Cloud-hosted Simulation-as-a-Service for High School STEM Education

Faruk Caglar, Shashank Shekhar, Aniruddha Gokhale, Satabdi Basu, Tazrian Rafi, John Kinnebrew, and Gautam Biswas

Department of Electrical Engineering and Computer Science
Vanderbilt University, Nashville, TN 37235, USA

Abstract

Despite their advanced status, nations such as the United States of America continue to face a STEM (science, technology, engineering and mathematics) crisis in their education system. Lack of effective teaching modalities that can leverage real-world examples to stimulate student interest in STEM concepts are identified as one of the reasons for this crisis. To address these challenges, our research is investigating the use of innovative and attractive modeling and simulation frameworks for concurrent, interactive and collaborative STEM education where vehicular traffic serves as the real-world example to reify STEM concepts. Existing traffic-related tools, such as traffic simulators, however, do not provide: (1) intuitive abstractions to construct, refine, and simulate various traffic models that are commensurate to the level of high school students, (2) concurrent and scalable model execution, and (3) collaborative learning environments. On the other hand, although intuitive abstractions such as Google Maps exist, these abstractions do not support semantics for dynamic behavior, which is representative of real-world traffic scenarios. To overcome both these challenges and address the STEM problem, this paper presents a Cloud-based, Collaborative, and Scaled-up Modeling and Simulation Framework for STEM Education called C^{2}SuMo. The key contribution of this paper lies in the design

[*] Faruk Caglar

Email address: faruk.caglar@vanderbilt.edu (Faruk Caglar, Shashank Shekhar, Aniruddha Gokhale, Satabdi Basu, Tazrian Rafi, John Kinnebrew, and Gautam Biswas)

and implementation of a cloud-based, elastic modeling and simulation framework that provides an intuitive, model-driven, collaborative, and concurrent visual simulation environment for STEM education. The paper also reports on insights we gained conducting a user study involving over sixty high school students.

Keywords: cloud, modeling, simulation-as-a-service, STEM

1. Introduction

STEM occupations in the United States are projected to grow by 17.0% from 2008 to 2018 [1]. Yet, there is a shortage of students pursuing STEM education in K-12 and undergraduate programs. As a result, there is an urgent need for increasing the number of students pursuing higher education in STEM fields. This can only be achieved by developing and increasing student interest in STEM domains right from their school years. A major contributor to the lack of student interest can be attributed to the inability of existing STEM teaching techniques to relate STEM concepts to the real world [2]. Consequently, students often question the importance and relevance of the STEM education. However, real-world STEM-related problem solving can be complex, and students may need to be scaffolded to help them understand and develop their solutions. Collaborative learning methods are also a useful approach to solving complex problems [3]. However, current tools for education have limitations in both these areas.

On the other hand, many advances have been made that enable advanced experimentation and analysis of real-world phenomena using simulation and emulation environments. For example, advanced simulators are available for flight operations, transportation systems, manufacturing plants, and power grid systems but these tools are designed primarily to be operated by experts. However, we believe that through proper scaffolding and the development of simpler interfaces, these simulation environments can support problem-based STEM education [4]. In other words, the lack of intuitive interfaces and accompanying scaffolding mechanisms have proven to be the biggest barriers to learning by exploration and experimentation in simulation environments. Furthermore, if the interfaces are extended to support collaborative experimentation and made more universally available through web-based services, the increased availability, the emphasis on problem solving, and convenience of use will provide students opportunities to learn and problem solve like experts, at the same time providing teachers an approach to linking the problem solving to fundamental science and mathematics concepts. Also, making these environments available through web-based cloud-hosted service helps individual schools and classrooms to avoid the expenses of acquiring these tools, and also helps students and teachers access to sufficient computing power to study large models and run complex simulations in their school and home environments.

In this paper, we present a novel approach to develop a challenge-based learning environment that supports deep learning and application of STEM concepts by building, analyzing, and problem solving with traffic flow models. As proof-of-concept, we have selected a transportation system simulator called Simulation of Urban MObility (SUMO) [5], which is an open-source, microscopic, continuous simulation package designed to model and simulate real-world traffic flow through large road networks, such as the road networks in cities. The simulation environment is geared to study physical concepts like traffic flow rate, throughput, waiting time, and congestion. Our goal in this project was to develop SUMO into an effective STEM learning tool for studying physics and calculus concepts like speed, acceleration, distance and their relations. Furthermore, analysis of flow pattern also introduces students to additional mathematical concepts, such as distributions associated with variables, and their means and standard deviations.

Three key challenges exist in using SUMO for high school STEM education. First, in concurrence with our motivation outlined above, the abstraction levels provided by SUMO for modeling traffic scenarios are not intuitive for high school students making it hard for them to construct, manipulate, and simulate traffic models, and relate simulation results to STEM concepts. Second, manipulating various parameters requires manipulating low-level file structures in SUMO, which is an unrealistic task to expect high-school students to master. For example, traffic model parameters, such as traffic light logic phases for an intersection can be tracked but cannot be modified through the SUMO user interface (UI). Third, SUMO is also not geared for collaborative experimentation and learning; rather it is designed for a single user interaction by traffic engineers and experts. Despite these disadvantages, its open-source license (i.e., affordability criteria) and its features (i.e., different kinds of experimentation and visualization tools) make a compelling case for using it for STEM education with modeling of traffic flow (something that almost every student in urban areas can relate to) as the compelling real-world domain.

We address these challenges through an intuitive, cloud-hosted interactive, collaborative STEM learning environment that provides students ubiquitous access. The learning environment we have developed, C^{2}SuMo, i.e., **C**loud-based, **C**ollaborative, and **S**caled-**u**p **Mo**deling and Simulation Framework for STEM Education. Our recent work presented preliminary ideas on C^{2}SuMo system [6]. This paper builds on the preliminary ideas, and delves into details of the environment, and how students use it to build models, run simulation experiments, and solve challenge problems. Overall, the work reported in this paper makes the following contributions:

1. The approach to designing the C^{2}SuMo simulation-as-a-service framework, which can elastically employ multiple SUMO simulators in the cloud and expose it as an intuitive and ubiquitously available Google Maps interface to support collaborative STEM learning. Additionally, C^{2}SuMo system incorporates Google Hangouts to enable synchronous collaborative learning.
2. Developing the executable semantics of the SUMO simulator so that the traffic simulation results can be mapped on to the static Google Maps interface.
3. Designing extensions to the Traffic Control Interface (TraCI) to overcome

the underlying cloud-hosted SUMO client's single instance limitation that disallows spawning multiple user connections from a single web server.

4. Providing empirical insights into the performance and scalability of C^{2}SuMo gained from a user study involving more than sixty high school students.

A detailed demonstration and discussion of the effectiveness of C^{2}SuMo in improving STEM learning outcomes, and formal evaluation of the of the role of the user interfaces in supporting STEM learning is outside the scope of this paper. Instead, the key focus of this paper is to discuss the design considerations for a general, scalable cloud-hosted simulation-as-a-service that supports STEM education.

The rest of this paper is organized as follows: Section 2 describes relevant related work comparing it with our contributions; Section 3 describes the technical challenges in realizing C^{2}SuMo simulation-as-a-service and the architecture that addresses these challenges; Section 4 validates the effectiveness of C^{2}SuMo and details the insights gained from a user study; and finally Section 5 presents concluding remarks alluding to future work.

2. Related Work

This section presents relevant related work and compares them with our contributions in C^{2}SuMo. We provide related work along two dimensions: Use of simulations in STEM education and cloud-hosted Simulation-as-a-Service.

2.1. Simulations for STEM Education

The simulation-based approach to learning and problem solving with STEM concepts, adopted by C^{2}SuMo in the context of STEM education is very appropriate as indicated by a recent study [7]. The authors conclude that simulations have a positive impact on student learning, and are an appropriate tool to exploit for improving student achievement in science. A related effort [8] emphasizes the importance of modeling activities in STEM education. Our work in C^{2}SuMo is an instance of these desired capabilities.

More recently, Wilkerson-Jerde, Wagh and Wilensky [9] have developed DeltaTick, a graphical simulation construction interface for the NetLogo modeling environment to successfully integrate simulation and computational methods into K-12 STEM education to help educators maintain balance between (a) addressing curricular content and practices, and (b) attending to student knowledge and interests. DeltaTick achieves this through the use of domain-specific, customizable construction libraries that sit on top of Netlogo [10], and allow learners to assemble and re-assemble predefined sets of behavior blocks to build simulations that represent a particular domain of study. Netlogo was designed as a multi-agent programming language and modeling environment for simulating dynamic behaviors of natural and social phenomena [11]. The modeling task involves programming the behaviors of hundreds or thousands of independent agents that all operate concurrently. This makes it possible to explore connections between micro-level behaviors of individuals (e.g., individual vehicles and their drivers) and macro-level patterns (e.g., traffic flow patterns, such as congestion) that emerge from their interactions. HubNet [12] extends the NetLogo environment to facilitate the running of participatory simulations in classrooms where all students in a class take part in enacting the behavior of a system by controlling a part of the system with a handheld calculator or personal computer. Therefore, HubNet enables a group of learners to learn collaboratively by exploring a simulation environment.

In our own work, we have developed CTSiM [13, 14], a system similar to DeltaTick, which exploits the synergy between Computational Thinking [15] and STEM disciplines to provide a cross-domain, scaffolded, visual-programming and agent-based learning environment to support learning of middle and high school science. Studies run with CTSiM have shown that students simultaneously gain a deep understanding of both computational and science concepts.

Another effort [16] showed that laboratory exercises in Physics alone were not sufficient for students to apply the knowledge to their domain – in their case a medicine-related use case. It is with this viewpoint that C^{2}SuMo integrates real-world problems into the learning activity. However, C^{2}SuMo differs from systems that target physics education [17] in that our focus is on applying the concepts learned to complex problem-solving situations.

Whitman and Witherspoon [18] investigate the use of LEGOs to engage high school students in STEM education. They discuss a mythical aircraft manufacturer using a LEGO airplane factory simulation. Students learn about an application of Industrial Engineering and gain an understanding of the purpose of the discipline. However, there is no direct co-relation between the knowledge gained and the curriculum taught in the high school syllabus.

Mataric et al. [19] describe the use of robotics for experiential and hands-on education by providing real-world meaning to the otherwise abstract knowledge. They attempt to minimize the limitations of hardware cost and experience in constructing a robot. Yet these factors remain a concern in providing affordable STEM education.

Skov-Petersen et al. [20] presented an agent based simulation model which is closest to our approach. The simulation tool, Kvintus.org, has modeling configuration in XML-format and uses Google Maps for displaying the movement of agent models. However, its application to educational systems is difficult to achieve without making considerable changes to the framework.

