Multi-Level Delegation for Flexible Business Process Modeling

Oumaima Saidani, Université Paris 1 – Panthéon – Sorbonne, Centre de Recherche en Informatique, 90, rue de Tolbiac, 75634 Paris cedex 13, France; E-mail: Oumaima.Saidani@malix.univ-paris1.fr

Selmin Nurcan, IAE de Paris, Sorbonne Graduate Business School, Université Paris 1 – Panthéon – Sorbonne, 21, rue Broca, 75005 Paris, France; E-mail: Selmin.Nurcan@univ-paris1.fr

ABSTRACT

In this paper we address issues related to delegation of responsibilities and their importance in increasing the flexibility and the effectiveness of business processes. Organizations usually establish a set of business rules regulating the way business processes are managed. For example, they specify which user should perform a given work in a given situation. In a changing and highly dynamic environment, rules can not be planned in advance in a fine-grained level. What's more, in actual circumstances users may delegate work assigned to them; indeed, it is not always possible to account for every responsibility required in a moving environment. These delegation activities should be controlled. In addition, unforeseen circumstances like absence could take place. That is, we introduce a delegation model in order to allow the predefined rules to be less exhaustive and the decision-taking mechanism to be decentralized, to control the delegation activities between actors and to take into account unforeseen circumstances.

1. INTRODUCTION

In current distributed and dynamic environments, the goal of companies is to well and quickly meet with customers' requirements. In some cases, developing complete rules specifying exhaustively how actors will proceed is inaccurate, because this limits their autonomy and efficiency when changes make some predefined conditions inapplicable.

Instead of developing fully business policies, we propose to provide the ability of delegation provided that it is controlled by an effective model. There are many requirements that may drive to provide delegation capabilities:

- Making easier process management by decentralizing the control and the decision-making and allowing actors to be more autonomous and confident.
- Collaborative work in human organizations requires the use of delegation as natural and useful way to cooperate. Actors could wish to cooperate in a project.
- Responsibilities may conflict, and specific policies may require that an actor delegate some of his duties in order to separate conflicting duties.
- An actor may lack resources (e.g. time, equipment) essential for achieving his responsibilities.
- Unforeseen circumstances, such as unplanned absences (illness, leaves), may require to change actors.
- Substitution: in some situations like business mission, the employee needs to delegate the achievement of the responsibility he ensures to another employee.

To deal with these requirements, we introduce a multi-level delegation in order to make business rules less complex and flexible, processes more efficient, and management and control more flexible.

In this paper, we address issues related to delegation of pieces of responsibilities and their importance in increasing flexibility of business processes. Selecting some parts of responsibilities gives the delegator great flexibility in choosing which work he/she wants to delegate.

By granting autonomy to actors and allowing them to delegate and to decide which parts of responsibilities they want to delegate, the development of the business rules by the manager is greatly simplified. In fact, the decision-making and the process control will be distributed between him/her and the other actors of the organization. Thus, the process manager has to define significant rules on a coarse-grained level without seeing details whereas actors that are allowed to delegate define exhaustively how tasks should be achieved and by whom. Therefore, this approach satisfies the process manager requirements as well as ones of the participating actors. The process manager will have less complex rules to handle (this fact is time-saving) and actors will be more autonomous and confident.

The paper is organized as follows: in section 2 we discuss related work and present our contributions. In section 3, we introduce a delegation model for flexible business process modeling, we provide a meta-model and we illustrate our model with a case study. Section 4 concludes the paper.

2. BACKGROUND AND MOTIVATION

The literature provides a considerable work dealing with various aspects of delegation. Delegation can take many forms. Gasser and McDermott [5] address user-to-machine delegation; they define it as "the process whereby a user in a distributed environment authorizes a system to access remote resources on his behalf". Henry and Gladny [7] deal with machine to machine delegation; they consider requirements for a digital library that emulates massive collections of physical media for clerical, engineering, and cultural applications. Nagaratnam and Lea [9] discuss process-to-process delegation in the distributed object environment. Sandhu et al. [11] address delegation among the role administrators. Delegation addressed in [2], [6], [7], [8], [9] and [11] is related to rights in a security context. Schaad and Moffett [1] address delegation of obligation and authorizations. Becker et al. [3] deal with delegation in distributed software process management allowing a client in interorganizational development processes to delegate parts of net-based process models to contractors.

