Dealing with Device Collaboration Rules for the PCSCW Model

Kahina Hamadache, Luigi Lancieri

Orange Labs, 42 rue des Coutures, 14000 Caen, France
LIFL, University Lille 1, Lille, France
kahina.hamadache@orange-ftgroup.com
luigi.lancieri@univ-lille1.fr

Abstract. In this paper, we describe the design, the development and the use of devices collaboration rules for the PCSCW (Pervasive Computing Supported Collaborative Work) Model. These rules rely on the precise description of roles, tasks, actions, resources required by these actions and constraints associated to these resources to select the proper way to make devices cooperate with the final objective to facilitate the collaboration of humans. We suggest that by defining constraints on resources as triplets composed of a parameter, a value and an associated criticality it allows us to quantify, estimate, compare and then choose between several candidate rules. The finality given by these rules is a simple but efficient way to make devices choose automatically the most appropriate way to cooperate.

Key words: Pervasive Computing, Collaborative Work, Constraints Modelling, PCSCW, Collaboration rules.

1 Introduction

The computer supported collaborative work (CSCW) domain is probably one of the most active research fields of recent years. Indeed, due to the facilitations brought by computers and smart devices it is almost impossible to find people working without them. For the past few years information technologies are evolving toward the multiplication of smart electronic devices such as smartphones, laptops, GPS and so on. Despite or maybe because of this proliferation the digital environment is a non-continuous space where miscellaneous devices can communicate, or not, with others. Thus, in order to make this space "continuous" the Pervasive Computing [7] is based on the communication between devices to smartly adapt their behaviour to the current context of users and offer them a seamless interaction with the digital world.

Given this aspect our work has rapidly focused on the way we could integrate the pervasive computing within CSCW. Such integration could bring various advantages: resource and time saving for companies, work simplification and task automation for workers. In a "green" consideration it could also help reducing work's energetic impact by accompanying users' in using lighter devices and services.

On the long road toward this accomplishment we have already sowed some seeds. Hence as we will describe in the next sections we have proposed the *PCSCW* model (see below) which is designed to improve the integration of collaborative work aspects with pervasive computing and make them benefit from each other, we propose a model that allows describing users with their roles, tasks, actions and resources required to perform them. Then by comparing required resources to available ones we can trigger device cooperation routines to facilitate the collaboration of users.

All devices don't natively support collaboration with others. In order to solve this kind of issue we argue that the definition of device collaboration rules could be of great help. These rules intend to define tasks that could be automatically performed by devices to collaborate in order to allow a user to do its own task. The main idea behind this is the following: a user needs two (or more) resources to complete an action related to a task; these resources are not available on a single device, but the combination of several of them can supply the resources. Thus, device collaboration rules define what actions can be performed by devices to collaborate, for finally providing required resources to the user. These collaborations can be of various kinds: network access sharing, heavy computing task delegation and notification of events, anything you can imagine to make several devices cooperate.

In this paper we will go down in the depths of these devices collaboration rules; we will present how these collaboration rules are designed and how we can use them.

This paper is organized as follows: we will first introduce the basic concepts of the PCSCW Model to be able in a second time to efficiently describe device collaboration rules we're using with. Third, to illustrate our work we propose a use case based on simple scenarios melting collaboration of users with cooperation of machines. The fourth section is dedicated to present a way to implement and use the PCSCW model and its collaboration rules. Next to the last we present a concise state of the art on pervasive computing modelling and reasoning. Finally we conclude by giving some perspectives of our future work.

2 PCSCW Model

As our research interest has focused on the integration of the pervasive computing aspect in the computer supported collaborative work, we have proposed in a previous work [1] an original model which aims at making smart devices cooperate seamlessly to improve and facilitate the collaboration of users. This model, named PCSCW for Pervasive Computing Supported Collaborative Work, relies on some simple but essential "sub-models":

- A *Task Model* composed of mainly two concepts:
  - *Task*: represents a meaningful process to be performed by one or more users to achieve a specific goal, for instance "creating a webpage", can be composed of a set of subtasks or actions;
  - *Action*: describes an atomic step of a task, it has no discriminatory meaning as it can't be understood outside of a task. To illustrate it we can consider the action "opening a web browser" that doesn't convey any specific meaning but can be integrated in tasks such as "searching the web" or "checking mails".

- A *Role Model* built above the task model, it extends it by providing one more concept and some refinements about tasks:
  - *Role*: it defines a role to be played by one or more users by wrapping tasks into subsets: mandatory, allowed and forbidden tasks;
  - Tasks can require one or more roles to be performed. Thus a single task may be shared among several roles and then becomes a “Collaborative Task”.
- A *Resource Model* providing a common ground to represent:
  - *Required resources*: the set of resources an action requires to be performed. By describing these requirements in term of software, hardware, human and social resources at the action level we can efficiently describe resources required for a given task;
  - *Available resources*: the set of resources available in user’s environment, it provides a structured representation of the context;
  - *Device*: the representation of a contextual device is merely a part of available resources but with a particular extent as it is considered as an active agent of the collaboration.

