Modeling Human Daily Preferences through a Context-Aware Web-Mapping System Using Semantic Technologies

Iosif Viktoratos[1], Athanasios Tsadiras[1], Nick Bassiliades[2],

[1]Department of Economics, [2]Department of Informatics,
Aristotle University of Thessaloniki
GR-54124 Thessaloniki, Greece
{viktorat, tsadiras, nbassili}@auth.gr

Abstract. In this paper, a novel geosocial networking service called "G-SPLIS" (Geosocial Semantic Personalized Location Information System) is presented. The paper provides a methodology to design, implement and share in a formal way human daily preferences regarding points of interest (POIs) and POI owners' group targeted offering policies, via user-defined preferences and policy rules. By adding rules at run time users have more flexibility and they do not rely on the pre-determined application's methods to get personalized information. Furthermore, G-SPLIS provides a large knowledge base for other systems in the web, because rules are easily sharable. To achieve the above, the presented system is compatible with Semantic Web standards such as the schema.org ontology and uses RuleML for rules that define regular users' preferences and POI owner's group-targeted offers. Finally, it combines at run-time the above to match user context with related information and visualizes personalized information.

Keywords: Rules, Semantic Web, Location Based Services, Context, Group-Targeted Offers.

1. Introduction

During the past few years, Location-Based Social Networking Services (LBSNSs) have gained a huge commercial recognition [1-3]. Applications such as Facebook Places[1], Foursquare[2] and CitySense[3] are used on a daily basis by millions of people. They are commercial applications that display users' geographical positions via representations on a map, enhanced with a social layer [2, 3]. By utilizing people collaboration and community-based knowledge they enable a higher perception of a local area than traditional LBS [4,5].Successful LBSNSs should offer relevant and proactive information delivery to their users [6, 7]. To achieve this, researchers and industries focus on collecting and utilizing contextual knowledge, every piece of information which is useful for inferring a user's situation and determining his/her requirements [6-8]. For example, data concerning profile (e.g. job), environment (e.g. location), social relationships (e.g. friends) and other are gathered for this purpose.

One of the most common methods to enhance contextual knowledge collection and perception process is the use of semantic web standards such as RDF/S and OWL (usually referred as ontologies) [8]. Ontologies offer the ability to represent the structure of physical entities and the associations between them (e.g. representing concepts such as user profiles, places etc.). Apart from this, they enable knowledge sharing and semantic interoperability, through reasoning. They allow seamless communication between heterogeneous systems, by providing a formal and general knowledge representation and reasoning standard [9-12]. Finally, they provide flexibility, since they can be reused and extended easily, saving a lot of time and effort for developers.

Context perception can be efficiently improved by combining ontologies with rules. Rule-based systems are more autonomous than systems that do not use rules because they are capable of conceiving context changes and respond accordingly without user intervention. Furthermore, they are more proactive, offering services beforehand [13-14].Such examples are OWL 2 RL [15] and SRWL [16].

In this work, an innovative location-based social networking service called "G-SPLIS[4]" [17, 18] will be presented in order to demonstrate how semantic web technologies can enhance LBSNSs and offer qualitative contextualized information to their users. G-SPLIS is an extension of a system called "SPLIS" which was presented in EC-Web 2013 [19]. "SPLIS" functionalities can be summarized as follows:

1. Collects POI data from Google Places API.
2. Is compatible with a widely accepted ontology such as schema.org to represent persons' profiles, POIs and their relations.
3. Offers POI owners the capability to enrich the schema at run time by adding their own properties.
4. Provides POI owners with a form-based web interface in order to deploy their offers according to their policy, regarding the appropriate target group and the context of each user (user's profile, place, time, weather, etc.).
5. Transforms these offers into machine understandable rules. RuleML format is used for knowledge sharing. Jess translation is employed to make them machine executable.
6. Stores metadata and rules in the form of RDF triples (using Sesame) for interoperability and reusability. Data, by being in RDF form, are system independent and they can be used by third party services using a SPARQL endpoint.
7. Displays personalized information on Google Maps[5] to regular users/potential customers. A user-friendly interface layer provides them with the opportunity to find straight away a place or an offer matching his/her profile.

G-SPLIS is an extension of a previous version combined with social dynamics. Apart from the above:

[1]https://www.facebook.com/about/location

[2]https://foursquare.com/

[3]https://www.sensenetworks.com/products/macrosense-technology-platform/citysense/

[4] Can be accessed at http://tinyurl.com/GeoSPLIS

[5]http://maps.google.com

1. The system collects data concerning persons' profiles (apart from the standard registration form) via two well-known social networks such as Facebook[6] and Google+[7].
2. Via a user friendly web editor, the system provides to regular users the capability to add their own contextualized rule-based preferences concerning POIs (for example the rule "If time is between 13:00-16:00 and weather is Sunny then I would like to visit a Coffee shop"). The system combines POI data with user context and supports preferences that involve every existing property of a POI, weather condition (e.g. if weather is sunny find me an ice-cream shop), time-day-month condition (e.g. I would like to visit a cinema, if it is Wednesday between 20:00-23:00) or user's location (e.g. I want a grocery store which is closer than 500 meters). Data[8] and rules are stored in Sesame in order to be interoperable and shareable with other systems, as described above.
3. Allows the user to use his/her preferences stored in a RuleML file uploaded to his/her Google+ account.
4. Executes and evaluates data and rules (user-defined rules and POI owners' offering policies) on the fly to provide high quality contextualized information to the user presented on Google Maps (relevant places and offers regarding his/her context).
5. Allows users to create social ties and communicate among each other, as in other LBSNSs. Additionally, the system provides users with the capability to share and combine their preferences with those of their nearby friends and matches them with POIs and personalized offers.

The aim of our work is to demonstrate how semantic web technologies can enhance the LBSNSs and offer qualitative contextualized information to their users. In everyday life people have preferences such as "if it is Sunday noon I would like some restaurants that serve Chinese cuisine" etc., which depend on their existing context. Such preferences are not completely random and present strong daily patterns [20]. For example, people usually visit a bar at night, a beach when the weather is hot, or a museum if it is morning [20]. Thus, it would be useful to support users with the capability to model such preferences. Unfortunately, existing LBSNs do not provide regular users with the capability to create and share their own rule-based preferences and flexible customize their experience. Most systems rely on the personalization system that their developers have implemented, except from a few that users can set preferences, such as their favorite restaurant type. However, these preferences are neither flexibly defined not contextualized compared to our rule based approach.This is the first time in a LBSNS where user-defined, knowledge-based (based on logical rules) policies are used in such way. In addition, as these preferences are based on semantic web standards, they can be enriched and reused by other systems on the web, providing a knowledge pool based on social intelligence and people collaboration. G-SPLIS's scientific and technical contribution will be presented in detail in the next section, after a short reference to related approaches that inspired our work. Section 3 describes the architectural components of the system and section 4 elaborates the system's processes. Next, some use case scenarios are illustrated. Finally, section 6 demonstrates the evaluation results and section 7 presents the conclusions of our work and discusses future directions.

2. Related work and Motivation

There are a lot of approaches that make use of semantic web technologies to present personalized information. Some of them that inspired our work are the following:

2.1. Semantic-based personalization in LBS

An approach similar to the one that we propose is that of Ciaramella et al. [21], which exploits predefined rules in SWRL language in order to determine a user's context and offer a couple of available services proactively to him/her. A rule-based approach was also adopted in Sem-Fit [22]. The service utilizes fuzzy rules to recommend hotels to the user. After that, the user can give feedback by evaluating the results, and the service updates the rules in order to provide more accurate recommendations. Keßler [23] developed a mobile application that recommends personalized surf spots at California's central coast by adopting a RESTful approach for editing SWRL rules. Another interesting service was developed by Niforatos et al. [24]. The service informs users about nearby offers while they are on the go. In addition, Armenatzoglou et al. [25] created an innovative conference assistant which uses semantic web technologies to provide contextualized notifications to conference attendees. Similarly, Yu Her et al. [26] implemented a flexible application which provides efficient solutions regarding the situation of a business meeting. Additionally, Fuchs et al. [27] built a context-aware driver assistant service using rules and ontologies for higher performance.

2.2. Semantic-based LBS that utilize social media data

Many existing works make use of social media data in order to give more efficient solutions regarding their users' needs and preferences. To start with, Serrano et al. [28] combined RDF data taken from sources such as foaf with predefined rules in SWRL language to recommend POIs related to users' social profiles. Moreover, Croitoru et al. [29] presented a service, which collects, models and integrates geosocial knowledge from various heterogeneous social media sources.Another example is PhotoMap [30], which uses SWRL rules to attach physical and social context to photo shots (for example where the photo was taken and who was there). Additionally, it supports an interface to manage those photos. Moreover, Li et al. [31] developed a semantic-based mobile ad hoc LBSN which connects users with similar interests. Furthermore, Woensel et al. [32] proposed a person matching application based on data retrieved from foaf profile and identification techniques such as RFID, Bluetooth etc.

2.3. Related approaches and comparison to G-SPLIS

Regarding the knowledge-based LBSs that are referred in section 2.1 above, apart from the advantages they have, the following drawbacks/challenges can be noticed:

[6]https://www.facebook.com/

[7] https://plus.google.com

[8]G-SPLIS Sesame server can be accessed at *http://platon.econ.auth.gr:8080/openrdf-sesame*and data can be accessed at *http://platon.econ.auth.gr:8080/openrdf-sesame/repositories/3*

1. They make use of a static and predefined set of rules, which has been created manually by the system's developers in advance. Regarding a LBSN service, these rules are not always enough to conceive a user's contextual situation and help the service to provide the adequate solutions to him/her.
2. Many system features change from time to time during its usage (for example persons' profiles, place data updates, modifications in connected APIs etc.), making the system's rule base insufficient and obsolete. The need for adjustments in the knowledge base is imperative in such situations, requiring time and cost.
3. The design, the development, the editing and the maintenance of a knowledge base (especially rules) is a laborious process which requires a lot of time and developers' effort.
4. Due to the fact that the rules have been constructed by the developers, they are not always matching all users' requirements and consequently they are not effective in such cases.

The aim of our work is to design a novel methodology and prove its value by implementing a service which deals with the shortcomings that described above. G-SPLIS's technical contribution can be summarized in the following:

1. The service adopts a well-known ontology, such as schema.org (also adopted by Google, Bing and Yahoo) for semantic interoperability, incorporating and extending dynamically its RDF Schema version. First of all, the schema.org ontology offers the capability to represent a wide range of physical and digital entities (persons, places, movies etc.), and also the associations between them. Furthermore, it can easily be connected with several services, such as Google+. Official mappings[9] with well-known ontologies such as Dbpedia, foaf, etc. have already been implemented as well and therefore it would be feasible to combine data from these sources with rules compatible with schema.org attributes such as these that are being supported by G-SPLIS.A consistently used and widely accepted ontology such as the above is crucial for the acceptance of the system by end-users and the reusability of its data and knowledge.
2. G-SPLIS possesses a fully dynamic knowledge base by allowing POI owners and regular users to add and edit properties and rules at run time and fully customize the provided information according to their requirements. Apart from this, instead of becoming obsolete in the long run, the system becomes more and more intelligent as soon as more rules are being added.
3. By supporting an intuitive user interface and engaging non-technical users to take part in the knowledge construction process (creating and sharing their rules), G-SPLIS assists in reducing the cost in rule creation process.
4. As soon as users take part in the knowledge construction process, their rules are more knowledgeable, intuitive and more efficient than those constructed by the systems' developers.

In addition to the above:

5. G-SPLIS provides highly targeted marketing opportunities because a) it supports POI owners with full customization concerning their data and their target group, providing them with the capability to insert into the system their specific group-targeted offering policies (target groups can be defined in rule condition, by customizing properties such as gender, age etc. e.g. male students under 20 years old)and b) it does real-time contextualization of content and offers in order to deal with information overload. Valid offers (i.e. offers which match a user's context, according to the rules of POI owners) are dynamically identified and highlighted to the regular user.
6. Regular users by belonging to social groups can gain additional functionalities.The proposed system presents high quality group-based information by combining regular users' preferences with those of their nearby friends to help them find their nearby friends that are interested in a specific POI and their nearby friends that have a valid offer, depending on their current context.

Regarding the scientific contribution of our approach the following should be mentioned:

1. A methodology to model and formalize human life patterns via logical rules is provided. As it was discussed earlier, this is the first time that user-defined logical rules are used in such a way in a LBSN. Created knowledge from user-defined rules will also help understanding user intentions and face challenges such as human behaviour prediction [20, 33].
2. Because of the fact that rules are declarative, users can easily conceive, create and edit them. Consequently, they are able to customize the service(s) according to their needs and not to rely only on the personalization system that each developer has implemented. Moreover, combining data from various sources (e.g. user profiles, POIs etc.), rules and ontologies provide users with great flexibility and advanced expressiveness. Therefore, users can easily define their preferences, cover a lot of possible situations and enjoy higher quality personalized information according to their requirements. For example, in our case, a) regular users can easily customize the time condition, the weather condition and other, b) POI owners some of the POI properties etc.
3. User-defined policies can easily be reused and enriched, based on the fact that representing rules in a formal way offers interoperability among different systems in the web. People cooperation and social intelligence can create large knowledge bases and, combined with other methods, provide more efficient solutions to users. For example, regarding user-defined rules for LBSNs, they can help in a lot of areas. First of all, in the "new city" problem (the problem that occurs when a user is visiting a city for the first time and so no historical data exists regarding his/her previous visits to be used for the current visit) and the data sparsity problem [34]. Traditional collaborative filtering techniques can be enhanced with such rules when not enough data are available. Furthermore, they can assist in the group-based problem and provide solutions for a group of (socially connected or not) people [35].
4. It deploys a methodology for sharing rules in a social context. This idea is relatively new and (to our knowledge) there is only one system called "ifttt"[10] that embraces this idea. However, this system concerns patterns for sharing data between different web services and G-SPLIS is a LBSN.

3. System's architectural design

The UML component diagram of Figure 1 illustrates G-SPLIS's architectural components. A basic module of the presented system is Sesame[11]. Sesame is integrated into java-based applications to perform processes such as storing, querying and inferring RDF triples [36].

[9] http://schema.rdfs.org/mappings.html

[10] https://ifttt.com/

[11] http://www.openRDF.org

Apart from the above, a computer understandable language is necessary to represent human understandable policies.In our case, RuleML and being more specific, Reaction RuleML (a clone of RuleML) was adopted because of the following reasons [37-41]:
First of all, it is a powerful markup language (XML with a predefined schema) and supports production (Condition Action) rules, which is the type of rules that was needed to represent POI owners' and regular users' rules. Apart from this, it offers interoperability among different systems on the web, by allowing rules to be represented in a generic manner. A user-friendly interface could be also implemented, since RuleML is structured enough to constrain the user from writing nonsensical expressions, but also expressive enough to allow flexibility.

RIF-PRD [42] could also have been employed instead, but it is not currently supported by tools as much as RuleML. It's worth mentioning that SWRL could be also used in the current version of the system, but since it is based on an open-world assumption which does not include negation-as-failure, which is necessary for a more expressible rule language, we have chosen not to use it.

