Open Financial Exchange Specification 1.0.2

May 30, 1997

Chapters 8 – 10

Contents

8. Activation & Account Information

8.1 Overview

The Signup message set defines three messages to help users get setup with their FI:

- Enrollment – informs FI that a user wants to use Open Financial Exchange and requests that a password be returned
- Accounts – asks the FI to return a list of accounts and the services supported for each account
- Activation – allows a client to tell the FI which services a user wants on each account

There is also a message to request name and address changes.

Clients use the account information request on a regular basis to look for changes in a user's account information. A time stamp is part of the request so that a server has to report only new changes. Account activation requests are subject to data synchronization, and will allow multiple clients to learn how the other clients have been enabled.

In Open Financial Exchange files, the <SIGNUPMSGSV1> aggregate identifies the Signup message.

8.2 Approaches to User Sign-Up with Open Financial Exchange

The message sets in this chapter are designed to allow both FIs and clients to support a variety of sign-up procedures. There are four basic steps a user needs to go through to complete the sign-up:

1. **Select the FI**. Open Financial Exchange does not define this step or provide message sets to support it. Client developers and FIs can let a user browse or search this information on a web site, or might define additional message sets to do this within the client. At the conclusion of this step, the client will have some minimal profile information about the FI, including the set of services supported and the URL to use for the next step.

2. **Enrollment and password acquisition.** In this step, the user identifies and authenticates itself to the FI *without a password*. In return, the user obtains a password (possibly temporary) to use with Open Financial Exchange. FIs can perform this entire step over the telephone, through a combination of telephone requests and a mailed response, or at the FI web site. FIs can also use the Open Financial Exchange enrollment message to do this by means of the client. The response can contain a temporary password or users can wait for a mailed welcome letter containing the password.

3. **Account Information.** In this step, the user obtains a list of accounts available for use with Open Financial Exchange, and which specific services are available for each account. Even if users have enrolled over the telephone, clients will still use this message set to help users properly set up the accounts within the client. Clients periodically check back with the FI for updates.

4. **Service Activation.** The last step is to activate specific services on specific accounts. The activation messages support this step. Synchronization is applied to these messages to ensure that other clients are aware of activated services.

The combination of media-interface through which an FI accomplishes these steps can vary. FIs might wish to do steps two through four over the telephone. Clients will still use Open Financial Exchange messages in steps 3 and 4 to automatically set up the client based on the choices made by the user over the phone. Other FIs might wish to have the entire user experience occur within the client. Either way, the Open Financial Exchange sign-up messages support the process.

8.3 Users and Accounts

To support the widest possible set of FIs, Open Financial Exchange assumes that individual users and accounts are in a many-to-many relationship. Consider a household with three accounts:

- Checking 1 – held individually by one spouse
- Checking 2 – held jointly by both
- Checking 3 – held individually by the other spouse

Checking 2 should be available to either spouse, and the spouse holding Checking 1 should be able to see both Checking 1 and 2.

Open Financial Exchange expects FIs to give each user their own user ID and password. Each user will go through the enrollment step separately. A given account need only be activated once for a service; not once for each user. Clients will use the account information and activation messages to combine information about jointly-held accounts.

If an FI prefers to have a single user ID and password per household or per master account, it will have to make this clear to users through the enrollment process. It is up to the FI to assign a single user ID and password that can access all three of the checking accounts described above.

8.4 Enrollment and Password Acquisition <ENROLLRQ> <ENROLLRS>

The main purpose of the enrollment message is to communicate a user's intent to access the FI by way of Open Financial Exchange and to acquire a password for future use with Open Financial Exchange. Some FIs might return a user ID and an initial password in the enrollment response, while others will send them by way of regular mail.

> ***NOTE:*** *Because the client does not know the user ID and password when the it sends the enrollment request, the <SONRQ> will not contain a valid user ID or password. However, since tags cannot be set to blank values, the user ID and password must contain at least one character each.*

Enrollment requests are not subject to synchronization. If the client does not receive a response, it will simply re-request the enrollment. If a user successfully enrolls from another client before the first client obtains a response, the server should respond to subsequent requests from the first client with status code:

```
13501 - user already enrolled.
```

8.4.1 User IDs

The Open Financial Exchange <SONRQ> requires a user ID to uniquely identify a user to an FI. The server must accept the user ID with or without punctuation.

Many FIs in the United States use social security numbers (SSNs) as the ID. Others create IDs that are unrelated to the users' SSNs. Some FIs have existing user IDs that they use for other online activities that they want to use for Open Financial Exchange as well. FIs might also create new IDs specifically for Open Financial Exchange. Finally, some FIs might assign IDs while others might allow users to create them. Because users do not usually know either their Open Financial Exchange sign-on user ID or their password at time of enrollment, the enrollment response is designed to return both. The enrollment request allows users to optionally provide a user ID, which an FI can interpret as their existing online ID or a suggestion for what their new user ID should be. Ideally, the enrollment process should explain ID syntax to users.

8.4.2 Enrollment Request

The enrollment request captures enough information to identify and authenticate a user as being legitimate and that it has a relationship with the FI.

FIs might require that an account number be entered as part of the identification process. However, this is discouraged since the account information request is designed to automatically obtain all account information, avoiding the effort and potential mistakes of a user-supplied account number.

It is **RECOMMENDED** that FIs provide detailed specifications for user IDs and passwords along with information about the services available when a user is choosing an FI.

The enrollment request must appear within an <ENROLLTRNRQ> transaction wrapper.

