Entrevista En Diferido 13: Rafa Ontivero

10 de Julio de 2019 a las 02:15

Hoy empezamos una nueva entrevista con Rafa Ontiveros, desarrollador de C++ y creador del podcast Leña al mono que es de goma.

Antes de todo, gracias por participar y apoyar este proyecto. La primera pregunta es una presentación por parte del entrevistado, para que los lectores te puedan conocer un poco.

Entrevista En Diferido: ¿Te podrías presentar en unas líneas?

Rafa: Si tuviera que hacerlo a nivel profesional, diría que soy un desarrollador de software con más de veinticinco años de experiencia en ensamblador (algo olvidado ya), C y C++, así como en C#, C++/CLI y tecnologías .NET desde el día en que salieron. Experiencia en desarrollo en tiempo real, programación de microprocesadores sin sistema operativo, Windows y algo de Linux así como amplio conocimiento de protocolos de comunicación. Algo de experiencia en Python y PHP.

Entre tu y yo, soy un jodío develópero al que le gusta tirar código más que a un gorrino la alberca y al que le encantan las situaciones Kobayashi Maru en el área del desarrollo (y en varias me he visto envuelto) y que ante la disyuntiva de ascender en el escalafón de la incompetencia, siempre ha decidido la otra opción.

Mis aficiones son la lectura, sobre todo ciencia ficción y literatura del período decimonónico, quiero creer que experto en Jules Verne. Me gusta escribir y soy podcaster aficionado.

EED: Para empezar , unas cuantas preguntas cortas para conocer mejor tu entorno tecnológico.

¿Ordenador principal?
¿Tablet y móvil?
¿Sistema operativo preferido?
¿Sistema operativo en el trabajo?
¿Editor de código o IDE que utilizas?
¿Libro físico o ebook?
¿Samsung o Apple?

Rafa:

¿Ordenador principal?
A caballo entre un iMac de 27 pulgadas del 2017 (que reemplaza a otro del 2011) y un MacBook Pro del 2017 (el del teclado mariposón y TouchBar) que reemplaza a un MacBook Pro Retina del 2012.

¿Tablet y móvil?
iPads (en plural), un iPhone (X)omer (S)impson (MAX)power (el XS Max) y un Note 9.

¿Sistema operativo preferido?
macOS, pero por poco, porque cada vez es más mierder.

¿Sistema operativo en el trabajo?
Windows 10 para desarrollar y Windows 7 como destino del desarrollo.

¿Editor de código o IDE que utilizas?
Visual Studio para casi todo y Visual Studio Code para la poca edición WEB que hago.

¿Libro físico o ebook?
Depende. Y dentro del depende vuelve a depender. Por ejemplo, libro físico para guardar, mismo libro electrónico para buscar.

¿Samsung o Apple?
Muy buena pregunta. Si me la hubieras hecho hace dos años, habría respondido que Apple sin pensar un momento. Hace un año habría dudado un poco más pero habría elegido Apple. Como me la haces ahora, te respondo que Apple y Samsung, y si tuviera que elegir o muerte, casi casi me quedo con Samsung. Seguiría con Apple por el ecosistema, pero por nada más.

EED: Esta es una pregunta fija de la entrevista, en tu caso estaba claro el contexto a escoger.

¿Qué añadirías a C++?
¿Qué eliminarlas de C++?
¿Qué modificarías de C++?

Rafa:

¿Qué añadirías a C++?
Nada. Es perfecto, ja ja. Siguiente pregunta. 😀

No, a ver, C++ es bastante críptico de leer, sobre todo cuando la base de código es grande y, aunque el concepto de biblioteca permite extenderlo hasta límites infinitos (y más allá, Cantor y palo mediante), la realidad es que cosas como Boost o la propia biblioteca base (que antes se llamaba STL) son increíblemente difíciles de usar debido a la potencia inherente de las mismas.

Por otro lado, entendiendo C++ como solo el lenguaje en sí, añadiría pre y post

condiciones, algo que tiene Eiffel (1). Si lo entendemos como un todo, añadiría una biblioteca estándar de salida por pantalla (que creo que es algo que está previsto para C++24 o así, o sea que saldrá en 2030 o después).

¿Qué eliminarías de C++?
Nada. En serio, nada. No se me ocurre absolutamente nada que le pueda sobrar. Eso no quiere decir que no pueda sobrarle algo, la cosa es que no se me ocurre nada.

