Meeter
Systemy agentowe w zastosowaniach

Anna Buchman

19 czerwca 2022

Spis treści

1 Wstęp

Meeter to systemem agentowym ułatwiającym grupom znajomych znalezienie i zarezerwowanie odpowiedniego miejsca do wspólnego spotkania. Celem systemu jest znalezienie miejsca dla grupy odpowiadającego jej preferencjom aktywności, lokalizacji oraz cenowym. Rodzaje miejsc będą inspirowane dostępnymi w Warszawie.

2 Opis projektu

System został zaprojektowany w języku Python przy użyciu platformy agentowej SPADE [2]. Projekt został podzielony na klasy odpowiadające agentom i pliki z enum-ami wykorzystywanymi w projekcie, dodatkowo zawiera 4 pliki służącą do odpalania projektu: main i 3 pliki scenariuszy.

2.1 Uruchomienie projektu

Aby uruchomić projekt należy najpierw przejść w terminalu do lokalizacji projektu i pobrać potrzebne biblioteki wykonując komendę:
pip install -r requirements.txt
Następnie trzeba wybrać uruchomienie głównego pliku main lub któregoś ze scenariuszy. W pliku main.py generowane są losowo atrybuty agentów. Aby odpalić główny plik main należy wykonać w terminalu komendę:
python main.py
Można również ustalić jaka będzie liczba regionów, miejsc, liderów, użytkowników w systemie wykonując komendę z parametrami przykładowo:
python main.py -r 3 -p 7 -u 15 -l 10
Liczby występujące po:

- -r to liczba regionów
- -p to liczba miejsc
- -u to liczba użytkowników
- -l to liczba liderów

Liczba agentów grup będzie równa liczbie liderów. Jest możliwość poznania znaczenia liter wpisując komendę:
python main.py -h
Scenariusze zostały tak przygotowane aby reprezentować możliwości systemu i nie ma możliwości parametryzowania ich. Mają podobną budowę do main.py jednak wykorzystane są inne funkcje do określania atrybutów agentów, dlatego nie można by przygotować skryptu shell-owego, który odpalałby plik main.py z konkretnymi parametrami. Aby uruchomić poszczególne scenariusze należy wykonać następujące komendy:

- *python scenario_everybody_one_day.py*
- *python scenario_no_friends.py*
- *python scenario_no_place_type_qfor_me.py*

2.2 Agenci

W systemie głównymi agentami są użytkownicy, liderzy agenci grup i agenci miejsc, do których grupy mogą się udać. Reszta agentów powinna ułatwiać i usprawniać komunikację między nimi. W systemie zostali istnieją następujący agenci:

- Agent obszaru (region_agent) - jest odpowiedzialny za posiadanie bieżących informacji na temat miejsc znajdujących się w jego obszarze takich, jak nazwa miejsca, typ miejsca, godziny otwarcia, liczba miejsc dla gości plus informacje techniczne jak jid agenta potrzebne do komunikacji. Komunikuje się z agentem grupy, wybiera miejsca ze swojego obszaru odpowiadające na podstawowe preferencje zadane przez grupę: czy istnieje miejsce o zadanym typie, czy grupa zmieści się w miejscu i czy miejsce jest otwarte w godzinach, w których grupa chce się spotkać. Wysyła listę miejsc zgodnych z preferencjami do agenta grupy.

