Лабораторная работа

Модель холла метро (Пешеходы)

Цель работы: модель холла Метро (пешеходы), моделирует движение пассажиров в наземном холле метро. Перед тем, как пройти к поездам метро, пассажиры проходят через турникеты, проверяющие наличие билетов. Те пассажиры, которые не купили билеты заранее, должны будут вначале приобрести их в находящейся в холле билетной кассе, и только потом они смогут пройти к поездам. Эта модель продемонстрирует, как промоделировать поток пешеходов и простейшие сервисы в Пешеходной библиотеке AnyLogic.

Содержание работы

1. Моделирование простого пешеходного потока
2. Добавление турникетов
3. Отображение карты плотности
4. Добавление автоматов продажи билетов

Краткие теоретические сведения

Пешеходная библиотека является высокоуровневой библиотекой для моделирования движения пешеходов в физическом пространстве. Она позволяет моделировать здания, в которых движутся пешеходы (станции метро, стадионы, музеи), улицы, парки отдыха и т.д. В моделях, созданных с помощью Пешеходной библиотеки, пешеходы движутся в непрерывном пространстве, реагируя на различные виды препятствий в виде стен и других пешеходов.

Пешеходная библиотека AnyLogic позволяет собирать статистику работы моделируемой системы, и наглядно визуализировать моделируемый процесс с помощью анимации. Вы можете отслеживать плотность пешеходов в различных областях модели для того, чтобы убедиться в том, что система сможет справиться с потенциальным ростом нагрузки, вычислить время пребывания пешеходов в каких-то определенных участках модели, выявить возможные проблемы, которые могут возникнуть при перепланировке интерьера здания, и т.д.

Модели движения пешеходов состоят из двух составляющих – среды и поведения. Под *средой* подразумеваются объекты физической среды - стены, различные области, сервисы, очереди и т.д. Объект среды задается специальным графическим элементом разметки, у которого задаются параметры объекта среды. Ресурсы (сервисы) также являются объектами среды. Поведение пешеходов задается блок-схемой.

Основным объектом библиотеки является пешеход. Пешеход задается с помощью объекта типа Ped. Пешеход “обитает” в заданном физическом пространстве (моделируемой среде) и передвигается согласно заданным правилам. С другой стороны, тип пешехода унаследован от типа агента *Agent*, поэтому пешеходы перемещаются по блок-схеме так же, как агенты.

Блок-схемы пешеходных моделей строятся с помощью объектов, содержащихся в Пешеходной библиотеке. Тип агента *Ped* является базовым типом для моделирования пешеходов. Как всегда, в библиотеке есть объекты для создания пешеходов и управления потоком пешеходов.

Пешеходы создаются объектами *PedSource*, затем они могут быть добавлены в моделируемую среду и направлены далее согласно созданной диаграмме процесса, составленной из блоков Пешеходной библиотеки. Хотя пешеходы движутся в блок-схеме, их движение между блоками блок-схемы определяется моделируемой средой. Например, продолжительность пребывания в объекте *PedGoTo* зависит от скорости пешехода, плотности пешеходов в данной области и других параметров среды.

Вы можете добавлять пешеходов в моделируемую среду и удалять их из нее с помощью объектов *PedEnter* and *PedExit* Пешеходной библиотеки. Обычно пешеходы удаляются объектом *PedSink*.

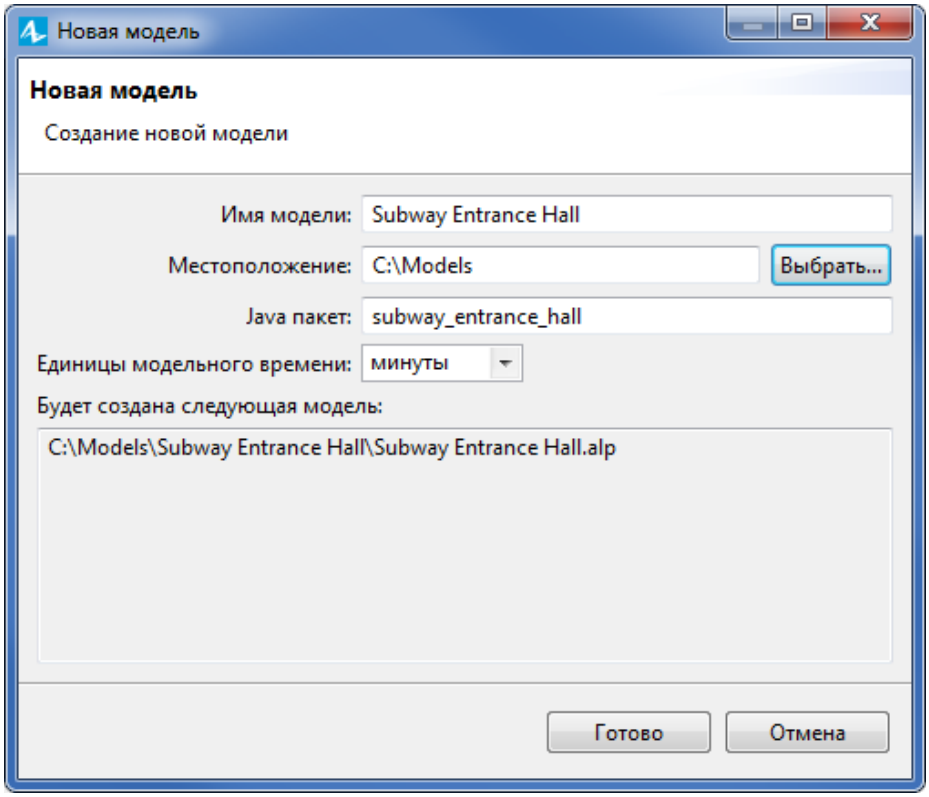
Ход выполнения работы

Шаг 1. Моделирование простого потока пассажиров

Вначале создадим простую модель потока людей, двигающихся внутри здания.

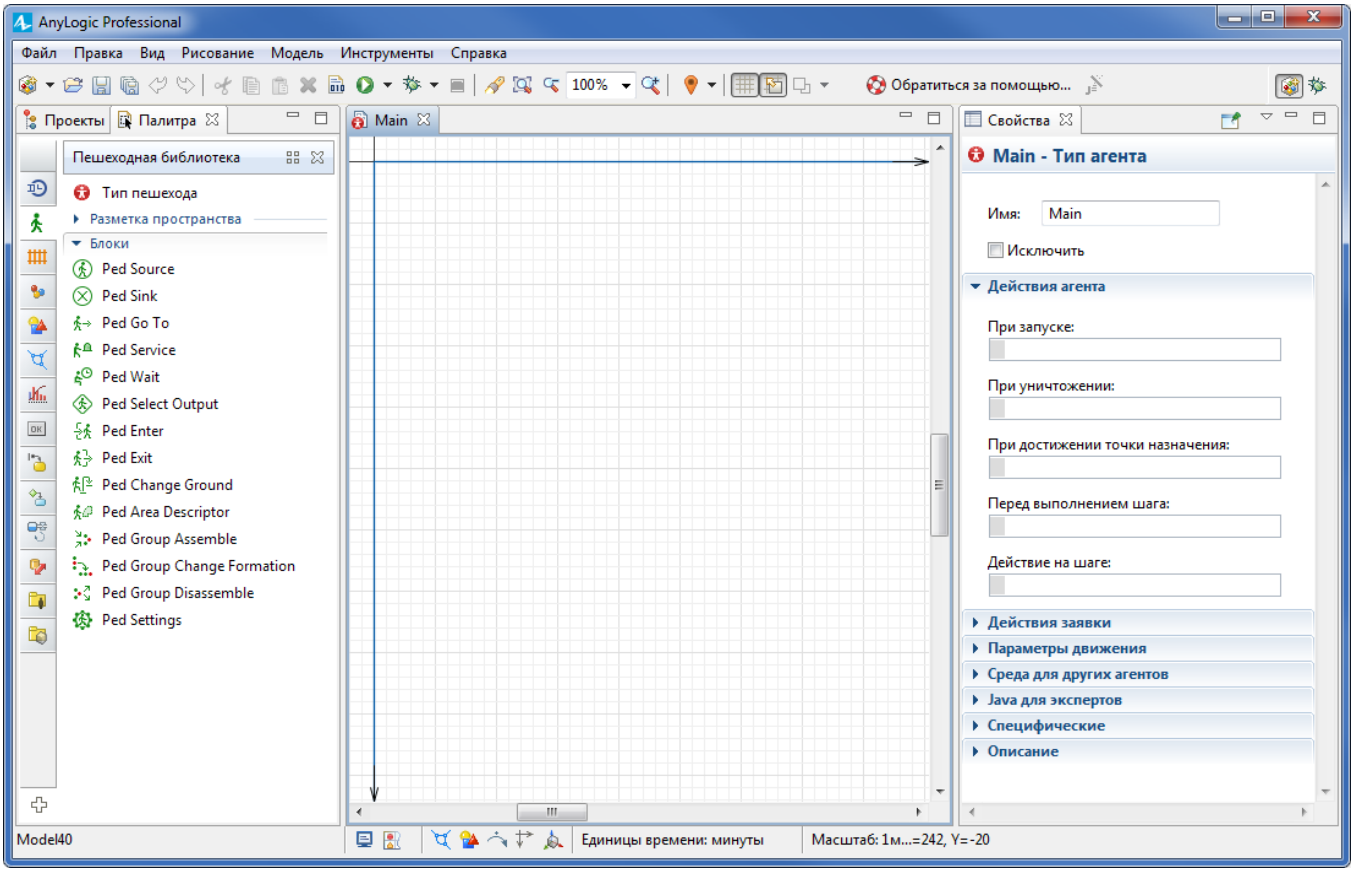
Создайте новую модель

1. Щелкните мышью по кнопке панели инструментов **Создать**  . Появится окно **Мастера создания модели**.
2. Задайте имя новой модели. В поле **Имя модели** введите Subway Entrance Hall.
3. Укажите каталог, в котором будут сохранены файлы модели. Выберите каталог с помощью диалога навигации по файловой системе, открывающегося по нажатию на кнопку **Выбрать**, или введите путь к каталогу в поле **Местоположение**.
4. Выберите **минуты** в качестве **Единиц модельного времени**.
5. Щелкните **Готово**, чтобы закончить создание модели.



