

```

1 model MTS
2
3 -- classes
4
5 class Hospital
6 attributes
7   name : String
8   phone : String
9   address : String
10 end
11
12 class Patient
13 attributes
14   firstName : String
15   lastName : String
16   id : String
17   gender : Gender
18   age : Integer
19   obtain_patient_info : Boolean
20   examined_airway : Boolean
21   examined_breathing : Boolean
22   checked_patient_for_shock : Boolean
23   pain_measured : Boolean
24   cardiac_pain_examined : Boolean
25   breathing_rate_measured : Boolean
26   pulse_checked : Boolean
27   pleuritic_pain_examined : Boolean
28   checked_for_persistent_vomiting : Boolean
29   examine_patients_cardiac_history : Boolean
30   pain_remeasured : Boolean
31   vomiting_checked : Boolean
32   mild_pain_examined : Boolean
33   problems_checked : Boolean
34   resuscitation_in_action : Boolean
35   patient_stabilized : Boolean
36   patient_supervised : Boolean
37   patient_reassessed : Boolean
38 end
39
40 class TreatedPatient < Patient
41 attributes
42   airwayCompromise : Boolean
43   inadequateBreathing : Boolean
44   shock : Boolean
45   severePain : Boolean
46   moderatePain : Boolean
47   minorPain : Boolean
48   cardiacPain : Boolean
49   acutelyShortBreath : Boolean
50   normalBreathing : Boolean
51   abnormalPulse : Boolean
52   normalPulse : Boolean
53   pleuriticPain : Boolean
54   persistentVomiting : Boolean
55   recentVomiting : Boolean
56   significantCardiacHistory : Boolean
57   recentMildPain : Boolean
58   recentProblem : Boolean
59 end
60
61 class Severity
62 attributes
63   severity: SeverityLevel
64 operations
65   examineAirway(p : TreatedPatient)
66   checkForAdequateBreathing(p : TreatedPatient)
67   checkForShock(p : TreatedPatient)
68   measurePain(p : TreatedPatient)
69   examineCardiacPain(p : TreatedPatient)
70   measureBreathingRate(p : TreatedPatient)
71   checkPulse(p : TreatedPatient)
72   examinePleuriticPain(p : TreatedPatient)
73   checkForPersistentVomiting(p : TreatedPatient)
74   examineForCardiacHistory(p : TreatedPatient)
75   reMeasureForPain(p : TreatedPatient)
76   checkVomiting(p : TreatedPatient)
77   examineForMildPain(p : TreatedPatient)
78   checkForProblems(p : TreatedPatient)
79 end

```

```

80
81 class Immediate
82 operations
83   resuscitationRequired(p : TreatedPatient)
84   continuousObservation(p : TreatedPatient)
85 end
86
87 class VeryUrgent
88 operations
89   stabilize(p : TreatedPatient)
90   monitor(p : TreatedPatient)
91 end
92
93 class Urgent
94 operations
95   supervision(p : TreatedPatient)
96 end
97
98 class Standard
99 operations
100   reassess(p : TreatedPatient)
101 end
102
103 class NonUrgent
104 operations
105   reassess(p : TreatedPatient)
106   canWait(p : TreatedPatient)
107 end
108
109 -- associations
110
111 composition ConsistOf between
112   Hospital[1] role hospital
113   Patient[1..*] role patient
114 end
115
116 association WithTreatment between
117   Patient[1] role patient
118   TreatedPatient[1] role treatedPatient
119 end
120
121 association BasedOnSeverityLevel between
122   TreatedPatient[1] role patientTreatment
123   Severity[1] role severityLevel
124 end
125
126 association ImmediateSeverity between
127   Severity[1] role severityLevel
128   Immediate[1] role immediateCare
129 end
130
131 association VeryUrgentSeverity between
132   Severity[1] role severityLevel
133   VeryUrgent[1] role veryUrgentCare
134 end
135
136 association UrgentSeverity between
