDefault CDATA #IMPLIED
                        informationProviderSchemeDefault CDATA #IMPLIED
                        rateSourcePageSchemeDefault CDATA #IMPLIED
                        calculationAgentPartySchemeDefault CDATA #IMPLIED
                        sideRateBasisSchemeDefault CDATA #IMPLIED
                        fxBarrierTypeSchemeDefault CDATA #IMPLIED
                        payoutSchemeDefault CDATA #IMPLIED
                        triggerConditionSchemeDefault CDATA #IMPLIED
                        touchConditionSchemeDefault CDATA #IMPLIED
                        strikeQuoteBasisSchemeDefault CDATA #IMPLIED
                        quoteBasisSchemeDefault CDATA #IMPLIED>
...

```

The type of the element is identified in the *type* attribute, which has a corresponding entity definition i.e. *notification* has a corresponding entity definition

BML_Notification. Types can either be BrokerML defined types or native XML types such as *string* and *date* etc.

The schemes that are used throughout the *notification* components and its sub-components are identified in the attribute list as the default schemes. These schemes can be overridden at individual component level as defined in the BrokerML design guidelines.

2.1 The *notificationHeader* Component

The *notificationHeader* has the common information regarding a notification.

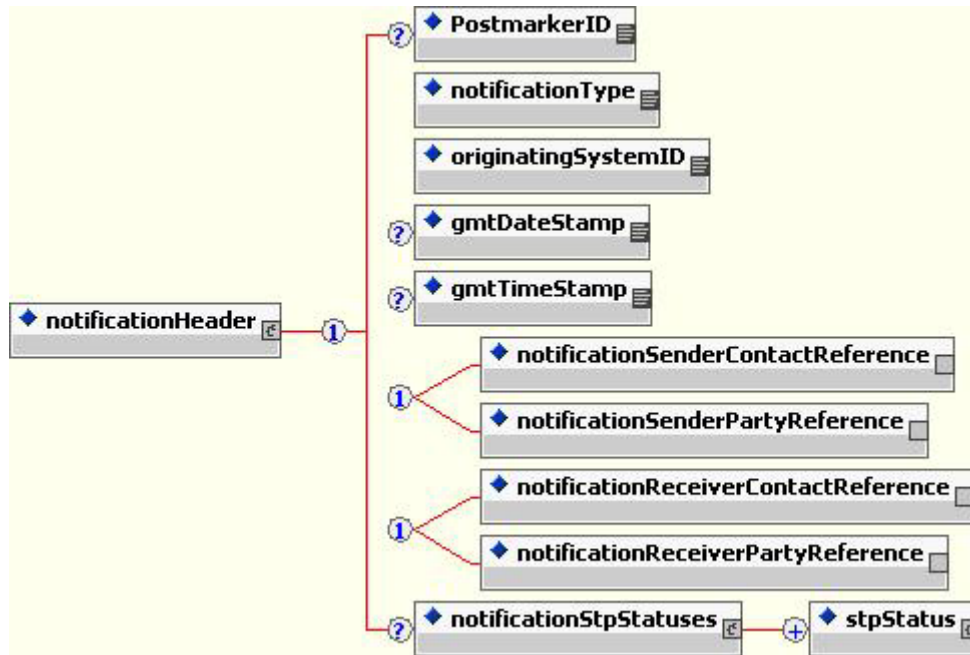


Figure 2 - Graphical representation of a notification header

The XML definitions for *notificationHeader* entities and elements are as follows

```

...
<!ENTITY % BML_NotificationHeader "PostmarkerID?, notificationType,
originatingSystemID,
                                gmtDateStamp?, gmtTimeStamp?,
                                (notificationSenderContactReference |
                                notificationSenderPartyReference),
                                (notificationReceiverContactReference |
                                notificationReceiverPartyReference),
                                notificationStpStatuses?">
...
<!ELEMENT notificationHeader (%BML_NotificationHeader;)>
<!ATTLIST notificationHeader type NMTOKEN #FIXED 'NotificationHeader'>
...
<!ELEMENT PostmarkerID (#PCDATA)>
<!ATTLIST PostmarkerID type NMTOKEN #FIXED 'string'>
...
<!ELEMENT notificationType (#PCDATA)>
<!ATTLIST notificationType type NMTOKEN #FIXED 'string'
                                notificationTypeScheme CDATA #IMPLIED>
...
<!ELEMENT gmtDateStamp (#PCDATA)>
<!ATTLIST gmtDateStamp type NMTOKEN #FIXED 'date'>
...
<!ELEMENT gmtTimeStamp (#PCDATA)>
<!ATTLIST gmtTimeStamp type NMTOKEN #FIXED 'time'>
...
<!ELEMENT notificationSenderContactReference EMPTY>
<!ATTLIST notificationSenderContactReference href CDATA #REQUIRED>
...

```

```

<!ELEMENT notificationSenderPartyReference EMPTY>
<!ATTLIST notificationSenderPartyReference href CDATA #REQUIRED>
...
<!ELEMENT notificationReceiverContactReference EMPTY>
<!ATTLIST notificationReceiverContactReference href CDATA #REQUIRED>
...
<!ELEMENT notificationReceiverPartyReference EMPTY>
<!ATTLIST notificationReceiverPartyReference href CDATA #REQUIRED>
...

```

The elements such as *notificationSenderPartyReference* use the *href* mechanism for internal references described in the BrokerML design guidelines.

The notificationSTPStatuses/notificationSTPStatus Component

The *notificationSTPStatus* contains the Straight Through Processing status of a notification.

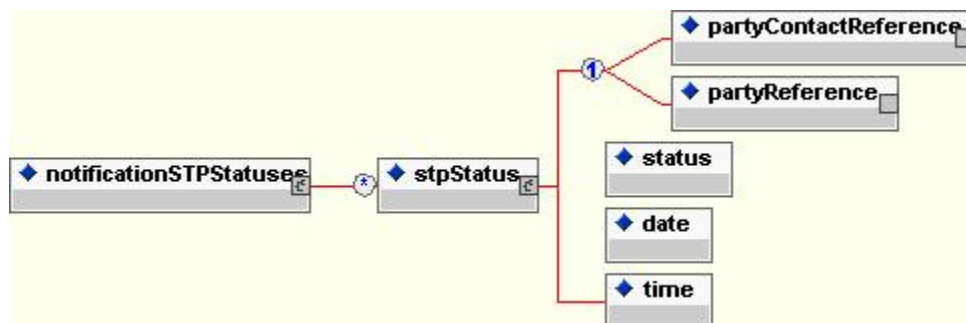


Figure 3 - Graphical representation of notification STP status

The XML definitions for *notificationSTPStatus* entities and elements are as follows

```

...
<!ENTITY % BML StpStatus "(partyContactReference | partyReference), status, date,
time">
...
<!ELEMENT notificationStpStatuses (stpStatus+)>
...
<!ELEMENT stpStatuses (stpStatus+)>
...
<!ELEMENT stpStatus (%BML_StpStatus;)>
<!ATTLIST stpStatus type NMTOKEN #FIXED 'stpStatus'>
...
<!ELEMENT partyContactReference EMPTY>
<!ATTLIST partyContactReference href CDATA #REQUIRED>
...
<!ELEMENT partyReference EMPTY>
<!ATTLIST partyReference href CDATA #REQUIRED>
...
<!ELEMENT status (#PCDATA)>
<!ATTLIST status type NMTOKEN #FIXED 'string'>
...
<!ELEMENT date (#PCDATA)>
<!ATTLIST date type NMTOKEN #FIXED 'date'>
...
<!ELEMENT time (#PCDATA)>
<!ATTLIST time type NMTOKEN #FIXED 'time'>
...

```

3 The *parties/party* Component

The *parties/party* component contains details about two or more organisations or people that have entered into a financial contract described in the *deal* component.

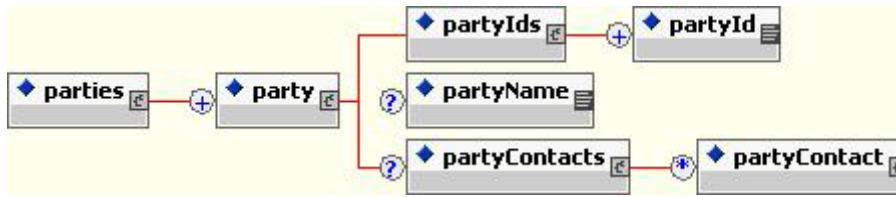


Figure 4 - Graphical representation of a party in a transaction

The XML definitions for *party* entities and elements are as follows

