1 Roll Call

<table>
<thead>
<tr>
<th>Participant</th>
<th>Membership Status</th>
<th>Present</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gunther Stuhec</td>
<td>Member</td>
<td>P P P P P</td>
</tr>
<tr>
<td>Hisanao Sugmata</td>
<td>Member</td>
<td>P P P P P</td>
</tr>
<tr>
<td>Jostein Fromyr (Secretary)</td>
<td>Member</td>
<td>P P P P P</td>
</tr>
<tr>
<td>Kim Lambert</td>
<td>Member</td>
<td>P</td>
</tr>
<tr>
<td>Kirs Ketels</td>
<td>Member</td>
<td></td>
</tr>
<tr>
<td>Luc Mouchot</td>
<td>Member</td>
<td></td>
</tr>
<tr>
<td>Marcel Jemio</td>
<td>Member</td>
<td>P P P P</td>
</tr>
<tr>
<td>Mark Crawford (Chair)</td>
<td>Member</td>
<td>P P P P P</td>
</tr>
<tr>
<td>Melanie Kudela</td>
<td>Member</td>
<td>P</td>
</tr>
<tr>
<td>Paula Heilig</td>
<td>Member</td>
<td>P P P P P</td>
</tr>
<tr>
<td>Thomas Bikeev</td>
<td>Member</td>
<td>P P P P P</td>
</tr>
<tr>
<td>Sue Probert</td>
<td>Member</td>
<td>P P P P</td>
</tr>
<tr>
<td>John Duker</td>
<td>Member</td>
<td>P</td>
</tr>
<tr>
<td>Greg Wilson</td>
<td>Candidate Member</td>
<td>P P</td>
</tr>
<tr>
<td>James Whittle</td>
<td>Candidate Member</td>
<td></td>
</tr>
<tr>
<td>Michael Dill</td>
<td>Candidate Member</td>
<td></td>
</tr>
</tbody>
</table>

Quorum achieved with 7 members present.

2 Agenda

Priority will be given to resolving the major issues, then on discussing specific rules.

The following agenda was agreed for the week:

Monday
1. Set agenda
2. Review action items
3. Discuss model

Tuesday
1. Modularity – Define and approve model
   - What classes will be defined as types and how
   - How will BBIE elements be declared
   - How will ABIE elements be declared
   - Importing schema
     - From other Root schema
     - From other internal schema
• Schema module for RTs
• Schema module for DTs
• Schema module for BBIEs
• Schema module for ABIEs

2. Namespace
   • Approach
   • Tokens

3. Versioning

Wednesday
4. Code list
5. Identifier list
6. Content model
   • Choice
7. Extension/restriction Use methodology
   • Application of context
8. Attributes
   • Global attributes (R117)
   • How should we handle UUIDs in schema and instance
9. List containers
10. R53 block attributes
11. Annotation
   • Documentation
     • What -
     • How – XHTML or other

Thursday
12. Review other projects
   • UML to XML transformation – Thomas will show an example
   • BPSS
   • XML representation of Core Components
   • XML check list
   • CPPA Implementation Guide
13. Review of individual rules

Friday
14. Review of individual rules
15. Structure of NDR document
16. Discuss Work Plan between now and Bonn

Need to raise issue of Registry for discussion in ATG

3  **Review of action items**
List of action items reviewed and updated.
4 Modularity – Define and approve model

Discussions were based on draft NDR v 0.2.1, where a proposal for the overall schema model is defined in section 3.3, figure 3-1, and the schema modules in section 6.1, figure 6-1.

Figure 3-1 as proposed is based on similar model in UBL, but is different with respect to the treatment of element declarations and the definition of complex type for ASBIE.

The group discussed and eventually modified the figure according to the following basic principles:

- The message assembly is represented as a complex type designated as the root element of the XML message.
- An ABIE is represented as a complex type.
- An ASBIE is represented as an element within the complex type representing the ABIE in which the ASBIE is defined. The ASBIE element is in itself based on (is of type) a complex type of an ABIE.
- A BBIE is represented as an element within the complex type representing the ABIE. The BBIE is based on (is of type) a DT.
- A DT is represented as a complex type. The DT complex type is based on the complex type of its corresponding CCT complex type.
- A CCT is represented as a complex type.

TBG 17 representatives present confirmed that TBG17 would not allow “standalone” BBIE, i.e. a BBIE will always be defined as part of a ABIE. Thus there is no need to define a complex type for the BBIE.

