LWOAD: A Specification Language to Enable the End-User Develoment of Coordinative Functionalities

Federico Cabitza and Carla Simone

Università degli Studi di Milano-Bicocca,
viale Sarca 336, 20126 Milano (Italy)
{cabitza,simone}@disco.unimib.it

Abstract. In this paper, we present an observational case study at a major teaching hospital, which both inspired and gave us valuable feedback on the design and development of LWOAD. LWOAD is a denotational language we propose to support users of an electronic document system in declaratively expressing, specifying and implementing computational mechanisms that fulfill coordinative requirements. Our focus addresses (a) the user-friendly and formal expression of local coordinative practices; (b) the agile mocking-up of corresponding functionalities; (c) the full deployment of coordination-oriented and context-aware behaviors into legacy electronic document systems. We give examples of LWOAD mechanisms taken from the case study and discuss their impact for the EUD of coordinative functionalities.

1 Requirements for EUD in Document-Mediated Cooperative Work

The fact that documents are ubiquitous means to support work activities is well known. Their initially undifferentiated role has been more recently investigated and articulated to understand why documents, which are so natural and widespread, still raise problems when they are transformed in digitized counterparts, not only when electronic documents are used as stand-alone artifacts but, above all, when they are parts and components of an *electronic document system* [1,2]. The solution of this paradox calls for a stronger user involvement in the definition and maintenance of functionalities that support actors in accomplishing their duties and coordinating their action; these functionalities relate closely to how users read and write their paper-based artifacts and to the often only implicit and ad-hoc practices and conventions that regard documents' use and interpretation. A very inspiring domain where to motivate this claim and highlight requirements for an EUD-based solution is the healthcare domain. This is, on the one hand, so complex and various that almost all considerations emerged from other cooperative domains apply naturally (e.g., [3,4]); on the other hand, this domain has been widely studied and specialist literature has provided interesting findings to leverage. For instance, in his comprehensive account on the role of documents in professional work, Hertzum [5] points out the

V. Pipek et al. (Eds.): IS-EUD 2009, LNCS 5435, pp. 146–165, 2009.

paradigmatic case of the *patient record*, in regard to both its *many-sidedness* in supporting cooperative work and its ability to speak different "voices", i.e., to convey different meanings according to the actor using it (e.g., doctor, nurse). According to Garfinkel [6], the patient record contains at least two clear intertwined voices: a voice reporting what clinician did what to inpatients; and another voice attesting that clinicians have honored claims for adequate medical care. Following in the same footsteps, Berg distinguishes between the coordinative and accumulative function [7] of patient records, respectively. Patient records exhibit the accumulative function whenever they play the role of official, inscribed artifacts that practitioners write to preserve memory or knowledge of facts or events occurred at the hospital ward. Patient records exhibit the coordinative function whenever they are used to support articulation and coordination of the work activities which are tightly coupled with data production and consumption. A very important point is that the accumulative function can refer to either a long-term role of records – typically when patient's data are *archived* for research or statistical purposes – or to a short-term role– typically when these data are memorized to *keep trace* of the care trajectory during the patient's hospital stay. This latter role is necessarily entangled with coordinative functions in not always trivial ways [8,9]. Accordingly, the specialist literature distinguishes between *primary* and *secondary* purposes, respectively. Primary purposes regard the demands for autonomy and support of practitioners involved in the direct and daily care of inpatients; while secondary purposes are the main focus of hospital management, which pursues them for the sake of rationalizing care provision and enabling clinical research. The investment policies in ICT are usually focused on secondary purposes (i.e., on cost savings and new pharmacological patents) and this leads to the design of Electronic Patient Records (EPR) where document structures and functionalities are aimed at supporting information inscription and use according to data quality and usability criteria [10], which tend to neglect (or heavily overlook) the primary purposes [11]. The additional effort of articulation work on the clinical record is then usually left to practitioners; as well as, often, the burden to reconfigure their coordinative practices once their habitual paper-based artifacts have been digitized [9].

In this scenario, document templates and masks are usually imposed from above to practitioners, irrespectively of their coordinative needs. Even in the best case where documents are cooperatively and participatorily defined, they tend to be given to actors once and for all so as to neglect the frequent tuning activities and adjustments that coordinative mechanisms require for their negotiated and participated nature [12,13]. Our observational studies in two wards of a large provincial hospital in Northern Italy confirmed other accounts from the CSCW literature (e.g., [14,7,8] on how practitioners try to reconcile primary and secondary purposes on the artifacts of daily use to make them useful both to store and retrieve information but also to support mutual learning, knowledge sharing and coordination of caring activities. To this aim, actors define, renegotiate and evolve ad-hoc practices, peculiar conventions, and agreed interpretations that are local and unique to their work settings; usually, these

conventions thrive either in the grey area of underspecification or through the mesh of the constraining specifications of organizational rules that the hospital management has imposed for law or quality standard compliance.

Our point is that the development of any technological support of the full usage of official records cannot do without considering these *local habits* and *conventions* as a primary source of information for the definition of functionalities that support cooperation and, to limit ourselves to our reference domain, effective care giving. Moreover, since these habits and conventions are *local and unique*, the technological support should provide *flexible* functionalities that preserve, or even foster the fluid, conventional and evolutionary nature of coordinative practices. Last but not least, these functionalities should be under the full control of actors themselves. Our research question is then how to facilitate this local management in a sustainable way. The paper aims to give a contribution in this direction by presenting a computational framework that was deeply influenced and partially tested by our field study in the above mentioned hospital [15]. The next section describes the field study context and provides the main motivations for the framework; Section 3 describes the framework and its denotational language, LWOAD, in more details; Section 4 illustrates the complex and real-life conventions we tested the framework on; Section 5 discusses the main findings of the case study; Section 6 illustrates the mockup we used as a proof-of-concept of the findings related to mechanism specification and Section 7 sums things up and sheds light on current and future directions of our research in the EUD field.

2 Bridging Conventions and EPR Applications

Our empirical research involved doctors and nurses of a major Neonatal Intensive Care Unit (NICU). We conducted this study through unobtrusive observations in the ward, informal talks, individual interviews with key doctors and nurses, and open group discussions with ward practitioners. These interactions were initially used to deal with the "descriptive" part of the research, and to reach a reasonable and common language. Yet, quite soon, the need emerged to find a more effective way to deal with what we called "local habits and conventions" in the previous section. In parallel with our investigation in the ward, practitioners had to interact with a team of software developers of a third-party IT firm, the Alpha ltd. The NICU head physician had been involved for months with this firm to produce an innovative EPR for the care of his premature newborns: a job order he was totally in charge of, with no time pressures and the concrete willingness to create a solution on the practitioners' side. Since Alpha designers had already developed a full-fledged prototype of a hospital-wide EPR, the NICU head physician soon adopted that prototype as a sort of "sandbox" where the programmer analysts of Alpha and some physicians of the NICU could experiment their innovative and peculiar ideas with no claim of officiality or exactitude. While Alpha's analysts were more oriented to the archival functionalities, we concentrated on the coordinative ones: together with the practitioners, we took the design of a

new EPR as the occasion to try to preserve as many as possible efficient, though idiosyncratic, coordinative practices and "graft" them onto the archival-oriented EPR by conceiving coordinative functionalities on-top-of it.

