|  |
| --- |
| Требования к разработке роботов |

|  |  |
| --- | --- |
| **Название документа:** | Требования к разработке роботов |
| **Версия документа:** | 3.0 |
| **Дата документа:** | 16.12.2019 |
| **Назначение документа:** | Определяет требования, которые должны выполнять все разработчики системы САРП  |

**История изменений**

|  |  |  |  |
| --- | --- | --- | --- |
| **Версия** | **Дата** | **Комментарий** | **Автор** |
| 1.0. | 26.06.2019 | Исходная версия документа | Волков Ю.В. |
| 1.1 | 04.09.2019 | Исправлена работа с TFS | Волков Ю.В. |
| 2.0 | 03.10.2019 | Добавлены принципы разработки роботов.Первая рабочая версия  | Волков Ю.В. |
| 3.0 | 16.12.2019 | Исправлены требования по работе с БД робота | Волков Ю.В. |

**Связанные документы**:

|  |  |  |
| --- | --- | --- |
| **Название документа** | **Номер версии / Имя файла** | **Дата** |
| Инструкция по использованию ALM для управления кодом роботов.docx | 2.0 | 20.09.2019 |

**Согласовано**

| **Роль** | **Должность** | **ФИО** | **Дата** | **Подпись** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Содержание

[1 Общая информация о системе роботизации в сети Газпром нефть 3](#_Toc21014739)

[1.1 Применяемая терминология 3](#_Toc21014740)

[1.2 Компоненты системы роботизации 3](#_Toc21014741)

[2 Принципы разработки роботов 4](#_Toc21014742)

[3 Требования к разработке роботов 4](#_Toc21014743)

[3.1 Общие требования по разработке роботов 4](#_Toc21014744)

[3.2 Специальная кодировка бизнес-процессов в системе роботизации ГПН 5](#_Toc21014745)

[3.3 Требования к размещению файлов и папок 6](#_Toc21014746)

[3.4 Требования к скриптам роботизируемого процесса 7](#_Toc21014747)

[3.5 Требования к шаблонам и результатам работы роботизируемого процесса 10](#_Toc21014748)

[3.6 Требования по работе с БД робота 10](#_Toc21014749)

[3.7 Требования по работе с оркестратором 12](#_Toc21014750)

[3.8 Требования по работе с Team Foundation Server (TFS) 12](#_Toc21014751)

[3.9 Требования к работе с прикладными системами 12](#_Toc21014752)

[3.10 Требования к действиям, выполняемым роботом 13](#_Toc21014753)

[Приложение 1. Шаблон опросного листа для анализа бизнес-процессов 14](#_Toc21014754)

1. Общая информация о системе роботизации в сети Газпром нефть
	1. Применяемая терминология
* **Задача робота** – перечень работ, которые должен выполнить робот для того, чтобы бизнес-процесс считался выполненным;
* **Транзакция** – работа в рамках одной задачи, которая слишком мала, чтобы быть выделена в отдельную задачу, но при этом по каждой транзакции бизнес-пользователям нужен отчет;
* Пример задачи и транзакции. Допустим, есть бизнес-процесс по анализу кредитоспособности контрагентов, и в данном случае задача – анализ всех контрагентов, а транзакция – обработка данных из одной системы и по одному контрагенту
* **Глобальные переменные** – переменные, объявляемые в скрипте Main и передаваемые в остальные скрипты в виде словарей с вложенными переменными
* **Локальные переменные** – переменные, объявляемые непосредственно в основном скрипте бизнес-процесса и не используемые в других бизнес-процессах
	1. Компоненты системы роботизации

Данные системы относятся непосредственно к роботам и составляют систему роботизации:

* Ландшафт разработки – представлен двумя серверами:
	+ SPB99-UP-APP01D - Сервер приложений. Используется для разработки роботов. На нем установлена UiPath Studio, локальный UiPath Orchestrator, а также несколько конфигураций 1С АСБНУ и 1С УПП
	+ SPB99-UP-DB01D - Сервер БД. На нем развернут MS SQL Server. Используется для хранения локальных конфигураций 1С, БД TFS, БД оркестратора разработки, а также БД RPA\_Data;
* Ландшафт тестирования – представлен тремя серверами:
	+ SPB99-UP-APP01T – Сервер приложений. Используется для тестирования роботов. На нем установлена UiPath Studio, UiPath Orchestrator.
	+ SPB99-UP-DB01T – Сервер БД. На нем развернут MS SQL Server. Используется для хранения БД тестового оркестратора, а также БД RPA\_Data;
	+ SPB99-UP-TS01T – Терминальный сервер. Используется бизнес-пользователями как промежуточный для входа на сервер приложений;
* Ландшафт продуктива – представлен четыремя серверами:
	+ SPB99-UP-APP01 – Сервер приложений. Используется для продуктивного запуска роботов. На нем работают роботы по процессам, в которых может выполняться обработка коммерческой тайны, но не может выполняться обработка персональных данных, а также данных службы ИБ ГПН;
	+ SPB99-UP-DB01 – Сервер БД. На нем развернут MS SQL Server. Используется для хранения БД продуктивного оркестратора, а также БД RPA\_Data;
	+ SPB99-UP-TS01 – Терминальный сервер. Используется бизнес-пользователями как промежуточный для входа на сервер приложений;
	+ SPB99-UP-OUP01 – Сервер оркестрации. На нем развернут продуктивный оркестратор, ElasticSearch и Kibana.

Схема серверов представлена в приложении 1.

Дополнительно в системе роботизации принимает участие система ALM ИТСК – система на базе Team Foundation Server 2017, используется для переноса кода роботов и БД между ландшафтами.

1. Принципы разработки роботов

Под принципами разработки роботов понимается общая методология разработки, без привязки к конкретным действиям, которые должны выполнять разработчики.

Принципы нужны в первую очередь для того, чтобы все разработчики одинаково понимали текущие обстоятельства разработки, и подстраивались под них, в отличие от требований из главы 3, где перечислены конкретные требования по выполнению тех или иных действий разработчиков.

Общие принципы разработки роботов:

* Придерживаться рекомендаций и требований центра компетенции по роботизации ГПН в ИТСК;
* Всегда держать актуальной документацию по описанию настроек робота;
* Требования по выполняемому функционалу роботов необходимо зафиксировать до начала разработки;
* Не создавать своими действиями условия, которые будут мешать другим разработчикам (включая доработки внутри робота, которые будут влиять на других роботов);
* Писать код робота таким образом, чтобы другой разработчик мог в любой момент мог понять, что в нем происходит и продолжить его работу (или взять на себя часть его работы);
* В ходе разработки робота нельзя разрабатывать робота таким образом, что он будет нарушать правила информационной безопасности ПАО «Газпром нефть»;
* Писать робота таким образом, чтобы он одинаково выполнялся на всех ландшафтах системы роботизации;
* Использовать существующую методологию разработки, базирующуюся на использовании собственного фреймворка RPA\_GPN\_Framework и системы ALM ИТСК (подробнее – в требованиях ниже).
* Роботов нужно разрабатывать таким образом, чтобы они максимально эффективно использовали ресурсы сервера приложений. Если в процессе участвует БД – необходимо большую часть нагрузки перекладывать на БД.
1. Требования к разработке роботов

В этом блоке указаны конкретные правила выполнения различных действий, которые нужно выполнить разработчикам для того, чтобы можно было пройти приемо-сдаточные испытания и перенести разработки в продуктив.

* 1. Общие требования по разработке роботов
* Разработка в основном должна вестись на сервере SPB99-UP-APP01D. Также допустимо некоторые моменты отлаживать на сервере SPB99-UP-APP01T;
* Папки со скриптами и обрабатываемыми файлами должны быть настроены согласно пункту 3.2;
* Разработка должна быть интегрирована в управляющего робота ГПН (см. пункт 3.3);
* Разработка должна максимально использовать встроенные возможности UiPath. Подробнее – в п.3.3;
* Вся разработка должна вестись с синхронизацией с ALM ИТСК (см. пункт 2.7). Там же должны храниться все шаблоны обычных файлов, используемые в процессе;
* Перед началом разработки необходимо подготовить драфт описаний настроек робота, где нужно прописать алгоритм будущей работы робота, выявить узкие моменты и зафиксировать с бизнесом требования к будущему роботу;
* У всех разработчиков должны быть мониторы с разрешением не менее Full HD (1920x1080). Все разработки должны вестись с разрешением экрана 1920x1080, глубиной цвета – 32, т.к. на продуктивном ландшафте роботы запускаются именно с этим разрешением экрана.



