**Интерактивное распознавание**

Данная технология относится к компьютерным технологиям. Она позволяет распознать источник входящих сигналов (ИВС) посредством его взаимодействия с интерактивной виртуальной средой. Под интерактивной виртуальной средой подразумевается графическое или игровое программное обеспечение, используемое для распознавания ИВС.

Главная причина несовершенства существующих технологий распознавания ИВС – использование стандартной системы, работа которой вследствие человеческого фактора может быть нарушена.

Задача, которую данное предлагаемое техническое решение решает, является создание нового уровня предотвращения проблемы человеческого фактора в работе комплексной системы распознавания.

Данная задача решается в несколько этапов.

Во-первых, интерактивное распознавание позволяет произвести собственное внедрение в уже существующие решения, для создания нового/уникального метода распознавания в процессе работы с программной системой.

Во-вторых, интерактивное распознавание состоит из двух программных продуктов, которые создают закрытую программную систему распознавания используя архитектуру клиент-сервер.

При регистрации нового ИВС, система интерактивного распознавания передает ему шаблоны, которые случайным образом выбираются из базы данных, находящейся на стороне сервера. При получении шаблонов, ИВС имеет возможность выбора алгоритма взаимодействия с каждым из выданных интерактивных шаблонов.

Таким образом, после взаимодействия с каждым из шаблонов, на стороне ИВС система формирует защищенный слепок его уникального решения/взаимодействия с паттернами.

**Структура системы**

**PART I**

Система состоит из следующих частей: **Блока 1 (Клиент)**, **Блока 2 (Сервер)** и **Кодировщика**.

**Блок 2** служит для хранения, обработки интерактивной информации, а также для взаимодействия посредством запросов с **Блоком 1**.

**Блок 1** представляет собой интерактивное приложение, которое получает данные с **Блока 2** и визуализирует их, ожидая определенных входных данных для подтверждения правильного взаимодействия с интерактивными данными.

**Кодировщик** работает в качестве системы запрос-ответ, и позволяет кодировать данные во время их передачи между **Блоком 1** и **Блоком 2**. Ключи для кодирования **Кодировщик** получает от **Блока 2** по запросу.

Рассмотрим детальнее алгоритм работы системы.

Для работы системы используется три внутренних компонента – **Блок 1**, **Блок 2**, и **Кодировщик**.

Первый запуск считается инициированным от имени владельца системы и авторизованного источника внешнего сигнала.

Сперва система запускает **Блок 2**, который после запуска инициализирует локальное хранилище данных, и переходит в статус ожидания запросов от **Блока 1**.

На втором этапе система запускает внутренний **Кодировщик**. Это отдельное приложение. После запуска, **Кодировщик** становится в статус ожидания запросов от **Блока 1** и **Блока 2** для кодирования/декодировки запросов.

На третьем этапе система запускает внутренний **Блок 1**. После запуска, **Блок 1** отправляет запрос **Кодировщику** с пометкой «Первый запуск» и свой уникальный ключ. **Кодировщик** закрывает запрос защитным ключом и переправляет его **Блоку 2**.

**Блок 2** получает запрос от **Блока 1**, декодирует его **Кодировщиком**, считывает ключ и проверяет его у себя в локальной базе данных. Если **Блок 1** с данным уникальным ключом запускается первый раз для данного пользователя, тогда **Блок 2** выбирает из **Базы Паттернов** случайный **Интерактивный Паттерн Распознавания**, собирает из него пакет данных и посылает **Кодировщику**. **Кодировщик** кодирует пакет данных и пересылает их **Блоку 1.**

**Блок 1** получает от **Кодировщика** пакет данных, раскодирует их с помощью ключа распознавания, и считывает **Интерактивный Паттерн Распознавания**.

**Интерактивный Паттерн Распознавания** представляет собой структуру данных, которая содержит расположение интерактивных элементов на устройстве вывода информации. При считывании **Интерактивного Паттерна Распознавания,** **Блок 1** забирает данные о размещении интерактивных элементов, и расставляет их на **Интерактивной сцене**.

**Интерактивная сцена** представляет собой набор статических и динамических элементов на экране устройства, с которой взаимодействует система на этапе распознавания.