In addition to these sub-models the PCSCW Model includes *Device Collaboration Rules*. The main principle of these rules is the following: by comparison of resources required to perform a task or an action with available devices resources we can trigger specific interactions between devices to make them cooperate to finally provide all required resources to the user. Going a little further these rules can even perform whole actions or tasks and prevent users from doing repetitive and thoughtless ones.

All these features create a model allowing smart devices of users’ context to automatically and seamlessly cooperate to facilitate, channel and enhance the collaboration of users.

3 Device Collaboration Rules

As it is our main focus for this article we will make an in-depth investigation of the devices collaboration rules, we will see how they are designed and how we can use them.

3.1 Rules

Making two or more devices collaborate doesn’t only rely on resource matching; indeed you need to have defined a set of behaviours to trigger when user’s context matches some rules requirements. To be coherent with its main principle, rule behaviour contains actions to be performed with the description of their associated resources. Indeed, each device collaboration rule is defined with the following syntax (1):

$$\text{IF } (\textit{context.resources} \equiv \textit{rule.resources}) \text{ THEN DO } \textit{rule.behavior} \quad (1)$$

Obviously, several rules can have a similar or partially equivalent set of required resources, it implies that more than one rule can be matched by the current context and lead to some kind of conflict. Besides we need to express a specific need here: all device collaboration rules must have the same knowledge for their reasoning, it implies that context representation has to be "locked" during the reasoning process. Then, to be able to select the adequate rule to trigger we need a tool to evaluate their relative suitability. To fill this requirement and as we'll see in the next section we propose to define constraints on rules' resources.

3.2 Constraints

The design of Device Collaboration rules for the PCSCW already requires describing roles, tasks, actions and resources. These resources can be of various kinds mainly categorized along hardware and software ones. In order to complete this design we need to be able to express constraints over the required resources.

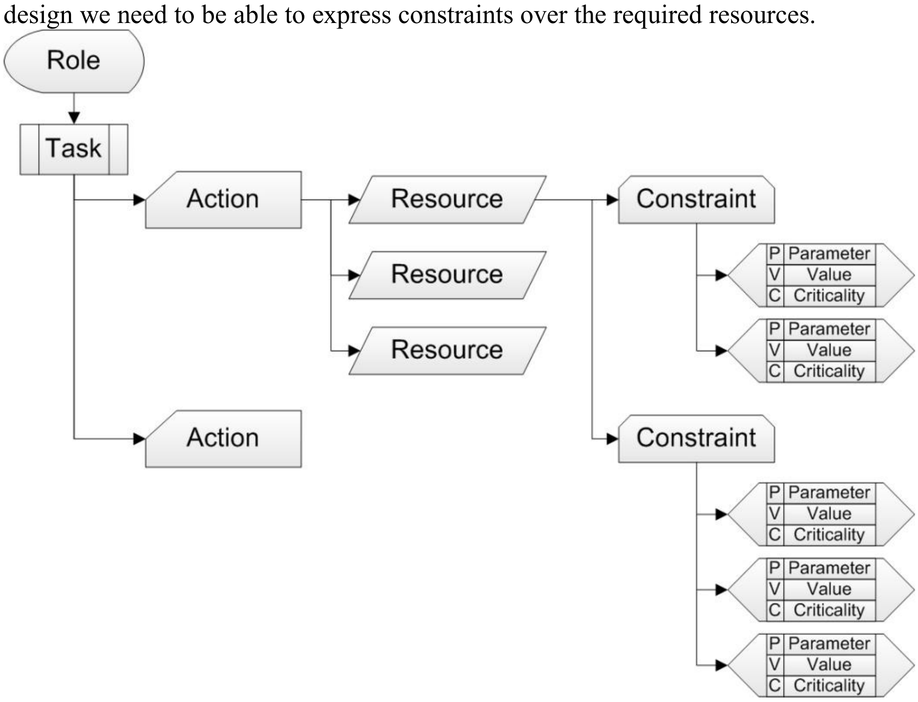

Fig. 1. Constraints for the PCSCW Model

Fig. 1 defines the addition of constraints on resources of the PCSCW Model. As it is depicted, a single resource may be related to several constraints, each of whom is described by a triplet {P, V, C} as: *P* a parameter which represents the precise point to be evaluated, *V* the expected (or required) value (or threshold) for this parameter and *C* the criticality of this parameter. This last component of a constraint has a specific impact as it is the one allowing a device to select the appropriate collaboration rule.

In addition to this triplet we propose to organize constraints in five main categories, facilitating and guiding rules designer in their work: *Availability, Cost, Privacy, Reliability and Security*.

Examples of parameters falling below these categories could be: CPU Load (Availability), connection price (Cost), data access (Privacy), website breakdown relative frequency (Reliability) and network secured protocol (Security). Given that values are related to parameters and are illustrated in the following use case, we won't give more examples of them.