Tag	*Description*
<ENROLLRQ>	Enrollment-request aggregate
<FIRSTNAME>	First name of user, *A-32*
<MIDDLENAME>	Middle name of user, *A-32*
<LASTNAME>	Last name of user, *A-32*
<ADDR1>	Address line 1, *A-32*
<ADDR2>	Address line 2, *A-32*
<ADDR3>	Address line 3, *A-32*
<CITY>	City, *A-32*
<STATE>	State or province, *A-5*
<POSTALCODE>	Postal code, *A-11*
<COUNTRY>	3-letter country code from ISO/DIS-3166, *A-3*
<DAYPHONE>	Daytime telephone number, *A-32*
<EVEPHONE>	Evening telephone number, *A-32*
<EMAIL>	Electronic e-mail address, *A-80*
<USERID>	Actual user ID if already known, or preferred user ID if user can pick, *A-32*
<TAXID>	ID used for tax purposes (such as SSN), may be same as user ID, *A-32*
<SECURITYNAME>	Mother's maiden name or equivalent, *A-32*
<DATEBIRTH>	Date of birth, *date*
<*XXX*ACCTFROM>	An account description aggregate for an existing account at the FI, for identification purposes only. For example, <BANKACCTFROM> or <INVACCTFROM>.
</*XXX*ACCTFROM>	
</ENROLLRQ>	

This enrollment request is intended for use only by individuals. Business enrollment will be defined in a later release.

8.4.3 Enrollment Response

The main purpose of the enrollment response is to acknowledge the request. In those cases where FIs permit delivery of an ID and a temporary password, the response also provides for this. Otherwise the server will send the real response to the user by way of regular mail, electronic mail, or over the telephone. If enrollment is successful, but the server does not return the ID and password in the response, a server is REQUIRED to use status code 10 and provide some information to the user by means of the <MESSAGE> element in the <STATUS> aggregate about what to expect next.

The enrollment response must appear within an <ENROLLTRNRS> transaction wrapper.

Tag	*Description*
<ENROLLRS>	Enrollment-response aggregate
<TEMPPASS>	Temporary password, *A-32*
<USERID>	User ID, *A-32*
<DTEXPIRE>	Time the temporary password expires (if <TEMPPASS> included), *datetime*
</ENROLLRS>	

8.4.4 Enrollment Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
13000	User ID & password will be sent out-of-band (INFO)
13500	Unable to enroll user (ERROR)
13501	User already enrolled (ERROR)

8.4.5 Examples

An enrollment request:

```
<ENROLLTRNRQ>
   <TRNUID>12345
   <ENROLLRQ>
      <FIRSTNAME>Joe
      <MIDDLENAME>Lee
      <LASTNAME>Smith
      <ADDR1>21 Main St.
      <CITY>Anytown
      <STATE>TX
      <POSTALCODE>87321
      <COUNTRY>USA
      <DAYPHONE>123-456-7890
      <EVEPHONE>987-654-3210
      <EMAIL>jsmith@isp.com
      <USERID>jls
      <TAXID>123-456-1234
      <SECURITYNAME>jbmam
      <DATEBIRTH>19530202
   </ENROLLRQ>
</ENROLLTRNRQ>
```

And the reply might be:

```
<ENROLLTRNRS>
   <TRNUID>12345
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
```

```
   <ENROLLRS>
      <TEMPPASS>changeme
      <USERID>jls
      <DTEXPIRE>19970105
   </ENROLLRS>
</ENROLLTRNRS>
```

8.5 Account Information

Account information requests ask a server to identify and describe all of the accounts accessible by the signed-on user. The definition of *all* is up to the FI. At a minimum, it is **RECOMMENDED** that a server include information about all accounts that it can activate for one or more Open Financial Exchange services. To give the user a complete picture of his relationship with an FI, FIs can give information on other accounts, even if those accounts are available only for limited Open Financial Exchange services.

Some service providers do not have prior knowledge of user account information. The profile allows these servers to report this, and clients then know to ask users for account information rather than reading it from the server.

Clients can perform several tasks for users with this account information. First, the information helps a client set up a user for online services by giving it a precise list of its account information and available services for each. Clients can set up their own internal state as well as prepare service activation requests with no further typing by users. This can eliminate data entry mistakes in account numbers, routing transit numbers, and so forth.

Second, FIs can provide limited information on accounts that would not ordinarily be suitable to Open Financial Exchange services. For example, a balance-only statement download would be useful for certificates of deposits even though a customer or an FI might not want or allow CDs to be used for full statement download.

For each account, there is one <ACCTINFO> aggregate returned. The aggregate includes one service-specific account information aggregate for each service available to that account. That, in turn, provides the service-specific account identification. Common to each service-specific account information aggregate is the <SVCSTATUS> tag, which indicates the status of this service on this account.

A server should return joint accounts (accounts for which more than one user ID can be used to access the account) for either user.

Requests and responses include a <DTACCTUP> element. Responses contain the last time a server updated the information. Clients *are* **REQUIRED** *to* send this in a subsequent request, and servers are **REQUIRED** to compare this to the current modification time and only send information if it is more recent. The server sends the entire account information response if the client's time is older; there is no attempt to incrementally update specific account information.

8.5.1 Request <ACCTINFORQ>

The <ACCTINFORQ> request must appear within an <ACCTINFOTRNRQ> transaction wrapper.

