SOA在金融行业的应用

随需应变的业务

前　言

变化是硬道理，你的企业能有效响应吗？在过去的十年，中国有很多公司，因跟不上市场迅速发展变化， 导致倒闭。同时期， 也有很多小企业，成为国内知名品牌。

IBM 业务咨询服务部在 2004 年的全球总裁调查报告总结，“企业的总裁们认为他们需要更有效的方法对不断变化的市场环境及风险做出判断，分析和响应”。同时，他们也意识到落后的资讯科技会带来业务的落后。在 2002 年底，Gartner Group 预测到 2008 年，SOA(面向服务系统架构)将成为占有绝对优势的软件工程技术，主流企业现在就应该了解和应用 SOA 技术。

SOA 的使用是基于重用的工能单元称为服务，通过运用这些服务之间定义良好的接口和契约联系来支持业务流程，使服务(业务)的编排和组合增加了灵活性和集成性的重要作用。SOA 关键是使用标准的服务接口和定义用松耦合连接。从而掩盖了 IT 环境底层的技术复杂性和繁琐性。

SOA 观念不是新的，今天的 SOA 与过去不同是基于已广泛接受的 Web 服务标准，从而提供了在每个不同厂商解决方案之间的相互性。

SOA 是从企业的需求开始，把 IT 系统和商业流程连合在一起，以服务集成形式实现新的而又灵活的应用功能。SOA 简化了 IT，让 IT 变得更有弹性， 以便更好地发展和优化业务流程，从而促进企业与合作伙伴的业务需要，也使供应商和客户之间运作流程的端到端整合，让企业可以快速灵敏地响应客户和市场不断变化的需求，实现随需应变企业(On Demand Business)。

实践 SOA 需要四方面的工作：企业组织及服务监控(governance)，以服务为定义的业务流程(process)，IT 系统和商业流程的配合(align IT with business)，企业建模及系统架构(architecture)。企业可以利用现有资源(如 EAI)，做一层封装对外提供服务接口，从而把系统迅速创建为服务单元。根据研究报告，适当使用 SOA 能降低开发和集成的成本，增加资源方面的使用效率。而与此同时可加强系统安全性，减轻维修工作量，减少潜在风险，管理和监视费用。

SOA 技术的成熟，为建设灵活，基于标准的 IT 基础设施提供了明确的答案。企业如何才能有效实现 SOA 策略？这是很大的挑战和充满了潜在的障碍。许多公司会有实践问题。我们建议企业不需要马上全部采用或大量修改现有的 IT 基础， 使用服务来完全代替它们。而应当优先选择利用服务接口解决最紧迫的集成问题，然后逐步扩展范围，封装出更多面向服务的业务流程，稳定的把公司转型为随需应变企业。

Process Choreographer(流程排演)技术

在日常活动中，我们经常为了一个目的，重复执行一系列的活动。这些可以明确分割、重复执行的活动序列通常被定义成一个独立的过程模型(process)。

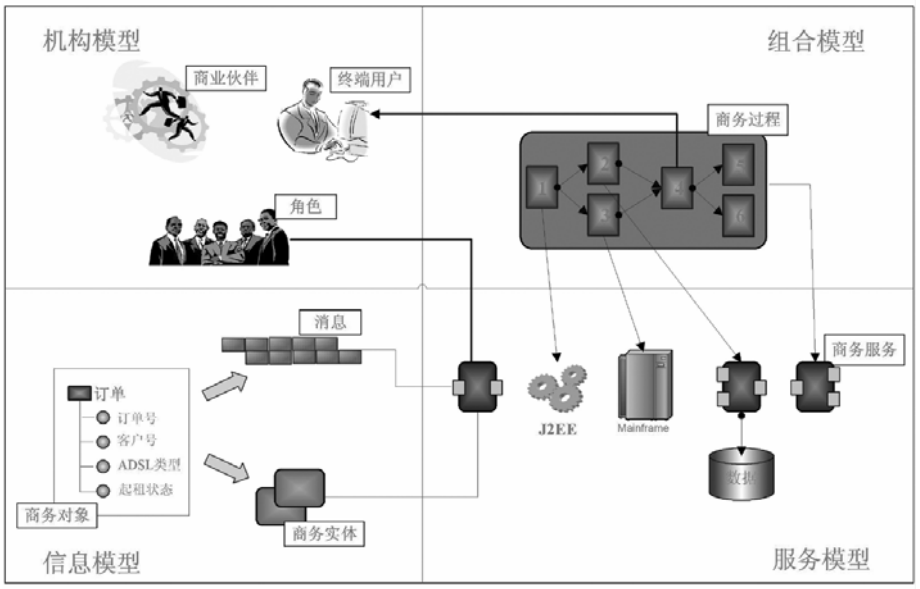
因此，一个商务过程是指一套以特定的顺序被调用来实现某个商务目标的商务活动流。商务过程定义了流的顺序、如何处理外部事件、如何与人交互和条件判断。如下图所示，一个商务过程主要由三个部分构成:

- 过程数据(信息模型)
  - 消息: 服务之间交互的请求/响应信息。
  - 商务实体和对象: 定义消息成员的类型，包含数据的组成和处理方法。
- 活动和服务(服务模型)
  - 商务服务: 商务过程的每个活动结点都对应一个服务执行步，这些服务可以是企 业内部的 J2EE 服务，Mainframe 访问服务，或数据库访问服务，也可以是企业的商务伙伴提供的外部应用服务。
- 过程参与者(机构模型)
  - 商业伙伴: 具有某种商务角色的服务提供者。在这里，我们将商务过程中的所有 服务提供者都称作商业伙伴。例如企业内部的帐务系统，CICS 系统，零售系统的企业外部供应商系统等。商务过程与服务提供者的交互选用 Web Services 的标准接口。
  - 终端用户: 以特定角色参与商务过程的人员，例如审计员，施工人员，管理员等。
  - 角色: 商业伙伴具有唯一的商务角色; 而终端用户的角色，可以对人员操作进行授权和管控。

将信息模型，服务模型和机构模型组合在一起，就构成了过程模型(即组合模型)。

例如，下图中商务过程一共调用了 6 个活动，活动 1(JavaTM 类或 EJB)的输出消息连接到活动 2(数据库访问服务)和活动 3(CICS 应用)，活动 1 结束后，活动 2 和活动 3 可以并行执行，也可以只执行一个，根据连接条件而定。过程流将在活动 4 的位置等待人员(终端用户)的“决定”，然后依据“决定”执行活动 5 或活动 6。

总之，在一个过程引擎中执行的商务过程，可以访问 SOA 架构的应用服务器上的所有商务应用。这可以有效地将商务流逻辑同每个具体的服务功能的实现过程相分离，并带来两个好处：一、显然，商务服务不需将所有内部的决策过程和数据管理模式都暴露给商业伙伴；二、商务服务内部的实现过程可以随意修改，而不会影响商务过程流的外部执行。



目前实现这种商务过程的标准就是 Business Process Execution Language(BPEL)，它是一种描述商务过程和商务交互方式的语言，扩展了 Web Services 的交互模型，使之能够支持商务交易，并且将 Web Service 接口层的关系结构封装在 partner link 中。总之，BPEL 定义了一种商务集成模式，提供了更高程度的企业内部和企业之间的 B2B 的业务整合的自动化。

