UNIVERSIDAD ANDRÉS BELLO
Facultad de Ingeniería

Sistema de apoyo a la toma de decisiones bajo ambientes dinámicos.

Tesis para optar al Título de Ingeniero Civil Informático

Autor: Raúl Silva Jeldes
Profesor Guía: Romina Torres

Viña del mar, 2019.

Índice general

Resumen

Las organizaciones son sistemas complejos que evolucionan rápidamente adaptándose a diferentes situaciones que surgen en el tiempo bajo un ambiente dinámico. En en este contexto que la toma de decisiones que consideren múltiples periodos, evaluadores, criterios es hoy más relevante que nunca. Los modelos de toma de decisión dinámicos han sido propuestos para abordar la toma de decisiones en un ambiente real.

Lamentablemente, a pesar de que estos modelos incorporan el consenso de cada tomador de decisiones durante cada periodo bajo consideración, la estabilidad del proceso de toma de decisiones en el tiempo se ve afectada si los diferentes elementos evaluados cambian constantemente y por lo tanto el significado de cada categoría de clasificación por cada atributo.

En este proyecto se construirá un sistema de apoyo a la toma de decisiones para ambientes dinámicos, donde se utiliza toma de decisiones lingüística debido a la naturaleza no cuantificable de los atributos. Para ello primero se construirán casos sintéticos que permitan emular ambientes dinámicos que afecten la estabilidad de los algoritmos de ranking de alternativas (fundamentales para apoyar la toma de decisión de selección) que utilizan modelos multi-periodos lingüísticos para apoyar la toma de decisiones. Segundo, se construirá un algoritmo que permita mitigar el efecto de la naturaleza de los ambientes dinámicos en los significados de cada categoría lingüística (clasificación) utilizada para evaluar por los evaluadores. Tercero, se incorporará en el método de agregación de estas evaluaciones un mecanismo de aprendizaje-olvido que permita mitigar el impacto del dinamismo en el significado de las categorías, con el fin de mejorar la estabilidad en el proceso de toma de decisiones en el tiempo.

Capítulo 1

Contexto

1.1. Toma de decisiones

La toma de decisiones grupales (GDM) es usualmente visto como el proceso de seleccionar una alternativa adecuada ante un conjunto de alternativas funcionalmente equivalentes de acuerdo al consenso de un grupo de evaluadores. Cuando los evaluadores solo tienen información parcial del entorno, ellos especifican los criterios de evaluación usando expresiones lingüísticas. En esos casos el modelo de toma de decisiones lingüística (LDM) se ha utilizado con éxito para resolver problemas de decisiones mal estructurados que no pueden ser resueltos con las herramientas clásicas de la teoría de decisiones o, que están bajo entornos inciertos.

Un modelo MCDM bajo evaluaciones lingüísticas es representado de la siguiente forma. Sea $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_n\}$ el conjunto de n alternativas $(i = 1, \ldots, n)$, Sea $G = \{G^1, G^2, \ldots, G^m\}$ ser el m atributo de las alternativas $(j = 1, \ldots, m)$, Sea $E = \{e_1, \ldots, e_l, \ldots, e_L\}$ los L evaluadores, Sea $v = (v_1, \ldots, v_L)^T$ la importancia relativa de los evaluadores, donde $v_l \geq 0$ $(l = 1, 2, \ldots, L)$, y sea $\omega = (\omega_1, \omega_2, \ldots, \omega_m)^T \in H$ la importancia del vector de atributos, $\omega \geq 0$ $(i = 1, 2, \ldots, m)$ y $\sum_{i=1}^{m} \omega_i = 1$. Asumiendo que los evaluadores usan una escala de evaluación lingüística aditiva (S_2), Ellos proveen sus preferencias sobre las alternativas $\mathcal{A}_i$ $(i = 1, 2, \ldots, n)$ con respecto a los atributos $G^j \in G$ y construyen una matriz de decisión lingüística $\mathbf{R} = (r_{ij})_{m \times n}$.

Una opinión grupal de los L evaluadores debe ser obtenida por cada atributo de cada alternativa. Asumiendo que un operador ha sido seleccionado para agregar las L diferentes opiniones (Por ejemplo LWA_2 [Xu08a])

se obtiene lo siguiente para cada miembro de la matriz $\mathbf{R}$:

$$r_{ij} = LWA_2(r_{ij1}, \ldots, r_{ijl}, \ldots, r_{ijL}) = v_1 r_{1j1} \oplus \ldots \oplus v_L r_{ijL}$$

Basado en la información de la decisión $\mathbf{R} = (r_{ij})_{m\times n}$, el valor total de la alternativa $\mathcal{A}_i$ puede ser expresado como

$$z_i(\omega) = \omega_1 r_{i1} \oplus \omega_2 r_{i2} \oplus \ldots \oplus \omega_m r_{im},\ i = 1, 2, \ldots, n$$

Mientras mas grande $z_i(\omega)$, mejor es la alternativa $\mathcal{A}_i$.

Considerando t como la variable tiempo, Xu [Xu08b] extendió el modelo MCDM a uno multi-periodo de la siguiente forma. Siendo t_p ser P diferentes periodos de tiempo ($p = 1, 2, \ldots, P$), Sea $\omega^t = (\omega^{t_1}, \omega^{t_2}, \ldots, \omega^{t_P})^T$ ser los vectores de peso de los atributos a través del tiempo, donde $\omega^{t_p} \geq 0$, $\sum_{p=1}^{P} \omega^{t_p} = 1$, Sea $\omega^{t_p} = (\omega_1^{t_p}, \omega_2^{t_p}, \ldots, \omega_m^{t_p})^T$ser los vectores de pesos de los atributos en el periodo t_p, donde $\omega_j^{t_p} \geq 0$ ($j = 1, 2, \ldots, m$) y $\sum_{j=1}^{m} \omega_j^{t_p} = 1$, sea $\mathbf{R}^{t_p} = (r_{ij}^{t_p})_{m\times n}$ ser las matrices de decisión lingüística de p diferentes periodos, donde $r_{ij}^{t_p}$ denota el valor de la evaluación del atributo proporcionada por el evaluador sobre la alternativa $\mathcal{A}_i$ con respecto al atributo G^j en el periodo t_p utilizando por ejemplo, la escala de evaluación lingüística aditiva S, y sea $s_\alpha(t)$ la etiqueta lingüística aditiva en el tiempo t, donde $s_\alpha(t) \in \overline{S}_2$ para cualquier t. si $t = t_1, t_2, \ldots, t_P$, entonces $s_\alpha(t_p)$ ($p = 1, 2, \ldots, P$) denota P etiquetas lingüísticas aditivas recogidas de P diferentes periodos.

Xu [Xu08b] propuso elPromedio ponderado lingüístico dinámico ($DLWA$) para el modelo de toma de decisiones de multi-periodos, un nuevo operador de agregación capaz de agregar evaluaciones lingüísticas basadas en el tiempo actual y anterior para cada experto sobre cada atributo de cada alternativa. Se define de la siguiente forma:

$$DLWA(s_\alpha(t_1), s_\alpha(t_2), \ldots, s_\alpha(t_p)) = \omega^{t_1} s_\alpha(t_1) \oplus \omega^{t_2} s_\alpha(t_2) \oplus \ldots \oplus \omega^{t_P} s_\alpha(t_P)$$

La clave del dinamismo del operador DLWA es determinar el vector de pesos r ω^t, donde el método basado en unidad básica de intervalo monotónico (BUM) [Yag04] se ha propuesto encontrarlo.

Sea $f : [0, 1] \to [0, 1]$ una funciona BUM (donde $f(0) = 0, f(1) = 1, f(x) \geq f(y)$ si $x > y$). El vector peso $\omega(t)$ es definido como

$$\omega^{t_p} = f\left(\frac{p}{P}\right) - f\left(\frac{p-1}{P}\right), p = 1, 2, \ldots, P$$

donde la secuencia $\{\omega^{t_p}\}$ es una secuencia monótona creciente(o decreciente), $\omega^{t_{p+1}} > \omega^{t_p}$ $(\omega^{t_{p+1}} < \omega^{t_p})$ con $p = 1, 2, \ldots, P-1$.

Xu [Xu08b] propuso el vector de peso de la siguiente forma:

$$\omega^{t_p} = \frac{e^{\frac{\alpha p}{P}} - 1}{e^{\alpha} - 1} - \frac{e^{\frac{\alpha(p-1)}{P}} - 1}{e^{\alpha} - 1} = \frac{e^{\frac{\alpha p}{P}} - e^{\frac{\alpha(p-1)}{P}}}{e^{\alpha} - 1} \tag{1.1}$$

donde f es una función monótona creciente dada por $f(x) = \frac{e^{\alpha x}-1}{e^{\alpha}-1}$, $\alpha > 0$.

El enfoque de 3 pasos usado para resolver problemas de múltiples-periodos y múltiples-atributos (MPMADM) es el siguiente:

- El primer paso es obtener los valores globales de atributos de las alternativas $\mathcal{A}_i$ en el periodo t_p mediante el uso de la versión básica del $DLWA$, es decir utilizando operador del promedio ponderado lingüístico (LWA)

$$\begin{aligned} z_i(t_p) &= LWA(r_{i1}^{t_p}, r_{i2}^{t_p}, \ldots, r_{im}^{t_p}) \\ &= \omega_1^{t_p} r_{i1}^{t_p} \oplus \omega_2^{t_p} r_{i2}^{t_p} \oplus \ldots \omega_m^{t_p} r_{im}^{t_p}, \end{aligned} \tag{1.2}$$

  con $p = 1, 2, \ldots, P$, $j = 1, 2, \ldots, m$.

- El segundo paso, es agregar el valor total de los atributos $z_i(t_p)$ $(p = 1, 2, \ldots, P)$ obtenidos de P periodos distintos, y obtener el valor total del atributo z_i de la alternativa $\mathcal{A}_i$ mediante el uso del operador $DLWA$ de la siguiente forma:

$$\begin{aligned} z_i &= DLWA_{\omega(t)}(z_i(t_1), z_i(t_2), \ldots, z_i(t_P)) \\ &= \omega^{t_1} z_i(t_1) \oplus \omega^{t_2} z_i(t_2) \oplus \ldots \omega^{t_P} z_i(t_P), \end{aligned} \tag{1.3}$$

  con $i = 1, 2, \ldots, n$.

- El tercer paso es clasificar las alternativas $\mathcal{A}$ de acuerdo al valor total de los atributos z_i.

otros operadores de agregación dinámicos han sido propuestos por Xu y Yagger[XY08], quién propuso los siguientes operadores "Promedio intuicionista dinámico ponderado difuso"(DIFWA) y "la incierta dinámica intuicionista difusa ponderada"(UDIFWA). Peng et al. [PW14] definió el operador "Promedio ponderado dinámico dudoso"(DHFWA) y el operador"Dudoso dinámico difuso ponderado geométrico"(DHFWG) con el fin de lidiar con las preferencias vacilantes a través del tiempo.

