Hailing a Taxi on the Web of Needs

Florian Kleedorfer[1], Fabian Suda[1], Max Stolze[1], and Christian Huemer[2]

[1] Research Studio Smart Agent Technologies, Thurngasse 8/16, 1090 Vienna, Austria
firstname.lastname@researchstudio.at,
[2] Institute for Software Technology and interactive Systems, Vienna University of Technology, Favoritenstraße 9-11, 1040 Vienna, Austria, huemer@big.tuwien.ac.at

Abstract. Matchmaking and cooperation are interactions widely available within internet-based platforms. However, to start using such functionality, one must first become a member of the platform. In an effort to push matchmaking and cooperation into the protocol stack of the Web, we demonstrate how an existing taxi service is integrated with a decentralized Web based infrastructure, the Web of Needs (WoN), using a chatbot that can be programmed declaratively to enter into certain types of agreements with users. The bot mediates between an existing Web service and users of the RDF-based WoN network.

1 Introduction

Business as well as private interactions are increasingly mediated by internet based two-sided or multi-sided platforms. A common trait of all such platforms is the fact that they offer tools for both sides of a transaction, for example, taxi drivers and passengers. Their technology requires that users in both roles be onboarded in a first step; afterwards, the intention of engaging an interaction is only published within the bounds of the platform and hence can only be satisfied within it.

Arguing that another – more open – mode of interaction is possible, we have developed a set of protocols we call the *Web of Needs(WoN)* [2]. Figure 1 gives an overview of its architecture. In order to allow for users of different domains or platforms to come to a mutually understood arrangement about how to interact with each other, we have developed an agreement protocol on top of the federated and completetly RDF [5] based chat protocol[1] used by WoN users to communicate [3]. This protocol allows users to create mutually agreed-upon RDF graphs as a result of their chat interactions. Moreover, in order to allow chatbots to create agreements when offering the services of third-party sytems in WoN, we have developed an approach for defining declaratively what sort of agreement a bot may create using SHACL [4] and a new approach for combining two RDF graphs we call *blending*.

In our applied research, we are aiming for building a WoN-based, decentralized solution for transportation. To get started, we chose the use case of calling a taxi because it is a simple case of transportation allowing for low-cost practical tests.

CEUR-WS.org/Vol-2198/paper_103.pdf

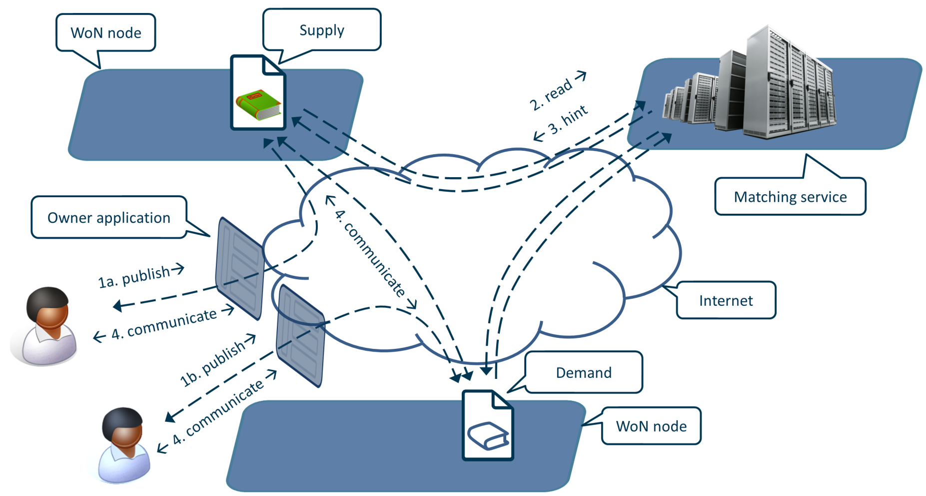


Fig. 1: Interaction diagram giving an overview of the Web of Needs. Users – referred to as *owners* – post their intentions (*needs*) using some client (*owner application*). These needs are stored on *WoN nodes* selected by the owner applications (1a, 1b). *Matching services*, subscribed to WoN nodes' updates, see the newly created needs and compare them to other ones they have seen before (2). When a matching service finds a good match for a need, it sends a *hint* message to that need (3). If the owners find the match useful, they can start to interact via a newly created communication channel (4).

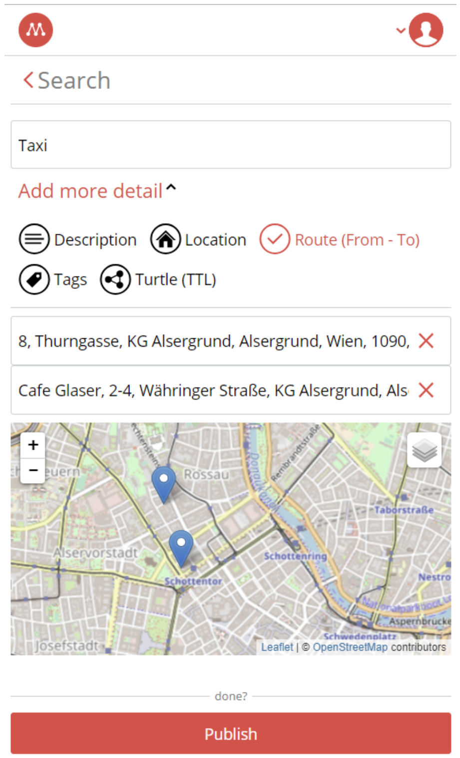
2 Demo: Hailing a Taxi

The demo at hand[3] shows the implementation of the agreement protocol in combination with automated negotiation. The interaction takes place between us (a human user) and a bot that is connected to the REST interface of a taxi service. First, we use the owner webapp to post the need for a taxi, stating pick-up and destination locations (Figure 2a). Subsequently, we are contacted by the taxi bot (Figure 2b). When announcing its taxi service, the bot sets the `won:NoHintForCounterpart` flag in its posting, which causes matchers not to send a hint to our posting[4]. Instead, only the bot is notified of an opportunity to interact; it can choose whether to make an offer or not. In our case, the bot connects with us, we accept the connection, and after the taxi bot has evaluated the data in the conversation (locations, times, etc.) it makes a proposal to execute the ride (Figure 2c). Finally, we accept that proposal (Figure 2d), which therefore becomes an agreement.[5]

