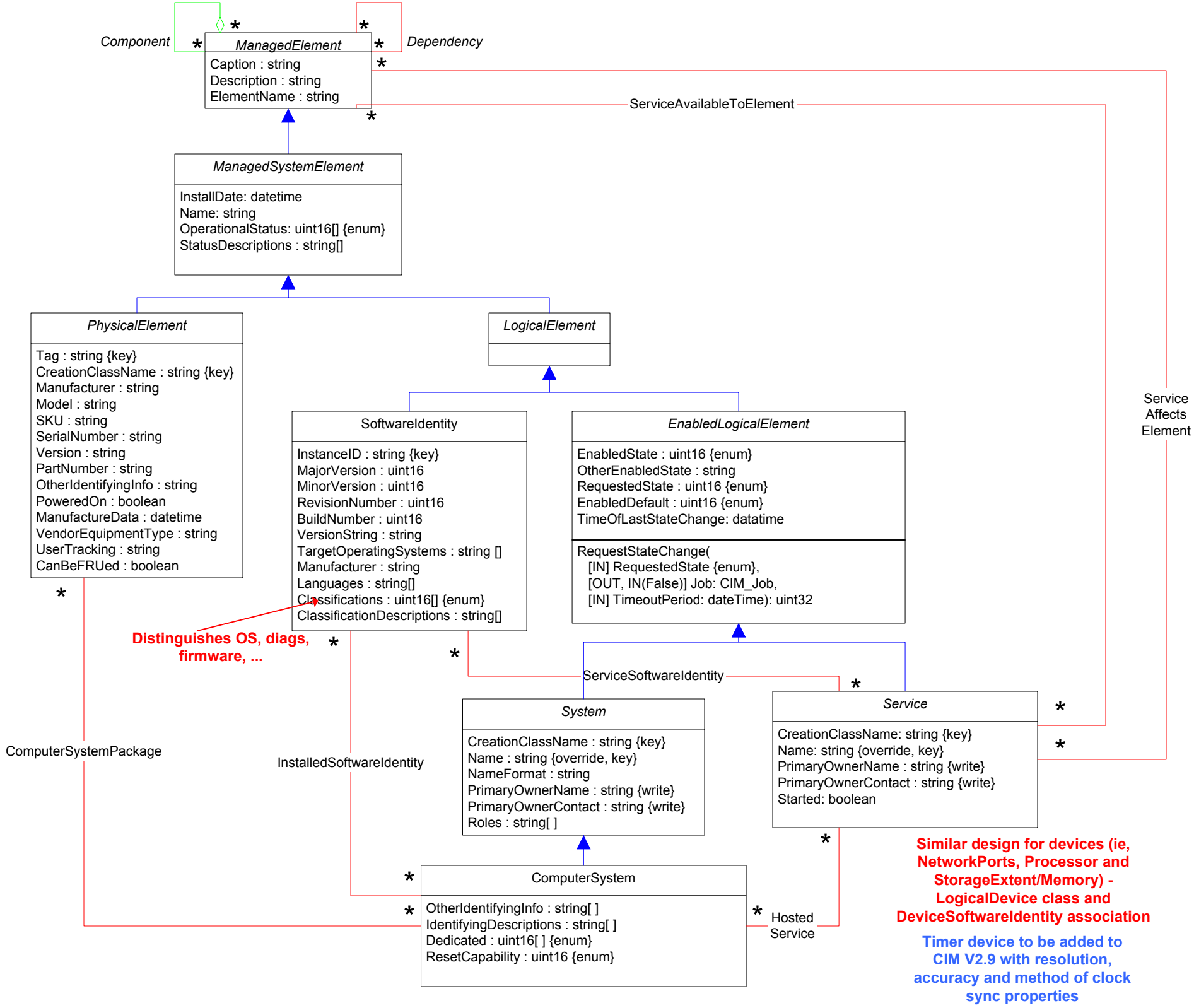
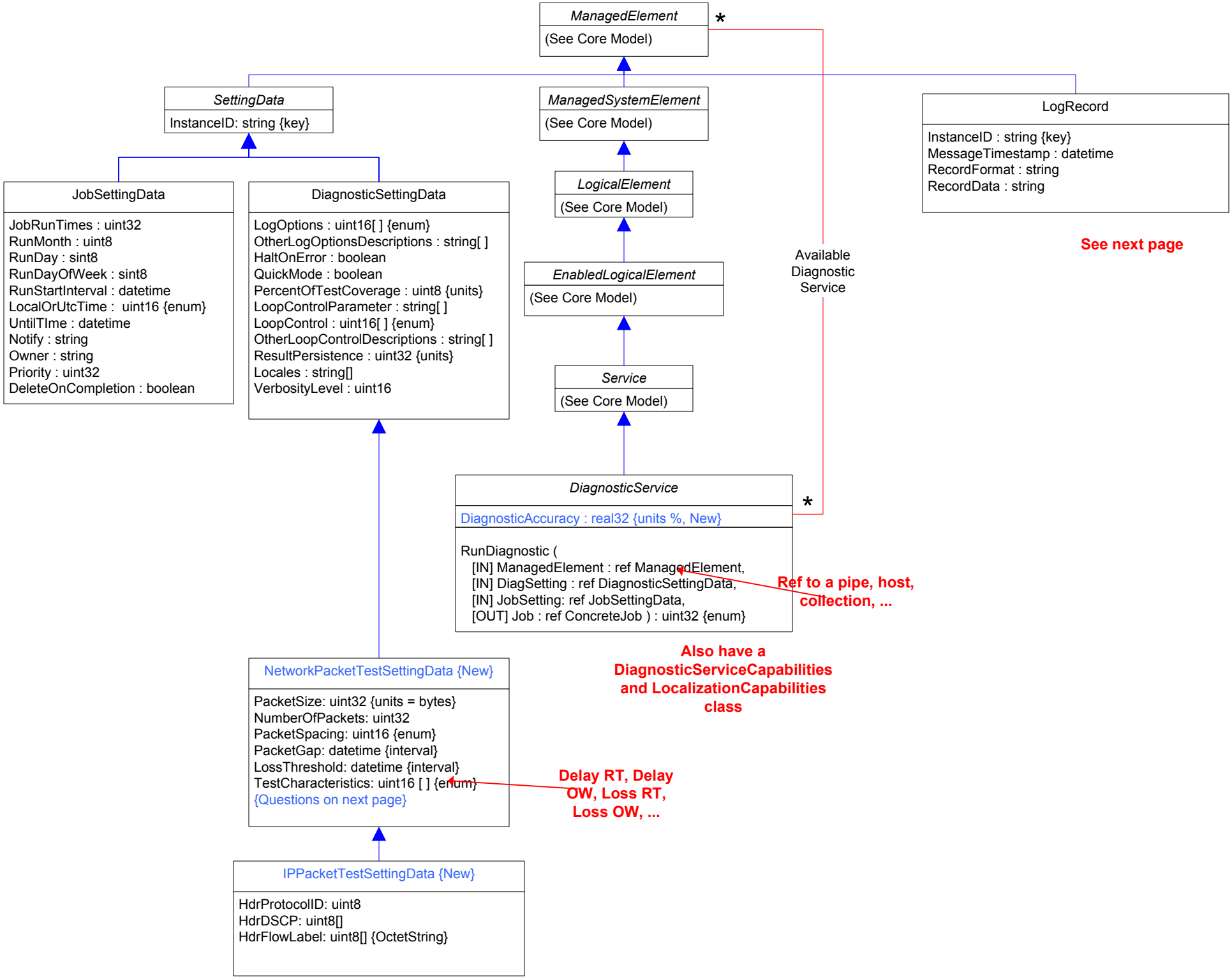
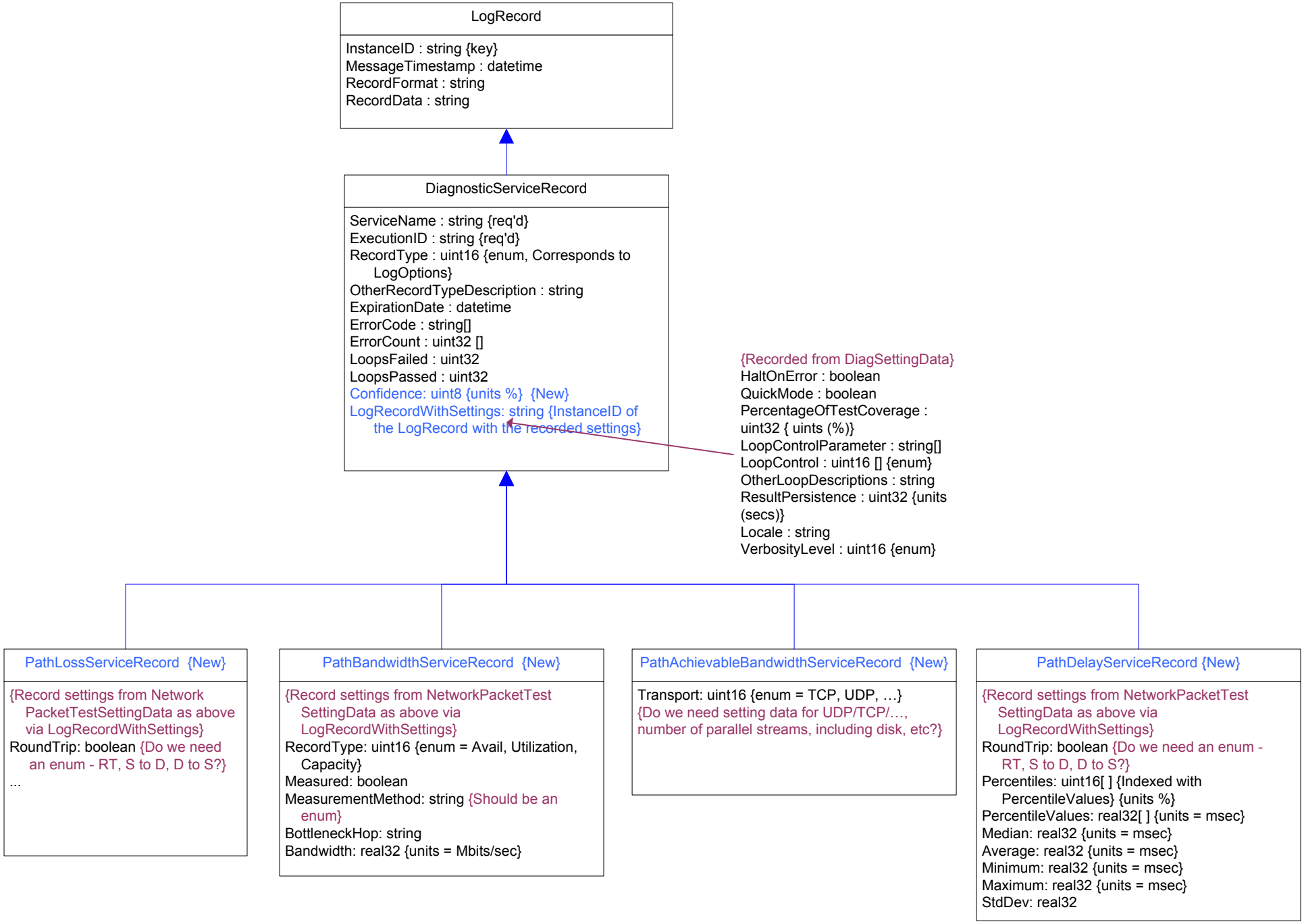

Component
ManagedElement
Caption : string
Description : string
ElementName : string
Dependency
ServiceAvailableToElement
ManagedSystemElement
InstallDate: datetime
Name: string
OperationalStatus: uint16[] {enum}
StatusDescriptions : string[]
PhysicalElement
Tag : string {key}
CreationClassName : string {key}
Manufacturer : string
Model : string
SKU : string
SerialNumber : string
Version : string
PartNumber : string
OtherIdentifyingInfo : string
PoweredOn : boolean
ManufactureData : datetime
VendorEquipmentType : string
UserTracking : string
CanBeFRUed : boolean
LogicalElement
SoftwareIdentity
InstanceID : string {key}
MajorVersion : uint16
MinorVersion : uint16
RevisionNumber : uint16
BuildNumber : uint16
VersionString : string
TargetOperatingSystems : string []
Manufacturer : string
Languages : string[]
Classifications : uint16[] {enum}
ClassificationDescriptions : string[]
EnabledLogicalElement
EnabledState : uint16 {enum}
OtherEnabledState : string
RequestedState : uint16 {enum}
EnabledDefault : uint16 {enum}
TimeOfLastStateChange: datatime
RequestStateChange(
[IN] RequestedState {enum},
[OUT, IN(False)] Job: CIM_Job,
[IN] TimeoutPeriod: dateTime): uint32
Service Affects Element
Distinguishes OS, diags, firmware, ...
