# Defining the Paperless Workplace with the Paper Metaphor - Not a Contradiction in Terms

**Gerald Weber**

Department of Computer Science
The University of Auckland
38 Princes Street, Auckland, New Zealand
Email: gerald@cs.auckland.ac.nz

**Abstract**

The transition from a paper-based work environment to a largely paperless environment is still in full swing, in healthcare as well as in other domains. Analysts predict a further decade of efforts is necessary at least. In reality, paperless IT-based workflows offer both advantages and disadvantages over paper-based solutions. This is in contrast to the naïve expectation that a paperless solution should be a strict improvement over paper-based processes. We identify a set of generic requirements that address common drawbacks of IT solutions, and we propose a system model that helps to create IT systems which preserve the advantages of paper-based processing. The main tenet is that the paperless solution should be based on a naturalistic paper metaphor. Our system model supports auditability of IT systems by direct reference to the paper metaphor and ensures that information is faithfully presented to the practitioner. The system model is intended for mission critical applications such as health record management.

## 1 Introduction

The transition from paper-based systems to largely paperless systems is a process that might still continue for many years. More conservative predictions for the healthcare sector (Ford et al. 2006) have come up with dates around 2024 for a thorough rollout of paperless solutions. The healthcare sector is well positioned in comparison to many other areas, not least because of a strong interest on the part of policy makers. New substantial stimulus packages have been created to achieve that goal, but at the same time they bear witness that we are not there yet.

As with other areas in IT, a plethora of reasons can cause the implementation of a paperless workflow to be riddled with problems. As a consequence, in some cases this means that the workflow becomes worse than with the paper predecessor. Immature or faulty underlying software packages, problems in process management or administration are possible reasons, although the impact of such problems is disputed (Glass 2006). However these are beyond the scope of our work. Our focus is on aspects of the system *architecture* that can either help or hinder the system to be better than the old style paper office. We want to capture these aspects of architecture not primarily as technical blueprints but by naming important concepts that the architecture must inherently support.

It should be noted that one aspect making the transition harder is the fact that the translation from paper to IT-based work paradigms is not uniquely defined. We propose that careful observation of the needs of important domains can lead to stricter requirements for paperless workplace solutions. The central tenet will be that the system should incorporate aspects of the pre-IT workflows with a naturalistic and well-defined paper metaphor. The physical paper is turned into a virtual successor, initially with as few changes as possible. Ideally, the virtual paper should maintain all advantages of the physical paper, while offering the possibility to get rid of all drawbacks of physical paper. However, it is not always straightforward to decide if a property of paper is an advantage or a drawback. These considerations also help us in identifying aspects where current IT-based systems fall behind paper-based systems.

The paper is structured as follows. In Section 2 we highlight maintainability and auditability risks in current multitier applications that will be addressed throughout the paper. We present alternative system models that contain a naturalistic paper metaphor in Section 3. In Section 4 we present a model in which the naturalistic paper metaphor can be combined with traditional form-based interfaces. In Section 5 we review how this approach relates to other domains of IT use.

## 2 Advantages and Challenges of Paperless Systems

Our work is related to earlier work on Hippocratic databases (Agrawal et al. 2002), a type of advanced database applicable to privacy-intensive applications including healthcare applications. These systems create the possibility of auditing which data has been accessed at which point in time and in principle by whom. On the one hand Hippocratic databases support enforcement of privacy policies; on the other hand they support audits of compliance. This means that they have to reconstruct faithfully what actually happened. The compliance audit is based on an audit trail that stores database accesses. This enables some detailed analysis using access statistics that are available to the database. If we want to interpret these database accesses, however, we have to make some assumptions on the information system that processes the data before the presentation to the user. In particular we have to know the user's identity. If a database is used as a backend in a classical multi-tier architecture, the database has no reliable access to the actual user identity. The database access happens through artificial system user accounts that do not correspond to a single natural person. More generally, as soon as an architecture has a classical application server tier, this tier has unrestricted ability to transform

the data. It is not directly possible to infer from the database accesses what the users have actually seen and entered. In contrast, we present here a framework based on a naturalistic paper metaphor. The audit happens much higher up in the tier architecture, as close to the user interface as possible. The system keeps an audit trail of exactly what has been presented to the user and what the user has entered in which context. Still this can only be achieved under certain assumptions of correctness of the implementation of the framework. If the framework records the input incorrectly, this assumption is broken. This again highlights that the transition to an electronic system remains to some extent precarious. A further assumption must be made that all information rendered by the GUI is actually presented visually to the user and not, for example, occluded by other windows. Occlusion-free user interface technologies can help here (Lutteroth 2006).

Paperless systems allow the storage of information in presentation-independent form. The classical current implementation of this feature is the storage of data from paper-like input forms in databases. This feature will be a central topic of our subsequent discussion, where we will shed a critical light on it. This presentation-independent storage requires what is known as the shredding of information. The input forms are taken apart and the user-provided information is stored in a presentation-independent form. The unity of the information that was presented on one input screen is not only broken occasionally, in certain prevailing database design methodologies the data is preferrably taken apart. Such shredding is also well known in XML processing (Amer-Yahia et al. 2004). Database normalization also asks for the decomposition of tables, if they are not in normal form (Codd 1971). However, following best practices (Brodie & Schmidt 1982), the data is then presented to higher layers in application specific virtual views. In a plain multitier architecture, there is no inherent guarantee from the system architecture that a value entered under, for example, the label "Age" is also presented under the same label in the system output. However, in our system model, this will be guaranteed. Such guarantees are also helpful in scenarios where more customized user interfaces could be used, for example in tele-health applications that are directed towards the end user (Singh et al. 2010).

