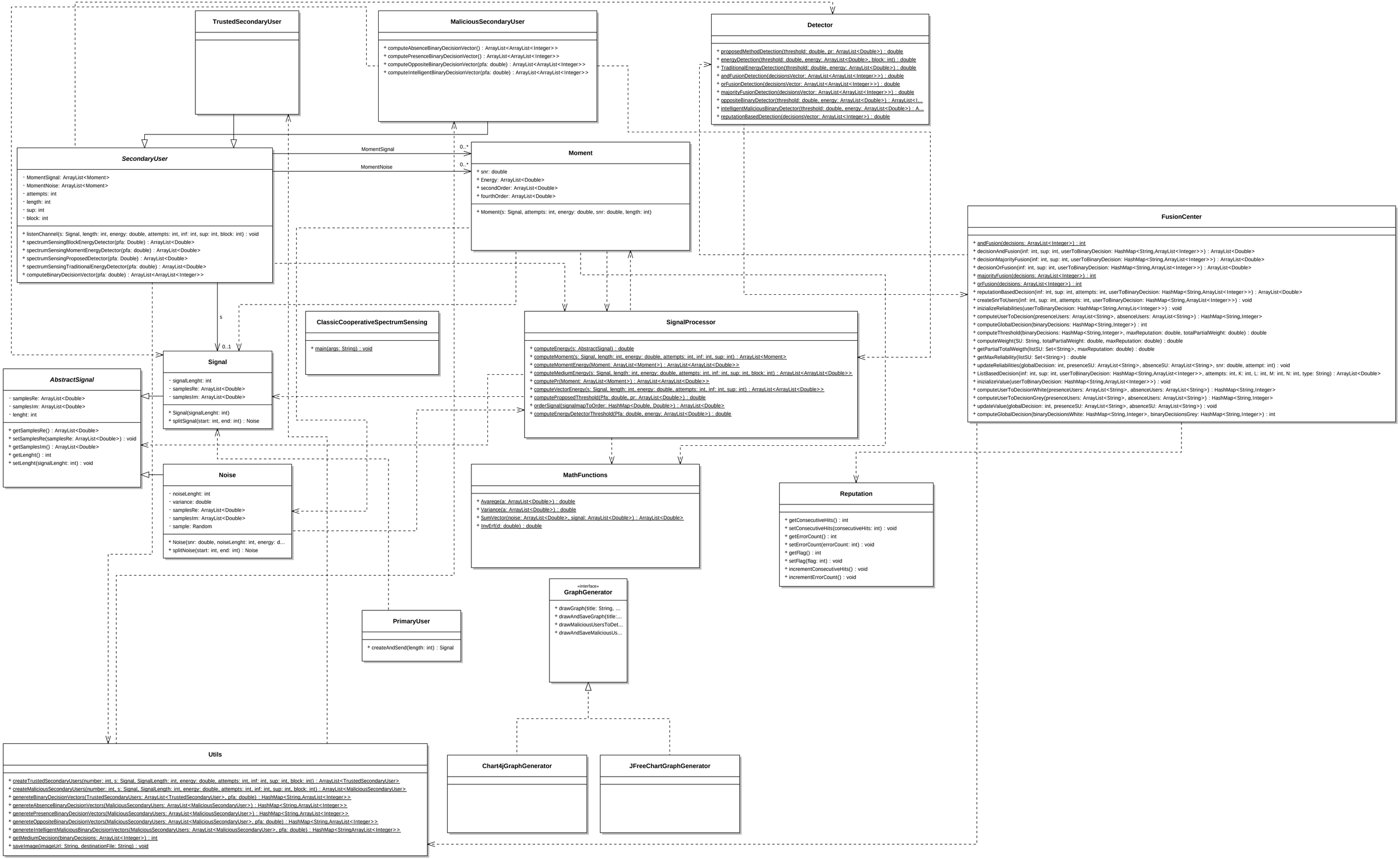
TrustedSecondaryUser
MaliciousSecondaryUser
Detector
+ proposedMethodDetection(threshold: double, pr: ArrayList<Double>) : double
+ energyDetection(threshold: double, energy: ArrayList<Double>, block: int) : double
+ TraditionalEnergyDetection(threshold: double, energy: ArrayList<Double>) : double
+ andFusionDetection(decisionsVector: ArrayList<ArrayList<Integer>>) : double
+ orFusionDetection(decisionsVector: ArrayList<ArrayList<Integer>>) : double
+ majorityFusionDetection(decisionsVector: ArrayList<ArrayList<Integer>>) : double
+ oppositeBinaryDetector(threshold: double, energy: ArrayList<Double>) : ArrayList<I...
