В поисках серебрянной пули.

FSM-based programming как основа архитектуры онлайн-игр.

Денис Подлужный
INTENIUM GmbH
16 мая 2010

Представление докладчика, компании и продукта

Про меня:

- Денис Подлужный, технический директор INTENIUM Studio, Калининград.

Про компанию:

- Студия разработки INTENIUM GmbH, одного из крупнейших Западно-Европейских издателей казуальных игр.

Про продукт:

- Alamandi, Casual MMO, Client-based, ~500k строк C++, Python и FSM кода, Win32/Solaris. Осень 2008го — текущее время.

О чем доклад

Результат эволюции архитектуры — переход к формально-описанным конечным автоматам как базовому примитиву.

Почему это хорошо:

- Для разработки сервера
- Для разработки кода игровой логики
- Для асинхронных user-interactions

Как нам двухфазный event-dispatching жить помогает.

И во что это вылилось на примере Alamandi.

О чем так долго говорили большевики

Подход стар как мир.

К сожалению был вытеснен из мейнстрима.

В геймдеве на слуху применительно к AI.

Область применения гораздо больше — так или иначе, явно или неявно конечные автоматы появляются в любом достаточно сложном продукте.

Естественная форма выражения механики.

Естественная форма описания асинхронного взаимодействия.

Так что конкретно?

Программа как граф объектов.

Объект как комбинация позиций в графах состояний, статических и динамических данных.

Состояние описывается набором реакций.

Переходы между состояниями только при реакциях.

Реакции возбуждаются только сообщениями.

Базовые типы событий — Event и Decision.

Двухфазный диспатчинг.

FSM-based и OO-based

Является расширением OOP.

В некотором приближении может описываться как OOP с мутабельным набором методов и тегом.

Миграция мышления — тут нечего бояться, ничего не теряем, можем поиметь плюшки. Основная проблема — tools.

Гораздо ближе к тому, как видят мир не программисты — в первую очередь дизайнеры. Становится проще понимать этих «странных» людей.

FSM-based и микротреды.

cons:

- Линейный код привычнее писать.
- Легче читается сторонним человеком.

pro:

- Persistance.
- Audit.
- Возможность визуального представления.
- Отладка.
- Синхронизация.
- Тестирование.
- Реплеи.
- ...

Free cakes, lots of it.

Persistance:

- Полезно при отладке, особенно для асинхронных и параллельных взаимодействий.
- Бесплатно получаем реплеи.
- Можно описывать в том числе и длительные по времени процессы, требующие гарантированный транзакционной целостности.

Free cakes, lots of it.

Event-driven:

- Сетевая синхронизация работает в тех же терминах что и игровой код.
- Программисты игровой логики изолированы от проблем сетевого взаимодействия и масштабирования.
- Уменьшается порог входа.

Free cakes, lots of it.

Audit:

- Представление от событий.
- Представление от состояний.
  - Второй способ лучше.

- Визуальное представление кода.
- И если очень хочется — визуальное программирование для друзей наших дизайнеров.

- Для состояния всегда можно выделить конечное число способов в него попасть.
- Легко понять «где сломалось».

Двухфазный event-dispatching.

Делим сообщения на событий и решения.
Раздельная их обработка.

Events:

- Могут посылаться любому объекту.
- Доступен весь графов объектов.
- Только чтение.
- Удобно делать локальными.

Decisions:

- Генерируются и применяются строго для себя.
- При обработке доступен только сам объект.
- Могут модифицировать состояние.
- Транслируются на все копии.

Двухфазный event-dispatching.

Можно выделить сторону-владельца.

Решения генерируются только владельцем.

Необходимо и достаточно синхронизировать только решения.

Адаптация к многопоточным и распределенным системам.

Жизнь вне онлайна

Нетривиальный пользовательский интерфейс.

Игровая механика.

Асинхронные взаимодействия.

Многопоточность без потоков.

Легковесная замена скриптов.

P2P — то же онлайн.

Реализация

Классические варианты:

- switch (m_state) …
- В динамических языках.
- Erlang-style.

У нас:

- Внешний файл, описывающий структуру конечного автомата (.fsm).
- Внешний файл, описывающий композицию объекта как набор конечных автоматов (.entity).
- Реакции как бинд события на именованный обрабочик.
- Необходимо встраивание мета-информации.

Вопросы?

Написать hate-mail:
kri2010@doarn.com

Узнать больше:
job@inteniumstudio.com