# Визуальный редактор графов

Составил Сергей Раш

Концепция разработки визуального редактора графов, рабочее название проекта GraphLab.

## Содержание

[1. Введение](#_1._Введение)

[2. Цель, задачи и ожидаемые результаты](#_2._Цель,_задачи)

[3. Этапы создания](#_3._Этапы_создания)

[4. Архитектура и программная модель](#_4._Архитектура_и)

[5. Структура объектной модели](#_4._Структура_объектной)

[6. Сроки и этапы реализации проекта](#_5._Сроки_реализации)

7. Внешнее АПИ

8. Критические факторы реализации проекта

9. Разработка документации

## 1. Введение

Визуальный редактор графов GraphLab предлагается в качестве альтернативной замены редактора IBM i2 Analyst’s Notebook.

При разработке концепции и архитектуры GraphLab ставилась цель не только реализовать клон i2 Notebook, но создать приложение нового поколения превосходящего возможности i2 Notebook по 2 параметрам: анализ данных и производительность графики.

GraphLab является инструментом для анализа данных. Пользователь взаимодействует с представлением, манипулирует структурами, их формой и цветом, чтобы выявить скрытые закономерности. Задача приложения - помочь анализу данных, создать или подтвердить гипотезу, интуитивно выявить шаблоны, изолировать структуры особенностей или найти ошибки в источнике данных.

GraphLab также является инструментом традиционной статистики и сопровождает визуальное мышление с помощью интерактивных интерфейсов для составления парадигмы предмета исследований в области визуальной аналитики.

## 2. Цель, задачи и ожидаемые результаты

Основная цель создания GraphLab снизить расходы на приобретение лицензии коммерческого приложения IBM i2Notebook;

В силу своей универсальности редактор может быть позиционирован в узкоспециализированных нишах как основной инструмент анализа.

### 2.1. Цель проекта

Основной целью является создание визуального редактора, который позволит производить создание и отображение графов в реальном режиме.

Данный редактор является универсальным решением, с его помощью можно также реализовать и другие цели, связанные с отображением диаграмм, графиков и др.

### 2.2. Задачи проекта

В рамках подготовки данного предложения был разработан ряд технических решений, моделирующих различные режимы работы данного редактора.

Формирование технического задания на проектирование, разработку и внедрение КХД.

Разработка организационных и технических регламентов по сбору (регламент информационного взаимодействия), обработке и загрузке данных .

Быстро кусочкам разрозненные данные в единую сплоченную разведывательную картину.

Определение ключевых людей события, связи и модели с инновационными функциями, как анализ социальной сети, "список наиболее подключен" и "найти связанные сети".

Повысить понимание структуры, иерархии и способа работы уголовных, террористических и мошеннических сетей.

Упростить связь сложных данных для своевременного и точного оперативного принятия решений.

Задействования быстрого развертывания, что обеспечивает повышение производительности быстро, используя хорошо налаженные визуальное решение для анализа.

## 3. Этапы создания

Разработка редактора производится в 3 этапа – разработка и тестирование визуального ядра, создание пользовательского интерфейса, разработка внешних методов интеграции.

3.1. Разработка и тестирование визуального ядра является самой сложной задачей и займет около 50% рабочего времени.

3.2 Разработка пользовательского интерфейса займет 30% времени

3.3 Разработка методов интеграции займет 20% времени

## 4. Архитектура и программная модель

### 4.1 Участники

Архитектура проекта состоит набора модулей и плагинов. Модули формируют базовую, фундаментальную структуру, плагины позволяют специализировать функции приложения. Модульная структура реализована в самом основании, что позвол

### 4.2 Разделение сущностей

Основная концепция в реализации архитектуры это предоставить разработчикам возможность работать с узкими моделями, а пользователям использовать заточенную под конкретную деятельность взаимодействие. Таким образом решается задача уменьшить сложность кода для программистов и приспособиться под конкретные нужды пользователей.