Delegation addressed in most common work is unconstrained and without any conditions, that is not convenient and may cause frustration. In addition, delegation is often defined as a substitution mechanism of all or a subset of actor's roles to one or more other actors such as in [2], or the ability of a user to delegate to another user some permissions related to a role [2] or single tasks [4].

Nevertheless, in some cases an actor needs to delegate only some functions held by his/her role. Furthermore, in some cases, role-based delegation is required. For instance, if the "loan manager" is absent, loan manager's responsibilities can be delegated to other employees based on their capabilities (roles) rather than their identities (individual actors). For instance, "Offer_preparing" can be delegated to the role "loan manager's assistant".

In this paper we focus on human delegation where a user delegates a part of his responsibilities to another user. This can be done directly or through membership to roles. These issues of delegation were informally discussed previously in [10]. We identified three main kinds of delegation: actor-to-actor, actor-to-role and role-to-role delegation. Each of them can be based on roles, functions and/or operational goals. In this paper, we study in depth these issues. To the original work, we add some extensions which enhance the effectiveness and the totality of our approach.

In summary, our research provides the following contributions:

- Delegation addressed in this paper can be based not only on functions, roles and operational goals as proposed in [10], but also on more coarse-grained

responsibilities (business processes and business goals) as required. This kind of delegation requires more confidence and autonomy of the delegatee.
- Current approaches focus only on Who, What and Whom facets of the delegation and omit the Why and How ones. In this paper, we discuss also the latter ones. The Why facet controls and justifies the delegation activities.
- We provide a meta-model including all concepts useful to define delegation capabilities, and we formalize their semantics by the means of formal logic.

This paper is the first attempt to model delegation of multi-level responsibilities in the context of business process flexibility.

3. MULTI-LEVEL DELEGATION MODEL

In order to satisfy the flexibility requirements related to business process modeling, we introduce in this section a new delegation model based on multi-level responsibilities. The proposed model reuses basic concepts introduced in [10] and extends them with new concepts. We will first present a summary of the basic concepts; then, we introduce the new concepts related to delegation. Examples used throughout the paper for illustrating concepts concern the loan handling process in a bank.

3.1. Overview of the Role-Based Approach for Modelling Flexible Business Processes

The central concepts in our approach are the role and the function. A role can represent competency to realize particular functions, e.g. "an engineer", and can embody authority and responsibility, e.g., "a project supervisor". A role can be responsible for the achievement of a business_process (BP) or a business goal. A function (i) is a collection of operational goals satisfied by performing operations, (ii) is hold by one role, and is a part of a BP. An organization is structured as a network of BPs in order to achieve business_goals. A business_goal is achieved by performing a BP which comprises many functions. An actor belongs to organizational_units, can play several roles based on his responsibilities and qualifications and performs functions specifying work steps in a BP. Organizational_units can be firm's branches or describe firms collaborating to achieve common processes.

$ACTORS,\ ROLES,\ FUNCTIONS,\ O\_GOALS,\ OPERATIONS,\ BPS,\ B\_GOALS,\ ORG\_UNITS$

define sets of actors, roles, functions, operational goals, operations, business processes, business goals and organizational units, respectively.

Let

$a \in ACTORS,\ r,\ r_1,\ r_2 \in ROLES,\ f \in FUNCTIONS,\ op \in O\_GOALS, o \in OPERATIONS,\ bp \in BPS,$
$bg \in B\_GOALS,\ ou \in ORG\_UNITS$

$Can\_play(a,r)$ means that a can play r.
$Comprises(f, op)$ means that f comprises op
$Satisfies(op, o)$ means that op is satisfied by achieving o.
$is\_responsible(r, bg)$ means that r is responsible for the achievement of bg.
$is\_responsible(r, f)$ means that r is responsible for the performance of f.
$Participates(r, f)$ means that r participates in the achievement of f by performing some operations satisfying operational goals of f.
$Comprises(bp, f)$ means that bp comprises f.

3.2. Facets of the Multi-level Delegation Model

We now introduce the supplementary components related to delegation. As mentioned in Section 2, we define five facets of the delegation capturing these questions:

- Who delegate the responsibility?
- To Whom the responsibility is delegated?
- What is the delegated responsibility?
- Why delegation takes place?
- How delegated work should be performed?