Tag	*Description*
<ACCTINFORQ>	Account-information-request aggregate
<DTACCTUP>	Last <DTACCTUP> received in a response, *datetime*
</ACCTINFORQ>	

8.5.2 Response <ACCTINFORS>

The <ACCTINFORS> response must appear within an <ACCTINFOTRNRS> transaction wrapper.

Tag	*Description*
<ACCTINFORS>	Account-information-response aggregate
<DTACCTUP>	Date and time of last update to this information on the server, *datetime*
<ACCTINFO>	Zero or more account information aggregates
</ACCTINFO>	
</ACCTINFORS>	End of account information response

8.5.3 Account Information Aggregate <ACCTINFO>

Tag	*Description*
<ACCTINFO>	Account-information-record aggregate
<DESC>	Description of the account, *A-80*
<PHONE>	Telephone number for the account, *A-32*
<XXXACCTINFO>	Service-specific account information, defined in each service chapter. For a given service *XXX*, there can be at most one <*XXX*ACCTINFO> returned. For example, you cannot return two <BANKACCTINFO> aggregates.
<XXXACCTFROM>	Service-specific account identification
</XXXACCTFROM>	
<SVCSTATUS>	AVAIL = Available, but not yet requested PEND = Requested, but not yet available ACTIVE = In use
</XXXACCTINFO>	
</ACCTINFO>	

NOTE: *A server uses the <DESC> field to convey the FI's preferred name for the account, such as "PowerChecking." It should not include the account number.*

8.5.4 Status Codes

Code	Meaning
0	Success (INFO)
1	Client is up-to-date (INFO)
2000	General error (ERROR)

8.5.5 Examples

An account information request:

```
<ACCTINFOTRNRQ>
   <TRNUID>12345
   <ACCTINFORQ>
      <DTACCTUP>19960101

   </ACCTINFORQ>
</ACCTINFOTRNRQ>
```

And a response for a user with access to one account, supporting banking:

```
<ACCTINFOTRNRS>
   <TRNUID>12345
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <ACCTINFORS>
      <DTACCTUP>19960102
      <ACCTINFO>
      <DESC>Power Checking
      <PHONE>8002223333

         <BANKACCTINFO>
            <BANKACCTFROM>
               <BANKID>1234567789
               <ACCTID>12345
               <ACCTTYPE>CHECKING
            </BANKACCTFROM>
         <SUPTXDL>Y
         <XFERSRC>Y
         <XFERDEST>Y
         <SVCSTATUS>ACTIVE
         </BANKACCTINFO>
      </ACCTINFO>
   </ACCTINFORS>
</ACCTINFOTRNRS>
```

8.6 Service Activation

Clients inform FIs that they wish to start, modify, or terminate a service for an account by sending service activation requests. These are subject to data synchronization, and servers should send responses to inform clients of any changes, even if the changes originated on the server.

Clients use these records during the initial user sign-up process. Once a client learns about the available accounts and services (by using the account information request above, or by having a user directly enter the required information), it sends a series of service ADD requests.

If a user changes any of the identifying information about an account, the client sends a service activation request containing both the old and the new account information. Servers should interpret this as a change in the account, not a request to transfer the service between two existing accounts, and all account-based information such as synchronization tokens should continue. If a user or FI is reporting that service should be moved between two existing accounts, service must be terminated for the old account and started for the new account. The new account will have reset token histories, as with any new service.

Each service to be added, changed, or removed is contained in its own request because the same real-world account might require different <*XXX*ACCTFROM> aggregates depending on the type of service.

8.6.1 Activation Request and Response

8.6.1.1 Request <ACCTRQ>

The <ACCTRQ> request must appear within an <ACCTTRNRQ> transaction wrapper.

Tag	*Description*
<ACCTRQ>	Account-service-request aggregate
Server account service actions. Specify either <SVCADD>, <SVCCHG>, or <SVCDEL>	Action aggregate, either <SVCADD>, <SVCCHG>, or <SVCDEL>
<SCVCADD>	Service-addition aggregate
</SCVCADD>	
-or-	
<SVCCHG>	Service-change aggregate
</SVCCHG>	
-or-	
<SVCDEL>	Service-deletion aggregate
</SVCDEL>	
<SVC>	Service to be added/changed/deleted
	BANKSVC = Banking service BPSVC = Payments service INVSVC = Investments
</ACCTRQ>	

8.6.1.2 Response <ACCTRS>

The <ACCTRS> response must appear within an <ACCTTRNRS> transaction wrapper.

Tag	*Description*
<ACCTRS>	Account-service-response aggregate
Action identification. Specify either <SVCADD>, <SVCCHG>, or <SVCDEL>	
<SCVCADD>	Service-addition aggregate
</SCVCADD>	
-or-	
<SVCCHG>	Service-change aggregate
</SVCCHG>	
-or-	
<SVCDEL>	Service-deletion aggregate
</SVCDEL>	
</*ACTION*>	
<SVC>	Service to be added/changed: BANKSVC = Banking service BPSVC = Payments service INVSVC = Investments
<SVCSTATUS>	AVAIL = Available, but not yet requested PEND = Requested, but not yet available ACTIVE = In use
</ACCTRS>	

8.6.1.3 Service Add Aggregate <SVCADD>

Tag	*Description*
<SVCADD>	Service-addition aggregate
<*XXX*ACCTTO>	Service-specific-account-identification aggregate (for example, <BANKACCTTO> or <INVACCTTO>)
</*XXX*ACCTTO>	
</SVCADD>	

