Issues in healthcare IT
Lecture notes

Stockholm, December 16, 2013

International masters program in health informatics

J. Martin Wehlou MD, CISSP, CSDP

martin@mitm.se

MiTM
Man in The Middle AB

Table of Contents

1 Introduction

I'm very happy to be able to give this lecture (again). I've gathered a number of issues I find important in healthcare informatics in this collection of notes. As I wrote them, I came to realize there is no way I can cover all this in one session, so some of the material in these notes will probably not be covered in the lecture. If you take issue with something I seem not to talk about, say so, preferably while I'm still there.

> *One point I need to make clear: I'm an opinionated SOB. To me, the field of healthcare informatics is brimming with unwarranted conclusions and unfounded beliefs, so I take it upon myself to, prophet like, bring my own truths to you.*

2 Why are we doing healthcare IT?

Before we embark on the road to revolutionize healthcare with information technology, we have to know what we mean by "revolutionize". Yes, we want to make healthcare better, but what do we mean by "better"? Do we mean:

- *Making people healthier* by better diagnostic tools and better treatments?
- Achieve *quicker* diagnosis and treatment?
- Make it *simpler*, less duplication of effort?
- Make healthcare *safer*?
- Finding *better diagnostic tools and treatments* through better analysis of what we're doing?
- Making healthcare *cheaper* by having better economic tracking tools?
- Making healthcare *more transparent* by allowing the patients to see into the electronic healthcare record?
- *Avoid accidents* by making the electronic healthcare record more widely accessible?
- Making healthcare more *reproducible and consistent* by automating parts of it?
- Make healthcare *computerized?*
- Making healthcare *need less doctors*?

You can't have it all, at least not immediately or at the same time, so you'd better choose which of the above goals you prioritize. That's very important, but it is even more important that you *express* what you just decided. There's nothing worse than having a bunch of otherwise well-meaning people trying to achieve contradictory goals together, while none of them is aware that they are working at cross purposes with each other.

> *In healthcare IT, more than in almost any other endeavor, people often don't have a common goal and tend to work at cross purposes.*

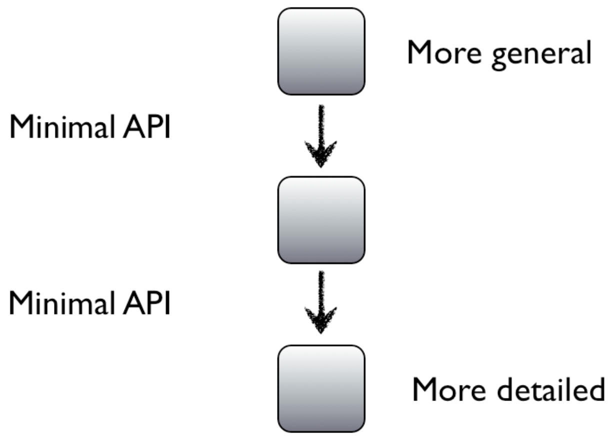
3 Encapsulation and separation of concerns

If we forget about healthcare for a minute, and look at how information is handled in other parts of society, we can see that as the information and knowledge volume increases, compartmentalization and delegation compensates for complexity and allows us to keep evolving. As an example that should be easy to visualize, let's take programming.

Without going into the entire history of computing, we can claim with confidence that two of the most important steps in the evolution to highly complex systems were the invention of structured programming and object orientation.

Object oriented programming (OOP) is based on the following ideas:

- the details of an implementation should remain invisible to everyone using an object in a larger context, so much so that the implementation can change as long as the external use of the object does not change
- the interface to the object should be as minimal as possible and contain nothing at all that depends on the exact inner workings of the object
- at each level of abstraction, the programmer composes objects and creates a new object with a higher level of abstraction

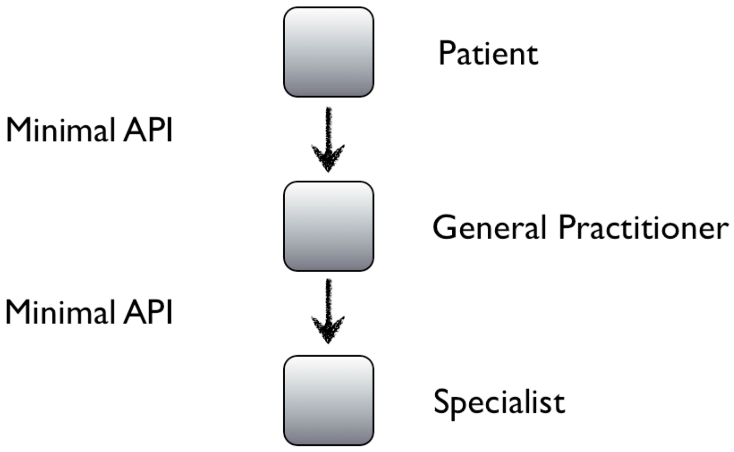


This way, OOP leads to ever higher levels of abstraction, each containing no details of lower levels of abstraction.

> *Correctly done, OOP removes from view all the internal complexity you don't need at any particular level of complexity. This staged reduction of detail, opaque encapsulation, is what turns a potentially exponential growth of detail into a linear process that can be handled by human minds.*

In medicine, the same process applies. The patient talks to the general practitioner (GP) using a high level API: *"I'm sick. I think it's my liver. It runs in the family."*

The GP object internally reasons in much more detail: *"The liver. Right... What he probably means is that he's prone to nausea and stomach aches, but that's probably because his whole family is living off hamburgers and get into drunken fist fights over the TV remote. Anyhow, I'll check his transaminases, not forgetting the gamma-GT."* But what he says to the patient after poking his abdomen for a bit is something like: *"Hm... your liver is maybe a little tender, we'll run some tests."*



If the GP had gone to a continued professional education (CPE) class and learned that there is now a virus causing a deadly disease involving symptoms of nausea, fights, and remotes, and that was brought to earth by the moon landing crew[1], he would still have responded the same way: *"We'll run some tests"*. In other words, the change in the GP's internal implementation of how to do medicine in a family practice in the space age does not lead to a change in his information interface to the patient. The GP is fully encapsulated and OOP compliant.

Let's push this example a bit and assume something is terribly wrong with the lab tests and the GP refers the patient to a gastroenterologist with the general question: *"What's up with this liver? Maybe a biopsy would be a good idea?"*

Now, let's further assume that the gastroenterologist agrees[2], goes on to make an appointment for a biopsy, chooses the most suitable ultrasound transducer, the right needle gauge and length, etc, etc, and does the biopsy. The pathologist also does his thing colouring, embedding, slicing and dicing, etc, etc, and all this results in an answer from the gastroenterologist to the GP: *"The biopsy showed a moderate degree of cirrhosis with some steatosis"*.

At this point, the GP is supposed to understand *"cirrhosis"* and *"steatosis"* and more or less what to do about it (cut out the alcohol and the hamburgers, the fist fights are no problem). But the GP does not need to know how to do a liver biopsy or how to prepare the samples for microscopy or even how cirrhosis looks in a microscope. Even if the specialist buys new equipment and then does his biopsies in a different and better way, this makes no difference in the interface between the GP and the specialist. In this example, the gastroenterologist is fully encapsulated versus the GP.

> *This encapsulation allows every layer of abstraction to evolve independently. The GP can change and improve his methods without the patient noticing[3]. The specialist can change and improve his methods without changing his interface towards the GP. This is the only way to allow medicine to evolve, going from the "super GP" who knew all of medicine in the middle ages[4] to the super specialists of today.*

It's relatively easy to see that the same process of layered levels of abstraction applies in all intellectual human endeavors, not only programming and medicine.

And here comes the moral of this story:

1 Don't worry. I'm lying.
2 This does happen.
3 Except as better, quicker, and perhaps even more gratifying encounters with primary care.
4 Which wasn't much.

> *To allow medicine to work efficiently, we must mirror the same levels of abstraction, encapsulation, and separation of concerns in the EHR as the EHR becomes our primary tool. If we keep flattening the EHR as is generally done today, with access to every detail at every level, we're moving medicine back into the middle ages instead of forwards into the 21st century.*

And, more bluntly:

> *Large, unified EHR systems are a really bad idea. A much better idea is loosely coupled specialist systems, each with a narrow interface, mirroring object oriented systems and allowing full knowledge encapsulation.*

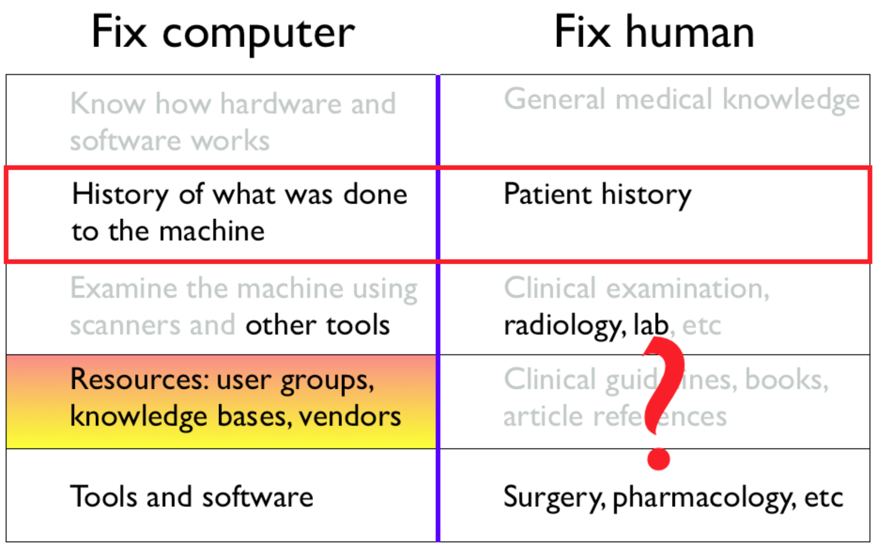
Exercise for the student: how does this destroy the idea of allowing the patient access to the EHR?