¿Qué modificarías de C++?
Simplificaría la biblioteca base, o más bien haría una sintaxis más sencilla para ella, o al menos permitiría usarla con una sintaxis más simple y, si quieres potencia, pues adelante: functors, allocators, algoritmos súper precisos, cien tipos de autopunteros, operadores lambda, rangos, semántica de movimiento, gallifante infinito, …).

Por ejemplo, para obtener la hora del ordenador, tienes que usar sintaxis como "std::chrono::system_clock::now()", que devuelve un "std:: chrono::time_point" que a su vez tienes que templatizar a un objeto diferente si quieres ver la fecha en segundos, horas, días, etc.. Es súper potente pero a su vez súper complicado. Yo haría que sí, que puedes usar todo eso, pero solo si quieres. Y eso con las funciones de tiempo, si entramos en acceso a ficheros de forma asíncrona, sockets o puertos serie… Bueno, te puedes volver loco sin la ayuda del, en el caso de Visual Studio, IntelliSense y los gurús que han implementado la biblioteca que toque y que te ponen ejemplos de casi todo, a sabiendas de que si no hay ejemplo de algo, lo más seguro es que termines en una cuneta con un tiro en la cabeza disparado por ti mismo.

Respecto al lenguaje en sí, creo que no tocaría nada. Puedes usar referencias si se te atascan los punteros, y punteros como arrays (que es lo recomendado para que luego el compilador pueda generar instrucciones en ensamblador como REPNZ), tienes un modelo de excepciones muy chulos, puedes hacer desarrollo imperativo, procedural y, si tienes cojones, funcional. Hasta código espagueti puedes hacer.

Bueno, sí, modificaría la forma en la que se incluyen las definiciones de las clases y las bibliotecas, pero eso ya está en camino para el siguiente estándar, y se llaman módulos (sí, a la Python).

(1) Esto requiere un poco de explicación. Una precondición es algo así como, si una función recibe un entero, definir como precondición que sea cero o positivo, o que un puntero no pueda ser nulo. De hecho algo hay, porque por ejemplo puedes definir un método como nothrow y entonces no lanza ninguna excepción y si el código de debajo lo hace, el compilador protesta alto y claro (si se da cuenta, claro, porque si está en una biblioteca a veces no se entera y luego se dispara una excepción en un código que no está preparado para ello y… bueno, puedes liarla parda). O los explicit en los constructores. Pero vamos, que falta mucho para unas pre y post condiciones como Eiffel manda, y el problema es que ese tipo de "modificadores" introducen mucha "magia del compilador", y es algo que no se quiere hacer en C++.

EED: Hay tres grandes sistemas operativos para escritorio disponibles (Windows,MacOS y Linux), has utilizado , en diferente grado y tiempo, los tres

sistemas. Me gustaría conocer un poco tu opinión sobre ellos.

¿Podrías decirme una ventaja de cada sistema operativo?
¿Y una desventaja?

Rafa: ¿Respecto de uno al otro? Vamos allá, voy a responder a las dos preguntas a la vez, y seguro que mi respuesta no le va a gustar a nadie.

A fecha de hoy creo que ventajas, ninguno tiene ninguna sobre el otro. De todos modos, Windows es… Windows. Es como un granero grande, viejo y con goteras pero sabes que puedes meter dentro a todas las ovejas, cabras y gorrinos. Pero a poco que te descuides te cría ratas, los búhos anidan en el tejado y seguro que las goteras se convierten en ríos de agua justo encima de tu cogote.

MacOS es… un traje de Armani por fuera. Hace unos años también era Armani por dentro, ahora es el mismo nido de ratas, goteras y búhos.

Linux es un tractor viejo y renqueante que a veces se detiene solo y tienes que bajar de él a darle golpes con una llave inglesa y que encima nunca sabes cuándo se va a parar ni por qué lo ha hecho.

Por lo tanto los tres son la misma basura con diferente collar, ya dependerá de las ganas que tengas de llenarte las manos de grasa, perder horas resolviendo problemas, y tu afinidad política.

Windows está cada vez más amariconado y muestra los errores como si fueran magias potagias (“algo ha pasado”, “algo fue mal y es nuestra culpa”, etc., y cuando te muestra algún mensaje con fundamento, salen errores del tipo “Error 0x443000033” que ni la propia Microsoft tiene ni puta idea de qué son y a qué se deben).
MacOS cada vez va peor y si algo no funciona, pues te jodes y esperas a que la magia actúe y se arregle solo o con la siguiente actualización dentro de seis meses.

Linux… bueno, linux te dirá exactamente qué bit de qué byte y en qué dirección de memoria falló. Lo demás es cosa tuya. Tendrás que mirar cincuenta tutoriales desactualizados, entrar en quince foros en los que te tratarán como imbécil, leer dos millones de líneas de código asqueroso escritas en C de los años 80 para, al final, esperar y rezar para que el enésimo cambio en el enésimo fichero de configuración arregle tu enésimo problema, o que la enésima actualización arregle tu problema y no rompa otra cosa porque el genio de turno pensó que el fichero “hosts” ahora se llama “hosts.conf” y no se lo dijo a nadie, el que hizo el script que lee el fichero lo cogió directamente del sistema de ficheros en lugar de preguntarle al sistema dónde está dicho fichero (y eso si existe esa llamada), etc., etc., etc..

Ah, coño, las ventajas. Windows es increíblemente seguro. Con macOS follas seguro en el Starbucks y con Linux… bueno, con Linux si te falla el desodorante no te van a decir nada porque, bueno, eres uno de esos.

EED: Uno de los ámbitos mas usado y donde hay mas desarrollos es el desarrollo web , aunque al principio de la web existía la programación en CGI, en la actualidad no se utiliza, existiendo otros lenguaje ampliamente usados como JavaScript,PHP, Ruby o Python.

¿Porqué C++ no se utiliza en el desarrollo web?

Rafa: ¿Porque los desarrolladores Web son unos completos inútiles que no saben hacer ni la O con un canuto? No, es broma, o parcialmente broma.

Te cuento una historia que me pasó hace unos años hablando con otro desarrollador comentando qué hacíamos cada uno en nuestro curro. Cuando yo le dije lo mío, me espetó que “yo antes también hacía eso, pero ya he salido de todos esos rollos y ahora me encuentro muy confortable desarrollando Web”, como si desarrollar hardaware o sistemas fuera algo apestoso.

Lo que ocurre es que para tirar líneas con C++ necesitas una disciplina y unos conocimientos que no todos los programadores quieren tener o quieren mantener a lo largo de su carrera. Llámalo gandulería intelectual, o como quieras, pero es un hecho que muchísima gente se acuerda con odio de sus tiempos de universidad cuando le obligaban a escribir código en, casi seguro, C++ mierdoso (entendiendo C++ como un mejor C, por oposición a C++ de verdad), sin darse cuenta de que gracias a que tiró código con C++, ahora es un buen programador.

Respecto a la etapa de los CGI, te veías abocado a dos problemas. El primero es que necesitabas desarrolladores caros, y si no lo eran, podían generar verdadera mierda de código y que luego otros programadores tenían que sudar la gota gorda para desentrañar esa porquería de programa o simplemente tirarlo a la basura y empezar desde cero una supuesta mejora de código pero que en realidad, al cabo de dos meses, se convertía en la misma basura pero con otros nombres de clases.

El segundo es el nivel. C++ tal cual es un lenguaje sin nada más, por lo que necesita de un conjunto de bibliotecas de soporte. Por lo tanto, antes de tirar la primera línea de código, necesitas tener una biblioteca. ¿Acceso a bases de datos dentro del CGI? Biblioteca. ¿Construir HTML? Biblioteca. ¿Acceso a ficheros? Biblioteca. ¿Que cambias de arquitectura de servidor? Pues recompila y si tienes suerte, tendrás esas bibliotecas en esa arquitectura nueva.

Por lo tanto eso evolucionó, lo creas o no, a un súper C++. Que puede llamarse PHP, Python, Javascript o cualquier otro. ¿Por qué digo esto? Pues es muy sencillo, incluso en supuestas compilaciones de este tipo de lenguajes, lo que hay debajo, es C ó C++. Es decir, cualquier llamada o ejecución de uno de estos lenguajes, termina llamando a las bibliotecas de toda la vida de C/C++, conocidas domo runtimes. Por lo tanto, son macros que agregan muchas llamadas a primitivas escritas en C ó C++.

Aparte, estos lenguajes que tienen una permisividad y una flojera en cuanto a

formalidad que tumba, violando todas y cada una de las reglas formales del desarrollo. Permisividad con la cual muchos desarrolladores se encuentran como pollo en la mierda, básicmente porque su mente ha perdido ese filo del desarrollador de más bajo nivel, programadores a los que les da igual que una variable sea global, o que estés pasando un bloque de 100K de memora en una jerarquía de doscientas llamadas, desarrolladores que no tienen esa disciplina mental para codificar en condiciones.

