Johner Institute

Development of medical software

With Daniel Reinsch, Prof. Dr. Christian Johner

Transcript

00:00:05 Speaker 1

Medical Device Insights, a podcast by the Johner Institute for medical device manufacturers, authorities and notified bodies.

00:00:18 Speaker 1

Last week there was a remarkable event in Munich, namely the Bits and Bretzels and of course it was also about the digital health startups, of which there were really many, many of them.

00:00:29 Speaker 1

And 1 of their central issues is, of course, yes, how do we develop software in this whole regulatory environment and they have quite a lot of respect for it.

00:00:41 Speaker 1

And so I thought this would be a good occasion to talk to someone who manages to support companies all day long in making this regulatory madness survivable, especially in the context of software.

00:00:53 Speaker 1

And that is Daniel Reinsch.

00:00:55 Speaker 1

I would like to introduce you to Daniel very briefly, so that our listeners know who you are, what you do here at the institute and yes, I just need an impression of you for a moment.

00:01:05 Speaker 2

Yes, I like to do it.

00:01:06 Speaker 2

Yes, Daniel Reinsch is my name.

00:01:09 Speaker 2

I've been with the Yoon Institute for a good two and a half years now, I previously developed software myself, co-developed cardiac assist systems and thermal disinfectors, as the software looks after shares, also as a project manager, software project manager.

00:01:24 Speaker 2

And exactly, at the Jona Institute I have now switched to the other side, so to speak, look more at the whole thing from a regulatory perspective and support companies in understanding the standards not only as a hurdle, but actually also as an aid, even if that is of course sometimes not quite so easy.

00:01:44 Speaker 2

Exactly, that's all in all, people who have a clue have thought about it.

00:01:51 Speaker 1

Yes, yes, and that wasn't meant cynically.

00:01:54 Speaker 1

But let's now perhaps really get into practice.

00:01:57 Speaker 1

Well, you observe them in large numbers.

00:02:01 Speaker 1

What are the things where you see, ah, maybe the startups are struggling with that right now, also the other companies in the context so forever and again or where is there, I know, criticism from the notified bodies, from the authorities?

00:02:13 Speaker 1

Yes, so it's not just the European agencies now, but perhaps also internationally.

00:02:17 Speaker 1

Are there so, don't know, 3 most typical mistakes that you see again and again and where you

00:02:23 Speaker 1

You say that it doesn't have to be that way.

00:02:26 Speaker 2

Yes, so a mistake that actually happens quite often is a certain inconsistency in the documentation.

00:02:33 Speaker 2

So a classic example is that the software is provided with a certain security class, class A is also quite common and then you see, but you take a quick look at the risk management file and see that the software can cause damage.

00:02:47 Speaker 2

It's just inconsistent.

00:02:49 Speaker 2

Where the mistake lies now you have to see but

00:02:52 Speaker 2

That doesn't work like that at first.

00:02:53 Speaker 2

Or a second example is that many companies have major problems with it, especially startups and especially in stand-alone software development, app development, digital health applications, coming from such a very free world, always forward-looking, no.

00:03:10 Speaker 2

So if a mistake comes, then I open a new ticket and when that's done, then it's closed and gone and to unite that with this regulated world, where just

00:03:20 Speaker 2

There has to be a history where you also have to look into the past, and also long into the past.

00:03:24 Speaker 2

The M.

00:03:25 Speaker 2

D.

00:03:25 Speaker 2

R.

00:03:25 Speaker 2

writes down 10 years of retention periods and some still get into a skid, as they somehow cleverly agree.

00:03:33 Speaker 1

Mhm, and others who come to mind.

00:03:37 Speaker 2

Exactly, what is another point, especially with regard to 62304, the standard that prescribes that you should identify subcomponents, third-party libraries

00:03:49 Speaker 2

and should take care of them and should look, many stumble over the fact that they have so many and and these 100 pieces or 200 pieces possibly even, how are they supposed to do that somehow and that this is compatible with the regulations, which say yes, you should document it somehow, you should monitor them, Some are also fighting.

00:04:08 Speaker 1

Yes, maybe just about what you just mentioned with the one, I open a ticket and close it again and everything is done, so to speak.

