1. Objetivo

O modelo proposto tem como por finalidade representar atividades onde um grupo de pessoas devem atuar de forma colaborativa com o propósito de resolver um problema. Esse problema pode ser dividido em etapas menores conhecidas como objetivos. Essas pessoas podem se relacionar entre sí, podem se relacionar com os artefatos presentes no meio onde elas atuam. Os artefatos também possuem a capacidade de se relacionar. Cada objetivo é concluído apenas se um ou mais relacionamentos forem realizados. A conclusão do objetivo também é função de certos agentes e artefatos que devem ser presentes.

Outra finalidade do modelo consiste em representar condições que devem ser mantidas ao longo da atividade, se essas condições forem desfeitas - a atividade deve ser encerrada de imediato, caso contrário as pessoas envolvidas nesta manutenção estarão submetidas a risco de morte. Dentro desta abordagem, alguém designado para cumprir com alguma atividade pode cometer erros. Esse modelo tem como por finalidade lidar com os seguintes tipos de erro: não executar algum ação quando essa ação deve ser executada, executar uma ação quando ela não deve ser executada, executar uma ação quando não há condições apropriadas para isso, manipular artefatos de maneira inapropriada ou inadvertida e escolher os artefatos inapropriados para cumprir com uma determinada atividade.

O modelo deve ser capaz de representar as consequências desses erros. Esse modelo está preocupado em representar dois tipos de consequências, essas são: 1 - imediatas que acontece sobre o indivíduo errante, 2 - a consequência manifesta em outro objetivo sobre o mesmo ou outro indivíduo pertencente ao grupo. Essas consequências são efeitos físicos negativos que alguém vêem a sofrer. A intensidade dessas consequências variam desde uma leve lesão a morte.

Outro aspecto deste modelo consiste representar objetivos cujo sucesso advem de certa característica aleatória presente na natureza da atividade. Essa característica aleatória consiste em um evento que possuem uma certa possibilidade de acontecer. Se esse evento acontecer, alguém sofre consequências ruins por conta disto. O modelo deve considerar relações entre erros onde as consequências se manifestam de forma indireta com eventos aleatórios.

2. Exemplo - Estudo de Caso

Sete profissionais de linha viva (profissionais que realizam manutenção em equipamentos elétricos energizados) são designados com o propósito de realizar a substituição de um isolador de pedestal. Os papéis desses desses profissionais são; 1 supervisor, 1 observador e 5 executores. A manutenção de ve ser executada apenas sobre as seguintes condições: céu ensolarado e umidade relativa do ar menor que 70 porcento. Todos os profissionais devem possuir os EPIS necessários: capacete, óculos de sol, roupa isolante e anti-chamas, luvas isolantes e botas isolantes. Os profissionais que entrarão no potencial deverão estar vestidos de roupa condutiva e cabo guarda. As ferramentas necessárias para resolver esse problema são: bastão garra de diametro 64 x 3600 mm, sela de diametro 65 , colar, corda de fibra sintética, carretilha, chave com catraca, bastão universal, soquete adequado, locador de pino, bastão com soquete multiangular. A substituição do isolador de pedestal pode ser escrita nos seguintes subobjetivos:

1. Limpar, secar e testar corda.

2. Instalar Bastão Garra na estrutura com o pedestal a ser substituído.
3. Instalar sela com colar na estrutura
4. Amarrar o bastão na parte superior da estrutura com a corda.
5. Amarrar o olhal do bastão ao cavalo da sela atras de uma corda.
6. Instalar um segundo conjunto bastão e sela no lado oposto da estrutura.
7. Enforcar um estropo de Náilon no corpo do isolador.
8. Colocar a extremidade do estropo no gancho da corda de serviço.
9. Afrouxar os parafusos do conector que prendem a barra ao isolador.
10. Terminar de retirar os parafusos com o bastão com o soquete multiangular.
11. Elevar a barra através da corda que une a sela ao bastão.
12. Apertar o colar através da porca borboleta.
13. Segurar firmemente a corda de serviço.
14. Sacar parafusos da base da coluna.
15. Baixar o isolador ao solo
16. Içar o Isolador
17. Colocar Parafusos na base da coluna.
18. Baixaer a barra para que a mesma apóie no novo isolador.
19. Colocar os parafusos do conector que prende a barra ao novo isolador.
20. Retirar Equipamentos

