Object-oriented programming is probably the most well-known and used programming paradigm [x]. It is based on the concept of objects and classes, which are data structures that contain attributes (information) and methods (behaviour), encapsulated.

The main advantages of this paradigm are the reusability of code, the modularity of the program and the ease of maintenance. In addition, it allows the abstraction of the functionalities of an existing element with respect to the technical difficulty that these present. The object interface exposes the necessary elements, presented in a synthetic way, which allows the handling of the underlying element in a safer and easier way, improving also the integration capabilities with upper architectures.

In the context of the new trend in industrial digitalization, the integration of the different elements of the production system, from the lowest level of the process to the highest level of the business, enables the usage and exploitation of the system data. Therefore, the data exchange between the different elements has become a key factor in industrial digitalization on the short them. Furthermore, the publication and usage of the functionalities of the different assets by other assets can be used to achieve a more flexible and adaptable production system.

In this regard, proposals such RAMI 4.0, from Platfform Industrie 4.0, is centered towards the assets, which are the elements that compose the production system, and the Administration Shell concept, which is the element that exposes the information and methods of the asset in a standardized way, allowing the integration of the asset in a superior production system in more advanced ways.

One of the key concepts to succed in the digitalization of ICS is the proper definition of data and behaviour models of the assets [x]. It has to be enought flexible and scalable to be successfull. The model of the asset composes the definition and core behaviour of the Administration Shell. Then, new assets can be created using techniques such as composition and inheritance, which can be used to build powerfull and complex systems of integration and interoperability, based on concepts such as inheritance and composition, similarly to the object-oriented programming paradigm, but generalized to the concept of assets.

In order to achieve well-performant, scalable and maintainable Asset Administration Shells, the underlying OT technologies have to be designed and deployed appropriately. From the ground up, the next generation OT systems need to be compliant with the requisites of the

Achieving software with these characteristics, that is, organized trough basic modules focused in the management of the actuators but dependent of the received inputs, would allow us to adapt the machine behavior, to the requirements of the operation, for example adding more types of products, changing the locations of the stations or completely altering the operating principle of the machine, all of them maintaining the same structure of retainers control, modifying just the flow of these through control variables. Is in this way that we obtain a flexible manufacturing system.

To carry out said implementation on the industrial cell at the OT level, an object-oriented program structure has been proposed for the PLCs programs, where each of the retainers of the system is instantiated as an object that executes the corresponding code. By this way, the PLC software is much more flexible to changes on the hardware needs, moreover, code reusability is highly increased, improving the scalability of the industrial process and enabling a fast deployment of new facilities with similar architectures.

The retainers and platforms in series are controlled in such a way as to avoid impact between trays, in addition the retainer can be locked by the control system. Those platforms located in bifurcations have several blocking points, so that the path to be followed by the cell can be selected. In addition, the retainers have an auxiliary input for the platform joints.

Falta explicar las modificaciones en el programa del PLC quizás aquí o despuñes de plantear la arquitectura.

The basic object around which the entire programming is based has been designed following a flexibility pattern. To do this, it is designed to allow for two input directions and two output directions, even though in most cases there is only one of each. Furthermore, activation and blocking signals can be used to determine which path a tray will follow throughout the system, enabling higher level applications to modify the predefined path.

For the transmission of the information a public variable is used upon which the n-1 retainer can write the information of the tray to the nth retainer. The process uses an intermediate variable where the data of this public variable is stored, and upon the ending of the move state this intermediate variable is written into the next retainer’s public one. This allows the retainer to receive a new tray whilst still being in the move state and thus not having yet sent the information to the following object. Once sent, an acknowledge variable is used, that assures that the information of the tray is passed between the different blocks, as a retainer cannot change to move state from ready state until the previous retainer has activated its acknowledge variable (which is done simultaneously with the sending of information of the tray). Once in the move state, the acknowledge variable is resetted.

As mentioned above, the use of object-oriented programming paradigms is key to providing flexibility to production systems, mainly thanks to the ability to provide abstraction of the functionalities of an existing element with respect to the technical difficulty that these present. The software exposes the necessary element, presented in a synthetic way, which allows the handling of the object in the desired way, avoiding the reprogramming of its fundamental functions of operation and easing the integration with upper architectures. Thus, enabling the integration of an effective Administration Shell concept.

Each of the basic objects must have two main elements, attributes (information about the object) and methods (functions that the object acts or performs). The IEC 61499 standard added the function block concept "Function Block" to PLC programming, a more flexible element than classical functions, which generalizes the function concept to allow application-centric design. In PLC programming environments where this standard has not been implemented, data type structures and functions can be used for their application.

Imagen objetos lab

This methodology would be the most similar to the object-oriented programming paradigm that can be implemented in PLCs, even with impediments that do not allow to assume all the usual characteristics of these but being useful for the proposed objectives, reducing the effort for the development of a reliable control system of the low-level requirements of the process.

Finally, an upper-level control application has been developed in the OT platform. It defines some of the business logic behaviour (production process), which is also available to be manipulated by the other actors of the CPS but also exposes some process functionalities, like M2M interactions with other machines related to the asset, production system behaviour enhancements or process control. In the developed architecture some virtual assets have been used to explore M2M functionalities.