In order to build a rule-based system like G-SPLIS, a rule representation language has to be translated into a computer executable language. Rule engines such as Jess, Drools, Prova [43-44] are being used to address this issue. Jess was chosen to implement the core of the proposed system because of the fact that it is a lightweight rule engine which connects well with web technologies that are necessary to build a system like G-SPLIS. Jess is based on an improved version of the Rete algorithm and uses adaptive indexing techniques to improve performance (full indexing can be too expensive). Liang et al. [43] implemented a comprehensive study of the performance and scalability for several rule engines. They performed a number of tests regarding different problem sets and the results signified that Jess is one of the best choices regarding our case (e.g. type of data, type of rules etc.).It's worth mentioning also that RuleML rules are transformed to Jess by using XSLT files [45].

Additionally, common client-oriented web technologies such as HTML, JavaScript and AJAX, were used for visualization. Java Server Pages (JSP) technology was chosen as server side component as well. JSP is based on Java and thus can easily be combined with Sesame and Jess [46].

It's worth mentioning that, apart from the standard web browser version of our system, a lightweight mobile[12] version of G-SPLIS for Android devices has been developed as well. The mobile application supports all functionalities about regular run time user personalized information (e.g. personal mode, friends' mode etc.). These processes are described in detail in the following section.

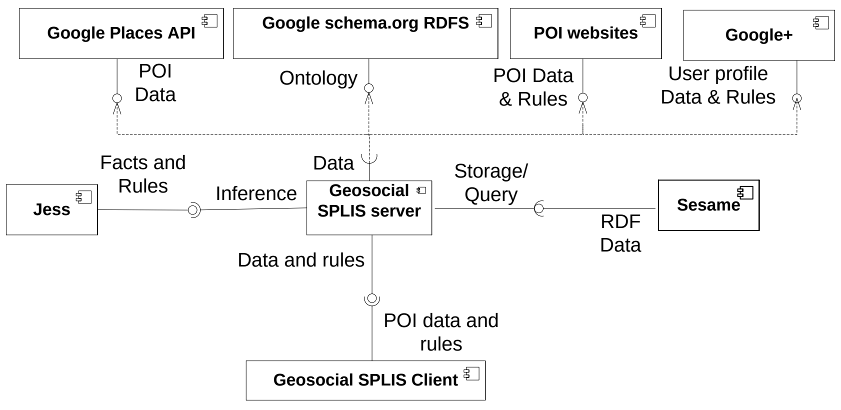


Figure 1. SPLIS architectural design

4. System processes

In this section, the system's processes are described in detail. It includes the following processes:
1. Data discovery and collection.
2. Information presentation.
3. Rule management and execution.
4. Social networking.
5. Planning mode.

4.1. Data discovery and collection

This operation is responsible for gathering and adding new data into the system repository, related to users' profiles and places. Regarding new users, they can register to the system by completing a simple registration form. Form fields are dynamically inferred from the ontology's related classes, namely the "Person" class of schema.org. Alternatively, they are able to connect G-SPLIS with their Google+ account (if any). In case they select this option, G-SPLIS collects profile data from their Google+ account (name, job title, age etc.). A data mapping is directly done as Google+ property names are compatible with the schema.org ontology. In a similar way, the system supports users with the capability to log into this with their Facebook account. The system is able to retrieve some standard profile attributes (gender, age, job, email etc.) and stores them into the repository in the form of RDF triples compatible with schema.org. The above functionality is being implemented by mapping manually each attribute with the corresponding schema.org attribute (e.g. Facebook's age field with schema.org age attribute).

[12]Can be downloaded at *http://tinyurl.com/GeoSoSPLIS*

After completing the registration process, the user is able to log into the system. Before going any further, it's worth mentioning that every time a user logs in his/herwith Google+ or Facebook account, the system updates existing data if there are any changes. If authentication process is completed successfully, POI data collection begins. The system obtains user's position (via GPS in a mobile device or via the IP address in a desktop/laptop PC) and retrieves the nearest POIs (so as to reduce the computational cost) from Google Places API. After that, it crawls their websites for additional data and rules. These data are stored in Sesame repository in the form of RDF triples.

4.2. Information presentation

After data collection, personalized information is presented to the end user. The following steps are included:

Data retrieval. After the data collection process, G-SPLIS server retrieves user data from the repository (properties, friendships, rules) and builds his/her context. Also by computing contextual property values (location, time, day, month, weather[13]) it builds system's context as it is defined in Appendix A. System context is volatile. It is created on the fly and it is not stored in the repository. Apart from the user's and the system's context there is also the POI's context [47].Each context type involves a different set of properties (see Appendix A).Regarding the weather contextual situation, it has to be referred that our system collects the current weather condition (e.g. sunny, rainy etc.) from the related external API that mentioned above. After finishing with user data, the server fetches data concerning nearby POIs (properties, owner, rules).A cache mechanism has been implemented, that stores recently retrieved POIs' properties and rules, in order to reduce system's response time in cases where multiple users are logged in concurrently from nearby locations.

Rule evaluation. Data mentioned above are asserted to the Jess production rule engine. Jess evaluates users' and POIs' rules using the asserted facts and updates POIs' property values according to the rules that fired. Regarding POIs' rules, Jess checks the conditions of a rule and concludes whether or not the values of the properties involved in the then-part of the rule have to be modified. The modified property values are not stored in the repository and they are used only for map visualization purposes, being calculated on the fly. About users' rules, it checks the if-part (they involve user's contextual properties and places' data) and concludes whether a POI is interesting or not for the user.

It's worth mentioning that in the mobile version (where the user's environment changes rapidly) the system: a) retrieves only 15 nearby POIs to reduce the response time (sorted by POIs that contain a group targeted offer), and b) updates user's position every 1 minute, providing updated information (if there is any change).

It has to be mentioned also that in the case of user preference rules, there can be no conflicting rules. Actually user preference rules recommend a place in their action. So when multiple rules fire, multiple places are recommended. Of course, these places might be conceptually conflicting, i.e. suggest eating at a restaurant or visiting a museum. Since the purpose of the system is to suggest options (possibly conflicting) and not to enforce decisions, we believe that a relaxed recommendation approach is appropriate. Furthermore, there can be no termination problems, since user preference rules are triggered by POI and / or system context properties and their effect is on the "recommend" property of the user context. Thus there can be no activation cycles among user preference rules.

In the case of POI offering rules, there can be conflicting rules. For example, there might be two rules offering different discounts for the same person. For example, one rule says "students get a discount 20%" while another rule says "women get a 10% discount on Thursday nights". It is the owner's responsibility to define which offer prevails by using the "salience" mechanism, which depends on a similar JESS mechanism. Specifically, the most important rule is the rule with the highest salience value and this rule is executed last in our system, contrary to the Jess mechanism that executes salient rule first. This is justified by the fact that the effect of the last rule persists, whereas the effects of the previously executed conflicting rules are overridden by the effect of the last executed rule.

Concerning termination problems, there can be no activation cycles among offering rules, since offering rules are triggered by system- and / or user- context properties, while their action has an effect on a POI context property.

Finally, from the above description it is obvious that there can be no conflicts between the two different rule sets (offering rules and user preference rules), because they have different predicates in their conditions and /or actions. However, there is always the possibility to have chained triggering between these two rule types, since the condition of the latter (POI context properties) is of the same type to the action of the former (POI context property) and vice-versa. However, the action of the user preference rules has an effect on just one property of the user context, namely the "recommend" property. Therefore, excluding this property from the condition of the offering rules we have avoided the possibility on rule non-termination problem.

Concerning the computational complexity and the execution time, since JESS uses the RETE algorithm[48] for matching rules, it is O(RFP), where R is the number of rules in the rulebase, F is the number of facts in the factbase, and P is the average number of patterns/conditions in the condition of the rules. In the case of G-SPLIS, there are two types of rules: the POI offering rules and the user preference rules, according to Appendix A.

In the case of offering rules the number of rules R is kept small, since we translate and run in Jess only the offering rules relevant for each POI, therefore R is small compared to the total number of rules. Furthermore, for each POI we use only the relevant facts, namely the system context (4 facts) and the user context for the currently logged-in user, which is again very small compared to the total number of users. In schema.org 43 properties are reported for the class sch: Person; however, in G-SPLIS we use a lot less (actually 4). Finally, the maximum number of condition patterns equals the sum of system and user context properties, which is actually 8. Therefore, the computational complexity is kept very low for offering rules.

In the case of user preference rules, the situation is similar to the case above. Specifically, the number of rules per user is a very small fraction of the total number of users' preference rules, the number of relevant facts includes the system context (4 facts) and place context (e.g. sch: Restaurant reports 63 properties), and the maximum number of patterns includes the sum of the system context and place context properties. In the case of POI context, it is more likely that users will include in their preference rules more properties than POI owners. Therefore, the computational complexity is a bit higher than that of offering rules but still kept a low.

In the case of nearby friends' mode operation, the situation is a bit more complex, since for each user and POI we run the rules both for the context of the user and for the context of his nearby logged-in friends. Therefore, the complexity changes to O (RNFP), where N is the number of logged-in nearby friends, while the rest of the parameters are as above. However, N cannot grow very large because it

[13]http://www.worldweatheronline.com API

is constrained by 2 facts: time (currently logged-in) and space (nearby within a distance threshold). Therefore, the computational complexity does not grow very much.

Presentation of personalized information. Finally, data transfer to the client is performed for illustrating favoured places and contextualized offers on Google map. A novel interface has been developed so that the user is able to spot POIs that are associated with his/her rules and contain valid offers for him/her. As shown in figure 2, on the map presented in G-SPLIS:

1. A bigger in size marker illustrates a POI which matches user preferences and it is recommended.
2. A red marker represents a POI with no offers at all.
3. A red star over the marker indicates that the current user is the owner of the specific POI (so the user is able to manage its properties and its rules).
4. A green marker is used for a place which contains a rule base of targeted offers and at least one rule has been fired for the specific user depending on his/her current context.
5. A yellow marker indicates that the place contains a rule base of targeted offers but no rule is fired for the current user.

It's worth mentioning that in Appendix A we include a formulation which describes the above process more accurately.

4.3. Rule management and execution

As it was discussed above, the rule management layer concerns either the POI owners or the regular users. To begin with the POI owners, they are able to add rule based group targeted offers via a user friendly web editor through completing specific forms. In more detail, POI owners are able to add rules that contain as constrains (IF part): a) any property related to the specific POI and/or b) any property regarding user context (e.g. Jobtitle) or system context (e.g. day, time and distance). POI owners' rules modify POIs' properties as a consequence (THEN part). For example, rules such as "If a person is a student and day is Saturday, then Espresso price has discount 10%" or "If time is between 12:00 and 17:00, Pizza price is reduced to 5€", regarding that Espresso and Pizza price are POI properties that have been created by the owners. It has to be mentioned that the POI owners are responsible for property addition and management. They can add their own properties (they can insert only POI properties) and select if the property will be attached into a class or only for their place. Thorough information about POI owners' operation processes can be found at [19].
A detailed description regarding regular users' rule based preferences follows.

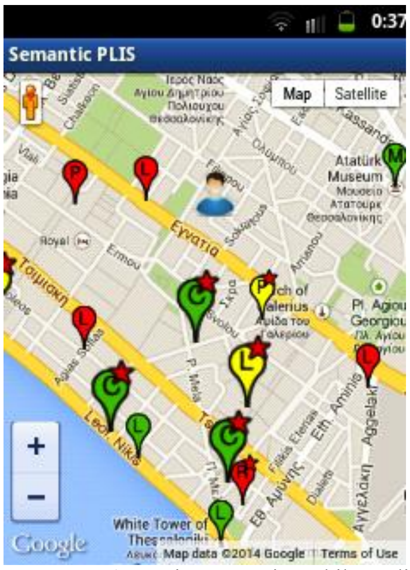


Figure 2.G-SPLIS's starting screen in mobile application

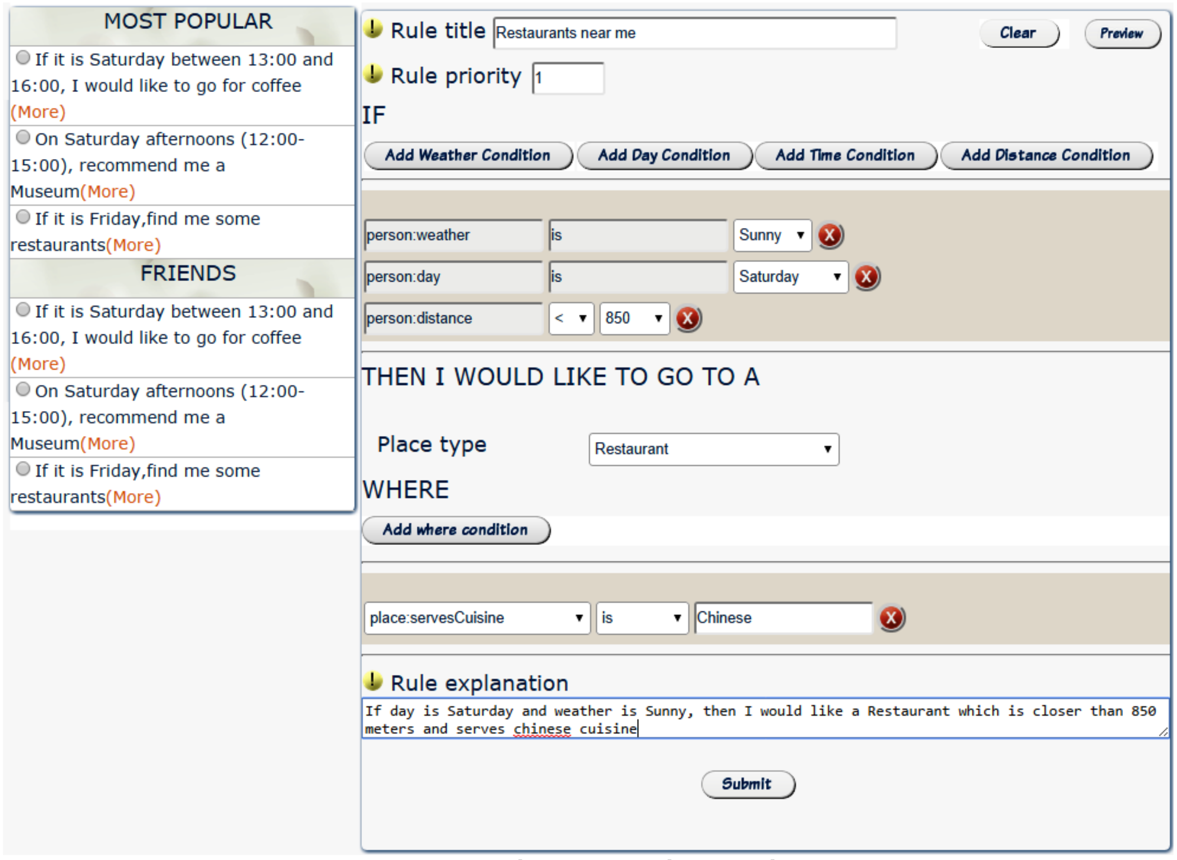
4.3.1. Rule editing

A user-friendly web interface has been designed so that the users can easily add their rules. An example which illustrates the rule creation through this editor is given in figure 3, where a user asserts the rule "*If day is Saturday and weather is Sunny, then I would like a Restaurant which is closer than 850 meters and serves Chinese cuisine*". As it can be seen in this figure, the web interface consists of the following four sections:

1. The form fields for inserting the title and the priority of the rule.
2. The four "Add...Condition" buttons, which are responsible for customizing the contextual condition. After clicking on one of this buttons, three form fields are generated:
   a) The property field (weather, day, time, distance),
   b) The operator field ("is" for day or weather and "<",">" for time or distance),
   c) The value.