Вы создали новую модель. В ней уже имеется один тип агента *Main* и эксперимент *Simulation*. Агенты - это главные строительные блоки модели AnyLogic. В нашем случае агент *Main* послужит местом, где мы зададим всю логику модели: здесь мы расположим чертеж павильона и зададим диаграмму процесса пассажиропотока.

В центре рабочей области находится графический редактор диаграммы типа агента *Main*.



Добавление чертежа моделируемого здания

При создании пешеходной модели вначале обычно добавляется рисунок - план моделируемого пространства (помещения, здания). Затем поверх стен на этом плане рисуются стены (с помощью специальных элементов разметки пространства AnyLogic), и затем создается диаграмма процесса: как пешеходы перемещаются внутри здания.

В этой модели мы будем использовать следующий рисунок - план входа в метро:

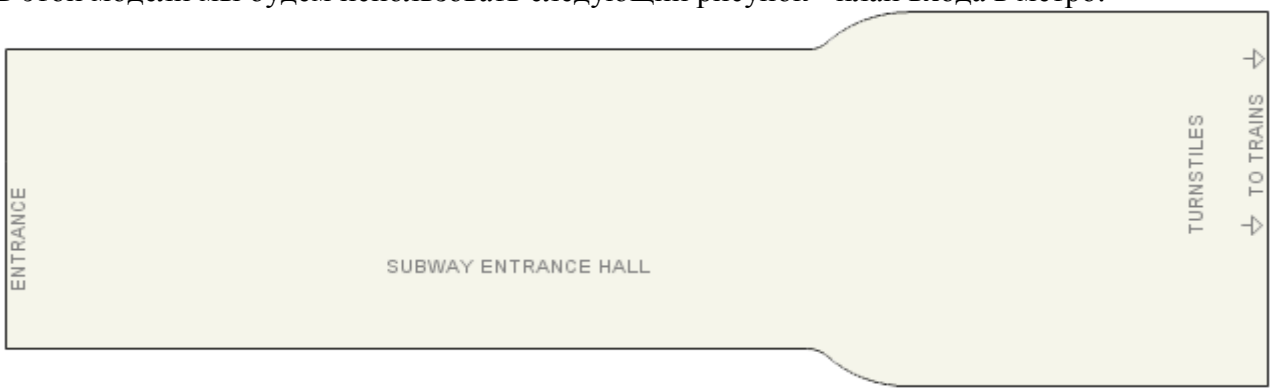
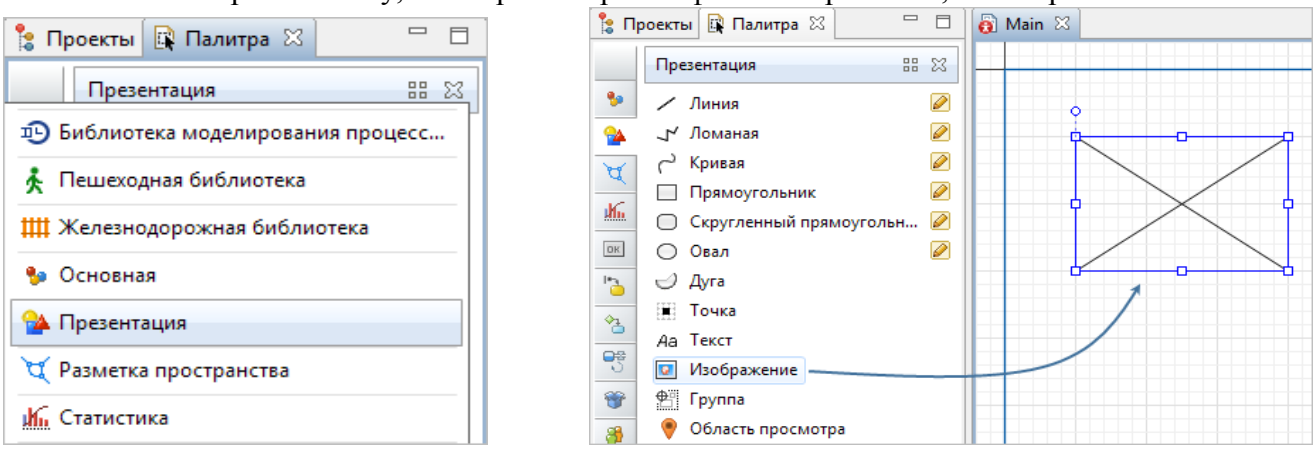


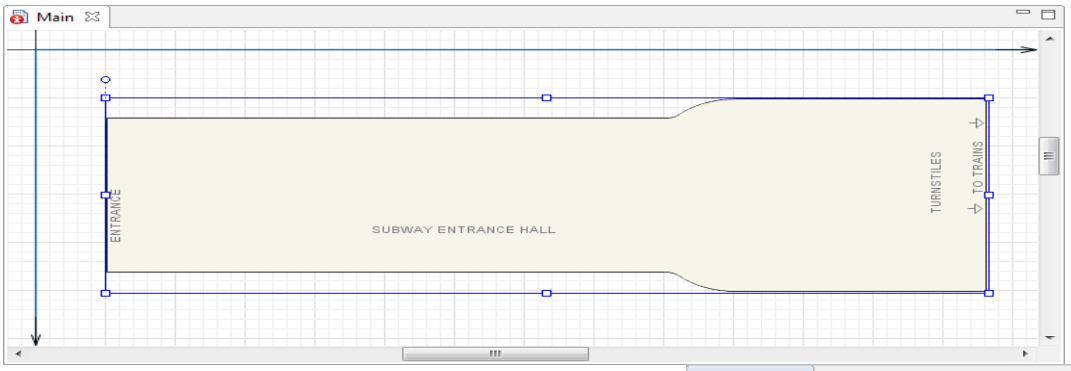
Схема рисунка сохранена в рабочей папке *Материалы для студентов* на Вашем компьютере**.**

Добавьте рисунок с изображением плана холла в модель

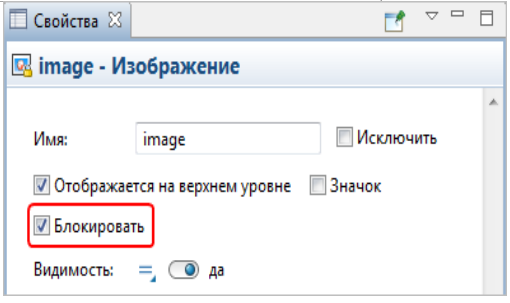
1. Вначале откройте палитру **Презентация**. Эта палитра содержит элементы, которые вы можете использовать для рисования анимации модели.
2. Перетащите элемент **Изображение** из палитры **Презентация** на графическую диаграмму Main.
3. Теперь необходимо выбрать рисунок для отображения. Диалог для выбора файла появится автоматически. Откройте папку, в которой сохранен файл изображения, и выберите его.



4. Изображение будет выглядеть в графическом редакторе следующим образом:



5. Заблокируйте изображение, установив флажок **Блокировать** в панели **Свойства**. Вы не сможете выбрать заблокированную фигуру в графическом редакторе до тех пор, пока вы не снимете с нее блокировку. Мы делаем так потому, что мы будем рисовать другие фигуры поверх этого изображения, и поэтому мы хотим исключить возможность случайного редактирования изображения при рисовании этих фигур.



