Оглавление

Перечень сокращений

ИМ – Имитационное Моделирование

СДС – Сложная Дискретная Система

IDE - Integrated Development Environment (Интегрированная Среда Разработки)

Терминология

Плагин – независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и/или использования её возможностей

График – диаграмма, изображающая при помощи кривых количественные показатели развития, состояния модели

Подсистема визуализации – подсистема среды RAO-XT, отвечающая за построение графиков изменения состояния модели

Парсинг – процесс анализа последовательности входных данных и преобразование их в необходимых формат

Парсер – компонент, выполняющий парсинг данных

Сериализация – процесс перевода какой-либо структуры данных в последовательность битов

Слайдер – элемент графического интерфейса, позволяющий отображать область окна, соответствующую его положению

Введение

Имитационное моделирование (ИМ) [1] на ЭВМ находит широкое применение при исследовании и управлении сложными дискретными системами (СДС) и процессами, в них протекающими. К таким системам можно отнести экономические и производственные объекты, морские порты, аэропорты, комплексы перекачки нефти и газа, ирригационные системы, программное обеспечение сложных систем управления, вычислительные сети и многие другие. Широкое использование ИМ объясняется тем, что размерность решаемых задач и неформализуемость сложных систем не позволяют использовать строгие методы оптимизации. Эти классы задач определяются тем, что при их решении необходимо одновременно учитывать факторы неопределенности, динамическую взаимную обусловленность текущих решений и последующих событий, комплексную взаимозависимость между управляемыми переменными исследуемой системы, а часто и строго дискретную и четко определенную последовательность интервалов времени. Указанные особенности свойственны всем сложным системам.

Проведение имитационного эксперимента позволяет:

1. Сделать выводы о поведении СДС и ее особенностях:
   - без ее построения, если это проектируемая система;
   - без вмешательства в ее функционирование, если это действующая система, проведение экспериментов над которой или слишком дорого, или небезопасно;
   - без ее разрушения, если цель эксперимента состоит в определении пределов воздействия на систему;
2. Синтезировать и исследовать стратегии управления.
3. Прогнозировать и планировать функционирование системы в будущем.
4. Обучать и тренировать управленческий персонал и т.д.

ИМ является эффективным, но и не лишенным недостатков, методом. Трудности использования ИМ, связаны с обеспечением адекватности описания системы, интерпретацией результатов, обеспечением стохастической

сходимости процесса моделирования, решением проблемы размерности и т.п. К проблемам применения ИМ следует отнести также и большую трудоемкость данного метода.

Интеллектуальное ИМ, характеризующиеся возможностью использования методов искусственного интеллекта и прежде всего знаний, при принятии решений в процессе имитации, при управлении имитационным экспериментом, при реализации интерфейса пользователя, создании информационных банков ИМ, использовании нечетких данных, снимает часть проблем использования ИМ.

1. Предпроектное исследование

1.1. Основные положения языка РДО

Основные положения системы РДО могут быть сформулированы следующим образом [1]:

- Все элементы СДС представлены как ресурсы, описываемые некоторыми параметрами. Ресурсы могут быть разбиты на несколько типов; каждый ресурс определенного типа описывается одними и теми же параметрами;
- Состояние ресурса определяется вектором значений всех его параметров; состояние СДС - значением всех параметров всех ресурсов;
- Процесс, протекающий в СДС, описывается как последовательность целенаправленных действий и нерегулярных событий, изменяющих определенным образом состояние ресурсов; действия ограничены во времени двумя событиями: событиями начала и событиями конца;
- Нерегулярные события описывают изменения состояния СДС, непредсказуемые в рамках продукционной модели системы (влияние внешних по отношению к СДС факторов либо факторов, внутренних по отношению к ресурсам СДС). Моменты наступления нерегулярных событий случайны;
- Действия описываются операциями, которые представляют собой модифицированные продукционные правила, учитывающие временные связи. Операция описывает предусловия, которым должно удовлетворять состояние участвующих в операции ресурсов, и правила изменения состояния ресурсов в начале и в конце соответствующего действия;
- Множество ресурсов R и множество операций O образуют модель СДС;

1.2.Графики в системе имитационного моделирования RAO-X

Имеется возможность визуально отображать процессы, происходящие в модели, в виде графиков. Добавление графиков состояния ресурса становится возможным только после запуска модели. Графики состояния ресурса можно добавлять как в процессе работы модели в режиме анимации, так и после завершения моделирования. График состояния ресурса может быть отображен только в том случае, если конкретный ресурс трассируется. Для добавления графика во вкладке Графики окна объектов в дереве модели следует найти необходимый вам параметр ресурса. Далее, чтобы создать новое окно графика, нужно щелкнуть два раза по выбранному параметру ресурса левой кнопкой мыши или один раз правой и в выпадающем меню выбрать команду «Plot». По оси абсцисс графика откладывается время наступления событий, при которых изменяется значение параметра выбранного ресурса. Как взаимодействуют между собой компоненты подсистемы показано на рис.1.

