IEC 61499 for developing distributed automation systems Environment

Salman Azim	245038
Prasun Biswas	267948
Shambhu Mishra	267600

Introduction:

goal of this assignment on Distributed Automation System Design course was to be familiar with IEC-61499 architecture and get skill on designing industrial operation following the protocol. The task requires design and implementation of a production system where a product is made after couple of assembly, welding, painting, and quality check operation, then released after packaging in a pack of four products. We implemented the whole system following ICE-61499 and MVC pattern for system design. We used FBDK tool by, HOLOBLOC for completion of the project. In the following part of the report we briefed about IEC-61499, MVC and in detail explained about how we implemented these using the functional options of FBDK tool.

Overview of IEC-61499:

The international standard IEC-61499, addressing the topic of function blocks for industrial process measurement and control systems which standardizes a typical model to follow for implementation of distributed control systems and is based on the IEC-61131[1]. The cycle execution model of previous version is replaced by event driven execution model where sequencing of execution order of function blocks is possible to design the system [1]. An application-centric design allows one/more than one application defined, to create a whole system and allows to distribute to available devices. As feature it provides combination of distributed programming language and legacy PLC programming with IEC-63313-3, generic modeling approach for distributed control applications, separation of data/event flow and function block concept [2]. Several function block types used are

- Service interface function block-SIFB
- Basic function block-BFB
- Composite function block-CFB
- Adapter interfaces
- Sub application

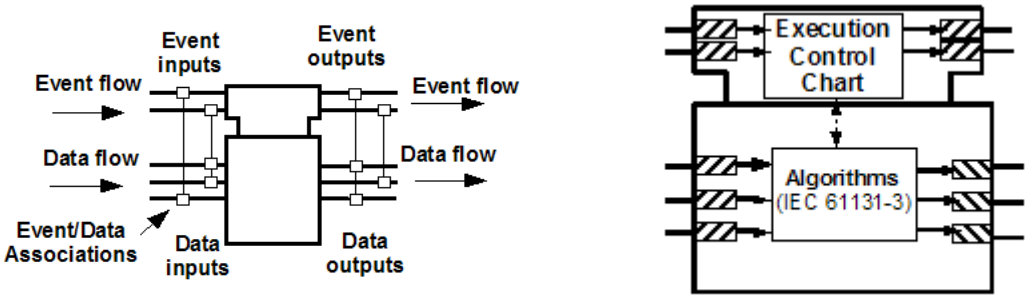

Figure: Function Block Types (1) and Basic Function Block (2)

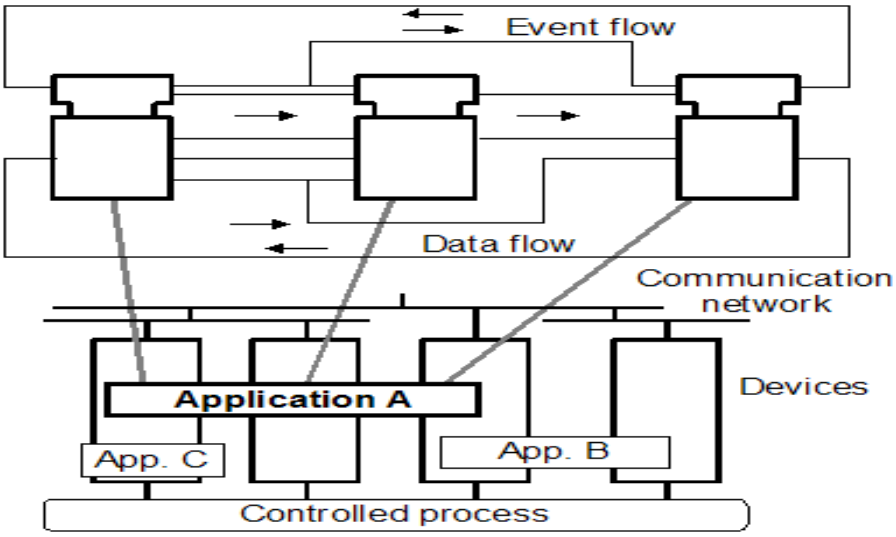

Figure: Distribution of event flow, data flow and controlled process

Overview of MVC:

MVC stands for Model View Controller. It's a software development methodology which has a main target of promoting code usability and implement separation of concerns. It requires division of application in three main components – model, view and controller.

