Graph-based Modeling and Simulation of Emergency Services Communication Systems

Jardi Martinez Jordan and Michael Stiber
Computing and Software Systems Division
School of Science, Technology, Engineering, and Mathematics
University of Washington Bothell
Bothell, WA, USA 98011
jardiamj@uw.edu, stiber@uw.edu

***Abstract*—Emergency Services Communication Systems (ESCS) are evolving into Internet Protocol based communication networks, promising enhancements to their function, availability, and resilience. This increase in complexity and cyber-attack surface demands better understanding of these systems' breakdown dynamics under extreme circumstances. Existing ESCS research largely overlooks simulation and the little work that exists focuses primarily on cybersecurity threats and neglects critical factors such as non-stationarity of call arrivals. This paper introduces a robust, adaptable graph-based simulation framework and essential mathematical models for ESCS simulation. The framework uses a representation of ESCSes where each vertex is a communicating finite-state machine that exchanges messages along edges and whose behavior is governed by a discrete event queuing model. Call arrival burstiness and its connection to emergency incidents is modeled through a cluster point process. Model applicability is demonstrated through simulations of the Seattle Police Department ESCS. Ongoing work is developing GPU implementation and exploring use in cybersecurity tabletop exercises.**

***Index Terms*—cybersecurity, graph-based simulation, emergency services communications system, GPU computing, cluster point process, Next Generation 911**

I. Introduction

An Emergency Services Communication System (ESCS) encompasses organizational, electronic, and virtual elements that answer emergency calls and coordinate responses. Despite their pivotal role as critical infrastructure, these systems can exhibit vulnerabilities to cyber-attack [1], [2] and extreme circumstances [3], [4]. As such, understanding when and how their proper functioning starts to collapse is essential.

ESCSes are currently undergoing significant infrastructure updates. In the United States, the Next Generation 911 (NG911) initiative is converting the Enhanced 911 (E911) system into an Internet Protocol (IP)-based system [5]. This IP-based ESCS infrastructure is also being implemented in Ecuador [6], Europe [7], and Asia [8]. This promises enhancements to the function, availability, and resilience of these systems by improving inter-connectivity between Public Safety Answering Points (PSAPs) and emergency responders, enhancing call localization, and allowing ESCSes to receive text and multimedia messages on top of traditional voice calls. However, this change increases system complexity and the cyber-attack threat surface.

PSAPs tend to be the main focus of modeling and simulation research because of their critical role of connecting the public with emergency responders. Since PSAPs are emergency call centers, analytical models such as Erlang equations are often used to model their operation [9]. However, the simplicity of analytical models requires that one makes significant and often unrealistic simplifying assumptions; such models can be unsuitable for complex systems, such as ESCSes.

This paper introduces an adaptable graph-based simulation framework and mathematical models that provide a comprehensive solution for ESCS simulation. This includes abstract models for the three discerning aspects of an ESCS: (1) network structure, (2) internal behavior of its main components, and (3) interactions among these components.

We designed our simulation framework using our general-purpose simulator for graph-based systems, Graphitti [10]. ESCS components are depicted as a graph-based structure supporting Communicating Finite State Machines (CFSMs). Each vertex in this graph represents an FSM operating within a discrete event queuing model; each edge denotes a communication link. Currently a sequential implementation, the architecture is designed to facilitate GPU implementation.

II. Related Work

Previous research on ESCSes has predominantly focused on data analysis and development of predictive models aimed at identifying incidents and call patterns; simulation remains relatively limited and unexplored. Reference [11] analyzed 35 papers delving into 911 data within policing contexts, revealing two main methodological approaches. One approach relies on standardized metrics such as call volume, types, and response time, offering a comprehensive yet less detailed perspective. The second approach employs complex techniques to model caller behavior, track call-type patterns over time, and identify obstacles to prompt responses. Notably, none of the studies in this review took advantage of simulation.

This work is in part sponsored by the National Security Agency under Grant Number H98230-20-1-0314. 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 Security Agency. This work is in part an outcome of InterPARES Trust AI, an international research partnership led by Drs. Luciana Duranti and Muhammad Abdul-Mageed, University of British Columbia, and funded by the Social Sciences and Humanities Research Council of Canada (SSHRC).

The little work that uses simulation primarily concentrates on vulnerabilities to cyber-attacks such as DDoS [1], [2]. The only exception is [12], who employed simulation within Operation Management of an Emergency Call Center. This research introduced a method for modeling burst behavior in the call arrival process that, although different than the cluster point process presented here, also leverages the link between emergency calls and the incidents that trigger them [13]. Reference [12] studies important aspects of ESCSes such as the non-stationarity of call arrivals, stochastic call-taker behavior, and the relationship between the geographical distance between PSAPs and their service area and service time. However, their simulations were tailored to a Swedish emergency call center provider, requiring extensive rework for other emergency call centers. In addition, their work recognizes the importance of call abandonment but does not include it in their models.

Reference [1] researched the vulnerability of the 911 system to DDoS attacks by anonymous unblockable calls originating from mobile phones. They conducted practical assessments by deploying bots within a limited cellular network. They then simulated the potential impact of a DDoS assault on an E911 network, finding that fewer than 6000 bots could disrupt emergency services throughout a state for extended periods. Their simulation employed a lumped discrete event simulation of the busiest time of the day to mirror the potential overload of Selective Routers due to a botnet; however, it lacked depth in representing the comprehensive behavior of 911 entities such as PSAPs and responders.