Youngmoo et. al. [21] designed a web-based game activity to teach mathematical and scientific concepts to students in K-12 grades. In this work, (1) musical instrument timbre, and (2) the cocktail party problems are the collaborative activities to collect the psychoacoustic data and educate students. Students make manipulations on the physics of musical instrument's timbre and other student players are challenged to identify the instrument. In the cocktail party problem, students try to identify certain sounds from a mixture of voices. The work in this paper differs from our work in that (1) only two problems exist in the game environment, (2) the collaboration between students is very limited, (3) it does not force students to identify the problems, and (4) no underlying cloud-based framework exists to expose simulations-as-a-service to the users is addressed. However, providing a collaborative visual learning environment is similar in both projects.

2.2. Simulation-as-a-Service

There has been some work in providing simulation as a cloud service. CSim [22] is designed for modelers, execution operators, analysts and administrators. It provides simulations as web services and includes modules such as user, security, service, simulation resource managers and allocator for managing them. It also presents scheduling algorithms for simulation jobs. mJADES [23] is a framework for executing concurrent simulations in the cloud with support for automatically leasing computing resources from the cloud. It also illustrates the experimental results for performance prediction applications. DEXSim [24] is another platform for executing replicated simulators in the cloud. While C^{2}SuMo also provides simulation-as-a-service, our stakeholders and its purpose are different: we focus on simplifying the modeling and simulation process so that it is easily accessible and usable by high school students in a STEM learning environment.

SUMO has been widely cited in the literature as a simulation engine. Bragard et al. [25] propose dSUMO, which is a distributed version of the SUMO traffic simulator. dSUMO provides the same accuracy as the traditional way of running simulations on a single machine when the problem space is scaled up and computationally expensive. dSUMO aims to handle large scale simulations fast and efficiently by utilizing multi-core, partitioning and running the traffic simulation models in a distributed manner. C^{2}SuMo does not yet deal with partitioning map data, but could benefit from dSUMO in the case that large map data and low-latency are required. dSUMO is a decentralized SUMO traffic simulator and its aim is not necessarily STEM education. Thus, their design goals are orthogonal to ours. For additional scalability, it is possible for C^{2}SuMo to exploit such distributed solutions, which forms part of our future work.

Goncalves et al. [26] presented a mobile-based Advanced Driver Assistance Systems (ADAS) simulation framework using the SUMO simulator and a driving simulator to simulate human factors. It provides a test bed for the developers for testing their applications in the context of Intelligent Transportation Systems (ITS) using a game-based approach. The SUMO simulator is used to control multi-agent driving behavior through TraCI [27]. Providing an intuitive mobile interface is similar to the web-based and mobile-based modeling and simulation user interfaces of C^{2}SuMo. However, we believe this work does not support a cloud-based service. Yet, C^{2}SuMo can benefit from the game-based approach in this related work to teach STEM concepts by modeling and simulation of traffic by assessing in-vehicle information. A web and game-based approach for STEM education is provided by Glasslab (`www.glasslabgames.org/`).

A Google map based traffic information system has been explained by Wu et al. [28]. In addition to the capability of demonstrating traffic flow condition, it also provides functiontionality to interact with users. However, this work lacks the executable semantics and functionality to manipulate the model that will have an impact on the entire system.

3. Design and Implementation of C^{2}SuMo

We now present the design and implementation of C^{2}SuMo, which is a cloud-hosted simulation-as-a-service used as a STEM learning environment for high school students. We first present the technical challenges faced in realizing C^{2}SuMo. We then present the architecture indicating how each challenge is resolved by the architecture. Finally, we present how C^{2}SuMo is used by students.

3.1. Challenges in Realizing C^{2}SuMo

The technical challenges described in this section are drawn from the list of key characteristics that must be illustrated by any cloud-based service as defined by NIST [29] and issues related to making such a service usable by high school students.

3.1.1. Challenge 1: Ease of Use

One of our motivations for not using an existing real-world simulator for STEM education was its perceived complexity for students. The SUMO traffic simulator is no exception to this concern. If a student were forced to use SUMO as an educational tool, he/she would face a steep learning curve, which in turn would further deter the student from pursuing STEM education. As an example of this complexity, SUMO defines map data such as a road, junction, and traffic light networks in XML-formatted network files. These network files are not generic, but are tightly-coupled and formatted specifically for SUMO. Even though SUMO provides several tools such as *netconvert*, *netedit*, and *netgenerate* to manipulate these SUMO-specific files, the SUMO GUI still lacks a toolbox to construct and refine the traffic models [30, 31], not to mention the learning curve in using these tools because the high-school students will need to learn the structure and schema of the SUMO network files.

Our approach to resolving this challenge hinges on masking the complexity of SUMO and instead presenting students with an intuitive interface that they are likely to be already familiar with and which they can easily interact with. Section 3.2.1 describes how we have resolved this challenge by adapting SUMO to Google Maps.

3.1.2. Challenge 2: Ubiquitous Access

A motivation for making a STEM education framework as a cloud-based service is to promote ubiquitous access. C^{2}SuMo aspires to provide the SUMO traffic simulator as a simulation-as-a-service. Its success depends on whether the service is available ubiquitously from a variety of client-side endpoints or not. The ubiquitous access should be such that a user or any third-party developer intending to extend the capabilities should be able to access the internal modeling and simulation control functionalities of the SUMO traffic simulator from a variety of remote endpoints. Our solution to address this challenge leverages web technology as described in Section 3.2.2.

3.1.3. Challenge 3: Performance Scalability and Data Persistence

The SUMO traffic simulator is intended to be used by a single user on their computing resources. However, simulating large traffic simulations will require substantial computing power that may not be readily available with a high school student or even in a school laboratory. Moreover, when different students intend to work on different parts of a large traffic region, it may be required to share a single simulation. However, this capability is not provided by SUMO. A cloud-based approach seems most promising to address these requirements, but we did not find any available solution that allowed a cloud-hosted SUMO simulator that can be elastically scaled up/down depending on the demand, or a solution where a single SUMO simulator could be shared and distributed across a number of users. Moreover, for any learning environment, there is a critical need for logging user actions and persist the students' activities so that students can revisit their earlier attempts and compare how they have made progress. This capability is also needed to understand a student's learning trajectory. None of these capabilities are directly supported by SUMO.

Our approach to resolving this challenge is based on providing a cloud-hosted simulation-as-a-service capability with appropriate adapters to mask the single user restriction. Section 3.2.3 describes how we have resolved this challenge.

3.1.4. Challenge 4: Support for Collaboration

A final challenge for any learning environment is the need for collaboration since students tend to learn better when working in teams and discussing solution approaches with each other. The SUMO traffic simulator enables external applications to control and interact with the traffic simulation through the TraCI protocol. Even though multiple users can connect to a running simulation using TRaCI, the system is not designed for collaboration. Secondly, since students or student groups that are collaborating may be located in different geographic locations, collaborations that enable virtual face-to-face conversation, e.g., via video conferencing, are preferred.

Our approach to resolving this challenge is based on enhancing the TraCI protocol [32] to support collaborative access to the cloud-hosted SUMO and integration with Google Hangouts video conferencing and meeting tool. Section 3.2.4 describes how we have resolved this challenge.

3.2. The C^{2}SuMo System Framework Design

The C^{2}SuMo modeling and simulation framework's high-level design along with its pluggable components is illustrated in Figure 1. C^{2}SuMo's architecture comprises the following key artifacts: (1) model construction, refinement, and simulation system, (2) C^{2}SuMo middleware, (3) distributed database, and (4) model execution modules.

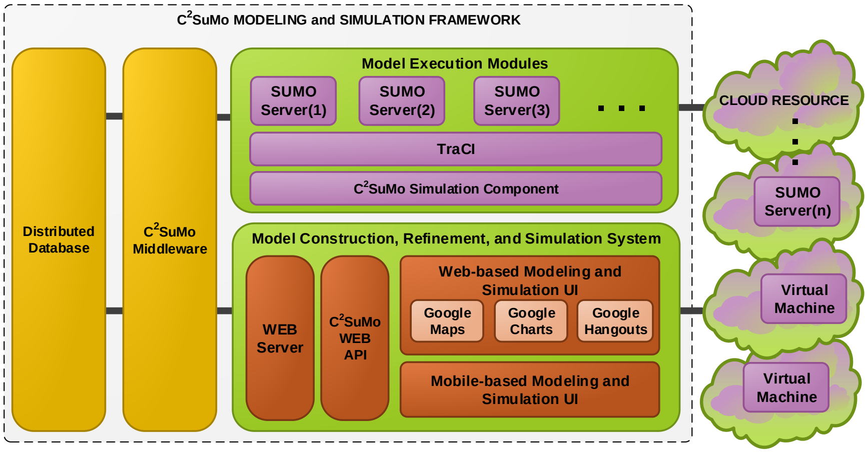


Figure 1: C^{2}SuMo Modeling and Simulation Framework Integrated with Pluggable Components

C^{2}SuMo framework allows students to study the problems in two distinct modes: (1) individual, and (2) collaboration. In individual mode, students model and simulate the problem individually. In the collaboration mode, groups of students collaborate with each other to model and simulate the traffic intersections assigned to them. In this mode, a change in the model of an intersection will have an impact on the other intersections being worked upon by other groups. The rest of this section provides a rationale for the design of C^{2}SuMo and shows how this design resolves the technical challenges outlined in Section 3.1.

3.2.1. Resolving Challenge 1: Ease of Use by Masking Complexity and Utilizing Service Mashups

C^{2}SuMo makes it easy for the system to be used by a student by masking the complexity of the SUMO traffic simulator and offering it in a form that is intuitive and easy to use. C^{2}SuMo hides the complexity of SUMO from the student and offers it as a Google Map interface. Google Maps is chosen because it provides a content-rich, multi-layered, up-to-date aerial imagery map service comprising highways, streets, bus stops, point of interests, and many more that are distributed into separate map layers. Most students are conversant with this interface and find it easier to use.

One drawback of the Google Maps interface is that it is static and does not provide executable semantics that are critical to observe the dynamics of traffic that SUMO is easily able to illustrate. To overcome this problem, C^{2}SuMo leverages the Google Maps API, which is a robust, powerful, and intuitive technology that can be embedded into web pages by using JavaScript or can be extended in a mobile application. All the state-of-the-art functionality that current Google Maps exposes to users can be accessed through the Google Maps API. The traffic dynamics produced by the masked and remotely hosted SUMO simulator are rendered on the user's interface through Javascript-enabled Google Map APIs.

Finally, students require a readily available capability to view results. To that end C^{2}SuMo creates a mashup by integrating with Google Charts. All the results generated by the remotely hosted SUMO simulator are transparently propagated to Google Charts, which in turn renders the results for the user on their browser or mobile platform interfaces. Additional mashups that include Google Hangouts for collaboration are discussed later.

3.2.2. Resolving Challenge 2: Ubiquitous Access using Web Technology

By using the Google Maps interface, C^{2}SuMo uses the web technology that is ubiquitously available via browsers. Thus, rather than having a student directly interact with SUMO, all the modeling, refining, and simulation of different traffic patterns are conducted through web-based and mobile-based user interfaces available through a client-side web browser. C^{2}SuMo also comprises the mobile interface shown in Figure 2, which is an Android application and an extension of Google Maps.

