HAL open science

Quality in Ubiquitous Information System Design

Sophie Dupuy-Chessa

▶ To cite this version:

Sophie Dupuy-Chessa. Quality in Ubiquitous Information System Design. 3rd Int. Conf. on Research Challenge in Information Science (RCIS'2009), 2009, Fez, Morocco. pp.343 - 352. hal-00953291

HAL Id: hal-00953291

https://inria.hal.science/hal-00953291

Submitted on 28 Feb 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Quality in Ubiquitous Information System Design

Sophie Dupuy-Chessa
LIG Laboratory, University of Grenoble
385 rue de la Bibliothèque
38041 Grenoble Cedex 9, FRANCE
Sophie.Dupuy@imag.fr

Abstract— **Information systems become ubiquitous. This opens a large spectrum of the possibilities for the end-users, but the design complexity is increasing. Therefore insuring quality during design is more than ever a challenge. In this article, we study this challenge by identifying the specificities of ubiquitous computing design and by considering the influence of these specificities on the quality of the various aspects of information system design (models, languages, processes and tools). For each aspect, we discuss its requirements on quality and present related works valuable for the definition and the evaluation of ubiquitous information system design quality.**

Quality;design;models;process;languages;tools

I. INTRODUCTION

Nowadays, technological progresses such as the microprocessors and sensors miniaturization, and the communication technologies explosion, allow end-users to access information everywhere at any time and in a personalized manner. In other words, information becomes instantaneous, universal and ubiquitous. This opens a large spectrum of the possibilities for the end-users: they can see their bus timetable on their mobile phone; they can see on special devices contextualized information while visiting a museum; etc. For instance, simply considering the new human-computer interaction possibilities may induce business evolution [1]. Therefore the design complexity is increased by adding parameters like the devices, the location, the user characteristics... As mentioned by [2], the difficulty to design systems has been moved up. So it becomes more important but also more difficult to achieve quality in the design of information systems: insuring quality in design is more than ever a challenge.

However it is trivial to define what is quality in design. If we consider the quality definition given by the ISO standards, quality is "the totality of features and characteristics if a product or service that bear on its ability to satisfy stated or implied needs". Therefore we need to understand requirements for quality in design before expressing its features or characteristics. This is the goal of this article: identifying requirements and some characteristics for achieving quality in the design of ubiquitous information systems so as to better understand the underlying challenge.

We consider that information systems design is driven by the method used to develop the system. A method is structured by four inseparable and complementary components [3]: models representing facets of the system, languages (by which models are constructed), process(es) guiding the development activities, and tools for allowing and facilitating the use and setup of processes, models and languages. This article studies for each component of a method (models, languages, processes and tools) what are quality requirements and characteristics considering new needs introduced by ubiquitous information systems.

The remainder of this paper follows the structure of our study. Section 2 presents the specificities of ubiquitous information system design while the other sections describe how these specificities can be taken into the method components. Each section presents the quality aspects for one component: it discusses the requirements concerning quality and presents related works valuable for the definition and the evaluation of the method component quality. Section 3 studies quality for models; section 4 present the quality aspects of languages; section 5 discusses processes and section 6 presents tools supporting a quality approach. Finally we conclude by summarizing our point of view on quality in ubiquitous information system design.

II. SPECIFICITIES OF UBIQUITOUS COMPUTING DESIGN

A. A large variety of models

Models are widely used in information system development. They are used to communicate with stakeholders, to analyse the problem, to document the system, to generate code etc. Generally software engineers use UML models. But considering ubiquitous information systems, many other models are used to take into account specific concerns such as the interactive devices [5] or the context of use [6].

Their corresponding languages are still research work. There is no consensus on the concepts that they must include nor on their notations. For instance, there are at least three different languages in the human-computer interaction domain, in order to represent the interaction modalities (vocal, gestural, etc) and their supporting devices in the case of systems mixing the virtual and the real worlds. So there is a clear need to better understand ubiquitous computing concepts so as to propose relevant modelling languages.

B. A large variety of designers and stakeholders

Many different people are involved in the design. Mainly, there are designers who realize models and there are stakeholders, sponsors or system end-users who read them.

With ubiquitous information systems, stakeholders and end-users are more involved in the design as the context of use (organisational, technical etc) is becoming more complex: they may be used to different interactive devices (mobile phone, PDA, computer etc); they may have different uses; they may have different requirements, which are sometimes too futuristic for the actual possibilities of ubiquitous computing. These various profiles must be taken into account while designing a new ubiquitous information system.

Designers also have different profiles. They may be specialists in ergonomics, human-computer interaction, security… They have neither the same education, nor the same experience in modelling. So they do not model in the same manner.

So it is not any more possible not to consider the variety of people while proposing a model.

C. Immature specific processses

The introduction of new aspects (context, user characteristics…) has led to the apparition of new models, but also sometimes to new processes for identifying and specifying these new aspects. For instance, [28] proposes a process for choosing the human-computer interaction of collaborative systems that merge information from the virtual and the real worlds. This kind of proposal is a step forward managing the design of ubiquitous information systems even if the proposed processes are still immature and need to be tested in larger applications.

Many aspects remain to be supported by appropriate processes. Most of the time, designers have no methodological help for using or realizing ubiquity-oriented models.

Anyway specific processes must be considered while designing an ubiquitous information system. But for the moment, they are not integrated into classical software development methods. It may be a long and overly complex task to incorporate them. Moreover each domain (human-computer interaction, software engineering etc) has its specialists with their specific practices. Proposing a global process for ubiquitous information systems may not correspond to every specialist's uses. So instead of building global rigid processes, it seems more appropriate to offer a way to build processes from existing ones [29], with the following consequences that 1) every domain specialist can keep his practices; 2) new processes can be easily added when they are identified.

III. QUALITY IN MODELS

A. Considering the large variety of models users

Involving stakeholders or end-users in the design is a necessity, but it is also a challenge. They are generally unused to models. So limiting the number of concepts used to model may be appropriate for them. For instance, the concepts used in a class diagram realized for stakeholders may only contain classes and associations for explaining them their domain. For developers, designers may use composition, inheritance to model the computerized solution specification. So models must be adapted to their audience.