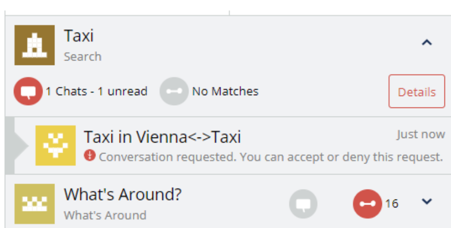
[3] See `https://matchat.org/`

[4] Note that our posting has no matches in Figure 2b

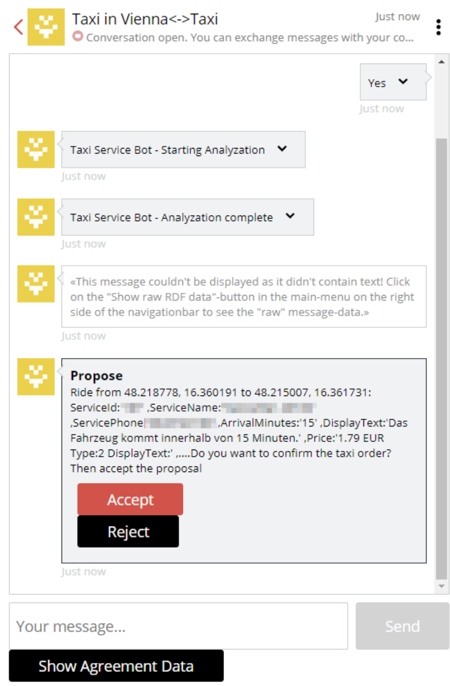
[5] Readers should note that the taxi bot that was used in this demo may not be running at all times, and that the outcome of conversations with the bot may vary over time.



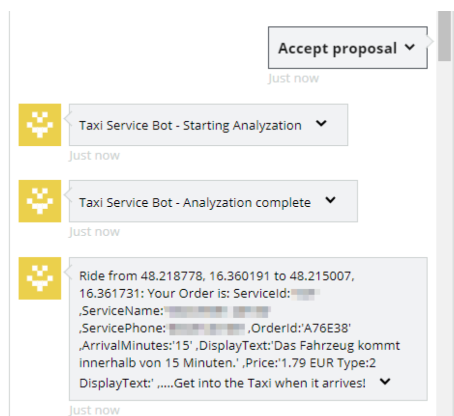
(a) Posting a search for a taxi, specifying start and destination of the ride.



(b) Our posting, searching for a taxi, has been contacted by the taxi bot. Next to it, a 'What's Around' posting that matches anything nearby, demonstrating that it is a general purpose application not specifically made for the taxi use case.



(c) The taxi bot makes a proposal after analyzing the conversation data.



(d) We accept the proposal. The taxi bot is sending a taxi.

Fig. 2: Screenshots from the WoN Owner application during the negotiation for a cab ride.

3 Future work

The purpose of this prototype is an end-to-end demonstration of a simple transportation use case, so far only showing the initiation of the interaction, for example, calling a cab. The later stages - communication about the actual arrival of the car, the arrival of the passenger, completion of the ride and success of the payment, are planned for future work. Our approach for representing the complete transaction in the conversation consists of representing the transaction as a state machine that can be evaluated independently on either side of the transaction. Our idea is that the initial proposal contains, among other data, the definition of the state machine that the transaction will then be represented with. The definition of that state machine contains states and defines how messages sent by the participants affect those states.

In addition to this extension of the protocol, we aim for representing and testing more complex cases of goods transports, as well as, eventually, cases from other domains.

4 Acknowledgements

This work demonstrates outcomes of the projects *OLN - Open Logistics Networks*, funded by the Austrian Federal Ministry for Digital and Economic Affairs in the pogram *COIN - Cooperation and Innovation* and *CoShA - Cooperation and Sharing Applications*, funded by the Austrian Ministry for Transport, Innovation and Technology in the program *Mobilität der Zukunft*.

References

1. F. Kleedorfer, C. M. Busch, C. Huemer, and C. Pichler. A linked data based messaging architecture for the web of needs. *Enterprise Modelling and Information Systems Architectures*, 11(3):1 – 18, 2016.
2. F. Kleedorfer, C. M. Busch, C. Pichler, and C. Huemer. The case for the web of needs. In *Business Informatics (CBI), 2014 IEEE 16th Conference on*, volume 1, pages 94–101. IEEE, 2014.
3. F. Kleedorfer, H. Friedrich, and C. Huemer. Agreements in a de-centralized linked data based messaging system. In *Proceedings of the Workshop on Decentralizing the Semantic Web (DeSemWeb)*, Vienna, Austria, October 2017.
4. H. Knublauch and D. Kontokostas. Shapes constraint language (shacl), 6 2017. [Last accessed on 2018/06/06].
5. F. Manola, E. Miller, and B. McBride. Rdf 1.1 primer, 2014. [Last accessed on 2017/07/26].

However, it is possible to test the agreement functionality with two distinct user accounts.