GUIDE
to software Usability principles

INDUSTRIAL AUTOMATION SOFTWARE
progea

Usability

The software usability (of any special purpose man-made object) means ease with which a person is able to perform their work coherent to the software functions.
This definition is exceedingly simple, because the easy/difficult concept is very subjective and does not take into account factors such as knowledge, skills, intuition and external influences that go into designing, creating and deploying the software in question.

In fact, those who use software and find it hard to use, therefore unusable, often do not have the know-how or are unaware of the software's restrictions of use. The program may not be capable of working in the way in which the user intended it for.

In simple terms, if a software results unusable it may either be the fault of the designer who developed a program unsuitable to the set task, or fault of the user who does not know the full extent to which the software has been designed for. Even though a very controversial argument, there are mixed feelings on what characteristics and functions software should have in order to reach a reasonable usability level. It would be however, best to consider the software typologies (which we will do below for the HMI interface) and apply these with suitable rules, methodologies and guidelines.

It is also important to distinguish between usability and utility aimed at satisfying the immediate needs and concepts, and cultural products and objects whose primary aim is to give pleasure, satisfaction and emotion, not to resolve the task at hand.

Below are some usability definitions from well known experts:

B. Shackel

Claims that "The usability of an artifact consists of its capacity, in terms of the human factor, ease and effectiveness of use by specific category of users, with specific practice and user support, to carry out specific categories of tasks, within specific environmental scenarios."

K. Kunkel, M. Bannert and W. Fach

Sustain that the usability context in general embraces the 4 principal components of a work situation:
user, task, system and ambient. Design engineering focused on usability means being able to harmonize the interaction of all these 4 factors.

A. Preece

Claims that the objective of Human Computer Interaction (HCI) is to develop and improve systems that provide the use of computer for users to perform their tasks without any problems, effectively, efficiently, fully exploiting their work tools.
These 4 factors together compile the usability concept

B. Shneiderman

There are 4 principal dimensions identified in the usability concept: efficiency, learnability, memorability and satisfaction.

G. Morris and A. Dillon

Morris and Dillon reflected on B. Shackel's definition, party agree on and partly on the human factor and working environment of product's destination, considering the rest as optional.

Jakob Nielsen

One of the authors used as a reference point for web usability, defines usability as a quality measure of user interaction experience with something, whether it be a website, conventional software application or any other instrument that can be used by a user.
According to Nielson, a product is usable when it can be easily learnt, efficient to use, easy to remember, easy to learn, interaction error tolerant.

Software Interface

Software is a product with particular characteristics.
To start with, it is not an object of beauty to astound and mesmerize. Its duty is not to entertain but to solve problems of a certain type. It cannot just materialize in your hands on its own but needs the help of another component, the computer, in order to do so and accomplish the assigned task.

Gone are the days when software designers/developers were also the end users. Now a consolidated separation has emerged between those who design and those who use it. This separation has left a great distance between the two due to evolved from product usability problems.

According to R. Norman software product usability measures the cognitive distance:

- The project planner's model, being the product model and its features which user is able to use (based on past knowledge) and applied in product, and
- The user model, being the product's functional model built by the user (also based on user knowledge and ability) and who regulates product interaction.

Communication takes place between:

- The software product, the interface (present) and the project planner's model (absent), and
- The user (present).

Movicon 11 is a Scada/HMI platform purposely designed for creating HMI and Scada Supervision graphics with great ease and intuition. All these tools, whether graphics or functions , are ready at a hand's distance for the project engineers to use in creating any user interface project efficiently and effectively.

Project engineering becomes a pleasure using Movicon™11 to obtain immediate visual impact in very little time, safeguarding your investment.

Human-Interface Communication

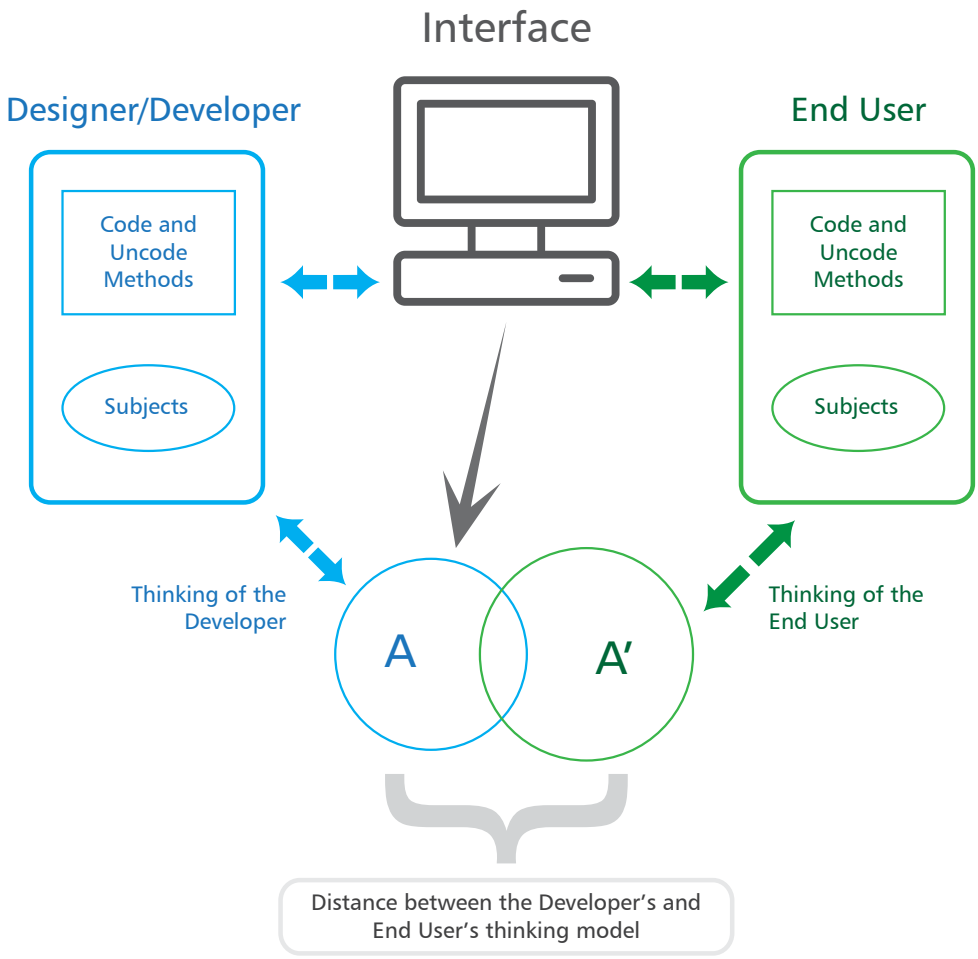
The human-interface communication is similar to the person-to-person communication model. The interface can be envisaged as a message sent by the project engineer-developer to the user. Both have their own mental (perception) model, which is a selective representation of reality, evolving with experience, contact and interaction with their surroundings, things and people. They also have a conceptual model of what the interface must do. It may be how the user conceives and understands the interface and how the project engineer conceives and incorporates into the interface.

As you can see by studying the above diagram, communication takes place between the user and interface (both always present). The engineer has set specific actions in order to get certain results. The user also thinks that these specific actions will get certain results but these results are not the same as originally thought by the user when exchanging messages with the interface, but those of the engineer. A mismatch has occurred.

In fact, if you indicate with A the significance of actions provided and incorporated into the interface by the engineer (according to their own mental and conceptual model) and indicate with A' the significance of the same actions implemented by the user within the same interface (according to their mental and conceptual model), then if:
A and A' match: the interface will respond adequately, user and engineer are on the same wave length,
A and A' do not match: the user's actions have not been foreseen for by the engineer and the interface responds inadequately, user and engineer are not on the same wave length.
The second example shows that the user needs to learn how the interface really works and its constraints.
The main objective of usability is to make technology disappear into the background until it becomes transparent enough for the user to concentrate only on his work without any interface distraction.

