http://www.hsdn.org
E-mail: info@hsdn.org

Программный интерфейс IBAPI 4

1.1. Описание составляющих модели

Структурная модель программного интерфейса IBAPI 4 состоит из нескольких уровней. Каждый уровень модели представляет собой набор классов, методов и функций, ориентированных на решение определенного спектра задач.

Понятия и определения

Транспортный модуль
Транспортный модуль представляет собой совокупность программно-аппаратных средств, предназначенных для приема и передачи данных между программным интерфейсом и клиентом.

Поставщик
Элемент транспортного модуля, описывающий какое-то определенное устройство взаимодействия с клиентом. Например, в случае, если транспортный модуль предназначен для работы с сетью ICQ, поставщиком будет являться процесс, обслуживающий какой-то UIN, на котором присутствует транспортный модуль. Переменная «адрес поставщика» в ответе от транспортного модуля, в этом случае, будет содержать именно этот номер UIN.

Клиент
Устройство, отправляющее и принимающее данные от транспортного модуля посредством поставщиков. В случае если рассматривать предыдущий пример, клиентом будет являться какой-то ICQ клиент, взаимодействующий с поставщиком транспортного модуля. В таком случае, переменная «адрес клиент» будет содержать UIN этого ICQ клиента.

Контроллер
Классы, методы или функции, предназначенные для выполнения (контроля) какой-то определенной задачи. Например, распознавание *реакций* по *массиву реакций*.

Реакция
Выполнение какой-либо операции в зависимости от контекста запроса, его вида и полноты. Например, если был задан запрос вида «foo bar», и при этом в массиве *контроллера выбора* (уровень операций) описана реакция «foo», то в случае его выполнения, будет выполняться реакция «foo» с *параметрами реакции* «bar».

Параметры реакции
Совокупность строковых данных, переданных в запросе (за исключением схождения обозначения самой *реакции*). Предыдущий пример наглядно демонстрирует, каким образом происходит нахождение параметров реакции. В зависимости от выбранного *операционного контроллера* (уровень операций) параметры реакции могут обрабатываться по-разному, и их конечное количество также зависит от применения того или иного операционного контроллера.

Уровень ввода
На данном уровне программный интерфейс получает сформированный (согласно определенным нормативам и правилам) *запрос интерфейса*, который в дальнейшем преобразуется в *структуру данных*.

Получение данных
На этом этапе происходит прием *запроса интерфейса*, поступившего от транспортного модуля. На следующем этапе запрос будет преобразован в структуру данных.

Преобразование данных
Преобразование запроса в структуру данных происходит путем применения класса-парсера (протокол IBTP). Преобразованные данные всегда содержат такую информацию, как: *идентификатор сессии*, *имя профиля, адрес поставщика, адрес клиента* и *тело запроса*.

Получение конфигурации профиля
После получения данных, происходит получение *конфигурации профиля* (уровень управления). В дальнейшем именно определенный на этом этапе профиль становится *текущим*, и будет использоваться всем программным интерфейсом.

Обработчик ввода
На этом этапе применяются различные операции по обработке текста *тела запроса*. Обработка состоит из двух этапов: *перекодировка* и применение определенных *функций обработки*. Перекодировка предназначена для преобразования текста из одной кодировки в другую. Исходная кодировка и кодировка преобразования заданы в текущем профиле. После того, как текст был перекодирован, на нем применяются различные функции обработки, определенные в профиле. Каждая такая функция принимает и возвращает только одно строковое значение. Выполнение функций обработки происходит по порядку их описания в профиле.

Вызов *контроллера выбора* (уровень операций)
После проведения процедур по получению и обработке данных, выполняется вызов *контроллера выбора* (уровень операций). Название контроллера вывода определено текущим профилем. В контроллер выбора передается только один аргумент, содержащий *тело запроса*.

Уровень управления
Этот уровень предназначен для хранения управляющих структур и конфигураций программного интерфейса. На этом уровне определяются все параметры работы интерфейса, а также устанавливаются его характеристики и возможности.

