[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
G06F 17/30 (2006.01)
G06Q 10/00 (2006.01)

[12] 发明专利说明书

专利号 ZL 200510076191.0

[45] 授权公告日 2008 年 9 月 3 日

[11] 授权公告号 CN 100416565C

[22] 申请日 2005.6.8
[21] 申请号 200510076191.0
[73] 专利权人 香港理工大学
地址 香港九龙红磡
[72] 发明人 刘湛华 李嘉敏
[56] 参考文献
CN1435780A 2003.8.13
CN1384451A 2002.12.11
US2003/0192041A1 2003.10.9
WO01/59613A2 2001.8.16
基于 PDML 的产品数据交换与集成. 章新燕，高亮，李培根，萧健. 机械设计与制造，第 3 期. 2002
虚拟企业中面向信息共享的知识管理. 王洪伟，吴家春，蒋馥. 计算机工程，第 30 卷第 3 期. 2004
审查员 张 潇
[74] 专利代理机构 隆天国际知识产权代理有限公司
代理人 张龙哺 郑特强

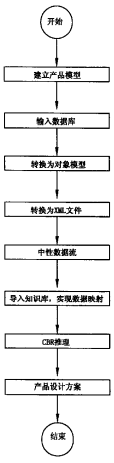
权利要求书 2 页 说明书 16 页 附图 7 页

[54] 发明名称

用于异质环境之间信息交换的自适应程序调用方法

[57] 摘要

一种用于异质环境之间信息交换的自适应程序调用方法，包括步骤：将数据库中的产品模型数据转换为对象模型，并进一步转换为扩展标记语言文件；利用扩展样式表语言，将扩展标记语言文件变换为中性数据格式的数据流；以及将数据流导入到知识库中，实现数据库的数据与所述知识库的范例之间的数据映射。 本发明的自适应程序调用方法将对象技术与产品信息标记语言相结合，实现了异质系统之间有效的产品信息交换，加速了产品设计的开发周期，缩短了市场变化的响应时间。



1．一种用于异质环境之间信息交换的自适应程序调用方法，包括如下步骤：

a)将数据库中的产品模型数据转换为对象模型，并进一步转换为扩展标记语言文件；

b)利用扩展样式表语言，将所述扩展标记语言文件变换为中性数据格式的数据流；以及

c)将所述数据流导入到知识库中，实现所述数据库的数据与所述知识库的范例之间的数据映射。

2．如权利要求 1 所述的方法，其中，在步骤 a)之前还包括如下步骤：

根据市场的变化，建立新产品模型，并向数据库输入产品模型数据。

3．如权利要求 1 所述的方法，其中，在步骤 c)之后还包括如下步骤：

经由范例推理接口引擎，从所述知识库中取回范例，作为新产品的设计参考方案。

4．如权利要求 1 所述的方法，其中，在步骤 a)中对产品模型数据进行对象建模时，包括对产品开发过程中与产品数据处理有关的操作进行建模。

5．如权利要求 1 所述的方法，其中，在步骤 a)中对产品模型数据进行对象建模时，包括对产品开发过程中涉及的具有行为的实体进行建模。

6．如权利要求 5 所述的方法，其中，当建模后的对象启动消息时，在消息签名中指定待执行的程序名称和待包含的参数。

7．如权利要求 6 所述的方法，其中，所述程序是标准的生产程序，用于实现一类对象的操作。

8．如权利要求 7 所述的方法，其中，所述操作可在所述扩展标记语言格式下通过多种所述程序来执行。

9．如权利要求 6 或 7 所述的方法，其中，当所述程序不满足设计变化时，删除和修订所述程序。

10．如权利要求 1 所述的方法，其中，所述对象模型具有多态性和继承性。

11．如权利要求 1 所述的方法，其中，所述对象模型包括名称域、属性域和行为域。

12．如权利要求 1 所述的方法，其中，所述中性数据格式包括用于其他信息系统的扩展标记语言格式、超文本标记语言格式和文本格式。

13．如权利要求 1 所述的方法，其中，还利用 VB 脚本、Java 脚本、C++，对扩展标记语言文件进行变换。

14．如权利要求 1 所述的方法，其中，所述范例具有信心值参数，该信心值参数表示该范例解决产品设计问题的成功率。

15．如权利要求 1 所述的方法，其中，所述知识库的内容和所述范例的数量均随着产品设计人员经验的增多而动态增长。

用于异质环境之间信息交换的自适应程序调用方法

技术领域

本发明涉及一种信息交换方法，尤其涉及一种用于异质环境之间信息交换的自适应程序调用方法。

背景技术

产品开发在企业运作的过程中有着举足轻重的地位和作用，它面临着日益加剧的行业竞争、快速更新的生产技术和日趋复杂的全球市场，越来越需要缩短用以满足市场需求的产品开发时间，这些挑战都迫切要求将基于知识的系统应用到相关产品信息的交换和更新中。

然而，沟通不足常常导致工程师无法在正确的产品规格版本下工作，特别是在当今产品日渐复杂的情况下，由于团队之间缺乏协作，设计师设计的零部件可能无法相互匹配。当前的产品数据管理(PDM)被用来在特定时间传递恰当信息，以便改善各种功能组之间的合作。产品开发的早期阶段中可用的信息是相当重要的，因为产品成本在很大程度上取决于此阶段。因此，使得产品设计在团队中更早得到确认和理解无疑是值得倡导的发展潮流。

在当今的知识经济趋势下，大多数增值服务在实际操作之前需要恰当信息的支持，按照消费者的要求来提供设计就是一种方兴未艾的增值过程，从原始设备制造商(OEM)向原始设计制造商(ODM)的范式转变是不可逆转的趋势，工程师需要基于他们的既往经验和知识而非具体的策划和分析来进行产品设计决策，从而完成每项订单。为了增强设计的敏捷性，有效协作能够使设计师获得足量信息用以决策。难以获得所需信息的状况已经成为历史，现在人们在数分钟之内即可轻易获得大量信息，但是有些信息仍然由于数据交换屏障而无法被获得，因为各公司具有自己专有的输入和输出文件，这些文件可能属于不同的领域和具有不同的语义。

这里说明一下企业知识库在企业中发挥的作用。企业知识库包含企业数据库，但是内涵更为广泛，包含所有与企业有关的信息和知识。知识库并没

有什么固定的模式，而是要根据组织的具体情况来定，例如可建立在企业内部网上，由安装在服务器上的软件构成，员工可利用该系统阅读公报和查找历史事件，并彼此在虚拟的公告板上相会。该知识库的内容可包括：人力资源状况、职位所需技能和评价方法、公司各部门和各地分公司的内部资料、公司历史上发生的重大事件、客户信息、竞争对手及合作伙伴资料、公司内部研究人员的研究文献和研究报告等等。一些著名公司在加速新产品开发速度的过程中发现 60％以上的技术问题其实已在其他小组的开发经验中碰到过且得到了解决。显然，如果建立一个“最佳方法知识库”，让每个人分享他人经验，就能大幅度减低问题重复出现的概率，可将新产品产出的速度提高近一倍。同时，数据库等商业化应用软件极大地方便了知识库的创建和使用，帮助企业知识库系统软件实现了集成性、外向型、决策支持等功能和特性。通过建立知识库，可以积累和保存企业的信息和知识资产，加快内部信息和知识的流通，实现组织内部知识的共享，具体表现在：使信息和知识有序化；加快知识和信息的流动，有利于知识共享与交流；有利于实现组织的协作与沟通；帮助企业实现对客户知识的有效管理。

一些产品数据标准和信息交换系统已被提出用来在并行工程中支持设计师进行信息管理和交换数据，这就需要一种有助于数据交换的通用标准，于是电子数据交换(EDI)应运而生。EDI 使得结构化商务文件的处理得以标准化，这些商务文件比如包括订购单、发货单、付款、配送、交割时间表等等。EDI 将这些文件转译成全球理解的商务语言，并利用安全的电信链路在贸易伙伴之间传送。然而，中小型企业无法承受 EDI 实施和维护的高额成本，于是利用可扩展、可读和同步数据的扩展标记语言(XML)有望取代结构不灵活的 EDI。