Torres et al. [TSA14] desarrolló dos nuevos operadores como extensión de los operadores "ponderado promedio dudoso priorizado"(HFPWA) y operador difuso priorizado ponderado geométrico dudoso (HFPWG). Estos nuevos operadores utilizan la función softmax modulada por el parámetro κ. La función softmax ha sido propuesta como una generalización de la función logística a múltiples variables. además a sido exitosamente aplicada en la literatura de machine learning[JJNH91, McC13, TSAM03]. La ecuación matemática para la función softmax es la siguiente:

$$\phi_\kappa(j, T_1, ..., T_n) = \frac{\exp(T_j/\kappa)}{\sum_{i=1}^{n} \exp(T_i/\kappa)} \qquad \kappa > 0. \tag{1.4}$$

donde la colección de valores $\{T_1, ..., T_j, ..., T_n\}$. Indicaremos la función softmax como $\phi_\kappa^j = \phi_\kappa(j, T_1, ..., T_n)$ para abreviar. La función softmax asegura que todos los valores de salida están entre 0 y 1 (i.e., $0 \leq \phi_\kappa^j \leq 1$, $j = 1...n$) y que su suma sea 1 (ejemplo., $\sum_{j=1}^{n} \phi_\kappa^j = 1$). La función softmax es una función no lineal, monótonamente creciente y limitada. Por otra parte, la función logística. $(1/(1 + e^{-x}))$ es un caso especial de la función softmax cuando solo se consideran dos valores ($x = T1 - T2$) [McC13].El parámetro de modulación κ Permite dar mayor o menor importancia al mayor valor de T_j en una forma flexible. Como el parámetro κ se vuelve mas pequeño, el mayor valor de T_j predomina, y en el limite como $\kappa \to 0$ El softmax converge en la maximización, ya que $\phi^j_{\kappa=0^+}$ tenderá a 1 para los más grandes T_j y se acercará a 0 para todos los demás casos. Por otro lado, como κ se hace más grande, todos los valores de salida de la función softmax tenderán a $\frac{1}{n}$,

- Promedio softmax priorizado dudoso y borroso(HFPSA):

$$\begin{aligned} \text{HFPSA}(h_1, ..., h_n, \kappa) &= \left(\phi_\kappa^1 h_1\right) \oplus \left(\phi_\kappa^2 h_2\right) \oplus ... \oplus \left(\phi_\kappa^n h_n\right) \\ &= \bigoplus_{j=1}^{n} \left(\phi_\kappa^j h_j\right) \end{aligned} \tag{1.5}$$

- El operador dudoso borroso priorizado softmax geométrico (HFPSG):

$$\begin{aligned} \text{HFPSG}(h_1, ..., h_n, \kappa) &= (h_1)^{\phi_\kappa^1} \otimes (h_2)^{\phi_\kappa^2} \otimes ... \otimes (h_n)^{\phi_\kappa^n} \\ &= \bigotimes_{j=1}^{n} (h_j)^{\phi_\kappa^j} \end{aligned} \tag{1.6}$$

donde $\phi_\kappa^j = \frac{\exp(T_j/\kappa)}{\sum_{i=1}^{n} \exp(T_i/\kappa)}$.

Sobre la base de la operación de los valores "vacilante y confusosz el principio de extensión, las siguientes propiedades pueden ser demostradas siguiendo el mismo mecanismo dado por G. Wei[Wei12] para los operadores HFPWA y HFPWG. En lo que sigue consideramos que tenemos una colección de HFE h_j,j=1,....,m la función softmax $\phi_\kappa^j = \frac{\exp(T_j/\kappa)}{\sum_{i=1}^{n}\exp(T_i/\kappa)}$, el producto acumulado T_j y la puntuación de valores $S_{\phi(j)}$ de $h_{\phi(j)}$, j = 1,....,m.

- (1) El valor agregado de del HFE h_j, j=1,....,m usando ya sea el operador de agregación HFPWA o HFPWG.

$$\begin{aligned} \text{HFPSA}(h_1,...,h_n,\kappa) &= \left(\phi_\kappa^1 h_1\right) \oplus \left(\phi_\kappa^2 h_2\right) \oplus ... \oplus \left(\phi_\kappa^n h_n\right) \\ &= \bigoplus_{j=1}^{n} \left(\phi_\kappa^j h_j\right) \end{aligned} \tag{1.7}$$

$$\begin{aligned} \text{HFPSG}(h_1,...,h_n,\kappa) &= (h_1)^{\phi_\kappa^1} \otimes (h_2)^{\phi_\kappa^2} \otimes ... \otimes (h_n)^{\phi_\kappa^n} \\ &= \bigotimes_{j=1}^{n} (h_j)^{\phi_\kappa^j} \end{aligned} \tag{1.8}$$

- (2)(Idempotencia) si los elementos de esta colección son todos iguales, entonces HFPSA(h_1,....h_m,k) y HFPSG (h_1,....h_m,k) = h.

- (3)(Delimitación) sea h- = $min_j h_j$ y h+ = $max_j h_j$. entonces las siguientes propiedades son ciertas:

$$h^-(h_1,....h_m,k) \leq h^+$$

.

$$h^-(h_1,....h_m,k) \leq h^+$$

.

- (4)(Monoticidad) Sea h_{j^*}, j=1,....,n otra colección de HFE. Si $h_j \leq$h$_{j^*}$ entonces:

$$HFPSA(h_1,....h_m,k) \leq HFPSA(h_{1^*},....h_{m^*},k)$$

$$HFPSG(h_1,....h_m,k) \leq HFPSG(h_{1^*},....h_{m^*},k)$$

1.2. Ejemplos Ilustrativos

A continuación se presentarán dos casos de estudios para ilustrar la toma de decisiones grupales, multiatributo, lingüística, multiperiodo y con evaluadores dudosos.

1.2.1. Caso de estudio 1: Enfoque de agregación de información difusa basada en el tiempo para problemas de toma de decisiones no cuantitativas

Una empresa multinacional de TI que brinda servicios de desarrollo y administración de sistemas de inteligencia empresarial en big data tiene una alta tasa de rotación de empleados. Este es un problema serio para su competitividad:

- Cuando un empleado clave acude a la competencia se entrega ventaja a los competidores.
- Interrupción en el trabajo Impacta la negatividad en la entrega del servicio a los clientes debido a la pérdida de memoria organizativa adquirida al perder un empleado.
- El reemplazo del empleado no es fácil debido a su conocimiento técnico, habilidades y experiencia, y el proceso de reemplazo tiene un alto costo.

Para enfrentar esto, la compañía ha definido una estrategia de lealtad y retención de los mejores talentos en posiciones centrales de negocios. Las posiciones se evaluarán como ingeniero de software, analista de negocios, ciencia de datos y gerente de proyectos. Todos los empleados en la misma posición, por ejemplo, ingenieros de software, serán evaluados y clasificados. Las primeras posiciones de cada ranking ingresarán al programa de talento superior con los siguientes beneficios:

- Anualmente, habrá un aumento en su compensación mensual: 15 por ciento al primer lugar y 8 por ciento a todos los demás talentos principales.
- Tendrán un mentor para diseñar y acelerar su plan de carrera, a través de la capacitación y el entrenamiento adaptados a sus propias necesidades

- Se les pagarán bonos semestrales, donde el primer lugar en el ranking del cargo, recibirá el 100 por ciento del bono, el segundo lugar recibirá el 80 por ciento del bono y los demás recibirán el 50 por ciento del bono.
- Cada año se sorteará una beca para estudiar un título de posgrado entre los mejores talentos que ocupan el primer lugar en el ranking de su posición.

Por otro lado, como parte de la estrategia de retención interna, el área de recursos humanos habrá definido los límites de negociación que tendrá con cada posición de la clasificación en caso de que el empleado tenga la intención de abandonar la organización.

La herramienta utilizada para tomar decisiones sobre quiénes son los empleados con mayor talento será la .Evaluación de personal". Los empleados han sido evaluados cada trimestre por diferentes y varios gerentes de diferentes áreas importantes de la compañía durante todos los años. El desafío en este proceso es promover la justicia en las decisiones sobre quiénes son los empleados con mayor talento, para que el programa no afecte negativamente a la organización. Por lo tanto, el Gerente de Recursos Humanos consideró primordial no solo la última evaluación sino también el historial completo. A continuación se mostrará el proceso para elegir a los mejores ingenieros de software
Sean i evaluadores, donde i =1,...,4, específicamente.

- e1=Gerente de ventas
- e2=Gerente de recursos humanos
- e3=Gerente de negocios
- e4=Gerente de post venta.

Los ingenieros de software a evaluar son:

- a1=ing-soft-1
- a2=ing-soft-2
- a3=ing-soft-3
- a4=ing-soft-4.

Cada miembro del panel de evaluadores, que es responsable de evaluar a los ingenieros de software, tiene la misma importancia en el proceso de toma de decisiones. Los aspectos evaluados son los siguientes:

- a1=Actitud de trabajo
- a2=Habilidad de comunicación
- a3=Resolución de problemas
- a4=Habilidad de aprendizaje

donde a1=0.5, a2=0.3, a3=0.1, a4=0.1.

- 1=Muy Bajo
- 2=Bajo
- 3=Medio
- 4=Bueno
- 5=Muy bueno.

Una vez que la evaluación está configurada con estos datos, el panel de evaluadores puede responder a las evaluaciones. Las evaluaciones se muestran en la tabla 1.1, tabla 1.2, tabla 1.3, tabla 1.4.

Nombre	**a1**	**a2**	**a3**	**a4**	**Resultado**
ing-soft-1	Muy Bajo	Bajo	Medio	Bueno	0.45
ing-soft-2	Muy Bajo	Muy Bajo	Muy Bajo	Muy Bajo	0.25
ing-soft-3	Muy bueno	Muy bueno	Muy bueno	Muy bueno	1.25
ing-soft-4	Muy Bajo	Bajo	Medio	Bueno	0.45
ing-soft-5	Muy bueno	Muy bueno	Muy bueno	Muy bueno	1.25

Tabla 1.1: Evaluación hecha por el Gerente de ventas.

El valor de ing-soft-1 en la Tabla 1.4 se obtiene multiplicando el valor de la etiqueta (Muy bajo = 1) * la importancia del criterio(a1=0.5) + Valor de la etiqueta (bajo = 2) * importancia del criterio (a2 = 0.3) + valor de la etiqueta (medio = 3) * importancia del criterio (a3 = 0.1) + valor de la etiqueta (bueno = 4) * importancia del criterio (a4 = 0.1).

Name	**a1**	**a2**	**a3**	**a4**	**Resultado**
ing-soft-1	Muy bueno	Muy bueno	Muy bueno	Muy bueno	1.25
ing-soft-2	0	0	0	0	0
ing-soft-3	Bajo	Muy Bajo	Medio	Bueno	0.5
ing-soft-4	Bajo	Bajo	Medio	Bueno	0.575
ing-soft-5	Muy bueno	Muy bueno	Muy bueno	Muy bueno	1.25

Tabla 1.2: Evaluación hecha por el Gerente de recursos humanos.

Nombre	**a1**	**a2**	**a3**	**a4**	**Resultado**
ing-soft-1	Bajo	Medio	Bajo	Medio	0.6
ing-soft-2	Bueno	Bajo	Medio	Bueno	0.825
ing-soft-3	Bueno	Bueno	Bueno	Bueno	1
ing-soft-4	Bueno	Medio	Bueno	Muy bueno	0.95
ing-soft-5	Muy bueno	Bueno	Muy bueno	Medio	1.125

Tabla 1.3: Evaluación hecha por el Gerente de negocios.

Nombre	**a1**	**a2**	**a3**	**a4**	**Resultado**
ing-soft-1	Bajo	Bajo	Muy Bajo	Bajo	0.475
ing-soft-2	Bueno	Medio	Bajo	Bueno	0.875
ing-soft-3	Medio	Bajo	Bueno	Medio	0.7
ing-soft-4	Medio	Bueno	Bajo	Bueno	0.825
ing-soft-5	Bueno	Medio	Muy bueno	Bajo	0.9

Tabla 1.4: Evaluación hecha por el Gerente de post venta

Así que para este ingeniero de software (a1) su valor de evaluación será: 1 * 0.5 + 2 * 0.3 + 3 * 0.1 + 4 * 0.1 = 1.8

El resultado de esta ponderación se multiplica por el peso del evaluador, en este caso el gerente de ventas (e1) vale 0.25:
1.8 *0.25=0.45.

El resultado final de la evaluación para el ing-soft-1 en la evaluación del Gerente de ventas es igual a 0.45. Este proceso se realiza para cada evaluador y para cada ingeniero de software.

Al agregar los resultados de todos los evaluados por el evaluador, se obtiene la puntuación total para cada software de ingeniero:

- ing-soft-1 = 0,45+1,25+0,6+0,475 = 2,775
- ing-soft-2 = 0,25+0+0,825+0,875 = 1,95
- ing-soft-3 = 1,25+0,5+1+0,7 =3,45
- ing-soft-4 = 0,45+0,575+0,95+0,825= 2,8
- ing-soft-5 = 1,25+1,25+1,125+0,9 =4,53

Si esta evaluación fuera hecha solo en un periodo el ing-soft-5 seria el mejor ingeniero de software, ingresando al programa de talento superior de la compañía de TI y obteniendo todos los beneficios todos los beneficios por ser el mejor rankeado entre los ingenieros de software.

Ahora, si quisiéramos ampliar el caso anterior y considerar las evaluaciones anteriores realizadas por los mismos evaluadores y a los mismos evaluados para obtener mejor data teniendo los resultados registrados por cada ingeniero de software y su puntuación en cada criterio en los años 2014, 2015 y 2016 como se ven en las siguientes tablas:

Fecha	**a1**	**a2**	**a3**	**a4**
2014	1.625	0.35	0.675	0.3250
2015	1.5	0.4	1.125	0.45
2016	1.25	0.35	0.9	0.275

Tabla 1.5: Evaluación de ing-soft-1 de 3 periodos distintos

Fecha	a1	a2	a3	a4
2014	1.75	0.4	1.05	0.275
2015	1.375	0.4	0.90	0.325
2016	1.125	0.225	0.45	0.15

Tabla 1.6: Evaluación de ing-soft-2 de 3 periodos distintos

Fecha	a1	a2	a3	a4
2014	1.875	0.375	1.125	0.30
2015	2.125	0.4	1.2	0.375
2016	1.75	0.4	0.9	0.4

Tabla 1.7: Evaluación de ing-soft-3 de 3 periodos distintos

Fecha	a1	a2	a3	a4
2014	1.75	0.425	1.35	0.45
2015	1.125	0.2749	Medio	0.30
2016	1.25	0.425	0.825	0.3

Tabla 1.8: Evaluación de ing-soft-4 de 3 periodos distintos

Fecha	a1	a2	a3	a4
2014	1.625	0.4	1.125	0.425
2015	1.375	0.3750	0.9	0.325
2016	2.375	0.375	1.2750	0.5

Tabla 1.9: Evaluación de ing-soft-5 de 3 periodos distintos

Teniendo datos de 3 períodos diferentes (2014, 2015, 2016) para cada ingeniero de software, donde cada año tiene el mismo peso, promediamos cada valor de los criterios por año de evaluación.

Hacemos un promedio de cada valor de los criterios por año de evaluación.

- a1: (2014*2015*1016)/3
- a2: (2014*2015*1016)/3
- a3: (2014*2015*1016)/3

- a4: (2014*2015*1016)/3

Ahora todo lo que tenemos que hacer es sumar todos los criterios (a1 + a2 + a3 + a4) para que nuestra nueva puntuación tenga en cuenta tres períodos.

1.2.2. Caso de estudio 2: Enfoque de agregación de información difusa basada en el tiempo para problemas de toma de decisiones cuantitativas

Usando los mismos criterios, evaluadores y evaluadores pero diferentes etiquetas:

- 0=Muy Bajo
- 0.25=Bajo
- 0.5=Medio
- 0.75=Bueno
- 1=Muy Bueno.

Suponemos que un ingeniero de software fue evaluado en 2014, 2015, 2016 con los siguientes resultados:

	Ev1	**Ev2**	**Ev3**	**Ev4**
2014	Muy Bajo	Bajo	Bajo	0
2015	Medio	Bajo	Medio	Medio
2016	Bueno	Bajo	Bajo	Medio

Tabla 1.10: Criterio 1

	Ev1	**Ev2**	**Ev3**	**Ev4**
2014	Medio	Bueno	Muy Bueno	Medio
2015	Muy Bueno	Muy Bueno	Bueno	Bueno
2016	Bueno	Muy Bueno	X	Muy Bueno

Tabla 1.11: Criterio 2

	Ev1	**Ev2**	**Ev3**	**Ev4**
2014	Bajo	X	X	Bueno
2015	Medio	X	0	Bajo
2016	Bueno	Bajo	Medio	Bueno

Tabla 1.12: Criterio 3

	Ev1	**Ev2**	**Ev3**	**Ev4**
2014	Muy Bueno	Muy Bueno	Medio	Muy Bueno
2015	Muy Bueno	Bueno	Bueno	Muy Bueno
2016	Muy Bueno	Muy Bueno	Muy Bueno	Muy Bueno

Tabla 1.13: Criterio 4

Creamos una nueva matriz llamada matriz H que elimina los valores repetidos de todos los criterios:

	C1	**C2**	**C3**	**C4**
2014	Muy Bajo;Bajo	Medio;Bueno;Muy Bueno	Bajo;Bueno	Medio;Muy Bueno
2015	Bajo;Medio	Bueno;Muy Bueno	Bajo;Bajo;Medio	Bueno;Muy Bueno
2016	Bajo;Medio;Bueno	Bueno;Muy Bueno	Bajo;Medio;Bueno	Muy Bueno

Tabla 1.14: Matrix H

A continuación, debemos calcular la función Score calculando el promedio de cada valor por criterio en los años 2014, 2015, 2016:

	C1	**C2**	**C3**	**C4**
2014	0.125	0.75	0.5	0.75
2015	0.375	0.875	0.25	0.875
2016	0.5	0.875	0.5	1

Tabla 1.15: Matrix H

Ahora calculamos los pesos temporales de cada criterio utilizando los datos de los próximos años multiplicando los criterios de los siguientes puntajes, pero como 2016 tiene el peso actualizado de todos los criterios, todos los criterios de ese año son iguales a 1.

	C1	**C2**	**C3**	**C4**
2014	0.1875	0.766	0.125	0.875
2015	0.5	0.875	0.5	1
2016	1	1	1	1

Tabla 1.16: Pesos temporales

Ahora, para cada año y para cada criterio, realizamos la suma de los pesos temporales, por ejemplo para c1 agregamos los datos de 2014 + 2015 + 2016 (0.1875 + 0.5 + 1) dando 1.6875 y para cada año dividimos el peso cual del criterio por la sumatoria de los criterios a través de los años. Ejemplo c1 en 2014=0.111 = 0.1875/1.6875.

Ahora tenemos valores para cada criterio:

- C1=Actitud:0, 025, 0,3070, 0,3770, 0,4629, 0,4799, 0,5887
- C2=Comunicación:0,6668, 0,7335, 0,7436, 0,8179, 0,7500, 0,8254, 0,8363, 0,9200, 0,8152, 0,8968, 0,9091, 1,000
- C3=Problemas:0, 0,25, 0,3094, 0,3830, 0,4740, 0,4715, 0,6084, 0,2720, 0,3367, 0,4168, 0,5158, 0,5349, 0,6620
- C3=Aprendizaje:0,7327, 0,8098, 0,9048, 1,00

Ahora el Score de cada criterio se calcula sacando el promedio del criterio obteniendo los siguientes resultados:

- C1=0,2054.
- C2=0,8262.
- C3=0,2915.
- C4=0,8618.

y, finalmente, el valor de cada criterio se eleva al peso asignado y se multiplica por cada criterio para obtener la puntuación final:
SCORE = $0{,}2054^{0,5}$ * $0.8262^{0,3}$ * $0.2915^{0,1}$ * $0.8618^{0,1}$
SCORE = 0.3728.

Capítulo 2

Problema

2.1. Problema

En este trabajo de título, nos centraremos en el enfoque de agregación de información difusa basada en el tiempo para problemas de toma de decisiones no cuantitativa, donde se utilizan modelos de decisiones lingüísticas y no necesariamente existen datos cuantitativos que permitan a los expertos apoyar la definición de los significados de cada categoría de cada atributo lingüístico utilizado a posteriori por los evaluadores para evaluar las alternativas.

Dada la competencia inherente que existe entre las diferentes alternativas, tanto la percepción de los expertos como de los evaluadores respecto del significado de cada categoría de cada valor lingüístico puede fluctuar en cada periodo ya que el ser humano tiende a tomar decisiones a corto plazo pensando solo en el presente sin tener en cuenta el tiempo. Por tanto la importancia que recibe cada evaluación por cada criterio puede ser diferente de manera de poder mitigar el efecto de las fluctuaciones en ambiente dinámico.

Si esto no es considerado, entonces la estabilidad del proceso de toma de decisiones en el tiempo puede verse afectado, produciendo decisiones prematuras o tardías.

2.2. Objetivo General

Desarrollar una herramienta web que permita utilizar el modelo MP-MCDM con el objetivo de minimizar el efecto del ambiente dinámico en la toma de decisiones.

2.3. Objetivos específicos

Los objetivos específicos del proyecto son:

1. Seleccionar un algoritmo de agregación de información lingüística que agregue las evaluaciones en el tiempo mitigando el efecto del dinamismo del ambiente y de los evaluadores con dudas durante el proceso de evaluación.

2. Construir casos sintéticos que muestren las fortalezas de la propuesta

3. Crear una herramienta que permita validar en ambiente real que el objetivo se alcanza.

Capítulo 3

Estado del arte

En muchos escenarios del mundo real, el proceso de generar flujos de datos no es estacionario, pero es caracterizado por un fenómeno intrínseco no estacionario [DRAP15] debido a la estacionalidad, cambios en los hábitos de los usuarios, envejecimiento de los sensores, por nombrar unos pocos. Esto ocurre cuando la variación del contexto induce cambios en el concepto objetivo[WK96]. El concepto objetivo son los valores lingüísticos, los cuales son creados utilizando un proceso basado en percepciones, donde en diferentes áreas ya se ha reconocido que los sistemas perceptivos pueden degradarse cuando los entornos están cambiando mucho [Hal06]. En otras palabras, el significado de los valores lingüísticos se pueden volver obsoletos debido al concepto drift. Sin embargo, las técnicas que se ocupan del concepto Drift suponen que los datos se ajustan a una distribución estacionaria[Las02], lo cual en un proceso basado en percepciones no es posible asumir. La solución obvia seria que cada vez que el modelo vaya a ser usado se deba definir la semántica del conjunto de términos lingüísticos. Desafortunadamente esto no seria lo suficientemente robusto contra todos los tipos de cambios del entorno y por lo tanto, esto podría provocar la toma de decisiones prematuras.