- Agent grupy - reprezentuje grupę klientów posiadającą atrybuty takie jak:

    1. Liczność grupy.
    2. Obszary w których chce się spotkać grupa.
    3. Preferowany rodzaj aktywności (typ miejsca) i opcjonalne n rodzaje aktywności (np preferowane są kręgle, opcjonalnie grupa może iść do kina lub na basen).
    4. Data spotkania.
    5. Przedział czasowy, w których grupa jest dostępna.
    6. Ile czasu chcą poświęcić na spotkanie.
    7. jid lidera potrzebne do komunikacji

    . Zadaniem agenta grupy jest znalezienie odpowiedniego miejsca dla grupy klientów, aby zrealizować to zadanie agent grupy negocjuje z agentami regionów w poszukiwaniu najlepiej pasującego miejsca. Dostaje od agentów regionów propozycje w postaci listy miejsc posortowaną od najbardziej do najmniej pasujących do preferencjom. Następnie zaczyna negocjacje z agentami miejsc z listy. Najpierw pyta agenta miejsca o dostępność na daną datę, po otrzymaniu które miejsca są dostępne wybiera miejsce i dokonuje rezerwacji.

- Agent miejsca - reprezentuje miejsce. Agent komunikuje się najpierw z agentem regionu. A następnie odpowiada na zapytania od agentów grup w celu rezerwacji. Atrybuty, które posiada miejsce:

    1. Nazwa miejsca
    2. Lokalizacja
    3. Liczba osób możliwa do przyjęcia
    4. Typ aktywności
    5. Cennik aktywności od osoby/grupy
    6. Lista cen promocyjnych przypisanych konkretnemu dniu
    7. Godziny otwarcia
    8. jid regionu w którym leży miejsce
    9. Kalendarz rezerwacji
    10. Kalendarz cen specjalnych

    Agent miejsca wysyła część danych agentowi obszaru w celu dodania go do listy miejsc w obszarze. Po zapytaniu od agenta grupy odpowiada na pytanie czy rezerwacja jest możliwa czy nie, jeśli dostanie zapytanie od agenta grupy o prośbę o rezerwację dodaje grupę do kalendarza rezerwacji. Ponadto obsługuje informację od agenta zegara, że data się zmieniła. Na datę "dzisiejszą"ustala promocyjną cenę. Jeśli agent grupy będzie chciał zarezerwować miejsce na dzisiaj to dostanie w odpowiedzi promocyjną cenę.

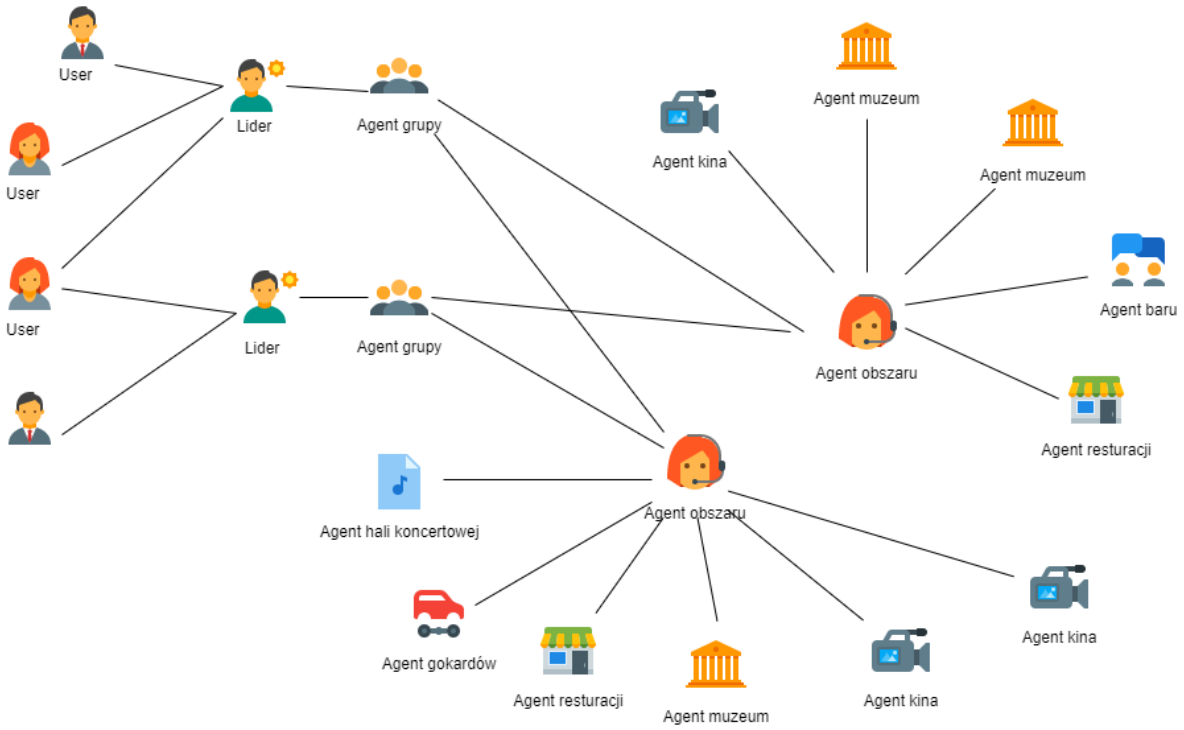
- Lider - jest osobą, która wychodzi z propozycją spotkania się do użytkowników - swoich znajomych. Zbiera informację od znajomych, o tym kto chce się spotkać zadanego dnia, gdzie, o której godzinie, w jakim obszarze i na jaki typ aktywności. Po zebraniu informacji wybiera 3 najbardziej preferowane obszary, 3 najbardziej preferowane typy miejsc, najbardziej preferowany przedział godzin i wysyła je do agenta grupy. Na koniec pracy agenta grupy otrzymuje informację czy udało się dokonać rezerwacji czy nie.
- Użytkownik - jest to symulacja znajomego agenta lidera. Agent użytkownik jest pytany przez lidera czy ma chęć się z nim spotkać. Jeśli agent użytkownik ma chęć to odsyła liderowi:
  1. Preferowany obszar
  2. Preferowany typ aktywności
  3. Przedział godzin w których jest wolny
- Agent zegar - co 3 sekundy wysyła wszystkim agentom miejsc że data się zmieniła.

