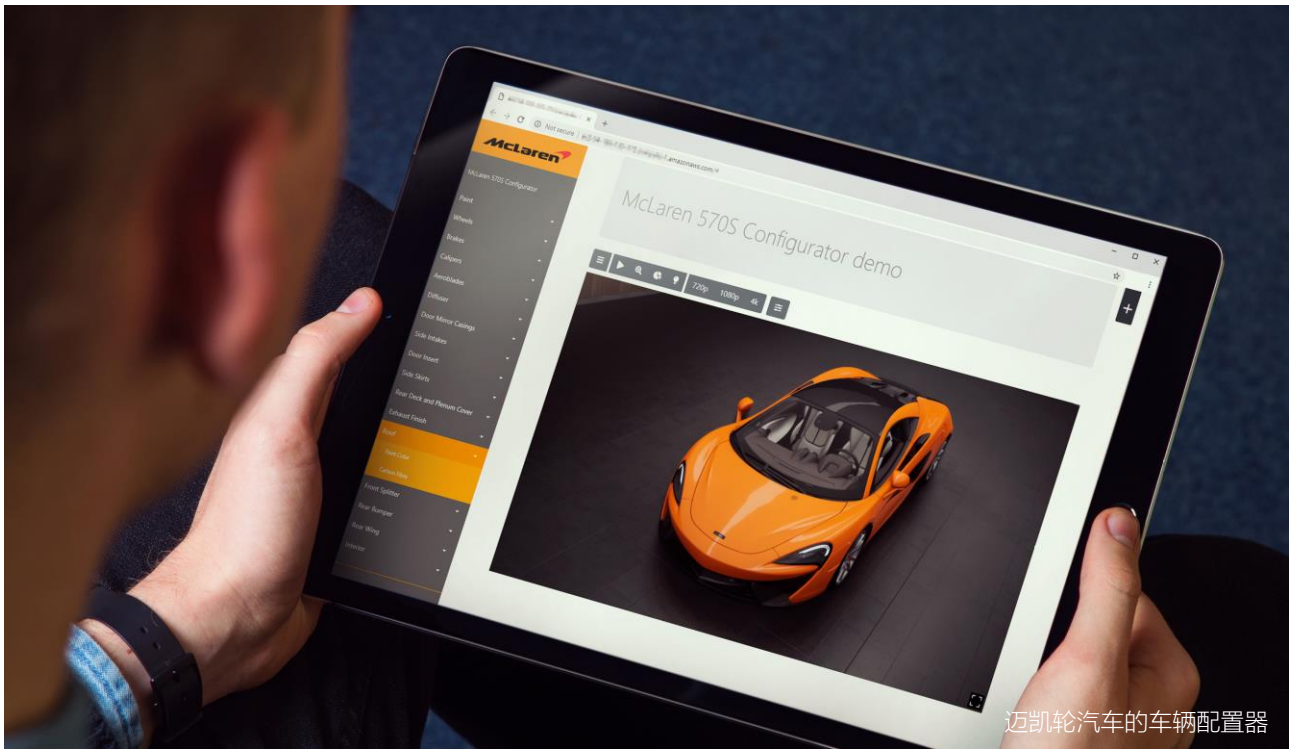
虚幻引擎



迈凯轮汽车的车辆配置器

将虚幻引擎内容流送到多个平台

比较 HTML5、WebGL 和像素流送

目录

简介

在开发联网用户体验时，如何共享内容始终影响着协作、生产和发布中的关键决策。假如用户在消费并与共享内容进行交互时会使用个人电脑、平板电脑、智能手机等各类设备，那么我们就会面临一个关键问题：如何向这些硬件性能各异的平台发布内容。

常见的办法是，首先确定目标受众可能使用的性能最低的平台，然后确定受众能够接受的质量级别，最后以此为基准开发内容。然而，这种方法的弊端是会普遍降低所有最终用户的体验。

而且在这类部署中，负责为最终用户显示实时内容的设备同时还需下载相关的数据和代码，并将结果渲染到屏幕上。将数据下载到设备中容易产生多种问题，比如为了减小下载文件的尺寸或者加快下载速度，有时不得不牺牲一些画面质量。

本文比较了多种可用方案，对允许以像素流送方式共享虚幻引擎体验的方案和工具进行了评估，旨在确保用户在使用基于虚幻引擎的非游戏类应用时，能够获得最高的画面质量和最佳体验。

图片由 Zaha Hadid Architects、Line Creative 和 Epic Games 提供

发布内容时面临的挑战

我们已跨越信息时代，迈入了体验时代；在这个时代中，那些拥有高质量画面、创意十足、并能提供沉浸式体验的内容最容易获得消费者的青睐。这类内容越来越多地开始加入交互式体验，并且引入了 3D 模型以及增强现实或混合现实相关的内容。为了尽可能触及到更多的消费者，开发人员需要让这些内容能在不同的平台上分享，包括手机、平板电脑、个人电脑以及交互式屏幕。

在一些基于 WebGL 或 HTML5 技术的传统或现代解决方案中，显示内容的真实程度和交互体验取决于消费者的设备性能，特别是该设备的硬件性能、显示机制和操作系统。这就意味着如果要让内容尽可能地触及到更多用户，就要以性能最低的设备为基准开发应用程序，并在所有其它平台上共享该应用，或者针对各个平台开发出多个版本的应用程序，以便满足不同平台用户的需求。

在创建流送解决方案时，开发人员面临的一大挑战是如何通过多个通信渠道交付内容，并且无论最终用户使用哪种设备，都能保证画面的真实感以及交互性，并忠实还原品牌的外观和感受。要实现这一目标，就需要将共享应用程序本身与运行它的高端硬件分离开来，让用户的显示设备只负责显示画面。

