# РАЗРАБОТКА МОДЕЛИ ПРЕДСТАВЛЕНИЯ СЛОЖНЫХ ОБЪЕКТОВ В ВИДЕ ГРАФОВ ОБЪЕКТОВ-СИСТЕМ ПРИ ПРОЕКТИРОВАНИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

*Степанов В.А.[1], Лоскутов В.В.[2],*
*[1]Томский политехнический университет, студент гр. 8К14, e-mail: vas134@tpu.ru*
*[2]Томский политехнический университет, ст. преподаватель ОИТ ИШИТР, e-mail: rewenger@tpu.ru*

## Введение

В связи с нарастающей сложностью проектируемых человеком систем приходит необходимость использовать различные инструменты для упрощения проектирования. Одними из таких инструментов являются модели систем. Модели позволяют получить структурированную информацию о системе и ее компонентах, а также связях между ними. Подобные подходы к проектированию повсеместно применяются во многих сферах деятельности от психологии до машиностроения. Использование моделей в программировании особенно актуально, так как здесь необходимо представлять объекты реального мира более абстрактно в виде структур данных. Использование единой модели, дающей возможность быстро создавать повторяемые решения (т.е. шаблонные), позволяет повысить эффективность написания программного кода. Это необходимо в случаях, если требуется описать поведение как целого объекта-системы, так и всех его элементов в отдельности.

## Описание модели

Представленная модель позволяет упростить проектирование систем высокой сложности. Любой сложный объект может быть представлен как система, состоящая из дочерних объектов, которые являются элементами данной системы. Элементы имеют собственные свойства, а также влияют на параметры системы в целом. Объект-система может обладать некоторыми свойствами, недоступными отдельным ее элементам. Так, например, самолет, состоящий из множества деталей, способен летать.

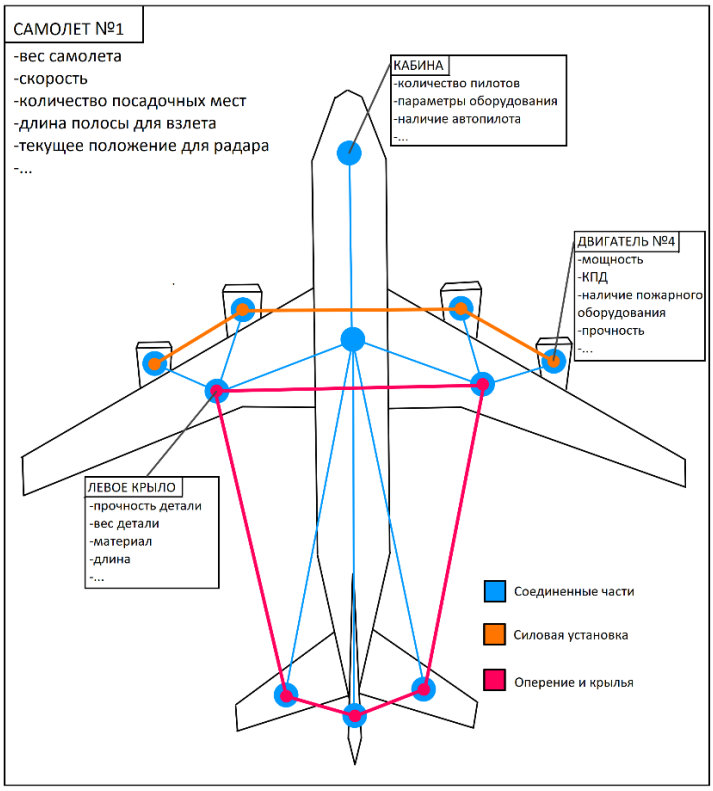


*Рис. 1. Модель транспортного самолета*

Система обладает композицией, то есть связями между элементами. Связи могут отражать различные отношения элементов друг к другу; более того, возможно множество связей, отражающих различные отношения. Наиболее удобно и абстрактно эти связи можно представить несколькими графами

с общим множеством вершин. Графы являются очень удобной структурой, зарекомендовавшей себя во многих сферах. Ее использование открывает доступ к множеству возможностей. Например, использование различных алгоритмов обхода при оповещении, изменении или отсоединении множества элементов системы. Также могут пригодиться алгоритмы для обнаружения циклов, построения остовного дерева, нахождения сильно связанных элементов, “раскраски” графа, и.т.д.