Конфигурации профилей
Представляют собой массивы данных, хранящие конфигурации различных *профилей*. Конфигурация профилей хранится в *массиве описания профилей*. Любой определенный здесь профиль имеет наименование и хранит такие данные как: *секретный ключ, название группы синонимов, название контроллера выбора* (уровень операций)*, описание обработчиков ввода* и *обработчиков вывода* (уровень ввода).

Конфигурации реакций
Реакции интерфейса хранятся в *массиве описания реакций* и разделяются на два типа: *функциональные реакции* и *контроллерные реакции*. Функциональные реакции содержат такие данные, как: *название операционного контроллера* (уровень операций), *название контроллера функций* (уровень функций). Контроллерные реакции описывают только секцию контроллера выбора, которая содержит *название контроллера выбора* (уровень операций) и его *аргументы*.

Конфигурации синонимов
Синонимы реакций хранятся в *массиве описания синонимов* и содержат такие данные, как: *пример* и *замена*. Количество описываемых синонимов не ограничено. После обработки тела запроса обработчиком ввода (уровень ввода), все вхождения, совпадающие со значением «примера», находящиеся в начале будут, заменены на значение параметра «замена». Обработка синонимов происходит по порядку их описания в массиве.

Уровень операций
На данном уровне происходит выполнение операций по проверке текста *тела запроса* на наличие определенных профилем, *реакций,* и вызов соответствующего *контроллера функций* (уровень функций).

Контроллеры выбора
Представляют собой совокупность методов и условий, предназначенных для проверки текста *тела запроса* на наличие в нем *реакций*. В зависимости от строения и задач, контроллеры выбора делятся на два типа: *автоматные* и *условные*. Автоматные контроллеры выбора обычно представляют собой конечный автомат, который в качестве своих состояний использует ключи из заданного *массива реакций* (уровень управления). Условные контроллеры выбора используют различные встроенные и внешние алгоритмы для нахождения *реакции*. После определения реакции, контроллер, в зависимости от типа реакции, вызывает либо *контроллер выбора* (если реакция имеет тип «контроллер»), либо *операционный контроллер* (если реакция имеет тип «функция»). В первом случае, в вызываемый *контроллер выбора* передается только один параметр, содержащий полное *тело запроса* (для дальнейшего его анализа вызываемым контроллером выбора). Во втором случае, в вызываемый *контроллер функции* передается два аргумента: *название контроллера функции* (уровень функций) и *текст параметров реакции*.

Операционные контроллеры
Представляет собой контроллер, предназначенный для нахождения *параметров реакции* в принятом значении, и вызова *контроллера функции* (уровень функций)*.* Принимаемые контролером аргументы состоят из *названия контроллера функций* и *текста параметров реакции*. В зависимости от функционального наполнения операционного контроллера, текст параметров реакции может быть обработан по-разному. Но в любом из случаев, операционный контроллер вызывает, заданный в первом аргументе, *контроллер функции,* и передает в него список из найденных *параметров реакций* в качестве его аргументов.

Уровень функций

Данный уровень отвечает за обработку всех *реакций*, полученных из запросов к программному интерфейсу. Здесь определены *контроллеры функций* и *модели*, при помощи которых программный интерфейс производит обработку запросов.

Контроллеры функций

Контроллер функции представляет собой набор различных операций для решения определенной задачи. Контроллер принимает различное число параметров в зависимости от его строения и назначения. Контроллеры функций могут использовать внешние *модели* приложения для расширения своих базовых возможностей. В не зависимости от строения, контроллер всегда возвращает на уровень вывода только одно значение, содержащее, как правило, результат обработки запроса или ошибку.

Модели

Представляют собой внешние или внутренние классы контроллеров и функций, призванных для расширения базовых возможностей *контроллеров функций*. Для моделей допустимо обращение к внешним классам, моделям и библиотекам.

Библиотеки

Представляют собой внешние классы контроллеров и функций, предназначенных для расширения базовых возможностей программного интерфейса. Для библиотек, как и для моделей допустимо обращение к другим классам, моделям и библиотекам.

Вызов *обработчика вывода* (уровень вывода)

После успешного завершения выполнения *контроллера функции* (уровень функций), происходит вызов *обработчика вывода* (уровень вывода), который принимает только один аргумент – определенное контроллером функции, строковое значение (*тело результата*).

