

Constellation Light Paper

Введение	1
Эволюция от синхронных консенсусов к асинхронным	1
Консенсус, основанный на репутации	4
Частичное/полное масштабирование узла	4
Hylochain — Поддержка приложений на основе каналов	5
Токеномика и её связь с Hylochain	7

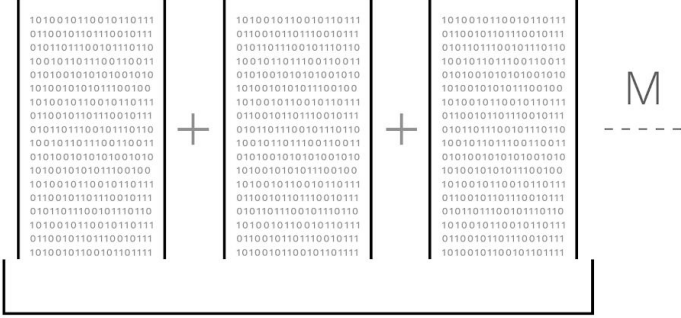
Автор: Wyatt Meldman-Foch

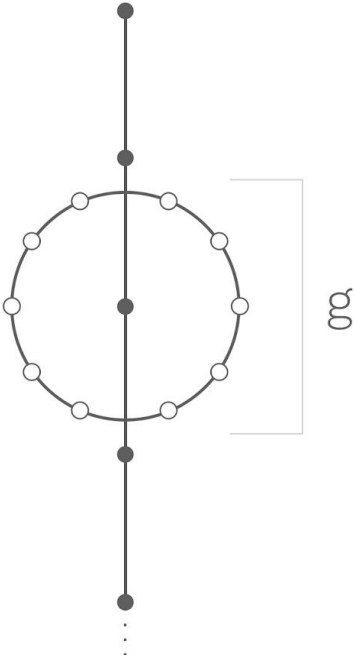
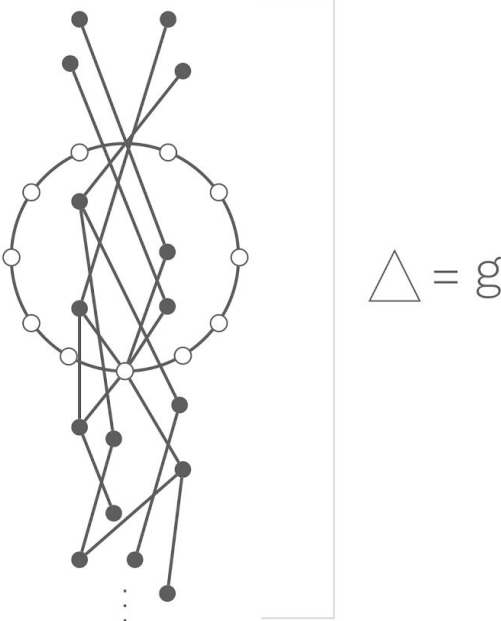
Введение

Всё нижеизложенное описывает основные технологические компоненты протокола Constellation и наглядно иллюстрирует механику его работы. Полное разделение архитектуры и алгоритмов будет в техническом документе, который мы выпустим ближе к запуску публичной сети.

