An Algebraic Algorithm for Structural Validation of Social Protocols

Willy Picard

Department of Information Technology,
The Poznań University of Economics,
ul. Mansfelda 4, 60-854 Poznan, Poland,
`picard@kti.ae.poznan.pl`,
WWW home page: `http://www.kti.ae.poznan.pl`

Abstract. Support for human-to-human interactions over a network is still insufficient. In this paper a model for human-to-human collaboration based on the concept of social protocol is presented and formalized. Then, semantical and structural validity of social protocols is defined. Next, an algebraic representation of social protocols is proposed. Based on this algebraic representation of social protocols, an algorithm for structural validation of social protocols is proposed and illustrated by three examples.

Keywords: collaboration modeling, algebraic representation of social protocols, semantical validation, structural validation.

1 Introduction

Enterprises are constantly increasing their efforts in order to improve their business processes, which may be explained by the fact that enterprises are exposed to a highly competitive global market. Among the most visible actions associated with this effort towards a better support for better business processes, one may distinguish the current research works concerning Web services and associated standards: high-level languages such as BPEL or WS-Coordination take the service concept one step further by providing a method of defining and supporting workflows and business processes.

However, most of these actions are directed towards interoperable machine-to-machine interactions over a network. Support for *human-to-human interactions* over a network is still insufficient and many research has to be done to provide both theoretical and practical knowledge to this field.

Among various reasons for the weak support for human-to-human interactions, two reasons may be distinguished: first, many *social elements* are involved in the interaction among humans. An example of such a social element may be the roles played by humans during their interactions. Social elements are usually difficult to model, e.g. integrating non-verbal communication to collaboration models. Therefore, their integration to a model of interaction between humans is not easy. A second reason is the *adaptation capabilities* of humans which are not only far more advanced than adaptation capabilities of software entities, but also are not taken into account in existing models for collaboration processes.

A model for human-to-human interactions which addresses, at least to some extent, the two characteristics of the interactions between humans is therefore needed. Such a model has already been presented in [1,2]. This model is based on the concept of *social protocol* which may be seen as a model of collaboration processes. A collaboration process may be modelled as a social protocol which describes the potential interactions of collaborators within this process.

In the case of complex collaboration processes, the design of a social protocol modeling these processes may be a complex task. A social protocol may be designed with errors potentially leading to unachievable collaboration processes, i.e. processes in which collaborators are locked and cannot continue their collaboration. Therefore, some techniques to check the validity of social protocols are needed.

In this paper, the concept of *validity* of social protocols is defined. Then, an algorithm for structural validation of social protocols is detailed. This algorithm is based on an algebraic representation of social protocols.

The rest of this paper is organized as follows. In Sect. 2, the concept of social protocol, used to model collaboration processes, is presented. Section 3 then expands on the definition of the concepts of semantical and structural validity of social protocols. Next, an algorithm for structural validation of social protocols is proposed in Sect. 4, and illustrated by three examples in Sect. 5. Then, related works are reviewed in Sect. 6. Finally, Section 7 concludes this paper.

2 Modeling Collaboration Processes as Social Protocols

A social protocol aims at modeling a set of collaboration processes, in the same way as as a class models a set of objects in object-oriented programming. In other words, a social protocol may be seen as a model which instances are collaboration processes. Social protocols model collaboration at a group level. The interactions of collaborators are captured by social protocols. Interactions are strongly related with social aspects, such as the role played by collaborators. The proposed model integrates some of these social aspects, which may explain the choice of the term *social protocols.*

2.1 Formal Model of Social Protocols

Before social protocols may be formally defined, others concepts must first be defined.

Definition 1. *A* role *is a label. Let denote R the set of roles.*

In a given group, a set of roles is played by the collaborators, which means that collaborators are labeled, are associated with given roles. The set of roles R_p for a given protocol p is a subset of R, i.e. $R_p \subseteq R$. Collaborators usually play different roles within a given collaboration process. Roles may be associated with collaborators to specify the way they should interact with the rest of the group. Interactions among collaborators are modeled with the concept of *action type*.