如何构建，配置和运行这些基于 SOA 的应用和商务过程，IBM 推出了 WebSphere Business Integration Modeler V5.1 (WBI Modeler V5.1)、 WebSphere Business Integration Server Foundation V5.1 (WBI-SF V5.1) 和 WebSphere Studio Application Integration Edition V5.1 (WSAD-IE V5.1)。WebSphere Business Integration Server 4.2 用于商业建模的 WBI Modeler、用于开发的 WSAD-IE 和用于运行、监控商务过程的 WBI-SF V5.1 WBI Server 4.2 相配合，形成了下一代的商务集成平台，使现存的 IT 资产的扩展和整合最优化。

WebSphere Business Integration Server Foundation V5.1

WBI-SF V5.1 建立在 WebSphere Application Server(WAS)的基础上，提供 Java 2 平台,J2EE 和基于 Web Services 技术的应用平台,可以为随需应变的动态 e-business 配置企业级 Web Services 解决方案。

它代表了 IBM 构建和配置 SOA 应用的趋势。支持可重用服务的创建，这些服务可以是新的也可以基于那些现存的服务、后台系统、Java 资产和封装的应用构建的。服务可以随意组合形成混合应用和商务过程，从而可以更多地利用商务规则来使这些应用和商务过程具有很好的可用性和可适应性。

支持的运行平台也有很多:

_ Windows 2000， 2003

_ Linux (不同的版本)

_ HP-UX

_ AIX

WebSphere Studio Application Integration Edition V5.1

WebSphere Studio Application Integration Edition V5.1 (WSAD-IE V5.1)为开发和测试基于 BPEL 的商务过程提供了必要的支持，特别是通过它的商务集成工具。这些商务集成工具主要用于支持 SOA，通过将所有的对象、组件、应用和过程都表示成开放标准的服务(services)来减少开发的复杂度。这些服务都是自包含、自描述的标准组件应用，可以跨 Internet 发布，定位和调用，帮助实现最广泛的商务功能的自动化。基于这种架构，开发者无须关注服务的底层实现就可以与所有的软件组件进行交互。
WSADIE 提供了图形化的过程编辑器和调试器，可以帮助用户快速的构建 J2EE 的资源、Web Service 和 BPEL 商务过程，并且一个活动一个活动的调试过程流。

目前支持的平台:

_ Windows XP

_ Windows 2000

_ Windows NT

_ Linux (不同的版本)

WBI-SF 进行企业业务整合的技术特点

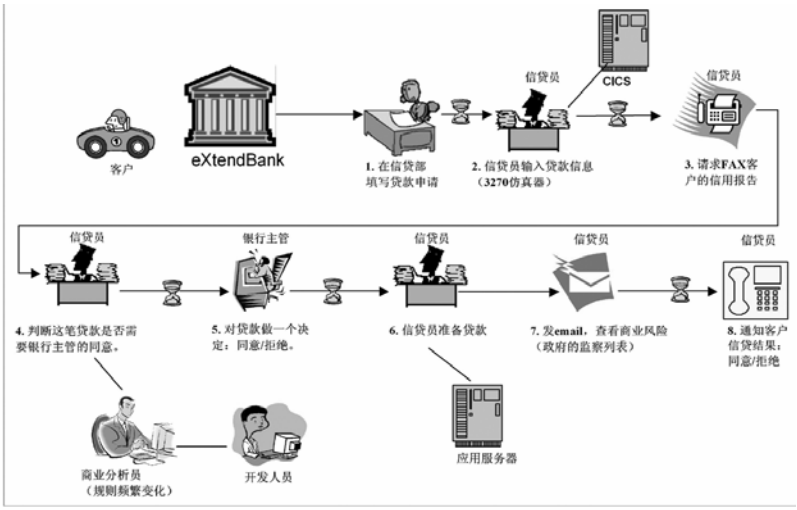
- 利用 IBM 现有的商务集成技术来实现企业内部应用之间、企业内部应用与外部应用之间、商务过程与业务人员之间的完全整合:
- 基于 SOA，构建开放体系结构的整合系统。
- 商务过程的建模(WBI Modeler): 提供给业务主管使用，从商务层面设计商务流程。
- 构建并配置商务过程(WSAD-IE): 供技术开发人员使用，可以将 WBI Modeler 建好的过程模型移植到 BPEL 编辑器，并配置好 BPEL 流程，生成可执行文件和代码。
- 执行并监控商务过程(WBI-SF): 作为应用服务器来运行配置好的 BPEL 商务过程，并可以对商务过程进行实时监控。
- 企业服务总线 (Messaging - WSIF): 应用之间、商务过程与应用之间、人与商务过程之间的通信都通过 WSIF 来实现。
- 对业务人员的权限和角色控制，并可以与企业内部的用户管理系统相联接，共同进行角色的安全控管。

SOA 与 eXtend Bank 业务的结合

由于 BPEL 是 WSAD-IE V5.1 平台上用于开发商务过程和集成解决方案的主要语言，我们就进一步的探究它如何在银行领域的实际业务实例中应用。

现有贷款申请系统面临的挑战

如下图所示，一幅现有的银行贷款申请流程图。



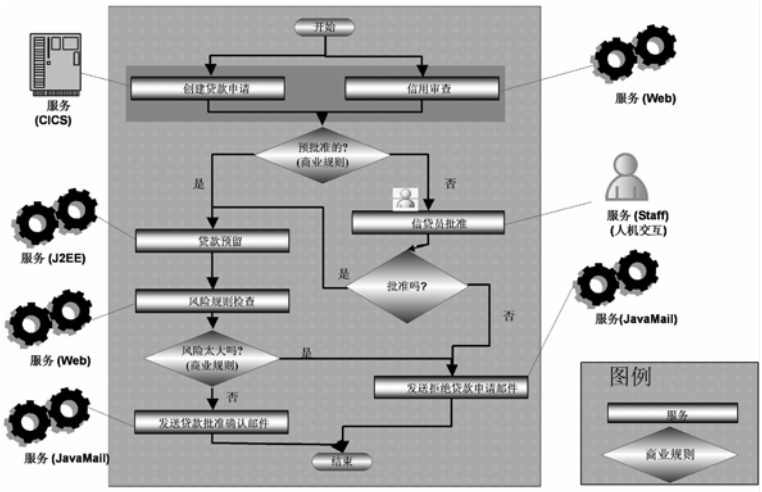
可以看出，旧的申请流程存在很多缺陷:

- 处理贷款申请的时间很长。
- 在处理贷款申请过程中，基于书面的人工参与出错的几率很大，比如电话的意思传递错误等。
- 信贷员需要熟悉和使用大量的技术，例如，后端 CICS 系统、应用服务器、EMAIL 应用。
- 使人、过程和信息一体化很困难。
- 如果整个过程中任何一个环节出了差错，需要人工来恢复已经做的工作，比如从应用服务器恢复已经“准备的贷款”，将申请信息从数据库中删除等。
- 对商务规则变化的快速响应很困难。