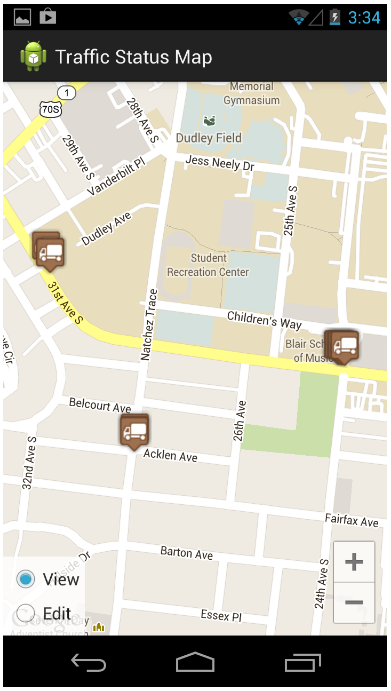


Figure 2: Mobile-based Modeling and Simulation Environment

To further promote ubiquitous access, C^{2}SuMo exposes its internal functionality, such as traffic light programming, turning probabilities, and traffic queue results through a low-level HTTP-based API. This API allows users to access, query, and manipulate the traffic simulation using a traditional web browser. This API can be embedded by client applications that might be interested in utilizing the underlying infrastructure and executing their traffic models in the cloud through the C^{2}SuMo framework.

The web interface of the framework can be run on all the mobile platforms that are able to run a web browser. This mobile application works in two modes: View and Edit. In the view mode, a user receives live simulation data corresponding to his/her group from the SUMO server associated with the appropriate simulation, which is then plotted on the Google Maps interface. It includes vehicles, traffic signals, images, and videos captured by other users. The frequency at which the vehicle location gets updated depends on the network speed and/or user configuration. In the edit mode, a user is provided an option to add/modify traffic signal, video, image, etc.

User made changes are committed to the model execution modules through a web server (described below) and the resulting change to the traffic simulation will propagate to all the clients who can then study the impact on attributes like traffic throughput and congestion. The user also has the option of capturing and uploading images and videos which can then be streamed by other users for coordination in a collaborative environment (also explained below). A mobile application does not support all the functionalities that are available on a traditional web user interface due to limitations of the mobile platform. However, the business logic behind both web and mobile interfaces is same and implements the same aforementioned mobile interface functionality besides capturing image and video.

3.2.3. Resolving Challenge 3: Performance Scalability and Data Persistence using Load Balanced Web Server and Backend Database

The next design question we had to handle was where to host the server-side web application framework comprising the backend SUMO simulation. A cloud-hosted solution was deemed most suited for C^{2}SuMo since it can derive the benefits of elasticity. Since the service is available via the web, C^{2}SuMo comprises a web server to host the dynamic and static content of the system, delivering requests from user interfaces to the communication middleware, and returning the responses to the request initiator. To that end, we have used the Apache web server and CherryPy [33] web framework. The web server also acts as the broker between various C^{2}SuMo components.

The CherryPy web server supports a flexible plugin system and has built-in tools for sessions, static content, caching, etc., which makes it ideal for handling requests from multiple client systems present in our framework. However, we noticed during trial runs that CherryPy was unable to handle high network traffic efficiently, which caused performance degradation. Therefore, we had to switch to Apache as the web server while CherryPy is still used as the web framework.

Figure 1 shows the C^{2}SuMo middleware, which is one of the key components of the C^{2}SuMo framework, and plays a crucial role for traffic modeling and simulation. Specifically, C^{2}SuMo middleware orchestrates the communication traffic between backend model execution modules and front-end user interfaces through the Apache web server.

The actual execution of the simulation models occur within virtual machines (VMs) in the cloud: each VM hosting a single SUMO server. The SUMO traffic simulator [30] is employed as a server application on a VM and the running simulation is accessed through a TCP connection on a specific port. The simulation engine in the model execution module is responsible for creating SUMO configuration files and handling various SUMO server related tasks. SUMO servers simulate the traffic in the backend based on configurations chosen by the user.

To provide scalability, it is mandatory for C^{2}SuMo to communicate, exchange information, and access multiple running simulations using a specific protocol. The TraCI API is a client/server architecture which enables an application to communicate with the running traffic simulation through TraCI protocol. TraCI allows a system to access the object information in the simulation environment and make modifications on objects' behavior at run-time [27]. Attributes, such as vehicle, lane, traffic lights, induction loop, route, junction, edge simulation, GUI values can be retrieved and the state of a lane, traffic lights, vehicle, route, edge, GUI can be manipulated through TraCI.

SUMO TraCI package tools and TraCI4J are two separate implementations of the TraCI protocol that allow developers to communicate with SUMO through Python and Java programming languages, respectively. Due to the maturity and ease of integration, we chose the Python-based TraCI implementation. However, even though the SUMO server can handle several incoming client requests, the default TraCI client implementation inhibits elasticity for the following reason.

The Python-based TraCI supports a singleton module which can communicate with only one instance of a SUMO server at a time. Consequently, although the web server can accept multiple user requests, it is limited to communicate with only one SUMO server running in a VM. Clearly, the default solution would defeat the elasticity criteria that is crucial in a cloud-based environment, such as ours, that spawns multiple SUMO servers based on client requests.

To overcome this technical limitation, we modified the Python-based TraCI implementation to use a class-based approach as opposed to a singleton module-based approach. Doing so enables us to instantiate as many TraCI client instances on the web server as the incoming requests. Consequently, it is now feasible for the web server to connect multiple incoming user requests to their own SUMO servers in parallel that can be spawned on demand inside VMs.

Finally, a key requirement for any learning environment is the ability to log user actions and past sessions. To support these needs, user profiles, intersections of interest, problem definitions, user authentication, and all other information are stored in MongoDB NoSQL database which has distributed, non-relational, and horizontally scalable characteristics. We use MongoDB [34] as our database for persisting media files and documents as well as applications' configuration such as user groups. It is a scalable, high-performance, open-source NoSQL database. It provides document-oriented storage and supports large amount of persistent data and also provides flexible data processing. This fits our need since we are working on huge data set which in future will require high performance processing.

3.2.4. Resolving Challenge 4: Supporting Collaboration through Enhanced TraCI and Google Hangout

One of the key challenges is to allow students to solve problems collaboratively where they can configure and run a common simulation and study the impact of one group's activities on other. Different groups can control different set of intersections, vehicles and traffic flows. Two key requirements must be met to address these needs. First, there must be a way for students to interact with each other, possibly via chat, voice and/or video. Second, the default behavior of SUMO does not provide any method for distributed configuration and control of the simulation.

To address the first need, we have integrated the Google Hangouts tool into the C^{2}SuMo framework to enable students to collaborate with each other by starting a video conference. This functionality allows students to share and exchange ideas, start a video conversation, and share their screens. On the C^{2}SuMo web interface, students see which groups are online/offline under the collaboration menu, and in turn, they can join the conversation and run the simulations collaboratively. C^{2}SuMo also provides the ability to save the simulation history that can be shared with other users for comparison and/or work individually on different scenarios of a common problem.

To address the limitation with SUMO, we allow collaborating students within a group to modify the parameters of the assigned traffic intersection after they have agreed with each other using the Google Hangouts session. Different groups work on different intersections, thus they model the simulation in a distributed fashion. When the simulation is started by the leader of the collaboration groups, all the configuration parameters saved in the distributed database are merged and XML configuration files are generated and passed to a single SUMO server through the web server. Further, the modified TraCI protocol is applied to communicate with the SUMO server during its execution.

3.2.5. C^{2}SuMo Request Flow

Figure 3 illustrates a typical request flow in C^{2}SuMo system. A C^{2}SuMo user uses the web/ mobile interface to view or modify the parameters, which can be to change the problem number, read the description, modify simulation parameters, view results etc. The request is received by the CherryPy request handler in Apache web server which in turn delegates it to the C^{2}SuMo middleware. The middleware contacts the database to save/ retrieve the results which gets sent back to the user via the web server.

The other important request flow is when the simulation gets started. In this

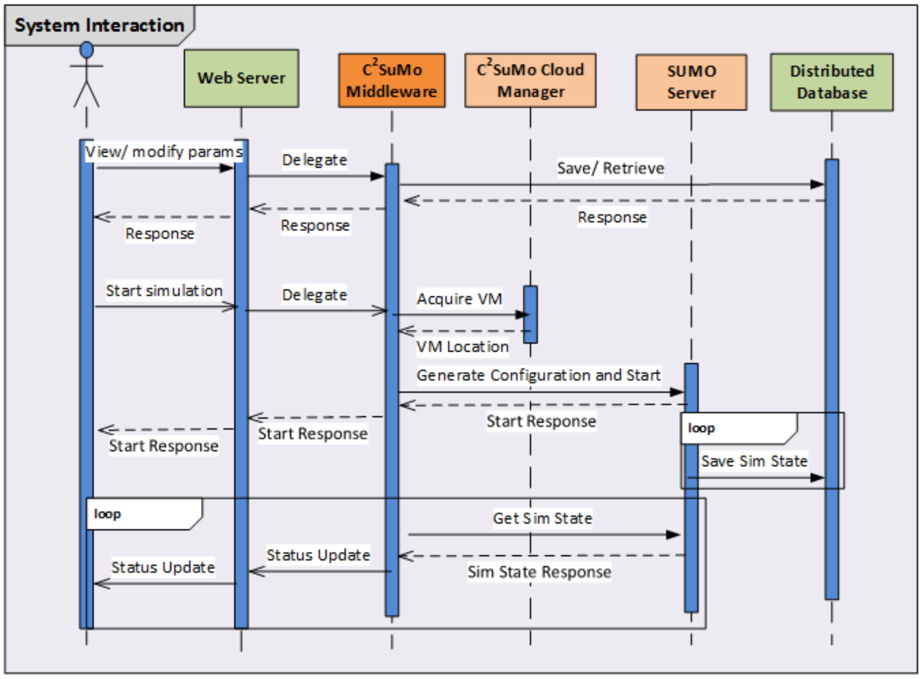
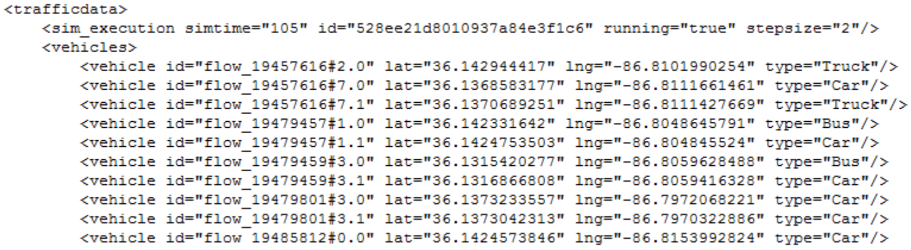


Figure 3: C^{2}SuMo Modeling and Simulation Framework Integrated with Pluggable Components

flow, the user sends the start simulation command to the web server which again gets delegated to the middleware. The middleware in turn contacts the cloud manager to assign a simulation virtual machine (VM). The cloud manager checks if the simulation id has previously assigned a VM, if not then it provisions a VM and returns the address to the middleware. If the cloud manager finds that it has exhausted the quota of VMs, it sends a failure message to middleware. The middleware in turn asks the user to wait for sometime, so that any previously running simulation completes and releases the VM. For the success condition, the middleware contacts the running VM and provides the details to configure the simulation. Simulation gets configured and the simulation is then started. At this point, a response is sent back to the user notifying that the simulation has started.

