Modelos de predicción del resultado de la trazabilidad de requerimientos en procesos de desarrollo de software

Juan Francisco Giró[1]

Resumen

La trazabilidad de requerimientos en los procesos de desarrollo de software reviste gran importancia, al igual que la necesidad de mejorar la comprensión de las condiciones que contribuyen a asegurar resultados exitosos. Estos aspectos han estimulado el desarrollo de modelos, conducentes a un mejor conocimiento del problema y a la posibilidad de anticipar los resultados que pueden esperarse en proyectos de diferentes tamaños y características. Para ello es necesario identificar los factores que tienen mayor impacto sobre los procesos de trazabilidad y proponer modelos que permitan hacer predicciones a partir de esos factores. En este trabajo se describen los factores adoptados y en base a los mismos se hace un análisis comparativo de tres modelos de predicción. Dos de estos modelos fueron evaluados con anterioridad, a través del análisis ROC (Característica Operativa del Receptor), a los que se suma un nuevo modelo implementado a través de una red neuronal multicapa de perceptrones. Se presentan y discuten los resultados obtenidos a partir de un caso de estudio, llegándose a la conclusión que los modelos, seguramente perfectibles, representan herramientas capaces de brindar diagnósticos muy útiles en la industria del software. Además, aún en el caso en que los modelos no llegaran a implementarse regularmente, el reconocimiento de los factores más sensibles en este tema permitirá señalar en forma anticipada los aspectos que deben ser tratados con mayor cuidado a fin de asegurar un proceso de desarrollo sólido y confiable.

Palabras clave: ingeniería de software, trazabilidad de requerimientos.

1 Docente Investigador Instituto Universitario Aeronáutico, Universidad Tecnológica Nacional (FRC).Argentina.

Abstract

The traceability of requirements in software development processes is of great importance, including the need to improve the understanding of the conditions that help to ensure successful results. These aspects have stimulated the development of models, leading to a better knowledge of the problem, and to enhance the ability to anticipate the expected results in projects of different sizes and characteristics. It is necessary to identify the factors that have major impact on traceability processes and to propose models to make predictions based on these factors. This paper describes the adopted factors and makes comparative analysis predictions using three different models. Two of those models were previously evaluated, through ROC analysis (Receiver Operating Characteristic), and a new model was implemented with a multilayer perceptron artificial neural network. The results obtained from a case of study are presented and discussed. The conclusion is that these models, although they are certainly perfectible, are capable of providing diagnostic tools very useful in the software industry. Moreover, even if these models were not regularly implemented, the recognition of the most sensitive factors in this topic will help to point out in advance the aspects that should be treated with utmost care to ensure a reliable development process.

Key words: software engineering, requirements traceability.

1. Introducción

La *ingeniería de software*, ideada con la finalidad de superar la llamada *crisis de software*, buscó desde sus comienzos respuestas tanto a los problemas específicos del desarrollo del producto software como a las actividades complementarias de apoyo. En este último grupo puede mencionarse, entre otras muchas, la planificación y gestión de proyectos, el aseguramiento de la calidad, la trazabilidad de requerimientos y la verificación y validación del producto. El objetivo fue incorporar al desarrollo de software las prácticas y herramientas habituales en las ingenierías convencionales para alcanzar productos de calidad dentro de plazos y presupuestos preestablecidos.

Entre las herramientas de apoyo pueden citarse los *modelos de estimación*, destinados a anticipar el esfuerzo requerido y el plazo de un proyecto cuando aún se sabe muy poco sobre las características del producto a ser desarrollado. Desafortunadamente, con la notable excepción de los COCOMO (COnstructive COst MOdel) [1], el valor estratégico de estos modelos ha hecho que muy pocos de ellos se difundan libremente en la comunidad informática.

A diferencia de lo ocurrido con los modelos de estimación, resulta curioso comprobar lo poco que se ha avanzado para anticipar la conveniencia de la trazabilidad de los requerimientos en los procesos de desarrollo. Se entiende por conveniencia a que están dadas las condiciones para su implementación y que será útil al proceso de desarrollo y/o económicamente beneficioso. Por otra parte, la *trazabilidad* comparte con la *estimación* su valor estratégico, lo que inhibe la amplia difusión de experiencias con valor práctico.

Para destacar la importancia de la trazabilidad cabe citar a Rubin y Goldberg [2] (1993): "un indicador fundamental de la madurez de una metodología de desarrollo está dado por la posibilidad de que sus resultados puedan ser siempre verificados y validados, siendo la trazabilidad la clave que permite alcanzar ese nivel de madurez".

Viniendo hacia nuestros días, ISO 9000.3 [3], MIL-STD-498 [4], IEEE 830 [5] y CMMI [6] designan la trazabilidad como un requisito esencial para el desarrollo de buen software. Sin embargo, a pesar de lo expuesto, hay múltiples evidencias de que los progresos en el campo de la trazabilidad de requerimientos de proyectos de desarrollo de software no siempre llegan a ser efectivamente aplicadas en la industria [7][8].

Habiéndose expuesto el problema, surge la presunción de que no es fortuito que un cierto proyecto pueda ser exitosamente trazado y otro no. Por el contrario, seguramente hay una combinación de condiciones, absolutamente determinísticas, que conducen a uno u otro resultado. De ser así, debería ser posible identificar los factores que condicionan un cierto proyecto de desarrollo de software para que pueda ser efectivamente respaldado por un sistema de trazabilidad.

