A privacy-friendly model for an efficient and effective activity scheduling inside dynamic virtual organizations

Salvatore F. Pileggi

INRIA & UPMC-LIP6, Paris, France
flavio.pileggi@lip6.fr
www.flaviopileggi.net

Abstract. The cooperation among people is one of the key factors for most processes and activities. The efficiency and the effectiveness of the cooperation has an intrinsic value, which significantly affects performances and outcomes. Open communities, as well as spontaneous or predefined virtual organizations, are demanding for a more solid and consistent support for activity scheduling and managing in a context of flexibility and respect of the individual needs. This paper proposes a privacy-friendly model to support virtual organizations in the scheduling and management of their most valuable resource: the time.

Key words: Collaborative Application, Distributed Computing, Privacy, Organisational Model

1 Introduction

The efficiency and the effectiveness of the cooperation among people has an intrinsic value, which significantly affects performances and outcomes of many processes and activities, at a quantitative and at a qualitative level [1]. Classic models used in the real world by companies have evolved at a theoretical level to integrate a more flexible philosophy (e.g.[2]). More recently, some of those models have started to be considered also in practice, as many companies are progressively leaving from classic schemas to evolve towards novel approaches where individual needs and effectiveness converge under the realistic assumption that the personal and the collective development come together. Indeed, open communities, as well as spontaneous or predefined virtual organizations, are demanding for a more solid and consistent support for activity scheduling and management in a context of flexibility and respect of individual needs. This paper focuses on activity scheduling and proposes a privacy-friendly model, called *Optimistic Scheduling* due to the implicit optimism that drives interactions among people to manage their most valuable resource: the time. The most novel solutions on the market offer features that the current reference tools (e.g. *Google Calendar*) are missing. The most dominating trends appear in coherence with the current technological climate [3] which pushes towards a progressive

socialization of tools and applications. Simple observations clearly show an unpredictable human behaviour, even in simple or well-known situations. Therefore, the model is analysed and evaluated by simulation as the function of complex user behaviours. An empirical overview of collaborative tools demonstrates as the simplest approaches (e.g.[4]) usually reach the best results, meaning products are well accepted by users and, indeed, are more usable in practice. Proposing a completely generic tool is hard and, in some case, not effective. In this work, the community is understood as a whole, meaning that users can interact each others at a global level. However, the ideal application domain assumes a *virtual organization* model, where existing and relatively static real groups (e.g. companies, institutions, teams, groups of friends) are integrated with dynamic groups that can change along the time (e.g. cooperative projects) or that are defined on the fly as the function of concrete tasks or activities. It is also assumed that users inside groups are peers. That is not always realistic as groups are often organized according to some structure or hierarchy. Moreover, it is supposed that a shared activity inside a dynamic group requires the participation of all the members. This ideal case could not suit real virtual organizations. Finally, individual preferences have a priority over groups (in contrast with most current approaches). This should push the cooperation and should optimize the use of the time.

2 Beyond the state of the art: overcoming a simple use case

The schedule of a shared event is commonly performed according to two different approaches:

- *Scheduling by invitation* implies the organizer sending a specific invitation to the interested participants by using some shared channel (e.g. email, message, sms). This is the simplest and most used method. But it is also very vulnerable as the synchronization schema among the users is weak. First of all, the event cannot be considered as committed if confirmations are not received from all group members. If at least one of the expected participants rejects the invitation, the organizer needs to restart the process sending a new invitation. Furthermore, a missed response from at least one member can generate misunderstandings (not received? Not seen? Not interested? Unable to respond?), requires further actions from the organizer (e.g. reminders) and can lead to a potential deadlock.
- *Scheduling by poll* is the most common alternative. The organizer proposes a set of possible slots and participants are asked to express their preferences. Having a poll of possible options reduces significantly the vulnerability of the model as the organizer has to set up a new poll only if there is no agreement. Unfortunately, a lack of response from some member proposes the same problems discussed above. Furthermore, once a member has expressed his preferences, he should wait for the final confirmation from the organizer before

using the time slots he potentially accepted with an evident inefficiency in the management of time.

Both methods have a further common weakness: what if a group member changes his plans after a commitment? The process should start over!

