# A Goal Oriented Approach for the Definition of a Business Process Requirement Model

Atsa Etoundi Roger
Department of Computer Sciences,
University of Yaoundé I,
Yaoundé, Cameroon

Fouda Ndjodo Marcel
Department of Computer Sciences,
University of Yaoundé I,
Yaoundé, Cameroon

Atouba Christian Lopez
Department of Computer Sciences,
University of Yaoundé I,
Yaoundé, Cameroon

## ABSTRACT

This paper presents a Goal Oriented Approach for the Definition of a Business Process Requirement Model, taking into account the level of importance and constraints inherent in these requirements. The level of importance of a goal is the credit which the user associates to it, while constraints are non-functional needs related to that which this goal must satisfy. The approach proposed in this article revolves around four major activities: the identification of user requirements; selection of the various goals; transformation of the requirements into knowledge bits and lastly, development of the requirement model. We showed in this paper that this step enabled us to describe in an exhaustive way a business process.

## General Terms

Requirement Modeling

## Keywords

Business Process Modeling; Requirement Engineering; Software Component; Requirement Relevancy; Application Engineering; Requirement representation

## 1. INTRODUCTION

Farida Semmak and Joel Brunet in [9] defined a goal oriented metamodel to specify domain requirements. However, they were not concerned about the difficulty, for users to express their needs properly. [1], [2], [3] state that, it is the source of incomprehension between application designer and users. Moreover, [13] reveals that the user requirements evolve quickly. It thus becomes important to be interested not only in the specific needs of users, but also, the general problems underlying the various needs. Apart from that, most authors in requirement engineering seem to act as though the need were an obvious data to obtain from user. If such were the case, studies [1], [2], [3], [17] and all the panoply of requirement engineering approaches would be senseless. Also, we think that it is necessary to formalize a reference model for the representation or specification of the requirements for the future user. We hope that if this representation is intuitive, users will not have difficulties to understand it and consequently, validating all the identified needs by the software engineers and the requirement model which arises from these needs. We believe that such a representation will make it possible to minimize misunderstanding evoked in [1], [2], [3]; to reconcile the expression of users needs to the formal representation of the said needs; and finally, to integrate the level of importance of the various aspects of the system, as well as the constraints inherent to these requirements. From this, we intend to enrich the representation of a need [9] by a set of attributes which enables us to model the level of importance of a need and the inherent constraints (non functional need) to the latter.

The approach proposed in this article revolves around four principal activities: The eliciting of user requirements, selection of the various goals, and the transformation of the requirement into knowledge bits and lastly, the development of the requirement model from the knowledge bits.

In continuation, our work will be articulated around the four activities of the approach. Nevertheless, we will start by presenting the basic concepts of the approach; then we will present our approach activity after activity according to the order in the preceding paragraph, and we will end with the conclusion and future work.

## 2. BASIC CONCEPTS

Our concern, in this section, is to define the concepts necessary in comprehension of the formal representation of a need expressed by a user.

### 2.1 User Requirement

We call requirement of a staff in an organization, an activity in the business process of the said organization, assigned to this staff and of which he would like to have automated in the future computing system. To model a need, it should be identified as a prerequisite. This is why we propose the elaboration of a form to identify user requirements.

### 2.2 Eliciting of Requirements

The document for requirement eliciting is a form to be submitted to users in order to collect from them, the literal description of their expectations. In each form, only an expectation of a staff is described. The form in question is structured as follows:

- Structure or service : describes the structure of the organization, where the staff is assigned;
- staff : represent the name of the staff who is expressing the requirement;
- priority of expectation : the priority of expectation associates a level of importance to that expectation;
- Goal : the usage intention of the user;
- rules : represent the business rule of the intention of use;
- Constraints: constraints indicate non-functional expectations which could impede the realization of user goal. They could be linked with the man-machine interface, security, etc;
- Domain of expectation: domain of expectation describes the context in which the user usage intention is expressed. This field is filled by the software engineer;
- Status of expectation: it indicates the expectation state. it can take one of the following values: proposed, rejected, validated, taken into account.

