

*Due proposte per dispositivi mobili Open Source: Firefox OS e Ubuntu Phone. Due diverse filosofie di utilizzo e di sviluppo per smartphone a basso costo.*



## Cosa c'è oltre Android e iOS?

Come osservavamo nella rubrica Linux pubblicata sull'ultimo numero di *Pc Professionale*, una vera convergenza fra web, desktop e terminali mobili, che potrebbe semplificare la vita di parecchi utenti, potrebbe realizzarsi solo se arrivassero standard aperti e accettati da tutti in almeno due campi: procedure di sviluppo e installazione delle relative applicazioni, e possibilità per queste ultime di sfruttare adeguatamente tutte le risorse locali, dai file alle notifiche di sistema e periferiche hardware, su qualunque terminale (o computer tradizionale).

Questo mese torniamo sull'argomento presentando due proposte Open Source in questo campo, che lavorano a livelli diversi ma con lo stesso ambizioso obiettivo: abbattere il duopolio attuale di Android e Apple su Web e terminali mobili.

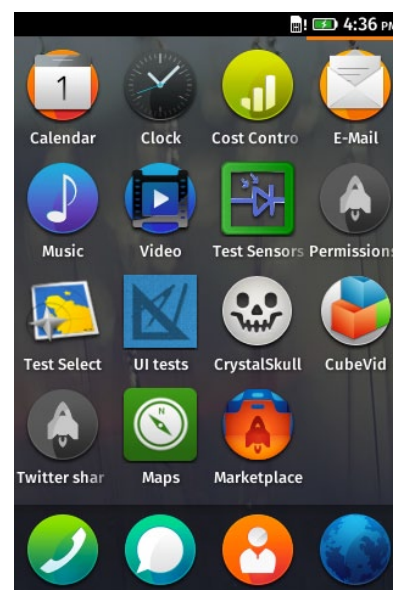
### **FIREFOX OS: IL WEB PRIMA DI TUTTO**

La Fondazione Mozilla, quella che dirige lo sviluppo del browser più famoso del mondo, mira a diventare il fornitore preferito di chiunque voglia usare uno smartphone Open Source moderno, al minor costo possibile. Già oggi è possibile acquistare terminali con il suo sistema operativo Firefox OS ([www.mozilla.org/it/firefox/os/2.0](http://www.mozilla.org/it/firefox/os/2.0)) in decine di

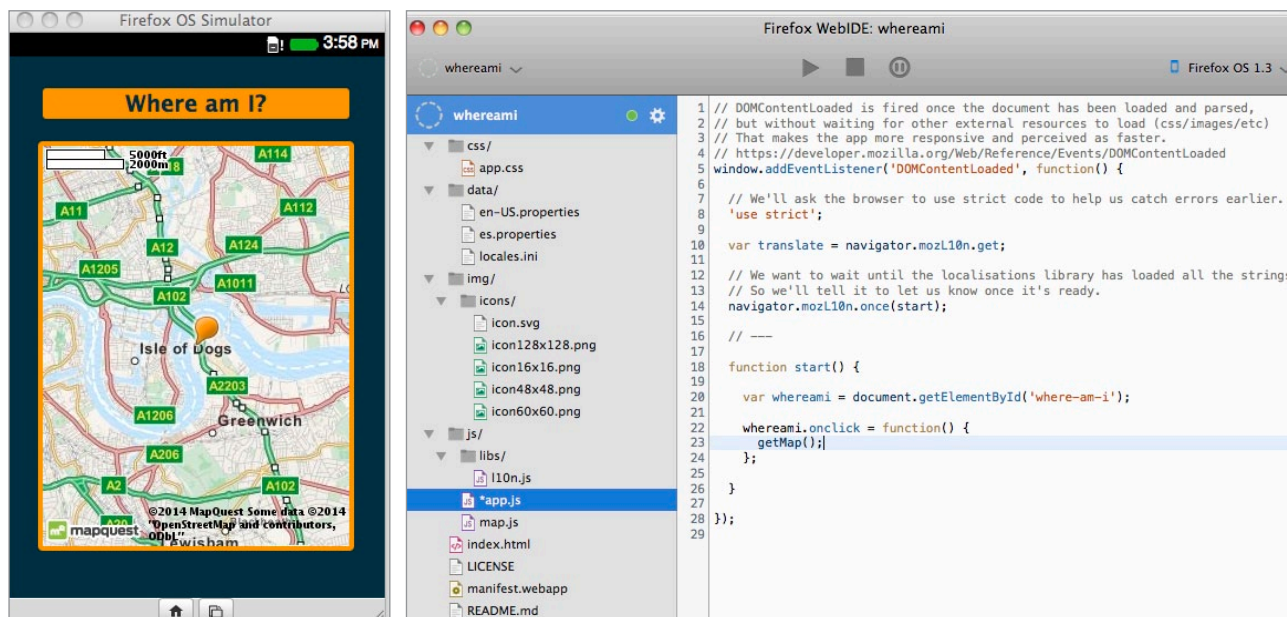
paesi dall'Australia al Venezuela. In quelli emergenti o in via di sviluppo, a partire dall'India dove ne sono già stati venduti oltre mezzo milione, gli smartphone Firefox OS possono costare anche poche decine di dollari, e c'è di più. Mozilla punta anche al mercato delle Smart Tv, come dimostrano un recente accordo con Panasonic e soprattutto Matchstick ([www.matchstick.tv](http://www.matchstick.tv)): una sorta di chiavetta Usb da venticinque dollari, controllabile da smartphone, per visualizzare qualsiasi contenuto online su un televisore.