While parameters and values are easily collectible from real use case, the criticality needs some more analysis and requires defining its own set of values. In this perspective we propose to use a really simple seven-level scale: *Optional*, *Very Low*, *Low*, *Average*, *High*, *Very High* and *Mandatory*.

Optional indicates the constraints doesn't need to be fulfilled but can provide a valuable benefit for the collaboration and can help choosing between two equivalent rules. On the contrary, the Mandatory level implies that if the constraint is not met the collaboration rule cannot be used in the current context.

3.3 Using Device Collaboration Rules

Until now we have described all required concepts to understand the PCSCW model. Let's have a look at the real use of all these descriptive levels and how the model helps at finding the right cooperative behaviour.

To formalize and facilitate the use of collaborations rules we have defined a six-step process describing how a specific rule can be triggered during the collaboration:

1. On context data update, an analysis of this update is automatically started;
2. If this analysis points out that some device collaboration rules may eventually improve the current collaboration by facilitating the accomplishment of an action we start the comparison between context information and rules activation requirements;
3. This comparison can end in three ways:
    a. No rule can effectively improve the collaboration in the current state of the context, we stop the process here;
    b. One rule can improve the collaboration, in this case we jump directly to step 6;
    c. Several rules can improve the collaboration, in this case we need to choose between them the most relevant and efficient, we go to step 4;
4. To choose between the selected rules we need to compare their suitability, their relative efficiency. To do it we confront action's required resources and their associated constraints with resources supplied and used in each rule' behaviour.
5. From this confrontation we bring out a score for each rule and all we have to do is to keep the rule with the higher score. In the case where several rules have the same score it means that none of them can be "automatically" preferred and the device has to take one of them arbitrary.
6. Run the chosen behaviour.

Hence to compare two selected rules we need to quantify each of them according to resources and constraints. In fact, the way we have defined constraints facilitates this comparison by relying on the quantification of criticality and the evaluation of the constraint fulfilment of each rule.

Resource	*Res₁*		*Res₂*	…	*Res_N*			*Suitability*
Constraint	$C_{1,1}$	$C_{1,2}$	$C_{2,1}$	…	$C_{N,1}$	…	$C_{N,M}$	
Rule R_A	$V_{A,1,1}$	$V_{A,1,2}$	$V_{A,2,1}$	…	$V_{A,N,1}$	…	$V_{A,N,M}$	$S_A = \sum V_{A,I,J}$
Rule R_B	$V_{B,1,1}$	$V_{B,1,2}$	$V_{B,2,1}$	…	$V_{B,N,1}$	…	$V_{B,N,M}$	$S_B = \sum V_{B,I,J}$
…								…
Rule R_X	$V_{X,1,1}$	$V_{X,1,2}$	$V_{X,2,1}$	…	$V_{X,N,1}$	…	$V_{X,N,M}$	$S_x = \sum V_{X,I,J}$

Table 1 Rule Comparison Matrix

Table 1 depicts the rule comparison process. On this matrix, each rule to be compared is evaluated along with its provided resources and their constraints. Thus, for each constraint of each resource we assign a value ($V_{r,i,j}$) which is limited by the criticality of the constraint evaluated ($V_{r,i,j} \leq Crit(C_{i,j})$). As for now we have decided to use a simple system: a value is comprised between 0 and 5, a *very low* criticality means 1, *low* means 2 and so on until *very high* which means 5. As we already evoked, an unsatisfied mandatory constraint eliminate the rule, while an optional one can only be used to decide between two equivalently suitable rules. Then if we consider a constraint with a *high* criticality the evaluation of this constraint can't be higher than 4. We know that the assignment of values can be sometimes problematic if the threshold value that was first defined in the constraint is not easily comparable. Thus, if we consider constraints such as *data encryption* it can be hard to give a value to a encryption method different from the one that was defined. Still, there are solutions to this kind of problem, for instance we can use predefined rankings.

Limiting rules' assigned values with the associated criticality and relying on constraints *Values* prevent from selecting a rule that does not satisfy most of the critical constraints but tremendously outperform a minor one. Thus, even if network bandwidth constraint has been defined with a *low* criticality and a value which has to be at least 0,5mbps, the rule allowing a ultra high speed connection faster than 100mpbs but poorly satisfying other constraints will certainly not be selected (except for the case where other rules are worse) as its connection can give it more than 2 "points".

Finally we obtain the suitability level of each rule by adding up all assigned values and we are able to compare rules and find the most appropriate by comparing their suitability.

4 Use Case

The description of the model and device collaboration rules' process is now complete. In order to illustrate their use and benefits we propose a use case based on a simple collaboration between 3 coworkers.

