INF-3996

MASTER'S THESIS IN

TELEMEDICINE AND E-HEALT

Writing Electronic Nursing Care Plans. An Approach to Facilitate Navigating the Standardized Nursing Vocabularies NANDA and NIC

Harald Igesund

June, 2007

FACULTY OF SCIENCE
Department of Computer Science
University of Tromsø

INF-3996

MASTER'S THESIS IN TELEMEDICINE AND E-HEALTH

Writing Electronic Nursing Care Plans. An Approach to Facilitate Navigating the Standardized Nursing Vocabularies NANDA and NIC

Harald Igesund

June, 2007

Preface

During my 25 years experience working in the health services, I have seen numerous nursing documentation systems come and go. Some seemed well-founded and thorough, others less so. Never the less, they all vanished. Sometimes they were to cumbersome, other times the enthusiasts that kept them alive disappeared, and new enthusiasts with a different view on how nursing documentation should be done showed up. But all in all I have seen no lasting improvement in nursing documentation over these years. 2 years ago I participated in a project where we introduced the use of electronic care plans using the standardized languages NANDA and NIC. This improved documentation, and the effect is still lasting. Then, and still, I am not convinced that the cost justified the benefit. The cost can be lowered, and that is some of the motivation behind this thesis. The benefits, together with better documentation, were a greater consciousness among nurses and other health professional what nursing was all about. I have a pragmatic relation to standardized nursing languages; they are clearly not the silver bullet to solve all our problems, and I see a lot of technological and subject matter issues to be solved if it is going to be really useful. But when we, in our project, looking only at the immediate benefits to ourselves, found it valuable, I feel that the total potential benefits of standardization is worth exploring the idea.

I had hoped to get access to the electronic version of the Norwegian translation of the NIC classification for this thesis in order to implement a complete search engine also for this classification, analogous to what is done for the NANDA classification. Unfortunately, this was not possible because of unclear licensing rights.

Acknowledgments

First of all I will thank my primary supervisor Gunnar Hartvigsen, who was able to help me structure and arrange some incoherent thoughts into this thesis. Gunnar Ellingsen, my external supervisor, deserves thanks for helping with the design of the usability tests, and additional advices.

Others who deserves thanks:

Kathy Mølstad from NSF and Marit L. Berntsen, Akribe Forlag, who let me use the electronic version of NANDA, without which this project would have been impossible.

Gunn von Krogh, who promptly let me use the electronic versions of the diagnostic - and interventions guides in the KPO-model.

Grete Furu and Laila Engen, leader and subject matter expert nurse at Spesialpsykiatrisk Avdeling (SPA), who kindly let me conduct usability testing during working hours at SPA, UNN. Gro Wangensteen, colleague and subject matter expert nurse at Alderpsykiatrisk Post, SPA, for countless discussions and honest feedback.

All the personnel at Alderspsykiatrisk Post, and especially those who participated in the tests.

And a special thank you to my dear wife Elsa Stiberg, for support and encouragement during the writing of this thesis.

Tromsø June 1, 2007 Harald Igesund

Contents

Summary

The introduction of the standardized nursing languages NANDA and NIC to write electronic care plans is hard for nurses, as reported by several projects. Both the need to relate to a new software tool and to the totally unknown domains of the standardized languages contributes to this. With an existing tool for care plan writing as basis, we have used the unified process to develop an improved tool, in order to make the introduction smother. Care has been taken to make the interface and workflow as intuitive as possible, allowing novice computer users to use the tool. An automatic presentation of the information in the languages is implemented, to further meet the needs of unskilled users. To ease navigation in the classifications of the standardized languages, a three level harmonized classification for the two languages was chosen. The two uppermost levels of the classification are presented in one screen, resulting in an efficient browsing for NANDA diagnoses and NIC interventions. To make it possible for users unacquainted with the language to find NANDA diagnoses, a search facility was developed. An information retrieval method was implemented, making it possible to search for diagnoses on the basis of signs and symptoms. The query interface was deliberately made as simple as possible; one or more keywords are entered, and a list of diagnoses is returned ranked according to relevance to the keywords. The search facility also helps in the process of exploring and learning the classification hierarchy. Usability tests shows improvement both in ease of use and support for exploring the standardized languages for the new system compared with the original.

1 Introduction

1.1 Background

In our technological and computerized world the need for standardization is evident. The transition from paper records to electronic health records (EHR) implies that the demand for standardization within the documentation of health services will be increasing. The Norwegian Nursing Association has recommended that nurses start using standardized vocabularies to write nursing care plans in order to evaluate these vocabularies' usefulness and completeness (NSF 2007). However, the size and complexity of many of these vocabularies indicates that it is a challenge to learn to master them and to use them to their full extent, which is a prerequisite for a thorough evaluation. Most EHRs offer the functionality of care plan writing, and a few also support the usage of standardized vocabularies, but true support for navigating and finding one's way within the vocabularies is lacking.

In an project at the University Hospital in Northern Norway (UNN) (Wangensteen et al. 2005), introducing the use of nursing care plans in DIPS (DIPS 2007), a Norwegian EHR, shortcomings in the EHR's support for navigating the classifications were found. One of the project's goals was that the nurses should use North American Nursing Diagnoses Association (NANDA) and Nursing Intervention Classification (NIC) to write nursing diagnoses and interventions, and only revert to free text diagnoses and interventions if they could not find appropriate descriptions in the vocabularies. One lesson learned from the project was that nurses took a long time to navigate in the classifications, but still gave in and used free text on diagnoses or interventions that after closer inspections showed to have their parallels in the classifications. Another lesson was that it was hard for users, especially those who were not fluent PC-users, to learn a new user interface with an intricate work flow and at the same time relate to the totally new domain of standardized vocabularies. For those who managed to master the work flow, it was necessary with frequent use in order to consolidate the skill; infrequent users often had to start the learning process at point zero if the time period between using the application was long. Some of the blame for this was ascribed a poorly developed search facility in the electronic health record, and a difficult and not very intuitive user interface for interaction with the integrated vocabularies (ibid). However, other projects implementing the use of standardized vocabularies together with a new computer system has had similar experiences:

> "[nurses] felt like going to a foreign country where you had to speak the language; to make matters worse, you had to go to a new country every day. More prosaically, they said that they felt they were going from being experts to novices."(Bowker et al. 2001)

Also the users of the HANDS-tool, a tool specially developed to navigate nursing vocabularies, experienced that it took a long time to master it (Keenan et al. 2002). This implies that other project using other software have had similar experiences regarding how difficult this process is. Our impression is that the user interface and searching facilities are equally bad or worse, or in most cases, totally lacking, in the few other electronic health records that support standardized nursing vocabularies.

1.2 Problem statement

On the basis of the experiences from the project at UNN (Wangensteen et al. 2005), in which we introduced nursing plans in psychiatric treatment, we came up with the following problem statement:

How can we make a tool for care plan writing that makes the process of learning NANDA and NIC easier, and that offers an easy interface for interaction between the user and the vocabularies?

This gives us the following sub-problems:

1. *How can we construct a tool that is as intuitive as possible, helping users not familiar with common Windows tasks to find their way around?*
2. *How can we support easy browsing through the classifications?*
3. *How can we create an efficient search facility to search for diagnoses and interventions?*

1.3 Methods

With DIPS' solution for care plan writing as a starting point, we have closely followed the Unified Process (UP) (IBM 2001) while developing our system. The UP is an iterative and incremental process that consists of a number of different disciplines and phases. We have chosen to confine our project within the disciplines requirements, analysis & design, implementation and test, and also within the phases inception and elaboration. As a result of this, our final system in this project can be considered as a not yet finished prototype. In addition to the UP, we have used parts of the goal-directed design framework (Cooper and Reimann 2003) and parts of usability engineering (Nielsen 1993) to better understand the functional requirements of our users.

1.4 Limitations

In this thesis, the focus is *not* on user interface, but rather on the work flow and interaction with the vocabularies. Because of the need for usability tests comparing our system with DIPS, the interface is deliberately made as equal as possible to DIPS, with the same colors and concepts, and the language is also Norwegian. DIPS uses another terminology than in the official Norwegian translations of the languages. For diagnosis, intervention and activity the DIPS' versions are "behandlingsplandiagnose", "behandligsplantiltak", and "forordning".

Interfaces to the rest of the EHR are not discussed. Connections to other parts of a documentation system, as for instance to the patient status note in the KPO-model (Krogh and Dale 2007) to give reasons for changes in the care plan are neither discussed.

DIPS has in later versions got evaluation items, that to some extent take care of the need to reason for changes in the care plan and to document outcome. These are implemented in our system but not discussed in this paper.

Because DIPS only has support for NANDA and NIC, these are the vocabulary used in the project. Since we did not get hold of the complete NIC – classification, NANDA is the classification that is used for most of the discussions.

Legal requirements as timestamp, author etc. for an EHR document is deliberately omitted.

1.5 Results

We have developed a system that in all areas defined in the problem statement is better than DIPS.

Usability tests show that the work flow is more intuitive in our system than in DIPS.

Our browsing facility is based on the three level classification hierarchy found in the KPO-model (Krogh and Dale 2007). Again usability tests show that this hierarchy is far more efficient and easier to navigate than the 2 level hierarchy offered by DIPS.

The key word search is implemented as a small Information Retrieval (IR) system. It is only implemented for the NANDA – classification. Diagnoses can be found based on key-word search after sign and symptoms, and diagnoses matching the search are ranked based on relevance to the key-words. The search functionality shows promising results regarding search in it self, and has also become a valuable tool to easier learn to know the KPO-classification. Additionally, it can be used to ease differential diagnosing.

1.6 Organization

Chapter 2 Theoretical Framework

This chapter gives an overview of nursing documentation framework and nursing standardized languages. State-of-the-art solutions of tools to create electronic care plans are presented. The chapter concludes with a presentation of the theory behind the vector model for information retrieval.

Chapter 3: Method

The Unified Process is presented in extent, followed of a few concept and ideas from the Goal-directed Design and usability engineering.

Chapter 4: Requirements Specification

The background for the requirements is presented, followed by the more superior vision of the new system. Functional requirement is presented in the form of use cases.

Chapter 5: Design

In the two phases inception and elaboration of the Unified Process, the design process is described. Sequence diagrams and domain diagrams are used to find an overall architecture of the system.

Chapter 6: Implementation

A detailed description of the implementation of the different packages of the system is given. The database, inverted index, care plan package and the user interface are illustrated through UML-diagrams.

Chapter 7: Results

A sequence of figures to illustrate the workflow in the two systems is initially given. Following is a presentation of the result of two usability tests aimed to measure workflow and navigability in the classification. The chapter concludes with a listing of experiences drawn from using the search functionality.

Chapter 8: Discussion

The result of the usability tests are first discussed, and proposals to further improvement of the system are presented. The chapter concludes with a discussion regarding the search facility. Both implementational aspects and experiences from using the feature are discussed

2 Theoretical framework

The work done by nurses has traditionally been almost invisible (Bowker 2003). The documentation of their work has not been needed or considered relevant for accounting or research purposes, and also among nurses themselves many has claimed that their place is by their patients' bedsides and not at the desk writing records. Thus, both external and internal forces have contributed to the erasure of nursing documentation (ibid). This picture is now changing, with new legislations that demand the nurses' documentation, and new initiatives for research among nurses.

2.1 Legislation

In the western world, legislation has secured the user of health services the right to access their health record, and also a right to participate in and shape their own treatment (Norris 2002). It is also a prerequisite that the patient gives his consent to treatment. To achieve this, strong conditions of the quality of the documentation must be set. In Norway, all personnel that perform health services on an independent basis, like nurses, have an obligation to document their work (HOD 2001). The content of the health record is regulated in "Forskrift for pasientjournal"(SOH 2000). The record shall serve different purposes, i. a.

- give an overview of actual and planned treatment
- give an overview of observations and assessments
- document outcome of treatment
- ensure communication between different health personnel
- be a basis for referral and transferal documents
- be a vehicle for the patient to get to know his health condition
- be part of the quality assurance and internal control of the enterprise
- be a basis for audits
- be a basis for research, education and training

2.2 Care plans

Nursing documentation can be categorized in 3 different formats, *sequential notes*, *care plans* and *documentation of deviations (Moen et al. 2002). Sequential notes* are chronological notes containing observations, evaluations and descriptions of incidents. Sequential notes are of different types, like shift reports, assessment notes, telephone notes and so on. The chronological nature of shift reports often only gives snapshots of a patient's situation at a given instant. *Documentation of deviations* is used together with care pathways and guidelines. Only deviations of the defined treatment are reported (ibid).

In order to give a more process oriented documentation, the nursing process and the *care plan* was introduced. The nursing process is an iterative process consisting of the *assessment* of the patient's need, *diagnosis* of human response needs (that nursing can assist with), *planning* of patient's care, *implementation* of care, and *evaluation* of the success of the implemented care (Wikipeda 2007b). The result of the nursing process may be concretized in a care plan, where the defined diagnoses, their goals and their interventions are written down. Important aspects of the care plan are (Wikipeda 2007a) (Moen et al. 2002):

- it focus on solving or minimizing an existing problem
- it is a product of a systematic process
- it relates to the future
- it adds structure when existing problems are identified
- it is easy to see which problems and interventions are identified
- the outcome is related to identified and understood problems and interventions

A drawback of care plans is that it is hard to find a format to properly organize the documentation. It has also traditionally imposed space problems to get the information within the proposed format when written on paper. If the patient additionally has lots of or composite problems, it has a tendency to become over-complex.
Because of the effort needed to write care plans, and also to change them, they have often been outdated or not written at all(Børmark 2007, Bowker 2003).
With the advent of the electronic medium, some of the earlier difficulties regarding format and editability are mitigated. Because of this and the emphasis of structure, overview and future

treatment, which secure many of the legal documentation requirements, the care plan is becoming central in electronic health records and documentation frameworks.

A lot of effort has been laid down to make template care plans. The most common diagnoses, goals and interventions relating to for instance a certain patient group or a certain ward are pre-entered into a plan, and the nurse selects those items that are appropriate for the actual patient. Functionalities to create template care plans are often incorporated into EHRs

2.3 Classification

The act of classification can be described as putting items, concepts or ideas into a set of boxes, real or imaginary (Bowker and Star 1999). A classification system should ideally have the following properties (ibid.):

1. *A unique and consistent classificatory principle*. A common classificatory principle is the genetic or inheritance principle; things are classified by their origin, an example is the animal kingdom classification. Other examples are temporal order, e. g. sorting letters by receiving date, or functional order, e. g. sorting recipes by frequency of use.
2. *Mutually exclusive categories*. There should be no doubt about which box a concept should be put into, and once put there, it should stay there for ever.
3. *It should be complete*. The total domain of the area described by a classification should be covered. All items or concepts of the area should fit into the boxes.

An ideal classification system has never existed, and probably never will. People disagree about or misunderstand classificatory principles, and sometimes local or not foreseen conditions generate sub classifications or new ways of order categories within a classification (ibid).. Regarding mutually exclusivity, there is almost always disagreement or ambiguity into which category items should be put, because concepts and terms are subjective, and also context dependent and may change over time (Coiera 2003). When it comes to completeness, political, ethical or economical considerations may dictate not to include items, sometimes for god reasons. In the medical world this may be diagnoses that are controversial or socially stigmatizing (Bowker and Star 1999). So the question whether a classification is complete is really a question of whether it is complete enough for the task it is going to fulfill.

The phenomena that nurses describe are overlapping, it is almost always impossible to clearly distinguish which category to place them in. This implies that the nurse often has to choose between two (or more) diagnoses, and finely pick the one the one that most closely describe the patients' situation (Lunney 2003). This is analogous to the way physicians choose diagnoses, (Perreault and Shortliffe 2001).

2.3.1 Terminology

The terms classification, taxonomy and ontology are often used interchangeably and without definition, and often with the same implied meaning.

The definition of *classification* in Merriam-Webster is:

> "systematic arrangement in groups or categories according to established criteria" (Merriam-Webster 2007)

For *taxonomy*, the definition is:

> "orderly classification of plants and animals according to their presumed natural relationships" (Merriam-Webster 2007)

The Web ontology working group defines ontology as

> "machine readable set of definitions that create a taxonomy of classes and subclasses and relationships between them" (WebONT 2003)

According to the above, a distinction between classification and taxonomy is that in a classification, also external criteria, such as time or other contextual properties that can change over time can be used to group items, while in a taxonomy, there must exist a mutual internal property for all the items within a group. Because of the relationship between the items, a taxonomy will also have a structure, or hierarchy, of groups with subgroups. In a classification, the requirement of a hierarchy is not strict.

An ontology can be seen as an extension of a taxonomy, and is typically containing much more information both concerning the structure of the hierarchy and concerning relationship between classes and items (Rees 2002).

The distinctions between these concepts are blurred. All nursing vocabularies used in this paper clearly can be defined as taxonomies. The creators of these vocabularies has laid down a lot of effort to group nursing phenomena with common internal properties into classes, and the classes are again subgroups of the domains. Nevertheless, a lot of literature that deal with these vocabularies and classification in general, use the concept of classification with the same meaning as taxonomy. This is also the case in the initial definition of classification in this chapter by Bowker and Star (Bowker and Star 1999). In this thesis we will therefore use classification and taxonomy interchangeable, both having the same definition as taxonomy in Merriam-Webster above.

2.4 Standardized nursing languages

The most common nursing languages meant for general nursing, e. g. covering all sub-specialties of nursing are the North American Nursing Diagnosis Association (NANDA) (NANDA 2003), the Nursing Interventions Classification (NIC) (Bulechek et al. 2006) , the Nursing Outcome Classification (NOC) (Ladwig and Ackley 2006), International Classification of Nursing Practice (ICPN) (ICN 2007) and SABACLASS (earlier Home Health Care Classification) (NSF 2007). NANDA, NIC and NOC are probably the most widespread languages to day (DIPS 2007), and are closer described in chapter 2.5. The development of ICPN is an initiative of the International Council of Nurses (ICN 2007). The language is a compositional terminology consisting of 7 axes, and is not yet considered mature enough to be used in clinical practice (Glomsås 2003). The other languages are all enumerative, listing all possible diagnoses, interventions and outcomes in advance.

Standardized nursing languages may benefit nursing practice, patients, organizations, and policy makers in a number of ways (DUKE 2007, Bulechek et al. 2006, NSF 2007):

- provides terms that convey the domain of nursing, and reflects knowledge and skills essential to nursing
- makes it easier for nurses to document their work, with consistent terms for diagnoses, interventions and outcomes
- consistent terms will help in developing electronic health records that contain tools for quality assurance, remainders and alerts, and that are capable of presenting patient data in different formats, such as problem oriented or task specific views

- facilitates research on and evaluation of nursing practice, supporting the use of evidence-based practice
- easier to compare nursing practice at different locations and times; promotes auditing, cost – benefit analyses and quality improvement
- provides an efficient and consistent documentation, which lead to more effective care, both within a unit and between different units, securing continuity of care
- provides a more efficient communication between nurses, and also between nurses and other health personnel
- help management to plan resource allocation
- provides statistical material that help politicians to plan future health services

2.5 NANDA, NIC and NOC

The three classification systems NANDA, NIC, and NOC, are presently in the process of being merged into one classification. NADA, NIC and NOC have been developed independent of each other by different organizations, and one obstacle in merging the classifications is that their taxonomies differ(Jones and Dochterman 2003). All 3 classifications are still evolving, and new revisions are published approximately every second year.

2.5.1 NANDA

NANDA is an acronym for the North American Nursing Diagnosis Association, which since 1973 has developed and published nursing diagnoses. The acronym has gradually become the name of the classification itself. The classification is an enumerated list of diagnoses. In the Norwegian translation there are 155 diagnoses (NANDA 2003). The diagnoses can be divided into three different groups with different targets; the individual, the family, or the community (Krogh and Dale 2007). The focus of the diagnoses may also be divided in three; one that is concerning the target's actual problems, one that concerns problems that the target may risk to develop, and one that concerns areas that it is possible to improve or strengthen. The 155 diagnostic concepts are divided into 13 domains and 47 classes. An example diagnosis can be found in figure 2-1. As can be seen in figure 2-1, the diagnosis consists of a label, defining characteristics, and related factors. In diagnoses targeting risks,

Label: Deficient Fluid volume

Definition: Decreased intravascular, interstitial, and/or intracellular fluid (refers to dehydration, water loss alone without change in sodium level)

Defining Characteristics: Decreased urine output; increased urine concentration; weakness; sudden weight loss (except in third-spacing); decreased venous filling; increased body temperature; decreased pulse volume/pressure; change in mental state; elevated hematocrit; decreased skin/ tongue turgor; dry skin/mucous membranes; thirst; increased pulse rate; decreased blood pressure

Related Factors: Active fluid volume loss; failure of regulatory mechanisms

Figure 2-1 Example NANDA diagnosis (DUKE 2007)

the related factors field is substituted with a risk factors field. All the text in the diagnosis is part of the standardized language.

When a diagnosis is placed into a care plan, it is typically annotated with text that detail and individualize the diagnosis related to the patient. The diagnosis label is embellished with etiology, either as free text or picked from the related factors of risk factors in the vocabulary, and sometimes additionally with a symptoms phrase, either as free text of picked from the definitions in the vocabulary. In nursing terminology, this is called PE(S), Problem, Etiology and Symptom (Ladwig and Ackley 2006).

2.5.2 NIC

NIC was developed at the University of Iowa. It is also list based and consists of 7 domains and 30 classes. In the Norwegian translation there are 514 interventions. Each intervention is based on research and developed inductively based on existing practice (Bulechek et al. 2006). An example intervention with activities is shown in figure 2-2. Only the label and definition

Fluid Monitoring 4130

DEFINITION : Collection and analysis of patient data to regulate fluid balance

ACTIVITIES :

- Determine history of amount and type of fluid intake and elimination habits
- Determine possible risk factors for fluid imbalance (e.g., hyperthermia, diuretic therapy, renal pathologies, cardiac failure, diaphoresis, liver dysfunction, strenuous exercise, heat exposure, infection, postoperative state, polyuria, vomiting, and diarrhea)
- Monitor weight
- Monitor intake and output
- Monitor serum and urine electrolyte values, as appropriate
- Monitor serum albumin and total protein levels
- Monitor serum and urine osmolality levels
- Monitor BP, heart rate, and respiratory status
- Monitor orthostatic blood pressure and change in cardiac rhythm, as appropriate
- Monitor invasive hemodynamic parameters, as appropriate
- Keep an accurate record of intake and output
- Monitor mucous membranes, skin turgor, and thirst
- Monitor color, quantity, and specific gravity of urine
- Monitor for distended neck veins, crackles in the lungs, peripheral edema, and weight gain
- Monitor venous access device, as appropriate
- Monitor for signs and symptoms of ascites
- Note presence or absence of vertigo on rising
- Administer fluids, as appropriate
- Restrict and allocate fluid intake, as appropriate
- Maintain prescribed intravenous flow rate
- Administer pharmacological agents to increase urinary output, as appropriate
- Administer dialysis, as appropriate, noting patient response

Figure 2-2 Example NIC Intervention, from (UI 2007)

are part of the standardized language. The activities listed are only suggestions; the nurse is free to use these and/or create her own activities (ibid).

Just like NANDA, but in a different purpose, NIC has three flavors of interventions. Originally, the developers of NIC concentrated on direct patient care interventions. Gradually, more indirect patient interventions like “Emergency Cart Checking” was added, and lately plain administrative interventions are added. The expansion of the classification to include especially the administrative interventions has been disputed, as the classification no longer reflects work that is specific for nurses (Bowker et al. 2001).

2.5.3 NOC

NOC is also developed at the University of Iowa, independent of NIC, but some of the researchers in the NIC team also participated in the NOC team. It has 7 domains, 29 outcome classes and 260 outcomes. Each outcome has a varying numbers of indicators which are given a rating from 1 to 5 on a Likert-type rating scale. The patient is rated according to his status on a given indicator. A score of 5 is given if the indicator is not compromised (healthy), while a score of 1 indicate compromised (sick). (Ladwig and Ackley 2006)

2.5.4 NNN

Representatives from all three classifications, together with other nursing and classification specialists meet regularly in an effort to develop a joint taxonomy. The proposed taxonomy from 2004 can be found in appendix 1. The intention is that over time the three languages shall be published together using this proposed joint taxonomy or a modification of it. For the time being, each organization will place their diagnoses, interventions and outcomes both in their original taxonomy and in the joint taxonomy in their forthcoming publications (Jones and Dochterman 2003).

2.6 Nursing Documentation Frameworks

There exists a number of nursing documentation models. Here we will only briefly present two systems that are relevant to this thesis.

2.7 VIPS

The VIPS model was developed in Sweden during the 1990ties. It is a not a classification system, but rather a framework for nursing documentation (Ehnfors et al. 2000). It helps the nurse to increase the quality of treatment by ensuring that all the aspects of nursing are assessed during anamnesis, se figure 2-3. The acronym is formed by the words Well-being, Integrity, Prophylaxis and Safety (Velvære, Integritet, Profylakse and Sikkerhet). For each of the nursing documentation parts anamnesis, patient status, diagnoses, interventions and patient’s results there are given a set of key words. The key-words for diagnosing are shown in the red rectangle in figure 2-3.