   Elements concerning properties and operators are represented by read-only texts and value elements by drop down menus. This approach is adopted to resolve user data heterogeneity and avoid any mistakes. Users are also able to remove a condition by clicking on the red X button. By repeating this process, users can add as many conditions as they like; a logical "AND" is implied among them. If there is a need for an OR condition this can be achieved by adding multiple rules with the same action.
3. The place customization section. First of all, there is a drop down menu for the users to select the place category they like. The available categories are compatible with the schema.org class hierarchy. So, for example, if someone selects the "FoodEstablishment" place type, all its subcategories are included as well (e.g. Bakery etc.). In addition, by clicking on the "Add Where Condition" button

they are able to make their rule more specific by customizing the POI properties. After clicking on this button, a) a property drop down menu, b) an operator drop down menu ("is" and "contains" for text, or "<",">" for numbers and dates) and c) a value field are generated. In our example "Chinese" is a value for the POI property serves Cuisine.
4. The users' help section. A textual explanation of the rule is supported so that the meaning of the rule can become clear both to the rule creator and also to other users.
Additionally the editor provides a preview button to check the rule before submitting it and a clear button to reset the process. User can also click on one of the most popular rules, or on one of his/her friends' rules in the left side of the screen and the forms concerning this rule are automatically filled in.


MOST POPULAR
If it is Saturday between 13:00 and 16:00, I would like to go for coffee (More)
On Saturday afternoons (12:00-15:00), recommend me a Museum(More)
If it is Friday,find me some restaurants(More)
FRIENDS
If it is Saturday between 13:00 and 16:00, I would like to go for coffee (More)
On Saturday afternoons (12:00-15:00), recommend me a Museum(More)
If it is Friday,find me some restaurants(More)
Rule title
Restaurants near me
Clear
Preview
Rule priority
1
IF
Add Weather Condition
Add Day Condition
Add Time Condition
Add Distance Condition
person:weather
is
Sunny
person:day
is
Saturday
person:distance
<
850
THEN I WOULD LIKE TO GO TO A
Place type
Restaurant
WHERE
Add where condition
place:servesCuisine
is
Chinese
Rule explanation
Submit

```
If day is Saturday and weather is Sunny, then I would like a Restaurant which is closer than 850
meters and serves chinese cuisine
```

Figure 3. A rule creation via editor example

Afterwards, the rule that is created by using the above editor, is transformed to RuleML syntax (for interoperability with other systems on the web) and then (via XSLT) to Jess (to become machine executable). It has to be mentioned that if a user leaves the explanation field blank and does not insert any message, an XSLT file (presented in Appendix B) creates an explanation message automatically from the generated RuleML code.
Table 1 illustrates the above rule ("*If day is Saturday and weather is Sunny, then I would like a Restaurant which is closer than 850 meters and serves Chinese cuisine*") in RuleML and Jess. Concerning Jess representation, notice the JESS salience definition used for resolving rule conflict issues; it is used only in cases where two POI owners' rules concern the same slot e.g. "If it is Saturday Coffee costs 2 €", "If a person is a student coffee costs 1,5 €".Furthermore, class "Restaurant" is a subclass of class "Place", however in Jess we represent this as a value of the slot "type" in order not to create new templates for every subclass. Also, a template called "recommendation" is introduced in order to store the relative places that match the rule. Finally, a variable called "explanation" is introduced to store the rule explanation. After that, the automatically generated explanation message from the XSLT file is illustrated. Actually, the automatic explanation is not used in this case because the user has inserted his/her own explanation message.
Finally, rule data are stored in the form of RDF triples. Some of them, regarding the rule above, are illustrated in Table 2. An extension has been made to the Schema.org RDF/S ontology, by adding the corresponding policy class and its properties such as:

a) "policy title", which is a title for the rule
b) "policy explanation", a message explaining the rule
c) "policy priority", the priority of the rule
d) "policy description", a text which is automatically created from form data and is used for helping in the rule editing and in case the creator inputs a non-comprehensible explanation message.
e) "policy_link", a link containing the rule in RuleML syntax
f) "policy_jess", the rule in Jess syntax.

A user can also directly find all his/her rules and modify or delete them by selecting the related icon. The same form-based interface, presented above for the rule insertion process, is provided for updating existing rules.

Table 1. Rule representations in RuleML, Jess and RDF format

RuleML representation

```
<?xml version="1.0" encoding="UTF-8"?>
<RuleML …">
<Assert>
<Rule style="active">
<label>chinese cuisine</label>
<explanation>
If day is Saturday and weather is Sunny, then I would like a Restaurant which is closer than 850 meters and serves chinese
cuisine
</explanation>
<if>
<And>
<Atom><Rel>place</Rel>
<slot><Ind>type</Ind>
<Ind>Restaurant</Ind></slot>
<slot><Ind>servesCuisine</Ind>
<Ind>chinese</Ind></slot>
<slot><Ind>uri</Ind>
<Var>id</Var></slot></Atom>
<Atom><Rel>person</Rel>
<slot><Ind>day</Ind>
<Ind>saturday</Ind></slot>
<slot><Ind>weather</Ind>
<Ind>sunny</Ind></slot>
<slot><Ind>distance</Ind>
<Var>d</Var></slot></Atom>
<Expr><Fun>&lt;</Fun>
<Var>d</Var>
<Ind>850</Ind></Expr></And></if>
<then>
<Assert>
<Atom><Rel>recommendation</Rel>
<slot><Ind>id</Ind>
<Var>id</Var></slot></Atom></Assert></then>
</Rule>
</Assert>
</RuleML>
```

Jess representation

```
(defrule latysfwc
   (declare (salience 1))
   (place(type Restaurant) (uri ?id) (servesCuisine chinese))
   (person (weather sunny) (day saturday) (distance ?d)) (test (<  ?d 850))=>
   (assert (recommendation(id ?id)))
   (store EXPLANATION " If day is Saturday and weather is Sunny, then I would like a Restaurant which is closer than 850
meters and serves chinese cuisine")
)
```

Automatically generated explanation

If day is Saturday and weather is sunny, then I would like a Restaurant where serves Cuisine is Chinese, whose distance is less than 850

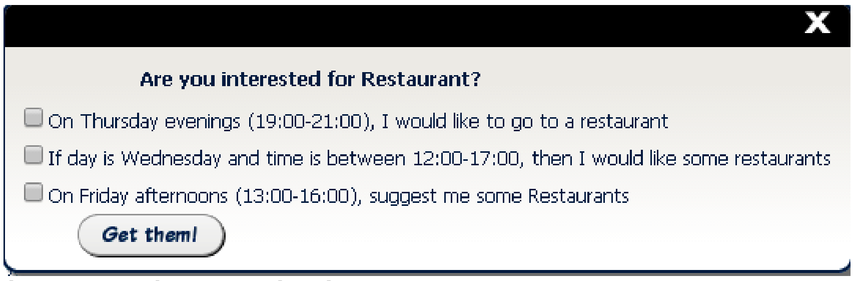
Table 2. RDF data about policy class.

```
@prefixsch: <http://schema.org/> .
sch:Person#19  sch:policy sch:policy1dc1e2a4-2c69-4e76-2a85-44423sb98w19.

sch:policy1dc1e2a4-2c69-4e76-2a85-44423sb98w19
sch:policy_description         "IF person:weather is Sunny AND person:day is Saturday AND person:distance< 850 THEN I
WOULD LIKE TO GO TO A place:typeRestaurant WHERE servesCuisine is Chinese";
sch:policy_explanation         "If day is Saturday and weather is Sunny, then I would like a Restaurant which is closer than 850
meters and serves chinese cuisine";
sch:policy_link"http://platon.econ.auth.gr/examples/plis_new_users/files/policy1dc1e2a4-2c69-4e76-2a85-
44423sb98w19.ruleml";
sch:policy_priority "1".
```

4.3.2. "Get-a-rule" process

To make the rule creation process easier and engage users as much as possible, G-SPLIS offers them the capability to obtain rules from other users. This capability is provided to them, in various ways. First of all, they are able to search among existing rules. For example they can search for rules by querying tags such as "Museum", "Sunny", etc. A list including all existing rules matching their query (if the user's search tag is contained in the policy explanation or description texts of a rule) is displayed and they are capable of incorporating them into their personal preference rule base by clicking the corresponding button. Additionally, G-SPLIS's starting screen displays a) the 3 most popular rules from all users and b) the 3 most popular rules from users' friends. Users can simply check and obtain them. Furthermore, as soon as a user makes a "check in" into a POI or a "like", a list of the 5 most popular rules concerning the POI category is also displayed in a pop up window. For example if they "like" a restaurant, the 5 most popular rules concerning restaurants will be displayed in a pop up window as in figure 4 below. Then the user simply can check and obtain any of them. Moreover, by clicking on their friends' profile they are able to view and obtain their rules. In order to avoid confusion when rules are modified (for example in cases where user A obtains a rule which was created by user B and then user B modifies it), as soon as a user edits any rule, a new rule is created. Similarly, if a user deletes a rule, the user is simply "unlinked" from the rule so as not to affect other users that have this rule. The rule is deleted if no one uses it.


X

Are you interested for Restaurant?

- [ ] On Thursday evenings (19:00-21:00), I would like to go to a restaurant
- [ ] If day is Wednesday and time is between 12:00-17:00, then I would like some restaurants
- [ ] On Friday afternoons (13:00-16:00), suggest me some Restaurants

Get them!

Figure 4. A pop up window containing the most popular rules.

4.3.3. Rule insertion via Google+ account

The system also provides users with the capability to add rules via their Google+ account. Google+ was selected because a) it is compatible with the schema.org ontology and b) it offers users the capability to upload links (actually a URL and a label describing it) to their profile page. Similarly to the other properties of the profile (e.g. name, job title, etc.), these links are available to other services through the corresponding API. Therefore, it is feasible for a user to upload a link to his/her profile, pointing to a RuleML file that contains rules in the format illustrated above, set as label the tag "policy_link" (see figure 5) and then share these preferences with other services. This operation has been implemented in order to demonstrate how social network profiles could be enhanced and contain information which represents no just simple data but as in our case, contextualized preferences concerning POIs. If users have difficulties in developing such rules, they can use G-SPLIS's web editor and add a rule as it was described at the previous section. After that they are able to receive a link which points to the RuleML file that has been stored to the server. An alternative would be to use any XML editor, provided they have access to the appropriate version of the RuleML schema file[14]. Finally, after users log into G-SPLIS with their Google+ account, the system parses the links that point to the RuleML files, converts them to Jess and then execute these rules on the fly. Rules are not stored to the repository to avoid maintenance problems (e.g. in case users change their policies links into their Google+ account).

4.4. Social networking

Social interactions play an important role in LBSNs. Apart from common social interaction processes, G-SPLIS supports a mode concerning a user and his/her logged in nearby friends. These processes are described in detail below.

4.4.1. Common social interaction processes

The system supports the creation of social ties. Users are able to search for new people, view their profile data (name, age etc.) and send a request message to them, asking to become friends. After two users create a friendship, they are able to view each other rules as well.

4.4.2. Nearby friends' mode process

A user is also able to spot his/her friends which are nearby and find common places and offers. In this mode G-SPLIS:

1. Collects a) user's rules, b) his nearby (logged in) friends' rules and c) contextual information.
2. Evaluates all the above rules (a) and (b) and fetches the nearby POIs which are recommended by the users' fired rules.
3. For these POIs, it gets their group targeted offers (POIs' rules), and evaluates them concerning all users contexts (the user and his/her friends).
4. Provides personalized information by displaying:
   a) With a red marker a POI that does not have any offers at all.
   b) With a yellow marker a POI that has at least one offer, but none of them is valid for any of the friends or the user at that moment.
   c) With a half yellow-half green marker a POI which has a valid offer for at least one of the friends or the user.
   d) With a green marker a POI that has a valid offer for all of the friends and the user.
   e) With a bigger marker a POI that is recommended by a user rule and at least one of his/her friends' rules (common POIs).

Appendix A includes a formulation which describes this process in detail. This process has been implemented to demonstrate how user - defined rules will help in solving group-based problems. A use case scenario in the next section will make this process more comprehensible and denote its usability.

[14] "http://www.ruleml.org/0.91/xsd
http://dl.dropbox.com/u/3157468/pr.xsd"

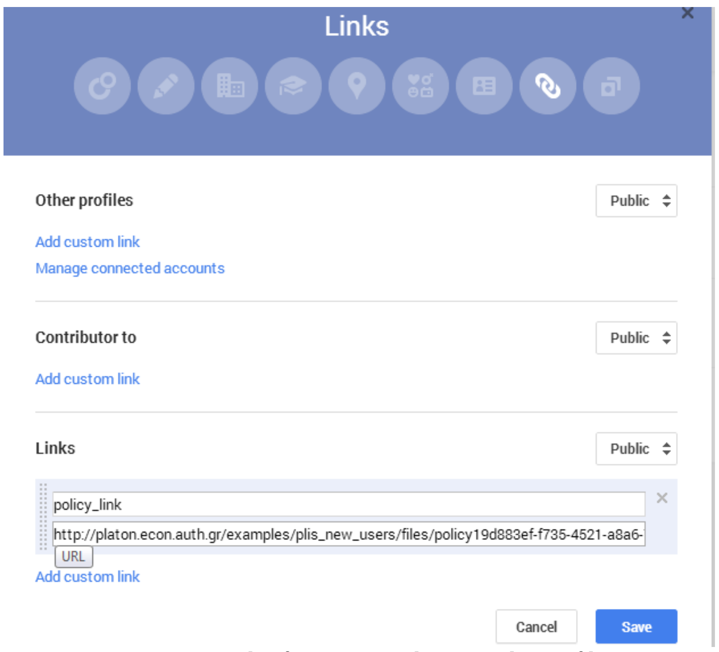


Figure 5. An example of inserting a rule to Google+ profile page

4.5. Planning mode

An additional process which has been developed to enhance the technical usability of our system is the "planning mode".When the user selects the planning mode, he/she can set the day and time he/she would like to visit a location. More specific, he/she is able to select a) after how many days he/she will visit a location and b) the time of the day that he/she will be there. After that, the system collects user's future contextual situation (day, time and weather condition) and evaluates his/her rules and POI group targeted offers depending on the future situation. It's worth mentioning that a) by right clicking on the map they can change position and b) the number of days is restricted to 5 due to available future weather information. This process will become apparent with a use case scenario in the following section.

5. Use case scenarios

In this section some use case scenarios are demonstrated so as G-SPLIS's functionalities to become explained better. A scenario concerning individuals will be presented in section 5.1, while in section 5.2 a scenario which exploits social ties is described. Finally section 5.3 demonstrates the systems capabilities in planning events.

