RISKNOWLOGY®

BEST PRACTICE

SIL and Functional Safety in a NUTSHELL

Everything you need to know to get started

Dr. Michel J.M. Houtermans

www.risknowlogy.com/best-practice

RISKNOWLOGY®

BEST PRACTICE

SIL and Functional Safety in a Nutshell

Everything you need to know to get started

Share this eBook

By Dr. Michel J.M. Houtermans

www.risknowlogy.com/best-practice

Copyright 2014 Risknowlogy. All Rights Reserved.

No part of this publication may be reproduced, stored in or introduced into a retrieval system, or transmitted, in any form, or by any means (electronic, mechanical, photocopying, recording, or otherwise), without the prior permission of the publisher. Requests for permission should be directed to riskfree@risknowlogy.com, or mailed to Risknowlogy, Baarerstrasse 11, 6300, Zug, Switzerland.

All pictures are copyright protected by their respective owners. Where possible we refer to the rightful owner. Sometimes we simply do not know the rightful owner and cannot make a reference. If you know who the rightful owner of the copyright protected material is then please contact us at riskfree@risknowlogy.com and we will update our publication.

Please do not participate in or encourage piracy of copyrighted materials in violation of the author's rights. Purchase only authorised editions.

Risknowlogy is committed to publishing works of quality and integrity. In that spirit, we are proud to offer this book to our readers; however, the words are the author's alone.

First eBook Edition: July 2014

ISBN: 978-3-9524357-0-0

Table of Contents

Foreword

Dear reader, in this book I try to explain functional safety in a nutshell. Many standards, guidelines and other publications exist that talk about functional safety. All of them with their own level of detail. In this short book I try not to go into the details of what has to be done according to what standard and what not. I try to explain main functional safety concepts so that you know everything you need to know to get started.

Functional safety is not rocket science and once you understand it and apply it a few times in practice you will notice it is just good engineering practice. This book tries to explain that good engineering practice. Functional safety standards can be confusing and contradicting at times and sometimes they require things that make no sense. You do not need to understand the standards in order to apply and be good at functional safety.

If you understand the concepts explained in this book you can apply functional safety into your organisation and to your products, with or without following the exact requirements of standards. Actually once you understand how functional safety works you most likely will go beyond what standards say and create your own functional safety organisation of excellence. And if you do that, well then you are well on your way in becoming a longterm winner as opposed to a short fuse.

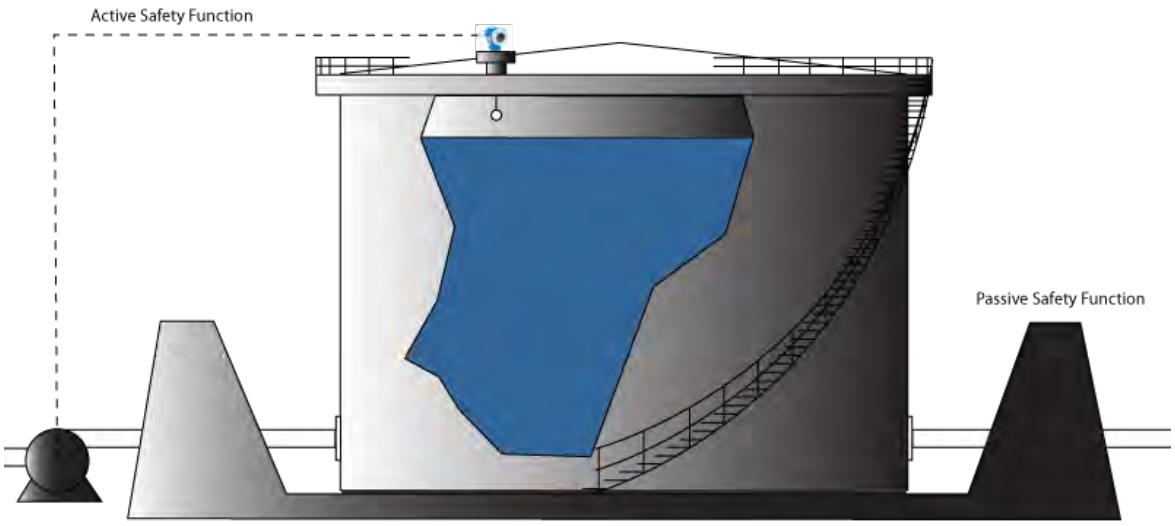
Enjoy this first Risknowlogy Best Practice book.

What is Functional Safety?

SIL and functional safety go hand in hand

Functional safety is a property of an active safety function, carried out by a safety system. An active safety function detects some kind of undesired situation and then takes action. For example an airbag that activates when the car detects a collision. Or an overfill prevention system of an oil tank, where a level measurement is used to decide whether the inlet flow needs to be stopped, or not.

Functional safety does not apply to passive safety functions. For example it does not apply to the exit signs that you can see above emergency exit doors. And also not to a spill containment system around the oil tank. See figure below.



A passive and active safety function for an oil tank

Products used in safety functions can fail. In theory they can fail in three different ways. When something wears out it can fail suddenly but we never know exactly when. For example the brakes of your car can fail after many years of operation but when exactly is unknown. We call those types of failures **random hardware failures**. Hardware can also fail due to environmental events like earthquake, flooding or lightning. When hardware fails in this way we call it a **common cause hardware**

failure. The real trouble maker in industry though is the so called **systematic failure**. Imagine you write a piece of software for your safety device but there is a bug in the software. You sell one hundred of these devices to your client but your client will end up with the same software bug in all one hundred devices. There was somewhere a systematic problem (wrong specification or wrong design or not tested properly, etc) and now we ended up with one hundred wrong devices.

Now that we know how products can fail we can give a definition for functional safety:

> A safety function is one hundred percent functionally safe if all its **random**, **common cause**, and **systematic** failures are one hundred percent under control and therefore cannot lead to malfunction of the safety function.

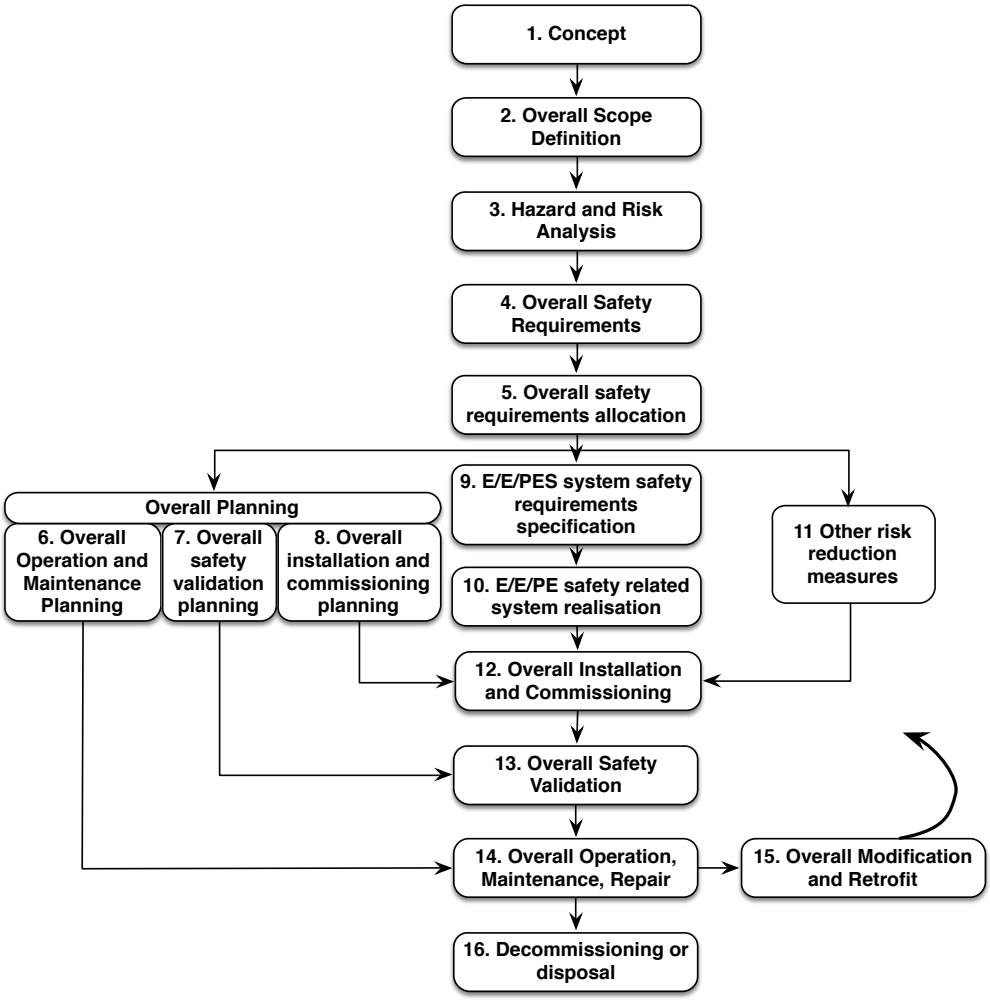
And, when the safety function cannot fail, it cannot injure or kill people, degrade the environment, or destroy assets. One hundred percent functional safety does not exist in this world but there are international standards, like IEC 61508, IEC 61511, IEC 62061, and others, which define rules on how to achieve SIL 1, SIL 2, SIL 3, or SIL 4 functional safety. **SIL** stands for **safety integrity level** and is a performance measurement of functional safety and thus of how well the safety function achieves functional safety.