These facets are represented respectively by the entities: *Delegator, Delegatee, Responsibility, Context, Instructions.*

An actor can delegate parts of his responsibilities to another actor which performs these responsibility parts; this can be done directly or through membership to roles. Delegation is controlled by the means of the relation Can_delegate: a 5-tuple with five attributes representing the five facets. $Can\_delegate(dtor, dte, resp, c, i)$, means that dtor (or the members of dtor if dtor is a role) can delegate the responsibility resp to dtee (or the members of dtee if dtee is a role) in the context c, forcing the achievement instructions i. Throughout the paper we will use the terms dtor, dte, resp, c and i to denote the actors/roles involved in the delegation (delegator and delegatee respectively), the responsibility to delegate, the context of the delegation, and recommendations or directives to be provided by the delegator to the delegatee.

An actor can take part in more than one delegation in different roles. In addition we extend the relation Can_hold [10] by two new relations between the entities Role and Function, which are Is_responsible and Participates. Similarly, two new relations are defined: Is_responsible, between the entities Role and Business_process, and between Role and Business_goal respectively. We will discuss these concepts.

- Who and whom facets: Our model supports individual delegation as well as role-based delegation. The delegator and the delegatee can be either actors or roles. Then, it is possible to define role-to-role, actor-to-actor and actor-to-role delegations. We assume that it is no significant to define role-to-actor delegation.
- What facet: The delegated responsibility can be at different levels of refinement (operational_goal, function, role, BP, business_goal).
- Why facet: The fourth facet is the context of delegation which answers the question "Why the responsibility is delegated". Context can be: unplanned absence, illness, leave, collaborative work, saving of time, lack of resources, decentralization of work, conflict of duties, etc. Responsibilities that an actor can delegate to another actor can differ depending of the context of delegation. For instance, a loan manager can delegate the function "Offer_validating" to his assistant in the context of "Lack_of_resources" or "Absence", but not in the context of "Conflict_ of_duties".
- How face: The last facet defines the way of specifying "How to achieve the delegated responsibility?" The delegator has to provide it to the delegatee. Confidence allowed to the delegatee depends on his/her competency, autonomy and experience. Delegation requires that the delegatee has sufficient experience and capacity to perform work. It may also give the delegatee some new responsibilities. The delegatee is responsible for performing the delegated work. The delegator is responsible for ensuring that the work was well carried out.

There are several ways of delegating a work. The delegatee may have no autonomy; he/she has to precisely follow delegator directives for achieving the delegated work. The delegator can give some recommendations but it is in the responsibility of the delegatee to decide how the delegated work is fulfilled. The delegator can also delegate a work without any recommendation, this requires a high level of confidence and analysis on behalf of the delegatee which has total autonomy, and he/she decides and acts without contacting the delegator.

For example, if the "Loan_manager" delegates the function "Loan_handling" to the "Loan_assistant", he/she can precise some recommendations for its achievement.

Recommendations and directives are defined by the delegator. They can be fine or coarse-grained, planned or ad-hoc. If they are coarse-grained, the delegatee has to enforce and refine them.

3.3. The Meta-Model of Delegation

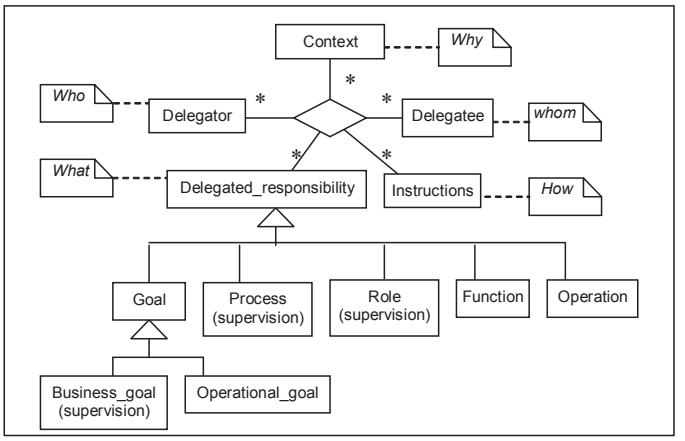
The meta-model of our delegation framework is represented by an UML diagram in Figure 1.

We represent now the delegation formally:

$DELEGATORS, DELEGATEES, D\_RESPONSIBILITIES, CONTEXTS, INSTRUCTIONS$

are set of delegators, delegatees, delegated responsibilities, contexts and ways.