因此，促成了基于知识的系统的出现，这些系统辅助相关人员掌握编码后的知识以完成相关任务，极大地依赖于信息更新，并利用相应的算法来解译知识。但是，通常考虑的是如何获得可增强解译算法的新技术，以便实现有效的知识共享和再利用，基于知识的系统和存储最新信息的联合数据库之间的连通往往被忽视。异质系统之间低效的数据交换会造成数据准确性的丧失，不利于产品开发工艺。

多数人关心的是构建类图，而未考虑对动态行为进行建模，即使注意到

了动态行为，却忽视了多态性，仅描述了类的定义和通用函数。对象建模技术主要关注结构性的建模，但是很少涉及行为性的建模。行为性建模的核心与数据流图(DFD)不同，后者被认为是非面向对象的设计和分析。同时，对象技术仅被用于设计和分析，因为通常认为仅利用单项技术是能够降低分析过程的复杂度。然而，当面临与不断变化的商业环境有关的新问题时则显得不敷使用。结果，应当在企业信息系统中同时运用对象模型和数据流。

下面回顾一下与产品开发有关的现有技术。当今的制造业面临着环境不断变化的挑战，需要将优秀的产品数据管理(PDM)系统应用到商业工作流和生产过程中以进行数据的有效管理，这意味着 PDM 将通过协调和控制信息的存取，在复杂的制造生命期内控制和操纵数据，从而优化这些过程。

传统的过程方式关注的是处理内容和步骤，而面向对象的方式有助于获得更佳的认知模型。对象技术作为一种建模技术已被应用到 PDM 中，其将现实世界的实体解译为具有显著身份和特征的对象。对象技术指导相关人员思考目标实体以及目标实体与消费者之间的关系，改变了程序员的思考方式，常用术语例如包括对象、类、行为、属性、继承性、多态性等等。在各种商业环境中，对象技术在重利用性和重构性方面的适用性得到了证实，已从计算机辅助策划、装配系统等传统领域开始扩展到产品开发领域。

当通用的数据交换标准不可用时，异质系统之间的数据通信成为重中之重。扩展标记语言(XML)作为一种新兴技术，能够在不同的平台和系统上操纵、存储和交换数据或元数据，被认为是下一代的数据表示手段，其在制造领域中的应用有助于有效的数据交换。XML 实际上是一种文件媒介，将来自各种资源的信息加以组装，并在通用的智能电子格式下进行编码。例如，XML 在电子商务的应用中，借助网络技术，允许消费者和供应链各成员相互协作地设计、构建和部署产品及服务，实现关键信息的充分利用。

产品数据标记语言(PDML)为产品数据交换提供了一种新的范式，其定义了一套应用程序交易集，用于明确数据请求，有助于商业过程的综合和协作。XML 还被认为是在指定的数据结构设计原理下的数据构造或编码模式。制造标记语言(MML)也并入 XML，在不同代理程序之间交换可制造性请求和产品信息，并在决策环境下提供关键域模型及其集成概况，将分布式对象标准与网标准和协议加以组合，在集成产品和过程开发(IPPD)环境下建立对

象网。该方式允许对企业知识库的单独存取，提供决策支持功能，以实现有效的设计和制造。至此，XML 作为一种管理元数据的网标准和协议，与产品数据管理(PDM)形成了更为完整的企业解决方案。

但是，当对象技术应用于制造系统时，在产品开发工艺中缺乏对象技术和 XML 之间的联系。关于实现有效的产品数据交换的技术数不胜数，但是这样的问题未得到应有的重视：如何跨越公司数据库和知识库，实现通用的数据交换，从而在产品开发中有效地利用知识库中存储的知识。

发明内容

鉴于上述问题，本发明的目的在于提供一种用于异质环境之间信息交换的自适应程序调用方法，其能够增强产品的开发操作，该自适应程序调用方法通过重构各种程序以完成相同操作，实现对于外界刺激的响应。该自适应程序调用方法采用成熟的对象技术，比如多态性和继承性，以支持产品变化的管理。首先，通过在不同条件下以特定的格式综合多个程序，来实施具体操作；然后，通过重构继承属性，以便为新的情形修改结果，从而将分类方法应用于支持产品变化请求的认定。这些可重构的方法被重新用来处理动态环境。

当一个对象启动一消息时，会在消息签名中规定其格式，用以指定待执行的方法名称和待包含的参数。同一操作可应用于不同的类，这样的操作被称为“多态性”，其表示在不同的类中表现为不同形式的同一操作。通常，一种程序用于实现一个类的操作。然而，在本发明中，一项操作还可通过在不同的条件下以特定的格式综合多种程序来实现。本发明提供的一种用于异质环境之间信息交换的自适应程序调用方法旨在使得对象能够调用各种行为用于同一消息，并且超越预定义的继承操作，从而将标准过程灵活地再用于产品开发活动，通过在产品寿命期的管理中识别基本的程序问题和评估当前的状态来适应动态的市场。

本发明的自适应程序调用方法旨在利用对象模型来表示产品开发环境中数据对象之间的处理和关系，从而在迅速变化的企业环境中将商业活动与日常操作的数据相匹配，实现一种能够便于建立有益的生产环境以提高产品开发效率的自适应方法。

为了实现“动态”信息交换能力，本发明将对象技术与产品信息脚本合并，有助于有效的信息交换过程。可采用通用的扩展标记语言，来实现数据库与知识库之间的数据交换，从而使实时数据和知识在整个企业内可用。也就是，本发明使得用于产品开发初始阶段的新产品数据方案公式化，允许为相同的消息调用各种行为，超越预定义的继承操作，从而能够在迭代的产品设计过程中使灵活的相关性公式化。

本发明提供的在异质环境下交换数据的方法和系统是基于规范的形成，例如产品信息标记语言(PIML)，来实现不同数据库模型之间的信息双向交换。该方法和系统利用了形成 PIML 的一部分的对象技术的特性(比如多态性和继承性)，由此提高产品信息交换性能。与现有的数据集成方式不同，PIML 使用扩展标记语言(XML)，形成了通用标准(即 PIML 是 XML 的修改版)，提供了与平台无关的用于交换信息的灵活方式，其通过利用对象技术来提供精确的语义描述，形成通用的数据交换标准，无论使用何种平台，PIML 都能够支持广泛多样的应用程序，利用文件对象模型(DOM)来呈现文件。这种跨平台的数据交换系统有助于建立有益的产品开发环境。

附图说明

图 1 是示出了根据本发明的基于对象的知识集成系统(OBKIS)的基础机构图；

图 2 是示出了根据本发明的用于异质环境之间信息交换的自适应程序调用方法的流程图；

图 3 是示出了根据本发明的 OBKIS 的概念层次图；

图 4 是示出了用于数据层的数据流图(DFD)；

图 5 是示出了实体层的实体关系图；

图 6 是示出了动态对象层的对象模型图；以及

图 7 是示出了 PIML 中范例推理(CBR)的界面图。

具体实施方式

根据本发明的基于对象的知识集成系统(OBKIS)具有兼容性和交互性，构成该系统的关键部分具有如下特征：

协作性——在贸易伙伴之间存取和交换信息；

协调性——保持组织内部各部门之间的畅通操作；

敏捷性——由于相似事件出现在不同情形下，所以该系统应当具有重构各种行为的能力，从而在各种环境下是动态的；

智能性——利用人工智能，在波动的产品或服务市场和灵活的制造环境中解决问题；

数据同步性——来自关系数据库的数据应当经常被复制或迁移到知识存储库中。

鉴于企业信息系统的需求，本发明的 OBKIS 是利用了用于系统建模的面向对象技术来开发的，因为该技术考虑了未来再利用、可维护和较少的后期错误。产品信息标记语言(PIML)(包括扩展标记语言(XML))作为信息基础构架上的数据交换媒介，而与组织之内的平台和系统环境无关。PIML 还有助于协作式的产品开发，团队成员可以是同一组织的同事或者是战略伙伴的人员。

