Abstract: this paper presents a new human-centred design approach which derived from conceptual design research and through a critical field research study. The proposed approach is applied to a Medical Dynamic Decision Support System (DSS) supporting the hospital Intensive Care Unit physicians’ decisions in order to predict nosocomial infections occurrence. The design approach in question is then validated.

Keywords: Medical Dynamic Decision support systems; Design; Software Engineering; Human-Computer Interaction, Nosocomial Infections.

1. INTRODUCTION
As the cost of health care continues to spiral up, the health care industry increasingly is looking at information technology (Flower 2004). Medical Decision Support Systems (DSS) are used properly in order to assist physicians in decision-making activity. Nosocomial Infections (NI) pose significant risks for all hospitalized patients and can cause minor discomfort and/or life-threatening events. It also substantially increases the cost of care and length of hospital stay. In order to decrease infection risk and consequently improve patients’ safety, a Medical Decision Support System has to be implemented in order to prevent NI occurrence. The decision given by this system is dynamic because it is based on the patient state described on several levels - in terms of a temporal factors set of which the unit of time is the day. The dynamic decision system is broken up into several stages corresponding to increasing levels of comprehension of the patient situation (scale of time). On each level can be generated a set of knowledge. For this reason our medical system is called Medical Dynamic Decision Support System (MDDSS).

Today there are significant improvements in the medical decision support technologies, concerning data storage volumes, data processing as well as data extraction. We are interested in a new generation of Business Intelligence application and technology which is the data Mining. This technology enriches the medical organization's business intelligence process, in order to achieve new stages in our MDSS. The DSS deals with the decision problem according to the human knowledge and aims to help users in the DSS process from beginning to end. So the Human-Computer cooperation is essential throughout the decision process. In this way, the Human-Computer Interaction (HCI) is a crucial aspect for the interactive decision support systems and its design leans on a user-centred approach. The DSS design process can be treated as a unified and iterative set of activities and operations. A development approach based on Unified Process (UP) methodology and UML (Unified Modelling Language) seems to be appropriate for DSS development. UP aims to incrementally build robust system architecture (Brandas 2007). However this methodology is often limited when the studied system is highly interactive because it does not implicate explicitly and systematically the user (Kolski et al. 2001). In this context, it is possible to propose an enriched SE development approach under the HCI angle. This proposed approach has to be applied to MDDSS design. The paper is organized into 5 sections. In Section 2, the theoretical background for this work is presented. In section 3, a design methodology is proposed; its basic relationships and interactive phases are explained. In section 4, we discuss how we have validated the new MDDSS Design approach through a MDDSS development for fighting against NI in the Intensive Care Unit (ICU). Our conclusions and perspectives end this paper.

2. THEORETICAL BACKGROUND
and the heart. These infections can be directly related to the care itself or simply occur at the time of the hospitalization independently of any medical act. In the ICU, the NI problem is far much more alarming, because the patients who are hospitalized there are more fragile. The importance and complexity of decision-making in NI control have been frequently highlighted in the research literature (Brossette et al. 2000). Work shows their effectiveness and their capacity to produce useful rules. But, such as they are described in the articles, their use by physicians seems difficult. Our system aims at daily estimating the probability of acquiring a NI, in percentage, during the ICU patient hospitalization. This probability is calculated using a data mining technique and according to some temporal factors (such as Urinary catheterization, Intubation, etc.), in order to predict infection risk on the basis of patient records.

At each day, the decision on the patient state depends on the NI probability and thus on the values of those factors to date but also at the previous days, as well as whole of knowledge obtained by learning in time and recording of former events. In fact, a basic decision is taken at the admission of the patient (t0). The future decision refers to a decision to be made after the consequences of a basic decision become (partially) known. A future decision is linked to the basic decision because the alternatives which will be available in the future depend on the choice made in the basic decision now. As time moves on, the future decision at current stage (t) becomes the basic decision at the next decision stage (t+1), when a new Knowledge extracted by data mining (probability of acquiring a NI) and future decision should be addressed. This link repeats itself till the NI is considered to be over. The decide-then learn pattern describes how the decision-maker responds to new knowledge gained during the decision-making process. The elements described above, especially the existence of linked decisions, clearly show that decision-making in NI control is a dynamic process. In this scope, the decision-making process necessitates the consideration in time of linked or interdependent decisions, or decisions that influence each other. This dynamic decision-making pattern is a chain of decide, then learn; decide, then learn more; and so on. Such a system is so called Medical Dynamic Decision Support System (MDDSS). It has to be designed. The design methodology used will be described in the following section.

