opentext™

GroupWise® Software Developer Kit

Object Event Notification

October 2023

Legal Notices

Contents

About This Guide

GroupWise Events provides the ability to receive notification of GroupWise events. It allows registered third parties to offer event notification and to be responsive to user queries.

IMPORTANT: Unless otherwise marked, the features in GroupWise Events will work with GroupWise 8 and later versions.

This guide contains the following sections:

Audience

This guide is intended for GroupWise developers.

Feedback

We want to hear your comments and suggestions about this manual and the other documentation included with this product. Please use the User Comment feature at the bottom of each page of the online documentation.

1 Overview

GroupWise Events provides the ability to receive notification of GroupWise events by providing programmatic access to event notifications for external applications.

Configurable event registration filters allow developers to receive notification for only those users and/or events in which they have interest. GroupWise Events offers developers the opportunity to receive both realtime notification of GroupWise events, and, through the event history, access to events that occur while the application is disconnected from the GroupWise POA.

GroupWise Events does this while maintaining responsiveness to user queries and not significantly degrading the overall performance of the POA.

This section provides an introduction to the GroupWise event system and contains information about the following:

XML and SOAP Resources

If you are going to use GroupWise Events, you should be familiar with XML and SOAP concepts, including those described in the following links:

XML Schema Specifications:

- http://www.w3.org/TR/xmlschema-0/ (http://www.w3.org/TR/xmlschema-0)
- http://www.w3.org/TR/xmlschema-1/ (http://www.w3.org/TR/xmlschema-1)
- http://www.w3.org/TR/xmlschema-2/ (http://www.w3.org/TR/xmlschema-2)

SOAP:

- http://www.w3.org/TR/soap12-part0/ (http://www.w3.org/TR/soap12-part0/)
- http://www.w3.org/TR/soap12-part1/ (http://www.w3.org/TR/soap12-part1/)
- http://www.w3.org/TR/soap12-part2/ (http://www.w3.org/TR/soap12-part1/)

Event Records

Event records track GroupWise event data on a user-by-user basis, and are stored in each user's database. Separate event records are created for each application that registers for a given event, meaning that there may be multiple event records for a single event, each intended for a different

receiving application. For example, if ApplicationA registers to receive event notifications for ItemDeleted for User1, and ApplicationB also registers to receive event notifications for ItemDeleted for User1, two separate event records are written to User1's database.

Event records have the following fields:

- TimeStamp: Identifies the time the event occurred.
- RecordID: Identifies the unique event record.
- Container RecordID: Identifies the folder or address book for which the event record was created.
- From Container RecordID: Identifies the source folder or address book (used with FolderItemMove). For proxy logins, this field provides the UserId of the user doing the proxy login.
- Event: Provides a description of the event.
- Key name: Identifies the application for which the event record was created.

By default, event records persist in the database for a maximum of seven days, or until they are removed by the registered application. However, GroupWise Events provides a persistence element in its XML, with a range of 0-20 days, that overrides the seven-day default, and sets an expiration value for event records. When an event record age exceeds the Persistence value, the nightly maintenance routine removes it from the user database.

If event records are not removed by the application within 21 days, the nightly maintenance routine not only removes the event records for that application, but also the configuration settings for that application key. This helps prevent unnecessary event monitoring, and limits the number of unused event records that build up in the system.

Cleanup of event configuration on the system on which the registered third-party application is running is the sole responsibility of the application. GroupWise Events does not provide cleanup or housekeeping on the external application systems.

GroupWise Events

GroupWise event records are configured and enabled through Event Definition Records in the user's database. Event Definition Record settings are only available programmatically, through the interface described in XML Reference.

Applications can also query for POA events. An application, through the use of an individual user's key, can query for all POA events related to that user. Applications can query for specific events and/or for events that occur within a certain time period. The application can also query multiple times on any given event.

