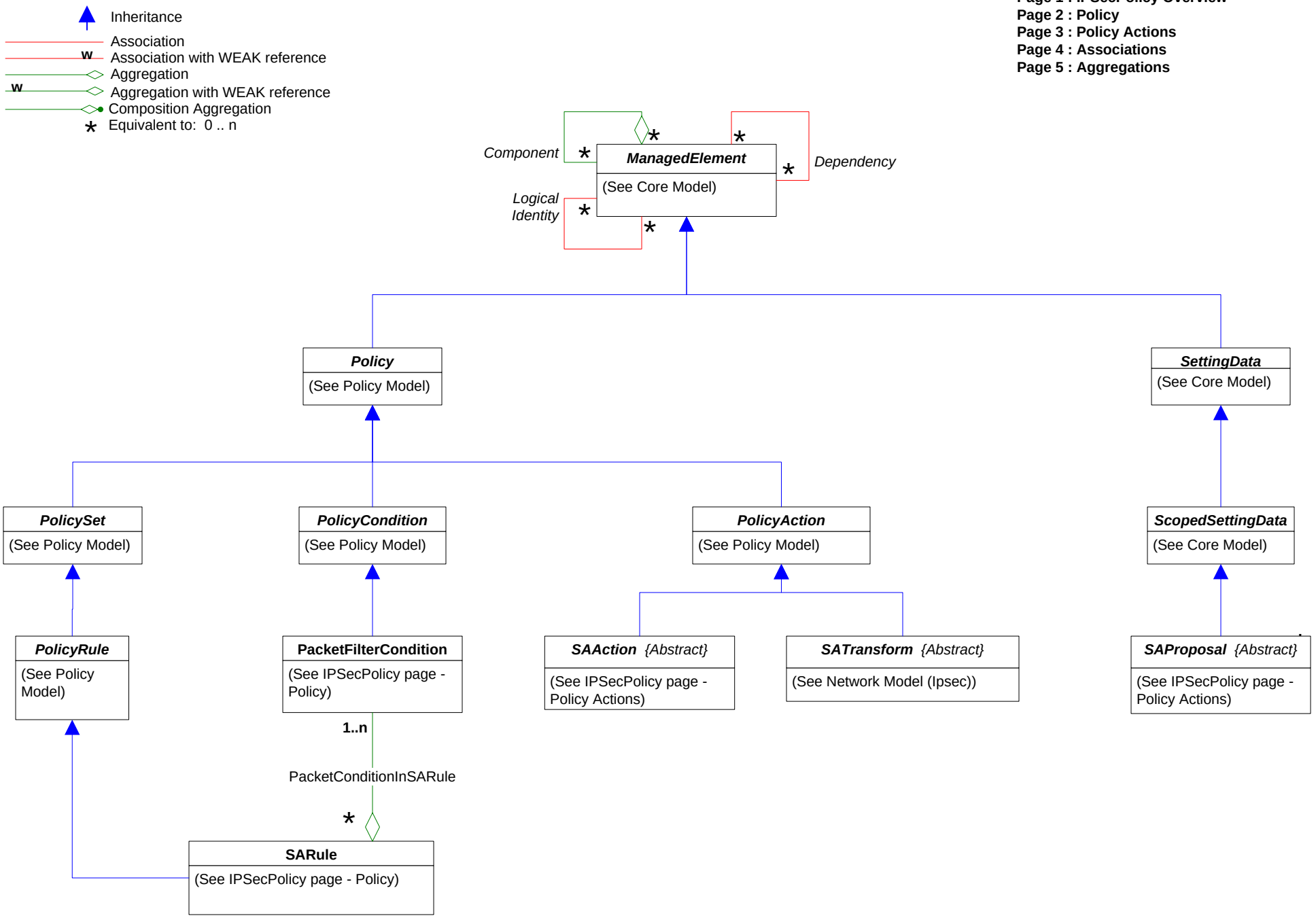
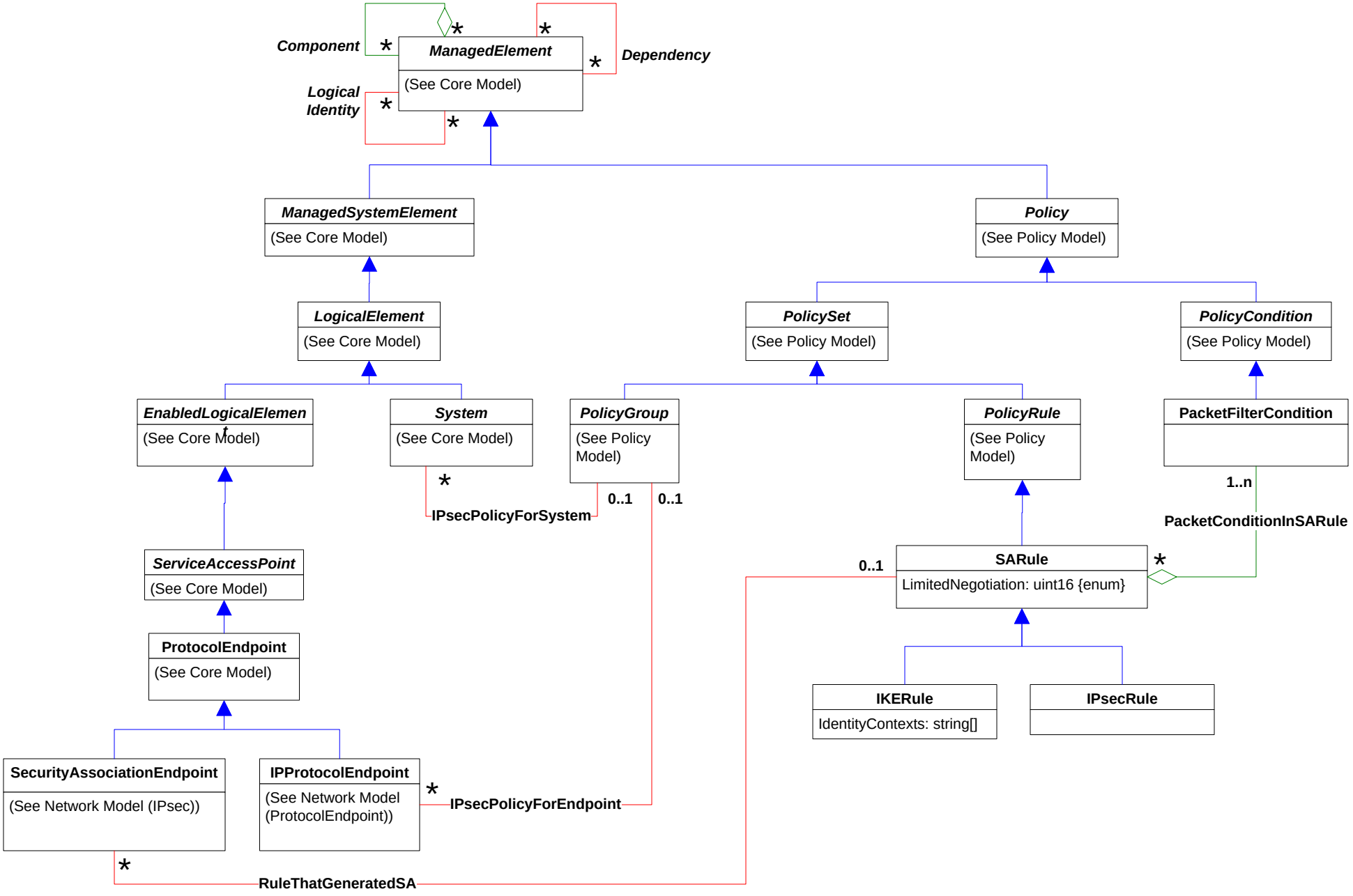
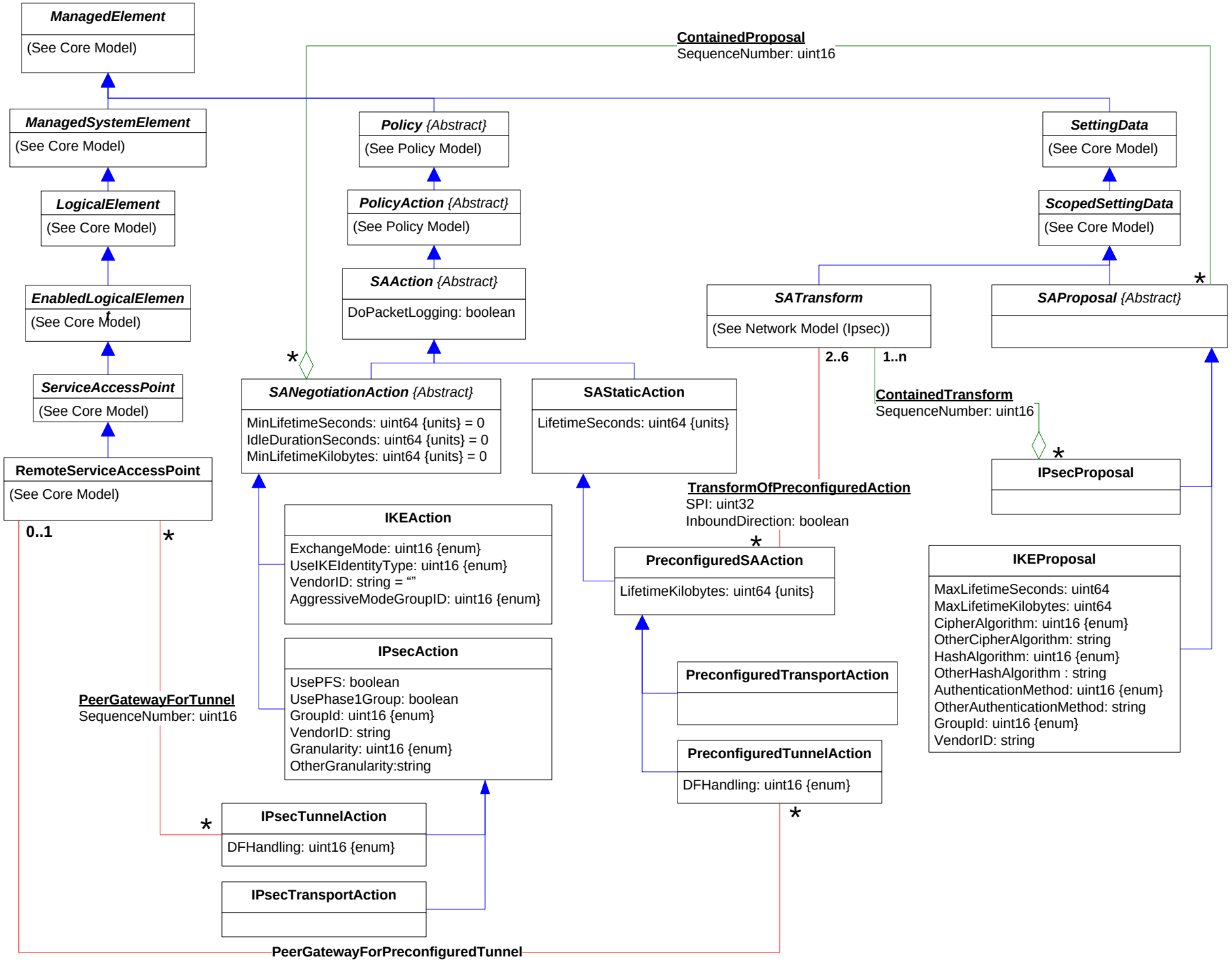
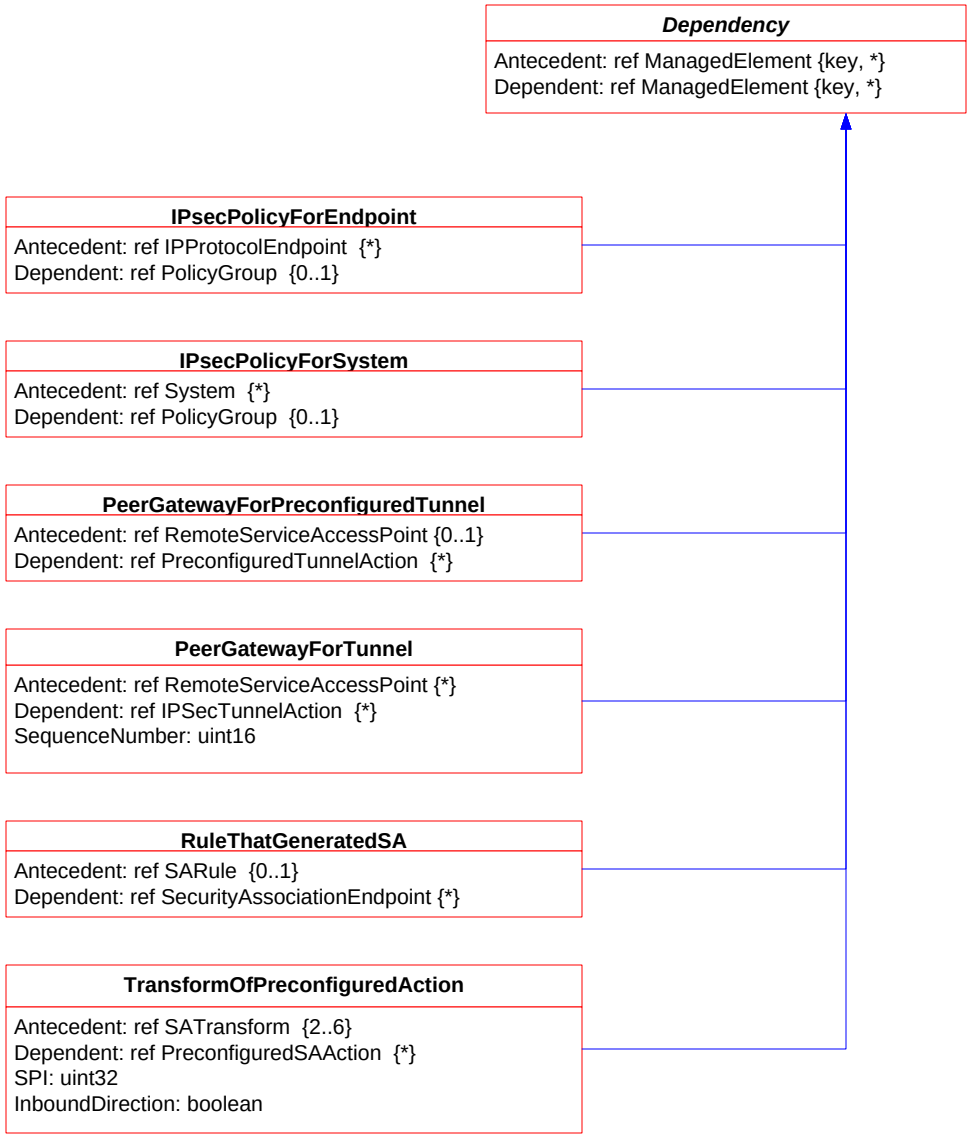
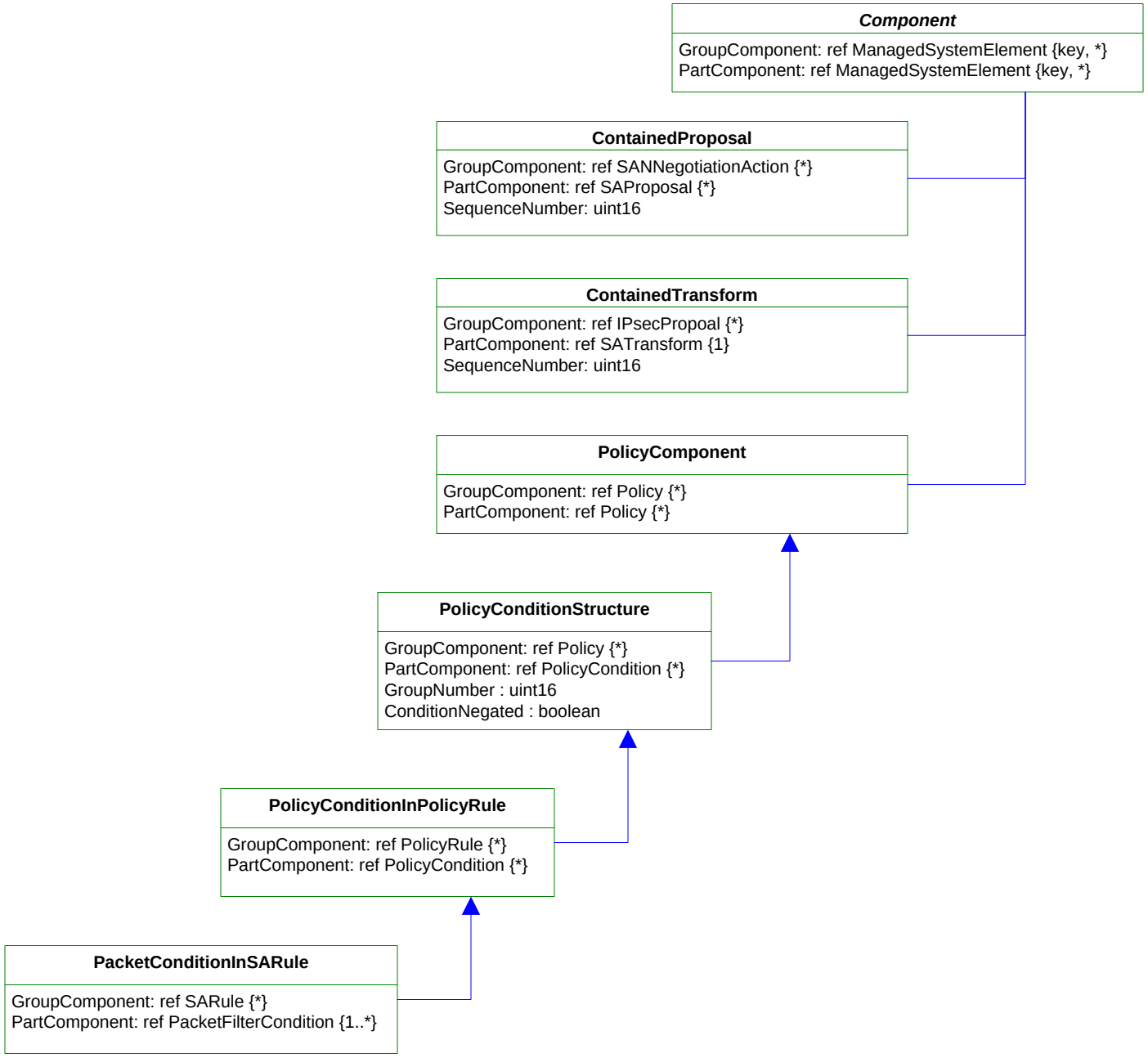
Page 1 : IPSecPolicy Overview
Page 2 : Policy
Page 3 : Policy Actions
Page 4 : Associations
Page 5 : Aggregations




Component
*
*
ManagedElement
(See Core Model)
*
*
Dependency
Logical Identity
*
*
ManagedSystemElement
(See Core Model)
Policy
(See Policy Model)
LogicalElement
(See Core Model)
PolicySet
(See Policy Model)
PolicyCondition
(See Policy Model)
EnabledLogicalElemen
(See Core Model)
System
(See Core Model)
PolicyGroup
(See Policy Model)
PolicyRule
(See Policy Model)
PacketFilterCondition
*