Even designers do not use a notation in the same manner. Purchase et al. [9] have performed an experiment to compare few UML notations. It appears that experts and novices do not prefer the same graphical symbol to represent the same concept (Fig. 1).

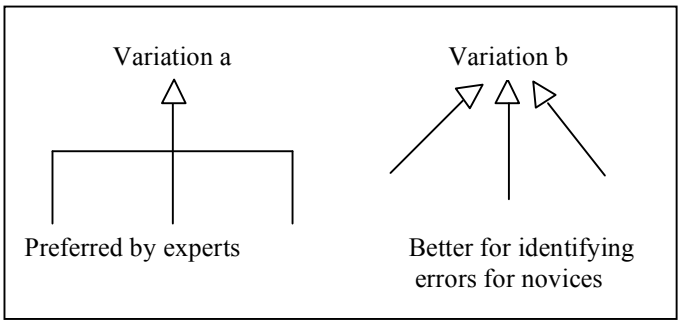

Figure 1. Variation in symbol choices

Finally, we can consider that all the people involved in design do not have the same cognitive abilities. It has already been shown by [4] that human cognition is related to modelling and that cognitive techniques can be used in conceptual modelling. For instance, achieving the correspondence between the model and the audience's interpretation of it, is a matter of interpreting a meaning with the lowest possible cognitive effort. With the increasing variety of people involved in modelling, cognition is more than ever required to adapt models to their audience's cognitive abilities.

To summarize, the models effectiveness is influenced by several factors among those: the analysts' own modelling experience and her cognitive abilities and interpreters' experience with conceptual models [11]. Therefore model quality depends on the users' point of view (designers, end-users, developers …).

B. Improving the quality of new models

The quality of models proposed for ubiquitous information systems is very important to achieve a good design. Generally, the quality of models concerns the correctness of the syntax and the semantics of models.

The syntactic quality relates the links of a model to its language constructs without considering the meaning. The syntax can be checked inside one model or between models. For example in UML, a class must be defined only once in a class diagram. If we also consider domain specific models, some of the consistency rules have been specified: for instance, in an ASUR model [5] that designs an interactive solution, a computerized object, which correspond to classes in UML, must also be specified only once.

The semantic quality indicates the link of a model to its domain or to the knowledge of the domain specialists. The complexity of ubiquitous information systems requires

simulation, proof or model-checking to achieve semantic quality. But the complexity of these techniques often limits them to specialists. At least, it is necessary to check the consistency with other models. For instance, the interactive techniques choices specified in an ASUR model must be relevant for a user task.

C. *Identifying generic charcteristics of models*

The model-based approach is more and more used in different domains so that limiting the study of quality to UML models [7] is no longer sufficient. Therefore quality in design requires defining characteristics for many different models, when possible characteristics must be generic enough to be applied to any kind of models.

In particular, it is time to consider the cognitive abilities of model audience so as to define ergonomics of models. Ergonomics is the scientific study of the relationship between human and its means, its practices and its working environments. In the human-computer interaction domain, requirements are often expressed in terms of usability. Usability for an interactive system characterizes the capacity of the system to achieve its goals (achieve a correct result with a given quality) with efficiency, comfort and security. Usability is evaluated by ergonomic criteria. For instance, in [8], the working charge criterion contains a property about the informational density, which specifies the importance of limiting the number of readable information at a given moment. If we consider design, such properties can be applied so as to obtain more readable models. Model usability can be defined as its capacity to allow users (designers, stakeholders etc) to achieve her goals with efficiency, comfort and security. The objective is to allow designers to produce usable models, independently from their underlying language, from a human user point of view.

D. *Defining quality of models*

There is little agreement among experts to define what makes a "good" model. If we consider the definition of quality given by ISO 9000 (cited in the introduction), [22] proposes to define the model quality as "the totality of features and characteristics of conceptual model that bear on its ability to satisfy stated and implied needs".

In the ISO standard for software product quality, software quality is decomposed into six quality characteristics, which are further divided into twenty-four quality sub-characteristics, which are measured by one hundred and thirteen metrics.

Considering models as a particular type of software product, the ISO structure must be applied to model quality [22]. Model quality must be defined by a set of characteristics such as consistency, precision or aesthetics, which answer to specific purposes. Of course there are relations between characteristics. For example, in [10], the first level of the model quality is the primary use of models, either development or maintenance. The primary uses are decomposed into purposes: for the development primary use, the purposes are communication, analysis, prediction, implementation and code generation. For each purpose, the required characteristics are specified. For instance, communication requires evaluating complexity, self-descriptiveness, conciseness and aesthetics. Complexity for instance is defined as the effort required for understanding a model. After selecting some quality characteristics, a set of metrics is identified to measure characteristics. For complexity, some metrics are the depth in inheritance tree or the number of classes per Use Case (Fig. 2).