8.6.1.4 Service Change Aggregate <SVCCHG>

Tag	*Description*
<SVCCHG>	Service-change aggregate
<*XXX*ACCTFROM>	Service-specific-account-identification aggregate (for example, <BANKACCTFROM> or <INVACCTFROM>)
</*XXX*ACCTFROM>	
<*XXX*ACCTTO>	Service-specific-account-identification aggregate (for example, <BANKACCTTO> or <INVACCTTO>)
</*XXX*ACCTTO>	
</SVCCHG>	

8.6.1.5 Service Delete Aggregate <SVCDEL>

Tag	*Description*
<SVCDEL>	Service-deletion aggregate
<*XXX*ACCTFROM>	Service-specific-account-identification aggregate (for example, <BANKACCTFROM> or <INVACCTFROM>)
</*XXX*ACCTFROM>	
</SVCDEL>	

8.6.1.6 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2002	General account error (ERROR)
2006	Source account not found (ERROR)
2007	Source account closed (ERROR)
2008	Source account not authorized (ERROR)
2009	Destination account not found (ERROR)
2010	Destination account closed (ERROR)
2011	Destination account not authorized (ERROR)
13502	Invalid service (ERROR)

8.6.2 Service Activation Synchronization

Service activation requests are subject to the standard data synchronization protocol. The scope of these requests and the <TOKEN> is the user ID. The request and response tags are <ACCTSYNCRQ> and <ACCTSYNCRS>.

8.6.3 Examples

Activating a payment:

```
<ACCTTRNRQ>
   <TRNUID>12345
   <ACCTRQ>
      <SVCADD>
         <BANKACCTTO>
            <BANKID>1234567789
            <ACCTID>12345
            <ACCTTYPE>CHECKING
         </BANKACCTTO>
      </SVCADD>
      <SVC>BPSVC
   </ACCTRQ>
</ACCTTRNRQ>
```

A response:

```
<ACCTTRNRS>
   <TRNUID>12345
      <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <ACCTRS>
      <SVCADD>
         <BANKACCTTO>
            <BANKID>1234567789
            <ACCTID>12345
            <ACCTTYPE>CHECKING
         </BANKACCTTO>
      </SVCADD>
      <SVC>BPSVC
      <SVCSTATUS>ACTIVE
   </ACCTRS>
</ACCTTRNRS>
```

8.7 Name and Address Changes <CHGUSERINFORQ> <CHGUSERINFORS>

Users may request that an FI update the official name, address, phone, and e-mail information using the <CHGUSERINFORQ>. Only the fields that should be changed are sent. The response reports all of the current values. For security reasons, some of the fields in the <ENROLLRQ> cannot be changed online, such as tax ID.

The transaction tag is <CHGUSERINFOTRNRQ> and <CHGUSERINFOTRNRS>. These messages are subject to synchronization, <CHGUSERINFOSYNCRQ> and <CHGUSERINFOSYNCRS>.

8.7.1 <CHGUSERINFORQ>

Tag	*Description*
<CHGUSERINFORQ>	Change-user-information-request aggregate
<FIRSTNAME>	First name of user, *A-32*
<MIDDLENAME>	Middle name of user, *A-32*
<LASTNAME>	Last name of user, *A-32*
<ADDR1>	Address line 1, *A-32*
<ADDR2>	Address line 2, *A-32*
<ADDR3>	Address line 3, *A-32*
<CITY>	City, *A-32*
<STATE>	State or province, *A-5*
<POSTALCODE>	Postal code, *A-11*
<COUNTRY>	3-letter country code from ISO/DIS-3166, *A-3*
<DAYPHONE>	Daytime telephone number, *A-32*
<EVEPHONE>	Evening telephone number, *A-32*
<EMAIL>	Electronic e-mail address, *A-80*
</CHGUSERINFORQ>	

8.7.2 <CHGUSERINFORS>

Tag	*Description*
<CHGUSERINFORS>	Change-user-information-request aggregate
<FIRSTNAME>	First name of user, *A-32*
<MIDDLENAME>	Middle name of user, *A-32*
<LASTNAME>	Last name of user, *A-32*
<ADDR1>	Address line 1, *A-32*
<ADDR2>	Address line 2, *A-32*
<ADDR3>	Address line 3, *A-32*
<CITY>	City, *A-32*
<STATE>	State or province, *A-5*
<POSTALCODE>	Postal code, *A-11*
<COUNTRY>	3-letter country code from ISO/DIS-3166, *A-3*
<DAYPHONE>	Daytime telephone number, *A-32*
<EVEPHONE>	Evening telephone number, *A-32*
<EMAIL>	Electronic e-mail address, *A-80*
<DTINFOCHG>	Date and time of update *datetime*
</CHGUSERINFORS>	

8.7.3 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
13503	Cannot change user information (ERROR)

8.8 Signup Message Set Profile Information

A server must include the following aggregates as part of the profile <MSGSETLIST> response, since every server must support at least the account information and service activation messages. In the <ENROLLPROF> aggregate, servers indicate how enrollment should proceed: via the client, a given web page, or a text message directing users to some other method (such as a phone call).

