# Phase 2 System Test Plan (STP)

## Georgia Department of Transportation: Safe Trips in a Connected Transportation Network ITS4US Deployment Project

www.its.dot.gov/index.htm

**Revised Draft Report — September 4, 2023**
**FHWA-JPO-23-996**

ITS4US
IT'S TRANSPORTATION FOR ALL OF US

U.S. Department of Transportation

Produced by the Georgia Department of Transportation (GDOT)
U.S. Department of Transportation
Intelligent Transportation Systems Joint Program Office
Federal Highway Administration
Office of the Assistant Secretary for Research and Technology
Federal Transit Administration

# Notice

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

The U.S. Government is not endorsing any manufacturers, products, or services cited herein and any trade name that may appear in the work has been included only because it is essential to the contents of the work.

## Technical Report Documentation Page

| 1. Report No. | 2. Government Accession No. | 3. Recipient's Catalog No. |
|---|---|---|
| **FHWA-JPO-23-996** | | |

| 4. Title and Subtitle | 5. Report Date |
|---|---|
| Phase 2 System Test Plan (STP)<br>Safe Trips in a Connected Transportation Network ITS4US Deployment Project | September 4, 2023 |
| | **6. Performing Organization Code** |

| 7. Author(s) | 8. Performing Organization Report No. |
|---|---|
| Alan Davis (GDOT), Kofi Wakhisi (ARC), Bennett Foster (ARC), Polly Okunieff (GOSS), Natalie Smusz-Mengelkoch (ICF), Joseph Yawn (ICF) | |

| 9. Performing Organization Name and Address | 10. Work Unit No. (TRAIS) |
|---|---|
| Georgia Department of Transportation – One Georgia Center<br>600 West Peachtree NW,<br>Atlanta, GA 30308 | |
| | **11. Contract or Grant No.**<br>693JJ32250011 |

| 12. Sponsoring Agency Name and Address | 13. Type of Report and Period Covered |
|---|---|
| U.S. Department of Transportation<br>ITS Joint Program Office<br>1200 New Jersey Avenue, SE<br>Washington, DC 20590 | Revised Draft Report |
| | **14. Sponsoring Agency Code**<br>HOIT-1 |

**15. Supplementary Notes**

The following USDOT partners are supporting this document development: Elina Zlotchenko (Program Manager), Sarah Tarpgaard (Agreement Officer), and Norah Ocel (Agreement Officer Representative)

**16. Abstract**

The Georgia Department of Transportation ITS4US Deployment project, Safe Trips in a Connected Transportation Network (ST-CTN), is leveraging innovative solutions, existing deployments, and collaboration to make a positive impact using transportation technology to support safety, mobility, sustainability, and accessibility. The ST-CTN concept is comprised of an integrated set of advanced transportation technology solutions (connected vehicle, transit signal priority, machine learning, predictive analytics) to support safe and complete trips, with a focus on accessibility for those with disabilities, older adults, and those with limited English proficiency.

The System Test Plan (STP) defines how the ST-CTN project will conduct the system and subsystem testing including test strategy, flow of testing and full system acceptance. This will be in preparation of operational deployment. The testing is divided into functional, performance, security, integration/interface, user, and automated. The STP also discusses the Agile testing approach including sprint demonstrations and any necessary regression testing. The documentation process for defects and deficiencies is described for all types of testing.

| 17. Keywords | 18. Distribution Statement |
|---|---|
| ITS4US; Complete Trip; Deployment; ITS; Intelligent Transportation Systems; ITS4US; Complete Trip; Accessibility; System Testing; System Test Plan | |

| 19. Security Classif. (of this report) | 20. Security Classif. (of this page) | 21. No. of Pages | 22. Price |
|---|---|---|---|
| Unclassified | Unclassified | **(74)** | N/A |

# Revision History

| Name | Date | Version | Summary of Changes | Approver |
|---|---|---|---|---|
| EMT / Tech Team | 26 July 2023 | 0.1 | Initial Draft | Matthew Smith |
| EMT / Tech Team | 31 July 2023 | 0.2 | Draft | Bennett Foster |
| EMT / Tech Team | 4 September 2023 | 0.3 | Revised Draft | Bennett Foster |

PAGE INTENTIONALLY LEFT BLANK

# Table of Contents


**1 Introduction ..... 1**

1.1 Document Scope ..... 1

1.2 Intended Audience ..... 1

1.3 Project Background ..... 2

1.3.1 USDOT'S ITS4US Deployment Program ..... 2

1.3.2 The ST-CTN Project ..... 3

1.4 Project Scope ..... 5

1.5 Project References Related to the STP ..... 8

1.6 Document Overview ..... 9

**2 System Test Plan Overview ..... 11**

2.1 Overview of High-Level Testing ..... 11

2.1.1 ST-CTN Testing Approach and the "Vee" Systems Engineering Diagram ..... 13

2.2 Testing Organization ..... 14

2.3 Types of Testing ..... 17

2.4 Verification Methods ..... 18

2.5 Test Documentation ..... 18

2.6 High-Level Testing Schedule ..... 19

**3 Agile Test Processes ..... 23**

3.1 Agile Development Process ..... 23

3.1.1 Backlog Tickets ..... 29

3.2 Agile Test Processes Overview ..... 29

3.2.1 Agile Test Environments ..... 31

3.2.2 Agile Test Case Storage ..... 31

3.3 G-MAP Testing Process ..... 33

3.3.1 OpenTripPlanner Routing Engine ..... 33

3.3.2 OTP Middleware ..... 34

3.3.3 Web Application User Interface (Front-End) ..... 34

3.3.4 Web Application Manual Quality Assurance and Testing Procedures ..... 36

3.3.5 Mobile Apps ..... 37

3.4 STM Testing Process ..... 37

3.4.1 Unit Testing ..... 37

3.4.2 Integration Testing ..... 37

**4 Bench and Configuration Testing ..... 39**
4.1 Indoor Navigation System ..... 39
4.1.1 Mapping Tools ..... 40
4.1.2 Inventory and Monitoring Tools ..... 40
4.1.3 Beacons ..... 40
4.1.4 SDK ..... 40
4.1.5 INS Pilot Test ..... 40
4.2 PED-X ..... 41
4.3 Ride Request ..... 41
4.4 Public PMD ..... 41
**5 Controlled Field and User Acceptance Testing ..... 43**
5.1 Alpha Testing Approach ..... 43
5.2 Beta Testing Approach ..... 44
5.3 Dependencies ..... 44
**6 Defect/Deficiency Tracking and Resolution Processes ..... 51**
6.1 Defect / Deficiency Categorization ..... 52
6.2 Defect / Deficiency Impacts on Testing ..... 53
6.3 Defect Issue Resolution ..... 53
6.4 Defect/Deficiency Tracking and Reporting ..... 53
**7 Test Cases and Requirements Traceability ..... 55**
**8 References ..... 57**

## List of Tables

Table 1. STP Related Project Documents and Relationship to Testing ..... 9
Table 2. Technical Team Roles and Responsibilities ..... 16
Table 3. High-Level Testing Schedule ..... 20
Table 4. Definitions of Agile Methodology Terms ..... 23
Table 5. Priority Ranking for Tickets ..... 29
Table 6. System Integration Testing and Dependencies ..... 46
Table 7. Requirements to Test Case Traceability Matrix ..... 56
Table 8. References ..... 57

## List of Figures

Figure 1. Phases of the ITS4US Deployment Program ..... 3
Figure 2. ST-CTN High-Level Context Diagram ..... 3
Figure 3. Traveler's Complete Trip ..... 6
Figure 4. High-Level Testing Approach ..... 12

Figure 5. Relationship Between ST-CTN Testing Approach and System Engineering “Vee” Diagram Testing Phases ....13
Figure 6. Technical Team Organization ....15
Figure 7. Hybrid Agile / Vee Development Process....28
Figure 8. Agile Testing Process ....32
Figure 9. Example of Automated Test Script User Interface....35
Figure 10. Example of Percy Test Results UI....36
Figure 11. Defect / Resolution Process ....51
Figure 12. Ticket States....54

# 1 Introduction

The Georgia Department of Transportation (GDOT) and Atlanta Regional Commission's (ARC) Safe Trips in a Connected Transportation Network (ST-CTN) project was selected by the U.S. Department of Transportation (USDOT) as a part of the ITS4US Deployment Program. The project seeks to enhance the traveler's complete trip travel experience by enhancing mobility, accessibility, reliability, and safety for system users, particularly for underserved communities, including people with disabilities, older adults, and users with limited English proficiency (LEP). This is done by leveraging innovative solutions, existing deployments and team collaboration such as integrating connected vehicle (CV) data with an open-source software (OSS) trip planner that is used to provision web-based and mobile application user access. The trip planner will provide users with the ability to create a personalized trip plan with information regarding the navigation of physical infrastructure, the ability to resolve unexpected obstacles, and ensure users' visibility throughout the trip. The proposed deployment will provide targeted users with the ability to dynamically plan and navigate trips.

## 1.1 Document Scope

The System Test Plan (STP) documents the test strategy used for the system to ultimately reach operational deployment. The document reviews the types of testing, the flow of testing for the individual components, integrating the subsystems and external systems, verification of requirements, and validation of needs to accomplish full system acceptance. The STP lays out the proposed testing schedule, test resources (platforms and tools), and defect processing.

Note that quality control and validation procedures for datasets will be described in the System Design Document (SDD) or Comprehensive Maintenance and Operations Plan (CMOP).

## 1.2 Intended Audience

The intended audience for this STP document is the USDOT and the ST-CTN project team, which includes GDOT, ARC, its consultants and partners, including:

- Gwinnett County
- Statewide Independent Living Council of Georgia (SILCGA)
- Georgia Institute of Technology
- ICF
- Kimley-Horn
- GO Systems and Solutions (GOSystems)
- Arcadis / IBI Group

The STP is intended to inform the USDOT on how the testing process will be documented and managed such that there is an understanding of how the project team will determine operational readiness. It is also intended to be a reference for other agencies and industry professionals that

are interested in deploying a project of similar nature to understand how testing was done for the ST-CTN project.

# 1.3 Project Background

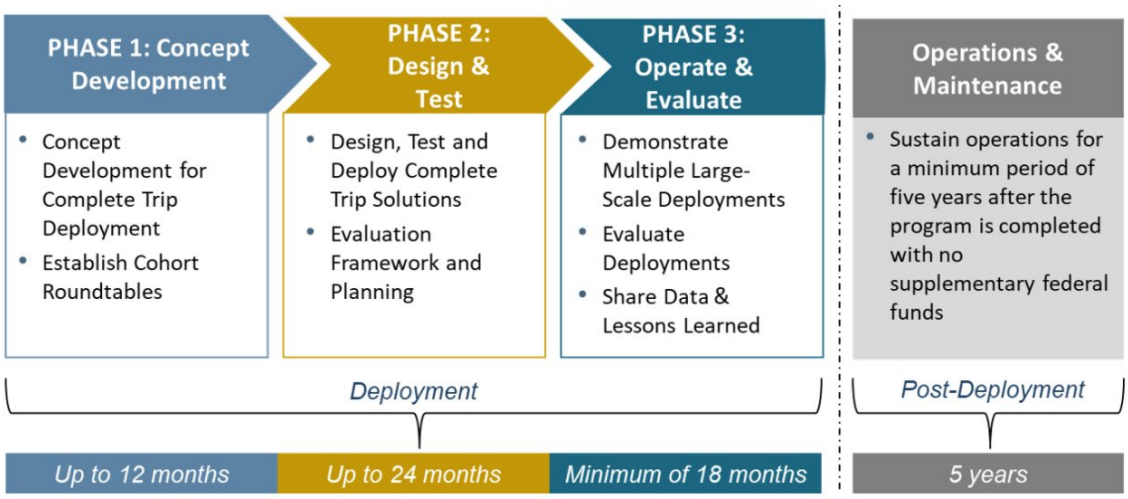
## 1.3.1 USDOT'S ITS4US Deployment Program

The USDOT's ITS4US Deployment Program seeks to identify ways to provide more efficient, affordable, and accessible transportation options for underserved communities that often face greater challenges in accessing essential services. The program aims to solve mobility challenges for all travelers with a specific focus on underserved communities, including people with disabilities, older adults, low-income individuals, rural residents, veterans, and limited English proficiency travelers. The program will enable communities to build local partnerships, develop and deploy integrated and replicable mobility solutions to achieve complete trips for all travelers.

The vision for the program is foster innovative and integrated complete trip deployment to support seamless travel for all users across all modes, regardless of location, income, or disability. Goals of the USDOT's ITS4US Deployment Program are to:

- Spur high-impact integrated Complete Trip deployments nationwide
- Identify needs and challenges by population
- Develop and deploy mobility solutions that meet user needs
- Measure impact of integrated deployments
- Identify replicable solutions and disseminate lessons learned

The ITS4US Deployment Program will be executed in three phases. As depicted in **Figure 1**, deployment sites are expected to go through a Concept Development Phase (Phase 1), followed by a Design and Test Phase (Phase 2), and then an Operate and Evaluate Phase (Phase 3). Post deployment, sites are expected to sustain operations for a minimum period of five years without supplementary federal funds. Phase 1 has been completed and Phase 2 is underway.

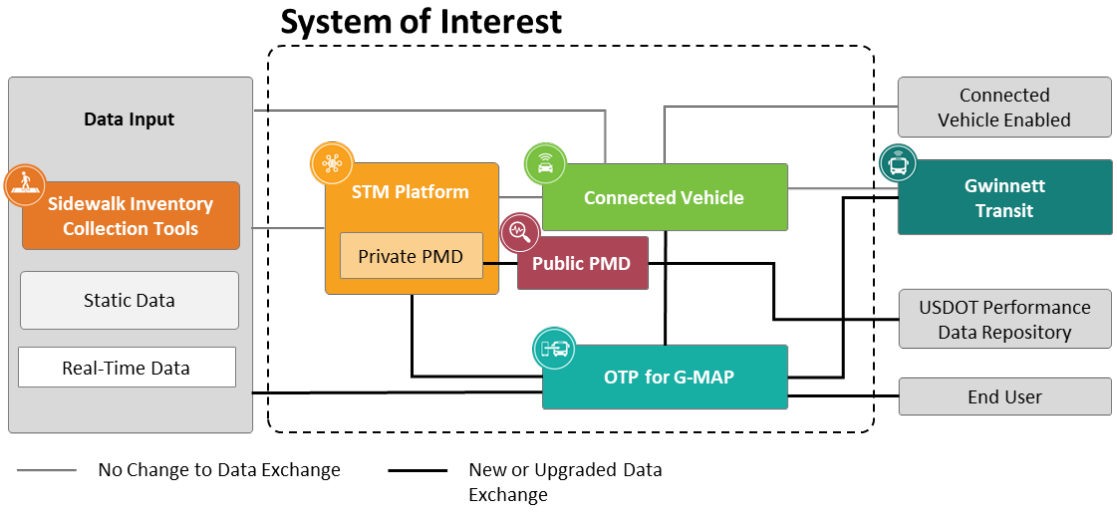



*Source: USDOT, 2021*

**Figure 1. Phases of the ITS4US Deployment Program**

## 1.3.2 The ST-CTN Project

The proposed ST-CTN deployment project will be led by GDOT with support from ARC in Gwinnett County, GA, which is a suburban county located directly Northeast of Atlanta. The project team intends to address multiple aspects of the complete trip through the development of the ST-CTN system, which will merge multiple technological innovations. The ST-CTN System of Interest (SOI) is composed of four major subsystems and several existing / external systems. The context diagram presented in **Figure 2** illustrates the concept developed in Phase 1 and shows the interactions among the end user and subsystems that will be tested.




