Advanced Messaging over DIDComm

A modular approach

Ariel Gentile

21 December 2022

Why advanced chat over DIDComm?

We believe that DIDComm has enormous potential to create an open and universal communication protocol that, combined with verifiable credentials, can provide good advantages over current dominating messaging platforms.

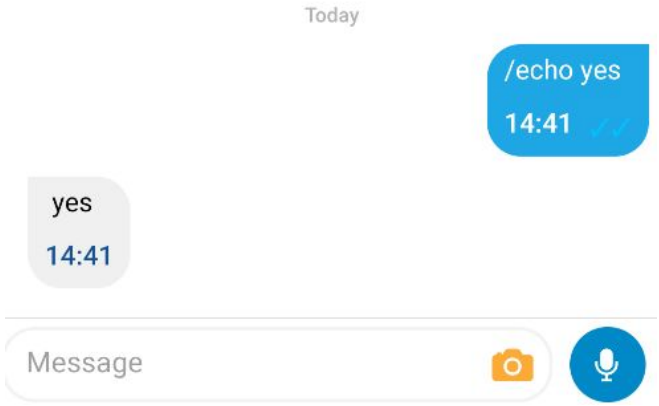
DIDComm spec is flexible and extensible enough to allow the implementation of most current Instant Messaging apps features, and even if there are still some challenges under discussion such as Push notifications and group messaging, we are sure its strong community will reach good solutions that will, at least, match current centralized messaging platforms.

A modular approach

- Instead of creating a single ‘Rich Chat Protocol’ or adapting an existing chat protocol to work over DIDComm, we achieve a similar behaviour by combining a number of different small DIDComm protocols that agents may optionally implement according to their capabilities and interests
- Some protocols well known by Aries community are used:
  - **Basic Message**: for simple text messages
  - **Question Answer**: for showing option menus and sending back responses
  - **Action Menu**: to present a contextual menu available during a chat session
- Plus other new protocols adding specific features:
  - **Media Sharing**: to send images, videos, voice notes and other files
  - **Receipts**: to provide message received/viewed/deleted status
  - **User Profile:** to exchange user information

The good-old Basic Message

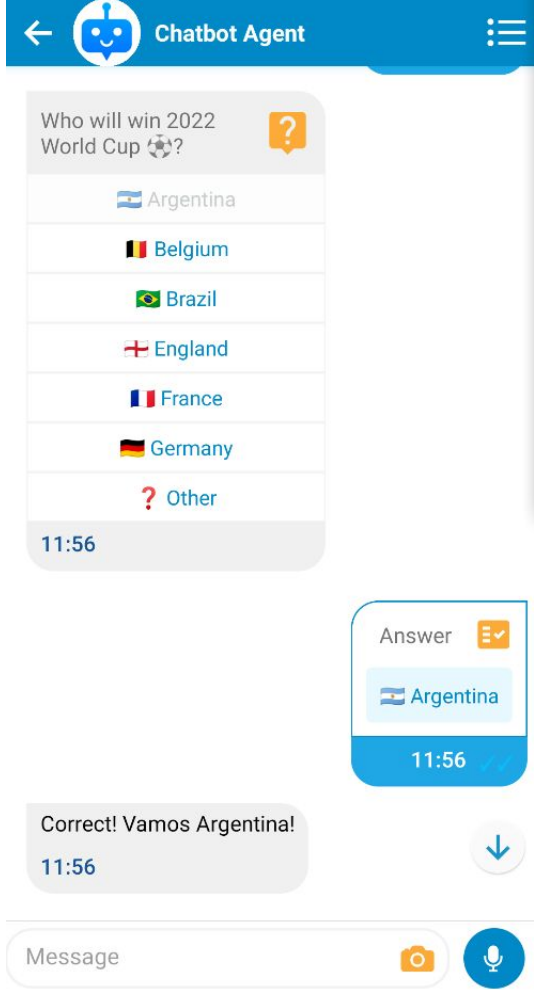
In its 1.0 version it almost specifies it’s just for testing purposes. However, it is enough for simple text messages and, as it’s implemented by all Aries agents, we prefer to simply keep it as is instead of adding new features or completely replacing it by other more flexible protocol.



Using Question Answer

Even if it's a bit basic, current version of this DIDComm protocol is enough to create simple queries to the user and, thanks to the magic of emojis, we can have a minimalist and good UX.