Conclusively in order to be usable a software product must:

- be suitable for the needs and expectations of end user requirements according to specific conditions;
- be easy to understand, learn, use and gratifying to use;
- allow activity specifications to be performed correctly, quickly with satisfaction.



The ten Nilsen heuristics

The 10 Nielsen heuristics on interface usability, derive from applying factor analysis techniques to 249 usability problems.

1. Visibility of system status

The system should always keep users updated on what is going on, by supplying appropriate feedback within reasonable time.
To know if an object is a link and where does it lead to
Icon or text under intensification means that function is not available Signaling of ongoing activity (hourglass, text messages, progress bar etc.)

2. Match between system and the real world

The system should speak the user's language, with words, phrases and concepts familiar with the user rather than system-oriented terms. Use of text messages, icons and actions should be concise and commonly known to the layman as user ("save with name", "recycle bin" icon, "copy and paste" action).
Guarantee associations between objects and information.

3. User control and freedom

Users should have full control of all information contents and be able to move about among the various arguments freely.
Avoid long intensive and compelling procedures (registering)
Avoid predefined paths without shortcuts. Avoid unwanted actions by user (automatic opening of unwanted page)

4. Consistency and standards

Users should expect that system follows interface conventions and not leave user wondering whether different words, situations or actions mean the same thing or not.
Each page should show some recognizable element (logo, graphical style, etc.). Keep same environmental feeling.

5. Error prevention

Avoid subjecting user to ambiguous situations, criticism that may cause error. Give user possibility to return back.
Prevent error occurrences with careful design.

6. Recognition rather than recall

Instructions for system use must be clearly visible and easily retrievable. Design simple and schematic layouts.
The user cannot be expected to remember the position of objects that characterize each page.
Make sure that user is not made to rediscover the interface each time they use it.

7. Flexibility and efficiency of use

Provide user with alternative interface uses (to cater for both for the non expert and expert user).
Provide hierarchical navigation for the less expert user.
Provide accelerators for the more expect user.

8. Aesthetic and minimalist design

Give more importance to content rather than appearance.
Do not put too much emphasis on irrelevant or rarely needed objects (over elaborated images, etc.).
Avoid putting information content in page background.
Avoid causing user distraction or confusion.

9. Help users

Help users recognize, diagnose and recover from errors.
Error messages should be expressed in plain language (without code).
Error messages should indicate the problem precisely and suggest an appropriate solution.
User confirmation request for important actions.

10. Documentation

Even though the system should be usable without documentation, it should however be made available for confrontation and help.
It should be easily retrievable.
Focus on user's task. Structured on a list of concrete steps that is not too long to carry out.

Usability Evaluation

Heuristics

Investigation on usability requirements by experts based on experience rather than theory.

Walkthrough

inspection method. A group of experts, engineers, technicians and users express their own evaluations on the effects of the interface on the end user. It is important that product results easy to use.

Proactive field study

Involves observing users 'thinking aloud' while performing tasks using the interface. This can be done in a laboratory or in user's working environment.

Interaction between user and evaluator is essential.

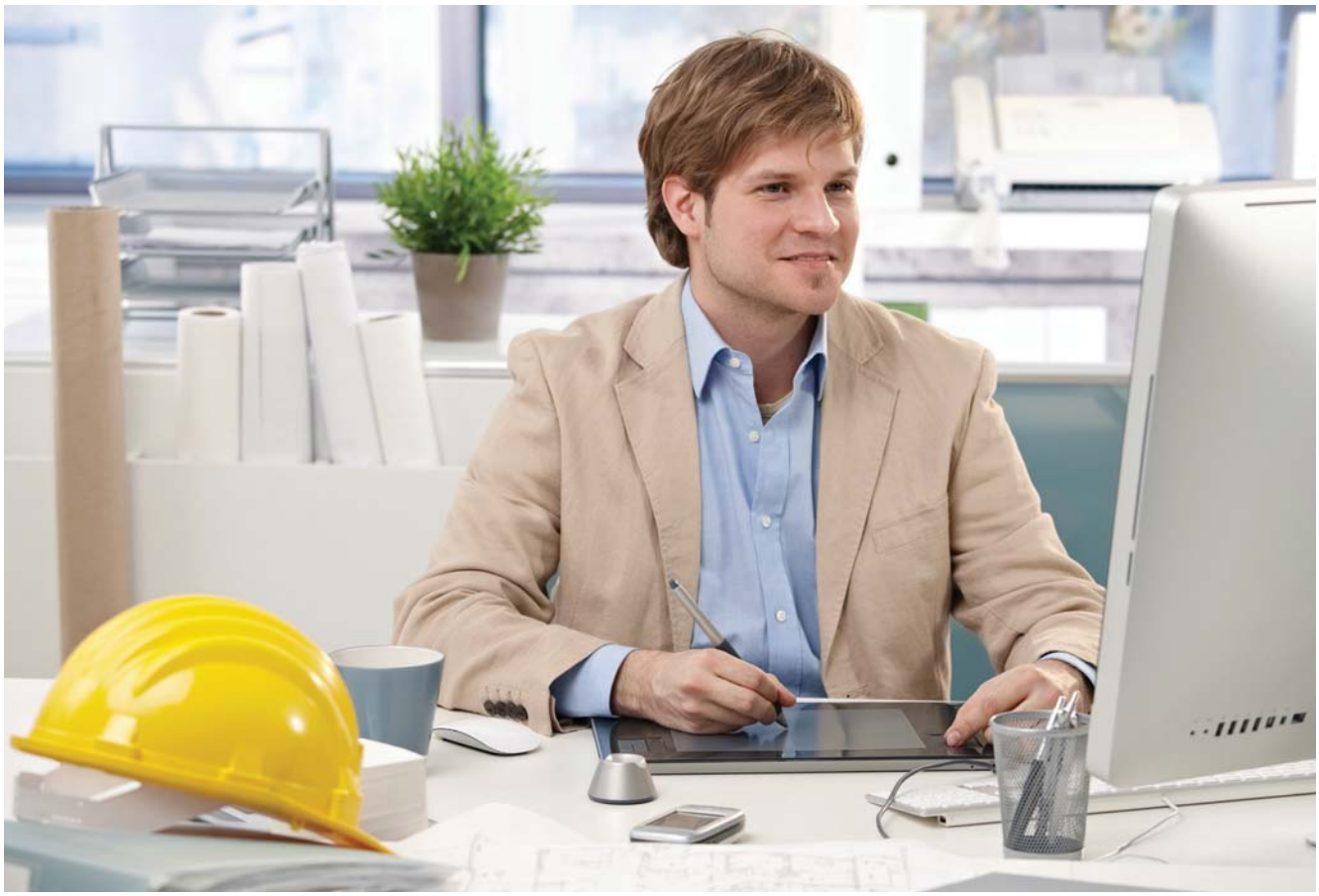
Usability testing

Observing user/interface interaction while user performs a set task with follow on behaviour analysis.
Quantitative measures are possible.

Questionnaires

Evaluation tools can be administered in the form of questionnaires and given to users from different backgrounds.
This implies assessing the user's interface knowledge through a "hands on" technique where each representative user is given the same set of tasks to work on with a questionnaire to fill in afterwards.
The evaluators can use the results from these questionnaires to see if the interface supports the users to do their tasks.