*Source: ARC, 2022*

**Figure 2. ST-CTN High-Level Context Diagram**

The subsystem components are described as follows:



**Georgia Mobility and Accessible Planner (G-MAP) Subsystem.** Based on Atlanta Rider Information and Data Evaluation System (ATL RIDES) which is being developed with the support of Federal Transit Administration's (FTA) Integrated Mobility Innovation (IMI) Grant program. The existing program provides more reliable and easier to access journey planning to all travelers making it easier to utilize the region's transit services. The ST-CTN will extend the ATL-RIDES functionality and software to provide additional user interaction to plan and safely complete single mode or multimodal trips that are based on their abilities; improve the transition between modes by providing additional details to users and transit service operators; and suggest dynamic routing changes based on infrastructure condition and calculated delay. The extended capabilities will be branded as G-MAP. It should be noted that during Phase 1, 'ATL RIDES' was used to describe the trip planning and user interface subsystem. However, the subsystem naming convention has been refined to be described as OTP for G-MAP (hereafter referred to as G-MAP) to clarify the distinct systems. OTP refers to the open-source software OpenTripPlanner which forms the core functionality of the G-MAP.

The G-MAP will include several additional functions to aid travelers in their journey. These include:

- **Ride Request –** The Ride Request application allows a transit passenger to send a stop request to an approaching transit vehicle. This application allows a transit vehicle to know that a passenger has requested a transit stop from a mobile device. The application will use the GCT Transit Ride Request application integrated in their Computer Aided Dispatch / Automatic Vehicle Location (CAD/AVL) system as the core functionality that G-MAP uses to alert a transit operator to a traveler's presence and stop request at a transit stop.

- **Mobile Accessible Pedestrian Signal System** – The Mobile Accessible Pedestrian Signal System is an application that allows for an automated pedestrian indication request via a center to center (C2C) information flow from G-MAP to the central traffic signal system (part of the CV Subsystem). The pedestrian signal request (PED-X) will request the pedestrian indication (actuation) at a specific intersection crossing in the direction of pedestrian travel. The application will also allow the pedestrian crossing indication's time to be extended if required based on the traveler's needs and preferences.

- **Indoor Navigation** – The indoor navigation function will be enabled by the installation of waypoints and aiding transceivers to support the wayfinding and routing functions in the G-MAP mobile app. The indoor navigation beacons will be installed in the MARTA Doraville Transit Station and Gwinnett Justice Administration Center.



**Connected Vehicle (CV) Subsystem.** The CV Subsystem is composed of the systems included as a part of the Gwinnett County and GDOT CV programs. Gwinnett County and GDOT are leaders in the deployment of CV infrastructure. The subsystem will leverage the current CV components deployed in the Gwinnett County Connected Corridor and the GDOT Regional Connected Vehicle Infrastructure Deployment Program (CV1K) projects including all the center, roadside, and vehicle components. As described in the current system, the center components consist of components located at the GDOT Traffic Management Center (TMC) and Gwinnett County Transportation Control Center (TCC) – Navigator TMC, central

system signal operations, Security Credential Management System (SCMS), MAP Server and ITS Hub connected through GDOT communications network. The roadside components consist of traffic signal systems owned and operated by GDOT and Gwinnett County. The equipment and operations of the subsystems will not be changed in the ST-CTN project. All CV applications, equipment and messages have been tested under the CV1K and Connected Corridor projects. The ST-CTN project will verify during end-to-end testing (full system integration) that the triggers that generate the messages (e.g., Pedestrian Safety Message and transit signal priority) work as expected.



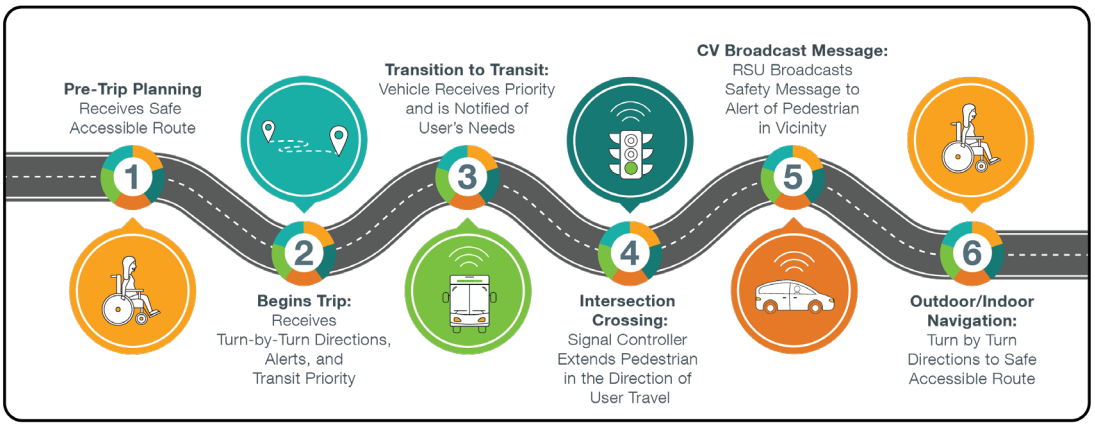
**Space Time Memory (STM) Subsystem**. The STM Platform predicts trajectories through the multimodal network based on network characteristics as congestion evolves in time and space and to provide users with more realistic estimates of commute duration, impedance value, and energy use across alternative departure times, routes, and transportation modes. The impedance-based network calculations are prepared in the STM subsystem and include all-path networks (APN). The APN will include sidewalks and all local roads and will correspond to an updated OpenStreetMap (OSM) network employed after modifications are made to add missing pathways and remove non-pathway links to improve speed.



**Performance Measurement Dashboard (PMD) Subsystem.** The PMD Subsystem is a data storage and distribution tool that archives operational, survey and performance metrics of the system. The PMD is envisioned to ingest, quality check, and curate all types of data – static and dynamic, structure and unstructured, open and private datasets, store the datasets so that they can be viewed and accessed based on user roles, and provide access to and present open-based data on an interactive dashboard to the public. The public PMD will be implemented using ARC's PowerBI license.

## 1.4 Project Scope

The ST-CTN project aims to upgrade and integrate existing technologies and services to assist underserved populations with completing their complete trip successfully, safely, and reliably. The vision of the project is to provide users complete trip functionality with directions, conditions, and status on the links between trip legs that are personalized based on the user's profile, while connecting the user to CV infrastructure to provide safer trips and more transportation network awareness. As an illustration of how the ST-CTN system will be used, transit-based trips were delineated into six segments (as depicted in **Figure 3**) to allow for easier understanding and a greater breakdown of priorities and goals.




*Source: ARC, 2020*

**Figure 3. Traveler's Complete Trip**

The delineated trip segments include the following steps and project components:

- **Step 1 Pre-Trip Planning.** The traveler plans for and receives a safe accessible route.
  - The ability to customize trip preferences based on the user's abilities.
- **Step 2 Begins Trip.** The traveler begins their trip and receives turn by turn directions, alerts, remote pedestrian activation, and can trigger transit signal priority (TSP) if the user requires additional time boarding or alighting a transit vehicle, is unable to stand for long periods, or is sensitive to weather conditions.
  - Turn by turn, shortest path, directions along pathways that meet user defined preferences.
  - Provides support services for users if they become disoriented or have issues accessing defined paths.
  - Activates TSP for buses if the user requires additional time boarding or alighting a transit vehicle, is unable to stand for long periods, or is sensitive to weather conditions.
- **Step 3 Transition to Transit.** The traveler transitions to transit and the transit vehicle receives priority and is notified of user(s) needs. TSP can be triggered if the bus is running behind schedule due to a longer boarding time needed by a user.
  - Provides userwith transit trips that have accommodations that meet user defined preferences.
  - Sends alerts to transit vehicles when a user needs additional time to board, navigate internally, or alight the transit vehicle.
  - Remotely requests service from transit vehicles while waiting to board or alight.
  - Triggers TSP if the bus is running behind schedule due to a user needing additional time to board or alight.

- **Step 4 Intersection Crossing.** When crossing a signalized intersection, the traveler indirectly interacts with the signal controller in that an authorized system call is made to the controller which extends the pedestrian phase in the direction of user travel.
    - Allows the user to communicate with connected intersections if they are unable to reach or press the crosswalk button.
    - Provides the user with information about the intersection crossing and adds time to the crossing if needed.
- **Step 5 CV Broadcast Message.** Roadside units (RSUs) broadcast safety message to alert CVs of pedestrians in the intersection.
    - Provides the ability for user to remotely request service from transit vehicles while waiting to board or alight.
    - Provides communications to CVs from pedestrian crossing signal system (via RSUs) to make them aware of pedestrians crossing a roadway.
    - Provides communications between transit vehicles and travelers waiting at a transit stop to make them aware of each other.
- **Step 6 Outdoor/Indoor Navigation.** The traveler is provided with turn-by-turn directions to a safe accessible route.
    - Hands-free navigation via mobile apps and/or wearables and accessible channels (haptic, voice, text).
    - Alerts and dynamic rerouting in response to changes in path conditions.
    - Provides the user with accessible routes into and through transit hubs within the project area.
    - Provides users with updates on the operating status of indoor infrastructure such as elevators and escalators.

System development and system integrations completed within the scope of this project will enable travelers – specifically those in the underserved community – to program and safely complete single mode or multimodal trips that are based on their abilities; improve the transition between modes by providing additional details to users and transit service operators; and suggest dynamic routing changes based on infrastructure condition and calculated delay.

The existing initiatives that are being leveraged to support the proposed ST-CTN system, and components that are being developed specifically to support ST-CTN project evaluation, are defined in more detail below. The icons and colors depicted below are used throughout the System Architecture Document (SAD) to clearly identify the critical components of ST-CTN. In some cases, partner agencies are upgrading the services within their current systems to create a more robust data set or toolset for the ST-CTN program.



**OTP for G-MAP.** ATL RIDES includes an OSS multi-modal trip planning and mobile application, integrated mobile fare payment options, and a Connected Data Platform (CDP) using regional General Transit Feed Specification (GTFS) transit service data. The tool supports multi-agency context, multilingual support, and mobile app turn-by-turn directions. The OTP architecture facilitates integration with additional OSS tools including a data analytics engine, call center module with application programming interfaces (APIs), and an account management system. The existing ATL RIDES application will be modified and enhanced

based on ST-CTN project needs and leveraged to create a new, independent application which will be differentiated as the *OTP for G-MAP* subsystem (hereafter referred to as G-MAP).



**SIDEWALKSIM.** SidewalkSim is an asset management system and shortest path (lowest impedance) routing tool for pedestrian pathways. Site inspections provide more detailed ADA and inclusive design and condition data for use in pathway accessibility analysis. SidewalkSim identifies the best path between any two points in the pedestrian network, given the set of pathway characteristics and any user-specified needs and route penalties.



**CV1K.** The Atlanta region is home to one of the largest CV deployments in the United States – CV1K. CV1K is deploying interoperable CV technologies at signalized intersections throughout the Atlanta region using both Dedicated Short-Range Communications (DSRC) and Cellular Vehicle to Everything (C-V2X) technologies to deliver safety and mobility-based applications. The program provides support to configure, operate, and maintain CV infrastructure and applications, including TSP. Gwinnett County hosts one of the largest concentrations of equipped intersections in the first phase of this deployment.



**CVTMP**. Gwinnett County's Connected Vehicle Technology Master Plan (CVTMP) sets out to develop and improve economic viability and quality of life, address the needs and challenges to motorized and non-motorized modes, establish guidelines for deploying technology, and have broad applicability to Gwinnett, other local jurisdictions, and across the state—to set the standard for implementing CVs. Among the high priorities is establishing a mobile accessible safety program and alternative strategies for TSP in Gwinnett County.



**STM**. The Space Time Memory (STM) platform processes traffic volume and speed data from multiple monitoring and modeling sources, tracks network performance measures, and predicts evolving route conditions using traditional and machine learning techniques. The STM projects trip trajectories through the transportation network, as network conditions change in space and time. This tool will be applied to analyze and predict performance through the multi-modal transportation network. The shortest path analysis will be applied to the combined roadway, transit, sidewalk, and Shared-Use path networks, allowing routing decisions to incorporate travel time, safety, and other impedances into path selection.

In addition, the ST-CTN system will include the development of a PMD that is not currently in use (described below).



**PMD.** The PMD is a data storage and distribution tool that archives operational, survey and performance metrics of the system. The PMD is envisioned to ingest, quality check, and curate all types of data – static and dynamic, structured and unstructured, open and private datasets. The PMD will store the datasets so that they can be viewed and accessed based on user roles. The PMD will include a public PMD for the public to access open data presented on an interactive dashboard to the public.

## 1.5 Project References Related to the STP

The STP references the systems engineering documents from the ST-CTN. **Table 1** provides a summary of these documents and their relationship to this document.

**Table 1. STP Related Project Documents and Relationship to Testing**

| Related Task | Summary and relationship to testing |
|---|---|
| **Phase 1 Concept of Operations (ConOps)** | The ConOps reviews the system concept and the needs that drive the system design. The needs are the basis for the system requirements which the system will be verified and validated by in Phase 2. Testing in the real world of ConOps use cases provide validation and acceptance of user needs and expectations. |
| **Phase 1 System Requirements (SyRS)** | The SyRS documents the system requirements that must be verified during system testing in Phase 2. System requirements are updated throughout Phase 2 and are managed under strict configuration control. |
| **Phase 1 System Engineering Management Plan (SEMP)** | The SEMP describes the systems engineering process including testing, verification and validation methods and processes for all subsystems and components. |
| **Phase 1 Integrated Complete Trip Deployment Plan (ICTDP)** | The ICTDP describes testing, verification and validation processes required to accept the system. These processes are refined in this document. |
| **Phase 2 System Design Document (SDD)** – working | The SDD provides detailed design of the system and its components. The design will include full traceability to the requirements. The design elements are also tracked to requirements and unit, integration and system tests are tracked to the design elements. |
| **Phase 2 Interface Control Document (ICD)** – working | The ICD provides details of the interfaces. Interfaces demonstrate how multiple systems and components are integrated. Open interfaces may include conformance and testing tools. Proprietary and emerging standards may need to be tested against the boundaries of the interface rules. To that end, conformance, exercising and testing interfaces are a critical part of testing. Testing platforms and example datasets are needed to fully test these interfaces. |
| **System Test Results Summary** | The System Test Results Summary documents the results of the executed test cases and procedures. |
| **Operational Readiness Plan (ORP)** | The ORP includes information on acceptance testing procedures and processing, including summary reporting of test procedures and stakeholder demonstrations that validate and verify system needs and stakeholder expectations. |

## 1.6 Document Overview

The ST-CTN STP is organized as follows:

- **Section 2: System Test Plan Overview** provides a high-level overview of system test phases and processes.
- **Section 3: Agile Test Processes** describes the agile tests conducted for each sprint.