* 1. Специальная кодировка бизнес-процессов в системе роботизации ГПН

В системе роботизации ГПН применяется следующая маска для кодирования бизнес-процессов:



XX-YY.ZZ.SSS, где:

* XX – тип бизнес-процесса. Определяет глобальные задачи, на которые рассчитаны процессы с одним и тем же типом. Пример кодов:

FI – используется для финансовых бизнес-процессов

HR – используется для бизнес-процессов, связанных с персональными данными

MD – используется для процессов, направленных на работу с мастер-данными и справочниками

DRP – используется для бизнес-процессов ДРП

IS – используется для автоматизации бизнес-процессов службы безопасности

* YY – порядковый номер группы бизнес-процессов определенного типа. Сюда входят все процессы, определенные логически как один бизнес-процесс и за которые ответственны одни и те же бизнес-пользователи;
* ZZ – код подпроцесса, определяет определенную последовательность логических действий части процесса. Как правило, это выполнение всех действий процесса. Допускается более чем один уровень вложенности подпроцессов в случае, если того требует бизнес-процесс. Например, FI-05.03.01
* SSS – конфигурация системы, используемая для определенного процесса. Используется только в тех процессах, где одно и то же логическое действие (например, выгрузка отчетов) может выполняться в совершенно разных типах систем (например, SAP и 1С)

Все новые процессы требуется кодировать именно таким образом и согласовывать все новые процессы с архитектором системы.

* 1. Требования к размещению файлов и папок
		1. Структура папок проекта (папка со скриптами)
* В корневой папке должен находиться только Main.Xaml и project.json, а также папки с общими скриптами, шаблонами и наименованиями роботизируемых процессов
* Папка со скриптами не должна содержать файлы, используемые роботом во время обработки (выгрузки из систем, отчеты для пользователей, и т.д.), иначе они будут автоматически подтягиваться в TFS и в оркестратор.
* В папке #Common\_templates должны находиться только те шаблоны обычных файлов, которые могут использоваться больше, чем в одном процессе
* В папке #Universal\_scripts должны находиться скрипты, которые могут использоваться более, чем в одном скрипте (тут могут быть не только скрипты UiPath)
* Все остальные скрипты должны находиться в нумерованной папке с кодом и наименованием процесса (пример: FI-01.PaymentsCheck). Внутри папки помимо скриптов UiPath должны находиться папки:
	+ External Scripts (где будут лежать хранимые процедуры других языков программирования, либо готовые функции)
	+ Templates (где будут храниться шаблоны обычных файлов, например: Word-шаблон договора и т.д.)
	+ Process\_Layer, где будут лежать папки с наименованиями систем, в которых робот выполняет действия. Внутри этих папок должны лежать скрипты, которые не вызываются из Main, но которые которые также используются в роботе
		1. Структура папок проекта (папка со бизнес-данными)
* Корневая папка содержит в себе папки с кодами + наименованиями бизнес-процессов. Также в корневой папке находится папка с обработанными задачами от робота;
* В папке с наименованием бизнес-процесса лежат следующие папки:
	+ Results – хранит в себе результаты обработки роботом продуктивных данных;
	+ Reports – в ней хранятся файлы, выгружаемые роботом из бизнес-систем и которые могут использоваться бизнес-пользователями, но которые не обрабатываются роботом;
	+ Tech\_files – хранит временные технические данные робота. В нее должны складываться данные, которые робот выгружает из системы и сразу обрабатывает, однако которые в дальнейшем не используются в процессе в «Сыром» виде;
	+ Hot Folder – бизнес-пользователи будут складывать задачи для роботов;
	+ Source – хранит исходные данные для процесса.
* Внутри Tech\_files будут лежать папки с логинами роботов. Каждый робот должен использовать свои собственные технические файлы, иначе могут быть проблемы, если несколько роботов попытаются открыть один и тот же файл;
* Внутри папки с техническими файлами конкретного робота лежат технические файлы, а также скриншоты с ошибками при работе робота. Сами скриншоты также разложены по папкам с датой, когда был сделан тот или иной скриншот;
* В папке Proceeded tasks лежат файлы, которые были получены от бизнес-пользователей и на основе которых робот создавал задачи в БД робота.
	1. Требования к скриптам роботизируемого процесса
* Все процессы Back-Office роботов должны запускаться из управляющего скрипта Main, расположенного в корне проекта Robotic Enterprise Main Framework. Создание нового проекта с использованием UiPath REFramework приведет к тому, что новые роботы будут несовместимы с роботами, которые уже работают в продуктивной среде;
* Все скрипты, вызываемые из скрипта Main, должны иметь префикс-код процесса, которому они соответствуют:
* Все скрипты должны поддерживать работу с управляющим роботом. Для этого нужно соблюдать следующие требования (**просьба особо внимательно отнестись к следующим требованиям!!!**):
	+ Требуется корректно разбивать разрабатываемый процесс на подпроцессы. Инструкции по выполнению этого действия располагаются в п.4.2, 4.3;
	+ Требуется корректно заполнять таблицы Tasks и TaskParameters, чтобы сохранялась целостность данных. Для этого существует 2 универсальных скрипта – DB\_AddNewTask и DB\_AddNewTaskParameter. Для создания задач требуется использовать именно их.
	+ В случае, если в процессе предполагается цикличный перебор каких-либо глобальных параметров (например, балансовые единицы, подразделения, контрагент и т.д.), где предполагается обработка отдельного отчета для каждого параметра, то необходимо в таком случае создавать одну общую задачу и несколько привязанных к ней параметров.
	+ В случае, если предполагается, что обработка параметров будет выполняться достаточно долго, то требуется соответственное разбитие их на несколько задач, чтобы обработка могла выполняться несколькими роботами одновременно;
	+ Все глобальные переменные, относящиеся к задаче, должны быть записаны в БД робота и попадать в процесс уже в виде стандартных библиотек (см. п.4.4)