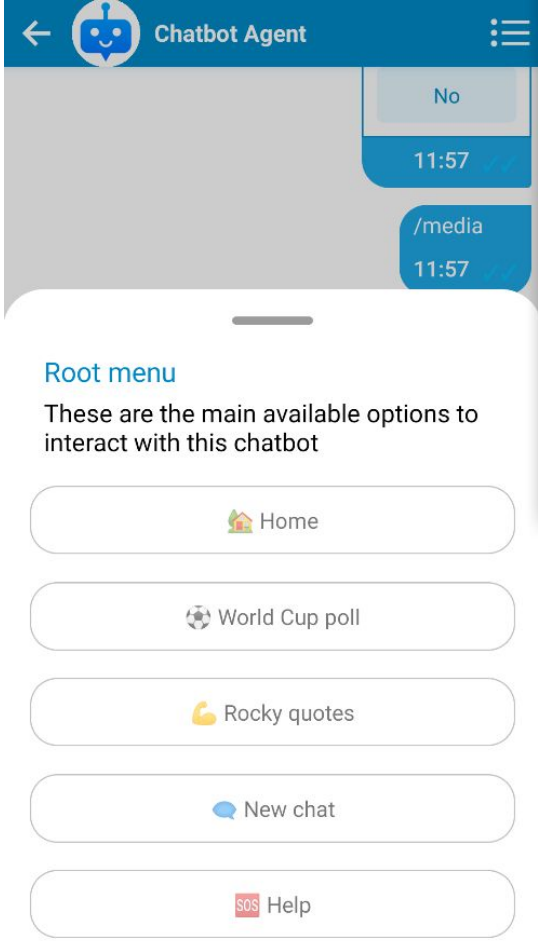
A future version of this protocol could be created to allow advanced queries using images, animations or forms elements such as combo boxes or calendar views.



Action Menu for context

Action Menu protocol is a good compliment for chat sessions in the sense that it can show a persistent menu, customized according to the user and current state.

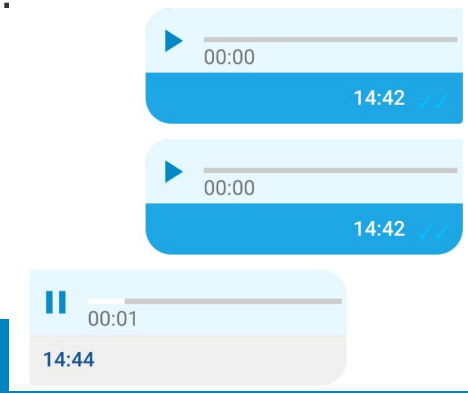
An evolution of this protocol could allow to make it more powerful by explicitly defining form elements in an standardized way.



Media Sharing

In order to have a true rich chat experience comparable to existing IM apps, we need to be able to send any kind of multimedia content: ideos, pictures, voice notes, etc.

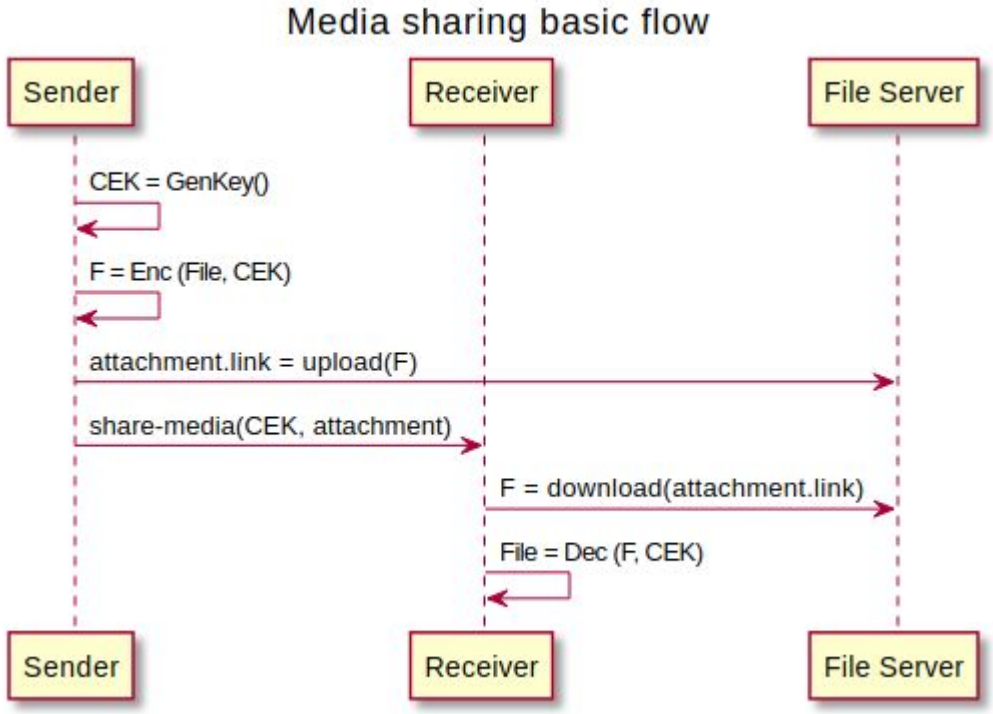
This protocol allows to achieve that by transferring files through a suitable protocol and relying on DIDComm trust and security to share file download details.