Definition 2. *An* action type *is an interface of a software entity. Let denote A the set of action types.*

An action may be for instance the execution of a web service, a commit to a CVS repository, the sending of an email. Within a group, collaborators are interacting by executing actions. The execution of actions is a part of the common knowledge of the group, i.e. all collaborators are aware of the execution of an action by one of the members of the group. An action type may be seen as a description of a given action, providing the name and type of parameters required to execute the action as well as the type of the result returned by the action execution.

Definition 3. *A* behavioral unit *is a pair $(role, action\_type)$. Let denote BU the set of potential behavioral units. Formally, $BU = R \times A$.*

The concept of behavioral unit comes from the idea that the behavior of a collaborator is to a large extent determined by the role he/she plays. Therefore, roles and action types have to be associated to determine the behavior, i.e. the set of actions that a collaborator playing a given role can perform.

A behavioral $bu = (r, a)$ is said to be executed iff a collaborator labeled with the role r executes the action of the given type a. It should be notice that only collaborators labeled with the role r can execute the behavioral unit $bu = (r, a)$.

Definition 4. *A* state *is a label associated with a given situation in a collaborative process. Let denote S the set of states.*

In a given protocol p, the set of states that may occur S_p is a subset of S, i.e $S_p \subseteq S$.

Definition 5. *A* transition *is a triplet $(bu, s_{\text{source}}, s_{\text{destination}})$. Let denote T the set of transitions. Formally, $T = BU \times S \times S$.*

In a given protocol p, the set of transitions T_p is a subset of T, i.e $T_p \subseteq T$.

Now that all concepts underlying social protocols have been formally presented, the concept of social protocol may be defined.

Definition 6. *A* social protocol *is a finite state machine. A social protocol consists of $\left\{S_p, {S_p}^{\text{start}}, {S_p}^{\text{end}}, R_p, A_p\right\}$ where ${S_p}^{\text{start}} \subset S_p$ is the set of starting states, ${S_p}^{\text{end}} \subset S_p$ is the set of ending states, ${S_p}^{\text{start}} \cap {S_p}^{\text{end}} = \emptyset$. Let denote P the set of social protocols.*

In a social protocol, collaborators are moving from state to state via the execution of behavioral units. In other words, the execution of behavioral units are transition conditions. As mentioned before, a behavioral unit may be executed only by a collaborator labeled with the appropriate role.

An extended definition of social protocols have been presented in [1]. An application of social protocols to electronic negotiations may be found in [3].

2.2 An Example of Social Protocol

The example of social protocol which is presented in this section is oversimplified for readability reasons. It is obvious that social protocols modeling real-world collaboration processes are usually more complex. The chosen collaboration process to be modeled as a social protocol may be described as follows: a set of users are collaborating on the establishment of a "FAQ" document. Some users only asks questions, while others, referred as "experts", may answer the questions. Other users, referred as "managers", may interrupt the work on the FAQ document. The work on the document may be terminated either by a success (the document has been written and the manager estimates that its quality is good enough to be published) or by a failure (the users did not find any way to collaborate and the manager has estimated that the work on the FAQ should be interrupted).

A possible social protocol modeling this collaboration process is presented in Fig. 1.

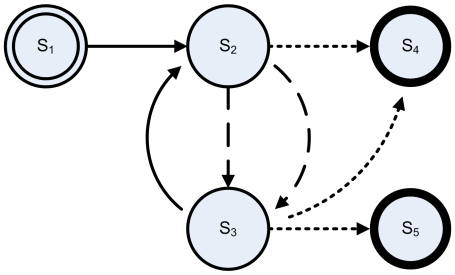

Fig. 1. An example of social protocol

In Fig. 1, five states $s_1, \ldots, s_5$ are represented as circles. State s_1 is a starting state, states s_4 and s_5 are ending states. The following states are defined:

- state s_1: waiting for a first question;
- state s_2: waiting for an answer;
- state s_3: waiting for a next question;
- state s_4: failed termination;
- state s_5: successful termination.

