THOMSON REUTERS
ELITE™

产品简介

3E 业务优化套件
3E 平台概览

THOMSON REUTERS®

基础

3E 业务优化套件™建立在功能强大的技术平台之上。该平台旨在达成以下目标：

- 系统可根据客户的独特需求进行配置和扩展
- 每项交易均支持工作流程
- 用户可在系统内开展协作
- 可扩展性：跨越不同的网络拓扑结构，可供大量用户同步使用
- 一致性：应用程序各部分应以同种方式运作
- 安全性
- 必须整合到事务所的整体 IT 环境之中

为奠定 3E® 产品达成上述目标的基础，我们需要了解 3E 与早期技术方法的不同之处。我们首先来看一下现代网络应用程序所固有的复杂性，对基于 Java® 或 Microsoft®.NET 组件的一般网络应用程序 [1] 中的部分软件层进行分析：

- 构建的组件必须提供远程/本地使用、身份识别和错误处理所需的接口。
- 必须创建和维护类，以实现程序接口、处理例外情况、管理网络连接、操作元数据。
- 必须利用系统服务来记录组件实例、进行状态管理和处理安全问题。
- 必须使用表单、脚本和服务器小应用生成网络服务器页面和客户端页面。
- 必须创建数据对象，用于处理队列、数据库连接和对象关系映射。
- 网络服务需要有描述、资源定位器、协议和图式结构。
- 连接原有系统和第三方系统需要通过特殊连接器和 API。
- 使用 HTML、开发语言源代码、XML、SQL、配置和资源文件等多种途径保存代码。
- 为部署应用程序，必须制作可执行文件并进行打包；必须协调编译器和接口生成器；应用程序需在目标系统（包括数据库服务器、应用程序服务器、网络服务器和消息传送系统）中进行复制、安装和注册。

这种架构虽然灵活性很高，但也极为复杂，软件开发人员将花费更多时间进行基础设施管理。为至少在应用程序业务逻辑端提升生产效率，开发团队试图对框架（代码对象集合）进行开发，但事实证明难有重大进展。截至目前为止，大多数开发团队极少能够实现对象的重复使用；相比之下，复制和修改现有应用程序代码片段则更为快捷。

此类软件工程模式使得软件供应商和用户都深受其害。软件应用程序在开发之初就需要耗费大量的时间、资金，承担过高的风险。但问题还不止于此。软件就如同生物有机体一样，必须不断地发展、变革方能存活。应用程序是对基础业务环境的反映，而基础业务环境是不断发展变化的。但是，当前这一代应用程序却并不能因应环境变化进行升级改进。其中的主要原因包括：

- **停滞：**开展大幅变更的成本和风险过高，因而只能尝试渐进式小幅改进。
- **疲乏：**设计之初急于求成，未有变更预期（这一问题也称为“在狗窝里加层”）。
- **脆弱性：**软件设计时并未考虑变更，因此会轻易崩溃。如发生更大变更，则使得软件更加脆弱。
- **冗余：**每次开发迭代时都要复制对象和组件并相应修改，导致必须同步多套代码和数据库元素。

这就导致应用程序不能适应不断变化的业务和技术要求。软件发布的间隔时间更长，可开展重大变更的程度更小。过去几年里，为应对上述问题，提出了一种称为“软件工厂”的全新方法。

[1] 本节内容根据 Jack Greenfield、Keith Short、Steve Cook、Stuart Kent 和 John Crupi 合著并由 Wiley 出版社于 2004 年发表的《软件工厂：使用模式、模型、框架和工具组装应用程序》一文编写。

示例：

某事务所希望在计算工时录入率 (GetRate) 的应用程序功能中添加独有的业务逻辑。对 customer.object 的代码修改将采用以下形式：

```
Public Overridable Function GetRate As
Public Overrides Function GetRate
    …. (customer code)
    MyBase.GetRate
    … (more customer code)
End Function
```

由此生成的业务对象对 GetRate 基类的功能进行了扩展。

甚至变更业务逻辑也可使用继承能力进行处理。根据应用程序性质的不同，可允许客户版本的对象覆盖基类或追加到基类中。

网页同样属于可继承的对象；网页与业务对象的不同之处在于，网页变更可通过图形表单编辑器进行处理。表面上是由客户创建新版本的表单并对其进行修改，但所有令人头痛的定制化问题也正源出于此。而 3E 平台对网页继承的管理，与对业务逻辑和原型继承的管理采用同样方式。

