# Modelado guiado por arquitecturas de software para un sistema de control de tráfico aéreo en territorio Argentino

Gabriela Vilanova[1], Silvia Rivadeneira Molina[2], Diana Cruz[2]

Instituto de Tecnología Aplicada (ITA) Dpto Cs Exactas y Naturales
[1]Unidad Académica Caleta Olivia
[2]Unidad Académica Río Turbio
Universidad Nacional de la Patagonia Austral
Acceso Norte Ruta 3 Cp 9011 Caleta Olivia,Santa Cruz, 0297 4854888
e-mails vilanova@uolsinectis.com.ar, sgrivadeneira@yahoo.com.ar, dianalcruz@gmail.com

## Resumen

El control del tráfico aéreo (ATC) es una de las aplicaciones de software más exigentes. Es una aplicación en tiempo real, indispensable para la seguridad nacional. Vidas humanas se pueden perder si el sistema no funciona correctamente o bien deja de funcionar por un lapso de tiempo, y es altamente distribuída, lo que requiere decenas de controladores trabajando en cooperación para orientar aeronaves a través del sistema de rutas aéreas.

El objetivo del presente trabajo es aplicar un proceso de diseño guiado por la arquitectura para un sistema de gran escala y de gran criticidad como lo es el software de asistencia al controlador aéreo de un Centro de control de área.

**Palabras clave:** sistemas críticos, atributos de calidad, arquitectura de software.

## Contexto

El proyecto de investigación PI 29B176 UNPA Modelado y diseño de software, un enfoque arquitectural, participa del programa de incentivos, se ha iniciado en 2014, es de tipo I, se basa en el proyecto finalizado 29B134 (2012-2014) es de apoyo a las carreras de pre grado, grado y posgrado del área informática que se dictan en UNPA en Unidades Académicas Caleta Olivia, Río turbio y río Gallegos. Dichas carreras son: Licenciatura en sistemas, Ingeniería, Analista de Sistemas y Tecnicaturas en Diseño Web y en Redes. Maestría en sistemas. Los integrantes son docentes, investigadores categorizados de dos unidades Académicas de UNPA a saber Unidad académica Río Turbio (UART) y Unidad académica Caleta Olivia. (UACO) hay un integrante docente con posgrado de la Universidad de Magallanes, docente visitante de UART.

# Introducción

La arquitectura de software juega un papel fundamental en el desarrollo a gran escala de software de calidad. Los ingenieros de software generalmente se refieren a la arquitectura de software como "la organización a nivel de un sistema de software integrado de los componentes, las relaciones entre ellos y las restricciones".[1][2]

Además, la arquitectura de un sistema es responsable de la captura de las abstracciones de la arquitectura esencial para garantizar un conjunto relevante de los factores de calidad. Según la definición de la arquitectura de software que figura en [1], diseño basado en la arquitectura se refiere principalmente a las siguientes cuestiones:

1. La comunicación entre las partes interesadas en el sistema o stakeholders, es decir usuarios u operadores, clientes, diseñadores, programadores etc. La arquitectura de software representa una abstracción común de un sistema pudiéndose utilizar por todos los interesados como base para el entendimiento mutuo, la negociación, el consenso y la comunicación.

2. Principios de las decisiones de diseño, es decir aplicación de heurísticas y patrones o estilos de diseños que se irán ajustando a los requerimientos de calidad deseados.

3. Abstracción transferible de un sistema. La arquitectura de software constituye el modelo intelectualmente comprensible de un sistema, de cómo está estructurado y cómo sus elementos trabajan juntos. Este modelo, es decir las distintas vistas se puede aplicar a otros sistemas que exhiban atributos de calidad y requisitos funciones similares pudiéndose promover el reúso a gran escala.

Varios estilos arquitectónicos y patrones recurrentes han sido reconocidos y codificados por la comunidad de arquitectura de software. [2,3]. Hay, en consecuencia, cierto acuerdo sobre el conjunto de normas de diseño para identificar qué tipo de componentes y conectores pueden ser usados para componer un sistema dentro de cada estilo, junto con las limitaciones locales o globales sobre la forma en que esta composición se debe hacer. Algunos de estos patrones implican algún tipo de razonamiento para lograr los atributos de calidad [4].