ISO Norms



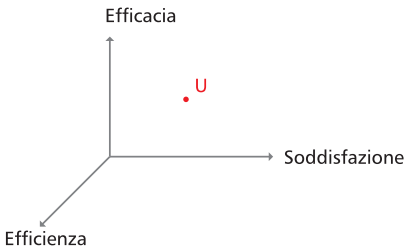
ISO 9241

Software usability is also regulated by the ISO normative, particularly the ISO 9241 standard for ergonomics of human-system interaction, defining the ergonomics requirements for offices using computer terminals. It is composed in 17 parts, each one deals with specific aspects that influence working at computer terminals. Parts 10 and 11 have particular relevance to this discourse.

Parte 10

Part 10 defines the general ergonomic principles to apply when designing human-computer dialogues which should result as:

- suitable for the task;
- self descriptive;
- user controlable;
- conformable with user expectations;
- error tolerable;
- suitable for individualisation;
- suitable for learning.



Parte 11

Part 11 defines usability being the extent to which the product can be used by specified users to achieve specified goals with

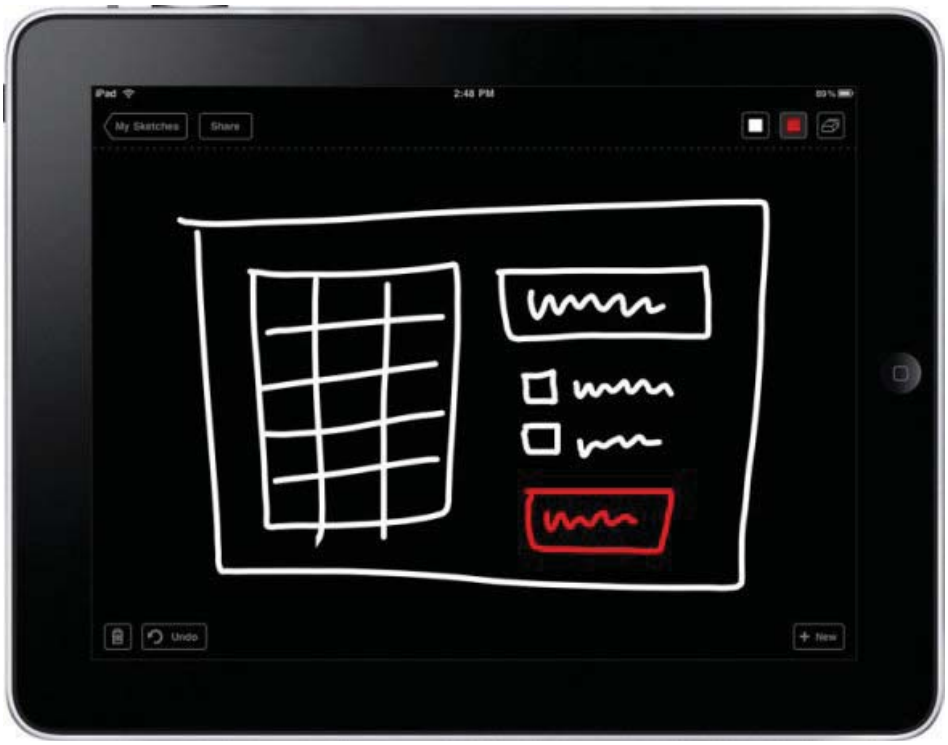
- efficiency, being the accurateness and completeness of results achieved;
- effectiveness, being the resources used to reach goals;
- satisfaction, being user acceptance and ability to work in comfort with the system in a specified context of use.

This last definition (with the repetitive use of the adjective "specific") specifies the concept of software usability.
In fact, those who design HMI software interfaces (and websites), should always ascertain:

- who to refer to and which type of user is going to use it
- why and what aim is it needed for
- where and in which situations is it going to be used.

These three parameters can be measured (at least empirically) to indicate which are the main ingredients needed in obtaining a certain degree of usability, keeping the user's particular requirements, task priorities and context in mind.

The distance of U from the theoretical system's origin represents the attributed system usability value.



Emerging HMI usability requirements

"A HMI interface is usable when it satisfies the information needs of the end user who is using and interrogating it, by being provided with facilitated access and browsing with contents that are easily understood. When there is a lack of information, process management becomes inefficient."

Navigability:

existence of a navigation and orientation system through the system's graphical pages. The user should not feel disorientated due to ambiguous or non homogenous control buttons.
The user must know where they are and how to return back to where they were last without any discrepancy.

User expectations:

information and/or services must match user expectations.
User expectations should be fulfilled for time spent visiting websites. Make sure website lives up to user expectations.
User expectations relating to speed and organization availability of data required for correct process management.

Concise and thorough contents:

information should be transparent and concise in every detail for easy user comprehension. It is not easy for a website to satisfy the information needs of every user in one single page content. The user needs to be guided with general information and links to more indepth information, which should be made visible only if there is a specific interest.
It is important that content amount and detail quality be suitable to argument type and divided into different groups under specific headings appropriately, thoroughly with clarity.
Often contents are regrouped within other groups within the same project, a particular usually requested by the intended audience.

Information Comprehensibility:

The format and quality with which the information and contents are presented within the system's graphical screen pages.
The type of language used is one of the most important aspects, especially for interactive operations. There must be some kind of general information classification which is comprehensible to all users, even though the final content may be specialistic.

Effective Dialogue:

Project communication strategy. Effective dialogue is a quantitative measure of interface system credibility and is based both on the institution/structure brand it presents, and on the capability to be transparent and thorough to obtain user confidence and trust.

Graphics Attractability:

The quality of the graphics and their visual attraction on the website. Graphics quality and pleasing project visuality.
Graphics must evoke the right balance between emotion and comfort to capture the user's interest and recognition of content use. It must not disguise the true scope of the system.
Data representations must be managed with harmony, order, moderately and with balance, using comprehensible graphical designs, adopting the most standard symbology or those most simple and comprehensible, using coherent, homogenous and moderate fonts and colors where needed according to standards.

Data Velocity:

System must be fully capable of updating data in "real time". This means quickly and adequately to the process managed by the system. Users would consider 0 to 3 seconds an acceptable time. This also goes for accessing requested pages. Quicker times would also be acceptable if not causing difficulty for user reactivity, diminishing user comfort and ease in managing the process. Longer times may be acceptable (between 2 and 4 seconds) for displaying large quantities of data on the premise that user acknowledges this. Nevertheless, longer times often cause difficulty for user in managing reactivity plus reduce user comfort in process management.

Balance

Before going into the specifics a fundamental and necessary characteristic in creating user interfaces should be understood and clearly defined.

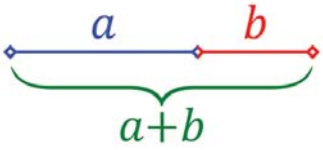
Proportions

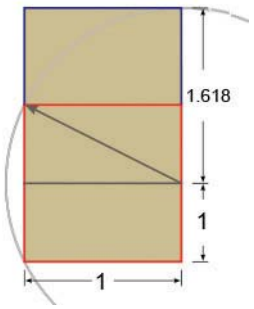
There is a structure at the base of every screen layout called the cage. This used for organizing the contents of any layout so that page after page, rubric after rubric, the various contents can be managed to keep a unified image, recognizable and organized according to the best reading keys possible. How do you find out which is the right position for data on sceen? By means of using the Golden section, that represents the
standard reference for perfection, grace and harmony whether for architecture, sculpture or paintings or within Nature itself.