在企业信息系统中，每项操作的方法的实施都需要大量数据，这些数据是公司资产，知识则是个人智力资本。尽管通过不同的定义能够将数据、信息和知识区分开，但是它们相互之间的互连是不可否认的。在知识存储库存储着范例和事实的同时，连接于关系数据库的“范例推理(CBR)”接口引擎形成了集成系统。本发明的 OBKIS 具有动态属性，其随着由范例推理所代表的经验而增长，从而允许使用者对于知识进行操作，这些知识受到来自关系数据库的更新数据支持。OBKIS 关注公司数据库和知识库的综合。数据库中存储的产品数据被转换成中性格式，因为不同数据库系统中的数据可被导出到中性格式和被同化到知识库。大多数基于知识的系统可支持中性格式，但处理迁移过程仍然很费劲，比如在关系数据库中搜索相对域，并导出到知识存储库中的恰当位置。于是采用对象化方式，其基于与对象技术相关联的技术，将关系数据库模式转换成对象模型。对象化可将数据簇变成相互关联的对象形式，提供与之有关的特征，比如继承性、多态性和封装性。一旦数据在对象形式下被重构，产品数据可基于 PIML 被转换成 XML。PIML 是中间级，连接客户端和数据库。具体来说，PIML 要求一种用于数据交换的有组织的文件结构，利用这些与提取信息、进行变化和查询文件等有关的数据

信息。OBKIS 框架如图 1 所示。其中，在此说明的是：PIML 是由 XML 发展而来，用来弥补用于产品数据管理的可扩展和柔性语言（expandable and flexible language）发展过程中的差异。因此，XML 可以与 XSL 结合，来产生用于其他系统、企业以及基于知识的系统（一种基于知识的系统是 CBR）。PIML 的详细操作过程如右下角的椭圆框中所示。

OBKIS 的分层特性是在层图中反映的，包括数据层、实体层和动态对象层。每层的输入和输出在图 3 中示出。解译规则和判断规则经过各层向上传送，决策和查询则是向下发送。

OBKIS 数据层的特征在于利用数据流图(DFD)进行系统建模，综合各种网络处理，比如变换和交换数据，它是 OBKIS 的基本层，有助于层组件之间的数据流动，常见的层组件包括：数据处理，其在 DFD 的循环中有所描述，对数据结构或数据中封装的信息进行转换，建立新的信息，其中利用了上下文图为系统建档，概括出整个系统的处理；数据存储，其包括计算机系统、CAD 库和统计系统，允许数据存储备用，并不进行任何操作，但允许输入值不同于输出值，因为输入流可能通过追加、删除元素或改变数值来修改存储的数据；参与者，其生产和消费数据，是 OBKIS 数据层的主要“数据驱动器”，在图中用方框表示，位于图边界上，同时也是终止者，比如个人、组织或系统；数据流，其表示为→，其形成了这些处理与数据存储之间或者这些处理与参与者之间的数据流线，代表着计算或分析处理之内某些临界点处的数据值，数据流在不同的过渡阶段中被大体分为控制流、更新流和结果流，承担着不同的角色。

构建数据层的规则包括：只有在所有输入流是可用的且仅一个输入流存在时，一项处理才能够执行它的功能；不允许从相同的处理传送相同的数据到两个输出流。如果一项处理产生多个数据流，则这些数据流是相互排斥的；为了解决产生多个数据流的问题，数据流可一分为二，两个数据流也可合而为一。

OBKIS 的实体层标识了数据实体的类型及其之间的关系，包括实体类型、关系、连通性、实体标识和描述。实体类型 E 包含 OBKIS 数据层中的参与者 A，还可以是资源 R 或交易。关系 R 是两个实体 R(Ea，Eb)之间存在的关联。连通性是一个实体与另一实体发生关系的次数，包括一对一、一对

多和多对多三种类型，可分别表示为 $R(R_a^1, E_b^1)$、(R_a^1, E_b^m)和(R_a^m, E_b^m)。实体标识和描述(I&D)用以标识和描述每个实体的出现。

OBKIS 的动态对象层由对象组成。对象(或类的事例)可被定义为具有行为的实体。类是一组对象，这些对象具有相似特征、公共操作和统一行为。继承性是码共享机制。类的子类继承了超类的数据结构和行为，但是含有超类所不具有的特定操作和属性。不同的对象在 OBKIS 对象层中具有不同角色或身份。行为可以是程序代码中的操作，其中这些操作在面向对象的程序语言或面向对象的数据库管理系统中被编码为方法或函数。人的活动或环境变化造成了状态变化。动态对象运行其操作，这些操作对该对象所特有的数据进行存取。利用对象的数据来实现数据库中存储的方法代码。为了响应环境变化，理解对象模型的注释，对于将动态对象编档来说是重要的。对象模型的注释在图 6 的右下方示出。第一个域显示了类或对象的名称，第二个域表示属性，第三个域包含对象的行为。具体来说，第三个域包含了在面向对象程序中将被开发成为函数的信息。在对象模型中说明了对象之间的关系，使所含交互的类型一目了然。OBKIS 动态对象层的两大特性即多态性和继承性将在具体实施例中将进一步被阐释，其中：多态性是指“在不同形式下出现”，可描述为具有相同名称和含义的多个实体的行为，但具有不同的程序代码以实现相同的目标；继承性则是共享相似特征的机制。

值得一提的是存取和交换各方(比如销售商、消费者和制造商)数据的能力。打破各方之间的屏障对于协作是有益的，可确保通畅的信息流。一旦关系数据库中的数据已被对象化，在对象模型中将这些数据结构化，从而为产品信息标记语言(PIML)的应用铺平道路。PIML 包括三个部分，即扩展标记语言(XML)、扩展样式表语言变换(XSLT)和脚本语言，提供了通用的产品信息交换标准，使“知识工人”获取充分和准确的信息。具体来说，PIML 主要应用于产品设计，涵盖了与制造过程和工程分析有关的信息。PIML 有助于将数据变换成三种不同格式，比如：用于其他系统的 XML；用于供应商和消费者的超文本标记语言(HTML)；以及用于基于知识的系统的文本格式。

范例库的主要数据源来自于关系数据库(例如 Microsoft SQL 服务器)。异质数据库只能接受中性格式下的数据交换，而 XML 正是一种能被多数数据库支持的交换标准。为了形成关系数据库与范例库之间的连接，关系数据库

的数据首先被转换成 XML，由于数据库的解译样式与范例库大相径庭，所以需要在查询交易或数据变换中借助扩展样式表语言变换(XSLT)，XSLT 支持编程流控制，这有助于对数据应用这些规则，然后进而将这些数据变换成基于知识的系统所需要的知识。PIML 读取 XML 源文件和关联 XSLT 样式表。PIML 将 XML 文件及其关联 XSLT 文件解析成节点树，每个节点对应于 XML 文件元素或属性。按照 XSLT 的规范，将 XSLT 变换应用于源树以产生结果树。这些结果树被序列化为输出文件，分别是 XML、HTML 或其他文本格式。换而言之，在关系数据库中的数据已被作为 XML 文件输出之后，相关联的 XSLT 被读取和解析到源和样式表树中。XSLT 中声明的模板规则被应用于源树中的每个匹配 XML 元素，模板规则的应用可产生结果树。一旦结果树被序列化为字符流，该 XML 流就被输入到范例库。

由于 XML 含有标记批注的元数据，HTML 比 XML 更易于呈现，授权方能够在更熟悉的格式(HTML)下查看数据。XSLT 还可对能够导出到基于知识的系统的文本格式数据进行变换，因为中性格式的数据可作为用于基于知识系统的数据源。然后，已被对象化和变换为 XML 的关系数据库中的更新数据变成了知识存储库的宝贵资源，使用者可经由用户接口通过接口引擎来利用这些知识。该接口引擎提供了推理能力，为各种问题提供建议。当过去相似的范例适用于产品开发活动时，在范例推理(CBR)中可得到推断结果。

简单的决策逻辑流可利用 XSLT 来编码，XSLT 是具有流控制的编程语言，含有诸如<xsl:if>、<xsl:for-each>、<xsl:choose>等标记。这些标记帮助设计师提取显式知识。为了实现运行决策的复杂逻辑流，XSLT 允许 VB 脚本、Java 脚本、C++与 XML 一起操作。对象的动态行为是通过加载 Java 脚本和 VB 脚本来执行的。

