一种基于NodeJS引擎的本地交互软件架构技术

#### 一：详细介绍技术背景。

在互联网PC时代，人们对基于B/S,C/S两种软件架构框架有了更深刻的体验。B/S意味着本地开发模式，对于用户而言，分布性强，维护方便、开发简单且共享性强，总体成本低。但数据安全性问题，对服务器要求过高，数据传输速度慢、软件的个性化特点明显降低等缺点。C/S意味着可以和用户有更多的交互性，可以给用户更好的本地体验,即更强大的功能，更高级别的安全性，但是相对比较“笨拙”，缺乏灵活性和造成系统资源的浪费。因此我们提出一种基于nodeJS引擎分布式软件开发架构，可以把这两者的优点相互结合，让软件开发人员有更好的开发体验。

本发明是一款基于XML+JS的跨平台分布式软件开发架构。nodeJS是一个让 JavaScript 运行在[服务端](https://baike.baidu.com/item/%E6%9C%8D%E5%8A%A1%E7%AB%AF/6492316" \t "https://baike.baidu.com/item/node.js/_blank)的开发平台，更是一个基于Chrome JavaScript运行时建立的平台， 用于方便地搭建响应速度快、易于扩展的网络应用。

XML语言是一种可扩展标记语言，它提供了解释，验证，查询，变换功能。它提供了一种比较简单的描述内容的方法，而且这种方法能表述列表、树等常见的数据结构，可扩展性好。因此在各个操作系统上都可以解析XML格式的语言。

B/S指浏览器/服务器模式，C/S指客户端/服务器模式，这两种软件架构框架都有各自的优点和缺点。因此本发明提出一款基于nodeJS引擎的跨平台分布式软件开发架构，兼具两种开发模式优点。

#### 二：现有技术缺点

B/S但数据安全性问题，对服务器要求过高，数据传输速度慢、软件的个性化特点明显降低等缺点。C/S有部署成本过高，缺乏灵活性和易造成系统资源的浪费。基于浏览器的B/S架构存在本地化体验痛点，很多情形下无法满足要求。

现有的本地化开发技术存在“平台异构问题”：不同操作系统需要开发不同的App，另外数据库和数据系统等也需要分开储存。从而导致移动App开发成本高，开发周期长，App内存空间受到限制。更重要的是，App不具有移植性，开发的App并不能跨平台使用，在不同的平台无法进行通用。因此本发明提出一款基于XML+JS的跨平台分布式软件开发架构，来解决以上提到的缺点。

#### 三：本发明优点

该模型内部嵌入了浏览器模块，可以拥有B/S浏览器优点，也有C/S客户端优点。本发明通过基于XML解析来实时获取服务器端传过来的数据。解决了用户浏览器模式下体验不佳，客户端模式下部署成本过高等问题。可以实现随时浏览网页数据，并且拥有客户端的本地体验。

另外本发明还能做到跨平台，即Windows/UNIX/IOS/Android/MACOS各个平台之间相通，用于解决平台异构问题。由于XML文本格式的易解读性，在上述平台上，都可以兼容该引擎。这也就是跨平台性。以常用的java为例，可以有多种方式解析XML文件，例DOM方式，SAX方式，JDOM方式，DOM4j方式。

#### 四：本发明带来的有益效果

与现有技术相比，本发明提供了一种本地化解析的有益效果。本发明提出一款基于nodeJS引擎的跨平台分布式软件开发架构，极大的提高工业软件开发效率和开发门槛。该引擎内部已经嵌入一些十分优秀的算法，对模型进行分析可以用到这些算法。未来通过软件迭代升级，算法库会得到进一步的完善。因此，开发者可以聚焦于核心算法，而不用去花费精力设计界面布局等。

#### 五．是否有可替代的方案

本发明提出的创新思路十分新颖，并无可替代方案。

#### 本发明技术方案的详细阐述

一种基于NodeJS引擎的本地交互软件架构技术。其特征在于，由nodejs用户端、Web服务端模块和云端模块组成，nodejs用户端和云端模块连接，Web服务端模块和云端模块连接；