**Firefox OS** è fondamentalmente il browser dello stesso nome, spezzettato e completato da vari servizi di supporto. Il suo punto di partenza è proprio il problema discusso nel numero scorso: i browser di oggi, Firefox incluso, non dispongono ancora di metodi standard per far interagire le app (o i siti Web a cui accedono) con l'hardware su cui girano, di qualunque tipo esso sia. La soluzione di Mozilla consiste nel trasformare ogni programma software in una applicazione Web, a partire dall'interfaccia utente. Questo ambiente, chiamato Gaia, definisce e gestisce (ma non crea, né esegue direttamente, come vedremo tra poco) qualsiasi oggetto rappresentato sullo schermo dopo che Firefox OS è stato avviato, dalla schermata di blocco o iniziale a tutte le

applicazioni standard. Gaia offre più o meno gli stessi servizi e strumenti di un desktop environment e window manager tradizionale per Linux, ma visivamente è un ambiente smartphone davvero standard. Nella parte alta del display c'è la classica barra di stato che indica la carica della batteria e il livello



Gaia divide lo schermo di uno smartphone in tre aree: pannello di sistema, griglia delle applicazioni, e area separata (in basso) per quelle più frequentemente usate.



L'ambiente di sviluppo WebIDE è ricco di funzionalità e fornisce sia un editor di codice integrato in Firefox (nell'immagine a destra) sia un simulatore software in cui collaudare immediatamente un'applicazione (a sinistra) per eseguire le verifiche.

di campo. Subito sotto, le icone di vari semplici programmi, un misto fra quelli integrati nei desktop Linux di qualche anno fa e quelli di qualsiasi smartphone economico commerciale: telefono, calendario, client di posta, software per foto e video e pannello di configurazione. Tutto questo è definito esclusivamente con gli stessi linguaggi di qualsiasi sito Web moderno: Html, CSS, e JavaScript. Gaia fornisce accesso a dati e hardware del telefono direttamente da funzioni HTML5, ma *solo* attraverso chiamate a "Web API" generiche. In Firefox OS, infatti non esistono né sono previste interfacce native e dirette fra sistema operativo e applicazioni, come potrebbero esserci fra programmi Gnome e Linux su un desktop tradizionale. Grazie a questa architettura, Gaia può girare con relativa facilità praticamente su qualsiasi browser e sistema operativo, anche se ovviamente non è garantito che i risultati siano sempre gli stessi.

## GECKO

Gaia e le app scritte per questo ambiente decidono cosa deve apparire su uno schermo Firefox OS, ma chi effettivamente "disegna" gli oggetti e li fa interagire con l'hardware e l'utente è un software già collaudato e usabile in vari browser e sistemi operativi, chiamato Gecko. È questo il nome in

codice dell'accoppiata fra un motore (in inglese, *engine*) di layout molto efficiente e una libreria di "componenti per browser".

In generale, i motori di layout sono programmi che dispongono e disegnano su schermo i vari elementi di una interfaccia software o di un documento digitale. Quelli nati per il Web come Gecko formattano e presentano *all'interno* di una singola finestra o scheda di un browser i documenti scritti (magari in tempo reale, da qualche altro software) in Html, Xml o altri formati, in base alle istruzioni fornite dai marcatori Html nei documenti stessi e da fogli di stile Css.