```

...
<!ENTITY % BML Parties "party+">
<!ENTITY % BML_Party "partyIds, partyName?, partyContacts?">
<!ENTITY % BML PartyIds "partyId+">
<!ENTITY % BML PartyContact "partyContactID, contactName,
                             preferredCommunicationMethod?,
                             communicationMethods?">
<!ENTITY % BML CommunicationMethod "communicationMedium,
                                     communicationMediumAddress">
...
<!ELEMENT party (%BML_Party;)>
<!ATTLIST party type NMTOKEN #FIXED 'Party' id ID #IMPLIED>

<!ELEMENT partyIds (%BML_PartyIds;)>
<!ATTLIST partyIds type NMTOKEN #FIXED 'PartyIds'>

<!ELEMENT parties (%BML_Parties;)>
<!ATTLIST parties type NMTOKEN #FIXED 'Parties'>

<!ELEMENT partyId (#PCDATA)>
<!ATTLIST partyId type NMTOKEN #FIXED 'string' partyIdScheme CDATA #IMPLIED>

<!ELEMENT partyName (#PCDATA)>
<!ATTLIST partyName type NMTOKEN #FIXED 'string'>

<!ELEMENT partyContactID (#PCDATA)>
<!ATTLIST partyContactID type NMTOKEN #FIXED 'string'>
...

```

3.1 The *partyContacts/partyContact* Component

The *partyContact* component holds contact information about people etc. of parties involved in a transaction.

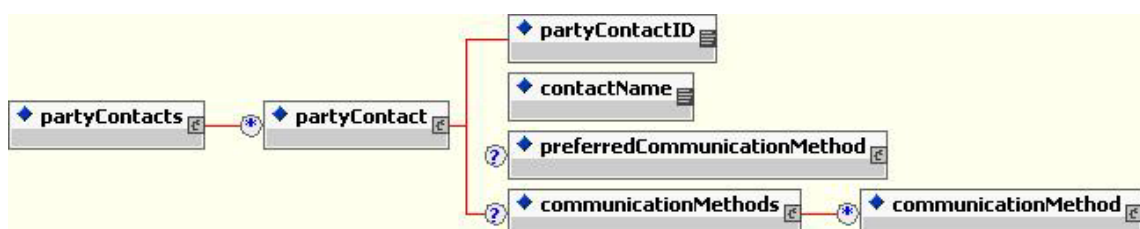


Figure 5 - Graphical representation of party contact details

The XML definitions for *partyContact* entities and elements are as follows

```

...
<!ENTITY % BML PartyContact "partyContactID, contactName,
                             preferredCommunicationMethod?,
                             communicationMethods?">

<!ENTITY % BML_CommunicationMethod "communicationMedium,
                                     communicationMediumAddress">
...
<!ELEMENT partyContacts (partyContact*)>

<!ELEMENT partyContact (%BML PartyContact;)>
<!ATTLIST partyContact type NMTOKEN #FIXED 'PartyContact' id ID #IMPLIED>

<!ELEMENT contactName (#PCDATA)>
<!ATTLIST contactName type NMTOKEN #FIXED 'string'>
<!ELEMENT preferredCommunicationMethod (%BML CommunicationMethod;)>
<!ATTLIST preferredCommunicationMethod type NMTOKEN #FIXED 'CommunicationMethod'>

<!ELEMENT communicationMethods (communicationMethod*)>

<!ELEMENT communicationMethod (%BML CommunicationMethod;)>
<!ATTLIST communicationMethod type NMTOKEN #FIXED 'CommunicationMethod'>
...

```

The *communicationMethods/communicationMethod* Component

The *communicationMethod* component contains information about means of communication for a party contact.



Figure 6 - Graphical representation of communication information

The XML definitions for *communicationMethod* entities and elements are as follows

```

...
<!ENTITY % BML CommunicationMethod "communicationMedium,
                                     communicationMediumAddress">
...
<!ELEMENT communicationMethods (communicationMethod*)>

<!ELEMENT communicationMethod (%BML CommunicationMethod;)>
<!ATTLIST communicationMethod type NMTOKEN #FIXED 'CommunicationMethod'>

<!ELEMENT communicationMedium (#PCDATA)>
<!ATTLIST communicationMedium type NMTOKEN #FIXED 'string'>

<!ELEMENT communicationMediumAddress (#PCDATA)>
<!ATTLIST communicationMediumAddress type NMTOKEN #FIXED 'string'>
...

```

The *deal* Component

The *deal* component contains details about an agreement between two or more parties to enter into a financial contract involving one or more trades. The *deal* element contains all the information required to confirm this agreement.

The XML definitions for *deal* entities and elements are as follows

```

...
<!ENTITY % BML Deal "dealHeader, trades, dealPrice?, options?,
                    additionalPayments?, instructions?">
...
<!ELEMENT deal (%BML_Deal;)>

```

```
<!ATTLIST deal tvpe NMOKEN #FIXED 'Deal'>  
...
```

4 The *deal* Component

A deal is a set of financial transaction contracts entered exchange and agreed between two or more parties. The *deal* component is the top level BML element containing details about such a contract involving one or more trades. The *deal* element contains all the information required to confirm this contract.

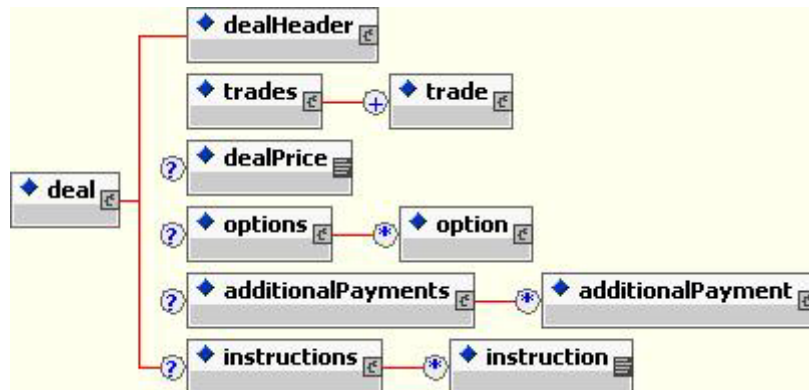


Figure 7 - Graphical representation of deal information

The XML definitions for *deal* entities and elements are as follows

```

...
<!ENTITY % BML Deal "dealHeader, trades, dealPrice?, options?,
                    additionalPayments?, instructions?">
...
<!ENTITY % BML DealHeader "dealType, dealHeaderBanners,
                           dealTrailerBanners, dealDescription,
                           dealStatus, dealDate, dealTime,
                           dealValueDate, dealSettlementDate,
                           dealDuration?, partyDealIdentifiers,
                           dealStpStatuses?">
...
<!ENTITY % BML Trade "tradeHeader, product, options?,
                     additionalPayments?, instructions?">
...
<!ENTITY % BML Option "buyerOrgReferences, sellerOrgReferences, (callAmount |
putAmount), premium, strikePrice, optionStartDate, optionExerciseDate,
optionExpiryDate, (%BML ExerciseSelection;), exerciseProcedure,
calculationAgentPartyReference+, cashSettlement?">
...
<!ENTITY % BML Payment "payerPartyReference, receiverPartyReference,
                        paymentAmount, paymentType?, paymentDate?,
                        adjustedPaymentDate?">
...
<!ELEMENT deal (%BML_Deal;)>
<!ATTLIST deal type NMTOKEN #FIXED 'Deal'>
...
<!ELEMENT dealHeader (%BML_DealHeader;)>
<!ATTLIST dealHeader type NMTOKEN #FIXED 'DealHeader'>
...
<!ELEMENT trades (trade+)>
...
<!ELEMENT trade (%BML_Trade;)>
<!ATTLIST trade type NMTOKEN #FIXED 'Trade'>
...
<!ELEMENT dealPrice (#PCDATA)>
<!ATTLIST dealPrice type NMTOKEN #FIXED 'decimal'>
...
<!ELEMENT options (option*)>
...
<!ELEMENT option (%BML_Option;)>
<!ATTLIST option type NMTOKEN #FIXED 'Option'>
...
<!ELEMENT additionalPayments (additionalPayment*)>
...
<!ELEMENT additionalPayment (%BML_Payment;)>

```

