sK1 Project: Перезавантаження (перспективи розвитку sK1 і UniConvertor)

Новіков І.Є.

sK1 Project (http://sk1project.org), igor.e.novikov @ gmail.com

Проект sK1 включає в себе векторний редактор sK1, орієнтований на професійне використання в поліграфії, і консольний конвертер векторних графічних форматів UniConvertor. Редактор sK1 починався як форк проекту Skencil в 2003 р., оскільки автори Skencil зупинили розробку. На даний момент джерельному коду проекту вже більше 12 років. За 7 років незалежної розробки sK1 в проекті було реалізовано багато унікальних особливостей, завдяки яким проект став відомий широкому колу користувачів і отримав ряд міжнародних нагород. Але з огляду на те, що архітектура вихідного проекту Skencil розроблялася для набагато більш скромних цілей і досить давно, назріла необхідність кардинальної перебудови проекту.

За роки розвитку проекту кардинально змінилися його цілі: спочатку Skencil розроблявся, як простий векторний редактор зі спрощеною MVC (Model-View-Controller) архітектурою, в якій не передбачалася ні модульність, ні мультиплатформність. MVC компоненти були жорстко зав'язані один з іншим, що робило портування проекту вкрай важким.

У процесі розробки UniConvertor'а на кодової базі sK1 нам вдалося частково розділити компоненти, оскільки до проекту були жорсткі вимоги в плані мутиплатформенності. Але в результаті ми отримали два проекти (sK1 і UniConvertor) з ідентичною кодової базою, яку доводилося регулярно синхронізувати, що гальмує розробку. Подальший рефакторінг проектів дозволив виділити в загальний пакет найбільш важливі елементи - фільтри імпорту/експорту, рушій шрифтів і управління кольором. Це вирішило частково питання синхронізації, але загалом сильно розробку не спростила.

У процесі рефакторінгу з'ясувалося, що в моделі документа і фільтрах імпорту/експорту спочатку закладені архітектурні недоліки. Тривалий аналіз коду, поряд з оглядом поточних змін у використовуваних інструментах і бібліотеках дав невтішну картину - проекту потрібно вкрай глибока переробка, щоб відповідати нинішньому стану речей.

Паралельно виникла необхідність зміни віджетсета. Skencil розроблявся в ті роки, коли Qt і Gtk + бібліотеки ще тільки починали розвиватися і тому інтерфейс програми створювався з використанням Tk віджетів, які на той момент вважалися вже стабільними і традиційними для додатків, написаних на Python. Але на жаль, Tk віджети досить сильно ізольовані від сучасних десктопних оточень і за довгі роки їх функціонал серйозно не розширився. Нам довелося дописувати безліч додаткових речей, щоб хоч якось забезпечити інтеграцію в Gnome і KDE: ці зміни вилилися в цілий додатковий пакет sk1sdk. І при цьому глибокої інтеграції так і не було досягнуто: програма мала інородні зовнішні вигляд і поведінку у всіх популярних графічних оболонках.

Спочатку було прийнято рішення перенести додаток на Gtk, оскільки на базі цих віджетів побудовані найбільш популярні графічні опенсорсні програми та Gnome є стандартним середовищем для найпопулярніших десктопних дистрибутивів: Ubuntu і Mint. Початкові розробки вже показали економічність додатків в плані витрат ресурсів і практично моментальний старт.

Подальший аналіз моделі виявив, що можна серйозно поліпшити показники споживання пам'яті. На даний момент sK1 в споживанні ресурсів RAM подібний до пропрієтарних додатків типу CorelDRAW і набагато більш економічний, ніж вільний редактор Inkscape. Але користувачі висувають більш серйозні запити, наприклад можливість роботи з об'єктами типу гільйошів (векторні сітки), які містять десятки тисяч компонентів. Розроблена нами нова модель документа дозволяє вкрай економічно витрачати пам'ять і вона набагато краще відповідає задуманий архітектурі програми: замість стандартного MVC патерну ми планує задіяти MVP (Model-View-Presenter), який дозволяє уникнути дуплікації коду в процесі виконання і спрощує організацію багатодокументного додатки.

У підсумку, на даному етапі проект фактично переписується з нуля. Але цей крок досить виважений і серйозно продуманий: нарощування функціоналу на старій кодової базі призведе до того, що зробити зміни в базових компонентах буде вкрай складно, а повноцінний автоматичний рефакторинг, як в мовах з суворою типізацією, в Python, на жаль недоступний.

І остання, найголовніша мета, яка буде реалізована після глибокої перебудови проекту - детальна переробка всіх фільтрів імпорту/експорту. Користувачів UniConvertor'a не цікавлять деталі імплементації моделі та інших модулів. Перш за все їх хвилює як можна більш повна підтримка різних форматів векторної графіки. І власне щоб спростити розробку і підтримку цих фільтрів, ми і затіяли весь цей процес. Спочатку фільтри імпорту/експорту створювалися під нескладні потокові формати. Але більшість сучасних форматів векторної графіки мають досить складну внутрішню структуру. Тому потоковий розбір таких форматів украй утруднений. На це також накладається той факт, що по суті в попередньому редакторі Skencil архітектура фільтрів розроблялася на основі процедурного патерну, що характерно для потокових форматів. Ми плануємо перевести розбір форматів на об'єктні моделі типу DOM, що серйозно полегшує розробку і підтримку фільтрів імпорту/експорту, оскільки дозволяє фрагментувати імплементацію, уникаючи процедурного стилю «спагеті» (в якому через якийсь час, не те що сторонньому розробнику, але і навіть автору з власною працею важко розібратись).