Al plantearse como hipótesis la existencia de tales factores y la posibilidad de identificarlos, es necesario reconocer que los factores: *i)* serán cambiantes en el

tiempo, por estar vinculados al "estado del arte" de la ingeniería de software en cada momento y *ii)* serán la base de modelos que permitan anticipar la conveniencia o no de aplicar trazabilidad a cierto proyecto concreto en un momento dado.

Este artículo presenta tres modelos destinados a anticipar la conveniencia de trazar proyectos de software, tomando como base estudios realizados con ese fin [9], y está organizado de la siguiente manera: en la sección 2 se identifican y describen los factores que impactan la trazabilidad, en la sección 3 se detallan las características de los modelos propuestos, en la sección 4 se presenta un caso de estudio y los resultados obtenidos, en la sección 5 se analizan esos resultados, finalmente en la sección 6 se presentan las conclusiones del trabajo y se delinean actividades futuras.

2. Factores que condicionan la trazabilidad

Los factores que condicionan la trazabilidad deben estar asociados a las dimensiones de las tres entidades principales del problema, estrechamente relacionadas entre sí: a) el propio producto software, b) el proyecto y c) la organización. El proyecto responde a cierto modelo de proceso y la organización constituye el escenario donde se desarrolla el proyecto hasta que se completa el producto.

Para seleccionar los factores se fijaron criterios, que establecen que sus variables deben ser: i) cuantificables, ii) familiares para facilitar su relevamiento, iii) independientes entre sí, iv) asociables a valores en el rango [0 ; 5] mediante métricas apropiadas y v) crecientes cuando mejoran la trazabilidad. A partir de estas pautas se adoptaron ocho factores: cuatro asociados al producto, dos al proyecto y otros dos a la organización. Las entidades, factores, variables, variables primitivas, descripciones, métricas e intervalos de las variables primitivas son presentados en la Tabla 1.

Tabla 1: Definición de entidades, factores, variables, variables primitivas, métricas e intervalos.

Entidad	Factor	Variable	Variable primitiva de referencia *Vr* y descripción	Métrica *f(Vr)*	Intervalo de *Vr*
Producto	Tamaño	t	Tamaño del producto: Puntos de Función ***PF***	$t = 5*PF / 1000$	100 - 1000
	Vigencia	v	Expectativa de vida útil ***VU*** [años]	$v = 5*VU / 10$	0,5 - 10
	Reutilización	r	Previsión de futura reutilización ***RE*** [%]	$r = 1 + RE/20$	0 - 80
	Confiabilidad	c	Indicador de la confiabilidad exigida ***CO*** *	$c = 1 + 0{,}8*CO$	0 - 5
Proyecto	Plazo	p	Previsión de duración del proyecto ***DP*** [años]	$p = DP$	0 - 5
	Equipo	e	Efectividad y capacidad del equipo ***EF*** **	$e = EF$	1 - 5
Organización	Madurez	m	Nivel de madurez ***CMMI*** *** de la organización	$m = CMMI$	1 - 5
	Dependencia	d	Autonomía ***NA*** **** en adopción de normas	$d = NA$	1 - 5

En la Tabla 2 se presenta el significado de los valores asignados a las variables: confiabilidad *CO*, Efectividad *EF*, Madurez *CMMI* y Autonomía *NA*.

Tabla 2: Definición de los indicadores ***CO*** *, ***EF*** **, ***CMMI***** y ***NA*** ******

Confiabilidad *CO*	
0	No importante
1	Baja
2	Media
3	Alta
4	Muy alta
5	Absoluta

Efectividad *EF*	
1	Pobre
2	Baja
3	Media
4	Alta
5	Muy alta

Madurez *CMMI*	
1	Inicial
2	Gestionado
3	Definido
4	Cuant.Gestionad.
5	Optimizado

Autonomía *NA*	
1	Independiente
2	Normas propias
3	Normas clientes
4	Nor. casa matriz
5	Combina 3 y 4

En la referencia [10] se presentan detalles sobre la selección de los ocho factores y en la referencia [11] se evalúa la conveniencia de otras métricas alternativas a las presentadas, utilizándose el análisis ROC [12] para hacer una comparativa del desempeño de los diferentes casos e identificar el más conveniente.

3. Modelos de predicción de trazabilidad

Una vez que se han identificado los factores que impactan el problema estudiado y se les han asignado métricas para cuantificarlos, el paso siguiente es proponer, implementar y evaluar modelos que permitan anticipar la conveniencia de trazar un determinado proceso de desarrollo de software. A continuación se describen tres modelos, que posteriormente son comparados entre sí en base a un caso de estudio.

