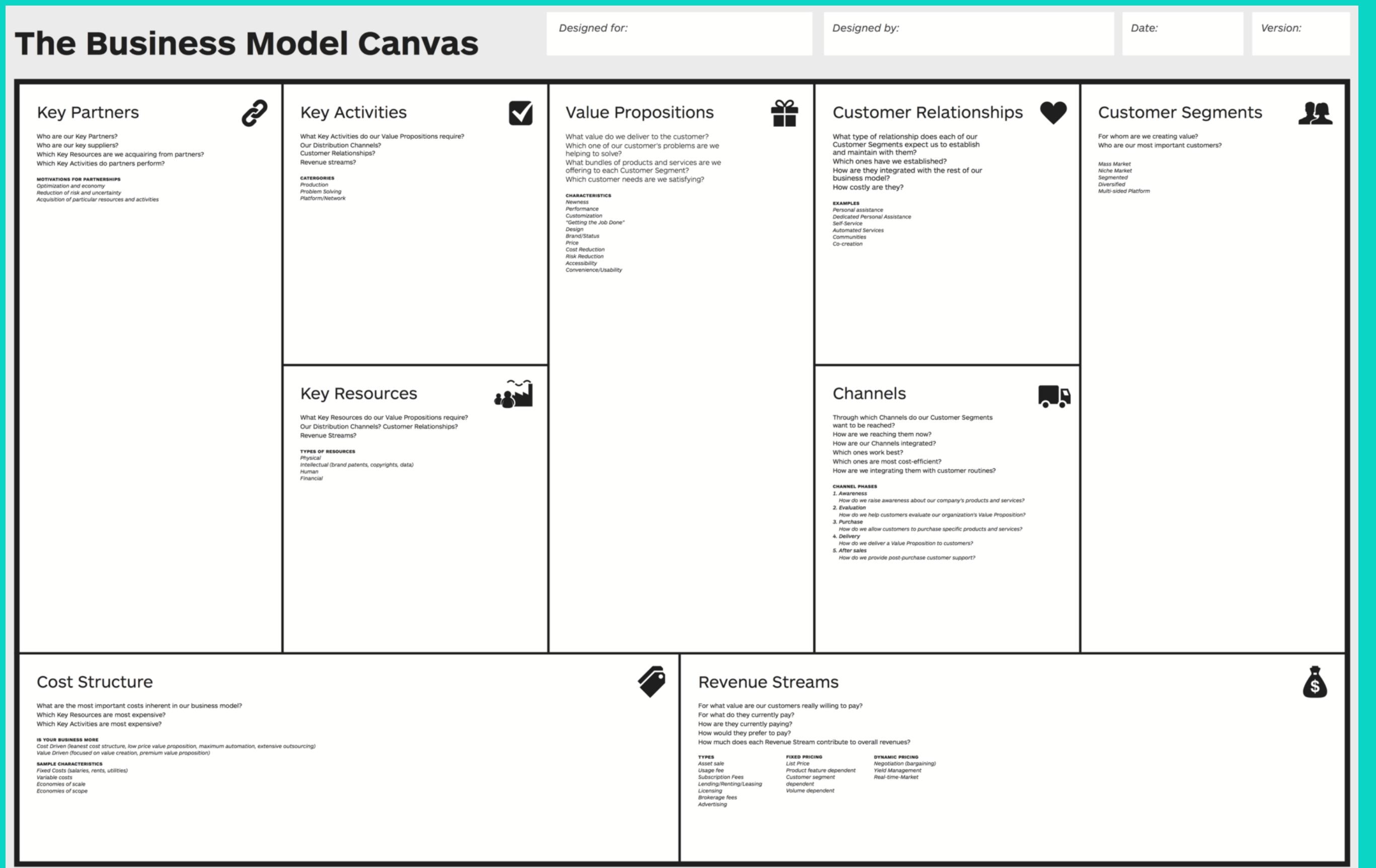


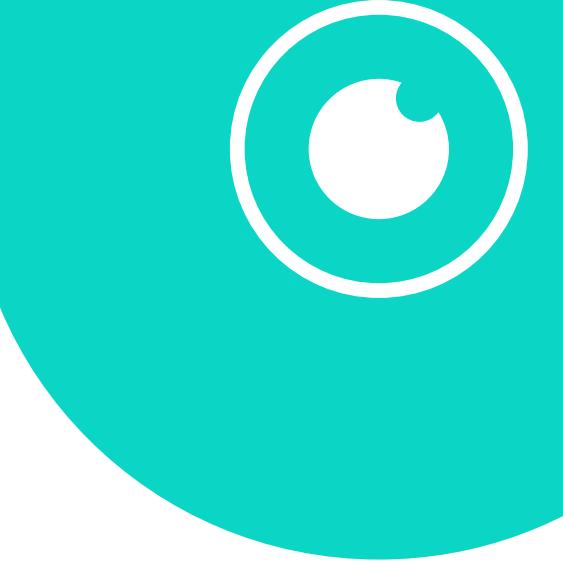
El Business Model Canvas (BMC)



La imagen es el primer resultado de Google Imágenes para la búsqueda "Business Model Canvas"
Tomada de https://muyagile.com/wp-content/uploads/2019/11/1920px-Business_Model_Canvas.png

Lenguaje visual	
Concepto	Símbolo
Bloque del modelo de negocio	Nombre (ícono representativo) Texto de contenido del bloque
Relación entre bloques	Se da por posición

Evaluación - Visualización



Claridad Semiótica

- ✓ Se reconoce cada bloque como importante mientras se diferencian un poco entre ellos gracias al uso sutil del ícono.
- ✗ No se representan algunos atributos del bloque de modelo de negocio definidos por su creador, como el pilar.

Discriminabilidad perceptiva

- ✓ Como cada uno de los bloques tiene su propio borde y posición, es fácil distinguir unos de otros, y distinguir los símbolos.
- ✗ Las relaciones no son completamente claras con esta disposición de los elementos.

Manejo de complejidad

- ✓ Utiliza dos símbolos o representaciones visuales, que facilitan el entendimiento y la diferenciación de símbolos.
- ✗ En el modelo tradicional, no hay una forma de entender en detalle cada uno de los bloques del modelo de negocio.

Expresividad visual

- ✓ El uso de los íconos, las cajas y la posición de los elementos permite que sea bastante expresivo. Balance entre textual y visual.
- ✗ Se podría utilizar color y/o intensidad para diferenciar los bloques o incluso para agruparlos.

Ajuste cognitivo

- ✓ Este modelo sencillo, con un lenguaje de baja complejidad, permite que sea fácil de usar con o sin experiencia.
- ✗ No se propone el uso de visualizaciones de detalle para personas expertas, las cuales podrían ser de gran utilidad.

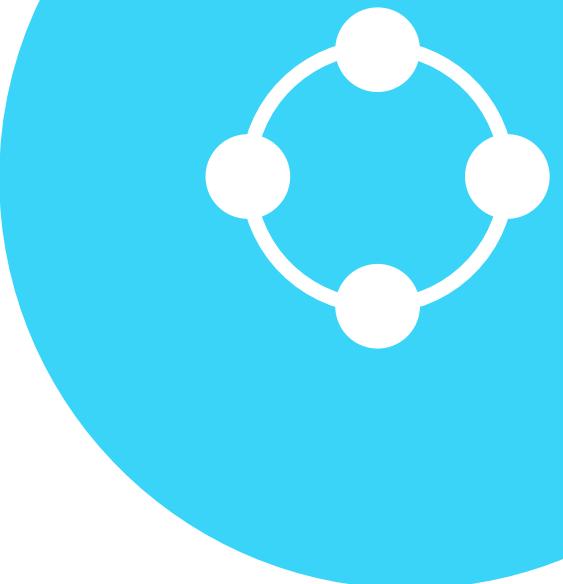
Transparencia Semántica

- ✓ El uso de íconos permite diferenciar cada uno de los bloques de acuerdo a su propósito y su nombre.
- ✗ Como las relaciones no son explícitas, estas podrían ser completamente ignoradas o reinterpretadas por el receptor.

Codificación dual

- ✓ Hay un muy buen balance entre lo textual y lo gráfico. Además, los elementos de texto y visuales conviven de forma natural.
- ✗ Algunas veces, dado el espacio, el contenido textual puede sobrepasar al contenido visual.

Evaluación - Colaboración



Sensación de terminado

- ✗ El lenguaje definido para el Canvas lleva a una estructura de bloque que se siente completamente terminada. Este principio depende completamente de la cantidad de texto que se ponga en cada caja, ya que si hay poco se siente la posibilidad de agregar más cosas y viceversa.

Impacto visual

- ✓ El Canvas permite ver los distintos aspectos que componen un negocio como un solo bloque, lo cual se alinea con la idea de compañía. Además, la organización por posición lleva la atención del centro hacia el exterior y genera intriga.

Generación de insights

- ✓ La cercanía de los componentes permite ver con claridad relaciones importantes.
- ✗ La estructura de bloque de este lenguaje hace que sea difícil ver insights entre bloques que no están usualmente conectados.

Modificabilidad

- ✓ El contenido de los bloques del lenguajes es de fácil modificación.
- ✗ La posición de los elementos es la misma en la mayoría de templates encontrados en Internet, lo cuál genera una limitación de modificar las relaciones.

Concentración dirigida

- ✓ Al usar pocos íconos, y además de tamaños pequeños, el modelo Canvas permite captar la atención de los receptores y mantener la concentración.

Manejo del discurso

- ✓ El nivel de abstracción del Canvas y la disposición de los bloques facilita un manejo del discurso desde el centro, tomando la propuesta de valor como base para el resto del modelo.

Evaluación - Arquitectura



Orden

- ✓ El lenguaje y su orden basado en la posición de los bloques permite que los modelos resultante sean muy ordenados.
- ✗ Este orden por posición también es una gran restricción ya que no permite "jugar" con los bloques y ver relaciones ocultas.

Consideración de puntos de vista

- ✓ La teoría original contiene sugerencias para el manejo de puntos de vista
- ✗ El lenguaje no contempla la posibilidad de puntos de vista de detalle, es decir, no se define un lenguaje para conceptos más detallados de cada bloque.

Escalabilidad

- ✓ Es escalable en la granularidad de contenido ya que permite agregar el contenido necesario a cada bloque.
- ✗ El modelo utilizado convencionalmente no es escalable en la granularidad de bloque, ya que se reduce a un bloque conformado siempre por los mismos bloques.

Claridad de las relaciones

- ✗ Las relaciones no son completamente claras ya que se basan en la posición de los componentes. Si una persona que no conoce el lenguaje encuentra el "template" del canvas y lo usa, puede que no entienda que los distintos bloques se relacionan entre sí.

Nivel de abstracción

- ✓ Buen nivel de abstracción. Balance en el uso de formas y de íconos.
- ✗ La abstracción de las relaciones las vuelve complicadas de ver. Debería reducirse la abstracción de esta representación.

Consistencia

- ✓ Gracias a su lenguaje sencillo y de baja complejidad, con pocos símbolos, los modelos creados con él siguen una misma línea gráfica y es difícil que queden muy diferentes entre sí.

Diferenciación de rels., conc. y atr.

- ✓ El lenguaje diferencia estos tres frentes correctamente en los símbolos que contiene.
- ✗ No se toman en consideración atributos que podrían ser de utilidad, como el pilar al que pertenece el bloque.

Conclusiones

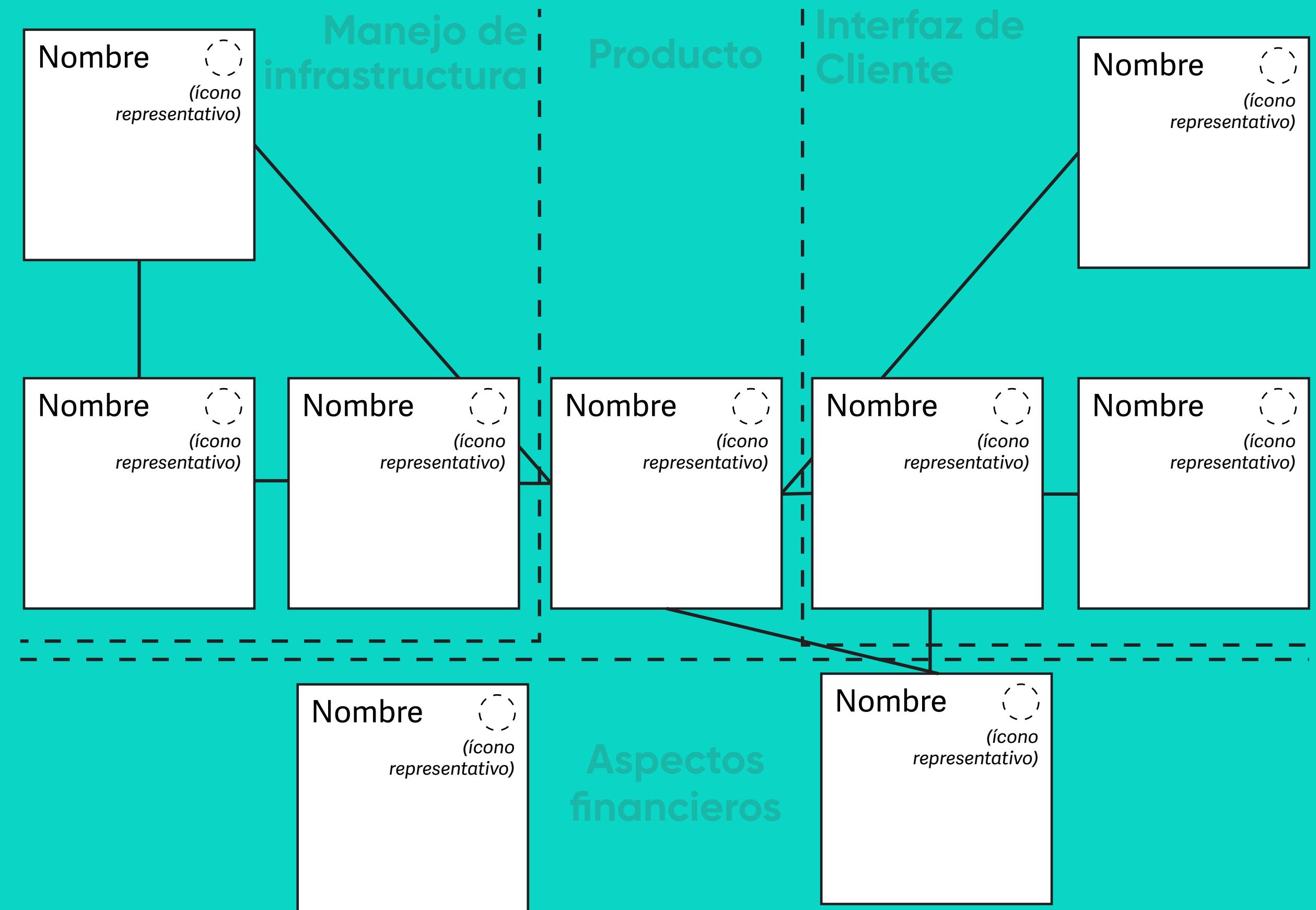
Posibilidad de innovación del BMC

El template de BMC que se encuentra múltiples veces en internet es fácil de utilizar ya que tiene un lenguaje de baja complejidad. Además, al ser tan reconocido, es fácil que otras personas lo entiendan y puedan trabajar sobre este.

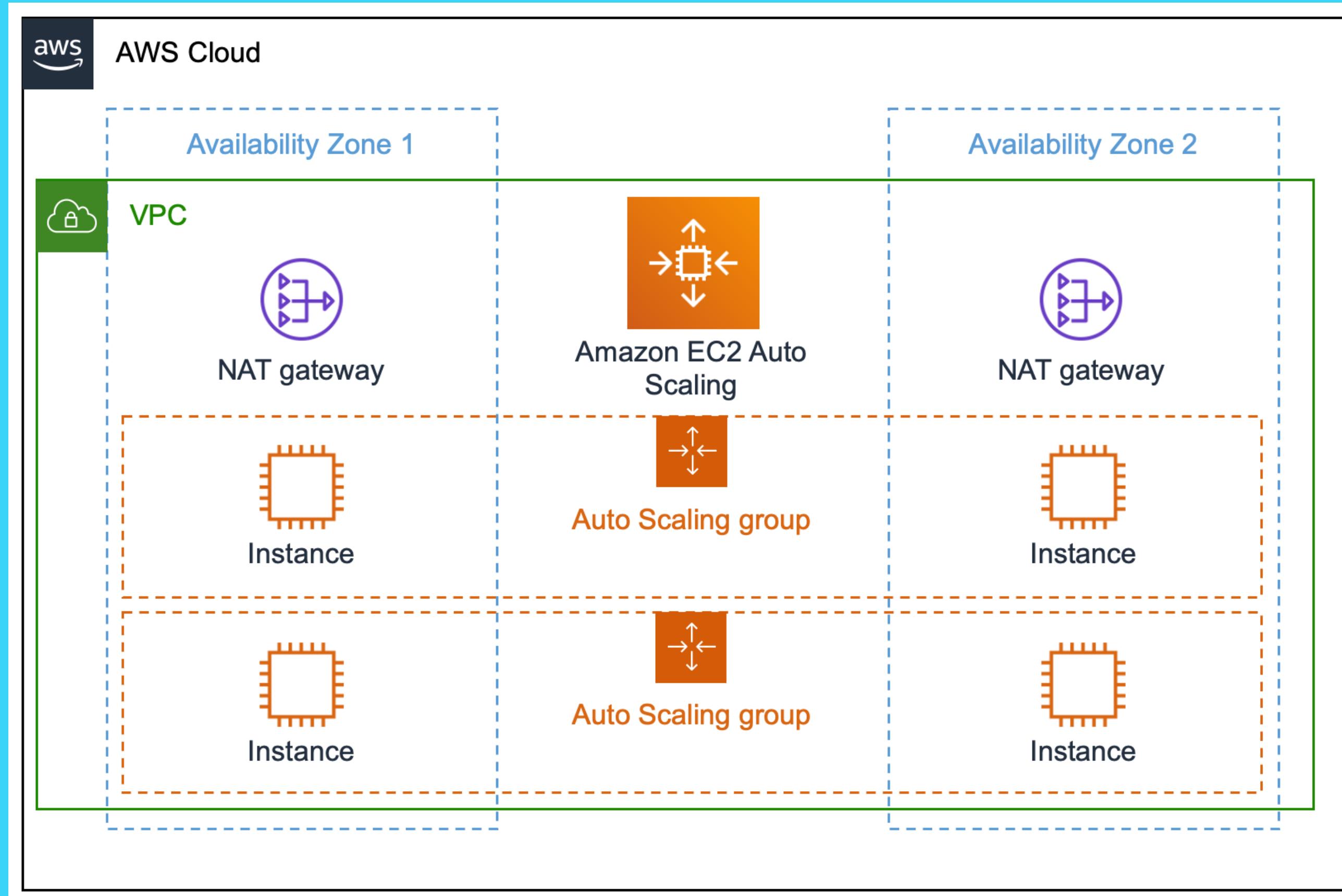
Aunque el BMC se utiliza muchas veces en el proceso de definición de un modelo de negocio, debería usarse al final del proceso para plasmar un modelo de negocio ya definido, ya que es un modelo ordenado, que se ve terminado y realmente es de gran utilidad para guiar conversaciones y tomas de decisiones, pero no es tan flexible para un proceso de ideación.

Al ser un template y tener una estructura de bloque, este lenguaje tradicional no facilita la modificación y la exploración de nuevos bloques que podrían aparecer gracias a los avances tecnológicos y la BMI (Business Model Innovation). Por ende, termina siendo restrictivo y haciendo más difícil el proceso de generación de insights o de brainstorming. Para estos procesos, se sugiere usar versiones desacopladas del modelo, en las que se ve la posibilidad de utilizar más variables visuales e incluso introducir nuevos atributos en el lenguaje visual.

Además, se ve la posibilidad de explorar las vistas de detalle propuestas por el autor que creó el BMC, para modularizar los modelos y ver más en detalle cada uno de los bloques que conforma el modelo de negocio.



Modelos de AWS



Lenguaje visual	
Concepto	Símbolo
Grupo Conexión entre múltiples servicios y recursos	AWS Cloud Siempre ícono y nombre. Relaciones usualmente de contención. Generic group
Relaciones (flechas) Describe el flujo de información y/o conecta partes del diagrama.	
Servicio Representan un servicio de AWS.	Hay un total de aprox. 269 íconos de servicios en 27 categorías que reutilizan una paleta de 7 colores
Recurso de servicio Representan un recurso de un servicio de AWS	No todos los servicios tienen recursos. De estos íconos hay un total de 404 recursos.
Recurso general Aplica a múltiples servicios o categorías.	

La imagen es el ejemplo que tiene AWS en el inicio de su paquete de recursos para diagramas de arquitectura AWS.
Tomada de <https://aws.amazon.com/architecture/icons/>

Evaluación - Visualización

Claridad Semiótica

- ✓ Todo concepto tiene un símbolo asociado, y cada uno tiene sus propias variaciones.
- ✗ Aquí hay un claro problema de economía gráfica. En el caso de estos diagramas, se podría contar con menos representaciones de los símbolos o mejores sistemas de agrupación con uso de texto.

Manejo de complejidad

- ✗ Aunque la cantidad de símbolos es pequeña, la cantidad de instancias de aquellos símbolos es muy grande. Debido a ello, se vuelve muy difícil entender estos diagramas sin conocer todos los íconos distintos. Se podría considerar la modularización a nivel de grupos, para luego pasar a los servicios de cada grupo.

Ajuste cognitivo

- ✗ El uso de la gran cantidad de íconos hace que los diagramas de AWS solo puedan ser entendidos por expertos en esta área. No es un lenguaje amigable a personas que están empezando a utilizar estos servicios.

Discriminabilidad perceptiva

- ✓ Debido al uso de múltiples íconos, los distintos símbolos tienen la posibilidad de ser fácilmente distinguibles.
- ✗ Sin embargo, esto también puede llevar a confundir los símbolos, si no hay un apoyo de texto. También es un problema que se repiten colores en las distintas secciones

Expresividad visual

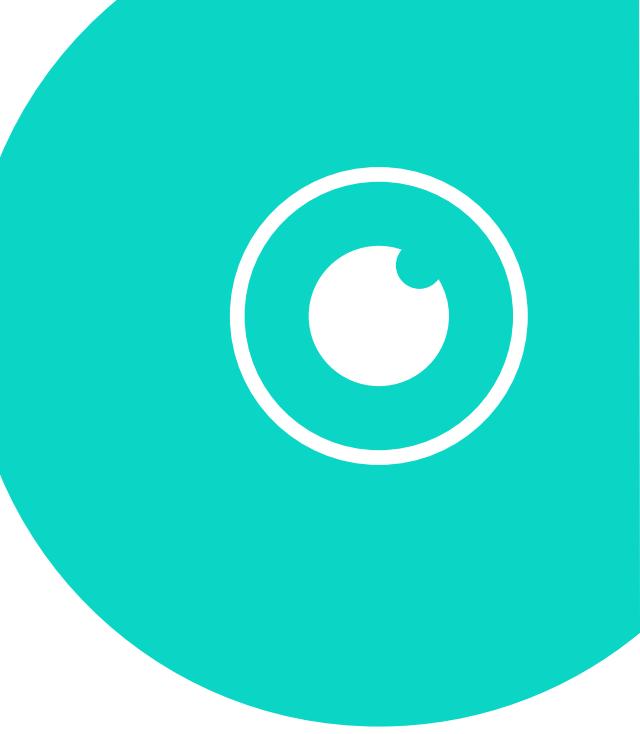
- ✓ El lenguaje hace uso de una gran cantidad de variaciones de forma, color y de relaciones de contención y con flechas.
- ✗ Se abusa un poco del uso de íconos y el uso de colores iguales en diferentes categorías, lo cual genera que estas variables visuales pierdan su utilidad.

Transparencia Semántica

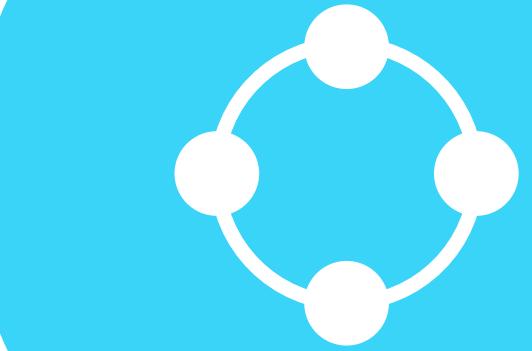
- ✓ El uso de íconos permite diferenciar los diferentes servicios de AWS y sus conexiones también permiten claridad.
- ✗ Si se hace uso únicamente de los íconos, sin su contraparte textual, todos pueden interpretarse de una forma diferente.

Codificación dual

- ✗ La codificación dual no es requerida por el lenguaje, se pueden usar únicamente los símbolos. Esto genera serios problemas de interpretación y entendimiento de los diagramas. Cuando va de la mano de pequeños textos descriptivos, es más fácil entender los diagramas y sus conexiones.



Evaluación - Colaboración



Sensación de terminado

- ✗ Un diagrama de AWS da la sensación de que ya está completo y no se pueden agregar más cosas. Esto se debe a la complejidad alta del lenguaje y el seccionamiento con el uso de múltiples cajas.

Impacto visual

- ✓ Gracias al uso de múltiples colores, cajas con distintas características e íconos, los diagramas llaman la atención de quién los ve.

Concentración dirigida

- ✗ Aunque tenga un gran impacto visual y llame la atención, la concentración del receptor se pierde rápidamente debido a la dificultad de lectura del diagrama. Además, la concentración se divide, ya que para lograr entender lo que está pasando se necesita de un manual o guía.

Generación de insights

- ✓ Gracias al uso de múltiples formas de representación de relaciones, y al espacio entre componentes, puede ser fácil encontrar insights de la arquitectura definida.

Modificabilidad

- ✓ Gracias al espacio entre componentes se ve la posibilidad de moverlos con facilidad y agregar cosas nuevas.
- ✗ Es difícil saber qué modificar debido a la dificultad de comprensión del diagrama.

Manejo del discurso

- ✗ Debido a la necesidad de conocer todos los símbolos para entender los diagramas creados con este lenguaje, el manejo del discurso no es una cualidad de este diagrama. Se recomienda usar otras representaciones nuevas para discutir con stakeholders que no conocen el lenguaje.

Evaluación - Arquitectura



Orden

- ✓ El lenguaje y el uso de múltiples símbolos, colores y cajas, dan como resultado visualizaciones organizadas
- ✗ Se debe evitar el cruce entre flechas y mantener suficiente espacio entre las instancias para que no hay más confusión.

Claridad de las relaciones

- ✓ En el lenguaje de AWS el uso de flechas y relaciones de contención permite que las relaciones se entiendan con facilidad, agrupando instancias y generando conexiones entre ellas.

Consideración de puntos de vista

- ✓ El uso de cajas para los grupos podría permitir crear una primera visualización general con cajas, para luego pasar al detalle de cada una con los símbolos.

Nivel de abstracción

- ✓ El nivel de abstracción es nivelado, aunque se usen muchos iconos sería difícil representar tantas instancias de otra manera.

Escalabilidad

- ✗ Los modelos creados con el lenguaje de AWS son difícilmente escalables. Si se quisiera utilizar un solo modelo para toda una arquitectura AWS, se necesitaría usar modularización. En caso contrario, un modelo muy grande es difícil de entender por la cantidad de diferentes íconos y colores que puede tener.

Consistencia

- ✓ El lenguaje utilizado en los textos que acompañan a los símbolos es consistente con el negocio y los conceptos relacionados a los servicios prestados por AWS.

Diferenciación de rels., conc. y atr.

- ✓ El lenguaje hace una distinción correcta de estos tres conceptos y genera los símbolos necesarios para definirlos.

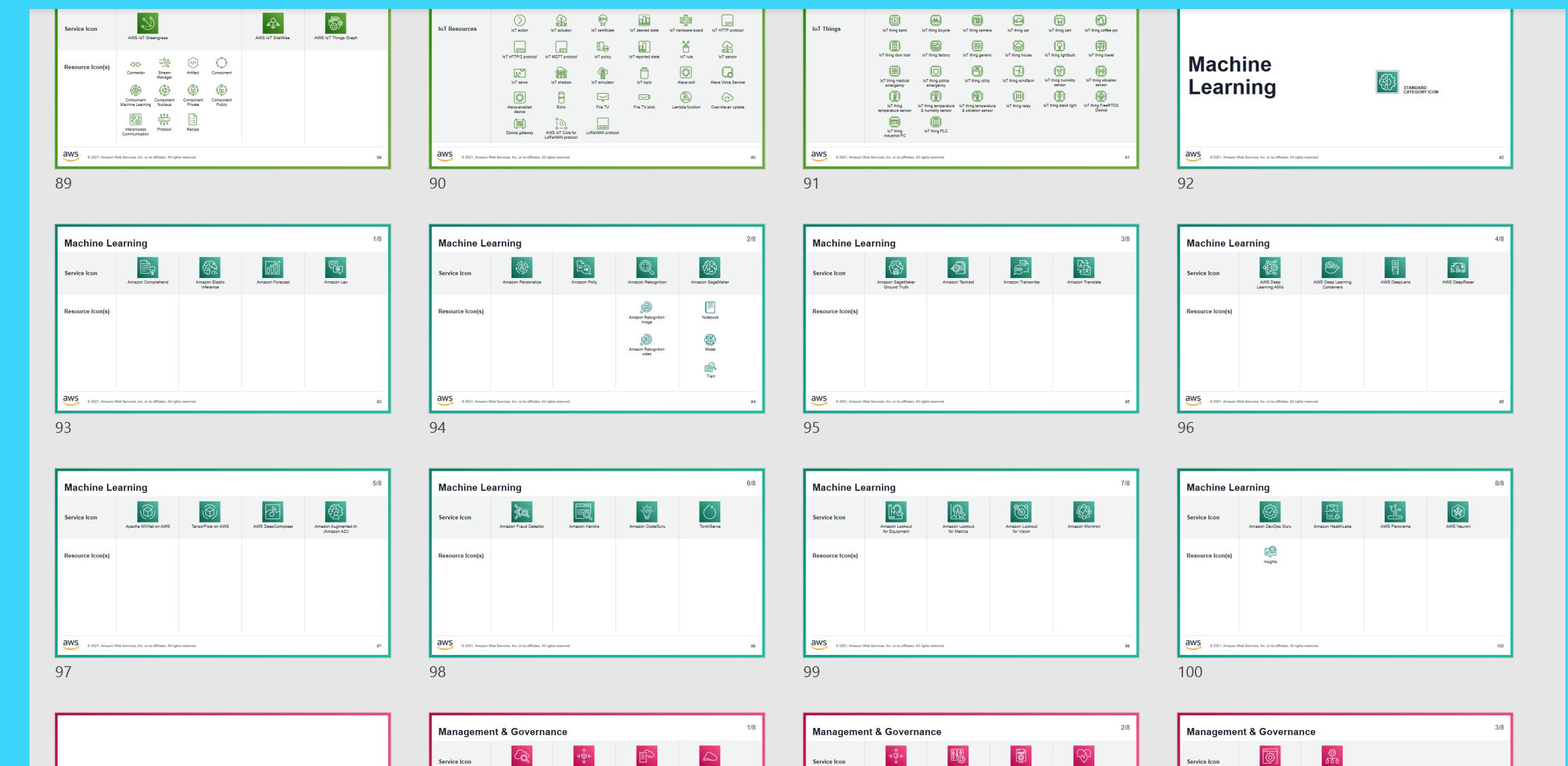
Conclusiones

Lenguaje AWS: solo para expertos

AWS es una tecnología muy conocida y muy utilizada para la creación de múltiples componentes de infraestructura tecnológica utilizados en una empresa y sus conexiones. Por ende, se muestra como una primera opción para varias empresas en crecimiento. Teniendo esto en cuenta, debería facilitar la creación de diagramas que permitan al sector de tecnología justificar sus decisiones de uso de ciertos servicios para el negocio en cuestión. Sin embargo, este lenguaje solo puede ser utilizado por personas expertas en su uso y conocedoras de AWS, y por ende solo puede comunicar efectivamente a estas mismas personas. Por ello, no se ve como un lenguaje para hablar con múltiples stakeholders en una empresa.

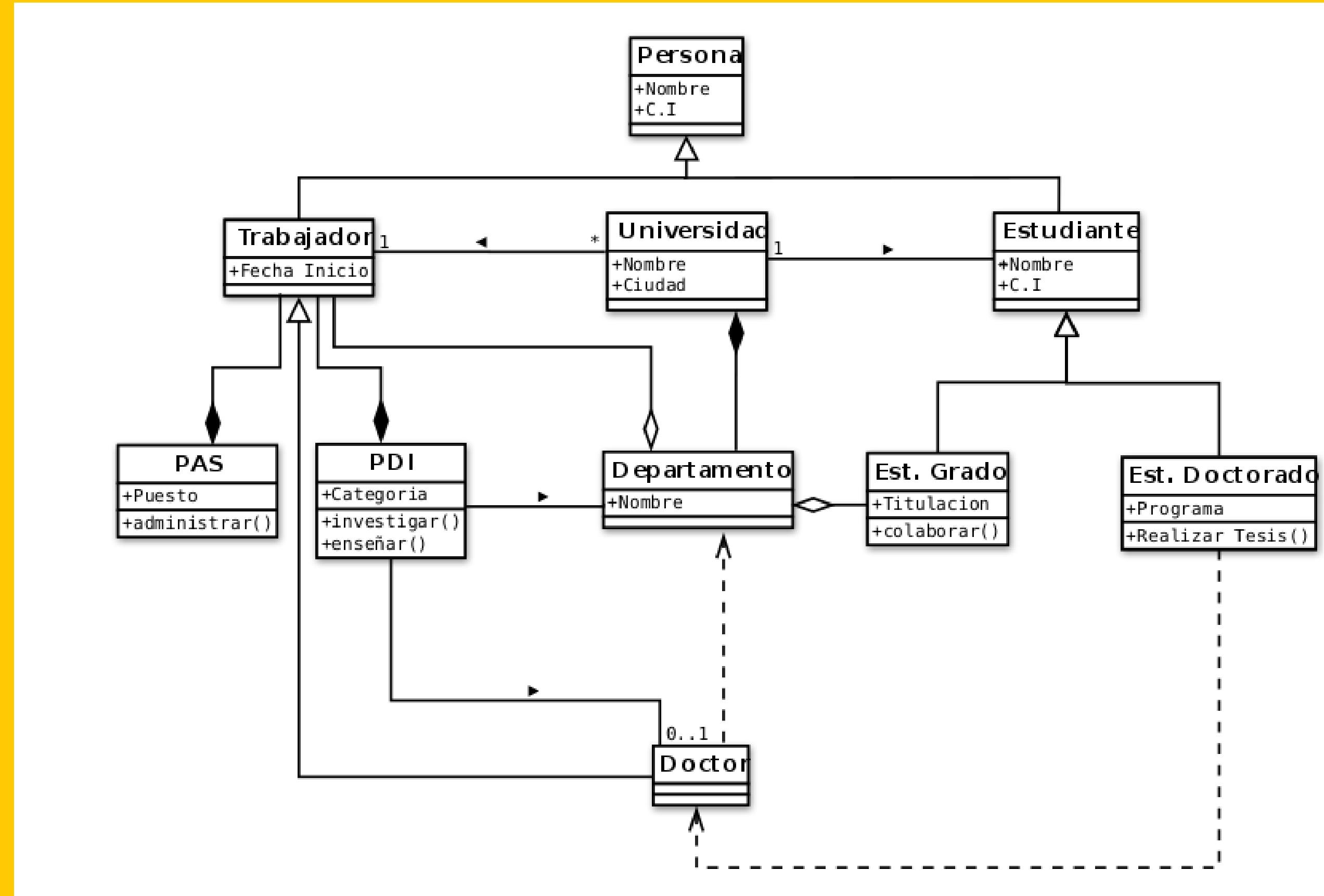
Los diagramas de AWS pueden ser de gran utilidad para organizar la arquitectura tecnológica de una empresa por módulos, ya que como se puede ver tiene puntos a favor en la sección de arquitectura. Sin embargo, son un recurso que se mantiene únicamente en el sector de tecnología, probablemente en DevOps, debido a la necesidad de conocimiento técnico previo. Se recomienda el uso de otros lenguajes o variaciones del lenguaje para procesos de toma de decisiones sobre la arquitectura que involucren a otros stakeholders, ya que como se puede ver en la sección de visualización, puede generar conflictos en la comunicación.

En caso de querer utilizar AWS, se recomienda modularización, uso de texto descriptivo para acompañar los servicios y recursos, y una lectura previa del paquete de recursos gráficos definido por Amazon.



UML - Unified Modeling Language

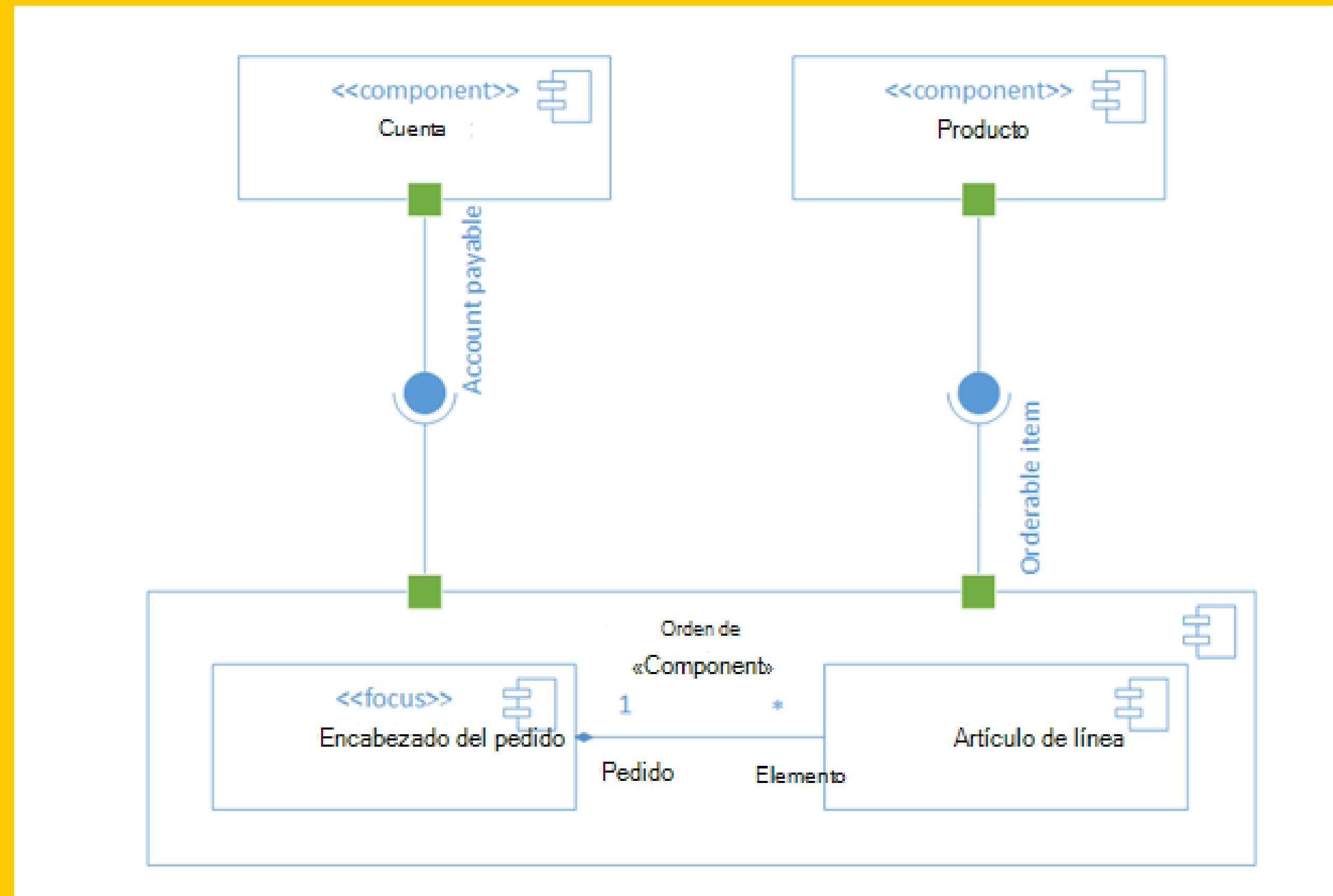
Diagrama de clases

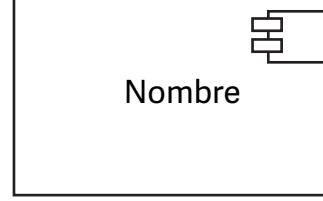
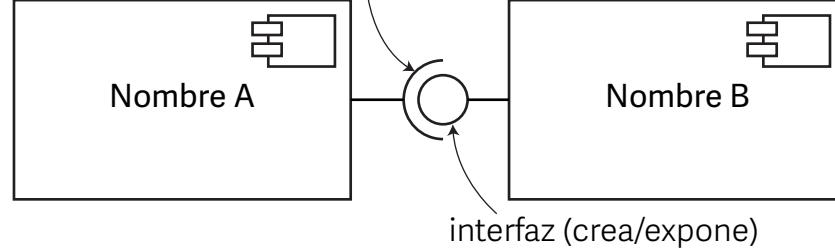
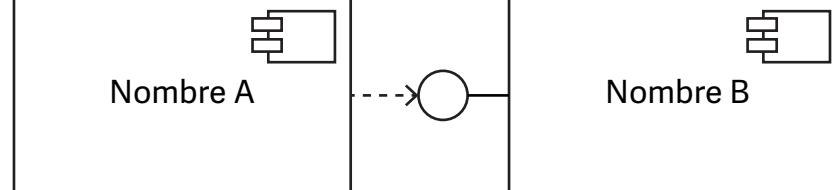
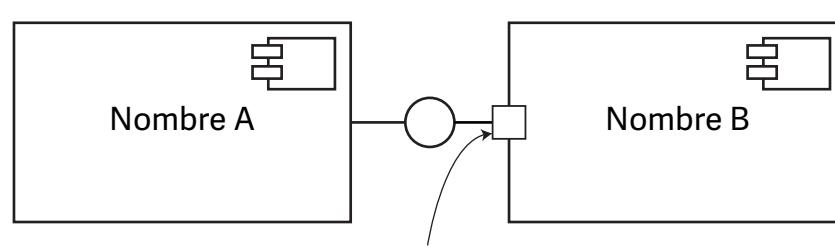


Lenguaje visual													
Concepto	Símbolo												
Clase Elemento o concepto.	<table border="1"> <tr> <td>Nombre</td> <td>Clase abstracta o Interfaz</td> </tr> <tr> <td>atributos</td> <td>Nombre método/función() método virtual()</td> </tr> <tr> <td>métodos/functions</td> <td>Nombre {Abstract}</td> </tr> <tr> <td>+ f. o a. público</td> <td></td> </tr> <tr> <td># f. o a. protegido</td> <td></td> </tr> <tr> <td>- f. o a. privado</td> <td></td> </tr> </table> <p>Notación visibilidad: + f. o a. público # f. o a. protegido - f. o a. privado</p>	Nombre	Clase abstracta o Interfaz	atributos	Nombre método/función() método virtual()	métodos/functions	Nombre {Abstract}	+ f. o a. público		# f. o a. protegido		- f. o a. privado	
Nombre	Clase abstracta o Interfaz												
atributos	Nombre método/función() método virtual()												
métodos/functions	Nombre {Abstract}												
+ f. o a. público													
# f. o a. protegido													
- f. o a. privado													
Relaciones	<table border="1"> <tr> <td>asociación</td> <td>A rol de A B rol de B</td> </tr> <tr> <td>dependencia y realización</td> <td>A es una interfaz, B la utiliza y depende de ella A es una interfaz, B la implementa</td> </tr> <tr> <td>herencia</td> <td>A B también es un A</td> </tr> <tr> <td>composición</td> <td>A B B es parte de A y solo de A La existencia de B depende de A</td> </tr> <tr> <td>agregación</td> <td>A B B es parte de A, puede ser parte de otras clases. Existen de forma independiente</td> </tr> </table>	asociación	A rol de A B rol de B	dependencia y realización	A es una interfaz, B la utiliza y depende de ella A es una interfaz, B la implementa	herencia	A B también es un A	composición	A B B es parte de A y solo de A La existencia de B depende de A	agregación	A B B es parte de A, puede ser parte de otras clases. Existen de forma independiente		
asociación	A rol de A B rol de B												
dependencia y realización	A es una interfaz, B la utiliza y depende de ella A es una interfaz, B la implementa												
herencia	A B también es un A												
composición	A B B es parte de A y solo de A La existencia de B depende de A												
agregación	A B B es parte de A, puede ser parte de otras clases. Existen de forma independiente												
Multiplicidad	<table border="1"> <tr> <td>cero o más</td> <td>*</td> </tr> <tr> <td>1 exacto</td> <td>1</td> </tr> <tr> <td>n exactos</td> <td>n</td> </tr> <tr> <td>cero o uno</td> <td>0 .. 1</td> </tr> <tr> <td>1 o más</td> <td>1 .. *</td> </tr> <tr> <td>de n a m</td> <td>n .. m</td> </tr> </table> <p>Cada A se relaciona con 1 B Cada B se relaciona con cero o más A's</p>	cero o más	*	1 exacto	1	n exactos	n	cero o uno	0 .. 1	1 o más	1 .. *	de n a m	n .. m
cero o más	*												
1 exacto	1												
n exactos	n												
cero o uno	0 .. 1												
1 o más	1 .. *												
de n a m	n .. m												

UML - Unified Modeling Language

Diagrama de componentes



Lenguaje visual	
Concepto	Símbolo
Componente Un bloque lógico del sistema. Abstracción más alta que una clase	 Nombre
Interfaz Una interfaz describe un grupo de operaciones utilizadas o creadas por los componentes e.g. A consume B	 interfaz requerida (usa/consume) interfaz (crea/expone)
Dependencia e.g. A depende de B	 Nombre A -----> Nombre B
Puerto Un puerto expone los recursos que un componente provee y requiere	 Nombre A ---○--- Nombre B puerto

Evaluación - Visualización



Claridad Semiótica

- ✓ Todo concepto tiene un símbolo asociado, y cada uno tiene sus propias variaciones.

Discriminabilidad perceptiva

- ✓ Las distintas relaciones se diferencian con facilidad. Las clases o componentes también, aunque ambos sean rectángulos.
- ✗ En algunos casos es difícil separar una variación de un concepto de otra, como en el caso de las clases normales y abstractas ,donde solo cambia la inclinación del nombre.

Manejo de complejidad

- ✓ Debido a la reducida cantidad de símbolos, la complejidad de estos diagramas es pequeña. Esto permite que los diagramas puedan crecer sin generar mayores efectos sobre la comprensión de los mismos. Además, aunque los diagramas de clases y componentes ambos usen UML, sus componentes se definen por separado.

Expresividad visual

- ✗ El lenguaje podría utilizar otras variables visuales útiles como la forma (pequeñas variaciones en las esquinas de los cuadros) y el color.

Ajuste cognitivo

- ✓ El uso de figuras básicas, en las que predomina el rectángulo, hace que sea sencillo entender el lenguaje y que sea de fácil uso para una persona que lo conoce por primera vez. Este lenguaje es apto para uso de novatos y expertos.

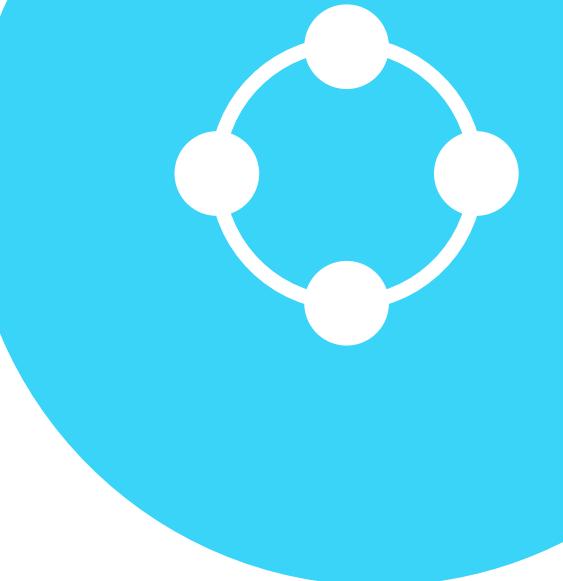
Transparencia Semántica

- ✓ Debido a que los elementos que se usan en cada diagrama son independientes (como las clases y componentes) es difícil que haya una confusión entre estos conceptos.
- ✗ UML tiene un nivel de abstracción bastante alto, ya que no utiliza iconos ni figuras que representen los conceptos asociados. Al basar los elementos en rectángulos, puede haber problemas de interpretación.

Codificación dual

- ✓ UML tiene un fuerte componente de codificación dual, ya que la mayoría de sus símbolos van acompañados de algún texto que facilita la comprensión de los diagramas. Además, esta permite navegar entre distintos niveles de granularidad al permitir ver el detalle de métodos y atributos de cada clase en el diagrama de clases.

Evaluación - Colaboración



Sensación de terminado

- ✓ Los diagramas UML dan una sensación de terminado suficiente para poder utilizarse para plasmar el diagrama final, pero también se ven abiertos a edición debido a su estructura de cajas y relaciones.

Impacto visual

- ✗ Debido a que no se utilizan muchas variables visuales, los diagramas UML en blanco y negro y sin rellenos no llaman mucho la atención.

Generación de insights

- ✓ Gracias al uso de múltiples formas de representación de relaciones, y al espacio entre componentes, puede ser fácil encontrar insights relacionados a los conceptos o componentes.

Modificabilidad

- ✓ Gracias al espacio entre componentes se ve la posibilidad de moverlos con facilidad y agregar cosas nuevas.
- ✗ Debido a la gran cantidad de relaciones que puede haber en un diagrama UML, es necesario un proceso de reorganización de los elementos cada vez que se agrega uno nuevo.

Concentración dirigida

- ✓ El uso de flechas para las relaciones, que muchas veces tienen una dirección, permite navegar entre los distintos elementos y por ende mantener la concentración de los receptores.

Manejo del discurso

- ✓ El uso de flechas para las relaciones, que muchas veces tienen una dirección, permite navegar entre los distintos elementos y por ende guiar a los receptores en un discurso que use el diagrama como base.

Evaluación - Arquitectura



Orden

- ✓ El lenguaje y sus símbolos basados en líneas y rectángulos permite que los diagramas resultante tengan un orden.
- ✗ Se debe evitar el cruce entre flechas y mantener suficiente espacio entre las instancias para que no hay más confusión.

Consideración de puntos de vista

- ✓ Con UML es sencillo modularizar, ya que basta con la conservación de las clases relevantes y las relaciones que hay entre ellos.
- ✗ Se puede pensar en un mecanismo de agrupación, que permita ver las relaciones entre grupos de una granularidad más baja.

Escalabilidad

- ✓ Debido a la reducida cantidad de símbolos, la complejidad de estos diagramas es pequeña. Esto permite que los diagramas puedan crecer sin generar mayores efectos sobre la comprensión de los mismos.
- ✗ El mayor problema de escalabilidad en UML es que las relaciones crecen mucho y empiezan a cruzarse generando problemas de interpretación.

Claridad de las relaciones

- ✓ El uso de flechas y la clasificación de estas en diferentes tipos, diferenciados por algunas variables visuales, permite que sea claro cómo se relacionan las clases y/o los componentes.

Nivel de abstracción

- ✓ El nivel de abstracción es alto, pero sencillo de interpretar al conocer la notación del lenguaje.

Consistencia

- ✓ UML define todo un estándar gráfico que permite que las visualizaciones diferentes hechas con él sean consistentes. Además, aunque se divide en varias notaciones para diferentes diagramas, hay consistencia entre sus elementos gráficos (e.g. en el de clases y el de componentes, la dependencia se representa con la misma flecha).

Diferenciación de rels., conc. y atr.

- ✓ El lenguaje hace una distinción correcta de estos tres conceptos y genera los símbolos necesarios para definirlos.
- Como las instancias son tan abiertas, en UML puede ser sencillo confundir lo que es una relación, un concepto o un atributo.

Conclusiones

UML: Un lenguaje versátil

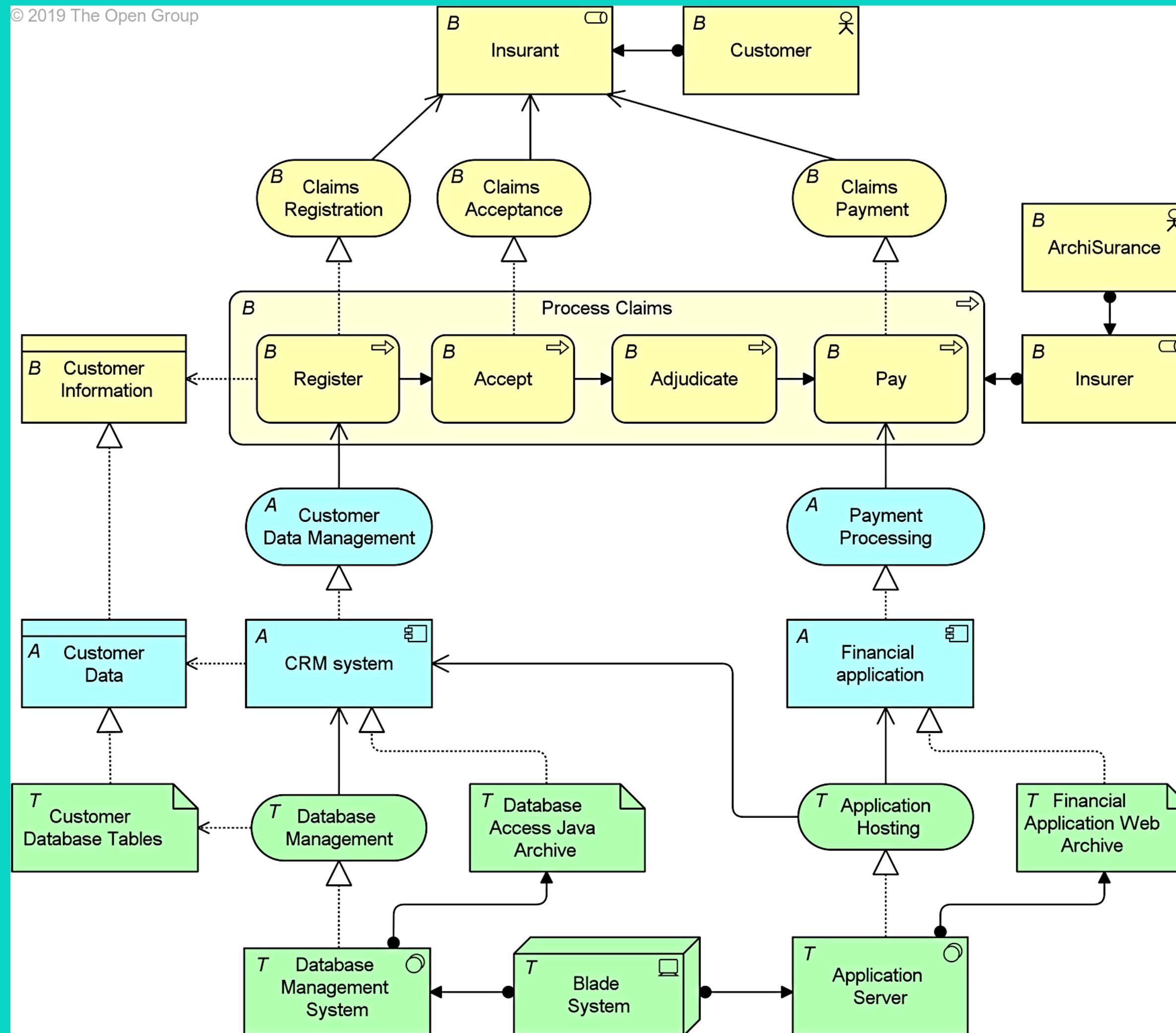
UML es el estándar utilizado para los diagramas de clases, componentes, de despliegue y de secuencia. Esto demuestra que, gracias a que se basa en figuras sencillas y no utiliza demasiadas variables visuales, puede adaptarse a varios contextos. Además, un diagrama como el de clases puede ser utilizado para múltiples propósitos, tanto en contextos de software como en contextos de arquitectura (e.g. los metamodelos).

En UML hay una fuerte presencia de texto que junto con las relaciones permite guiar fácilmente a quien lee el diagrama resultado. Por ello, facilita la creación conjunta de diagramas y la generación de discusiones a partir de los diagramas con cualquier grupo de stakeholders.

La sencillez del lenguaje permite que este pueda escalar con facilidad, por lo que los diagramas pueden ser modificados sin mayor esfuerzo en procesos de revisión e innovación. Gracias a ello, este lenguaje se puede utilizar para presentar un diagrama terminado pero también para un proceso de creación conjunta y brainstorming.

UML se encuentra como un lenguaje amigable, con una curva de aprendizaje más rápida que la de otros lenguajes aquí analizados debido a su pequeña cantidad de símbolos y la claridad de sus relaciones. Sin embargo, algo que puede ocurrir en el proceso es que se utilicen de forma incorrecta los conceptos de clase, atributo y relación. El uso correcto de estos conceptos requiere de mayor experiencia en el uso del lenguaje, lo cual puede hacer más lenta la curva de aprendizaje.

Archimate



La imagen es un ejemplo extraído de la Especificación de Archimate
Tomada de https://pubs.opengroup.org/architecture/archimate3-doc/chap12.html#_Toc10045440

ArchiMate - Notación

Concepts and their Notation

The ArchiMate language separates the language concepts (i.e., the constituents of the metamodel) from their notation. Different stakeholder groups may require different notations in order to understand an architecture model or view. In this respect, the ArchiMate language differs from languages such as UML or BPMN™, which have only one standardized notation. The viewpoint mechanism explained in Chapter 14 provides the means for defining such stakeholder-oriented visualizations.

Although the notation of the ArchiMate concepts can (and should) be stakeholder-specific, the standard provides one common graphical notation which can be used by architects and others who develop ArchiMate models. This notation is targeted towards an audience used to existing technical modeling techniques such as Entity Relationship Diagrams (ERDs), UML, or BPMN, and therefore resembles them. In the remainder of this document, unless otherwise noted, the symbols used to depict the language concepts represent the ArchiMate standard notation. This standard notation for most elements consists of a box with an icon in the upper-right corner. In several cases, this icon by itself may also be used as an alternative notation. This standard iconography should be preferred whenever possible so that anyone knowing the ArchiMate language can read the diagrams produced in the language.

Use of Colors and Notational Cues

In the metamodel pictures within this standard, shades of grey are used to distinguish elements belonging to the different aspects of the ArchiMate framework, as follows:

- White for abstract (i.e., non-instantiable) concepts
- Light grey for passive structures
- Medium grey for behavior
- Dark grey for active structures

In ArchiMate models, there are no formal semantics assigned to colors and the use of color is left to the modeler. However, they can be used freely to stress certain aspects in models. For instance, in many of the example models presented in this standard, colors are used to distinguish between the layers of the ArchiMate Core Framework, as follows:

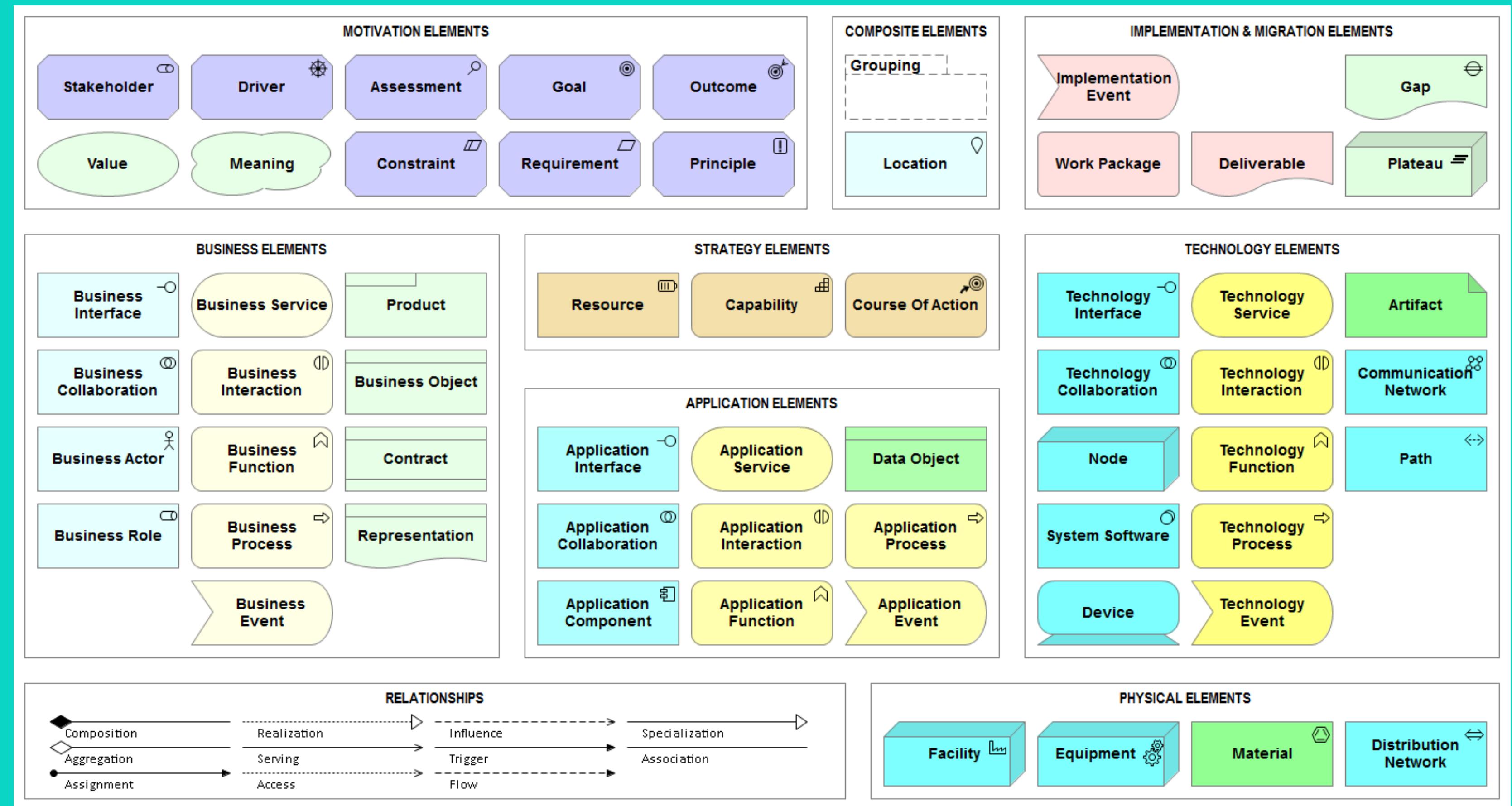
- Yellow for the Business Layer
- Blue for the Application Layer
- Green for the Technology Layer

They can also be used for visual emphasis. A recommended text providing guidelines is Chapter 6 of [1].

In addition to the colors, other notational cues can be used to distinguish between the layers of the framework. A letter M, S, B, A, T, P, or I in the top-left corner of an element can be used to denote a Motivation, Strategy, Business, Application, Technology, Physical, or Implementation & Migration element, respectively. An example of this notation is depicted in Example 34.

The standard notation also uses a convention with the shape of the corners of its symbols for different element types, as follows:

- Square corners are used to denote structure elements
- Round corners are used to denote behavior elements
- Diagonal corners are used to denote motivation elements



ArchiMate - Notación

ArchiMate® 3.1 Specification		
Relationships		
Structural Relationships		Notation
Composition	Represents that an element consists of one or more other concepts.	
Aggregation	Represents that an element combines one or more other concepts.	
Assignment	Represents the allocation of responsibility, performance of behavior, storage, or execution.	
Realization	Represents that an entity plays a critical role in the creation, achievement, sustenance, or operation of a more abstract entity.	
Dependency Relationships		Notation
Serving	Represents that an element provides its functionality to another element.	
Access	Represents the ability of behavior and active structure elements to observe or act upon passive structure elements.	
Influence	Represents that an element affects the implementation or achievement of some motivation element.	
Association	Represents an unspecified relationship, or one that is not represented by another ArchiMate relationship.	
Dynamic Relationships		Notation
Triggering	Represents a temporal or causal relationship between elements.	
Flow	Represents transfer from one element to another.	
Other Relationships		Notation
Specialization	Represents that an element is a particular kind of another element.	
Relationship Connectors		Notation
Junction	Used to connect relationships of the same type.	
	(And) Junction	○
	Or Junction	○

N190 Personal PDF Edition Copyright © 2019 The Open Group. All Rights Reserved. ArchiMate® is a registered trademark of The Open Group.

ArchiMate® 3.1 Specification		
Motivation Elements		
Element	Definition	Notation
Stakeholder	Represents the role of an individual, team, or organization (or classes thereof) that represents their interests in the effects of the architecture.	Stakeholder
Driver	Represents an external or internal condition that motivates an organization to define its goals and implement the changes necessary to achieve them.	Driver
Assessment	Represents the result of an analysis of the state of affairs of the enterprise with respect to some driver.	Assessment
Goal	Represents a high-level statement of intent, direction, or desired end state for an organization and its stakeholders.	Goal
Outcome	Represents an end result.	Outcome

N190 Personal PDF Edition Copyright © 2019 The Open Group. All Rights Reserved. ArchiMate® is a registered trademark of The Open Group.

ArchiMate® 3.1 Specification		
Motivation Elements – continued		
Element	Definition	Notation
Principle	Represents a statement of intent defining a general property that applies to any system in a certain context in the architecture.	Principle
Requirement	Represents a statement of need defining a property that applies to a specific system as described by the architecture.	Requirement
Constraint	Represents a factor that limits the realization of goals.	Constraint
Meaning	Represents the knowledge or expertise present in, or the interpretation given to, a concept in a particular context.	Meaning
Value	Represents the relative worth, utility, or importance of a concept.	Value

N190 Personal PDF Edition Copyright © 2019 The Open Group. All Rights Reserved. ArchiMate® is a registered trademark of The Open Group.

ArchiMate® 3.1 Specification		
Strategy Elements		
Element	Definition	Notation
Resource	Represents an asset owned or controlled by an individual or organization.	Resource
Capability	Represents an ability that an active structure element, such as an organization, person, or system, possesses.	Capability
Value stream	Represents a sequence of activities that create an overall result for a customer, stakeholder, or end user.	Value stream
Course of action	Represents an approach or plan for configuring some capabilities and resources of the enterprise, undertaken to achieve a goal.	Course of action

ArchiMate® 3.1 Specification		
Technology Layer		
Element	Definition	Notation
Node	Represents a computational or physical resource that hosts, manipulates, or interacts with other computational or physical resources.	Node
Device	Represents a physical IT resource upon which system software and artifacts may be stored or deployed for execution.	Device
System software	Represents software that provides or contributes to an environment for storing, executing, and using software or data deployed within it.	System software
Technology collaboration	Represents an aggregate of two or more technology internal active structure elements that work together to perform collective technology behavior.	Technology collaboration
Technology interface	Represents a point of access where technology services offered by a node can be accessed.	Technology interface
Path	Represents a link between two or more nodes, through which these nodes can exchange data, energy, or material.	Path
Communication network	Represents a set of structures and behaviors that connects nodes for transmission, routing, and reception of data.	Communication network

ArchiMate® 3.1 Specification		
Composite Elements		
Element	Definition	Notation
Grouping	The grouping element aggregates or composes concepts that belong together based on some common characteristic.	Grouping
Location	A location is a place or position where structure elements can be located or behavior can be performed.	Location

ArchiMate® 3.1 Specification		
Business Layer		
Element	Definition	Notation
Business actor	Represents a business entity that is capable of performing behavior.	Business actor
Business role	Represents the responsibility for performing specific behavior, to which an actor can be assigned, or the part an actor plays in a particular action or event.	Business role
Business collaboration	Represents an aggregate of two or more business internal active structure elements that work together to perform collective behavior.	Business collaboration
Business interface	Represents a point of access where a business service is made available to the environment.	Business interface
Business process	Represents a sequence of business behaviors that achieves a specific result such as a defined set of products or business services.	Business process
Business function	Represents a collection of business behavior based on a chosen set of criteria (typically required business resources and/or competencies), closely aligned to an organization, but not necessarily explicitly governed by the organization.	Business function

ArchiMate® 3.1 Specification		
Technology Layer – continued		
Element	Definition	Notation
Technology function	Represents a collection of technology behavior that can be performed by a node.	Technology function
Technology process	Represents a sequence of technology behaviors that achieves a specific result.	Technology process
Technology interaction	Represents a unit of collective technology behavior performed by (a collaboration of) two or more nodes.	Technology interaction
Technology event	Represents a technology state change.	Technology event
Technology service	Represents an explicitly defined exposed technology behavior.	Technology service
Artifact	Represents a piece of data that is used or produced in a software development process, or by deployment and operation of an IT system.	Artifact

ArchiMate® 3.1 Specification		
Business Layer – continued		
Element	Definition	Notation
Business interaction	Represents a unit of collective business behavior performed by (a collaboration of) two or more business actors, business roles, or business collaborations.	Business interaction
Business event	Represents an organizational state change.	Business event
Business service	Represents explicitly defined behavior that a business role, business actor, or business collaboration exposes to its environment.	Business service
Business object	Represents a concept used within a particular business domain.	Business object
Contract	Represents a formal or informal specification of an agreement between a provider and a consumer that specifies the rights and obligations associated with a product and establishes functional and non-functional parameters for interaction.	Contract
Representation	Represents a perceptible form of the information carried by a business object.	Representation
Product	Represents a coherent collection of services and/or passive structure elements, accompanied by a contract/set of agreements, which is offered as a whole to (internal or external) customers.	Product

ArchiMate® 3.1 Specification		
Physical Elements		
Element	Definition	Notation
Equipment	Represents one or more physical machines, tools, or instruments that can create, use, store, move, or transform materials.	Equipment
Facility	Represents a physical structure or environment.	Facility
Distribution network	Represents a physical network used to transport materials or energy.	Distribution network
Material	Represents tangible physical matter or energy.	Material

ArchiMate® 3.1 Specification		
Application Layer		
Element	Definition	Notation
Application component	Represents an encapsulation of application functionality aligned to implementation structure, which is modular and replaceable.	Application component
Application collaboration	Represents an aggregate of two or more application internal active structure elements that work together to perform collective application behavior.	Application collaboration
Application interface	Represents a point of access where application services are made available to a user, another application component, or a node.	Application interface
Application function	Represents automated behavior that can be performed by an application component.	Application function
Application interaction	Represents a unit of collective application behavior performed by (a collaboration of) two or more application components.	Application interaction
Application process	Represents a sequence of application behaviors that achieves a specific result.	Application process
Application event	Represents an application state change.	Application event
Application service	Represents an explicitly defined exposed application behavior.	Application service
Data object	Represents data structured for automated processing.	Data object

ArchiMate® 3.1 Specification		
Implementation and Migration Elements		
Element	Definition	Notation
Work package	Represents a series of actions identified and designed to achieve specific results within specified time and resource constraints.	Work package
Deliverable	Represents a precisely-defined result of a work package.	Deliverable
Implementation event	Represents a state change related to implementation or migration.	Implementation event
Plateau	Represents a relatively stable state of the architecture that	

Evaluación - Visualización



Claridad Semiótica

- ✓ Todo concepto tiene un símbolo asociado, el cual se compone en su mayoría de un cuadrado con un ícono. Las figuras que no tienen íconos son todas diferentes por lo que también es fácil distinguirlas entre ellas.

Manejo de complejidad

- ✗ Se vuelve difícil entender estos diagramas sin conocer todos los íconos distintos.
- ✓ Archimate en su especificación propone múltiples puntos de vista estándar para comunicar únicamente lo necesario a los diferentes stakeholders de la empresa en diferentes contextos.

Ajuste cognitivo

- ✓ Gracias a las variaciones posibles de los componentes y la cantidad de variables visuales que se usan para diferenciar los componentes permite que quien usa el lenguaje lo adapte a sus necesidades de aprendizaje.

Discriminabilidad perceptiva

- ✓ Debido al uso de múltiples íconos, los distintos símbolos tienen la posibilidad de ser fácilmente distinguibles.
- ✗ En símbolos como el del nodo y otros símbolos como el dispositivo, o el del software del sistema, puede haber problemas de interpretación, ya que es la misma caja.

Expresividad visual

- ✓ El lenguaje hace uso de una gran cantidad de variaciones de forma, color y de relaciones de contención y con flechas.

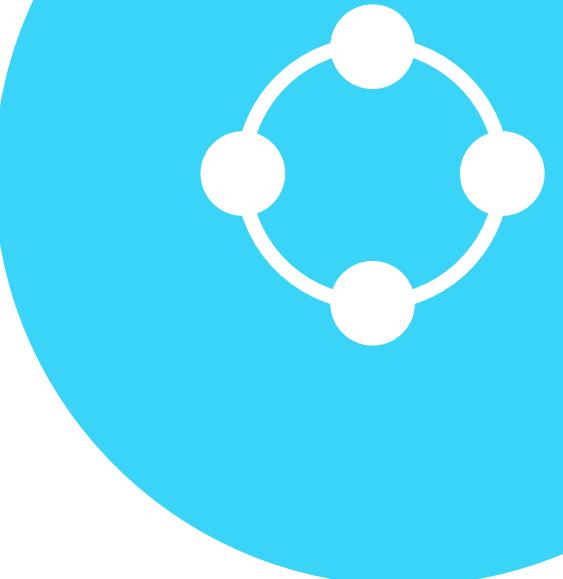
Transparencia Semántica

- ✓ El uso de íconos permite diferenciar los diferentes componentes y sus conexiones también permiten claridad. Además, la posibilidad de cambiar colores por una letra representativa del grupo de elementos o usarlos juntos es de gran utilidad.
- ✗ Si se hace uso únicamente de los íconos, sin su contraparte textual, todos pueden interpretarse de una forma diferente.

Codificación dual

- ✓ Funciona perfectamente al utilizar las cajas definidas por el lenguaje
- ✗ La codificación dual no es requerida por el lenguaje, se pueden usar únicamente los símbolos. Esto genera serios problemas de interpretación y entendimiento de los diagramas. Cuando va de la mano de pequeños textos descriptivos, es más fácil entender los diagramas y sus conexiones.

Evaluación - Colaboración



Sensación de terminado

- ✓ Gracias al uso de cajas y conexiones separadas, los diagramas de Archimate parecen haber sido terminados por completo, pero también dejan espacio para la intervención e inclusión de nuevas instancias.

Generación de insights

- ✓ Gracias al uso de múltiples formas de representación de relaciones, y al espacio entre componentes, puede ser fácil encontrar insights de la arquitectura definida e incluso encontrar conexiones nuevas entre componentes

Impacto visual

- ✓ Gracias al uso de múltiples colores, cajas con distintas características e íconos, los diagramas llaman la atención de quién los ve.

Modificabilidad

- ✓ Gracias al espacio entre componentes se ve la posibilidad de moverlos con facilidad y agregar cosas nuevas. Además la división por capas de los componentes en la creación del diagrama (capas horizontales) hace que sea más fácil saber en dónde agregar el contenido nuevo.

Concentración dirigida

- ✗ Aunque tenga un gran impacto visual y llame la atención, la concentración del receptor se puede perder si hay demasiados elementos en un solo diagrama.
- ✓ La disposición de elementos por capas en los diagramas facilita ver cada capa como un todo y por ende facilitar ver las conexiones dentro y fuera de estas, por lo que la concentración se puede mantener.

Manejo del discurso

- ✓ Gracias a las distintas variaciones que tiene el lenguaje, usando texto, íconos y algunas etiquetas para distinguir el grupo al que pertenecen los elementos, es fácil guiar una conversación de abajo hacia arriba de los elementos que se necesiten para la comunicación a los distintos stakeholders. Además, la guía presentada por Archimate para puntos de vista puede ayudar a un novato a modificar su diagrama resultado con respecto al proceso y/o stakeholders en cuestión.



Evaluación - Arquitectura

Orden

- ✓ El lenguaje y el uso de múltiples símbolos, colores y cajas, dan como resultado visualizaciones organizadas
- ✗ Se debe evitar el cruce entre flechas y mantener suficiente espacio entre las instancias para que no hay más confusión.

Consideración de puntos de vista

- ✓ La definición de posibles puntos de vista a partir de los diagramas creados es posible y se puede facilitar gracias a las guías que presenta Archimate para filtrar los diagramas de acuerdo al propósito que tengan.

Escalabilidad

- ✗ Al contar con un ícono, texto, una etiqueta y un color, el lenguaje de Archimate no escala muy bien. El nivel de complejidad del diagrama resulta aumenta considerablemente con la adición de nuevas instancias.

Claridad de las relaciones

- ✓ El uso de flechas y relaciones de contención permite que las relaciones se entiendan con facilidad, agrupando instancias y generando conexiones entre ellas.
- ✗ Hay demasiadas relaciones diferentes lo cual puede generar problemas de interpretación de las diferentes formas posibles para las flechas.

Nivel de abstracción

- ✓ El nivel de abstracción es bueno, ya que se usan distintos rasgos gráficos junto con íconos representativos que facilitan el entendimiento del símbolo.

Consistencia

- ✓ El lenguaje utilizado en los textos que acompañan a los símbolos es consistente con los elementos del negocio y los conceptos relacionados. Además, define un estándar de diseño y también permite ciertas variaciones que se pueden adaptar mejor al negocio.

Diferenciación de rels., conc. y atr.

- ✓ El lenguaje hace una distinción correcta de estos tres conceptos y genera los símbolos necesarios para definirlos.

Conclusiones

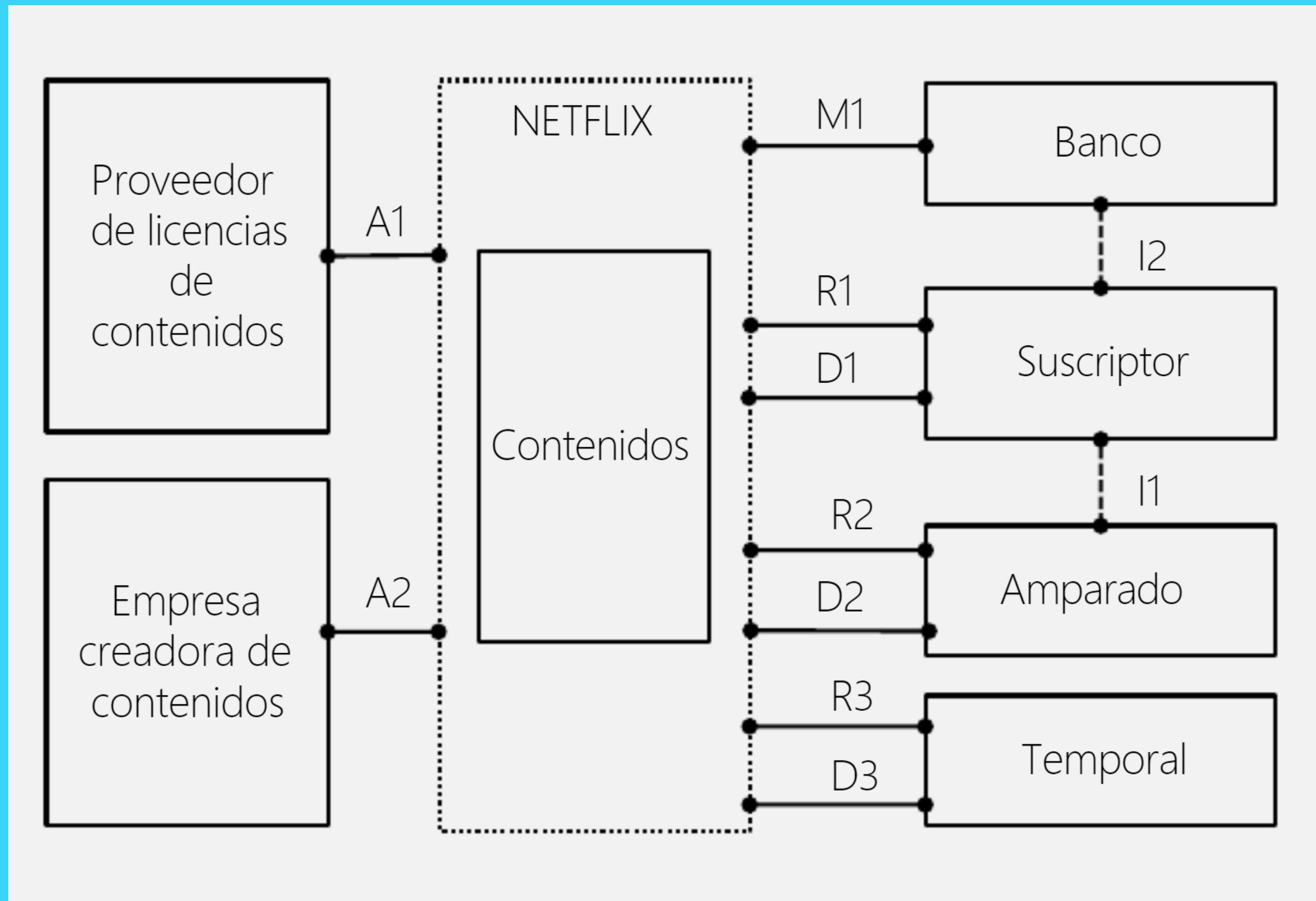
Archimate: el amigo de los stakeholders

Archimate se presenta como un lenguaje con el cuál se pueden comunicar temas de arquitectura de negocio a los stakeholders involucrados. De por sí, en su especificación el lenguaje se concibe de esta necesidad latente de comunicación en el contexto en el que es utilizado. El lenguaje tiene mecanismos que permiten modificar la forma en que es presentado y reemplazar o complementar algunas variables visuales con otras para adaptarse al contexto en el que serán utilizadas las visualizaciones creadas.

Esta versatilidad de los componentes, permite que sea fácil de adoptar por parte de personas nuevas a la sintaxis del lenguaje, ya que pueden tomar la versión de visualización de cada componente que les parezca más clara y sencilla de manejar. Por otro lado, debido a que el diagrama se organiza de abajo hacia arriba con respecto a las capas de las que hace parte cada componente, es mucho más fácil navegar y modificar el diagrama por lo que puede ayudar en procesos de toma de decisiones, generación de insights, entre otros.

En el contexto de arquitectura, suma varios puntos positivos por la posibilidad de generación de diferentes puntos de vista. Sin embargo, si se usan todas las representaciones posibles de los simbolos (la caja, con el texto, con el icono, con la etiqueta, con el color) el lenguaje se vuelve poco escalable.

Artefacto BAC-07

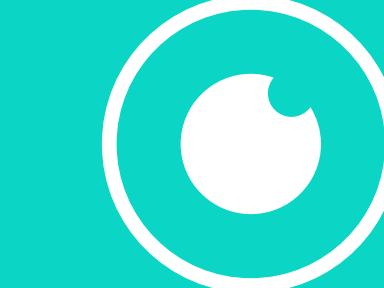


Lenguaje visual	
Concepto	Símbolo
Componente Elemento que participa en el negocio, que contiene info., dinero o valor y se comunica con otros componentes para intercambiar estos elementos.	Nombre Nombre
Canal Conectan componentes, describen el intercambio de info.	Componente A — Nombre canal — Componente B Componente A — I-Nombre canal — Componente B

Notación tipos:
R - Relacionamiento
A - Aprovisionamiento
D - Distribución
T - Transformación
M - Monetización
I - Indirecto

La imagen es el ejemplo que tiene AWS en el inicio de su paquete de recursos para diagramas de arquitectura AWS.
Tomada de <https://aws.amazon.com/architecture/icons/>

Evaluación - Visualización



Claridad Semiótica

- ✓ Todo concepto tiene un símbolo asociado, y cada uno tiene sus propias variaciones.

Discriminabilidad perceptiva

- ✓ Debido al uso de múltiples íconos, los distintos símbolos tienen la posibilidad de ser fácilmente distinguibles. Además, el uso de líneas continuas y discontinuas es una modificación visual fácil de percibir.

Manejo de complejidad

- ✓ Debido a la reducida cantidad de símbolos, la complejidad de estos diagramas es pequeña. Esto permite que los diagramas puedan crecer sin generar mayores efectos sobre la comprensión de los mismos.

Expresividad visual

- ✓ El lenguaje hace uso de variaciones de forma.
- ✗ Se podría utilizar otras variables visuales que permitan, por ejemplo, categorizar los componentes dependiendo del tipo de componente que es (Información, dinero o valor).

Ajuste cognitivo

- ✓ El uso de figuras básicas, como el rectángulo, el círculo y las líneas, junto con la mínima cantidad de símbolos hace que sea sencillo entender el lenguaje y que sea de fácil uso para una persona que lo conoce por primera vez. Este lenguaje es apto para uso de novatos y expertos.

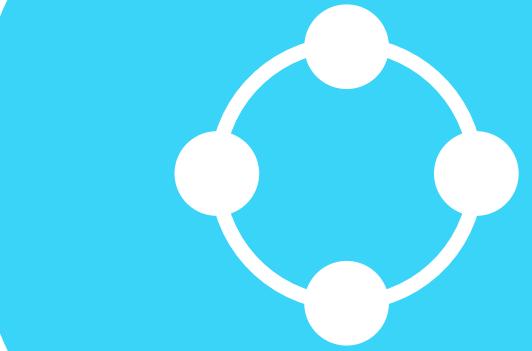
Transparencia Semántica

- ✓ Aunque los símbolos representan una abstracción alta, realmente es muy fácil entender cuál representa cada uno de los conceptos, gracias a que es un lenguaje simple y concreto.

Codificación dual

- ✓ El BAC-07 tiene un fuerte componente de codificación dual, ya que la mayoría de sus símbolos van acompañados de algún texto que facilita la comprensión de los diagramas.

Evaluación - Colaboración



Sensación de terminado

- ✓ Los diagramas creados con el lenguaje del BAC-07 dan una sensación de terminado suficiente para poder utilizarse para plasmar el diagrama final, pero también se ven abiertos a edición debido a su estructura de cajas y relaciones, junto con el aire que hay entre las instancias

Impacto visual

- ✗ Debido a que no se utilizan muchas variables visuales, los diagramas con BAC-07 en blanco y negro y sin rellenos no llaman mucho la atención de quien los ve.

Generación de insights

- ✓ Gracias al uso de múltiples formas de representación de relaciones, y al espacio entre componentes, puede ser fácil encontrar insights del modelo definido, que pueden ir desde relaciones nuevas o sobrantes hasta componentes nuevos o innecesarios.

Modificabilidad

- ✓ Gracias al espacio entre componentes, y lo sencillos que los símbolos son, se ve la posibilidad de moverlos con facilidad y agregar cosas nuevas.

Concentración dirigida

- ✓ El BAC-07 permite mantener la concentración del receptor, debido a que se basa en figuras sencillas que se pueden entender con facilidad y que se conectan con relaciones que también son de baja complejidad.

Manejo del discurso

- ✓ Gracias a la facilidad que otorga el BAC-07 para mantener la concentración y la facilidad de conocer y entender los distintos símbolos del lenguaje, manejar el discurso y guiar una conversación con un diagrama que lo usa es sencillo.

Evaluación - Arquitectura



Orden

- ✓ El lenguaje y sus símbolos basados en líneas y rectángulos permite que los diagramas resultado tengan un orden.
- ✗ Se debe evitar el cruce entre flechas y mantener suficiente espacio entre las instancias para que no haya confusión.

Consideración de puntos de vista

- ✓ Debido a su estructura basada en componentes y relaciones, es sencillo crear puntos de vista omitiendo algunos de ellos. Además, si en algún momento fuese necesario hacerlo, se podría separar la parte interna y externa del negocio (separadas por la frontera) en puntos de vista distintos. Además, con el BAC-07 se definen otros artefactos que modularizan la vista general del detalle.

Escalabilidad

- ✓ Debido a la reducida cantidad de símbolos, la complejidad de estos diagramas es pequeña. Esto permite que los diagramas puedan crecer sin generar mayores efectos sobre la comprensión de los mismos.

Claridad de las relaciones

- ✓ El uso de flechas y la clasificación de estas en diferentes tipos, diferenciados por algunas variables visuales, permite que sea claro cómo se relacionan las clases y/o los componentes.

Nivel de abstracción

- ✓ El nivel de abstracción es alto, pero sencillo de interpretar al conocer la notación del lenguaje.

Consistencia

- ✓ El BAC-07 permite que los términos utilizados para los componentes sean consistentes con el negocio en cuestión. Además define ejemplos que muestran el uso de un estilo gráfico consistente en sus diferentes diagramas.

Diferenciación de rels., conc. y atr.

- ✓ El lenguaje hace una distinción correcta de estos tres conceptos y genera los símbolos necesarios para definirlos.

Conclusiones

El BAC-07: Sencillo, efectivo y escalable

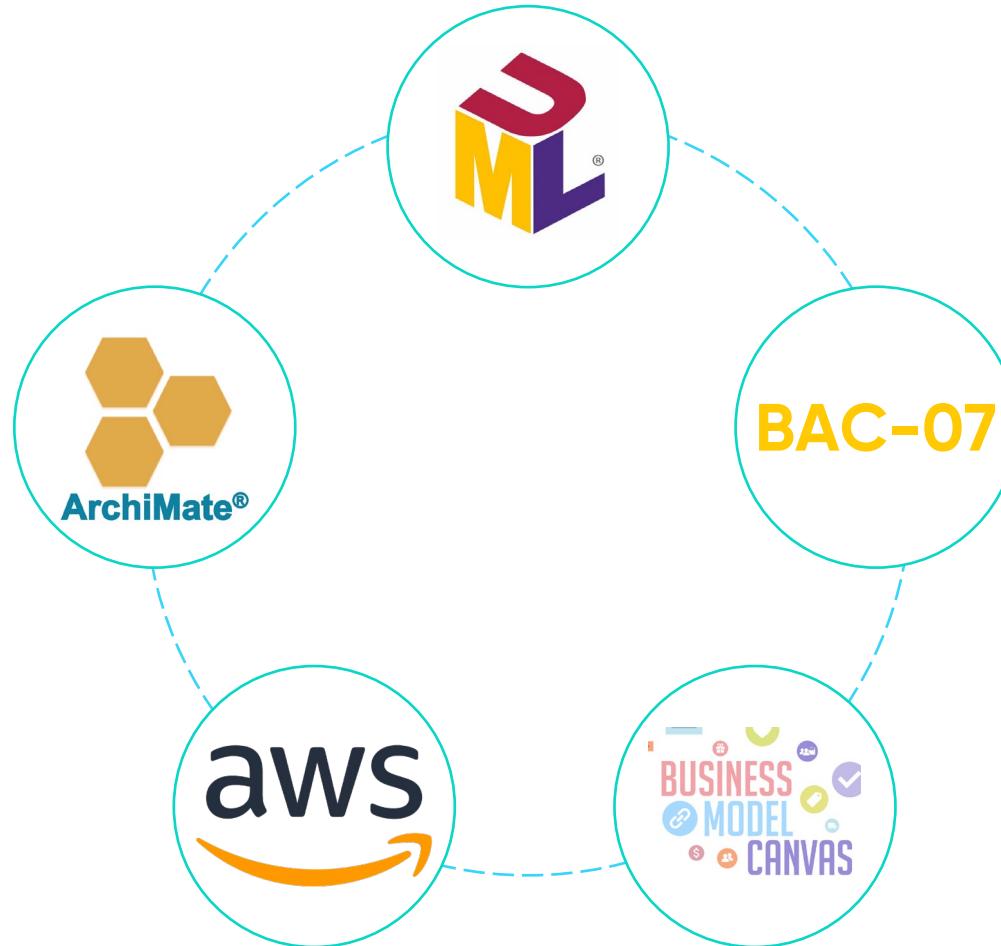
El lenguaje creado para el artefacto BAC-07 es bastante sencillo. Esto permite que se haga una abstracción clara, sencilla y fácil de un negocio, que además se puede utilizar para explicar el negocio a otros stakeholders interesados. Por otro lado, tiene la capacidad de ser fácilmente modificado y recibir nuevos componentes o relaciones, lo que permite ser usado en procesos de definición de un negocio nuevo o de innovación del modelo de negocio.

Al igual que en UML, este lenguaje tiene una curva de aprendizaje bastante rápida, que permite que personas que no conocen el lenguaje consigan un mayor nivel de experticia de su uso en un menor tiempo. Sin embargo, tiene una ventaja sobre UML: es más sencillo. Debido a que atributos o detalles más específicos se separan en otros artefactos y tablas, el BAC-07 se mantiene sencillo, únicamente con los nombres. Además, tiene una única relación que cambia dependiendo del canal que represente, el cuál está definido en las tablas que acompañan al artefacto.

Por último, el BAC-07 se presenta como un lenguaje escalable que puede acompañar el proceso de crecimiento del nuevo modelo, ya que es fácil incluir nuevos componentes y relaciones.

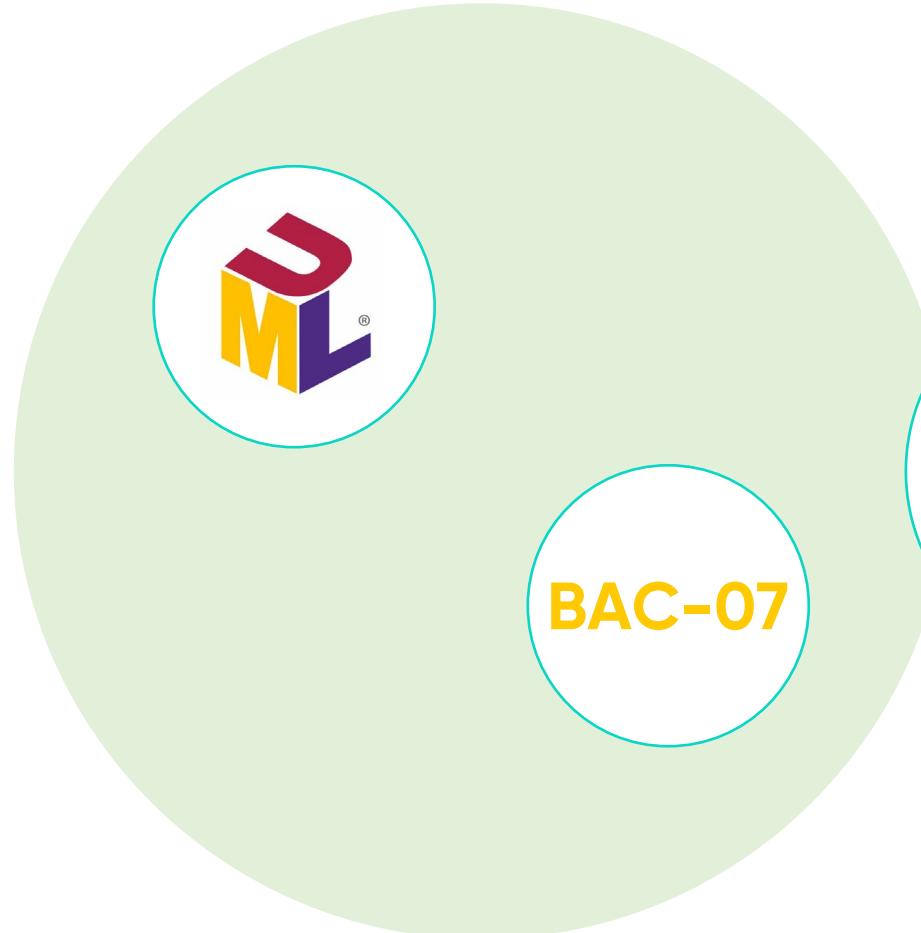
Resultados

Evaluación de 5 lenguajes populares en el contexto de AE

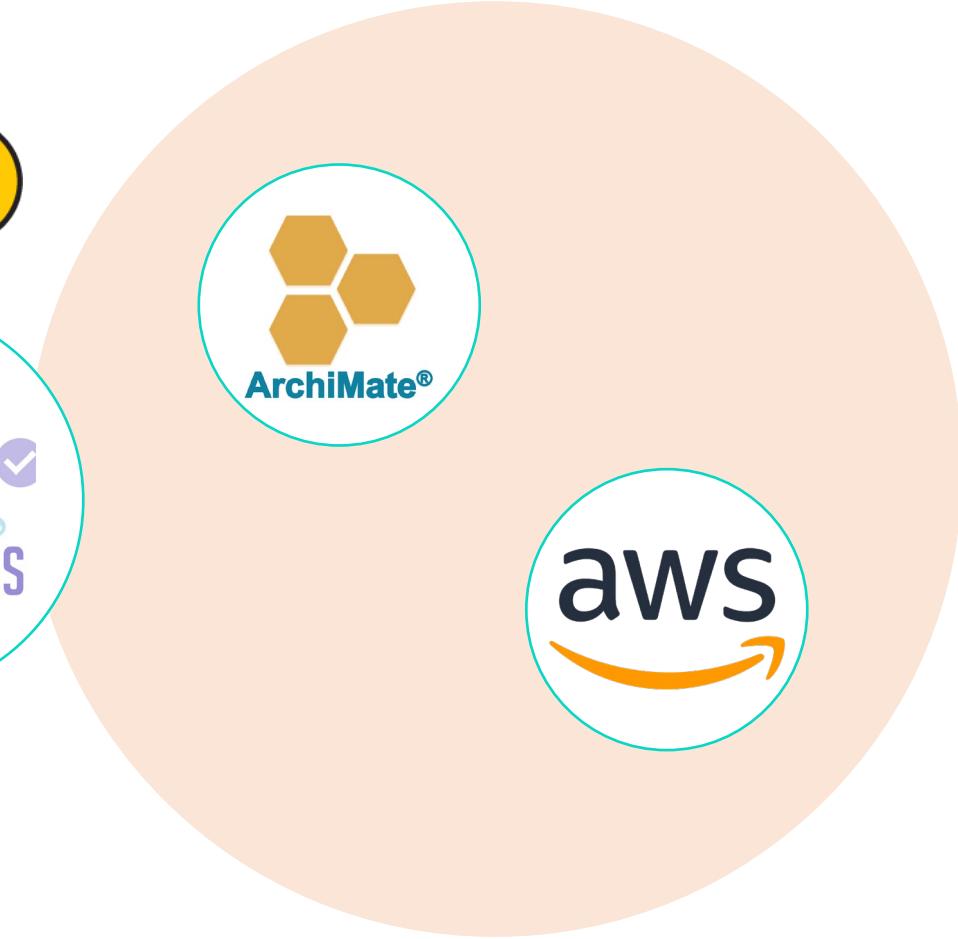


Resultados

Evaluación de 5 lenguajes populares en el contexto de AE



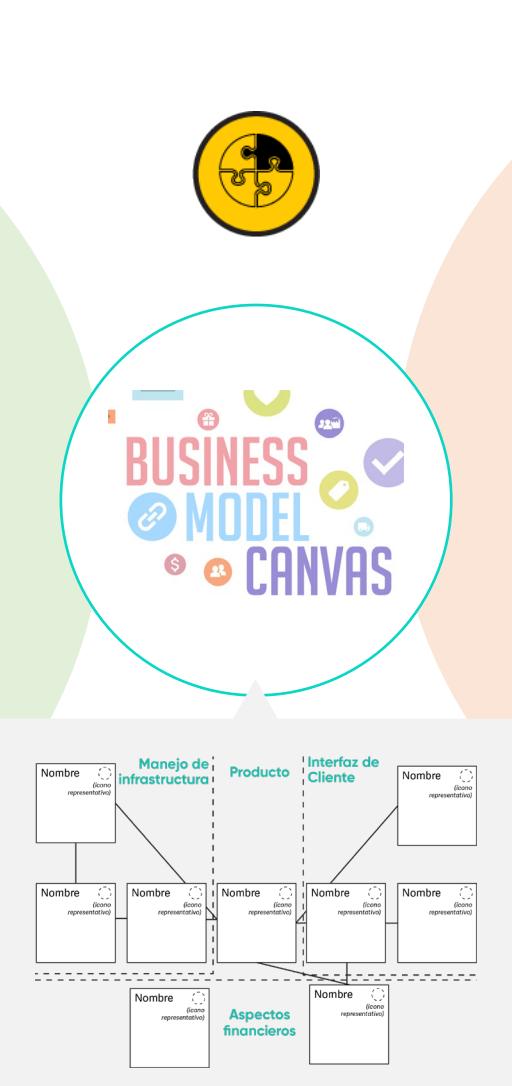
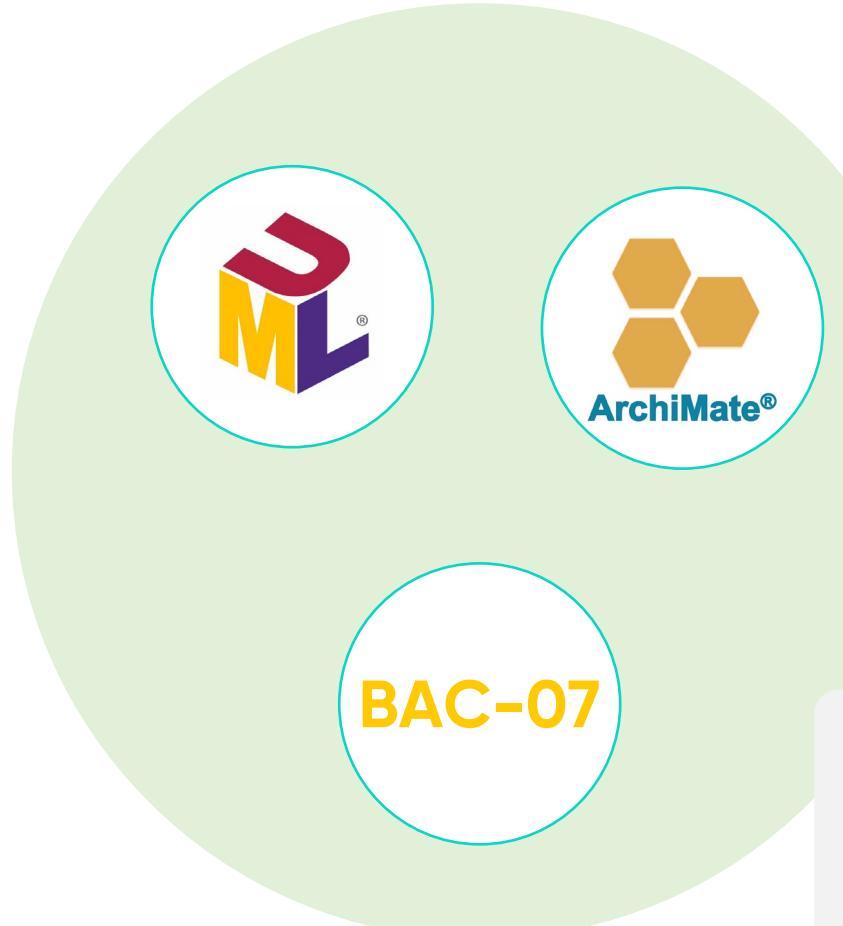
Propósito general



Dominio específico

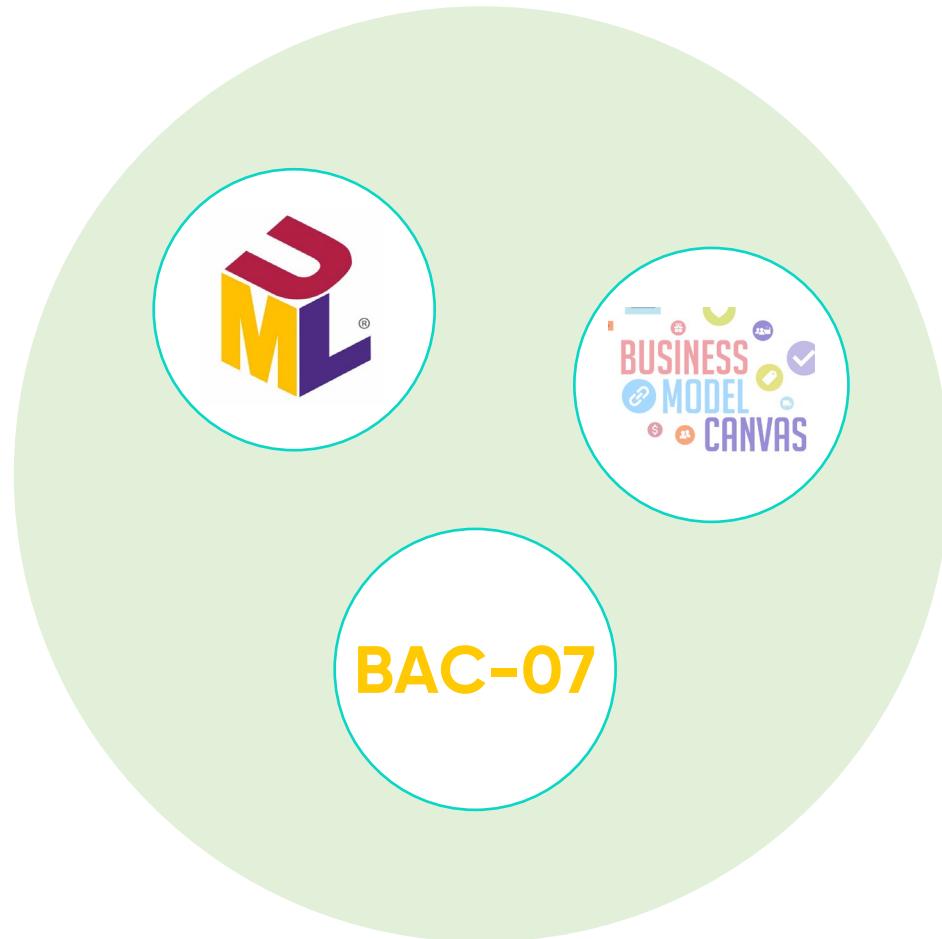
Resultados

Evaluación de 5 lenguajes populares en el contexto de AE

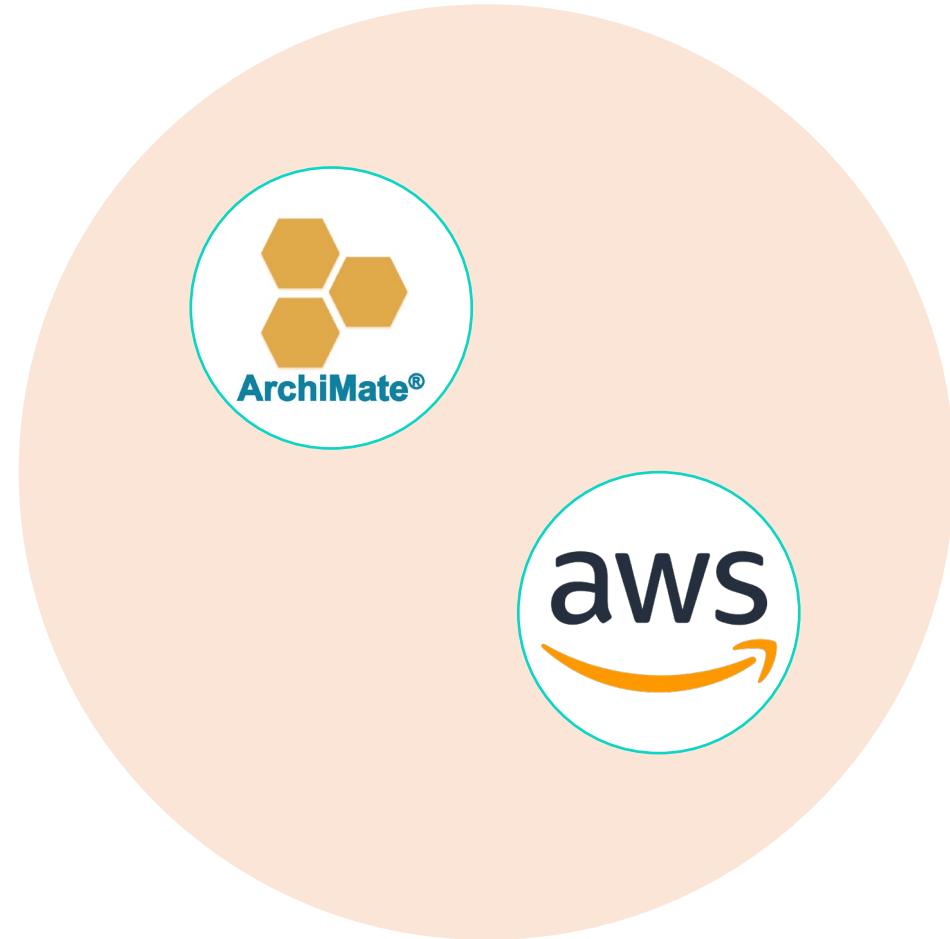


Resultados

Evaluación de 5 lenguajes populares en el contexto de AE



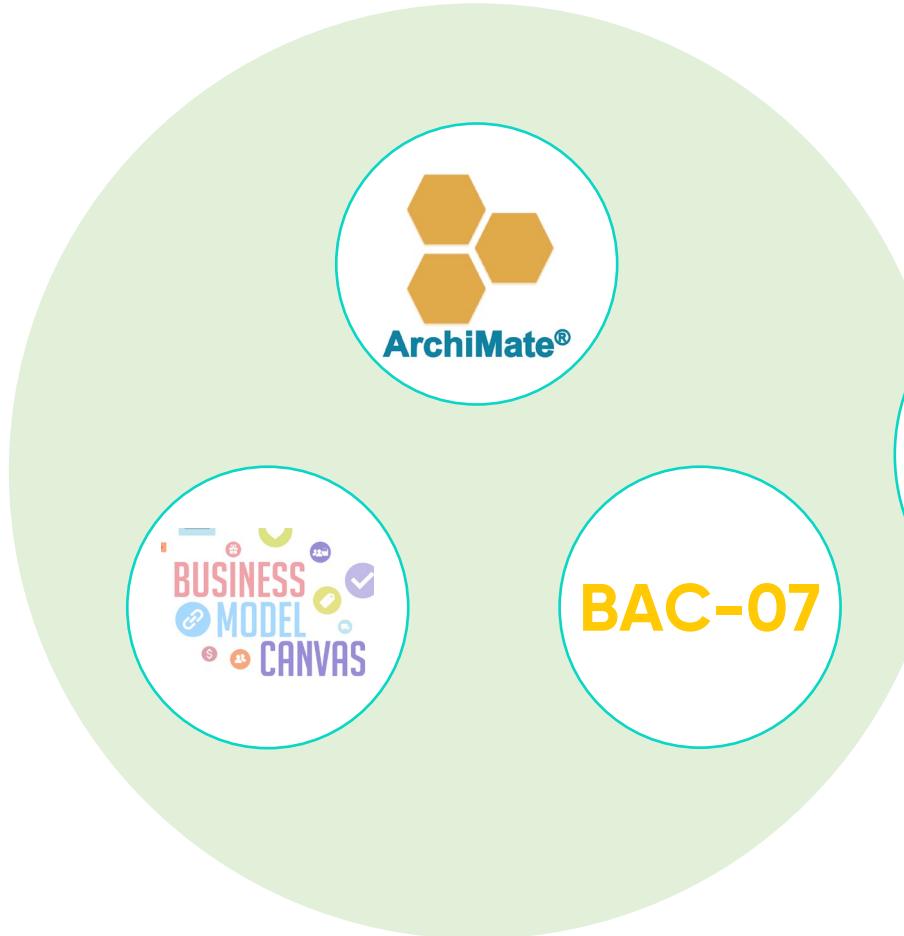
Curva de aprendizaje rápida



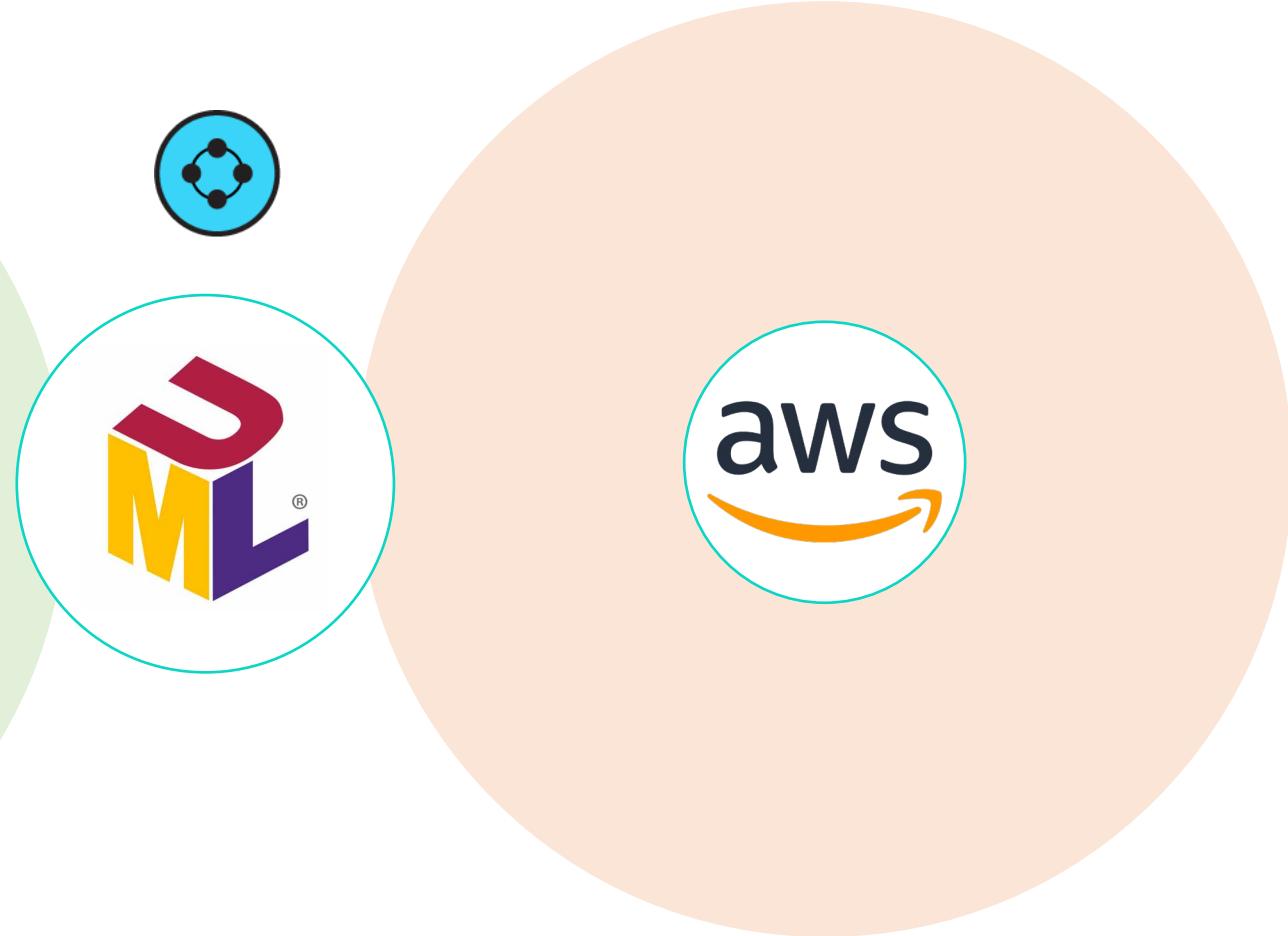
Curva de aprendizaje lenta

Resultados

Evaluación de 5 lenguajes populares en el contexto de AE



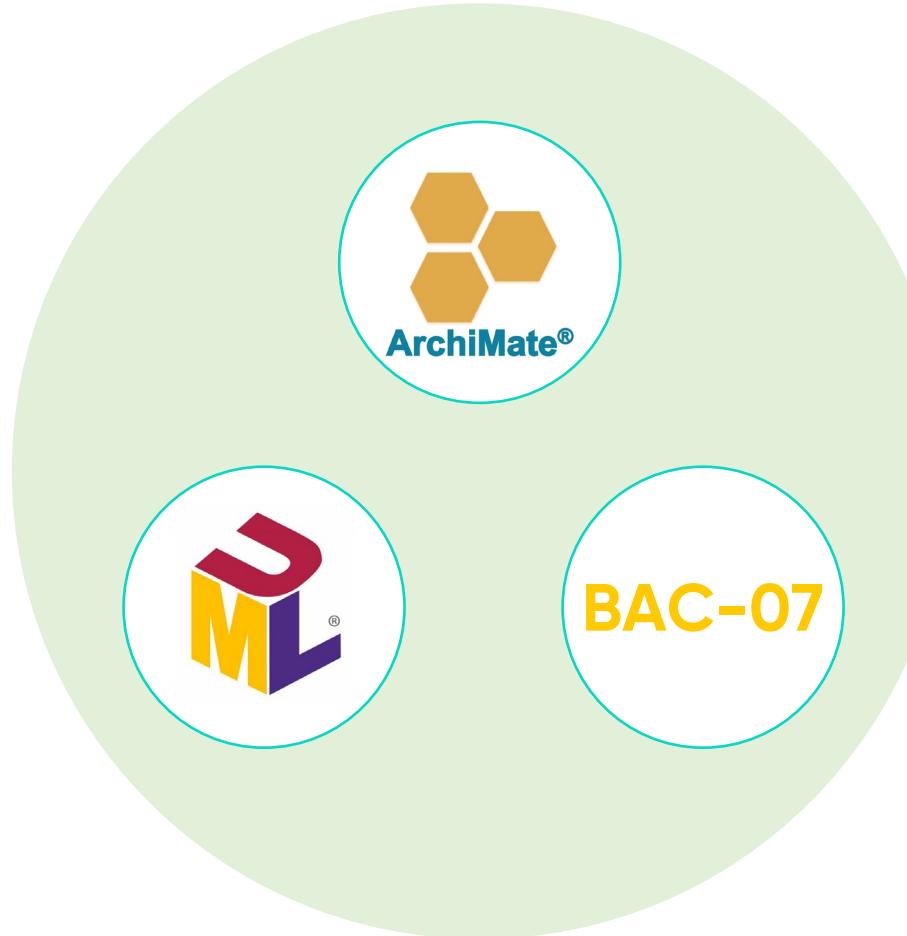
Apoya comunicación
con stakeholders.



Útil en áreas específicas

Resultados

Evaluación de 5 lenguajes populares en el contexto de AE



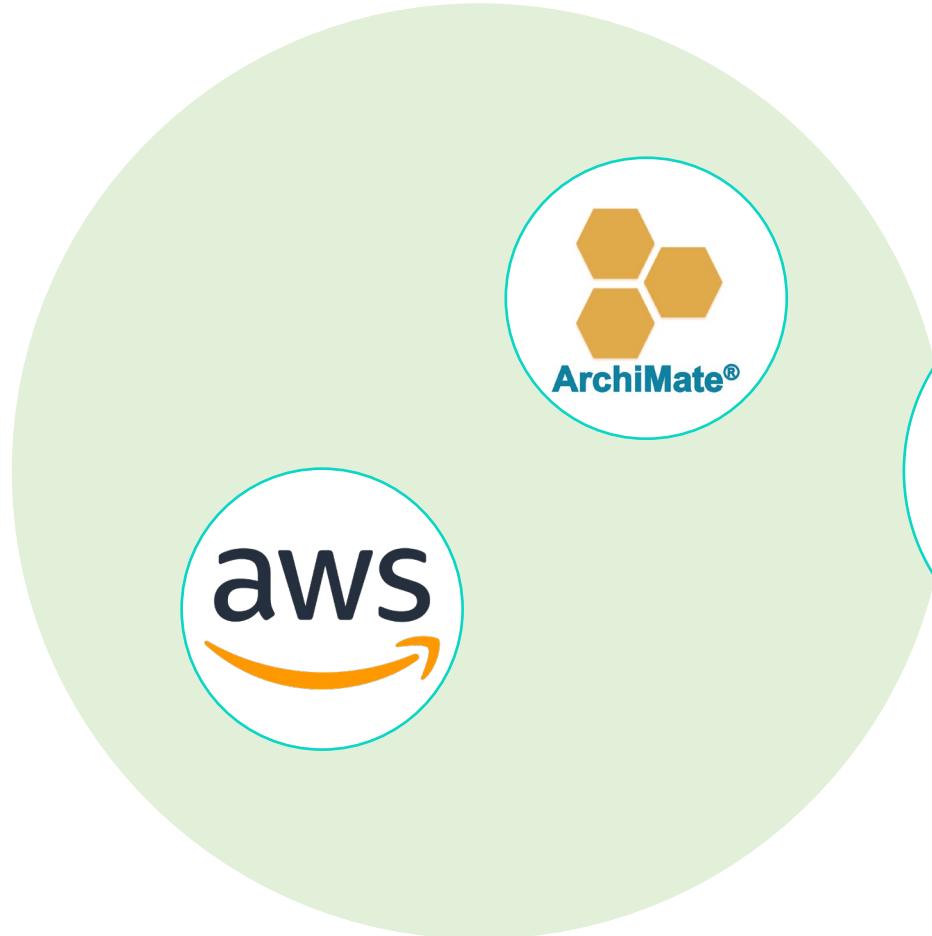
Apoya trabajo colaborativo



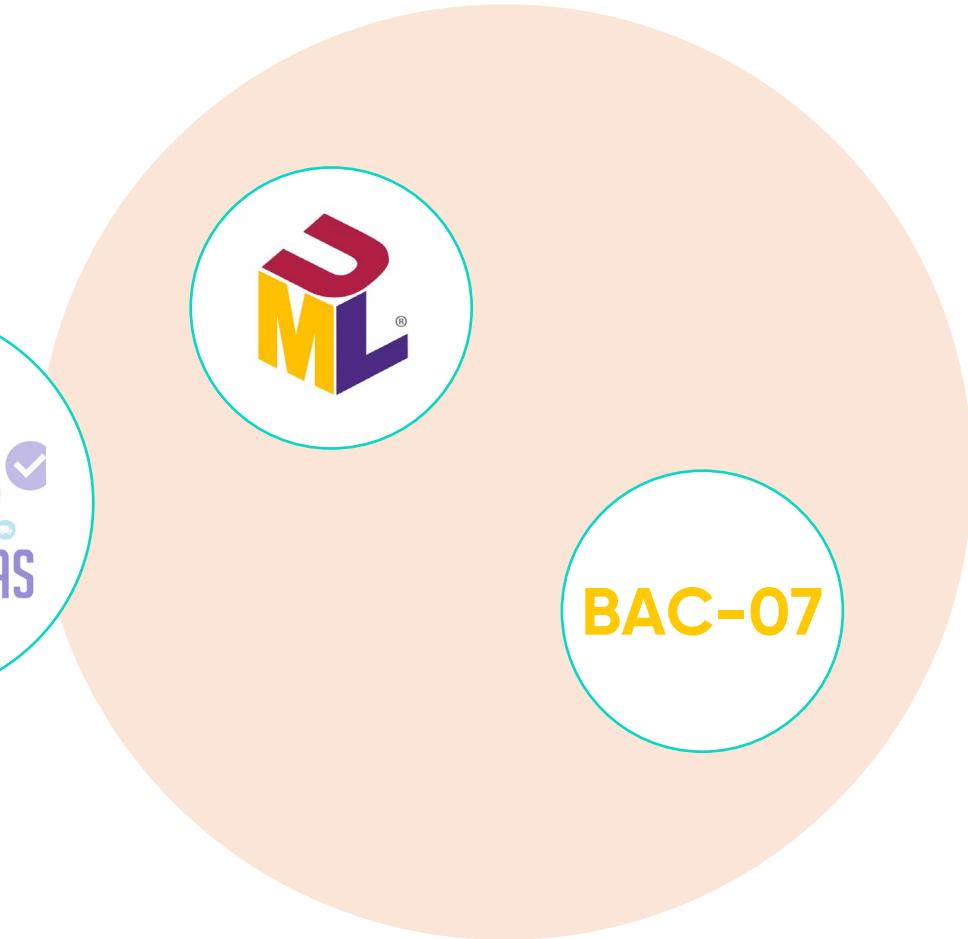
No se ven muchas
posibilidades de modificación

Resultados

Evaluación de 5 lenguajes populares en el contexto de AE



Capta la atención fácilmente



Poco expresivo