2.3 Komunikacja między agentami

Komunikacja między agentami w SPADE opiera się na protokole XMPP. Podczas implementacji zostało przetestowane 5 serwerów w poszukiwaniu odpowiadającym wymaganiom systemu, jakimi są: szybkość przesyłania wiadomości i liczba agentów między którymi można przesyłać wiadomości. Serwery były przeglądane na podstawie tej strony [1]. Używane serwery i ich wady zostały wypisane poniżej:

1. `https://www.jabbim.com/` ma ograniczenie do 10 agentów łączących się do serwera.
2. `https://www.shad0w.io/` ma ograniczeni do 10 agentów łączących się do serwera, ponadto strona na głównej stronie ma tytuł darknet.im więc wykorzystywanie jej może budzić wątpliwości.
3. `https://chatserver.space/` jest zlokalizowany w Stanach zjednoczonych, więc przesyłanie wiadomości między agentami z Polski jest bardzo wolne (jedna wiadomość wysyła się ok 1 s).
4. `https://jabber.hot-chilli.net/` po wystąpieniu kilkukrotnych błędów w formacie wiadomości lub kodzie systemu serwer przestaje odpowiadać, po około dobie można do nie go się ponownie połączyć, dlatego dewelopowanie przy użyciu tego serwera jest uciążliwe.
5. `https://jabbers.one/en/index.php` nie zostały znalezione żadne wady w czasie używania.

Poglądową komunikację między agentami można przeanalizować na rysunku 1. Kreski między agentami obrazuje komunikację pomiędzy agentami. Na rysunku nie została zaznaczona komunikacja między agentami grup a agentami miejsc w celu rezerwacji miejsca dla grupy.