Figure 1. Meta-model of delegation



$DELEGATORS \subseteq ACTORS \cup ROLES$, $DELEGATEES \subseteq ACTORS \cup ROLES$,

$D\_RESPONSIBILITIES \subseteq B\_GOALS \cup \beta\_GOALS \cup ROLES \cup FUNCTIONS \cup \theta\_GOALS \cup OPERATIONS$

$INSTRUCTIONS = \{f, \text{“Recommendations”}, \text{“Directives”}\}$

We now represent formally rules expressed by our model:

$\forall r, r_1, r_2 \in ROLES, f \in FUNCTIONS, \rho\_GOALS, o \in OPERATIONS, \forall dtor \in DELEGATORS,$
$dtee \in DELEGATEES, resp \in D\_RESPONSIBILITIES, \forall c \in CONTEXTS, w \in WAYS, i \in INST$

Hypothesis 1
If dtor can delegate f to dtee, then dtor can delegate any operational_goal comprised in f to dtor, he/she can also delegate any operation associated with that operational_ goal to dtee.

$Can-delegate(dtor, dte, f, c, i) \wedge Comprises(f, \rho\_GOALS) \rightarrow Can-delegate(dtor, dte, \rho\_GOALS, c, i)$
$Can-delegate(dtor, dte, f, c, i) \wedge Comprises(f, \rho\_GOALS) \wedge satisfies(o, \rho\_GOALS) \rightarrow Can-delegate(dtor, dte, o, c, i)$

Hypothesis 2
If dtor can delegate his/her responsibility of achieving bp to dtee, then dtor can delegate any function f comprised in BP to dtee. He/she can thus, according to hypothesis 1, delegate to dtee any operation and any operational_ goal associated to f.

$Can-delegate(dtor, dtee, \beta\_GOALS, c, i) \wedge comprises(\beta\_GOALS, f) \rightarrow Can-delegate(dtor, dtee, f, c, i)$

Hypothesis 3
If dtor can delegate bg to dtee, then dtor can delegate to dtee any bp reaching bg. He/she can thus, according to hypothesis 2, delegate to dtee any function associated to bp.

$Can-delegate(dtor, dte, \beta\_GOALS, c, i) \wedge reachs(\beta\_GOALS, \rho\_GOALS) \rightarrow Can-delegate(dtor, dte, \rho\_GOALS, c, i)$

Hypothesis 4
We suppose that role-to-actor delegation is not possible and express this rule as follows:

$\neg Can-delegate(dtor, dte, resp, c, i) \wedge dtor \in ROLES \wedge dte \in ACTORS$

Examples of instantiation of the model:

$Can-delegate(\text{"}loan\_manager\text{"}, \text{"}loan\_assistent\text{"}, \text{"}loan\_handling\text{"},$
$saving\_\beta\_time\text{"}, \text{"}recommendations\text{"})$

means that any actor who is member of the role “Loan_manager” can delegate the function “Loan_handling” to any actor who is member the role “Loan_ assistant” in the context of “saving of time” and the delegator has to provide recommendations to the delegatee.

3.4. The Delegation Process

In this section, we specify the different steps of a delegation. The delegator has to specify with respect to the predicate can-delegate (i) the responsibility to be delegated, (ii) the context of the delegation, (iii) a list of delegatees being able to receive the delegation in the determined context. The choice of the most appropriate delegatee is in the discretion of the delegator; it depends on the importance of the delegated responsibility and the delegatee competency. The selected delegatee should agree to receive the delegation. Once the delegatee is selected, both the delegator and the delegatee will agree on control and coordination methods which depend on the delegatee confidence, competency and autonomy.

Then, the delegator defines the directives or recommendations required for the enactment of the delegated responsibility. The refinement level of these recommendations or directives depends on the kind of the responsibility and the delegatee competency. Particular responsibilities should be carefully delegated. The delegatee has to refine the delegated responsibility following the delegator recommendations or directives. Some responsibilities require to be refined by the delegatee because each individual has his specific method to realize given tasks.

Supervising the delegation : in some cases, the delegator has to supervise the progress states of the delegated parts of the responsibility. In other cases, the delegatee does not have to return to the delegator the intermediary inputs and outputs associated with the delegated responsibility. Thus the delegatee has to contact the delegator when the delegated responsibility is achieved.

