Abstract—The convergence of Internet and mobile devices has radically changed the way people communicate and interact with each other, and demand for applications that are "social" enough to assist their daily interactions. To support such device mediated interactions, the social relationships between actors need to be systematically modeled and represented. In addition, an application facilitating such interactions should be able to deal with the task conflicts that occur when an actor is involved in multiple interactions simultaneously. To address these issues, in this paper we present an approach to modeling and coordinating social interactions with the notion of social context. It supports social interaction modeling from both the domain- and player-centric perspectives. In particular, the player-centric model provides the basis to coordinate multiple interactions in which an actor is involved. We further introduce a fuzzy logic based reasoning technique to infer the overall importance of each interaction, assisting the actor to resolve conflicts and make decisions. Finally, we validate our approach through a prototype implementation and test cases analysis.

Keywords—social interaction; social context; social awareness; role based modeling; coordination; fuzzy inference

I. INTRODUCTION

Social interactions are an essential part of our daily activities, be it for personal or business purposes. They are responsible for our well-being and for a productive working or business environment. For example, it has been estimated that globally the number of elderly is expected to exceed two billion by 2050 [1], and among the most beneficial services for elderly in the future are those by which they can maintain their social interactions [2]. The proliferation of mobile devices and ubiquitous computing technologies over the past decade, such as smart phones/PDAs, RFID and 3G networks has radically changed the way people communicate and interact with each other. To truly support human interactions, however, these ubiquitous devices and technologies have to be equipped with software applications that are social in understanding and helping the users and in coordinating, competing and collaborating with each other [3].

To provide dynamic support for social interactions, the software applications need to be aware of not only the user’s physical context but also its social context [4]. The former refers to contextual attributes such as where the user is, what the user is doing, and so on; whereas the latter concerns the social relationships between the user and other actors that influence an individual’s action with respect to a particular situation. For example, using mobile phone a person can coordinate his Son role to take care of an elderly mother staying at home alone when he goes to work in his office to play his Employee role. However, the involvement of an individual in multiple concurrent interactions might incur conflict as they may exceed the individual’s capability. In the above example, the son may be required to go home due to the mother’s sudden illness while being scheduled to be at a business meeting. This collision in roles represents the time-based conflict as the acting of one role prevents the playing of another [5]. This type of conflict also arises in business domains. In supply-chain, for example, two customers may want to place orders with a similar due dates. Accepting both orders may result in conflict in production schedule as the plant’s capacity would be exceeded. Therefore, a software application supporting social interactions should be able to help the user in deciding which interaction they need to respond by considering the relative importance of the competing interactions, and possibly carry out certain interactions on behalf of the user based on the user’s contextual information and preferences (e.g., calling an ambulance when a person is busy and his/her mother is having a heart-attack).

In providing such socially aware software applications, it is important to (1) have a systematic approach to model social relationships and (2) facilitate coordination (i.e. help to resolve the conflicts among an individual’s roles) to enable interactions [6]. While developments in the area of social networks have been dramatic in recent years, they primarily focus on social groups rather than personalised social interactions between individuals and their coordination and management. Much research has been carried out in context modeling and context-aware systems, focusing on physical contexts. However, there has been very limited effort in modeling social contexts to support social interactions. Similarly, there has also been extensive research in personal digital assistants (e.g., [7], [8]), social-mobile applications (e.g., [9], [10]) and multi-agent systems (e.g., [11], [12]), but very limited work has focused on social awareness. In particular, none of the work explicitly addresses modeling social interactions with coordination functionality to resolve task conflicts.

To address the above research issues, in this paper we introduce an approach to model and represent social interactions and coordinate them to resolve task conflicts during multiple concurrent interactions. We model social
interaction from both the domain and player (actor) perspectives with the notion of social context. The domain-centric modeling captures the interactions associated with a particular domain or environment such as home or office, while the player-centric perspective provides an overall view of all the interactions of an individual with personalised preferences and allows coordination among his/her interactions while taking into account the relevant physical and social contexts. In relation to the player-centric perspective, we further introduce a coordination technique to resolve task conflicts in an individual’s interactions. In particular, we address time-based conflict and propose a reasoning technique to infer the overall importance of each task in conflict based on its priority, preference and consequence, to aid task selection.

The paper is organized as follows. Section 2 presents a motivating scenario. Section 3 presents our approach to social interaction modeling from both the domain and player perspectives. A coordination technique for resolving task conflicts is introduced in section 4. Section 5 presents the system architecture and modeling tool for context-aware systems supporting social interactions, and a prototype implementation of such system. After some further discussion in section 6, section 7 reviews related research. Section 8 concludes the paper and highlights future work.