所述Web服务端包括储存子模块，计算子模块，连接子模块和辅助子模块。

所述连接子模块暴露在云端的接收和发送端口；所述计算子模块用于计算用户端发送的计算请求。所述储存子模块用于用户上传模型并保存。

nodeJS是基于 Chrome V8 引擎的[JavaScript](https://baike.baidu.com/item/JavaScript/321142" \t "https://baike.baidu.com/item/node.js/_blank)运行环境，具有轻量、可伸缩，适于实时数据交互应用等优点。它主要用于浏览器前端设计，但webGL交互体验问题比较严重，对于新手入门门槛更高。另外，NodeJS也正成为Web后端主流开发平台。Node.js处理高并发场景性能更高，可用于搭建高性能的web服务器。发明者既想兼容NodeJS的本地UI引擎，但必须降低其使用难度，因此提出一种基于NodeJS引擎的分布式软件架构技术，且引擎下基于XML+JS两种语言，来提供GUI描述接口，以便于用户有更好的交互体验。

本发明适应于跨平台开发计算软件，即Windows/UNIX/IOS/Android/MACOS各个平台之间相通。由于XML文本格式的易解读性，在上述平台上，都可以兼容该引擎，由不同技术构建的信息系统形成了一个个“信息孤岛”，这阻碍了应用程序间的互相协同、对数据库资源的共享。在面向软件框架使用者的接口层上底层实现是透明的，从而实现在各个平台上的统一GUI描述和开发模式。在Windows平台下，通过...实现，在UNIX平台下，通过...实现 ，在IOS平台下，通过...实现，在Android平台下，通过...实现,在MACOS平台下，通过...实现。通过以上方法，我们打破了“信息孤岛”，实现各平台的接口实现和分离。

首先部署数据传输到服务端，通过云端这个媒介，用户实现了客户端和浏览器的双重体验。用户登录DCiP引擎，进入服务器访问模块。用户发送请求到云端，云端从服务端请求数据，然后通过本地连接来传送回给用户端。因为它可以类比成本地客户端打开模型的过程，我们可以很快的打开模型。基于Websocket通信协议，在很多解析器上都可以打开XML文本。计算在云端，实现在云端。

在本发明内部，我们定义和实现一个通用的App Explorer。WebSocket通信协议使得客户端和服务器之间的数据交换变得更加简单，允许服务端主动向客户端推送数据，并且更加安全可靠。在WebSocket API中，浏览器和服务器只需要完成一次握手，两者之间就直接可以创建持久性的连接，并进行双向数据传输。用户输入网址发送请求，并启用发送因此我们基于Websocket协议远程加载GUI描述的XML+JS脚本，动态解析成本地应用，这样我们就有了本地客户端体验；JS脚本通过Websocket协议建立和服务端的通讯及访问机制，这样我们就可以有随时访问服务端地址的功能。从而实现了融合B/S ，C/S架构各自特点的分布式软件架构。

