Question/Answer Transaction Protocol
Draft for Trial Use

NISO Committee AZ
March 17, 2004

Purpose
Digital reference services constitute a rapidly growing extension of the traditional reference service offered to library patrons. While the service may be delivered via real-time chat or asynchronous e-mail, the essential characteristic of the service is the ability of the patron to submit questions and to receive answers via electronic means. Each service of interconnected users constitutes a digital reference domain. There is a growing interest in interconnecting these service domains. The protocol defined in this standard supports cross-domain communication for digital reference services.

Scope
The Question/Answer Transaction Protocol covers processing transactions for interchange of messages between digital reference domains. It defines a set of messages and associated rules of syntax and semantics for this interchange. It will support processing and routing of questions and responses and packaging of other information to be exchanged.

Metadata element sets needed to identify and describe key components of both question and answer data and institutional and personal data are defined, although some sets are maintained outside the standard.

1. Protocol Model

1.1 Introduction

1.1.1 Definitions

Accompanying Constraint -- a constraint that is expressed within a Question message from the client, or within an Answer message from the server (as opposed to being expressed within an explicit Constraint message).

Chaining. Client sends a question to the server who sends the question to a third party, who supplies the answer to the server, who supplies the answer to the client. These are are two separate transactions.

Clarification Redirect. A request for clarification asking that the user contact a specific individual to provide the clarification.

Client. the questioner.

Digital Reference Domain, DRD. A system.

Forwarding. The server sends a question from a client to a third party. Subsequently the client initiates a new transaction with that third party to carry out processing of the question.

Message Identifier. Identifier for a message which distinguishes it from any other message (from that system) of that transaction.

NetRef Package. A unit of information packaged for exchange between Digital Reference systems

Patron redirect. A user asks a client to provide access to a librarian. The client is not able to provide the user with local access to a librarian and to asks the server to intervene on the user's behalf.

Profile information. Information pertaining to actors involved in the digital reference transaction such as the person or organization asking a question

Protocol message. protocol information, content, and metadata.

*protocol operation. T*he transfer of a protocol message from client to server or from server to client. An instance of an operation type.

*Protocol. T*he specification of a set of rules for communication between systems to carry out a particular type of task. Protocol defines the messages exchanged, their semantics and format, and the rules governing their exchange.

QAT Protocol (QATP). Protocol governing messages exchanged between DRDs collaborating to process a question.

Reference Identifier. An identifier for a message when there is a response expected, so that the response can be correlated with the request.

Referral. The server responds to a question from the client recommending that it send the question to a third party, ending the transaction.

Server. The answerer.

System. An entity that exchanges protocol with another system.

Transaction Identifier. A unique identifier for a transaction, assigned by the client, and included in all messages of the transaction.

transaction. A series of related protocol operations carried out between two DRDs collaborating in the processing of a single question.

Unsolicited Clarification – A clarification provided by the client to the server even though no clarification was requested.

1.1.2 Related Standard

ISO 8601. International Date Format.

ISO 3166. Country codes.

ISO 4217. Currency codes

ISO 639-1. Codes for the representation of names of languages.
http://www.loc.gov/standards/iso639-2/

1.1.3 Basic Model

For purposes of this protocol, a *system* is an entity that exchanges protocol (defined next) with another system. A system is also referred to as a *Digital Reference Domain*, *DRD*. The terms "system" and "DRD" are used interchangeably; "system" is the preferred term, but because it is such a common term, "DRD" is used when there is a possibility of ambiguity or when the specific connotation of DRD is intended to be emphasized.

A Protocol is the specification of a set of rules for communication between systems to carry out a particular type of task. Protocol defines the messages exchanged, their semantics and format, and the rules governing their exchange. Specifically, the QAT Protocol (QATP) governs messages exchanged between DRDs collaborating to process a question.

If a DRD is itself capable of performing a function without external communication then it need not employ protocol to perform that function, and the protocol does not address internal processing or communication within that system used to carry out the function. The protocol describes communication between DRDs only.

Thus a DRD is atomic and autonomous from the point of view of another DRD. However, a DRD may itself include DRDs, not externally visible. Suppose **A** and **B** are DRDs; **A** itself may includes logically distinct components that communicate amongst one-another to process a question. For example suppose **A** includes logical components **A1** and **A2**, where **A1** receives questions and determines either (1) **A** is itself capable of answering the question, or (2) external communication with **B** is necessary. In the first case **A1** sends the question to **A2** who processes it and sends the answer to **A1** (as perhaps in use case 0.1 which is shown below). In that case, from the point of view of the world outside of **A**, none of the communication among **A**'s components is externally visible or subject to standardization. The second case requires communication between **A**

and **B** which is not possible unless governed by some agreed-upon protocol. However protocol may indeed be necessary even in the first case -- **A1** and **A2** might be distinct DRDs from each other's point of view (for example, they may be from different vendors).

When two DRDs communicate via QATP, the communication is described in the context of a *transaction*.

QATP is a client/server protocol; the *client* is the questioner and the *server* is the answerer. Any system may play either role, though not in the same transaction (a DRD's role is fixed within a given transaction); a system may be a client in one transaction and a server in another.

A *protocol operation* is the transfer of a *protocol message* from client to server or from server to client. An operation is an instance of an *operation type*, and a message is an instance of a *message type*. A *transaction* includes one or more operations.

Illustration: A client send a Question message (which includes a question) to the server. "Question" is a message type. "Question" is also an operation type. The act of sending the Question message is a Question operation. The server then sends an Answer message in response to the question. The act of sending the Answer message is an Answer operation. This concludes the exchange (the Question operation followed by the Answer operation), for this particular question; the Question operation and the Answer operation comprise a transaction.

All operations for this protocol are one-way, that is, the operation type defines a single message type. (For example, although a RequestClarification message from the server is normally followed by a Clarification message from the client, these are modelled as separate operations. There are other protocols that define operations in terms of more than a single message, for example a request followed by a response. This protocol does not do so, but may in the future.) Each operation type is given the same name as the message type it defines.

The protocol defines the following operation/message types:

Operation/Message	Invoked/sent by
Question	Client
Answer	Server
RequestClarification	Server
Clarification	Client
Constraint	either client or server
ConstraintReply	either client or server
ActionRequest	Client

Status	Server
Error	either client or server
Memo	either client or server
Other	either client or server

1.2 Exposition of Protocol Model

To begin describing the QATP Protocol model, we examine and contrast two of the use cases presented in the "Use Case" document. (And similarly, by convention, we usually use "**A**" and '**B**" to refer to the client and server respectively.)

1.2.1 The Basic Question/Answer Model

As in the *Use Case* document assume two DRDs, **A** and **B.**

System **A** has received a question from a user.

- **Use case 0.1:** ***A*** *chooses to answer the question itself.*
- **Use case 1.1:** ***A*** *chooses not to answer the question itself, but instead sends the question to* ***B****, requesting an answer.* ***B*** *processes the question, determines the answer and sends it to* ***A*** *who then supplies the answer to the user.*

Examining these two cases in some detail, in terms of how the protocol is involved:

The use case 0.1 process could be modeled as three events:

1. User sends question to **A**.
2. **A** processes the question.
3. **A** supplies answer to user.

The use case 1.1 process could be modeled as five events:

1. User sends question to **A**.
2. **A** sends question to **B**.
3. **B** processes the question.
4. **B** sends answer to **A**.
5. **A** supplies answer to user.

None of the steps in use case 0.1 has protocol significance and only steps 2 and 4 of use case 1.1 do. The protocol is not concerned with how the question got from the user to **A** or how **A** supplies the answer to the user. Nor does the protocol govern or care how a system processes the question: how **A** processes the question in use case 0.1 or **B** in use case 1.1.

Two protocol operation types so far are identified:

- **Question Operation**. The transfer of a question (Question message) from **A** to **B**.
- **Answer Operation**. The transfer of an answer (Answer message) from **B** to **A** .

A transaction may include multiple operations of either type.

Thus a transaction might consist of:

- A Question operation (Question message from client to server), followed by
- An Answer operation (Answer message from server to client).

Or:

- Question operation
- Answer operation
- Answer operation
- Answer operation
- (etc.)

Or:

- Question operation
- Answer operation
- Question operation
- Answer operation
- (etc.)

In general a transaction normally includes a Question operation which may be followed by any number of Question and/or Answer operations in any order. (The first Question operation may be preceded, for example, by Constraint and ConstraintReply operations, but not, for example, by Answer, RequestClarification, or Clarification operations. These sequencing rules are detailed in section 6.) Each Question message is referred to as a "question part", and each Answer message an "answer part". Question and answer parts may flow asynchronously during a transaction, and although there may be some logical grouping of question parts (and/or answer parts) and there may be relationships between a specific question part (or group) and a specific answer part (or group), this is not visible at the protocol level; the protocol does not attempt to group question or answer parts or correlate question parts with answer parts. (The protocol does provide for the server to indicate that a particular answer part responds to one or more specific question parts, though this is an optional feature.) From the protocol point of view, all of the question parts collectively constitute the question, and all of the answer parts, the answer. The protocol provides a means (the TransactionId) to associate all of the question and answer parts with a specific transaction, but the intellectual effort to make sense of the stream of question and answer parts is necessarily undertaken by humans.

The protocol provides for the server to indicate that a particular answer part is the last part and thus ends the answer: The server may include a lastPartFlag in the Answer

message, if it thinks that it's the last part. However this is a passive parameter that the protocol does not enforce -- the server may subsequently change it's mind and continue to send answer parts, and this is not considered to be a protocol violation. Or the server might send an answer part omitting the LastPartFlag (anticipating that it will send additional parts), and subsequently realize that it has no more parts to send, and not send any more parts. (The sever may subsequently send a Status message to indicate that the answer is complete.)

1.2.2 Clarification Model

The premise of the clarification model is that before or during processing of a question **B** might require clarification of the question before proceeding further.

Thus assume **A** has received a question from a user and has sent the question to **B**, requesting an answer. **B** determines that it needs clarification of the question, requests and receives clarification from **A**, proceeds to processes the question, and subsequently supplies the answer to **A**. This scenario could be modeled as the following events:

1. User sends question to **A**.
2. **A** sends question to **B**.
3. **B** begins processing the question.
4. **B** realizes that it needs clarification.
5. **B** requests clarification from **A**.
6. **A** requests clarification from the user.
7. The user provides clarification to **A**.
8. **A** provides the clarification to **B**.
9. **B** completes processing the question.
10. **B** sends answer to **A**.
11. **A** supplies answer to user.

Only steps 2, 5, 8 and 10 have protocol significance. (Steps 2 and 10 correspond to steps 2 and 4 of the question/answer model above.) The protocol is not concerned with how or why **B** determined that clarification was needed, or how, or even whether, **A** obtained clarification from the user. (Steps 6 and 7 need not occur; **A** might provide clarification without consulting the user, without affecting the protocol exchange.)

The protocol events would be the following:

- **A** sending the question to **B**,
- **B** requesting clarification from **A**,
- **A** supplying clarification to **B**, and
- **B** sending the answer to **A**.

Clarification adds two additional protocol operation types:

- **RequestClarification Operation**. The transfer of a request for clarification (RequestClarification message) from **B** to **A**.
- **Clarification Operation**.The provision of a clarification (Clarification message) from **A** to **B**.

