T2Plus Управление эффективностью активов

Описание технической архитектуры ПО

Т2СОФТ

2023

Оглавление

1. Свойства архитектуры

Технологическая архитектура системы на базе .NET Core и собственном ORM ориентирована на сокращение сроков разработки нового функционала и расширении существующего различными участниками процесса создания и эксплуатации решений, объединяет удобный и эргономичный визуальный интерфейс, и производительную технологию обработки данных и бизнес-логики ORM.

1.1. Единая технология для всех устройств

Возможно создавать функциональные и быстро реагирующие приложения, отвечающие потребностям постоянно меняющегося бизнеса и предназначенные как для стационарных рабочих мест (веб-клиент), для интернета (веб-клиент), так и для мобильных устройств (адаптированный для работы на сенсорных экранах малого размера веб-клиент). Технологическая архитектура реализует технологию быстрой разработки программного обеспечения — технологию, которая находится между традиционной ручной методологией разработки и полностью бескодовым подходом.

1.2. Пользовательский интерфейс, управляемый моделью

Модель данных любой сложности можно описывается и параметризуется в коде. На основе описанной модели приложения автоматически создается настраиваемый пользовательский интерфейс. Реализуются следующие виды форм: списковые, иерархические, анкетные формы данных, различные формы выбора, меню и навигация, отчеты, диаграммы, аналитика. При изменении пользовательских требований или сценария использования системы достаточно внести изменения в модель приложения, при этом автоматически меняются пользовательский интерфейс и частично бизнес-логика отображения и обработки данных.

2. MVC-архитектура

Архитектура Model-View-Controller (MVC, Модель – Представление – Контроллер) представлена ниже: на рисунке показаны основные блоки, когда и как они создаются, а также области, в которых возможно расширение.

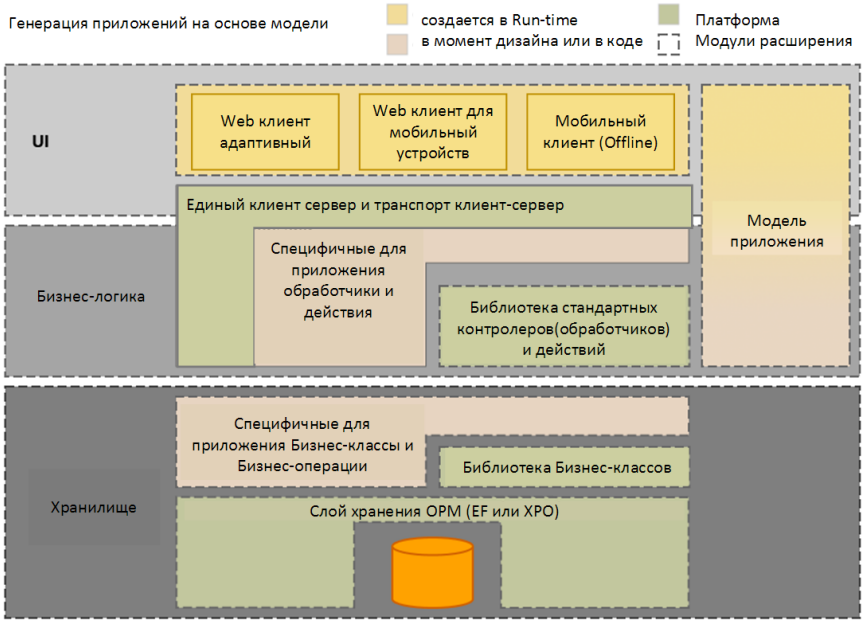

Рис. 1.

2.1. Хранилище, слой ORM

Система работает с данными через уровень ORM. При его применении не нужно вручную создавать базу данных, настраивать таблицы, отношения и проч., так как ORM автоматизирует эти операции.

Инструменты ORM позволяют использовать структуры кода (классы, свойства и их атрибуты) для описания данных приложения:

- хранимый класс для автоматически создает таблицу БД;
- общедоступные свойства класса определяют поля данных в таблице БД;
- фактический набор данных представляет собой набор экземпляров класса;
- отношения между таблицами (1:N, M:N и проч.) описываются атрибутами.

