

INFORME MANTENIMIENTO Y DESARROLLO ESPECIALIZADO SISTEMA DE INFORMACIÓN DE GESTIÓN DE PROYECTOS – SIGP

QUINTO PAGO CONTRATO 237-2025

TABLA DE CONTENIDO

INFORME MANTENIMIENTO Y DESARROLLO ESPECIALIZADO SISTEMA DE INFORMACIÓN DE GESTIÓN DE PROYECTOS - SIGP	1
7. Implementar una validación de puntajes y campos vacíos al finalizar la evaluación. Esta validación se activará al presionar el botón "Grabar". (No. Obligación Contrato: 2.2.1, 2.2.5, 2,2,6)	1
8. Implementar una funcionalidad en la sección de "Requisitos documentales" dentro de la configuración de la convocatoria que permita restringir el tipo de archivo a cargar (PDF, Excel, png, Word, etc. de acuerdo con la necesidad de la convocatoria). Esta restricción aplicará a los archivos que carguen los proponentes. (No. Obligación Contrato: 2.2.1, 2.2.5, 2,2,6)	8
9. En la etapa de subsanación de la propuesta, implementar una funcionalidad que conserve todos los documentos cargados inicialmente. Los documentos originales deben almacenarse y la opción de reemplazarlos debe bloquearse. (No. Obligación Contrato: 2.2.1, 2.2.5, 2,2,6)	15
INFORME DE ACTIVIDADES DE SOPORTE EN EL USO DEL SIGP EN LA CREACION DE CONVOCATORIAS Y TRANSFERIR EL CONOCIMIENTO DEL SIGP A LOS FUNCIONARIOS DE MINCIENCIAS, (No. Obligación Contrato: 2.2.3, 2.2.4)	31
Introducción	31
Resumen de Actividades	31
Detalle de Casos Atendidos.....	32
Solución de Errores	32
Requerimientos	34
Conclusiones.....	37

INFORME MANTENIMIENTO Y DESARROLLO ESPECIALIZADO SISTEMA DE INFORMACIÓN DE GESTIÓN DE PROYECTOS - SIGP

7. Implementar una validación de puntajes y campos vacíos al finalizar la evaluación. Esta validación se activará al presionar el botón "Grabar". (No. Obligación Contrato: 2.2.1, 2.2.5, 2,2,6)

Solicitud de requerimiento No. 159827

Descripción y alcance del Requerimiento

Se requiere implementar una validación integral al momento de grabar una evaluación. Esta validación debe activarse al presionar el botón "Grabar" y deberá verificar lo siguiente:

1. Validación de puntajes:
 - Asegurar que todos los criterios de evaluación tengan un puntaje asignado.
 - Verificar que el puntaje esté dentro del rango permitido (por ejemplo, 0 a 5 o según configuración de cada criterio).
 - Impedir el guardado si existe al menos un puntaje fuera de rango o vacío.
2. Validación de campos obligatorios:
 - Verificar que se haya diligenciado el campo de observaciones cuando el puntaje de algún criterio sea inferior al mínimo esperado (ej. menor a 3).
 - Validar que todos los campos marcados como obligatorios en la interfaz del evaluador (como selección de decisión, tipo de evaluación, u observaciones generales) estén correctamente completados.
3. Mensajes de error:
 - Mostrar mensajes de error claros y específicos, por ejemplo:
"Debe asignar un puntaje para el criterio X."
"El campo de observaciones es obligatorio cuando el puntaje es menor a 3."
"Faltan campos obligatorios por completar: Tipo de evaluación."
4. Bloqueo de grabación:

- Si alguna validación falla, el sistema no debe permitir el guardado de la evaluación hasta que todos los errores sean corregidos.

Desarrollo e Implementación

Este requerimiento se divide en dos desarrollos:

1. Se buscó implementar en el módulo de evaluación de proyectos y programas del aplicativo de gestión lógica de validación al momento de hacer clic en el botón de grabar Evaluación.
2. Recalcular los puntajes de evaluación una vez se califique cada criterio de calificación, con el fin de evitar errores de sumatorias de puntajes

1. Contexto y Alcance

El objetivo fue garantizar que, al presionar el botón **"Grabar"** en el módulo de Evaluación de Proyectos y Programas, se validara de forma integral que:

- Todos los criterios de evaluación cuenten con puntaje válido.
- Los campos obligatorios estén diligenciados.
- No existan registros incompletos que puedan generar inconsistencias en la base de datos.
- Los puntajes totales se recalculen automáticamente para mantener la integridad de la información.

2. Estructura de Validación

2.1 Validación de Calificación de Criterios

Se implementó un flujo de validación desde el controlador hasta la capa de persistencia. En el **botón de guardar/editar** se invoca el método:

```
verificarCamposVacios()
```

Este método:

1. Verifica que el campo **Concepto** no sea nulo o vacío.
2. Comprueba que todos los criterios de evaluación tengan calificación asignada.
3. Aplica la validación de cantidad mínima y máxima de palabras en el concepto, según parámetro configurado.

Fragmento de implementación en el controlador:

```
//Verificar Cantidad Mínima y Máxima de palabras en el concepto
if (proyectoEvaluacionDTO.getConcepto() != null &&
!proyectoEvaluacionDTO.getConcepto().isEmpty()) {

    //Verificar campos vacíos evaluación
    if (proyectoEvaluacionDTO.getViable() != null) {
        proyectosServiceFacade.verificarCamposVacios(proyectoEvaluacionDTO);
    }

    // Validamos que el concepto final tenga como minimo 60 palabras o cumpla con la
    configuración
    validarCantidadPalabrasConceptoFinal(proyectoEvaluacionDTO.getConcepto());
}
}
```

2.2 Flujo en capas de negocio

- **Facade EJB:** (proyectosServiceFacade.java)

Encapsula la llamada al servicio de negocio y maneja la conversión DTO → Entity.

```
public void verificarCamposVacios(ProyectoEvaluacionDTO dto) throws Exception {
    ObjectConverter<ProyectoEvaluacionDTO, ProyectoEvaluacion> converter =
    objectConverterFactory.checkOut();
    proyectosService.verificarCamposVacios(converter.toEntity(dto,
    ProyectoEvaluacion.class));
    objectConverterFactory.checkIn(converter);
}
```

- **Service EJB:** (proyectosServiceBean.java)

Llama al StoreManager para obtener el conteo de campos pendientes de calificación.

```
@Override
public void verificarCamposVacios(ProyectoEvaluacion proyectoEvaluacion) throws
SystemException {
    if (proyectoEvaluacion != null) {
        if (proyectoEvaluacion.getProyectoId() != null &&
proyectoEvaluacion.getEvaluacionProyectoId() != null) {
            int numberMissingEvaluation =
this.getStoreManager(ProyectosStoreManager.class)
            .validarCamposPorEvaluar(proyectoEvaluacion);
            if (numberMissingEvaluation != 0) {
                SystemException error = new
                SystemException("#{msg['gestion.proyectosProgramas.evaluadores.error.camposSinEvaluar']}");
            }
        }
    }
}
```

```
error.getParameters().add(String.valueOf(numberMissingEvaluation));
    throw error;
}
}
}
```

- **Persistence Layer:** (ProyectosStoreManager.java)

Consulta nativa que devuelve la cantidad de criterios sin calificación:

```
@Override
public int validarCamposPorEvaluar(ProyectoEvaluacion proyectoEvaluacion) {
    Query query = em.createNativeQuery("SELECT COUNT(*)\n"
        + " FROM PRY_EVALUACIONES_CALIFICACION p\n"
        + " WHERE p.proyecto_id = ?\n"
        + "       AND p.evaluacion_proyecto_id = ?\n"
        + "       AND p.calificacion IS NULL\n"
        + "       AND p.entrabable = 'S' ");
    query.setParameter(1, proyectoEvaluacion.getProyectoId().getProyectoId());
    query.setParameter(2, proyectoEvaluacion.getEvaluacionProyectoId());
    Object o = query.getSingleResult();
    int result = 0;
    if (o instanceof BigDecimal) {
        return ((BigDecimal) o).intValue();
    } else if (o instanceof Long) {
        return ((Long) o).intValue(); // Correcto: obtienes el valor de Long y lo
        convierte a int
    }
    return result;
}
```

2.3 Mensajes de error y bloqueo

Si la validación detecta inconsistencias, se bloquea la grabación y se muestra un mensaje parametrizado:

Español

```
gestion.proyectosProgramas.evaluadores.error.camposSinEvaluar = No puede
finalizar la evaluación sin antes completar todos los campos de la evaluación, actualmente
tiene {0} criterio(s) de evaluación pendiente(s) por evaluar
```

Ingles

```
gestion.proyectosProgramas.evaluadores.error.camposSinEvaluar = You cannot
complete the evaluation without first completing all evaluation fields, you currently have
{0} evaluation criteria pending to be evaluated
```

3. Recalculo Automático de Puntajes

3.1 En el Controlador

En el método `editarConceptoEvaluacion()` se agregó la llamada a:

```
recalcularPuntajesEvaluacion(proyectoEvaluacionDTO);
```

Este recalcule se ejecuta tanto al guardar exitosamente como en caso de excepción, evitando inconsistencias.