3. Modelo

3.1. Definição dos Conjuntos

1. $Entity = \{e_1, ..., e_n\}$ - conjunto de todas as Entidades.
2. $Agent = \{ag_1, ..., ag_n\}$ - conjunto dos Agentes.
3. $Artefact = \{at_1, ..., at_n\}$ - conjunto dos Artefatos.
4. $EntityGoal = eg = \{e_n, ..., e_m\}$ - conjunto das Entidades que devem estar presentes para concluir um determinado objetivo g_i.
5. $Relation = \{r_1, ..., r_n\}$ - conjunto dos Relacionamentos.
6. $RelationGoal = rg = \{r_n, ..., r_m\}$ - conjunto dos Relacionamentos que devem estar presentes para concluir um único objetivo g_i.
7. $Role = \{\rho_1, ..., \rho_n\}$ - conjunto dos Papéis.
8. $Goal = \{g_1, ..., g_n\}$ - conjunto dos Objetivos.
9. $GoalRandomic = \{gr_1, ..., gr_n\}$ - conjunto dos Objetivos Randomicos.
10. $GoalPrerequisit = gp = \{g_n, ..., g_m\}$ - conjunto de Objetivos que são pré-requisitos para alcançar um outro objetivo.
11. $Condition = \{c_1, ..., c_n\}$ - conjunto das Condições que devem ser mantidas ao longo da execução de todos os objetivos.
12. $ConditionToGoal = cg = \{c_n, ..., c_m\}$ - conjunto de condições que devem ser mantidas para concluir um único objetivo g_i.
13. $Risk = \{risk_1, ..., risk_n\}$ - conjunto dos Riscos na ocorrência de Eventos Ruins.
14. $Possibility = \{p_1, ..., p_n\}$ - conjunto das possibilidades de Eventos Ruins.
15. $Fatality = \{f_1, ..., f_n\}$ - conjunto das fatalidades que acontecem na existência de um evento ruim.

3.2. Definição das Relações Entre os Conjuntos

1. $Entity \equiv Agent \cup Artefact$
2. $Agent$ e $Artefact$ são disjuntos.
3. $GoalRadomic \subset Goal$
4. $\{gp_1, ..., gp_n\} \subset Goal$
5. $\{cg_1, ..., cg_n\} \subset Condition$
6. $\{eg_1, ..., eg_n\} \subset Entity$
7. $\{rg_1, ..., rg_n\} \subset Relation$

3.3. Definição dos Predicados

1. $relationHas(r_l, e_i, e_k)$ onde $i \neq j$ - Um determinado relacionamento r_l é composto por uma entidade e_i e e_k onde e_i não pode ser igual a e_j.
2. $hasRole(ag_n, \rho_m)$ - Um determinado agente ag_n tem um determinado papel ρ_m.
3. $hasObligation(\rho_m, g_j)$ - Quem assume o papel ρ_m é obrigado a concluir o objetivo g_j.
4. $hasPermission(\rho_m, g_j)$ - Quem assume o papel ρ_m tem a permissão de concluir o objetivo g_j.
5. $isReached(g_k)$ - O objetivo g_k foi alcançado.
6. $isPreRequisite(gp_i, g)$ - Os objetivos g pertencentes ao grupo gp_i devem ser concluidos para que haja condição de executar g.
7. $hasCondition(g_i, cg_n)$ - Um objetivo do tipo g_i possui certas condições c que deve estar presentes e devem se manter durante toda execução deste objetivo. Essas condições c devem estar conditdas em cg_n.
8. $hasEntity(g_i, eg_i)$ - Um objetivo g_i tem um conjunto de entidades eg_i onde todas as entidades presentes neste conjunto devem estar presentes no momento da execução desse objetivo.
9. $hasRelation(g_i, rg_i)$ - Um objetivo g_i tem um conjunto de relacionamentos rg_i onde todos esses relacionamentos devem ser feito para que este objetivo seja concluído.
10. $isPresent(cg_n)$ - Todas as condições pertencentes a cg_n estão presentes.
11. $tryReach(ag_i, g_j)$ - Um determinado agente ag_i tenta alcançar o objetivo g_j.
12. $violationCondition(ag_i, g_j, c_k)$ - Um determinado agente ag_i comete uma violação de condição no objetivo g_j sobre a condição c_k.
13. $violationNotPermission(ag_i, g_j)$ - O agente ag_i comete uma violação de não permissão g_j - tentativa de alcançar um objetivo mesmo sem ter permissão para isso.
14. $violationObligation(ag_i, g_j)$ - O agente ag_i comete uma violação de obrigação g_j - não executa o objetivo mesmo quando é obrigado a fazer isso e quando o objetivo está no momento certo de ser executado.
15. $violationRelation(ag_i, g_j, r_k)$ - O agente ag_i comete uma violação de Relacionamento no objetivo g_j por não realizar o relacionamento r_k.
16. $violationEntity(ag_i, g_j, e_k)$ - O agente ag_i comete uma violação de Entidade no objetivo g_j por tentar alcançar esse objetivo sem ter a entidade e_k presente.
17. $hasRisk(c_k, risk_j, f_m)$ - A condição c_k está associada a um risco $risk_k$ com uma certa fatalidade f_m.
18. $hasRisk(r_k, risk_j, f_m)$ - O relacionamento r_k está associado a um risco $risk_k$ com uma certa fatalidade f_m.

19. $hasRisk(e_k, risk_j, f_m)$ - A entidade e_k está associada a um risco $risk_k$ com uma certa fatalidade f_m.
20. $badEvent(ag, risk_j, f_m)$ - Agente ag sofre as consequências do risco $risk_j$ com a fatalidade f_m

Definição 7

$$attribution(ag, \rho_{r_i}) \wedge deonticRelation(\rho_r, g_{e_i}, d_{o_i}) \rightarrow obligation(ag, g_{e_i}) \quad (1)$$

Motivo

Se um agente posssui atribuição de um papel e se um determinado objetivo está dentro do escopo deste papel, então esse agente possui a obrigação de alcançar este objetivo.

Definição 8

$$attribution(ag, \rho_{r_i}) \wedge deonticRelation(\rho_{r_i}, g_{e_i}, d_{p_i}) \rightarrow permission(ag, g_{e_i}) \quad (2)$$

Motivo

Se um agente posssui atribuição de um papel e se um determinado objetivo está dentro do escopo deste papel, então esse agente possui a permissão de alcançar este objetivo.

Definição 9

$$obligation(ag, g_{e_i}) \rightarrow permission(ag, g_{e_i}) \quad (3)$$

Motivo

Toda a obrigação implica em uma permissão.

Definição 10

Se para iniciar um determinado objetivo g_{e_j} é necessário completar outros objetivos, $g_{e_i}, ...g_{e_k}$, se pelo menos um destes objetivos for falso para $isCompleted(g_x), x = i, .., k$, então $preCondition(g_j, (g_i, ...g_k))$ é falso, se não, então $preCondition(g_j, (g_i, ...g_k))$ é verdadeiro.

Sendo que existe um conjunto $g_{preRequisit} = \{g_i, ...g_k\}$, então

$$(preCondition(g_j, g_{preRequisite}) \rightarrow preRequisit(g_j)) \wedge \quad (4)$$
$$\neg(preCondition(g_j, g_{preRequisite}) \rightarrow \neg preRequisit(g_j)) \quad (5)$$

Motivo

Um objetivo pode ser executado apenas se os pre-requisitos forem alcançados.

Definição 11

Sendo cm conjunto de condições que devem ser mantidos o tempo inteiro para que a atividade possa acontecer,

$$preRequisit(g_{e_i}) \wedge hasMaintainer(g_{e_i}, cm_i) \wedge isOk(cm_i) \rightarrow canStart(g_{e_i}) \quad (6)$$

Motivo

Considerações que devem ser alcançadas e mantidas para que um determiando objetivo tenha condições de começar a ser alcançado. Essas considerações consiste em ter os seus pre-requisitos alcançados (Definição 10), e as condições temporais devem possibilitar a atividade.