Transitions are represented as arrows, and the line style is associated with the role of the users that may execute a given transition. Continuous line style is used to represent transitions that may be executed by “normal users”, fine-dashed style for transitions that may be executed by “experts”, and fine-dotted style for transitions that may be executed by “managers”. Transitions are summarized in Table 1.

Table 1. Transitions for the example of social protocol

Source state	Destination state	Role	Action
s_1	s_2	Normal	Ask question
s_2	s_3	Expert	Answer question
s_2	s_3	Expert	Suppress question
s_2	s_4	Manager	Failure ending
s_3	s_2	Normal	Ask question
s_3	s_4	Manager	Failure ending
s_3	s_5	Manager	Successful ending

3 Social Protocol Validity

Before the conditions for social protocol validity are presented, the concept of *social protocol validity* should be defined. Two kinds of social protocol validity may be distinguished: a social protocol may be *semantically valid* and/or *structually valid*. Finally, a social protocol is valid iff it is both semantically and structurally valid.

3.1 Semantical Validity

A given social protocol is semantically valid iff

1. all transitions leading to an ending state are associated with behavioral units whose actions end the collaboration;
2. no transition leading to a non-ending state is associated with behavioral units whose actions end the collaboration.

The first condition ensures that each transition leading to an ending state actually ends the collaboration. The second condition ensures that the collaboration cannot be “interrupted” by a transition leading to a non-ending state.

The semantical validity of a given social protocol may be relatively easily checked: 1) all behavioral units associated with transitions leading to an ending state should contain only ending actions; 2) all behavioral units containing ending actions should be associated only with transitions leading to ending states.

3.2 Structural Validity

A given social protocol is structurally valid iff

1. for each non-starting state s, it exists at least one path from one starting state to the state s;
2. for each non-ending state s, it exists at least one path from the state s to an ending state;
3. for each state s and each behavioral unit bu, it exists at most one transition from the state s associated with the behavioral unit bu.

The first condition ensures that each state is reachable, i.e. there is no state to which one may not move to from a starting state. The second condition ensures that there is no state from which one may not move to an ending state. The third condition ensures that there is no "ambiguity" in a protocol. An ambiguity may occur in a protocol in the case when it exists many transitions associated to a common behavioral unit, leading from a given state to various states. In such a protocol, the execution of the "shared" behavioral unit may not be performed as it is then impossible to decide which state should be the next one.

While the third condition may be relatively easily checked, checking that the first and second conditions are fulfilled is a more complex task which requires more advanced algorithms.

4 Structural Validation of Social Protocols

In this section, an algorithm for structural validation of social protocols is presented. This algorithm is based on an algebraic representation of social protocols.

4.1 Algebraic Representation of Social Protocols

Any social protocol may be represented in an algebraic form as a *transition matrix.* The formal definition of the transition matrix requires the definition of the concepts of *sorted states list* and *set of local behavioral units.*

A sorted states list Σ_p for a given protocol p is a list containing once all states of the protocol p and such that the first states of the list are starting states of the protocol and the last states of the list are ending states.

Definition 7. *A* sorted states list *for a given protocol* p *is a list* $\Sigma_p = \{\sigma_i \in S_p\}$ *with* $i \in \{1, \dots, |S_p|\}$ *such that:*

- $S_p \cap \Sigma_p = \emptyset$,
- $\forall (i,j) \in \{1, \dots, |S_p|\}^2, i = j \Leftrightarrow \sigma_i = \sigma_j$,
- $\exists (a,b) \in \{1, \dots, |S_p|\}^2, 1 \leq a < b \leq |S_p|$ *and*

$$\begin{cases} \forall x \in [1,a], & \sigma_x \in {S_p}^{\text{start}}, \\ \forall x \in]a,b[, & \sigma_x \in S_p - \left({S_p}^{\text{start}} \cup {S_p}^{\text{end}}\right), \\ \forall x \in [b, |S-p|], & \sigma_x \in {S_p}^{\text{end}}. \end{cases}$$

A *set of local behavioral units* ${\beta_p}^{s,s'}$ for a given protocol p is the set of behavioral units associated with a transition from state s to s'. Let denote ${\beta_p}^S$ the set of sets of local behavioral units.

