

INDUCTIVE AUTOMATION DESIGN SERIES

Design Like a Pro

Boost Your Skills in HMI / SCADA Project Development

Part 2: Developing Dynamic HMI / SCADA Projects with Speed and Precision

The design phase is the point in the project development process when the success of your project is most vulnerable; it is also the point when your project can go from ordinary to extraordinary.

The danger of getting lost in endless development work can bloat your project beyond its original budget, threatening to spiral your timeline out of control. To navigate this minefield of potential perils you have to work smart, you have to move swiftly, and you have to stay focused on the target.

In this paper we will share some time-saving tips to help you develop dynamic HMI (human machine interface) and SCADA (supervisory control and data acquisition) projects with speed and precision.

Design Like a Pro Series

At Inductive Automation we hear from professionals all the time about how they are using Ignition to create great HMI / SCADA projects. In addition to offering software solutions for the manufacturing industry, we are constantly striving to support the Ignition community with the training, consulting, and knowledge they need for success. As part of that ongoing effort, we are offering this "Design Like a Pro" series of white papers and webinars to give you information that will help you design a successful HMI / SCADA project every time.

In this installment of the series the design phase will be explored. Once you've planned out your project (phase 1), this second phase is focused on putting all the pieces together quickly and efficiently.



800.266.7798
www.inductiveautomation.com





THE DESIGN PHASE

The design phase is when all the pieces of your project are created and assembled; it is when your vision becomes reality. In the process of turning your project from pencil sketches into practical applications, it is important not to get sidetracked. If followed, the steps below will provide you solid footing on your way to successful completion of the design phase.

Design Phase Steps:

Step 1: Set Up

Establish all necessary connections to prepare for design work.

Step 2: Layout

Design the overall framework and navigation scheme for your project.

Step 3: Templates

Create consistent templates and user-defined types to expedite project development.

Step 4: Development

Assemble and link together all screens and components that make up the project.

Although every project is different, these steps should give you a solid outline for the design phase. For a checklist to help you stay on target throughout the design phase, see Appendix A of this document.

Lay the Foundation: Make Sure Your Plans Are in Order

To facilitate your execution of the project, you should already have it planned out. An impatient HMI / SCADA project developer will jump head first into the design phase without proper planning and foresight. In design terms, this is tantamount to setting out on a hike through the desert with no water, map or supplies. You might make it, but the odds won't be with you.

Be patient and thorough in planning your project and you will be rewarded in the end. Spending time to properly assemble all the pieces of your project may seem like slow going at first, but in the long run it will save you an incredible amount of time and grief.

The planning phase is covered in the first installment of this series – Laying the Foundation for Successful HMI / SCADA Projects. To review these steps go to www.inductiveautomation.com/resources to download the white paper and view the accompanying webinar for Design Like a Pro: Part 1.

Once you are done planning the project, it's time to start putting it all together – well, almost. Although much of your time in the design phase will be taken up in development time, the design phase is still largely about putting together the individual pieces of your project.

STEP 1: SET UP

Before you can start designing anything, you have to get your design environment set up. Setting up the design environment consists of connecting up the data sources you will need for your project.

Depending on the scope of your project, you may have many different sources of data, but at a minimum, most HMI / SCADA projects require you to connect to PLCs (programmable logic controllers), and many projects require database connectivity.

Connection Checklist

When setting up the design environment you have to make several connections. To do this you need the connection information for all the PLCs and databases to which you need to connect.

A good time-saving tip is to gather this information and save it all in one place. You can do this by creating a simple spreadsheet on which all the necessary connection information is stored.

The basic categorization of information you will need to connect to PLCs are the PLC name, IP address, and the type of PLC. For databases, you'll need the database name, IP address, type, instance, and login information (username and password). It may take some time to gather this information, but this very simple step can save you much aggravation and lost time spent on scrambling for the connection information at the last minute.

