TURNSTILE REQUIREMENTS

Prepared by: GE GRC
Niskayuna, NY
USA

Version: 2.0

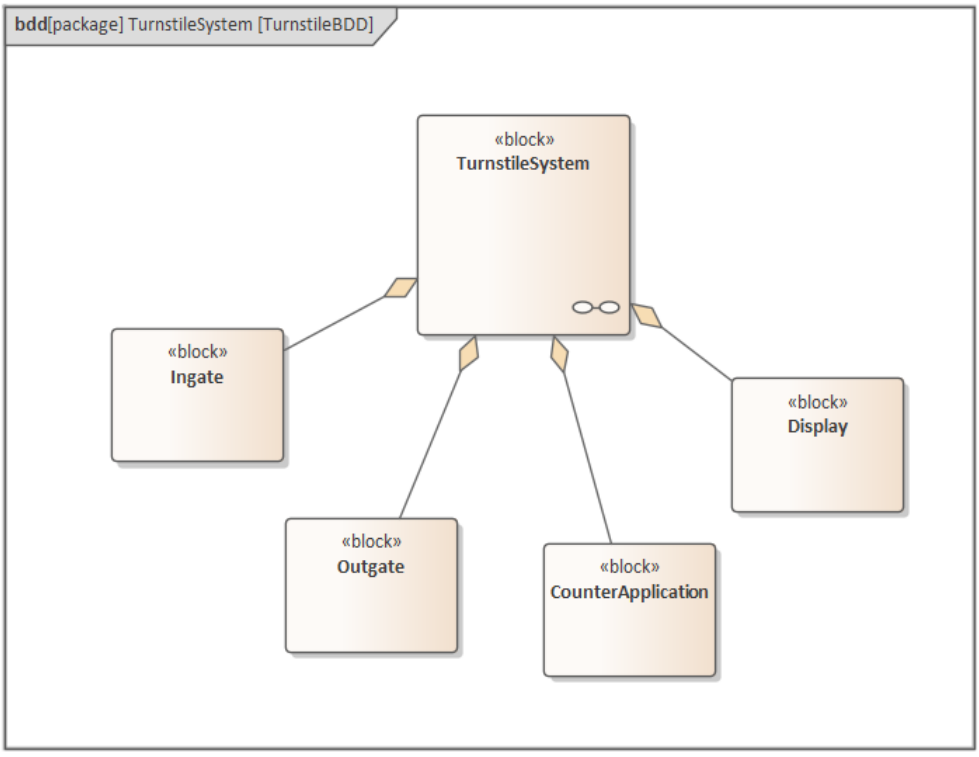
This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-20-C-0203. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).

Revision History

REQ ID	REQ TYPE	REVISION DATE	RELEASE DATE	DESCRIPTION
1: HLR-1	High-level	10/23/2021		
2: HLR-2	High-level	10/23/2021		
3: IN-LLR-2	Low-level	11/08/2021		
4: IN-LLR-3	Low-level	11/08/2021		

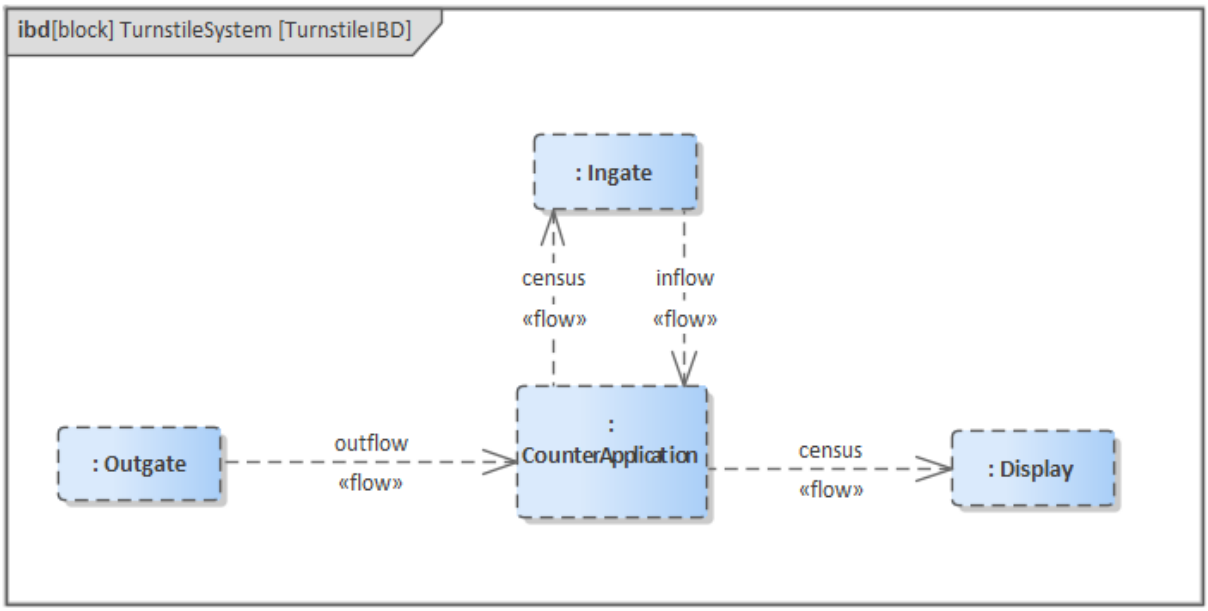
1. INTRODUCTION

The Turnstile example has been expanded to provide a more encompassing view of data that is expected to be loaded into RACK. This expanded version starts with planning documents and provides example data going up and down the development V. While not a complete DO-178C example this expanded version provides a much more representative example of how RACK can be used to curate the multitude of data associated with the development of software. This expanded example includes multiple levels of requirements, data dictionaries, testing, as well as the activities and agents responsible for their development.



This block definition diagram shows the basic components of the Turnstile System:

- Ingate is the physical gate that is used to meter the flow of people into the venue
- Outgate is the physical gate that counts the flow of people leaving the venue
- CounterApplication is the software that manages the running census
- Display is a simple display to show the count of people in the venue



The Turnstile Example provides various example requirements. System Level, High-level Software and Low-Level Software requirements are all represented in the example. In addition, Data Dictionary Terms are included to provide relationships between requirements at a given level. Examples of tracing to higher level requirement are included. Development activities are identified along with the Agents that performed the activities.

2. SYSTEM REQUIREMENTS

1) Requirement identification: Sys-1
    a. Description: Turnstile system shall track the number of people that travel through the in gate.
    b. Governs: Turnstile
2) Requirement identification: Sys-2
    a. Description: Turnstile system shall track the number of people that travel through the out gate.
    b. Governs: Turnstile
3) Requirement identification: Sys-3

a. Description: Turnstile system shall track the number of people that are currently in the park.
b. Governs: Turnstile

3. HIGH LEVEL REQUIREMENTS

1) Requirement identification: HLR-1
    a. Description: The Computer shall increment the counter when an inflow event is received, and the counter is less than max int.
    b. Satisfies : Sys-1
    c. Mitigates: H-1.2
    d. Governs : Counter Application
    e. Created by : HlrDev2
2) Requirement identification: HLR-2
    a. Description: The Computer shall decrement the counter when an outflow event is received, and the counter is greater than 0.
    b. Satisfies: Sys-2
    c. Mitigates :H-1.1
    d. Governs : Counter Application
    e. Created by : HlrDev2
3) Requirement identification: HLR-3
    a. Description: The Computer shall publish the counter at a 1 htz rate.
    b. Satisfies: Sys-3
    c. Governs : Counter Application
    d. Created by : HlrDev1

4. LOW LEVEL REQUIREMENTS

1) Requirement identification: EXE-LLR-1
    a. Description: Executive shall spawn Input Thread on powerup.
    b. Governs: ExecutiveThread
    c. Created by: LlrDev1
2) Requirement identification: EXE-LLR-2
    a. Description: Executive shall spawn Output Thread on powerup.
    b. Governs: ExecutiveThread
    c. Created by: LlrDev1
3) Requirement identification: EXE-LLR-3
    a. Description: Executive shall print a single '.' character to the console every second when running.
    b. Governs: ExecutiveThread
    c. Created by : LlrDev1
4) Requirement identification: IN-LLR-1
    a. Description : Input Thread shall initialize the park count to 0 on powerup.
    b. Governs: InputThread
    c. Created by: LlrDev1
5) Requirement identification: IN-LLR-2
    a. Description: Input Thread shall check for an incoming UDP message on port 62000.

b. Created by : LlrDev1
c. Governs : InputThread
d. Satisfies: HLR-1:v2, HLR-2:v2

6) Requirement identification: IN-LLR-3
   a. Description : Input Thread shall add the delta value received by the UDP to the park count and send the updated park count to the Output Thread when a valid UDP message is received, and the park count range is not exceed.
   b. Governs: InputThread
   c. Satisfies: HLR-1:v2, HLR-2:v2
   d. Created by: LlrDev1
7) Requirement identification: IN-LLR-4
   a. Description : Input Thread shall limit park count to between 0 and 1500.
   b. Governs: InputThread
   c. Created by: LlrDev1
8) Requirement identification: IN-LLR-5
   a. Description : Input Thread shall print 'Invalid Message' to the console when a invalid UDP message is received.
   b. Governs: InputThread
   c. Created by: LlrDev1
9) Requirement identification: IN-LLR-6
   a. Description : Input Thread shall print 'Limit Exceeded'' to the console when a valid UDP message is received, and the park count range is exceeded.
   b. Governs: InputThread
   c. Created by: LlrDev1
10) Requirement identification: OUT-LLR-1
   a. Description : Output Thread shall initialize the park count to 0 on powerup.
   b. Governs: OutputThread
   c. Created by: LlrDev1
11) Requirement identification: OUT-LLR-2
   a. Description : Output Thread shall broadcast a UDP message on port 62001 with the park count every second.
   b. Satisfies : HLR-3
   c. Governs: OutputThread
   d. Created by: LlrDev1