Как структура данных, объект-система имеет доступ к каждому своему элементу и ко всей информации о связях между ними. Если необходимо получить какую-либо информацию о системе извне, то следует обращаться к элементам через нее, а не напрямую. Компоненты системы сами могут являться системами, таким образом можно получить достаточную точность рассмотрения компонентов. Это необходимо, если изначальный объект является очень комплексной системой.

**Реализация модели на мультимедийном движке Unity**

Как говорилось выше, данная модель может использоваться почти в любой сфере, где приходится проектировать или рассматривать системы различной сложности. В данном случае будет рассмотрено проектирование системы, представляющей собой врага в видеоигре. Согласно концепции игры все объекты состоят из частей. Если игрок ударяет по части какого-либо объекта, то она теряет прочность. Если прочности не осталось, то часть покидает объект-систему и становится самостоятельным объектом, с которым можно взаимодействовать напрямую.

Все объекты обладают собственными поведением и параметрами, основным из которых является прочность структуры. Так как объект-система обладает несколькими элементами, очевидно, что при потере одной из них структурная прочность всей системы уменьшится. А если она достигнет нуля, то система перестанет существовать.

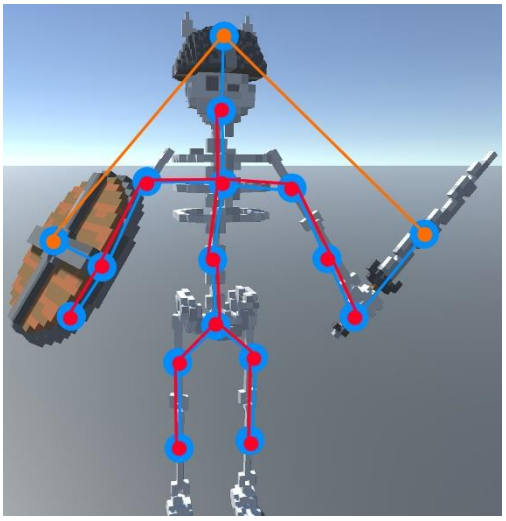

*Рис. 2. Пример использования модели для врага*

- Здесь синий цвет показывает соединения частей друг с другом. Если часть крепилась к другому элементу, покинувшему объект-систему, то по логике приложения ей будет не на чем держаться, и она тоже должна покинуть систему.
- Красным цветом выделено множество элементов, при потере которых структурная прочность уменьшится. Потеря таких элементов также способна сильно изменить поведение системы.
- Оранжевым цветом показаны части снаряжения. Они не влияют на прочность структуры, но изменяют остальные параметры системы.

Основное взаимодействие между объектами реализовано через считывание коллизии, то есть столкновений объектов друг с другом. Соответственно каждый элемент системы обладает как минимум своим коллайдером, описывающим границы объекта, и прочностью. При столкновении элемента с каким-либо другим объектом его собственная прочность уменьшается.

Благодаря использованию данной модели и возможностей объектно-ориентированного программирования возможно сильно упростить создание приложения. Так в представленной игре для реализа-

ции всех объектов-систем используется лишь два основных класса-скрипта (система, элемент), от которых происходит наследование всех остальных.

**Заключение**

В итоге была получена модель описания объектов-структур, которая позволяет упростить проектирование систем как для реальных объектов, являющихся слишком сложными и состоящими из множества частей, так и для представления этих объектов в программном коде приложения. Стоит отметить, что описанная модель является наиболее полным представлением объектов-систем и может быть упрощена в зависимости от комплексности систем и количества информации, которую необходимо получить в ходе работы с объектом.

**Список использованных источников**


1. Левенчук А. И. - Системное мышление: Учебник – “Издательские решения”. – 2018. – 398 с.
2. Павловская Т. А. – C#. Программирование на языке высокого уровня: Учебник для вузов. – СПб.: Питер, 2015. 432 с.: ил. – (Серия “Учебник для вузов”)
3. Unity Documentation [Электронный ресурс]. – URL: https://docs.unity.com (дата обращения 21.02.2023)