[JBR] Иерархическое управление задачами и голосования

Задачи

- Разработать модель и реализацию системы осуществления иерархического выполнения задач и контроля результатов.
- При помощи этой системы построить систему, которая будет реализовывать голосовалку по модели "жидкой демократии" (https://medium.com/organizer-sandbox/liquid-democracy-true-democracy-for-the-21st-century-7c66f5e53b6f).
- Реализовать систему в виде веб-сервиса и клиентского плагина для Slack.

Смысл

Исходная задача - это создать систему голосования, в которой каждый голосующий может сделать это сам или делегировать принятие решения по каждому конкретному вопросу или по всем вопросам, касающихся определенной сферы другому участнику голосования (или вообще внешней сущности). Эту задачу можно переформулировать следующим образом: в рамках системы есть заказчик, который заказывает опрос и спектр исполнителей (опрашиваемых), при этом каждый исполнитель может делегировать работу другому исполнителю (в этом случае исполнитель становится заказчиком субподряда; делегирование может происходить сколько угодно раз). При этом заказчика не волнует, кто именно выполнит работу (проголосует). Реализованную таким образом систему можно обобщить для выполнения произвольных задач и контроля их выполнения, поэтому она представляет интерес не только в смысле голосования, но и в смысле учета ресурсов и организации распределенных вычислений.

В смысле голосования система является полезнойдля получения информации не только о мнении контингента, но и о "кластеризации" голосующих и о наличии "лидеров общестенного мнения".

План

1. Разработать общую логическую модель пользователя, модель заказа и отчета о выполнении работы (поданного голоса)
2. Разработать модель передачи сообщений между пользователями:
    a. Первичный запрос
    b. Переадресация работы (постоянная или разовая)
    c. Подтверждение переадресации
    d. Отчет о выполненной работе
    e. Валидация отчета заказчиком (транзитивная)
3. Разработка REST-сервиса на основе этой модели.
4. Разработка клиента на Slack.

Не план

- Делать блокчейн
- Делать внутреннюю криптографическую защиту (на первом этапе)
- Делать защиту от махинаций со сторны заказчика

Технология

- Для прототипирования и сервиса будем использовать Kotlin.
- Возможно будем делать спецификацию на Swagger/OpenAPI.
- Предыдущий опыт с API Slack приветствуется.

Вопросы

- Какая модель пересылки сообщений? Централизованная (переадресация через первичного заказчика) или децентрализованная (субподряд - забота исполнителя)?
- Должен ли заказчик знать всю цепочку исполнителей и должен ли исполнитель иметь возможность ее скрыть?