Oltre a questo Gecko contiene e può disegnare anche *tutti* gli altri elementi di un browser: menu, barre degli strumenti o di scorrimento, pulsanti e così via. Completano la dotazione un interprete JavaScript e altre librerie per la gestione di sicurezza, aggiornamenti, buffer video e networking.

## GONK

A rigor di termini, proprio perché è internamente così simile a Firefox, Gecko può già girare su vari sistemi operativi. Ovvero mette a disposizione di applicazioni Web, già da oggi, *sempre le stesse*

*funzioni* di interazione con l'hardware, dalla telefonia al controllo dello schermo, su qualunque smartphone o computer tradizionale. In pratica, il codice *davvero* portabile al 100% è piuttosto raro, come ben sappiamo almeno fin dalla nascita di Java.

Per mantenere pieno controllo del suo progetto pur dimostrando prima possibile che può funzionare come promesso, Mozilla è stata costretta a chiudere il

cerchio, aggiungendo a Gaia e Gecko il pezzo che mancava: un mini-sistema operativo Open Source completo, di cui poter garantire davvero, almeno su dispositivi certificati, che Gecko funzionerà al massimo del

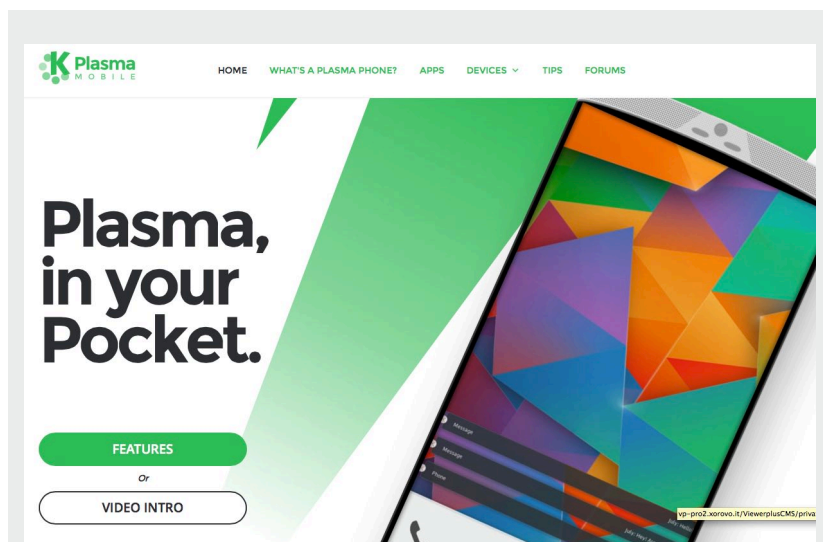
suo potenziale, sempre nello stesso modo. Questa piattaforma si chiama Gonk ed è una distribuzione Linux relativamente semplice. I suoi due componenti principali sono un kernel Linux vero e proprio, sostanzialmente identico a quello di Android, e uno strato di astrazione dell'hardware (*Hal*, in inglese). Quest'ultimo raggruppa tutte le funzioni di basso livello, a partire da quelle che variano da smartphone a smartphone.

Rientrano in questa categoria, condividendo per quanto è possibile il relativo

## La semplicità di Firefox OS

Solo standard aperti e condivisi direttamente con il Web per avere le funzionalità di base.