Leela, Amy and Philip are members of a team and have to collaborate on a new marketing campaign for the new product of their company. In this perspective they have to perform several tasks together. Let's suppose that they have to make a brainstorming session to design a new advertising board. Amy is working at their main office, but Leela and Philip are not physically present. Leela is working at her home while Philip is in mission in Kenya. In order to be able to work at the same time Amy has sent invitations to Leela and Philip for a virtual Brainstorming with a dedicated software at 3 PM (GMT). In a "device consideration" Amy is working on her usual workstation, Leela has its personal laptop, Philip on his side has a tablet-pc and a smartphone. At 3 Amy has started the server part of the application and has connected her station. At the same time Leela's laptop and Philip's tablet-pc need to connect to the Internet in order to be able to join the Brainstorming platform. To do it they rely on the PCSCW model that should allow their devices to make the right decision.

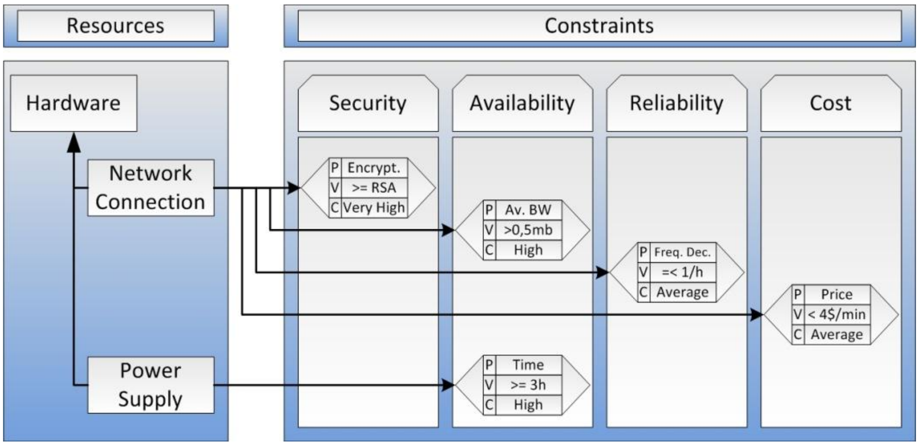

Fig. 2. Internet Connection Constraints

The task associated with the brainstorming activity described with the PCSCW implies several constraints on the resources used by the devices. **Fig. 2** defines and describes the set of constraints associated with the "Connect to Internet" action. On the left side we've got the set of resources required to perform the action and on the right their constraints. For this specific action there are two resources: a network connection and a power supply. For the network connection we've got:

- 1 security constraint: the encryption has to be at least RSA [4]; this constraint has a Very High criticality as the collaboration taking place is close to confidential;
- 1 availability constraint: the average provided bandwidth has to be at least 0,5mbps, this constraint has a High criticality as the application can work with less bandwidth but user's satisfaction and experience may be dramatically lowered by such limitation;
- 1 reliability constraint: the probability to experience network disconnections has to be less than 1 per hour. As this point doesn't completely stop the collaboration it has an Average criticality;
- 1 cost constraint: the price of the connection has to be less than four dollar a minute. As it doesn't obstruct the collaboration this constraint has an Average criticality.

As for the network connection we also have a constraint on the power supply resource:

- 1 availability constraint: the energy supplied has to be sufficient to maintain the connection for three hours in order to have enough time for the brainstorming session. This constraint has a High criticality.

Leela's laptop hasn't many choices and connects itself to the wifi access point of Leela's home's ADSL modem.