Уровень вывода

Данный уровень отвечает за последующую обработку текста *тела результата* и дальнейшую его передачу на операционный модуль в качестве *ответа*.

Обработчик вывода

Как и *обработчик ввода* (уровень ввода), обработчик вывода производит обработку текста при помощи последовательного выполнения ряда операций: применение *рекламного контроллера*, *перекодировки*, применение определенных *функций обработки вывода* и применение *сплиттера*. Рекламный контроллер представляет собой набор функций, предназначенных для добавления рекламной информации к тексту *тела результата*. Как и в случае с обработчиком ввода, перекодировка предназначена для преобразования текста *тела результата* из одной кодировки в другую. Исходная кодировка и кодировка преобразования заданы в текущем профиле. После того, как текст был перекодирован, на нем применяются различные функции обработки, определенные профилем. Выполнение функций обработки происходит по порядку их описания в профиле. Каждая такая функция принимает и возвращает только одно строковое значение. После применения функций, на тексте *тела результата* применяется *сплиттер*, который предназначен для разделения текста на несколько частей (по словам). Значение длины каждой части задается в текущем профиле.

Преобразование данных

Преобразование *ответа* и системных переменных в сформированный *ответ интерфейса*, происходит путем применения класса-билдера (протокол IBTP). Построенный преобразованный ответ данных всегда содержат такую информацию, как: *адрес поставщика, адрес отравителя* и *тело ответа*.

Отправка данных

На последнем этапе происходит отправка сформированного *ответа интерфейса* на транспортный модуль. Это финальная операция, в результате выполнения которой, транспортный модуль получает полноценные и достаточные данные для передачи их клиенту.

2.1. Структура модели

Уровень ядра

- Обработчик ошибок
  - Получение конфигураций профилей (уровень управления)
  - Получение конфигураций исключений (уровень управления)
  - Отправка сообщения об ошибке
    - Вызов обработчика вывода (уровень вывода)
- Ведение журналов
  - Получение конфигураций интерфейса (уровень управления)
  - Открытие файла журнала процесса
  - Открытие файла журнала ошибок
  - Запись процесса в журнал
  - Запись ошибок в журнал

Уровень ввода

- Получение данных
- Преобразование данных
  - Получение конфигураций профилей (уровень управления)
- Обработчик ввода
  - Получение конфигураций групп синонимов (уровень управления)
  - Перекодировка
  - Заданные функции обработки

Вызов контроллера выбора (уровень операций)

Уровень управления

- Конфигурации профилей
  - Название профиля
    - Секретный ключ
    - Название группы синонимов
    - Описание контроллера выбора
      - Название контроллера выбора
      - Аргументы контроллера выбора
    - Описание обработчиков ввода
      - Перекодировка
      - Заданные функции обработки
    - Описание обработчиков вывода
      - Перекодировка
      - Заданные функции обработки
      - Сплиттер (word-wrap)
- Конфигурации реакций
  - Название реакции
    - Название операционного контроллера (функциональная реакция)
    - Название контроллера функций (функциональная реакция)
    - Описание контроллера выбора (контроллерная реакция)
      - Название контроллера выбора (контроллерная реакция)
      - Аргументы контроллера выбора (контроллерная реакция)
- Конфигурации групп синонимов
  - Строка примера (синоним реакции)
    - Строка замены (название реакции)
- Конфигурация интерфейса
  - Кодировка интерфейса
  - Включение ведения журнала процесса
  - Включение ведения журнала ошибок
  - Название файла ведения журнала процесса
  - Название файла ведения журнала ошибок
- Конфигурация исключений
  - Код исключения
    - Текст сообщения

Уровень операций

- Контроллеры выбора
  - Получение конфигураций реакций (уровень управления)
  - Вызовы контроллеров выбора или операционных контроллеров (уровень операций)
- Операционные контроллеры
  - Нахождение и обработка параметров реакции
  - Вызовы контроллеров функций (уровень функций)

Уровень функций

- Контроллеры функций
  - Операции по обработке запроса
  - Вызовы моделей
- Модели
  - Операции по обработке запроса
  - Вызовы моделей