Evidentemente, todos estos lenguajes facilitan el desarrollo Web, pero también enmierdan el código lo que no está escrito, y para un programador malo que haya en C++, hay cien mil en desarrollo Web, básicamente porque cuando va por la línea diez mil uno de su código se da cuenta de que no tiene lo que hay que tener para desarrollar con C++ y se pasa a PHP, JS o cualquiera de sus variedades.

Y con esto no quiero decir que todos los programadores Web sean malos, solo quiero decir que es mucho más fácil ser un programador malo escribiendo Web que escribiendo C++.

Y esto me lleva a la conclusión final, y es que si no existieran esos lenguajes que facilitaran el desarrollo Web, la Web como tal no existiría porque no habría tenido suficientes monos para teclear el suficiente código.

EED: Cambiamos de tema, a otro mas analógico y menos digital. Eres un ávido lector, tienes un podcast sobre libros, como buen lector me gustaría preguntarte sobre autores malos.

¿Has leído mas de un libro de un mismo autor que después de leerlo has considerado que has perdido el tiempo?
¿Qué autor o autores?

Nota: El sobrevalorado Verne no cuenta.

Rafa: ¿Sobrevalorado? ¡Tu sí que estás sobrevalorado! Es broma. Lo cierto es que sí, está bastante sobrevalorado porque, básicamente, no anticipó nada de nada excepto en una oscura novelita que se descubrió hace unos años en una caja fuerte… No es mi intención dudar de este tipo de descubrimientos, pero vamos, el texto me suena más que fuera del hijo que del padre, y si es así, estaría escrito entre 1900 y 1920 (y no sería de Verne). Dicho esto, todo eso del viaje a la luna, del submarino, la bomba atómica, etc.., Verne no anticipó una mierda. En Verne hay otros mundos mucho más interesantes. Pero no es momento para hablar de ello.

Respecto a la pregunta principal, sí, me ha pasado, y más de una vez. Por ejemplo Harry Turtledove con la novela Ruled Britannia, que es una historia alternativa en la que España conquista Inglaterra porque la tormenta que destruyó a la Armada Invencible no se produjo. El libro no me gustó porque la verdad es que las cosas no terminaban

de cuajar, con un Lope de Vega bastante descafeinado y un Shakespeare todavía peor, sin contar al Marlowe que estaba metido con calzador, así que sin pensarlo ni un momento, compré, de tirón, las 8 novelas de la serie Worldwar para descubrir que eran todavía más malas.

Larry Niven es otro de ellos con, por ejemplo, la serie de Mundo Anillo y las novelas aledañas del "Espacio Conocido". Este caso me pica más, porque después de leer todas las de la serie original, y pasar a "Protector" y otras en colaboración con Pournelle, ambientadas en el mismo entorno mezcladas con la del "Co Dominio" y que son, básicamente, basura y que fueron escritas para aprovechar el tirón del Mundo Anillo original... ¡volví a leerlas todas!

Luego están los autores noveles, como por ejemplo Alberto Meneses. No es que el chaval sea muy malo escribiendo, pero a veces mete la pata hasta el fondo y suele repetir las mismas escenas pero con diferentes personajes una y otra vez (y un par de veces más, sin contar cinco o seis más). Pues bien, tras leerme su serie de "El ocaso de los dioses" y no gustarme, compré el primero de la siguiente y, como seguía a la anterior, volví a cascarme la trilogía original... y no llegué a leer nada del nuevo, tan harto terminé.

Y eso de lo que me acuerdo ahora así a bote pronto. Vamos, básicamente, que debo estar ubérrimamente tonto por encima de mis posibilidades. No sé, en mi tierna inocencia de lector voraz quiero creer que segundas lecturas podrían demostrarme que lo que en primer lugar consideré como morralla no lo era... Además, debo de tener algún tipo de fusible fundido, porque cuando recibo publicidad de algún bundle de libros de temática que me gusta, y a sabiendas de que el 99% es basura y el otro 1% morralla, voy y lo compro, para luego abandonar los libros cuando llevo el 10% de cada uno, que es mi línea de corte si entra dentro de lo que yo catalogo como basura.

