Система онлайн-тестирования Maintest®

# MAINTEST
# ОПИСАНИЕ ИНТЕРФЕЙСА API

Дата выпуска: 28.09.2021

HT Lab (ООО «ЭйчТи Лаб»)
https://ht-lab.ru

# Содержание

# 1. ВВЕДЕНИЕ

Документ описывает прикладной программный интерфейс системы онлайн-тестирования Maintest.

Документ предназначен для разработчиков программного обеспечения, реализующих интеграцию психометрических тестов Maintest в информационную систему заказчика. Для успешной работы с документом разработчики должны быть знакомы в той или иной степени с такими технологиями, как WebServices, WSDL, XML, SOAP.

# 2. ПОДКЛЮЧЕНИЕ К ИНТЕРФЕЙСУ

## 2.1. Условия подключения к интерфейсу

Для работы с системой Maintest через интерфейс API необходимо выполнение следующих условий:

- наличие учетной записи (личного кабинета) в системе Maintest;
- для личного кабинета должна быть подключена услуга доступа к API;
- в личном кабинете должен быть подключен сервис онлайн-тестирования.

Для возможности использования API также необходимо наличие среды разработки или языка программирования, в которых реализована поддержка работы с веб-сервисами или протоколом SOAP.

Для быстрого подключения к интерфейсу рекомендуется использовать файл спецификации интерфейса в формате WSDL, который доступен на сервере Maintest.

## 2.2. Спецификация интерфейса WSDL

В соответствии с правилами реализации веб-сервисов, структура интерфейса полностью описана при помощи спецификации в формате WSDL-документа. Спецификация текущей версии интерфейса в виде WSDL-документа доступна по следующим адресам (в зависимости от того, какой сервис является приоритетным в личном кабинете клиента):

https://client.ht-line.ru/maintest-5i/api/1.10.0.0/wsdl
https://client.ht-line.ru/m-tests/api/1.10.0.0/wsdl[1]

Как правило, наличие WSDL-спецификации позволяет разработчику автоматически сгенерировать описание класса SOAP-клиента для доступа к серверу с API, однако, такая возможность зависит от используемой среды разработки и языка программирования. Вообще, наличие WSDL-спецификации не является обязательным условием, но её использование обычно помогает упростить и ускорить работу с API.

[1] Версия в ссылке может отличаться. Актуальная направляется разработчиками Maintest вместе с параметрами доступа.

## 2.3. Авторизация пользователя в интерфейсе

При работе с интерфейсом для проверки прав доступа используется процедура авторизации пользователя. В качестве параметров авторизации используются логин и пароль пользователя для доступа в личный кабинет системы Maintest.

Процедура авторизации основана на механизме HTTP-авторизации, при этом параметры авторизации (логин и пароль к личному кабинету) передаются при вызове каждого метода интерфейса.

## 2.4. Интерактивный отладчик

Для удобства первичного ознакомления и дальнейшей работы разработчика с интерфейсом API, ему предоставляется доступ в отладочный веб-интерфейс, который позволяет в интерактивном режиме выполнять методы интерфейса API.

Данный инструмент может быть полезен при изучении методов API, создания сценариев использования API и при отладке собственного приложения, работающего с API.

Отладочный веб-интерфейс доступен по следующим адресам (в зависимости от того, как сервис является приоритетным в личном кабинете клиента):

https://client.ht-line.ru/maintest-5i/api/1.10.0.0/guide
https://client.ht-line.ru/m-tests/api/1.10.0.0/guide

Как и для доступа к API, для доступа к отладочному веб-интерфейсу необходимо пройти процедуру авторизации, используя логин и пароль для доступа к личному кабинету Maintest.

## 2.5. Пример запроса к интерфейсу

В качестве простого примера использования API рассмотрим вызов метода **getApiVersionNumber()**, который возвращает номер текущей версии интерфейса. В качестве языка программирования в примере используется PHP.

Пример 1. Запрос номера текущей версии API.

```
// Настройки подключения SOAP-клиента
$options = array(
      "encoding"  => "utf-8",
      "login"     => "user",
      "password"  => "pass"
);
```

```
// Для эффективности используем локальный WSDL-файл
$wsdlpath = "maintest.wsdl";

// Создаем объект SOAP-клиента
$SoapClient = new SoapClient($wsdlpath,$options);

// Выполняемзапросномераверсии
$SoapResponse = $SoapClient->getApiVersionNumber();
$VersionNumber = $SoapResponse["VersionNumber"];

// Выводимнаэкранномерверсии
echo "Maintest API version number: " . $VersionNumber;
```

В примере используется локально сохраненная копия файла WSDL-спецификации интерфейса. Это позволяет упростить работу с интерфейсом, так как в файле WSDL содержится вся необходимая информация для работы SOAP-клиента. В том случае, если WSDL не используется, нужно в массиве options заполнить такие поля, как location и uri, содержащие ссылки на адрес интерфейса и пространство имен.

Более подробную информацию об использовании SOAP-клиента и WSDL-спецификации Вы можете найти в документации к своей среде разработке, языку программирования или библиотеке, используемой для работы с SOAP.

# 3. ТИПОВЫЕ СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

Интерфейс обладает достаточной гибкостью и позволяет реализовывать различные сценарии работы, в зависимости от задач и возможностей заказчика.

## 3.1. Краткое описание сценариев использования

Типовыми сценариями использования интерфейса для интеграции тестов в информационную систему заказчика являются следующие сценарии:

- Сценарий 1: Стандартная интеграция
  (тестирование и генерация отчетов в системе Maintest).
- Сценарий 2: Расширенная интеграция
  (тестирование в системе заказчика, генерация отчетов - в Maintest).
- Сценарий 3: Специальная интеграция
  (тестирование и генерация отчетов в системе заказчика).

С учетом универсальности интерфейса кроме типовых сценариев интеграции возможны также альтернативные сценарии, рассчитанные под потребности конкретного заказчика.