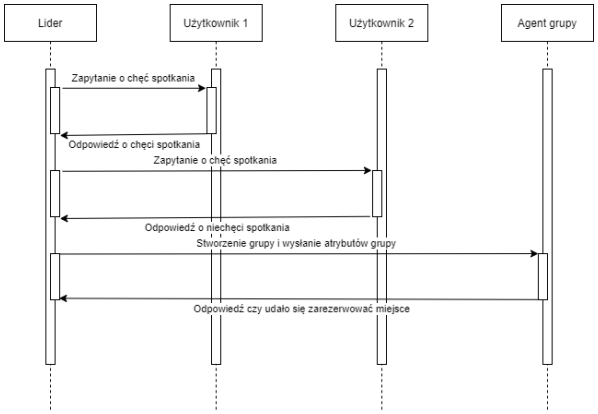


Rysunek 1: Poglądowy graf obrazujący komunikację między agentami

Przykładowa komunikacja miejsca z agentem obszaru została przedstawiona na rysunku 2 jako graf sekwencji. Miejsce może wykonać dwa rodzaje komunikacji z agentem obszaru: dodanie do obszaru i usunięcie z obszaru.

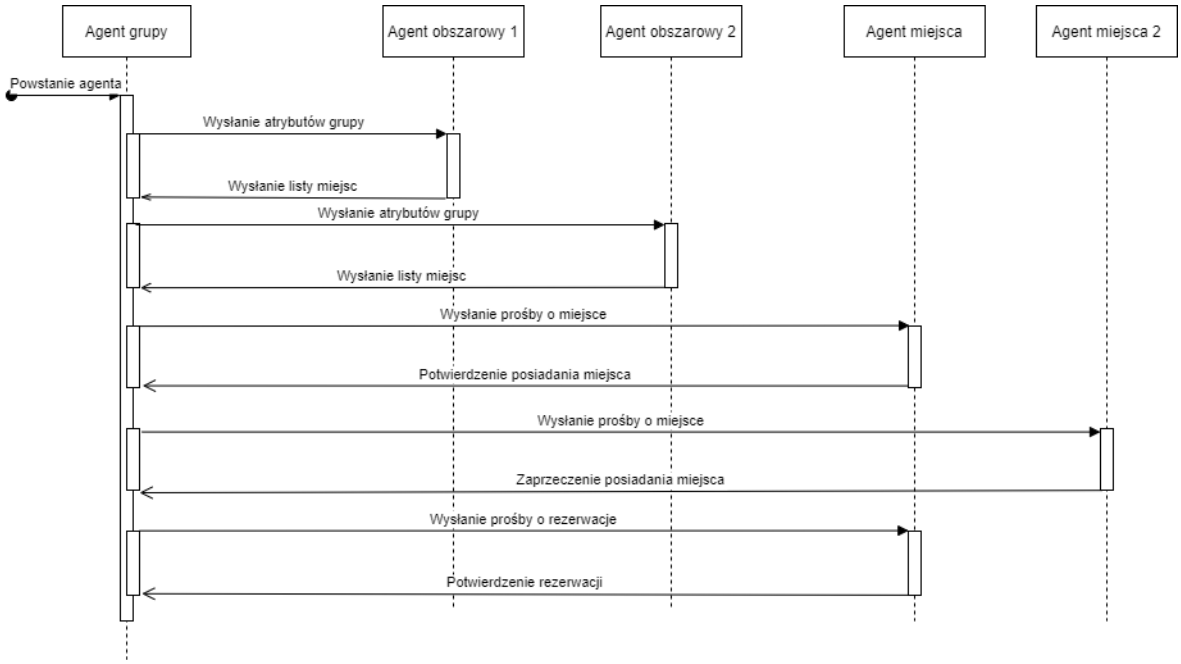
Rysunek 2: Poglądowy graf sekwencji agenta miejsca z agentem obszarowym

Przykładowa komunikacja w systemie agentów lidera, użytkownika i grupy została przedstawiona na rysunku 3



Rysunek 3: Poglądowy graf sekwencji lidera, użytkownika i agenta grupy

Przykładowa komunikacja w systemie agentów (od agenta grupy) ze sobą podczas próby rezerwacji poprzez miejsca dla grupy została przedstawiona na rysunku 4.