3.1 Modelo Vectorial "V"

Con el fin de reducir la dimensión del problema y facilitar la visualización de las poblaciones de datos se propuso la asignación de una variable a cada una de las tres entidades involucradas: el *producto* (η_1), el *proyecto* (η_2) y la *organización* ($\eta 3$). Para calcular el módulo resultante de cada una de las tres variables se tuvo en cuenta la independencia entre los factores definidos en la Tabla 1.

$$\eta_1 = \sqrt{\left(t^2 + v^2 + r^2 + c^2\right)} \quad , \quad \eta_2 = \sqrt{\left(p^2 + e^2\right)} \quad , \quad \eta_3 = \sqrt{\left(m^2 + d^2\right)} \qquad (1)$$

Luego, a partir de esta reducción de dimensiones surgió la idea de utilizar el módulo de la resultante de los ocho factores, en adelante denominado *"ρ"*, módulo del vector factores, como parámetro representativo o "indicador" de cada caso considerado:

$$\rho = \sqrt{\left(\eta_1^2 + \eta_2^2 + \eta_3^2\right)} = \sqrt{\left(t^2 + v^2 + r^2 + c^2 + p^2 + e^2 + m^2 + d^2\right)} \qquad (2)$$

Es importante reconocer el significado del indicador ρ, que representa el radio de una fracción de casquete esférico en el espacio de tres dimensiones, o en el hiperespacio de ocho. Asumiendo que existe un valor de ρ que separa los casos exitosamente trazables de los que no lo son; el objetivo será determinar el valor más apropiado de ρ, denominado ρ_c, que permita separar de la mejor forma las poblaciones de los proyectos no trazables de los trazables. Se trata de encontrar el radio ρ_c de una hipersuperficie que discrimina ambas poblaciones.

3.2 Modelo de Áreas "A"

Para definir este segundo modelo se optó por un "diagrama radar", esquema plano como el presentado en la Figura 1, con una cantidad de ejes radiales igual a la dimensión del problema, en este caso ocho, adoptándose como indicador de cada proyecto el área A encerrada por su polígono octogonal.

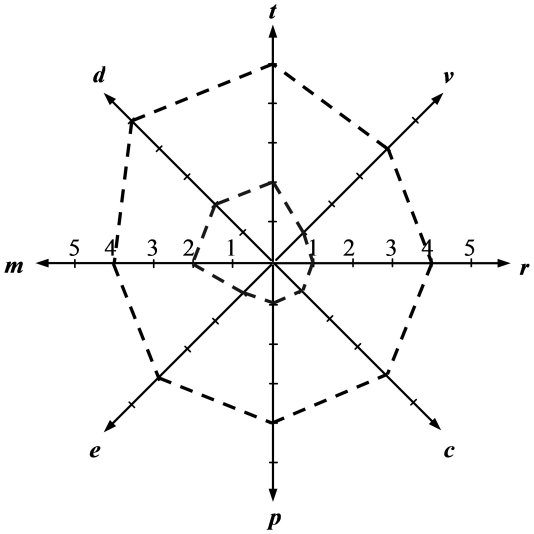


Figura 1: Gráfico de radar con la representación de polígonos de dos proyectos

Al haber ocho ejes y por lo tanto ocho triángulos con un ángulo interior de 45º, se suman los productos de cada uno de los ocho catetos (c_k) por las correspondientes alturas (c_{k-1} sen 45o) que están asociadas al cateto anterior. Teniendo en cuenta el orden cíclico resulta $c_0 = c_8$ y se obtiene la Ecuación (3): Aquí cada proyecto es representado por su indicador A, y se asume también que existirá Ac que separará las poblaciones de los proyectos trazables de los que no lo son.

$$A = \frac{1}{2}\sum_{k=1}^{8} c_k\, c_{k-1}\,\text{seno}(45) = \frac{\sqrt{2}}{4}(tv + vr + rc + cp + pe + em + md + dt) \quad (3)$$

Aquí cada proyecto es representado por su indicador A, y se asume también que existirá A_c que separará las poblaciones de los proyectos trazables de los que no lo son.

3.3 Modelo Neuronal "N"

La idea aquí es contar con un modelo regresivo que se apoye en los mismos ocho factores ortogonales ya descriptos y utilizados en los modelos anteriores. La intención es disponer de una expresión como la Ec. 4, que tenga como argumentos los ocho factores mencionados y que permita clasificar los casos en "trazables" (ξ=1) y "no trazables" (ξ=0):

$$\xi = g(t, v, r, c, p, e, m, d) = f(\eta_1, \eta_2, \eta_3) = \begin{cases} 1 \text{.....si es trazable} \\ 0 \text{ .....si no es trazable} \end{cases} \quad (4)$$

donde η_1, η_2, η_3 están definidos en la Ec. (1).

Se presume que (4) es una función no lineal y altamente compleja, desconociéndose la expresión matemática capaz de reproducir un comportamiento acorde a los datos provenientes de la industria referidos al tema tratado. Más aún, esta función no sólo debe ser establecida a partir de datos de un conjunto de casos, sino que debe ser progresivamente actualizada, ajustándola en la medida que se disponga de mayor cantidad de información. Lo expuesto condujo a adoptar un modelo neuronal multicapa de perceptrones, como el presentado esquemáticamente en la Figura 2, destinado a representar la función desconocida (Ec. 4) y posibilitar la evaluación de ξ.