A System that prescribes behaviours will predetermine the evolution of the design process (Klashner and Sabet 2007). Because “Good decisions result from sound process...”, (Kleindorffer et al. 1993) decision-makers typically define the problem both in terms of a solution within the process design and in terms of a certain life cycle model (Boehm 1988). For that reason, a strong correlation exists between what software process is chosen for the DSS design and the resulting decision support quality (Klashner and Sabet 2007). In fact, the decision of how to conceptualize the system’s evolution largely determines the success of a DSS. The present research is essentially premised on concepts from design research and various qualitative theoretical and methodological approaches. It is now a question of choosing an adequate design approach. In the next section, we will very briefly examine a set of design models in the Software Engineering (SE) field and in the HCI field and to highlight their relevance.

2.2 Design models and methods

Software design models and methods are available in SE and HCI domains. One can find in the literature several analysis and specification processes and methods, intended for the systems modelling. We review a set of these methods, in a representative but not exhaustive way.

A. the Software Engineering field: several development cycles are available: traditional cycles (such as the well-known waterfall, V and Spiral models) and more recent cycles (such as the Y model (André 1993) or also the Unified Process (UP) (Jacobson et al. 1999)). These models aim at the production of quality systems. But the most traditional models are too often directed towards the technical part and not towards the user, except for UP, until a certain level. For this reason, we focus on UP. Because of the inexistent HCI concepts in the SE models, an enrichment of traditional models under the HCI aspect makes the reconsideration of their structure and organization necessary (Kolski et al. 2001).

B. the Human-Computer Interaction field: The SE models remain at the basis of the methods and models used for human-machine interaction, called HCI enriched models. Among these models, we can quote the model of (Long and Denley 1990) which is close to the waterfall model, the Star model (Hix and Hartson 1993), the Nabra model (Kolski 1997) or the U model (Abed 1990; Lepreux et al. 2003). The principal preoccupation of these models must especially underline, from the methodological point of view, the fundamental aspects such as the human tasks modelling, the iterative development of prototypes and the interactive systems evaluation (Kolski et al. 2001). But they do not provide necessarily all the insurance that a project aiming at the design and the development of an interactive system is carried out with a total success. In fact, no perfect model exists; they all have their strong and weak points. So we propose to enrich a SE model under the HCI aspects.

3. PROPOSED METHODOLOGY

In the previous section, we have highlighted limits inherent in the development models. This is the context in which, for the past few years, our research projects have been aimed at defining a theoretical and methodological framework for the design and evaluation of interactive decisional systems (Ltifi et al. 2008) (Ltifi et al. 2009b). This framework is based on a Unified Process (Jacobson et al. 1999). In fact, modelling the system architecture with UML using the UP as main methodology helps the developers to rapidly construct and implement an accurate and scalable interactive system. The UP model offers appropriate framework for interactive system developing. With UP based on UML, developers can create complex decision models and databases (Data Mining
Applications). Moreover UML offers a high level of component reusability.