137   Severity[1] role severityLevel
138   Urgent[1] role urgentCare
139 end
140
141 association StandardSeverity between
142   Severity[1] role severityLevel
143   Standard[1] role standardCare
144 end
145
146 association NonUrgentSeverity between
147   Severity[1] role severityLevel
148   NonUrgent[1] role nonUrgentCare
149 end
150
151 -- enumeration
152
153 enum SeverityLevel {Immediate, VeryUrgent, Urgent, Standard, NonUrgent}
154 enum Gender {Male, Female}
155
156 -- constraints
157
158 constraints

```

```

159
160 context Hospital
161 inv AtLeastOnePatient: self.patient->size() >= 1
162 inv NonEmptyName: self.name <> ' ' or self.name.size > 1
163
164 context p : Patient
165 inv NumberOfPatients: Patient.allInstances()->size() >= 1
166 inv PatientsInHospital: p.hospital->size() = 1
167 inv NonEmptyFirstName: p.firstName <> ' ' or p.firstName.size > 1
168 inv NonEmptyLastName: p.lastName <> ' ' or p.lastName.size > 1
169 inv NonEmptyID: p.id <> ' ' or p.id.size > 1
170 inv UniquePatientID: Patient.allInstances()->forall(p, p2 | p.id <> p2.id)
171
172 context Severity :: examineAirway(p : TreatedPatient)
173 pre: p.obtain_patient_info = true
174 post: p.examinated_airway = true
175
176 context Severity:: checkForAdequateBreathing(p : TreatedPatient)
177 pre: p.examinated_airway = true
178 post: p.examinated_breathing = true
179
180 context Severity:: checkForShock(p : TreatedPatient)
181 pre: p.examinated_airway = true and p.examinated_breathing = true
182 post: p.checked_patient_for_shock = true
183
184 context Severity:: measurePain(p : TreatedPatient)
185 pre: p.checked_patient_for_shock = true
186 post: p.pain_measured = true
187
188 context Severity:: examineCardiacPain(p : TreatedPatient)
189 pre: p.pain_measured = true
190 post: p.cardiac_pain_examinated = true
191
192 context Severity:: measureBreathingRate(p : TreatedPatient)
193 pre: p.pain_measured = true and p.cardiac_pain_examinated = true
194 post: p.breathing_rate_measured = true
195
196 context Severity:: checkPulse(p : TreatedPatient)
197 pre: p.pain_measured = true and p.cardiac_pain_examinated = true and p.breathing_rate_measured = true
198 post: p.pulse_checked = true
199
200 context Severity:: examinePleuriticPain(p : TreatedPatient)
201 pre: p.pulse_checked = true
202 post: p.pleuritic_pain_examinated = true
203
204 context Severity:: checkForPersistentVomiting(p : TreatedPatient)
205 pre: p.pleuritic_pain_examinated = true
206 post: p.checked_for_persistent_vomiting = true
207
208 context Severity:: examineForCardiacHistory(p : TreatedPatient)
209 pre: p.pleuritic_pain_examinated=true and p.checked_for_persistent_vomiting = true
210 post: p.examine_patients_cardiac_history = true
211
212 context Severity:: reMeasureForPain(p : TreatedPatient)
213 pre: p.pleuritic_pain_examinated = true and p.checked_for_persistent_vomiting = true and p.examine_patients_cardiac_history = true
214 post: p.pain_remeasured = true
215
216 context Severity:: checkVomiting(p : TreatedPatient)
217 pre: p.pain_remeasured = true
218 post: p.vomiting_checked = true
219
220 context Severity:: examineForMildPain(p : TreatedPatient)
221 pre: p.vomiting_checked = true
222 post: p.mild_pain_examinated = true
223
224 context Severity:: checkForProblems(p : TreatedPatient)
225 pre: p.vomiting_checked = true and p.mild_pain_examinated = true
226 post: p.problems_checked = true
227