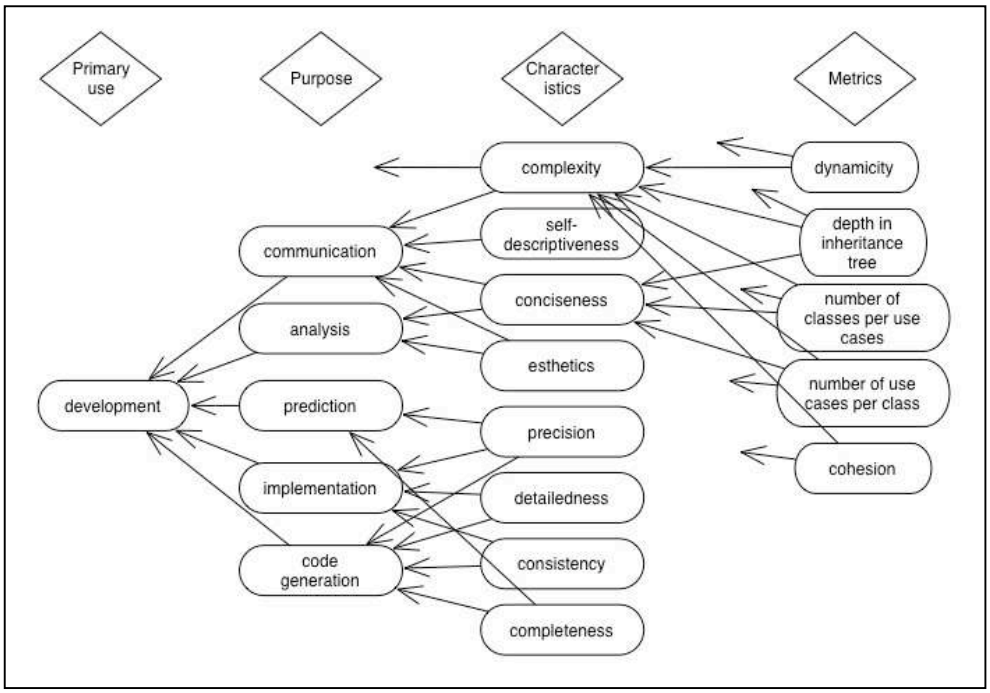

Figure 2. Lange and Chaudron's Quality Model

In the previously presented approaches, quality is viewed through a tree-structure. The root concept is the global quality, which is divided info different aspects of the quality. These aspects can themselves be refined into more specific aspects. The problem is that relations are often based on judgment. For instance, ISO and IEEE have different hierarchies of quality attributes for software quality.

Another approach to structure model quality is based on the study of signs processes. The corresponding frameworks are then called semiotic. They have been initiated by Lindland [12] to evaluate conceptual models in general. In [12], quality is detailed into syntactic, semantic and pragmatic aspects. The syntactic quality verifies how well a model corresponds to its language constructs without considering the meaning. Its goal is to achieve syntactic correctness. The semantic quality indicates the link of a model to its domain or to the knowledge of the domain specialists. There are two semantic goals: validity and completeness. Validity means that all statements of the model are correct and relevant to the problem. Completeness means that the model contains all the correct and relevant statements of the domain. Finally the pragmatic quality relates to the interpretation by the model audience. Its goal is comprehension.

This framework is more than a theoretical proposal: it has been validated by an empirical study [13] and it has given rise to numerous complementary works. For instance, [12] has been extended by Krogstie et al. in order to detail the pragmatic quality [14][27] (Fig. 3). The pragmatic quality depends on the model users. So there is social pragmatic quality for stakeholders, physical quality for modellers and technical pragmatic quality for technical actors (modelling tools). There is also the organizational quality for measuring if the model fulfils the modelling goals, and the empirical quality, which comprises comprehensibility matters like readability or lay-out.

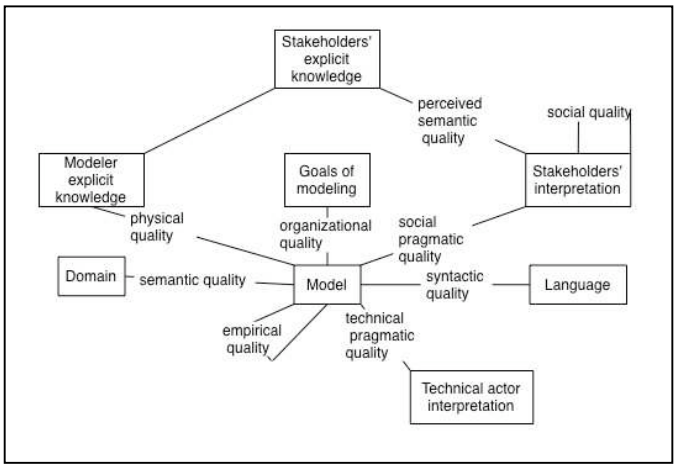

Figure 3. Krogstie et al's quality framework – model view

The semiotic frameworks have the interest of representing the modelling context and linking the quality with it. In the case of ubiquitous information systems where many models and the variety of the audience must be taken into account, the semiotic frameworks are particularly interesting: they consider any kind of models, they do not limit themselves to the technical quality of models but also consider the model audience. By considering characteristics such as comprehensibility, they provide a way of integrating ergonomics properties, which seem particularly important to us so as to improve the quality of any kind of models.

However they often stay at a theoretical level without proposing easily usable means to achieve quality. Identifying quality characteristics must be coupled with means to measure quality and to improve it.

E. Improving and Evaluating model quality

The evaluation of models can be specific to a problem such as the cognitive complexity. So problems are studied through user experiments or studies. The goal is to study the manner in which designers are developing models. This permits to propose preventive techniques such as modelling conventions, training, tools or adapted design processes.

The study of modelling defects is particularly interesting. [17] studies 16 industrial UML models to explore the level of defect occurrences in industrial UML modelling and to identify factors that influence the quality of models. [18] classifies the kind of modelling errors in order to help novice analysts in developing quality conceptual models. The identification of patterns of errors is a good way to identify the root causes of modelling defects and to change processes so as to prevent them from occurring in the future.

Another approach consists in realizing experiments in order to validate hypotheses on model quality. For instance, [19] realized experiments in an academic context to validate 1) the dependence between the structural complexity and size of UML class diagrams on the one hand and their cognitive complexity on the other hand; 2) the dependence between the cognitive complexity of UML class diagrams and their comprehensibility and modifiability. Empirical evaluation helps in understanding the cognitive aspects of modelling and in specifying or measuring model quality.

The most used approach to evaluate the quality of models is metrics. As for code quality, metrics provide a numerical evaluation of characteristics. For instance, in [10], after selecting quality characteristics, a set of metrics is identified. Some of them are traditional object-oriented metrics such as coupling (the dependence between classes) or cohesion (how closely the local methods are related to the local instance variables in a class), while a few of them are model-specific such as the number of crossing lines in a diagram. The metrics-based approach provides an abstract, fast but global evaluation of a model.