### 3.1 The Unified Process (UP)

The **Unified Process** consists of a set of generic principles that can be adapted to specific projects (Jacobson et al. 1999). It is thus, to some extent, a process pattern that can be adapted to a large category of software systems, various fields of application, different types of companies, different qualification levels and various project sizes. UP is (1) controlled by the use cases, representing the functional needs of the system, (2) centred on system architecture, which provides the structure used as framework for the work carried out during the iterations, and (3) iterative and incremental, with the aim of reducing complexity by controlling it, by breaking up a data-processing project into sub-projects that each represent one iteration. These iterations indicate the steps in the sequence of activities, while the increments correspond to the product development stages. An iteration takes into account a number of use cases. The aspects of the model being analyzed and designed are based on UML (Rumbaugh et al. 1999). A process defines who does what, when and how in order to achieve a fixed goal (Jacobson et al. 1999). UP has 4 phases: (1) inception, during which the project extent is defined through use cases and feasibility studies; (2) development, during which the needs are defined and the architecture specified; (3) construction, during which the software is built by means of several iterations and various system versions; and (4) transition, during which the system is delivered to the end-users and put into service and these end-users are trained and provided with tech support.

*Why to choose the UP as a starting point?:* On one hand, the UP allows the costs to be limited to the strict expenses of an iteration. It also allows limiting the risks of delaying the installation of the application to be developed. UP also permits potential problems to be identified in the first stages of development, rather than at the testing stage as with the traditional approaches. The rhythm of development can be accelerated because the objectives are clear and have been planned in the short term. This short-term planning is due to the fact that user needs and the corresponding requirements cannot be defined completely in advance. The system architecture provides the structure that is used as a framework for the work carried out during each iteration, while the use cases define the objectives and guide the work completed during the iteration. On the other hand, the UP considers the user’s needs, as was mentioned previously. However, Lemieux and Desmaraiss (2006) showed that the UP is not user-centred according to the ISO 13407 standard; this standard specifies the rules to be followed to adapt a software development process for user-centred design. They also note that the introduction of the use cases is not sufficient to make a design process user-centred. Our approach has to take into account the SE principles, and the HCI aspects. This approach is based on the UP principle of iterative and incremental development, which allows each task accomplished to be evaluated as soon as the first iterations of the development process have been completed. Our approach thus incorporates the continual presence and constant participation of the user.

### 3.2 Our proposed approach

We define a methodological framework relating to interactive system design. The approach recommended by this framework is based on the principles of a centred-user design as defined by (Gould and Lewis 1985) where several principal phases can be proposed, in coherence with various authors having proposed user-centred methodologies (Robert 2003). This general development framework falls under a process being based on the three principles of the generic UP (iterative development, controlled by the use cases, and centred on architecture) and proposing a cutting of the project in four phases (Inception, Development, Construction, Transition). The principal development activities led in an iterative and incremental way. In the approach suggested, we are interested particularly in the activities as illustrated by Fig. 1 where we locate our contribution.

![Fig. 1. The unified Process (Jacobson et al. 1999)](image)

**A. The needs assessment activity** (Fig. 2): it consists of the analysis of the future decisional system objectives, in terms of functional needs and needs relating to the HCI. This stage rests on: (1) the analysis of the normal and abnormal decisional situations allowing the first functional and structural description of the decisional domain; (2) as soon as possible in the project, we have to build first prototypes to imply rapidly the future users in giving them an outline of possible solutions; and (3) the user modelling allowing to represent the different types of users as well as their behaviours. The user model makes it possible to contribute to the centred-user interface design. This model is significant in interactive system development approach. After this expression and identification of needs, an early evaluation of this stage is provided compared to the system constraints (organization constraints, logical and/or temporal constraints, etc.). This evaluation leads to two possible results: (1) if there will be a problem, it is a question of making a feedback and of checking the first three stages described above; or (2) if not, it is a question of breaking up the decisional problem into under problems (or modules) to allow the identification of specific needs related to the decisional problem. Thereafter within the framework of each module, the objective is to identify and
organize the whole of tasks having to be filled by the Human-Computer couple, according to the goals fixed by logic of operation of the future system. These tasks could be divided into two categories: automatic tasks (in which only the applicative aspect is implied) and interactive tasks (which implying both the user and the system). Once the automatic and interactive tasks are defined, it is a question of evaluating them compared to the user’s needs.