228 context Immediate:: resuscitationRequired(p : TreatedPatient)
229 pre: p.examinated_airway = true
230 pre: p.examinated_breathing = true
231 pre: p.checked_patient_for_shock = true
232 post: p.resuscitation_in_action = true
233
234 context VeryUrgent:: stabilize(p : TreatedPatient)
235 pre: p.pain_measured = true
236 pre: p.cardiac_pain_examinated = true
237 pre: p.breathing_rate_measured = true

```

```
238 pre: p.pulse_checked = true
239 post: p.patient_stabilized = true
240
241 context Urgent:: supervision(p : TreatedPatient)
242 pre: p.pleuritic_pain_examined = true
243 pre: p.checked_for_persistent_vomiting = true
244 pre: p.examine_patients_cardiac_history = true
245 pre: p.pain_remeasured = true
246 post: p.patient_supervised = true
247
248 context Standard:: reassess(p : TreatedPatient)
249 pre: p.vomiting_checked = true
250 pre: p.mild_pain_examined = true
251 pre: p.problems_checked = true
252 post: p.patient_reassessed = true
253
254 context NonUrgent:: reassess(p : TreatedPatient)
255 pre: p.vomiting_checked = true
256 pre: p.mild_pain_examined = true
257 pre: p.problems_checked = true
258 post: p.patient_reassessed = true
259
```

```

1 model ESI
2
3 -- classes
4
5 class Hospital
6 attributes
7   name : String
8   phone : String
9   address : String
10 end
11
12 class Patient
13 attributes
14   firstName : String
15   lastName : String
16   id : String
17   gender : Gender
18   age : Integer
19   obtain_patient_info : Boolean
20   checked_for_immediate_intervention : Boolean
21   situation_assessed : Boolean
22   examined_if_confused : Boolean
23   examined_if_disoriented : Boolean
24   examined_if_lethargic : Boolean
25   checked_if_severe_pain : Boolean
26   checked_if_distressed : Boolean
27   number_resources_assessed : Boolean
28   resources : Integer
29   vitals_examined : Boolean
30 end
31
32 class TreatedPatient < Patient
33 attributes
34   requiresLifeSavingIntervention : Boolean
35   highRiskSituation : Boolean
36   confused : Boolean
37   lethargic : Boolean
38   disoriented : Boolean
39   severePain : Boolean
40   distress : Boolean
41   resourcesRequired : Integer
42   heartRate : Integer
43   respiratoryRate : Integer
44   saO2 : Integer
45 end
46
47 class Severity
48 attributes
49   severity : SeverityLevel
50 operations
51   checkIfInterventionNeeded(p : TreatedPatient)
52   assessSituation(p : TreatedPatient)
53   examine(p : TreatedPatient)
54   checkForPain(p : Patient)
55   checkResourcesNeeded(p : TreatedPatient)
56   measureVitalZones(p : TreatedPatient, resources : Integer)
57 end
58
59 class Immediate
60 operations
61   resuscitationRequired(p : TreatedPatient)
62   continuousObservation(p : TreatedPatient)
63 end
64
65 class VeryUrgent
66 operations
67   stabilize(p : TreatedPatient)
68   monitor(p : TreatedPatient)
69 end
70
71 class Urgent
72 operations
73   supervision(p : TreatedPatient)
74 end
75
76 class Minor
77 operations
78   reassess(p : TreatedPatient, resources : Integer)
79 end

```

```

80
81 class Delayed
82 operations
83   canWait(p : TreatedPatient)
84   reassess(p : TreatedPatient, resources : Integer)
85 end
86
87 -- associations
88
89 composition ConsistsOf between
90   Hospital[1] role hospital
91   Patient[1..*] role patient
92 end
93
94 association WithTreatment between
95   Patient[1] role patient
96   TreatedPatient[1] role treatedPatient
97 end
98
99 association BasedOnSeverityLevel between
100   TreatedPatient[1] role patientTreatment