GroupWise events are presented to third parties in XML, with tags being defined by the SOAP protocol. GroupWise Events provides access to the following events:

Event Category	Event Name
Address Book	◆ Address book created ◆ Address book deleted ◆ Address book modified

Event Category	Event Name
Address Book Item	◆ Category(s) changed ◆ Email address changed ◆ Item created ◆ Item modified ◆ Instant messaging address changed ◆ Name changed ◆ Phone number changed
Item	◆ Accept level ◆ Accepted ◆ Archived ◆ Attachments ◆ Category ◆ Classification ◆ Completed ◆ Declined ◆ Deleted ◆ Due date ◆ Duration ◆ Expiration date ◆ Item field changed ◆ Item state changed ◆ Marked private ◆ Marked read ◆ Marked unread ◆ Message body ◆ Personal subject ◆ Place ◆ Purged ◆ Security ◆ Send priority ◆ Start date ◆ Task category ◆ Task priority ◆ Unaccept ◆ Unarchived ◆ Uncomplete ◆ Unmarked private

Event Category	Event Name
Folder	◆ Folder accepted (share) ◆ Folder deleted ◆ Folder (item added) ◆ Folder (item deleted) ◆ Folder (item move) ◆ Folder modified ◆ Name ◆ Rights (share)
Login	◆ Login ◆ Login (Proxy) ◆ Login (Trusted Application) ◆ Logout ◆ Session Timeout
Proxy Access	◆ Rights modified ◆ User added ◆ User deleted

Schema

The foundation of the GroupWise Events event system is the schema definition. The GroupWise events schema is stored separately from other SOAP protocol schema files, in events.xsd. The schema file references only the types.xsd file. For more information on types.xsd, see the "Schema Elements" section in the *GroupWise SDK: Web Services (SOAP)* documentation.

For the GroupWise Events schema definition, see Schema.

2 XML Reference

This section describes the XML programming interfaces available for GroupWise Events.

GroupWise Events exposes two level of requests: engine methods and protocol (XML) methods. Applications interested in registering for GroupWise events call protocol methods. Supported protocol methods include the following:

cleanEventConfiguration

Performs nightly maintenance to clean up old events and keys.

Request

```
<cleanEventConfigurationRequest>
</cleanEventConfigurationRequest>
```

Remarks

For each event in the user's database, if the associated persistence setting has been exceeded (a setting of TRUE (1)), the event is deleted from the user's database. If no persistence element is specified, any event record older than the default seven days is deleted. If all events associated with a given key are deleted, the associated key is also deleted.

Any keys older then 20 days are deleted, including the key definition and all associated events.

configureEvents

Updates an existing definition; or, if there is not a definition with the same key, a new definition is added to the user's database. Event definitions can be added at any time, but are not active until the enabled tag on the events element is set TRUE (1).

Request

```
<configureEventsRequest>
   <enabled>
      <events enabled="">
      <key>
           <persistence>
           <ipAddress>
           <port>
           <event>
                <event>
             <event>
         . . .
             <type>
           </event>
         <containers>
         <container>
      </key>
   </events>
</configureEventsRequest>
```

Elements

enabled

Enables/disables event processing. If set to FALSE (0), event processing is disabled for that event configuration only. Other event configurations are not affected.

key

Uniquely identifies the application instance. It is up to the application to control the uniqueness of the application key. GroupWise Events uses any key that it is passed. If two applications or two instances of an application pass the same key, GroupWise Events maps them both to one event configuration structure in the user's database.

persistence

Controls removal of old event records in the user's database. Any event records an application has not removed are cleaned up during nightly maintenance according to the persistence element associated with that record. Persistence is defined in days, between 0 and 20 inclusive. If persistence is not defined, GroupWise Events defaults to seven days.

ipAddress

Identifies the IP address, or HTTP URL, that should be used for event notification.

port

Specifies the IP port on which the application is listening for event notifications.

http

Specifies whether event notification should occur through TCP/IP or HTTP. If set to FALSE (0), the value in ipAddress is treated as an IP address and a TCP/IP stream is used to provide notification. If set to TRUE (1), the value in ipAddress is treated as an HTTP URL (for example, http://www.acme.com/events).

event

Identifies a GroupWise event that has occurred. Event item types are stored in a <space> separated list. Event modified fields are also stored in a <space> separated list.

containers

Specifies the containers in which the application wants to track the specified events. Each container element, if present in containers, maps to one of the following events: FolderItemAdd, FolderItemDelete, ItemDelete, ItemUndelete. If no container is specified, events are reported system-wide.

Remarks

configureEventsRequest uses the Events structure, which defines the event configuration for one application.

This method does not affect other definitions that are already defined.

Example

Following is an example of an event configuration request:

```
<configureEventsRequest>
   <enabled>1</enabled>
  <events enabled="1">
         <key>Acme</key>
     <persistence>0</persistence>
           <ipAddress>app1.widgets.com</ipAddress>
           <port>5221</port>
           <event>
             <event>FolderItemAdd</event>
             <event>FolderItemMove</event>
             <event>FolderItemRemove</event>
             <event>ItemAccept</event>
             <event>ItemComplete</event>
             <event>ItemDecline</event>
```

```
            <event>ItemDelete</event>
            <event>ItemMarkRead</event>
            <event>ItemMarkUnread</event>
            <event>ItemPurge</event>
            <event>ItemUndelete</event>
            <type>Appointment Mail Note Task</type>
          </event>
        <containers>
           <container>
           <container>7.AutoDomain.AutoPO1.100.0.1.0.1@16</container>
           <container>A.AutoDomain.AutoPO1.100.0.1.0.1@19</container>
      </containers>
   </events>
</configureEventsRequest>
```

getEventConfiguration

Requests one or more configuration definitions. The getEventConfigurationRequest response returns the specified configuration definition, or, if the key element is not specified, all event definitions system-wide.

Request

```
<getEventConfigurationRequest/>
   <key>
```

Response

```
<getEventConfigurationResponse>
   <enabled>
     <events>
           <event enabled="">
             <key>
             <persistence>
             <ipAddress>
             <port>
             <http>
             <event>
               <event>
                   <event>
               . . .
                   </event>
                   <type>
          </event>
          <event enabled="">
                <key>
                <persistence>
                <ipAddress>
                <port>
                <http>
                <event>
                     <event>
                  <event>
                  . . .
                    </event>
             </event>
         <containers>
         <container>
     </events>
     <status>
          <code>
     </status>
</getEventConfigurationResponse>
```

Elements

key

Uniquely identifies the application instance. It is up to the application to control the uniqueness of the application key. GroupWise Events uses any key that it is passed. If two applications or two instances of an application pass the same key, GroupWise Events maps them both to one event configuration structure in the user's database.

enabled

Enables/disables event processing. If set to FALSE (0), event processing is disabled for that event configuration only. Other event configurations are not affected.

persistence

Controls removal of old event records in the user's database. Any event records an application has not removed are cleaned up during nightly maintenance according to the persistence element associated with that record. Persistence is defined in days, between 0 and 20. If persistence is not defined, GroupWise Events defaults to seven days.

ipAddress

Identifies the IP address, or HTTP URL, that should be used for event notification.

port

Specifies the IP port on which the application is listening for event notifications.

http

Specifies whether event notification should occur through TCP/IP or HTTP. If set to FALSE (0), the value in ipAddress is treated as an IP address and a TCP/IP stream is used to provide notification. If set to TRUE (1), the value in ipAddress is treated as an HTTP URL (for example, http://www.acme.com/events).

event

Identifies a GroupWise event that has occurred. Event item types are stored in a <space> separated list. Event modified fields are also stored in a <space> separated list.

containers

Specifies the containers in which the application wants to track the specified events. Each container element, if present in containers, maps to one of the following events: FolderItemAdd, FolderItemDelete, ItemDelete, ItemUndelete. If no container is specified, events are reported system-wide.

Example

Following is a sample response to the self-terminating and the associated response:

```
<getEventConfigurationResponse>
   <enabled>1</enabled>
     <events>
           <event enabled="1">
             <key>Acme</key>
             <persistence>0</persistence>
             <ipAddress>app1.widgets.com</ipAddress>
             <port>5221</port>
             <http>1</http>
             <event>
               <event>ItemAccept</event>
               <event>ItemComplete</event>
                   <event>ItemDecline</event>
                   <event>ItemDelete</event>
                   <event>ItemPurge</event>
                   <event>ItemMarkRead</event>
                   <event>ItemUndelete</event>
                   <event>ItemMarkUnread</event>
                   <event>FolderItemAdd</event>
                   <event>FolderItemMove</event>
                   <event>FolderItemRemove</event>
             </event>
             <type>Appointment Mail Note Task</type>
           </event>
           <event enabled="1">
             <key>AB</key>
             <persistence>0</persistence>
             <ipAddress>http://prestons/</ipAddress>
             <port>5221</port>
             <http>1</http>
             <event>
               <event>AddressBookDelete</event>
                   <event>AddressBookAdd</event>
                   <event>AddressBookItemDelete</event>
                   <event>AddressBookItemAdd</event>
             </event>
           </event>
         <container>
            <container>7.AutoDomain.AutoPO1.100.0.1.0.1@16</container>
            <container>A.AutoDomain.AutoPO1.100.0.1.0.1@19</container>
         </containers>
      </events>
   <status>
      <code>0</code>
   </status>
</getEventConfigurationResponse>
```

getEvents

Returns, by default, those events that have occurred since the last call to getEventsRequest. It is possible to set one IP address and port to listen for changes for multiple users. After all users for that IP address and port have been removed from the notification list, the IP socket is closed. Following is an example of an event request:

Request

```
<getEventsRequest>
   <key>
   <from>
   <until>
   <remove>
   <notify>
</getEventsRequest>
```

Response

```
<getEventsResponse>
   <events>
      <event>
           <event>
           <id>
           <timeStamp>
           <container>
           <key>
        </event>
      . . .
   </events>
   <status>
     <code>
   </status>
</getEventsResponse>
```

Elements

key

Uniquely identifies the application instance. It is up to the application to control the uniqueness of the application key. GroupWise Events uses any key that it is passed. If two applications or two instances of an application pass the same key, GroupWise Events maps them both to one event configuration structure in the user's database.

from

Provides the starting point of the date range for which events are returned to the application.

until

Provides the ending point of the date range for which events are returned to the application.

count

Specifies how many events to return in one response. If count is not specified, all events are returned.

remove

Specifies whether the event record is removed from the database upon a successful response to getEventsRequest. When set to TRUE (1), the event is removed from the database.

notify

Specifies whether the application wants to be notified of new occurrences of the types of events it has requested. The notification process maintains a list of all applications that want to be notified when that event occurs. Each time an application receives event notifications, it is removed from the notification list. To be placed back on the notification list, an application must send getEventsRequest with notify TRUE (1).

event

Identifies a GroupWise event that has occurred. Event item types are stored in a <space> separated list. Event modified fields are also stored in a <space> separated list.

id

Item identifier as defined in SOAP.

timeStamp

Specifies the date and time that the GroupWise event occurred.

container

Specifies the location in the GroupWise system where the event occurred.

status

Specifies whether getEventRequest was successful.

code

Provides the error number related to the event, if there is one.

Example

Following is a sample getEventsRequest:

```
<getEventsRequest>
   <key>Acme</key>
   <remove>1</remove>
   <notify>1</notify>
</getEventsRequest>
```

Following is a sample response to getEventsRequest:

```
<getEventsResponse>
   <events>
      <event>
           <event>FolderItemAdd</event>
              <id>41937EE0.AutoDomain.AutoPO1.100.1363230.1.272D.1</id>
            <timeStamp>2004-11-11T22:01:55Z</timeStamp>
              <container>7.AutoDomain.AutoPO1.100.0.1.0.1@16</container>
            <key>Acme</key>
           </event>
        <event>
     <event>FolderItemAdd</event>
           <id>41937F2C.AutoDomain.AutoPO1.100.1363230.1.272E.1</id>
           <timeStamp>2004-11-11T22:03:10Z</timeStamp>
     <container>7.AutoDomain.AutoPO1.100.0.1.0.1@16</container>
           <key>Acme</key>
        </event>
   </events>
   <status>
     <code>0</code>
   </status>
</getEventsResponse>
```

getItems

Retrieves items forwarded via getEventResponse, as defined in GroupWise SOAP. For more information on GroupWise SOAP, including more information on the structure and elements described herein, see the "Schema Elements" in the *GroupWise SDK: Web Services (SOAP)* documentation.

Request

```
<getItemsRequest>
   <container>
   <filter>
     <element>
           <op>
           <field>
         <value>
        </element>
   </filter>
</getItemsRequest>
```

Elements

container

Specifies the container the item is in.

count

Specifies how many items to retrieve in one response. If count is not specified, all items are returned.

element

Identifies the item to retrieve.

field

Specifies the GroupWise field or attribute to use in the element filter process.

filter

Specifies how to filter the items.

op

Specifies the boolean operator to use in the element filter process.

value

Specifies the value of the field or element to use in the element filter process.

Remarks

getItemsRequest requires a filter to match the IDs instead of using the IDs directly.

Example

Following is a sample getItemsResponse:

```
<getItemsRequest>
   <container>A.AutoDomain.AutoPO1.100.0.1.0.1@13</container>
   <filter>
      <element>
           <op>eq</op>
           <field>id</field>
         <value>419482D2.AutoDomain.AutoPO1.100.1363230.1.2734.1
            </value>
        </element>
   </filter>
</getItemsRequest>
```

removeEventConfiguration

Removes the configuration definition for a specified key.

Request

```
<removeEventConfigurationRequest>
   <key>
</removeEventConfigurationRequest>
```

Elements

key

Uniquely identifies the application instance. It is up to the application to control the uniqueness of the application key. GroupWise Events uses any key that it is passed. If two applications or two instances of an application pass the same key, GroupWise Events maps them both to one event configuration structure in the user's database.

removeEvents

Removes events from the database without an application first reviewing the events.

Request

```
<removeEventsRequest>
   <key>
</removeEventsRequest>
```

Elements

key

Uniquely identifies the application instance. It is up to the application to control the uniqueness of the application key. GroupWise Events uses any key that it is passed. If two applications or two instances of an application pass the same key, GroupWise Events maps them both to one event configuration structure in the user's database.

3 Objects

GroupWise Events includes several objects to control the event processing. These objects are part of the GroupWise Object API. The following objects are supported:

Event

Represents a GroupWise system event that has occurred.

Properties

This object contains the the following properties:

Property	Description
ReadOnly [Event]() As GWEEventContents	The enumeration of an event that has occurred.
ReadOnly Fields() As String	Fields that were changed in a modify event.
ReadOnly FolderID As String	Specifies the folder in which the event occurred. For example, the folder where an item was deleted.
ReadOnly FromID() As String	The folder ID of folder item move.
ReadOnly Key() As String	Identifies the GroupWise event that has occurred.
ReadOnly MessageID() As String	Item identifier as defined in SOAP.
ReadOnly TimeStamp() As Date	Date and time the event occurred.

EventCollection

Functions as a container for event records of events that have occurred.

Methods

This object contains the following methods:

Item2

Properties

This object contains the following properties:

Property	Description
ReadOnly Count() As Integer	Specifies the number of event records to get.

EventConfiguration

Specifies the configuration of a given event.

Methods

This object contains the following methods:

Delete

Properties

This object has the following properties:

Property	Description
Enabled() As Boolean	Enables/disables the event definition.
Http() As Boolean	Determines whether IPAddress refers to an HTTP destination or a TCP/IP destination.
IPAddress() As String	Specifies the address to which the EventDefinition object is returned.
Key() As String	Specifies an application-specific key name used in routing event information.
Persistence() As Integer	Specifies the number of days to keep the event record available to the application. The event record is maintained in the user database.
Port() As Integer	Specifies the port number to be used in conjunction with IPAddress.
ReadOnly Property Events() As EventDefinition	Returns the EventDefinition object.

EventConfigurations

Controls configuration of the events.

Methods

This object contains the following methods:

- Add
- Clean
- Item

Properties

This object has the following properties:

Property	Description
ReadOnly Count() As Integer	Specifies the number of event records to get.

Remarks

EventConfigurations also supports enumerations such as the For Each function in Visual Basic*. The collection starts at 1.

EventDefinition

Specifies the specific GroupWise events that are tracked in this event configuration.

Properties

Most of the following properties are Boolean values that specify whether the event is tracked in this event configuration:

- AddressBookDelete() As Boolean
- AddressBookAdd() As Boolean
- AddressBookItemAdd() As Boolean
- AddressBookItemDelete() As Boolean
- AddressBookItemModify() As Boolean
- AddressBookModify() As Boolean
- FolderAccept() As Boolean
- FolderAdd() As Boolean
- FolderDelete() As Boolean
- FolderItemAdd() As Boolean
- FolderItemDelete() As Boolean
- FolderItemMove() As Boolean
- FolderModify() As Boolean
- ItemAccept() As Boolean
- ItemArchive() As Boolean
- ItemComplete() As Boolean
- ItemDecline() As Boolean
- ItemDelete() As Boolean
- ItemMarkPrivate() As Boolean
- ItemMarkRead() As Boolean
- ItemMarkUnread() As Boolean
- ItemModify() As Boolean
- ItemPurge() As Boolean
- ItemUnarchive() As Boolean
- ItemUndelete() As Boolean
- Login() As Boolean
- Logout() As Boolean
- ProxyAdd() As Boolean
- ProxyDelete() As Boolean
- ProxyLogin() As Boolean
- ProxyModify() As Boolean

- SessionTimedOut() As Boolean
- TrustedApplicationLogin() As Boolean
- Fields() As String

  A space-delimited string of valid field types for each event type. Valid string values are found in Remarks.
- Types() As String

  A space-delimited string of item types. Valid string values are listed in Remarks.

Remarks

Valid Field string values include the following:

- AcceptLevel
- Attachment
- Category
- Classification
- DueDate
- Duration
- EmailAddress
- ExpirationDate
- IMAddress
- MessageBody
- Name
- PersonalSubject
- PhoneNumber
- Place
- Rights
- Security
- SendPriority
- StartDate
- TaskCategory
- TaskPriority

Valid Types string values include the following:

- AddressBookItem
- Appointment
- CalendarItem
- Contact
- Group
- Mail

- Note
- Organization
- PhoneMessage
- Resource
- Task

4 Methods

GroupWise Events objects contain the methods described in this section.

Add

Used with EventConfigurations to add a new event configuration.

Syntax

```
Add(
   ByVal Key As String)
As EventConfiguration
```

Parameters

key

Specifies the new configuration to add.

Remarks

Returns a new EventConfigurations object, which is used to configure events.

Clean

Used with EventConfigurations to clean up the events for a user.

Syntax

```
Clean(
   ByVal All)
As Boolean
```

Parameters

All

Specifies how the nightly maintenance routine cleans up old events. If All is FALSE (0), only the old events are removed. If All is TRUE (1), all events and the event configurations are removed.

Delete

Used with EventConfiguration to delete an EventConfiguration object.

Syntax

```
Delete()
```

Item

Used with EventConfigurations to access a particular EventConfiguration object.

Syntax

```
Item(
   ByVal Index As Integer)
As EventConfiguration
```

Parameters

Index

Offset into the EventConfigurations collection. An index value of one is the first EventConfiguration; a value of two is the second EventConfiguration, and so forth.

Remarks

Use the Count property, as described in EventConfigurations, to find out the number of EventConfiguration objects in the collection.

Item2

Used with EventCollection to retrieve a particular event record from an EventCollection.

Syntax

```
Item (
   ByVal Index As Integer)
As Event
```

Parameters

Index

Offset into the EventCollection. An index value of one is the first Event; a value of two is the second Event, and so forth.

Remarks

Use the Count property, as described in EventCollection to find out the number of EventConfiguration objects in the collection.

This object also supports enumeration, such as the ForEach function in Visual Basic. The collection starts at one (1.)

A Schema

The GroupWise events schema follows:

```
<?xml version="1.0" ?>
<xs:schema
    targetNamespace="urn:novell:schemas:ns:events"
     xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:types="http://schemas.novell.com/2003/10/NCSP/types.xsd"
     xmlns:tns="urn:novell:schemas:ns:events">

     <xs:element name="key" type="xs:string"/>
     <xs:element name="event" type="tns:EventType"/>
     <xs:element name="field" type="tns:FieldList" minOccurs="0"/>

     <xs:complexType name="Event">
           <xs:sequence>
              <xs:element ref="tns:event"/>
             <xs:element ref="types:id"/>
             <xs:element name="timeStamp" type="xs:dateTime"
               minOccurs="0"/>
             <xs:element ref="tns:field"/>
             <xs:element name="container" type="types:uid"
               minOccurs="0"/>
             <xs:element name="from" type="types:uid" minOccurs="0"/>
             <xs:element name="key" type="xs:string" minOccurs="0"/>
           </xs:sequence>
     </xs:complexType>

     <xs:complexType name="EventDefinition">
           <xs:sequence>
             <xs:element ref="tns:event" minOccurs="1" maxOccurs="33"/>
             <xs:element name="type" type="tns:ItemTypeList"
                minOccurs="0"/>
             <xs:element ref="tns:field"/>
             <xs:element name="containers" type="types:ContainerList"
                 minOccurs="0"/>
         </xs:sequence>
     </xs:complexType>

     <xs:complexType name="EventList">
           <xs:sequence>
             <xs:element name="event" type="tns:Event" minOccurs="0"
                maxOccurs="unbounded"/>
           </xs:sequence>
     </xs:complexType>

      <xs:complexType name="Events">
           <xs:sequence>
             <xs:element ref="tns:key"/>
             <xs:element name="persistence" type="xs:duration"
```

```
            minOccurs="0"/>
          <xs:element name="ipAddress" type="xs:string"
            minOccurs="0"/>
         <xs:element name="port" type="xs:int" minOccurs="0"/>
        <xs:element name="http" type="xs:boolean"
           default="false"/>
        <xs:element name="event" type="tns:EventDefinition"
           maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="enabled" type="xs:boolean" default="false"/>
</xs:complexType>

<xs:simpleType name="EventType">
  <xs:restriction base="xs:string">
        <xs:enumeration value="AddressBookAdd"/>
        <xs:enumeration value="AddressBookDelete"/>
        <xs:enumeration value="AddressBookModify"/>
        <xs:enumeration value="AddressBookItemAdd"/>
        <xs:enumeration value="AddressBookItemDelete"/>
        <xs:enumeration value="AddressBookItemModify"/>
        <xs:enumeration value="FolderAccept"/>
        <xs:enumeration value="FolderAdd"/>
        <xs:enumeration value="FolderDelete"/>
        <xs:enumeration value="FolderItemAdd"/>
        <xs:enumeration value="FolderItemDelete"/>
        <xs:enumeration value="FolderItemMove"/>
        <xs:enumeration value="FolderModify"/>
        <xs:enumeration value="ItemAccept"/>
        <xs:enumeration value="ItemArchive"/>
        <xs:enumeration value="ItemComplete"/>
        <xs:enumeration value="ItemDecline"/>
     <xs:enumeration value="ItemDelete"/>
        <xs:enumeration value="ItemMarkPrivate"/>
        <xs:enumeration value="ItemMarkRead"/>
        <xs:enumeration value="ItemMarkUnread"/>
        <xs:enumeration value="ItemModify"/>
        <xs:enumeration value="ItemPurge"/>
        <xs:enumeration value="ItemUnarchive"/>
        <xs:enumeration value="ItemUnmarkPrivate"/>
        <xs:enumeration value="Login"/>
        <xs:enumeration value="Logout"/>
        <xs:enumeration value="ProxyAccessAdd"/>
        <xs:enumeration value="ProxyAccessModify"/>
        <xs:enumeration value="ProxyAccessDelete"/>
        <xs:enumeration value="ProxyLogin"/>
        <xs:enumeration value="SessionTimedOut"/>
        <xs:enumeration value="TrustedApplicationLogin"/>
     </xs:restriction>
</xs:simpleType>

<xs:simpleType name="Field">
     <xs:restriction base="xs:string">
        <xs:enumeration value="AcceptLevel"/>
        <xs:enumeration value="Attachment"/>
        <xs:enumeration value="Category"/>
```

```
        <xs:enumeration value="Classification"/>
        <xs:enumeration value="DueDate"/>
        <xs:enumeration value="Duration"/>
        <xs:enumeration value="EmailAddress"/>
        <xs:enumeration value="ExpirationDate"/>
        <xs:enumeration value="IMAddress"/>
        <xs:enumeration value="MessageBody"/>
