Algorithms to Compute Appraisal Variables

Algorithm 1 (Relevance)

```
function IsEventRelevant(Event $\varepsilon_t$)

    Initialize graph $\mathcal{G}_t$ with current mental state $\mathcal{S}_t$.

    $g_t \leftarrow$ extractGoal($\mathcal{G}_t$)

    $\mathcal{P}_t \leftarrow$ extractPath($\varepsilon_t, g_t$)

    if ($\mathcal{P}_t = \emptyset$) then
        return FALSE
    else
        $\mathcal{U}_t \leftarrow$ getEventUtility($\varepsilon_t, g_t$)
        if ($\mathcal{U}_t \geq \tau_e$) then
            return TRUE
        else
            return FALSE
end function
```

Algorithm 2 (Desirability)

```
function IsEventDesirable(Event $\varepsilon_t$)

    Initialize graph $\mathcal{G}_t$ with current mental state $\mathcal{S}_t$.

    if ($topLevelTaskStatus() = $ ACHIEVED) then
        return TRUE
    else if ($topLevelTaskStatus() = $ BLOCKED) then
        return FALSE
    else if ($topLevelTaskStatus() = $ INPROGRESS) then

        if ($currentTaskStatus() = $ ACHIEVED) then
            return TRUE
        else if ($currentTaskStatus() = $ BLOCKED) then
            return FALSE
        else if ($currentTaskStatus() = $ INPROGRESS) then
            return TRUE
        else if ($currentTaskStatus() = $ UNKNOWN) then

            if ($taskPreconditionStatus() = $ SATISFIED) then
                return TRUE
            else if ($taskPreconditionStatus() = $ UNSATISFIED) then
                return FALSE
            else if ($taskPreconditionStatus() = $ UNKNOWN) then

                if ($doesContribute(\varepsilon_t, \vec{g_t}) = $ TRUE) then
                    return TRUE
                else if ($doesContribute(\varepsilon_t, \vec{g_t}) = $ FALSE) then

                    $g_t \leftarrow$ extractGoal($\mathcal{G}_t$)
                    $\mathcal{U}_t \leftarrow$ getEventUtility($\varepsilon_t, g_t$)
                    if ($\mathcal{U}_t \geq \tau_e$) then
                        return TRUE
                    else
                        return FALSE
                    end if

                end if
            end if
        end if
    end if
end function
```

Algorithm 3 (Expectedness)

```
function IsEventExpected(*Event* $\varepsilon_t$)

    Initialize graph $\mathcal{G}_{t-1}$ with previous mental state $\mathcal{S}_{t-1}$.
    Initialize graph $\mathcal{G}_t$ with current mental state $\mathcal{S}_t$.

    $g_{t-1} \leftarrow$ extractGoal($\mathcal{G}_{t-1}$)
    $g_t \leftarrow$ extractGoal($\mathcal{G}_t$)

    if ($g_t \neq g_{t-1}$) then
        if (IsAchieved($g_{t-1}$) = FALSE) then
            return FALSE
        else
            $\mathcal{P}_t \leftarrow$ extractPath($\varepsilon_t, g_t$)
            if ($\mathcal{P}_t = \emptyset$) then
                return FALSE
            else
                $\mathcal{U}_t \leftarrow$ getPathUtility($\mathcal{G}_t, g_t$)
                if ($\mathcal{U}_t \geq \tau_e$) then
                    return TRUE
                else
                    return FALSE
    else
        $\mathcal{U}_t \leftarrow$ getEventUtility($\mathcal{G}_t, g_t$)
        if ($\mathcal{U}_t \geq \tau_e$) then
            return TRUE
        else
            return FALSE
end function
```

Algorithm 4 (Controllability)

```
function IsEventControllable($\mathit{Event}\ \varepsilon_t$)

    Initialize graph $\mathcal{G}_t$ with current mental state $\mathcal{S}_t$.

    $g_t \leftarrow$ extractGoal($\mathcal{G}_t$)
    $\mathcal{P}_{g_t} \leftarrow$ extractPath($\varepsilon_t, g_t$)

    $\alpha^{agency}_{self/other} \leftarrow$ GetAgencyRatio($g_t$)

    $\beta^{autonomy}_{self/other} \leftarrow$ GetAutonomyRatio($g_t$)

    $\lambda^{predecessors}_{succeeded/total} \leftarrow$ GetSucceededPredecessorsRatio($g_t$)

    $\mu^{inputs}_{available/required} \leftarrow$ GetAvailableInputRatio($g_t$)

    $\mathcal{U}_t \leftarrow \frac{\omega_0 \cdot \alpha^{agency}_{self/other} + \omega_1 \cdot \beta^{autonomy}_{self/other} + \omega_2 \cdot \lambda^{predecessors}_{succeeded/total} + \omega_3 \cdot \mu^{inputs}_{available/required}}{\omega_0 + \omega_1 + \omega_2 + \omega_3}$

    if ($\mathcal{U}_t \geq \tau_e$) then
        return TRUE
    else
        return FALSE
end function
```

Algorithm 5 (Get Succeeded Predecessors Ratio)

```
function GetSucceededPredecessorsRatio($Goal\ g_t$)

    $\Phi_g \leftarrow$ extractPredecessors($g_t$)

    for each $\phi_g^i \in \Phi_g$ do
        if (IsAchieved($\phi_g^i$)) then
            $count_{predecessor}^{achieved} \leftarrow count_{predecessor}^{achieved} + 1$

        $count_{predecessor}^{total} \leftarrow count_{predecessor}^{total} + 1$
    end for

    return $\frac{count_{predecessor}^{achieved}}{count_{predecessor}^{total}}$
end function
```

Algorithm 6 (Get Available Input Ratio)

```
function GetAvailableInputRatio($Goal\ g_t$)

    $\mathcal{X}_g \leftarrow$ extractInputs($g_t$)

    for each $\chi_g^i \in \mathcal{X}_g$ do
        if (IsAvailable($\chi_g^i$)) then
            $count_{input}^{available} \leftarrow count_{input}^{available} + 1$

        $count_{input}^{required} \leftarrow count_{input}^{required} + 1$
    end for

    return $\frac{count_{input}^{available}}{count_{input}^{required}}$
end function
```

Algorithm 7 (Get Agency Value)

```
function GetAgencyValue(Goal $g_t$)

    $\Theta_g \leftarrow$ ExtractContributingTasks($g_t$)

    for each $\theta^i_{\vec{g}} \in \Theta_{\vec{g}}$ do
        if (GetResponsible($\theta^i_g$) = SELF) then
            $count^{self}_{responsibility} \leftarrow count^{self}_{responsibility} + 1$

        else if (GetResponsible($\theta^i_g$) = BOTH) then
            $count^{self}_{responsibility} \leftarrow count^{self}_{responsibility} + 0.5$
    end for

    return $\frac{count^{self}_{responsibility}}{\Theta_g.size()}$
end function
```

Algorithm 8 (Get Autonomy Value)

```
function GetAutonomyValue(Goal $g_t$)

    $\mathcal{P}_{g_t} \leftarrow$ extractPath($\varepsilon_t, g_t$)
    $\mathcal{M}_{g_t} \leftarrow$ extractMotive($\mathcal{P}_{g_t}$)

    if ($\mathcal{M}_{g_t} \neq \emptyset$) then
        if (getMotiveType($\mathcal{M}_{g_t}$) = INTERNAL) then
            return 1.0
        else if (getMotiveType($\mathcal{M}_{g_t}$) = EXTERNAL) then
            return 0.5
    else
        return 0.0
end function
```