Основная БД — PostgreSQL.

3. Пользовательский интерфейс

Системная архитектура отделяет бизнес-логику от визуального представления приложения, потенциально могут быть настроены различные пользовательские интерфейсы на основе одной и той же бизнес-логики. Формы могут быть настроены как адаптивные с динамической автоматической настройкой под размер экрана пользователя.

3.1. Визуальные формы

Пользовательский интерфейс CRUD генерируется автоматически на основе данных приложения (модели приложения).

В системе присутствуют следующие основные типы форм представлений:

1) Списочная форма.

   Это ключевые формы приложения. Обычно они отображают списки либо карточки экземпляров объектов (таблицы данных). Включают возможность сортировки, фильтрации и проч.

2) Детальная форма объекта.

   Этот тип ориентирован на один экземпляр объекта (одна запись) и представляет значения свойств с помощью автономных редакторов. Детальная форма отображается при создании нового экземпляра объекта либо при редактировании.

3) Дополнительная панель.

   Форма, являющаяся расширением основной формы и содержащая связанную информацию с данными, представленными на основной форме.

4) Мастер.

   Специализированный вид визуальной формы, помогающий пользователю проходить последовательные шаги, определенные в бизнес-процессе обработки данных.

4. Бизнес-логика

4.1. Встроенные обработчики (контроллеры)

Контроллеры — это объекты, которые управляют бизнес-логикой приложения. Они отвечают за взаимодействие с конечным пользователем.

Базовые контролеры отвечают за управление данными. С их помощью можно добавлять новые записи, удалять существующие, выполнять полнотекстовый поиск и т. д.

Контроллеры служат контейнерами для действий. Действия — это абстракции элементов взаимодействия с конечным пользователем: кнопка, меню и проч. Действие определяет визуальное представление элемента пользовательского интерфейса и связанного с ним кода. Таким образом, не нужно иметь дело с низкоуровневыми деталями реализации конкретных редакторов, панелей инструментов, контекстных меню или чего-то еще. В то же время этот более высокий уровень абстракции позволяет использовать одно и то же действие в настольных и веб-приложениях.

4.2. Модель приложения

Модель приложения хранит всю информацию для создания пользовательского интерфейса приложения. Эта информация включает типы редакторов, используемых отображения и редактирования полей, заголовки и подсказки к полям на формах.

Модель приложения автоматически заполняется метаданными из компонентов приложения, таких как бизнес-объекты или контроллеры.

Специализированный редактор модели используется для редактирования модели приложения как во время разработки, так и при адаптации у заказчика.

Файлы определения модели приложения хранятся в формате XML, их можно редактировать и вручную.

5. Компонентная структура

В состав программного продукта входит системное ядро и прикладные компоненты, содержащие специфику АРМ.

Компонент	Происхождение и описание
Системное ядро **Т2 Цифровая унифицированная платформа**	Собственная разработка. Системные библиотеки обработки и хранения данных, инструментарий, обеспечивающий создание и эксплуатацию трехуровневого приложения с веб-клиентом. Включая: • Безопасность — права и полномочия. • Отчеты — генерация печатных форм. • Панели мониторинга — интерактивный анализ данных. • Экспорт-импорт — настраиваемые форматы экспорта и импорта. • Регламенты документов и список работ пользователя. • Аудит — журнализация изменений. • Филиальность — изоляция данных различных субъектов автоматизации.
T2Plus.Apm.Common.dll	Собственная разработка. Описание общих типов и программных интерфейсов.
T2Plus.Apm.Core.Base.Calendar.dll	Собственная разработка. Функциональность календарей и расписаний.
T2Plus.Apm.Core.CD.Common.dll T2Plus.Apm.Core.CD.Module.dll	Собственная разработка. Общие НСИ и сервисы работы с ними.
T2Plus.Apm.Core.Gantt.Module.dll	Собственная разработка. Функционал диаграмм Ганта.
T2Plus.Apm.Core.Licensing.dll	Собственная разработка. Контроль лицензий на систему.

Компонент	Происхождение и описание
T2Plus.Apm.Core.Common.dll T2Plus.Apm.Core.Module.dll	Собственная разработка. Базовые сущности и общие сервисы.
T2Plus.Apm.Dashboards.dll T2Plus.Apm.Dashboards.Xum.dll	Собственная разработка. Аналитические панели и сервисы их настройки, адаптеры к источникам данных.
T2Plus.Apm.DataGenerators.dll T2Plus.Apm.StandardData.dll	Собственная разработка. Загрузка и инициализация отраслевых НСИ.
T2Plus.Apm.Images.dll	Собственная разработка. Библиотека изображений.
T2Plus.Apm.Module.dll T2Plus.Apm.Xum.dll	Собственная разработка. Сущности и сервисы функциональности **T2 Plus. Управление эффективностью активов**, включая платформозависимую логику.
T2Plus.Apm.ObjectStatuses.dll	Собственная разработка. Управление статусами документов и НСИ.
T2Plus.Apm.Persistent.dll	Собственная разработка. Описание модели и структуры хранилища данных.
T2Plus.Apm.Planning.dll	Собственная разработка. Сущности и сервисы функциональности прогнозирования и планирования ТОиР.
T2Plus.Apm.Presentation.dll	Собственная разработка. Настройки и сервисы сервера приложений, системы безопасности и других компонентов системного ядра.
T2Plus.Apm.Presentation.Xum.dll	Собственная разработка. Базовая настройка пользовательского интерфейса.
T2Plus.Apm.Reports.dll	Собственная разработка. Модуль отчетности.
T2Plus.Apm.Services.dll T2Plus.Apm.Utils.dll	Собственная разработка. Общие сервисы ТОиР.
T2Plus.Apm.Smeta.dll	Собственная разработка. Сервисы интеграции технологических карт со сметами из внешних систем.
T2Plus.Apm.Xum.App.dll	Собственная разработка. Сервер приложения системы.
T2Plus.AppAboutInfo.dll	Собственная разработка. Информация о системе.

В рамках конкретной конфигурации могут быть разработаны и поставляться дополнительные компоненты для расширения функциональных возможностей системы.

6. Технологии разработки

Инструменты создания и редактирования объектов разработки

- Целевой язык разработки C#.
- Интеграция с различными системами контроля версий GIT, SVN и проч.
- В текущий момент используется локально развернутый на инфраструктуре компании (система виртуализации) инструмент разработки исходного кода Microsoft Visual Studio.
- Технология адаптации выключает функционал настройки хранилища (БД), модели приложения, подсистемы безопасности и прочие административные модули.
- Для миграции и обновления БД используется встроенная технология DB Updater.

Инструменты поддержки совместной работы разработчиков

- Интеграция с различными системами контроля версий Git, SVN.
- В текущий момент используется локальная инсталляция в закрытой инфраструктуре компании (система виртуализации) продукта Microsoft Auze DevOps в качестве средства хранения исходного текста и объектного кода программного обеспечения, а также в качестве технических средств компиляции исходного текста в объектный код ПО.

Инструменты обучения для разработчиков расширений

- Используются промышленная технология .NET Core и язык разработки C#, что обеспечивает доступность огромного количества учебной и справочной литературы, курсов и справочных материалов.
- Методички курсов обучения разработчиков.

Инструменты разработки и адаптации отчетов

- Поддержка файлов в офисных форматах (XLS, XLSX, XLSM, CSV, DOC, DOCX, RTF, ODT, TXT, PDF и т. д.) без необходимости использования проприетарных систем (например, Microsoft Office).
- Возможность экспорта списков и отчетов в различные офисные форматы.
- Полнофункциональный генератор отчетов, поддерживающий различные виды отчетов, ориентированный на генерацию сложных отчетных печатных форм.
- Настраиваемые интерактивные карты анализа (dashboard-ы), ориентированные на формирование интерактивных преставлений для контроля данных и показателей. Содержит большое количество встроенных аналитических функций.

Механизмы расширения и доработки бизнес-функциональности

- Возможность расширения каждого слоя архитектуры, как адаптеры к новым СУБД и все остальные слои системы.
- Расширение классов, изменение обработчиков, сигналов, подписок в уже выпущенном приложении.
- Возможность замера реализации бизнес-правил и бизнес-операций.
- Настройка визуальных форм (как с поддержкой наследования от базовой, так и без).
- Возможность формирования отчетов, карт анализа.
- Настройки функциональных ключей и настроек пользователя.
- Настройка сигналов и подписок.
- Настройка регламентов документооборота и согласования документов.

Встроенные API и объекты разработки

- Интерфейсы .NET Core.
- Полная поддержка Unicode и RTL для визуальной части.
- Экспорт-импорт XML, JSON, CSV с настройкой структуры.
- Автогенерация хранилища на основе модели и технология миграции.
- Автогенерация визуальных форма на основе структур данных и модели приложения.
- Встроенная в язык поддержка SQL на базе лямбда-выражений.
- Подключения к сторонним СУБД PostgreSQL, MS SQL, SQLite, MySQL и другие.

Инструменты разработки пользовательского интерфейса

- Формирование адаптивных визуальных форм на основе модели приложения.
- Возможность расширения и настройки форм без программировани.
- Расширения и конфигурирование представлений в момент исполнения.
- Настройка правил отображения (цветовые выделения, доступность обработок и проч.) на основе гибких бизнес-правил (могут описываться на естественном языке, который является одним из диалектов языка правил).

Модульное и функциональное тестирование

- Система поставляется с гибкой настройкой модульного и функционального тестирования для основной целевой аудитории групп разработчиков расширений и администраторов.
- Благодаря модульной архитектуре можно писать быстрые и легкие модульные тесты или расширенные интеграционные тесты по мере необходимости.
- Встроенная технология автоматизированного функционального или сквозного тестирования с помощью C# и/или человеко-ориентированного языка сценариев.
- Готовность к системам непрерывной интеграции, таким как Azure DevOps, NUnit, Moq и другим популярным средам тестирования.
- Поддержка инструментов сборки и тестирования с помощью NuGet Gallery.

Системные базовые функции

- Многопоточные сервисы запуска фоновых процессов.
- Журнализация системы событий и действий пользователя, журнализация всех данных с возможностью просмотра и восстановления данных (отмены действий пользователя).
- Запись и воспроизведение действий пользователя — составляет ядро системы автоматического тестирования.
- Горизонтальное и вертикальное масштабирование на основе интеграции с технологиями виртуализации.
- Балансировка нагрузки — используются внешние интегрированные механизмы.
- Полная поддержка Unit of Work.
- Поддержка удобных современных моделей конкурентного параллельного программирования.
- Технология локализации визуального интерфейса и отчетов, включая поддержку RTL-языков.
- Доступ к функциям системы, в том числе к кастомизации управляется моделью приложения и подсистемой безопасности, и права доступа.
- Развитая подсистема прав и полномочий, как на основе прикладных сущностей ролей объектов и субъектов (документы, справочники, отчеты), так и на основе системных объектов и субъектов (элементов системы, меню, компонентов разработки).
- Поддержка SSO.
- Разграничение доступа на уровне системных сущностей — элементов системы, меню, компонентов разработки, объектов.

Инструменты инсталляции, конфигурирования обновления

- Поставка обновлений осуществляется в формате .NET Core пакетов.
- Управление компонентами через конфигурационные файлы и редактор модели приложений.
- Утилиты миграции и обновления базы данных.

Системы управления базами данных

- РСУБД на базе PostgreSQL.
- Возможность использования других СУБД.

Инструменты мониторинга

- Встроенные средства для профилирования и измерения производительности.
- Аудит и журналирование системных событий.
- Аудит и журналирование событий безопасности.
- Аудит и журналирование прикладных данных.
- Возможность интеграции с внешними средствами мониторинга.

Вычислительная инфраструктура

- Linux или Windows.
- Поддержка технологии контейнеров (Nomad, Docker…).
- Работа на разных типах процессорных архитектур x86, ARM и проч. (.NET Core).

Поставка системы

- Поставляется только программный комплекс. Работа на любом оборудовании заказчика.
- Возможно развертывание к облачной архитектуре.

Безопасность

- Аутентификация по паролю.
- Аутентификация по технологии единого входа.
- Аутентификация по аппаратным токенам.
- Аудит событий ИБ.