Media Sharing basic flow

Sender first encrypts and uploads media to a file server as an opaque file, and shares to the recipient through DIDComm the URI, file description and decryption material.

Based on MIME type, the recipient interprets the data and threats it accordingly

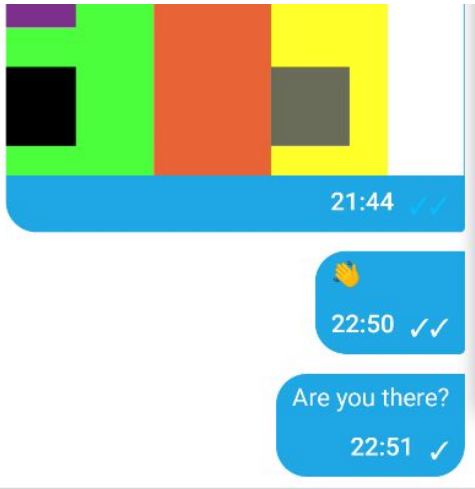


Message Receipts

DIDComm V1 by itself provides a simple way of sending a message receipt: Ack message. However, its meaning is mostly protocol-dependant and we would like to make receipts generic enough to also use them for some existing protocols (like Issue Credential or Present Proof).

Receipts protocol simply consists of a single message that might send a bunch of message receipts, each containing:

- message ID
- state: Received, Viewed, Deleted
- timestamp

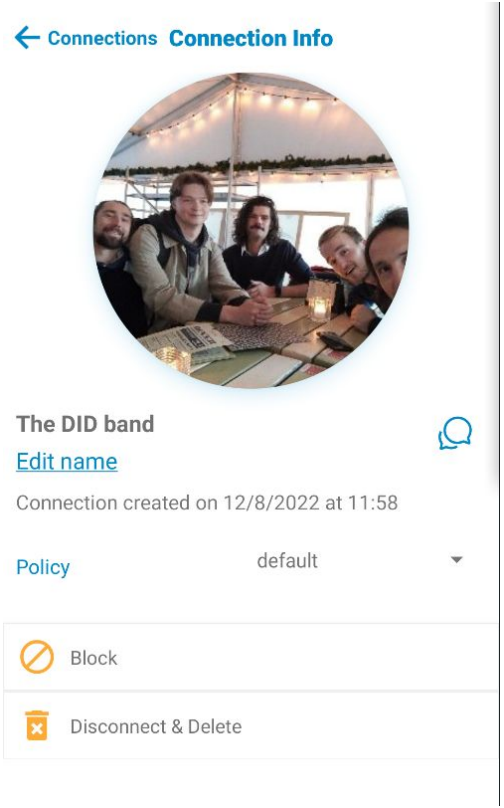


User Profile

This is another fairly simple protocol that allows an user to exchange their self-attested profile information.

Current supported fields are quite basic: picture, display name and description (bio).

Some more standard fields (e-mail address, phone number, birthdate) can be added and also a flag to indicate if user can also provide a VC to prove them



Other protocols and actions

Some other protocols and ideas using existing protocols to achieve IM features:

- **Signal**: to send a timed signal to a recipient. E.g.: typing indicator with an expiration date of *n* seconds. Online indicator with expiration date of *m* minutes
- **Call**: to set-up a video/audio call on WebRTC bootstrapped by DIDComm
- **Reactions**: similar to Receipts. Allow multiple reactions to a single message
- **Block user**: use Coordinate Mediation protocol to temporarily remove routing key
- **Contact sharing**: use Introduce protocol or an evolution of it
- **Location, Wi-Fi, BLE pairing**: media sharing (with inline attachments)
- **Polls**: Question-Answer is perfectly suitable for them

Specs and AFJ plug-ins

Draft specifications are currently written at the following branches in GitHub:

- Media Sharing: https://github.com/2060-io/aries-rfcs/tree/feature/media-sharing/features/xxxx-media-sharing
- Receipts: https://github.com/2060-io/aries-rfcs/tree/feature/receipts/features/xxxx-receipts
- User Profile: https://github.com/2060-io/aries-rfcs/tree/feature/user-profile/features/xxxx-user-profile

Initial implementation is being developed as Aries Framework Javascript plug-ins:

- Media Sharing: https://github.com/2060-io/aries-javascript-media-sharing
- Receipts: https://github.com/2060-io/aries-javascript-receipts
- User Profile: https://github.com/2060-io/aries-javascript-user-profile