The inventory of user needs makes it possible to identify in an exhaustive way user expectations. Expectations listed at the level of the users are by default in a proposed state. They pass to a rejected state if user expectations were not approved by a senior staff in rank and if not validated. The taken into account state indicates that an expectation was taken into account in the user requirements specification model. In a formal way, we will represent a user requirement $b$ by $b: (\upsilon, \chi, \nu, \gamma, \lambda, \delta, \psi, \varsigma)$ where $\upsilon$ denotes the structure to which the user is assigned, $\chi$ denotes the name of the user who expressed an expectation, $\nu$ denotes the level of importance of the user usage intention $\gamma$ the users' usage intention, $\lambda$ business $\delta$ constraints, $\psi$ the domain of the expectation, and $\varsigma$ the status of an expectation.

## 2.3 Domain

A domain is the field of application of the users' usage intention or simply the context in which an expectation is defined or applies.

## 2.4 Constraints

The constraints indicate the non-functional expectations which could impede the realization of the users' goal. It represents the state of the environment in which the task will be carried out.

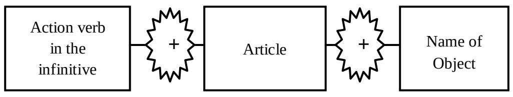
## 2.5 Goal

Definition 1: functional goal

A functional goal defines a requirement, expectation that the system can satisfy, it expresses what the user of the system would wish to do [9].

We deduce from this definition, that a functional goal is simply a text expressing the usage intention of an unspecified user. This implies the use of the active voice and action verbs. Consequently, there is existence, in the text, of a group of words translating the action which is likely to be accomplished by the user on the one hand, and on the other hand, the existence of a group of objects which are subjected to the action of this user; and if possible, the anticipated end result. It is stated in [14] that "an action verb expresses what the subject does or is subjected to" and [15] informs us that the infinitive is "the simplest form of a verbal expression". We shall chose to represent an action translated by the group words previously mentioned, by an action verb in its infinitive form. This gives us the following representation of the goal:

Rule (1) :

Goal:


(1)

- Possible articles are: either the definite articles (the) or indefinite article (a, an);
- Name of object or nominal group which represents either a set of entrants of the same nature as the system; either a set of artifacts of the same nature as the system. It is derived from the name of knowledge bits which encapsulates the goal.

In a formal manner, a goal $b$ will be represented by: $b: (\vartheta,\ \Delta_b(\tau),\ \tau)$ where :

$$\vartheta\ is\ a\ action\ verb\ in\ the\ infinitive$$
$$\tau\ is\ the\ name\ of\ the\ objet$$
$$\Delta_b(\tau) is\ the\ article\ of\ \tau$$

Definition 2 : consistent goal

We will say that a goal is consistent when its expression is in conformity with the rule (1).

Definition 3 : inconsistent goal

A goal is inconsistent when its expression is not in conformity to the rule (1).

Lemma (1): Any inconsistent goal can be transformed into a consistent goal.

Proof: By definition, a goal is a text translating the usage intention of a user. From [12] and [11] this text is compressible into a text representing the intention in its entirety. As we are in presence of a text translating an intention of use, so we can identify the action and the object on which this action applies. Given that the action is known according to what precedes then, it is trivial to find the verb that translates this action, on the one hand and on the other hand find the article of the group of objects.

The transformation process of an inconsistent goal into a consistent goal is called the globalization of the goal. Let $b$ be an inconsistent goal, and a function $\theta: L \to L$, the globalization function. $\theta(b)$ represents the consistent goal deduced from the globalization of $b$. $\theta(b)$ is called globalized goal or global goal of $b$. Applying [11] and [12], $\theta(b)$ globalizes the usage intention of the user. This means that, it presents the users' usage intention in its entirety. Thus, we can deduce that the intension of the user is a refinement of the globalized goal. We deduct from this fact that a globalized goal always admits at least one sub goal that represents the user's specific need. A goal that does not admit any sub goal is called: an elementary goal.

Let's denote a set of consistent goals of an organization by $B$; $A$, the set of intentions expressed by the staff of this organization; and $f$ a function defined as follows:

$$f: \begin{array}{l} A \to B \\ b \in A, f(b) = \begin{cases} b\ if\ b\ is\ consistent \\ \theta(b)\ \ if\ not \end{cases} \end{array}$$

In this approach the goal is the key concept. In the rest of this section we shall give definitions of the other fields of the knowledge bits. These other fields will have their importance when it will be necessary to pass our specification to other specifications.

## 2.6 Business Rules

Definition 5: business rules

Business rule defines a law in the domain to which the goal must conform itself. It helps in the setting up of a management process to achieve the goal.