4 How does the EHR help?

Another question is how the EHR helps in the work of the healthcare professional. Some functions of the EHR support data entry and communication, and these functions are generally fairly well developed in current systems. What is almost entirely lacking, however, is knowledge and process support. Some simpler processes for nursing can be found here and there, but nothing really significant is going on in this space.

To see how the EHR fails in assisting doctors in their work, we need to compare it to other known processes that do work much better, and the example I'm choosing is "fixing a computer", since that is what most of us do far too often.

If we put the process of "fixing a computer" right next to the process of "fixing a human", we can find the same five stages or phases in both processes:



Fix computer	Fix human
Know how hardware and software works	General medical knowledge
History of what was done to the machine	Patient history
Examine the machine using scanners and other tools	Clinical examination, radiology, lab, etc
Resources: user groups, knowledge bases, vendors	Clinical guidelines, books, article references
Tools and software	Surgery, pharmacology, etc

The only thing we have developed to a significant extent in EHR systems is the "history" part, the second step in the diagram. Just as in the "computer fixing" process, this is not the most important step. It's nice to have, sure, but surprisingly easy to live without. Some other process parts can be found in EHRs, such as radiology, lab, and most of the therapies, but the most important step is missing entirely.

As most of you know, the real "meat" of the process to fix a computer is in the fourth step: "Resources: user groups, knowledge bases, vendors". Not many computers would be fixed today without the ability to reference what other people have seen and how to fix it (or not, as may be). Trying to fix a computer without any reference to these resources is doomed to failure except in a few trivial cases.

Actually, the same is true for medicine. Since we don't have the same kind of easily used resources, medical practice is still in the 1980's if we compare with fixing computers. We can fix humans, but the only knowledge we can use is what was hammered into us at medical school, or that we can fortuitously remember from a more recent CPE class. This is not good.

> *Let's make another comparison: the architect's productivity has certainly increased with the introduction of email and software that allows him to both write and maintain textual documentation, but the real advance of architecture is enabled by CAD software. Where is the CAD software equivalent for medicine?*

5 The history of medical records

As in all lectures, there is this "history" thing. But in this case, the history is essential to understand why things are as bad as they are.

In what follows, remember that I'm old enough to have actually lived through the following stages myself.

5.1 The absence of records

Not long ago, some general practitioners actually had no medical records at all. When I first took over a GP practice in Belgium in the 80's, the "records" I got consisted of two collections of documents:

1. Letters, lab reports, and other documents that my predecessor had not yet seen and discussed with the patient, in reverse chronological order. In other words, they were dumped on top of each other as they came in.
2. The same kind of documents, after they'd been seen, in order of their processing. In other words, they were dumped on top of each other in the second heap once seen.

The system worked as follows: the patient comes in and asks what the specialist said or what his blood tests showed. The doctor then asked around what time the visit to the specialist occurred or the drawing of blood, then proceeded to locate the document in stack number one. After reading it and discussing it, he prescribed something or other and off the patient went. The document ended up on stack number two and was never seen again. The whole incident then lodged somewhat loosely in the memory of the doctor and hopefully more permanently in the memory of the patient.

5.2 Paper based mementos

Obviously, this was a terrible state of affairs. Around this time, most GPs in Belgium[5] started keeping a real medical record for two reasons:

1. Fear of lawsuits. If you're sued for malpractice and you have no records at all, you're doomed.
2. To memorize details, such as exactly which medicine was prescribed for exactly what period of time, exactly when. And blood pressure measurements, and such.

In other words, the paper record evolved to a record of hard to remember details in diagnosis and treatment on the one hand, and simultaneously to a log of all interactions with the patient for legal reasons. The knowledge about the patient as such, his diseases, preferences, and most of all the overall plan in the diagnosis and treatment was not so much written down as memorized by the

5 I'm talking about Belgium for two reasons: firstly, that's where I was. Secondly, Sweden had proper records for patients much earlier at least in hospital care and care centers ("Vårdcentraler"), while real independent GPs are a rarity here in Sweden and I don't know how they handled records back then.

doctor. There was no need to write down these things since they are fairly easy to remember. After all, the patient always went to the same doctor anyway, so why write it down?[6]

The key thing to remember is this:

> *The classic paper based medical record was only intended to support the family doctor in the maintenance of hard to memorize details and was never designed to contain the overall picture of the patient or any diagnostic or therapeutic plan. Since there is no assigned and constant doctor in most practices anymore, it makes no sense to automate the paper based records without considerable change. But that is exactly what has been done. Current medical records are characterized by a torrent of useless details without a unifying context.*

6 It was also a great incentive for the patient to stick to the same doctor.

6 Research the EHR (fallacy)

Very often we encounter the proposition that a huge collection of EHR records is something we can use for research, with the expectation that we would find new diseases, effective diagnostic tools, and not least, the most effective treatments. This is a fallacy in almost all cases.

Let's first mention what *can* usefully be done research-wise with EHR data, which isn't much. We can use it to signal new epidemics or as an early warning of side effects of pharmacological products, but only as warning flags. The actual determination of what the epidemic consists of, or if the side effect is real, has to be done using other methods.

As far as comparing treatments, the EHR is worse than useless, it's downright misleading, and this use must be avoided at all costs. To explain why, you have to know the difference between "evidence based" medicine and "anecdotal" medicine.

6.1 Anecdotal medicine

This is how "old, experienced" doctors learned medicine. If they'd tried medication M for disease A in their patients, and it worked 9 times out of 10, they would draw the conclusion that it did indeed work, so they'd keep on prescribing the same medication in the future.

The problem with this is that:

- even 9 times out of 10 may be a statistical fluke
- these particular patients may be preconditioned to react well to medication M, particularly as they will tend to belong to a very limited gene pool, live in the same area and be exposed to the same environment
- the patients may be influenced by the charisma and conviction of the doctor to feel better even if the medication doesn't in fact help
- the doctor may be preconditioned to see an improvement in the patient's condition even if there is none
- negative effects may be suppressed by both patient and doctor

The grand total of all these effects are that medicine doesn't advance much as long as it's based on anecdotal evidence, unless the anecdotal evidence is extremely strong (e.g. strychnine and large bullet wounds seem to kill you).

6.2 Evidence based medicine

Evidence based medicine seeks to set up an environment so that only the real influence of the treatment is measured. Since we can't eliminate influences from the environment, the gene pool, or the influence of wishful thinking, we can only take care that these influences are more or less equal in a group of patients receiving the treatment we are investigating and another group not receiving the treatment. To make sure we create two groups that differ in no respect to each other, except that one group receives treatment and the other does not, we have to do the following:

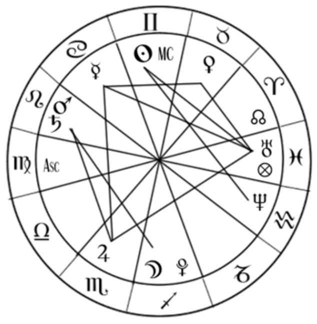
- patients must, one by one, be randomly selected to receive treatment or to not receive treatment (randomized, with a treatment group and a control group)
- both groups must be similar for all known characteristics such as: social group, gender, age, other known diseases, race, environment (the groups should be matched for these characteristics)

- the patients should not know if they receive the treatment or not, so another treatment, or placebo, must be administered to the control group patients (the study is "blinded")
- the healthcare worker that examines the patient and makes the judgement about results, should also not know if the patient got the real treatment or not (the study is "double blind")[7]
- the selection of the patients must be done *before* any treatment is done and the study should have specified limits (endpoints) defined beforehand, which define how the results should be interpreted (prospective)

Not every treatment can be double blinded. Operations of different kinds come to mind. But in the great majority of cases, you can do a prospective, randomized, controlled, and double blind study, and that is what you should always strive to do. We know that even if done entirely correctly, we still sometimes draw the wrong conclusion[8]. On the other hand, if we don't do it this way, we could just as well toss a coin or use astrology.

6.3 Electronic astrology

Using Electronic Health Care records to find out if a treatment works is another beast entirely. It's not prospective, it's retrospective. It's not controlled. It's not double blind, not even single blind. It's nothing more than automated anecdotal medicine. As the saying goes: *with computers, you can make the same mistakes much faster.* Don't.



6.4 Exceptions

National registries: by registering measures of outcomes from different hospitals or regions, including differing methodologies, large differences can be detected and studied further. The key here is that the indications obtained this way can generally *not* be used to draw conclusions about methods, but only to signal that a properly conducted scientific study could be warranted.

Epidemiological triggers: systems could signal unexpected correlations of incidences across records or systems, such as a suspicious number of very similar infections within a hospital or region. Any such signal would have to lead to a scientific investigation in its own right, but the signaling as such by the system could be very useful.

Neither of these exceptions lead directly to new science. Both need to be complemented by properly conducted studies in all but a very few exceptional cases of minor importance.

7 Which reminds me of a hilarious moment when I first read an abstract proposal from a collegue describing a "double-blind study of nifedipine in anesthetized dogs". Think about it. (He heard me laughing and changed it. Too bad.)

8 An often overlooked fact is that a perfectly executed study that shows a difference as a result of an intervention and the calculated "p" is given as "$p < 0.05$", actually means that the chance of the finding being just "luck" is 1/20. In other words, in a single issue of a medical journal with 30 articles, all of which claim "$p < 0.05$", one or two will show an erroneous conclusion. And this is under the assumption that all the studies were perfectly executed. Which, of course, they aren't, so the situation is actually much worse than this.

> *In other words, the role of the EHR as a tool of scientific discovery is very limited and should almost certainly remain so. Resist the urge to sell these systems as research tools. Please.*

7 What is the difference...

...between medical records, and say, accounting?

Is it the same thing, except we just need to exchange "customers" for "patients" and "invoices" for "prescriptions"? Nope, it ain't that simple.

There are a number of differences that we need to take into account. Let's go through a few.

7.1 Different kinds of data

In an accounting program, we have a set number of entities such as "customer", "supplier", "invoice", "payment", etc. Once we've got the entities down, we're not at some later stage going to encounter a new kind of entity we didn't know about, such as a "credit info song in C minor". There is no such thing and there never will be, not as far as accounting is concerned anyway.

In healthcare, though, we are continuously having to handle entities that weren't considered when the system was built. We may be able to handle x-rays, but what if someone comes along with a color x-ray[9]? Or an ultrasound in stereo[10]? Or an interactive microbiology report[11]? Just look at how many EHR systems there are that can't handle digital images, and you'll see what I mean.

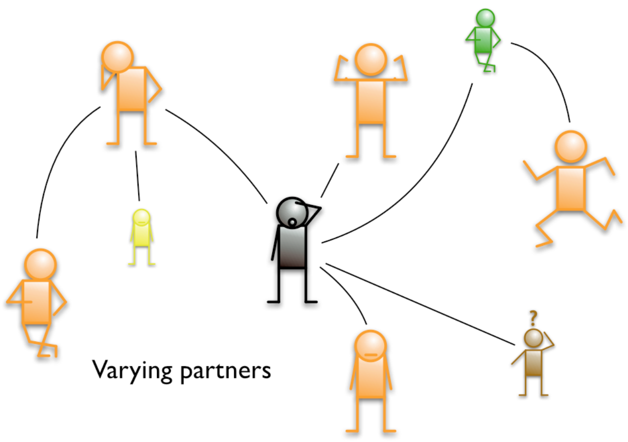
In short, the major part of medical records data is not well suited to relational databases. The demographic data, lab data, and most of what happens in operating theaters and intensive care units does fit the relational database concept. Yes, you *can* get anything into a relational database, but you are not exploiting their advantage if you dump formless blobs of binary info into extra wide single columns[12]. You're just faking it.

> *Try to avoid squeezing the data into a mold that isn't suitable[13]. Delegate the choice of database technology down to design and implementation, properly encapsulated, where it belongs. Don't make it a primary attribute of your EHR system.*

7.2 Indeterminate communication partners

An accounting system often communicates with other systems, but those systems are known ahead of time and rarely change. An accountant receives data files from his customers, but not from anyone else. A bank expects to talk to its customers and partner banks, not to just anyone at a random point in time.

9 These do exist, at least they did 20 years ago or so. Color was used to extend the dynamic range of film.
10 Why not? I made my own stereo stethoscope out of two regular ones once, and it's much better, especially for listening to small children.
11 Or with smell? Gangrene smell really gets your attention, so transmitting it through the application could be neat.
12 You know who you are.
13 The term in this case is "object-relational impedance mismatch".



In EHR systems, however, you may exchange data with the same small group of partners most of the time, but sometimes you need to exchange data with almost anyone in the world. There is no way you can usefully limit yourself to a preselected crowd of partners.

> *Avoid assuming commonality of standards with your communication partners. Again, use encapsulation and delegation to your advantage.*

7.3 Eternal data

In an accounting program, you don't have to be immoral or rude to let old data die. If you do a major system upgrade, it's totally acceptable to dump old data and only keep the end of year balances which are fairly easy to bring over to a new system.

In EHR systems, however, there is no "end of year balance", so all old data should be preserved more or less in its entirety when system upgrades or system replacements are done. The time period you have to keep data is open for interpretation and will differ from one country to another, but count on it being somewhere in the range of five to 100 years.

> *Electronic Healthcare Records data must be preserved across system upgrades.*

There is a complication or two hiding here. Do we upgrade *all* data every time, even unused data? If so, we'll have an exponentially growing data volume to handle. If we don't, we must maintain old software versions up to a century[14] just to be able to read old data. And even then, this is presuming that all data is in one place, which it won't be. There will be offline data, data on USB sticks, on (temporarily or permanently) disconnected machines, and so on.

7.4 Write only data

In healthcare, data doesn't change, while in accounting data is usually changeable, even though I know you expected me to say the exact opposite. You're not supposed to modify general ledger data but anything not actually posted should be entirely modifiable[15].

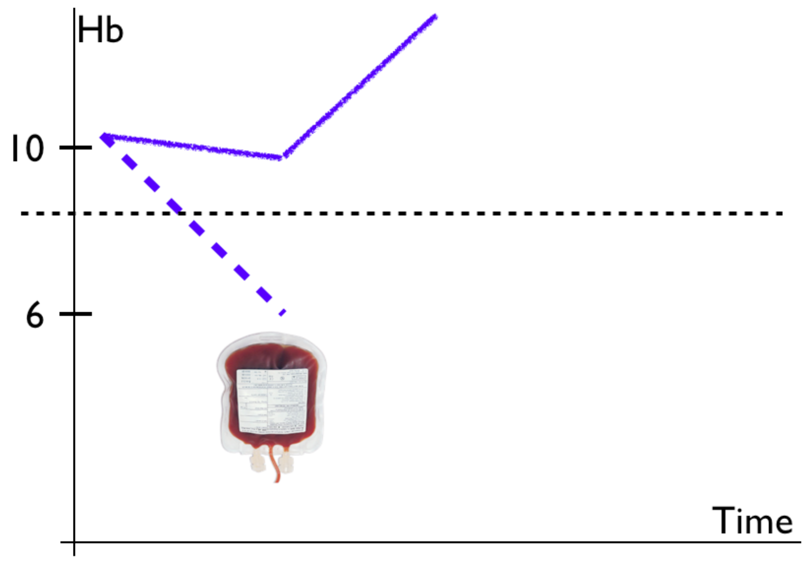
Healthcare data, however, is probably always best regarded as write only. Every change should be a new document or record that refers to the document it replaces, without actually removing the preceding document from the EHR. An example could help illustrate this:

14 Highly unlikely to work. How much digital data have *you* seen that is 100 years old and still is readable? (Yes, I know, computers weren't yet invented... it was a joke.)

15 I'd argue for having *everything* in accounting modifiable and I've built accounting systems like that. But we can't go into that here; it's an entire subject in its own right.

Imagine you get an Hb value of 6 from the lab, so you give the patient a unit of blood. Slightly later, the lab updates the value to 10 because they reran the test and came up with a different value. Now, if the original value of 6 is made to disappear and is replaced by the new better value of 10, the doctor who ordered the unit of blood is going to look mighty silly afterwards, and boy, do we hate looking silly.

> *All information that can have formed the basis for a clinical decision, even if that information is defective, must be preserved in the EHR. It should be made clear by the system, however, that other information has superseded it.*

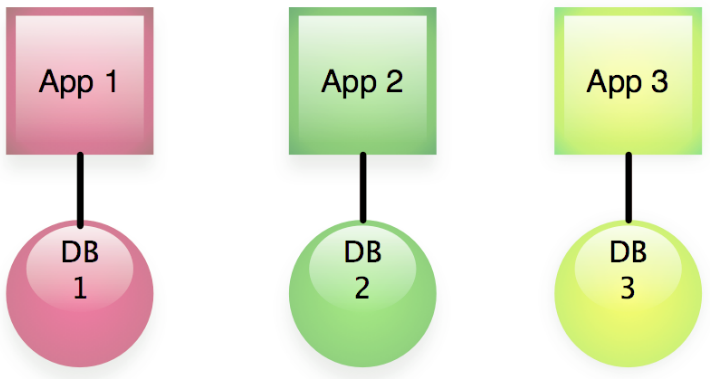


In the diagram, the outdated information is shown as a dashed line.

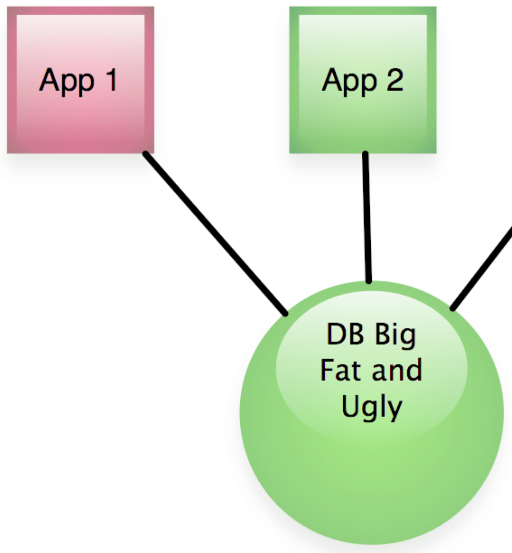
8 How to connect

When systems need to share data, there are different ways of doing that. First, let us not forget the need for encapsulation, as discussed earlier, but if you promise not to violate that principle, then we can discuss integration aspects.

Let's assume we have three systems which may or may not be of the same type.



These three systems can be integrated in different ways. The first and most obvious way is to simply share the same database for all three systems.