```

<!ATTLIST additionalPavment tvpe NMTOKEN #FIXED 'Pavment'>
...
<!ELEMENT instructions (instruction*)>

<!ELEMENT instruction (#PCDATA)>
<!ATTLIST instruction type NMTOKEN #FIXED 'string'>
...

```

4.1 The *dealHeader* Component

The *dealHeader* holds information about a deal which is common to the underlying trades and options.

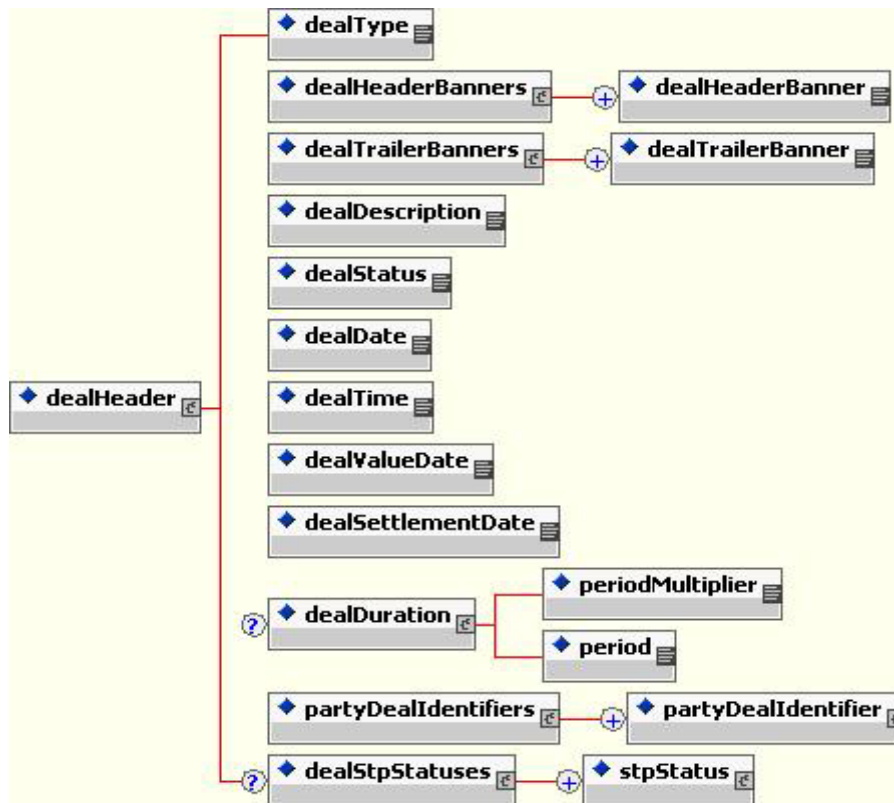


Figure 8 - Graphical representation of a deal header

The XML definitions for *dealHeader* entities and elements are as follows

```

...
<!ENTITY % BML_DealHeader "dealType, dealHeaderBanners, dealTrailerBanners,
dealDescription, dealStatus, dealDate, dealTime, dealValueDate, dealSettlementDate,
dealDuration?, partyDealIdentifiers, dealStpStatuses?">
...
<!ENTITY % BML_PartyTransactionIdentifier "partyReference, transactionReference,
previousTransactionReference?, linkedTransactionReferences?">
...
<!ELEMENT dealType (#PCDATA)>
<!ATTLIST dealType type NMTOKEN #FIXED 'string' dealTypeScheme CDATA #IMPLIED>
...
<!ELEMENT dealHeaderBanners (dealHeaderBanner+)>

<!ELEMENT dealHeaderBanner (#PCDATA)>
<!ATTLIST dealHeaderBanner type NMTOKEN #FIXED 'string'>

<!ELEMENT dealTrailerBanners (dealTrailerBanner+)>

<!ELEMENT dealTrailerBanner (#PCDATA)>
<!ATTLIST dealTrailerBanner type NMTOKEN #FIXED 'string'>
...
<!ELEMENT dealDescription (#PCDATA)>
<!ATTLIST dealDescription type NMTOKEN #FIXED 'string'>

```

```

...
<!ELEMENT dealStatus (#PCDATA)>
<!ATTLIST dealStatus type NMTOKEN #FIXED 'string' transactionStatusScheme CDATA
#IMPLIED>
...
<!ELEMENT dealDate (#PCDATA)>
<!ATTLIST dealDate type NMTOKEN #FIXED 'date'>
...
<!ELEMENT dealTime (#PCDATA)>
<!ATTLIST dealTime type NMTOKEN #FIXED 'time'>
...
<!ELEMENT dealValueDate (#PCDATA)>
<!ATTLIST dealValueDate type NMTOKEN #FIXED 'date'>
...
<!ELEMENT dealSettlementDate (#PCDATA)>
<!ATTLIST dealSettlementDate type NMTOKEN #FIXED 'date'>
...
<!ELEMENT dealDuration (%BML_Interval;)>
<!ATTLIST dealDuration type NMTOKEN #FIXED 'Interval'>
...
<!ELEMENT partyDealIdentifiers (partyDealIdentifier+)>
...
<!ELEMENT partyDealIdentifier (%BML_PartyTransactionIdentifier;)>
<!ATTLIST partyDealIdentifier type NMTOKEN #FIXED 'PartyTransactionIdentifier'>
...
<!ELEMENT dealStpStatuses (stpStatus+)>
...

```

The partyDealIdentifiers/partyDealIdentifier Component

The *partyDealIdentifiers* / *partyDealIdentifier* holds the information that a particular party to the trade uses to identify the deal.

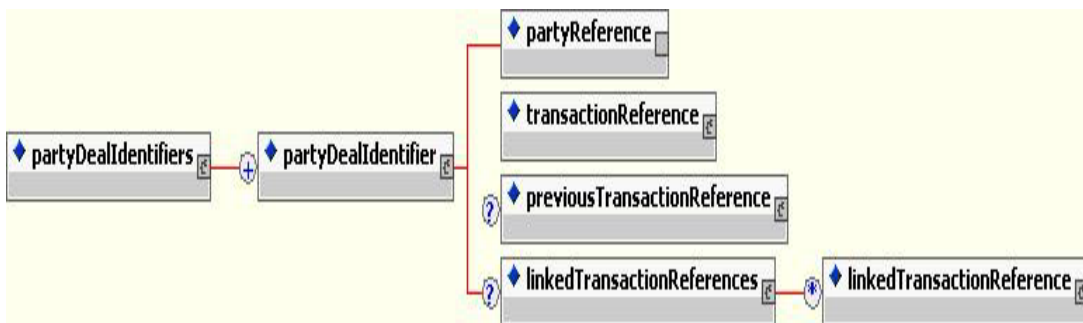


Figure 9 - Graphical representation of deal identifiers for parties

The XML definitions for *partyDealIdentifier* entities and elements are as follows

```

...
<!ENTITY % BML_PartyTransactionIdentifier "partyReference, transactionReference,
previousTransactionReference?, linkedTransactionReferences?">
...
<!ENTITY % BML_TransactionReference "transactionId, transactionVersion">
...
<!ELEMENT partyDealIdentifier (%BML_PartyTransactionIdentifier;)>
<!ATTLIST partyDealIdentifier type NMTOKEN #FIXED 'PartyTransactionIdentifier'>
<!ELEMENT partyDealIdentifiers (partyDealIdentifier+)>
...
<!ELEMENT partyReference EMPTY>
<!ATTLIST partyReference href CDATA #REQUIRED>
...
<!ELEMENT transactionReference (%BML_TransactionReference;)>
<!ATTLIST transactionReference type NMTOKEN #FIXED 'TransactionReference'>
...
<!ELEMENT previousTransactionReference (%BML_TransactionReference;)>
<!ATTLIST previousTransactionReference type NMTOKEN #FIXED 'TransactionReference'>
...
<!ELEMENT linkedTransactionReference (%BML_TransactionReference;)>
<!ATTLIST linkedTransactionReference type NMTOKEN #FIXED 'TransactionReference'>

```

