FSA Integration Partner

United States Department of Education
Federal Student Aid

eZ-Audit PEPS Integration Setup and Interval Data

Task Order #116

April 3, 2003

Revision History

Date	Name (Lotus Notes ID)	Description of Revision
04/03/2003	Jennifer C Sri	Initial Creation

Table of Contents

Introduction 1
- Purpose 1
- Scope 1
- Intended audience 1
- Background 1

Overview 1

Design Considerations 1
- Assumptions and Dependencies 2
- General Constraints 2
- Goals and Guidelines 2
- Development Methods 2
- Detailed Design 2

System Architecture 6
- Subsystem Architecture: SMTP Server 6
- SubSystem Architecture: ITA Logging and Exception Handling Framework 6

Detailed System Design 7
- SFASmtpClient JavaBean 7

Interaction Diagrams 8

References 9

Appendices 10
- Jar Files 10

Introduction

EZ-Audit will integrate with PEPS in order to maintain up-to-date information within eZ-Audit. Some financial data will continue to be worked in PEPS, therefore routine pulls of this data are vital to keeping eZ-Audit and PEPS in-synch. PEPS will likewise integrate with eZ-Audit, but this portion of the interface will not be discussed in the document. For further details on the PEPS portion of the interface, please reference the Trading Partner Agreement.

Purpose

This document provides a high-level summary and detailed technical design for the eZ-Audit interface with PEPS for setup and interval data. This setup and interval data consists of submission and institution data, which includes data fields such as fiscal year end, years in zone, and school group number.

Scope

This document covers only the setup and interval data portion of the eZ-Audit interface with PEPS. The other interfaces for school file data and Clearinghouse file data will not be discussed in this document.

Intended audience

This document is intended for eZ-Audit application programmers who need to understand the java programs written to import setup and interval data from PEPS.

Background

In the past, PEPS has been the sole owner and distributor of school and submission data. With the advent of eZ-Audit, some data is still owned by PEPS and consequently eZ-Audit requires setup and interval data from PEPS.

Overview

As eZ-Audit comes into existence, it will handle incoming annual submissions, fiscal year end changes, case team management, screening of submissions, change in ownership submissions, assignment of financial statements and audits, and resolution of these submissions. Although eZ-Audit will take over and create many different areas for schools and case team workers, it will not be able to handle every piece of the PEPS system. PEPS is used not only for financial data, but for many other areas and by many other users. Therefore, eZ-Audit must assume that data will still continue to be entered/updated in PEPS, and eZ-Audit will be responsible for getting this data at the time of “go-live” (named setup data) and as updates are made daily (called interval data).

Design Considerations

Assumptions and Dependencies

It is assumed that eZ-Audit will write a script and java program to be called daily. This script should be scheduled to run nightly. It is also assumed that the java program will utilize ITA's email and logging components to log errors/information and email these logs to operations team members.

General Constraints

The Email Framework will require that a Simple Mail Transport Protocol Server (SMTP) is available so that emails generated by the client application may route mail to it. Currently ITA uses ***sendmail*** that is available on Sun Solaris.

Goals and Guidelines

The goal of this development is to create a java program to grab setup and interval data from PEPS. Setup data will include all 2002 submission-level data currently submitted in PEPS, school group information, years in zone, consecutive years in zone, fiscal year end dates, audit firms, financial statement and audit TINs, and waivers/exemption information. Interval data will call the same methods except it will only retrieve 2002 and greater submissions that have been updated since the interval data program was last run. Also, waiver and exemption information will only be retrieved during setup. A manual business process will manage new or updated waivers and exemptions.

Development Methods

The java program and script were written independent of the eZ-Audit application and will be executed independent of the system. The java program was written and compiled using JDK 1.2.2, the JDK version available on the application server.

Detailed Design

Several java methods were written for each section of data needed from PEPS. In each method, a select statement was executed to retrieve the PEPS data, insert/update statement(s) were called to enter this information into eZ-Audit appropriately, logs containing errors and other pieces of information are created, a record is created in the interface info table to show that the method ran, and an email is sent containing the log files. A central java program (controlClass.java) has been written to call each of these individual methods. The following is the list of java programs and the sql statements used in each one:

- AuditFirm.java
- IntervalData.java

- SchoolGroup.java
- SetupData.java
- TinsAuditFS.java
- Waiver.java
- ZoneAndFYE.java

Audit Firm Data – auditFirm.java

```
select distinct FS.FISCAL_YR_END_DT, AR.AUDIT_REG_CD, AR.AUDIT_YR,
        AR.AUDIT_FISCAL_YR_RCVD_NBR, AR.AUDIT_SEQ_NBR,
        AR.AUDIT_RCVD_DT, FS.FIN_STMT_RCVD_DT,
        FS.LAST_UPDT_DT, AR.LAST_UPDT_DT,
        FS.FS_SYS_ID, AR.AUDIT_RPT_SYS_ID
from FINANCIAL_STATEMENT FS, AUDIT_REPORT AR,
        FS_PARTICIPANT FP, AUDIT_PARTICIPANT AP
where FP.SCH_NBR = ? and FP.SCH_NBR = AP.SCH_NBR
        and FS.FISCAL_YR_END_DT = AR.AUDIT_PER_END_DT
        and FS.FISCAL_YR_END_DT > to_date('12/31/2001', 'MM/DD/YYYY')
        and FS.FS_SYS_ID not like '06%'
        and FS.FIN_STMT_RCVD_DT is not null
        and AR.AUDIT_RCVD_DT is not null
        and FS.FS_SYS_ID = FP.FS_SYS_ID
        and AR.AUDIT_RPT_SYS_ID = AP.AUDIT_RPT_SYS_ID

insert into submission (submission_id, ope_id, creator_id,
        status, fiscal_yr, type, submission_date, acn, fac_acn,
        fsa_receipt_date, last_mod_time, peps_flag)
values (?,?,?,?,?,?,?,?,?,?,?,?)

update audit_firm
set name = ?, address_1 = ?, address_2 = ?, city = ?, state = ?,
        province = ?, country = ?, postal_code = ?, phone_number = ?, phone_number_ext = ?,
        fax_number = ?, contact_email_address = ?, last_mod_time = ?
where tin = ?
```

Interval Submission Data – intervalData.java

```
select FS.FISCAL_YR_END_DT, AR.AUDIT_REG_CD, AR.AUDIT_YR,
        AR.AUDIT_FISCAL_YR_RCVD_NBR, AR.AUDIT_SEQ_NBR,
        AR.AUDIT_RCVD_DT, FS.FIN_STMT_RCVD_DT, FS.LAST_UPDT_DT, AR.LAST_UPDT_DT,
        FS.FS_SYS_ID, AR.AUDIT_RPT_SYS_ID
from FINANCIAL_STATEMENT FS, AUDIT_REPORT AR,
        FS_PARTICIPANT FP, AUDIT_PARTICIPANT AP
where FP.SCH_NBR = ? and FP.SCH_NBR = AP.SCH_NBR
        and FS.FISCAL_YR_END_DT = AR.AUDIT_PER_END_DT
        and FS.FISCAL_YR_END_DT > to_date('12/31/2001', 'MM/DD/YYYY')
        and FS.FS_SYS_ID not like '06%'
        and FS.FIN_STMT_RCVD_DT is not null
        and AR.AUDIT_RCVD_DT is not null
        and FS.FS_SYS_ID = FP.FS_SYS_ID
        and AR.AUDIT_RPT_SYS_ID = AP.AUDIT_RPT_SYS_ID
```

```
        and (FS.LAST_UPDT_DT > ?
        or AR.LAST_UPDT_DT > ?)

insert into submission (submission_id, ope_id, creator_id,
        status, fiscal_yr, type, submission_date, acn, fac_acn,
        fsa_receipt_date, last_mod_time, peps_flag)
values (?,?,?,?,?,?,?,?,?,?,?,?)

update submission
set acn = ?, fac_acn = ?, last_mod_time = ?
where ope_id = ? and fiscal_yr = ? and peps_flag = 'Y'
```

School Group Data – schoolGroup.java

```
select sch_nbr, sch_group_nbr, end_dt
from school_group_member
order by sch_group_nbr, sch_nbr

update institution
set group_num = ?, ope_id_parent = ?
where ope_id = ?
```

Setup Submission Data – setupData.java

```
select distinct FS.FISCAL_YR_END_DT, AR.AUDIT_REG_CD, AR.AUDIT_YR,
        AR.AUDIT_FISCAL_YR_RCVD_NBR, AR.AUDIT_SEQ_NBR,
        AR.AUDIT_RCVD_DT, FS.FIN_STMT_RCVD_DT,
        FS.LAST_UPDT_DT, AR.LAST_UPDT_DT,
        FS.FS_SYS_ID, AR.AUDIT_RPT_SYS_ID
from FINANCIAL_STATEMENT FS, AUDIT_REPORT AR,
        FS_PARTICIPANT FP, AUDIT_PARTICIPANT AP
where FP.SCH_NBR = ? and FP.SCH_NBR = AP.SCH_NBR
        and FS.FISCAL_YR_END_DT = AR.AUDIT_PER_END_DT
        and FS.FISCAL_YR_END_DT > to_date('12/31/2001', 'MM/DD/YYYY')
        and FS.FS_SYS_ID not like '06%'
        and FS.FIN_STMT_RCVD_DT is not null
        and AR.AUDIT_RCVD_DT is not null
        and FS.FS_SYS_ID = FP.FS_SYS_ID
        and AR.AUDIT_RPT_SYS_ID = AP.AUDIT_RPT_SYS_ID

insert into submission (submission_id, ope_id, creator_id,
        status, fiscal_yr, type, submission_date, acn, fac_acn,
      fsa_receipt_date, last_mod_time, peps_flag)
values (?,?,?,?,?,?,?,?,?,?,?,?)
```

FS and Audit TINs – tinsAuditFS.java

```
select cpa.irs_nbr
from financial_statement fs, cpa_office cpa, fs_participant fp
where fp.sch_nbr = ?
        and fs.cpa_ofc_sys_id = cpa.cpa_ofc_sys_id
        and fs.fs_sys_id = fp.fs_sys_id
order by fiscal_yr_end_dt desc

select cpa.irs_nbr
from audit_report ar, cpa_office cpa, audit_participant ap
where ap.sch_nbr = ?
        and ar.cpa_ofc_sys_id = cpa.cpa_ofc_sys_id
        and ar.audit_rpt_sys_id = ap.audit_rpt_sys_id
order by audit_per_end_dt desc

update institution
set fs_tin = ?, fa_tin = ?
where ope_id = ?
```

Waiver and Exemption Data – waiver.java

```
select sch_nbr, waiver_fy1_dt, waiver_fy2_dt, waiver_fy3_dt,
        last_updt_dt, issue_dt
from fs_audit_waiver
where denial_dt is null
        and expire_dt is null
        and withdraw_dt is null
        and issue_dt is not null

insert into submission (submission_id, ope_id, creator_id,
        status, type, last_mod_time, peps_flag, submission_date)
values (?,?,?,?,?,?,?,?)

insert into institution_exempt_waiver
values(?,?,?,?,?)
```

Zone and Fiscal Year End Data – zoneAndFYE.java

```
select fs.fiscal_yr_end_dt, fs.conclusion_cd
from financial_statement fs, fs_participant fp
where fs.fs_sys_id = fp.fs_sys_id
        and fp.sch_nbr = ?
order by fiscal_yr_end_dt desc

update institution
set yrs_in_zone = ?, consecutive_yrs_in_zone = ?
where ope_id = ?
```

```
update institution
set fiscal_yr_end = ?
where ope_id = ?
        and (fiscal_yr_end is null OR fiscal_yr_end < ?)
```

System Architecture

The ITA Email Framework provides the ability to create, manipulate, and send Emails from any Java application running within a J2EE Application Server. Developers can use the simplified Email wrappers to read or send email or extend the JavaMail API's themselves for some very specific Email features.

Subsystem Architecture: SMTP Server

The ITA Email Framework requires that the framework have access to a Simple Mail Transport Protocol Server (SMTP). SMTP is the protocol that most Internet vendors implement to send emails across the Internet. The SMTP server is the actually workhorse that the Email Framework will connect to and forward emails that applications produce. Once the SMTP server has received the Email, it will connect to its forwarding partner (Another SMTP Server) and forward the Email out to the Internet. Currently the Email framework uses the SMTP Server that exists on the WebSphere machine that is provided via Solaris 2.6.

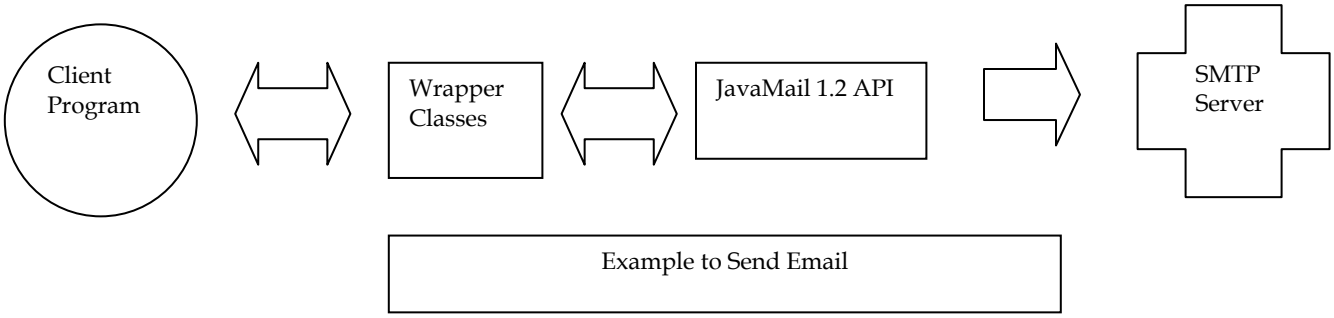
SubSystem Architecture: ITA Logging and Exception Handling Framework

The ITA Logging and Exception handling frameworks has been added to the Email framework to enhance debugging and tracing abilities. This should benefit operations ability to isolate problems that may occur within the framework. The Logging XML configuration document should be called rcs.xml and located in the home directory of the user making the java calls.