Código relevante:

```
public void recalcularPuntajesEvaluacion(ProyectoEvaluacionDTO dto) {  
    proyectoServiceFacade.callSBProyectosCalcularPuntaje(dto.getEvaluacionProyectoId());  
    ProyectoEvaluacionDTO evaluacionDB =  
    proyectoServiceFacade.obtenerEvaluacion(dto.getEvaluacionProyectoId(),  
    dto.getProyectoId().getProyectoId());  
    dto.setPuntaje(evaluacionDB.getPuntaje());  
}
```

3.2 Procedimiento de Base de Datos

Se llama al procedimiento:

```
CALL SB_PROYECTOS.calcular_calificacion(?)
```

Encargado de recalcular las sumatorias de calificación y persistir el puntaje actualizado.

3.3 Validaciones adicionales en `editarConceptoProyectoEvaluacion`

- Verificar puntaje mínimo elegible según la configuración de la convocatoria/mecanismo/temática.
- Validar que el concepto final haya sido ingresado.
- Confirmar fechas de inicio y recepción de la evaluación para calcular la duración.

Fragmento:

```
@Override
public void editarConceptoProyectoEvaluacion(ProyectoEvaluacion proyectoEvaluacion) throws
SystemException {
    Double totalCalificacion;
    Double puntajeMinimoElegible = 0.0;
    /**
     * Verificamos si es una convocatoria de mecanismos de participación o
     * es una convocatoria normal
     */
    if (proyectoEvaluacion.getProyectoId().getConvocatoriaMecanismoId() == null) {
        //Verificar que la convocatoria tenga configurado puntaje minimo elegible
        if (proyectoEvaluacion.getProyectoId().getConvocatoriaId().getPuntajeElegible()
!= null) {
            puntajeMinimoElegible =
proyectoEvaluacion.getProyectoId().getConvocatoriaId().getPuntajeElegible().doubleValue();
        } else {
            throw new
SystemException("#{msg['gestion.proyectosProgramas.evaluadores.error.nullPuntajeMinElejible'
]}");
        }
    } else //Verificar que el mecanismo de participacion o tematica tenga configurado
puntaje minimo elegible
    {
        if (proyectoEvaluacion.getProyectoId().getConvocatoriaMecanismoId() != null) {
            //Verificamos si el puntaje de la matriz esta configurado por tematica
            if (proyectoEvaluacion.getProyectoId().getMecanismoTematicaId() != null
&&
proyectoEvaluacion.getProyectoId().getMecanismoTematicaId().getPuntajeElegible() != null) {
                puntajeMinimoElegible =
proyectoEvaluacion.getProyectoId().getMecanismoTematicaId().getPuntajeElegible().doubleValue
();
                //Verificamos si el puntaje de la matriz esta configurado por
mecanismo
            } else if
(proyectoEvaluacion.getProyectoId().getConvocatoriaMecanismoId().getPuntajeElegible() !=
null) {
                puntajeMinimoElegible =
proyectoEvaluacion.getProyectoId().getConvocatoriaMecanismoId().getPuntajeElegible().doubleV
alue();
            } else {
                throw new
SystemException("#{msg['gestion.proyectosProgramas.evaluadores.error.nullPuntajeMinElejible.
mecanismo']}");
            }
        }
    }

    /**
     * Verificar que el Puntaje de la Evaluación no sea nulo, en ese caso el evaluador debiera
     * realizar primero la calificacion de la Evaluación antes de dar el concepto de la misma
     */
    if (proyectoEvaluacion.getPuntaje() == null) {
        throw new
SystemException("#{msg['gestion.proyectosProgramas.evaluadores.error.conceptoEvaluacion']}");
    }
    else {
        // Recalculamos los puntajes de las evaluaciones y seteamos puntaje actualizado
        proyectoEvaluacion.setPuntaje(recalcularPuntajesTotalesEvaluacion(proyectoEvaluacion));
    }
}
```

```

        totalCalificacion = proyectoEvaluacion.getPuntaje().doubleValue();
        /* Verificar que el total de la Calificación de la Evaluación no sea menor que
        el puntaje Mínimo Elegible en la Convocatoria
        y cuya recomendación final sea positiva*/
        if (totalCalificacion < puntajeMinimoElegible && proyectoEvaluacion.getViable()
        != null && proyectoEvaluacion.getViable().equals("S")) {
            SystemException error = new
            SystemException("#{msg['gestion.proyectosProgramas.evaluadores.error.puntajeMinElejible']}");
        ;
            error.getParameters().add(puntajeMinimoElegible.toString());
            throw error;
        }
    }

    //Verificar que el concepto final ha sido ingresado
    if (proyectoEvaluacion.getConcepto() == null ||
    proyectoEvaluacion.getConcepto().equals("")) {
        throw new
        SystemException("#{msg['gestion.proyectosProgramas.evaluadores.conceptoFinal.required']}");
    }

    /* Verificar que la fecha de asignacion del Evaluador y la fecha de Evaluacion hayan
    sido ingresadas
    en el concepto de evaluacion */
    if (proyectoEvaluacion.getFechaInicioEvaluacion() != null &&
    proyectoEvaluacion.getFechaRecepcionEvaluacion() != null) {

        proyectoEvaluacion.setDuracionEvaluacion(obtenerCantidadDias(proyectoEvaluacion.getFechaInicioEvaluacion(), proyectoEvaluacion.getFechaRecepcionEvaluacion()));
        asignarRecomendacionEvaluador(proyectoEvaluacion);
        this.modificar(proyectoEvaluacion, ProyectoEvaluacion.class);
    } else {
        throw new
        SystemException("#{msg['gestion.proyectosProgramas.evaluadores.error.fechaEvaluacion']}");
    }
}

```

```

/**
 * Metodo para recalcular el puntaje total de la evaluacion y calificaciones
 *
 * @param proyectoEvaluacion
 * @return
 * @throws SystemException
 */
public BigDecimal recalcularPuntajesTotalesEvaluacion(ProyectoEvaluacion
proyectoEvaluacion) throws SystemException {
    // Recalculamos los puntajes de las evaluaciones
    this.callSBProyectosCalcularPuntaje(proyectoEvaluacion.getEvaluacionProyectoId());

    // Consultamos y seteamos puntaje actualizado
    ProyectoEvaluacion evaluacionDB =
    this.obtenerEvaluacion(proyectoEvaluacion.getEvaluacionProyectoId(),
    proyectoEvaluacion.getProyectoId().getProyectoId());

    return evaluacionDB.getPuntaje();
}

```

- Con lo anterior, ya estamos recalculando los puntajes desde diferentes lugares del código, evitando incongruencias en las evaluaciones

4. Resultados de la Implementación

- Bloqueo total de grabación si existen criterios sin calificar o campos obligatorios vacíos.
- Mensajes de error específicos y traducidos (ES/EN).
- Recalculo automático de puntajes en múltiples puntos del flujo para prevenir desajustes.
- Código desacoplado por capas (Controlador → Facade → Service → Persistence).
- Validaciones configurables vía parámetros del sistema (ej. mínimo/máximo de palabras).

8. Implementar una funcionalidad en la sección de "Requisitos documentales" dentro de la configuración de la convocatoria que permita restringir el tipo de archivo a cargar (PDF, Excel, png, Word, etc. de acuerdo con la necesidad de la convocatoria). Esta restricción aplicará a los archivos que carguen los proponentes. (No. Obligación Contrato: 2.2.1, 2.2.5, 2,2,6)

Solicitud de requerimiento No. 159835

Descripción y alcance del Requerimiento

Se requiere implementar una funcionalidad dentro de la sección "Requisitos documentales" en la configuración de convocatorias del SIGP, que permita a los usuarios administradores (ej. Coordinadores o Técnicos OTSI) definir y restringir el tipo de archivo que puede ser cargado por los proponentes para cada requisito solicitado.