To this aim, we felt the need to develop specific ways (a) to express coordination-oriented requirements in a user-friendly manner for ICT laymen (as clinicians were); and (b) to formalize the corresponding functionalities in a way that they could become easily computable. Our goal was to support the effort of practitioners in making explicit, symbolic and also computable the relationship occurring between recurrent patterns of context and the conventional, local ways practitioners relied on to cope with this context. During the requirement collection and preliminary analysis, we observed that the simplest, and yet powerful, concept that practitioners grasped with fewer equivocations was that of *reactive behavior* and its computable counterpart, the *rule*, i.e., a well-defined and autonomous *if-then* statement (see Figure 1).

This finding heavily influenced how we were conceiving the WOAD framework [15] (an acronym for 'Web of Documental Artifacts'). WOAD is a design framework we were articulating during the NICU case study in order to bridge the gap between informal description of coordinative conventions – expressed in terms of agreed ways to cope with the current context – and the design of document-mediated functionalities supportive of these conventions. Our point was that practitioners themselves could bridge this gap, if the the computer-based support could provide them unobtrusive and additional information to promote collaboration awareness [16,17]. In the WOAD framework, we defined (a) a conceptual model of articulation by which to characterize the main entities and relationships involved in document-mediated cooperative work in terms of minimal sets of attributes; (b) a denotational language – LWOAD – which incorporates those concepts and relationships to represent the context of cooperative document domains and conceive specific computational mechanisms that convey Awareness Promoting Information (API) depending on the current context [17]; and (c) a high-level architecture for information sharing in context-aware and distributed computing settings where LWOAD is used and implemented. Since LWOAD plays a basic role in making the specification of rules expressing the above mentioned relationships computable, we briefly introduce it and discuss how we used it in our interaction with the NICU practitioners.

3 A Language to Express Coordinative Functionalities

Within the WOAD framework, LWOAD provides a set of high-level concepts – like those of *actor*, *documental artifact*, *fact*, *fact space*, and *fact-interpreter* – that we propose to guide the design of a rule-based reference architecture for context-aware and coordination-oriented electronic document systems. We conceive LWOAD as an astract programming interface by which to program functionalities that (a) process the content of a document according to local conventions of coordination; and (b) convey suitable API to support actors in articulating their document-centered activities.

LWOAD encompasses a set of both static and dynamic constructs by which designers can express either contextual, organizational or procedural knowledge about a work arrangement. Static data structures and dynamic behaviors of an application are expressed by two specific constructs: *facts* and *mechanisms*, respectively. In LWOAD, designers can model a cooperative arrangement in terms of its main relevant entities and the relationships between them by declaring *facts*. Whatever is given the suffix *-fact* (e.g., *activity-fact*, *relation-fact* and *API-fact*) is a *key-value* data structures, which programmers can use to characterize the relevant entities of a documental domain just assigning a value to their specific attributes. A *relation-fact*, for instance, is characterized by five attributes: i) a name, ii) a description, iii) a property telling whether the relation-fact indicates a relationship between classes (e.g., physicians and patients) or between instances (e.g., Dr Smith and Mr Jones), iv) an attribute that specifies the fact's name of the entity that is the source (i.e., the subject) of the relationship and v) an attribute specifying the target (i.e., object) entity of the relationship. LWOAD provides designers with templates (i.e., *entity-facts*) for the most generic categories of articulation work (cf. [13]), like those of actor, activity and artifact; yet, by means of the *extends* primitive, designers can also define domain-specific entities (such as patient, doctor and clinical activity) that specialize and inherit from those general categories.

Mechanisms can be seen as simple conditional statements, like *if-then* rules. They produce some output in virtue of the actions expressed in their *consequent* (the *then* part) whenever specific contextual conditions, which are expressed in their *antecedent* (the *if* part), are true. Antecedents are considered true whenever the conditions they express on the entities they refer to are met by the current WOAD-compliant representation of the context, i.e., by the content of the facts represented within the so called *fact space*. Any single mechanism is hence a symbolic way to make a relation explicit between some contextual conditions and some functionality that the system should exhibit whenever a specific case occurs.

Although symbolic and based on rules, LWOAD is far from being usable directly by practitioners, since it must comply with the typical syntactic constraints of a language interpretable by a computational engine. For this