图 2 中示出了按照本发明的用于异质环境之间信息交换的自适应程序调用方法的流程图。

首先，根据产品市场的变化，例如消费者对产品的需求，建立新产品的数据模型，并向企业数据库输入新的产品模型数据。

然后，利用面向对象技术，例如面向对象的编程语言，将数据库中的产品模型数据转换为对象模型，并进一步转换为扩展标记语言(XML)文件。

为了让其他系统、合作伙伴、知识库共享数据库中更新的产品数据，利

用扩展样式表(XSL)语言，将 XML 文件变换为中性数据格式的数据流。

进而，将中性格式下的数据流导入到企业知识库中，实现企业数据库的数据与企业知识库的范例之间的数据映射。

经由范例推理接口引擎，从企业知识库中取回范例，作为新产品的设计参考方案。

而且，在对产品模型数据进行对象建模时，对产品开发过程中与产品数据处理有关的操作、具有行为的实体等进行建模。

当建模后的对象启动消息时，在消息签名中指定待执行的程序名称和待包含的参数，每项程序用于实现一类对象的操作，这些操作可在 XML 格式下通过多种程序来实现。

上述的对象模型具有多态性和继承性，并且包括名称域、属性域和行为域，如图 6 中所示。

如图 1 中所述，上述的中性数据格式包括用于其他信息系统的扩展标记语言格式、超文本标记语言(HTML)格式和文本格式。

除了利用 XSL 语言进行 XML 文件的变换之外，还可利用 VB 脚本、Java 脚本、C++，对 XML 文件进行变换，实现动态的信息处理能力。

为了保持范例知识库的动态性和有效性，为每个范例设置有信心值参数，该信息值参数表示取回的范例解决产品设计问题的成功率，并且企业知识库的内容和范例的数量均随着产品设计师经验的增多而动态增长。

下面以电子产品制造商 GSL 为例，进一步阐释本发明的具体实施例。在 GSL 中，商业运作需要经验丰富的产品设计师每年处理数以百计的产品规格。以往需要设计师具备大量经验方可有效地使用专有和分离的数据库系统和平台中存储的数据。旧程序系统需要较高成本和较长时间才可开发出新产品，影响了产品开发的进度，GSL 逐渐意识到 XML 可作为通用的数据标准，进而利用一种全能的解决方案，产生具有强大协作性能的丰富规范，这就是对象技术和 PIML 的结合，用以产生新产品规范和评价设计变化的效果。每个设计变化的实施是按照内部和外部因素来进行的，这里有效的数据传输是必要的。与各式各样的后端企业数据系统的无缝集成，确保了与不同部门的处理有关的所有信息能够在产品设计过程中被快速存取和同化。GSL 电子产品开发生命期中的各层细节如下所述。

首先参照图 4 说明数据层。数据层标识了产品开发中的数据变换，通常首先包含通过市场调查来收集消费者偏好。数据流图(DFD)用于将系统的概念设计变成逻辑设计，着眼于产品开发工艺在各种功能部门之间的信息流。消费者在 DFD 中被标识为“参与者”，向 GSL 的营销部分提供消费者需求。从市场调查中发展出产品规格，用于优化产品设计。按照该产品规格，机械工程师对 CAD 库中存储的 PCB 绘图进行开发。在 PCB 绘图的最终版本发行之前，PCB 设计规格需要得到工程师的批准。按照该 PCB 绘图，工具制造者通过参照刀具 CAD 制造出模具、夹具和定位器，这些工具将用于新产品的生产。在达到预定的质量标准之前，不断进行修改。一旦产生的测量结果被接受，开始执行试运行。如果试运行是成功的，则进行批量生产，并将成品分发到零售商。图 4 示出了从概念设计、设计检查、工具制作、试运行、规模生产直至最终到达零售销售的数据处理。

然后参照图 5 说明实体关系层。产品开发生命期包含大量活动和任务。图 5 仅描述了图 4 所示整个开发生命期中的产品定义、设计和开发阶段。实体-关系(ER)图标识了物理和概念实体的静态和结构关系。以产品规格为基础的设计检查可被分为可行的设计变化和不可行的设计变化。图 5 示出了设计检查的结构关系，应用了 ER 图的不相交规则，因为超类型的事例不可能是两个(或更多)子类型的成员。为了修改既往设计，设计师可参照产品文件，比如技术分析报告，该报告包括印刷电路板图和组装图。如果超类型的事例同时是两个(或更多)子类型的成员，则可应用 ER 图的交迭规则。

再参照图 6 说明动态对象层。为了在早期的产品开发期间处理产品设计检查，图 6 借助对象“设计检查”的“删除设计变化”行为，示出了对象模型的特性，即继承性和多态性。“设计检查”被评价为“可行的设计变化”和“不可行的设计变化”。超类(设计检查)和子类(不可行的设计变化和可行的设计变化)之间的关系以三角形箭头来连接，其示范了对象的继承性。超类的公共行为包括“添加设计变化”和“删除设计变化”，被继承到“可行的实际变化”和“不可行的设计变化”。“设计检查”的属性包括“检查编号”、“零件编号”、“检查类型”、“检查细节”和“注释”，也被继承到“可行的实际变化”和“不可行的设计变化”。由于这些行为和这些属性都被超类继承，所以无需重写这些子类的行为和属性。然而可发现，行为“删除设

计变化”出现在子类“可行的设计变化”之下。这是因为超类“设计检查”中的“删除设计变化”表明了不同的情形，但是具有相同的声明。为了实施这些设计变化，需要各部门分别实施相同操作，也就是将数种方法组合起来，以执行一个操作，此即为多态性的延伸概念“适时可重构方法调用”(TRMI)，其通过采用对象技术，构造出集成的信息系统。这些被组合的方法例如包括：1．评价生产能力；2．核对生产计划表；3．进行决策(接受或拒绝设计变化)；4．通告生产和材料控制部门；5．与营销部门进行协商；6．实施多个设计变化过程；7．删除设计变化；等等。

当没有足够生产能力用于该设计变化时，需要删除新的或修订的设计特征，该行为在超类中示出。“设计检查”和“删除设计变化”被集成到子类“不可行的设计变化”。因此，在序列 1-2-3(拒绝)-5-7 下执行这些方法。

当出现可行设计的实例时，在设计变化处理已完成之后运行“删除设计变化”操作。因此，在序列 1-2-3(接受)-4-6-7 下执行这些方法。

行为“删除设计变化”的相同声明具有相同的含义，在以往的过程程序中不被允许，因为无法区别“删除设计检查细节”的语义。然而，面向对象程序允许“多态性”。操作“删除设计变化”的子类版本超越了来自超类的版本，因为该系统更倾向于使用来自子类的操作版本，对于该子类及其派生类都是如此。

在图 4 中，产品开发的数据流图(DFD)示出了制造商 GSL 产品开发中的不同过程，包括示出调查、数据处理、产品规格接受、PCB 开发、设计检查、工具开发和修改。在这些过程之中，将具体地探讨设计检查。

图 5 示出了静态的数据结构，图 6 示出了与 GSL 的设计检查有关的动态行为。下面将说明 PIML 如何利用图 6 中的对象模型，在基于知识的系统和数据库之间交换数据。

假设营销部门通过电子邮件收到了变化请求，营销主管在数据库中输入数据。GSL 的产品设计变化包括尺寸、地点和材料等的变化。SQL 服务器中的数据被变换为 XML。

工程师需要查阅修订后的规格，并重新安排生产计划。他们经过企业网浏览 HTML，这里 HTML 是 XSLT 和 XML 产生的输出文件。有经验的工程师判断这些设计变化对于序列过程是否有影响，同时从范例推理(CBR)系统

中获得可能的建议。PIML 的 XSLT 是具有数据流控制的编程语言，包含<xsl:if>、<xsl:for-each>、<xsl:choose>等标号。如果新的产品设计需要改变材料，则工程师能够从 CBR 系统中发现可能的解决方案。

XML 中的数据包含“检查类型”、“模型编号”、“零件编号”、“检查细节”和有经验的工程师的推荐，可利用 XSLT 将 XML 导出到基于知识的系统。通过数据映射，将不受具体平台或操作系统约束的中性格式文件引入到范例推理(CBR)系统。联系 PIML 和 CBR 的数据映射包含这样的数据映射结构，其用于将公司数据库的每个表格映射到知识存储库。

工程师取回与新的设计变化相似的以往记录。他们在发出变化请求之后，还能够利用 PIML 发现诸如延迟和机械缺陷等结果。图 7 示出了利用 CBR 函数封装后的 PIML 界面。例如，可使用一种商用 CBR 软件 Kaidara，它可通过取回以往的相似范例，帮助解决新产品的设计问题。

XSLT 允许 VB 脚本、Java 脚本、C++与 XML 一起使用，从而通过特定的句法，实现更为动态的 PIML。通过 Java 脚本增强后的系统，与生产模块进行动态和双向的交互，获得必要的工程信息，比如额定生产力、标准产量和作业优先级。PIML 之内的 Java 脚本可提供对关键数据源的存取，该存取与上下文有关，这些数据源用于实现前面所述的多态性。

OBKIS 将关系数据库重构成面向对象的格式，这种格式带来了更为有效、高效、稳定的数据和知识集成系统，其具体实施包括三个步骤。第一步骤是通过对象技术将关系数据库公式化，并标识这些对象和类。一旦定义了类，还需要识别相对属性以及行为，并建立类之间的关联关系。关系数据库和面向对象技术之间的一个关键区别在于，对象建模中的多对多关系需要被正规化。第二步骤是将对象嵌入到 XML 内容中。每个对象可被简单解译为 XML<tag>实体。除了在 XML 中示出静态的元数据之外，还通过脚本语言对于对象的动态行为进行编码，并允许使用者通过 XSLT 来定制数据输出的显示格式。应当对使用者的习惯和术语/词汇进行研究，以使 PIML 与使用者顺利地进行交互。第三步骤是域测试，并收集使用者的反馈。PIML 的原型表明了 XML 技术和无缝数据交换的适用性，并且更为简化和廉价。

CBR 属于人工智能，与模拟推理有关。当构建范例库时，需要保持足够多的范例用以解决问题；另一方面，范例库也不要过大，否则影响取回时间。

需要设置范例库的最大容量，并记录通过取回的范例来解决问题的成功率。另外，由于日益积累越来越多的范例，从范例库维护的角度看，还需要关注“范例能力”。随着商业环境的变化，范例也不断发展，需要定期检查原始范例解决新问题的能力。尽管范例推理能够利用取回的范例来解决新问题，但是遇到意外变化时仍然需要人工智能进行快速响应。在连续变化的竞争性环境中，取回的范例的信心值将低于正常情况。信心值能够反映取回的范例是否支持解决方案。可从相似性、典型性和偏差性等方面来度量信心值，也就是：取回范例和查询范例之间的相似性、查询范例和存储范例所共有的属性数量、每个所选属性的估计值与查询的估计值。其中，信心值与相似性和典型性成正比，与偏差性成反比。对于范例取回来说，相似性在信心水平中扮演关键角色，属性的权重是“最近邻近(NN)”算法的主要参数。因此，应当仔细调节属性的权重。

OBKIS 能够提供敏捷性、自学习能力和可定制等优点。制造业近十年发生转变的范式已经变得越来越依赖于信息系统的快速响应和敏捷性。通过利用面向对象技术，能够按照信息所描述的现实世界的对象来组织、操纵和处理这些信息。对象基本是“自我包含”的单元，它们封装了用于进行作业的各种信息和算法。继承性的引入降低了产品结果的复杂性，因为在超类中定义了公共操作。由于新的消费者需求可能触发变化，所以需要对产品开发工艺进行修改。可被初始化和在特定子类下进行单独操作的 TRMI 使得企业能够及时处理意外事件，因而特别地适用于动态市场。

为了适应市场变化，要求企业信息系统能够存取和操纵数据，从而知识表示可得到更新数据的支持。通过引入逻辑或启发式规则，修改以往范例的解决方案以解决新问题，这表现了系统的适应式学习能力。

由于许多未知因素遍布于整个设计过程中，所以将产品设计看成是反复的过程。PIML 能够作为一种产品信息标准，通过 XML 模式来操纵产品数据。除了在静态数据结构中利用 XSD 来表示 XML 之外，还描述了关于如何进行动态行为的 OBKIS 的另一特征。对唯一的类进行定制，从而每个类具有其自己的行为，以在各种条件下不同地反应。

简而言之，PIML 是按照如下方法来建立的。

1．建立产品开发模型，包括：识别消费者需求；定义设计输入要求；

定义产品开发工艺；标识产品开发工艺的工作流，这些工作流使得产品开发工艺相互联系；

2．将模型转换成对象模型，包括：标识对象、行为(动态)和属性(静态)；标识关联性(静态的继承性、动态的多态性)；构造对象模型；

3．将对象模型变换为 XML 模式，包括：识别 XML 模式与对象模型之间的类比性；构造 XML 模式，需要考虑数据域、属性的数据类型、元素的最大/最小长度等参数；利用 PIML 特征来对 XML 数据进行建模，这些特征包括对象/类之间的关联性、继承性、多态性等。

在上面的描述中，本发明的 OBKIS 将对象技术与涵盖 XML 的 PIML 和范例推理手段相结合，增强数据同化的有效性，从而在产品生命期循环中支持产品数据管理。一般来说，OBKIS 的显著特征包括：(1)敏捷性：用于处理产品开发早期阶段出现的意外变化；(2)自学习性：用于变换异质系统之间的数据；(3)可定制性：用于从根本上构造数据方案，以处理不断发展的产品开发工艺。在软件开发期间，对象技术起到工具性作用，用以实现该系统的基本特征和功能。除了对象技术之外，还可并入其他新兴技术，用于形成系统框架的主干，包括：(i)TRMI，用于在不同情况下重构方法；(ii)XML，用于与平台或操作系统无关地进行数据交换。

本发明的系统和方法开发出将数据库与知识存储库相连接的通用模型，以便将更新数据从公司数据库存取到知识存储库。波动的市场和变化的消费者需求造成了频繁的设计变化，这些变化可通过该系统的动态层来反映。与 PIML 相联系的推理引擎能够增强整个系统的响应性和适应性，达到短时间接近市场需求和占有大的市场份额的企业发展目标。

跨平台的数据交换方式能够有助于建立有益的开发环境，从而利用对象计算来管理工程变化和改进产品开发活动。TRMI 的特征在于为相同的消息调用各种行为，并超越预定义的继承操作，从而能够在反复的产品设计步骤中使灵活的关联过程公式化。具体来说，OKBIS 提供了通用的产品信息交换标准，该标准使得知识工人能够获取充分和准确的信息。本发明基于中性数据，实现了无缝的数据交换，该中性数据文件将公司数据库变换为范例存储库。PIML 包含了用于产品开发的特定 XML 集，利用了结构化的 XML 模式，提供易于理解的句法和自定义的标记语言，从而在产品数据交换方面满足特

定需要。

尽管上文已参照附图详细地描述了本发明的具体实施例，但是本领域的技术人员应当理解，这些附图和描述仅是说明性而非限制性的，本发明的范围并不囿于所示实施例，而是由所附权利要求书来确定。

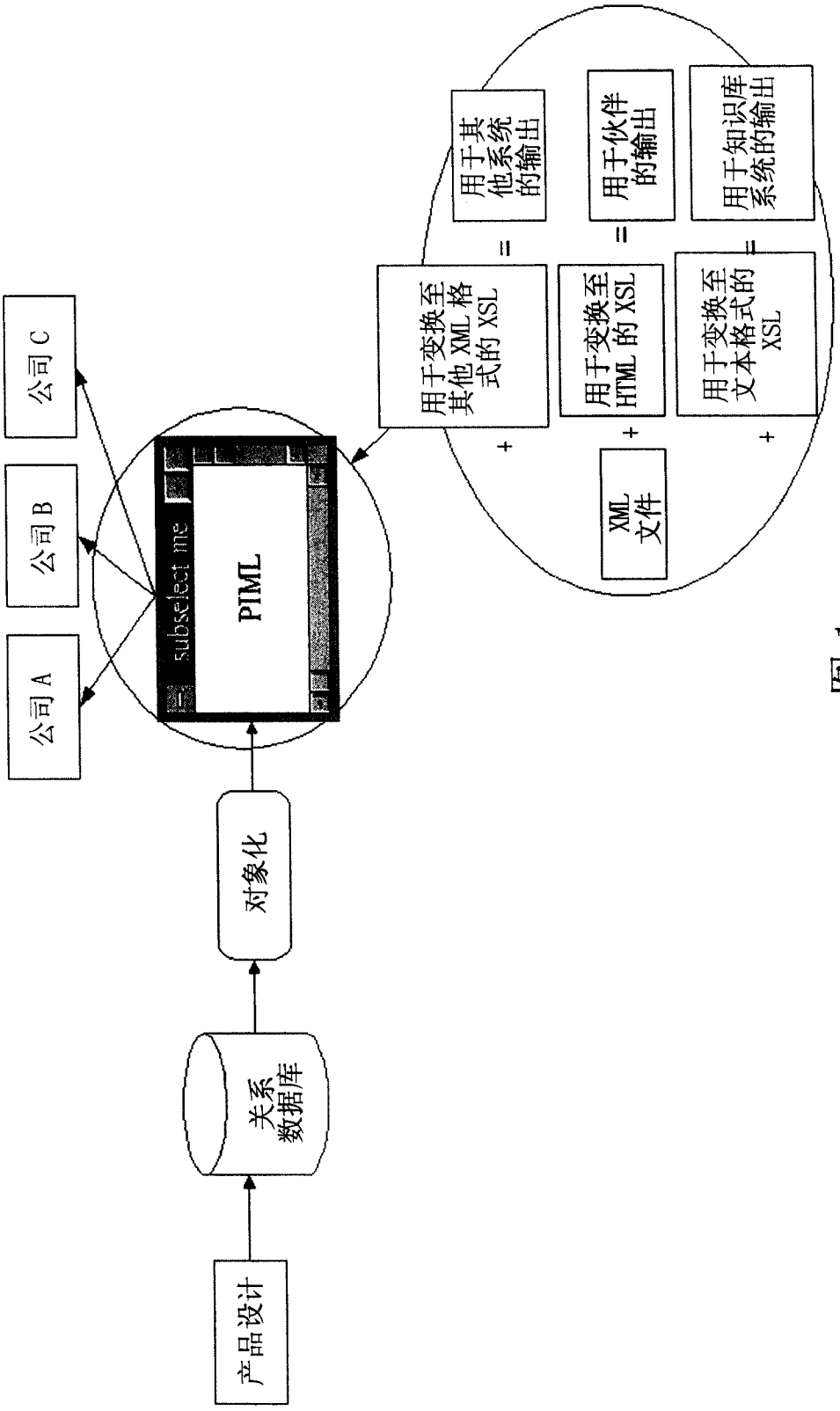

产品设计
关系数据库
对象化
subselect me
PTML
公司A
公司B
公司C
用于变换至其他 XML 格式的 XSL
用于变换至 HTML 的 XSL
用于变换至文本格式的 XSL
XML 文件
用于其他系统的输出
用于伙伴的输出
用于知识库系统的输出

图 1

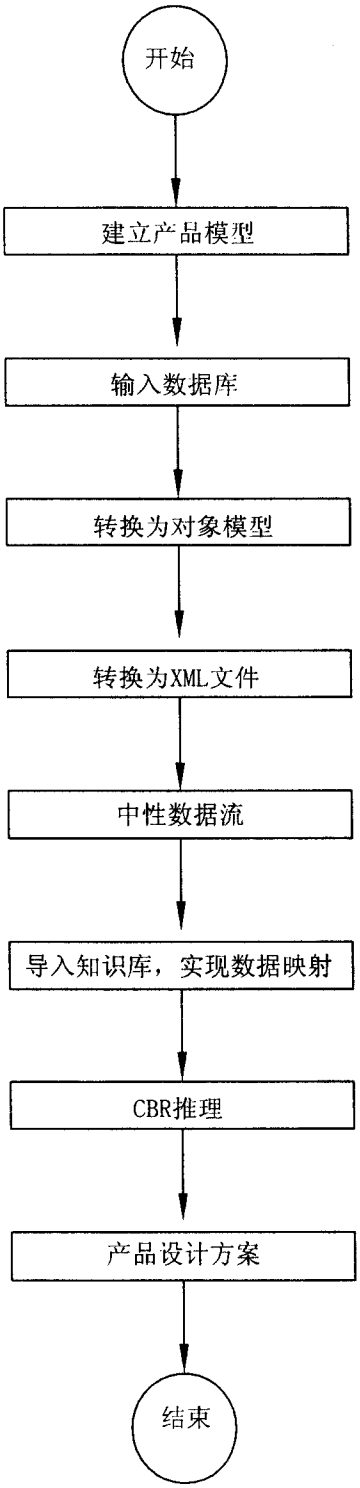

开始
建立产品模型
输入数据库
转换为对象模型
转换为XML文件
中性数据流
导入知识库，实现数据映射
CBR推理
产品设计方案
结束

图2

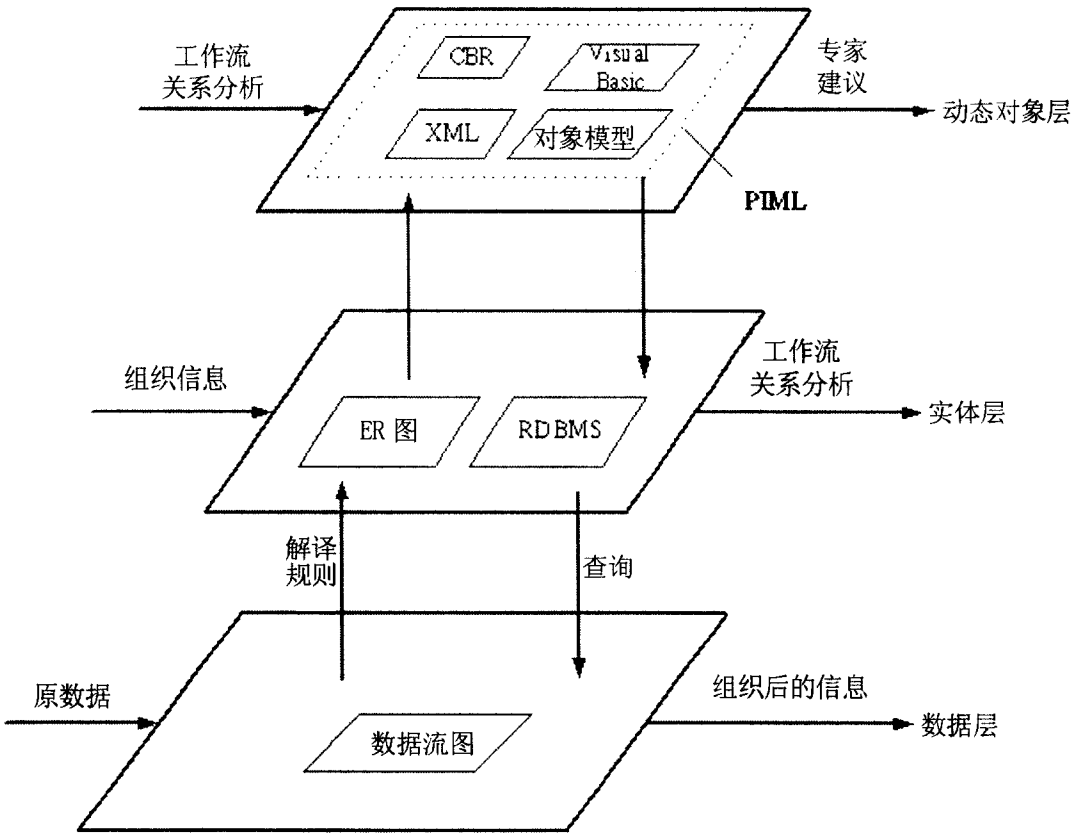

工作流
关系分析
CBR
Visual Basic
XML
对象模型
专家
建议
动态对象层
PIML
组织信息
ER 图
RDBMS
工作流
关系分析
实体层
解译
规则
查询
原数据
数据流图
组织后的信息
数据层