II. MOTIVATING SCENARIO

This section motivates the research by showing situations where adaptive behavior (scene#1) and coordination functionality (scene#2) facilitate interactions between people. The scenario begins with Mary, the elderly mother of John, who had recently been hospitalized due to heart attack. Although she had been discharged, the doctor recommends continuous monitoring of her condition. Upon return to her smart home with body sensor installed, her son, John acts as her principal caregiver. John needs an application that allows him to track the situations of his elderly mother staying at home alone when he goes to work in his office. Assume that a socially-aware application is already installed on John’s and Mary’s smart phone. The Mary’s application continuously communicates with the sensors to predict her situations and sends alert messages to John if any serious event happens. John’s application helps him to take decision in case of multiple requests from different actors (e.g., boss, mother) by quantifying the overall importance of each requested task based on his own preference, requester’s priority, and the consequence of rejecting that task. Moreover, the application can initiate actions depending on the seriousness of the events happening to his mother and with respect to his physical context and activities, such as where he is and what he is doing.

Scene #1: It happens that Mary falls down at home and the situation is so serious that she is unable to move and her pulse rate becomes abnormal. Mary’s application detects that event and first tries to contact John. But unfortunately John cannot respond as he is busy giving a seminar. As a consequence, John’s application directly contacts the emergency service and calls ambulance to bring his mother to hospital\(^1\) or the Mary’s application may call ambulance first. The ambulance picks Mary up and transports her to the hospital and informs John to the hospital address.

Scene #2: Another day John’s application is informed that his mother has asked him to come home as she is not feeling well. Apart of that information his application also reminds his scheduled meeting in 30 minutes. Considering the overall importance of each of those tasks the application assists him to take decision: John advised to send his apology and postpone the meeting and accept the task to go home.

In this scenario, John is playing a Son role in home domain, and an Employee role in office domain. When an actor plays multiple roles conflict might occur. In our scenario, John’s mother request to come home, conflicts with his boss order to attend a meeting. This type of conflict is well acknowledged in the social psychology literature [5] as well.

III. ROLE-BASED SOCIAL INTERACTION MODELING

According to the sociologist Max Weber, “an action is social if the acting individual takes account of the behavior of others and is thereby oriented in its course” [13]. Social interaction refers to the sequence of social action between individuals (or groups) by considering their relationships. The requirement of social interaction modeling poses two main challenges: First, it requires an understanding of constructed relationships, obligations, and constraints underlying the interactions among actors in an environment/domain. Here, we model this from domain perspective which also can be modeled from subjective [4] point of view. Second, an actor might operate in different domains simultaneously, which demands coordination functionality to manage his/her interactions. Each actor might have its own coordinated view of its social contexts that reflects the actor’s preferences in these interactions. To support this view, we introduce player-centric modeling which is based on the Role Oriented Adaptive Design (ROAD) [14] framework. The main advantages of ROAD framework are that it provides management, adaptation, ease of deployment, and particularly very useful to model the notion of interactions [4].

A. Domain-centric Modeling

A domain-centric social context model (DSCM) is a self managed composite, and is made of a collection of social relationships (defined as contracts) between social roles. The social roles represent the expected behaviors (functional interactions) with respect to a context model that are played by the corresponding players. For example, in the home social context (Fig. 1), we have three social

\(^1\) A survey found that more than two-thirds of Australians would not call an ambulance if they thought they were having a heart attack (http://www.heartfoundation.org.au) which is also supported by international studies.
roles Son, ElderlyMother, and EmergencyService, with three contacts (C1, C2 and C3) between the social roles. These social roles are respectively played by John, Mary, and Ambulance service. A social relationship or contract is an association between two social roles which mediates the interaction by defining what functional interactions (interaction clauses) can occur between the social roles. In addition, a contract also defines the non-functional requirements of the interactions in terms of Conversational and Obligation clauses that are imposed on the players associated with that contract. In addition, the social context has an organizer role which is responsible for dynamically managing the social context’s topology, regulating the social context by creating and changing roles, contracts, and binding players to social roles.