利用 IBM 的 Server Foundation 商务集成技术构建新的商务流程

为了去除上述缺陷，实现一个准确，快速，安全的银行贷款申请系统，我们完全采用商务过程(BPEL)来实现整套贷款申请业务流程。

构建出来的新的业务流程(QuickLoan)基于虚拟的网上银行贷款申请系统，此应用充分利用了 WSAD-IE V5.1 和 WBI-SF V5.1 的 Process Choreography 的特性，允许用户 ONLINE 申请贷款，如下图所示。



上图中的商务过程的逻辑如下:

1. 现存的 eXtendBank 客户登陆并输入贷款金额。

2. 商务过程通过 CICS 建立贷款申请并执行信用检查(这里，贷款申请创建活动和信用　　检查活动并行运行)。这里 WBI-SF 利用 BI(Business Integration) Adapters (CICS/ TXSeries)与 CICS 系统进行数据交换。

3. 商务规则决定贷款是否是“预批准的”。

4. 如果贷款不是“预批准的”，则贷款申请必须到信贷员处接受批准。

5. 如果信贷员拒绝这笔贷款申请，则一封拒信会发到客户的邮件里，并且 QuickLoan 过程中止。

6. 如果贷款申请是“预批准的”或者信贷员同意贷款，则资金被“预留”，并建立资金预留记录。

7. 一旦资金被“预留”，商务过程开始检查贷款的风险。

8. 如果风险太大，一封拒信发到客户邮件里。

9. 如果风险不大，确认贷款成功的邮件会发到客户邮件里。

如果在处理过程中贷款被拒绝(由信贷员拒绝或者因贷款风险太高)，“COMPENSATION”(“补偿”)被触发，自动恢复所有设置了“COMPENSATION”的活动到处理前的状态。在本应用中，只有“创建贷款申请”和“贷款预留”两个活动需要失败后补偿。

对比旧的银行贷款申请系统,在 WBI-SF 的开发、配置和运行平台上构建的基于 BPEL、Web Service 和 J2EE 的商务过程具有明显的优势:

- QuickLoan 的自动化商务过程减少了贷款申请的时间:
  - 人工操作可以被自动化的服务取代。
  - 任务可以并行执行。
  - 人、过程和信息可以很容易的被整合在一起:
  - CICS、J2EE 应用、PARTNER、邮件系统等。
  - 人员的角色可以进行安全控管。
- 过程中的失败可以很容易的被“undone”。
- 可以对商务规则的变化进行快速的响应。
- 面向服务的体系结构(SOA)允许即插即用的能力。

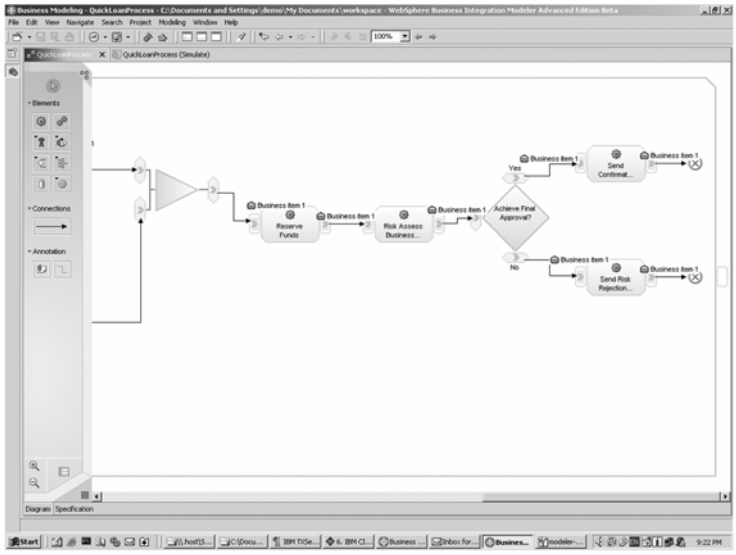
- QuickLoan 的活动结点可以很容易的集成到其他商务过程中。
- QuickLoan 的商务过程也可以很容易的集成到其他商务过程中。

下面我们来验证一下基于 WBI-SF 构建的银行贷款申请流程的执行能力(我们在系统

中验证三个用户的申请过程，包括申请成功的用户、风险高被拒绝的用户和信审员拒绝的用户)。

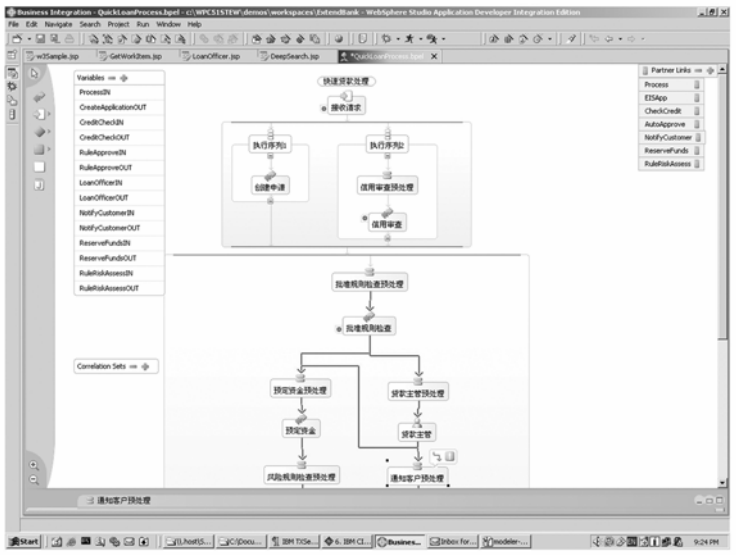
使用WBI MODELER建立业务模型

业务人员使用 WBI MODELER 从业务层面对贷款申请的商务流程进行设计，如下图所示:



将WBI Modeler建立的业务模型移植到WSAD-IE V5.1 中

将业务模型在 WSAD-IE V5.1 中转成 BPEL，如下图所示:

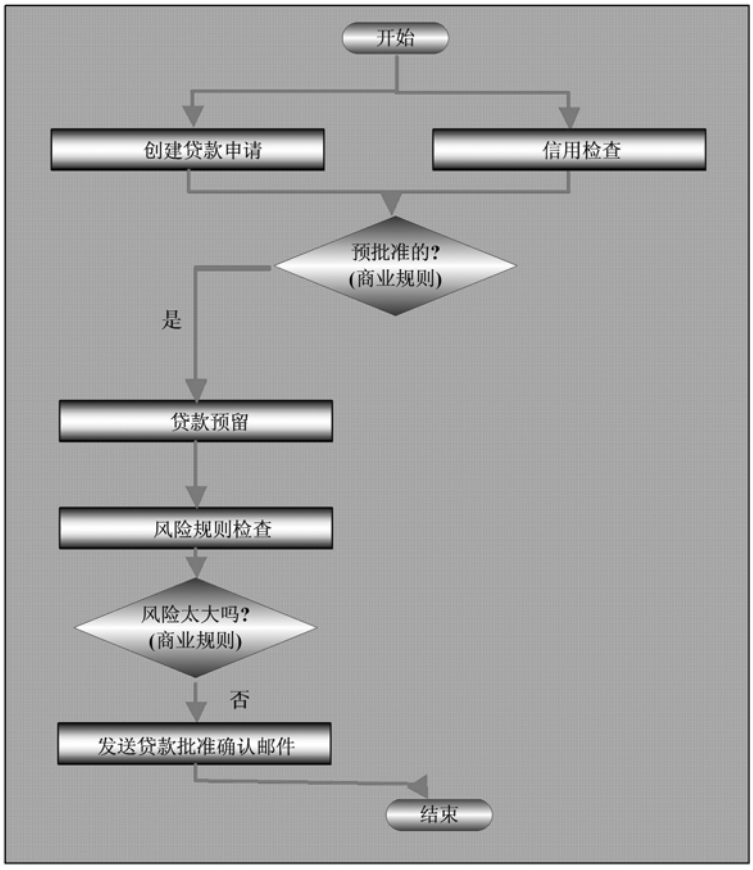


对此 BPEL 进行配置，生成可以在 WBI SF 测试服务器上运行的代码和文件。

在WBI SF的测试服务器上验证商务过程(BPEL)的执行

正常的流程(richly@demo.com)

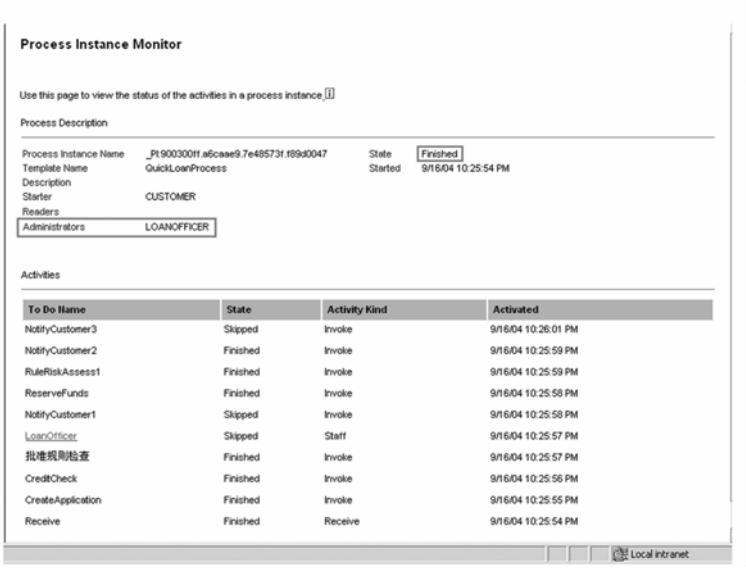
信用度为“A”，贷款金额 75000，贷款是“预批准的”，风险规则检查返回为“true”，申请成功。如下图所示:



首先，客户 Richly 在网上作贷款申请，如下图所示:

Richly 马上就可以收到申请被处理的信息，如下图，之后等待确认的 email。

执行过程中，可以通过管理员的监控窗口来实时监控商务过程的执行情况，如下图所示，管理员可以清晰地看到那些活动执行了，那些活动被跳过了，执行时间和状态等。



Process Instance Monitor

Use this page to view the status of the activities in a process instance.

Process Description

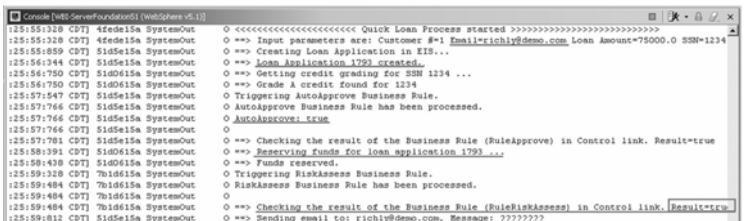
Process Instance Name	_Pt:900300ff.a6caae9.7e48573f.f89d0047	State	Finished
Template Name	QuickLoanProcess	Started	9/16/04 10:25:54 PM
Description			
Starter	CUSTOMER		
Readers			
Administrators	LOANOFFICER		

Activities

To Do Name	State	Activity Kind	Activated
NotifyCustomer3	Skipped	Invoke	9/16/04 10:26:01 PM
NotifyCustomer2	Finished	Invoke	9/16/04 10:25:59 PM
RuleRiskAssess1	Finished	Invoke	9/16/04 10:25:59 PM
ReserveFunds	Finished	Invoke	9/16/04 10:25:58 PM
NotifyCustomer1	Skipped	Invoke	9/16/04 10:25:58 PM
LoanOfficer	Skipped	Staff	9/16/04 10:25:57 PM
批准规则检查	Finished	Invoke	9/16/04 10:25:57 PM
CreditCheck	Finished	Invoke	9/16/04 10:25:56 PM
CreateApplication	Finished	Invoke	9/16/04 10:25:55 PM
Receive	Finished	Receive	9/16/04 10:25:54 PM

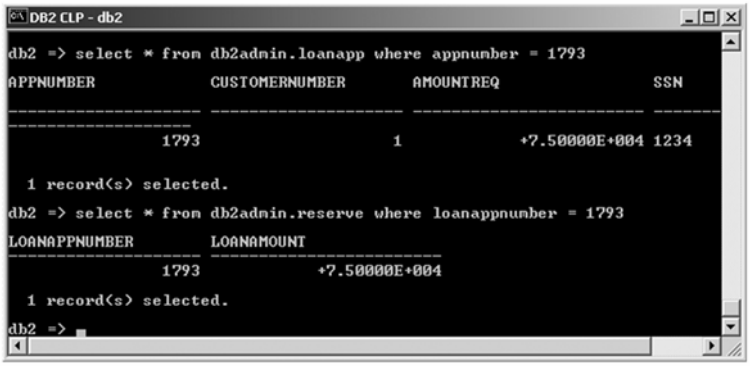
Local intranet

处理完毕后，测试服务器的控制台中显示如下图的执行过程：代号为 1793 的贷款申请和资金预留信息被创建，”AutoApprove:true”表示“预批准”，并且有确认信发出。



```
Console [WBI-ServerFoundation51 (WebSphere v5.1)]
:25:55:328 CDT] 4fede15a SystemOut     O <<<<<<<<<<<<<<<<<<<<<<<< Quick Loan Process started >>>>>>>>>>>>>>>>>>>>>>>>
:25:55:328 CDT] 4fede15a SystemOut     O ==> Input parameters are: Customer #=1 Email=richly@demo.com Loan Amount=75000.0 SSN=1234
:25:55:859 CDT] 51d5e15a SystemOut     O ==> Creating Loan Application in EIS...
:25:56:344 CDT] 51d5e15a SystemOut     O ==> Loan Application 1793 created.
:25:56:750 CDT] 51d0615a SystemOut     O ==> Getting credit grading for SSN 1234 ...
:25:56:750 CDT] 51d0615a SystemOut     O ==> Grade A credit found for 1234
:25:57:547 CDT] 51d5e15a SystemOut     O Triggering AutoApprove Business Rule.
:25:57:766 CDT] 51d5e15a SystemOut     O AutoApprove Business Rule has been processed.
:25:57:766 CDT] 51d5e15a SystemOut     O AutoApprove: true
:25:57:766 CDT] 51d5e15a SystemOut     O
:25:57:781 CDT] 51d5e15a SystemOut     O ==> Checking the result of the Business Rule (RuleApprove) in Control link. Result=true
:25:58:391 CDT] 51d0615a SystemOut     O ==> Reserving funds for loan application 1793 ...
:25:58:438 CDT] 51d0615a SystemOut     O ==> Funds reserved.
:25:59:328 CDT] 7b1d615a SystemOut     O Triggering RiskAssess Business Rule.
:25:59:484 CDT] 7b1d615a SystemOut     O RiskAssess Business Rule has been processed.
:25:59:484 CDT] 7b1d615a SystemOut     O
:25:59:484 CDT] 7b1d615a SystemOut     O ==> Checking the result of the Business Rule (RuleRiskAssess) in Control link. Result=tru
:25:59:812 CDT] 51d5e15a SystemOut     O ==> Sending email to: richly@demo.com. Message: ????????
```