La funcionalidad deberá:

1. Permitir seleccionar, por cada requisito documental, los tipos de archivos admitidos durante la configuración de la convocatoria. Ejemplos:

- .pdf (Adobe Acrobat)
 - .doc / .docx (Word)
 - .xls / .xlsx (Excel)
 - .png, .jpg, .jpeg (imágenes)
 - .zip, .rar (archivos comprimidos), si es necesario
2. Validar en tiempo real o al momento del cargue, que los archivos subidos por los proponentes correspondan al tipo permitido para ese requisito específico. En caso contrario:
- Bloquear la carga
 - Mostrar un mensaje de error informativo y específico (por ejemplo: “Solo se permite cargar archivos en formato PDF para este requisito”)
3. Respetar las configuraciones por convocatoria y por requisito, es decir, permitir que distintas convocatorias o incluso distintos requisitos dentro de una misma convocatoria tengan diferentes configuraciones de tipo de archivo admitido.
4. Incluir controles en la interfaz administrativa, tales como:
- Menú desplegable o casillas de verificación para seleccionar los formatos admitidos.
 - Vista previa o resumen de los formatos habilitados por cada requisito documental.
5. Registrar en bitácora o log del sistema el tipo de archivo cargado y validado por cada usuario, para efectos de trazabilidad y auditoría.

Desarrollo e Implementación

Se desarrolló una funcionalidad en el **Sistema de Información de Gestión de Proyectos (SIGP)** que permite a usuarios administradores configurar los tipos de archivos permitidos para cada requisito documental en la configuración de convocatorias.

La validación se aplica al momento de que el proponente intenta cargar un archivo, bloqueando el cargue si la extensión no coincide con la configuración establecida para ese requisito.

1. Desarrollo e Implementación

El desarrollo se realizó en tres frentes:

1.1. Cambios en Base de Datos

1. **Nuevo campo en la tabla SGP_CONVOCATORIAS_REQUISITOS** para almacenar las extensiones permitidas:

```
-- Add/modify columns
alter table SGP_CONVOCATORIAS_REQUISITOS add formatos_permitidos VARCHAR2(150);
-- Add comments to the columns
comment on column SGP_CONVOCATORIAS_REQUISITOS.formatos_permitidos
is 'Indica los tipos de extensiones/formatos permitidos para el documento asociado al
requisito';
```

2. **Carga de datos dinámicos** para lista de extensiones mediante ADM_TIPOS_DEFINICIONES y ADM_VALORES_DEFINICIONES, permitiendo administrar los formatos disponibles sin cambios de código.

Ejemplo de inserción para formato PDF:

- Tipos definiciones

```
INSERT INTO adm_tipos_definiciones (
    TIPO_DEFINICION,
    DESCRIPCION_TIPO,
    CLASE_DEFINICION,
    ACTIVO,
    REQUIERE_VALIDACION,
    TIPO_DATO,
    TAMANO,
    DECIMALES,
    USUARIO_CREADOR,
    FECHA_CREACION,
    USUARIO_MODIFICADOR,
    FECHA_MODIFICACION
)
VALUES (
    'EXTREQCONV',
    'Extensiones de archivos permitidos a configurar por requisitos de convocatoria',
    'S',
    'S',
    'N',
    'C',
    NULL,
    NULL,
    'SIGP_WEB',
    SYSDATE,
    NULL,
    NULL
);
COMMIT;
```

- Valores Definiciones

```
-- PDF(.pdf)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 10, 'PDF', 'S', 'CHARACTER', NULL,
NULL, '.pdf', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL, NULL, NULL,
NULL);

-- WORD(.docx, .doc)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 20, 'WORD', 'S', 'CHARACTER', NULL,
NULL, '.docx.doc', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL, NULL, NULL,
NULL, NULL);

-- EXCEL (.xlsx, .xls)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 30, 'EXCEL', 'S', 'CHARACTER', NULL,
NULL, '.xlsx.xls', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL, NULL, NULL,
NULL, NULL);

-- TEXTO (.txt)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 40, 'TEXTO', 'S', 'CHARACTER', NULL,
NULL, '.txt', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL, NULL, NULL,
NULL);

-- IMÁGENES(.png)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
```

```
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 50, 'IMÁGENES', 'S', 'CARACTER', NULL,
NULL, '.png,.jpeg', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL, NULL,
NULL, NULL);

-- ARCHIVOS COMPRIMIDOS(.zip)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 60, 'ARCHIVOS COMPRIMIDOS', 'S',
'CARACTER', NULL, NULL, '.zip', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL,
NULL, NULL, NULL, NULL);

-- POWER POINT(.pptx,.ppt)
INSERT INTO adm_valores_definiciones
(valor_definicion_id, tipo_definicion, valor_definicion, descripcion_valor_definicion,
activo, tipo_dato, tamano, decimales,
comentario, obligatorio, tipo_validacion, usuario_creador, fecha_creacion,
usuario_modificador, fecha_modificacion, orden,
valor_pond_i, valor_pond_d, valor_pond_a, valor_pond_b, valor_pond_s)
VALUES
(ADM_VALOR_DEFINICION_ID_SEQ.Nextval, 'EXTREQCONV', 70, 'POWER POINT', 'S', 'CARACTER',
NULL, NULL, '.pptx,.ppt', 'N', NULL, 'SIGP_WEB', SYSDATE, NULL, NULL, NULL, NULL, NULL,
NULL, NULL, NULL);

COMMIT;
```

Se cargaron registros para PDF, Word, Excel, Texto, Imágenes, Archivos Comprimidos y PowerPoint.

1.2. Cambios en Código (Back-End)

1. Mapeo de nuevo campo en la entidad ConvocatoriaRequisito.java y su respectivo DTO ConvocatoriaRequisitoDTO.java:

- Entidad

```
@Column(name = "FORMATOS_PERMITIDOS")
@Size(max = 150)
private String formatosPermitidos;
// getters y setters

public String getFormatosPermitidos() {
    return formatosPermitidos;
}

public void setFormatosPermitidos(String formatosPermitidos) {
    this.formatosPermitidos = formatosPermitidos;
}
```

- DTO (Data Transfer Object)

```
@Bind(name = "formatosPermitidos")
private String formatosPermitidos;
// getters y setters
public String getFormatosPermitidos() {
    return formatosPermitidos;
}

public void setFormatosPermitidos(String formatosPermitidos) {
    this.formatosPermitidos = formatosPermitidos;
}
```

2. **Campo auxiliar (extensionsList)** en el DTO para manejo de lista de formatos:

```
private List<ValorDefinicionDTO> extensionsList;

public List<ValorDefinicionDTO> getExtensionsList() {
    return extensionsList;
}

public void setExtensionsList(List<ValorDefinicionDTO> extensionsList) {
    this.extensionsList = extensionsList;
}
```

3. **Obtención dinámica** de la lista de extensiones desde configApiController:

```
extensionsList = new
ArrayList<>(configApiController.getMapValorDefinicion().get("EXTREQCONV").values())
    .stream()
    .sorted(Comparator.comparing(ValorDefinicionDTO::getValorDefinicion))
    .collect(Collectors.toList());
```

4. **Lógica de guardado** para setear los formatos seleccionados:

```
if (convocatoriaRequisitoDTO.getExtensionsList() != null &&
!convocatoriaRequisitoDTO.getExtensionsList().isEmpty()) {
    String extensions =
formatAllowedExtensions(convocatoriaRequisitoDTO.getExtensionsList());
    convocatoriaRequisitoDTO.setFormatosPermitidos(extensions);
}
```

1.3. Cambios en Código (Front-End)

1. **Formulario de configuración de requisitos (requisitos.xhtml)**

Se agregó un p:selectCheckboxMenu de PrimeFaces para seleccionar formatos permitidos:

```
<p:selectCheckboxMenu id="extPermitidas"
```

```
value="#{convocatoriaRequisitoViewController.selected.extensionsList}"
                                label="#{msg['sb.selectOneMenu.registro']}"
                                style="width: 100%"
                                multiple="true"
                                filter="false"
                                required="true"

requiredMessage="#{msg['gestion.configuracion.convocatoria.requisitos.formatosPermitidos.required']}"
                                panelStyle="width: 85rem"
                                converter="valorDefinicionConverter"
                                updateLabel="true">
    <f:selectItems value="#{convocatoriaRequisitoViewController.extensionsList}"
                var="ext"
                itemLabel="#{ext.descripcionValorDefinicion}
({ext.comentario})"
                itemValue="#{ext}"/>
</p:selectCheckboxMenu>
```

