IRP 1 Compendium

## 1. Introduction

Advantech Ltd., a company for manufacturing and delivery of sensitive and high value bottled liquids, have decided to build a new manufacturing facility that will automate the manufacturing process within the existing facility, provide advanced system for monitoring and controlling environmental conditions and access and security control.

This project is to develop this solution leveraging IoT concepts, synchronous programming and system-level designing. The chosen language and environment to create this is System J and Java. This is a written compendium detailing to a lower level the components and operation of this full solution.

# 2. Brief

The solution is facility wide. Incorporated in multiple parts of the physical facility as well as in the purpose of the facility, it has the following requirements:

* A developed Automated Bottling System (ABS), as this is facility’s purpose, to output orders of bottled goods
* Be able to monitor who is within the facility as well as specifically where they are
* Be able to provide security in the form of only allowing selected personnel in general and specific areas of the facility (e.g. restricted access to the main office)
* Continuously monitor and adjust the facility's climate factors such as humidity and heat to desired conditions. Even able to adjust the climate of specific areas differently within the facility
* Receive and process orders from registered customers for bottling

These are the high-level requirements from which more detailed and specific requirements stem from.

# 3. Design Notes

The following section explains the components, structure and operations of the ABS. Design approach and implementation of the solution are also explained in this section as well as any assumptions or matters to note.

## Approach & Implementation

In order to implement it was required whatever tools, language and environment to be used allowed the use of IoT concepts, synchronous programming and systems-level design. Prior to beginning the project it was already decided to use System J and Java for implementation.

### System J

System J is a synchronous programming language following the concept of creating globally asynchronous locally synchronous (GALS) systems. Its basic structure is that a collection of synchronous programming statements form a reaction. This reaction is able to have signals inputted into it to be utilised by the program statements as well as emit signals that can be broadcasted and received by other reactions. These signals can be pure signals simply representing a TRUE/HIGH or FALSE/LOW or they can be valued, being able to hold various data types and structures such as an array of integers. A collection of reactions all operating on the same clock tick is known as a clock domain. Clock Domains are asynchronous with each other and instead use defined channels to communicate signals and values to each other. A collection of clock domains is considered to be a System J system. This system will have asynchronous output and input of signals to some kind of environment. It is built atop Java, CSP and Esterel.

The benefit and reason for using System J within this project is that all computation and logic can be done on it, with different operations being able to be run at the same time. This concept is used to enable multiple stations with their own independent tasks to run at the same time and not wait for each other to complete as would be the case in an exclusively sequential, asynchronous programming language. Additionally, with it being built upon Java, Java libraries for connection and interfacing can be integrated well with the System J. The exact breakdown of clock domains and use of libraries is explained more as one goes along this document.

### Java

Java is an object-oriented programmer that makes use of a compiler and built-in virtual machine. It is structured around variables that when grouped together make an instance or object. A type or collection of objects are defined in class, as well as all its associated methods. The reason for using Java within this project is to leverage the various libraries, namely the Java Swing and Java Socket libraries. Java Swing is a library that allows for the creation of graphic user interfaces (GUIs) that can allows visualisation of a system as it operates as well as allow easier interfacing of a user inputting values and commands. This is library is used to make all GUIs that feed, as well as show information to and from the user of the system. Java’s socket library allows for the creation of TCP/IP sockets to communicate locally and externally with other computers and programs. This proves essential to allow for the information on the computation of the System J to be communicated for the GUI to display, as well as when the GUI wants to send collected information from the user.

### Design Approach

The approach was taken was to make the overall system of systems decentralized. The full solution is made of the four sub-systems:

* Automated Bottling System (ABS): The manufacturing component that bottles the liquids produced by mixing two or more liquids based on customised recipes.
* Safety and Access and Control System (ACS): Monitors and Controls presence of people in both major sections of the facility and enforces both safety and security measures according to the adopted regulations such as humans in certain areas.
* Environment Control System (ECS): Controls environmental conditions (temperature, humidity and cleanliness, as well as lighting conditions) in manufacturing and office sections of the facility, as well as control the conditions for storage of bottled products before delivery. Additionally, monitors safety of the environment in terms of presence of fire.
* Product Order System (POS): Accepts the orders from the trusted (registered) customers through an automated on-line system and launches and schedules the production automatically.

Each System has its own collection of clock domains for computation as well as GUI that is linked to help with interfacing with and showing the user.

Specifically, towards the ABS, the approach was taken to have everything centred around a single plant. This plant is the simulator of the entire ABS operation. The central plant receives signals from multiple controllers, one controller for each station of the ABS system. Once it receives these signals, the next step/changes occur within the plant. The result of these changes or next steps are communicated in channels back to the controllers so they can make the next computation and signals are sent to the ABS GUI to provide it with the information it requires to portray graphically what is occurring. The Gui itself is able to feed values back to the plant. The reason for this approach is allows for each station to be represented by a controller, each made in its own clock-domain. This allows the stations to operate independently of each other and at the same time. The plant being in the centre serves as the component that makes sure everything is done at the right timing to ensure a clean flow of operations. The controllers handle external signals from the other sub-systems so as to have the incoming data already sent to where it is being used. For a graphical representation for this, refer to the report which is included within the package alongside the compendium.

### Specification differences