What is an ergonomic tool?
- A “tool” is something to achieve a goal.
- It is “ergonomic” if it has been specifically optimized to be used by humans,
- taking our physical and cognitive limitations into account.

Think of a tennis rack. It is used to let a tennis ball fly with high speed in a chosen direction.
- Even an inexperienced player can achieve high ball velocities, much higher than by throwing
- Experienced players concentrate on goal (ball: which direction, how fast) he’ll forget the rack!
- As long as you concentrate on the rack, you’ll be a lousy player.
- And if the rack is broken, you’ll get a direct acoustic feedback while using it.

Today I’ll give an introduction to the design of ergonomic tools for the operation of an accelerator facility
I’m confident that operational tools based on these designs improve the availability of every machine.

I work now since about 20 years in the field of particle accelerators; since 10 years I am responsible for
the operation of the Swiss Light Source. I can assure you that I have been exposed to many very ill
designed operational interfaces. That of course qualifies me to give this talk. Another qualification of
mine is my interest in cognitive science. You’ll see that the limitation of our cognitive abilities are the
source for the state-of-the-art user interface design rules, the foundation for ergonomic operational tools.
Why do we care for good operational tools?
My main objective as an operations manager is the beam availability at my facility. Every beam interruption is scrutinized for its causes. If we look at out beam failure statistics of 2009, we do not find any contribution from “bad operational tools”, or do we? Well, about 3% of the outages were caused by faults of the operators. What does that mean? The only interaction of the operator with the accelerator is by his operational tools. If his action kills the beam, he was likely not able to properly predict the results of his actions. Or you could claim that his tools failed to communicate the risks of his actions to him in a timely manner. For every beam trip, independent of the cause, the operators actions are divided in three steps: First he has to diagnose what has happened. If something prevents him from making beam again, he’ll need to organize the repair. And finally he’ll recover beam. For the first and the last step he’ll depend on his tools. If the time to repair is in the same order than the time to diagnose and recover, then of course good tools can save a significant amount of time and therefore contribute to improve the availability.
Now we know that we need to have good tools. But what is required to get them? There are three aspects you need to understand to design a good operational tool.

- First you’ll need to understand the needs of the operator, who will use your tools. This involves psychology, cognitive science and the theory of perception. Often neither the application developer nor his customer, the operation responsible are familiar with this area. We’ll therefore focus on this aspect.

- Second you’ll need to know the software of your control system to build good operational tools. That is normally among the core competence of the application developer. Many contributions of this conference are covering this topic, I’ll skip it here.

- And third you’ll need a thorough understanding of the operational tasks, because finally the tool should help to achieve specific goals. We will therefore look into techniques to improve our knowledge on the operational task, too.
Often the application developer perfectly understands how his tool should interact with the facility;

But he does not have an appropriate model how the operator will interact with his tools.

We can use a simple model from interaction design: our operator will act on the tool, using the handles provided, like sliders for continuous control or buttons for discrete actions. His actions are driven by his goal and by his knowledge on how to achieve them. This knowledge can either be like a path, when he follows step-by-step instructions like a checklist. Or he has a mental map that allows him to find alternative routes to his goals. On his way to his goal he’ll perceive the information presented by his tool. He’ll interpret the information to find out if he’s approaching his goal or if he’s on the wrong path. The presented information can be a sharp picture to him, clearly showing his position on his chosen path. Or it can be a fuzzy, incomplete image, that needs to be completed by his interpretation.

We can learn from this model that the tool needs to provide the operator with the appropriate handles for his specific goals. The tools need to present the information relevant to the task and in a way that is suitable for the operator to be perceived as a clear picture. And we should support the operator to build up the knowledge required to achieve the goals.

But how can we do that?
Fortunately the profession of human-computer interaction has already solved this problem for us. There are well established guidelines for user interfaces for the design of ergonomic tools. Even better, we don’t necessarily need to understand the psychology behind them. Although often it’ll help if we are better aware of the limitations of human perception, in order to better apply these rules to the design of operational tools.

The user interface design guidelines are formulated differently in the various reference publications for human-computer interaction. But the guidelines all share the same concepts, which I’ll briefly introduce in the following slides with some examples. Your user interface needs to be:

Consistent, visually structured, task focussed, fault tolerant, responsive and it should support the memory of the user.

All of these aspects of user interface design are important, the operator will suffer if your tool fails in any of them. But more important he’ll likely do more errors and will be less confident to use the tools for the best performance of the machine.
Consistency and structure can be achieved by implementing a style guide for the design of the operational tools. You can define many aspects of the user interface: the general layout of the screens, like having a headline, where to place a help menu, or buttons to access related panels.

You may specify the type of handles to be used, their layout and how data should be displayed.

Colours should be defined and how they should be used. It is even useful to create a dictionary of the vocabulary to be used for buttons and labels to avoid ambiguity. Follow the simple rule “same name, same thing, different name, different thing”.

It is of course in most cases not sufficient to write such a style guide. You’ll need to train and support the application developers in how to adhere to it and you should control if they actually do.

But if you do that, you’ll get consistent and structured panels, and that will definitely make it easier for the operator to use them.
A funny thing about consistency is, that the very same tool can serve as a good or as a bad example, just depending on the context. Keep that in mind if you use tools from other facilities: they will likely not come according to your style guide. If you intend to keep them, then you should adapt them to your standard.

Consistency and structure are the most obvious aspects of the user interface design. For one it is obvious that inconsistent tools make it difficult to learn their usage. And it is clear that even a trained operator will be more likely to do wrong if his tools are not consistent. And it is a well obvious that structured information is easier to scan by human eyes for relevant data than unstructured text.

But those two aspects are obvious in a second sense, too: you can judge consistency and visual structure just by looking at the tool, without actually using them.
That is not the case with the other concepts. Operational tools should be task focussed. What does that mean?

Consider an example, the operators task concerning the RF. We’ll need to be able to switch the RF on after an interlock. Occasionally he’ll need to adjust the set-voltage or the phase of an RF station. The RF control tool was delivered by the manufacturer of the RF amplifier. It provides access to all parameters available for RF tuning and shows all measurement provided. The main panel allows to recover the RF from a trip and a sub panel, one of many, allows to set the high voltage power supply. The phase tuner acts on the cavity and has an independent control panel.

This is exactly the opposite of a task focussed application. The task of the operator requires him to open several screens where the handles he needs are buried in a large number of other handles that he is not supposed to ever touch. The information he needs is hidden too, in between a huge amount of information irrelevant to his task. This all does not mean that the application is bad, it is just not the appropriate tool for the task of the operator. It is may be still appropriate for the tasks of an RF expert.
What it obviously needs is an application tailored to be used by the operator. This application should show all actions and information relevant for the operator tasks and nothing more.

This dedicated panel provides the handles to switch the state of the amplifier, to set the cavity voltage and to adjust the RF phase. It shows as well the status information relevant to the tasks.
Ergonomic operational tools cannot fully prevent the operator from making mistakes. But they should make it hard for him to create severe problems and easy to recover from his mistakes.

An simple example is the SLS application to restore machine settings. If you restore a snapshot you’ll change dozens of machine parameters at once. You’ll have no guarantee that the restored snapshot will achieve a good machine state: it could have been a snapshot of a bad machine state, it could be corrupted, or it may does not work well with the current state of other parts of the machine.

If the effect of an action on the machine is severe, then it is a good idea to provide means to revert the action. Here an Undo button does the trick: the application saves a temporary snapshot, before it actually restores the selected one. If the restore does not lead to the desired result, the operator just presses Undo and the tool will restore the temporary snapshot to restore the previous machine state.

Providing means to recover from mistakes will increase the operators confidence in his tools. This will not just avoid severe problems, it’ll encourage him to use the tools more often to enhance the performance of the machine.
Responsiveness of software has proven to have a high impact on user satisfaction. More important an responsive application will tell the operator if his attention is currently required or not. Human time expectations range over several decades:

- Within a tenth of a second we expect an immediate response. That can just be a button to appear pressed. If it takes longer it will irritate us and if the application continue to show this behaviour it’ll frustrates us, because it violates our sense of causality.

- In about one second we expect a feedback that something has actually happened, a status feedback. We need to be confident that our actions actually triggered the desired process.

- A process that takes longer than 10 seconds to complete should provide some kind of updating time estimate for the operator. This assures the operator that the process is still ongoing and releases his concentration to focus on other topics.

The application does not need to be fast in performing the task in order to be responsive.

The panel to start-up the SLS electron source for example just displays the remaining heating time of the cathode in seconds. It cannot do anything to accelerate the process, but it can clearly tell the operator when the process will be finished, to set him free to do other things until then.

An even simpler implementation of responsiveness has the application to standardize the magnetic fields with the power supplies. It shows two types of information: the status of the process, where “Cycling” indicates that it is still ongoing, and the read back of the magnet current, that tells the operator that the process is really still acting on the power supply and at the same time allows the operator to estimate the remaining time for the process.
Consistency and structure is already a good measure to support the operators memory. The operator deals with hundreds of screens, a headline in each to indicate what they are for is appreciated. He’ll sometimes not find the correct screen immediately, therefore links to tools that support similar tasks are often useful. It is much easier to recognize the right action from a menu, than to recall it from the memory; if the number of choices is limited to the relevant it will make it even easier.

It is useful to provide checklists for long procedure: what needs to be done and what has been achieved. You can either integrate them into your logbook or you can provide checklist like applications.

The tools should gather the task relevant information for you, not just to display it.

For example a tool that allows you to recover from a failure should do the systematic part of the failure reporting for you.

Do not rely on the operator to document the failure from his memory. Here we see the application to diagnose a magnet power supply trip. The application allows to create a trouble report that can be saved to the electronic logbook. The logbook will e-mail the information to the power supply group for their power supply failure tracking. This saves time for the operator and assures that the failure reports will contain complete and consistent information.
If we succeed to make our tools according to these design guidelines, we will have happy operators and improve the beam availability and performance.

But how do we know that we succeed?

The problem with these design guidelines is, that they are just guidelines:
Often it will be still difficult to determine, how they can be properly applied to a specific problem.
Sometimes they may even contradict each other for an application.
In addition the operators are maybe similar compared to the humans in total, but they still may disagree when it comes to what is useful in an application and what is useless.

What can we do about it?
Again human-computer interaction tells us the right solution. Although in this case we can as well just use common sense: When you design a tool to be used by humans you’ll best go for an iterative design. That means you’ll design the tool to the best of your knowledge, You’ll implement it according to your design and then You’ll evaluate the implementation. The results of your evaluation will tell you, what should be re-designed and the circle continues.

In human-computer interaction it is stated that this process ends either if you run out of time or out of resources. For an accelerator facility it may continues until the end of the operation of the facility. But how to evaluate?
Task analysis and usability testing are two well established techniques to do exactly that.

In task analysis you try to understand your procedures. It has been developed primarily to create documentation, but it can be used to develop a conceptual model of the operation of your accelerator. That is an idealized view on how operation should work: what are the actions, what devices are involved and what are the relations between the action and the devices. A conceptual model should be simple and focussed on the tasks of the operator.

Usability testing has the goal to understand your tools. It allows you to asses risks in the usage of the tools and shows how they can be improved.

Both methods require you to observe the operator performing his tasks, while he is watched and somebody’s writing down what he’s doing. This is called “think-aloud” technique and it is useful to have other operators watching and asking questions, since that will reveal more of what the operator thinks what he’s doing while he’s using his tools. Does this sound familiar to you? Yes, we call it “operator training” at the Swiss Light Source. Once or twice a month we reserve 90 minutes beam time to do hands-on training with the operator: creating real randomized machine problems on demand from a “sabotage” applications. While one operator is assigned to fix the problem, several others are watching. His duty is to diagnose the problems, fix them and to recover beam. He has to explain what he is doing or attempts to do and to document everything. While the primary goal is to train the operator in failure analysis and beam recovery, it serves at the same time for a task analysis and for usability testing.

This has been proven very useful in the past years to identify risks in our operational tools and to improve their ergonomics.
• **Evaluate and iterate**
  • Task analysis to optimize procedures
  • Usability testing to optimize tools
  • Operator training helps to do both

• **Ergonomic tools increase beam availability & performance**
References

Books to start with: