UNIVERSIDAD NACIONAL DE UCAYALI

FACULTAD DE INGENIERÍA DE SISTEMAS E INGENIERÍA CIVIL

CARRERA PROFESIONAL DE INGENIERÍA CIVIL

"SOFTWARE V&L VERSIÓN 1.0 – 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO - UCAYALI"

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE

INGENIERO CIVIL

GIAMFRANCO CHÁVEZ SIFUENTES

DANIEL ALCIDES VENTURA ROJAS

PUCALLPA – PERÚ

2020

UNIVERSIDAD NACIONAL DE UCAYALI

FACULTAD DE INGENIERÍA DE SISTEMAS E INGENIERÍA CIVIL

Carrera Profesional de Ingeniería Civil

INFORME DE ASESORÍA DE TESIS

1. Tesistas : Bach. CHÁVEZ SIFUENTES GIAMFRANCO.
   Bach. VENTURA ROJAS DANIEL ALCIDES.

2. Tesis : "SOFTWARE V&L VERSIÓN 1.0 - 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO - UCAYALI".

3. Referencia : Resolución de Comité de Planeamiento N° 458 /2017-CP-FIS y de IC-UNU.

Que, los tesistas han cumplido con ejecutar la tesis titulada: **"SOFTWARE V&L VERSIÓN 1.0 - 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO - UCAYALI"** de conformidad con el reglamento de grados y títulos de la Universidad Nacional de Ucayali, por lo que mi asesoría declara: **APROBADA,** en consecuencia se encuentra apta para ser presentada y evaluada por la Comisión de Grados y Títulos de la Facultad de Ingeniería de Sistemas e Ingeniería Civil de la Universidad Nacional de Ucayali.

Se expide el presente documento, a solicitud de los interesados para los fines consiguientes.

UNU, 06 de agosto del 2019.

Ing. Miguel Antonio Mendoza Mendoza
Asesor de Tesis

UNIVERSIDAD NACIONAL DE UCAYALI

FACULTAD DE INGENIERIA DE SISTEMAS E INGENIERIA CIVIL
ESCUELA PROFESIONAL DE INGENIERIA CIVIL
CARRERA PROFESIONAL DE INGENIERIA CIVIL
COMISION DE GRADOS Y TITULOS

ACTA DE EVALUACION DE SUSTENTACION DE TESIS N° 002-2020

GRADUANDO : **Bach. CHAVEZ SIFUENTES GIAMFRANCO**
Bach. VENTURA ROJAS DANIEL ALCIDES

TEMA : **"SOFTWARE V&L VERSION 1.0 – 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PUBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO"**

ASESOR : **ING. MIGUEL ANTONIO MENDOZA MENDOZA**

CRITERIOS	PUNTAJE			
	3	2	1	0
I- PRESENTACION				
* MOTIVACION	X			
* TONO DE VOZ		X		
* CALIDAD DE MATERIAL AUDIOVISUAL		X		
II- DESARROLLO DE CONTENIDO				
* SECUENCIA	X			
* DOMINIO DE VOZ		X		
* USO ADECUADO DE MATERIAL		X		
* VALIDEZ DE LOS HALLAZGOS	X			
III- ABSOLUCION DE PREGUNTAS				
* RESPONDE A LAS PREGUNTAS FORMULADAS POR EL JURADO	X			
IV- APORTE CIENTIFICO Y TECNOLOGICO				
* AL DESARROLLO DE LA COMUNIDAD	X			
* A LA PROFESION		X		

EVALUACION

- EXCELENCIA 26 A 30 PUNTOS
- UNANIMIDAD 21 A 25 PUNTOS
- MAYORIA 16 A 20 PUNTOS
- DESAPROBADO 15 A MENOS

CALIFICACION FINAL:
PUNTAJE : 25 PUNTOS

CONCLUSIONES:
APROBADO POR : UNANIMIDAD
DESAPROBADO POR :

JURADO EVALUADOR

- ING. MG. HORACIO SORIANO ALAVA — Presidente
- ING. MSC. EUCLIDES PANDURO PADILLA — Miembro
- ING. MG. WALTER GILBERTO ROMAN CLAROS — Miembro

SECRETARIO ACADÉMICO: ING. MSC. CLOTILDE RÍOS HIDALGO DE CERNA

PUCALLPA, 14 DE ENERO DEL 2020

"SOFTWARE V&L VERSIÓN 1.0 -2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO - UCAYALI "

Por:

Bach. CHÁVEZ SIFUENTES GIAMFRANCO.
Bach. VENTURA ROJAS DANIEL ALCIDES.

Aprobado:

Ing. Mg. HORACIO SORIANO ALAVA
Presidente

Ing. Msc. EUCLIDES PANDURO PADILLA
Miembro

Ing. Mg. WALTER GILBERTO ROMÁN CLAROS
Miembro

Asesor:

Ing. MIGUEL ANTONIO MENDOZA MENDOZA

UNIVERSIDAD NACIONAL DE UCAYALI
VICERRECTORADO DE INVESTIGACION
DIRECCION GENERAL DE PRODUCCION INTELECTUAL

Constancia

N° 477

ORIGINALIDAD DE TRABAJO DE INVESTIGACION SISTEMA ANTIPLAGIO URKUND

La Dirección General de Producción Intelectual, hace constar por la presente, que el Informe Final (Tesis), titulado:

SOFTWARE V&L VERSIÓN 1.0 – 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO - UCAYALI

Cuyo autor (es)	:	**CHÁVEZ SIFUENTES, GIAMFRANCO**
		VENTURA ROJAS, DANIEL ALCIDES
Facultad	:	INGENIERÍA DE SISTEMAS Y DE INGENERÍA CIVIL
Escuela Profesional	:	INGENIERÍA CIVIL
Asesor	:	ING. MENDOZA MENDOZA, MIGUEL ANTONIO

Después de realizado el análisis correspondiente en el Sistema Antiplagio URKUND, dicho documento presenta un **porcentaje de similitud de 07 %.**

En tal sentido, de acuerdo a los criterios de porcentaje establecidos en el artículo 9 de la DIRECTIVA DE USO DEL SISTEMA ANTIPLAGIO URKUND, el cual indica que no se debe superar el 10%. Se declara, que el trabajo de investigación: SI Contiene un porcentaje aceptable de similitud, por lo que SI se aprueba su originalidad.

En señal de conformidad y verificación se FIRMA Y SELLA la presente constancia.

Fecha: 09/12/2019



REPOSITORIO DE TESIS DE LA UNIVERSIDAD NACIONAL DE UCAYALI

AUTORIZACIÓN DE PUBLICACIÓN DE TESIS

Yo, Gianfranco Chávez Sifuentes

Autor de la TESIS titulada:

"Software V&L versión 1.0 - 2017 para desarrollar valorizaciones de obras publicas por contrata, en la Provincia de Coronel Portillo - Ucayali"

Sustentada el año: 2020

Con la asesoría de: Ing. Civil. Miguel Antonio Mendoza Mendoza

En la Facultad de: Ingeniería de Sistemas e Ingeniería Civil

Carrera Profesional de: Ingeniería civil

Autorizo la publicación de mi trabajo de investigación en el Repositorio Institucional de la Universidad Nacional de Ucayali, bajo los siguiente términos: Primero: otorgo a la Universidad Nacional de Ucayali licencia no exclusiva para reproducir, distribuir, comunicar, transformar (únicamente mediante su traducción a otros idiomas) y poner a disposición del público en general mi tesis (incluido el resumen) a través del Repositorio Institucional de la UNU, en forma digital sin modificar su contenido, en el Perú y en el extranjero; por el tiempo y las veces que considere necesario y libre de remuneraciones. Segundo: declaro que la tesis es una creación de mi autoría y exclusiva titularidad, por tanto me encuentro facultado a conceder la presente autorización, garantizando que la tesis no infringe derechos de autor de terceras personas. Tercero: autorizo la publicación,

- [x] Total (significa que todo el contenido de la tesis en PDF será compartido en el repositorio).
- [ ] Parcial (significa que solo la carátula, la dedicatoria y el resumen en PDF serán compartidos en el repositorio).

De mi TESIS de investigación en la página web del Repositorio Institucional de la UNU.

En señal de conformidad firma la presente autorización.

Fecha: 20 / 01 / 2020

Email: gianfranchochavez17@gmail.com

Firma:

Teléfono: 912 229 455

DNI: 70750591

REPOSITORIO DE TESIS DE LA UNIVERSIDAD NACIONAL DE UCAYALI

AUTORIZACIÓN DE PUBLICACIÓN DE TESIS

Yo, Daniel Alcides Ventura Rojas

Autor de la TESIS titulada:

"Software V&L versión 1.0 - 2017 para desarrollar valorizaciones de obras públicas por contrata, en la provincia de Coronel Portillo - Ucayali"

Sustentada el año: 2020

Con la asesoría de: Ing. Miguel Antonio Mendoza Mendoza

En la Facultad de: Ingeniería de Sistemas e Ingeniería Civil

Carrera Profesional de: Ingeniería Civil

Autorizo la publicación de mi trabajo de investigación en el Repositorio Institucional de la Universidad Nacional de Ucayali, bajo los siguiente términos: Primero: otorgo a la Universidad Nacional de Ucayali licencia no exclusiva para reproducir, distribuir, comunicar, transformar (únicamente mediante su traducción a otros idiomas) y poner a disposición del público en general mi tesis (incluido el resumen) a través del Repositorio Institucional de la UNU, en forma digital sin modificar su contenido, en el Perú y en el extranjero; por el tiempo y las veces que considere necesario y libre de remuneraciones. Segundo: declaro que la tesis es una creación de mi autoría y exclusiva titularidad, por tanto me encuentro facultado a conceder la presente autorización, garantizando que la tesis no infringe derechos de autor de terceras personas. Tercero: autorizo la publicación,

- [x] Total (significa que todo el contenido de la tesis en PDF será compartido en el repositorio).
- [ ] Parcial (significa que solo la carátula, la dedicatoria y el resumen en PDF serán compartidos en el repositorio).

De mi TESIS de investigación en la página web del Repositorio Institucional de la UNU.

En señal de conformidad firma la presente autorización.

Fecha: 20 / 01 / 2020

Email: danielalcides ventura rojas @ gmail. com

Firma:

Teléfono: 931591423

DNI: 76801071

DEDICATORIA

A Dios,

Por habernos permitido llegar hasta este punto y darnos la salud necesaria para lograr nuestros objetivos.

A nuestros padres, que son el motivo y razón de seguir adelante, por sus consejos, valores y motivaciones, que nos permiten ser personas de bien, pero sobre todo, por su amor infinito que nos brindan día a día.

A los docentes, que fueron educadores y formadores en las aulas universitarias, y que día a día nos brindaron sus conocimientos, para ser profesionales competentes.

AGRADECIMIENTO

Al Ing. Miguel Antonio Mendoza Mendoza, por su apoyo y asesoramiento académico.

Al Ing. Jorge Alarcón Vásquez, por su invalorable apoyo en proveernos toda la información técnica y experiencia necesaria en el rubro de las valorizaciones.

Al Ing. Ronald Robles Rimac, por su apoyo en el tema de Ingeniería de Software.

Al Dr. Alfonso Chávez Guillen, por la enseñanza en metodología de investigación, gracias a dichos profesionales se dio el desarrollo de la presente tesis.

RESUMEN

La presente tesis se enfoca en el diseño e implementación de un software para el desarrollo de las valorizaciones de obra principal, obra adicional, mayores gastos generales variables e intereses, con un interfaz amigable, resultados óptimos y un corto tiempo en su proceso.

El objetivo principal de esta investigacion, es crear un nuevo software como alternativa de solución para el ahorro de tiempo cuando se desarrollen valorizaciones de obras publicas por contrata, en la provincia de Coronel Portillo – Ucayali.

Los tipos de investigación utilizados para la presente tesis son: Investigación aplicada, tecnológica, cualitativa y cuantitativa; y el nivel de investigación utilizado es de tipo descriptivo.

De acuerdo al análisis de las preguntas planteadas en las encuestas y cálculos realizados en el capítulo V, la media de los indicadores en la fase de post test es mayor que la media en la fase de pre test, es decir, 4.51116 > 2.83705, en consecuencia, se concluye que la hipótesis general planteada es aceptada.

Finalmente, la investigación demuestra que con el diseño e implementación del software V&L Versión 1.0 – 2017, se logran resultados óptimos, asímismo, se comprueban las hipótesis del proyecto utilizando el diseño de investigación en sucesión o en línea, conocido también como pre-test y post-test con un solo grupo, concluyéndose que existe una mejora importante en el desarrollo de los distintos tipos de valorizaciones, siendo el software una alternativa de solución adecuada.

Palabras claves: Valorización, Software, RUP, UML, Java, MySQL.

ABSTRACT

This thesis focuses on the design and implementation of software for the development of the valuations of main work, additional work, higher variable general expenses and interests, with a friendly interface, optimal results and a short time in its process.

The main objective of this research is to create a new software as an alternative solution for saving time when public works valuations are developed by contract, in the province of Coronel Portillo - Ucayali.

The types of research used for this thesis are: Applied, technological, qualitative and quantitative research; and the level of research used is descriptive.

According to the analysis of the questions posed in the surveys and calculations made in Chapter V, the average of the indicators in the post-test phase is greater than the average in the pre-test phase, that is, 4.51116 > 2.83705, in Consequently, it is concluded that the general hypothesis proposed is accepted.

Finally, the research shows that with the design and implementation of the V&L Version 1.0 - 2017 software, optimal results are achieved, and the hypotheses of the project are checked using the research design in succession or online, also known as pre-test and Post-test with a single group, concluding that there is a significant improvement in the development of the different types of valuations, the software being an adequate solution alternative.

Keywords: Valorization, Software, RUP, UML, Java, MySQL.

ÍNDICE DE CONTENIDO

DEDICATORIA viii
AGRADECIMIENTO ix
RESUMEN x
ABSTRACT xi
ÍNDICE DE CONTENIDO xii
ÍNDICE DE CUADROS xvii
ÍNDICE DE IMÁGENES xix
ÍNDICE DE DIAGRAMAS xxi
ÍNDICE DE TABLAS xxiii
CAPÍTULO I : PLANTEAMIENTO DEL PROBLEMA 1
1.1 DESCRIPCIÓN Y FUNDAMENTACIÓN DEL PROBLEMA 1
1.2 FORMULACIÓN DEL PROBLEMA 2
1.2.1. PROBLEMA GENERAL.....2
1.2.2. PROBLEMA ESPECÍFICOS.....2
1.3 OBJETIVOS 3
1.3.1. OBJETIVO GENERAL.....3
1.3.2. OBJETIVO ESPECÍFICOS.....3
1.4 JUSTIFICACIÓN E IMPORTANCIA 3
1.5 LIMITACIONES Y ALCANCES.....4
1.6 HIPÓTESIS.....5
1.6.1. HIPÓTESIS GENERAL 5
1.6.2. HIPÓTESIS ESPECÍFICOS 5
1.7 SISTEMA DE VARIABLES, DIMENSIONES E INDICADORES.....5
1.7.1. VARIABLE INDEPENDIENTE.....5
1.7.2. VARIABLE DEPENDIENTE.....5

CAPÍTULO II: MARCO TEÓRICO 6

2.1 ANTECEDENTES O REVISIÓN DE ESTUDIOS REALIZADOS 6

2.2 MARCO SITUACIONAL O DE REFERENCIA 10

2.3 ADMINISTRACIÓN DE CONTRATOS DE OBRAS PÚBLICAS 11

2.3.1. TIPOS DE EJECUCIÓN PRESUPUESTAL 11

2.3.2. SISTEMAS DE CONTRATACIÓN 11

2.3.3. MODALIDADES DE CONTRATACIÓN 12

2.3.4. MÉTODOS DE CONTRATACIÓN 13

2.3.5. INICIO DEL PLAZO DE EJECUCIÓN DE OBRA 14

2.3.6. ADELANTOS PARA OBRAS 15

2.3.7. VALORIZACIONES 16

2.3.8. TIPOS DE VALORIZACIONES 18

2.3.9. FÓRMULAS POLINÓMICAS 19

2.3.10. NORMAS PARA EL CÁLCULO DE REAJUSTES 25

2.3.11. VALORIZACIÓN DE OBRA ADICIONAL 34

2.3.12. VALORIZACIÓN DE MAYORES GASTOS GENERALES VARIABLES 39

2.3.13. VALORIZACIÓN DE INTERESES 42

2.4 INGENIERÍA DE SOFTWARE 43

2.4.1. PROCESO UNIFICADO DE RATIONAL (RUP) 45

2.4.2. LENGUAJE UNIFICADO DE MODELADO (UML) 49

2.4.3. PROGRAMACIÓN ORIENTADA A OBJETOS 56

2.4.4. NETBEANS 59

2.4.5. JAVA (LENGUAJE DE PROGRAMACIÓN) 60

2.4.6. phpMyAdmin 60

2.4.7. BASE DE DATOS (MySQL) 61

CAPÍTULO III: METODOLOGÍA DE LA INVESTIGACIÓN 62

3.1 METODOLOGÍA Y TÉCNICAS UTILIZADAS 62

3.1.1. TIPO Y NIVEL DE INVESTIGACIÓN 62

3.1.2. NIVEL DE INVESTIGACIÓN 62

3.1.3. MÉTODO DE INVESTIGACIÓN....63
3.1.4. DISEÑO DE INVESTIGACIÓN....63
3.1.5. TÉCNICAS DE RECOLECCIÓN Y TRATAMIENTO DE DATOS 64

3.2 POBLACIÓN Y MUESTRA 65

3.2.1. POBLACIÓN 65
3.2.2. MUESTRA....65

3.3 OBRAS SELECCIONADAS PARA LA APLICACIÓN DEL SOFTWARE ... .65

CAPÍTULO IV: SOFTWARE PROPUESTO PARA EL CÁLCULO DE VALORIZACIONES EN OBRAS PÚBLICAS …… 75

4.1 FASE INICIAL 75

A. Documento Visión del Negocio....75
A.1. Introducción 75
A.2. Descripción de los Stakeholders y usuarios 76
A.3. Objetivos de Modelamiento del Negocio....79
A.4. Requerimientos....80
B. Plan de Desarrollo del Software....84
B.1. Introducción 84
B.2. La organización del proyecto 85
B.3. El proceso de Dirección 85
B.4. Recurso para el proyecto....86
B.5. Presupuesto....87
B.6. Vistas Use case 87

4.2 FASE DE ELABORACIÓN....90

4.2.1. Requerimientos 90

A) Modelo Use Case....90
B) Diagrama de Actividades 92

4.2.2. Análisis y Diseño....93

A) Diagrama de Colaboración 93
B) Diagrama de Paquete de Análisis 95
C) Diseño de la Arquitectura 96
D) Diseño de Interfaces y Diagrama de Secuencias 96
E) Diagrama de Clases 107
F) Diagrama de Base de Datos 108
G) Diccionario de Base de Datos 109
4.3 FASE DE CONSTRUCCIÓN 123
4.3.1. Codigo fuente del Sistema V&L 123
4.3.2. Pruebas 230
A) Nivel de calidad 230
B) A nivel de estado 233
C) A nivel de prueba de tipo 233
CAPÍTULO V: RESULTADOS Y DISCUSIÓN 234
5.1 RESULTADOS DEL CONTEO PARA EL PRE TEST O PRUEBA DE ENTRADA 234
5.2 RESULTADOS DEL CONTEO PARA EL POST TEST O PRUEBA DE SALIDA 241
5.3 CÁLCULO DE LA DESVIACIÓN ESTANDAR Y VARIANZA PARA EL PRE TEST 248
5.4 CÁLCULO DE LA DESVIACIÓN ESTANDAR Y VARIANZA PARA EL POST TEST 249
5.5 INTERPRETACIÓN DE RESULTADOS PARA EL PRE TEST 250
5.6 INTERPRETACIÓN DE RESULTADOS PARA EL POST TEST 252
5.7 PRUEBA DE HIPÓTESIS 253
5.7.1. FORMULACIÓN DE CUADRO DE VALORES DE INDICADORES 253
5.7.2. SUPUESTOS DE LA PRUEBA DE HIPÓTESIS 254
5.7.3. CÁLCULO DEL VALOR CRÍTICO Y FUNCIÓN PRUEBA 255
5.7.4. HIPÓTESIS 257

CONCLUSIONES **260**

SUGERENCIAS **262**

REFERENCIAS BIBLIOGRÁFICAS **263**

MATRIZ DE CONSISTENCIA **266**

ANEXOS…… **267**

ÍNDICE DE CUADROS

Cuadro N° 01 : Plantel técnico de obra 65
Cuadro N° 02 : Exposición del Problema 75
Cuadro N° 03 : Sumario de Stakeholder 77
Cuadro N° 04 : Sumario de usuarios 77
Cuadro N° 05 : Necesidades Principales del usuario 78
Cuadro N° 06 : Entregables del proyecto 84
Cuadro N° 07 : Plan de proceso de desarrollo de acuerdo a fases 86
Cuadro N° 08 : Principales tareas del proyecto 86
Cuadro N° 09 : Presupuesto del proyecto 87
Cuadro N° 10 : Prueba de Carga. 233
Cuadro N° 11 : Frecuencia de la pregunta N° 01 234
Cuadro N° 12 : Frecuencia de la pregunta N° 02 234
Cuadro N° 13 : Frecuencia de la pregunta N° 03 235
Cuadro N° 14 : Frecuencia de la pregunta N° 04 235
Cuadro N° 15 : Frecuencia de la pregunta N° 05 236
Cuadro N° 16 : Frecuencia de la pregunta N° 06 236
Cuadro N° 17 : Frecuencia de la pregunta N° 07 237
Cuadro N° 18 : Frecuencia de la pregunta N° 08 237
Cuadro N° 19 : Frecuencia de la pregunta N° 09 238
Cuadro N° 20 : Frecuencia de la pregunta N° 10 238
Cuadro N° 21 : Frecuencia de la pregunta N° 11 239
Cuadro N° 22 : Frecuencia de la pregunta N° 12 239
Cuadro N° 23 : Frecuencia de la pregunta N° 13 240
Cuadro N° 24 : Frecuencia de la pregunta N° 14 240
Cuadro N° 25 : Frecuencia de la pregunta N° 01 241
Cuadro N° 26 : Frecuencia de la pregunta N° 02 241

Cuadro N° 27 : Frecuencia de la pregunta N° 03 242
Cuadro N° 28 : Frecuencia de la pregunta N° 04 242
Cuadro N° 29 : Frecuencia de la pregunta N° 05 243
Cuadro N° 30 : Frecuencia de la pregunta N° 06 243
Cuadro N° 31 : Frecuencia de la pregunta N° 07 244
Cuadro N° 32 : Frecuencia de la pregunta N° 08 244
Cuadro N° 33 : Frecuencia de la pregunta N° 09 245
Cuadro N° 34 : Frecuencia de la pregunta N° 10 245
Cuadro N° 35 : Frecuencia de la pregunta N° 11 246
Cuadro N° 36 : Frecuencia de la pregunta N° 12 246
Cuadro N° 37 : Frecuencia de la pregunta N° 13 247
Cuadro N° 38 : Frecuencia de la pregunta N° 14 247
Cuadro N° 39 : Cálculo de la desviación estándar y varianza del Pre Test..... 248
Cuadro N° 40 : Cálculo de la desviación estándar y varianza del Post Test 249
Cuadro N° 41 : Resultados de la Estadística Descriptiva del Pre Test..... 250
Cuadro N° 42 : Resultados de la Estadística Descriptiva del Post Test 252
Cuadro N° 43 : Resumen de Resultados Finales del Pre Test y Post Test..... 255

ÍNDICE DE IMÁGENES

Imagen 2.1 : Topes para procedimientos de selección para el año fiscal 2019 14
Imagen 2.2 : Relación de Índices Unificados 22
Imagen 2.3 : Modelo de Índices Unificados de Precios 23
Imagen 2.4 : Ejemplo de aplicación de los coeficientes de reajustes 26
Imagen 2.5 : Capas de la Ingeniería de Software 44
Imagen 2.6 : Tipos de diagramas en UML 2.5 50
Imagen 2.7 : Ejemplo de diagrama de clases 51
Imagen 2.8 : Ejemplo de diagrama de componentes 51
Imagen 2.9 : Ejemplo de diagrama de despliegue 52
Imagen 2.10 : Ejemplo de diagrama de objetos 52
Imagen 2.11 : Ejemplo de diagrama de paquetes 53
Imagen 2.12 : Ejemplo de diagrama de estructura compuesta 53
Imagen 2.13 : Ejemplo de diagrama de actividades 54
Imagen 2.14 : Ejemplo de diagrama de casos de uso 54
Imagen 2.15 : Ejemplo de diagrama de secuencia 55
Imagen 2.16: Ejemplo de diagrama de comunicación 55
Imagen 3.1 : Ubicación de la Obra N° 01 – AA.HH. Guadalupe 66
Imagen 3.2 : Ficha Técnica de la Obra N° 01 – AA.HH. Guadalupe 67
Imagen 3.3 : Ubicación de la Obra N° 02 – AA.HH. Isabel La Católica 67
Imagen 3.4 : Ficha Técnica de la Obra N° 02 – AA.HH. Isabel La Católica 68
Imagen 3.5 : Ubicación de la Obra N° 03 – AA.HH. Juan Silva Bocanegra 68
Imagen 3.6 : Ficha Técnica de la Obra N° 03 – AA.HH. Juan Silva Bocanegra 69
Imagen 3.7 : Ubicación de la Obra N° 04 – AA.HH. La Perla 69
Imagen 3.8 : Ficha Técnica de la Obra N° 04 – AA.HH. La Perla 70
Imagen 3.9 : Localización de la Obra N° 05 – CC.NN. Nuevo San Juan 70
Imagen 3.10 : Ubicación de la Obra N° 05 – CC.NN. Nuevo San Juan 71

Imagen 3.11 : Ficha Técnica de la Obra N° 05 – CC.NN. Nuevo San Juan 71
Imagen 3.12 : Ubicación de la Obra N° 06 – Afirmado en el Jr. Venecia 72
Imagen 3.13 : Ficha Técnica de la Obra N° 06 – Afirmado en el Jr. Venecia..... 72
Imagen 3.14 : Ubicación de la Obra N° 07 – Caserío San Lorenzo 73
Imagen 3.15 : Ficha Técnica de la Obra N° 07 – Caserío San Lorenzo..... 73
Imagen 3.16 : Ubicación de la Obra N° 08 – Jr. Arequipa..... 74
Imagen 3.17 : Ficha Técnica de la Obra N° 08 – Jr. Arequipa 74
Imagen 4.1 : interfaz de la creación de usuario..... 96
Imagen 4.2 : Interfaz de la modificación de usuario 97
Imagen 4.3 : Interfaz para la búsqueda de I.U.P.C. 98
Imagen 4.4 : Interfaz para cargar nuevos I.U.P.C. en formato Excel 99
Imagen 4.5 : Interfaz para eliminar I.U.P.C. del último mes 100
Imagen 4.6 : Interfaz para nueva obra 101
Imagen 4.7 : Interfaz para importar presupuestos y fórmulas polinómicas 102
Imagen 4.8 : Interfaz para crear una valorización 103
Imagen 4.9 : Interfaz para procesar una valorización..... 104
Imagen 4.10 : Interfaz para eliminar una valorización 105
Imagen 5.1 : Comparación de resultados mediante diagrama de barras 257

ÍNDICE DE DIAGRAMAS

Diagrama 01 : Modelo Use Case del Negocio.....87

Diagrama 02 : Diagrama de gestión de la valorización 88

Diagrama 03 : Diagrama de la gestión de los índices 89

Diagrama 04 : Dominio del problema 89

Diagrama 05 : Diagrama de requerimiento - gestión de índices.....90

Diagrama 06 : Diagrama de requerimiento - gestión de valorización 91

Diagrama 07 : Diagrama de actividades - gestión de usuario 92

Diagrama 08 : Diagrama de actividades - gestión de I.U.P.C. 92

Diagrama 09 : Diagrama de actividades - gestión de Valorización 93

Diagrama 10 : Diagrama de colaboración - gestión de I.U.P.C. 93

Diagrama 11 : Diagrama de colaboración - gestión de Valorización (Mantenimiento de Obras) 94

Diagrama 12 : Diagrama de colaboración - gestión de Valorización (Ingreso de presupuesto y fórmulas polinómicas) 94

Diagrama 13 : Diagrama de colaboración - gestión de Valorización (Creación de una Valorización).....95

Diagrama 14 : Diagrama de Paquete de Análisis.....95

Diagrama 15 : Diagrama de subsistemas por capas.....96

Diagrama 16 : Diagrama de secuencia – Nuevo Usuario.....97

Diagrama 17 : Diagrama de secuencia – Modificar Usuario 98

Diagrama 18 : Diagrama de secuencia – Buscar I.U.P.C.....99

Diagrama 19 : Diagrama de secuencia – Cargar nuevos I.U.P.C. 100

Diagrama 20 : Diagrama de secuencia – Eliminar I.U.P.C. del último mes.....101

Diagrama 21 : Diagrama de secuencia – Nueva Obra 102

Diagrama 22 : Diagrama de secuencia – importar presupuestos y fórmulas polinómicas 103

Diagrama 23 : Diagrama de secuencia – Nueva Valorización 104

Diagrama 24 : Diagrama de secuencia – Procesar Valorización.....105
Diagrama 25 : Diagrama de secuencia – Eliminar Valorización.....106
Diagrama 26 : Diagrama de Clases 107
Diagrama 27 : Diagrama de Base de Datos.....108

ÍNDICE DE TABLAS

Tabla 01: Especificaciones de la Tabla obras 109
Tabla 02: Especificaciones de la Tabla C1_fórmulas 110
Tabla 03: Especificaciones de la Tabla especialidades 111
Tabla 04: Especificaciones de la Tabla C1_polinómicas_solo_materiales 111
Tabla 05: Especificaciones de la Tabla C1_presupuestos 112
Tabla 06: Especificaciones de la Tabla Valorización_datos 112
Tabla 07: Especificaciones de la Tabla C1_polinómicas 113
Tabla 08: Especificaciones de la Tabla Resumen_partidas 113
Tabla 09: Especificaciones de la Tabla Valor_adelanto_directo 114
Tabla 10: Especificaciones de la Tabla valorización 114
Tabla 11: Especificaciones de la Tabla Resumen_general 115
Tabla 12: Especificaciones de la Tabla Mayor_gasto_generales 115
Tabla 13: Especificaciones de la Tabla reajustes 116
Tabla 14: Especificaciones de la Tabla Deducción_adelanto_directo 117
Tabla 15: Especificaciones de la Tabla Amortización_adelanto_directo 117
Tabla 16: Especificaciones de la Tabla ampliación 118
Tabla 17: Especificaciones de la Tabla Deducción_adelanto_materiales 119
Tabla 18: Especificaciones de la Tabla Amortización_adelanto_materiales 120
Tabla 19: Especificaciones de la Tabla usuario 121
Tabla 20: Especificaciones de la Tabla Reajuste_gasto_generales 121
Tabla 21: Especificaciones de la Tabla Interés_demora_pago_valorización 122

CAPÍTULO I

CAPÍTULO 1: PLANTEAMIENTO DEL PROBLEMA

1.1 DESCRIPCIÓN Y FUNDAMENTACIÓN DEL PROBLEMA

En el Perú existen diversas empresas dedicadas a la ejecución de obras públicas y cada una de estas tiene una forma distinta de valorizar, esto trae como consecuencia diversidad de criterios, incluyendo el uso de formatos muy variados y el empleo de tiempos diferentes.

Existe un postulado para el desarrollo de las valorizaciones de obras públicas, dicho postulado implica hacer uso de algún software que nos agilice el desarrollo de las mismas, como por ejemplo el Excel (Microsoft, 1987), el s10 – Sistema 10 (Corporación Sistema 10 S.A.C, 2005), el K&BCOP – Sistema de costos de obra (BLACKSA Blacker Computer & K&bsoft S. A, 2006), el CPOC - Costos y presupuestos en obras civiles (Ing. Alfredo Ochante Mendoza, 2013) y el DELPHIN EXPRESS (Gustavo A. Guzmán Vigo, 2016). En este sentido, gran parte de las empresas estila usar el Excel como primera opción para el desarrollo de sus valorizaciones, debido a que es una herramienta de gran utilidad y de fácil acceso dentro del campo de la ingeniería civil.

Sin embargo, se han encontrado limitaciones no solo en el Excel, sino también en otros programas que son menos accesibles como el módulo de valorizaciones del s10, el K&BCOP, el CPOC y el DELPHIN EXPRESS.

Los antecedentes que presentan estos programas nos han hecho tomar conciencia para investigar profundamente el campo de las valorizaciones y poder desarrollar un nuevo programa que nos permita disminuir tales limitaciones.

En tal sentido, debemos mencionar a las siguientes limitaciones:

Limitación fundamental al usar el Excel: Pérdida de tiempo.

Limitación fundamental para el módulo de valorizaciones del S10: Falta de mantenimiento al módulo y actualizaciones del sistema con normativa vigente.

Limitaciones para los otros programas: Módulos incompletos respecto a los tipos de valorizaciones, como por ejemplo: Valorización de obra principal (no contempla el cálculo de las deducciones y amortizaciones del adelanto directo y adelanto para materiales), Valorización de mayores gastos generales variables y valorización de intereses.

Finalmente y luego de haber descrito las limitaciones que presentan otros software para desarrollar las valorizaciones de obra, podemos indicar que la investigación se centra en proponer una alternativa de solución a dichas limitaciones mediante el diseño e implementación del software V&L Versión 1.0 – 2017, aplicando el lenguaje de programación Java versión 8, el cual estará conectado al gestor de base de datos MySQL.

1.2 FORMULACIÓN DEL PROBLEMA

1.2.1. PROBLEMA GENERAL

¿La creación de un nuevo software, es una alternativa de solución para el ahorro de tiempo cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali?

1.2.2. PROBLEMAS ESPECÍFICOS

a. ¿Es posible que con la creación de un nuevo software se minimicen las limitaciones existentes que presentan otros programas cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali?

b. ¿Es posible que implementando los distintos tipos de valorizaciones en el nuevo software se pueda realizar un control económico adecuado de las obras públicas por contrata, en la provincia de Coronel Portillo - Ucayali?

1.3 OBJETIVOS

1.3.1. OBJETIVO GENERAL

Crear un nuevo software como alternativa de solución para el ahorro de tiempo cuando se desarrollen valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali.

1.3.2. OBJETIVOS ESPECÍFICOS

a. Minimizar las limitaciones existentes que presentan otros software cuando se desarrollen valorizaciones de obras públicas por contrata, en la provincia de Coronel Portillo - Ucayali.

b. Implementar los distintos tipos de valorizaciones en el nuevo software para realizar un control económico adecuado de las obras públicas por contrata, en la provincia de coronel Portillo - Ucayali.

1.4 JUSTIFICACIÓN E IMPORTANCIA

Actualmente en la ingeniería civil, es común construir programas que ayuden a resolver problemas a los ingenieros que se presentan en sus labores cotidianas, siendo el lenguaje de programación una herramienta importante para la ingeniería, porque puede brindar toda una gama de oportunidades para resolver obstáculos, sobre todo cuando se desarrollan valorizaciones de obras públicas.

Debido a la experiencia en obras públicas; (asistente del residente, asistente del supervisor); nos vemos obligados a conocer a profundidad el desarrollo de las valorizaciones, con la cual se ha detectado una serie de factores limitantes que impiden desarrollar las valorizaciones en periodos óptimos y/o ideales, de tal manera que se ha creído conveniente diseñar e implementar un software que satisfaga las expectativas de los usuarios, sobre todo en el ahorro de tiempo cuando se desarrollan valorizaciones. Allí radica la importancia del proyecto, porque con el software se busca principalmente el

ahorro de tiempo y un correcto control económico del avance de obra bajo la perspectiva de equilibrio entre entidades contratantes y contratistas.

1.5 LIMITACIONES Y ALCANCES

➢ LIMITACIONES

1. El trabajo de investigación no abarca el desarrollo de la valorización de mayores gastos generales variables para obra adicional.
2. El trabajo de investigación no abarca el desarrollo de la valorización de intereses por demora de pago en obra adicional y mayores gastos generales variables.
3. En el desarrollo de la valorización principal no se ha considerado el caso de la retención por garantía de fiel cumplimiento.
4. Para el desarrollo de la valorización de obra principal y obra adicional, son necesarios los presupuestos y fórmulas polinómicas en formatos Excel, los cuales deben ser exportados del sistema s10.
5. La presente investigación no abarca el desarrollo de valorizaciones de obra publicas por administración directa.
6. La presente investigación no contempla el desarrollo de las valorizaciones con adelantos parciales (adelanto directo y materiales).
7. La presente investigación no contempla la reformulación del calendario de avance de obra valorizado, cuando se presentan menores metrados o deductivos vinculantes.

➢ ALCANCES

La investigación abarca el diseño e implementación de un software para el desarrollo de los distintos tipos de valorizaciones de obras públicas por contrata, como son: Valorización de obra principal, obra adicional, mayores gastos generales variables e intereses, específicamente en la Provincia de Coronel Portillo - Ucayali que es el área de estudio para la presente tesis.

1.6 HIPÓTESIS

1.6.1. HIPÓTESIS GENERAL

La creación de un nuevo software es una alternativa de solución cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali, por lo tanto, el ahorro de tiempo está garantizado.

1.6.2. HIPÓTESIS ESPECÍFICAS

a. Con la creación del nuevo software se minimizarán las limitaciones existentes que presentan otros programas cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo – Ucayali.

b. Con la Implementación de los distintos tipos de valorizaciones en el nuevo software, se logrará un control económico adecuado de las obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali.

1.7 SISTEMA DE VARIABLES, DIMENSIONES E INDICADORES

1.7.1. VARIABLE INDEPENDIENTE

Creación de un nuevo Software para desarrollar valorizaciones de obras públicas por contrata.

1.7.2. VARIABLE DEPENDIENTE

Ahorro de tiempo.

CAPÍTULO II

CAPÍTULO 2: MARCO TEÓRICO

2.1 ANTECEDENTES O REVISIÓN DE ESTUDIOS REALIZADOS

- ✓ **ESTUDIOS REALIZADOS EN LA CREACIÓN DEL EXCEL Y SU APLICACIÓN EN VALORIZACIONES:**

  La verdad es que Excel no nació sin competencia. Por el contrario, llegó al mundo teniendo dos fuertes competidores que lideraban el mercado de hojas de cálculo que nacieron en 1979 y 1982 respectivamente: Visicalc y Lotus 1, 2, 3.

  Excel hace su aparición en el año 1985, aun y cuando Microsoft lanzó al mercado una hoja de cálculo llamada Multiplan en 1982 que fue muy popular en grandes sistemas, pero no pudo superar a **Lotus 1-2-3** el cual funcionaba muy bien en computadores personales. Por esta razón Microsoft desarrolló una nueva hoja de cálculo para competir con Lotus 1-2-3, quien fue **El Excel.**

  La historia del Excel puede ser resumida viendo la lista de versiones que han sido lanzadas al mercado para Microsoft Windows y son:

  - 1987 Excel 2.0
  - 1990 Excel 3.0
  - 1992 Excel 4.0
  - 1993 Excel 5.0 (Office 4.2 & 4.3)
  - 1995 Excel 7.0 (Office ‘95)
  - 1997 Excel 8.0 (Office ‘97)
  - 1999 Excel 9.0 (Office 2000)
  - 2001 Excel 10.0 (Office XP)
  - 2003 Excel 11.0 (Office 2003)
  - 2007 Excel 12.0 (Office 2007)
  - 2010 Excel 14.0 (Office 2010) versión beta.

Finalmente, dado que el Excel es un software potente que permite desarrollar diversidad de hojas de cálculo, éste ha sido orientado al desarrollo de las valorizaciones de obras, tal es así que hoy en día se están dictando cursos de “Valorizaciones de obras públicas” utilizando la herramienta Excel”. (http://www.excellentias.com/historia-de-excel/).

- ✓ **ESTUDIOS REALIZADOS EN LA CREACIÓN DEL S10 CON SU MÓDULO DE VALORIZACIONES:**

En 1986 un grupo de profesionales peruanos con experiencia en las ramas de la construcción, gestión empresarial y la informática crean Sistema 10 S.A.C. con la misión de desarrollar un software integrado que cubriera las necesidades reales de las empresas constructoras.

En 1987 nace el S10, con el módulo de Presupuestos Versión 1.0 que se desarrolla para el sistema operativo DOS. Con el transcurso de los años se va generalizando el uso de las computadoras, la tecnología va en aumento y el software va cambiando de versiones y logrando éxitos.

En 1989 se lanza al mercado el Módulo de Presupuestos Versión 3.0 que tiene una gran acogida entre sus usuarios: empresas privadas, profesionales independientes, entorno académico y sector público.

En 1996 la permanente investigación de nuestro equipo junto con los aportes de los usuarios hace que el S10 llegue a un estado de madurez con la versión 7.1. Algunas empresas del extranjero comienzan a comprar el software y se firman convenios con importantes instituciones como el Colegio de Ingenieros del Perú, el Colegio de Arquitectos, etc.

El año 1997 es un año trascendental para Sistema 10 ya que se firma un convenio con la empresa multinacional Microsoft, estableciéndose una alianza estratégica de negocios. A partir de entonces se comenzaron a desarrollar productos íntegramente con tecnología Microsoft, utilizando Visual Studio y base de datos MS SQL SERVER 7.0.

En mayo de 2000, tras un arduo pero fructífero trabajo de desarrollo junto con Microsoft, se lanza al mercado la Versión Corporativa de nuestro producto estrella: el Módulo de Presupuestos Versión 2000. La presentación se lleva a cabo en el 1er Encuentro Nacional de Usuarios del S10, donde asisten más de 800 profesionales de todo el país.

En agosto de ese mismo año, S10 fue seleccionado como Solución Recomendada por Telefónica del Perú para formar parte de su catálogo de soluciones Aplica. El gran volumen de información que utilizan nuestros usuarios hace que en la actualidad todos los módulos del S10 utilicen el servidor de base de datos MS SQL SERVER.

Finalmente, debido al constante crecimiento de la empresa sistema 10 S.A.C, se crea en el año 2005 el módulo de valorizaciones, sin embargo, éste módulo se encuentra actualmente desactualizado debido a la falta de mantenimiento del sistema. (http://www.s10peru.com/historia_i.php).

- ✓ **ESTUDIOS REALIZADOS EN LA CREACIÓN DE OTROS PROGRAMAS PARA VALORIZAR:**

K&BCOP

- Creado en el año 1972 bajo lenguaje de programación Fortran.
- Actualizado en el año 1986 bajo Assembler y Basic.
- Actualizado en el año 1997 para plataforma Windows 95, 98, bajo programación orientada a objetos.
- Actualizado al 2005 para plataforma Win NT, 2000, XP Profesional soportando hasta 50 usuarios por servidor.
- Usado en la liquidación de más de 2000 obras.
- Última versión 2010 actualizada a Diciembre del 2012 para plataformas Windows Vista y Windows 7 de 32&64 bits.
- Si bien es cierto, este programa tiene un módulo de valorizaciones, pero presenta un interfaz poco amigable y confuso, así mismo, no contempla

las valorizaciones de obra adicional, de mayores gastos generales variables y de intereses.
(https://es.scribd.com/document/246812436/KYBCOP).

CPOC

Cpoc es un software especializado para la industria de la construcción que permite realizar presupuestos de obra y abarcar todas las etapas de obra a través de sus diferentes opciones.

La facilidad de manejo y la potencia de este programa lo ha hecho un paquete amigable para el usuario inexperto en el área, ya que tiene todas las opciones y herramientas necesarias para elaborar un presupuesto de cualquier especialidad o tipo de obra, constando desde una Hoja de Metrados hasta su Liquidación Técnica Financiera, sin hacer mayor esfuerzo y totalmente automático, lo cual lo hace el software más potente, accesible y amigable que ningún otro en su categoría.

Sin embargo, éste programa presenta aún más limitaciones que el software K&BCOP, debido a que no contempla los adelantos otorgados, por lo tanto, omite el cálculo de las deducciones y amortizaciones de los mismos. (https://es.scribd.com/document/204083996/Manual-Cpoc-Ver-4-01).

DELPHIN EXPRESS

Es un software creado en el año 2012 y ha distribuido licencias a países como Perú, Honduras, Colombia y Argentina.

Delphin Express, utiliza la mejor tecnología en programación para ofrecer una funcionalidad total del sistema y crear una experiencia para el usuario, interactiva, jamás experimentada. El modelo se basa en un diseño ERP, cuyo núcleo se sustenta en una plataforma, donde se accede y controla, los diferentes módulos del sistema.

Este software puede metrar, presupuestar y valorizar empleando una navegación amplia con capacidad de crear y visualizar en tiempo real la totalidad de costos y recursos, potencializando la experiencia y trabajo de los usuarios más experimentados, así como los que se inician en el análisis de los costos y presupuestos de obra.

No obstante, el software omite el cálculo de las deducciones y amortizaciones de los adelantos otorgados a los contratistas, así como también el desarrollo de la valorización de obra adicional, mayores gastos generales variables e intereses por demora de pago. (http://www.ddbexpress.com/#).

2.2 MARCO SITUACIONAL O DE REFERENCIA

En la actualidad existe una variedad de software para desarrollar valorizaciones de obras públicas, que se encuentran dentro del mercado laboral y que presentan ciertas dificultades al momento del desarrollo de las mismas, no porque sean deficientes, sino porque existen una serie de factores limitantes, entre estos, el tiempo considerable que se utiliza para desarrollar las valorizaciones en el caso explícito del Excel, falta de mantenimiento al módulo de valorizaciones y actualizaciones del sistema con normativa vigente, como es el caso del S10 y módulos incompletos de los tipos de valorizaciones existentes, con respecto a los otros software, que originan que los usuarios se encuentren con una incertidumbre.

En este sentido, se ha encontrado un punto importante al que le llamamos pérdida de tiempo, por lo tanto, nos vemos en la necesidad de realizar una investigación que consiste en el diseño e implementación de un nuevo software para el desarrollo de valorizaciones de obras públicas, el cual presentará un interfaz amigable, resultados óptimos y nos permitirá un corto tiempo en su proceso.

2.3 ADMINISTRACIÓN DE CONTRATOS DE OBRAS PÚBLICAS

2.3.1. TIPOS DE EJECUCIÓN PRESUPUESTAL

Según la Ley 28411 – Ley General del Sistema Nacional de Presupuesto, en su **artículo 59**, refiere lo siguiente:

Artículo 59.- Tipos de Ejecución Presupuestal de las Actividades, Proyectos y Componentes.

La ejecución de las Actividades y Proyectos así como de sus respectivos componentes, de ser el caso, se sujeta a los siguientes tipos:

a) Ejecución Presupuestaria Directa: *Se produce cuando la Entidad con su personal e infraestructura es el ejecutor presupuestal y financiero de las Actividades y Proyectos, así como de su respectivo Componentes.*

b) Ejecución Presupuestaria Indirecta: *Se produce cuando la ejecución física y/o financiera de las Actividades y Proyectos así como de sus respectivos Componentes, es realizada por una Entidad distinta al pliego; sea por efecto de un contrato de un convenio celebrado con una Entidad privada, o con una Entidad pública, sea a título oneroso o gratuito.*

2.3.2. SISTEMAS DE CONTRATACIÓN

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 35,** menciona los siguientes sistemas de contratación:

Artículo 35.- Sistemas de contratación

a) *Suma Alzada.*

b) *Precios Unitarios.*

c) *Esquema mixto de suma alzada, tarifas y/o precios unitarios.*

d) *Tarifas.*

e) *En base a porcentajes.*

f) *En base a un honorario fijo y una comisión de éxito.*

Para la presente investigación sólo se tomarán en cuenta los dos primeros sistemas de contratación, debido a que son aplicables a la ejecución de obras públicas.

SUMA ALZADA

Aplicable cuando las cantidades, magnitudes y calidades de la prestación estén definidas en las especificaciones técnicas, términos de referencia o, en el caso de obras, en los planos, especificaciones técnicas, memoria descriptiva y presupuesto de obra, respectivas.

PRECIOS UNITARIOS

Aplicable en las contrataciones de bienes, servicios en general, consultorías y obras, cuando no puede conocerse con exactitud o precisión las cantidades o magnitudes requeridas.

2.3.3. MODALIDADES DE CONTRATACIÓN

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 36** menciona las siguientes modalidades de contratación:

Artículo 36.- Modalidades de contratación

a) Llave en mano: *Aplicable para la contratación de bienes y obras. Tratándose de obras, el postor oferta en conjunto la construcción, equipamiento y montaje hasta la puesta en servicio y, de ser el caso, la elaboración del Expediente Técnico y/o la operación asistida de la obra.*

b) Concurso oferta: *Cuando el postor oferta la elaboración del Expediente Técnico y ejecución de la obra. Ésta modalidad solo puede aplicarse en la ejecución de obras que se convoquen bajo el sistema a suma alzada y siempre que el presupuesto estimado del proyecto o valor referencial corresponda a una Licitación Pública.*

2.3.4. MÉTODOS DE CONTRATACIÓN

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 53, inciso 53.1,** refiere lo siguiente:

Artículo 53.- Procedimientos de selección

53.1 *Para la contratación de bienes, servicios en general, consultorías u obras, la Entidad utiliza, según corresponda, los siguientes procedimientos de selección:*

a) *Licitación Pública.*

b) *Concurso Público.*

c) *Adjudicación Simplificada.*

d) *Subasta Inversa Electrónica.*

e) *Selección de consultores Individuales.*

f) *Comparación de Precios.*

g) *Contratación Directa.*

53.2 *La determinación del procedimiento de selección se realiza en atención al objeto de la contratación, la cuantía y las demás condiciones para su empleo previstos en la ley y el reglamento.*

Por otra parte, la determinación de los procedimientos de selección para efectuar las licitaciones públicas, concursos públicos, adjudicaciones simplificadas y selección de consultores individuales en todas las entidades del sector público comprendidas en el artículo 3 de la Ley 30225, Ley de Contrataciones del Estado, se sujetan a los montos siguientes:



TOPES PARA PROCEDIMIENTOS DE SELECCIÓN
Año 2019

TIPO	MONTOS (S/)		
	BIENES	SERVICIOS Y CONSULTORÍA DE OBRAS	OBRAS
Licitación Pública	>= a 400,000	-	>= a 1'800,000
Concurso Público	-	>= a 400,000	-
Adjudicación Simplificada	< de 400,000 > a 33,600	< de 400,000 > a 33,600	< de 1'800,000 > a 33,600
Selección de Consultores Individuales	-	≤ de 40,000 > a 33,600	-
Comparación de Precios	<= de 63,000 > a 33,600	<= de 63,000 > a 33,600	-
Subasta Inversa Electrónica	> a 33,600	> a 33,600	-
Contratación Directa	> a 33,600	> a 33,600	> a 33,600

Elaborado por PERÚ CONTRATA el 18/12/2018

Imagen 2.1: Topes para procedimientos de selección para el año fiscal 2019.

La presente investigación, solo enmarcará los procedimientos de selección de Licitación Pública y Adjudicación Simplificada (Ver imagen 2.1), debido a que el diseño e implementación del software V&L versión 1.0 – 2017 se centra específicamente en el desarrollo de valorizaciones en obras públicas de la provincia de Coronel Portillo.

2.3.5. INICIO DEL PLAZO DE EJECUCIÓN DE OBRA

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 176, inciso 176.1,** refiere lo siguiente:

Artículo 176.- Inicio del plazo de ejecución de obra

176.1 *El inicio del plazo de ejecución de obra rige desde el día siguiente de que se cumplan las siguientes condiciones:*

a) *Que la Entidad notifique al Contratista quién es el inspector o el supervisor, según corresponda;*

b) *Que la Entidad haya hecho entrega total o parcial del terreno o lugar donde se ejecuta la obra, según corresponda;*

c) *Que la Entidad provea el calendario de entrega de los materiales e insumos que, de acuerdo con las bases, hubiera asumido como obligación;*

d) *Que la Entidad haya hecho entrega del Expediente Técnico de Obra completo, en caso éste haya sido modificado con ocasión de la absolución de consultas y observaciones.*

e) *Que la Entidad haya otorgado al contratista el adelanto directo, en las condiciones y oportunidad establecidas en el artículo 181.*

2.3.6. ADELANTOS PARA OBRAS

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 180,** refiere lo siguiente:

Artículo 180.- Clases de Adelantos

Los documentos del procedimiento de selección pueden establecer los siguientes adelantos:

a) Directos, *los que en ningún caso exceden en conjunto el diez por ciento (10%) del monto del contrato original.*

b) Para materiales o insumos, *los que en conjunto no superan el veinte por ciento (20%) del monto del contrato original.*

2.3.7. VALORIZACIONES

Definición: Es la cuantificación económica de un avance físico en la ejecución de la obra, realizada en un periodo determinado.

Marco legal:

La actividad de la construcción se puede desarrollar dentro de dos sectores claramente definidos.

- **Sector Privado:**

Normado por el código Civil, título IX, Capítulo tercero, <<contrato de obra>>, desde el artículo 1771° al 1789°.

- **Sector Público:**

Normado por el Ley N° 30225, Ley de Contrataciones del Estado modificado por D.L. 1444 y su Reglamento aprobado con D.S. N° 344-2018-EF, vigente desde el 30 de enero del 2019.

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 194,** refiere lo siguiente:

Artículo 194.- Valorizaciones y metrados

194.1 *Las valorizaciones tienen el carácter de pagos a cuenta y son elaboradas el último día de cada periodo previsto en las bases, por el inspector o supervisor y el contratista.*

194.2 *En el caso de las obras contratadas bajo el sistema de precios unitarios, durante la ejecución de la obra, las valorizaciones se formulan en función de los metrados ejecutados con los precios unitarios ofertados, agregando separadamente los montos proporcionales de gastos generales y utilidad ofertados por el contratista; a este último se agrega, de ser el caso, el porcentaje correspondiente al Impuesto General a las Ventas.*

194.3 *En el caso de las obras contratadas bajo el sistema a suma alzada, durante la ejecución de la obra, las valorizaciones se formulan en función de los metrados ejecutados aplicando las partidas y precios unitarios del desagregado de partidas que dio origen a la propuesta y que fuera presentada al momento de ofertar, agregando separadamente los montos proporcionales de gastos generales y utilidad ofertados; a este monto se agrega, de ser el caso, el porcentaje correspondiente al Impuesto General a las Ventas.*

194.4 *En las obras contratadas bajo el sistema a precios unitarios se valoriza hasta el total de los metrados realmente ejecutados, mientras que en el caso de las obras bajo el sistema de suma alzada se valoriza hasta el total de los metrados de presupuesto de obra.*

194.5 *Los metrados de obra ejecutados se formulan y valorizan conjuntamente por el contratista y el inspector o supervisor, y son presentados a la entidad dentro de los plazos que establezca el contrato. Si el inspector o supervisor no se presenta para la valorización conjunta con el contratista, este la efectúa. El inspector o supervisor revisa los metrados durante el periodo de aprobación de la valorización.*

194.6 *El plazo máximo de aprobación por el inspector o supervisor de las valorizaciones y su remisión a la entidad para periodos mensuales es de cinco (05) días, contados a partir del primer día hábil del mes siguiente al de la valorización respectiva, y es cancelada por la Entidad en fecha no posterior al último día de tal mes. Cuando las valorizaciones se refieran a periodos distintos a los previstos en este numeral, las bases establecen el tratamiento correspondiente de acuerdo con lo dispuesto en el presente artículo.*

194.7 *A partir del vencimiento del plazo establecido para el pago de estas valorizaciones, por razones imputables a la Entidad, el*

contratista tiene derecho al reconocimiento de los intereses legales efectivos, de conformidad con los artículos 1234,1245, 1246 del Código Civil.

A) VALORIZACIONES SEGÚN EL SISTEMA DE CONTRATACIÓN

A.1) Metrados a valorizar en obras a Suma alzada

$$V = \left[\left(Mtdos.ejec.x\ P.U._{desag,\ partidas}\right) + Gastos\ Generales_{ofertados} + Utilidad_{ofertada}\right]$$

***Nota:** Se valoriza hasta el 100% del metrado que figura en el presupuesto contratado. En caso se presente la ejecución de mayores metrados, estos no serán reconocidos, y en caso se presenten menores metrados no serán deducidos.

A.2) Metrados a valorizar en obras a Precios Unitarios

$$V = \left[\left(Mtdos.ejec.x\ P.U._{ofertados}\right) + Gastos\ Generales_{ofertados} + Utilidad_{ofertada}\right]$$

***Nota:** Se valoriza los metrados realmente ejecutados. Obviamente, los mayores metrados que excedan los metrados del presupuesto contratado que procedan de una modificación y/o variación del expediente técnico se tramitarán y reconocerán con adicionales, previo cumplimiento de las normas específicas sobre este tema que contiene los artículos 205 y 206 (artículos modificados por D.S. 344-2018-EF) del Reglamento de la Ley N° 30225, Ley de Contrataciones del Estado, así como normas de la Contraloría General de la Republica.

2.3.8. TIPOS DE VALORIZACIONES

Según el reglamento, existen diferentes tipos de valorizaciones:

- Valorización de obra principal: Proviene de valorizar el monto contratado.

- Valorización de obra adicional: Proviene de valorizar las obras adicionales.
- Valorización de mayores gastos generales variables: Proviene de las ampliaciones de plazos.
- Valorización de intereses: Proviene de la demora en los pagos de los 3 tipos de valorizaciones anteriores.
- Valorización de mayores metrados: Proviene de modificaciones del expediente técnico en obras a precios unitarios.

Formato para Valorización de Obra principal de obras Públicas

El formato fue establecido por el ministerio de Economía y Finanzas, es el siguiente:

Valorización del periodo		V
Reajustes		R
Deducción de reajuste	(-)	D
Valorización Bruta		**VB=V+R-D**
Amortización de adelantos	(-)	A
Valorización Neta		**VN = VB - A**
IGV.		I.G.V. = 18% x VN

2.3.9. FÓRMULAS POLINÓMICAS

Base legal

El artículo 38 (artículo modificado por Decreto Supremo N° 344-2018-EF) del reglamento de la Ley N° 30225, Ley de Contrataciones del Estado, ha ratificado la vigencia de fórmulas de reajuste mediante fórmulas polinómicas al señalar lo siguiente:

"Tanto la elaboración como la aplicación de las fórmulas polinómicas se sujetan a lo dispuesto en el Decreto Supremo N° 011-79-VC y sus modificatorias, ampliatorias y complementarias…". El numeral 38.6 del artículo 38 del Reglamento establece que "no son de aplicación las fórmulas de reajuste cuando los documentos del procedimiento de selección establezcan que las ofertas se expresen en moneda extranjera…".

Definición

Se define fórmula polinómica a la expresión matemática que representa la composición o estructura de costos de un presupuesto de Obra, sin incluir el Impuesto General a las Ventas (I.G.V.), conformada por la suma de términos llamados Monomios que contienen la incidencia de los principales elementos o recursos del costo de una obra (mano de obra, materiales, equipos de construcción, varios, gastos generales y utilidad).

El monomio está conformado por Coeficientes de incidencia de un elemento o grupos de elementos significativos de la obra que se considera como 1.000, el cual representa la proporción del costo de cada elemento en relación al costo total de la obra multiplicado por el índice de precios que expresa la relación entre el precio de un elemento en un fecha con el precio que tuvo en una fecha anterior, considerado como cien (100) y denominado "base".

