

SLeD 中 QTI 服务的实现

田建伟,余 丹,陈莘萌

(武汉大学 计算机学院,湖北 武汉 430079)

摘 要:当前 IMS Learning Design(IMS LD)规范中虽然整合了 QTI(Question and Test Interoperability)规范,但是在当前的播放器中却没有很好地整合 QTI 服务,这就阻碍了播放器和 QTI 之间的通信,从而不能很好地实现 QTI 服务。文中针对 SLeD 体系结构,在该体系结构中加入中间软件层,利用这个中间软件层实现了播放阶段 QTI 和播放器的通信。

关键词:IMS Learning Design; SLeD; QTI; Coppercore; 中间软件层

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2007)06-0005-04

Implementing QTI Service in SLeD

TIAN Jian-wei, YU Dan, CHEN Xin-meng

(Computer School of Wuhan University, Wuhan 430079, China)

Abstract: At present, IMS Learning Design (IMS LD) Specification integrates QTI (Question and Test Interoperability) specification, however the integration of the QTI service in the LD Player has not done very well, which hinder the communication between the LD player and the QTI player. This paper aims at implementing communication between the LD player and the QTI player through introducing a middle software layer in the SLeD architecture and system.

Key words: IMS learning design; learning design; SLeD; QTI; Coppercore; middle software layer

1 IMS 学习设计编辑器和播放器

IMS 的学习设计规范是建立在荷兰开放大学的教育建模语言(Education Model Language, EML)基础上的,加入了一种类似 HTML 的内容模型来支持行为。IMS 于 2003 年 2 月 13 日公开发布的学习设计规范 1.0 版本整合和扩展了现有的一些学习规范^[1],它提供了一种描述方法,说明学习者在学习环境中,以一定的顺序进行学习以达到预定的学习目标,它能够模拟多角色的教与学过程,也支持个别化的学习进程和大量在线教学模式的应用。与 EML 相比,IMS LD (IMS Learning Design)除了继承了 EML 的一些规范和特点之外,其优点是把学习活动和教材两者独立开来,这样的优势在于教材可以运用到不同的学习活动上,学习活动可以在多个教材上应用。这样大大加强了它们的重用性和共享度。

学习设计的文档编写不仅需要普通的编辑工具的支持来编辑一些学习素材,也需要专业的编辑器支持,专业编辑器可以编辑出一个符合学习设计规范的学习

单元,当前的专业编辑器主要是 Reload、Copper-Author,这些编辑器让使用者根据不同的目的和需求编辑自己需要的学习单元,其次应用学习设计需要一个符合其规范播放器,通过播放器运行一个学习单元,它可以协调学生和老师之间的关系,跟踪他们的进程,在合适的时间交付合适的学习行为和资源。Coppercore 引擎的研发成功是一个伟大的进步,它除了验证学习单元的合法性之外,它还可以作为一个核心引擎,它提供了三个 API 接口,其他开发者可以在这个引擎基础上研发出其他的专业播放器。

Coppercore 是第一个能运行 LD 学习单元的播放器,更加重要的是他让其他致力于开发学习设计播放器的开发者看到一个播放器是如何表示和展现学习设计规范了定义的一些行为特征。Coppercore 每一个学习单元在进入学习设计播放器之前都要经过 Coppercore 引擎的验证,只有通过验证的合法的学习设计单元播放器才能运行,实现引擎和播放器的功能性分离。从图 1 可以看出, SLeD 在 Coppercore 的基础上增加了 Web Services 技术, SLeD^[2] 播放器和引擎之间依靠 Web Services 技术进行通讯从而加强了这种分离。这种方法让其他开发者可以实现一些功能,例如:Reload 播放器运用这种技术实现了图形界面的验证和测试。

收稿日期:2006-08-22

作者简介:田建伟(1982-),男,四川乐山人,硕士研究生,研究方向为分布并行处理和新型 E-learning 平台构件;陈莘萌,博士生导师,研究方向为分布并行处理。