B. The analysis and design activities (Fig. 3): the results of the first activity are the automatic and interactive tasks as well as the model of the user. Once the tasks have priority, they must be analyzed and conceived. The analysis of the automatic tasks relates to the analysis of the functional tasks being able to be accomplished within a decision-making process. For the analysis of the interactive tasks, it is significant to underline that this stage relates to the analysis of the human tasks in normal and abnormal situations (Abed 1990). This analysis is in close connection with the identification of the characteristic resources and cognitive limits of the user coming from the user model. The user interfaces can then be analyzed in order to define their behaviour. This analysis is focused on the relations between the user and the interactive system. It is a question of counting the ergonomic and technical needs rigorously, then to define the number of screens to be used, the sequence of the views, the information presentation modes, the human-computer dialogue methods, etc. The specification stage is followed by the design stage which allows carrying out the design models of the automatic tasks and the user interfaces analyzed. The design makes it possible to formalize the needs specifications to arrive at the definition of the algorithms which will be necessary. The algorithms defined in this stage will be transmitted to the implementation stage for being developed. An evaluation of the analysis and design models can be carried out on this level. This evaluation aims at checking their correspondence with the needs. The interest of the evaluations within our approach is that they make it possible to return without awaiting the end of the iteration. Once the specification and the design of the user interfaces are validated, an advanced prototyping of the user interfaces can be carried out with the collaboration of the experts and the users. All the information provided within the framework of these activities makes it possible to specify the architecture of the future system to be realized. This system developed and enriched progressively with the evolution of the iterations of our approach; and this by addition of the analysis and design models (UML models integrating the sequence, communication and interaction overview diagrams). The system architecture is composed of the following elements: (1) the user interface allowing interactions with the user; (2) a database containing information on the users, the procedures, the decision problems, the solutions, etc.; and (3) the software modules. This architecture must be evaluated; and according to the evaluation result, it is a question of doing a feedback or of starting the following activity.

C. The implementation activity (Fig. 4): it starts with a planning. A plan for the development, the levelling and the maintenance of software must be prepared to facilitate the long-term feasibility of this effort of development. The implementation consists in coding the functional parts and the HCI. This coding must respect the algorithms defined previously. The whole of the code components are then assembled and integrated (Meinadier 2002) to have a subsystem in order to build a prototype at the end of the iteration.

Fig. 2. The needs assessment activity

Fig. 3. The analysis and design activities

Fig. 4. The implementation activity

The prototypes are levels of the future software development which hold in account more and more details.
of specifications. In particular, the considerations of the effectiveness, the robustness and maintenance are taken into account from the development beginning (prototyping encourages the data abstraction). An evaluation of each realized prototype can be implemented on this level in order to check its correspondence compared to the system architecture. Each prototype must be validated at the same time by the users, the experts and technically.

D. The testing activity: it consists in making sure in particular that the user is able to carry out his/her task by means of the proposed user interface. The implementation tests will highlight the defects of the code. The detected defects can be functional, in connection with the choice and the performance; or interactive. We distinguish: (1) Unitary tests (Fig. 5): it is a question of testing the functions, which were developed during the iteration. It can begin as soon as the function was coded, checked and if it is necessary accepted. When non-conformity is detected, it is then necessary to correct the errors and the anomalies. If not, the components of code will be gradually assembled with the code tested in the preceding iterations. It is a question of also testing the user interfaces of the subsystem carried out, the way in which each one of them communicates and behaves in the new environment. In this evaluation, we are generally interested in the performances of the total system, according to users’ behaviors at the time of their interaction with the system. The real model of the resulting subsystem is established; it is compared with the ideal model defined in the stage of needs assessment. This comparison allows validating the subsystem compared to the defined needs (Abed 1990). We also evaluate the utility and the usability of the system; please note that interactive systems evaluation is a very rich domain (Jacko and Sears 2003). (2) Integration tests: these tests relate to the evaluation of the complete system. They aim at checking the interactions between the subsystems, in order to make it possible to the user to construct the whole of its decision-making process. This evaluation must ensure that the system allows a complete help and makes it possible to check the quality of the HCI.