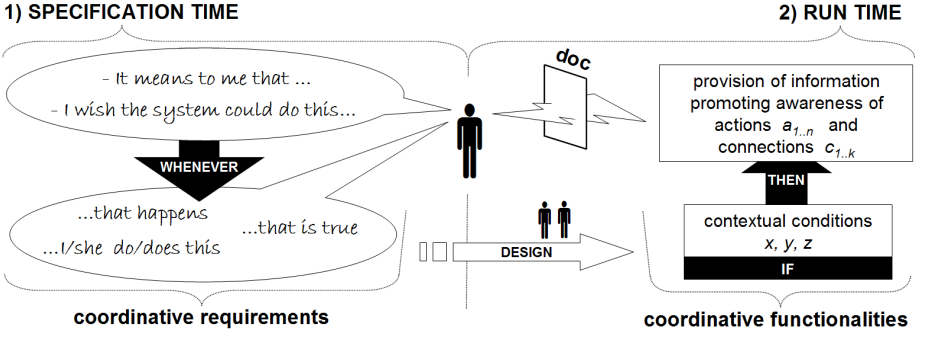

Fig. 1. The design-implementation loop inspired by the case study

reason, we adopted a two-step approach: first, for each identified coordinative mechanism, we invited the practitioners to indicate both the relevant set of attributes of the domain entities, events and documental data that the computational system should be sensitive to, and what conditions the system should evaluate on these relevant aspects of the work setting in order to activate the desired functionalities. The practitioners expressed the mechanisms in natural, tough structured and restricted, language and we translated them "on the fly' in LWOAD statements. In doing so, we could rapidly convey the "flavor" of the coordinative mechanisms envisioned by the NICU practitioners and we could support them in deciding whether the mechanisms had to be fully implemented in the hypothetical final release of the EPR. Once they had become familiar with this way of describing the desired mechanisms and had identified the basic patterns of conditions, we mocked-up an interface based on a wizard that could support practitioners in the construction of mechanisms (more details in Section 6). Our goal was to check whether the practitioners had become proficient in defining the desired mechanisms *autonomously*. Then, the needed mechanisms were translated in LWOAD to check its correctness with respect to both the application conditions and the desired outputs. Generally, practitioners did not find problems in expressing LWOAD mechanisms, probably for their intrinsic simplicity: antecedents are constituted by fact patterns and boolean tests and practitioners found natural express them as conjunction of facts that must be true in a given situation. On the other hand, consequents are sequences of WOAD primitives and practitioners mastered them in a relatively short time since our design choice was to limit LWOAD to the expression of functionalities that promote collaboration awareness and do not manipulate the data managed in the archival dimension of the EPR. Therefore, the effects of the consequents were only graphical cues added on top of documents' data. In a parallel work [17], we classified up to 13 different types of API – e.g., criticality, revision and schedule awareness – and identified together with the practitioners graphical ways to convey those various kinds of information. Although this latter identification is not completely experimented, it constituted a basis where to get an initial impression of how well the interface could be usable by practitioners on their own. In the next section, we provide the reader with two examples of mechanism specification in response to the specific coordinative requirements that we identified with the NICU practitioners during the first phase of our study.

4 Coordinative Requirements for EPRs

The patient record is the main documental artifact used in hospital care as it is the composite repository for the whole information concerning a single patient stay. During a patient stay, the whole patient record is split up into several sheets and documents; these are distributed in the ward and are very specific for a certain aspect of care so as to be usually used by different actors at the same time. During the study at NICU, practitioners recognized the need to conceive functionalities supportive of the conventional ways by which artifacts were used

both to document their work and to mutually articulate their activities with each other. In the what follows, we present some cases that call for the requirements of a *flexible definition* and *flexible combination* of coordinative functionalities.

Specification Flexibility for Structure-Related Conventions. Due to the fact that clinical data are usually scattered across multiple artifacts in different places, doctors at NICU found useful to rely on a summary of clinical data that are gathered into one single sheet that they call *Summary Sheet* (SS); they update the SS quite frequently by taking and synchronizing its content from the official patient record. The summary sheet is not part of the official patient record but, nevertheless, it is a very useful working document since it is often used to jot down offhand annotations and informal communications regarding clinical conditions of the patient at hand. Moreover, due to its informality, doctors are used to bringing the SS with them either as first page of sets of papers under their arm, or even folded in their pocket. Therefore, since the SS is usually the first document doctors have got in their hand during their hectic activities, they also use it to jot clinical data and prescriptions on-the-go, which they will have to replicate into the official record later as a rule of law. Hence, the summary sheet is not only a "passive view" of previously reported data, i.e., a *view* on data fetched by querying multiple tables of a clinical database on the illness course of a single patient. It is also an active *entry form*, into which practitioners insert data at the point of care and from which they copy data into the official records for the sake of accountability and liability. Doctors were well aware of this twofold functionality during the design of the digital counterpart of the SS into the "innovative" EPR; therefore, they were willing to express constraints and define conceptual connections between sections and fields of the summary sheet and corresponding sections and fields of the artifacts compounding the patient record. These connections were seen as symmetrical, i.e., equivalent and irrespective of where the *original* data were actually inserted first. They can be traced back to the class of connections that in [18] we denoted as enabling "*redundancy by duplicated data*", in that they make the association explicit between identical data that are reported in two or more documents of the patient record. In regard to the requirements for supportive functionalities, these kind of connections regard conventions of production and use of clinical documents: more specifically, they regard how data are organized within templates, what data type are allowed in what field (i.e., syntactic integrity) and also where people fill in data during their situated documental activities. Moreover, these connections are local and conventional both in their definition and, above all, in their use. In fact, it is only on a conventional and context-dependent basis that doctors want summary sheets be completely compiled *after* the patient record and, conversely, values reported in the SS first be fed *into* the patient record at proper time.

Two examples can better illustrate this point. Some members of the NICU staff team expressed the requirement that values on the weight of newborns would be reported into the summary sheet *only* whenever a newborn was in life-threatening conditions. In fact only in that case, these practitioners deemed

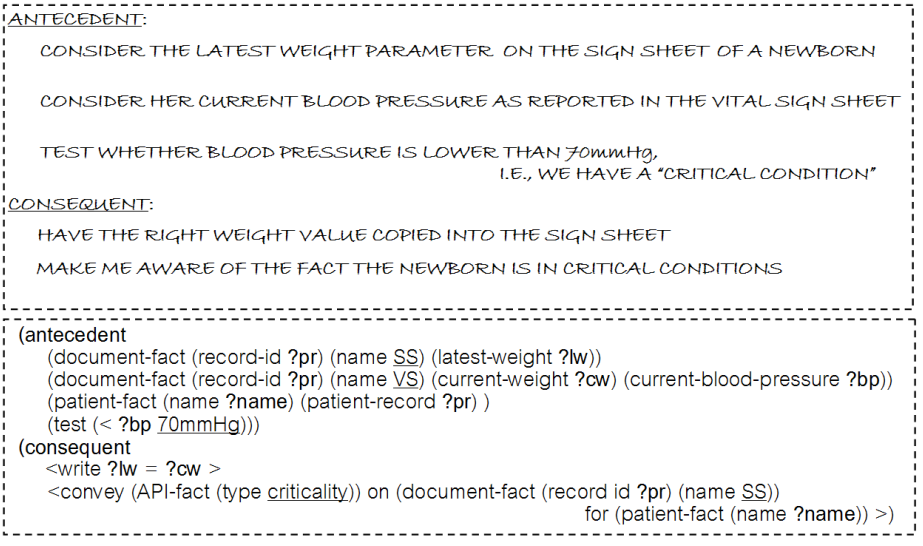

Fig. 2. A coordinative mechanism on conventional patterns of data redundancy. Above, as expressed by practitioners in their own terms. Below, how this is translated in terms of LWOAD facts and primitives.

necessary to rely on weight data at the point-of-care, so as to calculate drug dosage precisely. In the other cases, to have these data available on the SS would only result in an unnecessary information overload and, even more annoying, would undermine the role of unobtrusive reminder on critical conditions that the presence or absence of weight data in the summary sheet could play at the point of care. Likewise, at NICU, clinical data that are reported into the SS first are often deemed as still provisional and are reported there to have colleagues consider those data but also take them as not yet definitive, or even as an invitation for further check and inquiry. The need for doctors to be aware of what is still provisional and hence different from what constitutes an unmodifiable and legal account of accomplished deeds is essential to cooperatively structure the formation of decisions and judgments, as also reported in [19].

