# Методика анализа готовности к универсальному принятию для программного обеспечения, обрабатывающего доменные имена и адреса электронной почты

Настоящая методика дает общие понятия о доменной адресации, использовании интернационализированных доменных имен и адресов электронной почты, особенностях обработки приложениями новых общих доменов верхнего уровня. В данном документе представлена обобщенная модель тестирования программного обеспечения на предмет готовности к универсальному принятию, а также представлены рекомендации по проведению такого тестирования и оценке его результатов. Документ разработан на основе Universal Acceptance Readiness Framework.

## Оглавление

## Обозначения и сокращения

- Universal Acceptance, UA – универсальное принятие.
- Internet Corporation for Assigned Names and Numbers, ICANN – Интернет-корпорация по присвоению имен и номеров.
- American standard code for information interchange, ASCII – название кодировки, использующей только символы латинского алфавита, десятичные цифры, знаки препинания и управляющие символы.
- Unicode – стандарт кодирования символов, включающий в себя знаки почти всех письменных языков мира.
- UTF-8 – кодировка стандарта Unicode, обеспечивающая совместимость с кодировкой ASCII.
- UA Readiness – готовность к универсальному принятию.
- Internationalized Domain Names, IDN – интернационализированные доменные имена.
- E-mail Address Internationalization, EAI – интернационализация адреса электронной почты.
- Top-Level Domain, TLD – домен верхнего уровня.
- UA-идентификаторы – доменные имена, включающие в себя новые общие домены верхнего уровня, интернационализированные доменные имена и интернационализированные адреса электронной почты.
- Domain Name System, DNS – система доменных имен.
- Country code Top Level Domain, ccTLD – страновый или национальный домен верхнего уровня.
- Generic Top Level Domain, gTLD – общий домен верхнего уровня.
- Internet Engineering Task Force, IETF – Инженерный совет интернета.
- International Organization for Standardization, ISO – Международная организация по стандартизации.
- Punycode – обратимый механизм преобразования доменных имен между Unicode и ASCII.
- A-label – представление доменного имени в кодировке ASCII.
- U-label – представление доменного имени в кодировке Unicode.
- Graphical User Interface, GUI – графический пользовательский интерфейс.
- Normalization Form C, NFC – нормализация символов Unicode композицией. Процесс, при котором несколько символов по возможности объединяются в один.

## Введение

Технологии адресации в интернете, включая адресацию при помощи доменных имен, постоянно развиваются и меняются. В последние годы Интернет-корпорацией по присвоению имен и номеров (ICANN) были одобрены новые страновые и общие домены верхнего уровня, причем использующие как традиционные латинские символы ASCII, так и символы Unicode, например такие, как кириллический домен верхнего уровня - .рф.

Тем не менее многие приложения и сервисы оказались не готовы к использованию расширенного диапазона доменных имен. К тому же отсутствие в интернете единой, общепринятой практики использования стандартов обработки электронной почты, содержащей не-ASCII символы, породило большое количество различных реализаций этой обработки в приложениях и сервисах разных разработчиков, что зачастую приводило к потере возможности корректного почтового обмена этих приложений и сервисов между собой.

Поэтому в 2015 году была сформулирована основная идея универсального принятия, которая заключается в обеспечении равных возможностей для работы всех допустимых доменных имен и адресов электронной почты во всех интернет-ориентированных приложениях, сервисах и системах. Универсальное принятие имеет важное значение для дальнейшего расширения использования интернета и создает удобный интерфейс для миллиардов новых пользователей глобальной сети.