While the simulation runs, it keeps check-pointing the intermediate results to the database. Also, the middleware at an interval of twice a second queries the simulator using TraCI protocol and sends the vehicle positions, traffic signal, and induction loop data along with the simulation time to the web/ mobile interface which is rendered using JavaScript, this makes the vehicles move on the google maps. The intermediate results are in the form of XMLs, a snippet is shown in Figure 4.



```
<trafficdata>
    <sim_execution simtime="105" id="528ee21d8010937a84e3f1c6" running="true" stepsize="2"/>
    <vehicles>
        <vehicle id="flow_19457616#2.0" lat="36.142944417" lng="-86.8101990254" type="Truck"/>
        <vehicle id="flow_19457616#7.0" lat="36.1368583177" lng="-86.8111661461" type="Car"/>
        <vehicle id="flow_19457616#7.1" lat="36.1370689251" lng="-86.8111427669" type="Truck"/>
        <vehicle id="flow_19479457#1.0" lat="36.142331642" lng="-86.8048645791" type="Bus"/>
        <vehicle id="flow_19479457#1.1" lat="36.1424753503" lng="-86.804845524" type="Car"/>
        <vehicle id="flow_19479459#3.0" lat="36.1315420277" lng="-86.8059628488" type="Bus"/>
        <vehicle id="flow_19479459#3.1" lat="36.1316866808" lng="-86.8059416328" type="Car"/>
        <vehicle id="flow_19479801#3.0" lat="36.1373233557" lng="-86.7972068221" type="Car"/>
        <vehicle id="flow_19479801#3.1" lat="36.1373042313" lng="-86.7970322886" type="Car"/>
        <vehicle id="flow 19485812#0.0" lat="36.1424573846" lng="-86.8153992824" type="Car"/>
```

Figure 4: Traffic Data XML File

3.3. C^{2}SuMo: Administrator and User Roles and Responsibilities

The C^{2}SuMo modeling and simulation framework in Figure 1 as applied to the traffic domain provides high-level traffic components and simulation capabilities that enable students to learn STEM concepts. STEM learners can capture the analysis of traffic patterns they intend to conduct on a large region by subdividing the region into parts, localized solutions generated for each part, and then the individual results combined in a consistent manner to derive a global solution to a traffic problem with the help of the Google Maps. These real-world traffic components that are displayed on the Google Maps are mapped into a cloud-hosted SUMO traffic simulation model where the model is executed, and results are relayed and rendered back on the Google Maps user interface.

Figure 5 depicts the user interaction with the C^{2}SuMo framework. Two types of users are involved. An administrator configures the C^{2}SuMo framework for web and mobile users while a user is the STEM student. The rest of this section focuses on these two roles.

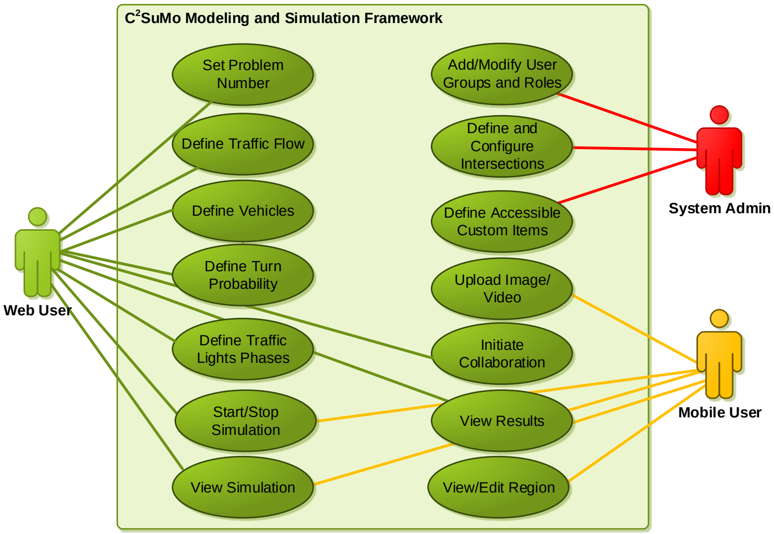


Figure 5: Use Case Diagram of the C^{2}SuMo Framework

3.3.1. Adminstrator Role

Recall that SUMO simulation requires the map data in a certain format. A Geographical Information System (GIS) map exported from OpenStreetMap [35] or any other GIS tool can be converted into a format compliant with the SUMO environment with the help of the *netconvert* application. An administrator is required to configure the learning environment using these tools. The newly generated road networks comprise components such as lane, junction, traffic lights, point of interests, and edges which then will be utilized to model the traffic simulation.

The Mobility Model Generator for Vehicular Networks (MOVE) [31] is another tool in which several realistic traffic models are automatically generated for Vehicular Ad-Hoc Networks (VANETs). This tool is not shown in Figure 1, but was the tool we used during the development process of the C^{2}SuMo framework. The MOVE tool allows rapid generation of the mobility models and simulation scenarios, and incorporates these models into the SUMO simulation environment. Map, vehicle movement, and simulation editors are major components of MOVE and allow generating nodes, edges, vehicle flows, trips, configuration, etc. for SUMO.

The administrator is also responsible for defining the problems that the students will solve as part of the learning experience, and setting up user accounts, user groups and collaboration groups including a leader for each group who is responsible for starting a simulation after the collaborators have agreed on a common solution. Figure 6 illustrates each experimental mode and how they interact with the running simulation.

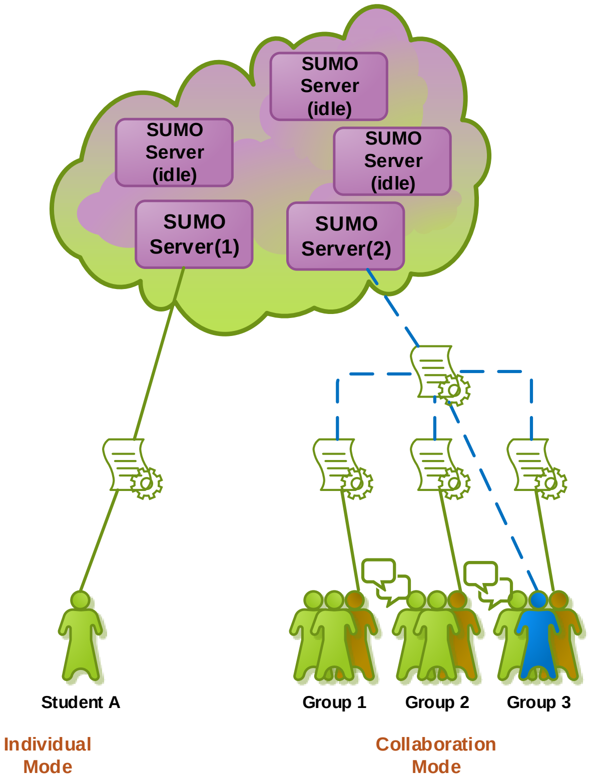
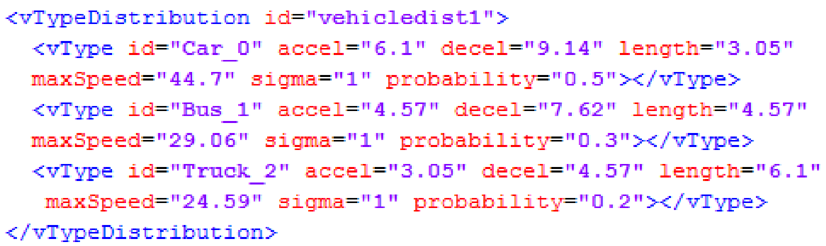


Figure 6: Experimental Modes in the C^{2}SuMo Framework

For C^{2}SuMo to simulate the traffic model in each experimental mode, the necessary configuration files must be generated. These configuration files defined initially by the administrator are in Extensible Markup Language (XML) format and consist of traffic flow, traffic light phases, turning probabilities, and vehicle types. These are the SUMO-specific configuration files and loaded by the SUMO at each time the simulation is executed. One of the SUMO-specific configuration file consists of vehicle distribution and it is illustrated in Figure 7. In the individual mode, student(s) are assigned to only an intersection and the configuration files associated with this intersection are generated. In collaboration mode, each group works on their intersection and simulation is started by the collaboration mode's leader (i.e., shown as the blue person in Figure 6) once all the groups are finished with their work. Only a single instance of SUMO server is employed by a VM in the resource pool in all of the experiment modes. In the current version of C^{2}SuMo, only a single, standalone SUMO map file is used to simulate the traffic by all SUMO servers in the resource pool. Depending on the configuration files generated, same/different scenarios are simulated by each independent SUMO server running in VMs.



```
<vTypeDistribution id="vehicledist1">
  <vType id="Car_0" accel="6.1" decel="9.14" length="3.05"
  maxSpeed="44.7" sigma="1" probability="0.5"></vType>
  <vType id="Bus_1" accel="4.57" decel="7.62" length="4.57"
  maxSpeed="29.06" sigma="1" probability="0.3"></vType>
  <vType id="Truck_2" accel="3.05" decel="4.57" length="6.1"
   maxSpeed="24.59" sigma="1" probability="0.2"></vType>
</vTypeDistribution>
```

Figure 7: XML Format of SUMO Vehicle Distribution File

3.3.2. User Role

Different groups of users have different traffic intersections/regions assigned to them. They are allowed to modify only their region/intersection, however, the impact of configuring the parameters for each individual intersection will be simulated system-wide. It is the job of administrator to define the regions/intersections for the user groups and their roles in it. Also, there are custom objects like traffic signals, turning probabilities, starting/stopping simulation, and speeding up simulation whose accessibility is configured by administrator. Another responsibility of the administrator is defining which users can upload images and videos.

The interaction of mobile and web users with the system is almost identical except for the additional role that the mobile user has to be play to upload images and videos. The users can add/remove custom objects and study their impact on the system. The objects can also be modified, e.g., the traffic signal phases can be changed amongst red, green, and yellow.

To simulate and model real-world traffic scenarios, students interact with the Google Maps embedded web interface of the C^{2}SuMo in Figure 8 and mobile interface in Figure 2 and manipulate a set of parameters shown in Figure 10. Both web and mobile interfaces provide two-way communication with the SUMO traffic simulation and rest of the model execution modules.