The RequestClarification and Clarification messages are correlated by parameters in the message. The server assigns a ClarificationId and includes it in the RequestClarification message. The client includes that id in the Clarification message.

A typical transaction involving clarification might be:

- Question operation
- RequestClarification operation
- Clarification operation
- Answer operation

And there are many possible variations. For example, in the following, the server provides part of the answer and then requests clarification:

- Question operation
- Answer operation
- RequestClarification operation
- Clarification operation
- Answer operation

RequestClarification and Clarification need not be synchronous; the following variation is valid:

- Question operation
- Answer operation
- RequestClarification operation
- Question operation
- Answer operation
- Clarification operation
- Answer operation

The server might send several requests for clarification, asynchronously, and the clarifications supplied in response need not necessarily be in order. Consider the following sequence:

- Question operation
- RequestClarification operation -- reference id=1
- RequestClarification operation -- reference id=2
- Clarification operation --reference id=2 (responds to second request)
- Clarification operation --reference id=1 (responds to first request)
- .....

Unsolicited Clarification
The client may send an unsolicited clarification for a question. Suppose for example **A** sends a question to **B**, and **A** subsequently realizes that it needs to provide clarification for the message that it sent. (In the interim, **B** might have sent the answer or part of the answer, which may have triggered the realization.) So **A** send an unsolicited clarification (**B** has not sent a request for clarification). **A** omits the reference id, to indicate that it is an unsolicited clarification. (Note: **A** might instead send an unsolicited clarification as an additional question part, that is, in a Question message, rather than in a Clarification message. The choice has no protocol significance.)

Clarification Redirect
The server may send a request for clarification asking that the user contact an individual at **B** to provide the clarification. In this case there is no subsequent Clarification message. See related "Patron Redirect" below.

1.2.3 Constraint Model
Constraints may be imposed by the server on the client, or by the client on the server. Either system may send a Constraint message, and the peer may (or may not) respond with a ConstraintReply. Within a Constraint message there is a parameter ResponseRequired which may be set to 'true' to indicate that the peer must respond (via a ConstraintReply message); however, the protocol does not enforce this behavior, the peer can ignore the message, and the peer may respond even if ResposeRequired is 'false'. Enforcement in this sense is the responsibility of the system sending the constraint: if it says "must respond" and the peer does not respond (within the time limit set) the sending system may choose to accept the peer's lack of response or not; if not, it may terminate the transaction.

The Constraint message also includes a ConstraintId, which should be echoed in a ConstraintReply message that responds to that Constraint message. The ConstraintReply message includes an Accept flag. If the peer accepts the constraint it sets the flag to 'true'; if not it sets the flag to 'false'.

A typical transaction involving a constraint sequence may be:

- Question Operation
- Constraint Operation initiated by server, MustRespond = "yes", id = 1
- ConstraintReply Operation initiated by client, Accept = "yes", id = 1
- Answer Operation.

In a ConstraintReply message when the Accept flag is set to 'false' (indicating that the responder does not accept the proposed constraint) the responder may propose alternative constraints. In that case, the original proposer may respond with a ConstraintReply message. Thus the following sequence is possible:

- Constraint Operation initiated by server: MustRespond = "yes"; id = 1

- ConstraintReply Operation initiated by client: id = 1, Accept = "no"; alternative proposal, id=2
- ConstraintReply Operation initiated by server: id = 2, Accept = "yes"

Note: constraint ids are assigned by the party issuing the constraint, and need be unique only to the extent that a system should not issue the same id twice during a transaction. However, two constraints, one issued by the client and the other issued by the server, might have the same id, and these would be distinguished because they are implicitly qualified by role -- issuer or responder.

Accompanying Constraint
A constraint may be imposed by a Constraint message, or as an *accompanying constraint* -- a parameter within a Question message from the client, or within an Answer message from the server. The resulting response (if there is one) must be sent via a ConstraintReply message (that is, both the Question and Answer messages accomodate constraints, but neither accommodates constraint replies).

1.2.4 Action/Status Model
The client may initiate an ActionRequest operation, sending an ActionRequest message to the server. The message may request that the server suspend processing of the transaction until another request to resume, suspend until a specified time, resume processing, reset the activity timer, close the transaction, or simply send a status report. The server may initiate a Status operation in response to a specific ActionRequest message from the client, reporting on the success or failure of that operation.

The ActionRequest and Status operations are related to one another much the same way as the RequestClarification and Clarification operations (though the roles are reversed). An ActionRequest message may include an id, which a responding Status message would include.

Note further that (also similar to the RequestClarification/Clarification relationship) the server may unilaterally send a Status message at any time, for purposes of sending an unsolicited status report.

Note also, the server may include a status report in any message that it sends.

Note (see section 4, Abstract Data Structures): The StatusReport parameter is included in the parameter MsgInfo, which is included in every message; MsgInfo is a collection of many of the parameters common to most or all messages. Thus the Status message does not include an explicit StatusReport parameter, but it can carry a status report via the MsgInfo parameter.

1.3 Topological Models
The first three of these models (referral, forwarding, and chaining) come into play when

when **A** sends a question to **B** who decides that a third system, **C**, is more appropriate to answer the question.

1.3.1 Referral

In the case of referral, **B** simply responds to **A** recommending that it send the question to **C**, and that ends the transaction.

1.3.2 Forwarding

In the case of forwarding, **B** sends ("forward") the question to **C**. There are typically three transactions involved in a (successful) forwarding scenario: (1) **A** sends a question to **B** (transaction **AB**); (2) **B** forwards the question to **C** (transaction **BC**); (3) **A** initiates a transaction with **C** who processes the question (transaction **AC**).

When **B** has received a question (part) from **A** (transaction **AB**) and it is **B**'s intention to forward the question, it first waits for the entire question, that is, it waits until it receives a question part indicating "last part". **B** then sends the entire question (all received parts consolidated into a single part) to **C** in a Question message (transaction **BC**). If **B** then subsequently receives additional question parts it may discard them, or it may hold them (in case the forwarded request is rejected), however it does not forward the additional parts.

Included in the Question message from **B** to **C** is a "RequestForward" parameter. The presence of that parameter means that **B** is explicitly not requesting **C** to answer the question directly but rather to indicate if it will accept the forward, i.e. if it will accept initiation from **A** of a transaction to process the question.

C could respond to **B** (transaction **BC**) in one of the following ways:

- **C** might simply reject the forward request.
- **C** might reject the forward request and instead include the answer in the Answer message (particularly if it is an easy answer), in which case B could then simply pass it along to **A** (and **A** would never need to know about the forward attempt).
- **C** might accept the forward request.
- **C** might accept the forward request and still supply the answer, in effect saying "I accept the forward, but here's the answer anyway".

When **C** accepts the forward (last two cases) it responds to **B** (transaction **BC**) with an Answer message, which includes a token, that **B** is to send to **A**, which **A** will supply later when contacting **C**.

B then sends an Answer message to **A** (transaction **AB**), which includes a "Forwarded" parameter, and that parameter includes the token (and its presence indicates that the question was forwarded). This ends transaction **AB**. **A** then initiates a third transaction (transaction **AC**), with **C**, with a Question message, where the question is omitted, but the parameter ForwardToken is included.

When **B** forwards the question to **C** it should include any applicable constraints (imbedded in the Question message, in transaction **BC,** with ForwardedConstraintFlag set) but only for the purpose of helping **C** to make an informed decision about whether to accept the question. When **A** subsequently contacts **C, A** should not assume that **C** has accepted any constraints; **A** should include all applicable constraints embedded in the initiating question message for transaction **AC**.

1.3.3 Chaining

In the chaining model **A** sends a question to **B** who sends the question to **C,** who supplies the answer to **B**, who supplies the answer to **A**. There are two transactions, one (**AB**) between **A** and **B** and another (**BC**) between **B** and **C**. **B** is an intermediary, assuming the role of client in transaction **AB** and server in transaction **BC**.

Chaining differs from forwarding – in the forwarding scenario **B** drops out of the transaction after forwarding to **C**. More significantly, chaining may be completely transparent to **A**, that is, **A** never need know that transaction **BC** takes place (as contrasted with forwarding, where there are typically three transactions, but they are related to one-another via a token). Chaining has no protocol significance but is described here for completeness.

1.3.4 Patron Redirect

A user might interact directly with a reference librarian (via email, chat, phone, etc.). The interaction is not governed by the protocol, however the protocol might be used to facilitate connecting the two parties.

Suppose a user, interacting with system **A**, asks **A** to provide access to a librarian. If **A** is able to do that directly (that is, without involving another DRD, e.g. "system **B**"), then the protocol is not involved in any way. However suppose **A** is not able to provide the user with local access to a librarian and wants to ask **B** to intervene on the user's behalf. Then there will be communication between **A** and **B**, communication that is governed by the protocol, for the purpose of arranging to provide the user with access to a librarian (at system **B**). Subsequently there may be communication between the user and librarian, which of course is not governed by the protocol.

In this scenario, **A** initiates a transaction with **B**, sending a Question message with a RequestPatronRedirect parameter included. The user may have supplied **A** with part or all of the question in which case it may be included in the Question message, or, and in any case, the Question may be omitted. Also, there may be a sense of urgency, not present in a normal transaction, when **A** needs **B** to respond immediately because it needs to know whether or not it can set up this out-of-band discussion, so that **A** can respond in real-time to the user. The RequestPatronRedirect parameter includes a RealTimeFlag for this purpose. **B** responds with an Answer message (which may or may not include part or all of the answer) which includes a PatronRedirected parameter, that supplies contact information for a librarian. The exchange of these two messages might constitute an entire transaction.

The PatronRedirected parameter may be sent unsolicited; that is it may be included in an Answer message even though the RequestPatronRedirect parameter was not included in a Question message during the transaction. For example suppose **A** sends a Question to **B,** who, after analyzing the question decides that it will be more productive to have the (remote) user contact the (local) librarian directly. **B** would include a PatronRedirected parameter in an Answer message, supplying contact information for the librarian.

Redirection may occur during the clarification process. See "Clarification Redirect" above.

2. Identifiers

2.1 Transaction Identifiers

Transactions each have a transaction identifier, the TransactionId. The client assigns the id, includes it in the first message of the transaction, and it is echoed in all subsequent messages of the transaction.

A fully qualified TransactionId is globally unique, consisting of the client-name and a string assigned by client. It is the client's responsibility, when assigning a string for a transactionId, to ensure that it has never assigned that string before.

2.2 Message Identifiers

Each message of a transaction may have a message identifier which distinguishes it from any other message (from that system) of that transaction. The message identifier may be used by the peer for example to refer to a message in error. It may also be useful information when the message is logged.

2.3 Reference Identifiers

A message may be assigned a reference identifier (parameter ReferenceId) in the case where there is a response expected, in which case the peer would echo the assigned value (parameter EchoReferenceId).

- A server may assign a referenceId to a RequestClarification message, and the client would echo that value in the Clarification message. The client might include multiple occurrences in the case where it has bundled several clarifications (corresponding to several RequestClarification messages) into a single Clarification message.
- A client may assign a referenceId to an ActionRequest message, and the server would echo that value in the ActionReply message.

A constraint may be assigned a reference id, for the same purpose (to match a Constraint with a ConstraintReply), however, the id is assigned specifically to the constraint, not to the message, since there may be multiple constraints in a single message.