2. Validación en componentes de carga de archivos (fileUpload) en el Formulario de Proyectos y Programas y Beneficios Tributarios:

```
accept="#{
requisitoProyectoSessController.requisitoSeleccionado.convocatoriaRequisitoDTO.formatosPermitidos ne null
?
requisitoProyectoSessController.requisitoSeleccionado.convocatoriaRequisitoDTO.formatosPermitidos
: '.pdf,.docx,.doc,.xlsx,.xls,.txt,.png,.jpeg,.zip,.pptx,.ppt'
}"
allowTypes="#{
    requisitoProyectoSessController.getAllowTypesAttrReq(
requisitoProyectoSessController.requisitoSeleccionado.convocatoriaRequisitoDTO.formatosPermitidos)
}"
```

3. Método para generar expresión regular de validación:

```
public String getAllowTypesAttrReq(String allowedExtensions) {
    if (allowedExtensions != null && !allowedExtensions.isEmpty()) {
        String allowTypes = "/(\\.|\\\\)(\\{0}\\)$/";
        String subString = allowedExtensions.replace(",", "|").replace(".", "");
        return allowTypes.replace("{0}", subString);
    } else {
        return "/(\\.|\\\\)(pdf|docx|doc|xlsx|xls|txt|png|jpeg|zip|pptx|ppt)$/";
    }
}
```

2. Internacionalización

Se agregaron cadenas en **ES** y **EN** para etiquetas y mensajes dinámicos:

Español

```
gestion.configuracion.convocatoria.requisitos.formatosPermitidos.label = Formatos Permitidos  
sb.mensajes.upload.invalidFile.dinamic = Formato de archivo no permitido. Solo se aceptan  
archivos con las siguientes extensiones:
```

Inglés

```
gestion.configuracion.convocatoria.requisitos.formatosPermitidos.label = Allowed Formats  
sb.mensajes.upload.invalidFile.dinamic = File format not allowed. Only files with the  
following extensions are accepted:
```

3. Resumen del Impacto

- **Base de datos:** nueva columna en SGP_CONVOCATORIAS_REQUISITOS, inserción de catálogos de formatos en adm_valores_definiciones.
- **Back-End:** mapeo de campo, lógica de guardado, carga dinámica de valores.
- **Front-End:** nuevos controles en configuración y validación dinámica en carga de archivos.
- **Internacionalización:** mensajes en dos idiomas para interfaz y validaciones.

9. En la etapa de subsanación de la propuesta, implementar una funcionalidad que conserve todos los documentos cargados inicialmente. Los documentos originales deben almacenarse y la opción de reemplazarlos debe bloquearse. (No. Obligación Contrato: 2.2.1, 2.2.5, 2,2,6)

Solicitud de requerimiento No. 159836

Descripción y alcance del Requerimiento

Durante la etapa de subsanación de propuestas dentro del proceso de evaluación, se requiere implementar una funcionalidad que garantice la conservación de todos los documentos originales cargados por el proponente durante la fase inicial de radicación.

Especificaciones técnicas del requerimiento:

1. Almacenamiento permanente de documentos originales:
 - Todos los archivos adjuntados en la etapa inicial de presentación de la propuesta deben ser almacenados en un repositorio seguro, con marcación de fecha, hora y usuario responsable de la carga.
 - Estos documentos deben permanecer inmutables, es decir, no deben ser sobrescritos ni eliminados en ninguna etapa posterior del proceso.
2. Bloqueo de reemplazo durante la subsanación:
 - En la etapa de subsanación, se debe deshabilitar la opción de eliminar o reemplazar los documentos cargados originalmente.
 - Solo se permitirá cargar nuevos documentos adicionales, que serán etiquetados como “documento de subsanación”.
3. Trazabilidad y control de versiones:
 - Los documentos nuevos deben conservar información de trazabilidad: usuario que carga, fecha y motivo de la subsanación.
 - La interfaz debe diferenciar claramente entre los documentos originales y los documentos de subsanación.
4. Visibilidad y auditoría:
 - Tanto evaluadores como responsables administrativos deben poder visualizar ambas versiones: los documentos originales y los agregados en subsanación.
 - Se debe generar un reporte descargable que liste los documentos originales, los nuevos adjuntos y la metadata asociada (fechas, usuario, descripción).

Desarrollo e Implementación

1. Aplicativo de Gestión

Se integró una nueva funcionalidad en el **Aplicativo de Gestión** para administrar el proceso de subsanación de documentos asociados a requisitos de convocatorias, manteniendo la trazabilidad y protegiendo los documentos originales.

1.1. Ajustes en la interfaz de usuario (XHTML)

En las vistas requisitos.xhtml de los módulos **Proyectos y Programas** y **Beneficios Tributarios**, se realizaron las siguientes modificaciones:

- **Nueva columna "Estado de Subsanación"**

Implementada mediante un <p:column> con badges dinámicos en función del valor del campo estadoSubsanacion (N, P, R, S, X), usando styleClass condicional para asignar colores y estilos.

```
<p:column width="10%" style="text-align: center;">
  <f:facet name="header">
    <h:outputText
value="#{msg['gestion.beneficiosTributarios.requisitos.estadoSubsanacion.label']}" />
  </f:facet>
  <h:outputText class="badge badge-#{item.requisitoDTO.estadoSubsanacion eq 'N' ? 'info' :
item.requisitoDTO.estadoSubsanacion eq 'P' ? 'warning' :
item.requisitoDTO.estadoSubsanacion eq 'R' ? 'success' :
item.requisitoDTO.estadoSubsanacion eq 'S' ? 'primary' :
item.requisitoDTO.estadoSubsanacion eq 'X' ? 'danger' :
'info'}"
value="#{item.requisitoDTO.estadoSubsanacion eq 'N' ?
msg['listaEstadoSubsanacion.N'] :
item.requisitoDTO.estadoSubsanacion eq 'P' ?
msg['listaEstadoSubsanacion.P'] :
item.requisitoDTO.estadoSubsanacion eq 'R' ?
msg['listaEstadoSubsanacion.R'] :
item.requisitoDTO.estadoSubsanacion eq 'S' ?
msg['listaEstadoSubsanacion.S'] :
item.requisitoDTO.estadoSubsanacion eq 'X' ?
msg['listaEstadoSubsanacion.X'] :
msg['listaEstadoSubsanacion.N']}"
/>
</p:column>
```

- **Botones de acción condicionales**

Se agregaron botones para:

- **Subsanar** (p:commandButton → método subsanarDocumento()).

```
<p:commandLink id="subsanarDocButton"
style="color: #fff"

actionListener="#{requisitoProyectoConvController.verifyAttachedDocument(item)}"
update="dlgSubsanar"
rendered="#{requisitoProyectoSessController.renderedSubsanarButton(item)}"
title="#{msg['gestion.beneficiosTributarios.requisitos.subsanar.title']}"
styleClass="btn btn-xs btn-primary"
```

```

        process="@this"
    >
    <i class="fa fa-clipboard"></i>
    <f:setPropertyActionListener value="#{item.documentoDTO}"
target="#{requisitoProyectoSessController.documentoSelected}" />
    <f:setPropertyActionListener value="#{item}"
target="#{requisitoProyectoSessController.requisitoSeleccionado}" />
</p:commandLink>

```

- **Historial de documentos** (p:commandButton → método cargarHistorialDocs()).

```

<p:commandLink style="color: #fff;"
update="historialDocForm"
rendered="#{!SeguridadSessController.noEsUsuarioRegistro()
and item.documentoDTO.documentoId ne null
and
proyectoBeneficiosSessController.selected.estadoProyectoId.descripcionValorDefinicion ne
'DILIGENCIANDO'}"
process="@this"

title="#{msg['gestion.beneficiosTributarios.requisitos.historialDoc.button.title']}"
styleClass="btn btn-xs btn-primary"

actionListener="#{requisitoProyectoSessController.cargarHistorialDocs(item.documentoDTO)}"
oncomplete="$('#dlgHistorialDoc').modal()"
>
    <i class="fa fa-history"></i>
    <f:setPropertyActionListener value="#{item}"
target="#{requisitoProyectoSessController.requisitoSeleccionado}" />
</p:commandLink>

```

- **Revisar** (p:commandButton → método reviewDocument()).

```

<p:commandLink id="revisionDocButton"
style="color: #fff"
rendered="#{item.requisitoDTO.estadoSubsanacion eq 'R'
or item.requisitoDTO.estadoSubsanacion eq 'P'}"
actionListener="#{requisitoProyectoConvController.onRequisitoSelect(item)}"
update="dlgRevision"

title="#{msg['gestion.beneficiosTributarios.requisitos.revisarDoc.button.title']}"
styleClass="btn btn-xs btn-primary"
process="@this"
oncomplete="$('#revision').modal();"
>
    <i class="fa fa-sliders"></i>
    <f:setPropertyActionListener value="#{item.documentoDTO}"
target="#{requisitoProyectoSessController.documentoSelected}" />
    <f:setPropertyActionListener value="#{item}"
target="#{requisitoProyectoSessController.requisitoSeleccionado}" />
</p:commandLink>

```

Estos botones utilizan el atributo rendered con expresiones EL que verifican:

- Rol del usuario (esEvaluador, esAdmin).
- Estado del requisito (estadoSubsanacion).