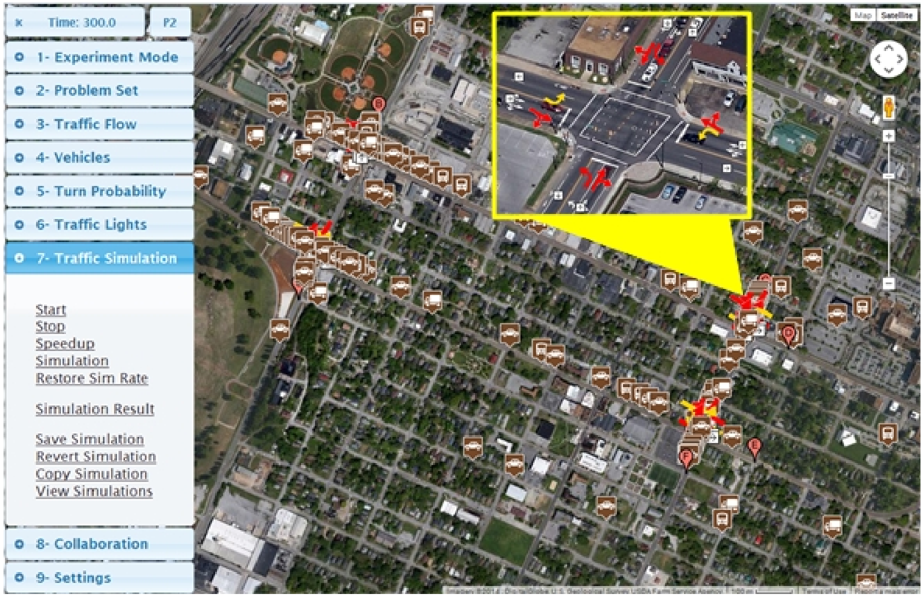
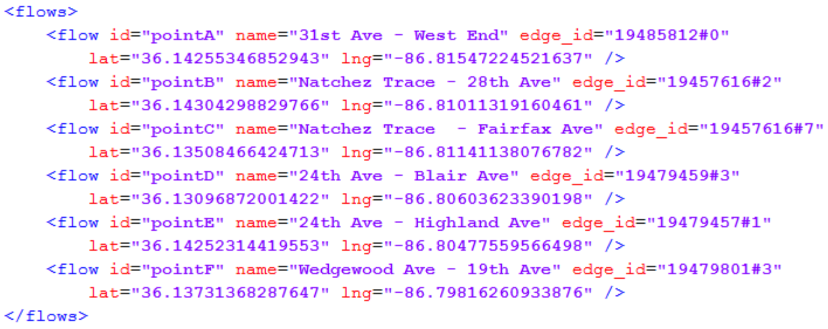


Figure 8: Web-based Modeling and Simulation Environment

The following parameters can be configured in our framework through the web interface:

- **Traffic Flow Starting Point:** The traffic flow starting points where the vehicles are generated and begin their route in the simulation are labeled in alphabetical order. Some of them are visible in Figure 8 such as B, D, E, F. Students enter vehicle generation rate, which is the number of vehicles generated per-hour for each starting point through the form marked with "1" in Figure 10. These traffic flow starting points are stored in the C^{2}SuMo flow file shown in Figure 9. As can be seen in Figure 9, elements such as *edge id*, *latitude*, and *longitude* are defined in this file and each element in this file is later mapped on to the artifacts on web-based modeling and simulation UI.



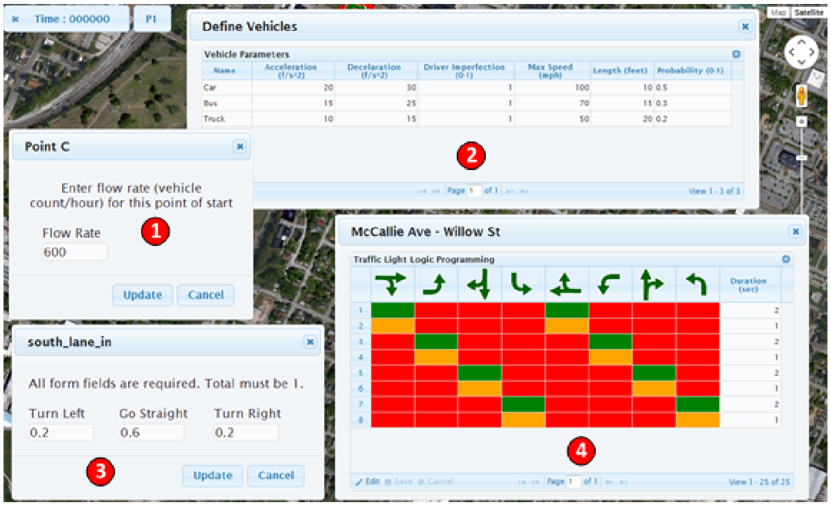
```
<flows>
    <flow id="pointA" name="31st Ave - West End" edge_id="19485812#0"
        lat="36.14255346852943" lng="-86.81547224521637" />
    <flow id="pointB" name="Natchez Trace - 28th Ave" edge_id="19457616#2"
        lat="36.14304298829766" lng="-86.81011319160461" />
    <flow id="pointC" name="Natchez Trace  - Fairfax Ave" edge_id="19457616#7"
        lat="36.13508466424713" lng="-86.81141138076782" />
    <flow id="pointD" name="24th Ave - Blair Ave" edge_id="19479459#3"
        lat="36.13096872001422" lng="-86.80603623390198" />
    <flow id="pointE" name="24th Ave - Highland Ave" edge_id="19479457#1"
        lat="36.14252314419553" lng="-86.80477559566498" />
    <flow id="pointF" name="Wedgewood Ave - 19th Ave" edge_id="19479801#3"
        lat="36.13731368287647" lng="-86.79816260933876" />
</flows>
```

Figure 9: Traffic Flow Starting Points in the C^{2}SuMo Format

- **Vehicles:** Vehicle parameters are defined through the form marked with "2" in Figure 8. This form is accessed through the "Define Vehicles" link under the "Vehicles" menu on the left side of Figure 8. By default, three vehicle types (i.e. car, bus, and truck) along with their default parameters are provided. However, additional vehicle types can easily be added to the simulation via the same form. Parameters, such as acceleration, deceleration, driver imperfection, maximum speed, length, and probability, are updated through this form by the STEM learners and propagated to the simulation by the C^{2}SuMo core framework.

- **Turning Probability:** Turning probabilities for each of the inbound traffic of an intersection are provided through the C^{2}SuMo modeling interface by right clicking on the arrows at a designated intersection in Figure 8 via the form marked with “3” in Figure 10. Left turn, right turn, and straight through are the default directions. Students can study the effects of different turning probabilities on queuing and waiting times, as well as compute the effective and efficient durations of traffic light phases.

- **Traffic Light Logic:** The form marked with “4” on the interface in Figure 10 is used for specifying the traffic light logic for a particular intersection. This form is accessed by clicking on any one of the traffic lights at an intersection in Figure 8. Students manage the traffic light logic at their intersection to control the traffic flow in all four directions by specifying the timing of the traffic light’s red, green, and yellow phases. By running experiments on the traffic light timing, students can learn how to optimize flow through a single intersection or a sequence of intersections. In this way they learn to apply STEM concepts to resolving real world problems that traffic engineers have to deal with.



Define Vehicles

Vehicle Parameters

Name	Acceleration (f/s^2)	Declaration (f/s^2)	Driver Imperfection (0-1)	Max Speed (mph)	Length (feet)	Probability (0-1)
Car	20	30	1	100	10	0.5
Bus	15	25	1	70	15	0.3
Truck	10	15	1	50	20	0.2

View 1 - 3 of 3

Point C

Enter flow rate (vehicle count/hour) for this point of start

Flow Rate
600

south_lane_in

All form fields are required. Total must be 1.

Turn Left	Go Straight	Turn Right
0.2	0.6	0.2

McCallie Ave - Willow St

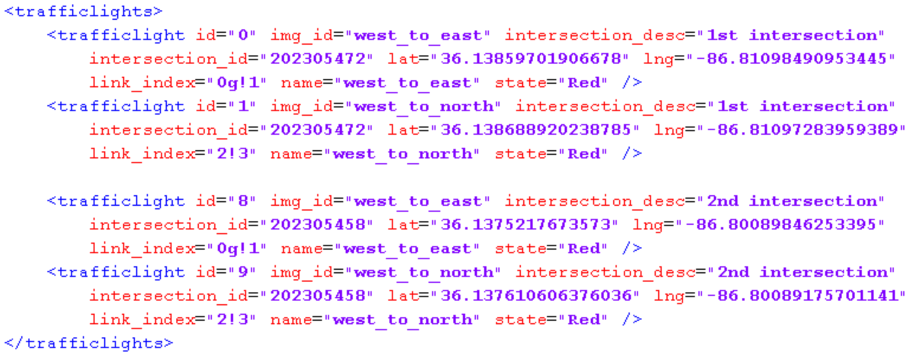
Traffic Light Logic Programming

									Duration (sec)
1	🟩	🟥	🟥	🟥	🟩	🟥	🟥	🟥	2
2	🟧	🟥	🟥	🟥	🟧	🟥	🟥	🟥	1
3	🟥	🟩	🟥	🟥	🟥	🟩	🟥	🟥	2
4	🟥	🟧	🟥	🟥	🟥	🟧	🟥	🟥	1
5	🟥	🟥	🟩	🟥	🟥	🟥	🟩	🟥	2
6	🟥	🟥	🟧	🟥	🟥	🟥	🟧	🟥	1
7	🟥	🟥	🟥	🟩	🟥	🟥	🟥	🟩	2
8	🟥	🟥	🟥	🟧	🟥	🟥	🟥	🟧	1

View 1 - 25 of 25

Figure 10: Web Forms for Traffic Modeling and Simulation

In Figure 11, a snippet from traffic light model transformed into the C^{2}SuMo format is depicted. Various information such as *name*, *intersection id*, *latitude*, *longitude* and *state* is stored in this XML file. Latitude and longitude information are used to place traffic light icons on the Google Maps. State information keeps the actual status of a traffic light at an intersection and is updated by the C^{2}SuMo middleware as the simulation is run. Each intersection is represented as a unique id in the SuMo network file and these intersection ids are mapped with the intersection artifacts on the C^{2}SuMo web user interface.