### 4.3 Качество

Широкое использование типизируемых структур и полиморфных методов позволяет отказаться от классического представления обработки данных. В зависимости от состава данных применяется соответствующий функционал обработки потока данных. Этот подход позволяет повысить отказоустойчивость, обратную совместимость, расширяемость, надежность, безопасность и удобство использования.

### 4.4 Внешний дизайн

Для оформления дизайна приложения используются библиотеки стилей, которые формируют общий дизайн UI и возможность выбора темы или общего цвета.

### 4.5 Компоновка

Компоновка приложения реализована в концепции Shell и представляет динамично формируемый пользовательский интерфейс. Настройки схемы расположения вспомогательных окон сохраняется по завершении работы приложения. В любой момент пользователь может восстановить расположение окон по умолчанию.

### 4.6 Метрика

Система статистики и метрики составляют наиболее общие показатели для анализа

## 5. Структура объектной модели

Объектная модель состоит из поверхности графического дизайнера (холста) представленного объектом Canvas, набором структур Nodes (вершины) и связей Links.

Во время построения сервис LayoutMesh следит за размещением объектов на схеме. Объекты, которые не имеют Location Point (расположения), будут размещены на схеме с помощью сервиса LayoutMesh.

Рендеринг объектов включает в себя создание на схеме узлов и связей и состоит из 2 этапов: вывод вершин Nodes и связей Links. Вывод вершин Nodes производится в соответствие с координатами вершин или с помощью сервиса LayoutMesh при отсутствии таковых. На втором этапе производится расчет и вывод связей, которые вычисляются по следующему сценарию.

Каждая вершина Node является скрытым прямоугольником (для облегчения математических расчетов сервиса LayoutMesh), который для пользователя может быть представлен любой другой фигурой или изображением.

Вершина Node содержит скрытый (или видимый) узел соединения, который, в простейшем случае, представлен центром пересечения диагоналей прямоугольника. Эта точка предназначена для вычисления геометрии связи между 2D координатами X1, Y1 => X2, Y2.

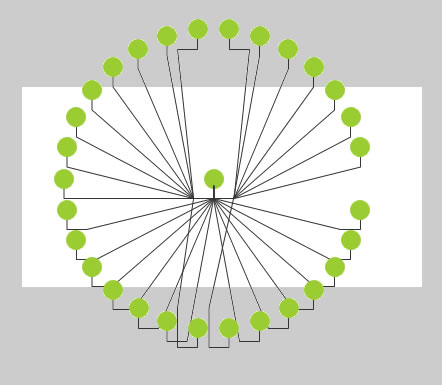
Таким образом, мы получаем идеальные условия для быстрого вычисления геометрии связи для прямой или кривой (Безье) траектории, третья точка в последнем случае выбирается из контекста. Допустимо применение и более сложных алгоритмов (выходит за рамки данного описания).

Передвижение

Для масштабирования и передвижения поверхности дизайнера Canvas используются сервисы Zoom и Pan соответственно. Эти сервисы работают в зависимости от сервиса LayoutMesh, которые не позволит выводить в графическую подсистему ЭВМ объекты, которые находятся за пределами текущей проекции поверхности дизайнера Canvas.

Для перемещения объектов на схеме используется сервис Drag-And-Drop, который в режиме реального времени вычисляет вектор смещения координат вершины и перестраивает геометрию связи соответственно. На событие Drop Down Event координаты передаются в сервис LayoutMesh для регистрации.

Реализация данного сценария позволяет отображать на поверхности дизайнера Canvas только те объекты, которые доступны в рамках проекции. Остальные объекты уничтожаются и визуализируются вновь, как только пользователь перемещает или масштабирует поверхность дизайнера Canvas.



На рисунке белым цветом изображена видимая пользователю поверхность Canvas, серым цветом выделены объекты за пределами проекции. Во избежание лишних математических вычислений эти объекты уничтожаются, если находятся вне проекции.