The problem-specific and the metrics-based approaches are complementary. The first one provides specific solutions such as the identification of typical errors while the second gives a global view of model quality. As mentioned by [20], both approaches are valuable and can be coupled to propose more easily applicable frameworks. By integrating the contributions of each approach, the authors expect 1) to augment the global approaches with metrics based on identified typical errors; 2) to associate to each typical modelling errors, correction transformations that can be suggested or automatically applied.

So the quality measure given through metrics is intended to be more relevant while some specific errors can be detected and corrected.

Evaluation is mainly considered for UML or Entity-Relationship models. For these models, it is the metrics-based approach, which is the most developed. The experimental approach is too often left aside. If it proposes a limited view of model quality, it can nevertheless give rise to usable solutions. However, evaluating the quality of models remains a challenge. As reported by J. Nelson, "evaluating the quality of conceptual models is still very much 'art' than 'science'" [21]: the definition and the goals of model quality must be clarified; quality characteristics must be better structured so as to understand their links; metrics-based evaluations must be developed.

For domain specific languages, everything remains to be undertaken. Their models do not give rise to evaluations. In particular, finding typical errors or realizing user experiments are good ways in order to understand their limits. The lack of experiments is to be regretted because it is a way of making a language more mature by adapting modelling conventions, by improving training, by developing tool support or by proposing adapted modelling processes. However even before that, many works need to have their languages properly defined.

IV. Quality in Modelling languages

As models are instances of languages, their quality partly depends on the quality of their modelling languages. This section identifies the requirements for modelling language quality in ubiquitous information systems and defines modelling language quality.

A. *Studying the adequacy of languages*

While new languages multiply rapidly to describe ubiquitous information systems, we may wonder whether they represent the relevant concepts. A language has semantics that express the meaning of its constructs. The concepts it features must be relevant for the domain. For instance, the specification of human-computer interaction in ubiquitous systems requires identifying the physical objects involved in the task. They can be either a tool used to realize the task such as the physical wand or a physical entity constituting the object of the task i.e., the object that the task acts on.

The relevant concepts must be expressed by the constructs of the language or the relationships between them. It is the syntax of the language. For example, ASUR [5] has two kinds of elements: components that figure out physical or digital entities, such as a pointer, a wand or a 3D volume, and relationships that connect these components. The syntax must be precisely defined in order to build models that respect the "grammar" of the language (abstract syntax).

But a language must also give the appropriate representation of its constructs (concrete syntax) so that the language users can easily learn its use. For instance, using a "stick figure" picture is a good way to represent the user construct: designers or stakeholders can easily understand what the representation means.

Moreover many languages are actually proposed for the various aspects of ubiquitous information systems. If we consider the human-computer interaction aspects, several notations have been proposed with the same goal: representing the way of interacting with physical everyday objects. Then it is a necessity to be able to compare them in order to choose the most adequate one for the considering system. The comparison will be based on the characteristics (abstract, concrete syntax and semantics) of each language.

B. *Checking the consistency*

At the level of models, we have noticed that the various models used to design ubiquitous information systems must be consistent (e.g. consistency between an ASUR and a task model). This consistency is based on rules that are not specific to models, but that express the semantic links between two languages. For instance, every ASUR model must be related to a task in a task trees.

So we need an open approach where different types of languages can be defined and rules between them specified. This approach is the one adopted in the model-driven engineering, MDE [23], where the first class concept is the model. In MDE, everything can be considered as a model. MDE is well-adapted to ubiquitous information systems design where many domain-specific languages are used: 1) the syntax and the semantics are specified by their meta-models, which define constructs and rules to build models; this is the basis to talk about language quality; 2) meta-models can be used to realize operations on models, in particular consistency rules between languages can be defined.

Actually only few [24] of the new languages for ubiquitous computing design adopt the MDE approach so that it is uneasy to clearly understand their concepts and the way to use them. So it is difficult to evaluate their utility or to compare them. However the MDE concepts seems like a definitive asset for achieving quality in modelling languages.

C. *Defining the quality of languages*

Previous, we identify needs for a clearly specified and appropriate syntax and for a precise semantics of languages. Therefore as we mentioned previously, it is very important for quality of modelling language to adopt an MDE approach.

However the MDE approach is not sufficient to guarantee the quality of languages. We must not forget the language users (modellers and future stakeholders). Krogstie [27] identifies five characteristics of modelling languages (Fig. 4):

- Domain appropriateness: the concepts of the languages are powerful enough to express anything in the domain, but no more. This is related to the quality of the language semantics.
- Participant language knowledge appropriateness: the language must be appropriate for the participants (modellers). It is best to base a language on experience with previously used languages.

- Knowledge externalizability appropriateness: There should be no statement on the explicit knowledge of the participants that cannot be expressed in the language. This is also related to the quality of the language semantics.
- Comprehensibility appropriateness: the language must be accessible for stakeholders. The language should not have too many concepts; the concepts must be distinguishable from each other etc.
- Technical actor interpretation appropriateness: the syntax and the semantics of the language must be formal enough so that technical actors i.e. modelling tools can automate some treatments on models.

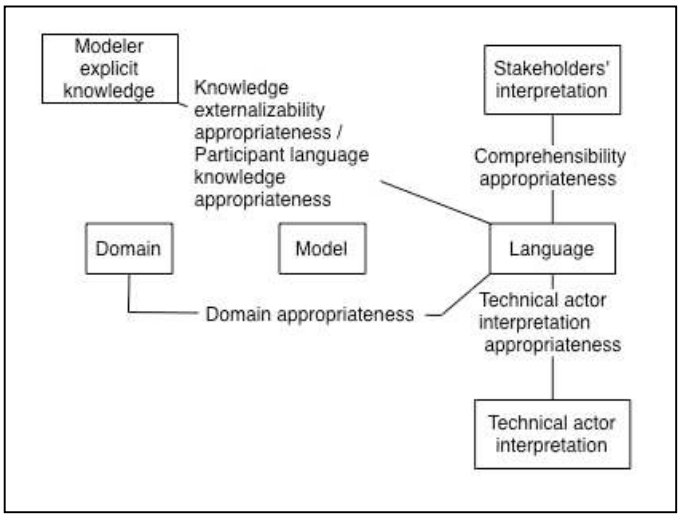

Figure 4. The Krogstie et al's language quality framework

This semiotic framework is very interesting for new ubiquitous languages. It shows to the language creators that they should not only consider the syntax and the semantics of their proposal, but also the way it can be manipulated by its future users.