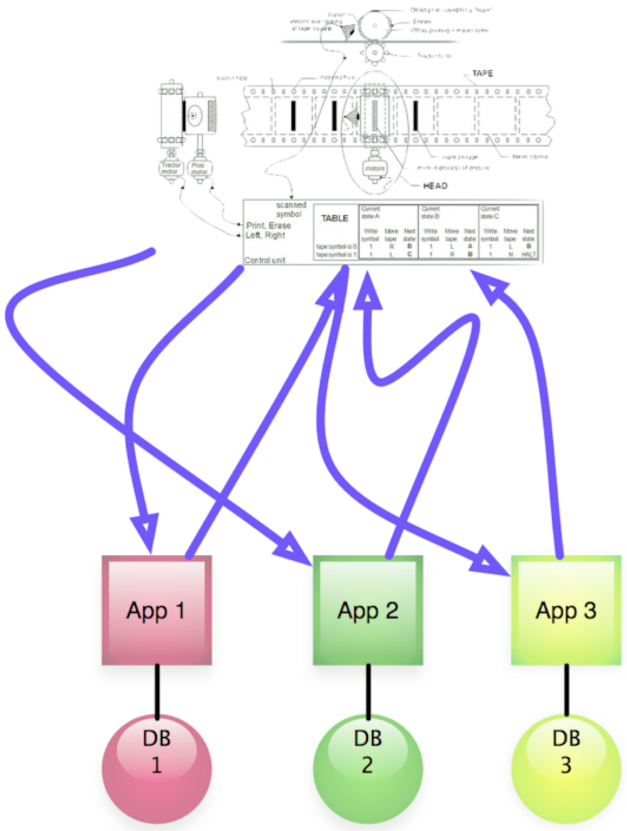


As you may figure out from the labels in the diagram, I'm not all that crazy about this idea. The problem is that the three apps will turn out to be three minor variations of the same application. Remember, in medical applications, not much processing or distinctive algorithms are going on, so the applications functionality is almost entirely based in the inventiveness of the data structures, and if you limit the data structures to one common structure, there is no place for competition in the remaining part of the applications. Hence, no sensible application vendor will ever want to work with the common database[16].

The idea behind this design is to reduce the data interchange problem by making all data equal to start with. But this doesn't work, since some data will *always* come from outside the system. Also, the ability of the system to adapt to change is severely limited, even though medical records systems need to be exceptionally agile and easy to change.

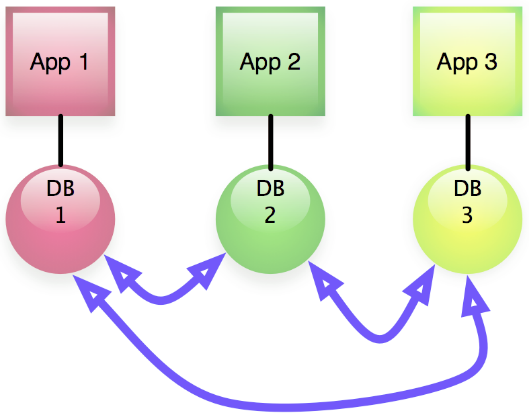
16 IMNSHO, this was the major reason for the very expensive failure of Stockholm's GVD project.

The next idea is what we see most today. It is based on the interchange of messages between applications. These messages usually pass through some sort of routing engine, where they also can be transformed from one message form to another.



Too much dependency on this model, however, makes it unnecessarily complicated to communicate within smaller organisations. Also, this model requires quite a bit of standards to be really effective, and too much reliance on standards makes for expensive and slow work[17].

My third model is based on the direct interchange of data between database instances.



Some minor conversion of formats can occur in these messages, but the main application is in synchronizing data between geographically distant instances of the same or very similar databases. This design achieves most of the advantages of the "big fat and ugly DB" above, with a cheap and resilient data design. This model is on its own not sufficient for all interchange[18], but complements the preceding messaging design nicely.

17 When a standard is needed, I'm all for it, but I'm even more in favor of avoiding the *need* for a standard if possible.
18 This is an important take-away: no single mechanism will suffice. You need a mix.

SICS in Stockholm was working on such a database for India[19]. The DIGHT database (open source) is intended to be slow synching over high- and low-speed links and be resilient in the face of unreliable links. This kind of database is very suitable to the write-only nature of most medical records data, where slow synch does not lead to data consistency problems.

Most obviously, any solution that involves a common database of the "old" kind, that is a humongous server running a single instance of Oracle or SQLServer is doomed to failure when we talk about 10^9 patient records as in India. And that is just the current population, more people keep being born...

Which leads us into another uncomfortable truth:

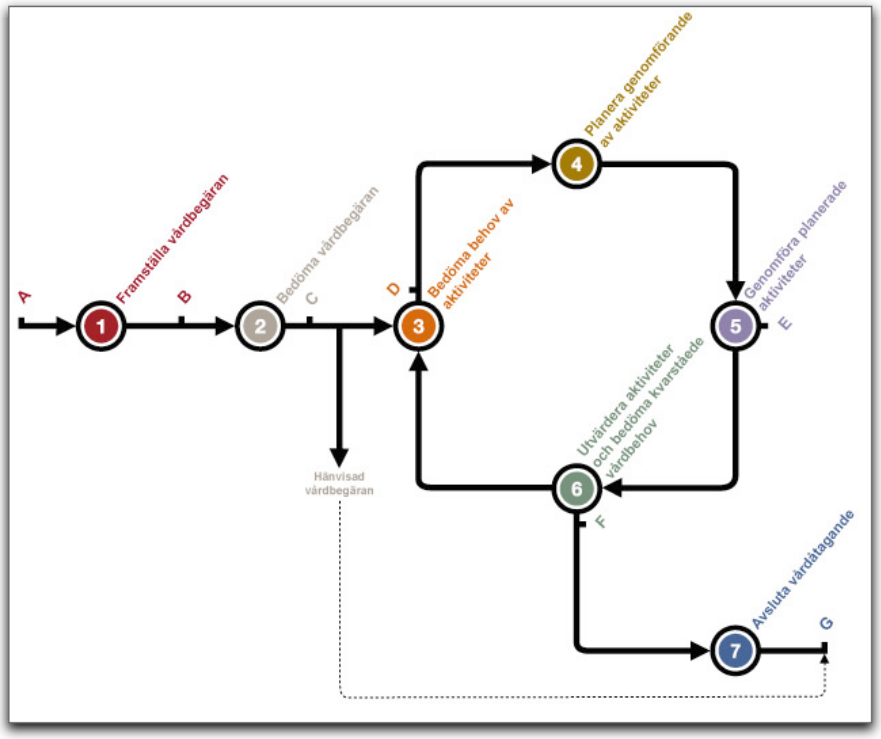
> *Any design that depends on Scandinavian conditions including very few patients, very reliable networks and infrastructure, perfect patient identification, government run purchasing programs, single payer, and pathologically obedient users, is bound to fail in the export market. These conditions of healthcare IT bliss simply do not exist in many other places and more particularly nowhere where there is a lot of people in large need of healthcare.*

On the other hand, if you design for adverse conditions in heavily populated third world countries, you will very possibly have an excellent product for sale even in Sweden. But we won't go into that here.

19 I checked yesterday (December 15, 2013) and, as far as I can see, nothing more has been done the last two years, so the project probably died. Too bad. The link: http://dight.sics.se

9 The oversimplification of processes

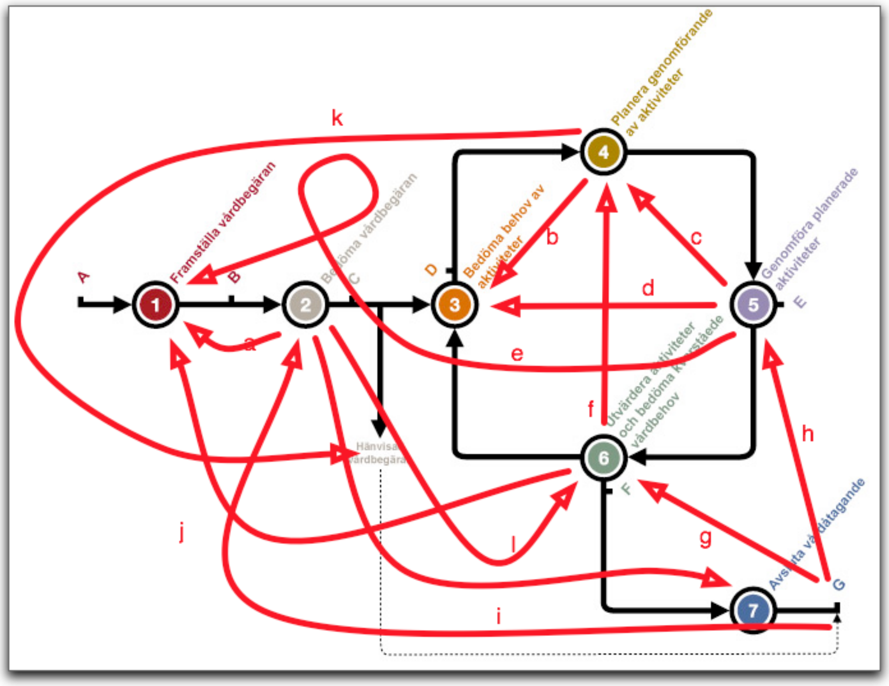
Just as the medical terminology is forced into a straightjacket by wishful thinkers[20], so are medical processes. Just witness the following diagram "borrowed" from SKL[21]:


A
1
Framställa vårdbegäran
B
2
Bedöma vårdbegäran
C
Hänvisad vårdbegäran
D
3
Bedöma behov av aktiviteter
4
Planera genomförande av aktiviteter
5
Genomföra planerade aktiviteter
E
6
Utvärdera aktiviteter och bedöma kvarstående vårdbehov
F
7
Avsluta vårdåtagande
G

Even without understanding every label in this diagram, you get the general idea: the patient has a need, the need is qualified and quantified, and if real you enter a treatment cycle consisting of planning, execution, evaluation, and possibly modification before another iteration, until done. Nice. At least it would be nice if reality deigned to conform to the diagram, but it doesn't. Adding in a few missed flows, the diagram should look more like this:

20 Coming up in a later section.

21 SKL = "Sveriges Kommuner och Landsting", an umbrella organization for regions and counties. The diagram can be found at http://www.flodesmodellen.se, a site dedicated, it seems, to this diagram and the idea behind it.


k
Planera genomförande av aktiviteter
Framställa vårdbegäran
Bedöma vårdbegäran
Bedöma behov av aktiviteter
Genomföra planerade aktiviteter
A
B
C
D
E
F
G
1
2
3
4
5
6
7
a
b
c
d
e
f
g
h
i
j
l
Utvärdera aktiviteter och bedöma kvarstående vårdbehov
Hänvisa vårdbegäran
Avsluta vårdåtagande

Even though my modifications may look like a joke, they are serious. I can give several real and daily examples of every one of those red arrows I added[22]. The thing is this:

> *If a process is inherently non-linear, but you need a linear model to implement a tool, you will fail however hard you try. Instead, embrace the non-linearity, especially since you have no choice. Your linear model would only describe a reality that doesn't exist and your tool will solve a problem that does not exist, in a way that won't work.*

Let's just take one example from the diagram. See that branch going from state 6 to state 7[23]? It says at state 6 to "evaluate and determine remaining care need" and at state 7 "terminate care". The entire diagram rests on the assumption that care is either given over and over or finally ended, but there are hardly any real world examples of this. When we "end" care, it's practically always a case of "suspended further care until something happens again", which it may never do. The only real end state is the death of the patient[24], since we do not include resurrection as a possibility.