SIL and functional safety can be applied to any active safety function that you can imagine, but it is mainly used where electrical, electronic and programmable electronic technologies play a role. And that is nowadays just about everywhere. You will find SIL in almost any industry from the automotive to the lift industry, from the process to the chemical to the oil and gas industry. SIL is applied in medical devices and in machinery, in airplanes and trains. And if SIL has not yet arrived in your industry then don't be surprised if it soon does.

Functional Safety Management

Any SIL project starts with FSM

In order to achieve functional safety it is first necessary to have excellent **functional safety management** in place. The objective of functional safety management is twofold. First of all it is necessary to define all technical and managerial **activities** that need to be carried out over the full life of the safety function and the safety system. The second objective is to make somebody **responsible** for these activities. Basically functional safety management manages that the right people are doing the right work at the right time with the right tools following the right procedures and guidelines. Functional safety management is driven by a **lifecycle** model and carried out by one of the functional safety managers.



IEC 61508 overall lifecycle model

The lifecycle model identifies all **phases** the safety function will go through during its full life. Once the phases are known it is easy to identify the necessary work activities in those phases. For example if the lifecycle phase is hazard and risk analysis then the work could be: perform HAZID, perform a FMECA, perform a LOPA, etc. Once the work activity is known it is possible to identify the required competencies for the professionals to carry out this work. If the required competency is not available in your company then you know that you either need to train your personnel or that you need to hire new professionals with the right competencies.

In order for your employees to carry out the work it is necessary to give them the right **input** information to do the work. Most of the time the input to a phase in the lifecycle is one or more documents. Once the work activities are done an **output** to that phase is created. Usually the output is one or more new documents and at some point the output is the real hardware and software of the safety system that carries out the safety function. In other words, the lifecycle phases help you identify which **documentation** or information is needed to do the work and which documents or output is created when the work is carried out.

When people work they can make mistakes (and that is ok, it can happen). One of the principles of functional safety is to carry out **verification**. Verification needs to be carried out for each lifecycle phase. It needs to be carried out by an **independent** person, i.e. a person not involved in the work in that particular phase. Verification is a technical task and checks whether the work activities were carried out correctly. In this way we can independently assure that the work was performed correctly. At least, if two competent professionals are involved, then the chances of making a mistake are getting smaller and smaller.

There is another problem that we need to solve though. How do we know that the work and the verification have actually been carried out? In order to answer this question we need to carry out a **functional safety assessment**. Functional safety assessments, like verification, also need to be carried out for each phase of the lifecycle and need to be carried out by someone **sufficiently independent**. The more SIL (i.e. the more functional safety) is needed, the more independence is needed for the assessment. In practice this could mean an independent person

working in the same department, in another department, or even in another organisation.

The functional safety assessor checks that the person who carried out the work is competent to carry out that work, checks that the work was actually carried out, checks that the person who carried out the verification is competent to carry out verification and checks that the verification was actually carried out. Furthermore, the functional safety assessor checks that the correct input information has been used to carry out the work, that the same input information has been used to carry out the verification, and that the correct output has been created. In other words, functional safety assessment is a kind of completeness check and is not technical in nature. Functional safety assessment assesses what the functional safety manager manages. The assessor assesses that the right people have done the right work at the right time with the right tools following the right procedures and guidelines.

Basically we have now the complete lifecycle under control as we systematically go through each phase of the lifecycle. For each phase, we carry out the work, we carry out the verification and we carry out the functional safety assessment. Once one phase is done, we move on to the next phase and do the same trick.

There is one special phase in the lifecycle and that is the **validation** phase. In the functional safety world validation is performed to find out whether the specified safety function is actually the one that is delivered. In other words validation is done on the real hardware and software, after it has been installed and commissioned. Because validation is a phase it needs to be verified and assessed.

In this way everything is under control, life is good. Until one day somebody wants to make a change. In the functional safety industry not every change is a change. Sometimes we call a change a **modification**.

Modifications are changes to work that has been carried out and has been verified. In other words, these are changes made to work that the professional and the verifier have previously agreed to be correct. In practice it means that we often go back to phases that have been completed already and start to make changes in those phases. Those type of changes are called modifications, and we definitely need to have modifications better under control. We need to have a modification procedure in place before we start all the lifecycle work, so that when the

day comes that we need to make a modification, we know how to perform the modification.

A modification procedure basically consists of a few simple steps. First of all we cannot just make modifications, but first a request to make a modification needs to be made which then needs to be submitted for approval. Once the request is approved it does not mean that the modification can start. It means that an **impact analysis** needs to be carried out. Basically for each modification we need to understand the impact. The impact analysis should answer some basic questions. Which safety function is impacted? Which hardware and/or software is impacted? Which documentation is impacted? Which tests need to be repeated? Which people need to be involved and so on. We need to have a complete picture of what it means to make this modification.

In practice the impact analysis is like a small project plan. If we want to modify this then we need to go back to this phase, we need to modify this hardware and that software, update those documents, do these tests, inform those people and if we do all of that then the safety function will be the correct safety function again and will be functionally safe again.

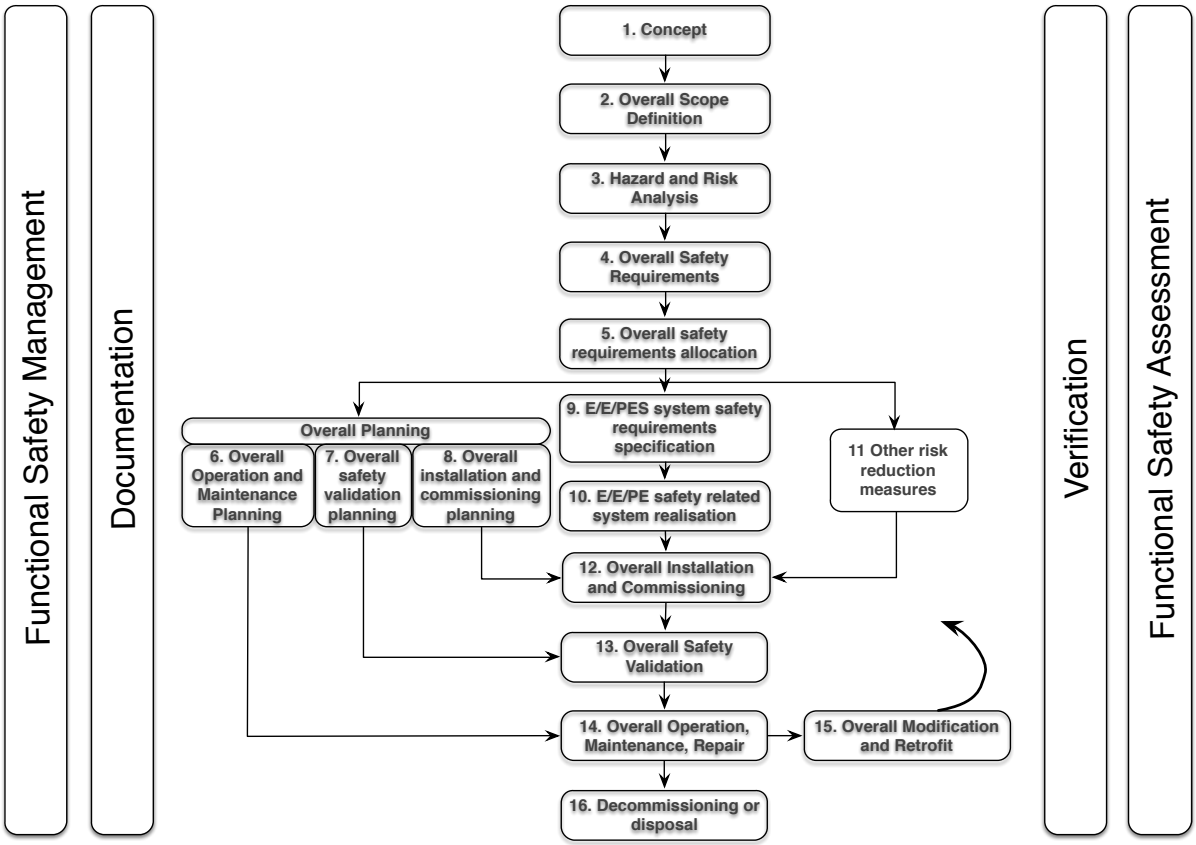
Once the impact analysis is done, the modification can still not start. First the impact analysis needs to be approved. And after approval of the impact analysis the real trouble starts. Now the modification needs to be carried out in practice and thus the real work needs to be done. But this should not be a problem as we know exactly what is impacted by this modification and what needs to be done to carry out the modification, so that in the end functional safety is achieved again.

In this way we have the lifecycle fully under control. But functional safety management needs to take care of one more thing. No matter how competent professionals and verifiers are, we need to help them avoid making mistakes. A good functional safety management system introduces measure to avoid failures. Measures to avoid failures help us deal with systematic failures and systematic failures are made by people.

These measures to avoid failures can be implemented for each phase of the lifecycle and depend on the kind of work that needs to be performed in each of them. A few measures to avoid failures have already been mentioned. For example, putting competent people on the job, doing verification and assessment, document everything. But also requiring the use of tools and checklists, separating safety from non-

safety, and so on are all easy measures that can help avoid making mistakes.