5.1. Scenario concerning individuals

At first, a use case scenario concerning two different persons is presented in order to demonstrate G-SPLIS's individual user's functionality. The scenario regards two individual users, James and Linda. We assume that these users are socially connected and have the following profiles:
User A ("James") is a 19-year old male student (table 3), and his current profile snapshot is taken on Sunday, at 14:15 in a location A where the weather is sunny. In the same spirit, User B ("Linda") is a 20-year old female student, who is logged in the system at the same time with James, in a location B which is near location A.
We assume that James and Linda have used the G-SPLIS's web editor (or their Google+ account as discussed in the previous section) and they possess the rules that are illustrated in table 4 below.
As it was referred above, after a user logs into the system, it gets his/her context, his/her rules and evaluates those rules along with those of the nearby POIs. As a result, after James makes a login into the system, rule 2 is fired because it is Sunday and time is 14:15. Consequently, available coffee shops are represented with a bigger marker and are suggested to him (figure 6). In order to help the user to find easier a POI category, the marker contains the first letter of the category it belongs (e.g. "C" if it is a CafeOrCoffeeShop). By clicking on the nearby POIs, James can get personalized info. As it was mentioned above, a big green marker represents a place he would like to go regarding his context, which has also an offer for him. Taking for example the POI "A Nice Pair" (table 5) which is represented with a big green marker, he is able to view, as shown in figure 7a, POI's data, the POI owner's message for the group targeted offer that matches his profile and his rule which was fired and recommended this place. Notice that the POI owner's offer ("Special freddo prices, only 2.5 euro for students") is valid for him because he is a student. The property "freddo" is also highlighted because it has been modified due to the owner's rule and its value for James is different than the standard value (for users that are not students). To avoid confusion, a user's rule/preference is illustrated with a person icon in front of the message and a POI owner's rule is represented by a marker icon (see figure 7). James can also contribute by writing a review, adding a rating, adding "likes", or even making a "check in". In other words, users do not only provide useful information to other people (e.g. to help them find a popular place) but they also create information that can be used during rule evaluation (e.g. place average rating). He can also view the reviews and ratings which have been submitted by other users. Apart from the available places, James can directly obtain some of the three most popular rules regarding all users and his friends (see left side of figure 6). Similarly for Linda, rules 1 and 3 are fired for her. As a result, coffee shops and museums are represented with bigger marker and suggested to her. Similarly, if she clicks on "A Nice pair" she gets the contextualized information illustrated in figure 7b.

Table 3. User data in RDF form

```
@prefixsch: <http://schema.org/> .
@prefix sch-per: <http://schema.org/Person#> .

sch-per:12   rdf:type sch:Person .
sch-per:12   sch:name     "James" .
sch-per:12   sch:birthDate     1995-02-10 .
sch-per:12   sch:gender   "male" .
sch-per:12   sch:jobTitle "student" .
```

Table 4. Users' rules

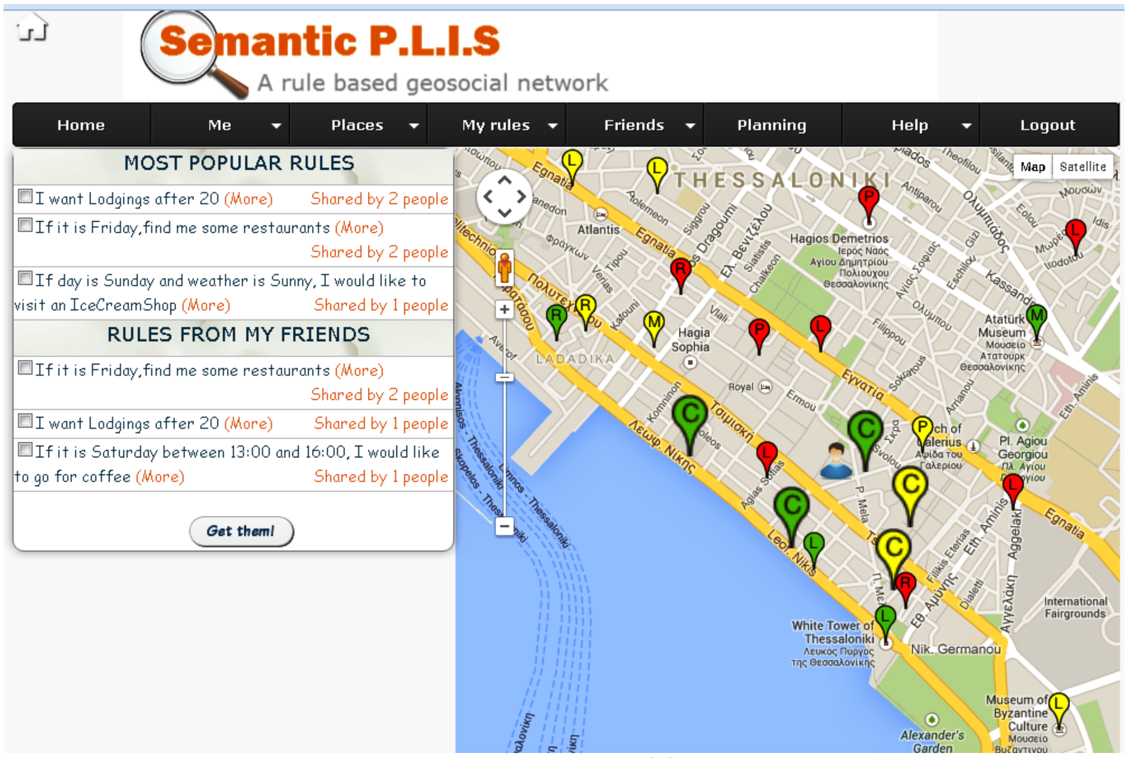
James's rules		Lindas's rules
Rule 1	'On Thursday evenings (19:00-21:00), I would like to go to a restaurant"	"If it is Sunday between 11:00 and 16:00, find me a museum"
Rule 2	'If it is Sunday and time is between 12:00 and 15:00, I would like to visit a coffee shop "	"On Monday's I would like to go to a restaurant which serves Chinese cuisine and is closer than 1000m"
Rule 3	'Find me an ice-cream shop if the weather is Sunny and it's Friday"	"I would like to go for coffee, if weather is Sunny and it's Sunday"

Table 5. Data about POI "A Nice pair" in RDF form

```
@prefixsch: <http://schema.org/> .
@prefixsch-p: <http://schema.org/Place#> .

sch-p:024f5a7a49cfc3aa483057c4eda2962df173fcaa   rdf:type  sch:CafeOrCoffeeShop .
sch-p:024f5a7a49cfc3aa483057c4eda2962df173fcaa   sch:curranciesAccepted "euro" .
sch-p:024f5a7a49cfc3aa483057c4eda2962df173fcaa   sch:email     "nicepair@gmail.com" .
   sch:policy sch:policy92431b85-fa65-4g93-b5d2-3dbbe8495aw4 .

                                        ….
```


Semantic P.L.I.S
A rule based geosocial network

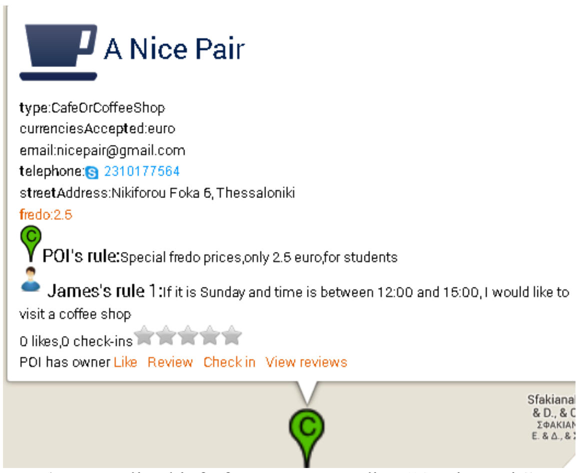
Home | Me | Places | My rules | Friends | Planning | Help | Logout

MOST POPULAR RULES
- I want Lodgings after 20 (More) Shared by 2 people
- If it is Friday,find me some restaurants (More) Shared by 2 people
- If day is Sunday and weather is Sunny, I would like to visit an IceCreamShop (More) Shared by 1 people

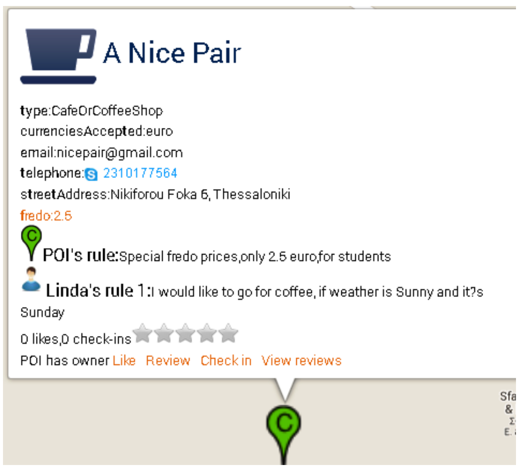
RULES FROM MY FRIENDS
- If it is Friday,find me some restaurants (More) Shared by 2 people
- I want Lodgings after 20 (More) Shared by 1 people
- If it is Saturday between 13:00 and 16:00, I would like to go for coffee (More) Shared by 1 people

Get them!

Figure 6. James starting screen in web browser version


A Nice Pair

type:CafeOrCoffeeShop
currenciesAccepted:euro
email:nicepair@gmail.com
telephone: 2310177564
streetAddress:Nikiforou Foka 6, Thessaloniki
fredo:2.5
POI's rule:Special fredo prices,only 2.5 euro,for students
James's rule 1:If it is Sunday and time is between 12:00 and 15:00, I would like to visit a coffee shop
0 likes,0 check-ins
POI has owner Like Review Check in View reviews


A Nice Pair

type:CafeOrCoffeeShop
currenciesAccepted:euro
email:nicepair@gmail.com
telephone: 2310177564
streetAddress:Nikiforou Foka 6, Thessaloniki
fredo:2.5
POI's rule:Special fredo prices,only 2.5 euro,for students
Linda's rule 1:I would like to go for coffee, if weather is Sunny and it?s Sunday
0 likes,0 check-ins
POI has owner Like Review Check in View reviews

a)Personalized info for James regarding "A Nice Pair" b) Personalized info for Linda regarding "A Nice Pair"

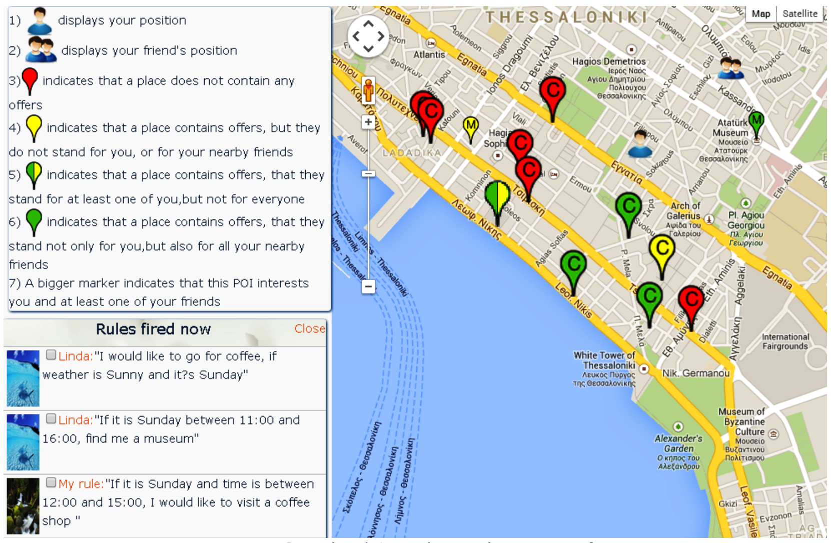
Figure7. Personalized info concerning the two users and the place "A Nice pair"

5.2. Scenario exploiting social ties

As it was mentioned above, an operation for providing group-based suggestions was implemented in order to demonstrate the technical capabilities of user-defined logical rules. Users can select the "Nearby friends" option of the menu to see which of their friends are nearby. Taking for example the user A (James) above, we make an assumption that Linda is his only nearby friend which is logged in at this time. After James selects the "Nearby friends" option and visits the related page, G-SPLIS:

1. Collects his and Linda's context and rules.
2. Evaluates all the above rules, and then fetches the nearby POIs which are recommended by the fired rules. In our case, James's second rule is fired and thus the system suggests coffee shops. Furthermore, Linda's rules 1 and 3 are fired. Consequently, the system retrieves and suggests museums (apart from nearby coffee shops).
3. For these POIs that result by the set of their rules, the system gets their offers (POIs' rules), and evaluates them based on James and Linda's contextual situations.
4. After that, it displays personalized information as it was discussed in section 4.4.2.

James' starting screen in "Nearby Friends' mode" is illustrated in figure 8. According to the above, all coffee shops (markers containing the letter "C") are displayed with a bigger marker because of the fact that Linda would like to go to a coffee shop at this time as well. In addition, museums are represented with a small marker because they interest only Linda. Also on the left side of the screen there is a description of the icons so as to help user understand easily the general concept. Below them, all the rules that are being fired and their possessor are also illustrated. By clicking on the corresponding markers James is capable of finding easily common places with his nearby friends (big markers), places with offers for all of them (green markers) etc. For example, he can directly find a POI where both of them would like to go, which has also an offer for him and Linda (a big green marker). After clicking on a marker he is able to view the POI rules (if any) and the user-defined rules that are being fired for this place, in order to understand who has an offer and why, or who would like to visit this POI at the moment. Taking for example the POI "A Nice Pair" which is represented by a green marker, he is able to view, as it is presented in figure 9a, that the offer is valid for both of them (they are students) and that both of them would like to go there. Similarly concerning the POI "Mr Jones Cafe" (figure9b), James can easily observe that they both would like to visit this place and that the offer is valid for neither of them because it is not Saturday.