## NEL FRATTEMPO, KDE...

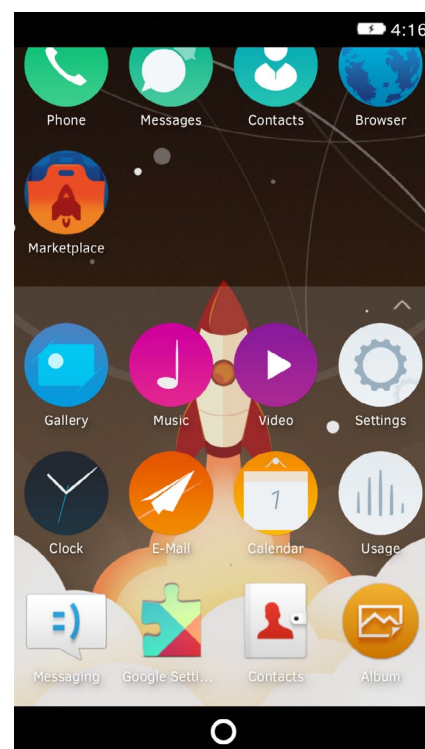
**I** pezzi da novanta come Mozilla e Canonical catturano quasi tutta l'attenzione di chiunque vorrebbe un sistema operativo mobile e Open Source, che però non sia Android. Dietro di loro, però, c'è anche un piccolo gruppo di sviluppatori che da anni lavora sullo stesso obiettivo, con mezzi molto più ridotti ma con un approccio niente affatto disprezzabile. Stiamo parlando del progetto Plasma Phone (<http://plasma-phone.org>), una piattaforma software che utilizza soprattutto librerie, programmi e interfacce utente del desktop Open Source Kde. Alcune tecnologie di Plasma sono le stesse adottate in Ubuntu Phone e quindi presentano, in teoria, gli stessi rischi di frammentazione descritti nell'articolo principale. La missione esplicita di Plasma Phone, però, è proprio di fornire agli utenti finali un ambiente completamente libero, in cui sia possibile installare applicazioni mobili di vario tipo, o crearne di nuove che siano immediatamente riutilizzabili altrove. Il tutto nel massimo rispetto della privacy.

**Sia per risparmiare risorse sia per principio**, in generale Plasma riutilizza quanto più possibile programmi e strumenti già standard in Kde, o al massimo in Linux. A basso livello, per esempio, le librerie per audio e messaggistica istantanea sono le stesse nate in quell'altro progetto, ovvero PulseAudio e Telepathy, mentre il Window Manager è Kwin. Il server grafico è Wayland, lo stesso che nel medio/lungo termine probabilmente sostituirà X.org in tutte le distribuzioni Linux da desktop. Solo le librerie e applicazioni davvero specifiche per smartphone, come quelle Ofono (<https://01.org/ofono>) per la telefonia, vengono direttamente da Android o progetti simili. L'interfaccia utente del Plasma Phone, infine, è la versione per smartphone di quella più recente di Kde ed è chiamata appunto Plasma Mobile (<http://plasma-mobile.org>). In confronto a Firefox OS e Ubuntu Phone, Plasma è un progetto amatoriale, ma va comunque tenuto d'occhio, per un motivo ben preciso: la sua architettura e filosofia potrebbero rendere molto più facile l'aggiunta di app di qualsiasi natura da parte di sviluppatori indipendenti. Un programmatore Linux già esperto potrebbe farlo, ad esempio, senza imparare da zero le interfacce e procedure di Android o Google Play. Se le cose vanno come previsto, inoltre, in futuro dovrebbe essere possibile installare app Android grazie al progetto Shashlik ([www.shashlik.io](http://www.shashlik.io)), il cui motto è appunto "applicazioni Android, ma su un Linux vero!"

codice con Android, tutti i driver dei chipset che gestiscono fotocamere, vibrazione, Gps e così via.

Nessuna delle funzioni corrispondenti è *direttamente* accessibile dalle Web Api di Gecko. Non potrebbe essere altrimenti, e non solo perché parte di quel software, per ovvie ragioni, potrebbe essere proprietario. Se uno sviluppatore dovesse scrivere una versione delle sue app con Web Api per Gonk e un'altra per gli altri casi, l'intero progetto Firefox OS sarebbe inutile. Per questo motivo, tutte le Web Api di Gecko hanno modalità di *fallback*, ossia di ripiego: quando effettivamente girano su Gecko svolgono esattamente quella che è la loro funzione principale. In tutti gli altri casi, è comunque garantito che verrà eseguita qualche operazione più o meno equivalente (per esempio, una notifica su schermo anziché una vibrazione) e che non ci saranno crash dell'applicazione.

In tutto il resto, Gonk è sostanzialmente Linux, o al massimo Android. Oltre al kernel, di cui abbiamo già scritto, Gonk riutilizza direttamente parecchio software presente in ogni distribuzione