Rysunek 4: Poglądowy graf sekwencji rezerwacji miejsca

Słowniki atrybutów wysyłane między agentami są konwertowane do jsonów, a następnie postronie odbiorcy rozpakowywane do słowników.
Wyświetlanie komunikacji między agentami została skrócona do minimum. Przykładowo zamiast pełnej komunikacji między liderem a użytkownikami jest wyświetlana liczba użytkowników chętnych do spotkania i nie chętnych do spotkania.

2.4 Słowniki atrybutów agentów

Projekt został przygotowany w ten sposób, aby była też możliwość stworzenia swojego pliku głównego z konkretnymi atrybutami nadanymi poszczególnym agentom. Agenci, których do stworzenia potrzebują podania słownika posiadają funkcje generującą te atrybuty losowo i funkcję która tworzy słownik atrybutów po podaniu każdego z atrybutów z osobna. Z powodu na liczbę atrybutów potrzebnych do stworzenia pełnego dużego systemu podjęto decyzję o przeprowadzaniu testów na losowo wygenerowanych atrybutach lub tak jak jest w plikach scenariuszy, na częściowo losowo wygenerowanych słownikach atrybutów. Przykładowo w pliku leader_agent.py występują 3 statyczne funkcje służące do generowania słowników: create_leader_dict, create_random_leader_dict i create_random_leader_dict_with_specific_date.

2.5 Nazwy agentów

Została wykorzystana biblioteka randomname, aby wygenerować losowe nazwy miejsc do których może się udać grupa oraz nazwy liderów. Nazwy miejscy były losowane z grup: 'gaming', 'music_instruments', 'astronomy'. Przykładowe nazwy miejsc to: reduced-envelope, other-level, obvious-tactics, strong-sport, juvenile-piano. Nazwy liderów grup były losowane z grupy: 'fish'. Przykładowe nazwy liderów to: adiabatic-carp, crabby-gong, fresh-tuna, annoying-anchovy.
Nazwy regionów pochodzą od nazw dzielnic Warszawy.

3 Scenariusze

Zostały wymyślne następujące scenariusze sprawdzające wydajność systemu:

1. Próba zarezerwowania miejsc przez wiele grup agentów w na tę samą datę. Agent miejsca rezerwuje miejsca dla grupy o danej liczności aż do momentu gdy pomieści grupę tego dnia. W momencie gdy grupa nie zmieści się w danym miejscu agent miejsca mówi, że rezerwacja miejsc jest niemożliwa. Można prześledzić poprawność scenariusza w poniższych logach z opisami:
   MIEJSCE:
   "place_name": "isobaric-squad", ńumber_of_guests": 18
   Miejsce isobaric-squad rezerwuje 6 miejsc dla grupy:
   "date": "2022-05-7", ńumber_of_guests": 6, łeader_name": affable-marlin"
   Leader affable-marlin received with content: Reservation made isobaric-squad
   Miejsce isobaric-sqad rezerwuje kolejne 6 miejsc dla grupy:
   "date": "2022-05-7", ńumber_of_guests": 6, łeader_name": ćhestnut-blowfish"
   Leader chestnut-blowfish received with content: Reservation made isobaric-squad
   Miejsce odmawia rezerwacji grupie 8 osobowej, ponieważ ma już tylko 6 miejsc z 18 wolnych tego dnia:
   "date": "2022-05-7", ńumber_of_guests": 8, łeader_name": "poky-herring"
   I have no place on that date
   Reservation is not possible in isobaric-squad

2. Lider nie posiada znajomych który chcą się z nim spotkać. Scenariusz został zaprojektowany w pliku scenario_no_friends.py. W takiej sytuacji lider wysyła informacje do swojego agenta grupy aby się zabił i sam się zabiją. Można to zaobserwować w następujących logach:
   Leader starting name: hushed-marlin, jid: meeter@jabbers.one/23
   yes users: 0, no users: 1
   I have no good friends
   Goodbye word, last words from group agent: meeter@jabbers.one/13 Wszyscy znajomi lidera odpowiedzieli nie na pytanie więc lider wysyła wiadomość do grupy aby się zabiła i sam się zabija. Jest tu możliwość rozwinięcia aplikacji, lider może próbować pytać znajomych jeszcze raz zmieniając datę.