Reference [2] also studied DDoS attacks, using a cyber-physical queuing-network model of the 911 system. Their simulation included models for a time-varying Poisson process for call arrival, an exponentially distributed service time, dispatching dynamics, responder travel time, and response time. However, their study omitted other important processes, such as call abandonment and redial, and focused on a small area of 10x15 miles (Charlotte, NC) with no discussion of model scalability.

III. Contributions

Rather than implementing a one-off simulation, we have developed a first-of-its kind, generalized framework for simulating graph-structured systems and applied it to ESCS operation. Our framework specifies graphs using GraphML, a standardized graph representation language. Call arrival and other stochastic processes are specified using Extensible Markup Language (XML). This provides flexibility for simulating heterogeneous ESCS networks and adapting diverse mathematical models for the stochastic processes. Modeling call arrivals is especially complex due to its multifaceted nature, including inter-day, intra-day, and seasonal variability [13]. A novel contribution of our mathematical models is the modeling of call arrivals as the realization of a cluster point process, capitalizing on the connection between emergency calls and the underlying emergency incidents. Moreover, our work delves into aspects of ESCSes frequently overlooked by other models, such as call abandonment, redial, and emergency personnel dispatch.

IV. The 911 Response Process

The 911 response is a dynamic process with multiple players. It starts with the person in need, who makes the initial call for help that connects them to a call-taker at the nearest PSAP. The call-taker enters essential details into a system that pinpoints the caller's location and categorizes the type of emergency. From there, a dispatcher, who might be the call-taker in some cases, sends the right emergency responders to the scene of the incident. Once the responders are on-site, they provide the necessary assistance. But it does not end there; the system keeps gathering valuable data. Responders file reports, cross-check their assessment with the call-taker's information, and record response times [11].

A. Call Initiation

Beyond connecting callers with the appropriate PSAP, telephony service providers are mandated to transmit the caller's location — a process that varies in form and precision depending on the calling technology: wireline phones, wireless cellular devices, or Voice over Internet Protocol (VoIP) services. For wirelines, this would typically be a physical address; for wireless, it could involve GPS in the handset or cell tower information, which could result in location inaccuracy of kilometers [14].

B. Call Routing and Delivery

Call routing aims to link the caller with the appropriate PSAP. Previously, the emergency call was directed to the nearest PSAP based on the caller's location, retrieved from the local telecommunications network. However, the evolution to NG911 is ushering in a revised routing mechanism reliant on Geographic Information System (GIS) data [15]. NG911 relies on GIS databases that contain the service boundaries of PSAPs and emergency responders, leading to the redirection of calls to the PSAP serving the caller's precise location.

The new routing framework is tailored to modern mobile calls and internet calling services, integrating wireline and older wireless services through a Gateway, thus standardizing their protocol. Furthermore, a Location-to-Service Translation (LoST) protocol server unifies location information, either civic addresses or geographic coordinates, into a Uniform Resource Identifier (URI) that is used to route the emergency call. Ultimately, all emergency calls go through the same LoST protocol and routing engine.

C. Call Processing and Dispatch

After an emergency call reaches a PSAP, call-takers follow a specific protocol to assess the emergency, determine necessary services, and provide information for responders [11]. While this protocol might vary among PSAPs, the National Emergency Number Association (NENA) outlines minimum requirements, suggesting that call-takers should gather critical details such as the incident's address or precise location, a

callback number, the nature of the call, the time it occurred, any identified hazards, and the caller's identity [16].

NENA also establishes standards for the time intervals between each step of the process. According to these guidelines, NENA recommends that 90% of calls should be answered within 15 seconds and 95% of calls within 20 seconds.

The call processing phase culminates ideally in dispatching an emergency response unit to the incident. Dispatch responsibilities vary between jurisdictions, often involving specialized dispatchers within certain PSAPs. Once essential information is gathered, call-takers proceed to either dispatch the appropriate response unit or transfer the call to a designated dispatcher. This dispatching process involves critical decisions regarding the types and quantity of response units required. All pertinent information must be communicated and logged into a Computer-Aided Dispatch (CAD) system that assists responders in swiftly identifying the required type and speed of response [11].

D. Incident Response

The primary goal of an ESCS is ensuring prompt and appropriate response. The interval between the initial call and a response unit arrival at the scene is known as the *response time*. This time frame is critical within an ESCS and serves as a key metric for evaluating performance, as noted in the work by [17]. Reference [18] supports this by linking a decrease in 911 response times to a significant reduction in homicide rates.

Once a response unit is dispatched, it remains occupied for the combined duration of the response time, the time it takes to travel to the site of the incident, and the period dedicated to delivering necessary services at the emergency scene. This delivery of emergency services marks the final stage of the emergency response process.

V. Methodology

Our framework is designed to simulate the operations of large heterogeneous ESCS networks, and to easily adapt diverse mathematical models for the various stochastic processes involved. It can be used for studies of cybersecurity threats, catastrophic events, and regular ESCS operation. We model three crucial aspects of an ESCS: (1) system *structure*, (2) its main components' internal *behavior*, and (3) the *interactions* among these components. These considerations are examined in the following subsections in order, from general to specific.

A. Structure: ESCSes as Graph-based Systems

Initiating a 911 call sets in motion a cascade of events that involve various layers of technology and emergency personnel. ESCSes are complex heterogeneous multi-network systems that have safety-critical, operational, and regulatory concerns. For any model to be useful, however, it must be simpler than the real system; thus, one must reduce it to the core elements. In light of this, the following three fundamental ESCS entities were identified and characterized: (1) Caller Regions (CRs), (2) PSAPs, and (3) First Responders.

Here, *Caller Regions* denote the geographic areas from where calls originate, *PSAPs* are emergency call centers responsible for answering emergency calls and dispatching first responders, and *Responders* represent emergency response centers (e.g. police and fire stations). These components are arranged in a network that underlies the GIS-based call routing and dispatching dynamics of an ESCS, as described in subsection IV-B.

ESCSes are part of a family of complex cyber-physical systems that are effectively represented using graphs. Graphs are used to model network nodes (*vertices*) and their connections (*edges*) in a pairwise relationship. The particular model, in our framework, is a *directed graph*, where vertices denote the aforementioned ESCS entities and edges represent the communication channels between them.

B. Behavior: Discrete Event Queueing Model

In essence, PSAPs are call centers specialized for emergencies. Therefore, discrete event queuing models extensively used in the management of call centers are suitable for modeling the processing of emergency calls by PSAPs. Furthermore, the same concept can be applied to emergency responder nodes, thus forming a multi-queuing model.

Conceptually, a call center contains k trunk lines with up to the same number of workstations ($w \leq k$) and agents ($n \leq w \leq k$). The number of trunk lines corresponds to the number of simultaneous calls, so the number of agents plus the capacity of the center's queue. One of three scenarios occurs when a call arrives: (1) it is answered right away if there is an available agent, (2) the call is placed in the queue (on hold) if there are no available agents, or (3) the caller receives a busy signal if there are no trunks available.

In this model, we think of an agent as a resource that is occupied while a caller is receiving assistance and immediately released once it has been served. Calls are lost due to blocking when all trunks are busy or when a caller abandons the queue due to impatience, possibly redialing soon after. Consequently, arriving calls come from either those who made an initial call, those who got a busy signal, or those who abandoned the queue after waiting.

Stochastic processes such as call arrivals, service time, and customer impatience — call abandonments — must be modeled through random variables, drawn from suitable probability distributions. Existing research in this area provides candidates but their goodness of fit must be evaluated based on data obtained from the real system [19]. The mathematical model for the call arrival process is discussed, in detail, in subsection V-D; the details of other stochastic processes are presented in subsection V-E.

C. Interactions: Communicating Finite State Machines

Communicating Finite State Machines (CFSMs) were researched by [20] in the domain of distributed computing. They employed a common representation of communicating processes, where each process functions as a finite-state machine and inter-process communication occurs through reliable

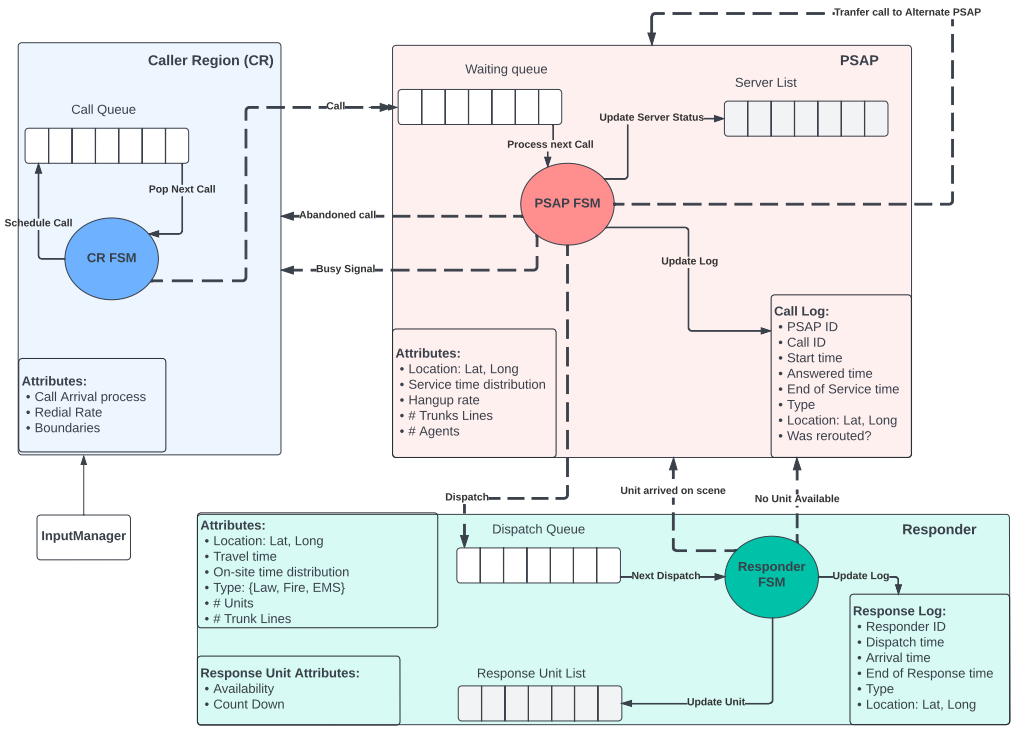


Fig. 1: 911 Communicating Finite State Machine model of an ESCS.

full-duplex first-in first-out (FIFO) channels. CFSMs were chosen for modeling the interactions between ESCS entities because they offer several advantages, such as the ability to analyze the execution of each process separately, rather than considering the system as a whole, thus partitioning complex behavior into manageable sub-problems. Notably, this model abstraction aligns with the potential for a future parallel GPU implementation, which Graphitti is designed to facilitate.