Definição 12

$$preRequisit(g_{e_i}) \wedge hasMaintainer(g_{e_i}, cm_i) \wedge \neg isOk(cm_i) \wedge tryReach(g_{e_i}, ag) \rightarrow \quad (7)$$
$$violation(g_{e_i}, ag, cm_i) \quad (8)$$

Motivo

Se o agente tentar executar um determinado objetivo mesmo que as condições

Definição 13

Se $attribution(ag, \rho_r) \wedge \neg deonticRelation(\rho_r, g_{e_i}, d_{p_i}) \wedge canStart(g_{e_i}) \wedge tryReach(g_{e_i}, ag)$ é verdade, então $violation(d_{p_i}, ag)$ é verdade. Contudo, se $attribution(ag, \rho_r) \wedge \neg deonticRelation(\rho_r, g_e, d_{p_i}) \wedge canStart(g_e) \wedge tryReach(g_{e_i}, ag)$ for falso, então $violation(d_{p_i}, ag)$.

Definição 14

Existe c_r, onde c_r é um conjunto que mapeia as relações das entidades da pela relação $isRelationOf(r_j, c_{r_i})$. Cada g_{e_i} possui um c_{r_i} dada pela relação $hasConditionRelation(g_{e_i}, c_{r_i})$. Se $permission(ag, g_{e_i}) \wedge canStart(g_{e_i}) \wedge hasConditionRelation(g_{e_i}, c_{r_i}) \wedge isRelationOf(r_j, c_{r_i}) \wedge \neg know(ag, r_j) \wedge tryReach(ag, g_{e_i})$ é verdade, então é verdade $violation(r_j, g_{e_i}, ag)$. Contudo $violation(r_j, g_{e_i}, ag)$ é falso para a condição de $permission(ag, g_{e_i}) \wedge canStart(g_{e_i}) \wedge hasConditionRelation(g_{e_i}, c_{r_i}) \wedge isRelationOf(r_j, c_{r_i}) \wedge \neg know(ag, r_j) \wedge tryReach(ag, g_{e_i})$ ser falsa.

Definição 15

Se $permission(ag, g_{e_i}) \wedge canStart(g_{e_i}) \wedge hasConditionRelation(g_{e_i}, c_{r_i}) \wedge isRelationOf(r_j, c_{r_i}) \wedge \neg execute(ag, r_j) \wedge tryReach(ag, g_{e_i})$ é verdade, então é verdade $violation(r_j, g_{e_i}, ag)$. Contudo $violation(r_j, g_{e_i}, ag)$ é falso para a condição de $permission(ag, g_{e_i}) \wedge canStart(g_{e_i}) \wedge hasConditionRelation(g_{e_i}, c_{r_i}) \wedge isRelationOf(r_j, c_{r_i}) \wedge \neg execute(ag, r_j) \wedge tryReach(ag, g_{e_i})$ ser falsa.

Definição 16

Se $attribution(ag, \rho_r) \wedge deonticRelation(\rho_r, g_{e_i}, d_{o_i}) \wedge canStart(g_{e_i}) \wedge \neg tryReach(ag, g_{e_i})$ então existe uma violação dada como $violation(d_{o_i}, ag, g_{e_i})$. Se não $violation(d_{o_i}, ag, g_{e_i})$ é falso.

Definição 17

Existe um conjunto c_e, condition entity, que mapeia as entidades necessárias $isEntityOf(e_k, c_{e_i})$ para que um determinado objetivo possa ser alcançado. Para cada g_{e_i} existe uma dada relação em que $hasConditionEntity(g_{e_i}, c_{e_i})$. Sendo assim, $permission(ag, g_{e_i}) \wedge tryReach(ag, g_{e_i}) \wedge canStart(g_{e_i}) \wedge hasConditionEntity(g_{e_i}, c_{e_i}) \wedge isEntityOf(e_k, c_{e_i}) \wedge \neg isInMomentOfGoal(e_k, g_{e_i}) \rightarrow violation(g_{e_i}, e_k, ag)$. Contudo, é verdadeiro $\neg violation(g_{e_i}, e_k, ag)$ se $hasConditionEntity(g_{e_i}, c_{e_i}) \wedge tryReach(ag, g_{e_i}) \wedge canStart(g_{e_i}) \wedge hasConditionEntity(g_{e_i}, c_{e_i}) \wedge isEntityOf(e_k, c_{e_i}) \wedge \neg isInMomentOfGoal(e_k, g_{e_i})$ é falso. A relação $isInMomentOfGoal(e_k, g_{e_i})$ indica se entidade estará disponível no momento que o agente ag tenta alcançar o objetivo g_{e_i}.