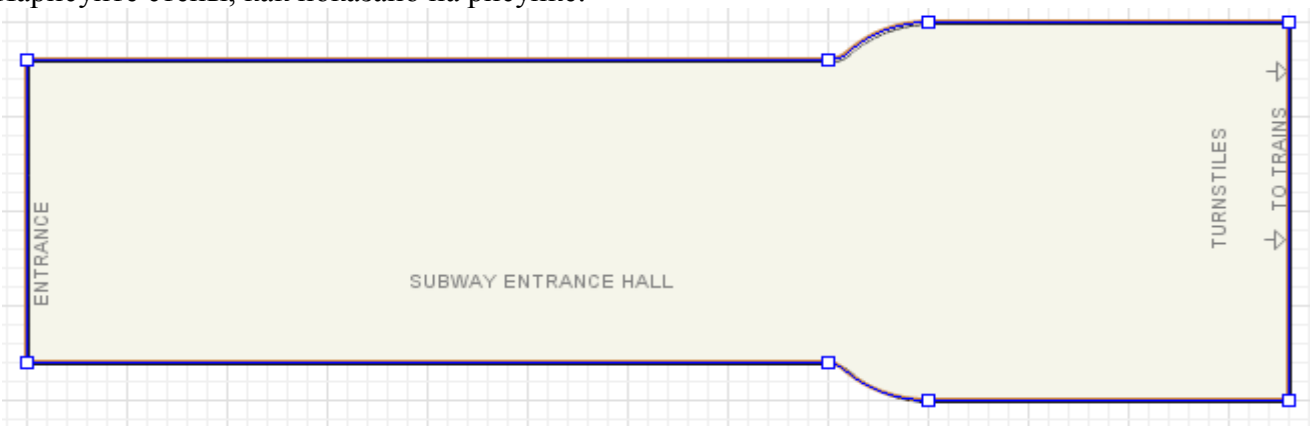
Только некоторые зоны отмечены на этом рисунке. Нам хотелось бы поэкспериментировать с разными диаграммами, и на данный момент нам не известно, где именно располагаются кассы и автоматы по продаже билетов. Поэтому на рисунке отмечены не все области.

Рисование границ здания

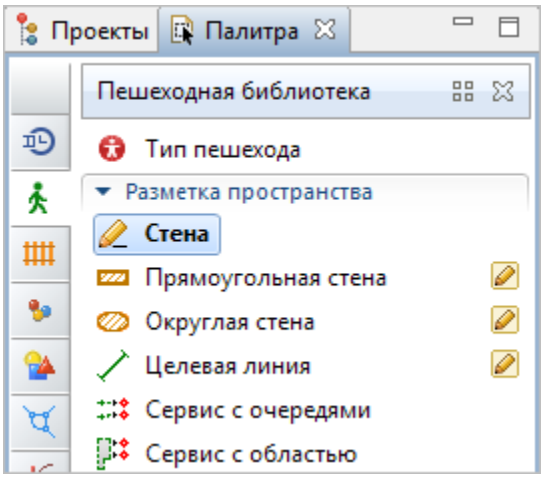
Теперь мы нарисуем на анимации объекты моделируемой среды. Вначале мы нарисуем границу моделируемого нами пространства, играющую роль стен здания.

Нарисуйте границы здания

1. Откройте палитру **Пешеходная библиотека.**
2. Нарисуйте стены, как показано на рисунке:



3. Стены рисуются так. Сначала выберите двойным щелчком мыши элемент **Стена** в разделе **Разметка** палитры **Пешеходная библиотека**. При этом его значок должен поменяться на этот: . Это значит, что режим рисования активен и вы можете рисовать стену в графическом редакторе точка за точкой.



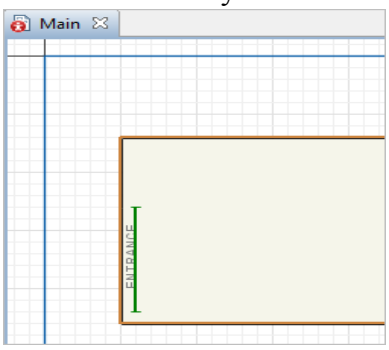
4. Последовательно щелкайте мышью в тех точках диаграммы, куда вы хотите поместить углы стены. Каждый щелчок добавляет часть линии той стены, которую вы рисуете.
6. Чтобы добавить кривую линию, щелкните левой кнопкой мыши в точку конца кривой линии и двигайте указатель мыши, удерживая кнопку. Пока вы ведете указатель мыши, вы заметите, как изменяется радиус кривизны. Чтобы нарисовать окружность, двигайте указателем ровно вдоль сетки координат. Отпустите левую кнопку мыши, когда рисунок готов. Чтобы завершить рисование, добавьте последнюю точку стены двойным щелчком мыши.

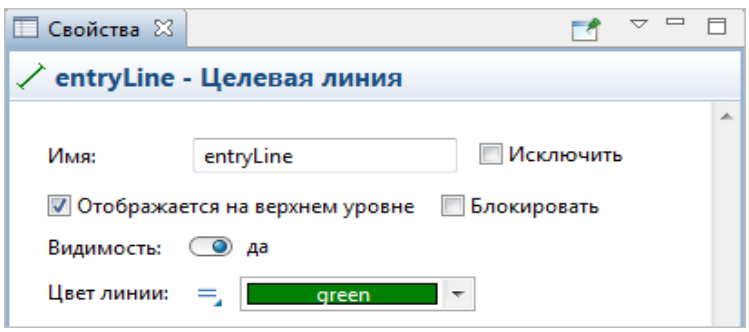
Рисование входа и цели движения для потока пешеходов

Теперь нужно задать области входа и выхода пешеходов. Вначале нарисуем область входа – линию, в которой пешеходы будут появляться. Линия входа для пешеходного потока может быть задана специальным элементом разметки **Целевая линия.** Необходимо нарисовать линию входа точно там, где на рисунке вход обозначен текстом - ENTRANCE:

Нарисуйте линию, где появляются пассажиры

1. Перетащите элемент **Целевая линия** из секции **Разметка** палитры **Пешеходная библиотека** в графический редактор.
2. Измените ее размер и расположите точно так, как показано на рисунке.
3. Назовите линию entryLine.



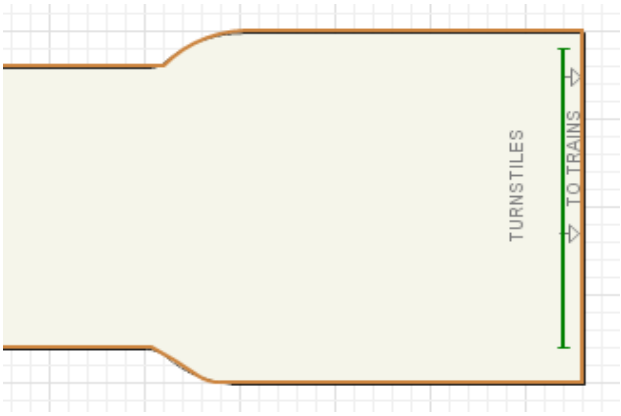


Теперь давайте добавим еще одну целевую линию, которая определяет место, куда движутся пассажиры после того, как они зашли в холл метро.

Мы хотим, чтобы они шли к поездам метро, поэтому давайте расположим эту целевую линию у прохода к поездам - над текстом TO TRAINS.

Нарисуйте целевую линию

1. Нарисуйте еще одну целевую линию и расположите ее так, как показано на рисунке. Зайдя в холл метро, пассажиры будут двигаться сюда, чтобы попасть к поездам метро.



Обратите внимание на то, что все элементы разметки (целевые линии и др.) должны находиться внутри стены.

Создание диаграммы, задающей поток пешеходов

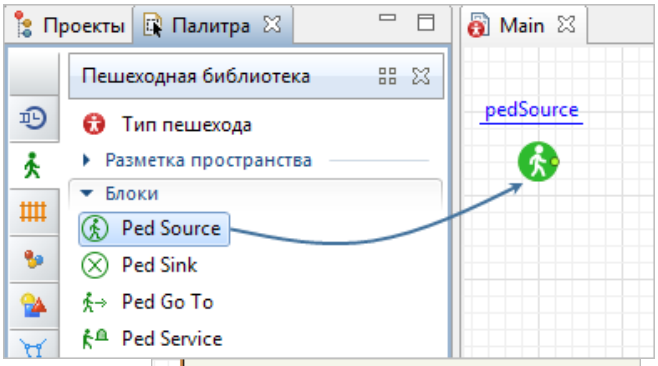
Теперь закончим создание модели, моделирующей простейший пассажиропоток, создав диаграмму моделируемого нами процесса из блоков **Пешеходной библиотеки**.

Начнем с самого простого процесса: пассажиры входят на станцию метро (там, где мы нарисовали entryLine) и затем двигаются в направлении поездов (к targetline).

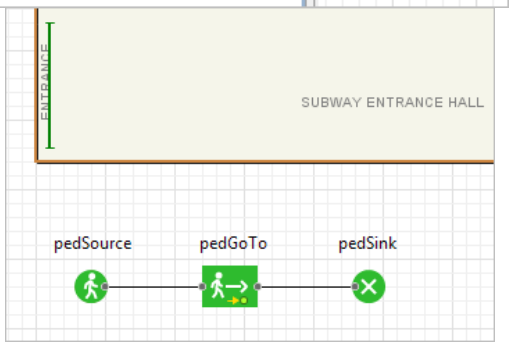
Диаграмма процесса в AnyLogic создается путем добавления объектов библиотеки из палитры на диаграмму типа агентов, соединения их портов и изменения значений свойств объектов в соответствии с требованиями вашей модели.

