

IDC 可视化软件开发平台中的配置与编译子系统^①

薛芸^{1,2}, 廉东本², 王俊霖³

¹(中国科学院大学, 北京 100049)

²(中国科学院 沈阳计算研究所, 沈阳 100168)

³(大连理工大学, 大连 116621)

摘要: 为了解决目前 IDC(互联网数据中心)机房可视化管理软件开发代价大、灵活性差、迭代周期长的问题, 提出了可定制的、模块复用的 IDC 可视化软件开发平台, 本文详述了该平台中核心子系统—配置与编译子系统的设计与实现。配置与编译子系统采用 AssetBundle 技术封装物体模型, 采用 DLL 封装功能模块, 并提出了一种新的结合编辑距离和分块计算的增量更新算法 AssetDiff, 从而满足了编译、管理可复用的物体模型和功能模块的需求, 并实现了管理、配置、发布 IDC 可视化软件以及已发布软件远程更新的功能。

关键词: 互联网数据中心; 可视化; 模块化; Unity3D; 增量更新

Configuration and Compilation Subsystem of IDC Visual Software Development Platform

XUE Yun^{1,2}, LIAN Dong-Ben², WANG Jun-Lin³

¹(University of Chinese Academy of Sciences, Beijing 100049, China)

²(Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang 110168, China)

³(Dalian University of Technology, Dalian 116024, China)

Abstract: Aiming at the problem of Long development cycle, high cost and poor flexibility in the field of IDC (Internet Data Center) computer room visual management software, an IDC Visual software development platform is proposed. This paper describes the design and implementation of the core subsystem, configuration and compilation subsystem of the platform. Configuration and compilation subsystem uses AssetBundle package object model, uses DLL encapsulation function module, and proposes a new incremental updating algorithm AssetDiff that combines edit distance and block computation. The system meets the need of compilation and management of Reusable Object Model and functional model and implements the function of management, configuration, releases IDC visualization software and the publishes software remote update.

Key words: internet data center; visualization; modularization; unity3d; incremental update

IDC(互联网数据中心)可视化软件是一种通过三维可视化技术开发的机房管理工具, 它将参照现实场景建模的虚拟场景与机房真实运行数据相结合, 从而形象化地呈现机房的各项运行参数(如温度、湿度、故障、空调新风、门禁状态、电气状态等), 从而为运维人员和管理层提供决策依据^[1]。由于不同企业的数据中心有着不同的物理环境和管理需求, 单独为每一个企业开发 IDC 可视化管理软件的成本较高、软件复用率比较低^[2]。因此, 具备可定制、模块化特点的 IDC 可视化软件开发平台是一个良好的解决方案。它将 IDC

领域内常用的三维模型与脚本代码提取出来封装为模块库, 开发人员无需考虑底层的实现细节, 仅需要通过客户需求来选择相应模块, 就可以组装产生一套新的适合需求的 IDC 可视化管理软件。该平台可以极大地提高可视化软件开发效率, 并对后期的客户需求变更、远程运维提供了良好的支持。

1 设计思想

IDC 可视化软件开发平台采用 B/S 架构, 通过结合 JavaWeb 技术、Microsoft .NET Framework 以及

① 基金项目:国家水体污染控制与治理科技重大专项(2012ZX07505004)

收稿时间:2016-07-04;收到修改稿时间:2016-07-29 [doi: 10.15888/j.cnki.csa.005650]

Unity3D 引擎来完成主体功能. 其总体架构图(图 1)中明确划分了场景编辑系统、配置与编译系统以及基于 Unity3D 引擎二次开发的内核容器系统三个部分.

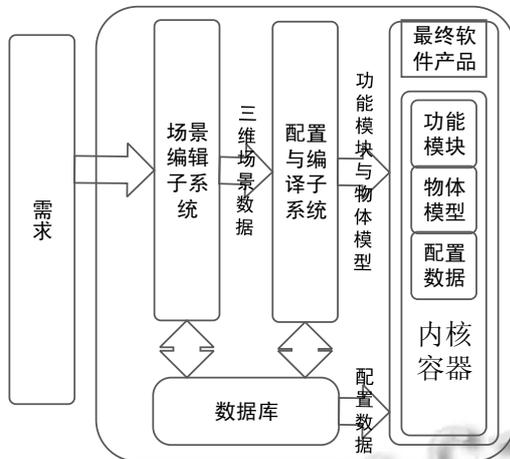


图 1 平台架构图

用户在场景编辑子系统中搭建与现实场景同步的机房三维场景后, 进入到配置与编译子系统中选择该机房的可视化管理软件需要的相关模块, 则可在内核容器中呈现出相应的可视化软件.

其中配置与编译子系统是整个平台的核心, 通过对平台业务需求的分析整合, 将配置与编译系统的主要业务过程分为以下几个阶段:

1) 将用户上传各种格式的三维模型打包压缩为平台标准物体模型, 将用户上传的某功能相关代码编译为平台标准功能模块.

2) 创建新的可视化软件: 录入该可视化软件的基本信息(软件名称、描述、部署地点、机房所属企业信息等).

3) 选择该可视化软件对应的现实机房场景的三维数据(楼体、楼层、机柜、机架、线路等物体模型在三维空间中的位置信息).

4) 配置该可视化软件的可用功能与相关参数: 根据场景数据中涉及的物体模型, 查询出该机房可以使用的可用功能模块全集: 包含告警、安防、门禁、电量监测、空间使用情况监测等功能, 相关参数包含有环境传感器与摄像探头的 Webservice 路径、软件中的 UI 主题颜色等.

5) 可视化软件发布: 将该可视化软件需要的物体模型和用户选择的功能模块按照配置参数进行装配, 生成配置数据与资源文件夹, 输出至内核容器, 从而

输出最终软件产品.

6) 对已部署的可视化软件产品实施增量更新: 当平台对该可视化软件相关数据进行修改时, 将使用增量策略对已部署软件进行远程更新.

根据以上业务需求, 将配置与编译子系统从整个平台中分离出来, 并将其划分为下列的模块, 如图 2.



图 2 系统模块划分

2 数据库设计

配置与编译子系统使用 MySQL 数据库, 由前文的需求与模块分析, 确定了本系统需要设计的数据库表有以下 7 个表: 物体模型表 Object_Models、功能模块表 Fun_Modules、模型_模块关系表 R_Model_Module、软件基本信息表 App_Info、软件_模型表 App_Model、软件_模块表 App_Module、软件_配置表 App_Config.

其中模型_模块关系表 R_Model_Module 用于存储物体模型和功能模块的对应关系, 如机柜模型与机柜故障变色提醒的功能模块间存在的对应关系.

3 关键技术介绍

3.1 模块复用的相关技术

配置与编译子系统以可复用的物体模型和功能模块作为软件配置的基本粒度, 并提供物体模型和功能模块的编译与管理服务.

3.1.1 AssetBundle 技术

在本系统中, 可复用的物体模型是以 AssetBundle 文件的形式被存储和管理的: AssetBundle 是 Unity3D Pro 提供的一项重要功能, 可以将各类资源如三维模型以及对应的材质、贴图等内容都封装并压缩到 AssetBundle 格式的文件中^[3]. Unity3D 引擎可以从外部

加载 AssetBundle 文件并呈现在应用程序中. 本系统通过调用 Unity3D 提供的 BuildPipeline (编译管线)API, 将用户上传的各种三维模型打包为 AssetBunlde 文件.

3.1.2 DLL 技术

在本系统中, 可复用的功能模块以 DLL(动态链接库)文件的形式存储和管理. Unity3D 引擎支持 C#作为脚本语言, 即可以使用 .NET 平台的反射机制动态加载外部 DLL. 在当对一个未知类型的 DLL 文件进行加载时, 反射机制可以从 DLL 文件中获取其中所有的类和其描述信息, 包括方法、属性和事件等关键信息, 使用这些信息, 动态地生成相应的对象^[4].

用户将一组功能相关的脚本文件上传至本系统中, 本系统将调用 .Net Framework 3.5 提供的 csc 命令将脚本编译为 DLL 文件. 在编译过程中, 将添加 UnityEngine.dll、UnityEngine.UI.dll 作为引用, 以保证 DLL 能够在 Unity3D 引擎中正常运行.

3.2 增量更新算法

IDC 领域中所涉及到的房间、机柜、设备等模型精度较高, 体积较大. 软件在部署到生产现场之后, 如果需要修改模型, 如更换贴图或者纹理等操作, 就必须从本系统中远程下载新的 AssetBundle 文件, 这就对网络传输速度提出了很高的要求. 为了解决这一问题, 本系统设计了一种结合编辑距离和分块计算的增量更新算法 AssetDiff 来实现 AssetBundle 的差异更新.

编辑距离(Levenshtein 距离), 是指两个字符串之间, 由一个转成另一个所需的最少编辑操作次数^[5]. 记录计算编辑距离的操作步骤即可获得一字符串转化为另一个字符串的增量文件. 编辑距离计算伪代码为算法 1.

算法 1: 编辑距离

```

输入: X of length n, Y of length m,
输出: D(n, m)
D(i, 0) = i;
D(0, j) = j;
for i = 1; i ≤ n; do
  for j = 1; j ≤ m; do
    D(i, j) = min(D(i - 1, j) + 1, D(i, j - 1) + 1, D(i - 1, j - 1) + if X(i) = Y(j) then 0 else 2);
  );
return D(n, m);

```

分块增量更新算法(Chunk 算法), 是指将文件按照一定长度拆分为块, 以块作为单位计算增量文件. 算

法流程图如图 3.

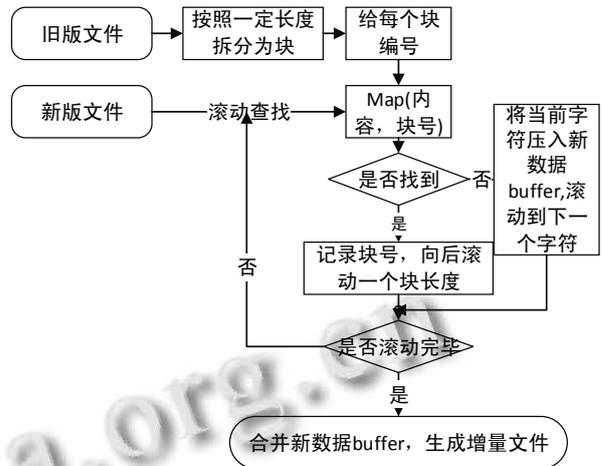


图 3 分块更新算法(Chunk 算法)流程图

编辑距离算法可以精确到字符级别, 但是计算过程中需要使用较大的内存. 而 Chunk 算法中增量更新的精度与块的大小有关, 无法避免冗余更新. AssetDiff 结合了以上两种算法: 设定一个阈值, 当文件长度小于该阈值是通过计算编辑距离获取增量文件, 当文件长度大于该阈值时, 对文件使用分块算法找出两个文件的最长公共连续块, 然后用这个连续块将文件切分为前缀、公共部分、后缀, 分别使用前缀、后缀作为参数递归调用 AssetDiff. 算法流程图如图 4.

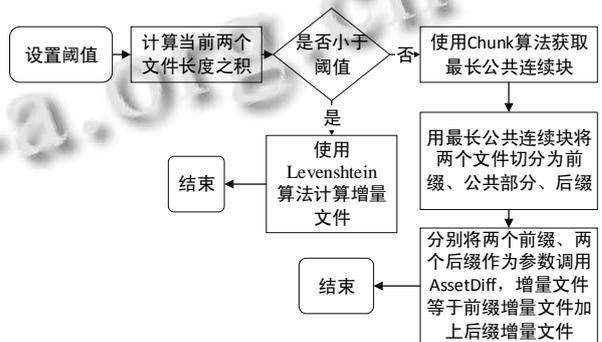


图 4 AssetDiff 算法流程图

一个 AssetBundle 文件中包含有模型、纹理、贴图等多个 AssetObject, AssetObject 的描述结构如下:

```

struct ObjectHeader {
  struct ObjectInfo {
    int pathID;
    int offset;
    int length;
  };
};

```

```

byte classID[8];
};
int ObjectCount;
ObjectInfo Object[];
};

```

其中 pathID 为每个 AssetObject 的唯一编号, 通过偏移量 offset 可以取得该 Object 相对于整个 AssetBundle 文件的位置. 所以, 在调用 AssetDiff 算法计算增量文件之前, 需要先将新旧文件反序列化, 然后按照 offset 进行分块, 剔除掉未发生修改的 AssetObject, 然后将剩余的部分作为 AssetDiff 的参数来计算增量文件.

如图 5 所示, 有两个 AsetBundle 文件 A、B, 其内容为两个完全一样的人物放置在不同材质的地面之上. A、B 文件大小都为 2.3MB, 使用 AssetDiff 算法求 A 变换到 B 的增量文件, 增量文件大小为 19KB, 降低了需要更新的数据量.

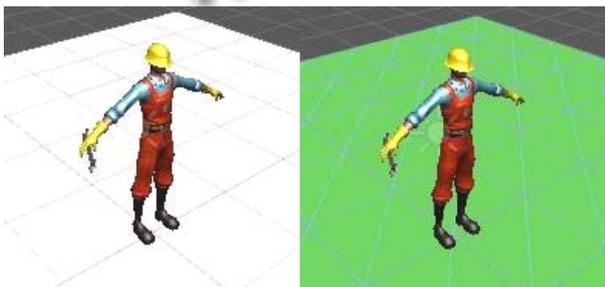


图 5 AssetBundle 文件增量更新示例

4 系统具体实现

基于上述的系统总体架构和关键技术, 实现了 IDC 可视化软件开发平台中的配置与编译子系统.



图 6 系统界面

图 6 所示的界面为软件配置表单, 在该页面中配置该可视化软件的可用功能与相关参数, 如应用的输出目录、UI 风格颜色、外部动态环境传感器数据的 Web Service 路径、监控探头的访问路径等配置项目.

最终, 本系统编译产生的应用效果如图 7 所示.



图 7 已发布软件效果

4 结语

本文主要研究讨论了 IDC 可视化软件开发平台中配置与编译子系统的设计思想与关键技术. 其关键技术之一是利用 AssetBundle 技术与 DLL 技术构建了可视化软件模块化的技术路线. 通过该技术路线实现了模块间的自由拼装. 并为该系统中软件的远程增量更新功能设计了一种结合了编辑距离和分块计算的增量更新算法 AssetDiff, 保证可靠更新的同时降低了冗余更新的数据量, 大大提高了远程更新的效率.

参考文献

- 1 Dillenberger DN, Osias MJ. Management of enterprise systems and applications using three-dimensional visualization technology. US, US7839401. 2010.
- 2 侯亮,唐任仲,徐燕申.产品模块化设计理论、技术与应用研究进展.机械工程学报,2004,40(1):56-61.
- 3 马瑞.三维虚拟数字校园系统的资源封包设计.长江大学学报自然科学版(理工旬刊),2014,(34):49-51.
- 4 陈涛,郝伟.基于.Net 反射原理的 DLL 动态加载机制设计与应用.齐齐哈尔大学学报(自然科学版),2014,(3).
- 5 石野,黄龙和,车天阳,等.基于语法树的程序相似度判定方法.吉林大学学报(信息科学版),2014,32(1):95-100.