- **Section 4: Bench and Configuration Testing**

- **Section 5: Full System Testing** summarizes the types of testing required for full system testing and documents the dependencies.

- **Section 6: Defect/Deficiency Tracking and Resolution Processes** documents the categories for defects or deficiencies, testing impacts, resolution, and tracking.

- **Section 7: Test Cases and Requirements Traceability** tracks the requirements against the test cases.

- **Section 8: References** identifies external documents referenced throughout the STP document.

# 2 System Test Plan Overview

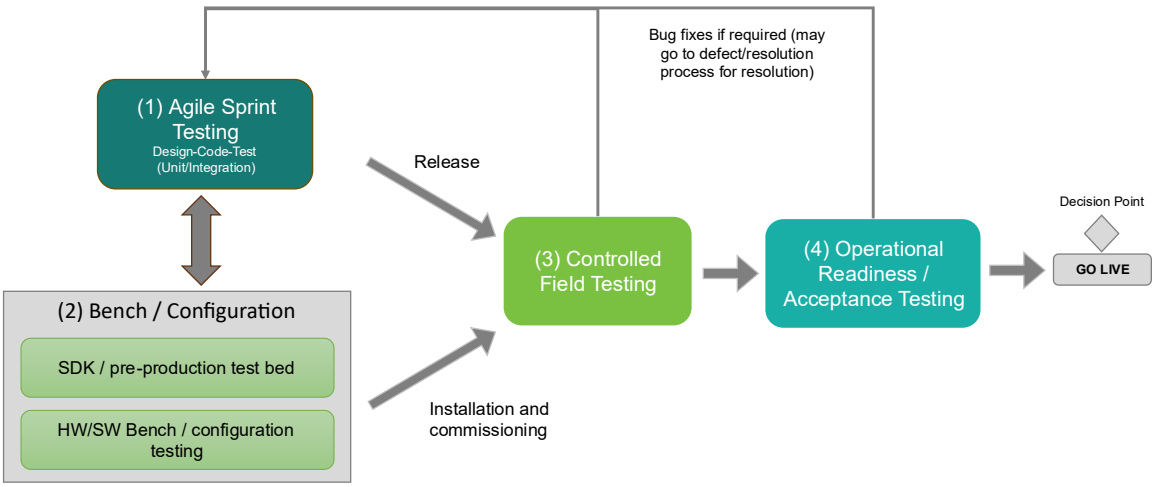
The STP describes the testing procedures for all components, subsystems, and dependencies of the ST-CTN project. The STP describes the types of testing that will be conducted for each subsystem, detailed test procedures and resulting documentation.

System testing is composed of two concurrent processes – Agile development and bench/configuration verification of external / existing systems. All system development will use an Agile development process for the major subsystem components including the STM / private PMD and G-MAP. The public PMD, CV components (e.g., PED-X), indoor navigation system (INS) and Ride Gwinnett – Ride Request functions *are existing or proprietary systems*. The STM/PMD and G-MAP will integrate these systems through interfaces. To that end, integration elements will be tested during the Agile process, while existing and external system functional testing will consist of verifying and validating that the results meet system requirements and user needs.

## 2.1 Overview of High-Level Testing

The high-level testing overview as shown in **Figure 4** describes the high-level approach for testing systems in-development (Agile) and existing systems (Bench/Configuration). The two processes are integrally connected through interface integration. APIs or Software Development Kits (SDK) are tested during the Agile development of the STM or G-MAP, while any hardware component or configurable commercial-off-the-shelf (COTS) software product is verified through its output or proprietary configuration tools. The testing approach merges into a single process flow once the hardware/ COTS is deployed into a controlled environment for verification and validation by stakeholders. These controlled tests are typically conducted manually by trained staff who follow a formal testing procedure. Once controlled field testing is completed, the system is deployed into a production environment where full, comprehensive testing occurs, and the system is demonstrated to stakeholders.

The decision point to go-live depends on the passage of acceptance criteria – verification that all requirements are met and validation that the system performs as expected (that user needs are met). The ORP will outline the system acceptance criteria.




*Source: GDOT, 2023*

**Figure 4. High-Level Testing Approach**

As depicted in **Figure 4**, the STP/PMD and G-MAP subsystems include three major testing phases and processes for defect tracking. A high-level summary of each expected test phase along with the process of documentation is described as the following:

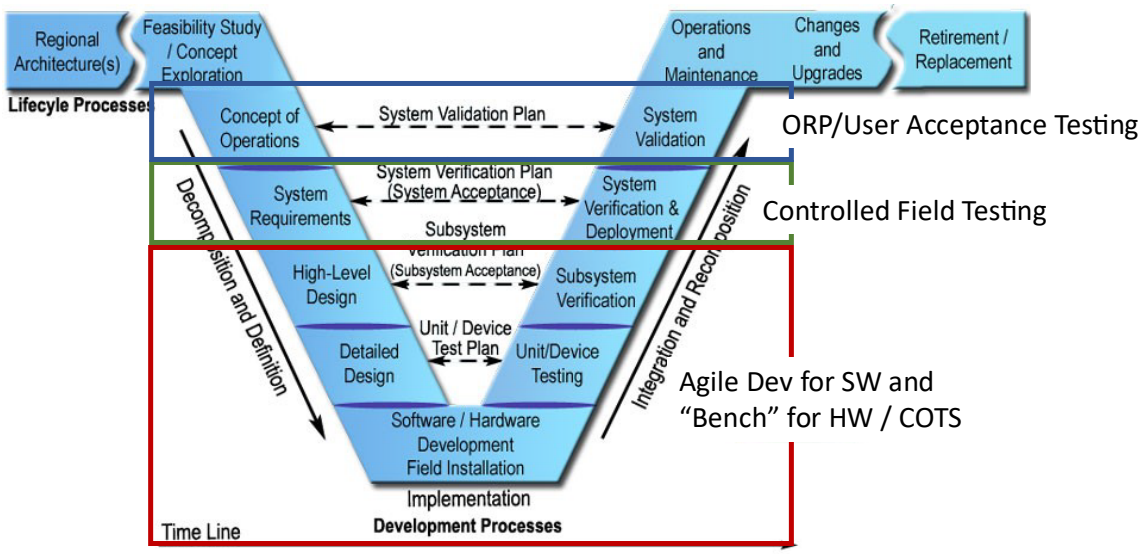
1. *Agile Sprint Testing*: Agile development includes a pre-production environment where each user story is tested and then merged and tested with the code base. Both the STM and G-MAP will undergo this process before it is released to controlled field testing. The expected types of testing to occur during Agile development consist of unit and integration testing.
    a. *Unit Testing:* Each user story will be tested for acceptance of functionality before the code is integrated into the larger code-base. A user story may consist of a self-defined unit of the subsystem (e.g., notification in G-MAP) or an integration element using an API or SDK from an external system (e.g., G-MAP and Ride Request API). Testing of the interface is typically conducted using a pre-production environment provided by the vendor of the existing system or COTS.
    b. *Integration (merge) Testing:* Integration testing is composed of merging new or revised code into the base code, for example, when a user story module is merged with the G-MAP code base.
2. *Bench and Configuration Testing*
    a. Bench testing is the inspection stage of receiving and testing equipment to ensure hardware or software meets the expected specifications. Examples of bench testing include inspecting that the battery is fully charged, determining if SDK can be compiled correctly, or if equipment is in working order. It might also consist of analyzing that the equipment operates within its specified parameters.
    b. Configuration testing consists of determining whether the system produces the desired results once the equipment or software is installed.
3. *Controlled Field Testing (CFT):* Once the Agile process issues a release, user acceptance Alpha testing will begin. Alpha testing continues after each release and includes the integrated components when they are in a ready-state to test (e.g., hardware has been inspection, and COTS configured). The testing may be conducted in the field, a simulated environment, or test bed. Since controlled testing will occur throughout the Agile process with interim releases, full system testing occurs on the system that is currently deployed. The final CFT for the full system occurs at the end of the sprints. This

final set of testing – using automated and manual test procedures for a comprehensive set of test scenarios and data sets- will cover the full system testing in a controlled environment.

4. *Operational Readiness / Full System Testing:* Once the final set of CFT occurs and the system is fully deployed, the full / operational testing will be performed in the production environment ("real-world"). Users will be recruited to participate in beta testing of the integrated systems in the production environment. Testing documentation will show that the tests have been verified against the requirements and validated the user needs / expectations. The use cases developed in the ConOps to meet user needs will be demonstrated to key stakeholders during acceptance demonstrations.
5. *Defect / Deficiency Tracking and Resolution Processes:* The testing process also includes a process to track defects and deficiencies discovered during the testing process. This tracking and resolution process tracks a defect through its lifecycle from detection to resolution, including describing the tests that are required (i.e., regression tests) to show that the defect has been fixed.

### 2.1.1 ST-CTN Testing Approach and the "Vee" Systems Engineering Diagram

The "Vee" systems engineering diagram visually represents a sequential development process, but does not necessarily describing the integration of COTS, existing systems, or Agile methodologies. The types of system development and verification and validation (V&V) approach used by the ST-CTN can be summarized by three tiers that correspond to the three phases depicted in **Figure 4** (Agile Development for software (SW) and "Bench" for hardware (HW)/COTS, CFT, and ORP/User Acceptance Testing). **Figure 5** describes the "Vee" diagram testing and V&V tiers in context of the three testing tiers defined by the ST-CTN project.




*Source: USDOT with Annotations by GDOT, 2023*

**Figure 5. Relationship Between ST-CTN Testing Approach and System Engineering "Vee" Diagram Testing Phases**

Starting from the junction (bottom) of the "Vee" diagram, the Agile process incorporates detailed design, software development and unit testing into incremental units (i.e., user stories). Prior to

moving to the next user story, the current user story code is integrated into the main code branch (or code base). In doing so, merging the current code with the main branch requires that a sample or all previous automated test procedures are executed to verify the code works as expected. Because each user story is associated with a requirement(s), the system engineering traceability is preserved from requirement to design and test procedure / case result. The Agile demonstration ceremony incorporates interim end-to-end demonstration (system verification) user acceptance (system validation) through stakeholder demonstrations. This provides stakeholders the opportunity to contribute input and see progress during development instead of waiting until the end of complete system development, when changes will be much more difficult to make.

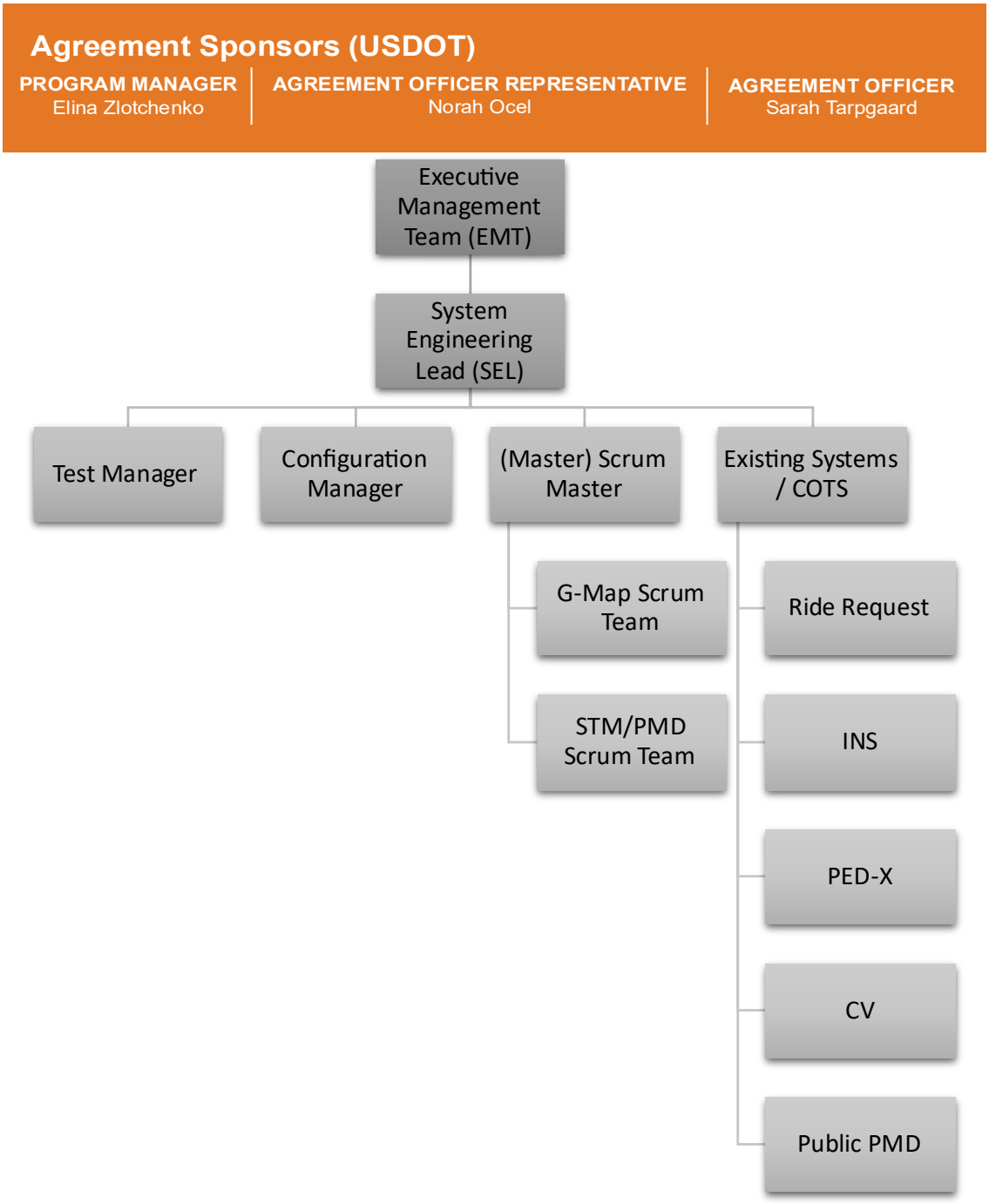
COTS and existing systems such as the indoor navigation system, traffic signal system, and the Ride Request application do not include detailed or high-level design elements for their functional components. These elements are proprietary or out of scope of the ST-CTN system. Rather, they require component inspection and analysis that the operations and/or results are as expected. This bench testing corresponds to device testing, and the configuration testing provides end-to-end results that correspond to the subsystem verification.

The CFT corresponds to the system acceptance or system verification plan. Several times during the Agile development process, a release will be produced on the pre-production platform that will be subject to CFT testing. CFT testing includes a combination of automated testing, data analysis, and alpha testing. Each requirement is tested in a controlled environment against boundary data and through formal test procedures. For example, the same test procedure may be tested on multiple test platforms and channels, such as web browsers (e.g., Edge, Chrome, Firefox) and mobile operating systems (iOS, Android). These tests are end-to-end tests (also called black box testing). All technical – functional, performance and operational - requirements will be tested in the CFT phase. User facing software tests will include accessibility and usability. The ST-CTN will conduct several CFT tests following each major software release (see schedule). At the final CFT, the system should be fully integrated and tested.

The ORP describes elements of the system validation plan which includes how this STP will be implemented. The ORP ensures that the system meets user needs and expectations. The ORP includes a summary of the controlled field test results and demonstration of the ConOps as documented by the use cases. Most user-facing systems also include beta testing or stakeholders to test the application in a real-world environment under real-world conditions.