Interaction clauses are permitted atomic message exchanges between the contracted parties. A clause’s definition includes an identifying message signature, a direction of the message (i.e., AtoB, BtoA or either) and the message exchange pattern (i.e., one-way or request-response). The reader is referred to [4] for more detailed descriptions. In Table I, for example, i2 in C1 contract specifies a request-response interaction that involves B (Son role) sending a situationAsking message (e.g., John enquires his mother’s situation) to A (Mother role), and A sending a situationAnswering message (e.g., Well, NotFellingGood, HeartAttack, FallDown, etc) to B. These interaction clauses also define the required functionality of each party involved. For example, party B is required to have capability to process all messages directed to B which are the contactSon and askToReturnHome. Conversation clauses are acceptable sequences of interactions defined by temporal constraints in the Interaction Rule Specification (IRS) language [15]. For example, c1 in C1 specifies that the message contactSon must be received before (i.e., precedes) the askToReturnHome message. Obligation defines the time constraints in interactions. For example, o3 in C1 states that the son must check his mother’s health condition periodically by every 30 minutes. o2 states that after receiving the situation query message, the mother should respond in 40 seconds.

### Table I. Two contracts associated with Son role

<table>
<thead>
<tr>
<th>Contract ID</th>
<th>Parties: A: Mother; B: Son; Interaction Clauses:</th>
<th>Obligations:</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1: ElderlyMother_Son</td>
<td>i1: {contactMother, BtoA, motherResponse}</td>
<td>o1: [i3, Timer, duration, &lt;, 30, seconds];</td>
</tr>
<tr>
<td></td>
<td>i2: {situationAsking, BtoA, situationAnswering}</td>
<td>o2: [i2, Timer, duration, &lt;, 40, seconds];</td>
</tr>
<tr>
<td></td>
<td>i3: {contactSon, AtoB, sonResponse}</td>
<td>o3: [i2, Timer, periodic, =, 30, minutes];</td>
</tr>
<tr>
<td>C2: Relative_EmergencyService</td>
<td>i6: {callAmbulance, BtoC, askAddress}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>i7: {informAddress, BtoC, sendAmbulance}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>i8: {notifyPickUp, CtoB}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>i9: {notifyHospitalAddress, CtoB}</td>
<td></td>
</tr>
</tbody>
</table>

### B. Player-centric Modeling

One of the key principles and novelty of our modeling technique is the player-centric perspective of social context modeling that allows an actor to coordinate his/her interactions. Each player-centric composite, which is based on the meta-model shown in Fig. 2, represents the actor’s personal social context model (PSCM). It comprises of five key elements: social role, coordinator role, role-centric contract, actor, and organizer role. Except role-centric contract and coordinator role all these elements are similar as in domain-centric modeling. Like DSCM, the PSCM (e.g., John’s Social Context in Fig. 1) is also a self-managed composite with all social roles are ‘personas’ played by the actor in question and are connected with a coordinator role through role-centric
contracts. Every incoming and outgoing messages to and from the personal social context composite are passed through the associated role-centric contract and intercepted by the coordinator role which is played by the actor’s socially-aware application (a kind of actor). This application coordinates the requested task(s) (we use the terms task and message interchangeable throughout this paper) and ultimately passes to the actor as appropriate. The organizer role of the composite provides the management functionality which is played by the owner (actor) of that context. The role-centric contract is the aggregation of all the contracts associated with a particular role in a domain-centric model. Instead of John, his PSCM plays the Son and Employee roles in the domain social contexts (see Fig. 1). Those roles should also be incorporated in John’s PSCM and are connected with coordinator role by C7 and C8 role-centric contacts. C7 (see Table II) is the son’s role-centric contract which is the aggregation of C1 and C2 in the domain-centric home composite. Similarly, we have employee’s role-centric contract (C8) which is the aggregation of C4 and C5 in office composite.

The role-centric contract represents the all functional and non-functional requirements specified in the contract(s) associated with that role in the domain-centric model. In addition, role-centric contract may contain context rules by which an actor can specify his/her personal preferences and the adaptive behaviour. For example, rules r1 and r2 in Table II respectively specifies a John’s personal preference to go home considering his mother’s situation and a social context fact (here conflict status), and an adaptive behaviour by considering his situation and his mother’s physical context facts (e.g., a heart monitor providing information on mum’s health). These physical context facts can be acquired from any fact providers where the social context fact (e.g., sConflict in Table II) is maintained by John’s application itself. Rule r2 initiates the emergency procedure by calling an ambulance when John is busy (e.g., Meeting) and his mother is having a heart-attack.

