1. Como se integra Python con otros lenguajes de programación?

R:

Podríamos comenzar por enfocarnos en los scripts, apis, librerias de Python que se pueden comunicar fácilmente con otras partes de una aplicación, usando una variedad de mecanismos de integración. Tal integración permite que Python pueda ser empleado como herramienta de personalización y extensión, el código Python puede invocar librerías C y C++, puede ser llamado desde programas C y C++, se puede integrar con códigos Java y con componentes .NET, puede comunicarse con DreamWorks como COM, puede servir de interface con dispositivos mediante puertos serie, y puede interactuar en red con interfaces como SOAP, XML-RPC, y CORBA. Por otro lado Python viene con una biblioteca de módulos estándar, y algunos módulos se integran en el intérprete; estos proveen acceso a operaciones que no son parte del núcleo del lenguaje pero que sin embargo están integrados, tanto por eficiencia como para proveer acceso al sistema operativo, como llamadas al sistema.

Por ejemplo [Cython](http://docs.cython.org/) es una librería que permite generar extensiones para Python escritas en un lenguaje muy parecido a Python, pero con la posibilidad de utilizar funciones o segmentos de código de C.

**from** distutils.core **import** setup

**from** distutils.extension **import** Extension

**from** Cython.Distutils **import** build\_ext

ext\_modules = [Extension("my\_module", ["my\_module.pyx"],libraries=[])]

setup(

name = 'my\_module',

cmdclass = {'build\_ext': build\_ext},

include\_dirs=[],

ext\_modules = ext\_modules

)

1. Patrones de arquitectura de software?
2. **Arquitectura en capas?**

Arquitectura por capas es una de las técnicas más comunes para dividir sistemas de software complicados ya que en términos de capas, se imaginan los principales subsistemas de software ubicados de la misma forma que las capas de un pastel, donde cada capa descansa sobre la inferior. En este esquema la capa más alta utiliza varios servicios definidos por la inferior, pero la última es inconsciente de la superior.

1. **Arquitectura orientada a eventos?**

Se basa en orquestar su comportamiento, se centra en torno a la producción, detección, consumo y respuestas ante “eventos”. Teniendo en cuenta que un evento es: cualquier ocurrencia identificable que tiene un significado para el hardware o el software del sistema, en otras palabras, cualquier cambio de estado significante para el sistema.

1. **Arquitectura de micro kernel?**

Se aplica a sistemas de software que tienen la necesidad de adaptar el sistema a requerimientos cambiantes. Este patrón separa la funcionalidad central mínima de la funcionalidad extendida y las partes específicas del cliente. El microkernel también sirve como un socket para comunicar estas partes y coordinar su colaboración.

1. **Arquitectura de micro servicios?**

Conocida como (Service Oriented Architecture), es un concepto de arquitectura de software donde el software consta de una composición de servicios, prestaciones y reglas, y son los requisitos del negocio los que dictaminan la manera en la que estas se ínter-relaciona.

1. **Arquitectura basada en espacios?**

Integra múltiples componentes enlazándolos y al mismo tiempo creando una especie de agrupación de alta disponibilidad ya integrada sobre una memoria cache.