Y debe ser algo congénito, porque con la cacharrería me pasa igual: vuelvo a repetir a ver si la mierda que era ya no es. No sé, debo tener la esperanza de que, mágicamente o por arte de birlibirloque, las cosas se transformen o yo las vea desde otro punto de vista.

EED: Llevas muchos años trabajando y te habrás encontrado todo tipo de proyectos y clientes.

¿Qué es buen cliente para tí?

Rafa: El que está muerto. La verdad es que sí, es que he visto de todo. Desde el cliente que te pide una pequeña base de datos pero que una vez que le presentas lo hecho y pactado, empieza con es que se me olvidó decirte, para convertirte el programa en un CRM con acceso en tiempo real a cinco sistemas bancarios online, bolsa, estimación estadística de los índices bursátiles y cambio de divisas en tiempo real, ah, y que de paso llame a mi mujer cuando voy a llegar tarde, hasta el que simplemente te dice yo es que quiero que me hagan la pelota, pasando por la cliente

que no hace más que enseñarte la pechuga para ver si le descuentas algo.

Todo ello casos verídicos. Al primero intenté reconducirlo pero fue imposible. Al segundo le dije que mañana volvería a hacerle una visita y no volví más, y a la tercera, bueno, decidí empezar a mirarla directamente a los ojos e ignorar escotes, cortes de falda y demás, con lo que fue ella la que dejó de llamarme en cuanto le dije el precio.

Y son solo ejemplos. En mis tiempos mozos iba de desarrollador independiente, y lo cierto es que tuve solo un cliente bueno, que fue el único que conservé cuando decidí que lo mejor era la cuenta ajena, y lo conservé hasta que se murió el hombre de viejo y sus herederos decidieron cerrar el chiringuito, que se dice pronto.

Un cliente bueno es aquel que acepta los sablazos sin rechistar, al que puedes llamar lúser… No, ciertamente no. Un buen cliente es aquel que es razonable y que entiende que no vives del aire y que a veces metes la pata, y que algunas cosas son gratis pero otras hay que pagarlas. Que pueda querer algo para ayer pero también que los jodíos develóperos no tenemos máquina del tiempo, y que a veces una cosita más puede significar una semana de trabajo y un me da miedo pedirte esto, cinco minutos, hecho en el momento y gratis.

También que no tenemos bola de cristal y que no funciona puede significar miles de cosas. Joder, y que pague cuando dice que te va a pagar.

Creo que eso es todo.

EED: Habrás trabajado con muchos programadores de diferentes cualidades y calidades.

¿Qué debe saber un buen programador?

Rafa: Pues sí, la verdad es que siempre me he movido por una fauna de lo más variopinta, y encima en general esos desarrolladores han sido de plataformas distintas a mi área de experiencia, salvo cuando estuve en Embarcadero, y aun así yo iba especializándome en los Kobayashi Maru (como casi siempre me ha pasado en las empresas grandes en las que he estado) mientras que en el grupo en el que yo estaba había Javatrónicos, parseadores esecueleros y desarrolladores de C++ sin más.

Respuesta corta: hacer su trabajo lo mejor que sepa y pueda.

Respuesta larga. Puedo entender, de hecho entiendo, que a no todo el mundo le puedan gustar los Kobayashi Maru (no, Sam, no me olvido), o que no tengan ni la constancia ni la voluntad de tirarse varios meses, bregando día tras día con lo mismo, para conseguir realizar algo que, en ese momento, está desahuciado en la empresa porque todos lo han intentado sin éxito. Y con esto no quiero decir que sea mejor que nadie, sino que tengo más de perro de presa que de otra cosa.

Por lo tanto también entiendo al desarrollador que pueda gustarle lo sencillo (sin que

necesariamente tenga que serlo, ojo), como por ejemplo el trabajar exclusivamente con acceso a base de datos realizando, mes tras mes, las mismas altas-consultas-modificaciones-listados o el equivalente en backend o frontend. En esta situación yo me iría de la empresa buscando aguas más turbias.

Lo que no puedo entender es al programador gandul, al programador que pasa de todo, que yo no aprendo eso porque ya sé demasiadas cosas, o el que puntualmente no quiere tocar PHP porque él hace C++.

Menos aun al que, bueno, como ya estoy en un puesto fijo, paso de todo, al que se sienta confortablemente en su silla y su comodidad diaria, de nueve a cinco, cumplo y a casita. Y con esto no quiero decir que debas quedarte dos horas más todos los días, lo que quiero decir es que, de nueve a cinco, se pueden hacer muchas, pero muchas cosas interesantes.