Готовность к универсальному принятию (UA Readiness) — это способность приложений и сервисов использовать идентификаторы (доменные имена и адреса электронной почты), которых не было на заре становления интернета. Говоря более конкретно, это состояние приложения или сервиса, в котором правильно и полностью поддерживаются:

- интернационализированные доменные имена (IDN) – потому как по умолчанию большинство приложений и сервисов ожидают, что доменное имя будет представлено в кодировке ASCII;
- интернационализированные адреса электронной почты (EAI) – потому как по умолчанию большинство приложений и сервисов ожидают, что локальная часть (набор символов слева от знака «@» для алфавитов с написанием слева направо) адреса электронной почты будет иметь кодировку ASCII, и они не готовы к получению писем с локальной частью адреса электронной почты в кодировке, отличной от нее;
- длинные строки доменов верхнего уровня – потому как некоторые приложения и сервисы ожидают, что длина строки TLD не будет превышать трех символов;

- актуальный реестр доменов верхнего уровня – потому как некоторые приложения и сервисы проверяют существование TLD на основе устаревшего и статичного списка или не имеют такой проверки вовсе.[1]

Цель настоящего документа — определить методику анализа того, правильно ли приложение или сервис поддерживает универсальное принятие, и выявить проблемы, которые необходимо устранить разработчикам для повышения готовности к универсальному принятию.

## Целевая аудитория

Этот документ предназначен для аудитории технических специалистов (разработчиков, руководящего технического персонала), которые могут быть знакомы с некоторыми аспектами интернет-технологий, но не обязательно владеют детальной информацией о том, как именно поддерживаются приложениями и сервисами интернационализированные и традиционные доменные имена, а также адреса электронной почты.

## Основные понятия

Настоящая методика использует ряд ключевых понятий построения доменной адресации и адресов электронной почты, кодировок, используемых в этих целях, и механизмов преобразований. В этом разделе представлено краткое введение в такие основные понятия.

### Доменные имена

Доменное имя - это удобный для человека идентификатор ресурсов в интернете. Оно обычно представляет собой последовательность символьных блоков, разделенных точками, например www.example.tld. Каждый такой блок представляет собой уровень в иерархии системы доменных имен.

На самом верхнем уровне, или в «корне», иерархии DNS находится корневой (или нулевой) домен. Обозначается он точкой, но в большинстве пользовательских программ, вроде браузеров, корневой домен опускается при вводе. Далее идут домены верхнего уровня, такие как .рф, например. На следующем уровне располагается поддомен TLD, называемый доменом второго уровня, а за ним поддомен домена второго уровня, называемый доменом третьего уровня, и так далее. Каждый уровень доменного имени отделен точкой.

[1] Стоит отметить, что TLD довольно часто добавляются в корневую зону или удаляются из нее, иногда даже ежедневно. Например, в течение 10-дневного периода в ноябре 2019 года из корня были удалены 7 TLD.

Например, трехуровневое доменное имя в общем случае выглядит так:

*Домен3Уровня.Домен2Уровня.ДоменВерхнегоУровня.*

***test.example.com***

***тестирование.поддерживаю.рф***

## Страновые домены верхнего уровня

Некоторые TLD делегируются конкретным странам или территориям. Они называются страновыми, или национальными, доменами верхнего уровня. В прошлом все страновые домены состояли из двух латинских букв, соответствующих коду в стандарте ISO 3166, присвоенному стране или территории Международной организацией по стандартизации: например, российский домен .ru, делегированный 7 апреля 1994 года. После 2010 года началось выделение интернационализированных страновых доменов верхнего уровня (IDN ccTLD), которые представляют собой название страны или территории на ее национальном языке. Один из первых таких доменов – российский кириллический домен .рф, делегированный 12 мая 2010 года.

## Общие домены верхнего уровня

Домены верхнего уровня, не являющиеся страновыми, называются общими доменами верхнего уровня (generic TLD или gTLD). В таких TLD регистрация либо открыта для всех желающих, либо ограничена членами определенного сообщества. Среди этих доменов знакомые всем .com, .net и .org. Благодаря первому раунду программы New gTLD – инициативе ICANN по расширению перечня доменов верхнего уровня - число gTLD увеличилось за счет множества новых доменов, представляющих бренды, сообщества, географические территории (города, регионы) – например, .москва, .дети, .онлайн и другие.

## Интернационализация доменных имен и электронной почты

Доменные имена изначально были ограничены подмножеством символов ASCII: латинские буквы a-z без учета регистра, цифры 0-9 и дефис "-". С самой первой регистрации в 1985 году домена *symbolics.com* число и характеристики доменных имен расширялись, чтобы отразить потребности растущего числа пользователей интернета. Сегодня большинство интернет-пользователей не являются носителями английского языка, который тем не менее продолжает быть доминирующим языком, используемым в сети.

Чтобы помочь интернационализации интернета, в 2003 году Инженерный совет интернета (IETF) начал выпуск стандартов IDNA, предоставляющих технические рекомендации для развертывания интернационализированных доменных имен через механизм преобразования, в котором могли бы использоваться не-ASCII символы доменных имен в любом Unicode-поддерживаемом языке, например .рф, .москва, .дети. и т.д.

Совет директоров ICANN одобрил процесс введения новых интернационализированных доменных имен в октябре 2009 г., и уже в мае 2010 г. в корневую зону были добавлены первые такие имена. В июне 2011 г. Правление ICANN одобрило и санкционировало запуск программы New gTLD, которая включала не только

новые ASCII, но и IDN TLD общего назначения. Первые новые общие домены верхнего уровня по этой программе были включены в корневую зону в 2013 году.

Спустя десять лет с того момента, как IETF выпустила технические рекомендации по работе IDN, а ICANN запустила программу New gTLD, в мире появилось более 1000 новых доменных зон. Однако до сих пор часть программного обеспечения не обновлена в соответствии с существующими стандартами IDN и не умеет обрабатывать новые TLD и IDN TLD. Это вызывает проблемы у пользователей сети и, в первую очередь, у тех, кто использует не-ASCII символы.

Подход универсального принятия призван гарантировать, что все действующие доменные имена и адреса электронной почты принимаются, проверяются, хранятся, обрабатываются и отображаются правильно и последовательно всеми интернет-приложениями, устройствами и системами. Например, каждый действительный веб-адрес позволяет получить доступ к ожидаемому ресурсу на правильном веб-сайте, а каждый действительный адрес электронной почты приводит к доставке почты ожидаемому получателю.

Система DNS была разработана для обработки только ASCII символов, и с появлением IDN доменов необходимо было создать дополнительный механизм преобразования, позволяющий представлять символы Unicode в виде набора символов ASCII.

Механизм, преобразующий кодировку Unicode в ASCII, называется Punycode, а результат его работы – A-метки, характерной чертой которых является то, что они всегда начинаются со следующих четырех символов *xn*--.

Преобразование Punycode является обратимым, корректная последовательность символов A-метки может быть преобразована обратно в Unicode последовательность, называемую U-меткой.

Punycode используется только для преобразований, связанных с интернационализированными доменными именами. Конечно, с его помощью можно преобразовывать и другие идентификаторы или их части, например локальную часть электронной почты, однако широкого применения подобное использование Punycode не нашло, и, следовательно, оно пока остается неприменимым для взаимодействия с другими приложениями и сервисами.

**Таблица 1. Пример представления IDN**

| U-метка | A-метка |
|---|---|
| *поддерживаю.рф* | *xn--80adfafgo7bio2n.xn--p1ai* |

Адреса электронной почты состоят из двух частей: локальная часть (перед символом «@») и доменная часть (после символа «@»). «До» и «после» следует понимать с учетом правил написания слева направо (LTR - Left-To-Right) и справа налево (RTL - Right-To-Left) в зависимости от языка, который используется в конкретном адресе электронной почты. Например, на арабском языке адрес электронной почты будет выглядеть как رضوان @ڈیٹا.بھارت.

Концепция EAI предполагает поддержку приложениями и сервисами любых адресов электронной почты, не важно в доменной и/или в локальной части содержатся Unicode символы, написана она справа налево или слева направо.

**Таблица 2. Примеры EAI-адресов**

| | |
|---|---|
| *example@test.рф* | Использует IDN TLD |
| *example@поддерживаю.рф* | Использует еще и IDN-домен второго уровня |
| *пример@поддерживаю.рф* | Использует еще и Unicode в названии адреса электронной почты (в локальной части адреса) |

### Система доменных имен

Система доменных имен – большая распределенная база данных с разделами для каждого TLD, которые называют зонами. Главный раздел, содержащий все доменные зоны, - это корневая зона, поскольку концептуально находится в корне дерева системы доменных имен. Все DNS-зоны, включая корневую, обновляются по мере необходимости. Как только новые зоны добавляются или старые удаляются, их имена, соответственно, добавляются или удаляются из корневой зоны.