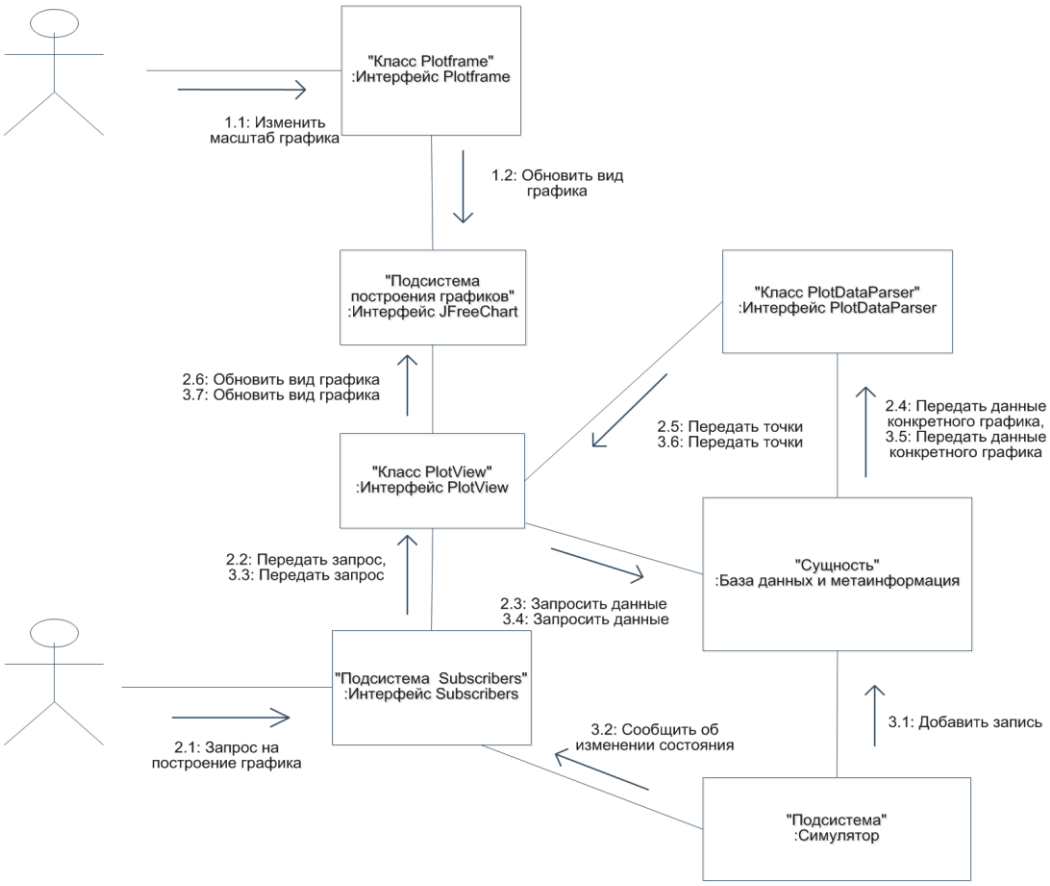


Рис.1 «Диаграмма коммуникаций участников процесса формирования графика»

При первом построении графика, масштаб по оси ординат устанавливается автоматически. Для изменения масштаба по оси абсцисс можно воспользоваться панелью инструментов Масштаб или использовать выпадающее меню при правом щелчке мышки по области графика. Основными функциями масштабирования являются:

- Увеличить масштаб – увеличение масштаба;
- Уменьшить масштаб – уменьшение масштаба;
- Автоматический масштаб – автоматический подбор масштаба по ширине области графика;
- Восстановить масштаб – восстановление начального масштаба;

Текущая подсистема вывода графиков обладает рядом существенных недостатков. Основные операции масштабирования реализованы через контекстное меню, что замедляет работу пользователя. Большой шаг масштабирования и нерабочие слайдеры осложняют восприятие информации с графиков и ограничивают доступ к ней. Также актуальная версия подсистемы не строит графики из одной или совпадающих точек, что может привести к неправильному пониманию результатов построения. Улучшение и доработка подсистемы будет производиться на языке родителя системы ИМ RAO-X- Java [3].

2. Формирование ТЗ

2.1.Общие сведения

Основание для разработки: задание на курсовой проект.