1) displays your position
2) displays your friend's position
3) indicates that a place does not contain any offers
4) indicates that a place contains offers, but they do not stand for you, or for your nearby friends
5) indicates that a place contains offers, that they stand for at least one of you,but not for everyone
6) indicates that a place contains offers, that they stand not only for you,but also for all your nearby friends
7) A bigger marker indicates that this POI interests you and at least one of your friends

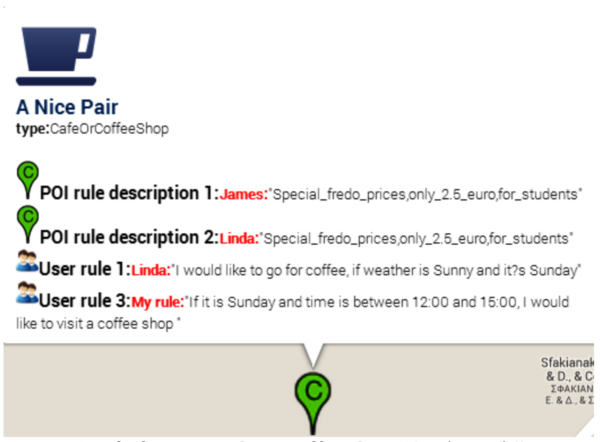
Rules fired now Close

Linda:"I would like to go for coffee, if weather is Sunny and it?s Sunday"

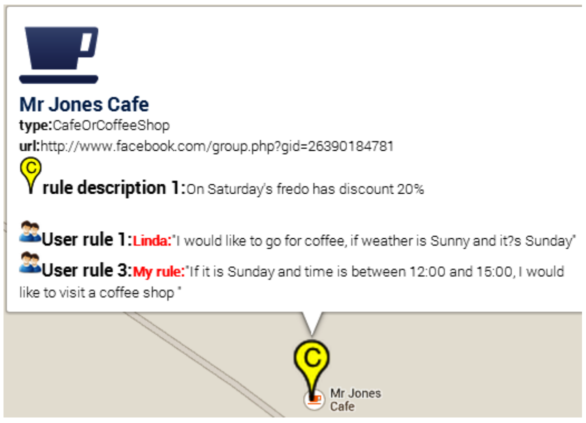
Linda:"If it is Sunday between 11:00 and 16:00, find me a museum"

My rule:"If it is Sunday and time is between 12:00 and 15:00, I would like to visit a coffee shop "

Figure 8. NearbyFriends' mode starting screen for James


A Nice Pair
type: CafeOrCoffeeShop
POI rule description 1: James:"Special_fredo_prices,only_2.5_euro,for_students"
POI rule description 2: Linda:"Special_fredo_prices,only_2.5_euro,for_students"
User rule 1: Linda:"I would like to go for coffee, if weather is Sunny and it?s Sunday"
User rule 3: My rule:"If it is Sunday and time is between 12:00 and 15:00, I would like to visit a coffee shop "

a) Info for James about coffee shop “A Nice Pair”

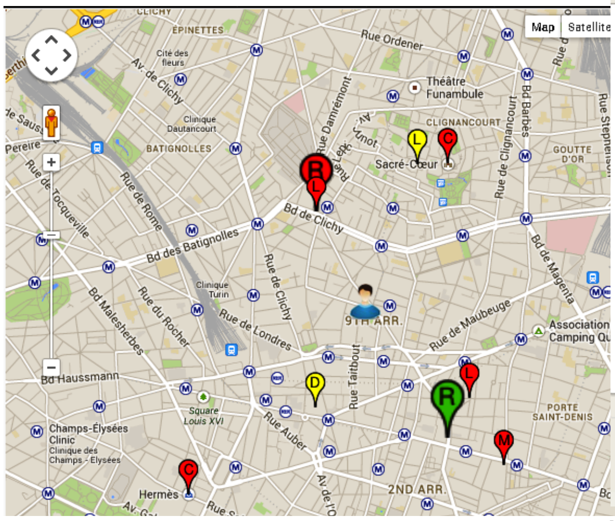

Mr Jones Cafe
type: CafeOrCoffeeShop
url:http://www.facebook.com/group.php?gid=26390184781
rule description 1: On Saturday's fredo has discount 20%
User rule 1: Linda:"I would like to go for coffee, if weather is Sunny and it?s Sunday"
User rule 3: My rule:"If it is Sunday and time is between 12:00 and 15:00, I would like to visit a coffee shop "

b) Info for James about coffee shop “Mr Jones Cafe”

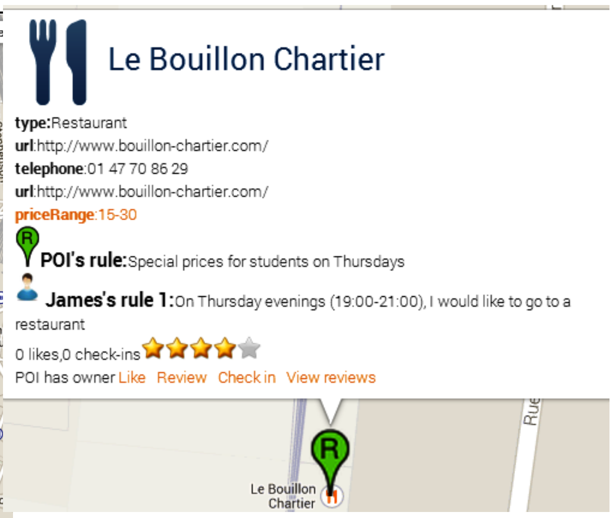
Figure 9. Friends’ mode starting personalized information for James

5.3. Scenario concerning the “planning” process

G-SPLIS supports a “planning mode” process which helps users planning future events. Let’s now assume that James has to visit Paris on Thursday evening and would like to receive information about POIs and offers there. In order to do this, he selects the “Planning” button from G-SPLIS’s menu and in the pop up window he selects the option “after 4 days” and the time he will be there, that is “20:00” in our case. After that, he is entered into “planning mode” and by right clicking on the map in Paris he is able to spot places and offers matching his preferences and POIs’ rules for the future situation (figure 10a). Because of the fact that the future contextual situation is on Thursday at 20:00, the rule “On Thursday evenings (19:00-21:00) I would like to go to a restaurant” fires. As a result the system recommends him some restaurants, which are being represented with a bigger marker. By clicking on one of these markers, he can get future, contextualized information, finding offers that are valid for him, the time he will be there. Taking, for example, the restaurant “Le Bouilon Chartier”, he can understand why it is recommended to him by reading the textual explanation of his rule based preference (figure 10b). Apart from matching his preferences, this POI also contains a valid offer for him regarding the time he will be in Paris. Similarly, by reading the POI owner’s explanation message, he is able to understand the reason why the offer is valid.



a)”Planning mode” screen for James


Le Bouillon Chartier
type: Restaurant
url:http://www.bouillon-chartier.com/
telephone:01 47 70 86 29
url:http://www.bouillon-chartier.com/
priceRange:15-30
POI's rule: Special prices for students on Thursdays
James's rule 1: On Thursday evenings (19:00-21:00), I would like to go to a restaurant
0 likes,0 check-ins ★★★★☆
POI has owner Like Review Check in View reviews

b) Future contextualized information for James regarding Restaurant “Le BouilonChartier”

Figure 10. Planning mode regarding James

6. Evaluation

In order to evaluate the proposed system and its functionalities, an extensive diverse evaluation was conducted, concerning the system’s response time and its usage in general.

6.1. Response time evaluation

A series of experiments were conducted to evaluate the system’s response time regarding real time usage scenarios. To examine the system under stretching conditions, the imposed scenarios were concerning the system’s response time when many users are logged in the system concurrently (e.g. a crowd of students attending a concert near the campus). In the scenarios, the response time of the system was measured, for the cases of 50, 100, 150, 200, 250, 300 and 350concurrent user calls for a typical user having 10 rules, being in a typical geographical position of 25 POIS that contain a total of 10 rules. Each call involves the downloading of G-SPLIS starting screen, such as that presented in figure 6, which involves the information presentation process. It should be noticed that concurrent user calls mean that the users are trying to download the starting screen at exactly the same time, which is stretching and different from the number of users that are logged in the system (that can be more). Apache JMeter[15], a program for testing a server's performance was used, in order to simulate the user calls and conduct the experiments. The experiment was conducted 30 times for every number of user calls in order to achieve better reliability in the results. All the experiments were run on a server with the following specifications: Windows Server 2003, Intel Xeon E5405 2.0 GHz, 4 cores as a processor, 8.0GB RAM. Apart from the average response time, standard deviation, minimum and maximum response times were measured for better understanding of system's behaviour. Table 6 and figure 11 illustrate the results of the experiments and demonstrate that the system (and consequently Jess and Sesame) performs well even in such stretching scenarios.For example, in the very stretching situation of 200 concurrent user calls the average response time is less than 100msec and the maximum response time is kept below than 1sec.

In order to examine user's browsing experience based on the system's performance, Nielsen's scale was taken into account [49]. According to this, if a system's response time is less than 0.1 second, it is regarded as instant access. Also a system's response time which is less than 1 second is regarded as the limit for users' flow of thought to stay uninterrupted.

Taking the above into consideration, the results concerning our experiments are presented in Table 7 and figure 12. Based on these, the following conclusions can be drawn regarding G-SPLIS:

1. For the cases of up to 100 concurrent user calls, G-SPLIS provides instant access, since it is capable of handling 100% of them in less than 0.1 second.
2. For the cases of up to 200 concurrent user calls, the users stay uninterrupted since the system is able to handle the 100% of them in less than 1 second.
3. Even in the very stretching case of 350 concurrent user calls, 78% of the users stay uninterrupted since they have access in less than 1 second.

The cache mechanism assists in achieving the above performance in such stretching scenarios. It is apparent that the results can be improved even further by making additional adjustments (e.g. upload the system into faster hosting servers, implement load balancing techniques in multiple servers, etc.).

Table 6. Response time (in milliseconds) evaluation results.

Number of concurrent user calls	**50**	**100**	**150**	**200**	**250**	**300**	**350**
Average response time in msec	9	12	46	94	376	892	1220
Standard Deviation	8	11	64	205	825	1534	1901
Minimum response time in msec	3	3	4	3	3	4	3
Maximum response time in msec	47	65	223	889	2479	3844	5217

Table 7. Response time evaluation results regarding the number of user calls.

Number of concurrent user calls	0	50	100	150	200	250	300	350	400
lessthan 0.1 sec	100%	100%	100%	95%	86%	80%	72%	69%	74%
lessthan 1 sec	0	0%	0%	5%	14%	8%	17%	9%	7%
morethan 1 sec	0	0%	0%	0%	0%	12%	11%	22%	19%

[15] http://jmeter.apache.org/

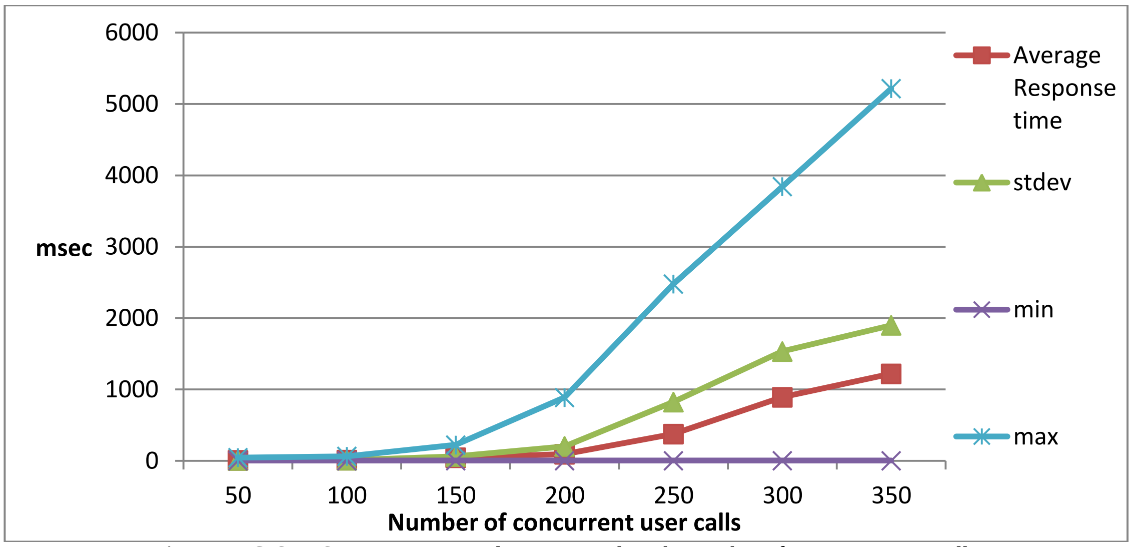


Figure 11.G-SPLISresponse time evaluation regarding the number of concurrent user calls

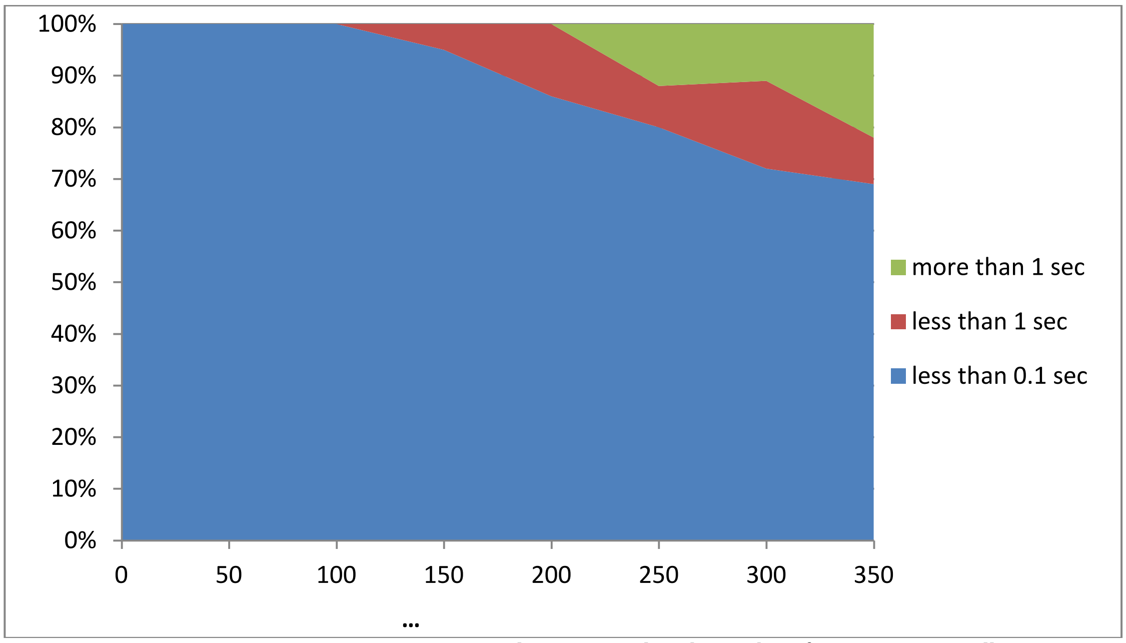


Figure 12. . G-SPLIS response time diagram regarding the number of concurrent user calls.

6.2. Qualitative evaluation of system's usage

A survey was conducted to evaluate the system's general idea and its usability. The survey sample consists of two different groups regarding the age, the technical and educational background. The first group included 83 undergraduate students of economics, between 18-22 years old, equally distributed between the two genders.

The second group included participants from different age groups with various technical and educational backgrounds. In detail, 18 people over 23 years old (between 26 and 51) took part and answered exactly the same questionnaire in order to compare the results. Concerning some useful information about the sample:

1. The participants were almost equally distributed between the two genders.
2. Regarding the age, 28% of the participants were between 23 and 30, 61% between 30 and 40 and 11% were over 40.
3. Regarding the participants' technical background, 61% answered that they considered their shelves very good or good users of personal computers and the rest 39% answered that they considered their shelves moderate or not good.

All the participants were asked to use G-SPLIS and afterwards answer some questions in an electronic questionnaire that was created. The questions had either 5 possible answers, a) Not all, b) Barely, c) Moderately d) Sufficiently and e) Very much (values from 1-5) or 2 a) No b) Yes. They were grouped into the following three themes:

1. Processes concerning rules and the personalization of information.
2. Social processes.
3. The system in general.

The next sections demonstrate and compare the results regarding the two samples, a) the students between 18-22 years old and b) the participants over 23.

6.2.1. Processes concerning rules and presentation of information

After a short introductory presentation to the system's main purposes and its functionalities, the participants created an account and were logged in. Initially, they were asked to add the rule "If day is Monday, then I would like a Restaurant" and then modify it to "If day is Wednesday, then I would like a Restaurant". After completing these tasks, they were asked to get a rule of their preference from another user and search for nearby POIs that match their rules. Finally, they answered the following questions:

Q1. How easy was to add a rule?
Q2. How easy was to modify a rule?
Q3. Are you satisfied with the provided interface?
Q4. How easy was to find and get a rule from another user?
Q5. How easy was to understand why a place was recommended?
Q6. How easy was to find a place that resulted by your rules and had an offer for you?