3 Optimistic Scheduling: Efficient and effective time sharing inside Virtual Organizations

Each community member has his own calendar TS_P that is considered a private asset as no-one else in the community has access to it. On this personal calendar, each user i performs a preliminary filtering as he sets a priori the slots that can be used for a personal activity (ts_P) or a shared activity (ts_S), defining his personal space K_i. An example of preliminary filtering is represented by a calendar that only includes normal work hours and that doesn't include leaves or other absences known a priori. The whole space K of slots is given merging users' spaces according to eq.1. On that filtered calendar, a user i can schedule personal activities (eq.2) for any available slot $k \in K_i$. According to the same logic, a user can try to schedule a shared event (eq.3) involving other members (group).

$$K = \bigcup_i K_i \quad \forall i \qquad (1) \qquad\qquad TS_P^i(t) = \sum_k ts_P^i(k,t) \qquad (2)$$

$$TS_S^i(t) = \sum_g \sum_k ts_S^g(k,t), \quad i \in g \quad (3) \qquad\qquad TS^i(t) = TS_P^i(t) \cup TS_S^i(t) \qquad (4)$$

The full activity set TS for a community member i is given by merging his personal activities (TS_P) and the shared activities (TS_S) as in eq.4. The whole potentially shared time can be at least equal to K, assuming people have no personal activities scheduled (eq.5). Users can only see their own calendars. In order to get an effective guide to schedule shared events, users can access, for any defined group they are joining, a shared structure FTS obtained by merging the personal calendars and returning the anonymized complementary set according to eq.6. That anonymized structure shows (fig.4) the slots that can be used, inside a certain group, to schedule a shared event. This simple operation allows, in fact, users to automatically understand the availability of a certain group in the respect of the members' privacy. Assuming a significant group size, inferring information is not easy, so the privacy is completely preserved. By using those structures, whichever member in a group can schedule a shared activity for a dynamic or static group with high probability of success (fig.4). The semantic of the model implicitly defines the main global invariant (a logical assertion that is held to always be true during a certain phase of execution of a program [5]):

if a time slot ts is used by a member i of a group g for a personal purpose, then that slot cannot be used for a shared activity inside any group i is member of (eq.7).

$$\bigcup_i TS^i(t) = K \Rightarrow TS^i_P(t) = \emptyset, \forall i \quad (5) \qquad FTS^g(t) = K - \bigcup_i TS^i(t), i \in g \quad (6)$$

$$\exists ts^i_P(k_a, t) \quad \Rightarrow \quad \nexists ts^g_S(k_a, t), \quad i \in g, \quad k_a \in K \quad \forall t \tag{7}$$

The model works assuming multiple simultaneous groups, providing an individual-specific view of each group in a privacy-friendly context. As individual needs have a priority on groups activity, users can still schedule their own activities also for slots that are already currently in use as shared resources. In this case the invariant defined by the eq.7 is not respected determining a non-valid state for the system that, coherently with the assumptions, reacts (for example cancelling the shared event and notifying the interested members about).

4 Model Analysis

The most significant issue for the analysis and the full understanding of this model is the definition of realistic user behaviours, being aware that people are or can be unpredictable. In this study, the synthetic actor that emulates users assumes a linear logic implementing three different behaviours:

- *Constructive.* The user is "cooperative" and, therefore, acts according to a logic that facilitates the successful scheduling of shared initiatives. A constructive user schedules his activities only in slots not currently in use for shared events and uses shared slots only if there is no other choice.
- *Disruptive.* This is the opposite of the previous as he schedules personal activities prioritizing the slots currently occupied by shared events. This behaviour causes the continuous reorganization of the shared events. It is not necessarily reproducing a malicious user, as it could also emulate an involuntary "noise" caused by random circumstances or periodic conflicts on the schedule.
- *Random/Independent.* Between the two extremes (constructive and disruptive) there is an infinite range of behaviours, including an independent user that acts according to a pseudo-random logic that doesn't take into account the existence of groups.

The metric to evaluate the model performance (eq.8) is directly proportional to the number of shared activities successfully scheduled and inversely proportional to the number of shared activities cancelled upon request of users.

$$\alpha(t) = \frac{1 + \sum_i TS^i_S(t)}{\sum_i TS^i_S(t) + \sum_i TS^i_S(t)_{|c}} \quad (8) \qquad \frac{d}{dt} TS^i(t) > 0 \tag{9}$$

The simulations are assuming a finite *sliding window* of size m to reproduce the time. At the time t, users can only schedule between the slot $t+1$ and the slot $t+m$. The logic transition from t to $t+1$ implies the slot t no more available (past) and a new free slot $t+m+1$ available. Furthermore, the experiment also assumes that the number of scheduled events tends to increase in the time (eq.9, meaning users scheduling a higher number of events than the number of events cancelled or consumed. For simplicity atomic slots don't overlap each others (eq.10).

$$ts^i_{P/S}(k_1, t) \cap ts^i_{P/S}(k_2, t) = \emptyset, \quad \forall k_1, k_2 \in K, \forall i \tag{10}$$

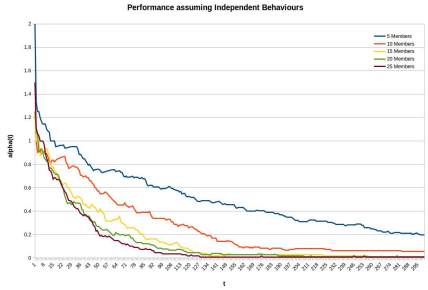

Fig. 1. Independent behaviour.