### 1.1 Proceso de diseño guiado por la Arquitectura de Software.

El diseño de una arquitectura no es una actividad independiente, pero es un paso dentro del desarrollo y proceso evolutivo del producto. Un ciclo de vida típico consiste en cuatro procesos iterativos. El proceso externo se refiere a la evolución de los requerimientos de un producto durante su maduración. El próximo proceso refiere a desarrollo iterativo del producto. Por ejemplo, cuando se construye sistema, el cliente puede no saber especificar exactamente cuáles son los requerimientos que tendrá. Sin embargo, esto puede ser descubierto durante un proceso de desarrollo iterativo. Los dos procesos iterativos siguientes definen los métodos de diseño de arquitectura orientado a los atributos de calidad, la iteración interna donde la arquitectura de software es diseñada, evaluada y transformada por sus requerimientos de calidad, y una iteración

externa donde la selección de requerimientos es realizada.

El foco de la iteración interna es la evaluación y transformación de los atributos de calidad. Los métodos de diseño proveen soporte para un objetivo, el proceso de diseño racional, balanceando y optimizando especialmente los requerimientos de calidad. El método iterativo evalúa el grado en que la arquitectura soporta cada requerimiento de calidad y mejora la arquitectura usando transformaciones hasta que todos los requerimientos de calidad hayan sido cumplidos.

El método de diseño de la arquitectura de software no solo define los procesos asociados con el diseño de la arquitectura, sino además un número de artefactos que documentan la arquitectura y las decisiones de diseño que encabezan una arquitectura.

Estos artefactos son divididos en dos amplias categorías, la especificación de requerimientos y la arquitectura de software. La especificación de requerimientos a su vez se divide en requerimientos funcionales y requerimientos de calidad. La arquitectura de software consiste en cuatro artefactos, el contexto del sistema, los arquetipos, la estructura arquitectural y las decisiones de diseño.

El contexto del sistema define las interfaces del sistema de software con su contexto. Los arquetipos representan las abstracciones centrales que posee el sistema cuando es construído. La estructura representa la descomposición de la arquitectura en sus componentes centrales y sus relaciones entre esos componentes. Existen tres tipos de decisiones de diseño, transformaciones, restricciones y reglas.

### 1.2 Modelado basado en Arquitectura de Software. Sistema de control de tráfico aéreo.

El Centro de control de área*:* (ACC) es la dependencia encargada del control de Área y parte del control de Aproximación. La función principal desempeñada por el ACC es brindar seguridad, ordenamiento y rapidez al tránsito aéreo en su jurisdicción. El ACC tiene jurisdicción sobre una FIR (Región de Información de Vuelo), controla las aeronaves desde que éstas liberan la zona controlada por las TWR hasta la siguiente zona de control o hasta que ingresa a la FIR controlada por otro ACC. Los vuelos están siempre bajo el control de una dependencia determinada, a excepción de aquellos que se realizan fuera de los espacios aéreos controlados, tales como vuelo deportivo o fumigadores. Por ejemplo el ACC Comodoro Rivadavia (ACC CR) comprende toda la zona sur del país, desde Bahía Blanca hasta Usuahia e islas Malvinas (Ver Figura 1).

Figura 1. Carta de navegación en rutas aéreas (Argentina Sur)

El ACC CR tiene comunicación vía radio con todos los aeropuertos ubicados bajo su área de control y brinda los permisos de control de tránsito aéreo para todas las aeronaves que por allí circulan, incluso los cruces del territorio argentino en tránsito hacia otros países. Como se mencionó al principio, el ACC realiza parte del control de aproximación, que es cuando las aeronaves abandonan el nivel de crucero para descender hacia el aeropuerto de destino. Las tareas del ACC son múltiples y complejas, por lo que cuenta con mayor cantidad de personal que una TWR. Las distintas divisiones del espacio aéreo realizadas dentro del ACC se denominan *Circuitos*, y cada uno de éstos es controlado por un controlador habilitado y un ayudante. No se debe confundir ACC con TWR, ésta se encarga de los arribos y despegues, en cambio el ACC controla principalmente la ruta.