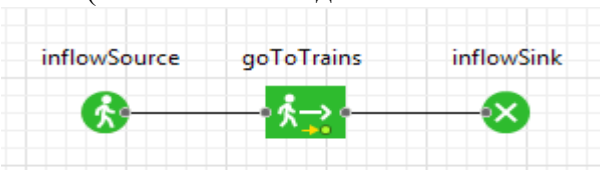
Создайте диаграмму процесса

1. Добавьте объекты **Пешеходной библиотеки** на диаграмму и соедините их так, как показано на рисунке. Чтобы добавить на диаграмму объект **Пешеходной библиотеки**, нужно открыть в панели **Палитра** палитру этой библиотеки, щелкнув мышью по панели с ее заголовком, а затем перетащить нужный вам объект из палитры на диаграмму типа агента.



2. Пока вы перетаскиваете блоки и располагаете их друг рядом с другом, вы можете видеть, как появляются соединительные линии между блоками. Будьте внимательны, эти линии должны соединять только порты, находящиеся с правой или левой стороны иконок. Это очень важно, потому что, например, присоединение порта блока *pedSink* к нижнему порту блока *pedGoTo* вызовет ошибку.
3. Переименуйте блоки. Назовите их *inflowSource*, *goToTrains*, *inflowSink*. (вы можете это сделать в свойствах блока.)



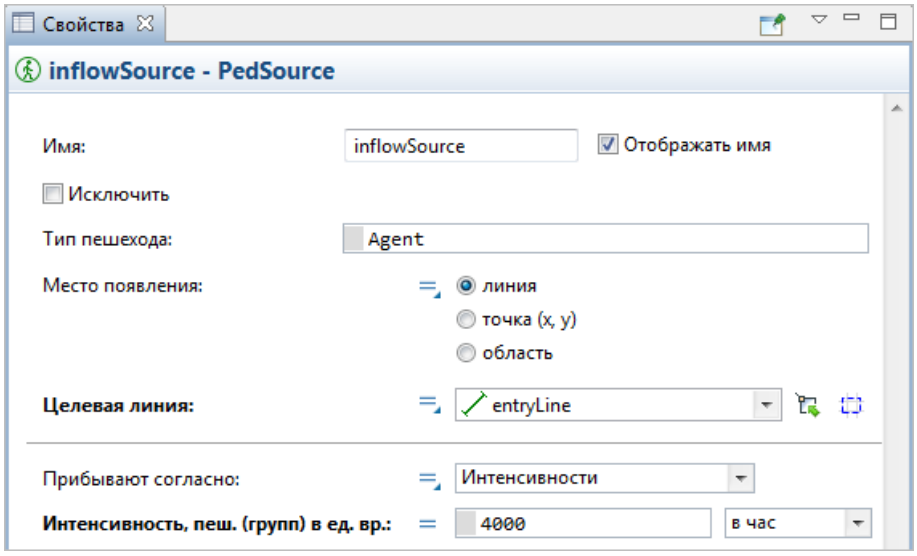


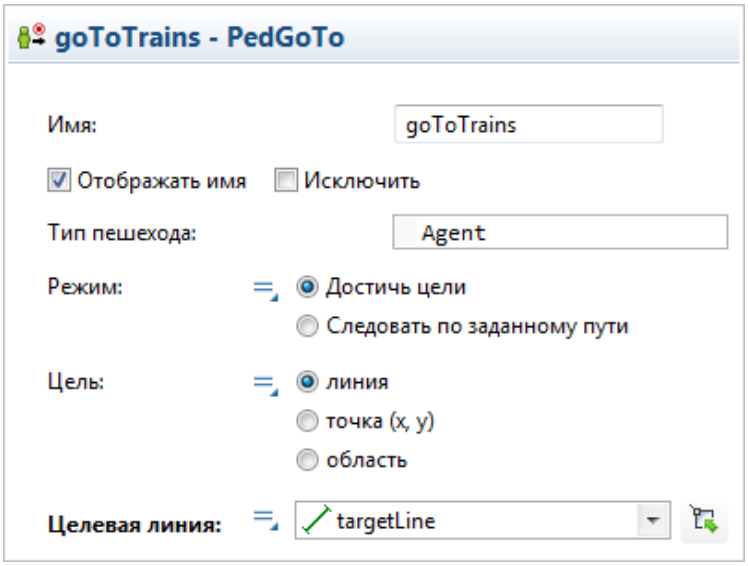
Скажем несколько слов об этих блоках диаграммы.

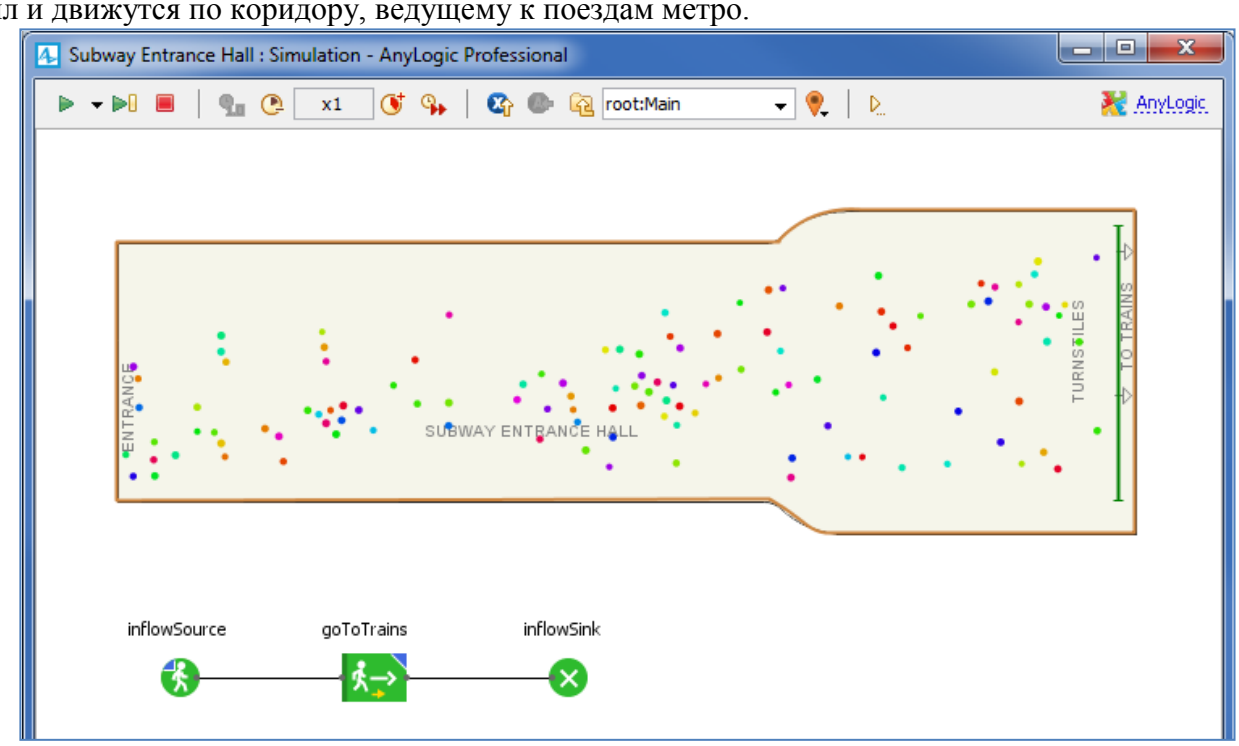
- Объект **PedSource** создает пешеходов. Обычно он используется в качестве начальной точки диаграммы процесса, формализующей поток пешеходов. В нашем примере он моделирует приход пассажиров в павильон.
- Объект **PedGoTo** моделирует перемещение пешеходов из текущего местоположения в другое (заданное параметром этого объекта). С помощью этого объекта мы будем моделировать то, как пассажиры перемещаются от входа в павильон к поездам метро.
- Объект **PedSink** удаляет поступивших в объект пешеходов из моделируемой среды. Обычно объект используется в качестве конечной точки диаграммы процесса.

Измените свойства блоков диаграммы

1. Выделите блок *inflowSource*. В панели **Свойства** задайте место, где появляются пассажиры. Выберите entryLine (название нашей целевой линии, нарисованной ранее у входа) из выпадающего списка **Целевая линия**.
2. Задайте 4000 в час в параметре **Интенсивность**.
3. Теперь измените свойства объекта *goToTrains*. Укажите пункт назначения для пассажиров. После того, как пассажиры войдут в здание, они будут двигаться к той цели, которую вы здесь укажете. На данный момент мы хотим, чтобы пассажиры, вошедшие в павильон, сразу двигались к поездам метро. Укажите targetLine (название целевой линии, которую мы нарисовали второй) в списке **Целевая линия**.
4. Оставьте все свойства объекта **PedSink** установленными по умолчанию.
5. Запустите модель. Модель запустится и вы сможете понаблюдать за динамикой моделируемого процесса с помощью нарисованной вами презентации на диаграмме типа агента *Main*.
Можно увидеть, что пассажиры входят в холл и движутся по коридору, ведущему к поездам метро.







Шаг 2. Моделирование турникетов

На начальном этапе мы промоделировали простой поток пешеходов: пассажиры входят в здание станции метро и движутся через холл к поездам.

Теперь необходимо, чтобы пассажиры проходили через турникеты для проверки билетов до того, как они проходят на платформу отправления поездов. Поэтому нужно добавить турникеты в конце холла.