## 2.2 Testing Organization

The technical team organization is depicted below in **Figure 6**. USDOT is the project sponsors and provide oversight and advisory support. The Test Manager is responsible for implementing the provisions of this Plan. The STP is governed by the System Engineering Lead (SEL) and the Executive Management Team (EMT). The set of roles and responsibilities related to each entry in the organization chart are listed in **Table 2**.




*Source: GDOT, 2023*

**Figure 6. Technical Team Organization**

**Table 2. Technical Team Roles and Responsibilities**

| Role | Responsibilities Related to Testing |
|---|---|
| Agreement Sponsors (USDOT) | Sponsor, advise and oversee project including processes, documentation, approval of system deployment and operational readiness. |
| Executive Management Team (EMT) | Oversee project including development and efficacy of system deployment. |
| System Engineering Lead (SEL) | Oversee and coordinate technical teams, establish development and testing schedules, anticipate and mitigate obstacles for deployment, review and escalate issues to EMT.<br>Implement system engineering processes including providing verification and validation of system requirements and user needs. |
| Test Manager | Responsible for implementing the Plan.<br>Coordinate Controlled Field and User Acceptance testing including:<br>• Review test procedures for Alpha Testing<br>• Assign test procedures to testers.<br>• Collect and aggregate test results and report issues to Master Scrum Master for Defect/Resolution Process (see **Section 6**).<br>• Review and track resolution process for testing<br>• Review comprehensive verification and validation of requirements-test traceability.<br>• Summarize results in test results documentation (see **Section 2.5**) |
| Configuration Manager | Manage and control configuration of system Needs to Requirements Traceability Matrix (NRTM) from needs, requirements, design, and testing. |
| (Master) Scrum Master | Manages the scrum teams and product owner. Identifies dependencies with existing and external system integrations. Design, code, and test subsystems and integrate systems (see **Table 4** for additional definitions of Agile / Scrum terms). Two scrum teams will be developing and testing code:<br>G-Map Scrum Team – Arcadis/IBI developing the G-MAP application<br>STM Scrum Team – Georgia Tech / Georgia Tech Research Institute (GTRI) developing the STM platform and private PMD. |

| Role | Responsibilities Related to Testing |
|---|---|
| Existing System / COTS (operators) | The operators of existing systems, software-as-a-service (SaaS) and COTS that are integrated into the ST-CTN deployment. These teams support the integration of existing or COTS with G-MAP and STM. These include the following systems and organizations:<br>• Ride Request – Gwinnett County, Ride Gwinnett and their vendor CAD/AVL Avail Technologies<br>• Indoor Navigation – GDOT and their indoor navigation vendor.<br>• PED-X – GDOT and their vendor, Q-Free<br>• Connected Vehicle (CV) – GDOT and their vendor, 360ns<br>• Public PMD – ICF using the ARC software license for Power BI |

## 2.3 Types of Testing

Each subcomponent will require different types of testing to ensure that systems are functioning as expected before integration which also requires specific tests. The different tests will be completed at different phases of the Agile process. The expected test types are provided with definitions below.

- **Functional Testing:** Tests that the core functionality operates as expected. Testing in this category may be completed using automated scripts and formal test procedures performed during all phases of testing.
- **Performance Testing**: Demonstrates that the system meets the expectations of the end-user. These tests use automated scripts that are designed to meet user acceptance criteria.
- **Security Testing:** Tests for security of user data and other cyber security concerns. These types of tests are typically accomplished through penetration testing that provides an authorized simulated attack on the system to evaluate the security of the application.
- **Integration or End-to-End Testing:** Integration testing will focus on the integration of the current sprint with the main code branch. Integration testing will be conducted to include a *comprehensive suite of unit tests* for the current code base in addition to the new test scripts developed for the current sprint. In addition, integration testing for interfaces will include end-to-end workflows that include data that corresponds to boundary and complex conditions in a simulated environment. Testing is conducted using automated test scripts.
- **User Interface Testing:** This testing includes a series of automated tests including accessibility, snapshot and pixel testing (these tests are defined in **Section 3.3**).
- **User Acceptance Testing (UAT)*:*** Traditional user acceptance testing is divided into two categories Alpha and Beta testing.
    a. **Alpha testing:** Alpha testing consists of formal testing that exercises all the acceptance criteria defined by the system design (and traced to the requirements). Testers are trained and submit formal test summary results based on one or more test platforms or criteria.
    b. **Beta Testing:** Beta testing will be conducted by recruited, real-world end-users where application performance, user acceptance, and user satisfaction will be

measured for acceptance. Beta testing is free-form and not tracked by the system.

- **Bench Testing:** Bench testing exercises the operational performance of acquired devices or software. This may include testing the battery charge, operational frequency and range, data transformation, or software compilation.
- **Configuration Testing:** Configuration testing examines the optimization following placement or entry of the recommended configuration parameters. Configuration testing may include tuning the configuration parameters to optimize the end results. This may also be considered an installation and commissioning activity.

Formal testing that occurs during the Agile and CFT (functional, performance, security, integration, user interface and alpha tests) will use testing procedures (e.g., automated or manual scripts) and be assigned a unique identifier that traces to one or more requirements (and Agile user stories). In addition, Bench and Configuration testing will include a checklist of requirements to inspect or analyze during intake, installation, or commissioning.

Beta tests will not be tracked because they will be tested by volunteer users who use the integrated system in a real-world environment. Reported bugs, anomalies or user feedback will be assigned a unique identifier to trace its occurrence, testing platform and other identifying information.

## 2.4 Verification Methods

The STP will use four verification methods to ensure that needs and requirements are met.

- **Inspection** – Where the requirement is verified by visual inspection or review of software code.
- **Demonstration** – Where the requirement is verified by demonstrating a specific capability or action. Demonstration is usually verified during the execution of the test itself; you witness the capability accomplishing the requirement.
- **Analysis** – Where the requirement is verified by an analysis method. As an example, performance metric algorithms are usually verified by analysis to ensure that the data and calculations are ingested and processed correctly.
- **Test** – Where the requirement is verified by a test that is run and then later verified by analysis, inspection or demonstration. Many times, performance requirements are verified by test as it may not be immediately obvious if a performance requirement was met until analyzing logs to determine the precise times when events occurred.

## 2.5 Test Documentation

This section lists the test documentation and artifacts that will be produced and stored. The documents include the following:

- **System Test Plan** – The STP describes the testing processes for each of the system subsystems, applications, and hardware elements. This document is the STP.
- **System Test Procedure** – Each test procedure includes an introduction, body and scoring section. The introduction describes the purpose, requirement(s) under test (if applicable), the execution conditions (i.e., trigger event), and inputs. The body of the test

procedure includes a sequence of actions, expected results, remarks, and blank entry for the tester to mark the results. The end of each test procedure includes a scoring section for the tester to mark pass or identify any defect in the system.

- **System Test Case** – A formal test case is developed and conducted during the CFT for each major release. The test case describes one or more **system test procedures** that are manually conducted to exercise the system on different platforms (web, mobile), and by different accessibility features.
- **System Test Case Results** – The system test case is an aggregation of system test procedures exercised on various platforms by users testing different access channels. The results will be documented in the **System Test Results Summary**. Each test case is an aggregation of a class of test procedures grouped by platform, release, accessibility feature, and function.
- **System Test Results Summary (STRS)** – System test case results will be summarized in a System Test Results Summary report. The report will include a high-level summary of each test procedure objective, test case results, and test results. In the event of a failed or inconclusive test, the conditions under which the test failed will be described and the remediation action taken. Each remediation action shall be indexed and tracked as part of the acceptance process. In addition, the **User Acceptance Testing Summary** is part of the STRS.
- **User Acceptance Testing Summary** – These are informal activities conducted by Beta Test volunteers in an unstructured test environment. UAT participant profiles and comments will be collected, logged, and addressed as needed. The UAT testing environment enables input by users to document any issues. The UAT summary will be published periodically. High priority issues will be submitted as a ticket via the ST-CTN Agile Development Process Tool (Jira) backlog. The summary will be published periodically during the UAT period. It will continue during Phase 3.
- **Operational Readiness Testing Plan** – The Operational Readiness Test Plan (ORTP) provides a summary of the set of tests and supporting demonstrations that together establish operational readiness. The System Test Results Summary is a subset of the Operational Readiness Testing Plan results. It also includes the UAT summary results.

Most testing documentation will be stored in the project Teams folder. Agile sprint and release testing activities will be verified in Jira and transferred to test artifacts that are stored in the project Teams folder.

## 2.6 High-Level Testing Schedule

The high-level Testing Schedule (see **Table 3**) includes the User Acceptance Tests only. The Agile development process integrates unit and integration testing (e.g., merge testing) prior to a sprint being confirmed as “Done” (described in **Section 3**). Alpha Release testing and Beta testing are part of the User Acceptance Testing (described in **Section 5**). In addition, there are several bench and configuration tests conducted on equipment, COTS or SaaS acquired systems that require “bench and configuration” inspections. These include the INS, Ride Request system and the Public PMD. The PED-X system will be tested as part of its integration with G-MAP Release 2 as listed in the Software Development Schedule (SDS).

**Table 3** includes the following columns:

**SDS WBS #:** The index from the System Design Schedule work breakdown structure (WBS) – see SDS.

**Test ID:** A unique index that identifies the test

**Name:** Name and description of the test

**Duration:** Number of days over which the test is conducted

**Start Date:** First day of testing

**End Date:** Last day of testing

**Table 3. High-Level Testing Schedule**

| SDS WBS # | Test ID | Name | Duration | Start Date | End Date |
|---|---|---|---|---|---|
| 2.E.6.1 | Alpha Release 1 | Alpha Testing for Release 1 – Core, Mobility Profiles and Live Tracking integrated | 21 | 9/11/2023 | 10/2/2023 |
| | STM Release 1 | STM Release 1 – Network Impedance API | 21 | 9/11/2023 | 10/2/2023 |
| | INS Bench Testing | Indoor Navigation Bench Testing (on-board and installation of beacons) | 21 | 2/5/2024 | 2/26/2024 |
| 2.E.6.2 | Alpha Release 2 | Alpha Testing for Release 2 - Enhanced Routing and Notifications | 21 | 2/5/2024 | 2/26/2024 |
| | STM Release 2 | STM Release 2 – Private PMD | 21 | 2/5/2024 | 2/26/2024 |
| | Ride Request System Testing | Ride Gwinnett – Ride Request System Testing | 21 | 2/5/2024 | 2/26/2024 |
| | Public PMD System Testing | Public PMD Configuration testing | 7 | 2/5/2024 | 2/26/2024 |
| 2.E.6.3 | Alpha Release 3 | Alpha Testing for Release 3 - Indoor Nav and Group Travel | 21 | 3/18/2024 | 4/8/2024 |
| | STM Release 3 | STM Release 3 – final ready for full system testing | 21 | 3/18/2024 | 4/8/2024 |

| SDS WBS # | Test ID | Name | Duration | Start Date | End Date |
|---|---|---|---|---|---|
| 2.E.6.4 | Alpha Release 4 | Alpha Testing for Release 4 - Additional Mobile App Features and Integration with External Systems<br><br>Alpha Release 4 will include Full System Testing of all subsystems in their production environments (e.g., in the field) | 21 | 4/29/2024 | 5/20/2024 |
| 2.E.6.5 | Alpha Release 5 | Alpha Testing for Release 5 - Bugfixes and Clean-up (and continuing into Phase 3) | 4 | 6/10/2024 | 6/14/2024 |
| 2.E.6.6 | Beta Test Training | Recruiting and training of beta testers | 67 | 4/8/2024 | 6/14/2024 |
| 2.E.6.7 | Beta Testing | Volunteers begin beta testing applications | 46 | 4/29/2024 | 6/14/2024 |
| | OR Demo | Operational Readiness Demonstration | 2 | 6/10/2023 | 6/14/2023 |

# 3 Agile Test Processes

This section provides an overview of the Agile Testing processes for the G-MAP and STM subsystems and their integration elements (interfaces with PED-X, INS and Ride Request). **Table 4** contains a set of terms that define key Agile/Scrum concepts.

**Table 4. Definitions of Agile Methodology Terms**

| Term | Description |
|---|---|
| Scrum Team | A scrum team includes a group of individuals, including a scrum master, product owner, and developers, collaborating to complete and deliver software products. |
| Scrum Master<br><br>Master of Scrum Masters | A scrum master guides the scrum process and supports the scrum team, product owner, and organization.<br><br>The master of scrum masters manages the scrum teams and product owner and identifies dependencies with existing and external system integrations. |
| Product Owner | Accountable for effective product backlog (requirements) management which includes that the product is accomplished, and value is delivered. |
| Sprint | The sprint is the phase during which the scrum team completes design, development and testing on one or more user stories that have been placed in the sprint backlog. |
| Grooming | Obtain and define requirements for the development including acceptance criteria for verifying and validating code. |

## 3.1 Agile Development Process

This section describes the Agile Development Process including elements that contribute to testing code developed for the STM and G-MAP. Delivery of the ST-CTN project will follow an Agile/Vee Hybrid methodology where Agile (Scrum) will be combined with a traditional systems engineering process (Vee-Process) (As referenced in **Section 2.1.1**). The approach, depicted in **Figure 7,** establishes user needs, requirements, architecture, and conceptual design early in the process consistent with the Vee-Process, and then pivots to an Agile process. The traditional stages of the Vee-Process will be followed until the system requirements and SAD are completed. These foundational components trace the needs and requirements to aid in the system deployment, while setting the stage for the Scrum process that allows for iterative and incremental development during the design and implementation stage. User needs and requirements (once developed and refined) will serve as inputs to epics and user stories in the

Product Backlog. The Product Backlog and the existing ATL RIDES code-base (which provides the initial main branch codebase) will define the epics and user stories and establish the minimal viable product (MVP).

Scrum is an Agile methodology in which the basic unit of development is called a Sprint. Every Sprint begins with a Sprint Planning meeting in which tasks for the Sprint are defined. This is also the part where the estimated commitment for the Sprint Goal is made. Products will be developed incrementally through several iterations throughout a Sprint process. A Sprint concludes with a Sprint Review meeting where progress is evaluated.

STN-CTN functionality will be incrementally delivered and tested during 3 week Sprints. Released Scrum software products will go through multiple Sprint cycles with each Sprint cycle delivering a portion of the product until the complete product (or planned increment release) is ready for integration with other parts of the system. When Scrum is complete, the process then reenters the Vee-Process for the system V&V activities (in this case after merging into main branch and integration testing).

**Step 1: Software Product and Release Planning.** Content from the Vee-Process deliverables will serve as inputs to the Agile Process. Using these documents as inputs, the ST-CTN Scrum Teams will work with ST-CTN Stakeholder Groups to perform Software Release Planning. During Software Release Planning, user needs and requirements will be broken down into Epics and Agile User Stories. Epics are large user stories, typically one that is too big to fit in a single Sprint. Epics need to be broken down into smaller user and system stories before implementation as part of a Sprint.