Think of cancer, for example. When is the care ended? After five years of no disease? Ten years? Let's assume we have two patients, one has a recurrence after nine years, the other after 11 years. If we set the limit to ten years, the first guy is still in the iterative part of the flowchart for his first cancer. The other guy is, according to SKL's diagram, either a victim of a new cancer, or is a non-existent patient. What are they going to do? Ship him off to Finland?

Finally, what is the motive for this model? As far as I can make out, the idea is to determine specific points in the process so you can measure how many cases of this or that is being actively treated, and how many have been "finished". This is a call-center mentality that does not fit medical processes very well. Imagine having your cancer diagnosed and treated by Dell's customer service, and you get what I'm talking about.

---

22 For the actual examples, see my blogpost at: http://vard-it.se/?p=1330 It's in Swedish, but if you don't read Swedish, ask an attractive native to translate for you.

23 It's not clear from the diagram which parts are states and which are transitions. There's numbers and letters all over the place. If somebody can figure it out, please let me know. Or maybe not. Who cares, really?

24 And even then... imagine having erroneously "terminated" a patient. How do you recover from that state?

10 Beware of false requirements

> *When defining healthcare IT systems, be careful to formulate correct requirements and avoid "false" requirements, which are solutions in disguise.*

The problem with false requirements is that they needlessly limit the solution space. Let's discuss a few examples.

One patient - one medical record

This requirement is often postulated under the dual assumption that "things" are better if all information of a patient is available all the time, and that this implies a common system. There are thus three unproven and undefined parts in this false requirement:

1. The "things" that will improve are desirable and appropriate, even though these "things" are never defined. See my discussion elsewhere in this document about "Where are we going with healthcare IT".
2. That the care of the patient improves if all information about the patient is available. And, implicitly, that the improvement of the care is highly significant[25].
3. That to make all the information available, a common system is necessary and sufficient.

Actually, none of these postulates are true, but why this is so is left as an exercise for the student :).

> *But what's worse is the unintended consequences of this false requirement, namely the ruling out of any solutions that do not involve humongous stovepipe systems, stifling innovation while leading an entire industry up the garden path.*

A common database

We see this "requirement" quite often as a requirement to integrate with a predefined database. This database is sometimes even a separate project set up by the customer, see for example the GVD project in Stockholm[26].

A common database cannot reasonably be a real requirement, however. It's an architecture, or a design solution, but not a requirement. Since there is no way an end user can detect if the information is kept in a common database or not, it can't be a requirement[27].

On top of all that, it is also one of the worst ideas for architectural solutions we've ever seen.

Terminology systems

We often see a requirement that medical systems must work with a standard terminology catalogue such as SNOMED CT, but again, this is a solution and not a requirement. If we try to reel back the thread that connects this solution to the requirement that it ought to be hanging from, we get nothing. It's a solution looking for a problem which hasn't been found yet[28].

25 See in particular my earlier discussion about encapsulation and separation of concerns. There is every reason to think that the availability of the entire corpus of detailed information about the patient at every point of contact actually makes healthcare much worse, not better.

26 No footnote needed. The scandal is well known.

27 Someone somewhere made the point that a criterion can only be a real requirement if its implementation makes a difference to the end user. If the end user cannot know if the criterion is fulfilled or not, it is not a requirement. Too bad I forgot who said this. By the way, the "user notices" is a necessary, but not sufficient, characteristic for something to be a requirement.

28 Since I originally wrote this, I've started to use SNOMED CT in iotaMed to good effect, partially solving the problem I had with translations. But I still haven't seen any other real problems with a similar solution yet.s

11 Avoid the need for standards

Note well, I'm *not* saying you should avoid standards, I'm saying you should avoid the *need* for standards.

11.1 Lab codes

If you want two systems to communicate some lab value, they probably need to agree on how that lab value is identified and how it is measured. Or maybe they don't. If the value is to be presented to a doctor, and that is all it is going to be, there really is no need for the system to have the faintest idea what is going on, all it needs to do is present the name and the value as a string and leave the interpretation of the value to the user.

If, on the other hand, the system is supposed to order all incoming results of a specific kind and display the values as a graph, then it needs to "understand" which lab test is which, and which ones belong together. However, this situation is much rarer than you'd think, so in most cases there really is no need to match different lab tests against each other.

But, for the sake of argument, let's pretend there is. Let's assume that you need to be able to identify the same kind of lab test from two different systems as the same kind of test, and further, that the tests are actually similar enough to be regarded as of equal value. How will you then design a coding system that will match these two?

One way to go is to name someone responsible for maintaining a common coding scheme, and then hope that everyone else adopts that coding scheme for their own use. Any similar tests would then, hopefully, have the same code in all systems[29]. Even though this has been tried many times over the years, leaving a trail of abandoned standards along the way, like so much roadkill, labs still use a whole zoo of different coding tables.

Now, one can argue that interconnecting systems would be easier if all labs used the same codes, but the extra effort needed to achieve a universal code and keep all the systems up to date would probably overshadow any savings in integration work. Think about it this way: if a lab changes a code in its own tables, it needs to update the interface to all its customers using that code. But if a standard changes a code, *all* systems need an update, regardless of if they need that code or not[30].

11.2 Terminologies

Healthcare records have a number of terminology systems. Those that are useful are usually limited in size, often containing a few thousand items, of which the average doctor needs to know tens or hundreds. Examples are ICD-10 for disease and therapy classifications and ATC for grouping of pharmaceutical products. Such terminology systems are great and have shown their usefulness over and over.

But then we arrive at current efforts to standardize the actual terms used in medical records for findings, examinations, and conclusions. The main contender is SNOMED CT with around 400,000 terms. Yes, you read that right. *400 ton of terms.* The sheer size of this list should tell you something isn't right here. This idea is fraught with problems, not the least of which is that there is *no obvious use* for such terms. Let's check up on the reasoning behind these common terms.

29 A lot more is required before two measurements of the same kind from two different labs are directly comparable.
30 Yes, I'm overdoing this argument and it's not entirely accurate, but it's snappier than the full explanation.

Making the computer understand the EHR

If I write this in an EHR: "Patient is coughing, has a fever, and I wouldn't be surprised if he's got pneumonia just like his father had last week", and I write in another record: "The pulmonary infection is slowly improving", I am talking about the same problem in different terms. It may be hard for the software to determine with absolute certainty that I'm talking about the same problem, and even worse, it is hard for the system to parse out if the patient *has* the problem, if he is getting worse or better, and finally, what made me conclude that he has pneumonia and what I decided to do about it. It would be much easier for the system to understand all these relationships if I expressed myself in a stricter fashion, using one of several established syntaxes to describe what is medically going on.

The problem here appears to be *how* to make the system understand what's going on, but the problem in actual fact is *why* we would want the system to have such understanding. We'll get back to that.

Is the data reliable and complete?

No, it isn't. The reason for this is found in how records are written. During a patient encounter, this is what usually happens:

1. The doctor reads up on the patient from the records. If available, he scans summaries of the patient's history. In a fearsome number of current systems, no such summary is available, so he reads notes for as long as he can bear it and his attention span reaches, which is usually 30 seconds to a minute or two.
2. The doctor, if he is smart, asks the patient for major history points he may have missed while reading the records.
3. The doctor asks the patient about his current problems, if any.
4. The doctor examines the patient, jotting down notes on a paper or doing his darndest to memorize key findings.
5. The doctor draws his conclusions, discusses them with the patient, decides on a plan together with the patient, writes documents and prescriptions and sends the patient away.
6. After the patient is gone, the doctor dictates (or writes himself) the notes, including findings and conclusions.
7. Goes out to get the next patient or goes to lunch.

As you can see, the doctor dictates or writes down the notes in the healthcare records only *after* drawing conclusions and discussing them with the patient. In other words, he reaches his conclusions *before* writing down the data that ostensibly led to those conclusions, reversing the order of cause and effect.

> *The result will practically always be that whatever findings he writes down or dictates will be relevant to, and supportive of, his conclusions. You will almost never find unexplained or contradictory findings in records written this way. The findings will always support the conclusions.*

This leads to an extremely important corollary:

> *A correctly programmed computer will never be able to draw any valid conclusion from the health care record that was not already drawn by the doctor, since there is no data available in the record supporting any such missed conclusion.*

11.3 What is the benefit of common terms?

But now we run into a problem: *why* should the system need to understand what is medically happening? The reasons I am aware proponents of coding claim are:

1. to increase ease of communication between systems
2. to allow the system to automatically make diagnoses missed by the doctor
3. to force the doctors to express themselves clearer and with less ambiguity

Let's murder those arguments one by one.