Figure 4-2 in the CCTS was discussed and some members pointed to the fact that the Message Assembly class may in itself be regarded as a specific type of ABIE. It was also noted that the Message Assembly class and the Assembly Component class were not further discussed in the CCTS document.

The chair informed that according to current work in the OASIS TC ASBIEs are not stored in an ebXML registry as a separate object, only as association between ABIEs.

The schema modularity approach is defined in figure 6-1, 6-2 and 6-3. These figures were combined and modified during the discussions. The basic principles applied were:

- The root schema for an XML schema will import other schema modules.
- There may be one internal schema modules defining all ABIEs and BBIEs that are defined local to a given root schema.
- The root schema may also import one or more external schema modules for reusable ABIEs. Several root schemas may use this schema module.
- Separate external schema modules will be defined for
  - Identifiers; one schema module for each identifier list,
  - code lists; one schema module for each code list,
  - qualified data types; one common schema module for all qualified data types,
  - unqualified data types; one common module for all except code and identifier,
  - CCTs as defined in the CCTS; one common module for all and
  - the identifier and code unqualified data type

These schema modules must be imported into the root schema.

- The schema module for unqualified data types must import the schema modules for code list and identifier, as well as the CCT schema module.
• The schema module for **qualified data types** must import schema modules for code list and identifier, as well as the CCT schema module.
• The schema module for the **identifier and code unqualified data type** must import the schema modules for code list and identifier, as well as the CCT schema module.
• The **code list** schema module must import the identifier and code unqualified data type schema module and the CCT schema module
• The **identifier** schema module must import the identifier and code unqualified data type schema module and the CCT schema module

5 **Namespace**

Every schema module should have its own namespace, except that the internal schema module will be in the same namespace as the root schema.

For each of the schema modules we need to define its name, token, location and versioning.

<table>
<thead>
<tr>
<th>Type code</th>
<th>Name</th>
<th>Token</th>
<th>Location</th>
<th>Versioning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root schema module</td>
<td>rsm</td>
<td>Unique name for each root schema</td>
<td>Rsm</td>
<td></td>
</tr>
<tr>
<td>Internal schema module</td>
<td>Same as root schema</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABIE module</td>
<td>ram</td>
<td>Unique identifier for each ABIE module</td>
<td>ram+ID</td>
<td></td>
</tr>
<tr>
<td>Unqualified data type module</td>
<td>udt</td>
<td>No applicable as there will be only one schema module</td>
<td>Udt</td>
<td></td>
</tr>
<tr>
<td>Qualified data type module</td>
<td>qdt</td>
<td>Unique identifier for each module</td>
<td>qdt+ID</td>
<td></td>
</tr>
<tr>
<td>Identifier and code unqualified data type</td>
<td>icu</td>
<td>No applicable as there will be only one schema module</td>
<td>Icu</td>
<td></td>
</tr>
<tr>
<td>Code list modules</td>
<td>clm</td>
<td>Derived from specific values of the supplementary components</td>
<td>Derived from specific values of the supplementary components, i.e. code list identifier and code list version</td>
<td></td>
</tr>
<tr>
<td>Identifier modules</td>
<td>ids</td>
<td>Derived from specific values of the supplementary components</td>
<td>Derived from specific values of the supplementary components,</td>
<td></td>
</tr>
</tbody>
</table>
The general structure for namespace name must be:

```
```

- **schematype** = a code identifying the type of schema module
- **status** = the status of the schema as: `draft` | `standard`. Conditional should only be given if `draft`.
- **name** = a unique name for each schema module (using underscore as separator)
- **major** = The major version number. Sequentially assigned, first release starting with the number 1. Not applicable to the Identifier Schema Module and the Code List Module
- **minor** = The minor version number within a major release. Sequentially assigned, first release starting with the number 0. Not applicable to the Identifier Schema Module and the Code List Module
- **revision** = Sequentially assigned for each revision of a minor release. Only applicable where status = `draft`. Not applicable to the Identifier Schema Module and the Code List Module

`un:unece:uncefact` should be registered with the IETF in the form of an RFC

Mark to coordinate with the UN/ECE Secretariat to determine 1) if any pieces of the URN `un:unece:uncefact` had been registered, 2) who we should coordinate with to develop the RFC, and 3) writing the RFC (comparable to RFC 3121).
NRD 0.2.2 were made available to the group.