00:04:17 Speaker 1

I have already experienced this myself as an auditor.

00:04:20 Speaker 1

There were companies, for example, I asked: ‚Yes, why do you have your requirements specification?' And then they said: ‚Injira.' Yes, and I said: ‚O.K.

00:04:28 Speaker 1

and where?' And yes, those are just the tickets that are included.

00:04:31 Speaker 1

And when you take a look, you realize that the tickets contain all kinds of things.

00:04:36 Speaker 1

Well, of course, sometimes it already says in it.

00:04:38 Speaker 1

Yes, what I implement, sometimes this, this use case is sometimes called.

00:04:42 Speaker 1

Sometimes it says, fix Bug X.

00:04:44 Speaker 1

Y.

00:04:44 Speaker 1

and sometimes it says that we need architecture refactoring.

00:04:48 Speaker 1

And now to exaggerate, and sometimes it says that someone would rather empty some dishwasher.

00:04:53 Speaker 1

And that's not surprising, because a Jira is a ticket system, so it's a task management system.

00:04:59 Speaker 1

And what is in these tickets are tasks and not necessarily specifications.

00:05:05 Speaker 1

so you will probably say computer scientists now, we have no type safety as far as this content of the tickets is concerned.

00:05:13 Speaker 2

Not exactly.

00:05:15 Speaker 2

So you can do it, you can at least assign types, no, that, no, by saying, that's a bug or that's a feature or that's, you could even say, that's or expand that and say, that's maybe even a kappa or something, no, and then accordingly

00:05:28 Speaker 2

react.

00:05:29 Speaker 2

There are possibilities, but actually the mixing is the danger and that you simply no longer keep track of what is actually here?

00:05:37 Speaker 1

Absolutely, and in the end, we also want to have documentation, it doesn't have to be documents in the classic sense, but I think now we might get to the tips, we'll talk about that later.

00:05:46 Speaker 1

Now let's actually look first, so you have now told us a few important things about what goes wrong, so you have talked about inconsistencies, I had just had the topic of Agenz, yes confusing tasks and

00:05:57 Speaker 1

yes, certain artifact types.

00:06:00 Speaker 1

You said that you briefly addressed the topic with the history, which is not maintained in a comprehensible way, on the subshell.

00:06:09 Speaker 1

Yes, now let's go, so to speak, that's the reality, now we're doing the other end of the scale, so to speak, what is actually required if you could give a short, maybe a short outline, although you had almost started that a bit.

00:06:21 Speaker 2

Yes,

00:06:22 Speaker 2

Exactly, so what is there to consider?

00:06:24 Speaker 2

Especially with standalone, i.e. software in general, we first have 62.304 as probably the most important standard in the context, which you should definitely know and pay attention to, which holds this process standard, which describes the software development life cycle and also marginal topics, how do you deal with changes, how do you deal with problems, bugs in development, You should just take that into account.

00:06:47 Speaker 2

A little bit more around it, especially for standalone software, because the 62304 has focused more on embedded software, there is now the 82304, so in the front only the 8 instead of the 6, which again makes the topics, how do you get to the requirement at all, i.e. the connection between the users actually or the stakeholders and the software development and at the end also addresses the validation of the software, a topic, which is otherwise a bit of a standalone software

00:07:15 Speaker 2

so lost between the regulations.

00:07:20 Speaker 2

A few points that also play a role in the 82304 are the accompanying documentation.

00:07:28 Speaker 2

We also have a lot of topics that are now coming out of the MDR that we have to take into account.

00:07:34 Speaker 2

The labeling, the instructions for use or even installation instructions.

00:07:39 Speaker 2

It also goes a bit in the direction of IT security, actually more.

00:07:42 Speaker 2

The

00:07:42 Speaker 2

you have relatively little else in the 62304, but actually more so on such a higher level, not very, not very detailed.

00:07:52 Speaker 2

Exactly, these are 2 essential norms now actually explicitly software developments.

00:07:57 Speaker 1

Of course, EMDR and IVDR must also be mentioned, even if that is not very productive now.

00:08:02 Speaker 1

But we have, for example, I think the sentence for both products that contain software or are software, this software must be state-of-the-art

00:08:12 Speaker 1

Verified and validated, yes, or yes, where the principles are also to be adhered to for the life cycle, also standard technology with verification, validation, risk management and I.

00:08:24 Speaker 1

T.

00:08:24 Speaker 1

Security was now also added.

00:08:25 Speaker 1

That is of course clear, that is too abstract, such a sentence, to be really directly enforceable now.

00:08:31 Speaker 1

And that's why we have the harmonized norms that you just mentioned, which take us a little more into depth.

00:08:38 Speaker 1

If we now perhaps take a very brief look in the direction of the U.S.A.

00:08:41 Speaker 1

.

00:08:42 Speaker 2

Yes, they have or are approaching the Europeans and the USA, so 62,304 is now also accepted there as a Recognized Consensus Standard.

00:08:56 Speaker 2

But of course they themselves have their own guidance documents for the documents that are to be submitted for a remarket submission.

00:09:05 Speaker 2

There is one that is now from 2005.

00:09:09 Speaker 2

and a more recent one, which is still in draft status from 2021, but which will probably become current in the next few years.

00:09:16 Speaker 2

There are a lot of overlaps with a few details, which are of course a little different or interpreted a little differently, or where you sometimes have to juggle a bit, how do you map them to each other.

00:09:27 Speaker 2

Sometimes a mapping table helps or you have to make sure that you explain it a little more at one point or another, so that it is understandable for both sides, so to speak.

00:09:37 Speaker 2

Exactly.

00:09:39 Speaker 2

Perhaps another very good guidance document, which I still like to use, is the off-the-shelf software guidance document, which avoids dealing with these third-party libraries not only in the software, but also in the development process.

00:09:52 Speaker 2

I think that's quite good, sometimes even better than the 62304.

00:09:56 Speaker 2

And there is also a guidance document on the validation of software, which is very broad, so to speak, not only covers validation in the actual sense, but all

00:10:06 Speaker 2

Activities in the entire life cycle that somehow check certain steps again, yes exactly, describe, are now actually all specific to the software life cycle.

00:10:17 Speaker 1

Yes, we also notice how the term software validation is understood differently, namely in a narrower sense.

00:10:25 Speaker 1

Yes, I have a product where I check whether it really meets the requirements.

00:10:31 Speaker 1

With a medical device, that would actually be more like a clinical evaluation or a usability, a mative usability evaluation, if you take a look at what.

00:10:40 Speaker 1

the F.D.A.

00:10:41 Speaker 1

Actually understood with software validation, it's actually all software lifecycle activities.

00:10:46 Speaker 1

Yes, we have in there what you have just described, so really of requirements, architecture, then the whole test stack, they pack everything under validation.

00:10:56 Speaker 1

You can see that it's actually to be understood in a broader sense, i.e. rather.

00:11:00 Speaker 1

as I said, as a metaphor for software quality assurance in general and not just a final test, for example with regard to clinical benefit, because they don't have that in there at all.

00:11:12 Speaker 1

Yes, now we've grazed the regulations a bit, so to speak.

00:11:16 Speaker 1

We could perhaps mention in U.S.A.

00:11:18 Speaker 1

die Cybersecurity Guidances.

00:11:20 Speaker 1

Yes, we're talking about the plural, but maybe not too deep and.

00:11:25 Speaker 1

Now I think it would also be important to understand what are your tips to deal well with these regulatory requirements, to avoid typical difficulties, not to shut down.

00:11:38 Speaker 1

So, what do you recommend to the companies you work with?

00:11:43 Speaker 2

So, a relatively important point is actually that you are, that you also, that you show the auditor the this traceability, no,

00:11:53 Speaker 2

That's something, then an auditor has to understand in a relatively short time or we too, no, when we check a software file, relatively short time, how everything is connected and the more clearly you explain it, the more clearly you describe it, the faster an auditor will find his way around.

00:12:10 Speaker 2

I think that's a very, very important point that is sometimes a bit underestimated, for example in everyday work, where you just deal with unit tests or the like at the very bottom.

00:12:19 Speaker 2

not to forget that you also have to create an understanding at a higher level for how it works.