A simulation begins with an initial configuration defined by specific parameters and proceeds for a predetermined number of time steps. The values of these parameters and simulation variables at each time step collectively constitute the "system state." For an ESCS graph, the system state fully represents every vertex and edge it contains. During each time step, ESCS entities receive inputs and undergo state transitions, potentially resulting in the transmission of outputs to other vertices.

Fig. 1 illustrates how ESCS entities, modeled as vertices, receive messages through incoming edges and transmit messages via outgoing edges. Each vertex is depicted as a Finite State Machine (FSM), with state transitions determined by events arising within a discrete event queuing model, such as call initiation, call answer, unit dispatch, call termination, and end-of-emergency response. Consequently, each vertex contains a First-In-First-Out (FIFO) queue that dictates event processing order and a set of attributes, some modeled as stochastic processes with probability distributions. The PSAP and responder vertices maintain lists of resources (servers and responder units, respectively) and record metrics for serviced calls and emergency responses.

D. Call Arrival as the Realization of a Spatiotemporal Cluster Point Process

A *spatiotemporal cluster point process* is a random process that captures the clustering of events in space and time, characterized by three key elements: (1) a primary process, (2) a secondary process, and (3) a systematic aggregation of secondary points. In simpler terms, a cluster point process aggregates clusters of events associated with the points of a primary process, often called the "parent process" [21].

Cluster point processes, like Neyman-Scott [22], excel in modeling natural phenomena such as cosmology [22], rainfall analysis [23], seismic events [24], and neural spike trains [25], highlighting their accuracy in capturing complex real-world phenomena. In this work, we use a spatiotemporal cluster point process for modeling the relationship between emergency calls and the emergency incidents that generate them. The primary process denotes emergency incidents, whereas the secondary process represents resulting calls, effectively capturing the spatial and temporal clustering of calls near emergencies.

1) The primary process: Any point process that effectively captures emergency events' temporal and spatial characteristics can serve as the primary process. The distributions of points in time and space operate independently, accommodating stochastic and deterministic occurrences, and regular and irregular patterns. Here, we use a Poisson distribution to

model the primary process in time, with events assumed to be uniformly distributed in space.

2) The Secondary Process: Emergency calls are simulated as a Poisson point process inside a circle, centered at the corresponding primary event. For each cluster, we model the number of calls, the calls' arrival times, and the location of each call.

Each cluster's circle is defined by a radius $r > 0$ and an intensity $i > 0$. The intensity indicates the number of points per unit area; thus the number of secondary points (n) is

$$n = \pi \cdot r^2 \cdot i \tag{1}$$

Our model captures the varying magnitudes of emergency events through the use of *prototypes* that define radius and intensity. In practice, the prototypes' values and rate of occurrence are determined by the incidents and call patterns observed at the PSAPs being studied. The parameters of the secondary process are then drawn from a normal distribution that aligns with the selected prototype. Given a prototype radius' mean (μ_r) and standard deviation (σ_r), the cluster radius is defined as

$$R \sim \mathcal{N}(\mu_r, \sigma_r^2) \tag{2}$$

where R follows a normal distribution. Likewise, from a prototype intensity's mean (μ_i) and standard deviation (σ_i), a normally distributed intensity is defined as

$$I \sim \mathcal{N}(\mu_i, \sigma_i^2) \tag{3}$$

The calls' arrival times are computed as follows: let T be a random variable representing the calls' inter-arrival interval. Then, T follows an exponential distribution

$$T \sim \text{Exp}(\sigma_t) \tag{4}$$

with rate parameter σ_t. Subsequently, the cumulative sum of T $(t_1, t_1+t_2, ..., t_1+t_2+t_3+...+t_n)$ is added to the timestamp of the emergency incident.

For the spatial domain, the emergency calls are scattered within the circle using polar coordinates (θ, ρ), drawn from a uniform distribution. The distribution of angle values (θ) is proportional to 2π, whereas the area of a circle is proportional to the square of its radius ($Area = \pi \cdot r^2$). Therefore, if U and V are two independent uniform random variables on the interval (0, 1), the polar coordinates of points uniformly located on a circle of radius r are given by

$$(\rho, \theta) = (r \cdot \sqrt{U}, 2\pi \cdot V) \tag{5}$$

E. Modeling Other Stochastic Processes

1) Call Abandonment: A customer who calls when all servers are busy is placed in a waiting queue (see subsection V-B); those who run out of patience before their call gets answered, hang up. This process is known as *abandonment*. The idea of customers being prepared to wait for a specific duration is used in the Palm/Erlang-A model to incorporate call abandonment into the Erlang-C equation. In the Erlang-A model, each call arrival is associated with an exponentially distributed *patience time* with mean θ^{-1} and gets an *offered waiting time* in the queue. If the waiting time exceeds the patience time, the call is considered abandoned [26].