Functional safety management makes sure that the above is all thought about before we start with a functional safety project. Besides a project manager, companies will need one or more positions for a **functional safety manager**. Now that functional safety management is under control we can start with the real work, the technical work. In practice this means we need to carry out a hazard and risk analysis.



IEC 61508 Lifecycle with FSM, Documentation, Verification and FSA for each phase

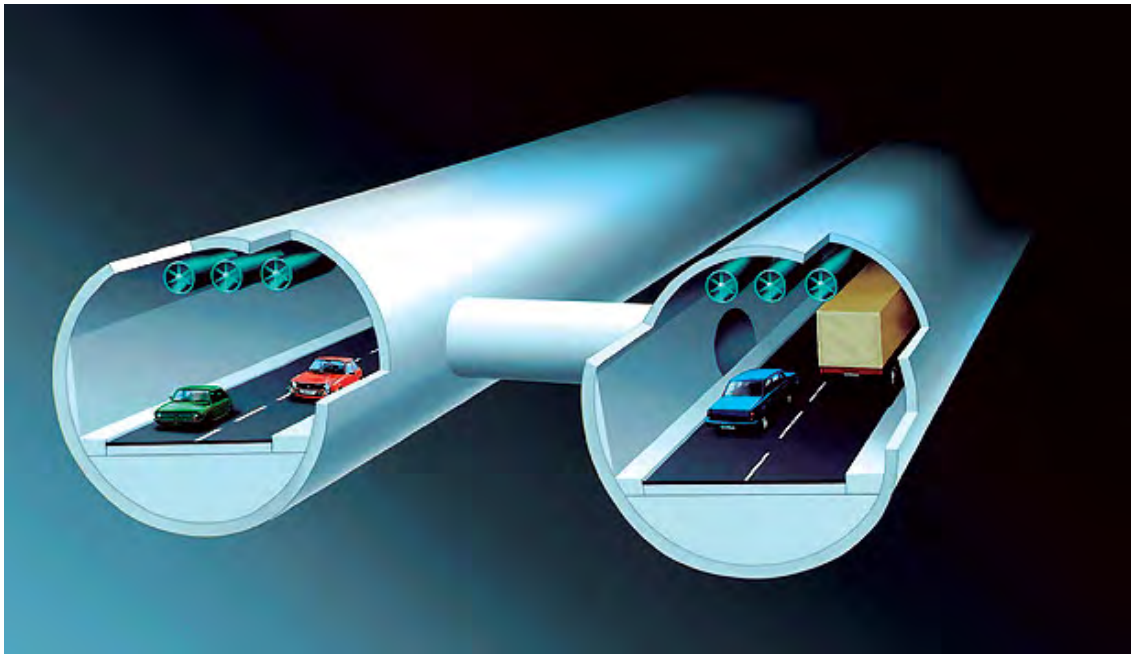
Hazard and Risk Analysis

If you don't know it, you cannot control it

From a functional safety point of view hazard and risk analysis is very simple. **Identify** the problem, **analyse** the problem (which means establish the risk associated with the problem) and where necessary **reduce** the problem to an acceptable level of risk. The more the risk needs to be reduced, the higher the SIL might be for the chosen safety function that implements this risk reduction. This is what functional safety wants you to do. In theory a simple task, in practice this can be months or even years of work. In practice there are also a number of approaches and techniques that can be used. And that does not make this task easier.

Hazard and risk analysis can be carried out qualitatively, quantitatively or semi-quantitatively. There are many techniques available to support hazard and risk analysis. Well known techniques are Risk Matrix, Risk Graph, FMEA, FTA, HAZID, HAZOP, AHA, OHA, ETA, LOPA, Dispersion Modelling and so on. Functional safety does not care which techniques you use as long as you can identify the problem, analyse the problem and then reduce the problem. But it is recommended to choose techniques that work for your company and in your industry and then apply them consistently. So do not change your risk approach from project to project. That is not good practice.

The end result of hazard and risk analysis might be the identification of a number of problems with high risk potential that need to be reduced. We can choose any approach on how to reduce the risk as long as it gets reduced to an acceptable level. We can come up with all kind of solutions. We can change the technology the manufacturing process uses to produce a product, i.e., change the process. For example, a low pressure manufacturing process instead of a high pressure process. Or instead of one tunnel shaft with opposing traffic we make two tunnels, one for each direction. We can change the basic materials we use. Maybe we are using flammable and explosive materials that can be replaced with materials that are less flammable and non-explosive. We can introduce new procedures, we can train our people. Or we add, and this should be a last resort, additional safety systems which feature active safety functions to increase the safety of our process.



Tunnel with twin tubes for opposing traffic - Inherent safety by design (Ref. Herren Tunnel, Lübeck, Germany)

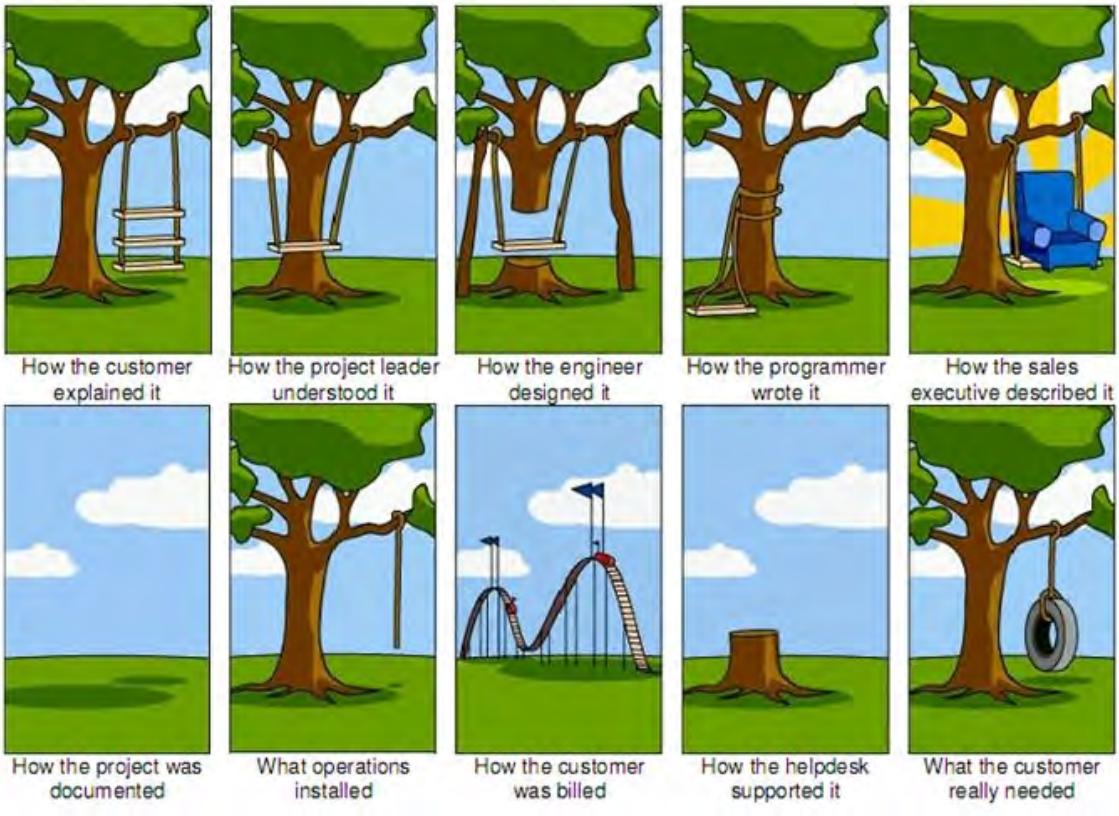
If we decide to use additional safety systems then it is very important to clearly define and specify what those safety functions need to do. Now is the time to write a so called **safety requirements specification** or **SRS**. The starting point of each safety requirements specification is the definition of each safety function. It is very important to clearly describe what the safety function is intended to achieve. If the intention of the safety function is not clearly defined, then all the subsequent work related to the design, development, manufacturing (and verification and assessment) of the safety function will be meaningless. You could be doing a lot of work, for nothing.

A good safety function description consists of five elements. First of all what is going to be measured. Are we going to sense the flow, the temperature, the height, the pressure, the speed, the vibration etc? Second, what logic is going to be solved with this measurement. If the temperature goes over 100 °C then... Thirdly, what action needs to be taken when the logic actually solves. If the temperature for example goes over 100 °C then switch off the burner. Fourth, how fast does this function needs to be carried out. For example do all of this within two seconds. These four elements represent the **functional description** of the safety function. For this function you can build in different levels of **safety integrity**. The fifth and last element therefore describes the

safety integrity level. Build this safety function according to the rules of SIL 2. A full safety function description can now read, for example, like this:

> Measure the temperature in the vessel and if the temperature goes over 100 °C, switch off the burner. Carry this function out within 2 seconds and on SIL 2 level.
>
> or like this
>
> Measure the smoke in the tunnel and if the smoke goes over 100 ppm, maximise the ventilation and close the inlet to the tunnel within 10 seconds and on SIL 1 level.
>
> or like this
>
> When any object comes within the safety perimeters of the machine, stop the machine within 200 ms. Do this on SIL 3 level.
>
> or like this
>
> When the train is 2 minutes away from the road crossing, close the road barriers within 1 minute. Perform this on SIL 4 level.