Estructura básica de la fórmula Polinómica:

$$k = a\frac{J_r}{J_o} + b\frac{M_r}{M_o} + c\frac{E_r}{E_o} + d\frac{V_r}{V_o} + e\frac{GU_r}{GU_o}$$

Índices Unificados de Precios del INEI

Los Índices Unificados de Precios del INEI son publicados todos los meses en *EL PERUANO,* entre el 15 y el 20 de cada mes, y

corresponden a la variación de precios de los diversos recursos de la construcción (mano de obra, materiales, equipo, etc.) del mes anterior al de su publicación.

Los Índices Unificados no se pueden prorratear en función a periodos del tiempo, los Índices Unificados son válidos para todo el mes.

Áreas Geográficas

El INEI ha dividido nuestro país en (6) regiones: 1, 2, 3, 4, 5 y 6.

Las áreas geográficas son las siguientes:

- Área Geográfica N° 1: Tumbes, Piura, Lambayeque, La libertad, Cajamarca, Amazonas y San Martin.
- Área Geográfica N° 2: Ancash, Lima, Callao e Ica.
- Área Geográfica N° 3: Huánuco, Pasco, Junín, Huancavelica, Ayacucho, y Ucayali.
- Área Geográfica N° 4: Arequipa, Moquegua y Tacna.
- Área Geográfica N° 5: Loreto.
- Área Geográfica N° 6: Cuzco, Puno, Apurímac y Madre de Dios.

RELACIÓN DE ÍNDICES UNIFICADOS

CÓDIGO	ELEMENTO
A	
01	Aceite
02	Acero de construcción liso
03	Acero de construcción corrugado
04	Agregado fino
05	Agregado grueso
06	Alambre y cable de cobre desnudo
07	Alambre y cable Tipo TW y THW
08	Alambre y cable tipo WP
09	Alcantarilla metálica
10	Aparato sanitario con grifería
11	Artefacto de alumbrado exterior
12	Artefacto de alumbrado interior
13	Asfalto
B	
14	Baldosa acústica
	Baldosa asfáltica (m)
16	Baldosa vinílica
17	Bloque y Ladrillo
C	
18	Cable telefónico (d)
19	Cable NYY-N2XY (g) y (p)
20	Cemento asfáltico
21	Cemento Portland Tipo I
22	Cemento Portland Tipo II
23	Cemento Portland Tipo V
24	Cerámica esmaltada y sin esmaltar
	Cerrajería importada (Índice 30) (a)
26	Cerrajería nacional
D	
27	Detonante
28	Dinamita
	Dólar (e)
30	Dólar más Inflación mercado USA (f)
30-1	% vigente del MUC según Resolución Cambiaría (l)
30-2	% vigente del MUC según Resolución Cambiaría (l)
30-3	% vigente del MUC según Resolución Cambiaría (l)
30-4	% Dólar oferta y demanda (i)
31	Ducto de Concreto
F	
32	Flete Terrestre
33	Flete aéreo
G	
34	Gasolina
	Gelatina (Índice 28) (a)
	Gelignita (Índice 28) (a)
H	
37	Herramienta Manual
38	Hormigón
I	
39	Índice General de Precios al Consumidor (INEI)
L	
40	Loseta
M	
41	Madera en tiras para piso
42	Madera importada para encofrado y carpintería
43	Madera nacional para encofrado y carpintería
44	Madera terciada para encofrado y carpintería
45	Madera terciada para encofrado
46	Malla de acero
47	Mano de obra (incluido leyes sociales)
47-1	Mano de Obra - Región Grau (j) (n)
48	Maquinaria y equipo nacional
49	Maquinaria y equipo importado
50	Marco y tapa de fierro fundido
P	
51	Perfil de acero liviano
52	Perfil de aluminio
53	Petróleo diesel
54	Pintura látex
55	Pintura temple
56	Plancha de Acero LAC
57	Plancha de Acero LAF
	Plancha de acero mediano LAC (Índice 56) (a)
59	Plancha de fibro-cemento (s)
60	Plancha de poliuretano
61	Plancha galvanizada
62	Poste de concreto
	Poste de fierro (Índice 65) (k)
T	
64	Terrazo
65	Tubería de acero negro y/o galvanizado
66	Tubería de PVC para la red de Agua Potable y Alcantarillado (q)
	Tubería de asbesto cemento de 18"a 24"(Índice 66) (a)
68	Tubería de cobre
69	Tubería de concreto simple
70	Tubería de concreto reforzado
71	Tubería de fierro fundido
72	Tubería de PVC para agua
73	Ducto telefónico de PVC (h)
	Tubería de PVC para electricidad (SAP) (Índice 72) (c)
	Tubería de PVC para electricidad (SEL) (Índice 72) (b)
V	
	Válvula de bronce importada (Índice 30) (a)
77	Válvula de bronce nacional
78	Válvula de fierro fundido nacional
79	Vidrio incoloro nacional (o)
80	Concreto Premezclado (r)

Imagen 2.2: Relación de Índices Unificados.

RESOLUCIÓN JEFATURAL N° 150-2019-INEI

ÍNDICES UNIFICADOS DE PRECIOS DEL MES DE ABRIL DEL 2019
Á R E A S G E O G R Á F I C A S
(Base : Julio 1992 = 100,0)

Cod.	1	2	3	4	5	6	Cod.	1	2	3	4	5	6
01	962.13	962.13	962.13	962.13	962.13	962.13	**02**	550.39	550.39	550.39	550.39	550.39	550.39
03	529.91	529.91	529.91	529.91	529.91	529.91	**04**	562.40	955.21	1049.27	592.03	374.92	760.14
05	450.57	216.77	424.81	603.86	(*)	632.12	**06**	943.96	943.96	943.96	943.96	943.96	943.96
07	678.73	678.73	678.73	678.73	678.73	678.73	**08**	898.27	898.27	898.27	898.27	898.27	898.27
09	277.63	277.63	277.63	277.63	277.63	277.63	**10**	423.52	423.52	423.52	423.52	423.52	423.52
11	253.83	253.83	253.83	253.83	253.83	253.83	**12**	321.72	321.72	321.72	321.72	321.72	321.72
13	1792.19	1792.19	1792.19	1792.19	1792.19	1792.19	**14**	262.90	262.90	262.90	262.90	262.90	262.90
17	622.27	674.48	706.27	847.77	793.17	877.01	**16**	354.83	354.83	354.83	354.83	354.83	354.83
19	786.34	786.34	786.34	786.34	786.34	786.34	**18**	338.34	338.34	338.34	338.34	338.34	338.34
21	471.42	423.68	449.56	438.28	449.56	420.47	**20**	2185.03	2185.03	2185.03	2185.03	2185.03	2185.03
23	439.25	439.25	439.25	439.25	439.25	439.25	**22**	367.30	367.30	367.30	367.30	367.30	367.30
27	469.07	469.07	469.07	469.07	469.07	469.07	**24**	238.51	238.51	238.51	238.51	238.51	238.51
31	381.47	381.47	381.47	381.47	381.47	381.47	**26**	385.64	385.64	385.64	385.64	385.64	385.64
33	857.95	857.95	857.95	857.95	857.95	857.95	**28**	582.33	582.33	582.33	572.80	582.33	582.33
37	319.98	319.98	319.98	319.98	319.98	319.98	**30**	489.59	489.59	489.59	489.59	489.59	489.59
39	457.08	457.08	457.08	457.08	457.08	457.08	**32**	471.42	471.42	471.42	471.42	471.42	471.42
41	434.54	434.54	434.54	434.54	434.54	434.54	**34**	505.25	505.25	505.25	505.25	505.25	505.25
43	757.99	693.10	933.84	657.17	1152.62	897.58	**38**	436.16	969.27	872.21	571.53	(*)	645.28
45	330.00	330.00	330.00	330.00	330.00	330.00	**40**	397.51	449.21	450.27	356.90	272.89	331.41
47	600.40	600.40	600.40	600.40	600.40	600.40	**42**	286.63	286.63	286.63	286.63	286.63	286.63
49	309.08	309.08	309.08	309.08	309.08	309.08	**44**	389.70	389.70	389.70	389.70	389.70	389.70
51	299.42	299.42	299.42	299.42	299.42	299.42	**46**	454.92	454.92	454.92	454.92	454.92	454.92
53	825.22	825.22	825.22	825.22	825.22	825.22	**48**	369.76	369.76	369.76	369.76	369.76	369.76
55	507.41	507.41	507.41	507.41	507.41	507.41	**50**	652.47	652.47	652.47	652.47	652.47	652.47
57	388.19	388.19	388.19	388.19	388.19	388.19	**52**	317.23	317.23	317.23	317.23	317.23	317.23
59	212.21	212.21	212.21	212.21	212.21	212.21	**54**	449.85	449.85	449.85	449.85	449.85	449.85
61	259.54	259.54	259.54	259.54	259.54	259.54	**56**	540.10	540.10	540.10	540.10	540.10	540.10
65	258.23	258.23	258.23	258.23	258.23	258.23	**60**	295.99	295.99	295.99	295.99	295.99	295.99
69	389.45	327.82	428.87	488.52	269.39	451.51	**62**	468.11	468.11	468.11	468.11	468.11	468.11
71	656.12	656.12	656.12	656.12	656.12	656.12	**64**	348.21	348.21	348.21	348.21	348.21	348.21
73	547.96	547.96	547.96	547.96	547.96	547.96	**66**	716.08	716.08	716.08	716.08	716.08	716.08
77	337.26	337.26	337.26	337.26	337.26	337.26	**68**	258.32	258.32	258.32	258.32	258.32	258.32
							70	218.25	218.25	218.25	218.25	218.25	218.25
							72	432.33	432.33	432.33	432.33	432.33	432.33
							78	495.55	495.55	495.55	495.55	495.55	495.55
							80	111.25	111.25	111.25	111.25	111.25	111.25

(*) Sin Producción.

Nota: El cuadro incluye los índices unificados de código: 30, 34, 39, 47, 49 y 53 que fueron aprobados mediante Resolución Jefatural N° 137-2019-INEI.

Imagen 2.3: Modelo de Índices Unificados de Precios.

Condiciones Normativas de las Fórmulas Polinómicas

El D.S. N° 011-79-VC determina que las fórmulas polinómicas deben cumplir con lo siguiente:

a) Número máximo de monomios = 8

Por lo general, se amplían los monomios para materiales. Así, se puede tener:

$$\mathbf{K} = a\frac{J_r}{J_o} + b1\frac{M1_r}{M1_o} + b2\frac{M2_r}{M2_o} + b3\frac{M3_r}{M3_o} + b4\frac{M4_r}{M4_o} + c\frac{E_r}{E_o} + d\frac{V_r}{V_o} + e\frac{GU_r}{GU_o}$$

K representa la variación del precio del presupuesto total de la obra a una fecha dada, mientras que los monomios representan la variación de precio de cada recurso de la obra a una fecha dada.

b) Cada monomio (a excepción de los monomios de mano de obra y el de gastos generales y utilidad, excepción práctica ya que la norma no lo señala) puede contener, como máximo, 3 Índices Unificados, esto en razón de que en una obra hay diversidad de materiales.

c) Los coeficientes de incidencia de cada monomio deben ser, como mínimo, igual o mayor a 5% (0.050). Luego: a, b, c, d, e > 0.050.

Por lo tanto, los recursos del presupuesto cuya incidencia sea menor a 5% (0.050) se deben reagrupar con o dentro de otros índices, como máximo 3, con la finalidad de alcanzar o superar el 5%.

d) En una obra, como máximo, pueden haber 4 formulas polinómicas. Así, por ejemplo:

- Obra de edificación: Arquitectura, Estructuras, Sanitarias, Eléctricas.
- Obra de carreteras: Movimiento de tierra, Pavimentos, Obras de arte y drenaje y Señalización.

Si es que hubieran más componentes o fases en el presupuesto (por ejemplo: arquitectura, estructuras, sanitarias, eléctricas y electromecánicas, son cinco fases del presupuesto), se podrían unir las instalaciones eléctricas con la electromecánicas, pero se recomienda que sean fórmulas para cada especialidad. Luego, en la práctica, se considerarían 5 formulas polinómicas.

e) En un contrato que agrupe varias obras, como máximo, debe haber hasta 8 formula polinómicas.
Sin embargo, en la práctica también se ven obras con más de 8 formulas polinómicas.

f) Las ultimas opiniones del OSCE indican que si bien la fórmula polinómica no puede ser objeto de modificación durante la ejecución contractual y, por tanto, tampoco en la liquidación; en aquellos casos en los que la fórmula polinómica hubiera sido elaborada contraviniendo las disposiciones del Decreto Supremo N° 011-79-VC, la entidad de manera excepcional y en una decisión de gestión de su exclusiva responsabilidad puede corregir sus alcances con la única finalidad de ajustar el contenido del contrato a lo dispuesto en el ordenamiento legal aplicable.

2.3.10. NORMAS PARA EL CÁLCULO DE REAJUSTES

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S N° 344-2018-EF, en su **artículo 195**, refiere lo siguiente:

Articulo 195.- Reajustes

En el caso de obras, los reajustes se calculan en base al coeficiente del reajuste "K" conocido al momento de la valorización. Cuando se conozcan los Índices Unificados de Precios que aplican, se calcula el monto definitivo de los reajustes que le corresponden y se pagan con

la valorización más cercana posterior o en la liquidación final sin reconocimiento de intereses.

SISTEMA DE REAJUSTES

Se encuentra normado en el Decreto Supremo N° 011-79-VC del 01 de marzo de 1979, y modificaciones por los D.S. N° 017-79-VC, D.S. N° 022-80-VC, D.S. N° 006-86-VC y el D.S. N° 011-89-VC.

Las normas que se revisarán son las indicadas a continuación:

a) Normas Generales.
b) Normas para Obras Atrasadas.
c) Normas para Obras Permanentemente Adelantadas.

A. NORMAS GENERALES

Las valorizaciones de obras efectuadas o de adicionales a precios originales del contrato, serán ajustadas multiplicándolas por el respectivo coeficiente de reajuste "K" que se obtenga de aplicar en la fórmula o fórmulas polinómicas, los índices de precios correspondientes al mes en que se debe ser pagada la valorización, de acuerdo al plazo legal o contractual estipulado.

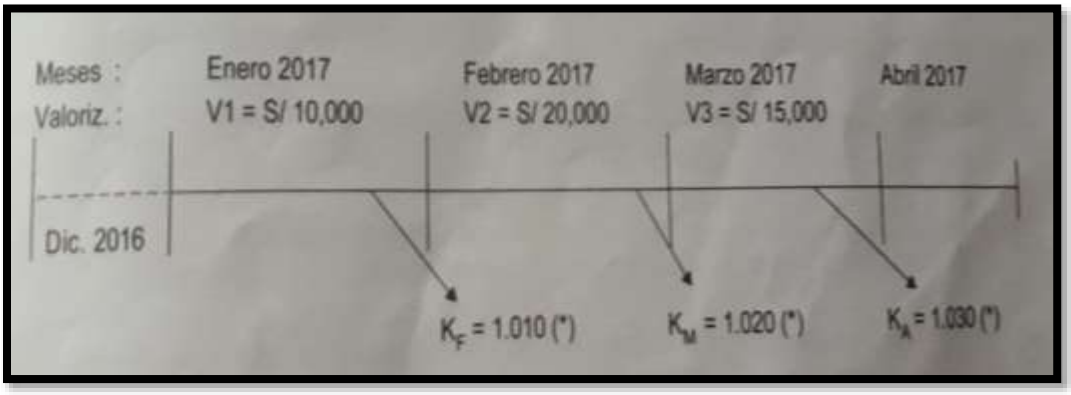


Imagen 2.4: Ejemplo de aplicación de los coeficientes de reajustes.

B. NORMAS PARA OBRAS ATRASADAS

$$\sum Valorizaciones\ Ejecutadas\ <\ \sum Valorizaciones\ Programadas$$

En este caso la norma sobre reajustes (Ítem B, art, 7° del D.S. N° 011-79-VC) señala lo siguiente:

a. El reajuste total acumulado sobre el avance realmente ejecutado no podrá superar el reajuste que hubiere correspondido al avance acumulado programado.

b. El reajuste que se abone al contratista en cada valorización de avance de obra, sumando con los ya pagados, no deberá superar el reajuste acumulado sobre el avance programado a la misma fecha.

c. Cuando en determinado momento, el avance real de la obra supera el atraso o se efectúa reprogramación de obra, además del reajuste que corresponda a la valorización del mes, se reintegrara la parte del reajuste dejada de abonar a consecuencia del atraso, a condición que se cumpla con lo establecido en el inciso a) de este rubro B.

1) En términos matemáticos la norma señala que, en una obra atrasada:

$$\sum Reajuste\ Ejecutado\ \leq\ \sum\ Reajuste\ Programado$$

2) La comparación de reajustes se hace por cada fórmula polinómica que exista en la obra, la cual debería estar complementado con su respectivo Calendario de Avance de obra Valorizado (por cada fase).

C. NORMAS PARA OBRAS PERMANENTEMENTE ADELANTADAS

En caso de reajustes, la norma (Ítem C, art. 7 del D.S. N° 011-79-VC) señala lo siguiente:

Las valorizaciones de obras permanentemente adelantadas con respecto al avance programado, serán reajustadas con el coeficiente de reajuste "K" respectivo y pagadas sin requerirse comparación con el reajuste el avance programado.

1) En términos matemáticos, la norma señala que, en una obra está permanentemente adelantada cuando:

$$\sum Valorizaciones\ ejecutadas > \sum Valorizaciones\ programadas$$

Nota: Se paga el reajuste ejecutado del mes sin comparar.

NORMAS PARA OBRAS CON ADELANTOS ESPECÍFICOS PARA MATERIALES E INSUMOS

El adelanto de materiales e insumos corresponde al monto que la entidad otorga al contratista, siempre y cuando las bases del proceso de selección lo hayan considerado y el contratista lo solicite (artículo 38° de la Ley N° 30225). De acuerdo con el artículo 180° del D.S. 344-2018-EF, la entidad puede otorgar adelantos para materiales e insumos, los que en conjunto no deberán superar el veinte por ciento (20%) del monto del contrato original, incluido el IGV.

$$\sum Adelanto\ Materiales \leq 20\%\ Monto\ del\ contrato\ original\ (inc.IGV)$$

Se debe precisar que el 20% es un tope máximo para el control; por lo tanto, no se debe otorgar el adelanto de materiales para dicho tope como es el caso del adelanto directo, aun cuando el contratista

presente garantía. Sobre las condiciones para la gestión del adelanto para materiales ver lo señalado en el artículo 182°.

El adelanto se calcula material por material, según procedimiento de cálculo mostrado en el ejemplo siguiente:

Solo es posible otorgar adelantos para materiales bajo los siguientes criterios:

a) El material solicitado debe estar representado en la (s) formula (s) polinómica (s) con su respectivo coeficiente de incidencia.
b) El material solicitado debe estar considerado en el calendario de adquisición de materiales presentado por el contratista para la firma del contrato.

Las Entidades suelen considerar en sus bases que, para otorgar el adelanto para materiales, el contratista presentará:

- ✓ Solicitud dirigida al Supervisor o Inspector, precisando monto, material y formula polinómicas.
- ✓ Cotizaciones u Órdenes de compra.
- ✓ Calendario de Adquisición de Materiales.
- ✓ Garantía por el monto solicitado (original se entrega a la entidad y la copia, al supervisor).
- ✓ El inspector o supervisor debe revisar y preparar su informe y elevarlo a la entidad.

La norma modificada señala que las bases fijen el plazo de la entidad para pagar el adelanto; sin embargo, si la demora llega a modificar el cronograma contractual, el contratista podría solicitar ampliación de plazo, según Opinión N° 023-2009 del OSCE.

Otorgado el adelanto al contratista está sujeto a la **Amortización y Deducción**. Respecto al D.S. N° 011-79-VC se tiene:

- El monto máximo específico para la compra de materiales agrupados bajo cada elemento representativo que podrá solicitar el contratista no deberá exceder al producto del coeficiente de incidencia del elemento representativo correspondiente, multiplicado por el porcentaje de participación de dicho elemento, así como el saldo bruto de obra por valorizar en el instante de hacerse efectivo el adelanto, afectado a su vez por la relación entre el índice de precio del citado elemento representativo a la fecha del adelanto y el que tuvo en la fecha del presupuesto base.

$$MONTO\ MÁX\ DEL\ ADEL.\ A\ OTORGAR = COEF.INCIDENCIA * PARTICIP.PORCENTUAL * SALDO\ BRUTO\ POR\ VALORIZAR * \frac{Ima}{Imo}$$

El Decreto Supremo N° 022-80-VC del 19.09.80 estable en su **articulo 1°** lo siguiente:

Artículo 1°.- El concepto de saldo de la obra por valorizar a que se refiere el Art. 7°. Párrafo D) del Decreto Supremo N° 011-79-VC, se aplicará separadamente a cada una de las fórmulas polinómicas referidas en el Art. 4° de ese mismo dispositivo.

En caso que se otorguen posteriores adelantos específicos para el mismo elemento, representativo, deberá verificarse que estos, incluyendo el saldo de los adelantos específicos anteriores, cumplan con lo establecido en el párrafo precedente.

Para calcular la deducción de los reajustes por variación de precios que puedan experimentar los materiales objeto de los citados adelantos a partir de la fecha en que la Entidad Publica Contratante los haya hecho efectivos y hasta su total utilización, se seguirá el siguiente procedimiento:

a. Se reajustará la valorización de acuerdo al coeficiente “K” proveniente de la fórmula polinómica de reajuste.

b. De la valorización reajustada, se deducirá el monto de reajuste que no corresponda, aplicándose la siguiente formula de deducción a cada adelanto otorgado para cada elemento representativo.

$$D = A\frac{(Imr - Ima)}{Imo}$$

Donde:

D: es la deducción en cada valorización bruta reajustada.

A: es el monto del adelanto utilizado en la valorización que se reajusta. Este monto se obtendrá multiplicando el coeficiente de incidencia del elemento correspondiente por el monto bruto de la valorización hasta completar el total del adelanto otorgado afectado por la expresión.

$$\frac{Imo}{ima}$$

Imo: Es el índice a la fecha del presupuesto base, del elemento representativo dentro del que se encuentra el material o materiales para los cuales se otorgó el adelanto.

Ima: Es el índice del mismo elemento representativo a la fecha efectiva del adelanto.

Imr: Es el índice del mismo elemento representativo a la fecha del reajuste.

c. En el caso que para un mismo elemento representativo, se abonara más de un adelanto en meses calendarios diferentes, se aplicará el procedimiento arriba fijado para cada adelanto; teniendo en cuenta que la utilización de un adelanto para el efecto del descuento

del reajuste se realizará una vez concluida la utilización del adelanto inmediato anterior.
El monto utilizado de ambos adelantos en una misma valorización no debe superar el producto de coeficientes de incidencias del elemento representativo por el monto de la valorización bruta.

d. La amortización de los adelantos específicos para materiales se efectuará en las valorizaciones que corresponden, en un monto igual al material utilizado en ellas, afectado por la relación entre el índice unificado del elemento representativo objeto del adelanto y el que tuvo a la fecha del presupuesto Base.

e. Los "materiales en cancha" suministrados por el contratista y valorizados a su solicitud por la entidad pública contratante serán considerados como adelanto, para efecto de los reajustes, a partir de la fecha en que se efectúe el pago de la valorización, siguiendo el mismo régimen señalado para los adelantos específicos.

f. Para el caso de que la entidad pública contratante, a solicitud del contratista, abone directamente a los proveedores facturas por materiales, se aplicara el mismo régimen señalado para los adelantos específicos.

NORMAS PARA OBRAS CON ADELANTO DIRECTO

El Adelanto Directo corresponde a un capital de trabajo que la entidad otorga al Contratista, siempre y cuando las Bases del procedimiento de Selección lo hayan considerado y el Contratista lo solicite.

Este adelanto puede ser hasta el 10% del Monto del Contrato original, incluido el IGV, y se otorga contra una Garantía por igual monto, renovable cada tres meses por el saldo del adelanto por amortizar (artículo 153 del D.S. 344-2018-EF).

$$Adelanto\ Directo\ \leq 10\%\ Contrato\ original\ (inc.IGV)$$

Una vez que la entidad ha otorgado el Adelanto, el Contratista está sujeto a la Amortización y Deducción de reajuste.

AMORTIZACIÓN DEL ADELANTO DIRECTO:

El artículo 183 del D.S. 344-2018-EF señala que la amortización es un descuento proporcional que se aplica en cada una de las valorizaciones de obra.

Las fórmulas para amortizar el adelanto son:

$$A = \%Adelanto\ Directo\ x\ Valorizacion\ del\ periodo$$

$$A = \frac{Adelanto\ Directo}{Monto\ Contrato}\ x\ Valorizacion\ del\ periodo$$

El adelanto directo se amortiza hasta completar el monto del adelanto otorgado sin IGV, ya que este impuesto corresponde a la SUNAT.

DEDUCCIÓN DEL REAJUSTE QUE NO CORRESPONDE POR EL ADELANTO DIRECTO:

Cuando el contratista reciba un adelanto directo por parte de la Entidad, se debe aplicar un descuento en el monto de la valorización que denominamos **Amortización**; por lo tanto, se evidencia que al contratista no debe percibir reajuste por esa parte de la Valorización que ha ejecutado con dinero adelantado por el Estado.

El descuento que se aplica sobre el reajuste esta normado en el D.S. N° 06-86-VC, el cual señala la siguiente formula general:

$$D = \frac{A.D.}{C}\ x\ V(\frac{K}{ka} - 1)$$

O su forma equivalente:

$$D = \frac{A.D.}{C} x \frac{V}{Ka}(K - Ka)$$

Donde:

D = Deducción de reajuste que no corresponde.

A.D. = Adelanto Directo otorgado sin IGV.

C = Monto del contrato Principal sin IGV.

V = Valorización ejecutada.

K = Coeficiente de reajuste de la valorización.

Ka = Coeficiente de reajuste del mes en que se otorgó el Adelanto Directo.

La deducción se aplica desde el mismo mes en que se otorga el adelanto directo, es continúa en cada valorización y se termina de aplicar cuando se termina de amortizar el Adelanto Directo.

2.3.11. VALORIZACIÓN DE OBRA ADICIONAL

Prestación adicional de obra:

Aquella no considerada en el expediente técnico de obra, ni el contrato original, cuya realización resulta indispensable y/o necesaria para dar cumplimiento a la meta prevista de la obra principal y que da lugar a un presupuesto adicional.

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S. N° 344-2018-EF, en su **artículo 205,** refiere lo siguiente:

Artículo 205.- Prestaciones adicionales de obras menores o iguales al quince por ciento (15%).

205.1 *Solo procede la ejecución de prestaciones adicionales de obra cuando previamente se cuente con la certificación de crédito presupuestario o previsión presupuestal, según las reglas previstas en la Normatividad del Sistema Nacional de Presupuesto Público y con la resolución del Titular de la Entidad o del servidor del siguiente nivel de decisión a quien se le hubiera delegado esta atribución y en los casos en que sus montos, restándole los presupuestos deductivos vinculados, no excedan el quince por ciento (15%) del monto del contrato original.*

205.2 *La necesidad de ejecutar una prestación adicional de obra es anotada en el cuaderno de obra, sea por el contratista, a través de su residente, o por el inspector o supervisor, según corresponda. En un plazo máximo de cinco (5) días contados a partir del día siguiente de realizada la anotación, el inspector o supervisor, según corresponda, ratifica a la Entidad la anotación realizada, adjuntando un informe técnico que sustente su posición respecto a la necesidad de ejecutar la prestación adicional. Además, se requiere el detalle o sustento de la deficiencia del expediente técnico de obra o del riesgo que haya generado la necesidad de ejecutar la prestación adicional.*

205.4 *El contratista presenta el expediente técnico de adicional de obra, dentro de los quince (15) días a la anotación en el cuaderno de obra, siempre que el inspector o supervisor, según corresponda, haya ratificado la necesidad de ejecutar la prestación adicional. De ser el caso, el inspector o supervisor remite a la entidad la conformidad sobre el expediente técnico de obra formulado por el contratista en el plazo de diez (10) días de presentado este último.*

205.5 *De existir partidas cuyos precios unitarios no están previstos en el presupuesto de obra, se adjunta al expediente técnico de obra el documento del precio unitario pactado con el contratista ejecutor de la*

obra. El acuerdo de precios se realiza entre el residente y el supervisor o el inspector, el cual es remitido a la Entidad para su aprobación como parte del presupuesto adicional de obra.

205.6 *En el caso que el inspector o supervisor emita la conformidad sobre el expediente técnico presentada por el contratista, la Entidad en un plazo de doce (12) días hábiles emite y notifica al contratista la resolución mediante la que se pronuncia sobre la procedencia de la ejecución de la prestación adicional de obra. La demora de la Entidad en emitir y notificar esta resolución puede ser causal de ampliación de plazo.*

205.7 *A efectos de aprobar la ejecución del adicional de obra la entidad cuenta con el informe de viabilidad presupuestal y la opinión favorable sobre la solución técnica propuesta en el expediente técnico presentada en el expediente técnico. Para emitir una opinión técnica sobre la solución técnica propuesta, la Entidad solicita el pronunciamiento del proyectista, de no contarse con dicho pronunciamiento o siendo negativo este, el órgano de la Entidad responsable de la aprobación de los estudios emite la opinión correspondiente.*

205.9 *En los contratos de obra, los presupuestos adicionales de obra se formulan con los precios del contrato y/o precios pactados y los gastos generales fijos y variables propios de la prestación adicional para lo cual se realiza el análisis correspondiente teniendo como base o referencia el análisis de los gastos generales del presupuesto original contratado. Asimismo, se incluye la utilidad del presupuesto ofertado y el Impuesto General a las Ventas correspondiente.*

205.10 *Cuando en los contratos suscritos bajo el sistema de precios unitarios ejecutar mayores metrados, estos son autorizados por el*

supervisor o inspector de obra a través de su anotación en el cuaderno de obra y comunicados a la Entidad, de forma previa a su ejecución.

205.11 *El contratista mediante anotación en cuaderno de obra solicita la ejecución de mayores metrados del residente. El supervisor autoriza su ejecución siempre que no se supere el quince por ciento (15%) del monto del contrato original, considerando el monto acumulado de los mayores metrados y las prestaciones adicionales de obras, restándoles los presupuestos deductivos. El monto a pagarse por la ejecución de estos mayores metrados se computa para el cálculo del límite para la aprobación de adicionales, previsto en el 34.5 del artículo de la ley.*

205.12 *No se requiere la aprobación previa de la Entidad para la ejecución de mayores metrados, pero si para su pago. El encargado de autorizar el pago es el Titular de la Entidad o a quien se le delegue dicha función.*

205.13 *De no existir precios unitarios de una determinada partida requerida en la prestación adicional, se pactan nuevos precios unitarios, considerando los precios de los insumos, tarifas o jornales del presupuesto de obra y, de no existir, se sustenta en precios del mercado debidamente sustentados. A falta de acuerdo con el contratista, y con la finalidad de no retrasar la aprobación y ejecución de la prestación adicional, el supervisor o inspector se encuentra facultado para fijar un precio, el cual se aplica sin perjuicio del derecho del contratista para someterlo al procedimiento de solución de controversias que corresponda dentro de los siguientes treinta (30) días hábiles contados desde la aprobación del presupuesto de la prestación adicional. El plazo señalado en el presente numeral es de caducidad.*

205.14 *El pago de los presupuestos adicionales aprobados se realizan mediante* ***valorizaciones adicionales.***

205.15 *Cuando se apruebe la prestación adicional de obra el contratista está obligado a ampliar el monto de garantía de fiel cumplimiento, debiendo entregar la actualización del valor de la garantía correspondiente en el plazo máximo de ocho (8) días hábiles de ordenada la prestación adicional.*
Igualmente, cuando se apruebe la reducción de prestaciones, el contratista puede reducir el monto de dicha garantía.

205.16 *En los casos en los que el contratista, para la elaboración del expediente técnico del adicional de obra, requiera realizar ensayos especializados de alta complejidad y/o la participación de algún especialista que no esté contemplado en la relación de su personal clave corresponde incluir su costo en los gastos generales propios del adicional.*

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S. N° 344-2018-EF, en su **artículo 206,** refiere lo siguiente:

Artículo 206.- Prestaciones adicionales de obras mayores al quince por ciento (15%).

206.1 *Las prestaciones adicionales de obras cuyos montos, restándole los presupuestos deductivos vinculados, superen el quince por ciento (15%) del monto del contrato original, luego de ser aprobadas por el Titular de la Entidad, requiere previamente, para su ejecución y pago, la autorización expresa de la Contraloría General de la Republica. La determinación del referido porcentaje incluye los montos acumulados de los mayores metrados, en contratos a precios*

unitarios. En el caso de adicionales con carácter de emergencia la autorización de la Contraloría General de la Republica se emite previa al pago.

Finalmente, la valorización de obra adicional es la cuantificación económica de una prestación adicional de obra, pero no incluye adelanto directo, ni adelanto para materiales, en consecuencia, no existe el cálculo de deducciones ni amortizaciones.

Formato para Valorización de Obra Adicional de obras Públicas

El formato fue establecido por el ministerio de Economía y Finanzas, es el siguiente:

Valorización del periodo	V
Reajustes	R
Valorización Bruta	**VB=V+R**
Valorización Neta	**VN = VB**
IGV.	I.G.V. = 18% x VN

2.3.12. VALORIZACIÓN DE MAYORES GASTOS GENERALES VARIABLES

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S. N° 344-2018-EF, en su **artículo 197,** refiere lo siguiente:

Artículo 197.- Causales de ampliación de plazo

El contratista puede solicitar la ampliación de plazo pactado por cualquiera de las siguientes causales: ajenas a su voluntad, siempre

que modifiquen la ruta crítica del programa de ejecución vigente al momento de la solicitud de ampliación:

a) Atrasos y/o paralizaciones por causas no atribuibles al contratista.

b) Cuando es necesario un plazo adicional para la ejecución dela prestación de adicional de obra. En este caso el contratista amplía el plazo de las garantías que hubiere otorgado.

c) Cuando es necesario un plazo adicional para la ejecución de los mayores metrados, en contratos a precios unitarios.

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S. N° 344-2018-EF, en su **artículo 199,** refiere lo siguiente:

Artículo 199. - Efectos de la modificación del plazo contractual

199.1 Las ampliaciones de plazo en los contratos de obra dan lugar al pago de mayores costos directos y mayores gastos generales variables, ambos directamente vinculados con dichas ampliaciones.

199.3 Los mayores gastos generales variables se determinan en función al número de días correspondientes a la ampliación multiplicado por el gasto general variable diario, salvo en los casos de las ampliaciones de plazo que se aprueben para la ejecución de prestaciones de adicionales de obra y que tienen calculados en su presupuesto sus propios gastos generales.

199.4 Solo cuando la ampliación de plazo sea generada por la paralización total de la obra por causas ajenas a la voluntad del contratista, da lugar al pago de mayores gastos generales variables debidamente acreditados, de aquellos conceptos que forman parte de

la estructura de gastos generales variables de la oferta económica del contratista.

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S. N° 344-2018-EF, en su **artículo 200,** refiere lo siguiente:

Artículo 200. – Cálculo del Gasto General Variable

En los contratos de obra el gasto general variable diario se calcula dividiendo los gastos generales variables ofertados entre el número de días de plazo contractual original, ajustados por el coeficiente "Ip/Io", en donde "Ip" es el Índice General de Precios al Consumidor (Código 39) aprobado por el Instituto Nacional de Estadística e Informática – INEI correspondiente al mes calendario en que ocurre la causal de ampliación del plazo contractual, e "Io" es el mismo índice de precios correspondiente al mes del valor referencial.

- **Ejemplo de cálculo de los Mayores Gastos Generales Variables (Obras a Suma Alzada y Precios Unitarios)**

$$\text{GGVD} = \frac{\text{Gastos Generales Variables del Presupuesto Ofertado}}{\text{Dias del plazo contractual original}} \text{ x} \frac{\text{Ip}}{\text{Io}}$$

Donde:

GGVD : Gastos Generales Variables diario.

$$\text{MGGV} = \text{GGVD x Dias de ampliacion de plazo otorgada}$$

Donde:

MGGV : Mayores Gastos Generales Variables.

Según el Reglamento de la Ley de Contrataciones del Estado (RLCE), aprobado con D.S. N° 344-2018-EF, en su **artículo 201,** refiere lo siguiente:

Artículo 201. – Pago de mayores Costos Directos y Mayores Gastos Generales Variables

201.1 Una vez que se haya aprobado la ampliación de plazo se formula una valorización de los mayores costos directos y mayores gastos generales variables para su pago, la cual es presentada por el residente al inspector o supervisor: dicho profesional, en un plazo máximo de quince (15) días contados a partir del día siguiente de recibida la mencionada valorización, la eleva a la Entidad con las correcciones a que hubiere lugar para su revisión y aprobación.

En caso la Entidad apruebe la referida valorización, es pagada en un plazo máximo de treinta (30) días contados a partir del día siguiente de recibida la valorización por parte del inspector o supervisor.

201.3 A partir del vencimiento del plazo establecido para el pago de esta valorización, el contratista tiene derecho al reconocimiento de intereses legales.

2.3.13. VALORIZACIÓN DE INTERESES POR DEMORA DE PAGOS

De acuerdo a la ley de contrataciones del estado:

“En caso de atraso en el pago por parte de la Entidad, salvo que se deba a causa fortuita o fuerza mayor, esta reconoce al Contratista los **intereses legales** correspondientes, debiendo repetir contra los responsables de la demora injustificada. Igual derecho corresponde a la Entidad en caso sea la acreedora”.

Para el pago de los intereses, el contratista debe desarrollar la valorización de intereses, ya sea por la demora en el pago de la valorización de obra principal, obra adicional y mayores gastos

generales variables, no existiendo el reconocimiento de los intereses por demora en el pago de los mismos.

$$Factor\ interés(i) = \frac{factor\ acum.\ tasa\ de\ interés\ legal\ corresp.\ al\ dia\ del\ pago\ real\ de\ la\ valorización}{factor\ acum.\ tasa\ de\ interés\ legal\ corresp.\ a\ la\ fecha\ maxima\ de\ pago\ de\ la\ val.} - 1$$

$$Intereses = Valorizacion\ neta\ x\ i$$

La valorización neta o liquida corresponde al monto en efectivo adeudado al contratista; por lo tanto, al momento de facturar el monto de intereses calculado, se le debe agregar el I.G.V.

2.4 INGENIERÍA DE SOFTWARE

De acuerdo con la definición de Fritz Bauer, la ingeniería de software es: El establecimiento y uso de principios fundamentales de la ingeniería con el objeto de desarrollar en forma económica un software que sea confiable y que trabaje con eficiencia en máquinas reales.

Según el Institute of Electrical and Electronics Engineers (IEEE), define a la ingeniería de software como: la aplicación de un enfoque sistemático, disciplinado y cuantificable al desarrollo, operación y mantenimiento del software; es decir la aplicación de la ingeniería al software. Por otra parte define también a la ingeniería como el estudio de los enfoques indicados líneas arriba.

La ingeniería de software es una tecnología con varias capas, las cuales son: herramientas, métodos, proceso y compromiso con la calidad. Cualquier enfoque de ingeniería (incluso la de software) debe basarse en un compromiso organizacional con la calidad.



Imagen 2.5: Capas de la Ingeniería de Software.

El fundamento en el que se apoya la ingeniería de software es el compromiso con la calidad.

El fundamento para la ingeniería de software es la capa proceso. El proceso de ingeniería de software es el aglutinante que une las capas de la tecnología y permite el desarrollo racional y oportuno del software de cómputo. El proceso define una estructura que debe establecerse para la obtención eficaz de tecnología de ingeniería de software. El proceso de software forma la base para el control de la administración de proyectos de software, y establece el contexto en el que se aplican métodos técnicos, se generan productos del trabajo (modelos, documentos, datos, reportes, formatos, etc.), se establecen puntos de referencia, se asegura la calidad y se administra el cambio de manera apropiada.

Los métodos de la ingeniería de software proporcionan la experiencia técnica para elaborar software. Incluyen un conjunto amplio de tareas, como comunicación, análisis de los requerimientos, modelación del diseño, construcción del programa, pruebas y apoyo. Los métodos de la ingeniería de software se basan en un conjunto de principios fundamentales que gobiernan cada área de la tecnología e incluyen actividades de modelación y otras técnicas descriptivas.

Las herramientas de la ingeniería de software proporcionan un apoyo automatizado o semiautomatizado para el proceso y los métodos. Cuando se integran las herramientas de modo que la información creada por una pueda

ser utilizada por otra, queda establecido un sistema llamado ingeniería de software asistido por computadora que apoya el desarrollo de software. (Pressman, 2010, p. 11).

2.4.1. PROCESO UNIFICADO DE RATIONAL (RUP)

El Proceso Unificado de Rational o RUP (por sus siglas en inglés de Rational Unified Process) es un proceso de desarrollo de software desarrollado por la empresa Rational Software, actualmente propiedad de IBM. Junto con el Lenguaje Unificado de Modelado (UML), constituye la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos.

El RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

Originalmente se diseñó un proceso genérico y de dominio público, el Proceso Unificado, y una especificación más detallada, el Rational Unified Process, que se vendiera como producto independiente.

PRINCIPALES CARACTERÍSTICAS

- Desarrollo iterativo.
- Administración de Requisitos.
- Uso de la arquitectura basada en componentes.
- Control de cambios.
- Modelado visual del software.
- Verificación de la calidad del software.
- Pretende implementar las mejores prácticas en Ingeniería de Software, de forma que se adapte a cualquier proyecto.

El RUP es un producto de Rational (IBM). Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Incluye artefactos (que son los productos tangibles del proceso como por

ejemplo, el modelo de casos de uso, el código fuente, etc.) y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

FASES DEL PROCESO UNIFICADO DE RATIONAL (RUP)

1. **Fase de Inicio (también llamado Incepción o Concepción)**: Esta fase tiene como propósito definir y acordar el alcance del proyecto con los patrocinadores o alumnos de un proyecto en el cual tenemos que, identificar los riesgos asociados al proyecto, proponer una visión muy general de la arquitectura de software y producir el plan de las fases y el de iteraciones posteriores.
2. **Fase de elaboración:** En la fase de elaboración se seleccionan los casos de uso que permiten definir la arquitectura base del sistema y se desarrollarán en esta fase, se realiza la especificación de los casos de uso seleccionados y el primer análisis del dominio del problema, se diseña la solución preliminar.
3. **Fase de Desarrollo (también llamado Implementación, Construcción):** El propósito de esta fase es completar la funcionalidad del sistema, para ello se deben clarificar los requisitos pendientes, administrar los cambios de acuerdo a las evaluaciones realizadas por los usuarios y se realizan las mejoras para el proyecto.
4. **Fase de transición (también llamado Cierre):** El propósito de esta fase es asegurar que el software esté disponible para los usuarios finales, ajustar los errores y defectos encontrados en las pruebas de aceptación, capacitar a los usuarios y proveer el soporte técnico necesario. Se debe verificar que el producto cumpla con las especificaciones entregadas por las personas involucradas en el proyecto.

El RUP comprende 2 aspectos importantes, por los cuales se establecen las disciplinas:

a) Proceso:

Las etapas de esta sección son: (revisar nuevamente la gráfica)

- Modelado de negocio.
- Requisitos.
- Análisis y Diseño.
- Implementación.
- Pruebas.
- Despliegue.

b) Soporte:

En esta parte nos encontramos con las siguientes etapas:

- Gestión del cambio y configuraciones
- Gestión del proyecto
- Entorno

  La estructura dinámica de RUP es la que permite que éste sea un proceso de desarrollo fundamentalmente iterativo, y en esa parte se ven inmersas en cuatro fases:

ARTEFACTOS

El RUP en cada una de sus fases realiza una serie de artefactos que sirven para comprender mejor tanto el análisis como el diseño del sistema (entre otros). Estos artefactos son los siguientes:

Inicio

- Documento Visión.
- Diagramas de caso de uso.
- Especificación de Requisitos.
- Diagrama de Requisitos.

Elaboración

Documento Arquitectura que trabaja con las siguientes vistas:

Vista Lógica

- Diagrama de clases.
- Modelo E-R (Si el sistema así lo requiere).

Vista de Implementación

- Diagrama de Secuencia.
- Diagrama de estados.
- Diagrama de Colaboración.

Vista Conceptual

- Modelo de dominio.

Vista física

- Mapa de comportamiento a nivel de hardware.
- Diseño y desarrollo de casos de uso, o flujos de casos de uso arquitectónicos.
- Pruebas de los casos de uso desarrollados, que demuestran que la arquitectura documentada responde adecuadamente a requerimientos funcionales y no funcionales.

Construcción

- Especificación de requisitos faltantes.
- Diseño y desarrollo de casos de uso y/o flujos de acuerdo con la planeación iterativa.
- Pruebas de los casos de uso desarrollados, y pruebas de regresión según sea el caso.

Transición

- Pruebas finales de aceptación.

- Puesta en producción.
- Estabilización.
  (https://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational).

2.4.2. LENGUAJE UNIFICADO DE MODELADO (UML)

El lenguaje unificado de modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el Object Management Group (OMG).

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados.

Es importante remarcar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo.

TIPOS DE DIAGRAMAS EN UML 2.5

Existen dos clases principales de tipos de diagramas: diagramas estructurales y diagramas de comportamiento. Estos últimos incluyen varios que representan diferentes aspectos de las interacciones. Estos diagramas pueden ser categorizados jerárquicamente como se muestra en el siguiente diagrama de clase:

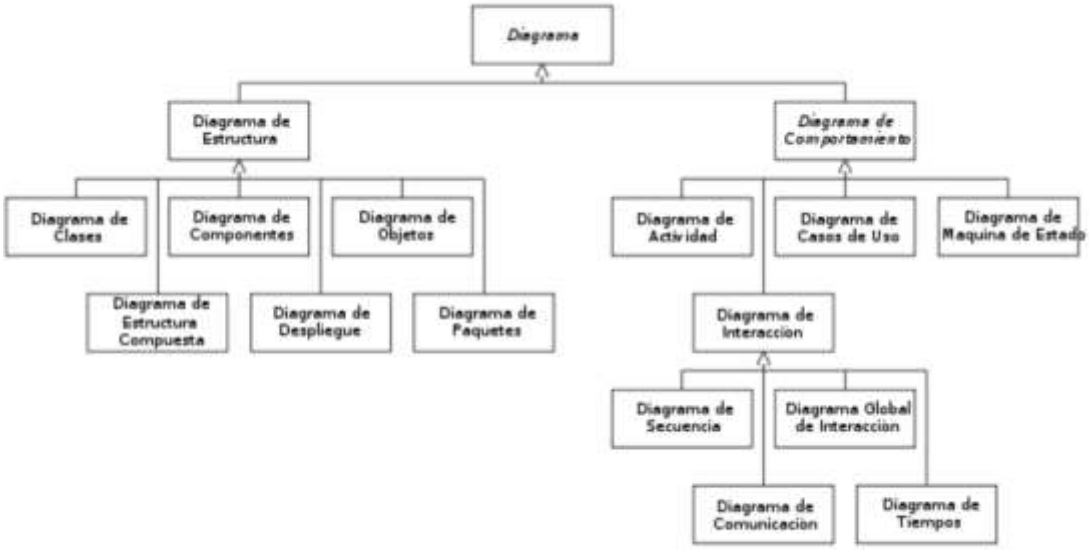


Imagen 2.6: Tipos de diagramas en UML 2.5.

ESTRUCTURALES

Muestran la estructura estática de los objetos de un sistema.

a) Diagrama de clases: Los diagramas de clase son, sin duda, el tipo de diagrama UML más utilizado. Es el bloque de construcción principal de cualquier solución orientada a objetos. Muestra las clases en un sistema, atributos y operaciones de cada clase y la relación entre cada clase. En la mayoría de las herramientas de modelado, una clase tiene tres partes, nombre en la parte superior, atributos en el centro y operaciones o métodos en la parte inferior. En sistemas grandes con muchas clases relacionadas, las clases se agrupan para crear diagramas de clases. Las diferentes relaciones entre las clases se muestran por diferentes tipos de flechas.

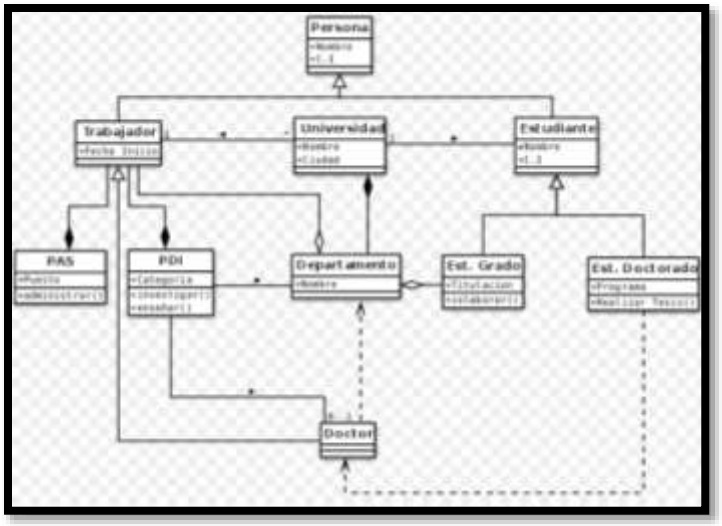


Imagen 2.7: Ejemplo de diagrama de clases.

b) Diagrama de componentes: Un diagrama de componentes muestra la relación estructural de los componentes de un sistema de software. Estos se utilizan principalmente cuando se trabaja con sistemas complejos que tienen muchos componentes. Los componentes se comunican entre sí mediante interfaces. Las interfaces se enlazan mediante conectores.

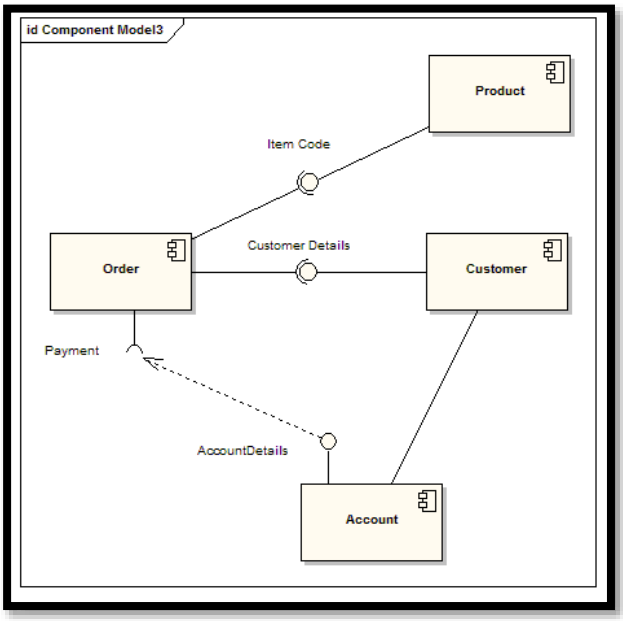


Imagen 2.8: Ejemplo de diagrama de componentes.

c) **Diagrama de despliegue:** Un diagrama de despliegue muestra el hardware de su sistema y el software de ese hardware. Los diagramas de implementación son útiles cuando la solución de software se despliega en varios equipos, cada uno con una configuración única.

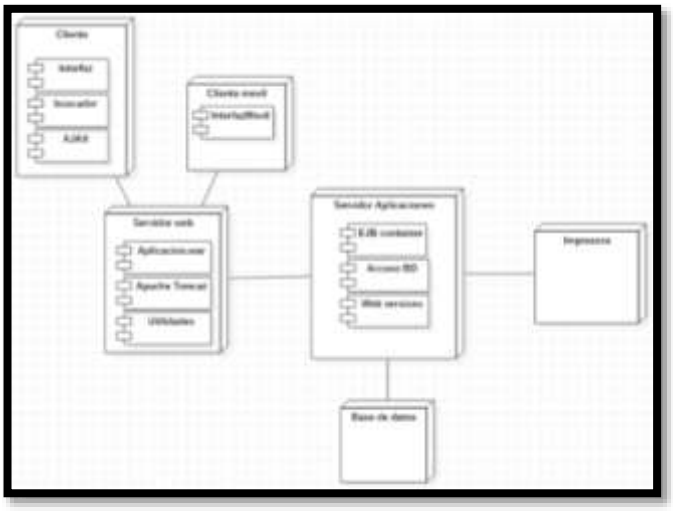


Imagen 2.9: Ejemplo de diagrama de despliegue.

d) **Diagrama de objetos:** Los diagramas de objetos, a veces denominados diagramas de instancia, son muy similares a los diagramas de clases. Al igual que los diagramas de clases, también muestran la relación entre los objetos, pero usan ejemplos del mundo real. Se utilizan para mostrar cómo se verá un sistema en un momento dado. Debido a que hay datos disponibles en los objetos, a menudo se utilizan para explicar relaciones complejas entre objetos.

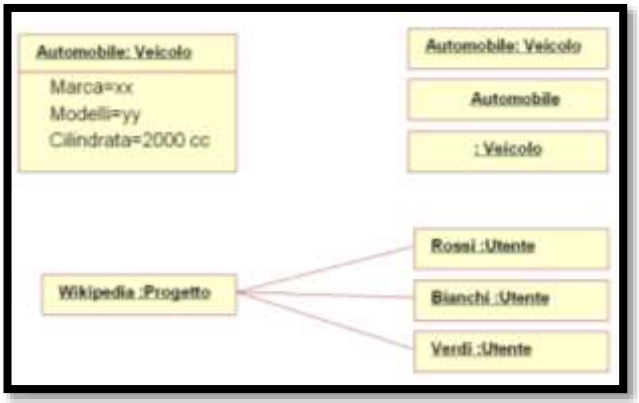


Imagen 2.10: Ejemplo de diagrama de objetos.

e) **Diagrama de paquetes:** Como su nombre indica, un diagrama de paquetes muestra las dependencias entre diferentes paquetes de un sistema.

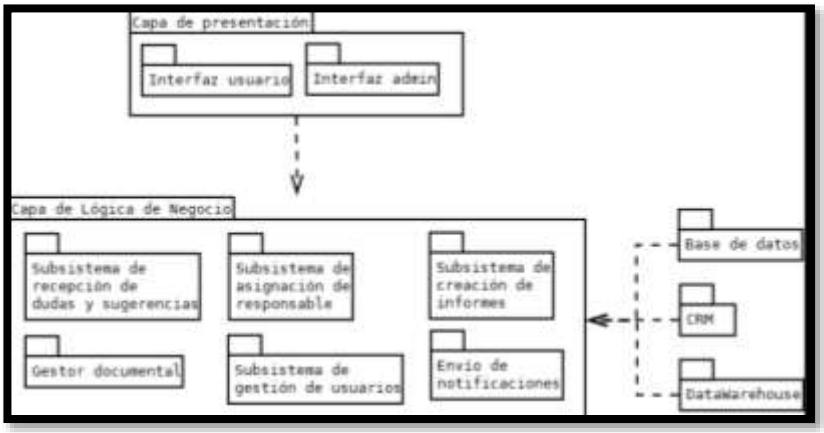


Imagen 2.11: Ejemplo de diagrama de paquetes.

f) **Diagrama de estructura compuesta:** Los diagramas de estructura compuesta se utilizan para mostrar la estructura interna de una clase.

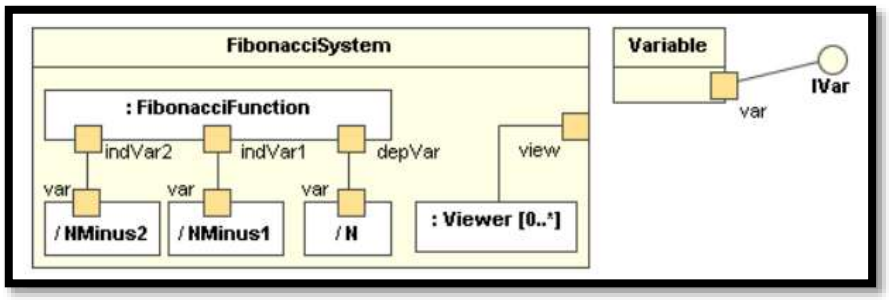


Imagen 2.12: Ejemplo de diagrama de estructura compuesta.

DE COMPORTAMIENTO

Muestran el comportamiento dinámico de los objetos en el sistema.

a) **Diagrama de actividades:** Los diagramas de actividad representan los flujos de trabajo de forma gráfica. Pueden utilizarse para describir el flujo de trabajo empresarial o el flujo de trabajo operativo de cualquier componente de un sistema. A veces, los diagramas de actividad se utilizan como una alternativa a los diagramas de máquina del estado.

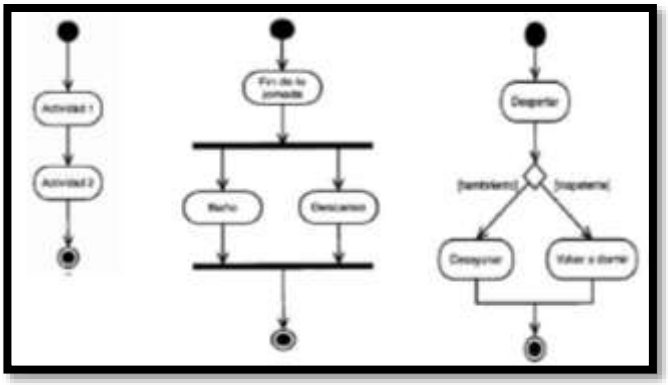


Imagen 2.13: Ejemplo de diagrama de actividades.

b) Diagrama de casos de uso: Como el tipo de diagrama de diagramas UML más conocido, los diagramas de casos de uso ofrecen una visión general de los actores involucrados en un sistema, las diferentes funciones que necesitan esos actores y cómo interactúan estas diferentes funciones. Es un gran punto de partida para cualquier discusión del proyecto, ya que se pueden identificar fácilmente los principales actores involucrados y los principales procesos del sistema.

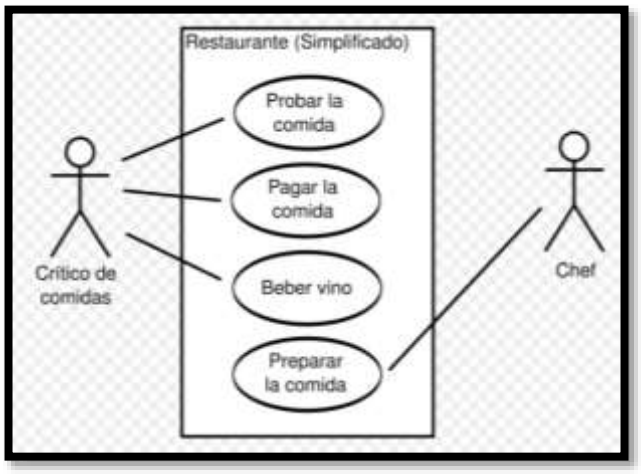


Imagen 2.14: Ejemplo de diagrama de casos de uso.

c) Diagrama de interacción: Los diagramas de interacción incluyen distintos tipos de diagramas:

- **Diagrama de secuencia**: Los diagramas de secuencia en UML muestran cómo los objetos interactúan entre sí y el orden en que se producen esas interacciones. Es importante tener en cuenta

que muestran las interacciones para un escenario en particular. Los procesos se representan verticalmente y las interacciones se muestran como flechas. Los diagramas de secuencia de UML forman parte de un modelo UML y solo existen dentro de los proyectos de modelado UML.

