IT support for clinical pathways—Lessons learned

R. Lenza\textsuperscript{a,∗}, R. Blaser\textsuperscript{b}, M. Beyer\textsuperscript{b}, O. Heger\textsuperscript{b}, C. Biber\textsuperscript{c}, M. B"aumlein\textsuperscript{c}, M. Schnabel\textsuperscript{c}

\textsuperscript{a}Department of Computer Sciences, University of Erlangen and Nuremberg, Germany
\textsuperscript{b}Institute of Medical Informatics, Philipps-University Marburg, Germany
\textsuperscript{c}Department of Trauma, Reconstructive and Hand Surgery, University Hospital, Giessen and Marburg, Location Marburg, Germany

\textbf{Article info}

Article history:
Received 21 December 2006
Received in revised form
26 March 2007
Accepted 23 April 2007

Keywords:
Information systems [MeSH]
Clinical pathways [MeSH]
Process alignment
Interaction design
Speech act theory

\textbf{Abstract}

Clinical pathways are an effective instrument to decrease undesired practice variability and improve clinician performance. IT-applications embedded into clinical routine work can help to increase pathway compliance. Successfully implementing such applications requires a responsive IT infrastructure and a participatory and iterative design process aimed at achieving user acceptance and usability. Experiences from the implementation and iterative improvement of an online surgical pathway at Marburg University Medical Centre have shown that pathway conformance actually could be improved by the use of IT. An analysis of the iterative design process has shown that future pathway projects can benefit from the lessons learned during this project. Based on these lessons recommendations for developing well adapted interaction mechanisms are presented, aimed at improving process alignment. Our goal is to build up a library of tested reusable components to reduce the number of iterations for pathway implementation.

© 2007 Elsevier Ireland Ltd. All rights reserved.

1. Introduction

There is an increasing consensus among healthcare experts that information technology (IT) can significantly contribute to improve healthcare quality and reduce costs, by timely providing patient information and relevant medical knowledge at the point of care [1,2]. Evidence based medicine (EBM) is aimed at integrating the best available evidence with clinical expertise and patient values [3]. Following the principles of EBM physicians are required to formulate questions based on patients’ problems, search the literature for answers, sort the wheat from the chaff with regard to study validity, then apply the information to patients [4]. Yet, searching and evaluating current evidence is nearly impossible to practice in everyday clinical care [5,6]. Medical guidelines are aimed at supporting clinicians in interpreting existing evidence by providing recommendations for decision making based on literature reviews and existing evidence. Guidelines are aimed at enabling the physician to make informed decisions, rather than establishing a “cookbook medicine” as many fear [5,7]. The goal is that physicians know what they are doing when they individually estimate patients’ chances and risks. Yet, there is a gap between the information contained in published clinical practice guidelines and the knowledge and information that are necessary to implement them [8,9]. Methods for closing this gap by using information technology have been in the focus of medical informatics research for decades (e.g. [9–11]).

Clinical pathways can be used to implement guidelines in a specific setting and reduce undesired practice variability [12]. In contrast to guidelines, pathways consider available resources like staff, level of education, available equipment, and hospital topology, and they typically also include a time component. While guidelines need to be consented among medical experts, clinical pathways require a consensus among different groups of hospital personnel involved in the patient treatment of patients.
treatment process. Thus, clinical pathways are planned process patterns aimed at improving both process quality and resource usage. Note, that a clinical pathway may deviate from a corresponding guideline (e.g. the specific hospital might not have the resources to follow a recommended procedure), and an individual treatment plan may again deviate from an existing pathway (e.g. due to some contraindication) [13].

Clinical pathways do not necessarily need any IT support but IT-applications can increase pathway compliance, as studies have shown the potential benefit of computerized decision support based on alerts and reminders [14–16]. To effectively improve clinical processes by IT-applications, however, their embedding in routine work practice is of paramount importance. Therefore, an IT-application that supports a clinical pathway should be integrated with routine documentation and with the hospitals electronic health record system in order to avoid redundant data entries. This, however, is only one factor contributing to successful process alignment. It is a well known fact that many IT projects fail and that multiple factors need to be considered to increase the probability for success [17,18]. Clinical systems and decision support systems depend on even more success factors than other systems [19]. A core reason is that an IT intervention in a healthcare setting will necessarily change a complex socio-technical system with often unpredictable results [20]. Consequently, IT projects in healthcare settings should be seen from a broad socio-technical perspective and should be accompanied with careful change management [21–24].