ABIE are only reusable within a defined context, thus identification of the context need to be part of the namespace name for the reusable ABIE module. Two alternatives (a) use an ID to identify the combination of context drivers, the specific values for each context driver is defined in a table (should allow wildcard), or (b) specify all context drivers as part of the namespace name. Group preferred (a). The groups discussed whether the qualifier of an ABIE was a reflection of the context drivers and if the name of the ABIE would then be unique. No consensus achieved. The group finally agreed to define one -1- schema module containing all ABIEs that may be reused by one or more roots schemas. In order to allow implementers to do customisation it was agreed to show how ABIE libraries from other organisations might be used together wit a UN/CEFACT root schema.

The namespace name for code list and identifier schemas should be derived from the values of the supplementary components (se section 4.6 in NDR 0.2.2).

5.1 Schema location

Group prefers http, i.e. ftp will not be used. The general structure for scheme location must be:
http://www.unece.org/uncefact/<schematype>/<name>_<major>._<minor>.[<revision>][<status>].xsd

For code lists two scenarios were addressed:
(a) responsible agency stores the code list under its own domain, in which case we can only recommend a schema location. The recommended structure for location is:
(b) for UN/CEFACT published code list the schema location must be:

Location for identifier schemas will be constructed similar to code lists.
(a) the recommendation to agency who stores the identifier schemas under their own domain is:
(b) for UN/CEFACT published code list the schema location must be:

6 Versioning

Discussed based on NDR v 0.2.2 (section 4.2).

The following elements will be used to identify versions on schema modules:

| Major | Sequentially assigned
|-------|-----------------------|
|       | Will be increased when incompatible changes occur, i.e.
|       | • Removing or changing values in enumerations
<p>|       | • Changing of element names, type names and element names |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Changes to the structures</td>
<td>Delete or add mandatory elements</td>
</tr>
<tr>
<td></td>
<td>Change cardinality from mandatory to optional</td>
</tr>
<tr>
<td>At draft status will start at 0</td>
<td></td>
</tr>
</tbody>
</table>

| Minor               | Sequentially assigned, starting at 0                           |
|                    | Will be increased when compatible changes occur, i.e          |
|                    | • Restrictions, but not to required elements                  |
|                    | • Adding values to enumerations                               |
|                    | • Optional extensions                                         |
|                    | • Restrictions on facets                                      |
|                    | • Delete or add optional elements                             |
|                    | • Change cardinality from optional to mandatory               |

| Revision            | Only used for schemas with draft status                       |
|                    | Sequentially assigned for each draft release, starting at 1   |

The group agreed to adopt the wording from UBL and adopt this to our unique namespace. Above examples of changes should be included in the text to explain the rules.

Versioning for code lists and identifier schemas published by external organisations is outside our control. As UN/CEFACT published code lists and identifier schemas the value of the `<Identification Schema. Version. Identifier>` will follow the same rules as for versioning of other schema modules.
NRD 0.2.3 were made available to the group.

7 Code list

Agreed to start discussion by addressing the following general issues?

1. How should we define code list modules?
2. How should we define codes in a code list?
3. How should we use each code list in “qualified data types”?
4. How should we use codes in elements?
5. How should we use codes in attributes?
6. How can we make simple and understandable?

Re 1 & 2 the group agreed that:
Each code list will be defined in its own schema module with its own specific target namespace. A comment was raised that this approach is effective if the code list rarely changes, but for a code list that changes frequently it may be a problem. It was agreed that minor changes to code lists should be allowed without creating a new schema module with a new target namespace. Minor changes was defined as:

- Adding new code values
- Changing the textual description without changing the semantics
- Marking some thing for deletion at a later stage

We will only define schema modules for UN/CEFACT code list. Wherever possible we should reuse (i.e. import) code list available from external organisations.

Within each code list module one, and only one, simple type must be defined for the content component.

The group discussed if restrictions, such as defining facets, should be defined for the simple type. The benefits would be that the pattern and length of the codes would be defined as part of the code list. This would however force schema validation against these restrictions and it was noted that this would represent an unneeded and duplicative control mechanism. It was also noted that this type of information was usually expressed as annotations to the code lists.

A group vote concluded that facets should not be declared.

A restriction has to be declared in order to define the content component (the simple type) as a restriction of the unqualified data type in order to comply with parser requirements. The restriction itself is the list of enumerations.

Code values must be defined as enumerations in the list.

Every enumeration must include annotation documentation providing the code name and may include a code description. The default language is Oxford English – other languages may also be provided, in which case the `xml:lang` attribute must be declared.

Other documentation attributes such as Code Status (Add, change, deleted, marked for deletions) may also be provided as required. CCTS will be used as token for documentation attributes.

Re 5 it was agreed that:
The allowed value domain for a qualified or unqualified data type is defined by the use of restrictions identifying the code list in which the restricted set of values exists.
The allowed value domain will be defined by the TBG17.

Re 3:
Our solution for qualified data type could allow for

- a combination of several individual code lists using `xsd:union`
- a choice between several code lists, using `xsd:choice`

Both of these can easily be accommodated in our syntax solution, although they are not likely to be frequently used. Group agreed that both should be allowed for both code list and identifiers.

The `xsd:enumeration` element may be used within reusable or external schema modules if the list of enumerated values are less or equal to 10 and no tokens are assigned. In case were tokens are assigned or the list exceeds 10 the enumeration must be expressed in a code list module.

The group then reviewed rules in section 7.5 of NDR v0.2.3:

[R134] should be reworded as: External code lists must be used wherever possible.
[R135] should be reworded as.
[R136] is not a rule, but should be expressed in text as a principle
[R137] is not a rule, but should be expressed in text as a principle
[R138] is not a rule, but should be expressed in text as a principle

Further rules will be extracted based on the above discussion.

8 Identifier list
Rules for stable identifier lists will be the same as for code lists.

For performance reason it is beneficial to concatenate the value for all supplementary components in to one schemaURI attribute, i.e supplementary components are expressed as a namespace name. The supplementary component Identification Scheme. Uniform Resource. Identifier will be used for this purpose.

9 Content model - choice
OTA are using `xsd:choice` in their current schemas. From a business point of view there is a need for the use of choice, e.g. to select one of two allowed structures.

Concerns were raised regarding the use of choice from a technical point of view. Difficulties are reported in respect of binding as well as ability to create the correct data structure.

Group agreed to allow the use of choice, but notes that it should be used with care. Choice must always be followed by elements as children.

10 Extension/restriction, use methodology
A member of the group proposed that as long as all input is from an information model, extensions and restrictions are not required. It was argued that extension and restrictions could be used for customisation purposed, in which case a feedback to the relevant TBG should be encouraged. Also when used for customisation it will be beneficial for provide rules on how extension and restrictions can be applied, but that this functionality was not required short term.

It was also pointed out that extension and restrictions could be used for internal production of schemas, but that this functionality was not required short term.

It was concluded that extension/restriction should have a low priority at this point in time.
11 Attributes
The registry will assign a UUID to each BIE registered. Question is should this be carried in an instance document. The benefit is a unique reference to where meta data may be found. It was agreed that whenever ICG assigns a UUID, that UUID must be expressed in whatever artefact it is used using the internal ID feature.

We are not using attributes for any other purpose than supplementary components, thus there is no need for attribute groups. 
[R38] should be changed to: Attribute groups must not be used.
Based on this decision [R116] and [R117] can be removed (section 7.2.1 in NDR 0.2.3).

Built in xsd attributes will however be used.

12 List containers
List containers are only an issue in respect of message assembly and appearance of the instance document. ATG2 is not able to define rules regarding list containers until we have a better understanding of the business requirements and how message assembly will be expressed.

Section 3.6 in NDR 0.2.3 should thus be removed.

13 R53 block attributes
Block attributes are use to control the use of substitution group. As we do not allow substitution groups, block attributes must not be used.

14 Annotation
UBL decided to include all columns from the TBG17 spread sheet as annotation. In addition there is at section included providing even further definition presented in xhtml.

It was agreed that ATG2 schemas must provide annotation for all of the documentation elements specified in section 7 of the CCTS. The annotation documentation must appear at the same level as the CCTS construct is being specified.

The addition of further annotation documentation does not make the schema non-conformant.
15 Review other projects

15.1 UML to XML transformation
Thomas provided as presentation for the Group on EANs approach to generation of XML schemas. EAN maintain all of their models in Rational Rose. Based on this model they generate XML schemas using the following steps:

- UML class diagram is converted into an excel spreadsheet. The spreadsheet is also used for manual validation and quality assurance.
- A set of java classes are created based on the spreadsheet. This facilitates automatic generation of consistent documentation as well as generation of XML schemas. Also allows for validation of the model through parsing of the java classes.

The same approach can be used for generation of XML schemas.

15.2 BPSS
No discussion

15.3 XML representation of Core Components
No discussion

15.4 XML check list
The group agreed to include a summary of all rules as an appendix in the NDR document. It was further agreed that no additional check list was required for the NDRs, but that a more general check list might be required in order to cover the whole production process within ATG2.

15.5 CPPA Implementation Guide
No discussion

16 Review of individual rules
Review based on NDR 0.2.4.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Status</th>
<th>Comments/discussion</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Adopted</td>
<td>Should be changed to read: In conformance with ISO/IEC/ITU/UN/CEFACT Memorandum of Understanding Management Group (MOUMG) Resolution 01/08 (MOU/MG01n83) as agreed to by UN/CEFACT, all UN/CEFACT XML must be instantiated using UTF. UTF-8 should be used as the preferred encoding, however UTF-16 may be used where necessary to support other languages.</td>
</tr>
<tr>
<td>6</td>
<td>Adopted</td>
<td>Mark has been asked about which version of OED to use. Latest release version is from 1989. Recommend using OED for writers and editors, second edition 2000. Re comment from Tony 2003.12.31: All UN/CEFACT work will be done using English. Adopted as written in NDR.</td>
</tr>
<tr>
<td>14</td>
<td>Delete</td>
<td>Revisited based on earlier discussions and comment from Tony 2003.12.31. If we create two schemas the file location must be different. The group concluded that only one fully annotated schema will be developed and maintained by UN/CEFACT.</td>
</tr>
<tr>
<td>26</td>
<td>Open</td>
<td>This chapter will be moved in to an appendix, and the rule will be replaced by a more general rule referencing the appendix.</td>
</tr>
</tbody>
</table>
Adopted Phrases for clarity as: Empty elements must not be used.

Delete 29, 37 and 38 all address the same issue. Need to rules to address use of the nillable attribute and xsi:nil. 29 should be deleted.

Adopted Phraserses as: The element declaration of xsi:nil shall not appear in any conforming instance.

Adopted Remain as is.

17 Structure of NDR document
Gunther presented a revised structure for the NDR document for discussion. The group reviewed and agreed on the following revised structure:

1. Introduction
   a. Scope and focus
   b. Audience
   c. Structure of this specification
   d. Terminology and notation
   e. Related documents
   f. Choice of XML schema language
   g. Conformance

2. General constructs
   h. Guiding principles
   i. Relationship to the CCTS
   j. UN/CEFACT Modularity model
      i. Modelling constraints
      ii. Internal schema module
      iii. External schema module
      iv. Overall schema construct (refer to appendix B)
      v. (Reusability schema – based on UBL section 3.3)
   k. Naming conventions
      i. Naming constraints
      ii. Acronyms and abbreviations
      iii. Module naming conventions
      iv. Type naming conventions
      v. Element naming conventions

3. General XML schema language conventions
   l. Schema construct
      i. Constraints on schema construction
   m. Namespace scheme
      i. Declaring namespace
      ii. Namespace name
      iii. Versioning
      iv. Schema location
   n. Attributes and element declarations
      i. Attributes
      ii. Elements
   o. Type definition
      i. Use of types
      ii. Simple type definition
      iii. Complex type definition
      iv. Constraints on type definitions
   p. Extension and restrictions
      i. Extension
      ii. Restriction
iii. Derivation
iv. Constraints on type type modifications

q. Annotation
i. Documentation
ii. Constraints on annotation

4. XML schema modules
r. Root schema
   (Sub sections used as needed)
   i. Use of … module
   ii. Schema construct
   iii. Namespace schema
   iv. Attribute and element declarations
   v. Type definitions
   vi. Extension and restriction
   vii. Annotations

s. Internal schema
t. ABIEs
u. Core component type
v. Unqualified data type
w. Identifier and code unqualified data type
x. Qualified data type
y. Code lists
z. Identifier schemes

5. XML instance documents
   aa. Root element
   bb. Validation
   cc. Character encoding
   dd. Schema instance namespace declaration
   ee. Empty content

6. Appendix A: Definition of terms
7. Appendix B: Overall structure
8. Appendix C: Bibliography

A document containing the agreed structure was uploaded to the eRoom as NDR 0.3.0.

Paula will assist Gunther to move text and rules from previous versions into the new structure.

18 Discuss Work Plan between now and Bonn

No conference call on Monday January 26. Conference calls will continue at the same time as from February 2.

ATG2 will give priority to finalising the NDR document and will not work on any other project prior to the Bonn meeting. First priority is to finalise and agree on all of the rules – the aim is to have this ready prior to the Bonn meeting.

Mark and Jostein will review official membership of the group.

Meeting adjourned.