ServiceSoftwareIdentity
System
CreationClassName : string {key}
Name : string {override, key}
NameFormat : string
PrimaryOwnerName : string {write}
PrimaryOwnerContact : string {write}
Roles : string[ ]
Service
CreationClassName: string {key}
Name: string {override, key}
PrimaryOwnerName : string {write}
PrimaryOwnerContact : string {write}
Started: boolean
ComputerSystemPackage
InstalledSoftwareIdentity
ComputerSystem
OtherIdentifyingInfo : string[ ]
IdentifyingDescriptions : string[ ]
Dedicated : uint16[ ] {enum}
ResetCapability : uint16 {enum}
Hosted Service
Similar design for devices (ie, NetworkPorts, Processor and StorageExtent/Memory) - LogicalDevice class and DeviceSoftwareIdentity association
Timer device to be added to CIM V2.9 with resolution, accuracy and method of clock sync properties


ManagedElement
(See Core Model)
*
SettingData
InstanceID: string {key}
ManagedSystemElement
(See Core Model)
LogRecord
InstanceID : string {key}
MessageTimestamp : datetime
RecordFormat : string
RecordData : string
See next page
JobSettingData
JobRunTimes : uint32
RunMonth : uint8
RunDay : sint8
RunDayOfWeek : sint8
RunStartInterval : datetime
LocalOrUtcTime : uint16 {enum}
UntilTIme : datetime
Notify : string
Owner : string
Priority : uint32
DeleteOnCompletion : boolean
DiagnosticSettingData
LogOptions : uint16[ ] {enum}
OtherLogOptionsDescriptions : string[ ]
HaltOnError : boolean
QuickMode : boolean
PercentOfTestCoverage : uint8 {units}
LoopControlParameter : string[ ]
LoopControl : uint16[ ] {enum}
OtherLoopControlDescriptions : string[ ]
ResultPersistence : uint32 {units}
Locales : string[]
VerbosityLevel : uint16
LogicalElement
(See Core Model)
EnabledLogicalElement
(See Core Model)
Service
(See Core Model)
Available Diagnostic Service
DiagnosticService
*
DiagnosticAccuracy : real32 {units %, New}
RunDiagnostic (
[IN] ManagedElement : ref ManagedElement,
[IN] DiagSetting : ref DiagnosticSettingData,
[IN] JobSetting: ref JobSettingData,
[OUT] Job : ref ConcreteJob) : uint32 {enum}
Ref to a pipe, host, collection, ...
Also have a DiagnosticServiceCapabilities and LocalizationCapabilities class
NetworkPacketTestSettingData {New}
PacketSize: uint32 {units = bytes}
NumberOfPackets: uint32
PacketSpacing: uint16 {enum}
PacketGap: datetime {interval}
LossThreshold: datetime {interval}
TestCharacteristics: uint16 [ ] {enum}
{Questions on next page}
Delay RT, Delay OW, Loss RT, Loss OW, ...
IPPacketTestSettingData {New}
HdrProtocolID: uint8
HdrDSCP: uint8[]
HdrFlowLabel: uint8[] {OctetString}


LogRecord
InstanceID : string {key}
MessageTimestamp : datetime
RecordFormat : string
RecordData : string
DiagnosticServiceRecord
ServiceName : string {req'd}
ExecutionID : string {req'd}
RecordType : uint16 {enum, Corresponds to LogOptions}
OtherRecordTypeDescription : string
ExpirationDate : datetime
ErrorCode : string[]
ErrorCount : uint32 []
LoopsFailed : uint32
LoopsPassed : uint32
Confidence: uint8 {units %} {New}
LogRecordWithSettings: string {InstanceID of the LogRecord with the recorded settings}
{Recorded from DiagSettingData}
HaltOnError : boolean
QuickMode : boolean
PercentageOfTestCoverage : uint32 { uints (%)}
LoopControlParameter : string[]
LoopControl : uint16 [] {enum}
OtherLoopDescriptions : string
ResultPersistence : uint32 {units (secs)}
Locale : string
VerbosityLevel : uint16 {enum}
PathLossServiceRecord {New}
{Record settings from Network PacketTestSettingData as above via LogRecordWithSettings}
RoundTrip: boolean {Do we need an enum - RT, S to D, D to S?}
...
PathBandwidthServiceRecord {New}
{Record settings from NetworkPacketTest SettingData as above via LogRecordWithSettings}
RecordType: uint16 {enum = Avail, Utilization, Capacity}
Measured: boolean
MeasurementMethod: string {Should be an enum}
BottleneckHop: string
Bandwidth: real32 {units = Mbits/sec}
PathAchievableBandwidthServiceRecord {New}
Transport: uint16 {enum = TCP, UDP, …}
{Do we need setting data for UDP/TCP/…, number of parallel streams, including disk, etc?}
PathDelayServiceRecord {New}
{Record settings from NetworkPacketTest SettingData as above via LogRecordWithSettings}
RoundTrip: boolean {Do we need an enum - RT, S to D, D to S?}
Percentiles: uint16[ ] {Indexed with PercentileValues} {units %}
PercentileValues: real32[ ] {units = msec}
Median: real32 {units = msec}
Average: real32 {units = msec}
Minimum: real32 {units = msec}
Maximum: real32 {units = msec}
StdDev: real32