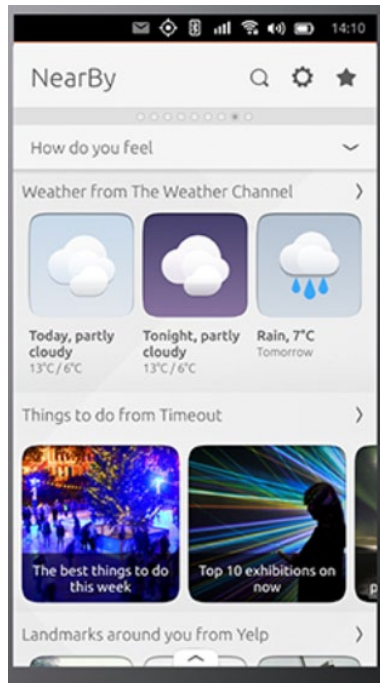


**Non c'è bisogno di comprare un nuovo smartphone per provare Firefox OS! Basta installare bg2droid per far girare Gaia, la sua interfaccia utente, su qualsiasi smartphone Android (fonte: [people.mozilla.org](http://people.mozilla.org))**

desktop, dal client DhcP a Wpa\_Suppliment per gestire connessioni Wi-Fi. Anche la procedura di boot è sostanzialmente la stessa. Le prime fasi possono variare parecchio da uno smartphone all'altro, soprattutto per quanto riguarda gli aggiornamenti del firmware. In ogni caso, dopo la schermata iniziale e l'attivazione della parte radio, prima o poi si esegue lo stesso processo speciale che avvia quasi tutti i server e desktop Linux del mondo: init, che monta i vari file system disponibili ed esegue tutti gli script necessari per lanciare i servizi di sistema. Su Firefox OS, il più importante di questi servizi è il processo chiamato b2g ("Boot to Gecko"). È lui che effettivamente ha accesso all'hardware tramite i vari driver, e che fa girare anche Gecko.

## DOVE SONO LE APP?

Come forse è già ovvio dalla lettura dei paragrafi precedenti, le app "native" di Firefox OS non possono essere diverse dall'interfaccia in cui girano, cioè Gaia. Internamente, non sono altro che insiemi di file sorgenti in senso stretto (sempre nei formati Html 5, Css e Javascript) e di altri "contenuti" di vario tipo, come immagini, audio e video, la cui distribuzione e installazione può avvenire in due modi. Le app cosiddette *packaged* sono le più simili ai tradizionali pacchetti binari delle varie distribuzioni Linux: singoli archivi in formato Zip, scaricati da un archivio online affidabile (*Marketplace*) e conservati nello smartphone. I gestori dell'archivio garantiscono, tramite analisi dei sorgenti o altre procedure, l'affidabilità e l'integrità del pacchetto, firmandone digitalmente ogni parte e soprattutto il suo *manifesto*, che include la lista completa di tutti i componenti. Le app *hosted* ("ospitate"), invece, sia tecnicamente sia dal punto di vista della sicurezza, sono molto più simili alle



Gli "Scope" di Ubuntu mettono le app in secondo piano. Ognuno di loro elenca direttamente le cose da fare, o i contenuti da visualizzare, in un certo contesto.

normali pagine Web. I vari file sono scaricati da un Url predefinito, solo quando effettivamente servono. Nello smartphone viene scaricato e conservato soltanto il manifesto, che contiene appunto quell'Url e altre informazioni generali.

Di qualunque tipo sia un'applicazione, essa per principio riceve *solo* i permessi di cui ha bisogno per essere installata e avviata. Qualsiasi altra autorizzazione deve essere dichiarata nel manifesto e quindi, nel caso di app packaged, preventivamente verificata e approvata dai gestori del suo Marketplace. Per questo, in generale, solo le app di questa

categoria possono avere i permessi più importanti, quelli che se abusati potrebbero distruggere dati dell'utente o comprometterne la privacy.

## SVILUPPARE PER FIREFOX OS

Lo sviluppo di applicazioni è un campo in cui Firefox OS si allontana parecchio dal mondo Linux tradizionale. I programmatori devono utilizzare un ambiente molto diverso per scrivere app e poi collaudarle in un *runtime*: il suo nome è WebIDE e gira per default all'interno di Firefox, ma in certi casi è utilizzabile anche con Chrome o Safari. Quanto al *runtime*, è semplicemente il nome in codice per indicare l'ambiente in cui l'app da collaudare andrebbe a girare: questo può essere uno smartphone che già usa Firefox OS, connesso al proprio computer tramite cavo Usb, oppure un simulatore software. La cosa migliore di tutto il sistema Firefox Os per sviluppare applicazioni è che basta pochissimo per farne anche versioni Android. Il servizio "Open Web Apps for Android" (vedi Box Risorse), a disposizione di tutti, può creare pacchetti Android in formato Apk di app già approvate, aggiungendovi uno strato di codice Java che gli permetterà di girare all'interno di una "Web Runtime" inclusa in Firefox per Android.