Please note that this description only describes the **intention**, not the **solution**. This is very important as sometimes we forget what we want to achieve and focus so much on the desired solution that we end up with a solution that becomes too expensive, complicated, or in the end does not do what we wanted to achieve in the first place. Focus on the intention, not the solution.



Understanding your requirements (Ref www.ricardoluizlopez.com)

Now that the intention of the safety function is clear, it is possible to define all the **safety requirements** of the safety function and safety system required to achieve this intention. Typically the requirements that need to be defined in the **safety requirements specification** include:

- Functional safety requirement;
- Environmental requirements;
- Basic safety requirements; and
- User documentation requirements.

This safety requirements specification needs to be carefully reviewed as it will be the basis of the safety function that will be designed, manufactured, installed, commissioned, operated, maintained and repaired. We do not want to do all this work when there are mistakes in our safety requirements specification. Imagine you write in the safety requirements specification that the system needs to be SIL 2 but in reality it needs to be SIL 3. If you do not catch this systematic failure you

would design, build, install, commission, validate, operate, maintain and repair the perfect SIL 2 but wrong safety system. Not funny.

In practice the safety requirements specification is often written on different levels. The first level is done by the end-user. In many industries though, the end-user is not the party that actually designs and builds the safety function and safety system. Most of the time engineering partners, system integrators and product suppliers are involved. Each can have their own version of the safety requirements. A version that reflects their required level of detail.

As soon as the end-user has his safety requirements specification and other parties are getting involved in actually designing and building the new function, three documents are becoming increasingly important. These three documents are the safety plan, the verification plan and the safety requirements specification. Again? Yes, again. We will explain why in the next chapter.

We ended this chapter on hazard and risk analysis, with writing the safety requirements specification. In other words, the first version of this document is actually the end result of the hazard and risk analysis. This is how it should be. The hazard and risk analysis job is not finished unless we have a first impression of what we want in terms of safety. Later we can work out the safety requirements specification in more detail.

Planning

Getting ready for your future safety system

The objective of the **safety plan** is to describe how we think functional safety will be achieved on the particular project. The safety plan describes the applied lifecycle, the phases of the lifecycle that we are responsible for, the work activities per phase, the kind of professionals and their competencies needed, the input and output documents, the modification procedures, etc. Do you recognise this information? Yes of course, this is the functional safety management information described earlier. The safety plan describes how we will manage to achieve functional safety on the project.

Since the safety plan describes the safety lifecycle activities, the work, we need to check that the work is done correctly and the correct work was done. In other words we need to do verification and validation, and thus we need to have a **verification plan** and **validation plan**. The objectives are very simple. Verify that the work was done correctly, validate that the correct work was done. Per lifecycle phase we describe who is going to perform the verification/validation, how they are going to do it, which tools, procedures, guidelines they need to use or follow, and so on. It is not rocket science. It is just project management but with a focus on solely functional safety.

The safety plan, verification and/or validation plan are management documents. This is how we manage our work. The safety requirements specification is a technical document. We already talked about the version that the end-user made. But as discussed, the end-user usually does not design and build the safety system. Engineering companies and/or system integrators do.

> A lack of planning on your part does not constitute an emergency on my part...
> Or maybe it does! Damn.

Very often there is a mismatch between what the end-user specifies, and finds perfectly clear, and how the designers and builders interpret the safety requirements from the end-user. It is good advice, and good engineering practice, to have the end-user send their safety requirements

specification to the engineering partners / system integrators and have them write their own version of the safety requirements specification. A version which interprets the end-users thoughts and translates them into requirements as they understand them.

Once the safety requirements specification is “translated” send the new version back to the end-user and let them verify it. Not only has verification automatically taken place, you can also be assured that the end-user has read and understood what it is the engineering companies / system integrators are actually going to design and build. A win-win for everybody as it will lead to no discussions once the system is being validated. Money and time well spent at the beginning of the lifecycle and not wasted at the end of the lifecycle.

The Hardware

Make it work with or without software

Now that it is clear what we want to build, and how we are going to do it, it is time to go and deal with the hardware. In the functional safety world there are a lot of hardware concepts that need to be dealt with. Some of these concepts are applicable to the **complete safety function**, i.e., from sensing element, to logic solver, to final element or actuator, and everything in between. But there are other hardware concepts that apply to **subsystems** only. A subsystem is for example only the sensors, or only the logic solver. Lets start with the safety function or loop concepts.

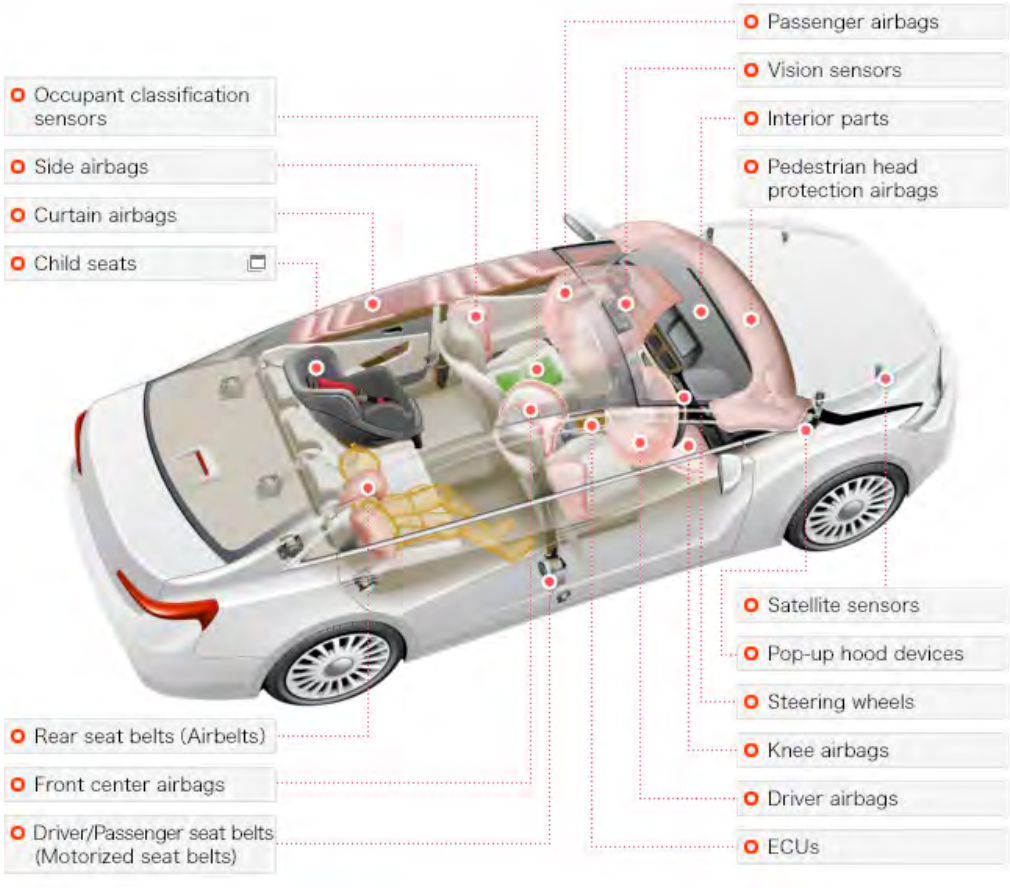
A safety function can be **de-energise to trip** or **energise to trip**. De-energise to trip functions do not need external energy (electrical, pneumatic, etc) to carry out the safety function. This is also known as **fail-safe,** but fail-safe is more than just a safety function that does not need energy to carry out the safety function. Energise to trip safety functions always need energy to carry out the safety function. An energise to trip safety function is, by definition, not fail-safe.

Spring return actuator and a sprinkler pump house

Furthermore a safety function can be a **low demand**, **high demand** or **continuous mode** function. Low demand and high demand are functions, as their names imply, that work upon demand. In other words the process that we are trying to protect places a demand on the safety function. The difference between the two is the frequency with which they are executed. With a low demand function the demand is placed less than once per year. With high demand mode functions the demand is placed more than once per year. It could even be every day or several

times a day. The continuous mode function is different. The demand is continuously there. In other words, the safety function is continuously required.

Lets demonstrate low demand, high demand, and continuous mode with a practical example that everybody understands. The airbag in your car is a low demand mode function (at least I hope it is). The brake in your car is a high demand mode function. And the steering wheel is a continuous mode function. When the failure of a function can lead to an immediate problem then you can be assured it is a continuous mode function. When the steering wheel fails, you will immediately loose control over the car and bump into something. When the brakes of the car fail you are only in trouble the moment you need to brake. But you need the brakes a lot. At least several times per drive. When the airbag fails you are only in trouble when you bump into something, which with a little bit of luck, will hopefully never happen.