Эволюция от синхронных консенсусов к асинхронным

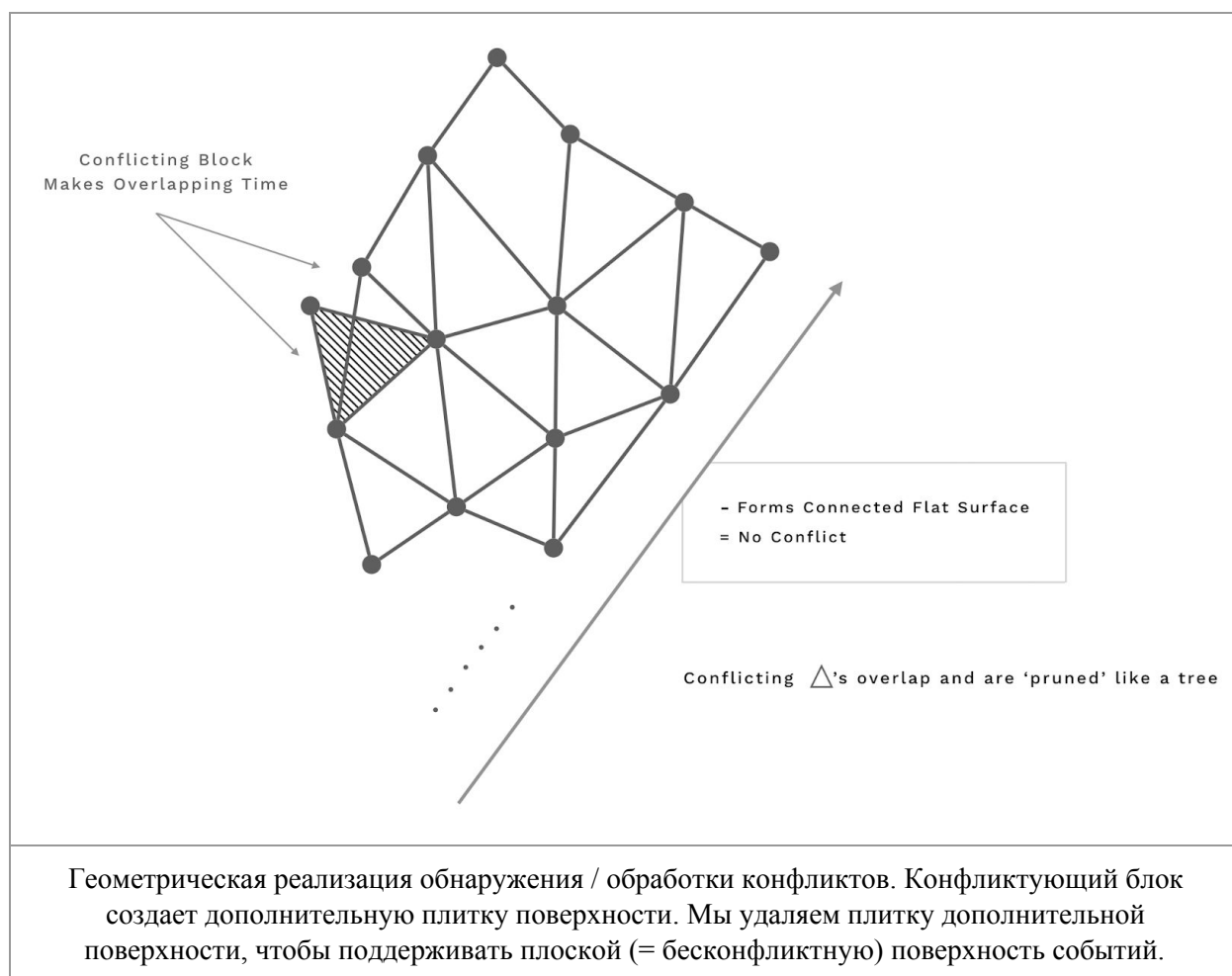
Узлы выбираются с использованием детерминированного процесса (того же, который используется в DHT, например, bittorrent), который динамически регулирует обязанности узлов для “облегчения” валидации или, что более понятно, для достижения консенсуса. Мы выбираем группы из 3 узлов и выполняем раунды консенсуса параллельно, чтобы один узел мог быть фасилитатором в нескольких блоках. Это позволяет нам обрабатывать транзакции асинхронно, что, по сути, означает, что у нас одновременно формируется несколько блокчейнов. Процесс подобен паутине, образованной множеством нитей, в отличие от узлов, формирующих одну цепочку с течением времени. Асинхронная или параллельная обработка являются основой масштабируемого программирования, поскольку оно позволяет использовать все ресурсы компьютера, ускоряя общие вычисления. Эта сеть называется ориентированным ациклическим графом или DAG в компьютерных науках.

 $g = \text{AREA OF PIPE}$	 $= M \cdot g = M \cdot \text{AREA OF PIPE}$
<p>BLOCKCHAIN</p> <p>Pipe width of a linear blockchain</p>	<p>DIRECTED ACYCLIC GRAPH</p> <p>Multiplicative effect of a DAG allowing for multiple concurrent blockchains</p>
<p>Ширина канала линейного блокчейна против мультипликативного эффекта DAG, где у нас есть несколько параллельных блокчейнов.</p>	

 <p>Linear Blockchain Geometric realization of a linear Blockchain</p>	 <p>Hierarchically Partitioned DAG Geometric realization of a Directed Acyclic Graph</p>
<p>Геометрическая реализация линейного блокчейна против DAG. Черные точки — это блоки,</p>	