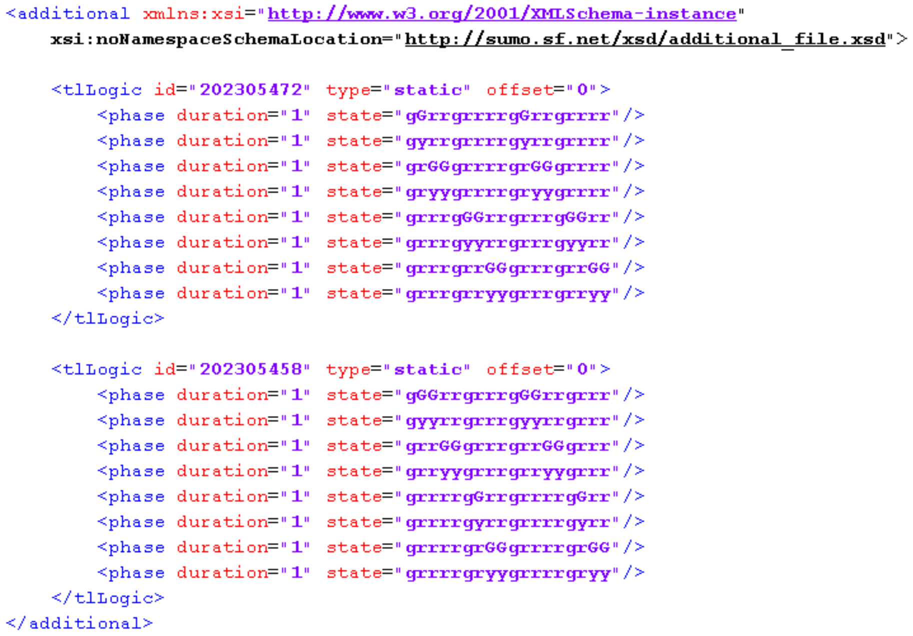


```
<trafficlights>
    <trafficlight id="0" img_id="west_to_east" intersection_desc="1st intersection"
        intersection_id="202305472" lat="36.13859701906678" lng="-86.81098490953445"
        link_index="0g!1" name="west_to_east" state="Red" />
    <trafficlight id="1" img_id="west_to_north" intersection_desc="1st intersection"
        intersection_id="202305472" lat="36.138688920238785" lng="-86.81097283959389"
        link_index="2!3" name="west_to_north" state="Red" />

    <trafficlight id="8" img_id="west_to_east" intersection_desc="2nd intersection"
        intersection_id="202305458" lat="36.1375217673573" lng="-86.80089846253395"
        link_index="0g!1" name="west_to_east" state="Red" />
    <trafficlight id="9" img_id="west_to_north" intersection_desc="2nd intersection"
        intersection_id="202305458" lat="36.137610606376036" lng="-86.80089175701141"
        link_index="2!3" name="west_to_north" state="Red" />
</trafficlights>
```

Figure 11: Traffic Light Model in the C^{2}SuMo Format

Trafic light logic at an intersection is defined as part of additional files in SuMo and a snapshot of it is depicted in Figure 12. Traffic light logic for each intersection is associated with their intersection ids and how traffic signals are switched between different states are represented by phase definitions in this format. Duration element in a phase represents the duration of the phase and state element describes the traffic light states of the associated phase. C^{2}SuMo traffic light model in Figure 11 is mapped into the appropriate artifacts on the web-based modeling and simulation UI. After students updated the traffic light phases, SuMo traffic light logic in Figure 12 is generated by the C^{2}SuMo middleware. Once the simulation is started, these files comprising traffic light logics for each intersection are loaded into SuMo network file.

All of these artifacts on the C^{2}SuMo web user interface are mapped on to the appropriate SUMO traffic simulation artifacts. Therefore, traffic simulation for the surrounding area of an intersection is (1) modeled, (2) mapped into SUMO, (3) simulated by SUMO, and (4) reflected back to the C^{2}SuMo user interface.



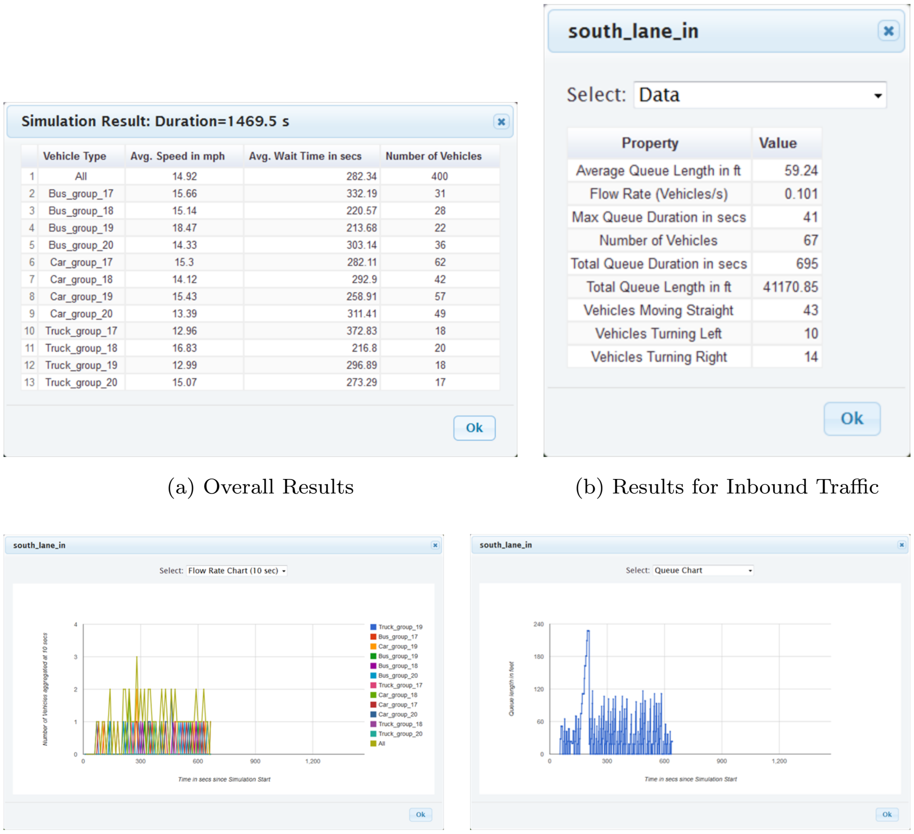
```
<additional xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://sumo.sf.net/xsd/additional_file.xsd">

    <tlLogic id="202305472" type="static" offset="0">
        <phase duration="1" state="gGrrgrrrrgGrrgrrrr"/>
        <phase duration="1" state="gyrrgrrrrgyrrgrrrr"/>
        <phase duration="1" state="grGGgrrrrgrGGgrrrr"/>
        <phase duration="1" state="gryygrrrrgryygrrrr"/>
        <phase duration="1" state="grrrgGGrrgrrrgGGrr"/>
        <phase duration="1" state="grrrgyyrrgrrrgyyrr"/>
        <phase duration="1" state="grrrgrrGGgrrrgrrGG"/>
        <phase duration="1" state="grrrgrryygrrrgrryy"/>
    </tlLogic>

    <tlLogic id="202305458" type="static" offset="0">
        <phase duration="1" state="gGGrrgrrrgGGrrgrrr"/>
        <phase duration="1" state="gyyrrgrrrgyyrrgrrr"/>
        <phase duration="1" state="grrGGgrrrgrrGGgrrr"/>
        <phase duration="1" state="grryygrrrgrryygrrr"/>
        <phase duration="1" state="grrrrgGrrgrrrrgGrr"/>
        <phase duration="1" state="grrrrgyrrgrrrrgyrr"/>
        <phase duration="1" state="grrrrgrGGgrrrrgrGG"/>
        <phase duration="1" state="grrrrgryygrrrrgryy"/>
    </tlLogic>
</additional>
```

Figure 12: C^{2}SuMo Traffic Light Format Mapped into the SuMo Format

As discussed earlier, students can run a number of experiments by various simulation parameters systematically. Figure 13a shows the overall simulation results. By analyzing these values, average waiting time and other metrics in the traffic simulation are captured. In Figure 13b, the simulation results of a particular inbound traffic of an intersection is shown in table format. Flow rate, number of vehicles, average queue length, etc. are analyzed by the metrics in this table. Figure 13c and Figure 13d show the flow rate and queue length of a particular inbound traffic of an intersection, respectively. These are enabled using Google Chart.

4. Experience Using C^{2}SuMo in User Studies

This section presents the insights we gained using C^{2}SuMo in a user study involving sixty four high school students. The goal of this section is primarily to demonstrate the effectiveness of C^{2}SuMo when applied in real settings. We also provide some insights into the learning gains though detailed analysis is outside



Simulation Result: Duration=1469.5 s

	Vehicle Type	Avg. Speed in mph	Avg. Wait Time in secs	Number of Vehicles
1	All	14.92	282.34	400
2	Bus_group_17	15.66	332.19	31
3	Bus_group_18	15.14	220.57	28
4	Bus_group_19	18.47	213.68	22
5	Bus_group_20	14.33	303.14	36
6	Car_group_17	15.3	282.11	62
7	Car_group_18	14.12	292.9	42
8	Car_group_19	15.43	258.91	57
9	Car_group_20	13.39	311.41	49
10	Truck_group_17	12.96	372.83	18
11	Truck_group_18	16.83	216.8	20
12	Truck_group_19	12.99	296.89	18
13	Truck_group_20	15.07	273.29	17

(a) Overall Results

south_lane_in

Select: Data

Property	Value
Average Queue Length in ft	59.24
Flow Rate (Vehicles/s)	0.101
Max Queue Duration in secs	41
Number of Vehicles	67
Total Queue Duration in secs	695
Total Queue Length in ft	41170.85
Vehicles Moving Straight	43
Vehicles Turning Left	10
Vehicles Turning Right	14

(b) Results for Inbound Traffic

south_lane_in

Select: Flow Rate Chart (10 sec)

(c) Flow Rate for Inbound Traffic

south_lane_in

Select: Queue Chart

(d) Queue Results for Inbound Traffic

Figure 13: Simulation Results in Table and Chart Form including Overall Results and for a Specific Inbound Traffic Intersection

the scope of this paper. Based on the experimental results, we also identify the application profile of C^{2}SuMo simulations that can later be used for designing cloud resource management solutions for the framework.

From a pedagogical viewpoint, C^{2}SuMo is part of a larger system, C^{3}STEM, a challenge-based, collaborative, community-situated STEM learning environment that is designed to support the learning of fundamental STEM and computing concepts when combined with scientific and engineering practices that include modeling, simulation, and problem solving. C^{3}STEM is designed to help students learn along with understanding and applying their learned knowledge to solving problems in the real world. It incorporates two core learning environments based on modeling and simulation: (1) C^{2}SuMo, which is the primary topic of this paper, and (2) CTSiM (Computational Thinking using Simulation and Modeling) that supports learning of fundamental physics- and driver-based modeling of vehicle and traffic operation [36, 37]. In our studies, students start with CTSiM, a visual modeling and simulation environment emphasizing computational thinking [38, 15], to learn how vehicle dynamics and driver behavior models can be implemented in an agent-based paradigm. Subsequently, they apply the concepts they have learned to solve more complex problems using the C^{2}SuMo framework.

4.1. User Study System Setup

The user study was conducted with students from two high schools located in Chattanooga, Tennessee as the system users. The total study duration was three hours each day for five days, however, the students worked on C^{2}SuMo only on the last two days since the first three days were spent getting the students acquainted with computational thinking and basic STEM concepts using CTSiM. There were a total of 64 students, with on an average, around 58 students working with the system as some of them were absent on the last two days. The cloud infrastructure was located in our private data center at Nashville, Tennessee, which is roughly 140 miles from Chattanooga. Thus, the Internet was involved for network traffic.