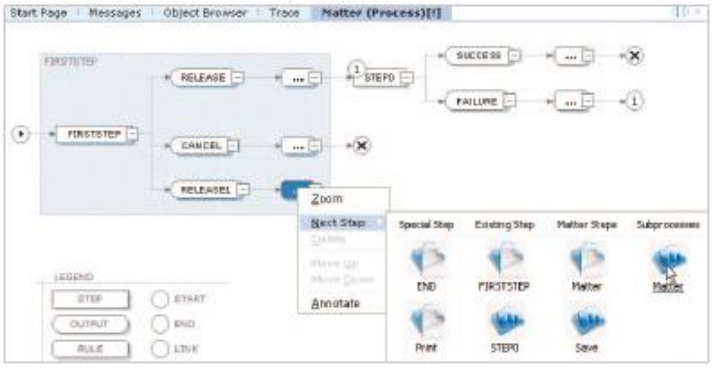
内置工作流与协作

工作流

3E 中的每个网页或表单始终构成流程的一部分。3E 平台设有流程管理器，管理员可用图形方式进行流程控制。以最简单的流程为例，系统显示一个页面，页面内容填写完成后存入数据库。若为更复杂的流程，将为页面设置一套业务规则（例如，发票金额是否超过 10000 美元），并将该页面转至审批步骤。只有该页面完成审批步骤后，相关内容才会存入数据库。

所有交易和数据采集页面都启用工作流，而无须对页面或底层业务对象做出更改。工作流是软件工厂提供的一项服务，为 3E 平台本身所固有。从概念而言，这也是 3E 平台的工作原理。

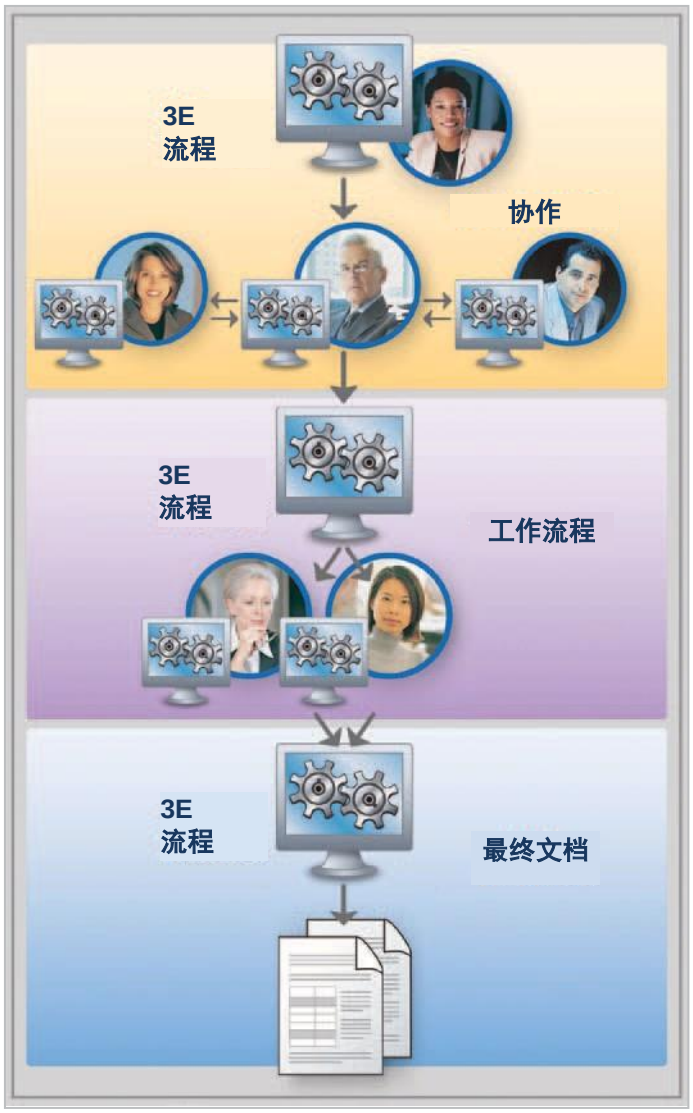
用户在网页中输入信息时，并非在与数据库“对话”，而是在构建全新的 XML 文档。该文档在 3E 术语中称为“草稿”，作为单独的对象存储在数据库中。用户完成页面操作时，流程管理器会根据先前制定的业务规则对 XML 文档进行审查。假设下一步将进行审批。流程管理器将知会流程中的下一个人（或角色）应采取的行动。然后，审批人会查看该页面（或者单独的审批页面），并转至流程的下一步。仅在流程管理器确定页面可以保存时，才会执行保存操作。保存将使用以 XML 格式存储的信息，并更新数据库中的相应表格。该网页的原始开发者并不知晓客户方的工作流程。3E 被设计为可对系统进行配置，并允许客户针对该应用程序创建特定的实施方式。



协作

协作可视作临时工作流程。工作流程建立在业务规则和预先界定的流程之上，而协作则是一种工具，它允许用户彼此发送页面征求对方意见或进行更新。仅举以下示例：账单编辑过程中，主管律师正在对形式账单进行审查。她发现账单中有两名高级律师的时间记录，但她却并不知情。此时，主管律师可使用协作服务，将形式账单发送给相关律师，征求其意见并进行更正。相关律师作出回复后，主管律师可接受、拒绝或作出修改，然后发送形式账单，走完其余工作流程。

协作服务利用了页面 XML 结构的优势。主管律师是在草稿上进行处理。该草稿 XML 文档包含与页面每个字段相对应的元素。使用协作流程，可对每个字段进行多个输入。3E 将对各版本的输入进行跟踪，并向协作发起人显示不同的版本。打个比方，Microsoft Office® 文档的创建者可在协作流程中跟踪对文档的修订；3E 已在事务应用程序中采用这一模式。再次申明，协作服务由软件工厂提供，应用程序开发人员可以“免费”获取这些服务。



协作和工作流程均建立在 3E 正拟申请专利的草稿技术之上。3E 将用户体验与数据库更新事件分离，打破了过去律所在应用程序之外进行系统定制或建立工作流程的藩篱。数据库保存事件由 3E 控制。3E 理解原型、业务对象和网页的 customer.object 版本，可在流程管理器确定的时间执行。利用 3E 的上述基本功能，可将律所的业务流程嵌入到应用程序之中。

可扩展性

3E 的设计允许用户壮大所使用的系统，以支持大量并发用户同时使用，而又不会产生技术瓶颈或导致性能下降。扩展性并不仅仅是技术或平台问题，同时也是应用程序的设计目标。因此，我们首先会审查应用程序设计是否允许扩展，然后再对 3E 技术平台作出与性能有关的设计决定。

并发使用

通常，金融应用程序会维护在线查询和报告所使用的汇总信息相关的数据库表。这类表格范围越广，查询速度越快。但会牺牲并发使用性能。如果多个用户同时试图更新汇总统计数据，系统会瘫痪，并拖慢事务流程。3E 通过在数据库中维护一个事务队列，解决了这一问题，事务队列是一个表格，其中保存了指向数据库中所有新事务的指针列表。举例来说，如果用户录入现金收据，则工作流程完成后，3E 会更新数据库中的事务表格，在事务队列中写入一行数据。而 3E 调度器则管理着一个单独的进程线程，该线程会读取队列数据并以“近乎实时”的方式更新汇总表。在线查询和报告功能可受益于汇总数据（如客户方和工时人员水平总计数据），而事务处理引擎则不受锁定开支的影响。

内置商业智能

性能下降最常见的原因之一是大量用户生成（纸质或电子）报告。为解决这一问题，系统设计人员和管理员曾试图构建数据仓库，将信息复制到独立的报告服务器，甚至限制用户生成报告。这类解决方案都不太理想：会产生大量管理成本，还有“你使用的数据是哪个版本”这样的问题。我们设计了 3E Metrics Engine（指标引擎），试图缓解性能下降的问题。管理员可安排在预定时间间隔内运行报告（通常在非高峰期或期末）；计算结果存储在特殊数据库表中。报告查看器可向用户提供这类预先设定的报告运行的结果。仅当用户需要绝对实时的信息时，才会执行新的计算过程（而且这类计算仅针对一小部分数据）。提供最新（前一晚或上一期末）的信息而非实时信息是一项有力工具，可消除这一普遍存在的周期性性能问题。

3E 架构

3E 技术平台为实现可扩展性功能而采用了三层设计。第一层是数据库服务器。3E 使用 64 位版的数据库引擎，使系统有更大的内存空间用于缓存和整体性能。此外，所有数据库访问操作均由 3E 对象查询语言 (OQL) 控制。3E 应用程序与数据库互动的任何请求，都必须通过 OQL 引擎提交。OQL 的其中一项优势在于能够优化数据库查询性能，并利用平台特定功能。连接池的管理也通过此 3E 数据访问层执行。对数据库引擎的连接，每次启动或关闭都会产生巨大的开销。3E 允许管理员维护一个连接池，供所有用户共享。每个 SQL 语句执行完毕后会释放连接，该连接将再次返回连接池。