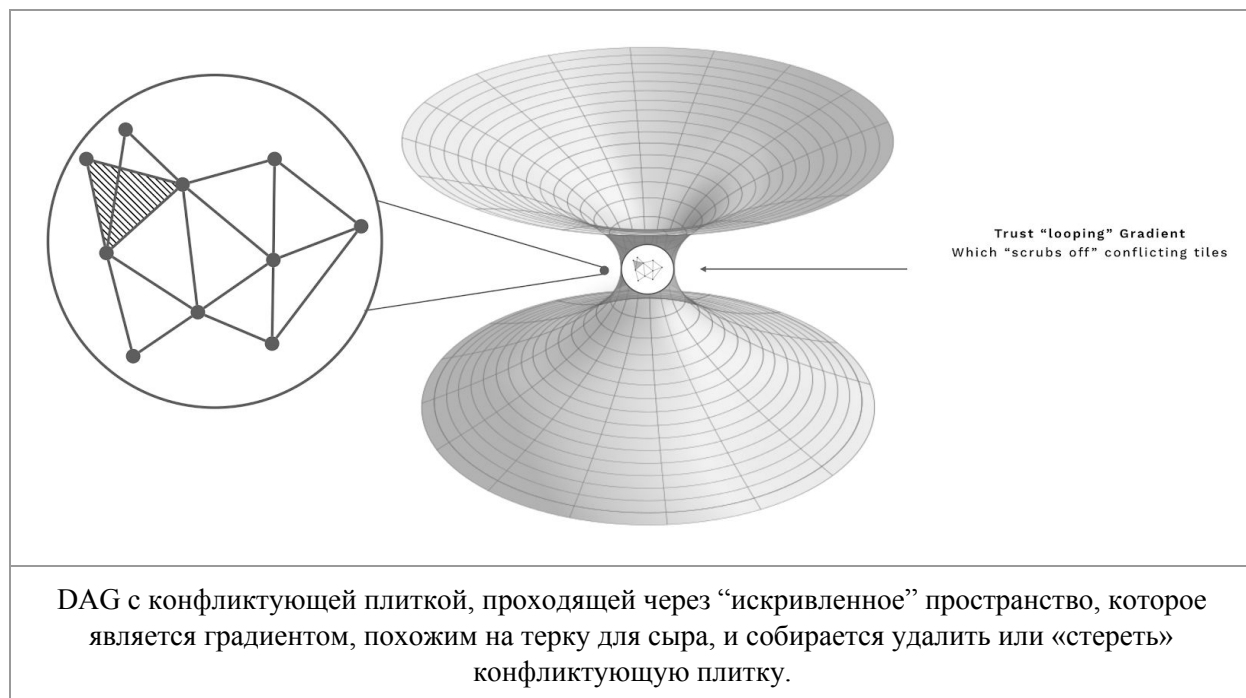
белые точки — это узлы

Мы используем 3 узла в каждом раунде консенсуса, потому что это дает нам некоторые интересные математические процессы для рассуждения о состоянии, формируя «плоскость поверхности» поперек данных в форме треугольников со связями. Затем протокол использует треугольники для «сшивания» оптимальной поверхности, которая не содержит избыточных или противоречивых данных и имеет минимально возможные треугольники. Алгоритмически — это аналогично «минимальному разрезу» графа, а математически — производной или функции оптимизации (из которых функция находит кратчайший путь, который она может пересечь по поверхности). Этот кратчайший путь эквивалентен оптимальному хранению данных (транзакций) в группе обеспечения доступности баз данных. Конфликтующие треугольные «плитки», чтобы поверхность события была ровной и без от конфликтов.