+ intelligentMaliciousBinaryDetector(threshold: double, energy: ArrayList<Double>) : A...
+ reputationBasedDetection(decisionsVector: ArrayList<Integer>) : double
SecondaryUser
- MomentSignal: ArrayList<Moment>
- MomentNoise: ArrayList<Moment>
- attempts: int
- length: int
- sup: int
- block: int
+ listenChannel(s: Signal, length: int, energy: double, attempts: int, inf: int, sup: int, block: int) : void
+ spectrumSensingBlockEnergyDetector(pfa: Double) : ArrayList<Double>
+ spectrumSensingMomentEnergyDetector(pfa: double) : ArrayList<Double>
+ spectrumSensingProposedDetector(pfa: Double) : ArrayList<Double>
+ spectrumSensingTraditionalEnergyDetector(pfa: double) : ArrayList<Double>
+ computeBinaryDecisionVector(pfa: double) : ArrayList<ArrayList<Integer>>
MomentSignal
MomentNoise
0..*
0..*
Moment
+ snr: double
+ Energy: ArrayList<Double>
+ secondOrder: ArrayList<Double>
+ fourthOrder: ArrayList<Double>
+ Moment(s: Signal, attempts: int, energy: double, snr: double, length: int)
FusionCenter
+ andFusion(decisions: ArrayList<Integer>) : int
+ decisionAndFusion(inf: int, sup: int, userToBinaryDecision: HashMap<String,ArrayList<Integer>>) : ArrayList<Double>
+ decisionMajorityFusion(inf: int, sup: int, userToBinaryDecision: HashMap<String,ArrayList<Integer>>) : ArrayList<Double>
+ decisionOrFusion(inf: int, sup: int, userToBinaryDecision: HashMap<String,ArrayList<Integer>>) : ArrayList<Double>
+ majorityFusion(decisions: ArrayList<Integer>) : int
+ orFusion(decisions: ArrayList<Integer>) : int
+ reputationBasedDecision(inf: int, sup: int, attempts: int, userToBinaryDecision: HashMap<String,ArrayList<Integer>>) : ArrayList<Double>
+ createSnrToUsers(inf: int, sup: int, attempts: int, userToBinaryDecision: HashMap<String,ArrayList<Integer>>) : void
+ inizializeReliabilities(userToBinaryDecision: HashMap<String,ArrayLis<Integer>>) : void
+ computeUserToDecision(presenceUsers: ArrayList<String>, absenceUsers: ArrayList<String>) : HashMap<String,Integer>
+ computeGlobalDecision(binaryDecisions: HashMap<String,Integer>) : int
+ computeThreshold(binaryDecisions: HashMap<String,Integer>, maxReputation: double, totalPartialWeight: double) : double
+ computeWeight(SU: String, totalPartialWeight: double, maxReputation: double) : double
+ getPartialTotalWeigth(listSU: Set<String>, maxReputation: double) : double
+ getMaxReliability(listSU: Set<String>) : double
+ updateReliabilities(globalDecision: int, presenceSU: ArrayList<String>, absenceSU: ArrayList<String>, snr: double, attempt: int) : void
+ ListBasedDecision(inf: int, sup: int, userToBinaryDecision: HashMap<String,ArrayList<Integer>>, attempts: int, K: int, L: int, M: int, N: int, type: String) : ArrayList<Double>
+ inizializeValue(userToBinaryDecision: HashMap<String,ArrayList<Integer>>) : void
+ computeUserToDecisionWhite(presenceUsers: ArrayList<String>, absenceUsers: ArrayList<String>) : HashMap<String,Integer>
+ computeUserToDecisionGrey(presenceUsers: ArrayList<String>, absenceUsers: ArrayList<String>) : HashMap<String,Integer>
+ updateValue(globalDecision: int, presenceSU: ArrayList<String>, absenceSU: ArrayList<String>) : void
+ computeGlobalDecision(binaryDecisionsWhite: HashMap<String,Integer>, binaryDecisionsGrey: HashMap<String,Integer>) : int
s
0..1
ClassicCooperativeSpectrumSensing
+ main(args: String) : void
SignalProcessor