Tag	*Description*
<SIGNUPMSGSET>	Signup-message-set-profile-information aggregate
<SIGNUPMSGSETV1>	Opening tag for V1 of the message set profile information
<MSGSETCORE>	Common message set information, defined in Chapter 7, "FI Profile"
</MSGSETCORE>	
Enrollment options. Choose one of <CLIENTENROLL>, <WEBENROLL>, or <OTHERENROLL>.	
<CLIENTENROLL**>**	Client-based enrollment supported
<ACCTREQUIRED>	Y if account number is required as part of enrollment, *Boolean*
</CLIENTENROLL**>**	
-or-	
<WEBENROLL**>**	Web-based enrollment supported
<URL>	URL to start enrollment process, *URL*
</WEBENROLL>	
-or-	
<OTHERENROLL**>**	Some other enrollment process
<MESSAGE>	Message to consumer about what to do next (for example, a phone number), *A-80*
</OTHERENROLL**>**	
<CHGUSERINFO>	Y if server supports client-based user information changes, *Boolean*
<AVAILACCTS>	Y if server can provide information on accounts with SVCSTATUS available, N means client should expect to ask user for specific account information, *Boolean*
<CLIENTACTREQ>	Y if server allows clients to make service activation requests (<ACCTRQ>), N if server will only advise clients via synchronization of service additions, changes, or deletions. *Boolean*
</SIGNUPMSGSETV1>	
</SIGNUPMSGSET>	

9. Customer to FI Communication

9.1 The E-Mail Message Set

The e-mail message set includes two messages: generic e-mail and generic MIME requests by way of URLs. In Open Financial Exchange files, the message set name is EMAILMSGSV1.

9.2 E-Mail Messages

Open Financial Exchange allows consumers and FIs to exchange messages. The message body can be placed in HTML so that FIs can provide some graphic structure to the message. Keep in mind that, as with regular World Wide Web browsing, an Open Financial Exchange client might not support some or all of the HTML formatting, so the text of the message must be clear on its own. Clients can request that graphics (the images referenced in an <IMG> tag) be sent as part of the response file, or clients can separately request those elements. If a server sends images, it should use the standard procedure for incorporating external data as described in Chapter 2, “Structure.” Servers are not required to support HTML or to send images, even if the client asks.

A user or an FI can originate a message. E-mail messages are subject to data synchronization so that a server can send a response again if it is lost or if it is used by multiple clients.

Because e-mail messages cannot be replied to immediately, the response should just echo back the original message (so that data synchronization will get this original e-mail message to other clients). When the FI is ready to reply, it should generate an unsolicited response (<TRNUID>0) and the client will pick this up during synchronization.

Client Sends	*Server Responds*
Account information	
From, To	
Subject	
Message	
	Account information
	From, To
	Subject
	Message
	Type

9.2.1 Regular vs. Specialized E-Mail

Several services with Open Financial Exchange define e-mail requests and responses that contain additional information specific to that service. To simplify implementation for clients and servers, this section defines a <MAIL> aggregate that Open Financial Exchange uses in all e-mail requests and responses. For regular e-mail, the only additional information is an account-from aggregate and whether to include images in the e-mail response or not.

9.2.2 Basic <MAIL> Aggregate

Tag	*Description*
<MAIL>	Core e-mail aggregate
<USERID>	User ID such as SSN, *A-32*
<DTCREATED>	When message was created, *datetime*
<FROM>	Who the message is from, *A-32*
<TO>	Who the message should be delivered to, *A-32*
<SUBJECT>	Subject of message (plain text, not HTML), *A-60*
<MSGBODY>	Body of message, HTML-encoded or plain text depending on <USEHTML>,
	HTML-encoded text = *A-10000*
	Plain text = *A-2000*
</MSGBODY>	End of message
<INCIMAGES>	Include images in the message body. *Boolean*
<USEHTML>	Y for HTML-formatted text. N for plain text. See section 9.2.2.2 for more information. *Boolean*
</MAIL>	

9.2.2.1 <INCIMAGES>

The meaning of the <INCIMAGES> tag depends on whether the tag appears in a request or response.

When used in a request, <INCIMAGES> indicates whether the client accepts mail that includes images in the message body.

When used in a request...	*Description*
<INCIMAGES>Y	The client accepts mail that includes images in the message body. In this case, the server can choose whether to send images in the response.
<INCIMAGES>N	The client does not accept mail that includes images in the message body. In this case, the server must not send images in the response.

When used in a response, <INCIMAGES> indicates whether the server included images in the message body.

When used in a response…	*Description*
<INCIMAGES>Y	The server included images in the message body.
<INCIMAGES>N	The server did not include images in the message body.

9.2.2.2 <USEHTML>

The meaning of the <USEHTML> tag depends on whether the tag appears in a request or response.

When used in a request, <USEHTML> indicates whether the client sends and accepts HTML-formatted text in the message body.

When used in a request…	*Description*
<USEHTML>Y	The client is including HTML-formatted text in the message body. In addition, the client will accept mail responses that include HTML-formatted text in the message body. In this case, a server can choose whether to respond with HTML-formatted text or plain text.
<USEHTML>N	The client is not including HTML-formatted text in the message body. In addition the client will not accept mail responses that include HTML-formatted text in the message body.

When used in a response, <USEHTML> indicates whether the message body includes HTML-formatted text or plain text.

When used in a response…	*Description*
<USEHTML>Y	The server is including HTML-formatted text in the message body.
<USEHTML>N	The server is including only plain text in the message body.

> ***NOTE:*** *When using HTML for the message body, clients and servers are* ***REQUIRED*** *to enclose the HTML in an SGML-marked section to protect the HTML markup: <![CDATA [... html ...]]>. For an example, see section 9.2.5.*

9.2.3 E-Mail <MAILRQ> <MAILRS>

E-mail is subject to synchronization. The transaction tag is <MAILTRNRQ> / <MAILTRNRS> and the synchronization tag is <MAILSYNCRQ> / <MAILSYNCRS>.

Tag	*Description*
<MAILRQ>	E-mail-message-request aggregate
<MAIL>	Core e-mail aggregate
</MAIL>	
</MAILRQ>	

In a response, the <TRNUID> is zero if this is an unsolicited message. Otherwise, it should contain the <TRNUID> of the user's original message. It is RECOMMENDED that servers include the <MESSAGE> of the user's message as part of the reply <MESSAGE>. The <MESSAGE> contents can include carriage returns to identify desired line breaks.

Tag	*Description*
<MAILRS>	E-mail-message-response aggregate
<MAIL>	Core e-mail aggregate
</MAIL>	
</MAILRS>	

9.2.3.1 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
16500	HTML not allowed (ERROR)
16501	Unknown mail To: (ERROR)

9.2.4 E-Mail Synchronization <MAILSYNCRQ> <MAILSYNCRS>

E-mail presents a special case with regards to synchronization. Since FIs will not immediately reply to a user's e-mail, the response to the user's e-mail only echoes the request and confirms that the e-mail was successfully received. The client receives the real response to the e-mail following a synchronization request.

Note that this synchronization action expects only the basic <MAILRS> responses. Specialized e-mail is received by means of their own synchronization requests.

Tag	Description
<MAILSYNCRQ>	E-mail-synchronization-request aggregate
Client synchronization option; <TOKEN>, <TOKENONLY>, or <REFRESH>	
<TOKEN>	Previous value of <TOKEN> received for this type of synchronization request from server; 0 for first-time requests; *token*
<TOKENONLY>	Request for just the current <TOKEN> without the history, *Boolean*
<REFRESH>	Request for refresh of current state, *Boolean*
<REJECTIFMISSING>	If Y, do not process requests if client <TOKEN> is out of date, *Boolean*
<INCIMAGES>	Y if the client accepts mail with images in the message body, N if the client does not accept mail with images in the message body, *Boolean*
<USEHTML>	Y if client wants an HTML response, N if client wants plain text, *Boolean*
<MAILTRNRQ>	Mail-transaction-request aggregate (0 or more)
</MAILTRNRQ>	
</MAILSYNCRQ>	

Tag	Description
<MAILSYNCRS>	E-mail-synchronization-response. aggregate
<TOKEN>	Server history marker, *token*
<LOSTSYNC>	Y if the token in the synchronization request is older than the earliest entry in the server's history table. In this case, some responses have been lost. N if the token in the synchronization request is newer than or matches a token in the server's history table. *Boolean*
<MAILTRNRS>	Missing e-mail response transactions (0 or more)
</MAILTRNRS>	
</MAILSYNCRS>	

9.2.5 E-Mail Example

In this example, a consumer requests information about the checking statement just downloaded. Since the financial institution will not immediately answer the inquiry, the immediate response only echoes the consumer's request and confirms that the request was successfully received.

The client receives the real response at a later time following a mail synchronization request. For an example of the mail synchronization request and response, see section 9.2.5.1.

> ***NOTE:*** *This example omits the <OFX> top level and the signon <SONRQ>. Since this example uses HTML for the message body, it must protect the HTML content in an SGML CDATA-marked section.*

The request:

```
<MAILTRNRQ>
   <TRNUID>54321
   <MAILRQ>
      <MAIL>
         <USERID>123456789
         <FROM>James Hackleman
         <TO>Noelani Federal Savings
         <SUBJECT>What do I need to earn interest?
         <DTCREATED>19960305
         <MSGBODY><![ CDATA [<HTML><BODY>I didn't earn any interest this month.
Can you please tell me what I need to do to earn interest on this
account?</BODY></HTML>
]]></MSGBODY>
         <INCIMAGES>N
         <USEHTML>Y
      </MAIL>
   </MAILRQ>
</MAILTRNRQ>
```

The response from the FI:

```
<MAILTRNRS>
   <TRNUID>54321
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
      </STATUS>
   <MAILRS>
      <MAIL>
         <USERID>123456789
         <FROM>James Hackleman
         <TO>Noelani Federal Savings
         <SUBJECT>What do I need to earn interest?
         <DTCREATED>19960305
         <MSGBODY><![ CDATA [<HTML><BODY>I didn't earn any interest this month.
Can you please tell me what I need to do to earn interest on this
account?</BODY></HTML>
]]></MSGBODY>
         <INCIMAGES>N
         <USEHTML>Y
      </MAIL>
   </MAILRS>
</MAILTRNRS>
```

9.2.5.1 E-Mail Synchronization Example

In the following example, the client has not yet received the reply to the e-mail sent in the previous example, so its <TOKEN> is one less than the server's. The server replies by giving the current <TOKEN> and the missed response.

```
<MAILSYNCRQ>
   <TOKEN>101
   <REJECTIFMISSING>N
   <INCIMAGES>N
   <USEHTML>Y
</MAILSYNCRQ>

<MAILSYNCRS>
   <TOKEN>102
```

```
   <MAILTRNRS>
     <TRNUID>0                          <!-- server initiated response →
     <STATUS>
        <CODE>0
        <SEVERITY>INFO
     </STATUS>
     <MAILRS>
     <MAIL>
        <USERID>123456789
        <DTCREATED>19960307
        <FROM>Noelani Federal Savings
        <TO>James Hackleman
        <SUBJECT>Re: What do I need to earn interest?
        <MSGBODY>><![ CDATA [<HTML><BODY>You need to maintain $1000 in this
account to earn interest. Because your balance was only $750 this month, no
interest was earned. You could also switch to our new Checking Extra plan that
always pays interest. Call us or check our web page http://www.fi.com/check-
plans.html for more information.
Sincerely,
Customer Service Department

Original message:
I didn't earn any interest this month. Can you please tell me what I need to do
to earn interest on this account?</BODY></HTML>
]]></MSGBODY>
        <INCIMAGES>N
        <USEHTML>Y
     </MAIL>
     </MAILRS>
   </MAILTRNRS>
</MAILSYNCRS>
```