User Stories will be captured in the Product Backlog which will be kept in the ST-CTN Agile Development Process Tool (i.e., Jira). The Backlog will be organized as a roadmap - an ordered list of the work to be done in order to create, maintain and sustain the ST-CTN products. It will be managed by the Scrum Master, with direction from the Product Owner, and input from stakeholders and will be visible to all team members depicting a real-time picture of the work that the Scrum Team plans to accomplish during each Sprint.

Once the SDD is created, the team will review the user and system stories and acceptance criteria in the SDD to develop a Product Roadmap and Release Schedule, which is essentially a phased deployment schedule that is shared with the project stakeholders. The Product Roadmap includes a high-level plan that shows when in the future new products are expected to be developed or introduced by the team. The Product Roadmap will define and present major releases of the ST-CTN system. The Product Roadmap captures the schedule for releases including the MVP. The Product Roadmap also identifies times when all software products and hardware will be integrated together. The Product Roadmap will guide the ST-CTN project team in Sprint Planning to ensure that key milestones are met.

The *Definition of Ready* and *Definition of Done* will also be established at this point. The *Definition of Ready* includes the conditions that need to be satisfied by the Product Backlog item before it is considered ready to pull into a Sprint during Sprint Planning.

Ready means that the story is ready to be used in a Sprint. The ***Definition of Ready*** will include the following:

- User Story is defined and complete
- User Acceptance Criteria are defined
- User Story dependencies are defined
- User Story is sized correctly by the Scrum Team
- The person who will implement the User Story is identified; the team members who will conduct the technical and hygiene reviews are identified
- The team has a good idea what it will mean to Demo the story

The ***Definition of Ready*** is encapsulated in the Development Design Template. The Sprint Team will review and ensure that the template is completed and meets all the aspects not only of the story, but the components that enable the function to operate correctly and as expected.

A clear ***Definition of Done*** will also be established that includes an agreed upon set of items that must be completed before a project or user story can be considered complete. The Definition of Done will consist of a checklist of items such as:

- Acceptance criteria is demonstrated to the Product Owner on the quality assurance QA environment
- Code is peer reviewed and is checked in
- Code is documented
- Code is merged back into the main branch
- Version number is updated, if needed
- Unit testing has been completed
- Integration testing (as needed) has been completed
- Development Design Template is reviewed to ensure completion of all identified activities.

**Step 2: Sprint Planning.** Sprint Planning consists of processes related to planning and estimating tasks for an upcoming Sprint. Prior to beginning a Sprint, user stories and their related acceptance criteria will be created and incorporated into the Prioritized Product Roadmap. The Product Owner, with support from the Scrum Master and the rest of the Scrum Team will be responsible for prioritization of the Roadmap. The Product Owner will have the final word on the prioritization. Acceptance criteria include the product characteristics, specified by the Product Owner, that need to be satisfied before they are accepted by the user, customer, or other authorized entities. These are used to measure and compare the characteristics of the final product with specified characteristics.

Once all stories are ready, a Sprint Planning meeting will be held with the team to create a Sprint Backlog containing all tasks to be completed in the Sprint. The team will consist of the Product Owner, Scrum Master, and Scrum Team, and other team members relevant to the Sprint. Other relevant team members may include system owners, COTS vendors, etc. as is necessary, based

on the focus of the development. The purpose of the Sprint Planning is to reach agreement with the team members of the time allotted to complete the tasks assigned to them for the sprint. The Scrum Master and ST-CTN Scrum Teams will estimate the effort required to develop the functionality described in each user story, and the ST-CTN Scrum Team will commit to delivering the tasks during the Sprint period.

**Step 3: Implementation.** Implementation includes the execution of the tasks and activities in a Sprint to iteratively create the ST-CTN product. In this step, the ST-CTN Scrum Teams will work on the tasks in the Sprint Backlog to create deliverables, features, or products. The ST-CTN Agile Development Process Tool, Jira, will be used to track the work and activities being carried out.

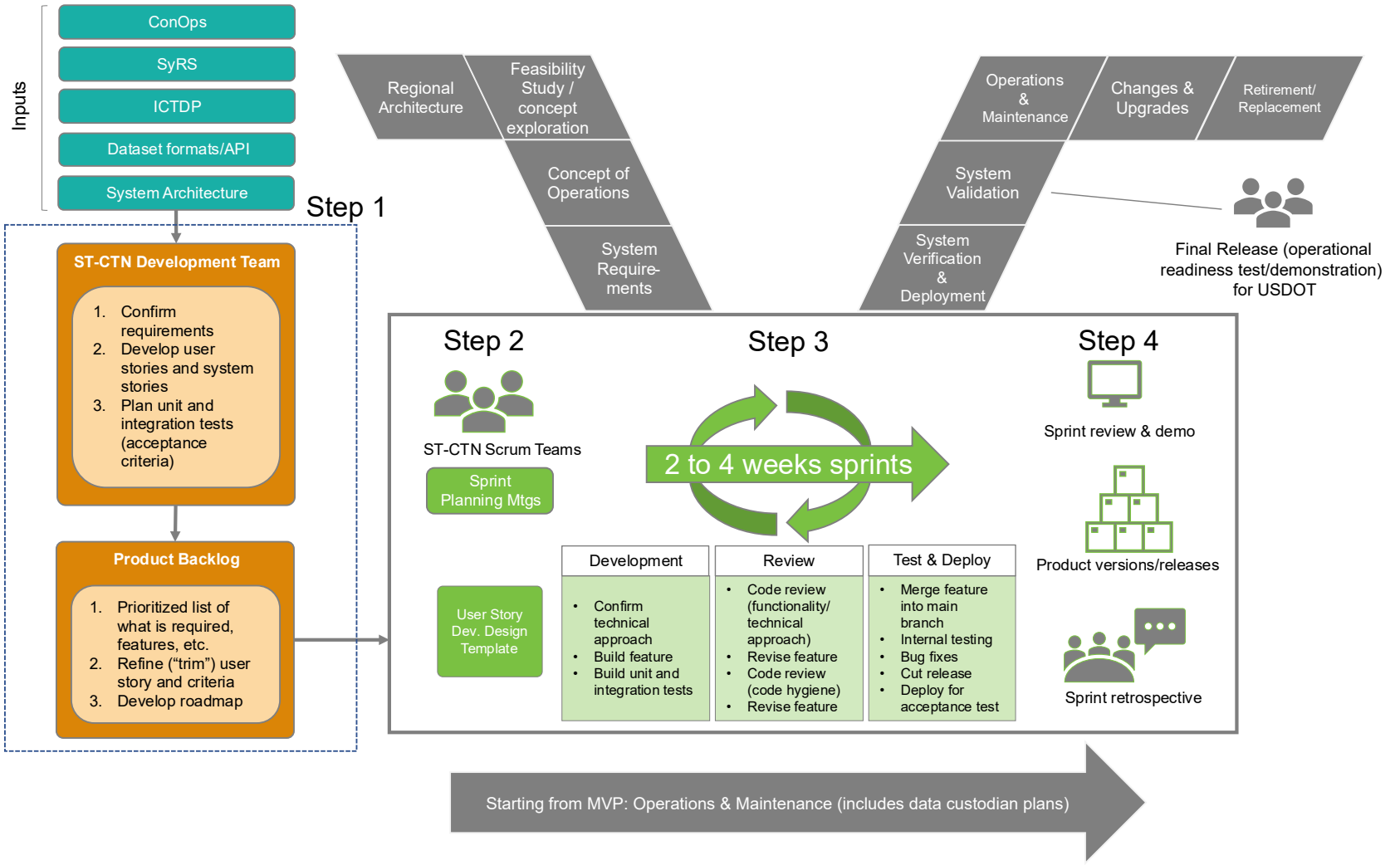
Key activities included in this step include:

- **Development.** Development includes confirmation of the technical approach, building features, and/or conducting unit and/or integration tests. The Scrum Team will work to deliver an MVP in accordance with the Product Backlog. The team will then incrementally develop additional features to deliver the full product.
- **Code Review.** Review includes conducting two types of code review (functional and technical) and code hygiene (style and documentation).
- **Documentation.** Documentation includes developing appropriate documentation, uploading materials to Jira.
- **Test and Deploy.** This activity includes testing user story features, conducting internal testing including user interface and accessibility tests for user facing code, and identifying and fixing bugs (if required). Testing proceeds with merging the features into the main branch first by ensuring a comprehensive set of unit tests pass and then end-to-end testing to verify integration into the main branch codebase. Rather than taking a big-bang approach to testing, testing will occur incrementally as part of the Agile Process. All user stories will be mapped to user needs and requirements – and will include acceptance criteria and test cases.

**Step 4: Sprint Demonstrations.** This step includes reviewing the deliverables, features, and products and the work that has been done in the current Sprint. The ST-CTN Scrum Team will demonstrate the Sprint deliverables to the Product Owner and stakeholders in a Sprint Demonstration meeting. Where appropriate, proxy users provide feedback. Discussions, actions, action items, and agreements will be documented in meeting notes and stored on the project document site. Feedback that enhances the product is documented as a “ticket” and is added to the product backlog. The purpose of a Sprint Demonstration is to secure approval and acceptance from the Product Owner and/or stakeholder groups for the deliverables created in the Sprint.

**Step 5: Stakeholder Demonstrations.** At each release, stakeholders or their proxies are invited to provide feedback feature releases. Discussions, actions, action items, and agreements will be documented in meeting notes and stored on the project document site. Feedback that enhances the product is documented as a “ticket” and is added to the product backlog. The purpose of the Stakeholder Demonstration is to confirm user acceptance from the stakeholder groups for the deliverables created in the release.

**Step 6: Release Version.** After each Release, the main branch code is updated and used during Alpha Testing. Alpha testing is testing the features that have been released to a pre-production environment so that the environment can be controlled and expected behaviors identified. If a defect is identified, a ticket is entered into backlog.




*source: GDOT, 2023*

**Figure 7. Hybrid Agile / Vee Development Process**

### 3.1.1 Backlog Tickets

Every activity is issued as a ticket. Issues due to bugs and defects are tracked as a new ticket in the product backlog, as a user story. When issued, the tickets become part of the product backlog or work to be done to complete the Agile product. All tickets are assigned a priority ranking as shown in **Table 5**. A defect, depending on its severity will be assigned a critical (P1), high priority (P2), medium to high priority (P3) ticket, or normal priority (P4). The issue priority is defined by a ticket type and scheduled based on its priority. Priorities are set by the Product Owner. . Most user stories and task items are designated a P4, or Normal Priority Ticket. .

**Table 5. Priority Ranking for Tickets**

| Ticket Type | Priority | Description |
|---|---|---|
| PI | Critical Ticket | System is not responding or system outage. Unscheduled downtime. |
| P2 | High Priority Ticket | Major feature or functionality within the system is not working, causing a critical disruption for users. Must be fixed immediately. |
| P3 | Medium to High Priority Ticket | Key defect or functionality that should be embedded into the **Active Sprint** to be worked on and completed before the sprint end date. These are the tickets that were not part of the sprint planning meeting but were identified after the sprint started and must be completed before the sprint end date. |
| P4 | Normal Priority Ticket | Defect or functionality that should be reported to the scrum master and go through the grooming process. Work items will be prioritized by the scrum team. |

## 3.2 Agile Test Processes Overview

As described in **Section 3.1**, a sprint comprises design, code and testing activities. An incremental module of code is written, tested and then merged (and tested) into the main code branch. Test planning begins as soon as a feature is included in the Sprint Backlog for development. Test planning for each software feature begins with specifying the inputs, testing cases, and expected results using scenarios for how the feature handles a variety of operations or conditions.

A user story may consist of ingesting data, exchanging messages or executing application tools (i.e., SDK) from an external component. When the user story includes ingesting or exchanging data with an external component, the unit test covers the interface or SDK tests. After each sprint is unit testing, the sprint is not considered Done until it is merged and tested with the main branch codebase. During merging previous unit tests from other previous sprints are rerun to ensure that the new code does not “break” the codebase. Following the merge tests, end-to-end testing is conducted to verify that the system performs as expected. The process is shown in **Figure 8.**

.

As shown in **Figure 8**, the Agile sprint process embeds unit and integration testing in each sprint. The sprint and release cycle consists of the following steps:

1. Prior to a sprint, the product owner, stakeholders and other subject matter experts refine the needs and requirements of the user story. This includes:
    - Design solutions to address gaps
    - Formulating user experience / user interfaces
    - Grooming user story to identify objective and acceptance criteria
    - Product owner review and approval for Ready to sprint
2. During each sprint, developers implement the design and test the code. STM and G-MAP unit and integration tests are described in **Section 3.3** and **3.4**). This includes:
    - Implement the design including various components (function, interfaces, data, etc.)
    - Test the code for acceptance criteria against acceptance criteria (unit test)
    - Merge the code with the base code (integration test) and perform regression testing to verify correct operations of the new sprint with the base code.
        - If the sprint is incomplete or a defect is detected, then the sprint is extended to the next sprint cycle.
    - Sprint code is demonstrated to product owner and limited set of stakeholders to verify acceptance criteria.
    - Sprint is considered Done and ready for internal versioning
3. Following each Epic, the code is versioned and readied for Release. The release constitutes a complete set of functionality that is ready for end-user stakeholder review. Prior to end-user stakeholder holder review, the following testing tasks are performed:
    - Develop and conduct Alpha testing cases and procedures in the controlled field environment
        - If a defect is detected, then the defect triggers the creation of a ticket, and the defect tracking and resolution process is initiated (see **Section 6**).
    - Simultaneously, the Release is ready for Stakeholder Demonstration
        - If stakeholders recommend improvement or alteration, then a ticket with the recommendation is included in the product backlog
    - Following Alpha testing and the stakeholder demonstration, the code is versioned for production.
        - If this is the third release or higher, then the product is ready and released for Beta Testing.
4. During Beta testing, end-users test the application in real-world scenarios. The testing tasks included the following:
    - Beta testers are trained in the how to use the applications and how to issue a defect notice to the developers.

    - If a defect is detected, then the defect triggers the creation of a ticket, and the defect tracking and resolution process is initiated (see **Section 6**).
  - As new code is generated or as defects are fixed, new versions of the software are generated, and updates pushed to the respective application production platforms. Users are alerted to the version updates through native application functions (e.g., Test Flight).
5. At the conclusion of all testing, an Operational Readiness Demonstration is conducted to highlight the operations and efficacy of the system.

### 3.2.1 Agile Test Environments

Several environments will be set up to develop and test the Agile code.

- For G-MAP, the development environment is used to perform unit testing. A separate test environment (referred to as the quality assurance (QA) environment) is used to merge sprint code with the main branch codebase. The QA environment includes test data generators and simulators for external system and interface exchange.

- The STM has a development and QA environment. Unit tests will be performed in the development environment and the QA environment will be used for pre-production integration testing. The Private PMD has the same development and QA environment structure as the STM. The PMD QA environment also includes data generators to allow for external interface testing.

Each external system with an SDK or API is deployed in a “sandbox” environment in which the Agile teams can exercise and test their interfaces during unit and integration testing.

The QA environment for the G-MAP and the QA environment for the STM will be used as the pre-production testing environment for the CFT. When the external / existing systems are on-line, the environments will change the interface / SDK to the operational / production platform (Alpha Release 1).