IV. CoORDINATION TECHNIQUE TO RESOLVE CONFLICT

John’s association to Son and Employee roles may incur conflict, scene#2 in section 2 shows an instance. Role theory suggests that conflict occurs when individuals engage in multiple roles that are incompatible [16]. It is a special form of social conflict that takes place when one is forced to take on two or more different and incompatible roles at the same time. Social psychology research examines different types of conflicts such as time-based, behavior-based, strain-based, and so on [5]. In this paper, our particular concern is to resolve time-based conflict. Here we assume that the conflicted tasks are not reshedulable, so the actor needs to take decision what to accept in order to continue.

In general, coordination can be defined as “the act of managing interdependencies between activities”. In our problem domain, coordination is the act of coordinating, playing different roles together for a goal or effect. In the case of time-based conflict the actor must make choices about what to accept/reject in order to continue. Our view is that deciding between what to accept and what to reject must be based on multi-dimensional reasoning about information and the individual and social consequence of adopting/rejecting requests/tasks. The technique evaluates possible options to take decision. Among the questions to be answered for evaluating what is gain/lost, are the following: (Q1) which task has higher priority from the viewpoint of requester? (Q2) Which task is the most preferred by actor? (Q3) With which requester has the actor a more important relationship? (Q4) How costly will it be to reject the request? (Q5) How much gain will it be to accept the request? And so on. Those questions are not exhaustive and also tightly related to the application scenario. However, in general, we can classify the aspects dealt with by the above questions into three factors. The first factor comprises those properties of information that are relevant to its priority (Q1), while the second considers the properties that are relevant to its preference (Q2 & Q3), and finally the third considers the properties that are relevant to its consequence (Q4 & Q5). Moreover, human behaviour research also identifies that in daily life people’s decision making to resolve conflicts is strongly influenced by those factors as well [17].

Priority. To evaluate a task, priority is very important. It measures the importance of the requested
task/interaction defined by the requester. When an actor sends a task request, he puts the priority on that task from his point of view. The priority depends on the discretion of the sender. We use \( \mathcal{P}_f(t) \) to denote priority of task \( t \) sent from actor \( S \) to actor \( R \).

**Preference.** It measures the importance of the task from receiver’s point of view. Usually it depends on the type of the task and the relationship with the requester of that task. In social interaction, people usually consider social relationship hierarchy to put the preference in a task. For example, in an organization task request from boss always get high preference than colleague as to an employee’s boss position is higher than the colleague. We use \( \mathcal{S}_p(t) \) to denote preference of a task \( t \) requested by actor \( S \) to actor \( R \). We model social interaction as one to one relationship between roles with predefined tasks that the players bound to those roles should performed. So all types of possible interactions are known by those two players and the priority and preferences can be assigned by themselves. However, in our proposed model, organizer role has the capability to add new tasks at runtime.

**Consequence.** Estimating priority and preference are not enough to decide what to reject. We also need to reason about the impact of rejecting a task request, in terms of how much an actor loss if a request is rejected in the existing situation. This may cause money, may incur losing useful relations or may affect one’s image. Another loss is that of potential gains that an actor might had obtained be adopting the retracted request. Examples in social life include that retracting care of elderly mother might cause her death or serious sickness, and that ignoring attendance at executive meeting might hamper job status. We quantify these aspects in a measure of utility. We assume we can compute a numerical estimate of consequence through a utility function – \( \mathcal{U}_f(t) \) is the estimated measure of consequence for task \( t \) from sender \( S \) to receiver \( R \). Like in case of priority and preference, there can be many aspects that determine utility and their complete delimitation may be possible only on the basis of each application domain in part. We choose 0 as lower and 1 as upper bound of those properties. We include 0 for modeling purpose, although it represents meaningless interaction.

**Overall importance.** We quantify a task \( t \) based on its priority, preference, and consequence and infer the overall importance \( I(t) \) of the task by cumulate those predicates value. The task with highest importance value might be selected for execution among conflicted tasks. In literature, a number of reasoning techniques has been proposed, a comparative analysis can be found in [18]. Among those fuzzy inference system attempts to emulate human thoughts, and is very useful in capturing and representing imprecise notions such as “importance”, “trustworthy”, and reasoning about them [19] (chap. 1, p.5). In our case, given a value of priority or preference or consequence, it is more natural to say it is high or low to a certain degree than to claim it is high or low in an absolute sense. We define three fuzzy sets low, medium, and high for priority and use triangulation membership function for each of these sets. The graphical representations of these functions are shown in Fig. 3(a). As we can see, a priority value of 0.3 belongs to fuzzy set low with truth degree of 0.4, but to fuzzy set medium with truth degree of 0.6. We choose the same fuzzy sets and membership functions for the preference and consequence as well. We also define five fuzzy sets not important, less important, moderately important, very important, and very much important for overall importance in a similar way in order to apply fuzzy rules and compute the importance. The graphical representations of these sets are illustrated in Fig. 3(b).