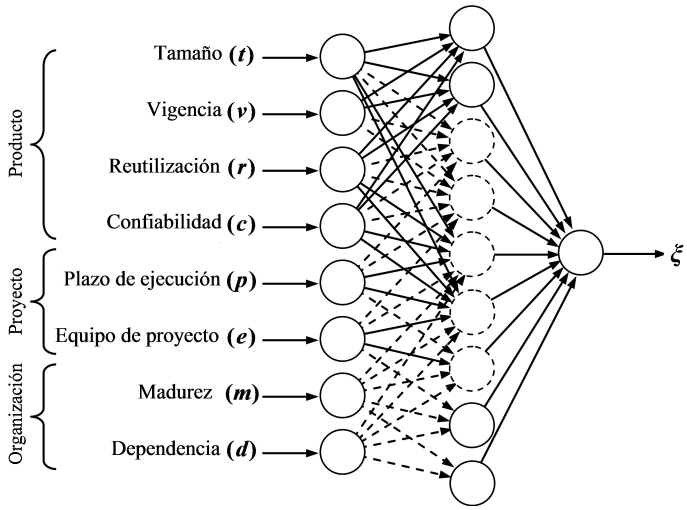


Figura 2: Esquema de una red multicapa de perceptrones

Al emplearse este tipo de modelo se puede imaginar una superficie compleja, definida en un espacio de tres dimensiones η_1, η_2, η_3, que discrimina entre las clases de casos trazables y no trazables. Esta superficie podría considerarse una generalización del casquete esférico que representa la superficie discriminante del modelo vectorial "**V**". Como se sabe, los modelos neuronales son apropiados para representar funciones de este tipo, para lo cual sus pesos sinápticos deben ser ajustados a partir de datos experimentales en un proceso definido como "de entrenamiento".

En estos casos el proceso de entrenamiento se realiza a través del método de "backpropagation" [13] y sus variantes, donde se ajustan progresivamente los pesos sinápticos del modelo según el error cometido en la salida, que es propagado hacia atrás a través de las sucesivas capas de unidades ocultas que componen la red. Hay que destacar que para este proyecto se hizo una implementación específica del método de "backpropagation", para posibilitar: una mejor comprensión del desempeño del modelo, su adecuación a necesidades particulares del problema específico y un estudio pormenorizado de los algoritmos involucrados.

Una vez seleccionada la herramienta, a partir de los factores considerados y del resultado esperado, quedó claro que se debe utilizar una red con ocho unidades de entrada y una unidad de salida. Sin embargo, no es posible prever ni la cantidad de capas de unidades ocultas necesarias, ni la composición de estas capas, ni el tipo de funciones de activación más conveniente para cada una. Esto debe inevitablemente ser realizado a través de un proceso de pruebas que incluye un análisis comparativo de los sucesivos resultados obtenidos con diferentes arquitecturas, hasta identificar la que exhiba el mejor desempeño. En efecto, uno de los puntos débiles de los modelos neuronales es la inexistencia de criterios definitivos que permitan seleccionar anticipadamente y en forma concluyente tanto la arquitectura como las propiedades más convenientes para la red.

Como paso previo al inicio del proceso de entrenamiento, la posterior validación y puesta en servicio de distintas variantes del modelo neuronal, se debe disponer de tres grupos de datos, que se describen a continuación.

3.3.a Datos de entrenamiento

Es un conjunto de pares de datos de entrada-salida disponible para "entrenar" el modelo, que constituye la referencia de lo que se desea que el modelo sea capaz de realizar. En el caso considerado cada par entrada-salida corresponde a un proyecto trazado y esta formado por un vector que contiene los valores específicos de los ocho factores y el correspondiente resultado del proceso de trazabilidad (1 = favorable / 0 = desfavorable).

3.3.b Datos de validación

Es otro conjunto de datos de entrada-salida, similar en su composición pero independiente del conjunto de entrenamiento. Su finalidad es comprobar que

el modelo neuronal tenga capacidad de generalización, es decir que sea capaz de responder correctamente ante datos con los que no ha sido "entrenado". En este caso se optó por definir el conjunto de validación a partir de promediar los factores homónimos de pares de casos de igual resultado, tomados del conjunto de entrenamiento y agrupados aleatoriamente.

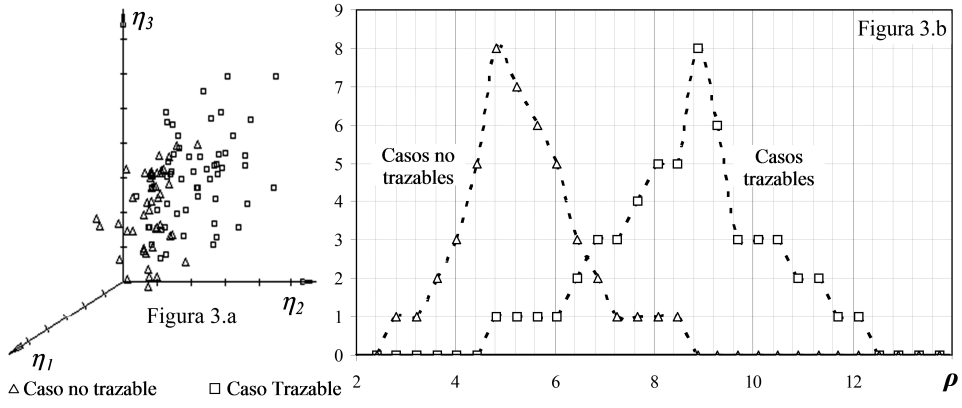
3.3.c Datos de prueba

Su finalidad es poner a prueba el modelo neuronal con casos de resultado desconocido, y comprobar que sean coherentes con los resultados obtenidos con otros dos modelos presentados con anterioridad. Para ello, se tomaron nuevamente datos del conjunto de entrenamiento y se hizo un promedio ponderado de los factores homónimos de pares de casos agrupados aleatoriamente, esta vez de distinto resultado.