This shredding of data, common in systems with relational database backends, immediately reinforces the aforementioned problem of long term data storage, since the shredding can amount to an obfuscation of the context and provenance of the data (Buneman & Tan 2007). Moreover the shredding creates problems for usability and auditability.

## 3 System Models with a Naturalistic Paper Metaphor

In contrast to system models that are based on a universal middle tier, we want to discuss models that present a naturalistic paper metaphor. As indicated in the introduction, we begin with a system that turns the physical paper into a virtual succcessor with as few changes in properties as possible. from that we move on to system models that include more and more of the additional features of an electronic system.

### 3.1 The Single Copy Model

Figure 1 shows a conceptual diagram of a paperless system that is one of the most direct translations of a paper-based system and that we call the *singleCopy* model. Each paper record is translated one-to-one into an electronic record. The records can be deposited in the central repository and taken out on demand. In this system model we assume that the electronic records still inherit a fundamental property of paper-based records, namely the fact that copying requires an explicit operation. In this model, a record can only be taken out by one practitioner at a time. The immediate improvement that we obtain over paper-based records is the instantaneous communication with the repository, and possibly quick search functions. The fact that only one person at a time can have the record is on the first glance a disadvantage for, e.g., reading, but it can have several natural positive consequences. First of all this paradigm prevents write conflicts. People with write access will work strictly sequentially on the record. In addition, the paradigm can also naturally limit any prolific and inflationary access to personal data. If we assume that the record is kept with each practitioner only while actually needed, then this minimizes situations where concurrent access is needed. This kind of system functionality can be useful in several domains. In health care it might support the logic that the record travels with the client. In human resources it supports the concept that personal records are only accessed if needed. In general, the singleCopy model has the advantage that it assigns a clear responsibility at each point in time.

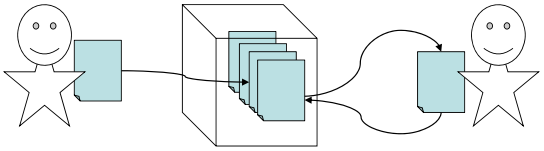

Figure 1: In the singleCopy model, each paper record is translated one-to-one into an electronic record. The user on the right hand side is taking out a record and putting it back into the old place within the repository.

The singleCopy model is a concurrency model, in that several collaborators could have interleaving access to the document, each one returning it as soon as possible. Such a concurrency model is often called a check in/check out model. It can be seen as a pessimistic model in that it prevents conflicts rather than trusting that conflicts would be rare. The singleCopy model is distinct from typical check in/check out models, in that the latter treat the process as a locking of a central resource, and therefore do not directly use the paper metaphor, i.e. a document that literally travels between workers.

### 3.2 Concurrent Read Access

Technically, the singleCopy model can of course be extended easily to include a concurrent read, on top of the exclusive write, since concurrent reads in themselves do not create write conflicts. This bends the naturalistic paper metaphor, and can best be explained with a copy on demand process that we will elaborate below. At this stage, the idea is that concurrent readers would get an explicitly immutable copy of the document. It is noteworthy that this extension immediately raises a well known concurrency issue, namely the question of which version the concurrent readers should see. For many applications it might be best if the concurrent readers always see the most recent version, in the interest of early information dissemination. However in other applications it might be appropriate that only finalized and authorized versions are seen by concurrent readers.

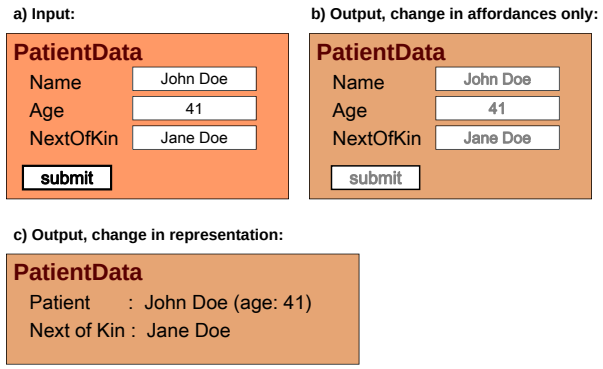


Figure 2: An input form and two different representations of the saved, immutable content.

We captured this requirement in an earlier discussion as the *Signed User Memo Viewpoint*, which requires that only signed and therefore final memos are distributed (Weber 2008). This approach then might require the possibility that one user can finalize and sign intermediate units of work without releasing control of the document.

We summarize the singleCopy model by comparing it against the goals of preserving the advantages of paper and reducing the disadvantages. The singleCopy model keeps the inherent advantage of physical paper with regard to conflict free writing. This advantage will become clearer when the alternative is discussed in the following copyOnDemand model. We have pointed out that in principle it is possible to introduce concurrent read access. One disadvantage of the physical paper is the possibility of loss through misplacement or destruction. This can of course be mitigated with an electronic system. An archiving component that keeps duplicates can be added.