* Не рекомендуется создавать и использовать кастомные библиотеки UiPath по причине того, что могут быть проблемы с зависимостями от пакетов. Вместо этого нужно универсальные скрипты нужно складывать в папку #Universal\_Scripts в корневой папке проекта;
* Все используемые пакеты должны ограничиваться необходимостью процесса, не следует «бездумно» подключать весь имеющийся список UiPath (например, если процесс не работает с форматом PDF, не следует подтягивать этот пакет из менеджера пакета в проект). Все необходимые для работы пакеты должны быть отражены в описании настроек.
* Если же требуется передача одних и тех же данных из одного процесса в другой, то следует использовать БД RPA\_Data;
* Использовать конструкцию Try Catch нужно только в самых исключительных ситуациях. В большинстве случаев ошибки должны отлавливаться управляющим скриптом Main;
* Переменные в скриптах UiPath должны быть инициализированы сразу при объявлении во всех случаях, за исключением тех, когда это невозможно. Переменные необходимо инициализировать следующим образом:

|  |  |
| --- | --- |
| Тип переменной | Первоначальное значение |
| Boolean | False |
| GenericValue | "" |
| String | "" |
| Numeral (int32, int64, double и др.) | 0 |
| DateTime | New System.DateTime |
| Array (of string, of int32 и т.д.) | {"","",…,""} , {0,0,…,0} , и т.д. |
| List | New List(of string) |
| DataTable | New DataTable |
| Dictionary | New Dictionary(of string,string) |
| Browser | New UiPath.Core.Browser |
| Window | New UiPath.Core.Window |
| MailMessage | New System.Net.Mail.MailMessage |

Пример правильной инициализации:



* Переменные в скриптах UiPath должны иметь определенные префиксы. Это позволяет лучше себя контролировать при большом количестве переменных, т.к. иногда после долгой разработки трудно уже вспомнить, для чего переменная была изначально создана.