Tampoco entiendo a las primas donnas, “yo soy tonto porque el mundo me ha hecho así”, soy mejor que tu y la tengo más gorda, cobro más y nunca llegarás a ser lo que yo soy. En general este tipo de develópero me da risa, y más cuando descubre que le han quitado la silla y va de cabeza contra el suelo, además, viendo cómo sus compañeros sonríen ampliamente.

O al que no colabora, no comparte su conocimiento, que es una variación del anterior. A ver, si realmente eres mejor que yo, y te preocupa toda esa gilipollez de primogenitura especial, por mucho conocimiento que compartas, cuando yo llegue a tu nivel, tu estarás a la misma distancia porque habrás evolucionado en la misma medida. Claro está, si no eres del tipo de los que he comentado más arriba, porque entonces, compartas o no, te van a rebasar por ambos lados.

En resumen, el programador ideal es aquel que se mantiene actualizado, humilde, inquieto, colaborador y, como decía un amigo, hormiguica.

Vale, y después de toda esta parrafada descubro que no he respondido a la pregunta. En fin, la neurona, que patina.

Un programador debería saberlo todo, ser dios en la tierra, comandar ejércitos más allá (y acá) de las Puertas the Tanhausser, hacerse un colgante con las estrellas de Orión y saber cómo plantarle cara a su jefe para hacerlo entrar en razón.

No, bueno, en serio. Debe saber programar. Y con eso no quiero decir que sepa manejar un IDE, quiero decir que tenga una base en lo que es un lenguaje de programación, cómo funciona, y luego saber qué tiene entre manos, sea un lenguaje, un framework o lo que sea.

Y por supuesto debe estar actualizado en lo suyo. Saber de patrones aunque no los aplique, de testing, de algoritmia (¿Alguien en la sala sabe qué es O(n)? ¿Y O(2n)? Pues eso). Debe conozer razonablemente el lenguaje (sin necesidad de llegar a ser un leguleyo del lenguaje), y las herramientas que esté usando.

Aunque no lo vaya a usar en su vida (por ejemplo porque es un desarrollador Web), debe tener conocimientos de sincronización, eventos, estructuras de datos, modelos de memoria... Y creo que no se me olvida nada.

También debe tener conocimientos generales de lo que hay debajo del lenguaje y sistema operativo para el que desarrolla (¿Quién sabe lo que es mapa del linker? ¿Y un intrinsic? ¿Y qué es heap exhaustion? ¿Sabe cómo reaccionar si su aplicación se queda sin memoria?).

Y seguro que se me olvida la mitad de cosas.

Y por supuesto, no debe ser un Bill Gates en relación a Knuth. (Un galiifante al que adivine la referencia, y si la sabe ya de antemano, que no diga nada).

EED: Imaginemos un mundo apocalíptico, donde nuestro entrevistado es el jefe supremo del campo de la tecnología e informática, un CTO con muchísimo poder , tiene el poder necesario para escoger que utilizarse y que no . Si solo se pudiera programar en un SOLO lenguaje de programación que debería ser utilizado en todos los ámbitos; web, redes,seguridad, escritorio...etc

¿Qué lenguaje escogerías?

Rafa: PHP.

Menudo susto, ¿eh? No, no es PHP ni de lejos, pero te lo puedes imaginar: C++.

Es una respuesta muy sencilla y para mi y para cualquiera con dos dedos de frente, la única alternativa, si abstraemos el hecho de en una situación como la planteada el único lenguaje de programación posible sería no el ensamblador, sino los códigos de operación, es decir, escribir directamente en hexadecimal. Pero vamos a asumir que justo antes de que se destruyera el mundo, me hubieran dado opción a elegir qué compilador/intérprete salvaría.

Elegiría C++. Porque con C++ podría acceder a programar un microprocesador sin sistema operativo, su sistema operativo, su pila TCP/IP, su navegador Web y sí, también los CGI para tener backend.

Ningún otro lenguaje me permitiría hacer todo eso sin grandes sacrificios y penalidades. ¿Cómo puñetas iba a meter un runtime de Java/PHP/Python dentro del micro de tu nevera, que tiene 1K de ROM, 256 bytes de RAM y un I/O de 4 líneas?