Por ejemplo: en la Figura 3.1 se muestra un caso en el que esta solución funciona perfectamente, las alternativas están mejorando a través del tiempo y por lo tanto cada vez que se vuelven a evaluar los modelos anteriores las alternativas rechazadas no son válidas nuevamente, lo que provoca que las decisiones prematuras pueden ser descartadas. Diferente es el caso presentado en la Figura 2 y 3, donde se muestran casos en los que los cambios aleatorios en el entorno a través del tiempo afectan a la percepción de los expertos sobre los conceptos excellent y fair. En el primer caso se puede ver

que el significado del concepto excellent en diferentes instancias del tiempo es fluctuante y, por lo tanto, la decisión tomada en el tiempo 1 y rechazada en el tiempo 3 podría ser válida nuevamente en el tiempo 5.

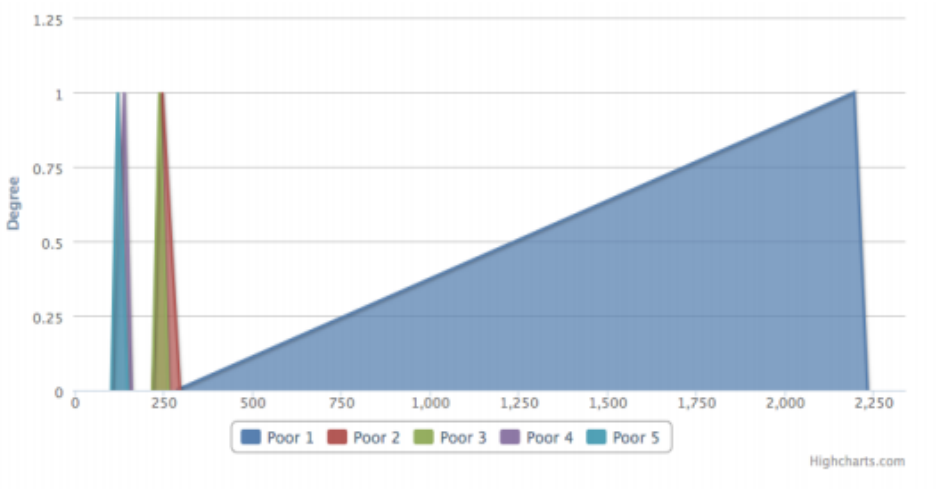


Figura 3.1: El termino lingüístico “Poor” en cinco instancias secuenciales de tiempo esta dado por Poor 1, Poor 2, Poor 3, Poor 4 y Poor 5 presentan una tendencia de mejora probablemente porque las alternativas están mejorando debido a la competencia.

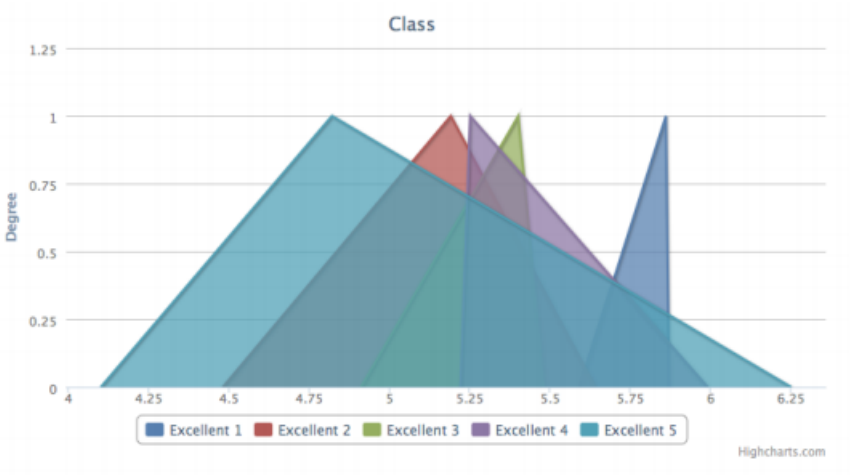


Figura 3.2: El termino lingüístico "Excellent" en cinco instancias secuenciales de tiempo esta dado por Excellent 1, Excellent 2, Excellent 3, Excellent 4 and Excellent 5 afectados por un ruido aleatorio que hace que las percepciones de los expertos fluctúen en el tiempo.

Figura 3.3: El termino lingüístico "Fair" en cinco instancias secuenciales de tiempo esta dado por Fair 1, Fair 2,Fair 3, Fair 4 y Fair 5 afectados por un ruido aleatorio que hace que las percepciones de los expertos fluctúen en el tiempo.

Según [GAGK18] un algoritmo para manejar el concepto Drift a través del tiempo debe tener un mecanismo de aprendizaje-olvido y una estrategia explicita de detección de cambios. Un mecanismo de aprendizaje-olvido hace que el algoritmo aumente en las tasas de aprendizaje y olvido cuando se detecta un cambio de concepto. Algunos algoritmos pueden mantener explícitamente una librería de los conceptos originales con el fin de desempeñarse mejor cuando el flujo de datos incluye conceptos recurrentes[GAGK18].
Existen también enfoques activos o pasivos para el aprendizaje en presencia del concepto Drift[DRAP15] de acuerdo a si se detecta el Drift o simplemente se actualiza el modelo cada vez que lleguen datos nuevos, independiente del Drift. La selección de un enfoque depende del dinamismo de los escenarios de aprendizaje, los recursos computacionales disponibles y las suposiciones sobre la distribución de los datos. El Razonamiento basado en casos (CBR) es un método de aprendizaje basado en modelos que almacena la experiencia pasada como reglas generalizadas y resuelve problemas nuevos. La mantención basada en casos es el proceso de revisitar los contenidos de un sistema CBR. Cuando ocurre el concepto Drift, los casos pasados pueden volverse obsoletos[LLZdM16]. Por lo que se ha propuesto el algoritmo de cambio rápido de contexto mejorado con ruido para evitar que se incluya el ruido durante el caso.[LLZdM16]. Los estudios en la detección del Drift son basados en :

- Distribución de datos(Estimación de los parámetros de distribución).
- Agrupación de datos(estimación de la cantidad de modificaciones en las particiones de datos)[Las02]

Para este ultimo caso Sethi et al. propuso la metodología de detección de drift de la densidad del margen(MD3)(la cual es una técnica de agrupamiento que utiliza una medida de desviación en las regiones de agrupación para detectar el Drift).

Torres[Tor14] propuso un mecanismo para computar y detectar Drifts en valores lingüísticos de variables lingüísticas en un índice de similitud entre fuzzy sets[GZ01]. En la teoría de fuzzy sets, no hay una noción incorporada de grado de subsistencia o grado de igualdad y similitud entre los fuzzy sets[NK08]. Luego, se denen utilizando las nociones básicas de unión e intersecciones entre ellas (union: $\mu_{A\cup B}(x) = S(\mu_A(x), \mu_B(x))$ y interseccion: $\mu_{A\cap B}(x) = T(\mu_A(x), \mu_B(x))$). De acuerdo a Gaweda y Zurada[GZ01] el grado de similitud ($Sim(A, B)$) entre dos fuzzy sets A y B es definido como subsistencia mutua de la siguiente forma:

$$Sim(A,B) \quad = Degree(A \subseteq B \ y \ B \subseteq A) = \frac{M(A\cap B)}{M(A\cup B)} = \frac{M(A\cap B)}{M(A)+M(B)-M(A\cap B)} \tag{3.1}$$

Donde $\cap$ y $\cup$ denota la intersección y unión de los fuzzy sets A y B respectivamente, y $M(\cdot)$ es el tamaño de los fuzzy set, y $0 \leq Sim(A,B) \leq 1$. Cuanto mas A y B sean similares, mayor será $Sim(A,B)$.

Sea A el fuzzy set representando un valor lingüístico en el tiempo t_a, sea B el fuzzyset representando el mismo valor lingüístico en el tiempo t_b (donde $t_b > t_a$). por lo tanto, cuán similares son las dos representaciones temporales del significado del valor lingüístico puede ser calculado. Independiente de cual ecuación de similitud es usada, definimos el Drift temporal entre las dos representaciones temporales del significado de un valor lingüístico específico como

$$Drift(A,B) = \varsigma(1 - Sim(A,B)) \tag{3.2}$$

donde ς ($\varsigma \in \{-1, 0, 1\}$) puede tomar el valor 1 si M_A es ubicado en el lado izquierdo de M_B y toma el valor -1 en el caso contrario. Si $M_A = M_B$ entonces ς es 0. El signo del índice de Drift proporcionará la información si el fuzzy set A se mueve hacia la izquierda o hacia el lado derecho. Figure 4 muestra los índices para los seis casos diferentes cuando se usan tringular fuzzy sets.

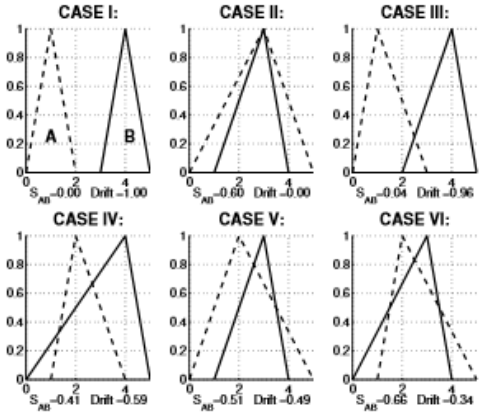


Figura 3.4: La similitud y Drift se calculan para seis casos diferentes que deben considerarse cuando se usa la ecuación de similitud para calcular la similitud de los triangular fuzzy numbers.

Drift temporal es un síntoma del concepto Drift. con el fin de mitigar

el riesgo de detectar ruido o valores atípicos en lugar de Drift, Torres et al. [TBA13] propuso contar el número de síntomas de drift en una ventana y también contarlo si, y solo si el drift temporal es mayor que un umbral, por ejemplo: $|Drift(A, B)| > \rho$. En este caso, aplicamos un algoritmo fuzzy c-means al flujo de datos sin procesar para emular los acuerdos de expertos (calcular las agrupaciones) que representan cada valor lingüístico para cada variable lingüística.

Capítulo 4

Metodología

En este capítulo, se explicará la arquitectura de una herramienta Web para apoyar la toma de decisiones grupales bajo ambiente dinámico junto con la metodología de gestión del proyecto.

4.1. Herramienta

diagrama de componentes explicación

Como se muestra en la figura, el usuario del sistema podrá acceder a través de una conexión a Internet al servidor. A través de distintas vistas se conecta con la base de datos y a través de diferentes criterios se realizará la evaluación y/o se verán las entidades evaluadas.

Para la creación de la herramienta Web se utilizará:

- El framework de trabajo django que utiliza el lenguaje python, el cual se usará la versión 3.7.

- Una base de datos tipo SQL(PostgresSQL).

4.2. Metodología de Gestión del proyecto

El proyecto se ejecutará con una metodología ágil en vez de una tradicional, específicamente Scrum ya que esta permite manejar el proyecto de manera flexible, permite la reestructuración en caso de atrasos, y permite detectar errores en etapas tempranas y permite la adaptación y flexibilidad de nuevos requisitos o cambios en estos. El proyecto se realizará en bloques temporales cortos (3 semanas) los cuales en Scrum se denomina como sprint, donde al final de cada iteración se debe entregar un resultado completo.