Golden Section

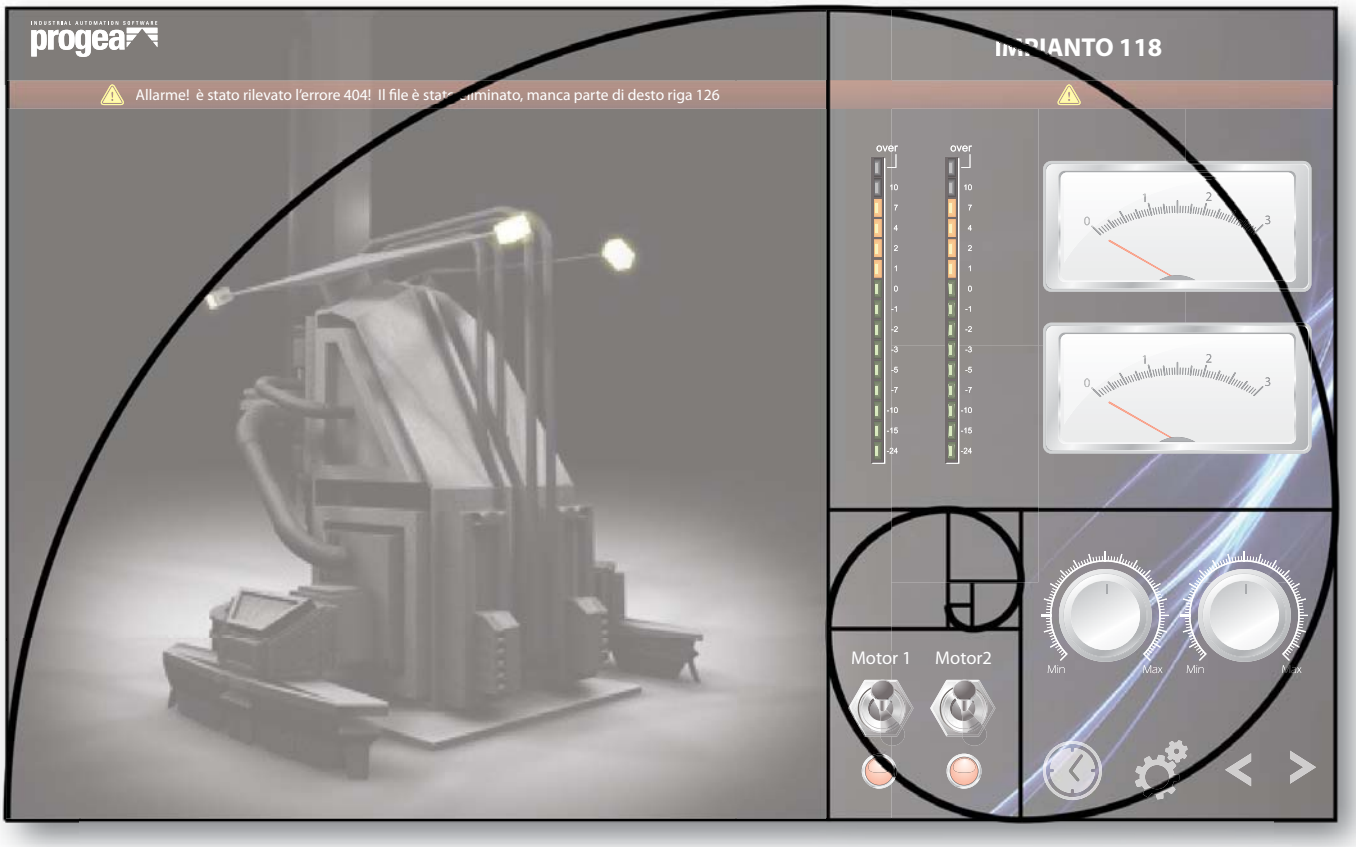
The Golden section has a three million year history.
The Golden section, in mathematics and in art, is a geometric proportion based on a specific ratio:

The larger quantity is to the smaller quantity as the total quantity is to the larger quantity





The example below shows a geometric proportion using the "Golden Spiral" as a hypothesis to define spaces and proportions within a graphical interface. The "Golden Spiral" is one of the possible shapes of the Golden Section applied with Geometry:
If a square is drawn in a golden rectangle with one side equal to the smaller side of the rectangle, the remaining difference also becomes a golden rectangle. This operation is repeated at least five times more until reaching an adequate visual effect.
Then place a compass point on the bottom left corner of the biggest square along the longest side of the rectangle and draw an arc to unite the two extreme corners of the square.
This operation is repeated for each square continuing from the point where the last arc left off until a spiral effect is created.



Human-Computer Compatibility

Interface usability (software product) does not solely depend on the computer's scientific-technical preparation but also on compatibilities involving social sciences and human behaviour. In fact human-computer cognitive compatibility (and therefore human-interface) should respect these three rules:

Compatibility

An interface should be physically compatible with morphology and human perceptions and actions; by means of using the three most involved senses (sight, touch, hearing), through use of character sizes, colours, text scrolling, command response time, as well as workstation space and environment.

Communication

An interface must also be compatible with those features concerning human communication, instincts and memory in solving problems; the interface should have an interpersonal, human-like way of dialoging, requiring interaction and redundancy, with short and long term memory capabilities, and finally human and automatic procedures in solving problems using different strategies where, in this case, trial and error is used by most.

Context

An interface must react accurately and in the right context; this clarification allows certain areas of application usability to be narrowed down and delimited.

Human-interface Communication

The human-interface communication is similar to the person-to-person communication model. The interface can be envisaged as a message sent by the project engineer-developer to the user. Both have their own mental (perception) model, which is a selective representation of reality, evolving with experience, contact and interaction with their surroundings, things and people.
They also have a conceptual model of what the interface must do. It may be the way in which the user conceives and understands the interface and the way in which the project engineer conceives and incorporates into the interface.

The main objective of usability is to make technology disappear into the background until it becomes transparent enough for the user to concentrate only on his work without any interface distraction.

Software interface usability principles

Based on a study of human factors and ergonomics in a user-centered approach to software product interaction, the usability principles underlie important issues and guidelines to be referred to when designing and evaluating product usability (Jakob Nielson's 'heuristic principles' a well known evaluation method, hence heuristic evaluation).
The principles explained in the rest of this document are to be considered a discrete written summary of a vastly dispersed argument:

- create a simple and natural dialogue
- simplify task structures
- facilitate recognition as opposed to user memory
- provide adequate feedback rendering system status clearly visible
- prevent interaction errors and facilitate recovery
- be consistent
- speak the user's language
- facilitate flexibility and efficiency for user interaction
- provide help and manuals

Create a simple and natural dialogue

The software product must be able to speak the same language and be coherent to the activity the user intends to use the system for. It must be easy to understand in order to create a good user/software product interaction relationship.
Engineering a simple and natural dialogue is mainly facilitated by two factors:

- always take into consideration the user's work style, character and requirements;
- choose interface solutions that focus on the main task priorities, recognition, feedback and errors.

Some guidelines to use when implementing simple and natural dialog principles:

- get to know and keep in mind your users and their life/work styles when:
  - organizing system contents and structure,
  - implementing interation logic,
  - definining menu orders,
  - defining window and page display orders,
  - organizing contents and objects within windows and pages;
- provide a good system conceptual model and make organization, interaction logic, and other, evident so that user can foresee the effects of their own actions: interacting with a product without having a clear picture on how to go about things is like walking down a dark alley way not knowing what or who is lurking in the shadows until it's too late. Users must be able to see and expect what will happen after each action;
- make evident information that is effectively relevant to the user's needs and intended tasks, by avoiding information which is ir relevant and rarely used. Irrelevant information may distract user's attention and perspective of task at hand;
- anticipate, as much as possible, information that user will find in the next page or following an action.

Simplify task structures

System interaction user tasks and activities should have simple structures. They should be designed and integrated in the system with the bare necessities even though it can be quite tempting for engineers to over elaborate due to the vast variety of design tools left at their disposition and imagination.

- Donald Norman suggests four approaches for task simplification:
- keep task unvaried, but offer mental sustenance;
- use technology to visualize all that would otherwise be invisible;
- automate, keeping task substantially unvaried;
- change nature of task.