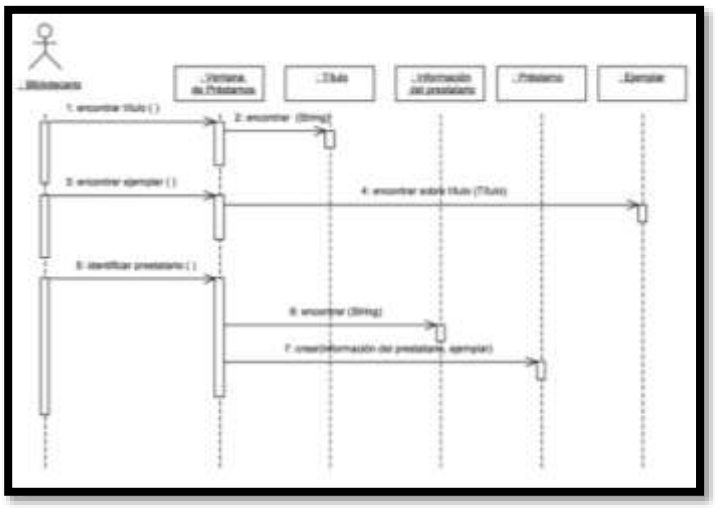


Imagen 2.15: Ejemplo de diagrama de secuencia.

- **Diagrama de comunicación:** El diagrama de comunicación se llamó diagrama de colaboración en UML 1. Es similar a los diagramas de secuencia, pero el foco está en los mensajes pasados entre objetos.

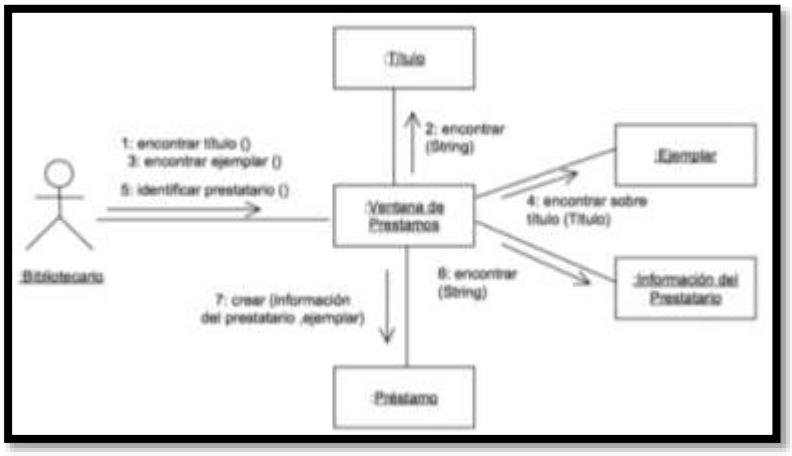


Imagen 2.16: Ejemplo de diagrama de comunicación.

(https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado).

2.4.3. PROGRAMACIÓN ORIENTADA A OBJETOS

La programación orientada a objetos (POO, en español; OOP, según sus siglas en inglés) es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos manipulan los datos de entrada para la obtención de datos de salida específicos, donde cada objeto ofrece una funcionalidad especial.

Muchos de los objetos prediseñados de los lenguajes de programación actuales permiten la agrupación en bibliotecas o librerías, sin embargo, muchos de estos lenguajes permiten al usuario la creación de sus propias bibliotecas.

CARACTERÍSTICAS DE LA PROGRAMACIÓN ORIENTADA A OBJETOS

Existe un acuerdo acerca de qué características contempla la "orientación a objetos". Las características siguientes son las más importantes:

Abstracción

Denota las características esenciales de un objeto, donde se capturan sus comportamientos. Cada objeto en el sistema sirve como modelo de un "agente" abstracto que puede realizar trabajo, informar y cambiar su estado, y "comunicarse" con otros objetos en el sistema sin revelar "cómo" se implementan estas características. Los procesos, las funciones o los métodos pueden también ser abstraídos, y, cuando lo están, una variedad de técnicas son requeridas para ampliar una abstracción. El proceso de abstracción permite seleccionar las características relevantes dentro de un conjunto e identificar comportamientos comunes para definir nuevos tipos de entidades en el mundo real. La abstracción es clave en el proceso de análisis y diseño orientado a objetos, ya que mediante ella podemos llegar a armar un conjunto de clases que permitan modelar la realidad o el problema que se quiere atacar.

Encapsulamiento

Significa reunir todos los elementos que pueden considerarse pertenecientes a una misma entidad, al mismo nivel de abstracción. Esto permite aumentar la cohesión (diseño estructurado) de los componentes del sistema. Algunos autores confunden este concepto con el principio de ocultación, principalmente porque se suelen emplear conjuntamente.

Polimorfismo

Comportamientos diferentes, asociados a objetos distintos, pueden compartir el mismo nombre; al llamarlos por ese nombre se utilizará el comportamiento correspondiente al objeto que se esté usando. O, dicho de otro modo, las referencias y las colecciones de objetos pueden contener objetos de diferentes tipos, y la invocación de un comportamiento en una referencia producirá el comportamiento correcto para el tipo real del objeto referenciado. Cuando esto ocurre en "tiempo de ejecución", esta última característica se llama asignación tardía o asignación dinámica. Algunos lenguajes proporcionan medios más estáticos (en "tiempo de compilación") de polimorfismo, tales como las plantillas y la sobrecarga de operadores de C++.

Herencia

Las clases no se encuentran aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Los objetos heredan las propiedades y el comportamiento de todas las clases a las que pertenecen. La herencia organiza y facilita el polimorfismo y el encapsulamiento, permitiendo a los objetos ser definidos y creados como tipos especializados de objetos preexistentes. Estos pueden compartir (y extender) su comportamiento sin tener que volver a implementarlo. Esto suele hacerse habitualmente agrupando los objetos en clases, y estas en árboles o enrejados que reflejan un comportamiento común. Cuando un objeto hereda de más de una clase, se dice que hay herencia múltiple; siendo de alta

complejidad técnica por lo cual suele recurrirse a la herencia virtual para evitar la duplicación de datos.

Modularidad

Se denomina "modularidad" a la propiedad que permite subdividir una aplicación en partes más pequeñas (llamadas módulos), cada una de las cuales debe ser tan independiente como sea posible de la aplicación en sí y de las restantes partes. Estos módulos se pueden compilar por separado, pero tienen conexiones con otros módulos. Al igual que la encapsulación, los lenguajes soportan la modularidad de diversas formas.

Principio de ocultación

Cada objeto está aislado del exterior, es un módulo natural, y cada tipo de objeto expone una "interfaz" a otros objetos que específica cómo pueden interactuar con los objetos de la clase. El aislamiento protege a las propiedades de un objeto contra su modificación por quien no tenga derecho a acceder a ellas; solamente los propios métodos internos del objeto pueden acceder a su estado. Esto asegura que otros objetos no puedan cambiar el estado interno de un objeto de manera inesperada, eliminando efectos secundarios e interacciones inesperadas. Algunos lenguajes relajan esto, permitiendo un acceso directo a los datos internos del objeto de una manera controlada y limitando el grado de abstracción. La aplicación entera se reduce a un agregado o rompecabezas de objetos.

Recolección de basura

La recolección de basura (garbage collection) es la técnica por la cual el entorno de objetos se encarga de destruir automáticamente, y por tanto desvincular la memoria asociada, los objetos que hayan quedado sin ninguna referencia a ellos. Esto significa que el programador no debe preocuparse por la asignación o liberación de memoria, ya que el entorno la asignará al crear un nuevo objeto y la liberará cuando nadie lo esté usando. En la mayoría de los lenguajes híbridos que se extendieron para soportar el

Paradigma de Programación Orientada a Objetos como C++ u Object Pascal, esta característica no existe y la memoria debe desasignarse expresamente.

TIPOS DE PROGRAMACIÓN ORIENTADA A OBJETOS

Para realizar programación orientada a objetos existen dos corrientes principales:

a) Basada en clases.- Es la más ampliamente usada por los lenguajes de programación orientada a objetos. Por ejemplo es usada por Java, C++ y C#. Se basa en crear una estructura molde llamada clase donde se especifican los campos y métodos que tendrán nuestros objetos. Cada vez que necesitamos un objeto creamos una instancia (o copia del objeto) usando la clase como molde.

b) Basada en prototipos.- Es soportado en Javascript, Python y Ruby. No hay clases, solo hay objetos. El mecanismo para la reutilización está dado por la clonación de objetos. Se crean directamente objetos y cuando se quiere generar otro con la misma estructura se usa clonación. Una vez clonado si queremos podemos agregar los campos y métodos necesarios. Un objeto prototípico es un objeto que se utiliza como una plantilla a partir de la cual se obtiene el conjunto inicial de propiedades de un objeto. Cualquier objeto puede ser utilizado como el prototipo de otro objeto, permitiendo al segundo objeto compartir las propiedades del primero. (https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos).

2.4.4. NETBEANS

NetBeans es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDE1 es un producto libre y gratuito sin restricciones de uso.

NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento. Sun MicroSystems

fundó el proyecto de código abierto NetBeans en junio del 2000 y continúa siendo el patrocinador principal de los proyectos (Actualmente Sun Microsystems es administrado por Oracle Corporation). (https://es.wikipedia.org/wiki/NetBeans).

2.4.5. JAVA (LENGUAJE DE PROGRAMACIÓN)

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling, de Sun Microsystems (constituida en 1982 y posteriormente adquirida el 27 de enero de 2010 por la compañía Oracle) y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son compiladas a bytecode (clase Java), que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

(https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)).

2.4.6. phpMyAdmin

phpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando un navegador web. Actualmente puede crear y eliminar Bases de Datos, crear, eliminar y alterar tablas, borrar, editar y añadir campos, ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 72 idiomas.

(https://es.wikipedia.org/wiki/PhpMyAdmin).

2.4.7. BASE DE DATOS (MySQL)

MySQL es un sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base datos de código abierto más popular del mundo, y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web. (https://es.wikipedia.org/wiki/MySQL).

CAPÍTULO III

CAPÍTULO 3: METODOLOGÍA DE LA INVESTIGACIÓN

3.1 METODOLOGÍA Y TÉCNICAS UTILIZADAS

3.1.1. TIPO Y NIVEL DE INVESTIGACIÓN

Los tipos de investigación utilizados para la presente tesis son:

a) Investigación Aplicada (denominada también **activa o dinámica),** éste tipo de investigación, consiste en la utilización de los conocimientos en la práctica, para aplicarlos, en la mayoría de los casos, en provecho de la sociedad. También podemos indicar que la investigación aplicada se encuentra estrechamente relacionada con la investigación pura, pues depende de los resultados y avances de ésta última.

b) Investigación Tecnológica, éste tipo de investigación tiene como objetivo la solución de problemas prácticos, la cual implica la intervención o transformación de la propia realidad, que se manifiesta en el diseño de nuevos productos.

c) Investigación cualitativa y cuantitativa (denominada también de tipo **híbrido o mixto),** éste tipo de investigación constituye la combinación del enfoque cualitativo y cuantitativo, puesto que no solo implica técnicas de descripción, observación y entrevistas, sino también, utiliza la medición numérica, conteo estadístico y encuestas para establecer con exactitud patrones de comportamiento en una población.

3.1.2. NIVEL DE INVESTIGACIÓN

El nivel de la investigación utilizada en la presente tesis es de tipo:

a) Descriptiva: Comprende la descripción, registro, análisis e interpretación del desarrollo de las valorizaciones de obras aplicando un software.

3.1.3. MÉTODO DE INVESTIGACIÓN

Deductivo: El método deductivo consiste en el estudio, en la investigación de la realidad que partiendo de lo general avanza hacia lo particular; es decir, que a partir de los conocimientos generales, de los principios generales, estudia los casos particulares o específicos.

3.1.4. DISEÑO DE INVESTIGACIÓN

El diseño de la investigación utilizado en la presente tesis es de sucesión o en línea, conocido también como pre-test y post-test con un solo grupo.

Se puede representar mediante la siguiente simbología: O, M_1, X, M_2.

O: M_1 ------ X ------ M_2

Donde:

- O: Objeto de estudio o unidad de análisis (pre experimental).
- M_1: Medición de la variable dependiente antes de aplicar la variable independiente (pre prueba).
- X: Aplicación de la variable independiente.
- M_2: Medición de la variable dependiente después de aplicar la variable independiente (post prueba).

Para la contrastación de la hipótesis mediante este diseño de investigación, se requiere la realización de cuatro pasos:

1. Edición de la variable dependiente antes de aplicar la variable independiente (Pre-Test).
2. Aplicación de la variable independiente (X).

3. Medición de la variable dependiente después de aplicar la variable independiente (Post-Test).
4. Finalmente comparar los resultados, para analizar las diferencias y obtener conclusiones respecto a la importancia o no, después del experimento realizado.

3.1.5. TÉCNICAS DE RECOLECCIÓN Y TRATAMIENTO DE DATOS

A) FUENTES, TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS.

Fuentes:

Fuentes primarias: Son los datos que serán recolectados producto de la visita a las distintas obras públicas.

Fuentes secundarias: Ley 30225 con modificatorias D.L. 1341 y D.L. 1444 y su Reglamento aprobado con D.S. 344-2018-EF, libros, material electrónico, internet.

Técnicas:

Cuestionario: Preguntas de las encuestas para Pre Test y Post Test.

Instrumento:

Encuesta: Instrumento utilizado en la presente investigación.

B) PROCESAMIENTO Y PRESENTACIÓN DE DATOS

Los datos obtenidos se procesaron de la siguiente manera:

- ✓ Visita a las distintas obras públicas.
- ✓ Entrega de los cuestionarios de pre test.
- ✓ Demostracion del desarrollo de valorizaciones de obras publicas por contrata, utilizando el software V&L.
- ✓ Entrega de los cuestionarios de post test.
- ✓ Agrupamiento de respuestas o datos utilizando la escala para medir actitudes conocida como escala de Likert.

Para la presentación de datos se utilizó cuadros estadísticos.

3.2 POBLACIÓN Y MUESTRA

3.2.1 POBLACIÓN

La población fue referencial y está constituida por 74 Obras Públicas de la Provincia de Coronel Portillo, las cuales fueron obtenidas del Sistema Electrónico de Contrataciones del Estado (SEACE), correspondientes al año 2018 y 2019.

3.2.2 MUESTRA

El método de muestreo utilizado, fue el no probabilístico, específicamente el muestreo intencional o de conveniencia, para ello se tomó como unidad de análisis ocho (08) obras públicas. Para la validación de la hipótesis, se consideró como muestra a 32 profesionales de las distintas obras públicas visitadas, los cuales se pueden distribuir como sigue:

Cuadro N° 01: Plantel técnico de obra.

N°	CARGO	CANTIDAD
1	Ingeniero Residente	01
2	Asistente de Residente	01
3	Supervisor o Inspector de obra, según corresponda	01
4	Asistente de Supervisión	01
Subtotal por cada obra		**04**
Total de la muestra (08 x 04)		**32**

Fuente: Elaboración propia.

3.3 OBRAS SELECCIONADAS PARA LA APLICACIÓN DEL SOFTWARE

Para la presente investigación se realizó la visita de campo respectiva, con la finalidad de encuestar al plantel técnico de las distintas obras en ejecución de la Provincia de Coronel Portillo. Asimismo, se encuestó a los funcionarios

y al plantel técnico de dos (02) Entidades Públicas (Municipalidad Distrital de Yarinacocha y Universidad Nacional de Ucayali), debido a que no todos los profesionales del plantel técnico de las obras accedieron a colaborar con las encuestas.

Por otra parte, se indica que las obras seleccionadas para las encuestas fueron convocadas por las siguientes Entidades:

- Gobierno Regional de Ucayali.
- Municipalidad Provincial de Coronel Portillo.
- Municipalidad Distrital de Yarinacocha.

A continuación se describen las fichas técnicas de cada una de las obras encuestadas:

a) OBRA N° 01: AA.HH. GUADALUPE.

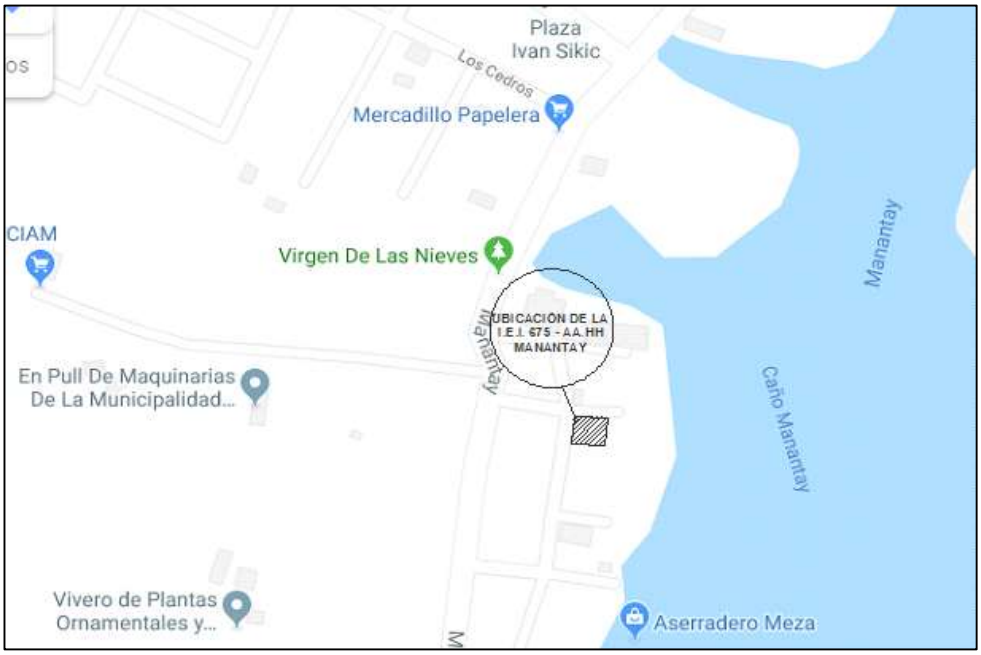


Imagen 3.1: Ubicación de la Obra N° 01 – AA.HH. Guadalupe.

FICHA TÉCNICA DE OBRA

OBRA	:	**"CREACIÓN DE LOS SERVICIOS DE EDUCACION INICIAL ESCOLARIZADAS EN LA I.E.I N° 674, N° 677, N° 676, N° 678, N° 675, N° 696, DISTRITO DE MANANTAY, PROVINCIA DE CORONEL PORTILLO - REGIÓN UCAYALI"**
SUBPRESUPUESTO 005	:	**I.E.I. N° 675 - AA. HH. GUADALUPE**
ENTIDAD	:	GOBIERNO REGIONAL DE UCAYALI
FECHA DEL CONCURSO	:	12 DE DICIEMBRE DEL 2018
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 1,928,001.39**
ADJUDICACION DE LA BUENA PRO	:	17 DE DICIEMBRE DEL 2016
CONTRATISTA	:	CONSORCIO MANANTAY
MONTO CONTRATADO (CON IGV)	:	**S/. 1,928,001.39**
CONTRATO DE OBRA	:	N° 006-2019-GRU
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
FECHA DE FIRMA DEL CONTRATO	:	01/02/2019
FECHA DE ENTREGA DE TERRENO	:	19/02/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	15/01/2019
MONTO ADELANTO DIRECTO S/.	:	**S/ 192,800.14 INC. IGV**
FECHA ENTREGA ADELANTO MATERIALES	:	**NO SE SOLICITÓ**
PLAZO DE EJECUCIÓN	:	**150 dias calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	20/02/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	19/07/2019
REPRESENTANTE LEGAL	:	GUSTAVO ADOLFO REYES BARDALES
RESIDENTE DE OBRA	:	ING. JUAN LUIS SUAREZ REYES
JEFE DE SUPERVISIÓN	:	ING. OSWALDO ARTURO PECHE SORIANO

Imagen 3.2: Ficha Técnica de la Obra N° 01 – AA.HH. Guadalupe.

b) OBRA N° 02: AA.HH. ISABEL LA CATÓLICA.

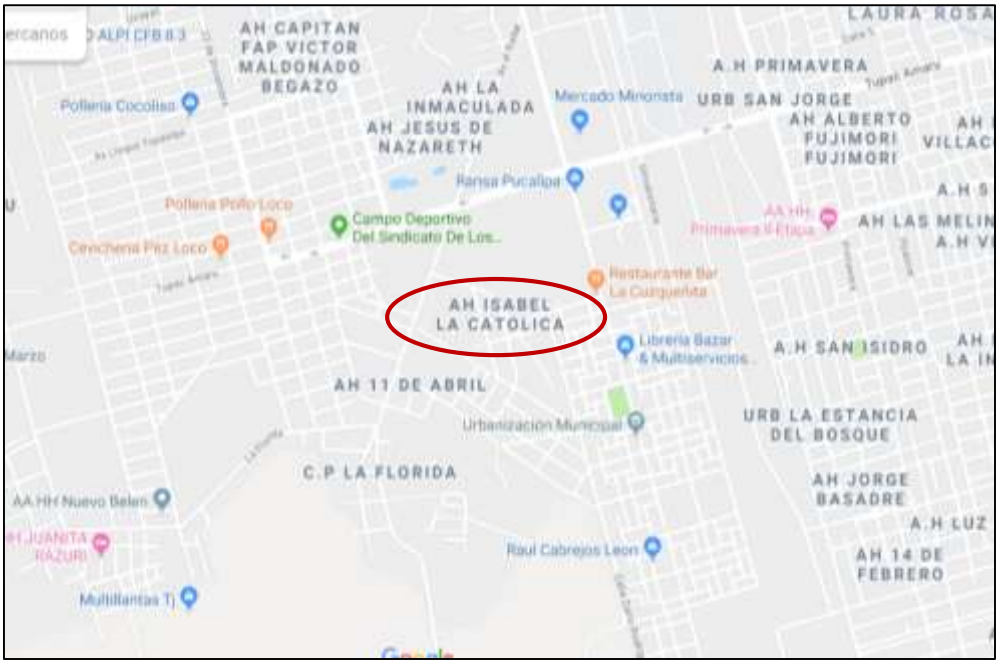


Imagen 3.3: Ubicación de la Obra N° 02 – AA.HH. Isabel La Católica.

FICHA TÉCNICA DE OBRA

OBRA	:	**"CREACIÓN DE LOS SERVICIOS DE EDUCACION INICIAL ESCOLARIZADAS EN LA I.E.I N° 674, N° 677, N° 676, N° 678, N° 675, N° 696, DISTRITO DE MANANTAY, PROVINCIA DE CORONEL PORTILLO - REGIÓN UCAYALI"**
SUBPRESUPUESTO 004	:	**IEI N° 678 - AA. HH. ISABEL LA CATOLICA**
ENTIDAD	:	GOBIERNO REGIONAL DE UCAYALI
FINANCIAMIENTO	:	RECURSOS POR OPERACIÓN OFICIALES DE CREDITO
MODALIDAD DEL CONCURSO	:	ADJUDICACION SIMPLIFICADA N° 047-2018-GRU-GR-CS
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 2,725,548.31**
CONTRATO DE EJECUCIÓN DE OBRA N°	:	0006-2019-GRU-GGR
MODALIDAD DE EJECUCIÓN	:	LLAVE EN MANO
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
FECHA DE FIRMA DEL CONTRATO	:	**01/02/2019**
CONTRATISTA	:	CONSORCIO MANANTAY
MONTO CONTRATADO (INC IGV)	:	**S/ 2,725,548.31**
FECHA DE ENTREGA DE TERRENO	:	19/02/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	15/02/2019
MONTO ADELANTO DIRECTO S/.	:	**S/ 272,554.83 INC IGV**
FECHA ENTREGA ADELANTO MATERIALES	:	28/03/2019
MONTO ADELANTO MATERIALES S/.	:	**S/ 571,367.85 INC IGV**
PLAZO DE EJECUCIÓN	:	**150 días calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	20/02/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	19/07/2019
RESIDENTE DE OBRA	:	ING. GUSTAVO P. P. MIÑOPE RODRIGUEZ
JEFE DE SUPERVISIÓN	:	ING. LUGI TORRES HOYLE

Imagen 3.4: Ficha Técnica de la Obra N° 02 – AA.HH. Isabel La Católica.

c) OBRA N° 03: AA.HH. JUAN SILVA BOCANEGRA.

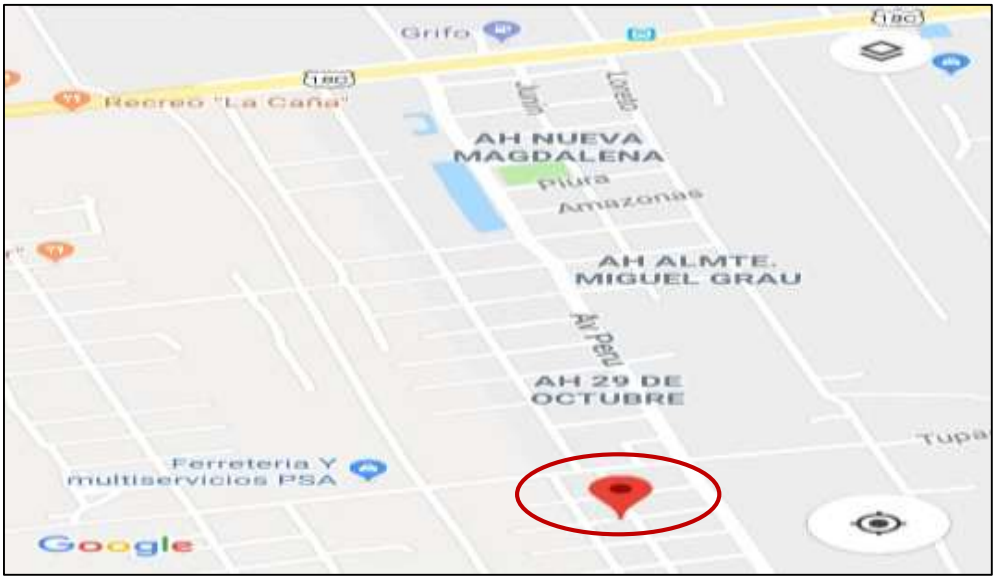


Imagen 3.5: Ubicación de la Obra N° 03 – AA.HH. Juan Silva Bocanegra.

FICHA TÉCNICA DE OBRA

OBRA	:	"CREACIÓN DE LOS SERVICIOS DE EDUCACION INICIAL ESCOLARIZADAS EN LA I.E.I N° 674, N° 677, N° 676, N° 678, N° 675, N° 696, DISTRITO DE MANANTAY, PROVINCIA DE CORONEL PORTILLO - REGIÓN UCAYALI"
SUBPRESUPUESTO 006	:	I.E.I.. N° 696- AA.HH. JUAN SILVA BOCANEGRA
ENTIDAD	:	GOBIERNO REGIONAL DE UCAYALI
FINANCIAMIENTO	:	RECURSOS POR OPERACIÓN OFICIALES DE CREDITO
MODALIDAD DEL CONCURSO	:	ADJUDICACION SIMPLIFICADA N° 047-2018-GRU-GR-CS
MONTO VALOR REFERENCIAL (CON IGV)	:	S/ 1,554,298.29
CONTRATO DE EJECUCIÓN DE OBRA N°	:	0006-2019-GRU-GGR
MODALIDAD DE EJECUCIÓN	:	LLAVE EN MANO
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
FECHA DE FIRMA DEL CONTRATO	:	01/02/2019
CONTRATISTA	:	CONSORCIO MANANTAY
MONTO CONTRATADO (INC IGV)	:	S/ 1,554,298.29
FECHA DE ENTREGA DE TERRENO	:	19/02/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	13/02/2019
MONTO ADELANTO DIRECTO S/.	:	S/ 155,429.83 INC IGV
FECHA ENTREGA ADELANTO MATERIALES	:	NO SE SOLICITÓ
MONTO ADELANTO MATERIALES S/.	:	NO SE SOLICITÓ
PLAZO DE EJECUCIÓN	:	150 dias calendarios
INICIO DEL PLAZO DE EJECUCIÓN	:	20/02/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	19/07/2019
RESIDENTE DE OBRA	:	ING. WAGNER GODOY GONZALES
JEFE DE SUPERVISIÓN	:	ING. RODOLFO CARLOS VILLAIZAN GOYAS

Imagen 3.6: Ficha Técnica de la Obra N° 03 – AA.HH. Juan Silva Bocanegra.

d) OBRA N° 04: AA.HH. LA PERLA.

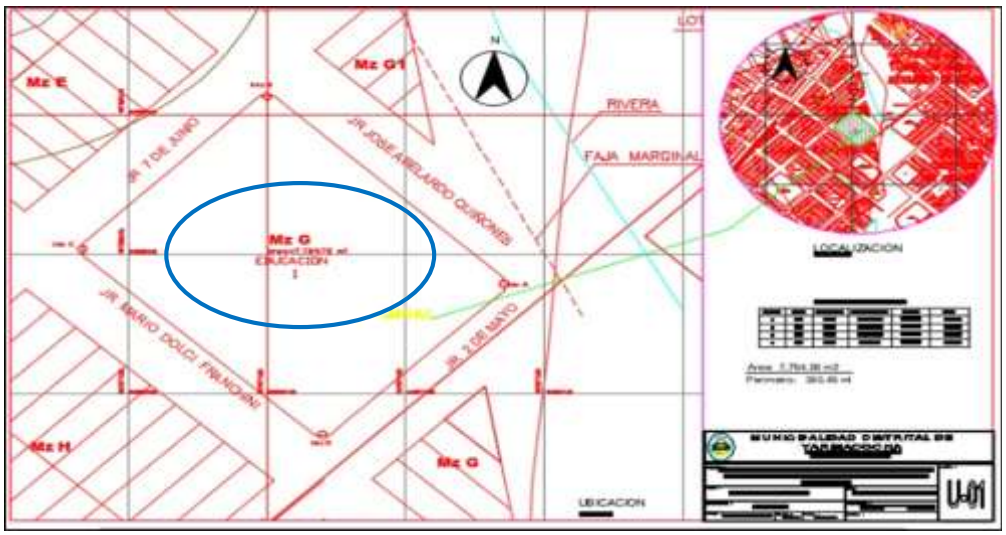


Imagen 3.7: Ubicación de la Obra N° 04 – AA.HH. La Perla.

FICHA TÉCNICA DE OBRA

OBRA	:	**MEJORAMIENTO DE LOS SERVICIOS COMPLEMENTARIOS EN LA INSTITUCION EDUCATIVA N°64668 AA.HH LA PERLA, DISTRITO DE YARINACOCHA - CORONEL PORTILLO - UCAYALI**
ENTIDAD	:	MUNICIPALIDAD DISTRITAL DE YARINACOCHA
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 315,638.49**
CONTRATO DE EJECUCION DE OBRA	:	**CONTRATO N° 003 - 2019 - MDY - GM**
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
CONTRATISTA	:	SL & JF INGENIERÍA Y CONSTRUCCIONES E.I.R.L.
MONTO CONTRATADO (SIN IGV)	:	**S/ 262,942.93**
FECHA DE ENTREGA DE TERRENO	:	05/06/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	**NO SE SOLICITÓ**
MONTO ADELANTO DIRECTO S/.	:	**NO SE SOLICITÓ**
FECHA ENTREGA ADELANTO MATERIALES	:	**NO SE SOLICITÓ**
MONTO ADELANTO MATERIALES S/.	:	**NO SE SOLICITÓ**
PLAZO DE EJECUCIÓN	:	**60 días calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	06/06/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	04/08/2019
RESIDENTE DE OBRA	:	ING. VALLES PEREZ DANY ROMY REG. CIP. 94297
JEFE DE SUPERVISIÓN	:	ING. MILOVAN KUCH MORI CIP. 172176

Imagen 3.8: Ficha Técnica de la Obra N° 04 – AA.HH. La Perla.

e) OBRA N° 05: CC.NN. NUEVO SAN JUAN.



Imagen 3.9: Localización de la Obra N° 05 – CC.NN. Nuevo San Juan.



Imagen 3.10: Ubicación de la Obra N° 05 – CC.NN. Nuevo San Juan.

FICHA TÉCNICA DE OBRA

OBRA	:	**"MEJORAMIENTO A NIVEL DE AFIRMADO DEL JR. NUEVO SAN JUAN, JR. ROGELIO FERNANDEZ RUIZ Y JR. PARIS EN LA CC.NN. NUEVO SAN JUAN KM.13.500 C.F.B. DISTRITO DE YARINACOCHA - CORONEL PORTILLO - UCAYALI"**
ENTIDAD	:	MUNICIPALIDAD DISTRITAL DE YARINACOCHA
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 1,324,649.58**
CONTRATO DE EJECUCIÓN DE OBRA	:	**CONTRATO N° 002 - 2019 - MDY - GM**
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
CONTRATISTA	:	CONSORCIO SAN JUAN
MONTO CONTRATADO (SIN IGV)	:	**S/ 1,122,584.39**
FECHA DE ENTREGA DE TERRENO	:	23/05/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	**NO SE SOLICITÓ**
MONTO ADELANTO DIRECTO S/.	:	**NO SE SOLICITÓ**
FECHA ENTREGA ADELANTO MATERIALES	:	**NO SE SOLICITÓ**
MONTO ADELANTO MATERIALES S/.	:	**NO SE SOLICITÓ**
PLAZO DE EJECUCIÓN	:	**60 días calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	24/05/2019
TERMINO DEL PLAZO CONTRACTUAL	:	22/07/2019
RESIDENTE DE OBRA	:	ING. ABRAHAM ANGELINO VARGAS BEDOYA REG. CIP. 29721
JEFE DE SUPERVISIÓN	:	ING. BLADIMIR FLAVIO VELEZ DE VILLA ALVARADO REG. CIP. 125435

Imagen 3.11: Ficha Técnica de la Obra N° 05 – CC.NN. Nuevo San Juan.

f) OBRA N° 06: AFIRMADO EN JR. VENECIA.



Imagen 3.12: Ubicación de la Obra N° 06 – Afirmado en el Jr. Venecia.

FICHA TÉCNICA DE OBRA

OBRA	:	**"MEJORAMIENTO A NIVEL DE AFIRMADO DEL JR. VENECIA HASTA CALLE EL ALAMO; INTERS.CALLE EL ALAMO - JR.SABINO ASIPALI HASTA CALLE LOS CIPRECES; INTERS. CALLE EL ALAMO - JR. FRANCISCO DEL AGUILA HASTA CALLE LOS CIPRECES EN LA JUNTA VECINAL EL ALAMO, DISTRITO DE YARINACOCHA - CORONEL PORTILLO - UCAYALI"**
ENTIDAD	:	MUNICIPALIDAD DISTRITAL DE YARINACOCHA
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 545,451.15**
CONTRATO DE EJECUCIÓN DE OBRA	:	**CONTRATO N° 001 - 2019 - MDY - GM**
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
CONTRATISTA	:	CONSORCIO VENECIA
MONTO CONTRATADO (SIN IGV)	:	**S/ 462,246.74**
FECHA DE ENTREGA DE TERRENO	:	25/05/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	**NO SE SOLICITÓ**
MONTO ADELANTO DIRECTO S/.	:	**NO SE SOLICITÓ**
FECHA ENTREGA ADELANTO MATERIALES	:	**NO SE SOLICITÓ**
MONTO ADELANTO MATERIALES S/.	:	**NO SE SOLICITÓ**
PLAZO DE EJECUCIÓN	:	**75 días calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	26/05/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	08/08/2019
RESIDENTE DE OBRA	:	ING. ALVARO VELA SANCHEZ REG. CIP. 43785
JEFE DE SUPERVISIÓN	:	ING. REMBRANDT DE LA CRUZ MONTALVAN REG. CIP. 78190

Imagen 3.13: Ficha Técnica de la Obra N° 06 – Afirmado en el Jr. Venecia.

g) OBRA N° 07: CASERÍO SAN LORENZO.



Imagen 3.14: Ubicación de la Obra N° 07 – Caserío San Lorenzo.

FICHA TÉCNICA DE OBRA

OBRA	:	**"CREACIÓN DE UN POZO TUBULAR Y TANQUE ELEVADO EN EL CASERÌO SAN LORENZO - DISTRITO DE YARINACOCHA - CORONEL PORTILLO - UCAYALI "**
ENTIDAD	:	MUNICIPALIDAD DISTRITAL DE YARINACOCHA
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 576,932.24**
CONTRATO DE EJECUCIÓN DE OBRA	:	**CONTRATO N° 006 - 2019 - MDY - GM**
SISTEMA DE CONTRATACIÓN	:	SUMA ALZADA
CONTRATISTA	:	OB INGENIEROS CONTRATISTA S.R.L.
MONTO CONTRATADO (CON IGV)	:	**S/ 680,760.04**
FECHA DE ENTREGA DE TERRENO	:	15/06/2019
MONTO ADELANTO DIRECTO S/.	:	**S/ 68,076.00 CON IGV**
MONTO ADELANTO MATERIALES S/.	:	**S/ 136,152.01 CON IGV**
PLAZO DE EJECUCIÓN	:	**90 días calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	09/07/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	06/10/2019
RESIDENTE DE OBRA	:	ING. ELVIN DAVID POLO LOPEZ REG. CIP. 85406
JEFE DE SUPERVISIÓN	:	ING. JUAN JESUS OCAÑA APONTE REG. CIP. 125103

Imagen 3.15: Ficha Técnica de la Obra N° 07 – Caserío San Lorenzo.

h) OBRA N° 08: JR. AREQUIPA.



Imagen 3.16: Ubicación de la Obra N° 08 – Jr. Arequipa.

FICHA TÉCNICA DE OBRA

OBRA	:	**"MEJORAMIENTO DEL SERVICIO DE AGUA POTABLE Y ALCANTARILLADO SANITARIO DEL JR. AREQUIPA CUADRAS 01, 02 ,03 Y 04, DISTRITO DE CALLERÍA, PROVINCIA DE CORONEL PORTILLO, UCAYALI"**
ENTIDAD	:	MUNICIPALIDAD PROVINCIAL DE CORONEL PORTILLO
MONTO VALOR REFERENCIAL (CON IGV)	:	**S/ 1,126,898.82**
CONTRATO DE EJECUCIÓN DE OBRA	:	**CONTRATO N° 001 - 2019 - MPCP**
SISTEMA DE CONTRATACIÓN	:	PRECIOS UNITARIOS
CONTRATISTA	:	CONSORCIO REY JESUS
MONTO CONTRATADO (SIN IGV)	:	**S/ 859,499.10**
FECHA DE ENTREGA DE TERRENO	:	02/05/2019
FECHA ENTREGA DEL ADELANTO DIRECTO	:	**NO SE SOLICITÓ**
MONTO ADELANTO DIRECTO S/.	:	**NO SE SOLICITÓ**
FECHA ENTREGA ADELANTO MATERIALES	:	**NO SE SOLICITÓ**
MONTO ADELANTO MATERIALES S/.	:	**NO SE SOLICITÓ**
PLAZO DE EJECUCIÓN	:	**75 días calendarios**
INICIO DEL PLAZO DE EJECUCIÓN	:	03/05/2019
TÉRMINO DEL PLAZO CONTRACTUAL	:	16/07/2019
RESIDENTE DE OBRA	:	ING. ELMER JHONATAN ROJAS CIENFUEGOS REG. CIP. 133184
JEFE DE SUPERVISIÓN	:	ING. TED ROLY GAMARRA MEZA REG. CIP. 88901

Imagen 3.17: Ficha Técnica de la Obra N° 08 – Jr. Arequipa.

CAPITULO IV

CAPÍTULO 4: SOFTWARE PROPUESTO PARA EL CÁLCULO DE VALORIZACIONES EN OBRAS PÚBLICAS

4.1. FASE INICIAL

A. Documento Visión del Negocio

A.1. Introducción

A.1.1. Oportunidad del Negocio

Las entidades Públicas y empresas contratistas de hoy, reconocen que un software es una herramienta fundamental y muy útil para el desarrollo de las valorizaciones de obras públicas. Por tal motivo, es necesario que todos los profesionales, empresas y entidades públicas conozcan de un software que les permita desarrollar las valorizaciones con un interfaz amigable, resultados óptimos y un corto tiempo en su proceso.

A.1.2. Exposición del Problema

Cuadro N° 02: Exposición del Problema.

Realidad Problemática	a) Limitación fundamental al usar el Excel. b) Limitación fundamental para el módulo de valorizaciones del S10. c) Limitaciones para los otros programas.
Análisis del problema	a) En el Perú existen diversas empresas dedicadas a la ejecución de obras públicas y cada una de estas tiene una forma distinta de valorizar, esto trae como consecuencia diversidad de criterios, incluyendo el uso de formatos muy variados y el empleo de tiempos diferentes.

	b) Se han encontrado limitaciones no solo en el Excel, sino también en otros programas que son menos accesibles como el módulo de valorizaciones del s10, el K&BCOP, el CPOC y el DELPHIN EXPRESS.
Afecta a	a) Ingenieros Residentes. b) Asistentes de Residentes. c) Ingenieros supervisores o Inspectores. d) Asistentes de Supervisores.
El impacto	a) Perdida de tiempo al usar el Excel, debido a la variedad de diversos criterios y formatos. b) Desarrollo incompleto de las valorizaciones con otros software.
Solución Exitosa	a) Diseñar e implementar un software que satisfaga las expectativas de los usuarios, sobre todo en el ahorro de tiempo cuando se desarrollan valorizaciones de obras públicas. b) Contar con cálculos confiables y verídicos para lograr un control económico adecuado.

Fuente: Elaboración propia.

A.2. Descripción de los Stakeholders y usuarios

A.2.1. Mercado

El software V&L será utilizado por aquellos profesionales que estén vinculados al control económico de las obras (valorizaciones de obras), los cuales pueden ser ingenieros residentes, asistente de residencia por parte de empresas contratistas, supervisores y/o inspectores de obras por parte de la Entidad.

El software tendrá las siguientes interfaces: Valorización de obra principal, Valorización de obra adicional, Valorización de Mayores Gastos Generales y Valorización por Intereses de demora de pago, índices de precios, impresión,

procesamiento de cálculo, reportes de resultados, resúmenes de cálculo y lista de proyectos creados.

A.2.2. Sumario de Stakeholder

Cuadro N° 03: Sumario de Stakeholder.

Nombre	Representantes	Rol
Tesistas	a) Daniel Alcides Ventura Rojas. b) Giamfranco Chávez Sifuentes.	Encargados de administrar, modificar y dar soporte al software V&L.

Fuente: Elaboración propia.

A.2.3. Sumario de usuarios

Cuadro N° 04: Sumario de usuarios.

Nombre	Descripción	Stakeholder
Plantel técnico de obra.	a) Ingeniero Residente. b) Asistente de Residente. c) Ingeniero supervisor o Inspector. d) Asistente de Supervisor.	Representa así mismo.

Fuente: Elaboración propia.

A.2.4. Ambiente de usuarios

Plantel técnico de obra: Tendrá acceso al software V&L, mediante un Nick y clave, el cual le permitirá desarrollar la Valorización de obra principal, Valorización de obra adicional, Valorización de Mayores Gastos Generales y Valorización de Intereses por demora de pagos.

A.2.5. Necesidades Principales de los Usuarios y Stakeholders.

Cuadro N° 05: Necesidades Principales del usuario

NECESIDAD	PRIORIDAD	REFERENCIA	SITUACIÓN ACTUAL	SOLUCIÓN PROPUESTA
Calcular la Valorización de obra principal	Alta	Tiempo de Respuesta	El usuario tabula, ordena celdas, registra las partidas de las diferentes especialidades y luego calcula mediante la herramienta Excel.	Implementar un módulo para la valorización de obra principal, que permita calcular en un tiempo óptimo, sin tabular ni ordenar celdas, debido a que tiene incorporado un formato predeterminado.
Calcular la Valorización de obra adicional	Alta	Tiempo de Respuesta	El usuario tabula, ordena celdas, registra las partidas de las diferentes especialidades y luego calcula mediante la herramienta Excel.	Implementar un módulo para la valorización de obra adicional, que permita calcular en un tiempo óptimo, sin tabular ni ordenar celdas, debido a que tiene incorporado un formato predeterminado.
Calcular la Valorización por Mayores Gastos Generales Variables	Alta	Tiempo de Respuesta	El usuario tabula, ordena celdas y registra las ampliaciones de plazo aprobadas sin tener un formato predeterminado en la herramienta Excel.	Implementar un módulo para la valorización por mayores gastos generales variables, que permita calcular en un tiempo óptimo, sin tabular ni ordenar celdas, debido a que tiene incorporado un

				formato predeterminado.
Calcular la Valorización de Intereses por demora de pagos	Alta	Tiempo de Respuesta	El usuario tabula, ordena celdas y registra las fechas de demora de pagos, para luego realizar el cálculo de los intereses, sin tener un formato predeterminado en la herramienta Excel.	Implementar un módulo de valorización de Intereses por demora de pagos que permita calcular de una manera óptima, sin tabular ni ordenar celdas, debido a que tiene incorporado un formato predeterminado.

Fuente: Elaboración propia.

A.3. Objetivos de Modelamiento del Negocio.

Proceso de Inicio

- Crear una nueva obra.
- Registrar los datos generales de la obra creada.
- Seleccionar los presupuestos que conforman la obra creada.
- Cargar los presupuestos y formulas polinómicas que conforman la obra creada.
- Elegir el método para trabajar con los coeficientes de reajustes, es decir, actualizando o sin actualizar los mismos.
- Calcular el monto del adelanto para materiales.

Proceso de Ingreso de Datos

- Ingresar los metrados ejecutados mensuales y los avances programados mensuales de acuerdo al calendario de avance de obra valorizado.
- Ingresar las ampliaciones aprobadas por la Entidad.

- Ingresar las fechas por demoras de pagos, según corresponda.

Proceso Resultados

- Visualizar los resultados de valorización del periodo, resumen de metrados ejecutados mensuales, cálculo de coeficientes de reajuste, cálculo de reintegros, cálculo de las deducciones y amortizaciones de los adelantos, resumen de valorización y curva “S”.
- Visualizar el resultado de la valorización de mayores gastos generales variables en caso haya sido aprobada una ampliación del plazo en la obra.
- Visualizar el resultado de la valorización de intereses ´por demora de pagos.

A.4. Requerimientos

A.4.1. Funcionales

Módulo de Valorización de obra principal

- Crear una nueva obra.
- Registrar los datos generales de la obra creada.
- Actualizar pie de presupuesto, según corresponda.
- Seleccionamos los presupuestos que conforman la obra creada.
- Cargar los presupuestos correspondientes a la obra creada.
- Cargar las formulas polinómicas correspondientes a la obra creada.
- Definir el criterio a utilizar del coeficiente de reajuste.
- Calcular el monto del adelanto para materiales.
- Crear la valorización N° 01 de la obra.

- Registrar los metrados ejecutados mensuales en la valorización del periodo.
- Procesar valorización N° 01 de la obra.
- Registrar los avances programados mensuales de la obra.
- Visualización de resultados:
    1. Resultado del cálculo de los coeficientes de reajuste.
    2. Resultado del cálculo de reajustes o reintegros.
    3. Resultados del cálculo de la deducción del reajuste que no corresponde por el adelanto directo.
    4. Resultados del cálculo de la deducción del reajuste que no corresponde por adelanto para materiales.
    5. Resultados del cálculo de la amortización por adelanto directo.
    6. Resultados del cálculo de la amortización por adelanto para materiales.
    7. Resultado del resumen de la Valorización y monto a facturar.
    8. Resultado del cálculo de la curva “S”.
- Crear valorización mensual “n”, y realizar los pasos descritos líneas arriba.

<u>Módulo de valorización de obra adicional</u>

- Registrar el número de ampliación de plazo correspondiente.
- Registrar el inicio y fin del plazo de ejecución de la obra adicional correspondiente.

- Actualizar pie de presupuesto de la obra adicional, según corresponda.
- Seleccionamos los presupuestos que conforman la obra adicional correspondiente.
- Cargar los presupuestos correspondientes a la obra adicional correspondiente.
- Cargar las fórmulas polinómicas correspondientes a la obra adicional correspondiente.
- Definir el criterio a utilizar del coeficiente de reajuste.
- Crear la valorización N° 01 de la obra adicional correspondiente.
- Registrar los metrados ejecutados mensuales en la valorización del periodo de la obra adicional.
- Procesar valorización N° 01 de la obra adicional correspondiente.
- Registrar los avances programados mensuales de la obra adicional correspondiente.
- Visualización de resultados:
    1. Resultado del cálculo de los coeficientes de reajuste.
    2. Resultado del cálculo de reajustes o reintegros.
    3. Resultado del resumen de la Valorización y monto a facturar.
    4. Resultado del cálculo de la curva “S”.
- Crear valorización mensual “n” de la obra adicional correspondiente, y realizar los pasos descritos líneas arriba.

Módulo de valorización de mayores gastos generales variables

- Registrar ampliaciones de plazo aprobadas por la Entidad.
- Registrar el monto de los Gastos Generales Variables Ofertados.
- Registrar el mes y duración de la causal, según corresponda.
- Visualización de resultados:
    1. Resultado del cálculo de los Mayores Gastos Generales Variables.

Módulo de valorización de intereses por demora de pagos

- Registrar la fecha de pago real de la valorización de obra principal respectiva.
- Registrar los factores acumulados de la tasa de interés legal, según la fecha máxima de pago y la fecha real de pago de las valorizaciones.
- Visualización de resultados:
    1. Resultado del cálculo de los Intereses por demora de pagos en las valorizaciones.

A.4.2. No Funcionales

- El software se desarrolló con lenguaje de programación Java y con base de datos MySQL, esto en razón de que dicho gestor de base de datos es acceso libre.
- La instalación del sistema V&L se puede realizar en una laptop o computadora de sistema operativo Windows 7, 8 y 10, de 64 bits.

- Para la visualización optima de los reportes del sistema V&L, se requiere como mínimo una pantalla de 21 pulgadas.

B. Plan de Desarrollo del Software

B.1. Introducción

B.1.1. Propósito

El objetivo de este plan de Desarrollo de Software es definir las actividades realizadas durante el desarrollo de las fases y las iteraciones requeridas para lograr el Sistema Propuesto.

B.1.2. Entregables del proyecto

Cuadro N° 06: Entregables del proyecto.

FASES	WORKFLOWS	ARTEFACTOS
INICIAL	MODELADO DEL NEGOCIO	Documentos de Visión del negocio. Plan de desarrollo del software. Modelo de Use Case. Modelo del dominio del Problema.
ELABORACIÓN	REQUERIMIENTOS	Modelo Use Case. Diagrama de Actividades.
	ANALISIS Y DISEÑO	Diagrama de colaboración. Diagrama de Paquete de Análisis. Diseño de la Arquitectura Diseño de Interfaces y Diagrama de Secuencias. Diagrama de Clases.

CONSTRUCCIÓN	ANALISIS DE DISEÑO	Diseño de Base de Datos.
	IMPLEMENTACION	Código fuente del sistema V&L.
TRANSICIÓN	PRUEBA	Prueba de Calidad. Prueba de Estado. Purebas de Tipo.

Fuente: Elaboración propia.

B.2. La organización del proyecto

B.2.1. Estructura Orgánica

El equipo de trabajo para el desarrollo del software está comprendido por dos Tesistas bachilleres en Ingeniería Civil y un asesor ingeniero Civil.

B.3. El proceso de Dirección

B.3.1. Estimación del proyecto

Las estimaciones del proyecto son basadas en otros estudios de un desarrollo de software, el tiempo, dedicación aproximado es la base y/o referencia del presupuesto y tiempo para la elaboración del proyecto.

B.3.2. Plan de Proyecto

a) Plan de fase

El software V&L será elaborado usando cuatro fases de iteración.

Las fases y el tiempo se muestran en el siguiente cuadro:

Cuadro N° 07: Plan de proceso de desarrollo de acuerdo a fases.

FASE	EMPIEZA	TERMINA
Fase de inicio (10%)	03/10/2017	17/12/2017
Fase de Elaboración (20%)	18/12/2017	27/05/2018
Fase de Construcción (60%)	28/05/2018	29/05/2019
Fase de Transición (10%)	30/05/2019	28/06/2019

Fuente: Elaboración propia.

b) Horario del proyecto

El horario del proyecto que contiene el nombre de las tareas, las fechas de inicio y fin se muestran a continuación

Cuadro N° 08: Principales tareas del proyecto.

FASE	EMPIEZA	TERMINA
Modelado del negocio	03/10/2017	17/12/2017
Requerimientos	18/12/2017	12/02/2018
Análisis y diseño	13/02/2018	27/05/2018
Implementación	28/05/2018	29/05/2019
Prueba	30/05/2019	28/06/2019

Fuente: Elaboración propia.

B.4. Recurso para el proyecto

B.4.1. Plan de Adquisición de Recursos.

Las obras públicas visitadas y los Tesistas aportarán la información necesaria para lograr el objetivo de la investigación.

B.5. Presupuesto

A continuación, se detalla la relación de gastos utilizado para el desarrollo de la investigación.

Cuadro N° 09: Presupuesto del proyecto.

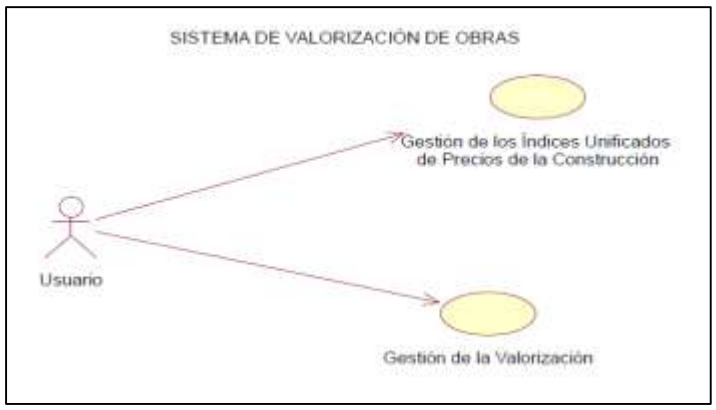
DESCRIPCION	UNIDAD	CANTIDAD	COSTO UNITARIO	COSTO TOTAL
BIENES:				**199.00**
Compra de bibliografía	und	5.00	30.00	150.00
Lapiceros	docenas	3.00	12.00	36.00
Papeles	millar	0.50	26.00	13.00
SERVICIOS:				**5905.00**
Internet	meses	24.00	55.00	1320.00
Energía eléctrica	meses	24.00	35.00	840.00
Impresión	hojas	1500.00	0.30	450.00
Empastado	und	3.00	35.00	105.00
Movilidad local	dia	730.00	3.00	2190.00
Viáticos-otros	global	1.00	1000.00	1000.00
PERSONAL				**3600.00**
Asesoría para el desarrollo del sistema V&L	meses	24.00	150.00	3600.00
SUBTOTAL				**S/. 9,704.00**
Imprevistos (10%)				**S/. 970.40**
TOTAL GENERAL:				**S/. 10,674.40**

Fuente: Elaboración propia.

B.6. Vistas Use case

B.6.1. Modelo Use Case del negocio.

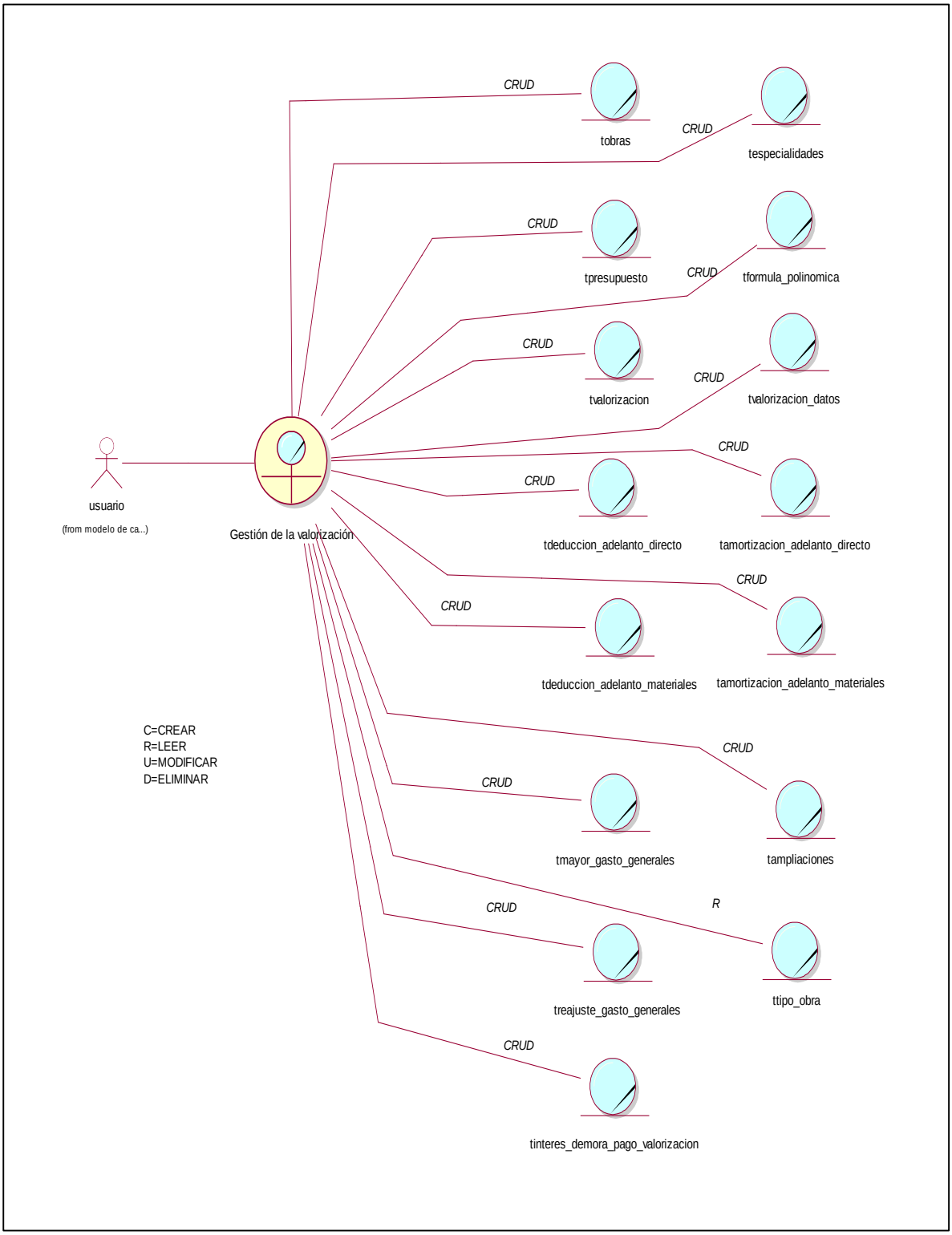
Diagrama 01: Modelo Use Case del Negocio.



Fuente: Elaboración propia.

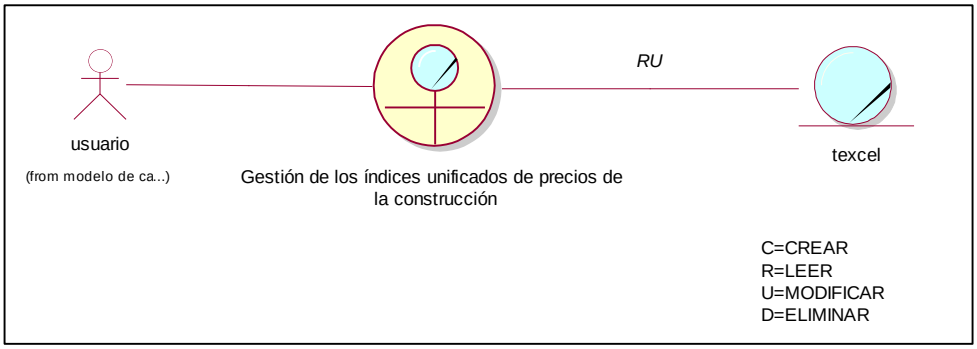
B.6.2. Modelo del objeto del negocio.

Diagrama 02: Diagrama de gestión de la valorización.