Definition 8. *A* set of local behavioral units *from s to s' is a set ${\beta_p}^{s,s'} = bu^{s,s'}$ such that:*

- $\forall bu^{s,s'} \in {\beta_p}^{s,s'}, bu^{s,s'} \in BU_p$,
- $\forall bu^{s,s'} \in {\beta_p}^{s,s'}, \exists t \in T_p$ *such that* $t = (bu^{s,s'}, s, s')$.

Sorted states lists and sets of local behavioral units are required to build a *transition matrix.* A transition matrix Θ_p is an $|S_p| \times |S_p|$ matrix which elements are sets of local behavioral units laid out according to a sorted states list.

Definition 9. *A* transition matrix *Θ_p is an $|S_p| \times |S_p|$ matrix such that* $\Theta_p : \{1, \ldots, |S_p|\} \times \{1, \ldots, |S_p|\} \rightarrow {\beta_p}^S$, $\Theta_p[ij] = {\beta_p}^{\sigma_i, \sigma_j}$.

The elements of a transition matrix are sorted according to a sorted states list, i.e. the first columns and rows are related with starting states, while last columns and rows are related with ending states. Each element of a transition matrix is a set of local behavioral unit for a given source state (in row) and a given destination state (in column).

A *transition cardinality matrix* $\Theta_{p,||}$ may be easily computed from a transition matrix. A transition cardinality matrix is an $|S_p| \times |S_p|$ matrix which elements are the *cardinality* of sets of local behavioral units laid out according to a sorted states list.

Definition 10. *A* transition cardinality matrix *$\Theta_{p,||}$ is an $|S_p| \times |S_p|$ matrix such that* $\Theta_{p,||} : \{1, \ldots, |S_p|\} \times \{1, \ldots, |S_p|\} \rightarrow \mathbb{N}$, $\Theta_{p,||}[ij] = |{\beta_p}^{\sigma_i, \sigma_j}|$.

Each element of a transition cardinality matrix is the number of transitions from the source state (in row) to the destination state (in column).

4.2 State Reachability Computation

The *reachability* of a state s' from state s in a protocol p means that there is a list of transitions in p connecting state s to state s'. To formally define the concept of reachability, let's first introduce the concept of *path.*

A path ${\pi_p}^{s,s'}$ from the state s to the state s' is a list of transitions connecting s to s'.

Definition 11. *A* path *from the state s to the state s', denoted ${\pi_p}^{s,s'}$, is such that* ${\pi_p}^{s,s'} = \langle s_1, t_1, s_2, t_2, \ldots, s_{n-1}, t_{n-1}, s_n \rangle$ *with*

- $s_1 = s$,
- $s_n = s'$,
- $\forall i \in [1, n-1]$, $t_n = (bu_n, s_n, s_{n+1})$.

The *length* of a path is defined as the number of its transitions.

A state s' is reachable from state s in a given protocol p iff it exists at least one path from state s to state s'.

Definition 12. *A state s' is n-*reachable *from s iff it exists at least one path of length n from s to s'.*

The n-reachability of a state s from state s' means that there is a list of exactly n transitions connecting s to s'. Let $\pi_{p,||=n}^{s,s'}$ denote the number of paths of length n from s to s' in protocol p.

Definition 13. *A* path cardinality matrix Π_p *is an $|S_p| \times |S_p|$ matrix such that*

$$\Pi_p = \sum_{n=1}^{|S_p|-1} \Theta_{p,||}^{n}$$

The path cardinality matrix contains information about the reachability of states: each element of the path cardinality matrix is the number of paths from the source state (in row) to the destination state (in column).