- Existencia de documento cargado (verifyAttachedDocument()).
- **Diálogos modales (PrimeFaces)**
Se incluyeron tres diálogos controlados con widgetVar:
 - dlgSubsanar → permite adjuntar documento de subsanación y observaciones.
 - dlgRevision → panel de aprobación/rechazo.
 - dlgHistorialDoc → tabla que lista versiones históricas con opción de descarga.

1.2. Lógica en controladores

Los controladores principales (RequisitoProyectoSessController, RequisitoProyectoConvController y RequisitoSessController) fueron ajustados para gestionar:

- **Limpieza de observaciones si el documento no es subsanable**

```
public void cleanObservacionDocumento(DocumentoDTO documento) {  
    if (documento != null && "N".equals(documento.getSubsanable())) {  
        documento.setObservacion(null);  
    }  
}
```

Esto asegura que si el campo subsanable es N (No subsanable), no queden observaciones residuales.

- **Carga del historial de documentos**

```
public void cargarHistorialDocs() {  
    this.historialDocumentos = beneficiosServiceFacade.getDocumentsHistory(  
        requisitoSeleccionado.getConvocatoriaRequisitoId(),  
        proyectoSeleccionado.getProyectoId()  
    );  
}
```

La información se obtiene a través del *facade* para aislar la capa de presentación de la lógica de acceso a datos.

- **Control de visibilidad de botones**

```
public boolean renderedSubsanarButton(RequisitoDTO req) {  
    return req.isSubsanable() && esUsuarioPermitido() && !req.tieneDocumentoPendiente();  
}
```

Esta validación combina reglas de negocio y permisos.

- **Validación previa a la subsanación**

```
public boolean verifyAttachedDocument(RequisitoDTO req) {  
    return req.getDocumentoId() != null && req.getDocumentoId() > 0;  
}
```

Evita la apertura del diálogo de subsanación sin un documento base.

- **Registro de subsanación**

```
public void subsanarDocumento() {  
    documentoSeleccionado.setFechaRegistraSubsanable(new Date());  
    documentoSeleccionado.setUsuarioRegistraSubsanable(usuarioLogueado);  
    proyectoServiceFacade.modificarDocumento(documentoSeleccionado);  
}
```

Actualiza metadatos de trazabilidad y persiste cambios.

- **Descarga desde el historial**

```
public void descargarArchivoFromHistoryDialog(Long documentoId) {  
    DocumentoDTO doc = proyectoServiceFacade.getDocumentoById(documentoId);  
    streamArchivo(doc.getRutaArchivo(), doc.getNombreArchivo());  
}
```

Permite a los usuarios descargar versiones anteriores.

1.3. Lógica en la capa EJB

En ProyectosServiceBean:

- **Consulta del historial de documentos con JPA**

```
@Override  
public List<Documento> getDocumentsHistory(Long requisitoConvocatoriaId, Long proyectoId) {  
    String where = (requisitoConvocatoriaId == -1L)  
        ? " WHERE d.convocatoriaRequisitoId IS NULL "  
        : " WHERE d.convocatoriaRequisitoId.convocatoriaRequisitoId =  
:convocatoriaRequisitoId ";  
  
    TypedQuery<Documento> query = em.createQuery(  
        "SELECT d FROM Documento d " + where +  
        " AND d.origenId = :proyectoId ORDER BY d.documentoId DESC", Documento.class  
    );  
  
    query.setParameter("proyectoId", proyectoId);  
    if (requisitoConvocatoriaId != -1L) {  
        query.setParameter("convocatoriaRequisitoId", requisitoConvocatoriaId);  
    }  
    return query.getResultList();  
}
```

Esta consulta retorna todas las versiones históricas de documentos asociadas a un requisito y un proyecto, ordenadas por el documento más reciente.

- **Actualización de documentos**

```
@Override
public void modificarDocumento(Documento documento) {
    documento.setFechaModificacion(new Date());
    documento.setUsuarioModificador(usuarioActual());
    em.merge(documento);
}
```

Se asegura el registro de auditoría en cada modificación.

1.4. Cambios en base de datos

- Uso de tablas existentes:
 - PRY_REQUISITOS → información de requisitos del proyecto. Se crea nuevo campo para registrar el estado de subsanación del requisito.

```
-- Add/modify columns
alter table PRY_REQUISITOS add estado_subsanacion VARCHAR2(1);
-- Add comments to the columns
comment on column PRY_REQUISITOS.estado_subsanacion
is 'Indica el estado del requisito referente a la Subsanación';
-- Create/Recreate check constraints
alter table PRY_REQUISITOS
add constraint PRYREQ_ESTSUB_CK
check (ESTADO_SUBSANACION IN ('N', -- NO_SUBSANAR
                                'P', -- PENDIENTE_SUBSANAR
                                'R', -- EN_REVISION
                                'S', -- SUBSANADO
                                'X' -- NO SUBSANADO
));
COMMIT;
```

- PRY_DOCUMENTOS → documentos asociados, con campos de auditoría (fecha_registra_subsanable, usuario_registra_subsanable).

```
-- Add/modify columns
alter table SGP_DOCUMENTOS add activo VARCHAR2(1);
alter table SGP_DOCUMENTOS add observacion VARCHAR2(2000);
alter table SGP_DOCUMENTOS add usuario_registra_subsanable VARCHAR2(30);
alter table SGP_DOCUMENTOS add fecha_registra_subsanable date;
-- Add comments to the columns
comment on column SGP_DOCUMENTOS.activo
is 'Indica si el documento adjuntado es el activo, cuando el documento del requisito ha sido subsanado.';
comment on column SGP_DOCUMENTOS.observacion
is 'Indica la observación cuando el documento del requisito ha sido subsanado.';
comment on column SGP_DOCUMENTOS.usuario_registra_subsanable
```

```
is 'Indica el usuario que solicitó subsanar el documento adjunto al requisito.';
comment on column SGP_DOCUMENTOS.fecha_registra_subsanable
is 'Indica la fecha en que solicitó subsanar el documento adjunto al requisito';

-- Create/Recreate check constraints
alter table SGP_DOCUMENTOS
add constraint SGPDOC_ACTIVO_CK
check (ACTIVO IN ('S' /*Si*/, 'N' /*No*/));

COMMIT;
```

- No se sobrescriben registros; se insertan nuevos documentos para subsanaciones y se mantienen referencias al documento original.

2. Formulario de Proyectos y Programas

2.1 Componentes

Para habilitar y controlar el proceso de subsanación de documentos asociados a los requisitos de una convocatoria, se desarrollaron y ajustaron diversos componentes en el módulo **Proyectos y Programas**.

La integración garantiza que los usuarios puedan **visualizar el estado de subsanación, adjuntar documentos adicionales y revisar observaciones** realizadas por los evaluadores, manteniendo la trazabilidad y control del proceso.

Los principales elementos involucrados son:

1. Vista (View)

- Archivo: requisitos.xhtml
- Define la estructura visual del formulario de requisitos.
- Incluye:
 - Columna para mostrar el **Estado de Subsanación** mediante badges de colores.
 - Botón **Observaciones Documento** para consultar comentarios realizados durante la subsanación.
 - Botón **Adjuntar** con habilitación/deshabilitación dinámica.