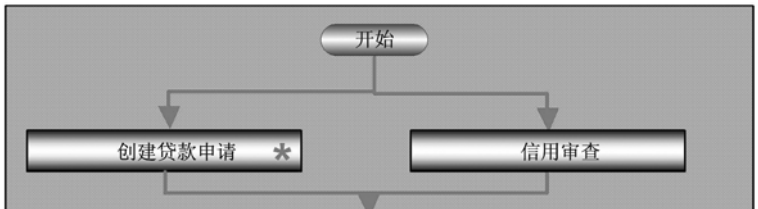
执行完后，发现 CICS 已经在后台的数据库表 LOANAPP 中创建了贷款申请，而 J2EE 的应用服务器也已经将“贷款预留”信息写到数据库的 RESERVE 表中，如下图所示。



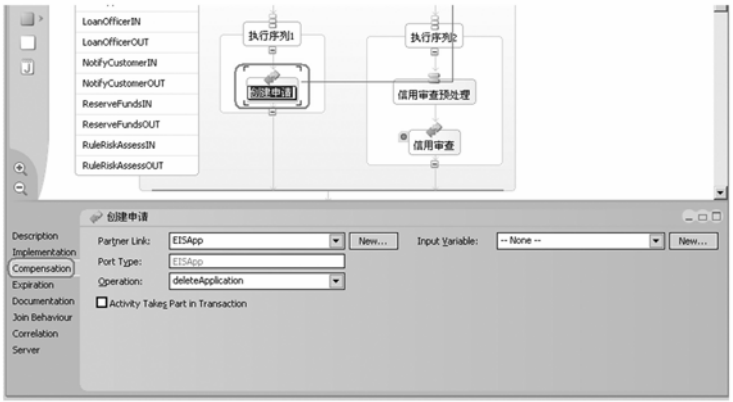
而客户 Richly 在商务过程执行完后，也会收到一封系统自动发出的确认信，如下图。

需要 Compensation（补偿）的流程(poorman@demo.com)

信用度为“B”，是“预批准的”，因风险高被拒绝，需要将已经做完的工作恢复到贷款申请前的状态，即对“创建贷款申请”和“贷款预留”两个活动执行补偿。

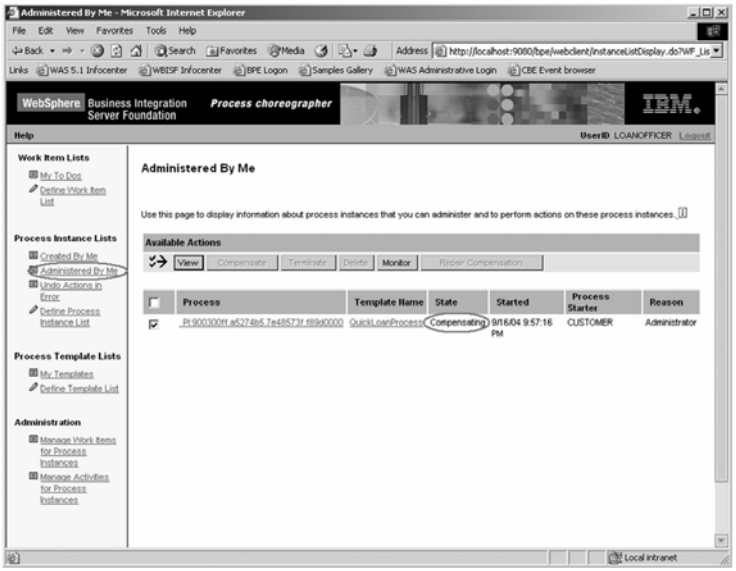


PEL 中需要指定做“补偿”的活动，如“创建申请”活动，并给出“补偿”的方法调用，如下图所示。同样“资金预留”活动也要做同样的“补偿”定义，这样过程失败后，这两个活动就可以恢复到原始状态。

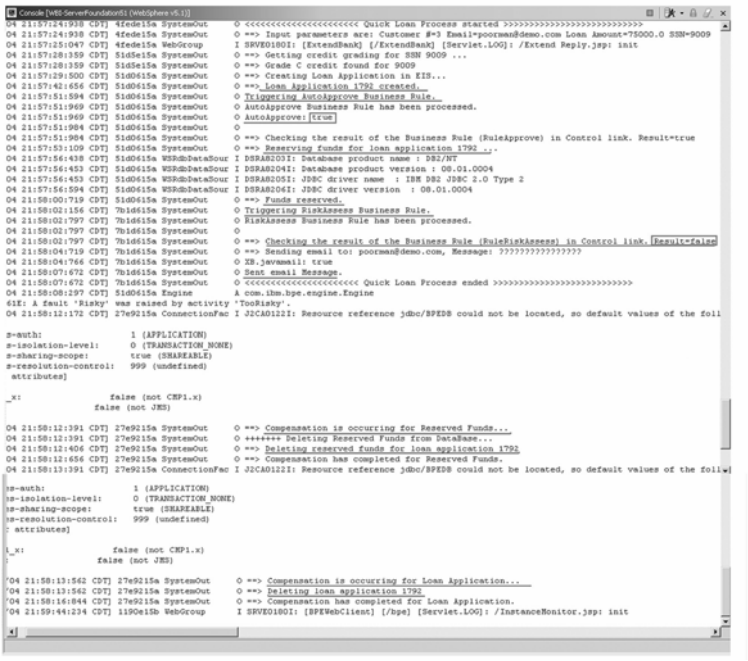


首先，Poorman 也要在网上提交贷款申请，如右图所示:

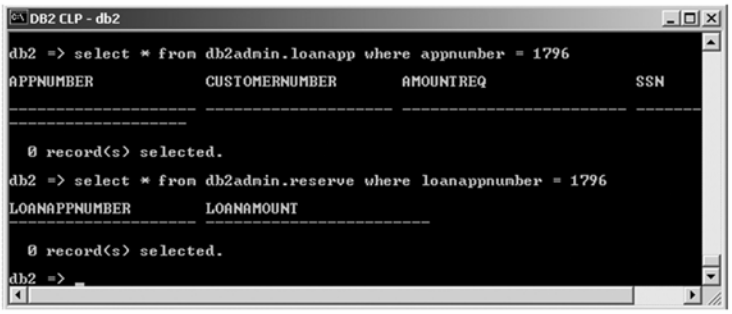
管理员可以在“进程监控窗口”中看到正在执行和已经执行完的活动，如右图所示，管理员可以查看正在“补偿”的商务过程。



过程执行完毕，测试服务器的控制台中显示如右图的执行过程：代号为 1792 的贷款申请和资金预留信息在过程执行的开始被创建，但是由于风险高，被“补偿”操作删除，并且有拒信发出。



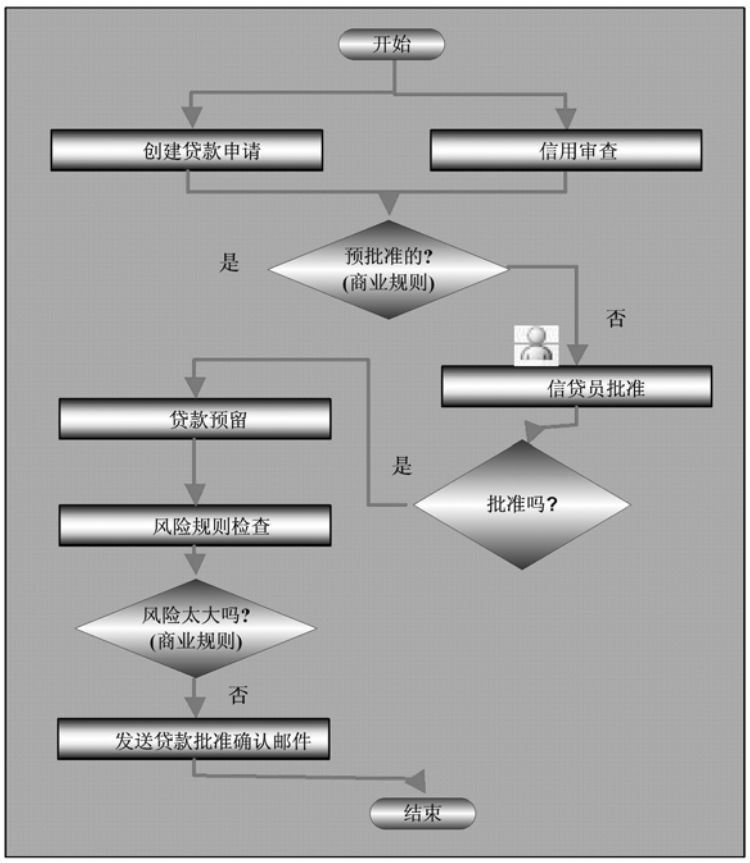
查看数据库的表，确实没有为 poorman 建立贷款申请和资金预留。



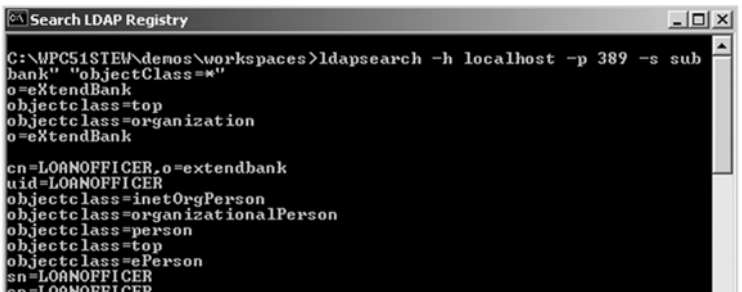
Poorman 查看自己的邮件，发现一封拒信：贷款申请被拒绝。如右图：

需要人员支持的流程

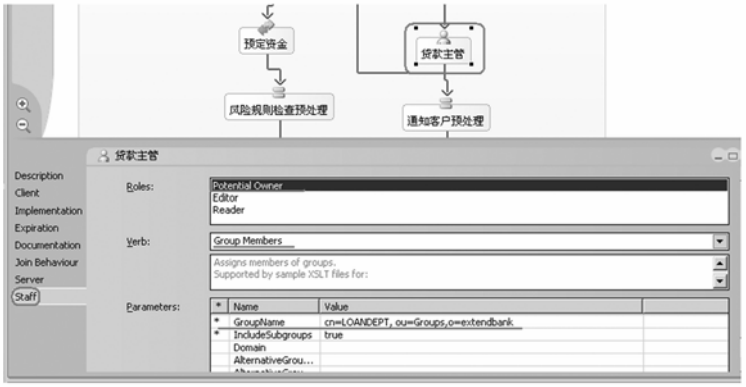
(middleman@demo.com)信用度为“C”，不是“预批准的”。需要信贷员人工批准，如果信贷员同意贷款，则风险检查为 true，即非“大风险”，确认信会发到 Middleman 的邮箱中。



银行的资金操作的安全级别很高，需要对信贷员进行角色控制和身份验证，这里我们选用 LDAP 作为后台的身份认证库，如下图所示，可以看出 LOANOFFICER 这个用户 ID 是一个组成员。

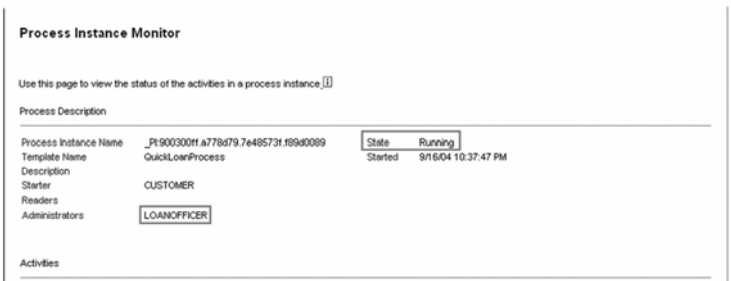


同样，在 BPEL 中需要对“信贷员”这一人工活动进行安全设置，如下图所示，可以充当“贷款主管”，行使贷款批准权限的角色是一个组的所有成员，从上面 LDAP 服务器中的搜索结果可以看出，LOANOFFICER 就是具有这种权利的组成员。

