# SimEvents

## Entities

Entity是可以在queues、servers、gates、switches中传送的离散实体，Entity可以携带数据，这些数据称为attributes。

Entities的根据任务的不同，被赋予不同的意义：

| Context of Sample Application | Entities |
| --- | --- |
| Airport with a queue for runway access | Airplanes waiting for access to runway |
| Communication network | Packets, frames, or messages to transmit |
| Bank of elevators | People traveling in elevators |
| Conveyor belt for assembling parts | Parts to assemble |
| Computer operating system | Computational tasks or jobs |

在traffic merging项目中，entities为CAV和Info。

### Storing Entities

可以保存entities的blocks包括：

Entity Generator、Entity Queue、Multicast Receive Queue、Entity Server、Entity Terminator、Discrete Event Chart、MATLAB Discrete Event System、Entity Replicator、Resource Acquirer、Resource Releaser

这些entities可以允许entities的到达，但是必须输出或者销毁：

Entity Input Switch、Entity Output Switch、Entity Multicast、Entity Gate、Composite Entity Creator、Composite Entity Splitter、Resource Pool

### Entities Types

entitity type是一个标签（identification tag），在generation block中给予type tab中其类别的名字。在模型运行的过程中，实体的类别不会变化，但是其attribute，timeout或者timer数据可能会变得越来越复杂。

entity generator可以产生三种entity type：Anonymous、Structured，Bus object。其中traffic merging项目中，entities都为bus objects。

Bus是一种数据类型，其对象仅指定总线的架构属性。例如，总线对象可以指定元素名称、层次结构、顺序和数据类型。但无法指定总线中信号的值。其类似C语言中的结构体，其定义总线的成员，但不创建总线。

## Data and Role of Entity Attributes

attributes可以为任何simulink支持的数据类型，甚至可以是structure。

在traffic merging项目中，CAV的attribute为：

distance、ExitTime、OptimalFuelConsumption、TravelTime、FuelConsumption、ID、Speed、ArrivalTime、FinalSpeed、FinalTime、Position、Lane、Destination、Acceleration、coe