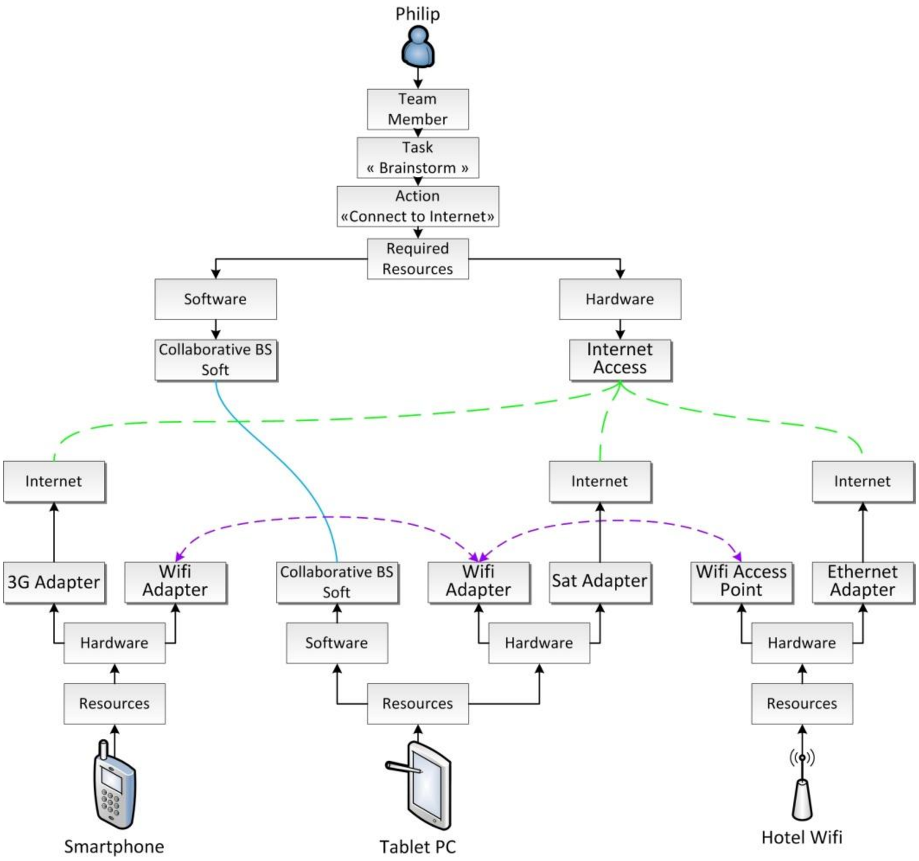

Fig. 3. Philip's Digital Environment

As depicted by **Fig. 3** Philip's situation is totally different. In addition to the tablet-pc, the smartphone and the hotel wifi access point, we've got a description of resources required for the connection to Internet. It also depicts the three possible scenarios to establish the Internet connection:

- Direct connection of the tablet-pc through its satellite network adapter;
- Connection of the tablet with hotel's wifi access point;
- Connection of the tablet with Philip's smartphone with a connection bridge between cellphone's wifi and 3G networks to allow the tablet to access to Internet.

Each one of these possibilities has advantages and drawbacks:

- Direct connection with satellite network:

  - Advantages: highly secured, only rely on tablet's energy, relatively stable;
  - Drawbacks: slow connection (~0,2mbs) and costly, occasional disconnections;
- Connection to hotel's wifi:
  - Advantages: good bandwidth(~2mbps), free, low energy consuming;
  - Drawbacks: poorly secured (WPA), variable bandwidth, disconnections every fifteen minutes;
- Connection with smartphone:
  - Advantages: as we use ad-hoc wifi the security is up to the two devices and can be relatively good, the average bandwidth is fair (~1 mbps) and the connection is relatively stable;
  - Drawbacks: power supply is limited by smartphone's battery life which is limited to 2.8 hours due to the high energy consumption of the 3G and wifi adapters.

We consider that Philip's tablet has already acquired all these information; he must now find the best solution. This is simply realized by analyzing solutions constraints fulfillments such as displayed by **Table 2**.

	Network Connection				*Power Supply*	*"Score"*
	Security	*Availability*	*Reliability*	*Cost*	*Availability*	
Satellite	5	2	2	0	4	13
Hotel Wifi	1	4	0	3	4	12
Smartphone	4	4	3	2	2	15

Table 2 Comparison of Connections

In our case, even if the score are relatively close, the smartphone scenario offers more advantages than others as it combine a good security, availability and reliability for a limited cost. For the battery life, as it is not a mandatory constraint and given that the smartphone can maintain the connection during more than 90% of the desired time with the eventual possibility for Philip to simply plug its charger, it offers a good compromise.

Finally, after having evaluated the situation, Philip's tablet decides to establish a wifi ad-hoc connection with his smartphone and create a bridge between the 3G and wifi connections. Philip can now connect to the brainstorming platform and collaborate with Amy and Leela.

5 Developing Device Collaboration Rules

As we have described all concepts of the PCSCW model and have illustrated it through a use case, the last point we'd like to present here is a way to implement and use the PCSCW model and its collaboration rules.

In this perspective we needed a way to simply and efficiently represent all concepts of the model and reason on it. From our past experiences ([2], [3]) on representing context information and modeling reasoning rules we decided to use the combination of OWL [5] and SWRL [6].

The Web Ontology Language (OWL) is a language for defining and instantiating ontologies, and it can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. Web Ontology Language was adopted as the recommendation by W3C in 2004. OWL provides the required elements to represent and use complex information models. The language itself is an extension of the RDFS [9] language and provides additional features for a greater expressiveness. OWL can be used to define classes and properties and also provides constructs to create new class descriptions as logical combinations (intersections, unions, or complements) of other classes, define cardinality restrictions on properties and so on.

The Semantic Web Rule Language (SWRL) is based on a combination of the OWL DL and OWL Lite sublanguages of the OWL Web Ontology Language with the Unary/Binary Datalog RuleML [11] sublanguages of the Rule Markup Language. It proposes the specification of rules in the form of an implication. The Atoms within the body and the head of the implication can be Class Predicates C(x) or Property Predicates P(x, y). Within the body or head, multiple atoms are treated as a conjunction. In order to use and manipulate the OWL and SWRL languages we rely on the Protégé [8] framework which is currently developed and maintained at the Stanford University. The reasoning itself is ensured through the Jess [10] rule engine which can be used within Protégé.

In a first time, let's see how the basic elements of the model are represented, then how collaboration rules are built on it.

5.1 Roles, Tasks, Actions, Resources and Constraints

The first step to use the basic elements of the PCSCW Model is to represent each of them and their relations. **Fig. 4** provides a view of the implementation of the PCSCW Model that has been made with the OWL language.