101   Severity[1] role severityLevel
102 end
103
104 association ImmediateSeverity between
105   Severity[1] role severityLevel
106   Immediate[1] role immediateCare
107 end
108
109 association VeryUrgentSeverity between
110   Severity[1] role severityLevel
111   VeryUrgent[1] role veryUrgentCare
112 end
113
114 association UrgentSeverity between
115   Severity[1] role severityLevel
116   Urgent[1] role urgentCare
117 end
118
119 association MinorSeverity between
120   Severity[1] role severityLevel
121   Minor[1] role minorCare
122 end
123
124 association DelayedSeverity between
125   Severity[1] role severityLevel
126   Delayed[1] role delayedCare
127 end
128
129 -- enumeration
130
131 enum SeverityLevel {Immediate, VeryUrgent, Urgent, Minor, Delayed}
132 enum Gender {Male, Female}
133
134 -- constraints
135
136 constraints
137
138 context Hospital
139   inv AtLeastOnePatient: self.patient->size() >= 1
140   inv NonEmptyName: self.name <> '' or self.name.size > 1
141
142 context p : Patient
143   inv NumberOfPatients: Patient.allInstances()->size() >= 1
144   inv PatientsInHospital: p.hospital->size() = 1
145   inv NonEmptyFirstName: p.firstName <> '' or p.firstName.size > 1
146   inv NonEmptyLastName: p.lastName <> '' or p.lastName.size > 1
147   inv NonEmptyID: p.id <> '' or p.id.size > 1
148   inv UniquePatientID: Patient.allInstances()->forall(p, p2 | p.id <> p2.id)
149
150 context Severity :: checkIfInterventionNeeded(p : TreatedPatient)
151   pre: p.obtain_patient_info = true
152   post: p.checked_for_immediate_intervention = true
153
154 context Severity :: assessSituation(p : TreatedPatient)
155   pre: p.checked_for_immediate_intervention = true
156   post: p.situation_assessed = true
157
158 context Severity :: examine(p : TreatedPatient)

```

```
159 pre: p.situation_assessed = true
160 post: p.examined_if_confused = true or p.examined_if_disoriented = true or p.examined_if_lethargic = true
161
162 context Severity :: checkForPain(p : TreatedPatient)
163 pre: p.situation_assessed = true
164 pre: p.examined_if_confused = true or p.examined_if_disoriented = true or p.examined_if_lethargic = true
165 post: p.checked_if_severe_pain = true or p.checked_if_distressed = true
166
167 context Severity:: checkResourcesNeeded(p : TreatedPatient)
168 pre: p.checked_if_severe_pain = true or p.checked_if_distressed = true
169 post: p.number_resources_assessed = true
170
171 context Severity :: measureVitalZones(p : TreatedPatient, resources: Integer)
172 pre: p.number_resources_assessed = true and resources > 1
173 post: p.vitals_examined = true
174
175 context Immediate :: resuscitationRequired (p : TreatedPatient)
176 pre: p.checked_for_immediate_intervention = true
177
178 context VeryUrgent :: stabilize(p : TreatedPatient)
179 pre: p.situation_assessed = true
180 pre: p.examined_if_confused = true or p.examined_if_disoriented = true or p.examined_if_lethargic = true
181 pre: p.checked_if_severe_pain = true or p.checked_if_distressed = true
182 pre: p.vitals_examined = true
183
184 context Urgent :: supervision(p : TreatedPatient)
185 pre: p.vitals_examined = true
186
187 context Minor :: reassess (p : TreatedPatient, resources: Integer)
188 pre: p.number_resources_assessed = true
189 pre: resources = 1
190
191 context Delayed:: reassess (p : TreatedPatient, resources: Integer)
192 pre: p.number_resources_assessed = true
193 pre: resources = 0
194
```