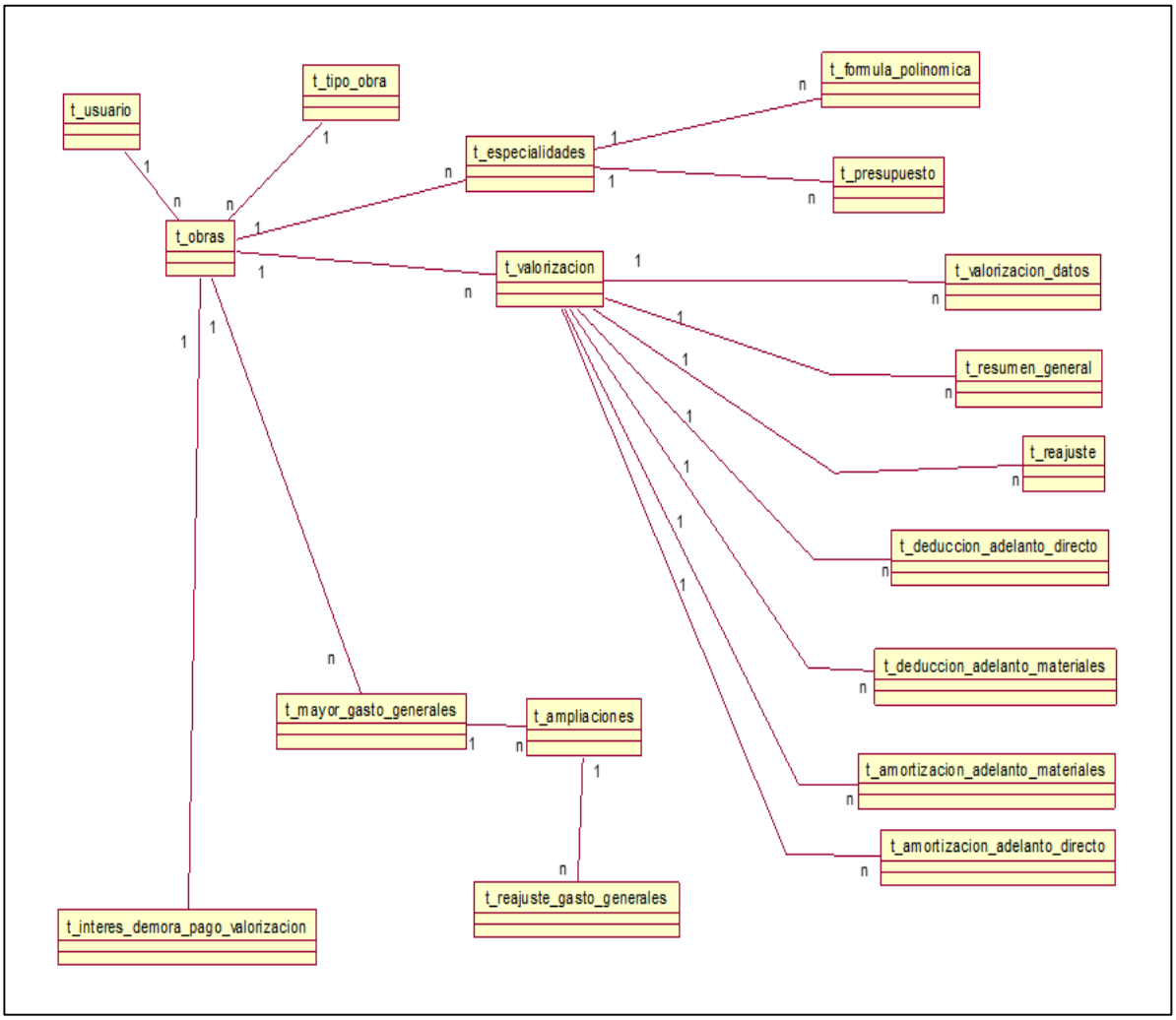
Fuente: Elaboración propia.

Diagrama 03: Diagrama de la gestión de los índices.



Fuente: Elaboración propia.

Diagrama 04: Dominio del problema.



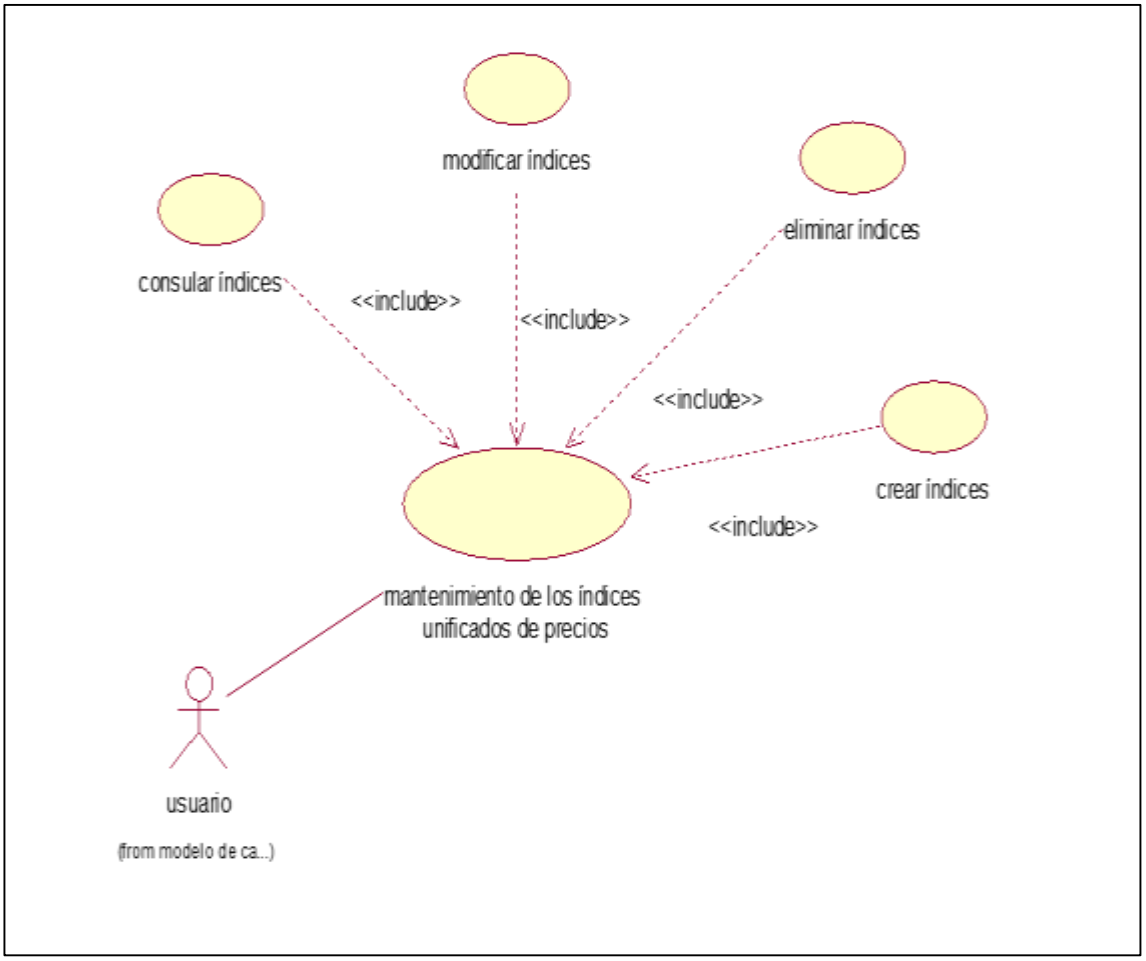
Fuente: Elaboración propia.

4.2. FASE DE ELABORACIÓN

4.2.1. Requerimientos

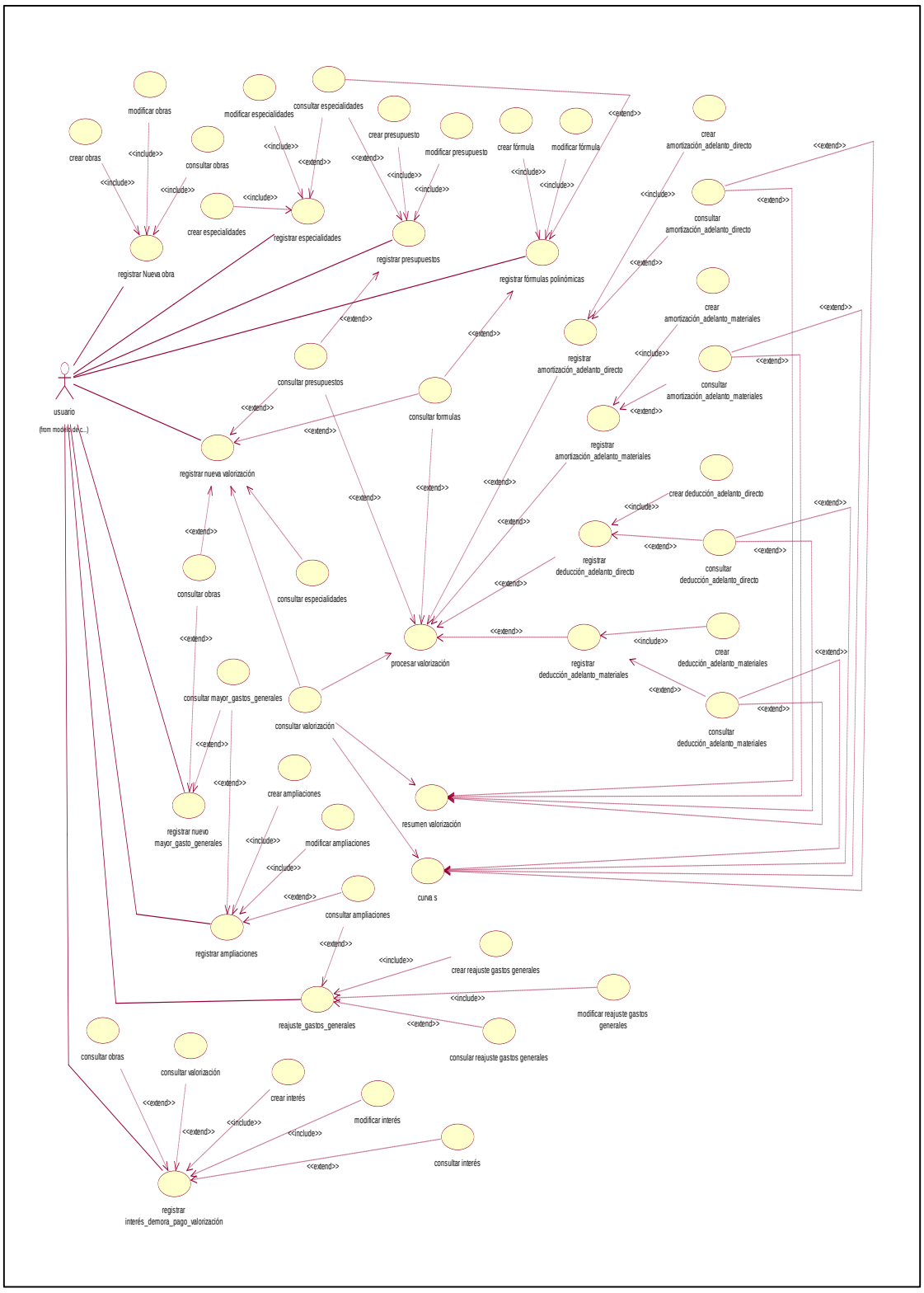
A) Modelo Use Case

Diagrama 05: Diagrama de requerimiento - gestión de índices.



Fuente: Elaboración propia.

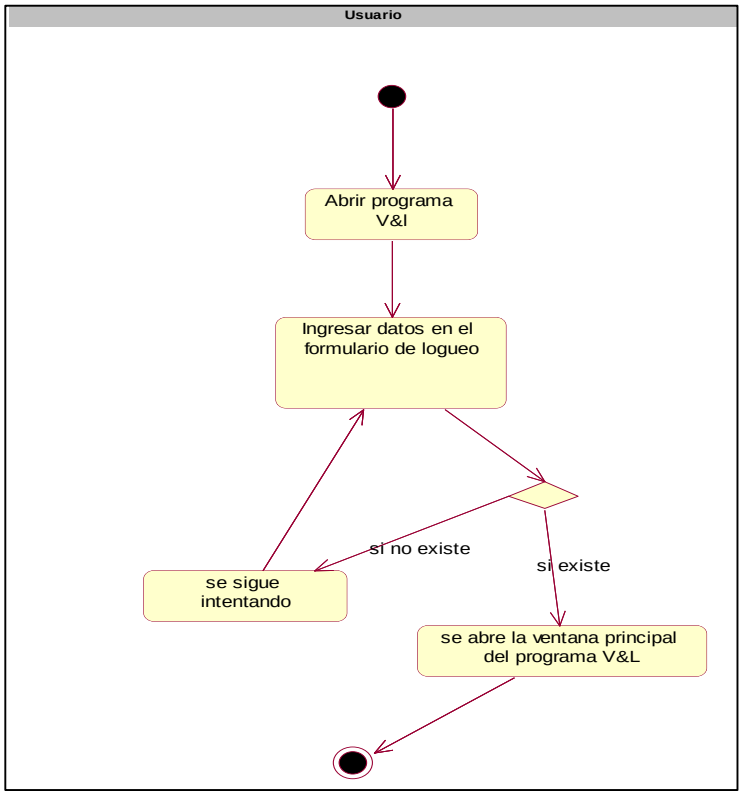
Diagrama 06: Diagrama de requerimiento - gestión de valorización.



Fuente: Elaboración propia.

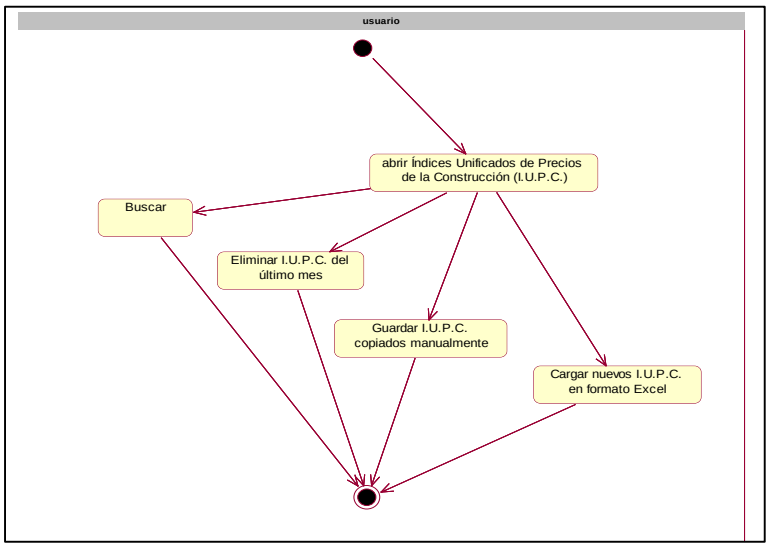
B) Diagrama de Actividades

Diagrama 07: Diagrama de actividades - gestión de usuario.



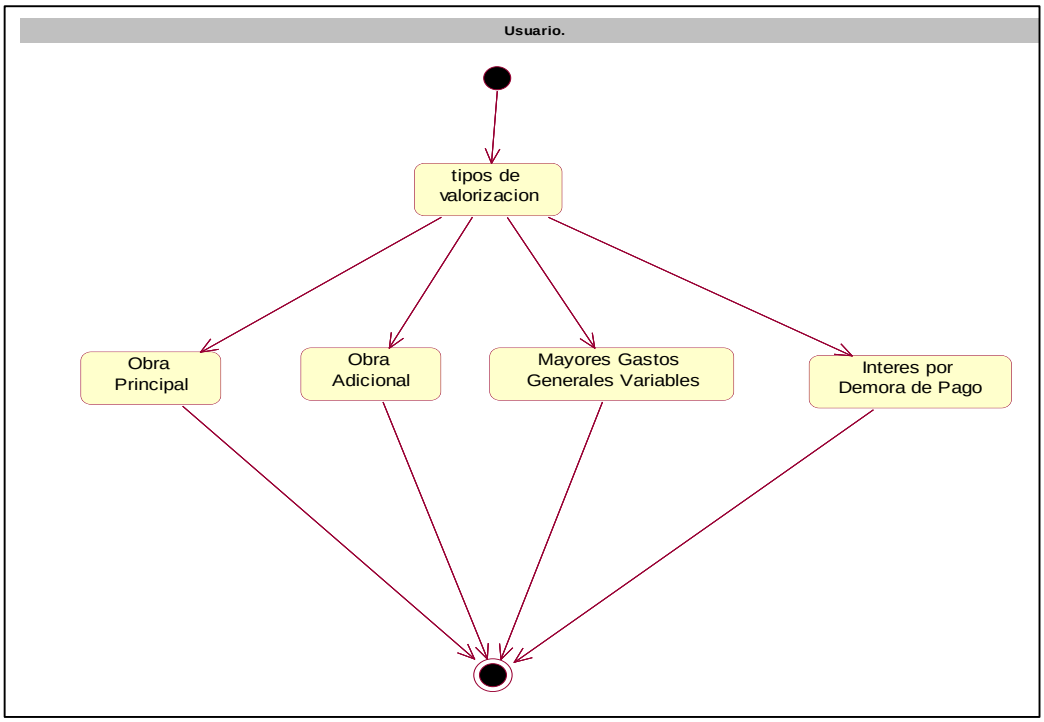
Fuente: Elaboración propia.

Diagrama 08: Diagrama de actividades - gestión de I.U.P.C.



Fuente: Elaboración propia.

Diagrama 09: Diagrama de actividades - gestión de Valorización

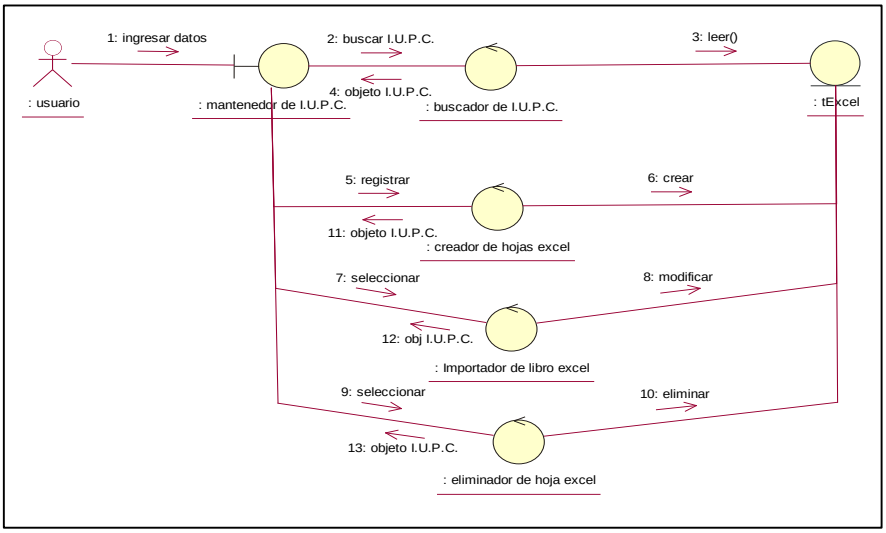


Fuente: Elaboración propia.

4.2.2. Análisis y Diseño

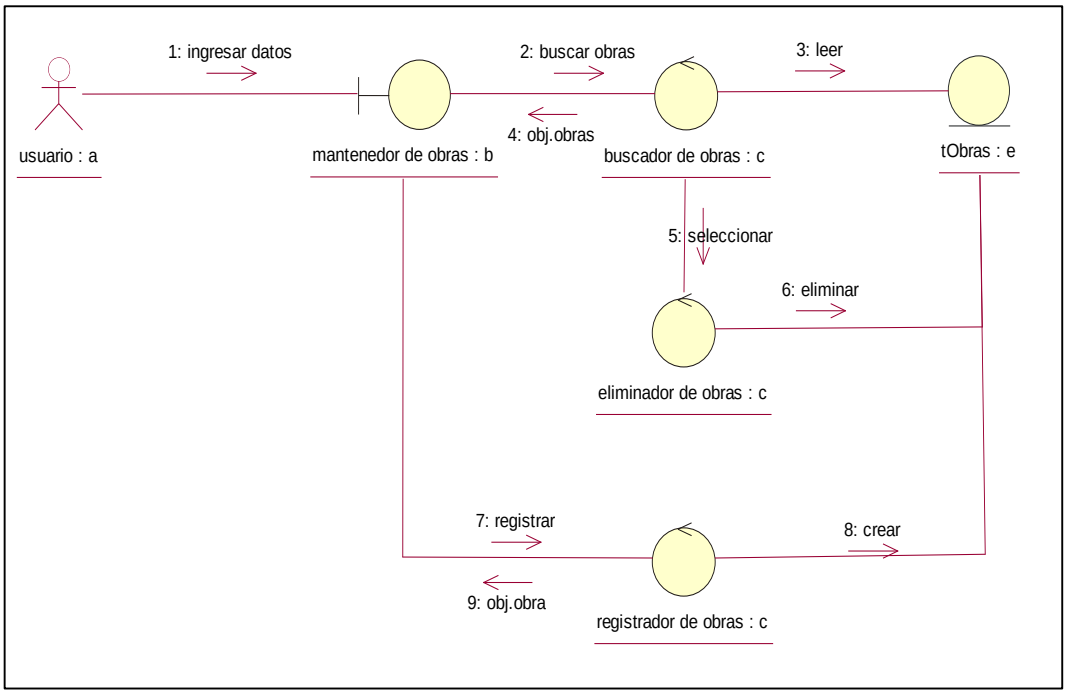
A) Diagrama de Colaboración

Diagrama 10: Diagrama de colaboración - gestión de I.U.P.C.



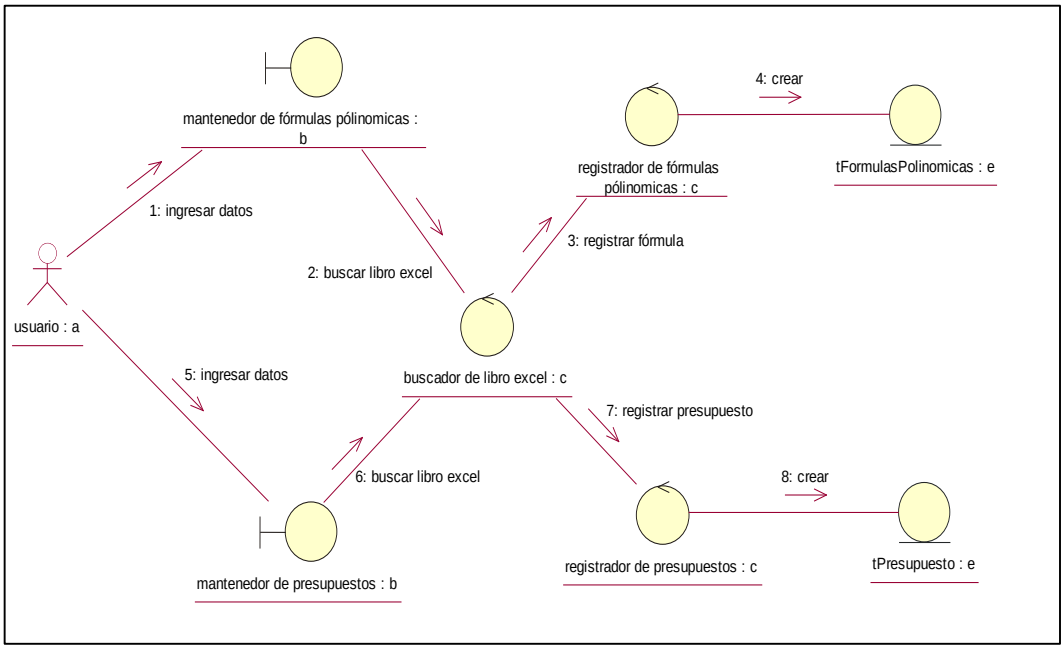
Fuente: Elaboración propia.

Diagrama 11: Diagrama de colaboración - gestión de Valorización (Mantenimiento de Obras).



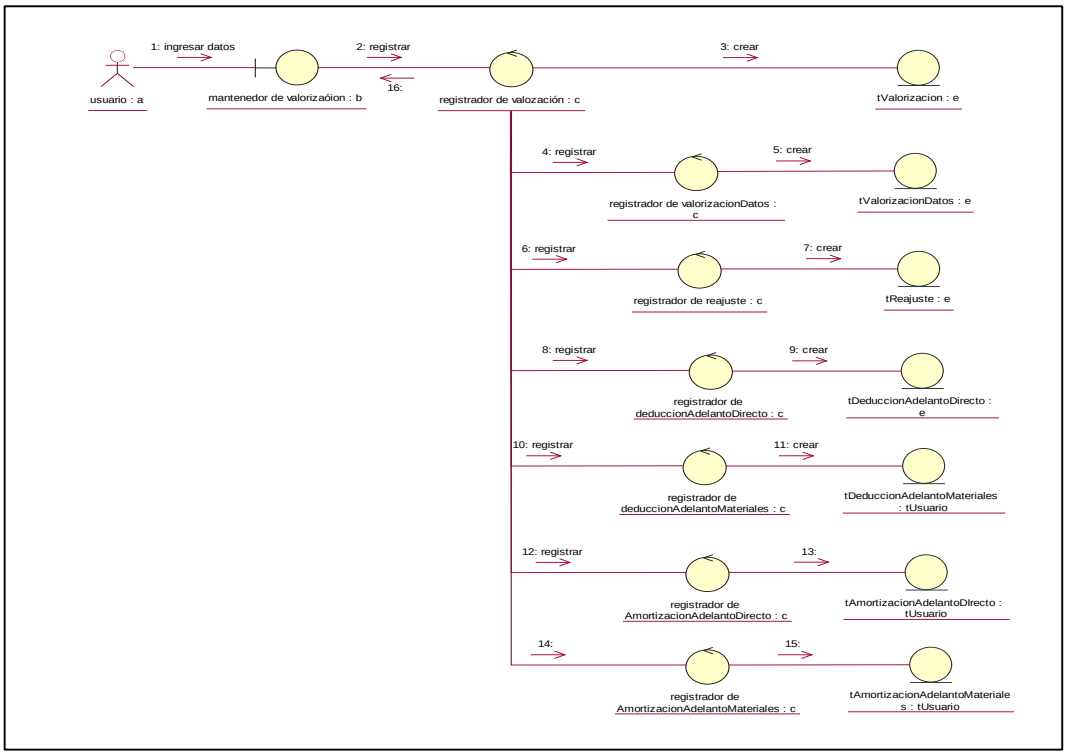
Fuente: Elaboración propia.

Diagrama 12: Diagrama de colaboración - gestión de Valorización (Ingreso de presupuesto y fórmulas polinómicas)



Fuente: Elaboración propia.

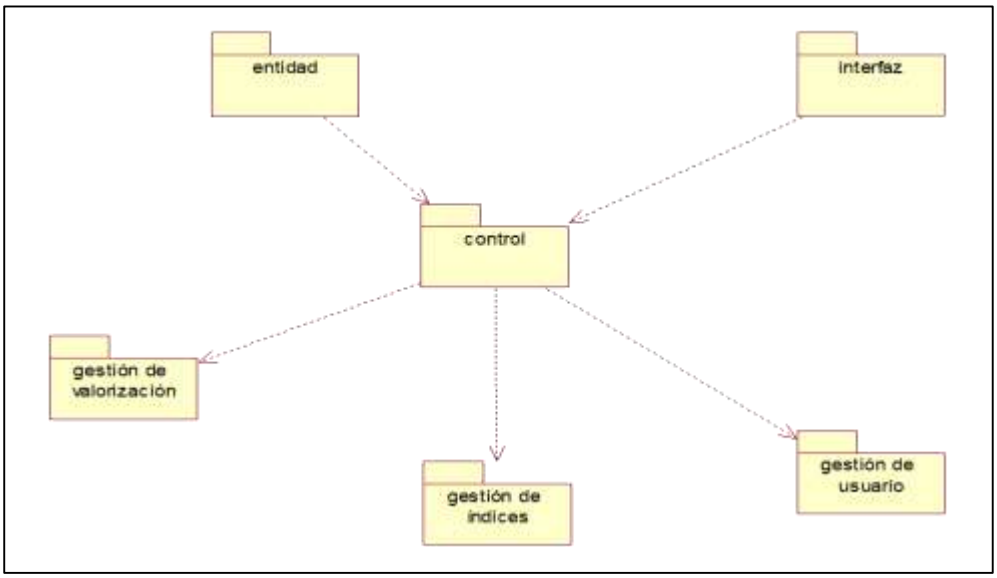
Diagrama 13: Diagrama de colaboración - gestión de Valorización (Creación de una Valorización)



Fuente: Elaboración propia.

B) Diagrama de paquete de análisis

Diagrama 14: Diagrama de Paquete de Análisis.

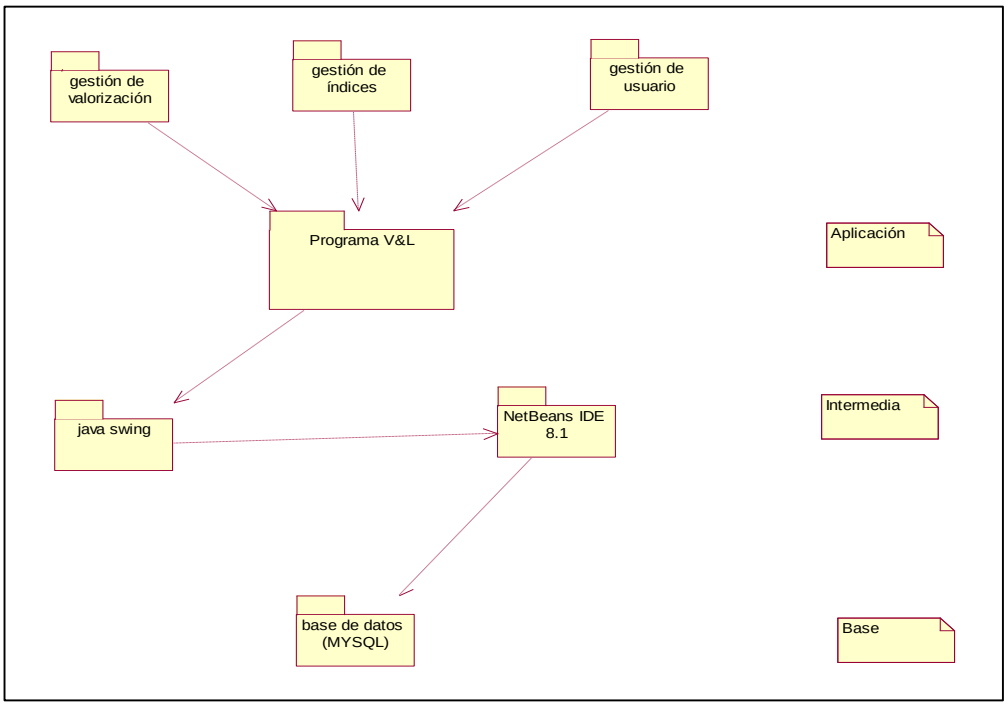


Fuente: Elaboración propia.

C) Diseño de la Arquitectura

C.1. Diagrama de subsistemas por capas

Diagrama 15: Diagrama de subsistemas por capas.



Fuente: Elaboración propia.

D) Diseño de Interfaces y Diagrama de Secuencias

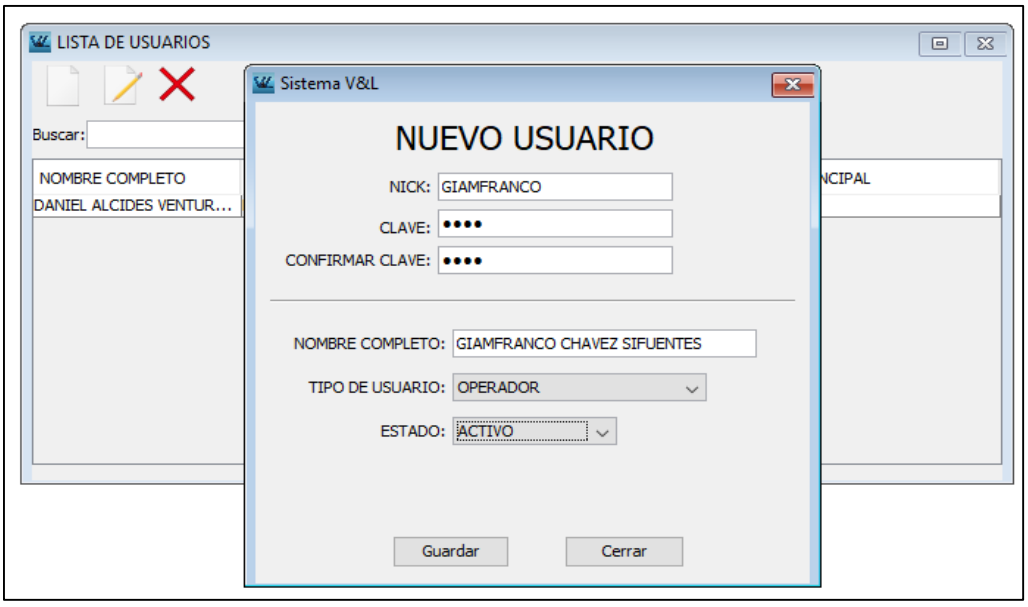
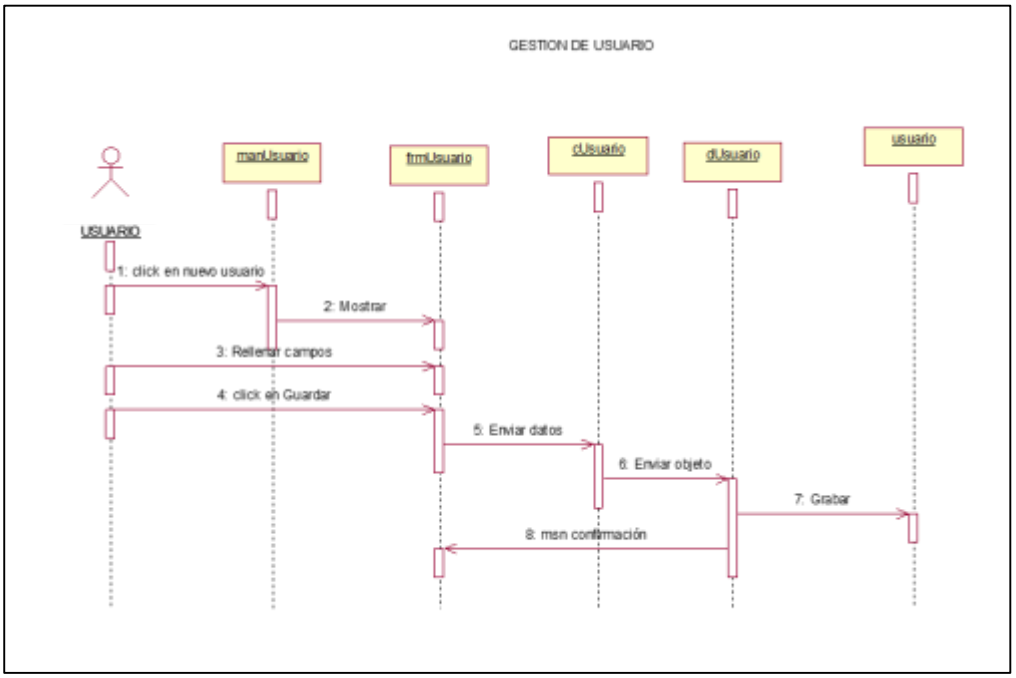


Imagen 4.1: interfaz de la creación de usuario.

Diagrama 16: Diagrama de secuencia – Nuevo Usuario.



Fuente: Elaboración propia.

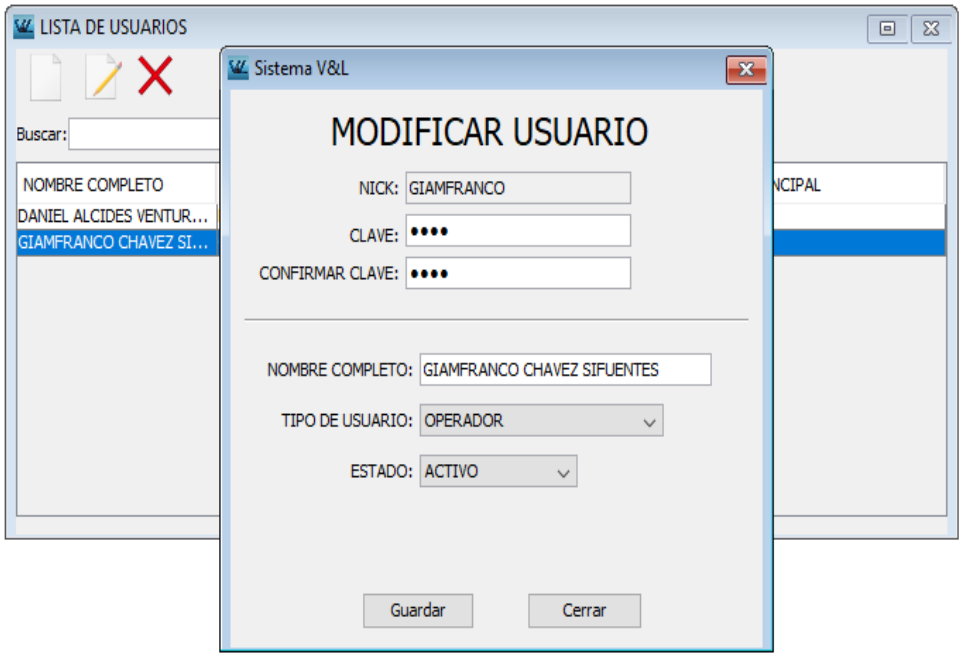
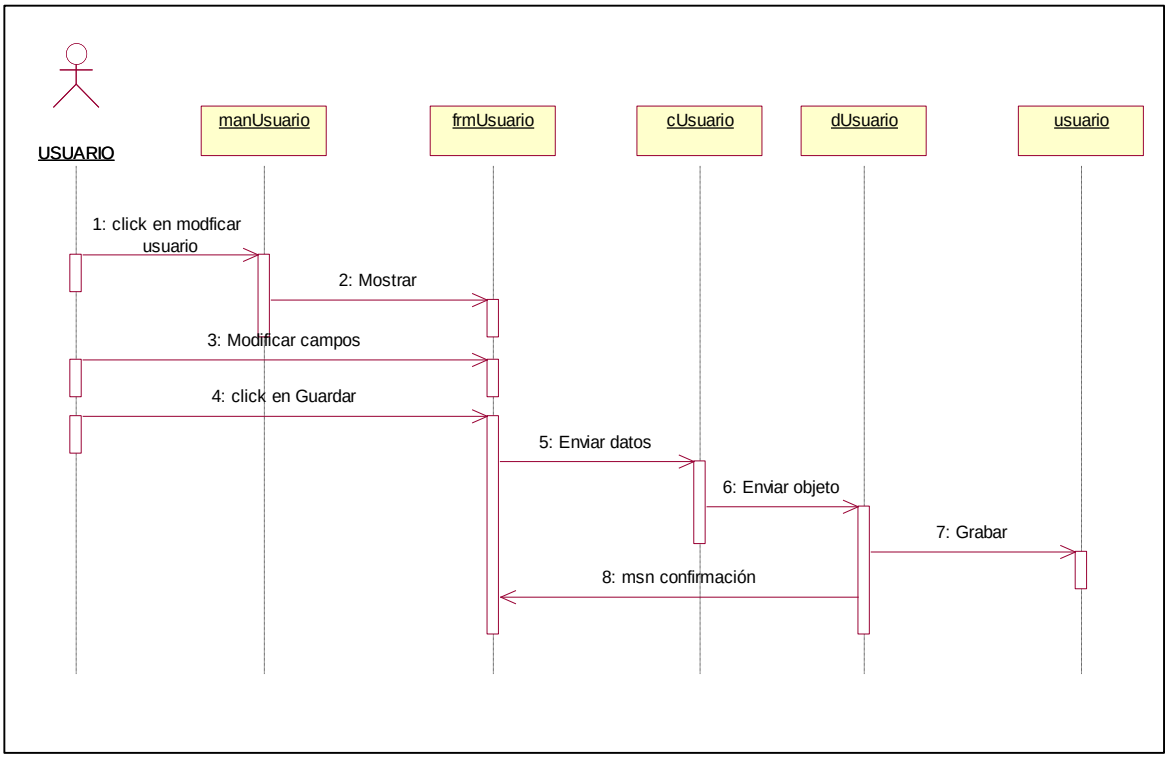


Imagen 4.2: Interfaz de la modificación de usuario.

Diagrama 17: Diagrama de secuencia – Modificar Usuario.



Fuente: Elaboración propia.

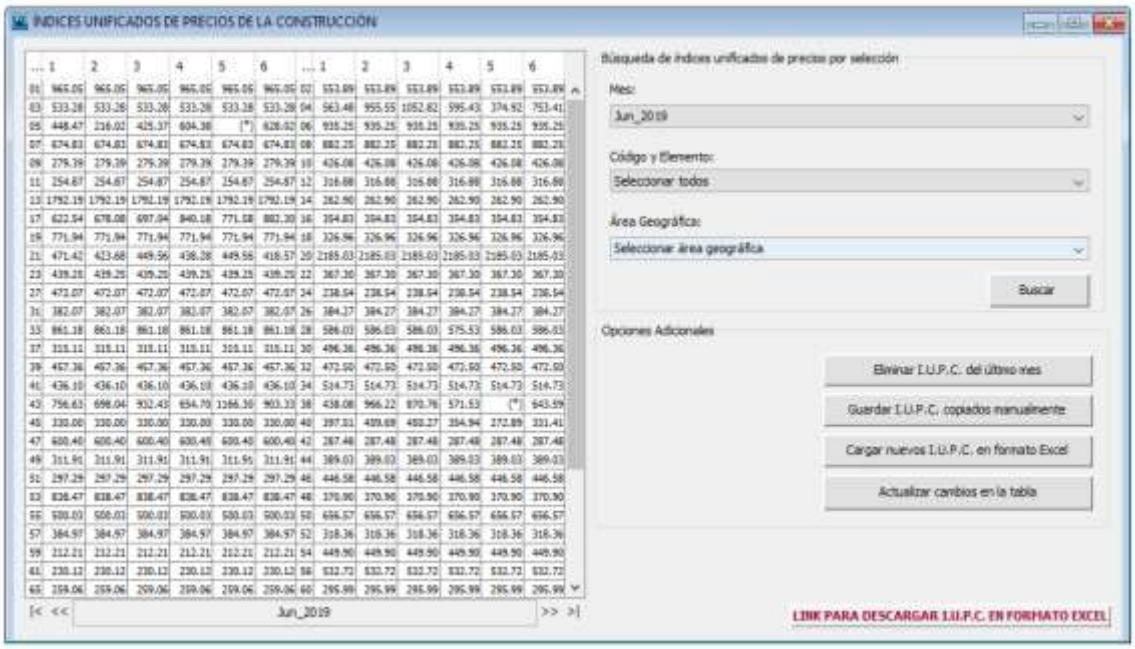
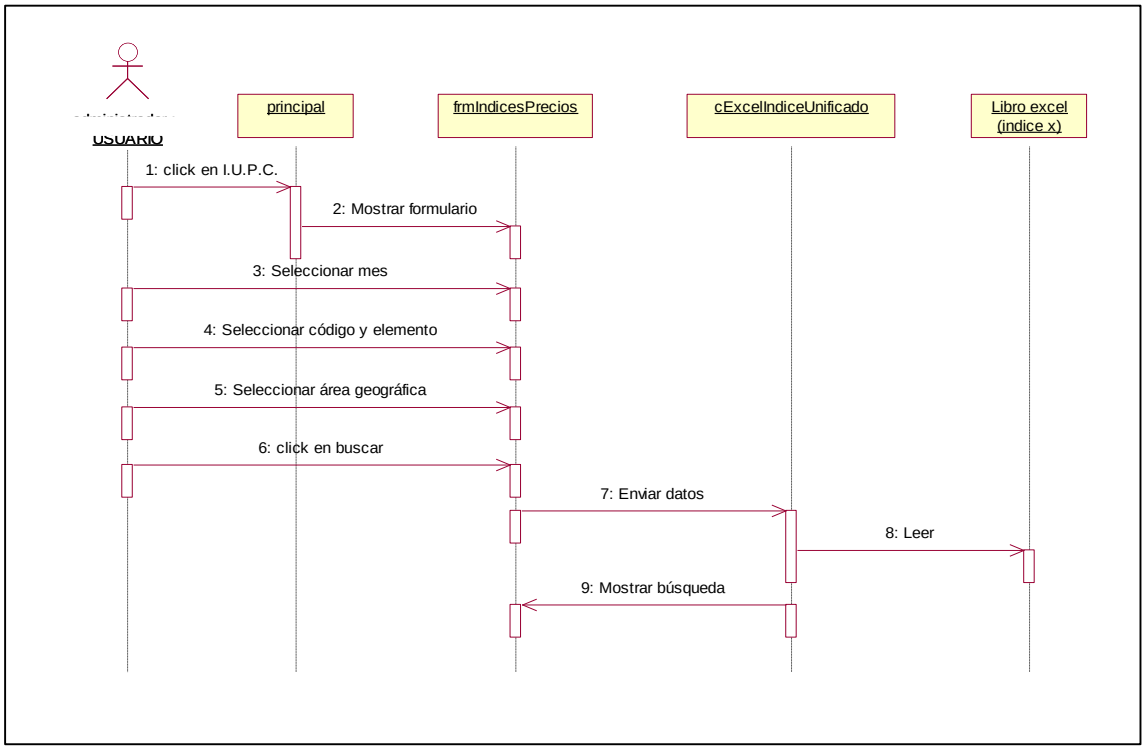


Imagen 4.3: Interfaz para la búsqueda de I.U.P.C.

Diagrama 18: Diagrama de secuencia – Buscar I.U.P.C.



Fuente: Elaboración propia.

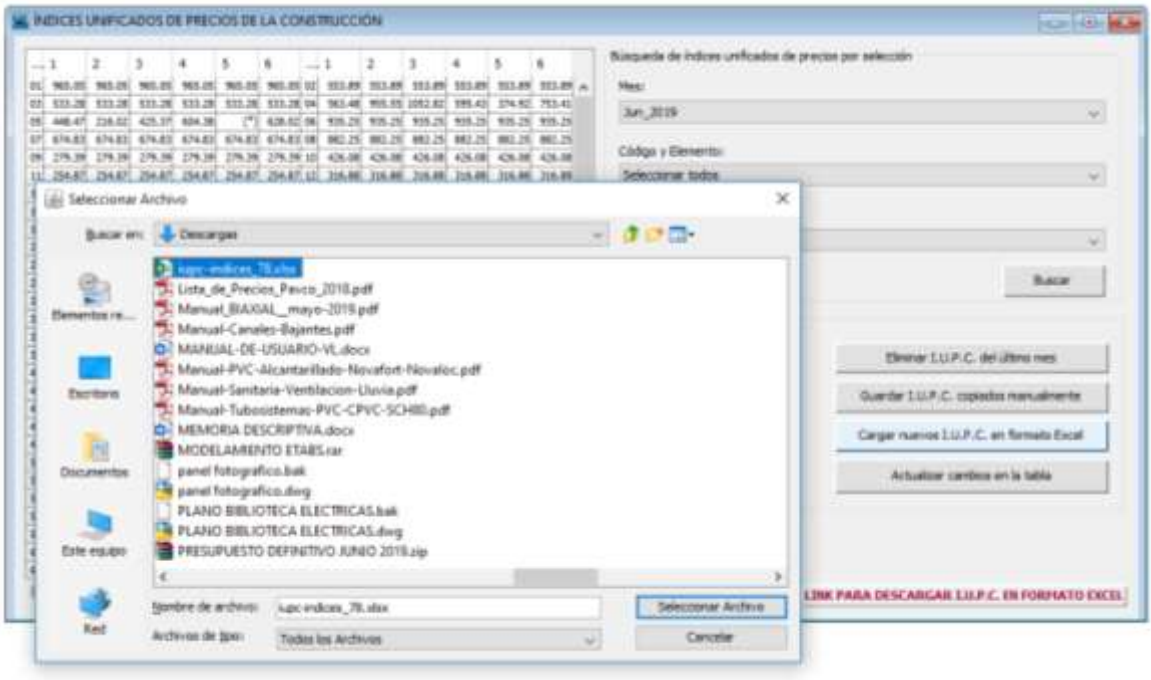
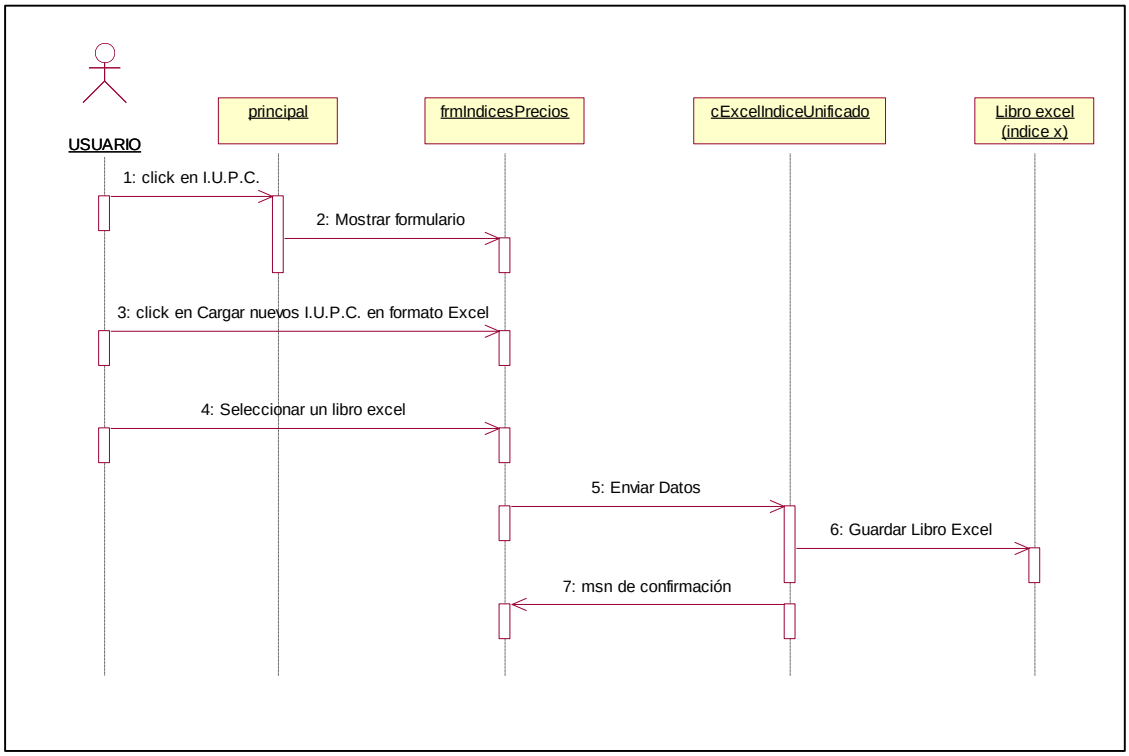


Imagen 4.4: Interfaz para cargar nuevos I.U.P.C. en formato Excel.

Diagrama 19: Diagrama de secuencia – Cargar nuevos I.U.P.C.



Fuente: Elaboración propia.

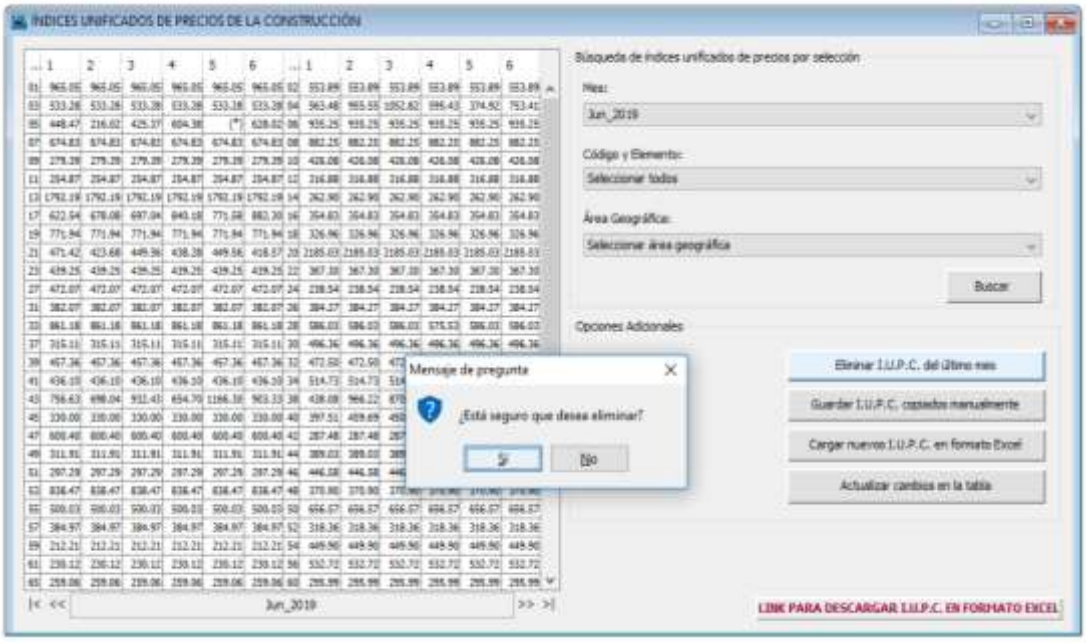
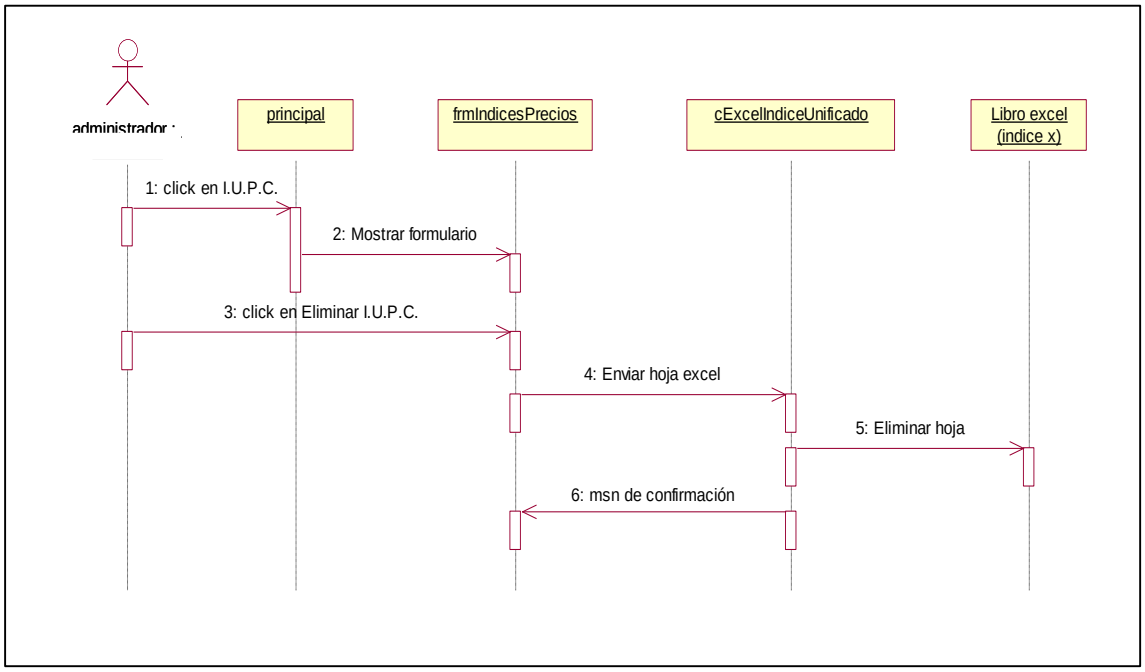


Imagen 4.5: Interfaz para eliminar I.U.P.C. del último mes.

Diagrama 20: Diagrama de secuencia – Eliminar I.U.P.C. del último mes.



Fuente: Elaboración propia.

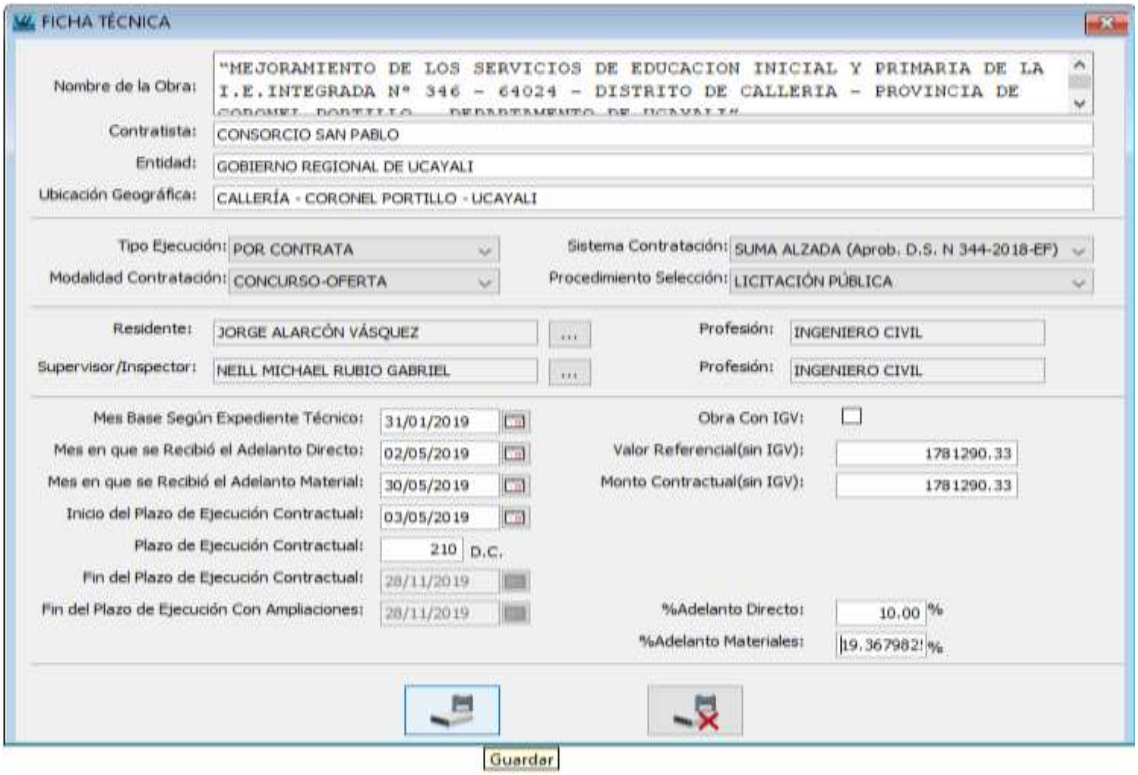
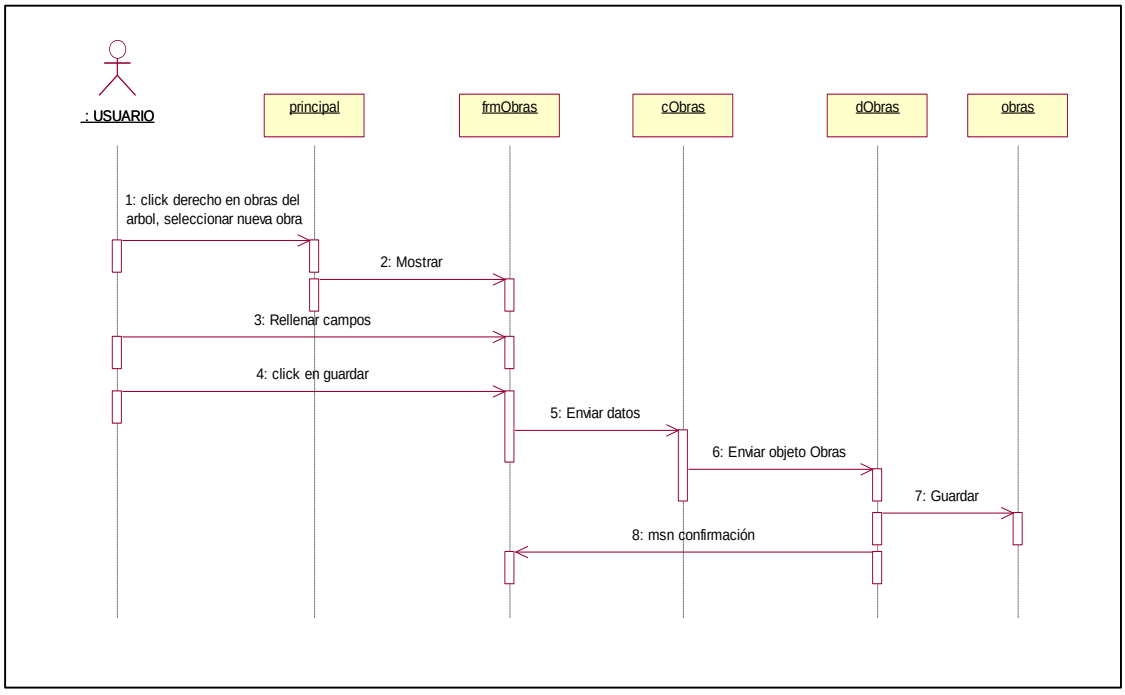


Imagen 4.6: Interfaz para nueva obra.