D. *Measuring the quality of languages*

Proposing a new language for one of the aspects of ubiquitous information systems is not enough to make it a "good" language. The quality of a language must be measured in order to validate the proposal or to improve it if necessary.

The general approach to measure a language quality is to realize empirical evaluations with user experiments. These evaluations are complex because they must not assess the quality of particular instances of the languages (e.g. a particular class diagram), but the quality of the language in general (the class diagram). [25] proposes a framework based on experiments, which can be applied to any type of languages, to evaluate comprehensibility. [35] uses an approach based on the human-information processing model Goals Operators Methods and Selection Rules (GOMS [36]) to evaluate UML diagrams. The authors measure the execution time to realize some UML diagrams so as to determine their complexity.

Another approach to measure a language complexity is to study its meta-model. A complex meta-model should lead to a greater expressive power and thus to smaller models [15]. Domain-specific languages such as the ones for ubiquitous information systems should have more expressive power. Therefore, they must be closer to the experts' knowledge, but as a consequence, they may be more difficult to learn for a novice. [26] explains that there exists an intrinsic dependency between the meta-model and the learnability of a language: a modelling language is composed of a set of diagrams for which some metrics are calculated. The conceptual complexity of a diagram is a sum vector of the above diagram metrics while the complexity of the whole language is a sum vector of the complexity of its diagrams. In such a view, the relations between diagrams are not considered. However the approach has permitted to compare several object-oriented languages and to conclude that object-oriented languages become more complex with time. Using the same approach, Siau and Cao cited in [27], show that UML is more complex than other object-oriented languages by a factor 2 to 11.

Based on their quality framework, Krogstie et al. have also evaluated UML in its 1.4 version. They concluded that UML is difficult to comprehend because there are many fundamentally different concepts, which are not always formally defined.

These works that measure the quality of languages are good examples to follow for the creators of conceptual languages for ubiquitous information systems. Without conducing users experiments, the meta-model of a new language can be used to evaluate its complexity. Then empirical studies can be achieved to measure its appropriateness.

V. QUALITY IN MODELLING PROCESSES

Processes are used to guide the design activity. They are the most effective way to improve the quality of products, in particular of models. Therefore achieving a good quality of modelling processes leads to higher quality in information systems and in their models.

A. *Dealing with immature specific processses*

In the specificities of ubiquitous information system design processes, we have specified that such processes are often immature. One way to strengthen them is to measure their quality so as to identify their strengths and their weaknesses. The first step for immature processes is to evaluate the quality of the resulting information system: is it useable? Is it secure? Etc… Achieving some concluding results for even one of this aspect would be a very useful contribution for designers.

When processes will become more mature, it is important to transfer these specific processes to industrial designers. It is the only way to ensure that industrial ubiquitous information systems take into account the specific practices in terms of models and activities.

B. *Coordinating processes*

Each aspects (human-computer interaction, security, functionalities …) can be designed following some specific processes. If the quality of these specific processes is important

and difficult to achieve, we must also address a more complex challenge: evaluating an orchestration of these processes build to design all the aspects of an ubiquitous information system.

The specific processes must be coordinated so as to give rise to consistent models e.g. the design of the interactive part of the ubiquitous system must be consistent with the design of its functionality. For instance, if the ergonomic specialist chooses an interaction with a wand to manipulate 3D buildings, the software engineer must specify the operations on buildings corresponding to the manipulation. So two types of joint endeavours are needed as defined by [30]:

- Coordination occurs when tasks are decomposed into common activities. A common planning is set up, actors are dispatched on these activities and a proper goal is assigned to each actor or activity. Drawing semantic relationships between models is a typical coordination activity example.
- On the other hand, cooperation consists in defining a common action for different actors, who share, at least partially, a common goal (and product) and organize their actions independently. Specialists filling different headings in a common form are for instance realizing a cooperation activity.

We can wonder whether this orchestration of processes is efficient: are coordination and cooperation activities well-situated in the global process so as to provide consistent models; what is the effort required to perform the design; is there duplicated activities… So in ubiquitous information system design where specific processes can be numerous, it is important to find a way to express, but also to evaluate the process orchestration.

C. *Defining the quality of modelling processes*

Defining the quality of a process is far from being trivial. If we consider mainly the sequencing of activities in a method, we can refer to work on method evaluation to define the quality of processes. In particular, the Method Evaluation Model [32] incorporates two aspects of method "success", actual efficacy and adoption in practice (Fig. 5):

- Actual efficiency: the extent to which the method reduces the effort required to performed the task;
- Actual effectiveness: the extent to which the method improves the quality of the results;
- Perceived Ease of use: the extent to which a person believes that using the method would be free of effort;
- Perceived usefulness: the extent to which a person believes that the method would be useful;
- Intention of use: the extent to which a person intends to use the method;
- Actual usage: the extent to which the method is actually used.