The first three approaches substantially envisage the importance of not changing the task the user has been assigned.
The first and second case suggests that external aid needs to be provided (that supports the user's cognitive capacity without forcing them to rely on their own memory) and feedback (that allows the user to control the system's invisible components to the ends of being easily able to keep an eye on their working status). For example, entering data on screen basically has the same identical task structure as handwriting on paper.

However, the former has the help of an automatic corrector to alert errors, allowing user to improve their quality of work.
The third case suggests that in order to keep the task structures unvaried, some of their parts (those more risky and complicated) should be left in the hands of technology and not the user.

The last approach leans towards those tasks deemed intrinsically complex for the ability requested, by suggesting that task prototype, after resulting unsuccessful in the test phase, should be redesigned from scratch if need be: the objective remaining, obviously, the same, but it should be assessed and accomplished from a different angle.

Effective human recognition rather than memory

The user should be able get a clear picture of what and how they must go about their work just by looking at the interface.
Once an action has been performed, the user should clearly understand what has happened and know what the results are.
From the moment it is easier to recognize and remember things by seeing them directly on screen, the less the user is constricted to retrieving information from computer memory.

The simplest way in making life easier for the user would therefore be to make things apparently visible to them through the interface, in other words provide users with external aid to help them remember of jog their memory.
It may seem trivial to say things should be visible, but on the contrary! Here are some examples of what this principle means by inattention:

- made to learn by memory certain commands or information (or revert to operating guide or more expert work colleagues and friends);
- not clear just by looking, whether an item (and edit field, link, button etc..), is selectable, modifiable or not;
- not clear whether the requested action has been executed by the system;
- not clear why action was not executed by system;
- feeling disorientated while page browsing by not knowing where you are or where you started from.

Some suggestions to facilitate recognition:

- exploit natural mapping, this means using the natural correlation between two things, effects, commands, their actions and results (an example of natural mapping would be the direct manipulation of objects. By using these techniques, the user is not forced to remember how to use the objects or describe the actions to be performed: they just simply carry these procedures directly out on screen by moving, for example, a document from the PC's virtue desktop to the recycle bin, as one would do in real life);
- make sure that actions consented within the interface are clearly visible;
- keep system status evident at all times and upon each action performed by user (results of action taken and in what context found, etc..);

- give pages one title, adequately signifying the type of information displayed or the actions to be performed in the window or on that page;
- use language and graphics that are concise in meaning to user to easily understand without needing any translation;
- use selection lists that have easy-to-remember admissible choices and formats;
- provide information preview of selected object;
- give icons and graphic symbols tool tips describing the functionality associated to them;
- enable or disable commands based on their operative content to remind the user which actions are obligatory or have sequential relationships to others;
- stick to consistency when organizing contents and objects within windows and pages, to prevent user from having to search the whole window or page to find what they are looking for.

Provide feedback to keep system status visible

Feedback supplies response information to actions that user has performed on the interface with the aim to keep the user visibly informed on the system's current situation to prevent errors, misunderstanding or interruptions during HCI (Human Computer Interaction).
This doesn't mean that feedback should only be given as error messages on incorrect user actions, but it should also involve other ways of dialoging with the user about what the system is doing at that current moment:

- which action the user has performed or is performing;
- which effects action will have on product;
- product's new status after action has been put into effect.

In addition to messages, mainly used for correcting interaction errors, the ways of supplying feedback on the product's current status is extremely varied:

- observations should be made visible in the window during scrolling selected objects to let user know that certain actions, such as "drag & drop", are being done correctly;
- mouse pointer should change shape after graphic tools have been selected to let user know that certain operations are or not possible, i.e. typing in a text ;
- the appearance of the progress indicator to let user know that product needs a certain amount of time to perform operation;
- Emphasis should be put on object unavailability to stress that a tool is not available or action is not allowed;
- No effect to interface accompanied with an alarm sound indicates that action being compiled is not consented by product's current status. And so forth.

How long should feedback take? Our recommendations are for industrial systems based on Jakob Nielsen's studies and evaluations:

- 0.3 seconds, the approximate time needed in order to give user the sensation that system has reacted instantly;
- 2 seconds approximately, is the maximum time needed to show results of user's actions, even though user may notice a delay in system response;
- 5 seconds approximately, is the maximum time needed to access the less frequented system pages. Pages most frequented should be accessible within the time previously indicated above.
- 7 seconds approximately, is the maximum time needed for accessing routine data logs, applying off-line production analysis on filters and database files (i.e. historical events or trends). The user may notice a slight delay, but action will not have any effect on the process being managed.

Movicon 11
MONITORING VISION AND CONTROL

The use of the Movicon Toolbox and library objects will accelerate project work in creating coherent user interfaces, by choosing the most suitable graphical model.

In addition to the toolbox objects, there are libraries containing a vast range of different ready-to-use symbols to help visualize a more exact and coherent representation of the plant quicker. The libraries can be amplified and customised.

Purposely designed tools have been provided to create project page navigation using the "embedded screens".

- 10-20 seconds approximately, the maximum time needed for accessing consistent data logs, for carrying out off-line production analysis applying filters and database files (i.e. extracting data for reports, statistics or other). The user may notice a delay but will be kept fully informed on the progress of actions without effecting the process run.
  In cases of log file consistency, it would be right to inform the user that the operation in progress will require some time, allowing the user to cancel the operation if need be.

Response times can depend on many factors (server performances and the amount of memory variable, type of connection speed to PLC, amount of information that must be transferred, etc.), but the users will not be interested in these reasons. If users are left waiting too long they tend to think the service is not up to standard and therefore blame the product's makers rather than their own incompetence. It is either a case of 'a bad workman always blames his tools' or 'I want it all and I want it now". But whatever the reason, this is where client trust starts to decline.

Prevent interaction errors and facilitate recovery

Committing interaction errors with a new product is quite normal. Errors are bound to happen now and again! Let's put it down to Murphy's law. Each user perceives their action as an attempt towards the right direction. The error, in fact, is nothing but an action specified incompletely or inaccurately. We are dealing with a natural user-system dialog phenomena which is to be tolerated enough to guarantee the right flexibility of use in order to allow users to surf freely without wandering down dark alleys and bumping into critical situations.
There are some types of errors which are difficult to eliminate, such as slipups or oversights:
an action can be unconsciously performed differently to the one originally intended, due to distraction or interruption.
Other kinds of errors, however, can be prevented with good interface designing: errors may occur due to applying the wrong interaction rule or nsufficient and inadequate information or knowledge. Submerged between these error types are those due to a dialog model which the user does not understand or does not meet their expectations.

The user is therefore apt to applying the wrong interface rules in respect to those required by the product.
The main contribution to interaction error prevention derives, therefore from the dialog principle view point, from tasks, recognition and feedback. These free factors are based on allowing the user to single out, refocus and adapt their actions to the possibilities offered through the interface.

Other ways of preventing errors involve using block functions, which impede the continuation of wrong actions or those which may result destructive.

Nevertheless, since errors are possible, it is also important that the system be designed to diagnose them when they occur and help correct them.
The easiest ways to ensure that this happens would be to:

- provide a function to cancel operations to restore conditions back to default and implement OK requests for those operations considered dangerous;
- provide an effective messaging service (also see feedback);
- avoid presenting pages without surf or browse options;
- keep functions for restoring program or returning to home page available at all times;
- provide commands for interrupting very long operations (where possible).

Consistency

Consistency means that dialog syntaxes (language, input fields, colours, etc.,) and semantics (behaviour associated to objects) should be uniform and coherent throughout the whole software product. Consistency permits the user to easily transfer their knowledge of one application to another, increases action and system component predictability and favors learning.
One problem usually found with consistency is related to fonts and menus or links. More than often pages of various font sizes, styles and colors can be seen within the same program. Analogically, links are proposed in a variety of formats and colours elaborated with a lot of fantasy.