Это означает, что любой фиксированный список TLD, сохраненный в приложении или в файле, рано или поздно неизбежно устареет. Чтобы надежно подтвердить правильность зоны в доменном имени, программное обеспечение может проверить его в режиме реального времени с помощью запроса к системе доменных имен или получить его с официального ресурса организации (IANA), осуществляющей поддержку реестра доменов верхнего уровня.

## Приложения

Практически все современные приложения имеют функционал, связанный с обменом данными по сети. Даже если приложение решает задачи, с сетью не связанные, в силу необходимости обновлять собственный код из-за найденных в нем ошибок или для модернизации функционала, такое приложение так или иначе имеет функционал сетевого взаимодействия. Обмен данными зачастую происходит при помощи доменной адресации, а для взаимодействия с пользователями в подавляющем большинстве случаев используется электронная почта.

В рамках данной методики предлагается условно разделить анализируемые приложения на две категории: *веб-приложения* и *GUI-приложения*. Каждая из предложенных категорий имеет следующие признаки.

**Веб-приложение:**

- доступ к пользовательскому интерфейсу осуществляется посредством браузера;
- большая часть кода выполняется удаленно;

- для доступа к GUI может использоваться как сторонний браузер (Chrome, Firefox, Safari, IE и т.д.), так и собственной разработки, встроенный в приложение.

**GUI-приложение:**

- обладает собственным графическим пользовательским интерфейсом;
- большая часть кода выполняется локально, непосредственно в операционной системе пользователя;
- преимущественно не использует браузер.

Стоит отметить, что некоторые приложения могут казаться GUI-приложениями, т.е. обладающими собственным графическим интерфейсом, но на самом деле иметь локальные веб-страницы, отображаемые с помощью встроенного браузера, т.е. являться веб-приложениями.

## Составляющие приложений

Приложение обычно состоит из набора логических составляющих, и у каждой из указанных выше категорий приложений он свой. В этой методике используется упрощенный подход и основное внимание уделяется только основным функциональным элементам приложения, таким образом стоит учитывать, что анализируемые приложения могут соответствовать предложенной модели не совсем точно. Однако разделение приложения на функциональные элементы позволяет понять, какой именно из них соответствует требованиям универсального принятия, а у какого есть с этим проблемы.

**Компоненты веб-приложения:**

- Клиентская часть, или браузер, — это внешнее приложение, необходимое для информационного обмена с веб-приложением, обеспечивающее доступ к его интерфейсу.
- Фронтенд, или внешний интерфейс, — элемент генерации кода для выполнения его в браузере. Он отвечает за создание веб-страниц и позволяет выполнять в браузере код HTML, CSS и Javascript.
- Бэкенд, или серверная часть, — отвечает за серверную, инфраструктурную часть веб-приложения, к примеру, за функционирование веб-сервера, за информационный обмен с базой данных и т.д.
- База данных — осуществляет хранение и структуризацию информации на серверной стороне.
- Файловая система — определяет способ организации, хранения и именования данных на серверных машинных носителях.
- Внешний сервис — элемент, отвечающий за использование сторонних сервисов, дополняющих встроенные функции приложения. Распространенным примером является аутентификация, осуществляющаяся посредством сторонней системы.

**Компоненты GUI-приложения:**

- Пользовательский графический интерфейс – графический интерфейс для взаимодействия с пользователем, отвечающий за ввод/вывод данных, например таких, как доменные имена или адреса электронной почты.
- Внутренняя часть – основной программный код, обрабатывающий вводимые пользователем данные и обеспечивающий выполнение функций приложения.
- База данных — система хранения и структуризации информации, может быть как локальной, так или удаленной.
- Файловая система — определяет способ организации, хранения и именования данных. Может быть удаленной, распределенной или локальной, принадлежащей операционной системе пользователя приложения.
- Внешний сервис — элемент, отвечающий за использование сторонних сервисов, дополняющих встроенные функции приложения.