El trabajo comienza con la visión general del producto, donde se especifican las funcionalidades que tendrá el sistema y se les otorga prioridad, luego éstas se separan en iteraciones para comenzar a trabajar. Los principales elementos de Scrum son: Product backlog, Sprint backlog, el resultado, las reuniones y los roles que son product owner, scrum master y el equipo de desarrollo. Product backlog es la lista de requisitos del sistema en la visión inicial del proyecto que puede ir creciendo a medida que avanza el mismo. Sprint backlog constituye las tareas que se deben realizar en cada sprint para generar el resultado esperado. Las reuniones son la clave de esta metodología. Existen tres tipos de reuniones: la primera es la de planificación del sprint donde se determina cuál va a ser el trabajo y los objetivos de la iteración a realizar; la segunda son las reuniones diarias donde se revisa el trabajo realizado y por realizar; y la última es la revisión del sprint donde se analiza y revisa el incremento generado.

4.2.1. Planificación

Al inicio del proyecto todos los requerimientos se pasan al product backlog en forma de historias de usuario, se hace una priorización junto con el product owner y una estimación de cada una de las historias con la técnica de planning póker. La cual consiste en que cada integrante del equipo dispone de un juego de cartas con un valor de la escala de Fibonacci de 1 a 100, además del product backlog del proyecto. Para cada historia de usuario cada integrante levanta la carta con la cifra que se aproxime más a la estimación que se cree le corresponde a esa historia, en caso de ser distintas las estimaciones se conversan las razones de la elección del valor y se llega a un acuerdo del valor final. En este proyecto al ser solo un integrante en el equipo se recurre al product owner para realizar la estimación.

1	Product Backlog				Sort Product Backlog		
2							
3							
4	**Story ID**	**Story name**	**Status**	**Size**	**Sprint**	**Priority**	**Comments**
7	3	Ver ranking	Planned	13			
8	4	Asignación de rol	Planned	13			
9	5	Comentar evaluación	Planned	5			
10	6	Crear registro de usuario.	planned	13			
11	7	Periodos de la evaluación	Planned	21			
12	8	Historial de evaluaciones	Planned	13			
13	9	Agregar criterios de evaluación	Planned	8			
14	10	Editar los criterios de evaluación.	Planned	8			
15	11	Importancia del periodo	Planned	21			
16	12	Importancia de los evaluadores	Planned	21			
17	13	Eliminar evaluación	Planned	5			
18	14	Agregar evaluadores.	Planned	13			

Figura 4.1: Product Backlog

Para gestionar los riesgos del proyecto se utiliza una matriz de riesgos, en ella se listan los riegos que se encontraron para la realización del proyecto, se establece lo que pasaría si el riesgo ocurre, la probabilidad de que ocurra, el impacto que tendría y la respuesta que se determinó para mitigar cada uno de los riesgos. Además, se establece una prioridad de 1 a 9 de acuerdo a la probabilidad y el impacto que tiene cada riesgo, donde 1 es el más alto.

N°	Riesgos	Plan de Contingencia	1ra Semana			2da Semana		
			Probabilidad/Amenaza	Impacto/Daño	Valor	Probabilidad/Amenaza	Impacto/Daño	Valor
1	No saber Django	Realizar un Spike	4	2	8	2	2	4
2	El codigo ya realizado no funcione en Django	Cambiar framework	1	3	3	1	3	3
3	Paro en la Universidad	Trabajar desde casa, mostrando los avances por Microsoft Teams	4	2	8	2	2	4
4	Problema para las reuniones	Realizar reunion por Zoom	4	1	4	4	1	4
5	Acumulación de trabajos	Organizar mejor el tiempo	4	2	8	4	2	8
6	Problemas de internet para trabajar	Pedir un note y trabajar en la universidad	2	2	4	2	2	4

Figura 4.2: Tabla de riesgos

4.2.2. Sprints

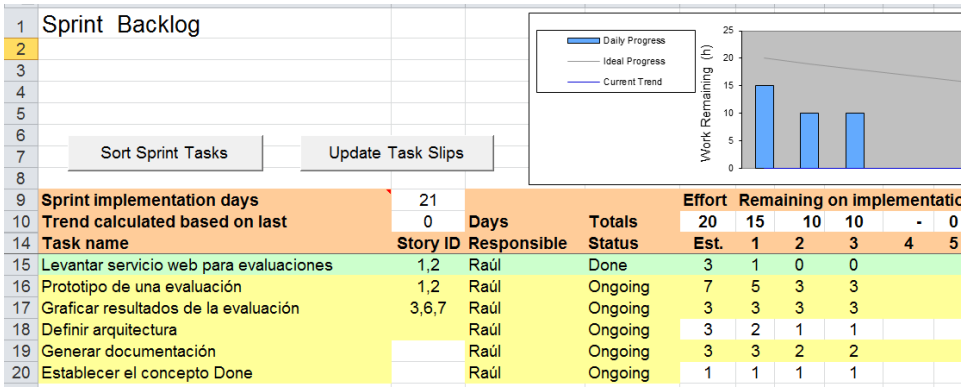
Para comenzar el sprint se realiza la reunión de planificación junto con el product owner donde se revisa el product backlog, y según la priorización se decide las historias de usuario que se van a desarrollar, las que se pasan al sprint backlog. También se establece el objetivo del sprint y la duración de este.

Para comenzar a trabajar en el sprint se divide el trabajo en tareas con una estimación en horas. Estas tareas se incluyen en una planilla de Excel que contiene las historias de usuario con su estimación en puntos de historia, las tareas con su estimación en horas, la fecha de inicio y término, el avance diario, el atraso diario, el gráfico Burndown de horas restantes y el estado de las tareas (pendiente, comenzado o terminado). Este Excel es actualizado diariamente y sirve para medir el progreso que en scrum se evalúa en la reunión diaria. Las reuniones con el product owner se realizan una vez a la semana para mostrar avance. Durante el desarrollo del sprint se revisan semanalmente los riesgos se revisan semanalmente para ver su evolución y la probabilidad de que ocurran o para ver si aparecen nuevos riesgos en la semana que deban agregarse a la lista y evaluar cuando es necesario ejecutar los planes de contingencia. En cuanto a las pruebas se realizan distintos tipos de pruebas que incluyen: pruebas unitarias, pruebas de aceptación, pruebas de integración, pruebas de estrés y pruebas de carga. Para finalizar el sprint se realiza la reunión de entrega con el product owner donde se revisa que se cumplan los criterios de aceptación y el product owner acepta o rechaza el sprint. En caso de aceptarlo se cierra el sprint y se genera la versión de liberación, en caso de rechazarlo se anotan los comentarios del product owner, y se pasan al product backlog para su posterior revisión. Una vez realizada esta reunión se realiza la retrospectiva, que consiste en evaluar cómo se desarrolló el sprint, las cosas que funcionaron bien, las que hay que mejorar, las que hay que dejar de hacer y las que hay que comenzar a hacer para el próximo sprint.

4.2.3. Control de actividades

Se utilizará un tablero Trello para gestionar las actividades del proyecto. Cada actividad se puede clasificar en 4 estados:

- Por hacer.
- Haciendo.
- Revisando.



9	**Sprint implementation days**	21			**Effort**	**Remaining on implementatio**				
10	**Trend calculated based on last**	0	**Days**	**Totals**	**20**	**15**	**10**	**10**	**-**	**0**
14	**Task name**	**Story ID**	**Responsible**	**Status**	**Est.**	**1**	**2**	**3**	**4**	**5**
15	Levantar servicio web para evaluaciones	1,2	Raúl	Done	3	1	0	0		
16	Prototipo de una evaluación	1,2	Raúl	Ongoing	7	5	3	3		
17	Graficar resultados de la evaluación	3,6,7	Raúl	Ongoing	3	3	3	3		
18	Definir arquitectura		Raúl	Ongoing	3	2	1	1		
19	Generar documentación		Raúl	Ongoing	3	3	2	2		
20	Establecer el concepto Done		Raúl	Ongoing	1	1	1	1		

Figura 4.3: Sprint Backlog

- Terminado.

Y cada actividad cuenta con una persona a cargo de ella.

4.2.4. Control de versiones

Se utilizará GitHub donde se irán subiendo los avances del software, empezando de la versión 0.1 y aumento en 0.1 por cada nueva versión. También en esta herramienta se marcarán los distintos ïssues"que surjan en el proyecto.

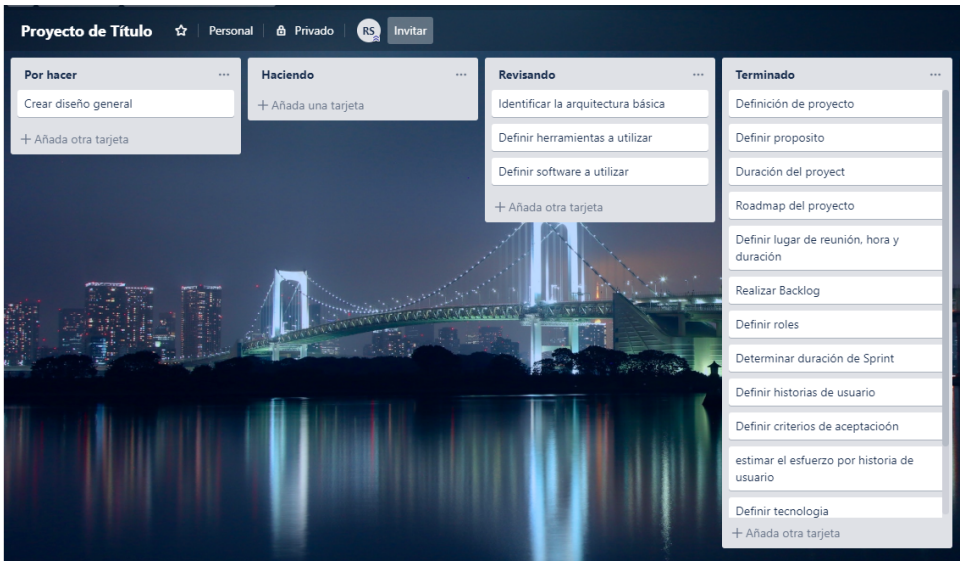


Figura 4.4: Tablero Trello

Capítulo 5

Resultados

Dentro de esta sección se encuentra:

- Componentes del sistema.
- Tipos de evaluaciones.
- Modelo de datos.
- Desarrollo de la aplicación.

Para la realización de estas tareas primero se debió definir como funcionaría la aplicación por lo que se creo un diagrama de componentes:

5.1. Componentes

Este diagrama muestra todos los componentes internos que utiliza la aplicación para la creación de una evaluación, esto se encuentra dividido en 3 capas, donde el usuario de la aplicación solo tiene acceso a la primera capa que es la que llama a los componentes de las demás capas para su correcto funcionamiento.