## 6. Сроки и этапы реализации проекта

В целях сокращения времени разработки и интеграции приложения в практическое применение на начальном этапе предполагается реализовать архитектуру, которая бы обеспечила в будущем непрерывное улучшение качества и производительности параллельно практическому использованию.

В связи с этим первая реализация не предполагает использование продвинутых подходов к функциям приложения, но создание архитектуры, которая в дальнейшем позволит это сделать.

Этапы реализации проекта версии 1.0 выделены в 4 стадии:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Стадия разработки | Цель | Описание цели | Срок | State |
| 0 | Pre-Alpha | Разработка прототипа | Создание архитектуры и набора основных функций | 1 месяц | Unstable |
| 1 | Alpha | Black-box validation | Тестирование в практическом применении | 1 месяц | Unstable |
| 2 | Beta | Feature complete | Usability testing, demo, public beta | 1 месяц | Stable |
| 3 | Release Candidate | Product stabilization | Code complete | 1 месяц | Stable |

Пояснение к таблице:

0. Pre-Alpha или нулевая стадия предусматривает создание архитектуры и набора основных функций приложения.

1. Alpha стадия предусматривает создание нестабильного приложения. По завершении стадии предполагается закрыть кейс внешней интеграции.

2. Beta – эта версия уже может применяться в тестовых проектах. Создание функционал на этой стадии завершено, АПИ интеграции выработано. Возможно практическое применение.

3. Release Candidate характеризуется состоянием «Code complete», т.е. добавление нового кода не предполагается, работа в основном ведется на улучшением существующего.

Основное заключение – продукт может уже быть использован со стадии 2. Дальнейшие изменения ожидаются, но уже не носят концептуальный характер.

## 7. Внешнее АПИ

### 7.1 Graph API

Интерфейс 1 Graph API имеет классические функции, необходимые для отображения графа и манипуляции узлами и связями.

Структура графа защищена блокировкой чтения-записи. Это означает, что несколько потоков могут читать график в тот же момент, но поток изменений является эксклюзивным. Если поток в настоящее время обновляет граф, потоки чтения должны ожидать завершения.

### 7.2 Layout API

Layout API управляет компоновкой узлов и связей на поверхности дизайнера. Пользователь может добавлять узлы и связи, изменять их расположение, добавлять атрибуты и пр.

### 7.3 Attributes API

Attributes API предоставляет доступ к атрибутам в к словарю ключ-значение. Различные виды данных могут быть установлены для каждого элемента. API позволяет определять столбцы данных с заголовком и типа в таблицах, а затем создает строку, чтобы передать значения данных. По умолчанию, есть две таблицы: узел и связь.

### 7.4 Statistics API

Statistics API обеспечивает асинхронное выполнение алгоритмов и инфраструктуры пользовательского интерфейса. API позволяет выполнять статистики и управлять StatisticsModel, по одному в каждой рабочей области.

### 7.5 Import API

Import API обеспечивает процесс импорта данных. Данные загружаются в АПИ, валидируются, при отсутствии некоторых параметров значения назначаются по умолчанию.

### 7.6 Export API

Export API осуществляет выгрузку данных со схемы графа.

### 7.7 Tools API

Tools API определяет действия которые доступны пользователям для визуализации.

### 7.8 Filters API

Filters API предоставляет функции фильтрации для создания производных графа.

API позволяют пользователям создавать и объединять фильтры запросов и отображать результаты на текущем графике. Новые фильтры могут быть легко добавлены. Фильтры объединены в категории и классифицированы в виде иерархического дерева.

### 7.9 Generator API

Generator API осуществляет создание графов из различных свойств

### 7.10 Project API

Project API осуществляет взаимодействие пользовательского интерфейса с другими модулями приложения.

### 7.11 LongTask API

LongTask API предоставляет функции для выполнения длительных операций. API может быть использован любым модулем, чтобы обеспечить безопасное выполнение задачи в фоновом потоке с поддержкой прогресса и отмены, а также управления ошибками.