Theorem 1. *A state s_j is reachable from s_i iff $\Pi_p[ij] \neq 0$.*

Proof. The transition cardinality matrix contains information about the 1-reachability of states. As $\Theta_{p,||}[ij] = |{\beta_p}^{\sigma_i,\sigma_j}|$, each element of the transition cardinality matrix is the number of transitions from the source state (in row) to the destination state (in column), i.e. the number of path of length 1.

The number of paths of length 2 may be calculated on the basis of the transition cardinality matrix. Let s, s', and s'' be three states. The number of paths of length 2 from the state s to the state s' through the state s'' equals the number of paths of length 1 from the state s to the state s'' multiplied by the number of paths of length 1 from the state s'' to the state s'. Therefore, the number of paths of length 2 from the state s to the state $s\prime$ equals the sum of the number of paths of length 2 from the state s to the state s' through any state $s'' \in S_p$. Formally,

$$\pi_{p,||=2}^{s,s'} = \sum_{i=1}^{|S_p|} \pi_{p,||=1}^{s,s_i} . \pi_{p,||=1}^{s_i,s'}$$

One may recognize in the former equation the classical multiplication of matrices. Moreover, as $\pi_{p,||=1}^{s_i,s_j} = |{\beta_p}^{\sigma_i,\sigma_j}| = \Theta_{p,||}[ij]$, it may be concluded that $\pi_{p,||=2}^{s,s'} = \Theta_{p,||}^{2}[ij]$. Therefore, each element of the $\Theta_{p,||}^{2}$ matrix is the number of paths of length 2 from a source state (in row) to a destination state (in column).

In a similar way, it may be demonstrated that each element of the $\Theta_{p,||}^{n}$ matrix is the number of paths of length n from a source state (in row) to a destination state (in column).

The reachability of states in a protocol p with $|S_p|$ states may be deduced from the logical sum of n-reachability where $n \in [1, |S_p| - 1]$. Indeed, the reachability of a given state s from a state s' means the existence of at least one path from s' to s. Moreover, the longest path going through all states only once has a maximal length of $|S_p| - 1$. Therefore, a state s is reachable from state s' iff it exists at least one path from state s' to s, of length less or equal to $|S_p| - 1$, i.e. $\Pi_p[ij] \neq 0$ □

.

4.3 Algorithm for Structural Validation

For a given protocol p, conditions 1. and 2. presented in Sect. 3.2 may be checked with the following algorithm:

1. Sort the states $s \in S$ from starting states to ending ones as a sorted states list $\Sigma_p = \{\sigma_i \in S_p\}$ with $i \in \{1, \ldots, a, \ldots, b, \ldots, |S_p|\}$ such that $\forall i \in [1, \ldots, a]$, σ_i are starting states, and $\forall i \in [b, \ldots, |S_p|]$, σ_i are ending states;
2. Compute the transition cardinality matrix $\Theta_{p,||}$ according to the sets of local behavioral units and the sorted states list Σ_p;
3. Compute the path cardinality matrix $\Pi_{p,||} = \sum_{n=1}^{|S_p|-1} \Theta^n_{p,||}$;
4. Condition 1. is fulfilled $\Leftrightarrow \forall j \in]a, |S_p|], \exists i \in [1, a]$ such that $\Pi_{p,||}[ij] > 0$.
5. Condition 2. is fulfilled $\Leftrightarrow \forall i \in [1, b[, \exists j \in [b, |S_p|]$ such that $\Pi_{p,||}[ij] > 0$.

5 Examples of Structural Validation

In this section, the validity of three protocols is checked to illustrate the algebraic representation of social protocols and the algorithm presented above. In the presented examples, it is assumed that the protocols are semantically valid and that they fulfill the third condition for structural validity. For the three protocols, conditions 1. and 2. for structural validity are checked with the algorithm presented in Sect. 4.3.

In all presented protocols, the states are assumed to be already sorted to improve the readability of the paper. Therefore, the first step of the algorithm for structural validation may be skipped.