Figure 2 depicts how the above mentioned conventions on data replication have been expressed in a dedicated and concise LWOAD mechanism. This mechanism has in its antecedent all and only the relevant aspects of context that are concerned with the coordinative functionality expressed in the consequent. While practitioners expressed this subset of contextual information in their own terms, we translated the consequent into four conditional elements, i.e., namely three patterns and an inequality test. The reason why even what seems a quite objective and scientific threshold of blood pressure is consider "conventional" (and hence ward- if not doctor-specific) is worthy a reflection. Quite surprisingly, doctors told us that also the notion of "critical condition" changes according to a number of contextual aspects that are mostly neglected by monitoring devices: their alarms are most of times consciously and rightly ignored by expert nurses, as reported in [20]. For this reason, doctors believe that these conditions are

utterly difficult to hardwire into procedural application logic in all but the most obvious cases. In fact, criticality – seen along the coordinative dimension as the condition of a patient that calls for a direct and immediate intervention of some practitioner – depends on several anamnestic and physiological elements, on the illness history of the patient, and also on even more situated aspects, like the attitude of attending practitioners and their current workload. This is an important point to challenge LWOAD against the requirement of flexible definition of mechanisms. Obviously, not all the above often-tacit contextual conditions can be immediately and comprehensively externalized into a mechanism and neither should they be: however, as long as recognizing a specific situation has a relevant coordinative value, practitioners can be motivated in characterizing it formally, by relying on some shared and broader conventional interpretation of data combinations or on the mutual acquaintance of the involved actors. In all these cases, the highly incremental structure and computational autonomy of mechanisms (in terms of their inner components and role in the control flow of the application, respectively) can facilitate stakeholders in expressing and updating mechanisms that are quite specific to complex and ever new situations. For instance, if the NICU practitioners had expressed the need not to be alerted for low pressure problems of their inpatients unless in more specific cases than that represented in Figure 2, the antecedent of that mechanism would have been enriched with a new combination of conditional elements: e.g., a test to evaluate whether the basal and physiologic blood pressure of the newborn is usually low, or whether she has been already treated for low pressure after the onset of the criticality, or even whether the latest drug that had been administered to her brings low pressure normally. The progressive tuning of coordinative requirements would not require a major rewriting of the application logic behind the corresponding functionality, but just call for the addition (or deletion) of specific conditional elements within the mechanism that triggers the provision of criticality API on those critical conditions.

Combination Flexibility For Run-time Connections. As said above, NICU practitioners expressed the need that executable mechanisms could be easy to define and modify. In addition to that, they also expressed the need the application (i.e., execution) of these constructs be dependent on the current context. In regards to this requirement, which is in the line of the major tenets of context-aware computing [21], they needed to conceive ways to manage connections that had been explicitly instantiated between data during their daily activities, and not just at schema level and at compile time as in the previous case. Thinking in terms of rules assured them that the whole set of mechanisms, once specified as a whole, is "rescaled" each time into smaller active subsets, i.e., those mechanisms whose antecedent is satisfied according to what actors do (as to any other contextual event). In fact, even multi-condition mechanisms – i.e., mechanisms that are very specific to a given situation – are considered for execution just when all their conditions are true; this releases practitioners from conceiving an arbitrarily long sequential flow of control in which this kind of mechanisms are discarded in all cases but that very specific situation. This

flexibility was deemed useful especially in the case of connections that were created at run-time across artifacts of the patient record, such as the *problem list* and the *doctors' diary*.

The *Problem List* (PL) is the artifact of the patient record where clinicians enumerate the patient's problems. This list is intended to document all those conditions and events that can be related to clinical hypotheses and procedures. The term "problem" is purposely left vague enough to comprise a number of factors like symptoms, any alterations to vital signs, and all the concomitant pathologies that could affect a patient's hospitalization. The PL is likely to change during the caring process since practitioners are supposed to update its content with respect to the actual improvements or aggravations exhibited by the patient but also with respect to the extent they can consolidate their diagnostic hypothesis. Therefore, the PL is more than a mere list of either concomitant or sequential problems affecting the patient: it is the artifact where doctors represent the main deviations and swerves of illness trajectories, and the results of the epicrises (i.e., summings up) doctors periodically accomplish in evolving and improving their diagnosis on a specific case. The epicrises can result in the need to "cross out" previously unrelated symptoms and substitute them with new comprehensive diagnostic items. On the other hand, changes that regard the acuteness of single problems previously stated are not represented into the PL explicitly. These are rather represented in the *Doctors's Diary* (DD). The DD is the central repository for the notes that physicians need to write down in order to account for the decisions and interventions they are responsible for, as well as to make impressions, opinions, or just lines of reasoning explicit, either for themselves as memorandum or as written notes to other colleagues.

The physicians called our attention on how useful would be for them to be capable of making explicit on the record itself the relationships between past problems and new problems as well as between problems of the PL and the daily entries reported into the DD. The former capability was seen as a way to reconstruct or, better yet, make the line of thought explicit by which symptoms have been rationalized into problems and unrelated problems into precise diagnosis. The latter was seen as a way to facilitate the *a posteriori* reconstruction of a problem progress from its outset, in order to give indications on how to head the course of clinical interventions towards its conclusion. These requirements point to a relevant coordinative need, besides that of keeping trace of relevant phases during the decisional/medical process: in fact doctors were also, sometimes implicitly, expressing the need to be informed on what problems they should address first and on the way their colleagues had coped with these problems that far.

We then asked practitioners which kinds of relationship they would more naturally employ to join two or more data that are not explicitly correlated by the patient record structure. The result was that practitioners found more natural to consider relationships as occurring between data entries, either already recorded or still to record on the patient record. In the former case, they pointed out the usefulness to relate data over distributed and different artifacts; in the latter case,

they referred to the capability to draw relationships between data *values* and *fields* yet to fill in, that is between documental activities and articulated work activities still to perform. While almost any doctor expressed her preference for a number of possible relationships that had small overlap (if any) with those pointed out by the others, we noticed that when these relationships were actually applied in the field of work, they all blur into three main categories: *causal*, *temporal* and *intentional* connections [18]. The generic semantics that pertain to the nature of the relationships between a source information and a target information could then be respectively rendered as: (a) "the source *because* of the target"; doctors would use this connection in order to hint a strict causal relationship between items of the patient record: e.g., the diagnosis 'pneumonia' – reported in the PL – can be indicated as cause of the symptom 'cough' – reported in the DD – as a way to explain the symptom itself. (b) "the source *after* the target"; doctors would use this connection not only in strict temporal sense, but also to hint a very weak or just supposed causal relationship: e.g., reporting that a skin rash – a symptom from the DD – occurred after having administered a drug – an order reported somewhere else in the PR – would indicate a hypothesized correlation between these two clinical facts. And (c) "the source *for* the target", that doctors would use in order to highlight evidence supporting a particular decision or to make an intention explicit (e.g., that the bacterial culture – an order – has been prescribed to verify the diagnostic hypothesis of pneumonia – an item in the PL).