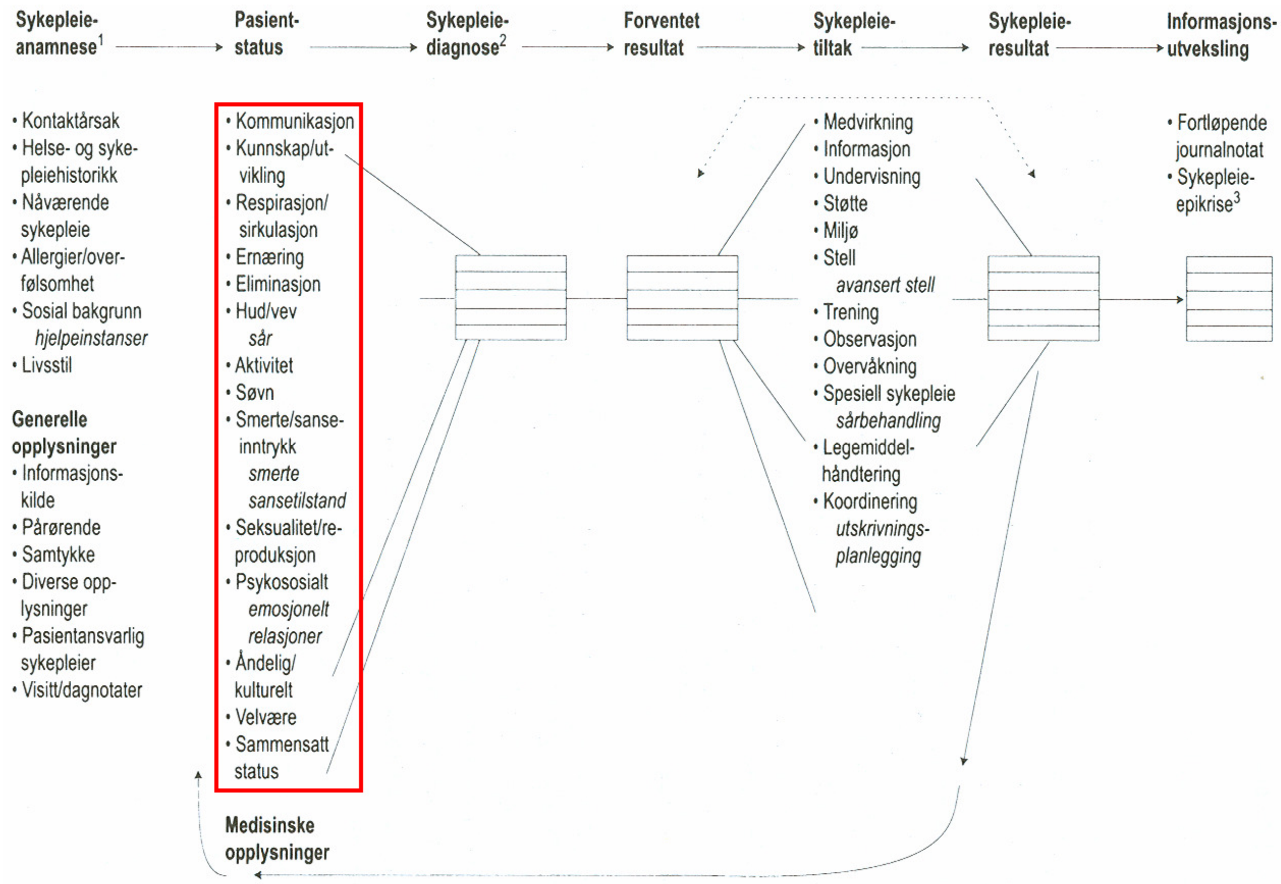

Figure 2-3 The VIPS model (Moen et al. 2002). Key words or functional areas are shown in the red rectangle.

Each of these key-words has its own subset of keywords or pointers, guiding the nurse as the diagnoses are written (KITH 2003). The diagnoses, interventions and results are written in free text. VIPS is the nursing documentation system used by all the university hospitals in Norway (NSF 2007). The functional areas shown in the red rectangle in figure 2-3 are used by DIPS, slightly modified, to categorize NANDA diagnoses and NIC interventions (DIPS 2007).

2.8 The KPO-model

The KPO-model is developed in Norway at the University College of Nursing in Oslo (Krogh and Dale 2007, Krogh et al. 2005). The words of the acronym KPO are Quality assurance; Problem solving and Care (Kvlitetssikring, Problemløsning and Omsorg). The model consists of a conceptual framework for nursing practice, a decision support system and an information management system. The conceptual framework was developed with a harmonization of the classifications of the NANDA, NIC and NOC as a basis. The taxonomy is shown in figure 2-4. The concepts in the framework

KPO-modellen – begrepsmessig rammeverk

1. Domene Funksjonell	2. Domene Fysiologisk	3. Domene Psykokognitiv	4. Domene Sikkerhet	5. Domene Eksistensiell	6. Domene Livsstil	7. Domene Familie	8. Domene Omgivelser
Domene for livsprosesser, essensielle funksjoner og ferdigheter	Domene for biofysiske strukturer og prosesser	Domene for mentale prosesser og mønstre	Domene for helsetrusler og kontekstuelle betingelser	Domene for opplevelser, erfaringer og livsanskuelser	Domene for helsekunnskap og helseoverbevisninger	Domene for reproduksjon, omsorgsevne og familierelasjoner	Domene for samfunn og helsesystem
Klasser:	**Klasser:**	**Klasser:**	**Klasser:**	**Klasser:**	**Klasser:**	**Klasser:**	**Klasser:**
Vekst/utvikling	Sirkulasjon	Kognitiv	Personlige trusler	Velvære	Helsekunnskap	Familieforøkelse	Samfunnshelse
Bevegelse	Respirasjon	Adaptiv	Helsetrusler	Mestring	Helseadferd	Omsorgsgiver	Helsevesen
Energi	Ernæring	Stemningsleie	Konteksttrusler			Familieenhet	
Egenomsorg	Eliminasjon						
Kommunikasjon	Hud/vev						
Sosial funksjon	Immunologisk respons						
	Nevrologisk respons						
	Farmakologisk balanse						

Figure 2-4 The taxonomy of the KPO – model (Krogh and Dale 2007).

reflect the domain of nursing practice, and defines what the nursing service sees as their special area as opposed to other health services (Krogh and Dale 2007). A starting point of the authors of this model is what they claim is deficiencies in the NNN taxonomies and VIPS framework. Conceptual frameworks have to be logical and consistent in order to promote knowledge and to identify gaps in current knowledge, and also small enough to be integrated by nurses as a cognitive map that can be used as a tool in clinical work (Krogh et al. 2005).

The decision support system consists of 5 guides that act as quality assurance tools; 1 for patient assessment, 1 for diagnosing, 2 for finding interventions and 1 for finding outcomes, see figure 2-5. It also contains planning templates related to ICD-10 diagnoses, the phase of different illnesses and for care pathways. All of these guides and templates relates to the conceptual framework and NANDA, NIC and NOC (Krogh and Dale 2007).

KPO-modellen – beslutningsstøttesystem

Pasientdomenet	Intervensjonsdomenet	Omgivelsesdomenet
Begrepsmessig rammeverk • Funksjonell • Fysiologisk • Psykokognitiv • Sikkerhet • Eksistensiell • Livsstil • Familie	**Begrepsmessig rammeverk** • Funksjonell • Fysiologisk • Psykokognitiv • Sikkerhet • Eksistensiell • Livsstil • Familie	**Begrepsmessig rammeverk** • Omgivelser - Samfunnshelse - Helsevesen • *Pasientforløp* • *Samarbeids- og informasjonshåndtering* • *Helseservice*
Guider • Kartleggingsguide • Diagnoseguide • Resultatguide	**Guide** • Intervensjonsguide	**Guide** • Helsesystemintervensjonsguide
	Planleggingsmaler • ICD-10 – planleggingsmaler • Sykdomsfase – planleggingsmaler • Sykdomsforløp – planleggingsmaler	

Figure 2-5 The decision-support system of the KPO-model (Krogh and Dale 2007)

The information management system consists of 4 structures; a patient status, a work plan, an outcome record and an information exchange structure, as seen in figure 2-6. These structures are the starting point for the user interface in an EHR. There is a clear connection between the guides and the 3 first information structures, the assessment guide results in the patient status note, and the other guides assist in working out the work plan and the outcome record. There is also a linear connection between the patient status notes and the work plan; the initial work plan is based on the first patient status note, then consecutive changes in the work plan is given reason for in short patient status notes (ibid).

KPO-modellen – informasjonshåndteringssystem

1. Informasjonsstruktur Pasientstatus	2. Informasjonsstruktur Arbeidsplan	3. Informasjonsstruktur Resultatmåling	4. Informasjonsstruktur Informasjonsutveksling
Notater Pasientstatusnotater *Psykokognitiv statusnotat* *Funksjonell statusnotat* *Sikkerhetsstatusnotat* *Fysiologisk statusnotat* *Eksistensiell statusnotat* *Livsstilstatusnotat* *Familiestatusnotat* Ønsket status/ NOC	Sykepleiediagnoser/NANDA Sykepleieintervensjoner/NIC *Intervensjonsregistrering*	Sykepleieresultater/NOC *Pasientspesifikt resultat* *Pasientkategoriresultat*	**Notater** Forløpsnotater *Innkomstnotat* *Sammendragsnotater* *Møtereferater* *Overflyttingsnotater* *Utskrivingsnotater* Hendelsesnotater *Miljønotater* *Telefonnotater* *Hjemmebesøksnotater* *Polikliniske notater*
Tekstformat Strukturert fritekst i notater Standarder for ønsket status	**Tekstformat** Standarder for diagnoser og intervensjoner Fritekst for spesifikasjoner og aktiviteter	**Tekstformat** Standarder for resultater Standarder for indikatorer	**Tekstformat** Narrativ tekst i notater

Figure 2-6 The information management system of the KPO – model (Krogh and Dale 2007). The red rectangle encloses the work plan, which corresponds closely to the care plan discussed in this thesis.

The taxonomy, or conceptual framework, of the KPO – model is used in this paper as the classification system for browsing NANDA diagnoses and NIC interventions. The solution for our care plan that is presented later in this paper resembles the work plan in the KPO-model, shown in the red rectangle in figure 2-6.

2.9 State-of-the-Art Care Plan

We have searched the literature for articles about electronic nursing care plans that incorporate standardized nursing vocabularies. There are very few articles on the subject, and most of them are written by nurses and is about experiences drawn from using one, both regarding better patient documentation and improved professional consciousness, but say little about the user interfaces and usability, and the support for searching and browsing the classifications. Only one of the articles found is discussing the process of developing a user interface (Keenan et al. 2002). A lot of commercial EHR vendors, especially from the USA, offer care plan writing and standardized nursing languages in their software. It is though impossible to evaluate these just in the light of the information on their web sites. We will therefore in this section present two Norwegian EHRs, and then present the article of Keenan et al. (Keenan et al. 2002). Two solutions that are intended for pedagogical use are also presented.

2.9.1 DIPS

DIPS is one of the major vendors of EHRs in Norway (DIPS 2007). In DIPS, it is possible to create a group of documents that relates to nursing documentation. In the group, different document types like shift reports, summaries, and so on can be defined. A special document type called the “treatment plan” can be used within this group. This is a document where the care plan and the shift report are merged into one. The name of the document type is deliberately chosen, the idea is that this document type also can be used by different health professions, not only nurses, to facilitate multidisciplinary treatment and documentation (ibid). In this document type, the patient’s care plan is opened in the lower part of the window, ready for editing, every time a new shift report is to be written. This promotes a more active use of the care plan the nurses traditionally has been used to. It may for instance be much faster and more instructive to create an intervention with activities in the care plan when the patient’s status changes, than to explain this in free-text in the shift report. Additionally, the presentation of the care plan makes it easier to relate the shift report directly to the defined focus in the care plan than if the plan had to actively be opened.

The treatment plan document type in DIPS has support for NANDA and NIC. It is possible to select the labels of diagnoses and interventions from lists. The help system has all the information related to the diagnoses, presenting the totally NANDA vocabulary. For the interventions, there is only a list of the labels. NIC definitions and activities have to be looked up elsewhere. The diagnoses and interventions are categorized under the function areas of VIPS[1], see figure 2-3 (DIPS 2007). This categorization without an intermediate level of classes is very coarse; there is in average more than 13 diagnosis and more than 40 interventions in every functional area. Many of the diagnosis and interventions are also placed under more than one functional area, violating even the weakest definition of classification in chapter 2.4.1. This “double bookkeeping” can now and again be confusing to the users.

2.9.2 DocuLive

DocuLive (Siemens) is also one of the major EHR vendors in Norway, serving approximately half of the hospitals. DocuLive has a number of predefined nursing document types, all with standard templates. A special document type for care plans does not exist, but document types for problems, interventions, outcomes and notes can be associated with each other in different ways, one to one, one to many and many to many, so that the connection both between the underlying data and their visual presentation becomes clear. A patient problem can for instance be presented at the top of a window, with the associated interventions and outcomes below. It is possible to connect the nursing module to different standardized nursing classification systems (Siemens 2007). According to Siemens, no hospital in Norway has used this possibility yet (Halvorsen 2007). The KPO-model has been tested at the Ullevål Univerity Hospital using DocuLive. In this solution the nurses have had type in the diagnoses, interventions and outcomes manually. Information of the classification has been accessible at the hospital’s intranet (Vedal 2006).

[1] DIPS has chosen to merge several of the 14 functional areas of VIPS, ending up with a total of 12.

2.9.3 The HANDS Project

This is the he only project that we have found that was dedicated to develop a user interface to enter standardized nursing vocabulary (Keenan et al. 2002). The goal of the project was to make a basic clinical nursing data set that could be shared between different settings. The sharing of the data should be achieved by making a web application where nurses in different organizations could enter clinical data, which then was merged and aggregated in a centralized database. A prototype of the application was built, using Access forms. The prototype supported documentation of NANDA diagnosis and NIC interventions at label level, and NOC outcomes at label and indicator level. It was possible to search each terminology either alphabetically or by keywords from the labels, or hierarchically by the taxonomies. Figure 2-7 shows that a hierarchical search is accomplished by selecting the actual language, and then selecting domain and class from two dropdown list boxes. By highlighting a label and pressing an “INFO” – button, additional information like the definition became available. In addition, subsets of the diagnoses like the

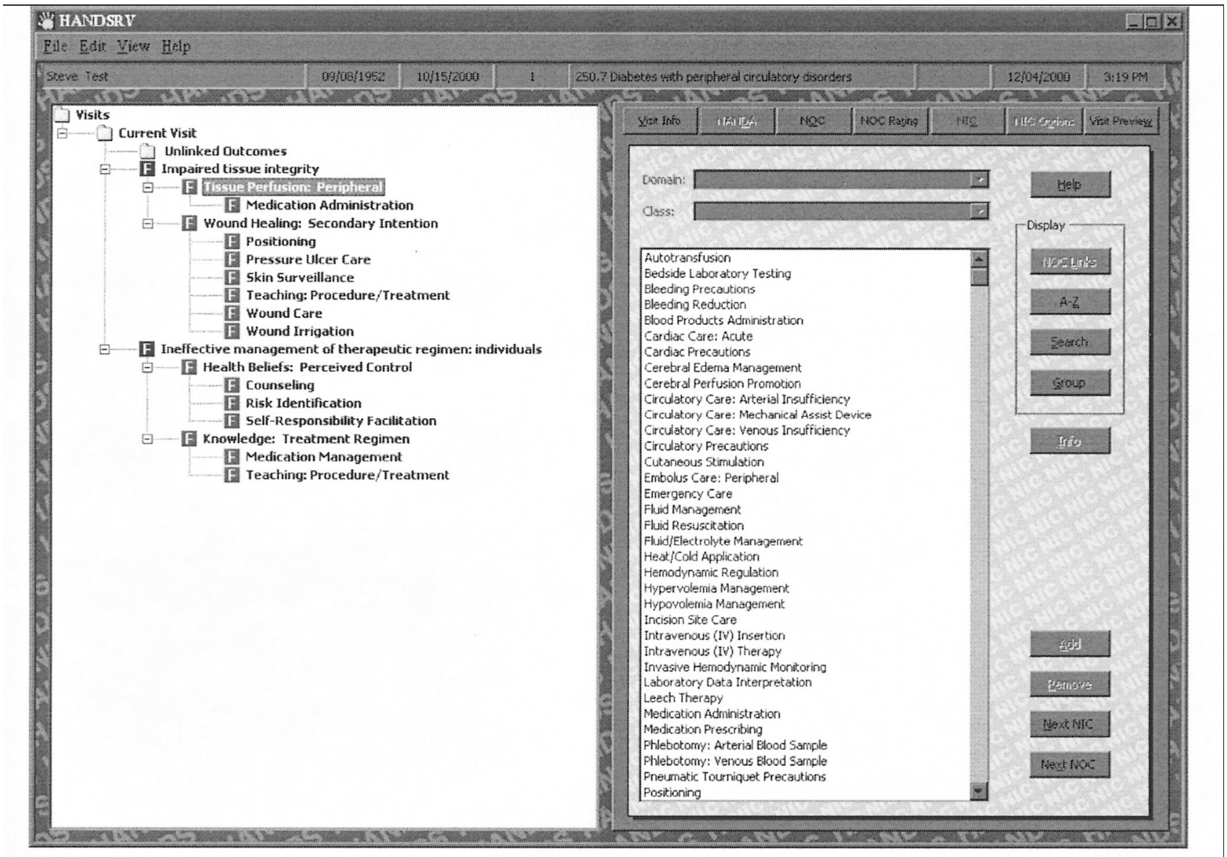

Figure 2-7 Hands tool Nursing Interventions Classification selections screen (Keenan et al. 2002).

most used NANDA diagnosis for a specific unit (intensive care, geriatric unit etc.), NOC outcomes associated with each NANDA diagnose and NIC interventions associated with each NOC outcome could be selected (ibid). Even not specifically stated, it is likely that the originally taxonomy for languages is used, so that the users had to relate to 3 different hierarchies at the same time.

The HAND – tool was used to gather data for scientific purposes, so it may not be comparable to writing nursing care plans. It was used by nursing consultants who used it intensively and every day, even so it is indicated that it took more than 2 weeks before the user felt they mastered it fluently.(Keenan et al. 2002)

2.9.4 Sanum

Sanum is a pedagogical tool for teaching NANDA, NIC and NOC. It is possible to find a diagnosis by browsing down the 13 domains and 47 classes of the original NANDA taxonomy. Alternatively a diagnosis can be found by key-word search of the labels. Definitions and other information of the diagnoses are readily availably at all levels of the process. When a diagnosis is found, an intervention can be selected from a list of suggested interventions, that is, the most common interventions for that diagnosis (Sanum 2007).

2.9.5 Evolve Care Plan Constructor

This is a care plan constructor that is meant as a tool to practice care plan writing based on the Nursing Diagnosis Handbook (Ladwig and Ackley 2006). It is possible to select NANDA diagnoses related to ICD-10 diagnoses. After a diagnosis is selected, associated options of NOC outcomes and NIC interventions are presented and can be selected. Information and descriptions of the care plan items are abundantly present, to a degree that make the care plan difficult to read (Evolve 2007).

2.10 Information Retrieval

In the context of this thesis, information retrieval is synonymous to document retrieval. A document retrieval system has mainly two tasks. The first is to find documents on the basis of a user query, and the second is to sort them according to relevance (Wikipedia 2007).

2.10.1 Basic concepts

In an information retrieval (IR) system, the total numbers of documents are called the collection. Each document in a collection is containing a set of keywords that is called *index* terms. Some information retrieval systems, as for instance Web search engines, use all the distinct words in the document collection as index terms. Other systems may use subsets of the words as index terms in order to speed up the search and to decrease the size of the indexes. A common approach is to eliminate common words (called stop words) like "a, and, he, is", etc.

Not all index terms are equally useful. The deciding of the importance of a term is not a simple task. This can be done manually, or automatically by for instance letting words appearing in headlines be more important than others. A term that is found in most of the documents contribute little in selecting relevant documents, while a term that only exists in a few documents will contribute a lot (Baeza-Yates and Ribeiro-Neto 1999). This is utilized in statistical methods where a numerical *weight* is assigned to each index term. An index term that is found in few documents will be assigned a higher weight than those found in many documents. For a system with t numbers of index terms, an index term vector $\vec{d}_j = (w_{1,j}, w_{2,j}, ..., w_{t,j})$ can be made for each document j, where $w_{i,j}$ denotes the weight of the i-th term in document j. For an index term that is not present in the document, $w_{i,j} = 0$ (Manning et al. 2007). The index term vector should be viewed as a conceptual construct; it is rarely or never used in practice (Lee et al. 1997).

A simplification often done, and that is done in this thesis, is that index term weights are considered mutually independent. To illustrate that this not always is the case, consider the nursing diagnosis Anxiety. Here the words anxiety and fear appear several times, and these two words are clearly correlated. However, assuming mutually independence simplify computation of term weights and allow fast ranking computations, but does not substantially degrade ranking. (Baeza-Yates and Ribeiro-Neto 1999)

2.10.2 Retrieval Performance Evaluation

The performance of an information retrieval model is often measured by how precisely the retrieved documents reflects the query. These measures are performed against test collections in laboratories. In figure 2-8 |R| is the number of relevant documents in the collection corresponding to a given query. These documents are beforehand chosen by domain specialists. |A| is the number of documents actually retrieved by the given query. |Ra| is the number of documents in the intersection of the two sets (Baeza-Yates and Ribeiro-Neto 1999).

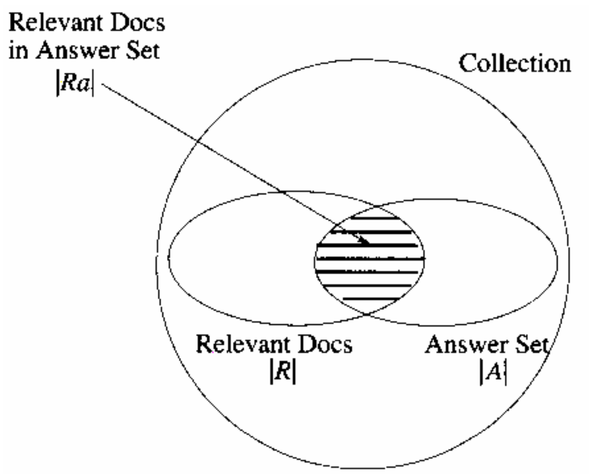

Figure 2-8:Precision and recall for an example information request (Baeza-Yates and Ribeiro-Neto 1999)

Recall is defined as: $Recall = \frac{|Ra|}{|R|}$

Precision is defined as: $Precision = \frac{|Ra|}{|A|}$

Recall is a measure of how many of the relevant documents the actual retrieval model is capable to retrieve, and *precision* is a measure of how many of the retrieved documents correspond to the given query (ibid). The better the retrieval model is, the larger the shaded area in figure 2-8 becomes.

Recall and precision are subjective measures, as different users or domain specialists have different opinions of which documents are relevant and non-relevant.

2.10.3 The Vector Model[2]

In the Vector Model, each index term in documents and queries are given a non-binary weight. Theses weights are used to compute the similarity of documents and queries, and to sort documents in decreasing order according to this similarity. Also documents that do not precisely match the query may be retrieved. The similarity between the document d_j and the query q is computed by constructing a document vector $\vec{d}_j = (w_{1,j}, w_{2,j}, ..., w_{t,j})$ and a query vector $\vec{q} = (w_{1,q}, w_{2,q}, ..., w_{t,q})$ where t is the total number of index terms in the system, and then finding the cosine angle between these two vectors (Baeza-Yates and Ribeiro-Neto 1999):

$$sim(d_j, q) = \cos(\vec{d}_j, \vec{q}) = \frac{\vec{d}_j \bullet \vec{q}}{|\vec{d}_j| \times |\vec{q}|} = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2}\sqrt{\sum_{i=1}^{t} w_{i,q}^2}} \quad \textbf{(Equation 2-1)}$$

Since all weights are greater than zero, $sim(d_j, q)$ varies from 0 to +1. A query where none of the words exists in document j, will result in $sim(d_j, q) = 0$, and for a query with a good match the similarity value will approach 1.

There exist a lot of different ways to calculate term weights among different retrieval models. The vector model uses statistical term weighting. One way of explaining statistical term weighting is to view the information retrieval process as a problem of clustering. The documents in the system are divided into two sets, one containing the relevant documents more or less vaguely described in the query, set A, and one containing irrelevant documents, set B. The *term frequency (the tf factor)* is a measure of intra-cluster similarity in set A; terms that occur frequent in different documents within set A imply a high degree of similarity between these documents. Similarly, the *inverse document frequency* (*idf*, the inverse of the number of documents where the term occur), is used as a measure of inter-cluster dissimilarity between set A and B (Baeza-Yates and Ribeiro-Neto 1999). As stated earlier, terms that occur in many documents are not very

[2] An other and more frequently used name for the Vector Model is the Vector Space Model

useful for distinguishing between documents. Different authors have tried to balance the intra- and inter-clustering effect in different ways. A good weighting schema that performs well for many different types of collections was proposed by Salton and Buckley (Salton and Buckley 1988).

The normalized frequency $f_{i,j}$ of term k_i in document d_j is given by

$$f_{i,j} = \frac{tf_{i,j}}{\max(freq_{l,j})}, \qquad \textbf{(Equation 2-2)}$$

where $\max(freq_{l,j})$ is computed over all terms in the document. The inverse document frequency is

$$idf_i = \log\frac{N}{n_i}, \qquad \textbf{(Equation 2-3)}$$

where N is the total number of documents in the collection and n_i is the number of documents containing the term k_i. Weights are then given by

$$w_{i,j} = f_{i,j} \times \log\frac{N}{n_i} \qquad \textbf{(Equation 2-4)}$$

For the query term weights Salton and Buckley suggest

$$w_{i,q} = \left(0.5 + \frac{0.5tf_{i,q}}{\max(freq_{l,q)}}\right) \times \log\frac{N}{n_i} \qquad \textbf{(Equation 2-5)}$$

2.10.4 Document Preprocessing

To be able to pick index terms, calculate frequencies and give weight to each term, the text in the documents has to be preprocessed. Figure 2-9 shows the different steps that can be included in the processing of a document from full text to different logical views ready for indexing (Baeza-Yates and Ribeiro-Neto 1999).

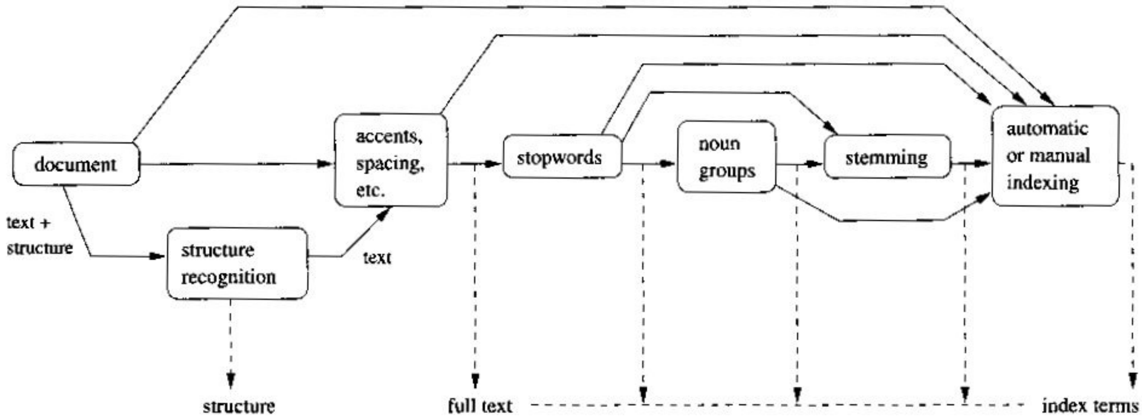

Figure 2-9: Logical view of a document throughout the various phases of text preprocessing (Baeza-Yates and Ribeiro-Neto 1999).

2.10.4.1 Structure recognition

In some context, it can be advantageous to take the structure of the documents into consideration. It may for instance be decided to let some elements of the text be different weighted than the rest (Baeza-Yates and Ribeiro-Neto 1999).

2.10.4.2 Accent, spacing, etc

Lexical analysis is the process of identifying the words in the text, excluding unwanted characters and signs. Hyphens, punctuation marks, digits, and the case of the letters need special considerations because excluding them may in some contexts cause an altering the semantics of words (Baeza-Yates and Ribeiro-Neto 1999).