Aquí es necesario aclarar que la validación mencionada en el punto "**b**" acompaña al proceso de entrenamiento, es decir se realiza por cada ciclo de ajuste de los pesos sinápticos. Esto es así para poder prevenir, a tiempo, que la red no sea "sobreentrenada". De lo contrario, el ajuste de pesos continuaría reduciendo el error en los puntos de valores conocidos y los incrementaría en los otros, perdiendo progresivamente su capacidad de generalización. Para evitar esta circunstancia adversa, durante todo el proceso de entrenamiento se observa la evolución de los valores medios cuadráticos de los errores obtenidos con los datos de entrenamiento y con los de validación.

4. Caso de estudio y resultados obtenidos

El "caso de estudio" utilizado en la evaluación de los modelos ya fue empleado anteriormente con fines similares [14]. Consiste en un lote de 102 muestras, que incluyen 55 proyectos trazados exitosamente (54%) y 47 proyectos con resultados negativos (46%), representados en los gráficos de las Figura 3.



Figuras 3: Representación de la población de casos disponibles en un sistema cartesiano ortogonal (3a) e histogramas de esos mismos datos (3b).

En la Figura 3.a se muestran los datos de los 102 casos sobre un sistema cartesiano ortogonal η_1, η_2, η_3, como el descripto con el modelo vectorial, y en la Figura 3.b los mismos datos como histogramas en función de $\boldsymbol{\rho}$, que es el módulo del vector de factores definido en la Ec. (2).

Llegado a este punto es necesario destacar una diferencia muy importante entre los modelos "V" y "A" con respecto al modelo "N" aquí implementado. Los dos modelos anteriores brindan como resultado un valor asociado a cada proyecto, un número real, habiendo un valor de corte ($\boldsymbol{\rho}_c$ o $\boldsymbol{A}_c$) que separa las poblaciones de los proyectos trazables de los que no lo son. El análisis ROC permite identificar el valor de corte más conveniente en cada caso, que brinda a cada modelo la mayor exactitud ε (cantidad de evaluaciones correctas sobre población total) y el más alto índice de Youden γ (tradicionalmente utilizado para comparar herramientas de diagnóstico). Para la evaluación global de los modelos se recurre al *coeficiente de Gini* $\boldsymbol{G}$ (propuesto por el italiano Corrado Gini como una medida de dispersión estadística), también determinable a partir del análisis ROC. Utilizando los datos del caso de estudio, en un trabajo anterior [14] se calcularon esos indicadores del desempeño de los modelos "V" y "A", que se muestran en la Tabla 3.

Tabla 3: Indicadores del desempeño de los Modelos "V" y "A"

Descripción del indicador	Modelo "V"	Modelo "A"
Valor de corte más conveniente	$\boldsymbol{\rho_c = 7{,}0}$	$\boldsymbol{A_c = 14{,}0}$
Máxima exactitud $\boldsymbol{\varepsilon}_{max}$ [%]	**90,20**	**89,22**
Máximo índice de Youden $\boldsymbol{\gamma}_{max}$	**0,8058**	**0,7814**
Coeficiente $\boldsymbol{G}$ de Gini	**0,9002**	**0,8344**

Los antecedentes expuestos habilitan el uso de estos dos modelos como una referencia apropiada para evaluar el desempeño del tercer modelo, denominado "**N**", que por tener una salida binaria (0 ó 1) directamente brinda como resultado un diagnóstico, que en el caso del problema tratado corresponde a "no trazable" o "trazable", ver Ec (4).

Para comenzar, y teniendo en cuenta que los valores $\boldsymbol{\rho}$ y $\boldsymbol{A}$ serán utilizados como referencia del modelo neuronal, resulta muy ilustrativo e interesante representar los 102 proyectos en un gráfico $\boldsymbol{\rho = f(A)}$ que es mostrado en la Figura 4.

Tal como fue anticipado en el punto 3.3.b, a partir de estos datos de entrenamiento se obtuvo el lote de datos de validación, que contiene 100 casos, 54 de ellos son proyectos trazados exitosamente y los otros 46 tienen resultados negativos.

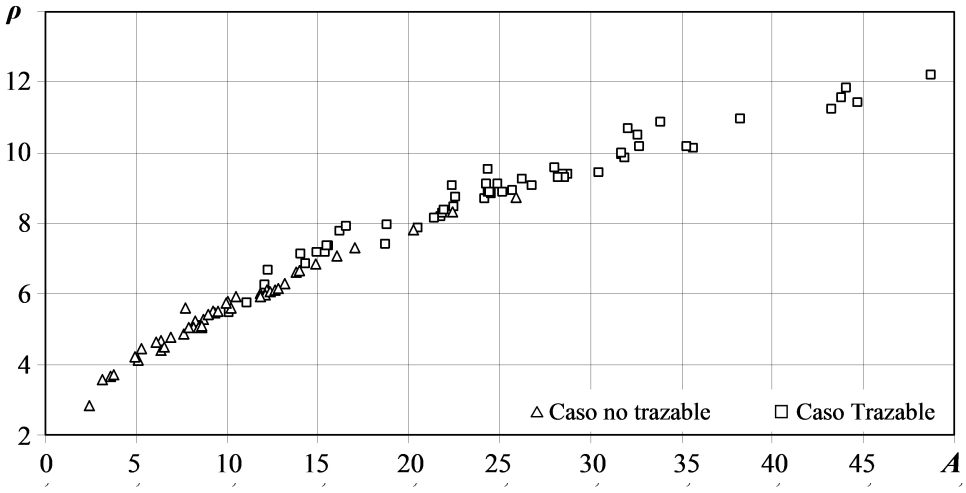


Figura 4. Lote de 102 datos de entrenamiento representados por ρ en función de A

Al igual que los datos de entrenamiento, los datos de validación son mostrados en la Figura 5 graficando la relación entre ρ y A. Nótese que en ambos casos las relaciones son no-lineales, en concordancia con la naturaleza de las expresiones (2) y (3) que las definen.