### Этапы обработки идентификаторов приложением

Обработка данных приложением осуществляется в несколько базовых этапов. Настоящая методика предлагает для анализа шесть таких этапов, каждый из которых может быть применим как к приложению в целом, так и к отдельному его элементу. В качестве обрабатываемых данных используются так называемые UA-идентификаторы: New gTLD, IDN или EAI. Обобщенная последовательность базовых этапов выглядит следующим образом:

1. Принятие – этап, на котором приложение принимает введенный пользователем идентификатор.
2. Проверка – этап, на котором приложение выполняет валидацию идентификатора.
3. Обработка (на входе) – этап, на котором приложение выполняет начальную обработку введенного идентификатора.
4. Хранение – этап, на котором приложение сохраняет идентификатор в базу данных.
5. Обработка (на выходе) – этап, на котором приложение обрабатывает идентификатор, полученный из базы данных.
6. Отображение – этап, на котором приложение отображает идентификатор.

Следует отметить, что для некоторых приложений какие-то этапы могут быть неактуальны. Например, приложение может не сохранять идентификаторы, следовательно, у него не будет этапов 4 и 5.

## Модель тестирования

Настоящая методика предлагает предварительную аналитическую модель тестирования элементов обоих категорий приложений на каждом из этапов обработки UA-идентификаторов. В предлагаемой модели рассматривается значимость влияния на общий результат каждого из элементов приложения на соответствующем этапе его тестирования. Стоит заметить, что предлагаемая аналитическая модель весьма обобщена и условна, из чего следует, что она вполне может быть изменена в зависимости от особенностей реализации конкретного анализируемого приложения.

В соответствующих категориям приложений таблицах ниже указаны следующие значения:

- **Да** – когда функциональный элемент приложения играет некоторую роль на соответствующем этапе;
- **М** – когда элемент может играть какую-то роль на этом этапе, а может и не иметь значимости;
- **Пустое значение** – когда элемент на данном этапе, скорее всего, не играет никакой роли.

**Таблица 3. Веб-приложение**

| | **Принятие** | **Проверка** | **Обработка** | **Хранение** | **Обработка** | **Отображение** |
|---|---|---|---|---|---|---|
| **Браузер** | Да | Да (только IDN-домены в URL) | | М (локальное хранилище) | | Да |
| **Фронтенд** | | М | М | | М | |
| **Бэкенд** | | М | М | | М | |
| **База данных** | | | | Да | | |
| **Файловая система** | | | | Да | | |
| **Внешний сервис** | | М | М | М | М | |

**Таблица 4. GUI-приложение**

| | **Принятие** | **Проверка** | **Обработка** | **Хранение** | **Обработка** | **Отображение** |
|---|---|---|---|---|---|---|
| **GUI** | Да | М | | | | Да |
| **Внутренняя часть** | | М | М | | М | |
| **База данных** | | | | Да | | |
| **Файловая система** | | | | Да | | |
| **Внешний сервис** | | М | М | М | М | |

На каждом этапе применяется соответствующий именно этому этапу тест корректности работы элемента с UA-идентификатором. Условные обозначения применяемых тестов следующие:

- **AT** – тест принятия.
- **VT** – тест валидации.
- **P1T** – тест обработки на входе.
- **ST** – тест хранения.
- **P2T** – тест обработки на выходе.
- **DT** – тест отображения.

Ниже представлена схема тестирования функциональных элементов приложения, где каждый тест применяется на соответствующем этапе:

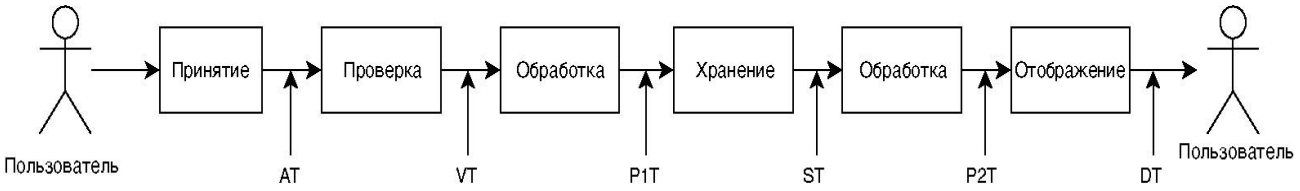


**Рисунок 1. Схема тестирования**

Архитектура современных программ представляет собой системы с контейнерами, микросервисами, многоуровневостью, кэш-памятью на всех уровнях, облачными сервисами и т. д. Фактически приложение представляет собой разветвленную структуру функциональных элементов, взаимодействующих друг с другом сложным образом. Из-за подобных особенностей даже сами разработчики приложения при проведении его тестирования могут испытывать трудности с выявлением проблем, связанных с универсальным принятием.

Кроме того, часть из принятых в методике этапов обработки идентификаторов в некоторых случаях может отсутствовать или некоторые этапы могут быть объединены. Например, проверка и обработка идентификаторов происходит одновременно в пределах одного функционального элемента, а элемент их хранения и вовсе отсутствует. Да и предложенные в модели границы между функциональными элементами зачастую размыты.

Более того, в архитектуре современных программ часто используются облачные сервисы разных поставщиков. Код самих сервисов и их компоненты обычно не раскрываются. Следовательно, эти сервисы приходится рассматривать как черные ящики без возможности их изучения, т.е. попытаться проверить их можно только посредством работы с интерфейсом. Учитывая доминирующее положение таких приложений на рынке разработки программного обеспечения, такую их особенность необходимо учитывать при проверке на соответствие требованиям универсального принятия. Примерами таких сервисов являются сторонние службы идентификации/аутентификации/авторизации. Разработчику программного обеспечения проще использовать Google, Facebook, Apple или другие уже

существующие службы аутентификации пользователей, поскольку это позволяет ему сосредоточиться на уникальных аспектах своего приложения, не тратя время и средства на «изобретение велосипеда» для часто встречающихся функций. Ко всему прочему это еще и повышает удобство работы конечных пользователей. Поскольку адреса электронной почты часто применяются в качестве аутентификационных идентификаторов или резервных средств их восстановления, они становятся важной составляющей соответствия приложения требованиям универсального принятия в случае, когда оно задействует подобные механизмы.

## Проведение тестирования

Тестирование функциональных элементов анализируемого приложения проводится на основе построенной модели тестирования в соответствии с установленными ниже требованиями к каждому элементу на соответствующем этапе. Требования детализированы для каждого типа UA-идентификатора и предполагается, что необходимо проверить каждый из них. Однако если, к примеру, обработка электронной почты в приложении в принципе не предусмотрена, то тестирование осуществляется на предмет соответствия требованиям только к доменам.

### Требования к тестированию

AT: Тест принятия

- принятие любого IDN-домена в кодировке UTF-8, или в виде A-метки кодировки ASCII, или в смешанном представлении;
- принятие любого New gTLD домена на латинице длиной в 3 символа и более;
- принятие любого EAI адреса электронной почты в кодировке UTF-8;
- конвертация при вводе в формы A-меток в U-метки.

VT: Тест валидации

- проверка существования TLD в составе идентификатора с учетом IDN TLD и New gTLD;
- проверка на допустимую длину идентификатора;
- проверка использования допустимых символов в идентификаторе.

P1T: Тест обработки на входе

- приведение символов идентификатора к нижнему регистру;
- выполнение при необходимости нормализации по NFC;
- в процессе обработки идентификаторы не обрезаются;
- не осуществляется недопустимых преобразований идентификаторов.

ST: Тест хранения

- хранение идентификаторов осуществляется в кодировке UTF-8;

- для удобства индексации IDN-домен в составе идентификатора может храниться еще и в виде A-метки в качестве дополнения к U-метке;
- средства хранения не обрезают идентификаторы.

P2T: Тест обработки на выходе

- идентификаторы обрабатываются в кодировке UTF-8;
- в процессе обработки идентификаторы не обрезаются;
- в идентификаторы не было внесено любых иных изменений, за исключением нормализации.

DT: Тест отображения

- идентификаторы отображаются корректно и полностью в кодировке UTF-8;
- при наличии сортировки идентификаторов она происходит правильным образом;
- идентификаторы могут отображаться с использованием A-меток, если они отображаются в дополнение к соответствующим U-меткам.