**Fuzzy Inference System.** The basic structure of a fuzzy inference system consists of three conceptual components: fuzzy sets, fuzzy "if-then" rules and fuzzy reasoning. The fuzzy sets and rules are usually derived from domain experts (e.g., social psychologists in our case). For our "work-family" problem domain, we derive

---

**TABLE III. FUZZY CONTROL RULES FOR INFERRING THE OVERALL IMPORTANCE**

<table>
<thead>
<tr>
<th>Overall importance when consequence = low</th>
<th>Preference low</th>
<th>Preference medium</th>
<th>Preference high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority low</td>
<td>Not important</td>
<td>Not important</td>
<td>Not important</td>
</tr>
<tr>
<td>Priority medium</td>
<td>Not important</td>
<td>Less important</td>
<td>Less important</td>
</tr>
<tr>
<td>Priority high</td>
<td>Less important</td>
<td>Less important</td>
<td>Moderately important</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Overall importance when consequence = medium</th>
<th>Preference low</th>
<th>Preference medium</th>
<th>Preference high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority low</td>
<td>Not important</td>
<td>Less important</td>
<td>Less important</td>
</tr>
<tr>
<td>Priority medium</td>
<td>Less important</td>
<td>Moderately important</td>
<td>Moderately important</td>
</tr>
<tr>
<td>Priority high</td>
<td>Less important</td>
<td>Moderately important</td>
<td>Very important</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Overall importance when consequence = high</th>
<th>Preference low</th>
<th>Preference medium</th>
<th>Preference high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority low</td>
<td>Moderately important</td>
<td>Moderately important</td>
<td>Very important</td>
</tr>
<tr>
<td>Priority medium</td>
<td>Moderately important</td>
<td>Very important</td>
<td>Very much important</td>
</tr>
<tr>
<td>Priority high</td>
<td>Very important</td>
<td>Very much important</td>
<td>Very much important</td>
</tr>
</tbody>
</table>
fuzzy rules from the validated hypothesis empirical study [17] in social psychology and human behaviour literature. There are total twenty seven rules (see Table III) for different values of priority, preference, and consequence. Generally, these rules are intuitive and straight forward. For example, when the priority is medium, preference is low and the consequence is medium, the overall consequence should use fuzzy set less important. When the given priority, preference, and consequence belong to their sets according to the set definition, the rule is applied by evaluating the membership values of these inputs. The largest membership value is transferred to the output side by cutting the whole area define by the corresponding fuzzy set for the overall importance at height of that value while the others are discarded. The process is called fuzzy reasoning, shown in Fig. 4.

Suppose a priority \( \Pr(t) = 0.3 \), a preference \( \ Pf(t) = 0.6 \), and a consequence is 0.4. By definition, we know this priority belongs to fuzzy sets low and medium with degrees 0.4 and 0.6 respectively, the preference belongs to medium and high with degrees 0.8 and 0.2 respectively and the consequence belongs to fuzzy sets low and medium with degrees 0.2 and 0.8 respectively. Thus there are eight rules to be applied. Fig. 4 only shows two of them. Steps 1 to 4 correspond to “if the priority is low, the preference is medium, and the consequence is low, then the overall importance is not important”. Steps 1 to 3 evaluate the degrees of variables belonging to fuzzy sets. The lower degree value is 0.2, thus we end up with shaded area in step 4. Steps 5 to 8 illustrate the rule that if the priority is medium, the consequence is medium, and the preference is medium, then the overall importance is moderately important. Thus we have another shaded area together with the previous one in step 8. We repeat the same process for all applicable rules and merge all areas into a combined one, indicated by red thick lines at step 9. Note that a few steps corresponding to other applicable rules are omitted between 8 and 9. The X coordinate of the centroid of this area is the overall importance that we desire, which is 0.39 in this case. This procedure is known as defuzzification [20]. Specifically, this technique is the MAX-MIN inference, where we choose to minimize the degrees of input parameters, and maximize the areas generated by each rule.