Моделирование сервисов

Турникеты являются типичным примером использования сервисов в моделях с пешеходами.

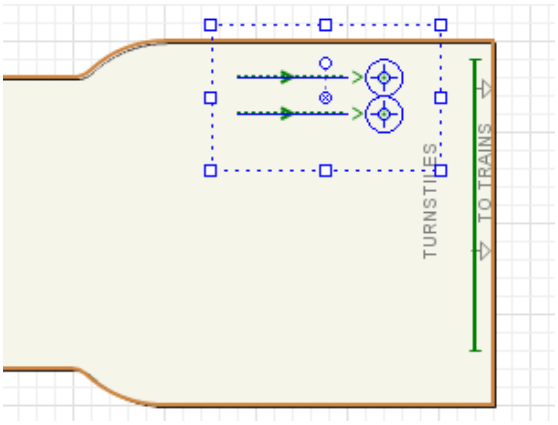
Имеется два типа элементов разметки пространства, которые можно использовать, чтобы добавить сервисы в модель:

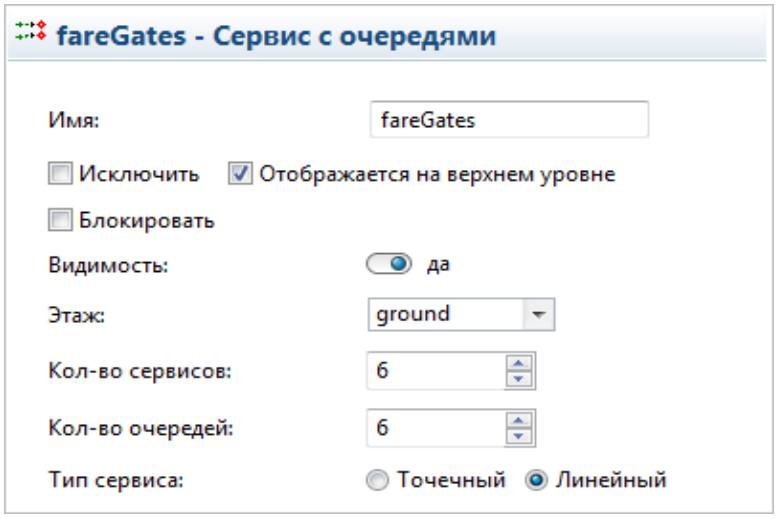
Сервис с очередями - используется для того, чтобы задавать сервисы, в которых пешеходы ждут в очереди, пока сервис не будет доступен.

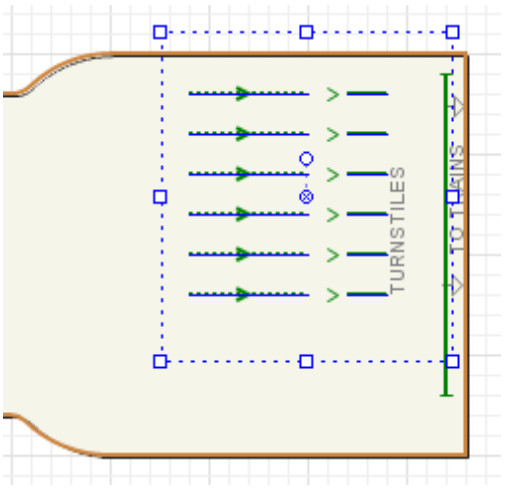
Сервис с областью - используется для того, чтобы задавать сервисы с электронной очередью. В таком случае пешеходы не стоят в очереди, а ждут в расположенной рядом области. Турникеты обычно моделируются элементом **Сервис с очередями.**

Нарисуйте турникеты

1. Перетащите элемент **Сервис с очередями** из раздела **Разметка** палитры **Пешеходная Библиотека** в графический редактор. Вы увидите две точки сервиса и две очереди, ведущие к этим точкам. Разместите эти элементы, как показано на рисунке:
2. Настройте сервисы. Назовите их *fareGates*.
3. Очевидно, что двух турникетов недостаточно. Увеличьте значение параметра **Количество сервисов** до 6. Соответственно, увеличьте значение параметра **Количество очередей** также до 6.
4. Измените значение свойства **Тип сервиса** с **Точечный** на **Линейный**.







Точечные и линейные сервисы

Есть два типа сервисов: *Линейные* и *Точечные*.

– *Линейные* сервисы используются тогда, когда пешеходы должны пройти вдоль заданной линии сервиса, от начальной точки до конечной. Турникеты обычно моделируются линейным сервисом.

– *Точечные* сервисы используются тогда, когда для того, чтобы быть обслуженным, пешеход должен просто подойти к любой точке фигуры, задающей соответствующий сервис, и подождать.

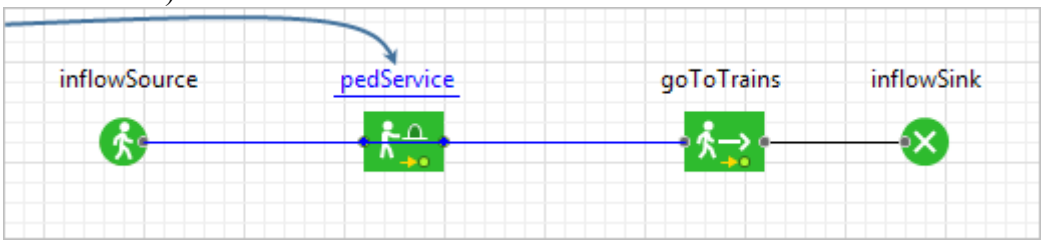
Вы увидите, что сервисные точки стали линиями:

Изменение диаграммы процесса

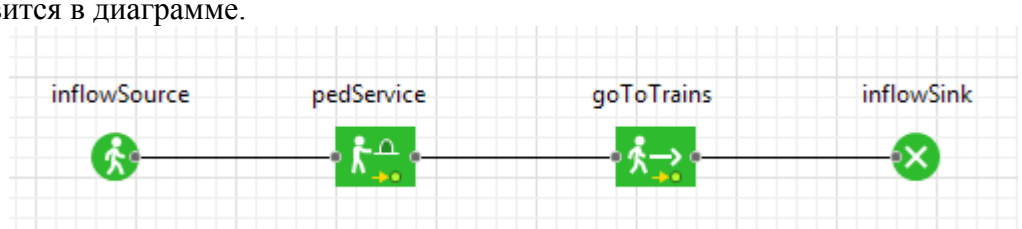
Теперь мы внесем небольшие изменения в диаграмму процесса.

1. Освободите место для нового блока на нашей диаграмме. Сдвиньте блоки *goToTrains* и *inflowSink* вправо, как показано на рисунке:

2. Теперь мы можем вставить блок **PedService** в нашу диаграмму. Блок **PedService** моделирует то, как пешеходы движутся к сервисам, заданным графически элементом разметки и проходят через сервис. Перетащите **блок PedService** из палитры **Пешеходной библиотеки** в графическую диаграмму, поместите сразу за блоком создания пешеходов. Размещая блок посередине соединителя, мы соединяем порты автоматически (но помните, что нужно присоединить правый порт блока *pedService*, а не нижний).



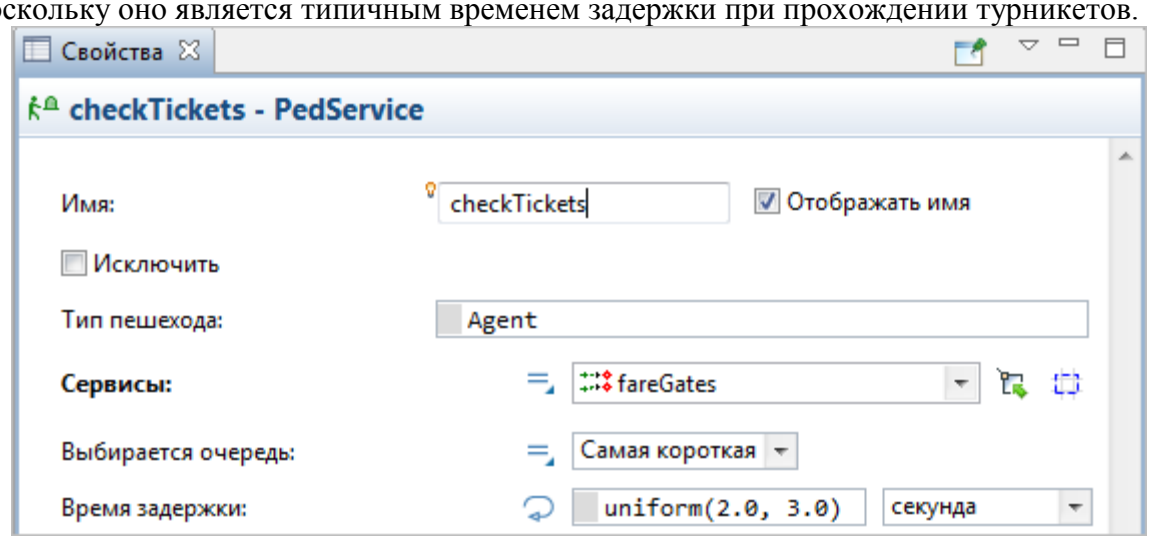
3. Объект появится в диаграмме.