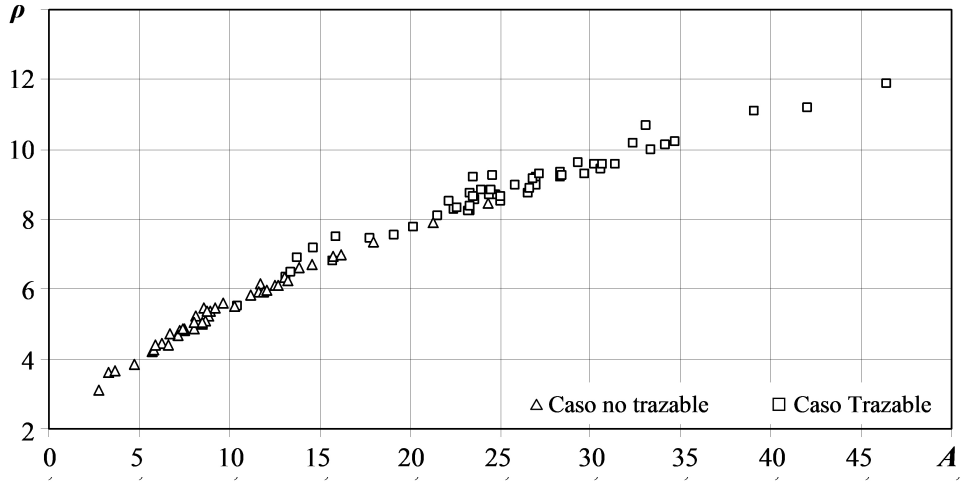


Figura 5. Lote de 100 datos de validación representados por el indicador ρ en función de A

Como ya se mencionó, los procesos de entrenamiento se realizaron con un software de desarrollo propio que se implementó como una herramienta integrada, que incluye opciones auxiliares de representación gráfica, almacenamiento y recuperación de datos. Como ejemplo, en la Figura 6 se muestra un gráfico de los

procesos de entrenamiento de dos configuraciones diferentes (11 y 21 unidades en la capa oculta). Se representan los errores de entrenamiento (curvas 1 y 2), errores de validación (curvas 3 y 4) y módulo del vector pesos de la red (curvas 5 y 6). Poder visualizar el módulo del vector pesos es muy conveniente ya que permite observar la evolución de este valor y confirmar la convergencia del proceso. Aquí debe destacarse que el gráfico no muestra un resultado final, sino la evolución progresiva de la magnitud de los errores y del módulo del vector pesos durante el proceso de entrenamiento, que en algunos casos puede demandar varios minutos. Esto permite advertir a tiempo procesos fallidos por ser oscilantes o divergentes, con el consiguiente ahorro de tiempo.

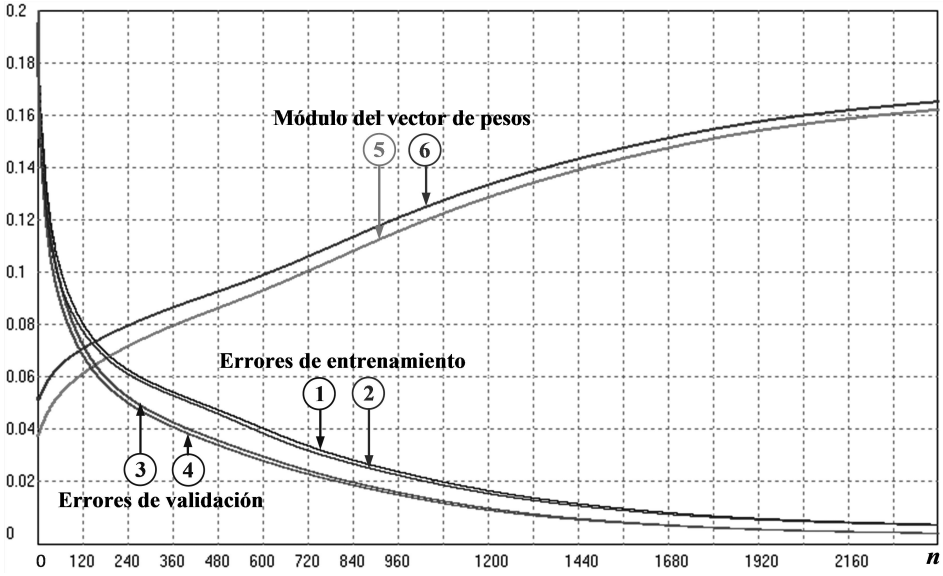


Figura 6. Representación del proceso de entrenamiento (ajuste de pesos) del modelo "N"

Debe tenerse en cuenta que el tiempo de proceso depende de la cantidad de pesos sinápticos de la red, de la cantidad "***n***" de ciclos de ajuste (en este caso fueron 2400 pero pueden llegar a ser cientos de miles), y de la población de datos. En efecto, al computarse el esfuerzo de cálculo se deben incluir todos los pares de datos entrada-salida, en este caso 102, por lo que la cantidad total de ajustes de pesos N en el caso mostrado del entrenamiento de una red con 21 unidades en la capa oculta y una unidad en la capa de salida fue de $N = [(22 \times 8) + 22] \times 102 \times 2400 = 48.470.400$.

