О применении библиотеки FastScript в своих проектах.
Цикл статей.
Основные тезисы

Используемые сокращения:

FS – FastScript;
FSI – Интерпретатор скриптов (в части Обвязки для FastScript);
GUI – Графический Пользовательский Интерфейс;
БД – База Данных;
ИС – Информационная Система.

Использование библиотеки FastScript (в том числе) является пожалуй, наилучшим решением при разработке различных по назначению и сложности проектов (включая, в том числе, и объектно-ориентированные, гибкие информационные системы), когда необходимо сформировать гибкий программный инструментарий для автоматизации предметных областей без концептуального ограничения как на перечень «охватываемых» предметных областей, так и на градиент изменения условий функционирования программного инструментария в процессе его эксплуатации.

Существенная часть задач, где может быть применена библиотека FastScript, это автоматизация выполнения различного рода прикладных (специфичных) задач конечного Пользователя − самим же Пользователем (в рамках эксплуатируемой ИС).

При этом, Пользователь (даже очень продвинутый), как правило НЕ является программистом со «всеми вытекающими». Чем более дружественным (комфортным) для него будет инструментарий, тем эффективнее он будет его применять.

Одним из факторов («облегчающих жизнь» конечному Пользователю), является максимально возможная локализация программного инструментария не только в части GUI, но и в части идентификаторов функций, процедур, переменных и констант (что при использовании FastScript легко реализуемо).

В силу того, что возможности библиотеки FastScript существенно велики, а «аппетит растет во время еды», то через довольно-таки небольшой промежуток времени количество «специальных» функций (которые добавляет Программист) может стать знАчимо «большим» (особенно, в «больших» ИС).

И в этом случае необходимо сразу предусматривать возможность группировки дополнительного функционала по отдельным модулям (которые будут подключаться по мере необходимости в процессе разработки конкретного скрипта или группы скриптов).

В ряде случаев, когда FastScript применяется в гибких ИС, тексты FS-скриптов хранятся в таблицах БД и загружаются для выполнения периодически, согласно принятого регламента, или ситуативно, по «требованию» внешних инициаторов информационного обмена.

«Требования» инициаторов информационного обмена могут включать некие исходные данные, которые должны быть учтены при выполнении FS-скрипта (входные параметры FSI), и возвращаемые данные, полученные в результате выполнения FS-скрипта (вЫходные параметры FSI).

Входные параметры FSI могут быть преобразованы в константы FastScript, а вЫходные параметры FSI – в переменные FastScript.

Т.е., некий функционал (из состава Обвязки) конвертирует поступившие входные параметры – в константы FastScript, а вЫходные параметры – в переменные FastScript.

А затем, соответствующий FS-скрипт выгружается из БД и (используя компонент `TfsScript` библиотеки FastScript) запускается на выполнение.

После того, как FS-скрипт выполнен, значения соответствующих переменных конвертируются Обвязкой в вЫходные параметры (для предоставления их инициатору информационного обмена в соответствии с требованиями).

В ряде случаев, когда FastScript применяется в гибких, объектно-ориентированных, распределенных (в том числе и иерархически) ИС, тексты FS-скриптов однозначно хранятся в соответствующих таблицах БД и участвуют в миграции данных в составе информационных потоков, предусмотренных соответствующими регламентами.

Кроме этого, определенные FS-скрипты могут реализовывать выполнение одних и тех же алгоритмов, но с специфическими исходными данными (зависящими от конкретных узлов распределенной сети ИС).

Как следствие:

идентификация FS-скриптов должна носить глобальный характер (для всей сети ИС);

процесс выгрузки взаимосвязанных FS-скриптов не должен быть привязан к конкретностям файловых систем.

Список статей цикла:

Часть-1. Расширение функционала FS (локализация идентификаторов программных объектов);

Часть-2. Входные и вЫходные параметры FS-скрипта (при информационном обмене с внешними инициаторами);

Часть-3. Группировка (распределение по модулям) добавляемых в FS программных объектов;

Часть-4. Парсинг раздела "Uses" FS-скрипта с целями: 1. "отвязки" от "абсолютного пути" в именах файлов; 2. корректной выгрузки библиотечных FS-скриптов, хранящихся в ИБ, и "подключения" их к "вызывающему" FS-скрипту.