2.4 URIs

This protocol defines a number of object classes where objects need to be unambiguously identified. These objects are identified by URIs. The object classes include metadata elements, constraints, diagnostics, and schemas.

Each of these classes includes a core set of object, maintainted by the Maintenance Agency for this standard (temporarily the Library of Congress), who will delegate additional authorities on request, who may define and register additional objects.

For the core objects, as well as a list of delegated authorities, see http://www.loc.gov/standards/netref/infoURI.html

As there are various authorities assigning URIs to identify NetRef objects, and there is no specific rule prescribing what URI scheme is to be used; the authority that defines a URI decides which scheme -- some NetRef identifiers are 'http:' URIs, others are 'info:' URIs. The core set are all ‘info:’ URIs; for more information see the URL cited above.

3. Timers and Lack of Activity

A message might indicate that a response is expected; this indication may be explicit or implicit: for example a Constraint message may include an explicit flag to say that a response is expected; a RequestClarification message implicitly requests a clarification and a Question message implicitly requests an answer (one or more Answer messages). Any of these types of messages may also include an ActivityTimer parameter (included in parameter MsgInfo, which is in every message), which indicates that some message is expected within that time.

A Question message might include a RequestAcknowledgementFlag parameter, which is intended to say "please acknowledge this question, even if you don't have the answer immediately". The server may subsequently send an Answer message with no answer included but which includes the AcknowledgementFlag parameter, effectively acknowledging the question. If the Question message also includes a LackOfActivity parameter, the client is saying that it expects some message within that time, which could be an answer, an acknowledgement, a constraint, or some other message, otherwise it might cancel the transaction.

A LackOfActivity timer within a Question message is not intended to indicate when an answer is expected. That should be represented as a constraint.

A timeout, or lack of activity, does not implicitly or necessarily close a transaction. However, client or server may unilaterally, and without provision of notification, consider the transaction terminated, due to lack-of-response (or, for that matter, for any reason at all, or for no reason).

4. Abstract Data Structures

The notation in the "Occurrence" Column has the following meaning:
[1] means "must occur, not repeatable" (exactly one occurrence).
[0,1] means "optional, not repeatable" (zero or one occurrence).
[0+] means "optional, repeatable" (zero or more occurrences).
[1+] means "must occur, repeatable" (one or more occurrences).

Datatype 'null' applies to "flag" parameters (those whose name ends with "Flag") which are boolean (true/false) flags, 'true' if the parameter occurs and 'false' if not.

4.1 The NetRef Package

A unit of information packaged for exchange between Digital Reference systems is called a *NetRef Package*. It includes two parts:

- A protocol message:
  - protocol information,
  - content, and
  - metadata.
- Profile information -- information pertaining to actors involved in the digital reference transaction such as the person or organization asking a question.

NetRef Package			
Part	**Occurrence**	**Type**	**Note**
ProtocolMessage	[1]	One of the protocol messages in 4.1.1	
Profile	[0+]	ProfileType	

4.1.1 Protocol Messages

Question Message			
Parameter	**Occurrence**	**Type**	**Note**
Question	[0,1]	ContentInfo	May be omitted if ForwardToken or RequestPatronRedirect is

			supplied, or if ‘LastPartFlag’ is set (for the case where there are no more question parts but the previous part did not indicate this, so this is a way for the client to say that the question is complete).
Part number	[0,1]	positive integer	Omitted if Question is omitted; may be omitted if this is the whole question (in which case no earlier question message has been sent, and no subsequent Question message may be sent).
RequestForward	[0,1]	ContactInfoType	see 1.3.2 "Forwarding": this parameter is supplied when **B** forwards the question to **C**. It identifies who **B** is forwarding on behalf of.
ForwardToken	[0,1]	string	See 1.3.2 "Forwarding": this parameter is supplied when **A** initiates a transaction with **C**.
RequestPatronRedirect	[0,1]	RequestPatronRedirectType	See 1.3.4.
LastPartFlag	[0,1]	null	
AccompanyingConstraint	[0+]	ConstraintType	
RequestAcknowledgeFlag	[0,1]	null	Please send an acknowledgement, particularly if the answer isn't immediately available.
RelatedTransaction	[0+]	RelatedTransactionType	
MsgInfo	[1]	MsgInfoType	

Answer Message			
Parameter	**Occurrence**	**Type**	**Note**

Answer	[0,1]	ContentInfo	
PartNumber	[0,1]	positive integer	Part number of answer (does not necessarily correspond to part number of question). Omitted if Answer is omitted; otherwise may be omitted only if this is the whole answer (no earlier Answer message has been sent, and no subsequent Answer message may be sent).
FailureDescription	[0,1]	DiagnosticType	Explanation of why the answer (or part of it) cannot be supplied.
LastPartFlag	[0,1]	null	
AcceptForward	[0,1]	AcceptForwardType	See 1.3.2 "Forwarding": this parameter is supplied when the server is in the role of **C**, responding to **B**, when the Question from **B** had included the RequestForward parameter.
Forwarded	[0,1]	ForwardedType	See 1.3.2 "Forwarding": this parameter is supplied when the server is in the role of **B**, responding to **A**.
AccompanyingConstraint	[0+]	ConstraintType	
QuestionPartReference	[0+]	positive integer	When this answer part corresponds to one or more specific question parts.
AlternativeReference	[0+]	TextType	When this answer corresponds to part of the question, but not specific question parts.
PartialRefFlag	[0,1]	null	When QuestionPartReference or AlternativeReference are supplied, this flag indicates that it is a partial reference, i.e. this is one of several parts for this reference.
PatronRedirected	[0,1]	ContactInfoType	
AcknowledgementFlag	[0,1]	null	This flag acknowledges the question. It may be included in

			lieu of the answer.
RelatedTransaction	[0+]	RelatedTransactionType	
MsgInfo	[1]	MsgInfoType	

RequestClarification Message			
Parameter	**Occurrence**	**Type**	**Note**
RequestedClarification	[1]	TextType	
QuestionPartReference	[0+]	positive integer	When the request pertains to one or more specific Question parts.
ClarificationRedirect	[0,1]	ContactInfoType	The server requests that the user contact an individual at the server's system to provide the clarification out-of-band, not via the protocol. If provided, normally there will be no subsequent Clarification message.
MsgInfo	[1]	MsgInfoType	

Clarification Message			
Parameter	**Occurrence**	**Type**	**Note**
Clarification	[1]	textType	
MsgInfo	[1]	MsgInfoType	

Constraint Message			
Parameter	**Occurrence**	**Type**	**Note**
Constraint	[1]	ConstraintType	
MsgInfo	[1]	MsgInfoType	

ConstraintReply Message			
Parameter	**Occurrence**	**Type**	**Note**
ConstraintReply	[1]	ConstraintReplyType	
MsgInfo	[1]	MsgInfoType	

ActionRequest Message			
Parameter	**Occurrence**	**Type**	**Note**
RequestedAction	[1]	Controlled list: 'suspend', 'suspendUntil', 'resume', 'closeTransaction', 'sendStatusReport', 'ping', 'resetActivityTimer'	If ‘ping’, request is for server to smply send a response (Status Message with AcceptFlag set).
Until	[0,1]	ISO 8601	when RequestedAction is 'suspendUntil'
MsgInfo	[1]	MsgInfoType	

Status Message			
Parameter	**Occurrence**	**Type**	**Note**
AcceptFlag	[0,1]	null	If omitted, request was rejected.
Reason	[0+]	DiagnosticType	Reason why request was rejected, if AcceptFlag is omitted.
MsgInfo	[1]	MsgInfoType	

Memo Message			
Parameter	**Occurrence**	**Type**	**Note**
Message	[1]	TextType	
MsgInfo	[1]	MsgInfoType	

Error Message			
Parameter	**Occurrence**	**Type**	**Note**
MsgId	[0+]	string	Id of message in error
Error	[1+]	DiagnosticType	
MsgInfo	[1]	MsgInfoType	

4.2Auxiliary Data Types for Protocol Messages

ContentInfo			
Parameter	**Occurrence**	**Type**	**Note**
MetadataElement	[1+]	MetadataElementType	This datatype is the Question or Answer parameter of the Question or Answer message, and is question metadata + question, or answer metadata + answer; respectively.
Content	[0+]	ContentWrapper	

MetadataElementType			
Parameter	**Occurrence**	**Type**	**Note**
Uri	[0,1]	URI	Identifies a metadata element. Must be supplied if ‘ElementName’ omitted.
ElementName	[0,1]	String	The name of the metadata element may be supplied: (a) In lieu of the URI, in case there is no URI, in the hope that the recipient can make sense of it. (b) In addition to the URI (redundantly), in the hope that if the recipient cannot process the URI it might make sense of the

			name. (Must be supplied if ‘uri’ omitted.)
Value	[0,1]	Interpretation of the value is governed by the URI	May be omitted only for null types.

ContentWrapper			
Parameter	**Occurrence**	**Type**	**Note**
SchemaId	[0,1]	URI	These three elements – SchemaId, Format, and Value -- are analogous respectively to the three elements of MetadataElementType -- Uri, ElementName, and Value.
Format	[0,1]	String	
Value	[1]	Interpretation of the value is governed by the schema identified by SchemaId	

ContactInfoType			
Parameter	**Occurrence**	**Type**	**Note**
Name	[0,1]	String	
Address	[0+]	string	
Phone	[0+]	String	
Fax	[0+]	String	
Email	[0+]	String	

MsgInfoType			
Parameter	**Occurrence**	**Type**	**Note**
TransactionInfo	[1]	TransactionInfoType	
MsgId	[0,1]	String	A unique id for each message (sent by the system) for this transaction. It's not mandatory, but recommended. Could be used for example, for reference by peer, in an error message. Or could be useful when the message is logged.
ReferenceId	[0,1]	String	An identifier assigned when a message is

			sent for which a response is expected: a RequestClarification, or ActionRequest. The peer should echo the value (parameter EchoReferenceId) in the corresponding Clarification, or Status message.
EchoReferenceId	[0+]	String	More than one may be included, for example, in the case where the server has issued multiple clarification requests, and this responds to more than one.
ActivityTimer	[0,1]	ISO 8601	Time by which some message (not necessarily the answer) is expected.
Message	[0+]	TextType	A text message to be displayed or otherwise conveyed to the operator.
StatusReport	[0,1]	StatusReportType	This may be included only when the message is sent by the server.

TransactionInfoType			
Parameter	**Occurrence**	**Type**	**Note**
TransactionId	[1]	TransactionIdType	Identifier of current transaction
TransactionHistory	[0,1]	TransactionHistoryType	Optional history of this transaction
QuestionHistory	[0,1]	QuestionHistoryType	Optional history of question (may include info generated in the course of previous transactions)

TransactionIdType			
Parameter	**Occurrence**	**Type**	**Note**
AssignedId	[1]	String	The string assigned by the client
ClientName	[1]	String	qualifies the AssignedId to make it globally unique

TransactionHistoryType			
Parameter	**Occurrence**	**Type**	**Note**
PreviousTransactions	[0+]	TransactionIdType	Transactions from which this one was generated (i.e. by forwarding or chaining or by consolidating multiple questions).
TransactionOriginated	[1]	Controlled list: 'question', 'constraint', 'memo', 'referral'.	Way in which transaction originated.
TransactionOriginationTime	[0,1]	ISO 8601	Time at which transaction originated
TransactionFinalized	[0,1]	Controlled list: 'answered', 'rejected', 'referred', 'cancelled', ‘timeout’, ‘error’.	Way in which transaction was finalized.
TransactionFinalizationTime	[0,1]	ISO 8601	Time at which finalization occurred.
TransactionCloseTime	[0,1]	ISO 8601	Time at which transaction was closed

QuestionHistoryType			
Parameter	**Occurrence**	**Type**	**Note**
QuestionEvent	[1+]	QuestionEventType	

QuestionEventType			
Parameter	**Occurrence**	**Type**	**Note**
EventType	[1]	Controlled list: 'Question', 'Answer', 'RequestClarification'.	What kind of event this is.