Diagrama 21: Diagrama de secuencia – Nueva Obra.



Fuente: Elaboración propia.

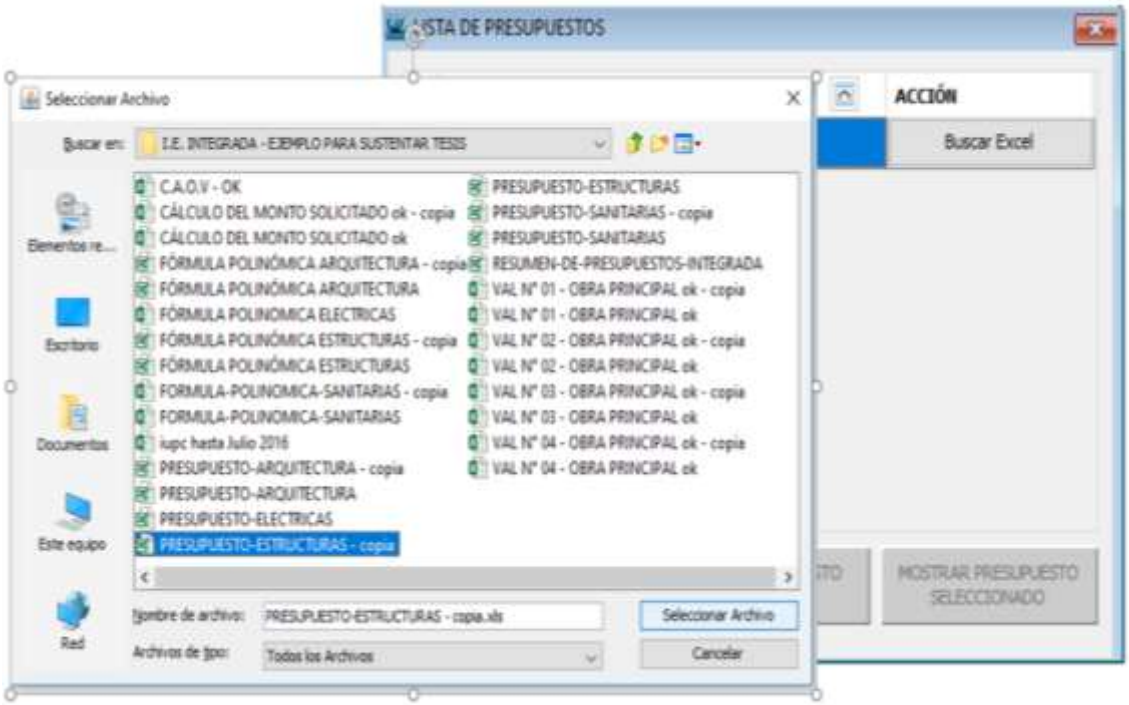
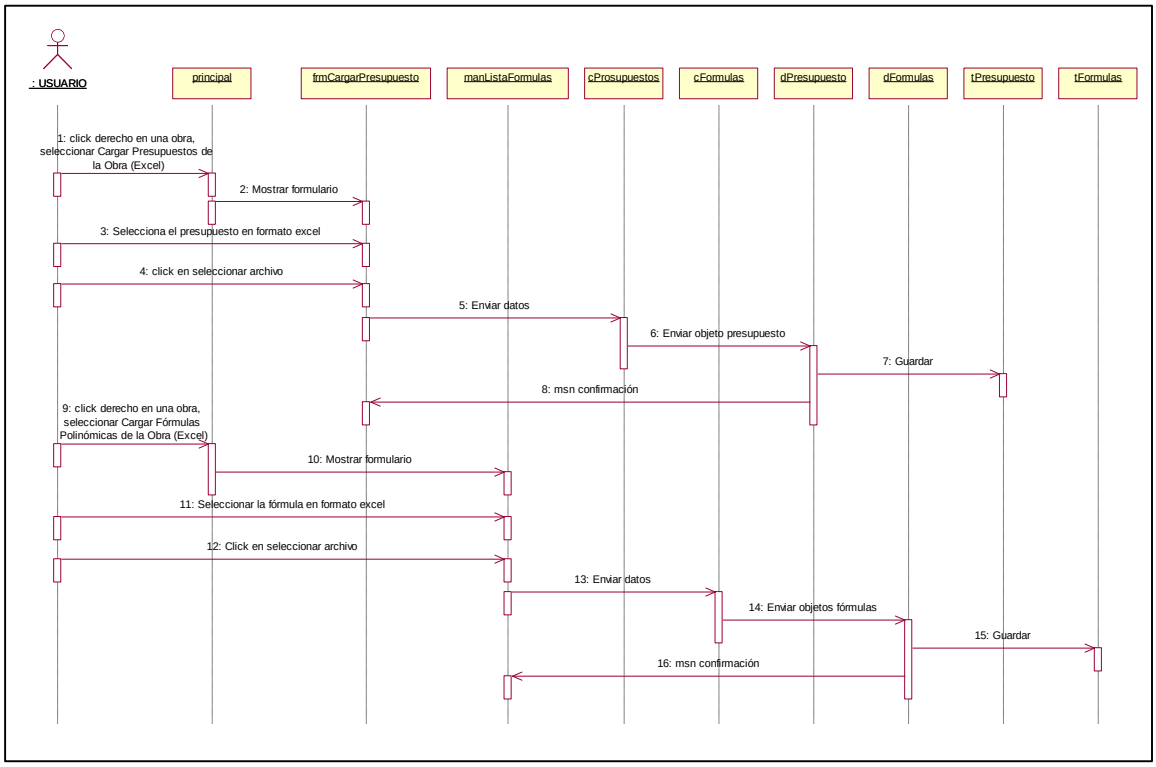


Imagen 4.7: Interfaz para importar presupuestos y fórmulas polinómicas.

Diagrama 22: Diagrama de secuencia – importar presupuestos y fórmulas polinómicas.



Fuente: Elaboración propia.

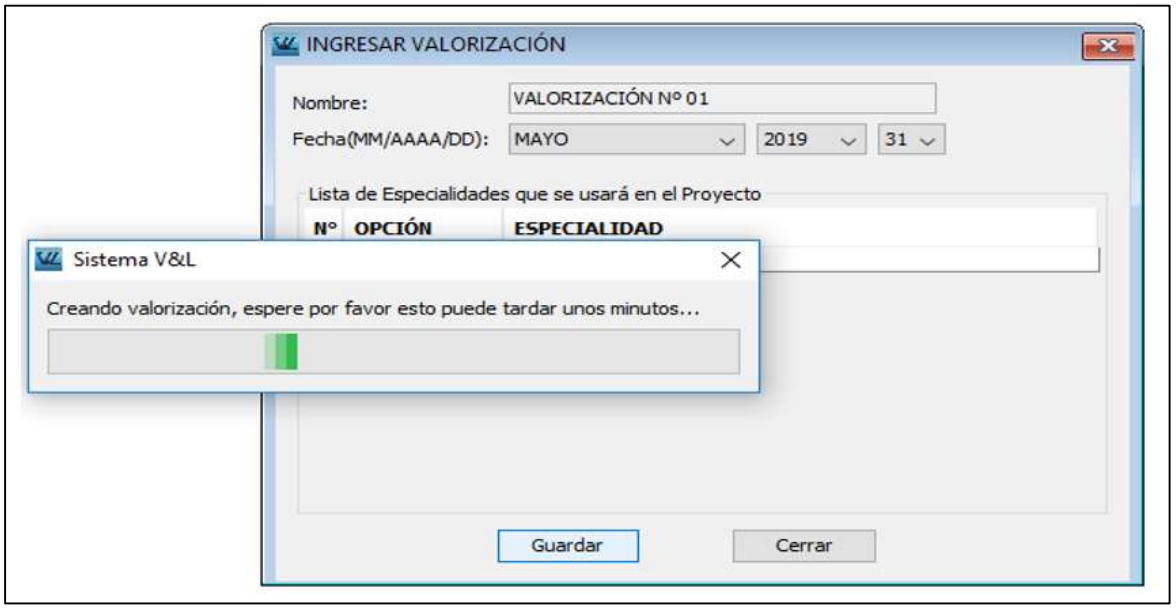
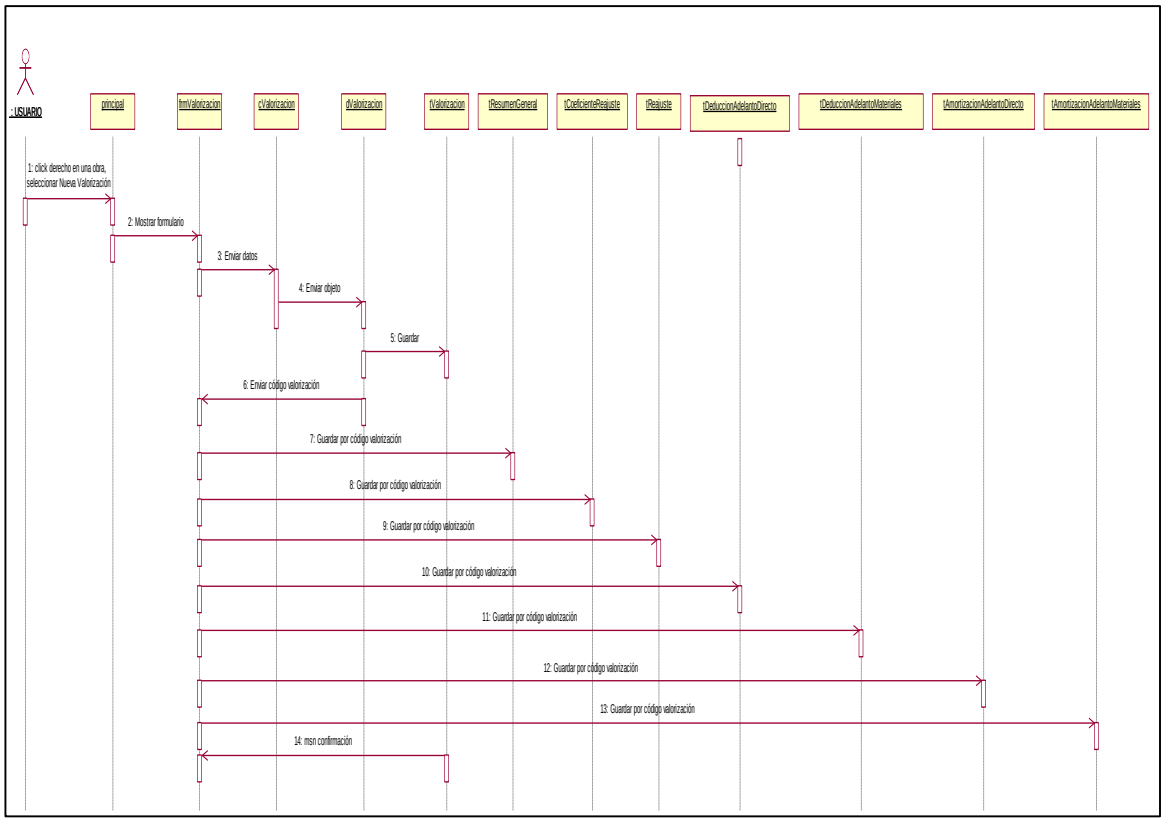


Imagen 4.8: Interfaz para crear una valorización.

Diagrama 23: Diagrama de secuencia – Nueva Valorización.



Fuente: Elaboración propia.

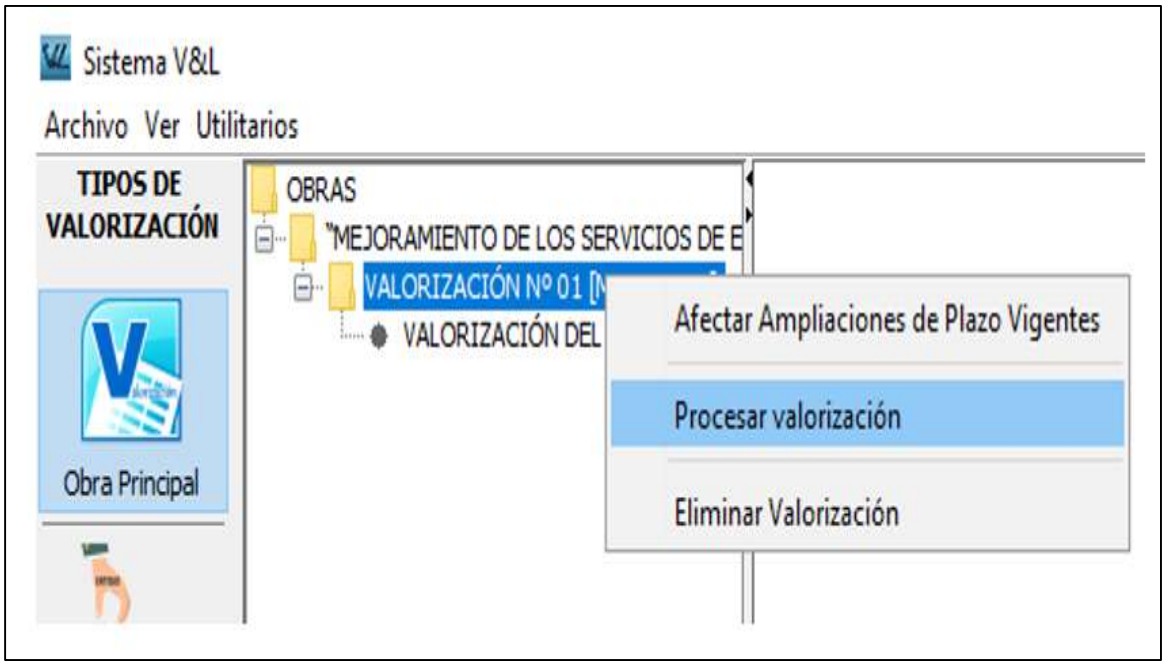
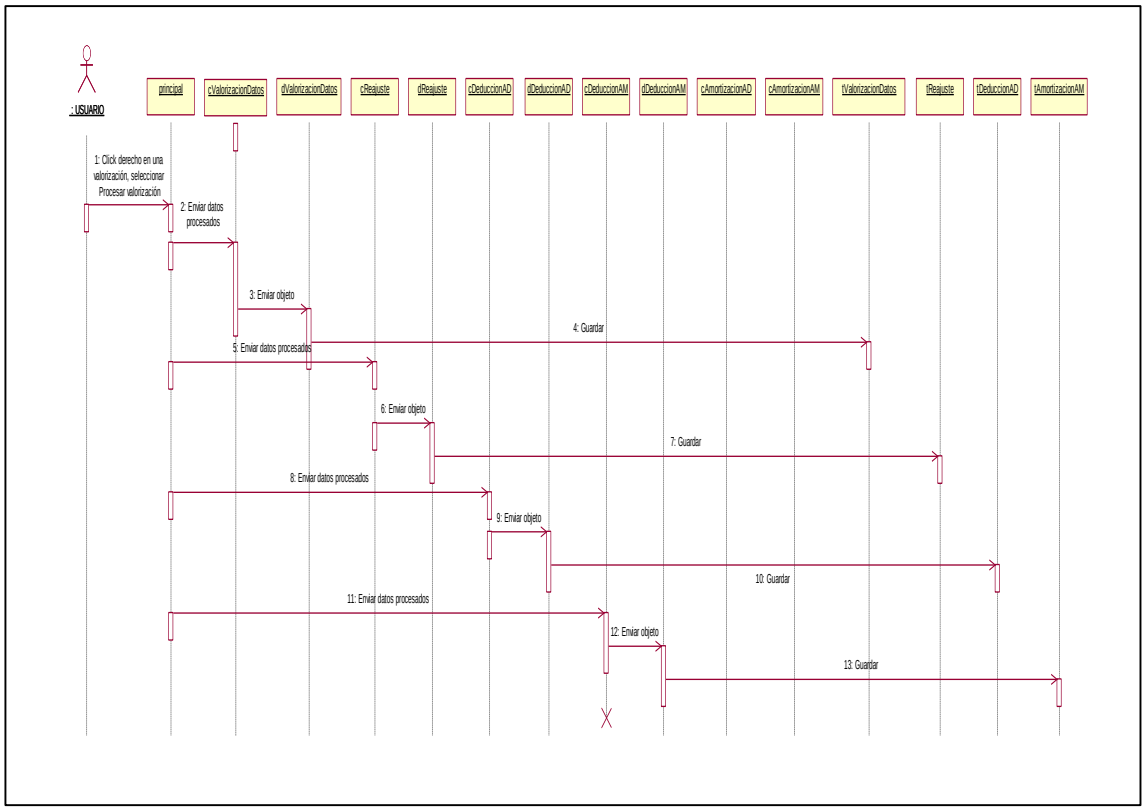


Imagen 4.9: Interfaz para procesar una valorización.

Diagrama 24: Diagrama de secuencia – Procesar Valorización.



Fuente: Elaboración propia.

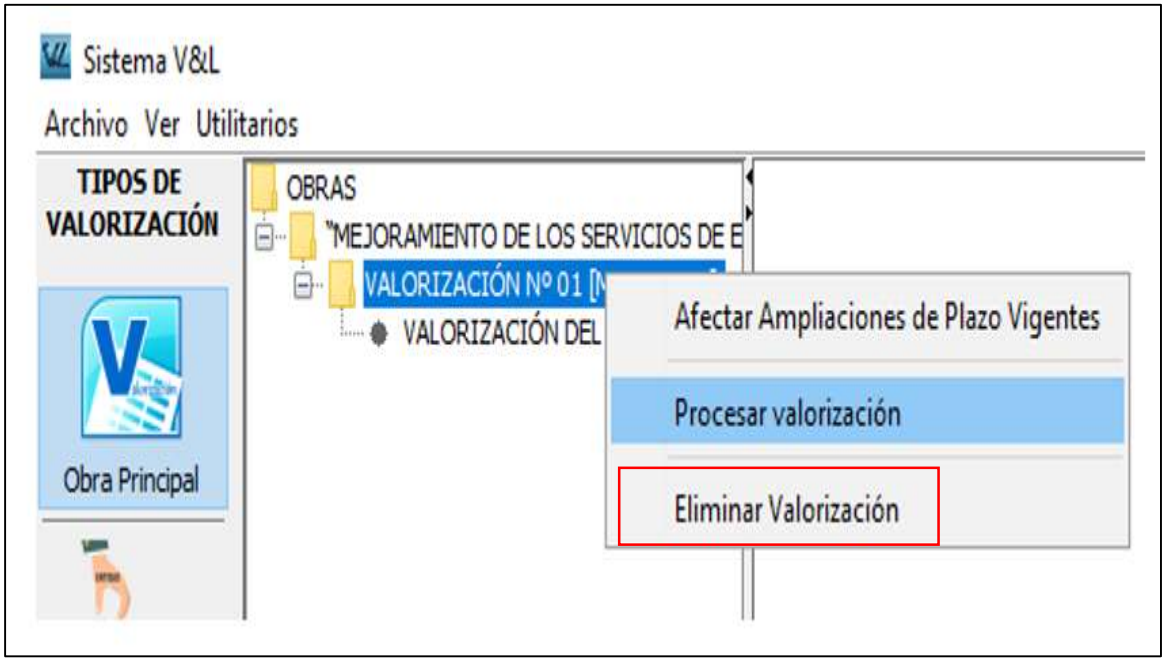
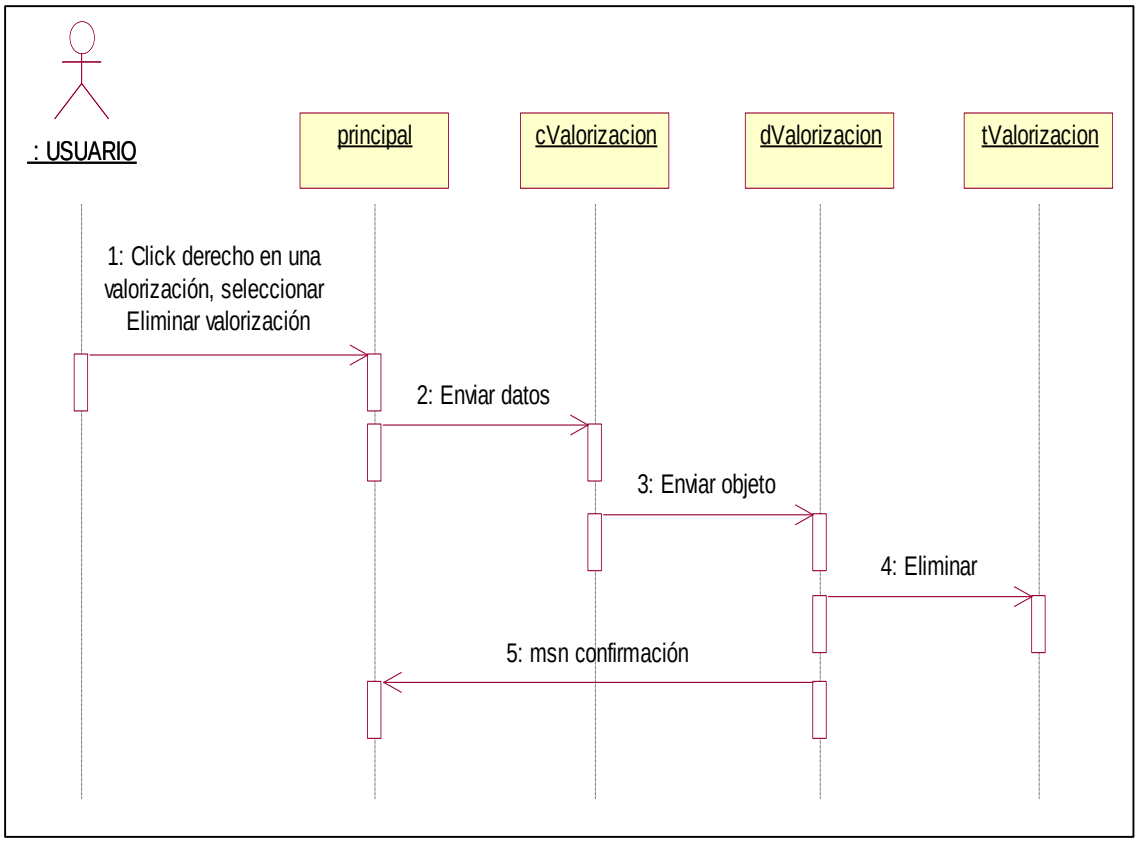


Imagen 4.10: Interfaz para eliminar una valorización.

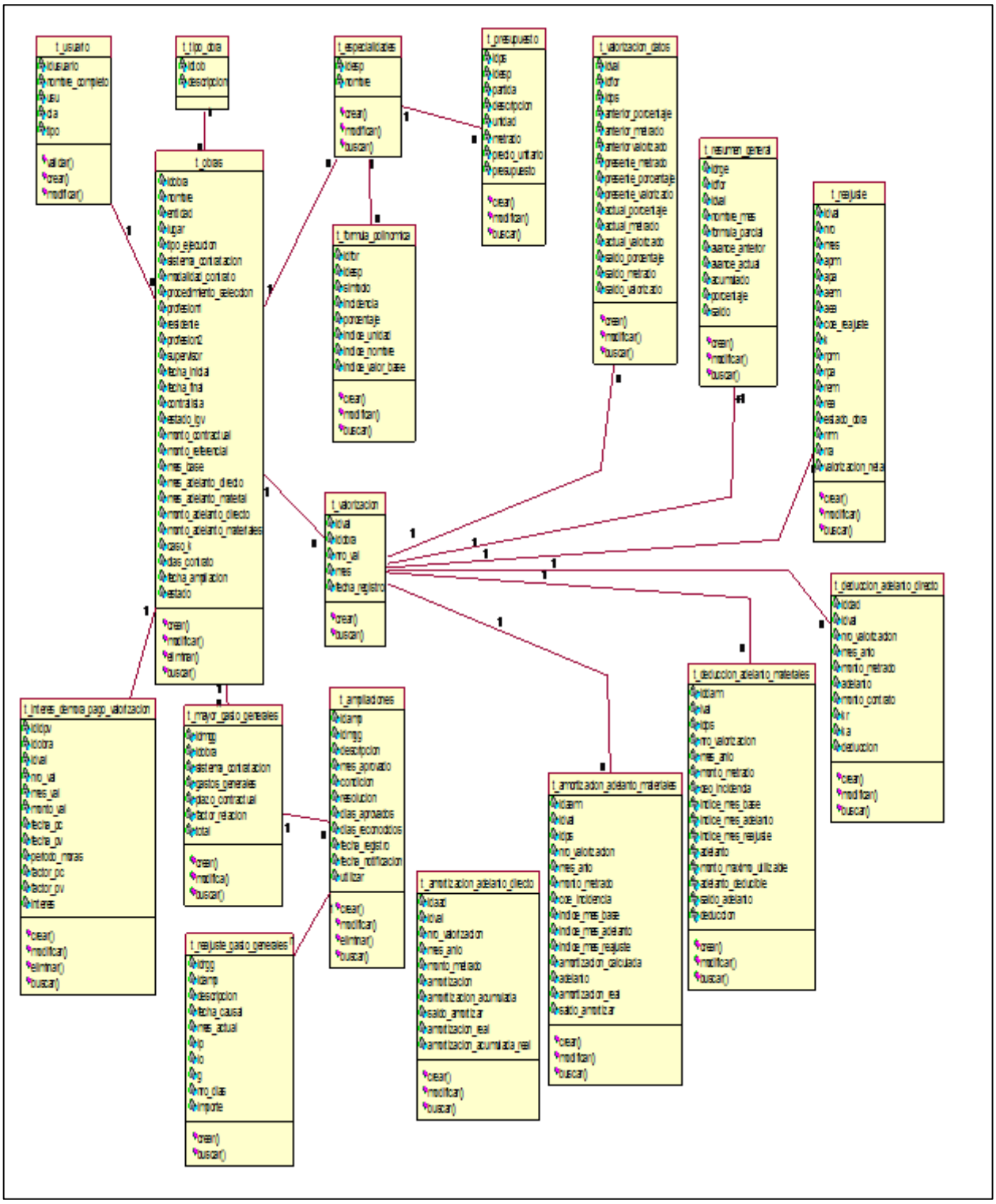
Diagrama 25: Diagrama de secuencia – Eliminar Valorización.



Fuente: Elaboración propia.

E) Diagrama de Clases

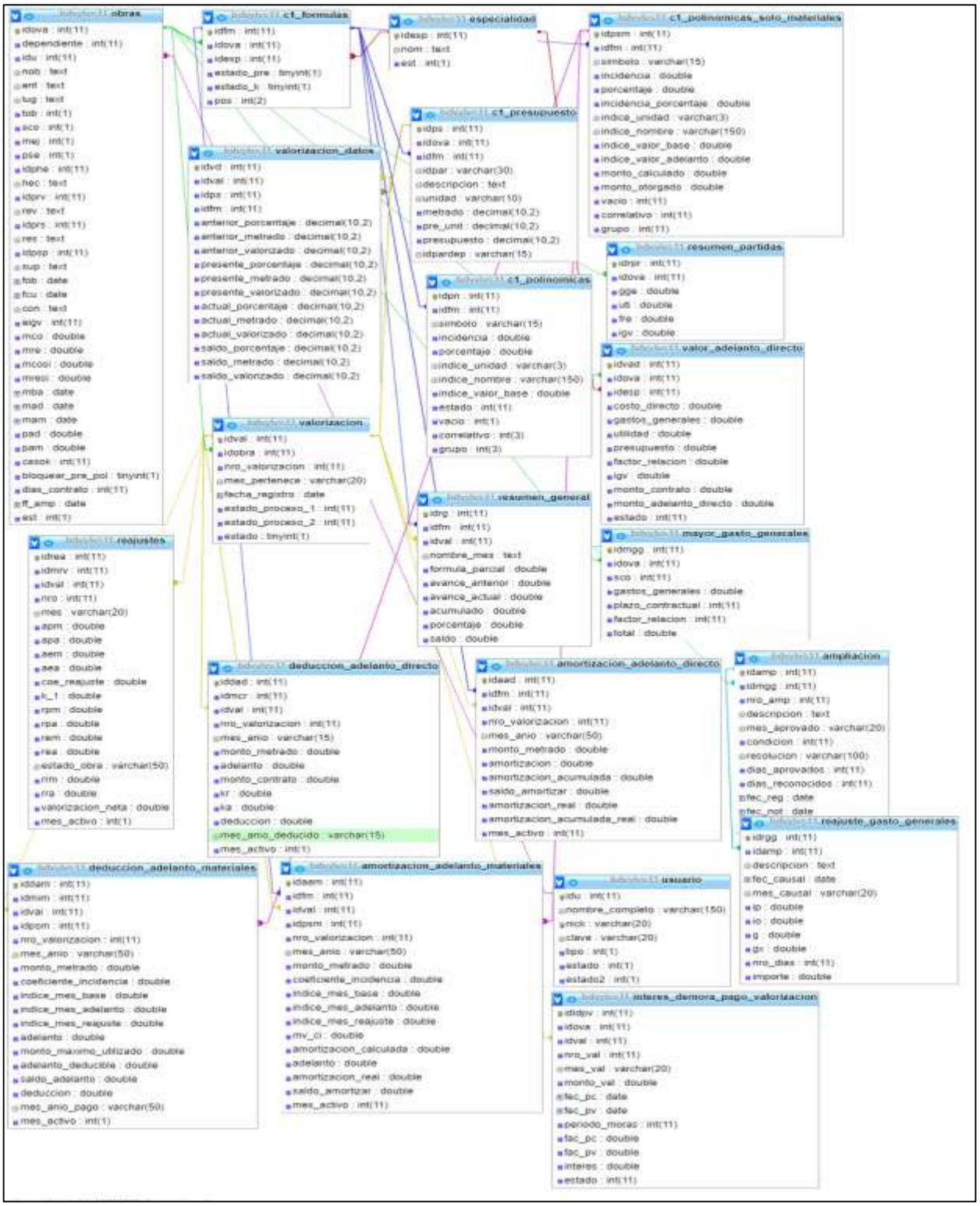
Diagrama 26: Diagrama de Clases.



Fuente: Elaboración propia.

F) Diagrama de Base de Datos

Diagrama 27: Diagrama de Base de Datos.



Fuente: Elaboración propia.

G) Diccionario de Base de Datos

Tabla 01: Especificaciones de la Tabla obras.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Obras**		
DEFINICIÓN	**TABLA PARA ALMACENAR LAS OBRAS DEL SISTEMA**		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idova	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
dependiente	Int	11	Clave foránea de la tabla obras.
idu	Int	11	Clave foránea de la tabla usuario.
nob	text		Nombre de la obra.
ent	Text		Entidad de la obra.
lug	Text		Lugar donde se ejecuta la obra.
tob	int	1	Tipo de obra.
sco	Int	1	Sistema de contratación.
mej	Int	1	Modalidad de ejecución.
pse	Int	1	Procedimiento de selección.
idprs	Int	11	Clave foránea de persona.
res	Text		Nombre completo del residente.
idpsp	Int	11	Clave foránea de persona.
sup	Text		Nombre completo del supervisor.
fob	Date		Inicio de plazo de ejecución contractual.
fcu	Date		Fin de plazo de ejecución contractual.
con	Text		Contratista de la obra.
eigv	Int	11	Estado del IGV en la obra.
mco	Doublé		Monto contractual de la obra sin IGV.
mre	Doublé		Monto del valor referencial de la obra sin IGV.
mcosi	Doublé		Monto contractual de la obra con IGV.
mresi	Doublé		Monto del valor referencial de la obra con IGV.

mba	Date		Mes base según expediente técnico de la obra.
mad	Date		Mes en que se realizó el adelanto directo.
mam	Date		Mes en que se realizó el adelanto para materiales.
pad	Doublé		Porcentaje de adelanto directo.
pam	Doublé		Porcentaje de adelanto para materiales.
casok	Int	11	Indica el uso del coeficiente de reajuste Kr.
bloquear_pre_pol	Int	1	Indica si se ingresó fórmulas polinómicas y presupuestos de la obra.
dias_contrato	Int	11	Almacena el total de días del plazo contractual de la obra.
ff_amp	Date		Indica la fecha de la ampliación de plazo.
est	int	1	Indica el estado de la obra.

Tabla 02: Especificaciones de la Tabla C1_fórmulas.

ESPECIFICACIONES DE LA TABLA			
TABLA	**C1_fórmulas**		
DEFINICIÓN	**TABLA PARA ALMACENAR LAS FÓRMULAS POLINÓMICAS**		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
Idfm	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
Idova	Int	11	Clave foránea de la tabla obra.
Idesp	Int	11	Clave foránea de la tabla especialidad.
Estado_pre	Int	1	Indica si se ingresaron los presupuestos de la obra.
Estado_k	Int	1	Indica si se ingresaron las fórmulas polinómicas de la obra.
pos	int	2	Indica orden en la visualización.

Tabla 03: Especificaciones de la Tabla especialidades.

ESPECIFICACIONES DE LA TABLA			
TABLA	**especialidades**		
DEFINICIÓN	TABLA PARA ALMACENAR LAS ESPECIALIDADES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
Idesp	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
Nom	Text		Nombre de la especialidad.
est	int	1	Estado de la especialidad.

Tabla 04: Especificaciones de la Tabla C1_polinómicas_solo_materiales.

ESPECIFICACIONES DE LA TABLA			
TABLA	**C1_polinómicas_solo_materiales**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS ÍNDICES DE LOS MATERIALES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
Idpsm	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idfm	int	11	Clave foránea de c1_formulas.
Símbolo	Varchar	15	Símbolo de la tabla.
Incidencia	double		Incidencia de la tabla.
Porcentaje	Doublé		Porcentaje de la tabla.
Incidencia_porcentaje	Doublé		Porcentaje de la incidencia de la tabla.
Índice_unidad	Varchar	3	Unidad del índice.
Índice_nombre	Varchar	150	Nombre del índice.
Índice_valor_base	Doublé		Valor base del índice.
Índice_valor_adelanto	Doublé		Valor adelanto del índice.
Monto_calculado	Doublé		Monto calculado.
Monto_otorgado	Doublé		Monto otorgado.
Vacio	Int	11	Indica si se ingresó datos.
Correlativo	Int	11	Indica posición en la tabla.
grupo	int	11	Indica el grupo en la tabla.

Tabla 05: Especificaciones de la Tabla C1_presupuestos.

ESPECIFICACIONES DE LA TABLA			
TABLA	**C1_presupuestos**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS PRESUPUESTOS		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
Idps	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
Idova	Int	11	Clave foránea de la obra.
Idfm	Int	11	Clave foránea de la c1_fórmulas.
Idpar	Varchar	30	Identificar las partidas.
Descripción	Text		Descripción de los presupuestos.
Unidad	Varchar	10	Unidad de los presupuestos.
Metrado	Decimal	10,2	Metrado de los presupuestos.
Pre_unit	Decimal	10,2	Precio unitario de los presupuestos.
Presupuesto	Decimal	10,2	Presupuestos de la tabla.

Tabla 06: Especificaciones de la Tabla Valorización_datos.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Valorización_datos**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS METRADOS DE LAS VALORIZACIONES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
Idvd	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
Idval	Int	11	Clave foránea de la tabla valorización.
Idps	Int	11	Clave foránea de la tabla presupuesto.
Idfm	int	11	Clave foránea de la tabla c1_formulas.
anterior_porcentaje	Decimal	10,2	Ingreso de valor decimal.
anterior_metrado	Decimal	10,2	Ingreso de valor decimal.
anterior_valorizado	Decimal	10,2	Ingreso de valor decimal.
presente_porcentaje	decimal	10,2	Ingreso de valor decimal.
presente_metrado	Decimal	10,2	Ingreso de valor decimal.
presente_valorizado	Decimal	10,2	Ingreso de valor decimal.

actual_porcentaje	Decimal	10,2	Ingreso de valor decimal.
actual_metrado	Decimal	10,2	Ingreso de valor decimal.
actual_valorizado	Decimal	10,2	Ingreso de valor decimal
saldo_porcentaje	Decimal	10,2	Ingreso de valor decimal.
saldo_metrado	Decimal	10,2	Ingreso de valor decimal.
saldo_valorizado	decimal	10,2	Ingreso de valor decimal.

Tabla 07: Especificaciones de la Tabla C1_polinómicas.

ESPECIFICACIONES DE LA TABLA			
TABLA	**C1_polinómicas**		
DEFINICIÓN	TABLA PARA ALMACENAR TODOS LOS ÍNDICES DE LA FÓRMULA		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idpn	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idfm	int	11	Clave foránea de c1_fórmulas.
símbolo	Varchar	15	Símbolo de la tabla.
incidencia	Doublé		Incidencia de la tabla.
porcentaje	Doublé		Porcentaje de la tabla.
índice_unidad	Varchar	3	Unidad del índice.
índice_nombre	Varchar	150	Nombre del índice.
índice_valor_base	Doublé		Valor base del índice.
estado	int	11	Estado de la tabla.
correlativo	int	3	Posición en la tabla.
grupo	int	3	Grupo en la tabla.

Tabla 08: Especificaciones de la Tabla Resumen_partidas.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Resumen_partidas**		
DEFINICIÓN	TABLA PARA ALMACENAR EL RESUMEN DE PARTIDAS DE LA OBRA		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idrpr	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idova	int	11	Clave foránea de la tabla obra.
gge	double		Gastos generales de la tabla.
uti	double		Utilidad de la tabla.
fre	double		Factor de relación de la tabla.

igv	double		IGV de la tabla.

Tabla 09: Especificaciones de la Tabla Valor_adelanto_directo.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Valor_adelanto_directo**		
DEFINICIÓN	TABLA PARA ALMACENAR EL VALOR DEL ADELANTO DIRECTO		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idvad	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idova	int	11	Clave foránea de la tabla obra.
idesp	int	11	Clave foránea de la tabla especialidad.
costo_directo	Doublé		Costo directo de la tabla.
gastos_generales	double		Gastos generales de la tabla.
utilidad	Doublé		Utilidad de la tabla.
presupuesto	double		Presupuesto de la tabla.
factor_relacion	Doublé		Factor de relación de la tabla.
igv	Doublé		IGV de la tabla.
monto_contrato	Doublé		Monto contrato de la tabla.
monto_adelanto_directo	double		Monto de adelanto directo de la tabla.
estado	int	11	Estado de la tabla.

Tabla 10: Especificaciones de la Tabla valorización.

ESPECIFICACIONES DE LA TABLA			
TABLA	**valorización**		
DEFINICIÓN	TABLA PARA ALMACENAR LAS VALORIZACIONES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idval	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idobra	int	11	Clave foránea de la tabla obra.
nro_valorización	int	11	Número de la valorización al que pertenece.

mes_pertenece	Varchar	20	Mes al que pertenece la valorización.
fecha_registro	Date		Fecha que se registró.
estado_proceso_1	int	11	Indica estado de ingreso de metrado.
estado_proceso_2	int	11	Indica estado proceso de metrado.
estado	int	1	Estado de la tabla.

Tabla 11: Especificaciones de la Tabla Resumen_general.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Resumen_general**		
DEFINICIÓN	TABLA PARA ALMACENAR EL RESUMEN GENERAL DE LA OBRA		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idrg	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idfm	Int	11	Clave foránea de la tabla c1_formulas.
idval	int	11	Clave foránea de la tabla valorización.
nombre_mes	text		Nombre de mes de la tabla.
fórmula_parcial	Doublé		Fórmula parcial de la tabla.
avance_anterior	Doublé		Avance anterior de la tabla.
avance_actual	Doublé		Avance actual de la tabla.
acumulado	Doublé		Acumulado de la tabla.
porcentaje	double		Porcentaje de la tabla.
saldo	Doublé		Saldo de la tabla.

Tabla 12: Especificaciones de la Tabla Mayor_gasto_generales.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Mayor_gasto_generales**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS MAYORES GASTOS GENERALES VARIABLES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idmgg	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idova	int	11	Clave foránea de la tabla obra.
sco	int	11	Sistema de contratación de la tabla.

gastos_generales	double		Gastos generales de la tabla.
plazo_contractual	int	11	Plazo contractual de la tabla.
factor_relación	int	11	Factor de relación de la tabla.
total	double		Total de la tabla.

Tabla 13: Especificaciones de la Tabla reajustes.

ESPECIFICACIONES DE LA TABLA			
TABLA	**reajustes**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS REAJUSTES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCION**
idrea	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idmrv	int	11	Clave foránea de la tabla mvr.
idval	Int	11	Clave foránea de la tabla valorización.
nro	Int	11	Número de la valorización.
mes	varchar	20	Mes de la tabla.
apm	Doublé		Avance programado mensual de la tabla.
apa	Doublé		Avance programado acumulado de la tabla.
aem	double		Avance ejecutado mensual de la tabla.
aea	double		Avance ejecutado acumulado de la tabla.
coe_reajuste	Doublé		Coeficiente de reajuste de la tabla.
k_1	Doublé		Kr-1 de la tabla.
rpm	Doublé		Reajuste programado mensual de la tabla.
rpa	Doublé		Reajuste programado acumulado de la tabla.
rem	Doublé		Reajuste ejecutado mensual de la tabla.
rea	double		Reajuste ejecutado acumulado de la tabla.
estado_obra	varchar	50	Estado de la obra.
rrm	Doublé		Reajuste reconocido mensual de la tabla.
rra	Doublé		Reajuste reconocido acumulado de la tabla.

valorización_neta	Doublé		Valorización neta de la obra.
mes_activo	Int	1	Indica si el mes está activo.

Tabla 14: Especificaciones de la Tabla Deducción_adelanto_directo.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Deducción_adelanto_directo**		
DEFINICIÓN	TABLA PARA ALMACENAR LA DEDUCCIÓN DE ADELANTO DIRECTO		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
iddad	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idmcr	Int	11	Clave foránea de la tabla mcr.
idval	Int	11	Clave foránea de la tabla valorización.
nro_valorización	int	11	Indica el número de la valorización.
mes_anio	varchar	15	Indica el mes del año.
monto_metrado	Doublé		Monto del metrado de la tabla.
adelanto	Doublé		Adelanto de la tabla.
monto_contrato	Doublé		Monto contrato de la tabla.
kr	Doublé		Kr de la tabla.
ka	Doublé		Ka de la tabla.
deducción	Doublé		Deducción de la tabla.
mes_anio_deducido	varchar	15	Mes año deducido.
mes_activo	int	1	Mes activo.

Tabla 15: Especificaciones de la Tabla Amortización_adelanto_directo.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Amortización_adelanto_directo**		
DEFINICIÓN	TABLA PARA ALMACENAR LA AMORTIZACIÓN DEL ADELANTO DIRECTO		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idaad	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idfm	Int	11	Clave foránea de la tabla c1_formulas.
idval	Int	11	Clave foránea de la tabla valorización.

nro_valorización	int	11	Número de la valorización de la tabla.
mes_anio	varchar	50	Mes año de la tabla.
monto_metrado	Doublé		Monto metrado de la tabla.
amortización	Doublé		Amortización de la tabla.
amortización_acumulada	Doublé		Amortización acumulada de la tabla.
saldo_amortizar	Doublé		Saldo amortizado de la tabla.
amortización_real	Doublé		Amortización real de la tabla.
amortización_acumulada_real	double		Amortización acumulada real de la tabla.
mes_activo	int	11	Indica si es mes activo en la tabla.

Tabla 16: Especificaciones de la Tabla ampliación.

ESPECIFICACIONES DE LA TABLA			
TABLA	**ampliación**		
DEFINICIÓN	TABLA PARA ALMACENAR LAS AMPLIACIONES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idamp	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idmgg	int	11	Clave foránea de la tabla mgg.
nro_amp	int	11	Numero de ampliación.
descripcion	text		Descripción de la tabla.
mes_aprobado	varchar	20	Mes aprobado de la tabla.
condición	int	11	Condición de la tabla.
resolucion	varchar	100	Resolución de la tabla.
dias_aprobados	int	11	Días aprobados de la tabla.
días_reconocidos	int	11	Días reconocidos de la tabla.
fec_reg	date		Fecha de registro de la tabla.
fec_not	date		Fecha de notificación de la tabla.
utilizar	int	11	Indica si va utilizar en la tabla.

Tabla 17: Especificaciones de la Tabla Deducción_adelanto_materiales.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Deducción_adelanto_materiales**		
DEFINICIÓN	TABLA PARA ALMACENAR LA DEDUCCIÓN DEL ADELANTO PARA MATERIALES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
iddam	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idmim	Int	11	Clave foránea de la tabla mim.
idval	Int	11	Clave foránea de la tabla valorización.
idpsm	Int	11	Clave foránea de la tabla psm.
nro_valorización	int	11	Número de la valorización de la tabla.
mes_anio	varchar	50	Mes año de la tabla.
monto_metrado	double		Monto metrado de la tabla.
coeficiente_incidencia	double		Coeficiente de incidencia de la tabla.
índice_mes_base	double		Mes base del índice en la tabla.
índice_mes_adelanto	double		Mes adelanto del índice en la tabla.
índice_mes_reajuste	double		Mes reajuste del índice en la tabla.
adelanto	double		Adelanto en la tabla.
monto_máximo_utilizado	double		Monto máximo utilizado en la tabla.
adelanto_deducible	double		Adelanto deducible.
saldo_adelanto	double		Saldo por adelanto de la tabla.
deducción	double		Deducción de la tabla.
mes_anio_pago	varchar	50	Pago de mes año en la tabla.
mes_activo	Int	11	Indica que mes está activo.

Tabla 18: Especificaciones de la Tabla Amortización_adelanto_materiales.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Amortización_adelanto_materiales**		
DEFINICIÓN	TABLA PARA ALMACENAR LA AMORTIZACIÓN DE ADELANTO DE MATERIALES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idaam	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idfm	Int	11	Clave foránea de la tabla c1_fórmula.
idval	Int	11	Clave foránea de la tabla valorización.
idpsm	Int	11	Clave foránea de la tabla psm.
nro_valorización	Int	11	Número de la valorización de la tabla.
mes_anio	varchar	50	Mes año de la tabla.
monto_metrado	Doublé		Monto metrado de la tabla.
coeficiente_incidencia	Doublé		Coeficiente de incidencia de la tabla.
índice_mes_base	double		Mes base del índice de la tabla.
índice_mes_adelanto	Doublé		Mes adelanto del índice de la tabla.
índice_mes_reajuste	Doublé		Mes reajuste del índice de la tabla.
mv_ci	Doublé		Monto valorización de la tabla.
amortización_calculada	Doublé		Amortización calculada de la tabla.
adelanto	Doublé		Adelanto de la tabla.
amortizacióon_real	Doublé		Amortización real de la tabla.
saldo_amortizar	double		Saldo por amortizar de la tabla.
mes_activo	int	11	Indica que mes está activo en la tabla.

Tabla 19: Especificaciones de la Tabla usuario.

ESPECIFICACIONES DE LA TABLA			
TABLA	**usuario**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS USUARIOS		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idu	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
nombre_completo	varchar	150	Nombre completo de la tabla.
nick	varchar	20	Nick de la tabla.
clave	varchar	20	Clave de la tabla.
tipo	int	1	Tipo de usuario de la tabla.
estado	int	1	Estado de activado de la tabla.
estado2	int	1	Estado de usuario principal de la tabla.

Tabla 20: Especificaciones de la Tabla Reajuste_gasto_generales.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Reajuste_gasto_generales**		
DEFINICIÓN	TABLA PARA ALMACENAR LOS REAJUSTES DE GASTO GENERALES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
idrgg	int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idamp	int	11	Clave foránea de la tabla ampliación.
descripción	text		Descripción de la tabla.
fec_causal	date		Fecha de la causal de la tabla.
mes_causal	varchar	20	Mes de la causal de la tabla.
ip	Doublé		Ip de la tabla.
io	Doublé		Io de la tabla.
g	Doublé		G de la tabla.
gx	Doublé		
nro_días	int	11	Número de días de la tabla.
importe	Doublé		Importe de la tabla.

Tabla 21: Especificaciones de la Tabla Interés_demora_pago_valorización.

ESPECIFICACIONES DE LA TABLA			
TABLA	**Interés_demora_pago_valorización**		
DEFINICIÓN	TABLA PARA ALMACENAR INTERES		
DEFINICIÓN DE COLUMNAS			
NOMBRE	**TYPE**	**LEN**	**DESCRIPCIÓN**
ididpv	Int	11	Clave primaria de la tabla, se genera en forma incremental y automática.
idova	Int	11	Clave foránea de la tabla obra.
idval	Int	11	Clave foránea de la tabla valorización.
nro_val	int	11	Número de la valorización de la tabla.
mes_val	varchar	20	Mes de la valorización de la tabla.
monto_val	double		Monto valorización de la tabla.
fec_pc	date		Fecha pc de la tabla.
fec_pv	date		Fecha de pv de la tabla.
periodo_moras	int	11	Periodos de moras de la tabla.
fac_pc	double		Factor de pc de la tabla.
fac_pv	double		Factor de pv de la tabla.
interés	Doublé		Interés de la tabla.
estado	int	11	Estado de la tabla.

4.3. FASE DE CONSTRUCCIÓN

4.3.1. Codigo Fuente del Sistema V&L

A. Clase conexión

```
package vconexion;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class conexion {

    private String URL;
    private String Driver;
    private String User;
    private String Password;
    private String Error;

    public conexion() {
        Driver = "com.mysql.jdbc.Driver";
        URL = "jdbc:mysql://localhost:3306/bdvylvs11";
        User = "root";
        Password = "";
    }

    public Connection getConnection() {
        try {
            Class.forName(Driver).newInstance();
            return DriverManager.getConnection(URL, User, Password);
        }   catch   (ClassNotFoundException   |   InstantiationException   |
IllegalAccessException | SQLException e) {
            System.out.println("error:" + e.getMessage());
            Error = e.getMessage();

        }
        return null;
    }

    public String getDriver() {
        return Driver;
    }

    public void setDriver(String Driver) {
        this.Driver = Driver;
    }

    public String getError() {
        return Error;
    }

    public void setError(String Error) {
        this.Error = Error;
    }
```

```
    public String getURL() {
        return URL;
    }

    public void setURL(String URL) {
        this.URL = URL;
    }
}
```

01. Clase conexión

```
package vconexion;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class conexion {

    private String URL;
    private String Driver;
    private String User;
    private String Password;
    private String Error;

    public conexion() {
        Driver = "com.mysql.jdbc.Driver";
        URL = "jdbc:mysql://localhost:3306/bdvylvs11";
        User = "root";
        Password = "";
    }

    public Connection getConnection() {
        try {
            Class.forName(Driver).newInstance();
            return DriverManager.getConnection(URL, User, Password);
        } catch (ClassNotFoundException | InstantiationException |
IllegalAccessException | SQLException e) {
            System.out.println("error:" + e.getMessage());
            Error = e.getMessage();

        }
        return null;
    }

    public String getDriver() {
        return Driver;
    }

    public void setDriver(String Driver) {
        this.Driver = Driver;
    }

    public String getError() {
        return Error;
    }
```

```
    public void setError(String Error) {
        this.Error = Error;
    }

    public String getURL() {
        return URL;
    }

    public void setURL(String URL) {
        this.URL = URL;
    }
}
```

02. Clase cObras

```
package OBJETOS;

import java.util.Date;

public class cObras {

    private int idova;
    private int dependiente;
    private int idu;
    private String nob;
    private String ent;
    private String lug;
    private int tob;

    private int sco;
    private int mej;
    private int pse;
    private int idphe;
    private String hec;

    private int idprv;
    private String rev;
    private int idprs;
    private String res;
    private int idpsp;

    private String sup;
    private Date fob;
    private Date fcu;
    private String con;

    private int eigv;

    private double mco;
    private double mre;

    private double mcosi;
    private double mresi;

    private Date mba;
    private Date mad;
    private Date mam;
```

```
private double pad;
private double pam;

private int casok;

private boolean bloquearPrePol;
private int dias_contrato;
private Date ff_amp;

private int est;

public cObras() {
}

public cObras(cObras oo) {
    this.idova = oo.getIdova();
    this.dependiente = oo.getDependiente();
    this.idu = oo.getIdu();
    this.nob = oo.getNob();
    this.ent = oo.getEnt();
    this.lug = oo.getLug();
    this.tob = oo.getTob();
    this.sco = oo.getSco();
    this.mej = oo.getMej();
    this.pse = oo.getPse();
    this.idphe = oo.getIdphe();
    this.hec = oo.getHec();
    this.idprv = oo.getIdprv();
    this.rev = oo.getRev();
    this.idprs = oo.getIdprs();
    this.res = oo.getRes();
    this.idpsp = oo.getIdpsp();
    this.sup = oo.getSup();
    this.fob = oo.getFob();
    this.fcu = oo.getFcu();
    this.con = oo.getCon();
    this.eigv = oo.getEigv();
    this.mco = oo.getMco();
    this.mre = oo.getMre();
    this.mcosi = oo.getMcosi();
    this.mresi = oo.getMresi();
    this.mba = oo.getMba();
    this.mad = oo.getMad();
    this.mam = oo.getMam();
    this.pad = oo.getPad();
    this.pam = oo.getPam();
    this.casok = oo.getCasok();
    this.bloquearPrePol = oo.isBloquearPrePol();
    this.dias_contrato = oo.getDiasContrato();
    this.ff_amp = oo.getFfAmp();
    this.est = oo.getEst();
}

/**
 * @return the idobra
 */
```

```
public int getIdova() {
    return idova;
}

/**
 * @param idobra the idobra to set
 */
public void setIdova(int idobra) {
    this.idova = idobra;
}

/**
 * @return the nob
 */
public String getNob() {
    return nob;
}

/**
 * @param nob the nob to set
 */
public void setNob(String nob) {
    this.nob = nob;
}

/**
 * @return the ent
 */
public String getEnt() {
    return ent;
}

/**
 * @param ent the ent to set
 */
public void setEnt(String ent) {
    this.ent = ent;
}

/**
 * @return the lug
 */
public String getLug() {
    return lug;
}

/**
 * @param lug the lug to set
 */
public void setLug(String lug) {
    this.lug = lug;
}

/**
 * @return the tob
 */
public int getTob() {
```

```
        return tob;
    }

    /**
     * @param tob the tob to set
     */
    public void setTob(int tob) {
        this.tob = tob;
    }

    /**
     * @return the sco
     */
    public int getSco() {
        return sco;
    }

    /**
     * @param sco the sco to set
     */
    public void setSco(int sco) {
        this.sco = sco;
    }

    /**
     * @return the mej
     */
    public int getMej() {
        return mej;
    }

    /**
     * @param mej the mej to set
     */
    public void setMej(int mej) {
        this.mej = mej;
    }

    /**
     * @return the pse
     */
    public int getPse() {
        return pse;
    }

    /**
     * @param pse the pse to set
     */
    public void setPse(int pse) {
        this.pse = pse;
    }

    /**
     * @return the idphe
     */
    public int getIdphe() {
        return idphe;
```

```
}

/**
 * @param idphe the idphe to set
 */
public void setIdphe(int idphe) {
    this.idphe = idphe;
}

/**
 * @return the hec
 */
public String getHec() {
    return hec;
}

/**
 * @param hec the hec to set
 */
public void setHec(String hec) {
    this.hec = hec;
}

/**
 * @return the idprv
 */
public int getIdprv() {
    return idprv;
}

/**
 * @param idprv the idprv to set
 */
public void setIdprv(int idprv) {
    this.idprv = idprv;
}

/**
 * @return the rev
 */
public String getRev() {
    return rev;
}

/**
 * @param rev the rev to set
 */
public void setRev(String rev) {
    this.rev = rev;
}

/**
 * @return the idprs
 */
public int getIdprs() {
    return idprs;
}
```

```
/**
 * @param idprs the idprs to set
 */
public void setIdprs(int idprs) {
    this.idprs = idprs;
}

/**
 * @return the res
 */
public String getRes() {
    return res;
}

/**
 * @param res the res to set
 */
public void setRes(String res) {
    this.res = res;
}

/**
 * @return the idpsp
 */
public int getIdpsp() {
    return idpsp;
}

/**
 * @param idpsp the idpsp to set
 */
public void setIdpsp(int idpsp) {
    this.idpsp = idpsp;
}

/**
 * @return the sup
 */
public String getSup() {
    return sup;
}

/**
 * @param sup the sup to set
 */
public void setSup(String sup) {
    this.sup = sup;
}

/**
 * @return the fob
 */
public Date getFob() {
    return fob;
}

/**
```

```
     * @param fob the fob to set
     */
    public void setFob(Date fob) {
        this.fob = fob;
    }

    /**
     * @return the con
     */
    public String getCon() {
        return con;
    }

    /**
     * @param con the con to set
     */
    public void setCon(String con) {
        this.con = con;
    }

    /**
     * @return the est
     */
    public int getEst() {
        return est;
    }

    /**
     * @param est the est to set
     */
    public void setEst(int est) {
        this.est = est;
    }

    @Override
    public String toString() {
        return nob; //To change body of generated methods, choose Tools |
Templates.
    }

    /**
     * @return the mco
     */
    public double getMco() {
        return mco;
    }

    /**
     * @param mco the mco to set
     */
    public void setMco(double mco) {
        this.mco = mco;
    }

    /**
     * @return the mba
     */
```

```
public Date getMba() {
    return mba;
}

/**
 * @param mba the mba to set
 */
public void setMba(Date mba) {
    this.mba = mba;
}

/**
 * @return the mad
 */
public Date getMad() {
    return mad;
}

/**
 * @param mad the mad to set
 */
public void setMad(Date mad) {
    this.mad = mad;
}

/**
 * @return the fcu
 */
public Date getFcu() {
    return fcu;
}

/**
 * @param fcu the fcu to set
 */
public void setFcu(Date fcu) {
    this.fcu = fcu;
}

/**
 * @return the mre
 */
public double getMre() {
    return mre;
}

/**
 * @param mre the mre to set
 */
public void setMre(double mre) {
    this.mre = mre;
}

/**
 * @return the pad
 */
public double getPad() {
```

```
        return pad;
    }

    /**
     * @param pad the pad to set
     */
    public void setPad(double pad) {
        this.pad = pad;
    }

    /**
     * @return the pam
     */
    public double getPam() {
        return pam;
    }

    /**
     * @param pam the pam to set
     */
    public void setPam(double pam) {
        this.pam = pam;
    }

    /**
     * @return the mam
     */
    public Date getMam() {
        return mam;
    }

    /**
     * @param mam the mam to set
     */
    public void setMam(Date mam) {
        this.mam = mam;
    }

    /**
     * @return the mcosi
     */
    public double getMcosi() {
        return mcosi;
    }

    /**
     * @param mcosi the mcosi to set
     */
    public void setMcosi(double mcosi) {
        this.mcosi = mcosi;
    }

    /**
     * @return the mresi
     */
    public double getMresi() {
        return mresi;
```

```
}

/**
 * @param mresi the mresi to set
 */
public void setMresi(double mresi) {
    this.mresi = mresi;
}

/**
 * @return the eigv
 */
public int getEigv() {
    return eigv;
}

/**
 * @param eigv the eigv to set
 */
public void setEigv(int eigv) {
    this.eigv = eigv;
}

/**
 * @return the casok
 */
public int getCasok() {
    return casok;
}

/**
 * @param casok the casok to set
 */
public void setCasok(int casok) {
    this.casok = casok;
}

/**
 * @return the bloquearPrePol
 */
public boolean isBloquearPrePol() {
    return bloquearPrePol;
}

/**
 * @param bloquearPrePol the bloquearPrePol to set
 */
public void setBloquearPrePol(boolean bloquearPrePol) {
    this.bloquearPrePol = bloquearPrePol;
}

/**
 * @return the dias_contrato
 */
public int getDiasContrato() {
    return dias_contrato;
}
```

```
    /**
     * @param dias_contrato the dias_contrato to set
     */
    public void setDiasContrato(int dias_contrato) {
        this.dias_contrato = dias_contrato;
    }

    /**
     * @return the ff_amp
     */
    public Date getFfAmp() {
        return ff_amp;
    }

    /**
     * @param ff_amp the ff_amp to set
     */
    public void setFfAmp(Date ff_amp) {
        this.ff_amp = ff_amp;
    }

    /**
     * @return the idu
     */
    public int getIdu() {
        return idu;
    }

    /**
     * @param idu the idu to set
     */
    public void setIdu(int idu) {
        this.idu = idu;
    }

    /**
     * @return the dependiente
     */
    public int getDependiente() {
        return dependiente;
    }

    /**
     * @param dependiente the dependiente to set
     */
    public void setDependiente(int dependiente) {
        this.dependiente = dependiente;
    }

}
```