Можно привести простой пример – допустим, если у нас есть переменная Partners, которая содержит список партнеров. В данном случае непонятно, словарь это или таблица с данными. Может быть даже такая ситуация, что в одном скрипте эта переменная типа DataTable, а в другом – Dictionary (это особенно критично, если над проектом работают несколько человек, и каждый должен использовать эту переменную).

Если же назвать переменную dt\_Partners, то ошибки в дальнейшем быть не может – переменные во всех скриптах должны быть типа DataTable, т.к. само название переменной содержит в себе эту подсказку.

|  |  |  |
| --- | --- | --- |
| Тип переменной | Краткое описание | Префикс |
| Boolean | Хранит значение либо True, либо False | bool\_ |
| GenericValue | Может хранить значение любого типа | gen\_ |
| String | Строковое значение | str\_ |
| Numeral (int16, int32, int64) | Целое число (отрицательное, либо положительное) | int\_ |
| Double | Число с плавающей точкой (дробной частью) | dbl\_ |
| DateTime | Время (год, месяц, неделя, день, час, минута, секунда) | date\_ |
| Array (string[], int32[] и т.д.) | Массив переменных. Используется для простых списков | astr\_ , aint\_ , и т.д. |
| List | Коллекция, используется для динамических списков | list\_ |
| DataTable | Таблица с данными | dt\_ |
| Dictionary | Словарь. Данные в словаре можно получить указывая ключ | dict\_ |
| Browser | Хранит в себе окно браузера для Attach-а | browser\_ |
| Window | Хранит в себе окно приложения для Attach-а | window\_ |
| MailMessage | Хранит в себе письмо со всеми элементами (напр. вложения) | msg\_ |

* При разработке должны максимально использовать встроенные возможности UiPath. Допускается использование VBA/VB/VBS скриптов. Скрипты, написанные на других языках программирования (JS, Python) должны использоваться только в тех случаях, когда их функционал нельзя заменить встроенными возможностями UiPath и платформы .NET, либо их использование оправданно (например, при использовании модулей Machine Learning, специфической работе с браузером, и т.д.);
* Все скрипты (не только UiPath, но и VBA/любые другие) должны содержать поясняющие комментарии. Рекомендуется как добавлять комментарии через кнопку Add Annotation, так и добавлять комментарии через специальный блок Comment (который выделяется на фоне остальных блоков за счет зеленого цвета).
* Папка со скриптами не должна содержать много мусора (старых скриптов, тестовых файлов, и т.д.);
* Блоки активностей в UiPath скриптах должны иметь нумерацию и корректное наименование, т.к. в случае ошибки именно оно будет записано в логи робота;



* Следует очищать скрипты робота от старых и неиспользуемых кусков кода (убирать старые закомментированные активности, блоки Writeline, и т.д.)
	1. Требования к шаблонам и результатам работы роботизируемого процесса
* Шаблоны, которые настраиваются один раз, должны храниться в папке со скриптами и копироваться в папку с результатами по мере выполнения процесса. Сами же результаты должны храниться в папке RPA\_Data, и их не должно быть в папке со скриптами;
* Путь к папкам не должен содержать абсолютные ссылки на папку RPA\_Data. Должна быть возможность оперативно менять папку с результатами работы робота путем изменения параметров в таблице Config
* Если в файле, создаваемом роботом, предполагается сохранение даты в наименовании файла, то рекомендуется писать дату в формате yyyy.MM.dd для того, чтобы эти файлы автоматически сортировались в любой программе;
	1. Требования по работе с БД робота
* Менять структуру БД следует только на среде разработки в БД RPA\_Data. Любое изменение в структуру БД робота (добавление/удаление таблиц, добавление/изменение типов/удаление столбцов, изменение View-таблиц, хранимых процедур и функций) должно быть согласовано с архитектором системы в ИТСК;
* Все изменения структуры БД робота должны попадать в систему ALM ИТСК через надстройку Redgate и переноситься на продуктив с помощью запросов на вытягивание веток Git с веткой master.
* Все разрабатываемые роботы должны максимально применять уже существующую архитектуру БД (включая справочники организаций, систем, статусов роботизации, таблицы с задачами, расписаниями, параметрами расписаний и т.д.)
* Следует применять встроенные в MS SQL механизмы сохранения целостности данных (настраивать связи первичных-вторичных ключей между таблицами, использовать каскадное удаление/изменение, и т.д.);
* В случае, если нет технической возможности настроить каскадное удаление/изменение полей (например, при многократных связях нескольких полей одной таблицы к одному полю другой таблицы), то требуется настроить триггеры для каскадного удаления/изменения;
* Следует индексировать поля, по которым планируется в дальнейшем выполнять поисковые запросы;
* Все поля таблиц должны соответствовать значению, которое хранится в этом поле. Например, дата должна храниться в поле типа DateTime, а сумма – в поле типа Float;
* При создании новых сущностей следует по возможности придавать ID полю определенную кодировку. Например, в таблице Processes используется кодировка из п.2.2;
* При добавлении новых таблиц/полей таблиц/табличных функций/хранимых процедур и view-таблиц, необходимо заполнять поле MS\_Description – это необходимо для автоматического составления описания БД робота;



* В БД робота следует использовать следующие префиксы перед сущностями:
	1. Перед таблицами не нужно ставить префиксы. Единственное – таблицы необходимо называть таким образом, чтобы сохранялась связь между родительской и дочерней сущностью. Например, Acts ->Act\_Interation
	2. Перед View-таблицей ставится префикс V\_<Код процесса>\_. Пример:
		+ - V\_FI01\_ActMapping – данная таблица используется в технических целях только для группы процессов FI-01.
			- V\_ALL\_TaskInfo – данная View-таблица используется в технических целях для всех процессов.
	3. Перед всеми хранимыми процедурами ставится префикс P\_ . После этого применяется та же логика, что и в пунктах 1 и 2. Пример:
		+ P\_FI01\_GetFreeActContracts – данная процедура используется только для группы процессов FI-01.
		+ P\_ALL\_UpdateSystemID – данная процедура используется для всех процессов
	4. Перед скалярными функциями ставится префикс F\_ . После этого применяется та же логика, что и в пунктах 1 и 2.
	5. Перед табличными функциями ставится префикс FT\_ . После этого применяется та же логика, что и в пунктах 1 и 2.
	6. Перед триггерами ставится префикс T\_ . После этого применяется та же логика, что и в пунктах 1 и 2.
	7. Требования по работе с оркестратором
* Нельзя отключать роботов от оркестратора через иконку в трее – оно повлияет на всех разработчиков, работающих на одном сервере;
* В ассетах оркестратора следует хранить только пароли. Все остальные глобальные переменные должны храниться в БД RPA\_Data;
* Кодировка паролей в ассетах оркестратора должна быть следующая:

<Код группы процессов>.<ландшафт>.<Имя системы>

 Пример: FI-01.PROD.1С\_АСБНУ\_Бизнес-сервис

* Шедулер оркестратора конфликтует с другими способами запуска процесса (по письму, по «горячей папке», поэтому вместо него следует использовать шедулер в БД RPA\_Data;
* Для всех роботов в оркестраторе следует настроить запускаемое разрешение экрана 1920x1080;
	1. Требования по работе с Team Foundation Server (TFS)
* Разработка роботов должна вестись в системе контроля версий Git.
* Все скрипты робота должны синхронизироваться через ALM ИТСК: <https://alm-itsk.gazprom-neft.local:8080/TFS/ITSK/RPA_Support/_git/RPA_GPN_Framework>

БД робота должна синхронизироваться через репозиторий <https://alm-itsk.gazprom-neft.local:8080/TFS/ITSK/RPA_Support/_git/RPA_DB>