### 3.3 Unity of Input and Output Representation

The strict application of the naturalistic paper metaphor requires that there is no difference between the form as it is presented while it is being filled out and the way it is represented when the entered data is recalled at a later stage. In this view, filling out a form should be directly equivalent to writing on a paper form. Hence, if this form needs to be accessed it should be represented in the same state as it was submitted. Most current form-based systems deviate rather strongly from this view, and this is typically directly coincidental with the process of shredding that we mentioned earlier.

However, we can also identify some motivation for changing the presentation. One plausible and justified reason to change the presentation relates to what is called a change of affordances. Before submission, the data is editable, and after submission it might be immutable. This is a change in affordance and justifies a change in representation. A faithful representation of entered data would use the same form layout, still presenting the data in the form fields, but having them represented non-editable. In Figure 2, the editable form is shown in a) and the immutable form is shown in b). The only change is a graying out of the form fields and submit button to indicate the change of affordance. The particular visual representation as grayed text is not our main concern and the illustration might show that graying is not an optimally readable solution, but the important part is that the change of affordance has just the character of an annotation.

In an electronic form, the text field is more than a reserved preferred space for information input; rather it is the only space where information can be entered. This is in contrast to paper forms, where technically every area of the form can be written to. It is appropriate to apply the notion of interface modes that refers to distinct changes in the reactivity of the interface. The notion of interface modes refers to temporal distinctions of interface behavior, such as when num lock is activated on the keyboard. But in an interface with a pointing device, cursor and focus, this temporal distinction is coupled with spatiality, namely with the position of the cursor. In this sense physical paper forms are modeless, i.e. they depend on the discipline of the writer to fill out the form neatly. In electronic forms the divide between editable fields and non-editable form area is much more fundamental; often these two areas are realized with strongly diverging technologies. In many form technologies, such as classical HTML forms as well as the more recent XForms, the labels of the form fields are indeed not a logical part of the form, but just peripheral text that is accidentally placed next to a form field.

This technological divide may well be in itself a contributing factor to the frequent change of representation between editable forms and the later representation of the entered data. Data will be entered in electronic forms, but presented in tables that no longer have the visual appearance of text boxes any longer. A possible such representation is shown in Figure 2 c). Just from the standpoint of a naturalistic paper metaphor, Version b) should be preferred over Version c). The change of affordance itself can be sometimes problematic in that it prevents practitioners to do important changes immediately. In principle, if a version control system is available, then one should present the form always in an editable mode, since earlier versions can be recalled. This discussion will be taken up again in Section 5 in particular with respect to wikis.

A natural consequence of this consideration is to base a paperless system on an underlying system that is closely based on the system model proposed in Figures 1 and 4, where the documents, including forms, on one level do not change appearance between different retrievals. With regard to our goal of preserving the advantages of paper and adding all possible support that an IT system can offer, we achieve the following. The representation of the form is still faithful to the way it was entered. However the system can be configured to prevent further editing of data, if this is wanted.

### 3.4 Historic Shift of Fundamental Form Semantics

It should be noted that the shift between the different form paradigms in paper forms and electronic forms also coincides with a shift in the apparent form semantics. One historic origin of paper forms, and a presumed root of the word "form", can be found in early legal documents, where standardized formulaic clauses were used with fill-in gaps for, e.g., the name of the person they apply to. Such a form might be a standardised contract. An example is a promissory note, an earlier and more perilous kind of an IOU. The interesting part here is that the semantics of the form must be completely expressed on the completed form. This means that after filling in the gaps, the completed form turns into a clause in natural language, although perhaps using some legal or other jargon. The effective power of the form, its semantics, is completely explainable from the resulting natural language clause without any recourse to its genesis as a fill-in-the-blanks text.

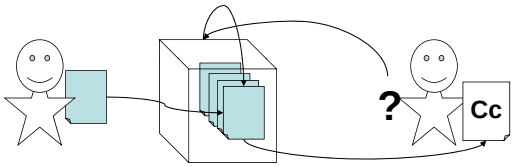


Figure 3: In the copyOnDemand model, if data is requested as indicated by the question mark, a new copy is created.

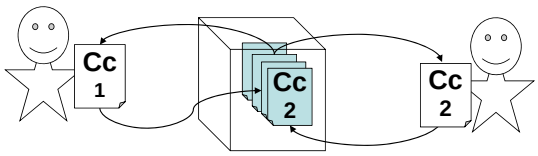


Figure 4: The copyOnDemand model allows parallel access to the same data, but can lead to conflicting updates.

In contrast, even before the advent of IT, many modern forms are designed, intended and presented as mere editable key/value pairs without any pretence to be self explanatory in their effect. The enterprise system behind the scene is the dominating element. The human readable elements of the form are merely making the required fields digestible to the untrained user and are semantically individual requests to divulge information: "Enter your name, enter your date of birth." This again coincides with the process of shredding the form data, and the fact that in a modern multitier information system only the filled-in fields are stored. Hence in a multi-tier application, even if the form fields were embedded in a fill-in-the-blanks text, this text would not have been recorded, and there would have been no tracking of the context in which the data was entered. From a human-computer interaction point of view this evolution is certainly not unproblematic.