00:12:24 Speaker 2

The standards also provide such a software development plan as an example of this, in that you can explain everything or even if you now see the FDA, which demands software description as a document, where you can also explain such topics, how is the software structured at all in order to be able to get started sensibly at all.

00:12:41 Speaker 2

I think that would be an important point.

00:12:43 Speaker 2

A clean separation from the activities as well, but that's nice

00:12:47 Speaker 2

I mentioned earlier with the Jira Tasks, where you empty the dishwasher or train a new DevOps developer or actually make a requirement for the software, that you have a clear separation.

00:13:00 Speaker 2

The tools usually offer ways to separate this well via labels, via certain types.

00:13:08 Speaker 2

If you start early and do it properly, then you can actually separate it quite well and process it cleanly.

00:13:17 Speaker 2

What I still often recommend is also to see, especially if you now maintain software requirements or architecture in Jira, Confluence or similar tools, that you think about it early on, how you can export it again as a PDF or export it in some form in case of doubt.

00:13:33 Speaker 2

PDF we know, you will still be able to read it in ten years, but how it is now with the Confluence page is such a question mark.

00:13:42 Speaker 2

There was only recently the turnaround from the

00:13:45 Speaker 2

Server, to the cloud version, some people have sweated a bit in their documentation about how they can keep it now.

00:13:52 Speaker 1

Yes, so very, very valuable.

00:13:54 Speaker 1

Maybe it's a thought of mine that I wouldn't recommend, so many praise themselves for agile development and often this is understood as a metaphor, yes, I can actually do what I want and that shouldn't be understood that way.

00:14:09 Speaker 1

So agility and if you have it right now, for example, Scrum, you have after every

00:14:13 Speaker 1

Sprint and potentially shippable increment and potential shippable is medical devices for us, software people just the software that is shippable and that's it only with the associated documentation and the artifacts, you mentioned earlier, we need the requirements, we need the software development plan, we need the software requirements, we need the software architecture, we need the trace, so the test stack and we need it in the software release and at the end

00:14:41 Speaker 1

of a sprint, we should be potential shippable and not just start documenting when we then celebrate the 10th anniversary.

00:14:47 Speaker 1

Sprint, because documentation is completely reduced to absurdity.

00:14:51 Speaker 1

Yes, that's what they are actually abusing so that the auditor is satisfied and if it didn't exist like that, I wouldn't document it either, and documentation is supposed to serve to ensure that we have clarity early on, that we can find errors early on, bring them out, before it is then implemented for a long time.

00:15:07 Speaker 1

So I think this thought, really,

00:15:09 Speaker 1

if you do Scrum, do Scrum properly and not just slamp.

00:15:14 Speaker 1

And maybe a second tip that I would add now, I don't know how you see it, many start to iterate quickly, especially in an agile context and I think it's good if we have relatively stable requirements and it's not

the job of the software, To raise them, apart from that.

00:15:34 Speaker 1

And if you also build such an upfront architecture, I know very little

00:15:40 Speaker 1

Architects, i.e. almost none at all, who are able to build a consistent and clean architecture almost continuously, yes, i.e. via sprints with an as yet unknown target image.

00:15:52 Speaker 1

So, most of them are challenged when they do the upfront, because it takes a lot of abstraction to be able to model something like that and then believe that you can do it in a more or less agile way.

00:16:03 Speaker 1

I experience most of them as

00:16:05 Speaker 1

overwhelmed and therefore such a small plea, i.e. a certain rough architecture at the beginning, which we can then iterate off afterwards, so to speak, in terms of tasks, I would recommend.

00:16:16 Speaker 1

But Daniel, maybe your thought about it, definitely.

00:16:20 Speaker 2

So if you‘re honest, probably most of them just didn‘t document it, no.

00:16:27 Speaker 2

Because when you start with something, you don‘t just start somehow, but you need a thought.

00:16:33 Speaker 2

If you have a cloud, is it in the cloud, is it a native app, where do you move there, where do you make the limit, no, which part do you want to have in the backend, which part in the frontend, no, if you‘re now with standalone software, but also the other way around, if you have a system development beforehand that tells you what kind of processors you have, no, or where you move, then of course you have, you have to

00:16:54 Speaker 2

deal with architecture.

