ANÁLISIS DE LAS CAPACIDADES DE UN SISTEMA EMPLEANDO ARTEFACTOS ARQUITECTURALES: UN CASO DE UN SISTEMA DE DISEÑO DE PRODUCTOS ORTOPÉDICOS HECHOS A LA MEDIDA.

JERSON ALONSO FLOREZ ROJAS
JORMARY NOGUERA MUÑOZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMATICA
BUCARAMANGA
2016

ANALISIS DE LAS CAPACIDADES DE UN SISTEMA EMPLEANDO ARTEFACTOS ARQUITECTURALES: UN CASO DE UN SISTEMA DE DISEÑO DE PRODUCTOS ORTOPÉDICOS HECHOS A LA MEDIDA.

JERSON ALONSO FLOREZ ROJAS
JORMARY NOGUERA MUÑOZ

Trabajo de grado para optar al título de
Ingeniero de sistemas

Director
PhD. RICARDO LLAMOSA VILLALBA

Codirectores
M.Sc DARÍO JOSÉ DELGADO QUINTERO
M.Sc CLARA ISABEL LÓPEZ GUALDRON

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMATICA
BUCARAMANGA
2016

Agradezco a la Universidad Industrial de Santander y a la Escuela de Ingeniería de sistemas e informática, por brindarme la formación como ingeniero y como persona integral.

A los profesores Ricardo Llamosa, José Darío Delgado, por su tiempo, compromiso y apoyo para realizar el presente proyecto de grado.

A mis padres, Esperanza Rojas y Alonso Flórez, a mis hermanas Vanessa, María Fernanda y Marley por ser mi motivación durante mi proceso de formación.

A ti Katte por tu comprensión, apoyo, dedicación y cariño.

A las personas que de alguna manera contribuyeron a mi formación tanto personal como académica, brindándome conocimiento, experiencia y apoyo.

Jerson Alonso Flórez Rojas.

Gracias a la Universidad Industrial de Santander por permitir mi formación profesional.

A los profesores Ricardo Llamosa y Darío Delgado por su paciencia y orientación.

A mis padres por su amor y apoyo incondicional.

Y a todos los que de alguna u otra manera aportaron algo para mi formación integral.

Jormary Noguera Muñoz

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

RESUMEN

TITULO: ANALISIS DE LAS CAPACIDADES DE UN SISTEMA EMPLEANDO ARTEFACTOS ARQUITECTURALES: UN CASO DE UN SISTEMA DE DISEÑO DE PRODUCTOS ORTOPÉDICOS HECHOS A LA MEDIDA.

AUTOR(ES): JERSON ALONSO FLOREZ ROJAS, JORMARY NOGUERA MUÑOZ

PALABRAS CLAVE: MODEL-CHECKING; CAPACIDADES; MODELOS ARQUITECTURALES.

DESCRIPCIÓN:

La técnica de evaluación Model-Checking (MC) ha demostrado ser útil a la hora de evaluar el cumplimiento de requerimientos establecidos para los sistemas, ya que permite validar la lógica y el comportamiento de los sistemas que son modelados en lenguajes como UML, SysML y BPMN. Aunque esta técnica detecta fallas de manera temprana en el sistema, estas fallas están más relacionadas con las necesidades del cliente que con las metas de la organización.

En este trabajo se aplica un modelo de detección de fallas asociadas a las metas de la organización. Este modelo se basa en la técnica model checking y en lugar de verificar el cumplimiento de requerimientos se hace un análisis del cumplimiento o no de las capacidades del sistema.

El análisis de las capacidades se hizo a un sistema de diseño de productos ortopédicos hechos a la medida. El método de evaluación toma como base modelos de requerimientos, capacidades y comportamiento de la arquitectura del sistema de diseño de productos ortopédicos, que están desarrollados bajo el marco arquitectural DoDAF 2.0, a partir de estos modelos arquitecturales se construye un modelo de comportamiento formal y ejecutable que permite determinar el cumplimiento o no de las capacidades del sistema.

* Trabajo de Grado.

** Facultad de Ingenierías Fisicoquímicas. Escuela de Ingeniería Sistemas e Informática. Director. PhD. Ricardo Llamosa Villalba. Codirectores. M.Sc Darío José Delgado Quintero, M.Sc Clara Isabel López Gualdron

ABSTRACT

TITLE: CAPABILITY ANALYSIS OF A SYSTEM THROUGH ARCHITECTURAL ARTIFACTS: A CASE OF A CUSTOM IMPLANTS DESIGN SYSTEM.

AUTHORS: JERSON ALONSO FLOREZ ROJAS, JORMARY NOGUERA MUÑOZ

KEYWORDS: MODEL-CHECKING; CAPABILITIES; ARCHITECTURAL MODELS, ARQUITECT ENTERPRISE.

DESCRIPTION:

The Model-Checking (MC) evaluation technique has proved to be useful in evaluating the fulfillment of established requirements for the systems, since it allows validating the logic and behavior of systems that are modeled in languages such as UML, SysML and BPMN. Although this technique detects failures early in the system, these failures are more related to the needs of the client than to the goals of the organization.

In this work a fault detection model is applied that is associated to the goals of the organization. This model is based on the model checking technique and instead of verifying the fulfillment of requirements it is made an analysis of the compliance or not of the capabilities of the system.

The analysis of the capabilities was made to a system of design of orthopedic products made to measure. The evaluation method is based on models of requirements, capabilities and behavior of the architecture of the design system of orthopedic products, which are developed under the DoDAF 2.0 architectural framework, from these architectural models a formal and executable behavior model is constructed that allows to determine the compliance or not of the capabilities of the system.

*Bachelor Thesis.

** Facultad De Ingenierías Físico-Mecánicas, Escuela De Ingenierías de Sistemas e Informática. Director. PhD. Ricardo Llamosa Villalba. Codirectores. M.Sc Darío José Delgado Quintero, M.Sc Clara López Gualdron.

INTRODUCCIÓN

Las arquitecturas empresariales han permitido realizar retos de ingeniería altamente complejos [1], donde está representa una ayuda para la toma de decisiones dentro de una organización; las arquitecturas son desarrolladas bajo una metodología y guías llamadas Marcos Arquitecturales (MA) que son un conjunto de vistas, propiedades, prácticas y conocimiento que les permiten a las organizaciones crear su propia arquitectura de acuerdo a su modelo de negocio. Siguiendo la metodología de los MA se generan modelos, documentos y reportes llamados artefactos arquitecturales. Hay que mencionar que los artefactos arquitecturales permiten la descripción de la organización desde diferentes perspectivas. Con el fin de determinar la pertinencia de los modelos arquitecturales cuando son desarrollados, se implementan metodologías para verificar estos artefactos. Perfilándose estas metodologías de verificación como uno de los grandes retos en el diseño de software y sistemas en general [19].

En el caso particular de un proyecto de software. En las etapas tempranas de diseño, la detección de falla es menor a un 10%. Sin embargo, en esta fase se introducen alrededor de un 40% de estas. Pero comparando el costo de solucionar una falla en fase de diseño a repararla en cualquier otra etapa, ya sea de implementación, pruebas, integración, implantación, el costo crece exponencialmente [2]. La relación entre el costo de reparar una falla y la etapa del ciclo de vida donde se encuentra esta falla para un sistema software y un sistema en general, se conserva. La detección de fallas en fases de diseño consiste en la verificación lógica de los modelos que los componen (UML, SysML, BPMN, etc) y en la verificación de cumplimiento de los requerimientos de diseño [3,4,5,6]. Sin embargo, la verificación lógica de los modelos solo asegura la detección de fallas relacionadas con las necesidades de los interesados (*stakeholders*), pero no se puede asegurar que el comportamiento del sistema pueda alcanzar sus metas.

ISO/IEC/IEEE 24765 [7] (*system and software engineering – Vocabulary*), define un requerimiento como "una condición o capacidad necesitada por un usuario para resolver un problema o alcanzar un objetivo", en un contexto arquitectural, los requerimientos soportan el diseño de la capacidades del negocio, los modelos arquitecturales (especialmente los modelos de negocio u operacionales) representa dichas capacidades, a su vez, el diseño de las capacidades busca satisfacer los requerimientos así como soportar las metas del negocio[8,9]. Por lo siguiente, una capacidad organizacional representa una habilidad que la organización tiene y puede generar un valor integral.

En este trabajo se busca aplicar un modelo para el análisis de capacidades basado en validación de modelos (*model-checking*) desarrollado por Delgado Dario J, et al 2016 [13], el cual verifica los modelos del sistema en términos de capacidades en etapas tempranas del diseño. Para la aplicación del modelo, se utiliza como caso de estudio un sistema para el diseño de implantes ortopédicos a la medida [10], con una arquitectura empresarial implementada utilizando el marco arquitectural DoDAF [11,12].

Organización del documento:

Para presentar el modelo de detección de fallas haciendo verificación en los modelos arquitecturales del sistema aplicado en este trabajo, la estructura del documento se define de la siguiente manera:

El capítulo uno habla del sistema de diseño de implantes a la medida sobre el cual se hace la aplicación del modelo de verificación enfocado en capacidades. También se hace una descripción conceptual de las arquitecturas de sistemas y se muestran los diferentes modelos arquitecturales usados para el análisis de las capacidades.

En el capítulo dos se presenta la importancia de la detección temprana de fallas en el desarrollo de sistemas y se expone la relación entre requerimientos y capacidades de donde se despliegan los supuestos de este nuevo modelo.

Para el capítulo tres se muestra el esquema general de la técnica *model checking. Ya que,* Esta es la técnica usada por el modelo de detección de fallas enfocado en capacidades, que también se ilustra en este capítulo.

Los resultados del modelo enfocado en capacidades aplicado en el sistema de diseño de implantes ortopédicos, son mostrados en el capítulo cuatro.

En los últimos capítulos: cinco y seis, se presentan las conclusiones y líneas de trabajo futuro.

1. CASO DE APLICACIÓN- SISTEMA DE DISEÑO DE MODELOS ORTOPÉDICOS A LA MEDIDA (SMOM)

El sistema de diseño de modelos ortopédicos hechos a la medida (SMOM) tiene como misión diseñar dispositivos ortopédicos adaptados a las necesidades específicas de pacientes que presentan fracturas complejas o cuya geometría no corresponde con las dimensiones de los implantes estándar [28]. Estos productos personalizados tienen efectos positivos como la precisión, la adaptación biomecánica y una recuperación en menor tiempo [14].

El sistema SMOM se soporta de tres herramientas software:

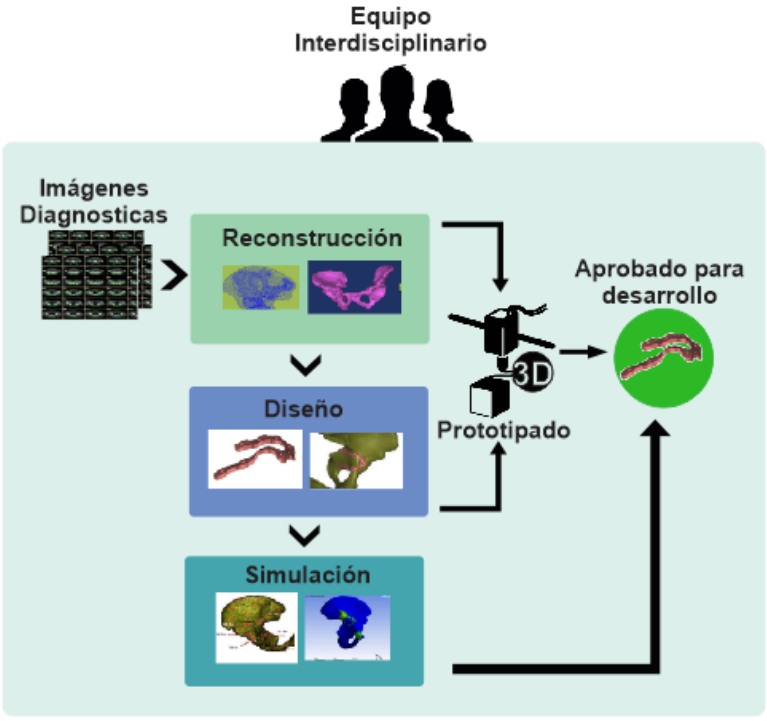
- BIOCAD para la reconstrucción de segmento óseo a tratar a partir de tomografías.
- CAD para modelado de los implantes diseñados
- CAE para el análisis biomecánicos y validación de los diseños obtenidos.

Complementariamente se utilizan equipos de prototipado rápido RP para así obtener modelos físicos de los volúmenes óseos tratados que ayudan para el análisis del trauma y así generar alternativas de diseño. Por lo cual, estos softwares se integran según la necesidad y complejidad del caso a tratar [15]

La figura 1 muestra el contexto en el que se encuentra el sistema de diseño, y presenta una vista de alto nivel de su funcionamiento. En el anexo A se presenta la Descripción e Información Resumida de SMOM.

El sistema SMOM pretende con el tiempo ser capaz no solo de diseñar sino también de manufacturar los productos ajustados a las necesidades del paciente. Es decir, que el producto terminando sea hecho con los materiales adecuados y totalmente validado para ser implantado en la lesión ósea del paciente, ya que hasta el momento se diseñan los dispositivos, pero estos no son usados para las intervenciones quirúrgicas. Ver anexo B.

Figura 1. Diagrama de estados del sistema SMOM



1.1. ARQUITECTURAS DE SISTEMAS

Una arquitectura de sistemas (AS) es una descripción formal de la representación de un sistema, organizada de forma tal que se pueda razonar acerca de la estructura y el comportamiento de este.

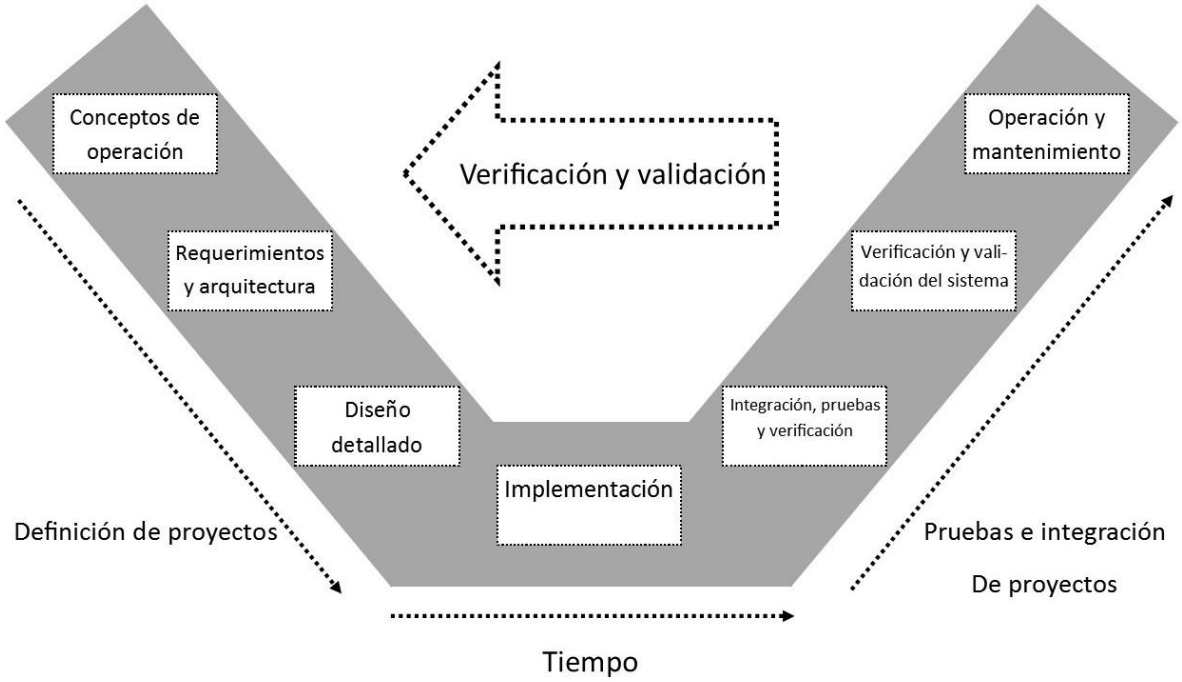
Las AS articulan las diferentes vistas que un sistema puede tener en documentos y modelos (usualmente construidos en lenguajes de modelado tales como SysML, UML, BPMN, entre otros), estas vistas permiten relacionar requerimientos del sistema con sus capacidades, su operación, la tecnología, los datos, los servicios, entre otros factores de interés para el análisis de los sistemas.

Como una parte en el ciclo de vida para el desarrollo de sistemas, una AS es empleada como soporte en el desarrollo de los modelos del sistema, específicamente en i) la definición de requerimientos, ii) la construcción de escenarios, iii) el desarrollo de un diseño bien definido mediante un proceso de eliminación de opciones, iv) un diseño y análisis estructurado.

En el proceso de diseño de un sistema complejo es común el uso de ciclos de vida, en el desarrollo de Software se emplean modelos de desarrollo como el de Cascada, Espiral, modelo V, entre otros [22], con estos modelos se pretende mejorar el desarrollo de los

sistemas, reducir costos, riesgos, entre otros aspectos. El modelo V, el cual es uno de los más utilizados en ingeniería de sistemas y el cual es la base para los modelos V+ y V++ [29], ubica las arquitecturas de un sistema en las etapas tempranas del análisis de soluciones y diseño para desarrollar o mejorar un sistema, ver figura 2.

Figura 2. Modelo V para procesos en ingeniería de sistemas (Tomado de [29])



Las AS proveen un método estructurado y repetible para evaluar inversiones y alternativas de inversión, así como la capacidad de aplicar un cambio organizacional efectivo, desarrollar nuevos sistemas, y desplegar nuevas tecnologías [30]. Las AS buscan describir un sistema en términos de capacidades requeridas en lugar de los requisitos del sistema, esto se debe a que los requerimientos del sistema están más relacionados con las necesidades de los *Stakeholders* , y las capacidades están más relacionadas con los objetivos operacionales y las metas visiónales, ver Figura 13, [22].

Para el desarrollo de las arquitecturas es usual encontrar Marcos Arquitecturales (MA) los cuales facilitan la construcción de las AS mediante el establecimiento de prácticas y herramientas las cuales brindan un marco de referencia para la creación de las mismas. Entre los marcos de trabajo arquitectural se destacan TOGAF, Zachman, FEAF, DoDAF, entre otros [16].

1.2. COMPONENTES ARQUITECTURALES DEL SISTEMA SMOM

La arquitectura del sistema SMOM está conformada por capas arquitecturales que se alinean todas en función de una visión estratégica que les dé sentido y, a la vez, que los convierta en recursos útiles para la toma de decisiones, ver figura 3.

- **Capa de capacidades:** En este nivel se define qué es lo que se tiene que hacer para cumplir con la misión del sistema. Están contempladas las habilidades que el sistema requiere para el diseño de productos ortopédicos a la medida. Estas capacidades son: Prototipado, planeación, comunicación, modelado de dispositivo, verificación de implante, exportación para manufactura, simulación estática estructural, simulación quirúrgica, apropiación de tecnología, adquisición de conocimiento, transferencia de conocimiento y reconstrucción ósea digital.

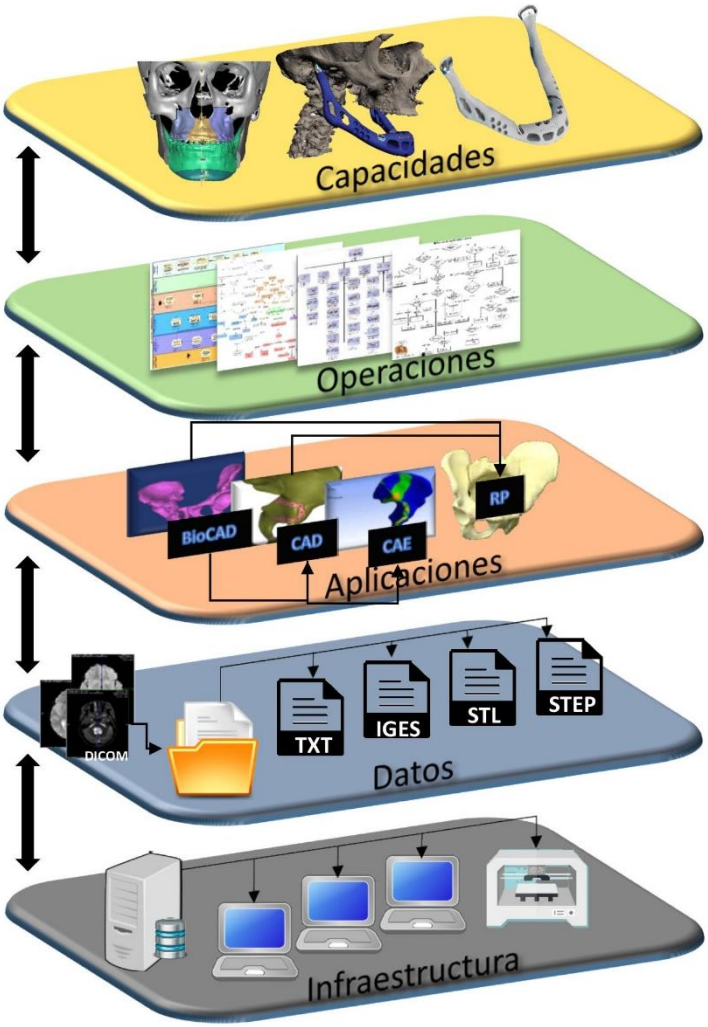
- **Capa de operación o de negocio:** define la estrategia de negocio, la estructura organizacional y los procesos clave de la organización. El sistema SMOM ubica en esta capa procesos tales como: identificación del contorno óseo, conversión de superficie a volumen, diseño de estudios biomecánicos, entre otros.

- **Capa de aplicaciones:** Provee la definición funcional para cada uno de los sistemas de información requeridos, las interacciones entre estos sistemas y sus relaciones con los procesos de negocio CORE de la organización. Las aplicaciones pilares usadas en el sistema SMOM son: BioCAD, CAD, CAE y RP.

- **Capa de datos:** Describe la estructura de datos físicos y lógicos de la organización y sus modelos de gestión. Datos como los requeridos por el proceso de reconstrucción ósea digital se encuentran en las tomografías computacionales de la lesión ósea. Para el proceso de diseño del dispositivo se requiere de información dada por las restricciones de cada tipo de lesión. Estos datos y otros se organizan de tal forma que soporten las metas del sistema.

- **Capa de infraestructura**: Describe la estructura hardware, software y comunicaciones, requerida para dar soporte a la implantación de los sistemas de información. Para el caso del sistema SMOM, aquí se encuentran los softwares donde se soporta el diseño de implantes, los equipos donde se ejecutan estos softwares, la impresora 3d para el prototipado rápido y todo el componente eléctrico y de comunicación para los dispositivos.