Заказчик: Кафедра «Компьютерные системы автоматизации производства» МГТУ им. Н.Э. Баумана

Разработчик: студент кафедры «Компьютерные системы автоматизации производства» Рахматулин В.В.

Наименование темы разработки: «Доработка подсистемы вывода графиков для системы имитационного моделирования RAO-X»

2.2. Назначение разработки

Доработать подсистему вывода графиков модели для системы имитационного моделирования RAO-X, устранить ошибки.

2.3. Требования к программе или программному изделию

2.3.1. Требования к функциональным характеристикам

Улучшенная подсистема должна обеспечить:

- Возможность прокручивать данные графика с помощью слайдеров после вызова операций масштабирования
- Возможность масштабирования графиков с мелким шагом, используя колесико мыши и клавиатуру.
- Вывод всплывающей подсказки с координатами ближайшей к курсору точки.
- Построение графиков при любом количестве точек

2.3.2. Требования к надежности

Основное требование к надежности направлено на поддержание в исправном и работоспособном состоянии ЭВМ, на которой происходит использование программного комплекса RAO-X.

2.3.3. Условия эксплуатации

- Эксплуатация должна производиться на оборудовании, отвечающем требованиями к составу и параметрам технических средств, и с применением программных средств, отвечающим требованиям к программной совместимости
- Аппаратные средства должны эксплуатироваться в помещениях с выделенной розеточной электросетью 220В ±10%, 50 Гц с защитным заземлением

2.3.4. Требования к составу и параметрам технических средств

Программный продукт должен работать на компьютерах со следующими характеристиками:

- объем ОЗУ не менее 1024 Мб
- микропроцессор с тактовой частотой не менее 1600 МГц
- требуемое свободное место на жестком диске – 4 Гб
- монитор с разрешением от 1366*768 и выше

2.3.5. Требования к информационной и программной совместимости

- операционная система Windows Server 2003 и старше или Ubuntu 15.10 и старше
- наличие в операционной системе ПО Eclipse DSL Tools Mars 2 и новее

2.3.6. Требования к маркировке и упаковке

Требования к маркировке и упаковке не предъявляются

2.3.7. Требования к транспортированию и хранению

Требования к транспортированию и хранению не предъявляются.

2.4. Требования к программной документации

Требования к программной документации не предъявляются.

2.5. Стадии и этапы разработки

Плановый срок начала разработки – 1 октября 2016 г.

Плановый срок окончания разработки – 20 декабря 2016г.

Этапы разработки:

- Концептуальный этап проектирования системы
- Технический этап проектирования системы
- Рабочий этап проектирования системы

2.6. Порядок контроля и приемки

Контроль и приемка работоспособности системы осуществляются с помощью ручного тестирования подсистемы визуализации.

3. Концептуальный этап проектирования подсистемы

3.1.Настройка слайдеров

Полосы прокрутки не появляются при использовании масштабирования через меню. Поэтому необходимо пересчитывать размер слайдеров и включать их после проведения каждой операции масштаба.

3.2. Разработка концепции масштабирования графиков

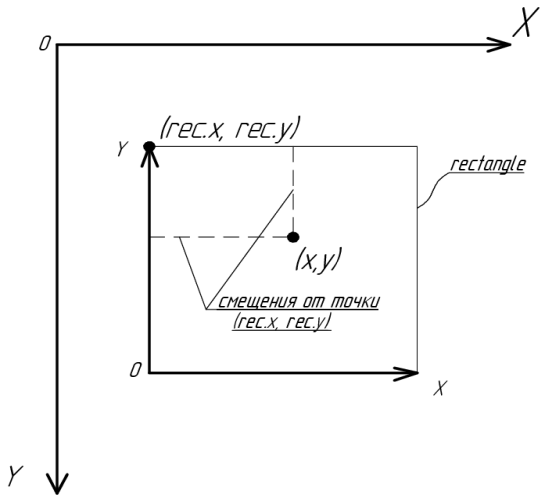
Разработана концепция масштабирования через колесо мышки. Выбор оси масштабирования будет реализован через зажатие клавиш на клавиатуре. По оси абсцисс - с помощью ***Ctrl***, по оси ординат - через ***Shift***. Шаг следует уменьшить, как минимум, вдвое. При одновременном зажатии клавиш - сразу по обеим осям. Клавиши в отпущенном состоянии запрещают увеличение-уменьшение графика.