3. Brak istnienia miejsca o typie odpowiadającym preferencją grupy. Agent grupy wtedy nie może znaleźć miejsca dla grupy, informuje lidera i lider i agent grupy się zabijają. Można to prześledzić w logach z opisami po wykonaniu skryptu scenario_no_place_type_qfor_me.py: Wszystkie miejsca mają typ:
   'EscapeRoom'
   Poszukiwane typy: [Ćinema", Żestaurant", "Bar"], agent grupy nie znalazł żadnego miejsca z listy, agent grupy i lider zabijają się:
   Leader received with content: No place to go, kill yourself
   I'm gonna kill myself, I hava no place to go with my friends
   Goodbye word, last words from leader agent name: savory-flounder, jid: meeter@jabbers.one/36
   Goodbye word, last words from group agent: meeter@jabbers.one/26
   Poszukiwane typy: [Żestaurant", ĘscapeRoom", "Pool"], agent grupy dostał wszystkie miejsca o typie EscapeRoom, wybiera miejsce, dokonuje rezerwacji przesyła ją liderowi i umiera:
   Reservation is possible in ordered-cello
   Reservation is possible in lazy-clan
   Reservation is possible in brown-viola
   GroupAgent: I can happily die, I have place to go
   Leader marked-carp received with content: Reservation made brown-viola
   Leader: I can happily die, my friends has place to go
   Goodbye word, last words from leader agent name: marked-carp, jid: meeter@jabbers.one/39
   Goodbye word, last words from group agent: meeter@jabbers.one/29

4 Dalszy rozwój projektu

Pomysły na dalsze rozszerzenia, lecz możliwe, że nie uda się ich zrealizować:

- Dodanie wybierania miejsca na podstawie konkretnych preferencji lidera. Wysyłanie listy miejsc do lidera przez agenta grupy i wybieranie z listy przez grupę znajomych a nie przez agenta grupy
- Dodanie możliwości promocji, jeśli jakieś miejsce zapłaci odpowiednią sumę będzie wybierane przez agentów obszarowych do proponowania agentom grup częściej.
- Dodanie wielu aktywności które można robić w jednym miejscu i oddzielnych kalendarzy rezerwacji.
- Rozszerzenie kalendarza rezerwacji o możliwość rezerwacji na konkretną godzinę. Do tego byłaby potrzeba połączenia się z bazą danych aby pomieścić liczbę rezerwacji w systemie.
- Dodanie kalendarza spotkań u użytkowników, odmawianie spotkań liderowi w momencie gdy ma spotkanie w tym samym czasie z inną grupą.

5 Podsumowanie

Został przygotowany i przetestowany projekt Meeter ułatwiający grupom znajomych rezerwację w miejscu odpowiadającym kryteriom grupy. Projekt został zaproponowany i zrealizowany w pojedynkę w skróconym czasie. Program składa się z 6 rodzi agentów komunikujących się ze sobą poprzez wysyłanie wiadomości. Agent miejsca może zaproponować agentowi grupy promocyjną cenę. System został zaprojektowany tak aby był łatwo rozszerzalny. Dodatkowo rozpoznany który z serwerów XMPP jest najlepszy do dewelopowania systemów agentowych przy użyciu biblioteki SPADE.

Bibliografia

[1] `https://list.jabber.at/` data wyświetlenia strony 01.06.2022

[2] `https://spade-mas.readthedocs.io/en/latest/readme.html` data wyświetlenia strony 1.06.2022