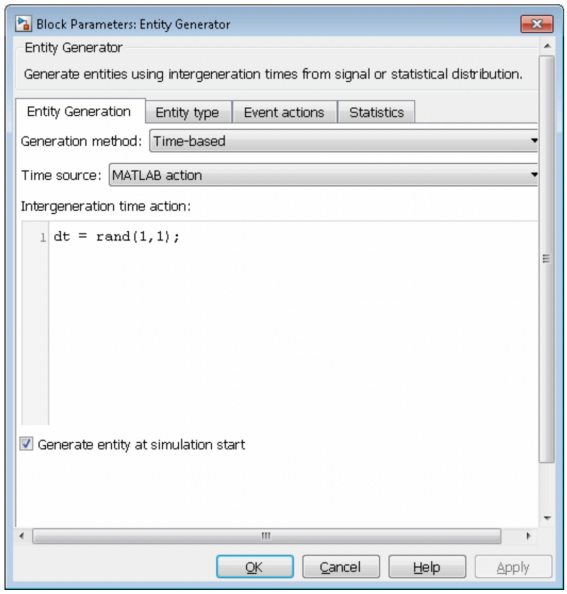
INFO的attribute为：

VehicleID、FinalTime、RandomEvent

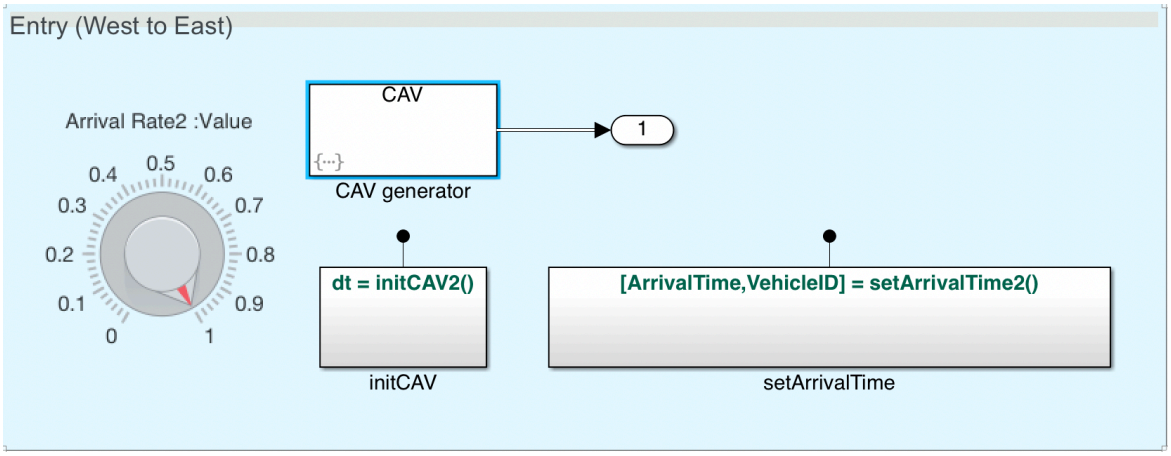
## Create Entities in a SimEvents Model

默认的entity产生方法为time-based，可以通过更改period参数来更改entity的产生时间。

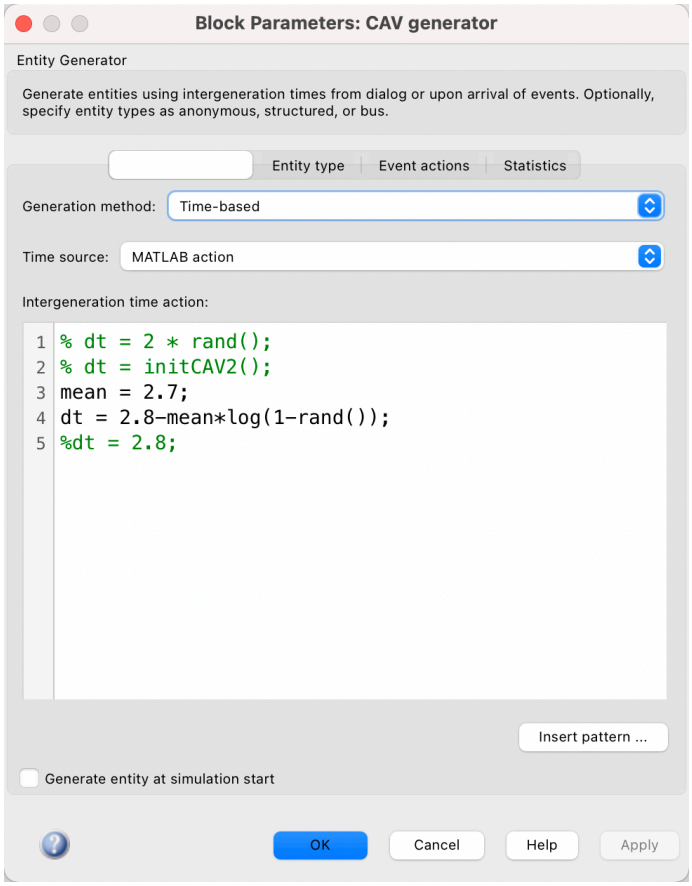
同时，可以使用rand方法来随机产生实体，如下图所示。




对于traffic merging项目，其产生vehicle的block如下图所示：




其车辆的产生服从如下分布：



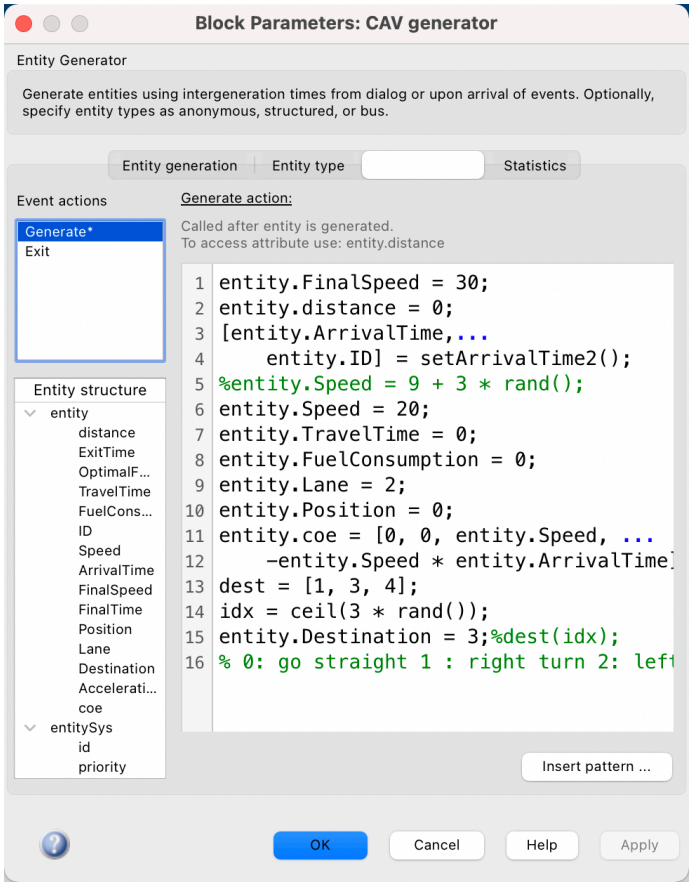

## Events and Event Actions

可以产生事件的条件为：

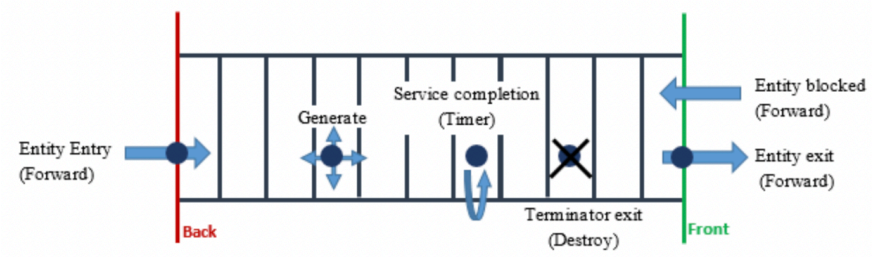
1. An entity is generated.
2. The entity advances from an Entity Generator block to an Entity Server block.
3. The Entity Server block completes the service of an entity.
4. The entity exits Entity Server block and enters an Entity Terminator block.
5. The entity is destroyed.

产生事件动作既可以使用matlab代码，也可以使用simulink中的函数。

在entity generate模块中，产生和退出都可以触发事件。在项目中，作者做了一些entity的初始化操作。




下图展示了entity在队列中可以触发event action的场景。可以更改attribute中的值，但不能改变其名字。




# matlab.DiscreteEventSystem

- Define properties of the object entity types, ports, and storage
  - `getEntityPortsImpl` — Define input ports and output ports of discrete-event system
  - `getEntityStorageImpl` — Define entity storage elements of discrete-event system
  - `getEntityTypesImpl` — Define entity types of discrete-event system
- Event initialization
  - `setupEvents` — Initialize entity generation events
- Runtime behavior of the object
  - `blocked` — Event action when entity forward fails
  - `destroy` — Event action upon entity destruction
  - `entry` — Event action when entity enters storage element
  - `exit` — Event action before entity exit from storage
  - `generate` — Event action upon entity creation

    - `iterate` — Event action when entity iterates
    - `modified` — Event action upon entity modification by the Entity Find block
    - `resourceAcquired` — Specify event actions upon successful resource acquisition.
    - `resourceReleased` — Specify event actions upon successful resource release.
    - `testEntry` — Event action to accept or refuse entity
    - `timer` — Event action when timer completes

While implementing these methods, define entity type, entity storage, create, schedule, and cancel events. Use these functions:

- Define entity type
    - `entityType` — Define entity type
- Define entity storage
    - `queueFIFO` — Define first-in first-out (FIFO) queue storage
    - `queueLIFO` — Define last-in last-out (LIFO) queue storage
    - `queuePriority` — Define priority queue storage
    - `queueSysPriority` — Define system priority queue storage
- Create events
    - `eventGenerate` — Create entity generate event
    - `eventIterate` — Create entity iterate event
    - `eventTimer` — Create entity timer event
    - `eventForward` — Create entity forward event
    - `eventDestroy` — Create entity destroy event
    - `eventTestEntry` — Create an event to indicate that the acceptance policy for the storage has changed and the storage retests arriving entities
    - `eventAcquireResource` — Create a resource-acquiring event
    - `eventReleaseResource` — Create an event to release previously acquired resources(This method allows for partial resource release)
    - `eventReleaseAllResources` — Create an event to release all the resources acquired by an entity
- Cancel events
    - `cancelDestroy` — Cancel previously scheduled entity destroy event
    - `cancelForward` — Cancel entity forward event
    - `cancelGenerate` — Cancel previously scheduled entity generation event
    - `cancelIterate` — Cancel previously scheduled iterate event
    - `cancelTimer` — Cancel previously scheduled timer event
    - `cancelAcquireResource` — Cancel previously scheduled resource acquisition event
- Resource Management
    - `getResourceNamesImpl` — Define resource pools from which the discrete-event system acquires the resources
    - `resourceType` — Specify an entity type and the name of the resources to be acquired by the specified entity
    - `eventAcquireResource` — Create a resource-acquiring event
    - `eventReleaseResource` — Create an event to release previously acquired resources (This method allows for partial resource release)

- `eventReleaseAllResources` — Create an event to release all the resources acquired by an entity
- `cancelAcquireResource` — Cancel previously scheduled resource acquisition event
- `resourceSpecification` — Specify the type and amount of resources for `eventAcquireResource` or `eventReleaseResource` requests
- `initResourceArray` — Initialize a `resourceSpecification` array, required for code generation
- `resourceAcquired` — Specify event actions upon successful resource acquisition
- `resourceReleased` — Specify event actions upon successful resource release

例子：

```
In this example, a custom block allows entities to enter its storage element through
its input port. The storage element is a priority queue that sorts the entities based
on their Diameter attribute in ascending order. Every entity entry to the block's
storage invokes an iteration event to display the diameter and the position of each
entity in the storage.

classdef CustomEntityStorageBlockIteration < matlab.DiscreteEventSystem

    % A custom entity storage block with one input port and one storage element.

    % Nontunable properties
    properties (Nontunable)
        % Capacity
        Capacity = 5;
    end
    % Create the storage element with one input and one storage.
    methods (Access=protected)

        function num = getNumInputsImpl(obj)
            num = 1;
        end

        function num = getNumOutputsImpl(obj)
            num = 0;
        end

        function entityTypes = getEntityTypesImpl(obj)
            entityType1 = obj.entityType('Wheel');
            entityTypes = entityType1;
        end

        function [inputTypes,outputTypes] = getEntityPortsImpl(obj)
            inputTypes = {'Wheel'};
            outputTypes={};

        end

        function [storageSpecs, I, O] = getEntityStorageImpl(obj)
```

```
            storageSpecs = obj.queuePriority('Wheel',obj.Capacity,
'Diameter','ascending');
            I = 1;
            O = [];

        end

    end
    % Entity entry event action
    methods

        function [entity, event] = WheelEntry(obj,storage,entity, source)
            % Entity entry invokes an iterate event.
            % Input argument 1 is the storage index for the iterate event, and '' is
the tag name.
            event = obj.eventIterate(1, '');
        end

        % The itarate event action
        function [entity,event,next] = WheelIterate(obj,storage,entity,tag,cur)
            % Display wheel id, position in the storage, and diameter.
            coder.extrinsic('fprintf');
            fprintf('Wheel id %d, Current position %d, Diameter %d\n', ...
                entity.sys.id, cur.position, entity.data.Diameter);
            if cur.size == cur.position
                fprintf('End of Iteration \n')
            end
            next = true;
            event=[];
        end

    end

end
```