Aragon Nest Proposal: Connect DAOs to centralized systems

General use cases

	Use Case	App	Function/Details	Implementation
1	Be sure not to miss any vote or survey... with a Telegram notification	Voting/Survey	`event StartVote (…)` `event StartSurvey (…)`	Watch an Event then send a Telegram message with `voteId` and `_metadata` parameters.

2	Track votes/surveys outcomes in RT (and maybe build a dashboard?)	Voting	`function initialize (…)`
			`event StartVote (…)`
			`function getVote (…)`
			
			A member of a DAO can verify when the `initialize` function initializes the vote app. By reading the `voteTime` parameter and passing this parameter via webhook to a centralized service, he can create an application that verifies the result of any vote at the end of the voting time only querying the getter function `getVote`.
			
			`function getSurvey (…)` gives you the data to build a real-time dashboard to measure sentiment around a given decision
				Watch a Contract + Web hook

3	Track remaining budgets in real-time		`function _getRemainingBudget(_token)`	Watch a Contract + Web hook / Send an email / Telegram

4	Get a notification when your vested tokens finally get unlocked	Token Manager	`function _spendableBalanceOf(_token)`	Watch a Contract + Telegram
5	Track votes' execution	Voting	`event ExecuteVote (…)` Monitor the execution of a specific vote by merely checking the `ExecuteVote` event.	Watch an Event + Web hook/Telegram/Send an email
6	Check all incoming deposits of a DAO for tax compliance purposes	Finance	Check `NewTransaction(…)` events with `_incoming` parameter set to true or use "Watch a Transaction" with `deposit(…)`	Watch an Event / Watch a transaction + An authenticated web hook storing the transfers in a centralized database
7	Get notified when you receive a payment from a DAO (and maybe track it for tax compliance purposes)	Finance	`event NewTransaction(…)` You can filter using the `_reference` parameter or the recipient address	Watch an Event / + Telegram or Store to a Google Spreadsheet
8	Track your DAO's agent!	Agent	`event Execute (…)` Using an Agent a DAO can interact with any Ethereum contract using `_target` and `_data` parameters. Right now the `Execute` event allows anyone to read this data but not to decode it. With our tool if the contract called with the `_target` parameter has a public ABI, it will be possible to read,	Watch an Event + Decoder + Webhook / mail or Telegram

			decode and create any filter/trigger based on the `_data` field, which includes all the details of the action the Agent is doing on Ethereum in behalf of the DAO.	

9	Monitor new vests and current vests' state	Token Manager	`function getVesting (…)`
			
			`event NewVesting (…)`
			
			By checking the `NewVesting` event you get the `vestingId` generated by the function `assignedVested`. Using the `vestingId` parameter, it's possible to use the getter `getVesting` function to know data related to current vests
				Watch an Event to get `vestingId` + Watch a Contract trigger to get data about a vest

10	Track DAO's balances in realtime	Vault	`function balance (…)`	Watch a Contract + Webhook for monitoring or running a dashboard
11	Get notified on Telegram when your DAO give you more power/new privileges	ACL	`event SetPermissionParams(…)`	Watch an Event + Telegram

These are just a few examples... This table could have been never-ending: track payments, new budgets, new vests, immediate payments, recurring payments, track surveys' and votes' options, tokens' minting and burning, check if a survey is opened (Watch a Contract with _isSurveyOpen) or has reached the quorum (isParticipationAchieved).

Fundraising use cases

Use cases based on the fundraising modules that Aragon Black is building.

	Use Case	App	Function/Details	Implementation
12	Get a	Batched	`function getStaticPricePPM(1)`	Time trigger +

	timeseries of the prices during a batched-fundraising	bancor market maker		Watch a contract + A web hook storing the data on Mysql, on a Google Sheet, etc
13	Get info about the current batch in real-time	Batched bancor market maker	`function getBatch(getCurrentBatchId())` Returns: batch.initialized, batch.cancelled, batch.supply, batch.balance, batch.reserveRatio, batch.totalBuySpend, batch.totalBuyReturn, batch.totalSellSpend, batch.totalSellReturn);	Watch a contract (track any change) + A web hook
14	Get a timeseries of the price of a token during a token presale	Presale	`function contributionToTokens(1)`	Time trigger + Watch a contract + A web hook storing the data on Mysql, on a Google Sheet, etc
15	Get notified for every contribution via Telegram	Presale	`event Contribute (…)`	Watch an Event + Telegram
16	Get a token's total supply variation in real time via Telegram	Token Manager	`token.TotalSupply()`	Watch a contract (track any change) + Telegram

These are just a few examples... Others: get notified for every new batch, beneficiary update, track any collateral token variations, build a BOT actived by any new/claimed/canceled order.

Notify Aragon communities

Another nice improvement we'd like to build? We want to enable smart contract developers to contact/notify all the members of a DAO via Telegram, email or iOS/Android push notification with one line of solidity code.
How it works:

- Our tool offers an opt-in tool to verify connections between an Ethereum address and an email or a Telegram account.
- To verify those connections the DAO member needs to sign a transaction via Metamask. As soon as our tool sees the transactions, it considers the connection verified
- Whenever a DAO want to notify all his members via a push notification, for example, the smart contract developer can simply write: `emit Notifier("AllUsers","PushN", Message)`. It would also be possible to use already existing events, without the need to edit current smart contracts.
- Our tool filters, among the participants of that specific DAO (= any token holder of that DAO), the users who opt-in, and send the message