5.1 Example of Valid Social Protocol

In Fig. 2, a first example of social protocol is presented. Starting states – σ_1 and σ_2 – are represented by a double circle, while ending states – σ_5 and σ_6 are represented by a bold circle. Transitions are represented as arrows.

For the protocol presented in Fig. 2, the transition cardinality matrix is the following one:

$$\Theta_{p,||} = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

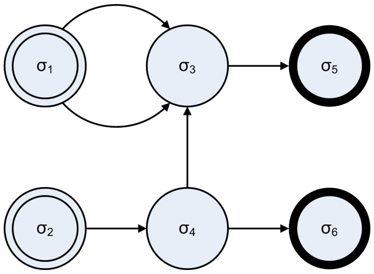

Fig. 2. Example of valid social protocol

As the protocol contains six states, the path cardinality matrix $\Pi_{p,||}$ is the sum of the powers of the transition cardinality matrix from power 1 to power 5, i.e $\Pi_{p,||} = \sum_{n=1}^{|S_p|-1} \Theta_{p,||}^{n} = \sum_{n=1}^{5} \Theta_{p,||}^{n} = \Theta_{p,||} + \Theta_{p,||}^{2} + \ldots + \Theta_{p,||}^{5}$

By a simple computation, $\Pi_{p,||} = \begin{bmatrix} 0\,0\,2\,0\,2\,0 \\ 0\,0\,1\,1\,1\,1 \\ 0\,0\,0\,0\,1\,0 \\ 0\,0\,1\,0\,1\,1 \\ 0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0\,0 \end{bmatrix}$

In Fig. 3, the area of the path cardinality matrix to be checked for the first condition of structural validity is highlighted. This area consists of all elements whose row number is lower or equal than $a = 2$ and whose column number is greater than $a = 2$. If in each column of this area it exists at least one element whose value is greater than 0, the first condition is fulfilled. In this first protocol, the first condition is fulfilled.

$$\Pi_{p,||} = \left[\begin{array}{cc|cccc} 0 & 0 & \mathit{2} & \mathit{0} & \mathit{2} & \mathit{0} \\ 0 & 0 & \mathit{1} & \mathit{1} & \mathit{1} & \mathit{1} \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

Fig. 3. Area to be check for the first condition of structural validity

In Fig. 4, the area of the path cardinality matrix to be checked for the second condition of structural validity is highlighted. This area consists of all elements whose row number is lower than $b = 5$ and whose column number is greater or equal to $b = 5$. If in each row of this area it exists at least one element whose value is greater than 0, the second condition is fulfilled. In this first protocol, the second condition is fulfilled.

$$\Pi_{p,||} = \left[\begin{array}{cccc|cc} 0 & 0 & 2 & 0 & \mathit{2} & \mathit{0} \\ 0 & 0 & 1 & 1 & \mathit{1} & \mathit{1} \\ 0 & 0 & 0 & 0 & \mathit{1} & \mathit{0} \\ 0 & 0 & 1 & 0 & \mathit{1} & \mathit{1} \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

Fig. 4. Area to be check for condition 2. of structural validity

5.2 Example of a Social Protocol Violating the First Condition

In Fig. 5, a second example of social protocol is presented. This protocol is similar to the protocol presented in Sect. 5.1. The only difference is that the transition from state σ_2 now leads to σ_1 instead of σ_4.

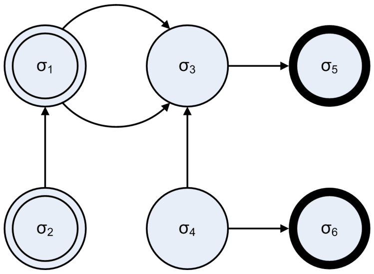

Fig. 5. Example of a social protocol violating the first condition

For the protocol presented in Fig. 5, the transition cardinality matrix is the following one:

$$\Theta_{p,||} = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & \mathbf{0} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

By a simple computation, $\Pi_{p,||} = \left[\begin{array}{cc|cccc} 0 & 0 & 2 & \mathbf{0} & 2 & \mathbf{0} \\ 1 & 0 & 2 & \mathbf{0} & 2 & \mathbf{0} \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$

As it may easily be notice on Fig. 5, this protocol does not fulfill the first condition of structural validity because states σ_4 and σ_6 are unreachable from the starting states. The analyzis of the path transition matrix leads to the same conclusion: in the area to be checked for the fist condition, the only value for states σ_4 and σ_6 is 0.