```
<!ELEMENT linkedTransactionReferences (linkedTransactionReference*)>
...
```

4.1.1.1. The transactionReferences Component

The *transactionReference* component holds information about the reference that a party to a transaction uses to identify it.

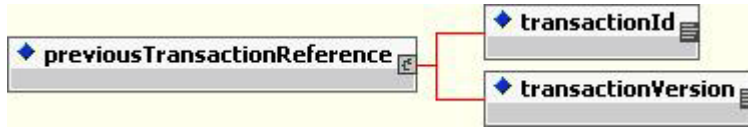


Figure 10 - Graphical representation of transaction references for parties

The XML definitions for *partyDealIdentifier* entities and elements are as follows

```

...
<!ENTITY % BML TransactionReference "transactionId, transactionVersion">
...
<!ELEMENT transactionReference (%BML_TransactionReference;)>
<!ATTLIST transactionReference type NMTOKEN #FIXED 'TransactionReference'>
...
<!ELEMENT transactionId (#PCDATA)>
<!ATTLIST transactionId type NMTOKEN #FIXED 'string' transactionIdScheme CDATA
#IMPLIED>
...
<!ELEMENT transactionVersion (#PCDATA)>
<!ATTLIST transactionVersion type NMTOKEN #FIXED 'string' transactionVersionScheme
CDATA #IMPLIED>
...

```

The dealDuration Component

The *dealDuration* contains the duration for which the deal is valid.

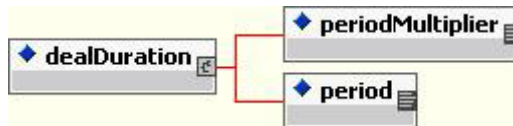


Figure 11 - Graphical representation of the duration of a deal

```

...
<!ENTITY % BML Interval "periodMultiplier, period">
...
<!ELEMENT dealDuration (%BML Interval;)>
<!ATTLIST dealDuration type NMTOKEN #FIXED 'Interval'>
...
<!ELEMENT period (#PCDATA)>
<!ATTLIST period type NMTOKEN #FIXED 'string' periodScheme CDATA #IMPLIED>
...
<!ELEMENT periodMultiplier (#PCDATA)>
<!ATTLIST periodMultiplier type NMTOKEN #FIXED 'integer'>
...

```

The dealSTPStatuses/dealSTPStatus Component

The *dealSTPStatuses/dealSTPStatus* contains the STP status of the deal.

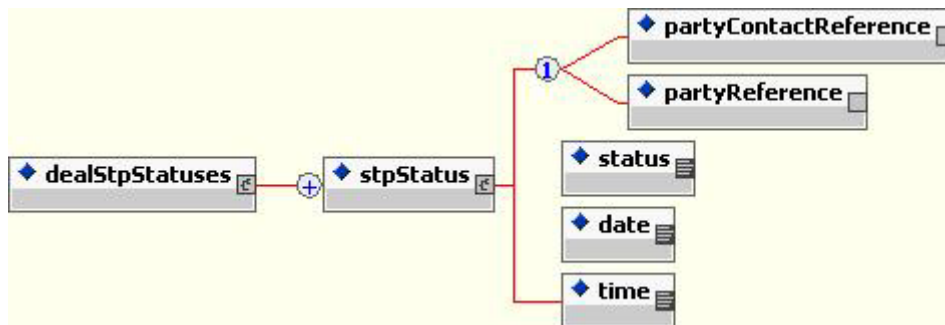


Figure 12 - Graphical representation of the STP status of a deal

The XML definitions for *dealSTPStatus* entities and elements are as follows

```

...
<!ENTITY % BML StpStatus "(partyContactReference | partyReference), status, date,
time">
...
<!ELEMENT dealStpStatuses (stpStatus+)>
...
<!ELEMENT stpStatuses (stpStatus+)>
...
<!ELEMENT stpStatus (%BML_StpStatus;)>
<!ATTLIST stpStatus type NMTOKEN #FIXED 'stpStatus'>
...
<!ELEMENT partyContactReference EMPTY>
<!ATTLIST partyContactReference href CDATA #REQUIRED>
...
<!ELEMENT partyReference EMPTY>
<!ATTLIST partyReference href CDATA #REQUIRED>
...
<!ELEMENT status (#PCDATA)>
<!ATTLIST status type NMTOKEN #FIXED 'string'>
...
<!ELEMENT date (#PCDATA)>
<!ATTLIST date type NMTOKEN #FIXED 'date'>
...
<!ELEMENT time (#PCDATA)>
<!ATTLIST time type NMTOKEN #FIXED 'time'>
...

```

4.2 The trades/trade Component

The *trades/trade* component in BML contains the economic information necessary to execute and confirm that trade. This includes a product definition, any optional components, any payments to be made to parties other than the principal parties (e.g. brokerage fee, clearing charges etc.) and any manual instruction.

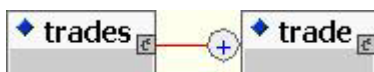


Figure 13 - Graphical representation of a set of trades

The XML definitions for *trade* entities and elements are as follows

```

...
<!ENTITY % BML_Trade "tradeHeader, product, options?,
additionalPayments?, instructions?">
...

```

```

<!ELEMENT trades (trade+)>
<!ELEMENT trade (%BML Trade;)>
<!ATTLIST trade type NMTOKEN #FIXED 'Trade'
id ID #IMPLIED>
...

```

4.3 The *options*/*option* Component

The *option* component in BML contains the information associated with an option granted by one party to another in order to be able to change the completion characteristics of a transaction.

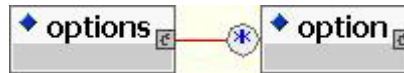


Figure 14 - Graphical representation of a set of options on the terms of a transaction

The XML definitions for *option* entities and elements are as follows

```

...
<!ENTITY % BML_Option "buyerOrgReferences, sellerOrgReferences, (callAmount |
putAmount), premium, strikePrice, optionStartDate, optionExerciseDate,
optionExpiryDate, (%BML ExerciseSelection;), exerciseProcedure,
calculationAgentPartyReference+, cashSettlement?">
...
<!ELEMENT options (option*)>
...
<!ELEMENT option (%BML_Option;)>
<!ATTLIST option type NMTOKEN #FIXED 'Option'>
...

```

4.4 The *additionalPayments*/*additionalPayment* Component

The *additionalPayments*/*additionalPayment* component contains information about payments such as brokerage paid to third parties which are not part of the economics of a transaction itself.

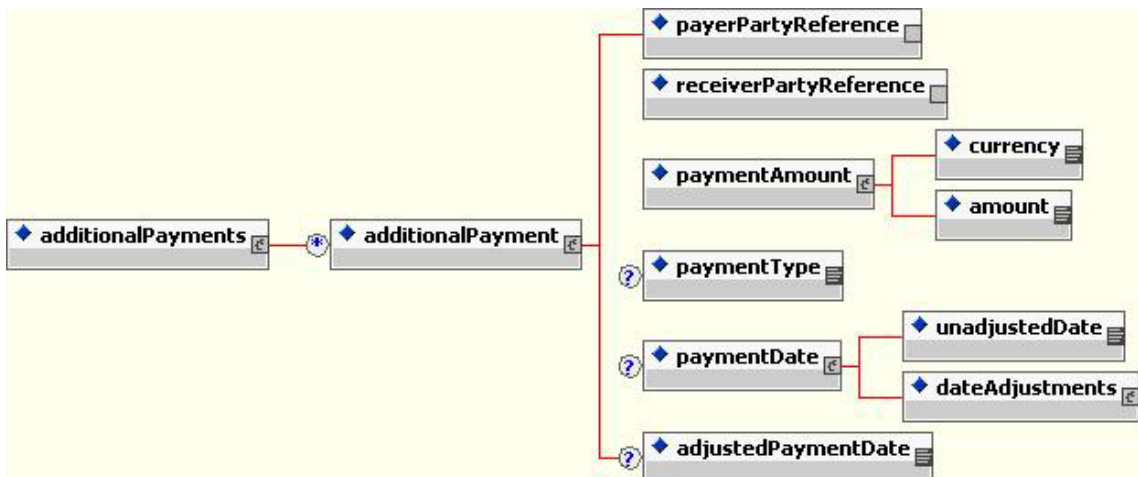


Figure 15 - Graphical representation of the additional payments for a transaction

The XML definitions for *additionalPayment* entities and elements are as follows