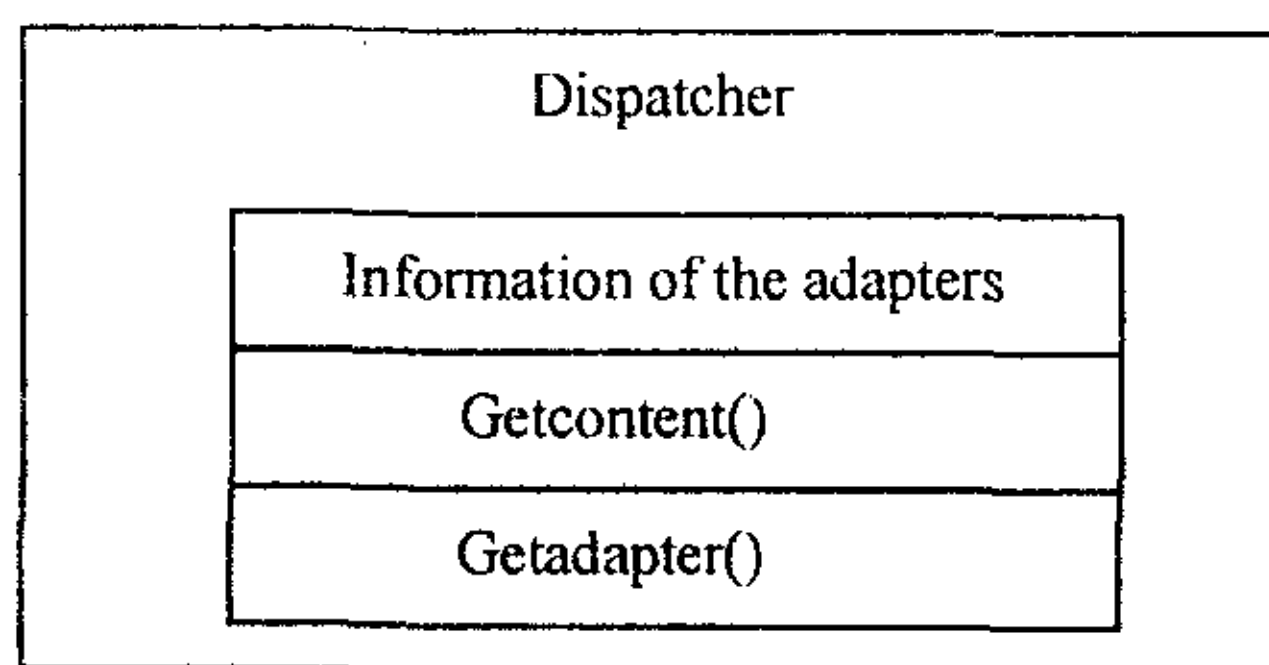


图 1 分发器结构

2 当前的播放器存在的问题

QTI 规范给出了练习(项)数据和测试(评估)数据的基本表示结构。QTI 规范应用 XML 定义,该规范具有可扩展性而且易于实现的优点。QTI 规范主要解决目前练习/测试数据的独享性和缺乏开放性问题。该规范使得测试/评估数据能够在不同的学习管理系统、内容开发系统和资源库等之间进行通信^[3]。

IMS 学习设计规范虽然整合了 QTI 规范,但是当前的播放器却没有整合 QTI 播放器,当一个用户要进行一次测试时,播放器不知道什么时候交换或改变什么属性,这阻碍了播放器引擎和 QTI 引擎之间进行双向通讯,播放器本身不能很好地解析 QTI 内容,所以进行测试时学习设计播放器需要一个 QTI 播放器。这对用户来说就要求在两种用户界面之间不断进行切换,这会给用户带来混淆。

现在整合 QTI 存在的难点在于在运行阶段播放器解析一个学习单元时,必须能探测到这个学习单元中是否整合了 QTI 内容,然后能调用相应的播放器来解析相应的文档内容,这时又产生了另外一个问题:学习单元里的一些事件的发生与否可能依赖于这个 QTI 测试的结果,这就要求学习单元里的一些属性机制必须和测试结果相关联。