5.3 Example of a Social Protocol Violating the Second Condition

In Fig. 6, a third example of social protocol is presented. This protocol is similar to the protocol presented in Sect. 5.1. The only difference is that the transition to state σ_5 now comes from σ_4 instead of σ_3.

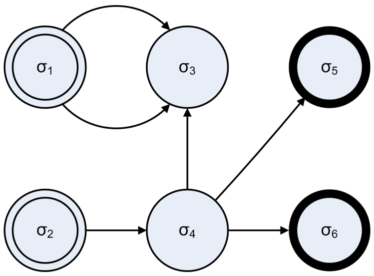

Fig. 6. Example of a social protocol violating the second condition

For the protocol presented in Fig. 6, the transition cardinality matrix is the following one:

$$\Theta_{p,||} = \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{0} & 0 \\ 0 & 0 & 1 & 0 & \mathbf{1} & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

By a simple computation, $\Pi_{p,||} = \left[\begin{array}{cccc|cc} 0 & 0 & 2 & 0 & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 1 & 0 & 1 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$

As it may easily be notice on Fig. 6, this protocol does not fulfill the second condition of structural validity because no ending state may be reached from states σ_1 and σ_3. The analyzis of the path transition matrix leads to the same conclusion: in the area to be checked for the second condition, the only value for states σ_1 and σ_3 is 0.

6 Related Works

As process modeling is concerned, many works have already been conducted in the research field of workflow modeling and workflow management systems.

Many works [4,5,6,7,8,9,10,11] have focused on formal models and conditions under which a modification of an existing – and potentially running – workflow retains workflow validity. However, to our best knowledge, current works concerning workflow adaptation focus on interactions the importance of social aspects, are not or insufficiently taken into account by these works.

Some interesting works have been done in the field of electronic negotiations to model electronic negotiations with the help of negotiation protocols. In [12], it is stated in that, in the field of electronic negotiations, "the protocol is a formal model, often represented by a set of rules, which govern software processing, decision-making and communication tasks, and imposes restrictions on activities through the specification of permissible inputs and actions". One may notice the similarity with the concept of social protocol. The reason for this fact is that the model presented in this paper was originally coming from a work on protocols for electronic negotiations [13]. However, to our knowledge, none of the works concerning negotiation protocols provides mechanisms for protocol validation. Moreover, these works are by nature limited to the field of electronic negotiations which is just a subset of the field of human collaboration.

7 Conclusions

While many works are currently done on modeling collaboration processes in which software entities (agents, web services) are involved, modeling collaboration processes in which mainly humans are involved is an area that still requires much attention from the research community. Some of the main issues to be addressed are the social aspects of collaboration and the adaptation capabilities of humans. In this paper the first issue is addressed. The concept of social protocol aims at being a start of answer to the question of computer support for social collaboration. The algorithm for structural validation of social protocols presented in this paper provides protocol designers and/or software supporting social protocol with means of checking the validity of social protocols, which leads to more robust support for social protocols.

The main innovations presented in this paper are 1) the algebraic representation of social protocols , 2) the algorithm for structural validation of social protocols based on their algebraic representation. The proposed concepts have been fully implementated in the *DynG* protocol [14], a social protocol-based platform.

The validation of social protocols is a requirement for 1) the design of robust collaboration models, 2) more advanced support for human-to-human collaboration. Among advanced features, the adaptation of social protocol – i.e. the possibility to modify a collaboration process and its associated social protocol at run-time – is necessary to weaken constraints usually limiting the interaction between collaborators, so that the adaptation capabilities of humans may be integrated in the life of a social protocol. With support of social protocol adaptation, methods for validation of adapted social protocols extending the algorithm presented in this paper are still to be proposed.