Figure 3 depicts how the need to be aware of impromptu connections (i.e., relationships) that were previously drawn by colleagues was computationally rendered in WOAD-compliant statements by practitioners with our support. The mechanism is sensitive to whether a connection exists between a specific entry and another entry anywhere else in the PR. Only whenever this situation occurs, the WOAD interpreter executes an instruction by which an API is conveyed to the actor through the form she is currently using (see last statement in Figure 3). This general mechanism can be made more specific in its antecedent by adding to the pattern for the relation-fact the explicit indication of the type of relationship

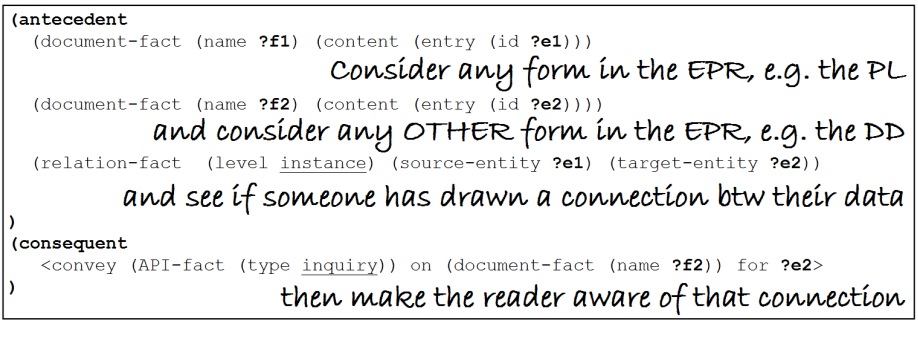

Fig. 3. A list-like representation of the mechanism of run-time creation of data connections

(e.g., causal) to be sensitive to. Likewise, designers can specify in the consequent what API to convey in relation to the kind of correlation.

5 LWOAD and the Flexible Specification of Coordinative Functionalities

LWOAD was presented to the clinicians as a sort of *specification language* by which to implement their coordinative requirements. These were intended to characterize an EPR that would not hinder, but rather foster, patterns of cooperative behaviors on the basis of how actors use official records and documents in their daily practice. The fact that users could be facilitated in "rapidly having a taste of a functionality" (as suggestively said by an interviewee) called for the twofold requirement that coordinative requirements must be *flexibly specified* – so as not to hinder their incremental re-definition – and the corresponding functionalities be *flexibly combined* – so as to fit an ever-changing and necessarily underspecified context.

This stress on flexibility has, on the one hand, motivated us in defining LWOAD as a language by which to render coordinative requirements in a computable but yet platform-independent and abstract manner; on the other hand, we were motivated in using it to express an upper layer of application logic that would be conceptually "on top of" a full-fledged electronic document system and that would endow that system with cooperation-oriented functionalities (see this general schema in Figure 4).

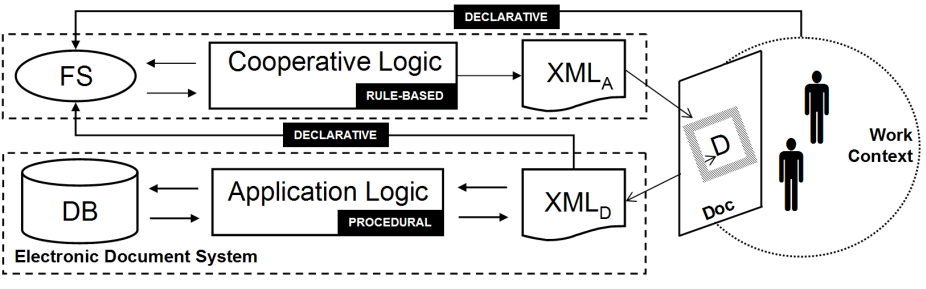

Fig. 4. The two-tier architecture designed to enhance electronic documents with collaboration awareness. `FS` stands for Fact Space, the memory where declarative representations of documental and working context are stored.

The adopted *declarative* and *rule-based* approach guaranteed that *coordinative functionalities* can be expressed in terms of *reactive and declarative mechanisms* [22]; these are symbolic statements intended to translate the typical question of users " ... and could I have the system do this, whenever that occurs?" into computable instructions. The declarativeness of these statements allows for the expression and formal specification of *what a system should do* rather than worrying about *how it really accomplishes it* at specification time. Declarativeness also allows mechanisms to be written without imposing a strict *control flow*,

which is hardly recognizable in actual work situations. On the other hand, reactivity allows mechanisms to be written by using circumscribed units of code (i.e., rules). This"convenience of definition" relates to flexibility in terms of a greater easiness of maintenance due to better modularity and incrementality; moreover, defining mechanisms in a higher-level way than by means of traditional procedural specification is also intended towards a better participation of users in the process of modelling and defining formal expressions, so that these could reflect how users really see their domain-specific knowledge and functionalities.

The rule-based layer of cooperation logic on top of the procedural application logic of a traditional electronic document system is sensitive to both the content of documents (in Figure 4 denoted with XML_D, i.e., data rendered in XML format) and the symbolic representation of context. The output of this context-aware layer is the conveyance of *additional* information, namely API, that does not change documental data but rather how the interface of a document system displays and "affords" them (in Figure 4 denoted with XML_A, i.e., API rendered in terms of XML metadata). In doing so, data conveyed in documents (denoted with a capital `D` in Figure 4) would be gathered from official repositories (e.g., a hospital DB) according to procedural organizational logic; conversely, the API attached to these data (denoted as an highlighted border all around the capital `D` in Figure 4) would be provided according to more flexible mechanisms on the basis of coordinative conventions.