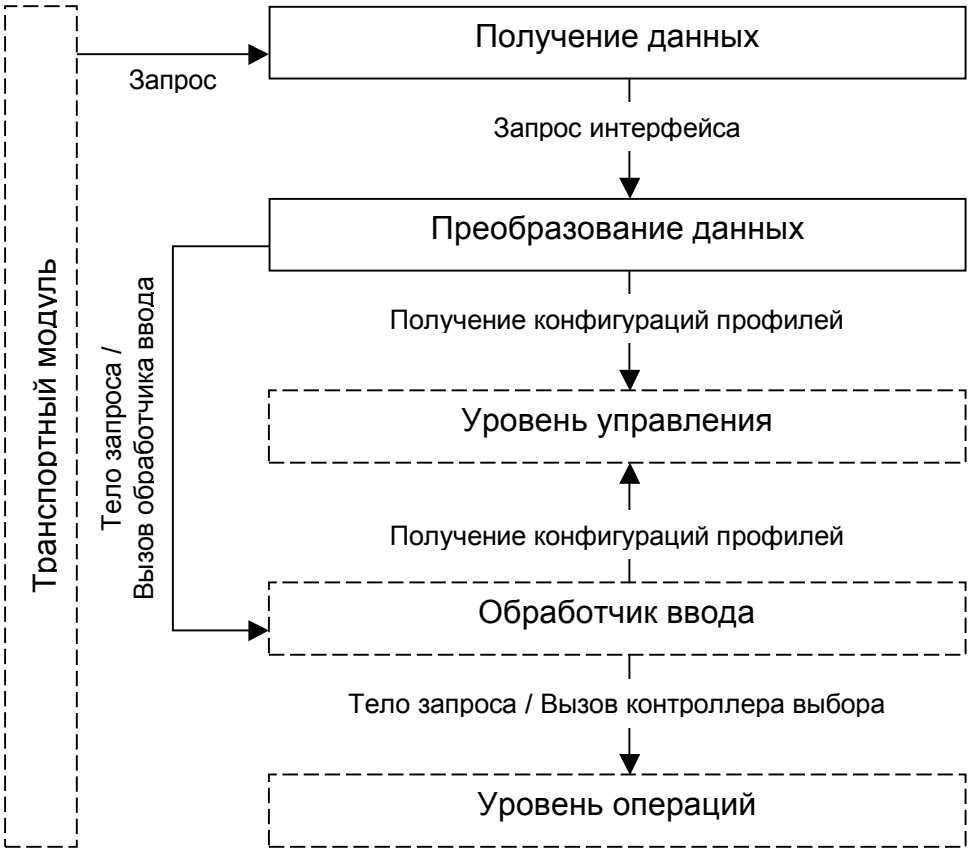
Вызов обработчика вывода (уровень вывода)