Figure 13 presents the results. It's worth mentioning that nearly over 80% of the participants of both samples were "sufficiently satisfied" or "very much satisfied" in every question. In addition to this, a mean value was calculated for each question depending on the answers of the participants. The average of the mean values, regarding the questions Q1-Q6, was calculated to 4.31 for the first group (18-22) and 4.38 for the second (23+). In order to validate if there are any statistically significant differences among the answers of the participants of the two age groups, one way ANOVA was conducted on the mean values of the questions Q1-Q6 [50]. Since the p-value was calculated at 0.53, which is more than 0.05 that the method requires, it can be concluded that there are no significant differences between the two groups (Table 8).

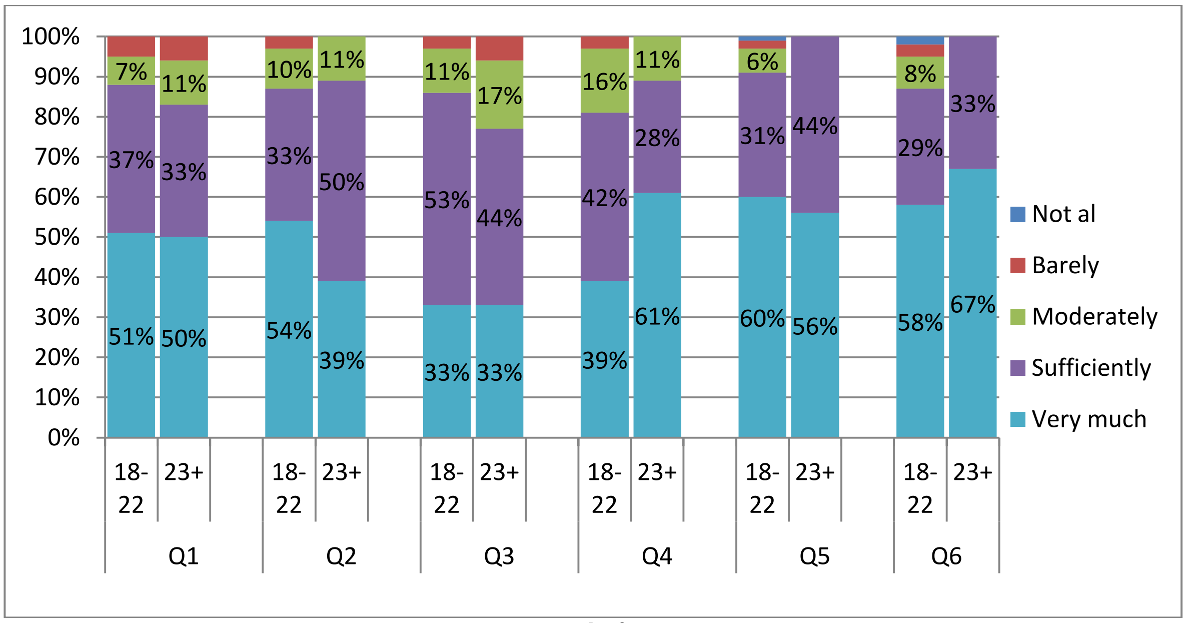


Figure 13. Survey results for questions Q1-Q6

Table 8. One-way ANOVA results

	Sum of squares	Mean square	F	p-significance
Between approaches	0.0147	0.0147	0.424	0.529

6.2.2. Social processes

As mentioned above, the second group of questions concerned the usability of the available social processes. The participants made groups of two or three persons, became friends with each other, and tested the "nearby friends' mode". Finally, they answered the following questions:

Q7. How easy was to send a friend request?
Q8. How easy was to understand which of your friends recommended a place and why?
Q9. How easy was to find common places for you and your friends?
Q10. How easy was to find places that resulted by your friends' rules and had an offer for you?

Figure 14 below displays the results, indicating that the participants of both samples coped very well with the system's social layer. Once again, in every question, over 80% of the participants of both samples stated that they were "sufficiently satisfied" or "very satisfied". The average of the mean values, regarding the questions Q7-Q6, was calculated to 4.28 for the first group (18-22) and 4.16 for the second (23+). In order to validate if there are any statistically significant differences among the answers of the participants of the two age groups, the same process was followed as above (one way ANOVA). Because of the fact that the p-value was calculated at 0.48, which is more than 0.05, there are no significant differences between the two groups (Table 9).

Table 9. One-way ANOVA results

	Sum of squares	Mean square	F	p-significance
Between approaches	0.026	0.026	0.546	0.487

6.2.3. System in general

After completing the above tasks, participants were asked to answer the following questions related to the system in general:

Q11. Will you continue using the system?

Q12. Would you recommend the system to your friends?

As illustrated in Figure 15, over 94% of the participants of both samples answered that they will continue using the system or recommend it to their friends.

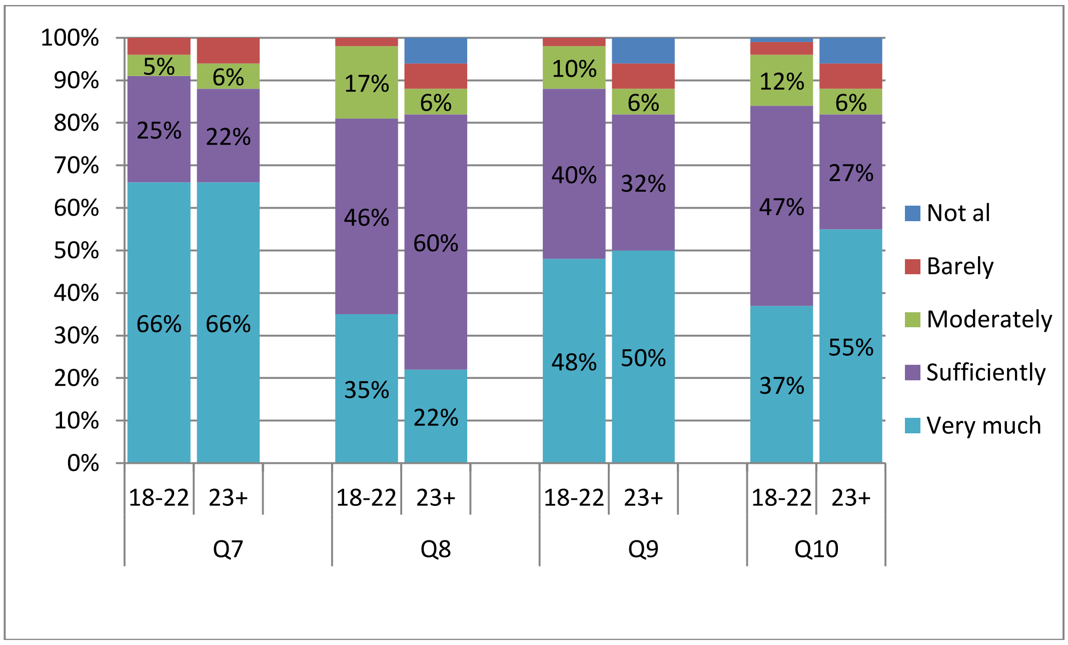


Figure 14. Survey results for questions Q7-Q10

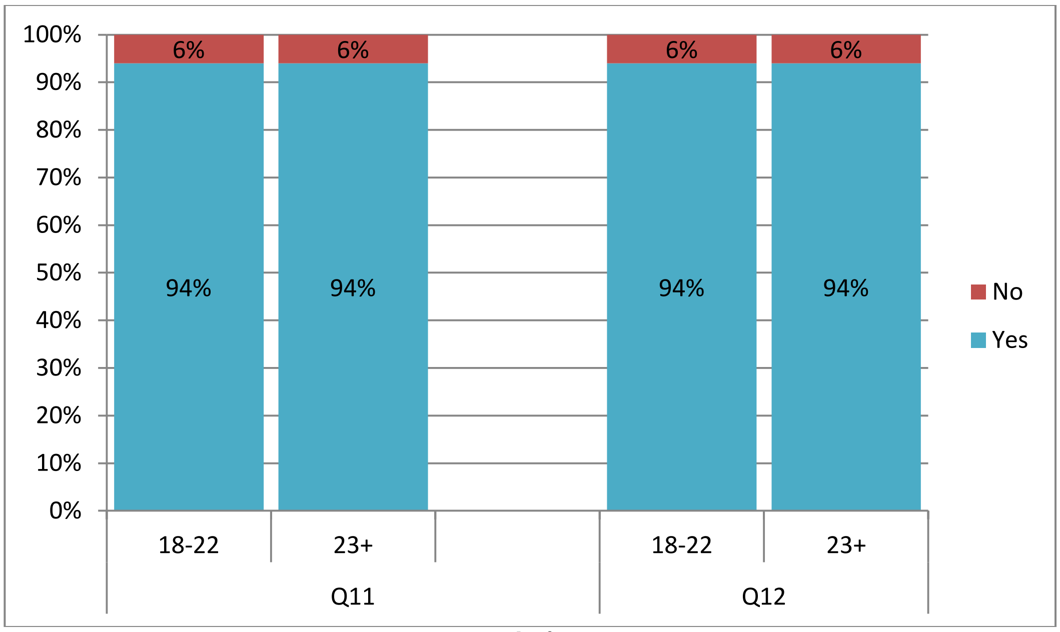


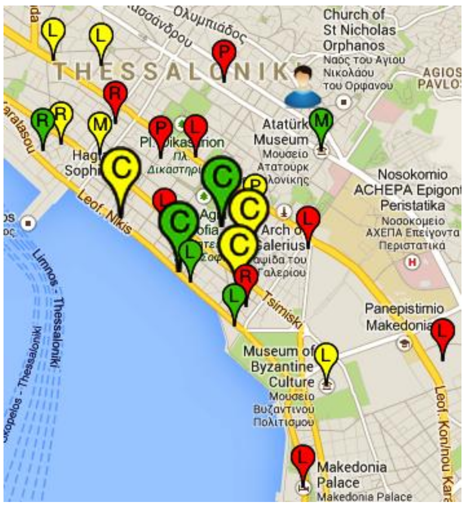
Figure 15. Survey results for questions Q11-Q12

6.3. Quantitative evaluation of system's usage

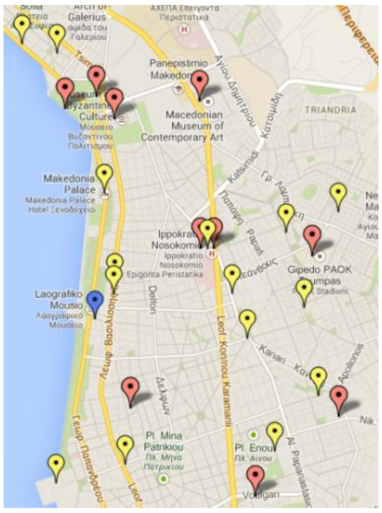
Apart from the qualitative approach above, a quantitative evaluation was also performed. In order to do so, our system, as shown in figure 16a, was compared with a stripped-down version of the same system. The second version had only red and yellow markers as illustrated in figure 16b below. A red marker indicates that a place does not have any offers and a yellow marker indicates that a place has an offer, which may match or not user's profile. Each participant had to click on the markers and a) view the POI's details so as to understand if it matches his/her preference and b) read POI owner's message to understand if the offers matches his/her profile or not.

After a short introduction to the general idea of the systems, an imaginary scenario was examined. The participants were informed that they had the following profile:

1. They were students (as actually).
2. They had the preference "If day is Wednesday, find me some Coffee shops".
3. They had a friend nearby which is logged in the system, who is unemployed and had the following preferences:
   a) "If day is Wednesday, recommend me some Restaurants"
   b) "I would like a Coffee shop if it is Wednesday"



a)G-SPLIS



b) Stripped-down form of G-SPLIS

Figure 16. Illustration of the compared systems

These two profiles with the above rule-based preferences have been created in G-SPLIS in advance so as the participants only have to log in and do the tasks. It's worth mentioning that these rules were chosen in order to be fired when the experiment took place. In the stripped-down version (which did not support rule evaluation and social connections) the above assumptions about their profile were just told to the participants. A number of 84 undergraduate students of economics took part in this evaluation and they were asked to perform the following 6 tasks, first in the stripped-down version of the system and afterwards in G-SPLIS:

Task1. Find how many places match with your preference and contain an offer which also matches with your profile.
Task2. Find how many places contain an offer which matches with your profile.
Task3. Find how many places match with your preference and contain an offer that does not match with your profile.
Task4. Find how many places you have in common with your friend, regarding your preferences (Coffee shops in our scenario), and contain a valid offer for both of you.
Task5. Find how many places contain a valid offer for both of you.
Task6. Find how many places you have in common with your friend, regarding your preferences (Coffee shops in our scenario), but they contain a valid offer for one of you.

These tasks were performed in different areas on the map and had equal number of solutions so as the experiment to be fair. In detail, the map included:

1. 25 places (POIs)
2. 2 places as solution to task 1
3. 6 places as solution to task 2
4. 3 places as solution to task 3
5. 2 places as solution to task 4
6. 6 places as solution to task 5
7. 1 place as solution to task 6

Figure 17 below illustrates the average and median time that was required by the participants to complete each task concerning the two systems. It can easily be noticed that users using G-SPLIS performed better in all tasks, thus the system assisted users in completing the tasks notably sooner. In order to assess whether the differences in times are statistically significant, one-way ANOVA was conducted on the median times for completing each task. Median time was used to eliminate the effect of outliers. As it is displayed in Table 10 below, there is a statistically significant difference in median times because p value was found 0.00008, which is much less than 0.05 that the method requires [50].

Table 10. One-way ANOVA results

	Sum of squares	Mean square	F	p-significance
Between approaches	2333.567	1166.783	18.77629	0.00008

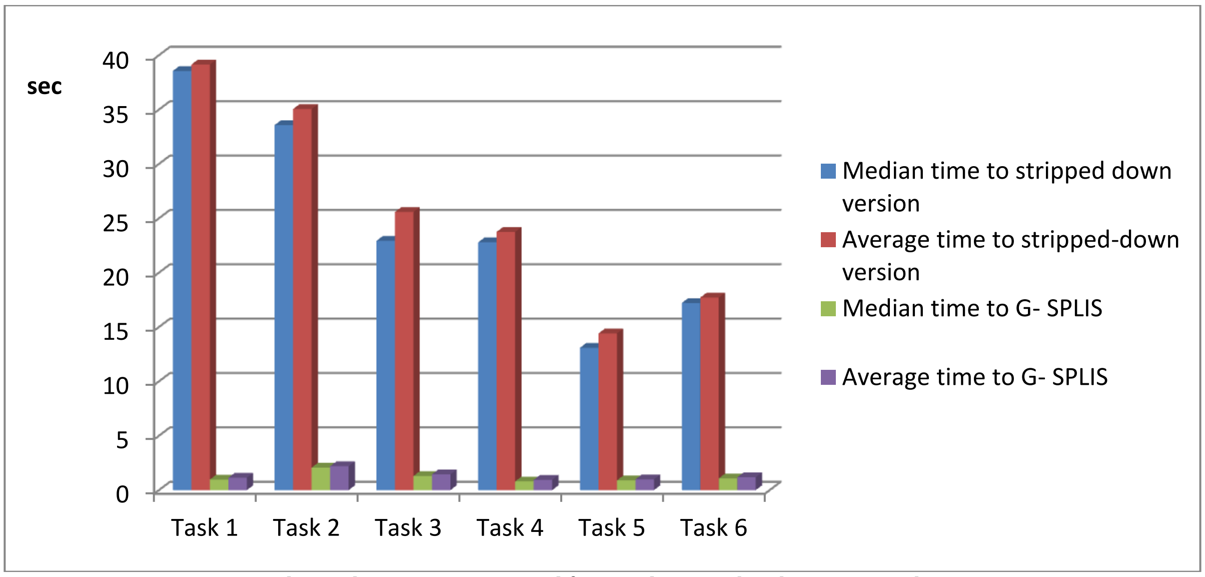


Figure 17. Median and average time required for completing each task, concerning the two systems.