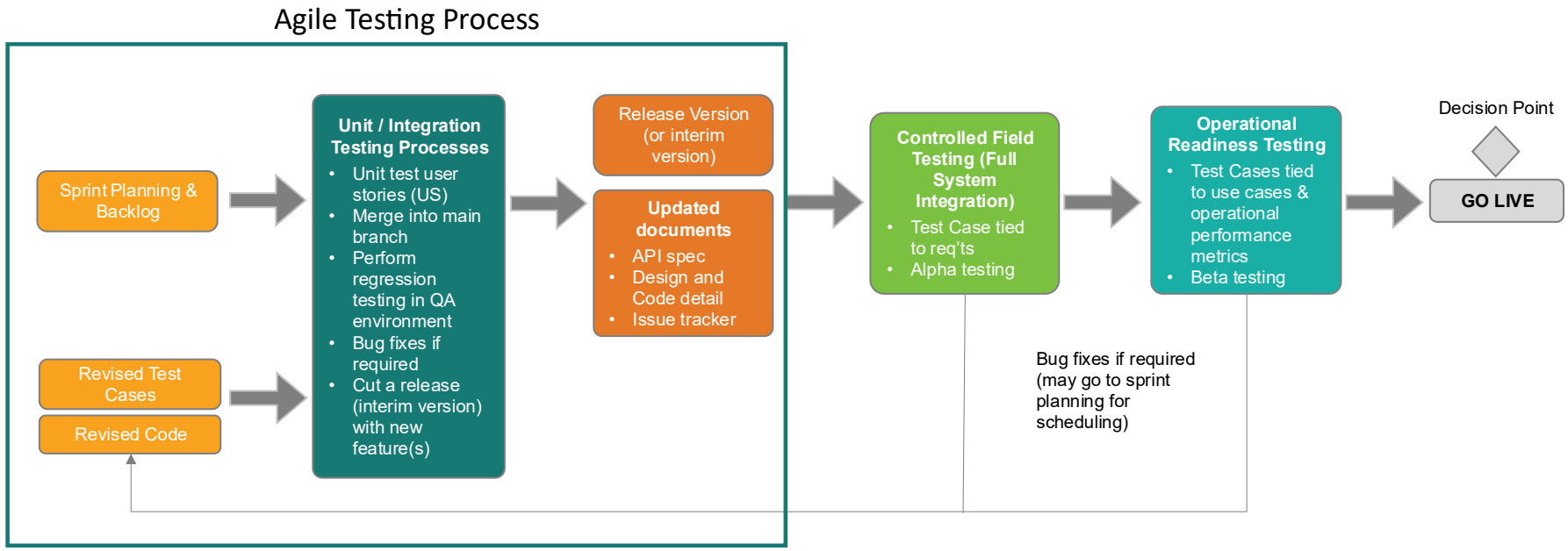
Once the system is fully tested, the code will be transferred to their respective production environments which will serve the User Acceptance and ORP test.

### 3.2.2 Agile Test Case Storage

All test cases, including their respective test procedures and test data and base version code links, are stored in a GitHub site.

The G-MAP repository (https://github.com/opentripplanner/OpenTripPlanner/tree/dev-2.x/src/test/java/org/opentripplanner) contains tests that address all the features, functions, and performance for the legacy and new components developed for this system. The collection is used for regression testing during the merge process to verify that the system continues to work safely, correctly and without defect.

The STM repository (https://github.com/gti-gatech/STM4US) contains tests that address core features, functions and performance tests for developed components. The collection is used for regression testing during the merge process to verify the system continues to work safely, correctly and without defect..




*Source: GDOT, 2023*

**Figure 8. Agile Testing Process**

## 3.3 G-MAP Testing Process

G-MAP testing includes multiple types of testing depending on the G-MAP component. These tests are conducted during a sprint based on the function and component. For example, web-based user interfaces testing includes specialized tests to verify correct operations, useability, accessibility and flows for web browser applications; similarly mobile application UI testing includes similar tests but using different set of tools. Testing tools include online libraries with access to existing automated testing scripts that can be used to test components of the system. These libraries and specialized automated testing scripts are described throughout the G-MAP testing process within the next five subsections describe tools used for different aspects of the G-MAP product. The subsections describe the testing and related tools corresponding to the different G-MAP components. These testing tools and methods are divided into the following categories:

1. OTP routing engine
2. OTP (User accounts middleware)
3. Web application user interface
4. Web application manual quality assurance and testing procedures
5. Native iOS/Android mobile app user interfaces

Each component includes automated unit testing and end-to-end (integration) testing procedures.

### 3.3.1 OpenTripPlanner Routing Engine

The core OpenTripPlanner project covers the multimodal routing server, and includes both automated unit testing and deployment-specific smoke testing (described below):

**Automated Unit Testing:** The OpenTripPlanner backend codebase includes a comprehensive suite of over 3000 unit tests which are run as part of the project's continuous integration (CI) workflow any time a pull request (PR) is opened or merged for the master, dev-1.x, or dev-2.x branches. The G-MAP OTP instance is deployed from an IBI-specific fork of the dev-2.x branch.

To view the detailed results of the unit test suite for a specific PR merge, open the corresponding PR on Github (e.g. https://github.com/opentripplanner/OpenTripPlanner/pull/4978), click the "Checks" tab, expand the "OTP CI Build" section, open the "build-windows" job, and expand the "run tests" section.

Code coverage (i.e., the percentage of project code covered by unit tests) for the OTP project is tracked at https://app.codecov.io/gh/opentripplanner/OpenTripPlanner/?trend=all%20time.

**Automated Smoke Testing:** In addition to the general unit tests, IBI's OTP deployment fork also includes a series of automated, region-specific "smoke tests" that set up a basic OTP deployment and perform standardized test searches, looking for unexpected changes in search results from one OTP build to the next. These tests are intended as a last sanity check when merging

upstream changes from the main OTP codebase because these changesets tend to be very large.

Each smoke test follows these steps:

1. Takes the latest upstream code
2. Downloads up-to-date GTFS and OSM files
3. Builds a complete graph
4. Starts an OTP instance
5. Possibly configures realtime updates for transit
6. Sends API requests against OTP and asserts that the result is within expected parameters

### 3.3.2 OTP Middleware

The OTP middleware layer sits between the OTP routing back-end and the various front-end clients (web and mobile), and handles functions such as user authentication, account/preference management, saved-trip monitoring, and data logging/analytics. The middleware consists of both a server/API component (otp-middleware) and a web-based administrative client (otp-admin-ui). Automated testing for the middleware layer includes both unit tests and end-to-end tests.

**Automated Unit Testing** is part of the continuous integration process for the otp-middleware module, which contains a suite of unit tests that is performed for each pull request. To view the test suite results for a specific merge, open the PR in Github (e.g. https://github.com/ibi-group/otp-middleware/pull/179), click the “Checks” tab, select “Maven CI”, select the “test” job, and expand the “Build with Maven (run unit tests)” section.

**Automated end-to-end testing** consists of replicating a live deployment of a software system (in this case both the otp-middleware and otp-admin-ui modules) and testing specific workflows from start to finish. The middleware end-to-end tests are contained in the otp-admin-ui codebase and use the Puppeteer library to simulate specific actions such as logging in as a user and updating account preferences, and verify that the actions have the intended outcome.

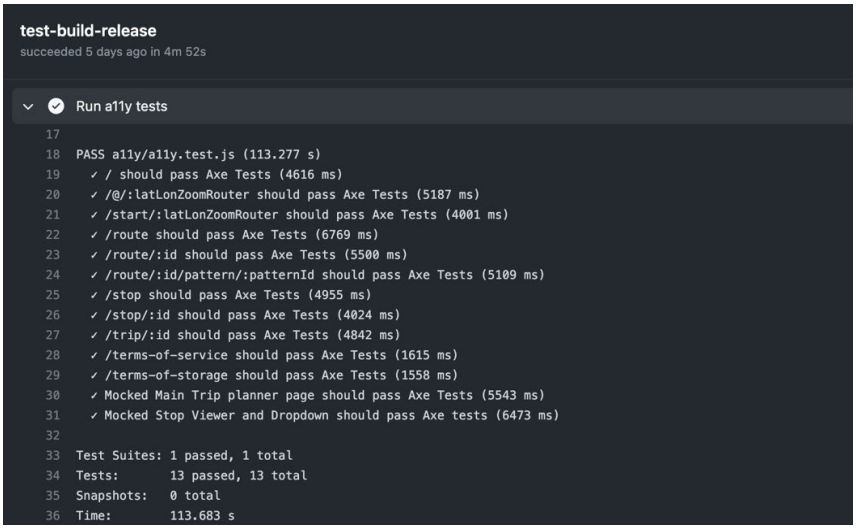
### 3.3.3 Web Application User Interface (Front-End)

The OTP web application user interface (front-end) code consists of two IBI-maintained code repositories: OTP-UI (the UI component library) and OTP-React-Redux (the standalone web application). Between these two repositories, a range of automated testing is performed, including unit testing, accessibility testing, snapshot testing, and pixel testing. A pull request will not be merged until any failing tests have been reviewed and resolved.

**Automated unit testing** for the web application is completed as part of the CI process. Every time a PR is created for a code change, the jest library runs a variety of pre-written unit tests to ensure core logic of the application remains sound.

To view test results for a merged PR on Github (e.g. https://github.com/opentripplanner/otp-react-redux/pull/816): from the main PR page, click on the “Checks” tab, then on the “Node.js CI” tab, then “test-build-release”, and expand the “Run tests” section.

**Automated accessibility testing** for the web application is completed as part of the continuous integration process. Every time a pull request is created for a code change, the Axe accessibility test suite is run and the results are reported as follows, with any test failures detailed here.




*Source: GDOT, 2023*

**Figure 9. Example of Automated Test Script User Interface**

To view test results for a merged PR on Github (e.g. https://github.com/opentripplanner/otp-react-redux/pull/816): from the main PR page, click on the “Checks” tab, then on the “Node.js CI” tab, then “test-build-release”, and expand the “Run a11y tests” section (see

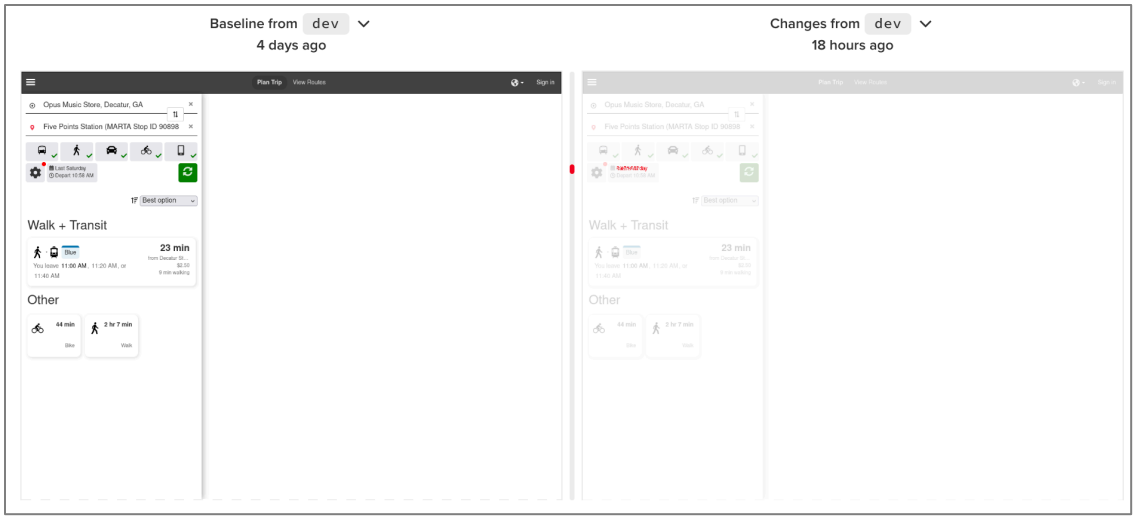
**Figure 9**).

**Automated HTML snapshot testing** for the web application is completed as part of the continuous integration process. Every time a pull request is created for a code change, a library called Storyshot generates HTML for each major component in OTP-RR in a variety of settings (over 400 total). If even one character is incorrect, the test will fail until a developer resolves the issue.

To view test results for a merged *OTP-UI* PR on Github (e.g. https://github.com/opentripplanner/otp-ui/pull/544): from the main PR page, click on the “Checks” tab, then on the “Node.js CI” tab, then “test-build-release”, and expand the “Run tests” section.

**Automated pixel testing** for the web application is completed as part of the continuous integration process. Every time a pull request is created for a code change, node scripts use Puppeteer to build a handful of copies of OTP-RR in different configurations, and then run a series of automated steps in a variety of sizes and user agents. These automated steps interact with almost every aspect of the web app. This includes everything from using the geocoder, to planning a trip, to using the route, trip, and stop viewers.

At predefined moments, a screenshot is taken. If even one pixel is mismatched from the dev branch, the pull request is marked as failing and cannot be merged until the issue is reviewed and resolved as needed.




*Source: GDOT, 2023*

**Figure 10. Example of Percy Test Results UI**

**Figure 10** shows an example of Percy test results. Pixels that changed from the previous build are flagged in red. In this case, the only change is the current date in the trip search form, an expected change than can be safely ignored.

Full pixel test results can be viewed on the Percy platform: https://percy.io/7293c471.

### 3.3.4 Web Application Manual Quality Assurance and Testing Procedures

For each new web application feature, the user interface is validated and inspected for efficacy. The process for manual validation and QA is as following:

*Validation*

- New feature development is assigned in Jira. PRs are reviewed and new features are tested by at least 2 IBI developers before deployment.
- Comments during review are maintained in Jira and GitHub. New features are first tested locally before being deployed to ST-CTN QA environment.
- Once a new feature has been approved, tested and merged, it is pushed to the ST-CTN QA environment.

*Quality Assurance*

- Testing of new features is dependent on the OTP components they affect. New features are tested internally by at least 2 IBI developers and 1 Project Manager.
- Once internal testing is finalized, the Product Owner is notified of new feature availability and can test in QA environment.

- After Release, Alpha testing is initiated and ST-CTN testing staff will test new features and report bugs or issues to IBI before new features are pushed to the Production environment.
- The developer will mitigate any issues related to new features and will follow 'Validation' protocols for any iterative feature development or modifications.
- New feature iterations will be pushed to QA for continued testing/comments from ST-CTN testing staff.

### 3.3.5 Mobile Apps

For the mobile apps, a series of test cases are performed manually for the iOS and Android app versions. These test scripts are done for regression testing and at the end of every development cycle to ensure that the base functionality is retained. The test cases are modified from the developer's base/generic OTP scripts. A full listing of test cases is provided in the appendix.

## 3.4 STM Testing Process

### 3.4.1 Unit Testing

Unit tests are an important part of regression testing to ensure that the code still functions as expected after making changes to the code and helps ensure code stability. After making changes to the code, the team will be able to run the unit tests to ensure that the existing functionality in other parts of the codebase has not been impacted by recent changes.

The STM feature is implemented in the python programming language. The unit test (https://docs.python.org/3/library/unittest.html) framework or similar testing framework will be used to automate STM testing (including aggregation of tests into collections). Testing will apply Golden Path (also called happy path) testing where the control flow follows an expected path or scenario with no exception or error conditions. Since human users do not touch the STM directly, failure paths through the STM will have to be corrected by development on either the STM or calling system side. The team will develop, setup and verify test cases for the "golden path" scenario.