Figura 3. Capas arquitecturales del sistema SMOM



1.3. ARQUITECTURA SMOM USANDO MARCO ARQUITECTURAL DODAF 2.0

El sistema SMOM para la construcción de su arquitectura se basó en DoDAF 2.0, marco arquitectural del Departamento de Defensa de los Estados Unidos. El cual proporciona una dirección general para el desarrollo, uso y gestión de las arquitecturas con énfasis en la interoperabilidad entre los sistemas complejos de gran tamaño [17], [18]. Los productos mostrados por el MA DoDAF, ver figura 4, con respecto al sistema SMOM son representados con descripciones graficas de una arquitectura utilizando lenguaje de

modelado SySML. Durante la descripción de la arquitectura SMOM se tienen en cuenta las vistas que describen las capacidades y operaciones del sistema. De acuerdo a los requerimientos establecidos por los interesados, ver figura 5. Dentro de los puntos de vista que posee DoDAF encontramos:

- Todas las Vistas (AV): corresponde a la descripción de la arquitectura completa, en la cual se tiene cuenta el contexto y alcance.
- Vista Operacional (OV): esta vista se encarga de proveer las descripciones de las tareas, elementos operativos y el intercambio de información necesario para alcanzar los objetivos definidos por la arquitectura.
- Vista de Capacidades (CV): esta vista se encarga de definir el alcance y las metas a cumplir establecidos por los interesados y usuarios dentro de la arquitectura.

Definiendo que una capacidad se refiere a la habilidad que tiene un sistema para realizar una tarea en particular o seguir un curso de acciones en particular para poder alcanzar las metas establecidas, en la vista de capacidades CV-2 (Jerarquía de Capacidades) ver figura 6. El sistema SMOM muestra las capacidades definidas entre el grupo de trabajo de ingenieros y los interesados, basados en el conjunto de requerimientos, en el modelo se presenta la jerarquía de un conjunto de capacidades el cual se encarga de describir un resumen en la línea de tiempo las capacidades que están implementadas actualmente y las que se quieren cumplir en un futuro.

Por otro lado, está la vista de capacidad CV-4 (Dependencia de Capacidades) ver figura 7, la cual proporciona la información para analizar las dependencias entre las capacidades, donde sus agrupaciones de capacidades son lógicas y el propósito de estas es guiar la gestión del sistema para así cumplir las metas establecidas por la organización.

Para tener una idea del comportamiento del sistema SMOM, se deben tener en cuenta las vistas operativas de la arquitectura (OV) las cuales describen las actividades operacionales, reglas y restricciones necesarias para llevar a cabo las operaciones del sistema teniendo en cuenta los requerimientos y las capacidades establecidas desde el inicio por los interesados y los ingenieros. Los artefactos tenidos en cuenta en las OV para el análisis del comportamiento del sistema son el OV-5a (Jerarquía de Actividades Operacionales) ver figura 8, el cual muestra la jerarquía de operaciones de cada una de las actividades establecidas para las diferentes tecnologías BioCAD, CAD, CAE y RP. La vista operacional OV-5b (Modelo de Actividad Operacional) para el sistema SMOM ver figura 9, muestra las relaciones de dependencias que existe entre las actividades para el comportamiento de cada una de las tecnologías, el cual describe las tareas que normalmente se llevan a cabo para lograr la misión u objetivo establecidos.

En la arquitectura del sistema SMOM, el modelo OV-6a (Modelos de Reglas Operacionales) ver figura 10, el cual muestra un panorama del comportamiento operacional del sistema, se puede observar las diferentes reglas y restricciones detalladas, las cuales muestran el comportamiento del sistema bajo condiciones incorporadas en el modelo, y el modelo OV-6b (Modelo de descripción de la transición de estados) el cual muestra un

panorama general de las posibles configuraciones del sistema en su comportamiento dinámico, ver figura 11.

Figura 4. Vistas arquitecturales del marco DoDAF (Tomado de [31])