IPsecPolicyForSystem
0..1
0..1
1..n
PacketConditionInSARule
ServiceAccessPoint
(See Core Model)
0..1
SARule
LimitedNegotiation: uint16 {enum}
*
ProtocolEndpoint
(See Core Model)
IKERule
IdentityContexts: string[]
IPsecRule
SecurityAssociationEndpoint
(See Network Model (IPsec))
IPProtocolEndpoint
(See Network Model (ProtocolEndpoint))
*
IPsecPolicyForEndpoint
*
RuleThatGeneratedSA


ManagedElement
(See Core Model)
ContainedProposal
SequenceNumber: uint16
ManagedSystemElement
(See Core Model)
Policy {Abstract}
(See Policy Model)
SettingData
(See Core Model)
LogicalElement
(See Core Model)
PolicyAction {Abstract}
(See Policy Model)
ScopedSettingData
(See Core Model)
EnabledLogicalElemen
(See Core Model)
SAAction {Abstract}
DoPacketLogging: boolean
SATransform
(See Network Model (Ipsec))
SAProposal {Abstract}
*
2..6
1..n
ServiceAccessPoint
(See Core Model)
SANegotiationAction {Abstract}
MinLifetimeSeconds: uint64 {units} = 0
IdleDurationSeconds: uint64 {units} = 0
MinLifetimeKilobytes: uint64 {units} = 0
SAStaticAction
LifetimeSeconds: uint64 {units}
ContainedTransform
SequenceNumber: uint16
IPsecProposal
RemoteServiceAccessPoint
(See Core Model)
TransformOfPreconfiguredAction
SPI: uint32
InboundDirection: boolean
IKEAction
ExchangeMode: uint16 {enum}
UseIKEIdentityType: uint16 {enum}
VendorID: string = “”
AggressiveModeGroupID: uint16 {enum}
PreconfiguredSAAction
LifetimeKilobytes: uint64 {units}
IKEProposal
MaxLifetimeSeconds: uint64