00:16:56 Speaker 1

Absolutely, oh, I would have, it‘s good that you said that, because now you even have, you might have to distinguish between 2 architectures.

00:17:03 Speaker 1

We have this technical and system architecture, yes, where we have the different, yes, also technical levels, so to speak, what do I know, from the hardware access layer or a database layer or a business logic layer, a presentation layer or a mobile client.

00:17:19 Speaker 1

That would be a technical architecture and I would say that we often have

00:17:23 Speaker 1

a relief by the fact that there are already frameworks that accompany us in a certain direction and in a proven direction.

00:17:32 Speaker 1

But now we have the domain data model.

00:17:36 Speaker 1

Yes, so it's really about what is a patient at all, for example, what attributes does he have, what is a diagnosis, is it, how does a catalog refer to it again, what does the different states have,

00:17:51 Speaker 1

Do we also have the information in it that will be available for billing afterwards and there you need a deep understanding of the domain and that should then also be designed in such a way that later, if we want to integrate into other systems and then connect via Fire, for example, we have the necessary information in the necessary granularity.

00:18:10 Speaker 1

And these are things that you don't like to replicate, because it can be that the entire domain model disintegrates.

00:18:19 Speaker 1

This is despite the fact that the technical model, i.e. the technical architecture, remains unchanged.

00:18:23 Speaker 1

So, thank you very much for helping to separate these two considerations as far as architecture is concerned, because they are really 2 different aspects that require different skills.

00:18:36 Speaker 1

So, one person is really the technician and the other is the person who has understood the domain and then also.

00:18:46 Speaker 1

Interfaces to the outside world are an example where it becomes clear again how important this is or everything, so to speak, everything that has to be given to the outside.

00:18:54 Speaker 1

So it was very, very helpful, Daniel.

00:18:57 Speaker 1

Yes, because we are almost a bit over time.

00:19:00 Speaker 1

What would you recommend to people who feel that a second opinion or even support would do them good?

00:19:10 Speaker 2

Yes, of course they are welcome to contact us.

00:19:12 Speaker 2

Of course, that's clear, we are very happy to support you,

00:19:15 Speaker 2

to check the files, to do a review of certain documents through a gap analysis, but maybe just to get advice on what detail is concerned or something like that, things that don't just emerge so clearly from the norm, no.

00:19:30 Speaker 2

We are very happy to support you in setting up a corresponding process that is compliant with, for example, F.D.A.

00:19:37 Speaker 2

and M.D.A.

00:19:38 Speaker 2

Requirements.

00:19:39 Speaker 1

Absolutely, I think that's also a very central point, because the

00:19:45 Speaker 1

Efficiency also depends on such software development, of course, decisively depends on this process.

00:19:51 Speaker 1

So, to give you an example, when we are in an agile setup, we go through certain stages such as tests or approvals much, much more often.

00:20:02 Speaker 1

That means that if we had an inefficiency in there, because now to put it extremely, 7 people have to sign first, who may also be difficult to reach, then we would simplify this problem of inefficiency, namely with every sprint.

00:20:14 Speaker 1

And that's why such a process instruction, a development process, software development instruction or even a software development plan must be very tailor-made for the organization and also for the product.

00:20:26 Speaker 1

And so, among other things, Daniel helps, of course, maybe take a look at our websites.

00:20:32 Speaker 1

We already have a relatively large amount with us, also in terms of all the regulatory foundations.

00:20:38 Speaker 1

and ran in the audit or at the seminars.

00:20:41 Speaker 1

Daniel, may I also say that you also give the compact seminar Medical Software.

00:20:45 Speaker 1

I think you really have that again, you can go one step deeper.

00:20:48 Speaker 1

But in today's podcast, we should just touch on this topic, i.e. really look at what we observe, what keeps going wrong, what is required by regulation and with which tips is it particularly easy to meet this requirement.

00:21:02 Speaker 1

Yes, Daniel, there is only one thing left for me to say to you once again very, very heartfelt thanks.

00:21:07 Speaker 2

With pleasure.

00:21:07 Speaker 2

With pleasure.