## LA RISPOSTA DI UBUNTU

Canonical, l'azienda che è il principale sponsor commerciale di Ubuntu, ha idee molto diverse da Mozilla, ma altrettanto interessanti, su come conquistare un posto sul podio dei sistemi operativi mobili. Pur essendo entrata in gara molto più tardi di Google e Apple, Canonical ha idee piuttosto particolari su come raggiungere quell'obiettivo (facendo tesoro dell'esperienza dei due



## RISORSE

**I**ue buone descrizioni, dettagliate ma non troppo, dell'architettura di Firefox OS e del suo ambiente di sviluppo WebIDE sono disponibili sul sito italiano per gli sviluppatori Mozilla, agli indirizzi [https://developer.mozilla.org/it/Firefox\\_OS/Platform/Architecture](https://developer.mozilla.org/it/Firefox_OS/Platform/Architecture) e, rispettivamente, <https://developer.mozilla.org/it/docs/Tools/WebIDE>. Per saperne di più su come distribuire applicazioni Firefox OS nel formato Apk di Android, suggeriamo di leggere la pagina [www.androidworld.it/2015/06/09/firefox-os-android-apk-307651](http://www.androidworld.it/2015/06/09/firefox-os-android-apk-307651). Per Ubuntu Phone, o Touch, conviene invece partire dall'articolo [www.ubuntuuphone.it/news/cosa-e-ubuntu-touch-278](http://www.ubuntuuphone.it/news/cosa-e-ubuntu-touch-278) e dalle varie guide disponibili sullo stesso portale, tutte in italiano. Su YouTube si trova infine un demo di KDE Plasma Mobile nel video [www.youtube.com/watch?v=auuQAQ8qpM](http://www.youtube.com/watch?v=auuQAQ8qpM).

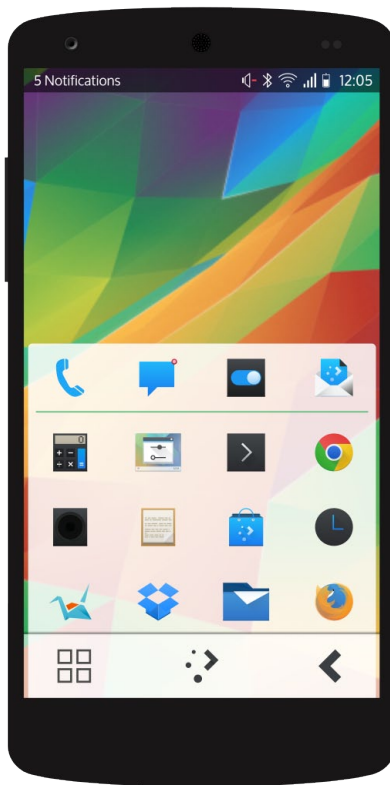
concorrenti). I suoi sviluppatori lavorano infatti da anni a una versione mobile di Ubuntu, chiamata prima Touch e poi Ubuntu Phone, guardando soprattutto a due classi di smartphone: i top di gamma, e quelli base, destinati soprattutto a chi non ha alcun interesse alle "app" in quanto tali, ma userebbe volentieri il telefono anche e soltanto per navigare su Internet e usare la posta elettronica. Da queste considerazioni è nata una versione di Ubuntu che gira sullo stesso hardware su cui gira Android, ma con interfaccia e applicazioni piuttosto diversi anche quando funziona soltanto come smartphone, nel senso che spiegheremo fra poco. La prima differenza è il fatto che in Ubuntu Phone le app, più o meno ordinate in righe e colonne, quasi spariscono dietro pannelli a tutto schermo, chiamati "scope" (contesti) oppure "home screen" e indipendenti fra loro. Non c'è nemmeno bisogno di pulsanti, su schermo o hardware. Al loro posto, Ubuntu Phone usa un sistema chiamato Edge Magic, in cui basta scorrere il pollice da un lato dello schermo verso il centro. Partendo dal basso si aprono i controlli dell'app usata in quel momento, e dall'alto si apre la barra per notifiche e configurazioni di sistema.

Scorrendo da sinistra si apre la griglia delle funzioni e app più usate, e da destra la lista delle sole app aperte in un dato momento.