In all cases, once the delegated responsibility is achieved, the delegator can revoke the delegation and have to measure the quality of the achieved work. We represent in Figure 2 the algorithm capturing the main phases of the delegation process. Metrics can be used to measure the variance between the required results and the obtained ones. Then, the delegator can decide to improve the obtained results. These aspects are out of the scope of this paper and will be discussed in a future work.

3.5. Illustration

For illustrating the semantics of our model, we use the case of a loan handling process in a bank.

Figure 2. Algorithm of delegation

Algorithm1. The delegation process

INPUT: $dtor_i \in DELEGATORS, resp_j \in D\_RESPONSIBILITIES, c_k \in CONTEXTS$

Let *Establish-dtee* a function which, for a given 3-tuple $(dtor_i, resp_j, c_k)$ return a sorted list $\{(dte_l, inst_m)\}$ such that $dte_l \in DTE, inst_m \in INST$ and $can-delegate(dtor_i, dte_l, resp_j, c_k, inst_m)$.

BEGIN
$List-dte \leftarrow Establish-dte(dtor_i, resp_j, c_k)$
if $List-dte \neq \emptyset$
  $path \leftarrow 1$
  while $path \neq 0$ **do**
    If $agree(List-dtee[path])$
    $path \leftarrow 0$
    endif
  endwhile
endif

if $List-dte = \emptyset$ or $path = length(List-dte)$
$Success-delegation \leftarrow false$

Figure 3. Examples of assignations

Actor	Role
Jane	Customer
Maria	Loan_ assistant
Steve	Loan_ assistant
Smith	Financial_responsible
Ravi	Commercial_responsible
George	Loan_manager
Alexandra	Loan_manager
John	Agent

***Can_play* relation - examples of assignations**

Role	Function
Customer	Loan_request_submitting
Loan_manager	Loan_handling
Financial_responsible	Financial_evaluation
Commercial_responsible	Commercial_evaluation

***Is-responsible* relation - examples of assignations**

Role	Function
Loan_manager	Loan_Handling
Loan_assistant	Loan_handling
Agent	Loan_request_submitting
Agent	Loan_handling

***Participates* relation - examples of assignations**

Function	Operational_goal
Loan_handling	Loan_request_handling
	Final _evaluation
	Offer_preparing
Financial_evaluation	Financial_evaluation_preparing
Commercial_evaluation	Commercial_evaluation_preparing

***Comprises* relation - examples of assignation**

Operational_goal	Operation
Loan_request_handling	Customer_interviewing
	Loan_request_registring
Offer_preparing	Offer_drafting
	Letter_sending
Financial_evaluation_preparing	Customer's_guarantee_checking
	Internal_financial_situation_checking
	Financial_evaluation_drafting

***Satisfies* relation - examples of assignations**

A customer's loan request is accepted only if its features are compatible with the financial and commercial strategies and interests of the bank.

The process starts by a customer loan request, then, an agent registers this request, which will be next evaluated by both the financial and commercial departments. The first evaluation is performed by the financial person responsible and involves financial aspects related to both the customer and the bank, for instance, the guarantees provided for refunding the loan. The commercial evaluation is performed by the commercial responsible and involves commercial aspects like the possibility to acquire new regular customers and is performed by a loan manager.

Basing on the commercial and the financial evaluation, the loan manager performs the final evaluation and he may reject the request; in this case an agent writes the refusal letter. He/she may also propose a counterproposal which will be prepared by the loan manager's assistant. He/she may as well accept the request, then the loan manager's assistant establishes a complete proposition including the duration, the amount, the interest rate, the refunding instructions, etc. We provide some assignation examples in Figure 3.

Roles involved in this process are:

$$"Customer",\ "Financial\_responsible",\ "Commercial\_responsible",$$
$$"Loan\_manager",\ "Loan\_assistent",\ "Agent"$$

In the following, we present a possible instantiation of the delegation model.