Business rules have an impact on the external environment (the organization), as well as on the internal environment (the system) [16]. They are tree types: scheduling rule, trigger rule or static constraints [9]. A scheduling rule is a law of the domain that describes the order in which goals must be achieved, for an instance of a given object. The trigger rules and the static constraints keep the same semantics as in [9].

Definition 6 : Polysemous goals

A goal is termed polysemous if the business rule that governs it changes from one domain to another.

## 2.7 Importance of an Expressed Requirement

The "level of importance of the expressed requirement" expresses the credit that a user gives to this expressed requirement. It can take one of the following values: 'very important and indispensable,' 'important and indispensable', 'indispensable', 'important, necessary', and 'in bonus'. It is possible to choose a numeric value to express the level of importance of a user's expectation. We noted through a survey carried out on a sample of twenty-five Cameroonian government services that the staff had difficulty assigning some numeric values to materialize the credit associated to their expectations. To this effect, we recommend the use of an assessment scale "level of importance" of requirements according to the target population (futures users of the system)

## 2.8 Knowledge Bits or Expressed Requirements

An expressed requirement or knowledge bit is defined in the following manner:

$\partial = (\psi, \omega, \lambda, \delta, \nu)$ where :

$\partial$ *is the name knowledge bit*
$\psi$ *is the context in which the goal is defined*
$\omega$ *is the gaol*
$\lambda$ *is the business rule*
$\delta$ *represents the contraint*
$\nu$ *is the level of importance of the goal*
$\partial$ represents the name of a concept of the domain $\psi$.

In the definition [9], we replaced the "conceptual pieces" by "constraints, level of importance" of the user usage intention, because in the course of this work we shall be interested exclusively in the formalization of the representation of user requirements. In the intention to integrate the level of importance of user goals, we introduced the field "level of importance" in the knowledge bits. The "constraint" field was added to exhibit the non functional requirements which are hidden behind the usage intention of the user. These non functional needs could be refined by software engineers.

<u>Constraint:</u> - The eligible action verbs henceforth shall be those that translate actions, by preference management actions, susceptible of being automated by a computer system. For example, verbs: to speak, to eat, to jump, to run, to laugh, to sell, to paint, etc. . , are excluded; - our work is exclusively about abstract goals; - goals expressed by sentences in the active voice.

We have defined the basic concepts necessary to understand our approach. In the following section we are going to formalize the above mentioned concepts.

## 3. GOAL ORIENTED APPROACH FOR THE DRAWING UP OF REQUIREMENT MODEL

### 3.1 Eliciting User Requirements

The Elicitation of user requirements is an activity aimed at collecting user needs, as well as validating user requirements. With this intention, the software engineer must use a form structured as the requirement eliciting document as defined in section 2 paragraphs 2.2. The software engineer should first have the different identified requirements validated by a competent staff of the organization in the presence of the users who initially expressed the requirements. Once validated, the software engineer must complete the domain field in each form. As we proceed, only validated requirements shall henceforth be considered. We shall then indicate user expectations or expressed requirements; requirements whose state is validated, $\Omega$ will represent the set of expressed requirements of the organization. Let's consider $\Omega$, $b.xxxx$ will represent the field $xxxx$ of $b$

### 3.2 Selection of Requirements

Let's consider a human language $L$ (French, English, German, etc. ), we define the function $\approx$ as follows:

$$\approx: \begin{array}{l} L \times L \rightarrow \{true, \quad false\} \\ (a,b) \in L \times L, a \approx b = \begin{cases} -true, if\ a\ et\ b\ are \\ \quad similar\ in\ meanig \\ -false, if\ not \end{cases} \end{array}$$

The symbol $\urcorner$ shall be used to express negation, $\urcorner b$ represents an expression opposite in meaning to $b$.

Let's consider a and b two requirements of the organization, with a and b elements of $\Omega$ . We have:

$a : (\upsilon, \chi, \nu, \gamma, \lambda, \delta, \psi, \varsigma)$ and $b : (\upsilon', \chi', \nu', \gamma', \lambda', \delta', \psi', \varsigma')$ .

- <u>Property 1</u> (inconsistent requirements): $a$ will be said inconsistent if and only if $a.\gamma$ is inconsistent.