La herramienta de entrenamiento desarrollada permite además visualizar los datos, los valores de los pesos, los resultados del entrenamiento y los resultados de la validación. En cuanto al propio proceso de entrenamiento, su desempeño depende de los valores del factor de aprendizaje α y de momento β [13], que

regulan su rapidez y deben ser convenientemente ajustados buscando un equilibrio, ya que valores altos conducen a procesos que se tornan inestables y valores bajos demandan más tiempo que el verdaderamente necesario. Aquí cabe destacar que la herramienta computacional desarrollada incluye la posibilidad de usar un factor α variable, que se ajusta en forma automática de acuerdo a la pendiente que tiene la curva error en cada momento. Por último, y con respecto a la red utilizada, es necesario señalar que en las unidades de la capa oculta se probaron funciones de activación tangencial y sigmoidal, optándose finalmente por la primera.

Una vez entrenado, el desempeño del modelo fue puesto a prueba con un lote de 44 casos obtenidos de acuerdo a lo descripto en el punto 3.3.c, que consistió en hacer promedios ponderados de los ocho factores de pares de casos con resultados opuestos y combinados aleatoriamente.

En la Figura 7 se representan, en un gráfico similar a los anteriores, los diagnósticos brindados por el modelo neuronal con respecto a la conveniencia (cuadrado) o no (triángulo) de someter a procesos de trazabilidad los proyectos considerados.

Sobre la misma figura se han mostrado los valores de corte ρ_c y A_c obtenidos con el análisis ROC [14], con el fin de ilustrar con un análisis comparativo el desempeño de los tres modelos.

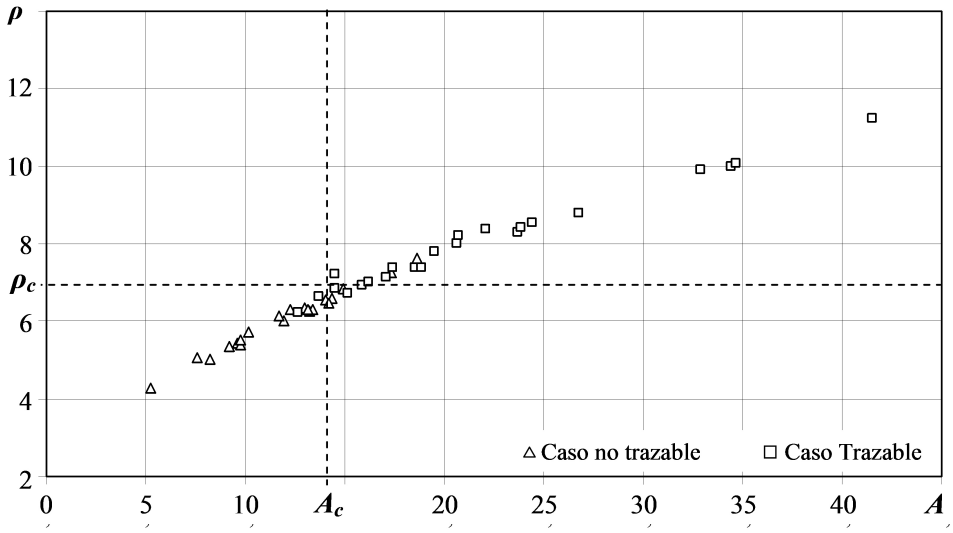


Figura 7. Diagnósticos asignados por el modelo neuronal "N" para los casos de prueba

5. Discusión de resultados

El desarrollo de los tres modelos de predicción de trazabilidad y las pruebas realizadas sobre ellos brindaron numerosas enseñanzas, que se presentan a continuación:

a) Los dos modelos más simples, que tienen por indicadores a $\boldsymbol{\rho}$ y $\boldsymbol{A}$, muestran desempeños similares en cuanto a la capacidad de predicción. Sin embargo, su simplicidad es aparente porque se requiere un proceso de análisis ROC para determinar los valores de corte $\boldsymbol{\rho}_c$ y $\boldsymbol{A}_c$ que hacen posible la clasificación.
b) Una ventaja del modelo de áreas "$\boldsymbol{A}$" es que el gráfico de radar resulta un buen complemento, al posibilitar un rápido reconocimiento visual de las condiciones que ofrece cada proyecto, lo que permite anticipar sus puntos más débiles.
c) El modelo neuronal "$\boldsymbol{N}$" es más complejo, y si bien normalmente es suficiente una única capa de unidades ocultas, la cantidad de estas unidades en algunos casos es crítica y está siempre asociada a ciertos valores de los factores de aprendizaje α y de momento β, que deben ser determinados. Con respecto a las funciones de activación, se obtuvieron los mejores desempeños utilizando funciones tangenciales en las unidades de la capa oculta y una función sigmoidal en la unidad de salida.
d) Observando los resultados de los 44 casos de prueba mostrados en la Figura 7, se comprueba que valores $\boldsymbol{A}>18$ ó $\boldsymbol{\rho}>7{,}5$ aseguran la conveniencia de trazar un proceso y valores $\boldsymbol{A}<12$ ó $\boldsymbol{\rho}<6{,}5$ aseguran lo contrario, mientras que para valores $12\leq\boldsymbol{A}\leq18$ o $6{,}5\leq\boldsymbol{\rho}\leq7{,}5$ se presentan tanto diagnósticos positivos como negativos. Esto es consecuencia de que la población de datos de entrenamiento no queda totalmente separada ni por $\boldsymbol{A}$, ni por $\boldsymbol{\rho}$, como lo pone en evidencia la Figura 4, dando lugar a falsos positivos y falsos negativos en los diagnósticos.
e) Es interesante comprobar en la Figura 7 que el modelo neuronal tiene capacidad de discriminación en la zona que los otros dos modelos no tienen, ya que emiten algunos diagnósticos incompatibles con los valores de corte de $\boldsymbol{\rho}$ y $\boldsymbol{A}$, aunque hasta ahora no se tiene evidencia de que el modelo neuronal discrimine siempre correctamente. Este importante tema debe ser motivo de un estudio más profundo.
f) Existen casos donde resulta laborioso encontrar las condiciones necesarias para un buen desempeño del modelo neuronal (convergencia y estabilidad en la evolución de los pesos sinápticos), ya sea porque es necesario explorar variantes en la arquitectura o en los factores de aprendizaje. Se ha comprobado que esto tiene dos causas: *i)* se trata de casos "atípicos" pero posibles o *ii)* hay inconsistencias en los valores de algunos de los factores. Esto significa que la necesidad de mayor esfuerzo para ajustar el modelo puede ser una advertencia y estar delatando problemas en la población de casos estudiados.

