**System/Subsystem Design Description**

**for the**

**UMBC Virtual Tour 2.0 System**

**Document # CMSC447-05-FA2018-G03-SSDD-01A**

Revision A

16 November 2018

Table of Contents

[**0.1 16 November 2018: CMSC447-05-FA2018-G03-SSDD-01A** 2](#_Toc529786652)

[1 Scope 3](#_Toc529786653)

[**1.1 Identification** 3](#_Toc529786654)

[**1.2 Definitions, Acronyms and Abbreviations** 3](#_Toc529786655)

[**1.3 References** 4](#_Toc529786656)

[**1.4 Document Overview** 4](#_Toc529786657)

[2 Overall Description 4](#_Toc529786658)

[**2.1 System Purpose and Perspective** 5](#_Toc529786659)

[**2.1 System Functions** 5](#_Toc529786660)

[**2.2 User Characteristics** 6](#_Toc529786661)

[**2.3 Constraints** 6](#_Toc529786662)

[**2.4 Assumptions and Dependencies** 7](#_Toc529786663)

[3 System-Wide Design Decisions 7](#_Toc529786664)

[4 System Architectural Design 7](#_Toc529786665)

[**4.3.1 Interface Identification and Diagrams** 7](#_Toc529786666)

[5 Requirements Traceability 7](#_Toc529786667)

[6 Notes 7](#_Toc529786668)

[A Appendixes 7](#_Toc529786669)

**0 Revision History**

**0.1 16 November 2018: CMSC447-05-FA2018-G03-SSDD-01A**

Release A contains the basic system/subsystem design description for the UMBC Virtual Tour 2.0 System.

**1 Scope**

This design description presents the designs used or intended to be used in implementing version 1.0 of a software application enabling virtual tours of the UMBC campus. The designs follow the the requirements identified in the Software Requirements Specification for this project (CMSC447-05-FA2018-G03-SSDD-02A).

**1.1 Identification**

Title: UMBC Virtual Tour 2.0

Abbreviation: VT2

Version Number: 1.0

**1.2 Definitions, Acronyms and Abbreviations**

API Application Programming Interface

CSCI Computer Software Configuration Item

GUI Graphical User Interface

HTML Hyper Text Markup Language

IRC Imaging Research Center

OSM Open Street Map

SIMD Single Instruction, Multiple Data

SRS Software Requirements Specification

SSE2 Streaming SIMD Extensions 2

TCP Transmission Control Protocol

UMBC University of Maryland, Baltimore County

Unity The Unity cross-platform game engine

VCE Virtual Campus Explorer CSCI

VPF Virtual Parking Finder CSCI

VTI Virtual Tour Interface CSCI

VT2 UMBC Virtual Tour 2.0

VUE Virtual Customized Unity Engine CSCI

WebGL The Web Graphics Library, a cross platform JavaScript API for rendering 2D and 3D graphics in a web browser

XML Extensible Markup Language

UDP User Datagram Protocol

**1.3 References**

The following standards apply:

CMSC447-05-FA2018-G03-SSDD-02A https://github.com/noahj1/UMBC-VT-2.0

IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications

MIL-STD-498 Military Standard Software Development and Documentation

UMBC Style Guide https://styleguide.umbc.edu/

Unity User Manual

(2018.2) <https://docs.unity3d.com/Manual/index.html>

WebGL <https://docs.unity3d.com/Manual/webgl.html>

**1.4 Document Overview**

This document is organized as follows: Section 1 identifies the scope of this document and lists the definitions, abbreviations, acronyms, and references used therein. Section 2 provides an overview of the system and a brief description of its architecture.

**2 Overall Description**

**2.1 System Purpose and Perspective**

A virtual campus tour is an important component of a university’s strategy for recruiting students. In an environment of intense competition for students of all types—domestic and international, in-state and out-of-state, and undergraduate and graduate—a strong virtual tour application can convince a prospective student to apply or visit the campus in person. Additionally, virtual campus tours can help current students and visitors navigate their way to their classes or special events.

UMBC currently has several websites that nominally offer virtual tours of the campus. The Undergraduate Admissions UMBC Virtual Tour (located at https://undergraduate.umbc.edu/visit/virtual-tour.php) provides 9 panoramic views of the campus, though it claims to offer 25 views. A virtual tour site for the graduate school (https://gradschool.umbc.edu/discover/vtour/) simply provides a link to the same site that hosts the panoramic campus views noted above. Additionally, a UMBC undergraduate student created a basic virtual tour mobile application for Android devices in 2014 titled, “Introducing UMBC Tours - A Virtual Campus Tour Experience for Android” (https://www.youtube.com/watch?v=zRI61jkUDT4). However, this implementation had extremely limited functionality and did not represent a significant improvement on the applications offered on the UMBC website.

The software described in this specification (the UMBC VT2 system) shall dramatically improve the currently available UMBC virtual tour applications by importing the UMBC campus map and building information into the Unity game engine. It shall allow users to select any location on a three-dimensional map of the campus and allow them to explore it freely. It shall provide browser-based access to this system through a web application. Moreover, the new system shall provide several other useful features, including the ability to highlight valid parking locations based on user status. The system will primarily benefit prospective students seeking to familiarize themselves with the campus environment and current students, faculty, and visitors trying to find their classes or event venues and seeking the best place to park.