- <u>Property 2</u> (ambiguity of requirements): $a$ and $b$ will be said ambiguous if and only if :

$\gamma \approx b.\gamma'$, $a.\lambda \neq b.\lambda'$ and $a.\psi \neq b.\psi'$.

- <u>Property 3</u> (similarity of requirements): $a$ and $b$ will be said similar if and only if: $a.\gamma \approx b.\gamma'$ and $a.\lambda \neq b.\lambda'$.

- <u>Property 4</u> (contradiction between requirements): $a$ and $b$ will be said to be contradictory if and only if at least one of the following conditions are satisfied

$(a)\ a.\gamma \approx \urcorner b.\gamma'$ *and* $a.\psi \approx b.\psi'$;
$(b)\ a.\gamma \approx b.\gamma'$ *and* $a.\lambda \approx \urcorner b.\lambda'$

- <u>Property 5</u> (identity of requirements): $a$ and $b$ will be said to be identical if and only if :

$a.\gamma \approx b.\gamma'$ *and* $a.\psi \approx b.\psi'$ *and* $a.\lambda \approx \urcorner b.\lambda'$.

- <u>Property 6</u> (Consistency of requirements): $a$ is said to be consistent if and only if for any requirement b none of the above properties is satisfied.

Let's consider $\nabla$ as set of retained requirements, and R those rejected. The process of requirement selection consists in:

```
1-  $\nabla = \emptyset, R = \emptyset$
2-  $\Omega \cup \nabla \cup R; \nabla = \emptyset, R = \emptyset$
3-  if a verifies the property 1 : rejected a ;
    $\Omega = \Omega - \{a\}; R = R \cup \{a\}$
4-  if there exist a and b verifying the property 2 to
    4 : reject a and b ; $\Omega = \Omega - \{a,b\}$ ;
    $R = R \cup \{a,b\}$
5-  if there exist a and b verifying the property 5 :
    $\Omega = \Omega - \{a,b\}; \nabla = \nabla \cup \{a\}$
6-  if a verifies the Property 6 : $\Omega = \Omega - \{a\}$ ;
    $\nabla = \nabla \cup \{a\}$
Repeat steps 3 and 6 until $\Omega = \emptyset$.
```

Elements of R must be the subject of discussion with the staff who expressed them. At the end of discussions repeat steps 2-6. This activity aims at discovering and deleting any requirement with an "unnecessary" user goal. The software engineer must rely on his understanding of the different usage intentions of the user. When one of properties 2-5 is verified, it is recommended to discuss with those who expressed these intentions in question. It is only if the latter is confirmed to be of same intention that this group is validated. At the end, the software engineer summarizes the requirements by goal and submits them once more for validation by the organization. This grouping ensures that the semantics of goals is the same for everybody (users and software engineers) and that there is no double use of a goal. This is why the classification of user requirements is done by goal. It enables us to detect cases of double use.

## 3.3 Transformation of Requirements into Knowledge Bits

Let's consider a requirement a of Ω, $a : (\upsilon, \chi, \nu, \gamma, \lambda, \delta, \psi, \varsigma)$ , and $C$ a Knowledge bit, by definition, C can be expressed in the form $C : \partial = (\psi, \omega, \lambda, \delta, \nu)$ . Our objective is to be able to construct $C$ from $a$. From rule (1), we have :

$$f(a.\gamma) : (\vartheta,\ \Delta_b(\tau),\ \tau).$$

*Rule 2* (translation of requirements):

$$\partial = f(a.\gamma).\tau\ ;\ \psi = a.\psi;\ \ \omega = f(a.\gamma);$$
$$\lambda = a.\lambda\ ;\ \delta = a.\delta\ ;\ and\ \nu = a.\nu.$$

This activity aims at transforming users requirements into knowledge bits, integrating the level of importance of each expressed requirements. This procedure is repeated for each element in Ω. Knowledge bits from the translation form Σ the set of knowledge bits of the organization.

## 3.4 Development of Requirement Model

### 3.4.1 Problem Frames

A problem frame [17] (or problem diagram) is a diagram that defines in an intuitive manner a class of identified problems in terms of its context and domains characteristics, interfaces and requirements. The system to be developed is represented by a "machine". For each problem frame, a diagram is established. Simple rectangles denote application domains (which already exist), rectangles with a double bar denote domains "machine" which are to be realized, and requirements are noted by an oval dotted line. Lines joining them represent interfaces, also called "shared phenomena". Jackson distinguishes "causal" domains which obey certain laws, lexical domains which are the physical representations of data, and domains "biddable" (give out orders) which are people. The use of a problem diagram consists in instantiating domains, interfaces and requirements.