Консенсус, основанный на репутации

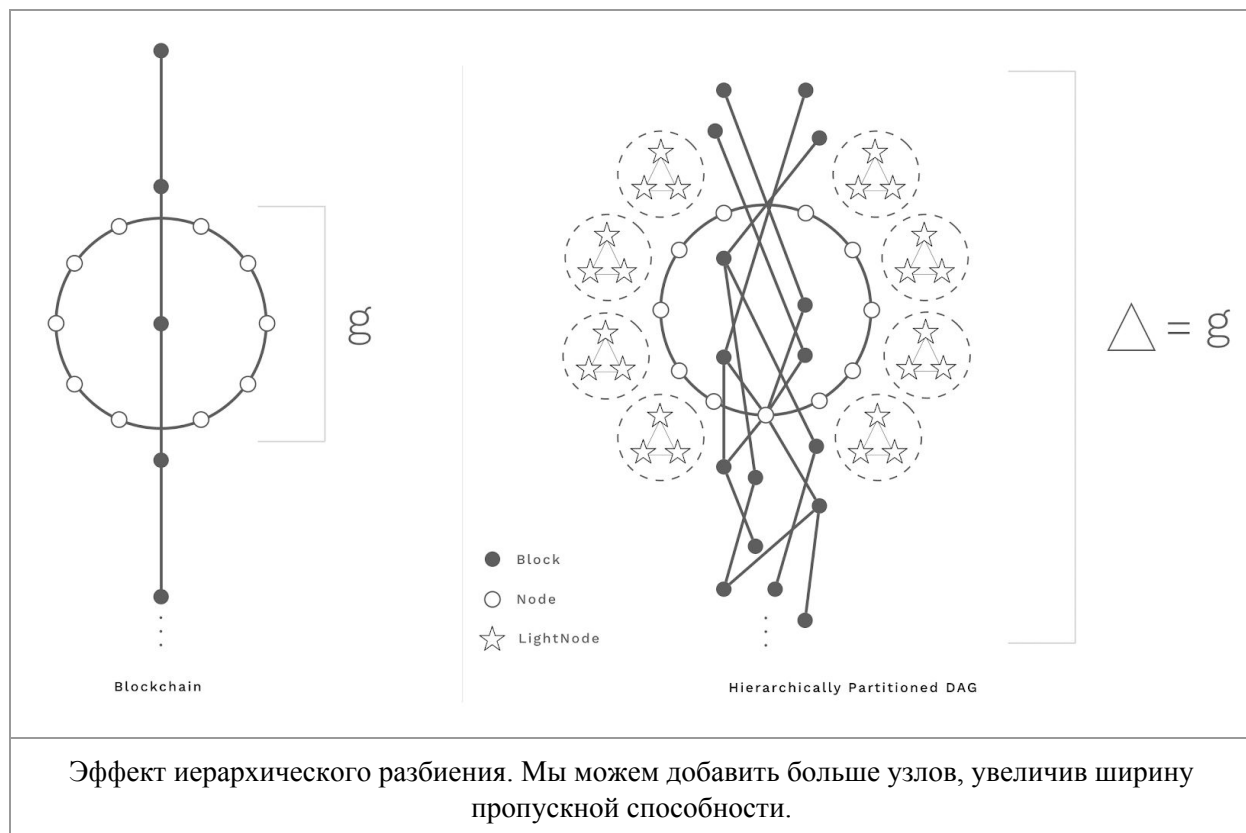
В оптимальной децентрализованной p2p системе репутации каждый узел должен иметь возможность самостоятельно определять свое доверие к другим узлам. Наша система использует специальную модель, которая включает транзитивные отношения или отношения, которые узел имеет с другими узлами, при назначении глобальной оценки. “Вы так же хороши, как и ваша компания”. Конечным результатом является “искажение” или градиент, основанный на транзитивном доверии или репутации во всех узлах в \$DAG или штатном канале. Это можно рассматривать как кисть или терку для сыра, которая стирает поверх “плоскости поверхности” и выбирает, какие “треугольные плитки” стереть, а какие оставить. Вот как логика конфликта на самом деле удаляет “треугольные плитки”.