On the top of the figure we can see the representation of an *Agent* as an OWL Class. As depicted an *agent* can be a *human* or a *device*. These *agents* can form groups by associating themselves to accomplish a specific *Task*.

Between the concepts of *Agent* (and *Groups*) and *tasks* we've got the *Role* concept. *Agents* and *Groups* are linked to *Roles* by the "*plays*" object property. Obviously *agents* and *groups* can be related with several *roles*.

Considering links between *roles* and *tasks* there are 3 different object properties involved: *mandatory*, *allowed* and *forbidden*.

The representation of *tasks* is relatively simple: a *task* may be composed of several *actions* or/and sub*tasks*.

Actions are mainly composed of requirements: they require *resources* and may also require the completion of some other *actions*. To organize them inside *tasks*, *actions* have a relative *order* as datatype property.

At this point we get back in the resource model by the description of *resources*. A *resource* can be required by an *action* or provided by an *agent*. A *resource* can also be *composed of* other *resources* and can naturally have some specific datatype properties. *Resources* can mainly be of three types: *human resources*, *device* *resources* and *other data* (such as current time, number of inhabitants in Springfield, all information that can't directly be related to an *agent*).

Finally, *resources* may have *constraints* that contain three datatype properties: *parameter, value* and *criticality.*

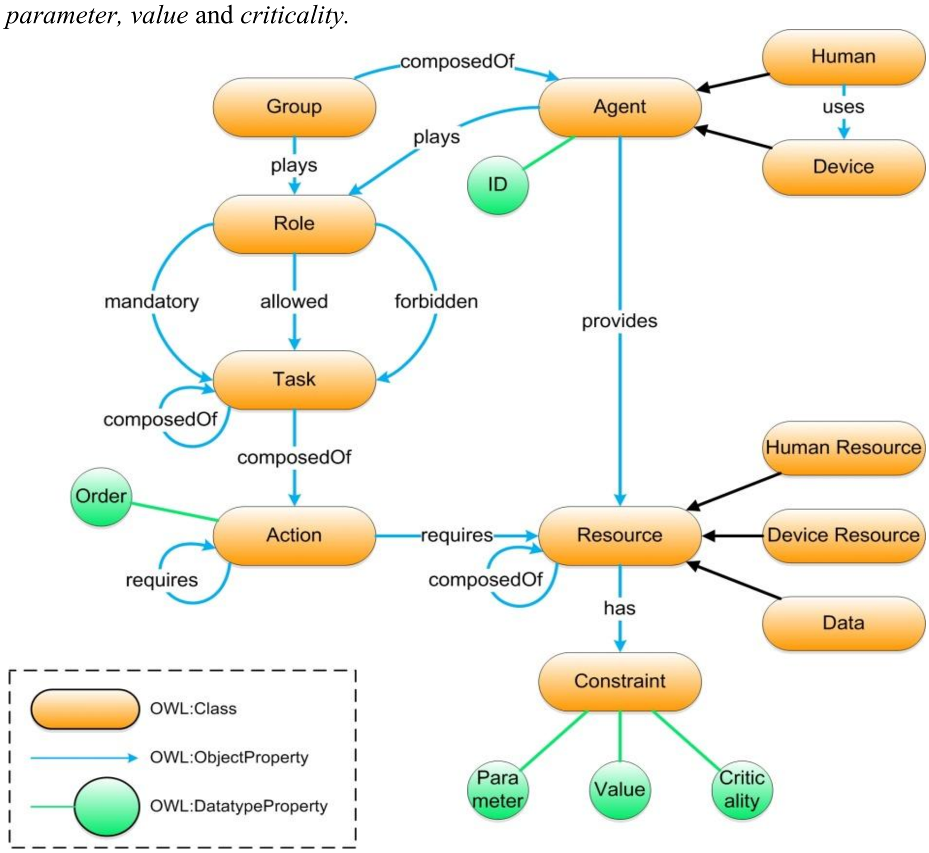

Fig. 4. OWL representation of the PCSCW Model

We have presented how the PCSCW model is mapped to an ontology with the OWL language. Hence the next step is to see how collaboration rules are represented.

5.2 Collaboration Rules

As it has been already evoked we have decided to use the SWRL language to represent device collaboration rules for its integration with OWL and its efficiency.

If we refer to (1) we know that devices collaboration rules are basically a comparison of required resources against available ones and a resulting behaviour. Then, as SWRL Rules are composed of an antecedent and a consequent, each of whom is containing a set of atoms the natural way to represent collaboration is to define required resources as the antecedent and resulting behaviour as the consequent.

$$\text{RES}_{r1}(r_{r1}?) \wedge \ldots \text{RES}_{rn}(r_{rn}?) \wedge \text{ACT}_{r1}(a_{r1}?) \ldots \text{ACT}_{ro}(a_{ro}?)$$

$$\Rightarrow \quad (2)$$

$$\mathrm{RES}_{i1}(r_{i1}?) \wedge \ldots \mathrm{RES}_{ip}(r_{ip}?) \wedge \mathrm{ACT}_{i1}(a_{i1}?) \ldots \mathrm{ACT}_{it}(a_{it}?)$$

Formula (2) summarizes the expression of collaboration in SWRL: the antecedent contains a set of resources and a set of actions representing the requirements while the consequent stores actions with their relative resources representing the behaviour of the rule.