As the overall goal is process improvement, IT projects should be driven by demand rather than technology. Demand-driven system evolution requires a responsive IT infrastructure optimized for adaptation to changing requirements. At Marburg University Medical Centre we tried to approximate this goal by an extensible holistic Hospital Information System (HIS) [25–27]. Responsiveness is achieved through an integrated Rapid Application Development (RAD) tool [27,28] and agile programming techniques [29] with close end user involvement [26]. An iterative and participatory software engineering process has been developed to support continuous system improvement and process alignment [21,26].

In Ref. [30] we described our approach of utilising this infrastructure to implement a guideline-based clinical pathway for patients with proximal femoral fracture at the Department of Trauma, Reconstructive and Hand Surgery. The basic idea was to support major parts of the pathway by integrating patient-specific advice according to pathway recommendations into routine documentation. In this paper, we will have a closer look at the lessons learned during the iterative development of the pathway application and draw conclusions for future pathway projects.

2. Background and objectives

The clinical pathway for patients with proximal femoral fracture was systematically developed based on the results from a prospective study of the care process in 2001/2002 and it was initially introduced without additional IT support in 2003/2004 [30]. As a result, process management could be improved significantly. However, evaluation of the pathway documentation also showed that important information, like the side of a fracture or medication details, was still often missing. In order to further increase pathway compliance and improve documentation quality, an adapted IT-application was developed in 2004 which was aimed at bringing pathway-conformant recommendations to the point of care by reusing online routine documentation. As described in Ref. [30] the application is closely integrated into the hospitals electronic patient record system. It is based on workflow-enabled electronic forms, in which coded data from a central database are reused to place reminders and alerts or to automatically parameterise order sets. The pathway application can be viewed as a set of electronic checklists for pathway documentation. In order to decrease documentation overhead and to improve user acceptance the pathway application makes use of the principle “charting by exception” as proposed by Short [31]. The idea is to minimize documentation time by only recording deviations from the pathway (variance documentation). In Ref. [30] we already described that pathway conformance could be improved by iteratively developing an IT-application for pathway documentation. A number of iterations was required to correct “intuitive” but counterproductive first attempts. The objective of this paper is to summarise our experiences made during the iterative software development for clinical pathway documentation and to draw conclusions for future pathway projects. The goal is to derive a set of generally applicable rules or recommendations for developing online support for clinical pathways, which may help to reduce the number of iterations and improve process alignment from the beginning.

3. Methods

According to our adapted software engineering process in Ref. [26] end users were intensively involved in the development of the IT-application. In addition, application development was embedded in an overall change management process in order to improve integration with the clinical workflow [21]. Thereby, user participation was seen as a dialogue where both users and software engineers learned from each other. User participation covered academic detailing for effective training of clinicians during clinical routine, qualitative user surveys and usability analyses, and continuous pathway controlling.

Systematic pathway controlling included continuous monitoring of the usage rate of the clinical pathway for patients with proximal femoral fracture, deliberate pathway deviation (variance documentation), and indicators for process quality (e.g. preoperative length of stay, time to first thromboembolism prophylaxis) and documentation quality (e.g. completeness and specificity of medication information).

In order to detect and analyse problems resulting from the IT intervention, controlling was expanded for a “methodical controlling”: User feedback was evaluated and pathway documentation was compared with the actual treatment process to find out whether there were discrepancies

- between the documentation intended by the user and the actually documented data (e.g. do users unwillingly take over preselected default values?), or
between the pathway documentation and the actual treatment process (e.g. did documented process steps actually take place?).

4. Results

The application as described in Ref. [30] is in routine use since August 2004 with several updates in 2005. Process management and documentation quality already could be improved by early versions of the application (e.g. reduced time to first thromboembolism prophylaxis, consistent documentation of the side of a fracture).