2.10.4.3 Eliminating Stopwords

Words that occur frequent in many of the documents in a collection can be eliminated, because they don't distinguish well between the documents. Usually, stopword lists consist of the most frequently used word in a language. If the document collection consists of document from a specific domain, words that are frequently used in that domain terminology may also be added to the stopword list. The use of stopwords has also a considerably effect of the size of the indexing structure, more than 40% reduction have been reported.(Baeza-Yates and Ribeiro-Neto 1999)

2.10.4.4 Stemming

Stemming is the process of removal of suffixes and prefixes from a word. Languages have different syntactical variations of words depending of the word's inflection, e.g. plurals, gender, and past tense. For instance will a search for the term "promoting" not find documents containing "promote", promotes", "promoted" etc (Baeza-Yates and Ribeiro-Neto 1999). By removing the affixes from the words and keep only the stem in the indexing structure, retrieval of relevant documents may increase, but so will also the retrieval of non-relevant documents, i.e. recall will increase on expense of precision. This technique will also reduce the size of the indexing structure.

Different strategies for stemming exist; one commonly used is suffix removal by the Porter algorithm (Frakes and Baeza-Yates 1992). The Porter algorithm has to be changed and adopted to the language that is going to be stemmed (Porter 2007).

2.10.4.5 Index Term Selection

The logical view of the document can be further abstracted by not using all of the words as candidates for indices. Indexes can be manually selected, as have been common in the bibliographic sciences. For instance are articles in MEDLINE manually indexed based on the Medical Subject Headings (MeSH) vocabulary (Hersh 2003). Another approach is to automatically select index terms. Most of the semantics in the text is carried by the nouns, and by systematically eliminate other word categories like verbs, adverbs etc. one will end up with only nouns, which then are indexed.

2.10.5 Inverted Index

The index terms that result from the document preprocessing has to be processed further and placed in a data structure to facilitate efficient searching. A common structure used in IR systems is the inverted index, see figure 2-10. The index terms are kept alphabetically in an array

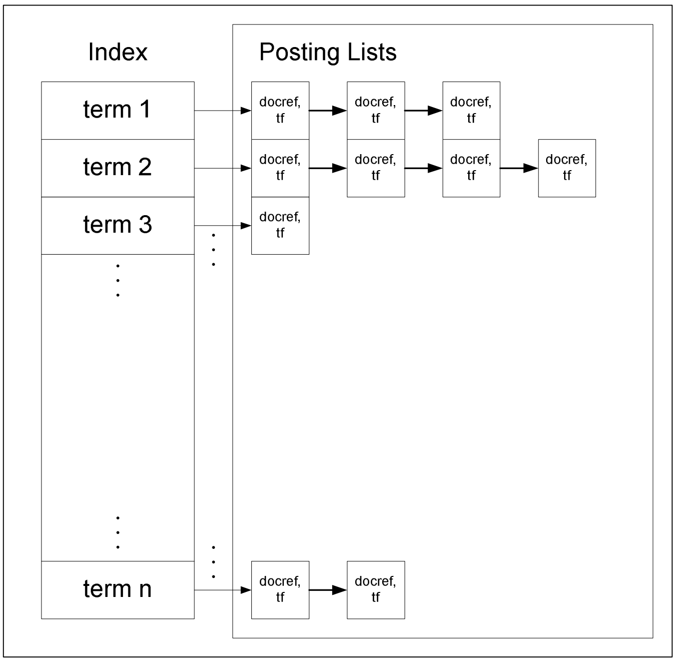

Figure 2-10 Inverted Index. From (Grossman and Frieder 2004).

called Index in figure 2-10. Each term has a pointer to a Posting List, a linked list where each node keeps the address of where the term occurs. The granularity of the address is dependent on the system; it may be just a reference to the document, or a reference to the document and where in the text the term occur(s). In addition each node may keep the value of the term frequency within each document (Hersh 2003).

The Index is often stored in a data structure like a hash table, a trie or B-tree to speed up searching (Baeza-Yates and Ribeiro-Neto 1999). The term "inverted" in inverted index indicate that the text is presented in alphabetical sequence in stead of its original positional sequence.

2.10.6 Thesaurus

A simple method to increase recall is to use an ordinary thesaurus and for each query term also search for its synonyms.

3 Method

3.1 The Unified Process

We have chosen to use the Unified Process (UP) to model our project. The UP describes “best practices” for developing software (IBM 2001):

1. *Develop software iteratively*. Very early in the development process a running implementation is built based on some core requirement. This prototype is then refined based on feedback of future users in a series of iterations. For each iteration, the understanding of the domain is increased, and requirements are adjusted and new one added (Larman 2002). By managing the highest risk items in early iterations, the total risk profile of the project can be reduced (IBM 2001).

- *Manage requirements*. High risk requirements are resolved as early as possible. Use cases are used to capture functional requirements, and to drive design, implementation and testing of the software (ibid). Other artifacts to capture requirements not easily captured in use cases are Supplementary Specification, Vision and Glossary. Supplementary Specification can include FURPS+ requirements – functionality, usability, reliability, performance and supportability. The Vision tries to capture the “big ideas” of the project, and the Glossary may contain unclear and ambiguous terms (Larman 2002).

2. *Use component-based architecture.* Components are sub-systems that fulfill a clear function. Components promote reuse of software (IBM 2001).
3. *Visually model software.* The Unified Modeling Language (UML) is used to conceptualize components, architecture and communication between components (Booch et al. 1999).
4. *Verify software quality.* The UP process contains artifacts that support assessment of requirements such as reliability, functionality, application performance and system performance (IBM 2001).

5. *Control changes to software.* The UP process promotes configuration management, and it is common to use tools that control version changes and allow for separate workspaces and automatic integration and build management (ibid) (Hansen et al. 2003).

The UP presents a framework for the developing process that consists of a large set of optional activities and artifacts. As shown in figure 3-1, it can be organized along two axes. Along the time axis, the process is separated into different phases. Each phase ends with a milestone where certain goals must have been achieved.

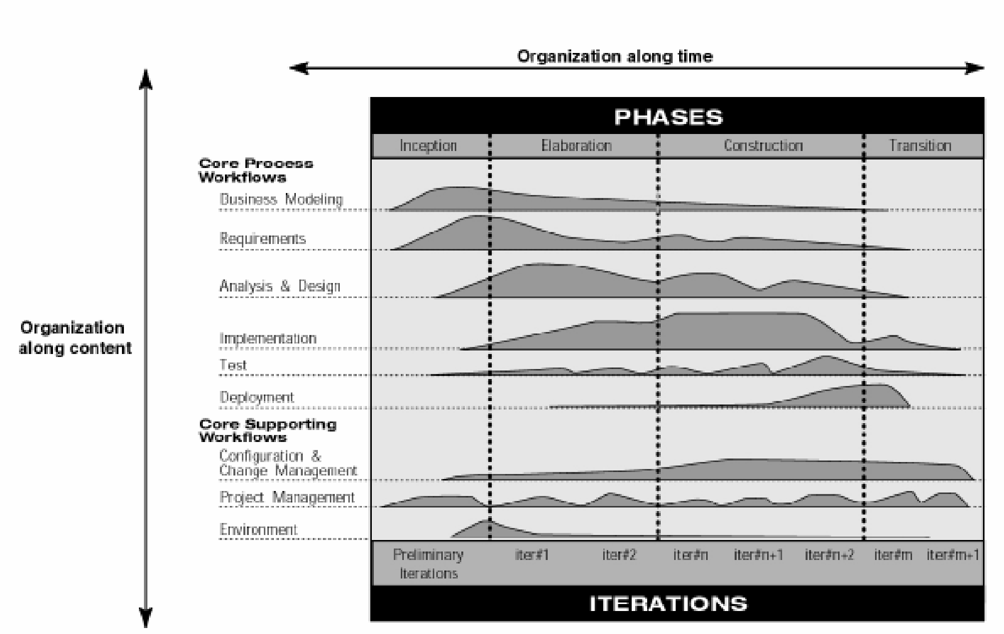

Figure 3-1 The organization of the UP along the 2 dimensions time and content (IBM 2001). [3]

The *Inception* phase includes initial requirements analyses, estimates about resources needed for the project, and a risk assessment. Among the milestone evaluation criteria are cost/gain, risk evaluation, and requirement understanding. The project may be cancelled if it fails to pass this milestone (IBM 2001).

[3] The UP term "workflow" was in 2001 replaced with the new term "discipline" to harmonize with international standardization efforts. Many still use the old notation, although this is not strictly correct.

In the *Elaboration* phase, the gross part of requirements is defined, a core architecture is built, high risk elements are resolved and an overall estimate of resources needed are done. Short risk-driven iterations are done, with runable implementations that are adapted based on realistic tests and feedback from users and developers (Larman 2002). Among the milestone evaluation criteria for the elaboration phase are the stability of the vision and architecture, the degree to which the risk factors are resolved, and the agreement among the stakeholders that the current vision can be achieved. If the project fails to meet this milestone, it may be aborted or may be largely reconsidered (IBM 2001).

The *Construction* phase consists mostly of implementation, in addition to continuing requirement definition, analysis and design. The amount of testing is increasing, and a series of iterations is gradually bringing the "skeleton" prototype from the elaboration phase to a fully functional "beta" version. User documentation is also written (Andreassen et al. 2003). Milestones criteria after this phase are the stability of the release product, user's readiness to start transition, and resource expenditures. The transition may be postponed if the project fails to meet the milestones criteria (IBM 2001).

In the *Transition* phase, activities required to transfer the software to its final environment is emphasized. Focus is on testing, management and formal document writing (ibid).

Along the content axis in figure 3-1, the UP defines different disciplines (denoted workflows in the figure).

The main focus in *Business Modeling* is to create and maintain the similarity between the business model and the software model(IBM 2001). Often the process of software development may lead to discovery of better business models, and these changes are managed in this discipline. In many projects business modeling is not of interest.

In the *Requirements* discipline, the main task is to describe what the system should do, and let the future users and developers agree of that description. Candidate requirements and functional- and non-functional requirements are gathered, and a *vision* document is created. The vision document summarize the big ideas of the project; why it was proposed, what the problems are, who the stakeholders are, and what the proposed solution should look like (Larman 2002). *Actors* and *use cases* are identified, and the use cases are described in detail. The use cases are a unifying thread throughout the development face. Non functional requirements are described in the *Supplementary Specifications(IBM 2001).*

The *Analysis & Design* discipline results in a design model that is an abstraction of the source code, like a blueprint of how the code are structured. The emphasis is on how the system will be realized in the implementation phase. The design model consists of design classes that are placed into packages, each with a well defined interface, and the collaboration between packages is described. Most of the activities are centered on the description of the architecture of the system, with simplified abstractions of the design, where details are omitted to gain an overall view of the system (Larman 2002). Sample artifacts used in this discipline are conceptual class diagrams, sequence diagrams, use-case models, and software class diagrams (Larman 2002).

The purpose of the *Implementation* discipline is to implement the system into components. Subsystems are organized into layers, and individual components tested and built. These components are then organized to comprise the complete system (ibid).

During the *Test* discipline proper interaction between objects and proper integration of components is verified. It should also be verifed that all the requirements are correctly implemented. Testing is done throughout the whole project, but is especially emphasized at the end of the construction phase (ibid).

In the *Deployment* discipline, a final release product is produced, and delivered to the end users (ibid).

The attention and emphasis of the different disciplines varies throughout the different phases. This is illustrated by the height of the shaded area under the graph for each discipline in figure 3-1. There will also be substantially variations of which disciplines are most emphasized and the length of the phases between different types of projects.

The number of artifacts and activities described in the UP are huge. It is important to realize that most of these are optional, and that the process should be customized. Principles like iterative and risk-driven development and continuous verification of quality are unalterable. However, when it comes to artifacts, only those that address particular problems and needs should be selected. A small set of artifacts that are illustrative and gives insight into the problem at hand is better than an excessive use of artifacts and documentation (Larman 2002).

3.2 Use-Cases

End users of software have goals or needs that they want the software to fulfill. When developing software the developers and the future users have to cooperate and speak the "same language" to explore and capture these goals. The simpler this language is the less is the risk to miss essential goals. Use cases was introduced in 1986 by Ivar Jacobsen, and is a means to accomplish this mutual language (Jacobson 1992). They are stories of using the system to a meet a specific goal. The users of the system are not limited to humans; it is everything with a behavior, such as other computer systems, organizations or persons. The users are called actors. A sequence of interactions between an actor and the system is called a scenario and represents a path through the use case. A scenario describes actors using the system to achieve a goal, and depending on if the goal is reached or not, the scenarios are success or failure scenarios. These scenarios then constitute the use case (Larman 2002).

The use cases may be written in different formats, depending of the need of details for the present project. The brief format is a terse one paragraph summary, usually consisting of only one scenario. The casual format may include several scenarios, while the fully dressed format include all thinkable details describing every step while trying to achieve a goal, with preconditions and departure or alternate scenarios due to conditions met under way (Cockburn 1999).

When writing use cases, it is recommended to investigate user goals, rather then tasks and procedures. In this way the focus is lifted to *what* the software should achieve on behalf of the user rather than the trivial *how* it is going to achieve this. The *essential writing style* avoids User Interface (UI) details and focus on intent. In this way, new ways of performing or automating tasks may be discovered (Larman 2002). Another concept describing the same focus on "what" versus "how" is *black-box use cases*, where the system is said to have responsibilities, and where the internal workings of the system and how it will fulfill its responsibilities is not described

(Cockburn 1999). An important exception of the recommended black-box writing style is when developing user interfaces (Larman 2002).

Identifying goals, actors and use cases is an iterative process with ongoing personal communication with stakeholders and future users. Goals often are defined on different levels, as for instance business goals, enterprise goals, user goals, goals for user tasks, and so on. It is important not to be to detailed and define many goals at low levels (Larman 2002).

It is fruitful to divide actors into categories. Primary actors are users that reach their goals by using the system. By identifying these, goals and use cases are found. Supporting actors provides service to the system, like a system administrator or other systems. Identifying these clarify external interfaces. Offstage actors have an interest in the use case, like government or health authorities. It is important to identify these such that all stakeholders and interests are identified and satisfied (ibid).

Use cases are not object-oriented in them selves, but are a central input to classic Object Oriented Analysis and Design activities (Larman 2002).

The use case model is a model of the system's functionality and environment. Use cases can be viewed as functional requirements indicating what the system will do (Booch et al. 1999).

3.3 Goal-directed design

In his book "The Inmates are Running the Asylum"(Cooper 1999), Cooper deal with the software industry, which he claims has failed in creating user friendly products. The "Inmates" in the title refers to the software programmers, which he calls "Homo Logicus". Homo Logicus trade simplicity for control, focus on understanding how thing works, and goes for what is possible in stead of what is probable. Software products have the last decennia reached a broader public, and the technology regarding user interface has improved immensely. Yet, it is still the programmers that have the final saying in user interface design, forcing their logical and complex thinking on users that want simplicity and ease of use (ibid). Cooper stresses the importance of designing before programming, contrasting usability testing after programming. Interaction design is based on thorough understanding of users and cognitive processes. Achieving user goals is the primary aim, forms and esthetics come secondly (Cooper and Reimann 2003).

Cooper has created his own framework for software design (Cooper and Reimann 2003), see figure 3-2. We feel that some of his concepts are useful as supplements to the UP and as a mean to better understand the interaction between software and users. In the following these concepts are presented in isolation, as we refrain from presenting the whole theoretical framework.

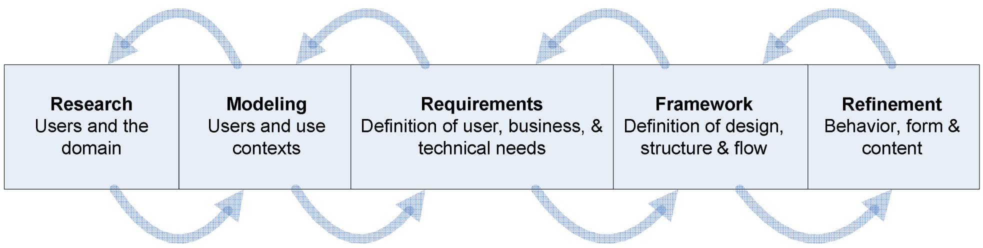

Figure 3-2: The Goal - Directed Design process. From (Cooper and Reimann 2003)

3.3.1 Personas

A persona is a user archetype, not a real person but yet personified in great detail with behavior, goals and motivations, and with names and a picture, to help the designers and developers to get a deeper grasp of their "inner emotions". Personas populate *scenarios* that are used in the later requirements definition phase. 6 different persona types are defined, of which the primary- and secondary persona are the most important. A primary persona needs a distinct interface form and behavior. A secondary persona can use the same interface as the primary persona with only minor modifications(Cooper 1999, Cooper and Reimann 2003). Personas can be compared to the UP's actors, but are more detailed and are the result of a much more involved process, where ethnographical field study techniques are employed.

3.3.2 User Goals

End goals are the goals that the user expects to achieve as outcome while using the software, analogous to the goals discussed in connection with use cases. Beyond these, it is important to take into account what users want to *feel* while using the product, the *experience goals (Cooper and Reimann 2003)*:

- Don’t feel stupid
- Don’t make mistakes
- Feel competent and confident
- Have fun (or at least not be too bored)

3.3.3 Beginners, Experts and Intermediates

The gross amount of users of a software product is intermediates. Everyone start as beginners, but sooner or later most slide into the intermediate category; those who does not quit using the product. A few users become experts. An important aspect is that users may change experience level quite rapidly. Experts have to maintain their knowledge by using the software frequently, or they will go back to the intermediate category. The same is true for intermediates; infrequent use may cause a drop down to the beginner level. The overall tendency is that users gravitate towards intermediacy (Cooper and Reimann 2003). Most software products neglect this fact and are designed for beginners and experts, instead of being optimized for intermediates (ibid).

3.3.4 Mental Models versus Implementation Models

According to cognitive psychologists, a mental model is an internal representation of an external reality. It is built on the fly and used to make decisions when a person has to cope with new experiences. Among human computer interaction (HCI) practitioners, it has a slightly different meaning, describing a person’s conception of how the software works (Davidson et al. 1999).
The implementation model can be defined to be the actual way the program is working from the programmer’s point of view.
To better the usability of the system, the designer’s task is to make an interface and interaction, or a representational model, which resemble the user’s mental model as much as possible, and hide the inner workings of the system.
Cooper claims that most software today is designed according to the implementation model, because it is designed by engineers (Cooper and Reimann 2003).

3.4 Usability

3.4.1 Usability attributes

Usability is often measured along multiple axes, which each describes different properties of the user interface (Nielsen 1993), se figure 3-3. The *learnability* attribute indicates how easy it is to

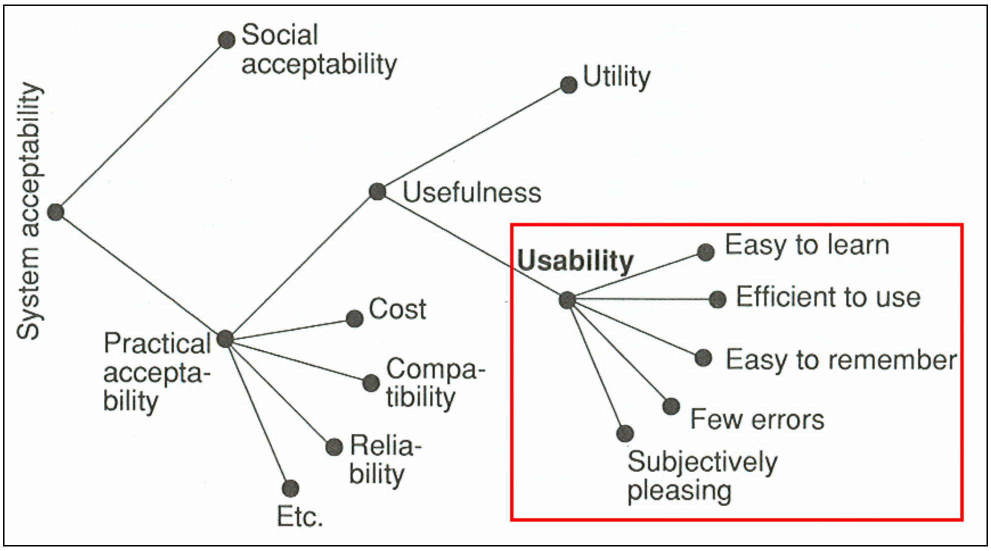

Figure 3-3 The attributes of system acceptability; the usability attributes are shown in the red rectangle, from (Nielsen 1993).

learn the system and become productive using it. Once learned, the system should be efficient to use so that a high level of productivity is reached, this is measured through the *efficiency* attribute. The *memorability* attribute indicates how easy it is for casual users to use the system after a period without using it. The *error* rate of the system should of cause be low, and recovery from errors should be easy. The *satisfaction* attribute says something about how pleased and satisfied the user is while using the system. Each of the above attributes is more precise and measurable than the composite concept "usability", and they are therefore the basis for usability testing (ibid).

It is not always possible (or desirable) to reach a high level for all of these attributes for a given system, for instance can learnability conflict with efficiency. The degree of which of these attributes is most important will also depend on the system under consideration; a pilot would probably be more concerned about efficiency and a low error rate than learnability and memorability when using a navigational system (Cooper 1999).

3.4.2 User Experience

Like usability, user experience is also profitably divided into different axis, see figure 3-4.

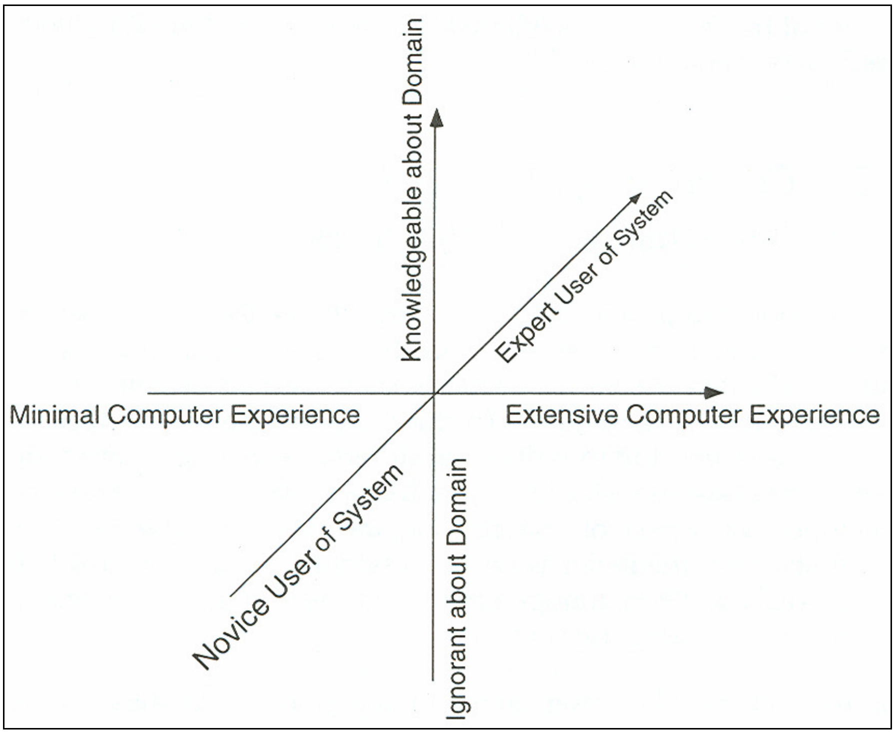

Figure 3-4 The three main dimensions of user' experience; general computer knowledge, knowledge of using the specific system, and knowledge of task domain (Nielsen 1993)

The two axes that are most important to consider when constructing a user interface are the potential users' general computer knowledge and their knowledge of the task domain. Little knowledge in both of these areas clearly set the highest demand on the user interface, with need for guidance both regarding navigating the user interface and for finding their way in the domain. The third axis, knowledge about using a specific system, is often what people think of when talking about beginners or experts (Nielsen 1993).

When conducting usability testing, the three dimension of the users experiences should be taken into consideration, both when selecting users for tests and when doing the assessment after tests (Dumas and Redish 1999).

4 Requirements Specification

This chapter corresponds to the Requirements discipline in the UP.

4.1 The basis for the requirements

4.1.1 The UNN Project

The staring point for this project is our experiences during the project at UNN (Wangensteen et al. 2005). Working with and teaching users that tried to master both a new computer tool and a totally new nursing vocabulary gave us a unique insight in the problems they had to face. Even though we were deeply involved in the process, and had no consciousness about nor education in observing and preserving our observations, we think that our conclusions in retrospect to some degree can be compared to those that would have resulted from a planned ethnographical field study, as recommended by Cooper (Cooper and Reimann 2003). To become more conscious about which users we are developing for, we have found that a simplified presentation of two personas have been fruitful. The primary persona presented below is the user group this project is focusing on. The secondary persona represents an additional user group that would have to been targeted if this project were going to continue into a real product. The two groups could also have been divided into novices and intermediates regarding general computer knowledge and domain knowledge in accordance with Nielsen's user experience dimensions (Nielsen 1993). However, we feel that the rich description of different aspects around personas is more fruitful, even though this is a somewhat distorted use of the concept secondary persona. A more correct use of these concepts in a project like ours would for instance be nurses as the primary persona and physiotherapists and occupational therapists as secondary personas, because they could have used the same interface as nurses with only minor changes.

Primary Persona

Pauline, 45 years old, is a registered nurse and specialist in geriatric nursing.
She is very ambitious in her work, tries to be updated on the subject of nursing, but feels it is hard to get enough time. She participates on seminars and courses at work. She sees the importance of documenting, but prioritizes to be at the patients' bedside if the time is sparse. She has got a few

hours superficial information about NANDA and NIC, and is a little bit skeptical about the idea of standardized nursing languages.
She is not interested in computers, they are somewhat frightening, but she thinks they can be OK to get specific tasks done. She uses a computer to surf on the Internet and read mail, and writes letters and notes in Word. She is not aware of that it is possible to change from one program to another; she is always closing the one she is using before starting a new. She get confused when asked to choose where to save documents, and is not used to hierarchical thinking. She is also confused when the layout of the screen that she is used to changes or when something unexpected is happening. She has seldom or never used a help file. She is a slow typist using two fingers.
Her end goal is to create care plans, and be able to find the right diagnoses and interventions.
Her experience goals are to don’t feel stupid, and to don’t make mistakes.

Secondary persona
June, 24 years old, is a registered nurse.
She is also ambitious in her work, and feels she is learning a lot from colleagues. She plans on educating into a nursing specialty area, either geriatric or psychiatric.
As newly educated, she is conscious about the need to document her work. She has had some lectures about standardized nursing languages, also about NANDA and NIC, and knows a little abut it, and is very curious to learn more.
She takes computers for granted, and is very seldom using a help file, rather she actively tries out new functionality by the try and fail method. She has written many scholarly subjects and projects on PC, knows how to crate catalogs under My Documents to save documents somewhat systematically. She types very quickly.
Her end goal is to create care plans, and rapidly and efficiently find diagnoses and interventions.
Her experience goals are to feel competent and confident and not be too bored.