During the definition of device collaboration rules we mentioned the fact that all device collaboration rules must have the same set of context information. To fulfil this requirement we rely on internal mechanisms of Protégé which are able to load knowledge inside a “Rule Engine Bridge” which has in charge to make the interface between the core of Protégé and the current rule engine. As we load context information only once for all rules (and each time we need to reason) it ensures that all rules use the same set of knowledge.

6 Related Works

We have presented our work on pervasive computing supported collaborative work. Obviously it hasn’t come out of nowhere and there are numerous researches that inspired us. In this section we present some of these works related to the context modelling and reasoning.

In [12] the authors present a rough set based methodology [16] to generate the appropriate minimal set of design rules for the Ubiquitous Smart Device (USD) design collaboration. They point out the fact that the USD design can be semantically represented in ontology, however, the computational complexity of semantic reasoning is a very sophisticated and time consuming task. They proposed in future work to compare the manually defined SWRL rules and inducted rules to validate their rough set framework.

In [13] the authors propose an ontology-based generic context management model; their model facilitates context reasoning by providing structure for contexts, rules and their semantics. Rules are derived or defined by users based on the requirements and policies of a specific application domain. Their underlying context model has been developed using the RDF and OWL languages. Obviously this model is meant to be extended and enriched by domain specific information according to its use and the requirements of the system. Reasoning rules are composed of two distinct sets: rules based on the ontology itself and user defined rules. Ontology based rules are those concerning the standard features of OWL such as “inverseProperty”, transitivity of properties and so on. Users’ defined can be of any type and express semantic implications in the ontology. This work is particularly interesting as it tries to give a generic model and a way to reason over it.

In a previous work [3] we have proposed a context modeling for communication services based on ontology. This ontology is enhanced into an active model by providing it a rule engine and a set of inference rules. We have used the SWRL rule language to reason on the context model. This mechanism consists in two specific phases. First, in the context model, we have defined a property named "hasAssociatedRule" which domain is Context_Behavior and range is swrl:imp. The class swrl:imp is the one that represents the SWRL Rules. In our ontology rules are directly associated to the classes by the owl restriction owl:hasValue. The second phase is the selection of the rules. This is quite simple since we have the set of rules associated with the classes of the behavior. This preceding work was far simpler than our current one but it gave us a good opportunity to evaluate the feasibility of such reasoning mechanisms and the efficiency of OWL and SWRL.

In [14] the authors propose Smart Device Collaboration for ubiquitous computing environment, which aims to establish the collaboration between portable devices and embedded computers, while realizing the basic function of portable devices and also applying the maximum advantages of embedded computers.

In [15] the authors propose an OWL context ontology (CANON) for modelling context in pervasive computing environment, and for supporting logic-based context reasoning. CANON provides an upper context ontology that captures general concepts about basic context, and also provides extensibility for adding domain-specific ontology in a hierarchical manner. They also studied and implemented the use of logic reasoning to check the consistency of context information, and to reason over low-level.

All these works are interesting, but we have to notice that we found extremely few researches dealing with the modelling of collaboration in a pervasive computing environment.

7 Discussion

In this work we have presented the real engine of our PCSCW Model. Indeed, the model itself is an essential part for our research toward the complete integration of Pervasive Computing within the computer supported collaborative work, it provides solid foundations to represent context information, should they be humans, devices, resources, roles, etc. Still, without reasoning it loses most of its interest. Thus, by completing it with device collaboration rules we dramatically increased its potential and usefulness. Indeed, more than an efficient structure, we've got a full process to decide how devices should cooperate to help users collaborate. Besides, this decision is achieved with no other features than the model, the rules and comparisons with current context information.

Even if devices collaboration rules are efficient, we already have some enhancement trails we'd like to explore. For instance, the description of constraints relies on the definition of criticality levels. We know the actual levels are sufficient for our needs, but we need to evaluate if more levels could bring a better accuracy for rules' selection.

In the perspective of providing a better adaptation to current context we also want to find out if there may be constraints on other elements of the PCSCW model. For this trail we'd like to consider if defining constraints directly on actions and tasks could give relevant information and rules selection refinements.

Until now we have set several theoretical bases, and we are currently focusing on the finalisation of the development of a simulator for pervasive computing collaborative work. This tool will assist us to validate and evaluate our model. Thus it should tell us if our model is actually working and if it really improves the collaboration of users. Besides, in a near future it could help us design collaboration rules by providing a simple way to evaluate and compare them.