3 解决方案

(1) 编辑阶段。

IMS LD 学习单元中的属性值可以影响整个学习过程,比如在条件里通过判断属性的值可以影响环境的可见性或学习行为的顺序。如下面例子的文档,通过判断属性“depend-on-test-result”的值,如果属性值小于 3,测试就不通过,如果大于 3,测试通过,接着学习第五课:

```

<imsld:properties>
  <imsld:loppers-property identifier="depend-on-test-result">
    <imsld:title>The result for the test carried out as the first
step in a learning flow</imsld:title>
    <imsld:datatype datatype="integer"/>
    <imsld:initial-value>0</imsld:initial-value>

```

```

</imsld:loppers-property>
</imsld:properties>
<imsld:conditions>
  <imsld:if>
    <imsld:less-than>
      <imsld:property-ref ref="depend-on-test-result"/>
      <imsld:property-value>3</imsld:property-value>
    </imsld:less-than>
    </imsld:if>
    <imsld:then>
      <imsld:show>
        <imsld:learning-activity-ref ref="do-the-test-again"/>
      </imsld:show>
    </imsld:then>
    <imsld:else>
      <imsld:show>
        <imsld:learning-activity-ref ref="do-the-lesson5"/>
      </imsld:show>
    </imsld:else>
  </imsld:conditions>

```

在 QTI 文档里也有类似于 IMS LD 中属性的元素叫变量,用来记录测试的结果。如例子文档中的“TEST_SCORE”:

```

<imsqti:outcomes-test>
  <imsqti:outcomes>
    <imsqti:decvar varname="test_score" vartype="Integer"
defaultval="0"/>
  </imsqti:outcomes>
</imsqti:outcomes-processing>

```

在编辑阶段 IMS LD 和 QTI 的整合是通过属性元素的命名机制来实现的^[4]:在属性的名称的基础上加上相关联的变量的名称。举以上的 IMS LD 文档和 QTI 文档为例,如果要想 QTI 测试结果“test_score”影响 LD 文档里的属性“depend-on-test-result”,那么就把属性名称命名为“depend-on-test-result.testscore”。这样在运行阶段就可以把它们看成是共享变量,指向同一个值。

如果教师需要在运行学习单元时首先做一个测试,编辑一个学习单元时就用 QTI 编辑器编辑一个 QTI 的 XML 文档。在学习设计编辑器里嵌入一个 QTI 编辑器,QTI 编辑器的进入是通过先进入 LD 编辑器,然后 LD 编辑器里的接口进入到 QTI 编辑器。下面介绍如何编辑一个 QTI 文档。

比如用户需要一个名为“Assessmentitem”的测试文档,这个测试内容为一个或几个问题和练习组成(item),而一个问题或练习有一个问题呈现和结果处理组成。问题呈现包含了要呈现的问题或练习的内容

(可以是文本型、按钮等方式)的所有信息,结果处理定义了答完问题后计算成绩的方法。结果处理由一个初始值和一个判断器组成。判断器里有判断逻辑:如果返回值等于 X 或是空值就给初始值重置一个值,或是在初始值的基础上加上一个值。

下面的 QTI 内容文档是一个单项选择的例子,题目是“where is YaoMing from?”,从三个选项 China, America, Japan 中选一个:

```
<assessmentItem identifier = "choice" title = "where is YaoMing from?">
  <responseDeclaration identifier = "RESPONSE" cardinality = "single" baseType = "identifier">
    <correctResponse>
      <value>A</value>
    </correctResponse>
  </responseDeclaration>
  <outcomeDeclaration identifier = "SCORE" cardinality = "single" baseType = "integer">
  </outcomeDeclaration>
  <presentation>
    <choiceInteraction responseIdentifier = "RESPONSE" shuffle = "false" maxChoices = "1">
      <simpleChoice identifier = "A">China</simpleChoice>
      <simpleChoice identifier = "B">America</simpleChoice>
    </choiceInteraction>
    <simpleChoice identifier = "C">Japan</simpleChoice>
  </presentation>
  <responseProcessing template = "www.qtiprocess.com"/>
</assessmentItem>
```

在编辑器里了嵌入一个 QTI 编辑器后,因为编辑出来的都是 XML 文档,为了在播放学习单元的时候让播放器能解析这个 QTI 内容,在资源引用的时候必须把资源的类型申明为:imsqti_item_xmlv2p0^[4]。例如:

```
<imscp:resource identifier = "Question_1" type = "imsqti_item_xmlv2p0" href = "choice_1.xml">
  <imscp:file href = "choice_1"/>
</imscp:resource>
```

这样在播放阶段,播放器检测到类型为“imsqti_item_xmlv2p0”的 XML 文档时就要求调用 QTI 播放器来解析这个 QTI 文档。将在下面的播放阶段说明如何调用 QTI 播放器。

(2)播放阶段。

为了让 LD 播放器和 QTI 播放器之间能协调工作以及 LD 属性值和 QTI 变量值的同步变化,可以在现有播放器框架的基础上再加上一个中间软件层。这个

软件层由一个分发器 (Dispatcher) 和多个转接器 (Adapter) 组成^[5]。分发器里记录了每个转接器所对应的服务,分发器的任务是协调各转接器之间的工作和给客户端(如 SLeD)返回一个合适的转接器。转接器和播放器相联系,它实现了进入到播放器的接口,所以通过调用转接器处理事件,就相当于调用对应播放器的服务。转接器的任务是监听事件和通知分发器有事件发生。

如图 1 所示,分发器由三个功能模块组成:转接器信息模块,取 XML 文档模块,调用转接器模块。转接器信息模块记录了该分发器所对应的各个转接器的信息,以便能调用正确的转接器。取 XML 文档模块的功能是取得来自客户端的 XML 文档,然后作出分析,检查是否含有 QTI 文档内容。调用转接器模块的功能是调用正确的转接器。

如图 2 所示,转接器由三个功能模块组成:取 XML 文档模块,应答模块,接口模块。取文档模块的功能是取来自分发器的模块,然后通过接口让相应的播放器作出处理。在这里面应答模块是实现属性值和变量值同步更新的关键,应答模块是把播放器的处理结果返回给客户端,客户端根据这个返回再作下一步的处理。接口模块实现中间软件层与播放器的接口。

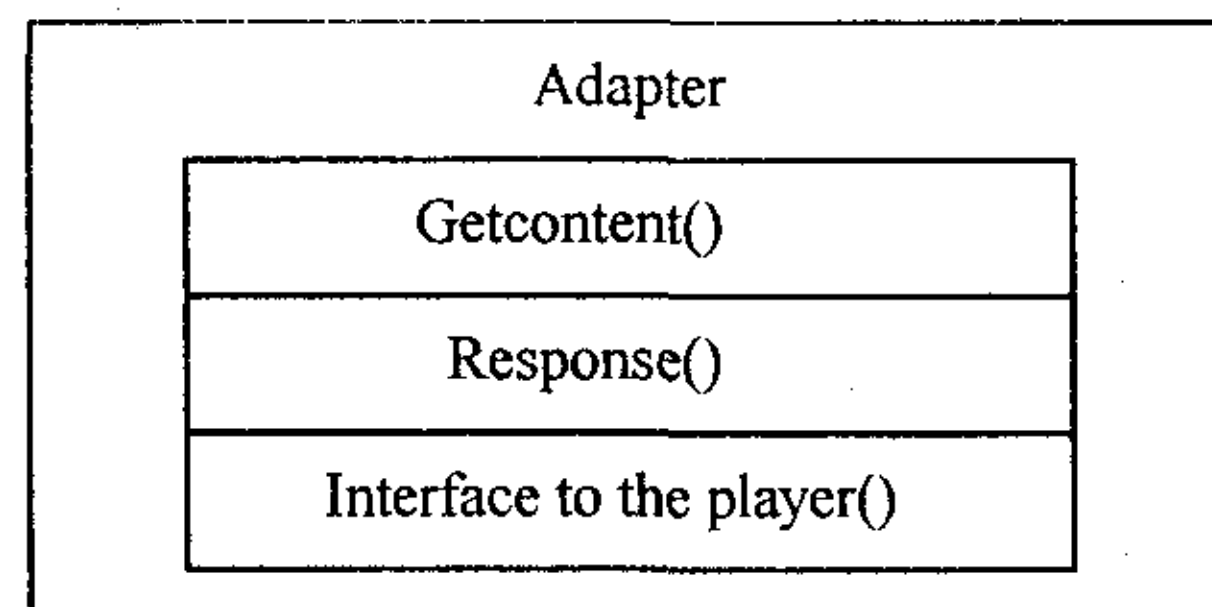


图 2 转接器结构

如图 3 所示,当客户端得到一个学习单元时首先调用分发器,分发器通过 Getcontent() 模块取得 XML 内容,因为分发器里有 Information of the adapters 模块记录了每个转接器的信息,所以可以通过 Getadapter() 模块调用对应的转接器。转接器由 Getcontent() 模块取得来自分发器的 QTI 内容或是 LD 内容,然后通过接口让播放器来处理这个内容。处理结果通过 Re-