$$BUSINESS\_PROCESSES = \{"Loan\_handling"\}$$

$$ACTORS = \{"Jane","John","Maria","Steve","Smith","Georges","Alexandra","Ravi"\}$$

$$ROLES = \begin{Bmatrix} "Customer","Agent","Loan\_manager","Loan\_assistent","Financial\_responsible", \\ "Commercial\_responsible" \end{Bmatrix}$$

$$FUNCTIONS = \begin{Bmatrix} "Loan\_request\_submitting","Loan\_handling", \\ "Financial\_evaluation","Commercial\_evaluation" \end{Bmatrix}$$

$$\theta\_GOALS = \begin{Bmatrix} "Loan\_request\_handling","Commercial\_evaluation\_preparing", \\ "Financial\_evaluation\_preparing","Commercial\_evaluation\_preparing" \end{Bmatrix}$$

$$OPERATIONS = \begin{Bmatrix} "Customer\_interviewing","Loan\_request\_registring", \\ "Customer's\_garantee\_checking","Internal\_financia\_situation\_checking", \\ "Financial\_evaluation\_drafting","Condition\_evaluating", \\ "Couterproposal\_drafting","refusal\_letter\_drafting","Offer\_drafting" \\ "Letter\_sending" \end{Bmatrix}$$

$$Can\_play("Jane","Customer"),$$
$$Can\_play("Smith","Customer"),$$
$$Can\_play("John","Agent"),$$
$$Can\_play("Maria","Loan\_assistent")$$
$$Can\_play("Steve","Loan\_assistent")$$
$$Can\_play("Smith","Financial\_responsible")$$
$$Can\_play("Georges","Loan\_manager")$$
$$Can\_play("Alexandra","Loan\_manager")$$
$$Can\_play("Ravi","Commercial\_responsible")$$

$$Is\_responsible\_for("Customer","Loan\_request\_submitting")$$
$$Is\_responsible\_for("Loan\_manager","Loan\_handling")$$
$$Is\_responsible\_for("Financial-responsible","Financial\_evaluation")$$
$$Is\_responsible\_for("Commercial-responsible","Commercial\_evaluation")$$

$$Participates("Loan\_manager","Loan\_handling")$$
$$Participates("Loan\_assistent","Loan\_handling")$$
$$Participates("Agent","Loan\_handling")$$
$$Participates("Agent","Loan\_request\_submitting")$$

$$Comprises("Loan\_handling","Loan\_request\_handling")$$
$$Comprises("Loan\_handling","Final\_evaluation")$$
$$Comprises("Loan\_handling","Offer\_preparing")$$

$$Satisfies("Loan\_request\_handling","Customer\_interviewing")$$
$$Satisfies("Loan\_request\_handling","Loan\_request\_registring")$$
$$Satisfies("Offer\_preparing","Offer\_drafting")$$
$$Satisfies("Offer\_preparing","Letter\_sending")$$
$$Satisfies("Financial\_evaluation\_preparing","Internal\_financial\_situation\_evaluating")$$
$$Satisfies("Financial\_evaluation\_preparing","Financial\_evaluation\_drafting")$$
$$Satisfies("Financial\_evaluation\_preparing","Customer's\_garantee\_checking")$$

We identify in the following some examples illustrating delegation.

Example 1.

$$Can-delegate("George","Maria","Loan\_manager","saving\_time","Rrecommendations")$$

This is an actor-to-actor delegation which means that "George" can delegate his role "Loan manager" to "Maria" in the context of "time_ saving", he has to give her recommendations.

This means also that "George" can delegate to "Maria" all functions, operational_ goals and operations associated with this role. Thus, "Maria" can perform these responsibilities as well as delegate some of them to other roles/actors participating in the functions of delegated role.

Example 2.

$$Can-delegate("George","Maria","Loan\_handling","Lack\_of\_resources","Directives")$$

This is an actor-to-actor delegation which means that "George" can delegate the function "Loan_handling" to "Maria" in the context of "Lack_of_resources", he has to give her directives.

This means also that "George" can delegate to "Maria" all operational_goals and operations associated with this function.

Example 3.

Can – delegate(" *Loan _ manager*","*Loan _ assistent*","*Laoan _ manager*", "*Urgent _ situation*","*Directives*")

This is a role-to-role delegation which means that any actor member of the role "Loan_manager" can delegate this role to any actor member of the role "Loan_assistant", in the context of "Urgent_situation" and with directives

DISCUSSION

In our example, the "loan_manager" can create new operational_goals or functions if needed. For instance, with reference to his experience, he may judge that throughout the year, in particular periods, instead of delegating all the operations of the operational goal "Loan_request_handling" to an agent, he performs him-self the operation "Customer_interviewing" and only delegate the operation "Loan-request_registring" to an agent. Thus, after interviewing the customer, he may stop the process even before "loan_request_registering" if he judges that it is useless to continue the process and to perform the operation "Loan_request_registring" followed by the operational goal "Financial_evaluation_preparing", etc. knowing that in any way (even if the financial evaluation is positive), the loan request will be rejected. An agent can not take the decision of stopping the loan handling process in a beginning stage; the loan handler can do it. Stopping the process saves the time of the actors participating in the rest of the process including the financial responsible, the decision of the loan manager avoids him to perform a useless financial evaluation knowing that the commercial evaluation will stop the process later.