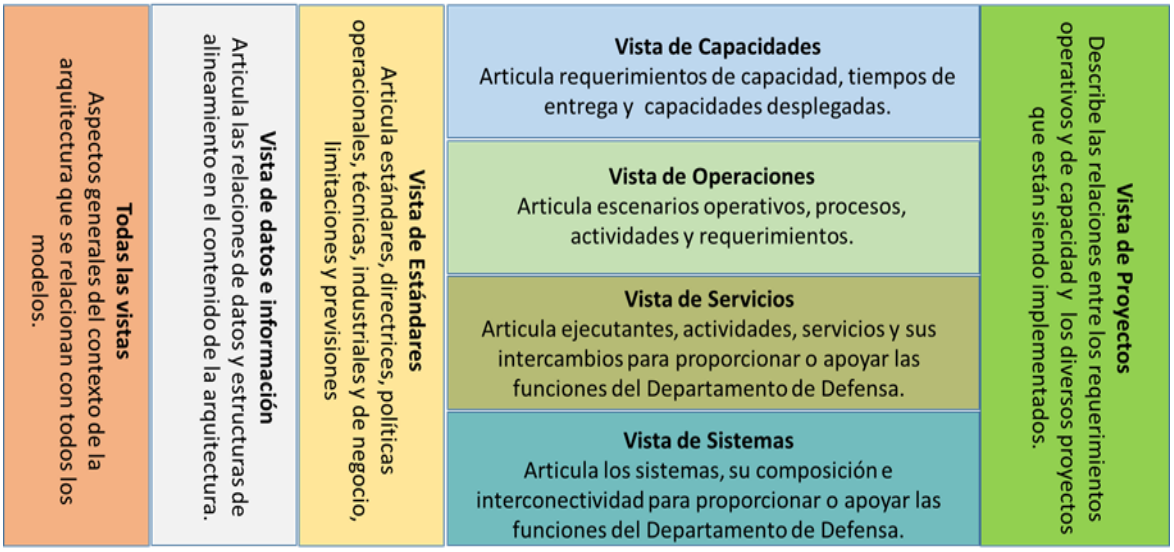


Figura 5. Diagrama de requerimiento de SMOM, ver anexo C

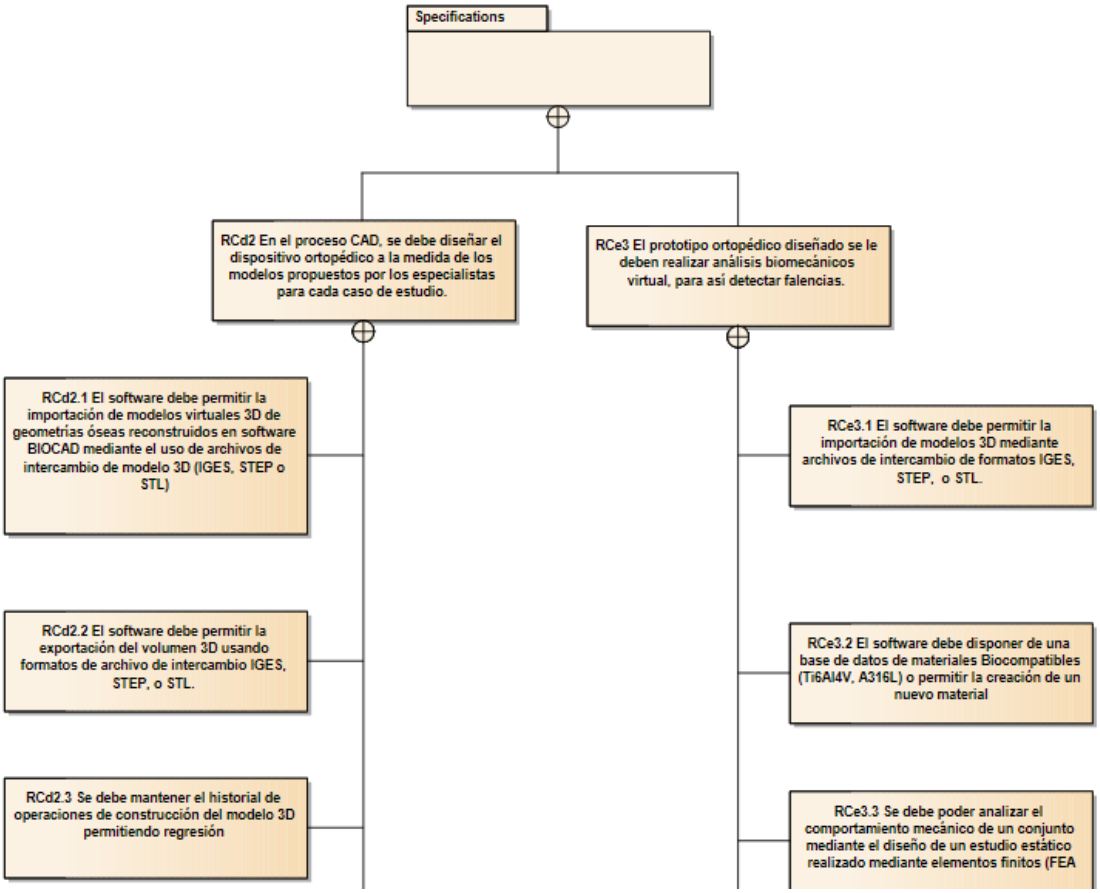


Figura 6. CV-2 Jerarquía de Capacidades de SMOM

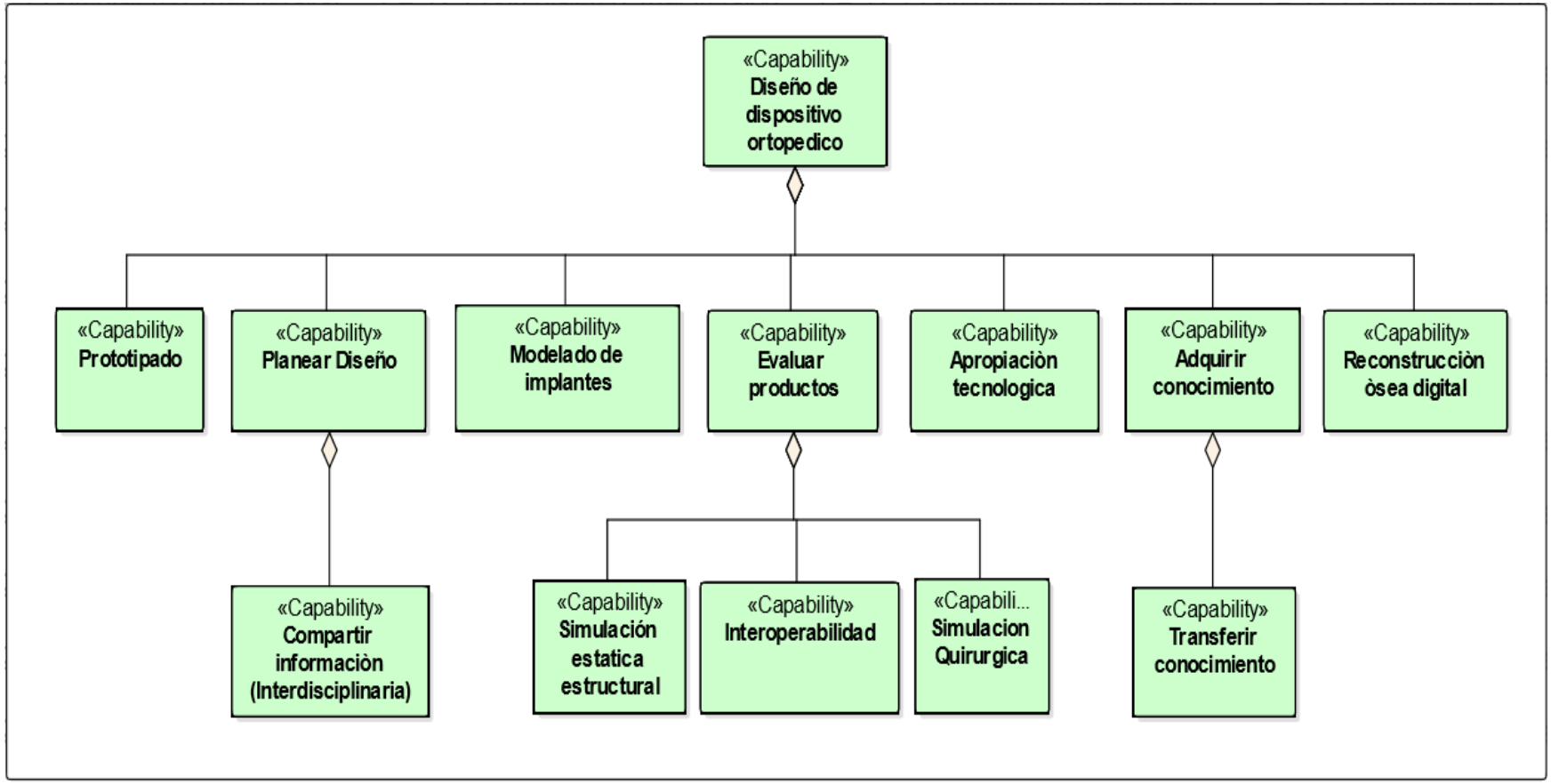


Figura 7. CV-4 Dependencias de Capacidades de SMOM

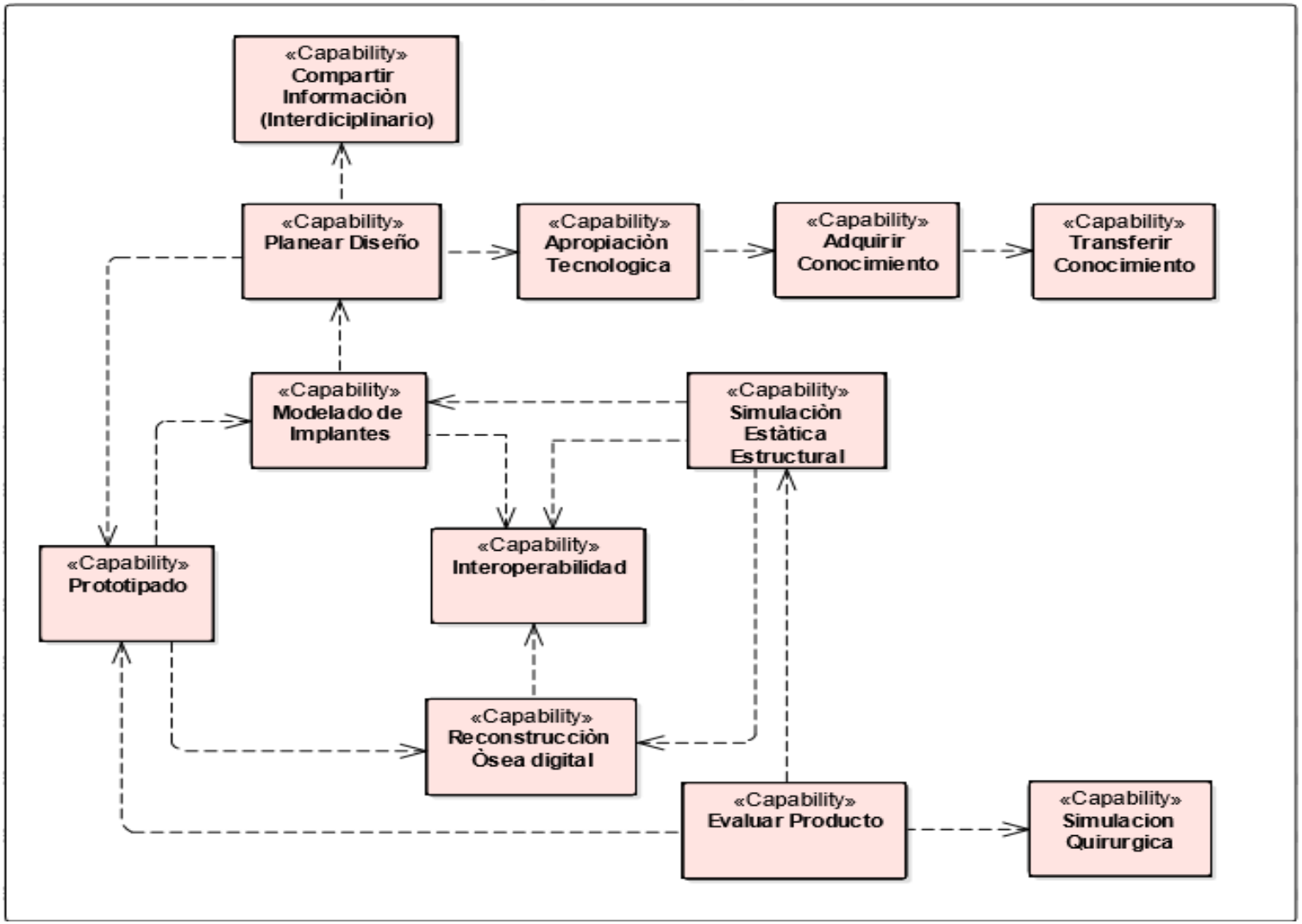


Figura 8. OV-5a Jerarquía de Actividades Operacionales de SMOM, ver anexo F

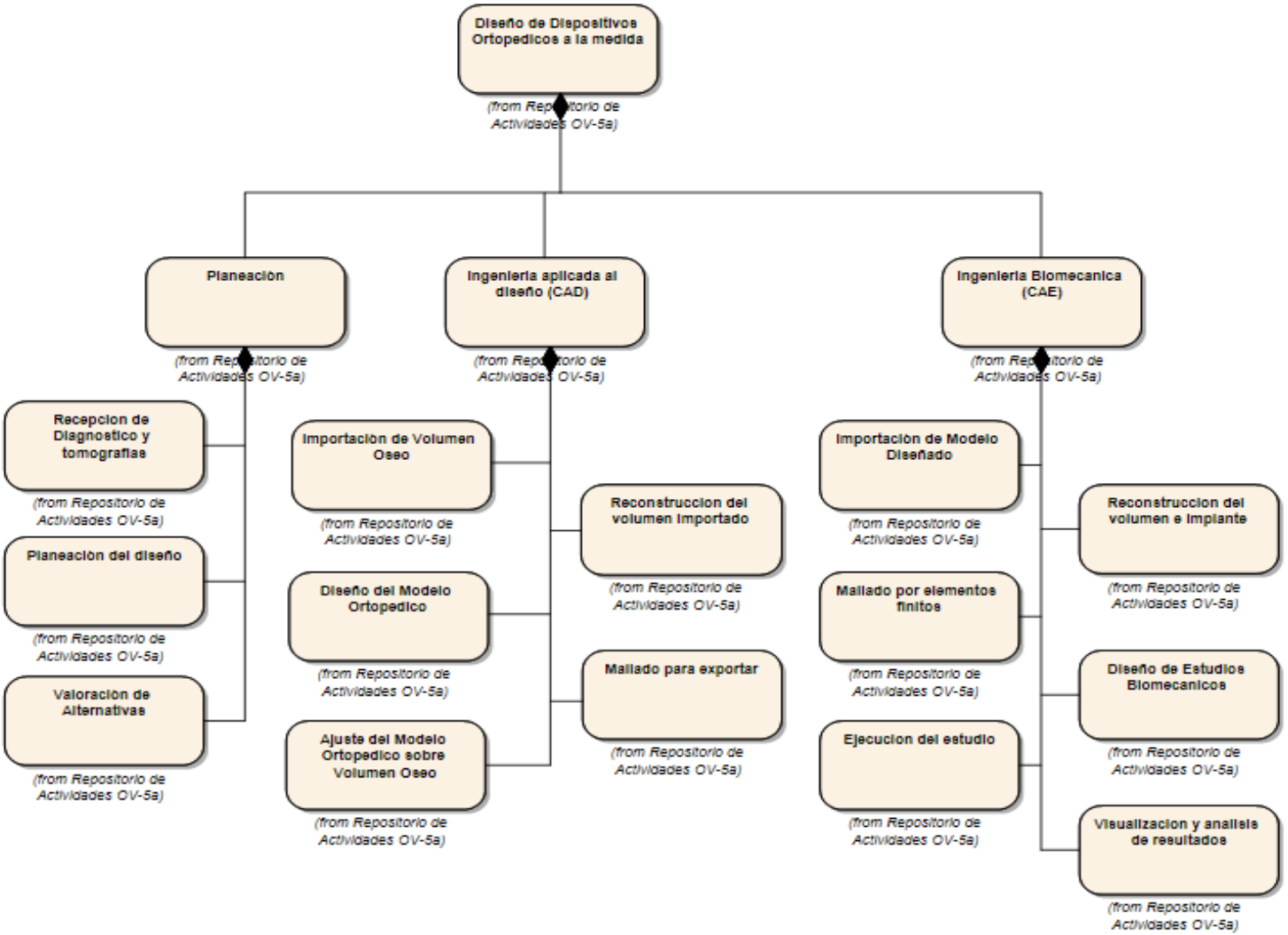


Figura 9. OV-5b Modelo de Actividad Operacional de SMOM, ver anexo G

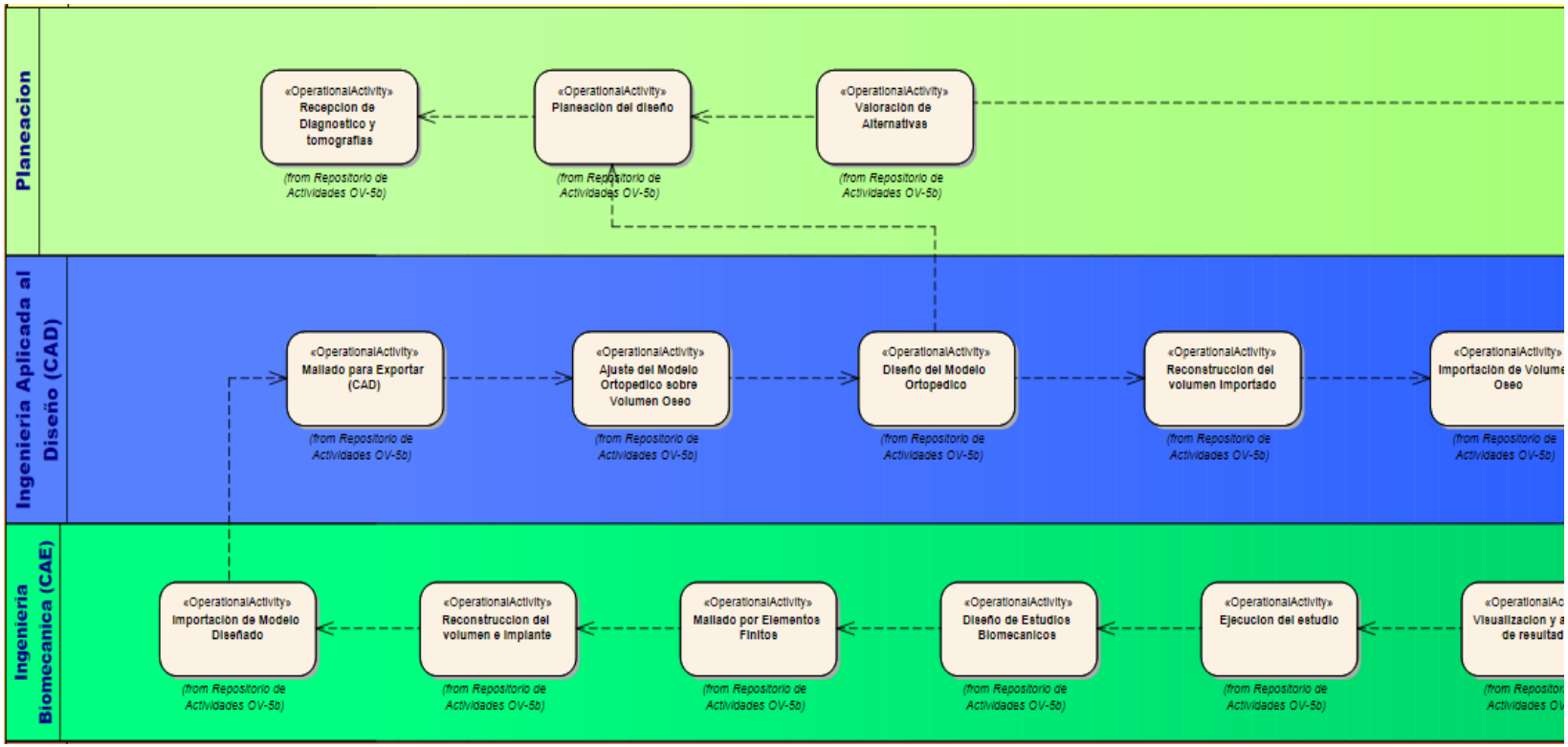


Figura 10. OV-6a Modelo de Reglas Operacionales de SMOM, ver anexo H

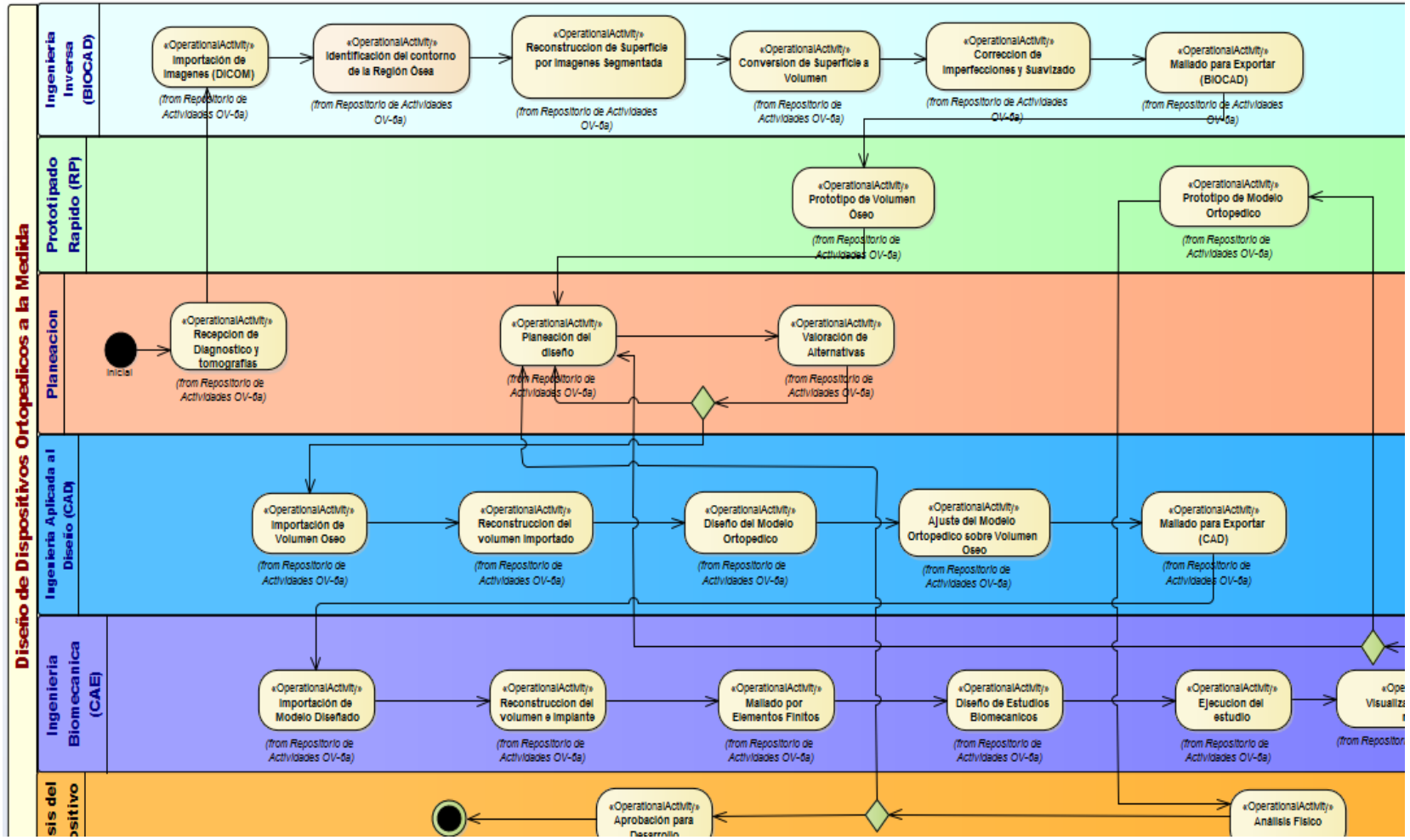
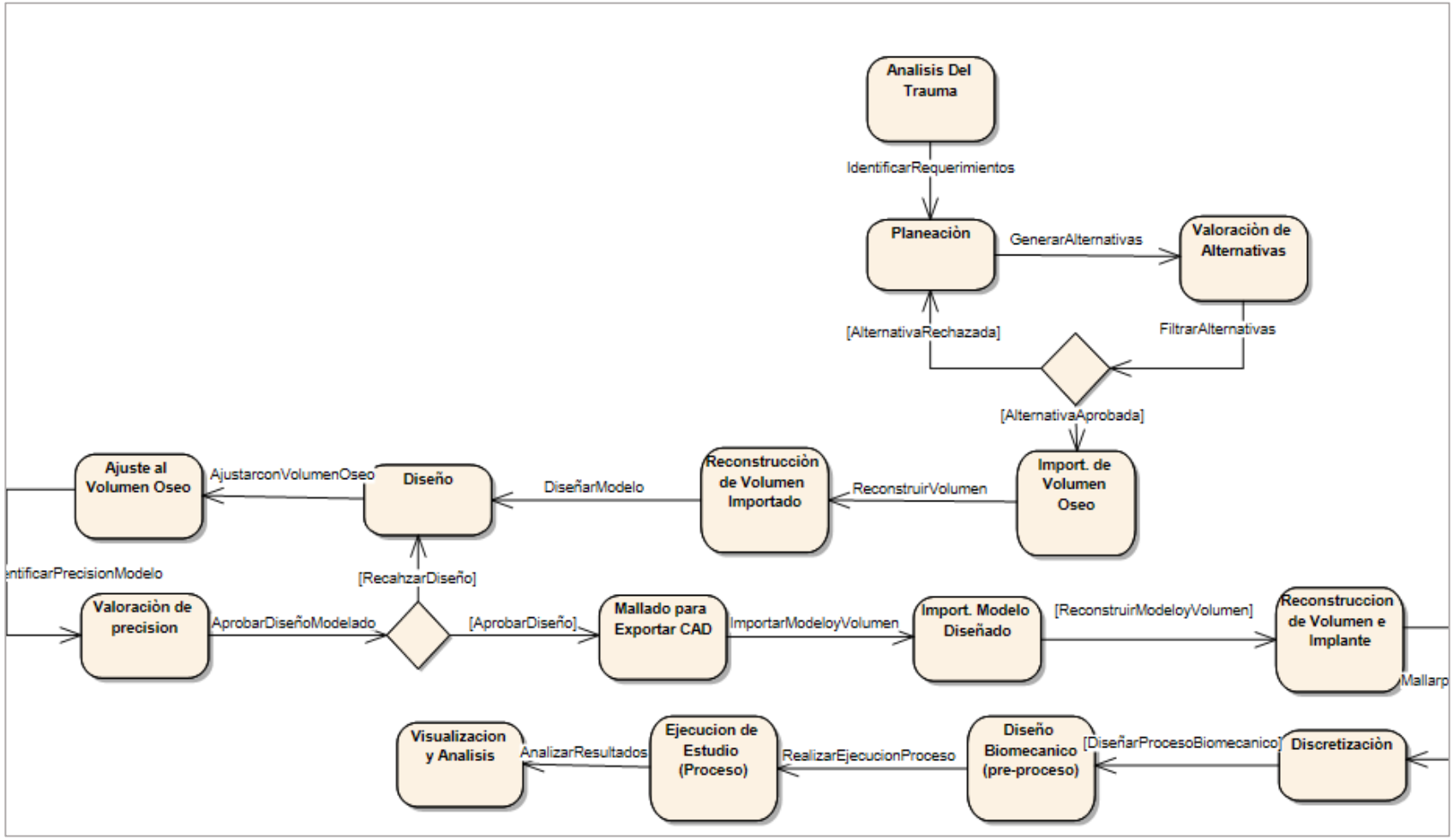


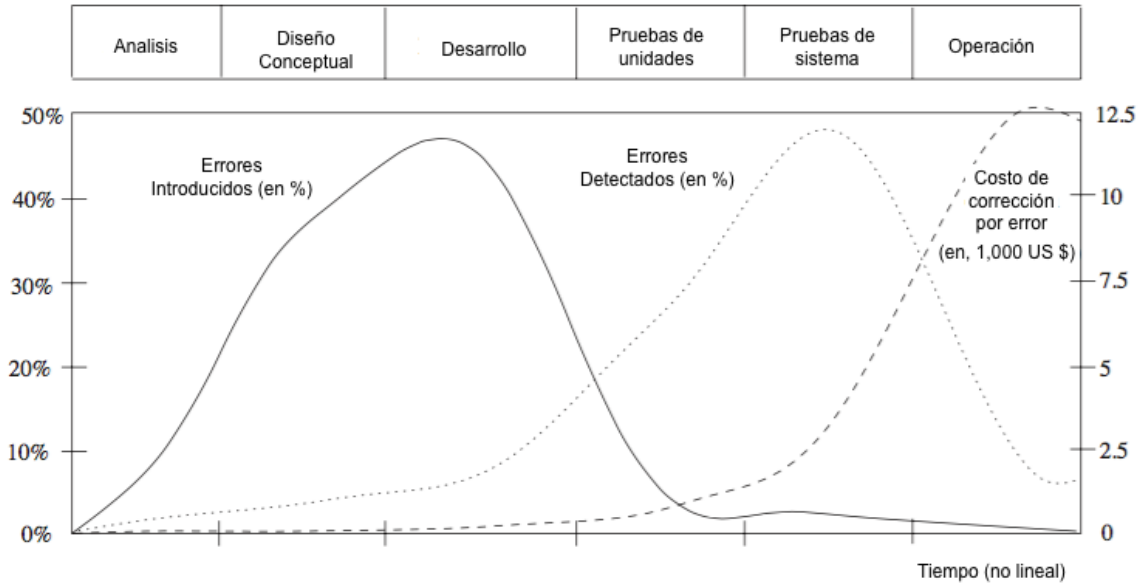
Figura 11. OV-6b Modelo de Descripción de Transición de Estados de SMOM, ver anexo I



2. DESCRIPCIÓN DE LA SITUACIÓN Y SOLUCIÓN DEL PROBLEMA

Uno de los grandes retos en el diseño de software y sistemas en general es la detección de errores en fases tempranas del ciclo de vida. El costo de reparar una falla durante el mantenimiento de un sistema es aproximadamente 500 veces más alto que repararlo en fase de diseño [19]. Como se puede ver en la Figura 12, el costo de reparar una falla en fase de diseño es extremadamente benéfico en comparación a reparar una falla en fase de pruebas u operación. Por tanto, acelerar el proceso de verificación de un sistema empleando artefactos de diseño es un reto.

Figura 12. Costos de introducción detección y reparación de errores, tomado de [19]



A la fecha, los enfoques empleados para la detección de fallas en fase de diseño se han basado en la verificación de la lógica de los modelos (UML, SySML, BPMN, etc) y en la verificación del cumplimiento de requerimientos en los mismos.

La verificación lógica de modelos asegura la detección de fallas asociadas con las necesidades de los stakeholders, sin embargo, el enfoque no puede asegurar que el sistema logre sus metas de comportamiento.

Una capacidad de un sistema representa la habilidad que tiene este para llevar a cabo una actividad que genere valor. Existe una relación cercana entre los requerimientos y las capacidades. En ISO/IEC/IEEE 24765 [20] (System and software engineering – vocabulary, 2010) se define un requerimiento como "una condición o capacidad que necesita un usuario para resolver un problema o para alcanzar un objetivo". Desde un punto

de vista arquitectural, los requerimientos de negocio soportan el diseño de las capacidades del negocio, los modelos arquitecturales representan estas capacidades, y el diseño de capacidades busca satisfacer los requerimientos y lograr las metas organizacionales [13,21,22], ver figura 13.

Figura 13. Relación entre requerimientos, capacidades y metas del sistema (Tomado de [27])

Según lo dicho anteriormente, es apropiado el uso de una metodología que se enfoque en buscar fallas en etapas tempranas del ciclo de vida de un sistema relacionadas con las metas de la organización; es decir un enfoque de la metodología de verificación de modelos o *Model checking* (MC) en términos de capacidades.

En el caso del sistema de diseño de productos ortopédicos hechos a la medida, el uso de MC en términos de capacidades da como resultado el cumplimiento o no de las capacidades representadas en los modelos arquitecturales de dicho sistema, haciendo posible una verificación de una obtención correcta de las metas del sistema en términos de los modelos arquitecturales.

3. MODEL-CHECKING COMO TÉCNICA DE DETECCIÓN DE FALLAS

Para la comprensión de este trabajo, es necesario entender algunos conceptos relacionados con la técnica de detección de fallas para modelos no formales de una arquitectura empresarial. Por lo tanto, con el fin de establecer el estado actual de los conceptos relevantes, se presenta en este capítulo, los conceptos destacados para la validación de modelos desarrollado por (Delgado Darío J, et al 2016). [13]

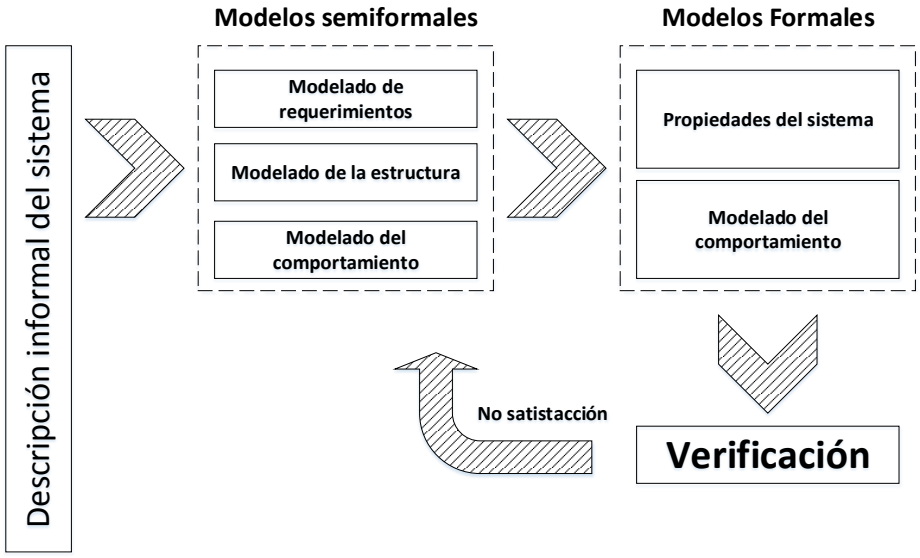
3.1. VALIDACIÓN DE MODELOS (MODEL-CHECKING MC)

La técnica de model-checking (MC) propuesta por Clarke y Emerson [22], es una técnica de verificación de modelos para sistema de estados finitos con respecto a algunas especificaciones lógicas temporales; su enfoque estándar se centra en los modelos semi-formales como SysML, UML, BPMN y entre otros, especialmente aquellos que se usan para describir el comportamiento de un sistema. Por consiguiente, usando los modelos de comportamiento se pueden desarrollar modelos formales (matemáticos y ejecutables), los cuales son utilizados para verificar los aspectos dinámicos del sistema y así poder detectar los posibles fallos usando una herramienta de comprobación de modelos. Observe la figura 14.

La detección de fallas usando MC sigue los siguientes pasos:

i. Convertir los modelos de comportamiento del sistema en modelos formales, por ejemplo, en Labelled transitions System.
ii. Especificar los requerimientos del sistema en términos de Logic Temporal Specifications (LTL).
iii. Utilizar una herramienta para MC y así poder ejecutar los LTL y determinar su cumplimiento.

Figura 14. Esquema general de validación de requerimientos usando Model-Checking (Tomado de [16])

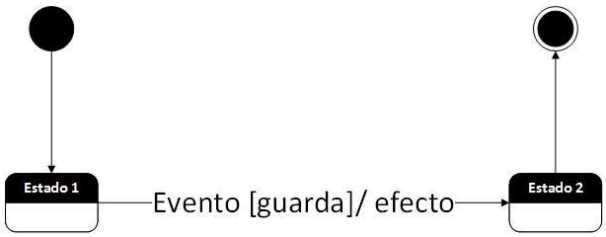


3.2. DIAGRAMA DE ESTADOS Y ESTRUCTURA KRIPKE

El adecuado comportamiento de un sistema está relacionado con el cumplimiento de los requerimientos dados por los interesados; las técnicas de validación de modelos buscan a partir de los modelos de comportamiento verificar si los requerimientos se cumplen. Los diagramas de estados de máquinas son los empleados para dicha verificación [23]. Los diagramas de estados (DE) son una herramienta grafica útil para describir el comportamiento dinámico de un sistema [24], los DE son un tipo de modelo proveniente de UML e incorporado en SysML, al ser descritos por un lenguaje semiformal los DE tienen una sintaxis y una semántica definida para la descripción del comportamiento de un sistema cualquiera.

Un diagrama de estados es una tupla, $DE = S, Var, A, E, G, Edges$ donde S es el conjunto de estados, E es un conjunto de eventos, Var es un conjunto de variables, G es el conjunto de restricciones, $Edges$ es el conjunto de transiciones, los cuales describen el comportamiento de la estructura del sistema, ver Figura 15.

Figura 15. Esquema de parámetros de una transición de estados.



En términos concretos de validación de modelos, el procedimiento a seguir se centra en la utilización de los modelos semiformales que describen el comportamiento del sistema, y a partir de ellos la construcción de modelos formales que permitan emplear una técnica de verificación de modelos para determinar el cumplimiento de los requerimientos.

Los diagramas de estados realizados en UML a pesar de tener una sintaxis y una semántica definida, no dejan de ser especificaciones gráficas y estáticas del comportamiento dinámico de un sistema, por tanto, es necesario plantear los DE en términos de un modelo ejecutable y con una formalidad matemática como la que posee la estructura krepki que es una variación de los *Transitions Systems* [19] con los cuales también se puede describir el comportamiento de sistemas. Una comparación entre los DE y una estructura Krepki se presenta a continuación, ver Tabla 1.

Los DE y las estructuras Krepki comparten cierta información en común, el conjunto de estados S, las transiciones que en los DE esta definida por E y G y para las estructuras Krepki por Act, y el conjunto de transiciones en donde para los DE esta representado por el conjunto $Edges$ y para las estructuras Krepki por $\rightarrow \subseteq S \times Act \times S$. Esta información se puede obtener de forma automática a partir de los diagramas de estados para la creación de la estructura ejecutable, sin embargo, los estados iniciales deben ser especificados, y las proposiciones atómicas y la función de etiquetado deben ser construidas.

Para ello se estipulan los siguientes pasos para adecuar los modelos semiformales del sistema a modelos formales útiles para la validación de modelos:

- Convertir los diagramas de estados que describen el comportamiento del sistema en estructuras Kripke,
- Plantear las proposiciones atómicas en términos de los requerimientos y el comportamiento del sistema.
- Plantear las funciones LTL a verificar en términos de las proposiciones atómicas y las estructuras kripke desarrolladas.

Tabla 1. Caso comparativo de DE y LTS

Diagrama de Estados	**Estructura Krepki (LTS)**
$SC =< S, Var, A, E, G, Edges >$ • S es el conjunto de estados • E es el conjunto de eventos • Var es el conjunto de variables • G es el conjunto de guardas • $Edges$ es el conjunto de transiciones	$TS =< S, Act, \rightarrow, I, AP, L >$ • S es el conjunto de estados • Act es el conjunto de acciones • $\rightarrow \subseteq S \times Act \times S$, son las relaciones de transición • $I \subseteq S$ es el conjunto de estados iniciales • AP es el conjunto de proposiciones atómicas • $L: S \rightarrow 2^{AP}$

3.2.1. Planteamiento de Proposiciones Atómicas AP

Teniendo en cuenta que una AP es una acción oculta del sistema, y que estas se plantean en términos de los requerimientos del mismo, y que lo que se busca es la relación existente entre un estado $s_i \in S$ y una proposición $p_i \in AP$. Los requerimientos estipulados para el sistema se deben plantear en términos de acciones concretas de las cuales se representarán mediante un conjunto de etiquetas.

La relación entre las proposiciones atómicas y los estados se representa mediante la función de etiquetado $L: S \rightarrow 2^{AP}$, teniendo en cuenta que una AP usualmente se definen de la forma $(u_i \, igual \, v)$ en donde u_i es una variable de estado del sistema y v es el valor asociado a u_i. Y que la relación $(u_i \, igual \, v)$ es verdadera en todos los estados en los que u_i tiene un valor v. La función L se define en los mismos términos. Logrando de esta manera asociar los requerimientos del sistema con el comportamiento diseñado del mismo, y ver la ejecución del sistema en términos de los requerimientos.

3.2.2. Lógica Temporal Lineal LTL

La ejecución de un TS son acciones secuenciales (2^{AP}) donde TS se puede ejecutar a partir de un estado inicial I, un $TS = (S, Act, \rightarrow, I, AP, L)$ transita a $TS' =< S, Act, \rightarrow, I', AP, L >$ con una acción a, denotada como $TS \xrightarrow{a} TS' \leftrightarrow (I, a, I') \in \rightarrow$.

El operador "L" toma un LTS $ts =< S, Act, \rightarrow, I >$ y un conjunto proposiciones atómicas u acciones ocultas $AP \subseteq \tau$, y retorna $TS =< S, Act, \rightarrow, I, AP, L >$, en donde TS etiqueta los estados y las acciones visibles con el conjunto de acciones invisibles en el sistema.