Per quanto riguarda gli scope, in linea di massima, c'è n'è uno "per ogni aspetto della vita", per usare le parole di Canonical. Quello principale visualizza tutti i dati e servizi usati più spesso, qualunque sia la loro natura o l'app che effettivamente li gestisce: contatti, siti Web, brani musicali, tutto insieme. A titolo di esempio, fra gli scope predefiniti spicca quello chiamato NearBy (in inglese "qui intorno" o "vicino a me"), per tutti i bisogni dell'utente che dipendono dalla sua posizione: è da qui si aprono le mappe di OpenStreetMap o Google o si trovano ristoranti oppure orari degli autobus. Lo scope chiamato News, invece, raggruppa le fonti d'informazione che si consultano regolarmente.

## UBUNTU PHONE... O DESKTOP?

Perché, se si ha già qualcosa potente come un Pc in tasca non usarlo anche in quel modo? Ma questo è proprio quel che si può fare con uno smartphone su



**Questo è solo un disegno, non una vera schermata, ma mostra chiaramente come potrebbe essere uno smartphone con l'interfaccia Plasma: il tradizionale desktop Kde, completamente ridisegnato per terminali mobili.**

cui gira Ubuntu Phone: basta connetterlo, per esempio con un hub Usb, a tastiera, mouse e monitor tradizionali e sullo schermo apparirà esattamente lo stesso desktop, con le stesse applicazioni, che si otterrebbe installando Ubuntu su un normale computer.

Questo è un approccio estremamente interessante, sia nel breve termine sia soprattutto dal punto di vista dell'hardware che potrebbe produrre. Si potrebbero avere, per esempio, "laptop" a costo, peso e consumi molto inferiori a quelli di oggi: se connessi a un Ubuntu Phone, infatti, questi dispositivi non dovrebbero essere altro che gusci per schermo, tastiera e pochissimo altro. Oltre al risparmio, potrebbero esserci benefici notevoli anche dal punto di vista ambientale, riducendo la quantità complessiva di hardware di cui si ha bisogno per lavorare e comunicare. E si potrebbe avere tutto questo non solo mantenendo la convenienza e facilità d'uso di uno smartphone moderno sempre in tasca, ma potendo telefonare o scambiare Sms anche in modalità desktop.

## LE APP, VERSIONE UBUNTU

Oltre al software tradizionale per desktop, che sarebbe utilizzabile come abbiamo appena descritto, Ubuntu Phone può supportare sia Web App in Html 5, sia app "native". I programmi della prima categoria sono sostanzialmente identici alle app di Firefox Os, o ai browser "monouso" descritti nel numero scorso. A questi si aggiungono delle app "native", più veloci perché scritte in maniera completamente diversa da quelle di Firefox OS.

Queste app girano in particolari contenitori (qmlscene, <http://doc.qt.io/qt-5/qtquick-qmlscene.html>) e usano altre interfacce, sempre Open Source ma specifiche di Ubuntu, che permettono di sfruttare l'hardware di uno smartphone molto meglio di quanto non sia possibile, per precise scelte di design, su Firefox OS. Questo vantaggio è però anche un problema, perché impedisce di riutilizzare le app su Android, o in generale su qualsiasi piattaforma, anche Linux, diversa da Ubuntu.

## CONCLUSIONI

Firefox OS è una sorta di derivato di Android per smartphone economici, deliberatamente fondato su massima semplicità e uso esclusivo di standard aperti direttamente condivisi con il Web: il suo slogan ufficiale è "darti esattamente ciò che ti serve, quando ti serve". Ubuntu Phone ha l'ambizione di reinventare le interfacce mobili Open Source, trasformando allo stesso tempo gli smartphone abbastanza potenti in desktop ultrapiattali. In teoria, almeno per utenti avanzati o business, questo secondo approccio è molto, ma molto più interessante di Firefox OS, o anche di Android.

Un vero desktop Ubuntu in uno smartphone renderebbe finalmente davvero "mobili", senza perdere nulla, migliaia di applicazioni, da quelle enterprise a semplici script shell. Il suo maggior difetto, niente affatto trascurabile, è il rischio di frammentare il mondo Linux in maniera diversa, e più seria, di quanto non lo sia già. Chi vincerà?