Основные различия в распределении функций между системой Maintest и ИС заказчика при различных сценариях интеграции представлены в таблице ниже.

| **Распределение функций при различных сценариях интеграции** | **Стандартная интеграция** | **Расширенная интеграция** | **Специальная интеграция** |
|---|---|---|---|
| Предъявление тестовых заданий (тестовый плеер) | Maintest (через IFRAME) | ИС заказчика (свой плеер) | ИС заказчика (свой плеер) |
| Обработка протокола тестирования (ответов участника на вопросы теста) | Maintest | Maintest | Maintest |
| Генерация HTML-страниц отчетов с результатами тестирования | Maintest (через IFRAME) | Maintest (через IFRAME) | ИС заказчика |
| Генерация документов отчетов с результатами тестирования в форматах HTML/DOC/PDF | Maintest | Maintest или ИС заказчика | ИС заказчика |

### Сценарий 1: Стандартная интеграция.

В этом сценарии система Maintest выполняет максимальное число функций:

- реализует режим тестирования при помощи своего тестового плеера;
- сохраняет и обрабатывает полученные протоколы тестирования;
- генерирует индивидуальные отчеты по результатам тестирования.

Информационная система заказчика может работать с системой Maintest, используя фрейм (IFRAME) для загрузки тестового плеера и отчетов с результатами тестирования на страницы, загружаемые пользователями системы.

Архитектура тестового плеера позволяет кастомизировать внешний вид интерфейса. Если заказчик желает кастомизировать предъявляемый тестовый плеер, он должен предоставить макет интерфейса разработчикам Maintest. Соответствующие работы закладываются в смету проекта.

Сценарий удобен для быстрого встраивания тестов в корпоративный сайт, систему дистанционного обучения или кадровую информационную систему заказчика. Этот сценарий будет подробно рассмотрен в данном руководстве далее.

**Сценарий 2: Расширенная интеграция**

В отличие от первого типового сценария в этом сценарии предъявление теста респонденту происходит через тестовый плеер информационной системы заказчика. Система Maintest выполняет роль обработчика протоколов тестирования и генератора отчетов.

Такой подход позволяет унифицировать пользовательский веб-интерфейс, с которым работает респондент (ему показывается привычный для него тестовый плеер той системы, с которой он обычно работает), что упрощает работу с системой для конечного пользователя.

Данный сценарий используется для интеграции с такими информационными системами заказчика, как системы дистанционного обучения или системы тестирования, которые имеют развитый функционал предъявления тестовых заданий (развитый тестовый плеер). При этом тестовый плеер системы заказчика должен быть совместим с типами тестовых заданий и правилами их предъявления, используемыми в тестах системы Maintest.

Данные работы требуют существенных затрат, как со стороны разработчиков системы Maintest, так и со стороны разработчиков информационной системы, с которой осуществляется интеграция и требуют согласования дополнительной проектной документации и должны учитываться в смете проекта.

### Сценарий 3: Специальная интеграция

В отличие от первого и второго типовых сценариев в этом сценарии у системы Maintest остаётся минимальный функционал, который заключается в обработке протоколов тестирования и генерации структуры отчета.

Информационная система заказчика при таком подходе берет на себя реализацию функций предъявления теста и функций генерации отчетов, а также, возможно, функции использования числовых данных, полученных по результатам тестирования (в качестве которых могут выступать, например, оценки по компетенциям).

Этот сценарий является наиболее сложным в реализации, однако позволяет наиболее гибко встроить тестовые возможности системы Maintest в собственную информационную систему заказчика, так как внешний вид и процедуры тестирования, и отчетов по результатам тестирования, полностью определяются правилами информационной системы заказчика.

## 3.2. Описание сценария стандартной интеграции

Рассмотрим более подробно сценарий стандартной интеграции. Этот сценарий является единственным доступным для большинства Заказчиков, в связи с тем, что второй и третий сценарии требуют трудоемкой реализации в информационной системе заказчика механизма тестового плеера, удовлетворяющего структуре и логике тестовых методик системы Maintest. Второй и третий сценарии, как правило, используются только в случае интеграции с системами дистанционного обучения и системами тестирования, имеющими развитые механизмы предъявления тестовых заданий.

### Обобщенный алгоритм работы

Основная логика работы сценария выглядит следующим образом:

1) Создание нового сеанса тестирования и получение ссылки на сеанс: Метод createTestingSessionEx().

2) Загрузка полученной ссылки во фрейм на странице в информационной системе заказчика.

3) Информирование об окончании сеанса тестирования:
   Автоматическое оповещение по URL, указанному в п.1, или метод isTestingSessionComplete().

4) По окончании тестирования – получение ссылки на результат:
   Метод getResultsReportUrl().

5) Загрузка полученной ссылки во фрейм на странице в информационной системе заказчика.

6) При необходимости – получение результатов в других форматах:
   Методы: getResultsReportHtml(), getResultsReportHtmlEx(), getResultsReportPdf(), getResultsReportPdfEx(), getResultsReportXml() и getResultsValuesList().

7) Сохранение полученных результатов в других форматах в системе.

## Параметры вызова методов

Метод createTestingSessionEx()

Параметры вызова:

- TestName – системное имя теста.
  Это идентификатор теста в каталоге тестов. Список доступных в каталоге тестов со всеми их атрибутами можно заранее получить при помощи метода getTestsAttribsList().

- ExternalSessionGuid – идентификатор сеанса тестирования.
  Это произвольный уникальный идентификатор сеанса тестирования, определяемый на стороне удаленной системы. Этот идентификатор в дальнейшем используется почти во всех последующих запросах к интерфейсу.

- IsAnonymousSession – признак анонимного сеанса.
  В случае установки значения параметра в 1, тестовый плеер не выводит анкету с личными данными, которая обычно предъявляется в начале тестирования, и тестирование проходит без передачи персональных данных респондента в систему Maintest.