```

...
<!ENTITY % BML_Payment "payerPartyReference, receiverPartyReference, paymentAmount,
paymentType?, paymentDate?, adjustedPaymentDate?">
...

```

```

...
<!ENTITY % BML_AdjustableDate "unadjustedDate, dateAdjustments">
...
<!ENTITY % BML_BusinessDayAdjustments "businessDayConvention,
(businessCentersReference | businessCenters)?">
...
<!ELEMENT additionalPayments (additionalPayment*)>

<!ELEMENT additionalPayment (%BML_Payment;)>
<!ATTLIST additionalPayment type NMTOKEN #FIXED 'Payment'>

<!ELEMENT payerPartyReference EMPTY>
<!ATTLIST payerPartyReference href CDATA #REQUIRED>

<!ELEMENT receiverPartyReference EMPTY>
<!ATTLIST receiverPartyReference href CDATA #REQUIRED>

<!ELEMENT paymentAmount (%BML_Money;)>
<!ATTLIST paymentAmount type NMTOKEN #FIXED 'Money'>

<!ELEMENT currency (#PCDATA)>
<!ATTLIST currency type NMTOKEN #FIXED 'string' currencyScheme CDATA #IMPLIED>

<!ELEMENT amount (#PCDATA)>
<!ATTLIST amount type NMTOKEN #FIXED 'decimal'>

<!ELEMENT paymentType (#PCDATA)>
<!ATTLIST paymentType type NMTOKEN #FIXED 'string' paymentTypeScheme CDATA #IMPLIED>

<!ELEMENT paymentDate (%BML_AdjustableDate;)>
<!ATTLIST paymentDate type NMTOKEN #FIXED 'AdjustableDate'>
...
<!ELEMENT unadjustedDate (#PCDATA)>
<!ATTLIST unadjustedDate type NMTOKEN #FIXED 'date'>

<!ELEMENT dateAdjustments (%BML_BusinessDayAdjustments;)>
<!ATTLIST dateAdjustments type NMTOKEN #FIXED 'BusinessDayAdjustments'>
...
<!ELEMENT adjustedPaymentDate (#PCDATA)>
<!ATTLIST adjustedPaymentDate type NMTOKEN #FIXED 'date'>
...

```

The paymentDate Component

The *paymentDate* component holds information about ways to calculate a future payment date based on the supplied parameters.

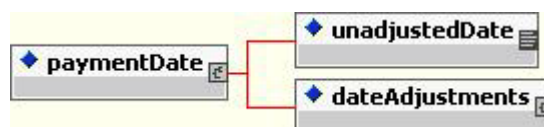


Figure 16 - Graphical representation of the payment date for a transaction

The XML definitions for *paymentDate* entities and elements are as follows

```

...
<!ENTITY % BML_AdjustableDate "unadjustedDate, dateAdjustments">
...
<!ENTITY % BML_BusinessDayAdjustments "businessDayConvention,
(businessCentersReference | businessCenters)?">
...
<!ELEMENT unadjustedDate (#PCDATA)>
<!ATTLIST unadjustedDate type NMTOKEN #FIXED 'date'>

<!ELEMENT dateAdjustments (%BML_BusinessDayAdjustments;)>
<!ATTLIST dateAdjustments type NMTOKEN #FIXED 'BusinessDayAdjustments'>
...

```

4.4.1.1. The dateAdjustments Component

The *dateAdjustments* component holds information about ways to adjust a future date.



Figure 17 - Graphical representation of the date adjustments for a transaction

The XML definitions for *dateAdjustments* entities and elements are as follows

```

...
<!ENTITY % BML_BusinessDayAdjustments "businessDayConvention,
(businessCentersReference | businessCenters)?">
...
<!ELEMENT dateAdjustments (%BML BusinessDayAdjustments;)>
<!ATTLIST dateAdjustments type NMTOKEN #FIXED 'BusinessDayAdjustments'>
...
<!ELEMENT businessDayConvention (#PCDATA)>
<!ATTLIST businessDayConvention type NMTOKEN #FIXED 'string'
businessDayConventionScheme CDATA #IMPLIED>

<!ELEMENT businessCenters (businessCenter+)>
<!ATTLIST businessCenters id ID #REQUIRED>

<!ELEMENT businessCenter (#PCDATA)>
<!ATTLIST businessCenter type NMTOKEN #FIXED 'string' businessCenterScheme CDATA
#IMPLIED>

<!ELEMENT businessCentersReference EMPTY>
<!ATTLIST businessCentersReference href CDATA #IMPLIED>
...

```

4.5 The *instructions/instruction* Component

The *instructions/instruction* component contains any instructions for settlement, confirmation etc. specified by any of the parties to the deal.

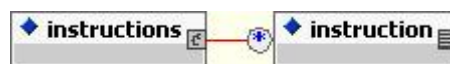


Figure 18 - Graphical representation of textual instructions for a transaction

The XML definitions for *instruction* entities and elements are as follows

```

...
<!ELEMENT instructions (instruction*)>

<!ELEMENT instruction (#PCDATA)>
<!ATTLIST instruction type NMTOKEN #FIXED 'string'>
...

```

5 The *trades/trade* Component

A trade is an agreement between two parties to enter into a financial contract for exchanging products in return for a payment. The *trades/trade* component in BML contains the economic information necessary to execute individual transactions that are part of the overall transaction contained in a *deal* component.

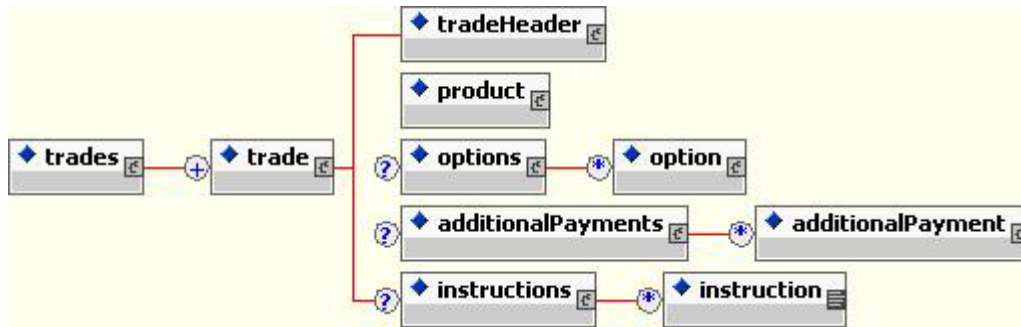


Figure 19 - Graphical representation of a trade

The XML definitions for *trade* entities and elements are as follows

```

...
<!ENTITY % BML_Trade "tradeHeader , product , options? , additionalPayments? ,
instructions?">

<!ENTITY % BML_TradeHeader "tradeType , tradeStatus , tradeDate , tradeTime ,
tradeValueDate , partyTradeIdentifiers , tradeSettlementDate? , tradeStpStatuses? ,
tradeDuration?">
...
<!ENTITY % BML_Product "termDeposit | fra | bond | repo | swap | future |
fxSingleLeg | fxOption">
...
<!ENTITY % BML_Option "buyerOrgReferences, sellerOrgReferences, (callAmount |
putAmount), premium, strikePrice, optionStartDate, optionExerciseDate,
optionExpiryDate, (%BML_ExerciseSelection;), exerciseProcedure,
calculationAgentPartyReference+, cashSettlement?">
...
<!ENTITY % BML_Payment "payerPartyReference, receiverPartyReference, paymentAmount,
paymentType?, paymentDate?, adjustedPaymentDate?">
...
<!ELEMENT trades (trade+)>

<!ELEMENT trade (%BML_Trade;)>
<!ATTLIST trade type NMTOKEN #FIXED 'Trade' id ID #IMPLIED>

<!ELEMENT tradeHeader (%BML_TradeHeader;)>
<!ATTLIST tradeHeader type NMTOKEN #FIXED 'TradeHeader'>
...
<!ELEMENT product (%BML_Product;)>
<!ATTLIST product type NMTOKEN #FIXED 'Product'>
...
<!ELEMENT options (option*)>