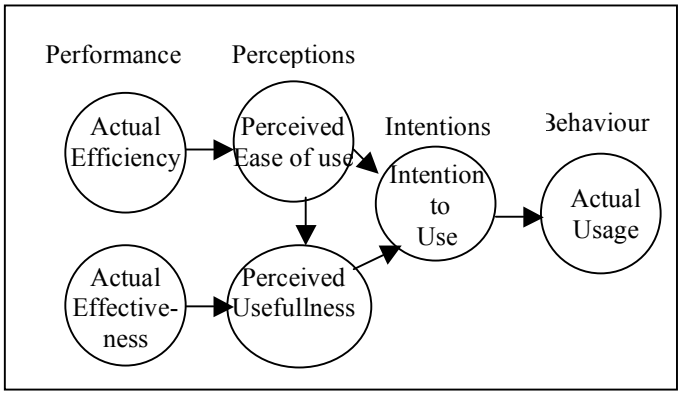

Figure 5. The Method Evaluation Model

The characteristics of the Method Evaluation Model can be applied to processes. The quality of a process can be evaluated through its performance (actual efficiency and effectiveness) but also according to the perceptions, intentions and behaviour in use of its potential users.

This model can be used to characterize the quality of specific process, but also of their orchestration. The evaluation of an orchestration will be very difficult to achieve: the quality of its composing processes must, of course, be defined before considering its own; then the most complex task will be to measure the perception of composed processes.

D. *Measuring the quality of modelling processes*

The quality of processes is generally measured by evaluating the process model. For instance, [38] studies the characteristics that make a process model understandable. Many other works like [38] have been realized in the domain of business process models. Even if they are sometimes limited to the automated business processes, they may be a source of inspiration to evaluate the models of processes: they base their evaluations on metrics or empirical experiments to understand the quality characteristics of business process models and to prevent designers from modelling errors.

However the task is more difficult for modelling processes that are often imprecise and immature. In such case, empirical evaluations must not be missed. Without them, it is impossible to measure the perception or the intention of the process users. We must also evaluate the actual efficiency of processes, which gives some idea of the complexity and the usability of processes. D. Moody shows that such process evaluation is possible: he conducts experiments in order to compare the perceived ease of use and usefulness and the intension of use for some methods proposed for documenting and maintaining large data models [37]. Such work is a good example for processes proposed to support ubiquitous information system design.

If we consider that process quality has an impact on the quality of products (models or softwares), improving the quality of processes will lead to higher quality in ubiquitous information systems. However specific processes for ubiquitous information systems are rarely identified, nor specified. The knowledge on ubiquitous system design must be capitalized in order to widespread such systems but also to help

designers in producing good models and then good systems. While specifying them, ubiquitous system specialists must take care of the quality of their proposals and try to measure it, in particular with empirical evaluations.

VI. QUALITY IN MODELLING TOOLS

Modelling tools provide support for dealing with models in more automated ways. Nowadays, they mainly belong to two categories: CASE tools, like Rational Rose, and MDE tools to define meta-models, to transform models etc. Those tools are used in different activities of the design process. Due to the fact that the designers' requirements in terms of models management are various, CASE tools and MDE tools vary in functionalities. Different tools manipulate different models and perform different actions on models (e.g. transformation, consistency checking,…). Our goal in this section, is not to detail all the possible modelling tools, but to focus on requirements for tools supporting ubiquitous information systems modelling.

In the previous sections, we have noticed the following modelling requirements for ubiquitous information systems:

- The management of a large variety of models and of immature modelling languages, which induces then use of the MDE approach;
- The management of different profiles for designers and stakeholders;
- The identification and the specification of processes.

These three main needs must be supported by appropriate tools.

A. Interoperability between modelling tools

Ubiquitous information system design needs MDE tools, which support domain-specific languages and models. With such tools, specific languages can be specified, the syntactic quality of models can be checked and so on.

One of advantages of using the MDE approach, is that all the languages will be defined in the same manner i.e. by specifying its meta-model. Then it becomes possible to create links between models, to check their consistency or to transform a model into another one: it means that modelling tools must interoperate.

B. Changeability of modelling tools

Domain specific languages are generally immature: they have been proposed, but they have not been evaluated neither by empirical experiments, nor by specific metrics. So we can imagine that they will evolve over time. This induces that modeling tools for these specific languages must also be able to easily change. For instance, they must provide functionalities for easily adding or modifying languages and their graphic representations.

C. Ergonomics of modelling tools

The specific aspects of ubiquitous information system can be designed by specialists, such as ergonomics, who are not experts in computer science. It would be valuable to provide them usable tools, where models can be easily modified or evaluated. Tool builders cannot suppose that any designer can easily use tools that manage many models and their consistency. The complexity of actual modelling tools suggests the need for more ergonomic support. In particular, it is time to introduce into modelling tools interaction techniques to visualize a large amount of information or to navigate through graphs.

D. Support of flexible processes

Process is the component of a method, which is the least supported by tools. Generally, if tools propose methodological guidance, it is "hard-coded": it not possible to change the process or to add a new one. However tools must support process engineering with operations such as the selection of an appropriate process, the adaptation of the chosen process or its orchestration with another process.

E. Quality of modelling tools

The quality of modelling tools can be considered as a special case of software quality. Software quality is widely studied and has give rise to standards such as the ISO/IEC 9126. The ISO/IEC 9126 software quality standard is one of the most, widespread quality standard available in the software engineering community. It fixes six top-level characteristics which are refined into twenty-seven sub-characteristics which are in turn decomposed into properties that the software products belonging to the domain of interest exhibit:

- functionality, which represents the set of attributes that bear on the existence of a set of functions and their specified properties. The functions are those that satisfy stated or implied users' needs. The sub-characteristics of functionality are suitability, accuracy, interoperability, compliance and security.
- reliability, is a set of attributes that bear on the capability of software to maintain its level of performance under stated conditions for a stated period of time. It is decomposed into maturity, recoverability and fault tolerance.
- usability, which regroup attributes that bear on the effort needed for use, and on the individual assessment of such use, by a stated or implied set of users. Usability is based on learnability, understandability and operability.
- efficiency, states for a set of attributes (time and resource behaviours) that bear on the relationship between the level of performance of the software and the amount of resources used, under stated conditions.
- maintainability relates the effort needed to make specified modifications. Its sub-characteristics are stability, analyzability, changeability and testability
- portability refers to the software ability to be transferred from one environment to another. It refers to its installability, its replaceability and its adaptability.