In the continuation we shall rely on problem diagram concepts to do demonstrations. We will designate by machine, the machine presented earlier. This machine transforms knowledge bits (problem of the real world) into future exigencies of the system; Σ shall represent the space of knowledge bits (problems) of the organization and $PF$ "problem frame" of the organization, it is the set of exigencies of the organizations' future system. $S^{\emptyset}$ will represent the set of all objects of the organizations' information system. The latter are expressed in business rules.

### 3.4.2 Basic Axioms for the Development of the Requirement Model of an Organization

Let's consider a knowledge bit $b$. $b$ element of $\Sigma$, $b : \partial = (\psi, \omega, \lambda, \delta, \nu)$, a machine $W$ as expressed in [18] and an object $\propto$ of $S^{\emptyset}$. We note :

$$\bar{\bar{W}}(\propto, \partial.\psi, \partial.\omega, \partial.\lambda, \partial.\delta),$$

The processing of the object $\propto$ by the machine W, in the context $\partial.\psi$, under the rule $\partial.\lambda$ and the constraints $\partial.\delta$, such that the goal $\partial.\omega$ is satisfied. We shall say such a machine recognizes requirement $b$. We construct $S^{\partial}$ in the following manner:

$$S^{\partial} = \{a | a \in S^{\emptyset},\ \bar{\bar{W}}(a, \partial.\psi, \partial.\omega, \partial.\lambda, \partial.\delta)\}.$$

$S^{\partial}$ is the set of objects of the organizations' computer system for which the expectation $\partial.\omega$ is satisfied under the rule $\partial.\omega$ and the constraint ∂.δ. next, the notation $b : \partial = (\psi, \omega, \lambda, \delta, \nu)$ shall be replaced by $b = (\psi, \omega, \lambda, \delta, \nu)$ and $S^{\partial}$ by $S^{b}$ ; the knowledge bit shall be called requirement or expectation ; $PF_b$ shall represent the set of exigencies of the system (software requirement) which are satisfied in the context $\partial.\psi$, under the rule $\partial.\omega$ and the constraint $\partial.\delta$ ; $*$ is use to represent an undetermined value of a field.

#### 3.4.2.1 Axiom 1 : Coherence of Knowledge Bits

Let's consider an expectation b of Σ, we say that $b = (\psi, \omega, \lambda, \delta, \nu)$ is a coherent requirement if and only if :

(1) -$S^{b} \neq \emptyset$;
(2) - $PF_{b} \subseteq PF$

#### 3.4.2.2 Axiom 2 : Concept of Sub-requirement

Let $a$ and $b$ two coherent requirements of Σ and two machines V and W such that V recognizes a and W recognizes $b$.

(3) we say that $a = (\psi, \omega, \lambda, \delta, \nu)$ is a sub-requirement of $b = (\psi', \omega', \lambda', \delta', \nu')$ or that ω is a refinement of ω′, if and only if: $S^{a} \subset S^{b}$ and $\exists e \in S^{b}$, such that $\bar{\bar{W}}(e, b)$ and the execution of V(e,a) does not satisfy $b$.

(4) We say that b is a generalization of $a$ if and only if $b$ is a sub requirement of $a$. We note $\rho^{a}$ that generalization of $a$. $\omega$ is called the specific goal of ω′

(5) We say that $a$ and $b$ of Σ are traceable if and only if : $b = \rho^{a}$ or $a = \rho^{b}$.

(6) A requirement $b$ is incomplete if it has only one daughter requirement. Incomplete requirements must be resolved by addition of daughter requirements. If a daughter requirement has a daughter then add other daughter requirements; else merge father and daughter requirements as a unique requirement.

(7) if $b = \rho^{a}$ , then λ is $a$ scheduling rule of daughter requirements of $b$.