PNL (Plan de Vuelo): es la Información especificada que, respecto a un vuelo proyectado, o a parte de un vuelo de una aeronave, se somete a las dependencias de los servicios de tránsito aéreo, concretamente, el PLN contiene información acerca de un vuelo determinado indicando origen, destino, nivel de vuelo en ruta, tiempo de vuelo, aeropuertos de alternativa, autonomía, etc; es presentado a la autoridad aeronáutica por el piloto o su representante (despachante de aeronaves) en la oficina ARO-AIS. Cuando un vuelo progresa desde su aeropuerto de salida a su aeropuerto de llegada, se ocupan varias entidades ATC que guía de forma segura a través de cada porción de rutas aéreas (y las instalaciones de tierra Figura 3.) que está utilizando.

Fig. 2 Volar desde el punto A al punto B en el sistema de control del tráfico aéreo

En resumen, el sistema del ATC debe hacer lo siguiente:

-Convertir las hojas de ruta y los ingresos que realice el controlador sobre las comunicaciones con los pilotos y los controladores de aeropuertos dentro de su jurisdicción, Cada consola elige los informes que necesita para mostrar, cualquier consola es capaz de mostrar cualquier zona en la pantalla como si en tiempo real tuviéramos un radar disponible (Simulación Dinámica).

-Manejar alertas de conflicto, el sistema deberá transmitir al controlador las potenciales colisiones de aviones o cualquier eventualidad que pueda resultar en riesgo para las aeronaves.

-Proveer capacidad de respaldo de información si un evento de falla sucede.

-Proporcionar la posibilidad de grabación de las comunicaciones para su posterior reproducción.

-Estar disponible en todo momento

## Líneas de Investigación, Desarrollo e Innovación

Las distintas líneas de trabajo pretenden cubrir en la temática de modelado y diseño de sistemas críticos, complejos, áreas de interés de estudio tales como modelado de requerimientos funcionales y no funcionales desde punto de vista de arquitecturas de software, arquitecturas orientadas a servicios aplicando procesos ágiles, de manera de disminuir el impacto de decisiones tomadas de diseño (patrones de grano fino y grueso) logrando niveles de calidad aceptables tanto en producto como en proceso de desarrollo del software.

## Resultados y Objetivos

Se ha definido el proceso arquitectural, como empezar a pensar una arquitectura, como lograr que en distintos niveles cada uno de los atributos de calidad sean satisfechos por la arquitectura lograda. El sistema de apoyo deberá tener una disponibilidad total y un alto rendimiento alto debido a su larga vida prevista, su gran tamaño y su importante función dentro de su campo de aplicación. Para ello la arquitectura propuesta debe utilizar una amplia gama de mecanismos de tolerancia a fallos, incluyendo la redundancia en hardware y software, y un proceso de detección de fallas por capas para que el fallo no se propague dentro del sistema.

## Formación de Recursos Humanos

El proyecto de investigación PI 29B176 se ha iniciado en 2014, es de tipo I, se basa en proyecto finalizado 29B134 (2012-2014) es de apoyo a las carreras de informática que se dictan en UNPA, Licenciatura en sistemas, Ingeniería, Analista de Sistemas y Tecnicaturas en Diseño Web y en Redes y Maestría en sistemas. Los integrantes son docentes, investigadores categorizados de dos unidades Académicas de UNPA a saber UART y UACO. La temática ha sido abordada como caso de estudio para trabajo final de cátedra en asignaturas de carreras analista de sistemas y optativa Arquitectura de software de Ingeniería en sistemas.

## Referencias

[1] Bass, P. Clements and R. Kazman. *Software Architecture in Practice.* Addison Wesley.

[2] D. Garlan and M. Shaw. An Introduction to software architecture. In *Advances in Software Engineering and Knowledge Engineering*, pages 1-39, Singapore, 1993. World Scientific Publishing Company.

[3] Bosch Jan Design and Use of Software Architectures. Addison Wesley.

[4] B. Boehm, P. Bose, E. Horowitz and M. J. Lee. Software requirements negotiation and renegotiation aids: A theory-W based spiral approach. In *Proc 17th International Conference on Software Engineering*, 1994.