The graphical pages are synonymous with pages of a book. You will never find a 'serious' book with different character types for each paragraph, and different colourful sized headings positioned at in the middle of one page header and to one side on the next page. Focus is easily lost when confronted with a new style each time the page is turned!

Font, page structure and graphics inconsistency not only causes confusion but gives the impression that no attention or care has been taken in design and therefore implies a lack of professionalism on behalf of product makers.
To summarize, consistency should be guaranteed at several levels:

- language and graphics consistency: the same word, icons and colours should each identify the same type of information or the same type of action throughout the product;
- effects consistency: the same command, action and object should have the same behaviour and produce the same effects in equivalent situations; do not associate the same command, action or object with different behaviours;
- presentation consistency: the same object or the same types of information should be basically collocated in the same position, in the same format and order;
- consistency with different application environments: an application and a website are not worlds apart! Users use different applications to surf different websites, and learn how certain interface objects work. When entering in our application/ website one expects to find the same object typology that behaves in the way already known to users. Alternative solutions use unconventional graphical objects that behave in ways un known to users, promoting user uncertainty and discomfort and, without a doubt, open the gateways to interaction errors.

Movicon 11
MONITORING VISION AND CONTROL

The use of text in Movicon has been designed for giving the project engineer a hand in language consistency.
Wherever a text is entered, it will be formatted accordingly to project setup (font, style, dimensions and text color).

The texts can be managed using the String Table Resource, compatible with Excel, to guarantee dynamic language change support. The project strings can be linked together and exported or imported. Unicode support guarantees use of the Asiatic, Arabic and other languages.

Speak the User's Language

The language used at interface level should be simple and familiar to the user and reflect the concepts and terminology well known to them.
Technical terms and system oriented language should be avoided as much as possible for those users who do not understand foreign languages. Words such as "default", "directory" or phrases such as "document can be transferred via ftp" should be avoided and replaced with words recognizable to user.

Icons and metaphoric images and shapes are also used as language in visual imagery form to convey concepts that when designed effectively are easier, more efficient and direct to understand than actual words. Defining a suitable and significant language for the user, especially if symbolical, is however not an easy job and entails getting to know the user and their world to accomplish this.
Two rules of the thumb:

- check user's language capacity (labels, instructions, lists, etc..);
- stick to using already tried and tested icons and symbols; if they wish to use the original symbols, best test their comprehensibility with the product's end user first.

Facilitate flexibility of use and user efficiency

Concentrate on the User's productivity! Purposely design tools with simplified flexibility and efficiency capable of meeting the needs of the user and their tasks in relation to their level of experience and know-how in computer technology.
The results in relation to these two aspects, the level of requested support, tools used and interaction strategies put into action by the users, may be different.
For instance, a guided step-by-step approach would be more suited to the non-expert user whereas a more expert user would work better using shortcuts, which of course can be used by the non-expert users as they gain experience.
Flexibility and efficiency of use can be facilitated by:

- insert 'facilities' (i.e. system anticipation of entered terms/text) and accelerators (key combinations such as CTRL V);
- skip or jump to main points to avoid passing through intermediary points;

- custom interface functions by modifying a few of the systems aspects based on task needs, user's character and their personal preferences. Note that once customized, the system should maintain those settings appropriated for user when re-starting.

Some interface aspects that may need customizing are:
- the language,
- character sizes,
- default settings,
- data formats and detail level,
- graphical object availability.

Another aspect of efficiency to consider is the system's response time to user actions, which is one of the most critical problems in web applications.
Response time to each user call may depend on many factors (server and network performances, connection speed, and the amount of information to be transferred, just to name a few). Not concerned with these endeavors, the user is rather apt to blame the product itself, triggering off a breakdown in trust with the product vendors.

Provide help and manuals

The argument for documentation (on-line help or user manuals) is rather a controversial issue for many a reason:

- a good product, theoretically, should not require user to consult documentation;
- documentation tends to compensate for product usability problems;
- in most cases, users usually don't bother using these support tools.

As regards to the last point, users are apt to seek help on-line or in documentation as a last resort and do not always find the solution they are looking for.

The documentation contents seldom cover indepth information on most subjects and do not provide adequate information at a glance or by simply skimming through the first few lines quickly. Lastly, as for all written texts, if the help online and manuals are not controlled accurately together with the end users, it cannot be guaranteed a reliable source.

By taking into account these aspects when documentation actually does become a necessity for the user, it should be written with the aim to guarantee:

- easy consultation,
- comprehensibility with brief and concise texts,
- orientated towards user's activity,
- problem solution effectiveness.

Using Touch Screens

Interfaces which are found in PC applications have been mainly thought and designed with input devices in mind which have been used in PCs for more than 20 years, such as the keyboard and mouse. For the last few years the Touch Screen has become very popular for major user intuitivity used as input devices.
The advancing multitouch technology today has brought along with it different aspects to the game rules. The iPhone has only opened the way: now the multitouch technology is standard in the field of middle/high ranging telephony, becoming the driving force in launching a new category of 'tablet' devices, gradually becoming a forerunner in the field of notebooks.

The possibilities opened up by more "natural" and simple interaction (NUI, Natural User Interfaces) are numerous and should be fully exploited by new generation interfaces, including those industrial, keeping in mind the limitations imposed to the designer. For example, finger touching commands on screen can be tricky and not as precise as the mouse pointer.

We therefore propose a series of techniques and helpful guide lines to make your interfaces more usable, simple and pleasurable with the use of the touch screen:

1. Use natural interactions
2. Size interactive elements adaquately
3. Keep wandering hands in mind
4. Forget hovering and tooltips
5. Optimize textual inputs

Movicon 11

MONITORING VISION AND CONTROL

Movicon 11 offers a graphical editing environment which simplifies the realization of pleasant and coherent user interfaces. The use of Layout tools, zoom, alignments, duplications and column formation, dynamic event functions, transparency, coloring, tooltips, and many others available from menus and toolbars, allow the project engineer to quickly and efficiently apply the "usability" concepts and coherency to their projects.
The properties of these objects provide numerous tools for managing both the statistical and dynamical parts of each individual object and drawing.

1. Use “natural” accredited to this

One of the biggest attributes of this new generation of interfaces is its ability to make interaction more natural eliminating fast becoming obsolete input devices such as the mouse.

How? The onscreen elements can be easily manipulated for immediate interaction using a series of spontaneous hand gestures.

2. Size interactive elements adequately

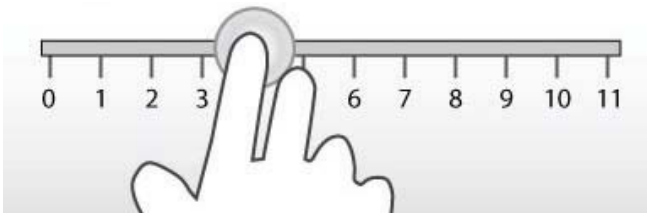
One of the first things that you notice when using a touch screen interface (and probably the most criticized) is the scarce precision our fingers have on the input device after being too accustomed to using the mouse.
This, however, is partly true: this is also due to the fact that interfaces were (and still are) designed to work with the mouse cursor, which has 1-2 pixel precision that fingers do not have.
Touch target sizes must be big enough to permit accurate interaction. Exactly how big?

A MIT study, entitled Human Fingertips to Investigate the Mechanics of Tactile Sense (PDF), discovered that an average sized fingertip measures 8-10 mm. From here emerges the recommended minimum size for each interactive element.

