ICS 35.240.99
L67

中华人民共和国国家标准

GB/T XXXXX—XXXX

信息技术 学习、教育和培训 虚拟实验构件服务接口

Information technology – Learning, education and training - Virtual experiment component service interface

（报批稿）

XXXX－XX－XX 发布　　XXXX－XX－XX 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

本标准按照GB/T 1.1-2009给出的规则起草。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本标准由全国信息技术标准化技术委员会（SAC/TC28）提出并归口。

本标准起草单位：华中师范大学、中国电子技术标准化研究院、北京邮电大学、华中科技大学。

本标准主要起草人：吴砥、文福安、程文青、蒋文斌、余云涛、徐建、宝艳、林贤能、张耀丹、彭娴、张家琼、任慧、罗莉捷、王紫琴、饶景阳、李莹。

信息技术 学习、教育和培训
虚拟实验构件服务接口

1 范围

本标准定义了虚拟实验教学支撑平台的服务接口，规范了服务接口架构和接口定义。

本标准适用于虚拟实验教学系统的开发。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

GB/T 18793-2002 信息技术 可扩展置标语言(XML)1.0

GB/T 21364-2008 信息技术 学习、教育和培训 基于规则的XML绑定技术

3 术语和定义

下列术语和定义适用于本文件。

3.1

服务接口 service interface

一种抽象或可重用的计算机后台程序提供的功能，可被多个功能的实现定义实例化和引用，是一个自动化系统与另一个自动化系统或人之间的共享边界。

3.2

虚拟实验构件 virtual experiment component , VEC

虚拟实验环境中的独立操作单位，是可操作、可控制的虚拟实验基础组成对象。

3.3

构件 component

软件系统中具有相对独立功能、可以明确辨识、接口遵循约定的协议、可独立部署、可组装的软件实体。

3.4

插件 plug-in

插件是一种遵循一定规范的应用程序接口编写出来的程序。其只能运行在程序规定的系统平台下（可能同时支持多个平台），而不能脱离指定的平台单独运行。

4 缩略语

下列缩略语适用于本文件。

IP：互联网协议（Internet Protocol）

XML：可扩展置标语言（Extensible Markup Language）

5 虚拟实验构件服务接口

5.1 概述

虚拟实验构件服务接口的组织按照模块划分，包括虚拟实验教学可视化（模块 1 ）、虚拟实验教学资源构件的多领域建模与装配（模块2）、虚拟实验指导与管理（模块3）、虚拟实验开放式支撑平台（模块4）四个模块，各模块之间的关系如图1所示，其中：

a) 模块 1：构建基于图形的虚拟实验构件建模工具，基于几何的虚拟实验构件建模工具和虚拟实验场景的构建及可视化环境。为模块 2 提供创建教学实验构件的基本工具和环境；

b) 模块 2：建立多学科统一建模实验构件库结构，各个构件可相互调用。按学科划分存储结构，方便管理与测试，将国际单位和数学物理常量定义为通用标准库，保证各学科间相互调用的一致性。通过模块 4 平台进行通信以及数据传输，本模块与模块 1、模块 3、模块 4 分别商定相关的传输内容规范；

c) 模块 3：本模块为虚拟实验指导与管理系统模块。实现实验知识辅助学习子系统，虚拟实验过程的指导和实验结果自动批改、指导，虚拟实验答疑，实验室教学管理功能。本模块需要调用模块 2 的实验构件和实验资源，形成一个完整的教学实验；

d) 模块 4：本模块为虚拟实验门户平台，将模块 1、模块 2 和模块 3 的功能集成整合在一起，实现创建实验到完成实验整个过程的数据交换、求解任务调度和计算模块整合，最终把实验过程和结果数据存储到支持平台。