7. Conclusions and Future Work

In this paper, an innovative knowledge-based geosocial networking service called G-SPLIS was presented. Our system:

1. Gathers data from external sources such as Google Places API, POIs' websites, Google+ and Facebook accounts.
2. Adopts a widely adopted ontology, such as schema.org, to represent persons' profiles, POIS and their associations.
3. By supporting a user friendly web-based editor, it offers users and POI owners the capability to author, maintain and deploy rules at run time.
4. Transforms these rules into RuleML format, in order to be interoperable with other systems in the web, and then into Jess rule engine so as to be machine executable.
5. Stores metadata and rules in the form of RDF triples (using the Sesame repository) for knowledge sharing and reusability.
6. Supports users with the capability to create social connections and interact among each other, as in other LBSNSs. They can also share and combine their preferences with those of their nearby friends and find interesting POIs and offers for all of them.
7. Displays personalized information on Google Maps.

The system proposes a novel methodology to model and formalize human daily preferences and POI owners' group targeted policies by providing users with the capability to add user-defined logical rules. This is the first time that logical rules are used in such way in a LBSN. Due to the fact that these rules can be shared among different systems, such user-defined policies can be easily reutilized and extended providing large knowledge bases in the web that can help in the personalization problem. Engaging non-technical users to generate content is a great challenge and previous Web 2.0 collaborative content creation platforms (e.g. YouTube, Twitter) and the huge commercial adoption of ifttt service prove that this is feasible.

During G-SPLIS's experimental evaluation, its technical potential became apparent. The proposed system can evolve without developer's intervention, as more and more users are registered. The more users add rules, the more interesting and intelligent it becomes as soon as there are rules for various contexts. Besides, it contains more flexible and effective rules, because of the fact that users are able to add or modify them according to their needs. To conclude, enabling non-technical users to add data and rules at run-time can help in addressing some of the limitations of knowledge-based systems such as:

1. The knowledge base is static and it is manually created by the developer, who is not able to foresee and define every possible situation.
2. Rule creation and modification is a time consuming process for the developers.
3. Developers' rules are not always effective for every user or every situation.
4. Many system components change from time to time during its usage (e.g. persons' profiles, POIs' properties), making the system's rule base insufficient and outdated.

All the above can help in providing information of higher quality (compared with traditional systems). For example, in our case:
POI owners enjoy a more efficient marketing strategy because they are able to customize their data and their target group (they can insert to the system their specific group-targeted offering policy), to send contextualized offers avoiding information overload and deploy their

marketing strategy into a group of socially connected people. On the other hand, regular users enjoy proactive contextualized information concerning POIs that match their preferences and also group-based information (by combining their preferences with those of their nearby friends and find who is interested in a specific POI and who has a valid offer), depending on their current situation.

Evaluation results verified G-SPLIS's reliability and usability. Response time evaluation results verified that the system is capable of handling a large number of concurrent user calls and can cope sufficiently with stretching situations. Regarding the qualitative evaluation, it showed a high degree of satisfaction among users. In the first two question groups, operations about a) rules and b) social processes, early over 80% of the participants were satisfied. In addition, over 90% of the participants stated that they will continue using the system or recommend it to their friends. Additionally, the quantitative evaluation verified system's usefulness by offering users the capability to implement the tasks significantly faster.

G-SPLIS's implementation can evolve in the future in various ways. One future direction will be the extension of Facebook integration by letting users to upload RuleML rules in their Facebook account (similarly with Google+), or the connection with other social media sources such as Foursquare, Twitter etc. It would be fascinating if social media could provide users with the capability to add rules in their account via a web editor. Combining such capability with the semantic interoperability will let users to customize their experience in the web. For example, an imaginary scenario would concern a user who logs into a LBS via his/her Facebook account and the service recommends to him/her some POIs to visit based on the rules that he/she possesses into Facebook. To the best of our knowledge, such functionality would also help in alleviating the cold start problem of recommendation systems in some situations [51]. Another possible research direction would be the adoption of a tagging system (by letting users to add tags) so as G-SPLIS to be able to support rules that take tags into consideration. For example, "If it is Saturday night I would like to go to a place that contains tag 'rock music' ". Furthermore, it would be interesting to expand the system's rule language and editor to express rules about groups of socially connected people and/or rules with extended expressiveness, e.g. using default negation, such as "if there is a student whose distance (from the POI) is d and there is no other student with distance d_1 closer to the POI ($d_1<d$), then make this offer to the student". In the future, we also plan to expand the web editor in order to support a wider range of human daily patterns (e.g. preferences about movies, videos etc.).

Another possible research direction could be the system's evolution in rule creation/generation in order to encourage more non-technical users. For example, one future direction could be the capability of the system to induce the rules automatically by mining users' behavior (e.g. previous check-ins, likes, comments, topics of interest etc.). Last but not least, such types of rules could be useful in other domains. For example, in advertising domain, by providing highly targeted marketing opportunities to the advertisers.

References

[1] Ilarri S., A. lllarramendi, E. Mena, A. Sheth. 2011. Semantics in Location-Based Services. IEEE Internet Computing (Vol. 15, No. 6) pp. 10-14.

[2] Roick O. S. Heuser. Location Based Social Networks – Definition, Current State of the Art and Research Agenda, 2013. Transactions in GIS. Volume 17, Issue 5, 763–784(2013)

[3] Zheng Y., Xie X. and Ma, W.Y. GeoLife: A Collaborative Social Networking Service among User, Location and Trajectory, IEEE Data Eng. Bull. 33 (2), 32-39, 2010

[4] Liu, X., &Aberer, K. (2013, May). SoCo: a social network aided context-aware recommender system. In Proceedings of the 22nd international conference on World Wide Web (pp. 781-802).International World Wide Web Conferences Steering Committee.

[5] Kjeldskov j., C. Christiensen and K. Rasmussen, "GeoHealth: a location-based service for home healthcare workers", Journal of Location Based Services, vol. 4, no. 1, Mar, 2010. pp. 3-27(2010)

[6] Hosseini-Pozveh M., M. Nematbakhsh, N. Movahhedinia. 2009. A multidimensional approach for context-aware recommendation in mobile commerce. (IJCSIS) International Journal of Computer Science and Information Security,Vol. 3, No. 1

[7] Mokbel M., Justin J. Levandoski. 2009. Towards context and preference-aware location-based services, Proceedings of the Eighth ACM International Workshop on Data Engineering for Wireless and Mobile Access.

[8] Murukannaiah P. and M. Singh,.Platys Social: Relating Shared Places and Private Social Circles. IEEE Internet Computing 2012.

[9] Tryfona N., D. Pfoser: Data Semantics in Location-based Services. Journal on Data Semantics III, vol. 3534, pp. 168–195 (2005)

[10] Bizer C., Tom Heath, and Tim Berners-Lee. Linked data - the story so far. International Journal on Semantic Web and Information Systems, 5(3):1–22, (2009)

[11] Taehee Lee, Ig-hoon Lee, Suekyung Lee, Sang-goo Lee, Dongkyu Kim, Jonghoon Chun, Hyunja Lee and Junho Shim. Building an operational product ontology system. Electronic Commerce Research and Applications, vol. 5, Issue 1, pp. 16-28 (2005)

[12] Weigand H., W. J. Heuvel, and M. Hiel, Rule-based service composition and service-oriented business rule management. ReMoD, France (2008)

[13] Patkos T, Bikakis A, Antoniou G, Plexousakis D, Papadopouli M. 2007.A semantics-based framework for context-aware services: lessons learned and challenges. In: Proceedings of 4th international conference on Ubiquitous intelligence and computing (UIC-2007), Vol. 4611 of LNCS, Springer, pp 839–848(2007)

[14] Giurca A., M. Tylkowski, M. Muller. RuleTheWeb!: Rule-based Adaptive User Experience. 2012. Proceedings of the RuleML2012@ECAI Challenge, at the 6th International Symposium on Rules, Montpellier, France, August 27th-29th, 2012, CEUR Workshop Proceedings, Vol-874 (2012)

[15] Motik Boris, Grau Bernardo Cuenca, Horrocks Ian, Wu Zhe, FokoueAchille, Lutz Carsten, (Eds.), "OWL 2 Web Ontology Language Profiles", 2nd Edition, W3C Recommendation 11 December 2012, http://www.w3.org/TR/owl2-profiles

[16] Horrocks Ian, Patel-Schneider Peter F., Boley Harold, Tabet Said, Grosof Benjamin, Dean Mike, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", W3C Member Submission 21 May 2004, http://www.w3.org/Submission/SWRL/

[17] Viktoratos I., Tsadiras A., Bassiliades N.:"Using Rules to Develop a Personalized and Social Location Information System for the Semantic Web".RuleML 2014: 82-96

[18] Viktoratos I., Tsadiras A., Bassiliades N.:"Providing a context-aware location based web service through semantics and user-defined rules".WIMS 2014: 9

[19] Viktoratos I., Tsadiras A., Bassiliades N. 2013.A Rule Based Personalized Location Information System for the Semantic Web.14th International Conference, EC-Web 2013, Prague, Czech Republic, August 27-28. Proceedings. pp 27-38 (2013)

[20] Ye M. et al. On the semantic annotation of places in location-based social networks. In Proc. 17th Int. Conf. ACM SIGKDD Knowledge Discovery and Data Mining,520–528 (2011)

[21] Ciaramella, A., Cimino, M.G., Lazzerini, B., Marcelloni, F. 2009. Situation-Aware Mobile Service Recommendation with Fuzzy Logic and Semantic Web. NinthInt. Conference on Intelligent Systems Design and Applications, IEEE

[22] García-Crespo, Á., López-Cuadrado, J.L., Colomo-Palacios, R., González Carrasco, I., Ruiz-Mezcua, B. 2011. Sem-Fit: A semantic based expert system to provide recommendations in the tourism domain. Expert systems with applications, 38.

[23] Keßler C. 2010, A RESTful SWRL Rule Editor. RR, volume 6333 of Lecture Notes in Computer Science, pp. 235-238.

[24] Niforatos E.,Karapanos E.,Sioutas S. 2012. PLBSD: a platform for proactive location-based service discovery. Location Based Services Journal, vol. 6.

[25] Armenatzoglou et al. FleXConf: A Flexible Conference Assistant Using Context-Aware Notification Services. 2009.On the Move Confederated International Conference and Workshops pp. 108-117.

[26] Her Y., S.Kim, Y.Jin. A Context-Aware Framework using Ontology for Smart Phone Platform International Journal of Digital Content Technology and its Applications Vol. 4(5), August, (2010).

[27] Fuchs S., S. Rass, B. Lamprecht, K. Kyamakya. A Model for OntologyBased Scene Description for Context-Aware Driver Assistance Systems, ACM SIGCHI, ICST Canada, (2008)

[28] Serrano D., R. Hervás, J. Bravo. 2011. Telemaco: Context-aware System for Tourism Guiding based on Web 3.0 Technology. International Workshop On Contextual Computing and Ambient Intelligence in Tourism.

[29] Croitoru A., A. Crooks, J. Radzikowski, A. Stefanidis 2013 Geosocial gauge: a system prototype for knowledge discovery from social media. International Journal of Geographical Information Science Vol. 27, Iss. 12.

[30] Viana W., J. B. Filho, J. Gensel, M. Villanova-Oliver, H. Martin. 2008. PhotoMap: From location and time to context-aware photo annotations. In Journal of Location Based Services 2008 vol. 2(3), p.p. 211-235.

[31] Juan Li , Hui Wang , SameeUllah Khan. 2012. A Semantics-based Approach to Large-Scale Mobile Social Networking. Mobile Networks and Applications, v.17 (2),192-205.

[32] Woensel V., S. Casteleyn and O. Troyer.Applying semantic web technology in a mobile setting: "the person matcher," Web Engineering, pp. 506- 509(2010)

[33] Cho E. , Myers S. A. , Leskovec J., Friendship and mobility: user movement in location-based social networks, Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining, August 21-24, 2011, San Diego, California, USA

[34] JieBao , Yu Zheng , Mohamed F. Mokbel, Location-based and preference-aware recommendation using sparse geo-social networking data, Proceedings of the 20th International Conference on Advances in Geographic Information Systems, November 06-09, 2012, Redondo Beach, California

[35] Yung-Ming Li , Chia-Ling Chou, Lien-Fa Lin. A social recommender mechanism for location-based group commerce. Information Sciences 274 (2014) 125–142

[36] Broekstra J., Kampman A., and van Harmelen F. 2001. Sesame: An architecture for storing and querying RDF data and schema information. In H. Lieberman D. Fensel, J. Hendler and W. Wahlster, editors, Semantics for the WWW. MIT Press.

[37] Yuh-Jong Hu, Ching-Long Y., L. Wolfgang. Challenges for Rule Systems on the Web. RuleML '09 Proceedings of the 2009 International Symposium on Rule Interchange and Applications (2009)

[38] Paschke A., Kozlenkov A., H. Boley. A Homogenous Reaction Rule Language for Complex Event Processing, 2nd International Workshop on Event Drive Architecture and Event Processing Systems (EDA-PS 2007), Vienna, Austria, (2007)

[39] E. Kontopoulos, N. Bassiliades, and G. Antoniou. DeployingDefeasible Logic Rule Bases for the Semantic Web. 2008. Data and Knowedge. Engineering,66(1),116–146.

[40] Grosof, B.: Representing E-Commerce Rules Via Situated Courteous Logic Programs in RuleML. Electronic Commerce Research and Applications (2003)

[41] Boley H., S. Tabet and G.Wagner. Design Rationale of RuleML: A Markup Language for Semantic Web Rules, Proc. SWWS'01, Stanford, (July/August 2001)

[42] Christian de Sainte Marie, Gary Hallmark, Adrian Paschke. RIF Production Rule Dialect (2nd Edition), W3C Recommendation 5 February 2013, http://www.w3.org/TR/rif-prd/

[43] Liang S., FodorP.,Wan H., Kifer M. OpenRuleBench:An Analysis of the Performance of Rule Engines. WWW 2009 Madrid (2009)

[44] Ernest Friedman-Hill: Jess in Action.Rule-Based Systems in Java. Manning Publications. ISBN-10: 1930110898, pp. 32-33(2003)

[45] Sherman G. A Critical Analysis of XSLT Technology for XML Transformation.Senior Technical Report. (2007)

[46] Duane K. FieldsMark A. KolbShawn Bayern. Web Development with Java Server Pages. Manning Publications ISBN:193011012X(2001)

[47] Ipina D. and Katsiri E. An ECA Rule-Matching Service for Simpler Development of Reactive Applications. *Published as a supplement to the Proc. of Middleware 2001 at IEEE Distributed Systems Online*, vol. 2, no. 7, (2001)

[48] Charles L. Forgy. 1991. Rete: a fast algorithm for the many pattern/many object pattern match problem. In Expert systems, Peter G. Raeth (Ed.). IEEE Computer Society Press, Los Alamitos, CA, USA 324-341 (1991)

[49] Nielsen, J., 1993, Response times: the three important limits. Excerpt from Chapter 5 of Usability Engineering by Jakob Nielsen, Academic Press, 1993

[50] Kulinskayaa E, Dollingerb MB (2007) Robust weighted one-way ANOVA: improved approximation and efficiency. J Stat Plan Inference 137:462–472

[51] Lika B.,Kolomvatsos K., Hadjiefthymiades S. Facing the cold start problem in recommender systems, Expert Systems with Applications, Vol 41, Issue 4, Part 2, pp. 2065-2073, March 2014.