The importance of incorporating call abandonment into a model is illustrated by [27] and [26]. Reference [27] used the Erlang A queuing model, demonstrating that even a small fraction of abandoned calls during heavy traffic significantly impacts system performance. Similarly, [26] showed that in a heavily loaded call center, 3.1% abandonment reduces the *average speed of answer* from 8.8 seconds to 3.7 seconds, because abandonment effectively decreases the workload precisely when it is most needed.

We adopt an exponentially distributed patience time. However, the abandonment rate θ is not directly observable because patience time is only measurable for customers who abandon the queue. For those who receive service, the waiting time is only a lower bound of their patience time. Here, we implement the method suggested by [26], based on the relationship between the average wait in the queue ($E[W]$) and the fraction of customers that abandon it ($P\{Ab\}$), for estimating the abandonment rate θ as

$$\theta = \frac{P\{Ab\}}{E[W]} = \frac{\text{Abandonment Fraction}}{\text{Average Wait}} \tag{6}$$

2) Redial: If all call takers and trunks are busy when a customer initiates a call, they receive a busy signal and the call is dropped. NENA determines that 85% of customers whose calls drop immediately redial. In our model, the redialing decision is represented by a discrete random variable (Rd) that follows a Bernoulli distribution with probability of success of $P = 0.85$. The redialing probability can be fitted to a given region from call data that uniquely identifies the callers. However, obtaining call data with unique identifiers from the 911 system was unfeasible for our work due to technical obstacles, privacy considerations, and legal compliance issues.

3) Service Time: Service time represents the interval between the initial call answer and the dispatch of a first responder to the emergency incident. Here, service time is assumed to follow an exponential distribution and is predetermined at the outset of the simulation.

4) Responder Dispatch and Response Time: Dispatching responders starts with selecting the most suitable one for a given emergency event. This involves consideration of responder availability, expertise, and proximity to the event. In our modeling framework, responder units are dispatched from different stations, with priority given to the unit stationed closest to the emergency incident.

To enhance simulation realism, emergency calls, and therefore responses, are categorized into one of three types: (1) Law: requiring a police response, (2) EMS: requiring an Emergency Medical Service response, or (3) Fire: requiring a fire unit response.

The time a response unit remains occupied includes both the driving time to the incident location and the on-site time dedicated to addressing the emergency. Driving time is determined by calculating the time it takes for a responder to

cover the Euclidean distance between the dispatching center and the incident location. The duration of on-site time is assumed to be exponentially distributed, with an average on-scene time of 20 minutes considered reasonable based on prior research studies [28], [29].

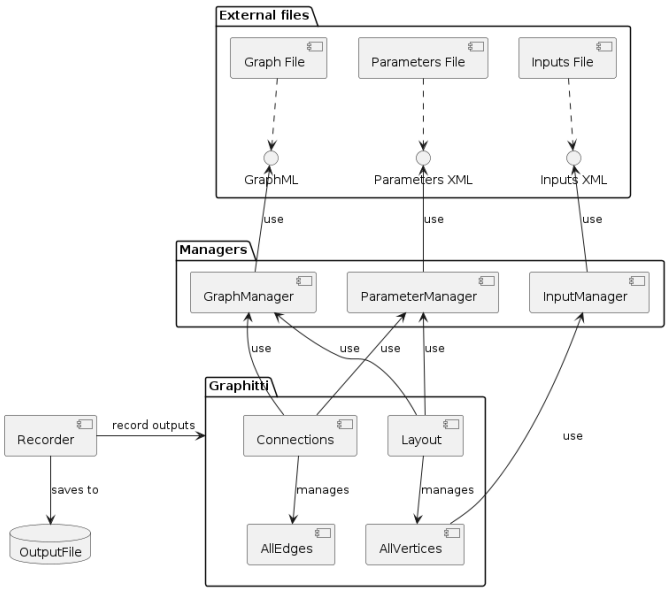


Fig. 2: Graphitti's Component Diagram.

F. Implementation Within the Graphitti Simulator

These simulations were implemented using Graphitti, our in-house graph-based simulator [10]. Graphitti is designed to simulate discrete and continuous graph-based systems, and facilitate the migration and validation of large-scale (tens of thousands of vertices, millions of edges) and long-duration (billions of time steps) simulations to GPUs.

Fig. 2 provides a component diagram for our simulation within Graphitti's architecture, where the ESCS graph, simulation parameters, and input modeling are fed to the simulator through external files, with graphs and call arrivals created by external Python scripts.

The representation of a graph within Graphitti is contained in four components: Connections, Layout, AllEdges, and AllVertices. The *Layout* component manages the location and parameters of the vertices contained in *AllVertices*; *Connections* manages the properties of the edges contained in *AlllEdges*. *Connections* and *Layout*, in addition to using the parameters from the *ParameterManager* component, rely on the graph contained within *GraphManager* to generate Graphitti's internal representation of the network. The *InputManager* loads the list of events into internal queues, one per vertex, to be retrieved by the respective vertex during the simulation. This design enables handling a wide range of graph structures and input sources, real-world or synthetically generated. As a result, one can conduct experiments by adjusting network structure or input arrival rate without making changes to the simulator's code.