#### 3.4.2.3 Axiom 3 : Merging Requirements

Let a et $b$ be two coherent requirements of Σ , and $M, V$, $W$ three machines such that M recognizes $a$ and $V$ recognizes $b$,

(8) We say that $a = (\psi, \omega, \lambda, \delta, \nu)$ can be merged with $b = (\psi', \omega', \lambda', \delta', \nu')$ if and only if:

$\Omega = \omega', \exists\, c = (\psi'', \omega'', \lambda'', \delta'', \nu'') \in \Sigma$,
$\overline{\overline{W}}(*, b) \,/\, S^b \cup S^a \subseteq S^c$,
$e \in S^a,\ d \in S^b \rightarrow \overline{\overline{W}}(e, c)$ and $\overline{\overline{W}}(d, c)$ ;

where :

$\psi'' = \psi' \cup \psi$,
$\omega''$ : is a goal including $\omega'$ and $\omega$.
$\lambda''$ : scheduling rule;
$\delta'' = \delta' \cup \delta$,
$\nu''$ : is the highest level of importance between $\nu'$ and $\nu$

### 3.4.2.4 Axiom 4: Importance of Requirements.

Consider two coherent requirements $a = (\psi, \omega, \lambda, \delta, \nu)$ and $b = (\psi', \omega', \lambda', \delta', \nu')$ of $\Sigma$ with $\nu$ and $\nu'$ of values taken from an ordered set. We shall say that a is more important than b if and only if : $\rho^a = \rho^b$ *and* $\nu > \nu'$.

### 3.4.2.5 Axiom 5: Ambiguous requirements

Two coherent requirements $a = (\psi, \omega, \lambda, \delta, \nu)$ et $b = (\psi', \omega', \lambda', \delta', \nu')$ of $\Sigma$, are said ambiguous if and only if :
$\psi = \psi'$ , $S^a = S^b$ *and* $\omega \neq \omega'$.

### 3.4.2.6 Axiom 6: Partitioning of Requirements

Let n coherent requirements $c_1,\ c_2, \ldots,\ c_n$ of $\Sigma$, and a coherent requirements $h = (\psi', \omega', \lambda', \delta', \nu')$ of $\Sigma$, such that $S^h = \bigcup_{i=1}^{n} S^{C_i}$, then the requirements h is sub divisible into $n$ sub requirements $h_1,\ h_2, \ldots,\ h_n$ where $h_i = c_i$ and $\rho^{h_i} = \rho^{c_i}$.

### 3.4.2.7 Axiom 7: Identical Requirements

Two coherent requirements $a = (\psi, \omega, \lambda, \delta, \nu)$ and $b = (\psi', \omega', \lambda', \delta', \nu')$ of $\Sigma$, shall be said to be identical if and only if :

$$\psi = \psi', \omega = \omega' \text{ and } S^a = S^b.$$

### 3.4.3 Can we say that the requirements of an organization are classified in hierarchy?

Our objective in this section is, firstly, to adopt a formal proof that requirements as we represent them enable the description of all the activity of an organization; secondly, to formally define when we can consider requirements of an organization as entirely described; and thirdly, to give some characteristics of the set of requirements of an organization.

### 3.4.3.1 Formal Proof on the Completeness of Description of a Business Process

To show that our organizations' requirement representation model enables the description in an exhaustive manner the requirements of a business process of this organization, it is sufficient to show that this representation is another way to describe a business process. For this we are going to rely on the work of R. Atsa and Mr. Fouda in [20, 21] firstly, we shall construct a requirement from the concept of " task " and its underlying notions; secondly, we are going to show that the properties relative to the description of the business process, elaborated in [20] are applicable by the set of requirements of a business process. R. Atsa and Mr. Fouda defined in [20] the concept of business process of an organization and its underlying concepts. In their vision a business process is a set of tasks that must be realized in a context, for the attainment of objectives or precised goals. A task is seen as data of: a realization context; a set of states; a transition function between states; and of an objective to attain or goal. In this respect the achievement of a goal is realized by the observation of values associated to each indicator, linked to the execution of one or a set of tasks. A state is the set of objects of the organization on which the task acts to achieve fixed objectives

Let's consider a requirement $a = (\psi, \omega, \lambda, \delta, \nu)$ of $\Sigma$, and a task t. we have :

- $const(t)$ : the set of constraints linked to the execution of $t$
- $t$
- $obser(t)$ : the set of values of indicators from which we observes the achievement of the objective associated to $t$
- $f_t : state \rightarrow state$ : the transition function between states associated to $t$
- $cont(t)$ : the context of execution of the task $t$.
- $etats(t)$ : the set of objects of the organization on which the execution of $t$ acts.