03. Clase Presupuesto

```
package OBJETOS;

public class cC1Presupuesto
```

```
{
    private int idps;
    private int idova;
    private int idfm;
    private String idpar;
    private String descripcion;
    private String unidad;
    private double metrado;
    private double pre_unit;
    private double presupuesto;
    private String idpardep;

    public cC1Presupuesto() {
    }

    public cC1Presupuesto(int idps, int idova, int idfm, String idpar, String
descripcion,  String  unidad,  double  metrado,  double  pre_unit,  double
presupuesto, String idpardep) {
        this.idps = idps;
        this.idova = idova;
        this.idfm = idfm;
        this.idpar = idpar;
        this.descripcion = descripcion;
        this.unidad = unidad;
        this.metrado = metrado;
        this.pre_unit = pre_unit;
        this.presupuesto = presupuesto;
        this.idpardep = idpardep;
    }

    /**
     * @return the idps
     */
    public int getIdps() {
        return idps;
    }

    /**
     * @param idps the idps to set
     */
    public void setIdps(int idps) {
        this.idps = idps;
    }

    /**
     * @return the idova
     */
    public int getIdova() {
        return idova;
    }

    /**
     * @param idova the idova to set
     */
    public void setIdova(int idova) {
        this.idova = idova;
    }
```

```java
/**
 * @return the idfm
 */
public int getIdfm() {
    return idfm;
}

/**
 * @param idfm the idfm to set
 */
public void setIdfm(int idfm) {
    this.idfm = idfm;
}

/**
 * @return the idpar
 */
public String getIdpar() {
    return idpar;
}

/**
 * @param idpar the idpar to set
 */
public void setIdpar(String idpar) {
    this.idpar = idpar;
}

/**
 * @return the descripcion
 */
public String getDescripcion() {
    return descripcion;
}

/**
 * @param descripcion the descripcion to set
 */
public void setDescripcion(String descripcion) {
    this.descripcion = descripcion;
}

/**
 * @return the unidad
 */
public String getUnidad() {
    return unidad;
}

/**
 * @param unidad the unidad to set
 */
public void setUnidad(String unidad) {
    this.unidad = unidad;
}

/**
```

```
     * @return the metrado
     */
    public double getMetrado() {
        return metrado;
    }

    /**
     * @param metrado the metrado to set
     */
    public void setMetrado(double metrado) {
        this.metrado = metrado;
    }

    /**
     * @return the pre_unit
     */
    public double getPre_unit() {
        return pre_unit;
    }

    /**
     * @param pre_unit the pre_unit to set
     */
    public void setPre_unit(double pre_unit) {
        this.pre_unit = pre_unit;
    }

    /**
     * @return the presupuesto
     */
    public double getPresupuesto() {
        return presupuesto;
    }

    /**
     * @param presupuesto the presupuesto to set
     */
    public void setPresupuesto(double presupuesto) {
        this.presupuesto = presupuesto;
    }

    /**
     * @return the idpardep
     */
    public String getIdpardep() {
        return idpardep;
    }

    /**
     * @param idpardep the idpardep to set
     */
    public void setIdpardep(String idpardep) {
        this.idpardep = idpardep;
    }

}
```

04. Clase formulas

```
package OBJETOS;

public class cC1Formulas {

    private int idfm;
    private int idova;
    private int idesp;
    private boolean estadoPre;
    private boolean estadoK;
    private int pos;

    private cEspecialidad oe;

    public cC1Formulas() {
        oe = new cEspecialidad();
    }

    public cC1Formulas(int idfm, int idova, int idesp, boolean estadoPre,
boolean estadoK, int pos) {
        this.idfm = idfm;
        this.idova = idova;
        this.idesp = idesp;
        this.estadoPre = estadoPre;
        this.estadoK = estadoK;
        this.pos = pos;
        oe = new cEspecialidad();
    }

    /**
     * @return the idfm
     */
    public int getIdfm() {
        return idfm;
    }

    /**
     * @param idfm the idfm to set
     */
    public void setIdfm(int idfm) {
        this.idfm = idfm;
    }

    /**
     * @return the idesp
     */
    public int getIdesp() {
        return idesp;
    }

    /**
     * @param idesp the idesp to set
     */
    public void setIdesp(int idesp) {
        this.idesp = idesp;
    }
```

```java
/**
 * @return the estadoPre
 */
public boolean isEstadoPre() {
    return estadoPre;
}

/**
 * @param estadoPre the estadoPre to set
 */
public void setEstadoPre(boolean estadoPre) {
    this.estadoPre = estadoPre;
}

/**
 * @return the estadoK
 */
public boolean isEstadoK() {
    return estadoK;
}

/**
 * @param estadoK the estadoK to set
 */
public void setEstadoK(boolean estadoK) {
    this.estadoK = estadoK;
}

/**
 * @return the pos
 */
public int getPos() {
    return pos;
}

/**
 * @param pos the pos to set
 */
public void setPos(int pos) {
    this.pos = pos;
}

/**
 * @return the oe
 */
public cEspecialidad getOe() {
    return oe;
}

/**
 * @param oe the oe to set
 */
public void setOe(cEspecialidad oe) {
    this.oe = oe;
}

/**
```

```
     * @return the idova
     */
    public int getIdova() {
        return idova;
    }

    /**
     * @param idova the idova to set
     */
    public void setIdova(int idova) {
        this.idova = idova;
    }

}
```

05. Clase Usuario

```
package OBJETOS;

public class cUsuario {

    private int idu;
    private String nombre_completo;
    private String nick;
    private String clave;
    private int tipo;
    private int estado;
    private int estado2;

    public cUsuario() {
    }

    public cUsuario(int idu, String nombre_completo, String nick, String
clave, int tipo, int estado, int estado2) {
        this.idu = idu;
        this.nombre_completo = nombre_completo;
        this.nick = nick;
        this.clave = clave;
        this.tipo = tipo;
        this.estado = estado;
        this.estado2 = estado2;
    }

    /**
     * @return the idu
     */
    public int getIdu() {
        return idu;
    }

    /**
     * @param idu the idu to set
     */
    public void setIdu(int idu) {
        this.idu = idu;
    }
```

```
/**
 * @return the nombre_completo
 */
public String getNombre_completo() {
    return nombre_completo;
}

/**
 * @param nombre_completo the nombre_completo to set
 */
public void setNombre_completo(String nombre_completo) {
    this.nombre_completo = nombre_completo;
}

/**
 * @return the nick
 */
public String getNick() {
    return nick;
}

/**
 * @param nick the nick to set
 */
public void setNick(String nick) {
    this.nick = nick;
}

/**
 * @return the clave
 */
public String getClave() {
    return clave;
}

/**
 * @param clave the clave to set
 */
public void setClave(String clave) {
    this.clave = clave;
}

/**
 * @return the tipo
 */
public int getTipo() {
    return tipo;
}

/**
 * @param tipo the tipo to set
 */
public void setTipo(int tipo) {
    this.tipo = tipo;
}

/**
```

```
     * @return the estado
     */
    public int getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(int estado) {
        this.estado = estado;
    }

    /**
     * @return the estado2
     */
    public int getEstado2() {
        return estado2;
    }

    /**
     * @param estado2 the estado2 to set
     */
    public void setEstado2(int estado2) {
        this.estado2 = estado2;
    }



}
```

06. Clase Valorizacion

```
package OBJETOS;

import java.util.Date;

public class cValorizacion {

    private int idval;
    private int idobra;
    private int nro_valorizacion;
    private String mes_pertenece;
    private Date fecha_registro;
    private int estado_proceso_1;
    private int estado_proceso_2;
    private boolean estado;

    public cValorizacion() {
    }

    public cValorizacion(int idval, int idobra, int nro_valorizacion, String
mes_pertenece,    Date    fecha_registro,    int    estado_proceso_1,    int
estado_proceso_2, boolean estado) {
        this.idval = idval;
```

```
        this.idobra = idobra;
        this.nro_valorizacion = nro_valorizacion;
        this.mes_pertenece = mes_pertenece;
        this.fecha_registro = fecha_registro;
        this.estado_proceso_1 = estado_proceso_1;
        this.estado_proceso_2 = estado_proceso_2;
        this.estado = estado;
    }

    public String getNroValAlterado() {
        return (nro_valorizacion < 10 ? "0" + nro_valorizacion : "" +
nro_valorizacion);
    }

    /**
     * @return the idval
     */
    public int getIdval() {
        return idval;
    }

    /**
     * @param idval the idval to set
     */
    public void setIdval(int idval) {
        this.idval = idval;
    }

    /**
     * @return the idobra
     */
    public int getIdobra() {
        return idobra;
    }

    /**
     * @param idobra the idobra to set
     */
    public void setIdobra(int idobra) {
        this.idobra = idobra;
    }

    /**
     * @return the nro_valorizacion
     */
    public int getNroValorizacion() {
        return nro_valorizacion;
    }

    /**
     * @param nro_valorizacion the nro_valorizacion to set
     */
    public void setNroValorizacion(int nro_valorizacion) {
        this.nro_valorizacion = nro_valorizacion;
    }

    /**
```

```
     * @return the mes_pertenece
     */
    public String getMesPertenece() {
        return mes_pertenece;
    }

    /**
     * @param mes_pertenece the mes_pertenece to set
     */
    public void setMesPertenece(String mes_pertenece) {
        this.mes_pertenece = mes_pertenece;
    }

    /**
     * @return the estado
     */
    public boolean getEstado() {
        return estado;
    }

    /**
     * @param estado the estado to set
     */
    public void setEstado(boolean estado) {
        this.estado = estado;
    }

    @Override
    public String toString() {
        return "VALORIZACIÓN N° " + (getNroValorizacion() < 10 ? "0" +
getNroValorizacion() : getNroValorizacion()) + " [" + getMesPertenece() +
"]"; //To change body of generated methods, choose Tools | Templates.
    }

    /**
     * @return the fecha_registro
     */
    public Date getFecReg() {
        return fecha_registro;
    }

    /**
     * @param fecha_registro the fecha_registro to set
     */
    public void setFecReg(Date fecha_registro) {
        this.fecha_registro = fecha_registro;
    }

    /**
     * @return the estado_proceso_1
     */
    public int getEstado_proceso_1() {
        return estado_proceso_1;
    }

    /**
     * @param estado_proceso_1 the estado_proceso_1 to set
```

```
     */
    public void setEstado_proceso_1(int estado_proceso_1) {
        this.estado_proceso_1 = estado_proceso_1;
    }

    /**
     * @return the estado_proceso_2
     */
    public int getEstado_proceso_2() {
        return estado_proceso_2;
    }

    /**
     * @param estado_proceso_2 the estado_proceso_2 to set
     */
    public void setEstado_proceso_2(int estado_proceso_2) {
        this.estado_proceso_2 = estado_proceso_2;
    }

}
```

07. Clase DeduccionAdelantoDirecto

```
package OBJETOS;

public class cDeduccionAdelantoDirecto {

    private int iddad;
    private int idmcr;
    private int idval;
    private int nro_valorizacion;
    private String mes_anio;
    private double monto_metrado;
    private double adelanto;
    private double monto_contrato;
    private double kr;
    private double ka;
    private double deduccion;
    private String mes_anio_deducido;
    private int mes_activo;

    public cDeduccionAdelantoDirecto() {
    }

    public cDeduccionAdelantoDirecto(int iddad, int idmcr, int idval, int
nro_valorizacion, String mes_anio, double monto_metrado, double adelanto,
double monto_contrato, double kr, double ka, double deduccion, String
mes_anio_deducido, int mes_activo) {
        this.iddad = iddad;
        this.idmcr = idmcr;
        this.idval = idval;
        this.nro_valorizacion = nro_valorizacion;
        this.mes_anio = mes_anio;
        this.monto_metrado = monto_metrado;
        this.adelanto = adelanto;
        this.monto_contrato = monto_contrato;
```

```
        this.kr = kr;
        this.ka = ka;
        this.deduccion = deduccion;
        this.mes_anio_deducido = mes_anio_deducido;
        this.mes_activo = mes_activo;
    }



    /**
     * @return the idvd
     */
    public int getIddad() {
        return iddad;
    }

    /**
     * @param idvd the idvd to set
     */
    public void setIddad(int idvd) {
        this.iddad = idvd;
    }

    /**
     * @return the idmia
     */
    public int getIdmcr() {
        return idmcr;
    }

    /**
     * @param idmia the idvlp to set
     */
    public void setIdmcr(int idmia) {
        this.idmcr = idmia;
    }

    /**
     * @return the idval
     */
    public int getIdval() {
        return idval;
    }

    /**
     * @param idval the idval to set
     */
    public void setIdval(int idval) {
        this.idval = idval;
    }

    /**
     * @return the nro_valorizacion
     */
    public int getNroValorizacion() {
        return nro_valorizacion;
    }
```

```
/**
 * @param nro_valorizacion the nro_valorizacion to set
 */
public void setNroValorizacion(int nro_valorizacion) {
    this.nro_valorizacion = nro_valorizacion;
}

/**
 * @return the mes_anio
 */
public String getMesAnio() {
    return mes_anio;
}

/**
 * @param mes_anio the mes_anio to set
 */
public void setMesAnio(String mes_anio) {
    this.mes_anio = mes_anio;
}

/**
 * @return the monto_metrado
 */
public double getMonMet() {
    return monto_metrado;
}

/**
 * @param monto_metrado the monto_metrado to set
 */
public void setMonMet(double monto_metrado) {
    this.monto_metrado = monto_metrado;
}

/**
 * @return the adelanto
 */
public double getAde() {
    return adelanto;
}

/**
 * @param adelanto the adelanto to set
 */
public void setAde(double adelanto) {
    this.adelanto = adelanto;
}

/**
 * @return the monto_contrato
 */
public double getMonCon() {
    return monto_contrato;
}

/**
```

```
     * @param monto_contrato the monto_contrato to set
     */
    public void setMonCon(double monto_contrato) {
        this.monto_contrato = monto_contrato;
    }

    /**
     * @return the kr
     */
    public double getKr() {
        return kr;
    }

    /**
     * @param kr the kr to set
     */
    public void setKr(double kr) {
        this.kr = kr;
    }

    /**
     * @return the ka
     */
    public double getKa() {
        return ka;
    }

    /**
     * @param ka the ka to set
     */
    public void setKa(double ka) {
        this.ka = ka;
    }

    /**
     * @return the deduccion
     */
    public double getDed() {
        return deduccion;
    }

    /**
     * @param deduccion the deduccion to set
     */
    public void setDed(double deduccion) {
        this.deduccion = deduccion;
    }

    /**
     * @return the mes_anio_deducido
     */
    public String getMesADed() {
        return mes_anio_deducido;
    }

    /**
     * @param mes_anio_deducido the mes_anio_deducido to set
```

```
     */
    public void setMesADed(String mes_anio_deducido) {
        this.mes_anio_deducido = mes_anio_deducido;
    }

    public String mostrarDatos() {
        return "" + this.iddad + "\t"
                + this.idmcr + "\t"
                + this.idval + "\t"
                + this.nro_valorizacion + "\t"
                + this.mes_anio + "\t"
                + this.monto_metrado + "\t"
                + this.adelanto + "\t"
                + this.monto_contrato + "\t"
                + this.kr + "\t"
                + this.ka + "\t"
                + this.deduccion + "\t"
                + this.mes_anio_deducido;
    }

    /**
     * @return the mes_activo
     */
    public int getMesActivo() {
        return mes_activo;
    }

    /**
     * @param mes_activo the mes_activo to set
     */
    public void setMesActivo(int mes_activo) {
        this.mes_activo = mes_activo;
    }

}
```

08. Clase DeduccionAdelantoMateriales

```
package OBJETOS;

/**
 *
 * @author pekeño
 */
public class cDeduccionAdelantoMateriales {

    private int iddam;
    private int idmim;
    private int idval;
    private int idpsm;
    private int nro_valorizacion;
    private String mes_anio;
    private double monto_metrado;
    private double coeficiente_incidencia;
    private double indice_mes_base;
    private double indice_mes_adelanto;
```

```
    private double indice_mes_reajuste;
    private double adelanto;
    private double monto_maximo_utilizado;
    private double adelanto_deducible;
    private double saldo_adelanto;
    private double deduccion;
    private String mes_anio_pago;
    private int mes_activo;

    public cDeduccionAdelantoMateriales() {
    }

    public cDeduccionAdelantoMateriales(int iddam, int idmim, int idval, int
idpsm, int nro_valorizacion, String mes_anio, double monto_metrado, double
coeficiente_incidencia, double indice_mes_base, double indice_mes_adelanto,
double indice_mes_reajuste, double adelanto, double monto_maximo_utilizado,
double adelanto_deducible, double saldo_adelanto, double deduccion, String
mes_anio_pago, int mes_activo) {
        this.iddam = iddam;
        this.idmim = idmim;
        this.idval = idval;
        this.idpsm = idpsm;
        this.nro_valorizacion = nro_valorizacion;
        this.mes_anio = mes_anio;
        this.monto_metrado = monto_metrado;
        this.coeficiente_incidencia = coeficiente_incidencia;
        this.indice_mes_base = indice_mes_base;
        this.indice_mes_adelanto = indice_mes_adelanto;
        this.indice_mes_reajuste = indice_mes_reajuste;
        this.adelanto = adelanto;
        this.monto_maximo_utilizado = monto_maximo_utilizado;
        this.adelanto_deducible = adelanto_deducible;
        this.saldo_adelanto = saldo_adelanto;
        this.deduccion = deduccion;
        this.mes_anio_pago = mes_anio_pago;
        this.mes_activo = mes_activo;
    }

    /**
     * @return the iddam
     */
    public int getIddam() {
        return iddam;
    }

    /**
     * @param iddam the iddam to set
     */
    public void setIddam(int iddam) {
        this.iddam = iddam;
    }

    /**
     * @return the idmim
     */
    public int getIdmim() {
        return idmim;
```

```
}

/**
 * @param idmim the idvlp to set
 */
public void setIdmim(int idmim) {
    this.idmim = idmim;
}

/**
 * @return the idval
 */
public int getIdval() {
    return idval;
}

/**
 * @param idval the idval to set
 */
public void setIdval(int idval) {
    this.idval = idval;
}

/**
 * @return the idpsm
 */
public int getIdpsm() {
    return idpsm;
}

/**
 * @param idpsm the idvad to set
 */
public void setIdpsm(int idpsm) {
    this.idpsm = idpsm;
}

/**
 * @return the nro_valorizacion
 */
public int getNroVal() {
    return nro_valorizacion;
}

/**
 * @param nro_valorizacion the nro_valorizacion to set
 */
public void setNroVal(int nro_valorizacion) {
    this.nro_valorizacion = nro_valorizacion;
}

/**
 * @return the mes_anio
 */
public String getMesAnio() {
    return mes_anio;
}
```

```
/**
 * @param mes_anio the mes_anio to set
 */
public void setMesAnio(String mes_anio) {
    this.mes_anio = mes_anio;
}

/**
 * @return the monto_metrado
 */
public double getMonMet() {
    return monto_metrado;
}

/**
 * @param monto_metrado the monto_metrado to set
 */
public void setMonMet(double monto_metrado) {
    this.monto_metrado = monto_metrado;
}

/**
 * @return the coeficiente_incidencia
 */
public double getCoeInc() {
    return coeficiente_incidencia;
}

/**
 * @param coeficiente_incidencia the coeficiente_incidencia to set
 */
public void setCoeInc(double coeficiente_incidencia) {
    this.coeficiente_incidencia = coeficiente_incidencia;
}

/**
 * @return the indice_mes_base
 */
public double getImo() {
    return indice_mes_base;
}

/**
 * @param indice_mes_base the indice_mes_base to set
 */
public void setImo(double indice_mes_base) {
    this.indice_mes_base = indice_mes_base;
}

/**
 * @return the indice_mes_adelanto
 */
public double getIma() {
    return indice_mes_adelanto;
}
```

```
/**
 * @param indice_mes_adelanto the indice_mes_adelanto to set
 */
public void setIma(double indice_mes_adelanto) {
    this.indice_mes_adelanto = indice_mes_adelanto;
}

/**
 * @return the indice_mes_reajuste
 */
public double getImr() {
    return indice_mes_reajuste;
}

/**
 * @param indice_mes_reajuste the indice_mes_reajuste to set
 */
public void setImr(double indice_mes_reajuste) {
    this.indice_mes_reajuste = indice_mes_reajuste;
}

/**
 * @return the adelanto
 */
public double getA() {
    return adelanto;
}

/**
 * @param adelanto the adelanto to set
 */
public void setA(double adelanto) {
    this.adelanto = adelanto;
}

/**
 * @return the monto_maximo_utilizado
 */
public double getMmu() {
    return monto_maximo_utilizado;
}

/**
 * @param monto_maximo_utilizado the monto_maximo_utilizado to set
 */
public void setMmu(double monto_maximo_utilizado) {
    this.monto_maximo_utilizado = monto_maximo_utilizado;
}

/**
 * @return the adelanto_deducible
 */
public double getAdeDed() {
    return adelanto_deducible;
}

/**
```

```
     * @param adelanto_deducible the adelanto_deducible to set
     */
    public void setAdeDed(double adelanto_deducible) {
        this.adelanto_deducible = adelanto_deducible;
    }

    /**
     * @return the saldo_adelanto
     */
    public double getSalAde() {
        return saldo_adelanto;
    }

    /**
     * @param saldo_adelanto the saldo_adelanto to set
     */
    public void setSalAde(double saldo_adelanto) {
        this.saldo_adelanto = saldo_adelanto;
    }

    /**
     * @return the deduccion
     */
    public double getDed() {
        return deduccion;
    }

    /**
     * @param deduccion the deduccion to set
     */
    public void setDed(double deduccion) {
        this.deduccion = deduccion;
    }

    /**
     * @return the mes_anio_pago
     */
    public String getMesPag() {
        return mes_anio_pago;
    }

    /**
     * @param mes_anio_pago the mes_anio_pago to set
     */
    public void setMesPag(String mes_anio_pago) {
        this.mes_anio_pago = mes_anio_pago;
    }

    public String mostrar() {
        return "" + this.iddam + "\t"
                + this.idmim + "\t"
                + this.idval + "\t"
                + this.idpsm + "\t"
                + this.nro_valorizacion + "\t"
                + this.mes_anio + "\t"
                + this.monto_metrado + "\t"
                + this.coeficiente_incidencia + "\t"
```

```
                + this.indice_mes_base + "\t"
                + this.indice_mes_adelanto + "\t"
                + this.indice_mes_reajuste + "\t"
                + this.adelanto + "\t"
                + this.monto_maximo_utilizado + "\t"
                + this.adelanto_deducible + "\t"
                + this.saldo_adelanto + "\t"
                + this.deduccion + "\t"
                + this.mes_anio_pago;
    }

    /**
     * @return the mes_activo
     */
    public int getMesActivo() {
        return mes_activo;
    }

    /**
     * @param mes_activo the mes_activo to set
     */
    public void setMesActivo(int mes_activo) {
        this.mes_activo = mes_activo;
    }

}
```

09. Clase AmortizacionAdelantoDirecto

```
package OBJETOS;

public class cAmortizacionAdelantoDirecto {

    private int idaad;
    private int idfm;
    private int idval;
    private int nro_valorizacion;
    private String mes_anio;
    private double monto_metrado;
    private double amortizacion;
    private double amortizacion_acumulada;
    private double saldo_amortizar;
    private double amortizacion_real;
    private double amortizacion_acumulada_real;
    private int mes_activo ;

    public cAmortizacionAdelantoDirecto() {
    }

    public cAmortizacionAdelantoDirecto(int idaad, int idfm, int idval, int
nro_valorizacion, String mes_anio, double monto_metrado, double amortizacion,
double     amortizacion_acumulada,     double     saldo_amortizar,     double
amortizacion_real, double amortizacion_acumulada_real, int mes_activo) {
        this.idaad = idaad;
        this.idfm = idfm;
```

```
        this.idval = idval;
        this.nro_valorizacion = nro_valorizacion;
        this.mes_anio = mes_anio;
        this.monto_metrado = monto_metrado;
        this.amortizacion = amortizacion;
        this.amortizacion_acumulada = amortizacion_acumulada;
        this.saldo_amortizar = saldo_amortizar;
        this.amortizacion_real = amortizacion_real;
        this.amortizacion_acumulada_real = amortizacion_acumulada_real;
        this.mes_activo = mes_activo;
    }


    /**
     * @return the idaad
     */
    public int getIdaad() {
        return idaad;
    }

    /**
     * @param idaad the idaad to set
     */
    public void setIdaad(int idaad) {
        this.idaad = idaad;
    }

    /**
     * @return the idfm
     */
    public int getIdfm() {
        return idfm;
    }

    /**
     * @param idfm the idvlp to set
     */
    public void setIdfm(int idfm) {
        this.idfm = idfm;
    }

    /**
     * @return the idval
     */
    public int getIdval() {
        return idval;
    }

    /**
     * @param idval the idval to set
     */
    public void setIdval(int idval) {
        this.idval = idval;
    }

    /**
     * @return the nro_valorizacion
```

```
 */
public int getNroVal() {
    return nro_valorizacion;
}

/**
 * @param nro_valorizacion the nro_valorizacion to set
 */
public void setNroVal(int nro_valorizacion) {
    this.nro_valorizacion = nro_valorizacion;
}

/**
 * @return the mes_anio
 */
public String getMesAnio() {
    return mes_anio;
}

/**
 * @param mes_anio the mes_anio to set
 */
public void setMesAnio(String mes_anio) {
    this.mes_anio = mes_anio;
}

/**
 * @return the monto_metrado
 */
public double getMonMet() {
    return monto_metrado;
}

/**
 * @param monto_metrado the monto_metrado to set
 */
public void setMonMet(double monto_metrado) {
    this.monto_metrado = monto_metrado;
}

/**
 * @return the amortizacion
 */
public double getAmor() {
    return amortizacion;
}

/**
 * @param amortizacion the amortizacion to set
 */
public void setAmor(double amortizacion) {
    this.amortizacion = amortizacion;
}

/**
 * @return the amortizacion_acumulada
 */
```

```
    public double getAmorAcu() {
        return amortizacion_acumulada;
    }

    /**
     * @param amortizacion_acumulada the amortizacion_acumulada to set
     */
    public void setAmorAcu(double amortizacion_acumulada) {
        this.amortizacion_acumulada = amortizacion_acumulada;
    }

    /**
     * @return the saldo_amortizar
     */
    public double getSalAmo() {
        return saldo_amortizar;
    }

    /**
     * @param saldo_amortizar the saldo_amortizar to set
     */
    public void setSalAmor(double saldo_amortizar) {
        this.saldo_amortizar = saldo_amortizar;
    }

    /**
     * @return the amortizacion_real
     */
    public double getAmorReal() {
        return amortizacion_real;
    }

    /**
     * @param amortizacion_real the amortizacion_real to set
     */
    public void setAmorReal(double amortizacion_real) {
        this.amortizacion_real = amortizacion_real;
    }

    /**
     * @return the amortizacion_acumulada_real
     */
    public double getAmorAcuReal() {
        return amortizacion_acumulada_real;
    }

    /**
     * @param amortizacion_acumulada_real the amortizacion_acumulada_real to
set
     */
    public void setAmorAcuReal(double amortizacion_acumulada_real) {
        this.amortizacion_acumulada_real = amortizacion_acumulada_real;
    }

    public String mostrar() {
        return "" + this.idaad + "\t"
                + this.idfm + "\t"
```

```
                + this.idval + "\t"
                + this.nro_valorizacion + "\t"
                + this.mes_anio + "\t"
                + this.monto_metrado + "\t"
                + this.amortizacion + "\t"
                + this.amortizacion_acumulada + "\t"
                + this.saldo_amortizar + "\t"
                + this.amortizacion_real + "\t"
                + this.amortizacion_acumulada_real;
    }

    /**
     * @return the mes_activo
     */
    public int getMes_activo() {
        return mes_activo;
    }

    /**
     * @param mes_activo the mes_activo to set
     */
    public void setMes_activo(int mes_activo) {
        this.mes_activo = mes_activo;
    }

}
```

01. Clase AmortizacionAdelantoMateriales

```
package OBJETOS;

public class cAmortizacionAdelantoMateriales {

    private int idaam;
    private int idfm;
    private int idval;
    private int idpsm;
    private int nro_valorizacion;
    private String mes_anio;
    private double monto_metrado;
    private double coeficiente_incidencia;
    private double indice_mes_base;
    private double indice_mes_adelanto;
    private double indice_mes_reajuste;
    private double mv_ci;
    private double amortizacion_calculada;
    private double adelanto;
    private double amortizacion_real;
    private double saldo_amortizar;
    private int mes_activo;

    public cAmortizacionAdelantoMateriales() {
    }
```

```
    public cAmortizacionAdelantoMateriales(int idaam, int idfm, int idval,
int idpsm, int nro_valorizacion, String mes_anio, double monto_metrado,
double coeficiente_incidencia, double indice_mes_base, double
indice_mes_adelanto, double indice_mes_reajuste, double mv_ci, double
amortizacion_calculada, double adelanto, double amortizacion_real, double
saldo_amortizar, int mes_activo) {
        this.idaam = idaam;
        this.idfm = idfm;
        this.idval = idval;
        this.idpsm = idpsm;
        this.nro_valorizacion = nro_valorizacion;
        this.mes_anio = mes_anio;
        this.monto_metrado = monto_metrado;
        this.coeficiente_incidencia = coeficiente_incidencia;
        this.indice_mes_base = indice_mes_base;
        this.indice_mes_adelanto = indice_mes_adelanto;
        this.indice_mes_reajuste = indice_mes_reajuste;
        this.mv_ci = mv_ci;
        this.amortizacion_calculada = amortizacion_calculada;
        this.adelanto = adelanto;
        this.amortizacion_real = amortizacion_real;
        this.saldo_amortizar = saldo_amortizar;
        this.mes_activo = mes_activo;
    }



    /**
     * @return the idaam
     */
    public int getIdaam() {
        return idaam;
    }

    /**
     * @param idaam the idaam to set
     */
    public void setIdaam(int idaam) {
        this.idaam = idaam;
    }

    /**
     * @return the idfm
     */
    public int getIdfm() {
        return idfm;
    }

    /**
     * @param idfm the idvlp to set
     */
    public void setIdfm(int idfm) {
        this.idfm = idfm;
    }

    /**
     * @return the idval
```

```
     */
    public int getIdval() {
        return idval;
    }

    /**
     * @param idval the idval to set
     */
    public void setIdval(int idval) {
        this.idval = idval;
    }

    /**
     * @return the idpsm
     */
    public int getIdpsm() {
        return idpsm;
    }

    /**
     * @param idpsm the idvad to set
     */
    public void setIdpsm(int idpsm) {
        this.idpsm = idpsm;
    }

    /**
     * @return the nro_valorizacion
     */
    public int getNroVal() {
        return nro_valorizacion;
    }

    /**
     * @param nro_valorizacion the nro_valorizacion to set
     */
    public void setNroVal(int nro_valorizacion) {
        this.nro_valorizacion = nro_valorizacion;
    }

    /**
     * @return the mes_anio
     */
    public String getMesAnio() {
        return mes_anio;
    }

    /**
     * @param mes_anio the mes_anio to set
     */
    public void setMesAnio(String mes_anio) {
        this.mes_anio = mes_anio;
    }

    /**
     * @return the monto_metrado
     */
```

```
public double getMonMet() {
    return monto_metrado;
}

/**
 * @param monto_metrado the monto_metrado to set
 */
public void setMonMet(double monto_metrado) {
    this.monto_metrado = monto_metrado;
}

/**
 * @return the coeficiente_incidencia
 */
public double getCoeInc() {
    return coeficiente_incidencia;
}

/**
 * @param coeficiente_incidencia the coeficiente_incidencia to set
 */
public void setCoeInc(double coeficiente_incidencia) {
    this.coeficiente_incidencia = coeficiente_incidencia;
}

/**
 * @return the indice_mes_base
 */
public double getImo() {
    return indice_mes_base;
}

/**
 * @param indice_mes_base the indice_mes_base to set
 */
public void setImo(double indice_mes_base) {
    this.indice_mes_base = indice_mes_base;
}

/**
 * @return the indice_mes_adelanto
 */
public double getIma() {
    return indice_mes_adelanto;
}

/**
 * @param indice_mes_adelanto the indice_mes_adelanto to set
 */
public void setIma(double indice_mes_adelanto) {
    this.indice_mes_adelanto = indice_mes_adelanto;
}

/**
 * @return the indice_mes_reajuste
 */
public double getImr() {
```

```
        return indice_mes_reajuste;
    }

    /**
     * @param indice_mes_reajuste the indice_mes_reajuste to set
     */
    public void setImr(double indice_mes_reajuste) {
        this.indice_mes_reajuste = indice_mes_reajuste;
    }

    /**
     * @return the mv_ci
     */
    public double getMvCi() {
        return mv_ci;
    }

    /**
     * @param mv_ci the mv_ci to set
     */
    public void setMvCi(double mv_ci) {
        this.mv_ci = mv_ci;
    }

    /**
     * @return the amortizacion_calculada
     */
    public double getAmoCal() {
        return amortizacion_calculada;
    }

    /**
     * @param amortizacion_calculada the amortizacion_calculada to set
     */
    public void setAmoCal(double amortizacion_calculada) {
        this.amortizacion_calculada = amortizacion_calculada;
    }

    /**
     * @return the adelanto
     */
    public double getAde() {
        return adelanto;
    }

    /**
     * @param adelanto the adelanto to set
     */
    public void setAde(double adelanto) {
        this.adelanto = adelanto;
    }

    /**
     * @return the amortizacion_real
     */
    public double getAmoReal() {
        return amortizacion_real;
```

```
    }

    /**
     * @param amortizacion_real the amortizacion_real to set
     */
    public void setAmoReal(double amortizacion_real) {
        this.amortizacion_real = amortizacion_real;
    }

    /**
     * @return the saldo_amortizar
     */
    public double getSalAmor() {
        return saldo_amortizar;
    }

    /**
     * @param saldo_amortizar the saldo_amortizar to set
     */
    public void setSalAmor(double saldo_amortizar) {
        this.saldo_amortizar = saldo_amortizar;
    }

    public String mostrar() {
        return "" + this.idaam + "\t"
                + this.idfm + "\t"
                + this.idval + "\t"
                + this.idpsm + "\t"
                + this.nro_valorizacion + "\t"
                + this.mes_anio + "\t"
                + this.monto_metrado + "\t"
                + this.coeficiente_incidencia + "\t"
                + this.indice_mes_base + "\t"
                + this.indice_mes_adelanto + "\t"
                + this.indice_mes_reajuste + "\t"
                + this.mv_ci + "\t"
                + this.amortizacion_calculada + "\t"
                + this.adelanto + "\t"
                + this.amortizacion_real + "\t"
                + this.saldo_amortizar;
    }

    /**
     * @return the mes_activo
     */
    public int getMes_activo() {
        return mes_activo;
    }

    /**
     * @param mes_activo the mes_activo to set
     */
    public void setMes_activo(int mes_activo) {
        this.mes_activo = mes_activo;
    }

}
```

01) Clase dObras

```
package DAO;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import OBJETOS.cObras;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.SQLException;

public class dObras {

    public static int CON_IGV = 1;
    public static int SIN_IGV = 0;
    public static int CON_ACTUALIZAR_K = 1;
    public static int SIN_ACTUALIZAR_K = 0;

    private final String nombreClase = "dObras";

    public int getIdMax() {

        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT MAX(idova) id FROM `obras`";
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getInt("id");
                } catch (NumberFormatException e) {
                    System.out.println("Error no hay id max");
                    id = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("Ubicacion del Error[dObras]:{public int
getMaxId()}");
            System.out.println("[" + nombreClase + "]{}\n" + "Error :" +
e.getMessage());
        }
```

```
        return id;
    }

    public boolean crear(cObras o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }

            String sql = "INSERT INTO `obras`( `dependiente`,`idu`,`nob`,
`ent`, `lug`, `tob`, `sco`, `mej`, `pse`, `idphe`,"
                    + " `hec`, `idprv`, `rev`, `idprs`, `res`, `idpsp`,
`sup`, `fob`, `fcu`, `con`,`eigv`,"
                    + "`mco`,`mre`,`mcosi`,`mresi`, `mba`, `mad`,`mam`,
`pad`, `pam`, `casok`, `dias_contrato`, `ff_amp`, `est`) "
                    + "VALUES (" + o.getDependiente() + "," + o.getIdu() +
",'" + o.getNob() + "','" + o.getEnt() + "','" + o.getLug() + "'," + o.getTob()
+ "," + o.getSco() + ","
                    + "" + o.getMej() + "," + o.getPse() + "," + o.getIdphe()
+ ",'" + o.getHec() + "'," + o.getIdprv() + ",'" + o.getRev() + "',"
                    + "" + o.getIdprs() + ",'" + o.getRes() + "'," +
o.getIdpsp() + ",'" + o.getSup() + "','" + fechas.formatAMD(o.getFob()) +
"','" + fechas.formatAMD(o.getFcu()) + "','"
                    + o.getCon() + "'," + o.getEigv() + "," + o.getMco() +
"," + o.getMre() + "," + o.getMcosi() + "," + o.getMresi() + ",'"
                    + fechas.formatAMD(o.getMba()) + "','" +
fechas.formatAMD(o.getMad()) + "','" + fechas.formatAMD(o.getMam()) + "',"
                    + o.getPad() + "," + o.getPam() + "," + o.getCasok() +
"," + o.getDiasContrato() + ",'" + fechas.formatAMD(o.getFfAmp()) + "'," +
o.getEst() + ")";
            System.out.println("dObras->crear:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean modificar(cObras o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
```

```
            }

            String sql = "UPDATE `obras` SET `nob`='" + o.getNob() +
"','ent`='" + o.getEnt() + "',`lug`='" + o.getLug() + "',"
                    + "`tob`=" + o.getTob() + ",`sco`=" + o.getSco() +
",`mej`=" + o.getMej() + ",`pse`=" + o.getPse() + ",`idphe`=" + o.getIdphe()
+ ","
                    + "`hec`='" + o.getHec() + "',`idprv`=" + o.getIdprv() +
",`rev`='" + o.getRev() + "',`idprs`=" + o.getIdprs() + ",`res`='" +
o.getRes() + "',"
                    + "`idpsp`=" + o.getIdpsp() + ",`sup`='" + o.getSup() +
"',`fob`='" + fechas.formatAMD(o.getFob()) + "',`fcu`='" +
fechas.formatAMD(o.getFcu()) + "',`con`='" + o.getCon() + "',"
                    + "`mco`=" + o.getMco() + ",`eigv`=" + o.getEigv() +
",`mre`=" + o.getMre() + ",`mcosi`=" + o.getMcosi() + ",`mresi`=" +
o.getMresi() + ",`mba`='" + fechas.formatAMD(o.getMba()) + "',"
                    + "`mad`='" + fechas.formatAMD(o.getMad()) + "',`mam`='"
+ fechas.formatAMD(o.getMam()) + "',`est`=" + o.getEst() + " ,`pad`=" +
o.getPad() + ",`pam`=" + o.getPam() + ",`dias_contrato`=" +
o.getDiasContrato() + ",`ff_amp`='" + fechas.formatAMD(o.getFfAmp()) + "' "
                    + "WHERE `idova`=" + o.getIdova();
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificar}\n" + error
+ e.getMessage());
        }
        return error == null;
    }

    public boolean modificarCasoK(int idobra, int casoK) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }

            String sql = "UPDATE `obras` SET `casok`=" + casoK + " "
                    + "WHERE `idova`=" + idobra;
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la OBRA: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
```

```
            System.out.println("[" + nombreClase + "]{modificarCasoK}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean modificarBloqueoPrePol(int idobra, boolean bloqueo) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }

            String sql = "UPDATE `obras` SET `bloquear_pre_pol`=" + bloqueo
+ " "
                    + "WHERE `idova`=" + idobra;
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la OBRA: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{modificarBloqueoPrePol}\n" + error + e.getMessage());
        }
        return error == null;
    }

    public boolean modificarEstado(String idestado) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `obras` SET `estado`=false WHERE `idobra`="
+ idestado;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificarEstado}\n" +
error + e.getMessage());
        }
        return error == null;
```

```
    }

    public boolean eliminar(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `obras` WHERE `idova`=" + id;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarObrasAdicionales(int idova) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `obras` WHERE `dependiente`=" + idova;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{eliminarObrasAdicionales}\n" + error + e.getMessage());
        }
        return error == null;
    }

    public cObras BuscarId(String id) {
        cObras p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
```

```
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from obras where idova="
+ id);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{BuscarId}\n" + "Error
:" + e.getMessage());
        }
        return p;
    }

    public List<cObras> ListTodo(int idu) {
        List<cObras> lista = new ArrayList<cObras>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from obras where
dependiente=0 and idu=" + idu + " order by fob asc");
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodo}\n" + "Error
:" + e.getMessage());
        }
        return lista;
    }

    public List<cObras> ListTodoIdovaDependiente(int idova) {
        List<cObras> lista = new ArrayList<cObras>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from obras where
dependiente=" + idova + " order by fob asc");
            while (rs.next()) {
                lista.add(getResult(rs));
            }
```

```
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodo}\n" + "Error
:" + e.getMessage());
        }
        return lista;
    }

    private cObras getResult(ResultSet rs) throws SQLException {
        cObras p = new cObras();
        p.setIdova(rs.getInt("idova"));
        p.setDependiente(rs.getInt("dependiente"));
        p.setIdu(rs.getInt("idu"));
        p.setNob(rs.getString("nob"));
        p.setEnt(rs.getString("ent"));
        p.setLug(rs.getString("lug"));
        p.setTob(rs.getInt("tob"));
        p.setSco(rs.getInt("sco"));
        p.setMej(rs.getInt("mej"));
        p.setPse(rs.getInt("pse"));

        p.setIdphe(rs.getInt("idphe"));
        p.setHec(rs.getString("hec"));

        p.setIdprv(rs.getInt("idprv"));
        p.setRev(rs.getString("rev"));

        p.setIdprs(rs.getInt("idprs"));
        p.setRes(rs.getString("res"));

        p.setIdpsp(rs.getInt("idpsp"));
        p.setSup(rs.getString("sup"));

        p.setFob(rs.getDate("fob"));
        p.setFcu(rs.getDate("fcu"));
        p.setCon(rs.getString("con"));

        p.setEigv(rs.getInt("eigv"));

        p.setMco(rs.getDouble("mco"));
        p.setMre(rs.getDouble("mre"));
        p.setMcosi(rs.getDouble("mcosi"));
        p.setMresi(rs.getDouble("mresi"));

        p.setMba(rs.getDate("mba"));

        p.setMad(rs.getDate("mad"));
        p.setMam(rs.getDate("mam"));

        p.setPad(rs.getDouble("pad"));
        p.setPam(rs.getDouble("pam"));

        p.setCasok(rs.getInt("casok"));
        p.setBloquearPrePol(rs.getBoolean("bloquear_pre_pol"));
```

```
        p.setDiasContrato(rs.getInt("dias_contrato"));
        p.setFfAmp(rs.getDate("ff_amp"));

        p.setEst(rs.getInt("est"));
        return p;
    }
}
```

02) Clase presupuesto

```
package DAO;

import OBJETOS.cC1Presupuesto;
import vconexion.conexion;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

public class dC1Presupuesto {

    private final String nombreClase = "dC1Presupuesto";

    public int getIdVlpa() {

        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT max(idps) id FROM `c1_presupuesto`";
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getInt("id");
                } catch (NumberFormatException e) {
                    System.out.println("Error no hay id max");
                    id = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{dValorizacionListaPartidasActivas}\n" + "Error :" + e.getMessage());
        }
```

```
        return id;
    }

    public boolean crear(cC1Presupuesto o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idps`, null,
            String sql = "INSERT INTO `c1_presupuesto`(`idova`, `idfm`,
`idpar`, `descripcion`, `unidad`, `metrado`, `pre_unit`, `presupuesto`,
`idpardep`) "
                    + "VALUES (" + o.getIdova() + "," + o.getIdfm() + ",'" +
o.getIdpar() + "',?,'" + o.getUnidad() + "'," + o.getMetrado() + "," +
o.getPre_unit() + "," + o.getPresupuesto() + ",'" + o.getIdpardep() + "')";
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            st.setString(1, o.getDescripcion());
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdfm(int idfm) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `c1_presupuesto` WHERE idfm=" + idfm;
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
```

```
        return error == null;
    }

    public List<cC1Presupuesto> listTodoIdovaIdfm(int idova, int idfm) {
        List<cC1Presupuesto> lista = new ArrayList<cC1Presupuesto>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from c1_presupuesto where idova=" + idova
+ " and idfm=" + idfm;
            System.out.println("sql :" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodoIdovaIdfm}\n"
+ "Error :" + e.getMessage());
            lista = new ArrayList<cC1Presupuesto>();
        }
        return lista;
    }

    public int contarIdObraIdFm(int idova, int idfm) {
        int n = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select count(idps) total from c1_presupuesto where
idova=" + idova + " and idfm=" + idfm;
            System.out.println("query :" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    n = rs.getInt("total");
                } catch (NumberFormatException e) {
                    n = 0;
                }

            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
```

```
            System.out.println("[" + nombreClase + "]{contarIdObraIdFm}\n" +
"Error :" + e.getMessage());
            n = 0;
        }
        return n;
    }

    public double montoPresupuestoIdObraIdfm(int idova, int idfm) {
        double n = 0.0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String   sql   =   "SELECT   SUM(presupuesto)   presupuesto   FROM
`c1_presupuesto` "
                    + "WHERE idova=" + idova + " AND idfm =" + idfm + " AND
LENGTH(unidad)>0";// + " AND length(idpar)=2";
            System.out.println("query :" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                n = rs.getDouble("presupuesto");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("["          +          nombreClase          +
"]{montoPresupuestoIdObraIdEsp}\n" + "Error :" + e.getMessage());
        }
        return n;
    }

    public double montoAnteriorIdObraIdfm(int idobra, int idfm, int idVal) {
        double n = 0.0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT SUM(v2.anterior_valorizado) anterior FROM
`c1_presupuesto` v1 "
                    + "INNER JOIN `valorizacion_datos` v2 ON v2.idps=v1.idps
"
                    + "WHERE v1.idova=" + idobra + " AND v1.idfm=" + idfm +
" and v2.idval=" + idVal + " and LENGTH(v1.unidad)>0";
            System.out.println("query :" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                n = rs.getDouble("anterior");
            }
            rs.close();
```

```
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("["          +          nombreClase          +
"]{montoAnteriorIdObraIdEsp}\n" + "Error :" + e.getMessage());
        }
        return n;
    }

    public double montoPresenteIdObraIdfm(int idobra, int idfm, int idVal) {
        double n = 0.0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT SUM(v2.presente_valorizado) presente FROM
`c1_presupuesto` v1 "
                    + "INNER JOIN `valorizacion_datos` v2 ON v2.idps=v1.idps
"
                    + "WHERE v1.idova=" + idobra + " AND v1.idfm=" + idfm +
" and v2.idval=" + idVal + " and LENGTH(v1.unidad)>0";
            System.out.println("query :" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                n = rs.getDouble("presente");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("["          +          nombreClase          +
"]{montoPresenteIdObraIdEsp}\n" + "Error :" + e.getMessage());
        }
        return n;
    }

    public double montoActualIdObraIdfm(int idobra, int idEsp, int idVal) {
        double n = 0.0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String  sql  =  "SELECT  SUM(v2.actual_valorizado)  actual  FROM
`c1_presupuesto` v1 "
                    + "INNER JOIN `valorizacion_datos` v2 ON v2.idps=v1.idps
"
                    + "WHERE v1.idova=" + idobra + " AND v1.idfm=" + idEsp +
" and v2.idval=" + idVal + " and LENGTH(v1.unidad)>0";
            System.out.println("query :" + sql);
            Statement st = con.createStatement();
```

```
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                n = rs.getDouble("actual");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{montoActualIdObraIdEsp}\n" + "Error :" + e.getMessage());
        }
        return n;
    }

    public double montoSaldoIdObraIdfm(int idobra, int idfm, int idVal) {
        double n = 0.0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT SUM(v2.saldo_valorizado) saldo FROM `c1_presupuesto` v1 "
                    + "INNER JOIN `valorizacion_datos` v2 ON v2.idps=v1.idps "
                    + "WHERE v1.idova=" + idobra + " AND v1.idfm=" + idfm + " and v2.idval=" + idVal + " and LENGTH(v1.unidad)>0";
            System.out.println("query :" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                n = rs.getDouble("saldo");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{montoSaldoIdObraIdEsp}\n" + "Error :" + e.getMessage());
        }
        return n;
    }
//*/

    private cC1Presupuesto getResult(ResultSet rs) throws SQLException {
        cC1Presupuesto p = new cC1Presupuesto();
        p.setIdps(rs.getInt("idps"));
        p.setIdova(rs.getInt("idova"));
        p.setIdfm(rs.getInt("idfm"));
        p.setIdpar(rs.getString("idpar"));
        p.setDescripcion(rs.getString("descripcion"));
        p.setUnidad(rs.getString("unidad"));
        p.setMetrado(rs.getDouble("metrado"));
        p.setPre_unit(rs.getDouble("pre_unit"));
```

```
        p.setPresupuesto(rs.getDouble("presupuesto"));
        p.setIdpardep(rs.getString("idpardep"));
        return p;
    }
}
```

03) Clase dformulas

```
package DAO;

import OBJETOS.cC1Formulas;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.sql.SQLException;
import OBJETOS.cEspecialidad;

public class dC1Formulas {

    private final String nombreClase = "dC1Formulas";

    public int getMaxId() {

        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT max(idfm) id FROM `c1_formulas`";
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getInt("id");
                } catch (NumberFormatException e) {
                    System.out.println("Error no hay id max");
                    id = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{getMaxId}\n" + "Error
:" + e.getMessage());
        }
        return id;
```

```
    }

    public boolean crear(cC1Formulas o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idvaldat`,null,
            String sql = "INSERT INTO `c1_formulas`(`idfm`,`idova`, `idesp`,
`estado_pre`,`estado_k`,`pos`) "
                    + "VALUES (null," + o.getIdova() + "," + o.getIdesp() +
"," + o.isEstadoPre() + "," + o.isEstadoK() + "," + o.getPos() + ")";
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean modificarPos(int idfm, int pos) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idvaldat`,null,
            String sql = "UPDATE `c1_formulas` SET `pos`=" + pos + " WHERE
`idfm`=" + idfm;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificarPos}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean modificarEstPre(int idfm, boolean estado) {
```

```
        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idvaldat`,null,
            String sql = "UPDATE `c1_formulas` SET `estado_pre`=" + estado +
" WHERE `idfm`=" + idfm;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificarEstPre}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean modificarEstK(int idfm, boolean estado) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idvaldat`,null,
            String sql = "UPDATE `c1_formulas` SET `estado_k`=" + estado + "
WHERE `idfm`=" + idfm;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificarEstK}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public int countEspIdova(int idova) {
        int total = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
```

```
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT COUNT(`idfm`) total FROM
`c1_formulas` WHERE `idova`=" + idova);
            if (rs.next()) {
                try {
                    total = rs.getInt("total");
                } catch (NumberFormatException e) {
                    total = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{countEspIdova}\n" +
"Error :" + e.getMessage());
        }
        return total;
    }

    public int sumPresupuestoIdova(int idova) {
        int total = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT SUM(`estado_pre`) total
FROM `c1_formulas` WHERE `idova`=" + idova);
            if (rs.next()) {
                try {
                    total = rs.getInt("total");
                } catch (NumberFormatException e) {
                    total = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase +
"]{sumPresupuestoIdova}\n" + "Error :" + e.getMessage());
        }
        return total;
    }

    public int sumPolinomioIdova(int idova) {
        int total = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
```

```
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT SUM(`estado_k`) total FROM
`c1_formulas` WHERE `idova`=" + idova);
            if (rs.next()) {
                try {
                    total = rs.getInt("total");
                } catch (NumberFormatException e) {
                    total = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{sumPolinomioIdova}\n"
+ "Error :" + e.getMessage());
        }
        return total;
    }

    public cC1Formulas buscarId(int idfm) {
        cC1Formulas p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT * FROM `c1_formulas` WHERE idfm=" + idfm;
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\n" + "Error
:" + e.getMessage());
            p = null;
        }
        p.setOe(new dEspecialidad().BuscarId("" + p.getIdesp()));
        return p;
    }

    public List<cC1Formulas> listarFormulasIdova(int idova) {
        List<cC1Formulas> lista = new ArrayList<cC1Formulas>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
```

```
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT v1.*,v2.nom FROM `c1_formulas` v1 "
                    + "INNER JOIN `especialidad` v2 ON v2.idesp=v1.idesp "
                    + "WHERE v1.idova=" + idova + " "
                    + "ORDER BY v1.pos ASC";
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult2(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{listarFormulasIdova}\n" + "Error :" + e.getMessage());
            lista = new ArrayList<cC1Formulas>();
        }
        return lista;
    }//*/

    private cC1Formulas getResult(ResultSet rs) throws SQLException {
        cC1Formulas p = new cC1Formulas();
        p.setIdfm(rs.getInt("idfm"));
        p.setIdesp(rs.getInt("idesp"));
        p.setEstadoPre(rs.getBoolean("estado_pre"));
        p.setEstadoPre(rs.getBoolean("estado_k"));
        p.setPos(rs.getInt("pos"));
        return p;
    }

    private cC1Formulas getResult2(ResultSet rs) throws SQLException {
        cC1Formulas p = new cC1Formulas();
        p.setIdfm(rs.getInt("idfm"));
        p.setIdesp(rs.getInt("idesp"));
        p.setEstadoPre(rs.getBoolean("estado_pre"));
        p.setEstadoK(rs.getBoolean("estado_k"));
        p.setPos(rs.getInt("pos"));
        p.setOe(new cEspecialidad());
        p.getOe().setNom(rs.getString("nom"));
        return p;
    }

    public List<cC1Formulas> conteoAPMPorFormulasOrdAscIdval(int idval) {
        List<cC1Formulas> lista = new ArrayList<cC1Formulas>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT v1.idfm,COUNT(*) conteo FROM `c1_formulas` v1 "
```

```
                    +    "INNER    JOIN    mes_reajuste_valorizado    v3    ON
v3.idfm=v1.idfm "
                    + "INNER JOIN reajustes v2 ON v2.idmrv=v3.idmrv "
                    + "WHERE v2.idval=" + idval + " GROUP BY v1.idfm ORDER
BY v1.idfm ASC";
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                cC1Formulas p = new cC1Formulas();
                p.setIdfm(rs.getInt("idfm"));
                p.setPos(rs.getInt("conteo"));
                lista.add(p);
            }
            rs.close();
            st.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{conteoAPMPorFormulasOrdAscIdval}\n" + "Error :" + e.getMessage());
            lista = new ArrayList<cC1Formulas>();
        }
        return lista;
    }

}
```