		'Clarification', 'Refer', 'Chain', 'Reject'.	
EventTime	[1]	ISO 8601	When the event occurred
EventTransaction	[0,1]	TransactionIdType	Transaction with which the event was associated
EventDomain	[0,1]	String	What client or server executed this event
EventInitiator	[0,1]	String	What person or agent initiated this event
EventTarget	[0,1]	String	For events of type 'Refer', the server to which the question is referred.
EventContent	[0,1]	ContentInfo or TextInfo	For events of type 'Question', 'Answer', 'RequestClarification', 'Clarification', this field can be optionally used to include the actual content of the message.

RelatedTransactionType			
Parameter	**Occurrence**	**Type**	**Note**
TransactionId	[1]	TransactionIdType	
Relation	[1+]	TextType	

ConstraintType			
Parameter	**Occurrence**	**Type**	**Note**
Constraint	[1]	ConstraintInfo	
ForwardedConstraintFlag	[0,1]	null	For a constraint accompanying a forwarded question. See 1.3.2 "Forwarding". This flag is set when the constraint originated from **A** and is being passed from **B** to **C**.

MustIncludeFlag	[0,1]	null	This constraint must be included if the question is sent to another system
Originator	[0,1]	string	Name of the system that imposed the constraint.
MustRespondFlag	[0,1]	null	
RespondBy	[0,1]	ISO 8601	only if MustRespondFlag is set
ReferenceId	[0,1]	string	Because there can be more than one constraint per message, referenceId in MsgInfo isn't sufficient
CloseFlag	[0,1]	null	'true' when this is a constraint that the proposer already knows the responder cannot or is unwilling to comply with. Occurs only if MustRespond is 'false'.

ConstraintReplyType			
Parameter	**Occurrence**	**Type**	**Note**
AcceptFlag	[0,1]	Null	
SelectedChoice	[0,1]	ConstraintInfo	When the constraint message included a list of choices, this parameter, if present, selects one, and in addition, Accept should be set.
AlternativeProposal	[0+]	ConstraintInfo	When the responder rejects the constraint (AcceptFlag not set) and offers an alternative proposal, which was not offered in the constraint message.
EchoReferenceId	[0,1]	String	

ConstraintInfo			
Parameter	**Occurrence**	**Type**	**Note**

Uri	[0,1]	uri	These three elements – Uri, constraintName, and Value -- are analogous respectively to the three elements of MetadataElementType --Uri, ElementName, and Value.
constraintName	[0,1]		
Value	[0,1]	TextType	

DiagnosticType			
Parameter	**Occurrence**	**Type**	**Note**
Uri	[0,1]	URI	These two elements – Uri and DiagnosticName -- are analogous respectively to the first two elements of MetadataElementType –Uri and ElementName.
DiagnosticName	[0,1]		
DiagnosticNote	[0+]	TextType	Any error diagnostic may be supported with commentary.

AcceptForwardType			
Parameter	**Occurrence**	**Type**	**Note**
AcceptFlag	[0,1]	null	Either AcceptFlag is supplied along with ForwardInfo....
ForwardInfo	[1]	ForwardInfoType	
RejectFlag	[0,1]	null	.....Or RejectFlag, along with Reason.
Reason		DiagnosticType	

ForwardedType			
Parameter	**Occurrence**	**Type**	**Note**
Forwardee	[1]	string	system to which question was forwarded
ForwardInfo	[1]	ForwardInfoType	

ForwardInfoType			
Parameter	**Occurrence**	**Type**	**Note**
Token	[1]	string	token that forwardee supplied in forward response, to use when contacting forwardee.
Timer	[0,1]	ISO 8601	Estimate of how long (until when) **C** will await contact from **A** before discarding the question.

RequestPatronRedirectType			
Parameter	**Occurrence**	**Type**	**Note**
User	[1]	ContactInfoType	
SessionLog	[0,1]	textType	
RealTimeFlag	[0,1]	null	

StatusReportType			
Parameter	**Occurrence**	**Type**	**Note**
QuestionStatus	[1]	Controlled list: 'received', 'pending', 'stopped', 'assigned', 'answered', 'referred', 'cancelled', 'rejected', 'other'.	Current status of question (in regard to human processing.) If 'other', then QuestionStatusNote should give further information.
QuestionStatusTime	[0,1]	ISO 8601	Time at which the question was placed in this status.
QuestionStatusLocalAgent	[0,1]	String	Person or agent currently responsible for this question (optional.)
QuestionStatusNote	[0,1]	String	Optional text giving

			additional information about question status.
TransactionStatus	[1]	Controlled list: 'open', 'suspended', 'closed'	Current status of transaction (in regard to protocol)
SuspendedUntil	[0,1]	ISO 8601	This is only meaningful if the current TransactionStatus is 'suspended' and the suspension was requested for a definite time, in which case it indicates when the suspension will cease.
CancelPending	[0,1]	ISO 8601	This field should be reported if a 'cancel' request has been received but not yet acted upon. The value should be the time at which the request was received.
SuspendPending	[0,1]	ISO 8601	Same, if a 'suspend' request has been received but not yet acted upon.
ClarificationWait	[0,1]	ISO 8601	This field should be reported if a RequestClarification message has been sent but a reply has not yet been received. The value should be the time at which the RequestClarification message was sent.

TransactionHistoryType			
Parameter	**Occurrence**	**Type**	**Note**
Previous Transactions	[0+]	TransactionIdType	transactions from which this transaction was

			generated
TransactionOriginationTime	[0,1]	ISO 8601	The date/time the initial message of a transaction was sent.
TimeForwarded	[0,+]	ISO 8601	The date/time a question was forwarded.
TimeRejected	[0,+]	ISO 8601	The date/time a question was rejected.
TimeClosed	[0,1]	ISO 8601	The date/time a transaction was closed.
TimeCancelRequestSent	[0,+]	ISO 8601	The date/time a transaction was cancelled.
TimeCancelRequestReveived	[0,+]	ISO 8601	The date/time a cancelled transaction was reinstated.
TimeCancelAccepted	[0,+]	ISO 8601	The date/time a transaction cancellation was accepted
TimeCancelRejected	[0,+]	ISO 8601	The date/time a transaction cancellation was rejected.
TimeOpened	[0,1]	ISO 8601	The time at which a message is opened
Assignment	[0+]	AssignmentType	
Referral	[0+]	ReferralType	

AssignmentType			
Parameter	**Occurrence**	**Type**	**Note**
TimeAssignment	[0,1]	ISO 8601	The time at which a message is assigned in the “local” environment
AssignedTo	[0,1]	ContactInfo	The identification of who is handling the question

ReferralType			
Parameter			
Occurrence	**Type**	**Note**	
TimeReferred	[0,1]	ISO 8601	The time at which a query is referred (this is a “local” referral, not a protocol referral).
ReferredTo		ContactInfo	The identification of to whom the question has been referred (this is a “local” referral, not a protocol referral).
ReferralId			if applicable

TextType			
Parameter	**Occurrence**	**Type**	**Note**
Language	[0,1]	string	
Text	[1]	text	

4.3 Profile Information

Currently, most non-protocol virtual reference transactions contain some form of profile information. Although protocol transactions could proceed with no profile information

being present, many protocol transactions will contain information about the originator of the question or the agent/intermediary sending the protocol message, or will describe a link to a profile directory where such information might be obtained.

Profile information may need to be present in a transaction for a number of reasons. Some of these are:

- There may be requirements on either the sending or receiving side for authoritative identification of persons or agents involved before any work may be done.
- If fee-based transactions or transactions involving access to licensed material are undertaken, authenticating information will have to be supplied.
- An answering agent may be asked to respond directly to an end-user. In this instance, contact information for the end-user must be available.
- Agencies involved in a membership-governed network may need to identify themselves to each other to gain specific services or privileges.
- Minimally, a link to a directory/registry where profile information may be obtained may need to be included in a transaction.

ProfileType			
Parameter	**Occurrence**	**Type**	**Note**
Originator	[0,1]	PersonInstIdType	
Agent	[0+]	PersonInstIdType	
Authentication			Information, such as passwords or codes, needed to authenticate one party to another during the transaction.
AuthenticationUse-Constraint			A statement setting out the use to which the authentication data may be put. This may address security levels, confidentiality requirements, privacy ratings, etc
Referral		ReferralType	The name of a person or agent to which the transaction has been referred or forwarded.Accompanied by an identifying code or symbol for the person or agent to which the transaction has been referred or forwarded. This element occurs as an historical element in the initial transaction, not in the following transaction stemming from the action of referral or forwarding.
ReasonforResearch			Contextual information about the question that may assist in provision of an answer. E.g., “The

			user is investigating the effect of cold weather on tigers in northern zoos.”

PersonInstIdType			
Parameter	**Occurrence**	**Type**	**Note**
Person	[0,1]	string	
Institution	[0,1]	string	
Id	[0,1]	string	

AgentInfo			
Parameter	**Occurrence**	**Type**	**Note**
PersonInstId	[0,1]	PersonInstIdType	
Role	[0,1]	string	
Constraint	[0,1]	string	Information constraining the participation of an agent, such as when it is desired that a particular person in an answering institution handle the question. E.g., “Only Susie Smith may answer this question.”

5. XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSPY v2004 rel. 3 U (http://www.xmlspy.com) by Cary Gordon (NISO) -->
<!-- last update TUE16MAR2004 -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="NetRef">
        <xs:annotation>
            <xs:documentation>Root Element (for use in XML). It consists of all metadata transferred in an
interchange. The metadata consists of two parts: Protocol package (information needed by the protocol to
function); and, Profile Package (information relating to the agents involved in the transaction).</xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="Protocol"/>
                <xs:element ref="Profile" minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
```

```
    <xs:element name="Protocol">
        <xs:complexType>
            <xs:choice>
                <xs:element name="QuestionMessage">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Question" type="ContentInfoType" minOccurs="0">
                                <xs:annotation>
                                    <xs:documentation>May be omitted if ForwardToken or
RequestPatronRedirect is supplied.</xs:documentation>
                                </xs:annotation>
                            </xs:element>
                            <xs:element name="PartNumber" type="xs:positiveInteger" minOccurs="0"/>
                            <xs:element name="RequestForward" type="ContactInfoType" minOccurs="0"/>
                            <xs:element name="ForwardToken" type="xs:string" minOccurs="0"/>
                            <xs:element name="RequestPatronRedirect" type="RequestPatronRedirectType"
minOccurs="0"/>
                            <xs:element name="LastPartFlag" type="xs:boolean" minOccurs="0"/>
                            <xs:element name="AccompanyingConstraint" type="ConstraintType" minOccurs="0"
maxOccurs="unbounded"/>
                            <xs:element name="RequestAcknowledgeFlag" type="xs:boolean" minOccurs="0"/>
                            <xs:element name="RelatedTransaction" type="RelatedTransactionType"
minOccurs="0" maxOccurs="unbounded"/>
                            <xs:element name="MsgInfo" type="MsgInfoType"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="AnswerMessage">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="Answer" type="ContentInfoType" minOccurs="0"/>
                            <xs:element name="PartNumber" type="xs:positiveInteger" minOccurs="0"/>
                            <xs:element name="FailureDescription" type="DiagnosticType" minOccurs="0"/>
                            <xs:element name="LastPartFlag" type="xs:boolean" minOccurs="0"/>
                            <xs:element name="AcceptForward" type="AcceptForwardType" minOccurs="0"/>
                            <xs:element name="Forwarded" type="ForwardedType" minOccurs="0"/>
                            <xs:element name="AccompanyingConstraint" type="ConstraintType" minOccurs="0"
maxOccurs="unbounded"/>
                            <xs:element name="QuestionPartReference" type="xs:positiveInteger" minOccurs="0"
maxOccurs="unbounded"/>
                            <xs:element name="AlternativeReference" type="TextType" minOccurs="0"
maxOccurs="unbounded"/>
                            <xs:element name="PartialRefFlag" type="xs:boolean" minOccurs="0"/>
                            <xs:element name="PatronRedirected" type="ContactInfoType" minOccurs="0"/>
                            <xs:element name="AcknowledgementFlag" type="xs:boolean" minOccurs="0"/>
                            <xs:element name="RelatedTransaction" type="RelatedTransactionType"
minOccurs="0" maxOccurs="unbounded"/>
                            <xs:element name="MagInfo" type="MsgInfoType"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="RequestClarificationMessage">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="RequestedClarification" type="TextType"/>
                            <xs:element name="QuestionPartReference" type="xs:positiveInteger" minOccurs="0"
maxOccurs="unbounded"/>
                            <xs:element name="ClarificationRedirect" type="ContactInfoType" minOccurs="0"/>
                            <xs:element name="MsgInfo" type="MsgInfoType"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="ClarificationMessage">
                    <xs:complexType>
                        <xs:all>
                            <xs:element name="Clarification" type="TextType"/>
```

```
                    <xs:element name="MsgInfo" type="MsgInfoType"/>
                </xs:all>
            </xs:complexType>
        </xs:element>
        <xs:element name="ConstraintMessage">
            <xs:complexType>
                <xs:all>
                    <xs:element name="Constraint" type="ConstraintType"/>
                    <xs:element name="MsgInfo" type="MsgInfoType"/>
                </xs:all>
            </xs:complexType>
        </xs:element>
        <xs:element name="ConstraintReplyMessage">
            <xs:complexType>
                <xs:all>
                    <xs:element name="ConstraintReply" type="ConstraintReplyType"/>
                    <xs:element name="MsgInfo" type="MsgInfoType"/>
                </xs:all>
            </xs:complexType>
        </xs:element>
        <xs:element name="ActionRequestMessage">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="RequestedAction">
                        <xs:simpleType>
                            <xs:restriction base="xs:string">
                                <xs:enumeration value="suspend"/>
                                <xs:enumeration value="suspendUntil"/>
                                <xs:enumeration value="resume"/>
                                <xs:enumeration value="closeTransaction"/>
                                <xs:enumeration value="sendStatusReport"/>
                                <xs:enumeration value="ping"/>
                                <xs:enumeration value="resetActivityTimer"/>
                            </xs:restriction>
                        </xs:simpleType>
                    </xs:element>
                    <xs:element name="Until" type="xs:dateTime" minOccurs="0"/>
                    <xs:element name="MsgInfo" type="MsgInfoType"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="StatusMessage">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="AcceptFlag" type="xs:boolean" minOccurs="0"/>
                    <xs:element name="Diagnostic" type="DiagnosticType" minOccurs="0"
maxOccurs="unbounded"/>
                    <xs:element name="MsgInfo" type="MsgInfoType"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="MemoMessage">
            <xs:complexType>
                <xs:all>
                    <xs:element name="Message" type="TextType"/>
                    <xs:element name="MsgInfo" type="MsgInfoType"/>
                </xs:all>
            </xs:complexType>
        </xs:element>
        <xs:element name="ErrorMessage">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="MsgID" type="xs:string" minOccurs="0"
maxOccurs="unbounded">
                        <xs:annotation>
                            <xs:documentation>Really unbounded?</xs:documentation>
```

```
                    </xs:annotation>
                </xs:element>
                <xs:element name="Error" type="DiagnosticType" maxOccurs="unbounded"/>
                <xs:element name="MsgInfo" type="MsgInfoType"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:choice>
</xs:complexType>
</xs:element>
<xs:element name="Profile" type="ProfileType"/>
<xs:complexType name="AcceptForwardType">
    <xs:sequence>
        <xs:element name="Accept" type="xs:boolean" minOccurs="0"/>
        <xs:element name="ForwardInfo" type="ForwardInfoType"/>
        <xs:element name="Reject" minOccurs="0">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="RejectFlag" type="xs:boolean"/>
                    <xs:element name="Reason" type="DiagnosticType" minOccurs="0"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="AgentInfoType">
    <xs:sequence>
        <xs:element name="PersonInstId" type="PersonInstIdType"/>
        <xs:element name="Role" type="xs:string" minOccurs="0"/>
        <xs:element name="Constraint" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="Assignme ntType">
    <xs:sequence>
        <xs:element name="TimeAssignment" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="AssignedTo " type="ContactInfoType" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="ConstraintInfoType">
    <xs:choice>
        <xs:sequence>
            <xs:element name="URI" type="xs:anyURI"/>
            <xs:element name="ConstraintName" type="xs:string" minOccurs="0"/>
            <xs:element name="Value" type="xs:anyType" minOccurs="0"/>
        </xs:sequence>
        <xs:sequence>
            <xs:element name="ConstraintName" type="xs:string"/>
            <xs:element name="Value" type="xs:anyType" minOccurs="0"/>
        </xs:sequence>
    </xs:choice>
</xs:complexType>
<xs:complexType name="ConstraintReplyType">
    <xs:sequence>
        <xs:element name="Accept" type="xs:boolean" minOccurs="0"/>
        <xs:element name="SelectedChoice" type="ConstraintInfoType" minOccurs="0"/>
        <xs:element name="AlternativeProposal" type="ConstraintInfoType" minOccurs="0"
maxOccurs="unbounded"/>
        <xs:element name="EchoReferenceID" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="ConstraintType">
    <xs:sequence>
        <xs:element name="Constraint" type="ConstraintInfoType"/>
        <xs:element name="ForwardedConstraintFlag" type="xs:boolean" minOccurs="0"/>
        <xs:element name="MustIncludeFlag" type="xs:boolean" minOccurs="0"/>
        <xs:element name="Originator" type="xs:string" minOccurs="0"/>
```

```
            <xs:element name="MustRespondFlag" type="xs:boolean" minOccurs="0"/>
            <xs:element name="RespondBy" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="ReferenceId" type="xs:string" minOccurs="0"/>
            <xs:element name="CloseFlag" type="xs:boolean" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ContactInfoType">
        <xs:sequence>
            <xs:element name="Name" type="xs:string" minOccurs="0"/>
            <xs:element name="Address" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="Phone" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="Fax" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="Email" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ContentInfoType">
        <xs:sequence>
            <xs:element name="Metadata" type="MetadataElementType" maxOccurs="unbounded"/>
            <xs:element name="Content" type="ContentWrapperType" minOccurs="0" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ContentWrapperType">
        <xs:choice>
            <xs:sequence>
                <xs:element name="SchemaID" type="xs:anyURI"/>
                <xs:element name="Format" type="xs:string" minOccurs="0"/>
                <xs:element name="Value" type="xs:anyType"/>
            </xs:sequence>
            <xs:sequence>
                <xs:element name="Format" type="xs:string"/>
                <xs:element name="Value" type="xs:anyType"/>
            </xs:sequence>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="DiagnosticType">
        <xs:choice>
            <xs:sequence>
                <xs:element name="URI" type="xs:anyURI"/>
                <xs:element name="DiagnosticName" type="xs:string" minOccurs="0"/>
                <xs:element name="DiagnosticNote" type="xs:string" minOccurs="0"/>
            </xs:sequence>
            <xs:sequence>
                <xs:element name="DiagnosticName" type="xs:string"/>
                <xs:element name="DiagnosticNote" type="xs:string" minOccurs="0"/>
            </xs:sequence>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="ForwardInfoType">
        <xs:sequence>
            <xs:element name="Token" type="xs:string"/>
            <xs:element name="Timer" type="xs:dateTime" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="ForwardedType">
        <xs:all>
            <xs:element name="Forwardee" type="xs:string"/>
            <xs:element name="ForwardInfo" type="ForwardInfoType"/>
        </xs:all>
    </xs:complexType>
    <xs:complexType name="MetadataElementType">
        <xs:choice>
            <xs:sequence>
                <xs:element name="URI" type="xs:anyURI"/>
                <xs:element name="ElementName" type="xs:string" minOccurs="0"/>
                <xs:element name="Value" type="xs:anyType" minOccurs="0"/>
            </xs:sequence>
```

```
            <xs:sequence>
                <xs:element name="ElementName" type="xs:string"/>
                <xs:element name="Value" type="xs:anyType" minOccurs="0"/>
            </xs:sequence>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="MsgInfoType">
        <xs:sequence>
            <xs:element name="TransactionInfo" type="TransactionInfoType"/>
            <xs:element name="MsgId" type="xs:string" minOccurs="0"/>
            <xs:element name="ReferenceId" type="xs:string" minOccurs="0"/>
            <xs:element name="EchoReferenceId" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="ActivityTimer" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="Message" type="TextType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="StatusReport" type="StatusReportType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="PersonInstIdType">
        <xs:choice>
            <xs:element name="Person" type="xs:string"/>
            <xs:element name="Institution" type="xs:string"/>
            <xs:element name="ID" type="xs:ID"/>
        </xs:choice>
    </xs:complexType>
    <xs:complexType name="ProfileType">
        <xs:sequence>
            <xs:element name="Originator" type="PersonInstIdType" minOccurs="0"/>
            <xs:element name="Agent" type="AgentInfoType" minOccurs="0" maxOccurs="unbounded"/>
            <xs:element name="Authentication" minOccurs="0"/>
            <xs:element name="AuthenticationUserConstraint" minOccurs="0"/>
            <xs:element name="Referral" type="ReferralType" minOccurs="0"/>
            <xs:element name="ReasonForResearch" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="QuestionEventType">
        <xs:sequence>
            <xs:element name="EventType">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="question"/>
                        <xs:enumeration value="requestClarification"/>
                        <xs:enumeration value="clarification"/>
                        <xs:enumeration value="refer"/>
                        <xs:enumeration value="chain"/>
                        <xs:enumeration value="reject"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="EventTime" type="xs:dateTime"/>
            <xs:element name="EventTransaction" type="TransactionIdType" minOccurs="0"/>
            <xs:element name="EventDomain" type="xs:string" minOccurs="0"/>
            <xs:element name="EventInitiator" type="xs:string" minOccurs="0"/>
            <xs:element name="EventTarget" type="xs:string" minOccurs="0"/>
            <xs:element name="EventContent" minOccurs="0">
                <xs:complexType>
                    <xs:choice>
                        <xs:element name="ContentInfo" type="ContentInfoType"/>
                        <xs:element name="Text" type="TextType"/>
                    </xs:choice>
                </xs:complexType>
            </xs:element>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="QuestionHistoryType">
        <xs:sequence>
            <xs:element name="QuestionEvent" type="QuestionEventType" maxOccurs="unbounded"/>
```

