Антипаттерн

Анти-паттерны (anti-patterns), также известные как **ловушки** (pitfalls) — это классы наиболее часто внедряемых плохих решений проблем. Они изучаются, как категория, в случае когда их хотят избежать в будущем, и некоторые отдельные случаи их могут быть распознаны при изучении неработающих систем.

Термин происходит из информатики, из книги «Банды четырёх» *Шаблоны проектирования*, которая заложила примеры практики хорошего программирования. Авторы назвали эти хорошие методы «шаблонами проектирования», и противоположными им являются «анти-паттерны». Частью хорошей практики программирования является избегание анти-паттернов.

Концепция также прекрасно подходит к машиностроению. Несмотря на то, что термин нечасто используется вне программной инженерии, концепция является универсальной.

Некоторые различаемые анти-паттерны в программировании

См. Категория:Анти-паттерны для более подробного списка.

Анти-паттерны в управлении разработкой ПО

- Дым и зеркала (Smoke and mirrors): Демонстрация того, как будут выглядеть ненаписанные функции (название происходит от двух излюбленных способов, которыми фокусники скрывают свои секреты).
- Раздувание ПО (Software bloat): Разрешение последующим версиям системы требовать всё больше и больше ресурсов.
- Функции для галочки: Превращение программы в конгломерат плохо реализованных и не связанных между собой функций (как правило, для того, чтобы заявить в рекламе, что функция есть).

Анти-паттерны в разработке ПО

- Инверсия абстракции (Abstraction inversion): Создание простых конструкций поверх сложных (спорный)
- Неопределённая точка зрения (Ambiguous viewpoint): Представление модели без спецификации её точки рассмотрения
- Большой комок грязи (Big ball of mud): Система с нераспознаваемой структурой
- Блоб (Blob): см. Божественный объект (God object)
- Бензиновая фабрика (Gas factory): Необязательная сложность дизайна
- Затычка на ввод данных (Input kludge): Забывчивость в спецификации и выполнении поддержки возможного неверного ввода
- Раздувание интерфейса (Interface bloat): Изготовление интерфейса очень мощным и очень трудным для осуществления
- Магическая кнопка (Magic pushbutton): Выполнение результатов действий пользователя в виде неподходящего (недостаточно абстрактного) интерфейса. Например, в системах типа Delphi это написание прикладной логики в обработчиках нажатий на кнопку.
- Перестыковка (компьютер) (Re-Coupling): Процесс внедрения ненужной зависимости
- Дымоход (Stovepipe system): Редко поддерживаемая сборка плохо связанных компонентов
- Гонки (Race hazard, Race condition): Ошибка в определении последовательности различных порядков событий

Анти-паттерны в объектно-ориентированном программировании

- Базовый класс-утилита (BaseBean): Наследование функциональности из класса-утилиты вместо делегирования к нему
- Вызов предка (CallSuper): Для реализации прикладной функциональности методу класса-потомка требуется в обязательном порядке вызывать те же методы класса-предка.
- Ошибка пустого подкласса (Empty subclass failure): Создание класса (Perl), который не проходит «проверку пустоты подкласса» («Empty Subclass Test») из-за различного поведения по сравнению с классом, который наследуется от него без изменений
- Божественный объект (God object): Концентрация слишком большого количества функций в одиночной части дизайна (классе)
- Объектная клоака (Object cesspool): Переиспользование объектов, чьё состояние не удовлетворяет (возможно неявному) контракту переиспользования.
- Полтергейст (компьютер) (Poltergeist): Объекты, чьё единственное предназначение — передавать информацию другим объектам
- Проблема йо-йо (Yo-yo problem): Структура (например: наследования) которая тяжело понятна вследствие избыточной фрагментации
- Синглетонизм (Singletonitis): Избыточное использование паттерна синглетон