Increase communications between systems

There is no hindrance to this communication today. Journal text is transmitted as plain text which is just as useful for a doctor or a nurse as text where every keyword is recognized by the computing system. As long as the formatting and alignment isn't just plain awful, it doesn't matter one bit if the program recognizes a term or not. Actually, it won't even be possible for the user to determine if the system has "recognized" a term or not[31]. And finally, there is no predetermined action that the system should be taking on recognizing the term. Which leads me to conclude that the recognition of a term by the system leads to no detectable change in behavior of the system, so it is not needed.

To allow the system to find missed diagnoses

If you look back to the previous section, you will find my claim that the EHR contains no findings about the patient which the doctor did not already explain in his conclusions[32]. This means that the system will never find data *in the records* that points to other conclusions than those already drawn. The only way to fill the EHR with "objective" data to be interpreted by the system would be to mindlessly collect as much clinical data about the patient as possible, without drawing any conclusion about them, leaving that to the system.

> *The only way I can see we can provide the system with sufficient and impartial data for automated decision making is to feed it using the medical tricorder[33], which, sadly, hasn't been invented yet. Or, subject every patient to an hours long all-encompassing clinical examination by a "doctor" who can do every clinical examination in the book without understanding the meaning of any of them.*

Forcing doctors to express themselves with less ambiguity

The lack of precision in medical records is often a source of frustration to non-doctors. They can't understand the waffling that is always going on, all the different ways something is said, or is almost said. On reading a medical record, you'd be pardoned for regularly crying out: "*Does* he or does he *not* have an abdominal problem requiring a laparotomy...?!" or something similar. This is then presumed to be easily solved by forcing doctors to come out and make a clear and definite choice when recording findings and conclusions and not let them get away with vague language.

If the cause of the vagueness was the vague terminology, there would be some value in this reasoning, but we're not that fortunate. The cause of the vagueness is fuzzy thinking. When you can't figure out what a doctor was really thinking from the records, it's because *the doctor himself* couldn't figure out what he was thinking when he wrote it.

31 If you paid attention, you recognize this as the characteristic of a design element that is *not* a requirement. QED.
32 I realize I'm repeating myself here, but this fact needs some hammering in.
33 If you don't know what this is, ask around until you find a Star Trek fan. He will explain it to you.

> *Forcing the doctor to choose between two clearly different and mutually exclusive findings or conclusions against his will, will only result in him either not writing down anything at all, or him choosing one of the alternatives more or less at random. In either case, the utility of the data will have become drastically worse, while at the same time looking better. This is a clear case of induced false precision.*

And this, dear reader, is the worst thing you can do to otherwise respectable data. The uncertainty in the findings and the diagnosis is of at least as much importance as the actual diagnosis itself[34]. By forcing a terminology on the doctor, you filter out the element of uncertainty and the attribute of open-endedness[35], thereby probably removing the most important part of the information. The solution could theoretically be to incorporate a measure of uncertainty to complement each term, but the complications of using such a system would quickly get out of hand[36].

The ability of common language and discourse to express findings and conclusions, together with exactly the right measure of ambiguity and uncertainty, cannot be replaced by a synthetic language without becoming completely unwieldy and useless. And even if it could be done, it would replace the current ambiguity with a synthetic ambiguity which would not have resolved the problem it set out to resolve. The problem is that reality isn't what we'd like it to be. Tough luck.

> *The ambiguity in medical records do not stem from a lack of terminology, but a lack of ability of the doctor himself to pin down the findings or the conclusions with more certainty. A terminology change does not change this root cause. It may only hide it.*

Too many synonyms

There is another part in the motivation for common terms that is simply ludicrous and that is that doctors use too many synonyms for the same diagnosis or finding. A common term list would force us to use one and the same term.

Well, seriously, if you had the choice between training tens of thousands of doctors to use the same term or program a computer to match say five different terms to one, which would you think is more rational and expedient?

Curiously, this effort to merge clinical terms into a smaller set is more difficult than one would expect. Terms turn out to have subtle differences in meaning, making it difficult to just dump them into the same bucket. What you have to understand, though, is that these subtle differences in meaning are absolutely essential and should not be forcibly eliminated. They're part and parcel of how medicine works.

34 Inspired by a similar deduction in a non-related area in the section "Don't cross a river if it is (on average) four feet deep" in "The Black Swan", N.N. Taleb, 2nd edition. As I read it on Kindle, I can't give you a page reference.

35 My spell checker insists there is no such word, but I'm not going to let my spell checker run my life.

36 You can do that in SNOMED. I once found a powerpoint presentation with a total of 28 slides showing how to encode a systolic murmur found during auscultation of a horse heart. 28 slides. The author wasn't ironic, he was proud of the result. Think about it.

12 iotaMed™

Finally we arrive at the solution for all our problems, iotaMed. Yes, this is my invention, but let me plug it for a while anyway. Consider it putting my medical records where my mouth is. This is what I would build if I could, and I can.

iotaMed is an initiative intended to solve the major problem in healthcare today, the lack of consistency and knowledge at the point of care.

> *Many years ago there was an initiative called "Problem Oriented Medical Records" (POMR). iotaMed has some traits in common with POMR, but differs in most important respects. POMR largely failed because it was before its time and it had a number of deficiencies which are not present in iotaMed. The central idea in iotaMed, the issue as clinical guideline, was entirely absent in POMR. Largely to shed the association with POMR and the cloud of failure that surrounds it, I've chosen to call the central concept "issue" and not "problem". Marketing is important; don't associate with failures.*

When you work with iotaMed as a doctor, you generally work in an interactive clinical guideline instead of a classic chronological record. The likeness with a pilot's checklist is striking[37], as is the parallel with many clinical guidelines used in specialized care[38]. There are a few major differences, however:

- The guideline is interactive and the progress through the guideline is preserved.
- Any data entered in one guideline is reused verbatim in any other guideline referring to the same data.
- The chronological record is still maintained and produced, but mainly as a product of the changes done in the clinical guidelines.
- National registries can be implemented simply as a compact guideline and since it makes use of the same data already entered into other guidelines, there is no double entry.
- At the start of a new consultation when the issues the patient has are still undefined, a classic clinical template is used, and as soon as one or more guidelines are activated, these data points are reused automatically, again without double entry.
- If a standard terminology system is used, the iotaMed system provides automatic translation of both guidelines and clinical input between languages[39].

iotaMed does not replace current EHR system, but forms the clinical interface to the EHR system for the doctor.

12.1 Issues

iotaMed is short for "issue oriented tiered architecture for medicine". Each "issue" corresponds to a

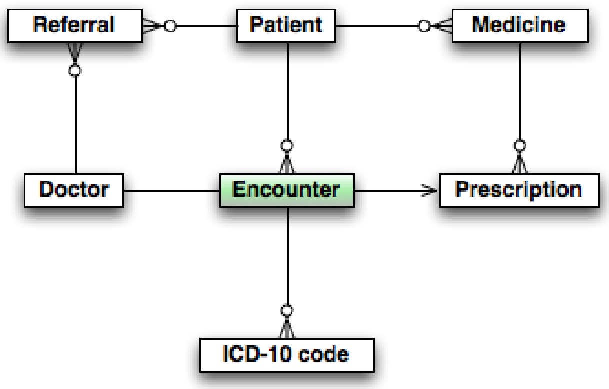
37 The incredibly positive impact of simple checklists on the improvement of quality in surgical care is described by Atul Gawande in the book "The Checklist Manifesto". iotaMed issues can be seen as checklists on steroids.

38 The more specialized the care is, the easier it is to introduce checklists and guidelines. The most payback from guidelines, however, will probably come from primary care.

39 Which, interestingly enough, makes iotaMed maybe the first system that could actually employ SNOMED CT in a useful way. Sadly, there aren't many translations. Even worse, when you start using it, you immediately find there are lots of really basic clinical findings missing, so even though they provide almost 400,000 terms, they seem to have missed the most common and important ones. How that is possible, I don't know. Your tax dollars at work?

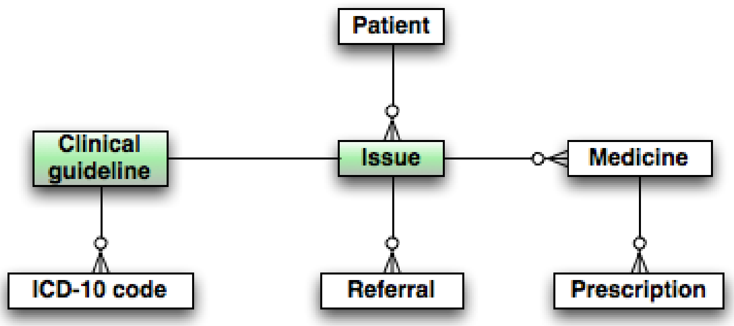
clinical guideline and can represent a disease or simply a symptom. "Headache" is an issue, for instance, and "Migraine" is a more specialized form of the "Headache" issue. Each issue has an interactive guideline, and each issue is coded as one or more ICD-10 codes or ranges of codes.

"Issues" form the data type "disease" in the EHR, which (amazingly) is entirely missing in most current EHR systems[40]. This is, roughly speaking, the data structure of a current EHR:



As you can see, the primary child element to the "patient" is the "encounter" and somewhere under that there is a list of ICD-10 codes tacked on for budgetary reasons. These codes are for a number of reasons not useable for finding out what diseases a patient actually has. The "encounter" is also there only for historical reasons; it has extremely little clinical value, but this was the way we took notes back when everything was on paper.

iotaMed uses another model:



In this model the patient comes first and immediately under that, you'll find the "issue". Clearly, most actions in healthcare relate to whatever ails the patient (the issue), not when and where he happened to get an appointment (the encounter).

