УДК 004.457

Андрианов И.В.
студент
5 курс, факультет «Информатики и робототехники»
Уфимский государственный авиационный технический университет
Россия, г. Уфа

ПРОБЛЕМА КОНВЕРТАЦИИ СТАРОГО ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА В НОВУЮ ДИСПЕТЧЕРСКУЮ СИСТЕМУ

Аннотация: Статья посвящена выявлению способов конвертации пользовательского интерфейса (экранных форм) устаревшей диспетчерской системы (Genesis32) в новую (WinCC).

Ключевые слова: SCADA-система, диспетчерская система, экранные формы, Genesis32, WinCC.

Annotation: The article is devoted to identifying ways to convert the user interface (screen forms) of the outdated dispatching system (Genesis32) into a new one (WinCC).

Key words: SCADA-system, dispatching system, screen forms, Genesis32, WinCC.

Организация-заказчик заказывает доработку существующей АСУ ТП и присылает в рассматриваемую организацию проектную документацию (включает в себя, помимо прочего: техническое задание на доработку, общие технические требования, требования к экранным формам, перечень имен сигналов) а также существующую конфигурацию АСУ ТП, которая на данный момент эксплуатируется заказчиком.

После инициации проекта по доработке системы разработчики решают следующие задачи:

– создают экранные формы операторов системы, используя в качестве образца экранные формы существующей конфигурации;

– создают точки данных (сигналы), руководствуясь перечнем имен сигналов и техническим заданием на доработку;

– настраивают тестовую среду для отладки конфигурации.

Команда тестировщиков проводит тестирование созданных элементов системы, проверяя их на соответствие требованиям заказчика. Данные об обнаруженных ошибках фиксируются в системе управления проектами.

Одним из основных этапов доработки существующей конфигурации SCADA-системы является отрисовка экранных форм оператора, так называемых мнемосхем. При разработке новой системы отрисовка производится «с нуля», и разработчик не может использовать наработки предыдущих версий. Однако при доработке такая возможность есть, но в рассматриваемом процессе она практически не используется – экранные формы перерисовываются «на глаз».

Учитывая тот факт, что SCADA-система может содержать десяток различных экранных форм, каждая из которых может состоять из нескольких десятков различных графических элементов, работа по ручному копированию может занимать сотни часов рабочего времени. Ручное копирование также создает повышенный риск возникновения ошибки из-за человеческого фактора, что может привести к дополнительным издержкам.

Таким образом, существующий процесс доработки конфигурации SCADA-системы имеет следующий недостаток: создание экранных форм на новой платформе выполняется вручную, без использования уже существующих экранных форм, созданных для старой платформы. Данный недостаток является причиной следующих негативных эффектов:

– большая длительность процесса создания экранных форм;

– неэффективное использование рабочего времени разработчиков;

– повышенный риск появления ошибок во вновь созданных экранных формах, как следствие, дополнительные издержки, связанные с контролем ошибок и их исправлением.

Возможным решением выявленной проблемы является создание автоматизированной системы конвертации экранных форм, которая позволила бы преобразовывать файлы экранных форм, разработанных на старой платформе, в файлы экранных форм, совместимых с новой платформой.

Поставленная задача конвертации экранных форм может быть решена следующими способами:

– файлы экранных форм Genesis32 экспортируются в небинарный XML-подобный формат, и затем преобразуются по определенному набору правил в XML-подобные файлы, совместимые с WinCC;

– скриншоты экранных форм Genesis обрабатываются с помощью программы, в основе которой лежит один из алгоритмов машинного обучения. Результатом обработки будет список классифицированных графических элементов формы, который затем компонуется в XML-подобный файл, совместимый с WinCC.

Оба способа имеют свои преимущества и недостатки. Так, способ, основанный на алгоритмах машинного обучения, требует создания большой обучающей выборки (не менее нескольких тысяч элементов). Недостатком первого способа является сложность определения правил преобразования элементов.

Первый способ может быть реализован с помощью библиотеки с инструментами для разбора XML-файлов, второй способ требует использования библиотек с инструментами машинного обучения.

В настоящее время разработчику доступно большое число различных библиотек, реализующих требуемый функционал; встает задача выбора языка программирования, позволяющего решить требуемые задачи с наименьшими затратами времени и денег.

В результате предварительного отбора были выбраны следующие языки программирования, используя которые можно решить поставленную задачу:

– Python – интерпретируемый язык программирования высокого уровня с динамической сильной типизацией. Разрабатывается и поддерживается некоммерческой организацией Python Software Foundation. Экосистема Python содержит множество различных платных и бесплатных средств разработки, библиотек, фреймворков. Одним из главных преимуществ языка является его простота и выразительность, позволяющая быстро писать сложный программный код;

– C# – компилируемый язык программирования высокого уровня со статической сильной типизацией. Разрабатывается и поддерживается компанией Microsoft. Наиболее популярным и функциональным средством разработки является платная IDE Microsoft Visual Studio;

– C++ – компилируемый язык программирования высокого уровня со статической сильной типизацией. Имеет множество различных реализаций, долгую историю развития и, как следствие, большую экосистему, содержащую множество различных платных и бесплатных инструментов.

Рассмотренные альтернативы будут оцениваться по следующим критериям:

– минимальная стоимость необходимых инструментов разработки с разрешением использования в коммерческих целях – объективный показатель, измеряемый в рублях;

– сложность языка программирования – субъективный показатель, значение которого определяется экспертом;

– наличие современных поддерживаемых библиотек для работы с XML;

– наличие современных поддерживаемых библиотек для работы с алгоритмами машинного обучения.

В ходе анализа допустимых решений было выявлено, что наилучшим языком программирования, позволяющим решить требуемые задачи с наименьшими затратами времени и денег, является «Python».

Заключение

Таким образом, для решения проблемы были определенны способы конвертации экранных форм, а также был выбран язык, на котором можно будет выгодней всего реализовать такой функционал - Python. К тому же для обработки XML с помощью Python существует библиотека lxml, а для работы с алгоритмами машинного обучения – библиотека keras.

Список литературы

1) Федорова Г.Н., Информационные системы / Г.Н. Федорова. –М: Академия, 2013 – 208 с. [Электронный ресурс]. URL: http://bolohovomt.ru/doc/informazionnie_sistemi.pdf .

2) Ипатова Э.Р., Методологии и технологии системного проектирования информационных систем. / Э.Р. Ипатова Ю.В Ипатов. –М: Издательство «Флинта», 2011 – 256 с.

3) Куликов Г.Г. Теория систем и системный анализ: учеб. пособие / Г. Г. Куликов, К. А. Конев, К. Э. Маликова, Р. И. Файзрахманов. – Уфа: Уфимск. гос. авиац. техн. ун-т, 2014. – 165 с.