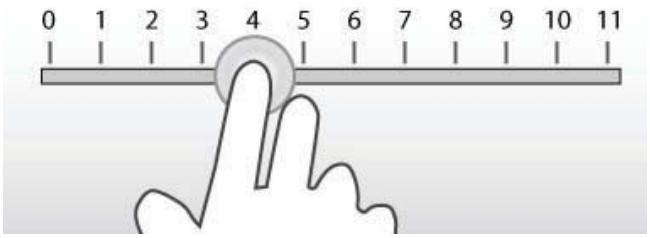
The design should follow this simple equation “+ important = + bigger” to make life easier for the user.
The interactive elements should then be adequately spaced out: targets too close together increase error probability, where user may press the wrong element unintentionally.
Standard spacing between elements is 1-2 mms.
All these considerations are relative in one way or another to the Fitt law, which describes the time needed to reach a target, its size and distance from starting point using a mathematical formula.

3. Keep wandering hands in mind

One of the characteristics of NUI interfaces (touchscreens) is the fact that the input device corresponds to the output device (the screen in both cases). This, as we have seen, makes things easier but at the same time may create problems: hands covering important interface elements when they shouldn’t be.



The interface should therefore be designed keeping in mind these restrictions: positioning information so that it does not get covered by hands when needed. Avoid positioning labels underneath the interacting elements.
The first example shown is modified and corrected in the next example below.



4. Forget hovering and ToolTips

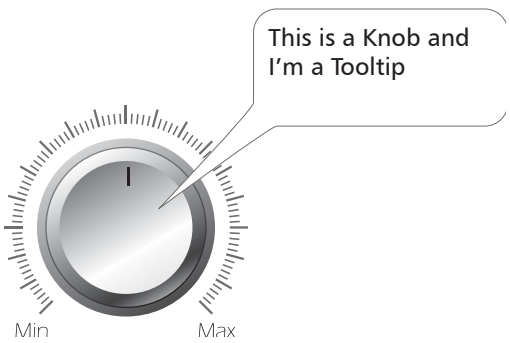
The touch screen interface design engineer must remember that Tooltips and Hovers are absolutely useless in touch screen systems. These tools are very handy when passing the mouse pointer over to an object or graphical element, which highlights upon contact or a Tooltip pop up appears.
In a touch screen system, however, these are useless because when the user touches the graphical element directly it activates without any roll-over effect. Therefore these tools should not be considered in graphical interface designs created exclusively for touch screen systems.
A very important verdict of this observation is the following:

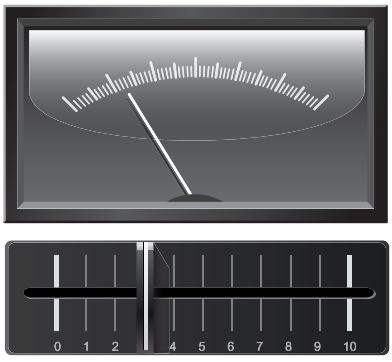
"Make immediately clear what is interactive and what is not in the interface"

In a touch screen system it must be clear to the user what can be touched, manipulated, dragged, etc., and what cannot be touched without relying on the mouse to pass over it.

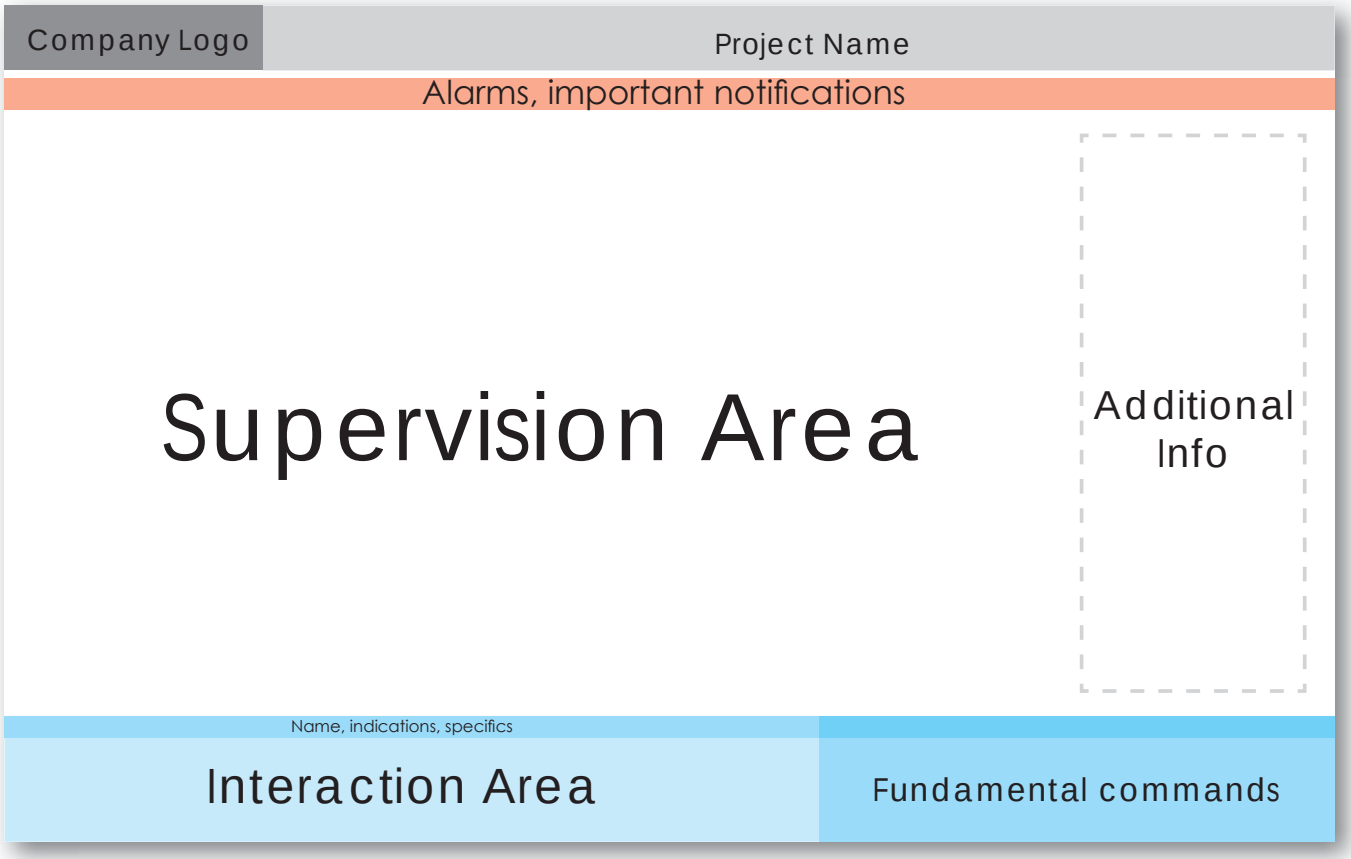
5. Optimize textual inputs

Even though the best graphical interface, users may often have to digit and compose numeric values and text strings with ultimate precision (i.e. set points, log-in and password), and the input from the keyboard onto the touchscreen interface should remain firm in a designated point for efficient use. It is no coincidence that, parallel to launching the iPad, Apple also proposes a mini docking station for its new tablet product, equipped with the classical QWERTY keyboard which is very handy to use whenever at home.
This means that in a touch screen interface system, user interaction should be managed-as much as possible-by means of using the predisposed graphical objects to avoid entering numbers and texts (i.e. cursors, sliders, spin, etc.).
Written input requests for the user must be kept to the minimum. Obviously there are cases where this is not always possible (i.e. inserting user names and passwords), but in other cases this can be simplified or substituted with something else.
When necessary virtual numeric or alphanumeric keyboards can be exploited as long as they do not cover or overlap vital information on the interface where they are displayed.