9.3 Get HTML Page

Some responses contain values that are URLs intended to be separately fetched by clients. Clients can use their own HTTP libraries to perform this fetch outside of the Open Financial Exchange specification. However, to insulate clients against changes in transport technology, and to allow for fetches that require the protection of an authenticated signon by a specific user, Open Financial Exchange defines a transaction roughly equivalent to an HTTP Get. Any MIME type can be retrieved, including images as well as HTML pages.

9.3.1 MIME Get Request and Response <GETMIMERQ> <GETMIMERS>

The following table lists the components of a request:

Tag	*Description*
<GETMIMERQ>	Get-MIME-request aggregate
<URL>	URL, *URL*
</GETMIMERQ>	

The response simply echoes the URL. The actual response, whether HTML, an image, or some other type, is always sent as a separate part of the file using multi-part MIME.

Tag	*Description*
<GETMIMERS>	Get-MIME-response aggregate
<URL>	URL, *URL*
</GETMIMERS>	

9.3.1.1 Status Codes

Code	*Meaning*
0	Success (INFO)
2000	General error (ERROR)
2019	Duplicate request (ERROR)
16502	Invalid URL (ERROR)
16503	Unable to get URL (ERROR)

9.3.2 MIME Example

A request:

```
<GETMIMETRNRQ>
   <TRNUID>54321
   <GETMIMERQ>
      <URL>http://www.fi.com/apage.html
   </GETMIMERQ>
</GETMIMETRNRQ>
```

A response – the full file is shown here to illustrate the use of multi-part MIME:

```
HTTP 1.0 200 OK
Content-Type: multipart/x-mixed-replace; boundary =--boundary—

--boundary--
Content-Type: application/x-ofx
Content-Length: 8732

OFXHEADER:100
DATA:OFXSGML
VERSION:102
SECURITY:TYPE1
ENCODING:USASCII

<OFX>
     <!-- signon not shown
     message set wrappers not shown -->
<GETMIMETRNRS>
   <TRNUID>54321
   <STATUS>
      <CODE>0
      <SEVERITY>INFO
   </STATUS>
   <GETMIMERS>
      <URL>http://www.fi.com/apage.html
   </GETMIMERS>
</GETMIMETRNRS>
</OFX>

--boundary--
Content-Type: text/html
<HTML>
   <!-- standard HTML page -->
</HTML>

--boundary--
```

9.4 E-Mail Message Set Profile Information

If either or both of the messages in the e-mail message set are supported, the following aggregate must be included in the profile <MSGSETLIST> response.

Tag	*Description*
<EMAILMSGSET>	E-mail-message-set-profile-information aggregate
<EMAILMSGSETV1>	Opening tag for V1 of the message set profile information
<MSGSETCORE>	Common message set information, defined in the profile chapter
</MSGSETCORE>	
<MAILSUP>	Y if server supports generic e-mail message, *Boolean*
<GETMIMESUP>	Y if server supports get MIME message, *Boolean*
</EMAILMSGSETV1>	
</EMAILMSGSET>	

10. Recurring Transactions

Open Financial Exchange enables users to automate transactions that occur on a regular basis. Recurring transactions are useful when a customer has payments or transfers, for example, that repeat at regular intervals. The customer can create a "model" at the server for automatic generation of these instructions. The model in turn creates payments or transfers until it is canceled or expires. After the user creates a recurring model at the server, the server can relieve the user from the burden of creating these transactions; it generates the transactions on its own, based on the operating parameters of the model.

10.1 Creating a Recurring Model

The client must provide the following information to create a model:

- Type of transaction generated by the model (payment or transfer)
- Frequency of recurring transaction
- Total number of recurring transactions to generate
- Service-specific information, such as transfer date, payment amount, payee address

The model creates each transaction some time before its due date, usually thirty days. This allows the user to retrieve the transactions in advance of posting. This also gives the user the opportunity to modify or cancel individual transactions without changing the recurring model itself.

When a model is created, it can generate several transactions immediately. The model does not automatically return responses for the newly created transactions. It returns a response only to the request that was made to create the model. For this reason, clients should send a synchronization request along with the request to create a model. This allows the server to return the newly created transaction responses, as well as the response to the request to set up a new model.

10.2 Recurring Instructions <RECURRINST>

The Recurring Instructions aggregate is used to specify the schedule for a repeating instruction. It is passed to the server when a recurring transfer or payment model is first created.

Tag	*Description*
<RECURRINST>	Recurring-Instructions aggregate
<NINSTS>	Number of instructions
	If this tag is absent, the schedule is open-ended, *N-3*
<FREQ>	Frequency, see section 10.2.1
</RECURRINST>	

10.2.1 Values for <FREQ>

Value	*Description*
WEEKLY	Weekly
BIWEEKLY	Biweekly
TWICEMONTHLY	Twice a month
MONTHLY	Monthly
FOURWEEKS	Every four weeks
BIMONTHLY	Bimonthly
QUARTERLY	Quarterly
SEMIANNUALLY	Semiannually
ANNUALLY	Annually

Rules for calculating recurring dates of WEEKLY, BIWEEKLY, and TWICEMONTHLY are as follows:

- WEEKLY = starting date for first transaction, starting date + 7 days for the second
- TWICEMONTHLY = starting date for first, starting date + 15 days for the second
- BIWEEKLY = starting date for first, starting date + 14 days for the second

Examples:

Start date of May 2: next transaction date for WEEKLY is May 9; TWICEMONTHLY is May 17; next transfer date for BIWEEKLY is May 16.