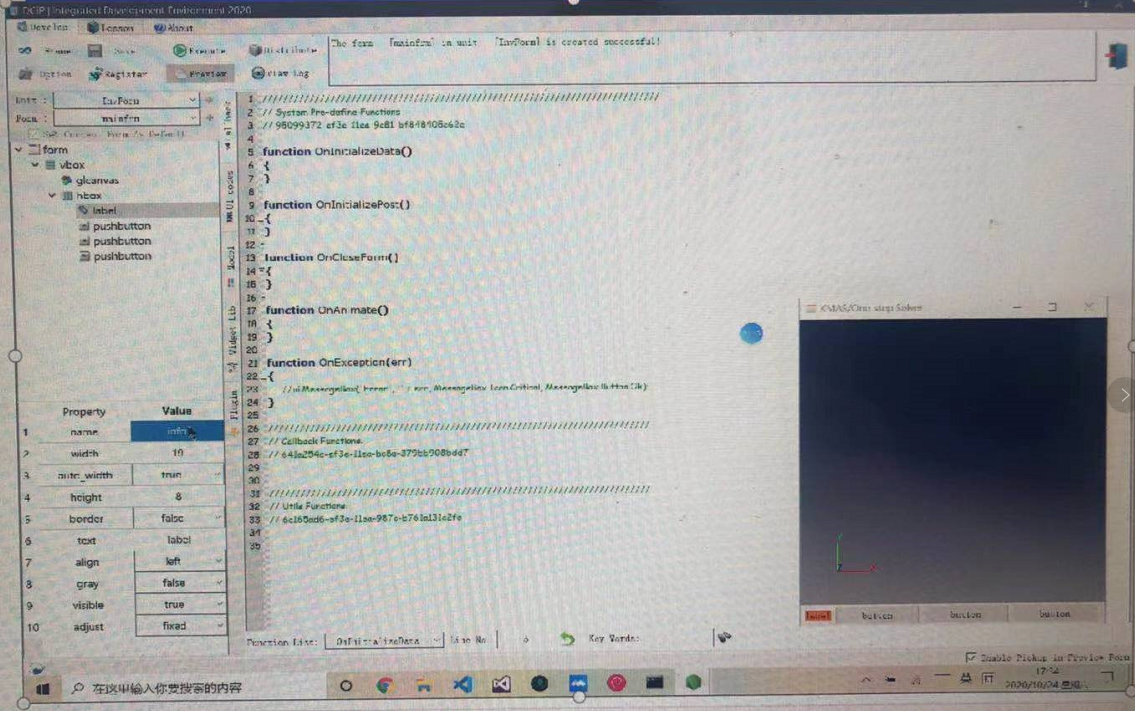
#### 七： 通过实施例对技术方案进行清楚完整的描述。

一：快速生成界面。在页面的左侧，生成一个按钮，调整按钮显示内容，并且可以写它的响应函数，高度，宽度等属性。在代码区就会自动生成代码来对应生成的控件。然后通过控制控件之间的逻辑关系，即并列，继承，包含等关系。通过上述的操作，我们可以快速生成软件基本模型和布局界面。本发明拥有着基于数据描述的UI控制逻辑特性，先通过赋值实现界面控制逻辑，不需要很高级的操作，只需要赋值即可，因此极大的降低了开发门槛。3D图元创建，后处理云图创建，颜色调，加载Nastran文件，3D图形操作，3D图形选取，光照等均由相关代码来控制，以便于快速生成计算软件。该软件就可以快速生成XML+JS界面，然后修改一个选项建立Websocket服务端应用，详情见例图；

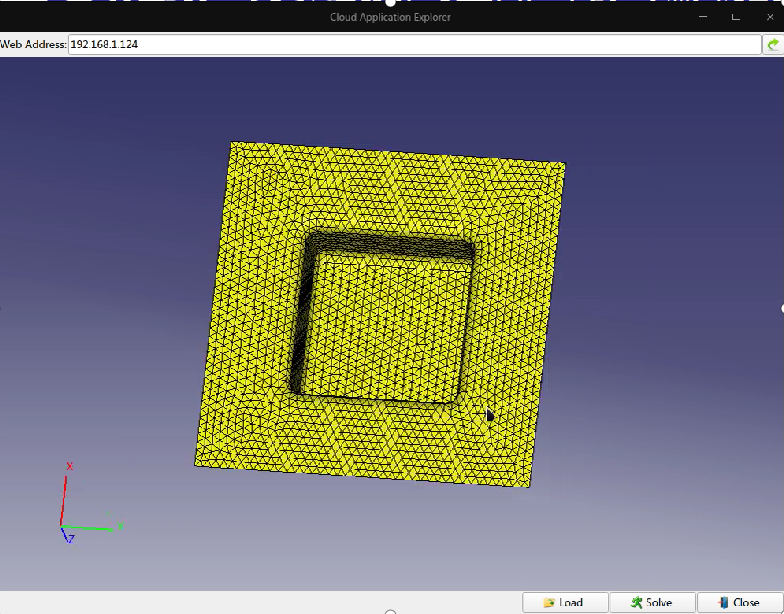
二：在客户端内部嵌入浏览器模块中运行浏览器。首先部署数据传输到服务端，通过云端这个媒介，用户实现了客户端和浏览器的双重体验。用户登录DCiP引擎，进入服务器访问模块。用户发送请求到云端，云端从服务端请求数据，然后通过本地连接来传送回给用户端。因为它可以类比成本地客户端打开模型的过程，我们可以很快的打开模型。通过XML通信协议，在很多解析器上都可以打开XML文本。计算在云端，实现在云端。