3.3. Разработка концепции вывода координат точек на графике

При наведении курсора на график необходимо вычислить ближайшую к нему точку. Если курсор достаточно близок хотя бы к одной точке графика, то будет выводиться сообщение в виде окошка с координатами. Выделение будет осуществляться аннотацией в виде выколотого кружка. Цвета линии аннотации такие же как и цвет линий исследуемой кривой. Если мышка покидает поле графика, или расстояние становится большим, то гаснет окно и исчезает выделение.

3.4. Разработка концепции перевода координат график-виджет

Был разработан алгоритм перевода координат точек из графика в систему виджета. Направление и расположение осей приведено на рис.2



«OXY» – система координат виджета.

«oxy»- система координат графика.

(rec.x, rec.y) – координаты левого верхнего угла области внутри осей графика.

(x,y) –координаты точки в системе графика.

Рис.2: «Расположение осей в виджете»

Нужно посчитать смещения от точки rectangle в системе графика, а затем посчитать их размер в системе виджета. Функция необходима, для корректного подсчета расстояний между курсором и точками на графике.

3.5. Разработка концепции вывода графиков с одной или совпадающими точками

Был разработан алгоритм построения графиков с одной точкой. Так как системой построения графиков используется StepRendering (ступенчатая отрисовка), то для устранения проблемы в конце симуляции добавляется одна точка. Это позволит механизму провести прямую. Чтобы отследить необходимость добавления, необходимо создать подписчик, который будет срабатывать при получении сообщения о конце симуляции. При получении сообщения будет проводиться проверка на количество точек на графике.

4. Технический этап проектирования подсистемы

4.1. Доработка слайдеров

Управление слайдерами осуществляется классом ***PlotFrame.***

setSliders*()** – метод, задающий параметры горизонтального и вертикального слайдеров. Внутри метода заданы два анонимных класса, добавляющие слушатели событий ***SelectionListener() к слайдерам. Они позволяют реализовывать прокрутку графиков.

updateSliders() **–** метод, который включает видимость слайдеров и обновляет их размеры в соответствии с масштабом графика. Но только для ***horizontal slider***, поэтому добавим код для ***vertical slider***.

В конце каждой операции масштабирования вызывается метод ***updateSliders().*** Для этого переопределим исходные методы изменения масштаба.

4.2. Доработка системы масштабирования

Созданы внутренние классы в ***PlotFrame***.

Класс ***PlotKeyListener*** [4]**,** реализующий интерфейс ***KeyListener***, отслеживает нажатия на кнопки ***Ctrl***, ***Shift***, и ***Space.*** Методы ***keyPressed()*** и ***keyReleased()*** задают значения флагов. Нажатие на ***Space*** вызывает метод ***restoreAutoBounds()***, возвращающий исходный масштаб у графика.

Класс ***PlotMouseWheelListener***, реализующий интерфейс ***MouseWheelListener***, отслеживает прокрутку колеса мыши. В методе ***mouseWheelScrolled()*** проверяется прокрутка колеса и значение флагов и вызывается подходящий метод масштабирования.

4.3.Разработка метода преобразования координат точек из системы графика в систему виджета

Разработан метод **PlotToSwt (x,y)**, возвращающий координаты в системе виджета. Алгоритм работы приведен на рис.3