Fig. 5. Unitary tests

For the modelling, we use UML (Rumbaugh et al. 1999). In fact, UP is based on UML. In addition it is a language that allows models to be represented without defining their development processes. So it can be used transparently with any other software development process, allowing using this language with the approach that does not define an obligatory modelling language. So our approach consists in proceeding in several iterations from inception to the transition. We have applied the five activities (with their actions described above) in each iteration. The resulting models are represented using UML. In the next section we will present the proposed approach application.

4. CASE STUDY IN HEALTHCARE DOMAIN

In the preceding section we presented our global solution for interactive system development. This approach was applied to a concrete case in the healthcare domain. It aims to help physicians, to understand and prevent NI. It is currently under use in the ICU of the Teaching hospital Habib Bourguiba in Sfax, Tunisia. The design and the realization of this MDDSS were carried out according to the four phases of UP. Each stage of these iterations was carried out while following the actions suggested in this paper. These actions are not carried out with the same intensity in each iteration. Let us take the example of the user modelling: it began progressively at the beginning of the project (UP inception phase), became dominant from the first iteration of the development phase, it is reduced to almost nothing at the last iterations of the construction and transition phases. At the moment we are testing the 9th version of the application. This gives an outline of the difficulties related to this application field, and especially of the need to go ahead step by step in very close link with the users/decision makers suggesting gradually of new improvements and tests, within sight of the results obtained at the time of the preceding iterations. The development of the MDDSS based on the data mining was subdivided in two modules (applications): (1) one for the storage and the preparation of the data; (2) and the other one for the data mining application. For the first module, we studied especially the context and the definition of the objectives by organizing frequent meetings with the physician who deals with the problem of NI, and having the double role of decision maker and user. Sometimes he invited his collaborators to attend the meetings.

Concerning the data mining module and according to the needs expressed by the experts (users), we developed an application for the daily prediction of the NI appearance in the ICU. The data mining technique used is the K Nearest Neighbours (KNN) (Riesbeck et al. 1989). Indeed, the principle consists in identifying the K nearest neighbours basing on the calculation of the distances between the patient characteristics and those of the patients already hospitalized and recorded in the ICU database. The distance that we used to calculate the neighbours is the Euclidian distance. With an aim of having the best rate of discrimination, we chose three values of K which are (1, 3, 5). If the probability is higher than 50%, there is a great risk to have a NI. To apply the KNN algorithm, we implemented two user interfaces. The first is devoted to the calculation of the distances and the sorting of KNN. The second shows the probability of the NI appearance. The KNN algorithm is applied each day during the patient hospitalization. We quote in this article the example of a patient of which the probability of having an infection is modified (it is 0% the first day and 60% the third day (Fig. 6)); and this is because of some risk factors such as the antibiotics catch. Such a prediction helps the physicians to take the precautions necessary in order to protect the patient. The second module was held in 13 iterations.
improve the proposed approach. We wish also to propose a methodology for evaluating MDDSS taking into account evaluation methods and techniques used in two fields: HCI and visual data mining (Ltifi et al. 2009a).

6. REFERENCES
André, J. (1993). Le cycle de vie en Y. Arche SQL.

5. CONCLUSION
Current DSS Development frameworks and methodologies must be able to integrate the human-computer interaction aspects. That is why we propose an approach using the UP and UML, in which our contribution is to detail and enrich each activity according with HCI aspects. Developers must have a unified view of system design and implementation. In order to create and develop a MDDSS in an integrated manner based on activities workflow which will be able to identify, model and implement accurately the decision-making requirements, we recommend the usage of the proposed approach. This approach should allow a realistic and well documented system able to meet most user requirements and support the development of flexible and accurate DSS. In order to validate this approach, we currently develop a MDDSS aiming at the supervision of the NI contracted in ICU. The realization of the project is done in constant connection with the physicians, users of the system: needs assessment, tasks definition, identification of the users’ characteristics, and prototyping, evaluations and validations with them. Its finalization of is not totally completed. Further works will be held to improve the proposed approach.