Fig. 6 focuses on quality characteristics that we have identified of prime importance for modelling tools supporting ubiquitous information systems design: 1) its functionality (its suitability to the users' practices in terms of languages and processes and its interoperability with other tools), 2) its usability especially for non software engineering designers, 3) its maintenability, with in particular its changeability necessary to manage the immaturity of the ubiquitous information system design domain.

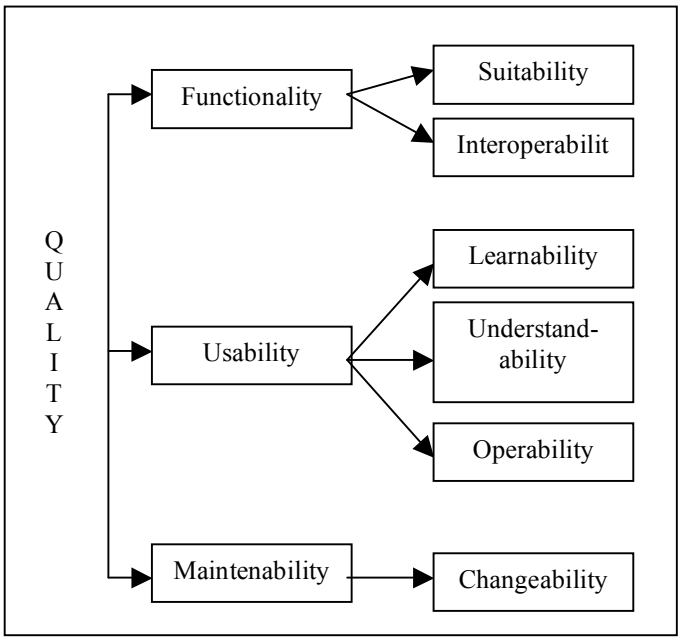

Figure 6. Main quality characteristics for modelling tools

F. Measuring the quality of modelling tools

If many works have treated the evaluation of softwares in general, few work have focused on modelling tools.

Unhelkar [33] provides a checklist for evaluating UML-based CASE tools, with criteria such as "compliance with UML" and "Ability to follow a process". Such a list can be developed for modelling tools in general. It can contain information such as the aspects (devices, context, ..) of ubiquitous information system design taken into account by tools. However it will not be possible to identify a complete list for ubiquitous information system design tools for which the modelling domain is immature.

Moreover experiments must be undertaken to validate more human aspects such as usability, which is often forgotten while building tools. In particular, the MDE tools, which are necessary to support ubiquitous information systems design, must be evaluated in order to determine whether they are usable by non-MDE specialists. We guess that their textual representation of models can be a problem for some designers, not specialized in modelling.

VII. CONCLUSION

The design of ubiquitous information systems has its specificities: new people use specific models following appropriate processes. These new practices are often immature and need to be improved in order to achieve quality in design. We have tried to understand this challenge by identifying the quality needs and characteristics for models, languages, processes and tools dedicated to ubiquitous information system design.

To summarize the quality aspects that we have described for every component of a method, we can use the meta-model proposed by [34]. Quality cannot be defined independently from its goals. Quality goals can be decomposed into sub-goals. Each goal is implemented by a target and is related to a stakeholders' purpose. They are realized by properties that are artifacts or activities that are needed to achieve a quality goal. Properties are achieved by practices such as modeling conventions or training, and are evaluated either quantitatively or qualitatively with an evaluation method. For ubiquitous information systems, this vision of quality has the advantages of 1) not forgetting the various stakeholders; 2) bringing the necessity of practices to light; 3) and making explicit the need of evaluation methods. This generic meta-model can be applied to any contribution (model, language, process, tool) in the design of ubiquitous information systems.

Referring to this meta-model or to the quality frameworks can help ubiquitous information systems design initiators to improve their proposals about models, languages, processes or tools that must be systematically studied and evaluate so as to insure quality. It can be a step in order to achieve quality in ubiquitous information system design.

ACKNOWLEDGMENT

Thanks to G. Godet-Bar for his useful comments on this article.

REFERENCES

[1] G. Godet-Bar, S. Dupuy-Chessa, D. Rieu, "When interaction choices trigger business evolution", 20th International Conference on Advanced Information Systems Engineering (CAiSE'08), LNCS 5074, Springer, Montpellier, France, pp 144-147, June 2008.

[2] L. Palen, "Beyond the Handset: Designing for Wireless Communications usability". ACM Transactions on Computer-Human Interaction, 9(2), pp. 125-151, june 2002.

[3] D. Rieu, « Ingénierie des Systèmes d'Information », Mémoire d'Habilitation à Diriger les Recherches, Institut National Polytechnique de Grenoble, 1999. (in french)

[4] K. Siau, X. Tan, "Improving the quality of conceptual modeling using cognitive mapping techniques", Data & Knowledge Engineering, vol 55, pp 343-365, 2005.

[5] E. Dubois, P.D. Gray, N. Nigay, "ASUR++: a Design Notation for Mobile Mixed Systems". Interacting With Computers, Vol. **15,** pp. 497–520, 2003.

[6] K. Henricksen, J. Indulska, "Modelling and Using Imperfect Context Information", Proc the CoMeRea workshop at Percom'2004, Orlando, USA, 2004.

[7] B. Unhelkar, "Verification and Validation for Quality of UML2.0 Models", Wiley, 2005.

[8] C. Bastien, D. Scapin, "A validation of ergonomics criteria for the evaluation of the human computer interfaces", International Journal of Human-Computer Interaction, Vol 4, N° 2, pp. 183-196, 1992.

[9] H.C. Purchase, L. C. Colpoys, M. Mac Gill, D. Carrington, C.Britton, "UML Class Diagram Syntax: An Empirical Study of Comprehension"",

Proc. Australian Symposium on Information Visualization, Vol. 9, pp. 113-120, 2001.

[10] C. Lange, M. Chaudron, "Managing Model Quality in UML-based Software Development", Proc. Of the 13th Int. Workshop on Software Technology and Engineering Practice (STEP'05), pp. 7-16, 2005.