Furthermore, the rule-based approach addresses the flexibility requirement from the combination point of view. In rule-based programming, a rule is executed automatically on the basis of any significant event and data change only after that: i) its "applicability criteria" have been matched by the rule engine against current data, i.e. what constitutes the symbolic description of a situation at run-time; and ii) after that it has been selected among all other rules as the most suitable to that situation, according to some strategy (e.g., specificity, recentness). We agree with [22] that rule-based programming have some important advantages over procedural programming in grasping and aligning with cooperative work, especially for its data- and event-driven nature. In addition, the particular kind of action that LWOAD mechanisms trigger, i.e., augmenting the interface with graphical cues and indications promoting collaboration awareness, brings down the problem of mutual consistency of the rule set. This problem often makes the adoption of this form of declarative specification difficult to be understood and managed by layman users. Our case is different from production systems and expert system where possibly long chains of rules are consecutively executed to infer a line of action on the basis of progressively true conditions. Conversely, we adopt a rule-based approach in order to separate functional concerns into single mechanisms (not into chains of their executions); and, for the mechanism design, we advocate the principle that the consequent of each mechanism should be expressed as simply as possible, i.e., that each mechanism should only address a single and punctual functionality that the system must exhibit against possibly over-detailed and specific contextual conditions (which are specified in the mechanism's antecedent). Moreover, the fact that LWOAD

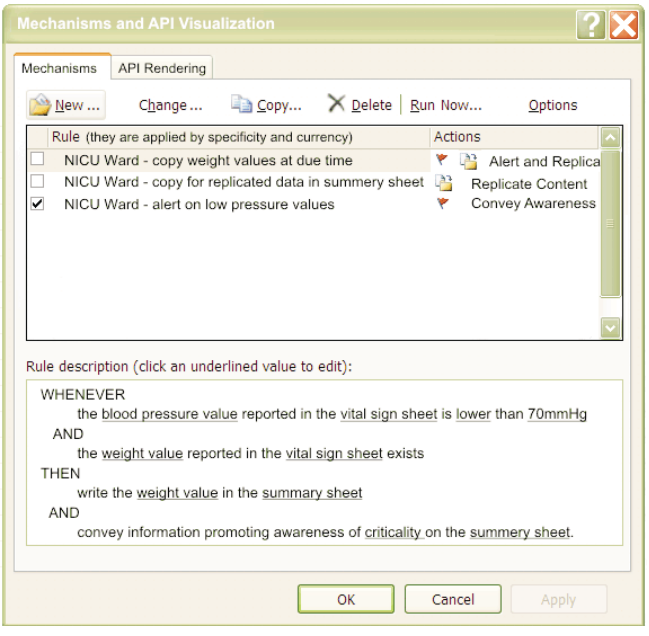

Fig. 5. Screenshot of the mockup for the mechanism editor, first windows

consequents do not change data (and hence the state of the world) but rather convey APIs, and that APIs are conceived as orthogonal guarantees that data inconsistency can not occur for their execution. Moreover, possible conflicts in alerts (e.g., when two mechanisms trigger the same API but with different values) can be "caught" before execution by the mechanism interpreter itself (i.e., by monitoring the execution agenda). In this latter case, the system can propose the conflict to users as particular situations that call for their interpretation and resolution on the basis of their experience and knowledge.

6 Simulation and Mockup Tests of LWOAD Specification

In what follows, we illustrate the mockup that we designed after the requirement analysis. This was meant as a proof-of-concept for the prospective application that users would use to develop coordinative mechanisms by themselves. Since mechanisms are but rules, the main idea was to assimilate mechanisms development to rule configuration: we then conceived the LWOAD mechanism editor similar to an interactive help utility, much alike those provided by email clients to guide users through the configuration of personal filters and mechanisms of message filing. The mockup was realized in MS PowerPoint and intended as a sequence of dialog boxes where users could select options and fill in details; each slide was endowed with active areas corresponding to the buttons and links of the prospective interface in order to simulate the typical interaction involved in mechanism creation.

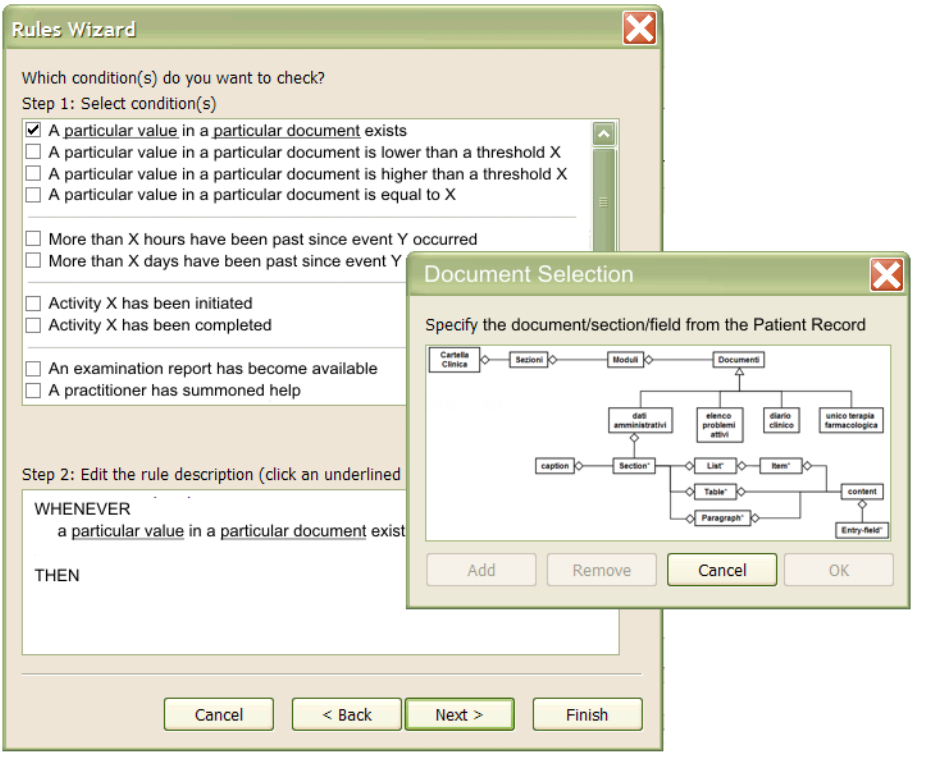

Fig. 6. Screenshot of the mockup for the definition of the mechanism's antecedent