Стоит помнить о нормализации строк, так как визуально одна и та же строка может кодироваться в кодировке Unicode по-разному. В стандарте Юникод есть несколько форм нормализации, если априорно неизвестно, какая именно форма используется в приложении, рекомендуется для тестирования принять использование формы C (NFC).

## Результаты тестирования

Приложение полностью удовлетворяет требованиям универсального принятия, если все его значимые функциональные элементы прошли все необходимые тесты в соответствии с принятой моделью тестирования с положительным результатом. Такое приложение считается полностью готовым к универсальному принятию и ему присваивается статус UA-Ready. Приложение, у которого не все результаты тестирования оказались положительными, но у значимых элементов положительных результатов большинство, считается готовым к универсальному принятию лишь частично. Приложение с незначительным количеством положительных результатов тестирования или с полным отсутствием таковых считается не готовым к универсальному принятию и требует соответствующей доработки. Значимость элементов определяется при построении модели тестирования. В общем случае очевидно, что элементы, взаимодействующие с пользователем, являются значимыми.

Примером достижения положительных результатов тестирования может послужить следующий набор тестов:

AT: Тест принятия

- принимаются все идентификаторы, как в кодировке UTF-8, так и в кодировке ASCII. Например, IDN-домен *тест.рф* в идентификаторе можно задать в виде A-метки *xn--e1aybc.xn--p1ai;*
- доменная часть идентификатора в виде A-метки «налету» конвертируется в U-метку, т.е., к примеру, при вводе в форму интерфейса *xn--e1aybc.xn--p1ai* она преобразуется в *тест.рф* средствами предобработки входных данных.

VT: Тест валидации

- TLD идентификатора существует. Проверка осуществляется при помощи актуального официального перечня доменов верхнего уровня (https://data.iana.org/TLD/tlds-alpha-by-domain.txt);
- максимальная длина идентификатора соответствует требованиям, причем с учетом длины A-метки его доменной части: 64 символа для локальной части (RFC 5321) и 255 символов на доменное имя - по 63 символа на каждый уровень в нем, начиная со второго (RFC 1035);
- в состав идентификатора входят только допустимые символы[2]:
    - символы в доменной части должны соответствовать правилам соответствующей зоны. Проверка может осуществляться DNS-запросом – если домен делегирован, то он соответствует этим правилам, или проверить можно, проанализировав сами правила;
    - символы локальной части идентификатора (при наличии таковой) могут включать спецсимволы: дефис(-), точку(.) и нижнее подчеркивание(_), цифры (от 0 до 9) и символы, принадлежащие к одному алфавиту. Кроме того, локальная часть не может начинаться и заканчиваться со спецсимволов и не может иметь два спецсимвола подряд. Проверка может осуществляться при помощи соответствующих регулярных выражений.

P1T: Тест обработки на входе

- идентификаторы преобразуются к нижнему регистру. Например, почта «Иванов@тест.РФ» преобразуется в «иванов@тест.рф»;
- идентификаторы нормализуются по форме C (NFC) в случае наличия в них символов, требующих нормализации;
- Доменные имена верхнего уровня длиннее трех символов в составе идентификатора не обрезаются до трех. Например, .online в .onl;
- Символы в идентификаторах не преобразуются для упрощения. Например, «ё» в «е»;
- Идентификаторы, полученные в результате обработки, представлены в кодировке UTF-8.

ST: Тест хранения

- в базе данных используется UTF-8 кодировка записей идентификаторов;
- если базе данных есть идентификаторы в виде A-меток, то им соответствуют записи в базе данных в виде U-меток;
- Средства базы данных не обрезают длинные домены верхнего уровня до трех символов.

[2] Приведенные в методике ограничения основаны на использовании лучшей практики для доменных имен и почтовых адресов. Стоит учесть, что в RFC 5322 указан более широкий перечень допустимых символов, однако на практике они используются крайне редко. Этот тест может выполняться как исходя из общих соображений, так и с учетом специфики правил конкретных реестров TLD.

