Интеграционная шина Help-Pro

Введение

Документ описывает общие аспекты интеграции приложений Help-Pro через интеграционную шину.

Термины

Примечание: *Термины написаны так, как они употребляются в тексте. Некоторые пишутся с большой буквы, как имена, некоторые -- с маленькой, как обычные слова.*

- **Адаптер** -- библиотека, позволяющая Приложению общаться с другими Приложениями через Шину.
- **Интеграционная шина**, **Шина**, **service bus** -- коммуникационная среда, обеспечивающая интеграцию приложений Help-Pro.
- **Макросистема** -- совокупность Узлов, осуществляющих взаимодействие через Шину. Пример: “Система-1” + “ПроСтор” + “ПроЛог”.
- **Паттерн** -- 1) тип взаимодействия через Шину; 2) роль Узла при взаимодействии через Шину. См. Паттерны.

- **Приложение** -- совокупность функций в рамках решения набора логически связанных задач. Примеры: “Система-1”, “Простор”.
- **Сообщение** -- посылка данных, переданная через Шину. См. Сообщения.
- **Узел** -- инсталляция Приложения, взаимодействующая с Узлами других Приложений через Шину. Узел реализует в себе один или несколько Паттернов.

Требования к Шине

Вне зависимости от выбранного решения Шина должна соответствовать следующим требованиям:

1. “Compatibility”. Должен быть надёжный доступ из PHP, Ruby и других языков, на которых мы будем разрабатывать в ближайшие 3-5 лет.
2. “Security”. Необходимы механизмы аутентификации Узлов, защиты трафика от перехвата. См. пояснения.
3. “Patterns”. Режимы взаимодействия: request-reply (REQ-REP), publish-subscribe (PUB-SUB).
4. “Parallel”. Для запросов REQ-REP должна присутствовать возможность распараллеливания обработки на несколько процессов/потоков.
5. “Ease of use”. Создание Узла должно быть очень простой для разработчика приложения задачей.
6. “Stateless and trusted”. Взаимодействие Узла Шиной должно происходить на stateless-основе, а также сообразно принципу, что Шина -- проверенное (trusted) и безопасное место. См. пояснения.
7. “Connectivity”. **Не должно** быть обязательным требованием, чтобы все до единого Узлы находились в одной доверенной сети. См. пояснения.
8. “Testability”. Логика работы с Шиной и логика генерации/потребления фактических сообщений **должны быть разделены** чтобы допускать возможность тестирования, отладки, установки заглушек и пр.

Пояснения

Security:

- Голой IP-безопасности недостаточно. Предполагаем, что главный message broker может находиться в не-trusted сети (в Интернете).

Stateless and trusted:

- Узел не должен перед каждым запросом делать миллион проверок, шифрований и дешифрований.

Connectivity:

- **То есть** вполне допустимо, чтобы Узлы находились в разных сетях и при этом общались через экземпляр Шины, который существует “в Интернете”.
- Остальные требования по безопасности и т.д. должны соблюдаться.

Сообщения

На сегодня существует 3 типа Сообщений:

- запрос (request);
- ответ (reply);
- уведомление (notice).

Запрос и ответ относятся к Паттернам (см. Паттерны) REQ-REP client и REQ-REP server.

Уведомление относится к Паттернам PUB-SUB publisher и PUB-SUB subscriber.

Паттерны

На сегодня поддерживается 4 Паттерна:

- REQ-REP client (тж. “Клиент”),
- REQ-REP server (тж. “Сервер”),
- PUB-SUB publisher (тж. “Publisher”),
- PUB-SUB subscriber (тж. “Subscriber”).

Можно также говорить о составных Паттернах: REQ-REP и PUB-SUB.

REQ-REP client

Узел отправляет запросы одному или нескольким REQ-REP серверам, получает и обрабатывает их ответы.

REQ-REP server

Узел принимает из Шины запросы от REQ-REP клиентов, обрабатывает их, и отправляет клиентам ответы.

PUB-SUB publisher

Узел отправляет в Шину уведомления.

PUB-SUB subscriber

Узел принимает из Шины уведомления и обрабатывает их ведомым ему образом.

(Недописанное/размышления/мусор)

Сценарии прототипирования

В основном для скринкаста.

Case 1

Узлы Макросистемы:

- Узел **samba_ctl**.
  Сервис. Имитирует интерфейс управления пользователями Samba-сервера.
- Узел **reverser**.
  Сервис. Переворачивает слово, полученное в запросе, задом наперёд.
- Узел **console**.
  Управляющая консоль. Позволяет генерировать запросы в адрес серверных Узлов.
- Узел **dashboard**.
  Получает из Шины широковещательные уведомления от Узлов и отображает их на экране.
- Узел **samba_mon**.
  Получает из Шины широковещательные уведомления от Узлов **samba_ctl**, запрашивает от **samba_ctl** дополнительную информацию и отображает её на экране.

Цепочка запросов и ответов при samba_mon (для скринкаста):

1. **console** отправляет запрос `CreateUser` на **samba_ctl**.
2. **samba_ctl** отправляет уведомление `UserCreated` (широковещательно).
3. **samba_mon** получает уведомление `UserCreated`.
4. **samba_mon** отправляет запрос `ListUsers` тому Узлу **samba_ctl**, который отправил уведомление.
5. **samba_mon** делает полезную работу, получив ответ на `ListUsers`.