        <xs:enumeration value="Name"/>
        <xs:enumeration value="PersonalSubject"/>
        <xs:enumeration value="PhoneNumber"/>
      <xs:enumeration value="Place"/>
        <xs:enumeration value="Rights"/>
        <xs:enumeration value="Security"/>
        <xs:enumeration value="SendPriority"/>
        <xs:enumeration value="StartDate"/>
        <xs:enumeration value="TaskCategory"/>
        <xs:enumeration value="TaskPriority"/>
     </xs:restriction>
</xs:simpleType>

<xs:simpleType name="FieldList">
   <xs:list fieldType="Field"/>
</xs:simpleType>

<xs:simpleType name="ItemType">
  <xs:restriction base="xs:string">
        <xs:enumeration value="AddressBookItem"/>
        <xs:enumeration value="Appointment"/>
        <xs:enumeration value="CalendarItem"/>
        <xs:enumeration value="Contact"/>
        <xs:enumeration value="Group"/>
        <xs:enumeration value="Mail"/>
        <xs:enumeration value="Note"/>
        <xs:enumeration value="Organization"/>
        <xs:enumeration value="PhoneMessage"/>
        <xs:enumeration value="Resource"/>
        <xs:enumeration value="Task"/>
     </xs:restriction>
</xs:simpleType>

<xs:simpleType name="ItemTypeList">
   <xs:list itemType="ItemType"/>
</xs:simpleType>

<xs:element name="cleanEventConfigurationRequest">
   <xs:complexType>
        <xs:sequence>
          <xs:element name="all" type="xs:boolean" default="false"/>
        </xs:sequence>
     </xs:complexType>
</xs:element>

<xs:element name="cleanEventConfigurationResponse">
  <xs:complexType>
     <xs:sequence>
```

```
 <xs:element name="status" type="types:Status"/>
      </xs:sequence>
     </xs:complexType>
</xs:element>

<xs:element name="configureEventsRequest">
   <xs:complexType>
        <xs:sequence>
           <xs:element name="enabled" type="xs:boolean"/>
             <xs:element name="events" type="tns:Events"
            minOccurs="0"/>
        </xs:sequence>
     </xs:complexType>
</xs:element>

<xs:element name="configureEventsResponse">
   <xs:complexType>
        <xs:sequence>
          <xs:element name="status" type="types:Status"/>
        </xs:sequence>
     </xs:complexType>
</xs:element>

<xs:element name="getEventConfigurationRequest">
  <xs:complexType>
      <xs:sequence>
 <xs:element ref="tns:key" minOccurs="0"/>
  </xs:sequence>
     </xs:complexType>
</xs:element>

<xs:element name="getEventConfigurationResponse">
   <xs:complexType>
         <xs:sequence>
           <xs:element name="enabled" type="xs:boolean"/>
           <xs:element name="events" type="tns:Events"
           minOccurs="0"/>
             <xs:element name="status" type="types:Status"/>
        </xs:sequence>
     </xs:complexType>
</xs:element>

<xs:element name="getEventsRequest">
  <xs:complexType>
        <xs:sequence>
           <xs:element ref="tns:key"/>
             <xs:element name="from" type="xs:dateTime" minOccurs="0"/>
           <xs:element name="until" type="xs:dateTime"
            minOccurs="0"/>
 <xs:element name="count" type="xs:int" default="-1"/>
 <xs:element name="remove" type="xs:boolean"
            default="false"/>
           <xs:element name="notify" type="xs:boolean"
            default="false"/>
          </xs:sequence>
```

```
        </xs:complexType>
    </xs:element>

    <xs:element name="getEventsResponse">
      <xs:complexType>
         <xs:sequence>
               <xs:element name="events" type="types:EventList"/>
               <xs:element name="status" type="types:Status"/>
           </xs:sequence>
      </xs:complexType>
    </xs:element>

    <xs:element name="removeEventConfigurationRequest">
       <xs:complexType>
            <xs:sequence>
              <xs:element ref="tns:key"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="removeEventConfigurationResponse">
       <xs:complexType>
            <xs:sequence>
               <xs:element name="status" type="types:Status"/>
            </xs:sequence>
       </xs:complexType>
    </xs:element>

    <xs:element name="removeEventsRequest">
       <xs:complexType>
            <xs:sequence>
              <xs:element ref="tns:key"/>
              <xs:element name="from" type="xs:dateTime" minOccurs="0"/>
              <xs:element name="until" type="xs:dateTime"
                minOccurs="0"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <xs:element name="removeEventsResponse">
       <xs:complexType>
            <xs:sequence>
              <xs:element name="status" type="types:Status"/>
            </xs:sequence>
       </xs:complexType>
    </xs:element>
</xs:schema>
```