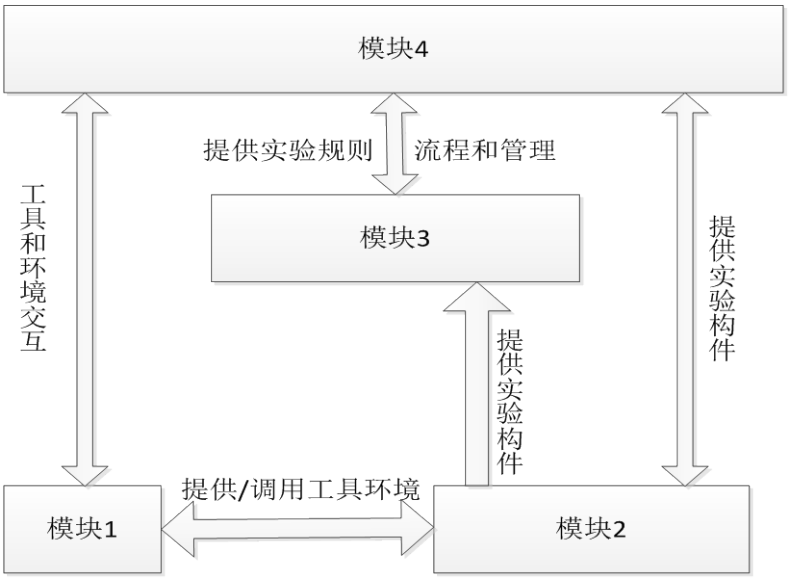

图 1 虚拟实验服务模块关系

接口应用场景示例见附录A。

5.2 接口编号格式

接口的编号格式为X-Y-Z，其含义如下：

——X：接口定义方；

——Y：接口使用方；

——Z：接口编号。

5.3 接口定义形式

接口定义形式由描述、功能、主要实现过程、输入参数和输出参数五部分组成，其中描述、功能、输入参数和输出参数是每个接口定义必须的。本标准接口描述遵循GB/T 21364-2008与GB/T 18793-2002所规定的语法规则和绑定规则。

5.4 模块 4 和模块 1 之间的接口

5.4.1 模块 4 为模块 1 提供的接口

5.4.1.1 保存实验构件

编号：4-1-1

描述：saveElement(in ElementXML:ElementXML):boolean

功能：保存实验构件，将XML文件表示的实验构件保存到数据库中

主要实现过程：解析XML文档，生成各个字段的相关信息，然后插入数据库中。数据的传输基于SOAP协议

输入参数：ElementXML，这是由模块1生成的构件的XML描述文件，该XML文件的描述性定义如下：

```
<Element>
  <Element-id></Element-id><!--构件 ID 由模块 1 定义标准，按学科统一编码 -->
    <visual-model><!--可视化模型，描述实验场景的表现方法及其显示 -->
      <name></name><!--可视化表现方法的名字 -->
      <type></type><!--可视化表现方法的类型 -->
      <description></description><!-- 文字描述 -->
      <experiment></experiment><!--实验的信息，如所属学科等-->
      < geometry-graph></ geometry-graph><!-- 几何模型 -->
      <image ><!--图片的信息描述-->
      <request><!--request 标签中指定对图片 URL 的访问协议-->
      </request>
      </image >
    </visual-model>
    <logic-model><!--直接存储逻辑模型的描述代码，字符串-->
    </logic-model>
    <property-list>
      <property name=" "  value=" " /><!-- 属性名称，值，范围，编号等-->
      <property name=" "  min =" "  max=" " />
      <property name=" " enum=" " />
      ……
      <connector-list><!-- 连接关系列表，（如：机械，电子实验，构件是连接关系）-->
        <connector id=/>
        <connector id=/>
      <connector-list>
    </property-list>
    <action-list><!-- 动作，即操作方法，不需要规定运动方式，由初始状态算出-->
```

```
        <action  name=" "   value=" "/><!-- 数据驱动，初始值-->
        <action  name=" "   value=" "/>
        ……
      </action-list>
    </Element>
```

输出参数：boolean，若保存成功，返回true，保存失败，返回false

5.4.1.2 删除实验构件

编号：4-1-2

描述：deleteElement(in ElementID:int):boolean

功能：删除数据库中已有的实验构件

输入参数：ElementID，构件的整型ID号，构件的ID号由客户端通过其他的公有服务来获取

输出参数：boolean，若删除成功，返回true，删除失败，返回false

5.4.1.3 获取实验构件

编号：4-1-3

描述：getElement(in ElementID:int):ElementXML

功能：获取数据库中的实验构件信息

主要实现过程：通过ID号从数据库中读取数据，然后拼接成XML文件，通过SOAP协议发送给客户端

输入参数：ElementID，构件的整型ID号

输出参数：ElementXML

5.4.1.4 获取实验构件列表

编号：4-1-4

描述：getElementList(in typeName:String):ElementListXML

功能：根据构件类型获取实验构件列表，主要供于教师在实验设计阶段设计构件时使用，用于索引构件列表，注意此处构件列表信息不是用于生成学生实验时可用的构件

输入参数： typeName，构件类别

输出参数：ElementListXML，包含一系列实验构件ID号的XML文档，该XML文档的定义如下：

```
<Element-list>
  <Element-id></Element-id>
  <Element-id></Element-id>
  ……
</Element-list>
```

5.4.1.5 保存实验配置信息

编号：4-1-5

描述：saveExperimentConfig (in experimentConfigXML:ConfigXML):boolean

功能：保存实验配置信息。

输入参数：experimentConfigXML，实验配置的XML文件，定义如下:

```
  <experiment-config ><!--实验配置信息 -->
    <experiment-id ></experiment-id>
    <Element-list>
```

```
      <Element id=”123321”><!--属性包括：值，范围，序号等-->
        <set-property  name=” ”  value=” ”  />
        <set-property  name=” ”  min=” ”  max=” ”/>
        <set-property  name=” ”  enum=”?,?”/>
        ……
        </Element>
        <connect-rule></connect-rule><!--连接规则，供模块4使用，预留扩展 -->
      ……
    </Element-list>
  </experiment-config>
```

输出参数：boolean，若保存成功，返回true，保存失败，返回false

5.4.1.6 删除实验配置信息

编号：4-1-6

描述：deleteExperimentConfig(in experimentConfigID : Number) : boolean

功能：删除实验配置信息

输入参数：experimentConfigID，实验配置的ID号

输出参数：boolean，若删除成功，返回true，删除失败，返回false

5.4.1.7 获取实验配置信息

编号：4-1-7

描述：getExperimentConfig(in configID : Number) : ConfigXML

功能：从数据库中获取实验配置数据

输入参数：configID，实验配置的ID号

输出参数：ConfigXML，实验配置信息的XML文件

5.4.1.8 保存实验场景

编号：4-1-8

描述：saveScene(in sceneXML : String) : Number

功能：保存运行时的实验场景，包括实验场景中所使用的构件、构件之间的位置及衔接等逻辑关系

输入参数：SceneXML，实验运行时的场景文件，以XML的形式进行定义。在数据库中以字符串的形式进行保存。

```
  <scene ><!--场景描述-->
    <scene-id ></scene-id>
    <Element-list>
      <Element id=”123321”>
<location x=’ ’  y=’ ’  /><!--位置信息-->
        ……
        </Element>
      ……
    </Element-list>
    <connection from=’123321’  to=’123322’  /><!--描述连接关系  -->
    ……
```

</scene>

输出参数：返回所保存的实验场景的ID号，若返回值不为0，保存成功，若返回值为0，保存失败。

5.4.1.9 获取实验场景

编号：4-1-9

描述：getScene(in sceneID : Number) : SceneXML

功能：获取实验场景，通过实验场景文件恢复上一次所保存的实验状态

输入参数：sceneID，保存场景时生成的实验场景ID号

输出参数：sceneXML，实验场景描述的XML文件。该XML描述文件在数据库中以字符串形式进行保存。

5.4.1.10 仿真命令接口

编号：4-1-10

描述：doSimulation(in sceneID : Number) : ComputeNode

功能：通过场景文件全局ID获取场景文件，转换为逻辑运算文件，由模块4进行调度，在相应节点运算并返回运算结果

输入参数：场景文件全局ID

输出参数：返回值为计算节点的信息，开始仿真，接收数据，直到仿真结束，关闭连接；若为NULL，给出提示。

```
<compute-node ><!--节点信息，模块 1 调用模块 2 仿真环境，是通过模块 4 分配计算节点-->
<destination><!-- 目标节点 -->
  <ip></ip><!-- IP 地址 -->
    <port></port><!-- 端口号 -->
     <thread></thread><!-- 进程 ID -->
    </destination>
<requester><!-- 请求方信息 -->
<ip></ip><!-- IP 地址 -->
<port></port><!-- 端口号 -->
<thread></thread><!-- 进程 ID -->
</requester>
  <session></session><!--客户端和后台计算节点建立会话的信息-->
  …
</compute-node>
```

5.4.2 模块 1 为模块 4 提供的接口

5.4.2.1 清除当前工作集数据

编号：1-4-1

描述：clearWorkspace() : void

功能：清除当前工作集(即用户当前所创建的实验场景)数据，恢复到场景的初始状态；供Portal调用，提供统一接口

输入参数：无(返回success/fail)

输出参数：无

5.4.2.2 刷新当前工作集

编号：1-4-2

描述：freshWorkspace() : void

功能：在客户端可视化渲染速度有限或者网络传输速度有限的情况下，刷新当前工作集数据；供Portal调用，提供统一接口

输入参数：无

输出参数：无

5.4.2.3 运行实验

编号：1-4-3

描述：orderSimulation() : void

功能：调用4-1-10 doSimulation接口实现；供Portal调用，提供统一接口

输入参数：无

输出参数：无

5.4.2.4 计算节点发生改变

编号：1-4-4

描述：simulationNodeChanged(in node : ComputeNode): void

功能：如模块4对应的计算节点失效，通过调用该接口通知模块1，并让用户选择合适的计算节点，由模块1重新在该节点执行仿真任务；供Portal调用，提供统一接口

输入参数：节点信息node

输出参数：无

5.4.2.5 保存场景

编号：1-4-5

描述：storeScene() : void

功能：保存用户的实验场景，调用接口4-1-8 saveScene；供Portal调用，提供统一接口

输入参数：无

输出参数：无

5.4.2.6 恢复实验场景

编号：1-4-6

描述：restoreScene() : void

功能：供用户恢复实验场景，即恢复上一次的实验状态，继续实验，通过调用接口4-1-9 getScene实现；供Portal调用，提供统一接口

输入参数：无

输出参数：无

5.5 模块 4 和模块 2 之间的接口

5.5.1 模块 4 为模块 2 提供的接口

5.5.1.1 保存用户实验场景的逻辑信息

编号：4-2-1

描述：saveLogicalInfo (in sceneID:Number):Boolean /(考虑用户id，场景id)

/保存对应的用户，对应的场景，对应的Mo文件*/
<logic-scene>
<userID></userID>
<sceneID></sceneID>
<Mofile><!--模型文件描述 -->
<Element-list>
<Element id=”123321”><!--用了哪些构件，属性包括：值，范围，序号等-->
<enum></enum><!-- 个数-->
</Element>
<connector from “123” to “124”></connector><!--连接关系描述 --><
……
</Mofile>

功能：保存实验的逻辑信息Mofile，用于模块2的编译连接，以及模块3的智能指导

输入参数：CompileInfoXML，实验编译信息的XML文件，对模块4来说，同样可以把他当成字符串来进行处理，不关注XML文件的结构细节

输出参数：boolean，若保存成功，返回true，保存失败，返回false

5.5.1.2 保存仿真运行结果

编号：4-2-2

描述：saveRunningResult (in CompileInfoXML：String):boolean

功能：保存实验的运行结果信息到课题5

输入参数：RunningresultXML，实验运行结果信息的XML文件，对模块4来说，同样可以把他当成字符串来进行处理，不关注XML文件的结构细节

输出参数：boolean，若保存成功，返回true，保存失败，返回false

5.5.2 模块 2 为模块 4 提供的接口

5.5.2.1 可视化场景（XML）转化成求解输入源文件

编号：2-4-1

描述：convertMoFile(in sceneXML: string):Mo

功能：将可视化场景(XML)转换成求解输入源文件，并返回给模块4保存

输入参数：scene，实验场景的拓扑结构，具体形式可以是一个字符串描述

输出参数：Mo文件（模型文件，可为Modelica，3DMAX，VRML支持格式），即仿真程序的源文件

5.6 模块 4 和模块 3 之间的接口

5.6.1 模块 4 为模块 3 提供的接口

5.6.1.1 显示实验帮助信息

编号：4-3-1

描述：showHelpInfo(mySession :SessionInfo, myhelp :helpInfo):Boolean;

功能：模块4为模块3提供一个帮助信息的弹出窗口，针对不同的用户，显示对应所需要的帮助信息

输入参数：会话信息和帮助信息

输出参数：返回帮助结果的XML文件

<session ><!--会话的配置信息-->

```
<sessionId></sessionId><!--会话 ID-->
<request><!--请求会话-->
<getSession></getSession><!--获取会话-->
  <createTime></createTime><!--会话创建时间-->
  <LastAccessTime></LastAccessTime><!--会话最后访问时间-->
  <isNew></isNew><!--是否新的会话-->
  <set Attribute “”= ><!--设置各种属性-->
</request>
……
</session>
```

5.6.2 模块 3 为模块 4 提供的接口

5.6.2.1 获取构件逻辑信息

编号：3-4-1
描述：sendElementInfo(void): ElementXML
功能：模块3将构件逻辑信息给模块4
输入参数：实验内容信息
输出参数：通过模块3提供接口，调用模块4数据库信息，返回构件逻辑信息的XML文件

5.6.2.2 获取实验内容信息

编号：3-4-2
描述：sendExperimentContent(void): ContentInfo
功能：模块4将实验内容给模块3
输入参数：空
输出参数：返回实验内容信息的XML文件

```
<experiment-content><!--实验内容描述，参照实验配置，实验构件描述 -->
  <userID></userId><!-- 用户 ID -->
  <subject></subject><!--所属学科 -->
  <experiment-info><!--实验描述 -->
    <Element></Element><!--构件信息（ID，数目等）-->
    <action></action><!-- 各种操作，包括拖动位置，连接关系等-->
  <experiment-info>
</experiment-content>
```

5.6.2.3 获取实验中间结果信息

编号：3-4-3
描述：sendExperimentTempResult(void): ContentInfo
功能：模块4将中间实验内容给模块3
输入参数：实验内容信息
输出参数：返回中间结果的 XML 文件

附 录 A
（资料性附录）
接口应用场景示例

A.1 保存场景示例

编号：1-4-5 storeScene

数据交互流程：点击平台上的保存实验后，数据交换层会调用插件定义的方法，该方法用于获取插件中要保存的内容字符串，通过这个方法数据交换层能够获取要保存的数据，然后发给后台，执行相应的保存动作。

接口定义方：客户端插件

接口使用方：数据交换层

接口定义：

Flex：

```
public function getSceneDataForSave( ):String
```

Applet：

```
public String getSceneDataForSave( )
```

A.2 恢复实验示例

编号：1-4-6 restoreScene

数据交互流程：和恢复实验场景对应。在点击平台上的恢复实验后，数据交换层从后台获取实验场景数据，然后调用插件里相应的方法，将数据传递给客户端插件。客户端插件拿到这些数据后，将实验场景恢复。

接口定义方：客户端插件

接口使用方：数据交换层

接口定义：

Flex：

```
public function setSceneDataForRecover(xmlstr:String):void
```

Applet：

```
public void setSceneDataForRecover (String xml)
```

A.3 运行实验示例

编号：1-4-3 orderSimulation

数据交互流程：在点击运行实验后，数据交换层向客户端插件获取实验运行所需的初始化数据，然后将数据发送给后台计算模块。后台计算模块计算完成后，将结果数据发还给数据交换层，最后数据交换层将数据传递给客户端插件。

接口定义方：客户端插件

接口使用方：数据交换层

接口定义：

Flex：

```
public function getSceneDataForRun():String
```

Applet：

```
public String getSceneDataForRun()
```

A.4 获取实验结果数据示例

编号：1-4-3 orderSimulation

数据交互流程：后台计算模块计算完成后，将结果数据发还给数据交换层，最后数据交换层将数据传递给客户端插件。

接口定义方：客户端插件

接口使用方：数据交换层

接口定义：

Flex：

```
public function setRunningResult(str:String):void
```

Applet：

```
public void setRunningResult(String xml)
```

A.5 清除当前工作集数据示例

编号：1-4-1 clearWorkspace

数据交互流程：用户可以通过接口上的按钮，清除当前插件面板上的实验构件，开始一个新的实验。

接口定义方：客户端插件

接口使用方：数据交换层

接口定义：

Flex：

```
public function clearScene():void
```

Applet：

```
public void clearScene()
```

A.6 获取实验构件示例

编号：4-1-3 getElement

数据交互流程：插件提交一个实验构件的ID或名称给数据交换层，数据交换层代理这个插件的请求，从后台获取相应的实验构件数据，最后将实验构件数据返回给插件。这是一个“请求—回应”的模式。接口的定义方和调用方要分成“请求阶段”和“回应阶段”两个部分来描述：

e) 请求阶段：

接口定义方：数据交换层。插件调用这个数据交换层的方法，发起这个数据交互的动作，并将数据传递给数据交换层。

接口使用方：客户端插件

接口定义：

JavaScript:

functioncomponentRequest(String condition)

f) 回应阶段：

接口定义方：客户端插件。数据交换层调用这个插件方法，向插件返回请求的应答结果。

接口使用方：数据交换层

接口定义：

Flex：

public function componentResponse(xml:String):void

Applet:

public void componentResponse(String xml)

A.7 获取实验构件列表示例

编号：4-1-4 getElementList

数据交互流程：在用户选择某个实验后，接口会加载实验所对应的客户端插件。在插件加载完成后，数据交换层会向后台请求做实验所需的构件列表，最后将构件列表信息发送给客户端插件。

接口定义方：客户端插件

接口使用方：数据交换层

接口定义：

Flex：

public function setComponents(xmlstr:String):void

Applet：

public void setComponents(String xml)

参 考 文 献

[1] GB 4880-1991 语种名称的编码表示（ISO 639:1998）.
[2] GB/T 26222-2010 信息技术 学习、教育和培训 内容包装 .
[3] GB 13000-2010 信息技术 通用多八位编码字符集(UCS)（ISO/IEC 10646:2003，IDT）.