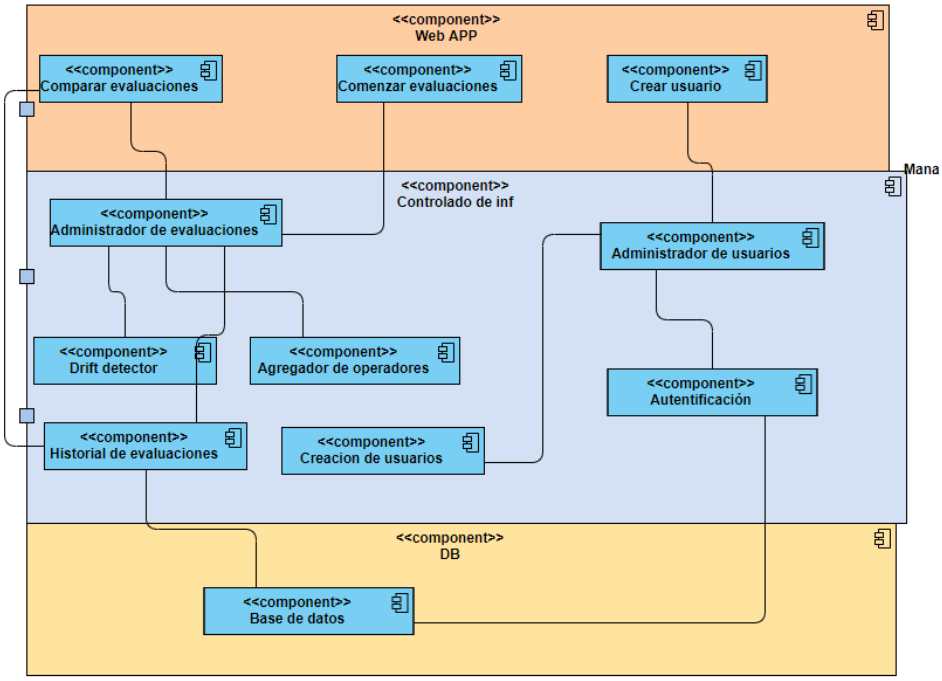


Figura 5.1: Componentes del sistema

5.2. Tipos de evaluaciones:

Una evaluación puede ser tanto manual o automática, para la automática los datos son recogidos de una carpeta en especifico donde se rescatarán todos los componentes necesarios para realizar la evaluación, mientras que para una evaluación manual cada componente de la evaluación es realizado por los actores de dicha evaluación como se verá a continuación.

5.2.0.1. Evaluación Manual:

Se definen los roles de .Administrador", .Evaluador", .Evaluadoz .Experto"los cuales son los actores principales para la realización de todo el proceso de creación de una evaluación.

5.2.0.2. Administrador:

Es el encargado de crear la evaluación, se encarga de asignar los roles de experto, evaluado, evaluador, asignar los criterios que se utilizarán, el tiempo en el que se generará un nuevo periodo para la evaluación y finalmente es el encargado de decidir la mejor opción a elegir de la evaluación.

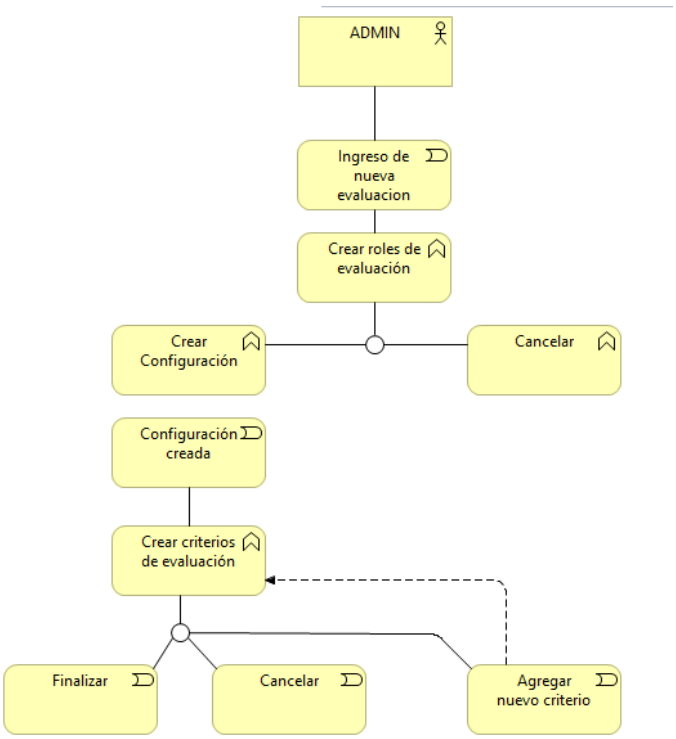


Figura 5.2: Proceso de creación de roles y criterios por parte del administrador

5.2.0.3. Experto:

Termina de configurar la evaluación creada por el experto, creando las etiquetas lingüísticas para los criterios creados previamente por el administrador de la evaluación.

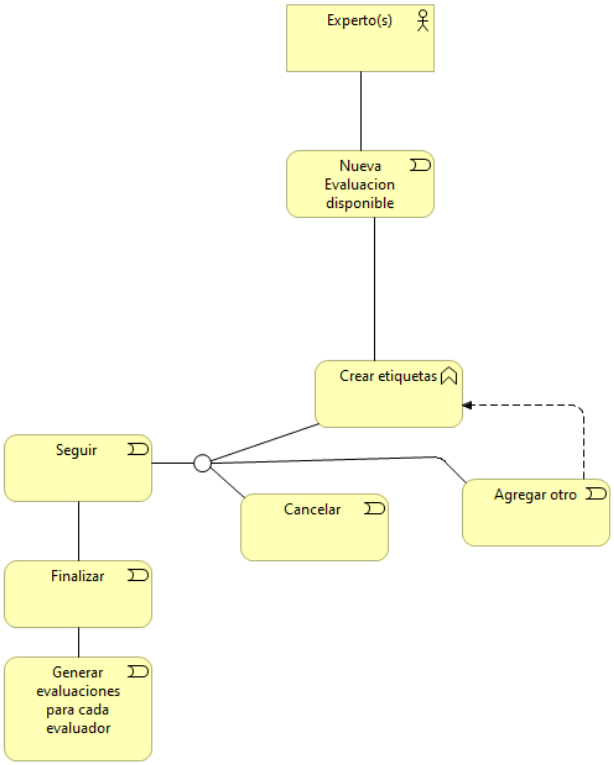


Figura 5.3: Proceso de creación de etiquetas realizado por el experto

5.2.0.4. Evaluador:

Su papel empieza una vez que el experto termina de crear las etiquetas de la evaluación, pudiendo ver en su lista de evaluaciones las evaluaciones en las que participa. Es el encargado de evaluar a los evaluados según los criterios y las etiquetas creadas anteriormente

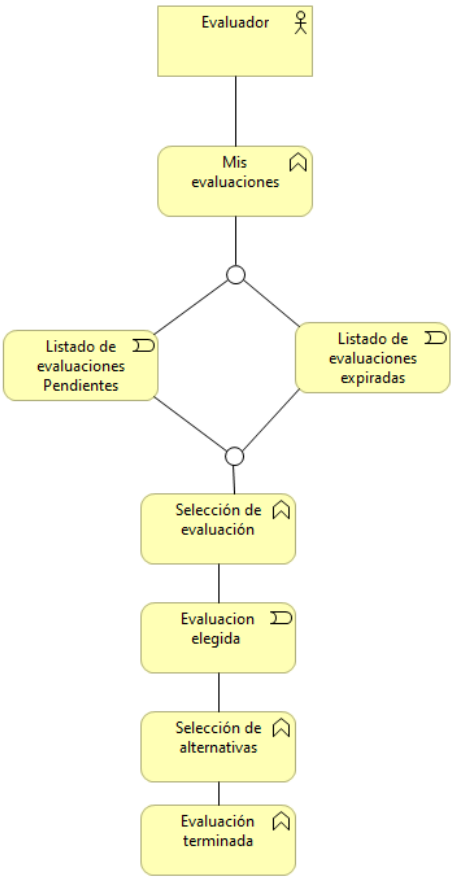


Figura 5.4: Proceso de evaluación realizado por el Evaluador

5.2.0.5. Evaluado:

El solo tiene el rol de ver las evaluaciones en las que ha sido evaluado, pudiendo compararse con los demás evaluados de dichas evaluaciones.

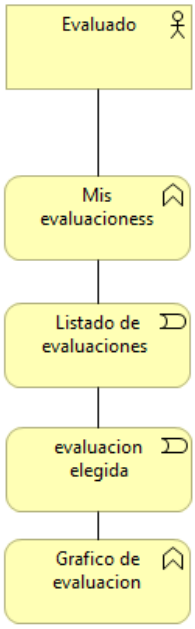


Figura 5.5: Proceso de ver ranking de evaluaciones por parte del evaluado

Realizando todo el proceso llegaríamos a un resultado como el que se puede ver a continuación:

Figura 5.6: Interfaz de gráfico de una evaluación manual

5.2.0.6. Evaluación automática:

A diferencia de una evaluación manual donde tanto los criterios como las etiquetas lingüísticas eran definidas por personas, la evaluación automática se caracteriza de obtener estos datos a través de archivos csv, donde el administrador crea una nueva evaluación de tipo automática subiendo un archivo csv del cual se usaran los atributos designados en ese archivo, al crear la configuración se creará una carpeta con el nombre de la evaluación la cual será el destino de los próximos archivos que se envíen al sistema. Estos archivos deben seguir un patrón especifico, donde la primera linea es el nombre de la alternativa a evaluar, en la segunda linea están los atributos de dicha alternativa, y desde la tercera linea hacia abajo, la primera columna representa la fecha de obtención de los datos, el resto de las columnas representa el valor numérico obtenido de dicho atributo. (Cada linea de este archivo representa un periodo distinto). Una vez se obtienen los atributos, el experto puede completar la configuración de la evaluación designando las etiquetas de los atributos encontrados.

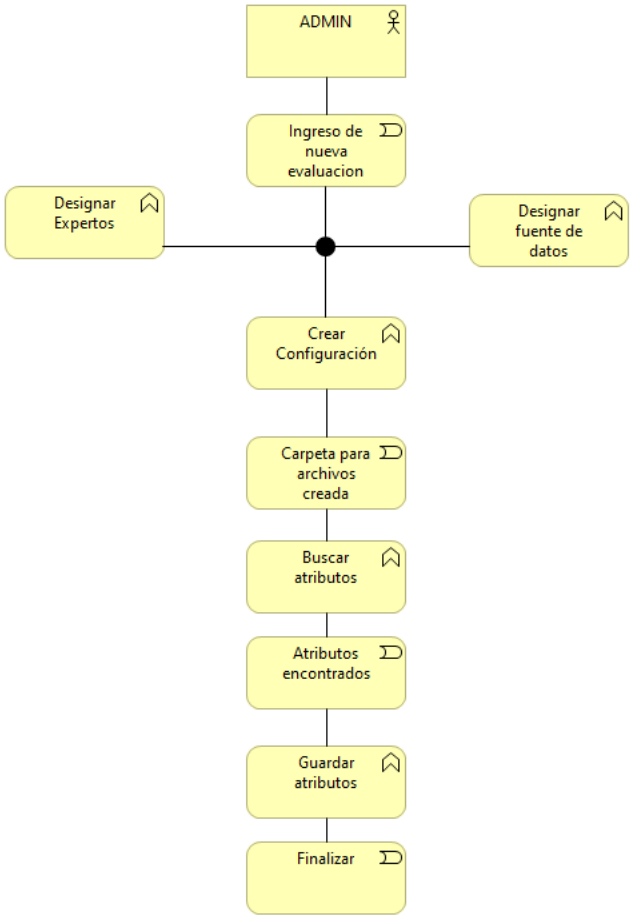


Figura 5.7: Proceso de evaluación realizado por el administrador

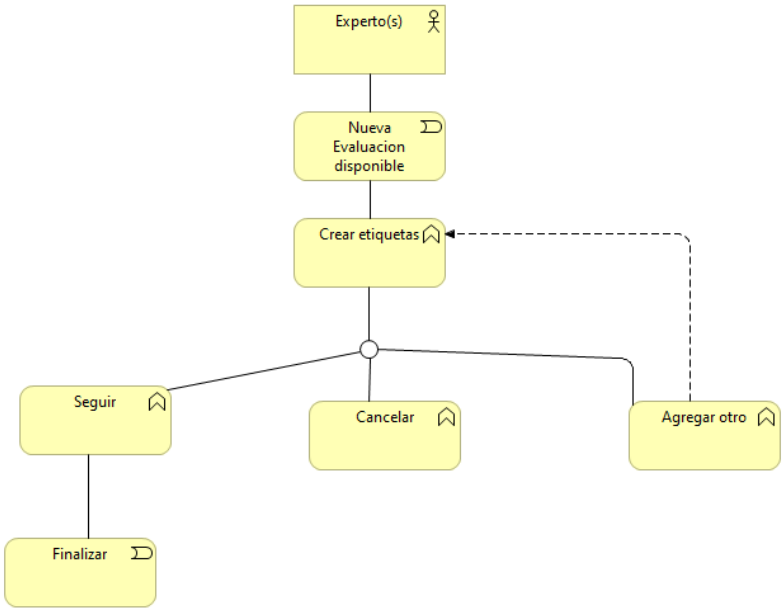


Figura 5.8: Proceso de evaluación realizado por el experto

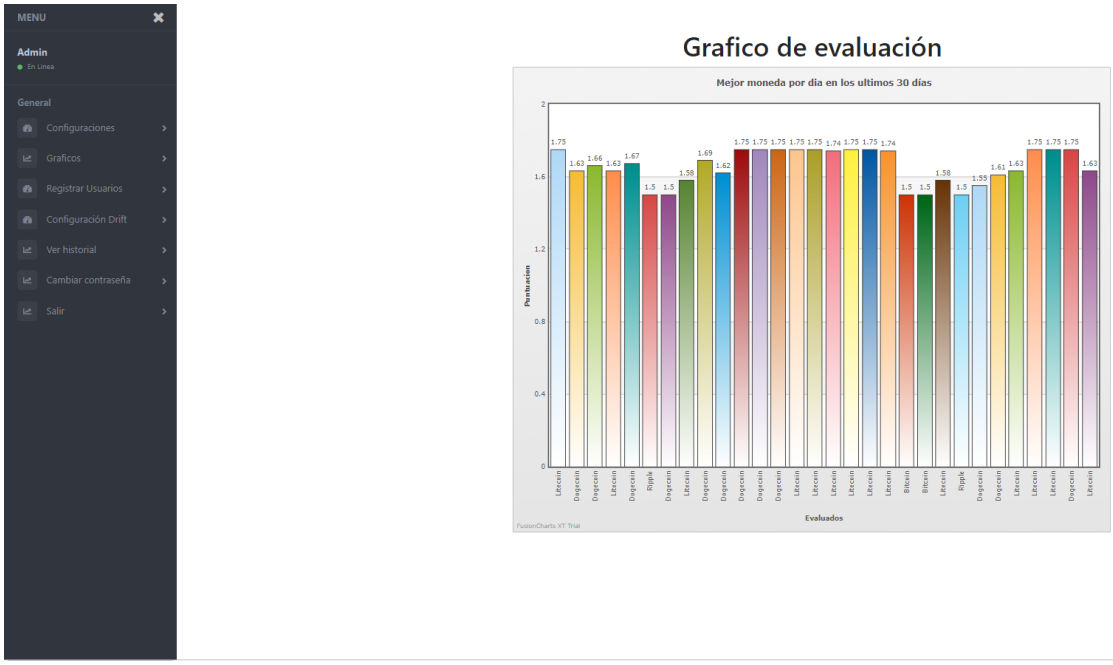


Figura 5.9: Gráfico resultante de una evaluación automática de 5 monedas distintas

5.3. Modelo de datos

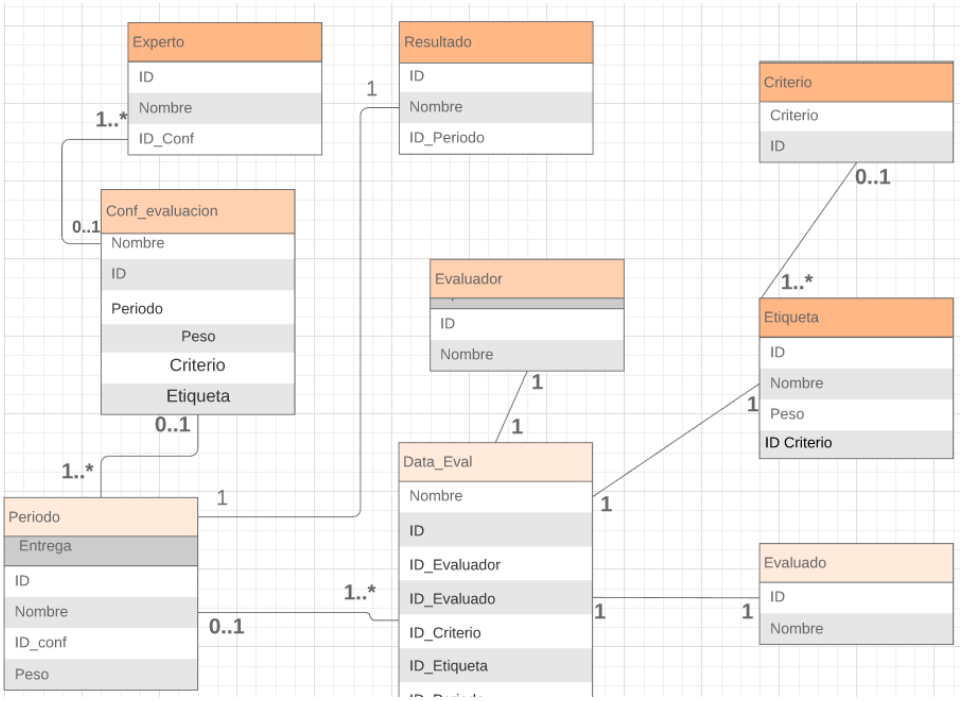


Figura 5.10: Modelo de datos

5.4. Desarrollo

Para todo esto se necesitó instalar Python 3.7 y Django 2.2.2. Para la instalación de django basta con instalarlo mediante el comando pip install django
Teniendo esto ya instalado a través de la terminal de windows se creó un proyecto django a través del comando django-admin startproject myproject, donde la frase myproject es el nombre del proyecto. Al ejecutar este comando se creará el proyecto en el directorio donde uno se encuentre al ejecutar el comando y se auto generará un código básico que consiste en las configuraciones básicas para crear un proyecto.
Ya que django no cuenta con una herramienta para graficar se buscaron APIS externas para la realización de esta tarea, eligiendo finalmente Fusioncharts. Para utilizar esta API en django primero que todo hay que descargar su librería desde la página https://www.fusioncharts.com/download/fusioncharts-suite-xt la cual descarga un archivo .rar que contiene todas las dependencias necesarias para utilizar esta API en django.
Habiendo descargado y extraído las dependencias necesitamos:

- Colocarlas dentro del proyecto para lo cual crearemos en el directorio raíz una carpeta con el nombre template en la cual se creará un archivo html con el nombre index en el que se referenciarán las dependencias y otra carpeta con el nombre static donde se copiará la carpeta fusioncharts la cual contiene los archivos necesarios para la creación de gráficos.

- Luego de haber incluido todas las dependencias en el proyecto necesitamos actualizar el proyecto con las nuevas carpetas con sus dependencias a través del comando python manage.py collectstatic. Y necesitamos agregar en el archivo de configuración del proyecto(setting.py) el PATH de los nuevos directorios quedando de la siguiente forma:

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.2/howto/static-files/

STATIC_URL = '/static/'
# STATIC_ROOT = "myproject/template/static"
STATICFILES_DIRS = [
    os.path.join(BASE_DIR, "myproject/templates/static"),
]
```

Figura 5.11: Settings necesario para el proyecto

- Teniendo todo configurado para la realización del gráfico hay que crear una aplicación y una base de datos para el proyecto por lo que en la terminal colocamos python manage.py startapp evaluación el cual creará una aplicación que será donde se realizarán las tareas, esta nueva aplicación debe ser referenciada en el archivo de configuración del proyecto llamado settings.py. Dentro de esta aplicación hay un archivo llamado models.py que es donde se creará la base de datos, por lo que debemos abrir el archivo y editarlo para crear nuestro modelo de datos, una clase tiene la siguiente forma:

```
class Eval_Data(models.Model):
    Nombre = models.CharField(max_length=50)

    Evaluador = models.OneToOneField(
        Evaluadores,
        on_delete=models.CASCADE,

    )

    Criterio = models.OneToOneField(
        Criterios,
        on_delete=models.CASCADE,

    )

    Etiqueta = models.OneToOneField(
        Etiquetas,
        on_delete=models.CASCADE,
    )

    Periodo = models.ForeignKey(Periodos,
on_delete=models.CASCADE)
    Etiqueta = models.ForeignKey(Etiquetas,
on_delete=models.CASCADE)
    Evaluado = models.ForeignKey(Evaluados,
on_delete=models.CASCADE)