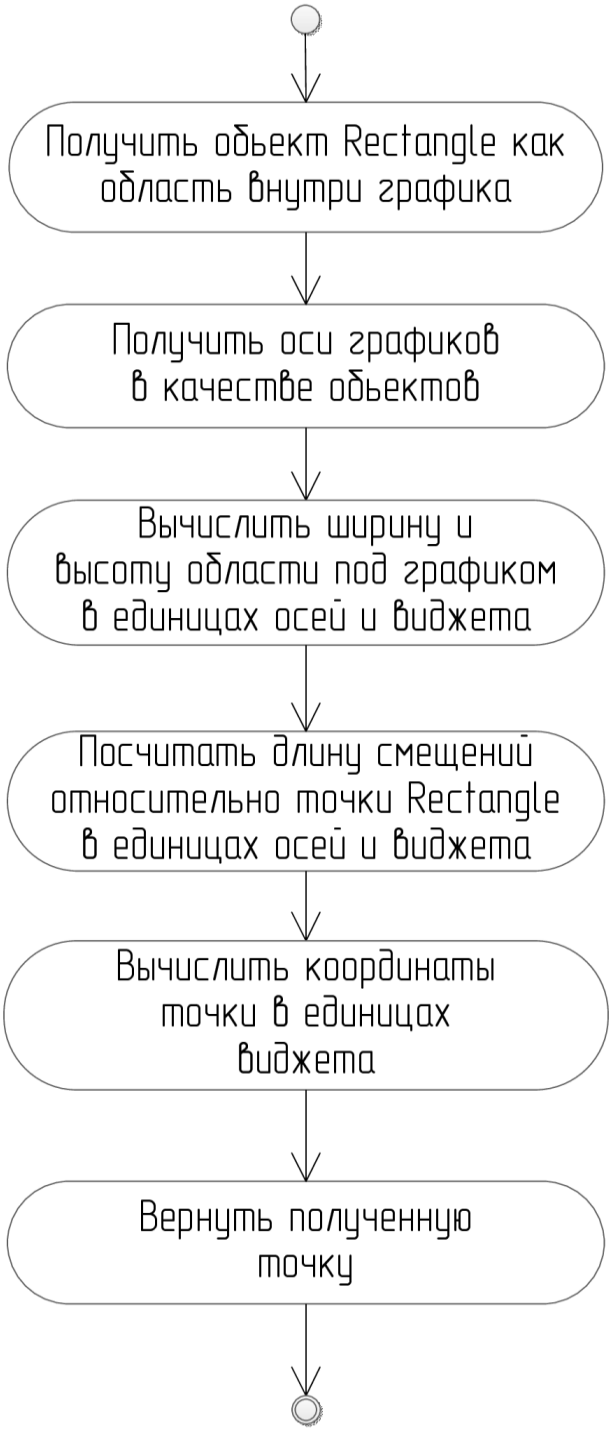


Рис.3 «Алгоритм метода перевода координат»

4.4. Разработка вывода всплывающей подсказки для точек на графике

Создан внутренний класс ***PlotMouseMoveListener***, реализующий интерфейс ***MouseMoveListener***. Схема работы приведена на рис.4.