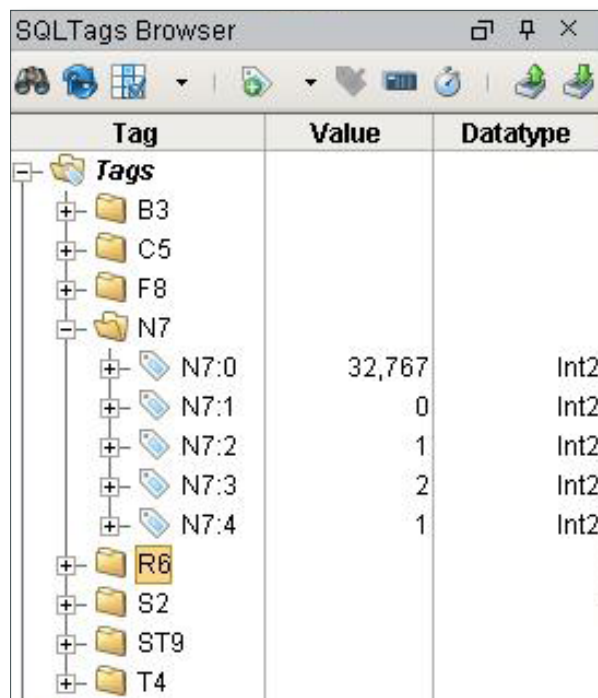
Tag Organization

Upon connecting to a PLC, the individual tags of the PLC can be seen in your design environment. Each SCADA application handles this differently, but many have a tag browser where tags can be managed and organized. In order to make working with tags easier it is important to take the time to organize them. This is especially true if your project has to connect to numerous PLCs and deals with a large number of tags.

While some of the newer PLCs have their own organization for tags, you should not necessarily rely solely on that. Organize your tags in a way that makes sense for the project and screens.

Keep in mind that it needs to make sense to you, as well as the next person who works on the project. Otherwise, the next person to make changes may get lost in your tag organization.

Use folders to create a hierarchy of tags. For your main folders, start with big categories such as line 1 and line 2. Then use sub-folders inside your main folder to further categorize the information. Ideally, your folder organization should be clear enough that someone could easily navigate to a specific tag without knowing too much about the project. Making individual tags easy to find can save you a lot of time hunting for the right tag on the PLC.



Tag	Value	Datatype
Tags		
B3		
C5		
F8		
N7		
N7:0	32,767	Int2
N7:1	0	Int2
N7:2	1	Int2
N7:3	2	Int2
N7:4	1	Int2
R6		
S2		
ST9		
T4		

Leveraging Databases

The best tip about databases is to use them. With so much information available to track, it really is essential to use a database as your data repository. Data that is in a database is infinitely more flexible, accessible and useful than data saved in a proprietary file on someone's computer, or sitting in the memory of a PLC.

STEP 1: SET UP

If you want to save time and effort in the future, it is certainly smart to get your data in a database. It's also important to choose an HMI / SCADA software that can easily work with standard databases such as MySQL, Microsoft SQL Server, and Oracle. You don't want to spend all your design time fighting with software that makes connecting and working with databases hard.

Define Database

To ensure that you make it clean and easy to find information in your database it's important to take the time to define its structure. Before you create a new database and throw information into it, you need to specify where in the database the information will be stored. In terms of an HMI / SCADA project, defining a database means specifying which tags in a PLC

will be saved to which table and in which database. If you're doing recipe management or product tracking, this also includes specifying the tables and data relationships for these systems.

Database-driven Ignition software has the built-in ability to define tables in a database for you. If using other software, you will need to define tables in a database application such as MySQL. In either case, you want to keep your database tables organized by naming them.

Name your database tables with some sort of reference to the type of information that will be stored in them. Applying organization and structure to the information that powers your project will save you time because it makes finding and working with data easier.



STEP 2: LAYOUT

Upon connecting the necessary PLCs and databases, the next step is to lay out the basic framework of your project. In simple terms, this means the construction of the basic window that will serve as the skeleton on which the rest of the project will be built.

Projects can vary, but typically the layout of a project framework will consist of a header area for the company logo and project identification information, an area for navigation, and a main area where the components of your project will be displayed. Taking the time to lay out a clear and organized framework for your project will make it clean, consistent, and user friendly.

Consistent Navigation Scheme

The layout of a project's navigation scheme consists of the organization, placement, and display method of links that lead users to the various main screens of the project. A clear and consistent navigation scheme is key to the overall success of your project. Projects with poor navigation can prove to be frustrating and difficult to use. On the other hand, projects with an effective navigation make finding information and performing functions quicker and easier for the end user.