После расстановки интерактивных элементов на экране устройства вывода, **Блок 1** переходит в режим ожидания новых входных данных.

Во время взаимодействия с интерактивными элементами на экране устройства (**N**), система формирует определенный порядок перехода от одного интерактивного элемента к другому интерактивному элементу (**M**). Данный порядок **Блок 1** сохраняет в последовательность данных. После завершения взаимодействия с **Интерактивным Паттерном Распознавания**, **Блок 1** формирует пакет данных из собранных входных данных о последовательности действий и посылает его **Кодировщику.**

**Кодировщик** получает эти данные, кодирует их с помощью ключа распознавания и передает на **Блок 2.**

**Блок 2** получает кодированный пакет данных от **Кодировщика**, раскодирует их, и получает готовую последовательность взаимодействия входных данных системы с **Блоком 1**. Данную последовательность **Блок 2** сохраняет в **Базе Источников Внешнего Сигнала** с уникальным ключом, который получает посредством функции кодирования, куда передается уникальный идентификатор системы и дата ее регистрации.

**Блок 2** создает пакет данных о распознавании **Источника Входящего Сигнала**, высылает его **Кодировщику**, который пересылает данный уже кодированный пакет данных **Блоку 1.**

**Блок 1** получает кодированный пакет данных от **Кодировщика,** раскодирует их с помощью ключа распознавания, и дает уведомление о регистрации **Источнику Входящего Сигнала**.

**PART II**

При повторном запуске система запускает **Блок 2**, который после запуска инициализирует хранилище данных, и переходит в статус ожидания запросов от **Блока 1**.

На втором этапе система запускает внутренний **Кодировщик**. После запуска, **Кодировщик** становится в статус ожидания запросов от **Блока 1** и **Блока 2** для кодирования/декодировки запросов.

На третьем этапе система запускает внутренний **Блок 1**. После запуска, **Блок 1** отправляет запрос **Кодировщику** с пометкой «Запуск» и свой уникальный ключ. **Кодировщик** закрывает запрос защитным ключом и переправляет его **Блоку 2**.

**Блок 2** получает запрос от **Кодировщика**, декодирует его, считывает ключ и проверяет его на совпадение в **Базе Источников Внешних Сигналов**. Если уникальный ключ найден, **Блок 2** забирает из **Базы Источников Внешних Сигналов** по этому ключу данные об **Интерактивном Паттерне Распознавания**, передает его **Кодировщику**, который в свою очередь его кодирует, и перенаправляет **Блоку 1.**

**Блок 1** получает данные, раскодирует их с помощью **Кодировщика** и ключа распознавания, и на основе этих данных (**N**) строит интерактивную сцену на экране устройства. После построения интерактивной сцены **Блок 1** переходит в ожидание входных данных (**M**).

При взаимодействии **ИВС** с интерактивными элементами на экране устройства, создаются входные данные (**M**), которые подаются на **Блок 1. Блок 1** принимает эти данные, упаковывает свой уникальный идентификатор и собранные данные, передает их **Кодировщику**, который в свою очередь их кодирует, упаковывает и передает **Блоку 2.**

**Блок 2** получает данные (**M**), раскодирует их с помощью **Кодировщика**. С помощью уникального идентификатора, который **Блок 2** получил в текущем пакете данных от **Кодировщика**, **Блок 2** достает из **Базы Источников Внешних Сигналов** данные о правильном взаимодействии с **Интерактивным Паттерном Распознавания,** и сравнивает с данными о взаимодействии с **Интерактивным Паттерном Распознавания,** полученном от **Блока 1.**

**Блок 2** проверяет обе последовательности на совпадения по размеру и по порядку следования данных (**Триггер**), и передает сообщение **Кодировщику,** который кодирует это сообщение и перенаправляет **Блоку 1.**

**Блок 1** получает сообщение от **Блока 2,** передает его **Кодировщику**, который с помощью уникального ключа распознавания раскодирует его, и передает обратно **Блоку 1.**

**Блок 1** анализирует раскодированное сообщение и, в случае совпадения, **Блок 1** переходит в состояние, при котором разрешается продолжение работы. Если **Блок 1** получил сообщение о несовпадении, **Блок 1** переходит в состояние ожидания входных данных.