Non lo sappiamo, ma (formato delle app di Ubuntu a parte) non ci dispiacerebbe se fosse uno di questi sistemi. O anche tutti e due, visto che hanno scopi e scenari d'uso molto diversi, e in buona parte complementari. •



# NEWS

## UN CONTENITORE PER QUALSIASI NUVOLA: KUBERNETES 1.0



Ogni volta che ci serviamo di Gmail o Drive, facciamo una normale ricerca con Google o usiamo un qualunque altro servizio cloud di altri fornitori, possiamo farlo grazie a diversi pezzi di software di cui non sospettiamo nemmeno l'esistenza, ma senza i quali quelle cose non potrebbero affatto esistere. A luglio 2015 uno dei più importanti fra questi programmi "oscuri" ma indispensabili è diventato Open Source. Oggi ogni singola copia delle applicazioni che effettivamente costituiscono Gmail o molti altri servizi del genere non gira in un computer virtuale completo, ma in un contenitore software, molto più piccolo e lei riservato. In questo modo si possono creare nuvole molto più veloci, e molto più adattabili alle variazioni del carico. Per farlo però serve software come Kubernetes (<http://kubernetes.io>): un sistema di gestione integrata per i contenitori creati con il sistema Open Source Docker ([www.docker.com](http://www.docker.com)), capace di crearne e aggiornarne automaticamente a migliaia e dividerli in gruppi secondo le esigenze del momento. Oltre alle nuvole di Google, Kubernetes (traducibile più o meno come "timoniere") può gestire già quelle di Microsoft Azure ed è già usato in produzione da compagnie come eBay. La sua versione 1.0, quella rilasciata con licenza Open Source, ha fra i suoi autori anche sviluppatori di giganti dell'industria come IBM e Intel e sarà gestita da una fondazione dedicata, proprio per garantirne la compatibilità con nuvole di qualsiasi fornitore. Per saperne di più su come funzionano Docker e Kubernetes si può leggere il semplice articolo all'indirizzo [www.cloudtalk.it/container-docker-kubernetes](http://www.cloudtalk.it/container-docker-kubernetes).

## È ARRIVATO LIBREOFFICE 5!



**L**e aziende che non hanno alcun bisogno di funzioni nuove e mettono la stabilità prima di tutto possono continuare a servirsi della serie 4.4 di LibreOffice. Per tutti gli altri, invece, da questa estate è possibile installare LibreOffice 5 su Linux, Mac OS e anche su Windows 10! Oltre alla piena compatibilità con questo sistema operativo, LibreOffice 5 è alla base delle versioni mobili per Android e Ubuntu Phone, di quella per il cloud ed è anche pieno di nuove funzioni. L'interfaccia utente è stata semplificata, usando più efficacemente lo spazio su schermo grazie a nuove icone, menu e barra laterale. L'uso degli stili nel word processor è semplificato da una funzione di anteprima, mentre nel foglio elettronico sono presenti nuove funzioni per formattazione condizionale e indirizzamento delle tabelle. Aggiornati anche i filtri che permettono di usare i formati di file non standard di suite come Microsoft Office o Apple iWork. Infine, è possibile aggiungere la marca temporale standard ai file PDF.

## UBUNTU/CANONICAL E LINUX FANNO PACE. QUASI

**T**roppi utenti finali, è inutile negarlo, ancora oggi prestano pochissima o nessuna attenzione alle licenze del software che usano. Questo non toglie che sia indispensabile garantirne il rispetto e soprattutto, almeno in campo Open Source, la piena compatibilità fra diverse licenze nello stesso progetto, e fra loro e il modo di lavorare di tutti gli sviluppatori coinvolti. Per questo è un'ottima notizia l'annuncio, arrivato a metà estate 2015, che dopo due anni di discussioni la Free Software Foundation e Canonical, il principale sponsor di Ubuntu, hanno quasi completamente chiuso, con esito positivo, una disputa proprio su questo tema.

Canonical, infatti, ha finalmente aggiornato le norme relative a proprietà intellettuale e distribuzione dei pacchetti che governano lo sviluppo di Ubuntu, in maniera che elimina definitivamente certe ambiguità segnalate dalla Fondazione.

Prima dell'accordo, per esempio, non era completamente chiaro come e in che modo, in alcuni casi, era lecito ridistribuire pacchetti Ubuntu in formato binario. Si tratta certo di questioni invisibili e incomprensibili per gli utenti finali ma, ripetiamo, fondamentali per garantire la piena accettazione di Ubuntu, e dell'Open Source in generale, in grandi aziende o Pubbliche Amministrazioni. Per questo l'accordo fra Fondazione e Canonical è un'ottima notizia. Il motivo per cui il problema è "quasi" risolto è che la nuova versione delle norme di Canonical ancora non copre i componenti di Ubuntu con licenza Open Source, ma diversa da quella Gpl di cui è custode la Free Software Foundation.