a) Let's construct a as a function of $t$ and its underlying concepts:

From definition 1, $\omega$ is expressed in the form of a triplet $(\vartheta,\ \Delta_b(\tau),\ \tau)$ under the conditions of section 2.5. express $\omega$ as a function of t in the following manner:
$\omega = (\text{traiter},\ \ \Delta_b(\text{obser(t)}),\ \ \text{obser(t)})$

From definition 5, write : $\lambda = f_t$ ; and from definition of domain (cf. section 2.3), express : $\psi = \text{cont(t)}$ ; While relying on the definition of constraints (cf. section 2.4), write: $\delta = \text{const(t)}$ ; similarly (cf. section 3.4.1), we express $S^a$ in a similar manner as follows: $S^a = \text{etats(t)}$. Without deviating from the general rule, we shall consider the entire task in the approach of [20, 21] have the same importance. To this effect we put $\nu = 1$.

$a$ is expressed as a function of t in the following manner:

$a = \left(cont(t), \omega, f_t, const(t), 1\right)$, *where :*
$\omega = (traiter,\ \ \Delta_b(obser(t)),\ \ obser(t))$

b) Let's show that properties related to our specifications are applicable to tasks as defined by R. Atsa et M. Fouda. In [21], several properties have been elaborated on the set of states of the environment. They induce a set of dependency rules between the different tasks of the business process. Key among these are the following: consistent state (Useful State), Equivalence between states (Equivalent of state), sub - states (Sub-state). Our work shall consist in showing that axiom 1, 2 and 7 are applicable to the concepts of tasks as defined in [20].

Let's consider two requirements $a$ and $b$, with $a = (\psi, \omega, \lambda, \delta, \nu)$ and $b = (\psi', \omega', \lambda', \delta', \nu')$ of $\Sigma$, and two tasks $t$ and $t'$. We suppose that a is associated to $t$ and b at $t'$ :

1st case (consistent requirement):

$S^a \neq \emptyset$, *from what precedes,* $etats(t) \neq \emptyset$ *; where,* $etats(t)$ *is a consistent state.*

2nd case (identical requirement):

$\psi = \psi',\ \omega = \omega'$ and $S^a = S^b$, from what precedes, we deduce that $states(t) = states(t')$, *where* $haract(states(t)) = charact(states(t'))$ .

3rd case (sub requirement) :

$S^a \subset S^b$, from what precedes $\text{states(t)} \subset \text{states(t')}$ consequently
$\text{charact(states(t))} \subset \text{charact(states(t'))}$.

We have just shown that our requirement modeling approach of an organization is another manner to describe a business process. R. Atsa and Mr. Fouda described the business process from the angle of tasks and states of the environment, in this paper we showed that a description of the business process can also be made from the angle of requirements and goals.

### *3.4.3.2 When can we consider that the requirements of an organization have been described entirely ?*

We are going to consider that needs of an organization have been described entirely when:

- all traceable requirements are complete, and the associated business rules to these requirements are the scheduling rules.
- the elementary requirements are constituted of all elementary tasks,
- there exist no knowledge bits that could either be father of another, nor daughter of another.

### *3.4.3.3 Some characteristics of the set of requirement*

Lemma (2): The requirements of an organization can be classified on the basis of hierarchy.

Proof: Let's consider any business process $P$, according to what precedes; we can obtain $besoins(p)$, the set of requirements which characterize $P$. Let's show that for any element $a$ in $besoins(p)$. Let $a$ be either father, or daughter of a requirement. From [18] the business process can be split into tasks and forms a hierarchical set. We showed in section 3. 4. 3. 1 that to each associated task of a professional process is associated a unique requirement. A sub business process is by definition a set of divisible requirements in which $c_i$ $(i \in N)$ represents the tasks of this sub process.

## 4. CONCLUSION AND FUTURE WORKS

This paper presents a goal oriented approach for the definition of a business process requirement model, taking to consideration the level of importance of every user goal. This approach aims at establishing a formal link between the expression of user requirements and their formal representation. We noted that the scientific community attributes a particular attention on the engineering approach. Forgetting, as well that the requirement is at the center of the development of computer products; and from this fact, its formal description is an indispensable task in the success of a computer project. The results of these works can be used in the domain of requirement engineering, as well as in model driven architecture; or in the development of applications on the basis of software component. What explains this state of things is that work that has been done in this respect is prior to all these research domains.