V. SYSTEM DESIGN AND IMPLEMENTATION

A. System Architecture

We extend the conventional architecture view of context-aware systems [21] for enabling personalize (player-centric) social interactions. As a result, the architecture that we have developed consists of four main layers (Fig. 5). In comparison to conventional architecture, Layer 3 (highlighted in Fig. 5) is an additional layer responsible for the management of both player- and domain-centric social contexts. The application layer is application-specific that facilitates user’s task and interactions. A person can play and interact with his/her own social context model (PSCM) using applications running on mobile devices. At runtime, the PSCM evolves (add/delete roles, contracts, and so on) based on the person’s (who owns that PSCM) associations with different DSCMs. The PSCM also behaves adaptively according its owner preferences (defined as context rules, e.g., r1 in Table II) and situations acquired from the physical context management (PCM) layer. The details are out of the scope of this paper and will be discussed elsewhere. The PCM layer represents external resources.
(e.g., SituationDetector) where the system acquires physical context facts. The Social context composite could be deployed in the same device hosting the application or distributed across the network.

B. Social Interaction Modeling Tool

We have developed the ROADdesigner (see Fig. 6), which is a graphical modeling tool supporting the design of social context models. It is implemented using the Eclipse modeling frameworks (i.e., EMF/GMF). The software developer uses ROADdesigner to build a social context model by dragging and dropping model components (i.e., roles and contracts) onto a canvas representing the social context composite. These roles are then connected with contracts, and properties and clauses are added to the contracts. Optionally, endpoint references to candidate actors can also be specified at design time. After validation against a pre-defined schema, the model is translated into an XML document. The ROADfactory takes this XML-based context model as input and creates runtime ROAD composites, where each social context is implemented as a ROAD composite and exposed as a set of services which can be invoked by applications. ROADfactory integrates the Drools engine to define and fire context rules.

C. System Prototype

We have implemented three desktop applications (each one for John, his mother Mary, and his boss Bob) to demonstrate how the proposed interaction model and architecture can be applied for enabling social interactions, see Fig. 7. We use MATLAB to build the fuzzy inference model. John’s application uses the fuzzy inference system in the backend to quantify tasks for decision making when a conflict arises. While John is online and receives multiple task requests (for example, his mother asks him to come home as she is not feeling well, and his boss calls an urgent project meeting), his application helps him in taking a decision by providing overall importance of each task. Moreover, it provides historical interactions information by maintaining a log.

VI. DISCUSSIONS

A. Trust

Intelligibility is one of the required principles of context-aware applications, which allow users to understand and participate in the decisions of context-aware applications [22]. Lack of system intelligibility can lead to loss of user trust, satisfaction and acceptance of these applications. One mechanism to alleviate this lack of intelligibility in intelligent context-aware systems is through automatically generated explanations. This approach has been shown to be effective in other domains such as decision making [23] and recommender systems [24] where providing explanations led to increased trust and acceptance. Lim et al. [25] suggest that providing reasoning trace explanations for context-aware applications to novice users, and in particular Why explanations, can improve user's understanding and trust in the system. So, in our system, we provide the explanation of the inferring task-importance to the users (see Fig. 8) by showing the three topmost fuzzy rules that contribute most in the inference process for a particular value of priority, preference and consequence. Moreover, the system enables users to change a variety of settings or rules (see Fig. 9). In addition, the system enables users to refine (add/modify) their preferences at run-time.

2 http://www.jboss.org/drools/
B. Validation

In this paper, we have proposed a fuzzy inference system to quantify task importance that assists users to take decisions in case of task conflict. Hence, the justification of this system would be how intuitively it quantifies the task importance for given input values and particular situations. It is worth to mention that fuzzy inference system provides the basis for representing the imprecise inherent in an expert's knowledge. And we cannot always extract all the information that we need to use. In this case we still wish to come to a conclusion, if possible. This is true even if the conclusion is less certain [26]. The result produced by this system is obviously depends on the specified fuzzy sets of inputs and output predicates, and fuzzy rules.

The fuzzy sets (here low, high, less important, and so on) need to be defined correctly in the domain (here "work-family") of discourse. However, there is no uniform definition of these sets across domains [27]. For example, the fuzzy set high would have a distinctly different definition for temperature from what it would have for river water level. The sets will be derived by the domain expert. It seems likely that different definitions for a term (in a specific domain of discourse) such as high might be obtained. Slight differences in such definitions will probably be acceptable, as these are imprecise concepts. There will be a "range" of possible definitions that will allow the system to function correctly. Hence, the focus must not be on an exact definition for a fuzzy set, but on one that is acceptable in the system. A fuzzy inference system is not expected to be extremely sensitive to small changes in the definitions of the sets [27].