用户体验和性能因素

在为各类用户设备开发共享应用程序时，需要考虑到多种因素，例如用户体验及其背后的驱动技术。

交互——在成功的部署方案中，用户在浏览器中输入后，服务器必须在规定的时间内给出反馈，否则用户就无法获得流畅的体验。这包括内容在展示前的等待时间。一般来说，非游戏类内容的用户会比那些体验快节奏游戏的玩家更有耐心。例如，在使用车辆配置器时，用户通常不介意一到两秒的操作延迟，也不介意为了看到最终渲染效果而等上 10 秒钟。

播放速度——内容本身可能需要针对实时性能进行优化。首先，你需要界定这些应用程序的受众可接受的播放速度范围。虽然让用户产生沉浸感的公认标准帧率为 60FPS，但有些应用程序可能只需要 5FPS，而其他应用程序则可能需要 90FPS。

图像质量——用户体验到的画面质量和真实程度，通常取决于应用程序端的内容设置（包括实时播放相关因素）、编码质量和硬件的 GPU 性能。画面质量预期以及成本考量会影响商业用例的实现方式，从而确定该设置方案的最终输出效果。

技术因素

主机和数据——从技术上说，分布式体验既可以从客户端运行，也可以从应用程序端运行。但为了避免受到客户端的质量瓶颈限制，同时避免向客户端传输数据时产生时间、存储和安全方面的问题，最理想的方式是从应用程序端进行流送。如果应用程序比较简单并且用户数量有限，那么从单个可访问工作站进行流送通常就足够了。如果应用程序拥有大量用户，并且每个用户都能完全控制一个唯一版本的应用程序，云解决方案就能提高内容的可用性和可延展性。此外，如果某个应用程序十分复杂，并且用户数量很少，那么它同样适用云解决方案。如果采用此方案，开发人员必须根据存储成本谨慎决定内容的数据大小。

用户负载——用户数量和应用程序的复杂程度将决定单个托管实例能够发送哪些内容，以及需要何种负载平衡来响应并发用户。一个拥有高真实度和完全交互式体验的车辆配置器，可能需要一万个拥有相同设置的实例。所需服务器的数量取决于首次交互发生时所需的速度，其往往是一项关键因素。

流送带宽——如果目标用户使用某个特定平台，比如移动设备，那么你可能就需要准备好大小不同的视频流送内容，以保证流送内容不会超出该平台的最大带宽限制。

更新——如何部署更新并修复错误？是替换整个应用程序，还是想办法实时更新内容？你还需要考虑如何在大量实例上大规模更新内容。

安全性——使用用例首先会受到主机和客户端之间的访问情况和安全性因素的影响。如果系统要求对设计审查中的保密数据进行分享和探讨，或者必须在一定时间范围内部署新信息并且要稍后才能访问，那么就有必要在网络或访问站点上实施安全措施。

指标——如果用户行为和场景统计数据属于关键绩效指标，那么你就需要对需要跟踪和保存到分析软件的事件进行定义、激活和关联。

不同发布方案之间的比较

Epic Games 最初在为虚幻引擎（UE4）寻求内容发布方案时，开发团队的想法是在现有技术中寻找方案并以此为基础进行开发。在分析了 WebGL 和 HTML5 这两种方案后，Epic 决定创建一个全新的像素流送插件，以此作为新的解决方案。

在本小节中，我们介绍了团队对 WebGL 和 HTML5 这两种内容发布方案的研究，并着重阐述了它们能否满足上文中提到的需求，最后我们介绍了选择开发像素流送技术的原因。

WebGL

WebGL 是一种基于 JavaScript 的 API 接口，它能渲染交互式 2D 和 3D 图形，并在网页浏览器中显示结果，用户无需下载任何应用程序或插件就能访问内容。WebGL 由 Khronos Group——一个专注于创建免费开放标准的非盈利性组织设计和维护。

WebGL 基于 OpenGL ES（OpenGL 的衍生版本，专用于嵌入式系统）实现，OpenGL 专为智能手机、平板电脑、视频游戏主机和掌上电脑等嵌入式系统设计。WebGL 通过 HTML5 的 Canvas 元素工作，该元素可用于在网站页面上绘制图形。

WebGL 支持基于 GPU 加速的物理模拟、图像处理和效果。

WebGL 的部署内容包含两部分：

- 由 JavaScript 编写的控制代码
- 由 OpenGL ES 着色器语言（OpenGL ES SL）编写的着色器代码

按照其实现方式，WebGL 在部署前不需要编译。

为 WebGL 编写内容

为 WebGL 构建内容的最有效方法就是使用能够显示最终结果的创作软件。这通常意味着所用工具能够通过浏览器进行在线测试。

PlayCanvas——PlayCanvas 编辑器是一个高级 WebGL 编辑环境，其本质上是一种 WebGL 游戏引擎。JavaScript 可用于编写 2D 或 3D 图形模拟。所有的代码都采用 HTML5 编写；作为一种在标准上兼容多种平台的语言，HTML5 能够用于所有主流浏览器和设备。

Sketchfab——Sketchfab 是一个用于分享和发布 3D 内容的在线商城。3D 查看器无需安装插件就能部署在浏览器上，并且支持 VR 和 AR 功能。所有内容必须托管在 Sketchfab 的服务器上并从那里发布内容。

WebGL 的局限性

WebGL 作为一款广泛使用的内容发布工具，它的主要局限在于内容的创建和发送上。内容必须使用 OpenGL ES SL 着色器语言生成，而这种语言目前只能用于在网站上部署的交互式应用程序。如果要部署基于虚幻引擎这类实时引擎的内容，内容和交互元素就必须符合 WebGL 的框架。