2. Controladores (Controllers)

- **RequisitoProyectoSessController, RequisitoSessController, RequisitoConvController**
- Gestionan:
 - Lógica de presentación.
 - Carga y renderización condicional de botones.
 - Validaciones previas a acciones.
 - Comunicación con la capa de servicios EJB.

3. Servicio EJB

- Clase: ProyectosServiceBean
- Contiene métodos para:
 - Obtener el historial de documentos (getDocumentHistory).
 - Actualizar requisitos (modificarRequisito).

4. Base de Datos

- Entidad: RequisitoDTO
- Mapeada con JPA hacia la tabla PRY_REQUISITOS.
- Gestiona información de estado, subsanabilidad y referencias a documentos.

5. JPA

- Uso de CriteriaBuilder y consultas JPQL.
- Gestión de persistencia para entidades como Documento y Requisito.

6. PrimeFaces

- Implementación de componentes interactivos como <p:column>, <p:commandLink> y <p:dataList>.
- Soporte para paginación, estilos dinámicos y mensajes internacionalizados.

2.2 Codificación

2.2.1 Interfaz de Usuario (requisitos.xhtml)

Columna "Estado de Subsanción"

Se añadió una nueva columna que utiliza `<p:column>` para mostrar visualmente el estado de cada requisito con un badge de Bootstrap, diferenciando mediante colores y textos configurados en `messages.properties`.

```
<p:column width="13%" style="text-align: center;">
    <f:facet name="header">
        <h:outputText
value="#{msg['formulario.proyecto.requisitos.estadoSubsanacion.label']}" />
    </f:facet>
    <h:outputText class="badge badge-#{item.requisitoDTO.estadoSubsanacion eq 'N' ?
'default' :
                                item.requisitoDTO.estadoSubsanacion eq 'P' ?
'warning' :
                                item.requisitoDTO.estadoSubsanacion eq 'R' ?
'success' :
                                item.requisitoDTO.estadoSubsanacion eq 'S' ?
'primary' :
                                item.requisitoDTO.estadoSubsanacion eq 'X' ?
'danger' :
                                'default'}"
        value="#{item.requisitoDTO.estadoSubsanacion eq 'N' ?
msg['listaEstadoSubsanacion.N'] :
                                item.requisitoDTO.estadoSubsanacion eq 'P' ?
msg['listaEstadoSubsanacion.P'] :
                                item.requisitoDTO.estadoSubsanacion eq 'R' ?
msg['listaEstadoSubsanacion.R'] :
                                item.requisitoDTO.estadoSubsanacion eq 'S' ?
msg['listaEstadoSubsanacion.S'] :
                                item.requisitoDTO.estadoSubsanacion eq 'X' ?
msg['listaEstadoSubsanacion.X'] :
                                msg['listaEstadoSubsanacion.N']}" />
</p:column>
```

Puntos clave:

- **CSS dinámico:** El class asigna colores según el estado.
- **Internacionalización:** El value obtiene textos desde .properties (listaEstadoSubsanacion.*).
- **Accesibilidad:** El estilo centrado mejora la lectura.

Botón "Observaciones Documento"

Muestra un diálogo modal con las observaciones del documento cuando el estado es "P" (Pendiente de subsanar).

```
<p:commandLink id="observacionesDocButton"
  actionListener="#{requisitoSessController.loadDocumentObservations(item.documentoDTO)}"
  oncomplete="$('#observacionesDoc').modal();"
  rendered="#{item.requisitoDTO.estadoSubsanacion eq 'P'}"
  styleClass="btn btn-xs btn-primary"
  update="dlgObservacionesDoc"
  title="#{msg['formulario.proyecto.requisitos.observacionesDoc.title']}">
  <i class="fa fa-commenting"></i>
  <f:setPropertyActionListener target="#{requisitoSessController.requisitoSeleccionado}"
  value="#{item}"/>
</p:commandLink>
```

Cambio en botón "Adjuntar"

La habilitación se controla con el método desactivarRequisitos del ProyectoConvController:

```
disabled="#{proyectoConvController.desactivarRequisitos(item)}"
```

Diálogo "Observaciones Documento"

Se implementó con <p:dataList> para mostrar un historial paginado:

```
<p:dataList value="#{requisitoSessController.documentHistoryList}" var="doc"
  type="unordered" paginator="true" rows="5" rowsPerPageTemplate="5,10,20"
  paginatorPosition="bottom" pageLinks="5"
  paginatorTemplate="{CurrentPageReport} {FirstPageLink} {PreviousPageLink}
{PageLinks}
{NextPageLink} {LastPageLink} {RowsPerPageDropdown}">
```

2.2.2 Controlador de Vista – RequisitoSessController

Método loadDocumentObservations

```
public void loadDocumentObservations(DocumentoDTO documentoDTO) {
    try {
        if (documentoDTO != null) {
            this.documentHistoryList = proyectoServiceFacade.getDocumentHistory(
                documentoDTO.getConvocatoriaRequisitoId().getConvocatoriaRequisitoId(),
                this.appSessController.getProyectoSelected().getProyectoId());
        }
    } catch (Exception ex) {
        LOG.error("Error al cargar el historial de documentos del requisito", ex);
    }
}
```

```
}  
}
```

- **Entrada:** DocumentoDTO
- **Acción:** Obtiene historial con getDocumentHistory.
- **Salida:** Lista documentHistoryList para <p:dataList>.
- **Control de errores:** Logging de excepción.

2.2.3 Controlador de Conversación – ProyectoConvController

Método desactivarRequisitos

```
public boolean desactivarRequisitos(RequisitoVO requisitoVO) {  
    Boolean disabled = false;  
  
    if (this.verificarConvocatoria() && !appSessController.getSubsanabilidadActivada())  
        disabled = true;  
    if (requisitoVO.getRequisitoDTO().getCumple().equals("S")) disabled = true;  
    if  
    (appSessController.getProyectoSelected().getEstadoProyectoId().getValorDefinicion().equals("290")) disabled = true;  
  
    if  
    (appSessController.getProyectoSelected().getEstadoProyectoId().getValorDefinicion().equals("270")) {  
        disabled = !(requisitoVO.getDocumentoDTO().getDocumentoId() != null  
            && requisitoVO.getDocumentoDTO().getSubsanable().equals("S")  
            && requisitoVO.getRequisitoDTO().getEstadoSubsanacion().equals("P"));  
    }  
    return disabled;  
}
```

- **Evalúa:**
 - Estado del proyecto.
 - Cumplimiento del requisito.
 - Condición de subsanable.
- Retorna true si debe inhabilitar el botón "Adjuntar".

2.2.4 Servicio EJB – ProyectosServiceBean

Método getDocumentHistory

```
@Override
public List<Documento> getDocumentHistory(Long convocatoriaRequisitoId, Long proyectoId) {
    String sql = "SELECT d FROM Documento d WHERE
    d.convocatoriaRequisitoId.convocatoriaRequisitoId = :convocatoriaRequisitoId "
        + "AND d.origenId = :origenId ORDER BY d.fechaRegistraSubsanable DESC";
    Query query = em.createQuery(sql);
    query.setParameter("convocatoriaRequisitoId", convocatoriaRequisitoId);
    query.setParameter("origenId", proyectoId);
    return query.getResultList();
}
```

- **Consulta JPQL** con ORDER BY para mostrar los más recientes primero.
- Uso de parámetros nombrados para seguridad y legibilidad.

Método modificarRequisito

```
@Override
public void modificarRequisito(Requisito requisito) throws SystemException {
    this.modificar(requisito, Requisito.class);
}
```

- Persistencia genérica para actualizar un requisito.

3. Formulario de Beneficios Tributarios

3.1. Componentes Implementados

Con el objetivo de habilitar y controlar el proceso de subsanación de documentos asociados a los requisitos de una convocatoria en el módulo de Beneficios Tributarios, se integraron y adaptaron los siguientes componentes:

- **Vista (View):**
Archivo requisitos.xhtml, encargado de la estructura visual del formulario, incluyendo la tabla de requisitos, botones de acción y diálogo de observaciones.
- **Controladores de Vista y Conversación:**
Clases RequisitoProyectoSessController, RequisitoSessController y RequisitoConvController, responsables de la lógica de presentación, renderización condicional de botones, validaciones y coordinación con la capa de servicios.