图 3

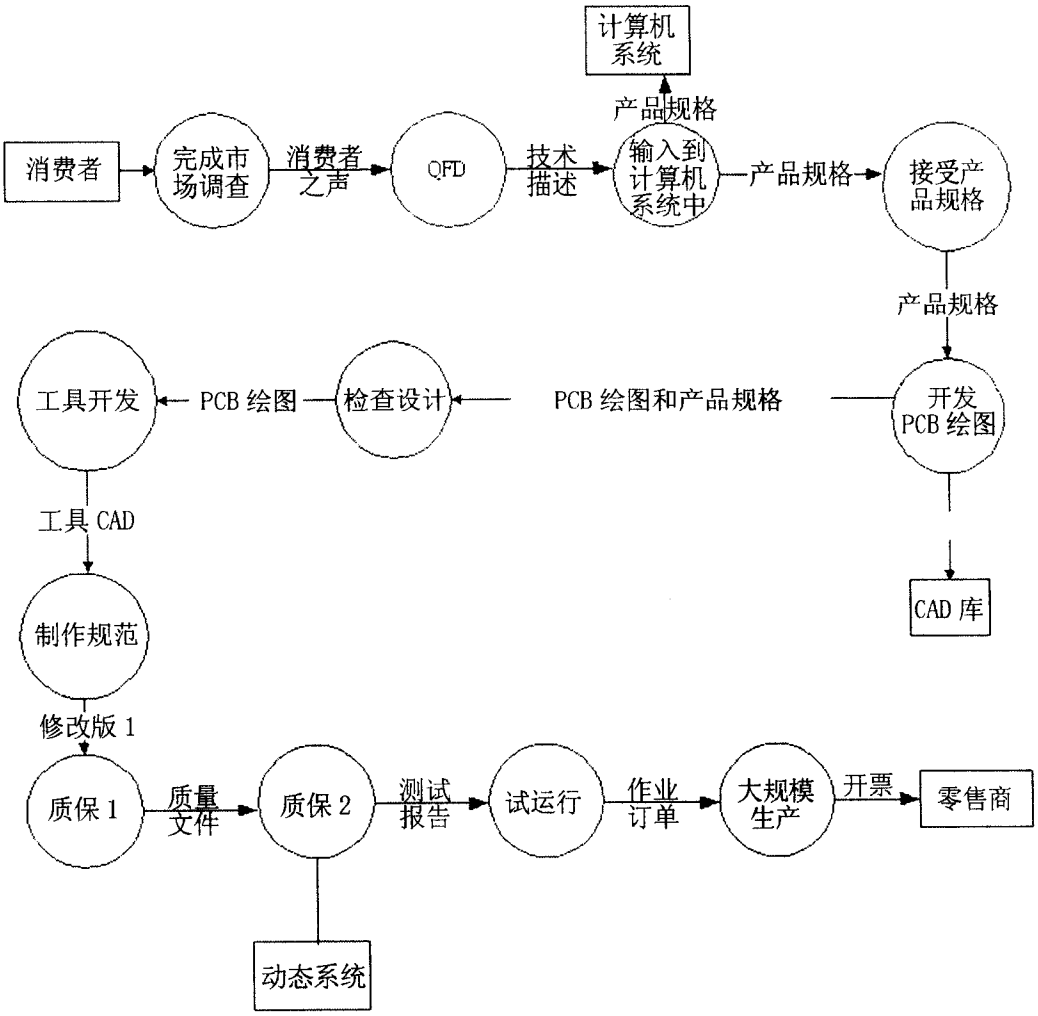

计算机系统
产品规格
消费者
完成市场调查
消费者之声
QFD
技术描述
输入到计算机系统中
产品规格
接受产品规格
产品规格
工具开发
PCB 绘图
检查设计
PCB 绘图和产品规格
开发 PCB 绘图
工具 CAD
CAD 库
制作规范
修改版 1
质保 1
质量文件
质保 2
测试报告
试运行
作业订单
大规模生产
开票
零售商
动态系统

图 4

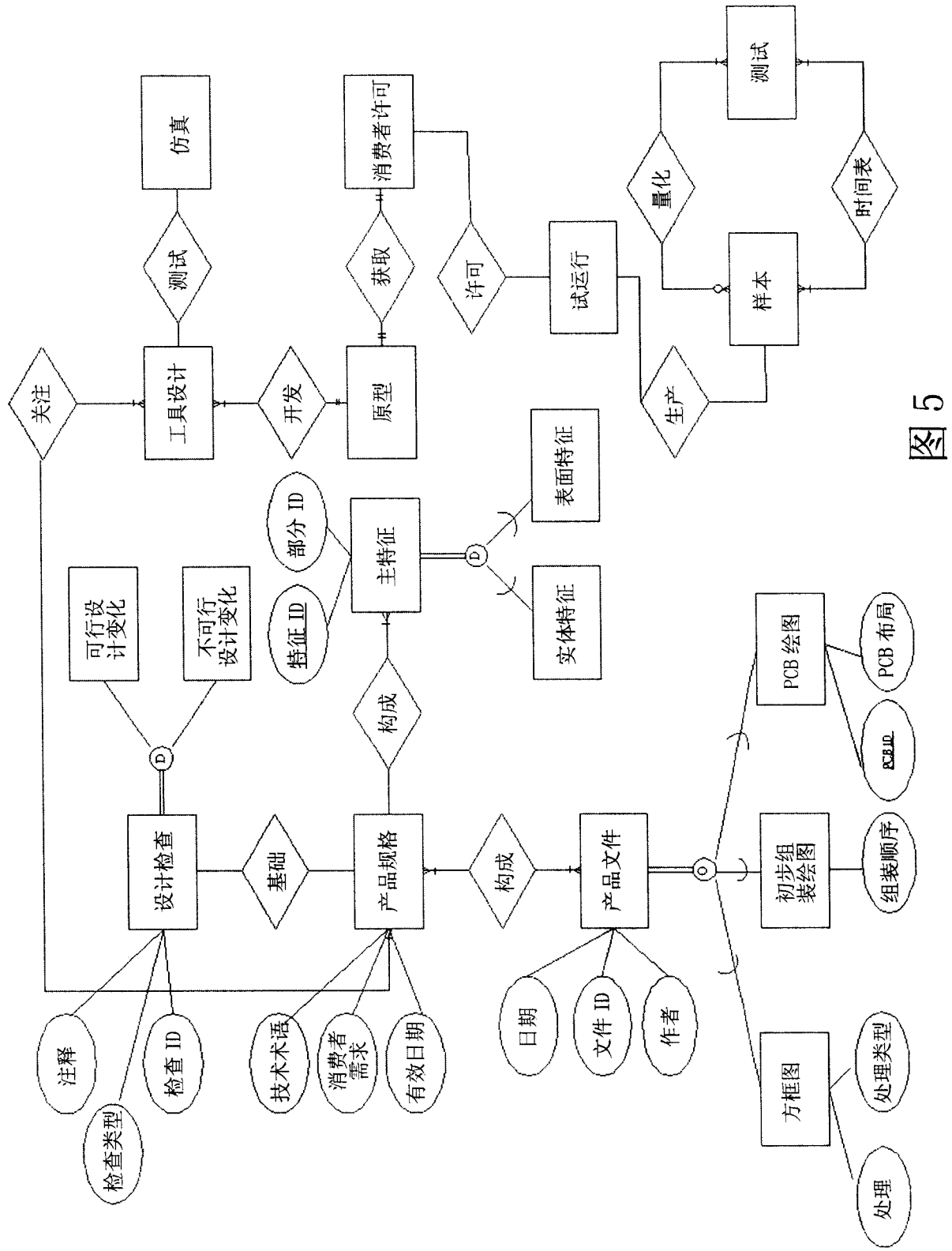

关注
工具设计
测试
仿真
开发
原型
获取
消费者许可
许可
试运行
生产
样本
量化
测试
时间表
设计检查
基础
产品规格
构成
产品文件
构成
初步组装绘图
PCB 绘图
PCB 布局
PCB ID
组装顺序
方框图
处理类型
处理
主特征
部分 ID
特征 ID
表面特征
实体特征
可行设计变化
不可行设计变化
注释
检查类型
检查 ID
技术术语
消费者需求
有效日期
日期
文件 ID
作者