G. Vertex Communication and State Transition

The CFSM abstraction allows us to consider the execution sequence of each vertex independently, with interactions occurring through message exchanges across connecting edges. This leads to a system with two distinct phases: (1) a *Communication Phase*, where vertices receive messages, and (2) a *State Transition Phase*, where vertices undergo state transitions due to internal or external events.

During the *Communication Phase*, vertices pull messages from their incoming edges such as emergency calls, dispatch instructions, and availability status as illustrated in Fig. 1. Although our current implementation is a sequential loop over all incoming edges, this framework was purposely designed for future GPU parallelization. This "pulling strategy" prevents multiple processes from simultaneously accessing and modifying data structures, eliminating the need for locking and enabling the use of a parallel reduction algorithm for efficiently managing numerous incoming edges. Each vertex stores the received messages in a waiting queue if available space exists, otherwise, the message is dropped and subsequently handled by the source vertex.

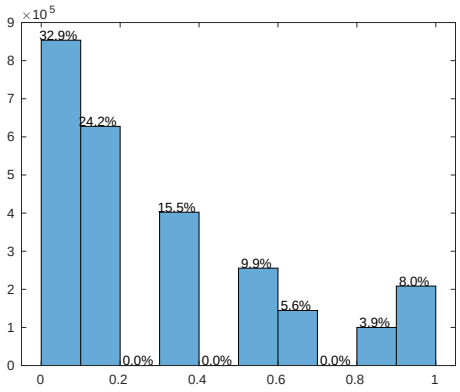
During the *State Transition Phase*, vertices respond to internal and external events, potentially undergoing state transitions. As presented in subsection V-B, the state transitions in each ESCS entity follow a discrete-event queuing model, further discussed in this section.

1) Caller Region State Transition: A Caller Region defines the geographical area where calls originate, modeled as a cluster point process. Graphitti simulations are organized into epochs, each comprising multiple time steps. As illustrated in Fig. 1, each caller region contains a call queue that, in between epochs, is loaded with the calls scheduled for the next epoch. During each time step, the region performs two actions: (1) checks for dropped calls and initiates redialing if necessary, and (2) routes the next call to the appropriate PSAP via an outgoing edge. The current implementation ensures that only one call is processed per Caller Region per time step, achieved by subdividing a PSAP's service zone or adjusting the time step duration.

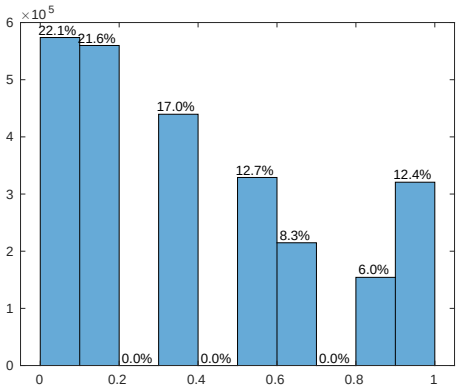
2) PSAP State Transition: The primary role of a PSAP is to handle emergency calls from its Caller Regions and coordinate the dispatch of appropriate emergency responders. Each PSAP contains a waiting queue and a list of servers as seen in Fig. 1. The waiting queue holds the calls not yet answered by a "server", responsible for call handling. At each time step, a PSAP undertakes two key actions: (1) managing servers that have concluded their service for a call, and (2) assigning new calls to available servers.

After completing a call, a server becomes available for new ones. However, before taking on new calls, servers fulfill dispatch responsibilities such as sending dispatch messages to the nearest suitable responder via an outgoing edge and record essential call metrics, such as start time, response time, conclusion time, and instances of call abandonment.

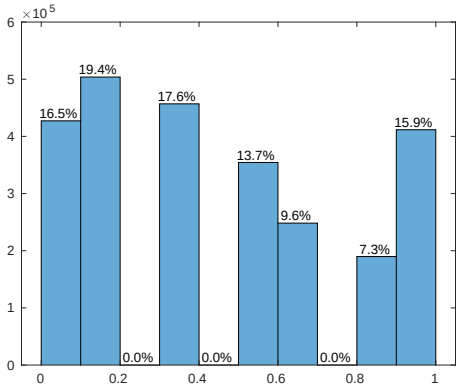
Available servers are processed iteratively for call assignments until no more servers are available or calls remain in



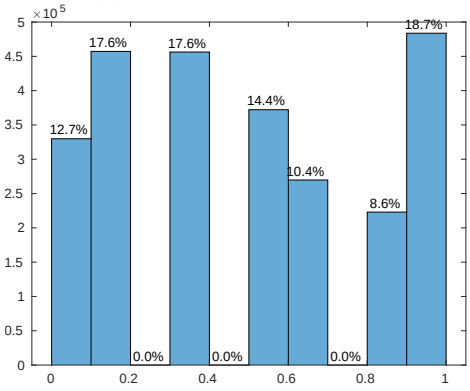
(a) Arrival rate = 45.6 calls/hr



(b) Arrival rate = 63 calls/hr



(c) Arrival rate = 76 calls/hr



(d) Arrival rate = 84.2 calls/hr

Fig. 3: Call arrival rate impact on system utilization

the waiting queue. Upon receiving a call assignment, a server is occupied for the duration of the call. Additionally, call abandonments are managed within this process as described in subsubsection V-E1. If a caller's wait time exceeds their patience threshold — the duration they are willing to wait in the queue — the call is classified as abandoned.

3) Emergency Responder State Transition: A "Responder" vertex represents an emergency response center. As illustrated in Fig. 1, responder vertices operate in a discrete event queuing model similar to PSAPs. Response units address emergency events from a dispatch queue in two phases. First, units completing an emergency response document their metrics and return as available units. Then, new emergency incidents are allocated to available units until all units are busy or no incidents remain in the queue. Once an incident has been assigned, a unit remains occupied for the duration of travel to the incident location plus the time spent on-scene.

VI. Illustrative Application

To demonstrate the applicability of our graph-based simulation framework, we developed a graph representation of the ESCS network of King County, Washington, extracted from a Geographic Information System (GIS) dataset (provided by the Washington State 911 Coordination Office) comprised of five layers that map the boundaries of PSAPs, police, fire, and EMS services. We focused on the Seattle Police Department (PD) composed of 1 PSAP, 34 Fire Stations (also providing EMS services), and 5 Police stations. The Seattle PD PSAP, now known as Community Assisted Response and Engagement (CARE), is the largest of the 12 PSAPs in King County, serving about 10% of all 911 calls.

A month-long dataset composed of 41,217 calls with a call rate of 57.25 calls/hr was used to parameterize the discrete event queuing model for CARE. The dataset revealed a maximum hourly call rate of 137, an abandonment rate of 9.42%, and call durations ranging from 4 to 2016 seconds (204 seconds average). Using Erlang equations, estimations suggest a need for 6 call takers, considering a 10-second caller waiting tolerance (Erlang C), and 16 trunk lines to manage a 1% probability of blocking during peak times with an additional 10 seconds of service time for post-processing.

We conducted simulations using one month of synthetic call data generated through the cluster point process algorithm discussed in subsection V-D. These simulations used call arrival rates ranging from 45.6 to 84.2 calls/hr in 10% increments. Other parameters were kept consistent throughout the simulations. The redial probability used was 85% with an average patience time of 49.36 seconds (see subsubsection V-E1). Additionally, the average on-scene duration was set at 20 minutes (see subsubsection V-E4), whereas the minimum and average service times were set to 4 and 204 seconds, respectively. Simulations were run on a computer with a 14-

core Intel i7-12700H 2.3 GHz processor, requiring 47 seconds for a one-month simulation of this single-PSAP network.

The selected metrics for analyzing simulation results were customer waiting time and system utilization. System utilization is defined as the fraction of call-takers or responder units busy at any given time. Utilization histograms in Fig. 3 are presented for the different arrival rates. At 45.6 calls/hr, the system surpasses 80% utilization merely 11.9% of the time, coinciding with an average waiting time of 6.8 seconds. As the arrival rate grows to 84.2 calls/hr, utilization above 80% increases to 27%; the average waiting time increases moderately to 9.4 seconds. However, at this threshold the abandonment rate triples from 9 to 27.3%.

VII. Conclusion

In this work, we presented a detailed architectural framework and mathematical models for simulating Emergency Services Communication Systems, designed to easily accommodate diverse ESCS networks and their stochastic processes. In addition, we introduced a novel approach for modeling call arrival burstiness as the realization of a cluster point process, capitalizing on the relationship between emergency calls and underlying emergency events. We then used our framework to implement one-month-long simulations of the Seattle Police Department ESCS with varying call arrival rates.

We are currently implementing a GPU version of our models within Graphitti to improve performance for long durations and large ESCS networks. We also plan to evaluate the use of our simulator to support cybersecurity tabletop exercises.

Acknowledgment

We extend our appreciation to additional contributors. Scott Sotobeer offered invaluable insights into the workings of ESCSes and their associated policies. Bennett Ye and Jacob White developed tools for ESCS graph visualization and editing, and Divya Kamath and Xiang Li made substantial architectural improvements to Graphitti.

References

[1] Y. Mirsky and M. Guri, "Ddos attacks on 9-1-1 emergency services," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 6, pp. 2767–2786, 2021.

[2] M. Xue and S. Roy, "Cyber-physical queueing-network model for risk management in next-generation emergency response systems," 2021.

[3] B. Dearstyne, "The FDNY on 9/11: Information and decision making in crisis," *Government Information Quarterly*, vol. 24, pp. 29–46, Jan. 2007.

[4] R. Simon and S. Teperman, "The world trade center attack: Lessons for disaster management," *Critical Care*, vol. 5, no. 6, pp. 318–320, 2001.

[5] National Emergency Number Association (NENA) 911 Core Services Committee and i3 Architecture Working Group, "NENA i3 standard for next generation 9-1-1," tech. rep., National Emergency Number Association, 2021.

[6] D. Corral-De-Witt, E. V. Carrera, J. A. Matamoros-Vargas, S. Munoz-Romero, J. L. Rojo-Alvarez, and K. Tepe, "From e-911 to NG-911: Overview and challenges in ecuador," *IEEE Access*, vol. 6, pp. 42578–42591, 2018. Publisher: Institute of Electrical and Electronics Engineers (IEEE).

[7] F. Liberal, J. O. Fajardo, C. Lumbreras, and W. Kampichler, "European NG112 crossroads: Toward a new emergency communications framework," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 132–138, 2017. Conference Name: IEEE Communications Magazine.

[8] E. K. Markakis, A. Lykourgiotis, I. Politis, A. Dagiuklas, Y. Rebahi, and E. Pallis, "EMYNOS: Next generation emergency communication," *IEEE Communications Magazine*, vol. 55, no. 1, pp. 139–145, 2017. Publisher: Institute of Electrical and Electronics Engineers (IEEE).

[9] R. Russell and D. Mazeau, "PSAP staffing guidelines report," Tech. Rep. NENA-REF-001-2003, National Emergency Number Association, 2003.

[10] J. Martinez Jordan, C. O'Keefe, M. Stiber, J. Brown, V. Gandhi, M. Sorvik, T. Salvatore, D. Kamath, P. Pal, K. Dukart, X. Li, C. McIntosh, and A. Rudrawar, "UWB-Biocomputing/Graphitti: It's 2017 in 2023," June 2023.

[11] S. R. Neusteter, M. Mapolski, M. Khogali, and M. O'toole, "The 911 call processing system: A review of the literature as it relates to policing," *Vera Institute Of Justice*, pp. 240–254, 2019.

[12] K. Gustavsson, *Stochastic Modeling and Management of an Emergency Call Center*. PhD thesis, Mid Sweden University, 2018.

[13] K. Gustavsson, P. L'Ecuyer, and L. Olsson, "Modeling bursts in the arrival process to an emergency call center," 2018.

[14] R. Barnes and B. Rosen, "911 for the 21st century," *IEEE Spectrum*, vol. 51, no. 4, pp. 58–64, 2014. Conference Name: IEEE Spectrum.

[15] Federal Communication Commission, "Legal and Regulatory Framework for Next Generation 911 Services," Report to Congress and Recommendations 112-96, Federal Communications Commission, Feb. 2013.

[16] National Emergency Number Association (NENA) and others, "NENA standard for 9-1-1 call processing," 2020.

[17] J. M. Stevens, T. C. Webster, and B. Stipak, "Response Time: Role in Assessing Police Performance," *Public Productivity Review*, vol. 4, no. 3, pp. 210–230, 1980. Publisher: Taylor & Francis, Ltd.

[18] T. Stratmann and D. C. Thomas, "Dial 911 for murder: The impact of emergency response time on homicides," *SSRN Electronic Journal*, 2016. Publisher: Elsevier BV.

[19] A. M. Law and W. D. Kelton, *Simulation modeling and analysis*. McGraw-Hill series in industrial engineering and management science, McGraw-Hill, 3rd ed ed., 2000.

[20] D. Brand and P. Zafiropoulo, "On communicating finite-state machines," *Association for Computing Machinery (ACM)*, vol. 30, no. 2, pp. 323–342, 1983. Place: New York, NY Publisher: Association for Computing Machinery.

[21] R. F. Serfozo, "Chapter 1 point processes," in *Stochastic Models*, vol. 2 of *Handbooks in Operations Research and Management Science*, pp. 1–93, Elsevier, 1990.

[22] J. Neyman and E. L. Scott, "Statistical approach to problems of cosmology," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 20, no. 1, pp. 1–43, 1958.

[23] P. S. P. Cowpertwait, C. G. Kilsby, and P. E. O'Connell, "A space-time neyman-scott model of rainfall: Empirical analysis of extremes," *Water Resources Research*, vol. 38, pp. 6–1–6–14, aug 2002.

[24] S. Anwar, M. Yaseen, and S. A. Mahmood, "Higher order gibbs point process modeling of 2005-kashmir earthquakes," *Modeling Earth Systems and Environment*, vol. 9, pp. 1335–1347, oct 2022.

[25] L. Gómez, R. Budelli, R. Saa, M. Stiber, and J. P. Segundo, "Pooled spike trains of correlated presynaptic inputs as realizations of cluster point processes," *bc*, vol. 92, pp. 110–127, Feb. 2005.

[26] A. Mandelbaum and S. Zeltyn, *Service Engineering in Action: The Palm/Erlang-A Queue, with Applications to Call Centers*, pp. 17–45. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007.

[27] N. Gans, G. Koole, and A. Mandelbaum, "Telephone call centers: Tutorial, review, and research prospects," *Manufacturing & Service Operations Management*, vol. 5, no. 2, pp. 79–141, 2003.

[28] G. David and T. Brachet, "Retention, learning by doing, and performance in emergency medical services," *Health Services Research*, vol. 44, pp. 902–925, June 2009.

[29] C. Vincent-Lambert and T. Mottershaw, "Views of emergency care providers about factors that extend on-scene time intervals," *African Journal of Emergency Medicine*, vol. 8, p. 1, Mar. 2018. Publisher: African Federation for Emergency Medicine.