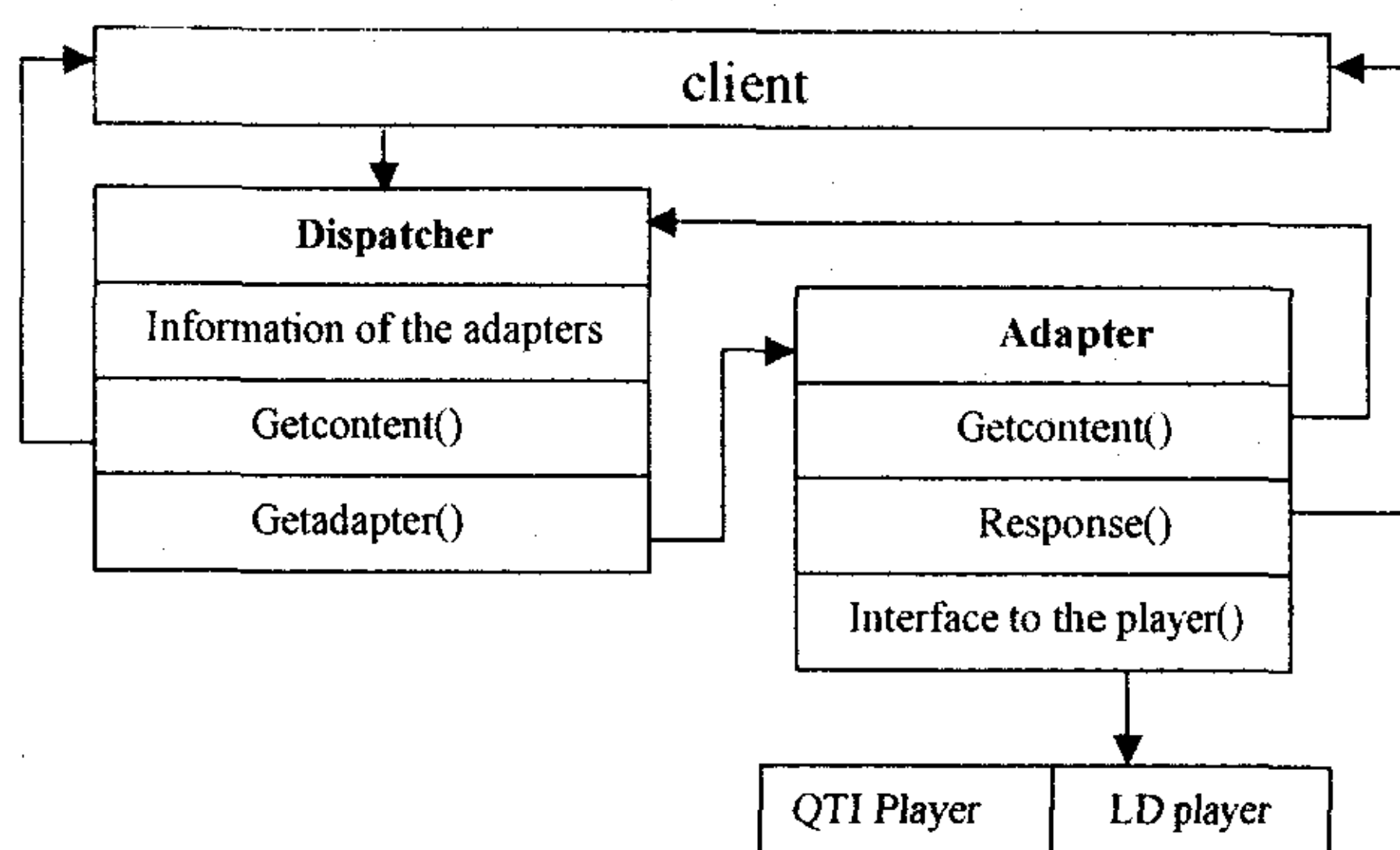


图 3 一个学习单元的处理过程

sponse()模块返回给客户端,如果接口调用的是 QTI 播放器,并改变了 QTI 文档里的变量值,那么客户端根据这个返回再调用 LD 播放器让它重新设置属性的值,所以应答模块是实现属性值同步更新的关键。

这样当客户端读入 XML 文档时就先通知分发器让它调用相应的转接器,转接器再处理这个文档。通过这个中间软件层,LD 播放器和 QTI 播放器的整合就不用改变现有存在的播放器的结构和代码,而是把这个整合封装到这个中间软件层里(如图 4 所示)。