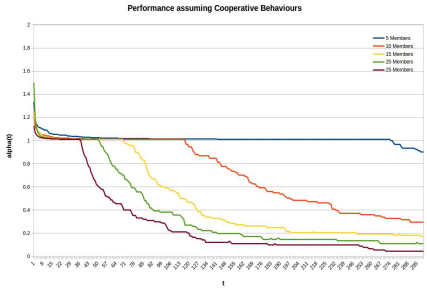

Fig. 2. Cooperative behaviour.

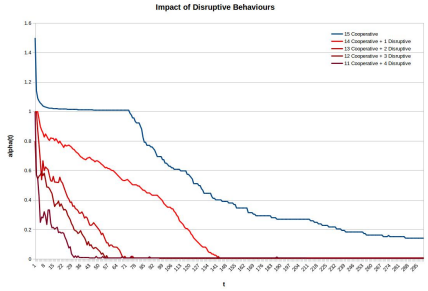

Fig. 3. Impact of disruptive behaviours.

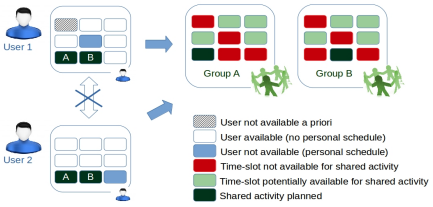

Fig. 4. Potentially shared time, shared activities and multiple groups.

The simulations assume a sliding window of 12 weeks to schedule activities and members averagely available 35 hours (slots) per week, as in a common work calendar. The calendar is empty when the simulation starts, so there is a transitory period. The members schedule averagely an activity per day and the 25% of the planned activities are shared. The simulation ends when the system is saturated (no more possibility to schedule events) or when the system has reached stationary/stable conditions. Fig.1 shows the performance decreasing as the function of the group size assuming independent behaviours. That is a very

good approximation of performance inside a virtual organization where users are not acting according to this model. Fig.2 proposes the same statistics but assuming a cooperative behaviour. Is emulates a community that acts according to the proposed model. Performances are evidently higher than the previous and decrease only for the natural saturation of the system determined by the quantitative behaviour (eq.9). The chart in fig.3 provides an overview of the potential impact of disruptive behaviours on the whole performance. As showed, if one or more members is acting according to a disruptive behaviour, then performances quickly decrease and the system tends after a very short time to the saturation. Disruptive behaviours are part of real life and have to be taken into account at the time of designing real tools. They are easy to detect in common mechanisms (such as invitation and polls) due to the explicit character of the interactions. On the contrary, disruptive behaviours are hidden in a privacy-friendly context. At a model level, the global invariants (eq.7) can be relaxed to mitigate the effect of disruptive behaviours. This approach introduces at least one significant and critical trade-off between effectiveness and privacy/simplicity. Indeed, assuming that a slot inside a group can be used simultaneously for a personal and a shared activity protects the system from disruptive behaviours but also introduces ambiguities in the understanding and the management of the system state. Considering anonymous non-availabilities, the organizer cannot know who is missed, so the further steps of the activity planning could be negatively affected. On the other hand, concessions about privacy could invalidate most premises and, consequently, modify significantly the whole model focus. Anyway, integrating a complex could lead to applications that miss their aimed simplicity.

Acknowledgements

This research is supported in part by European FP7 project SyncFree (grant agreement 609551).

References

1. John A Wagner. Studies of individualism-collectivism: Effects on cooperation in groups. *Academy of Management journal*, 38(1):152–173, 1995.
2. Gareth R Jones and Jennifer M George. The experience and evolution of trust: Implications for cooperation and teamwork. *Academy of management review*, 23(3):531–546, 1998.
3. Salvatore F. Pileggi, Carlos Fernandez-Llatas, and Vicente Traver. When the social meets the semantic: Social semantic web or web 2.5. *Future Internet*, 4(3):852–864, 2012.
4. Murali Raman. Wiki technology as a" free" collaborative tool within an organizational setting. *Information systems management*, 23(4):59–66, 2006.
5. Marc Shapiro, Nuno Preguiça, Carlos Baquero, and Marek Zawirski. Conflict-free replicated data types. In *Stabilization, Safety, and Security of Distributed Systems*, pages 386–400. Springer, 2011.