<!ELEMENT option (%BML_Option;)>
<!ATTLIST option type NMTOKEN #FIXED 'Option'>
...
<!ELEMENT additionalPayments (additionalPayment*)>

<!ELEMENT additionalPayment (%BML_Payment;)>
<!ATTLIST additionalPayment type NMTOKEN #FIXED 'Payment'>
...
<!ELEMENT instructions (instruction*)>

<!ELEMENT instruction (#PCDATA)>
<!ATTLIST instruction type NMTOKEN #FIXED 'string'>
...

```

5.1 The *tradeHeader* component

The information common across all the components of a trade is contained within the *tradeHeader* component.

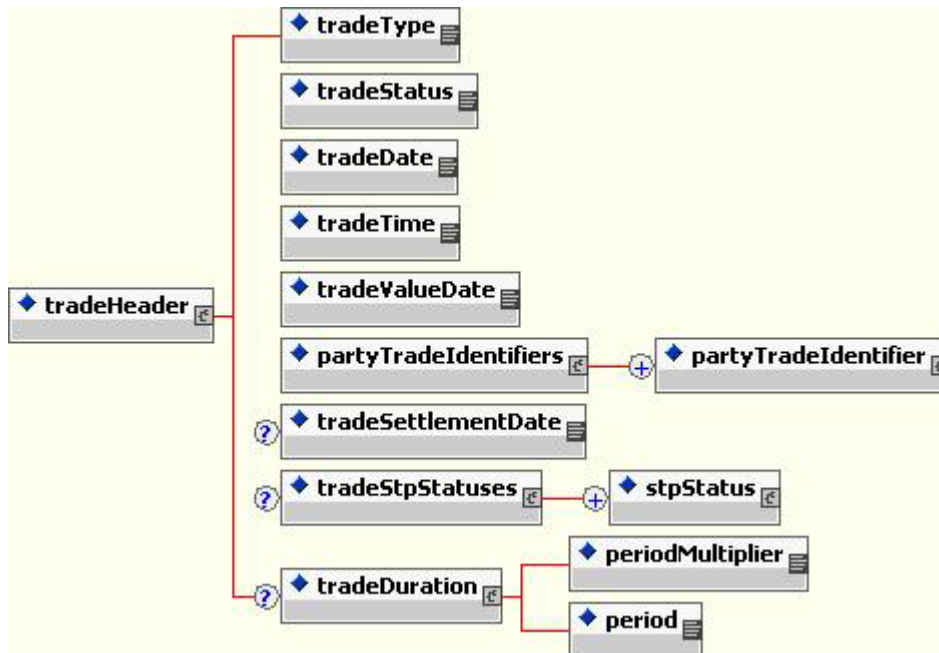


Figure 20 - Graphical representation of a trade header

The XML definitions for *tradeHeader* entities and elements are as follows

```

...
<!ENTITY % BML TradeHeader "tradeType , tradeStatus , tradeDate , tradeTime ,
tradeValueDate , partyTradeIdentifiers , tradeSettlementDate? , tradeStpStatuses? ,
tradeDuration?">
...
<!ENTITY % BML PartyTransactionIdentifier "partyReference, transactionReference,
previousTransactionReference?, linkedTransactionReferences?">
...
<!ENTITY % BML StpStatus "(partyContactReference | partyReference), status, date,
time">
...
<!ENTITY % BML Interval "periodMultiplier, period">
...
<!ELEMENT tradeType (#PCDATA)>
<!ATTLIST tradeType type NMTOKEN #FIXED 'string' tradeTypeScheme CDATA #IMPLIED>

<!ELEMENT tradeStatus (#PCDATA)>
<!ATTLIST tradeStatus type NMTOKEN #FIXED 'string' transactionStatusScheme CDATA
#IMPLIED>

<!ELEMENT tradeDate (#PCDATA)>
<!ATTLIST tradeDate type NMTOKEN #FIXED 'date'>

<!ELEMENT tradeTime (#PCDATA)>
<!ATTLIST tradeTime type NMTOKEN #FIXED 'time'>

<!ELEMENT tradeValueDate (#PCDATA)>
<!ATTLIST tradeValueDate type NMTOKEN #FIXED 'date'>

<!ELEMENT partyTradeIdentifiers (partyTradeIdentifier+)>

<!ELEMENT partyTradeIdentifier (%BML PartyTransactionIdentifier;)>
<!ATTLIST partyTradeIdentifier type NMTOKEN #FIXED 'PartyTransactionIdentifier'
id ID #IMPLIED>

<!ELEMENT tradeSettlementDate (#PCDATA)>
<!ATTLIST tradeSettlementDate type NMTOKEN #FIXED 'date'>

```

```

<!ELEMENT tradeStpStatuses (stpStatus+)>
...
<!ELEMENT tradeDuration (%BML Interval;)>
<!ATTLIST tradeDuration type NMTOKEN #FIXED 'Interval'>
...
<!ELEMENT stpStatus (%BML StpStatus;)>
<!ATTLIST stpStatus type NMTOKEN #FIXED 'stpStatus'>
...

```

The partyTradeIdentifiers/partyTradeIdentifier Component

The *partyTradeIdentifier* contains the information that a particular party to the trade uses to identify the trade.

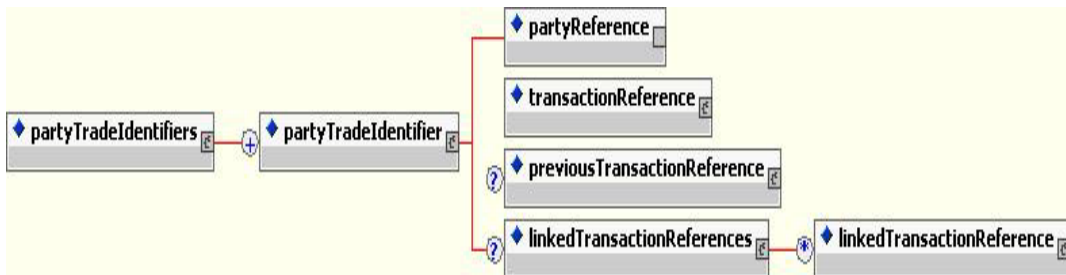


Figure 21 - Graphical representation of a trade header

The XML definitions for *partyTradeIdentifier* entities and elements are as follows

```

...
<!ENTITY % BML_PartyTransactionIdentifier "partyReference, transactionReference,
previousTransactionReference?, linkedTransactionReferences?">
...
<!ENTITY % BML TransactionReference "transactionId, transactionVersion">
...
<!ELEMENT partyTradeIdentifiers (partyTradeIdentifier+)>

<!ELEMENT partyTradeIdentifier (%BML_PartyTransactionIdentifier;)>
<!ATTLIST partyTradeIdentifier type NMTOKEN #FIXED 'PartyTransactionIdentifier'
id ID #IMPLIED>
...
<!ELEMENT partyReference EMPTY>
<!ATTLIST partyReference href CDATA #REQUIRED>

<!ELEMENT transactionReference (%BML TransactionReference;)>
<!ATTLIST transactionReference type NMTOKEN #FIXED 'TransactionReference'>

<!ELEMENT previousTransactionReference (%BML TransactionReference;)>
<!ATTLIST previousTransactionReference type NMTOKEN #FIXED 'TransactionReference'>