Definição 18

$$attribution(ag, \rho_a) \rightarrow obligation(ag, g_d) \tag{9}$$

Definição 19

Se for verdade $obligation(ag_i, g_d) \wedge delegate(ag_i, \rho_{r_i}, ag_j) \wedge deonticRelation(\rho_{r_i}, g_{e_i}, d_p) \wedge hasConditionRelation(g_{e_i}, c_{r_i}) \wedge isRelation(r_k, c_{r_i}) \wedge \neg know(ag_j, r_k)$ então ocorre uma violação em $violation(\rho_a, g_d, ag_i)$. Se não, $\neg violation(\rho_a, g_d, ag_i)$ é verdade.

Definição 20

$$violation(d_{p_i}, ag) \rightarrow stopOperation \tag{10}$$
$$\neg violation(d_{p_i}, ag) \rightarrow \neg stopOperation \tag{11}$$

Definição 21

$$violation(r_j, g_{e_i}, ag) \wedge isGoalRandon(g_{e_j}) \wedge hasPossibilityBadEvent(g_{e_j}, p_1) \tag{12}$$
$$\rightarrow hasPossibilityBadEvent(g_{e_j}, p_2) \wedge p_2 > p_1 \tag{13}$$
$$\neg violation(r_j, g_{e_i}, ag) \wedge isGoalRandon(g_{e_j}) \wedge hasPossibilityBadEvent(g_{e_j}, p_1) \tag{14}$$
$$\rightarrow hasPossibilityBadEvent(g_{e_j}, p_1) \tag{15}$$

Definição 22

$$violation(r_j, g_{e_i}, ag) \rightarrow sanction(g_{e_i}, ag, risk) \rightarrow badEvent(risk, fatality, ag, g_{e_i}) \tag{16}$$
$$\neg violation(r_j, g_{e_i}, ag) \rightarrow \neg sanction(g_{e_i}, ag, risk) \rightarrow \neg badEvent(risk, fatality, ag, g_{e_i}) \tag{17}$$

Definição 23

$$hasPossibilityBadEvent(g_{e_j}, p) \wedge eventBadHappens(g_{e_j}) \wedge tryReach(ag, g_{e_j}) \quad (18)$$
$$\rightarrow badEvent(risk, fatality, ag, g_{e_j}) \quad (19)$$
$$hasPossibilityBadEvent(g_{e_j}, p) \wedge \neg eventBadHappens(g_{e_j}) \wedge tryReach(ag, g_{e_j}) \quad (20)$$
$$\rightarrow \neg badEvent(risk, fatality, ag, g_{e_j}) \quad (21)$$
$$hasPossibilityBadEvent(g_{e_j}, p) \wedge eventBadHappens(g_{e_j}) \wedge \neg tryReach(ag, g_{e_j}) \quad (22)$$
$$\rightarrow \neg badEvent(risk, fatality, ag, g_{e_j}) \quad (23)$$

Definição 24

$$violation(d_{o_i}, ag, g_{e_i}) \rightarrow stopOperation \quad (24)$$
$$\neg violation(d_{o_i}, ag, g_{e_i}) \rightarrow \neg stopOperation \quad (25)$$

Definição 25

$$violation(g_{e_i}, e_k, ag) \rightarrow sanction(g_{e_i}, ag, risk) \rightarrow badEvent(risk, fatality, ag, g_{e_i}) \quad (26)$$
$$\neg violation(g_{e_i}, e_k, ag) \rightarrow \neg sanction(g_{e_i}, ag, risk) \rightarrow \neg badEvent(risk, fatality, ag, g_{e_i}) \quad (27)$$

