|  |  |  |  |
| --- | --- | --- | --- |
| **Lenguaje** | **Características** | **Ventajas** | **Desventajas** |
| Java | \*Semi-interpretado, orientado a objetos, que corre sobre una máquina virtual.  \*Simple de aprender a usar, ya que es intuitivo y tiene sintaxis similar a la de C.  \*Al ser orientado a objetos alienta a utilizar buenas prácticas de programación, sin embargo, no nos limita a hacerlo  \*Es un lenguaje multiplataforma y al correr sobre una máquina virtual permite de forma relativamente sencilla portar las aplicaciones a otras arquitecturas.  \*Existen versiones de Java para la mayoría de las arquitecturas (Windows, Linux, Solaris, Mainframes). | \*Gratuito. Puede armarse toda la infraestructura de desarrollo con productos gratuitos:  Linux en los servidores.  Apache / Tomcat como servidor de aplicaciones.  Linux en la Pc de los programadores.  Eclipse o netbeans como IDE de desarrollo.  \*Además como es muy utilizado mundialmente hay mucha comunidad, encontramos librerías gratuitas, foros, tutoriales y ayudas varias.  \*Es una tecnología ampliamente probada. | \*Es Open source, por lo cual no existe soporte licenciado, por lo que puede generar desconfianza en las empresas.  \*Es un lenguaje de propósito general, que no está específicamente pensado para que un desarrollo sea rápido.  \*Se pueden lograr cosas muy personalizadas y es ideal para aplicaciones fuera de lo común, pero no viene listo para ser productivo y eficiente en forma inmediata.  \*Debemos armarnos un framework, recolectar varias apis, etc. Por lo cual contamos con un tiempo de startup para poder comenzar a trabajar. |
| .NET | \*.Net, no es un lenguaje de programación, es una solución entera, incluye un lenguaje IDE, editor visual, servidor web etc.  \*Está desarrollado para utilizar la solución completa.  \*Este paquete viene listo para empezar a desarrollar, lo que facilita la productividad. Se instala el paquete y ya tenemos todo configurado para comenzar a trabajar.  \*Se tiene una forma de encarar el proyecto por parte de Microsoft, que no tiene por qué ser la mejor ni la más óptima pero ya está hecha, probada y funciona. | \*Soporte licenciado  \*Comienzo de desarrollo inmediato.  \*Desarrollo más rápido que en otras plataformas.  \*Aplicaciones integradas en el Framework Microsoft.  \*Ampliamente probado. | \*Adaptabilidad a otros productos (por ej. Google Maps o Google Docs.)  \*La oferta de programadores capacitados está en crecimiento pero aun bastante más chica que la comunidad java.  \*Licenciamiento, el 90% de las herramientas utilizadas son pagas. |
| PHP | \*Es open source al igual que Java.  \*Simplicidad y facilidad de uso.  \*Para un diseñador web, PHP es el lenguaje más simple de aprender  \*Es un lenguaje 100% interpretado, no hay una compilación intermedia como en java.  \*El hecho de ser interpretado puro podría parecer que es lento, pero a diferencia de JavaScript, PHP fue pensado de entrada para interpretarse lo más rápido posible. | \*Es open source.  \*Simplicidad y facilidad de uso.  \*Comienzo de programación inmediato.  \*No se requieren deploys o compilaciones previas.  \*Su Application Server Apache es muy confiable.  \*Posee arquitectura de procesos en lugar de threads, de allí su confiabilidad. | \*Genera mucho procesamiento al Sistema Operativo.  \*El código muchas veces esta diseminado por todos los formularios.  \*Menor robustez.  \*Soporte limitado. |
| Arquitectura de Procesos  (Apache) | \*Hay un proceso principal que es el propio servidor  \*Por cada pedido web o request se crea un proceso para atenderlo. Este proceso atiende el pedido procesa su lógica, accede a la base de datos, genera una salida y luego muere  \*Su tiempo de ejecución suele ser muy corto, del orden de los milisegundos  \*En un servidor apache se crean y se mueren procesos constantemente  \*El principal problema de esto es que crear un nuevo proceso requiere intervención del sistema operativo y tiene un costo asociado  \*Hay que crear un montón de estructuras internas en el Sistema Operativo solo para que el proceso dure una fracción de segundo | | |
| Arquitectura de Threads  (Java-TomCat) | \*Surgen para liberar de trabajo al Sistema Operativo.  \*Un threads es como un proceso, pero más liviano.  \*Ejecuta dentro del área de memoria de un proceso ya existente.  \*Los threads comparten memoria.  \*Crear y matar hilos es mucho más liviano que matar procesos.  \*Solo hay un proceso en el Aplication Server, todos los procesos se atienden creando hilos de este proceso.  \*Si un thread no limpia la memoria utilizada sigue quedando ocupada, produciendo un memory leak.  \*La arquitectura de servidor de threads se adapta mucho mejor a la naturaleza de las aplicaciones Web, pero requiere una mejor programación. | | |
| SAP | \*SAP es un producto enlatado genérico que consiste en un modelo armado y probado.  \*Tiene varios módulos ya hechos, ya programados que cubren las necesidades más comunes de una gran empresa: Recursos humanos, Liquidación de sueldos, Contabilidad, Facturación, Ciclo de vida de un producto, Cadena de suministros. | \*SAP es un producto ya hecho y probado.  \*Es ampliamente utilizado por grandes empresas en todo el mundo.  \*No requiere tiempos de desarrollo previos.  \*Permite su parametrización y customización.  \*Ya viene con una forma de encarar la solución al problema.  \*Tiene un amplio soporte. | \*La customización que puede realizarse es limitada.  \*El costo del producto es muy alto.  \*No hay gran oferta de programadores ABAP, lo cual aumenta el costo de su customización.  \*Los tiempos de implementación del producto son altísimos. |

Interfaces de Usuario: Es fundamental en el desarrollo de software. Es la forma en que el sistema muestra la información. Es lo único que el usuario puede ver.

Interfaz gráfica Desktop (Java; .Net; C++)

Interfaz gráfica Web (Ajax, Tecnologías RIA)