Release notes 02.002.00 (iNavigator)

# Механизм предварительного ручного контроля и просмотра данных

Данный механизм добавляет в iNavigator возможность генерации базы данных для ограниченной группы пользователей, и только после проверки – публикацию на всю аудиторию.

Ключевые понятия:

**Черновик** – SQLITE БД для приложения, которая доступна ограниченному кругу пользователей

**Чистовик(опубликованная версия)** - SQLITE БД для приложения, которая доступна всем пользователям

Сценарий использования:

1. Администраторы БД настраивают список пользователей, которым доступен предварительный просмотр
   1. Необходимо заполнить таблицу MIS\_IPAD\_GENERATOR..GENERATOR\_GROUP\_USERS используя следующие SQL вызовы в БД генератора:

exec ADD\_USER\_TO\_DRAFT\_GROUP ‘application’,’email@sberbank.ru’

1. Пользователь заходит в административную часть генератора и переходит в пункт задания.
2. Для требуемого файла задания выставляет режим генерации “Черновик”.
3. Запускает генерацию.
4. После успешной генерации и передачи в сигму, файл будет доступен пользователям добавленным на шаге 1 для текущего приложения.
5. Генерировать черновик можно неоднократно.
6. После того, как база данных была успешно проверена администратор нажимает на кнопку опубликовать ( зеленая птичка). В течение минуты в произойдет публикация в сигме и все пользователи увидят опубликованную БД.

Возможные режимы генерации:

**Черновик** – при ручном/автоматическом запуске генерируется черновик, доступный ограниченному кругу пользователей

**Публикация** – сгенерированная база становится доступна всем пользователям.

Статусы публикации:

**Не определено** – в системе нет информации о сгенерированных черновиках/чистовиках.

**Черновик** – в системе содержится информация о сгенерированном черновике

**Опубликованно/не актуально** – В системе содержится информация об опубликованном чистовике, но одновеременно с этим есть неопубликованный черновик

**Опубликованно/актуально** – В системе содержится информация об опубликованном чистовике, не опубликованного черновика нет

Краткое описание работы механизма:

1. На стороне альфа –генератора:
   1. Данные и MD5 опубликованного файла и черновика хранятся в соответствующих полях таблицы SYNC\_CACHE\_STATIC\_FILES. При публикации – md5 опубликованного файла замещается значением md5 черновика.
2. На стороне сигма-кешсервера
   1. Запущен поток, который раз в минуту кеширует в сигме содержимое таблицы SYNC\_CACHE\_STATIC\_FILES и GENERATOR\_GROUP\_USERS.
   2. Сравнивая MD5 пришедшего файла с MD5 закешированных значений из альфы, механизм определяет какой статус у сгенерированного файла.
   3. Публикация в альфе ( замещение MD5) приводит к удалению черновика в fileCache и переводе его в чистовик.

# Оптимизация административных страниц для работы с сервисами

Список изменений административной части:

1. В логи работы сервисов добавлен журнал, позволяющий отображать логи в табличном виде с возможностью пейджинга, фильтрации и выгрузки в Excel.
2. Доработана страница настроек во всех приложениях – сохранение происходит без перезагрузки страницы.
3. Доработана система управления сервисами. Пуск/старт/обновление свойств происходит без перезагрузки страницы

# Реализация модуля центр аудита

Данный механизм встроен во все серверные приложения iNavigatorа и позволяет определить какие действия совершал администратор в системе. Интерфейс состоит из журнала с возможностью фильтрации, пейджинга и выгрузки сообщений аудита. В систему аудита попадают сообщения о:

1. Входе/выходе пользователя из системы
2. Изменении настроек
3. Запуске/остановки сервисов
4. Работе с пользователями
5. Изменения режима логирования мобильных приложений
6. Работе с заданиями в генераторе

# Реализация механизма Push Notifications

Данный механизм позволяет отправлять PUSH уведомления(короткие текстовые сообщения) пользователям системы в автоматическом и ручном режиме.