Appendix A

Table 11. Formulation regarding some of G-SPLIS processes.

In the following, "*sch*" stands for the namespace of the schema.org ontology.

Set of POIs: $POI = \{ p(i) \mid p(i) \in \text{EXT}(sch{:}Place) \wedge p(i).loc \propto u_{cur}.window \wedge i \in [1..\text{POImax}]\}$, where:

- POImax is a system defined parameter (e.g. in the desktop version of the system is set to 40),
- EXT(*Class*) is the extension of a class, namely the set of its direct and indirect instances,
- $p(i).loc$ is the location (coordinates) of the POI, $u_{cur}.window$ is the map window currently visible to the currently logged-in user u_{cur}, and
- operation $\propto$ means that the POI is located within the boundaries of the currently visible map window.

Each POI $p(i)$ has a set of associated properties: $p(i).Prop^{POI} = \{ p(i).property(k) \mid sch{:}Place \sqsupseteq \text{domain}(property(k)) \sqsupseteq Class(p(i)) \}$, i.e. the set of properties that have as their domain the class of $p(i)(Class(p(i)))$ or all of the direct and indirect superclasses of $Class(p(i))$ up to class *sch*:*Place*.

Set of Users: $U = \{ u(j) \mid j \in [1..M]\}$, where M is the number of registered users.

Each user $u(j)$ has a set of associated properties: $u(j).Prop^{u} = \{ u(j).property(l) \mid sch{:}Person \sqsupseteq \text{domain}(property(l)) \sqsupseteq Class(u(j)) \}$, i.e. the set of properties that have as their domain $Class(u(j))$ or all of its direct and indirect superclasses up to class *sch*:*Person*.

System context: $Cont^{sys}(t) = \{ date(t), t, weather(t), location(t)\}$, t is a point in time
User context: $Cont^{usr}(x) = u(x).Prop^{u}$, where x is a user of the system
Place (POI) context: $Cont^{pl}(i) = p(i).Prop^{POI}$
Set of marker Colors: $C = \{ green, yellow, red \}$
Set of marker Sizes: $S = \{ big, small \}$

Ruleset of offers for a single POI $p(i)$: $p(i).R^{o} = R^{o}(i) = \{ r^{o}(i,k) \mid r^{o}(i,k) = \langle Cond^{o}(i,k) \rightarrow action^{o}(i,k) \otimes salience(i,k) \rangle \}$, where:

- $Cond^{o}(i,k) = \{ cond^{o}(i,k,n) \mid cond^{o}(i,k,n) = \langle pred^{o}(i,k,n)\ op(i,k,n)\ val(i,k,n)\rangle \wedge pred^{o}(i,k,n) \in (Cont^{sys}(t) \setminus \{location(t), date(t)\}) \cup \{dist(p(i).loc, location(t)), day(date(t))\} \cup (Cont^{usr}(u) \setminus \{u.recommend\}) \wedge op(i,k,n) \in Ops \wedge val(i,k,n) \in Const \}$, where:
  - ○ *Const* is set of all literals / constants used.
  - ○ $Ops = \{=,>,<\}$ is the set of built-in comparison operators. Notice that the "=" operator appears as "is" at the system GUI.
  - ○ *dist* is a function that returns the distance between the location of the POI $p(i)$ (namely $p(i).loc$) and the logged user's current location $location(t)$, which is derived from the system context.
  - ○ *day* is a function that returns the day of a date.
- $action^{o}(i,k) = \langle pred^{o}(i,k)\ val(i,k)\rangle \wedge pred^{o}(i,k) \in Cont^{pl}(i) \wedge val(i,k) \in Const$
- $salience(i,k)$ is an integer value between -10000 and 10000 that indicates the rule priority in case of conflicts. The higher the number the more important the rule; therefore, its action effect prevails over conflicting rules. See below on the rule execution.

Ruleset of POI offers: $R^{o} = \bigcup_{p(i)\in POI} R^{o}(i)$

Ruleset of preferences for a single user $u(j)$: $u(j).R^{pr} = R^{pr}(j) = \{ r^{pr}(j,m) \mid r^{pr}(j,m) = \langle Cond^{pr}(j,m) \rightarrow action^{pr}(j,m) \rangle \}$, where:

- $Cond^{pr}(j,m) = \{\langle p(i).type = val(j,m)\rangle \mid p(i) \in POI \wedge val(j,m) \in POITypes)\} \cup$
$\{ cond^{pr}(j,m,n) \mid cond^{pr}(j,m,n) = \langle pred^{pr}(j,m,n)\ op(j,m,n)\ val(j,m,n)\rangle \wedge pred^{pr}(j,m,n) \in Cont^{pl}(i) \cup Cont^{sys}(t) \setminus \{location(t), date(t)\} \cup \{dist(p(i).loc, location(t)), day(date(t))\} \wedge p(i) \in POI \wedge op(j,m,n) \in Ops \wedge val(j,m,n) \in Const \}$, where:
  - ○ $p(i).type$ is the value of the type property of the $p(i)$ POI, and
  - ○ *POITypes* is the set of all different types of POIs that G-SPLIS supports: $POITypes = \bigcup_{p(i)\in POI} \{p(i).type\}$
- $action^{pr}(j,m) = \langle u(j).recommend\, p(i)\rangle$, where *recommend* is a G-SPLIS property of class *sch*:*Person*.

Ruleset of users' preferences: $R^{pr} = \bigcup_{u(j)\in U} R^{pr}(j)$

Offerings rules fire when their condition *Cond* is satisfied for a user context (of a user u) and a system context (for time point t), $Sat(Cond,u,t)$ and then the action is executed, namely the value of the property that the action denotes changes.

$Sat(Cond^{o}(i,k),u,t)$ iff $\forall cond^{o}(i,k,n) \in Cond^{o}(i,k) \rightarrow Sat(cond^{o}(i,k,n),u,t)$

$$Sat\left(cond^{o}(i,k,n),u,t\right) \leftarrow eval\left(\left\langle pred^{o}\left(i,k,n\right) op\left(i,k,n\right) val\left(i,k,n\right)\right\rangle\right) = True$$

Notice that rule execution of oferring rules follows the order indicated by rule salience. Namely, the rules with the highest salience are executed last so that their effect prevails / overrides the effects of previously executed rules.

$Exec(r^{o}(i,k))$ if $Sat(Cond^{o}(i,k),u_{cur},t) \wedge \neg(\exists\ r^{o}(i,m) \in R^{o}(i) \wedge salience(i,m) < salience(i,k) \wedge executed(r^{o}(i,m)) = false)$, where:

- *Exec* is the procedure that applies the effect of the action of a rule, and
- *executed* is a rule property that is equal to *false* before the rule is executed and gets equal to *true* when the rule is executed.

Preference rules fire when their condition *Cond* is satisfied for a POI context (of a POI p) and a system context (for time point t), $Sat(Cond,p,t)$ and then the action is executed, namely the value of the property that the action denotes changes.

$Sat(Cond^{pr}(j,m),p,t)$ if $(\forall cond^{pr}(j,m,n) \in Cond^{pr}(j,m) \rightarrow Sat(cond^{pr}(j,m,n),t)) \wedge p.type = val(j,m)$

$$Sat\left(cond^{pr}(j,n,m),t\right) \leftarrow eval\left(\left\langle pred^{pr}\left(j,m,n\right) op\left(j,m,n\right) val\left(j,m,n\right)\right\rangle\right) = True$$

Notice that rule execution of user preference rules does not have to follow a specified order; thus the absence of the rule salience.

Presentation of personalized information

- A bigger in size marker illustrates a POI which matches user preferences and it is recommended.
- A red marker represents a POI with no offers at all.
- A green marker is used for a place which contains a rule base of targeted offers and at least one rule has been fired for the specific user depending on his/her current context.
- A yellow marker indicates that the place contains a rule base of targeted offers but no rule is fired for the current user.

Assignment of marker color to POIs for a system context in time point t: $Color$: $POI \times T \rightarrow C$

$$Color\left(p(i),t\right) = \begin{cases} green, \text{ if } \exists r^{o}(i,k) \in R^{o}(i) \wedge Sat\left(Cond^{o}(i,k),u_{cur},t\right) \\ yellow, \text{ if } \exists r^{o}(i,m) \in R^{o}(i) \wedge \left(\forall r^{o}(i,k) \in R^{o}(i) \rightarrow \neg Sat\left(Cond^{o}(i,k),u_{cur},t\right)\right) \\ red, \text{ if } \neg\exists r^{o}(i,k) \in R^{o}(i) \end{cases},$$

where u_{cur} is the current logged-in user.

Assignment of marker size to POIs for a system context in time point t: $Size$: $POI \times T \rightarrow S$

$$Size\left(p(i),t\right) = \begin{cases} big, \text{ if } \exists r^{pr}(cur,k) \in R^{pr}(cur) \wedge Sat\left(Cond^{pr}(cur,k), p(i), \text{t}\right) \\ small, \text{ otherwise} \end{cases}$$

where cur is the index of the currently logged-in user u_{cur}.

For the normal operation the following functions are evaluated to assign color and size to POIs: $Color(p(i),now)$, $Size(p(i),now)$, where now is the current time point.
For the planning mode the system feeds a different spatio-temporal context for a time point f (future) designated by the user to the same functions: $Color(p(i),f)$, $Size(p(i),f)$.

Nearby friends' mode process

a) Collects a) user's rules, b) his nearby (logged in) friends' rules and c) contextual information.
b) Evaluates all the above rules (a) and (b) and fetches the nearby POIs which are recommended by the users' fired rules.
c) For these POIs, it gets their group targeted offers (POIs' rules), and evaluates them concerning all users contexts (the user and his/her friends).
d) Provides personalized information by displaying:
e) With a red marker a POI that does not have any offer at all.
f) With a yellow marker a POI that has at least one offer, but none of them is valid for any of the friends or the user at that moment.
g) With a half yellow-half green marker a POI which has a valid offer for at least one of the friends or the user.
h) With a green marker a POI which has a valid offer for all of the friends and the user.
i) With a bigger marker a POI that is recommended by a user rule and at least one of his/her friends' rules (common POIs).

Each user has the following associated properties:
$u(i).logged$ (datatype property, Boolean)
$u(i).friend$ (object property, which points to the OID of a friend)
$u(i).loc$ (datatype property, a pair of geographical coordinates where the user $u(i)$ is currently located)
$dist(u(i).loc, u(j).loc)$ is a function that returns the distance between the current locations of the users $u(i)$ and $u(j)$.

Let $NBF(i)$ be the currently logged-in nearby friends of user $u(i)$.
$NBF(i) = \{\, u(i).friend \mid u(i).friend.logged = true \wedge dist(u(i).loc, u(i).friend.loc) \leq thr \,\}$, where thr is a system-defined threshold.

Ruleset of preferences for nearby friends: $R^{pr}\left(NBF(i)\right) = \bigcup_{u(j) \in NBF(i)} R_{j}^{pr}(j)$

Assignment of marker color to POIs for a system context in time point t: $Color$: $POI \times T \rightarrow C$

$$Color\left(p(i),t\right)=\begin{cases} green, \text{ if } \exists r^{o}(i,k)\in R^{o}(i)\wedge\left(\forall u\in\left\{u_{cur}\right\}\cup NBF(cur)\rightarrow Sat\left(Cond^{o}(i,k),u,t\right)\right) \\ yellow-green, \text{ if } \exists r^{o}(i,k)\in R^{o}(i)\wedge\exists u\in NBF(cur)\wedge Sat\left(Cond^{o}(i,k),u,t\right) \\ yellow, \text{ if } \exists r^{o}(i,m)\in R^{o}(i)\wedge\left(\forall r^{o}(i,k)\in R^{o}(i),\forall u\in\left\{u_{cur}\right\}\cup NBF(cur)\rightarrow\neg Sat\left(Cond^{o}(i,k),u,t\right)\right) \\ red, \text{ if } \neg\exists r^{o}(i,k)\in R^{o}(i) \end{cases}$$

where u_{cur} is the currently logged-in user.

Assignment of marker size to POIs for a system context in time point t: $Size$: $POI \times T\rightarrow S$

$$Size\left(p(i),t\right)=\begin{cases} big, \text{ if } \exists r^{pr}(cur,k)\in R^{pr}(cur)\wedge Sat\left(Cond^{pr}(cur,k),p(i),\text{t}\right)\wedge \\ \exists u\in NBF(cur)\wedge\exists r^{pr}(u,m)\in R^{pr}(u)\wedge Sat\left(Cond^{pr}(u,m),p(i),\text{t}\right) \\ small, \text{otherwise} \end{cases}$$

For the normal nearby friends' mode operation the following functions are evaluated to assign color and size to POIs: *Color*(*p*(*i*),*now*), *Size*(*p*(*i*),*now*), where *now* is the current time point.

Appendix B

Table 12. XSLT file for generating an explanation message from the RuleML file.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="2.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:fn="http://www.w3.org/2005/xpath-functions">
<xsl:output method="text" version="1.0" encoding="UTF-8" indent="yes"/>
<xsl:template match="//Rule">
If <xsl:apply-templates select="if//Atom[Rel='person']" mode="person"/>, then I would like a <xsl:apply-templates
select="if//Atom[Rel='place']" mode="place"/>
<xsl:if test="if//Expr">, whose <xsl:apply-templates  select="if//Expr"/></xsl:if>
</xsl:template>
<xsl:template match="Atom" mode="person">
<xsl:for-each select="slot">
<xsl:if test="position()>1 and not(Var)"> and </xsl:if>
<xsl:if test="Ind[2]"><xsl:value-of select="Ind[1]"/> is <xsl:value-of select="Ind[2]"/></xsl:if>
</xsl:for-each>
</xsl:template>
<xsl:template match="Atom" mode="place">
<xsl:value-of select="slot[Ind[1]='type']/Ind[2]"/>
</xsl:template>
<xsl:template match="slot">
<xsl:value-of select="Ind[1]"/><xsl:if test="Ind[2]"> is <xsl:value-of select="Ind[2]"/></xsl:if><xsl:if
test="position()!=./last()"> and </xsl:if>
</xsl:template>
<xsl:template match="Expr">
<xsl:variable name="var" select="Var"/>
<xsl:value-of select="../Atom[Rel='person']/slot[Var=$var]/Ind"/> is
<xsl:choose>
<xsl:when test="Fun='&lt;'"> less than </xsl:when>
<xsl:when test="Fun='&gt;'"> more than </xsl:when>
<xsl:otherwise><xsl:value-of  select="Fun"/></xsl:otherwise>
</xsl:choose>
<xsl:value-of select="Ind"/>
</xsl:template>
</xsl:stylesheet>
```