### 3.4.2 Integration Testing

Integration testing will follow the unit testing. Integration testing ensures that related units and existing codebase are correctly connected. This reduces the need for manual testing, and tests more of the system as well as its integration between components in the system. Python has a widely-used test framework, pytest, that is used for integration testing. Pytest utilizes fixtures which are functions executed before each test function. Fixtures feed test data such as databases connections, URLs to test, and input data. The STM will be able to test inputs and to the STM and expected outputs using pytest or a similar integration testing framework.

# 4 Bench and Configuration Testing

Bench testing for hardware and COTS components are considered part of the in-take / installation process (see CIP). The components that use bench and configuration testing methods are acquired, procured, and existing devices or systems that are treated as a trusted IT solution that has already been unit and integration tested. Testing of these systems and components consists of verifying that the system or device operates as expected out-of-the-box or once configured. Because the ST-CTN includes a variety of acquired or existing systems, the type of inspections will differ based on its form, fit and function. Example inspections may include:

- Verify proper operations, i.e., SDK can be compiled as directed, batteries are fully charged, hardware operates within specified ranges, etc.
- Match serial numbers and bill of materials, as appropriate.
- Validate hardware / COTS are configured as directed.
- Validate hardware / COTS are installed as directed.
- Verify hardware configuration / COTS are tuned and optimized.
- Verify hardware / COTS can interface (communicate) with the system's back-office SaaS component.
- Verify COTS and SaaS can communicate with ST-CTN subsystem / components.

The specific inspection depends on the type of equipment and vendor specific systems. High level inspection actions for each major integration element are described in the CIP. The following sections provide a high-level overview of the bench and configuration testing for the ST-CTN components. Note that the integration of these components with the G-MAP, STM or the Private PMD are associated with the Agile Test Process where they will be integrated, coded and tested as part of the Agile processes. In addition, end-to-end or full system testing is conducted as part of CFT and User Acceptance Testing.

## 4.1 Indoor Navigation System

The INS is composed of several components:

- Back-office mapping tools (Mapping Tools)
- Back-office monitoring tools (Inventory and Monitoring Tools)
- iBeacons and configuration tool (Beacons)
- SDK for iOS and Android (SDK)

### 4.1.1 Mapping Tools

Initially, the vendor will ingest a Computer Aided Design (CAD) drawing of the facility. Inspection of the digital facility maps will be conducted to verify its accuracy.

Project and Gwinnett County staff will review the electronic version for accuracy and completeness. Any variances will be coordinated and reconciled with the vendor.

### 4.1.2 Inventory and Monitoring Tools

The inventory and monitoring tools will be used once the iBeacons are in place. During the intake and installation, the installers will document the location of each beacon in the facility.

Project and Gwinnett County staff will verify inventory information matches the real-world placement of the devices.

Once operating, the project and Gwinnett County staff will verify that the monitoring tool output against the real-world information is consistent and accurate. Note that this operating information is not wayfinding information, rather specific system performance. Further details will be included in the CIP.

### 4.1.3 Beacons

The beacon types selected are iBeacon v5.1 (i.e., Bluetooth low energy (BLE)). The test and inspection activities will be part of the intake, configuration, installation and commissioning processes.

Intake procedures include entering asset identifiers into the back-office inventory, checking battery charge, general operations, connectivity, etc. As the beacons are being configured for installation, location optimization will be conducted. Installation procedures will be reviewed and inspected by Gwinnett County staff. Finally the signal strength and gaps will be inspected by project staff walking through the facility, checking all public pathways and destinations throughout the building.

### 4.1.4 SDK

The SDK is provided for iOS and Android operating systems. The SDKs will be compiled and verified that they operate per the vendor benchmarks before they are used. Configuration includes entering the appropriate URL and security features that enable users to reach the INS back-office.

The G-MAP developer will perform the compilation and verification of the SDKs.

### 4.1.5 INS Pilot Test

The INS will include a pilot conducted at the G-MAP vendor facility to evaluate how to calibrate the system by assessing sensor positioning and signal strength and to test various handset capabilities. These tests will be replicated in field tests at the two facilities where beacons are deployed. In addition, the pilot tests will support development of installation, calibration and

commissioning procedures that support INS back office (mapping, inventory, and monitoring tools) and on-site beacon tools when tested and deployed in the field.

The pilot test environment will serve to test the INS SDK with the G-MAP Agile process. Unit and integration testing of the Android and iOS platforms with the G-MAP functions (epics that include indoor and outdoor navigation user stories) will apply the unit and integration testing process described in the G-MAP Agile Testing Process (**Section 3.2**). Controlled field testing will be conducted in the facilities during installation, configuration and commissioning activities, and followed by Alpha Testing (see **Section 5.1**).

## 4.2 PED-X

Internal testing of the PED-X system was conducted during its deployment outside of this project. Integration testing of the PED-X interfaces with G-MAP will be conducted as part of the G-MAP sprints and during UAT / end-to-end testing.

Note: An agreement between the GDOT and the signal system vendor is still underway. Additional insight into additional test protocols will be available later in the development.

## 4.3 Ride Request

Internal testing of the Ride Gwinnet Ride Request application was conducted during its deployment outside of this project. Integration testing of the Ride Request interfaces with G-MAP will be conducted as part of the G-MAP sprints and during UAT / end-to-end testing.

Note: An agreement between the GDOT team and Gwinnett's CAD/AVL vendor is still underway. Additional insight in any additional test protocols will be available later in the development.

## 4.4 Public PMD

Using the Power BI tool, the team will configure and quality check data to ingest, analyze and publish performance measures to the public. The team will apply existing quality checking routines or generate new procedures to curate the data. The project team will verify and validate API results coming from the Private PMD for accuracy and quality.

Test procedures will consist of documenting the curation methods used to check, load, and transform the ingested data. Similarly, internal tools for analyzing and visualizing the data will be employed to test the efficacy of the analytics and presentations. All API end-point data will be validated against authoritative data within the Private PMD database.

Finally, a portal will provide open data for download by the Independent Evaluators and the public. End-to-end testing will be conducted as part of Release 3 / Full System Testing to validate and demonstrate the accuracy of the download process. Observation and / or analysis will be conducted to verify that the performance measurements are accurate.

# 5 Controlled Field and User Acceptance Testing

The Controlled Field Testing is conducted at intervals when software is released from the Agile process. When it is released, all currently integrated and installed systems will also be tested; hence, Full System Testing occurs at each Agile Development release. Prior to its release, as part of the Agile process, automated integration and regression testing on the current main branch code and COTS / existing systems will be performed. When the version is released into CFT, manual end-to-end testing or **Alpha Testing**, is performed to validate that it meets user expectations. Alpha testing uses structured tests that exercise each requirement.

When all the features and releases are completed and all hardware and COTS are fully tested, volunteers will be recruited to perform **Beta Testing** of the G-MAP and its integration components. In addition, USDOT and the Independent Evaluators will be recruited to access PMD data feeds and services. Note that the PMD functionality and performance will be evolving throughout the project duration, since new data and increased number of users will be available throughout the duration of Phase 3.

## 5.1 Alpha Testing Approach

Alpha testing consists of comprehensive, structured, and end-to-end testing of G-MAP and its integration components including STM, PMD, INS, PED-X, Ride Request, and data integration elements (e.g., GTFS-realtime, GTFS).

Alpha tests will be structured to test every functional, performance and operational requirement. All end-to-end functional and performance tests will be conducted on the various channels provisioned by the requirements including:

- Web platforms: Edge, Chrome, Safari, Firefox
- Mobile platforms: iOS, Android

Various versions of each platform will be tested by virtual and field testers. In addition, the test procedures will be tested by accessibility feature to meet the needs identified to support underserved populations.

The Alpha tests will include step by step instructions that walk the tester through the process of using each function. The test procedure will include the following entries: Action / step, expected result, pass / fail, comment (including description of input or anomaly).

The Test Manager will compile the results and report any failures to the Scrum Master for posting as a Ticket on the Scrum Board, as described in **Section 6**. All failures will be identified

immediately to the Scrum Master. At the end of the Final release, the Test Manager will compile the results, grouped by platform, function, accessibility feature and release. The Test Manager will generate graphics of the test results to be included in the System Test Report Summary and ORP results.

## 5.2 Beta Testing Approach

The purpose of the beta testing is to ensure that the end-to-end systems meet stakeholder expectations. To that end, Beta Testing consists of trained volunteers who are recruited based on their diversity and accessibility needs as described by the user needs (ConOps). During this time, all operational components including the Ride Gwinnett Call Center and Operators will be operational with trained staff.

User training will consist of training on ST-CTN functions and features including the various G-MAP user channels and call center information. In addition, users will be trained on how to report a defect in the G-MAP applications or sidewalk mapping features / conditions.

The Test Manager will compile the results and report any failures to the Scrum Master for posting as a Ticket on the Scrum Board, as described in **Section 6**. All failures will be identified immediately to the Scrum Master. At the end of the Beta Testing period, the Test Manager will compile the results, grouped by platform, function, and accessibility feature. The Test Manager will generate graphics of the test results to be included in the ORP results.

## 5.3 Dependencies

Integration Testing will be performed to verify that operational, functional and performance testing results produce expected outcomes for all subcomponents. Each unit will be designed, coded, and tested during each Agile Sprint test, which will be conducted in three week sprints. Each sprint will develop unit-testing stories that will be tested, bug fixes will be applied if necessary, and updated code will be merged into the main branch. The updated features will be included in new releases (or interim versions) and be made available for integration testing.

Once releases are ready to be moved to controlled field testing, a full system integration test will be conducted. System test cases will be developed to conduct a formal system test that tests end-to-end function across subsystems and integrates external systems. Dependencies occur in the following areas:

- Internal G-MAP dependencies include:
    - Tests on different platforms (web browsers, mobile operating systems, call center)
    - Access to qualify review data sources including GTFS/GTFS-realtime and OSM
- Procured or existing COTS / SaaS integrated with G-MAP including INS and Ride Request which can be tested and deployed. Dependencies include:
    - Hardware deployed in the field

  - SW configuration and acquisition of SDK / API
  - Software SDK / APIs sandbox or pre-production environment to test integration with G-MAP
  - Integration of data sets needed to support navigation or wayfinding (e.g., maps for indoor navigation)

- Developed or existing COTS integrated with G-MAP including PED-X, Private PMD, and STM dependencies include:
  - Agreements with vendors (PED-X only)
  - SW configuration and acquisition of SDK / API
  - Software SDK / APIs sandbox or pre-production environment to test integration with G-MAP

- Internal STM dependencies include:
  - Sidewalk data collection
  - Network Impedance Modeling
  - Access to qualify review data sources including the same data used or produced by G-MAP and other static and real time data traffic and mobility data including:
    - G-MAP Trip Planner API
    - Gwinnett County TCC
    - GDOT CV
    - Gwinnett County Transit Management Center

- Data feed from the Private PMD to the Public PMD. Dependencies include:
  - Integration of performance metrics data feed from the STM (private PMD) to the Public PMD

**Table 6**shows the release phases along with the functionality that will be tested during that release phase. The table also highlights the dependencies that will be integrated at the point of Alpha testing.

**Table 6. System Integration Testing and Dependencies**

| Releases | Release Testing Functions | Dependencies: Applications / Installation / Integration Elements |
|---|---|---|
| Alpha Release 1 | **Alpha Release 1 for G-MAP Release 1 – Core, Mobility Profiles and Live Tracking**<br><br>Epic 2 - PII Management<br><br>Epic 3 - Support Service Options<br><br>Epic 3a - User Accounts/Mobility Preferences<br><br>Epic 3b - Routing Enhancements<br><br>Epic 3c - Notifications<br><br>Epic 4 - Live Tracking<br><br>Epic 16 - Call Center Web App | • Configure dev platform<br><br>• Test core OTP, Middleware and Data layers<br><br>• Ingest and QC GTFS (Gwinnett and Metropolitan Atlanta Rapid Transit Authority (MARTA)<br><br>• Ingest and QC OpenStreetMap data for Metro Atlanta<br><br>• Collect GPS sensitivity data for live track tuning<br><br>• Hold Stakeholder meetings to discuss:<br><br>1. Notifications<br><br>2. Mobility Categories (previously called VRU Categories) for Profile Development – coordinate with STM to establish features used to develop impedances |

| Releases | Release Testing Functions | Dependencies:<br>Applications / Installation / Integration Elements |
|---|---|---|
| INS Bench Testing | Indoor Navigation Bench Testing | • Executed vendor contract<br>• Acquire and on-board beacons / set up asset management system in INS<br>• Establish a lab environment to test beacon sensitivity and configuration<br>• Transform building layout to graph<br>• Beacons will integrate with STM and G-MAP Trip Planner |
| Ride Request System Testing | Ride Gwinnett – Ride Request System Testing | • G-MAP Trip Planner integration with Gwinnett County Transit Management Center. Other dependencies under Ride Gwinnett include functional Bus on-board equipment (OBE) and trained Bus Operators. |
| STM | **S1 Data Ingestion Engine**<br>Collect and QC sidewalk data<br>Generate STM Network<br>Generate Whole Road Network<br>Set up STM AWS environment | • Acquire datasets and establish licensing agreements to generate ingestion modules for external datasets (list from DMP):<br>• OSM, Indoor Pathways, G-DOT NaviGator, Building Pathway Asset and Design Condition, Building Wayfinding Asset and Design Condition, Transit Stop, Transit Vehicle, GTFS (Gwinnett and MARTA), GTFS-RT (Gwinnett and MARTA), basic safety message (BSM), Fixed-Route Ridership, Ride Gwinnett Compliant Log. |

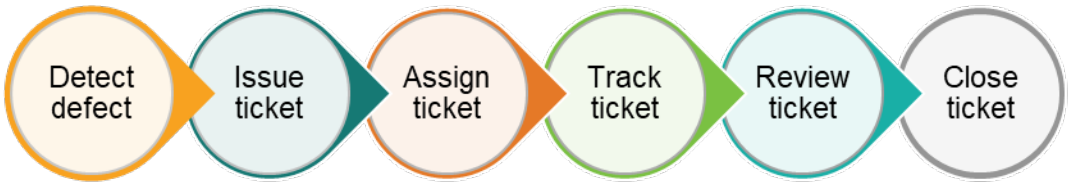
| Releases | Release Testing Functions | Dependencies:<br>Applications / Installation / Integration Elements |
|---|---|---|
| Alpha Release 2 | **Release 2 - Enhanced Routing and Notifications**<br><br>Epic 3b - Routing Enhancements<br><br>Epic 3c - Notifications<br><br>Epic 5 – Ped-X<br><br>Epic 8 - Dynamic Trip Planning<br><br>Epic 17 - Ride Request | • Coordinate Ride Request with Ride Gwinnett (Epic 17)<br><br>• Coordinate PED-X with Kinetics (Epic 5)<br><br>• Integrate STM Network Impedance API (Epic 8) with STM S2 Data Impedance Engine |
| Alpha Release 3 | **Release 3 – Indoor Nav and Group Travel**<br><br>Epic 6 - Third Party Access / Group Travel<br><br>Epic 14 - Indoor Navigation | • Access to INS Sandbox<br><br>• Compilation of INS SDKs (for iOS and Android)<br><br>• Proof of Concept Testing in laboratory environment |

| Releases | Release Testing Functions | Dependencies: Applications / Installation / Integration Elements |
|---|---|---|
| Alpha Release 4 | **Release 4 - Additional Mobile App Features and Integration with External Systems**<br><br>Epic 1 - Survey and Feedback<br><br>Epic 2 - PII Management<br><br>Epic 11 - Trip Histories<br><br>Epic 12 - Completed Trips Logging<br><br>Epic 13 - Travel Statistics<br><br>Epic 15 - G-MAP to PMD | • Coordinate push of Mobile App Logs and trip Feedback APIs with STM (Epics 11 and 12)<br><br>• Coordinate survey instrument configuration with performance management team (Epics 1 and 2) |
| STM | S1 - Data Ingestion Engine | • Data Dependencies: Sidewalk Inventory, ARC Regional Model, GTFS data, Transit Telemetry, CV data, GDOT NaviGator Server, Indoor Pathway Data |
| STM | S2 - Impedance Computation Engine | • Data Dependencies: OSM, OTP, INS |
| Public PDM | Dashboard Design / Prototype | • Integration with Private PMD |
| Public PDM | Additional datasets processed | • Data Dependencies: SidewalkSIM, OSM, OTP, STM |

# 6 Defect/Deficiency Tracking and Resolution Processes

A Defect and Resolution Process was implemented to resolve any defects or discrepancies identified during ST-CTN following a release (during the CFT) or during user acceptance testing. Defects that arise during each sprint will not enter this defect / resolution process; these issues will not proceed to a state of "Done", and hence will not be released into the main code branch. Additionally, defects occurring after the system goes live will be addressed through an incident management process that will be documented in the CMOP. The CMOP incident management process will manage defects, issues, and risks.

