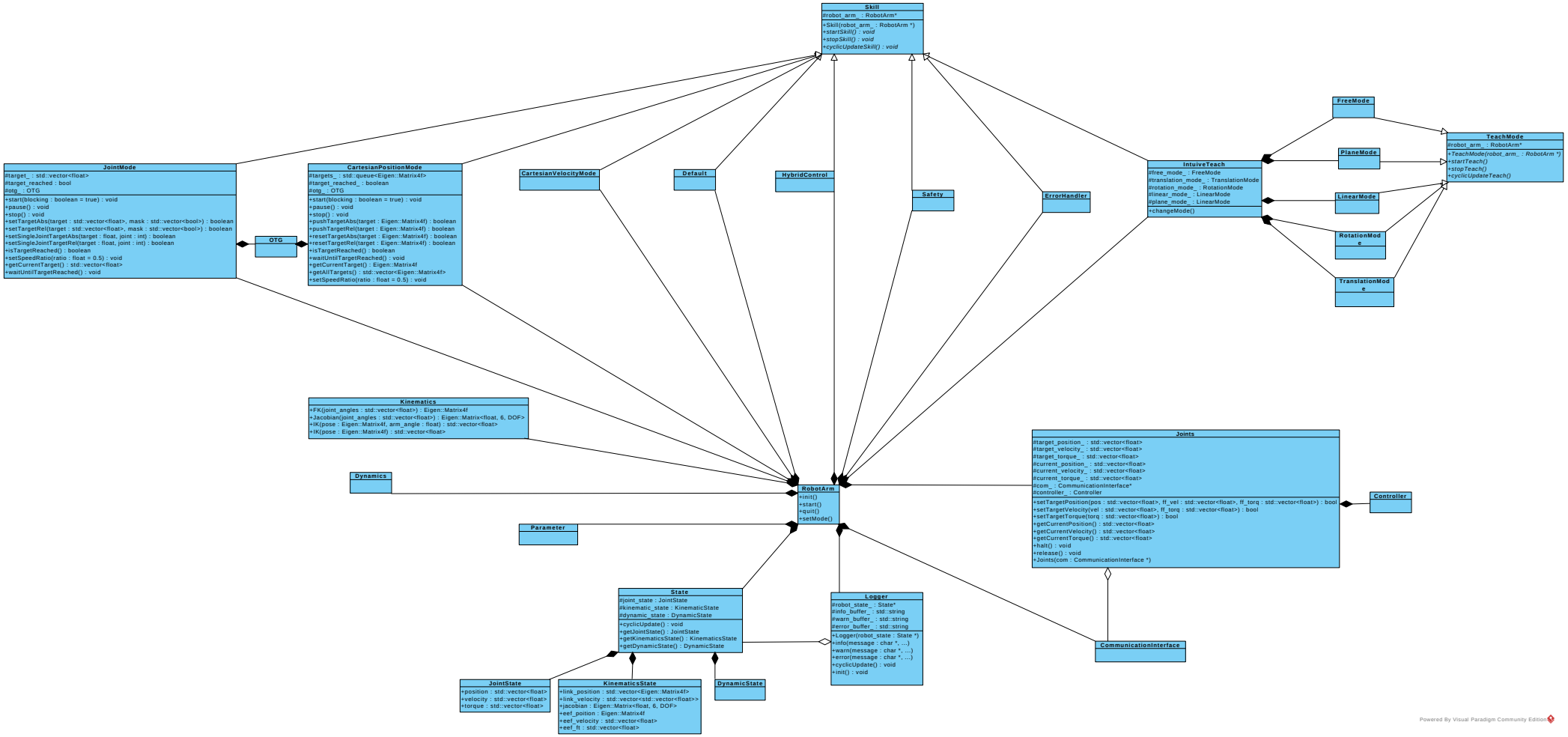
Skill
#robot_arm_ : RobotArm*
+Skill(robot_arm_ : RobotArm *)
+startSkill() : void
+stopSkill() : void
+cyclicUpdateSkill() : void
JointMode
#target_ : std::vector<float>
#target_reached : bool
#otg_ : OTG
+start(blocking : boolean = true) : void
+pause() : void
+stop() : void
+setTargetAbs(target : std::vector<float>, mask : std::vector<bool>) : boolean
+setTargetRel(target : std::vector<float>, mask : std::vector<bool>) : boolean
+setSingleJointTargetAbs(target : float, joint : int) : boolean
+setSingleJointTargetRel(target : float, joint : int) : boolean
+isTargetReached() : boolean
+setSpeedRatio(ratio : float = 0.5) : void
+getCurrentTarget() : std::vector<float>
+waitUntilTargetReached() : void
OTG
CartesianPositionMode
#targets_ : std::queue<Eigen::Matrix4f>
#target_reached_ : boolean
#otg_ : OTG
+start(blocking : boolean = true) : void
+pause() : void
+stop() : void
+pushTargetAbs(target : Eigen::Matrix4f) : boolean
+pushTargetRel(target : Eigen::Matrix4f) : boolean
+resetTargetAbs(target : Eigen::Matrix4f) : boolean
+resetTargetRel(target : Eigen::Matrix4f) : boolean
+isTargetReached() : boolean
+waitUntilTargetReached() : void
+getCurrentTarget() : Eigen::Matrix4f
+getAllTargets() : std::vector<Eigen::Matrix4f>
+setSpeedRatio(ratio : float = 0.5) : void
CartesianVelocityMode
Default
HybridControl
Safety
ErrorHandler
IntuiveTeach
#free_mode_ : FreeMode
#translation_mode_ : TranslationMode
#rotation_mode_ : RotationMode
#linear_mode_ : LinearMode
#plane_mode_ : LinearMode
+changeMode()
FreeMode
PlaneMode
LinearMode
RotationMode
TranslationMode
TeachMode
#robot_arm_ : RobotArm*
+TeachMode(robot_arm_ : RobotArm *)
+startTeach()
+stopTeach()
+cyclicUpdateTeach()
Kinematics
+FK(joint_angles : std::vector<float>) : Eigen::Matrix4f
+Jacobian(joint_angles : std::vector<float>) : Eigen::Matrix<float, 6, DOF>