Анти-паттерны в программировании

- Ненужная сложность (Accidental complexity): Внесение ненужной сложности в решение
- Действие на расстоянии (Action at a distance): Неожиданное взаимодействие между широко разделёнными частями системы
- Накопить и запустить (Accumulate and fire): Установка параметров подпрограмм в наборе глобальных переменных
- Слепая вера (Blind faith): Недостаточная проверка (a) корректности исправления ошибки или (b) результата работы подпрограммы
- Лодочный якорь (Boat anchor): Сохранение более не используемой части системы
- Активное ожидание (Busy spin): Потребление ресурсов ЦПУ (процессорного времени) во время ожидания события, обычно при помощи постоянно повторяемой проверки, вместо того, чтобы использовать систему сообщений
- Кэширование ошибки (Caching failure): Забывать сбросить флаг ошибки после её обработки
- Проверка типа вместо интерфейса (Checking type instead of membership, Checking type instead of interface): Проверка того, что объект имеет специфический тип в то время, когда требуется только определённый интерфейс
- Инерция кода (Code momentum): Сверхограничение части системы путём постоянного подразумевания её поведения в других частях системы
- Кодирование путём исключения (Coding by exception): Добавление нового кода для поддержки каждого специального распознанного случая
- Таинственный код (Cryptic code): Использование аббревиатур вместо мнемоничных имён
- Жёсткое кодирование (Hard code): Внедрение предположений об окружении системы в слишком большом количестве точек её реализации
- Мягкое кодирование (Soft code): Патологическая боязнь жёсткого кодирования, приводящая к тому, что настраивается всё что угодно, при этом конфигурирование системы само по себе превращается в программирование.
- Поток лавы (Lava flow): Сохранение нежелательного (излишнего или низкокачественного) кода по причине того, что его удаление слишком дорого или будет иметь непредсказуемые последствия
- Магические числа (Magic numbers): Включение чисел в алгоритмы без объяснений

- Процедурный код (Procedural code): Когда другая парадигма является более подходящей
- Спагетти-код (Spaghetti code): Системы, чья структура редко понятна, особенно потому что структура кода используется неправильно
- Мыльный пузырь (Soap bubble): Класс, инициализированый мусором, максимально долго притворяется, что содержит какие-то данные.

Методологические анти-паттерны

- Программирование методом копирования-вставки (Copy and paste programming): Копирование (и лёгкая модификация) существующего кода вместо создания общих решений
- Дефакторинг (De-Factoring): Процесс уничтожения функциональности и замены её документацией
- Золотой молоток (Golden hammer): Сильная уверенность в том, что любимое решение универсально применимо. Название происходит от английской поговорки «когда в руках молоток, все проблемы кажутся гвоздями».
- Фактор невероятности (Improbability factor): Предположение о невозможности того, что сработает известная ошибка
- Преждевременная оптимизация (Premature optimization): Оптимизация на основе недостаточной информации
- Изобретение колеса (Reinventing the wheel): Ошибка адаптации существующего решения
- Изобретение квадратного колеса (Reinventing the square wheel): Создание плохого решения, когда существует хорошее

Анти-паттерны управления конфигурацией

- Ад зависимостей (Dependency hell): Проблемы с версиями требующихся продуктов, особенно в системах UNIX/GNU/Linux
- DLL-ад (DLL hell): Проблемы с версиями, доступностью и увеличением количества DLL, особенно в Microsoft Windows.

Некоторые организационные анти-паттерны

- Аналитический паралич (Analysis paralysis): Выделение непропорционально больших усилий в фазе анализа проекта
- Дойная корова (Cash cow): Закрытый продукт, приносящий выгоду, часто ведёт к самоуспокоенности относительно новых продуктов
- Продолжительное устаревание (Continuous obsolescence): Выделение непропорционально больших усилий портированию системы в новые окружения
- Сваливание расходов (Cost migration): Перенос расходов на проект к уязвимому отделу или бизнес-партнёру
- Ползущий улучшизм (Creeping featurism): Добавление новых улучшений в ущерб качеству системы
- Разработка комитетом (Design by committee): Результат того, что имеется много содействующих разработке, но не имеется единого видения
- Эскалация обязательств (Escalation of commitment): Продолжение реализации решения в том случае, когда неправильность его доказана.
- Я тебе это говорил (I told you so): Когда игнорируется предупреждение эксперта, являющееся оправданным
- Управление основанное на числах (Management by numbers): Уделение избыточного внимания численным критериям управления, когда они неважны или стоимость их получения слишком высока
- Драконовские меры (Management by perkele): Военный стиль управления без толерантности к диссидентству

- Управление грибами (Mushroom management): Удержание работников в неинформированном и занятом состоянии
- Расползание рамок (Scope creep): Дозволение рамкам проекта расти без должного контроля
- Замкнутость на продавце (Vendor lock-in): Изготовление системы, жёстко привязанной к одному поставщику.
- Тёплое тело (Warm body): Человек, чей вклад в проект под сомнением, особенно если рассмотрен в панике
- Единственный знающий человек (Single head of knowledge): ЕЗЧ (SHOK) применим в том случае, когда единственная личность во всей организации контролирует жизненно-важную область ноу-хау или информации о внутренностях системы.
- Рыцарь на белом коне (Knight in shining armor): РНБК (KISA) происходит тогда, когда личность, которая не совершает ошибок, появляется на сцене и пытается починить всё, без сообщений о том, какие изменения он/она сделал/сделает и почему.