图 5

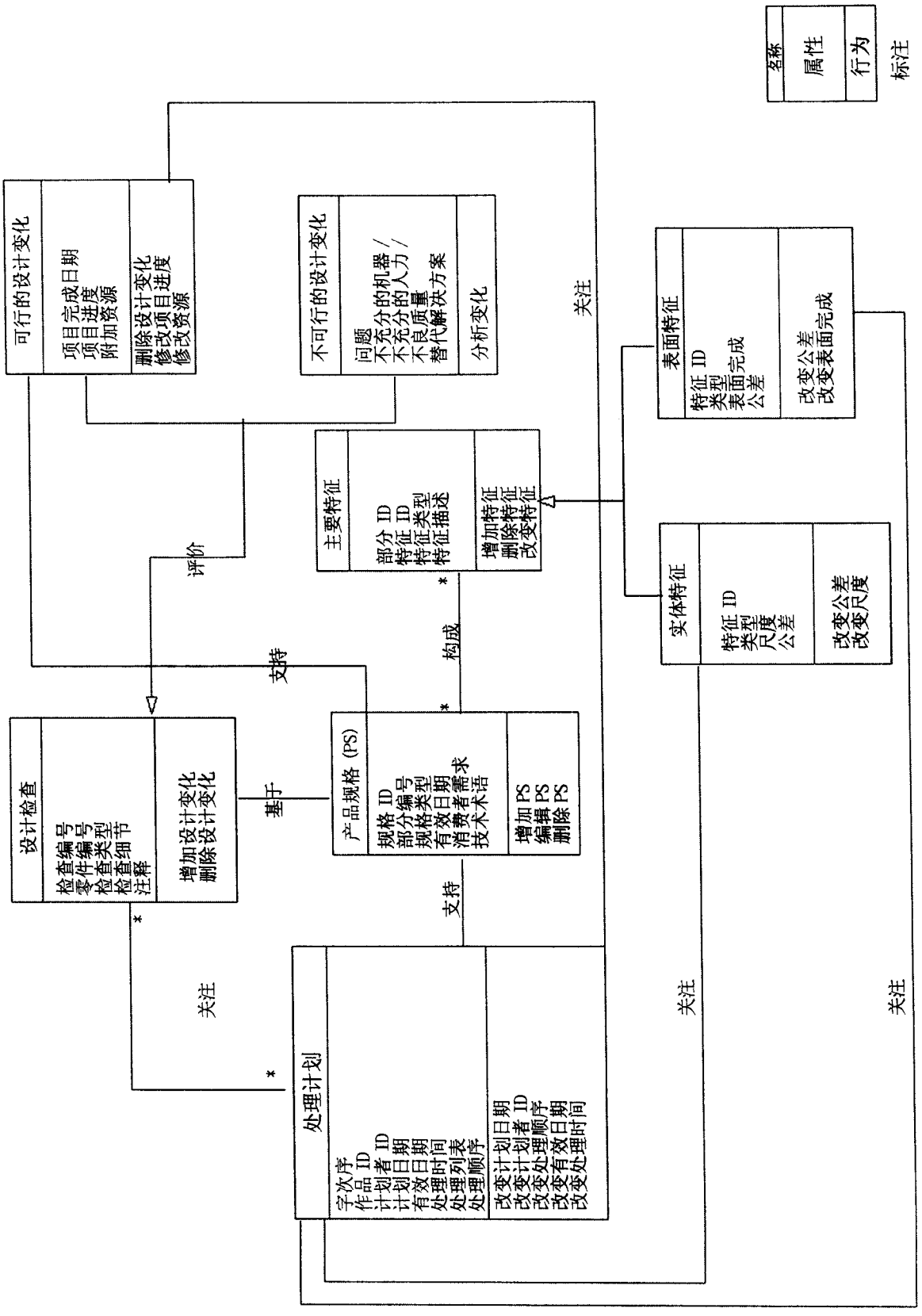

名称
属性
行为
标注
可行的设计变化
项目完成日期
项目进度
附加资源
删除设计变化
修改项目进度
修改资源
不可行的设计变化
问题
不充分的机器 / 人力 /
不充分的
不良质量
替代解决方案
分析变化
关注
表面特征
特征 ID
类型
表面完成
公差
改变公差
改变表面完成
主要特征
部分 ID
特征 ID
特征类型
特征描述
增加特征
删除特征
改变特征
评价
支持
构成
实体特征
特征 ID
类型
尺度
公差
改变公差
改变尺度
设计检查
检查编号
零件编号
检查类型
检查细节
注释
增加设计变化
删除设计变化
基于
产品规格 (PS)
规格 ID
部分编号
规格类型
有效日期
消费者需求
技术术语
增加 PS
编辑 PS
删除 PS
支持
关注
关注
关注
处理计划
字次序
作品 ID
计划者 ID
计划日期
有效日期
处理时间
处理列表
处理顺序
改变计划日期
改变计划者 ID
改变处理顺序
改变有效日期
改变处理时间

图 6

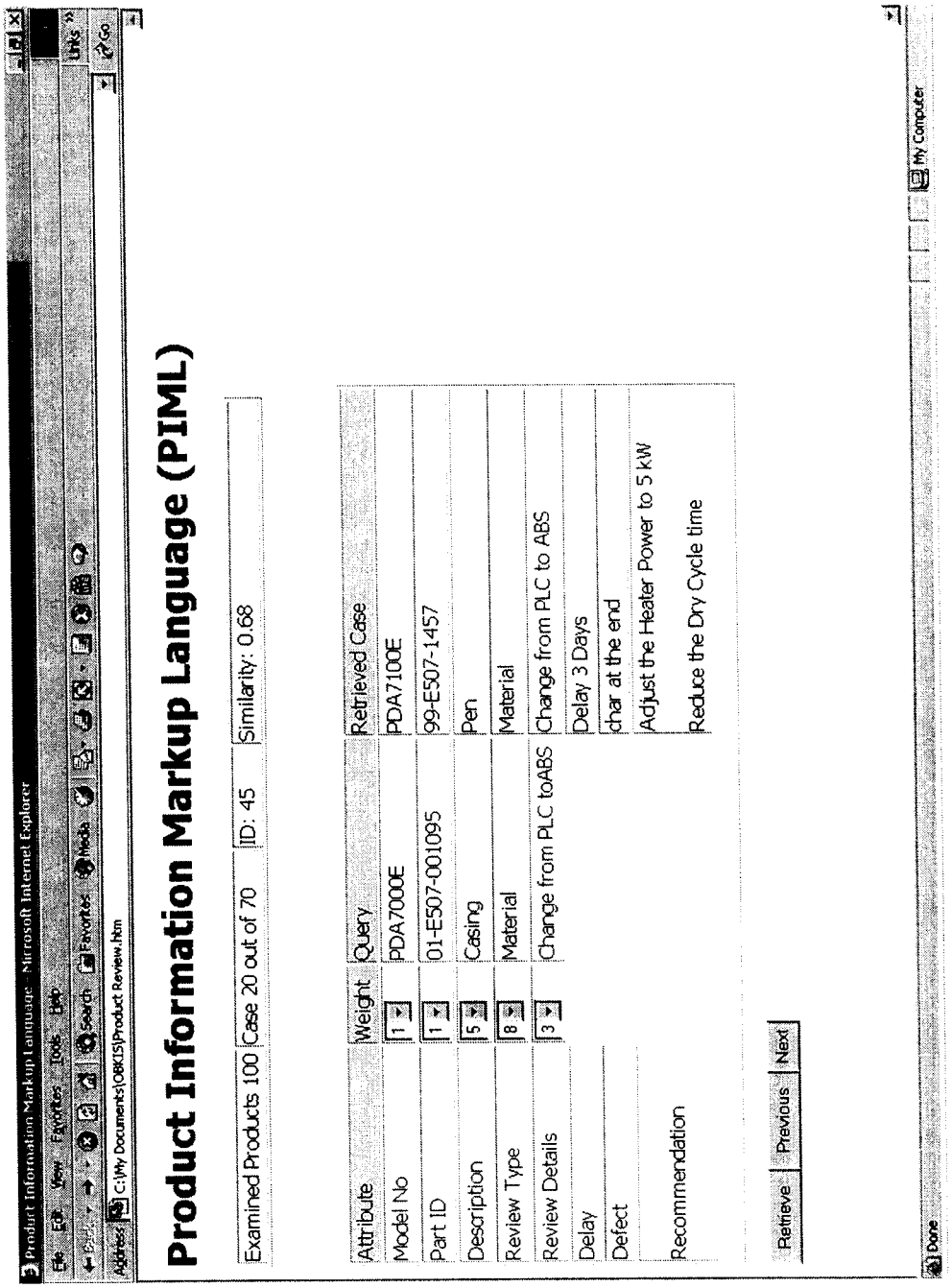

Product Information Markup Language (PIML)
Examined Products 100
Case 20 out of 70
ID: 45
Similarity: 0.68
Attribute
Weight
Query
Retrieved Case
Model No
PDA7000E
PDA7100E
Part ID
01-E507-001095
99-E507-1457
Description
Casing
Pen
Review Type
Material
Material
Review Details
Change from PLC toABS
Change from PLC to ABS
Delay
Delay 3 Days
Defect
char at the end
Recommendation
Adjust the Heater Power to 5 kW
Reduce the Dry Cycle time
Retrieve
Previous
Next

图 7