- CallbackUrl – URL для оповещения об окончании сеанса тестирования.
  Подробнее – в описании метода createTestingSessionEx() и механизма callback.

Для большинства остальных методов основным параметром является параметр ExternalSessionGuid – использованный при создании сеанса ранее уникальный идентификатор сеанса тестирования.

## Типовой алгоритм

Рассмотрим типовой алгоритм интеграции Портала X (далее «портал») с системой Maintest.

Для любой системы Заказчика система Maintest представляет собой совокупность контента, тестового плеера и базы данных с результатами тестирования (рисунок 3.1).

Контент представляет собой саму тестовую методику, алгоритмы расчета шкал и показателей этой методики, словесные интерпретации результатов тестирования.

Тестовый плеер – это интерфейс предъявления заданий теста. Респондент в тестовом плеере отвечает на вопросы, и его результаты сохраняются в БД Maintest.

БД результатов хранит данные о сеансе тестирования, самом тестировании, а также рассчитанные результаты в различных видах (сырой балл, бал в процентах и т.д.).

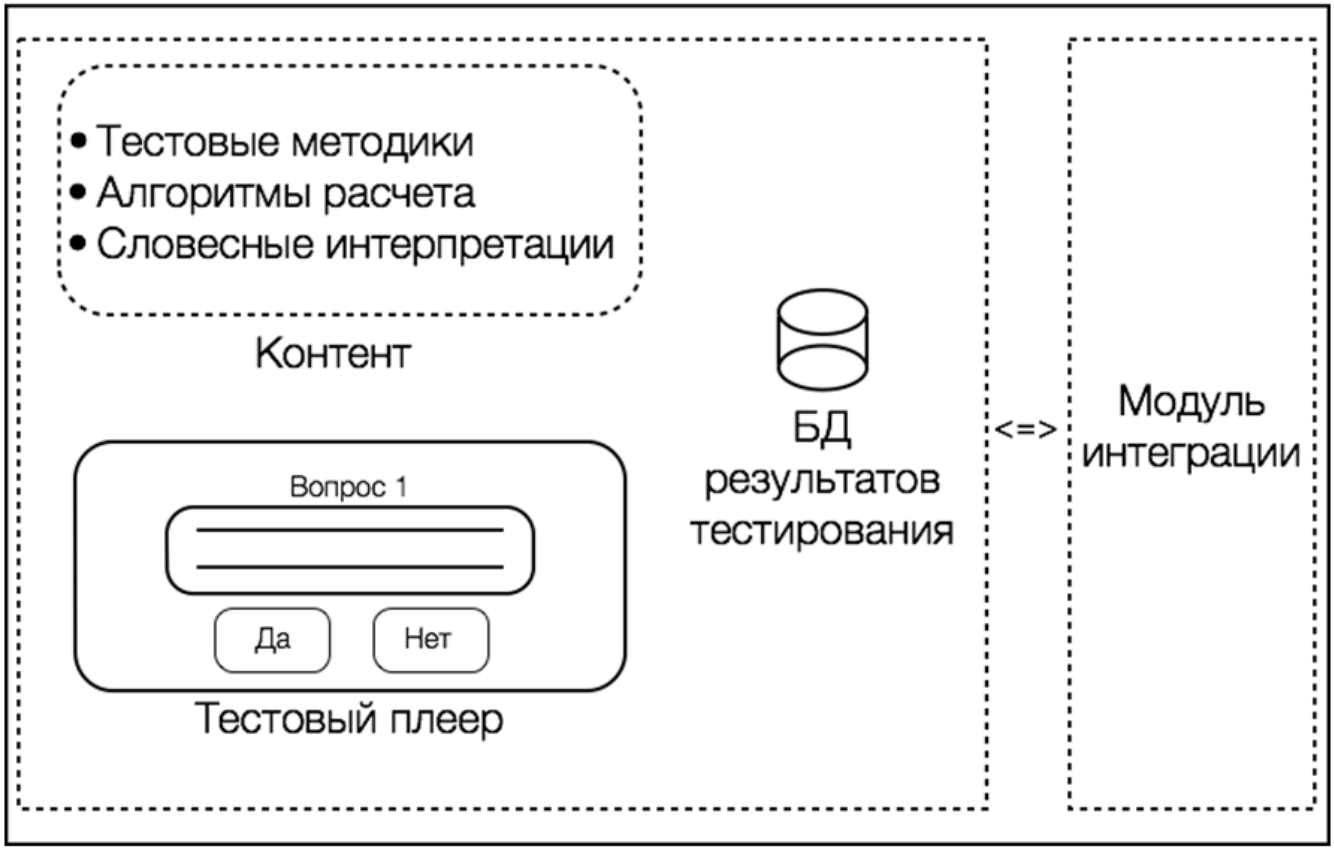


Рисунок 3.1.