Very interestingly, rearranging the data elements this way makes a lot of problems in current EHR systems go away almost automatically.

12.1.1 Confidentiality borders

In current systems, confidentiality borders is a real PITA[41]. Usually, if they're implemented at all, they relate to departments or specialities, sometimes to geography, or in the worst case, ownership of the care providing entity[42]. This makes no sense. The patient doesn't really care who is paying the doctor's salary or on which floor in a hospital he has his department, the patient cares who gets to know he has a certain disease. As things stand now, the assumption is that a potentially shameful

40 Somewhat like an accounting system that would lack the concept of money.
41 "PITA": Pain In The A... rear.
42 The latter is what the law seems to prescribe in Sweden. But the law actually allows for borders around issues as an alternative. Yay!

venereal disease is usually treated by a urologist, so anything the urologist says or does should then be confidential. Or similarly for a psychiatrist, for instance. But even though the urologist sometimes treats venereal diseases, he also treats bladder stones or a phimosis. If you as patient don't want anyone but your urologist to know about your syphilis, then you have to limit access to all of the urologist's records, but then nobody will know about your bladder stones either. Even though they could be relevant to your diet or medication for other problems, for instance.

But if you introduce "issues", then you've got a much better entity to hang confidentiality attributes on. The "issue" in this case would be "syphilis" and anything related to that issue would be limited to the doctors the patient allows to see that. Much more sensible.

12.1.2 Contraindications

Practically everyone in this business makes a lot of hay of "interaction warnings". As a doctor, I admit that automatic warnings for interactions between pharmaceutical products may occasionally be useful, but not as useful by far as everyone seems to think they are. Firstly, most warnings are ridiculous. The system I'm using consistently warns me of the risk of unintended pregnancy from using tetracycline ear drops, even if the patient is 80 years old and does not take the pill. (It is only a concern for a small number of younger women on the pill and if the tetracycline antibiotic is taken as a tablet.)

Another reason I don't respect interaction warnings much is that interactions are not so hard to know for doctors, and they are a rather infrequent problem. Reading the newspapers and "national programmes in healthcare" one would get another impression, but I stand by my opinion. It's not a major problem.

Contraindications, however, is another matter entirely. These are hard to remember for doctors, frequently a problem, and often with deadly results. Contraindications are really, really important, but current EHR systems never warn for them, because they don't have the data to do that.

A contraindication is when a pharmaceutical product should not be given for one disease in the presence of another. That other disease forms the contraindication. Some products for the prostate or heart should not be given in the presence of glaucoma of the eye, for instance. Some products for the heart should not be given if the patient has prostate problems.

The reason that these warnings are never implemented is that the EHR system does not have the concept "disease" so it can't know if the patient has "glaucoma" or "prostate hypertrophy", however many times these words or synonyms are mentioned in the EHR text. But "issues" solve that. It is very easily done to warn against certain products if any issue is active for the patient that constitutes a contraindication according to the pharmaceutical database (where it *is* included[43]).

But we can go further: even issues covered by confidentiality rules can be automatically included in the contraindication testing. iotaMed could warn for a contraindication that the doctor would normally not be allowed to see. This is impossible in current EHR systems.

The goodness doesn't stop there. If a contraindication develops *after* a medication has been started, for instance a kidney failure occurs in someone who is on medication that should not be given in kidney failure, the system can warn about that as the issue "kidney failure" is activated. And this is a very frequent problem in current systems, as it is far too easy to miss reviewing old therapies as new organ failures occur.

43 The contraindication information is present in the textual data for every product, but not coded into the national pharmaceutical databases we have in Sweden. The reason, given to me every time I asked why not, is that nobody asked for it. Several other people I know have gotten that same answer. We're all, obviously, nobodies.

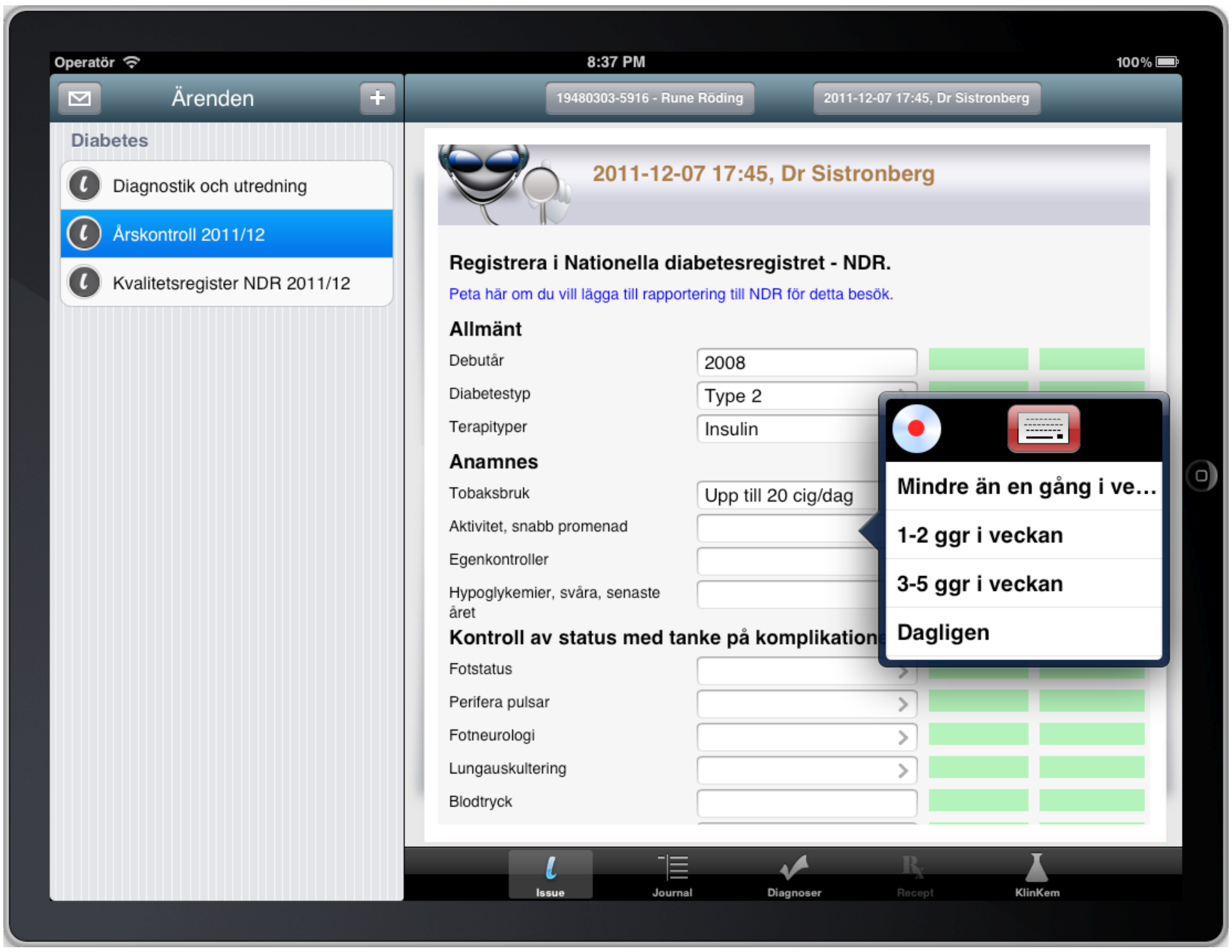
12.1.3 Dosage tables

A couple of years back, there was a big flap about the Astrid Lindgren children's hospital, and how they have a problem with dosage errors of medication in children. The dosage tables they normally used weren't good enough, since the correct dosages for a medication depends on the age of the child, organ problems such as kidney failure, and the exact disease the medication is given for. Antibiotics, for instance, are differently dosed for throat infections, pulmonary infections, and ear infections, and the difference is huge.

They've come up with fairly complicated verification processes and maybe they've worked around the problem somewhat. But if they'd introduced "issues", the correct dosage table could much more easily be selected and the problem would never have taken on the proportions it has now.

12.2 Demo

If we have the time, I'll show a demo of a prototype of iotaMed in the lecture. But in case you missed that, or I never showed it, or you just want to enjoy the screens and concepts once more, here are a few screenshots.



2011-12-07 17:45, Dr Sistronberg

Registrera i Nationella diabetesregistret - NDR.

Peta här om du vill lägga till rapportering till NDR för detta besök.