It is worth noting that in certain critical applications such as bank cheques, which are incidentally closely related to promissory notes, the old style is still used and the filled in cheque contains a natural language clause. The aforementioned *Signed User Memo Viewpoint* is much closer to the earlier view of a form as a clause in natural language (Weber 2008). Under this viewpoint the form submission is separated into two distinct phases: first the form is prepared, this phase concluding with a pre-submit operation, and then the system presents the resulting memo that the user is about to authorise. This memo is not editable and it is made clear that the submission process is equivalent to the signature process. As a consequence the memo must be self-explanatory.

The modern key/value pair view is justified in circumstances where the form semantics is essentially data capture. Many healthcare applications fall into this category, for example if a vital parameter is regularly checked. However, the aforementioned problem remains: a multi-tier architecture offers no inherent guarantee that the data is presented under the same heading as it was entered. The shredding of information has another motivation in the recently increased interest in data integration. There is an interest to integrate data from different forms that might use slightly different field labels, into a single database. This requires a matching of equivalent fields. However, we recommend using a table style proposed in an earlier work (Lutteroth & Weber 2006) that keeps the original label with each data entry as seen in the following example:

| last name: | first name: | phone: |
|---|---|---|
| Johnson | Kelly | 0987654 |
| family name: | given name: | phone: |
| Gray | Pat | 0123456 |
| surame: | first name: | ph: |
| Smith | Bo | 3456789 |

### 3.5 Copy on Demand

We have gained the first system model, the singleCopy model, by a direct translation of paper properties to the IT world. We will now move away from that model in order to capture more of the possibilities of electronic media. However it should be noted upfront that the result of our considerations will be a model that is much closer to this initial model than to some of the more flexible general-purpose IT systems.

Many superficially exciting features of IT systems come with a price for serious, critical business processes. The first such property that turns out to be a mixed blessing for reasons already outlined is that paperless systems allow the parallel access to the same data. In Figure 3 we present a modified system model that we call the *copyOnDemand* model. Here records are obtained by copying information. This system model is a natural consequence of digital information processing, where it is simple to copy data, but it would require a deliberate effort to implement a restricted model such as the singleCopy model. Accordingly, the copyOnDemand model can have the problem of unrestricted proliferation of copies of the data with unclear semantics. In particular, version conflicts can arise if several people access the data at the same time, as shown in Figure 4. Sometimes such conflicts can be solved with automatic merge. Sometimes they require manual merge, however. This is hard enough for experienced software engineers; it would be naïve to assume that non-IT professionals can solve such problems, not because of a lack of understanding but because the necessary merge tools have a complex interface that an occasional user is not familiar with.

## 4 The Form-Oriented System Model

We will now introduce and apply the form-oriented system model (Draheim & Weber 2004), which is a simple yet powerful model for IT systems in the paperless office. This model will allow us to naturally address a number of the aforementioned issues and will ensure that at least the common drawbacks of paperless systems are mitigated. The model is easy to apply to all platforms and we plan to further explore its application to systems based on standardization efforts such as openEHR.

In the form-oriented system model, all possible input and output screens are defined by explicit screen prototypes that show the structure of information presented, as shown in Figure 5. The form-oriented methodology is designed for so-called submit-response style systems, a system class which includes classical Web interfaces, but also most proprietary enterprise application frameworks from individual vendors. The form-oriented system model is therefore suitable for a whole range of systems and can be used to improve the architecture of such systems. The model can, for example, be applied to systems based on Java EE. The model is, however, equally applicable to other technologies such as commercial of-the-shelf enterprise software and can be used to provide better system documentation, for example in the context of a comprehensive quality management.

In the past, a focus of form-oriented system models

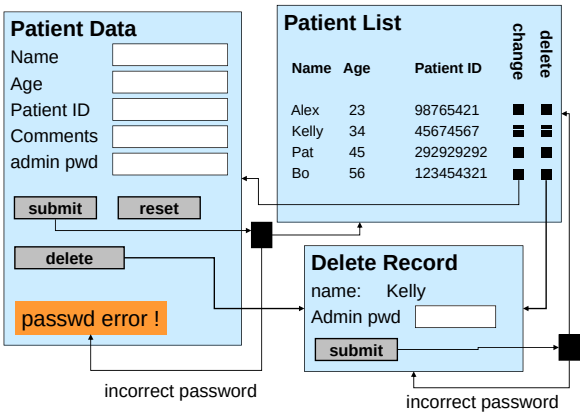


Figure 5: A form-oriented screen diagram, showing prototypes of the actual system screens and state transitions between them.

was on defining the flow of dialogues, and the necessary state machine defining the dialogue. One such state machine is represented in Figure 5 by the rectangular elbow arrows connecting the screens. The diagram in Figure 5 is called a screen diagram. It is a largely self-explanatory type of diagram that can be refined in this methodologyto more formal models such as bipartite formcharts. In our discussion here we will leave the flow of dialogue screens aside and focus on the individual dialogue screens and their mutual consistency.