Definição 26

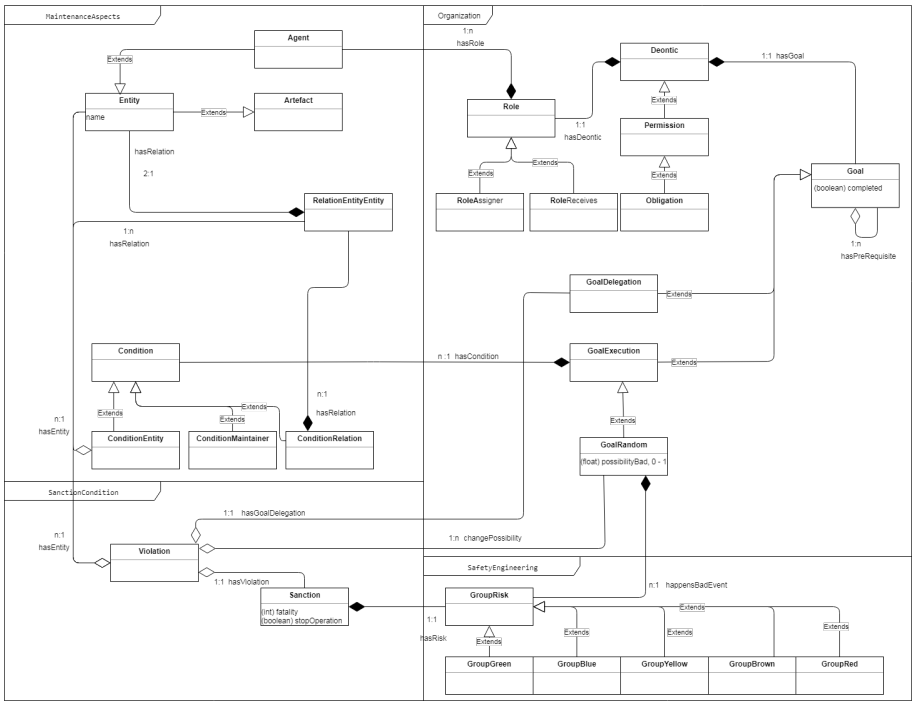
$$violation(g_{e_i}, ag, cm_i) \rightarrow sanction(g_{e_i}, ag, risk) \rightarrow badEvent(risk, fatality, ag, g_{e_i}) \quad (28)$$
$$\neg violation(g_{e_i}, e_k, ag) \rightarrow \neg sanction(g_{e_i}, ag, risk) \rightarrow \neg badEvent(risk, fatality, ag, g_{e_i}) \quad (29)$$

Definição 27

$$badEvent(risk, fatality, ag, g_{e_i}) \rightarrow stopOperation \quad (30)$$
$$\neg badEvent(risk, fatality, ag, g_{e_i}) \rightarrow \neg stopOperation \quad (31)$$

Definição 28

Para todos os agentes que receberam uma obrigação de executar g_{e_i}, ao cumprir com a condição $tryReach(g_{e_i}, ag) \wedge \neg stopOperation$ então $isCompleted(g_{e_i})$, se não, então é verdade $failed(g_{e_i})$


MaintenanceAspects
Agent
Extends
Entity
name
Extends
Artefact
hasRelation
2:1
RelationEntityEntity
1:n
hasRelation
Condition
Extends
Extends
Extends
n:1
hasEntity
ConditionEntity
ConditionMaintainer
ConditionRelation
n:1
hasRelation
SanctionCondition
n:1
hasEntity
Violation
1:1 hasViolation
Sanction
(int) fatality
(boolean) stopOperation
Organization
1:n
hasRole
Role
1:1
hasDeontic
Deontic
Extends
1:1 hasGoal
Permission
Extends
Extends
Extends
RoleAssigner
RoleReceives
Obligation
Goal
(boolean) completed
1:n
hasPreRequisite
GoalDelegation
Extends
n:1 hasCondition
GoalExecution
Extends
Extends
GoalRandom
(float) possibilityBad, 0 - 1
1:1 hasGoalDelegation
1:n changePossibility
SafetyEngineering
n:1 happensBadEvent
GroupRisk
1:1
hasRisk
Extends
Extends
Extends
Extends
GroupGreen
GroupBlue
GroupYellow
GroupBrown
GroupRed