<!ELEMENT linkedTransactionReferences (linkedTransactionReference*)>

<!ELEMENT linkedTransactionReference (%BML TransactionReference;)>
<!ATTLIST linkedTransactionReference type NMTOKEN #FIXED 'TransactionReference'>
...

```

The tradeSTPStatuses/tradeSTPStatus Component

The *tradeSTPStatuses/tradeSTPStatus* contains the STP status of the deal.

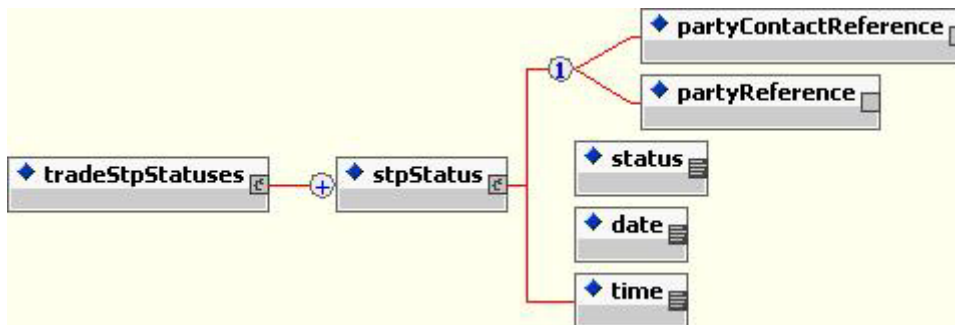


Figure 22 - Graphical representation of the STP status of a trade

The XML definitions for *tradeSTPStatus* entities and elements are as follows

```

...
<!ENTITY % BML StpStatus "(partyContactReference | partyReference), status, date,
time">
...
<!ELEMENT partyContactReference EMPTY>
<!ATTLIST partyContactReference href CDATA #REQUIRED>
...
<!ELEMENT partyReference EMPTY>
<!ATTLIST partyReference href CDATA #REQUIRED>
...
<!ELEMENT status (#PCDATA)>
<!ATTLIST status type NMTOKEN #FIXED 'string'>

<!ELEMENT date (#PCDATA)>
<!ATTLIST date type NMTOKEN #FIXED 'date'>

<!ELEMENT time (#PCDATA)>
<!ATTLIST time type NMTOKEN #FIXED 'time'>
...

```

The tradeDuration Component

The *tradeDuration* contains the duration for which the deal is valid.

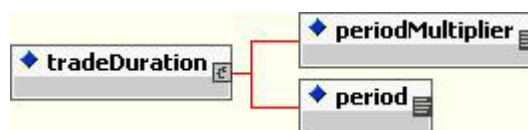


Figure 23 - Graphical representation of the duration of a deal

The XML definitions for *tradeDuration* entities and elements are as follows

```

...
<!ENTITY % BML Interval "periodMultiplier, period">
...
<!ELEMENT tradeDuration (%BML_Interval;)>
<!ATTLIST tradeDuration type NMTOKEN #FIXED 'Interval'>
...

```

It consists of

- A period definition – *period*.
- A multiplier, which combined with the period, results in the actual duration – *periodMultiplier*.

5.2 The *options*/*option* Component

The *option* component in BML contains the information associated with an option granted by one party to another in order to be able to change the completion characteristics of a transaction.

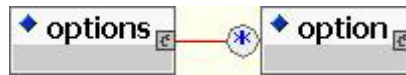


Figure 24 - Graphical representation of a set of options on the terms of a transaction

The XML definitions for *option* entities and elements are as follows

```
...
<!ENTITY % BML Option "buyerOrgReferences, sellerOrgReferences, (callAmount |
putAmount), premium, strikePrice, optionStartDate, optionExerciseDate,
optionExpiryDate, (%BML ExerciseSelection;), exerciseProcedure,
calculationAgentPartyReference+, cashSettlement?">
...
<!ELEMENT options (option*)>
...
<!ELEMENT option (%BML Option;)>
<!ATTLIST option type NMTOKEN #FIXED 'Option'>
...
```

5.3 The *additionalPayments*/*additionalPayment* Component

The *additionalPayments*/*additionalPayment* component contains information about payments such as brokerage paid to third parties which are not part of the economics of a transaction itself.

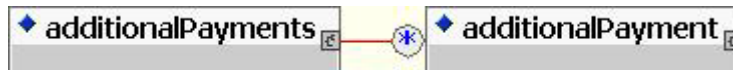


Figure 25 - Graphical representation of the additional payments

The XML definitions for *additionalPayment* entities and elements are as follows

```
...
<!ENTITY % BML Payment "payerPartyReference, receiverPartyReference, paymentAmount,
paymentType?, paymentDate?, adjustedPaymentDate?">
...
<!ELEMENT additionalPayments (additionalPayment*)>
...
<!ELEMENT additionalPayment (%BML Payment;)>
<!ATTLIST additionalPayment type NMTOKEN #FIXED 'Payment'>
...
```

5.4 The *instructions*/*instruction* Component

The *instruction* component contains any instructions for settlement, confirmation etc. specified by any of the parties to the deal.

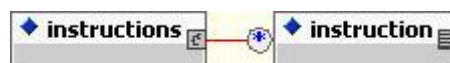


Figure 26 - Graphical representation of textual instructions for a transaction

The XML definitions for *instruction* entities and elements are as follows

```
...  
<!ELEMENT instructions (instruction*)>  
  
<!ELEMENT instruction (#PCDATA)>  
<!ATTLIST instruction type NMTOKEN #FIXED 'string'>  
  