Если вы выделите соединитель щелчком мыши, и его конечные точки в портах будут подсвечиваться светло-зеленым цветом, то это будет означать, что вы успешно соединили порты. Иначе же вам придется проверить, обе ли конечные точки соединителя были помещены точно в порты, и если нет, то передвиньте их туда.

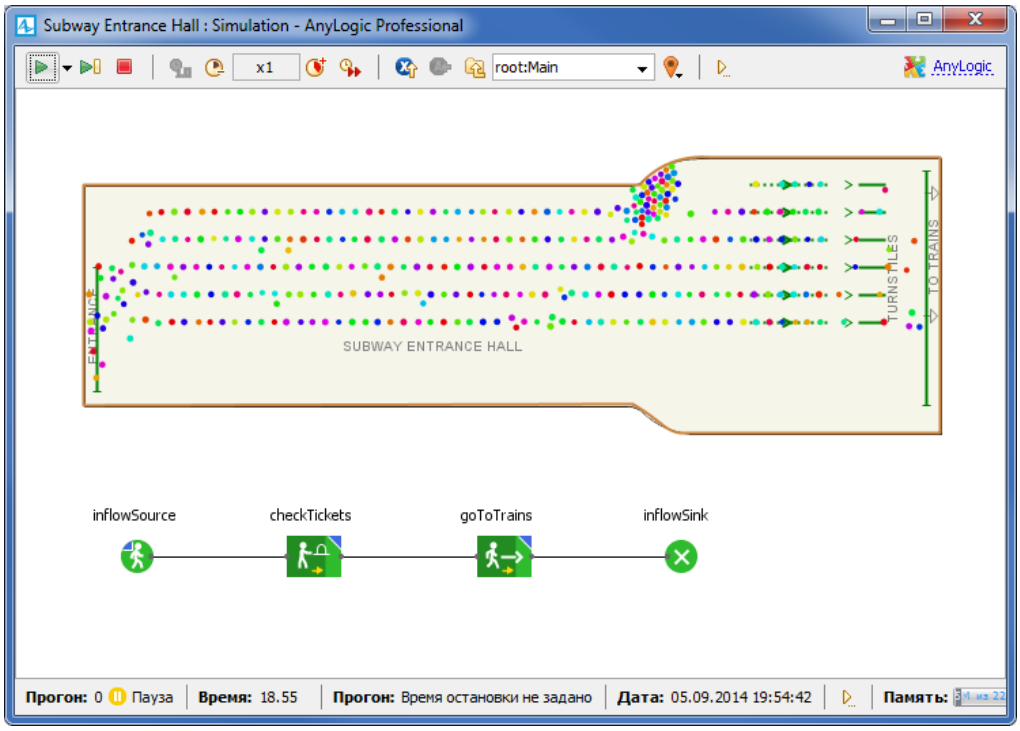
Измените блоки диаграммы

1. Откройте панель **Свойства** блока *pedService*.
2. Измените название блока на *checkTickets*.
3. Выберите fareGates (название нашего элемента разметки **Сервис с очередями**) в поле **Сервисы**.
4. Вы увидите, что заданное **Время задержки** распределено по равномерному закону с минимальным значением, равным 2 секундам и максимальным, равным 3 секундам. Оставьте это значение, поскольку оно является типичным временем задержки при прохождении турникетов.



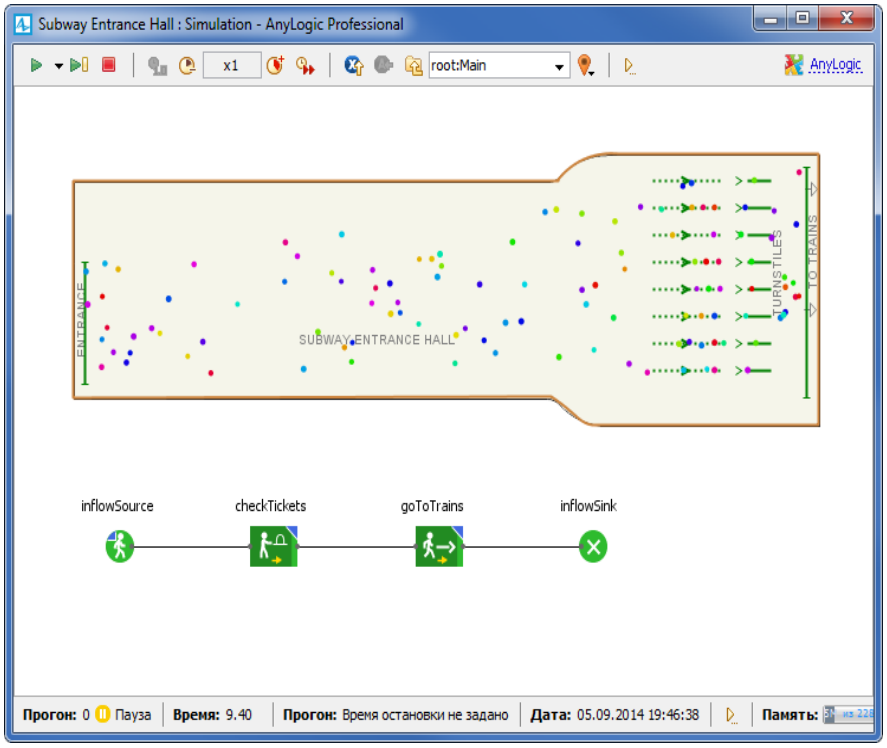
Мы задали новую логику и теперь можем запустить модель и наблюдать за динамикой моделируемого процесса. Скомпилируйте и сохраните изменения в модели.

5. Запустите модель. Вы можете увидеть, что теперь пассажиры проходят через турникеты и перед турникетами быстро образуются длинные очереди. Значит, нам необходимо увеличить количество турникетов.
6. Остановите модель и снова откройте свойства элемента сервиса. Увеличьте количество сервисов и очередей до 7. Снова запустите модель и проследите за положением вещей сейчас. Вам может показаться, что теперь все проходит благополучно.



Но, на самом деле, моделируемый процесс исполняется медленно, и, чтобы увидеть всю динамику, мы рекомендуем вам увеличить скорость исполнения модели, чтобы понять, будут ли расти очереди.

На данный момент, лучше всего переключиться в режим виртуального времени, чтобы модель выполнялась на максимально возможной скорости и вы смогли быстро промоделировать работу системы за долгий период времени.



Итак, очереди все еще растут, не так быстро как раньше, но иногда они могут стать довольно значительными. Семи турникетов до сих пор недостаточно, чтобы обслуживать поток из четырех тысяч пешеходов в час. (Не забывайте, что наши турникеты имеют время задержки, равное двум-трем секундам). Давайте еще раз увеличим количество турникетов. Измените количество сервисов и количество очередей до 8 и снова запустите модель. Наконец наша конфигурация приемлема: станция успешно справляется с таким потоком пешеходов.

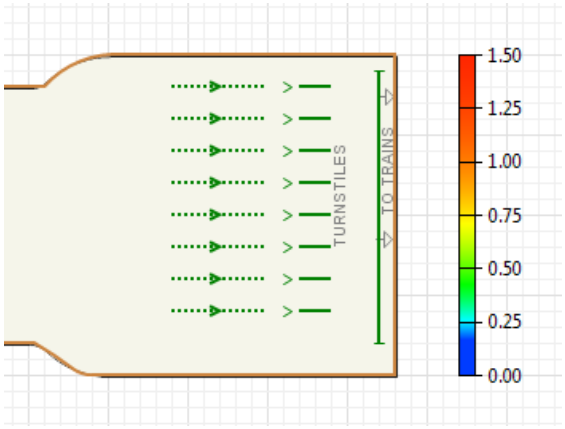
Итак, мы впервые извлекли практическую пользу из нашей модели. Данная модель помогла нам найти требуемое количество точек сервиса. Вы можете изменять интенсивность пассажиропотока в настройках блока **PedSource** и наилучшим образом моделировать средства обслуживания согласно нагрузке.

Шаг 3. Отображение карты плотности пешеходов

Создали модель динамики пешеходного потока в здании холла метро. Теперь нам хотелось бы получить статистические данные этого потока. Самым значительным инструментом в моделировании пешеходов является **Карта плотности пешеходов**.

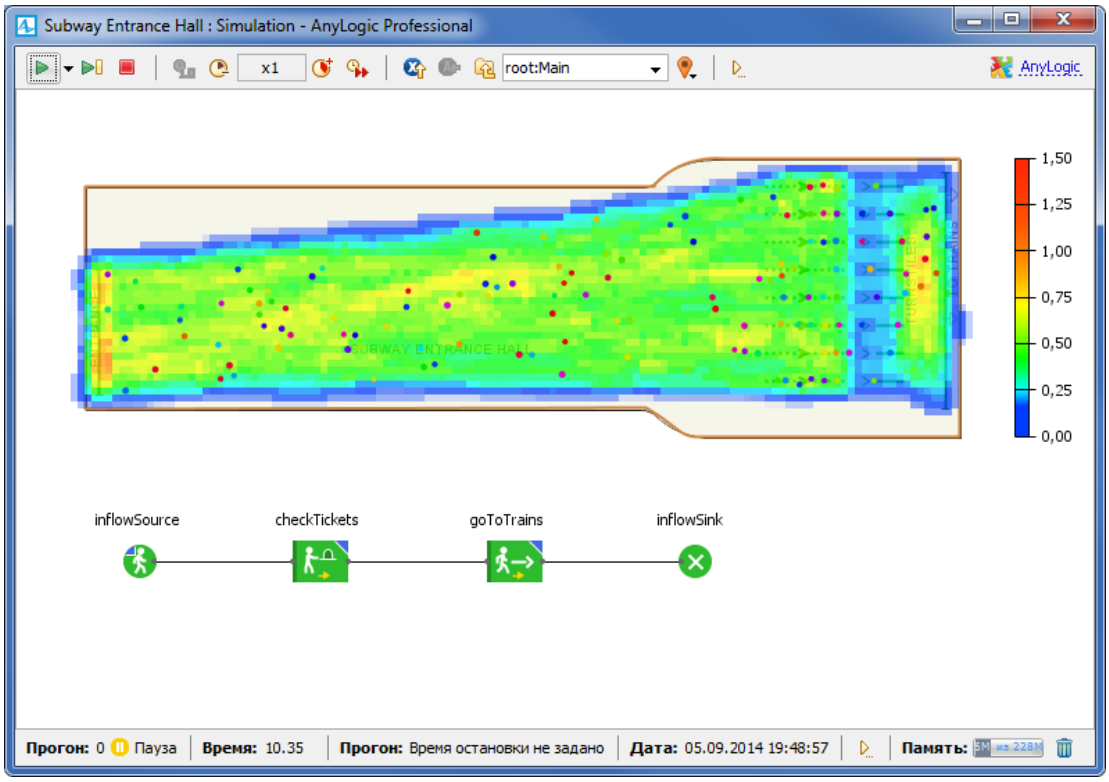
Отобразите карту плотности пешеходов

1. Перетащите элемент **Карта плотности пешеходов**  из секции **Разметка пространства** палитры **Пешеходная библиотека** в графический редактор. В отличие от других объектов библиотек, у этого объекта вместо привычного небольшого значка отображается шкала. На анимации отображается легенда карты. Легенда карты плотности помогает понять, какие цвета соответствуют каким значениям плотности.
2. Откройте панель **Свойства** этого блока и сбросьте флажок **Отображать имя**.
3. Выберите значение *ground* из списка **Этаж**.



Теперь вы можете запустить модель и наблюдать за динамикой моделируемого процесса.

Карта плотности



Видно, что по мере того, как пешеходы двигаются в моделируемом пространстве, план помещений будет постепенно закрашиваться различными цветами. В каждой точке пространства цвет будет соответствовать измеренной в этой точке плотности пешеходов. Карта плотности постоянно перерисовывается в соответствии с актуальными значениями, при изменении плотности на определенном участке цвет динамически перерисовывается другим цветом.

Карта плотности чаще всего используется для обнаружения участков пространства, на которых значение плотности достигает критических значений. Такие области отображаются на карте плотности красным цветом. По умолчанию значение критической плотности задано равным 1,5 пешехода на квадратный метр. Вы можете изменить это значение в свойстве **Критическая плотность (отображается красным), пешеходов/м^2** элемента **Карта плотности пешеходов**. Синий цвет используется для участков низкой плотности. При нулевой плотности закрашивание соответствующего участка не производится вообще. Приведенная на рисунке шкала информирует нас о том, что, например, желтый цвет на карте плотности будет соответствовать плотности 0,75 пешеходов/м ^{2}.

По умолчанию AnyLogic использует логарифмическую цветовую схему. При логарифмической схеме цвет стремительно приближается к "критическому" (красному) только при приближении к зоне критических значений плотности, а при малых значениях остается нейтральным. Вы можете сменить логарифмическую схему на линейную, выбрав **Линейная** в свойстве **Цветовая схема**. В этом случае цвета будут меняться от синего к красному линейно согласно градиенту спектра цветов. При желании вы можете задать и свою собственную цветовую схему любой сложности, выбрав **Другая** в свойстве **Цветовая схема**. В расположенном ниже свойстве **Другой цвет** вы можете написать выражение, которое, в зависимости от значения плотности, будет возвращать тот или иной цвет.

Вы можете заметить, что даже когда в области совсем нет пешеходов, карта плотности все равно отображается. Это обусловлено там, что карта плотности может либо запоминать и отображать цвета, соответствующие максимальным историческим значениям (когда затухание выключено), либо карта будет соответствовать картине, полученной только за недавнее время (когда затухание включено). Затухание включено по умолчанию. Если опция выбрана, и карта плотности будет затухать, то цвет будет постепенно стремиться к текущему значению плотности (не моментально, а с заданным опозданием). Чтобы включить затухание, установите флажок **Включить затухание** в свойствах элемента **Карта плотности пешеходов.**

Если вас не устраивает плотность карты на чертеже тем, что его практически не видно, вы можете увеличить степень прозрачности отображаемой на анимации карты плотности с помощью бегунка **Прозрачность** в свойствах элемента **Карта плотности пешеходов.**

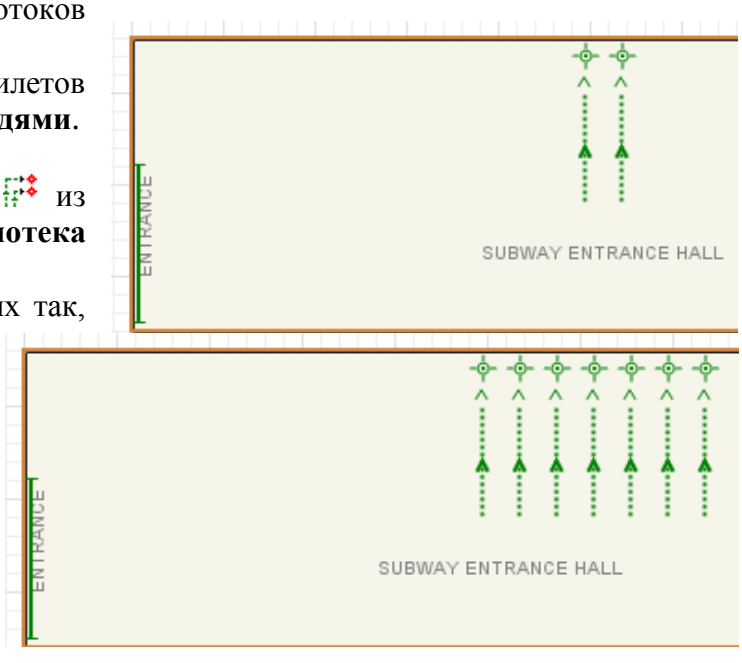
Вы можете отключить отображение карты плотности, но при этом продолжать собирать соответствующую статистику, без отображения карты при анимации модели. Для этого сбросьте флажок **Показывать карту плотности на анимации**. В таком случае вы можете увеличить скорость выполнения модели. Эта статистика может храниться в наборах данных, откуда, например, по окончании моделирования, записываться в базу данных для последующего статистического анализа.

Шаг 4. Добавление автоматов продажи билетов

На данный момент все пассажиры в нашей модели входят в холл метро, затем проходят через турникеты и направляются к поездам. Таким образом, мы предполагаем, что все пассажиры заранее купили себе билеты. Но вряд ли это верно на самом деле. Некоторые люди входят в холл метро уже с билетами в кармане, но большинство людей покупают билеты, лишь когда заходят в станцию метро.

На станции могут находиться различные виды услуг продажи билетов. Небольшие корридоры метро могут быть оборудованы только автоматами по продаже билетов, а большие и просторные станции могут также иметь билетные кассы.

Давайте сначала добавим в нашу модель автоматы продажи билетов. Создавая такую модель, нам необходимо знать количество автоматов, требуемое для того, чтобы успешно обслужить такое количество пассажиров; также, мы сможем найти самое подходящее место расположения автоматов, чтобы минимизировать пересечения потоков пассажиров и образование толп.



Как и турникеты, автоматы продажи билетов логично моделировать элементом **Сервис с очередями.**

Нарисуйте автоматы продажи билетов

1. Перетащите элемент **Сервис с очередями** из секции **Разметка** палитры **Пешеходная Библиотека** в графический редактор.
2. Повращайте элементы сервиса и разместите их так, как показано на рисунке:
3. Откройте страницу свойств сервисов и настройте эту группу сервисов.
4. В этот раз наши сервисы не линейные, а точечные. Точечные сервисы используются тогда, когда для того, чтобы быть обслуженным, пешеход должен просто подойти к любой точке фигуры, задающей соответствующий сервис и провести там

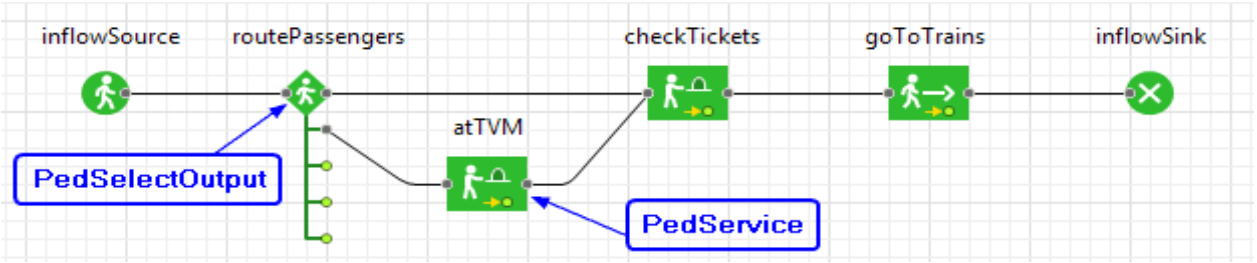
время, заданное как **Время задержки** для этого сервиса. Так что оставьте выбор **Точечный** в свойстве **Тип сервиса**.

5. Назовите сервисы *ticketMachines*.
6. Увеличьте параметр **Количество сервисов** до 7. Соответственно, увеличьте параметр **Количество очередей** так же до 7.

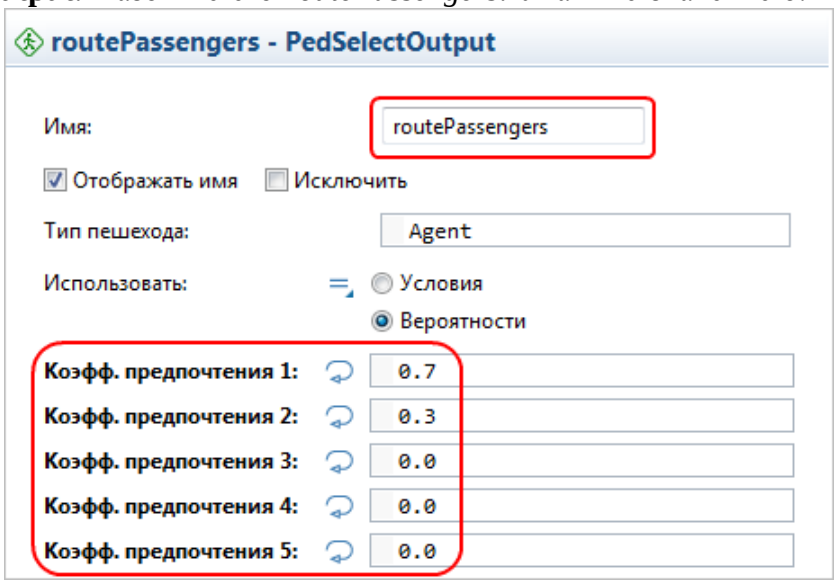
Теперь мы хотим направить некоторых из пассажиров сразу к турникетам, а некоторых - на обслуживание у автоматов продажи билетов.

Измените диаграмму модели

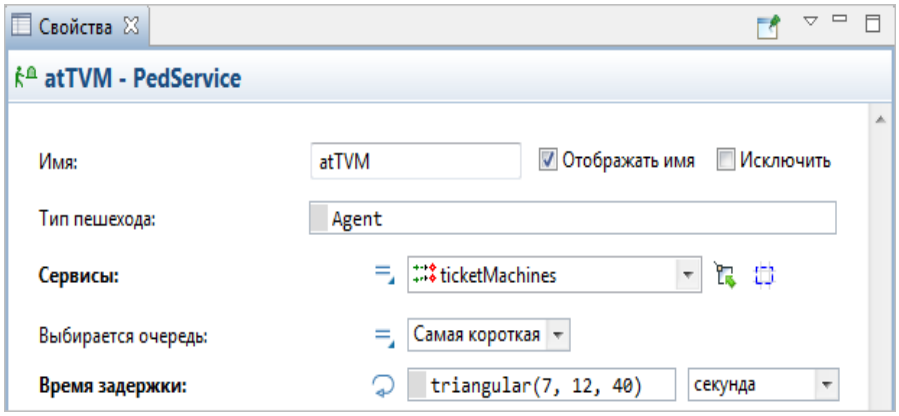
1. Добавьте объект **PedSelectOutput**, чтобы разделить поток пассажиров. Нам нужен этот объект, чтобы перенаправлять пассажиров без билетов к автоматам продажи билетов, а пассажиров с билетами – к турникетам. Объект **PedSelectOutput** является блоком принятия решения Пешеходной библиотеки. Пешеход, вошедший в блок **PedSelectOutput**, будет перенаправляться в один из пяти выходных портов в зависимости от заданных для этих портов коэффициентов предпочтения.
2. Добавьте еще один объект **PedService**. Этот блок будет моделировать обслуживание пассажиров у автоматов продажи билетов. Поместите его между объектом **PedSelectOutput** и ранее созданным объектом **PedService** (*checkTickets*).
3. Соедините блоки, как показано на рисунке.



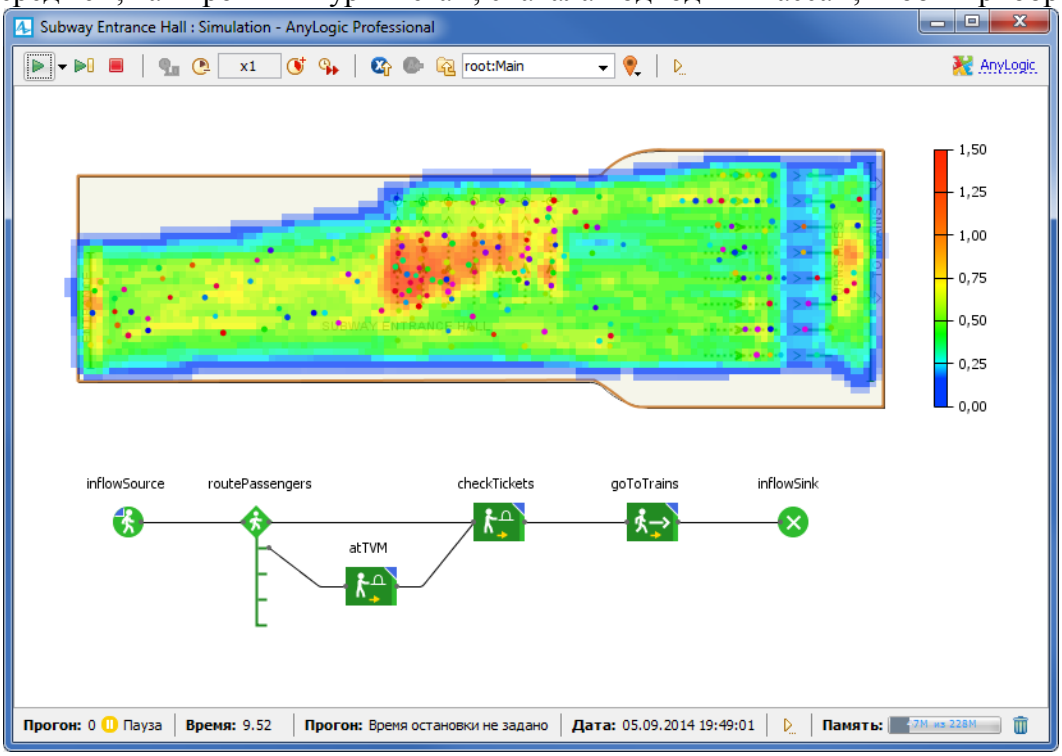
4. Измените свойства объекта **PedSelectOutput**. Назовите его *routePassengers*. Укажите значение *0.7* в поле **Коэфф. предпочтения 1** (коэффициент для потока, направляющегося напрямую к турникетам) и значение *0.3* в поле **Коэфф. предпочтения 2** (коэффициент для потока пассажиров, направляющихся к автоматам продажи билетов соответственно). На этой диаграмме мы допускаем, что количество пассажиров, которые уже купили билеты, значительно выше. Укажите в полях **Коэфф. предпочтения 3, 4, 5** значение, равное *0*.



5. Настройте только что добавленный объект **PedService**. Переименуйте его как *atTVM*.
6. Выберите *ticketMachines* (название нашего элемента разметки **Сервис с очередями**) в свойстве **Сервисы**.
7. Измените параметр **Время задержки**. Введите в поле: triangular(7, 12, 40) и выберите секунды в качестве единиц времени. Мы допускаем, что время обслуживания неравнозначно распределено с минимальным значением 7 секунд, средним 12, и максимальным 40 секунд.



Давайте запустим модель и понаблюдаем за ее динамикой. Вы увидите, что теперь некоторые пассажиры перед тем, как пройти к турникетам, сначала подходят к кассам, чтобы приобрести билет.



Контрольные вопросы:

1. Опишите возможности и назначение Пешеходной библиотеки AnyLogic.
2. Расскажите об основных этапах создания модели холла метро (Пешеходы).
3. Из каких двух основных частей состоит Модель движения пешеходов?
4. С помощью какого типа объекта задается основной элемент модели - Пешеход?
5. Опишите назначение блоков Пешеходной библиотеки:
   - PedSource
   - PedGoTo
   - PedEnter
   - PedExit
   - PedSink
6. Перечислите блоки поведения пешеходов, представленные в диаграмме процесса пассажиропотока, в модели холла метро (Пешеходы).
7. Какие существуют элементы разметки пространства?
8. Какие существуют типы сервисов, дайте им характеристику?