Another issue considering the adaptation of devices behaviour to their current context is their possible engagement in another activity. Indeed if the evolution of user's context brings out the necessity to provide him specific resources for the accomplishment of his tasks, it may involve the modification of current devices behaviours and then create a conflict between two concurrent actions. We think this problem is naturally tackled by the PCSCW model as it is used to represent the current state of the context. In this perspective resources currently in use would not be available for the adaptation to context changes. A smarter way to manage potential conflicts it to allow the interruption of some part of current device behaviour in order to allow the accomplishment of a more important action. Such perspective seems not to be too complex as it could possibly be managed through reasoning rules.

A known challenge of our device collaboration rules is their elaboration. Indeed, even if the base principle is relatively simple and writing them with SWRL isn't really complex, the real difficulty is to design small, generic but still efficient rules. For this complex aspect of the life of our model we have planned to analyze real collaboration situations represented with the PCSCW Model and extract those where devices could potentially have cooperated. Then, we should be able to find the most relevant cooperation scenarios and their associated device collaboration rules.

On the long range such device collaboration rules could bring a new opportunity for coworkers, repetitive actions and even tasks could be automatically performed by a team of devices. In this perspective, if group of devices are able to automatically perform tasks, then they should be given a role. Going a little further with this idea we'll have to formalize how devices and group of devices can play some specific roles in the collaboration. Besides, we also have to consider that devices and humans can be mixed within a single role.

A very noticeable and original aspect of our work is the approach of the pervasive computing from the collaboration point of view. Indeed, even if there are numerous works related to the development of ontologies and systems to model and support the pervasive computing paradigm for users, there are very few works focusing on the computer supported collaborative work. Considering this lack we think our work around the PCSCW model can sets some bases to fully integrate the pervasive computing model within the collaboration of users.

One of the last points to evoke here is the evolution of rules. Indeed, the device collaboration rules used will doubtlessly need to evolve along with the life of the collaboration to adapt themselves to collaboration patterns and evolution of devices. In this perspective this evolution has to be at least partially dynamic; that is to say the system has to be able to evolve without the intervention of humans. This evolution may involve the modification of the ontology representing resources and roles and the modification of device collaboration rules. To allow this dynamicity we think the best way is the development of a knowledge deriving and learning mechanism. These mechanisms consist in the capacity of the system to infer and extrapolate context information and adapt device collaboration rules. Besides, the learning mechanism shall allow devices to use new collaboration rules extracted from extrapolation of already existing rules but also learn new collaboration rules from other devices. These learning mechanisms should allow the PCSCW model and its collaboration rules to evolve autonomously.

The PCSCW model and its associated collaboration rules form a coherent and consistent base allowing smart devices to continuously interact with each other and adapt their behaviour to users' context and collaboration. This base makes us foresee even more challenges toward the real ambient intelligence but strengthen us for our future work.

References

1. Hamadache, K., Lancieri, L.: Role-Based Collaboration Extended to Pervasive Computing. In Proceedings of the International Conference on Intelligent Networking and Collaborative Systems, Barcelona, Spain, Nov. 04-06, (2009)
2. Hamadache, K., Manson, P., Lancieri, L.: Pervasive services, brainstorming in situation of mobility. In: 3rd International Conference on Pervasive Computing and Applications. Alexandria, Egypt, pp. 709-714. (2008)
3. Hamadache, K., Bertin, M., Bouchacourt, A., Benyahia, I.: Context-aware communication services: an ontology based approach. In: 2nd International Conference on Digital Information Management (ICDIM'07). October, Lyon, France. (2007)
4. RSA patent: http://www.google.com/patents?q=4405829
5. OWL: http://www.w3.org/TR/owl-features/
6. SWRL: http://www.w3.org/Submission/SWRL/
7. Weiser, M.**:** The Computer of the 21st Century», Scientific American*,* vol. 265, no. 3, September. 1991, pp. 66-75. (1991)
8. Protégé: http://protege.stanford.edu/
9. RDFS: http://www.w3.org/TR/rdf-schema/
10. Jess: http://www.jessrules.com/jess/index.shtml
11. RuleML: http://www.dfki.uni-kl.de/ruleml/.
12. Kyoung, K., Keunho, C., Ohbyung, K.: Rule Selection for Collaborative Ubiquitous Smart Device Development: Rough Set Based Approach. UIC 2008: 386-396, (2008)
13. Dejene, E., Vasile, S., Lionel, B.: An Ontology-Based Approach to Context Modeling and Reasoning in Pervasive Computing. In: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2007: 14-19 (2007)
14. Naohiko, K., Kenta, M., Kazunori, T., Hideyuki, T.: Smart Device Collaboration for Ubiquitous Computing Environment , UbiComp'2003 (2003)
15. Xiao, H.W., Da Qing, Z., Tao, G., Hung K. P.: Ontology based context modelling and reasoning using OWL. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops, PERCOMW'2004. (2004)
16. Pawlak, Z.: Rough sets- Theoretical aspects of reasoning about data. Kluwer Academic Publishers, Dordrecht (1991).