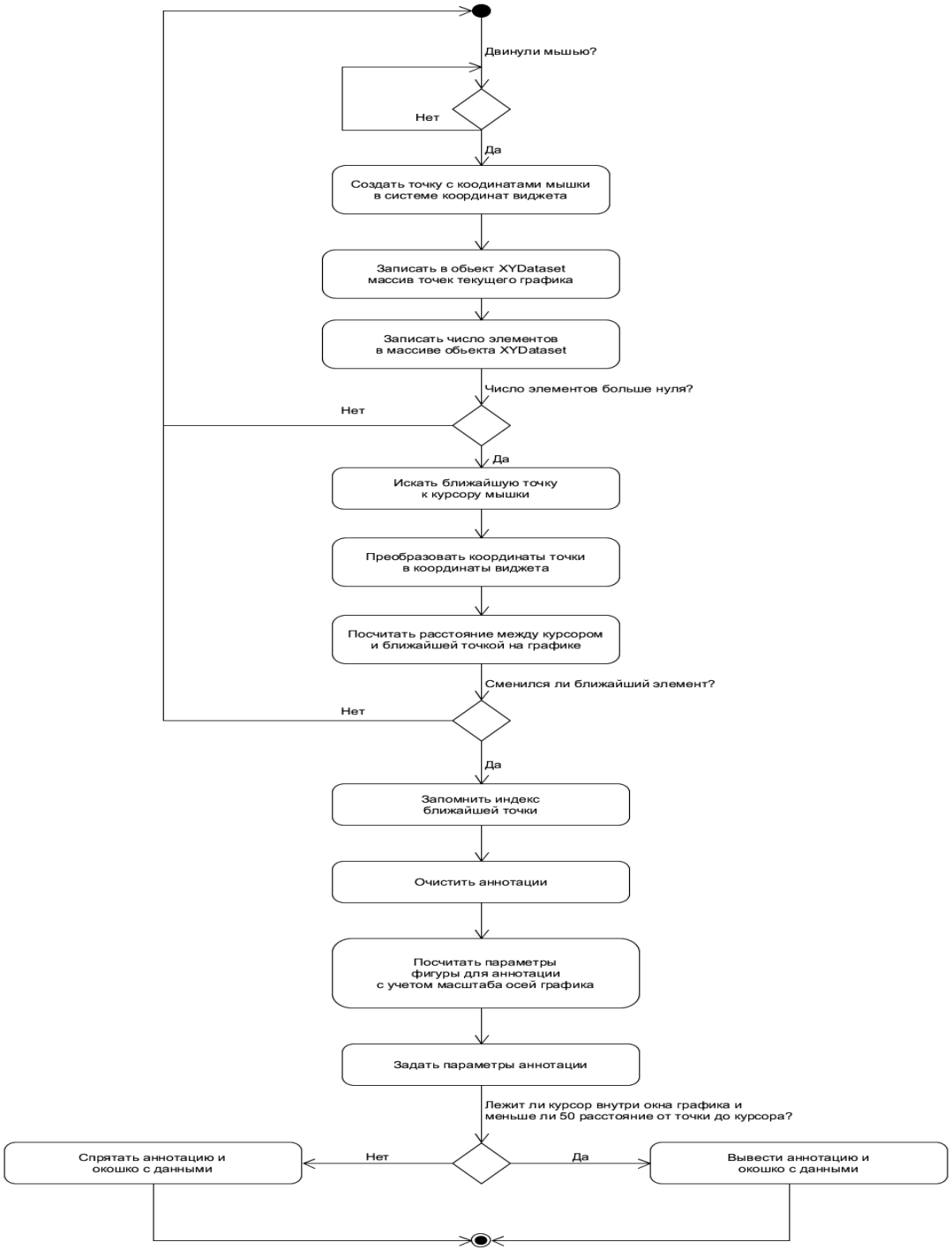


Рис.4 «Диаграмма активности возникновения всплывающей подсказки»

mouseMove()- метод, который перехватывает движение мышью. Внутри метода проводится расчет расстояния от ближайшей точки до курсора в переменную ***distanceToMouse***. Если индекс точки сменился и ***distanceToMouse<50***, то высвечивается окошко со значениями координат ближайшей точки [5] в системе графика.

Подсвечивание точки выполняется с помощью метода ***XYShapeAnnotation***(). В качестве Shape передается обьект Ellipse2D [4], размеры которого предварительно рассчитывается исходя из масштаба графика.

4.5.Построение графиков с одной или совпадающими точками

Создаем подписчик ***endSubscriber***, подписанный на состояние симулятора «execution_completed». Он меняет значение флага ***isLastEntry*** на ***true***. И обращается к обьекту ***realtimeupdate*** .

Расширяем метод ***run()*** во внутреннем классе ***PlotView RealTimeUpdateRunnable***. При значении ***isLastEntry=true*** выполняется проверка на количество точек. Если одна или две совпадающие, то добавляется точка в конец графика.

Навигация по графику происходит с помощью вертикального и горизонтального слайдеров.

Размеры и положение слайдеров зависят от масштабирования, поэтому при каждом изменении масштаба графика необходимо переопределять данные характеристики слайдеров. Это выполняется в методах ***setSliders()*** и ***setSlidersMaximum()*** в графической части подсистемы визуализации.

5. Рабочий этап проектирования подсистемы

На рабочем этапе проектирования системы были реализованы разработанные на предыдущих этапах схемы и концепции.

5.1. Реализация правильной работы слайдеров

Для корректного обновления слайдеров, необходимо дополнить метод ***updateSliders()*** условием для пересчета ***vertical slider***. Условие аналогичное horizontal slider:

```
if (verticalMaximum - rangeAxis.getRange().getUpperBound() > sliderConst) {
                        verticalSlider.setVisible(true);
                        verticalSlider.setEnabled(true);

verticalRatio = verticalSlider.getMaximum() / verticalMaximum;
verticalSlider.setThumb((int) Math.round((rangeAxis.getUpperBound() - rangeAxis.getLowerBound()) *
verticalRatio));
                        verticalSlider
                                        .setSelection((int) Math.round((verticalMaximum -
rangeAxis.getUpperBound()) * verticalRatio));
                } else {
                        verticalSlider.setVisible(false);
                        verticalSlider.setEnabled(false);
                }
```

Чтобы обновление размеров слайдеров выполнялось при исполнении любого метода масштабирования, следует в каждый из них добавить метод **updateSliders().**

```
@Override
        public void zoomInDomain(double x, double y) {
                super.zoomInDomain(x, y);
                updateSliders();
        }
```

С помощью ключевого слова ***super*** вызывается стандартный метод из класса ***ChartComposite***. В след за ним в переписанном методе масштаба вызывается ***updateSliders().***

5.2. Реализация удобного способа масштабирования

В конструкторе ***PlotFrame*** регистрируем слушатели клавиатуры и колеса мыши:

```
addSWTListener(new PlotKeyListener());
addMouseWheelListener(new PlotMouseWheelListener());
```

Задаем шаг масштабирования: setZoomInFactor(0.75); setZoomOutFactor(1.25);
В методах класса ***PlotKeyListener*** создается переключатель, который меняет значение флагов ***isRangeZoomable*** и ***isDomainZoomable***. Нажатие на пробел сбрасывает масштаб в исходное состояние.

```
@Override
		public void keyPressed(KeyEvent e) {
			switch (e.keyCode) {
			case SWT.SHIFT: {
				setRangeZoomable(true);
				return;
			}
			case SWT.CTRL: {
				setDomainZoomable(true);
				return;
			}
			case SWT.SPACE:
				restoreAutoBounds();
				return;
			}
		}
@Override
		public void keyReleased(KeyEvent e) {
			if (e.keyCode == SWT.SHIFT) {
				setRangeZoomable(false);
			} else if (e.keyCode == SWT.CTRL) {
				setDomainZoomable(false);
			}
		}
	}
```