In the first window, users have access to the macro-functionalities of the editor (see Figure 5) as regards either mechanism composition or API visualization. In this paper, we do not address the functionalities of API rendering, i.e., the association between API types and rendering functionalities (like, e.g., colors, icons, highlighting) provided by the documental platform. In regards to mechanism composition, a list of existing mechanisms is displayed in the top frame of the window. Users can read the textual description of each mechanism by selecting the corresponding row: the description is then displayed in the bottom frame (in Figure 5 we report the same mechanism illustrated in Figure 2). From the textual description of a mechanism, users can directly modify its parameters by clicking on the underlined elements (i.e., variables of the mechanism's pattern). Users can also change the structure of the mechanism (clicking on '`change...`'); delete it, "activate" it (by checking the corresponding checkbox); and run the mechanism to check its functioning (clicking on '`Run Now...`'). If the user clicks on '`New`', the mechanism wizard starts a three-step process; in the first window, the system proposes two options: to create a mechanism from a template, or to compose it from scratch, i.e., from a blank template. We will consider this second case. In this case, the system opens a new window in place of the former, like that depicted in Figure 6 (left side, background). From the top frame of this

window, the user can select any number of conditions the mechanism should be sensitive to (in its antecedent). In-depth analysis and participatory design sessions have allowed to list together all the relevant conditions that practitioners wanted to be catched with respect to the records' content, time and the clinical context. By selecting a condition from the list, the associated conditional statement is added in the bottom frame. As in the case of the first screen (Figure 5), the user can specify the value of the parameters the mechanism should monitor by clicking on the underlined parts of the statement. In doing so, corresponding input boxes are displayed to allow users insert the value (e.g., 70 mmHg, a blood pressure value as in the case reported in Figure 2). If the user wants to specify the document where to check the condition, the system opens a box like that depicted in Figure 6 (right side, foreground). Here, the user can consult a tree-like schema of the official documentation and select the document/s (or their inner sections) whose data must be matched with the pattern's values. Once the antecedent of the mechanism has been defined, the wizard proposes a third window (in place of the previous one – see Figure 7) where the user can specify what the system is supposed to do when the conditions are true, i.e., the elements of the consequent part. Also in this case, the user can select a number of different actions from the top frame; and then specify a value for each key presented in the textual description in the bottom frame. The list depicted in

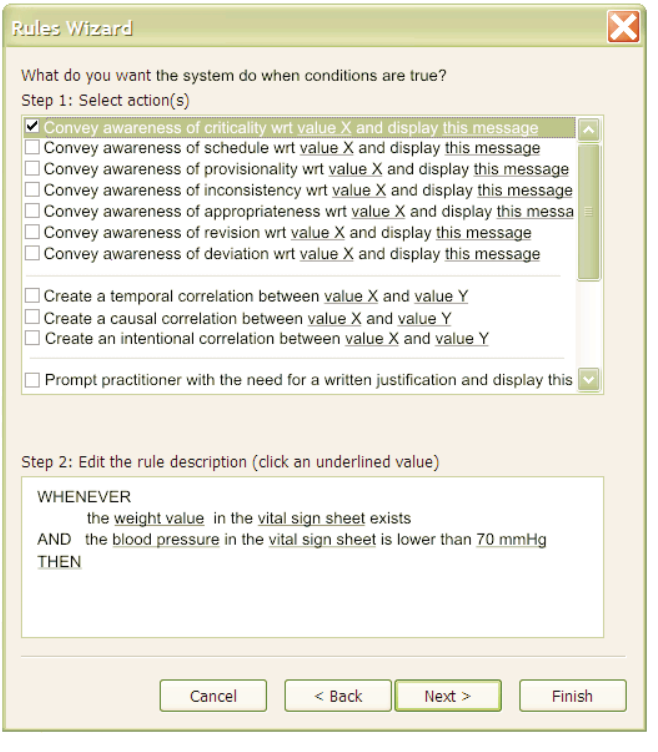

Fig. 7. Screenshot of the mockup for the mechanism's consequent definition

Figure 7 presents the main options selected by practitioners on a relevance basis during the interviews. The system groups these options together by similar category: e.g., API provision, connection definition, data replication and insertion and the like. In the case the action regards API provision, the user can also insert a textual explanation. This would be displayed on the clinical record only if requested in case a user can not interpret the API clue conveyed. By clicking the button '`Finish`', the system creates the mechanism that is executable by the LWOAD interpreter. This is currently a compiler that renders LWOAD constructs into corresponding structures of `Jess` [1], which are then executed by the fast and reliable rule engine provided within this scripting environment. Jess was chosen after our positive experience during the development of a distributed version of Jess for the construction of applications in the Collaborative Ubiquitous Computing and Ambient Intelligence domains [23,24]. The strong decoupling we pursued in the design of the WOAD architecture between cooperative logic and the operational platform allows for the development of other compilers by which to translate LWOAD statements into other rule-based scripting languages. Currently, the implementation of a WOAD compiler compliant with the `JBoss Rules`[2] engine is under consideration to overcome the limits of Jess in dealing with data structures more complex than lists.

7 Conclusions and Future Work

The paper illustrates the trajectory we have followed to approach the definition of a framework where layman users can specify mechanisms supporting their cooperation mediated by documental artifacts. This trajectory is not completed but currently covers the most important part of the research path: namely, i) understanding the kind of functionalities users need; ii) identifying a way to express the functionalities and iii) defining an architecture where the functionalities can then be implemented and validated. Also in regards to how users should interact with this architecture and system, our research agenda covers the incremental involvement and increasing skills of users: namely, we started having practitioners express conditional mechanisms in natural language; then we stimulated them to use an application to compose and detail condition-action statements of increasing complexity; lastly, we envision the opportunity to have users tweak and adjust LWOAD statements created with the former application in case of progressive customization and compliance to local needs. In this study, the choice of a rule-based representation seemed the most suitable one for the different types of flexibility it allows: namely, flexibility in specifying computational mechanisms and in combining them together at execution time.

The empirical findings we gathered so far refer to the healthcare domain and to a hospital setting. In this case, the problem was to endow the implementation of an EPR with means that preserve or even support the conventions

[1] Java Expert System Shell:`http://www.jessrules.com/`

[2] See `http://www.jboss.com/products/rules`

that practitioners adopt to make their cooperation smooth and seamless. A first natural question regards how much our empirical findings can be generalized to other settings, even within the same domain: we acknowledge to have found a group of doctors and nurses that were extraordinarily helpful to try and co-develop innovative solutions with computer researchers and professionals; they were extremely motivated in molding any tool that could help them in providing a better care and re-delivering more healthy newborns to their parents. For this reason, scalability and generalization of our proposal is part of our research agenda. Our next activities will also include the full implementation of the interface, informally validated through the mockup illustrated in Section 6: our pilot sessions confirm its feasibility as a tool that makes users autonomous in specifying condition-action mechanisms, once the set of patterns has been identified for their antecedents and a rich palette of graphical cues has been proposed as output of their consequents. This approach however opens a new area of problems: a tool like that depicted in Section 6 interprets EUD more as a flexible kind of customization than as a real development environment [25]. In fact, the predefined set of patterns cannot fulfill the needs of increasingly skilled users wanting to extend the "localization" of the desired support. To fulfill this requirement, users must have access to the implementation environment also: here it is where the WOAD framework, and specifically its specification language – LWOAD – can play a relevant role for its declarative, abstract and modular approach that divides complex situations into a bunch of supportive functionalities that are called reactively with respect to the current context. The first phase of the study showed how (relatively) easily layman users can transform informal rules of their particular setting into executable statements, due to the "isomorphic" nature of the involved representations.