04) Clase dValorizacion

```
package DAO;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;import OBJETOS.cValorizacion;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.SQLException;

public class dValorizacion {

    private String nombreClase = "dValorizacion";

    public int getUltimoIdVal() {

        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT max(idval) id FROM `valorizacion`";
            System.out.println("sql:" + sql);
```

```
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getInt("id");
                } catch (NumberFormatException e) {
                    System.out.println("Error no hay id max");
                    id = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{getUltimoIdVal}\n" +
"Error :" + e.getMessage());
        }
        return id;
    }

    public int getUltimoIdValIdobra(int idObra) {

        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT max(idval) id FROM `valorizacion` WHERE
idobra=" + idObra;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getInt("id");
                } catch (NumberFormatException e) {
                    System.out.println("Error no hay id max");
                    id = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{getUltimoIdValIdobra}\n" + "Error :" + e.getMessage());
            id = 0;
        }
        return id;
    }

    public int getPrimerIdVal(int idobra) {
```

```
        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT min(idval) id FROM `valorizacion` WHERE
idobra=" + idobra + "";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                id = rs.getInt("id");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{getPrimerIdVal}\n" +
"Error :" + e.getMessage());
        }
        return id;
    }

    public int getTotalVal(int idobra) {

        String error = null;
        int id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT count(idval) nro FROM `valorizacion` WHERE
idobra=" + idobra + "";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getInt("nro");
                } catch (NumberFormatException e) {
                    System.out.println("Error no hay no hay nada");
                    id = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
```

```
            System.out.println("[" + nombreClase + "]{getTotalVal}\n" +
"Error :" + e.getMessage());
        }
        return id;
    }

    public boolean crear(cValorizacion o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "INSERT INTO `valorizacion`( `idobra`,
`nro_valorizacion`, `mes_pertenece`,
`fecha_registro`,`estado_proceso_1`,`estado_proceso_2`, `estado`) "
                    + "VALUES (" + o.getIdobra() + "," +
o.getNroValorizacion() + ",'" + o.getMesPertenece() + "','" +
fechas.formatAMD(o.getFecReg()) + "'," + o.getEstado_proceso_1() + "," +
o.getEstado_proceso_2() + "," + o.getEstado() + ")";
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean modifcarEstadoProceso1(int idval, int estado_proceso_1) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `valorizacion` SET "
                    + "`estado_proceso_1`=" + estado_proceso_1 + " "
                    + "WHERE `idval`=" + idval;
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
```

```
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{modifcarEstadoProceso1}\n" + error + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminar(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `valorizacion` WHERE `idval`=" + id;
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdObra(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `valorizacion` WHERE `idobra`=" + id;
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdObra}\n" +
error + e.getMessage());
        }
        return error == null;
    }
```

```
    public cValorizacion BuscarId(int idval) {
        cValorizacion p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from valorizacion where idval=" + idval;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{BuscarId}\n" + "Error
:" + e.getMessage());
        }
        return p;
    }

    public cValorizacion BuscarIdovaNro(int idova, int nro) {
        cValorizacion p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from valorizacion where idobra=" + idova
+ " and nro_valorizacion=" + nro;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{BuscarId}\n" + "Error
:" + e.getMessage());
            p = null;
        }
        return p;
    }

    public List<cValorizacion> ListTodoIdobraASC(int idobra) {
        List<cValorizacion> lista = new ArrayList<cValorizacion>();
        try {
```

```
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "select * from valorizacion where idobra=" + idobra
+ " order by idval asc";
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodo}\n" + "Error
:" + e.getMessage());
        }
        return lista;
    }

    public List<cValorizacion> ListTodoHastaIdvalDeIdobraASC(int idobra, int
idval) {
        List<cValorizacion> lista = new ArrayList<cValorizacion>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "select * from valorizacion where idobra=" + idobra
+ " and idval<=" + idval + " order by idval asc";
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodo}\n" + "Error
:" + e.getMessage());
        }
        return lista;
    }

    public  cValorizacion  buscarIdobraASCmenorIdvalLimit(int  idobra,  int
idval) {//buscar val anterior
        cValorizacion p = null;
        try {
            conexion dbm = new conexion();
```

```
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from valorizacion where idobra=" + idobra
+ " and idval<" + idval + " order by idval desc limit 1";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{buscarIdobraASCmenorIdvalLimit}\n" + "Error :" + e.getMessage());
        }
        return p;
    }

    public List<cValorizacion> ListTodo(int idobra, int idesp) {//=
        List<cValorizacion> lista = new ArrayList<cValorizacion>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT v.* FROM `valorizacion` v "
                    + "INNER JOIN `valorizacion_lista_partida_tipo` vlpt on
vlpt.`idval`=v.`idval` "
                    + "WHERE v.`idobra`=" + idobra + " and vlpt.`idesp`=" +
idesp + " order by v.idval desc";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodo}\n" + "Error
:" + e.getMessage());
        }
        return lista;
    }

    private cValorizacion getResult(ResultSet rs) throws SQLException {
        cValorizacion p = new cValorizacion();
        p.setIdval(rs.getInt("idval"));
        p.setIdobra(rs.getInt("idobra"));
```

```
            p.setNroValorizacion(rs.getInt("nro_valorizacion"));
            p.setMesPertenece(rs.getString("mes_pertenece"));
            p.setFecReg(rs.getDate("fecha_registro"));
            p.setEstado_proceso_1(rs.getInt("estado_proceso_1"));
            p.setEstado_proceso_2(rs.getInt("estado_proceso_2"));
            p.setEstado(rs.getBoolean("estado"));
            return p;
        }

}
```

05) Clase dValorizacionDatos

```
package DAO;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;import OBJETOS.cValorizacionDatos;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.SQLException;

public class dValorizacionDatos {

    private final String nombreClase = "dValorizacionDatos";

    public dValorizacionDatos() {
    }

    public boolean crear(cValorizacionDatos o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idvd`,null,
            String sql = "INSERT INTO
`valorizacion_datos`(`idval`,`idps`,`idfm`, `anterior_porcentaje`,
`anterior_metrado`, `anterior_valorizado`, `presente_porcentaje`,
`presente_metrado`, `presente_valorizado`, `actual_porcentaje`,
`actual_metrado`, `actual_valorizado`, `saldo_porcentaje`, `saldo_metrado`,
`saldo_valorizado`) "
                    + "VALUES (" + o.getIdval() + "," + o.getIdps() + "," +
o.getIdfm() + ","
                    + "" + o.getAnterior_porcentaje() + "," +
o.getAnterior_metrado() + "," + o.getAnterior_valorizado() + ","
                    + "" + o.getPresente_porcentaje() + "," +
o.getPresente_metrado() + "," + o.getPresente_valorizado() + ","
```

```
                    +    ""    +    o.getActual_porcentaje()    +    ","    +
o.getActual_metrado() + "," + o.getActual_valorizado() + ","
                    +    ""    +    o.getSaldo_porcentaje()    +    ","    +
o.getSaldo_metrado() + "," + o.getSaldo_valorizado() + ")";
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean crearEspecial(List<cValorizacionDatos> lsvd) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            //`idvd`,null,
            String sql = textoPlano(lsvd);
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crearEspecial}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean modificar(cValorizacionDatos o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `valorizacion_datos` SET "
```

```
                    + "`presente_porcentaje`=" + o.getPresente_porcentaje()
+       ",`presente_metrado`="       +       o.getPresente_metrado()       +
",`presente_valorizado`=" + o.getPresente_valorizado() + ","
                    + "`actual_porcentaje`=" + o.getActual_porcentaje() +
",`actual_metrado`=" + o.getActual_metrado() + ",`actual_valorizado`=" +
o.getActual_valorizado() + ","
                    + "`saldo_porcentaje`=" + o.getSaldo_porcentaje() +
",`saldo_metrado`=" + o.getSaldo_metrado() + ",`saldo_valorizado`=" +
o.getSaldo_valorizado() + " "
                    + "WHERE `idvd`=" + o.getIdvd();
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificar}\n" + error
+ e.getMessage());
        }
        return error == null;
    }

    public boolean modificarCampos(String tipoCampoQuery, String idvd) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "update valorizacion_datos set " + tipoCampoQuery +
"  where idvd=" + idvd + "";
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificarCampos}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminar(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
```

```
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `valorizacion_datos` WHERE `idvd`=" +
id;
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdVal(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `valorizacion_datos` WHERE `idval`=" +
id;
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdVal}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdObraDesdeValorizacion(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM valorizacion_datos vd INNER JOIN
valorizacion v ON v.idval=vd.idval WHERE v.idobra=" + id;
```

```
            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdObraDesdeValorizacion}\n" + error + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdObraDesdeValorizacionListaPartidasActivas(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM valorizacion_datos vd INNER JOIN valorizacion_lista_partidas_activas vlpa ON vlpa.idps=vd.idps WHERE vlpa.idobra=" + id;
//            System.out.println("" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdObraDesdeValorizacionListaPartidasActivas}\n" + error + e.getMessage());
        }
        return error == null;
    }

    public cValorizacionDatos buscarId(int id) {
        cValorizacionDatos p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from valorizacion_datos where idvd=" + id;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
```

```
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\n" + "Error
:" + e.getMessage());
            p = null;
        }
        return p;
    }

    public cValorizacionDatos buscarIdps(int idps) {
        cValorizacionDatos p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from valorizacion_datos where idps=" +
idps;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{BuscarIdps}\n" + "Error
:" + e.getMessage());
            p = null;
        }
        return p;
    }

    public cValorizacionDatos buscarIdvalIdpsIdfm(int idval, int idps, int
idfm) {
        cValorizacionDatos p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select * from valorizacion_datos where idval=" +
idval + " and idps=" + idps + " and idfm=" + idfm;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
```

```
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarIdvalIdpsIdfm}\n" + "Error :" + e.getMessage());
            p = null;
        }
        return p;
    }

    public int conteoIdvalIdobra(int idval, int idobra) {
        int conteo = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "select COUNT(vd.idvd) conteo from valorizacion_datos vd "
                    + "inner join c1_presupuesto vlpa on vlpa.idps=vd.idps "
                    + "where vd.idval=" + idval + " and vlpa.`idova`=" + idobra;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    conteo = rs.getInt("conteo");
                } catch (NumberFormatException e) {
                    conteo = 0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{conteoIdvalIdobra}\n" + "Error :" + e.getMessage());
        }
        return conteo;
    }

    public List<cValorizacionDatos> ListTodoIdvalIdovaIdfm(int idval, int idova, int idfm) {
        List<cValorizacionDatos> lista = new ArrayList<cValorizacionDatos>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
```

```
            }
            String sql = "select vd.* from valorizacion_datos vd "
                    + "inner join c1_presupuesto ps on ps.idps=vd.idps "
                    + "where vd.idval=" + idval + " and ps.`idova`=" + idova
+ " and ps.`idfm`=" + idfm + " order by vd.idvd asc";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{ListTodo}\n" + "Error
:" + e.getMessage());
            lista = new ArrayList<cValorizacionDatos>();
        }
        return lista;
    }

    public double sumaTotalValorizacionMensual(int idval, int idfm) {
        String error = null;
        double id = 0;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String  sql  =  "SELECT  SUM(vd.presente_valorizado)  suma  FROM
c1_presupuesto v1 "
                    + "INNER JOIN valorizacion_datos vd ON vd.idps=v1.idps "
                    + "WHERE vd.idval=" + idval + " and v1.idfm=" + idfm + "
AND vd.presente_metrado>0";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    id = rs.getDouble("suma");
                } catch (NumberFormatException e) {
                    System.out.println("Error con el presente_valorizado");
                    id = 0.0;
                }
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{sumaTotalValorizacionMensual}\n" + "Error :" + e.getMessage());
        }
        return id;
```

```
    }

    private cValorizacionDatos getResult(ResultSet rs) throws SQLException {
        cValorizacionDatos p = new cValorizacionDatos();
        p.setIdvd(rs.getInt("idvd"));
        p.setIdval(rs.getInt("idval"));
        p.setIdps(rs.getInt("idps"));
        p.setIdfm(rs.getInt("idfm"));

        p.setAnterior_porcentaje(rs.getDouble("anterior_porcentaje"));
        p.setAnterior_metrado(rs.getDouble("anterior_metrado"));
        p.setAnterior_valorizado(rs.getDouble("anterior_valorizado"));

        p.setPresente_porcentaje(rs.getDouble("presente_porcentaje"));
        p.setPresente_metrado(rs.getDouble("presente_metrado"));
        p.setPresente_valorizado(rs.getDouble("presente_valorizado"));

        p.setActual_porcentaje(rs.getDouble("actual_porcentaje"));
        p.setActual_metrado(rs.getDouble("actual_metrado"));
        p.setActual_valorizado(rs.getDouble("actual_valorizado"));

        p.setSaldo_porcentaje(rs.getDouble("saldo_porcentaje"));
        p.setSaldo_metrado(rs.getDouble("saldo_metrado"));
        p.setSaldo_valorizado(rs.getDouble("saldo_valorizado"));
        return p;
    }

    public String textoPlano(List<cValorizacionDatos> ls) {
        String sql = "INSERT INTO `valorizacion_datos`(`idval`,`idps`,`idfm`,
`anterior_porcentaje`,      `anterior_metrado`,      `anterior_valorizado`,
`presente_porcentaje`,      `presente_metrado`,      `presente_valorizado`,
`actual_porcentaje`,        `actual_metrado`,        `actual_valorizado`,
`saldo_porcentaje`, `saldo_metrado`, `saldo_valorizado`) ";
        sql += "VALUES ";
        int i = 0;
        for (cValorizacionDatos ovd : ls) {
            if (i < ls.size() - 1) {
                sql += ovd.getTextoPlano();
            } else {
                sql += ovd.getTextoPlano() + ",";
            }
            i++;
        }
        sql += ";";
        return sql;
    }
}
```

06) Clase dReajuste

```
package DAO;

import OBJETOS.cReajustes;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.Connection;
```

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.sql.SQLException;

public class dReajustes {

    private final String nombreClase = "dReajustes";

    public boolean crear(cReajustes o) {
        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "INSERT INTO `reajustes`(`idmrv`, `idval`, `nro`,
`mes`, `apm`, `apa`, `aem`, `aea`, `coe_reajuste`, `k_1`, `rpm`, `rpa`, `rem`,
`rea`, `estado_obra`, `rrm`, `rra`, `valorizacion_neta`,`mes_activo`) "
                    + "VALUES (" + o.getIdmrv() + "," + o.getIdval() + "," +
o.getNro() + ",'" + o.getMes() + "',"
                    + "" + o.getApm() + "," + o.getApa() + "," + o.getAem()
+ "," + o.getAea() + "," + o.getCoeRea() + ","
                    + "" + o.getK_1() + "," + o.getRpm() + "," + o.getRpa()
+ "," + o.getRem() + "," + o.getRea() + ","
                    + "'" + o.getEstObra() + "'," + o.getRrm() + "," +
o.getRra() + "," + o.getValNeta() + "," + o.getMesActivo() + ")";
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean modificar(cReajustes o) {
        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `reajustes` SET `nro`=" + o.getNro() +
",`mes`='" + o.getMes() + "',`apm`=" + o.getApm() + ","
                    + "`apa`=" + o.getApa() + ",`aem`=" + o.getAem() +
",`aea`=" + o.getAea() + ",`coe_reajuste`=" + o.getCoeRea() + ","
```

```
                    + "`k_1`=" + o.getK_1() + ",`rpm`=" + o.getRpm() +
",`rpa`=" + o.getRpa() + ",`rem`=" + o.getRem() + ","
                    + "`rea`=" + o.getRea() + ",`estado_obra`='" +
o.getEstObra() + "',`rrm`=" + o.getRrm() + ","
                    + "`rra`=" + o.getRra() + ",`valorizacion_neta`=" +
o.getValNeta() + ",`mes_activo`=" + o.getMesActivo() + " WHERE `idrea`=" +
o.getIdrea();

            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{modificar}\n" + error
+ e.getMessage());
        }
        return error == null;
    }

    public boolean modificarApmApa(int idrea, double apm, double apa) {
        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `reajustes` SET `apm`=" + apm + ",`apa`=" +
apa + " WHERE `idrea`=" + idrea;

            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{modificarApmApa}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminar(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
```

```
            }
            String sql = "DELETE FROM `reajustes` WHERE `idrea`=" + id;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + error +
e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdova(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `reajustes` WHERE `idova`=" + id;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdova}\n" +
error + e.getMessage());
        }
        return error == null;
    }

    public cReajustes buscarId(String id) {
        cReajustes p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from reajustes where
idrea=" + id);
            if (rs.next()) {
                p = getResult(rs);

            }
            rs.close();
            st.close();
```

```
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\nError :" +
e.getMessage());
        }
        return p;
    }

    public cReajustes buscarIdvalOrigenNro(int idfm, int idval_origen, int
nro_valorizacion) {
        cReajustes p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "select * from reajustes v1 "
                    +   "inner   join   mes_reajuste_valorizado   v2   on
v2.idmrv=v1.idmrv  "
                    + "where v2.idfm=" + idfm + " and v1.idval=" +
idval_origen + " and v1.nro=" + nro_valorizacion;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);

            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\nError :" +
e.getMessage());
        }
        return p;
    }

    public cReajustes buscarIdfmIdvaIdvalOrigenlNro(int idfm, int idval, int
idval_origen, int nro_valorizacion) {
        cReajustes p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "select v1.* from reajustes v1 "
                    +   "inner   join   mes_reajuste_valorizado   v2   on
v2.idmrv=v1.idmrv  "
                    + "where v2.idfm=" + idfm + " and v2.idval=" + idval + "
and  v2.idval_origen="  +  idval_origen  +  "  and  v2.nro_valorizacion="  +
nro_valorizacion;
```

```
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);

            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\nError :" +
e.getMessage());
        }
        return p;
    }

    public int totalEstadoAtrasado(int idval, int idfm) {
        int t = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT COUNT(r.idrea) t FROM `reajustes` r "
                    +   "INNER   JOIN   mes_reajuste_valorizado   v1   ON
v1.idmrv=r.idmrv "
                    + "WHERE v1.idfm=" + idfm + " AND r.idval<" + idval + "
AND r.`estado_obra`='ATRASADA' ";
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    t = rs.getInt("t");
                } catch (NumberFormatException e) {
                    System.out.println("NO HAY ATRAZADOS" + e.getMessage());
                    t = 0;
                }

            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{totalEstadoAtrasado}\nError :" + e.getMessage());
        }
        return t;
    }

    public double sumaRemAnteriores(int idval, int idfm) {
        double t = 0.0;
        try {
            conexion dbm = new conexion();
```

```
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT SUM(r.rem) t FROM `reajustes` r "
                    +   "INNER   JOIN   mes_reajuste_valorizado   v1   ON
v1.idmrv=r.idmrv "
                    + "WHERE v1.idfm=" + idfm + " AND r.idval<=" + idval;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    t = rs.getDouble("t");
                } catch (NumberFormatException e) {
                    t = 0.0;
                }

            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["         +         nombreClase         +
"]{sumaRemAnteriores}\nError:" + e.getMessage());
        }
        return t;
    }

    public double montoRRMIdValNro(int idval, int nro) {
        double t = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT SUM(v1.rrm) rrm FROM `reajustes` v1 "
                    +   "INNER   JOIN   `mes_reajuste_valorizado`   v2   ON
v2.idmrv=v1.idmrv "
                    + "WHERE v2.idval=" + idval + " AND v1.nro=" + nro;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                t = rs.getDouble("rrm");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("["         +         nombreClase         +
"]{montoRRMIdValNro}\nError :" + e.getMessage());
        }
        return t;
```

```
    }

    public double montoRRAIdValNro(int idval, int nro) {
        double t = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT SUM(v1.rra) rra FROM `reajustes` v1 "
                    +   "INNER   JOIN   `mes_reajuste_valorizado`   v2   ON
v2.idmrv=v1.idmrv "
                    + "WHERE v2.idval=" + idval + " AND v1.nro=" + nro;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                t = rs.getDouble("rra");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("["          +          nombreClase          +
"]{montoRRAIdValNro}\nError :" + e.getMessage());
        }
        return t;
    }

    public List<cReajustes> listarIdValIdfmNro(int idval, int idesp, int nro)
{
        List<cReajustes> lista = new ArrayList<cReajustes>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT * FROM `reajustes` v1 "
                    +   "INNER   JOIN   `mes_reajuste_valorizado`   v2   ON
v2.idmrv=v1.idmrv "
                    + "WHERE v2.idval=" + idval + " AND v2.idesp=" + idesp +
" AND v1.nro=" + nro);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (Exception e) {
            System.out.println("["          +          nombreClase          +
"]{buscarIdValIdEspNro}\nError :" + e.getMessage());
        }
```

```
        return lista;
    }

    public List<cReajustes> listTodoIdValIdfm(int idval, int idfm) {
        List<cReajustes> lista = new ArrayList<cReajustes>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT v1.* FROM `reajustes` v1 "
                    +   "INNER   JOIN   `mes_reajuste_valorizado`   v2   ON
v2.idmrv=v1.idmrv "
                    + "WHERE v2.idval=" + idval + " AND v2.idfm=" + idfm + "
"
                    + "order by v1.nro asc";
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{listTodoIdValIdfm}\nError :" + e.getMessage());
            lista = new ArrayList<cReajustes>();
        }
        return lista;
    }

    public List<cReajustes> sumaApmAemPorIdval(int idval) {
        List<cReajustes> lista = new ArrayList<cReajustes>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet   rs   =   st.executeQuery("SELECT   r.nro,sum(r.apm)
montoapm,sum(r.aem) montoaem FROM `reajustes` r "
                    +   "INNER   JOIN   `mes_reajuste_valorizado`   v   ON
v.idmrv=r.idmrv "
                    + "WHERE v.idval=" + idval + " GROUP BY r.nro ORDER BY
r.nro ASC");
            while (rs.next()) {
                cReajustes p = new cReajustes();
                p.setNro(rs.getInt("nro"));
                p.setApm(rs.getDouble("montoapm"));
                p.setAem(rs.getDouble("montoaem"));
                lista.add(p);
            }
```

```
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{sumaApmAemPorIdval}\nError :" + e.getMessage());
        }
        return lista;
    }

    private cReajustes getResult(ResultSet rs) throws SQLException {
        cReajustes p = new cReajustes();
        p.setIdrea(rs.getInt("idrea"));
        p.setIdmrv(rs.getInt("idmrv"));
        p.setIdval(rs.getInt("idval"));
        p.setNro(rs.getInt("nro"));
        p.setMes(rs.getString("mes"));

        p.setApm(rs.getDouble("apm"));
        p.setApa(rs.getDouble("apa"));
        p.setAem(rs.getDouble("aem"));
        p.setAea(rs.getDouble("aea"));
        p.setCoeRea(rs.getDouble("coe_reajuste"));

        p.setK_1(rs.getDouble("k_1"));
        p.setRpm(rs.getDouble("rpm"));
        p.setRpa(rs.getDouble("rpa"));
        p.setRem(rs.getDouble("rem"));
        p.setRea(rs.getDouble("rea"));

        p.setEstObra(rs.getString("estado_obra"));
        p.setRrm(rs.getDouble("rrm"));
        p.setRra(rs.getDouble("rra"));
        p.setValNeta(rs.getDouble("valorizacion_neta"));
        p.setMesActivo(rs.getInt("mes_activo"));
        return p;
    }

}
```

07) Clase dDeduccionAdelantoDirecto

```
package DAO;

import OBJETOS.cDeduccionAdelantoDirecto;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
```

```
public class dDeduccionAdelantoDirecto {

    private final String nombreClase = "dDeduccionAdelantoDirecto";

    public boolean crear(cDeduccionAdelantoDirecto o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "INSERT INTO `deduccion_adelanto_directo`(`idmcr`,
`idval`,  `nro_valorizacion`,  `mes_anio`,  `monto_metrado`,  `adelanto`,
`monto_contrato`, `kr`, `ka`, `deduccion`, `mes_anio_deducido`, `mes_activo`)
"
                    + "VALUES (" + o.getIdmcr() + "," + o.getIdval() + "," +
o.getNroValorizacion() + ","
                    + "'" + o.getMesAnio() + "'," + o.getMonMet() + "," +
o.getAde() + "," + o.getMonCon() + ","
                    + "" + o.getKr() + "," + o.getKa() + "," + o.getDed() +
",'" + o.getMesADed() + "'," + o.getMesActivo() + ")";
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + "Error :"
+ e.getMessage());
        }
        return error == null;
    }

    public boolean modificar(cDeduccionAdelantoDirecto o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `deduccion_adelanto_directo` SET "
                    + "`monto_metrado`=" + o.getMonMet() + ","
                    + "`adelanto`=" + o.getAde() + ","
                    + "`monto_contrato`=" + o.getMonCon() + ","
                    + "`kr`=" + o.getKr() + ","
                    + "`ka`=" + o.getKa() + ","
                    + "`deduccion`=" + o.getDed() + ","
                    + "`mes_anio_deducido`='" + o.getMesADed() + "',"
                    + "`mes_activo`=" + o.getMesActivo() + " "
                    + "WHERE `iddad`=" + o.getIddad();
            System.out.println("sql:" + sql);
```

```
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificar}\n" + "Error
:" + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminar(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `deduccion_adelanto_directo` WHERE
`iddad`=" + id;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + "Error
:" + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdova(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `deduccion_adelanto_directo` WHERE
`idova`=" + id;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
```

```
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdova}\n" +
"Error :" + e.getMessage());
        }
        return error == null;
    }

    public cDeduccionAdelantoDirecto buscarId(String id) {
        cDeduccionAdelantoDirecto p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet     rs     =     st.executeQuery("SELECT     *     FROM
`deduccion_adelanto_directo` WHERE `iddad`=" + id);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\n" + "Error
:" + e.getMessage());
        }
        return p;
    }

    public cDeduccionAdelantoDirecto buscarIdvalIdfmNro(int idval, int idfm,
int nro) {
        cDeduccionAdelantoDirecto p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet    rs    =    st.executeQuery("SELECT    v1.*    FROM
`deduccion_adelanto_directo` v1 "
                    +    "INNER    JOIN    mes_coeficiente_reajuste    v2    ON
v2.idmcr=v1.idmcr "
                    + "WHERE v2.idval=" + idval + " AND v2.nro_valorizacion="
+ nro + " AND v2.idfm=" + idfm);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
```

```
            System.out.println("[" + nombreClase + "]{buscarIdvalIdfmNro}\n"
+ "Error :" + e.getMessage());
        }
        return p;
    }

    public double montoDeduccionAcumuladoIdVal(int idval) {
        double t = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String  sql  =  "SELECT  SUM(v1.deduccion)  sum_deduccion  FROM
`deduccion_adelanto_directo` v1 "
                    +  "INNER  JOIN  `mes_coeficiente_reajuste`  v2  ON
v2.idmcr=v1.idmcr "
                    + "WHERE v2.idval=" + idval;
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                t = rs.getDouble("sum_deduccion");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{montoDeduccionAcumuladoIdVal}\n" + "Error :" + e.getMessage());
        }
        return t;
    }

    public List<cDeduccionAdelantoDirecto> listTodoIdObra(int idObra) {
        List<cDeduccionAdelantoDirecto>          lista          =          new
ArrayList<cDeduccionAdelantoDirecto>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet     rs     =     st.executeQuery("select     *     from
deduccion_adelanto_directo where idova=" + idObra + " and estado=1");
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdObra}\n" +
"Error :" + e.getMessage());
```

```
        }
        return lista;
    }

    public List<cDeduccionAdelantoDirecto> listTodoIdvalIdfm(int idval, int
idfm) {
        List<cDeduccionAdelantoDirecto>         lista         =         new
ArrayList<cDeduccionAdelantoDirecto>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT v1.* FROM `deduccion_adelanto_directo` v1 "
                    +   "INNER   JOIN   `mes_coeficiente_reajuste`   v2   ON
v2.idmcr=v1.idmcr "
                    + "WHERE v2.idval=" + idval + " AND v2.idfm=" + idfm + "
order by v2.nro_valorizacion asc";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdvalIdfm}\n"
+ "Error :" + e.getMessage());
        }
        return lista;
    }

    private   cDeduccionAdelantoDirecto   getResult(ResultSet   rs)   throws
SQLException {
        cDeduccionAdelantoDirecto p = new cDeduccionAdelantoDirecto();
        p.setIddad(rs.getInt("iddad"));
        p.setIdmcr(rs.getInt("idmcr"));
        p.setIdval(rs.getInt("idval"));
        p.setNroValorizacion(rs.getInt("nro_valorizacion"));
        p.setMesAnio(rs.getString("mes_anio"));
        p.setMonMet(rs.getDouble("monto_metrado"));
        p.setAde(rs.getDouble("adelanto"));
        p.setMonCon(rs.getDouble("monto_contrato"));
        p.setKr(rs.getDouble("kr"));
        p.setKa(rs.getDouble("ka"));
        p.setDed(rs.getDouble("deduccion"));
        p.setMesADed(rs.getString("mes_anio_deducido"));
        p.setMesActivo(rs.getInt("mes_activo"));
        return p;
    }

    public cDeduccionAdelantoDirecto _mInicializarClase() {
```

```
        return new cDeduccionAdelantoDirecto(0, 0, 0, 0, "", 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, "", 0);
    }

}
```

08) Clase dDeduccionAdelantoMateriales

```
package DAO;

import OBJETOS.cAdelantoDirecto;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import OBJETOS.cDeduccionAdelantoMateriales;

public class dDeduccionAdelantoMateriales {

    private final String nombreClase = "dDeduccionAdelantoMateriales";

    public boolean crear(cDeduccionAdelantoMateriales o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String           sql           =           "INSERT           INTO
`deduccion_adelanto_materiales`(`idmim`,          `idval`,          `idpsm`,
`nro_valorizacion`,  `mes_anio`,  `monto_metrado`,  `coeficiente_incidencia`,
`indice_mes_base`, `indice_mes_adelanto`, `indice_mes_reajuste`, `adelanto`,
`monto_maximo_utilizado`,       `adelanto_deducible`,       `saldo_adelanto`,
`deduccion`, `mes_anio_pago`, `mes_activo`) "
                    + "VALUES (" + o.getIdmim() + "," + o.getIdval() + "," +
o.getIdpsm() + "," + o.getNroVal() + ",'" + o.getMesAnio() + "',"
                    + o.getMonMet() + "," + o.getCoeInc() + "," + o.getImo()
+ "," + o.getIma() + "," + o.getImr() + ","
                    + o.getA() + "," + o.getMmu() + "," + o.getAdeDed() + ","
+  o.getSalAde()  +  ","  +  o.getDed()  +  ",'"  +  o.getMesPag()  +  "',"  +
o.getMesActivo() + ")";
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
```

```
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + "Error :"
+ e.getMessage());
        }
        return error == null;
    }

    public boolean modificar(cDeduccionAdelantoMateriales o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `deduccion_adelanto_materiales` SET "
                    + "`monto_metrado`=" + o.getMonMet() + ","
                    + "`coeficiente_incidencia`=" + o.getCoeInc() + ","
                    + "`indice_mes_base`=" + o.getImo() + ","
                    + "`indice_mes_adelanto`=" + o.getIma() + ","
                    + "`indice_mes_reajuste`=" + o.getImr() + ","
                    + "`adelanto`=" + o.getA() + ","
                    + "`monto_maximo_utilizado`=" + o.getMmu() + ","
                    + "`adelanto_deducible`=" + o.getAdeDed() + ","
                    + "`saldo_adelanto`=" + o.getSalAde() + ","
                    + "`deduccion`=" + o.getDed() + ","
                    + "`mes_anio_pago`='" + o.getMesPag() + "',"
                    + "`mes_activo`=" + o.getMesActivo() + " "
                    + "WHERE `iddam`=" + o.getIddam();
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificar}\n" + "Error
:" + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminar(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `deduccion_adelanto_materiales` WHERE
`iddam`=" + id;
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
```

```
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + "Error
:" + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdObra(int idObra) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM deduccion_adelanto_materiales "
                    + "WHERE idk IN(SELECT vk.idk FROM valor_k vk INNER JOIN
valor_especialidad ve ON ve.idvesp=vk.idvesp WHERE ve.idobra=" + idObra +
")";
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdObra}\n" +
"Error :" + e.getMessage());
        }
        return error == null;
    }

    public cDeduccionAdelantoMateriales buscarId(String id) {
        cDeduccionAdelantoMateriales p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet     rs     =     st.executeQuery("SELECT     *     FROM
`deduccion_adelanto_materiales` WHERE `idkm`=" + id);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
```

```
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\n" + "Error
:" + e.getMessage());
        }
        return p;
    }

    public cDeduccionAdelantoMateriales buscarIdvalIdespIdvadNro(int idval,
int idesp, int idvad, int nro) {
        cDeduccionAdelantoMateriales p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet    rs    =    st.executeQuery("SELECT    v1.*    FROM
`deduccion_adelanto_materiales` v1 "
                    +    "INNER    JOIN    mes_indice_materiales    v2    ON
v2.idmim=v1.idmim "
                    + "WHERE v1.idval=" + idval + " AND v1.nro_valorizacion="
+ nro + " AND v2.idesp=" + idesp + " AND v1.idvad=" + idvad);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{buscarIdvalIdespIdvadNro}\n" + "Error :" + e.getMessage());
        }
        return p;
    }

    public double montoDeduccionAcumuladoIdVal(int idval) {
        double t = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String  sql  =  "SELECT  SUM(v1.deduccion)  sum_deduccion  FROM
`deduccion_adelanto_materiales` v1 "
                    +    "INNER    JOIN    `mes_indice_materiales`    v2    ON
v2.idmim=v1.idmim "
                    + "WHERE v2.idval=" + idval;
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                t = rs.getDouble("sum_deduccion");
            }
            rs.close();
```

```
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{montoDeduccionAcumuladoIdVal}\n" + "Error :" + e.getMessage());
        }
        return t;
    }

    public int CountRegistros(int idvesp) {
        int pos = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("SELECT count(idkm) contar FROM `valor_k_material` WHERE `idvesp`=" + idvesp + " AND LENGTH(`simbolo`)>0");
            if (rs.next()) {
                pos = rs.getInt("contar");
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{CountRegistros}\n" + "Error :" + e.getMessage());
        }
        return pos;
    }

    public List<cDeduccionAdelantoMateriales> listTodo() {
        List<cDeduccionAdelantoMateriales> lista = new ArrayList<cDeduccionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery("select * from valor_k_material where estado=1");
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodo}\n" + "Error :" + e.getMessage());
```

```
        }
        return lista;
    }

    public List<cDeduccionAdelantoMateriales> listTodoIdvesp(int idvesp) {
        List<cDeduccionAdelantoMateriales>        lista        =        new
ArrayList<cDeduccionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "select * from valor_k_material where idvesp=" +
idvesp + " order by correlativo asc";
            System.out.println("" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdvesp}\n" +
"Error :" + e.getMessage());
        }
        return lista;
    }

    public     List<cDeduccionAdelantoMateriales>     listTodoIdObraIdEsp(int
idobra, int idesp) {
        List<cDeduccionAdelantoMateriales>        lista        =        new
ArrayList<cDeduccionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT vkm.* FROM `valor_k_material` vkm "
                    +    "INNER    JOIN    `valor_especialidad`    ve    on
ve.idvesp=vkm.idvesp "
                    + "WHERE ve.idobra=" + idobra + " and ve.idesp=" + idesp
+ " order by correlativo asc";
            System.out.println("" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
```

```
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdObraIdEsp}\n" + "Error :" + e.getMessage());
        }
        return lista;
    }

    public List<cDeduccionAdelantoMateriales> listTodo(int idVal) {//=por observar
        List<cDeduccionAdelantoMateriales> lista = new ArrayList<cDeduccionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "select e.* from especialidad e "
                    + "inner join mes_indice_materiales v on v.idesp=e.idesp "
                    + "where v.idval=" + idVal;
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodo}\n" + "Error :" + e.getMessage());
        }
        return lista;
    }

    public List<cDeduccionAdelantoMateriales> listTodoIdvalIdfm(int idval, int idfm, int idpsm) {
        List<cDeduccionAdelantoMateriales> lista = new ArrayList<cDeduccionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT v1.* FROM `deduccion_adelanto_materiales` v1 "
                    + "INNER JOIN mes_indice_materiales v2 ON v2.idmim=v1.idmim "
                    + "WHERE v2.idval=" + idval + " AND v2.idfm=" + idfm + " AND v1.idpsm=" + idpsm;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
```

```
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdvalIdfm}\n"
+ "Error :" + e.getMessage());
        }
        return lista;
    }

    private cDeduccionAdelantoMateriales getResult(ResultSet rs) throws
SQLException {
        cDeduccionAdelantoMateriales p = new cDeduccionAdelantoMateriales();
        p.setIddam(rs.getInt("iddam"));
        p.setIdmim(rs.getInt("idmim"));
        p.setIdval(rs.getInt("idval"));
        p.setIdpsm(rs.getInt("idpsm"));
        p.setNroVal(rs.getInt("nro_valorizacion"));

        p.setMesAnio(rs.getString("mes_anio"));
        p.setMonMet(rs.getDouble("monto_metrado"));
        p.setCoeInc(rs.getDouble("coeficiente_incidencia"));
        p.setImo(rs.getDouble("indice_mes_base"));
        p.setIma(rs.getDouble("indice_mes_adelanto"));

        p.setImr(rs.getDouble("indice_mes_reajuste"));
        p.setA(rs.getDouble("adelanto"));
        p.setMmu(rs.getDouble("monto_maximo_utilizado"));
        p.setAdeDed(rs.getDouble("adelanto_deducible"));
        p.setSalAde(rs.getDouble("saldo_adelanto"));

        p.setDed(rs.getDouble("deduccion"));
        p.setMesPag(rs.getString("mes_anio_pago"));
        p.setMesActivo(rs.getInt("mes_activo"));
        return p;
    }
}
```

09) Clase dAmortizacionAdelantoMateriales

```
package DAO;

import OBJETOS.cAmortizacionAdelantoMateriales;
import utilitarios.fechas;
import vconexion.conexion;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
```

```
public class dAmortizacionAdelantoMateriales {

    private final String nombreClase = "dAmortizacionAdelantoMateriales";

    public boolean crear(cAmortizacionAdelantoMateriales o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "INSERT INTO
`amortizacion_adelanto_materiales`(`idfm`, `idval`, `idpsm`,
`nro_valorizacion`, `mes_anio`, `monto_metrado`, `coeficiente_incidencia`,
`indice_mes_base`, `indice_mes_adelanto`, `indice_mes_reajuste`, `mv_ci`,
`amortizacion_calculada`, `adelanto`, `amortizacion_real`,
`saldo_amortizar`, `mes_activo`) "
                    + "VALUES (" + o.getIdfm() + "," + o.getIdval() + "," +
o.getIdpsm() + "," + o.getNroVal() + ",'" + o.getMesAnio() + "'," +
o.getMonMet() + "," + o.getCoeInc() + "," + o.getImo() + "," + o.getIma() +
"," + o.getImr() + "," + o.getMvCi() + "," + o.getAmoCal() + "," + o.getAde()
+ "," + o.getAmoReal() + "," + o.getSalAmor() + "," + o.getMes_activo() +
")";
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{crear}\n" + "Error :"
+ e.getMessage());
        }
        return error == null;
    }

    public boolean modificar(cAmortizacionAdelantoMateriales o) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "UPDATE `amortizacion_adelanto_materiales` SET "
                    + "`monto_metrado`=" + o.getMonMet() + ","
                    + "`coeficiente_incidencia`=" + o.getCoeInc() + ","
                    + "`indice_mes_base`=" + o.getImo() + ","
                    + "`indice_mes_adelanto`=" + o.getIma() + ","
                    + "`indice_mes_reajuste`=" + o.getImr() + ","
                    + "`mv_ci`=" + o.getMvCi() + ","
                    + "`amortizacion_calculada`=" + o.getAmoCal() + ","
```

```
                    + "`adelanto`=" + o.getAde() + ","
                    + "`amortizacion_real`=" + o.getAmoReal() + ","
                    + "`saldo_amortizar`=" + o.getSalAmor() + ", "
                    + "`mes_activo`=" + o.getMes_activo() + " "
                    + "WHERE `idaam`=" + o.getIdaam();
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{modificar}\n" + "Error
:" + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminar(int id) {
        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `amortizacion_adelanto_materiales`
WHERE `idaam`=" + id;
            System.out.println("sql:" + sql);
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminar}\n" + "Error
:" + e.getMessage());
        }
        return error == null;
    }

    public boolean eliminarIdova(int id) {

        String error = null;
        conexion dbm = new conexion();
        try {
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "DELETE FROM `amortizacion_adelanto_materiales`
WHERE `idaam`=" + id;
            System.out.println("sql:" + sql);
```

```
            PreparedStatement st = con.prepareStatement(sql, 1005, 1007);
            if (st.execute()) {
                error = "No se puede actualizar datos de la empresa: ";
            }
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{eliminarIdova}\n" +
"Error :" + e.getMessage());
        }
        return error == null;
    }

    public cAmortizacionAdelantoMateriales buscarId(String id) {
        cAmortizacionAdelantoMateriales p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT * FROM `amortizacion_adelanto_materiales`
WHERE `idaam`=" + id;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{buscarId}\n" + "Error
:" + e.getMessage());
        }
        return p;
    }

    public   cAmortizacionAdelantoMateriales   buscarIdvalIdespIdvadNro(int
idval, int idfm, int idpsm, int nro) {
        cAmortizacionAdelantoMateriales p = null;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            String sql = "SELECT FROM `amortizacion_adelanto_materiales` "
                    + "WHERE idval=" + idval + " AND nro_valorizacion=" + nro
+ " AND idfm=" + idfm + " AND idpsm=" + idpsm;
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
```

```
                p = getResult(rs);
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase +
"]{buscarIdvalIdespIdvadNro}\n" + "Error :" + e.getMessage());
        }
        return p;
    }

    public double montoAmortizadoAcumuladoIdVal(int idval) {
        double t = 0;
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT SUM(amortizacion_real) suma FROM
`amortizacion_adelanto_materiales` "
                    + "WHERE idval=" + idval;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            if (rs.next()) {
                try {
                    t = rs.getDouble("suma");
                } catch (NumberFormatException e) {
                    t = 0.0;
                }

            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase +
"]{montoAmortizadoAcumuladoIdVal}\n" + "Error :" + e.getMessage());
        }
        return t;
    }

    public List<cAmortizacionAdelantoMateriales> listTodoIdObra(int idObra)
{
        List<cAmortizacionAdelantoMateriales> lista = new
ArrayList<cAmortizacionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
```

```
            String sql = "select * from amortizacion_adelanto_materiales
where idova=" + idObra + " and estado=1";
            System.out.println("sql:" + sql);
            Statement st = con.createStatement();
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdObra}\n" +
"Error :" + e.getMessage());
        }
        return lista;
    }

    public List<cAmortizacionAdelantoMateriales> listTodoIdvalIdespIdvad(int
idval, int idfm, int idpsm) {
        List<cAmortizacionAdelantoMateriales>       lista       =       new
ArrayList<cAmortizacionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT * FROM `amortizacion_adelanto_materiales` "
                    + "WHERE idval=" + idval + " AND idfm=" + idfm + " AND
idpsm=" + idpsm;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("["          +          nombreClase          +
"]{listTodoIdvalIdespIdvad}\n" + "Error :" + e.getMessage());
        }
        return lista;
    }

    public    List<cAmortizacionAdelantoMateriales>    listTodoIdvalIdvad(int
idval, int idpsm) {
        List<cAmortizacionAdelantoMateriales>       lista       =       new
ArrayList<cAmortizacionAdelantoMateriales>();
        try {
            conexion dbm = new conexion();
            Connection con = dbm.getConnection();
            if (con == null) {
```

```
                throw new NullPointerException(dbm.getError());
            }
            Statement st = con.createStatement();
            String sql = "SELECT * FROM `amortizacion_adelanto_materiales` "
                    + "WHERE idval=" + idval;
            System.out.println("sql:" + sql);
            ResultSet rs = st.executeQuery(sql);
            while (rs.next()) {
                lista.add(getResult(rs));
            }
            rs.close();
            st.close();
            con.close();
            dbm = null;
        } catch (NullPointerException | SQLException e) {
            System.out.println("[" + nombreClase + "]{listTodoIdvalIdvad}\n"
+ "Error :" + e.getMessage());
        }
        return lista;
    }

    private cAmortizacionAdelantoMateriales getResult(ResultSet rs) throws
SQLException {
        cAmortizacionAdelantoMateriales p = new
cAmortizacionAdelantoMateriales();
        p.setIdaam(rs.getInt("idaam"));
        p.setIdfm(rs.getInt("idfm"));
        p.setIdval(rs.getInt("idval"));
        p.setIdpsm(rs.getInt("idpsm"));
        p.setNroVal(rs.getInt("nro_valorizacion"));
        p.setMesAnio(rs.getString("mes_anio"));
        p.setMonMet(rs.getDouble("monto_metrado"));
        p.setCoeInc(rs.getDouble("coeficiente_incidencia"));
        p.setImo(rs.getDouble("indice_mes_base"));
        p.setIma(rs.getDouble("indice_mes_adelanto"));
        p.setImr(rs.getDouble("indice_mes_reajuste"));
        p.setMvCi(rs.getDouble("mv_ci"));
        p.setAmoCal(rs.getDouble("amortizacion_calculada"));
        p.setAde(rs.getDouble("adelanto"));
        p.setAmoReal(rs.getDouble("amortizacion_real"));
        p.setSalAmor(rs.getDouble("saldo_amortizar"));
        p.setMes_activo(rs.getInt("mes_activo"));
        return p;
    }