El operador Composición *paralela* "||" es un operador conmutativo y asociativo que combina el comportamiento de dos o más LTSs mediante la sincronización de acciones comunes en sus alfabetos y la intercalación de las acciones restantes. Sea $TS_1 =< S^1, Act^1, \rightarrow^1, I^1, AP^1, L^1 >$ y $TS_2 =< S^2, Act^2, \rightarrow^2, I^2, AP^2, L^2 >$. Entonces $TS_1 || TS_2 =< S, Act, \rightarrow, I, AP, L >$, en donde $S = S^1 \times S^2$, $Act = Act^1 \cup Act^2$, $I = (I^1, I^2)$, y $\rightarrow$ se define de la siguiente manera, sea a una acción observable:

- $\frac{TS_1 \xrightarrow{a} TS'_1, a \notin Act^2}{TS_1 || TS_2 \xrightarrow{a} TS'_1 || TS_2}$
- $\frac{TS_1 \xrightarrow{a} TS'_1, TS_2 \xrightarrow{a} TS'_2}{TS_1 || TS_2 \xrightarrow{a} TS'_1 || TS'_2}$

Dado un conjunto de proposiciones atómicas PA, una formula LTL bien formada es definida inductivamente utilizando operadores booleanos, y los operadores temporales X (Siguiente) y U (Fuerte hasta) como sigue:

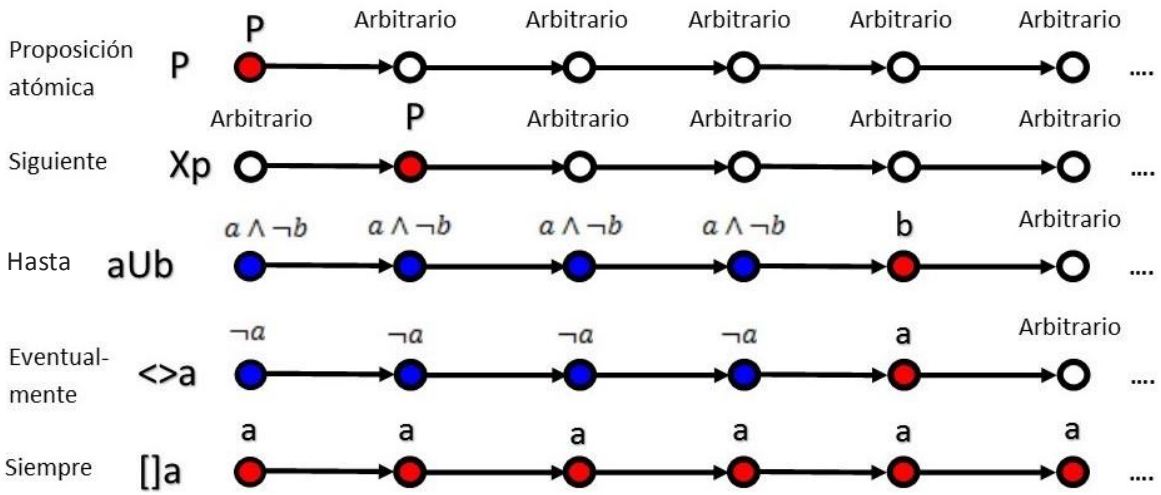
- Cada miembro de AP es una formula,
- Si φ y ψ son formulas, entonces $\neg\varphi$, $\varphi \vee \psi, \varphi \wedge \psi, X\varphi, \varphi U \psi$ también lo son.

Una interpretación de una formula LTL es una palabra infinita $w = x_0 x_1 x_2 \cdots$ sobre 2^{AP}. Es decir, una interpretación mapea a cada instante de tiempo un conjunto de proposiciones que se mantienen en ese instante. Sea w_i la palabra w iniciando en x_i. La semántica de un LTL se define de la siguiente forma, Observar la figura 16.

- $w \vDash p \; si \; p \in x_0, para \; p \in AP$
- $w \vDash \varphi \vee \psi \; si \; (w \vDash \varphi) \vee (w \vDash \psi)$
- $w \vDash \neg\varphi \; si \; \neg(w \vDash \varphi)$
- $w \vDash \varphi \wedge \psi \; si \; (w \vDash \varphi) \wedge (w \vDash \psi)$
- $w \vDash X\varphi \; si \; w_i \vDash \varphi$
- $w \vDash \varphi U \psi \; si \; \exists \; i \geq 0 \mid (w_i \vDash \psi) \wedge \; Para \; todo \; 0 \leq j \leq i, (w_i \vDash \varphi)$

Operadores y abreviaciones adicionales propuestos por Giannak opoulou [26] son: "$Verdadero \equiv \varphi \vee \neg\varphi$", "$falso \equiv \neg Verdadero$", el operador booleano $\Longrightarrow$ que se define: $\varphi \Longrightarrow \psi \equiv \neg\varphi \vee \psi$, el operador temporal F (Eventualmente), G (Siempre), W (Débil hasta) los cuales se definen en términos de los operadores temporales principales: $F\varphi \equiv Verdadero \; U\varphi$, $G\varphi \equiv \neg F \neg\varphi$, y $\varphi W \psi \equiv ((\varphi U \psi) \vee G\varphi)$.

Figura 16. Visión intuitiva de un LTL



3.3. UTILIZAR *TRANSITIONS SYSTEMS* PARA VERIFICAR LAS FORMULAS LTL

El esquema básico para la verificación de LTL se basa en el uso de los de Büchi automatons (BA) [25], un BA es una 5-tuple $B =< Q, \Sigma, \delta, q_0, F >$, donde Q es el conjunto de estados finitos, Σ es un conjunto finito de etiquetas, $\delta \subseteq Q \times \Sigma \times Q$ es la relación de transición de etiquetado, $q_0 \in Q$ es el estado inicial, y $F \subseteq Q$ es el conjunto de estados aceptados. Por lo siguiente, un B es la ejecución en una palabra finita $< a_0 a_1 a_2 \cdots >$ sobre Σ, que es una palabra infinita $< s_0 s_1 s_2 \cdots >$ sobre Q, tal que $s_0 = q_0$ y $\forall\, i \in N, (s_i, a_i, s_{i+1}) \in \delta$. Una ejecución es aceptada si algún elemento en F se produce infinitamente, una palabra infinita w sobre Σ es aceptado por el automaton B si existe alguna ejecución aceptada desde B in w.

Para una formula LTL sobre un conjunto de AP, un BA puede ser formulado para aceptar las palabras sobre 2^{AP} que satisface φ, los estados finitos de un sistema son verificado por una especificación LTL φ, y así calculando la intercepción entre el sistema de aplicación y el BA ($\neg\varphi$); Los TS satisface φ si la intercepción de palabras no es aceptada.

Las funciones LTL se encargan de verificar los requerimientos en los modelos de comportamiento de los sistemas, las funciones LTL están diseñados en términos de AP, para así determinar la función LTL donde es necesario utilizar con un conjunto de operadores, observar la tabla 2.

Tabla 2. Operadores para formulas LTL

Operadores Unitarios	**Operadores Binarios**
[] Siempre (G)	U Fuerte hasta
< > Eventualmente (F)	W débil hasta
X Siguiente	&& Operador lógico AND
! Negación Lógica	|| Operador lógico OR
	→ Implicación
	↔ Equivalencia

Una vez formuladas las funciones LTL, una por cada requerimiento, se procede a verificar dicha función empleando una herramienta adecuada de validación de modelos. Las herramientas para la verificación de modelos permiten evaluar las formulas LTL en los modelos ejecutables LTS construidos a partir de los modelos de comportamiento de los sistemas. Usualmente estas herramientas sintetizan algoritmos que permiten verificar el correcto funcionamiento y diseño de un sistema (Frappier, M., Fraikin, B., Chossart, R., Chane-Yack-Fa, R., & Ouenzar, 2010). Existen múltiples herramientas que permiten llevar a cabo procesos de verificación de modelos como los postulados en este trabajo. A continuación, Tabla 3, se presentan algunas herramientas frecuentemente mencionadas en la literatura y que pueden ser utilizadas para la implementación del proceso de evaluación de modelos estipulado anteriormente.

Cualquiera de las herramientas mencionadas puede ser empleada para verificar propiedades LTL en modelos LTS. Sin embargo, para el desarrollo de este trabajo la herramienta utilizada fue LTSA *Analyser,* la cual en comparación con las demás herramientas mencionadas resulta más sencilla de utilizar e interpretar sus resultados.

La validación concreta de un requerimiento en particular puede generar tres resultados diferentes:

- La propiedad (requerimiento) se cumple,
- La propiedad (requerimiento) no se cumple y se muestra un contraejemplo,
- La prueba no genera un resultado concluyente para la propiedad (requerimiento) evaluada.

Tabla 3. Herramientas para verificación de modelos que permitan verificar propiedades LTL en LTS

Herramienta	Lenguaje	Propiedades	Referencia
SPIN	Promela	LTL	(Holzmann, 1997)
NuSMV	SMV	CTL, LTL, PSL	(Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., ... & Tacchella, 2002)
UPPAAL	Timed Automata, C	TCTL, LTL	(Larsen, K. G., Pettersson, P., & Yi, 1997)
LTSA	FSP	LTL	(Magee, Kramer, & Uchitel, 2004
ProB	B-Method, Event- B, Z, TLA+, CSP	Assertions, LTL, CTL	(Leuschel, M., & Butler, 2003)
Spot	Petri nets, DVE Input Language	LTL, PSL	(Duret-Lutz, A., & Poitrenaud, 2004)

3.4. ANÁLISIS DE CAPACIDADES USANDO MODEL CHECKING

Una capacidad de un sistema representa la habilidad de realizar o alcanzar ciertas acciones o resultados a través de un conjunto de características, funciones, procesos, servicios, facultades controlables y medibles; desde un punto de vista arquitectural, las capacidades se modelan para satisfacer las necesidades de los interesados y los objetivos de la organización. Por lo siguiente se entiende que para un conjunto de requerimientos $R = \{r_1, r_2, \cdots, r_m\}$, se diseñan un conjunto de capacidades $C = \{c_1, c_2, \ldots, c_n\}$ que son encargados de satisfacer los requerimientos establecidos por los expertos; las capacidades C están representados en la arquitectura como un conjunto de modelos de comportamiento que despliegan estas capacidades en términos de actividades operacionales.

Para el caso de estudio de este trabajo se propone un enfoque como verificación de capacidades utilizando un esquema de model-checking, esté enfoque es basado en las siguientes suposiciones:

i. Una capacidad es diseñada para satisfacer, uno o muchos requerimientos.

ii. Un requerimiento es una condición o capacidad necesaria por un usuario para así resolver un problema o alcanzar un objetivo.
iii. Las capacidades proveen un entorno que permite lograr las metas de la organización.

Por consiguiente, el análisis de capacidades usando la técnica de model-ckecking se realiza presentando los pasos generales a seguir.

i. Relacionar las capacidades con los requerimientos.
ii. Validar los requerimientos usando un enfoque Model-Checking.
iii. Analizar y categorizar los resultados en términos de capacidades.

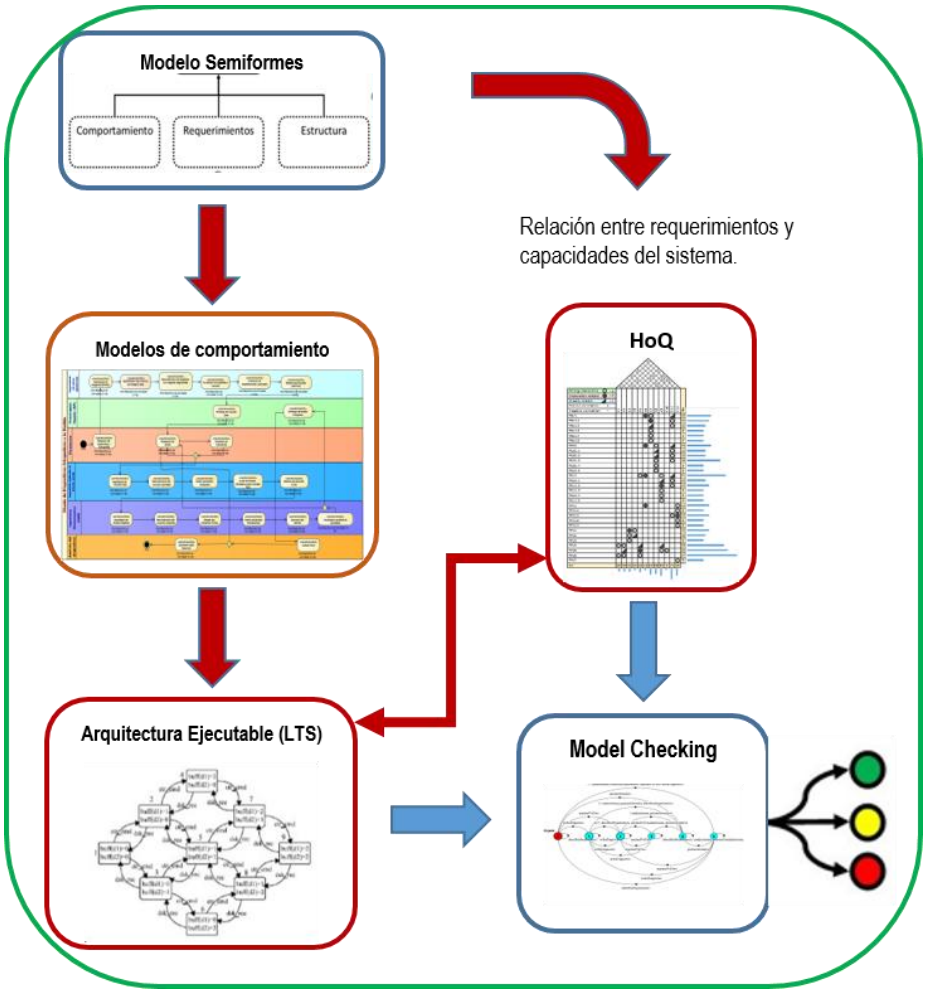
3.4.1. Método de Análisis de Capacidades

La figura 17 ilustra los pasos fundamentales que se llevan a cabo con la técnica model – checking para verificar el cumplimiento de las capacidades de un sistema. Se inicia extrayendo información a partir de modelos semiformales o artefactos arquitecturales para establecer el modelo de comportamiento y la relación entre requerimientos y capacidades. Con esto establecido se procede a la construcción de un modelo del comportamiento dinámico del sistema haciendo uso de LTS. La utilización de este modelo formal la reorganización de los requerimientos de acuerdo a las capacidades en el sistema y la utilización de una herramienta de verificación de modelos (model-checking) permite determinar el cumplimiento o no de una capacidad.

3.4.1.1. Relacionar las Capacidades con los Requerimientos

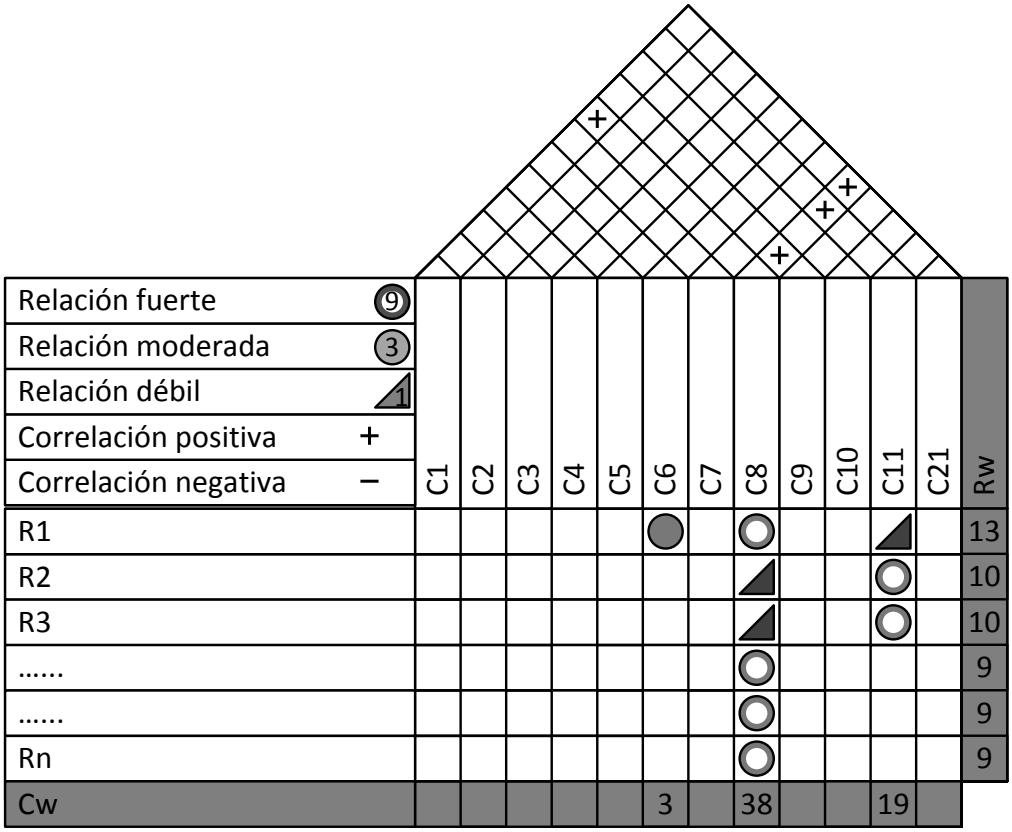
La primera asunción, una capacidad es diseña para satisfacer uno o más requerimientos. Esta suposición sugiere que una capacidad puede ser analizada a partir de un conjunto de requerimientos. Sin embargo, una capacidad es un componente formal del sistema, pero los requerimientos son necesidades informales de los *stakeholders*. Las propiedades de interés en el sistema son las capacidades. Para asociar capacidades con requerimientos el modelo a seguir sugiere utilizar una adaptación de la casa de la calidad para relacionar la voz de los interesados con los aspectos formales del sistema, ver figura 18. Las propiedades de interés en el sistema son las capacidades. Se denota la relación entre las capacidades y los requerimientos como fuertes ⓪, moderadas ○, o débiles Δ, y también en esta etapa se denotan la relación entre capacidades como positivamente correlacionadas +, y negativamente correlacionadas -, Para crear esta relación se necesitan los modelos CV-2 para conocer cuáles son las capacidades y el modelo CV-4 Para conocer la relación entre capacidades, así como los requerimientos del sistema.

Figura 17. Esquema general para proceso de validación de modelos.



A partir de la relación entre capacidades y requerimientos, se obtiene un vector de pesos asociados a las capacidades C_w, así como un vector de pesos asociado a los requerimientos R_w. Del cuadro en la HOQ, se obtiene para cada capacidad los vectores $C_{R,i}$ to $i = 1,2,\cdots n$, donde n es el número de capacidades en el sistema. $C_{R,i}$ Contiene todos los requerimientos (fuertes, moderados, o débiles) relacionados con la capacidad C_i. Además, se obtiene también la matriz de correlación entre capacidades C_c que contiene las relaciones entre las capacidades, ver el techo de la HOQ en la figura 18.

Figura 18. Relación entre capacidades y requerimientos (Tomado de [27])



3.4.1.2.Validar los Requerimientos Utilizando Model-Checking

Para el análisis de capacidades se necesita: un modelo de comportamiento del sistema el cual representa las capacidades diseñadas para satisfacer los requerimientos, se necesita también las formulas LTL para cada requerimiento, y una herramienta que permita mediante una técnica de *model-checking* verificar la valides de las LTL en el modelo de comportamiento.

A partir del vector $C_{R,i}$, que contiene los requerimientos para la capacidad C_i, se definen tres vectores adicionales $R_{s,i}, R_{a,i}, R_{l,i} \in R$ en donde $R_{s,i} \cap R_{a,i} \cap R_{l,i} = \emptyset$, los cuales representan los requerimientos fuertes, moderados y débiles relacionados con la capacidad C_i.

Adicionalmente, para las formulas LTL relacionadas con los requerimientos $R_{s,i}, R_{a,i}, R_{l,i}$, se formulan los vectores $\varphi_{s,i}, \varphi_{a,i}, \varphi_{l,i}$ los cuales contienen las formulas LTL para su respectivo conjunto de requerimientos (fuertes, moderados, y débiles). Sea ξ el modelo de

comportamiento, empleando una herramienta para *model-checking* se procede a verificar si el modelo ξ satisface las formulas $\varphi_{s,i} \Vdash \xi$, $\varphi_{a,i} \Vdash \xi$, $\varphi_{l,i} \Vdash \xi$.

Cuando el conjunto de propiedades $\varphi_{s,i}, \varphi_{a,i}, \varphi_{l,i}$ es verificado, se puede obtener los vectores $O_{s,i}, O_{a,i}, O_{l,i}$ que contienen los resultados del proceso de verificación de los requerimientos en los modelos de comportamiento.

Para definir si una capacidad es alcanzada, el siguiente conjunto de variables es definido, sea $N_i$el numero de propiedades a verificar en ξ para la capacidad C_i, sea $K_i, L_i, T_i \in \mathbb{N}$ el número de propiedades evaluadas en $O_{s,i}, O_{a,i}, O_{l,i}$, sea $\alpha_{s,i}, \beta_{s,i}$, $\gamma_{s,i}$ los resultados positivos, negativos e inconclusos obtenidos relacionados con las propiedades fuertes $\varphi_{s,i}$, de la misma forma se definen $\alpha_{a,i}, \beta_{a,i}$, $\gamma_{a,i}$ para las propiedades moderadas $\varphi_{a,i}$, y $\alpha_{l,i}, \beta_{l,i}$, $\gamma_{l,i}$ para las propiedades débiles $\varphi_{l,i}$. Para realizar el análisis de capacidades se definen las ecuaciones 1, 2, y 3 que indican el número de propiedades a verificar para los requerimientos fuertes, moderados y débiles relacionados con una capacidad en particular C_i, y la ecuación 4 representa el número total de requerimientos relacionados con la misma capacidad.

$$\alpha_{s,i} + \beta_{s,i} + \gamma_{s,i} = K_i \quad (1)$$

$$\alpha_{a,i} + \beta_{a,i} + \gamma_{a,i} = L_i \quad (2)$$

$$\alpha_{l,i} + \beta_{l,i} + \gamma_{l,i} = T_i \quad (3)$$

$$K_i + L_i + T_i = N_i \quad (4)$$

Para un conjunto en particular $O_{s,i}, O_{a,i}$, o $O_{l,i}$ hay seis posibles tipos de resultados cuando se realiza el proceso de *model-checking*.

i. Para la capacidad C_i, todo el conjunto de propiedades se cumplen,
ii. Algunas propiedades no se cumplen y se muestran los contraejemplos
iii. Ninguna propiedad se cumple y se muestran los contraejemplos
iv. Algunos resultados son no concluyentes
v. El conjunto complete de propiedades es no concluyente
vi. Algunas propiedades del conjunto de propiedades se cumplen

En una verificación de capacidades utilizando *model-checking*, es posible obtener una combinación de resultados cuando los tres conjuntos de propiedades $O_{s,i}, O_{a,i}$, y $O_{l,i}$ se tienen en cuenta. Para resumir los resultados obtenidos se presenta la tabla 4.

Una explicación más detallada de las ecuaciones en la tabla 4, se tiene en la tabla 5, en donde se muestran los seis posibles resultados a obtener.

Tabla 4. Análisis de resultados para la capacidad Ci (Tomado de [13])

	Os_j	Oa_j	Ol_j
i	$\alpha_s = k$	$\alpha_a = l$	$\alpha_l = t$
ii ∧ iv	$\beta_s + \gamma_s = k$	$\beta_a + \gamma_a = l$	$\beta_l + \gamma_l = t$
iii	$\beta_s = k$	$\beta_a = l$	$\beta_l = t$
v	$\gamma_s = k$	$\gamma_a = l$	$\gamma_l = t$
ii ∧ iv ∧ vi	$\alpha_s + \beta_s + \gamma_s = k$	$\alpha_a + \beta_a + \gamma_a = l$	$\alpha_l + \beta_l + \gamma_l = t$
iv ∧ vi	$\alpha_s + \gamma_s = k$	$\alpha_a + \gamma_a = l$	$\alpha_l + \gamma_l = t$
ii ∧ vi	$\beta_s + \alpha_s = k$	$\beta_a + \alpha_a = l$	$\beta_l + \alpha_l = t$

De acuerdo con las tablas 4 y 5, solo una de las seis opciones es seleccionada como salida para los conjuntos $O_{s,i}$, $O_{a,i}$, $O_{l,i}$. Para generar un resultado cuantitativo relacionado con el análisis de las salidas provenientes del proceso de *model-checking*, se propone asignar un valor a cada posible resultado obtenido, tabla 6.