The form-oriented model provides a synthesis of the singleCopy model and the copyOnDemand model in that it adds structure to the version space of Figure 3 yet matches well the natural structure of IT systems. The form-oriented model takes into account that in critical applications, data updates have to be done consciously and carefully. For that purpose the system interaction is modeled with traditional input forms as they are encountered on paper as well as, for example, in web forms. The filling out of the form is viewed as a preliminary activity that has no durable semantics in itself and only serves the purpose of preparing the actual submission of the data. The submission of the data happens through an atomic operation, typically hitting a button, which should only happen after carefully reviewing the data.

This model delivers a two-stage interaction paradigm, the preliminary preparation of the data and the atomic, heavyweight submission of the data. This matches a traditional paper-based work pattern, where people prepare and submit forms. In an IT system context it is, however, natural that after the submission the system continues in a specific state. The dialogue with the system is a continuous alternation of screens offered by the system and user interactions with the screen. Therefore the submission process in our model is called a page change, and after submission the user finds the next set of choices. For simplicity's sake, browsing actions of the user are modeled in the same paradigm as memo preparation. Hence in the form-based model, filling in a search screen is modeled as the same type of form as the submission of a professional assessment.

All these forms will be handled and archived in the same way. The form-oriented system model enables a system architecture, where all the important information is archived in the shape in which it was exchanged between user and system. This gives maximum flexibility for dealing with the system in a uniform way, yet still offers a very easy way to ignore unimportant data, such as all search forms, should they be irrelevant for the task at hand. The storage

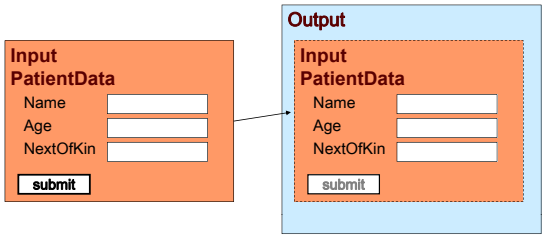


Figure 6: The form oriented system model allows the simple literal reuse of input data in subsequent output.

of all input forms in an audit trail still allows the on-demand blackening of sensitive data, thus enabling limited disclosure (LeFevre et al. 2004).

### 4.1 Usability Problems of Shredded Data and Solutions

In our research we are interested in crosscutting aspects of usability in IT systems (Weber 2008). One focus is on usability problems that create what is known as a gulf of evaluation (Norman & Draper 1986). A typical functional requirement of an e-health system would be, for example, that lab data for a client is captured and stored. This however does not guarantee availability of the data. A sensible requirement would be that all lab data that is created at a single point in time can also be viewed as a unit in the future. We call this a faithfully reporting system. This would be a direct consequence of the paper metaphor; the lab report is preferably always presented in the same, complete version. This avoids a possible gulf of evaluation, namely that the practitioner sees one lab value, but cannot necessarily see other lab data or even ascertain which tests were done at the same time. This example is a more complicated requirement than the aforementioned requirement that one lab value should be always presented with the same label. We will now discuss how far this requirement is addressed in a plain multi-tier architecture, and whether it is addressed in the form-oriented architecture.

Plain multi-tier systems store data in a database backend that can be used by a variety of applications. These applications in turn present themselves to the user through input and output screens which can be designed completely independently (Brodie & Schmidt 1982). With the flexibility comes a risk. In the fundamental system model of multi-tier applications there is no guarantee that data that was entered at the same time is also presented in the output at the same time. This is a direct consequence of shredding, as defined above. Hence in the plain multi-tier architecture a system design can easily be created where this gulf of evaluation arises. Since the presentation of the data is completely configurable, selected data can be presented, which might seem sensible to the developer, but leads to problems for the practitioner who uses the system. Avoiding the aforementioned gulf of evaluation requires an effort, namely active and adequate checking by the developers that the right information is provided at every place. Hence in plain multi-tier architectures it is harder to create a faithfully reporting model which will follow the terminology used before.

In the form-oriented system model, as opposed to the multi-tier model, faithful representation of the data as it was entered is always available. This is shown in Figure 6, where on the left side an input screen for client data is shown. The figure shows an example of a form-oriented model of the screens in a

very minimal system. This model, however, already captures one feature of a realistic system, namely corresponding input and output. In this model the output screen is shown on the right-hand side. As we see, we can define this output screen in the form oriented methodology by reusing the input screen one-to-one in the output. If, as shown in this example, only a single input screen is quoted, we would not need this representation, but could use the representation used in Figure 2 b), but this example serves as a demonstration for the more complex reuse examples that will be discussed in due course. The submit button of the input screen is disabled, but the content is represented faithfully. This model has a rigorous formal underpinning in the semantic model of form-oriented analysis (Draheim & Weber 2004) but it is also intuitively understandable. The arrows denoting reuse are always shown as straight arrows that are neither horizontal nor vertical, to distinguish them from the rectangular elbow arrows indicating state transitions.

This system model addresses the problem of shredding. It is now simpler to create a faithfully reporting model than a different, shredded model, using the terminology from before. Further questions arise now with respect to the long-term consistency of the model over time, and we will address them in the following by proposing suitable modeling solutions. In that way the reuse concept that we have applied in Figure 6 can be extended to a system design methodology that allows flexible creation of rich and interesting user interfaces, yet mitigates the problem of shredding. Shredding will not be completely prevented, since this would mean that the modeling method would cease to be universal. We deem it sufficient that a shredded model would be more complex than a non-shredded model in a metric that counts modeling elements.