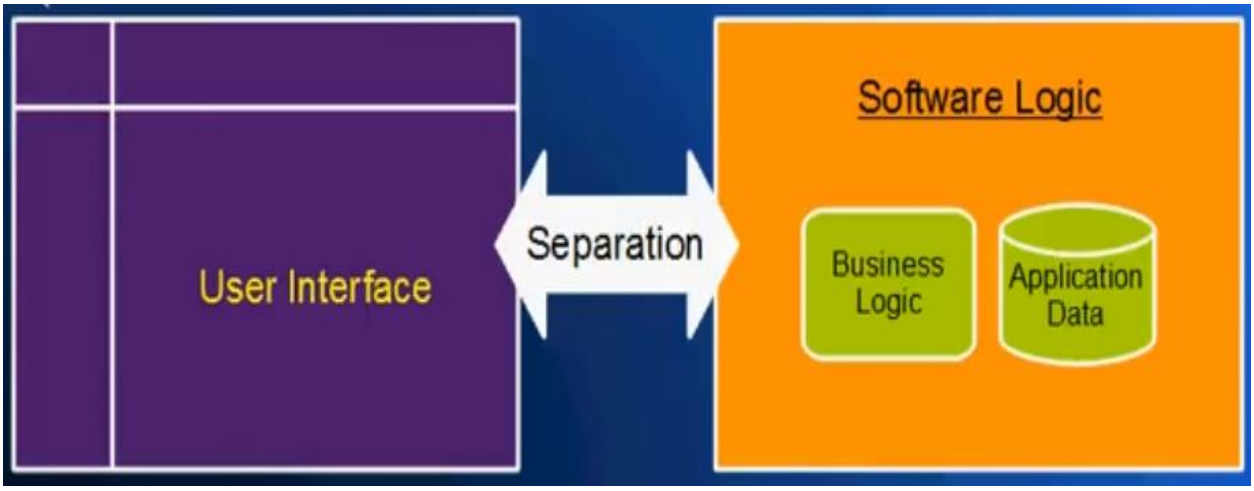

Figure: Separation of concerns by MVC .

Model is the brain of the application which contains basic rules which defines what basic functions the applications is supposed to do with the logic with the data type that the app is supposed to work on and so on. The view component is the presentation layer. It has all the information regarding what to display on the screen, what would be the color and layout, how

many fields etc. The controller components job is to regulate communication between model, view, and users. Users would be interacting using the interface that is provided by view component. The communication will be handled by the controller, where it receives the interactions from and sends it to the model which will perform certain tasks based on the logics, which will give the data back to the controller and then the data will be view on the screen using one of the views

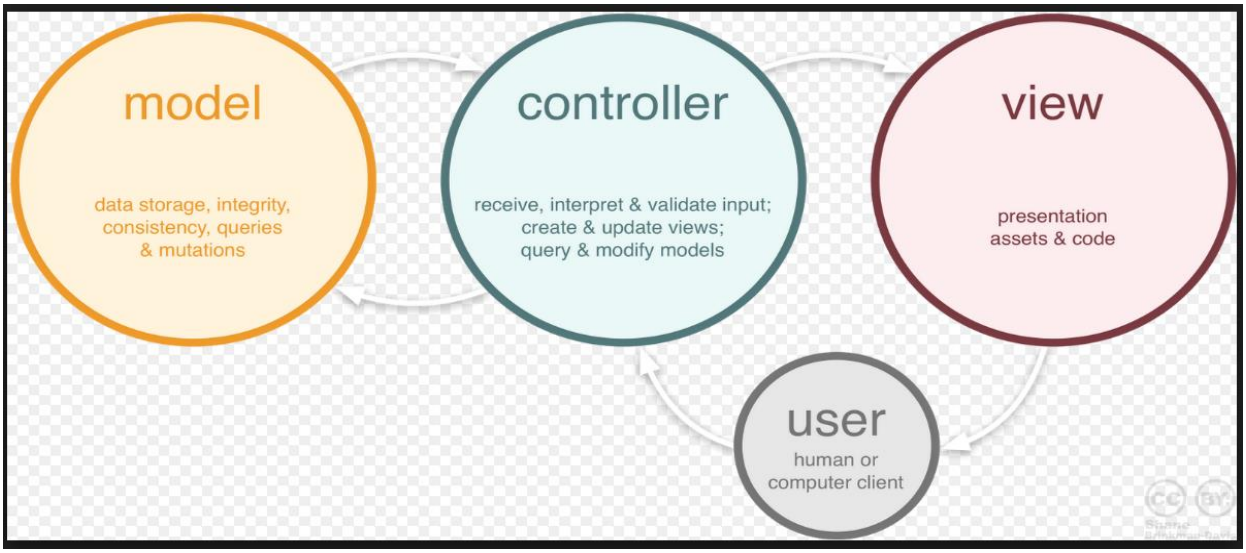

Figure: representation of MVC framework .

Brief discussion on used blocks:

To develop the system, we had to study and understand how the blocks available on FBDK tool works on different events to execute certain logic. Although there are lots of useful blocks provided by the FBDK tool, here we used only those few blocks that are most useful to implement project based on our logical combination of whole system. In the following paragraph, we mentioned the name of those used blocks.

FLOW_PANEL, FRAME_DEVICE, PANEL_RESOURCE, VIEW_PANEL, MECH_VIEW, XSPACE, E_SPLIT, SUBL_0, SUBL_1, SUBL_3, PUBL_0, PUBL_1, PUBL_3,CNV_MDL,E_TOGGLE, E_CYCLE, OUT_EVENT, OUT_BOOL, E_RESTART, FB_NOT,E_MERGE, E_PERMIT, E_CTU, IN_ANY, OUT_ANY, FB_LABEL, IN_EVENT, E_SR, E_R_TRIG, FB_LT_UNIT.

Composite Block: To match our requirement we built one composite block and named it **BOOL2**.

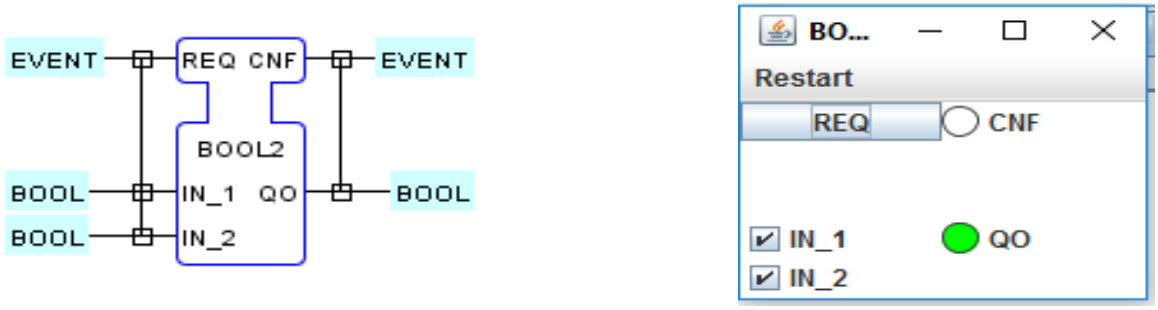

Figure: Composite Block "BOOL2"

This block generates a high bool output while both of the bool input is high and request is received . Along with high bool output it also sends a confirmation signal.

Basic divisions of application:

To build the application four basic division was made to follow MVC design methodology. HMI, Control, Model, and View work together to run the whole application.

HMI:

HMI stands for human machine interface. In this project, HMI is used for two basic purpose. Firstly, its sends different values and commands to start the execution of the application. Number of pallet and speed of the motors is sent from this layer as user input at the beginning, while the start button is pressed, and the speed also can be controlled while the application in running. It also has a reject button which send command to reject a product from quality check phase. Secondly it shows the status of the motor with circle, where red circle means the motor is off and green means the motor is running.

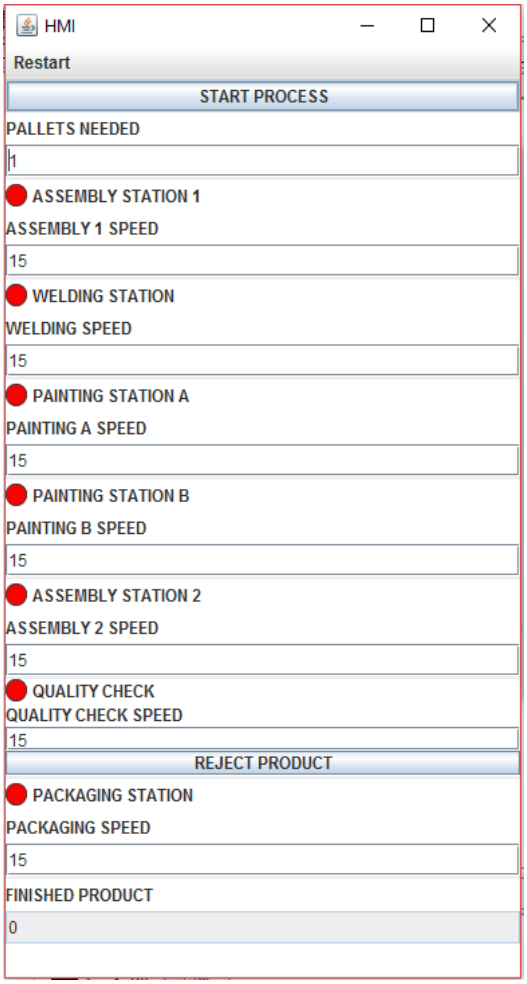

Figure: HMI

For completion of the HMI layer eleven types of blocks are used. **PANEL_RESOURCE, IN_EVENT, PUB_0, IN_ANY, PUB_1, FB_LABEL, SUB_0, E_TOGGLE, OUT_BOOL, E_SR, OUT_ANY.**

View:

The view shows a graphical representation of the system. On the top left position one vertical and one horizontal conveyer is displayed where the first assembly block is located. This block is start working by loading parts which goes to the next conveyer of the assembly block after assembly process is done. It enters the next block, shown with red color, to go through coloring process. After unloading from painting station A it goes to the first entry point of the next station named assembly station 2.

While loading parts in the first assembly station another part is also loaded in the welding station and at the same time with assembly station 1 it goes through welding process. it then enters the next station named painting station B. after the paint job is done it then joins the second entry point of the next station named assembly station 2.

While both parts reach at the end of the conveyers logically the assembly process is completed and then it enters the next station for checking of quality. Here the process is partly automated and partly done by user's decision. if uses decides that a product is good to pass the quality check, a reject button is clicked and the product goes to the downward facing conveyer as a rejected product. Otherwise it is automatically considered as a quality passed product and enter the packaging station. When 4 products are reached on the packaging station the pallet is formed and load/unload process complete the release of the total package.

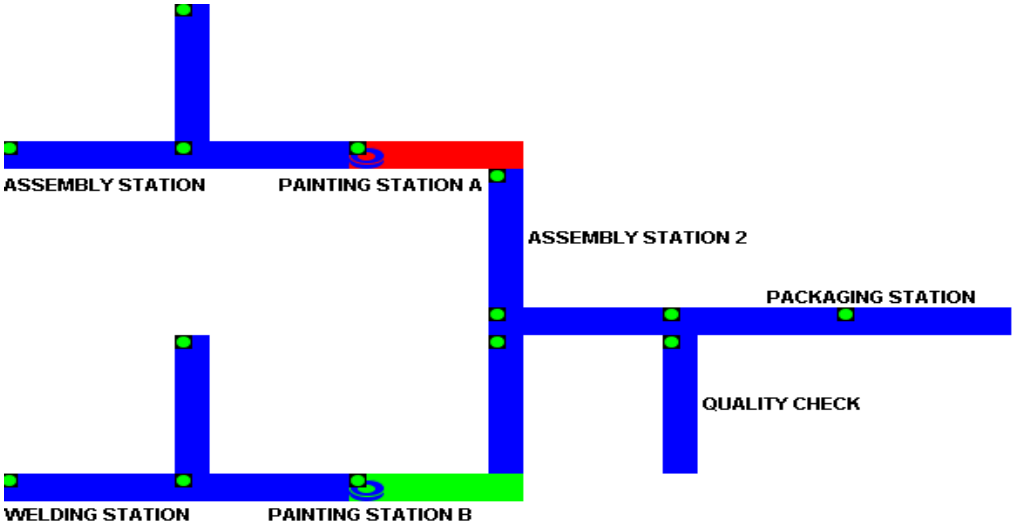

Figure : graphical representation of process.

For the completion of the VIEW layer five types of function block is used. **MECH_VIEW, XSPACE, E_SPLIT, SUBL_3.**

MODEL:

The most important block that the model layer contains is the CNV_MDL. It is a functional representation of a conveyer in the process. which has event input CLK, LOAD, UNLD and event output INDR, INDS. It has data input MTR_ON, FAULT, VF, LPOS, STYLE, WKPC and data output END, POS, STYLO, WKPO. This block contains data about possible speed, color, position, load/unload of pieces, type of piece etc. Based on the model controlled communicate and calculate data to generate output for execution of the logic to implement the whole process. This block communicates with controller through **PUBL_0, PUBL_1, PUBL_2, SUBL_0, SUBL_1, SUBL_3, E_TOGGLE** from this layer.

CONTROLLER:

The whole process is controlled from this layer. It receives user input from HMI layer about the number of pallet to be produce, speed of the conveyers, when to start the process and when to reject product from quality check cycle. It communicates with model layer to check if certain conditions are met or not and from there data is sent to view layer through models. Used block in this layer are **PANEL_RESOURCE, SUBL_0, SUBL_1, SUBL_3, PUBL_0, PUBL_1, PUBL_3,**

BOOL2, FB_MUL_UNIT, E_CTU, E_R_TRIG, E_PERMIT, FB_LT_UNIT, FB_NOT, E_MERGE, E_CYCLE, E_SPLIT, OUT_EVENT.

Workload Distribution:

Primary task was to watch the recorded class lecture, youtube tutorial and study the assignment to understand the development process. to develop the graphical interface and logical operations we met as a group to discuss the whole concept and methodologies. As we followed the described process in the uploaded video by you, we first tried to build the small parts of the projects. We started with designing of VIEW. Then we tried to understand how the provided function blocks by FBDK tool works properly. After that we worked to develop how the complete logical operation will be implemented. We started the implementation by developing the model layer. At first all the logical operation was inside the model layer. After the systems seems to be running well we then started to divide the whole process in model, controller, and HMI layer. We repeated this process for the whole process step by step for all the small blocks.

Names and contribution (%)	Procedures
Salman (30), Prasun(30), Shambhu(40)	VIEW
Salman (40), Prasun (40), Shambhu(20)	Analysis & logic development
Salman(40) , Prasun40(), Shambhu(20)	Model and controller
Salman(30) , Prasun(30), Shambhu(40)	Report

Problem Faced:

At first it was a bit of trouble to understand how the XSPACE works. Finding the right function blocks that matched our developed logic took some effort. The biggest problem was that the software seems to crash several time and sometime the previous works were lost. We also faced problem while transferring the operations from model layer to HMI and controller layer finally. Another problem was that there is not much resource online to now about the process. But the uploaded video was extremely helpful throughout the whole assignment.

References:

[1]. https://en.wikipedia.org/wiki/IEC_61499

[2]. http://www.iec61499.de/index.htm

[3]. http://www.holobloc.com/papers/iec61499/overview.htm