```
    </xs:sequence>
</xs:complexType>
<xs:complexType name="ReferralType">
    <xs:sequence>
        <xs:element name="TimeReferred" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="ReferredTo" type="ContactInfoType" minOccurs="0"/>
        <xs:element name="ReferralID" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="RelatedTransactionType">
    <xs:sequence>
        <xs:element name="TransactionId" type="TransactionIdType"/>
        <xs:element name="Relation" type="TextType" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="RequestPatronRedirectType">
    <xs:sequence>
        <xs:element name="User" type="ContactInfoType"/>
        <xs:element name="SessionLog" type="TextType" minOccurs="0"/>
        <xs:element name="RealTimeFlag" type="xs:boolean" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="StatusReportType">
    <xs:sequence>
        <xs:element name="QuestionStatus">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="received"/>
                    <xs:enumeration value="pending"/>
                    <xs:enumeration value="stopped"/>
                    <xs:enumeration value="assigned"/>
                    <xs:enumeration value="answered"/>
                    <xs:enumeration value="referred"/>
                    <xs:enumeration value="cancelled"/>
                    <xs:enumeration value="rejected"/>
                    <xs:enumeration value="other"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="QuestionStatusTime" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="QuestionStatusLocalAgent" type="xs:string" minOccurs="0"/>
        <xs:element name="QuestionStatusNote" type="xs:string" minOccurs="0"/>
        <xs:element name="TransactionStatus">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="open"/>
                    <xs:enumeration value="suspended"/>
                    <xs:enumeration value="closed"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:element>
        <xs:element name="SuspendedUntil" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="CancelPending" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="SuspendPending" type="xs:dateTime" minOccurs="0"/>
        <xs:element name="ClarificationWait" type="xs:dateTime" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="TextType">
    <xs:sequence>
        <xs:element name="Language" type="xs:string" minOccurs="0"/>
        <xs:element name="Text" type="xs:string"/>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="TransactionHistoryType">
    <xs:sequence>
```

```
            <xs:element name="PreviousTransactions" type="TransactionIdType" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="TransactionOriginated">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="question"/>
                        <xs:enumeration value="constraint"/>
                        <xs:enumeration value="memo"/>
                        <xs:enumeration value="referral"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="TransactionOriginationTime" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="TimeFinalized" minOccurs="0">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:enumeration value="answered"/>
                        <xs:enumeration value="rejected"/>
                        <xs:enumeration value="referred"/>
                        <xs:enumeration value="cancelled"/>
                        <xs:enumeration value="timeout"/>
                        <xs:enumeration value="error"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:element>
            <xs:element name="TimeFinalizationTime" type="xs:dateTime" minOccurs="0"/>
            <xs:element name="TimeCloseTime" type="xs:dateTime" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:complexType name="TransactionIdType">
        <xs:all>
            <xs:element name="AssignedID" type="xs:ID"/>
            <xs:element name="ClientName" type="xs:string"/>
        </xs:all>
    </xs:complexType>
    <xs:complexType name="TransactionInfoType">
        <xs:sequence>
            <xs:element name="TransactionId" type="TransactionIdType"/>
            <xs:element name="TransactionHistory" type="TransactionHistoryType" minOccurs="0"/>
            <xs:element name="QuestionHistory" type="QuestionHistoryType" minOccurs="0"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="ISO-639">
        <xs:annotation>
            <xs:documentation>Language (http://www.loc.gov/standards/iso639-
2/langcodes.html)</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
            <xs:enumeration value="aar"/>
            <xs:enumeration value="abk"/>
            <xs:enumeration value="ace"/>
            <xs:enumeration value="ach"/>
            <xs:enumeration value="ada"/>
            <xs:enumeration value="ady"/>
            <xs:enumeration value="afa"/>
            <xs:enumeration value="afh"/>
            <xs:enumeration value="afr"/>
            <xs:enumeration value="aka"/>
            <xs:enumeration value="akk"/>
            <xs:enumeration value="alb"/>
            <xs:enumeration value="ale"/>
            <xs:enumeration value="alg"/>
            <xs:enumeration value="amh"/>
            <xs:enumeration value="ang"/>
            <xs:enumeration value="apa"/>
            <xs:enumeration value="ara"/>
```

```
<xs:enumeration value="arc"/>
<xs:enumeration value="arg"/>
<xs:enumeration value="arm"/>
<xs:enumeration value="arn"/>
<xs:enumeration value="arp"/>
<xs:enumeration value="art"/>
<xs:enumeration value="arw"/>
<xs:enumeration value="asm"/>
<xs:enumeration value="ast"/>
<xs:enumeration value="ath"/>
<xs:enumeration value="aus"/>
<xs:enumeration value="ava"/>
<xs:enumeration value="ave"/>
<xs:enumeration value="awa"/>
<xs:enumeration value="aym"/>
<xs:enumeration value="aze"/>
<xs:enumeration value="bad"/>
<xs:enumeration value="bai"/>
<xs:enumeration value="bak"/>
<xs:enumeration value="bal"/>
<xs:enumeration value="bam"/>
<xs:enumeration value="ban"/>
<xs:enumeration value="baq"/>
<xs:enumeration value="baq"/>
<xs:enumeration value="bas"/>
<xs:enumeration value="bat"/>
<xs:enumeration value="bej"/>
<xs:enumeration value="bel"/>
<xs:enumeration value="bem"/>
<xs:enumeration value="ben"/>
<xs:enumeration value="ber"/>
<xs:enumeration value="bho"/>
<xs:enumeration value="bih"/>
<xs:enumeration value="bik"/>
<xs:enumeration value="bin"/>
<xs:enumeration value="bis"/>
<xs:enumeration value="bla"/>
<xs:enumeration value="bnt"/>
<xs:enumeration value="bos"/>
<xs:enumeration value="bra"/>
<xs:enumeration value="bre"/>
<xs:enumeration value="btk"/>
<xs:enumeration value="bua"/>
<xs:enumeration value="bug"/>
<xs:enumeration value="bul"/>
<xs:enumeration value="bur"/>
<xs:enumeration value="bur"/>
<xs:enumeration value="byn"/>
<xs:enumeration value="cad"/>
<xs:enumeration value="cai"/>
<xs:enumeration value="car"/>
<xs:enumeration value="cat"/>
<xs:enumeration value="cau"/>
<xs:enumeration value="ceb"/>
<xs:enumeration value="cel"/>
<xs:enumeration value="cha"/>
<xs:enumeration value="chb"/>
<xs:enumeration value="che"/>
<xs:enumeration value="chg"/>
<xs:enumeration value="chi"/>
<xs:enumeration value="chk"/>
<xs:enumeration value="chm"/>
<xs:enumeration value="chn"/>
<xs:enumeration value="cho"/>
<xs:enumeration value="chp"/>
<xs:enumeration value="chr"/>
```

```
<xs:enumeration value="chu"/>
<xs:enumeration value="chv"/>
<xs:enumeration value="chy"/>
<xs:enumeration value="cmc"/>
<xs:enumeration value="cop"/>
<xs:enumeration value="cor"/>
<xs:enumeration value="cos"/>
<xs:enumeration value="cpe"/>
<xs:enumeration value="cpf"/>
<xs:enumeration value="cpp"/>
<xs:enumeration value="cre"/>
<xs:enumeration value="crh"/>
<xs:enumeration value="crp"/>
<xs:enumeration value="csb"/>
<xs:enumeration value="cus"/>
<xs:enumeration value="cze"/>
<xs:enumeration value="dak"/>
<xs:enumeration value="dan"/>
<xs:enumeration value="dar"/>
<xs:enumeration value="day"/>
<xs:enumeration value="del"/>
<xs:enumeration value="den"/>
<xs:enumeration value="dgr"/>
<xs:enumeration value="din"/>
<xs:enumeration value="div"/>
<xs:enumeration value="doi"/>
<xs:enumeration value="dra"/>
<xs:enumeration value="dsb"/>
<xs:enumeration value="dua"/>
<xs:enumeration value="dum"/>
<xs:enumeration value="dut"/>
<xs:enumeration value="dyu"/>
<xs:enumeration value="dzo"/>
<xs:enumeration value="efi"/>
<xs:enumeration value="egy"/>
<xs:enumeration value="eka"/>
<xs:enumeration value="elx"/>
<xs:enumeration value="eng"/>
<xs:enumeration value="enm"/>
<xs:enumeration value="epo"/>
<xs:enumeration value="est"/>
<xs:enumeration value="ewe"/>
<xs:enumeration value="ewo"/>
<xs:enumeration value="fan"/>
<xs:enumeration value="fao"/>
<xs:enumeration value="fat"/>
<xs:enumeration value="fij"/>
<xs:enumeration value="fin"/>
<xs:enumeration value="fiu"/>
<xs:enumeration value="fon"/>
<xs:enumeration value="fre"/>
<xs:enumeration value="frm"/>
<xs:enumeration value="fro"/>
<xs:enumeration value="fry"/>
<xs:enumeration value="ful"/>
<xs:enumeration value="fur"/>
<xs:enumeration value="gaa"/>
<xs:enumeration value="gay"/>
<xs:enumeration value="gba"/>
<xs:enumeration value="gem"/>
<xs:enumeration value="geo"/>
<xs:enumeration value="ger"/>
<xs:enumeration value="gez"/>
<xs:enumeration value="gil"/>
<xs:enumeration value="gla"/>
<xs:enumeration value="gle"/>
```

```
<xs:enumeration value="glg"/>
<xs:enumeration value="glv"/>
<xs:enumeration value="gmh"/>
<xs:enumeration value="goh"/>
<xs:enumeration value="gon"/>
<xs:enumeration value="gor"/>
<xs:enumeration value="got"/>
<xs:enumeration value="grb"/>
<xs:enumeration value="grc"/>
<xs:enumeration value="gre"/>
<xs:enumeration value="grn"/>
<xs:enumeration value="guj"/>
<xs:enumeration value="gwi"/>
<xs:enumeration value="hai"/>
<xs:enumeration value="hat"/>
<xs:enumeration value="hau"/>
<xs:enumeration value="haw"/>
<xs:enumeration value="heb"/>
<xs:enumeration value="her"/>
<xs:enumeration value="hil"/>
<xs:enumeration value="him"/>
<xs:enumeration value="hin"/>
<xs:enumeration value="hit"/>
<xs:enumeration value="hmn"/>
<xs:enumeration value="hmo"/>
<xs:enumeration value="hsb"/>
<xs:enumeration value="hun"/>
<xs:enumeration value="hup"/>
<xs:enumeration value="iba"/>
<xs:enumeration value="ibo"/>
<xs:enumeration value="ice"/>
<xs:enumeration value="ice"/>
<xs:enumeration value="ido"/>
<xs:enumeration value="iii"/>
<xs:enumeration value="ijo"/>
<xs:enumeration value="iku"/>
<xs:enumeration value="ile"/>
<xs:enumeration value="ilo"/>
<xs:enumeration value="ina"/>
<xs:enumeration value="inc"/>
<xs:enumeration value="ind"/>
<xs:enumeration value="ine"/>
<xs:enumeration value="inh"/>
<xs:enumeration value="ipk"/>
<xs:enumeration value="ira"/>
<xs:enumeration value="iro"/>
<xs:enumeration value="ita"/>
<xs:enumeration value="jav"/>
<xs:enumeration value="jbo"/>
<xs:enumeration value="jpn"/>
<xs:enumeration value="jpr"/>
<xs:enumeration value="jrb"/>
<xs:enumeration value="kaa"/>
<xs:enumeration value="kab"/>
<xs:enumeration value="kac"/>
<xs:enumeration value="kal"/>
<xs:enumeration value="kam"/>
<xs:enumeration value="kan"/>
<xs:enumeration value="kar"/>
<xs:enumeration value="kas"/>
<xs:enumeration value="kau"/>
<xs:enumeration value="kaw"/>
<xs:enumeration value="kaz"/>
<xs:enumeration value="kbd"/>
<xs:enumeration value="kha"/>
<xs:enumeration value="khi"/>
```

```
<xs:enumeration value="khm"/>
<xs:enumeration value="kho"/>
<xs:enumeration value="kik"/>
<xs:enumeration value="kin"/>
<xs:enumeration value="kir"/>
<xs:enumeration value="kmb"/>
<xs:enumeration value="kok"/>
<xs:enumeration value="kom"/>
<xs:enumeration value="kon"/>
<xs:enumeration value="kor"/>
<xs:enumeration value="kos"/>
<xs:enumeration value="kpe"/>
<xs:enumeration value="krc"/>
<xs:enumeration value="kro"/>
<xs:enumeration value="kru"/>
<xs:enumeration value="kua"/>
<xs:enumeration value="kum"/>
<xs:enumeration value="kur"/>
<xs:enumeration value="kut"/>
<xs:enumeration value="lad"/>
<xs:enumeration value="lah"/>
<xs:enumeration value="lam"/>
<xs:enumeration value="lao"/>
<xs:enumeration value="lat"/>
<xs:enumeration value="lav"/>
<xs:enumeration value="lez"/>
<xs:enumeration value="lim"/>
<xs:enumeration value="lin"/>
<xs:enumeration value="lit"/>
<xs:enumeration value="lol"/>
<xs:enumeration value="loz"/>
<xs:enumeration value="ltz"/>
<xs:enumeration value="lua"/>
<xs:enumeration value="lub"/>
<xs:enumeration value="lug"/>
<xs:enumeration value="lui"/>
<xs:enumeration value="lun"/>
<xs:enumeration value="luo"/>
<xs:enumeration value="lus"/>
<xs:enumeration value="mac"/>
<xs:enumeration value="mac"/>
<xs:enumeration value="mad"/>
<xs:enumeration value="mag"/>
<xs:enumeration value="mah"/>
<xs:enumeration value="mai"/>
<xs:enumeration value="mak"/>
<xs:enumeration value="mal"/>
<xs:enumeration value="man"/>
<xs:enumeration value="mao"/>
<xs:enumeration value="map"/>
<xs:enumeration value="mar"/>
<xs:enumeration value="mas"/>
<xs:enumeration value="may"/>
<xs:enumeration value="may"/>
<xs:enumeration value="mdf"/>
<xs:enumeration value="mdr"/>
<xs:enumeration value="men"/>
<xs:enumeration value="mga"/>
<xs:enumeration value="mic"/>
<xs:enumeration value="min"/>
<xs:enumeration value="mis"/>
<xs:enumeration value="mkh"/>
<xs:enumeration value="mlg"/>
<xs:enumeration value="mlt"/>
<xs:enumeration value="mnc"/>
<xs:enumeration value="mni"/>
```

```
<xs:enumeration value="mno"/>
<xs:enumeration value="moh"/>
<xs:enumeration value="mol"/>
<xs:enumeration value="mon"/>
<xs:enumeration value="mos"/>
<xs:enumeration value="mul"/>
<xs:enumeration value="mun"/>
<xs:enumeration value="mus"/>
<xs:enumeration value="mwr"/>
<xs:enumeration value="myn"/>
<xs:enumeration value="myv"/>
<xs:enumeration value="nah"/>
<xs:enumeration value="nai"/>
<xs:enumeration value="nap"/>
<xs:enumeration value="nau"/>
<xs:enumeration value="nav"/>
<xs:enumeration value="nbl"/>
<xs:enumeration value="nde"/>
<xs:enumeration value="ndo"/>
<xs:enumeration value="nds"/>
<xs:enumeration value="nep"/>
<xs:enumeration value="new "/>
<xs:enumeration value="nia"/>
<xs:enumeration value="nic"/>
<xs:enumeration value="niu"/>
<xs:enumeration value="nno"/>
<xs:enumeration value="nob"/>
<xs:enumeration value="nog"/>
<xs:enumeration value="non"/>
<xs:enumeration value="nor"/>
<xs:enumeration value="nso"/>
<xs:enumeration value="nub"/>
<xs:enumeration value="nya"/>
<xs:enumeration value="nym"/>
<xs:enumeration value="nyn"/>
<xs:enumeration value="nyo"/>
<xs:enumeration value="nzi"/>
<xs:enumeration value="oci"/>
<xs:enumeration value="oji"/>
<xs:enumeration value="ori"/>
<xs:enumeration value="orm"/>
<xs:enumeration value="osa"/>
<xs:enumeration value="oss"/>
<xs:enumeration value="ota"/>
<xs:enumeration value="oto"/>
<xs:enumeration value="paa"/>
<xs:enumeration value="pag"/>
<xs:enumeration value="pal"/>
<xs:enumeration value="pam"/>
<xs:enumeration value="pan"/>
<xs:enumeration value="pap"/>
<xs:enumeration value="pau"/>
<xs:enumeration value="peo"/>
<xs:enumeration value="per"/>
<xs:enumeration value="per"/>
<xs:enumeration value="phi"/>
<xs:enumeration value="phn"/>
<xs:enumeration value="pli"/>
<xs:enumeration value="pol"/>
<xs:enumeration value="pon"/>
<xs:enumeration value="por"/>
<xs:enumeration value="pra"/>
<xs:enumeration value="pro"/>
<xs:enumeration value="pus"/>
<xs:enumeration value="que"/>
<xs:enumeration value="raj"/>
```

```
<xs:enumeration value="rap"/>
<xs:enumeration value="rar"/>
<xs:enumeration value="roa"/>
<xs:enumeration value="roh"/>
<xs:enumeration value="rom"/>
<xs:enumeration value="rum"/>
<xs:enumeration value="run"/>
<xs:enumeration value="rus"/>
<xs:enumeration value="sad"/>
<xs:enumeration value="sag"/>
<xs:enumeration value="sah"/>
<xs:enumeration value="sai"/>
<xs:enumeration value="sal"/>
<xs:enumeration value="sam"/>
<xs:enumeration value="san"/>
<xs:enumeration value="sas"/>
<xs:enumeration value="sat"/>
<xs:enumeration value="scc"/>
<xs:enumeration value="sco"/>
<xs:enumeration value="scr"/>
<xs:enumeration value="scr"/>
<xs:enumeration value="sel"/>
<xs:enumeration value="sem"/>
<xs:enumeration value="sga"/>
<xs:enumeration value="sgn"/>
<xs:enumeration value="shn"/>
<xs:enumeration value="sid"/>
<xs:enumeration value="sin"/>
<xs:enumeration value="sio"/>
<xs:enumeration value="sit"/>
<xs:enumeration value="sla"/>
<xs:enumeration value="slo"/>
<xs:enumeration value="slv"/>
<xs:enumeration value="sma"/>
<xs:enumeration value="sme"/>
<xs:enumeration value="smi"/>
<xs:enumeration value="smj"/>
<xs:enumeration value="smn"/>
<xs:enumeration value="smo"/>
<xs:enumeration value="sms"/>
<xs:enumeration value="sna"/>
<xs:enumeration value="snd"/>
<xs:enumeration value="snk"/>
<xs:enumeration value="sog"/>
<xs:enumeration value="som"/>
<xs:enumeration value="son"/>
<xs:enumeration value="sot"/>
<xs:enumeration value="spa"/>
<xs:enumeration value="srd"/>
<xs:enumeration value="srr"/>
<xs:enumeration value="ssa"/>
<xs:enumeration value="ssw"/>
<xs:enumeration value="suk"/>
<xs:enumeration value="sun"/>
<xs:enumeration value="sus"/>
<xs:enumeration value="sux"/>
<xs:enumeration value="swa"/>
<xs:enumeration value="swe"/>
<xs:enumeration value="syr"/>
<xs:enumeration value="tah"/>
<xs:enumeration value="tai"/>
<xs:enumeration value="tam"/>
<xs:enumeration value="tat"/>
<xs:enumeration value="tel"/>
<xs:enumeration value="tem"/>
<xs:enumeration value="ter"/>
```

```
            <xs:enumeration value="tet"/>
            <xs:enumeration value="tgk"/>
            <xs:enumeration value="tgl"/>
            <xs:enumeration value="tha"/>
            <xs:enumeration value="tib"/>
            <xs:enumeration value="tig"/>
            <xs:enumeration value="tir"/>
            <xs:enumeration value="tiv"/>
            <xs:enumeration value="tkl"/>
            <xs:enumeration value="tli"/>
            <xs:enumeration value="tmh"/>
            <xs:enumeration value="tog"/>
            <xs:enumeration value="ton"/>
            <xs:enumeration value="tpi"/>
            <xs:enumeration value="tsi"/>
            <xs:enumeration value="tsn"/>
            <xs:enumeration value="tso"/>
            <xs:enumeration value="tuk"/>
            <xs:enumeration value="tum"/>
            <xs:enumeration value="tup"/>
            <xs:enumeration value="tur"/>
            <xs:enumeration value="tut"/>
            <xs:enumeration value="tvl"/>
            <xs:enumeration value="twi"/>
            <xs:enumeration value="tyv"/>
            <xs:enumeration value="udm"/>
            <xs:enumeration value="uga"/>
            <xs:enumeration value="uig"/>
            <xs:enumeration value="ukr"/>
            <xs:enumeration value="umb"/>
            <xs:enumeration value="und"/>
            <xs:enumeration value="urd"/>
            <xs:enumeration value="uzb"/>
            <xs:enumeration value="vai"/>
            <xs:enumeration value="ven"/>
            <xs:enumeration value="vie"/>
            <xs:enumeration value="vol"/>
            <xs:enumeration value="vot"/>
            <xs:enumeration value="wak"/>
            <xs:enumeration value="wal"/>
            <xs:enumeration value="war"/>
            <xs:enumeration value="was"/>
            <xs:enumeration value="wel"/>
            <xs:enumeration value="wen"/>
            <xs:enumeration value="wln"/>
            <xs:enumeration value="wol"/>
            <xs:enumeration value="xal"/>
            <xs:enumeration value="xho"/>
            <xs:enumeration value="yao"/>
            <xs:enumeration value="yap"/>
            <xs:enumeration value="yid"/>
            <xs:enumeration value="yor"/>
            <xs:enumeration value="ypk"/>
            <xs:enumeration value="zap"/>
            <xs:enumeration value="zen"/>
            <xs:enumeration value="zha"/>
            <xs:enumeration value="znd"/>
            <xs:enumeration value="zul"/>
            <xs:enumeration value="zun"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="ISO-4217">
        <xs:annotation>
            <xs:documentation>Currency (http://www.xe.com/iso4217.htm)</xs:documentation>
        </xs:annotation>
        <xs:restriction base="xs:string">
```