### 4.2 Reuse of Screens in rich User interfaces

The reuse of input screens in output screens is the most basic and most fundamental example of a system that relies on the faithful representation of data. The form-oriented models are suitable as models to be developed in a standard graphical editor. Assuming that Figure 6 was created in a standard graphical editor, the question arises as to what keeps the model consistent if the definitions of the input forms are changed. We therefore use a different graphical representation in form-oriented models that makes the reuse explicit by avoiding the duplication of model elements. This is shown in Figure 7. The part of the input screen that is reused is called a snippet. The snippet is not repeated at the place where it is applied, but a placeholder is entered that is connected with the original snippet. The figure also shows that the definition of snippets can be nested. The semantics of this notation is that changes to the original snippet apply immediately and consistently to all other applications of the snippet. In this way we have achieved the long-term guarantee of a faithfully reporting system.

From the perspective of a naturalistic paper metaphor the snippet character should be visibly retained so that the end user can see that the snippet is part of a possibly wider input form. A more comprehensive solution could be found by applying the technology of transclusion in a document-oriented approach that we explored earlier (Lee et al. 2010). The notation in Figure 7 uses a special notation for selections. In order to change or delete a record the user of the system as shown in Figure 5 could click a button next to the person. In principle this can be supported also with our new approach. We choose here to use a slightly different but interesting approach. Every patient record has a small handle indicated with a

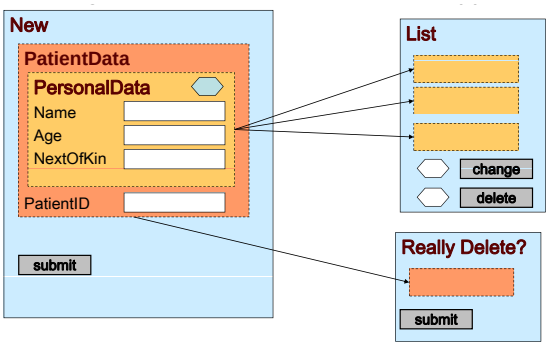


Figure 7: Non-redundant reuse ensures that there is a single place of maintenance for snippets.

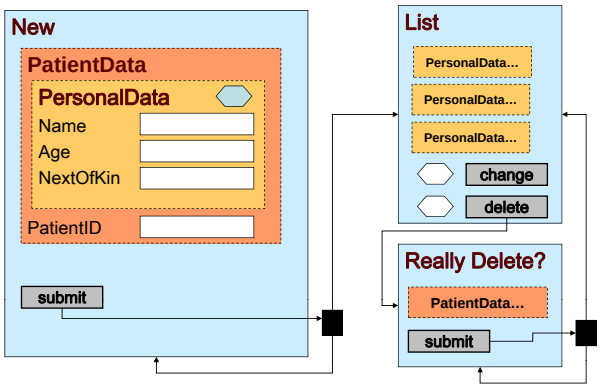


Figure 8: The semantics of the snippet reuse is symbolic, and a symbolic notation can scale up to large, multipage system models. The sympolic notation is used here within a screen diagram.

hexagonal shape. This small handle is a symbolic representation of the id of this record. In the system represented in Figure 7 the user can drag and drop this handle into a form field of the same shape, akin to a simple jigsaw puzzle. The two forms for change and delete have such a matching form field each. The small handles that enable drag-and-drop we call *opaque references* in form-oriented analysis (Draheim & Weber 2004, Draheim et al. 2005).

It is important to note that the reuse is not a purely graphical notation. In large systems, graphical models are often not scalable. The semantics underlying this reuse model is symbolic and this is shown in Figure 8. The snippets are addressed with symbolic names, and in the spirit of semantic faithfulness, by default this is the screen title of a snippet. Hence the Snippet "PersonalData" is invoked by using that name in a placeholder element, followed by the elision operator "...".

The symbolic notation makes it particularly easy to combine the model with the form oriented model of page change. This is also shown in Figure 8. The screen diagram also highlights the semantics of opaque references: on the right side, we see that after submitting the delete form on the list page we are taken to the delete page.

The reuse of snippets already used in other places is an intuitive first step to more maintainability. In large projects, the reuse model as shown in Figure 8 would, however, still exhibit an asymmetry. The maintenance of the snippet is tied to the first page where it is used, or at least the current page where it is placed at a given point in time. In a large-scale system development it is worth-while to use the snippet approach to perform a separation of concerns. The

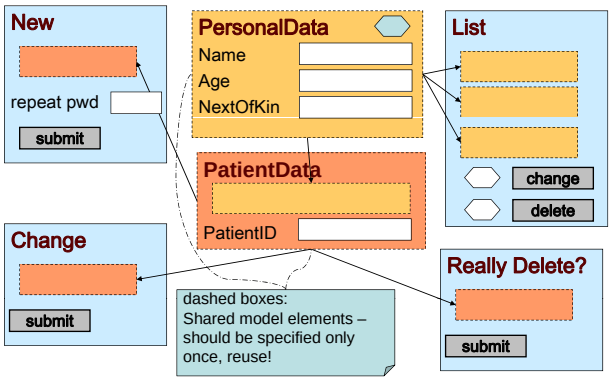


Figure 9: A reuse model that distinguishes between the shared model elements and the actual input and output.

snippets that are reused are maintained in a common shared model, and are kept separate from the pages where they appear. All placements on actual input and output screens use placeholders and hence are treated equally. This is shown in Figure 9.

### 4.3 Risks in Bulk Storage of Sensitive Data

Paperless systems allow easier duplication through archiving for disaster recovery. As a consequence, however, paperless systems harbour the danger that complete copies of sensitive data can be lost or misused. Sensitive health records for a whole population fit onto a memory stick of the size available at corner newsagents. Several instances of accidental loss of data on thousands of childcare claimants, social insurance holders, law enforcement officials, and other cases have been documented (BBC 2008) and the possibility of deliberate misuse of personal data for financial gain looms large.

Although paperless systems allow disaster-safe archiving of records, the long term storage is problematic. All currently available storage media are exposed to several independent long term risks: the accessibility of the data formats is not guaranteed, since the access to the information is highly indirect and depends in every case on a whole stack of formats. Furthermore currently available physical digital high-density storage media have a much shorter life expectancy than paper. Crucially for personal healthcare records, the life expectancy of digital media is typically shorter than the average human lifespan, while paper records have an established life expectancy longer than the average human lifespan. Experts in the field of digital data curation (Buneman et al. 2008) feel compelled to recommend explicit occasional paper copies of all sensitive data, e.g. printouts of XML data, a somewhat ironic state of affairs in the transition to a paperless process. Microfilm remains accepted as a higher density medium with similar life expectancy to paper.

## 5 Relation to other Domains

There is increasing interest in Electronic Laboratory Notebooks (ELN) which are likely to replace the traditional notebook, for example, in a research setting (Kihln 2005). These ELN are intended to help with IP disputes many years after the entries have been made. It is a requirement that ELN records are guaranteed authentic and tamper-proof, but also presented in a human-readable form (Drake 2007). This matches closely our considerations for faithful representation of the original entries, even if the main motivations might be slightly different. For a lab-book, the singleCopy system model is a natural first model. Two competing challenges are that, on the one hand, instant notetaking should be supported and therefore a modeless interface is intended. On the other hand it might be desirable that data series can be stored, and such data series are best represented in a tabular form.

A system model that comes very close to our goals can be found in Wikis. In particular the set of functionality provided by the MediaWiki platform is widely used now. Such a wiki supports on the one hand a concept of pages with identities that is close to the singleCopy model, in that there can be only one current version of the page. On the other hand it supports a copyOnDemand style approach to editing, since all the previous versions of a page are kept discoverable in the page history. There is a certain difference in that the singleCopy model has no checkout, thus Wikis are clearly an optimistic model. This model has some obvious advantages as well.

Another area which is concerned with reducing the divide between paper based workflows and electronic workflows is that of pen-based input. Pen-based systems approach the problem from the side of input devices, in enabling a style of interaction that is very close to working with paper, but is also enhanced by the new possibilities of electronic processing, including handwriting and drawing recognition (Plimmer & Apperley 2007).

In general, the problems addressed here are common to form-based systems and are encountered for example in more commercial applications. A classical case study is Electronic Data Interchange (EDI) (Emmelhainz 1992) that is widely used in the commercial world even before the advent of web-based e-commerce (Alonso et al. 2004).

This initiative has been valuable for retailers and suppliers since it allows them the electronic exchange of daily orders and business communication (Dowdeswell & Lutteroth 2005). The natural inflexibility of the automated systems contributed to a strong standardization of business communication. Only a few rigorously defined business message types are allowed (Kimberley 1991) in those systems, which contributed to interoperability between internationally distributed participants. In the healthcare area, openEHR is an example of a similar evolution. In the wake of the electronification of health records we observe a strong incentive to standardize health records on a global scale (Eichelberg et al. 2005). In a somewhat daring generalization from these two example observations we conjecture that there is a natural connection between the electronification of a business domain and the creation of stricter standard interchange formats.

## 6 Conclusion

The era of paper-dominated bureaucracy might well be in decline. The IT systems that replace paper are now a part of our everyday life. Once the initial excitement about the new tools has waned, users will expect rigorous quality attributes from such systems. IT systems are not automatically superior to paper-based solutions. Sometimes paper-based solutions come with natural and intuitive semantics. If push comes to shove, people are always able to perform at least basic functions with a stack of paper, such as browsing through all pages and looking out for headlines. In computer systems, even these basic functions have to be learned, and differ on every one of the competing systems at least.

In this work we have made a case that a naturalistic paper metaphor should be part of mission critical IT systems. We have focused on advanced parts of this approach, including dealing with snippets to make the various screens in a system consistent. We have discussed arguments why a system model that is close to the paper metaphor can foster better usability. A naturalistic paper metaphor can establish a well-defined abstraction layer; the technology that establishes the paper metaphor is separated from the application that uses the metaphor. The aim is to make the transition to a paperless system easy and intuitive for the practitioner, and hence give everyone working in important areas such as healthcare a bit more time to focus on what is most important, namely the professional service to the client.

## References


Agrawal, R., Kiernan, J., Srikant, R. & Xu, Y. (2002), Hippocratic databases, *in* 'VLDB', Morgan Kaufmann, pp. 143–154.

Alonso, G., Casati, F., Kuno, H. & Machiraju, V. (2004), *Web Services: Concepts, Architecture and Applications*, Springer Verlag.

Amer-Yahia, S., Du, F. & Freire, J. (2004), A comprehensive solution to the xml-to-relational mapping problem, *in* 'WIDM '04: Proceedings of the 6th annual ACM international workshop on Web information and data management', ACM, New York, NY, USA, pp. 31–38.

BBC (2008), 'Extent of data losses is revealed', *BBC News - Politics, 21st August* .

Brodie, M. L. & Schmidt, J. W. (1982), 'Final report of the ansi/x3/sparc dbs-sg relational database task group', *SIGMOD Rec.* **12**(4), 1–62.

Buneman, P., Cheney, J., Tan, W.-C. & Vansummeren, S. (2008), Curated databases, *in* 'PODS '08: Proceedings of the twenty-seventh ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems', ACM, New York, NY, USA, pp. 1–12.

Buneman, P. & Tan, W.-C. (2007), Provenance in databases, *in* 'SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data', ACM, New York, NY, USA, pp. 1171–1173.

Codd, E. F. (1971), 'Further normalization of the data base relational model', *IBM Research Report, San Jose, California* **RJ909**.

Dowdeswell, B. & Lutteroth, C. (2005), A message exchange architecture for modern e-commerce., *in* D. Draheim & G. Weber, eds, 'TEAA', Vol. 3888 of *Lecture Notes in Computer Science*, Springer, pp. 56–70.

Draheim, D., Lutteroth, C. & Weber, G. (2005), Robust content creation with form-oriented user interfaces, *in* 'Proceedings of CHINZ 2005 - 6th International Conference of the ACM's Special Interest Group on Computer-Human Interaction', ACM Press.

Draheim, D. & Weber, G. (2004), *Form-Oriented Analysis - A New Methodology to Model Form-Based Applications*, Springer.

Drake, D. J. (2007), 'Eln implementation challenges', *Drug Discovery Today* **12**(15-16), 647 – 649.

Eichelberg, M., Aden, T. & Riesmeier, J. (2005), 'A survey and analysis of electronic healthcare record standards', *ACM Computing Surveys* .

Emmelhainz, M. A. (1992), *EDI: Total Management Guide*, John Wiley & Sons, Inc., New York, NY, USA.

Ford, E. W., Menachemi, N. & Phillips, M. T. (2006), 'Predicting the Adoption of Electronic Health Records by Physicians: When Will Health Care be Paperless?', *Journal of the American Medical Informatics Association* **13**(1), 106–112.

Glass, R. L. (2006), 'The standish report: does it really describe a software crisis?', *Commun. ACM* **49**(8), 15–16.

Kihln, M. (2005), 'Electronic lab notebooks - do they work in reality?', *Drug Discovery Today* **10**(18), 1205 – 1207.

Kimberley, P. (1991), *Electronic Data Interchange*, McGraw Hill.

Lee, L.-C., Lutteroth, C. & Weber, G. (2010), Improving end-user gui customization with transclusion, *in* B. Mans & M. Reynolds, eds, '33rd Australasian Computer Science Conference (ACSC 2010)', Vol. 102 of *CRPIT*, ACS, Brisbane, Australia, pp. 163–172.

LeFevre, K., Agrawal, R., Ercegovac, V., Ramakrishnan, R., Xu, Y. & DeWitt, D. J. (2004), Limiting disclosure in hippocratic databases, *in* M. A. Nascimento, M. T. Özsu, D. Kossmann, R. J. Miller, J. A. Blakeley & K. B. Schiefer, eds, 'VLDB', Morgan Kaufmann, pp. 108–119.

Lutteroth, C. (2006), AP1: A platform for model-based software engineering, *in* D. Draheim & G. Weber, eds, 'TEAA', Vol. 4473 of *Lecture Notes in Computer Science*, Springer, pp. 270–284.

Lutteroth, C. & Weber, G. (2006), User interface layout with ordinal and linear constraints, *in* 'AUIC '06: Proceedings of the 7th Australasian User Interface Conference', Australian Computer Society, Darlinghurst, Australia, Australia, pp. 53–60.

Norman, D. A. & Draper, S. W. (1986), *User centered system design: new perspectives on human-computer interaction*, Lawrence Erlbaum Associates, Hillsdale.

Plimmer, B. & Apperley, M. (2007), Making paperless work, *in* 'Proceedings of the 7th ACM SIGCHI New Zealand chapter's international conference on Computer-human interaction: design centered HCI', CHINZ '07, ACM, New York, NY, USA, pp. 1–8.

Singh, J., Wünsche, B. & Lutteroth, C. (2010), Framework for Healthcare4Life: a ubiquitous patient-centric telehealth system, *in* 'Proceedings of the 11th International Conference of the NZ Chapter of the ACM Special Interest Group on Human-Computer Interaction on ZZZ', ACM, pp. 41–48.

Weber, G. (2008), A platform-independent approach for auditing information systems, *in* 'HDKM '08: Proceedings of the second Australasian workshop on Health data and knowledge management', Australian Computer Society, Inc., Darlinghurst, Australia, Australia, pp. 65–73.