Уровень вывода

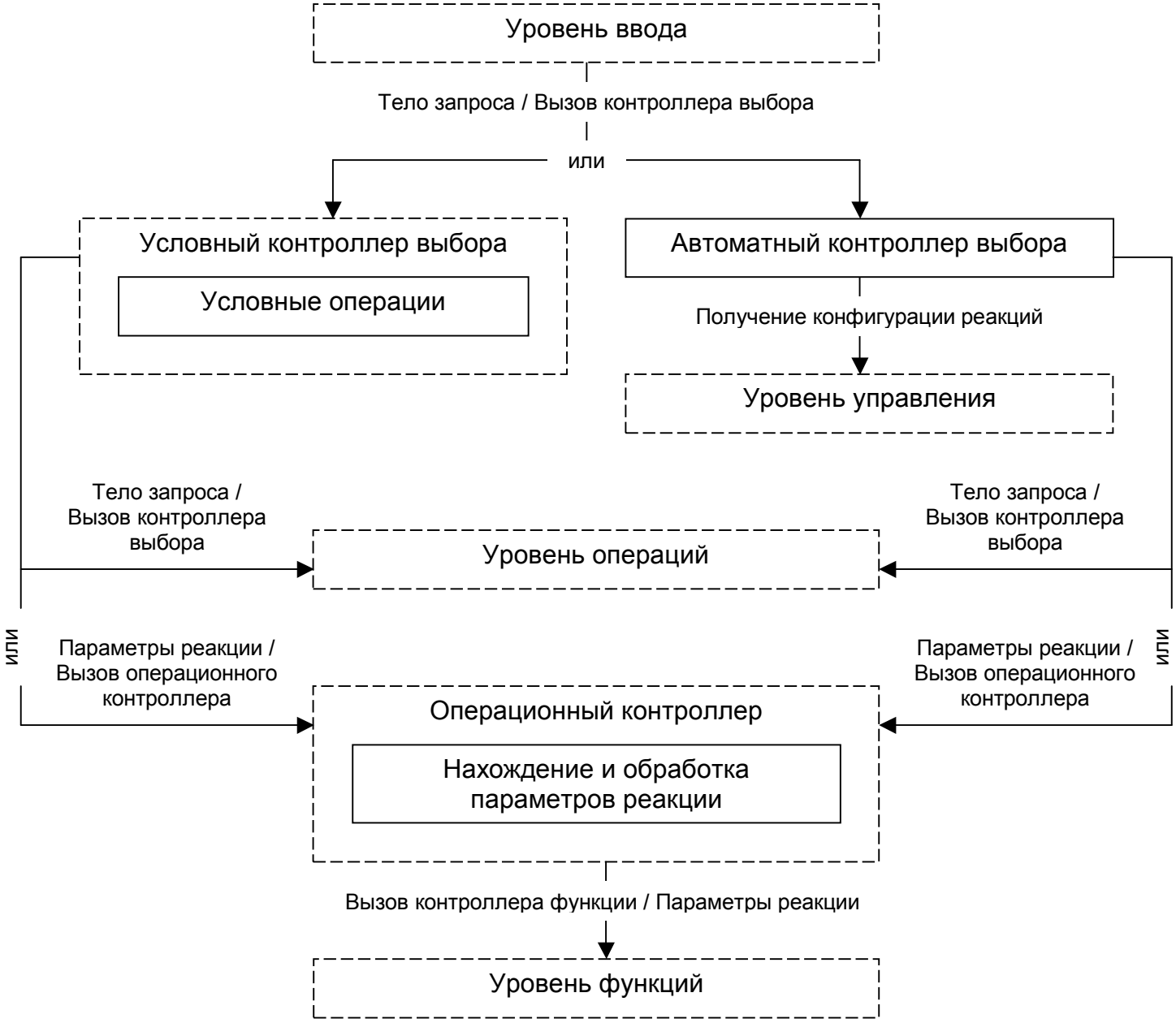
- Обработчик вывода
  - Получение конфигураций профилей (уровень управления)
  - Рекламный контроллер
  - Перекодировка
  - Заданные функции обработки
  - Сплиттер
- Преобразование данных
- Отправка данных