The primary requirement for our project is to make a system and interface that meets the needs for Pauline, the primary persona. Secondarily, it should be possible to expand this interface to meet June’s needs of efficiency and speed, without distorting or blurring the first interface.

Pauline may also after a while become more confident, both regarding general computer use and use of this specific system, and become an intermediate that will appreciate the more efficient user interface meant for June.

4.1.2 State of the Art

The different hierarchical presentations of classifications of the Hands tool and Sanum are the inspiration for the requirement of hierarchical browsing.

The Hands tool, Sanum and Evolve care planner all offer the ability to select subgroups of diagnoses or interventions, either related to ICD-10 diagnoses, NANDA diagnoses or sub-specialties within nursing. These are features that would have desirable in our system. The official NIC classification contains a section of the most frequent used NIC-interventions related to each NANDA-diagnosis, and a section of core NIC-interventions within sub-specialties of nursing. Unfortunately, we did not obtain permission to use the electronic version of these sections of the NIC-classification, so this is requirements that we have to reject for this project.

The requirement for a key-word search facility to search for NANDA-diagnoses is based on our own idea. We had wished to implement an analogous search facility for NIC-interventions. This is a requirement we also had to reject because we did not get access to the electronic version of the Norwegian translation of the vocabulary.

DIPS is the starting point for our project. In spite of its lack of user-friendly user interface, it has many desired features. Especially the close connection between the “treatment plan” and the shift report is valuable. This is a feature that is outside the scope of our project, since we are focusing on the care plan alone. For a production project, we would have considered this feature as a high priority requirement.

DocuLive has a solution that makes it possible to connect problems to interventions in a number of ways. It is possible to connect one intervention to several problems. The solution is interesting, but seems complicated to relate to, because the user for every item deliberately has to connect it to other(s). This is though a feature that may be necessary to incorporate in our solution.

4.2 Vision statement

To describe the overall vision of the project, we use a simplified vision document with only the high level goals presented. Parts of these visions stem from the project work at UNN (Wangensteen et al. 2005), and parts have originated later.

High-level goal	Priority	Current Solution	Problems and Concerns
Relevant information regarding the diagnosis and interventions should pop up automatically, without the need for the user to actively fetch it.	High	Current system require that the user selects the diagnosis or intervention, push a button and wait while the help system opens up and display the information	Time to access information. Users may choose wrong diagnosis or intervention based only on the label. Users may find it difficult to relate to the help system
Presentation of a classification hierarchy, preferably several levels deep, making it possible to browse for a diagnosis or intervention	High	In the current system it is possible via an awkward way to browse through diagnoses and interventions categorized under one of the functional areas of the VIPS model	Time-consuming to browse. Only two level deep hierarchy, to many diagnoses and interventions in every category. Impossible or very hard to get an overview of available diagnosis and interventions
A search facility, making it possible to search for diagnosis and interventions based on signs or related keywords	High	Current system has a possibility to search, but only for words that occur in the labels	Very time-consuming to find labels for users not familiar with the classification. Tendency to use only a sub-set of labels. No incitement to explore and getting to know the classifications
An intuitive and easy workflow to create diagnoses and interventions	High	The current system is counter-intuitive	Very hard to learn current workflow, also hard to remember when not used for a while.
The layout of the care plan should reflect that interventions are related to a diagnosis, thus showing a	High	Current system presents the diagnoses grouped together at the top, and the interventions grouped	Counter-intuitive for most users and readers. Does not promote the good practice of relate an

hierarchy of diagnosis over interventions		together below.	intervention to a defined problem.
It should be possible to move diagnosis, interventions and activities up and down in the care plan by drag and drop	Medium	In the current system, it is possible to move an item one step at the time by selecting it and pressing a button	Cumbersome to move an item only one step at the time. It is important to rank items related to importance
An intuitive interface for editing and deleting items	Medium	In the current system, the edit and delete buttons are placed above the care plan in a button row	Not very intuitive interface, the user has to select the item, and then find the button for editing or deleting.
It should be possible to select related- and risk factors from the standardize vocabulary	Mdium	In the current system, these have to be entered as free text.	Users are often confused about the intention and intended content of the related to factor

4.3 Supplementary specification (partial)

Performance

The system should respond within reasonable time (fraction of a second).

Implementation Constraints

The system should preferably be implemented with free software, and free open source components should be used.

Search facility

The search facility should be as easy as possible to learn and to use. The result should be ranked according to relevance.

4.4 Use cases

Since we are developing a system where the design of the user interface is essential, we will not follow the black-box writing style, but rather be quite specific about user interface details.
We first present 3 use cases in fully dressed form, and then 10 more in brief form. The first 3 use cases give a detailed overview of the interaction between the user and the system and are a good starting point in developing the user interface. The rest of the use cases are very similar to the

first ones, and the user interfaces related to them can be implemented in a similar way. An elaboration of these will give little new insight into the problem at hand, and is therefore omitted.

Use Case UC1: Create Care Plan

Primary Actor: Nurse
Stakeholders and Interests:

- Nurse: Wants a care plan that accurately and completely describes the patients' problems and her interventions and activities to treat or mitigate these problems.
- Patient: Wants a comprehensive documentation of given and planned treatment.
- Hospital Management: Wants an overview of the complexity of patient care to assign resources and decide nursing acuity.
- Government: Wants to assure that documentation is done according to existing legislation.
- Other health professionals: Wants to quickly and in a well arranged way get an overview of the care the patient is receiving.
- Researchers: Wants statistics of the usage of different diagnoses and interventions related to outcome.
- Politicians: Wants more detailed information of patient care to better plan future health care delivery.

Preconditions: Nurse has logged in to the EHR and is authenticated. The patient is activated in the EHR.
Success Guarantee (Postconditions): A care plan with a time stamp and the author's signature is saved. All the statistical data relating to the standardized language is saved.

Main Success Scenario:

1. Nurse opens a new document to write the care plan.
2. Nurse selects to write a new diagnosis.
3. System presents different options about how to register the diagnosis.
4. UC2: Create Diagnosis.
5. The created diagnosis is presented in the care plan.
6. System presents different options about how to create an intervention related to the diagnosis.
7. UC3: Create Intervention
8. The created intervention is presented in the care plan.

Nurse repeat step 6-8 until indicated done.
Nurse repeat step 2-8 until indicated done.

9. Nurse accepts the care plan and the care plan are saved as a legal document into the EHR.

Use Case UC2: Create Diagnosis

Primary Actor: Nurse
Stakeholders and Interests:
- Nurse: Wants a diagnosis that describes one of the patients' problems.
- Patient: Wants a comprehensive documentation of given and planned treatment.
- Other health professionals: Wants to get an overview of the problems of the patient's.
- Administrators: Want to determine acuity.
- Researchers: Wants statistics of the usage of different diagnoses.
- Politicians: Wants more detailed information of patient care to better plan future health care delivery.

Preconditions: Nurse has logged in to the EHR and is authenticated. The patient is activated in the EHR. A care plan is open for edition.
Success Guarantee (Postconditions): A standardized diagnosis is created and presented in the care plan.

Main Success Scenario:
1. Nurse selects to write a new diagnosis.
2. System present different options about how to create a diagnosis
3. Nurse selects to browse the classification for a diagnosis
4. System present the domains and classes of the classification
5. Nurse selects a class
6. System presents the diagnoses within the selected class
7. When the nurse hover the mouse over one of the diagnosis, the system instantly presents all the information about the diagnosis
8. Nurse selects a diagnosis
9. System presents a list of factors (fetched from the classification) that is related to the diagnosis.
10. Nurse selects one of the "related to" factors from the list.
11. System present the finished diagnosis in the care plan

Alternative flows:

1-10. Nurse selects to cancel the process
   1. System returns to the care plan opened for edition with no changes

3-7. Nurse selects to cancel the process
   1. System presents different options about how to create a diagnosis

3-7. Nurse selects to search for a diagnosis using keywords
   1. System present a text field to enter one or more keywords
   2. As the nurse enters keyword(s), the system presents a list of diagnoses ranked according to the relevance of the keyword(s)
   3. When the nurse hover the mouse over one of the diagnosis in the list, the system instantly presents all the information about the diagnosis

3-10. Nurse selects to write a free text diagnosis
   1. System presents a text field where the nurse can enter the diagnosis
   2. Nurse enters the diagnosis

3. System presents a text field where the nurse can enter a “related to factor”
4. Nurse enters a “related to factor”
   4a. Nurse leaves the “related to factor” text field empty

10a. Nurse enters a free text “related to” factor
10b. Nurse leaves the related to factor empty

Use Case UC3: Create Intervention

Primary Actor: Nurse
Stakeholders and Interests:

- Nurse: Wants an intervention that describes the activities done to meet the patients’ problem.
- Patient: Wants a comprehensive documentation of given and planned treatment.
- Other health professionals: Wants to get an overview of the interventions planned against the patient.
- Researchers: Wants statistics of the usage of different interventions.
- Politicians: Wants more detailed information of patient care to better plan future health care delivery.

Preconditions: Nurse has logged in to the EHR and is authenticated. The patient is activated in the EHR. A care plan is open for edition.
Success Guarantee (Postconditions): A standardized intervention is created and presented in the care plan.

Main Success Scenario:

1. Nurse selects to write a new intervention related to a previously entered diagnosis.
2. System present different options about how to create an intervention
3. Nurse selects to open a list of the most common interventions related to the actual diagnosis.
4. When the nurse hover the mouse over one of the interventions in the list, the system instantly presents all the information about that intervention
5. Nurse selects an intervention.
6. System presents a list of activities (fetched from the classification) that is related to the intervention.
7. Nurse selects zero or more activities from the list.
8. System presents a text field to enter free text activities
9. Nurse enters zero or more free text activities
10. When indicated done, system presents the finished intervention (and eventually related activities) in the care plan

Alternative flows:

1-10. Nurse selects to cancel the process

1. System returns to the care plan opened for edition with no changes

3-7. Nurse selects to use another option to find an intervention

1. System presents different options about how to create an intervention

3 Nurse selects to open a list of all the interventions in the same class as the actual diagnosis.

3-5a. Nurse selects to browse the classification for interventions

1. System present the domains and classes of the classification
2. Nurse selects a class
3. System presents the interventions within the selected class
4. When the nurse hover the mouse over one of the interventions, the system instantly presents all the information about the intervention
5. Nurse selects an intervention

3-5b. Nurse selects to search for an intervention using keywords

1. System present a text field to enter one or more keywords
2. As the nurse enters keyword(s), the system presents a list of interventions ranked according to the relevance of the keyword(s)
3. When the nurse hover the mouse over one of the interventions in the list, the system instantly presents all the information about the intervention
4. Nurse selects an intervention

3-5c. Nurse selects to write a free text intervention

1. System presents a text field where the nurse can enter the intervention
2. Nurse enters the intervention

Use Case UC4 Create objective setting: Nurse decides to create an objective setting related to an intervention. The system presents a dialog box where the nurse can enter a free text objective setting.

Use Case UC5 Create activity: Nurse decides to enter a new activity related to an intervention. The system presents a dialog box where the activities proposed under the actual intervention in NIC is listed. The nurse can select one of these activities or enter a free text activity. Also predefined entries for the situation and frequency of the activity can be selected, or the nurse can enter free text descriptions of the situation and/or frequency.

Use Case UC6 Edit diagnosis: Nurse decides to edit a diagnosis. The system presents a dialog box where the related to or risk factors is listed. The nurse can select one of these factors or enter a new free text factor. The label of the diagnosis should <u>not</u> be editable.

Use Case UC7 Edit intervention: Nurse decides to edit an intervention. The system presents a dialog box where the nurse can enter a new free text objective setting related to the intervention. The label of the intervention should not be editable.

Use Case UC8 Edit objective setting: Nurse decides to edit an objective setting. The system presents a dialog box where the nurse can enter a new free text objective setting.

Use Case UC9 Edit activity: Nurse decides to edit an activity. The system presents a dialog box where the activities proposed under the actual intervention in NIC is listed. The nurse can select one of these activities or enter a new free text activity. Also predefined entries for the situation and frequency of the activity can be selected, or the nurse can enter new free text descriptions of the situation and/or frequency.

Use Case UC10 Delete diagnosis: Nurse decides to delete a diagnosis. The system presents a dialog box where the nurse can enter a reason for the deletion.

Use Case UC11 Delete intervention: Nurse decides to delete an intervention. The system presents a dialog box where the nurse can enter a reason for the deletion.

Use Case UC12 Delete objective setting: Nurse decides to delete an objective setting. The system presents a dialog box where the nurse can enter a reason for the deletion.

Use Case UC13 Delete activity: Nurse decides to delete an activity. The system presents a dialog box where the nurse can enter a reason for the deletion.

5 Design

This chapter corresponds to the Analysis and Design discipline in the UP.

The prototype presented in this project can be seen as the result of one of several iterations in the elaboration phase. In an earlier project (On the CD: INF-3982) much of the work in the inception phase was done, which is represented by the blue rectangle in figure 5-1. The red rectangle in figure 5-1 represents the work done in this thesis, were two iterations have been carried through. The amount of work in the earlier project has been approximately 1/3 of the amount in this thesis. In figure 5-1, this ratio is distorted; this merely illustrate that the UP has to be adjusted to each single project.

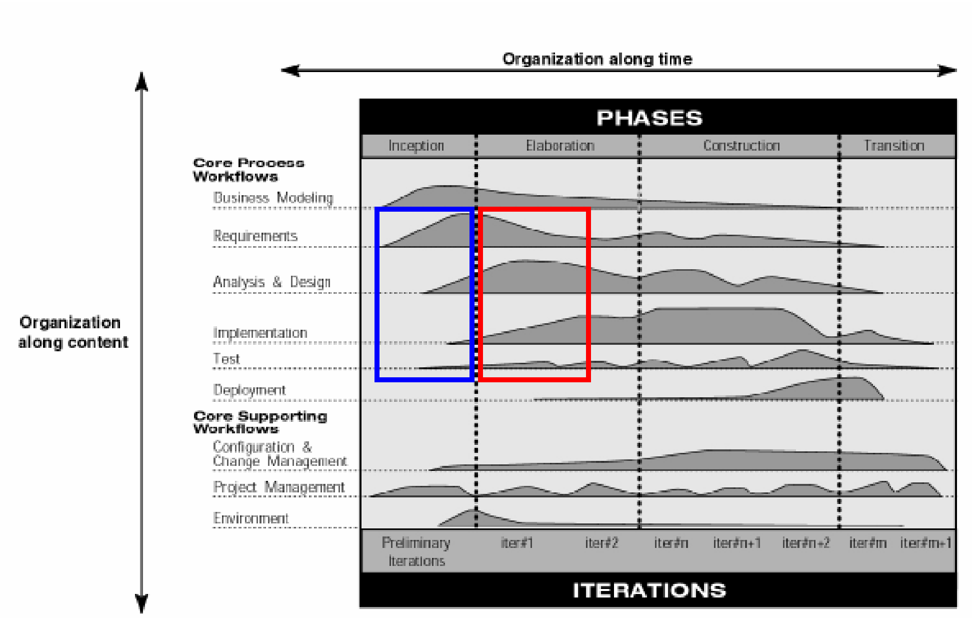

Figure 5-1 The development process of the prototype described in this paper, from (Baeza-Yates and Ribeiro-Neto 1999).

5.1 Inception Phase

Many of the requirements were settled in this phase, and risk factors as access to free components, development tools and knowledge about IR were sorted out. Also a preliminary architecture of the system was sketched. Most of the search for state-of-the art care plans was done in this phase.

The most important decision during the inception phase was how the key-word search facility should be implemented. To meet the requirement of easy querying and ranking for the search facility, an information retrieval (IR) technique was considered the best solution. It was decided to use one of the classical IR models, since these would perform good enough to test the concept. The classical IR models consist of the boolean model, the vector model and the probabilistic model (Baeza-Yates and Ribeiro-Neto 1999).

The boolean model assumes that the users master the construction of complex queries with boolean operators. This would require a lot of education of the users, and also frequent use of the system in order to maintain their knowledge. As this is contrary to the idea of the whole project, this model is not feasible. Additionally the model does not offer ranking of the retrieved documents.

Both the vector- and the probabilistic model offer the possibility of using queries with natural language. In addition, both offer retrieval of documents with terms that only partially matches the query, and both offer ranking of the retrieved documents. The probabilistic model does not offer index term weighting, and this is a shortcoming, since it would be preferable to be able to put more weight on words that occur in the labels and definitions of the diagnoses than to word that occur elsewhere in the text.

Because of this we have chosen to use the vector model in this project. A large amount of other ranking retrieval strategies have been compared to the vector model, but despite its simplicity the vector model has been superior to or just as good as most of its alternatives (Baeza-Yates and Ribeiro-Neto 1999). The popularity and rich documentation of this model has also contributed to this choice.

5.2 Elaboration Phase

5.2.1 Sequence diagram

To get an overview of the collaboration between the nurse and the system, a system sequence diagram can be used. The sequence diagram for the Use Case UC2: Create Diagnosis is shown in figure 5-2. We have chosen to define the system to be constituted of the Application and Database, and have drawn the system border around these in figure 5-2.

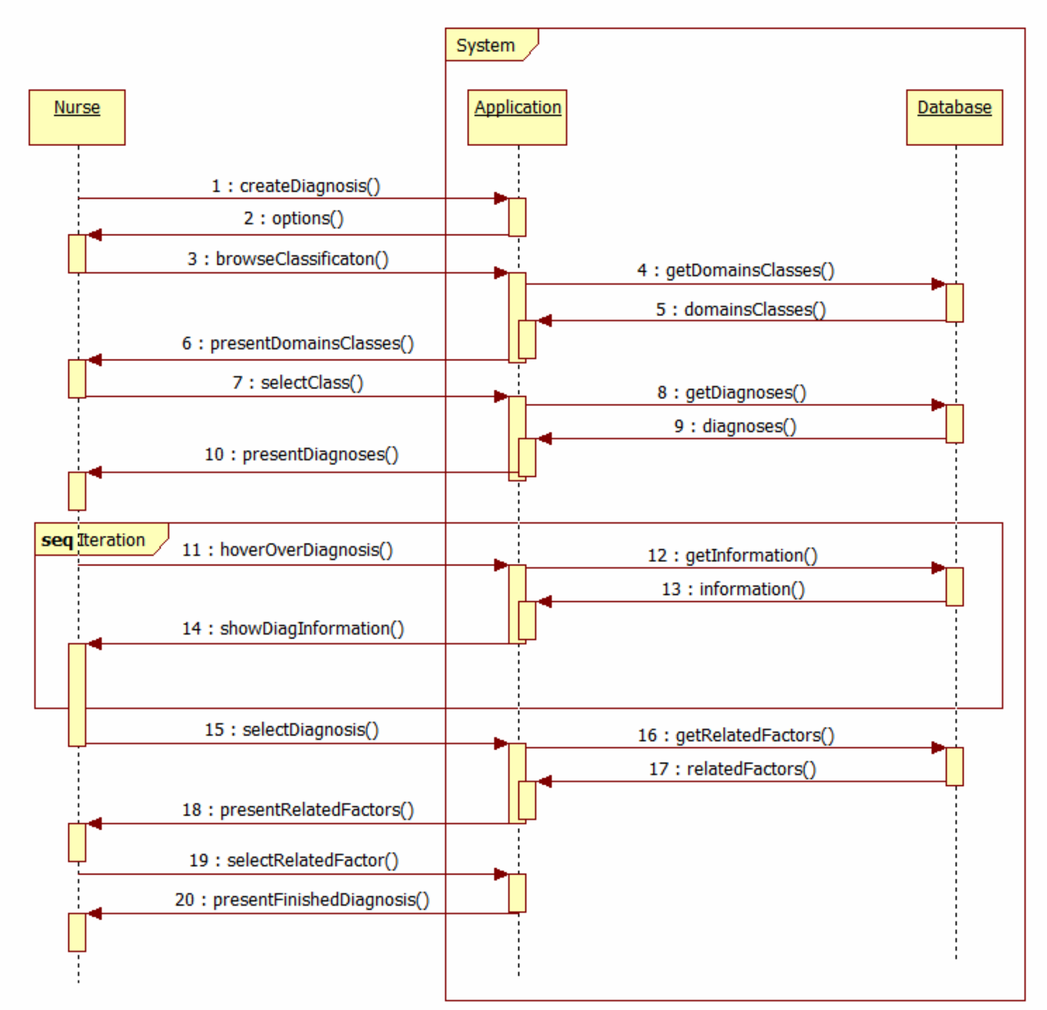

Figure 5-2 System sequence diagram for use case Create Diagnosis

The events between the Application and the database are drawn in order to give an overview of the necessary communication with the database.

The system events created by the user (nurse) are then as shown in figure 5-3:

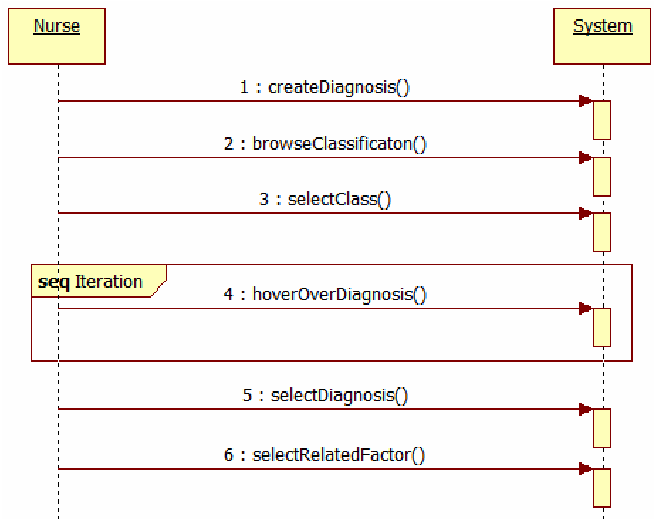

Figure 5-3 User-created system events

The system events created by the user makes a good staring point to create the user interface, which must provide items such as buttons, dialog boxes, and lists corresponding to each of the system event.

5.2.2 Domain/Class diagram

The domain model is used as a map to concepts in the domain. Since we are developing a user interface, we choose to see the domain of the system as interface items; forms, dialog boxes and controls within forms, which create events. This domain map is so close to the implementation classes, that we skip the domain model and present a simplified implementation class diagram, shown in figure 5-4. The methods in the classes would in the domain model have been represented as actions done on controls. For instance could the createDiagnosis() in the ApplicationForm class have been represented as a button click in the domain model.

The main purpose of the model in figure 5-4 is to get an overview of how dialog boxes or window forms can cooperate to accomplish the task of creating a new diagnosis.

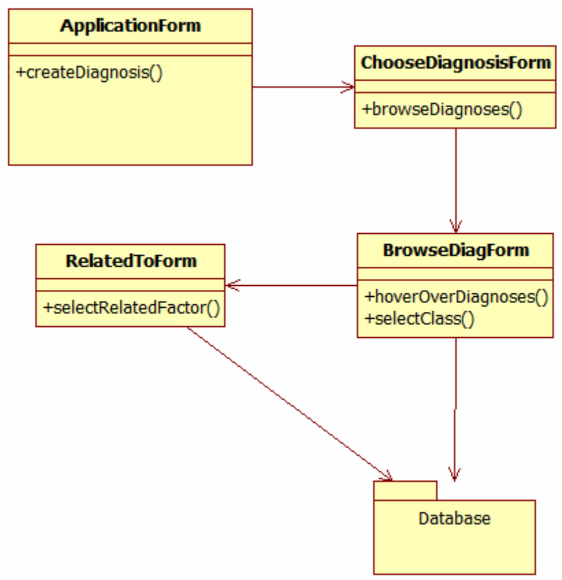

Figure 5-4 Simplified class diagram

5.2.3 Architecture

The ApplicationForm of figure 5-4 needs to cooperate with the database in order to get to the diagnoses and the information related to them, as indicate in the sequence diagram in figure 5-2. The data created in the ApplicationForm also needs to be taken care for, so these are put into the Careplan data package. Additionally, cooperation with the IR-system in order to use the search facility is necessary. The packages of the system are shown in figure 5-5.

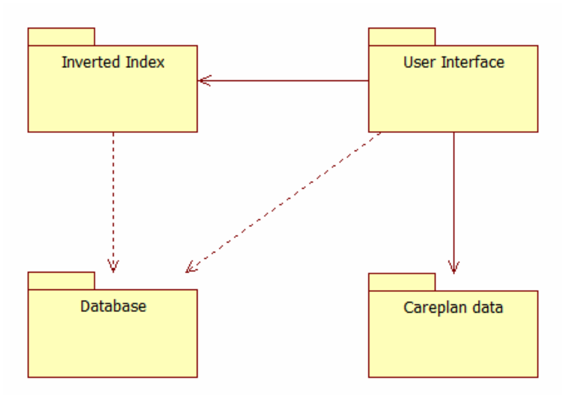

Figure 5-5 The overall architecture of the system

The User Interface is the window forms and dialogs that the user sees and interacts through. As the application starts, an instance of an Inverted Index is created and kept in memory as a member of the main form of the User Interface. Also an instance of the CarePlan is created. The CarePlan contains the data created by the application, and can be serialized and deserialized. The database consists of the standardized language and the KPO classification. The User Interface queries the database for information about a diagnosis or the structure of the classification whenever the need arise. The Inverted Index additionally uses the database to build the index.

6 Implementation

6.1 Software and hardware

The database was implemented using Microsoft SQL Server 2005 Express Edition (Microsoft 2007b).

The solution is implemented using C# with Microsoft Visual C# 2005 Express Edition (Microsoft 2007a).

In order to be able to place controls into the tree view presentation of the care plan, a commercial component was purchased: Lidor System's IntegralUI TreeView (Lidor 2007).

The whole system has been implemented and is run on a Dell Latitude D610 portable PC with an Intel® Pentium® M processor 1.73 GHz and 1 GB RAM.

6.1.1 The database

The table structure shown in figure 6-1 was made using the Management Studio IDE that comes with the SQL Server.

Since we got the electronic version of the NANDA diagnoses first, the first table to be populated with data was the NANDA table in figure 6-1. The file with the diagnosis was converted to a plain text file simply by copying the original file via the windows clipboard to a text editor (On the CD: Text versions\NANDA.txt). A c# program was created to parse the text file and put it into the database (On the CD: Source Code\NandaToDatabase\Code\Program.cs). The program reads one diagnosis after another and puts it into a row in the table. The code, label, defining characteristic, related factors and risk factors of each diagnosis are split and put into different fields, as indicated in figure 6-1. The reason to split the diagnoses is to be able to give different weight to words that occur in different part of the diagnoses, and to be able to fetch only parts of a diagnosis, as a label or risk factor, when need be. Each diagnosis is then considered a document unit in the IR system.

The AddedWords field was populated with words resulting from manually reading through the indexes and dividing compound words.