Allmänt

Debutår	2008
Diabetestyp	Type 2
Terapityper	Insulin

Anamnes

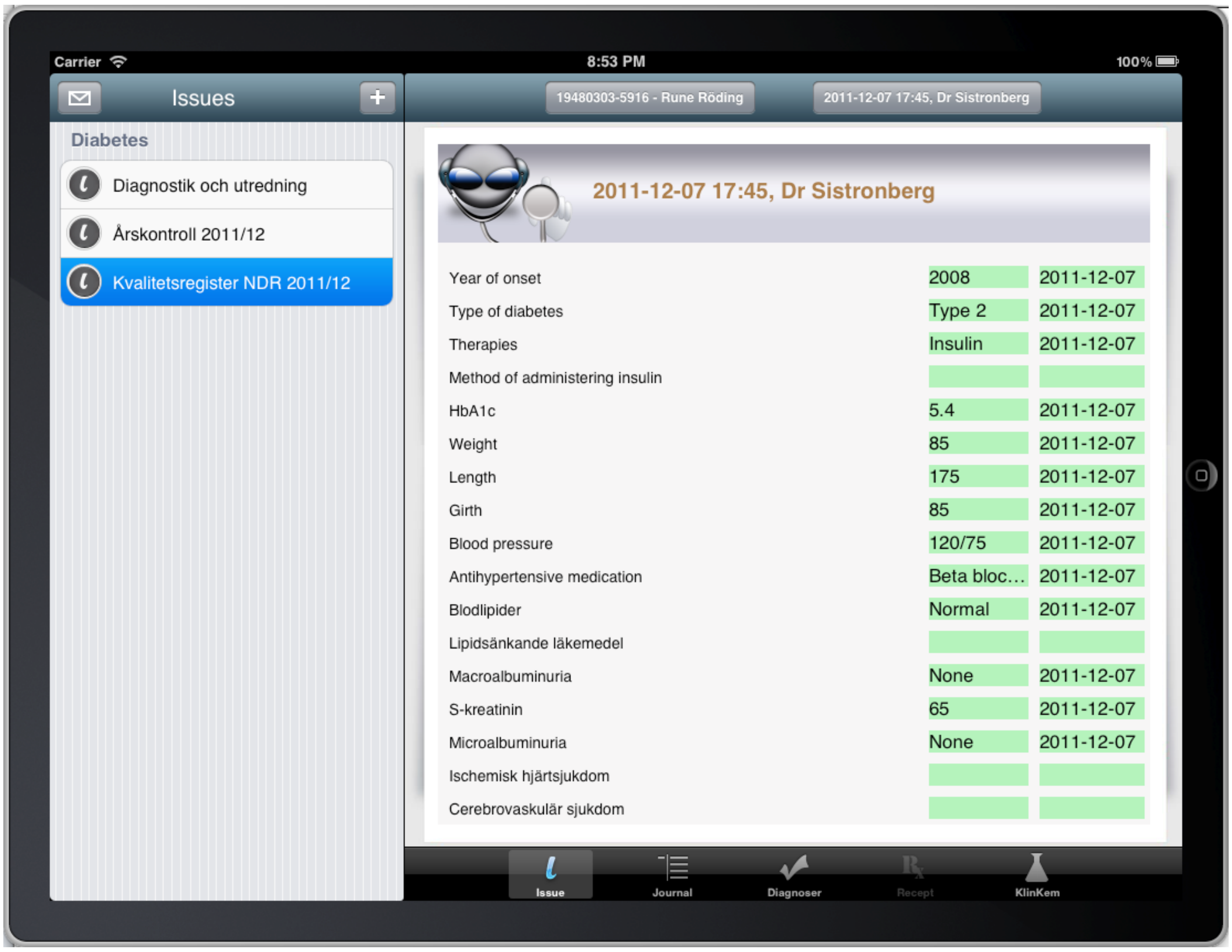
Tobaksbruk	Upp till 20 cig/dag
Aktivitet, snabb promenad	
Egenkontroller	
Hypoglykemier, svåra, senaste året	

Kontroll av status med tanke på komplikation

Fotstatus	
Perifera pulsar	
Fotneurologi	
Lungauskultering	
Blodtryck	

Mindre än en gång i ve...
1-2 ggr i veckan
3-5 ggr i veckan
Dagligen

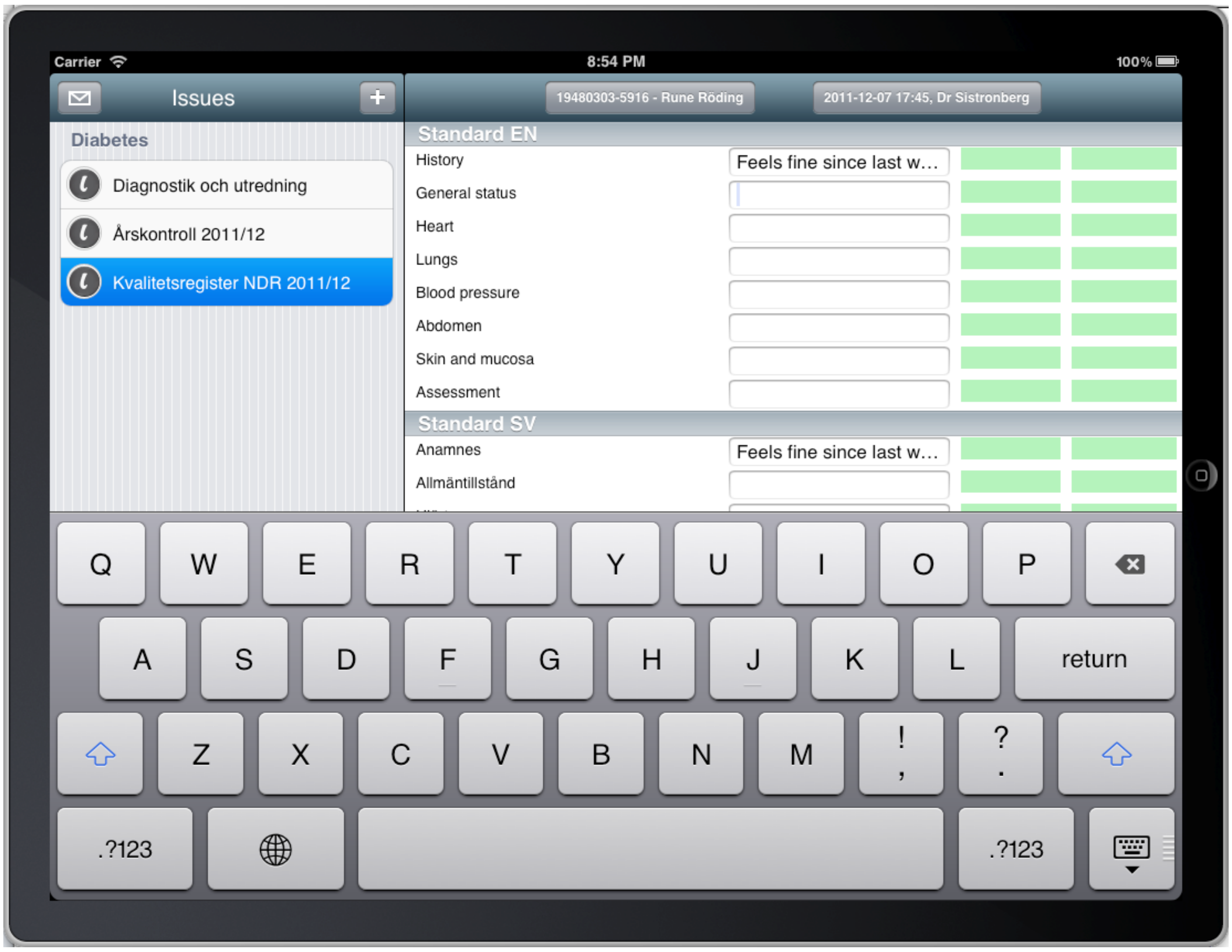
In this first shot you can see how a normal issue could look. The text itself comes from a clinical guideline on http://viss.nu, the recommended guidelines for Stockholm. The text has been instrumented with entry fields and relevant older data is displayed at the end of some lines in the green fields. If the document recommends a set of lab tests, you tap the button and a lab request for those tests is prepared and ready for you to send. The same way works with referrals. Since they originate in a clinical guideline containing the reasons for referral, there is very little or nothing for the user to add to the text iotaMed itself can build for the referral.



2011-12-07 17:45, Dr Sistronberg

Year of onset	2008	2011-12-07
Type of diabetes	Type 2	2011-12-07
Therapies	Insulin	2011-12-07
Method of administering insulin		
HbA1c	5.4	2011-12-07
Weight	85	2011-12-07
Length	175	2011-12-07
Girth	85	2011-12-07
Blood pressure	120/75	2011-12-07
Antihypertensive medication	Beta bloc…	2011-12-07
Blodlipider	Normal	2011-12-07
Lipidsänkande läkemedel		
Macroalbuminuria	None	2011-12-07
S-kreatinin	65	2011-12-07
Microalbuminuria	None	2011-12-07
Ischemisk hjärtsjukdom		
Cerebrovaskulär sjukdom		

This next screen shows the "NDR" issue, which refers to the form for the "National Diabetes Registry" in Sweden. Since there are easy to use attributes in the template file that determine which values to select, the reporting to the national registry is very flexible, and very accurate. No double entry is required either.



Diabetes

- Diagnostik och utredning
- Årskontroll 2011/12
- Kvalitetsregister NDR 2011/12

Standard EN

History	Feels fine since last w…
General status	
Heart	
Lungs	
Blood pressure	
Abdomen	
Skin and mucosa	
Assessment	

Standard SV

Anamnes	Feels fine since last w…
Allmäntillstånd	

The third screen shot shows how the chronological record looks in iotaMed. It starts out with a free form patient history and then with a series of standard entry fields. These fields correspond exactly to what we have in current EHR systems, but the actual data values are reusable in issue templates. Also, any values entered in any template that is not already part of the default in the journal record, is automatically added.

This chronological record in fact shows the same information as the issues, but in a chronological fashion. To make a programming comparison with source code control: you could say that the chronological record is the change log, and the issues are the source files.

To find out more about the system, go to **http://iota.pro**

13 Appendix

13.1 Contact information and links

My email	*martin@mitm.se*
Phone	*+46-70-5581217*
iotaMed project home page. Lecture notes and video from this and other lectures or demos will also be found there.	*http://iota.pro*
We have a forum for discussions around iotaMed and other issues in healthcare IT. It's currently in Swedish, but if there is interest for an English section, we'll make one	*http://vard-it.se/vi*
My Swedish language blog about healthcare IT	*http://vard-it.se*
Some of my posts are also on "IT i Vården"	*http://itivarden.se*
I have an English language blog about programming, security, and healthcare IT at	*http://ursecta.com*
And then again, a site with quite some programming material, and a mixed bag of stuff	*http://wehlou.com*