以浏览一个方盒模型为例，输入服务端地址192.168.1.124就可以访问网站地址，点击“Load”这个Button；软件就会通过Websocket通信协议，来访问云端内容。紧接着XML+JS语言通过Websocket协议被动态加载解析，Websocket实现了浏览器与服务器的全双工通信，浏览器和服务器任何一方随时都能够主动给对方说话，从而建立本地化应用。JS代码通过客户端和服务端已经约定好的Websocket通信协议和服务端通讯，实现对远程服务中的数据和计算资源进行访问。详情见例图。这样就达到了B.S和C/S两者优点相结合的目的。

//快速开发软件案例



//客户端访问云端网址示例



八．本发明的关键点和欲保护点

上述具体的如快速开发一个轻量级软件用例，本发明的专利保护范围包括但不限于上述具体的实施方式,任何符合本发明的基于NodeJS引擎的分布式软件架构及方法权利要求书的且任何所述技术领域普通技术人员对其做出的适当变化或者替换,皆应落入本发明的专利保护范围。此外此发明兼具B/S，C/S两种软件开发模式优点，这种开发模式和开发理念也应该被落入相应的专利保护范围。另外，在此发明内部嵌入的一些算法如KMAS/One-Step也应该受到保护。

尽管已经示出和描述了本发明的实施例,对于本领域的普通技术人员而言,可以理解在不脱离本发明的原理和精神的情况下可以对这些实施例进行多种变化、修改、替换和变形,本发明的范围由所附权利要求及其等同物限定。

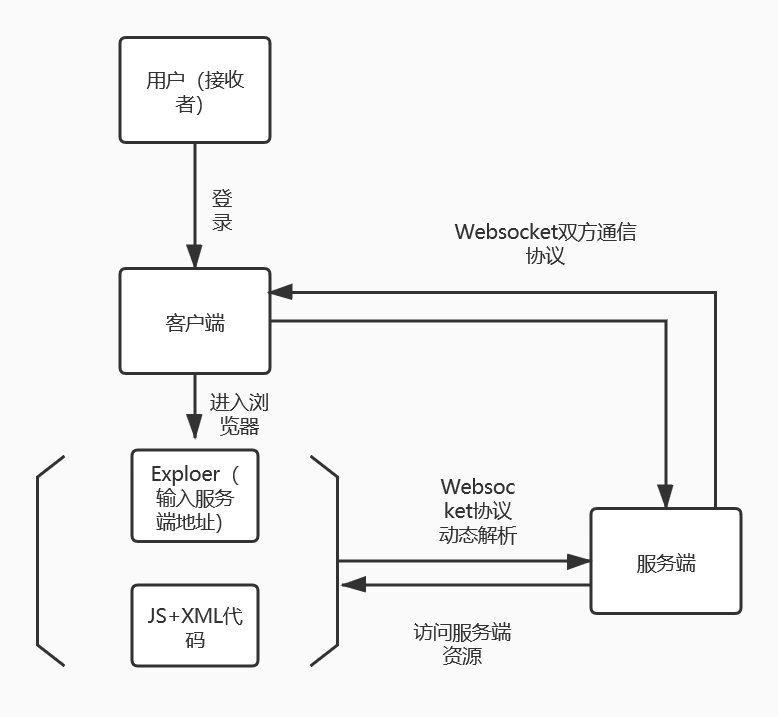


图1