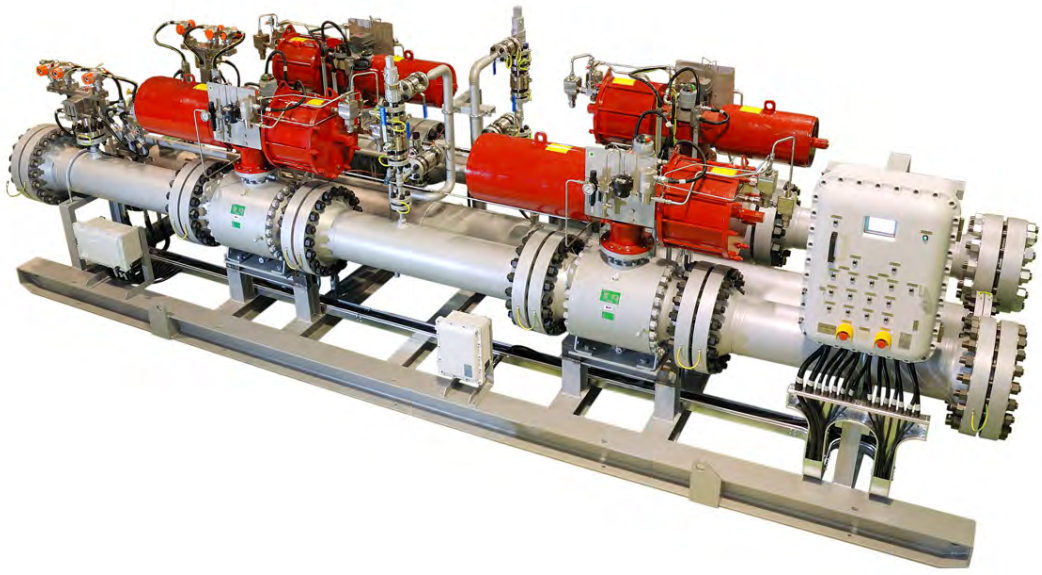
Low demand, high demand, and continuous safety systems in a modern car (Ref. www.takata.com)

Another hardware concept that applies to the complete safety function is the so called **target failure measure**. Each safety function can, and is, allowed to fail dangerous. A safety function has failed dangerous when it does not work when we want it to work. For each safety function a performance target is set based on the **probability of fail dangerous** or **PFD**. The higher the SIL, level the lower the probability of a dangerous failure. Each SIL level higher means a PFD that is a factor of ten lower than the previous level. The higher the SIL level the more often the safety function will be available when we need it to be available.

> Note: An end-user should not want to have safety functions with high SIL levels. The reason is very simple. When an end-user has a need for a safety function with a high SIL level it means the end-user has a high risk problem that needs to be solved. And you don't want high risk problems in your process. In that case you want to design a more inherently safe process.

When we divide our safety function into subsystems we have a whole new set of concepts that apply. Remember a subsystem could be a set of sensors or the CPU modules of a logic solver, and so on.

A subsystem can be redundant. We can have two or three sensors in a subsystem. **Redundancy** means that we use multiple means to carry out the same, or part of, safety function. A redundant subsystem can be diverse. **Diversity** means we use different means to carry out the same, or part of, safety function. For example different brand sensors, one from manufacturer A and one from manufacturer B.



Redundant but not diverse HIPPS valves

(Ref. www.foothills.ca)

The next concepts that we can discuss are **voting** and **hardware fault tolerance**. Voting is a very important concept as voting can basically change the redundancy benefits. Let's assume we have a subsystem with three temperature sensors. All three measure the temperature and reaching the temperature limit of 100 ^{o}C will activate the safety function. The question though is, how many of these sensors need to measure a temperature of over 100 ^{o}C? We can design the subsystem in such way that one, two, or all three sensors need to measure over 100 ^{o}C in order to activate the safety function. This is called voting and in this example the voting is respectively 1003 (one out of three), 2003 or 3003.

In practice voting has two implications. In order to explain this we first need to discuss the difference between a **dangerous failure** and a **safe failure**. Let's take the example of the temperature measurement again. A dangerous failure is one where the sensor fails in such way that it will never report a temperature above the 100 ^{o}C limit. In other words the sensor will never know whether the actual temperature is above 100 ^{o}C. A safe failure is a failure where the temperate sensor fails in such a way that it will always give a temperature over 100 ^{o}C. In other words we

carry out the safety function even if in reality the temperature is below 100 °C.

Now lets go back to our voting example as this means two things in terms of voting. If the voting is 1oo3 then two dangerous failures are allowed to happen and the subsystem can still carry out the safety function. This is great for functional safety and SIL. But, if only one sensor has a safe failure the subsystem will have already carried out its safety function. This is not good as it causes a so called **spurious trip**. The safety function gets activated when we do not want it to be activated. 1oo3 has a good SIL level but a bad spurious trip level or **STL**.

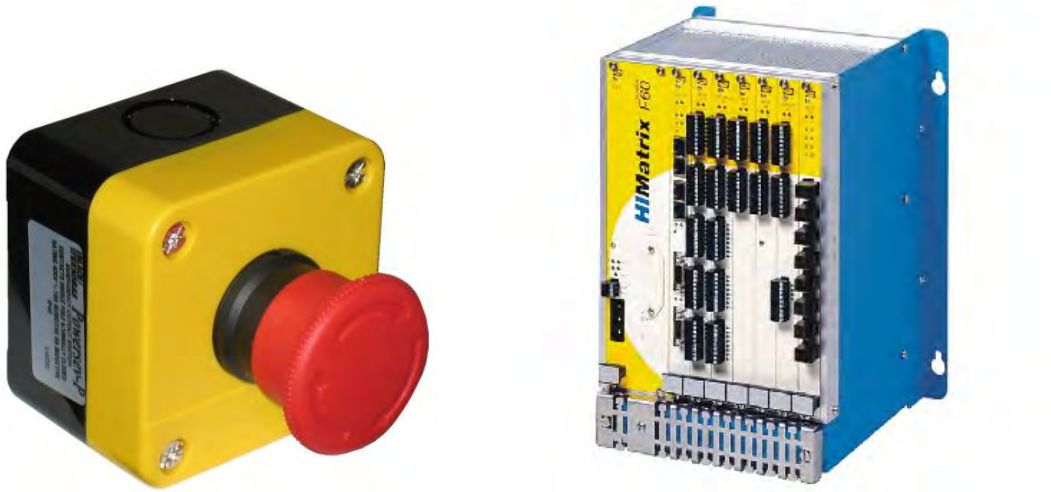
With a 3oo3 voting system the story is the opposite. If only one of the sensors fails dangerous the subsystem cannot carry out the safety function any more. This is bad for functional safety and bad for SIL. But a 3oo3 subsystem can handle two sensors with a safe failure. No spurious trip will happen with two safe failures. We need three safe failures to cause a spurious trip. 3oo3 has a low SIL level but has a high STL level.

The number of dangerous failures that a subsystem can handle is expressed as the hardware fault tolerance or HFT. A HFT of X means it can handle X dangerous failures and it still works. X+1 failures and the subsystem and thus the safety function have failed dangerous. The voting decides how much hardware fault tolerance a subsystem has. Voting removes the benefits of redundancy if all devices in the subsystem are needed to activate the safety function.

Not sure how much voting and HFT this is now…

Hardware fault tolerance tells us something about the architecture of the subsystem. But not one hundred percent. A hardware fault tolerance of zero could mean a 1001, 2002, or 3003 architecture. A hardware fault tolerance of 1 could mean a 1002, 2003, 3004, etc. architecture.

The devices that can be used in these subsystem can be either **Type A** devices or **Type B** devices. Essentially a Type A device is internally a simple device. If this device has an internal failure then we understand what happens to the device. A Type B device, on the other hand, is internally a complex device. It is so complex that for some of the internal failures we do not understand what can happen to the device. In practice all mechanical safety devices (relay, switches, push buttons, valves, etc) are Type A devices. All devices with software inside are Type B devices. Software means micro chip, micro chip means internal failures that we do not fully understand.



A Type A Emergency Stop Button and
a Type B Electronic Logic Solver

There is a big advantage that applies to Type B devices. Type B devices are basically based on (programmable) electronic technology. The advantage they have is called **diagnostics**. Because of built in diagnostics we can detect failures. As already discussed devices inside subsystems can fail dangerous or safe. When a device has diagnostics we can have **dangerous detected** (**DD**) and **dangerous undetected** (**DU**) failures and we can have **safe detected** (**SD**) and **safe undetected** (**SU**) failures. Each device in a safety loop has these four

failure modes. Although devices without diagnostics, basically Type A devices, only have dangerous undetected and safe undetected failures modes.

In reality we can reveal failures inside devices in two other ways besides diagnostics. Failures can also be revealed by monitoring abnormal **process behaviour** or by **proof testing**. Revealing failures by abnormal process behaviour is not very useful. Take this extreme example. If the factory explodes it means that something was not working, something had a failure. But we would like to know about this failure before the explosion happens. Bottom line is we do not depend on revealing failures in safety functions by looking at the process behaviour.

The third method is proof testing. Proof testing means that periodically we test part of or all parts of the safety function to see if it still works. Proof testing is very good, it has only one flaw. The frequency can be very low compared to diagnostics testing. Proof testing might take place only once per year. If you proof test today and tomorrow the failure happens you will not know about it until the next proof test. Diagnostics on the other hand are carried out with high frequency. We are talking about milliseconds, seconds, minutes or at the most a few hours. Diagnostics is great for electronic devices. Mechanical devices do not have diagnostics and thus we can only rely on proof testing. Proof testing can also be used on electronic devices to try to find those internal failures that could not be detected by diagnostics. But the actual proof test **coverage**, i.e., how many failures can be detected, is very low for electronic devices. In other words a proof test cannot really find a lot of failures in a type B device. You better rely on diagnostics in that case.