The Defect / Resolution process is shown in **Figure 11**. The process is initiated when a defect or deficiency is identified. When reported, the "Master of Scrum Masters" will issue a Jira ticket, assign an individual or team to address the issue, and schedule its resolution in the backlog. As noted in the Agile process, the Jira Issues board will manage all the deficiencies even if they are related to existing or acquired systems and unrelated to the software developed under this project. During the Sprint Planning meeting the ticket will be assigned to an individual or team to resolve. If the resolution requires external support (e.g., vendor), the SEL will manage the issue and report back to the team. The ticket will be tracked on the Jira Issues board and reviewed during each Sprint ceremony until its resolution and closure. The USDOT will provide defect / deficiency tracking and resolution oversight by monitoring the Jira Issues board and will receive documentation of the events as part of the software artifacts.




*Source: ARC, 2021*

**Figure 11. Defect / Resolution Process**

Each stage of the Defect / Resolution process is described as follows:

- **Detect issue:** If the identified issue is low impact and can be handled outside of defect process, then it is resolved and will not be considered a defect. If critical, a ticket will be issued and addressed immediately by the Scrum Master (and assigned to a Scrum Team member for immediate resolution).
- **Issue ticket**: All tickets will be reviewed during the Sprint Planning meetings. If the issue is related to the User Interface / User Experience (UI/UX), then it will be added to the

Product Backlog. If other, it will be added to the Sprint Backlog for scheduling at each Sprint.

- **Assign ticket:** New tickets will be reviewed and assessed during the Sprint Planning meetings (unless identified by Scrum Master as critical). Tickets will be prioritized and assigned to technical staff to identify the cause of the issue and resolution.
- **Track and review ticket:** Once assigned during a Sprint Planning meeting, each ticket (where state is open or in-progress) will be reviewed until resolved or determined that the issue is no longer applicable (and should be abandoned). A ticket may be escalated to a higher criticality if the issue cannot be resolved by revising or refactoring code.
- **Close ticket:** A ticket is closed when the issue is resolved (tested, accepted, and demonstrated).

The priority of the ticket will depend on its categorization and impact on testing. Issues pass through multiple states to track its progress including if the issue becomes increasingly significant. Descriptions of the categories, impacts and issue states are described in the sections below.

## 6.1 Defect / Deficiency Categorization

During detection, the defect will be categorized by type and criticality. Defect types will be assigned to a subsystem if known. The type of defect will also be identified, for example, UI/UX, security, privacy, data, interface, etc. These are used to track and develop metrics on the issue type. The list of categories will evolve as new issues arise.

Defects are also associated with a low, medium, or high criticality level as listed below.

- Low
  - Impacts to user experience and usability, e.g., feedback from users on UI, notifications, turn-by-turn directions
  - Incorrect information due to data input (e.g., GTFS, OpenStreetMap triggered by incorrect directions)
  - Minor failure identified during CFT procedure (e.g., missing flow in user process)
- Medium
  - Impacts to operations or integrated operations that do not impact safety, privacy, or security
  - Impacts to user accessibility when challenges arise in UI/UX
- High
  - Impacts to participant safety or system security that is localized (e.g., at a specific intersection)
  - Persistent defect or anomaly occurring in system
- Critical
  - Impacts that disrupt or shuts down system operations
  - Cybersecurity issue that impacts Personally Identifiable Information (PII)

## 6.2 Defect / Deficiency Impacts on Testing

The defect categorization identifies the criticality and timeliness of the resolution strategy to be taken. For example, a low or medium defect will be cycled into the current or next sprint period. A high or critical designation may require that testing be halted until the issue is mitigated or resolved. If not obvious, the defect will need to be assigned to someone to research the cause and resolution.

During Sprint ceremonies, the scrum masters, product owner and scrum teams will discuss the most appropriate resolution strategy to be taken depending on how the defect affects the loosely coupled subsystems.

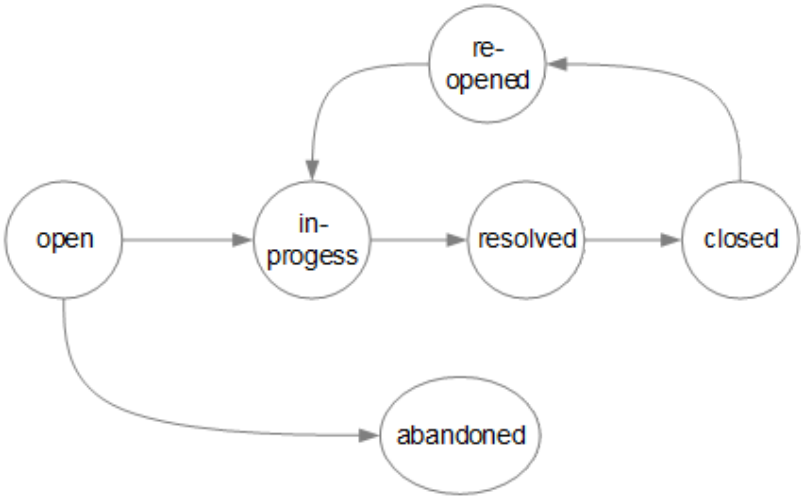
## 6.3 Defect Issue Resolution

As noted in the Defect / Resolution process (**Figure 11**), prior to being resolved, a defect must be tested, accepted, and demonstrated. Testing for Agile developed code will be subject to unit and integration testing processes (including regression testing) identified in **Section 3**, while COTS and existing systems will be subject to verification and validation as described in **Section 4**. Following the Agile process or bench testing, the system will be subject to CFT and then User Acceptance / Operational Readiness demonstrations.

## 6.4 Defect/Deficiency Tracking and Reporting

The Test Manager will review the Defect / Resolution tickets during the Sprint Planning and Review meetings. The “ticket” will be managed as a controlled list of actions at each state. The ticket state flow is depicted in (shown in **Figure 12**). The state descriptions include:

- Open – defect is identified as an issue that must be resolved and is designated a ticket
- In-progress – ticket is assigned to a developer to resolve
- Resolved – a ticket was resolved and is waiting for final testing and approval
- Closed – a ticket received final testing and approval from the technical team
- Abandoned – a ticket was abandoned because it no longer is a problem (code was changed) or due to an external factor
- Re-opened – a closed ticket was reopened because the problem continues to occur




*Source: ARC, 2021*

**Figure 12. Ticket States**

Tickets will be tracked and formed into metrics by the Test Manager. The Test Manager will report to the EMT on the metrics by defect categories, quantity, type, criticality (impact), and state of the ticket.

# 7 Test Cases and Requirements Traceability

The Requirements Traceability Matrix (RTM) documents the verification method and test case for each ST-CTN requirement. The RTM is managed by the Configuration Manager and used to ensure that all requirements are achieved. **Table 7** provides the current requirements to test case traceability and includes the following fields:

**Requirement ID:** a unique identifier for the requirement.

**Component:** the subsystem, application, device or component which is associated with the test case.

**Test Case:** a reference to the test case (and or procedures) that was used to verify and validate the requirement.

**Test Case Description:** a description of the test case that is used to verify the requirement.

**Verification Method:** the manner in which the requirement was verified. Verification methods are defined in **Section 2.4**.

Note: the final version of this document will include a table that includes the master NRTM. In the meantime, a work-in-progress document, an Excel file that contains the RTM with verification method traced to each test case, stores the working copy of the Test Case matrix. **Table 7** provides an example of the requirements to test case traceability.

**Table 7. Requirements to Test Case Traceability Matrix**

| Requirement ID | Component | Test Case | Test Case Description | Verification Method |
|---|---|---|---|---|
| 1.2.0-018<br>0.0.0-237 | Mobile G-MAP | Launch Page | Allow "OTP" to use your location? | Demonstration |
| 7.2.0-206<br>7.2.0-207 | Mobile G-MAP | Launch Page | Share feedback page | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Delete My Account | Delete my account | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Sign Out | Sign Out my account | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Login | Enter valid User Name and valid Password | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Reset Password | Enter valid Email address | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Sign Up | Enter Valid Email and valid password | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Account Settings | Edit, Save and Delete account settings | Demonstration |
| 2.3.0-082<br>2.3.0-083 | Mobile G-MAP | Plan Trip | Checking Accessibility to plan a trip | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Map View | Routes page | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Map View | Plan trip by selecting the modes | Demonstration |
| 2.3.0-082<br>2.3.0-083 | Mobile G-MAP | Map View | Checking Accessibility for Sign In to the account | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Map View | When network is turned OFF, sign in | Demonstration |
| 1.2.0-018 | Mobile G-MAP | Routes | Searching routes | Demonstration |
| 1.2.0-018 | Mobile G-MAP | My Trips | Editing of the trip | Demonstration |

# 8 References

This section includes a list of documents referenced during the plan, including URLs and USDOT Publication Numbers, where possible.

**Table 8. References**

| ID | Referenced Documents |
|---|---|
| [CIP] | Atlanta Regional Commission. Deliverable Task 2 Comprehensive Installation Plan. Atlanta: U.S. Department of Transportation. (2023). |
| [ConOps] | Atlanta Regional Commission. Deliverable Task 2 Concept of Operations. Atlanta: U.S. Department of Transportation. (2021). |
| [CV1K] | Georgia Department of Transportation. "The Regional Connected Vehicle Program Scope of Work." Atlanta: Georgia Department of Transportation. |
| [CVTMP] | AECOM. "Gwinnett County Connected Vehicle Technology Master Plan (CVTMP)." Duluth: Gwinnett County Department of Transportation. (2019). |
| [DMP] | Atlanta Regional Commission. Deliverable Task 3 Data Management Plan. Atlanta: U.S. Department of Transportation. (2021). |
| [GTFS] | GTFS. General Transit Feed Specification Reference. Washington D.C.: GTFS. (2019). |
| [ICD] | Georgia Department of Transportation. Deliverable Task 2B Interface Control Document. Atlanta: U.S. Department of Transportation. (2023). |
| [IPFP] | Atlanta Regional Commission. Deliverable Task 10 Institutional, Partnership, and Financial Plan. Atlanta: U.S. Department of Transportation. (2022). |
| [PMESP] | Atlanta Regional Commission. Deliverable Task 5 Performance Measurement and Evaluation Support Plan. Atlanta: U.S. Department of Transportation. (2021). |
| [SDS] | Software Development Schedule. Deliverable Task 2E Software Development and Integration (incorporated into the Phase 2 Program Management Plan (PMP)). |
| [SySR] | Atlanta Regional Commission. Deliverable Task 6 System Requirements Specification. Atlanta: U.S. Department of Transportation. (2021). |

# Appendix A. Definitions, Acronyms, and Abbreviations

This section provides acronyms, abbreviations, and glossary of terms used throughout the STP.

## Acronyms

API – application programming interface

APN – all-path network

ARC – Atlanta Regional Commission

ATL RIDES – Atlanta Rider Information and Data Evaluation System

AVL – Automatic Vehicle Location

BSM – basic safety message

C2C – center to center

CAD – Computer Aided Dispatch

CDP – Connected Data Platform

CFT – Controlled Field Testing

CI – continuous integration

CMOP – Comprehensive Maintenance and Operations Plan

ConOps – Concept of Operations

COTS – commercial-off-the-shelf

CV – connected vehicle

CV1K – Regional Connected Vehicle Infrastructure Deployment Program

C-V2X – Cellular Vehicle to Everything

CVTMP – Connected Vehicle Technology Master Plan

DPP – Data Privacy Plan

DSRC – Dedicated Short-Range Communication

EMT – Executive Management Team

FTA – Federal Transit Administration

GDOT – Georgia Department of Transportation

G-MAP – Georgia Mobility and Accessible Planner

GOSystems – GO Systems and Solutions

GTFS – General Transit Feed Specification

GTRI – Georgia Tech Research Institute

HW – hardware

ICD – Interface Control Document

ICTDP – Integrated Complete Trip Deployment Plan

IMI – Integrated Mobility Innovation

INS – indoor navigation system

LEP – Limited English Proficiency

MARTA – Metropolitan Atlanta Rapid Transit Authority

MVP – minimum viable product

NRTM – Needs to Requirements Traceability Matrix

ORP – Operational Readiness Plan

ORTP – Operational Readiness Test Plan

OSM – OpenStreetMap

OSS – open-source software

OTP – Open Trip Planner

PED-X – pedestrian signal request

PMD – Performance Measurement Dashboard

PR – pull request

QA – quality assurance

RSU – roadside unit

RTM – Requirements Traceability Matrix

SaaS – software-as-a-service

SAD – System Architecture Document

SCMS – Security Credential Management System

SDD – System Design Document

SDK – Software Development Kit

SDS – Software Development Schedule

SEL – Systems Engineering Lead

SEMP – System Engineering Management Plan

SILCGA – Statewide Independent Living Council of Georgia

SOI – System of Interest

ST-CTN – Safe Trips in a Connected Transportation Network

STM – Space Time Memory

STP – System Test Plan

STRS – System Test Results Summary

SW – software

SyRS – System Requirements

TCC – Traffic Control Center

TMC – Traffic Management Center

TSP – transit signal priority

UAT – User Acceptance Testing

UI – user interface

US – user stories

USDOT – U.S. Department of Transportation

UX – user experience

V&V – verification and validation

WBS – work breakdown structure

U.S. Department of Transportation
ITS Joint Program Office-HOIT
1200 New Jersey Avenue, SE
Washington, DC 20590

Toll-Free “Help Line” 866-367-7487
www.its.dot.gov

FHWA-JPO-23-996



U.S. Department of Transportation