此外，WebGL 的显示效果完全依赖于客户端浏览器的功能以及客户端本身的硬件性能；图像质量则取决于浏览器的显示能力。数据必须下载到客户端中，而下载时间决定了用户必须等待一段时间才能开始体验。如果用户的体验内容十分复杂并且包含大量数据，那么客户端还必须保证具有足够的数据储存空间。

如果数据十分敏感或者需要保密，那么还需要在客户端使用额外措施以保证数据安全。

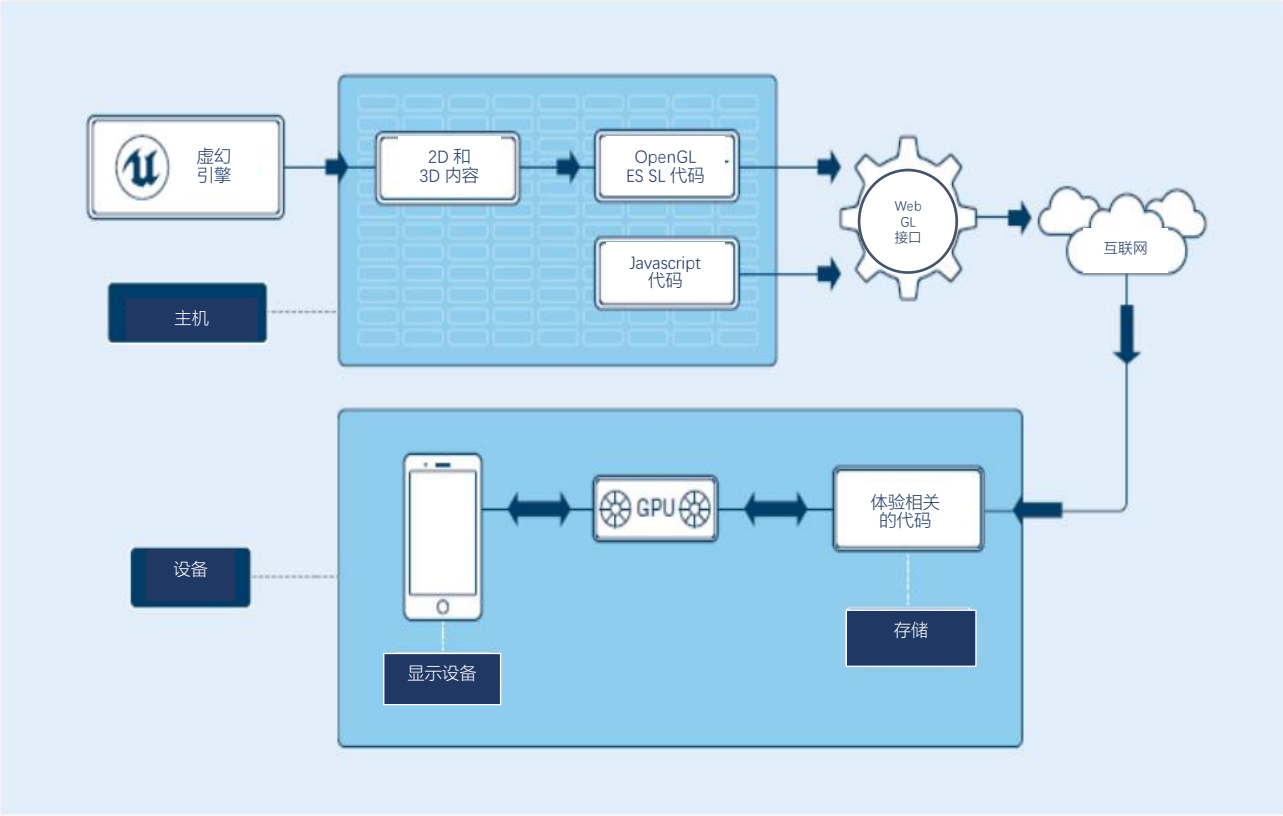


图 1：WebGL 部署中的数据流

如果 WebGL 内容需要与其他途径生成的内容混合使用，那么 WebGL 的画面质量将成为用户体验的界定标准。如果考虑到其它一些平台，则系统必须至少维持两套数据、着色器模型和创意设置，以便用于两种不同的终端通道。

关于指标，可以通过网站的常用通道收集登录和会话时间数据。

尽管有其局限性，但 WebGL 作为一种易于部署的方案，十分适合那些数据量小、安全要求低以及画质要求不高的用户体验。为了支持这类体验，Epic 正在为虚幻引擎开发一种能够直接从 UE4 导出到 WebGL 的选项，导出内容包括 glTF 格式的文件以及一个用于更换着色器的支持库。我们计划在 2020 年第一季度推出该选项。

针对拥有更高要求的虚幻引擎体验，Epic 会继续研发更加合适的解决方案。

HTML5

HTML5 是 HTML（网页开发标记语言）的最新版本。在 HTML5 内部提供了多种解决方案，它们允许开发人员使用 Canvas 元素构建 3D 体验，而不必使用 WebGL。所有这些解决方案都需要开发人员对 GPU 编程有一定了解，并且需要他们知道如何打包数据以供客户端下载，从而导致用户在体验时遇到画质和操作功能方面的局限性。