}
```

4.3.2. PRUEBAS

EL Rational Unified Process propone que las pruebas se dimensionen en tres niveles: Calidad, Estado y Pruebas de Tipo. Para la presente investigación se optó por realizar algunas de estas pruebas, siendo las siguientes:

A) Nivel de calidad

A.1 Operatividad

La operatividad tiene por objeto demostrar que las funcionalidades del sistema operan en forma correcta e independiente una de otra, estas pruebas podrán ser preparas y ejecutadas por el desarrollador.

Los posibles resultados de las pruebas son:

- Éxito: El resultado de la prueba es conforme al resultado esperado.
- Aceptable: El resultado de la prueba indica que el sistema difiere con los resultados esperados, se tendrá que analizar si se requiere cambios en la aplicación o se requiere hacer cambios en la especificación funcional.
- Error: El resultado de la prueba es incorrecto; la falla deberá ser corregida en un tiempo determinado.

A continuación, se presenta un caso de prueba cuyo objetivo es comprobar el funcionamiento del registro de los datos generales de una obra creada.

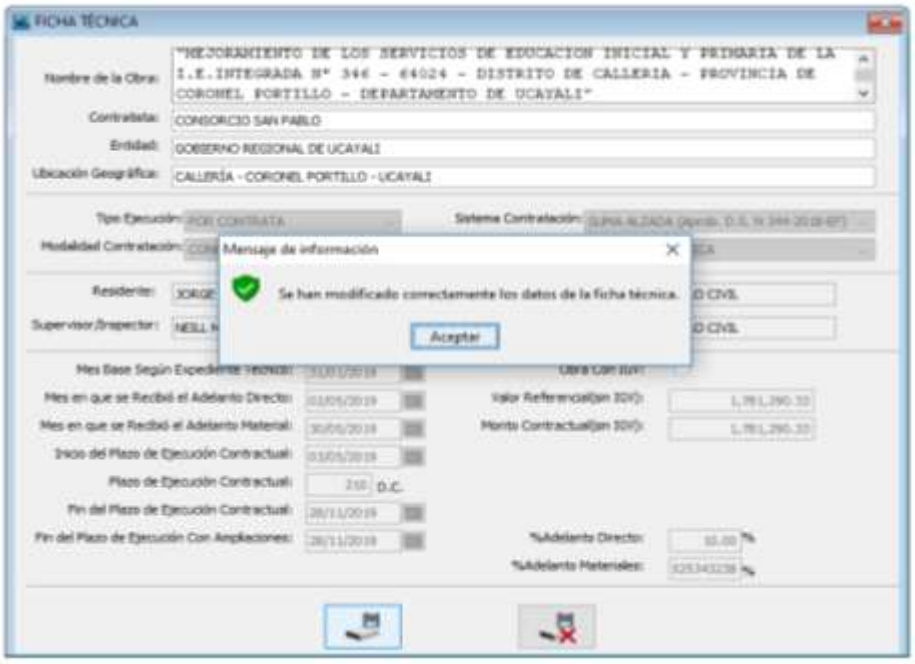


Imagen 4.11: Caso de prueba del registro de los datos generales de una obra creada, con resultado de éxito.

Por otra parte, se procede a mostrar la prueba de operatividad de los cuatro (04) módulos.

VALORIZACIÓN DE OBRA PRINCIPAL

DESCRIPCIÓN DE PRUEBA	ESTADO
Ingreso de datos a la ficha técnica	**Aprobado**
Modificación del pie de presupuesto	**Aprobado**
Cargar Presupuestos de la obra (Excel)	**Aprobado**
Cargar Fórmulas Polínomicas de la obra (Excel)	**Aprobado**
Definición de criterio para el uso del ”Kr”	**Aprobado**
Cálculo el monto de adelanto de materiales	**Aprobado**
Cálculo de la valorización de la obra principal	**Aprobado**
Eliminación de la valorización	**Aprobado**

VALORIZACIÓN DE OBRA ADICIONAL

DESCRIPCIÓN DE PRUEBA	ESTADO
Registro de fechas en la ficha técnica de la obra adicional	**Aprobado**
Modificación del pie de presupuesto de la obra adicional	**Aprobado**
Cargar Presupuestos de la obra adicional (Excel)	**Aprobado**
Cargar Fórmulas Polínomicas de la obra adicional (Excel)	**Aprobado**
Definición de criterio para el uso del ”Kr”	**Aprobado**
Cálculo de la valorización de la obra adicional	**Aprobado**
Eliminación de la valorización de obra adicional	**Aprobado**

VALORIZACIÓN DE MAYORES GASTOS GENERALES VARIABLES

DESCRIPCIÓN DE PRUEBA	ESTADO
Registrar ampliaciones de plazo aprobadas	**Aprobado**
Registrar datos generales de las ampliaciones de plazos	**Aprobado**
Cálculo de la valorización de mayores gastos generales variables	**Aprobado**
Eliminación de las ampliaciones de plazo	**Aprobado**

VALORIZACIÓN DE INTERESES POR DEMORA DE PAGOS

DESCRIPCIÓN DE PRUEBA	ESTADO
Registrar fecha real de pago de las valorizaciones	**Aprobado**
Registrar factores acumulados de la tasa de interés legal	**Aprobado**
Cálculo de la valorización de los intereses por demora de pagos.	**Aprobado**

B) A nivel de estado

B.1 Sistema

- Seguridad
  El administrador ingresa al sistema V&L mediante un Nick y clave.

B.2 Aceptación

- Interfaz del Sistema
  De acuerdo a las encuestas realizadas a las distintas obras públicas de la Provincia de Coronel Portillo, se determinó un grado de aceptación alto respecto a la facilidad del uso de la interfaz del sistema.V&L.

C) A nivel de prueba de tipo

C.1 Prueba de Carga

La base de datos y el ejecutable del sistema V&L han sido instalados en diferentes dispositivos (laptop y computadora de escrtiorio), cumpliendo satisfactoriamente su función.

Cuadro N° 10: Prueba de Carga.

APLICACIÓN	RESULTADO
Base de datos	Levantó
Ejecutable del sistema V&L	Levantó

Fuente: Elaboración propia.

CAPÍTULO V

CAPÍTULO 5: RESULTADOS Y DISCUSIÓN

5.1 RESULTADOS DEL CONTEO PARA EL PRE TEST O PRUEBA DE ENTRADA

Cuadro N° 11 : Frecuencia de la pregunta N° 01.

PE1.- ¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas mediante el software que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**	///	3	9%	9%
Regular	**3**	//////////////////////////	24	75%	84%
Bueno	**4**	////	4	13%	97%
Muy Bueno	**5**	/	1	3%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**3.09375**		

Fuente: Creación Propia.

Cuadro N° 12 : Frecuencia de la pregunta N° 02.

PE2.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado)?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	////////////////	16	50%	53%
Regular	**3**	//////////////	14	44%	97%
Bueno	**4**	/	1	3%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.46875**		

Fuente: Creación Propia.

Cuadro N° 13 : Frecuencia de la pregunta N° 03.

PE3.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización mediante un software (Excel u otros), que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	/ / / / / / / /	8	25%	28%
Regular	**3**	/ / / / / / / / / / / / / / / / / / / /	20	63%	91%
Bueno	**4**	/ / /	3	9%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.78125**		

Fuente: Creación Propia.

Cuadro N° 14 : Frecuencia de la pregunta N° 04.

PE4.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización, mediante un software (Excel u otros) que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/ /	2	6%	6%
Malo	**2**	/ / / / /	5	16%	22%
Regular	**3**	/ /	22	69%	91%
Bueno	**4**	/ / /	3	9%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.81250**		

Fuente: Creación Propia.

Cuadro N° 15 : Frecuencia de la pregunta N° 05.

PE5.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización, mediante un software (Excel u otros) que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	//	2	6%	6%
Malo	**2**	/////////	9	28%	34%
Regular	**3**	//////////////////////	21	66%	100%
Bueno	**4**		0	0%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.59375**		

Fuente: Creación Propia.

Cuadro N° 16 : Frecuencia de la pregunta N° 06.

PE6.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización, mediante un software (Excel u otros) que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	/	1	3%	6%
Regular	**3**	////////////////////////////	28	88%	94%
Bueno	**4**	//	2	6%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.96875**		

Fuente: Creación Propia.

Cuadro N° 17 : Frecuencia de la pregunta N° 07.

PE7.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización, mediante un software (Excel u otros) que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	/ / / / / / /	7	22%	25%
Regular	**3**	/ /	24	75%	100%
Bueno	**4**		0	0%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.71875**		

Fuente: Creación Propia.

Cuadro N° 18 : Frecuencia de la pregunta N° 08.

PE8.- ¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar con el software (Excel u otros), que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/ /	2	6%	6%
Malo	**2**	/ / / / / / / / / / /	11	34%	41%
Regular	**3**	/ / / / / / / / / / / / / / / / /	17	53%	94%
Bueno	**4**	/	1	3%	97%
Muy Bueno	**5**	/	1	3%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.625**		

Fuente: Creación Propia.

Cuadro N° 19 : Frecuencia de la pregunta N° 09.

PE9.- ¿Cómo evalúa los resultados obtenidos, mediante el software (Excel u otros), que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**	////////////////////////	23	72%	72%
Bueno	**4**	////////	8	25%	97%
Muy Bueno	**5**	/	1	3%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**3.3125**		

Fuente: Creación Propia.

Cuadro N° 20 : Frecuencia de la pregunta N° 10.

PE10.- ¿Cómo evalúa el desarrollo de la valorización de una obra adicional, mediante el software (Excel u otros), que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	//	2	6%	9%
Regular	**3**	//////////////////////////	27	84%	94%
Bueno	**4**	//	2	6%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.9375**		

Fuente: Creación Propia.

Cuadro N° 21 : Frecuencia de la pregunta N° 11.

PE11.- ¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables, mediante el software (Excel u otros), que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	/ /	2	6%	9%
Regular	**3**	/ /	28	88%	97%
Bueno	**4**	/	1	3%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.90625**		

Fuente: Creación Propia.

Cuadro N° 22 : Frecuencia de la pregunta N° 12.

PE12.- ¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago, mediante el software (Excel u otros), que usted utiliza?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**	/ /	2	6%	6%
Regular	**3**	/ /	28	88%	94%
Bueno	**4**	/ /	2	6%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**3.00000**		

Fuente: Creación Propia.

Cuadro N° 23 : Frecuencia de la pregunta N° 13.

PE13.- ¿Qué tan dinámico y amigable le parece el software que usted utiliza para el desarrollo de las valorizaciones de obras públicas?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**	///	3	9%	9%
Regular	**3**	//////////////////////	23	72%	81%
Bueno	**4**	//////	6	19%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**3.09375**		

Fuente: Creación Propia.

Cuadro N° 24 : Frecuencia de la pregunta N° 14.

PE14.- ¿Cómo califica usted las revisiones de las valorizaciones en función al tiempo y software que utilizan los profesionales revisores (Supervisor, Inspector u otros)?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**	/	1	3%	3%
Malo	**2**	//////////////////	18	56%	59%
Regular	**3**	//////////////	12	38%	97%
Bueno	**4**	/	1	3%	100%
Muy Bueno	**5**		0	0%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**2.40625**		

Fuente: Creación Propia.

5.2 RESULTADOS DEL CONTEO PARA EL POST TEST O PRUEBA DE SALIDA

Cuadro N° 25 : Frecuencia de la pregunta N° 01.

PE1.- ¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	////////////	10	31%	31%
Muy Bueno	**5**	//////////////////////	22	69%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.68750**		

Fuente: Creación Propia.

Cuadro N° 26 : Frecuencia de la pregunta N° 02.

PE2.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado), mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	///////////	11	34%	34%
Muy Bueno	**5**	//////////////////////	21	66%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.65625**		

Fuente: Creación Propia.

Cuadro N° 27 : Frecuencia de la pregunta N° 03.

PE3.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	////////////	11	34%	34%
Muy Bueno	**5**	//////////////////////	21	66%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.65625**		

Fuente: Creación Propia.

Cuadro N° 28 : Frecuencia de la pregunta N° 04.

PE4.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	//////////////////	18	56%	56%
Muy Bueno	**5**	//////////////	14	44%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.43750**		

Fuente: Creación Propia.

Cuadro N° 29 : Frecuencia de la pregunta N° 05.

PE5.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	//////////|	10	31%	31%
Muy Bueno	**5**	//////////////////////	22	69%	100%
TOTAL		***n=***	**32**	**100%**	
		Media (X)=	**4.68750**		

Fuente: Creación Propia.

Cuadro N° 30 : Frecuencia de la pregunta N° 06.

PE6.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	////////////////////|	19	59%	59%
Muy Bueno	**5**	////////////|	13	41%	100%
TOTAL		***n=***	**32**	**100%**	
		Media (X)=	**4.40625**		

Fuente: Creación Propia.

Cuadro N° 31 : Frecuencia de la pregunta N° 07.

PE7.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización mediante el uso del software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	///////	7	22%	22%
Muy Bueno	**5**	//////////////////////////	25	78%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.78125**		

Fuente: Creación Propia.

Cuadro N° 32 : Frecuencia de la pregunta N° 08.

PE8.- ¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar con el software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**	/	1	3%	3%
Bueno	**4**	//////////////	13	41%	44%
Muy Bueno	**5**	//////////////////	18	56%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.53125**		

Fuente: Creación Propia.

Cuadro N° 33 : Frecuencia de la pregunta N° 09.

PE9.- ¿Cómo evalúa los resultados obtenidos con el software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	////////////////////	20	63%	63%
Muy Bueno	**5**	//////////////	12	38%	100%
TOTAL		***n=***	**32**	**100%**	
		Media (X)=	**4.37500**		

Fuente: Creación Propia.

Cuadro N° 34 : Frecuencia de la pregunta N° 10.

PE10.- ¿Cómo evalúa el desarrollo de la valorización de una obra adicional utilizando el software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	/////////////////////	21	66%	66%
Muy Bueno	**5**	//////////////	11	34%	100%
TOTAL		***n=***	**32**	**100%**	
		Media (X)=	**4.34375**		

Fuente: Creación Propia.

Cuadro N° 35 : Frecuencia de la pregunta N° 11.

PE11.- ¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables utilizando el software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**		0	0%	0%
Bueno	**4**	////////////////////	17	53%	53%
Muy Bueno	**5**	////////////////	15	47%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.46875**		

Fuente: Creación Propia.

Cuadro N° 36 : Frecuencia de la pregunta N° 12.

PE12.- ¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago utilizando el software V&L?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**	/	1	3%	3%
Bueno	**4**	////////////////////	17	53%	56%
Muy Bueno	**5**	////////////////	14	44%	100%
TOTAL		***n*=**	**32**	**100%**	
		Media (X)=	**4.40625**		

Fuente: Creación Propia.

Cuadro N° 37 : Frecuencia de la pregunta N° 13.

PE13.- ¿Qué tan dinámico y amigable le parece el software V&L para el desarrollo de las valorizaciones de obras públicas?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**	/	1	3%	3%
Bueno	**4**	////////////////////	19	59%	63%
Muy Bueno	**5**	//////////////	12	38%	100%
TOTAL		***n=***	**32**	**100%**	
		Media (X)=	**4.34375**		

Fuente: Creación Propia.

Cuadro N° 38 : Frecuencia de la pregunta N° 14.

PE14.- ¿Cómo califica al software V&L para utilizarlo en las revisiones de las valorizaciones de obras públicas?					
Variables	**Escala de Likert**	**Conteo**	**Frecuencia**	**Porcentaje (%)**	**Porcentaje acumulado (%)**
Muy malo	**1**		0	0%	0%
Malo	**2**		0	0%	0%
Regular	**3**	/	1	3%	3%
Bueno	**4**	//////////////////////	18	56%	59%
Muy Bueno	**5**	//////////////	13	41%	100%
TOTAL		***n=***	**32**	**100%**	
		Media (X)=	**4.37500**		

Fuente: Creación Propia.

5.3 CÁLCULO DE LA DESVIACIÓN ESTANDAR Y VARIANZA PARA EL PRE TEST

Cuadro N° 39 : Cálculo de la desviación estándar y varianza del Pre Test.

	PREGUNTAS													
PERSONAS	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	4	3	3	3	3	3	3	2	3	2	3	3	3	3
2	3	3	2	3	2	3	3	2	3	3	3	3	3	2
3	3	3	4	3	3	3	3	2	3	3	3	3	3	2
4	2	2	2	2	2	3	3	2	3	3	3	3	3	2
5	3	2	3	3	3	3	3	3	4	4	3	3	3	3
6	3	2	2	3	2	3	2	2	3	3	3	3	3	3
7	4	3	3	3	3	3	3	3	3	3	3	3	3	3
8	4	4	4	3	3	4	3	4	4	4	4	3	4	3
9	4	3	4	4	3	3	3	3	4	3	3	3	4	2
10	3	2	3	2	2	3	3	2	3	3	2	3	3	2
11	3	3	3	3	3	3	3	3	3	3	3	3	3	4
12	3	2	3	3	3	3	2	2	3	3	3	3	3	2
13	3	2	2	2	2	2	2	1	3	1	1	2	2	2
14	3	3	3	4	3	4	3	3	3	3	3	3	4	2
15	3	2	3	3	3	3	3	2	4	3	3	3	3	2
16	3	3	3	3	3	3	3	3	4	3	3	3	4	3
17	3	2	2	1	1	3	3	3	5	3	3	3	4	2
18	3	3	3	3	3	3	2	3	4	3	3	3	3	2
19	2	1	2	2	2	3	3	3	3	3	3	3	2	3
20	3	2	3	3	3	3	3	3	3	3	3	4	3	2
21	3	2	3	3	2	3	2	5	4	3	3	3	4	2
22	3	3	3	3	3	3	3	3	3	3	3	3	3	3
23	3	3	3	3	3	3	3	3	3	3	3	3	3	2
24	3	2	3	3	3	3	3	2	3	3	3	4	3	3
25	5	3	3	3	3	3	3	3	3	3	3	3	3	3
26	2	2	1	1	1	1	1	1	3	3	2	2	3	1
27	3	2	3	4	2	3	3	3	3	2	3	3	2	3
28	3	2	3	2	3	3	3	2	3	3	3	3	3	2
29	3	2	2	3	2	3	2	2	4	3	3	3	3	2
30	3	3	3	3	3	3	3	3	3	3	3	3	3	3
31	3	3	3	3	3	3	3	3	3	3	3	3	3	2
32	3	2	2	3	3	3	2	3	3	3	3	3	3	2
MEDIA (X) =	**3.09375**	**2.46875**	**2.78125**	**2.81250**	**2.59375**	**2.96875**	**2.71875**	**2.62500**	**3.31250**	**2.93750**	**2.90625**	**3.00000**	**3.09375**	**2.40625**
DESVIACION ESTANDAR (S) =	**0.58802**	**0.62136**	**0.65915**	**0.69270**	**0.61484**	**0.47413**	**0.52267**	**0.79312**	**0.53506**	**0.50402**	**0.46555**	**0.35921**	**0.53033**	**0.61484**
VARIANZA (S^2) =	**0.34577**	**0.38609**	**0.43448**	**0.47984**	**0.37802**	**0.22480**	**0.27319**	**0.62903**	**0.28629**	**0.25403**	**0.21673**	**0.12903**	**0.28125**	**0.37802**

5.4 CÁLCULO DE LA DESVIACIÓN ESTANDAR Y VARIANZA PARA EL POST TEST

Cuadro N° 40 : Cálculo de la desviación estándar y varianza del Post Test.

	PREGUNTAS													
PERSONAS	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	5	4	4	4	4	5	5	5	5	4	5	4	4	4
2	5	5	5	4	5	4	5	5	5	4	5	5	4	5
3	5	5	5	5	5	5	5	5	5	5	5	5	5	5
4	5	5	5	4	5	4	5	5	4	4	4	4	5	5
5	4	4	4	4	4	4	4	4	4	4	4	4	4	4
6	5	5	5	5	5	5	5	5	4	4	4	4	4	4
7	4	4	4	4	4	4	4	4	4	4	4	4	4	4
8	4	4	4	4	4	4	4	4	4	4	4	4	4	4
9	4	4	4	4	4	4	4	3	4	4	4	3	4	3
10	5	5	5	4	5	4	5	5	5	4	5	4	5	5
11	4	4	4	4	4	4	4	4	5	4	4	4	4	4
12	5	5	4	4	5	4	5	5	4	4	4	4	5	4
13	4	4	5	5	4	4	5	4	4	5	5	5	4	5
14	5	5	5	4	4	4	5	4	4	4	5	5	4	4
15	4	4	4	4	4	4	4	4	4	4	4	4	4	4
16	5	5	5	4	5	4	5	4	4	4	4	4	4	4
17	4	5	4	4	5	4	5	4	4	4	4	4	4	4
18	5	5	4	4	5	4	5	5	4	4	4	4	4	4
19	5	5	5	5	5	5	5	5	5	5	5	5	4	5
20	5	5	5	5	5	4	5	4	4	4	4	4	4	4
21	4	4	5	5	5	5	5	4	5	5	5	5	3	4
22	5	5	5	5	5	5	5	5	5	5	5	5	5	5
23	5	4	4	4	4	4	4	4	4	4	4	4	4	4
24	5	5	5	5	5	5	5	5	5	5	5	5	5	5
25	5	5	5	4	5	4	5	5	4	4	4	4	5	5
26	5	5	5	5	5	5	5	5	4	5	4	5	4	5
27	5	4	5	5	5	5	5	4	4	4	5	5	5	5
28	5	5	5	4	5	4	5	5	4	5	5	5	5	4
29	4	5	5	5	5	5	5	5	4	4	4	4	4	5
30	5	5	5	5	5	5	5	5	5	5	5	5	5	4
31	5	5	5	5	5	5	5	5	5	5	5	5	5	5
32	5	5	5	5	5	5	5	5	5	5	5	5	5	4
MEDIA (X) =	**4.68750**	**4.65625**	**4.65625**	**4.43750**	**4.68750**	**4.40625**	**4.78125**	**4.53125**	**4.37500**	**4.34375**	**4.46875**	**4.40625**	**4.34375**	**4.37500**
DESVIACION ESTANDAR (S) =	**0.47093**	**0.48256**	**0.48256**	**0.50402**	**0.47093**	**0.49899**	**0.42001**	**0.56707**	**0.49187**	**0.48256**	**0.50701**	**0.55992**	**0.54532**	**0.55358**
VARIANZA (S^2) =	**0.22177**	**0.23286**	**0.23286**	**0.25403**	**0.22177**	**0.24899**	**0.17641**	**0.32157**	**0.24194**	**0.23286**	**0.25706**	**0.31351**	**0.29738**	**0.30645**

5.5 INTERPRETACIÓN DE RESULTADOS PARA EL PRE TEST

Cuadro N° 41 : Resultados de la Estadística Descriptiva del Pre Test.

N°	PREGUNTAS	n	Media	Desviación Estándar (S)	Varianza (S^2)
PE1	**¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas mediante el software que usted utiliza?**	32	3.09375	0.58802	0.34577
PE2	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado)?**	32	2.46875	0.62136	0.38609
PE3	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización mediante un software (Excel u otros), que usted utiliza?**	32	2.78125	0.65915	0.43448
PE4	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización, mediante un software (Excel u otros) que usted utiliza?**	32	2.81250	0.69270	0.47984
PE5	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización, mediante un software (Excel u otros) que usted utiliza?**	32	2.59375	0.61484	0.37802
PE6	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización, mediante un software (Excel u otros) que usted utiliza?**	32	2.96875	0.47413	0.22480
PE7	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización, mediante un software (Excel u otros) que usted utiliza?**	32	2.71875	0.52267	0.27319
PE8	**¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar con el software (Excel u otros), que usted utiliza?**	32	2.62500	0.79312	0.62903

PE9	**¿Cómo evalúa los resultados obtenidos, mediante el software (Excel u otros), que usted utiliza?**	32	3.31250	0.53506	0.28629
PE10	**¿Cómo evalúa el desarrollo de la valorización de una obra adicional, mediante el software (Excel u otros), que usted utiliza?**	32	2.93750	0.50402	0.25403
PE11	**¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables, mediante el software (Excel u otros), que usted utiliza?**	32	2.90625	0.46555	0.21673
PE12	**¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago, mediante el software (Excel u otros), que usted utiliza?**	32	3.00000	0.35921	0.12903
PE13	**¿Qué tan dinámico y amigable le parece el software que usted utiliza para el desarrollo de las valorizaciones de obras públicas?**	32	3.09375	0.53033	0.28125
PE14	**¿Cómo califica usted las revisiones de las valorizaciones en función al tiempo y software que utilizan los profesionales revisores (Supervisor, Inspector u otros)?**	32	2.40625	0.61484	0.37802
PROMEDIO			**2.83705**	**0.56964**	**0.33547**

Fuente: Creación Propia.

Interpretación: Del cuadro mostrado se puede inferir que al aplicar el instrumento de recolección de datos en la etapa de pre test, se obtuvo una media de 2.83705, y que de acuerdo a la escala de Likert se encuentra en el intervalo de [Malo, Regular].

5.6 INTERPRETACIÓN DE RESULTADOS PARA EL POST TEST

Cuadro N° 42 : Resultados de la Estadística Descriptiva del Post Test.

N°	PREGUNTAS	n	Media	Desviación Estándar (S)	Varianza (S^2)
PE1	**¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas mediante el uso del software V&L?**	32	4.68750	0.47093	0.22177
PE2	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado), mediante el uso del software V&L?**	32	4.65625	0.48256	0.23286
PE3	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización mediante el uso del software V&L?**	32	4.65625	0.48256	0.23286
PE4	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización mediante el uso del software V&L?**	32	4.43750	0.50402	0.25403
PE5	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización mediante el uso del software V&L?**	32	4.68750	0.47093	0.22177
PE6	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización mediante el uso del software V&L?**	32	4.40625	0.49899	0.24899
PE7	**¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización mediante el uso del software V&L?**	32	4.78125	0.42001	0.17641
PE8	**¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar con el software V&L?**	32	4.53125	0.56707	0.32157
PE9	**¿Cómo evalúa los resultados obtenidos con el software V&L?**	32	4.37500	0.49187	0.24194

PE10	**¿Cómo evalúa el desarrollo de la valorización de una obra adicional utilizando el software V&L?**	32	4.34375	0.48256	0.23286
PE11	**¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables utilizando el software V&L?**	32	4.46875	0.50701	0.25706
PE12	**¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago utilizando el software V&L?**	32	4.40625	0.55992	0.31351
PE13	**¿Qué tan dinámico y amigable le parece el software V&L para el desarrollo de las valorizaciones de obras públicas?**	32	4.34375	0.54532	0.29738
PE14	**¿Cómo califica al software V&L para utilizarlo en las revisiones de las valorizaciones de obras públicas?**	32	4.37500	0.55358	0.30645
PROMEDIO			**4.51116**	**0.50267**	**0.25425**

Fuente: Creación Propia.

Interpretación: Del cuadro mostrado se puede inferir que al aplicar el instrumento de recolección de datos en la etapa de post test, se obtuvo una media de 4.51116, y que de acuerdo a la escala de Likert se encuentra en el intervalo de [Bueno, Muy Bueno].

5.7 PRUEBA DE HIPÓTESIS

5.7.1. FORMULACIÓN DE CUADRO DE VALORES DE INDICADORES

La contrastación de la hipótesis se realiza de acuerdo al diseño de investigación, el cual es conocido como Pre-Test y Post-Test, que se representa mediante la siguiente simbología:

$$M: O_1 \text{ ------ } X \text{ ------ } O_2$$

Donde:

- M: Muestra representada por el plantel técnico de las distintas obras visitadas en la Provincia de Coronel Portillo.

- O_1: Análisis de los resultados antes de implementar el sistema V&L para el desarrollo de las valorizaciones de obras públicas (Pre – Prueba).
- X: Aplicación del sistema (software).
- O_2: Análisis de los resultados después de implementar el sistema V&L para el desarrollo de las valorizaciones de obras públicas (Post – Prueba).

El procedimiento consiste en determinar en primer lugar una tabla de rango de valores, la cual nos permite ubicar valores cuantitativos de los indicadores, por medio de valores cualitativos expresados en este rango. Posteriormente realizamos la comparación de valores entre indicadores de acuerdo al diseño de contrastación; esta comparación nos permite finalmente aceptar o rechazar la hipótesis de acuerdo a los estándares de la estadística.

Para los valores cuantitativos:

Muy Malo : 1	Malo: 2	Regular: 3	Bueno: 4	Muy Bueno: 5

5.7.2. SUPUESTOS DE LA PRUEBA DE HIPÓTESIS

1. Los datos muestrales fueron seleccionados por conveniencia o de manera intencional, comprendiendo un total de 32 profesionales, los cuales están distribuidos entre residentes de obras, supervisores y/o inspectores, asistentes de obras y funcionarios públicos.

2. La hipótesis nula H_o es la negación de la relación existente entre la variable independiente y la variable dependiente, y, la hipótesis H_a es la afirmación correspondiente.

3. Para la presente investigación se utilizó un nivel de confianza del 95% y un nivel de significancia del 5%.

4. Se acepta la hipótesis nula si el valor de Z_p es menor al valor en tabla de Z_c, caso contrario se rechaza la Ho y se acepta la hipótesis alternativa Ha.

5.7.3. CÁLCULO DEL VALOR CRÍTICO Y FUNCIÓN PRUEBA

Después de analizar las diferencias entre los indicadores en la fase de Pre-Test y Post-Test, se concluye que existe una mejora después de implementar el sistema V&L.

Resumen de resultados de valores

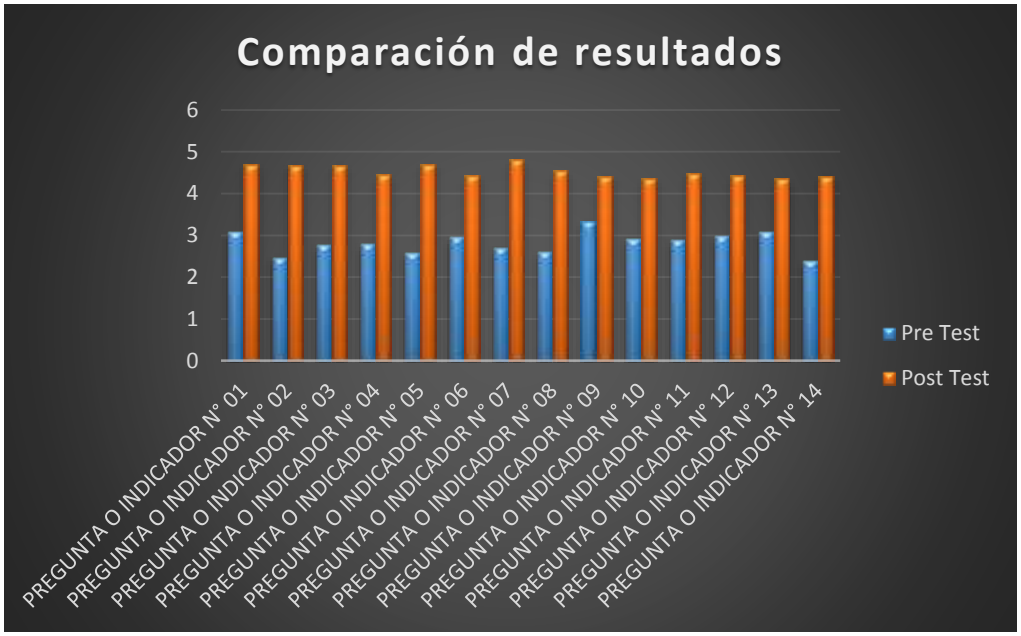
Cuadro N° 43: Resumen de Resultados Finales del Pre Test y Post Test.

N°	PREGUNTAS O INDICADORES	Pre Test	Post Test
PE1	¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas?	3.09375	4.68750
PE2	¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado)?	2.46875	4.65625
PE3	¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización?	2.78125	4.65625
PE4	¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización?	2.81250	4.43750
PE5	¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización?	2.59375	4.68750
PE6	¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización?	2.96875	4.40625

PE7	¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización?	2.71875	4.78125
PE8	¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar?	2.62500	4.53125
PE9	¿Cómo evalúa los resultados obtenidos?	3.31250	4.37500
PE10	¿Cómo evalúa el desarrollo de la valorización de una obra adicional?	2.93750	4.34375
PE11	¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables?	2.90625	4.46875
PE12	¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago?	3.00000	4.40625
PE13	¿Qué tan dinámico y amigable le parece el software para el desarrollo de las valorizaciones de obras públicas?	3.09375	4.34375
PE14	¿Cómo califica usted las revisiones de las valorizaciones en función al tiempo que utilizan los profesionales revisores (Supervisor, Inspector u otros)?	2.40625	4.37500
	MEDIA (X)	***2.83705***	***4.51116***
	VARIANZA (S^2)	***0.33547***	***0.25425***
	DESVIACIÓN ESTÁNDAR (S)	***0.56964***	***0.50267***
	MUESTRA	***32***	***32***

Fuente: Creación Propia.

Imagen 5.1: Comparación de resultados mediante diagrama de barras.



Fuente: Creación Propia.

5.7.4. HIPÓTESIS

H_o: $O_1 \geq O_2$

H_a: $O_2 > O_1$

Donde:

- O_1 : Análisis de los resultados antes de implementar el sistema V&L para el desarrollo de las valorizaciones de obras públicas (Pre – Prueba).

- O2 : Análisis de los resultados después de implementar el sistema V&L para el desarrollo de las valorizaciones de obras públicas (Post – Prueba).

Nivel de confianza = 95%.

Nivel de significancia= 5%.

Criterio:

Si $Z_p > Z_c$, entonces se rechaza la hipótesis nula o teórica (H_o) y se acepta la hipótesis alterna (H_a).

Donde:

- Z_c : Coeficiente de confiabilidad crítico.
- Z_p : Coeficiente de confiabilidad de prueba.

A) Determinación del Z_c:

Para el cálculo del Z_c, se utilizó la tabla de distribución normal, la cual se adjunta en los anexos.

Zc	f(z)
1.65	0.951
X	0.95
1.64	0.949

Realizando la interpolación correspondiente, se determinó el valor de Z_c.

Por lo tanto, Z_c=1.645.

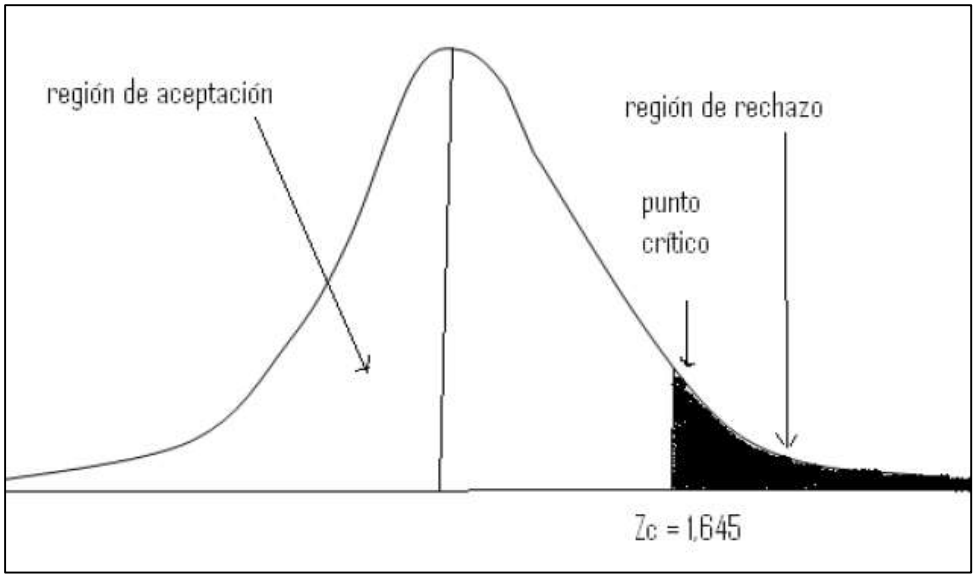
Determinación del Z_P:

Por formula, el valor de Z_p se calcula de la siguiente manera:

$$Z_P = \frac{O_2 - O_1}{\sqrt{\frac{(S_2)^2}{n_2} + \frac{(S_1)^2}{n_1}}}$$

Remplazando los valores, tenemos:

$$Z_P = \frac{4.51116071 - 2.83705357}{\sqrt{\frac{(0.50266632)^2}{32} + \frac{(0.56964179)^2}{32}}} = 12.47$$



Por lo tanto:

Como Z_p=12.47 es mayor que Z_c=1.645 (Zona crítica), entonces se rechaza la hipótesis nula o teórica (H_o) y se **acepta** la hipótesis alterna (H_a : $O_2 > O_1$); es decir, el análisis de los resultados después de implementar el sistema V&L para el desarrollo de las valorizaciones de obras públicas (Post – Prueba), es mayor que el análisis de los resultados antes de implementar el sistema V&L para el desarrollo de las valorizaciones de obras públicas (Pre – Prueba).

CONCLUSIONES

Según el análisis realizado y teniendo en consideración los objetivos de la investigación se establecieron las siguientes conclusiones.

1. De acuerdo al análisis de las preguntas planteadas en las encuestas y cálculos realizados en el capítulo V, la media de los indicadores en la fase de post test es mayor que la media en la fase de pre test, es decir, 4.51116 > 2.83705, en consecuencia, se concluye que la hipótesis general planteada: *"La creación de un nuevo software es una alternativa de solución cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali, por lo tanto, el ahorro de tiempo está garantizado"*, es aceptada.

2. De acuerdo al análisis de las preguntas N° 04, 05, 06, 07, 10, 11 y 12 planteadas en las encuestas y cálculos realizados en el capítulo V, la media de estos indicadores en la fase de post test es mayor que la media en la fase de pre test, es decir, 4.50446 > 2.84821, en consecuencia, se concluye que la hipótesis específica N° 01 planteada: *"Con la creación del nuevo software se minimizarán las limitaciones existentes que presentan otros programas cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo – Ucayali"*, es aceptada.

3. De acuerdo al análisis de las preguntas N° 10, 11, 12 y 14 planteadas en las encuestas y cálculos realizados en el capítulo V, la media de estos indicadores en la fase de post test es mayor que la media en la fase de pre test, es decir, 4.39844 > 2.81250, en consecuencia, se concluye que la hipótesis específica N° 02 planteada: *"Con la Implementación de los distintos tipos de valorizaciones en el nuevo software, se logrará un control económico adecuado de las obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali"*, es aceptada.

4. El sistema V&L desarrolla valorizaciones de obras públicas, con normativa vigente (Ley de Contrataciones del Estado N° 30225, modificado con D.L. 1444. y su Reglamento aprobado con D.S. 344-2018-EF).

5. El sistema V&L tiene incorporado en su base de datos los índices unificados de precios de la construcción publicados mensualmente por el Instituto Nacional de Estadística e Informática (INEI), además incluye un link de descarga de los índices antes mencionados.

6. El sistema V&L permite importar o alimentar a su base de datos los presupuestos y formulas polinómicas de las obras que son exportadas del S10 en archivo Excel.

7. Finalmente, se concluye que las hipótesis planteadas son aceptadas, por lo tanto, queda demostrado, que el “SOFTWARE V&L VERSIÓN 1.0 – 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO – UCAYALI”, constituye una alternativa de solución para los problemas considerados dentro de la investigación, con un nivel de confianza del 95% y nivel de significancia del 5%.

SUGERENCIAS

1. Se recomienda utilizar el MySQL como gestor de una base de datos, cuando se desee diseñar e implementar un nuevo software para desarrollar valorizaciones de obras públicas.

2. Para la instalación del sistema V&L, se recomienda instalar previamente el Xampp Control Panel y el Java Runtime Environment, para el correcto funcionamiento del mismo.

3. Se recomienda importar mensualmente a la base de datos del sistema V&L los índices unificados de precios de la construcción publicados por el INEI, para el cálculo correcto en las valorizaciones.

4. Para el desarrollo de las valorizaciones con el sistema V&L, se recomienda utilizar una laptop o computadora de escritorio con una pantalla mínima de 21 pulgadas, para una visualización óptima de los reportes generados por el sistema.

5. Se recomienda difundir la presente investigación a través de la Escuela Profesional de Ingeniería Civil, con la finalidad de aportar una base para futuras investigaciones de software para desarrollar valorizaciones de obras.

REFERENCIAS BIBLIOGRÁFICAS

BIBLIOGRAFÍA FÍSICA

Comisión Permanente del Congreso de la República (2019). *Ley 30225 con modificatorias D.L. 1341 y D.L. 1444 y su Reglamento aprobado con D.S. 344-2018-EF.* Lima, Perú: Congreso de la República del Perú.

Eco, U, (2007). *Cómo se hace una tesis.* México D.F.: Editorial Mantaro.

Jiménez MarÍn, R.J.P. (2018). *Diseño e Implementación de un Sistema Aplicando el Lenguaje de Programación LUA para el Análisis de Elementos Estructurales en Edificaciones (Tesis de pregrado)*. Universidad Nacional de Ucayali, Perú.

Miano Montero, O. (2015). *Supervisión en la ejecución de obras públicas y privadas*. Lima, Perú: Editorial Miano.

Pressman, R.S. (2010). *Ingeniería del Software, un enfoque práctico*. México, D.F.: Editorial McGraw-Hill.

Ramos Salazar, J. (2015). *Sistema de reajuste de precios en la construcción mediante fórmulas polinómicas*. Lima, Perú: Editorial Miano.

Román Claros, W.G. (2018). *Pautas para elaborar proyecto de tesis y trabajos de investigación en ingeniería (Parte I).* Pucallpa, Perú.

Salinas Seminario, M. (2009). *Administración de Contratos de Obras Públicas.* Lima, Perú: Editorial ICG.

Tafur Portilla, R. (1994). *La tesis Universitaria*. Lima, Perú: Editorial Mantaro.

Valderrama Mendoza, S. (2009). *Pasos para elaborar proyecto y tesis de investigación científica*. Lima, Perú: Editorial San Marcos.

BIBLIOGRAFÍA ELECTRÓNICA

https://psicologiaymente.net/miscelanea/tipos-de-investigacion#!

http://www.excellentias.com/historia-de-excel/

http://www.s10peru.com/historia_i.php

https://es.scribd.com/document/246812436/KYBCOP

https://es.scribd.com/document/204083996/Manual-Cpoc-Ver-4-01

http://www.ddbexpress.com/#

https://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational

https://es.wikipedia.org/wiki/Lenguaje_unificado_de_modelado

https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)

https://es.wikipedia.org/wiki/NetBeans

https://es.wikipedia.org/wiki/PhpMyAdmin

https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos

https://es.wikipedia.org/wiki/Metodolog%C3%ADa_de_desarrollo_de_software#Incremental

https://es.wikipedia.org/wiki/MySQL

https://computerhoy.com/noticias/internet/que-es-localhost-ip-127001-que-utiliza-48046

https://www.youtube.com/watch?v=iXJfDZAt2qs

https://www.youtube.com/watch?v=tCLeJCuBSZM&list=PLK4lKI3prIaDcvFcS-3oPXUAMgG7tBrer

https://www.youtube.com/watch?v=oIW4enAXNkU&t=433s

https://sabermetodologia.wordpress.com/2016/02/08/variables-dimensiones-e-indicadores/

MATRIZ DE CONSISTENCIA

TÍTULO: "SOFTWARE V&L VERSIÓN 1.0 – 2017 PARA DESARROLLAR VALORIZACIONES DE OBRAS PÚBLICAS POR CONTRATA, EN LA PROVINCIA DE CORONEL PORTILLO - UCAYALI"

FORMULACIÓN DEL PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLES	DIMENSIONES	INDICADORES	INSTRUMENTO	METODOLOGÍA
PROBLEMA GENERAL ¿La creación de un nuevo software, es una alternativa de solución para el ahorro de tiempo cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali?	**OBJETIVO GENERAL** Crear un nuevo software como alternativa de solución para el ahorro de tiempo cuando se desarrollen valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali.	**HIPÓTESIS GENERAL** La creación de un nuevo software es una alternativa de solución cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali, por lo tanto, el ahorro de tiempo está garantizado.	**Variable Independiente** Software para desarrollar valorizaciones.	- Software para desarrollar valorización de obra principal.	- Desarrollo de la valorización de obra principal.	Cuestionario o Ficha de Encuesta de Pre Test y Post Test. Ley 30225 con modificatorias D.L. 1341 y D.L. 1444 y su Reglamento aprobado con D.S. 344-2018-EF	**TIPO DE INVESTIGACIÓN** Los tipos de investigación utilizados para la presente tesis son: - Investigación Aplicada. - Investigación Tecnológica. - Investigación Cualitativa y Cuantitativa (Híbrido o Mixto) **NIVEL DE INVESTIGACIÓN** El nivel de la investigación utilizada en la presente tesis es de tipo: **a) Descriptiva:** Comprende la descripción, registro, análisis e interpretación del desarrollo de las valorizaciones de obras aplicando un software. **POBLACIÓN** - La población fue referencial y está constituida por 74 Obras Públicas de la Provincia de Coronel Portillo, las cuales fueron obtenidas del Sistema Electrónico de Contrataciones del Estado (SEACE), correspondientes al año 2018 y 2019. **MUESTRA** - El método de muestreo utilizado, fue el no probabilístico, específicamente el muestreo intencional o de conveniencia, para ello se tomó como unidad de análisis ocho (08) obras públicas. Para la validación de la hipótesis, se consideró como muestra a 32 profesionales de las distintas obras públicas visitadas. Diseño estadístico de prueba: O: M_1 ------ X ------ M_2
				- Software para desarrollar valorización de obras adicionales.	- Desarrollo de la valorización de obras adicionales.		
				- Software para desarrollar valorización de mayores gastos generales variables.	- Desarrollo de la valorización de mayores gastos generales variables.		
				- Software para desarrollar valorización de intereses por demora de pagos.	- Desarrollo de la valorización de intereses por demora de pagos.		
			Variable Dependiente Ahorro de tiempo.		- Desarrollo de los cuatro (04) tipos de valorización.		
PROBLEMAS ESPECÍFICOS **a.** ¿Es posible que con la creación de un nuevo software se minimicen las limitaciones existentes que presentan otros programas cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo -	**OBJETIVOS ESPECÍFICOS** a. Minimizar las limitaciones existentes que presentan otros software cuando se desarrollen valorizaciones de obras públicas por contrata, en la provincia de Coronel Portillo - Ucayali.	**HIPÓTESIS ESPECÍFICAS** a. Con la creación del nuevo software se minimizarán las limitaciones existentes que presentan otros programas cuando se desarrollan valorizaciones de obras públicas por contrata, en la Provincia de Coronel Portillo – Ucayali.					
b. ¿Es posible que implementando los distintos tipos de valorizaciones en el nuevo software se pueda realizar un control económico adecuado de las obras públicas por contrata, en la provincia de Coronel Portillo - Ucayali?	b. Implementar los distintos tipos de valorizaciones en el nuevo software para realizar un control económico adecuado de las obras públicas por contrata, en la provincia de coronel Portillo - Ucayali.	b. Con la Implementación de los distintos tipos de valorizaciones en el nuevo software, se logrará un control económico adecuado de las obras públicas por contrata, en la Provincia de Coronel Portillo - Ucayali.					

ANEXOS

UNIVERSIDAD NACIONAL DE UCAYALI
FACULTAD DE INGENIERÍA DE SISTEMAS Y DE INGENIERÍA CIVIL
CARRERA PROFESIONAL DE INGENIERÍA CIVIL

Núm. De Encuesta: _____

Fecha: _____

ENCUESTA PARA EL PRE TEST O PRUEBA DE ENTRADA

ENCUESTA DIRIGIDA A LOS INGENIERIOS RESIDENTES, SUPERVISORES O INSPECTORES DE OBRAS, ASISTENTES DE OBRAS Y AFINES.

Indicaciones Previas

Ing. (a) Residente, Supervisor o Inspector de obras, asistente de obras y afines, a continuación les presento unas preguntas que pretenden medir la efectividad del desarrollo de las valorizaciones mediante un software (Excel u otros), que ustedes utilicen y/o conocen actualmente en su centro de trabajo u obra. Por favor sírvase contestar el siguiente test, marcando con un aspa donde crea conveniente:

A. INFORMACIÓN GENERAL

PG1: INFORMACIÓN BÁSICA DE LA PERSONA NATURAL O EMPRESA

1. Nombre y Apellidos: _____
2. Entidad o Empresa donde labora: _____
3. Área: _____
4. Cargo que desempeña:

   Ing. Residente ( ) Ing. Supervisor o Inspector ( ) Asistente de Obra ( )

   Otros y/o Afines ( )

PG2: SEXO

Masculino ( ) Femenino ( )

PG3: EDAD

1. De 22 a 35 años ( )
2. De 36 a 65 años ( )
3. Más de 65 años ( )

B. PREGUNTAS ESPECÍFICAS

Instrucciones: Lea detenidamente las preguntas y marque con un círculo la respuesta que usted crea conveniente.

Preguntas de Encuesta de Pre - Test

PE1.- ¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas mediante el software que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE2.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado)?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE3.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización mediante un software (Excel u otros), que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE4.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización, mediante un software (Excel u otros) que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE5.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización, mediante un software (Excel u otros) que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE6.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización, mediante un software (Excel u otros) que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE7.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización, mediante un software (Excel u otros) que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE8.- ¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar con el software (Excel u otros), que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE9.- ¿Cómo evalúa los resultados obtenidos, mediante el software (Excel u otros), que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE10.- ¿Cómo evalúa el desarrollo de la valorización de una obra adicional, mediante el software (Excel u otros), que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE11.- ¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables, mediante el software (Excel u otros), que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE12.- ¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago, mediante el software (Excel u otros), que usted utiliza?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE13.- ¿Qué tan dinámico y amigable le parece el software que usted utiliza para el desarrollo de las valorizaciones de obras públicas?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE14.- ¿Cómo califica usted las revisiones de las valorizaciones en función al tiempo y software que utilizan los profesionales revisores (Supervisor, Inspector u otros)?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

UNIVERSIDAD NACIONAL DE UCAYALI
FACULTAD DE INGENIERÍA DE SISTEMAS Y DE INGENIERÍA CIVIL
CARRERA PROFESIONAL DE INGENIERÍA CIVIL

Núm. De Encuesta: _____

Fecha: _____

ENCUESTA PARA EL POST TEST O PRUEBA DE SALIDA

ENCUESTA DIRIGIDA A LOS INGENIERIOS RESIDENTES, SUPERVISORES O INSPECTORES DE OBRAS, ASISTENTES DE OBRAS Y AFINES.

Indicaciones Previas

Ing. (a) Residente, Supervisor o Inspector de obras, asistente de obras y afines, a continuación les presento unas preguntas que pretenden medir la efectividad del desarrollo de las valorizaciones mediante el uso del software V&L. Por favor sírvase contestar el siguiente test, marcando con un aspa donde crea conveniente:

A. INFORMACIÓN GENERAL

PG1: INFORMACIÓN BÁSICA DE LA PERSONA NATURAL O EMPRESA

5. Nombre y Apellidos: _____
6. Entidad o Empresa donde labora: _____
7. Área: _____
8. Cargo que desempeña:

   Ing. Residente ( ) Ing. Supervisor o Inspector ( ) Asistente de Obra ( )

   Otros y/o Afines ( )

PG2: SEXO

Masculino ( ) Femenino ( )

PG3: EDAD

4. De 22 a 35 años ( )
5. De 36 a 65 años ( )
6. Más de 65 años ( )

B. PREGUNTAS ESPECÍFICAS

Instrucciones: Lea detenidamente las preguntas y marque con un círculo la respuesta que usted crea conveniente.

Preguntas de Encuesta de Post - Test

PE1.- ¿Qué le parece el tiempo que le toma desarrollar las valorizaciones de obras públicas mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE2.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar una primera valorización (sin tener un formato predeterminado), mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE3.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de los coeficientes de reajuste (Kr) de una valorización mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE4.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto directo de una valorización mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE5.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la deducción del reajuste que no corresponde por adelanto para materiales de una valorización mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE6.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto directo de una valorización mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE7.- ¿Cuál es el calificativo que le da al tiempo de demora para desarrollar el cálculo de la amortización del adelanto para materiales de una valorización mediante el uso del software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE8.- ¿Qué le parece la forma de registrar y ordenar todas las partidas de las distintas especialidades por valorizar con el software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE9.- ¿Cómo evalúa los resultados obtenidos con el software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE10.- ¿Cómo evalúa el desarrollo de la valorización de una obra adicional utilizando el software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE11.- ¿Cómo evalúa el desarrollo de la valorización de mayores gastos generales variables utilizando el software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE12.- ¿Cómo evalúa el desarrollo de la valorización de intereses por demora de pago utilizando el software V&L?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE13.- ¿Qué tan dinámico y amigable le parece el software V&L para el desarrollo de las valorizaciones de obras públicas?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PE14.- ¿Cómo califica al software V&L para utilizarlo en las revisiones de las valorizaciones de obras públicas?

1) Muy malo 2) Malo 3) Regular 4) Bueno 5) Muy Bueno

PANEL FOTOGRÁFICO



Foto N° 01: Encuesta realizada al Ing. supervisor de la I.E.I. N° 675 – AA.HH Guadalupe.



Foto N° 02: Encuesta realizada al Ing. Residente de la I.E.I. N° 675 – AA.HH Guadalupe.



Foto N° 03: Visualización del desarrollo de valorizaciones con el Sistema V&L.



Foto N° 04: Fotografía con el Ing. Residente, después de la visualización del Sistema V&L.



Foto N° 05: Encuesta realizada al asistente técnico del supervisor de la I.E.I. N° 675 – AA.HH Guadalupe.



Foto N° 06: Fotografía con los asistentes técnicos, después de la visualización del Sistema V&L.



Foto N° 07: Visita de campo a la construcción de la I.E.I. N° 678 - AA. HH. Isabel la Católica.



Foto N° 08: Entrevista para la aplicación de las encuestas con el Ingeniero Supervisor e Ingeniero Residente.



Foto N° 09: Explicación del desarrollo de valorizaciones con el Sistema V&L, en compañía del Ingeniero residente.



Foto N° 10: Fotografía para validar la visita a la ejecución de obra I.E.I. N° 678 - AA. HH. Isabel la Católica.



Foto N° 11: Visita de campo a la construcción de la I.E. N° 64668 – AA.HH La Perla, ubicada en el distrito de Yarinacocha, provincia de Coronel Portillo, Ucayali.



Foto N° 12: Encuesta realizada a la asistente técnica del residente de la I.E. N° 64668 – AA.HH La Perla.



Foto N° 13: Explicación del desarrollo de valorizaciones con el Sistema V&L.



Foto N° 14: Vista del interfaz en el Sistema V&L.



Foto N° 15: Culminación del desarrollo de valorizaciones con el Sistema V&L.



Foto N° 16: Aplicación de la encuesta Post test, con el Ingeniero residente y asistente de obra.



Foto N° 17: Visita de campo a la construcción de la I.E.I. N° 696 - AA. HH. Juan Silva Bocanegra.



Foto N° 18: Fotografía para validar la visita a la ejecución de obra I.E.I. N° 696 - AA. HH. Juan Silva Bocanegra.



Foto N° 19: Explicación del desarrollo de valorizaciones con el Sistema V&L, con el asistente de residente de obra.



Foto N° 20: Introducción del desarrollo de valorizaciones con el Sistema V&L, con el Ingeniero Supervisor de la obra I.E.I. N° 696 - AA. HH. Juan Silva Bocanegra.



Foto N° 21: Explicación detallada del desarrollo de valorizaciones con el Sistema V&L, con el Ingeniero Supervisor de obra.



Foto N° 22: Fotografía con el Ingeniero Supervisor, después de la aplicación del Sistema V&L.



Foto N° 23: Introducción del desarrollo de valorizaciones con el Sistema V&L, con la asistente del residente de la obra denominada: “Mejoramiento a nivel de afirmado del jr. Venecia hasta calle el Álamo”.



Foto N° 24: Explicación detallada del desarrollo de valorizaciones con el Sistema V&L, con la asistente de obra del residente.



Foto N° 25: Vista de la ficha técnica de obra en el Sistema V&L.



Foto N° 26: Visualización del desarrollo de valorizaciones con el Sistema V&L.



Foto N° 27: Introducción y explicación del desarrollo de valorizaciones con el Sistema V&L, con el asistente del residente de la obra denominada: “Mejoramiento a nivel de afirmado del Jr. Nuevo San Juan, Jr. Rogelio Fernández Ruiz y jr. Paris en la CC.NN. Nuevo San Juan”.



Foto N° 28: Introducción y explicación del desarrollo de valorizaciones con el Sistema V&L, con el asistente de obra Ing. Marvin Martin Ríos Rengifo.



Foto N° 29: Aplicación de la encuesta Pre test, con el Ingeniero residente de obra, Elmer Rojas Cienfuegos.



Foto N° 30: Introducción del desarrollo de valorizaciones con el Sistema V&L.



Foto N° 31: Explicación del desarrollo de valorizaciones con el Sistema V&L.



Foto N° 32: Fotografía con el Ingeniero Elmer Rojas Cienfuegos, luego de observar el desarrollo de valorizaciones con el Sistema V&L.

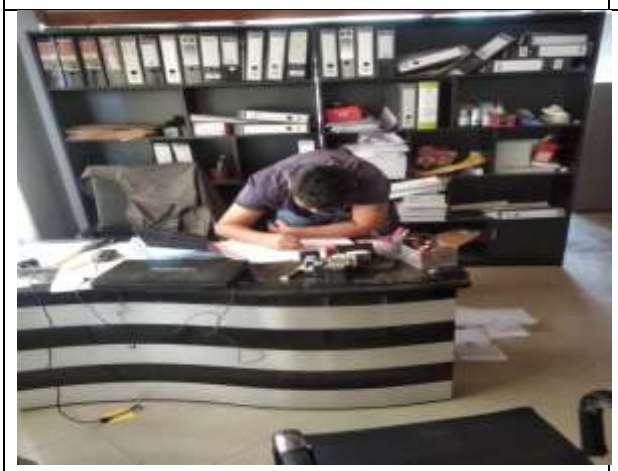


Foto N° 33: Aplicación de la encuesta Pre test, con el Ingeniero Residente Elvin David Polo López.



Foto N° 34: Fotografía con el Ingeniero Elvin David Polo López, luego de observar el desarrollo de valorizaciones con el Sistema V&L y contestar la encuesta de post test.



Foto N° 35: Visita de campo a la Municipalidad Distrital de Yarinacocha.



Foto N° 36: Visita a la Subgerencia de obras Públicas de la MDY, a cargo del Ing. Juan Carlos Ruiz Borbor.



Foto N° 37: Aplicación de la encuesta Pre test y Post Test, con el Sub Gerente de Obras Públicas de la Municipalidad Distrital de Yarinacocha.



Foto N° 38: Fotografía con el Ingeniero Juan Carlos Ruiz Borbor, luego de observar el desarrollo de valorizaciones con el Sistema V&L.



Foto N° 39: Visita a la Gerencia de Obras Públicas de la MDY, a cargo del Ing. Harold Paolo Flores Ruiz.



Foto N° 40: Aplicación de la encuesta Pre test y Post Test, con el Gerente de Obras Publicas de la Municipalidad Distrital de Yarinacocha.



Foto N° 41: Fotografía con el Ing. Harold Paolo Flores Ruiz, luego de observar el desarrollo de valorizaciones con el Sistema V&L.



Foto N° 42: Visita al plantel técnico de la Subgerencia de Obras Públicas de la Municipalidad Distrital de Yarinacocha.



Foto N° 43: Entrevista con el Ing. Sandro Augusto Tuesta Ruiz.



Foto N° 44: Entrevista con la Bach. Rossmery Armas Gálvez.



Foto N° 45: Entrevista con el Bach. Gino Paolo Modena Navas.



Foto N° 46: Entrevista con el Ing. Álvaro Ernesto Mejía Vásquez.

TABLA PARA LOS VALORES DE Z_C

z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)
-4	0.000	-3.55	0.000	-3.1	0.001	-2.65	0.004	-2.2	0.014	-1.75	0.040	-1.3	0.097	-0.85	0.198	-0.4	0.345
-3.99	0.000	-3.54	0.000	-3.09	0.001	-2.64	0.004	-2.19	0.014	-1.74	0.041	-1.29	0.099	-0.84	0.200	-0.39	0.348
-3.98	0.000	-3.53	0.000	-3.08	0.001	-2.63	0.004	-2.18	0.015	-1.73	0.042	-1.28	0.100	-0.83	0.203	-0.38	0.352
-3.97	0.000	-3.52	0.000	-3.07	0.001	-2.62	0.004	-2.17	0.015	-1.72	0.043	-1.27	0.102	-0.82	0.206	-0.37	0.356
-3.96	0.000	-3.51	0.000	-3.06	0.001	-2.61	0.005	-2.16	0.015	-1.71	0.044	-1.26	0.104	-0.81	0.209	-0.36	0.359
-3.95	0.000	-3.5	0.000	-3.05	0.001	-2.60	0.005	-2.15	0.016	-1.7	0.045	-1.25	0.106	-0.8	0.212	-0.35	0.363
-3.94	0.000	-3.49	0.000	-3.04	0.001	-2.59	0.005	-2.14	0.016	-1.69	0.046	-1.24	0.107	-0.79	0.215	-0.34	0.367
-3.93	0.000	-3.48	0.000	-3.03	0.001	-2.58	0.005	-2.13	0.017	-1.68	0.046	-1.23	0.109	-0.78	0.218	-0.33	0.371
-3.92	0.000	-3.47	0.000	-3.02	0.001	-2.57	0.005	-2.12	0.017	-1.67	0.047	-1.22	0.111	-0.77	0.221	-0.32	0.374
-3.91	0.000	-3.46	0.000	-3.01	0.001	-2.56	0.005	-2.11	0.017	-1.66	0.048	-1.21	0.113	-0.76	0.224	-0.31	0.378
-3.9	0.000	-3.45	0.000	-3	0.001	-2.55	0.005	-2.1	0.018	-1.65	0.049	-1.2	0.115	-0.75	0.227	-0.3	0.382
-3.89	0.000	-3.44	0.000	-2.99	0.001	-2.54	0.006	-2.09	0.018	-1.64	0.051	-1.19	0.117	-0.74	0.230	-0.29	0.386
-3.88	0.000	-3.43	0.000	-2.98	0.001	-2.53	0.006	-2.08	0.019	-1.63	0.052	-1.18	0.119	-0.73	0.233	-0.28	0.390
-3.87	0.000	-3.42	0.000	-2.97	0.001	-2.52	0.006	-2.07	0.019	-1.62	0.053	-1.17	0.121	-0.72	0.236	-0.27	0.394
-3.86	0.000	-3.41	0.000	-2.96	0.002	-2.51	0.006	-2.06	0.020	-1.61	0.054	-1.16	0.123	-0.71	0.239	-0.26	0.397
-3.85	0.000	-3.4	0.000	-2.95	0.002	-2.50	0.006	-2.05	0.020	-1.6	0.055	-1.15	0.125	-0.7	0.242	-0.25	0.401
-3.84	0.000	-3.39	0.000	-2.94	0.002	-2.49	0.006	-2.04	0.021	-1.59	0.056	-1.14	0.127	-0.69	0.245	-0.24	0.405
-3.83	0.000	-3.38	0.000	-2.93	0.002	-2.48	0.007	-2.03	0.021	-1.58	0.057	-1.13	0.129	-0.68	0.248	-0.23	0.409
-3.82	0.000	-3.37	0.000	-2.92	0.002	-2.47	0.007	-2.02	0.022	-1.57	0.058	-1.12	0.131	-0.67	0.251	-0.22	0.413
-3.81	0.000	-3.36	0.000	-2.91	0.002	-2.46	0.007	-2.01	0.022	-1.56	0.059	-1.11	0.133	-0.66	0.255	-0.21	0.417
-3.8	0.000	-3.35	0.000	-2.9	0.002	-2.45	0.007	-2	0.023	-1.55	0.061	-1.1	0.136	-0.65	0.258	-0.2	0.421
-3.79	0.000	-3.34	0.000	-2.89	0.002	-2.44	0.007	-1.99	0.023	-1.54	0.062	-1.09	0.138	-0.64	0.261	-0.19	0.425
-3.78	0.000	-3.33	0.000	-2.88	0.002	-2.43	0.008	-1.98	0.024	-1.53	0.063	-1.08	0.140	-0.63	0.264	-0.18	0.429
-3.77	0.000	-3.32	0.000	-2.87	0.002	-2.42	0.008	-1.97	0.024	-1.52	0.064	-1.07	0.142	-0.62	0.268	-0.17	0.433
-3.76	0.000	-3.31	0.000	-2.86	0.002	-2.41	0.008	-1.96	0.025	-1.51	0.066	-1.06	0.145	-0.61	0.271	-0.16	0.436
-3.75	0.000	-3.3	0.000	-2.85	0.002	-2.40	0.008	-1.95	0.026	-1.5	0.067	-1.05	0.147	-0.6	0.274	-0.15	0.440
-3.74	0.000	-3.29	0.001	-2.84	0.002	-2.39	0.008	-1.94	0.026	-1.49	0.068	-1.04	0.149	-0.59	0.278	-0.14	0.444
-3.73	0.000	-3.28	0.001	-2.83	0.002	-2.38	0.009	-1.93	0.027	-1.48	0.069	-1.03	0.152	-0.58	0.281	-0.13	0.448
-3.72	0.000	-3.27	0.001	-2.82	0.002	-2.37	0.009	-1.92	0.027	-1.47	0.071	-1.02	0.154	-0.57	0.284	-0.12	0.452
-3.71	0.000	-3.26	0.001	-2.81	0.002	-2.36	0.009	-1.91	0.028	-1.46	0.072	-1.01	0.156	-0.56	0.288	-0.11	0.456
-3.7	0.000	-3.25	0.001	-2.8	0.003	-2.35	0.009	-1.9	0.029	-1.45	0.074	-1	0.159	-0.55	0.291	-0.1	0.460
-3.69	0.000	-3.24	0.001	-2.79	0.003	-2.34	0.010	-1.89	0.029	-1.44	0.075	-0.99	0.161	-0.54	0.295	-0.09	0.464
-3.68	0.000	-3.23	0.001	-2.78	0.003	-2.33	0.010	-1.88	0.030	-1.43	0.076	-0.98	0.164	-0.53	0.298	-0.08	0.468
-3.67	0.000	-3.22	0.001	-2.77	0.003	-2.32	0.010	-1.87	0.031	-1.42	0.078	-0.97	0.166	-0.52	0.302	-0.07	0.472
-3.66	0.000	-3.21	0.001	-2.76	0.003	-2.31	0.010	-1.86	0.031	-1.41	0.079	-0.96	0.169	-0.51	0.305	-0.06	0.476
-3.65	0.000	-3.2	0.001	-2.75	0.003	-2.30	0.011	-1.85	0.032	-1.4	0.081	-0.95	0.171	-0.5	0.309	-0.05	0.480
-3.64	0.000	-3.19	0.001	-2.74	0.003	-2.29	0.011	-1.84	0.033	-1.39	0.082	-0.94	0.174	-0.49	0.312	-0.04	0.484
-3.63	0.000	-3.18	0.001	-2.73	0.003	-2.28	0.011	-1.83	0.034	-1.38	0.084	-0.93	0.176	-0.48	0.316	-0.03	0.488
-3.62	0.000	-3.17	0.001	-2.72	0.003	-2.27	0.012	-1.82	0.034	-1.37	0.085	-0.92	0.179	-0.47	0.319	-0.02	0.492
-3.61	0.000	-3.16	0.001	-2.71	0.003	-2.26	0.012	-1.81	0.035	-1.36	0.087	-0.91	0.181	-0.46	0.323	-0.01	0.496
-3.6	0.000	-3.15	0.001	-2.7	0.003	-2.25	0.012	-1.8	0.036	-1.35	0.089	-0.9	0.184	-0.45	0.326	0	0.500
-3.59	0.000	-3.14	0.001	-2.69	0.004	-2.24	0.013	-1.79	0.037	-1.34	0.090	-0.89	0.187	-0.44	0.330		
-3.58	0.000	-3.13	0.001	-2.68	0.004	-2.23	0.013	-1.78	0.038	-1.33	0.092	-0.88	0.189	-0.43	0.334		
-3.57	0.000	-3.12	0.001	-2.67	0.004	-2.22	0.013	-1.77	0.038	-1.32	0.093	-0.87	0.192	-0.42	0.337		
-3.56	0.000	-3.11	0.001	-2.66	0.004	-2.21	0.014	-1.76	0.039	-1.31	0.095	-0.86	0.195	-0.41	0.341		

z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)	z	f(z)
4	1.000	3.55	1.000	3.1	0.999	2.65	0.996	2.2	0.986	1.75	0.960	1.3	0.903	0.85	0.802	0.4	0.655
3.99	1.000	3.54	1.000	3.09	0.999	2.64	0.996	2.19	0.986	1.74	0.959	1.29	0.901	0.84	0.800	0.39	0.652
3.98	1.000	3.53	1.000	3.08	0.999	2.63	0.996	2.18	0.985	1.73	0.958	1.28	0.900	0.83	0.797	0.38	0.648
3.97	1.000	3.52	1.000	3.07	0.999	2.62	0.996	2.17	0.985	1.72	0.957	1.27	0.898	0.82	0.794	0.37	0.644
3.96	1.000	3.51	1.000	3.06	0.999	2.61	0.995	2.16	0.985	1.71	0.956	1.26	0.896	0.81	0.791	0.36	0.641
3.95	1.000	3.5	1.000	3.05	0.999	2.60	0.995	2.15	0.984	1.7	0.955	1.25	0.894	0.8	0.788	0.35	0.637
3.94	1.000	3.49	1.000	3.04	0.999	2.59	0.995	2.14	0.984	1.69	0.954	1.24	0.893	0.79	0.785	0.34	0.633
3.93	1.000	3.48	1.000	3.03	0.999	2.58	0.995	2.13	0.983	1.68	0.954	1.23	0.891	0.78	0.782	0.33	0.629
3.92	1.000	3.47	1.000	3.02	0.999	2.57	0.995	2.12	0.983	1.67	0.953	1.22	0.889	0.77	0.779	0.32	0.626
3.91	1.000	3.46	1.000	3.01	0.999	2.56	0.995	2.11	0.983	1.66	0.952	1.21	0.887	0.76	0.776	0.31	0.622
3.9	1.000	3.45	1.000	3	0.999	2.55	0.995	2.1	0.982	1.65	0.951	1.2	0.885	0.75	0.773	0.3	0.618
3.89	1.000	3.44	1.000	2.99	0.999	2.54	0.994	2.09	0.982	1.64	0.949	1.19	0.883	0.74	0.770	0.29	0.614
3.88	1.000	3.43	1.000	2.98	0.999	2.53	0.994	2.08	0.981	1.63	0.948	1.18	0.881	0.73	0.767	0.28	0.610
3.87	1.000	3.42	1.000	2.97	0.999	2.52	0.994	2.07	0.981	1.62	0.947	1.17	0.879	0.72	0.764	0.27	0.606
3.86	1.000	3.41	1.000	2.96	0.998	2.51	0.994	2.06	0.980	1.61	0.946	1.16	0.877	0.71	0.761	0.26	0.603
3.85	1.000	3.4	1.000	2.95	0.998	2.50	0.994	2.05	0.980	1.6	0.945	1.15	0.875	0.7	0.758	0.25	0.599
3.84	1.000	3.39	1.000	2.94	0.998	2.49	0.994	2.04	0.979	1.59	0.944	1.14	0.873	0.69	0.755	0.24	0.595
3.83	1.000	3.38	1.000	2.93	0.998	2.48	0.993	2.03	0.979	1.58	0.943	1.13	0.871	0.68	0.752	0.23	0.591
3.82	1.000	3.37	1.000	2.92	0.998	2.47	0.993	2.02	0.978	1.57	0.942	1.12	0.869	0.67	0.749	0.22	0.587
3.81	1.000	3.36	1.000	2.91	0.998	2.46	0.993	2.01	0.978	1.56	0.941	1.11	0.867	0.66	0.745	0.21	0.583
3.8	1.000	3.35	1.000	2.9	0.998	2.45	0.993	2	0.977	1.55	0.939	1.1	0.864	0.65	0.742	0.2	0.579
3.79	1.000	3.34	1.000	2.89	0.998	2.44	0.993	1.99	0.977	1.54	0.938	1.09	0.862	0.64	0.739	0.19	0.575
3.78	1.000	3.33	1.000	2.88	0.998	2.43	0.992	1.98	0.976	1.53	0.937	1.08	0.860	0.63	0.736	0.18	0.571
3.77	1.000	3.32	1.000	2.87	0.998	2.42	0.992	1.97	0.976	1.52	0.936	1.07	0.858	0.62	0.732	0.17	0.567
3.76	1.000	3.31	1.000	2.86	0.998	2.41	0.992	1.96	0.975	1.51	0.934	1.06	0.855	0.61	0.729	0.16	0.564
3.75	1.000	3.3	1.000	2.85	0.998	2.40	0.992	1.95	0.974	1.5	0.933	1.05	0.853	0.6	0.726	0.15	0.560
3.74	1.000	3.29	0.999	2.84	0.998	2.39	0.992	1.94	0.974	1.49	0.932	1.04	0.851	0.59	0.722	0.14	0.556
3.73	1.000	3.28	0.999	2.83	0.998	2.38	0.991	1.93	0.973	1.48	0.931	1.03	0.848	0.58	0.719	0.13	0.552
3.72	1.000	3.27	0.999	2.82	0.998	2.37	0.991	1.92	0.973	1.47	0.929	1.02	0.846	0.57	0.716	0.12	0.548
3.71	1.000	3.26	0.999	2.81	0.998	2.36	0.991	1.91	0.972	1.46	0.928	1.01	0.844	0.56	0.712	0.11	0.544
3.7	1.000	3.25	0.999	2.8	0.997	2.35	0.991	1.9	0.971	1.45	0.926	1	0.841	0.55	0.709	0.1	0.540
3.69	1.000	3.24	0.999	2.79	0.997	2.34	0.990	1.89	0.971	1.44	0.925	0.99	0.839	0.54	0.705	0.09	0.536
3.68	1.000	3.23	0.999	2.78	0.997	2.33	0.990	1.88	0.970	1.43	0.924	0.98	0.836	0.53	0.702	0.08	0.532
3.67	1.000	3.22	0.999	2.77	0.997	2.32	0.990	1.87	0.969	1.42	0.922	0.97	0.834	0.52	0.698	0.07	0.528
3.66	1.000	3.21	0.999	2.76	0.997	2.31	0.990	1.86	0.969	1.41	0.921	0.96	0.831	0.51	0.695	0.06	0.524
3.65	1.000	3.2	0.999	2.75	0.997	2.30	0.989	1.85	0.968	1.4	0.919	0.95	0.829	0.5	0.691	0.05	0.520
3.64	1.000	3.19	0.999	2.74	0.997	2.29	0.989	1.84	0.967	1.39	0.918	0.94	0.826	0.49	0.688	0.04	0.516
3.63	1.000	3.18	0.999	2.73	0.997	2.28	0.989	1.83	0.966	1.38	0.916	0.93	0.824	0.48	0.684	0.03	0.512
3.62	1.000	3.17	0.999	2.72	0.997	2.27	0.988	1.82	0.966	1.37	0.915	0.92	0.821	0.47	0.681	0.02	0.508
3.61	1.000	3.16	0.999	2.71	0.997	2.26	0.988	1.81	0.965	1.36	0.913	0.91	0.819	0.46	0.677	0.01	0.504
3.6	1.000	3.15	0.999	2.7	0.997	2.25	0.988	1.8	0.964	1.35	0.911	0.9	0.816	0.45	0.674	0	0.500
3.59	1.000	3.14	0.999	2.69	0.996	2.24	0.987	1.79	0.963	1.34	0.910	0.89	0.813	0.44	0.670		
3.58	1.000	3.13	0.999	2.68	0.996	2.23	0.987	1.78	0.962	1.33	0.908	0.88	0.811	0.43	0.666		
3.57	1.000	3.12	0.999	2.67	0.996	2.22	0.987	1.77	0.962	1.32	0.907	0.87	0.808	0.42	0.663		
3.56	1.000	3.11	0.999	2.66	0.996	2.21	0.986	1.76	0.961	1.31	0.905	0.86	0.805	0.41	0.659		