The students connected to `https://c3stem.org` that was preconfigured with four load balanced servers, each dedicated for 16 users. A static load balanced architecture was opted for based on test runs conducted a week before the user studies where the performance was found to be sluggish due to limited network bandwidth between Google Map-based client and the Apache web server as the rate of data transfer for plotting the simulated vehicles was high.

The users were provided with Google accounts that they used to log their simulation results on the Google Drive and also to collaborate with each other using the Google Hangouts-based collaboration platform explained earlier.

Table 1: Hardware and Software specification of Physical Servers

Processor	2.1 GHz Opteron
Number of CPU cores	12
Memory	32 GB
Hard disk	500 GB
Operating System	Ubuntu 14.04 64-bit

Table 1 provides the configuration of each rack server used in our experiments. The four load balanced servers were each assigned 18 virtual machines for running the simulations. The virtual machines resided on three cloud nodes in our private cloud. Our cloud infrastructure is managed by OpenNebula [39] 4.6.2 cloud management software. It consists of one master node and 20 homogeneous cluster nodes. For the experiments, we used three of the cluster servers having the characteristics defined in Table2 in addition to the configuration defined in Table 1.

Table 2: Configuration of Cluster Nodes

Kernel	Linux 3.13.0-24
Hypervisor	Qemu-KVM
Number of Virtual Machines	24
Overbooking Ratio	2.0
Guest CPUs	1
Guest Memory	1 GB
Guest OS	Ubuntu 14.04 64-bit

In our setup, each VM had a C^{2}SuMo simulation component and SUMO 0.19.0 server. Whenever a user started a simulation from the interface, the web server component connected to C^{2}SuMo simulation component that in turn spawned the SUMO simulation based on the user input on one of the VMs. Also, at the end of the simulation, the C^{2}SuMo simulation component connected to the distributed database and saved the final results. We measured the virtual machine performance during the execution of SUMO server and present the results.

4.2. User Study Challenge Problems

The study used a learning activity progression that included a combination of four CTSiM traffic flow modeling and simulation units, followed by four C^{2}SuMo traffic simulation units with associated problems for students to solve. At the start of the study, students were given an overview of computational thinking and modeling of traffic flows, and then they took the hour long pre-test that consisted of seven computational thinking questions and about 12 short answer questions on STEM content. The CT test questions required students to predict outputs of given code snippets and generate algorithms for scenarios described in text form using primitives specified in the questions. Simple questions tested use of a single CT construct, while modeling complex scenarios involved modeling multiple agents using a variety of primitives that included CT constructs like conditionals and loops, and other scenario-specific constructs. This tested students' abilities to use the given abstractions and combine them to generate meaningful algorithms.

The domain pre/post-test assessed whether students understood the concepts important to the traffic domain like the relations between speed, acceleration, time and distance traveled, and the relations between traffic light cycles, vehicular speeds and traffic flow rates and wait times. The test required students well-known physics formulas relating initial and final speeds, time, distance, and acceleration to solve for a given unknown variable, interpreting speed-time graphs, and calculating flow rates and wait times based on traffic light cycles.

After the pre-test, students worked on the four CTSiM units before they switched to working on the three C2SuMo units. While working on these units, the researchers often helped the students one-on-one with their model building activities, and periodically there were discussions on conceptual and problem solving issues in front of the class. After students finished building their models, they used them to conduct simulation experiments, and generated reports to present their solutions to the problem. This data has not been analyzed in detail, but will be completed in the near future. When students were done on day 5, most students took an hour long post-test that was identical to the pre-test.

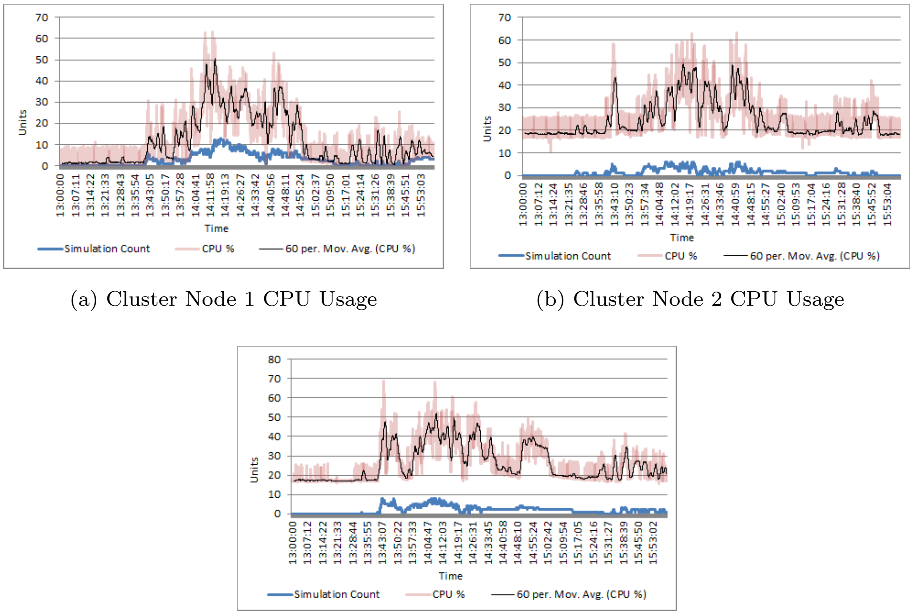
4.3. Insights on System Performance after User Studies

Table 3 displays the details of the C^{2}SuMo simulations performed by the students over the last two days of the study. These simulations were executed on separate virtual machines residing on cluster nodes. Most of the simulations were in individual mode where the students were solving the problems and running the simulations on their own. This mode also resulted in the maximum number of concurrent simulations running on the cluster nodes and our system proved resilient enough to work seamlessly without any significant issue. The average number of simulation steps was highest in the collaboration mode since multiple intersections from different groups were combined to run larger simulations.

Table 3: Simulation Result

Mode	Simulation Count	Avg. Number of Steps
Individual	393	1067
Group	112	1034
Collaboration	15	1347
All	520	1067

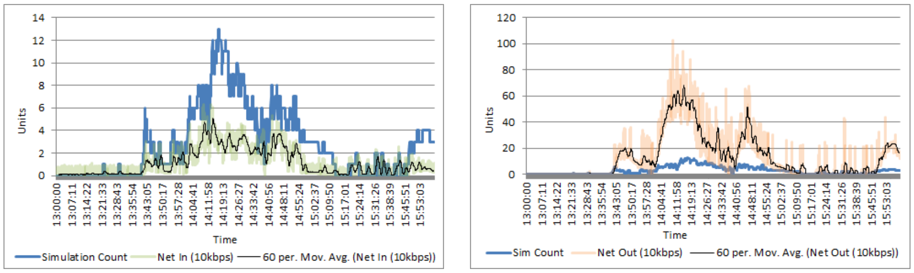
Figure 14a depicts the CPU utilization of cluster node 1 versus the number of simulations executing on that host for day 1 of the experiments. We do not include the day 2 data here as the results were similar. Figure 14b and Figure 14c show the trends for cluster node 2 and cluster node 3, respectively for CPU usage. Figure 15a displays the trend for incoming network traffic and Figure 15b the trend for outgoing traffic for cluster node 1.



(a) Cluster Node 1 CPU Usage (b) Cluster Node 2 CPU Usage

(c) Cluster Node 3 CPU Usage

Figure 14: CPU Utilization on Cluster Nodes



(a) Cluster Node 1 Incoming Network Traffic (b) Cluster Node 1 Outgoing Network Traffic

Figure 15: Network Traffic on Cluster Node 1

The highest number of concurrent simulations were observed on node 1 with 13 simulations. This was the time instant during the two days of the user studies when maximum number of students were active at the same time. As illustrated by Figure 14a, the maximum CPU utilization of 63% was also observed around the same instant for that host. From Figure 15a and Figure 15b, we can infer that network traffic is also correlated with the number of active simulations. However, the outgoing traffic was much higher compared to incoming since the simulation progress was pulled up by web servers to display to the users. We observed similar behavior for the other two hosts and calculated correlation values between the number of simulations and resource utilization as shown in Table 4. From these values we deduce that the C^{2}SuMo simulation application profile is both CPU- and network-intensive.

Table 4: Correlation between Number of Simulations and Resource Consumption

Host	CPU	Network In	Network Out
Node 1	0.84	0.84	0.79
Node 2	0.77	0.53	0.42
Node 3	0.79	0.60	0.51

4.4. Insights on Learning Gains

We analyzed students' STEM and CT learning separately through domain and CT assessments. On the domain assessments, students gained significantly with a moderate effect size (see Table 5), though they had scope for much greater improvement. We also investigated students' performances on individual domain questions and found that there were strong gains on (a) questions requiring application of kinematics formulae relating speed, distance, and acceleration, and (b) a question testing understanding of the concept of gap acceptance for making a safe unprotected left turn. However, students seemed to be having problems with interpreting velocity-time, and distance-time graphs.

On the other hand, the CT assessments did not show much learning gains. There was only one question testing students' understanding of the conditional construct by making them predict the output of a given code snippet, where students showed significant gains ($p = 0.01$). Further investigation showed that there were two clusters of students – one with positive CT learning gains, and the other with negative gains. When we only looked at the first cluster comprising

17 students, we saw that their gains were not only positive but also significant (p < 0.001) with an effect size of 0.68. Since we did not collect any prior knowledge or prior exposure data for the students, it was hard to conclude why we had the two different clusters. Exploring the data further and ensuring that we help all students, irrespective of their prior knowledge, learn both science and CT, is something we will strive for in future studies.

Table 5: Pre-Post Learning Gains for Science and Computational Thinking

N=37	**Pre-score**	**Post-score**	**2-tailed p-value**	**Effect size**
Domain (max=142)	7.07 (8.06)	11.92 (14.34)	0.01	0.42
CT (max=56)	18.93 (11.22)	20 (12.21)	0.41	0.09

At the end of the study that lasted five days, we conducted exit interviews with six students selected by their teachers. The interviews provided us with more detailed information on the effectiveness and impact of C^{2}SuMo, and helped us determine if parts of the system would have to be redesigned to improve its understandability and effectiveness. All of the students interviewed said they liked the collaboration task to solve the challenge problems, and that Google Hangouts was a very convenient tool for collaborations. However, students did note that the total number of students required to collaborate was probably too many, and did not allow sufficient communication, therefore, discussion and equal participation was affected. Students preferred smaller groups and support for more detailed discussion, and joint derivation of problem solutions. This feedback was very useful because in subsequent studies we plan to keep the size of collaboration groups to 2-3 students.

Students found the traffic light programming problem to be very interesting and challenging at the same time. Manipulating the timing sequences of the traffic lights and exploring how a change in one parameter affects the overall city traffic was one of the primary tasks they enjoyed experimenting with. Simulating the city traffic where they live in and controlling the intersections that they always drove through were thought to be very engaging and motivating.