Yet, several remaining quality problems could be detected by pathway controlling, causing additional design iterations. Statements about how to continue previous outpatient medication of the patient were given in most admission reports (in 97.7% (n=77) of the admission reports compared to 72.9% (n=85) before IT intervention) but only a small part of these data contained enough detailed information on drug, dose, and time of intake (in 29.9% (n=77) of the admission reports compared to 51.8% (n=85) before IT intervention). Furthermore, there were several discrepancies between pathway documentation and the actual treatment process. Examples are:

- 5000 units of Heparin, as recommended by the pathway, were frequently documented but actually not applied,
- the recommended number of blood bottles was documented but not ordered,
- a bladder catheter was documented but not inserted.

The problems observed were carefully analysed and usually multiple factors contributing to a problem were detected. Some of the most influential factors concerning application design were:

- Insufficient differentiation between different tasks: for example, documentation of previous outpatient medication and instructions for further medication on the ward were insufficiently separated.
- Inappropriate usage of default values: for example, taking over default values without any kind of explicit commitment led to unintended documentations.
- Insufficient differentiation between different kinds of documentation: for example, prospective and retrospective documentation of activities were not distinguished.
- Layout and wording: for example, ambiguous instructions or complex screens.

Other factors beyond application design were observed to contribute to success and failure. In particular, training and academic detailing were found to be extremely important. For example, we observed that newly introduced process steps (e.g. application of Heparin during initial treatment) were in danger of being documented but not performed. The
staff tends to continue its routine tasks as before. Introducing an IT-application that recommends a different behaviour according to a new pathway is only effective if it is accompanied by extensive academic detailing: physicians and nurses must know what they are doing and they must be convinced that they are doing the right thing. The IT-application can only support them but not enforce a new behaviour. In this example, the IT-application did not succeed in establishing the newly recommended behaviour, since the new recommendation of applying 5000 units of Heparin during initial treatment was a preselected pathway recommendation which needed no explicit commitment.

Another example is the ordering of blood bottles: the initial IT-application contained a preselected button for ordering blood bottles. An associated prefilled data field contained the recommended number of blood bottles. If a different number of bottles was ordered this field had to be corrected and an additional documentation of the pathway deviation became necessary. To an untrained user it remained unclear whether this button and the associated data field were already an order or just the documentation of an independently placed order. In fact, the actual order had to be placed independently by a conventional paper form. As a consequence the number of ordered bottles had to be documented twice—a programmed source for inconsistencies.

To improve the pathway application, the user interface was modified:

- Data input for documentation of previous medication was explicitly separated from the medication instructions for the ward.
- Presenting medication recommendations by preselected choices was replaced by clearly highlighting the recommended choices.
- Preselection was avoided in order to reduce the risk of unintentionally taking over default values. To keep the documentation overload low an additional button was introduced for explicit commitment of recommended default values (see Fig. 1).
- The wording of the instructions for the ward was clarified (e.g. responsibilities for ordering blood bottles were clarified by improving instructions).

Continuous pathway controlling showed that the modified application actually did improve pathway conformance and documentation quality. Evaluation of medication documentation shows a considerably higher rate of detailed information on how to continue previous medication (in 60.3% of the admission reports; n = 58). Fig. 2 summarises the results from the different implementation phases.

5. Conclusions

Our philosophy of closely involving end users in application development helped to develop a well accepted pathway application that actually did improve pathway compliance in many ways. Yet, we are convinced that the number of iterative improvements and the time to create such an application can be substantially reduced if we carefully analyse why additional iterations became necessary. Subsequently, we summarise some general recommendations for interaction design based on our project experiences. We then identify a common pitfall in the previously reported anecdotic incidents by applying speech act theory to IT-supported clinical pathways.

Many of our iterative improvements were related to interaction design. Good interaction design can greatly influence usability of an application and also speed up documentation. Lessons learned in this context were:

- General recommendations for clearness: forms should not be overloaded, core messages should be highlighted, scrolling should be avoided, dynamic build-up of forms should be used sparingly. Such general advice can also be found in the literature (for specific recommendations see e.g. [32], or for more general advice see [33]). Our participatory design process is aimed at considering user feedback as far as possible, but for interaction design professional knowledge of IT developers is required, as users might not know which options are possible and what side effects they could cause. Design decisions should be made by experienced developers according to the users’ goals and led by their feedback.
- Signalise pathway recommendations: documentation forms should signalise pathway-conformant decisions in an unobtrusive way. Physicians should easily recognize when pathway recommendations are not followed and when variance documentation is needed. Follow the principle “give feedback to the user”, so that he can see what he has already done and what still needs to be done. Application layout should clearly separate recommendations from decisions.
- Carefully use default selections: some of the negative effects we initially had to deal with resulted from our intention to reduce documentation efforts as far as possible. We still believe that this is very important for pathway applications to achieve user acceptance. Yet, default selections increase the risk of unconsciously documenting process steps that did not take place. Therefore, additional effort for data entry and documentation quality have to be balanced carefully.