```
<xs:enumeration value="AED"/>
<xs:enumeration value="AFA"/>
<xs:enumeration value="ALL"/>
<xs:enumeration value="AMD"/>
<xs:enumeration value="ANG"/>
<xs:enumeration value="AOA"/>
<xs:enumeration value="ARS"/>
<xs:enumeration value="AUD"/>
<xs:enumeration value="AWG"/>
<xs:enumeration value="AZM"/>
<xs:enumeration value="BAM"/>
<xs:enumeration value="BBD"/>
<xs:enumeration value="BDT"/>
<xs:enumeration value="BGN"/>
<xs:enumeration value="BHD"/>
<xs:enumeration value="BIF"/>
<xs:enumeration value="BMD"/>
<xs:enumeration value="BND"/>
<xs:enumeration value="BOB"/>
<xs:enumeration value="BRL"/>
<xs:enumeration value="BSD"/>
<xs:enumeration value="BTN"/>
<xs:enumeration value="BWP"/>
<xs:enumeration value="BYR"/>
<xs:enumeration value="BZD"/>
<xs:enumeration value="CAD"/>
<xs:enumeration value="CDF"/>
<xs:enumeration value="CHF"/>
<xs:enumeration value="CLP"/>
<xs:enumeration value="CNY"/>
<xs:enumeration value="COP"/>
<xs:enumeration value="CRC"/>
<xs:enumeration value="CUP"/>
<xs:enumeration value="CVE"/>
<xs:enumeration value="CYP"/>
<xs:enumeration value="CZK"/>
<xs:enumeration value="DJF"/>
<xs:enumeration value="DKK"/>
<xs:enumeration value="DOP"/>
<xs:enumeration value="DZD"/>
<xs:enumeration value="EEK"/>
<xs:enumeration value="EGP"/>
<xs:enumeration value="ERN"/>
<xs:enumeration value="ETB"/>
<xs:enumeration value="EUR"/>
<xs:enumeration value="FJD"/>
<xs:enumeration value="FKP"/>
<xs:enumeration value="GBP"/>
<xs:enumeration value="GEL"/>
<xs:enumeration value="GGP"/>
<xs:enumeration value="GHC"/>
<xs:enumeration value="GIP"/>
<xs:enumeration value="GMD"/>
<xs:enumeration value="GNF"/>
<xs:enumeration value="GTQ"/>
<xs:enumeration value="GYD"/>
<xs:enumeration value="HKD"/>
<xs:enumeration value="HNL"/>
<xs:enumeration value="HRK"/>
<xs:enumeration value="HTG"/>
<xs:enumeration value="HUF"/>
<xs:enumeration value="IDR"/>
<xs:enumeration value="ILS"/>
<xs:enumeration value="IMP"/>
<xs:enumeration value="INR"/>
<xs:enumeration value="IQD"/>
```

```
<xs:enumeration value="IRR"/>
<xs:enumeration value="ISK"/>
<xs:enumeration value="JEP"/>
<xs:enumeration value="JMD"/>
<xs:enumeration value="JOD"/>
<xs:enumeration value="JPY"/>
<xs:enumeration value="KES"/>
<xs:enumeration value="KGS"/>
<xs:enumeration value="KHR"/>
<xs:enumeration value="KMF"/>
<xs:enumeration value="KPW"/>
<xs:enumeration value="KRW"/>
<xs:enumeration value="KWD"/>
<xs:enumeration value="KYD"/>
<xs:enumeration value="KZT"/>
<xs:enumeration value="LAK"/>
<xs:enumeration value="LBP"/>
<xs:enumeration value="LKR"/>
<xs:enumeration value="LRD"/>
<xs:enumeration value="LSL"/>
<xs:enumeration value="LTL"/>
<xs:enumeration value="LVL"/>
<xs:enumeration value="LYD"/>
<xs:enumeration value="MAD"/>
<xs:enumeration value="MDL"/>
<xs:enumeration value="MGA"/>
<xs:enumeration value="MKD"/>
<xs:enumeration value="MMK"/>
<xs:enumeration value="MNT"/>
<xs:enumeration value="MOP"/>
<xs:enumeration value="MRO"/>
<xs:enumeration value="MTL"/>
<xs:enumeration value="MUR"/>
<xs:enumeration value="MVR"/>
<xs:enumeration value="MWK"/>
<xs:enumeration value="MXN"/>
<xs:enumeration value="MYR"/>
<xs:enumeration value="MZM"/>
<xs:enumeration value="NAD"/>
<xs:enumeration value="NGN"/>
<xs:enumeration value="NIO"/>
<xs:enumeration value="NOK"/>
<xs:enumeration value="NPR"/>
<xs:enumeration value="NZD"/>
<xs:enumeration value="OMR"/>
<xs:enumeration value="PAB"/>
<xs:enumeration value="PEN"/>
<xs:enumeration value="PGK"/>
<xs:enumeration value="PHP"/>
<xs:enumeration value="PKR"/>
<xs:enumeration value="PLN"/>
<xs:enumeration value="PYG"/>
<xs:enumeration value="QAR"/>
<xs:enumeration value="ROL"/>
<xs:enumeration value="RUR"/>
<xs:enumeration value="RWF"/>
<xs:enumeration value="SAR"/>
<xs:enumeration value="SBD"/>
<xs:enumeration value="SCR"/>
<xs:enumeration value="SDD"/>
<xs:enumeration value="SEK"/>
<xs:enumeration value="SGD"/>
<xs:enumeration value="SHP"/>
<xs:enumeration value="SIT"/>
<xs:enumeration value="SKK"/>
<xs:enumeration value="SLL"/>
```

```
            <xs:enumeration value="SOS"/>
            <xs:enumeration value="SPL"/>
            <xs:enumeration value="SRG"/>
            <xs:enumeration value="STD"/>
            <xs:enumeration value="SVC"/>
            <xs:enumeration value="SYP"/>
            <xs:enumeration value="SZL"/>
            <xs:enumeration value="THB"/>
            <xs:enumeration value="TJS"/>
            <xs:enumeration value="TMM"/>
            <xs:enumeration value="TND"/>
            <xs:enumeration value="TOP"/>
            <xs:enumeration value="TRL"/>
            <xs:enumeration value="TTD"/>
            <xs:enumeration value="TVD"/>
            <xs:enumeration value="TWD"/>
            <xs:enumeration value="TZS"/>
            <xs:enumeration value="UAH"/>
            <xs:enumeration value="UGX"/>
            <xs:enumeration value="USD"/>
            <xs:enumeration value="UYU"/>
            <xs:enumeration value="UZS"/>
            <xs:enumeration value="VEB"/>
            <xs:enumeration value="VND"/>
            <xs:enumeration value="VUV"/>
            <xs:enumeration value="WST"/>
            <xs:enumeration value="XAF"/>
            <xs:enumeration value="XAG"/>
            <xs:enumeration value="XAU"/>
            <xs:enumeration value="XCD"/>
            <xs:enumeration value="XDR"/>
            <xs:enumeration value="XOF"/>
            <xs:enumeration value="XPD"/>
            <xs:enumeration value="XPF"/>
            <xs:enumeration value="XPT"/>
            <xs:enumeration value="YER"/>
            <xs:enumeration value="YUM"/>
            <xs:enumeration value="ZAR"/>
            <xs:enumeration value="ZMK"/>
            <xs:enumeration value="ZWD"/>
        </xs:restriction>
    </xs:simpleType>
</xs:schema>
```

6. Protocol Procedures

6.1 Rules

1. Messages must be formulated correctly, according to the prescribed syntax and semantics.
2. A transaction begins when an operation, initiated by the client, bears a new TransactionId. From the client perspective this means a TransactionId that it has never used before; from the server perspective this means a TransactionId that it has never seen before. (If the client initiates an operation with a transactionId that it has used before but the server has never seen before, that consititutes an error.) Transactions are initiated in this manner only. All subsequent operations bearing that TransactionId belong to that transaction.

3. Messages must not be sent out-of-role.
    - Only clients may initiate Question, Clarification, or ActionRequest operations.
    - Only servers may initiate Answer, RequestClarification, or Status operations.
4. Operations may not be initiated out-of-sequence. Specifically: An Answer or RequestClarification operation may not be initiated if there has not been a Question operation. A ConstraintReply operation may not be initiated if there has not been a Constraint operation.
5. Aside from the exceptions listed in rules 3 and 4, any message, from either client or server, is valid at any time.
6. ReferenceId must be used properly. A system must not assign a ReferenceId to more than one initiating operation (RequestClarification, ActionRequest, or clarification). The responding party should not use an id that has not been assigned by the requestor, and should echo the ReferenceId in the appropriate response.

6.2 State Tables

Since protocol procedures are fairly simple, state tables are not defined for this protocol. It should be especially noted that the protocol is not intended to enforce any message sequencing other than stated above.

6.3 Error Handling

Behavior that violates any of the above rules is a protocol error. Treatment of errors is not emphasized by this protocol. Basic guideline: it is recommended that a system detecting a protocol error send an error message. Having done so, the system, at its discretion, may then continue or terminate the transaction.

6.4 Out-of-Band Messages

When a message is received with a transactionId of a closed or unknown transaction, the protocol does not address how that message is to be handled. The recipient may discard the message, attempt to process it, send it back, etc. The recomended procedure is to respond with an error message.

6.5 Termination

Either client or server may consider a transaction to be closed whenever it wants to. A well-behaved system will (but is not required to) attempt to either (a) carry out the transaction to its logical conclusion, or (b) signal the peer that it considers the transaction to be closed.

7. Mappings to Lower Layer Protocols

Lower layer protocols used by QATP

- HTTP/HTTPS – hypertext/secure hypertext transfer protocol (Version 1.1 RFC 2616)

- SMTP – simple mail transfer protocol (RFC 821)
- SOAP – simple object access protocol (Version 1.1 W3C working group). An XML protocol for exchange of information in a decentralized, distributed environment. It consists of three parts: an envelope that defines a framework for describing what is in a message and how to process it, a set of encoding rules for expressing instances of application-defined datatypes, and a convention for representing remote procedure calls and responses.

This list is not intended to exclude the addition of other protocols in the future.

QATP messages are encapsulated using SOAP and communicated using either HTTP/HTTPS or SMTP.

Bindings

- QATP to SOAP bindings are described via a WSDL specification (to be supplied later). WSDL, web services description language (Version 1.1 W3C working group),
  is an XML-based description language for describing network services as a set of endpoints (where "endpoint" in this context means DRD) operating on messages containing either document-oriented or procedure-oriented information.
- SOAP to HTTP/HTTPS bindings are structured as described in Section 6 of Simple Object Access Protocol (SOAP) 1.1 (W3C Note 08 May 2000).
- SOAP to SMTP bindings are structured as described in SOAP Version 1.2 Email Binding (W3C Note 3 July 2002).