HTML5 的局限性

在这类方案中，由于应用程序需要在客户端上运行，所以客户端必须在体验开始前下载整个数据集。下载速度和本地可用空间往往是这类方案中的关键因素，为了获得丰富的体验效果，客户端往往需要下载若干 GB 的数据，从而导致漫长的等待时间，并对储存空间提出严苛的要求，对于移动设备来说，这些障碍尤为明显。

与 WebGL 一样，这类方案要求系统维持两套数据标准，并且无法保证用户在查看原始内容和发布内容时，能够获得一致的用户体验。如果要更新和维护发布内容，必须完全替换它，这对于那些需要频繁更新的内容来说非常麻烦。此外，HTML5 和 WebGL 拥有相同的安全性考量。

虚幻引擎内置了将项目发布到 HTML5 平台的工具。不过，从 UE 4.24 开始，Epic 把 HTML5 的支持功能迁移到了 GitHub，只通过社区形式支持该平台，所以官方将不再提供支持。

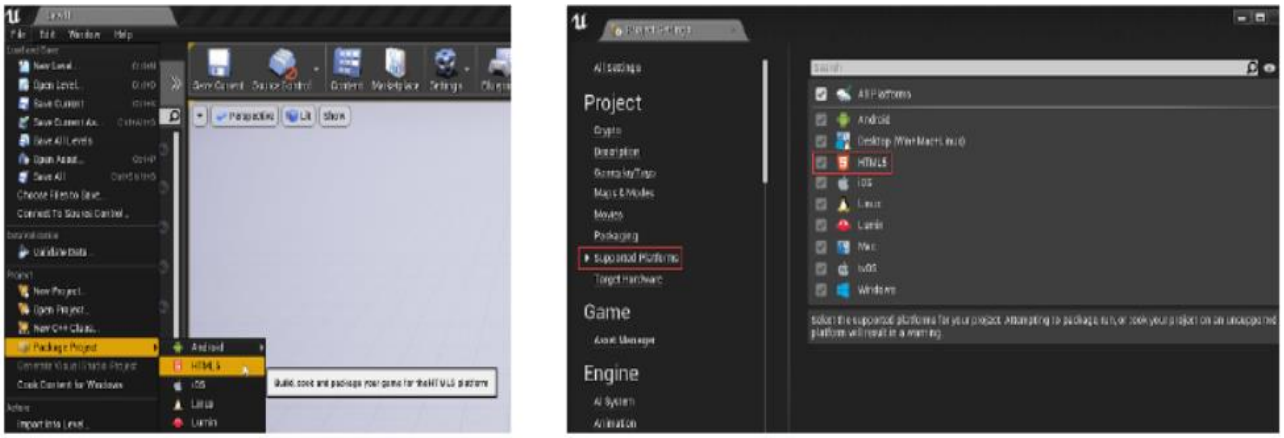


图 2：虚幻引擎界面中的 HTML5 选项

像素流送

在寻找哪种方案最适合将实时内容分发到各种设备上时，Epic Games 分别研究了 WebGL 和 HTML5 的工具和特性。但是考虑到这些技术的局限性，Epic 最终决定自行开发一种技术，即所谓的像素流送。

Epic 认为，一个理想的跨平台交互式实时内容发送系统应该包括以下特性：

- 主机可以向客户端流送像素，同时客户端无需下载任何数据
- 高端和低端用例共享一组数据设置
- 独立于平台的部署
- 所有平台和设备的质量都是确定性的
- 高真实度和高质量
- 能够展示虚幻引擎的所有特性
- 简单明了，易于维护和更新
- 体验刚开始就能快速访问内容
- 在部署前后和部署期间保护数据和设置的安全
- 能为不同用例提供各种潜在配置
- 具备可延展性；可以通过单台服务器或云部署
- 能够捕获用户和会话的数据指标

像素流送以插件形式集成在虚幻引擎中。插件会对主机服务器的图形流送信息进行编码，然后通过 WebRTC 协议将其发送给位于接收端的浏览器和设备。事实上，通过在高性能主机系统上运行虚幻引擎，用户能在所有终端设备上享受到与主机相同的画质，并且能体验到所有的虚幻引擎功能。

由于数据保存在主机上，并且只有像素被流送到查看设备上，所以诸如像素流送这类流送解决方案本质上要比客户端下载方案更快速、更安全。此外，用户会话数据可以在 UE4 中捕获以满足任何指标需求。