In other circumstances, the loan manager may prefer that the loan request handling be performed by his assistant rather than an agent, he may judge that the loan manager's assistant may, in the stage of loan request handling, be able to propose a counterproposal if needed, rather than taking this decision later, this is also time saving.

Furthermore, in this example, the business manager has only to specify the process phases on a coarse-grained level (as level process, goal and functions). The "Loan_manager" is responsible for the function "Loan_handling" has to enforce and refine the manager's specification in a lower level using operational_goals and operations.

For instance, the manager can only specify the steps of a process as top-level functions, the role holding each function of the process has to specify its achievement using operational goals and operations. However, if the manager identifies only processes allowing the achievement of a business goal, actors holding the role which is responsible for each business process have to define exhaustively functions representing the process steps. Nevertheless, he can delegate parts of this responsibility to other roles (for instance, specifying the steps as top-level operational goals and operations or the achievement of some function fragments consisting of a number of operational goals including their relationships).

4. CONCLUSION

In this paper we discussed the importance of delegation in flexibility and effectiveness of business processes, and we proposed a multi-level delegation model which supports three types of delegation (actor-to-actor, actor-to-role and role-to-role delegation), and five levels of delegated responsibility (role, function and operational_goal, business_goal and process). The ability to delegate responsibilities greatly simplifies the process management and control by decentralising management and decision-making.

The multi-level delegation proposed in this paper responds to actual requirements related to the decentralisation of decision-making by allowing actors to be more autonomous. Our approach meets also requirements of collaborative work. It resolves problems related to conflict of duties, lack of resources (e.g. time, equipment), unforeseen circumstances, such as unplanned absence (illness, leave) and actors' substitution.

The work presented in this paper is the first attempt to model delegation based on roles, functions, goals and processes.

Delegation mechanisms raise many issues which need further research such as:

- Controlling that delegation is not ill-advisedly used.
- Revocation of delegation.
- Management of delegation.
- delegation in the context of inter-organisational collaborative work
- Tool support.

REFERENCES

[1] Schaad, A., Moffett, J. (2002) Delegation of Obligations. 3rd IEEE International Workshop on Policies for Distributed Systems and Networks.

[2] Barka, E., Sandhu., R. (2000) A role-based Delegation Model and Some Extensions. Proceedings 23rd National Information Systems Security Conference.

[3] Becker, S., Jäger, D., Schleicher, A., Westfechtel. B. (2001) A delegation Based Model for Distributed Software Process Management. In Ambriola, V., ed.: Proceedings 8th European Workshop on Software Process Technology (EWSPT). LNCS 2077, Witten, Springer (2001) 130–144

[4] Dowson, M. (1987) Integrated project support with Istar. IEEE Software, 4(6).

[5] Gasser, M., McDermott, E. (1990) An Architecture for practical Delegation in a distributed System. IEEE Computer Society Symposium on Research in Security and Privacy.

[6] Goh, C., Baldwin, A. (1998) Towards a more Complete Model of Role. Proc. of 3rd ACM Workshop on Role-Based Access Control.

[7] Henry, M., Gladny (1997) Access Control for Large Collections. ACM Transactions on Information Systems, 15(2), 154-194.

[8] Lampson, B., Abadi, M., Burrows, M., Wobber, E. (1992) Authentication in Distributed Systems: Theory and Practice. ACM Transactions on Computer Systems.

[9] Nagaratnam, N., Lea, D. (1998) Secure Delegation for Distributed Object environments. USENIX Conference on Object Oriented Technologies and Systems.

[10] Saidani, O., (2006) Nurcan, S. A Role-Based Approach for Modelling Flexible Business Processes. Workshop on Business Process Modelling, Development, and Support (BPMDS), T. Latour and M. Petit (Eds.), Luxembourg, pp. 111 - 120.

[11] Sandhu, R., Bhamidipati, V., Munawer., Q. (1999) The ARBAC97 Model for Role-Based Administration of Roles. ACM Transactions on Information and System Security, 2(1).