Частичное/полное масштабирование узла

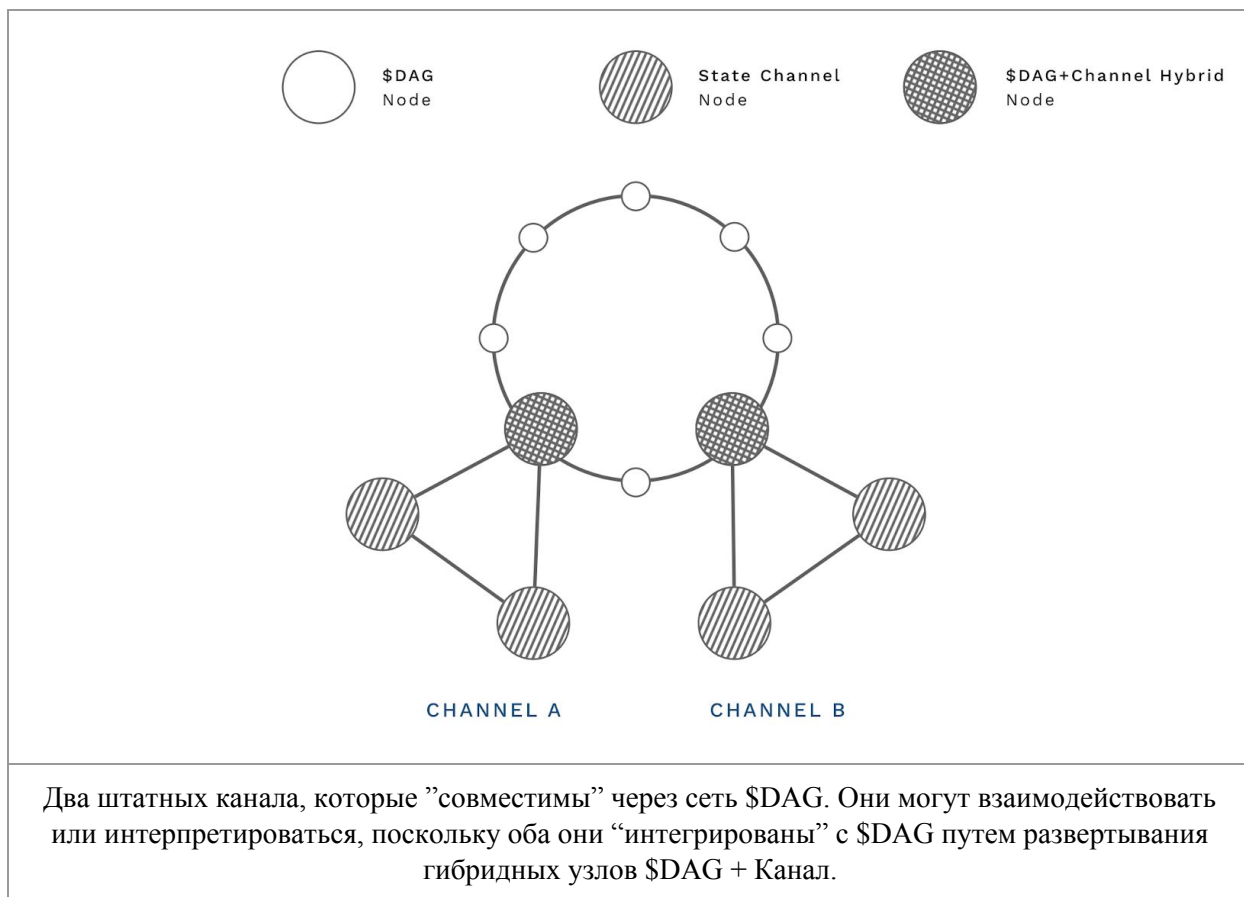
В теории сетей, как правило, оптимальное распределение известно как “без масштабирования”, которое может быть описано как иерархическое расположение с большими центральными узлами, управляющими многими более мелкими периферийными узлами. Это распределение видно в

природе и, прежде всего, в Интернете. Constellation использует эту архитектуру для “масштабирования”, или увеличения пропускной способности или ширины нашего Графа.

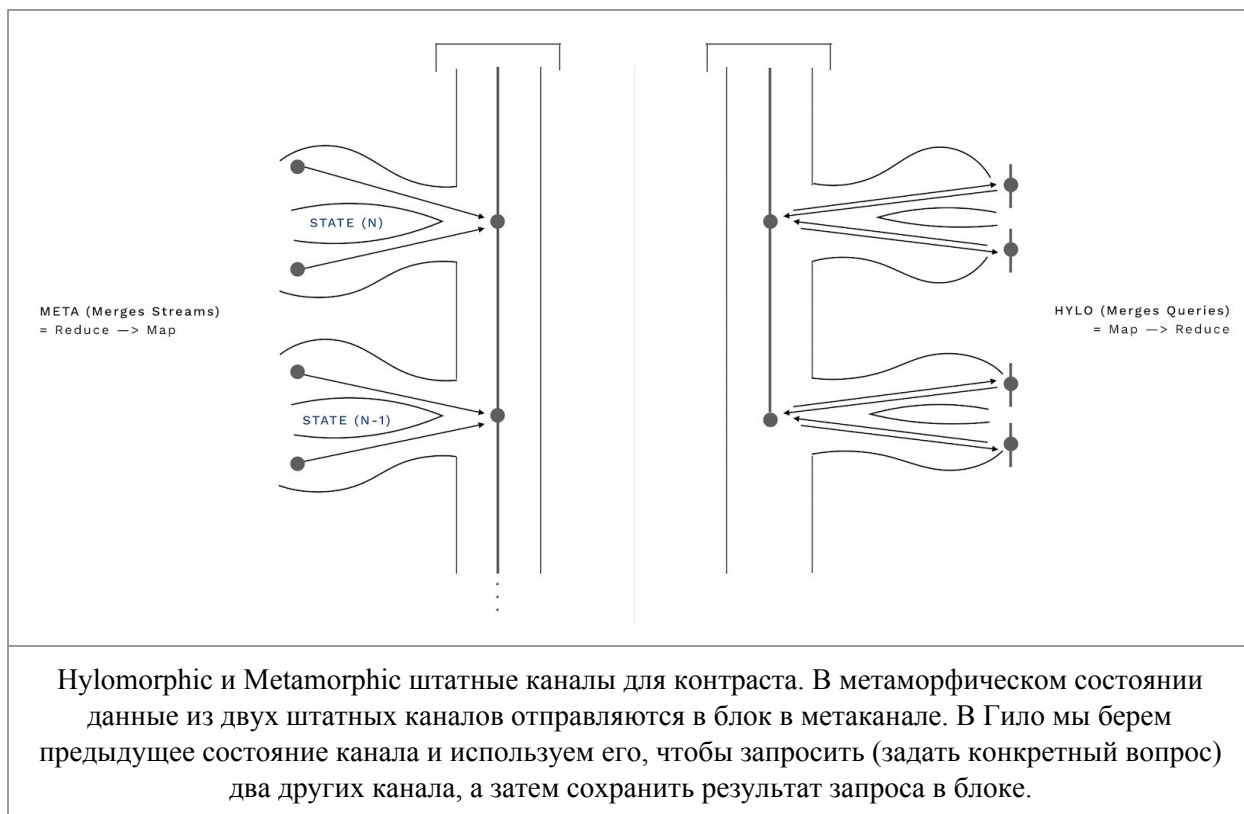


Nylochain — Поддержка приложений на основе каналов

Наш подход к поддержке приложений можно рассматривать как “децентрализованную платформу умных контрактов”. Вместо центральной сети, выполняющей всю логику и обрабатывающей все данные от приложения, Constellation координирует данные приложения со “штатными каналами”, которые можно рассматривать как телевизионную станцию, транслирующую все данные из штатной системы. Каждый штатный канал может реализовывать свою собственную логику проверки, позволяющую решить проблему оракулов путем сквозной проверки подлинности производителей данных и транзитивной проверки составных штатных систем. Сети штатных каналов обеспечивают параллельную поддержку приложений, ускоряя время принятия, которое в сети с умными контрактами ограничено традиционным синхронным консенсусом.



Причина, по которой он называется Hylochain, заключается в том, что в нашем подходе к поддержке приложений использовалась функциональная модель программирования Recursion Schemes для создания интерфейса MapReduce. В частности, схемы рекурсии Hylomorphism (Гиломорфическая) и Metamorphism (Метаморфическая) могут быть интегрированы для создания проверяемых запросов и потоковых соединений по штатным каналам путем проверки алгебраических типов данных так же, как проверяются ор-коды для умных контрактов. Конечным результатом является функциональный интерфейс MapReduce, который знаком инженерам данных и совместим с существующей технологией больших данных.



Токеномика и её связь с Nylochain

Когда штатный канал создан, он может быть интегрирован в канал \$DAG, но с использованием интерфейса ACI или Application Chain Interface. Этот интерфейс представляет собой просто объект JSON с информацией о конфигурации и открытым ключом, связанным с самим каналом. Причина, по которой мы связываем открытый ключ со штатным каналом, заключается в создании брокерского механизма для данных штатного канала. Когда штатный канал развернут, разработчики настраивают сами, как платежи из сети \$DAG распределяются между узлами и операторами.