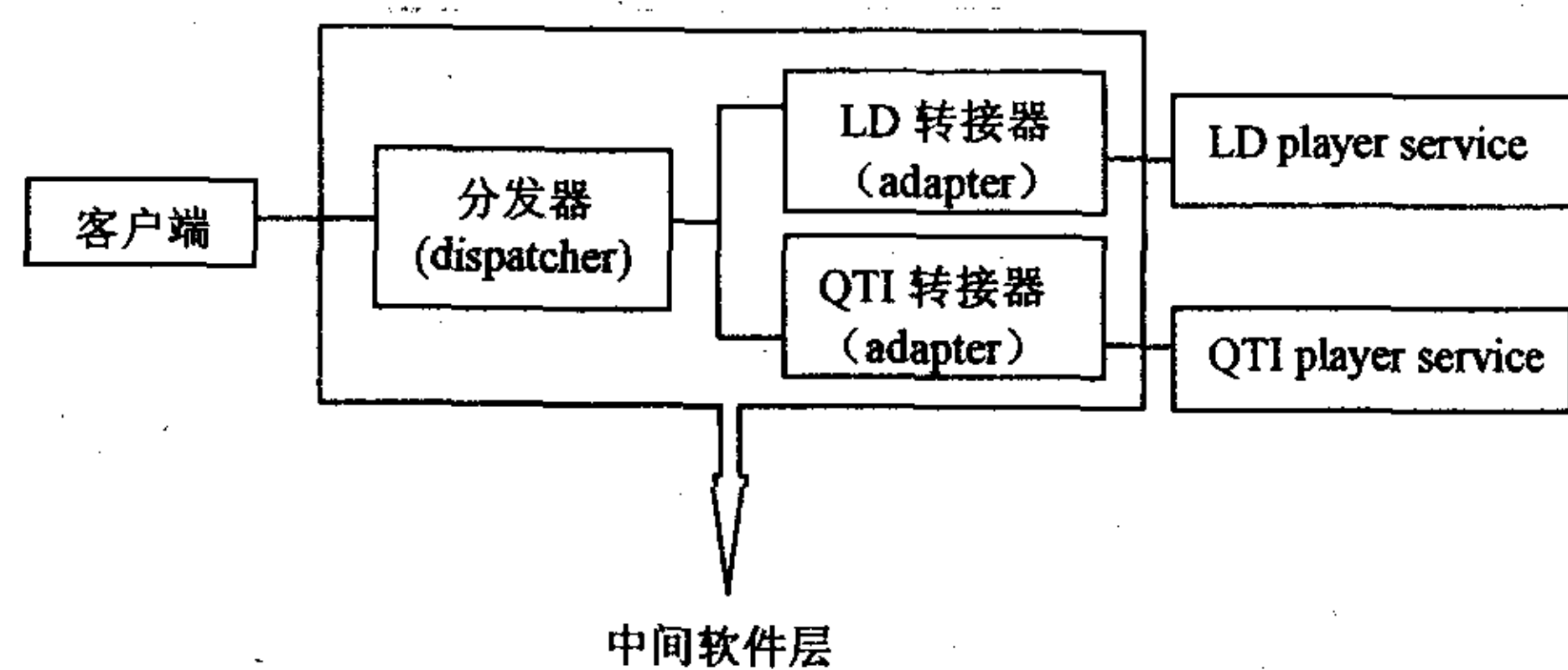


图 4 中间软件层

4 改进的 SLeD 体系结构

在解决了以上问题后可以在 SLeD 体系结构的基础上加上这个中间软件层。

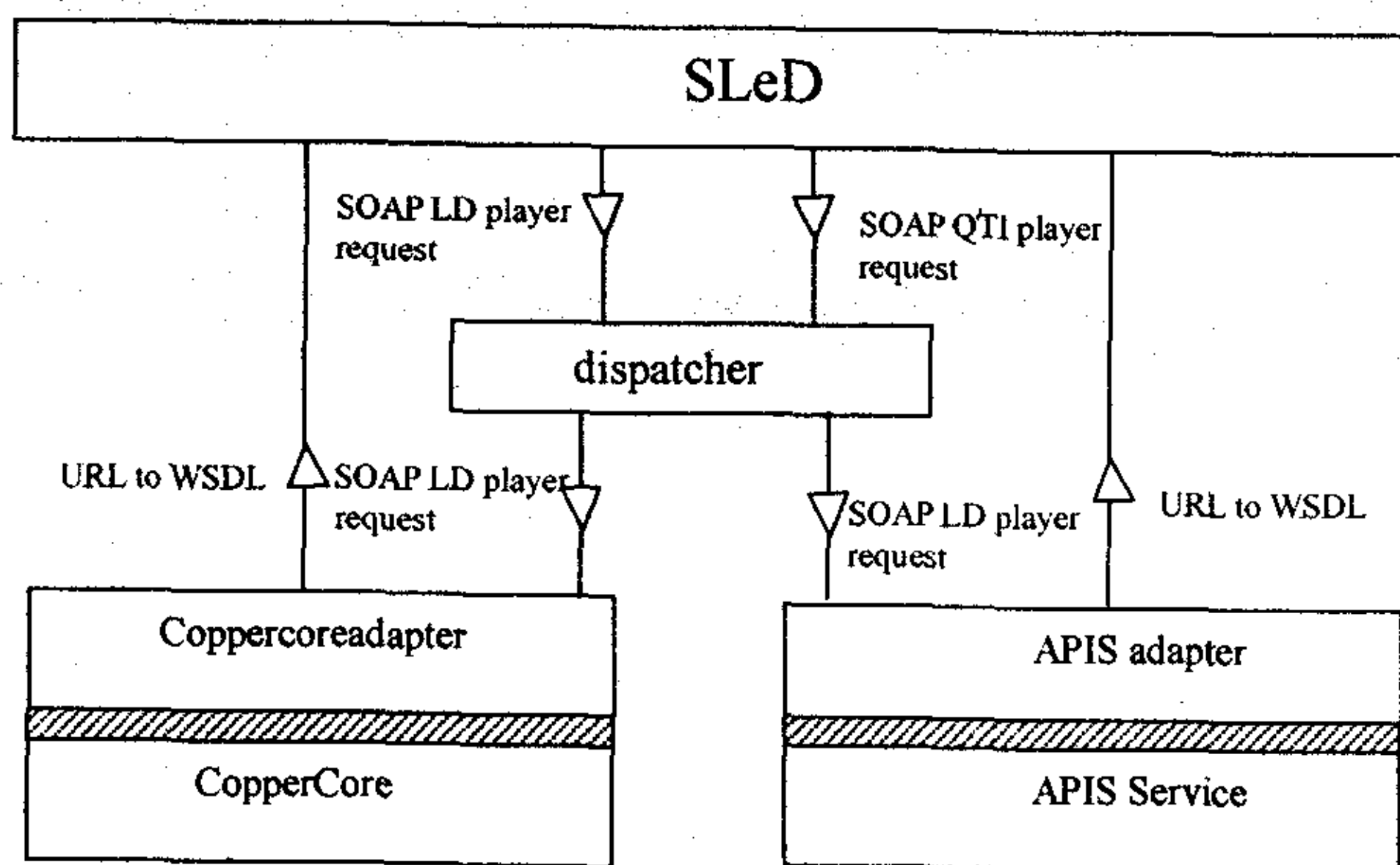


图 5 改进 SLeD 体系结构

当 SLeD 在调用 Coppercore 播放器解读一个 IMS 文档时,遇到一个类型为“imsqti-item-xmlv2p0”的资源时,SLeD 通过分发器调用 APIS^[4](Assessment Pro-

vision through Interoperable Segments)来处理这个文档,APIS 把自己的 URL 给 SLeD。用户做完这个测试文档之后就返回一个应答给 APIS,APIS 根据这个应答设置测试文档里的变量值,然后发出一个改变 LD 文档里属性值的事件请求。分发器根据这个事件请求调用 Coppercore 播放器,来设置属性的值,这样就达到了变量值和属性值的同步更新。

5 结论和展望

介绍了 E-learning IMS 学习设计规范,讨论了当前学习设计播放器中存在的问题,并提出了解决方法:在 SLeD 中引进中间软件层。这样在运行一个学习单元的时候不用来回地切换用户界面,也达到了属性和变量值的同步更新。下一步,将从以下几个方面做进一步的研究:

(1)对转接器接口作进一步的研究,让它更加具有适应性和兼容性,能应用于各种不同的播放器。

(2)让应答模块能够识别多种测试结果。

(3)在此体系结构的基础上开发出集成 APIS 播放器的学习设计播放器。

参考文献:

- [1] IMS Learning Design Information Model Version 1.0 Final Specification[C/OL]. 2003. <http://www.imsglobal.org.cn>.
- [2] McAndrew P, Nadolski R, Little A. Developing an approach for Learning Design Players[J/OL]. Journal of Interactive Media in Education, 2005(14)[2005]. www-jime.open.ac.uk/2005/14/mcandrew-2005-14.pdf.
- [3] 全国信息技术标准委员会. CELTS-10.1 练习\测试互操作规范 V2.0[S]. [s.l.]:[s.n.], 2004.
- [4] IMS QTI. IMS Question and Test Interoperability Integration Guide[C/OL]. 2006. <http://www.imsglobal.org/question/qti-v2p0/imsqti-intgv2p0.html>.
- [5] Vogten H, Martens H, Nadolski R, et al. Integrating IMS Learning Design and IMS Question and Test Interoperability using Coppercore Service Integration[C/OL]. 2006. <http://dSPACE.ou.nl/bitstream/1820/569/1/ccsi.pdf>.

(上接第 4 页)

sensor networks: A quantitative comparison[J]. The Int'l Journal of Computer and Telecommunications Networking, 2003, 43(4): 499-518.

- [10] Fok C L, Roman G C, Lu C. Mobile agent middleware for sensor networks: an application case study[C]// Information

Processing in Sensor Networks, 2005, IPSN 2005, Fourth International Symposium, 2005-04-15. NJ, USA: IEEE Press, 2005: 382-387.

- [11] 张云勇, 刘锦德. 移动 Agent 技术[M]. 北京: 清华大学出版社, 2003.