* Для начала работы с TFS следует изначально клонировать репозиторий в папку D:\RPA\_Configuration\RPA\_GPN\_Framework\<Логин текущего разработчика>
* Недопустимо делать привязку проекта RPA\_GPN\_Framework к TFS из чужой папки. Работать с TFS можно только в своей папке. Что касается репозитория RPA\_DB, то тут ситуация противоположная – всем необходимо работать из одного общего репозитория при работе с определенной веткой.
* Перед началом работы необходимо создать ветку с определенными предполагаемыми доработками. Если процесс большой, то за основу нужно взять ветку с наименованием процесса (например, FI-01.Акты сверки), если же процесс небольшой и разработчик работает один, то за основу следует взять ветку develop.
* Ветки следует называть по маске: <Код процесса>.<Код бизнес-инициативы/код обращения КСУИТ>.<Краткое наименование изменений>
* Во время работы с Git в UiPath нужно тщательно следить за тем, чтобы не была взята в работу чужая ветка. Для этого в UiPath нужно войти в меню управления ветками и выбрать нужную ветку с помощью кнопки Check Out Branch.
* Недопустимо отправлять код UiPath напрямую в ветки test и master. Изменения в эти ветки должны вноситься исключительно через запросы на вытягивание.
* В случае БД для быстрых изменений следует использовать ветку fast-track. Для долгосрочных изменений – ветку develop
* Сразу после релиза необходимо накатывать ветку master обоих репозиториев на ветку test (RPA\_GPN\_Framework) и ветку develop (RPA\_DB)
* Нельзя сливать исходную ветку разработки роботов с веткой test. Необходимо сначала сделать ее копию, а затем уже сливать копию ветки с веткой test.
* Следует писать корректный комментарий во время возвращения скрипта в TFS. Комментарий должен начинаться с префикса процесса, к которым относятся загружаемые изменения. Например, комментарий: «FI-01. Добавлен механизм сверки актов, внесены изменения в блок “3.Обработка договоров” скрипта ActsCompare. »
* При слиянии веток с веткой release должен быть актуализирован документ «Описание настроек робота» по текущему процессу.
* При слиянии веток веткой release с помощью запросов на вытягивание должны быть подробно описаны изменения, которые вносились в скрипты.
* При слиянии веток в запросах на вытягивание обязательно должна быть ссылка на задачу, в рамках которой выполнялись изменения.
	1. Требования к работе с прикладными системами
* SAP ERP:
	+ Требуется включить скриптинг в настройках SAP Logon следующим образом:



* + Требуется отключить уведомления о всплывающих окнах при выгрузке файлов из SAP:



* + При разработке нельзя ставить галку «Больше не спрашивать» при появлении окна о выборе формата выгружаемого файла. Это приведет к тому, что при добавлении новых учетных записей эти настройки не будут учтены, что приведет к ошибкам.
	+ При разработке робота для работы с SAP ERP требуется использовать SAP-скрипты вместо работы через селекторы UiPath (там, где это возможно)
* 1C:
	+ При разработке следует использовать новый тип селекторов в UiExplorer – UiAutomation – только они хорошо работают с 1С;
	+ При разработке нельзя разрабатывать внешние обработки 1С, или использовать COM-подключение к 1С (запрещено службой ИБ);
	1. Требования к действиям, выполняемым роботом
* В роботе недопустимо использовать стандартную UiPath Activity под названием Kill Process, т.к. он влияет на всех роботов, работающих на одной машине. Вместо этого следует использовать скрипт KillProcesses, который расположен в папке Universal Scripts
* Недопустимо, чтобы возникала ситуация, когда несколько роботов могут предпринимать попытку обращения к одному и тому же файлу (например, таблице Excel или файлу Word). Для этого необходимо либо сохранять файл в личную техническую папку робота, либо использовать БД робота;
* Нельзя, чтобы робот в процессе работы менял глобальные переменные, которые записаны в таблице Config. Также нельзя, чтобы он менял какие-либо переменные, которые используются в Main, иначе это может привести к ошибке в самом Main.
* Также нельзя, чтобы робот в процессе своей работы вносил серьезные изменения в справочники в БД. Например, запрещено вносить изменения в таблицы AffilatedOrganizations, Processes, Credentials, AffOrg\_RobotizationStatuses.
* Нельзя, чтобы робот в процессе своей работы удалял какие-либо базовые папки с результатами работы робота или со скриптами;
* Недопустимо запускать скрипты, находящиеся не в текущей папке проекта (т.е. которые находятся в папке, уровень который выше той папки, на которой находится Main.xaml и project.json

Приложение 1. Шаблон опросного листа для анализа бизнес-процессов