The major part of fuzzy inference system validation lies in the validation of fuzzy rules (knowledge) [27]. The inference rules we use are based on validated hypothesis from social psychology and human behavior literature [5][17]. In our approach, we take the investigation as revealing the nature of the knowledge that is involved in social behavior and interactions. Our aim is to provide a generic approach for the capture, representation and use of this knowledge in the development of an interactive application. The insight into the nature of social interactions from sociological or psychological sources has provided us principles and background knowledge for understanding and modeling interactions. Together with domain and application knowledge, they can be used to design coordination structures and guide the interactions among actors. Here, we further demonstrate the applicability using several test cases.

Revisiting the scenario. Let us consider the scene#2 in section 2. At a certain time, John receives request messages from his Mother and Boss respectively with a certain priority value. His application identifies the conflict and infers the overall importance (I) of both interactions (0.7 and 0.6 respectively) based on the given priority, consequence, and John’s personal preference (e.g., r1 rule in Table II). Cases #1 and #2 show that consequence (U) of comingBackHome task depends on the mother's physical condition, for example, it is 0.85 when she is not feeling good and 0.4 when she is well. In latter case, even the mother asked to come home with priority 0.9, the importance infers 0.58 which is less than the former case. It makes intuitively sense. Because the receiver (here John) does not have any control on priority value of a received task request. So, in case #2, even Mary puts high priority value (0.9) to the comingBackHome task in a good physical condition, the model infers less importance value (0.58) for the same preference value (0.8) as the consequence of that task is 0.4 (Mary's physical condition is good, so will not loss much if John ignores that request). Cases #3 and #4 show that the importance of the attendMeeting task does not vary by increasing the priority whereas the preference and consequence are fixed. It means that as the consequence of not attending the meeting and John's preference to attend that meeting remain the same, the requester's priority value to attend does not increase the overall importance of that task. It is worth to note that the expected result of quantification might vary from person to person. However, based on empirical user studies, the human behavior literature identifies that generally in this problem domain human quantifies the task most likely in the above way [17].

VII. RELATED WORK

Within the scope of this paper, literature from various areas has been studied and grouped into three main streams: (1) Social interaction modeling, (2) Conflict resolution techniques, and (3) Social applications.

A. Social Interaction Modeling

A range of methods already exist to enable interactions between separated individuals, including email, phone call, instant messaging (IM), social networks and so on. Phone call is usually intrusive, because people do not know when the phone will ring, and what it is about. Email time-decouples the sender and receiver as transferring a message from a sender to a receiver might take a long time and the receiver might not read message immediately upon receipt. In an IM system, users can send message only to people who has already subscribed to it and currently available. Social networks such as Facebook and Twitter have experienced an enormous growth in recent years. However, these methods suffer from several drawbacks. For example, usually they require a major effort to keep relationships between social groups alive. They adopted friend-of-friend ontology and usually used to build network and share contents among people of that network. They are hardly support interactions between specific

<table>
<thead>
<tr>
<th>#</th>
<th>Task</th>
<th>Pr</th>
<th>Pf</th>
<th>U</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>comingBackHome(s1)</td>
<td>0.8</td>
<td>0.8</td>
<td>0.85</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>comingBackHome(s2)</td>
<td>0.9</td>
<td>0.8</td>
<td>0.4</td>
<td>0.58</td>
</tr>
<tr>
<td>3</td>
<td>attendMeeting</td>
<td>0.9</td>
<td>0.7</td>
<td>0.5</td>
<td>0.6</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>1.0</td>
<td>0.7</td>
<td>0.5</td>
<td>0.6</td>
</tr>
</tbody>
</table>

s1:Mother.situation=NotFeelingGood, s2:Mother.situation=Well
individuals’. Moreover, they do not provide a systematic way to personalize social interactions with coordination functionality.

Social interaction has also been studied in multi-agent systems (e.g., [11]). In a multi-agent system, agents usually interact in order to achieve better goals of them or the system in which they exist. Kalenka et al. incorporate social concepts within the computer system level paradigm to enhance the agent's interaction performance [12]. Rather than defining social relationships, the paper proposes multi-furioso social interaction attitudes ranging from self-interested to the purely altruistic. In BRAIN [11], Cabri et al. propose a framework to model agent interactions. Although the framework relies on the concept of role for developing agent based applications, it does not consider the user's perspective of interaction modeling with the issue of playing multiple roles. Moreover, they do not provide any tool support for a novice user to ease the task of application development.