Detailed System Design

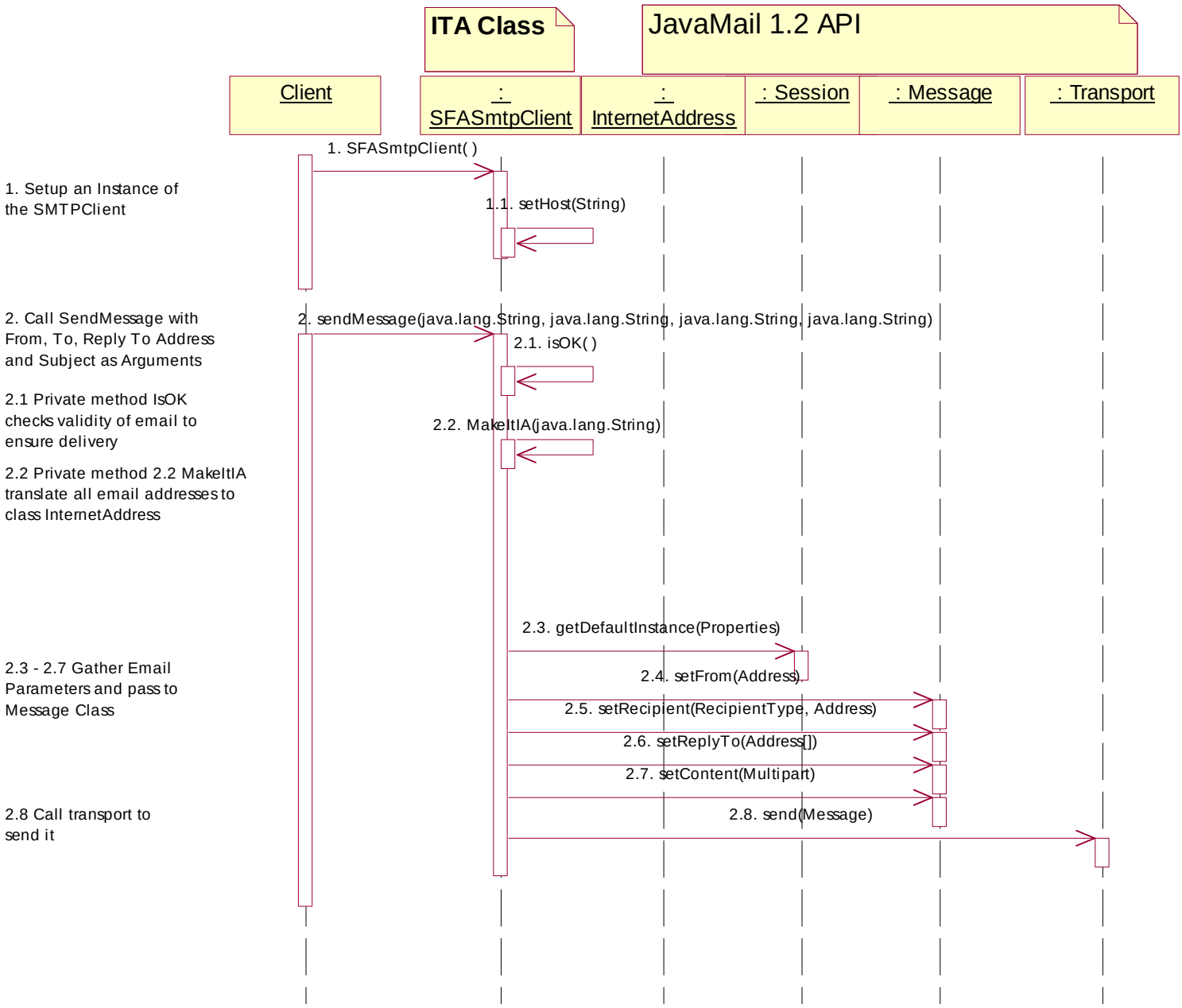
SFASmtpClient JavaBean

The ITA Email Framework includes a wrapper class that encapsulates the JavaMail API. This wrapper provide a simplified interface to the Sun JavaMail API and allow easy creation and manipulation of email traffic. The SFASmtpclient JavaBean is aimed at supporting Simple Mail Transport Protocol (SMTP), which is the predominate protocol for sending email. As specified in the Usage example a program needs to instantiate an instance of SFASmtpClient and then using the public methods build and manipulate and send the email.



Interaction Diagrams

This sequence illustrates the interaction between a client object and SFASmtpClient class to build and send a email to a SMTP Server



References

Java Activation Framework JavaDocs online
http://java.sun.com/products/javabeans/glasgow/javadocs
JavaMail 1.2 JavaDocs Online
http://www.javasoft.com/products/javamail/1.2/docs/javadocs/index.html
Jnet JavaBean Store
http://www.java-shop.com/jnet.htm

Appendices

Jar Files

The following Jars files are part of the ITA Email Framework.

Mail Jar Files
Activation.jar
Mail.jar
MailApi.jar
Rcs_Email_01.jar
Pop3.jar
Smtp.jar

Logging Jar Files
Jakarta-oro-2.0.1
Jdom-B6
Protomatter-1-1-5.jar
Utility.jar
Xerces.jar
Xml.jar

Logging XML Document
Rcs.xml