One of the students expressed concern about the performance of the system and had issues about viewing the simulation results late. It takes some time for C^{2}SuMo to save the simulation results and display on the screen after the user clicked on the relevant button. This issue was limited to only one student; others found the system response to be quick and timely. Therefore, we came to the conclusion that the problem observed by one student was likely a client-side issue, and confined to his/her machine being overloaded.

5. Conclusion

To address the STEM education crisis and engage students in effective STEM learning, this paper presented the need to use real-world examples to reify STEM concepts in high school students. To that end this paper proposes the use of advanced simulation tools used in the analysis of real-world systems. To make these tools ubiquitously available to students who can use them in an interactive and collaborative manner, and to overcome any challenges stemming from a steep learning curve associated with these sophisticated tools, this paper presents a cloud-hosted, simulation-as-a-service framework for high school STEM education.

Our framework called C^{2}SuMo (Cloud-based, Collaborative, and Scaled-up Modeling and Simulation Framework for STEM Education) comprises: (1) model construction, refinement, and simulation system, (2) C^{2}SuMo middleware, (3) model execution modules, and (4) distributed database. Through the ubiquitous web-based and mobile-based modeling and simulation interfaces, C^{2}SuMo enables learners to study STEM concepts individually and collaboratively. The C^{2}SuMo framework elastically employs multiple SUMO simulators by spawning them on-demand in the VMs of the resource pool in the cloud and provides simulation-as-a-service.

This paper also presented results from a user study involving sixty-four high school students from which the following insights were learned and opportunities for future work manifested.

- Based on feedback from the students in the user study, our decision to use Google Maps as the user interface was the right decision since it is more conversant to students and easier to manipulate.

- It was deemed that modeling multiple intersections and collaborating between multiple groups with more than five students was not effective. First, the Google Hangout session illustrated performance problems. Second, the real-estate on the display screen is a limitation for video conference among a large collaboration.

- In the current version of C^{2}SuMo, live video streaming of the city traffic was not provided. It would be helpful for the students to observe the live traffic of an intersection and get some sense of the statistics of the traffic via the streaming video. Our future work will investigate on how to provide such data in our system.

- Although C^{2}SuMo has utilized the SUMO simulation, other traffic simulation tools can also be used. In fact, traffic light alone need not be the only use case used; rather other problems from transporation are needed to explain other STEM concepts. Creating a new problem definition requires significant manual effort from the system administrator. Thus, future work should seek solutions to automate many of these tasks.

- The insights gained from the C2SuMo trace logs gathered during the user study helped us to profile the C2SuMo and will later be utilized to design dynamic resource management tools.

- Finally, future research is needed in extending the framework's capabilities for other real-world domains.

The C^{2}SuMo is accessible for users from `https://www.c3stem.org`.

Acknowledgments.

This work was supported in part by NSF CNS SHF 0915976 and CNS CAREER 0845789. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] D. Langdon, G. McKittrick, D. Beede, B. Khan, M. Doms, Stem: Good jobs now and for the future. esa issue brief# 03-11., US Department of Commerce.

[2] B. Barron, D. L. Schwartz, N. J. Vye, A. Moore, A. Petrosino, L. Zech, J. D. Bransford, Doing with understanding: Lessons from research on problem- and project-based learning, Journal of the Learning Sciences 7 (3-4) (1998) 271–311.

[3] G. Stahl, T. Koschmann, D. Suthers, Computer-supported collaborative learning: An historical perspective, Cambridge handbook of the learning sciences 2006 (2006) 409–426.

[4] N. Rutten, W. R. van Joolingen, J. T. van der Veen, The learning effects of computer simulations in science education, Computers & Education 58 (1) (2012) 136–153.

[5] M. Behrisch, L. Bieker, J. Erdmann, D. Krajzewicz, Sumo-simulation of urban mobility-an overview, in: SIMUL 2011, The Third International Conference on Advances in System Simulation, 2011, pp. 55–60.

[6] S. Shekhar, F. Caglar, A. Dukeman, L. Hou, A. Gokhale, J. Kinnebrew, G. Biswas, An Evaluation of a Collaborative STEM Education Framework for High and Middle School Students, in: Poster Paper at 121st ASEE Annual Conference, K-12 and Pre-Engineering Track, ASEE, Indianapolis, IN, USA, 2014.

[7] C. D'Angelo, D. Rutstein, C. Harris, G. Haertel, R. Bernard, E. Borokhovski, Review of computer-based simulations for stem learning in k-12 education (2013).

[8] P. Fishwick, S. Brailsford, S. J. E. Taylor, A. Tolk, A. Uhrmacher, Modeling for everyone: Emphasizing the role of modeling in stem education, in: Proceedings of the 2014 Winter Simulation Conference, WSC '14, IEEE Press, Piscataway, NJ, USA, 2014, pp. 2786–2796.
URL `http://dl.acm.org/citation.cfm?id=2693848.2694202`

[9] M. WILKERSON-JERDE, A. Wagh, U. Wilensky, Balancing curricular and pedagogical needs in computational construction kits: Lessons from the deltatick project, Science Education 99 (3) (2015) 465–499.

[10] U. Wilensky, Netlogo 4.0. 2 user manual, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, Illinois, USA.

[11] S. Tisue, U. Wilensky, Netlogo: A simple environment for modeling complexity, in: International conference on complex systems, 2004, pp. 16–21.

[12] P. Blikstein, D. Abrahamson, U. Wilensky, Netlogo: Where we are, where were going, in: Proceedings of the annual meeting of Interaction Design and Children, Press, 2005.

[13] S. Basu, A. Dickes, J. S. Kinnebrew, P. Sengupta, G. Biswas, Ctsim: A computational thinking environment for learning science through simulation and modeling., in: CSEDU, 2013, pp. 369–378.

[14] P. Sengupta, J. S. Kinnebrew, S. Basu, G. Biswas, D. Clark, Integrating computational thinking with k-12 science education using agent-based computation: A theoretical framework, Education and Information Technologies 18 (2) (2013) 351–380.

[15] J. M. Wing, Computational thinking and thinking about computing, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences 366 (1881) (2008) 3717–3725.

[16] M. Plomer, K. Jessen, G. Rangelov, M. Meyer, Teaching physics in a physiologically meaningful manner, Physical Review Special Topics-Physics Education Research 6 (2) (2010) 020116.

[17] J. L. Docktor, J. P. Mestre, B. H. Ross, Impact of a short intervention on novices' categorization criteria, Physical Review Special Topics-Physics Education Research 8 (2) (2012) 020102.

[18] L. E. Whitman, T. L. Witherspoon, Using legos to interest high school students and imtrove k12 stem education, in: 2013 IEEE Frontiers in Education Conference (FIE), Vol. 2, IEEE, 2003, pp. F3A6–10.

[19] M. J. Mataric, N. Koenig, D. Feil-Seifer, Materials for enabling hands-on robotics and stem education, in: AAAI spring symposium on robots and robot venues: Resources for AI education, 2007.

[20] H. Skov-Petersen, P. Kefaloukos, B. Snizek, Kvintus. org-a choice based agent-based simulation model integrated with google maps, Reseedings for MMV4 (in the present volume).

[21] Y. E. Kim, T. M. Doll, R. Migneco, Collaborative online activities for acoustics education and psychoacoustic data collection, Learning Technologies, IEEE Transactions on 2 (3) (2009) 168–173.

[22] X. Liu, X. Qiu, B. Chen, K. Huang, Cloud-based simulation: the state-of-the-art computer simulation paradigm, in: Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation, IEEE Computer Society, 2012, pp. 71–74.

[23] M. Rak, A. Cuomo, U. Villano, Cloud-based concurrent simulation at work: Fast performance prediction of parallel programs, in: Enabling Technolo-

gies: Infrastructure for Collaborative Enterprises (WETICE), 2012 IEEE 21st International Workshop on, IEEE, 2012, pp. 137–142.

[24] C. Choi, K.-M. Seo, T. G. Kim, Dexsim: an experimental environment for distributed execution of replicated simulators using a concept of single-simulation multiple scenarios, Simulation (2014) 0037549713520251.

[25] Q. Bragard, A. Ventresque, L. Murphy, dsumo: Towards a distributed sumo, First International Conference on Simulation of Urban Mobility, SUMO.

[26] J. S. V. Goncalves, J. Jacob, R. J. F. Rossetti, A. Coelho, R. Rodrigues, Towards a Mobile-based ADAS Simulation Frameworkdoi:{10.5281/zenodo.10206}.
URL http://dx.doi.org/10.5281/zenodo.10206

[27] Traci protocol [cited 09/15/2014].
URL http://sumo-sim.org/userdoc/TraCI.html

[28] Y.-J. Wu, Y. Wang, D. Qian, A google-map-based arterial traffic information system, in: Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE, IEEE, 2007, pp. 968–973.

[29] P. Mell, T. Grance, The nist definition of cloud computing, National Institute of Standards and Technology 53 (6) (2009) 50.

[30] D. Krajzewicz, G. Hertkorn, C. Rossel, P. Wagner, SUMO (Simulation of Urban MObility): An Open-source Traffic Simulation, in: 4th Middle East Symposium on Simulation and Modelling (MESM2002), sn, 2002, pp. 183–187.

[31] F. K. Karnadi, Z. H. Mo, K.-c. Lan, Rapid generation of realistic mobility models for vanet, in: Wireless Communications and Networking Conference, 2007. WCNC 2007. IEEE, IEEE, 2007, pp. 2506–2511.

[32] A. Wegener, M. Piórkowski, M. Raya, H. Hellbrück, S. Fischer, J.-P. Hubaux, Traci: an interface for coupling road traffic and network simulators, in: Proceedings of the 11th communications and networking simulation symposium, ACM, 2008, pp. 155–163.

[33] S. Hellegouarch, CherryPy essentials: Rapid python web application development, Packt, 2007.

[34] K. Chodorow, M. Dirolf, MongoDB: the definitive guide, O'Reilly Media, 2010.

[35] M. Haklay, P. Weber, Openstreetmap: User-generated street maps, Pervasive Computing, IEEE 7 (4) (2008) 12–18.

[36] S. Basu, A. Dickes, J. S. Kinnebrew, P. Sengupta, G. Biswas, CTSiM: A Computational Thinking Environment for Learning Science through Simulation and Modeling, in: Proceedings of the 11th International Conference of the Learning Sciences, Aachen, Germany, 2013, pp. 369–378.

[37] A. Dukeman, L. Hou, S. Shekhar, F. Caglar, J. Kinnebrew, G. Biswas, A. Gokhale, D. Fisher, Modeling Student Program Evolution in STEM Disciplines, in: Poster paper at the 121st ASEE Annual Conference, K-12 and Pre-Engineering Track, ASEE, Indianapolis, IN, USA, 2014.

[38] J. M. Wing, Computational thinking, Communications of the ACM 49 (3) (2006) 33–35.

[39] J. Fontán, T. Vázquez, L. Gonzalez, R. Montero, I. Llorente, Opennebula: The open source virtual machine manager for cluster computing, in: Open Source Grid and Cluster Software Conference, 2008.