Realistic proof tests are always the best

We need to know about safe and dangerous failures and about detected and undetected failures because each device has a so called **safe failure fraction**. The safe failure fraction is expressed as a percentage. A high safe failure fraction is good. It tells us a lot about the device. It either means that from all the possible internal failures a lot of them are safe failures. This is good because it means that the safety function will be executed by a safe failure. On the other hand it can also mean that a lot of the internal failures are dangerous detected failures. This is also good because if diagnostics detects a dangerous failure, then we can do something with it. For example give an alarm and repair it.

The safe failure fraction is calculated as a ratio between failure rates:

$$\frac{\text{Safe Detected + Safe Undetected + Dangerous Detected}}{\text{Safe Detected + Safe Undetected + Dangerous Detected + Dangerous Undetected}}$$

The lower the dangerous undetected failure portion inside a device, the higher the safe failure fraction. Just remember one thing. The proof test is not allowed to be used to “detect” failures and thus to determine the safe failure fraction. Only built in diagnostics can be used for that.

In summary, the safe failure fraction is a measure of the fail safe design and/or the built in diagnostics of a device. The higher the safe failure fraction, the lower the portion of dangerous undetected failures in

the device. The lower the portion of dangerous undetected failures the more functional safety we can achieve with that device.

We need to understand all these hardware concepts because there is one hardware subsystem concept where everything just comes together. When we design a complete safety function then we cannot just design anything we want. The more functional safety we need to achieve the more hardware fault tolerance we need to build in. Hardware fault tolerance is a concept that applies on subsystem level. In the functional safety world the needed hardware fault tolerance is, in the end, determined by the SIL level of the safety function, the type (A or B) of the device and the safe failure fraction of the device used in the subsystem. The table below shows how the IEC 61508 standards defines the **architectural constraints** of a subsystem. From this table it becomes clear that we can achieve the required SIL with different architectures. Depending on the chosen architecture we need to select devices with the right safe failure fraction in order to achieve our target SIL.

A table that shows how the architecture is based on SIL, SFF, and Type

	Type A			Type B		
Safe Failure Fraction (SFF)	Hardware Fault Tolerance (HFT)			Hardware Fault Tolerance (HFT)		
	0	1	2	0	1	2
< 60 %	SIL 1	SIL 2	SIL 3	N.A.	SIL 1	SIL 2
60 % - < 90%	SIL 2	SIL 3	SIL 4	SIL 1	SIL 2	SIL 3
90 % - < 99%	SIL 3	SIL 4	SIL 4	SIL 2	SIL 3	SIL 4
> 99 %	SIL 3	SIL 4	SIL 4	SIL 3	SIL 4	SIL 4

Of all the functional safety hardware concepts that exist the architectural constraints is the most important one. The architectural constraints are more important then the PFD calculation. In ninety-nine percent of the cases if the architecture is correct, the PFD calculation will not be a show stopper. But if the attention is first to calculate the PFD, the chances are big that the architecture is not compliant. And that is a big problem. Especially when the safety function is already installed and

commissioned. Any modifications you need to make to the architecture at this point will be very costly.

While we design all this hardware we need to take into account one thing and that is **measures to control failures**. Hardware can fail randomly or due to a common environmental causes like a lightning strike or flooding. The more functional safety we want to achieve the more we need to control these kind of failures. We never know when they will happen, but we can make a design that can control them. A typical measure is, for example, redundancy, but there are many more that can and must be applied.

The Software

The Necessary Evil

Software behaves completely differently compared to hardware. Hardware can fail due to a random failure or a common environmental cause. Software on the other hand, does not fail random, or due to flooding or a lightning strike. Actually some experts even claim that software never fails. And they have a point. If software does not do what we want it to do then we programmed it in the wrong way. In other words, software failures are systematic failures and systematic failures are human problems. If software failed, then we made the wrong software. The software just did what we programmed it to do.

This means that if the software does not behave correctly during the operational phase then the software problem or mistake was caused a long time before the operational phase. If we want to deal with this issue then we need to implement measures that help us to avoid making mistakes. And if we did our job right then these **measures to avoid failures** were already planned during functional safety management and the creation of the safety plan.

What we want to achieve in the functional safety world is **safe software**. Safe software is software that is able to carry out the safety function even under fault conditions. It does not matter whether these fault conditions come from hardware or from the software itself. In theory, we do not care how buggy the safety software is as long as it can shutdown. Even if it means that it shuts down every five minutes. In practice we would like to have, of course, safe and bug free software.

In the functional safety world there are three distinct pieces of software. First of all there is the **embedded software**. This software makes the safety device run. It includes the operating system, libraries, diagnostics software, support functions, etc. The embedded software is the software that you get when you buy the safety device and take it out of the box.

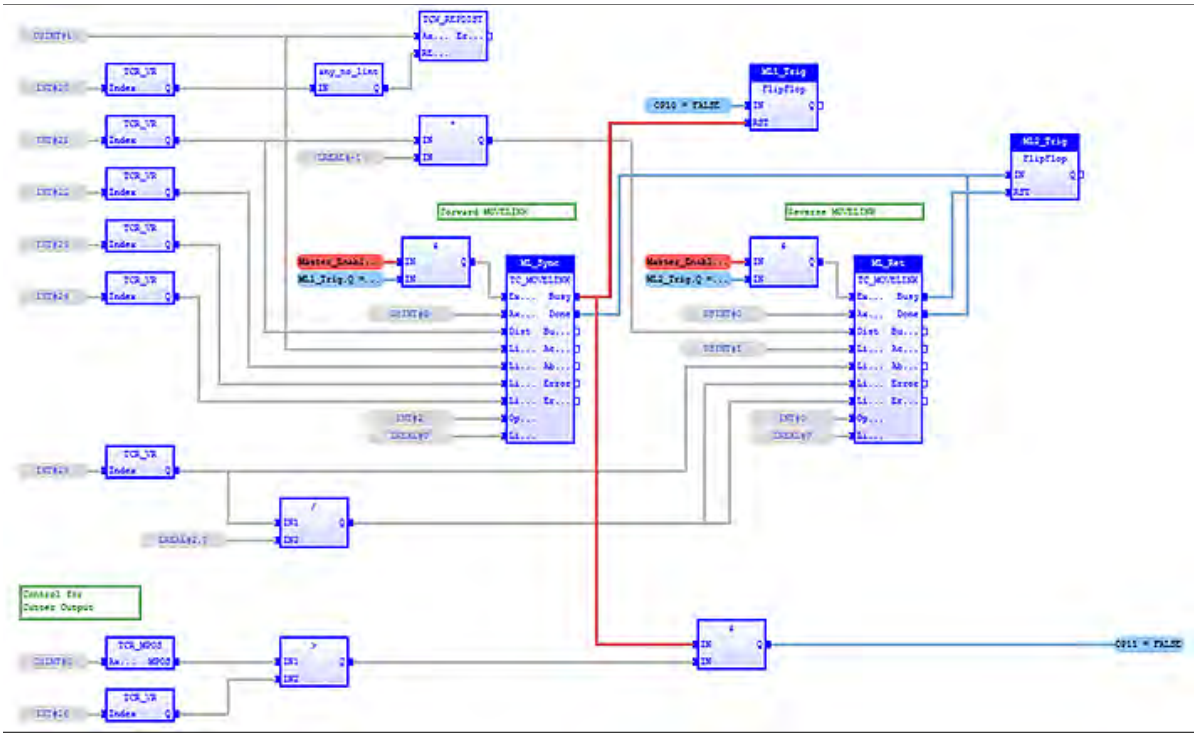
On top of the embedded software sits the **application software**. The application software is the software that takes care of the unique safety application. This is the logic and includes, for example, the set limits and voting blocks. This is the software the system integrator

usually programs for the end-user. This is the software that the safety function is all about. But the application software cannot do any good if the embedded software is not working as it should.

There is a third piece of software that we need to deal with, the so called **utility software**. This is the software in the engineering station or the hand held device. The utility software is the software that we need to actually program the application software. It might sound strange, but this software is not as important as the embedded or application software itself. The reason is very simple. In the field, there is only the embedded and application software and this is the only software that needs to be working. Whether the engineering station is working or not does not matter, as long as the embedded and application software is the software we want it to be.

These three distinct pieces of software are programmed in three types of software languages. And this is important as it will influence how much work your company will have when writing software. The first type is called **Full Variability Language** or **FVL**. You use FVL when you program C, C++, Pascal, etc. Basically it means that the programming language is so flexible that you can program anything you want. And if you can program anything you want you can also make a lot of mistakes. This is the most difficult type to deal with when it comes to functional safety and is the languages needed to program the embedded software. You need to have strict working processes and apply the measures to avoid failures in the best possible way, when working with FVL programming languages.

Many safety applications make use of general purpose safety PLC's. These general purpose safety PLCs run the embedded software and the application software. But the application software is often programmed in specific languages like Function Blocks, Structured Text or even Ladder Logic. When these types of languages are used, then we are dealing with **Limited Variability Languages** or **LVL**. With this type of language you can program voting blocks, times, alarm messages. You do not have the flexibility as with the FVL languages and thus you cannot make as many mistakes. But you can still make them. These are the languages that the system integrators and end-users deal with.



Example LVL language: Function Blocks

The last type of programming language is the **Fixed Programming Language** or **FPL**. Basically this means you have a device with software inside but you cannot really program it. You might be able to set some parameters like process related parameters. Maybe you can switch from psi to Barg, or from Celsius to Fahrenheit, or you can set the range of a certain value but that is about it. You cannot program logic with it. You cannot do much with this software but that does not mean you cannot make any more mistakes. One small setting mistake and you might compromise the complete safety function.

Each type of programming language (FVL, LVL, FPL) has its own approach towards achieving safe software. The complexity and amount of work involved in achieving safe software is the highest with FVL and the lowest with FPL software. All languages need the standard work, verification and assessment approach. The FVL languages needs very strict software engineering and quality assurance processes. The good news is real programmers, I mean not hackers, are at work when FVL are used.

The users of LVL and FPL are not real programmers. They are normally process application experts. With LVL you are still programming logic and thus you still need software engineering and

quality assurance processes, but not as strict and as detailed as with FVL. The FVL languages themselves are much more friendly and much more restricted in nature. The software development process is much easier including the testing phase.

The FPL languages usually do not require a software engineering process, but they do require quality assurance as performed by functional safety management. In practice, using FPL languages usually means using a hand held programming device to set the parameters of the safety related field device. The parameters should be clearly specified, the parameters should be set according to specification and somebody independent should verify that the chosen parameters are correct.

A hand held device to set the parameters of a field device (Ref. Endress+Hauser)

Certification, Proven in Use & Data

It is supposed to make your life easier

It is the end-user of the safety function who is ultimately responsible for achieving the right safety function that works as it should. But the end-users are usually not experts in the hardware and the software of the safety function. They should not be, they should focus on drilling oil, their chemical process, the roller coaster or the nuclear power plant that they try to operate. Their expertise is with the process that needs to be protected. As it should be.

But how can they assure that the safety function they are using is actually functionally safe? The answer is that one can make use of certified devices or proven in use devices. And both options need reliability data.

Certification can help because the end-user trusts that a third party has verified that everything is done according to the rules of functional safety. In other words the end-user does not need to check it himself. Somebody independent has done the work for them. Today, anything that is related to functional safety can be certified. At Risknowlogy we certify safety related products and complete systems. We also certify the functional safety procedures of organisations as well as risk, reliability, and safety related studies (like FMEA, HAZOP, LOPA, etc). And we even train professionals and certify their competence (TUV Functional Safety Professional, HAZOP Leaders, etc).

Buying all these certified devices and services and hiring certified professionals does not take away the responsibility of the end-user. He still needs to make sure that everything fits together and achieves functional safety over the full life of the safety function. Think of it this way if you are an end-user. Who will be the first one standing with their name in the news paper when an accident happens? You or your supplier? Exactly.

RISKNOWLOGY®

CERTIFICATE

Risknowlogy Certified Professional

Holder:	John Doe
Student ID:	1234
Basis of training:	IEC 61511:2003
Course:	SIL Verification and Calculation
Industry:	Process Industry
Field:	Safety Instrumented Systems
Eligibility requirements:	☑ Practical experience ☑ Completed training in full ☑ Passed examination

Results:

Risknowlogy certifies that the above certificate holder has completed the Risknowlogy personal development program for the above mentioned course and has successfully met the eligibility requirements.

Date: 2014-05-22

dr.ir. Michel Houtermans
Certifier

Ricardo Vittoni
Trainer

This could be your personal certificate

Sometimes you have been using certain devices for many years. Apparently these devices are performing well, you would have replaced them a long time ago. There is no need in the functional safety industry to replace an older well performing device, you can use them as they are **proven in use**. Proven in use is in some cases a better option than certified devices. Especially when it comes to mechanical devices. And guess what, proven in use can be certified.

Many times it happens that third parties are certifying products but do not have **reliability data** available. The same counts for proven in use studies. Sometimes safety devices are used that you know are working well, because they never failed, but you do not have data for them. Twenty years ago, this would have been a problem. But today there are so many sources of data that you can always get some kind of data that more or less supports your device.

Most likely the data source you selected does not represent exactly your situation 100%. But do not forget, the PFD calculation is only a prediction of what we think might happen in the future. It is not supposed to be a perfect calculation. We predict and then we monitor the real situation in the field. If our prediction was too negative, well no problem, it means the real safety behaviour is better. If our prediction was too positive then we notice the problem, because we are monitoring it, and we adjust accordingly. We add more redundancy or we buy devices with better reliability parameters. And that should solve the problem again.

In other words, when you are working with hardware or with software, you cannot always analyse it yourself. But when you make use of certification or proven in use, you can trust that the devices you implemented are functionally safe and thus your safety function becomes functionally safe. This is the best alternative, instead of analysing everything yourself.

Operation, Maintenance and Repair

Last but certainly not the least in the SIL lifecycle

Once the safety system is designed, and the correct hardware and software have been selected and built, it is time to **install and commission** the system. When the system is commissioned and working as it should, we still need to do one final check before we start using this safety system. We need to **validate** the safety system, which means we need to check that what we ordered is actually what we got.

If the perfect SIL 2 system is installed and commissioned, then validation might find out that actually it should have been a SIL 3 system. In other words, we ordered a SIL 3, they designed, manufactured and installed the perfect SIL 2 and thus we ended up with the perfect wrong safety system. These kind of issues can happen when, for example, a mistake is made during the design. The specification said SIL 3, but somewhere during the design process we ended up with the requirements for SIL 2. Hopefully, in practice, a significant mistake like this is not , but you have to validate it to make sure.

Once the validation is done it is time to use the safety system. We need to **operate, maintain** and **repair** it correctly. During the planning phase of the safety system we already know what kind of safety system we would get. So we are able to work already on the procedures to operate, maintain and repair the safety system. Now it is just time to apply those procedures correctly.

The operational phase is a very long phase though. It could take ten, twenty, thirty, maybe fifty years before the phase is over. We need to check that during this time everybody is working according to plan. But in practice a functional safety assessment is done at the end of a lifecycle phase. It would not be very useful to wait thirty years and then do an assessment to find out if everybody worked according to plan for the last thirty years. Instead we start to do functional safety audits. A **functional safety audit** is the same as an assessment only periodic. In this way we make sure that the safety system remains functionally safe during the operation phase.

If a demand is placed on the safety system, we record the time and date of the demand. We record whether everything worked as planned, or not. And if it didn't, we take corrective action.

Sometimes the safety system fails. We repair it and document the reason for failure. If it was a random or common cause failure, then we collect data so that we can make better probability calculations in the future. If it was a systematic problem then we better buy a different product, or we fix a procedure or manual or guideline, so it does not happen again in the future. No matter what happens, we document it and we feed it back into the system, so that we get better and safer over time.

During the operational phase it is also possible that we have to make modifications. We can make as many modifications as we want, as long as we follow the modification procedures we defined during functional safety management and assure ourselves that functional safety is at all times guaranteed. The best, and easiest, time to make modifications is when the process, which the safety system is protecting, is not running.

Sometimes we need to make modification on a live, running, process. Also this is allowed, even if you have to bypass the safety function for a period of time. You can make as many bypasses as necessary as long as you have them under control and do not jeopardise functional safety. Make sure you have a bypass procedure. Limit the duration of a bypass, inform everybody that you made the bypass. Place big signs informing everybody and see if you have alternative means to bring the process to a safe state, if necessary. Make sure you have bypasses under control, at all times.

Some Final Words

And that is it. Basically everything has been described that needs to be thought of when applying functional safety. A full chapter could have been developed for each topic mentioned here with all their ins and outs. However, many of the actual details depend on the features of the project at hand and the industry you are in. Once you have functional safety thinking though, you can apply functional safety to any project, with or without the use of standards.

If you think about it, functional safety is nothing more than good engineering practice. And good safety engineering practice can be implemented if you create a culture of safety in your organisation. If you think functional safety is just another property of your safety device or process, then you will never excel in functional safety and you will stay a mediocre player in this field. I think mediocre is not an option when it comes to safety. What do you think?

Share If You Liked It

About The Author

Dr. Michel Houtermans decided to study Mechanical Engineering at Eindhoven University of Technology because his big brother was studying it as well. The first three years he was a disaster in class because he was very good at taking things apart, but not in putting them back together: a true mechanical engineer. A lightning strike hit and he realised that disasters were bad for business. He joined the reliability department of Professor Brombacher for his final assignment and graduated Cum Laude in 1995. He found his niche and was happy.

Under the motto “no risk, no glory” he moved to Boston (U.S.A.) to join Dr. Dimitrios Karydas of the Risk Engineering department at FM Global as a research and project engineer. While working at FM Global he had the opportunity to cooperate and study with Professor George Apostolakis at the Nuclear Engineering department of the Massachusetts Institute of Technology (MIT). Although he took lots of risks, he was in excellent hands and got glory when he received his PhD in Safety and Risk Management in 2001.

In the mean time, he had joined in 1998 the German certification body TUV SUD in the USA. No better place to learn functional safety than from the people that invented it. As, TUV SUD department manager responsible for functional safety for North and South America, he invented, created and marketed in 1999 the first functional safety people certification program. Today it can easily be said that this program changed the functional safety world forever. Worldwide over ten thousand people have been trained of which many received TUV certification.