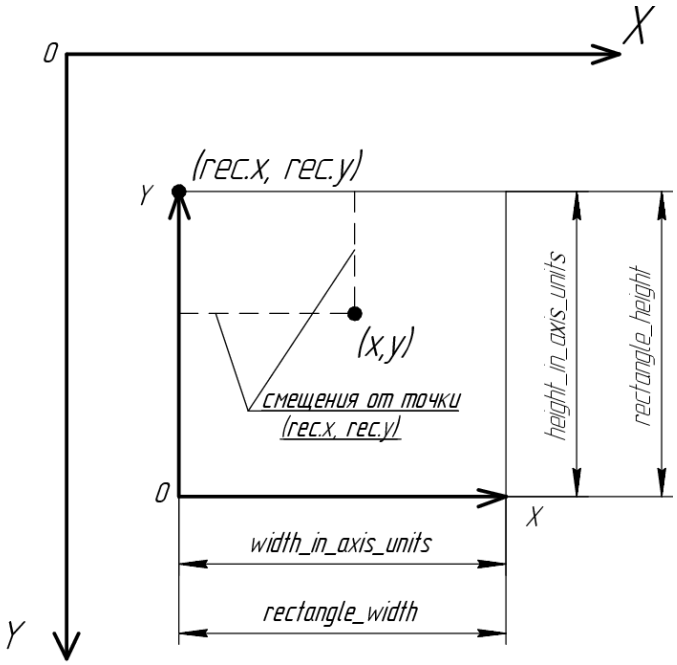
Переменная ***e.count*** отвечает за направление прокрутки колеса мыши.

```
	class PlotMouseWheelListener implements MouseWheelListener {

		@Override
		public void mouseScrolled(MouseEvent e) {

			if (e.count > 0 && isRangeZoomable()) {
				zoomInRange(e.x, e.y);
			}
			if (e.count < 0 && isRangeZoomable()) {
				zoomOutRange(e.x, e.y);
			}
```

```
                if (e.count > 0 && isDomainZoomable()) {
                        zoomInDomain(e.x, e.y);
                }
                if (e.count < 0 && isDomainZoomable()) {
                        zoomOutDomain(e.x, e.y);
                }
                if (e.count > 0 && isDomainZoomable() && isRangeZoomable()) {
                        zoomInBoth(e.x, e.y);
                }
                if (e.count < 0 && isDomainZoomable() && isRangeZoomable()) {
                        zoomOutBoth(e.x, e.y);
                }
        }
}
```

5.3. Реализация перевода координат точки графика в координаты виджета

Метод ***plotToSwt(double x, double y)*** понадобится для решения задачи с выводом всплывающей подсказки с координатами точек. Пояснения к методу приведены на рис.5



Обьект ***rectangle*** содержит в себе прямоугольную область внутри осей графика

width_in_axis_units содержит в себе ширину абсцисс в единицах оси.

height_in_axis_units содержит в себе высоту ординаты в единицах оси.

Обьекты класса **ValueAxis domainAxis** и ***rangeAxis*** содержат в себе оси графика.

Рис.5 «Графическая иллюстрация к методу plotToSwt»

```
Rectangle rectangle = PlotFrame.this.getScreenDataArea();
final ValueAxis domainAxis = getChart().getXYPlot().getDomainAxis();
final ValueAxis rangeAxis = getChart().getXYPlot().getRangeAxis();
```

```
double width_in_axis_units = domainAxis.getUpperBound() - domainAxis.getLowerBound();
double height_in_axis_units = rangeAxis.getUpperBound() - rangeAxis.getLowerBound();

        double relativeX = (x - domainAxis.getLowerBound()) / width_in_axis_units;
        double relativeY = (rangeAxis.getUpperBound() - y) / height_in_axis_units;
        double screenX = relativeX * rectangle.width + rectangle.x;
        double screenY = relativeY * rectangle.height + rectangle.y;
```

Возврат посчитанных значений:

```
        return new Point((int) Math.round(screenX), (int) Math.round(screenY));
}
```

5.4. Реализация вывода окошка с координатами точек на графике

Задаем параметры окошка в конструкторе ***PlotFrame***:

Определяем тип виджета: `toolTip = new DefaultToolTip (this, SWT.BALLOON, true);`

Задаем цвета окошка:

```
toolTip.setBackgroundColor (new Color(comp.getDisplay(), 255, 255, 255));  toolTip.setForegroundColor (new Color(comp.getDisplay(), 128, 128, 128));
```

Задаем смещение: `toolTip.setShift(new Point(0, -50));`

Параметры появления/исчезновения окошка:

```
toolTip.setRespectDisplayBounds(true);
toolTip.setHideDelay(0);
toolTip.setHideOnMouseDown(false);
toolTip.setPopupDelay(0);
```

Регистрируем слушатель движений мышки:

```
addSWTListener(new PlotMouseMoveListener());
```

5.5. Реализация построения графиков с одной или с двумя совпадающими точками

Создаем переменную-флаг ***isLastEntry*** и метод ***changeLastValueFlag()*** , который меняет значение флага при срабатывании.

```
private boolean isLastEntry;
```

```
private void changeLastValueFlag() {
                        isLastEntry = true;          }
```

Создаем подписчик ***endSubscriber***:

```
private final Subscriber endSubscriber = new Subscriber() {
        @Override
        public void fireChange() {
                realTimeUpdateRunnable.changeLastValueFlag();
                PlatformUI.getWorkbench().getDisplay().asyncExec(realTimeUpdateRunnable);
        }
};
```

Добавляем подписчик ***endSubscriber*** на событие симулятора «execution_completed».

```
private final void initializeSubscribers() {
        simulatorSubscriberManager.initialize(
                        Arrays.asList(new SimulatorSubscriberInfo(commonSubcriber,
ExecutionState.EXECUTION_STARTED),
                                        new SimulatorSubscriberInfo(endSubscriber,
ExecutionState.EXECUTION_COMPLETED)));
```

Расширяем метод ***run()***, добавив в него проверки условий на количество точек в конце симуляции.

```
if (isLastEntry) {
                        final XYSeriesCollection newDataset = (XYSeriesCollection)
plotFrame.getChart().getXYPlot()
                                        .getDataset();
                        final XYSeries newSeries = newDataset.getSeries(0);
                        if (newSeries.getItemCount() == 2) {
                                @SuppressWarnings("unchecked")
                                List<XYDataItem> seriesitems = newSeries.getItems();
                                if (seriesitems.get(0).equals(seriesitems.get(1))) {
                                        newSeries.add(CurrentSimulator.getTime(),
seriesitems.get(1).getY());
                                        plotFrame.setChartMaximum(newSeries.getMaxX(),
newSeries.getMaxY());
                                        plotFrame.updateSliders();
                                }

                        } else if (newSeries.getItemCount() == 1) {
                                @SuppressWarnings("unchecked")
                                List<XYDataItem> seriesitems = newSeries.getItems();
                                newSeries.add(CurrentSimulator.getTime(), seriesitems.get(0).getY());
                                plotFrame.setChartMaximum(newSeries.getMaxX(),
newSeries.getMaxY());
                                plotFrame.updateSliders();

                        }
                }
```

Заключение

В рамках данного курсового проекта были получены следующие результаты:

- Реализовано масштабирование через колесо мышки и клавиатуру
- Настроены полосы прокрутки графиков
- Добавлена функция перевода координат график-виджет
- Отображается всплывающее окошко с координатами ближайшей к курсору точки
- Строятся графики с одной или несколькими совпадающими точками

Список используемых источников

1. **Емельянов В.В., Ясиновский С.И.** Введение в интеллектуальное имитационное моделирование сложных дискретных систем и процессов. Язык РДО. - М.: "Анвик", 1998. - 427 с., ил. 136.
2. **Документация по языку РДО** [электронный ресурс] // Справка по языку РДО URL: http://www.rdostudio.com/help/index.html (дата обращения: 15.12.2016).
3. **Герберт Шилдт**. [Herbert Schildt] Java 8: руководство для начинающих, 6-е изд. Пер. с англ. - М. ООО "И.Д. Вильяме", 2015 -720 с.
4. **SWT Documentation** [электронный ресурс] // Документация по библиотеке SWT в открытом доступе URL: https://www.eclipse.org/swt/docs.php (дата обращения: 20.11.2016).
5. **AWT Documentation** // Документация по библиотеке AWT в открытом доступе URL: https://docs.oracle.com/javase/8/docs/api/ (дата обращения: 20.11.2016)

Список использованного программного обеспечения

1. RAO-Studio
2. Eclipse IDE for Java Developers Mars 2.0
3. openjdk version "1.8.0_40-internal"
4. UMLet 14.2
5. Inkscape 0.91
6. Microsoft® Office Word 2010
7. Microsoft® Visio 2010