A number of context modeling techniques has been developed, including graphical approach, object oriented, mark-up scheme, logic-based, and ontological models (as reviewed in [18]). However, these approaches are particularly useful for modeling physical context. They are very limitedly support social context modeling requirements [4] including player-centric modeling.

B. Conflict Resolution Techniques

An intensive effort has been done to resolve conflicts between multiple users of context-aware applications in pervasive environments. Basically, the focus is how to deal with conflicts caused by multiple users accessing a single context-aware application. Several techniques have been proposed to understand and address this problem. Dynamic conflict resolution automatically resolves the conflict of contradicting actions of users by determining the outcome with the lowest deviation from what each wanted [28]. This approach effectively computes the optimal solution of a continuous value and thus can be applied to automatic control of appliances such as lights and air-conditioners. A different approach used to automatically resolve conflict is to assign priorities to the users and their preferences based on their profiles [29]. The mechanism is useful when priorities can be easily assigned.

Rather than automatically resolve conflict, another approach is to use mediation techniques that give users information to help resolve conflict. Jukola, for example, is a music mediator for selecting songs for customers in a coffee shop [30]. It recommends candidate songs to the customers, allows customers to vote for them and plays the item with the most votes. Public spaces, where users are less likely to engage in mediation with the strangers around them, are particularly appropriate for this type of conflict resolution approach. The main drawback of this technique is that users are requires to be active participants in conflict resolution.

In general, the above techniques can not be adopted directly to our research domain for two reasons. First of all, rather than resolving physical resource conflicts between multiple users; our focus is to resolve time conflict of a person that occurs when multiple task requests come simultaneously. Secondly, the above techniques applied in an environment where users are co-located (e.g., users are sharing TV at home), while in our problem domain usually persons are physically located in different places (e.g., while a person in office receives task request from his mother staying at home).

C. Social Applications

In the area of personal digital assistants, Bosse et al. [7] propose a personal assistant agent to monitor the user’s state and task execution. Helmy et al. [8] present a personal information manager based on the agent-oriented paradigm to support speech recognition and calendar scheduling. However, those do not consider the user’s social context at all.

In the area of social-mobile applications, RENO is proposed for locating friends by calculating the device's approximate location [31]. Within MobiSoft project [9], Erfurth et al. investigate the application of social-mobile software agents as personalized assistants on mobile devices to find human partners in close proximity with same interest. Pernek et al. [32] introduce SocioNet, a context-aware rule based system for mobile devices and consider performance, privacy and usability issues to find best matching persons in proximity. Hung et al. [33] propose a system of social network agents for managing social contacts over the web. It is designed to help users to retrieve contact information by tagging via a simple user interface. Stoica et al. [10] present a context-aware application to support personalized interaction across multiple augmented spaces. Devlic et al. present a rule-based inference approach for deriving user social relations with his/her communication partners based on the log-data collected by their mobile phone [34]. However, none of these facilitate personalized social interactions with coordination functionality.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we have presented an innovative approach to modeling social interactions from both domain and player perspectives with the notion of social context. Particularly, the personal social context (player perspective) model enables to define preferences and adaptive behavior that facilitates coordination functionality. Moreover, it might assist an application developer to build socially-aware applications that aid users to maintain and coordinate their daily life interactions using mobile devices (e.g., PDAs, phones) from anywhere. Specifically the personal preferences can be used to quantify the importance of conflicted tasks/interactions, while the adaptive behavior (defined as context rules) enables applications to act on behalf of user considering his/her situations. In relation to coordination, we further address time-based conflict and introduce a fuzzy inference system to quantify the importance of conflicted tasks, assisting the user to make decisions. This
work is a necessary first step in exploring this research area, in particular mediated conflict resolution.

In our future work, we plan to investigate dynamically adapting the fuzzy inference model at run-time based on user's feedback. We plan to extend our approach to support other types of conflicts and propose a model driven approach for developing socially-aware applications. We also aim to empirically study the usability of the system. In addition to a focus on improving the technology, we also plan to study our system with real applications in real environments.

**ACKNOWLEDGMENT**

This original research was proudly supported by the Commonwealth of Australia, through the Cooperative Research Centre for Advanced Automotive Technology.

**REFERENCES**