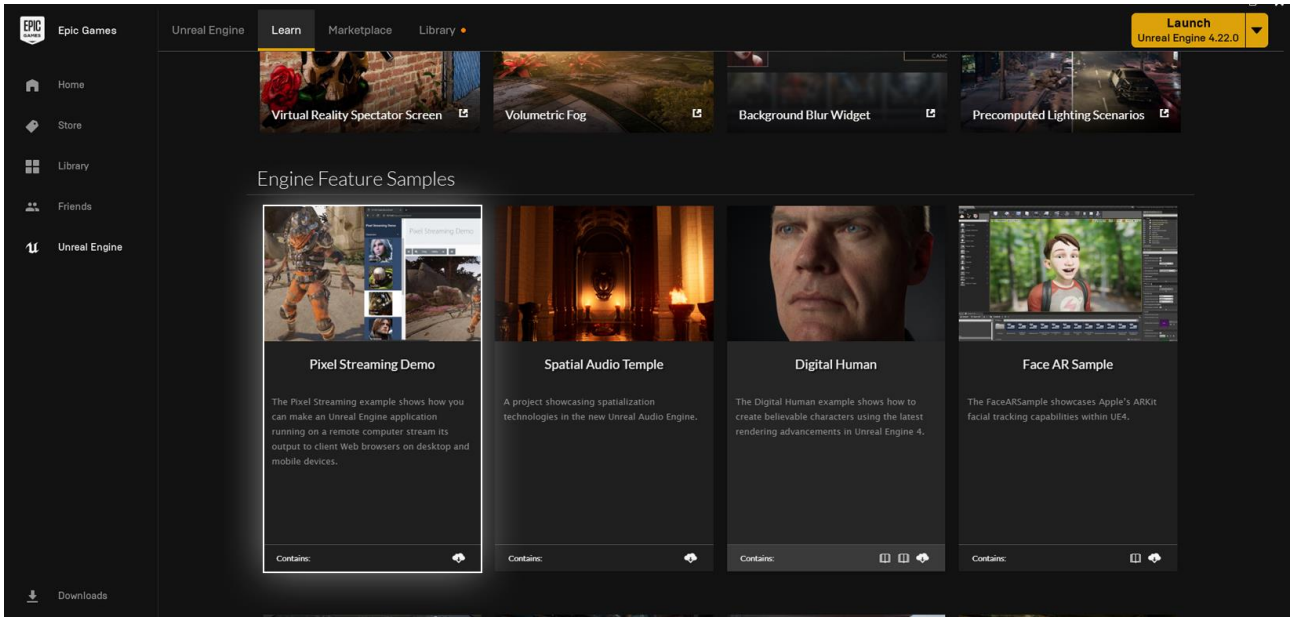


图 3：位于启动器界面中的像素流送插件

WebRTC 协议

WebRTC（网页实时通信）是一种通过网页浏览器和移动应用程序进行实时通信的协议。该协议允许以直接链接的方式传输音频和视频，用户无需下载任何插件或应用程序。通信命令通过 API 接口提交。

要求

像素流送可以通过单台服务器运行，也可以通过允许动态扩展并提供足够硬件的 GPU 云环境运行。在这些情况中，关键之处是对所需规模进行分析，因为这直接关系到最终主机环境的成本以及用户体验的流畅程度。

借助 WebRTC 协议，像素流送插件可以在主机服务器上与网络中的服务器或客户端进行通信。最简单的方式就是通过本地 IP 地址和网络端口 80、8124 和 8888 访问本地主机。

显卡

英伟达的 GPU 自从 Kepler 架构开始都包含一个名为 NVENC 的硬件编码器，它能为视频编码提供独立于图形性能的全面硬件加速功能。通过将涉及编码的复杂计算任务转移到 NVENC 上，图形引擎和 CPU 将能够专注于其他操作。NVENC 使得以下工作成为可能：

- 在不使用 CPU 的情况下，以高质量和超低延迟对游戏和应用程序进行编码和流送
- 为归档、OTT 流送和网络视频提供极高质量的编码
- 单次流送编码的功耗极低（瓦特/流送）

视频编码硬件 NVENC 可通过英伟达的视频编码解码器 SDK 获得。该专用加速器支持对 Windows 和 Linux 上的许多常见视频编码解码器进行硬件加速编码。

如需了解更多信息，请参阅英伟达的视频编解码器 SDK 文档。

随着行业标准的不断发展以及 GPU 和驱动程序的不断普及，虚幻引擎 4.24 版本将支持采用 AMD 显卡进行像素流送。我们将持续探索其他硬件支持方案，以便满足未来版本的需要。

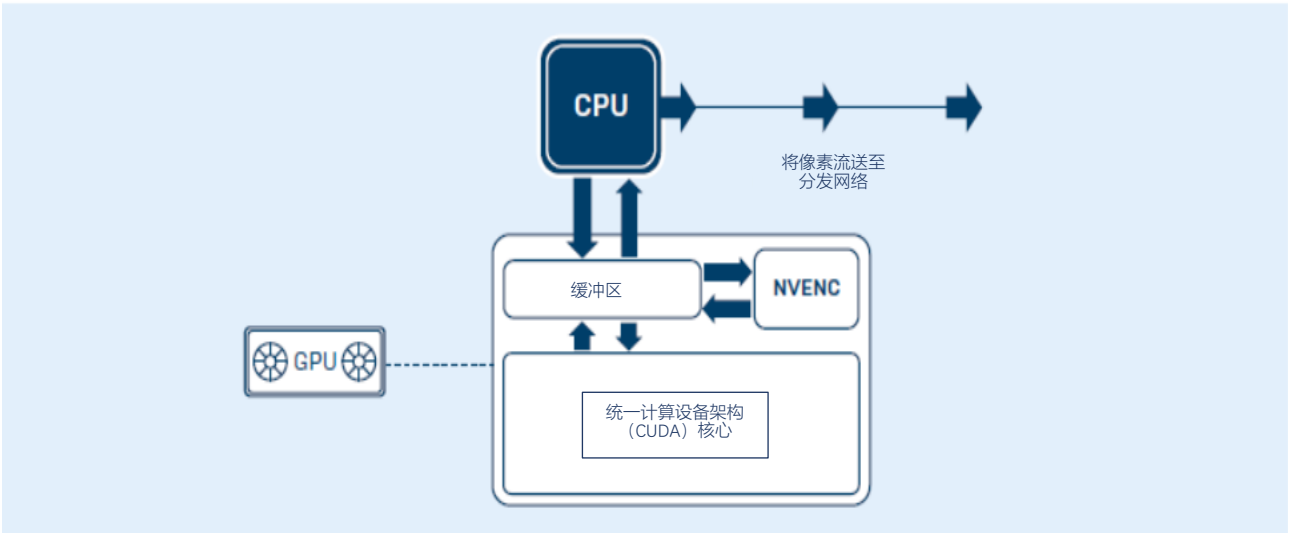


图 4：硬件编码以及像素流送的生成过程。NVENC 编码器通过缓冲区与 CPU 和 CUDA 核心协同工作并生成像素流，然后通过 CPU 发送到分发网络中。

像素流送的用例设置

这里让我们来看一些常见的像素流送设置及其配套技术。如需了解更多信息，请参见虚幻引擎像素流送文档中的"创建主机和网络连接指南"专题。

注意，如果主机和客户端因网络防火墙而无法通信，你可能需要搭建 STUN（NAT 的会话遍历实体）服务器和 TURN（使用围绕 NAT 的中继遍历）服务器为主机和客户端建立通信。由于常规限制，某些客户端的防火墙可能会禁止访问或接收主机服务。如遇到这种情况，STUN/TURN 服务器会尝试各种合理方法来为主机和发出请求的客户端建立有效通信线路。借助这种方式，STUN/TURN 服务器能够为防火墙背后的用户清除障碍，否则用户每次需要使用流送应用程序时，都必须联系系统管理员以获得特殊权限。

局域网中的协作

在这个最基本的共享设置中，主机和需要共享开发环境的客户端（两台或两台以上）都位于同一个局域网（LAN）中。内容会通过内置的服务器插件进行流送，并依赖于 WebRTC 接口在主机和客户端之间建立通信。

注意，如果局域网被防火墙分割，你可能需要使用 STUN/TURN 服务器。

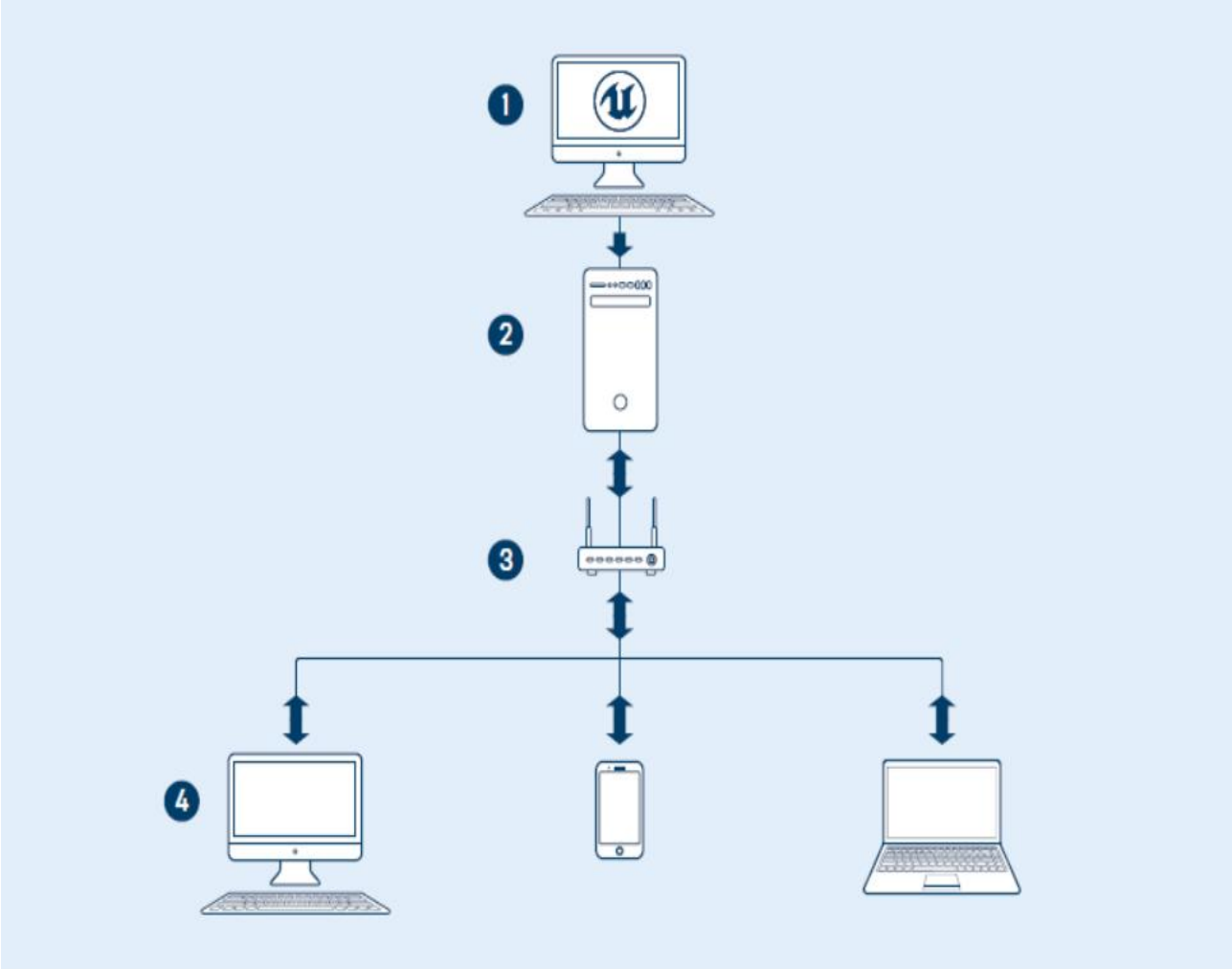


图 5：适用于简易协作的局域网布局：
1：虚幻引擎开发内容 2：局域网中采用 WebRTC 协议的 UE4 服务器应用 3：路由器 4：用户的显示/交互设备

沟通、审核、展示

某些场合中，用户需要使用高保真度的系统来讨论项目，或将内容展示给某些已知的远程客户端（该客户端位于其它局域网中且受防火墙保护），这时像素流送就能提供一种安全的通信方式（无需传输产品数据）。在这类设置中，需要用到 STUN/TURN 服务器来处理主机和客户端之间的通信。