3E 架构的第二层是应用程序服务器。应用程序代码的执行和网页生成正是通过这一层实现。此外，3E 还会在这一层利用 64 位服务器交付最大操作性能。随着系统发展，客户可通过增加低成本应用程序服务器的数量来进行系统扩展，操作十分简单。如果采用前一代技术，只能通过成本高昂的硬件更换才能实现系统扩展，而 3E 方案则可在机架中添加简单的双处理器应用程序服务器，以此方式实现渐近式扩展。也可添加负载均衡器和其他管理工具，以此对系统进行微调以优化硬件利用。

网络整体性能是最后一层。3E 利用以下两种基本技术对速度和可扩展性进行优化：

- 用户如处于面向交易的页面，则仅可在网络上发送 XML 文件来实现与网络服务器的通信。然后，客户端网络浏览器将 XML 渲染成 HTML 并显示该页面。所发送的 XML 文件仅占最终呈现的 HTML 文件的一小部分。在广域网环境中，操作受带宽和延迟限制，客户端渲染的性能可能会有巨大差异。
- 3E 充分利用缓存工具提升响应时间。在客户端开展版本控制，可确保 XSL 文件、Java script、运行时表单定义和信息文件会自动缓存，仅在相关内容修订时才会刷新。应用程序服务器端的特殊算法可高效加载数据对象，并尽可能通过服务器进行缓存。

最后，Thomson Reuters Elite™ 在性能测试工具之上，还建立了可一起运作的测试脚本库。系统设计人员和管理员可通过调整该等脚本，在自有环境中验证系统性能。

安全

3E 安全模型在概念上可分为三个层次：

1. 控制用户可访问的流程
2. 控制用户可访问的对象元素
3. 控制用户可访问的数据库部分

前两个层次受 3E 内部安全模型控制。用户可被分配一个或多个角色。每个角色可分配一系列访问控制原则。这类控制原则确定用户是否有权限访问某个流程。如赋予该类权限，则对用户能否查看或更新底层对象的每个属性进行控制。举例来说，某用户可能被分配“律师”角色。律师角色可访问客户流程及其底层客户对象。不过，他们可能仅被授予信息查看权限，而未被授予信息更新权限。即使如此，仍有可能限制他们查看特殊费率或初始数据的权限。

一个用户可能具有多个角色。例如，律师可能被分配“合伙人”和“专业服务组主管”等角色。针对所授予的两个角色，3E 安全模型将始终向该用户授予最为宽裕的访问权限。举例来说，律师被授予合伙人这一基本角色，可能仅能查看自身的摘要信息。如果该用户同时被分配专业服务组主管这一角色，则可能有权限查看该小组层面的摘要数据。

3E 可在数据库行层面进行安全管理。可限制某个用户的访问权限，使之仅可查看或添加特定的数据。普通用户的查看权限可能限制为仅可查看所在办事处的统计数据，或禁止其查看信息隔离墙之外的信息。3E 利用 SQL Server 2005 的新功能处理这一问题：能够针对每个用户创建特殊模式和视图。每当用户执行数据库事务之时，会以其用户身份连接至数据库，并执行针对该用户的数据库视图。数据库严格按照允许该用户查看的数据对查询做出回应。该等视图虽由 3E 管理，但操作活动本身由数据库引擎执行。

该架构有一个优势：使用第三方工具也无法规避安全规则。举例来说，如果用户通过 Microsoft Excel® 连接到数据库，并使用数据查询工具将信息提取到电子表格中，此时 SQL Server 引擎仍将强制执行该用户在数据库中的受限视图。

一致性

如今，用户无需接受专门培训即可访问互联网上几乎全部商业网站。内部应用程序用户也有同样的期望。从事务所角度看，大范围培训花费不菲，而且往往并不可行。如果应用程序采用特殊操作方式，可能导致过于侧重专长知识，不能灵活地对员工开展交叉培训。计费流程操作人员仅熟悉计费应用程序如何操作；应付账款操作人员也仅了解他们的应用程序。3E 采用软件工厂技术解决了这一问题。开发人员仅关注业务逻辑和功能，3E 平台生成用户界面。基本屏幕形式、协作操作控制、工作流程、查询和查找均受 3E 控制。用户只要了解如何对客户进行查询，就能对供应商进行查询。只要用户能够输入总账分录，就会操作使用系统中的其他交易界面。