STEP 2: LAYOUT



Consistency

A key to the effectiveness of any navigation scheme is consistency. If the organization, placement and display method of navigation is different on each project page it can be very disorienting to the end user. It will appear to them that each project screen is somehow disconnected, and as a result, the project will not feel cohesive.

If the navigation is consistent, your project will feel more solid and user-centric. This will increase the overall usability of your project, as well as save time and reduce stress for end users.

Organization

You should organize the links of your navigation according to what project screen will be most used. Overview screens display a wide variety of information and are typically very commonly used. As a result, overview screens are often listed as the first link in the navigation of many projects.

Organize project screens into broad categories and use those as the main navigation links. Also keep in mind the need for secondary navigation. For example, if you have a screen that shows all the lines of your plant, you also may want links that bring up each line individually (such as line 1, line 2, etc...).

Placement

Project navigation is essential to lead end users to the information and functions they need within the project. For this reason, when you lay out your project it's important that the navigation be placed in an area of the screen that easily catches the eye.

The top and the left side of the screen are both effective areas for the navigation to reside. When people read, their eyes typically scan the page from the top to the bottom and from left to right. Our eyes are trained to start looking at the top left of a screen when viewing it, so your navigation is sure to be seen if it is placed there.

Display Method

The display method for links can vary widely. Plain text links, buttons, and tabs are all popular display methods for navigation.

All these methods work fine as long as you follow two simple rules. First, don't make it too loud or flashy. Your navigation should be clear, but also subtle enough that it doesn't overpower the other information displayed on the screen. Second, be consistent; simply find a display method that works for the navigation of your project and use it on every screen.

Using a Skeleton Project

Some SCADA applications come with built-in skeleton projects (sometimes called project templates) that can be used. This is a great way to shave off some time when you lay out your project. Choose a skeleton project that will be large and flexible enough to fit your project's needs, and then customize it.

Using skeleton projects can be a great way to get a jump start on a new project, but be cautious. Keep in mind that skeleton projects are purposely designed to be simple and generic so that they work for a large variety of projects. Don't just assume it will work for your project. Take the time to evaluate the skeleton project and make the necessary adjustments to really customize it to fit your project's needs.

Shortcuts like this are great when used properly, but don't let them be detrimental to the overall effectiveness of your project. Lazy design is not good design. A good rule of thumb to follow is that if you are adjusting the organization and structure of your project to fit a skeleton project, and not the other way around, you are probably on the wrong track.

STEP 3: TEMPLATES

Leveraging the power of templates when building your HMI / SCADA project is one of the most powerful time-saving activities you can do. Fully utilizing templates is one of the hallmarks of a true HMI / SCADA project development professional. You'll save time in two significant ways: cutting development time in the short term, and reducing development time in the long-term.

Advantages of Using Templates

Advantage #1

Utilizing templates correctly can cut down development time to a fraction of what it would take to build each element manually. You can put a template together once and then use it over and over again. This can save an enormous amount of time while building a project.

However, the truth is that an HMI / SCADA project is never "finished", it's very common for a project to be frequently updated to fit the changing needs of its users. This is where the next major time-saving aspect of templates comes into play.

Advantage #2

The second advantage is that templates can help you future-proof your project. Templates are built on

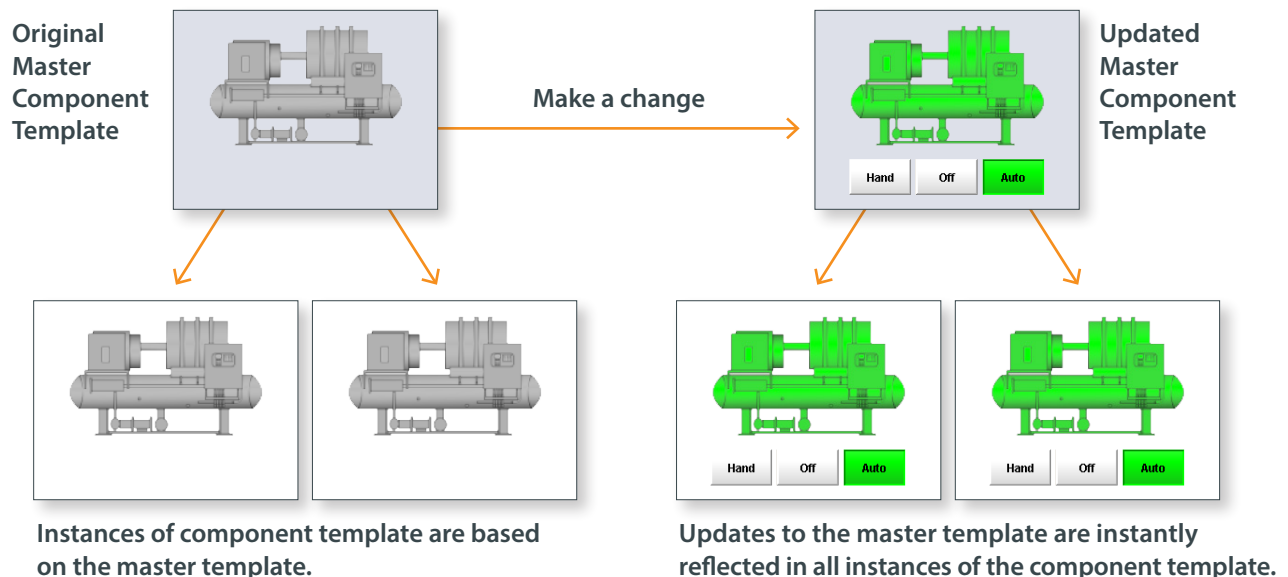
the idea of creating a master template that affects all the other instances where the template is used. If you make a change to the master, that change is reflected everywhere that template is used throughout the project.

This is a huge benefit when making changes to your project in the future. If you want to change the way a template looks or works, it can be done all in one place – instead of having to manually make a change in multiple places throughout the project. A project built upon templates is very flexible and easy to update, resulting in enormous time savings when you're called back in for future project development tasks.

Using the Right Template for the Job

To take full advantage of templates, you must plan for them. Templates must be part of your project development strategy. Do an evaluation of what functions your project will need to perform and what information it will display.

Any repetition in your project is an opportunity to use a template to cut down on project development time. Next, let's discuss some different types of templates you can use when designing your project.



STEP 3: TEMPLATES

Component Templates

A “component” is a fairly general term in the context of an HMI / SCADA project. It is essentially an object that performs some kind of function in the project. A component can range in complexity from the humble button or the common pump to a powerful chart or table. Components are the basic building blocks for constructing HMI / SCADA projects.

A component template is exactly what it sounds like: It is a component that can be saved and used over and over again throughout the project, eliminating the need to recreate it each time.

To take advantage of this feature, make sure you are using HMI / SCADA software that has this ability. Create a new component and then save it as a template. Once saved, the template can simply be dragged into the project screen to instantly recreate the component.

If you need to make any changes to the component template, just open the master template, make a change, and watch as the change is instantly propagated to every instance of the template throughout the project.



To effectively use component templates, create a library of them to use throughout the project. When you begin the design phase, review your project plan to find any cases where the same component will be used repeatedly. Spend time at the beginning of the design phase to create a new template for each of these components. Once you finish you will have assembled your own component template library. With a full library at your disposal you will be able to speed through the process of developing the rest of your project screens.

User-Defined Types

A user-defined type (UDT) is a template for the structure of data. Typically a UDT defines what data from a PLC will be seen in the context of your HMI / SCADA project. To explain how a UDT could be used, consider the following example.

Let’s say you want to monitor five different data points associated with a water tank on the plant floor. Each one of the data points has a specific location on the PLC connected to the tank. A UDT could be used to define the following: where to find the data points on the PLC, what type of data the points are collecting, and whether or not to store historical data for tags.

In this example, if you only had one tank at your plant, the advantages of using a UDT are not that apparent. However, if you had a hundred such tanks, the advantages of using a UDT become clearer. You could apply the tank UDT you created to any PLC attached to the same type of tank and instantly gather the appropriate data points you needed, without recreating the data structure.