    def __str__(self):
        return self.Nombre
```

Figura 5.12: Ejemplo de creación de modelo de datos

- Teniendo ya la base de datos creada necesitamos empezar a trabajar en la interfaz web de la aplicación para eso necesitamos crear una plantilla que será el menú que se repetirá en todas las páginas de la aplicación, donde para esto crearemos un archivo html llamado base.html el cual editado lucirá de la siguiente manera:

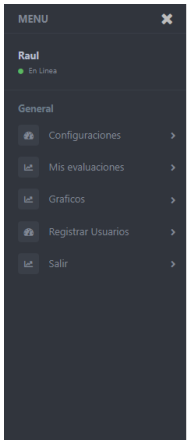


Figura 5.13: Template base que se reutilizará en todos los template de la aplicación

Ahora dentro de nuestra aplicación abriremos el archivo views que es el encargado de mostrarle al usuario de la aplicación los datos que nosotros queramos mostrarle por lo que dentro de este archivo crearemos las vistas de los procesos mencionados anteriormente quedando la vista de nueva evaluación manual de la siguiente manera:

Figura 5.14: Interfaz de nueva evaluación manual

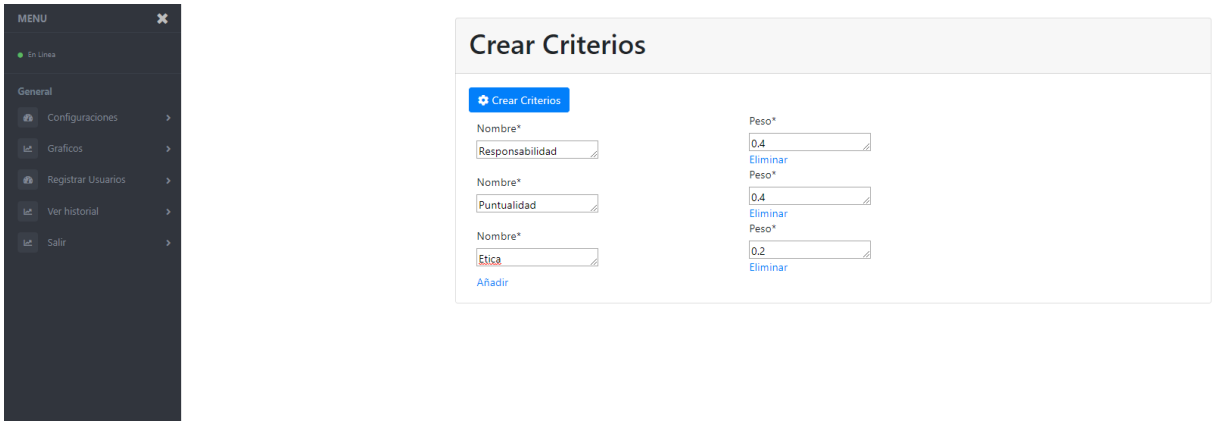


Figura 5.15: Interfaz de creación de criterios

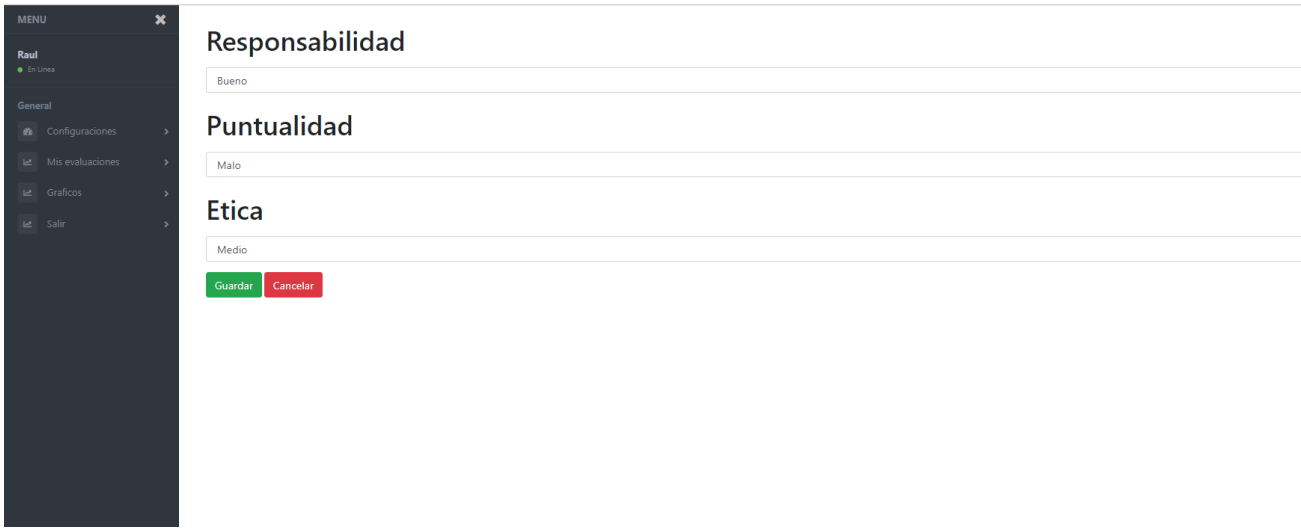


Figura 5.16: Interfaz de completar evaluación

Teniendo la base de datos ya con datos para poder realizar una evaluación, solo necesitamos crear y mostrar el gráfico a partir de esos datos, por lo que crearemos una carpeta llamada utils en el directorio de nuestro proyecto para dejar ahí todo nuestros archivos necesarios para la creación de una evaluación y dentro de esta crearemos el archivo init, el cual le dirá a nuestro proyecto que el directorio es parte del proyecto por lo que se pueden llamar a los archivos dentro de esta carpeta. Ahora que el directorio es parte del proyecto crearemos el archivo seleccion.py el cual se encargará de tomar los atributos de la base de datos a través de funciones para definir los criterios, los evaluados, los periodos, el peso de los periodos, las etiquetas y el peso de los criterios de cualquier evaluación. También se creará el archivo resultados.py el cual tomará estos atributos para llevarlos a una evaluación específica, donde se guardaran en variables los atributos de la evaluación para así poder usarlas en el archivo xu.py el cual se encarga de tomar todos estos atributos y realizar la evaluación cuyo resultado es guardado y enviado al archivo views.py el cual se encarga de tomar este resultado y graficarlo mediante el uso de la API fusioncharts, donde la puntuación de cada evaluado va en el eje Y los nombres de los evaluados en el eje X. Para poder llamar la API de fusioncharts basta con importarla en views.py de la siguiente manera from fusioncharts import FusionCharts, ya teniéndola importada necesitamos crear una lista llamada datasource donde dentro de esta lista irá primero el eje Y y luego el eje X. Ya teniendo los datos en la lista basta con crear un objeto FusionCHarts para la construcción del gráfico: column2D = FusionCharts(çolumn2d", .ex1", "600", "350", çhart-1", "json", dataSource), siendo los atributos de este objetos el tipo de gráfico, el tema del gráfico, su tamaño, el tipo de archivo que se enviará y los parámetros para los ejes del gráfico. Luego de esto regresamos el objeto renderizado de la siguiente manera: return render(request, 'index.html', 'output': column2D.render()) quedando el gráfico en pantalla de la siguiente manera:

```
from django.shortcuts import render
from django.http import HttpResponse

# Include the `fusioncharts.py` file that contains functions to embed the charts
from fusioncharts import FusionCharts

from polls.models import *

# The `chart` function is defined to generate Column 2D chart from database.
def chart(request):
    # Chart data is passed to the `dataSource` parameter, as dict, in the form o
    dataSource = {}
    dataSource['chart'] = {
        "caption": "Evaluacion",
            "subCaption": "PT",
            "xAxisName": "Alternativas",
            "yAxisName": "Aceptacion",
                 "theme": "zune"
        }

    # The data for the chart should be in an array where each element of the arr
    # having the `label` and `value` as key value pair.

    dataSource['data'] = []
    # Iterate through the data in `Revenue` model and insert in to the `dataSour
    for key in Revenue.objects.all():
      data = {}
      data['label'] = key.Month
      data['value'] = key.MonthlyRevenue
      dataSource['data'].append(data)

    # Create an object for the Column 2D chart using the FusionCharts class cons
    column2D = FusionCharts("column2D", "ex1" , "600", "350", "chart-1", "json",
    return render(request, 'index.html', {'output': column2D.render()})
```

Figura 5.17: Código necesario para graficar

Capítulo 6

Conclusiones

Como resultado de la investigación realizada es posible concluir que utilizando un algoritmo para controlar el drift, el cual mediante un mecanismo de aprendizaje-olvido y una estrategia explicita de detección de cambios controla el concepto drift podemos aumentar la estabilidad del proceso de toma de decisiones ya que esto minimizaría la fluctuación de las alternativas en el tiempo poniendo fin al problema de la toma de decisiones en un ambiente dinámico ya que ahora si nada de este ambiente seria un impedimento a la hora de tomar decisiones. Es por eso que actualmente se esta desarrollando una herramienta web que junta los modelos de múltiples criterios y múltiples periodos con información dudosa junto con el algoritmo para la detección de drift para mitigar el efecto del dinamismo del ambiente.

Bibliografía

[DRAP15] Gregory Ditzler, Manuel Roveri, Cesare Alippi, and Robi Polikar. Learning in nonstationary environments: A survey. IEEE Computational Intelligence Magazine, 10(4):12–25, 2015.

[GAGK18] I.A.D. Gunn, A. Arnaiz-González, and L.I. Kuncheva. A taxonomic look at instance-based stream classifiers. Neurocomputing, 286(1):167–178, 2018.

[GZ01] Adam E. Gaweda and Jacek M. Zurada. Data-driven design of fuzzy system with rational input partition. In 2001 IEEE International Conference on Fuzzy Systems, pages 610–613. IEEE, 2001.

[Hal06] Daniela Hall. Automatic parameter regulation of perceptual systems. Image and Vision Computing, 24(8):870 – 881, 2006.

[JJNH91] R. Jacobs, M. Jordan, S. Nowlan, and P. Hinton. Adaptive mixtures of local experts. Neural Computation, 3:79–81, 1991.

[Las02] M. Last. Online classification of nonstationary data streams. Intelligent Data Analysis, 6(2):129–147, 2002.

[LLZdM16] Ning Lu, Jie Lu, Guangquan Zhang, and Ramon Lopez de Mantaras. A concept drift-tolerant case-base editing technique. Artificial Intelligence, 230:108 – 133, 2016.

[McC13] J.L. McClelland. Integrating probabilistic models of perception and interactive neural networks: a historical and tutorial review. Frontiers in Psychology, 4(503):1–25, 2013.

[NK08] H.T. Nguyen and V. Kreinovich. Computing degrees of subsethood and similarity for interval-valued fuzzy sets: fast algorithms. In Proceedings of the 9th International Conference on Intelligent Technologies InTech, pages 47–55, 2008.

[PW14] Ding-Hong Peng and Hua Wang. Dynamic hesitant fuzzy aggregation operators in multi-period decision making. Kybernetes, 43(5):715–736, 2014.

[TBA13] Romina Torres, Nelly Bencomo, and Hernan Astudillo. Addressing the QoS drift in specification models of self-adaptive service-based systems. In Second International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE), ICSE 2013, 2013.

[Tor14] Romina Torres. Time-aware specifications to support runtime adaptation of service-based s PhD thesis, Universidad T'ecnica Federico Santa Mar'ia, August 2014.

[TSA14] Romina Torres, Rodrigo Salas, and Hernan Astudillo. Time-based hesitant fuzzy information aggregation approach for decision-making problems. International Journal of Intelligent Systems, 29(6):579–595, 2014.

[TSAM03] R. Torres, R. Salas, H. Allende, and C. Moraga. Robust expectation maximization learning algorithm for mixture of experts. Lecture Notes in Computer Science. Artificial Neural Nets Problem Solving Methods, 7th International Work-Conference on Artificial and Natural Neural Networks, IWANN2003, 2686:238–245, 2003.

[Wei12] G. Wei. Hesitant fuzzy prioritized operators and their application to multiple attribute decision making. Knowledge-Based Systems, 31:176–182, July 2012.

[WK96] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. Mach. Learn., 23(1):69–101, April 1996.

[Xu08a] Zeshui Xu. Linguistic aggregation operators: An overview. In Humberto Bustince, Francisco Herrera, and Javier Montero, editors, Fuzzy Sets and Their Extensions: Representation, Aggregation and Models, volume 220 of Studies in Fuzziness and Soft Computing, pages 163–181. Springer Berlin Heidelberg, 2008.

[Xu08b] Zeshui Xu. On multi-period multi-attribute decision making. Knowledge-Based Systems, 21(2):164–171, March 2008.

[XY08] Zeshui Xu and Ronald R. Yager. Dynamic intuitionistic fuzzy multi-attribute decision making. International Journal of Approximate Reasoning, 48(1):246 – 262, 2008. Special Section: Perception Based Data Mining and Decision Support Systems.

[Yag04] Ronald R. Yager. OWA aggregation over a continuous interval argument with applications to decision making. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 34(5):1952–1963, 2004.