我们设想将用户分为两类：第一类为一般用户，该类用户访问系统的主要目的是查看信息或进行简单的编辑操作；第二类为高级用户，他们需要了解所在领域的全部功能。为实现操作的一致性和简洁性，第一类用户应只需简单了解 3E 常规规定即可进行操作。第二类用户则应更专注于所在领域的专业知识，而非如何操作应用程序。举例来说，对应付账流程操作员的培训，可集中于 1099 报表或增值税如何正确处理，而不是屏幕上的按钮如何操作。

整合

任何应用程序都不能孤立运行。正如其他优秀的软件公民一样，3E 需要与同侪共同运作并分享信息。使用上一代技术，这一过程非常痛苦。每个供应商各自提供一套 API（独有的程序接口），如何将所有部分连接起来成了客户的问题。不同软件版本中使用的这类接口不能保持一致，无法对验证、错误处理和管理进行标准化。

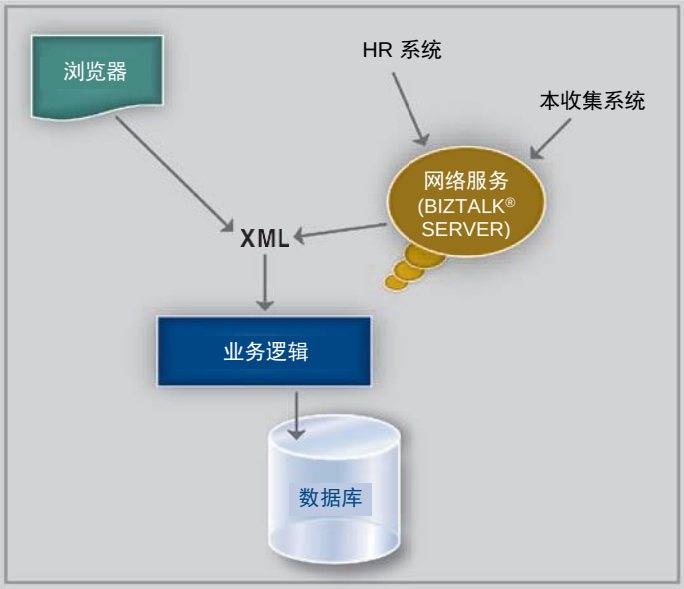
整个软件行业正拟使用网络服务来解决上述问题。行业正在开展网络服务标准化，方便应用程序随时互动，在生产环境中也可对交易流程和执行进行管理。3E 采用基于服务的架构 (SOA)，专门针对这类集成环境而设计。

同样，我们采用最简单的方法，以工作流程和协作为切入点，对这一架构进行讲解。每个用户可通过网页进行操作，创建底层 XML 文档（草稿）。无论这一 XML 流是通过网络服务生成，还是由用户通过互动操作直接生成，3E 都可接受。XML 提交给 3E 后，将同样受到针对每个用户设置的安全和流程管理规则的约束。举例来说，某事务所可能使用工时人员录入界面新增工时人员；另一家事务所可能选择通过内部 PeopleSoft HR 应用程序添加工时人员，然后使用网络服务器在 3E 中进行信息更新。工时人员应用程序（也即底层对象）具有不可知性，也并不在意 XML 信息的传递方式。

3E 使用 Microsoft BizTalk® Server 这一工具对网络服务流进行协调。业界普通认为，BizTalk Server 是服务环境管理的有力工具。3E 作为团队成员，应能够实现整合，并降低成本和复杂性。

结语

Thomson Reuters Elite 对 3E 平台及底层 .NET 技术有所投入。关于这一系统环境的全部功能和灵活性，本文仅触及部分表面认识。我们深信，3E 这一名副其实的平台，将令客户与我们共同受益。3E 将融入客户的业务运营，在业务所在之地，3E 平台都将促进客户运营的变革和改进。



迈向成功之路的伙伴

Thomson Reuters Elite 为律所和专业服务机构提供端到端企业业务管理解决方案，可整合业务运营流程的各个领域，包括业务发展、风险管理、客户与事项管理以及财务管理。作为全球行业领先机构，我们具有律所业务和财务运营方面的专长，我们的工具可以简化流程，提升效率，还可根据客户业务的变化和增长进行灵活调整。

如需了解业务优化套件的更多详情，或获取全球办公地点列表，可访问:
thomsonreuters.cn/zh/products-services/legal/elite-3e2.html

THOMSON REUTERS®