+ computeEnergy(s: AbstractSignal) : double
+ computeMoment(s: Signal, length: int, energy: double, attempts: int, inf: int, sup: int) : ArrayList<Moment>
+ computeMomentEnergy(Moment: ArrayList<Moment>) : ArrayList<ArrayList<Double>>
+ computeMediumEnergy(s: Signal, length: int, energy: double, attempts: int, inf: int, sup: int, block: int) : ArrayList<ArrayList<Double>>
+ computePr(Moment: ArrayList<Moment>) : ArrayList<ArrayList<Double>>
+ computeVectorEnergy(s: Signal, length: int, energy: double, attempts: int, inf: int, sup: int) : ArrayList<ArrayList<Double>>
+ computeProposedThreshold(Pfa: double, pr: ArrayList<Double>) : double
+ orderSignal(signalmapToOrder: HashMap<Double, Double>) : ArrayList<Double>
+ computeEnergyDetectorThreshold(Pfa: double, energy: ArrayList<Double>) : double
AbstractSignal
- samplesRe: ArrayList<Double>
- samplesIm: ArrayList<Double>
- lenght: int
+ getSamplesRe() : ArrayList<Double>
+ setSamplesRe(samplesRe: ArrayList<Double>) : void
+ getSamplesIm() : ArrayList<Double>
+ getLenght() : int
+ setLenght(signalLenght: int) : void
Signal
- signalLenght: int
- samplesRe: ArrayList<Double>
- samplesIm: ArrayList<Double>
+ Signal(signalLenght: int)
+ splitSignal(start: int, end: int) : Noise
Noise
- noiseLenght: int
- variance: double
- samplesRe: ArrayList<Double>
- samplesIm: ArrayList<Double>
- sample: Random
+ Noise(snr: double, noiseLenght: int, energy: d...
+ splitNoise(start: int, end: int) : Noise
MathFunctions
+ Avarege(a: ArrayList<Double>) : double
+ Variance(a: ArrayList<Double>) : double
+ SumVector(noise: ArrayList<Double>, signal: ArrayList<Double>) : ArrayList<Double>
+ InvErf(d: double) : double
Reputation
+ getConsecutiveHits() : int
+ setConsecutiveHits(consecutiveHits: int) : void
+ getErrorCount() : int
+ setErrorCount(errorCount: int) : void
+ getFlag() : int
+ setFlag(flag: int) : void
+ incrementConsecutiveHits() : void
+ incrementErrorCount() : void
PrimaryUser
+ createAndSend(length: int) : Signal
«interface»
GraphGenerator
+ drawGraph(title: String, ...
+ drawAndSaveGraph(title:...
+ drawMaliciousUsersToDet...
+ drawAndSaveMaliciousUs...
Utils
+ createTrustedSecondaryUsers(number: int, s: Signal, SignalLength: int, energy: double, attempts: int, inf: int, sup: int, block: int) : ArrayList<TrustedSecondaryUser>
+ createMaliciousSecondaryUsers(number: int, s: Signal, SignalLength: int, energy: double, attempts: int, inf: int, sup: int, block: int) : ArrayList<MaliciousSecondaryUser>
+ genereteBinaryDecisionVectors(TrustedSecondaryUsers: ArrayList<TrustedSecondaryUser>, pfa: double) : HashMap<String,ArrayList<Integer>>
+ genereteAbsenceBinaryDecisionVectors(MaliciousSecondaryUsers: ArrayList<MaliciousSecondaryUser>) : HashMap<String,ArrayList<Integer>>
+ generetePresenceBinaryDecisionVectors(MaliciousSecondaryUsers: ArrayList<MaliciousSecondaryUser>) : HashMap<String,ArrayList<Integer>>
+ genereteOppositeBinaryDecisionVectors(MaliciousSecondaryUsers: ArrayList<MaliciousSecondaryUser>, pfa: double) : HashMap<String,ArrayList<Integer>>
+ genereteIntelligentMaliciousBinaryDecisionVectors(MaliciousSecondaryUsers: ArrayList<MaliciousSecondaryUser>, pfa: double) : HashMap<StringArrayList<Integer>>
+ getMediumDecision(binaryDecisions: ArrayList<Integer>) : int
+ saveImage(imageUrl: String, destinationFile: String) : void
Chart4jGraphGenerator
JFreeChartGraphGenerator