Los valores en la tabla 6 se relacionan con el nivel de confianza en el posible impacto sobre las capacidades que puede tener el cumplimiento de un requerimiento. Bajo el supuesto de que el incumplimiento de un requerimiento tiene un mayor impacto sobre la capacidad si proviene de un requerimiento fuertemente relacionado que si este proviene de uno medianamente o débilmente relacionado. Por ello se observa en la columna que los requerimientos fuertemente relacionados tienen una mayor penalización que la de los medianamente relacionados, y esta a su vez tiene una mayor penalización que la de los débilmente relacionados.

Tabla 5. Explicación de los seis posibles tipos de resultados en el proceso de análisis de capacidades (Tomado de [13])

Caso	**Resultado**	**Explicación**
Caso 1	i	Todas las propiedades se cumplen
Caso 2	ii & iv	Algunas propiedades no se cumplen y las propiedades restantes son no concluyentes.
Caso 3	iii	Ninguna propiedad se cumple
Caso 4	v	Todas las propiedades tienen resultados no concluyentes
Caso 5	ii & iv & vi	Algunas propiedades no se cumplen, las propiedades restantes, algunas son no concluyentes y algunas se cumplen.
Caso 6	iv & vi	Algunos resultados son no concluyentes y las propiedades restantes se cumplen.
Caso 7	ii & vi	Algunas propiedades no se cumplen y las propiedades restantes se cumplen.

Tabla 6. Asignación de valores propuesta para los resultados obtenidos al evaluar. $O_{s,i}$, $O_{a,i}$, $O_{l,i}$ *(Tomado de [13])*

	Of_j	Om_j	Od_j
i	3	3	3
$ii \wedge iv$	-2	-2	-1
iii	-3	-3	-3
v	0	0	0
$ii \wedge iv \wedge vi$	-1	0	1
$iv \wedge vi$	1	2	2

Para poder obtener un resultado general, es decir que resuma los resultados ubicados en la tabla 6 para los conjuntos $O_{s,i}$, $O_{a,i}$, $O_{i,i}$. Se propone un valor que generalice el alcance de una capacidad, ver Ecuación 5. $\chi \in \{-3,3\}$ Representa el valor de certidumbre asociado con el cumplimiento de los requerimientos para la capacidad C_i, ver tabla 7.

$$\chi = ceiling \quad \frac{Of_J + Om_J + Od_J}{3} \qquad (5)$$

Tabla 7. Niveles de certidumbre para el cumplimiento de una capacidad (Tomado de [13])

Valor de Certidumbre	**Interpretación de Resultado**
-3	Hay un alto nivel de certidumbre de incumplimiento
-2	Hay un nivel medio de certidumbre de incumplimiento
-1	Hay un mínimo nivel de certidumbre de incumplimiento
0	No hay certidumbre de cumplimiento o incumplimiento
1	Hay un mínimo nivel de certidumbre de cumplimiento
2	Hay un nivel medio de certidumbre de cumplimiento
3	Hay un alto nivel de certidumbre de cumplimiento

3.4.1.3.Analizar y categorizar los resultados en términos de capacidades

Para el sistema en general, se puede obtener un análisis de cumplimiento de las capacidades, e incluso un indicador de cumplimiento, ver Tabla 8.

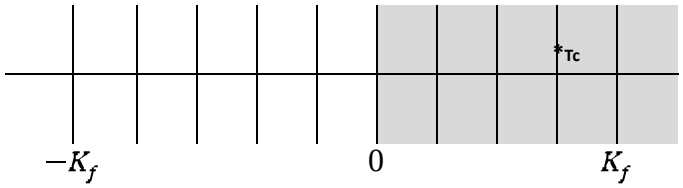
Sea $\chi_{t,i} = \chi_i \times C_{w,i}$ el factor de cumplimiento para la capacidad C_i, $T_i = C_{w,i}$ es el aporte total de las capacidades, $T_c = \sum_{i=1}^{n} \chi_{t,i}$ es el indicador de cumplimiento general de las capacidades respecto al sistema, y $K_f = T_i \times 3$ es el dominio en el que existe $\chi_{t,i} = \chi_i$.

Una representación gráfica del indicador de cumplimiento es desarrollada para analizar las implicaciones de los resultados en el análisis de capacidades, ver Figura 19.

Tabla 8. Análisis de cumplimiento e indicador de cumplimiento(Tomado de [13])

Capacidades	$C_{w,i}$	Factor de impacto							Medición
		Bajo	-2	-1	0	1	2	Alto	$\chi_{t,i} = \chi_i \times C_{w,i}$
Capacidad 1	$C_{w,1}$					x			$\chi_{t,1} = \chi_1 \times C_{w,1}$
Capacidad 2	$C_{w,2}$				x				$\chi_{t,2} = \chi_2 \times C_{w,2}$
Capacidad …	$C_{w,\dots}$	x							$\chi_{t,\dots} = \chi_{\dots} \times C_{w,\dots}$
Capacidad n	$C_{w,n}$						x		$\chi_{t,n} = \chi_n \times C_{w,n}$
Total importancia T_i:	$\sum_{i=1}^{n} C_{w,i}$	Total medición T_c:							$\sum_{i=1}^{n} \chi_{t,i}$
								$K_f = T_i \times 3$	

Figura 19. Representación gráfica del indicador de cumplimiento



Finalmente se revisa la matriz de correlación entre capacidades C_c, con la cual se puede determinar las implicaciones en el sistema producto del cumplimiento o incumplimiento de una capacidad en particular C_j.

4. ANÁLISIS DE RESULTADOS DEL CASO DE APLICACIÓN USANDO LA METODOLOGÍA MODEL-CHECKING

Definidos los conceptos involucrados en este trabajo; Este capítulo presenta la aplicación en SMOM del modelo de verificación de modelos arquitecturales mediante la utilización de una técnica *model checking* enfocada en el análisis de las capacidades.

4.1. RELACIÓN DE CAPACIDADES Y REQUERIMIENTOS

Lo que primero se asume es que una capacidad está diseñada para satisfacer uno o muchos requerimientos, se propone que la capacidad puede estar compuesta por un conjunto de requerimientos, sin embargo, una capacidad es un componente formal del sistema, pero un requerimiento es una necesidad informal por parte de los interesados.

Para el caso de aplicación las capacidades se extraen de la vista CV-2 del marco arquitectural DoDAF, donde el modelo representa una jerarquía de capacidades; es decir, estas capacidades se muestran en un contexto de línea de tiempo donde puede mostrar las capacidades actuales y futuras. A continuación, en la tabla 9 se muestra la lista de capacidades del caso de aplicación.

Una de las técnicas eficientes y organizadas para adquirir y traducir la voz de los interesados y de los diseñadores por parte de ingeniería es el proceso de función de despliegue de calidad (QFD) [27]; QFD es una herramienta que se utiliza para asociar las necesidades o requerimientos ambiguos y cualitativos dentro de los atributos de un sistema. Para la aplicación de este trabajo es necesario una relación entre la adaptación del QFD y la casa de calidad HOQ y así poder relacionar los requerimientos o las necesidades de los interesados con propiedades específicas del sistema; principalmente es necesario resaltar que las propiedades de gran interés para la aplicación de este trabajo son las capacidades del sistema. Las relaciones entre los requerimientos y capacidades se indica con unos valores como:

- ✓ ⓪ si la relación es fuerte.
- ✓ ○ si la relación es moderada o media.
- ✓ Δ si la relación es débil.

Las relaciones entre las capacidades, son correlaciones positivas +, y correlaciones negativas -, como se puede observar en la figura 20.

Tabla 9. Lista de capacidades obtenidas por el modelo CV-2

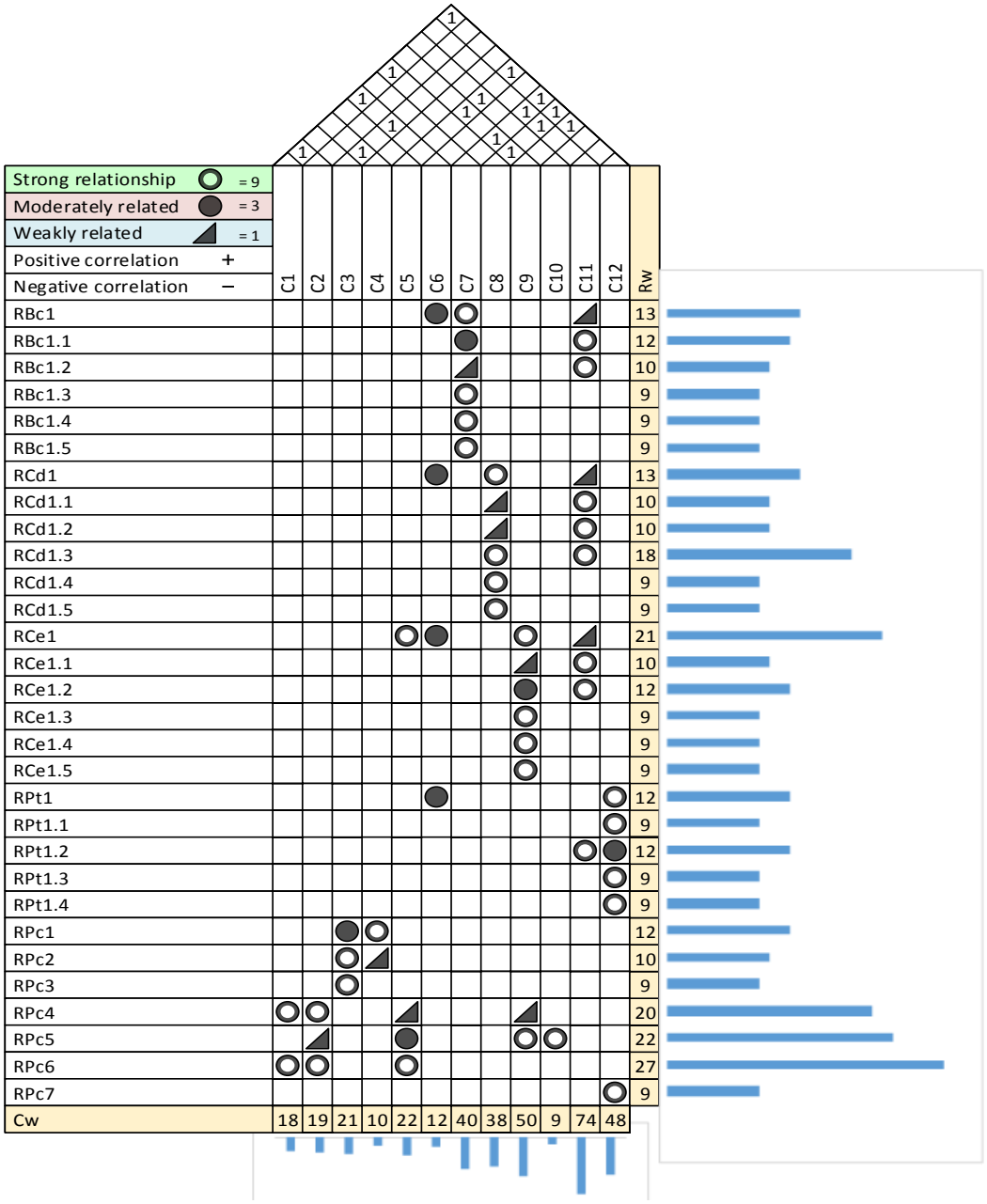
CODIGO	CAPACIDAD
C1	Planear Diseño
C2	Compartir Información (Interdisciplinaria)
C3	Adquirir Conocimiento
C4	Transferir Conocimiento
C5	Evaluar Productos
C6	Apropiación tecnológica
C7	Reconstrucción Ósea Digital
C8	Modelado de Implantes
C9	Simulación Estática Estructural
C10	Simulación Quirúrgica
C11	Interoperabilidad
C12	Prototipado

En la casa de calidad para las relaciones de capacidades y requerimientos, se obtiene un vector C_w para dar un valor ponderado a cada una de las capacidades y R_w para los valores de los requerimientos; a partir de la figura 19, se obtiene para cada capacidad el vector $C_{R,i}$ donde $i = 1,2,\cdots n$, y donde n son las capacidades del sistema, $C_{R,i}$ se encarga de contener todos los requerimientos fuertes, moderados y débil, así mismo se encarga de relacionarlos con las capacidades C_i, observen el ejemplo dado en la tabla 10. Adicionalmente HOQ obtienen una correlación entre las capacidades del sistema de diseño de productos óseos hechos a la medida, como se muestra en la figura 20.

Tabla 10. Relación de requerimientos con la capacidad C8

$C_{R,8}$	
RCd1	Fuerte
RCd1.1	Débil
RCd1.2	Débil
RCd1.3	Fuerte
RCd1.4	Fuerte
RCd1.5	Fuerte

Figura 20. Casa de calidad para la relación entre capacidades y requerimientos



4.2.COMPROBACIÓN DE MODELOS UTILIZANDO LTL.

Los modelos de comportamiento y los sistemas de transición definidos en los capítulos anteriores, se utilizan para representar el comportamiento y la comunicación entre los componentes del sistema de diseño de productos ortopédicos a la medida; Estos sistemas de

transición forman una tupla $(S, Act, \rightarrow, I, AP, L)$. A partir del modelo de reglas operacionales OV-6ª se obtienen las proposiciones atómicas AP con las cuales se puede definir la función de etiquetado L; como se menciona en el capítulo anterior, las AP son consideradas como acciones ocultas en el sistema y su punto de acceso se definen en términos de requerimientos que son establecidos por los interesados y los expertos. La tabla 11 representa el conjunto de AP por cada uno de los requerimientos establecidos por los expertos e interesados.

Tabla 11. Lista de requerimiento relacionada con las proposiciones atómicas obtenidas del modelo OV-6a.

Código	**Requerimiento**	**Propiedades Atómicas**
RBc1	El en proceso BIOCAD, la tecnología debe permitir la reconstrucción de las tomografías del trauma musculo esquelético usando técnica de ingeniería inversa.	
RBc1.1	El software debe permitir la importación de Tomografías Axiales Computarizadas mediante archivos en formato universal DICOM.	importarTomografia
RBc1.2	El software BIOCAD debe permitir la exportación del volumen 3D usando formatos de archivo de intercambio IGES, STEP o STL.	suavizarVolumen exportarArchivoSTL, exportarArchivoSTEP, exportarArchivoIGES
RBc1.3	Se debe permitir la selección de región de imagen mediante mascaras	identificarContorneoOseo, importarTomografia
RBc1.4	Se debe poder reconstruir volúmenes mediante calculo 3D	reconstruirSuperficie, identificarContorneoOseo
RBc1.5	Se debe poder realizar la corrección de imperfecciones originadas en el proceso de calculo	corregirImperfecciones, reconstruirSuperficie
RBc1.6	Se debe poder realizar el suavizado de la superficie volumétrica	suavizarVolumen, corregirImperfecciones
RCd1	En el proceso CAD, se debe diseñar el dispositivo ortopédico a la medida de los modelos propuestos por los especialistas para cada caso de estudio.	
RCd1.1	El software debe permitir la importación de modelos virtuales 3D de geometrías	generarAlternativa importarArchivoSTL,

	óseas reconstruidos en software BIOCAD mediante el uso de archivos de intercambio de modelo 3D (IGES, STEP o STL)	importarArchivoSTEP, importarArchivoIGES
RCd1.2	El software debe permitir la exportación del volumen 3D usando formatos de archivo de intercambio IGES, STEP, o STL.	valorarPrecisionModelo exportarArchivoSTL, exportarArchivoSTEP, exportarArchivoIGES
RCd1.3	Se debe mantener el historial de operaciones de construcción del modelo 3D permitiendo regresión	reconstruirVolumen, importarArchivoSTL, importarArchivoSTEP, importarArchivoIGES
RCd1.4	Se debe permitir la edición de parámetros de las operaciones de construcción del modelo	diseñarModelo, rechazarDiseño, reconstruirVolumen,
RCd1.5	Se deben poder hacer planos técnicos utilizando normas estándar	diseñarModelo, reconstruirVolumen,
RCe1	Al prototipo ortopédico diseñado se le deben realizar análisis biomecánicos virtual, para así detectar falencias.	
RCe1.1	El software debe permitir la importación de modelos 3D mediante archivos de intercambio de formatos IGES, STEP, o STL.	[exportarArchivoSTL, exportarArchivoSTEP, exportarArchivoIGES] [importarArchivoSTL, importarArchivoSTEP, importarArchivoIGES]
RCe1.2	El software debe disponer de una base de datos de materiales Biocompatibles (Ti6Al4V, A316L) o permitir la creación de un nuevo material.	consultarMateriales crearMateriales, mallarElementoFinitos
RCe1.3	Se debe poder analizar el comportamiento mecánico de un conjunto mediante el diseño de un estudio estático realizado mediante elementos finitos (FEA)	consultarMateriales crearMateriales, ejecucionProceso
RCe1.4	Se deben proporcionar los datos de análisis de tensiones, desplazamientos, deformaciones unitarias y factos de seguridad resultantes de estudios estáticos	analizarResultados, ejecucionProceso
RCe1.5	Se debe poder realizar estudios estáticos a conjuntos de piezas	
RCe1.6	Se debe poder seleccionar del sistema de ajuste "no separación" entre piezas de un conjunto.	
RPt1	El sistema debe proporcionar los prototipos necesarios para realizar análisis	imprimirModelo, imprimirVolOseo

	y pruebas de verificación en cada una de las fases requeridas.	
RPt1.1	El tamaño de impresión de los prototipos debe ser igual o superior a 30 centímetros en 3D.	
RPt1.2	Archivo de entrada de datos para el inicio del prototipado rápido deben ser en formato STL.	importarArchivoSTL
RPt1.3	Debe soportar materiales de impresión ABS o PLA	
RPt1.4	El material de soporte debe ser removible con agua	
RPc1	El conocimiento adquirido por parte del grupo de trabajo debe ser proporcionado a los nuevos integrantes por medio de archivos, tutoriales y capacitaciones	
RPc2	Los nuevos integrantes del grupo de trabajo deben se capacitar antes de iniciar el proyecto.	
RPc3	Los integrantes del grupo se deben especializar en un área del sistema musculo esquelético.	
RPc4	Los diagnósticos para cada uno de los casos se deben definir con los especialistas	recibirDiagnostico imprimirVolOseo analizartrauma identificarRequerimiento generarAlternativa
RPc5	Se deben realizar los procesos de simulación para el abordaje quirúrgico.	imprimirModelo imprimirVolOseo analizarPrototipo aprobarDesarrollo rechazarPrototipo
RPc6	El sistema debe ofrecer un diseño de prótesis ajustado a las necesidades de cada paciente.	ajustarVolumen, valorarPrecisionModelo, diseñarModelo rechazarDiseño exportarArchivoSTL, exportarArchivoSTEP, exportarArchivoIGES
RPc7	El sistema debe proveer modelos de los traumas musculo esquelético en 3d.	imprimirVolOseo, suavizarVolumen,

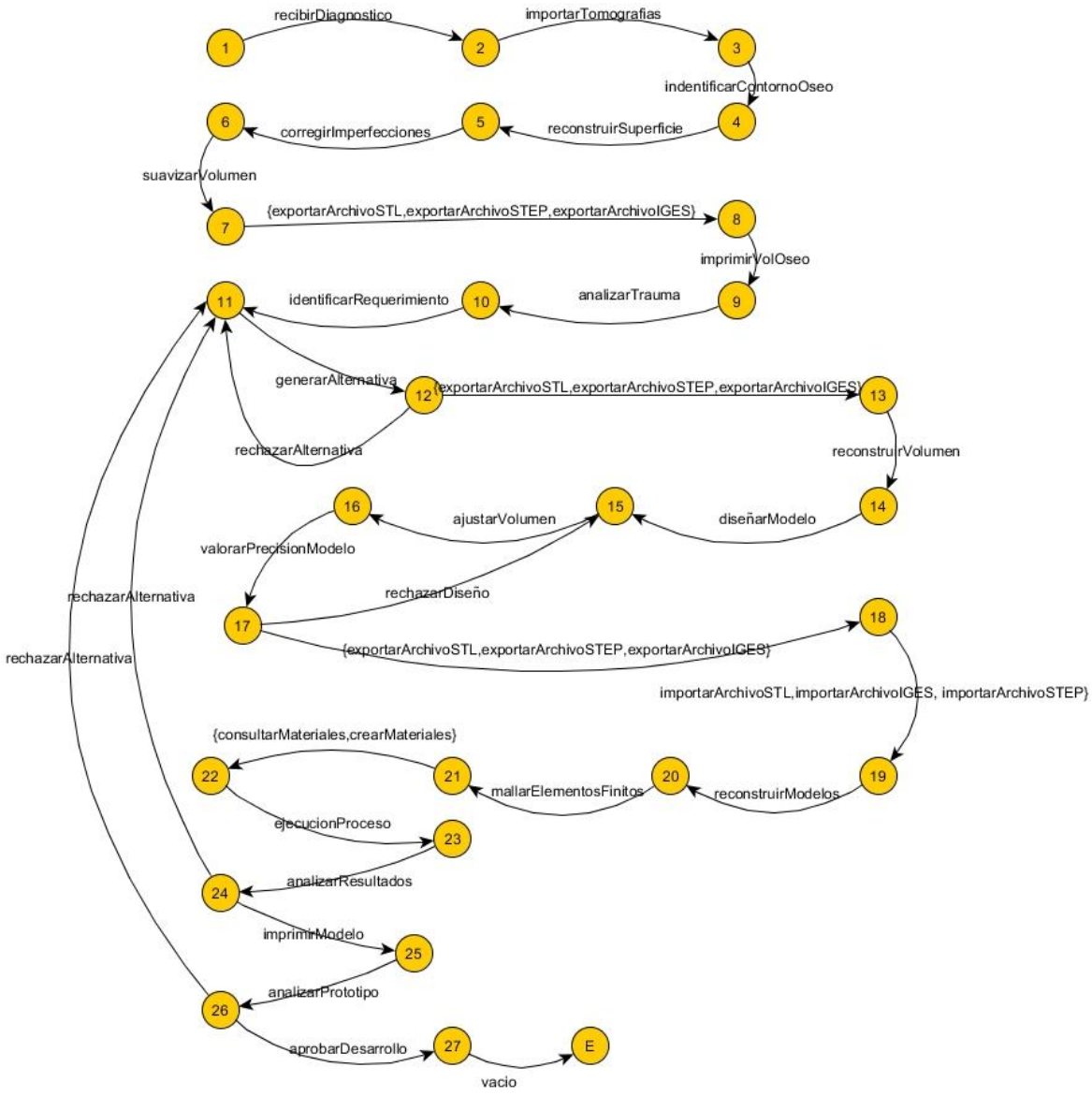
Por otro lado, las funciones de etiquetados L crean una relación entre los estados y sus proposiciones atómicas, ver tabla 12. Con los AP y la función L se crea una relación entre los requerimientos del sistema y los modelos de comportamiento, por lo cual antes de transformar el modelo de descripción de estados de transición OV-6b en un sistema de transición TS, es necesario representar la etiqueta en cada uno de los requerimientos del sistema donde las etiquetas representan acciones específicas, estas etiquetas son obtenidas del modelo OV-6a, donde anteriormente se menciona que este diagrama representa las actividades del sistema, el OV-6a contiene un conjunto detallado de operaciones, actividades y guardas, donde se especifica cada uno de los estados y se satisface cada uno de los requerimientos, en resumen, la función de etiquetado L permite la ejecución de los TS en términos de los requerimientos del sistema. El diagrama para el sistema de diseño de productos ortopédicos en TS se muestra en la figura 21.

Tabla 12. Lista de relación entre las preposiciones atómicas y los estados obtenidos del modelo OV-6b.

Estados	**Propiedades Atómicas**
Recopilación de Información	recibirDiagnostico
Import. Imágenes DICOM	importarTomografia
Región Ósea Identificada	identificarContorneoOseo
Superficie Segmentada	reconstruirSuperficie
Corrección de Imperfección	corregirImperfecciones
Volumen Suavizado	suavizarVolumen
Mallado Para Exporta (BIOCAD)	exportarArchivoSTL, exportarArchivoSTEP, exportarArchivoIGES
Prototipo Volumen Óseo	imprimirVolOseo
Análisis del Trauma	Analizartrauma
Planeación	identificarRequerimiento, rechazarAlternativa, rechazarModelo, rechazarPrototipo
Valoración de Alternativas	generarAlternativa

Import de Volumen Óseo	importarArchivoSTL, importarArchivoSTEP, importarArchivoIGES
Reconstrucción de Volumen Importado	reconstruirVolumen
Diseño	diseñarModelo, rechazarDiseño
Ajuste al Volumen Óseo	ajustarVolumen
Valoración de Precisión	valorarPrecisionModelo
Mallado para exportar CAD	exportarArchivoSTL, exportarArchivoSTEP, exportarArchivoIGES
Import. Modelo Diseñado	importarArchivoSTL, importarArchivoSTEP, importarArchivoIGES
Reconstrucción de Volumen e Implante	reconstruirModelo
Discretización	mallarElementoFinitos
Diseño Biomecánico (pre-proceso)	consultarMateriales, crearMateriales
Ejecución de Estudio (Proceso)	ejecucionProceso
Visualización y Análisis	analizarResultados
Prototipado de Modelo	imprimirModelo
Análisis Físico	analizarPrototipo
Aprobación Para Desarrollo	aprobarDesarrollo

Figura 21. Modelo ejecutable (Labeled Transition System – LTS) en términos de los requerimientos para el modelo OV-6b



4.3. FORMULAS LTLS USANDO SISTEMAS DE TRANSICIONES TS

Como se mencionó en el capítulo anterior, la verificación de LTL se basa en el uso de los autómatas de Büchi (BA), para la creación de las formulas LTL se necesitan las AP y los requerimientos del sistema, donde un BA puede ser formulado para aceptar las palabras 2^{AP} obtenidas, donde los estados finitos del sistema son verificados y así calcular la intersección entre se sistema de diseño de modelos ortopédicos óseos a la medida y el BA,

por lo cual las funciones LTL se encargan de verificar el conjunto de requerimiento en cada capacidad del sistema y para poder definir las funciones LTL es necesario utilizar el conjunto de operadores que se muestran en la tabla-3, por lo siguiente una vez ya formulada las funciones LTL, donde cabe aclarar que es una función por cada requerimiento se procede a verificar dicha función empleando una herramienta adecuada para la validación de modelos.

Para el caso de aplicación de este trabajo se muestra en la tabla 13 cada una de las funciones requeridas por cada requerimiento de los procesos y software CAD.

Tabla 13. Formulas LTL para los requerimientos CAD

Requerimiento	**Formula LTL**
RCd1.1	Rcd11= <>((generarAlternativa)->(importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP))
RCd1.2	Rcd12 = <>((valorarPrecisionModelo)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) W (importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP)))
RCd1.3	Rcd13 = <>((importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP) -> (reconstruirVolumen))
RCd1.4	Rcd14 = (<>(reconstruirVolumen ->((diseñarModelo W rechazarDiseño)))->diseñarModelo)
RCd1.5	Rcd15 = <>((reconstruirVolumen) -> (diseñarModelo))

4.4. ANÁLISIS DE CAPACIDADES

El análisis de capacidades es necesario ya que nos representa la satisfacción de los requerimientos que fueron estipulados por los interesados y para el desarrollo de este trabajo es necesario ya tener definidos las formulas LTL por cada uno de los requerimientos como se muestra en la tabla 10. Por consiguiente, el conjunto de propiedades es probados o se le hacen pruebas para cada capacidad, de acuerdo con lo anterior cada capacidad contiene un conjunto de requerimientos para ser probados, en la tabla 14 se muestra los resultados obtenidos de cada una de las propiedades que son probadas para el caso de aplicación donde se muestra la capacidad modelado de implantes C8. Cabe mencionar que para el conjunto de requerimientos de la capacidad C8 se dividen en tres subconjuntos, lo

cuales son los requerimientos fuertes, los requerimientos medios y los requerimientos débiles.

Tabla 14. Reporte de fórmulas validadas para la capacidad C8

Sub-conjunto	Propiedad	Salida
$O_{s,8}$	Rcd13 = <>((importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP) -> (reconstruirVolumen))	✓
	assert Rcd14 = (<>(reconstruirVolumen ->((diseñarModelo W rechazarDiseño)))->diseñarModelo)	✓
	Rcp61 = <> ((diseñarModelo)->(ajustarVolumen)) Rcp62 = <> ((Rcp61)->(valorarPrecisionModelo)) Rcp63 = <> ((Rcp62)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) || (rechazarDiseño)))	✓
$O_{a,8}$		
$O_{i,8}$	Rcd11= <>((generarAlternativa)->(importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP))	✓
	Rcd12 = <>((valorarPrecisionModelo)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) W (importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP)))	✓

Para la comprobación de las funciones LTL se utilizó la herramienta Ltsa tools, que es una herramienta de verificación de sistemas concurrentes, por lo cual se comprueba mecánicamente que las especificación de un sistema concurrente satisface las propiedades requeridas por su comportamiento, este trabaja realizando análisis de accesibilidad de la composición que se busca exhaustivamente por violaciones de propiedades deseadas por medio de los LTS que contiene todos los estados que describe el sistema. El autómata obtenido utilizando la herramienta Ltsa tools se puede observar en el anexo J. Todos los reportes acerca de la validación de fórmulas LTL se encuentran en el anexo K.

Realizando la evaluación en cada una de las capacidades se puede generar un análisis general de las capacidades teniendo en cuenta la cuantificación del cumplimiento de cada capacidad comprobada y estudiada por el factor de cumplimiento X_j y por el peso de cada capacidad Wc como se muestra en la casa de calidad; a partir de X_j y Wc se puede construir la tabla 15, mediante la cual se puede calcular un indicador general de cumplimiento de las capacidades empleadas para el caso de aplicación de diseño de modelos ortopédicos a la medida ósea.

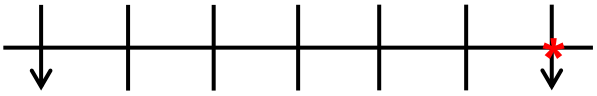
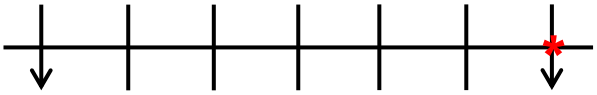
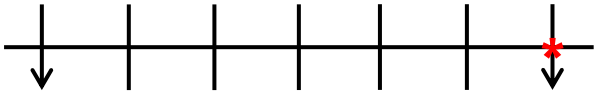
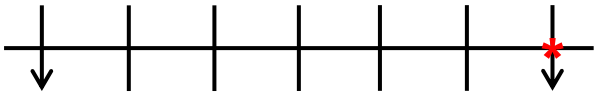
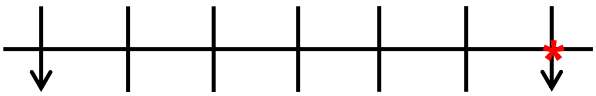
Tabla 15. Análisis e indicador de cumplimiento

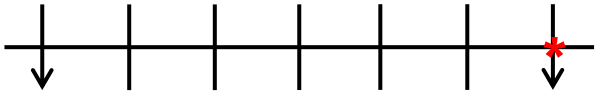
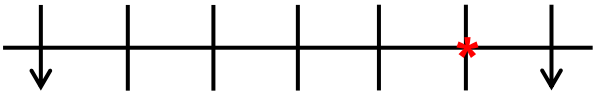
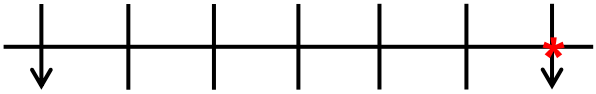
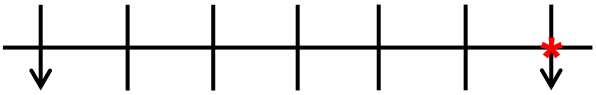
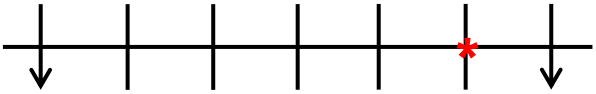
Capacidades	$C_{w,i}$	Factor de impacto							Medición
		Bajo	-2	-1	0	1	2	Alto	$\chi_{t,i} = \chi_i \times C_{w,i}$
Planear Diseño	$C_{w,1} = 18$							x	$\chi_{t,1} = 54$
Compartir Información	$C_{w,2} = 19$							x	$\chi_{t,2} = 57$
Adquirir Conocimiento	$C_{w,3} = 21$				x				$\chi_{t,3} = 0$
Transferir Conocimiento	$C_{w,4} = 10$				x				$\chi_{t,4} = 0$
Evaluar Productos	$C_{w,5} = 22$							x	$\chi_{t,5} = 66$
Apropiación tecnológica	$C_{w,6} = 12$							x	$\chi_{t,6} = 36$
Reconstrucción Ósea Digital	$C_{w,7} = 40$							x	$\chi_{t,7} = 120$
Modelado de Implantes	$C_{w,8} = 38$							x	$\chi_{t,8} = 114$
Simulación Estática Estructural	$C_{w,9} = 50$						x		$\chi_{t,9} = 100$
Simulación Quirúrgica	$C_{w,10} = 9$							x	$\chi_{t,10} = 27$
Interoperabilidad	$C_{w,11} = 74$							x	$\chi_{t,11} = 222$
Prototipado	$C_{w,12} = 48$						x		$\chi_{t,12} = 96$
Total importancia T_i:	$\sum_{i=1}^{n} C_{w,i} = 361$	Total medición T_c:							$\sum_{i=1}^{n} \chi_{t,i} = 892$
								$K_f = T_i \times 3$	1083

4.5. ANÁLISIS DE RESULTADOS

Según el desarrollo del marco para el cumplimento de capacidades en el sistema de diseño de dispositivos ortopédicos a la medida, se tienen los siguientes resultados, ver tabla 16:

Tabla 16. Resultados de aplicación

Capacidades	Cumplimiento
Planear Diseño	
Compartir Información	
Adquirir Conocimiento	
Transferir Conocimiento	
Evaluar Productos	
Apropiación tecnológica	
Reconstrucción Ósea Digital	

Modelado de Implantes	
Simulación Estática Estructural	
Simulación Quirúrgica	
Interoperabilidad	
Prototipado	

En la tabla 16, Las capacidades de adquirir conocimiento y de trasferencia de conocimiento, tienen un factor de cumplimiento igual a 0. Esto se debe a que no se encuentran actividades operacionales que satisfagan sus requerimientos, por esto, los resultados se interpretan como no concluyentes y se les asigna un valor de 0, que no significa que la capacidad no se esté cumpliendo, sino que no hay certidumbre de cumplimiento o incumplimiento.

Las capacidades de simulación estática estructural y de prototipado, presentan un factor de cumplimiento igual a 2. El no cumplimiento de algunos requerimientos para estas capacidades arroja este resultado. Y según con el modelo presentado en este trabajo, hay un nivel medio de certidumbre de cumplimiento para estas capacidades.

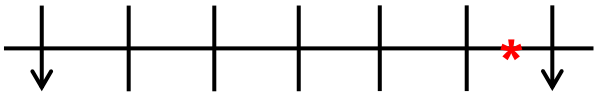
En el caso de las otras capacidades, los resultados arrojan un factor de cumplimiento alto, esto se debe a que todos los requerimientos que las componen son satisfechos.

El indicador de cumplimiento general de capacidades Tc = 892 no alcanzo el máximo valor que podía tener para este caso en kf=1083, ver figura 22. Por lo tanto, se puede decir que no

se cuentan con el total de todas las habilidades necesarias para cumplir las metas de la organización.

No se puede afirmar que el cumplimiento total de estas capacidades tenga una relación directa con el cumplimiento de las metas organizacionales; lo que se puede decir es que el cumplimiento de estas capacidades permite que se cumplan de mejor manera las metas deseadas.

Figura 22. Representación gráfica de indicador de cumplimiento.



5. CONCLUSIONES

El modelo de verificación enfocado en las capacidades del sistema, permite encontrar fallas relacionadas con las necesidades de los stakeholders con respecto al funcionamiento del sistema, debido a que no valida requerimientos concretos si no conjuntos de requerimientos asociados a habilidades particulares del sistema. Donde se puede determinar si el sistema es capaz o no de cumplir una función o habilidad que le provea valor directamente a las metas del sistema.

Al hacer la verificación de modelos del sistema utilizando su arquitectura y no modelos aislados de comportamiento o estructura se puede hacer una trazabilidad del impacto en el sistema, producto de la detección de fallas en su comportamiento y la repercusión que estos tienen sobre los demás componentes del sistema. Esto debido a que las arquitecturas relacionan los modelos de comportamiento con los objetivos misionales y también con los modelos de aplicaciones, datos e infraestructura.

Se abordó *model checking* de la forma más básica, sin embargo, existen alternativas más especializadas de *model checking* como la validación de modelos con información probabilística y con información de tiempo que permite extender los tipos de requerimientos utilizados en este trabajo y por ende la precisión con la que se determina si una capacidad se cumple o no.

Construir herramientas que permiten verificar el buen diseño de modelos arquitecturales permite reducir la posible inserción de fallas en etapas tempranas del diseño de sistemas. Teniendo en cuanta que las fallas que se encuentran están directamente relacionadas con las habilidades del sistema y no con requerimientos específicos, la reducción de costos producto de la detección de fallas no se da solo en la detección y reparación de este si no en la reducción de posibles consecuencias negativas producto del incumplimiento de una habilidad particular del sistema.

6. TRABAJO FUTURO

Según las observaciones obtenidas durante la realización de este proyecto, se realizan las siguientes recomendaciones para trabajos futuros:

Ya que algunos de los requerimientos no se pudieron verificar por la técnica de model checking básica, se recomienda hacer una evaluación de los modelos empleando procesos probabilísticos y de tiempo usados también en la técnica de *model checking*.

La verificación de los requerimientos en los modelos arquitecturales hechos por el modelo aplicado en este trabajo, puede usarse como una herramienta para la evaluación de arquitecturas de sistemas.

Aunque el marco arquitectural DoDAF es de los marcos más populares, en Colombia no cuenta con tanto reconocimiento, es por esto que se recomienda crear marcos de aplicación usando como marco arquitectural otros más conocidos a nivel nacional.

En la técnica de detección de fallas usada en este trabajo, uno de los procesos que toma bastante tiempo es el de pasar los modelos de comportamiento a modelos formales, es por esto que sería conveniente profundizar en la automatización de este proceso.

Cada vista de la arquitectura tiene asociada un conjunto de capacidades para cumplir con las necesidades del sistema, es preciso evaluar en mayor detalle las capacidades relacionadas con otras vistas arquitecturales.

7. REFERENCIAS BIBLIOGRÁFICAS

[1] Zachman, J. A.; "A framework for information systems architecture", in IBM Systems Journal; 1999; 38, 2/3; ABI/INFORM Global pg. 454, 1987.

[2] J.-P. K. Christel Baier, Principles of model checking. 2008.

[3] M. P. Vitor Lima, Chamseddine Talhi, Djedjiga Mouheb, Mourad Debbaibi, Lingyu Wang,"Formal verification and validation of UML 2.0 secuence diagrams using source and destination of messages.," Electron. Notes Theor. Comput. Sci., vol. 254, pp. 143–160, 2009.

[4] P. Baldan, A. Corradini, and F. Gadducci, "Specifying and verifying UML activity diagrams via graph transformation," in Global Computing, 2004, pp. 18–33.

[5] L. Apvrille and P. de Saqui-Sannes, "Static analysis techniques to verify mutual exclusion situations within SysML models," in SDL 2013: Model-Driven Dependability Engineering, Springer, 2013, pp. 91–106.

[6] M. E. Beato, M. Barrio-Solórzano, C. E. Cuesta, and P. de la Fuente, "UML automatic verification tool with formal methods," Electron. Notes Theor. Comput. Sci., vol. 127, no. 4, pp. 3–16, 2005.

[7] "Systems and software engineering -- Vocabulary," ISO/IEC/IEEE 24765:2010(E), pp. 1–418, 2010.

[8] T. Blevins, "The Open Group Architecture Framework (TOGAF TM 9) and the US Department of Defense," vol. 0, no. July, 2010.

[9] K. Griendling, F. Drive, and D. N. Mavris, "Development of a Dodaf-Based Executable Architecting Approach to Analyze System-of-Systems Alternatives," 2011.

[10] G. L. C. Castro Ginna P, BRAVO I. EDNA, "EVALUACIÓN DE UN MODELO DE INTEGRACIÓN DE HERRAMIENTAS SOFTWARE DIRIGIDO AL SECTOR BIOMÉDICO-ORTOPÉDICO," Universidad Industrial de Santander,

2013.

[11] TOGAF TM Version 9 A Pocket Guide.

[12] F. Dandashi, R. Siegers, J. Jones, and T. Blevins, “The Open Group Architecture Framework (TOGAF) and the US Department of Defense Architecture Framework (DoDAF)(2007).”.

[13] Delgado Dario J, Flórez Jerson, Noguera Jormary, Lopez Clara L. R. (2016). *Architectural capability analysis using a model-checking technique.* http://doi.org/10.13140/RG.2.2.24883.22566

[14] Emad Ebeid, Davide Quaglia, and Franco Fummi. Generation of systemc/tlm code from uml/marte sequence diagrams for verification. In Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2012 IEEE 15th International Symposium on, pages 187–190. IEEE, 2012.

[15] Vitor Lima, Chamseddine Talhi, Djedjiga Mouheb, Mourad Debbabi, Lingyu Wang, and Makan Pourzandi. Formal verification and validation of uml 2.0 sequence diagrams using source and destination of messages. Electronic Notes in Theoretical Computer Science, 254:143–160, 2009.

[16] E. M. Clarke and E. A. Emerson, “Synthesis of Synchronization Skeletons for Branching Time Temporal Logic. Logic of Programs: Workshop, Yorktown Heights, NY, May 1981, volume 131 of Lecture Notes in Computer Science,” *Lect. Notes Comput. Sci.*, vol. 131, 1981.

[17] Shahrah, A.Y.; Hossain, M.A.; Alghamdi, A.S., "A collaboration architecture for distributed smart surveillance systems based on DoDAF 2.0," in Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on, vol., no., pp.305-310, May 30 2012-June 1 2012

[18] Biswas, A.; Hayden, J.; Phillips, M.S.; Bhasin, K.B.; Putt, C.; Sartwell, T., "Applying DoDAF to NASA Orion Mission Communication and Navigation Architecture," in Aerospace Conference, 2008 IEEE, vol., no., pp.1-9, 1-8 March 2008.

[19] Christel Baier, Joost-Pieter Katoen, et al. Principles of model checking, volume 26202649. MIT press Cambridge, 2008.

[20] "Systems and software engineering -- Vocabulary," *ISO/IEC/IEEE 24765:2010(E)*, pp. 1–418, 2010.

[21] T. Blevins, "The Open Group Architecture Framework (TOGAF ™ 9) and the US Department of Defense," vol. 0, no. July, 2010.

[22] K. Griendling, F. Drive, and D. N. Mavris, "Development of a Dodaf-Based Executable Architecting Approach to Analyze System-of-Systems Alternatives," 2011.

[23] J.-P. K. Christel Baier, Principles of *model checking*. 2008.

[24] Cruz-Lemus, J. A., Genero, M., Manso, M. E., & Piattini, M. (2005). Evaluating the effect of composite states on the understandability of UML statechart diagrams. In *Model Driven Engineering Languages and Systems* (pp. 113-125). Springer Berlin Heidelberg.

[25] M. Y. Vardi and P. Wolper, "An automata-theoretic approach to automatic program verification," in *Proceedings of the First Symposium on Logic in Computer Science*, 1986, pp. 322–331.

[26] D. Giannakopoulou, "Model checking for concurrent software architectures," Imperial College, 1999.

[27] K. A. Griendling, "Architect: the architecture-based technology evaluation and capability tradeoff method," Georgia Institute of Technology, 2011.

[28] GINNA PAOLA CASTRO CASTAÑO, evaluación de un modelo de integración de herramientas software dirigido al sector biomédico-ortopédico.

[29] Forsberg, K., & Mooz, H. (2001). Visual explanation of development methods and strategies including the Waterfall, Spiral, Vee, Vee+, and Vee++ models. In INCOSE 2001 International Symposium. inproceedings.

[30] Bachmann, F., Bass, L., Chastek, G., Donohoe, P., & Peruzzi, F. (2000). The architecture based design method (techreport).

[31] Department Of Defense, "DoD Architecture Framework Version 2.0 - Architectural Data and Models Architect's Guide", 2009

8. BIBLIOGRAFÍA

Apvrille, L. and De Saqui-Sannes, P. "Static analysis techniques to verify mutual exclusion situations within SysML models," in SDL 2013: Model-Driven Dependability Engineering, Springer, 2013, pp. 91–106.

Bachmann, F., Bass, L., Chastek, G., Donohoe, P., & Peruzzi, F. (2000). The architecture based design method (techreport).

Baier, C., Katoen, J. P., & Larsen, K. G. (2008). Principles of model checking. Cambridge, USA: MIT press.

Baldan, P., Corradini, A. and Gadducci, F. "Specifying and verifying UML activity diagrams via graph transformation," in Global Computing, 2004, pp. 18–33.

Beato, M. E., Barrio-Solórzano, M., Cuesta, C. E. and De la Fuente, P. "UML automatic verification tool with formal methods," Electron. Notes Theor. Comput. Sci., vol. 127, no. 4, pp. 3–16, 2005.

Biswas, A.; Hayden, J.; Phillips, M.S.; Bhasin, K.B.; Putt, C.; Sartwell, T., "Applying DoDAF to NASA Orion Mission Communication and Navigation Architecture," in Aerospace Conference, 2008 IEEE, vol., no., pp.1-9, 1-8 March 2008.

Blevins, T., Dandashi, F., & Tolbert, M. (2010). The Open Group Architecture Framework (TOGAF 9) and the US Department of Defense Architecture Framework 2.0 (DoDAF 2.0). misc, The Open Group 4.

CASTRO CASTAÑO, Ginna Paola. Evaluación de un modelo de integración de herramientas software dirigido al sector biomédico-ortopédico. Bucaramanga, 2014. Trabajo de Grado (Diseñadora Industrial). Universidad Industrial de Santander. Facultad de Ingenierías Fisicomecánicas. Escuela de Diseño Industrial.

CASTRO, Ginna P, BRAVO I. EDNA, G. L. C. (2013). Evaluación de un modelo de integración de herramientas software dirigido al sector biomédico-ortopédico. Universidad Industrial de Santander - Bucaramanga Colombia.

Clarke, E. M., & Emerson, E. A. (1981). Synthesis of Synchronization Skeletons for Branching Time Temporal Logic. Logic of Programs: Workshop, Yorktown Heights, NY, May 1981, volume 131 of Lecture Notes in Computer Science. In Lecture Notes in Computer Science, Springer Berlin Heidelberg (Vol. 131). Article.

Cruz-Lemus, J. A., Genero, M., Manso, M. E., & Piattini, M. (2005). Evaluating the effect of composite states on the understandability of UML statechart diagrams. In Model Driven Engineering Languages and Systems (pp. 113-125). Springer Berlin Heidelberg.

Dandashi, F., Siegers, R., Jones, J. and Blevins, T. "The Open Group Architecture Framework (TOGAF) and the US Department of Defense Architecture Framework (DoDAF)(2007)."

Delgado Dario J, Flórez Jerson, Noguera Jormary, Lopez Clara L. R. (2016). *Architectural capability analysis using a model-checking technique.* http://doi.org/10.13140/RG.2.2.24883.22566

Ebeid, E., Quaglia, D., and Fummi, F. Generation of systemc/tlm code from uml/marte sequence diagrams for verification. In Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2012 IEEE 15th International Symposium on, pages 187–190. IEEE, 2012.

Forsberg, K., & Mooz, H. (2001). Visual explanation of development methods and strategies including the Waterfall, Spiral, Vee, Vee+, and Vee++ models. In INCOSE 2001 International Symposium. inproceedings.

Giannakopoulou, D. "Model Checking for Concurrent Software Architectures", PhD Thesis, Imperial College of Science Technology and Medicine, University of London, March 1999.

Griendling, K. A. (2011). Architect: the architecture-based technology evaluation and capability tradeoff method. Doctoral thesis: Georgia Institute of Technology.

Griendling, K., & Mavris, D. N. (2011). Development of a dodaf-based executable architecting approach to analyze system-of-systems alternatives. In IEEE Aerospace Conference Proceedings. http://doi.org/10.1109/AERO.2011.5747654

ISO/IEC/IEEE. (2011). INTERNATIONAL STANDARD ISO/IEC/IEEE 24465, System and software engineering-vocabulary. Standard, Switzerland: ISO.

Lima, V., Talhi, C., Mouheb, D., Debbabi, M., Wang, L. and Pourzandi, M. Formal verification and validation of uml 2.0 sequence diagrams using source and destination of messages. Electronic Notes in Theoretical Computer Science, 254:143–160, 2009.

Shahrah, A.Y.; Hossain, M.A.; Alghamdi, A.S., "A collaboration architecture for distributed smart surveillance systems based on DoDAF 2.0," in Computer Science and Software Engineering (JCSSE), 2012 International Joint Conference on, vol., no., pp.305-310, May 30 2012-June 1 2012

ToG. (2009). 9, the open group architecture framework (togaf). The Open Group, 1. Article.

Vardi, M. Y. and Wolper, P. "An automata-theoretic approach to automatic program verification," in Proceedings of the First Symposium on Logic in Computer Science, 1986, pp. 322–331.

Zachman, J. a. (1987). A framework for information systems architecture. IBM Systems Journal, 26(3), 276–292. http://doi.org/10.1147/sj.263.0276

9. ANEXOS

ANEXO A. AV-1: Descripción e Información Resumida – Arquitectura de un sistema de diseño de modelos ortopédicos a la medida (SMOM).

AV-1. SISTEMA DE DISEÑO DE MODELOS ORTOPÉDICOS A LA MEDIDA.

El primer producto sugerido por DoDAF 2.0 es la vista AV-1, el cual, es una descripción general del problema. El AV-1 sirve para mostrar una visión general y un resumen de la información que define el problema, el alcance y el propósito.

1. Identificación.

Nombre: ARQUITECTURA DE UN SISTEMA PARA EL DISEÑO DE MODELOS ORTOPÉDICOS A LA MEDIDA (SMOM).

Versión: V1.0

2. Antecedentes.

En la actualidad se están desarrollando diseños de modelos ortopédicos a la medida (SMOM), este tipo de sistema tiene como finalidad el proceso para el desarrollo de nuevos productos en el sector ortopédico específicamente implantes ajustados a la geometría ósea para pacientes que presenta casos pocos particulares. El proceso de SMOM implica la detección de requerimientos, análisis de diseño, diseño de modelo, verificación del modelo y ajuste a la geometría ósea, y envío a desarrollo de manufactura. Este tipo de sistemas son desarrollados actualmente en la escuela de Diseño Industrial de la Universidad Industrial de Santander donde se han obtenido avances como procesos de diseños de modelos ortopédicos para traumas craneoencefálicos.

3. Metas del Sistema

- Integrar cada uno de los softwares del proceso del sistema.
- Recolectar datos en tiempo real.
- Finalizar el proceso de diseño en tiempo real
- Mejorar el rendimiento de diseño de modelos
- Crear metodología para mejorar curva de aprendizaje.
- Creación de guías quirúrgicas.

4. Requerimientos Generales del Sistema

- En el proceso BIOCAD, la tecnología debe permitir la reconstrucción de las tomografías del trauma musculo esquelético usando técnica de ingeniería inversa.
- En el proceso CAD, se debe diseñar el dispositivo ortopédico a la medida de los modelos propuesto por los especialistas para cada caso de estudio.
- El prototipo ortopédico diseñado se le deben realizar análisis biomecánicos virtual para así detectar falencias.
- Los diagnósticos, para cada uno de los casos se deben definir por los especialistas.
- Se deben realizar los procesos de simulación para el abordaje quirúrgico.
- El sistema debe ofrecer un diseño de prótesis ajustado a las necesidades de cada paciente.
- El sistema debe proveer modelos de los traumas musculo esquelético en prototipos 3d.
- El sistema debe proporcionar los prototipos necesarios para realizar análisis y pruebas de verificación en cada una de las fases requeridas.

5. Preguntas Críticas.

- ¿Qué recursos son necesarios para la creación y realización de guías quirúrgicas?
- ¿Qué funciones y operaciones son necesarios para el proceso del sistema SMOM?
- ¿Cuál es la confiabilidad del sistema existen redundancias durante la ejecución del proceso de diseño?
- ¿El sistema cumple con los estándares permitidos para la realización de los modelos a diseñar?
- ¿El análisis de los modelos de la arquitectura del sistema afectara el proceso para el diseño de los modelos ortopédicos?

6. Alcance.

La arquitectura describirá los requerimientos del sistema donde se va a contemplar los estándares dados por el marco arquitectural y así facilitar la toma de decisiones para futuros cambios que se pueda presentar en el sistema. A partir del uso de MA DoDAF 2.0 se obtendrán descripciones del sistema desde diversos puntos de vista como: vista del sistema, vista de capacidades y vista de operaciones. Por lo siguiente, durante la elaboración de la arquitectura se tendrán en cuenta el estado actual del sistema, los escenarios, la visión, la misión y las metas trazadas por la entidad, con los resultados obtenidos se generarán indicaciones para el mejoramiento del sistema estudiado.

ANEXO B. CV-1: Visión del sistema de diseño de dispositivos ortopédicos

Visión

La visión del Sistema de Diseño de Dispositivos Ortopédicos a la medida es llegar a ser un laboratorio de diseño rápido y desarrollo de productos ortopédicos ajustados a las necesidades de cada paciente, apoyando el proceso de recuperación de lesiones óseas para que sea mejor y más rápido.

Metas del Sistema

- Integrar cada uno de los softwares del proceso del sistema.
- Recolectar datos en tiempo real.
- Finalizar el proceso de diseño en tiempo real
- Mejorar el rendimiento de diseño de modelos
- Crear metodología para mejorar curva de aprendizaje.
- Creación de guías quirúrgicas.

Efectos deseados

- Obtener un mejor diagnóstico de la fractura
- El diseño del implante debe ser quedar ajustado a la estructura ósea del paciente
- El diseño debe soportar las cargas de esfuerzos necesitadas por el paciente
- El desarrollo del implante debe ser factible
- El diseño del implante debe ser validado por un ortopedista y cirujano
- El diseño pueda ser utilizado para simular un abordaje quirúrgico
- Socialización entre el grupo de diseño de los conocimientos adquiridos para cada proyecto.

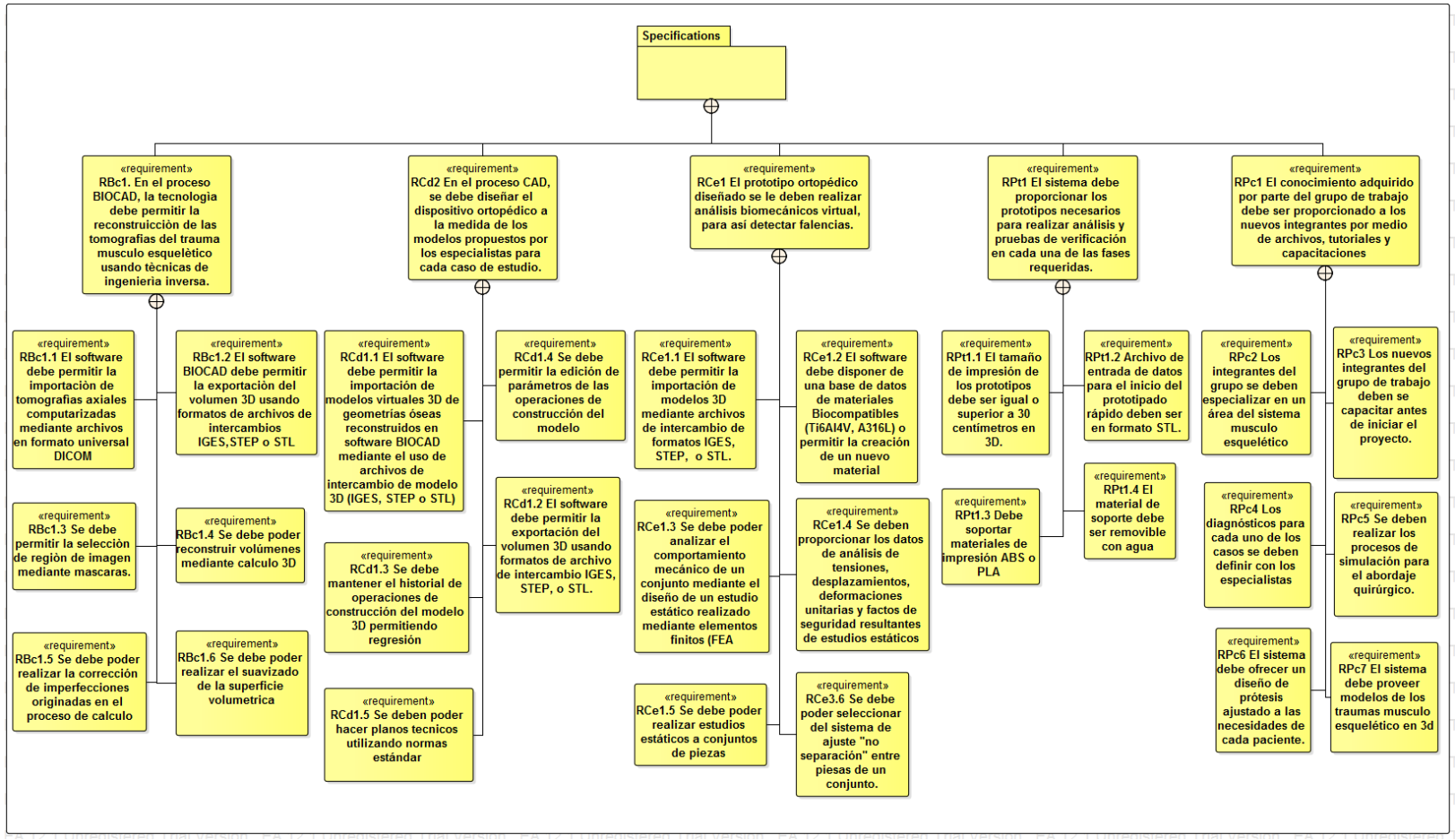
Capacidades de alto nivel

- Planear diseño
- Modelado de implantes
- Evaluar productos
- Apropiación tecnológica
- Adquirir conocimiento
- Reconstrucción ósea digital
- Prototipado

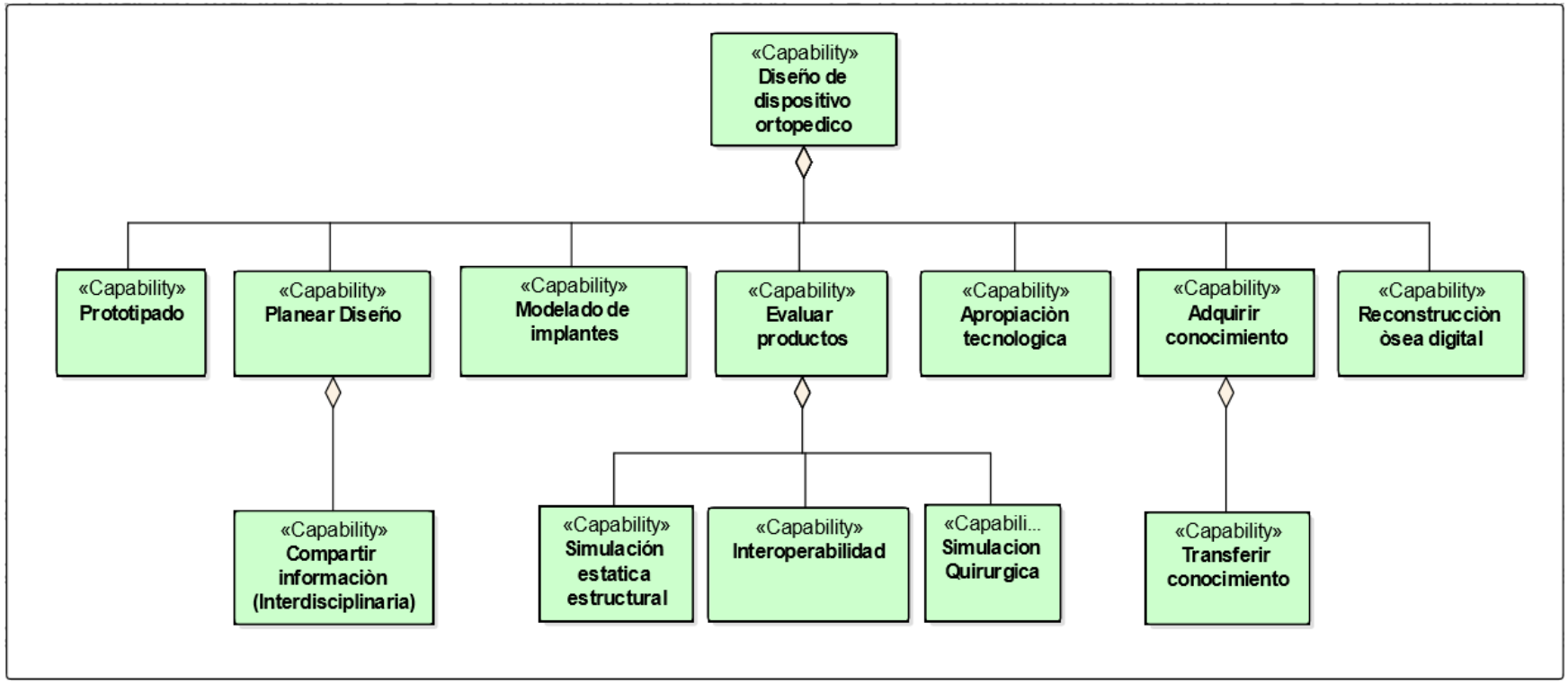
Conexión de capacidades de alto nivel con efectos deseados.

CAPACIDADES DE ALTO NIVEL / EFECTOS DESEADOS	Planear diseño	Modelado de implantes	Evaluar productos	Apropiación tecnológica	Adquirir conocimiento	Reconstrucción ósea digital	Prototipado
Obtener un mejor diagnóstico de la fractura	x					X	X
El diseño del implante debe ser quedar ajustado a la estructura ósea del paciente	X	X				X	X
El diseño debe soportar las cargas de esfuerzos necesitadas por el paciente			X				
Ser factible el desarrollo del implante	X						
El diseño del implante debe ser validado por un ortopedista y cirujano	X		X				X
El diseño pueda ser utilizado para simular un abordaje quirúrgico			X				
Socialización entre el grupo de diseño de los conocimientos adquiridos para cada proyecto.	X				X		

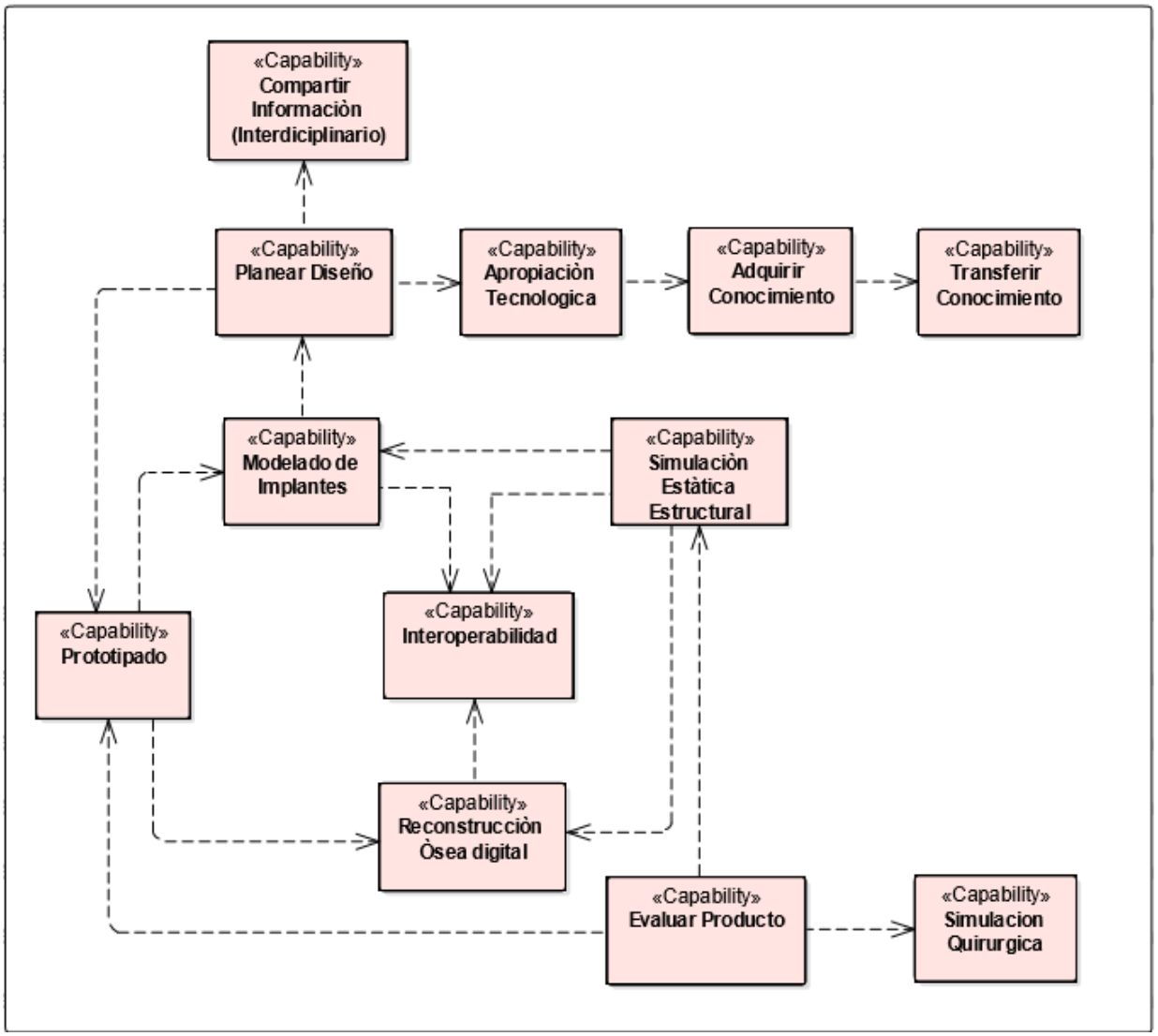
ANEXO C. Diagrama de requerimientos



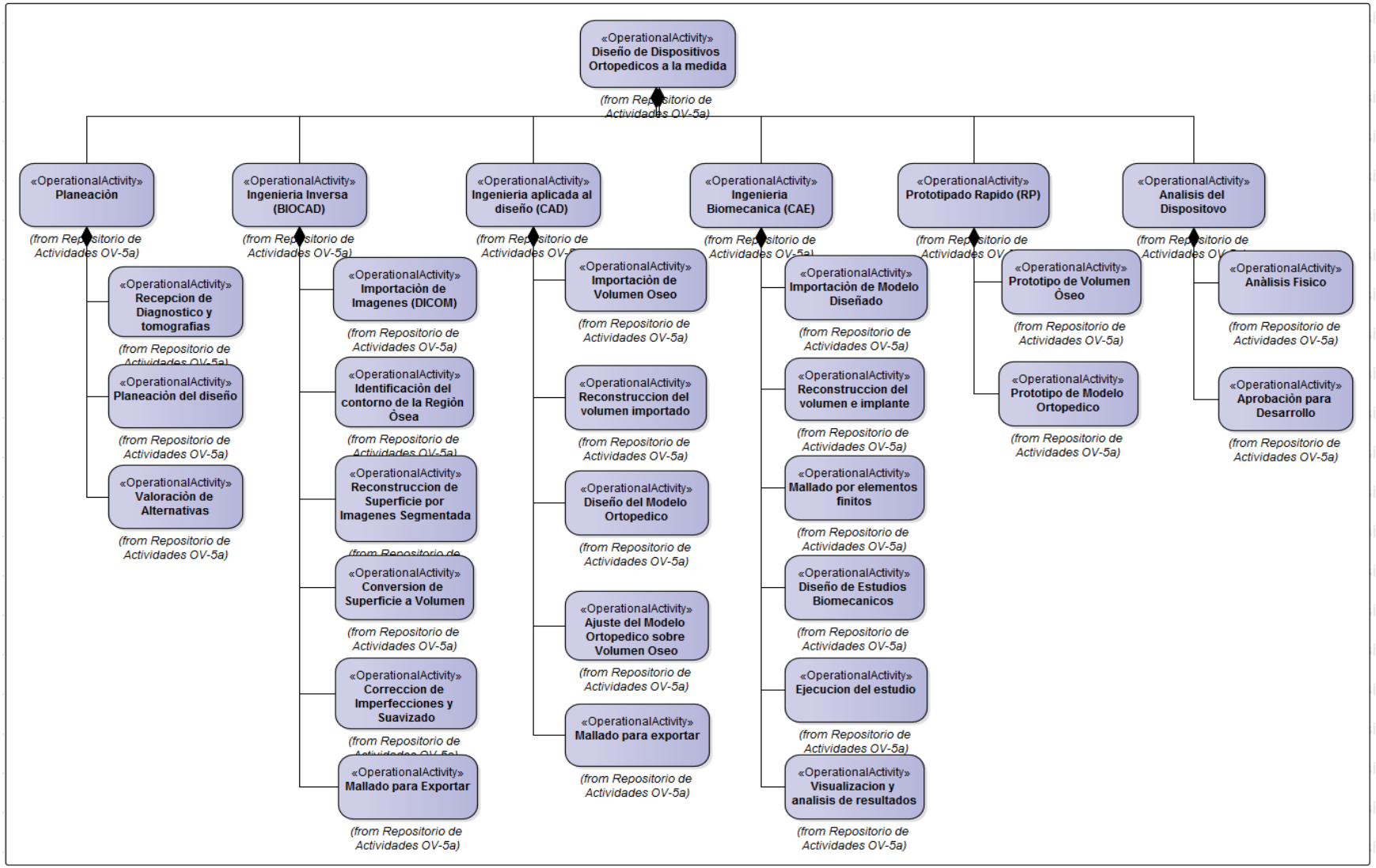
ANEXO D. CV-2: Taxonomía de capacidades



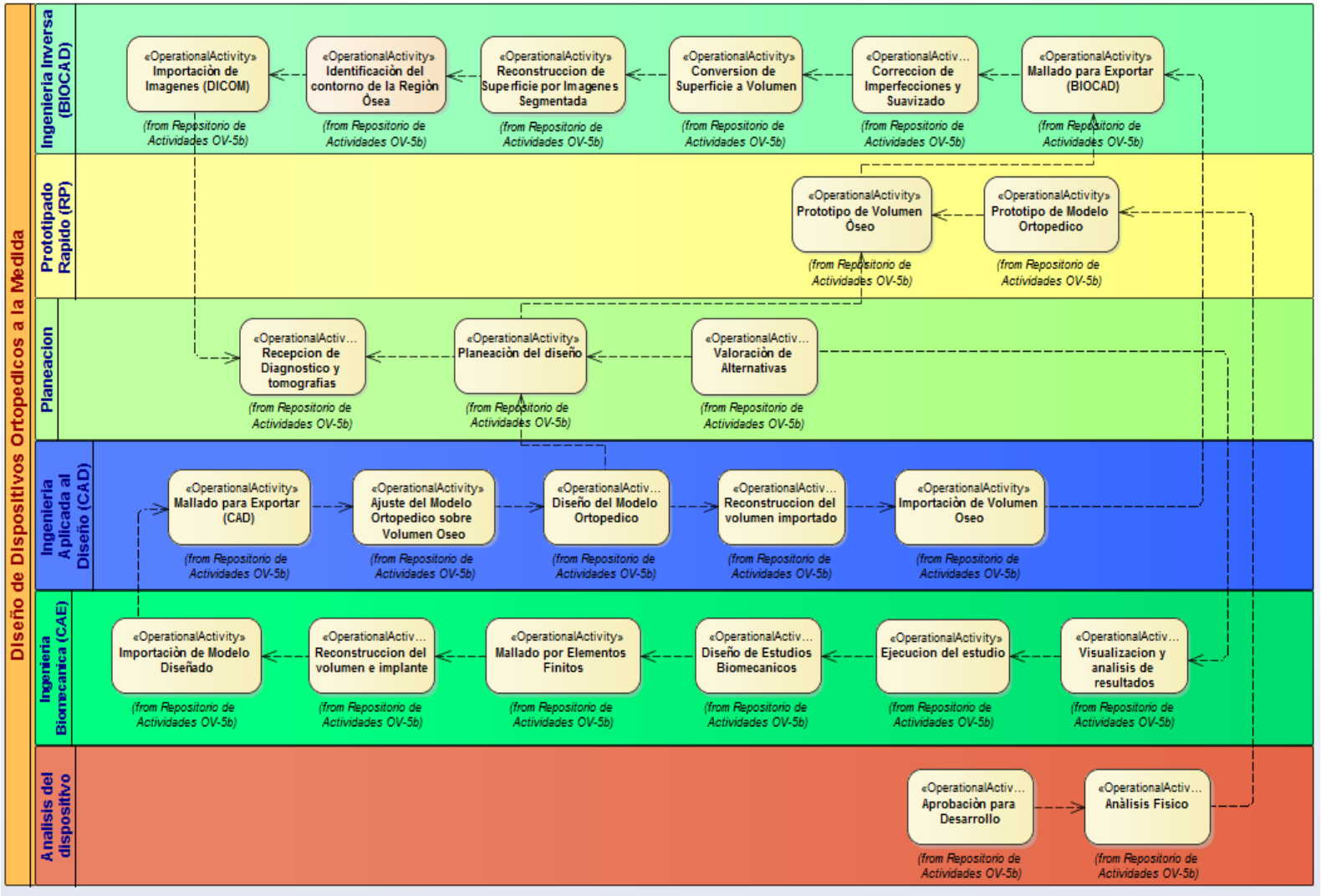
ANEXO E: CV-4: Dependencia de capacidades



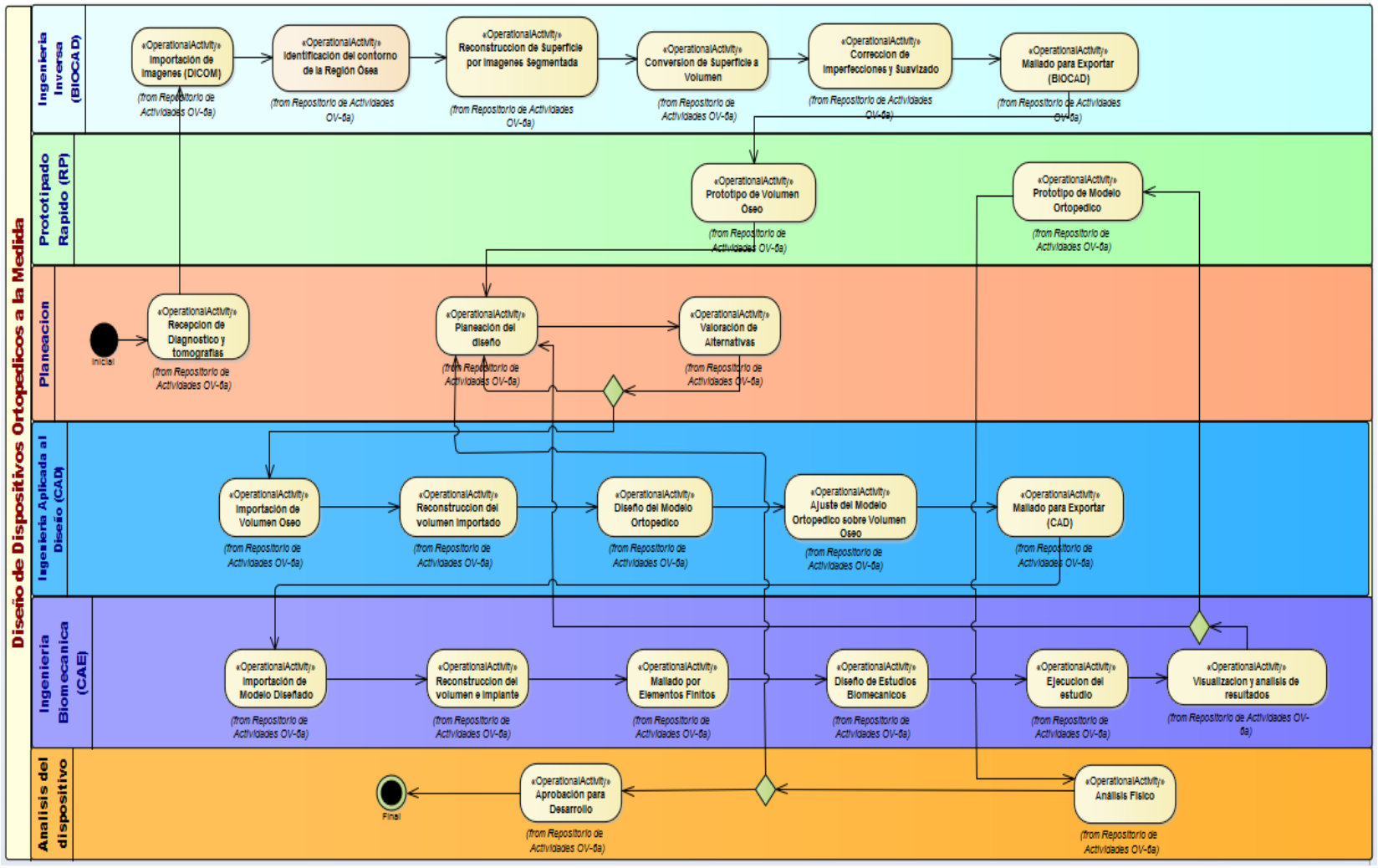
ANEXO F: OV-5A: Jerarquía de actividades operacionales



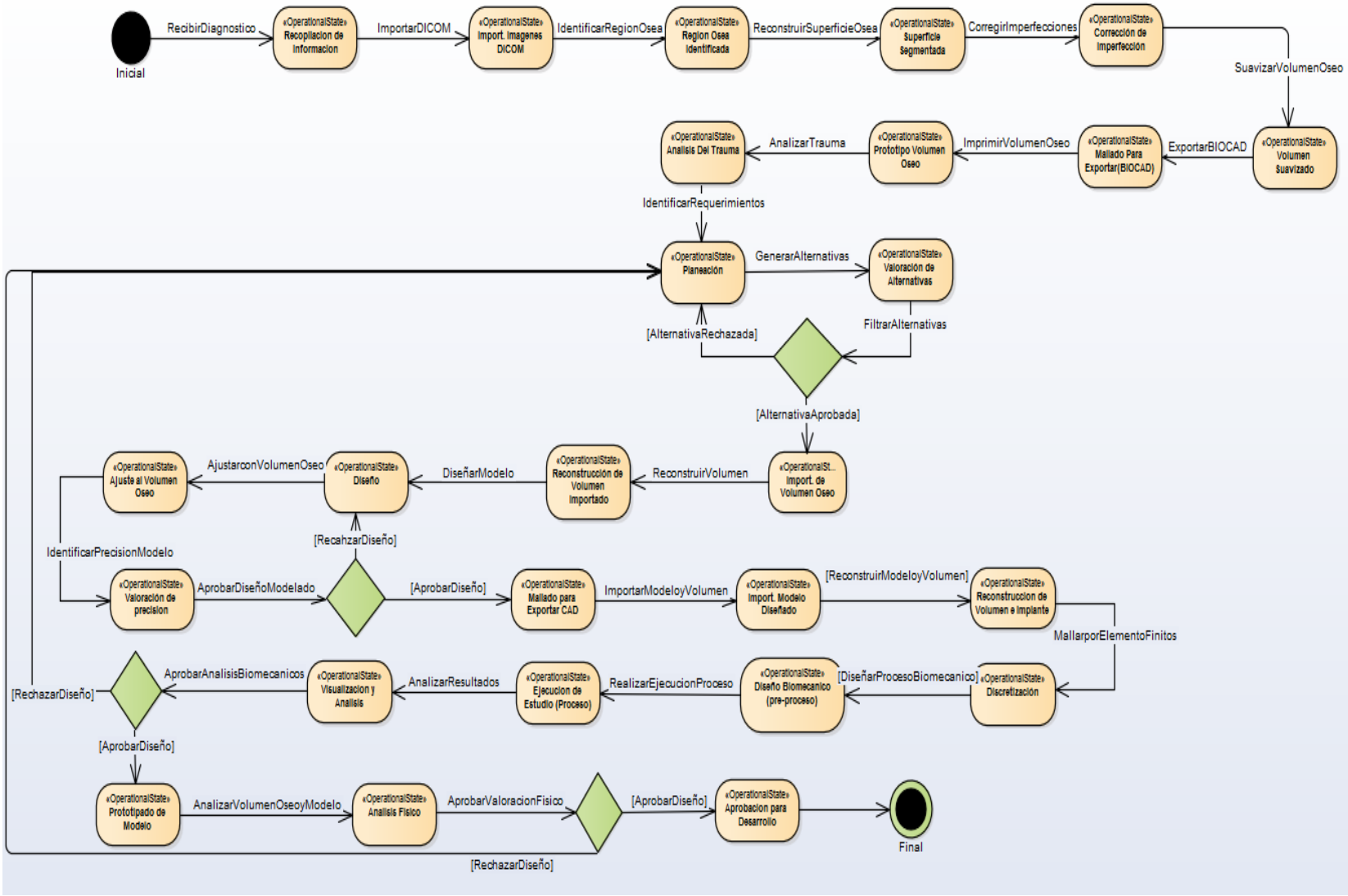
ANEXO G: OV-5B: Modelo de actividades operacionales



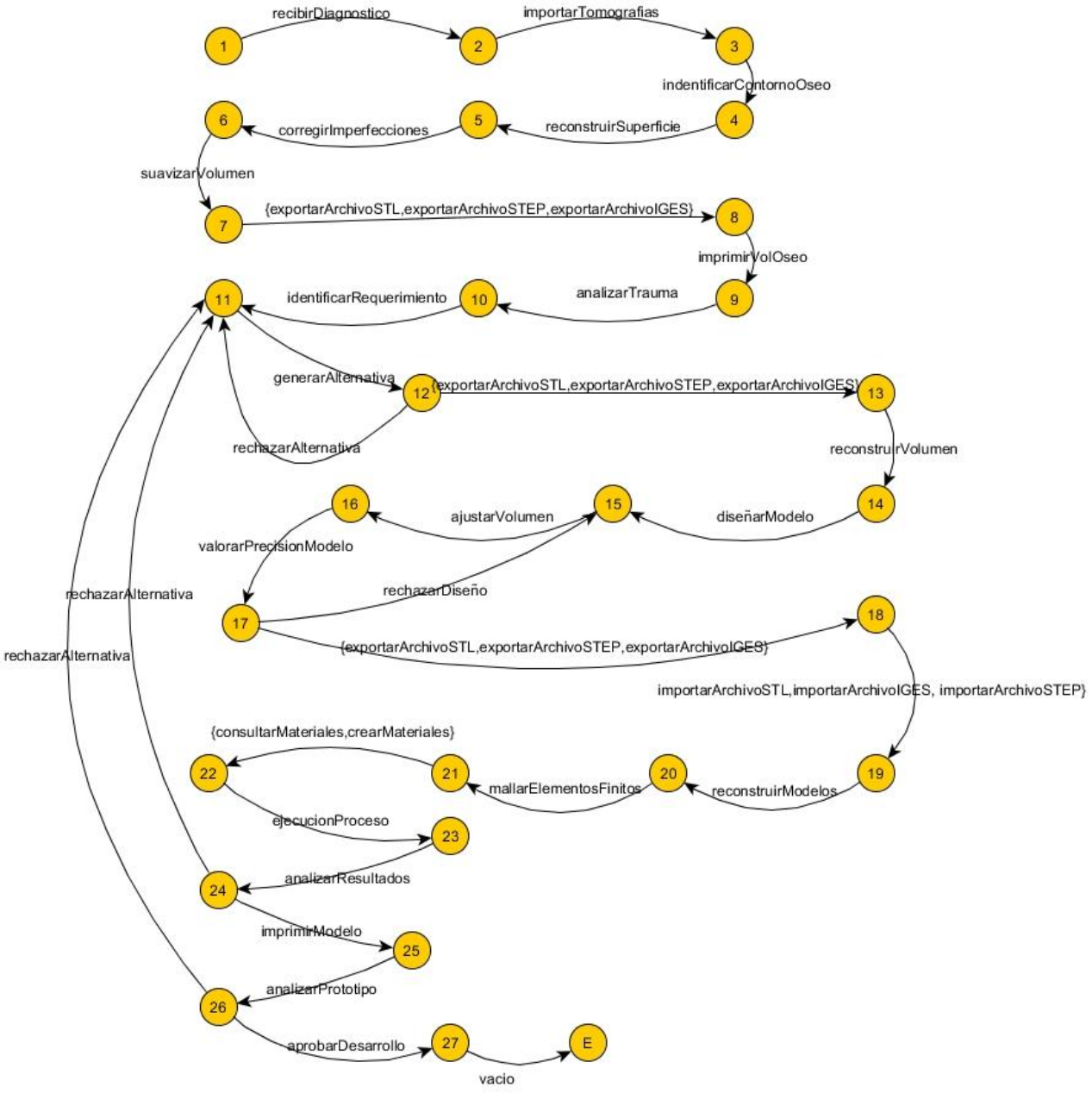
ANEXO H: OV-6A: Modelo de reglas operacionales



ANEXO I: OV-6B: Descripción de transición de estados



ANEXO J: Diagrama de estados del sistema de diseño de dispositivos ortopédicos a la medida en bünci autómata



ANEXO K. Informes de pruebas de las capacidades

CAPACIDAD	SUBCONJUNTO	PROPIEDAD (φ)	SALIDA
C1	$O_{s,1}$	Rpc41 = <>((recibirDiagnostico) W (imprimirVolOseo)) Rpc42 = <>((Rpc41)->(analizartrauma)) Rpc43 = <>((Rpc42)->(identificarRequerimiento)) Rpc44 = <>((Rpc43)->(generarAlternativa))	☑
		Rpc61 = <> ((diseñarModelo)->(ajustarVolumen)) Rpc62 = <> ((Rpc61)->(valorarPrecisionModelo)) Rpc63 = <> ((Rpc62)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) || (rechazarDiseño)))	☑
		Rpc7 = <> ((suavizarVolumen)->(imprimirVolOseo))	☑
	$O_{a,1}$		
	$O_{l,1}$	Rpt1 = <>((imprimirModelo) W (imprimirVolOseo))	☑
C2	$O_{s,2}$	RPc4 = <>((recibirDiagnostico) W (imprimirVolOseo)) assert Rpc42 = <>((Rpc41)->(analizartrauma)) assert Rcp43 = <>((Rpc42)->(identificarRequerimiento)) assert Rcp44 = <>((Rcp43)->(generarAlternativa))	☑

		Rcp6 = assert Rcp61 = <> ((diseñarModelo)->(ajustarVolumen)) assert Rcp62 = <> ((Rcp61)->(valorarPrecisionModelo)) assert Rcp63 = <> ((Rcp62)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) || (rechazarDiseño)))	
	$O_{a,2}$		
	$O_{l,2}$	RPc5 = assert Rcp51 = <>((imprimirVolOseo) W (imprimirModelo)) assert Rcp52 = <>((Rcp51)->(analizarPrototipo)) assert Rcp53 = <>((Rcp52)->((aprobarDesarrollo) || (rechazarPrototipo)))	
C3	$O_{s,3}$	RPc2**	Indefinido
		RPc3*	
	$O_{a,3}$	RPc1**	Indefinido
	$O_{l,3}$		
C4	$O_{s,4}$	RPc1**	Indefinido
	$O_{a,4}$		
	$O_{l,4}$	RPc2**	Indefinido
C5	$O_{s,5}$	RCe1*	

		RPc6 = assert Rcp61 = <> ((diseñarModelo)->(ajustarVolumen)) assert Rcp62 = <> ((Rcp61)->(valorarPrecisionModelo)) assert Rcp63 = <> ((Rcp62)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) || (rechazarDiseño)))	
	$O_{a,5}$	RPc5 = assert Rcp51 = <>((imprimirVolOseo) W (imprimirModelo)) assert Rcp52 = <>((Rcp51)->(analizarPrototipo)) assert Rcp53 = <>((Rcp52)->((aprobarDesarrollo) || (rechazarPrototipo)))	
	$O_{l,5}$	RPc4 = <>((recibirDiagnostico) W (imprimirVolOseo)) assert Rpc42 = <>((Rpc41)->(analizartrauma)) assert Rcp43 = <>((Rpc42)->(identificarRequerimiento)) assert Rcp44 = <>((Rcp43)->(generarAlternativa))	
C6	$O_{s,6}$		
	$O_{a,6}$	RBc1*	
		RCd1*	
		RCe1*	
		RPt1*	
	$O_{l,6}$		
C7	$O_{s,7}$	RBc1*	
		RBc1.3 = <> (importarTomografia -> identificarContorneoOseo)	
		RBc1.4 = <> (identificarContorneoOseo ->	

		reconstruirSuperficie)	
		RBc1.5 = <> (reconstruccionSuperficie -> corregirImperfecciones)	
		RBc1.6 = <> (corregirImperfecciones-> SuavizarVolumen)	
	$O_{a,7}$	RBc1.1 = <> importarTomografia	
	$O_{l,7}$	RBc1.2 = <> (SuavizarVolumen -> exportarArchivoSTL,exportarArchivoSTEP,exportarArchivoIGES)	
C8	$O_{s,8}$	RCd1*	
		RCd1.3 = <>((importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP) -> (reconstruirVolumen))	
		RCd1.4 = (<>(reconstruirVolumen -> ((diseñarModelo W rechazarDiseño)))-> diseñarModelo)	
		RCd1.5 = <>((reconstruirVolumen) -> (diseñarModelo))	
	$O_{a,8}$		
	$O_{l,8}$	RCd1.1= <>((generarAlternativa)-> (importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP))	
		RCd1.2= <>((valorarPrecisionModelo)-> ((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) W (importarArchivoSTL || importarArchivoIGES ||	

		importarArchivoSTEP)))	
C9	$O_{s,9}$	RCe1*	☐
		RCe1.3 = <>((consultarMateriales || crearMateriales) -> (ejecucionProceso))	☐
		RCe1.4 = <>((ejecucionProceso) -> (analizarResultados))	☐
		RCe1.5**	Indefinido
		RCe1.6**	Indefinido
		RPc5 = assert Rcp51 = <>((imprimirVolOseo) W (imprimirModelo)) assert Rcp52 = <>((Rcp51)->(analizarPrototipo)) assert Rcp53 = <>((Rcp52)->((aprobarDesarrollo) || (rechazarPrototipo)))	☐
	$O_{a,9}$	RCe1.2 = <>((mallarElementoFinitos) -> (consultarMateriales || crearMateriales))	☐
	$O_{l,9}$	RCe1.1 = <>((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) -> (importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP))	☐
		RPc4 = <>((recibirDiagnostico) W (imprimirVolOseo)) assert Rpc42 = <>((Rpc41)->(analizartrauma)) assert Rcp43 = <>((Rpc42)->(identificarRequerimiento)) assert Rcp44 = <>((Rcp43)-	☐

		>(generarAlternativa))	
C10	$O_{s,10}$	RPc5 = assert Rcp51 = <>((imprimirVolOseo) W (imprimirModelo)) assert Rcp52 = <>((Rcp51)->(analizarPrototipo)) assert Rcp53 = <>((Rcp52)->((aprobarDesarrollo) || (rechazarPrototipo)))	
	$O_{a,10}$		
	$O_{l,10}$		
C11	$O_{s,11}$	RBc1.1 = <> importTomography	
		RBc1.2 = <>((smooth)->(exportSTL || exportSTEP || exportGES))	
		RCd1.1= <>((generarAlternativa)->(importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP))	
		RCd1.2= <>((valorarPrecisionModelo)->((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) W (importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP)))	
		RCd1.3 = <>((importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP) -> (reconstruirVolumen))	
		RCe1.1 = <>((exportarArchivoSTL || exportarArchivoSTEP || exportarArchivoIGES) ->	

		(importarArchivoSTL || importarArchivoIGES || importarArchivoSTEP))	
		RCe1.2 = <> importTomography	
		RPt1.2 = <>(importarArchivoSTL)	
	$O_{a,11}$		
	$O_{i,11}$	RBc1*	
		RCd1*	
		RCe1*	
C12	$O_{s,12}$	RPt1 = <>((imprimirModelo)W(imprimirVolOseo))	
		RPt1.1**	Indefinido
		RPt1.3**	Indefinido
		RPt1.4**	Indefinido
	$O_{a,12}$	RPt1.2 = <>(importarArchivoSTL)	
	$O_{i,12}$		