Y por otro lado, sí, la construcción del primer navegador Web sería complicada por encima de las posibilidades de cualquiera, pero una vez construidas las bibliotecas base... solo sería cuestión de usarlas.

Y no me vengáis con monsergas con que Python o cualquier otro churrimangui de lenguaje también se pueden hacer bibliotecas. Os recuerdo que Bjarne Stroustrup

inventó el C++ porque le encargaron hacer una aplicación con Smalltalk para monitorizar la red de la empresa en la que estaba, y no había empezado y ya ocupaba todo el ancho de banda de toda la red y de su ordenador de desarrollo.

EED: Continuamos con el contexto anterior, CTO con mucho poder, también tiene el poder de escoger el sistema operativo que se utiliza en esa sociedad. Si solo se pudiera utilizar un sistema operativo, a escoger entre Windows,Linux y MacOS. Igual que en la pregunta anterior, se debe utilizar en todos los ámbitos; servidores, escritorio,móviles/tablet,domótica y cualquier otro dispositivo.

¿Qué sistema operativo escogerías?

Rafa: Linux. Y no, no es broma.

Si no tuviera más cojones que escoger, iría a por Linux más que nada por el modelo bazar, pero lo transformaría en catedral dejando abierta la colaboración ciudadana, pero habría un dictador horroroso que os ibais a reír del Tolvards.

A ver, para los fundamentalistas: GNU/Linux, ¿vale? Es decir, software libre.

El mayor motivo sería por el concepto teórico (y nada más que teórico, porque en la práctica… bueno, no debo decir palabrotas) de la libertad entendida no como cerveza gratis sino como agacha el lomo y curra como un maldito.

Eso sí, forzaría algunas cosas con extrema dureza. Cualquier cambio debería estar documentado al milímetro, los cambios de diseño/interfaz, más, los forks solo si documentas el motivo y yo te autorizo.

Establecería API (o ABI) específicas y fijas entre fronteras. Por ejemplo, ¿has hecho un driver para mover una pelota vasca? Perfecto. Tu driver debe cumplir estas normas y utilizar este interfaz. ¿Has cambiado algo del Kelmer que se comunica con algo externo (como puede ser otra parte del kelmer)? Pues nada, a cumplir ABI y API. ¿Estás haciendo un nuevo Kelmer desde cero? Cojonudo, sabes que tienes 16.345 llamadas de API que debes cumplir, y el ABI es este. Lo mismo con la glib. X. Lo que sea. La frontera debe estar completamente definida, incluso la forma de leer un fichero de configuración.

Que, por cierto, es lo que hace Microsoft internamente.

Ah, y para aquellos que todavía no se hayan extrañado lo suficiente, mi orden de elección sería: Linux, Windows y macOS.

EED: Como última pregunta de la entrevista , una pregunta un poco diferente.

¿Qué te hubiera gustado que te preguntase? Evidentemente debes responderla.

Rafa: Me gusta que me hagas esa pregunta, porque así puedo explayarme en todo lo maldito e incorrecto que hay en el mundo.

Realmente son dos. La primera de ellas es ¿Cuándo fue tu primera vez… con el desarrollo a nivel profesional? (Mal pensados que sois). Pues mira, me gusta que me hagas esa pregunta porque fue un proyecto muy pero que muy interesante. Guiño guiño.

Y que quizás predijo por completo mi futuro. Andaba yo haciendo Formación Profesional Segundo Grado allá por los ochenta del siglo pasado y conocí al hijo de un empresario del calzado e hicimos buenas migas. Años después, pero no muchos, siendo yo un veinteañero de pro, se puso en contacto conmigo porque tenía una idea.

Ninguno de vosotros sabéis lo que es una máquina de escalado de hormas, modelo Incoma, y no os lo voy a explicar. Básicamente es un pantógrafo industrial 3D (pero de los años de catapúm), todo ajustes manuales. Pues bien, este chaval, ya a cargo de parte de la empresa de su padre, había estado en una feria en Italia y había visto cómo versiones más modernas de esa máquina, contaban con un ordenador y se podían "programar" configuraciones, que una vez cargadas en la máquina, esta movía los ajustes (ahora automáticos), y estaba lista para funcionar, sin los inconvenientes de que el operario se equivocara moviendo los pomos, pues todo de ajustaba mediante pomos rotatorios, incluso en el modelo antiguo.