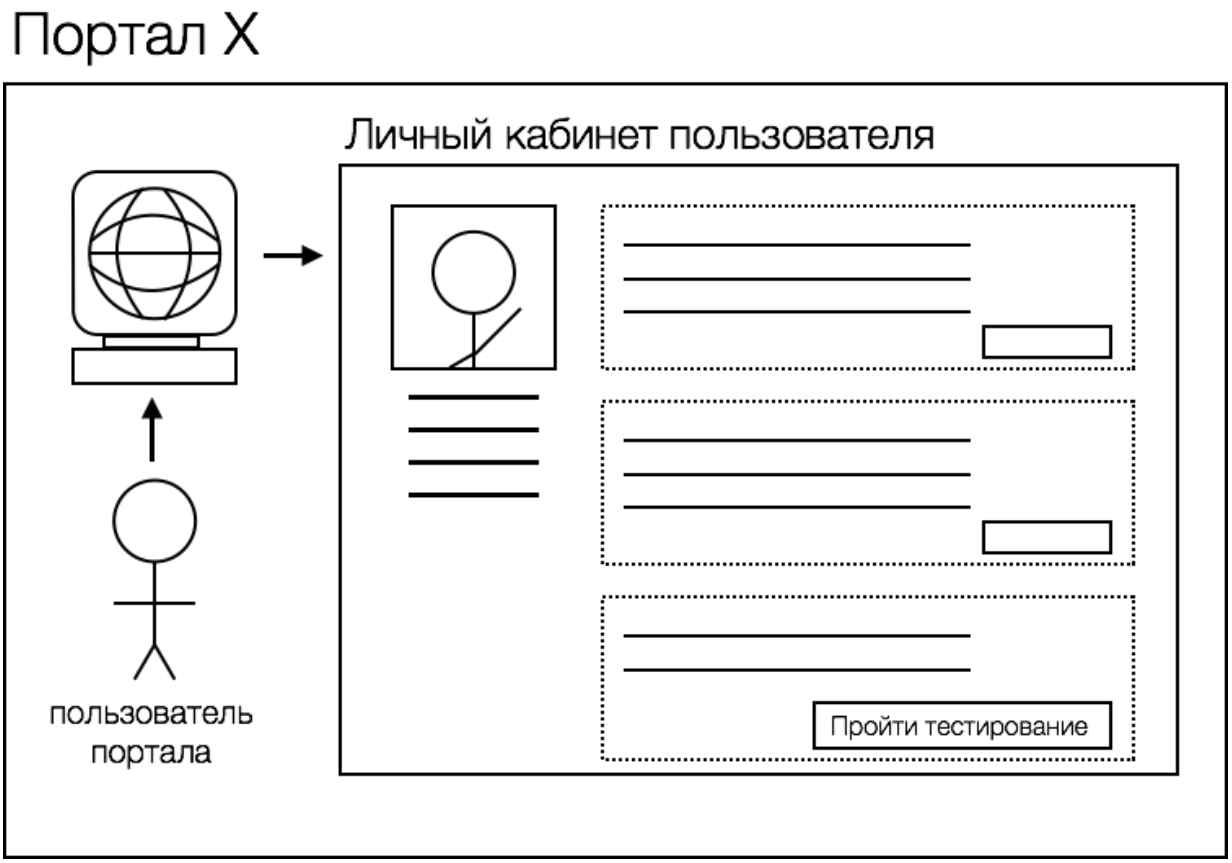


Рисунок 3.2.

Предположим, что портал для конечного пользователя (кто и будет в дальнейшем проходить тестирование) представляет собой личный кабинет. Помимо другого функционала личного кабинета на портале, у этого пользователя есть возможность пройти тестирование. В примере, изображенном на рисунке 3.2, данный функционал представлен кнопкой «Пройти тестирование», нажав на которую пользователь перейдет на страницу с тестовым плеером.

В таком случае типовой алгоритм взаимодействия Портала (П) с системой Maintest (HTL) будет выглядеть следующим образом:

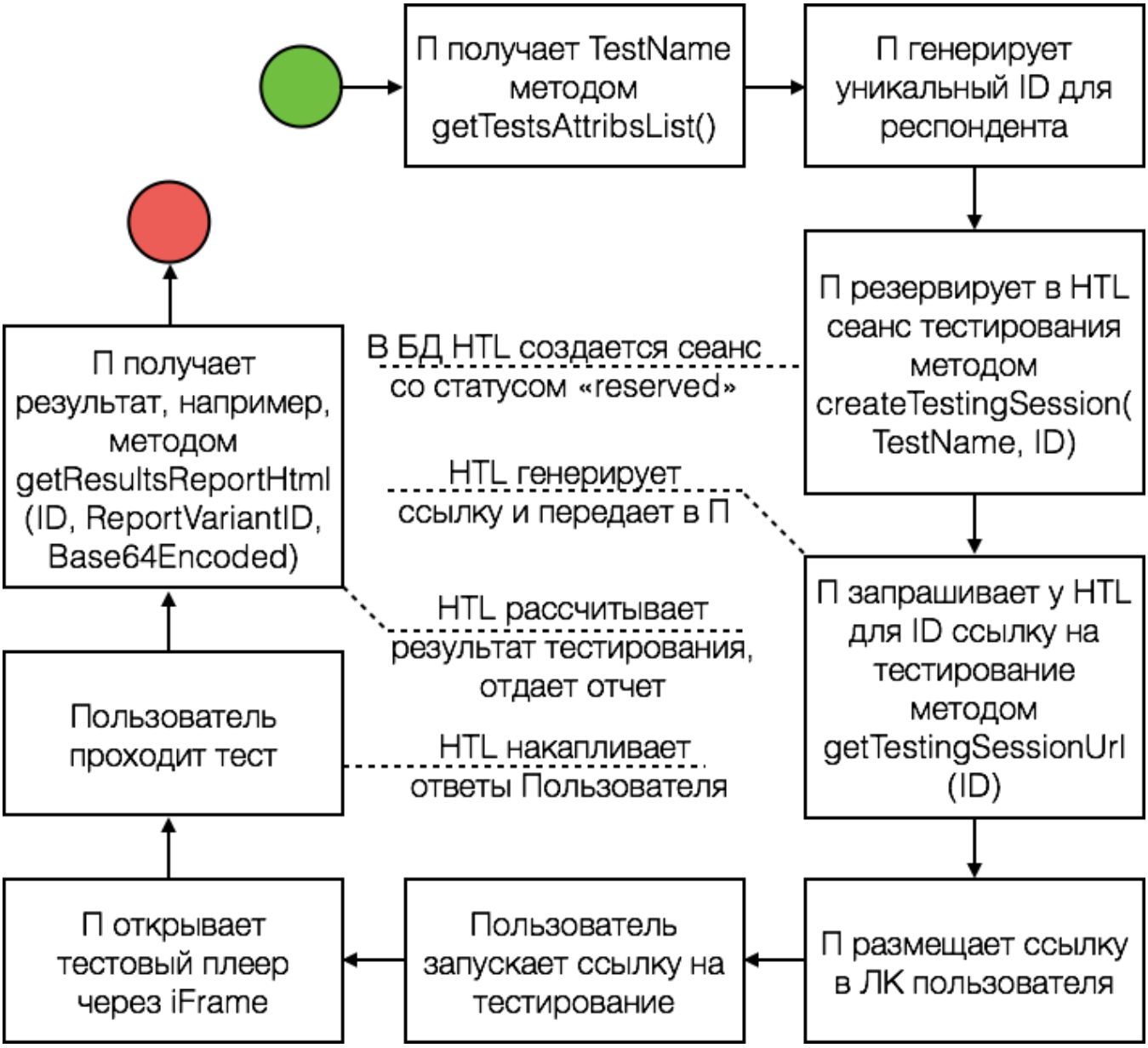
П получает TestName методом getTestsAttribsList()
П генерирует уникальный ID для респондента
П резервирует в HTL сеанс тестирования методом createTestingSession( TestName, ID)
В БД HTL создается сеанс со статусом «reserved»
HTL генерирует ссылку и передает в П
П запрашивает у HTL для ID ссылку на тестирование методом getTestingSessionUrl (ID)
HTL рассчитывает результат тестирования, отдает отчет
HTL накапливает ответы Пользователя
П получает результат, например, методом getResultsReportHtml (ID, ReportVariantID, Base64Encoded)
Пользователь проходит тест
П размещает ссылку в ЛК пользователя
Пользователь запускает ссылку на тестирование
П открывает тестовый плеер через iFrame