We did not put to the foreground the level of importance of requirements and we did not formalize the representation of business rules. This is the subject of current work in our laboratory and in the same order of idea we envisage in the coming days:

- to formalize the specification of a rule;
- to define a platform which enables us to obtain a system requirement model and in the same line to identify the reusable requirements;
- to enrich work on the selection software components.

The purpose of this work is the putting in place of a platform for component based software development from the nearest specification possible to human language. This will surely minimize incomprehension between developers and users, and to produce systems of lower cost based on software components.

## 5. REFERENCES

[1] Lubars, M., Potts, C., Richer, C.: A review of the state of the practice in requirements modeling.Proc. IEEE Symp. Requirements Engineering, San Diego 1993.

[2] Karen Mc Graw, Karan Harbison, User Centered Requirements, The Scenario-Based Engineering Process. Lawrence Erlbaum Associates Publishers, 1997.

[3] The Standish Group, Chaos. Standish Group Internal Report, http://www.standishgroup.com/chaos.html, 1995

[4] Voas J., « COTS Software - The Economical Choice ? », IEEE Software, vol. 15 (3), p. 16-19,March, 1998.

[5] Tran V., Liu D.-B., « A Procurement-Centric Model for Engineering Component Based Software Engineering », Proceedings of the 5th IEEE International Symposium on Assessment of Software tools (SAST), Los Alamitos, California, USA, June, 1997.

[6] Bornwsword L., Obendorf P., Sledge C., « Developing New Processes for COTS-Based Systems », IEEE Software, vol. 34 (4), p. 48-55, July-August, 2000.

[7] Güngör-En C., Baraçli H., « A Brief Literature Review of Enterprise Software Evaluation and Selection Methodologices : A Comparison in the Context of decision- Making Methods », Procedings of the 5th International Symposium on Intelligent Manufacturing Systems,p. 874-883, May, 2006.

[8] Kahina Hassam, Bart George, Régis Fleurquin, Salah Sadou, « utilisation de la transformation de modèles pour faciliter la sélection de composants logiciels », IDM'2008 5-6 juin Mulhouse, 2008.

[9] Farida Semmak, Joël Brunet, « Un métamodèle orienté buts pour spécifier les besoins d'un domaine », 23e Congrès INFORSID, pp 115-132, mai 2005.

[10] Bouchra El Asri, Mahmoud Nassar, Bernard Coulette, Abdelaziz Kriouile, « Architecture d'assemblage de composants multivues dans VUML », Revue RSTIL'Objet, PP 1- 32, 2005.

[11] Minel J.-L., « Le résumé automatique de textes : solutions et perspectives», in Proceedings of TAL, Résumé automatique de textes, vol. 45/1, p. 7-13, 2004.

[12] Mehdi Yousfi-Monod,Violaine Prince, « Compression de phrases par élagage de leur arbre morpho-syntaxique : Une première application sur les phrases narratives », RSTI-TSI–25/2006. Document numérique, pp 437-468, 2006

[13] G. Grosz, « Ingénierie des besoins : problèmes et perspectives », Centre de recherche , Paris Sorbonne 1, 2005.

[14] François Ott, Pierre Vaast, « livre de grammaire : sixième en troisième», Hatier, 1991

[15] François Ott, Pierre Vaast, « livre de grammaire: Première en Terminale», Hatier, 1991

[16] Zave P., Jackson M., « Four Dark Corners of requirements Engineering », ACM Transactions on Software Engineering and Methodology, 1997

[17] Michael Jackson. «Problem Frames. Analyzing and software development problems». Addison-Wesley, 2001

[18] Jean-Noël Gillot, « La gestion des processus métiers: l'alignement des objectifs stratégiques de l'entreprise et du système d'information par les processus», pp132-147, IDM'1997

[19] Alan W. Brown, Sterling Software et Kurt C.Wallnau, «the Current State of CBSE», IEEE Software, Software Engineering Institue, October 1998

[20] R. Atsa Etoundi, M. Fouda Ndjodo, « Human Resource Constraints driven Virtual Workflow Specification », IEEE SITIS pp 176-182, 2005.

[21] R. Atsa Etoundi, M. Fouda Ndjodo, « Feature-Oriented Business Process and Workflow », IEEE SITIS pp 114-121, 2005.