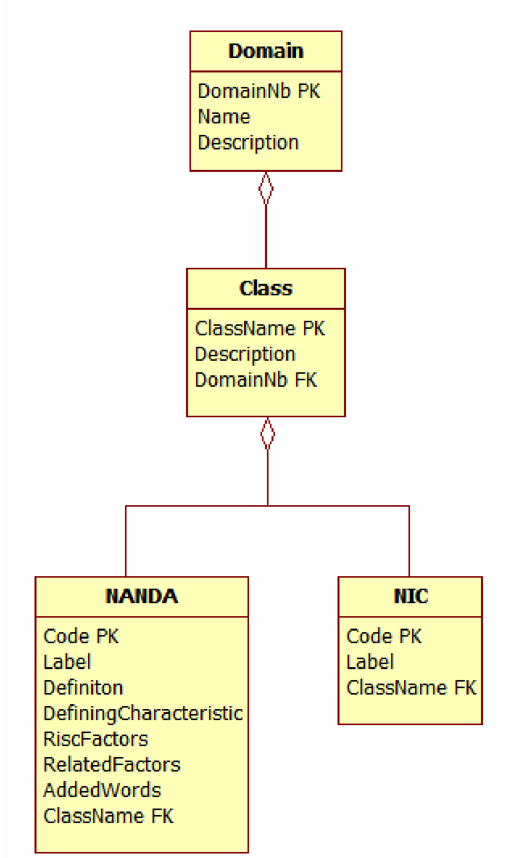

Figure 6-1 The database structure. PK and FK indicates primary- and foreign keys respectively.

The KPO-classification was also copied to plain text versions (On the CD: Text versions\ KpoInterventions.txt and KpoDiagnoses.txt). A program to parse the KpoInterventions.txt was created (On the CD: Source Code\KpoToDatabase\Code\DomainClassNIC\Program.cs) to populate the Domain, Class and NIC tables in figure 5. To update the ClassName field of the NANDA table, this program was slightly rewritten and the writing to the Domain and Class was commented out (On the CD: Source Code\KpoToDatabase\Code\NANDA \Program.cs). It was then run with the KpoDiagnoses.txt as input.

6.1.2 The Inverted Index

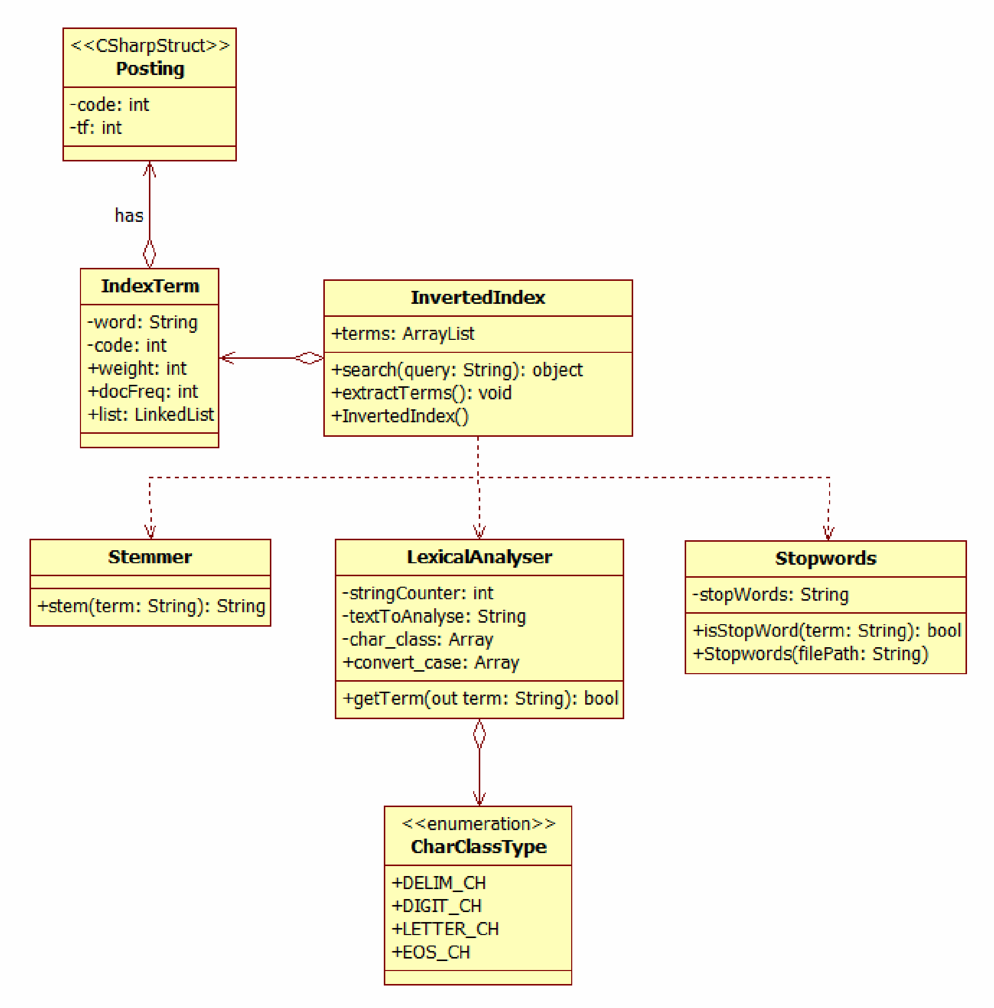

Figure 6-2 Class diagram of the Inverted Index

The `InvertedIndex` class is used to create an inverted index, se figure 6-2. The method extractTerms() is building the index, and is typically called only once at the startup of the program. The class's method, `search(query)`, is used to search the inverted index for a query passed as an input parameter. Both these methods operate against the NANDA table in the database. The class contains to other methods, extractTermsNIC() and searchNIC(), with its corresponding attribute indexTermsNIC. These methods and attribute are used against the NIC table in the databases, and are merely replicas of the first methods, and they will therefore not be further discussed.

The classes used by the `InvertedIndex` will first be described.

LexicalAnalyser:

The `LexicalAnalyser` (On the CD: Source Code\Overall Design\InvertedIndex\ LexicalAnalyser.cs) is implemented with a C-language example in Frakes and Baeza-Yates (Frakes and Baeza-Yates 1992) as a template. It has three attributes, a string, an integer and a `CharClassType`. The string is the text to analyze, and is given as a parameter to the constructor of the class. The integer is a counter which keeps track of the index of the actual character to be analyzed. The `CharClassType` is an enumeration defining four different types of characters; `delimiter, digit, letter` or `EOS` (End Of String). A static array gives the `CharClassType` of a character based on its ASCII-value as index, and a static array `convert_case` is used to convert all characters to lower case. The actual text analyzing takes place in the method `getTerm(),` which is implemented as a finite state machine, see figure 6-3. The state machine starts at the present counter value in state 0, going to state 1 whenever a letter is found. State 1 returns true when a valid term is found, and at the same time a valid term is returned as an out parameter. A valid term is a sequence of characters or digits, not starting with a digit. When the end of string is reached, the machine goes from state 0 to state 2, which returns false.

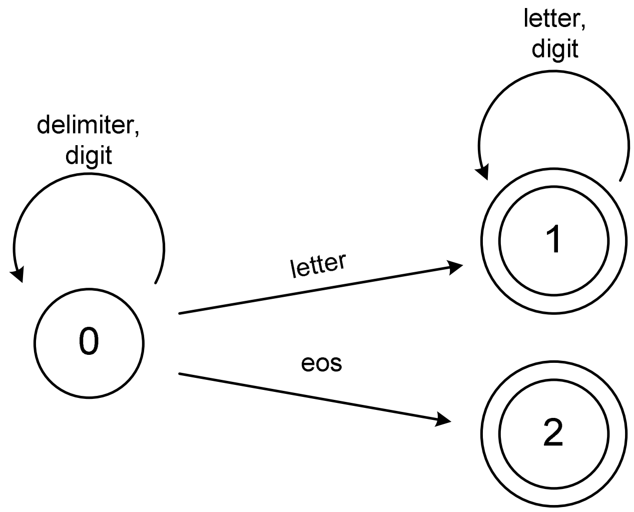

Figure 6-3: Transition diagram for the query analyzer. States with two circles are final states.

<u>Stemmer</u>

The basis for the `Stemmer` class is a program that stems according to english rules.(Hazelwood 2001). This program is almost totally rewritten to stem according to norwegian rules (On the CD: Source Code\OverallDesign\InvertedIndex\PorterStemmer.cs). The stemming rules found at (Porter 2007) are implemented. The class is tested against a sample vocabulary and its stemmed

equivalent found at (Porter 2007), and its output is identical to the sample stemmed vocabulary when compared in Microsoft's WinDiff (Microsoft 2003).
In the original program, the word to be stemmed is represented as a character arry. It has two overloaded `add()` methods, one that add one character at the time and one that add a character array to the word to be stemmed. We have added a third `add()` method that accept a word represented as a string to be added to the stemmer.

Stopwords

The `Stopwords` class simply reads a file containing stopwords and places these into a string attribute (On the CD: Source Code\OverallDesign\InvertedIndex\Stopwords.cs and stopwords.txt). The method `isStopword(term)` returns true if the parameter `term` is contained within this string.

IndexTerm

The class `IndexTerm` is for convenience reasons used in two different contexts. First it is used in the first pass extracting all the words in the collection, and second it is used to store the final terms in the `Index` of the `InvertedIndex`(On the CD: Source Code\OverallDesign\InvertedIndex_IndexTerm.cs). It therefore contains attributes to cover both of these usages. The `word` attribute is the term stored as a string, and is used in both contexts. The code of the diagnosis from where the term is extracted and the weight of the term are stored in its respective attributes in the first pass. The `docFreq` attribute is the number of document where the terms occur, and is used in the final context. The same is the `list` attribute, which is a reference to a liked list containing information of where the term can be found and the term frequency.

Posting

The `Posting` class is used in the nodes of the linked list pointed to by each final term. It holds the code and term frequency of the term (On the CD: Source Code\OverallDesign\InvertedIndex\Posting.cs).

`InvertedIndex`

The `InvertedIndex` class (On the CD: Source Code\OverallDesign\InvertedIndex\ InvertedIndex.cs) contains the attributes `indexTerms` and `docLengths`. The `indexTerms` is an

ArrayList of the final terms that comprises the index, while the `docLengths` is a Dictionary where the key of each item is a diagnosis code and the value is the square root of the number of words in the diagnosis.

The method `extractTerms`() creates a local ArrayList used to store all the terms from the NANDA – table as they are extracted. A connection to the database is established, and then each diagnosis is read one after another. Within each diagnosis, the 7 fields are read into a string. An instance of a `LexicalAnalyser` is created, with the string of the field as the parameter to the constructor. The method `getTerm`() of the `LexicalAnalyser` is then called until there are no more terms in the string. Each term returned is checked against the `Stopword's isStopWord`() method. If it is not a stopword, an instance of the `Stemmer` is used to stem the word. An `IndexTerm` instance is created, and the term, the actual diagnosis' code and the field's weight are stored in this object. The field's weight is given the value 5 for field 1 (the label), 2 for field 2 (the definition), and one for the rest of the fields. This implies that each word occurring in the label or the definition, get a final score as if they occurred 5 respective 2 times in the total document. The term is then added to the local ArrayList, and the value of a counter to count the document length incremented. After each diagnosis is read, the code and the square root of the document length are added as a new key/value pair to the attribute `docLengths`.
When all diagnoses are read, the connection to the database is closed. The local ArrayList of `IndexTerms` is sorted in descending order on the `word` field (the default compare field in the `IndexTerm` class). The sorted ArrayList is then looped through. For each new `word`, a LinkedList with a `Posting` is created. Following terms with equal words is looped through, and new `Postings` are added to the linked list if it is the first occurrence of the word for a diagnosis' code. If the code already has a `Posting`, the weight is updated. When there are no more equal words, an instance of an `IndexTerm` (used in the second and final context) is created, with the actual `word` and linked list as attribute values. Also the `docFreq` is set to the length of the actual linked list. This new `IndexTerm` is added to an ArrayList of merged index terms, which is cloned to the class atribute `indexTerms` when all the terms are merged.

The `search`() method processes the query string similar to what is done in the `extractTerms`() method. The terms in the query, which is given as an input parameter, are extracted. Then they

are stemmed and added to a local ArrayList. The terms are not checked against the stopword list; if a stopword should be entered in a query, it will be a miss in the search process.
A binary search of the `indexTerms` is then performed for each query term, and the indexes for the hits are saved. A dictionary named `score`, where the key is the diagnosis' code and the value is the score is created. Then for each index term in the hits, the *idf* (inverse document frequency) is calculated, with the index term's `docFreq` as a denominator. The linked list for the index term is examined, and for each posting the value is calculated as the term frequency times the inverse document frequency. The value is then divided by the document length collected from the attribute dictionary `docLength`. This is a simplification when compared to equation 2-5, because the weight of the query terms is set to 1. The value is then added to the score dictionary under the actual code's key. In the case that the query contains more than one query term, and the actual code already is in the dictionary, its value is updated.
The score dictionary is then sorted on its value, resulting in a ranked list of diagnoses where the first diagnosis has the highest score.

6.1.3 The Care Plan

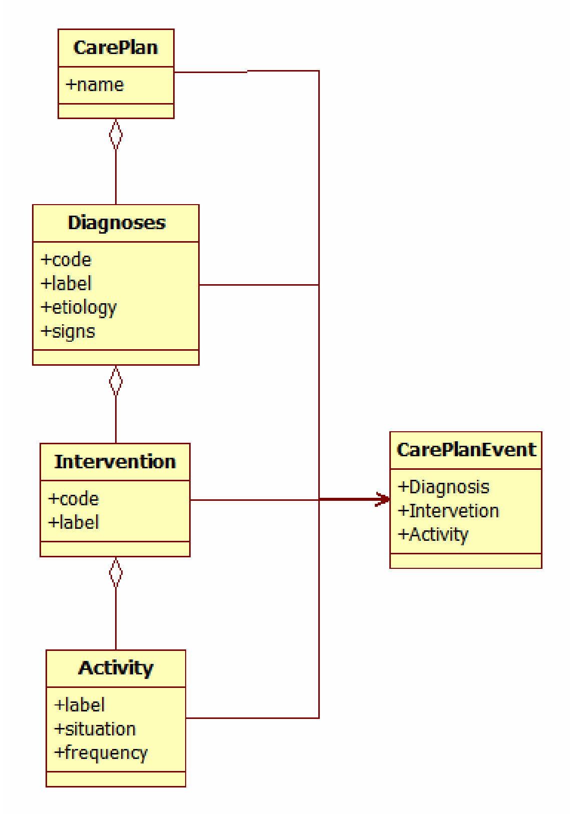

Figure 6-4 The care plan

The care plan is implemented as a hierarchy of classes where each class is aggregated in the class above (On the CD: Code\OverallDesign\ CarePlan*.cs).

Every change in the care plan is creating a `CarePlanEvent`. This makes it possible to have a low coupling between this package and other packages (Larman 2002). This is though not fully carried out in the current prototype. The classes, except the `CarePlanEvent`, are serializable. Event handlers have to be implemented manually in classes that deserialize the classes.

6.1.4 The user interface

AppForm

The main class in the application is the `AppForm` in the `UserInterface` package (On the CD: Source Code\OverallDesign\UserInterface\AppForm.cs). When loading, the `AppForm` creates an inverted index by calling the `extractTerms`() method of the `InvertedIndex` class. The presentation of the care plan is done with the tree view from Lidor Systems (Lidor 2007).

During the startup of the application, a new care plan object is created, and the intervention "Frittstående tiltak" and the activity "Frittstående forordninger" is added to this care plan. For the care plan, and all objects added to it, a `CarePlanEventHandler` is added. This implies that all changes in the care plan are handled by the `CarePlanEnentHandler carePlan_Changed()`. The handler is responsible for painting the tree view, which is painted according to the underlying data in the care plan. The presentation of the care plan can be done using two modus, presentation- and edit mode. When in edit mode, button-nodes to create new items are placed at the uppermost location for its kind; the diagnosis at the top node, the interventions in the top node at the level below its diagnosis, and the activities at the top node under its interventions. When a new item is added, it is placed in the first index of its ArrayList, and hence is painted right below the button-node that was used to initiate the creating of the item. This makes some consistency to the layout.

Every item in the care plan can be moved by drag and drop, which is implemented in the tree view's event handlers `tvwECP_DragDrop` and `tvwECP_DragOver`. Different rules regarding as to where an item can be dropped are implemented, depending on which item is dragged. Diagnoses can be moved up or down at root level. Interventions can be moved up and down between other interventions under the same diagnosis, or moved over to another diagnosis. Activities can be moved up and down between other activities within an intervention, or to other interventions, irrespective of what diagnosis this intervention is placed under. All the sub items of an item are moved along when an item is dragged and dropped. Standard Window visual feedback is given while an item is dragged; a rectangle indicates that dropping is allowed, while a stop sign indicates not.

When a user selects an item in edit mode, a button is presented, as shown in figure 6-5.

Figure 6-5 The Edit, Delete and Move button. The button is shown in the red rectangle

The button is created as a `UserControl` in Microsoft's .net terminology (On the CD: Source Code\EditDeleteButton\EdDelButtons.cs). A user control is built as a dynamic link library, and is activated and used through a reference to it. A click on the "Endre"- button activates a dialog box which offers possibilities to change the care plan item. A click on the "Slett"- button deletes the item from the care plan, without any further warning. Clicking the arrows moves the item up or down within its level. If an item is on the highest (or lowest) position at its level, nothing happens when clicking the up (or down) button and no visual feedback is given. Hence it is not possible to move a child node to another parent node by using the up/down buttons. This is a functionality that remains from and earlier iteration before the drag and drop functionality was implemented.

In a real world application, the nurse would approve the care plan when she was finished editing it, and it would be saved to the EHR as a legal/formal document. To mimic this, it is possible to save a care plan to disk. Similarly, to mimic continuing editing of a patient's already instituted care plan, it is possible to open an earlier saved care plan. This functionality is implemented in the methods `btnSave_Click()` and `btnOpen_Click()`. The `btnOpen_Click()` also connects each element in the care plan to the `CarePlanEventHandler's carePlan_Changed()`.

Dialog boxes

There are 14 different dialog boxes in the application. These are used to create and edit diagnoses, interventions and activities. Since this is a prototype, new dialogs have just been added as the need arose, and code reuse has not been emphasized.

The dialog `BrowserForm` (On the CD: Source Code\OverallDesign\UserInterface\BrowserForm.cs) is an exception, it is used both to browse for NANDA diagnoses and NIC interventions. Since this dialog and the dialog `SearchDiaForm` (On the CD: Source Code\OverallDesign\UserInterface\SearchDiaForm.cs) is the most complex dialogs in the application, a short description of their functionality follows. When the `BrowserForm` opens, either the method `showNanda`() or `showNic`() is called, depending on a parameter given to its constructor. These methods establish a connection to the database and uses queries to build up a hierarchical presentation of the classification in a tree view control in the dialog. All the domains and classes are expanded, and for the `showNic`() method, also the class corresponding to the diagnosis under which it is called, is expanded. The label of the diagnosis is also passed as a parameter to the constructor.

The tree view event `NodeMouseHover` is used to present a help or information window. In this method also a connection to the database is established, and all the information regarding the diagnosis is presented in the information window.

The `SearchDiaForm` also presents an information window when the mouse pointer is hovered over an item in its list view control. The functionality is equal to that of the `BrowserForm`, but additionally the keyword(s) from the search is highlighted with a blue colored font in the information window.

7 Results

We present the results from two usability tests and some observations from how the search functionality functions. In order to appreciate the results and follow the discussions concerning the usability tests, we think that an understanding of the workflows in the systems is valuable, so we start by a presentation of these.

7.1 Illustration of the Work Flow in the two Systems

To illustrate the difference between the two systems, the workflow for crating a diagnosis is shown.

7.1.1 Workflow in DIPS

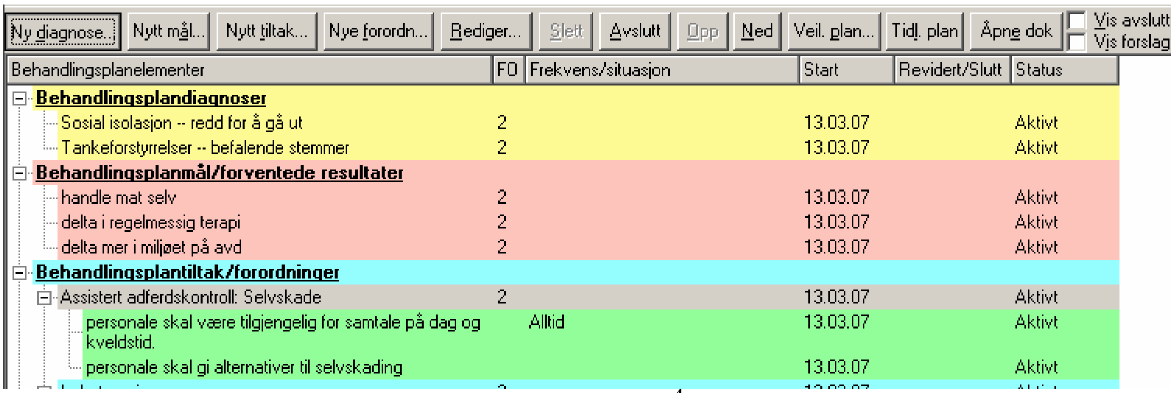

Figure 7-1. Initiating the creating of a new diagnosis in DIPS[4]

In figure 7-1, the "Ny diagnose…" button in the top left corner is clicked to start the process of creating a new diagnosis. The system then presents the dialog box in figure 7-2.

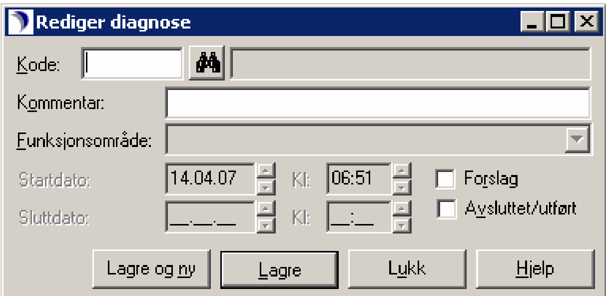

Figure 7-2. The edit diagnosis dialog box

Here it is possible to enter the code of the diagnosis. Totally there are 155 different codes to choose among in NANDA. This suggests that it is assumed that the user already has looked up the diagnosis or intervention in a book or in the help system. Fortunately, and almost as a second thought it seems, there is placed a search button alongside the text field, marked width a binocular. By clicking this button, the dialog box in figure 7-3 is presented.

[4] Neither pictures of DIPS nor our system contains real patient's data. The pictures of DIPS where taken using the test version of the program.

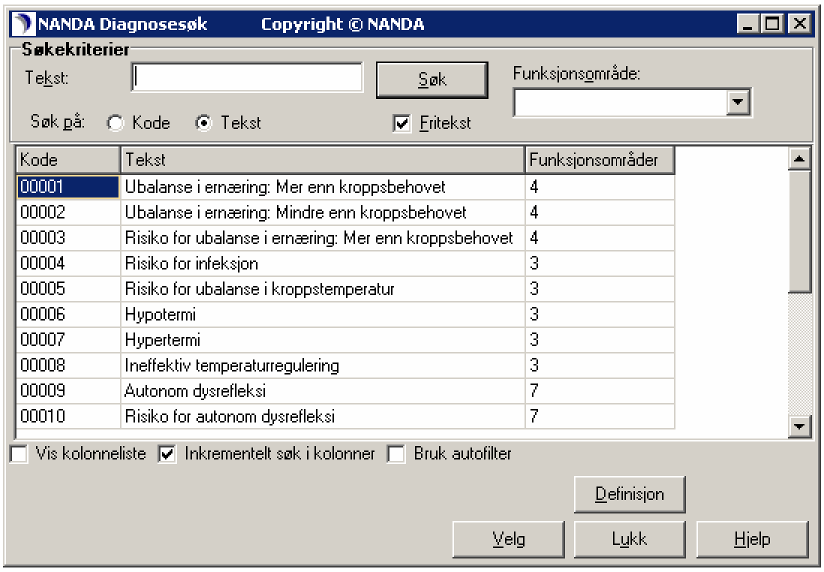

Figure 7-3. The initial dialog box to search for NANDA diagnosis.

This dialog box gives the option to search for a diagnosis either by code or by text. For the text search, only exact matches of words in the label of the diagnosis gives results. This option can be useful for users that are somewhat acquainted with the classification. It is also possible to narrow the search within a functional area by selecting one from the combo box as shown in figure 7-4.

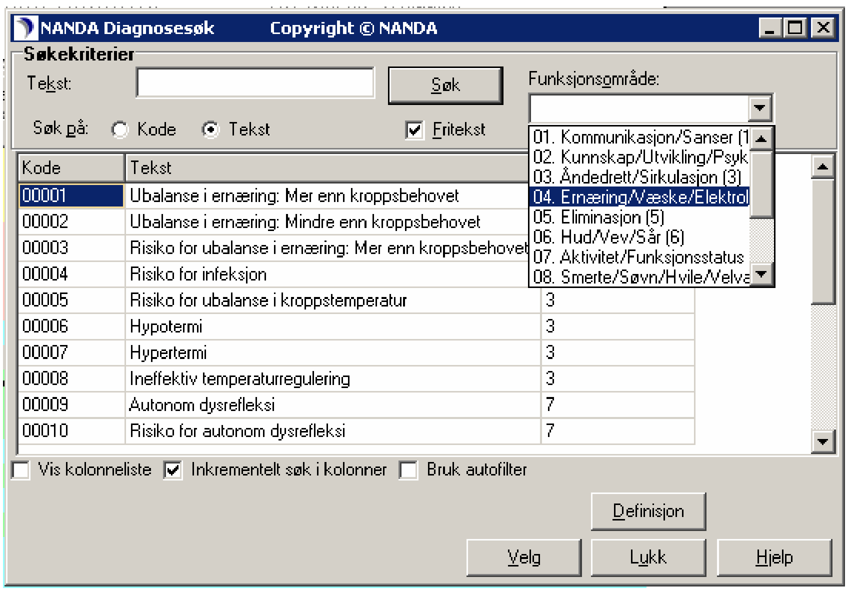

Figure 7-4. Narrowing the search

In figure 7-3 and 7-4, all the diagnoses are listed sequentially in the list view, from code 00001 to code 00155.