Некоторые социальные анти-паттерны

Статус некоторых из них может быть спорным.

- Цензура (Censorship): Подавление дискуссии для предотвращения политического, социального и научного прогресса
- Концентрация власти (Political corruption, Concentrated power): Индивидуальное злоупотребление властью, даже с изначально хорошими помыслами
- Демократия (Democracy): Большая группа индивидов не может принимать аргументированные решения, а руководствуется лишь поверхностной информацией.
- Диктатура (Dictatorship): Ни один индивид не имеет всех умений необходимых для управления; также власть развращает
- Дискриминация (Discrimination): Концентрация на неуместных особенностях усиливает экономическую неэффективность и социальную напряжённость
- Догма (Dogmatic religion): Догма подавляет индивидуальное мышление и тормозит прогресс
- Нетерпимость (Intolerance): Настаивание на изменении нежелательных-но-безопасных особенностей других людей влечёт усиление напряжённости и также является бесконечной задачей
- Монополия (Monopoly): Без соперничества большинство эффектов свободного рынка не работают, и частная компания не имеет стимула действовать честно
- Система голосования на основе большинства (Plurality voting system): Политика при голосовании на основе большинства вырождается в две полярно-противоположные партии, результатом чего является подавление других политических воззрений
- Соревнование в популярности (Popularity contest): Популярность становится самодостаточной величиной и не сопоставима ни с каким другими параметрами или достоинствами
- Сегрегация (Racial segregation): Разделение по равноправию весьма редко, если вообще существует; ведёт к напряжённости
- Однопартийная система (Single-party system): Без избирательного соревнования партия не имеет побуждения управлять честно
- Тоталитаризм (Totalitarianism): Подавление индивидуальности ведёт к напряжённости, вдобавок одобренный способ жизни никогда ещё не был годен для всех
- Преступление без жертв (Victimless crime): Подавление безопасного поведения создаёт субкультуру людей, постоянно-живущих-по-другим-законам, для которых эта правовая система является врагом
- Охота на ведьм (Witch hunt): Легко отыскать козла отпущения, но если проблема никогда не решается в действительности, результатом будет являться поиск всё новых и новых козлов отпущения
- Нулевой Год (Year Zero): Социальное изменение является долгим процессом, ускорение его влечёт катастрофу

Шуточные анти-паттерны

- Паблик Морозов: Класс-потомок, созданный в соответствии с этим антипаттерном, выдает по запросу все данные класса-предка, независимо от степени их сокрытия. Название данного анти-паттерна — это каламбур, основанный на созвучии ключевого слова public (паблик), часто означающего открытый доступ к методам и полям класса в объектно-ориентированных языках программирования, и имени пионера-героя Павлика Морозова, известного тем, что он выдал своего отца-кулака.

См. также

- Запах кода

Литература

- *Perl Design Patterns* — A free online book
- *William J. Brown, Raphael C. Malveau, Hays W. McCormick III, and Thomas J. Mowbray* AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis. — John Wiley & Sons, 1998. — ISBN 0471197130

Ссылки

- Tutorial on anti-patterns [1]
- Anti-patterns catalog [2]
- Worse Than Failure [3]
- SourceMarking [4]
- Анти-паттерн «Паблик Морозов» — [5]
- Перевод статьи «Resign Patterns» — Проломы проектно-дизориентированного проектирования [6]
- Perl Design Patterns book [7]

Сноски

[1] http://www.antipatterns.com/briefing/
[2] http://c2.com/cgi/wiki?AntiPatternsCatalog
[3] http://worsethanfailure.com/
[4] http://sourcemaking.com/antipatterns
[5] http://grandmag.livejournal.com/66768.html
[6] http://www.developers.org.ua/archives/a4/2007/02/14/resign-patterns/
[7] http://perldesignpatterns.com/perldesignpatterns.html

Источники и основные авторы

Антипаттерн *Источник*: http://ru.wikipedia.org/w/index.php?oldid=24065231 *Основные авторы*: AVRS, BioVitrum, Blueslocal, Dpakoha, Fractaler, Gribozavr, Kaganer, Knyf, Koterpillar, MaxBet, Mercury, Stevebest, Toyota prius 2, VSGI, Varnav, Yaroslav Blanter, Yuriy75, Чобиток Василий, 45 анонимных правок

Лицензия