Работа механизма PUSH уведомлений состоит из следующих этапов:

1. При добавлении возможности PUSH уведомлений в очередное приложение, разработчик выпускает ключ+сертификат на портале Apple
2. Сертификат и ключ добавляются на сервер (в корень папки PUSH\_CERTIFICATE\_CONFIG\_FOLDER)
   1. В корне папка IOS ( код операционной системы)
   2. Внутри подпапка с названием приложения (например Balance)
   3. Внутри два файла \_password( текстовый файл с паролем) и файл с расширением \*.p12 – контейнер ключа и сертификата

**Внимание файл p12 и пароль должны быть получены от разработчиков для каждого приложения поддерживающего PUSH-уведомления. На текущий момент поддержка встроена в “Конкуренты”**

1. При запуске, приложение автоматически регистрируется на портале Apple и получает clientToken
2. Приложение вызывает сервис syncserver-а, передавая ему свой clientToken, Email и прочую информацию о приложении.
3. Далее администратор заходит на страницу уведомлений, выбирает группу пользователей ( фильтруя ниспадающим фильтром с кода операционной системы до выбора приложения и версии или выбора email пользователя или конкретных его устройств)
4. Пользователь вводит сообщение и нажимает кнопку “Отправить”, после чего сообщение попадает в очередь
5. В течение минуты сообщение будет отправлено на сервера apple с указанием clientToken.
6. Apple автоматически произведет рассылку сообщения всем требуемым адресатам.

# Реализация модуля центр логирования (клиентские логи и простая серверная часть)

Данный механизм позволяет запрашивать логи с мобильного устройства налету, сохраняя их на сервере. Функция доступна в confserver ( страница “Логирование”).

Для активизации режима логирования необходимо

1. Добавить email пользователя через кнопку добавления (+). При этом имя устройства можно оставить пустым.
2. Активировать режим сбора логов через выбор комбо-бокса в списке пользователей.
3. Перейти в журнал логов, нажав на ссылку-email пользователя.
4. В журнале в реальном времени будут появляться логи с устройства. После анализа необходимо выключить режим логирования (сбросить чекбокс в пункте 2)

**Внимание! Не рекомендуется оставлять надолго включенным режим сбора логов, т.к. это будет создавать дополнительную нагрузку на сервер.**

# Внедрение сервера сбора аналитической статистики

Данный сервер сбора статистики служит для промежуточного сбора статистических сообщений и пересылки их во внешние аналитические системы.

Система состоит из трех основных частей:

1. Клиентский SDK (в данный момент используется Countly Community SDK)- набор средств разработчика, для внедрения сбора статистики в мобильное приложение
2. Промежуточный веб сервер – statistical\_server.war – веб приложение, которое собирает статистические сообщения от клиентов и периодически отправляет их во внешние системы через механизм адаптеров выгрузки.
3. Внешняя аналитическая система – система, в которую выгружаются статистические сообщения, и далее отображаются в удобном для анализа визуальном виде.

На текущий момент, в качестве внешней аналитической системы используется google analytics. Промежуточный сервер хранит временно полученные сообщения в памяти и периодчески выгружает их во внешнюю систему.

Подробности см. в документе StatisticalServerAdminGuide.docx.

**Внимание! В текущей версии функциональность выгрузки статистической информации на внешние сервера отключена. Данные остаются на внутренних серверах приложений. Она будет включена в одном из следующих релизов.**

# Диагностика соединения между IPAD сервером и сервером Sigma

Данная утилита представляет собой мобильное приложение для IOS, которое позволяет диагностировать возможные проблемы на пользовательском устройстве, а именно:

1. Доступность Интернет
2. Доступность Wi-Fi
3. Доступность Configuration Server
4. Проверка версии и приложения
5. Доступность серверов данных iNavigator
6. Проверка валидности серверного сертификата
7. Проверка валидности клиентского сертификата

# Передача single-файлов по фрагментам

На этапе LocalDeflater сгенерированный sqlite файл разбивается на набор фрагментов, каждый из которых не превышает размер FILE\_FRAGMENT\_SIZE\_MB(поле в настройках генератора) и далее отправляется в сигму.

Каждый фрагмент представляет из себя zip-архив(с раширением \*.fzip), содержащий:

1. Файл с названием приложения.sqlite с бинарным фрагментом исходного целого файла.
2. Файл metainfo – проперти файл с набором служебных полей ( количеством фрагментов, гуидом файла, датой актуальности и т.д.) позволяющими корректно склеить полный файл в сигме, собрав все его фрагменты

Склейка фрагментов в единый файл происходит в сигме на этапе LocalInflater.