P2T: Тест обработки на выходе

- доменные имена верхнего уровня длиннее трех символов в составе идентификатора не обрезаются до трех;
- символы в идентификаторах не преобразуются для упрощения;
- идентификаторы, полученные в результате обработки, представлены в кодировке UTF-8.

DT: Тест отображения

- все выводимые идентификаторы отображаются в кодировке UTF-8, все символы в идентификаторах отображаются корректно и полностью, с точностью до установленных у конечного пользователя шрифтов;
- если идентификаторы выводятся списком и применяется алфавитная сортировка, то она должна учитывать особенности алфавита, используемого в идентификаторах, т.е. они должны быть отсортированы корректно;
- идентификаторы могут выводится в виде A-меток, только в случае если они выводятся совместно с соответствующими U-метками, например с целью их сопоставления.

## Заключение

Предлагаемый подход к проверке соответствия требованиям универсального принятия состоит в следующем:

- Выделить базовые функциональные элементы приложения.
- Определиться с моделью тестирования.
- Проверить по выбранной модели соответствие элементов предложенным в методике требованиям универсального принятия.
- Тестирование функциональных элементов приложения проводить по принципу анализа «черного ящика», т.е. анализировать вводимые и выводимые элементом данные, не разбирая его программный код.
- Если результат тестирования функционального элемента оказался неудовлетворительным, попытаться увеличить его детализацию разбиением на более мелкие логические элементы и протестировать каждый из них, используя настоящую методику.

Следует отметить, что если какой-то элемент приложения или его составной части не соответствует требованиям универсального принятия, то это не всегда означает, что приложение в целом им не соответствует. Возможно, что соответствие приложения этим требованием достигается посредством остальных элементов приложения или путем использования специального расширения функционала проблемного элемента дополнительным элементом. Такие нюансы должны обязательно учитываться в модели тестирования.

Настоящая методика предназначена для тестирования программных приложений. В представленном виде она может быть неприменима в ряде других сценариев, таких как тестирование программных библиотек или компонент специфической инфраструктуры. Примером такого сценария может быть обмен SMS-сообщениями, который обеспечивает возможность передачи интернационализированных идентификаторов, но может не подойти для тестирования с использованием данной методики.

## Список использованных источников:


- Universal Acceptance Readiness Framework, https://uasg.tech/wp-content/uploads/documents/UASG026-en-digital.pdf
- Introduction to Universal Acceptance, https://uasg.tech/wp-content/uploads/documents/UASG007-en-digital.pdf
- Email Address Internationalization – Technical Perspective, https://uasg.tech/wp-content/uploads/documents/UASG019B-en-digital.pdf
- RFC 5321 - Simple Mail Transfer Protocol, https://tools.ietf.org/html/rfc5321
- RFC 5322 - Internet Message Format, https://tools.ietf.org/html/rfc5322
- RFC 1035 - Domain Names - Implementation And Specification, https://tools.ietf.org/html/rfc1035
- RFC 3492 - Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications https://www.ietf.org/rfc/rfc3492.txt
- Стандарт Internationalized Domain Names for Applications https://www.ietf.org/rfc/rfc5890.txt https://www.ietf.org/rfc/rfc5891.txt https://www.ietf.org/rfc/rfc5892.txt https://www.ietf.org/rfc/rfc5893.txt https://www.ietf.org/rfc/rfc5894.txt https://www.ietf.org/rfc/rfc5895.txt
- Стандарт EAI, интернационализации электронной почты https://tools.ietf.org/html/rfc6530 https://tools.ietf.org/html/rfc6531 https://tools.ietf.org/html/rfc6532 https://tools.ietf.org/html/rfc6533
- Поддерживаю.РФ – проект развития экосистемы поддержки доменных имен и адресов электронной почты на национальных языках https://поддерживаю.рф/
- Группа управления по универсальному принятию Universal Acceptance Steering Group, UASG https://uasg.tech/
- Юникод консорциум https://home.unicode.org/
- Формы нормализации Юникод http://www.unicode.org/reports/tr15/
- IANA Top Level Domains List https://data.iana.org/TLD/tlds-alpha-by-domain.txt