3.1. Диаграмма взаимодействия уровней модели

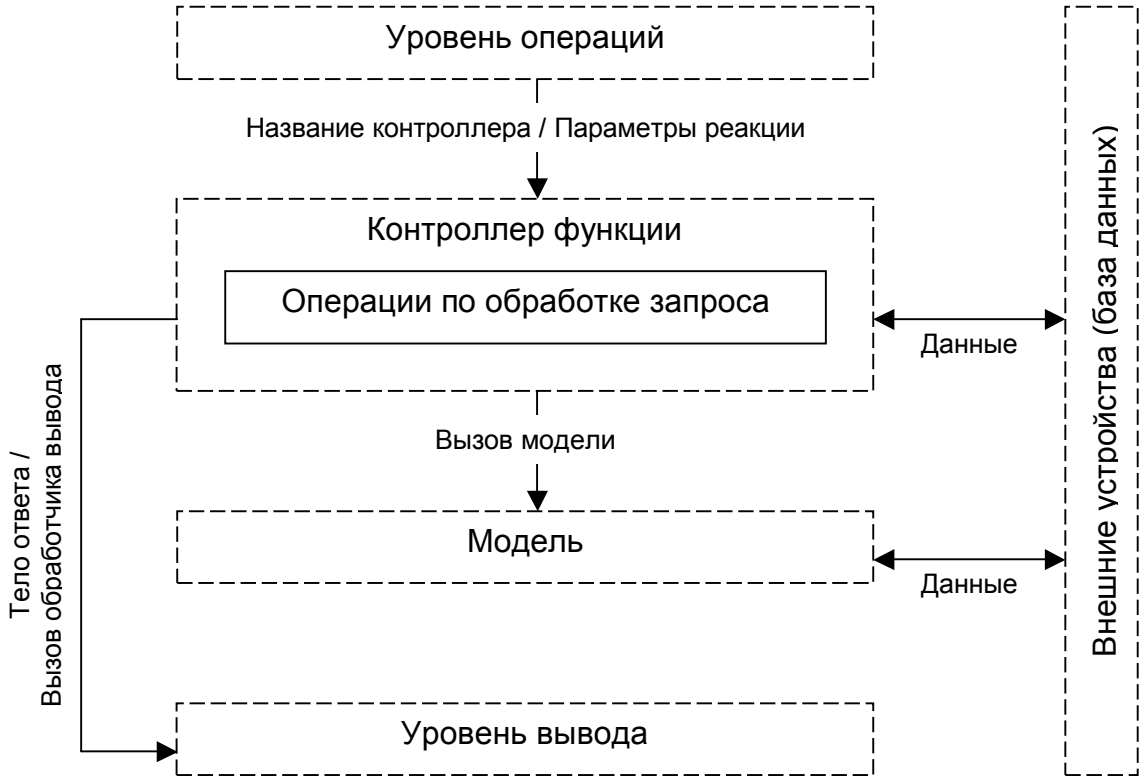
3.2. Диаграмма строения уровня ввода



3.3. Диаграмма строения уровня операций



3.4. Диаграмма строения уровня функций



3.5. Диаграмма строения уровня вывода

4.1. Описание файловой структуры

Директория	Название файла	Описание
/	index.php	Вызов фронт-контроллера
/base	FrontController.class.php	Фронт-контроллер (ядро)
/base	Exceptions.class.php	Класс обработки исключений (ошибок)
/base	Logging.class.php	Класс ведения журналов
/base	Input.class.php	Класс-контейнер уровня ввода
/base	InputParser.class.php	Преобразователь данных
/base	InputProcessing.class.php	Обработчик ввода
/base	Management.class.php	Класс управления
/structure/ManagementReactions	<NAME>.inc.php	Конфигурации реакций
/structure/ManagementProfiles	<NAME>.inc.php	Конфигурации профилей
/structure/ManagementAliases	<NAME>.inc.php	Конфигурации синонимов
/base	Operation.class.php	Класс операций
/structure	OperationSelectors.class.php	Класс контроллеров выбора
/structure	OperationControllers.class.php	Класс операционных контроллеров
/base	Function.class.php	Класс уровня функций
/structure	FunctionControllers.class.php	Класс контроллеров функций
/structure	FunctionalStatistics.class.php	Класс ведения статистики
/structure/FunctionModels	<NAME>.class.php	Модели
/base	Output.class.php	Класс вывода
/base	OutputProcessing.class.php	Обработчик вывода
/base	OutputBuilder.class.php	Преобразователь данных
/base	ProcessingAdvertisement.class.php	Рекламный обработчик
/structure	ProcessingFunctions.class.php	Функции обработки ввода-вывода
/libraries	Database.class.php	Класс работы с базой данных
/configuration	Database.inc.php	Файл конфигурации базы данных
/configuration	Interface.inc.php	Файл конфигурации интерфейса
/configuration	Exceptions.inc.php	Файл конфигурации ошибок
/configuration	Constants.inc.php	Файл конфигурации констант

4.2. Описание директорий

Корневая директория
Содержит поддиректории приложения, а также индексный файл, из которого происходит вызов фронт-контроллера.

Директория *base*
Содержит основные компоненты приложения, которые не подлежат редактированию при изменении функциональности программного интерфейса.

Директория *structure*
Содержит компоненты приложения, из которых строится структура функциональности программного интерфейса.

Директория *libraries*
Содержит различные библиотеки, предназначенные для расширения базовых возможностей приложения.

Директория *configurations*
Содержит файлы конфигурации приложения.