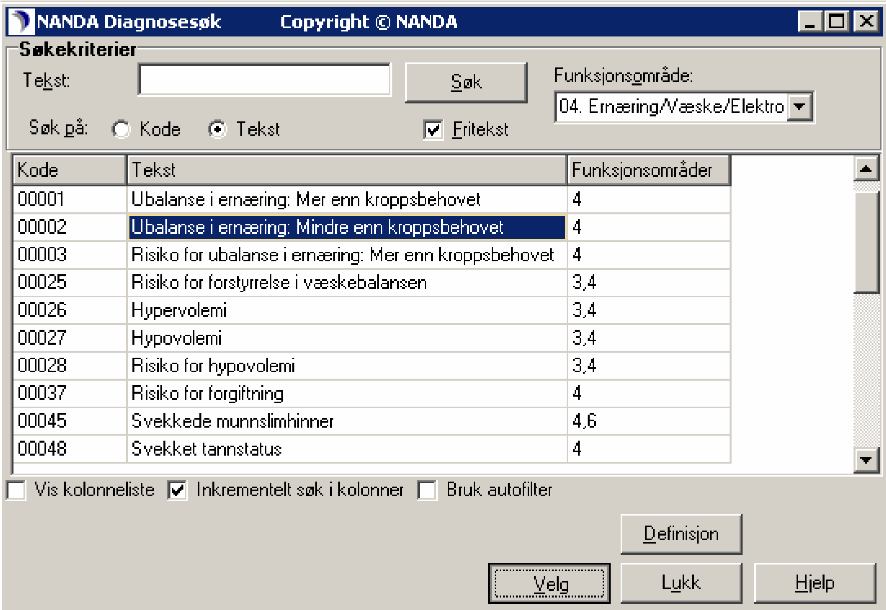

Figure 7-5. Browsing within a functional area

By selecting a functional area in the dropdown list box and clicking the "Søk"-button with an *empty* textbox, the diagnoses within this area is listed, as shown in figure 7-5. The original list of 155 diagnoses is then diminished to a list of 20 – 25 diagnoses[5]. This makes it feasible to browse the diagnoses for those who are not familiar with the classification. By selecting a diagnosis in the list and clicking the "Definisjon" – button in figure 5, the help system of DIPS is opened and presents the information of the selected diagnosis, see figure 7-6.

[5] This was a technique developed during the UNN project to be able to browse for diagnoses at all. We do not know if other users of DIPS are employing this technique.

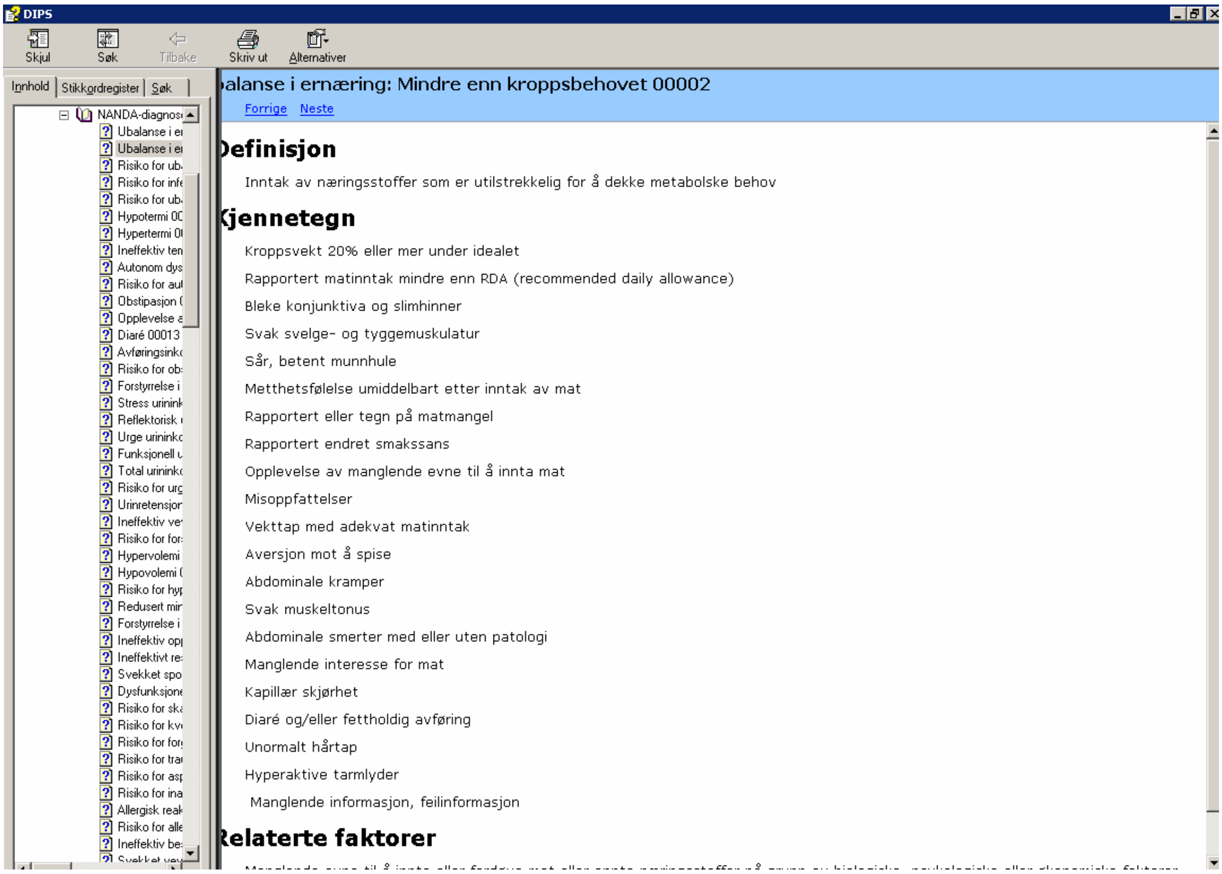

Figure 7-6. The help system presenting the selected diagnosis.

The process described in figure 7-5 and 7-6 can be reiterated until an appropriate diagnosis is found, but the user has to switch between the help system and the main program. By clicking the "Velg" button in figure 7-5 dialog in figure 7-7 appears.

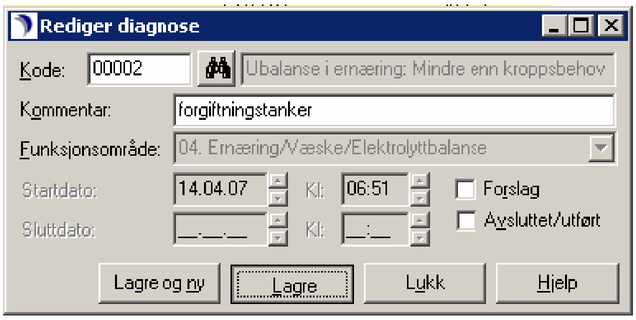

Figure 7-7. The "related to" dialog box

This dialog box gives an opportunity to comment the diagnosis, for instance by writing the etiology or a "related to factor" of the diagnosis. By clicking the "Lagre" button, the diagnosis with an eventual comment is placed in the care plan as shown in figure 7-8.

Figure 7-8. Part of the care plan with the new diagnosis selected

7.1.2 Workflow in our solution

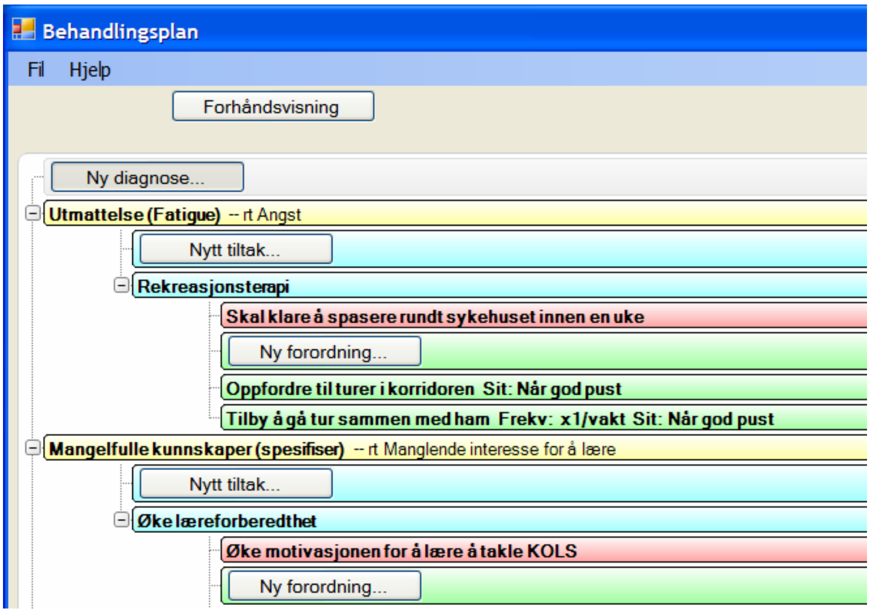

Figure 7-9. Initiating the creating of a new diagnosis in our solution

To start the crating of a new diagnosis, the "Ny diagnose…" – button at the top of the care plan in figure 7-9 is clicked. The system then presents the dialog box shown in figure 7-10.

Figure 7-10 Selection of different paths to create a new diagnosis

By clicking the "Klassifikasjon…" – button, the window in figure 7-11 is opened.

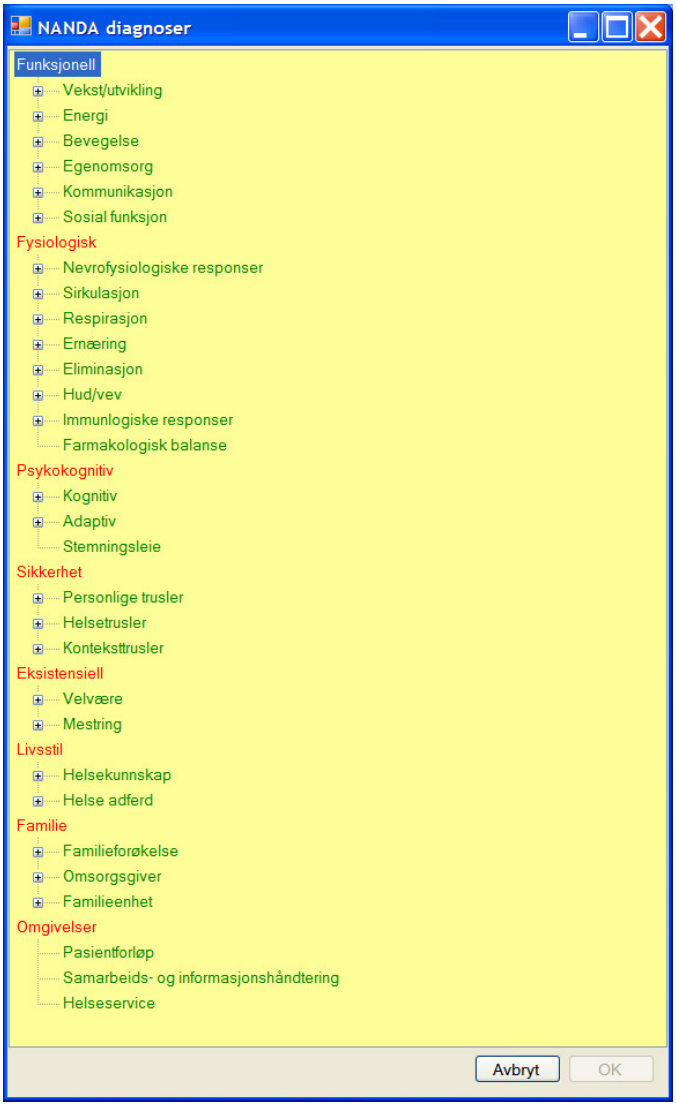

Figure 7-11 Browsing the classification to find NANDA diagnoses

Figure 7-11 shows the domains and classes of the KPO-model.

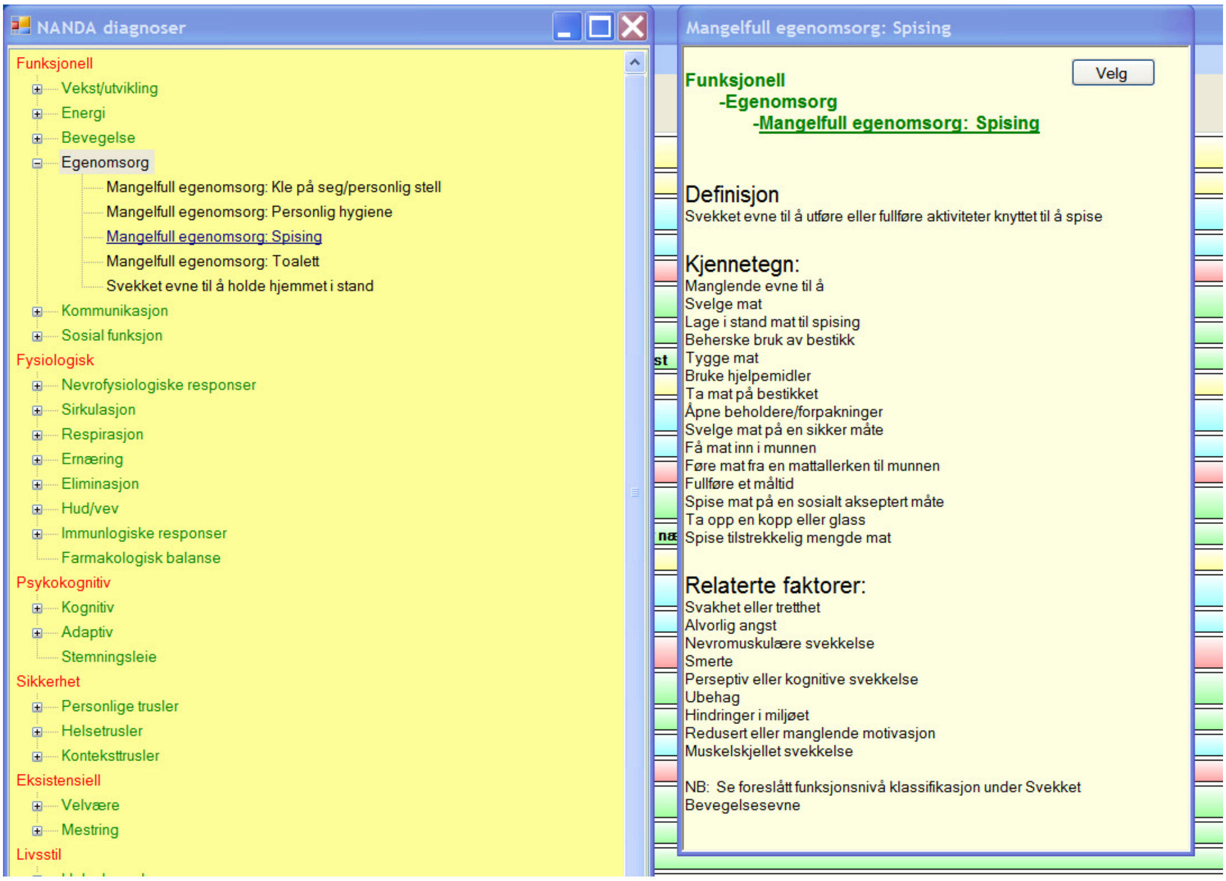

Figure 7-12 Browsing for and selecting a diagnosis

By expanding a class node, the diagnoses within the class are shown, as in figure 7-12. By hovering the mouse over a diagnosis, the information window to the right pops up, showing the related information. The information in this window changes to display the related information as the mouse hover over different diagnoses. A diagnosis can be selected by double-clicking on its node, by selecting the node and clicking an OK- button at the bottom of the window (not shown in figure 7-12), or by clicking on the "Velg" – button in the information window. The windows are mutually dependent, and the closing of one implies the closing of the other.

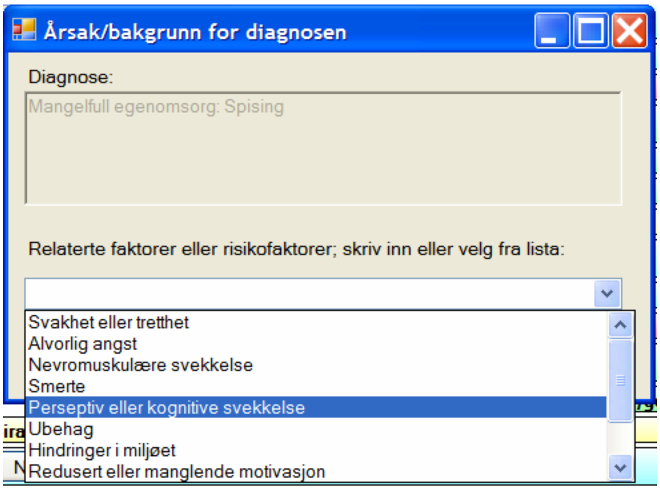

Figure 7-13 Selecting or writing a related factor

When a diagnosis is selected, the dialog box in figure 7-13 is presented. It is possible to select a related to- or risk factor fetched from the classification, or write a free text comment.

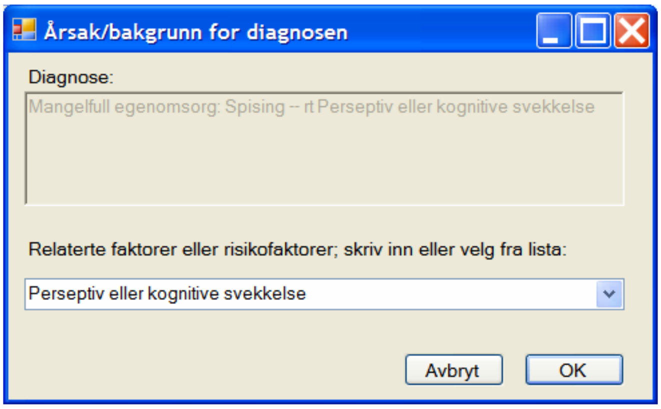

Figure 7-14 Accepting the diagnosis

Pressing the OK – button in the dialog box in figure 7-14 places the diagnosis into the care plan as shown in figure 7-15.

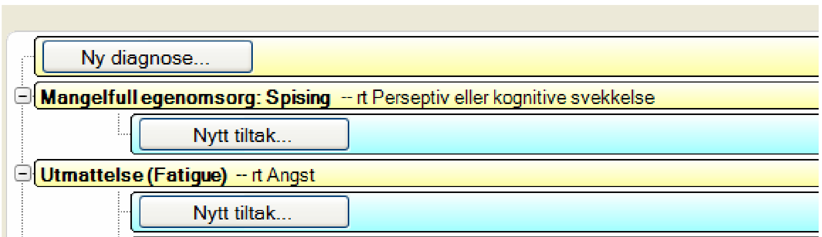

Figure 7-15 The care plan with the newly created diagnosis

To find a diagnosis by keyword search, the “Søk…”-button in figure 7-10 must be clicked. The dialog box in figure 7-16 is then opened.

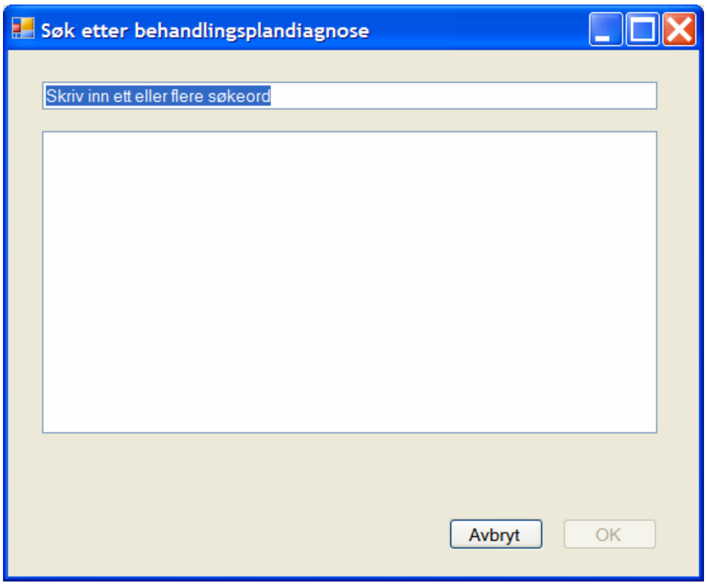

Figure 7-16 The initial keyword search dialog box

The result after typing in two keywords and selecting a diagnosis is shown in figure 17.

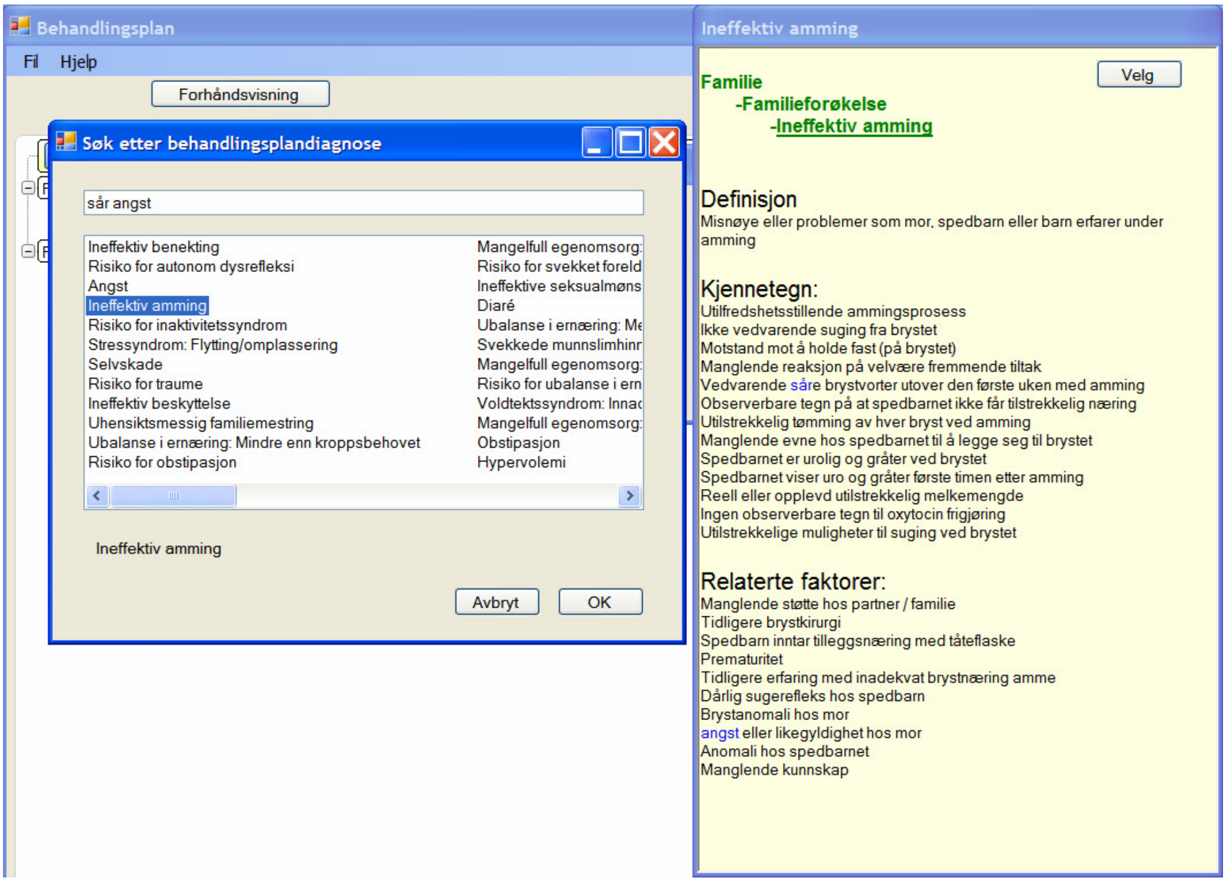

Figure 7-17 Selecting a diagnosis after a keyword search

As in figure 7-12, hovering the mouse over a diagnosis in the list at the left shows the relevant information in the window to the right. The selection of the diagnosis is also done in the same way, and results in the opening of the dialog box shown in figure 13. The path to finalize the diagnosis is the same as described above from figure 7-13 to figure 7-15.

7.2 Usabillity tests

To test out our solution, we have carried out two usability assessments, one as a pure usability test and one as a combined usability test and semi-structured interview. The assessments were performed with the help of DIPS users. For both, our system was compared to DIPS. The test users all came from the psychogeriatric ward at The Department of Special Psychiatry (SPA) at the University Hospital in the Northern Norway. The ward is examining and diagnosing elderly with a complex and comprehensive clinical picture. Approximately 100 patients are accepted a year, with an average stay of 6 – 8 weeks. For the last 2 years, the nurses have documented their work in DIPS, using both shift reports and care plans. Care plans are used very actively, and has and have to some extent diminished both the need for and length of the shift reports (Wangensteen 2007, Wangensteen et al. 2005). Typically, an initial care plan is made within a day, or even within hours, after a patient's admittance, and is updated and completed as the patient's needs and problems are unveiled. The percentual amount of standardized diagnoses and interventions (NANDA and NIC) in the care plans has steadily been around 90 (ibid).

7.2.1 Assessment 1

The intention of assessment 1 was to get an impression of the learnability and efficiency of the two systems, and also to compare the workflows of the systems. It was conducted as a plain usability test, where the need to understand the standard vocabularies was eliminated as far as possible (see appendix 2 for a schematic outline of the test in Norwegian). According to Nielsen (Nielsen 1993, Nielsen 2000), the best cost – benefit result for usability testing is achieved using 5 users, so correspondingly, we used 5 users. The users were from 20 to 55 years, 2 were employed as auxiliary nurses while the rest was unskilled workers. Using one of the systems, the users were shown how to create a predetermined diagnosis, write a predetermined related to/risk factor, and then change the related to/risk factor. The user was afterwards asked to complete the same task himself. The sequence of instruction and user completion was repeated using the other system. The time to complete the tasks was measured, and observations on how the user was performing were made. The test concluded with an informal debriefing, where the users were asked to compare the two systems regarding which they would prefer using. In order to get a best possible comparison, we aimed at getting users that had not used DIPS to write care plans earlier. However, all but one of the users had earlier more or less informal been shown how to write care plans, and 2 had actually written care plans in DIPS.

The time taken to complete the tasks is shown in table 7-1.

User	Time DIPS	Time our system	Started with	Errors
User 1	4 : 30	3 : 00	Our system	
User 2	2 : 25	2 : 08	Our system	Forgot to type in the related to factor in DIPS, not corrected. Selected wrong intervention in our system, corrected during session.
User 3	1 : 57	1 : 25	Our system	

User 4	2 : 20	1 : 18	DIPS	The intervention was placed under "Frittstående tiltak" in our solution, not corrected
User 5	1 : 35	1 : 12	DIPS	Selected wrong diagnosis in our system, corrected during session

Table 7-1: Results of utility test 1. The time taken to complete a predefined task in DIPS and in our system

When a user did not know what to do next, the test leader waited for approximately 10 seconds before giving instructions. This relatively short time interval was chosen because we did not want to embarrass the testers more than necessary. The amount of instructions given is shown below (one of the users stood for 6 instructions, 3 in both systems, and this user's result is therefore omitted):

Total amount of instructions in DIPS: 4

Total amount of instructions in our system: 1

One flaw with our system that was not foreseen was detected during this session. One of the test persons was confused when the hierarchy for selecting interventions was opened, because the hierarchy did not look similar to the hierarchy for the diagnoses, see figure 8-18. The window opened with the class of the diagnosis opened. Contributing to the confusion was the fact that the hierarchy window opened with the content scrolled down, so the topmost domain and classes did not show. The amount of scrolling is dependent of which class is opened and how many interventions they contain.

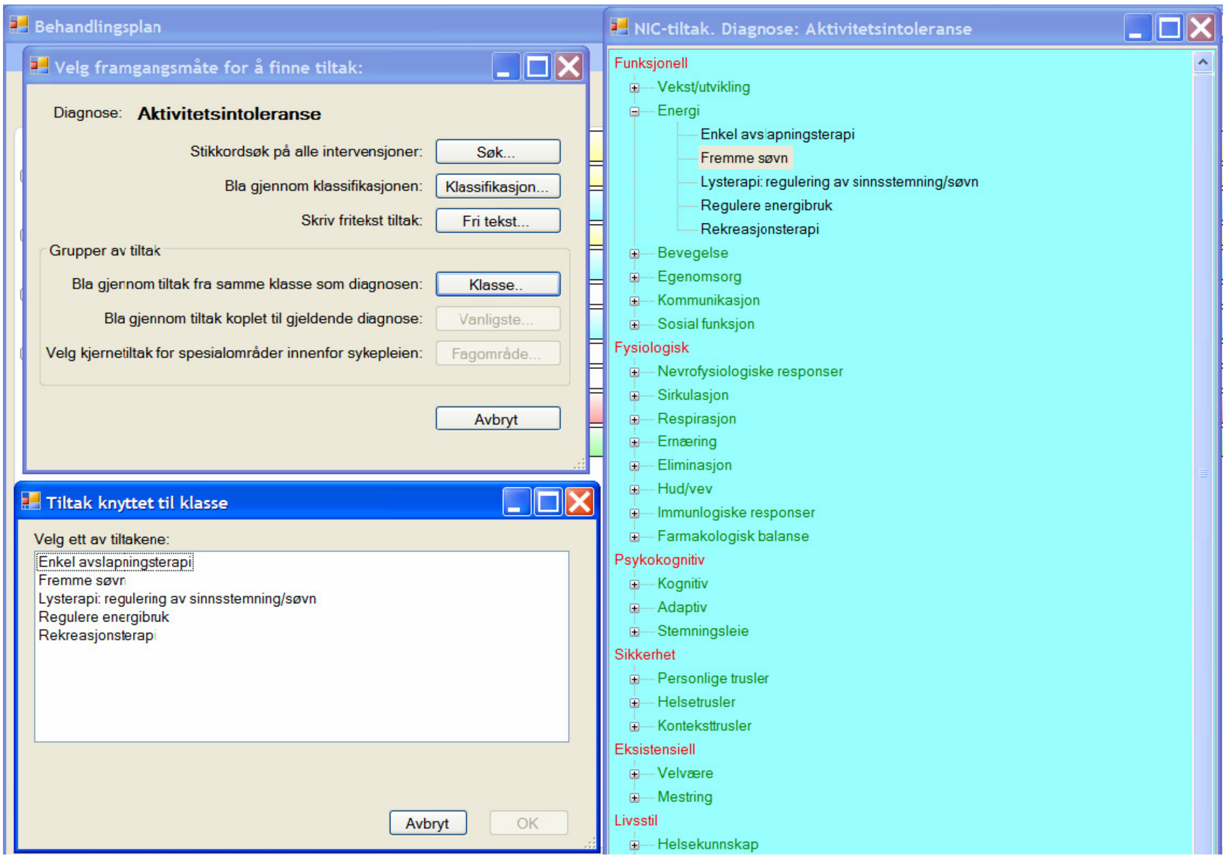

Figure 7-18: Two different procedures to find interventions from the same class as the actual diagnosis. The blue dialog window to the right is a result of clicking the "Klassifikasjon…" – button, while the dialog in the lower left is from cliking the "Klasse…" – button.

7.2.2 Assessment 2

4 users, 3 nurses and 1 with a background from social science, who all were experienced in writing and editing care plans participated in test 2. The frequency of writing care plans varied from very often to "once in a wile" among the participants. The intention of the test was to have an impression of how the hierarchical browsing through the classification worked and how easy it was to navigate. First our system was demonstrated. A few diagnoses and interventions were created, both by browsing through the classification and by using the search facility. During the demonstration and afterwards, the semi-structured interview was held, se appendix 2. At the end of the interview, the participants were asked to create a care plan with the two diagnoses "Urge urinary incontinence" and "Toileting self-care deficit", and then add 1 self-selected intervention and 1 self-selected activity to each of the diagnoses. This was done both in our system and in

DIPS, and the time elapsed was measured, se table 7-2. During this test, no instructions were given.

User	**Time DIPS**	**Time our system**	**Started with**	**Notes**
User 1	1 : 45	1 : 37	Our system	
User 2	Not measured	Approximately 1 : 00	Our system	Used the search functionality for the 2 diagnoses, program crashed when searching for interventions
User 3	3 : 15	3 : 20	DIPS	
User 4	2 : 15	1 : 57	DIPS	

Table 7-2: Results of test 2

A summary of the results from the semi-structured interview combined with observations during the test follows.

7.2.2.1 Workflow

All but one of the users felt that it was easiest to navigate and find one's way in our system. The one that stood out compared the two systems as equal regarding this functionality. Observations during the test confirmed this result. This user had initially some problems with the layout of the care plan in our system, and spent some time to figure out which buttons to use to initiate the creation of a diagnosis and an intervention, and where these were placed in the care plan when created. The user maked out these questions alone without the need of help from the test leader. All the other users fluently found their way without hesitation in both systems, except from two participants that had some minor problems finding out how to open/expand the class node in the classification hierarchy in our system.

7.2.2.2 KPO - classification

One of the users had seen the KPO – classification once before while evaluating an earlier version of the system. This participant stated that the classification felt natural, because it was recognizable from other nursing theories, but pointed at some shortcomings and did not agree with all categories. The user believed it would function well as a cognitive map.

For the other users, it was hard to come up with substantiate statements after this short session. Two of the participants felt positive about the hierarchy, one felt that the three level hierarchical presentation was better and clearer than the presentation in DIPS, while the another felt it was hard to relate to the domains, but still it was easier to relate to and find diagnosis within the classes of the KPO – model than under a functional areas of DIPS. She added that it probably was just a matter of exercise and experience to be able to relate to the structure of the classification. The last participant was uncertain about which system was the best.
Observations from the test confirmed the statements above, where one user systematically used the domain and classes to find the diagnoses, while the two others scanned the classes, trying to deduce from their name if they contained the diagnosis. One of the participants did not use the classification to find the diagnosis, se next section.

Another observation done was that it was a lot easier for the nurses to relate to the classification than for the participant with another background.

7.2.2.3 Searching

All of the participants appreciated the search functionality. Some comments about it were that it would be nice for novices, and one participant said: "… should have had this one when we started out using NANDA…". Another comment was: "Very good, searching and a ranked list together with the immediate presentation of the standard language gives a real opportunity to choose among different diagnosis and find the one that correspond to the patient's problem." This user later spontaneously used this functionality in the test. Typing in "urine" resulted in 13 diagnoses where the wanted diagnosis was ranked as second, while "self-care" resulted in 12 diagnosis where the wanted was ranked at top. Unfortunately, when the user was trying to search for an appropriate intervention, the program crashed, so that the test had to be interrupted[6]. The user claimed that he without a doubt would have needed much longer time to fulfill the task in DIPS than in our system.

---

[6] The search functionality for NIC interventions where implemented in an early prototype, and is merely a blue-copy of the inverted index and search functionality for NANDA diagnoses, with a lot of functionality commented out in the code, because only the labels of the classification are available. It was not meant to be used, and thus not tested as the prototype developed. Besides being an example of bad test preparation; the users should have been instructed to use the browsing facility, this incidence also illustrate the importance of usability testing; users always do unexpected things!

7.2.2.4 Related Factors and Risk Factors

Only two of the participants had viewpoints on this issue. One stated that this was a good feature, which would save time and also function as a pointer or guide to what the related to factor was meant to be. Users would become more conscious about what to write when they chose to use free text instead of the predefined factors, and not use for instance medical diagnoses, but rather etiological factors or a deepening and explanation of the diagnosis. The other participant felt that the language used was somewhat academic and hard to understand, but it was all right to get these factors from the standardized vocabularies as long as the possibility to detail the diagnosis in free text was maintained. She also thought that this functionality would have been better if it was possible to add free-text related-to factors that were frequently used on the ward, i.e. that these also should be added to the dropdown list and be selectable.

7.2.2.5 The Information Window

All of the participants praised this feature. One of the participant's comment were: "Very good, people seldom look up the definitions, now they is forced to relate to the vocabulary". Even one of the users in test 1, where this feature was not an issue, spontaneously exclaimed when she saw this feature: "This would have been nice to have when writing the shift report, then I would better know what all these concepts are all about!"

7.2.2.6 Diagnosis and Interventions from the same Class

All of the participants agreed that in most cases, it would be far from sufficient to use interventions from the same class as the diagnosis.

7.2.2.7 The layout of the care plan

This issue was the one with most discrepancy among the participants. One felt that the layout in DIPS was the best, but added that this could be because he was used to it. Another as well claimed that our new layout seemed strange, but that it might have been useful if it was possible to easily expand or close groups of items, so that for instance only the diagnoses were shown. It could also be useful to expand only one diagnosis with interventions and activities when doing work related to this particular diagnosis. The two last participants both felt that our layout was the best. One of these had strong arguments for her view. Several times when she had followed patients that were transferred to a nursing home, she had printed out the care plan as part of the transferal note. The nurses at the nursing homes always had a hard time to understand the care plan. She said: "… and to have them to understand, I had to sit there and draw arrows between the diagnoses and interventions to show the connection, (…) that made it much clearer for them".

7.2.2.8 Drag and Drop

All the participants appreciated this feature, and all mentioned the importance of being able to rank the diagnoses and interventions according to urgency.

7.3 Experiences with the search functionality

We will here present some observations done related to the search facility. The observations are done while developing and testing the search functionality, during the usability tests.

7.3.1 Standalone search facility

For simple searches based on signs and symptoms, the general impression is that recall is pretty good. For most searches that consist of one or two key-words, relevant diagnoses are found within the 5 to 10 highest ranked hits. Most users rapidly get used to change their key-word to a synonym or add additional key-words when they are not satisfied with the hit.

There are though no guaranties that important diagnosis related to the keywords are found. Some examples of this are presented in the following. The keyword "fever" gives 2 diagnoses; "Disturbed sleep patterns" and "Impaired swallowing". Changing the search to "temperature" gives 15 results, of which all the diagnoses directly related to this sign is present. Searching for "pollenallergy[7]" gives no result, while "allergy" gives two diagnoses, of which one is relevant to "pollenallergy". Searching for "delusion" gives only the 2 diagnoses "Sleep deprivation" and

[7] Pollen allergy is written in one word in Norwegian.

"Risk for other-directed violence", while the gross number of the diagnoses in the "Psychocognitive" domain and a few other diagnoses are much more relevant to this symptom.

7.3.2 Support to learn the classification

The search functionality can also serve as a supporting tool to learn the classification. During test 2, one of the users tried to find the diagnosis "Anxiety" browsing the classification. She did not find it under the domain and class where she meant it naturally belonged, and got frustrated about this. A search for "anxiety" resulted in 32 diagnoses, where "Anxiety" was ranked number two. When hovering the mouse over the diagnosis the information window is popping up, and the placement of the diagnosis in the hierarchy is presented at the top of this window.

7.3.3 A Tool for Differential Diagnosis

As indicated by one of the participants in test 2, the search facility can be used as a tool for differential diagnosis. A search for "urine incontinence" gives 15 diagnosis, of which the 8 highest ranked is directly related to the key words. Using the readily available information, the difference between the diagnoses can easily be explored.

8 Discussion

The two usability tests both gave insight to the workflow and the navigation of the classification hierarchies, to a greater extent than originally expected. This was due to observations done during the tests, which gave far more information than we had thought beforehand.

8.1 Assessment 1

We did not have full control of picking the users to participate in this test. Testing was done during an afternoon shift, and the five participants was the total sample of potential users that was available. Two of the participants had extensive computer experience; two had some and one had minimal experience, according to Nielsen's three dimensions of user experience (Nielsen 1993). The user with minimal computer experience, although being in the target group of this development project, did not fit into this test design. She needed help with almost all tasks during the test. We have therefore decided to omit the result from this participant in the following discussion. This user still gave valuable information of flaws in our system.

During the project period at UNN (Wangensteen et al. 2005), tree points in the workflow where most users had problems was identified. Users did not know what to do with the edit diagnoses dialog in figure 7-2, and had problems proceeding from this dialog. Also the awkward way to filter diagnoses according to functional areas stopped many users. The third point was that to edit, delete or move a diagnosis or intervention, the user has to select it, and then use buttons above the care plan to carry out the desired action. All of these problematic points were observed during the tests, and all but one of the users was hesitating while performing these tasks in DIPS. 2 of the users had to get instructions about how to proceed from the dialog box, and 2 got instructions about editing the diagnosis. For our system, 1 user needed help about how to edit the diagnosis. This indicates that our system allover has an easier and more intuitive workflow than DIPS. 4 of the 5 users also confirmed this by clearly expressing that they would prefer our system before DIPS. When it comes to the functionality of editing an item in the care plan, a clear conclusion is harder to draw. In both solutions the user has to know that it is necessary to click on the item he wants to edit. We had hoped that it should be possible to implement a user interface where this functionality was pronounced and obvious for the user. In an earlier iteration, we tried a solution where the EditDelete button (figure 6-5) was permanently shown in every item when in edit mode, but we abandoned this solution because the excess crowding of buttons degraded the overall overview of the care plan. A solution with the EditDelete button visible to the far right of each item may be tried in a later iteration. 2 of the users thought that our solution was slightly preferable because of the immediate visual feedback of where to click, and one of these spontaneously used the delete functionality of the button to remove a wrongly entered diagnosis during the test session.

All of the participants completed the test session faster in our system than in DIPS, se table 7.1 This was to a large extent due to the issues discussed above, but the cost of having to find an item in a two level hierarchy compared with a 3 level hierarchy also contributed to the different time consumption. Absolutely all of the participants used more time to scan through the long lists of items under each functional area in DIPS than the shorter lists under the classes of the KPO model. The number of necessary rereading of the lists because of overlooking an item was also higher in DIPS. There is consequently a limit to how many diagnoses or interventions that should be aggregated in one class. In other words, the relation between a wide and shallow hierarchy and a narrow and deep affect the efficiency of browsing for known items. This relation will probably also affect the ease of navigating for unknown concepts, as when a nurse is looking for a diagnosis that match a patient' problem. It is clear that the classification of the KPO – model is superior compared to DIPS' model in this respect. It would have been interesting to examine this relation between the KPO – classification and the proposed joint taxonomy of the NNN-alliance. The proposed taxonomy of NNN can be found in appendix 1.

To sum up, the conclusion must be that we have succeeded in creating a system that is easier to learn and is more intuitive than DIPS. In addition, it is more efficient.

Another conclusion that is easy to draw is that DIPS would improve the usability of their system immensely simply by just removing the first dialog box with the binocular in figure 7-2, and by automatically filter the diagnosis and interventions when a new functional area is chosen in the combo box in the following dialog box, se figure 7-4.

8.2 Assesment 2

The questions in the semi-structured interview are discussed in sequence, and observations from the usability test that relate to the questions are discussed when appropriate.

8.2.1 Workflow

All but one user navigated through our system without any noticeable difficulties. One had some initial problems, but managed to sort these out without external help. All the users evaluated our system as easier as and more intuitive to navigate than DIPS. Also the fact that all but one of the users had never seen our system before the test session, but yet accomplished the test tasks as fast or faster than in DIPS, indicate that our system has a better workflow and is more efficient than DIPS, see table 7-2.

8.2.2 KPO - classification

In our system, the minimum number of mouse clicks needed to create a diagnosis is 5, wile in DIPS this is 8, additionally some scrolling is often necessary in DIPS. The measured times listed in table 7-2 show that both user 1 and user 4 did the task faster in our system then in DIPS. User 3, which used the same amount of time in both systems, experienced some navigational problems in our system, and additionally had some problems browsing for the diagnosis in both systems. User 1 and 4 both found their way and browsed for diagnosis almost without hesitations, in both systems. The difference of mouse clicks and browsing accounted for some of the time difference, but as in the first test, most of the difference was due to the different hierarchy levels of the two systems. The presentation of the domain and classes of the KPO-model in 1 screen, also allowed for a much better overview and direct and fast access to the classes, and to separate out the class that presumably contained the searched for diagnosis. The fact that one of these users had seen and tested our system with the KPO - classification once, and the other never had seen it before, while they have used DIPS to browse for diagnosis thousands of times, again strongly indicate the strength of a 3 level classification hierarchy presented in one screen, over the 2 level hierarchy used in DIPS.(henvisninger)
1 skjerm overikt domene cognitive map

The user with a social science background was the one that had most problems relating to the KPO-classification. This seems reasonable with a classification originating from nursing theory.

The present KPO-hierarchy with domains and classes is implemented using a list view. The list view was chosen because of its commonality and well known functionality. This is apparently functioning well, as shown in the tests. Yet, this is in practice more an implementation model than reflecting the users' model of the hierarchy, and it should be investigated if for instance a representation of the hierarchy as shown in figure 2-4 is better. The representation in figure 2-4 may be easier to understand for users that are not used to relate to hierarchies. If the diagnoses or interventions also automatically were opened while the mouse was hovering over a class, we feel that this may be a better presentation than the current one.

In the next iteration, he features for the secondary persona have to be incorporated into the current interface. It should be possible to select multiple diagnoses and interventions while browsing the hierarchy. With the complete NIC language present in the database, it should also be possible to select a range of the proposed activities from the information window. All of these additions can be achieved by using check boxes, and does not incur much disturbance to the current interface. The process of making care plans is one of thorough assessments of patients' status. It is time-consuming, and the use of a tool to write it down is only a small part of this process. The need for efficiency should be seen in seen in this context, and not be emphasized too much.

8.2.3 Searching

This functionality is more thoroughly discussed in chapter 8.3. All the users saw that this functionality could help novices to start using the classifications, and one clearly demonstrated that it could be used to speed up the creation of diagnoses.

The information window that shows when a diagnosis is searched up shows the placement of the diagnosis in the classification. Our hope is that this can help speeding up the learning process when novices are trying to learn the classification. In future iterations, a direct link from the information window to the classification presentation window could also prove helpful to promote the learning process.

In the interface for the secondary persona or for intermediate users it should be possible to search for diagnoses and interventions directly from the care plan, for instance from a field to the right of the new diagnosis and new intervention buttons. If a search after each character entered is implemented, and the result is presented in a drop down list like for instance with IntelliSense in a programming IDE, the user only have to type a few character before he can select the wanted diagnoses or intervention (MSDN 2007).

8.2.4 Related Factors and Risk Factors

(diskuter I forb med å legge til ord i databasen)

Early in this project, one idea was that the system should “harvest” the free text embellishments of the diagnoses and add these words to the database as extra key-words to improve the search functionality. This was abandoned after a read-through of a few care plans, because the quality of the embellishments varied considerably, and sometimes was even misleading. This is an observation that also others have done (Krogh and Dale 2007). Because so many users, especially in the start of the project at UNN, had difficulties with how to embellish the diagnoses, we decided to fetch the related to- and risk factors from the standard vocabulary, and let the user have the ability to select from these. The thought was that these factors could function as examples of how to detail the diagnosis, and sometimes, when appropriate, be used unchanged. The responses from the users confirmed this thought. Additionally, one of the users suggested that free text embellishments and preferred embellishments could be added to the list. In this way a ward could tailor the embellishments toward an “internal” standard to for instance get a more coherent detailing diagnosing within NANDA diagnoses, which are wide or not very specific. This is a functionality that is easy to add and could be implemented in the next iteration.
It is though important that irrespective what solution is chosen, the possibility of entering free text to embellish the diagnoses is maintained (Ellingsen et al. 2007).

8.2.5 The Information Window

The basis of this functionality was to assure that also the novice data user should get access to the definitions and other information regarding the standardized vocabularies. The response from the experienced users indicates that this is a functionality that also they appreciate. We were afraid that this popping up of windows beyond the users control could be seen as annoying, but none of the reactions from the test users point in this direction.

The placements of the dialog boxes and the information window are hard coded in the present prototype. The users should preferably have a chance to configure this by him self and this should be considered implemented in present iterations. Whether it should be possible to turn off windows is questionable. Experiences show that especially inexperienced computer users may turn off functionality which they later is not capable of turning on again, at the risk of missing important information (Wangensteen et al. 2005) .

8.2.6 Diagnosis and Interventions from the same Class

The harmonization of the classifications of NANDA, NIC and NOC, and also the KPO-classification, may leave the impression that it is possible to create a direct link between the nature and classification of a patient's problem and the treatment of this problem, or in other words, if you can classify the problem, you have also classified the treatment. This is far from the reality; the harmonization rather accentuates the inherent problem of classifying nursing practice. This is easily demonstrated in our system. Looking at the diagnoses for instance in the class "nutrition" and the interventions in the same class, it is easy to see that the interventions are aimed at the physiological reasons for the diagnosis. For instance will the diagnosis "Imbalanced Nutrition: less then body requirement" in most contexts require nursing interventions that also address reasons other than the physiological. The harmonized classifications should therefore be looked upon as a tool for navigation among the concepts, and not as a tool that connects the concepts from the different original classifications.

Originally, the workflow for creating an intervention under a diagnosis was as shown in figure 7-18, except that the class according to the diagnosis' class was not expanded in the blue dialog when the "Klassifikasjon…" button was clicked. In the last iteration of our system, it was proposed to reject the "Klasse…" button and the belonging dialog, down to the left in figure 7-18. Instead, this functionality should be incorporated into the classification presentation, by expanding the actual class, as shown in the blue dialog window. This is a more effective interface for fluent users, because two operations are merged into one; the user can scan the interventions in the class, and immediately be ready to browse the rest of the classification if need be. The confusion that this solution may cause for an inexperienced user, as shown in test 1, is alone a good enough reason to go back to the original workflow. Another and more serious consequence of this expansion of the class could be that new users not acquainted to the classification easily could interpret that it was preferred or even mandatory that interventions should be fetched from the same class as the diagnosis. We have therefore decided to revert to the original configuration, where the user actively has to open the class to look for suitable interventions. This makes it clearer that the interventions in the same class as the diagnosis merely constitute a group of interventions that *may* be relevant for the problem at hand, analogous to other groups that help filtering out interventions. Among other such groups are the groups of the most common interventions for a given diagnosis or the groups of the most common interventions for a nursing specialty area.

8.2.7 The layout of the care plan

During the project period at UNN, approximately 90 % of the users commented on or was dissatisfied with the layout of the care plan. They wanted to relate interventions to a diagnosis. The experiences of one of the participants in test 2 also showed that other health personnel not used to the layout of DIPS instinctively want this connection between the diagnoses and the interventions. Our belief is that patients and relatives also better will grasp the intention of the care plan with a hierarchical layout, which corresponds to the general comprehension of these concepts. In DIPS, it is a deliberate design choice to separate the NANDA diagnoses from the NIC interventions. The only explanation given for this choice is the need for flexibility, to meet situations where two diagnoses are related to one intervention and vice versa (Flø 2005). In the KPO-model, it is also explicitly stated that interventions should not relate to a diagnosis. The reason given for this is that the intervention should be selected on the basis of the individual patient's need, and not on the basis of the diagnosis, because there is not a linear connection between the diagnosis and the intervention (Krogh and Dale 2007). This is as we see it, motivated by the same issue as discussed above in chapter 8.3.6. If the users are conscious about the shortcomings of the harmonization of the classifications, we feel that this restriction is fare too rigid, and prevent the ability to see the patient's problems and treatment as a whole. In all other literature we have seen, the direct connection between the diagnoses and interventions are emphasized, see for instance (Bulechek et al. 2006, Ladwig and Ackley 2006, Wilkinson 2000).

A consequence of this design choice of DIPS is that the implementation model of the software does not correspond to the mental model of most of the users.

In our view, one of the strengths of using NANDA and NIC is that nurses is forced to reflect over the patient's problems and put them into a more broad and general category of a diagnosis, and then reflect on an abstract and high level over which interventions that is suitable to meet these problems. The task of adding concrete activities can be postponed until the end of this process. Traditionally, nurses have been used to formulate problems and then choose concrete interventions to meet these problems. From our experience at the project at UNN, we found that one of the hardest tasks for nurses learning to use NANDA and NIC was this transition from a concrete 2 step process of problem and intervention formulation to the more abstract and involved 3 step process of NANDA and NIC. The layout of the care plan, not relating the diagnoses and the interventions, counteracted this process. Users often felt that they had to delearn their mental model to relate to the interface of the care plan, and then relearn it to relate the classifications. This created a lot of confusion and frustration in the start of the learning process.

It is always a good practice to relate interventions to a defined problem. This also follows the legal requirement to give reasons to treatment. The process of creating a care plan at UNN is always founded on this principle (Wangensteen et al. 2005). The only exceptions are acute situations where a patient's condition changes rapidly so that new interventions as soon as possible should be reflected in the care plan. This is why we have decided to have the items "Frittsående tiltak" and "Frittstående forordninger" in the care plan. The intention is that later these acute entered interventions and activities should be moved under appropriate diagnoses or interventions.

Also in the daily use of DIPS, the lack of connection between the diagnoses and interventions is problematic. The only clue of which interventions belong to which diagnoses is a number relating to the functional area placed to the right of each item in the care plan. Because many diagnoses and interventions are placed in more than one functional area, and also because there may be good reasons to choose an interventions from a different functional area, analogous to the discussion in chapter 8.3.6, these numbers often do not correspond. This is creating uncertainty among personnel that for instance writes shift reports that shall reflect the content of the care plan.

The design choice done by dips also implies some serious shortcomings regarding future improvements of the EHR. Reflecting the lack of external connection between the diagnosis and the interventions is the lack of an internal connection. If one decides to offer a problem-oriented view of the nursing documentation, the diagnoses and the interventions have to be shown separately. A view where diagnoses and related interventions (and NOC outcomes) is shown together would be much more helpful, with the possibility to follow one diagnosis and different interventions and their outcomes. It is always possible to make systems that can make this connection posterior, based on time stamps and educated guessing. Such systems are inaccurate and error prone, and in our view it is better that the nurse does this linkage in real time.

The hierarchical presentation provides for the relationship of one diagnosis to one or several interventions. In the rare situations that two diagnoses demand the same intervention, the intervention can be registered under both diagnoses. Eventually, a solution where the nurse deliberately connects the intervention to the two (or more) diagnoses could be implemented, as is done in DocuLive. However we believe that the first solution is the best, because the latter is contrary to the simplicity that we desire.

8.2.8 Drag and Drop

This is a feature that is added in later versions of DIPS, so the difference between the two systems was not and issue.

The possibility to move items in relevance to their importance is important in order to make the care plan easier to read. The KPO-model recommend to place the nursing diagnoses that relate to the main ICD – 10 diagnosis of the patient at the top, followed of the diagnoses that target acute or actual problems and place the "risk" diagnoses at the bottom (Krogh and Dale 2007).

8.2.9 Free Text Diagnoses and Interventions

We have chosen to include the possibility to write diagnoses and interventions in free text. We feel that this opportunity still has to be present in an EHR that offer care plan writing in order to get acceptance from the clinical field. At the ward at UNN 10% of the diagnosis and interventions still is in free text. It may be this percentage can be lowered with a better tool to navigate the classifications. This would probably especially be obtainable regarding the NIC – classification since this is the more extensive of the two classifications. It is though clear that some of the percentage for the NANDA – classification's part is constituted of a "sub-standard" of free text diagnoses that is developed at the ward. Several of the nurses participating in test 2 wanted that is should be possible to integrate these diagnoses into the tool, so they could be selected in addition to the diagnoses from the classification. A better approach would have been to critically go through these diagnoses and, with the help of the improved searching and browsing facilities in our solution, try to find standard diagnoses that could be used in stead. With help of carefully chosen embellishments these standard diagnoses could express the same clinical picture as the free-text diagnoses they replace. The embellishments should also be incorporated into the tool.

The final goal of using a classification is to "close" the free text opportunity, and use only standard concepts. The KPO-model for instance has no room for free text diagnoses We feel that this is to early

Have to evaluate more before "closing" the classification. Tool like analyzing text sort diagnoses and interventions

8.3 The Search Functionality

Since the only complete vocabulary we have got access to is the NANDA, the search functionality is discussed with this vocabulary as a basis.

8.3.1 The Implementation of the Inverted Index

The total number of different index terms in the collection is 4703 (See INF-3982 on the accompanying CD) extracted without the use of any stopwords. This number is further decreased after stemming. Since the number of distinct index terms is very low, the inverted index is implemented as a simple sorted arraylist, and binary search is used to look up index terms. This is also the best solution if one later decides to implement searching on prefixes, where a new search is done after every character the user types in. The speed of the search could possibly be increased by using a hash table, even though this is hardly likely with a small number of index terms like in this project, but a more effective searching structure may be necessary if huge numbers of index terms are added to the collection.

The number of documents in the collection is also very low, compared to ordinary IR systems. When stopwords are eliminated, no index term will occur in more than 60 documents (See INF-3982 on the accompanying CD), and most will occur in just a few documents. The posting lists will therefore in average be very short, most will consist of 1 to a few nodes. Stemming increases the lists, but not significantly. The posting lists are therefore implemented as ordinary linked lists.

For the prototype, the search and ranking performance is more than adequate; the result of a search is displayed with no noticeable delay.

8.3.1.1 Simplifications in Implementing Ranking

To compute the dot product in the numerator in equation 2-1 we have used the following algorithm (Grossman and Lee 2001) :

for every query term q in Q **do** **(Algorithm 5-1)**
retrieve the postings list for q from the inverted index
for each document d indexed in the postings list **do**
$score(d) = score(d) + tf_{d,q} \times idf_q$
end
end

The algorithm results in an array with the score for every document that contains at least one of the query terms. Notice that only the index terms expressed in the query have to be looked up in the inverted index.

The denominator of equation 2-1 is a normalizing factor that smoothes out the difference of document length. Long documents are more likely to have high term frequency of a given term, and dividing on the vector length will counteract this. The computation of the vector length is expensive, because it requires scanning the posting lists of every term in the index, not only the query terms. The documents in the Nanda collection differ in length from 20 to over 350 terms, so some form of normalization is preferable.
For collections that changes (i.e. documents are added or deleted) often, the vector length can not be stored, because the inverted document frequency (idf) will change, leading to change of the term weight, and thereby the vector length. The NANDA vocabulary changes very infrequently, diagnoses may be added or revised only after each Nanda-conference that are held every second year, so it would be possible to store the vector length. However, if one wishes to manually or automatically add index terms to the diagnosis, in order to improve the searching process, the vector length would have to be recalculated for every index term added. A method to approximate normalization is to use the square root of the number of terms in the document as a normalization factor. This approach has shown to give the same or in some cases even better precision compared to using the vector length (Lee et al. 1997). Since this factor can be stored, and easily updated in case of adding terms to a document, this is the approach we have chosen. The dictionary `docLengths`, an attribute to the `InvertedIndex` class, is used to hold the square roots of the document lengths. The key value into the dictionary `docLengths` is of course the code for the document. For the small NANDA vocabulary, implementing functionality to enable adding one term at the time may not be worthwhile, after all it takes only around 1 to 2 second to

rebuild the whole inverted index. But for the NIC vocabulary, which contains roughly estimated around 64 times as many words as the NANDA vocabulary, this may become an issue.

The scores of the different documents in the resulting array of algorithm (5-1) is divided by its respective normalization factor, and a sorting of the array according to the normalized scores gives a ranked result.

Another simplification made is that the query terms are not weighted before processed. Thus equation 2-5 in chapter 2.11.3 is reduced to $w_{i,q} = tf_{i,q} \times \log \frac{N}{n_i}$. This can be justified width the very little probability that a search key-word should be repeated the way the system is intended to be used in this thesis.

8.3.1.2 Thesaurus

We have not included a thesaurus in our solution. Because of the sparse wording of the vocabulary, a good thesaurus would have been of great value to increase recall. We have however not found a good enough open source or free Norwegian thesaurus. We have looked into OpenThesaurus (OpenThesaurus 2007), a Norwegian thesaurus meant to be used in OpenOffice.org-2.0 (OpenOffice 2007). It is gradually built up by volunteers that add terms and synonyms. The thesaurus is delivered as a text file with a structure that is easy to parse. It is licensed under the GNU General Public Licenses (GNU 2007). Some trial searches was not promising, there were few hits. Because of the domain specific terminology used in NANDA, this was not unexpected.

8.3.2 Evaluation

The theoretical reasoning and statistical term weighting technique that is the foundation for the equations in chapter 2.11 is based on retrieval of documents within huge collections, typically consisting of ten thousands to millions of documents. The documents used have also been relatively long and containing general text. Traditionally, testing of the retrieval performance of general IR systems has been done using huge reference collections and predefined queries (Baeza-Yates and Ribeiro-Neto 1999). This is definitely not possible for our solution, consisting of 155 very short documents with a very special terminology, so other test approaches must be employed. For small IR systems like the present, an empirical approach using real users and the whole collection is feasible (Draper 1995). One possibility would be to log events while the system was used in a real environment. Events to log could be key-words and their related hits and key-words that did not result in a hit. The logs could then be used to improve the system.

A thorough testing of the search functionality is beyond the possibility of this project. We will still present some observations and thoughts about its performance on the basis of the results presented in chapter 7.4. The coloring of the keywords in the information window that is presented after a search, points out which words that is the basis for the ranking, and also their placement in the diagnoses' field. This has been helpful to evaluate the ranking functionality.

8.3.2.1 Retrieving performance

As shown in chapter 7.4, there is no guaranty that all relevant documents are retrieved. The easiest way to increase the recall is to add a thesaurus to the system. This will also decrease precision, but as long as the best hits are ranked high, this is not a problem. By looking through the highest ranked diagnoses, the user can find the relevant one(s).

Another possibility, since the document collection is small, is to manually add key-words to the diagnoses. If synonyms directly related to the text within the diagnosis is added, and possible also additional key-words which can help search up the diagnosis, a recall of nearly 1 with no significant increase in precision could be achieved. By for instance using a consensus group of nurses, related key-words could be added. Logs as mentioned above could also be used to decide the most frequent used search words that did not result in a hit, and thus the need to add these to relevant diagnoses.

If the search facility is going to be used as a differential diagnosing tool it is especially important that the recall is as good as possible.

In order to improve recall, we have manually browsed through the around 4000 words in the classification and split the compound words. The word resulting from this is put into the AddedWords field in the database. For convenience, and to save time, the structural weighting of the original word was not taken into account for the resulting words.

One of the experiences of using the tool is that compound key-words may hard to cope with. The automatic splitting of compound words is a challenge, and also error prone (Koehn and Knight 2003) . Still some form of splitting of compound words in the query would be desirable. A solution where a searching is conducted after each letter entered would at least solve this problem for the first word in compound key-words.

8.3.2.2 Ranking

The normalization of term frequency in equation 2-2 in chapter 2.11.3 did counteract the structure weighting we have implemented, so in this solution we have used the raw frequency to calculate the term weights. Equation 2-2 is therefore reduced to $f_{i,j} = tf_{i,j}$, and this is also reflected in algorithm 5-1 in chapter 8.4.1.1.

In an earlier iteration the weight of the words in the label of the diagnoses were sat to 3. With this value, many diagnoses that are aimed against signs and symptoms that are typical in nursing, such as Anxiety and Pain, was ranked to low, because these words were used repeatedly in many other diagnoses. We have therefore increased this value to 5 in the present version, and this seems to give a more appropriate ranking.

While searching with more than one key-word, the vector model weight each word equally, that is, diagnoses where one of the word has a high weight will be ranked higher than diagnoses where the combined weight of the words are lower. This is not what is desired; a nurse searching for two or more symptoms is probably most interested in diagnoses containing both. A mechanism to give more weight to queries containing multiple words should therefore be tested out in future version.

8.3.2.3 The Lacking Maturity of the NANDA Classification

Another factor that influences the searching performance is the lacking maturity of the classification. A few diagnoses consist only of the label and the definition, and quite a few is lacking one or two of the remaining fields. The degree of effort worthwhile to put into the improvement of the search facility must also be evaluated in this respect. It is OK to fine-tune the search facility, but it cannot be better than the classification allows.

9 Conclusion

By using DIPS as a staring point, we have succeeded in making a system that has proven equal to or better than the original in all areas defined in the problem statement. With some additional features and fine-tuning a system like ours should make it possible for users with only minimal knowledge of NANDA and NIC to start writing care plans using these languages, and extend their knowledge during practical usage. A consequence of this would be a faster introduction of the NANDA and NIC. Improved browsing- and search functionality would also raise the quality of the evaluation of the usefulness and completeness of the languages, by allowing a much more thorough and efficient exploration of the classifications than is possible in existing systems.

The search functionality presented in this project, implemented as a “miniature” information retrieval system, has even in its current version wich does not include thesaurus shown very valuable to find diagnoses based on key-word search. Further adjustments remain to improve recall and ranking, and a formal test to state its retrieval performance has to be conducted. We have not found other projects that have used the IR approach to search classification systems. The concept should though easily be applicable to other classification systems, also outside the nursing domain.

Hospitals and vendors of EHRs have traditionally, with good reason, been concerned with hard requirements as reliability, integratability and scalability. Now that large groups of health personnel are starting to use EHRs, our hope is that more of their attention should be focused on usability.

10 References

Andreassen, D., Overaa, M. A. and Myrvang, T. (2003) Connect IT. Prosjektoppgave. Bachelor - IT utdanningen, Høgskolen i Bodø.

Baeza-Yates, R. and Ribeiro-Neto, B. (1999) Modern information retrieval, ACM Press, New York.

Booch, G., Jacobson, I. and Rumbaugh, J. (1999) The unified modeling language user guide, Addison-Wesley, Reading, Mass.

Bowker, G. C. (2003) In Unifying nursing languages : the harmonization of NANDA, NIC, and NOC(Ed, McCloskey, J. a. J., Dorothy A.) American Nurses Publishing, Washington, D.C., pp. 132 s.

Bowker, G. C. and Star, S. L. (1999) Sorting things out : classification and its consequences, MIT Press, Cambridge, Mass.

Bowker, G. C., Star, S. L. and Spasser, M. A. (2001) Classifying Nursing Work. Online Journal of Issues in Nursing. http://www.nursingworld.org/ojin/tpc7/tpc7_6.htm.

Bulechek, G. M., Dochterman, J. M. and Iowa Intervention Project (2006) Klassifikasjon av sykepleieintervensjoner (NIC), Akribe AS, Oslo.

Børmark, S. R. (2007) Elektronisk sykepleidokumentasjon, utfordringer og ansvar. NSF's IKTkonferanse, Oslo. http://www.v-chi.dk/shi2005/pdf/7B/SidselBormark.pdf.

Cockburn, A. (1999) Writing Effective Use Cases, in preparation for Addison-Wesley Longman. www.infor.uva.es/~mlaguna/is2/materiales/BookDraft1.pdf

Coiera, E. (2003) Guide to health informatics, Arnold, London.

Cooper, A. (1999) The inmates are running the asylum, Sams, Indianapolis, Ind.

Cooper, A. and Reimann, R. (2003) About Face 2.0 : the essentials of interaction design, Wiley, Indianapolis, Ind.

Davidson, M. J., Dove, L. and Weltz, J. (1999) Mental Models and Usability. Depaul University, Cognitive Psychology. http://www.lauradove.info/reports/mental%20models.htm

DIPS (2007) DIPS ASA, Electronic Health Record Vendor. http://www.dips.no/dipsnew.nsf/Display/Startside.

Draper, S. (1995) Brief ideas about IR evaluation. http://www.psy.gla.ac.uk/~steve/IReval.html.

DUKE (2007) Standardized Nursing Languages http://www.duke.edu/~goodw010/vocab/index.html.

Dumas, J. S. and Redish, J. C. (1999) A practical guide to usability testing, Intellect, Exeter.

Ehnfors, M., Ehrenberg, A. and Thorell-Ekstrand, I. (2000) VIPS-boken
om en forskningsbaserad modell för dokumentation av omvårdnad i patientjournalen, Vårdförbundet, Stockholm.

Ellingsen, G., Monteiro, E. and Munkvold, G. (2007) Standardisation of work: co-constructed practice. The Information Society, In Press.

Evolve (2007) Evolve Care Plan Constructor. Website. http://www1.us.elsevierhealth.com/Evolve/Ackley/NDH6e/Constructor/#New.

Flø, K. (2005) Peronal conversation.

Frakes, W. B. and Baeza-Yates, R. (1992) Information Retrieval: Data Structures & Algorithms, Prentice Hall, Englewood Cliffs, N.J.

Glomsås, H. S. (2003) ICNP ikke god nok for bruk i klinisk praksis. Sykepleien http://www.sykepleien.no/article.php?articleID=859.
GNU (2007) GNU General Public License. http://www.gnu.org/copyleft/gpl.html.
Grossman, D. A. and Frieder, O. (2004) Information Retrieval: Algorithms and Heuristics, Kluwer, Boston.
Grossman, D. A. and Lee, M. (2001) Vector Space Model, Handout from IR course. http://ir.iit.edu/~dagr/cs529/files/handouts/03VectorSpaceImplementation-6per.PDF.
Halvorsen, K. H. (2007) Personal communication May 2007.
Hansen, T. B., Hjertø, G. and TISIP (2003) Kvalitet og programvareutvikling, TISIP
Gyldendal akademisk, [Trondheim]
Oslo.
Hazelwood, A. (2001) CSharp Porterstemmer, http://www.tartarus.org/~martin/PorterStemmer/csharp.txt.
Hersh, W. R. (2003) Information Retrieval : A Health and Biomedical Perspective, Springer, New York.
HOD (2001) Lov om helsepersonell m.v. (helsepersonelloven). , Helse- og omsorgsdepartementet.
ibid.
IBM (2001) Rational Unified Process. Best Practices for Software Development Teams. Rational Software White Paper, TP026B, Rev 11/01 http://www-128.ibm.com/developerworks/rational/library/content/03July/1000/1251/1251_bestpractices_TP026B.pdf.
ICN (2007) ICNP. http://www.icn.ch/icnp.htm.
Jacobson, I. (1992) Object-oriented software engineering : a use case driven approach, Addison-Wesley, Wokingham.
Jones, D. A. and Dochterman, J. M. (2003) Unifying nursing languages : the harmonization of NANDA, NIC, and NOC, American Nurses Publishing, Washington, D.C.
Keenan, G. M., Stocker, J. R., Geo-Thomas, A. T., Soparkar, N. R., Barkauskas, V. H. and Lee, J. L. (2002) The HANDS project: studying and refining the automated collection of a cross-setting clinical data set. Comput Inform Nurs, 20(3): 89-100.
KITH (2003) Veileder for elektronisk dokumentasjon av sykepleie. R 14/03.
Koehn, P. and Knight, K. (2003) In Proceedings of the tenth conference on European chapter of the Association for Computational Linguistics Vol. 1 Budapest, Hungary
Krogh, G. v. and Dale, C. (2007) KPO-modellen. Kvalitetssikring - Problemløsning - Omsorg. En teoribasert modell for dokumentasjon av sykepleie, Akribe AS, Oslo.
Krogh, G. v., Dale, C. and Nåden, D. (2005) A Framework for Integrating NANDA, NIC, and NOC Terminology in Electronic Patient Records. Journal of Nursing Scholarship, 37 (3): 275–281.
Ladwig, G. B. and Ackley, B. J. (2006) Nursing Diagnosis Handbook : A Guide to Planning Care, Mosby, St. Lois.
Larman, C. (2002) Applying UML and patterns : an introduction to object-oriented analysis and design and the unified process, Prentice Hall PTR, Upper Saddle River, N.J.
Lee, D. L., Chuang, H. and Seamons, K. (1997) Document Ranking and the Vector-Space Model. Software, IEEE, 14(2): 67-75
Lidor (2007) IntegralUI TreeView. http://www.lidorsystems.com/.
Lunney, M. (2003) In Unifying Nursing Languages. The Harmonization of NANDA, NIC, and NOC.(Ed, McCloskey, J. a. J., Dorothy A.) nursesbooks.org.

Manning, C. D., Raghavan, P. and Schütze, H. (2007) An Introduction to Information Retrieval. Preliminary draft., Cambridge University Press.
Merriam-Webster (2007) Merriam-Webster Online Dictionary, http://mw1.merriam-webster.com/dictionary.
Microsoft (2003) Windiff Overview http://technet2.microsoft.com/windowsserver/en/library/ac1ff104-dde3-43ea-98af-e5cf5e513d4f1033.mspx?mfr=true.
Microsoft (2007a) Microsoft Visual C# 2005 Express Edition. http://msdn.microsoft.com/vstudio/express/visualcsharp/default.aspx.
Microsoft (2007b) SQL Server 2005 Express Edition. http://www.microsoft.com/sql/editions/express/default.mspx.
Moen, A., Hellesø, R., Quivey, M. and Berge, A. (2002) Dokumentasjon og informasjonshåndtering : faglige og juridiske utfordringer og krav til journalføring for sykepleiere, Akribe, Oslo.
MSDN (2007) Using IntelliSense. http://msdn2.microsoft.com/en-us/library/hcw1s69b(vs.71).aspx.
NANDA (2003) Norsk redaksjonsutvalg for klassifikasjonssystemene NANDA NIC og NOC. NANDA sykepleiediagnoser : definisjoner & klassifikasjon, 2001-2002, Akribe, [Oslo].
Nielsen, J. (1993) Usability engineering, Academic Press, Boston, Mass.
Nielsen, J. (2000) Why You Only Need to Test With 5 Users. http://www.useit.com/alertbox/20000319.html.
Norris, A. C. (2002) Essentials of telemedicine and telecare, John Wiley, Chichester.
NSF (2007) Dokumentasjon av sykepleie i elektronisk pasientjournal. En veileder fra Norsk Sykepleierforbunds forum for IKT og Dokumentasjon. http://www.sykepleierforbundet.no/getfile.php/www.sykepleierforbundet.no/Faggrupper/IKT%20og%20dokumentasjon/Publisert%202007/Veileder%20%20Sykepleiedokumentasjon.pdf.
OpenOffice (2007) http://www.openoffice.org/.
OpenThesaurus (2007) OpenThesaurus - Synonymordliste for norsk bokmål. http://synonymer.merg.net/.
Perreault, L. E. and Shortliffe, E. H. (2001) Medical informatics : Computer Applications in Health Care and Biomedicine, Springer, New York.
Porter, M. (2007) Norwegian stemming algorithm, http://snowball.tartarus.org/algorithms/norwegian/stemmer.html.
Rees, R. v. (2002) Clarity in the Usage of the Terms Ontology, Taxonomy and Classeification http://vanrees.org/research/papers/2003_cib.pdf.
Salton, G. and Buckley, C. (1988) Term Weighting Approaches in Automatic Text Retrieval. Information Processing & Management, 24(5): 513-523.
Sanum (2007) Sanum. Website. http://sanum.no/site/sanumpleieplan.asp.
Siemens (2007) Medical Solutions. http://www.siemens.com/index.jsp?sdc_p=dpo1405557c132l70s2&.
SOH (2000) Forskrifter for pasientjournal (forskrift 2000-12-21 nr. 1358) Oslo, Sosial- og helsedepartementet. http://www.lovdata.no/for/sf/ho/to-20001221-1385-0.html.
UI (2007) NIC Intervention Example Website. University of Iowa. http://www.nursing.uiowa.edu/about_us/nursing_knowledge/clinical_effectiveness/nicintervention.htm.

Vedal, T. (2006) Bruk av standardisert begreper i elektronisk pasientjournal. Innlegg på HelseIT 2006. http://www.kith.no/upload/3104/torun_vedal.pdf.
Wangensteen, G. (2007) Da NIC og NOC kom til alderspsykiatrisk post. Sykepleien(01/07): 54-55.
Wangensteen, G., Igesund, H. and Dybdal, L. (2005) Elektronisk sykepleiedokumentasjon. http://unn.no/category11302.html.
WebONT (2003) Web Ontology (WebONT) Working Group Charter http://www.w3.org/2002/11/swv2/charters/WebOntologyCharter.
Wikipeda (2007a) Nursing Care Plan. http://en.wikipedia.org/wiki/Nursing_care_plan.
Wikipeda (2007b) Nursing Process. http://en.wikipedia.org/wiki/Nursing_process.
Wikipedia (2007) Document Retrieval. http://en.wikipedia.org/wiki/Document_retrieval.
Wilkinson, J. M. (2000) Nursing Diagnosis Handbook, Prentice-Hall, Upper Saddle River, N.J.

Appendixes

Appendix 1

TABLE 2-6 Taxonomy of Nursing Practice

Domains			
I. Functional Domain Includes diagnoses, outcomes, and interventions to promote basic needs.	**II. Physiological Domain** Includes diagnoses, outcomes, and interventions to promote optimal biophysical health.	**III. Psychosocial Domain** Includes diagnoses, outcomes, and interventions to promote optimal mental and emotional health and social functioning.	**IV. Environmental Domain** Includes diagnoses, outcomes, and interventions to promote and protect the environmental health and safety of individuals, systems, and communities.
Classes includes diagnoses, class outcomes, and interventions that pertain to:			
Activity/Exercise—Physical activity, including energy conservation and expenditure.	**Cardiac Function**—Cardiac mechanisms used to maintain tissue profusion.	**Behavior**—Actions that promote, maintain, or restore health.	**Health Care System**—Social, political, and economic structures and processes for the delivery of healthcare services.
Comfort—A sense of emotional, physical, and spiritual well-being and relative freedom from distress.	**Elimination**—Processes related to secretion and excretion of body wastes.	**Communication**—Receiving, interpreting, and expressing spoken, written, and nonverbal messages.	**Populations**—Aggregates of individuals, or communities having characteristics in common.
Growth and Development—Physical, emotional, and social growth and development milestones.	**Fluid and Electrolyte**—Regulation of fluid/electrolytes and acid base balance.	**Coping**—Adjusting or adapting to stressful events.	**Risk Management**—Avoidance or control of identifiable health threats.
Nutrition—Processes related to taking in, assimilating, and using nutrients.	**Neurocognition**—Mechanisms related to the nervous system and neurocognitive functioning, including memory, thinking, and judgment.	**Emotional**—A mental state or feeling that may influence perceptions of the world.	
Self-Care—Ability to accomplish basic and instrumental activities of daily living.	**Pharmacological Function**—Effects (therapeutic and adverse) of medications or drugs and other pharmacologically active products.	**Knowledge**—Understanding and skill in applying information to promote, maintain, and restore health.	
Sexuality—Maintenance or modification of sexual identity and patterns.	**Physical Regulation**—Body temperature, endocrine, and immune system responses to regulate cellular processes.	**Roles/Relationships**—Maintenance and/or modification of expected social behaviors and emotional connectedness with others.	
Sleep/Rest—The quantity and quality of sleep, rest, and relaxation patterns.	**Reproduction**—Processes related to human procreation and birth.	**Self-Perception**—Awareness of one's body and personal identity.	
Values/Beliefs—Ideas, goals, perceptions, spiritual, and other beliefs that influence choices or decisions.	**Respiratory Function**—Ventilation adequate to maintain arterial blood gases within normal limits.		
	Sensation/Perception—Intake and interpretation of information through the senses, including seeing, hearing, touching, tasting, and smelling.		
	Tissue Integrity—Skin and mucous membrane protection to support secretion, excretion, and healing.		

This structure is in the public domain and can be freely used without permission; neither the structure nor a modification can be copyrighted by any person, group, or organization; any use of the structure should acknowledge the source.
The papers in the monograph are copyrighted by the authors, and permission to use parts of the papers should be sent to the authors.

Figure 19 Proposed Taxonomy of Nursing Practice, from (Jones and Dochterman 2003)

Appendix 2

CP = Care Plan, vår løsning, KPO = Kvalitet - Problemløsning - Omsorg

For testene ble startprogram byttet for halvparten av deltakerne

Utility test for nybegynnere

Innledningsvis: Orientering om testens innhold og formål

DIPS
Ny diagnose
Finn fram til diagnosen Funksjonsområde = Sosialt, planlegging av utskrivelse -> Diagnose = Risiko for ensomhet
Skriv inn risikofaktor: Sosial isolasjon

Endre risikofaktoren til Manglende hengivenhet

CP
Ny diagnose
Finn fram til diagnosen Funksjonell - Sosial funksjon - Risiko for ensomhet
Velg eller skriv inn risikofaktor: Sosial isolasjon

Endre risikofaktoren til Manglende hengivenhet

I etterkant: Ustrukturert samtale om hvilke inntrykk testpersonen har av de to løsningene, spesielle ting de liker eller misliker.

Vurdering av CP av sykepleiere som er vant med DIPS og pleieplanskriving

Ta utgangspunkt i symptomene angst, feber, urinveisinfeksjon.

Gi en demonstrasjon av vårt system:
Bruk KPO til å finne diagnoser som samsvarer med disse symptomene.
Bruk søkefunksjonen til å finne diagnoser som samsvarer med disse symptomene.
Lage noen tiltak forordninger.

Semi - strukturert intervju:
Vurdere arbeidsflyt, lett å finne fram?
Vurdere KPO, naturlig hierarki, lett å finne fram, mind-map?.
Vurdere søkefunksjonen
Vurdere relaterte faktorer/ risikofaktorer hentet fra standardisert språk.
Vurdere visning av informasjon, irriterende/nyttig?
Vurdere forordninger knyttet til klasse, dekker det alle behov eller er det for snevert?

Vurdere layout på pleieplanen. (Diagnose – tiltak – forordning) Bedre og mer hensiktsmessig enn i DIPS?
Vurdere dra og slipp.

Test:
Sette opp en pleieplan med 2 diagnoser, 1 tiltak og 2 forordninger. Bruk diagnosene Urge urininkontinens og Mangelfull egenomsorg; toalett.
Tiden tas i DIPS og for CP