UDTs also simplify future changes to the structure of data. Using the same example, let’s say you wanted to set a data point for a tank water level to trigger an alarm. If you used one UDT for all the tanks, then you could simply make a change to that UDT and the change would be reflected on each of your tanks.

Component Templates + UDTs

You can really get the most out of UDTs and component templates when you use them together.

Using the tank example again, if you also created a component template for the tank you could attach your tank UDT to that template to create a component template with dynamic data properties. Use the template in multiple locations throughout your project and simply point it at the PLC data you want to collect. Combining these two types of templates creates an instance of the tank with live data in seconds.

Effectively using templates takes some forethought to put together, but building a project founded on templates can totally revolutionize how you develop HMI / SCADA projects.

STEP 4: DEVELOPMENT



Once you've set up all your data sources and created component templates and UDTs, you should have all the pieces in place and ready to assemble. This stage can vary in length and complexity, but if you've done the work upfront to prepare, then it can be a relatively painless process. However, like any part of creating a project, there are pitfalls that can slow you down.

The next step is to assemble your project in an order that keeps it in perspective. You'll also want to use powerful shortcuts like parameterized windows to help you avoid getting too bogged down in project development time.

Assemble the Project

The order in which you construct the screens of your project can help you move more confidently through development. Generally you should start with the largest screens first; this includes the overview screen as well as the other major sections of your project such as the controls, reports or admin screens.

After that, begin drilling down into the smaller details of each of the larger project screens. For example, on the overview page you may also have links for two different lines in the plant. Each line may need its own project page, and each cell of that line may also need its own project page.

Building a project from big to small in this manner will allow you to see the project as a whole instead of just a collection of parts.

It's common for project developers to get overwhelmed by all the details of a project, leading to a myopic view of one single aspect of the project while the rest goes under-developed.

To avoid this, keep your initial focus broad, and slowly work your way down to the smaller details. Remember that the usefulness of your project will be judged on the whole, so don't get caught up spending a disproportionate amount of time on one minor part of your project.

STEP 4: DEVELOPMENT

Construct Parameterized Windows

Leveraging the power of parameterized windows is a very effective tip for quickly and efficiently developing HMI / SCADA projects. A parameterized window is a blank project window that is populated by information that the user “defines” – usually by selecting something in the project to view.

To illustrate how and why you might use a parameterized window, it’s easiest to look at a simple example. Let’s say that on your project overview screen is a display of several of the compressors on your plant floor. When you click on one of the compressors – let’s say compressor #1 – a pop-up window appears. The pop-up window displays more detailed information for compressor #1 as well as a start / stop button.

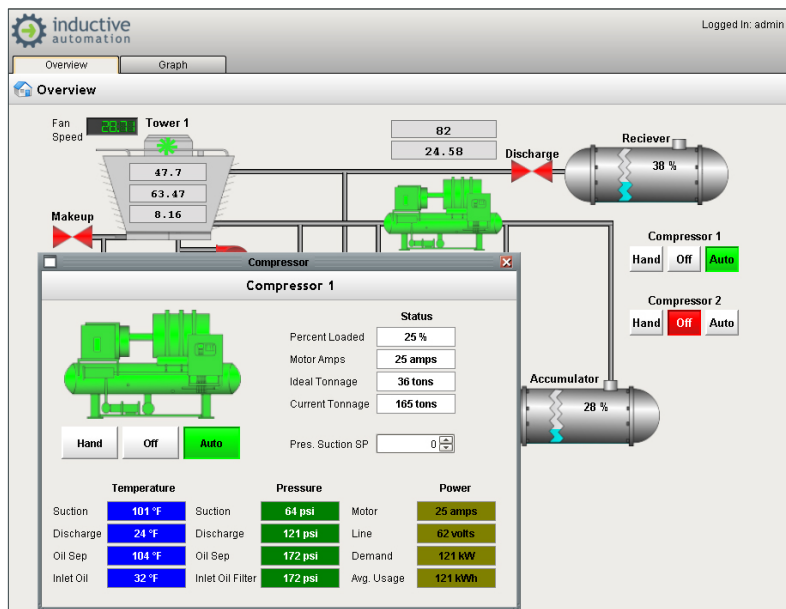
After you are done with that window, you close it and then click on compressor #2, and a similar pop-up window appears with exactly the same type of information and functionality that appeared on the pop-up window for compressor #1.