Since his first job at FM Global, Dr. Houtermans had the opportunity to participate and contribute to national and international safety committees like ISA S84, IEC 61508, IEC 61511 and others. At the age of twenty five he was the youngest committee member and one of the most knowledgeable ones in reliability and safety. He wrote part four of the ISA TR84 technical report (Determining SIL of a SIF via Markov Modelling). Unfortunately, he learned that unless you had grey hair it was very tough to be accepted as a safety professional. But that changed in his favour when he started to lose his hair and got his Ph.D. He is not sure which one of the two contributed more though.

On 2-2-2002 Dr. Houtermans founded the Risknowlogy company and turned his personal passion, risk, reliability and safety, into the only objective of the company. Today, Risknowlogy is a global operating company servicing end-users, engineering companies, system integrators, product suppliers, third parties and governments.

Dr. Houtermans is married and has two children. His wife had a good, risk free job but decided to join Risknowlogy anyway. She is an excellent risk manager. His oldest daughter shows no signs of engineering side effects, but is an expert arguer. If she does not decide to become a politician, he hopes she will join Risknowlogy as the company lawyer. His youngest daughter is still fearless, knows no risk, and takes everything apart. We all know what that means. To be updated in time.

About The Best Practice

The goal of the Risknowlogy Best Practice publications is to be the leading destination when it comes to risk, reliability and safety knowledge. It aims to provide professionals around the world with rigorous insights and best practices to help them to become leading risk, reliability and safety practitioners for the benefit of all.

Get involved - Become an author

Thanks for considering working with us. We believe that if companies and organisations would understand their hazards and risks better, if they would understand how to build reliable solutions to manage those risks and if they would know how to achieve safety in an effective manner, then everybody - the employees, the bosses, the customers, the people, the environment, the investors - and the whole world would be better off. So we try to arm our readers with knowledge and ideas that help them to identify and manage risks better, to design sufficiently reliable solutions and to be more safe at work. To do that we enlist the foremost experts in risk, reliability and safety theory and practice, collaborating to express their best thoughts in the most influential ways possible.

If you have a new piece of research, a new approach, an unexpected perspective on a current event or an original way of looking at a perennial risk, reliability or safety problem in any industry, we would love to hear about it. Here's what we look for, when we're considering what to publish at Risknowlogy:

1. Expertise — You don't have to have grey hair, or no hair at all for that matter, or to be well known to be a contributor to the Risknowlogy Best Practice, but you must know a lot about the subject you're writing about.

2. Evidence — It's not enough to know your subject deeply; you have to prove it to the reader. You can refer to research or practical work of others. You can also demonstrate your thoughts with practical and relevant examples. Use data, create charts, create graphs, do anything creative to share your ideas. Our audience is keen to see your thoughts in action.

3. Be Original — Risk, reliability and safety problems are not new. The world always had them and always tried to solve them. There aren't that many wholly new ideas in risk, reliability and safety and the problems practitioners face have probably been solved already somehow. So if you're writing about a well-worn topic, you need to find a fresh and creative approach. The best way to do this is to be very specific and to rely on your own research, observations, and experience. The worst way to do this, generally, is to give the same solution a new jacket or use a fancy, clever new phrase.

4. Usefulness — Whatever you submit it needs to be useful to our readers. Our readers don't read our Best Practices just to stay on top of new developments in risk, reliability and safety thinking. Much more, they are looking for help in changing the way they and their organisations actually approach risk, reliability and safety topics. Try to explain your thinking so that the reader understands how to begin to apply it in a real situation. Making your ideas useful will make it a lot more powerful.

5. Be Persuasive - Make your publication a pleasure to read. The Risknowlogy Best Practice readers are smart and skeptical and busy like everybody else in the world. If you don't capture, and keep, their attention, they will not hesitate to move on to something else. Use compelling language, get straight to the point in the first paragraph, avoid jargon, and spend the extra time necessary to make your language sharp and compelling throughout. Use pictures, graphs, use humour, anything is allowed to keep their attention.

Some notes about our editorial process

We are looking for quality, not quantity. We only publish if we feel it contributes in the right way. The best thing to do is to send us a short pitch first so we can give you feedback early enough. Nothing will be published unless we have seen a full draft. Most likely our editors will ask you to revise parts of your publication. We will have it read by more than one editor, to make sure we get the most out of your contribution. Send your pitch or full draft to your editor or if you do not have an editor send it to bestpractice@risknowlogy.com.

Just remember our editorial process is more thorough than many other publishers'. We will work with you to make your contribution the

best for our readers. Contributors tell us frequently that they really appreciate the extra care and attention their work receives. We also retain final decision rights over headlines. Our editors have spent years learning what kind of headlines give Risknowlogy Best Practice pieces the best chance of being read, found on the web and shared both on social media and in offices around the world. We will very likely rewrite the title you suggest; if we do so, it's because we believe the revised version will help your publication reach the audience it deserves.

We want you to write your publication yourself, in your own voice coming from your heart. Please don't submit something written by your PR representative or a ghostwriter or something that was published already elsewhere. We don't publish pieces that have appeared elsewhere or that come across as promotional.

bestpractice@risknowlogy.com

About Risknowlogy

Experts in Risk, Reliability and Safety

Risknowlogy was founded in 2002 with a passion for risk, reliability and safety. We are particularly known for our leading role in functional safety. At Risknowlogy we apply all typical risk, reliability and safety techniques you might heard of: Bowtie, HAZOP, HAZID, LOPA, AHA, OHA, PHA, QRA, FMEA, FMECA, ETA, Markov, FTA, Reliability Block Diagrams, FMEDA, FSM, SIL Assessment, SIL Verification, SIL Certification, Calibration Risk Matrix and Risk Graph, STL and so on. But our services go beyond the application of the classic and standard techniques. Contact Risknowlogy if you are in need of risk, reliability or safety services.

At Risknowlogy we apply risk, reliability and safety techniques so that our customers become more profitable. Our services help companies to be compliant with standards and thus to meet regulatory requirements, to meet the requests of insurance companies, to meet and exceed their availability goals. And more availability leads to more profitability.

Not so typical for industry, but for us bread and butter, services we have carried out for our customers:

- Setup a risk, reliability, and safety competence program for the technical employees of a chemical plant
- Implement a risk management program including a functional safety handbook for an oil refinery
- Calculate the availability of gas supply (needed for electricity, cooling) for a major city in the middle east
- Decision support for the implementation of safety functions including their proof test frequency for a petrochemical plant
- Pipeline risk management program for country's pipeline operator
- Governmental functional safety audit program for tunnel operators
- Decision support for the best out of five infrastructure solutions taking into account image, environmental, cost, legal, sustainability aspects for a local government.

Contact Risknowlogy if you are in need of a customised risk, reliability or safety solution.

Certification - Increase The Trust

If you do not trust it, certify it. Risknowlogy has developed a Certification Program that is unique in the world, as we are the only company in the world that certifies risk, reliability and safety parameters of products, functions, systems, organisations and professionals.

The Risknowlogy Certification Program has been developed to support end-users. End-users are in need of competent personnel and contractors. End-users need the right procedures and need to make sure that their employees and contractors follow and implement these procedures. End-users need to trust the products and systems they buy and use to operate their processes, factories and plants. End-users can take full advantage of the Risknowlogy Certification Program.

Typical certification projects we have carried out for companies and people:

- Risknowlogy certified over 3000 people in the TUV SUD and Siemens certification program
- Risknowlogy SIL certified products like level transmitters, temperature transmitters, pressure transmitters, actuators, valves, relays, sensors, pumps
- Risknowlogy certified Safety Availability in terms of SIL for functions and systems like ESD, HIPPS, BMS, Smoke and Detection, Overfill Prevention Systems
- Risknowlogy certified Process Availability taking into account malfunction of Safety Instrumented Systems
- Risknowlogy certified Functional Safety Management systems for System Integrators according to IEC 61508 and IEC 61511

Contact Risknowlogy if you are in need of certification of people, products, functions, solutions, organisations or a unique in-house certification program customised and tailored to your company's unique needs and requirements.

History

Risknowlogy was founded in 2002 and is an employee owned business. Today we have offices in Argentina, Colombia, Germany, India, the Netherlands, Switzerland (HQ), the United Arab Emirates, the United Kingdom and Uruguay. We offer our services in Dutch, English, French, German, Italian, and Spanish.

- 2002 - Risknowlogy was founded in Schinveld, The Netherlands
- 2007 - Risknowlogy moved headquarters to Zug in Switzerland
- 2008 - Risknowlogy opened the Buenos Aires office in Argentina
- 2009 - Risknowlogy opened the Karlsruhe office in Germany
- 2011 - Risknowlogy opened the Dubai office in the United Arab Emirates
- 2012 - Risknowlogy opened the Mumbai office in India and the Bogota office in Colombia
- 2013 - Risknowlogy opened the UK office in the South of England
- 2014 - Risknowlogy opened the Montevideo office in Uruguay

Contact Risknowlogy if you would like to explore opportunities.

RISKNOWLOGY®

Experts in Risk, Reliability and Safety
Baarerstrasse 11, 6300, Zug, Switzerland

ISBN 978-3-9524357-0-0