**2.1 System Functions**

The system has four primary functions that correspond to the CSCIs specified in section 3:

1. **Virtual Campus Explorer (VCE):** This CSCI shall allow the user to freely explore the virtual UMBC campus map from a chosen starting point.
2. **Virtual Parking Finder (VPF):** This CSCI shall identify parking lots where the user is allowed to park based on the user’s status (faculty member, commuter student, residential student, visitor, etc.).
3. **Virtual Tour Interface (VTI):** This CSCI shall provide a browser-based interface for the VT2 system based on the WebGL framework. The interface shall provide the main menu for the system, allowing the user to select either the VCE function or the VPF function.
4. **Virtual Unity Engine (VUE):** This CSCI shall provide a customized version of the Unity Engine that includes accurate 3D renderings and textures of the UMBC campus buildings. It shall allow natural movement around the campus’ exterior spaces with motion and camera effects generally expected of first and third-person point of view video games.

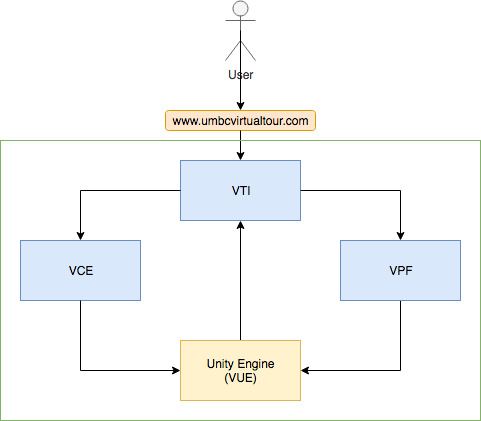


Figure 1 System Architecture

**2.2 User Characteristics**

Users of this system are assumed to possess basic familiarity with internet browsers and websites, but no other special knowledge or skills are required. All user interaction with the system will take place through browser-based menus, and the system will prompt the user to take action with clear and simple instructions when necessary.

**2.3 Constraints**

The system relies on UMBC’s IRC office to provide the object files necessary for creating three-dimensional renderings of the campus buildings. As of late October 2018, the IRC department did not have renderings and textures for some of the buildings. As a result, version 1.0 of the web application will only contain a subset of the 43 buildings that comprise the UMBC main campus.

Additionally, the WebGL framework that allows the Unity engine to be run within a web browser is not currently supported on mobile devices. Therefore, the user must access the system through a desktop computer or laptop. Also, a compatible browser as described in the Unity WebGL manual (https://docs.unity3d.com/Manual/webgl-browsercompatibility.html) must be used. Most modern browsers (Firefox, Chrome, Safari, Microsoft Edge) are supported. Future versions of the system will provide support for mobile computing using a different framework.

Finally, version 1.0 of the software will not provide the user with directions to or from parking lots or buildings. It is intended that the system will provide this capability in a future release.

**2.4 Assumptions and Dependencies**

It is assumed that the CSCIs identified in this version of the SRS are the basic CSCIs necessary to meet customer requirements. Once the software engineering team has successfully implemented these CSCIs, the team may proceed with implementation of “reach goals” such as direction-finding algorithms and mobile support upon agreement with the customer.

No other special assumptions or dependencies have been identified.

**3 System-Wide Design Decisions**

**4 System Architectural Design**

The VT2 system is basically a web server with a customized Unity engine.

- The customized engine is based on Unity Personal version 2018.2.13.

- Exported a map selection containing the UMBC campus from OpenStreetMap (https://www.openstreetmap.org/) as a .map file

- Converted OpenStreetMap .map file to a 3D object model with OSM2World (http://osm2world.org/)

- Import 3D object model of UMBC map into Unity as a .obj file

- Obtain models (.obj, .mtl) and textures (.png, .tga) for 25 UMBC campus buildings from the IRC

- Import models and textures into map in Unity

- Enhance the map with additional features such as grass, water, and tress to increase realism

An **OBJ file** is a standard 3D image format that can be exported and opened by various 3D image editing programs. It contains a three-dimensional object including 3D coordinates, texture maps, polygonal faces, and other object information.

Material library **files** contain one or more material definitions, each of which includes the color, texture, and reflection map of individual materials. These are applied to the surfaces and vertices of objects. Material **files** are stored in ASCII format and have the .**mtl** extension.

Truevision **TGA**, often referred to as TARGA, is a raster graphics **file** format created by Truevision Inc. (now part of Avid Technology). It was the native format of TARGA and VISTA boards, which were the first graphic cards for IBM-compatible PCs to support Highcolor/truecolor display.

**4.1 System Components**

**4.2 Concept of Execution**

**4.3 Interface Design**

**4.3.1 Interface Identification and Diagrams**

**5 Requirements Traceability**

**6 Notes**

**A Appendixes**