A core observation is that the actual patient treatment process and the online application for process documentation are necessarily closely intertwined (e.g. [34]). Process improvement is much more likely to be achieved if application development is embedded into a comprehensive change management process (cf. [21–24]). Consequently, IT-applications for pathway documentation should not only be viewed as accompanying add-ons. Instead, an online documentation should be viewed as a set of speech acts which are part of the pro-

![Fig. 2 - Completeness and quality of medication information.](Image)
cess and actually affect the process. Speech act theory, going back to Austin [35] and Searle [36], is based on the assumption that language is not only used to formulate statements (which can be true or false), but also to perform acts, which can succeed or fail and thereby influence other acts of the communicating parties. Different speech acts, such as “declaration”, “recommendation”, “order”, or “promise”, have characteristic communication patterns that determine how different communicating partners (roles within a speech act pattern) act in order to “succeed”. If we analyse the additional iterations for software adaptation that were necessary to optimize our pathway application it becomes apparent that many pitfalls have one thing in common: different speech acts were not distinguished sufficiently when the pathway application was initially designed. Thus, a consequence is to consider speech act theory from the beginning during the software engineering process in order to avoid unnecessary software adaptations.

The applicability of speech act theory to software engineering has been increasingly recognized (e.g. [37]). Formal representation of speech acts in message interchange formats such as EDI has been proposed [38]. In HL7 Version 3 speech acts are formally represented by assigning a so-called “Mood Code” to an act [39]. Communication patterns can also be found in software integration standards in the healthcare domain: the IHE\(^1\) integration profiles [40] describe interaction patterns with formally defined roles. These approaches are aimed at supporting interoperation of different software components according to some defined action plan. Our focus, however, is on improving process alignment. Formally representing speech acts is not necessary for that—instead, we propose to actually consider typical speech act patterns during the software engineering process in order to effectively optimize not only a piece of software, but a complex socio-technical system. Prominent software engineering models such as Extreme Programming (XP) [29] or the Rational Unified Process (RUP) [41,42] typically do not mention speech act theory, probably simply because intuitive design and iterative improvement will eventually lead to well adapted IT-applications. Yet, explicitly considering typical speech act patterns during software engineering might help to improve process alignment and reduce the number of iterations needed until the software converges to an optimized version.

In our pathway implementation we observed several such typical communication patterns which were initially not recognized as such. For example, by clicking on 5000 units of Heparin, the physician actually promises that he will ensure that the patient receives Heparin. It is important to note that a promise is not an online order and that an explicit “uptake” is necessary to anchor this speech act within an appropriate sequence of related acts in order to be successful. It is also important to note that computer based communication is not spoken language and “computerized” speech acts need careful consideration of their socio-technical embedding in order to be effective. Speech acts are illocutionary utterances that include an intention to interact in a specific way. If a computerized form of communication substitutes spoken language it is important to take care that the intention of the user matches with the implemented communication pattern.

Another example from our pathway implementation is the documentation of medication. In our initial implementation it was unclear whether the unspecific documentation was an order, a declaration, or a promise and which actions had to be taken as a consequence of this documentation. By introducing a clear separation of previous, current and future medication both documentation quality and process alignment could be significantly improved. An initial checking for the underlying communication pattern (e.g. is this an order or a statement?) could have avoided this additional iteration. Since medication is a step which typically occurs in many contexts we have also identified a candidate for a reusable module in future pathway applications.

So far we have only started to identify typical communication patterns or speech acts in the context of clinical process documentation. We plan to develop a catalogue of such patterns and a toolset of related IT interaction mechanisms, which should be helpful to increase process alignment in future pathway applications.

\[\text{REFERENCES}\]


\(^1\) IHE = Integrating the Healthcare Enterprise.