Este chaval preguntó el precio y se volvió a su casa en España igual que había ido, pero se acordó de mi y entre él, un chaval que diseñó el hardware con motores paso a paso, y dos puertos paralelos en un PC, duplicamos la máquina. Además, como el código fuente estaba disponible, al ajustamos a sus preferencias y lo que es más, preparamos un entorno de compilación para que el propio chaval pudiera hacer cambios.

¿Sabéis cuál fue el lenguaje con el que lo hicimos? Sí, lo habéis adivinado. Borland C++, pero no usamos TurboVision (porque era la versión 1.0 y era bastante mierder), sino que el que esto suscribe escribía directamente en la memoria de vídeo en modo texto. Recuerdo haber medio duplicado algunos controles de Windows.

Ese fue mi primer proyecto de desarrollo profesional, y en ello sigo.

Bueno, ahora la segunda pregunta: ¿Por qué dices que Jules Verne no predijo una mierda? Pues bien, corrían los años veinte de… Bueno, mejor esta os la respondo en un Loleido o en un Leña al Mono. Estad atentos.

EED: Hoy es el último día de la entrevista y es el momento de la despedida, pero antes

me gustaría agradecerte tu participación, espero que haya sido una experiencia interesante y entretenida para tí.

Puedes indicar tus métodos de contacto y si tienes algún proyecto, web, podcast o evento que quieras promocionar tienes este espacio para hacerlo.

Por último, me gustaría que me propusieras a una persona que tú creas que estaría dispuesto a participar en una futura entrevista.

Ha sido un placer , hasta la próxima.

Rafa: También ha sido un placer para mi, pese a que seas un pythoner irredento, pero nadie es perfecto excepto yo, que desarrollo con C++.

Me lo he pasado bien dando caña, y también con la que me habéis dado (aunque esperaba más, la verdad). Ha sido toda una gozada.

Respecto a promocionar algún proyecto personal, pues lo cierto es que hace bastantes años que dejé de tener relevancia en la comunidad, aunque en su momento fui bastante conocido a causa de ser uno de los únicos noventa y pocos MVP de Microsoft en el lenguaje C++ de todo el mundo, pero el cansancio inherente a estar continuamente alerta, más cierta degradación de los citados premios hizo que fuera perdiendo interés… y aquí estamos.

Por lo tanto, respecto al desarrollo, no tengo ni colaboro en ningún proyecto, pero sí que formo parte de algunas redes:

- Sospechoso Habitual de la Red de Podcasts (https://feedpress.me/sospechososhabituales con el mismo nombre.
- Irredento y socio fundador de la comunidad de WinTablet, compuesta por un blog (http://wintablet.info), un slack (https://wintablet.slack.com), y un podcast (http://www.youtube.com/results?search_query=#wintabletinfo), en donde unos viejunos barrigones de las generaciones ’68/’70 del siglo pasado hablamos de lo que nos sale y como nos sale.
- Amo y señor de mi podcast sobre cualquier tema que quiera tratar, Leña al mono que es de goma (https://www.ivoox.com/podcast-lena-al-mono-es-goma_sq_f1479496_1.html). Irredento, iconoclasta y cañero en el que no dejo títere con cabeza.
- Mi otro podcast, algo más serio, Loleido, libros sin horas (https://www.ivoox.com/podcast-loleido-libros-sin-horas_sq_f1650517_1.html) en el que hablo de literatura, pero de la literatura que he leído el mes anterior, con dos entradas por mes: la lectura, en la que hablo sobre el libro realizado el mes anterior que más me ha gustado y la anti-lectura, en la que hablo sobre el libro o libros que menos me han gustado, también durante el mes anterior al de la salida del correspondiente episodio.

Propongo para entrevistar a Sergio Navarro Pino (jodío develópero de C#) y a Peni (@FOldani, otro jodío develópero, pero en este caso de Visual Basic, .NET y del otro). Y eso es todo, sed buenos y no olvidéis sospechosohabitualizaros. Ah, y que no os la pique un pollo.

EED: Doy por finalizada la entrevista, pero no se vayan muy lejos que mañana comienza la siguiente . Este mes, muy excepcional, haré tres entrevistas, mañana comienza la segunda.

El entrevistado tiene que perfil diferente al anterior , en el ámbito de redes.

Si quieres obtener mas información, hay disponible una cuenta de Twitter https://twitter.com/EDiferido y un grupo privado de Telegram https://t.me/joinchat/AF1F5g2rHkPxUk3r4c_QWA