+IK(pose : Eigen::Matrix4f, arm_angle : float) : std::vector<float>
+IK(pose : Eigen::Matrix4f) : std::vector<float>
Dynamics
RobotArm
+init()
+start()
+quit()
+setMode()
Parameter
Joints
#target_position_ : std::vector<float>
#target_velocity_ : std::vector<float>
#target_torque_ : std::vector<float>
#current_position_ : std::vector<float>
#current_velocity_ : std::vector<float>
#current_torque_ : std::vector<float>
#com_ : CommunicationInterface*
#controller_ : Controller
+setTargetPosition(pos : std::vector<float>, ff_vel : std::vector<float>, ff_torq : std::vector<float>) : bool
+setTargetVelocity(vel : std::vector<float>, ff_torq : std::vector<float>) : bool
+setTargetTorque(torq : std::vector<float>) : bool
+getCurrentPosition() : std::vector<float>
+getCurrentVelocity() : std::vector<float>
+getCurrentTorque() : std::vector<float>
+halt() : void
+release() : void
+Joints(com : CommunicationInterface *)
Controller
State
#joint_state : JointState
#kinematic_state : KinematicState
#dynamic_state : DynamicState
+cyclicUpdate() : void
+getJointState() : JointState
+getKinematicsState() : KinematicsState
+getDynamicState() : DynamicState
Logger
#robot_state_ : State*
#info_buffer_ : std::string
#warn_buffer_ : std::string
#error_buffer_ : std::string
+Logger(robot_state : State *)
+info(message : char *, ...)
+warn(message : char *, ...)
+error(message : char *, ...)
+cyclicUpdate() : void
+init() : void
CommunicationInterface
JointState
+position : std::vector<float>
+velocity : std::vector<float>
+torque : std::vector<float>
KinematicsState
+link_position : std::vector<Eigen::Matrix4f>
+link_velocity : std::vector<std::vector<float>>
+jacobian : Eigen::Matrix<float, 6, DOF>
+eef_poition : Eigen::Matrix4f
+eef_velocity : std::vector<float>
+eef_ft : std::vector<float>
DynamicState
Powered By Visual Paradigm Community Edition