The next step is to allow users define more general rules by selecting the needed pieces of information to build the antecedent of the rules out of the documental artifacts in a natural way. Practitioners proposed a solution that could mimic how users of a spreadsheet copy data from cell to cell just by clicking on them and pasting them where needed. Likewise, users should be able to express contextual conditions from a predefined palette of templates (concerning, e.g., time, frequency, iterations, etc.) and specialize them by expressing simple key-value pairs and selecting data structures and data values directly from their documentation. Of course, this additional flexibility would ask for a strong interoperability between the coordinative layer and the archival layer, i.e., the EPR, or at least the capability to export and represent suitable views of clinical data, irrespectively of how they are organized and memorized. In our opinion, and on the basis of our interaction with practitioners, this kind of interoperability could bring data presentation strategies to EPR that are more natural and closer to the way practitioners use the current paper-based clinical record fruitfully. This positive mutual influence is the final goal we aim to pursue in our planned interactions with users in the healthcare domain.

Acknowledgements

The work presented in this paper has been partially supported by the F.A.R. 2008. The authors would like to thank the management and the Neonatal Intensive Care Unit personnel of the Alessandro Manzoni Hospital of Lecco for their kind collaboration. In particular, we would like to acknowledge the invaluable help and courtesy of Dr Bellù and Mrs Colombo.

References

1. Braa, K., Sandahl, T.: Introducing digital documents in work practices - challenges and perspectives. Group Decision and Negotiation 9(3), 189–203 (2000)
2. Sellen, A.J., Harper, R.H.R.: The Myth of the Paperless Office. MIT Press, Cambridge (2003)
3. Terzis, S., Nixon, P., Wade, V., Dobson, S., Fuller, J.: The future of enterprise groupware applications. Enterprise Information Systems, 99–106 (2000)
4. Xiao, Y.: Artifacts and collaborative work in healthcare: methodological, theoretical, and technological implications of the tangible. J. of Biomedical Informatics 38(1), 26–33 (2005)
5. Hertzum, M.: Six roles of documents in professionals' work. In: ECSCW 1999: Proceedings of the Sixth European conference on Computer supported cooperative work, pp. 41–60. Kluwer Academic Publishers, Norwell (1999)
6. Garfinkel, H.: "Good" organizational reasons for "bad" clinic records. In: Studies in Ethnomethodology, pp. 186–207. Prentice-Hall, New Jersey (1967)
7. Berg, M.: Accumulating and Coordinating: Occasions for Information Technologies in Medical Work. Computer Supported Cooperative Work, The Journal of Collaborative Computing 8(4), 373–401 (1999)
8. Fitzpatrick, G.: Integrated care and the working record. Health Informatics Journal 10(4), 291–302 (2004)
9. Winthereik, B.R., Vikkelso, S.: Ict and integrated care: Some dilemmas of standardising inter-organisational communication. Computer Supported Cooperative Work, The Journal of Collaborative Computing 14(1), 43–67 (2005)
10. Cabitza, F., Simone, C.: "You Taste Its Quality": Making sense of quality standards on situated artifacts. In: MCIS 2006: Proceedings of the First Mediterranean Conference on Information Systems, Venice, Italy, AIS (2006)
11. Berg, M., Goorman, E.: The contextual nature of medical information. International Journal of Medical Informatics 56, 51–60 (1999)
12. Schmidt, K., Simone, C.: Coordination mechanisms: Towards a conceptual foundation of CSCW systems design. Computer Supported Cooperative Work 5(2/3), 155–200 (1996)
13. Divitini, M., Simone, C.: Supporting different dimensions of adaptability in workflow modeling. Computer Supported Cooperative Work 9(3), 365–397 (2000)
14. Heath, C., Luff, P.: Documents and Professional Practice: 'bad' organisational reasons for 'good' clinical records. In: CSCW 1996: Proceedings of the international conference on computer-supported cooperative work, pp. 354–363. ACM Press, Cambridge (1996)
15. Cabitza, F., Simone, C.: "... and do it the usual way": fostering awareness of work conventions in document-mediated collaboration. In: ECSCW 2007: Proceedings of the Tenth European Conference on Computer Supported Cooperative Work (ECSCW), Limerick, Ireland, September 24–28, pp. 119–138. Springer, Heidelberg (2007)

16. Dourish, P., Bellotti, V.: Awareness and coordination in shared workspaces. In: CSCW 1992: Proceedings of the 1992 ACM conference on Computer-supported cooperative work, pp. 107–114. ACM Press, New York (1992)
17. Cabitza, F., Sarini, M., Simone, C.: Providing awareness through situated process maps: the hospital care case. In: GROUP 2007: Proceedings of the 2007 International ACM SIGGROUP Conference on Supporting Group Work, pp. 41–50. ACM Press, New York (2007)
18. Cabitza, F., Simone, C.: Supporting practices of positive redundancy for seamless care. In: CBMS 2008: Proceedings of the 21st IEEE International Symposium on Computer-Based Medical Systems, Jyväskylä, Finland, June 17-19, 2008, pp. 470–476. IEEE Computer Society, Los Alamitos (2008)
19. Hardstone, G., Hartswood, M., Procter, R., Slack, R., Voss, A., Rees, G.: Supporting informality: team working and integrated care records. In: CSCW 2004: Proceedings of the 2004 ACM conference on Computer supported cooperative work, pp. 142–151. ACM Press, New York (2004)
20. Randell, R.: Accountability in an alarming environment. In: CSCW 2004: Proceedings of the 2004 ACM conference on Computer supported cooperative work, pp. 125–131. ACM Press, New York (2004)
21. Dourish, P.: Seeking a Foundation for Context-Aware Computing. Special Issue on Context-Aware Computing HCI Journal 16 (2001)
22. Wulf, V., Stiemerling, O., Pfeifer, A.: Tailoring groupware for different scopes of validity. Behaviour and Information Technology 18(3), 199–212 (1999)
23. Cabitza, F., Seno, B.D., Sarini, M.: DJess – a context-sharing middleware to deploy distributed inference systems in pervasive computing domains. In: ICPS 2005: Proceedings of the IEEE International Conference on Pervasive Services, Santorini, Greece, pp. 229–238 (2005)
24. Cabitza, F., Locatelli, M., Sarini, M., Simone, C.: CASMAS: Supporting collaboration in pervasive environments. In: PerCom 2006: Proceedings of the Fourth Annual IEEE International Conference on Pervasive Computing and Communications, Pisa, Italy, pp. 286–295. IEEE, Los Alamitos (2006)
25. Liebermann, H., Wulf, V., Paternò, F. (eds.): End-User Development. Kluwer Academic Publishers, Dordrecht (2006)