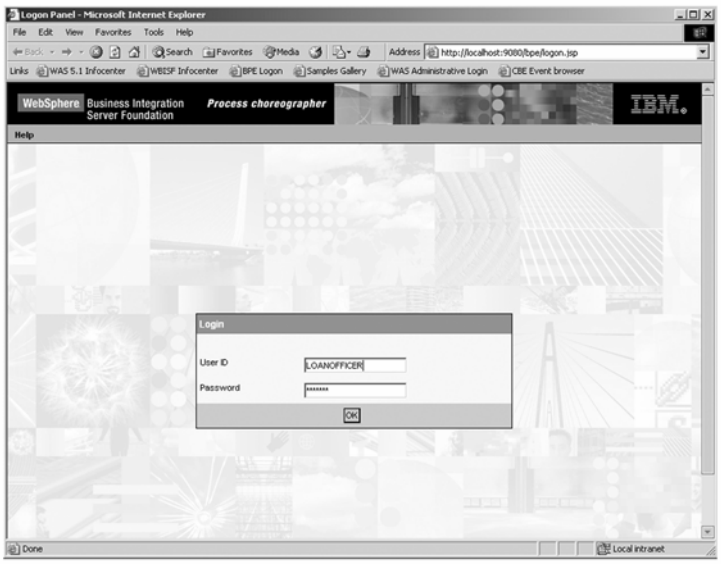


首先 Middleman 在网上申请贷款，客户代号为 2，贷款金额为 75000。

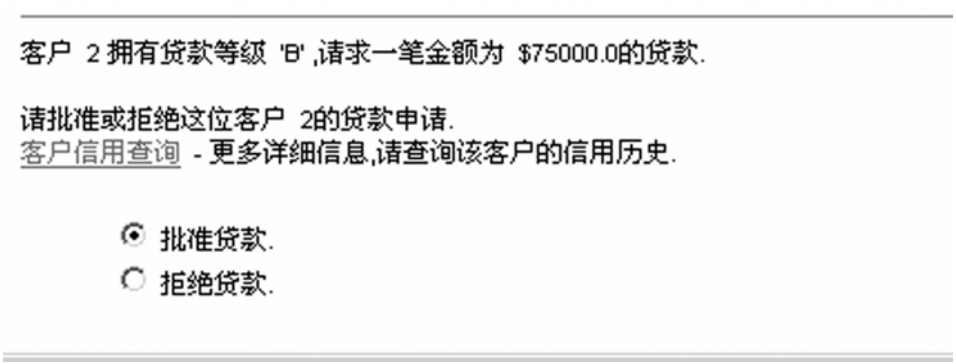
管理员可以在“进程监控窗口”中查看运行状态，如下图所示，商务过程正等待信贷员的批准。



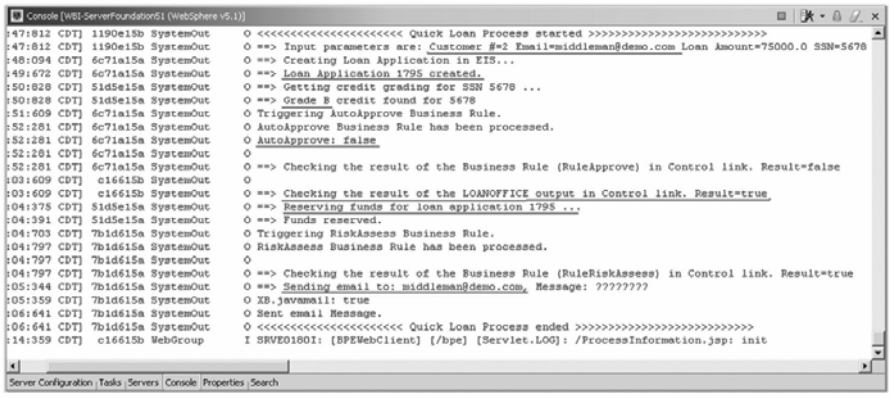
LOANOFFICER 登陆，因为他是具有信贷权利的组成员，所以在他的“待处理事宜”窗口中，会出现 Middleman 的贷款申请。



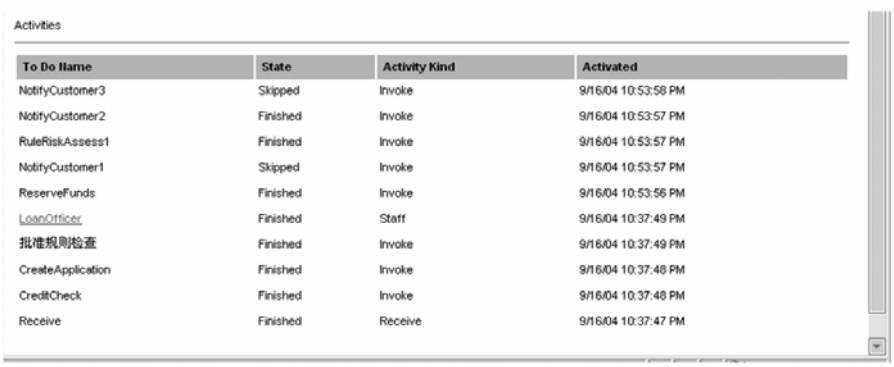
管理员在自己的“待处理事宜”窗口中对 Middleman 的贷款申请做批准，如下图。



过程执行完毕，测试服务器的控制台中显示如下图的执行过程，“AutoApprove:false”表示此贷款申请不是“预批准的”，需要信贷员(LOANOFFICER)人工批准，查看 LOANOFFICER 的结果是“TRUE”，即同意。风险规则检查也是“TRUE”，一封确认信发到 Middleman 的邮箱中。

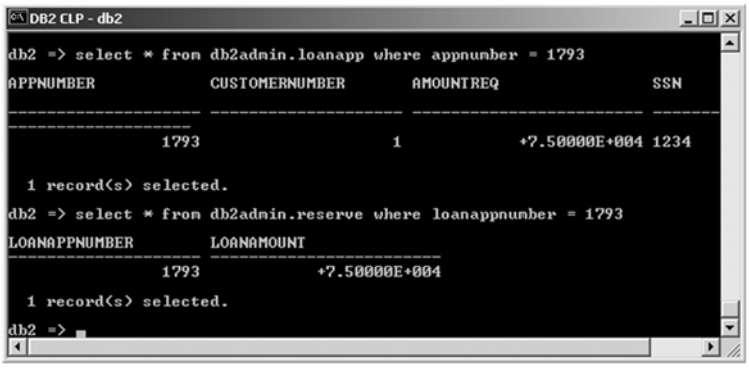


管理员查看“进程监视窗口”，如下图所示。


Activities

To Do Name	State	Activity Kind	Activated
NotifyCustomer3	Skipped	Invoke	9/16/04 10:53:58 PM
NotifyCustomer2	Finished	Invoke	9/16/04 10:53:57 PM
RuleRiskAssess1	Finished	Invoke	9/16/04 10:53:57 PM
NotifyCustomer1	Skipped	Invoke	9/16/04 10:53:57 PM
ReserveFunds	Finished	Invoke	9/16/04 10:53:56 PM
LoanOfficer	Finished	Staff	9/16/04 10:37:49 PM
批准规则检查	Finished	Invoke	9/16/04 10:37:49 PM
CreateApplication	Finished	Invoke	9/16/04 10:37:48 PM
CreditCheck	Finished	Invoke	9/16/04 10:37:48 PM
Receive	Finished	Receive	9/16/04 10:37:47 PM

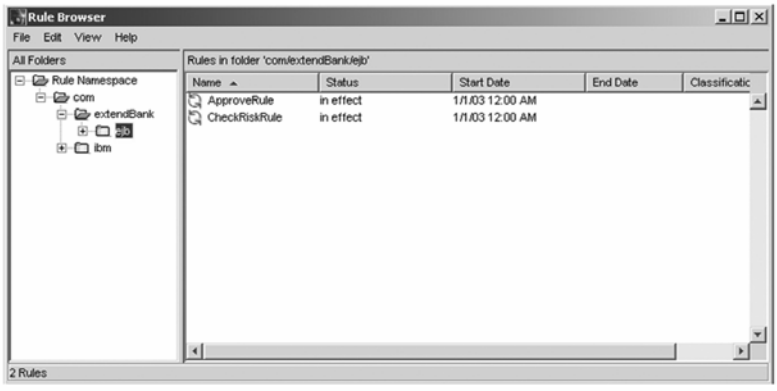
数据库中，Middleman 的贷款信息和资金预留信息已经入库。如右图所示:



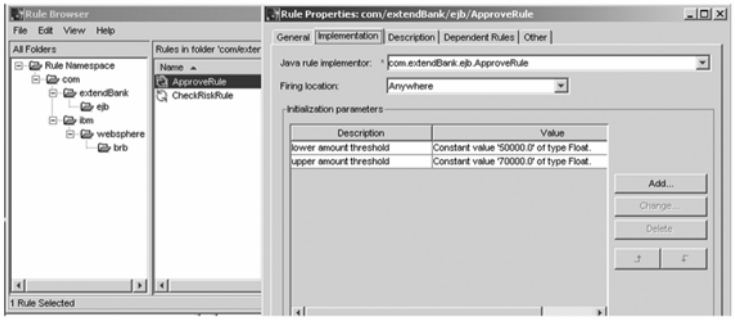
Middleman 会收到一封如右的确认邮件:

商务规则变化的动态响应

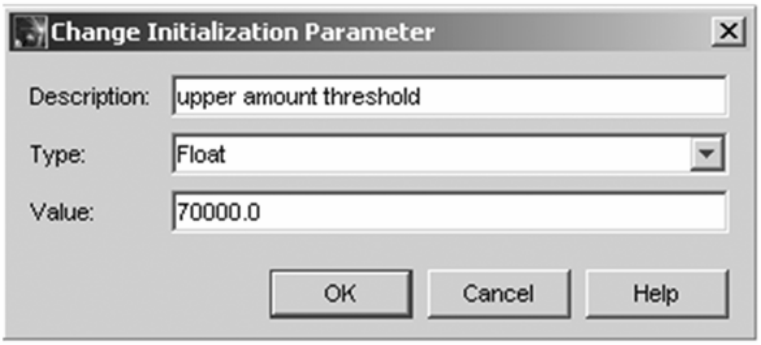
利用 WSAD-IE V5.1 自带的规则实现接口和规则管理器，定制贷款申请所需的商务规则并可以在商务过程运行的任何时刻依需修改规则，在本应用中有两条商务规则——“贷款预批准”的规则和“风险程度”的规则，如右图所示，在规则管理器中设定的规则。



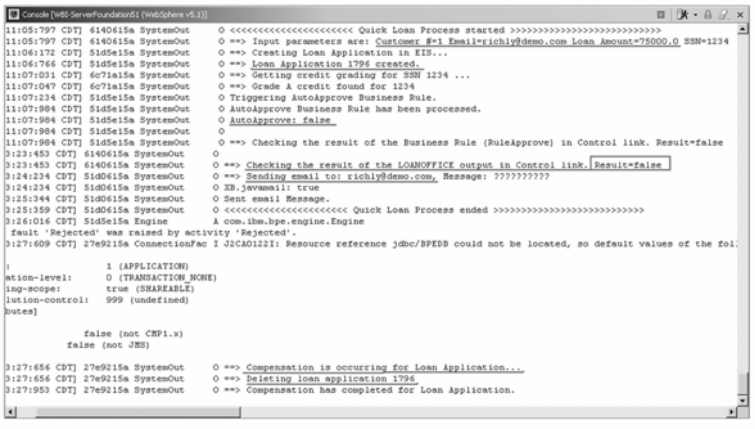
查看“Approve Rule”(预批准规则)，如下图所示。



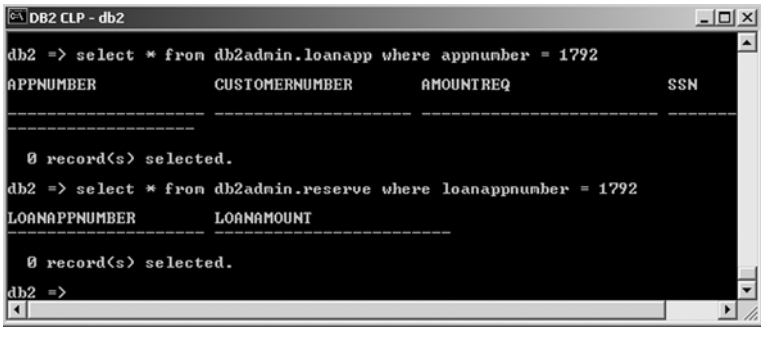
修改预批准规则的上限为 70000，如下图所示。



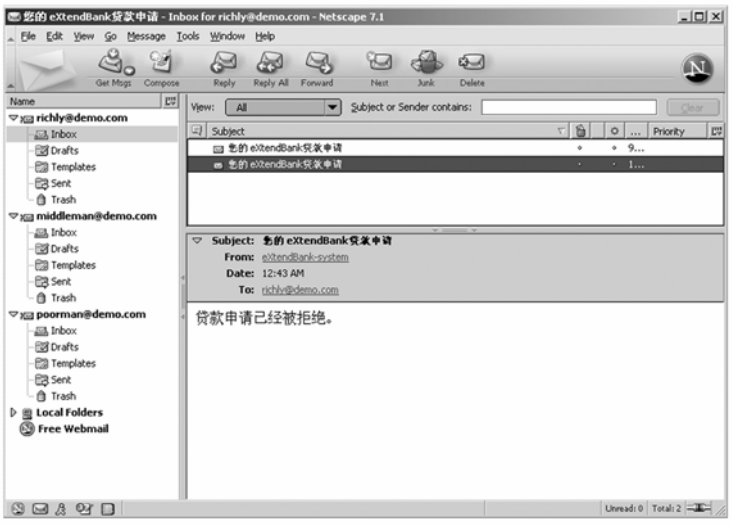
重复执行 Richly@demo.com 用户的贷款申请，仍然输入贷款金额 75000，此时控制台中的执行结果如下所示，贷款的“预批准”审查为 false(AutoApprove:false)，因为贷款金额超过了“预批准”规则定义的上线值，此时需要 LOANOFFICER 的人工批准，但是 LOANOFFICER 拒绝了用户的贷款申请，所以系统自动“补偿”已经写入数据库的“贷款申请”，将之删除。而此时还没有作“资金预留”，因此无需对“资金预留”进行“补偿”。



查看数据库，果然没有此用户的信息。



此时 Richly 查看邮件，收到的是贷款申请拒绝的邮件。



贷款申请的商务流程仅仅是银行内部协同工作的证明

银行内部的业务远不止一个贷款申请系统，许多业务都涉及到应用的整合和协同工作。上面的演示很好地说明了 WBI-MODELER、WBI-SF 和 WSAD-IE 如何配合来实现这种整合，也显示了它们在实现银行业务整合中的优势:

- 快速的商务过程建模。
- 快速的商务过程配置。
- 快速的处理速度。
- 动态响应商务规则的变化。
- 商务过程的实时监控。
- 对失败商务过程的恢复/“补偿”。
- 对操作员的角色控制，保证了安全性。