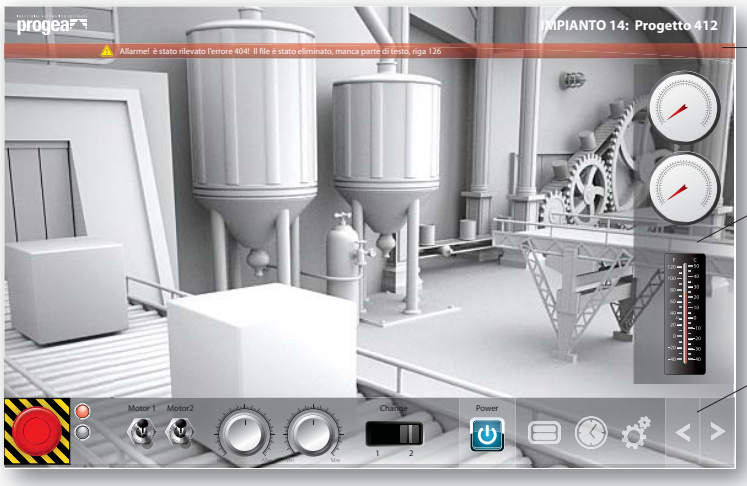




Layout



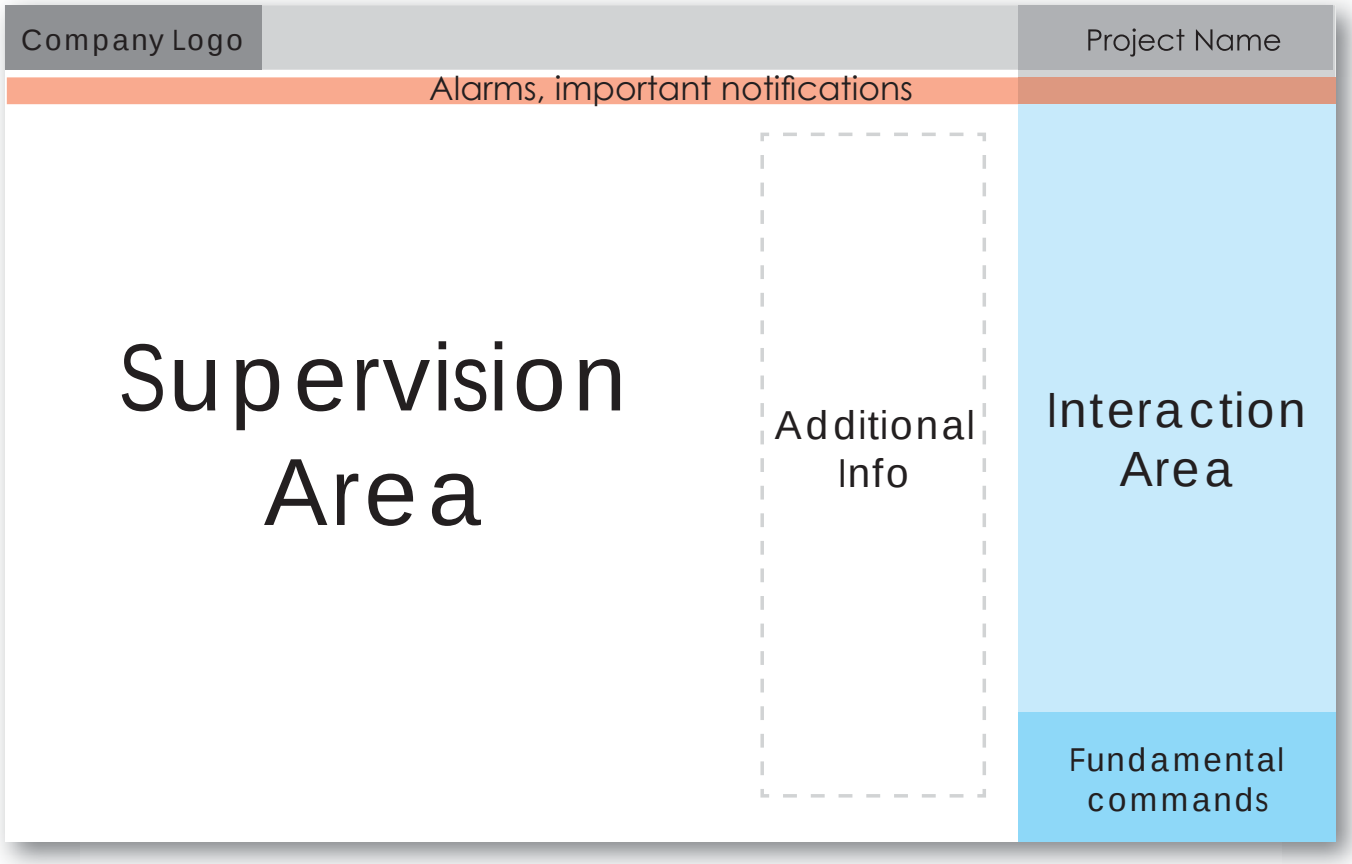
The Layout software must provide the user with a dialog model which is consistent to the user's activity model and system expectations and that, in any case, interacting with the actual product is very intuitive.
The tasks or activities that the user must perform by interacting with the system must have a simple structure.
They must be designed and implemented in the system keeping information processing to the minimum for the user.



The user should be kept informed on which context they are working in and which project/page/section they are currently working in for maximum clarity of the related information displayed.

The secondary controls should be clearly visible, they should not disturb the general working of the interface and should only carry out their task in providing information or commands referred to the context in question.

Command positions are fundamental for navigating projects.



Usability Principles

Based on a study of human factors and ergonomics in a user-centered approach to software product interaction, the usability principles underlie important issues and guidelines to be referred to when designing and evaluating product usability (Jakob Nielson's "heuristic principles" a well known evaluation method, hence heuristic evaluation).

The principles explained in the rest of this document are to be considered a discrete written summary of a vastly dispersed argument:

- create a simple and natural dialogue
- simplify task structures
- facilitate recognition as opposed to user memory
- provide adequate feedback rendering system status clearly visible
- prevent interaction errors and facilitate recovery
- be consistent
- speak the user's language
- facilitate flexibility and efficiency for user interaction
- provide help and manuals

Alerts must be positioned and evident so that the user can see and view them at any moment.

Commands should be positioned so that the user can interact with them intuitively.

The supervision area should be big enough for easy viewing.

INDUSTRIAL AUTOMATION SOFTWARE

progea

Progea is a leading company in Scada/HMI software solutions.
In operation in the software sector for automation since 1990, Progea now boasts innovation-oriented technology beyond the cutting-edge, with a young highly motivated team and steady growth in the international automation market.

Progea Srl
via S.Anna 88/E
I-41122 Modena Italy
tel +39 059 451060
fax +39 059 451061
info@progea.com
www.progea.com

Italian Branch
via XX Settembre 30
Tecnocity Alto Milanese
20025 Legnano (MI) Italy
tel +39 0331 486653
fax +39 0331 455179

Progea Deutschland GmbH
Marie-Curie-Str. 12
D-78048
VS-Villingen - Deutschland
tel +49 (0) 7721 9925992
fax +49 (0) 7721 9925993
info@progea.de

Progea USA LLC
2800 East Enterprise Avenue
Appleton, WI 54914
Tel. 888.305.2999
Fax. 920.257.4213
info@progea.us

Progea International Ltd
via Penate 16
6850 Mendrisio - Switzerland
tel +41 (91) 9676610
fax +41 (91) 9676611
international@progea.com

Quality Certified
UNI-EN ISO 9001:2008

OPC FOUNDATION

Microsoft CERTIFIED Partner