MaxLifetimeKilobytes: uint64
CipherAlgorithm: uint16 {enum}
OtherCipherAlgorithm: string
HashAlgorithm: uint16 {enum}
OtherHashAlgorithm : string
AuthenticationMethod: uint16 {enum}
OtherAuthenticationMethod: string
GroupId: uint16 {enum}
VendorID: string
0..1
IPsecAction
UsePFS: boolean
UsePhase1Group: boolean
GroupId: uint16 {enum}
VendorID: string
Granularity: uint16 {enum}
OtherGranularity:string
PreconfiguredTransportAction
PeerGatewayForTunnel
SequenceNumber: uint16
PreconfiguredTunnelAction
DFHandling: uint16 {enum}
IPsecTunnelAction
DFHandling: uint16 {enum}
IPsecTransportAction
PeerGatewayForPreconfiguredTunnel


Dependency
Antecedent: ref ManagedElement {key, *}
Dependent: ref ManagedElement {key, *}
IPsecPolicyForEndpoint
Antecedent: ref IPProtocolEndpoint {*}
Dependent: ref PolicyGroup {0..1}
IPsecPolicyForSystem
Antecedent: ref System {*}
Dependent: ref PolicyGroup {0..1}
PeerGatewayForPreconfiguredTunnel
Antecedent: ref RemoteServiceAccessPoint {0..1}
Dependent: ref PreconfiguredTunnelAction {*}
PeerGatewayForTunnel
Antecedent: ref RemoteServiceAccessPoint {*}
Dependent: ref IPSecTunnelAction {*}
SequenceNumber: uint16
RuleThatGeneratedSA
Antecedent: ref SARule {0..1}
Dependent: ref SecurityAssociationEndpoint {*}
TransformOfPreconfiguredAction
Antecedent: ref SATransform {2..6}
Dependent: ref PreconfiguredSAAction {*}
SPI: uint32
InboundDirection: boolean


Component
GroupComponent: ref ManagedSystemElement {key, *}
PartComponent: ref ManagedSystemElement {key, *}
ContainedProposal
GroupComponent: ref SANNegotiationAction {*}
PartComponent: ref SAProposal {*}
SequenceNumber: uint16
ContainedTransform
GroupComponent: ref IPsecPropoal {*}
PartComponent: ref SATransform {1}
SequenceNumber: uint16
PolicyComponent
GroupComponent: ref Policy {*}
PartComponent: ref Policy {*}
PolicyConditionStructure
GroupComponent: ref Policy {*}
PartComponent: ref PolicyCondition {*}
GroupNumber : uint16
ConditionNegated : boolean
PolicyConditionInPolicyRule
GroupComponent: ref PolicyRule {*}
PartComponent: ref PolicyCondition {*}
PacketConditionInSARule
GroupComponent: ref SARule {*}
PartComponent: ref PacketFilterCondition {1..*}