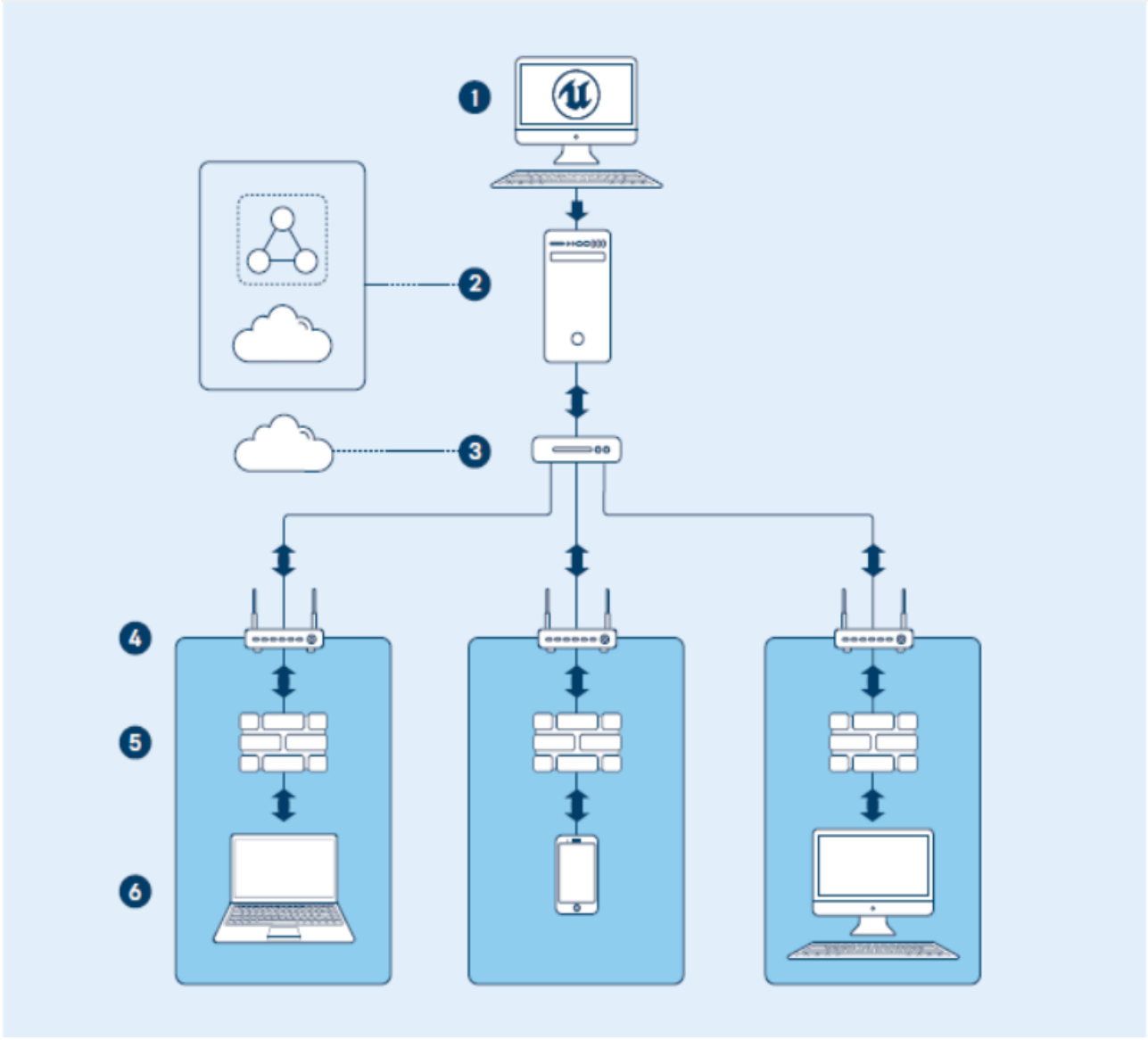


图 6：采用单个接口的连接布局，内容来自本地网络或云

1：UE4 开发内容 2：位于本地网络或云中的 UE4 服务器应用程序 3：云中的 STUN/TURN 服务器 4：路由器/客户端网络入口 5：防火墙 6：显示/交互设备

远程协作设计、共享体验、消费者应用程序、配置器

如果某个应用程序需要触及大量的最终用户，并且这些用户所使用的硬件和软件设置五花八门，那么不妨采用拥有可延展特性的云环境来实现像素流送。图中展示了汽车配置器的常见流送布局，该布局同样适用于对数据访问存在安全要求并且需要跟踪记录的共享体验。此外，这种布局还适用于远程团队协作产品设计。

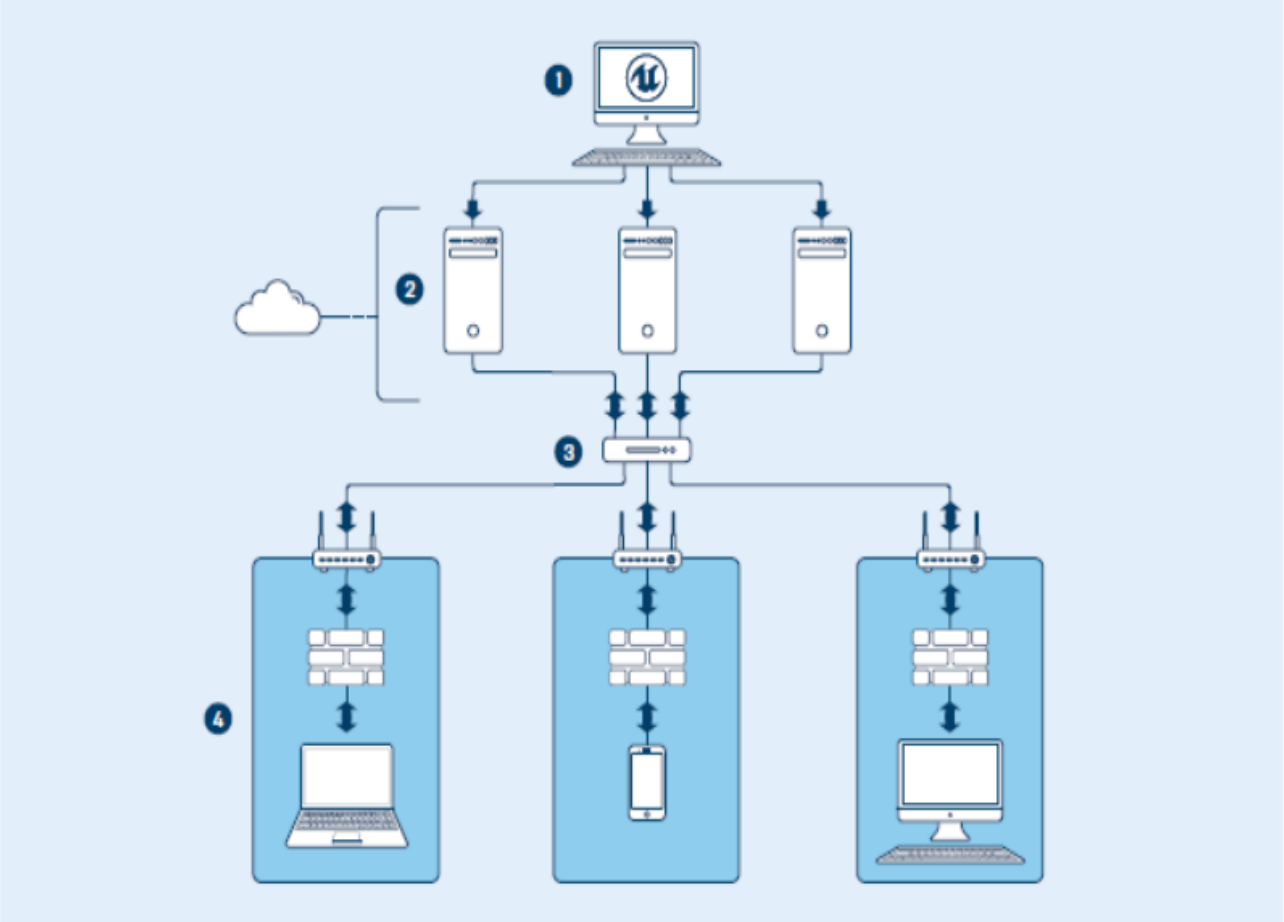


图 7：所有用户都能享受到独一无二的体验

1：虚幻引擎开发内容 2：采用 Web RTC 协议并且位于云中的 UE4 服务器应用程序 3：STUN/TURN 服务器 4：用户显示/交互设备

在大多数商业用例中，不同用户需要不同的交互体验和流送内容。该系统会为每个用户运行一个单独的像素流送组件堆栈，并将每个用户指向一台单独的网络服务器和主机（如有可能）。

每个堆栈都需要一个唯一的标识符和端口来控制体验。许多消费级显卡最多只能同时运行两个编码器，从而限制了电脑上可以运行的实例数量。对于专业级显卡来说，例如英伟达的 Quadro 或 Tesla 系列，或者基于云的 GPU 实例（AWS）则没有这些限制。

匹配器会负责将每个请求者重定向至属于它的信令和网络服务器，从而为客户端及其 WebRTC 代理服务器建立连接。只要用户在服务器上一直保持活跃状态，匹配器就会不断为用户流送内容。匹配器组件可以在虚幻引擎以及其他服务器组件中找到。如需了解更多信息，请访问虚幻引擎像素流送文档，查阅创建主机和网络连接指南专题中的"配对时的多个完整堆栈"一节。

总结

针对如何将高质量的虚幻引擎内容流送到各种设备时，Epic Games 研究了多种方案，并且最终选择通过开发像素流送技术来实现这一目标。

在当前的主机环境中，WebGL 的部署成本低廉，并且能够满足多数简单用例。Epic 正在为虚幻引擎开发一个 WebGL 导出器，不过它只适用于实现一些简单效果。

基于 HTML5 的解决方案则很难带来商业利润，并且该领域的开发尚未涉及商业用途。这些方案的部署将依赖于自定义开发，并且重点将位于目标用例和应用程序上。

像素流送满足了在体验时代流送内容所需考虑的所有用户和技术因素，其重点是让用户在享受到最佳内容的同时又能获得最真实的内容体验。像素流送可以分享高端体验，而且无需担心客户端或平台的限制；今后它将被用于更为复杂的平台布局，并且这些平台布局将能够包含各种用于离线或在线内容的渠道。

相关内容

如需进一步了解 UE4 中的像素流送设置方法，请参阅以下链接中的设置示例：

https://launcher-website-prod07.ol.epicgames.com/ue/learn/pixel-streaming-demo

详细步骤参见下方链接：

https://docs.unrealengine.com/en-US/Resources/Showcases/PixelStreamingShowcase/index.html

关于本文

作者

Heiko Wenczel

贡献人员

Sébastien Miglio

Marco Anastasi

编辑

Michele Bousquet