[11] G. Shanks, R. Weber, "Research commentary: Information Systems and conceptual modeling: a research agenda", Information Systems Research, Vol. 13, Num. 4, pp. 363-376, 2002.

[12] O. I. Lindland, G. Sindre and A. Solvberg, "Understanding quality in conceptual modeling". 1EEE Software, pp 42-49, April 1994.

[13] D. Moody, G. Sindre, T. Brasethvik, A. Solvberg, "Evaluating the Quality of Information Models : Empirical Testing of a Conceptual Model Quality Framework", Proc. Of the 25th Int. Conf. on Software Engineering, IEEE, pp 295-305, 2003.

[14] J. Krogstie, "Integrating the Understanding of Quality in Requirements Specification and Conceptual Modeling", Software Engineering Notes, ACM SIGSOFT, 23(1), pp 86-91, Janvier 1998.

[15] P. Mohagheghi, J. Aagedal, "Evaluating Quality in Model-Driven Engineering", Int. Workshop on Modeling in Software Engineering (MISE'2007), IEEE, 2007.

[16] I. Solheim, T. Neple, "Model Quality in the Context of Model-Driven Development", 2nd Int. Workshop on Model-Driven Entreprise Information Systems (MDEIS'06), pp. 27-35, 2006.

[17] C. Lange, R. Chaudron, "Defects in Industrial UML Models – A Multiple Case Study", Workshop on Quality in Modelling at MODELS'2007, Nashville, USA, pp. 50-64, 2007.

[18] N. Bolloju, F. Leung, "Assisting Novice Analysts in Developing Quality Conceptual Models with UML", CACM, Vol. 49, num 7, pp. 108-112, July 2006.

[19] M. Manso, J. Cruz-Lemus, M. Genero, M. Piattini, " Empirical Validation of Measures for UML Class Diagrams : a Meta-Analysis Study", Workshop on Quality in Modelling at MODELS'2008, Toulouse, France, pp. 59-73, 2008.

[20] J. Lemaitre, J.-L. Hainaut, "A Combined Global-Analytical Quality Framework for Data Models", Workshop on Quality in Modelling at MODELS'2008, Toulouse, France, pp. 46-58, 2008.

[21] J. Nelson, G. Poels, M. Genero, M. Piattini, "Quality in Conceptual Modeling: five examples of the state of the art", Data & Knowledge Engineering, vol 55, pp 237-242, 2005.

[22] D. Woody, "Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions", Data & Knowledge Engineering, Vol 55, pp. 243-276, 2005.

[23] J. M. Favre, "Towards a basic theory to model model driven engineering". In Workshop on Software Model Engineering (WISME), Lisboa, Portugal, 2004.

[24] S. Dupuy-Chessa and E. Dubois, "Requirements and Impacts of Model Driven Engineering on Mixed Systems Design", 1ères journées sur l'Ingénierie Dirigée par les Modèles (IDM'2005), ISBN 2-7261-1284-6, Paris, France, pp 43-54, June 2005,

[25] J. Aranda, N. Ernst, J. Horkoff and S. Easterbrook, "A Framework for Empirical Evaluation of Model Comprehensibility", Int. Workshop on Modeling in Software Engineering (MISE'07), IEEE, 2007.

[26] M. Rossi, S. Brinkkemper, "Complexity Metrics for System Development Methods and Techniques", Information Systems, Vol. 21, num. 2, pp. 209-227, 1996.

[27] J. Krogstie, "Evaluating UML Using a Generic Quality Framework", chapter UML and the Unified Process, Idea Group Publishing, pp. 1-22, 2003.

[28] L. Nigay, P. Salembier, P. Renevier, L. Pasqualetti, T; Marchand, "Mobile and Collaborative Augmented Reality: A Scenario based design approach", In *Proceedings of Mobile HCI2002, Pisa. Lecture Notes in Computer Science (LNCS), Vol. 2411*, 2002.

[29] J. Ralyté, C. Rolland, "An Approach for Method Reengineering". Proceedings of the 20th International Conference on Conceptual Modeling (ER'2001), Yokohama , Japan , November 2001. H. Kunii, S. Jajodia , A. Solvberg (Eds.), LNCS 2224, Springer-Verlag, pp.471-484, 2001.

[30] K. Grebici, E. Blanco, D. Rieu, « Toward non mature information management in collaborative design processes », in: Proceedings of the International Conference on Engineering Design ICED'05, Melbourne, Australia, 2005.

[31] J. Mendling, H. Reijers, J. Cardoso, "What Makes Process Models Understandable?", Proc. Of BPM'2007, LNCS 4714, Springer-Verlag, pp. 48-63, 2007.

[32] D. Moody, " Dealing with Complexity: A Practical Method for Representing Large Entity Relationship Models", PhD Thesis, University of Melbourne, Australia, 2001.

[33] B. Unhelkar, "Verification and Validation for Quality of UML2.0 Models, Wiley, 2005.

[34] P. Mohagheghi, V. Dehlen, "Towards a Tool-Supported Quality Model for Model-Driven Engineering", Workshop on Quality in Modelling at MODELS'2008, Toulouse, France, pp. 74-88, 2008.

[35] K. Siau, Y. Tian, "The Complexity of Unified Modeling Language: A GOMS Analysis", 22th Int. Conference on Information Systems, pp.443-447, 2001.

[36] S. Card, T. Moran, A. Newell, "The Psychology of Human-Computer Interaction", Lawrence Erlbaum Associates, Mahwah 1983.

[37] D. L. Moody, "Comparative Evaluation of Large Data Model Representation Methods: The Analyst's Perspective", In ER 2002 - 21st International Conference on Conceptual Modeling, Tampere, Finland, October 7-11, 2002.

[38] J. Mendling, G. Nemann, W. van der Aalst, "On correlation between process Model Metrics and Errors", Proc. 26th Int Conference on Conceptual Model (ER'2007),Nex Zealand, 2007.