...
```

6 The *options*/*option* Component

The *option* component in BML contains the information associated with an option granted by one party to another in order to be able to change the completion characteristics of a transaction.

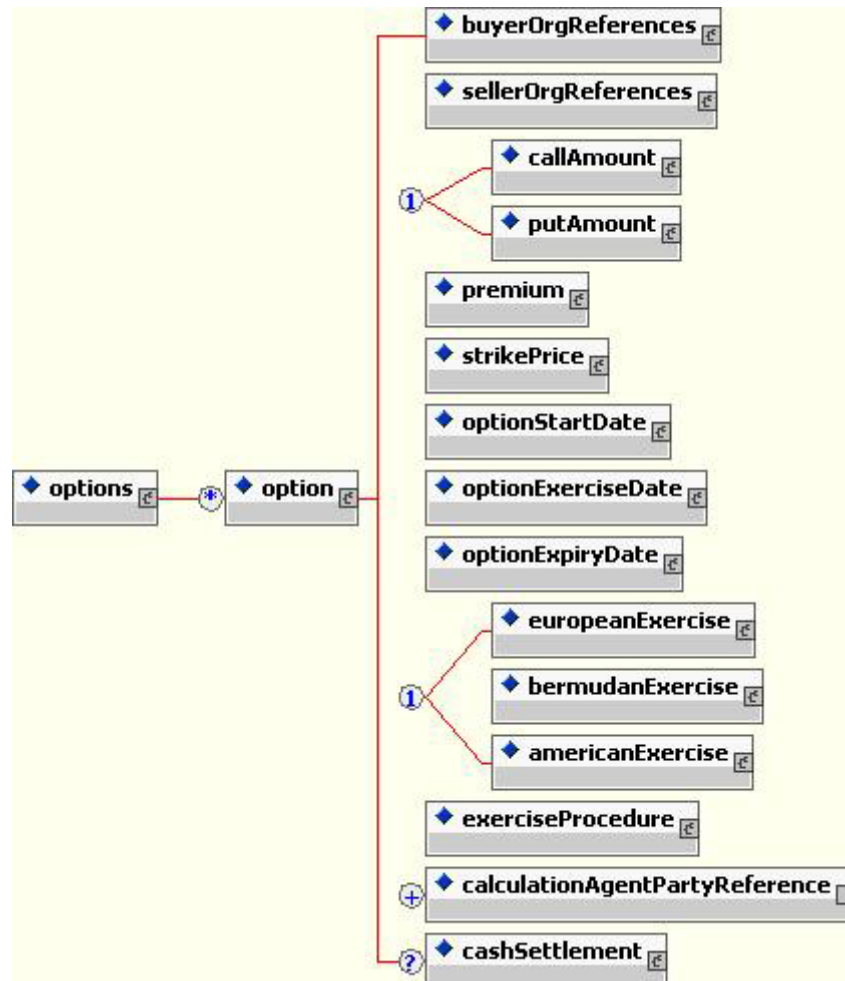


Figure 27 - Graphical representation of option information

6.1 The *buyerOrgReferences* and *sellerOrgReferences* Components

The organization references for a party to a financial transaction are listed in *buyerOrgReferences* and *sellerOrgReferences* components. These include references to the party as well as any people that might be involved in the transaction. The references are to the party or contact information defined elsewhere in the document and are named according to the roles of the various entities in the transaction.

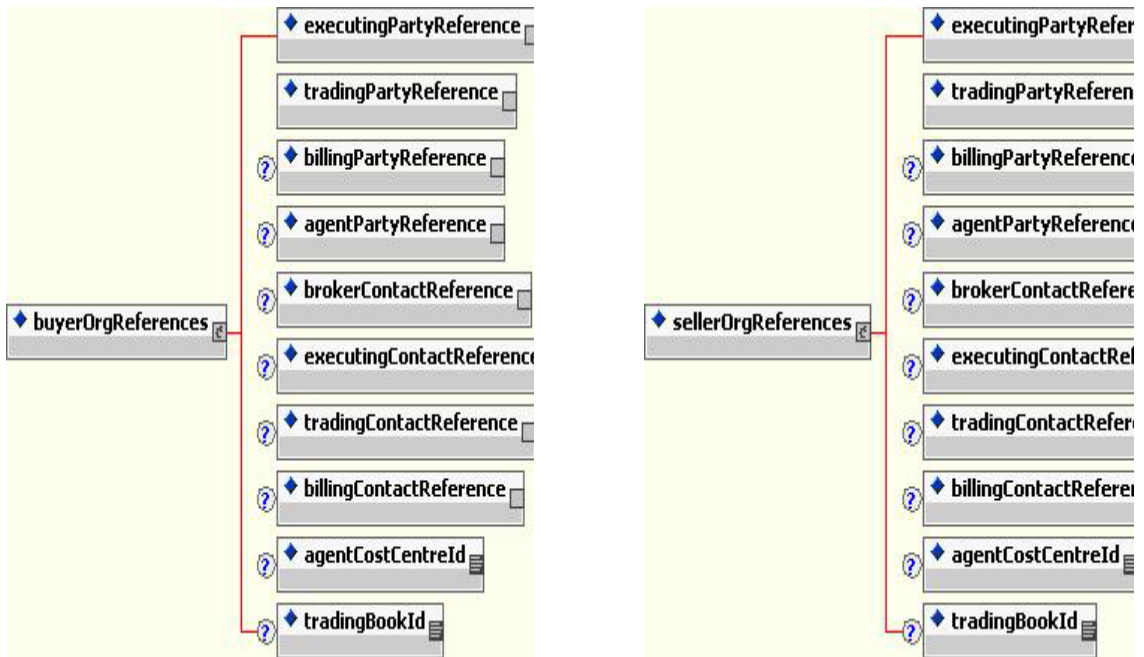


Figure 28 - Graphical representation of organisation references

6.2 The *premium* Component

The *premium* component in BML contains the information regarding the premium payment on an option.

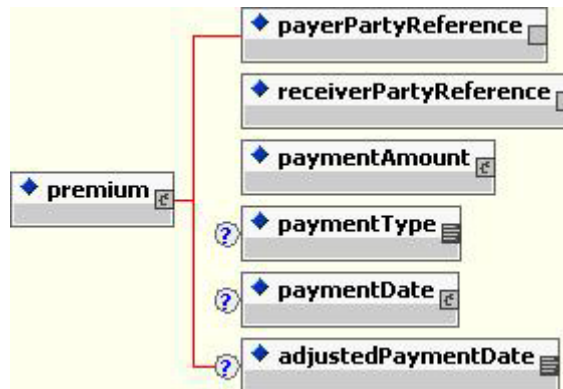


Figure 29 - Graphical representation of option premium

6.3 The *optionStartDate* Component

The *optionStartDate* component in BML contains the information regarding the starting date of an option. It is specified as either an adjustable date or a relative date.

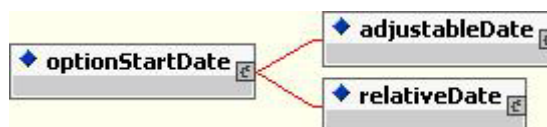


Figure 30 - Graphical representation of option start date

6.4 The *optionExerciseDate* Component

The *optionExerciseDate* component in BML contains the information regarding the date when an option can be exercised. It is specified as either an adjustable date or a relative date.

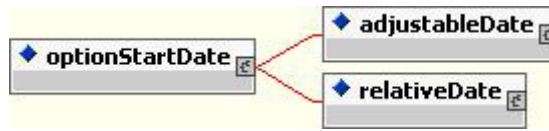


Figure 31 - Graphical representation of option exercise date

6.5 The *optionExpiryDate* Component

The *optionExpiryDate* component in BML contains the information regarding the date when an option is deemed to have expired. It is specified as either an adjustable date or a relative date.

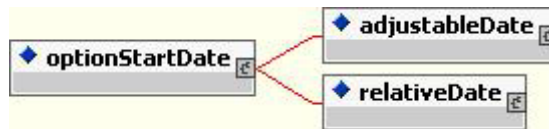


Figure 32 - Graphical representation of option expiry date

7 The *product* Component

The *product* component specifies the financial instrument being traded. This component captures the economic details of the trade. Because of the complexity of the OTC product domain such as Interest Rate Derivatives that BML covers, composing these products from various building blocks is a key aspect of the design approach.

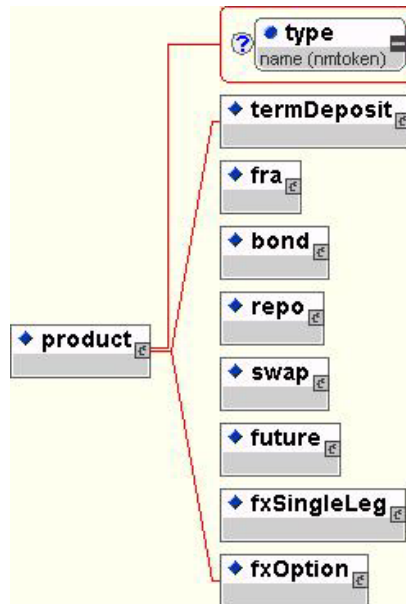


Figure 33 - Graphical representation of a product

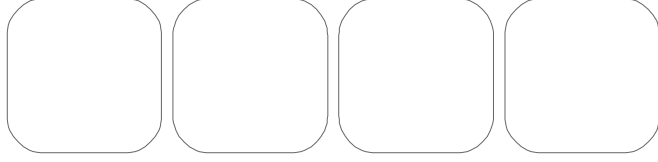
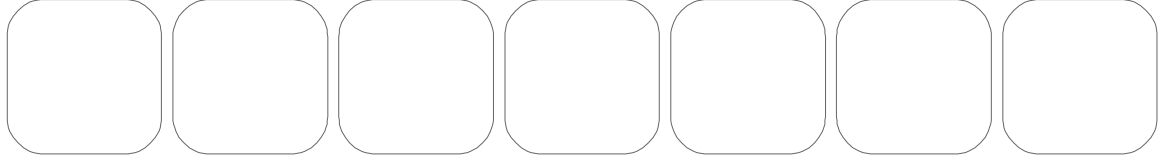
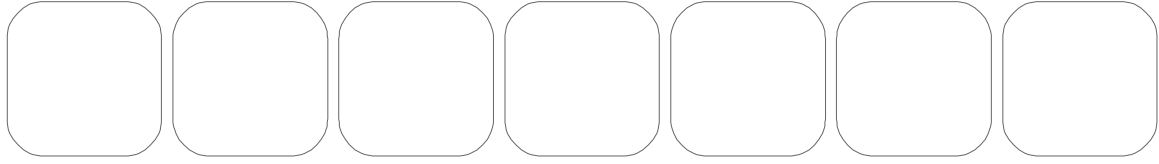
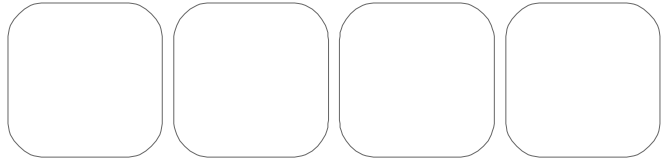
The XML definitions for *product* entities and elements are as follows

```
...
<!ENTITY % BML Product "termDeposit | fra | bond | repo | swap | future |
fxSingleLeg | fxOption">
...
<!ELEMENT product (%BML_Product;)>
<!ATTLIST product type NMTOKEN #FIXED 'Product'>
...
```

The individual product definitions are defined in individual specifications for these products.



Tullett Liberty
Cable House,
54-62 New Broad Street,
London,
EC2M 1ST
ENGLAND
www.tullib.com



Tullett Liberty