- **Servicio EJB (ProyectosServiceBean):**
Contiene métodos de consulta y actualización de requisitos y documentos en base de datos, mediante API de persistencia.
- **Base de Datos:**
Entidad RequisitoDTO y su respectiva tabla PRY_REQUISITOS para almacenar el estado y propiedades de los requisitos.
- **JPA:**
Uso de CriteriaBuilder y consultas JPQL para obtener y modificar registros en base de datos.
- **PrimeFaces:**
Provisión de componentes visuales (tablas, formularios, badges, diálogos) con estilos y validaciones dinámicas.

3.2. Codificación

3.2.1. Interfaz de Usuario (requisitos.xhtml)

Columna "Estado de Subsanación"

Se añadió una columna <p:column> que presenta el estado de subsanación mediante un badge con estilos dinámicos, según el código del estado (N, P, R, S, X):

```
<p:column width="12%" style="text-align: center;">
  <f:facet name="header">
    <h:outputText
value="#{msg['formulario.estimulos.proyecto.requisitos.estadoSubsanacion.label']}" />
  </f:facet>
  <h:outputText class="badge badge-#{...}" value="#{...}" />
</p:column>
```

- **Estilos dinámicos:** Uso de class="badge badge-#{...}" para asignar color según el estado.
- **Texto dinámico:** Carga desde messages.properties (listaEstadoSubsanacion.N, etc.).

Botón "Observaciones Documento"

Permite visualizar observaciones asociadas a un documento en estado "P":

```
<p:commandLink id="observacionesDocButton"
```

```
actionListener="#{requisitoSessController.loadDocumentObservations(item.documentoDTO)}"
oncomplete="$('#observacionesDoc').modal();"
rendered="#{item.requisitoDTO.estadoSubsanacion eq 'P'}"
styleClass="btn btn-xs btn-primary"
update="dlgObservacionesDoc"

title="#{msg['formulario.estimulos.proyecto.requisitos.observacionesDoc.title']}"
<i class="fa fa-commenting"></i>
<f:setPropertyActionListener target="#{requisitoSessController.requisitoSeleccionado}"
value="#{item}"/>
</p:commandLink>
```

- **Condicional de renderizado:** Solo visible si estadoSubsanacion == 'P'.
- **Carga de datos:** Invoca loadDocumentObservations() para recuperar historial del documento.
- **Apertura modal:** Uso de jQuery para mostrar diálogo tras carga.

Cambio en botón "Adjuntar"

Se centralizó la lógica de habilitación en el método desactivarRequisitos() de ProyectoConvController:

```
disabled="#{proyectoConvController.desactivarRequisitos(item)}"
```

Diálogo "Observaciones Documento"

Usa <p:dataList> para listar observaciones con paginación:

```
<p:dataList value="#{requisitoSessController.documentHistoryList}" var="doc"
paginator="true" rows="5" rowsPerPageTemplate="5,10,20"
type="unordered" itemType="none">
</p:dataList>
```

3.2.2. Controlador de Vista (RequisitoSessController)

Método loadDocumentObservations()

Carga historial de observaciones de un documento:

```
public void loadDocumentObservations(DocumentoDTO documentoDTO) {
    try {
        if (documentoDTO != null) {
            this.documentHistoryList = proyectoServiceFacade.getDocumentHistory(
```

```

        documentoDTO.getConvocatoriaRequisitoId().getConvocatoriaRequisitoId(),
        this.appSessController.getProyectoSelected().getProyectoId());
    }
} catch (Exception ex) {
    LOG.error("Error al cargar el historial de documentos del requisito", ex);
}
}

```

- Valida que el documento no sea nulo.
- Consulta en proyectoServiceFacade usando convocatoriaRequisitoId y proyectoId.
- Almacena resultados en documentHistoryList.

3.2.3. Controlador de Conversación (ProyectoConvController)

Método desactivarRequisitos()

Evalúa múltiples condiciones para habilitar/deshabilitar el botón de adjuntar documento:

```

public boolean desactivarRequisitos(RequisitoVO requisitoVO) {
    Boolean disabled = false;

    if (this.verificarConvocatoria() && !appSessController.getSubsanabilidadActivada())
        disabled = true;
    if ("S".equals(requisitoVO.getRequisitoDTO().getCumple())) disabled = true;
    if
("290".equals(appSessController.getProyectoSelected().getEstadoProyectoId().getValorDefinici
on())) disabled = true;

    if
("270".equals(appSessController.getProyectoSelected().getEstadoProyectoId().getValorDefinici
on())) {
        disabled = !(requisitoVO.getDocumentoDTO().getDocumentoId() != null
            && "S".equals(requisitoVO.getDocumentoDTO().getSubsanable())
            && "P".equals(requisitoVO.getRequisitoDTO().getEstadoSubsanacion()));
    }
    return disabled;
}

```

3.2.4. Servicio EJB (ProyectosServiceBean)

Método getDocumentHistory()

Consulta historial de documentos desde la base de datos:

```

@Override
public List<Documento> getDocumentHistory(Long convocatoriaRequisitoId, Long proyectoId) {
    String sql = "SELECT d FROM Documento d WHERE
d.convocatoriaRequisitoId.convocatoriaRequisitoId = :convocatoriaRequisitoId "

```

```
+ "AND d.origenId = :origenId ORDER BY d.fechaRegistraSubsanable DESC";  
Query query = em.createQuery(sql);  
query.setParameter("convocatoriaRequisitoId", convocatoriaRequisitoId);  
query.setParameter("origenId", proyectoId);  
return query.getResultList();  
}
```

Método modificarRequisito()

Actualiza datos del requisito:

```
@Override  
public void modificarRequisito(Requisito requisito) throws SystemException {  
    this.modificar(requisito, Requisito.class);  
}
```

INFORME DE ACTIVIDADES DE SOPORTE EN EL USO DEL SIGP EN LA CREACION DE CONVOCATORIAS Y TRANSFERIR EL CONOCIMIENTO DEL SIGP A LOS FUNCIONARIOS DE MINCIENCIAS, (No. Obligación Contrato: 2.2.3, 2.2.4)

Fecha: 16/Junio/2025 – 16/Julio/2025

Elaborado por: Leonardo Andres Suarez Baldovino

Entidad: Ministerio de Ciencias, Tecnología e Innovación - Minciencias

Introducción

El presente informe detalla las actividades de soporte realizadas en el uso del Sistema de Información para la Gestión de Proyectos (SIGP), abarcando la parametrización de convocatorias, la generación de reportes, la solución de requerimientos y la resolución de errores.

Durante este período, se atendieron diversas solicitudes reportadas por los usuarios en la plataforma. A continuación, se presenta el resumen de las actividades realizadas, agrupadas por tipo de solicitud.

Resumen de Actividades

Tipo de Solicitud	Cantidad de Casos
Solución de Errores	1

Tipo de Solicitud	Cantidad de Casos
Requerimientos	1
Total	2

Detalle de Casos Atendidos

A continuación, se detallan las solicitudes gestionadas, incluyendo la descripción del caso, su estado y la acción realizada:

Solución de Errores

Se solucionaron errores reportados por los usuarios en la plataforma:

N° Solicitud	Descripción	Estado	Observaciones	Acción Realizada
159698	- Error Descarga Ficha Proyecto 115560	Solucionado	-	[Ver detalle]

Solicitud 159698

Asunto			Tiempo total de actividad
Descargar la ficha del ID 115560			00:23:33
Descripción			
Amablemete solicito de su colavoración para descargar la ficha del ID 115560 de la plataforma SIGP ya que no permite la descarga para la validacion de la propuesta.			
Fecha/hora de apertura	Última modificación	Fecha/hora de resolución	Fecha/hora de cierre
11/06/2025 10:30:38	07/07/2025 11:35:51	04/07/2025 12:35:50	07/07/2025 11:35:51

Acción Realizada 159698

En atención al caso reportado sobre la imposibilidad de descargar la ficha técnica del proyecto No. 115560 en el SIGP, se efectuó la revisión de los registros del servidor de aplicaciones, identificando errores recurrentes durante el proceso de generación del documento, originados por la carga de imágenes insertadas mediante enlaces externos (principalmente desde dominios *googleusercontent*).