To the user, the two pop-up windows appear to be two separate but identical windows with the information for each compressor. In actuality, the two

pop-up windows are one and the same; the only thing that changed was the data that was displayed to the user. The user “defined” what they wanted to see by clicking on a button (for compressor #1 or compressor #2), and the window displayed the information the user defined based on the parameters established for the window.

Data is passed to the window through a technique called indirection. Indirection is a way to link together project windows with data sets in a relative way instead of an absolute way. Instead of telling the window the exact location to look for data – like you would tell someone your specific address to find you – indirection tells the window where to find the data relative to where the window is located in the project’s file organization scheme.

You could approach the creation of this type of pop-up window in two ways. On one hand, you could create each pop-up window separately for the entire project, but on the other hand you could create one parameterized window and simply use it over and over again. This can be a huge time saver, especially when working on a large project that has hundreds of such windows.



A parameterized window – like the pop-up window shown in this example – can display information based off of what the user selects on the screen.

MOVING INTO THE START-UP PHASE

By properly setting up your design environment, creating the main windows of your project, building a custom library of component templates and UDTs, and putting all the pieces together in development, you have successfully navigated your way through the design phase.

Following the tips outlined in this paper will help you cut down on project development time so you can create HMI / SCADA projects with speed and precision. Spending less time on the tedious tasks of project development will give you more time to refine the details and innovate new features for your project, making it the best it can be.

The next phase you'll encounter is the start-up phase. The start-up phase is the final leg of your project's journey. During this time, your project must be optimized, secured and tested before it's ready for use.

Inductive Automation is committed to delivering meaningful knowledge, training and products that support the manufacturing and process control industry in HMI / SCADA development. For more design tips, check out the entire Design Like a Pro Series for a collection of white papers and webinars focused on development topics: www.inductiveautomation.com/resources

The image shows the cover of a white paper. At the top, there is an orange banner with the text 'INDUCTIVE AUTOMATION DESIGN SERIES'. Below this, the title 'Design Like a Pro' is written in large white letters. Underneath the title, the subtitle 'Boost Your Skills in HMI / SCADA Project Development' is written in smaller white letters. The background features a dark blue grid with various geometric shapes, including gears and arrows, in shades of blue and white. At the bottom, the main title 'Part 3: Designing HMI / SCADA Projects that Deliver Results' is written in large orange letters. Below this, the text 'Coming in April 2012' is written in white.

INDUCTIVE AUTOMATION DESIGN SERIES

Design Like a Pro

Boost Your Skills in HMI / SCADA Project Development

Part 3: Designing HMI / SCADA Projects that Deliver Results

Coming in April 2012

Design Phase Checklist

Appendix A

PROJECT DEVELOPMENT PHASE: DESIGN

Designing a successful project can be difficult; it requires patience and a focus on the task at hand. To help you stay on course, here is a checklist of the basic steps to take as you move through the design phase.

Every project is different, so one path won't work for every project, but this list is a good reference point and should help you stay on solid footing.

1. Set Up

<input type="checkbox"/> a. Create Connection Checklist	due by:	assigned to:
<input type="checkbox"/> b. Connect to PLCs	due by:	assigned to:
<input type="checkbox"/> Organize Tags	due by:	assigned to:
<input type="checkbox"/> c. Connect to Databases	due by:	assigned to:
<input type="checkbox"/> Define Databases	due by:	assigned to:

2. Layout

<input type="checkbox"/> a. Create Main Windows	due by:	assigned to:
<input type="checkbox"/> b. Create Navigational Scheme	due by:	assigned to:

3. Templates

<input type="checkbox"/> a. Create Component Templates	due by:	assigned to:
<input type="checkbox"/> b. Create UDTs	due by:	assigned to:

4. Development

<input type="checkbox"/> a. Assemble Main Project Screens	due by:	assigned to:
<input type="checkbox"/> b. Create Parameterized Windows	due by:	assigned to:
<input type="checkbox"/> c. Complete Remaining Screens	due by:	assigned to: