Fort he first version of Flowkloriko, it will be available to create spark workflows from the user interface by specifying at least one of the three required components:

**Source** :[File, Directory, Spooldir, Hive, ElasticSearch]

**Processor:**

**Sink:**

Before of executing the Workflow, the software will be sure that it does acomplish the requirements for the given configurations of each component.

Then the list of **Actors** that will participate on the preparations for the job to be run are:

**Master:**

*. workflowRegisty: Map[String, ActorRef]*

* *CreateWorkflow(id: String)*
* *AlterWorkflow(id: String)*
* *DeleteWorkflow(id: String)*

**GenericDataBaseController** extends DataPointController

* CheckTableExistance(tableName: String)
* CheckDataBaseExistance(dbName: String)
* TableExistanceConfirmation()
* DataBaseExistanceConfirmation()

**HdfsPathController** extends DataPointController

* CheckFileExistance
* CheckFilePermissions
* CheckDirPermisions
* CheckDirExistance

**Workflow:**

*.flows: List[Flow]*

*.schedule: WorkflowSchedule*

* *RunWorkflow(id: String)*
* *Flow(source: Source, processor: Processor, sink: Sink)*
* *sealed trait DataPoint { val id: String}*
* *trait Source extends DataPoint*
* *trait Sink extends DataPoint*
* *case class HiveSource(id: String, dbName: String, tableName: String,*

*whereCondition: WhereCond\*, limit: Int) extends Source*

* *case class HiveSink(id: String, dbName: String, tableName: String,*

*hiveTablePartition: HiveTablePartition, sparkPartitions: Int) extends Sink*

* *HiveTablePartion(field: String, value: String)*
* *trait Processor*
* *JoinProcessor(left: JoinSide, right: JoinSide, joinType: String) extends Processor*
* *JoinSide(tableName: String, fieldName: String)*
* *DropColumn(fieldName: String) extends Processor*
* *TypeCast(fieldName: String, from: String, to: String)*

***WorkflowScheduler***

*.workflowSchedule: Map[String, (Frequency, ActorRef)]*

* *RunScheduledWorkflows(time: SimpleTimeFormat (yyyy-MM-dd.HH)*
* *SheduleWorkflow(id: Sring, worklfow: ActorRef, frequency: Frequency)*
* *ScheduleWorkflow(id: String)*
* *UnscheduleWorkflow(id: String)*
* *Frequency(???)*