Рисунок 3.3.

# 4. СПИСОК ОСНОВНЫХ МЕТОДОВ ИНТЕРФЕЙСА

В данном руководств представлены только основные методы, которые потребуются Заказчику для выполнения стандартной интеграции. Весь список методов доступен в интерактивном отладчике.

## 4.1. Метод getTestsAttribsList()

Получение списка атрибутов для всех тестов.

Входные параметры:отсутствуют

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| TestsAttribsList | TypeTestsAttribsList | список атрибутов доступных тестов |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

## 4.2. Метод createTestingSessionEx

Создание сеанса тестирования с заданным идентификатором сеанса, получение ссылки на тестирование и регистрация URL для информирования о завершении прохождения сеанса тестирования.

Входные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| TestName | string | системное имя теста |
| ExternalSessionGuid | string | уникальный идентификатор сеанса тестирования (генерируется на стороне клиента) |
| IsAnonymousSession | Integer | параметр запроса личных данных в начале тестирования (0 – не запрашивать, 1 - запрашивать) |
| CallbackUrl | string | URL, на который будет отправлен POST-запрос по окончании тестирования (необязательный параметр) |

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ExternalSessionGuid | string | уникальный идентификатор созданного сеанса тестирования |
| TestingSessionUrl | string | адрес страницы сеанса тестирования |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

Формат POST-запроса, отправляемого по CallbackUrl:

| | | |
|---|---|---|
| Passed | integer | индикатор прохождения тестирования |
| ExternalSessionGuid | string | уникальный идентификатор созданного сеанса тестирования |

### 4.3. Метод getTestingSessionUrlEx()

Получение адреса страницы для сеанса тестирования (с учетом дополнительных параметров).

Входные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |
| IsCompactMode | integer | параметр компактного режима тестирования (0 - нет, 1 - да) |
| FrameWidth | integer | ширина фрейма тестирования в пикселях (0 - по умолчанию) |
| FrameHeight | integer | высота фрейма тестирования в пикселях (0 - по умолчанию) |

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| TestingSessionUrl | string | адрес страницы сеанса тестирования |

| | | |
|---|---|---|
| OperationStatus | TypeOperationStatus | результат выполнения операции |

## 4.4. Метод isTestingSessionComplete()

Проверка состояния завершенности сеанса тестирования*.

Входные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |

Выходные параметры

| Параметр | Тип | Описание |
|---|---|---|
| IsSessionComplete | boolean | признак завершенности сеанса тестирования |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

**Механизм CallBack**

Существует более эффективная альтернатива периодическому использованию метода isTestingSessionComplete().

По умолчанию для контроля текущего состояния сеанса тестирования нужно периодически вызывать метод API isTestingSessionComplete(), который возвращает текущий статус сеанса тестирования - "завершен" или "не завершен". Однако, при большом числе одновременных сеансов тестирования данный подход может повлечь за собой неоправданную избыточную нагрузку, как на сервер, так и на клиента API.

Для того чтобы оптимизировать решение задачи уведомления клиента API о завершении сеанса тестирования, на сервере API реализован механизм колбека - обратного вызова, который передается клиентской системе по окончании сеанса тестирования.

Механизм колбека позволяет настроить автоматический обратный вызов метода в информационной системе клиента, который будет вызываться каждый раз при завершении сеанса тестирования.

Базовая версия колбека реализована в методе createTestingSessionEx. При создании сеанса в метод передается URL, на который по завершении сеанса тестирования будет отправлен POST-запрос с указанием идентификатора сеанса ExternalSessionGuid.

Также возможна более гибкая настройка обратного вызова, которая потребует участия разработчиков системы Maintest. В этом случае нет жестких ограничений на тип и состав обратного вызова – может быть передан произвольный набор

параметров, могут быть использованы как вызовы HTTP и SOAP, так и более специфические варианты (это зависит от особенностей информационной системы клиента, которая интегрируется с системой Maintest). Для реализации такого сценария разработчики клиента API должны обратиться к разработчикам Maintest и передать информацию о методе обратного вызова (HTTP, SOAP и т.д.), адресе (URI) и списке передаваемых параметров (по умолчанию - ExternalSessionGuid и подпись). В ответ на такое обращение разработчики системы Maintest реализуют и настроят соответствующий обратный вызов в системе Maintest.