Ejemplo de imagen de enlace externo:

https://lh7-rt.googleusercontent.com/docsz/AD_4nXcPD8pTVcxUoOuR_Q0eOrtCxtlCBsF97mmMgFPnsWkPkI7bjbXBgAKL5yHXirP6cO07rcuJTmn- R_KADGI0RhMuZK9X9LSipUDvV5hp2hzCYalmvA_Hj7-77uIdfrO0UJWxp7GVwswElalsrDSU?key=FME6senla68oL4BdweHCog

Estos recursos externos, referenciados en los campos **Metodología Propuesta** y **Antecedentes y Novedad del Proyecto**, generaban fallos de tipo **Network unreachable** debido a que las URLs no eran accesibles desde el entorno del servidor.

Evidencia – Error registrado en logs:

```
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXfmmT7cbBLhYBXvGtU5DyCoSPpF8FAotR-WoyB4agRQmEh78_Q4oFlaxEp1wQ-ppTGdsGaV94q7no3Y0tzrgKXdaTG8v_m12_7veDqGsNNit0ydSHmXy00fi83b4-M7key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXcXzBh099YXqh4QVCK10YmpSuBC99BQDCe5z3
nhLlqXtehSaKzYEUKaAGJfthKod8csTiV9vzYbKXXTfY3ycrQzXsv02qG67Ezc8apkasm5VpQ_81P9D5M1LNh7MBWE?key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXcQWE9H2Xwid7IAaF-fl6rzq-QpYP0hHrRp-h
QVfQ8qdSEderw_nBk00SievncD0ZLs3a0nSVP7xUoHVl-J26wEH7YHFZ0QeVBl_X-XE17viGeZJn-lqq13LFG5uV6I?key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXcnNKcV5FL1zdVwvuuM-Q6tieyS_HkqaYwpHi
bH-FPKXmG7SKb4yM1uFcZR8FR6s9HxhNoRKdqi17pSwNsPtc9XI4eJKZIZxsU_l1-noJHQM9meNRiSn7fvldieq1407key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXfimPbncsYoWqd-YK49kC78sPXdxo7P54dRJe
XPR4v1MeqwtIxaM58idy0jZ5SDt-yVvSKh0ZvvRdQITV8p0YySgB_ihpGg0KIKsM5iaTYK3S0m1wTaxZp4acxT3ZzA?key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXcUsmxkVYAY3PutWWSxebFKhYsWshXinwpToR
SuA_KTCKN7G1b-9nAG5h-D55AscLKzDq61NTIqStb0901b0sSikpiEhQn6CYM0hankmCDy4SGJX0SpQAtpsUmQ6tZk?key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXeliJhiVQ0PY68cGNAYFeHrQ5r537xnJziQfk
BWRSDhdqDemAdk-Z_hdtW5Uxq0Zvxo9exrkwc1vGZhb1M008eCVRv9_Gw1biuxTyYvJw7WpitLZVlW1Y190yDICw47key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
Error: Failed to load https://lh7-rt.googleusercontent.com/docsz/AD_4nXdsra0DkdP3k3NBRivBRKluwI2Xhkz2YD9Cz
Q69H_YewcM8HNBKjI80nxlTQbYfSxxovJUfgIOkrQmZuStt9iK7qDq-nmd88hgFQX1To0mE4ChCHXQ0PJweBTBhbF07key=FME6senla68o
k status code 99 and http status code 0 - Network unreachable
```

Para resolver el incidente, se procedió a:

1. **Analizar y extraer** el contenido HTML original de los campos afectados.
2. **Sustituir** las referencias externas por imágenes embebidas en formato Base64, garantizando su disponibilidad directa en el documento.

3. **Validar** la correcta generación y descarga de la ficha técnica desde el sistema SIGP.

Como resultado, el archivo pudo ser descargado exitosamente y se adjuntó a la respuesta, junto con las versiones HTML de respaldo (original y ajustada) para trazabilidad.

Requerimientos

Se recibieron solicitudes de implementación de nuevas funcionalidades en el sistema:

N° Solicitud	Descripción	Estado	Observaciones	Acción Realizada
160870	- Renderizar Campos Formulario BT	Solucionado	Producción	[Ver detalle]

Solicitud 160870

Asunto	Tiempo total de actividad
<p>Estimados SB Software Builder</p> <p>Se solicita la eliminación de los siguientes botones dentro del aplicativo del Sistema de Información de Gestión de Proyectos (SIGP):</p> <p>¿Vincula Proyecto Financiado por Minciencias?</p> <p>¿Registra Código BPIN?</p> <p>¿Registra Número Con</p>	00:23:36
Descripción	
<p>Estimados SB Software Builder</p> <p>Se solicita la eliminación de los siguientes botones dentro del aplicativo del Sistema de Información de Gestión de Proyectos (SIGP):</p> <p>¿Vincula Proyecto Financiado por Minciencias?</p> <p>¿Registra Código BPIN?</p> <p>¿Registra Número Contrato?</p>	

Estos elementos se encuentran actualmente habilitados en el formulario de registro de proyectos. El ajuste consiste en poder deshabilitar la visualización de estos botones, garantizando que no generen alertas, errores o validaciones en el flujo del proceso de inscripción de proyectos en convocatorias vigentes o futuras.

La eliminación de estos botones responde a la necesidad de asegurar la alineación del aplicativo SIGP con los lineamientos actuales establecidos en el documento de Tipologías de Proyectos versión 8. En esta versión, se establece que todos los proyectos deben ser evaluados nuevamente en el marco de cada convocatoria. Por tanto, los contenidos, características y estructuras de los proyectos antiguos no son válidos ni comparables con los criterios actualizados, lo que hace innecesaria y confusa la solicitud de información relacionada con códigos BPIN, números de contrato anteriores o financiación previa por parte de Minciencias.

Eliminar estos campos redundantes optimiza el proceso de inscripción, evita errores de interpretación por parte de los usuarios y mejora la trazabilidad de los proyectos en función de los criterios actuales.

Se solicita de manera urgente la estimación de esfuerzos y posterior desarrollo en producción.

Fecha/hora de apertura	Última modificación	Fecha/hora de resolución	Fecha/hora de cierre
01/07/2025 16:34:23	06/08/2025 10:49:07		

Acción Realizada 160870

1. Estimación de esfuerzos

- Se elabora y diligencia el **formato de estimación de esfuerzos**, donde se detallan los recursos necesarios, tiempos estimados y complejidad del desarrollo.

- Se revisa y ajusta la estimación en función de los requerimientos del usuario y las capacidades del equipo de desarrollo.

2. Inicio del desarrollo

- Se implementan las funcionalidades solicitadas de acuerdo con los requerimientos técnicos y funcionales.
- Se realizan pruebas unitarias para verificar el correcto funcionamiento del desarrollo antes de su despliegue.

3. Despliegue en ambiente de pruebas

- Se despliega el requerimiento en **ambiente de pruebas** para su validación.
- Se ejecutan pruebas funcionales y de integración para detectar posibles errores o mejoras.
- Se documentan los resultados de las pruebas y se hacen los ajustes necesarios.

4. Aprobación y despliegue en producción

- Se presenta el desarrollo para aprobación por parte del usuario funcional o equipo de calidad.
- Una vez aprobado, se coordina el **despliegue en ambiente de producción** asegurando que los cambios no afecten la estabilidad del sistema.
- Se realiza una validación post-despliegue para confirmar el correcto funcionamiento en producción.

5. Pendientes

- La solicitud aún no ha sido cerrada debido a que no se cuenta aún con el formato de pruebas de aceptación firmado y cargado en la plataforma por el líder funcional del SIGP.

Estado Actual:

- ✓ Estimación de esfuerzos completada.
- ✓ Desarrollo implementado.

- ✓ Despliegue en pruebas realizado.
- ✓ Pendiente de aprobación para producción.
- ✓ Despliegue final en producción.
- ✗ Pendiente subida formato de pruebas de aceptación

Conclusiones

Durante este período se brindó soporte técnico en la plataforma SIGP, permitiendo la parametrización de convocatorias, parametrización de rubricas de evaluación, la generación de reportes, la solución de errores y la gestión de requerimientos.

Se continúa con el seguimiento de los casos que permanecen en estado **Pendiente Usuario** y **Pendiente Proveedor**, a la espera de la validación o documentación requerida.

Para garantizar una mejor eficiencia en la resolución de solicitudes, se recomienda:

- Mantener una comunicación constante con los usuarios para agilizar las aclaraciones necesarias.
- Optimizar los tiempos de respuesta en los casos que requieren intervención de terceros.
- Seguir documentando los cambios realizados para futuras referencias.

Firma:

César A. Riaño D.

Ing. César Augusto Riaño Delgado
Gerente Proyecto