Start date of May 20: next date for WEEKLY is May 27; TWICEMONTHLY is June 4; next date for BIWEEKLY is June 3.

TWICEMONTHLY recurring transactions will occur each month on those days adjusting for weekends and holidays. BIWEEKLY will occur every 14 days.

10.2.2 Examples

The following example illustrates the creation of a repeating payment. The payment repeats on a monthly basis for 12 months. All payments are for $395.

The request:

```
.
.
.
<RECPMTRQ>
   <RECURRINST>
      <NINSTS>12
      <FREQ>MONTHLY
   </RECURRINST>
   <PMTINFO>
      <BANKACCTFROM>
         <BANKID>555432180
```

```
         <ACCTID>763984
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <TRNAMT>395.00
      <PAYEEID>77810
      <PAYACCT>444-78-97572
      <DTDUE>19971115
      <MEMO>Auto loan payment
   </PMTINFO>
</RECPMTRQ>
.
.
.
```

The response includes the <RECSRVRTID> that the client can use to cancel or modify the model:

```
.
.
.
<RECPMTRS>
   <RECSRVRTID>387687138
   <RECURRINST>
      <NINSTS>12
      <FREQ>MONTHLY

   </RECURRINST>
   <PMTINFO>
      <BANKACCTFROM>
         <BANKID>555432180
         <ACCTID>763984
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <TRNAMT>395.00
      <PAYEEID>77810
      <PAYACCT>444-78-97572
      <DTDUE>19971115
      <MEMO>Auto loan payment
   </PMTINFO>
</RECPMTRS>
.
.
.
```

10.3 Retrieving Transactions Generated by a Recurring Model

Once created, a recurring model independently generates instructions. Since the client has not directly generated these transactions, the client has no record of their creation. To enable users to modify and/or cancel pending instructions, the client must use data synchronization in order to retrieve these transactions.

The client has two purposes for synchronizing state with the server with respect to recurring models:

- Retrieve any added, modified, or canceled recurring models
- Retrieve any added, modified, or canceled transactions generated by any models

The client must be able to synchronize with the state of any models at the server, as well as the state of any transactions generated by the server.

10.4 Modifying and Canceling Individual Transactions

Once created and retrieved by the customer, recurring payments and transfers are almost identical to customer-created payments or transfers. As with ordinary payments or transfers, you can cancel or modify transactions individually. However, because servers generate these transfers, they are different in the following respects:

- Recurring transactions must be retrieved as part of a synchronization request.
- Recurring transactions are related to a model. A server can modify or cancel transactions if the model is modified or canceled.

10.5 Modifying and Canceling Recurring Models

A recurring model can be modified or canceled. When a model is modified, all transactions that it generates in the future will change as well. The client can indicate whether transactions that have been generated, but have not been sent, should be modified as well. The actual elements within a transaction that can be modified differ by service. See the recurring sections within Chapter 11, “Banking,” and Chapter 12, “Payments” for details.

A user can cancel a model immediately or at a future date. If a user cancels the model immediately, the client cancels any transactions that it has not yet sent. If the client schedules the cancel for a future date, the client will not cancel pending transactions.

10.5.1 Examples

Canceling a recurring payment model requires the client to pass the <RECSRVRTID> of the model. The client requests that pending payments also be canceled. The server cancels the model immediately and notifies the client that both the model and any scheduled payments were canceled.

The request:

```
.
.
.
   <RECPMTCANCRQ>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRQ>
.
.
.
```

The response:

```
.
.
.
   <RECPMTCANCRS>
      <RECSRVRTID>387687138
      <CANPENDING>Y
```

```
   </RECPMTCANCRS>
.
.
.
```

The server also cancels any payments that have been generated but not executed. In the example shown above, the client would not learn of this immediately. To receive notification that the model and all generated payments were canceled, the client would need to include a synchronization request in the file. The following example illustrates this alternate approach.

The request file now includes a synchronization request:

```
.
.
.
   <RECPMTCANCRQ>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRQ>
   <PMTSYNCRQ>
      <TOKEN>12345
      <REJECTIFMISSING>N
      <BANKACCTFROM>
         <BANKID>123432123
         <ACCTID>516273
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
   </PMTSYNCRQ>
.
.
.
```

The response file now contains two responses (assuming one payment was pending), one for the canceled model and one for the canceled payment.

```
.
.
.
   <RECPMTCANCRS>
      <RECSRVRTID>387687138
      <CANPENDING>Y
   </RECPMTCANCRS>
   <PMTSYNCRS>
      <TOKEN>3247989384
      <BANKACCTFROM>
         <BANKID>123432123
         <ACCTID>516273
         <ACCTTYPE>CHECKING
      </BANKACCTFROM>
      <PMTTRNRS>
         <TRNUID>10103
         <STATUS>
            <CODE>0
            <SEVERITY>INFO
         </STATUS>
         <PMTCANCRS>
            <SRVRTID>1030155
         </PMTCANCRS>
      </PMTTRNRS>
   </PMTSYNCRS>
.
.
```

.