### 4.4.1. PostMessage после завершения тестирования в iFrame

Если сеанс был запущен в iFrame, то после завершения тестирования в родительское окно будет отправлено сообщение «completed» с помощью метода postMessage().

## 4.5. Метод getResultsReportUrl()

Получение адреса страницы отчета по результатам тестирования.

Входные параметры:

| **Параметр** | **Тип** | **Описание** |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |

Выходные параметры:

| **Параметр** | **Тип** | **Описание** |
|---|---|---|
| ResultsReportUrl | string | адрес страницы отчета с результатами тестирования |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

Примечание. Метод отдает адрес страницы для варианта отчета по умолчанию - как правило, в качестве такого варианта выступает отчет для респондента.

*Внимание! По прохождении тестирования в системе Maintest формируется отчет. Отчет может быть в нескольких вариантах. Например «Отчет для респондента» и «Отчет для руководителя». Контент отчетов разных вариантов отличается. Для каждого вариант отчета есть свой идентификатор ReportVariantId.*

## 4.6. Метод getResultsReportXml()

Получение структурированного контента отчета в формате XML.

*Внимание! В случае, когда отчет генерируется средствами ИС заказчика, то метод getResultsReportXML() является самым удобным в использовании. Позволяет одним вызовом получить контент всех вариантов отчетов в удобном для парсинга XML формате.*

Входные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |
| ReportVariantId | integer | идентификатор варианта отчета (0 – получить все варианты отчетов) |
| Base64Encoded | integer | признак кодирования контента в кодировке Base64 (0 - нет, 1 - да) |

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ReportContentXml | string | структурированный контент отчета в формате XML |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

## 4.7. Метод getResultsReportHtml()

Получение контента отчета в формате HTML.

Входные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |
| ReportVariantId | integer | идентификатор варианта отчета (0 - вариант по умолчанию) |
| Base64Encoded | integer | признак кодирования контента в кодировке Base64 (0 - нет, 1 - да) |

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|

| | | |
|---|---|---|
| ReportContentHtml | string | контент отчета в формате HTML |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

## 4.8. Метод getResultsReportDocx()

Получение контента отчета в формате DOCX.

Входные параметры:

| **Параметр** | **Тип** | **Описание** |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |
| ReportVariantId | integer | идентификатор варианта отчета (0 - вариант по умолчанию) |
| Base64Encoded | integer | признак кодирования контента в кодировке Base64 (0 - нет, 1 - да)* |

*Метод работает только со значением параметра Base64Encoded = 1.

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ReportContentDocx | string | контент отчета в формате DOCX |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

## 4.9. Метод getResultsValuesList()

Запрос списка значений результатов тестирования.
Входные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ExternalSessionGuid | string | идентификатор созданного ранее сеанса тестирования |
| ReportVariantId | integer | идентификатор варианта отчета (0 – получить список значений по всем шкалам) |
| ValuesMetrics | string | метрикадляScaleValue ("sten", "source", "percent", "hit", "iq", "custom"; поумолчанию "sten" |

Выходные параметры:

| Параметр | Тип | Описание |
|---|---|---|
| ResultsValuesList | TypeResultsValuesList | список значений результатов тестирования по шкалам теста |
| OperationStatus | TypeOperationStatus | результат выполнения операции |

# 5. СЛОЖНЫЕ ТИПЫ ДАННЫХ

В методах интерфейса используются как простые, так и сложные (комплексные) типы данных. Как правило, в качестве аргументов методов используются простые типы данных (строки и целые числа), а в качестве возвращаемых результатов – сложные (структуры и массивы).

Например, результат выполнения любого метода интерфейса содержит структуру данных OperationStatus сложного типа TypeOperationStatus, которая содержит такие поля, как числовой код результата выполнения метода (номер ошибки), мнемонический код результата (мнемокод ошибки) и текстовое описание результата (текст сообщения об ошибке). Значение полей этой структуры можно использовать для протоколирования номера ошибки и для вывода пользователю системы текста сообщения о возникшей ошибке.

## 5.1. Тип данных TypeOperationStatus

Этот тип данных предназначен для описания результата выполнения метода интерфейса. В случае, если произошла ошибка при выполнении метода, в структуре данных этого типа (OperationStatus) содержится числовое и текстовое описание ошибки.

Описаниетипа:

```
<xsd:complexType name="TypeOperationStatus">
<xsd:sequence>
<xsd:elementminOccurs="1" maxOccurs="1" name="ErrorNumber" type="xsd:int"/>
<xsd:elementminOccurs="1" maxOccurs="1" name="ErrorCode" type="xsd:string"/>
<xsd:elementminOccurs="1" maxOccurs="1" name="ErrorMessage"
type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- ErrorNumber - числовой код выполнения операции (номер ошибки);
- ErrorCode - мнемонический код выполнения операции (мнемокод ошибки);
- ErrorMessage - текстовое описание ошибки.

В случае отсутствия ошибок при выполнении метода номер ошибки равен нулю (ErrorNumber=0), а мнемокод ошибки содержит строку "OK" (ErrorCode="OK").

## 5.2. Тип данных TypeTestAttribs

Этот тип данных предназначен для описания атрибутов одного из тестов из каталога тестов. В полях этой структуры описаны все атрибуты теста, которые могут потребоваться при работе с интерфейсом: идентификатор теста, название теста, статус теста, число доступных лицензий и число собранных ранее протоколов тестирования.

Описаниетипа:

```
<xsd:complexType name="TypeTestAttribs">
<xsd:sequence>
<xsd:element name="TestId" type="xsd:int"/>
<xsd:element name="TestUid" type="xsd:string"/>
<xsd:element name="TestName" type="xsd:string"/>
<xsd:element name="MaintestName" type="xsd:string"/>
<xsd:element name="TestTitle" type="xsd:string"/>
<xsd:element name="TestEnabled" type="xsd:boolean"/>
<xsd:element name="IsLibraryTest" type="xsd:boolean"/>
<xsd:element name="QuestionsCount" type="xsd:int"/>
<xsd:element name="LicensesCount" type="xsd:int"/>
<xsd:element name="ProtocolsCount" type="xsd:int"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- TestId - уникальный числовой идентификатор экземпляра теста;
- TestUid - уникальный строковый идентификатор экземпляра теста;
- TestName - системное имя теста;
- TestTitle - название теста;
- TestEnabled - статус теста (0 - отключен, 1 - включен);
- IsLibraryTest - принадлежит ли тест сервису Maintest-5i (0 – тест не является одним из тестов Maintest-5i, 1 - является);
- MaintestName - системное имя теста в формате Maintest-4;
- QuestionsCount – число вопросов в тесте;
- LicensesCount - число доступных для использования лицензий;
- ProtocolsCount - число накопленных протоколов тестирования.

## 5.3. Тип данных TypeTestsAttribsList

Этот тип предназначен для описания атрибутов сразу всех доступных для пользователя тестов из каталога тестов. Он содержит массив структур типа TypeTestAttribs, описанных выше.

Описание типа:

```
    <xsd:complexType name="TypeTestsAttribsList">
    <xsd:sequence>
    <xsd:element name="TestAttribs" type="tns:TypeTestAttribs"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
```

Назначение полей:

- TestAttribs - атрибуты теста в виде структуры типа TypeTestAttribs.

## 5.4. Тип данных TypeTestScale

Этот тип данных предназначен для описания одной из измерительных шкал теста. В полях этой структуры содержится название измерительной шкалы, а также такие ее атрибуты, как тип шкалы, число полюсов и порядковый номер шкалы в тесте.

Описаниетипа:

```
<xsd:complexType name="TypeTestScale">
<xsd:sequence>
<xsd:element name="ScaleNumber" type="xsd:int"/>
<xsd:element name="ScaleTitle" type="xsd:string"/>
<xsd:element name="ScaleType" type="xsd:string"/>
<xsd:element name="ScalePoles" type="xsd:int"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- ScaleNumber - порядковый номер шкалы в тесте;
- ScaleTitle - наименование измерительной шкалы;
- ScaleType - мнемокод типа измерительной шкалы (внутренне понятие системы Maintest);
- ScalePoles - число полюсов шкалы (1 или 2).

## 5.5. Тип данных TypeTestsScalesList

Этот тип данных предназначен для описания полного списка измерительных шкал теста. Он содержит массив структур типа TypeTestScale, описанных выше.

Описание типа:

```
    <xsd:complexType name="TypeTestsScalesList">
    <xsd:sequence>
    <xsd:element name="TestScale" type="tns:TypeTestScale"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
```

Назначение полей:

- TestScale - описания шкалы теста в виде структуры типа TypeTestScale.

## 5.6. Тип данных TypeReportVariant

Этот тип предназначен для описания атрибутов варианта отчета теста.

Описаниетипа:

```
<xsd:complexType name="TypeReportVariant">
<xsd:sequence>
<xsd:element name="VariantId" type="xsd:int"/>
<xsd:element name="VariantName" type="xsd:string"/>
<xsd:element name="VariantTitle" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- VariantId - номер варианта отчета;
- VariantName - системное имя варианта отчета;
- VariantTitle - название варианта отчета.

## 5.7. Тип данных TypeReportVariantsList

Этот тип данных предназначен для описания полного списка вариантов отчета теста. Он содержит массив структур типа TypeReportVariant, описанных выше.

Описание типа:

```
    <xsd:complexType name="TypeReportVariantsList">
    <xsd:sequence>
    <xsd:element name="ReportVariant" type="tns:TypeReportVariant"
minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
```

Назначение полей:

- ReportVariant - описание варианта в виде структуры типа TypeReportVariant.

## 5.8. Тип данных TypeResponderProtocol

Этот тип содержит данные о протоколе и респонденте: идентификатор протокола, имя и название теста, данные анкеты респондента.

Описаниетипа:

```
<xsd:complexType name="TypeResponderProtocol">
<xsd:sequence>
<xsd:element name="ProtocolId" type="xsd:string"/>
<xsd:element name="TestName" type="xsd:string"/>
<xsd:element name="TestTitle" type="xsd:string"/>
<xsd:element name="FirstName" type="xsd:string"/>
<xsd:element name="MiddleName" type="xsd:string"/>
<xsd:element name="LastName" type="xsd:string"/>
<xsd:element name="Name" type="xsd:string"/>
<xsd:element name="Age" type="xsd:string"/>
<xsd:element name="BirthDate" type="xsd:string"/>
<xsd:element name="Gender" type="xsd:string"/>
<xsd:element name="Email" type="xsd:string"/>
<xsd:element name="Phone" type="xsd:string"/>
<xsd:element name="Field1" type="xsd:string"/>
<xsd:element name="Field2" type="xsd:string"/>
<xsd:element name="Field3" type="xsd:string"/>
<xsd:element name="Field4" type="xsd:string"/>
<xsd:element name="Field5" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- ProtocolId – уникальный числовой идентификатор протокола.
- TestName – идентификатор теста.
- TestTitle – название теста
- FirstName – имя респондента.
- MiddleName – отчество респондента.
- LastName – фамилия респондента.
- Name – краткое имя респондента.
- Age – возраст респондента
- BirthDate – дата рождения респондента в формате ГГГГ-ММ-ДД
- Gender – пол респондента: m – мужской, f – женский.
- Email – emailреспондента
- Phone – телефон респондента
- Field1…Field5 – настраиваемые поля анкеты респондента

## 5.9. Тип данных TypeResponderProtocolsList

Этот тип данных предназначен для описания полного списка данных о протоколе и респонденте. Он содержит массив структур типа TypeResponderProtocol,описанных выше.

Описание типа:

```
<xsd:complexType name="TypeResponderProtocolsList">
<xsd:sequence>
<xsd:element name="ResponderProtocolsList" type="tns:TypeResponderProtocol"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
```

Назначениеполей:

- ResponderProtocolsList – описание протокола и респондента в виде структуры типа TypeResponderProtocol.

## 5.10. Тип данных TypeResultsScaleValues

Этот тип содержит данные о значениях результатов тестирования по шкале.

Описание типа:

```
<xsd:complexType name="TypeResultsScaleValues">
      <xsd:sequence>
            <xsd:element name="ScaleNumber" type="xsd:int"/>
            <xsd:element name="ScaleTitle" type="xsd:string"/>
            <xsd:element name="ScaleValue" type="xsd:double"/>
            <xsd:element name="StenValue" type="xsd:double"/>
            <xsd:element name="SourceValue" type="xsd:int"/>
            <xsd:element name="PercentValue" type="xsd:int"/>
            <xsd:element name="HitValue" type="xsd:int"/>
            <xsd:element name="IqValue" type="xsd:int"/>
```

```
            <xsd:element name="CustomValue" type="xsd:double"/>
      </xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- ScaleNumber – номер шкалы;
- ScaleTitle –наименование шкалы;
- ScaleValue – значение по шкале;
- StenValue – результат по шкале в стенах;
- SourceValue – результат по шкале в сырых баллах;
- PercentValue – результат по шкале в процентах;
- HitValue – результат по шкале в виде числа ключевых ответов;
- IqValue – результат по шкале в iq – баллах.
- CustomValue – значение по настраиваемой шкале.

## 5.11. Тип данных TypeResultsValuesList

Этот тип данных предназначен для описания полного списка данных о численных значениях результата тестирования по шкале. Он содержит массив структур типа TypeResultsScaleValues, описанных выше.

Описание типа:

```
<xsd:complexType name="TypeResultsValuesList">
  <xsd:sequence>
        <xsd:element    name="ScaleValues"    type="tns:TypeResultsScaleValues"
minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

Назначениеполей:

- ScaleValues – описание значений результатов тестирования по шкале в виде структуры типа TypeResultsScaleValues.

## 5.12. Тип данных TypeSessionsIdsList

Этот тип данных предназначен для задания списка идентификаторов сеансов тестирования, по которым необходимо произвести выборку данных.

Описание типа:

```
<xsd:complexType name="TypeSessionsIdsList">
<xsd:sequence>
<xsd:element       name="sessionID"       type="xsd:string"       minOccurs="1"
maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- sessionID – идентификатор сеанса тестирования

## 5.13. Тип данных TypeScaleResult

Тип описывает значения шкал факторного профиля.

Описание типа:

```
<xsd:complexType name="TypeScaleResult">
<xsd:sequence>
<xsd:element name="ScaleId" type="xsd:string"/>
<xsd:element name="ScaleScore" type="xsd:int"/>
<xsd:element name="ScaleTime" type="xsd:string"/>
<xsd:element name="ScaleData" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- ScaleId – номер шкалы
- ScaleScore – значение шкалы
- ScaleTime – время, затраченное на прохождение тестирования в формате HH:MM:SS.uuu
- ScaleData – дополнительная информация

## 5.14. Тип данных TypeScalesResultsList

Тип представляет собой массив данных типа TypeScaleResult.

Описание типа:

```
<xsd:complexType name="TypeScalesResultsList">
```

```
    <xsd:sequence>
    <xsd:element   name="ScalesResults"   type="tns:TypeScaleResult"   minOccurs="0"
maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
```

## 5.15. Тип данных TypeSessionResult

Тип описывает сеанс тестирования.

Описание типа:

```
<xsd:complexType name="TypeSessionResult">
<xsd:sequence>
<xsd:element name="sessionid" type="xsd:string"/>
<xsd:element name="Status" type="xsd:int"/>
<xsd:element name="EndDate" type="xsd:string"/>
<xsd:element name="ScalesResults" type="tns:TypeScalesResultsList"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- sessionid – идентификатор сеанса тестирования
- Status – статус сеанса тестирования (0 – не найдена, 1 – не начата, 2 – начата, 3 – закончена)
- EndDate – дата/время завершения тестирования по нулевому меридиану в формате YYYY-MM-DD HH:MM:SS.uuu
- ScalesResults – массив оценок по каждой шкале для завершенных сеансов тестирования

## 5.16. Тип данных TypeSessionsResultsList

Тип представляет собой массив данных типа TypeSessionResult.

Описание типа:

```
<xsd:complexType name="TypeSessionsResultsList">
<xsd:sequence>
<xsd:element  name="SessionsResults"  type="tns:TypeSessionResult"  minOccurs="0"
maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
```

## 5.17. Тип данных TypeCompletedSessionResult

Тип описывает завершенный сеанс тестирования.

Описание типа:

```
<xsd:complexType name="TypeCompletedSessionResult">
<xsd:sequence>
<xsd:element name="sessionid" type="xsd:string"/>
<xsd:element name="EndDate" type="xsd:string"/>
<xsd:element name="ScalesResults" type="tns:TypeScalesResultsList"/>
</xsd:sequence>
</xsd:complexType>
```

Назначение полей:

- sessionid – идентификатор сеанса тестирования

- EndDate – дата окончания тестирования по нулевому меридиану в формате YYYY-MM-DD HH:MM:SS.uuu
- ScalesResults – факторный профиль

## 5.18. Тип данных TypeCompletedSessionsResultsList

Тип представляет собой массив данных типа TypeCompletedSessionsResult.

Описание типа:

```
    <xsd:complexType name="TypeCompletedSessionsResultsList">
    <xsd:sequence>
    <xsd:element                                        name="CompletedSessionsResults"
type="tns:TypeCompletedSessionResult" minOccurs="0" maxOccurs="unbounded"/>
    </xsd:sequence>
    </xsd:complexType>
```