References

1. Picard, W.: Computer support for adaptive human collaboration with negotiable social protocols. In Abramowicz, W., Mayr, H.C., eds.: Proc. of the 9^{th} International Conference on Business Information Systems. Volume 85 of LNI., GI (2006) 90–101
2. Picard, W.: Modeling structured non-monolithic collaboration processes. In Camarinha-Matos, L., Afsarmanesh, H., Ortiz, A., eds.: Collaborative Networks and their Breeding Environments, Proc. of the 6^{th} IFIP Working Conference on Virtual Enterprises (PRO-VE 2005), Valencia, Spain, Springer (September 2005) 379–386
3. Picard, W.: Towards support systems for non-monolithic electronic negotiations. the contract-group-message model. Journal of Decision Systems: Special Issue on Electronic Negotiations - Models, Systems and Agents **13** (2004) 423–439
4. van der Aalst, W.M.P.: The Application of Petri Nets to Workflow Management. The Journal of Circuits, Systems and Computers **8**(1) (1998) 21–66
5. van der Aalst, W.M.P., Basten, T., Verbeek, H.M.W., Verkoulen, P.A.C., Voorhoeve, M.: Adaptive workflow: On the interplay between flexibility and support. In Filipe, J., ed.: Proc. of the 1^{st} International Conference on Enterprise Information Systems. Volume 2., Setúbal, Portugal, Kluwer Academic Publishers (March 1999) 353–360
6. van der Aalst, W.M.P.: Workflow verification: Finding control-flow errors using petri-net-based techniques. In Aalst, W., Desel, J., Oberweis, A., eds.: Business Process Management, Models, Techniques, and Empirical Studies. Volume 1806 of Lecture Notes in Computer Science., Springer (2000) 161–183
7. Sadiq, W., Orlowska, M.E.: Analyzing process models using graph reduction techniques. Information Systems **25**(2) (2000) 117–134
8. Sadiq, S.W., Orlowska, M.E., Sadiq, W., Foulger, C.: Data flow and validation in workflow modelling. In Schewe, K.D., Williams, H.E., eds.: Proceedings of the 15^{th} Australasian Database Conference, ADC 2004. Volume 27 of CRPIT., Australian Computer Society (2004) 207–214
9. Sadiq, S.W., Orlowska, M.E., Sadiq, W.: Specification and validation of process constraints for flexible workflows. Information Systems **30**(5) (2005) 349–378
10. ter Hofstede, A.H.M., Orlowska, M.E., Rajapakse, J.: Verification problems in conceptual workflow specifications. Data Knowledge Engineering **24**(3) (1998) 239–256
11. ter Hofstede, A.H.M., Orlowska, M.E., Rajapakse, J.: Verification problems in conceptual workflow specifications. In Thalheim, B., ed.: Conceptual Modeling - ER'96, 15th International Conference on Conceptual Modeling, Cottbus, Germany, October 7-10, 1996, Proceedings. Volume 1157 of Lecture Notes in Computer Science., Springer (1996) 73–88
12. Kersten, G.E., Strecker, S.E., Lawi, K.P.: Protocols for electronic negotiation systems: Theoretical foundations and design issue. In: Proc. of the 5^{th} Conference on Electronic Commerce and Web Technologies (ECWeb04), Sarragoza, Spain, IEEE Computer Society (2004)
13. Picard, W., Huriaux, T.: Dyng: A protocol-based prototype for non-monolithic electronic collaboration. Lecture Notes in Computer Science **3865**(CSCW in Design 2005) (2006) 41–50
14. Huriaux, T., Picard, W.: Dyng: a multi-protocol collaborative system. In Funabashi, M., Grzech, A., eds.: Proc. of the 5^{th} IFIP International Conference on

e-Commerce, e-Business, and e-Government (I3E 2005), Poznań, Poland (2005) 591–605