6. Conclusiones

El interés por desarrollar una herramienta capaz de anticipar la conveniencia de trazar proyectos de desarrollo de software llevó a identificar los factores determinantes para el éxito de la trazabilidad y desarrollar modelos apropiados para brindar el diagnóstico requerido. Para ello se propusieron factores que se fueron depurando progresivamente y se presentaron modelos de diferente complejidad y capacidad, hasta llegar a uno basado en una red multicapas de perceptrones. Los resultados presentados muestran que los modelos cumplen su misión con razonable eficacia y pueden ser considerados una buena respuesta al problema planteado. Su utilización regular en la industria del software permitirá continuar evaluándolos y perfeccionándolos, incluyendo el análisis crítico de los factores elegidos y las métricas adoptadas. Como mínimo, los factores cumplen una función orientativa señalando los aspectos de mayor impacto en un proceso de desarrollo sólido y confiable.

Como se mencionó en la introducción, el tema tratado tiene gran valor estratégico en las organizaciones, por lo que es muy probable que las ideas expuestas conduzcan al desarrollo de herramientas de uso interno, pero se observa poco interés en contribuir con información para compartir un modelo de diagnostico de uso público. De hecho, esto último fue comprobado durante la búsqueda de la información necesaria para conformar el caso de estudio de este trabajo. Los resultados obtenidos alientan la práctica de la trazabilidad en procesos de desarrollo de software, por lo que se continuará trabajando en perfeccionar las ideas expuestas, incluyendo lo mencionado en el punto e) de la discusión de resultados.

Referencias

Boehm B., Abts C., Brown A., Chulani S., Clark B., Horowitz E., Madachy R., Reifer D., Steece B.; Software Cost Estimation with COCOMO II, Prentice Hall, 2000.

Rubin, K., Goldberg, A.: Getting to Why. Journal of Object-Oriented Programming, Vol. 6, No. 4, July-Aug., pp. 5, 8-10, 13, 1993.

Kehoe, R., Jarvis A.; ISO 9000.3, A Tool for Software Product and Process Improvement, Springer-Verlag, 1996.

MIL-STD-498 (Military-Standard-498); Requirements for software development and documentation, 1994.

IEEE 830, Software requirements specifications, 1998.

Chrissis, M., Honrad, M., Shrum, S.: CMMI: Guía Para la Integración de Procesos y Mejora de Productos. 2a Edición, Pearson Educación, 2009.

Kannenberg, A., Saiedian, H.: Why Software Requirements Traceability Remains a Challenge. CrossTalk: The Journal of Defense Software Engineering. July/ August, 14-19, 2009.

Torkar R., Gorschek T., Feldt R., Svahnberg M., Raja U., Kamran K.; Requirements traceability: a systematic review and industry case study,; International Journal of Software Engineering and Knowledge Engineering, Vol. 22, No. 3, 1-49, 2012.

Giró, J., Vazquez, J., Meloni, B., Constable, L.; Expectativas y realidades de la trazabilidad de requerimientos en proyectos de desarrollo de software, M&Copias impresiones, Córdoba, 2014.

Giró, J., Vazquez, J., Meloni, B., Constable, L., Jornet, A.: Uso del Análisis ROC para anticipar la conveniencia de trazar proyectos de software. Workshop de Ingeniería de Software, CACIC 2012. Universidad Nacional del Sur, Ciudad de Bahía Blanca, 2012.

Giró, J., Vazquez, J., Meloni, B., Constable, L.; Evaluación de variantes en modelo destinado a anticipar la conveniencia de trazar proyectos de software, Workshop de Ingeniería de Software (WIS), CACIC 2013, CAECE, Ciudad de Mar del Plata, 2013.

Giró, J., Vazquez, J., Meloni, B., Constable, L.; Comparación de modelos destinados a anticipar la conveniencia de trazar proyectos de desarrollo de software, CoNaIISI 2013, Fac. Regional Córdoba UTN, Ciudad de Córdoba, 2013.